

STAPO: Selective Trajectory-Aware Policy Optimization for LLM Agent Training

Qiuyi Qi^{♠♦*}, Tian Liang^{♠♦*}, Mutian Bao^{♠♦*}, Jinjian Zhang[◇], Dongnan Liu[◇],
Wei Zhou[◇], Linjian Mo[◇], Ming Kong^{♠†}, Jie Liu^{♠†},
Feng Zhang[♠], Qiang Zhu^{♠†}

♠ Zhejiang University, ◇ Ant Group, ♣ City University of Hong Kong
{qiqiuyi, zhuq}@zju.edu.cn

Abstract

Reinforcement Learning (RL) is the dominant paradigm for training Large Language Model (LLM) agents on long-horizon tasks. However, sparse and delayed rewards often lead to *trajectory neglect*, in which agents lose focus on the task goal and interaction history at intermediate steps. Prior work has explored step-level supervision using Shannon-entropy-based uncertainty signals, which conflate inherent state complexity with agent confidence and therefore provide unreliable estimates of decision reliability. To address this issue, we propose *normalized entropy*, which measures confidence deviations relative to an agent’s average behavior under a given state, thereby strengthening the association between low-quality actions and trajectory neglect. Building on this insight, we introduce **Selective Trajectory-Aware Policy Optimization (STAPO)**, a hierarchical group-based RL framework. STAPO leverages normalized entropy to locate outlier steps associated with trajectory neglect and optimizes them via a joint mechanism of *trajectory-aware reward* and *trajectory-independent penalty*, enhancing trajectory awareness while preserving training stability. Extensive experiments on ALFWorld, WebShop, and Search-Augmented QA demonstrate that STAPO achieves state-of-the-art performance while substantially alleviating trajectory neglect, validating its effectiveness and robustness for agentic tasks.

1 Introduction

Large Language Models (LLMs) have evolved from static text generators to autonomous agents

*Q. Qi, T. Liang and M. Bao contributed equally to this work.

†Q. Zhu, M. Kong and J. Liu are corresponding authors. Q. Zhu is with the College of Artificial Intelligence, Shanghai Institute for Advanced Study, Zhejiang University. M. Kong is with the School of Earth Sciences, Zhejiang University. J. Liu is with the Department of Computer Science, City University of Hong Kong.

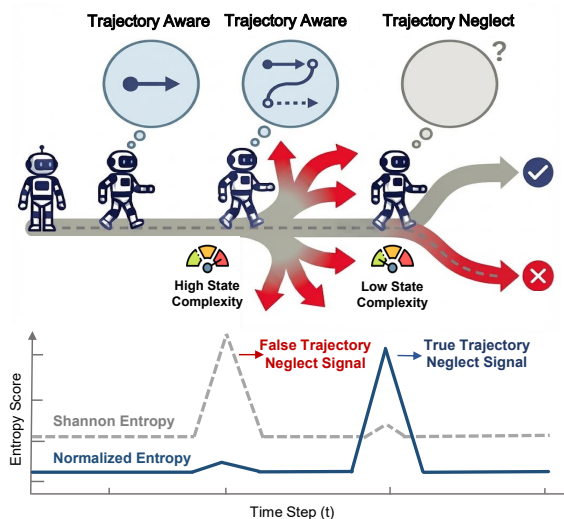


Figure 1: Illustration of the differences between *Shannon entropy* and *normalized entropy* in locating trajectory neglect.

capable of executing complex, long-horizon tasks, such as embodied control (Shridhar et al., 2021; Li et al., 2024) and web navigation (Furuta et al., 2024; Zheng et al., 2024; Gou et al., 2025).

To enable long-horizon reasoning in LLM agents, recent work has increasingly adopted Reinforcement Learning (RL) for post-training. In particular, group-based RL algorithms such as RLOO (Kool et al., 2019; Ahmadian et al., 2024) and GRPO (Shao et al., 2024) estimate advantages via group sampling, avoiding explicit critic training while improving scalability and training stability. However, training agents for long-horizon tasks remains challenging due to the *temporal credit assignment* problem induced by sparse and delayed rewards. As illustrated in Figure 1, agents often exhibit *trajectory neglect*, where the model loses focus on the task goal and interaction history during extended interaction sequences, leading to low-quality actions at certain intermediate steps (see Appendix A for detailed case studies).

While recent adaptations like GiGPO (Feng et al., 2025) and RLVMR (Zhang et al., 2025) attempt to mitigate the credit assignment issue by achieving fine-grained step-level advantage estimation, they typically optimize all steps indiscriminately, lacking a selective mechanism to precisely locate and address these outlier steps caused by trajectory neglect. Furthermore, several works have explored leveraging internal uncertainty signals to assist training, including entropy-based regularization and supervision strategies (Xu et al., 2025; Cui et al., 2025). However, these approaches predominantly rely on *Shannon entropy* computed from step-level action distributions, which conflates inherent state complexity with agent confidence. As shown in Figure 1, a high-entropy step may simply reflect a genuinely complex state with numerous plausible actions, rather than a lack of confidence. Consequently, raw Shannon entropy provides an unreliable signal for locating outlier steps associated with trajectory neglect (as empirically illustrated in Appendix B).

To address these limitations, we first propose a novel metric: *normalized entropy*. By aggregating actions from the same state across sampled trajectories, normalized entropy measures uncertainty relative to the state average, effectively decoupling inherent state complexity from model confidence. Our empirical analysis shows that outliers located by normalized entropy consistently correlate with trajectory neglect, which we empirically find to be strongly associated with low-quality actions.

Building on this insight, we introduce **Selective Trajectory-Aware Policy Optimization (STAPO)**, a hierarchical group-based RL framework. STAPO operates in two phases: (1) **Outlier Localization**, which dynamically locates outlier steps using normalized entropy statistics; and (2) **Selective Optimization**, which targets these steps associated with trajectory neglect. Specifically, STAPO incorporates a *trajectory-aware reward* to encourage stronger focus on trajectories, balanced by a *trajectory-independent penalty* to mitigate the risk of reward hacking.

Extensive experiments on representative agentic benchmarks, including ALFWorld (Shridhar et al., 2021), WebShop (Yao et al., 2022), and Search-Augmented QA, demonstrate that STAPO consistently outperforms strong baselines such as GiGPO (Feng et al., 2025) and EMPG (Wang et al., 2025a), while attribution analysis further confirms its effectiveness in alleviating trajectory neglect.

Our contributions are summarized as follows:

- We propose *normalized entropy* to decouple state complexity from agent confidence and effectively locate low-quality actions and trajectory neglect.
- We introduce *STAPO*, a hierarchical group-based RL framework that integrates outlier localization with a joint optimization mechanism of trajectory-aware reward and trajectory-independent penalty.
- Extensive experiments demonstrate that STAPO achieves superior performance and robustness across diverse agentic tasks, while substantially mitigating trajectory neglect.

2 Related Work

2.1 LLM Agents

Large Language Models (LLMs) have increasingly been developed into autonomous agents capable of perceiving, reasoning, and acting in dynamic environments (Zhang and Zhang, 2024; Wang et al., 2024a; Zitkovich et al., 2023). Early approaches primarily relied on prompt-based methods, such as ReAct (Yao et al., 2023) and Reflexion (Shinn et al., 2024), as well as enhanced memory, retrieval mechanisms (Wang et al., 2024b; Tan et al., 2024), and integration with external tools (Schick et al., 2023; Xie et al., 2024; Zhang et al., 2024). More recent work has shifted toward adapting model parameters through supervised fine-tuning (SFT) (Zhang and Zhang, 2024) or reinforcement learning (RL) (Sutton and Barto, 2018), enabling agents to learn directly from environment interactions.

2.2 Group-based Policy Optimization

To avoid the high overhead and instability associated with training independent critic models, group-based RL methods have emerged as promising alternatives to traditional actor-critic methods (like PPO (Schulman et al., 2017)). These algorithms, such as GRPO (Shao et al., 2024), RLOO (Kool et al., 2019; Ahmadian et al., 2024), and DAPO (Yu et al., 2025) leverage a group of samples from the same query and estimate the advantages accordingly. Recently, GiGPO (Feng et al., 2025), RLVMR (Zhang et al., 2025), and Tree-GRPO (Ji et al., 2025) further advance this paradigm by introducing hierarchical grouping structures to achieve fine-grained step-level advantage estimation without relying on auxiliary models. However, they

optimize all steps indiscriminately, lacking a selective mechanism to precisely locate and address outlier steps caused by trajectory neglect.

2.3 Entropy-based Reinforcement Learning

To overcome the challenges of sparse external rewards, recent works have explored using internal feedback (like model uncertainty, frequently quantified by entropy) to assist training (Agarwal et al., 2025). However, most of them rely on raw Shannon entropy, which conflates inherent state complexity with agent confidence, and primarily utilize it as a proxy for regularization or re-weighting. For instance, EPO (Xu et al., 2025) stabilizes exploration via entropy smoothing regularization; EMPG (Wang et al., 2025a) modulates the advantage term using entropy weights; and Cui et al. (2025) leverages entropy to truncate gradients for specific tokens. In contrast, we propose normalized entropy to decouple inherent state complexity from agent confidence, employing it to precisely locate outlier steps for selective trajectory-aware optimization.

3 Empirical Analysis

In this section, we formalize the task definitions, detail our statistical strategy for outlier localization based on a novel normalized entropy metric, and empirically validate its effectiveness in locating low-quality actions and trajectory neglect.

3.1 Task Definitions

Formally, we consider a general setting where an LLM agent interacts with an environment in discrete time steps $t = 1, 2, \dots, T$. The initial input instruction x consists of a **system prompt** p (defining general rules and roles) and a specific **task goal** g . At each step t , the agent receives a **current observation** s_t and maintains an **interaction history** $h_t = (s_1, a_1, \dots, s_{t-1}, a_{t-1})$. The complete prompt at step t is denoted as $c_t = (p, g, h_t, s_t)$, conditioned on which the model policy π_θ generates an action a_t .

To analyze agent behavior, we sample a group of N full trajectories $\{\tau_1, \tau_2, \dots, \tau_N\}$ for each instruction x . We define an *anchor state* \tilde{s} as the canonical state identity used for grouping semantically equivalent step contexts within the current batch (exact-match keys for structured observations and similarity-based keys for free-text observations). Within a single training epoch, each sampled step is assigned to *exactly one* anchor state via

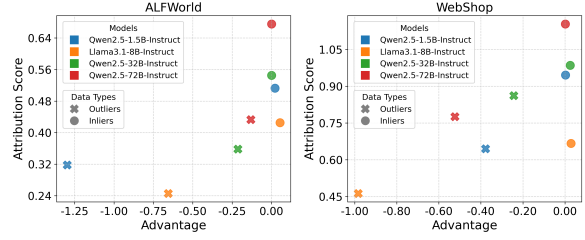


Figure 2: Empirical analysis on ALFWorld and WebShop. Markers denote the average step-level advantage and attribution score for outliers and inliers across various models.

hard clustering (exact match or greedy maximum-similarity assignment), so no step simultaneously belongs to multiple groups. We then utilize the *anchor state grouping* mechanism (Feng et al., 2025) to aggregate actions from the same state across sampled trajectories. Let $\mathcal{G}^S(\tilde{s}) = \{(a_t^{(i)}, R_t^{(i)}) \mid s_t^{(i)} = \tilde{s}, 1 \leq i \leq N, 1 \leq t \leq T\}$ denote the step-level group for an anchor state \tilde{s} , where $a_t^{(i)}$ is the action taken at step t in trajectory τ_i , and $R_t^{(i)}$ is the cumulative reward starting from step t (back-calculated from the outcome reward $R(\tau_i)$).

3.2 Outlier Localization

To precisely locate outliers in model confidence, we implement a statistical strategy as follows:

Normalized Entropy Calculation. We first formalize the step-level Shannon entropy H . For an action $a_t^{(i)}$ consisting of a sequence of M tokens (w_1, \dots, w_M) , $H(a_t^{(i)})$ is calculated as the average entropy over the generated tokens:

$$H(a_t^{(i)}) = -\frac{1}{M} \sum_{k=1}^M \sum_{v \in \mathcal{V}} p(v \mid c_t^{(i)}, w_{<k}) \log p(v \mid c_t^{(i)}, w_{<k}) \quad (1)$$

where $p(v \mid c_t^{(i)}, w_{<k})$ is the probability of token v from the vocabulary \mathcal{V} .

To decouple inherent state complexity from agent confidence, we propose normalized entropy H_n , which measures the deviation of H from the state-specific baseline. This metric facilitates unbiased comparisons of confidence across diverse states in multi-turn interactions. We perform intra-group normalization on H based on the statistics of the anchor state group $\mathcal{G}^S(\tilde{s})$:

$$H_n(a_t^{(i)}) = \frac{H(a_t^{(i)}) - \mu(\{H(a_t^{(j)}) \mid (a_t^{(j)}, \cdot) \in \mathcal{G}^S(\tilde{s})\})}{\sigma(\{H(a_t^{(j)}) \mid (a_t^{(j)}, \cdot) \in \mathcal{G}^S(\tilde{s})\}) + \epsilon} \quad (2)$$

where μ and σ denote the mean and standard deviation respectively, and ϵ is a small constant added for numerical stability.

Outlier Mask Generation. Using H_n , we employ the Interquartile Range (IQR) method to dynamically locate outliers without manual thresholding. Let $\mathcal{R} = [Q_1 - \lambda \cdot \text{IQR}, Q_3 + \lambda \cdot \text{IQR}]$ denote the inlier interval, where Q_1 and Q_3 are the first and third quartiles of H_n in the current training batch, $\text{IQR} = Q_3 - Q_1$, and λ is the IQR coefficient, typically set to 1.5. We then define an indicator mask $M_t \in \{0, 1\}$ for each step:

$$M_t(a_t^{(i)}) = \mathbb{I}(H_n(a_t^{(i)}) \notin \mathcal{R}) \quad (3)$$

Steps with $M_t = 1$ are flagged as outliers, while others are treated as inliers. Appendix C lists pseudocode for outlier localization.

3.3 Characterization of Outliers

To validate the effectiveness of H_n and investigate the nature of outliers, we conduct an empirical analysis on the test sets of ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022). We visualize the relationship between action quality (measured by step-level advantage, calculated via the same intra-group normalization on $R_t^{(i)}$) and trajectory focus (measured by attribution score, detailed in Appendix D) for both outliers and inliers. As shown in Figure 2:

- **Outliers indicate low-quality actions.** Outliers (marked with ‘ \times ’) are consistently clustered in the negative region of the Advantage axis. In contrast, inliers (marked with ‘ \bullet ’) are positioned significantly to the right, exhibiting higher advantages. These results indicate that outliers tend to correspond to low-quality actions.
- **Outliers indicate trajectory neglect.** On the Attribution Score axis, outliers consistently reside in the lower region compared to their corresponding inliers. This indicates that at outlier steps, the model’s focus on the trajectory is significantly weaker.

In summary, empirical results reveal that outliers located by normalized entropy serve as a reliable indicator of trajectory neglect, which is a primary cause of low-quality actions in long-horizon tasks.

4 Methodology

To address the challenge of trajectory neglect identified in Section 3, we propose Selective Trajectory-Aware Policy Optimization (STAPO), a hierarchical group-based RL framework.

4.1 Outlier Localization Integration

As illustrated in the left panel of Figure 3, STAPO integrates the localization mechanism validated in Section 3.2 directly into the training loop. Specifically, for each step, we calculate the normalized entropy H_n (Equation 2) and derive the indicator mask M_t (Equation 3). This mask functions as a dynamic gating mechanism: it filters the training stream so that only steps flagged as outliers ($M_t = 1$) trigger the selective optimization module, while inliers ($M_t = 0$) are optimized solely using the base objective.

4.2 Selective Optimization

For the located outlier steps, we introduce a joint optimization mechanism, depicted in the center panel of Figure 3, composed of a trajectory-aware reward and a trajectory-independent penalty.

Trajectory-Aware Reward. To encourage the model to focus on the trajectory when responding, we propose a trajectory-aware reward. We define a masking operation $(\cdot)_{\setminus \mathcal{T}}$ that strips the goal and history from the complete prompt $c_t^{(i)}$, yielding a trajectory-blind prompt $(c_t^{(i)})_{\setminus \mathcal{T}} = (p, \emptyset, \emptyset, s_t^{(i)})$. We then calculate the KL divergence between the model’s output log-probabilities under these two conditions (denoted as π_θ and π_θ^{mask} in Figure 3):

$$R_{TA}(c_t^{(i)}, a_t^{(i)}) = \mathbb{D}_{KL}[\pi_\theta(a_t^{(i)} | c_t^{(i)}) \parallel \pi_\theta(a_t^{(i)} | (c_t^{(i)})_{\setminus \mathcal{T}})] \quad (4)$$

Conceptually, R_{TA} measures how much the agent’s action distribution depends on the full trajectory context. A higher R_{TA} implies that the agent is actively leveraging the trajectory information to shape its output distribution.

Trajectory-Independent Penalty. To mitigate the risk of reward hacking on R_{TA} at outlier steps, we introduce a trajectory-independent penalty. Unlike the standard global KL penalty which operates on the complete prompt $c_t^{(i)}$, this penalty specifically constrains the model’s behavior on the masked prompt $(c_t^{(i)})_{\setminus \mathcal{T}}$ to remain close to the reference policy π_{ref} (denoted as π_θ^{mask} and π_{ref}^{mask}

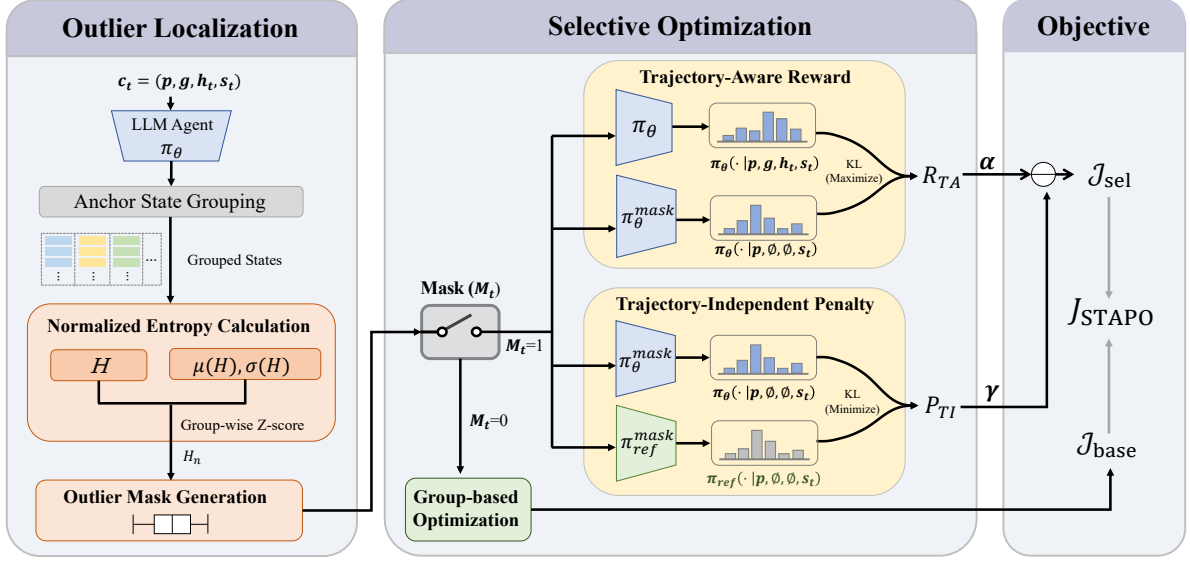


Figure 3: The framework of STAPO. It dynamically locates outlier steps utilizing normalized entropy statistics and optimizes them via a joint mechanism of trajectory-aware reward and trajectory-independent penalty.

in Figure 3):

$$P_{TI}(c_t^{(i)}, a_t^{(i)}) = \mathbb{D}_{KL}[\pi_\theta(a_t^{(i)} | (c_t^{(i)}) \setminus \tau) \parallel \pi_{ref}(a_t^{(i)} | (c_t^{(i)}) \setminus \tau)] \quad (5)$$

This constraint ensures that the increase in R_{TA} stems from the model’s enhanced focus on the trajectory, rather than from mode collapse.

4.3 STAPO Objective

As illustrated in the right panel of Figure 3, the STAPO framework integrates these selective terms into a group-based RL objective. Inspired by GiGPO (Feng et al., 2025), we adopt a two-level advantage formulation:

Advantage Estimation. First, the episode-level advantage A^E evaluates the overall trajectory quality. For trajectory τ_i , it is calculated via intra-group normalization of outcome rewards:

$$A^E(\tau_i) = \frac{R(\tau_i) - \mu(\{R(\tau_j)\}_{j=1}^N)}{\sigma(\{R(\tau_j)\}_{j=1}^N) + \epsilon} \quad (6)$$

Second, the step-level advantage A^S provides fine-grained credit assignment. Utilizing the anchor state group $\mathcal{G}^S(\tilde{s})$ defined in Section 3.1, we calculate the advantage of action $a_t^{(i)}$ relative to the average performance in that state:

$$A^S(a_t^{(i)}) = \frac{R_t^{(i)} - \mu(\{R_t^{(j)} \mid (a_t^{(j)}, R_t^{(j)}) \in \mathcal{G}^S(\tilde{s})\})}{\sigma(\{R_t^{(j)} \mid (a_t^{(j)}, R_t^{(j)}) \in \mathcal{G}^S(\tilde{s})\}) + \epsilon} \quad (7)$$

The overall advantage A combines these two terms: $A(a_t^{(i)}) = A^E(\tau_i) + \omega A^S(a_t^{(i)})$, where ω is a weighting coefficient.

Optimization Objective. Incorporating the estimated advantage and the selective optimization terms, the final clipped policy optimization objective for STAPO is:

$$J_{STAPO}(\theta) = \mathbb{E}_{\substack{x \sim p(X) \\ \{\tau_i\}_{i=1}^N \sim \pi_{\theta_{old}}}} \left[\frac{1}{NT} \sum_{i=1}^N \sum_{t=1}^T (\mathcal{J}_{base}^{(i,t)} + \mathcal{J}_{sel}^{(i,t)}) \right] \quad (8)$$

where $p(X)$ is the task distribution and $\rho_\theta(a_t^{(i)})$ is the importance sampling ratio. The two components are defined as:

$$\mathcal{J}_{base}^{(i,t)} = \min[\rho_\theta(a_t^{(i)})A(a_t^{(i)}), \text{clip}(\rho_\theta(a_t^{(i)}), 1 \pm \epsilon)A(a_t^{(i)})] - \beta \mathbb{D}_{KL}[\pi_\theta \parallel \pi_{ref}] \quad (9)$$

$$\mathcal{J}_{sel}^{(i,t)} = M_t \cdot (\alpha R_{TA}(c_t^{(i)}, a_t^{(i)}) - \gamma P_{TI}(c_t^{(i)}, a_t^{(i)})) \quad (10)$$

where $\mathcal{J}_{base}^{(i,t)}$ represents the base group-based RL objective, $\mathcal{J}_{sel}^{(i,t)}$ denotes our proposed selective component, and α, β, γ are weighting coefficients.

5 Experiments

5.1 Experiment Setup

Benchmarks. We first train the LLM agents on two challenging benchmarks: ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022). ALFWorld is an embodied environment designed to assess the ability of LLM agents to perform multi-step decision-making. In each episode, the agent receives a text goal and must accomplish it through multi-turn interaction with the environment. It includes 3,827 task instances across six categories of common household activities: Pick & Place (Pick), Examine in Light (Look), Clean & Place (Clean),

Type	Method	ALFWorld							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
<i>Closed-Source Model</i>										
Prompting	GPT-4o	75.3	60.8	31.2	56.7	21.6	49.8	48.0	31.8	23.7
Prompting	Gemini-2.5-Pro	92.8	63.3	62.1	69.0	26.6	58.7	60.3	42.5	35.9
<i>Qwen2.5-1.5B-Instruct</i>										
Prompting	Qwen2.5	5.9	5.5	3.3	9.7	4.2	0.0	4.1	23.1	5.2
Prompting	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8	40.1	11.3
Prompting	Reflexion	35.3	22.2	21.7	13.6	19.4	3.7	21.8	55.8	21.9
Outcome RL	PPO	64.8 \pm 3.5	40.5 \pm 6.9	57.1 \pm 4.9	60.6 \pm 6.6	46.4 \pm 4.0	47.4 \pm 1.9	54.4 \pm 3.1	73.8 \pm 3.0	51.5 \pm 2.9
Outcome RL	RLOO	88.3 \pm 3.0	52.8 \pm 8.6	71.0 \pm 5.9	62.8 \pm 8.7	66.4 \pm 5.5	56.9 \pm 4.7	69.7 \pm 2.5	73.9 \pm 5.6	52.1 \pm 6.7
Outcome RL	GRPO	85.3 \pm 1.5	53.7 \pm 8.0	84.5 \pm 6.8	78.2 \pm 7.9	59.7 \pm 5.0	53.5 \pm 5.6	72.8 \pm 3.6	75.8 \pm 3.5	56.8 \pm 3.8
Process RL	EMPG	85.5 \pm 4.8	33.5 \pm 6.4	78.9 \pm 2.5	76.2 \pm 9.7	74.7 \pm 1.9	69.1 \pm 6.4	73.7 \pm 2.7	80.4 \pm 0.7	60.8 \pm 1.3
Process RL	GiGPO	<u>94.4\pm5.9</u>	<u>67.5\pm4.6</u>	94.8\pm3.8	94.4\pm7.8	<u>79.8\pm4.7</u>	<u>76.4\pm5.4</u>	<u>86.7\pm1.7</u>	83.1 \pm 1.6	<u>65.0\pm3.2</u>
Process RL	STAPO	97.7\pm4.0	77.8\pm4.8	<u>93.4\pm0.5</u>	<u>90.2\pm9.4</u>	85.6\pm2.6	95.1\pm0.4	92.2\pm1.2	85.9\pm0.6	69.0\pm2.5
<i>Qwen2.5-7B-Instruct</i>										
Prompting	Qwen2.5	33.4	21.6	19.3	6.9	2.8	3.2	14.8	26.4	7.8
Prompting	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
Prompting	Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.7	58.1	28.8
Outcome RL	PPO	92.3 \pm 4.0	64.0 \pm 8.4	92.5 \pm 2.4	<u>89.5\pm7.0</u>	80.3 \pm 2.0	68.8 \pm 8.3	80.4 \pm 2.7	81.4 \pm 3.1	68.7 \pm 5.1
Outcome RL	RLOO	87.6 \pm 4.3	78.2 \pm 8.3	87.3 \pm 5.8	81.3 \pm 7.6	71.9 \pm 5.2	48.9 \pm 8.4	75.5 \pm 4.6	80.3 \pm 3.2	65.7 \pm 4.0
Outcome RL	GRPO	90.8 \pm 5.1	66.1 \pm 6.7	89.3 \pm 5.4	74.7 \pm 6.9	72.5 \pm 5.4	64.7 \pm 7.3	77.6 \pm 5.2	79.3 \pm 2.8	66.1 \pm 3.7
Process RL	EMPG	92.9 \pm 2.9	75.2 \pm 3.8	74.8 \pm 3.9	86.3 \pm 4.7	73.7 \pm 2.6	65.3 \pm 5.8	78.5 \pm 1.7	81.0 \pm 1.4	69.3 \pm 0.5
Process RL	GiGPO	<u>97.7\pm1.6</u>	<u>82.7\pm7.9</u>	98.8\pm1.6	83.7 \pm 7.2	<u>89.3\pm8.2</u>	<u>79.2\pm6.6</u>	<u>90.8\pm1.3</u>	84.4 \pm 2.9	<u>72.8\pm3.2</u>
Process RL	STAPO	97.8\pm3.9	100.0\pm0.0	98.8\pm2.1	89.9\pm7.5	92.7\pm5.2	100.0\pm0.0	96.9\pm1.6	89.1\pm1.3	77.6\pm3.9

Table 1: Performance on ALFWorld and WebShop. Results are averaged over 3 random seeds. For ALFWorld, we report the average success rate (%) for each subtask as well as the overall result. For WebShop, we report both the average score and the average success rate (%). The best and second-best results are **bold** and underlined. Methods highlighted in gray are our implemented results. The remaining results are adopted from GiGPO (Feng et al., 2025) and EMPG (Wang et al., 2025a).

Heat & Place (Heat), Cool & Place (Cool), and Pick Two & Place (Pick2). *WebShop* is a complex, web-based interactive environment designed to test the LLM agents in realistic online shopping scenarios. To complete the task, the agent must interact with a simulated HTML-based shopping website to search for, navigate to, and ultimately purchase a suitable item. It contains over 1.1 million products and 12k user instructions, providing a rich and diverse action space. In addition, we also evaluate the multi-turn tool calling performance of STAPO on *search-augmented QA tasks*, including single-hop QA datasets (NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2022)) and multi-hop QA datasets (HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2022)).

Baselines. For ALFWorld and WebShop, we compare our approach with a range of competitive baselines under the same evaluation protocol: (1) Closed-source LLMs: GPT-4o (Achiam et al., 2023) and Gemini-2.5-Pro (Team et al., 2023), which represent state-of-the-art capabilities in general-purpose reasoning and language understanding. (2) Prompting agents: ReAct (Yao

et al., 2023) and Reflexion (Shinn et al., 2024), which rely on in-context prompting to guide multi-step behavior without parameter updates. (3) Outcome-supervised RL: Classical algorithms optimized solely on sparse outcome rewards, including PPO (Schulman et al., 2017), RLOO (Kool et al., 2019; Ahmadian et al., 2024), and GRPO (Shao et al., 2024). (4) Process-supervised RL: Recent works utilizing process signals to assist RL training, including EMPG (Wang et al., 2025a) and GiGPO (Feng et al., 2025). For search-augmented QA, we compare STAPO with R1-Instruct, Search-R1 (Jin et al., 2025), ZeroSearch (Sun et al., 2025), and StepSearch (Wang et al., 2025b) beyond GiGPO.

Implementation Details. We use the Qwen2.5 series and Llama3.1-8B-Instruct as our base models. The weighting coefficient ω is set to 1 with no further tuning¹. For ALFWorld and WebShop, we employ exact matching (via hashmaps) for anchor state grouping, set the rollout group size N to 8, and the max turns to 50 and 15, respectively. For search-augmented QA, we follow the same settings in Search-R1 (Jin et al., 2025) and use E5 (Wang et al., 2022) as the retriever. We adopt similarity-

¹Consistent with the setting in GiGPO.

Type	Method	Single-Hop QA			Multi-Hop QA				Avg.
		NQ [†]	TriviaQA [*]	PopQA [*]	HotpotQA [†]	2Wiki [*]	MuSiQue [*]	Bamboogle [*]	
<i>Qwen2.5-3B-Instruct</i>									
Outcome RL	R1-Instruct	21.0	44.9	17.1	20.8	27.5	6.0	19.2	22.4
Outcome RL	Search-R1	34.1	54.5	37.8	32.4	31.9	10.3	26.4	32.5
Outcome RL	ZeroSearch	41.4	57.4	<u>44.8</u>	27.4	30.0	9.8	11.1	31.7
Process RL	StepSearch	-	-	-	34.5	32.0	17.4	34.4	-
Process RL	GiGPO	<u>42.0</u>	<u>59.5</u>	42.4	36.9	37.0	12.6	<u>64.1</u>	<u>42.1</u>
Process RL	STAPO	43.1	60.7	45.8	36.9	<u>36.4</u>	<u>13.0</u>	64.9	43.0
<i>Qwen2.5-7B-Instruct</i>									
Outcome RL	R1-Instruct	27.0	53.7	19.9	23.7	29.2	7.2	29.3	27.1
Outcome RL	Search-R1	39.3	61.0	39.7	37.0	40.1	14.6	36.8	38.5
Outcome RL	ZeroSearch	43.6	61.8	51.5	34.6	35.2	18.4	27.8	39.1
Process RL	StepSearch	-	-	-	38.6	36.6	22.6	40.0	-
Process RL	GiGPO	46.4	<u>64.7</u>	46.1	41.6	43.6	<u>18.9</u>	<u>68.9</u>	<u>47.2</u>
Process RL	STAPO	48.8	66.0	<u>48.7</u>	43.0	45.0	18.1	69.4	48.4

Table 2: Performance on search-augmented QA tasks. STAPO is trained on NQ and HotpotQA. † and * indicate in-domain and out-of-domain datasets, respectively. The best and second-best results are **bold** and underlined. Methods highlighted in gray are our implemented results. The remaining results are adopted from GiGPO (Feng et al., 2025).

Variant	ALFWorld	WebShop
<i>Localization Strategy</i>		
w/ Shannon Entropy	89.6	67.2
<i>Masking Strategy</i>		
w/ None	86.7	65.0
w/ Goal Only	89.0	66.6
w/ History Only	90.9	68.2
<i>Penalty Strategy</i>		
w/ Single Penalty	88.3	65.9
w/ Inc Penalty Coef	84.6	67.0
w/ STAPO	92.2	69.0

Table 3: Ablation study on ALFWorld and WebShop. The results represent success rate (%).

based anchor state grouping (longest matching subsequence with a threshold of 0.9) following GiGPO. The rollout group size N is set to 5 and the max turn is set to 4. All training is conducted on a single node with 8 A100 GPUs. Unless otherwise specified, the IQR coefficient λ is set to 1.5 (adopted from Tukey’s fences, the standard statistical convention for defining IQR outliers), and the weighting coefficients α , β , and γ (Equations 9–10) are kept equal. Full training settings and hyperparameter details are provided in Appendix E.

5.2 Results

Performance on ALFWorld and WebShop. Table 1 demonstrates the strong performance of STAPO on both ALFWorld and WebShop, where it consistently and significantly outperforms all baselines across different base model scales. Compared

to the state-of-the-art method GiGPO, STAPO achieves superior results on ALFWorld, reaching near-perfect performance (96.9%) with Qwen2.5-7B-Instruct. On WebShop, STAPO also shows a clear advantage, improving the success rate from 65.0% (GiGPO) to 69.0% with Qwen2.5-1.5B-Instruct. These results suggest that precise localization and optimization of outlier steps effectively enhance agent performance in long-horizon tasks. See Appendix F for further experiments on STAPO’s generalization to other group-based RL methods (e.g., DAPO) and its adaptation across different model architectures.

Performance on Search-Augmented QA Tasks.

As shown in Table 2, STAPO achieves consistent gains on multi-turn search-augmented QA tasks. While it outperforms strong baselines on average, the improvement over GiGPO ($\approx 1\%$) is more modest than on ALFWorld and WebShop. We attribute this to the shorter interaction horizons of search-augmented QA tasks, which naturally mitigate trajectory neglect and thus limit the gains from STAPO’s selective optimization. Table 6 in Appendix G reports robustness of QA performance to the similarity threshold used for anchor state grouping.

5.3 Ablation Study

We validate the effectiveness of STAPO through ablation studies on localization, masking, and penalty strategies using Qwen2.5-1.5B-Instruct, with a sensitivity analysis of the weighting coefficients (α

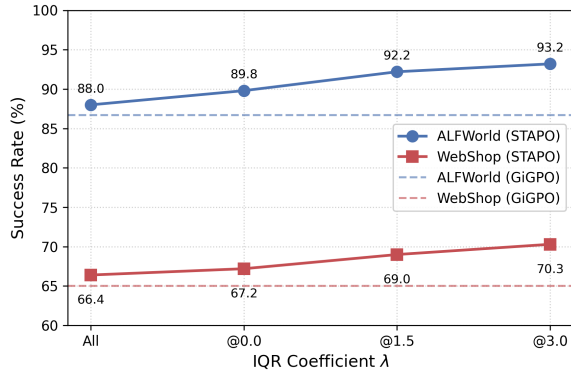


Figure 4: Ablation study of localization sensitivity. “All” means non-selective.

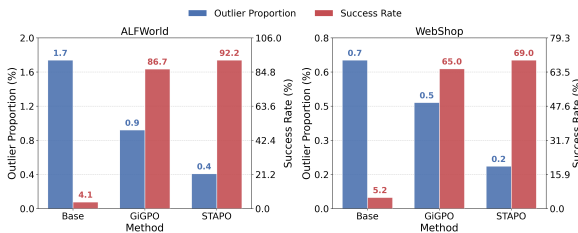


Figure 5: Attribution analysis on ALFWorld and WebShop. The blue bars (left axis) represent the proportion of outlier steps, while the red bars (right axis) indicate the success rate.

and γ) provided in Appendix H.

Localization Strategy. We evaluate the proposed outlier localization mechanism by examining the uncertainty metric choice and the sensitivity of the localization criterion.

- **Entropy Type.** As shown in the top block of Table 3, using normalized entropy for localization consistently outperforms raw Shannon entropy. This result indicates that normalized entropy more effectively decouples state complexity from agent uncertainty, enabling more accurate identification of true outlier steps, whereas raw Shannon entropy is more susceptible to state-dependent noise.
- **Localization Sensitivity.** Figure 4 shows the effect of the IQR coefficient λ controlling outlier strictness. STAPO consistently outperforms GiGPO across all settings, including “All”, and performance improves as λ increases from 0.0 to 3.0, suggesting that selectively optimizing a smaller set of confidently located outlier steps is beneficial.

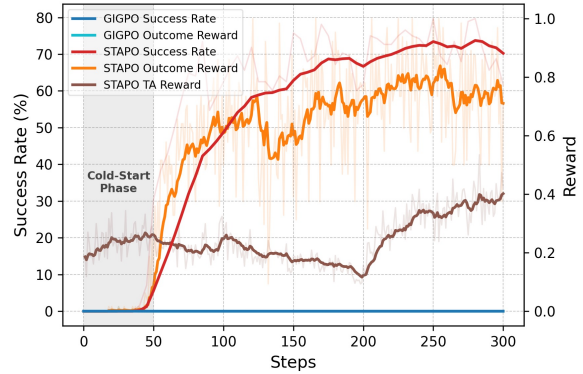


Figure 6: Training dynamics on WebShop using LLaMA-3.1-8B-Instruct. Solid lines denote exponential moving averages. “TA” refers to *Trajectory-Aware*.

Masking Strategy. We compare different input masking strategies for reward calculation, as shown in the middle block of Table 3. Masking both the goal and interaction history (STAPO) consistently achieves the best performance. Notably, “w/ History Only” outperforms “w/ Goal Only”, and both surpass the “w/ None” baseline (GiGPO), indicating that historical context is more critical than goal information alone for long-horizon decision-making. The further improvement from combining both signals highlights the importance of jointly reinforcing goal awareness and trajectory consistency to mitigate trajectory neglect.

Penalty Strategy. The bottom block of Table 3 examines the effect of the trajectory-independent penalty. Compared to employing a standard global KL regularization (“w/ Single Reg”), simply increasing its weight (“w/ Inc Reg Coef”) can lead to performance degradation (e.g., 88.3% \rightarrow 84.6% on ALFWorld), indicating that overly strong global constraints may hinder effective learning. In contrast, STAPO achieves the best overall results by applying the penalty selectively at outlier steps. This design encourages anchor policy updates to the reference policy at steps prone to trajectory neglect, while avoiding unnecessary constraints on well-behaved steps. As a result, the increase in R_{TA} reliably enhanced trajectory focus rather than unintended effects such as mode collapse or overly conservative policy updates.

5.4 Attribution Analysis

To gain deeper insights into STAPO’s mechanism, we statistically analyze the proportion of outlier steps generated by the Qwen2.5-1.5B-Instruct model on the test sets before and after training. As

shown in Figure 5, the model’s performance exhibits a negative correlation with the proportion of outliers. Compared to GiGPO, STAPO not only achieves better results but also yields trajectories with a significantly lower proportion of outliers. This result demonstrates that STAPO enhances performance fundamentally by mitigating low-quality actions caused by trajectory neglect. Refer to Appendix I for an analysis of the internal composition (high vs. low entropy) of these outliers. Appendix K further reports test-time KL divergence between full and trajectory-blind prompts together with average trajectory attribution scores, as complementary evidence of increased reliance on trajectory.

5.5 Exploration Guidance

STAPO demonstrates superior exploration capability, particularly in cold-start scenarios. In challenging settings such as training Llama3.1-8B-Instruct on WebShop, the weak foundational policy fails to obtain any sparse outcome rewards, causing RL baselines like GiGPO to fail in optimizing effectively (as shown by the flatlined zero success rate in Figure 6). Conversely, STAPO’s trajectory-aware reward R_{TA} provides continuous supervisory signals. Especially in the initial phase (e.g., steps 0-50) where external outcome rewards are absent, R_{TA} offers non-zero guidance, effectively steering the model towards meaningful actions and breaking the exploration deadlock.

6 Conclusion

In this work, we identify trajectory neglect as a primary cause of low-quality actions in long-horizon tasks and propose normalized entropy to precisely locate it by decoupling state complexity from agent confidence. Building on this, we introduce STAPO, a framework that selectively optimizes outliers via a joint mechanism of trajectory-aware reward and trajectory-independent penalty. Extensive experiments demonstrate that STAPO achieves superior performance and robustness across diverse benchmarks, providing a strong and competitive approach for trajectory-aware LLM agent training.

Limitations

Regarding training efficiency, calculating the trajectory-aware reward and trajectory-independent penalty requires constructing masked inputs and performing additional forward passes. However,

as detailed in Appendix J, this computational overhead is marginal ($\sim 5.7\%$) and proves significantly more cost-effective than scaling the group size.

Our main experiments focus on text-based agents. Appendix L provides additional pilot results on vision-language agents, but broader multimodal settings and outlier behavior under diverse visual interfaces remain to be fully characterized.

Normalized entropy assumes sufficient within-batch support for each anchor state: when a state is visited only once in a batch, intra-group normalization is ill-defined and the statistic effectively reduces to raw Shannon entropy, which again conflates state complexity with confidence. Environments with very large admissible action spaces make state revisitation rarer and can exacerbate this boundary behavior, occasionally yielding false positives or false negatives in outlier localization.

Ethical considerations

In this work, we introduce a framework to enhance the performance of LLM agents in long-horizon tasks. While improving agent capabilities is crucial for beneficial applications such as digital assistants and automated web navigation, we acknowledge that more capable agents could potentially be exploited for malicious purposes, such as automated cyber-attacks or large-scale disinformation campaigns. However, our research is conducted strictly using established, public academic benchmarks (ALFWorld, WebShop, and QA datasets) that do not contain personally identifiable information (PII) or offensive content.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant 42394060 and 42394064, Ant Group Research Fund, and Information Technology Center and State Key Lab of CAD&CG, Zhejiang University.

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. 2025. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*.

- Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting reinforce style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267.
- Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, and 1 others. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.
- Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*.
- Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. 2024. Multimodal web navigation with instruction-finetuned foundation models. In *The Twelfth International Conference on Learning Representations*.
- Boyuan Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2025. Navigating the digital world as humans do: Universal visual grounding for GUI agents. In *The Thirteenth International Conference on Learning Representations*.
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.
- Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xi-angxiang Chu, and Liaoni Wu. 2025. Tree search for llm agent reinforcement learning. *arXiv preprint arXiv:2509.21240*.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.
- Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.
- Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and 1 others. 2020. Captum: A unified and generic model interpretability library for pytorch. *arxiv. arXiv preprint arXiv:2009.07896*, 2:5.
- Wouter Kool, Herke van Hoof, and Max Welling. 2019. Buy 4 reinforce samples, get a baseline for free! In *ICLR 2019 Workshop*.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.
- Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, and 1 others. 2024. Embodied agent interface: Benchmarking LLMs for embodied decision making. *Advances in Neural Information Processing Systems*, 37:100428–100534.
- Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khachabi, and Hannaneh Hajishirzi. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.
- Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. DeepSeek-Math: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.
- Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.
- Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Fei Huang, and Yan Zhang. 2025. ZeroSearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*.
- Richard S Sutton and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction*. MIT press.

- Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Gang Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, and 1 others. 2024. Cradle: Empowering foundation agents towards general computer control. In *NeurIPS 2024 Workshop on Open-World Agents*.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024a. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*.
- Jiawei Wang, Jiakai Liu, Yuqian Fu, Yingru Li, Xintao Wang, Yuan Lin, Yu Yue, Lin Zhang, Yang Wang, and Ke Wang. 2025a. Harnessing uncertainty: Entropy-modulated policy gradients for long-horizon llm agents. *arXiv preprint arXiv:2509.09265*.
- Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024b. Mobile-Agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *Advances in Neural Information Processing Systems*, 37:2686–2710.
- Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.
- Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. 2025b. StepSearch: Igniting LLMs search ability via step-wise proximal policy optimization. *arXiv preprint arXiv:2505.15107*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.
- Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. 2024. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Wujiang Xu, Wentian Zhao, Zhenting Wang, Yu-Jhe Li, Can Jin, Mingyu Jin, Kai Mei, Kun Wan, and Dimitris N Metaxas. 2025. Epo: Entropy-regularized policy optimization for llm agents reinforcement learning. *arXiv preprint arXiv:2509.22576*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. WebShop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757.
- 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*.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, and 1 others. 2025. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.
- Chaoyun Zhang, Liquan Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, and 1 others. 2024. UFO: A UI-focused agent for windows OS interaction. *arXiv preprint arXiv:2402.07939*.
- Zhuosheng Zhang and Aston Zhang. 2024. You only look at screens: Multimodal chain-of-action agents. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 3132–3149.
- Zijing Zhang, Ziyang Chen, Mingxiao Li, Zhaopeng Tu, and Xiaolong Li. 2025. Rlvmr: Reinforcement learning with verifiable meta-reasoning rewards for robust long-horizon agents. *arXiv preprint arXiv:2507.22844*.
- Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. GPT-4V (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*.
- Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, and 1 others. 2023. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR.

A Case Study of Trajectory Neglect in Multi-Turn Interactions

To motivate the necessity of trajectory-aware optimization, we analyze a failed trajectory of a Qwen2.5-1.5B-Instruct agent trained with GiGPO from ALFWorld. The task goal is “*look at cd under the desk lamp.*” Throughout this episode, the agent exhibits two primary failure modes correlated with trajectory neglect: (1) Task Goal Neglect, where the agent wanders to irrelevant locations (e.g., the bed) despite the goal explicitly constraining the context to a desk lamp, and (2) Interaction History Neglect, resulting in cyclic and redundant actions.

Case: Neglect of Task Goal. As shown in Figure 8, during Steps 21–25, the agent successfully acquires cd 1. However, instead of attempting to place it under a desk lamp, the agent begins examining the object and wanders to bed 1 (Step 23, highlighted with an underline), a location irrelevant to the desk lamp. This indicates a degradation of the goal signal over long horizons.

Case: Neglect of Interaction History. As shown in Figure 9, during Steps 34–38, the agent demonstrates a severe inability to track its recent actions. After identifying cd 2 on shelf 1, the agent performs the action `look at cd 2`. In the very next step, despite the observation remaining static, the agent repeats the exact same action (Step 35, highlighted with an underline). Furthermore, the agent later returns to desk 2, a location it had already visited in Step 25. This suggests that the agent is not effectively tracking its interaction history.

B Case Study: Normalized vs. Shannon Entropy

To empirically validate the effectiveness of normalized entropy in decoupling state complexity from model confidence, we present qualitative examples from the ALFWorld evaluation using the Qwen2.5-1.5B-Instruct base model. These cases demonstrate how normalized entropy avoids false positives in states with a large action space and reduces false negatives in states with a small action space, thereby providing a more reliable signal for outlier localization.

Case: Avoiding False Positives in States with a Large Action Space. As shown in Figure 10, at Step 27, the agent is positioned in a kitchen environment with a vast action space (32 selectable

actions). The agent chooses to navigate to “countertop 1”, a valid exploration step denoted by a positive advantage. However, due to the sheer number of admissible actions, the raw probability distribution is naturally dispersed, resulting in a high Shannon entropy. A standard threshold-based method (e.g., IQR) would incorrectly flag this valid step as an outlier (a “false positive”). In contrast, normalized entropy calibrates this value against the group statistics of the same state. It correctly recognizes that high entropy is intrinsic to this specific state and classifies the step as an inlier, preventing the unnecessary penalization of valid exploration.

Case: Reducing False Negatives in States with a Small Action Space. As shown in Figure 11, at Step 43, the agent operates in a restricted context with a limited action space (16 selectable actions). Here, the agent exhibits suboptimal behavior by closing an empty cabinet it has already inspected (interaction history neglect), a redundant action indicated by a negative advantage. Despite this low-quality decision, the limited number of choices results in a naturally sharper probability distribution and a relatively low Shannon entropy. Consequently, a standard threshold-based method (e.g., IQR) would likely miss this error (a “false negative”). However, normalized entropy detects that the agent’s raw Shannon entropy—while numerically low—is significantly higher than the baseline expected for this specific state. This anomaly correctly flags the step as an outlier, triggering the STAPO to optimize the behavior.

C Pseudocode for Outlier Localization

Algorithm 1 summarizes anchor state formation, normalized entropy calculation, and outlier mask generation.

D Details for Empirical Analysis

In this paper, we adopt Feature Ablation (implemented via FeatureAblation in CapTum (Kokhlikyan et al., 2020)) as our method for calculating trajectory attribution scores for LLMs. For ALFWorld and WebShop, we randomly select 128 samples from the test set and conduct empirical analysis on the trajectories generated by 8 rollouts per sample.

Algorithm 1 Outlier localization

Require: Batch of steps with prompts $\{c_t^{(i)}\}$ and actions $\{a_t^{(i)}\}$, grouping rule $\text{ANCHORKEY}(\cdot)$, stability ϵ , Tukey coefficient λ

- 1: Initialize empty groups $\{\mathcal{G}^S(\tilde{s})\}$
- 2: **for** each step (i, t) in the batch **do**
- 3: $\tilde{s} \leftarrow \text{ANCHORKEY}(c_t^{(i)})$
- 4: Append $(a_t^{(i)}, R_t^{(i)}, c_t^{(i)})$ to $\mathcal{G}^S(\tilde{s})$
- 5: **end for**
- 6: **for** each group $\mathcal{G}^S(\tilde{s})$ **do**
- 7: **for** each $(a_t^{(i)}, R_t^{(i)}, c_t^{(i)}) \in \mathcal{G}^S(\tilde{s})$ **do**
- 8: Compute $H(a_t^{(i)})$ using Equation 1
- 9: **end for**
- 10: Compute $\mu_{\tilde{s}}, \sigma_{\tilde{s}}$ over $\{H(a_t^{(j)})\}$ in $\mathcal{G}^S(\tilde{s})$
- 11: **for** each $(a_t^{(i)}, \dots) \in \mathcal{G}^S(\tilde{s})$ **do**
- 12: $H_n(a_t^{(i)}) \leftarrow (H(a_t^{(i)}) - \mu_{\tilde{s}}) / (\sigma_{\tilde{s}} + \epsilon)$
 \triangleright Equation 2
- 13: **end for**
- 14: **end for**
- 15: Let \mathcal{H}_n be the multiset of all H_n values in the batch
- 16: Compute quartiles Q_1, Q_3 of \mathcal{H}_n ; $\text{IQR} \leftarrow Q_3 - Q_1$
- 17: **for** each step (i, t) **do**
- 18: $M_t(a_t^{(i)}) \leftarrow \mathbb{I}(H_n \notin [Q_1 - \lambda \cdot \text{IQR}, Q_3 + \lambda \cdot \text{IQR}])$
 \triangleright Equation 3
- 19: **end for**

E Experiment Details

E.1 Details of Training

Hyperparameters for ALFWorld. The maximum prompt length is 2048 tokens, and the maximum response length is 512 tokens. Each episode allows up to 50 environment steps. The learning rate is set to 1e-6 for the actor. We adopt a rule-based reward, assigning a reward of 1 for success and 0 for failure. To handle invalid actions generated by the agent, we apply a reward penalty of -0.1. We use a group size of 8 and sample 16 different groups per rollout, resulting in a total of $16 \times 8 = 128$ environments. The rollout temperature is set to 1.0, while the validation temperature is set to 0.4. The mini-batch size is 256, and the weighting coefficient ω is fixed at 1 without further tuning. The IQR coefficient λ is set to 1.5, and the weighting coefficients α, β , and γ (Equations 9–10) are all set to 0.01.

Hyperparameters for WebShop. The maximum prompt length is 4096 tokens, and the maximum response length is 512 tokens. Each episode is limited to 15 environment steps. The learning rate is 1e-6 for the actor. We adopt a rule-based reward, assigning a reward of 1 for success and 0 for failure. Invalid actions are penalized with a reward of -0.1. We use a group size of 8 and sample 8 groups per rollout, totaling $8 \times 8 = 64$ environments. The rollout temperature is set to 1.0, while the validation temperature is set to 0.4. The mini-batch size is 64, and the weighting coefficient ω is fixed at 1 without further tuning. The IQR coefficient λ is set to 1.5, and the weighting coefficients α, β , and γ (Equations 9–10) are all set to 0.01.

Hyperparameters for Search-Augmented QA.

The maximum prompt length is 4096 tokens, and the maximum response length is 512 tokens. The max turn is set to 4. The learning rate is 1e-6 for the actor. We adopt a rule-based reward, assigning a reward of 1 for success and 0 for failure. Invalid actions are penalized with a reward of -0.01. We set the train data size to 256 and use a group size of 5. Rollout and validation temperatures are set to 1.0 and 0.0, respectively. The mini-batch size is 512, and the weighting coefficient ω is set to 1 without additional tuning. The IQR coefficient λ is set to 1.5, and the weighting coefficients α, β , and γ (Equations 9–10) are all set to 0.001.

Computing Details. Qwen2.5-1.5B-Instruct uses 2xA100 GPUs and Qwen2.5-7B-Instruct uses 4xA100 GPUs, with training extending to 150 iterations on ALFWorld and 300 on WebShop. For search-augmented QA, Qwen2.5-3B-Instruct uses 4xA100 GPUs and Qwen2.5-7B-Instruct uses 8xA100 GPUs, each for 200 iterations.

E.2 Prompts

The prompts we use for STAPO are presented in Figures 12–17. These prompt templates are constructed using Python-style string formatting, where placeholders enclosed in curly braces ($\{\}$) represent semantic slots. These placeholders, such as $\{\text{task_description}\}$, $\{\text{step_count}\}$, and $\{\text{current_observation}\}$, are dynamically populated at runtime via Python’s `.format()` function.

The `<think>` `</think>` block instructs the agent to explicitly perform step-by-step reasoning, thereby promoting chain-of-thought (Wei et al., 2022) style deliberation. The `<action>` `</action>` block is

Type	Method	ALFWorld							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
<i>Qwen2.5-1.5B-Instruct</i>										
Outcome RL	DAPO	88.1	61.4	82.5	90.1	83.9	69.5	80.8	84.6	66.1
Process RL	GiGPO _{dynamic}	<u>93.3</u>	75.0	<u>96.9</u>	<u>93.8</u>	<u>91.7</u>	<u>81.8</u>	<u>91.4</u>	<u>87.5</u>	<u>75.0</u>
Process RL	STAPO_{dynamic}	99.1	<u>72.2</u>	100.0	100.0	95.8	93.5	96.1	91.3	80.2

Table 4: Generalization performance on ALFWorld and WebShop when integrating strategies adapted from DAPO. The best and second-best results are **bold** and underlined.

Method	ALFWorld						
	Pick	Look	Clean	Heat	Cool	Pick2	All
<i>Llama3.1-8B-Instruct</i>							
GiGPO	<u>89.7</u>	<u>83.3</u>	<u>96.8</u>	100.0	<u>82.6</u>	<u>90.0</u>	<u>91.4</u>
STAPO	93.1	100.0	100.0	<u>89.5</u>	95.7	95.0	95.3
<i>Qwen2.5-14B-Instruct</i>							
GiGPO	99.1	<u>77.8</u>	100.0	<u>89.9</u>	<u>86.4</u>	<u>90.1</u>	<u>93.2</u>
STAPO	99.1	91.7	100.0	100.0	94.4	91.7	97.2

Table 5: Adaptation performance on ALFWorld across different model architectures and scales. The best and second-best results are **bold** and underlined.

used to clearly indicate the final action decision. The search agent outputs reasoning traces within `<think>` `</think>`, issues search queries within `<search>` `</search>`, provides answers within `<answer>` `</answer>`. Retrieved evidence from the retriever is presented in `<information>` `</information>` tags.

F Generalization and Adaptation

To further verify the versatility and robustness of STAPO, we conduct extensive experiments focusing on its compatibility with advanced RL methods and its scalability across different model architectures. The results are summarized in Table 4 and Table 5.

Generalization to Other Group-Based RL Methods. The STAPO framework is orthogonal to advancements in single-turn group-based RL, allowing it to seamlessly incorporate complementary techniques. To demonstrate this, we follow the setup of DAPO (Yu et al., 2025), which enhances GRPO with dynamic sampling and clip-higher mechanisms. We integrate these strategies into both GiGPO and STAPO, yielding variants denoted as GiGPO_{dynamic} and STAPO_{dynamic}. As shown in Table 4, while GiGPO_{dynamic} improves over the DAPO baseline, STAPO_{dynamic} achieves significantly superior results. Specifically, on WebShop, STAPO_{dynamic} attains a success rate of 80.2%, outperforming GiGPO_{dynamic} (75.0%) by a substantial

margin. On ALFWorld, it reaches a remarkable 96.1% success rate. These results confirm that STAPO can effectively synergize with and amplify the benefits of DAPO, establishing a new state-of-the-art performance.

Adaptation across Different Model Architectures and Scales. We further evaluate the scalability of STAPO by applying it to models with larger scales and different architectures: Llama3.1-8B-Instruct and Qwen2.5-14B-Instruct. As presented in Table 5, STAPO consistently outperforms the GiGPO baseline. For Llama3.1-8B-Instruct, STAPO improves the overall success rate on ALFWorld from 91.4% to 95.3%. Similarly, on the larger Qwen2.5-14B-Instruct model, STAPO pushes the performance to a near-perfect 97.2%, compared to 93.2% for GiGPO. This consistent improvement highlights that the trajectory-aware optimization in STAPO is model-agnostic and remains highly effective as model scale increases.

G Robustness to QA Similarity Threshold

For search-augmented QA we use Qwen2.5-3B-Instruct and vary the longest matching subsequence similarity threshold for anchor state grouping from 0.9 to 0.5. Table 6 shows per-dataset performance, the average over seven datasets, and the average group size. Performance stays relatively flat while group size grows, suggesting robustness to moderate grouping noise.

Thres.	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	Avg. Perf.	Avg. Group Size
<i>Qwen2.5-3B-Instruct</i>									
0.9	43.1	60.7	45.8	36.9	36.4	13.0	64.9	43.0	2.21
0.8	43.2	60.1	44.4	37.3	37.4	13.0	65.3	43.0	2.20
0.7	45.2	60.6	45.8	37.5	33.5	12.8	62.9	42.6	2.37
0.6	44.0	60.0	46.2	36.8	36.3	12.1	61.7	42.4	3.04
0.5	43.8	61.1	46.3	38.1	36.8	13.2	60.9	42.9	3.15

Table 6: Effect of anchor state similarity threshold on search-augmented QA tasks.

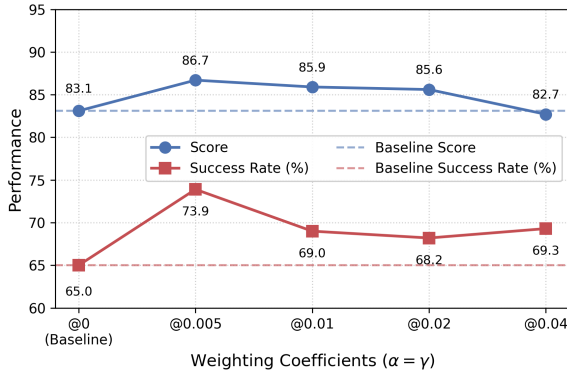


Figure 7: Sensitivity analysis of weighting coefficients α and γ on WebShop, under the constraint $\alpha = \gamma$. The standard global KL penalty coefficient is fixed at $\beta = 0.01$.

H Sensitivity Analysis of Weighting Coefficients α and γ

To investigate the impact of the selective optimization components on model performance, we conduct a sensitivity analysis on the weighting coefficients for the trajectory-aware reward α and the trajectory-independent penalty γ .

Settings. We conduct this analysis on the WebShop benchmark using the Qwen2.5-1.5B-Instruct model. Based on the hypothesis that the reward signal and the penalty constraint should be balanced to prevent optimization instability, we impose the constraint $\alpha = \gamma$ and sweep their values across the set $\{0, 0.005, 0.01, 0.02, 0.04\}$. The coefficient for the standard global KL penalty, β , is fixed at 0.01 throughout the experiments. Note that the setting $\alpha = \gamma = 0$ corresponds to the GiGPO baseline, where no selective trajectory-aware optimization is applied.

Analysis. The results, illustrated in Figure 7, demonstrate that STAPO benefits significantly from the proposed trajectory-aware mechanisms within a reasonable hyperparameter range.

- **Optimal Performance at Moderate**

Weights: The model achieves peak performance at $\alpha = \gamma = 0.005$, attaining a reward score of 86.7 and a success rate of 73.9%. This represents a substantial improvement over the baseline, which yields a score of 83.1 and a success rate of 65.0%, demonstrating the effectiveness of introducing trajectory-aware supervision.

- **Robustness and Degradation:** As the weights increase to 0.01 and 0.02, the performance decreases slightly from the peak but remains consistently superior to the baseline. However, further increasing the weights to $\alpha = \gamma = 0.04$ leads to a performance drop, with the score falling to 82.7, slightly below the baseline level (83.1). This suggests that excessively high weights cause the auxiliary objectives to dominate the total loss, overshadowing the primary task reward and leading to optimization instability.

In summary, STAPO demonstrates robustness to hyperparameter variations within the range of $[0.005, 0.02]$, with 0.005 providing the best results.

I Internal Composition of Outliers

While the main text demonstrates that reducing the proportion of outliers correlates with performance gains, this section investigates the internal composition of these outliers. Specifically, we analyze whether the outliers identified by our normalized entropy metric stem from high uncertainty or excessive certainty, both of which can be manifestations of trajectory neglect.

We classify outliers into two categories based on the IQR thresholds: high-entropy outliers ($H_n > Q_3 + \lambda \cdot \text{IQR}$) and low-entropy outliers ($H_n < Q_1 - \lambda \cdot \text{IQR}$). Table 7 presents the distribution of these outlier types across the Qwen2.5-1.5B-Instruct model series. We observe two key phenomena:

Method	ALFWorld (Outliers, %)		WebShop (Outliers, %)	
	Low-Entropy	High-Entropy	Low-Entropy	High-Entropy
<i>Qwen2.5-1.5B-Instruct</i>				
Base	3.0	97.0	9.1	90.9
GiGPO	0.0	100.0	0.0	100.0
STAPO	0.0	100.0	0.0	100.0

Table 7: Distribution of low-entropy vs. high-entropy outliers within the outlier set located by normalized entropy. The statistics are calculated based on the Qwen2.5-1.5B-Instruct model series on the test sets.

Method	Group Size	Single-Hop QA			Multi-Hop QA			Avg.	Time per Step	
		NQ [†]	TriviaQA* [*]	PopQA* [*]	HotpotQA [†]	2Wiki* [*]	MuSiQue* [*]			Bamboogle* [*]
<i>Qwen2.5-7B-Instruct</i>										
GiGPO	5	<u>46.4</u>	64.7	<u>46.1</u>	41.6	<u>43.6</u>	18.9	68.9	<u>47.2</u>	244s
	10	45.3	<u>65.2</u>	46.0	<u>41.8</u>	42.4	17.4	69.4	46.8	474s
STAPO	5	48.8	66.0	48.7	43.0	45.0	<u>18.1</u>	69.4	48.4	<u>258s</u>

Table 8: Efficiency analysis with different group sizes on search-augmented QA tasks. Results are reported in terms of performance and training time per step. The best and second-best results are **bold** and underlined.

Method	Avg. KL	Avg. Attr.	Succ. (%)
<i>Qwen2.5-1.5B-Instruct</i>			
Base	0.40	0.51	4.1
GiGPO	0.77	0.60	86.7
STAPO	0.92	0.67	92.2

Table 9: Trajectory reliance analysis on ALFWorld.

Method	Avg. KL	Avg. Attr.	Succ. (%)
<i>Qwen2.5-1.5B-Instruct</i>			
Base	0.70	0.94	5.2
GiGPO	1.22	1.06	65.0
STAPO	1.61	1.12	69.0

Table 10: Trajectory reliance analysis on WebShop.

Dominance of High-Entropy Outliers. Across all models and datasets, the vast majority of outliers (over 90% in the base model) are high-entropy. This indicates that trajectory neglect in long-horizon tasks primarily manifests as uncertainty or "getting lost" rather than stubborn over-confidence.

Elimination of Low-Entropy Outliers via RL. The base model does exhibit a small but noticeable fraction of low-entropy outliers (e.g., 9.1% in WebShop), typically corresponding to repetitive loops. However, after applying RL (both GiGPO and STAPO), these low-entropy outliers are completely eliminated (0.0%).

J Training Efficiency

To evaluate the computational cost-effectiveness of our approach, we analyze the training efficiency

Method	Sokoban (6×6)	EZPoints
<i>Qwen2.5-VL-3B-Instruct</i>		
Base	11.7	3.1
GRPO	67.1	86.9
GiGPO	76.9	100.0
STAPO	80.5	100.0

Table 11: Success rates (%) on VLM agents.

on search-augmented QA tasks using Qwen2.5-7B-Instruct. As shown in Table 8, STAPO incurs a negligible computational overhead compared to the GiGPO baseline with the same group size ($N = 5$)—increasing training time per step by only $\sim 5.7\%$ (from 244s to 258s) primarily due to the additional forward passes required for the calculation of trajectory-aware reward and trajectory-independent penalty—yet achieves a substantial performance gain. Furthermore, STAPO demonstrates superior efficiency over brute-force scaling; simply doubling the group size for GiGPO ($N = 10$) nearly doubles the training cost (474s) while paradoxically degrading performance (46.8). These results confirm that STAPO effectively trades a marginal computational cost for significant robustness and performance improvements, establishing it as a more efficient training paradigm than simple sample scaling.

K Trajectory Reliance Analysis

On the ALFWorld and WebShop test sets, we measure (i) the average KL divergence between the policy output under the full prompt c_t and under the trajectory-blind prompt $(c_t)_{\setminus \mathcal{T}}$, and (ii) the average

trajectory attribution score. Higher KL indicates stronger sensitivity to trajectory context; higher attribution indicates greater use of trajectory in the model’s decision. Tables 9 and 10 summarize the results together with success rate.

L Vision-Language Agent Experiments

We extend STAPO to vision-language model (VLM) agents by applying the same masking $(\cdot)_{\setminus \mathcal{T}}$ to strip the task goal and interaction history (including all previous paired visual frames and textual instructions, as well as past actions). We train Qwen2.5-VL-3B-Instruct with the same GiGPO baseline configuration on two visual interactive environments: **Sokoban** (6×6), a spatial planning puzzle from image observations, and **EZPoints**, a multi-step symbolic reasoning task over visual playing cards. Table 11 reports success rates (%).

Case: Neglect of Task Goal

Task: look at cd under the desklamp.

Step 21

Observation: You pick up the cd 1 from the shelf 1.

Reasoning: The previous step was to take the cd 1 from the shelf 1. Now, the current situation is that I have the cd 1 in my possession. I need to look at the cd 1 to understand what it is. I also need to make sure it's necessary to look at the cd 1 or if there's something else I need to do first.

Action: examine cd 1

Step 22

Observation: There's nothing special about cd 1.

Reasoning: The previous action was to take cd 1 from the shelf 1. Now, there is only one cd left on the shelf 1.

Action: go to desk 1

Step 23

Observation: You arrive at desk 1. On the desk 1, you see a cellphone 1, a creditcard 2, and a creditcard 1.

Reasoning: At the previous steps, I picked up and examined the CD1, and then went to the desk 1. Now, I am at step 23, at the desk 1. I see several items on the desk: a cellphone1, a creditcard2, and a creditcard1. There don't appear to be any CDs or shelves that I haven't referred to in this scenario. However, I might want to ensure I've correctly inspected the items, especially if this information is necessary for later actions.

Action: go to bed 1 ← *Deviates from the goal "desklamp"*

Step 24

Observation: You arrive at bed 1. On the bed 1, you see a laptop 2, a laptop 1, a pillow 1, and a teddybear 1.

Reasoning: The previous actions were to go to desk 1 and examine bed 1. Now, at step 24, I am at bed 1. The most recent observation is that I am at bed 1. On the bed 1, I see a laptop 2, a laptop 1, a pillow 1, and a teddybear 1.

Action: go to desk 2

Step 25

Observation: You arrive at desk 2. On the desk 2, you see a book 1, a pen 2, a pen 1, and a pencil 2.

Reasoning: The previous step showed you arriving at desk 2, where you found the following objects: book 1, pen 1, pen 2, and pencil 2. Now in step 25, you should look at the cd under the desk 2. To accomplish this, you need to move the cd from your bed (desk 1) to desk 2.

Action: examine cd 1

Figure 8: Illustration of task goal neglect in multi-turn interactions.

Case: Neglect of Interaction History

Task: look at cd under the desklamp.

Step 34

Observation: You arrive at shelf 1. On the shelf 1, you see a cd 2.

Reasoning: At step 33, you went to shelf 1, where you found a cd 2. Now, you need to look at cd 2.

Action: look at cd 2

Step 35

Observation: Nothing happens.

Reasoning: The previous actions were: 1. Going to shelf 1 and finding a CD 2. 2. Looking at the CD 2. Now, I need to decide which action to take next.

Action: look at cd 2 ← *Redundant action*

Step 36

Observation: Nothing happens.

Reasoning: The previous steps showed that I have already looked at the cd 2 on the shelf 1. Now, I need to find out if there is a cd in the garbagecan, desk, shelves 2 to 5, or if I need to do something else.

Action: look in garbagecan 1

Step 37

Observation: Nothing happens.

Reasoning: The user looked at the cd 2 and examine the cd 1 but there's nothing happening as no directory change or file operation instructions were provided. Also, there are some steps not taken as per the environment's logical flow. I need to plan how to reach the CD directory.

Action: go to desk 2

Step 38

Observation: You arrive at desk 2. On the desk 2, you see a book 1, a pen 2, a pen 1, and a pencil 2.

Reasoning: The environment has given us two possible choices: look at cd under the desklamp. Let's check both the desk 2 and the garbagecan 1. Then, we can choose the appropriate action based on the new information.

Action: examine cd 1

Figure 9: Illustration of interaction history neglect in multi-turn interactions.

Case: Avoiding False Positives in States with a Large Action Space

Task: put a cool bowl in cabinet.

Action Type: high-quality action (positive advantage). Identified as an outlier by raw Shannon entropy (IQR), but correctly as an inlier by normalized entropy.

Step 27

Observation: You arrive at cabinet 10. The cabinet 10 is open. In it, you see nothing.

Admissible Actions: ['close cabinet 10', 'examine cabinet 10', 'go to cabinet 1', 'go to cabinet 11', 'go to cabinet 12', 'go to cabinet 13', 'go to cabinet 2', 'go to cabinet 3', 'go to cabinet 4', 'go to cabinet 5', 'go to cabinet 6', 'go to cabinet 7', 'go to cabinet 8', 'go to cabinet 9', 'go to coffeemachine 1', 'go to countertop 1', 'go to diningtable 1', 'go to drawer 1', 'go to fridge 1', 'go to garbagecan 1', 'go to microwave 1', 'go to shelf 1', 'go to shelf 2', 'go to shelf 3', 'go to sinkbasin 1', 'go to stoveburner 1', 'go to stoveburner 2', 'go to stoveburner 3', 'go to stoveburner 4', 'go to toaster 1', 'inventory', 'look'] (Total: 32 selectable actions).

Reasoning: You are at cabinet 10, the cabinet is open. There is nothing inside. The goal is to put a cool bowl in a cabinet. It seems cabinets with exposed items are more likely to contain bowls as these are often kept next to food for convenience. You should go to cabinet 1 to look for a cool bowl or any other container. Inventory of cabinets: (from observation 26) - No items found. Inventory of other locations: (additional observations) - Stoves, sink, countertops, and fridge have no cool bowls. Cabinets used in the environment typically hold food items tightly sealed. Going to other cabinets to look for bowls still seems unlikely given the cold temperature would be necessary for a bowl to be cool. The table or countertops at dining table, or any shelves or drawers within them, should also contain bowls. Placing a bowl on dining table countertops could be a viable approach too if it is a dining facility environment. However, since cabinets are often recommended for preserving shorter shelf-life items like bowls and it features free space on the countertop or shelves, reaching for a bowl is likely the best choice. Combining thoughts, going to counting table or dining table seems most likely.

Action: go to countertop 1

Figure 10: Illustration of normalized entropy avoiding false positives in states with a large action space.

Case: Reducing False Negatives in States with a Small Action Space

Task: put two spraybottle in cabinet.

Action Type: low-quality action (negative advantage). Identified as an inlier by raw Shannon entropy (IQR), but correctly as an outlier by normalized entropy.

Step 43

Observation: The cabinet 2 is open. In it, you see nothing.

Admissible Actions: ['close cabinet 2', 'examine cabinet 2', 'go to bathtubbasin 1', 'go to cabinet 1', 'go to cabinet 3', 'go to cabinet 4', 'go to countertop 1', 'go to garbagecan 1', 'go to handtowelholder 1', 'go to sinkbasin 1', 'go to sinkbasin 2', 'go to toilet 1', 'go to toiletpaperhanger 1', 'go to towelholder 1', 'inventory', 'look'] (Total: 16 selectable actions).

Reasoning: In the current situation, we have taken 42 steps, but we have not seen the two spraybottles placed anywhere. We have checked cabinet 2, but it is empty. To place the spraybottles, we need to clear the space inside the cabinet first, and then find the spraybottles.

Action: close cabinet 2

Figure 11: Illustration of normalized entropy reducing false negatives in states with a small action space.

Complete Prompt Template for ALFWorld

You are an expert agent operating in the ALFRED embodied Environment. Your task is to: {task_description}. Prior to this step, you have already taken {step_count} step(s). Below are the most recent {history_length} observations and the corresponding actions you took: {action_history}. You are now at step {current_step} and your current observation is: {current_observation}. Your admissible actions of the current situation are: [{admissible_actions}].

Now it's your turn to take an action. You should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 12: The complete prompt template of ALFWorld agents.

Trajectory-Blind Prompt Template for ALFWorld

You are an expert agent operating in the ALFRED Embodied Environment. You are now at step {current_step} and your current observation is: {current_observation}. Your admissible actions of the current situation are: [{admissible_actions}].

Now it's your turn to take an action. You should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 13: The trajectory-blind prompt template of ALFWorld agents.

Complete Prompt Template for WebShop

You are an expert autonomous agent operating in the WebShop e-commerce environment. Your task is to: {task_description}. Prior to this step, you have already taken {step_count} step(s). Below are the most recent {history_length} observations and the corresponding actions you took: {action_history}. You are now at step {current_step} and your current observation is: {current_observation}. Your admissible actions of the current situation are: [{available_actions}].

Now it's your turn to take one action for the current step. You should first reason step-by-step about the current situation, then think carefully which admissible action best advances the shopping goal. This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 14: The complete prompt template of WebShop agents.

Trajectory-Blind Prompt Template for WebShop

You are an expert autonomous agent operating in the WebShop e-commerce environment. You are now at step {current_step} and your current observation is {current_observation}. Your admissible actions of the current situation are: [{available_actions}].

Now it's your turn to take one action for the current step. You should first reason step-by-step about the current situation, then think carefully which admissible action best advances the shopping goal. This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 15: The trajectory-blind prompt template of WebShop agents.

Complete Prompt Template for Search

You are an expert agent tasked with answering the given question step-by-step. Your question: {task_description}. Prior to this step, you have already taken {step_count} step(s). Below is the interaction history where <search> </search> wrapped your past search queries and <information> </information> wrapped the corresponding search results returned by the external search engine. History: {memory_context}.

Now it's your turn to respond for the current step. You should first conduct reasoning process. This process MUST be enclosed within <think> </think> tags. After completing your reasoning, choose only one of the following actions (do not perform both):

- (1) If you find you lack some knowledge, you can call a search engine to get more external information using format: <search> your query </search>.
- (2) If you have enough knowledge to answer the question confidently, provide your final answer within <answer> </answer> tags, without detailed illustrations. For example, <answer>Beijing</answer>.

Figure 16: The complete prompt template of Search agents.

Trajectory-Blind Prompt Template for Search

You are an expert agent tasked with answering the given question step-by-step.

Now it's your turn to respond for the current step. You should first conduct reasoning process. This process MUST be enclosed within <think> </think> tags. After completing your reasoning, choose only one of the following actions (do not perform both):

- (1) If you find you lack some knowledge, you can call a search engine to get more external information using format: <search> your query </search>.
- (2) If you have enough knowledge to answer the question confidently, provide your final answer within <answer> </answer> tags, without detailed illustrations. For example, <answer>Beijing</answer>.

Figure 17: The trajectory-blind prompt template of Search agents.