Proximity-Based Multi-Turn Optimization: Practical Credit Assignment for LLM Agent Training

Yangyi Fang[1,*], Jiaye Lin[1,*], Xiaoliang Fu[2,*], Cong Qin[3,*], Chang Liu[4,*], Haolin Shi[1,†]
[1]Tsinghua University [2]Fudan University [3]Peking University [4]Lanzhou University
shihaolin0720@gmail.com

Abstract

Multi-turn LLM agents are becoming pivotal to production systems, spanning customer service automation, e-commerce assistance, and interactive task management, where accurately distinguishing high-value informative signals from stochastic noise is critical for sample-efficient training. In real-world scenarios, a failure in a trivial task may reflect random instability, whereas success in a high-difficulty task signifies a genuine capability breakthrough. Yet, existing group-based policy optimization methods rigidly rely on statistical deviation within discrete batches, frequently misallocating credit when task difficulty fluctuates. To address this issue, we propose **Proximity-based Multi-turn Optimization (ProxMO)**, a practical and robust framework engineered specifically for the constraints of real-world deployment. ProxMO integrates global context via two lightweight mechanisms: success-rate-aware modulation dynamically adapts gradient intensity based on episode-level difficulty, while proximity-based soft aggregation derives baselines through continuous semantic weighting at the step level. Extensive evaluations on ALFWorld and WebShop benchmarks demonstrate that ProxMO yields substantial performance gains over existing baselines with negligible computational cost. Ablation studies further validate the independent and synergistic efficacy of both mechanisms. Crucially, ProxMO offers plug-and-play compatibility with standard GRPO frameworks, facilitating immediate, low-friction adoption in existing industrial training pipelines. Our implementation is available at: https://github.com/GithubX-F/ProxMO-RL.

1 Introduction

Reinforcement learning (RL) has become essential for training Large Language Model (LLM) agents in complex interactive tasks (Schulman et al., 2017; Christiano et al., 2017). Among various RL algorithms, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) has emerged as a scalable approach by computing advantages through group-based normalization without explicit value networks, demonstrating strong performance in single-turn scenarios.

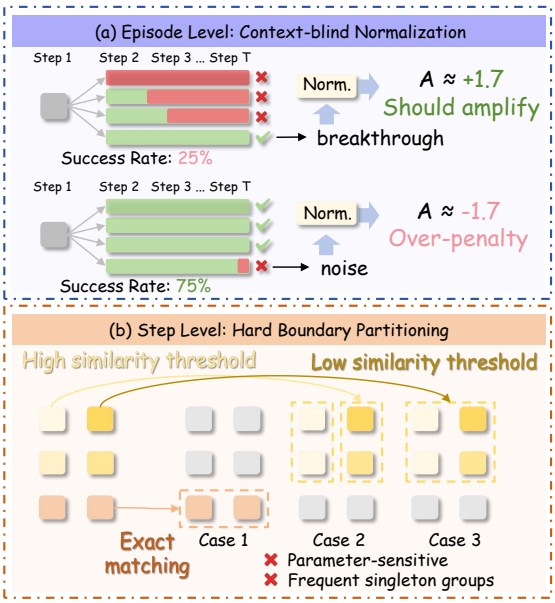

Figure 1: Motivating challenges in multi-turn policy optimization. **(a) Context-blind normalization:** identical z-score magnitudes yield uniform advantage intensities across high-success (e.g., 75%) and low-success (e.g., 25%) groups, ignoring informational heterogeneity. **(b) Hard boundary partitioning:** binary participation (in/out based on threshold) with equal intra-group weighting, causing singleton degeneracy under strict criteria or incorrect equal weighting under loose criteria.

Recent efforts have extended GRPO to real-world multi-turn interactive tasks (Shridhar et al., 2020; Yao et al., 2022a), requiring agents to navigate sequential decision-making processes over extended horizons. In such practical settings, agents confront a highly heterogeneous landscape where task difficulty varies dramatically, while managing observation spaces laden with high-dimensional

[*] Equal contribution. [†] Corresponding author.

lexical ambiguity. This phenomenon creates a fundamental credit assignment challenge: *outcomes carry inherently context-dependent informational values*, meaning a failure may reflect random noise or critical error, and a success may represent routine performance or rare breakthrough. Yet, prevailing optimization methods rely solely on within-group statistical deviation, systematically ignoring these vital contextual distinctions and consequently misallocating learning signals during training.

This context-dependence manifests detrimentally at both hierarchical levels: (i) **At the episode level**, while standard z-score normalization effectively captures statistical deviation, it ignores the reality that identical deviations yield vastly different informational values. Specifically, a failure in a task with a 75% success rate likely reflects mere stochastic noise, whereas a success in a complex scenario with only a 25% success rate represents a genuine capability breakthrough—yet conventional approaches assign similar advantage magnitudes to both based purely on statistical position. This fundamental asymmetry is analyzed in detail in Appendix A.1. (ii) **At the step level**, existing methods employ hard boundary partitioning (Feng et al., 2025b) via exact matching or similarity thresholds, creating discrete clusters where states either fully participate or are entirely excluded. This produces an irresolvable trade-off: strict criteria generate singleton groups that preclude baseline comparison, while loose criteria indiscriminately weight states regardless of semantic proximity. The prevalence and consequence of this degeneracy are illustrated in Figure 1 and quantified in Appendix A.2.

To effectively address these limitations, we propose **Proximity-based Multi-turn Optimization (ProxMO)**, a practical framework incorporating global context at two distinct hierarchical levels. At the episode level, we introduce *success-rate-aware modulation*, a mechanism that adapts credit allocation relative to task difficulty, attenuating noise penalties in high-success regimes while amplifying breakthrough signals in low-success regimes. At the step level, *proximity-based soft aggregation* replaces hard boundaries with continuous weighting where all states contribute proportionally to semantic proximity, eliminating singleton degeneracy and equal-weight limitations. Experiments on ALFWorld and WebShop demonstrate consistent improvements, with ablations confirming independent and synergistic contributions. Our core contributions are summarized as follows:

- We propose ProxMO, a unified framework incorporating global context into the credit assignment process at both episode and step levels, addressing limitations of existing methods that rely solely on within-group statistics.
- At the episode level, we introduce success-rate-aware modulation, adapting gradient intensity to task difficulty. At the step level, we design a proximity-based soft aggregation mechanism that replaces hard boundaries with continuous weighting, where all states contribute proportionally to semantic proximity.
- Comprehensive experiments on ALFWorld and WebShop validate ProxMO's effectiveness, with ablations revealing independent and synergistic contributions from both mechanisms.

2 Related Work

2.1 Multi-Turn Interaction with LLM Agents

LLMs have evolved from static responders to autonomous agents capable of sustained multi-turn interaction—embodied tasks (Shridhar et al., 2020; Li et al., 2024), GUI navigation (Furuta et al., 2023; Zheng et al., 2024; Gou et al., 2024; Feng et al., 2025a), strategic gameplay (Wang et al., 2023, 2024)—maintaining coherent perception-reasoning-action loops over extended trajectories. Early approaches relied on prompt engineering (Yao et al., 2022b; Shinn et al., 2023), memory systems (Wang et al., 2024; Tan et al., 2024), and tool integration (Schick et al., 2023; Xie et al., 2024), but these static methods struggle with distribution shifts. Recent work has shifted toward learning-based adaptation via supervised fine-tuning (Zhang and Zhang, 2024) or reinforcement learning (Sutton et al., 1998), though multi-turn settings introduce unique challenges: sparse rewards complicate credit assignment while sequential engagement inflates sample costs. A parallel line of work improves agent self-improvement via world models (Fang et al., 2025) and iterative exploration (He et al., 2025), highlighting the importance of credit-aware optimization in long-horizon settings.

2.2 RL for Agentic Optimization

RL has evolved from actor-critic RLHF (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Schulman et al., 2017) to group-relative

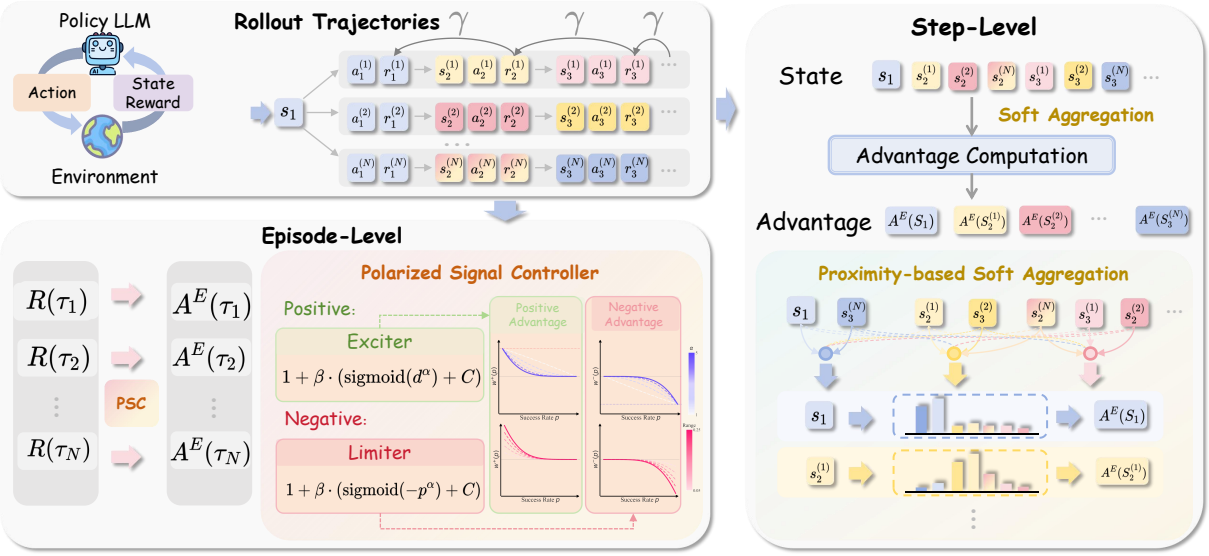

Figure 2: The overview of ProxMO. Episode-level: success-rate-aware modulation adapts credit to task difficulty (i.e., p). Step-level: proximity-based soft aggregation eliminates discrete boundaries for robust baseline estimation.

methods (Shao et al., 2024; Kool et al., 2019; Ahmadian et al., 2024; Liu et al., 2025b; Yu et al., 2025; Fu et al., 2026a,b) that compute advantages within sample batches without value networks, demonstrating strong performance in single-turn reasoning tasks (Guo et al., 2025; Jin et al., 2025; Sun et al., 2025; Qian et al., 2025).

Multi-turn extensions face fundamental challenges in credit assignment. GiGPO (Feng et al., 2025b) introduces step-level credit assignment through exact state matching, but produces singleton groups in high-dimensional spaces where within-group normalization becomes undefined. STEP (Chen et al., 2025) incorporates task difficulty at the *sampling level* by dynamically allocating rollouts to low-success tasks, improving data collection efficiency. Similarly, Fang et al. (2026b) jointly optimize rollout allocation and advantage modulation for policy gradient methods. CSO (Li et al., 2026) further addresses step-level credit by verifying critical steps, yet still relies on discrete verification signals rather than continuous proximity weighting. Question augmentation methods (Fang et al., 2026a) complement these by reshaping problem difficulty at the input level.

In contrast, ProxMO addresses task difficulty at both levels: episode-level modulation adjusts learning intensity based on success rates, while step-level aggregation eliminates discrete boundaries through continuous proximity weighting.

3 Preliminaries

We formalize the multi-turn agent tasks as a sequential decision-making process in which a parameterized LLM policy π_θ interacts with a dynamic environment over multiple steps. At each time step t, the agent observes a current state $s_t \in \mathcal{S}$, generates a corresponding action $a_t \in \mathcal{A}$, receives a scalar reward $r_t \in \mathbb{R}$, and transitions to the subsequent state s_{t+1} governed by the environment dynamics. Consequently, a complete interaction episode is defined as a trajectory:

$$\tau = \{(s_1, a_1, r_1), (s_2, a_2, r_2), \ldots, (s_T, a_T, r_T)\}, \quad (1)$$

where T denotes the horizon of the episode. The total return of such a trajectory is defined as:

$$R(\tau) = \sum_{t=1}^{T} r_t, \quad (2)$$

and the goal is to find the optimal policy π_θ that maximizes the expected return $\mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]$.

Following GRPO (Shao et al., 2024), we sample N trajectories $\{\tau_1, \tau_2, \ldots, \tau_N\}$ under the identical task instruction and initial state. GRPO computes advantages purely from group statistics without requiring the critic networks. Specifically, the advantage of trajectory τ_i is derived as:

$$A(\tau_i) = \text{GroupComputation}\big(\{R(\tau_1), R(\tau_2), \ldots, R(\tau_N)\}\big), \quad (3)$$

where GroupComputation$(\cdot)$ typically entails normalizing returns relative to the intra-group mean and standard deviation. This group-based approach

eliminates the need for explicit value function estimation, making it highly memory-efficient and scalable for practical LLM training.

4 Methodology

ProxMO incorporates global context at two hierarchical levels (as shown in Figure 2): episode-level modulation adapts credit assignment to task difficulty and step-level aggregation leverages semantic proximity to eliminate discrete group boundaries.

4.1 Episode-Level: Success-Rate-Aware Advantage Modulation

For a task x, we sample N trajectories $\{\tau_i\}_{i=1}^N$ from identical initial states, computing returns $R(\tau_i) = \sum_{t=1}^T r_t^{(i)}$. Assuming the binary rewards ($R(\tau_i) \in \{0, 1\}$), the episode-level group is:

$$G^E = \{(\tau_1, R(\tau_1)), (\tau_2, R(\tau_2)), \ldots, (\tau_N, R(\tau_N))\}. \quad (4)$$

Standard GRPO computes episode advantages by employing z-score normalization:

$$A^E(\tau_i) = \frac{R(\tau_i) - \mu}{\sigma}, \quad (5)$$

where $\mu = \frac{1}{N}\sum_{j=1}^N R(\tau_j)$ denotes the mean return and $\sigma = \sqrt{\frac{1}{N}\sum_{j=1}^N (R(\tau_j) - \mu)^2}$ denotes the standard deviation. However, identical z-scores yield uniform advantage magnitudes regardless of task difficulty—an issue we address next.

At the episode level, we introduce *Polarized Signal Controller* (PSC) to implement success-rate-aware modulation. Let p denote the empirical success rate of the episode group G^E, we define success-rate-dependent scaling weights as:

$$w(R, p) = 1 + \beta \cdot f(R, p),$$
$$f(R, p) = \begin{cases} \text{Sigmoid}(d^\alpha) - 0.5, & \text{if } R = 1, \\ \text{Sigmoid}(-p^\alpha) - 0.5, & \text{if } R = 0. \end{cases} \quad (6)$$

Here, $d = 1 - p$ is the failure rate, $\text{Sigmoid}(x) = 1/(1 + e^{-x})$ serves as the non-linear activation, while hyperparameters β and α govern the modulation strength and steepness, respectively. Successes are amplified in low-success groups to consolidate rare breakthroughs, while failures are attenuated in high-success groups to reduce noise penalties. The modulated episode-level advantage is:

$$\tilde{A}^E(\tau_i) = w(R(\tau_i), p) \cdot A^E(\tau_i). \quad (7)$$

4.2 Step-Level: Proximity-Based Soft Aggregation

While episode-level advantages provide necessary trajectory-wide feedback, they inherently lack granularity, failing to differentiate action quality within trajectories, e.g., a failed trajectory indiscriminately assigns identical advantages to all steps despite varying action quality. Meanwhile, existing step-level methods employ hard boundary partitioning (Feng et al., 2025b) via exact matching or similarity thresholds. Strict criteria produce singleton groups where normalization becomes undefined, while loose criteria assign equal weight to states with vastly different semantic proximity.

To overcome these limitations, we introduce *Proximity-based Soft Aggregation* (PSA). Rather than partitioning states into discrete groups, we compute baselines by aggregating returns from all states weighted by their semantic proximity. For each action $a_t^{(i)}$ taken from state $s_t^{(i)}$, we define its discounted return from step t onward:

$$R_t^{(i)} = \sum_{k=t}^T \gamma^{k-t} r_k^{(i)}, \quad (8)$$

where $\gamma \in (0, 1]$ is the discount factor. This formulation explicitly reflects the long-term impact of action $a_t^{(i)}$ beyond immediate reward $r_t^{(i)}$.

For efficiency and scalability, we adopt Term Frequency–Inverse Document Frequency (TF-IDF) to measure semantic similarity. We represent each state $s_t^{(i)}$ via TF-IDF vectors $\mathbf{v}_t^{(i)} = \text{TF-IDF}(s_t^{(i)})$ and compute L2-normalized cosine similarity:

$$\text{sim}(s_t^{(i)}, s_t^{(j)}) = \frac{\mathbf{v}_t^{(i)} \cdot \mathbf{v}_t^{(j)}}{\|\mathbf{v}_t^{(i)}\|_2 \|\mathbf{v}_t^{(j)}\|_2}. \quad (9)$$

Let $\mathcal{G}(i)$ denote the index set of trajectories sharing the same task as trajectory i. For step t, we compute the temperature-scaled weights, restricting step-level comparisons within the episode groups:

$$w_{ij} = \begin{cases} \dfrac{\exp\big(\text{sim}(s_t^{(i)}, s_t^{(j)})/\tau\big)}{\sum_{k \in \mathcal{G}(i)} \exp\big(\text{sim}(s_t^{(i)}, s_t^{(k)})/\tau\big)}, & \text{if } j \in \mathcal{G}(i), \\ 0, & \text{otherwise}, \end{cases} \quad (10)$$

where τ is the temperature controlling the concentration of weights around high-similarity states. The soft baseline and step-level advantage are:

$$B_t^{(i)} = \sum_{j \in \mathcal{G}(i)} w_{ij} R_t^{(j)}, \quad (11a)$$

$$A^S(a_t^{(i)}) = R_t^{(i)} - B_t^{(i)}. \quad (11b)$$

When $\tau \to 0$, weights concentrate on the nearest neighbor (approximating exact matching); when $\tau \to \infty$, weights become uniform (degenerating to episode-level baseline). In practice, we set $\tau = 0.1$ to balance proximity sensitivity and stability.

4.3 Unified Training Objective

In this section, we combine episode-level and step-level advantages via a unified weighted summation:

$$A(a_t^{(i)}) = \tilde{A}^E(\tau_i) + \omega \cdot A^S(a_t^{(i)}), \tag{12}$$

where ω balances episode and step signals ($\omega = 1$ by default). The policy optimization objective follows the clipped PPO formulation:

$$\begin{aligned}\mathcal{J}(\theta) = \mathbb{E}\Bigg[\frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\min\Big(\rho_t^{(i)}A(a_t^{(i)}),\\ \text{Clip}(\rho_t^{(i)}, 1\pm\epsilon)A(a_t^{(i)})\Big)\Bigg]\\ -\beta_{\text{KL}}\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}}),\end{aligned} \tag{13}$$

where $\rho_t^{(i)} = \pi_\theta(a_t^{(i)}|s_t^{(i)}, x)/\pi_{\theta_{\text{old}}}(a_t^{(i)}|s_t^{(i)}, x)$ is the importance sampling ratio, ϵ is the clipping parameter, and expectations are estimated over the task distribution $x \sim p(X)$ and the sampled trajectories $\{\tau_i\}_{i=1}^N \sim \pi_{\theta_{\text{old}}}$.

5 Experiments

5.1 Experimental Setup

Benchmarks. We evaluate ProxMO on two real-world multi-turn interactive benchmarks: (i) *ALFWorld* (Shridhar et al., 2020), an embodied environment with 3,827 task instances across six household activity categories, i.e., Pick & Place (Pick), Examine in Light (Look), Clean & Place (Clean), Heat & Place (Heat), Cool & Place (Cool), Pick Two & Place (Pick2). (ii) *WebShop* (Yao et al., 2022a), a web-based shopping environment with 1.1M products and 12K user instructions requiring HTML navigation and purchase decisions.

Baselines. We compare ProxMO with a range of competitive baselines: (i) *Closed-source LLMs*, specifically GPT-4o (Achiam et al., 2023) and Gemini-2.5-Pro (Team et al., 2023), which represent leading general-purpose reasoning capabilities. (ii) *Prompting agents*, including ReAct (Yao et al., 2022b) and Reflexion (Shinn et al., 2023), which rely on in-context prompting without parameter updates. (iii) *RL training methods*: comprising GRPO (Shao et al., 2024), a group-based critic-free method that performs advantage estimation over trajectory groups, and GiGPO (Feng et al., 2025b), a recent advancement that introduces step-level credit assignment via exact state matching.

Training Details. We employ Qwen2.5-1.5B/7B-Instruct models (Qwen et al., 2024) as our experimental backbones for prompting agents and RL training methods, with hyperparameters as follows: $\alpha = 4.0$, $\beta = 0.1$, $\tau = 0.1$, $\gamma = 0.95$, $\omega = 1.0$, $N = 8$. All methods use identical configurations for fair comparison, and the complete training details are provided in Appendix C.1.

5.2 Main Results

As detailed in Table 1, ProxMO consistently outperforms baselines across LLM scales and task types, with pronounced gains in long-horizon tasks (e.g., Look, Cool, Pick2) that demand precise credit assignment. Notably, our trained small models (1.5B/7B) match or even surpass leading closed-source LLMs like GPT-4o and Gemini-2.5-Pro, demonstrating exceptional industrial viability.

The improvements stem from resolving two critical limitations: unlike GRPO's context-agnostic normalization that misallocates credit or GiGPO's discrete grouping that suffers from sparsity in high-dimensional spaces, ProxMO employs episode-level success-rate-aware modulation to amplify rare successes in challenging tasks, and step-level proximity-based soft aggregation to enable meaningful baseline comparisons where exact state matching degenerates into singletons. This hierarchical design provides more informative advantage estimates that accelerate convergence while maintaining stability in complex, practical environments.

5.3 Hyperparameter Sensitivity

In Figure 3, we analyze the hyperparameter sensitivity of ProxMO with Qwen2.5-1.5B-Instruct, where episode-level hyperparameters (α, β) and step-level temperature (τ) reveal consistent stability across broad intervals, with optimal values at $\alpha = 4.0$, $\beta = 0.1$, $\tau = 0.1$. Across all three hyperparameters, performance degrades gracefully at the boundaries rather than exhibiting abrupt collapse, indicating that the underlying mechanisms are inherently robust to misspecification. Notably, the same hyperparameter configuration achieves near-optimal results across different LLM scales and task types, demonstrating robustness that is critical for large-scale practical deployment, where

Table 1: Comparison of benchmark results on unseen test instances (averaged over 3 random seeds) across Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct. Best results are **bold**, second-best are underlined. Δ *vs GRPO* denotes relative improvement (%) over GRPO. Key metrics (All, Succ.) are highlighted.

Method	ALFWorld							WebShop	
	Pick	Look	Clean	Heat	Cool	Pick2	**All**	Score	**Succ.**
Closed-Source LLMs									
GPT-4o	75.3	60.8	31.2	56.7	21.6	49.8	48.0	31.8	23.7
Gemini-2.5-Pro	92.8	63.3	62.1	69.0	26.6	58.7	60.3	42.5	35.9
Qwen2.5-1.5B-Instruct									
Base	5.9	5.5	3.3	9.7	4.2	0.0	4.1	25.1	6.3
ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8	42.1	14.3
Reflexion	37.8	24.0	23.3	14.5	20.7	3.9	23.5	58.6	23.5
GRPO	80.0	50.0	75.0	88.9	63.2	50.0	70.3	73.1	52.2
GiGPO	**95.3**	80.2	**92.9**	**92.7**	70.6	78.5	85.2	81.7	62.3
ProxMO (Ours)	94.3	**92.9**	89.3	92.2	**89.5**	**87.0**	**90.6**	**85.3**	**67.1**
Δ *vs GRPO*	+17.9%	+85.8%	+19.1%	+3.7%	+41.6%	+74%	+28.9%	+16.7%	+28.5%
Qwen2.5-7B-Instruct									
Base	34.8	22.9	18.1	7.3	2.5	3.6	16.2	25.1	8.4
ReAct	50.1	33.8	35.7	12.5	17.3	18.9	29.8	47.8	21.0
Reflexion	63.4	40.2	46.5	29.7	37.9	22.6	44.1	56.3	30.2
GRPO	90.7	66.2	94.1	91.2	78.9	70.5	79.8	79.2	67.2
GiGPO	97.5	81.3	88.5	85.7	90.0	83.5	89.5	85.5	74.8
ProxMO (Ours)	**98.4**	**88.6**	**95.7**	**93.8**	**91.3**	**89.8**	**94.5**	**87.2**	**76.5**
Δ *vs GRPO*	+8.5%	+33.8%	+1.7%	+2.9%	+15.7%	+27.4%	+18.4%	+10.1%	+13.8%

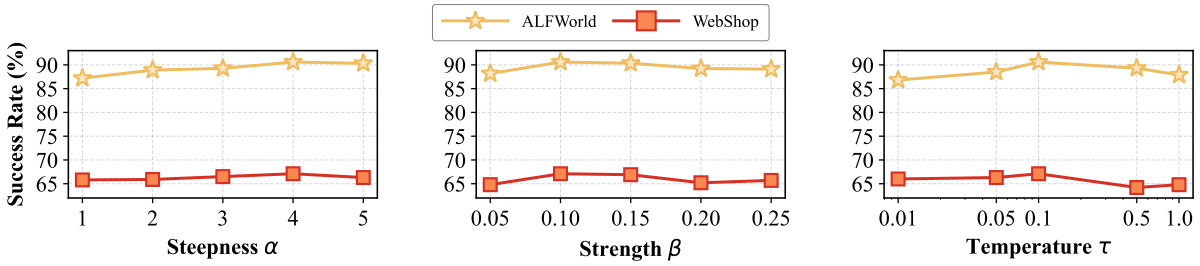

Figure 3: Hyperparameter sensitivity analysis on ALFWorld and WebShop, with ProxMO maintaining stable high performance across broad parameter configurations. For clarity, temperature τ is visualized on a logarithmic scale.

extensive hyperparameter tuning is infeasible. This cross-domain consistency stems from ProxMO operating on normalized quantities—success rates and L2-normalized similarities—that are inherently scale-invariant and task-agnostic, thereby eliminating the need for per-task hyperparameter search and further reducing the adoption barrier in production settings.

5.4 Ablation Study

To validate the effectiveness of each proposed mechanism, we conduct a comprehensive ablation study on ALFWorld with Qwen2.5-1.5B-Instruct. As visualized in Figure 4, removing either episode-level modulation (PSC) or step-level aggregation (PSA) results in consistent degradation across all task categories, confirming their independent contributions. Notably, removing PSA causes more severe performance drops, particularly in long-horizon tasks requiring precise action sequencing, while removing PSC primarily affects tasks with high success-rate variance where context-dependent gradient scaling proves critical. This asymmetry aligns with our theoretical motivation: PSA provides the fine-grained per-step discrimination that is essential for complex sequential decisions, whereas PSC calibrates the overall learning intensity according to task difficulty. Crucially, the full ProxMO not only surpasses all ablated variants but also outperforms the strong baseline GiGPO, demonstrating genuine synergistic effects that exceed the additive expectation from individual mechanisms. This synergy is most pronounced in complex tasks where episode-level difficulty adaptation amplifies the value of step-level credit precision, enabling stable learning in heterogeneous environments where either mechanism alone proves insufficient.

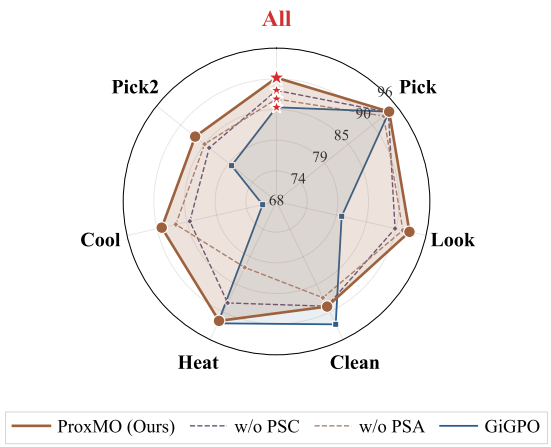

Figure 4: Ablation study on ALFWorld, where ProxMO outperforms all variants and the strong baseline GiGPO.

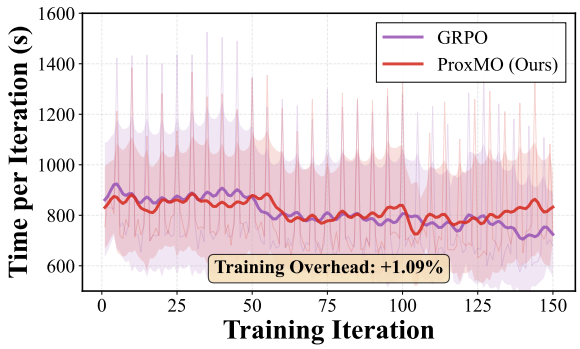

Figure 5: Training time comparison on ALFWorld (shaded regions denote confidence intervals) reveals ProxMO incurs a minimal additional overhead (+1.09%) versus GRPO across training iterations, confirming its computational efficiency for scalable pipelines.

5.5 Computational Efficiency

As illustrated in Figure 5, we compare the training time between ProxMO and GRPO on ALFWorld with Qwen2.5-1.5B-Instruct. Despite introducing episode-level modulation and step-level aggregation, ProxMO incurs only negligible overhead across training iterations, as both mechanisms operate through lightweight arithmetic operations without requiring additional neural networks or model forward passes (unlike critic-based methods (Schulman et al., 2017)). Specifically, the episode-level PSC involves only scalar multiplications conditioned on empirical success rates, while the step-level PSA computes TF-IDF similarities and softmax-weighted baselines—operations that execute efficiently on CPU without competing for GPU resources during model training. The near-identical training curves with narrow confidence intervals confirm that ProxMO delivers precise credit assignment without compromising the throughput and scalability essential for large-scale industrial adoption, enabling immediate deployment in resource-constrained production environments.

5.6 Case Study

We exemplify the advantage of ProxMO through a complex ALFWorld multi-object task requiring the agent to find two remotes and place them in an armchair. Detailed multi-step trajectories are provided in Appendix E. The ProxMO-trained agent (11 steps, success) exhibits superior policy adherence, systematically exploring high-probability locations and immediately depositing objects at the target destination to secure a direct success. In stark contrast, the GPT-4o baseline (14 steps, failure) exhibits characteristic goal drift: after locating the first remote, it hallucinates an unnecessary decision to store the item in a cabinet "for safekeeping" rather than the target armchair. Although the agent later attempts error correction, this initial misalignment triggers an irreversible error cascade that prevents task completion.

This divergence reveals ProxMO's core strength, i.e., step-level proximity-based aggregation, which yields robust value estimates across semantically similar states. Specifically, states like "holding object near placement location" receive credit only when actions align with task-specific targets (armchair, not cabinet). Without this fine-grained aggregation, agents rationalize locally coherent but globally misaligned decisions through myopic heuristics. Our continuous semantic weighting prevents such fragmentary reasoning, enforcing consistency between intermediate decisions and global task objectives that is critical for long-horizon real-world tasks where early errors cascade irrecoverably.

6 Conclusion

In this paper, we propose ProxMO, a robust framework that incorporates global context into multi-turn credit assignment at both levels. Episode-level modulation adapts gradient intensity to task difficulty, attenuating noise in high-success groups while amplifying breakthroughs in low-success groups. Step-level aggregation replaces hard boundaries with continuous weighting proportional to semantic proximity, eliminating singleton degeneracy. Experiments demonstrate consistent improvements, with ablations confirming independent and synergistic contributions. ProxMO has been deployed in industrial agent training pipelines, underscoring the value of context-dependent credit assignment for multi-turn RL with LLM agents.

Limitations

This work focuses on demonstrating the effectiveness of context-aware credit assignment for multi-turn LLM agent training. While our results are consistent and robust, the scope of our investigation naturally leaves several research directions unexplored.

Our experimental validation concentrates on resource-efficient model scales, which are representative of practical industrial deployment scenarios. Extending the evaluation to significantly larger foundation models, including dense architectures and mixture-of-experts configurations, would provide a more comprehensive understanding of how context-aware credit assignment interacts with model capacity across the full spectrum. In particular, it would be valuable to investigate whether the relative contribution of episode-level versus step-level mechanisms shifts as models acquire stronger intrinsic reasoning abilities at larger scales. One hypothesis is that larger models may benefit more from step-level aggregation due to their capacity to exploit fine-grained credit signals, while smaller models may rely more heavily on episode-level modulation for stable learning. Exploring diverse architectural families beyond the Qwen series (Qwen et al., 2024) also represents a natural next step toward establishing broader generalizability across the heterogeneous model ecosystem prevalent in production environments.

The benchmarks employed in this study, ALFWorld (Shridhar et al., 2020) and WebShop (Yao et al., 2022a), represent two distinct and well-established multi-turn interaction paradigms. Nevertheless, the rapidly expanding landscape of agentic applications spans GUI navigation across operating systems (Xie et al., 2024; Feng et al., 2025a), open-ended strategic gameplay (Wang et al., 2023; Tan et al., 2024), autonomous software engineering, and complex tool orchestration with external APIs (Schick et al., 2023; Qian et al., 2025), all of which present rich opportunities for applying and adapting ProxMO's hierarchical credit assignment principles. Of particular interest are domains where episodes span hundreds of steps and involve compositional subtask structures, as these settings would further stress-test and potentially amplify the benefits of fine-grained proximity-based credit attribution. Environments with partially observable states or stochastic transitions would also serve as informative testbeds, since the distinction between noise-induced failure and genuine policy deficiency becomes more pronounced under such conditions.

Our current investigation operates within the text-based single-agent paradigm, which constitutes the dominant setting for contemporary LLM agent research. As the field advances toward multimodal agents that process visual observations (Furuta et al., 2023; Gou et al., 2024), interact with graphical interfaces (Wang et al., 2024; Zhang and Zhang, 2024), and ground language in physical environments (Li et al., 2024), adapting proximity-based mechanisms to cross-modal state representations opens an important avenue. In such multimodal settings, the notion of state proximity naturally extends beyond lexical similarity to encompass visual and structural correspondences, inviting the design of unified cross-modal similarity measures that preserve the lightweight and scalable properties central to ProxMO. How to balance representation fidelity with computational overhead in this cross-modal regime remains an open and practically significant question.

Multi-agent cooperative settings present a natural extension of ProxMO's core principles, where disentangling individual contributions from emergent collective dynamics amplifies the credit assignment challenge. Context-dependent modulation and cross-trajectory proximity comparison become particularly relevant for stable joint policy optimization. The success-rate-aware mechanism also translates naturally to multi-agent scenarios, where coordination difficulty varies with task configurations and team compositions.

Beyond empirical extensions, formal theoretical characterization—such as convergence rate analysis under our modulation and aggregation dynamics—would enrich understanding of how episode-level and step-level mechanisms jointly shape the optimization landscape. Establishing conditions under which their synergy provably accelerates learning, and analyzing how hyperparameters influence advantage geometry, would provide principled guidance for practitioners. A related direction is studying the interplay between ProxMO's advantage estimation and stabilization techniques like KL regularization, whose joint effects remain largely unexplored. These directions collectively point toward a broader agenda of context-sensitive reinforcement learning for capable real-world LLM agents.

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*.

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. *arXiv preprint arXiv:2402.14740*.

Yuhan Chen, Yuxuan Liu, Long Zhang, Pengzhi Gao, Jian Luan, and Wei Liu. 2025. Step: Success-rate-aware trajectory-efficient policy optimization. *arXiv preprint arXiv:2511.13091*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. 2025. Webevolver: Enhancing web agent self-improvement with co-evolving world model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 8959–8975. Association for Computational Linguistics.

Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, and Haolin Shi. 2026a. Placing puzzle pieces where they matter: A question augmentation framework for reinforcement learning. *Preprint*, arXiv:2604.15830.

Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, Haolin Shi, Chaowen Hu, Lu Pan, Ke Zeng, and Xunliang Cai. 2026b. How to allocate, how to learn? dynamic rollout allocation and advantage modulation for policy optimization. *arXiv preprint arXiv:2602.19208*.

Yangyi Fang and Haolin Shi. 2026. Multi-granularity intent modeling with adversarial robustness for sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 14702–14710.

Lang Feng, Weihao Tan, Zhiyi Lyu, Longtao Zheng, Haiyang Xu, Ming Yan, Fei Huang, and Bo An. 2025a. Towards efficient online tuning of vlm agents via counterfactual soft reinforcement learning. *arXiv preprint arXiv:2505.03792*.

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

Xiaoliang Fu, Jiaye Lin, Yangyi Fang, Chaowen Hu, Cong Qin, Zekai Shao, Binbin Zheng, Lu Pan, and Ke Zeng. 2026a. From $\mathbf{log}$ to $\boldsymbol{\pi}$: Taming divergence in soft clipping via bilateral decoupled decay of probability gradient weight. *arXiv preprint arXiv:2603.14389*.

Xiaoliang Fu, Jiaye Lin, Yangyi Fang, Binbin Zheng, Chaowen Hu, Zekai Shao, Cong Qin, Lu Pan, Ke Zeng, and Xunliang Cai. 2026b. Maspo: Unifying gradient utilization, probability mass, and signal reliability for robust and sample-efficient llm reasoning. *arXiv preprint arXiv:2602.17550*.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. 2023. Multimodal web navigation with instruction-finetuned foundation models. *arXiv preprint arXiv:2305.11854*.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2024. Navigating the digital world as humans do: Universal visual grounding for gui agents. *arXiv preprint arXiv:2410.05243*.

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 arXiv:2501.12948*.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Hongming Zhang, Tianqing Fang, Zhenzhong Lan, and Dong Yu. 2025. Openwebvoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 27545–27564. Association for Computational Linguistics.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, 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*.

Wouter Kool, Herke van Hoof, and Max Welling. 2019. Buy 4 reinforce samples, get a baseline for free!

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.

Mukai Li, Qingcheng Zeng, Tianqing Fang, Zhenwen Liang, Linfeng Song, Qi Liu, Haitao Mi, and Dong Yu. 2026. Verified critical step optimization for LLM agents. *CoRR*, abs/2602.03412.

Peiyang Liu, Ziqiang Cui, Di Liang, and Wei Ye. 2025a. Who stole your data? a method for detecting unauthorized rag theft. *arXiv preprint arXiv:2510.07728*.

Peiyang Liu, Sen Wang, Xi Wang, Wei Ye, and Shikun Zhang. 2021a. Quadrupletbert: An efficient model

for embedding-based large-scale retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3734–3739.

Peiyang Liu, Xi Wang, Lin Wang, Wei Ye, Xiangyu Xi, and Shikun Zhang. 2021b. Distilling knowledge from bert into simple fully connected neural networks for efficient vertical retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3965–3975.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025b. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. Toolrl: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2024. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, 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, Yang Wu, and 1 others. 2024. Deepseekmath: 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. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

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*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. 2025. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*.

Richard S Sutton, Andrew G Barto, and 1 others. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, and 1 others. 2024. Cradle: Empowering foundation agents towards general computer control. *arXiv preprint arXiv:2403.03186*.

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*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *Advances in Neural Information Processing Systems*, 37:2686–2710.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, and 1 others. 2024. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022a. 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. 2022b. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

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.

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*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Analysis of Baseline Limitations

This section provides quantitative evidence for the two fundamental limitations of existing group-based methods motivating ProxMO.

A.1 Z-Score Normalization and Success-Rate Context

We prove that GRPO's z-score normalization produces informationally asymmetric credit allocation as a function of group success rate.

Theoretical Analysis. GRPO computes advantages via z-score normalization:

$$A^E(\tau_i) = \frac{R(\tau_i) - \mu}{\sigma}, \tag{14}$$

where $\mu = \frac{1}{N}\sum_j R(\tau_j)$ and $\sigma = \sqrt{\frac{1}{N}\sum_j (R(\tau_j) - \mu)^2}$.

For binary rewards ($R \in \{0, 1\}$), the success rate $p = \mu$ directly determines the standard deviation. A group with success rate p has:

$$\sigma^2 = p(1-p). \tag{15}$$

Therefore:

$$\sigma = \sqrt{p(1-p)}. \tag{16}$$

For success outcomes ($R = 1$), the z-score is:

$$z_{\text{succ}} = \frac{1-p}{\sqrt{p(1-p)}} = \sqrt{\frac{1-p}{p}}. \tag{17}$$

For failure outcomes ($R = 0$), the z-score is:

$$z_{\text{fail}} = \frac{0-p}{\sqrt{p(1-p)}} = -\sqrt{\frac{p}{1-p}}. \tag{18}$$

Fundamental asymmetry: Inversion symmetry around $p = 0.5$. These two functions exhibit a critical symmetry: they are *inverted reciprocals* around the point $p = 0.5$. Observe that:

$$z_{\text{succ}}(p) = \sqrt{\frac{1-p}{p}}, \tag{19}$$

$$z_{\text{fail}}(p) = -\sqrt{\frac{p}{1-p}} = -\frac{1}{z_{\text{succ}}(p)}. \tag{20}$$

More precisely, if we denote $\phi(p) = \sqrt{\frac{1-p}{p}}$, then:

$$\phi(p) \cdot \phi(1-p) = \sqrt{\frac{1-p}{p}} \cdot \sqrt{\frac{p}{1-p}} = 1. \tag{21}$$

This means:

- $z_{\text{succ}}(p) = \sqrt{\frac{1-p}{p}}$ and $z_{\text{succ}}(1-p) = \sqrt{\frac{p}{1-p}} = \frac{1}{z_{\text{succ}}(p)}$
- Equivalently: $z_{\text{fail}}(p) = -z_{\text{succ}}(1-p)$

The information-value inversion problem. This inversion symmetry creates a fundamental mismatch with information density:

- When $p \to 1$ (high-success task):
  - Successes are *common* (low information), yet $z_{\text{succ}} \to 0$ (weak credit)
  - Failures are *rare* (high information), yet $|z_{\text{fail}}| \to \infty$ (severe penalty)
- When $p \to 0$ (low-success task):
  - Successes are *rare* (high information), yet $z_{\text{succ}} \to \infty$ (strong credit)
  - Failures are *common* (low information), yet $|z_{\text{fail}}| \to 0$ (weak penalty)

The core insight is that z-score normalization allocates advantage magnitudes proportional to *statistical rarity*, but this is *inversely related* to information value in the context of credit assignment. In a low-success environment ($p \approx 0$), every failure is expected (low information $\Rightarrow$ should receive weak penalty), yet statistical deviation assigns it maximal penalty. Conversely, a rare success (high information $\Rightarrow$ should receive strong reward) receives maximal credit only coincidentally when p is sufficiently small.

ProxMO's correction. ProxMO's success-rate-aware modulation (Eq. 6) explicitly *inverts* this relationship by weighting advantages as a function of p itself: amplifying signals in low-success regimes ($p \to 0$) where information density is highest, while attenuating signals in high-success regimes ($p \to 1$) where outcomes are predominantly noise.

A.2 Step-Level Grouping and Singleton Degeneracy

Existing step-level credit assignment methods partition states into discrete groups via hard thresholds (e.g., exact matching or similarity thresholds). This approach encounters a fundamental challenge in high-dimensional state spaces.

The discrete grouping dilemma. State representations in multi-turn environments encode rich, trajectory-specific contextual information (current location, observed objects, task history, action history). Two trajectories that reach the same location at different steps, with different prior trajectories or with different observed outcomes, produce semantically similar yet lexically distinct state representations.

Hard partitioning methods face a trade-off:

- **Strict criteria** (e.g., exact matching $s_t^{(i)} = s_t^{(j)}$): Ensures only truly identical states group together, but in high-dimensional spaces, exact matches become rare. Groups frequently degenerate to singletons: $|\mathcal{G}(i)| = 1$.
- **Loose criteria** (e.g., similarity threshold $\text{sim}(s_t^{(i)}, s_t^{(j)}) > \tau$): Captures semantically similar states, but indiscriminately assigns equal weight to all group members, failing to distinguish between high-proximity and low-proximity states.

The singleton problem: Empirical evidence on ALFWorld. When a state forms a singleton group ($|\mathcal{G}(i)| = 1$), baseline computation reduces to:

$$B_t^{(i)} = R_t^{(j)} \text{ where } j = i, \tag{22}$$

$$A^S(a_t^{(i)}) = R_t^{(i)} - B_t^{(i)} = 0. \tag{23}$$

This yields zero advantage, providing no learning signal. We quantify the frequency of this degeneracy on ALFWorld during training:

Across all training iterations, singleton groups persistently comprise 30-36% of trajectory steps. These steps cannot leverage within-group baselines and receive zero advantage signals, hindering credit assignment. Even non-singleton groups provide limited discrimination: combined, groups of size 2-3 account for only 25-30% of steps, leaving the majority of baseline comparisons to underperform in high-dimensional state spaces.

ProxMO's continuous proximity approach. Rather than enforcing discrete group boundaries, ProxMO computes step-level advantages through continuous proximity-based weighting (Eq. 10). All states contribute proportionally to their semantic similarity via TF-IDF:

$$B_t^{(i)} = \sum_{j \in \mathcal{G}(i)} w_{ij} R_t^{(j)}, \tag{24}$$

$$w_{ij} \propto \exp(\text{sim}(s_t^{(i)}, s_t^{(j)})/\tau). \tag{25}$$

This ensures: (1) no training signal is discarded (all states contribute), (2) proximity is continuously modeled (high-similarity states dominate, low-similarity states receive diminishing weight), and (3) credit discrimination is preserved without discrete group boundaries.

B Extended Related Works

Beyond the core topics discussed in §2, several adjacent research threads inform the broader context of this work. Efficient dense retrieval (Liu et al., 2021b,a) and retrieval robustness (Liu et al., 2025a) are foundational for agent systems that rely on tool use or grounded observation. Structured sequential modeling under adversarial conditions (Fang and Shi, 2026) further demonstrates the importance of intent-aware decision processes in multi-step settings, paralleling the credit assignment challenges ProxMO addresses in RL-based agent training.

C Complete Experimental Details

C.1 Training Configuration

In this section, we provide complete details of the experimental setup.

Model Configuration. We use Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (Qwen et al., 2024) as base models, both pretrained on diverse web corpora with instruction tuning.

Hyperparameter Settings. We adopt base configurations from GiGPO (Feng et al., 2025b): discount factor $\gamma = 0.95$, balance coefficient $\omega = 1.0$, group size $N = 8$, learning rate 10^{-6}, and clip

Training Iteration	Group Size 1	Group Size 2	Group Size 3	Group Size > 3
Iteration 40	36.2%	15.6%	11.2%	37.0%
Iteration 80	34.2%	14.2%	12.3%	39.3%
Iteration 120	30.2%	16.2%	13.6%	40.0%

Table 2: Distribution of step-level group sizes during training on ALFWorld. Singleton groups (size 1) consistently account for 30-36% of all steps, depriving these steps of meaningful baseline comparisons and credit signals.

ratio $\epsilon = 0.2$. ProxMO introduces three additional hyperparameters tuned on ALFWorld (Qwen2.5-1.5B): episode steepness $\alpha = 4.0$, episode strength $\beta = 0.1$, and step temperature $\tau = 0.1$. As demonstrated in §C.3, these values generalize well across model scales (7B) and task domains (WebShop) without further tuning.

Parameter	Value
ProxMO-specific	
Episode steepness (α)	4.0
Episode strength (β)	0.1
Step temperature (τ)	0.1
Following GiGPO (Feng et al., 2025b)	
Step discount (γ)	0.95
Balance (ω)	1.0
Group size (N)	8
Learning rate	10^{-6}
Clip ratio (ϵ)	0.2

Setting	ALFWorld	WebShop
Maximum episode length	50 steps	15 steps
Maximum prompt length	2048 tokens	4096 tokens
Training iterations	150	150

Task-Specific Settings. All experiments use 3 random seeds. We report mean and standard deviation across seeds. All methods use identical configurations for fair comparison.

C.2 Benchmark Descriptions

ALFWorld. An embodied environment designed to assess multi-step decision-making abilities of LLM agents. In each episode, the agent receives a text-based goal (e.g., "put a hot apple in the fridge") and must accomplish it through multi-turn interaction with the environment. The benchmark contains 3,827 task instances across six categories of common household activities, detailed in Table 3. Task difficulty varies dramatically across categories, with success rates ranging from 20% (Pick2) to 95% (Pick) for baseline methods, making it ideal for evaluating success-rate-aware credit assignment.

WebShop. A complex web-based interactive environment designed to test LLM agents in realistic online shopping scenarios. To complete each task, the agent must interact with a simulated HTML-based shopping website to search for, navigate to, and ultimately purchase a suitable item matching user requirements (e.g., "buy cheap wireless headphones with good reviews").

C.3 Hyperparameter Analysis

We analyze ProxMO's hyperparameter sensitivity and design principles.

Episode-Level Modulation. **Steepness** (α) controls modulation sharpness (Eq. 6). Low values fail to differentiate task difficulties; high values destabilize training through extreme weight fluctuations. Moderate values balance sensitivity and stability. **Strength** (β) determines adjustment magnitude. Small values reduce effectiveness; large values introduce gradient variance. Performance remains stable across wide ranges, indicating inherent robustness.

Step-Level Aggregation. **Temperature** (τ) governs weight concentration (Eq. 10). Low temperatures approximate exact matching, causing singleton degeneracy in high-dimensional spaces. High temperatures produce near-uniform weights, losing discrimination. Optimal values balance precision and robustness. Performance degrades gradually at extremes rather than sharply, confirming mechanism stability.

Cross-Task Consistency. Optimal configurations remain consistent across ALFWorld and WebShop despite differing task structures (embodied navigation vs. web interaction), episode lengths, and observation types. This consistency stems from ProxMO's design: both mechanisms operate on normalized quantities (success rates, L2-normalized similarities) that are scale-invariant and task-agnostic. Hyperparameters encode relative relationships rather than absolute scales, enabling transfer without per-task tuning.

Category	Description
Pick & Place (Pick)	Locate and move objects to target locations
Examine in Light (Look)	Pick up objects and examine under lamps
Clean & Place (Clean)	Clean objects and place them appropriately
Heat & Place (Heat)	Heat objects in microwaves before placing
Cool & Place (Cool)	Cool objects in fridges before placing
Pick Two & Place (Pick2)	Manipulate two objects sequentially

Table 3: ALFWorld task categories and descriptions.

Component	Details
Product catalog	Over 1.1 million real products from Amazon
User instructions	12,000 diverse instructions across categories
Observations	Rich HTML requiring semi-structured parsing
Action space	Search queries, navigation, attribute filtering
Evaluation metrics	Score (attribute matching), Success (completion)
Episode constraint	15-step limit (simulating user patience)

Table 4: WebShop environment specifications.

Deployment Implications. Wide stability ranges mean practitioners need not precisely tune hyperparameters. Cross-task consistency eliminates per-domain search. Gradual degradation provides operational safety against misspecification. These properties reduce adoption barriers in production systems where tuning is expensive and domain expertise limited. Default configurations generalize reliably across diverse settings—a critical requirement for practical deployment.

D Prompt Templates and Agent Behavior

The prompts used for LLM agents are constructed using Python-style string formatting, where placeholders enclosed in curly braces represent semantic slots. These placeholders are dynamically populated at runtime. For fair comparison, we adopt the same prompt template configurations as GiGPO.

The `<think>` `</think>` block instructs agents to perform step-by-step reasoning, promoting chain-of-thought deliberation. The `<action>` `</action>` block clearly indicates the final action decision.

E Qualitative Examples

To illustrate how ProxMO enables fine-grained reasoning and credit assignment, we present multi-step trajectories of agents trained using our method. These examples demonstrate emergent structured reasoning behaviors on challenging tasks where precise per-step decision-making is essential.

E.1 ALFWorld Example: Complex Multi-Object Task (Pick Two Objects)

We show a complete episode trajectory from ALFWorld where the agent must locate and retrieve two remote controls and place them in an armchair. This task requires strategic exploration and sequential object management across multiple steps.

This example demonstrates how ProxMO-trained agents execute structured reasoning across multi-step episodes. As shown in Figures 8–18, the agent plans the overall strategy, explores multiple locations, manages object state and placement, and adapts its search behavior based on observation results. The trajectory shows how reasoning decisions vary across the episode. Early steps (Figures 8–9) establish the search strategy and locate the first target; middle steps (Figures 10–14) evaluate locations and adapt when targets are not found; final steps (Figures 15–18) execute object placement and task completion. This reasoning structure emerges from ProxMO's two-level credit assignment: episode-level modulation provides feedback on overall task completion, while step-level proximity-based aggregation provides fine-grained feedback on individual navigation and placement decisions. Together, these mechanisms enable the policy to develop multi-step behaviors aligned with task objectives.

E.2 Baseline Comparison: Failed Episode with Execution Error

To illustrate the improvements enabled by ProxMO, we present a representative episode from GPT-4o.

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 for 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 the current step and present it within <action> </action> tags.

Figure 6: The prompt template for ALFWorld agents.

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 for 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 the current step and present it within <action> </action> tags.

Figure 7: The prompt template for WebShop agents.

Despite GPT-4o's strong reasoning capabilities, the trajectory reveals critical limitations in planning and execution consistency without fine-grained credit assignment. The agent initially pursues a reasonable strategy but compounds early inefficiencies with a catastrophic placement error: after locating the first remote control, the agent places it in the wrong location (a storage area rather than the target armchair), fundamentally compromising task completion. This error exemplifies how agents without step-level proximity-based credit assignment lack incentive to maintain consistency between declarative reasoning and actionable execution across extended horizons.

This trajectory reveals a critical failure mode absent from standard metrics but crucial for deployment: despite successfully discovering both remote controls and demonstrating reasonable search patterns, GPT-4o fails the task through a goal misalignment error. After acquiring the first remote, the agent decides to store it temporarily in a cabinet (Step 8) rather than directly placing it in the target armchair as specified in the task. The agent later realizes this mistake (Step 12) but cannot recover within the episode horizon. This exemplifies a fundamental limitation without fine-grained step-level credit assignment: the agent lacks incentive to maintain explicit consistency between the global task objective ("place remotes in armchair") and intermediate action decisions ("store in cabinet"). Each action generates a local reward signal that might be individually rational given myopic heuristics ("a cabinet is a safe storage location"), but without proximity-based state-level credit aggregation, the agent fails to recognize and penalize the divergence from global task semantics until recovery is no longer feasible.

ProxMO's mechanisms directly address this failure mode through two complementary pathways. Episode-level modulation with success-rate awareness would recognize the ultimate task failure and

Environment (Step 1): Initial State

You are in the middle of a room. Looking quickly around you, you see a drawer 15, a drawer 19, a drawer 7, a armchair 1, a drawer 3, a drawer 16, a drawer 2, a drawer 17, a sidetable 3, a sidetable 5, a drawer 4, a drawer 20, a sofa 1, a drawer 13, a drawer 12, a drawer 9, a drawer 21, a sidetable 4, a sidetable 2, a drawer 5, a drawer 11, a drawer 1, a garbagecan 1, a dresser 1, a cabinet 1, a drawer 8, a drawer 18, a sidetable 1, a drawer 14, a drawer 6, and a drawer 10.
Your task is to: find two remotecontrol and put them in armchair.

Agent (Step 1): Initial Strategy

<think>
1. I need to find two remote controls and place them in armchair 1.
2. The room contains many drawers, side tables, a sofa, and a dresser—common places to find remote controls.
3. I should systematically search the most likely locations (side tables, sofa, dresser) before checking drawers.
4. Once I find a remote control, I'll take it to the armchair and place it there, then repeat for the second one.
5. Strategy: Start with side tables and the sofa as they're common places for remote controls.
</think>
<action>go to sidetable 2</action>

Figure 8: Step 1: Task understanding and exploration strategy.

provide strong corrective credit to the entire trajectory, teaching the agent to prioritize destination consistency. Step-level proximity-based soft aggregation would compute value estimates that compare semantically similar states: states semantically near "arriving at placement location with object" would receive credit only when the action correctly aligns with the task-specific target location. This semantic alignment mechanism ensures agents learn robust task execution patterns rather than fragmentary heuristics. The ProxMO-trained agent (Figures 8–18) maintains consistency throughout, successfully completing the task within 11 steps, demonstrating that fine-grained credit assignment enables not just efficiency but also correctness in multi-step execution.

Environment (Step 2): First Location

You arrive at loc 34. On the sidetable 2, you see a remotecontrol 1.

Agent (Step 2): First Object Found

<think>
1. Excellent! I found the first remote control on the sidetable 2.
2. I should take it and transport it to the armchair 1.
3. Then I'll need to continue searching for the second remote control.
</think>
<action>take remotecontrol 1 from sidetable 2</action>

Figure 9: Step 2: First remote control discovered and acquired.

Environment (Step 3): Object in Hand

You pick up the remotecontrol 1 from the sidetable 2.

Agent (Step 3): Transport to Destination

<think>
1. I have successfully picked up the first remote control.
2. Now I need to navigate to armchair 1 to place it there.
3. The armchair is the target location for both remote controls.
</think>
<action>go to armchair 1</action>

Figure 10: Step 3: Navigating with first object to destination.

Environment (Step 4): At Armchair

You arrive at loc 1. On the armchair 1, you see nothing.

Agent (Step 4): Place First Object

<think>
1. I've reached the armchair 1 and it's currently empty.
2. I should place the first remote control here.
3. After this, I need to search for the second remote control.
</think>
<action>put remotecontrol 1 in/on armchair 1</action>

Figure 11: Step 4: Placing first remote control in armchair.

Environment (Step 5): First Object Placed

You put the remotecontrol 1 in/on the armchair 1.

Agent (Step 5): Continue Searching

<think>
1. First remote control successfully placed in the armchair.
2. Now I need to find the second remote control.
3. Let me check the dresser 1—it's another common location for electronics like remote controls.
</think>
<action>go to dresser 1</action>

Figure 12: Step 5: First object stored, searching for second remote.

Environment (Step 6): Dresser Exploration

You arrive at loc 18. On the dresser 1, you see a creditcard 1, a television 1, a laptop 1, a book 1, a statue 2, and a cellphone 2.

Agent (Step 6): No Remote Here, Continue Search

<think>
1. The dresser has electronics but no remote control.
2. I need to check other locations—sidetable 5, the sofa, and other areas.
3. Let me try sidetable 5 next.
</think>
<action>go to sidetable 5</action>

Figure 13: Step 6: Dresser checked, no remote control found.

Environment (Step 7): Sidetable 5 Exploration

You arrive at loc 17. On the sidetable 5, you see a box 1, and a creditcard 2.

Agent (Step 7): No Remote at Sidetable 5

<think>
1. No remote control at sidetable 5.
2. The sofa 1 is a likely place to find a remote control.
3. Let me go to the sofa next.
</think>
<action>go to sofa 1</action>

Figure 14: Step 7: Sidetable 5 checked, moving to sofa.

Environment (Step 8): Sofa Discovery

You arrive at loc 2. On the sofa 1, you see a newspaper 1, a pillow 1, and a remotecontrol 2.

Agent (Step 8): Second Remote Found!

<think>
1. Perfect! I found the second remote control on the sofa 1.
2. I should take it immediately.
3. Then transport it to the armchair where the first remote control is already placed.
</think>
<action>take remotecontrol 2 from sofa 1</action>

Figure 15: Step 8: Second remote control located on sofa.

Environment (Step 9): Second Object Acquired

You pick up the remotecontrol 2 from the sofa 1.

Agent (Step 9): Return to Armchair

<think>
1. I've successfully picked up the second remote control.
2. Now I need to navigate back to armchair 1 to place it there.
3. This will complete the task.
</think>
<action>go to armchair 1</action>

Figure 16: Step 9: Second object acquired, returning to armchair.

Environment (Step 10): Final Placement

You arrive at loc 1. On the armchair 1, you see a remotecontrol 1.

Agent (Step 10): Task Completion

<think>
1. I've returned to armchair 1 where the first remote control is already placed.
2. Now I'll place the second remote control here to complete the task.
3. This fulfills the goal of finding two remote controls and placing them in the armchair.
</think>
<action>put remotecontrol 2 in/on armchair 1</action>

Figure 17: Step 10: Final remote control placed, task complete.

Environment (Step 11): Success!

You put the remotecontrol 2 in/on the armchair 1. **You won!**

Figure 18: Step 11: Task successfully completed.

Environment (Step 1): Initial State

You are in the middle of a room. Looking quickly around you, you see a drawer 15, a drawer 19, a drawer 7, a armchair 1, a drawer 3, a drawer 16, a drawer 2, a drawer 17, a sidetable 3, a sidetable 5, a drawer 4, a drawer 20, a sofa 1, a drawer 13, a drawer 12, a drawer 9, a drawer 21, a sidetable 4, a sidetable 2, a drawer 5, a drawer 11, a drawer 1, a garbagecan 1, a dresser 1, a cabinet 1, a drawer 8, a drawer 18, a sidetable 1, a drawer 14, a drawer 6, and a drawer 10.
Your task is to: find two remotecontrol and put them in armchair.

Agent (Step 1): Initial Planning

<think>
1. I need to find two remote controls and place them in armchair 1.
2. The room contains many storage locations. I should explore systematically.
3. Let me start by checking the drawers—they often contain various household items.
</think>
<action>go to drawer 1</action>

Figure 19: Failed Step 1: Initial search direction—choosing drawers.

Environment (Step 2): Drawer 1 Exploration

You arrive at loc 45. The drawer 1 is closed.

Agent (Step 2): Opening Drawer

<think>
1. The drawer is closed. I need to open it to see if there's a remote control inside.
</think>
<action>open drawer 1</action>

Figure 20: Failed Step 2: Attempting to open drawer.

Environment (Step 3): Drawer 1 Contents

You open the drawer 1. The drawer 1 is open. In it, you see a pencil 1 and a notepad 1.

Agent (Step 3): No Target in Drawer 1

<think>
1. No remote control in this drawer. Let me continue checking other drawers.
</think>
<action>go to drawer 2</action>

Figure 21: Failed Step 3: Dead-end drawer—wasted exploration.

Environment (Step 4): Drawer 2 Exploration

You arrive at loc 46. The drawer 2 is closed.

Agent (Step 4): Opening Drawer 2

<think>
1. Another drawer to check.
</think>
<action>open drawer 2</action>

Figure 22: Failed Step 4: Checking another drawer.

Environment (Step 5): Drawer 2 Contents

You open the drawer 2. The drawer 2 is open. In it, you see a paperclip 1, a stapler 1, and some files 1.

Agent (Step 5): Another Dead-End

<think>
1. No remote control here either. I'm wasting effort on drawers. Let me reconsider—remote controls are typically found on furniture surfaces like sofas or side tables.
</think>
<action>go to sidetable 2</action>

Figure 23: Failed Step 5: Realization and strategy correction.

Environment (Step 6): Sidetable 2

You arrive at loc 34. On the sidetable 2, you see a remotecontrol 1.

Agent (Step 6): First Remote Found

<think>
1. Finally! Found the first remote control on the sidetable.
2. I should take it and place it in the armchair.
</think>
<action>take remotecontrol 1 from sidetable 2</action>

Figure 24: Failed Step 6: First remote discovered after dead-end exploration.

Environment (Step 7): Object Acquired

You pick up the remotecontrol 1 from the sidetable 2.

Agent (Step 7): Decision on Storage

<think>
1. I have the first remote control.
2. I should find a good place to store it temporarily while I search for the second one. A cabinet would be a secure storage location.
</think>
<action>go to cabinet 1</action>

Figure 25: Failed Step 7: Strategic decision to temporarily store first remote.

Environment (Step 8): At Cabinet

You arrive at loc 40. At cabinet 1, you see some items inside.

Agent (Step 8): Execution Error—Misaligned Goal

<think>
1. Perfect! The cabinet is a secure location. I'll store the first remote here temporarily.
2. After finding the second remote, I'll need to gather both and place them in the armchair.
</think>
<action>put remotecontrol 1 in/on cabinet 1</action>

Figure 26: Failed Step 8: Critical error—storing remote in cabinet instead of placement location.

Environment (Step 9): Error Consequence

You put the remotecontrol 1 in/on the cabinet 1.

Agent (Step 9): Search for Second Remote

<think>
1. First remote is stored. Now searching for the second one.
2. Let me check the sofa next.
</think>
<action>go to sofa 1</action>

Figure 27: Failed Step 9: Beginning search for second remote.

Environment (Step 10): Sofa Check

You arrive at loc 2. On the sofa 1, you see a newspaper 1, a pillow 1, and a remotecontrol 2.

Agent (Step 10): Second Remote Found

<think>
1. Great! Found the second remote control on the sofa.
2. I should take it and place it in the armchair.
</think>
<action>take remotecontrol 2 from sofa 1</action>

Figure 28: Failed Step 10: Second remote located.

Environment (Step 11): Second Object Acquired

You pick up the remotecontrol 2 from the sofa 1.

Agent (Step 11): Return to Armchair

<think>
1. I have the second remote. Now I'll go to the armchair to place both remotes there.
</think>
<action>go to armchair 1</action>

Figure 29: Failed Step 11: Traveling to armchair with second remote.

Environment (Step 12): At Armchair

You arrive at loc 1. On the armchair 1, you see nothing.

Agent (Step 12): Realization of Mistake

<think>
1. Wait—the armchair is empty. Where did I put the first remote control?
2. I realize now I mistakenly placed it in the cabinet instead of here.
3. I need to retrieve it from the cabinet and bring both remotes to the armchair.
</think>
<action>go to cabinet 1</action>

Figure 30: Failed Step 12: Realizing mistake, attempting to retrieve first remote from cabinet.

Environment (Step 13): At Cabinet

You arrive at loc 40. At cabinet 1, you see remotecontrol 1 inside.

Agent (Step 13): Recovery Attempt

<think>
1. Found the first remote control. I'm currently holding remotecontrol 2.
2. I need to take remotecontrol 1 and then go back to the armchair.
</think>
<action>take remotecontrol 1 from cabinet 1</action>

Figure 31: Failed Step 13: Attempting to retrieve first remote from cabinet.

Environment (Step 14): Task Failed

You pick up remotecontrol 1 from cabinet 1. After multiple failed attempts to properly place both remotes in the armchair, the task cannot be completed. You have remotecontrol 1 in hand and remotecontrol 2 remains in the armchair. The initial misplacement decision has made the task unrecoverable. **Task Failed—Execution Error.**

Figure 32: Failed Step 14: Task failure due to cascading effects of initial misalignment decision.