



TIDE: Trajectory-based Diagnostic Evaluation of Test-Time Improvement in LLM Agents

Hang Yan^{1,2,3*} Xinyu Che^{1,3*} Fangzhi Xu^{1,2,*†} Qiushi Sun⁴
Zichen Ding⁵ Kanzhi Cheng⁶ Jian Zhang^{1,2} Tao Qin¹ Jun Liu^{1,2,3†} Qika Lin^{7†}

¹School of Computer Science and Technology, Xi'an Jiaotong University, China

²National Engineering Research Center for Visual Information and Applications

³Shaanxi Province Key Laboratory of Big Data Knowledge Engineering

⁴The University of Hong Kong ⁵Shanghai AI Laboratory

⁶Nanjing University ⁷National University of Singapore

hyan@stu.xjtu.edu.cn fangzhixu98@gmail.com

liukeen@xjtu.edu.cn qikalina@foxmail.com

* means equal contribution † denotes corresponding authors

Abstract

Recent advances in autonomous LLM agents demonstrate their ability to improve performance through iterative interaction with the environment. We define this paradigm as **Test-Time Improvement (TTI)**. However, the mechanisms under how and why TTI succeed or fail remain poorly understood, and existing evaluation metrics fail to capture their task optimization efficiency, behavior adaptation after erroneous actions, and the specific utility of working memory. To address these gaps, we propose **Test-time Improvement Diagnostic Evaluation (TIDE)**, a lightweight, agent-agnostic and environment-agnostic framework that decomposes TTI into three comprehensive and interconnected dimensions. The framework measures (1) the overall temporal dynamics of task completion and (2) identifies whether performance is constrained by recursive looping behaviors or (3) by burdensome accumulated memory. Through extensive experiments, **TIDE** highlights that improving agent performance calls for explicitly optimizing the interaction dynamics between the agent and the environment.

1 Introduction

Through active interaction with the environment, LLM agents (Yao et al., 2023; Xi et al., 2025) demonstrate substantial potential for handling complex real-world tasks. This capability has enabled a wide range of practical applications, including coding agents (Wang et al., 2024; Chen et al., 2025) and GUI agents (Qin et al., 2025; Wu et al., 2024).

Due to the complexity and unpredictability of real-world environments (Jimenez et al., 2024; Zhou et al., 2024; Wang et al., 2025a), relying solely on internal reasoning is often insufficient

for optimal task completion (Parisi et al., 2024). This challenge necessitates a critical adaptive capacity, in which the agent accumulates experience through continuous interaction to iteratively rectify its actions. We define this dynamic improvement process as **Test-Time Improvement (TTI)**. Despite the critical role of TTI for agent autonomy, a rigorous understanding of how such improvement unfolds, stagnates, or deteriorates remains a missing piece. As shown in Figure 1(c), we argue that TTI reflects the interplay of three fundamental aspects of interactive optimization, structured into three complementary research questions as follows.

Firstly, beyond eventual success, a competent agent should improve efficiently and progressively through interaction with the environment. However, widely adopted static metrics, such as Success Rate (SR) (Wang et al., 2024; Xu et al., 2025b), collapse the informative trajectory into a single binary outcome, treating an efficient one-step success as equivalent to a delayed success after extensive exploration. Based on this observation, we propose RQ I: *How to quantify the optimization efficiency of an agent's performance evolution?*

Secondly, errors are inevitable in complex environments, and effective agents must demonstrate behavior adaptation that corrects their actions by learning from failures. However, existing metrics, such as the number of interaction turns (Zhang et al., 2025; Pitre et al., 2025), are largely content-agnostic, conflating genuine corrective behavior adaptation with repetitive failure actions. As a result, an agent may appear active while repeatedly executing ineffective strategies. This prompts RQ II: *How can we formalize the boundary between behavior adaptation and recursive failure?*

Thirdly, interaction inevitably accumulates work-

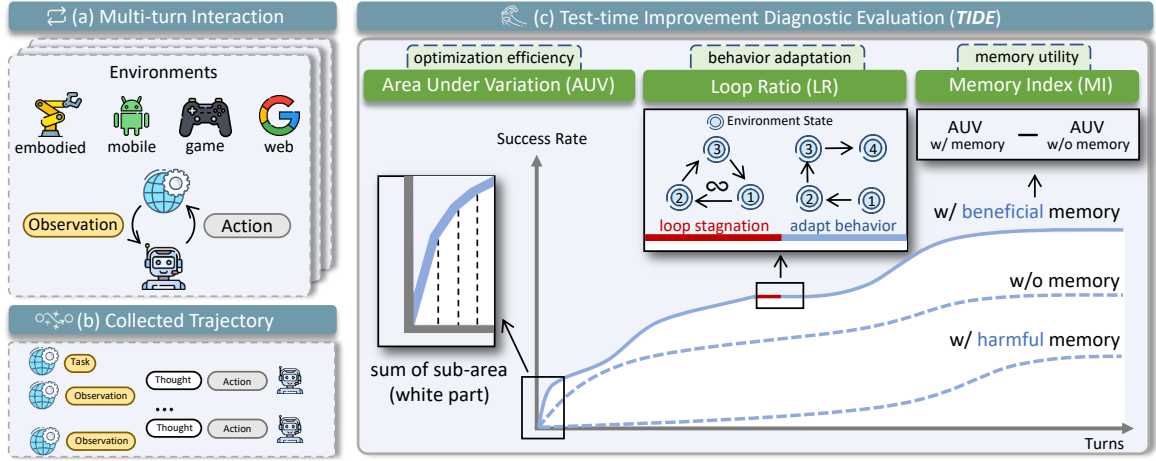


Figure 1: Overview of our trajectory-based diagnostic evaluation framework. (a) An agent completes tasks through multi-turn interaction. (b) Interaction trajectories are collected for diagnostic analysis. (c) **TIDE** provides a unified and interconnected diagnosis of TTI trajectories via three complementary metrics. AUV quantifies optimization efficiency by aggregating sub-areas along the trajectory; LR distinguishes loop-induced stagnation from behavioral adaptation; MI measures performance sensitivity to working-memory availability.

ing memory that may contain both useful experiences and misleading noise. While longer contexts with more information are often assumed to be beneficial, their actual impact on decision quality remains unclear, as existing analyses (Chhikara et al., 2025; Qiu et al., 2025) typically conflate memory effects with other confounding factors, such as model scale or interaction length. Accordingly, we propose RQ III: *How to quantify the utility of accumulated interaction memories to agent performance?*

To systematically address the research questions, we introduce Test-time Improvement Diagnostic Evaluation (**TIDE**), a lightweight diagnostic framework for analyzing how LLM agents evolve during interaction. **TIDE** operates on trajectory logs for AUV and LR, while MI requires a simple memory-removal evaluation protocol. Specifically, **TIDE** formalizes TTI via three metrics corresponding to the three above-mentioned RQs. AUV (Area Under Variation) quantifies optimization efficiency by capturing how quickly and steadily an agent succeeds over interaction. LR (Loop Ratio) measures behavior stagnation by identifying loop patterns, enabling the distinction between effective behavior adaptation and recursive failure. MI (Memory Index) isolates the utility of accumulated interaction history, quantifying how working memory contributes to performance. Through comprehensive experiments, we highlight the need to optimize the dynamics of agent–environment interaction.

We highlight our main contributions as follows:

(1) **Conceptualization of Test-Time Improve-**

ment for LLM agents. We first formalize Test-time Improvement as a multi-dimensional, interaction-driven process beyond accuracy.

(2) **Diagnostic Evaluation Framework.** We introduce **TIDE**, a lightweight, agent-agnostic, and environment-agnostic diagnostic framework that decomposes TTI into optimization efficiency, behavior adaptation, and memory utility, operating on recorded trajectories in a post-hoc manner.

(3) **Empirical Diagnosis Across Environments and Models.** Extensive experiments reveal failure modes and bottlenecks that are invisible to existing metrics, providing actionable insights for optimizing interactive LLM agents.

2 Preliminary

We formalize the multi-turn interaction between the LLM agent and the environment as a Partially Observable Markov Decision Process (POMDP), defined by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{F}, \mathcal{R}, g \rangle$. $\mathcal{S}$ represents the latent state space. $\mathcal{A}$ denotes the action space. $\mathcal{O}$ is the observation space. The transition function $\mathcal{F} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. The interaction process yields a rollout $\tau = [o_0, a_0, o_1, a_1, \dots, o_T]$, where $a \in \mathcal{A}$ and $o \in \mathcal{O}$. $\mathcal{R}(\tau) \rightarrow \{0, 1\}$ indicates whether the trajectory τ reaches the goal g . In our main experiments, we adopt five benchmarks: BlocksWorld, FrozenLake, Sudoku, Alf-World, Webshop. Additionally, we also include several GUI environments for post-hoc analysis. Detailed setup is in Appendix C.1. During task evaluation, we include the average success rate (SR) for comparison.

Model	BlocksWorld		FrozenLake		Sudoku		AIfWorld		WebShop	
	SR↑	AUV↑	SR↑	AUV↑	SR↑	AUV↑	SR↑	AUV↑	SR↑	AUV↑
Non-thinking Model										
Qwen3-4B-Instruct	40.0	30.8	59.0	46.1	29.0	19.3	52.1	38.3	15.0	9.6
Qwen3-30B-A3B-Instruct	64.0	45.4	61.0	49.0	77.0	48.8	62.1	48.1	20.8	13.7
Llama-3.1-8B-Instruct	32.0	21.9	12.0	8.4	4.0	2.1	16.4	12.8	20.8	13.1
Llama-3.3-70B-Instruct	96.0	68.2	44.0	34.7	40.0	24.9	59.3	46.7	26.8	17.6
GLM-4-9B-Chat	23.0	18.2	10.0	7.2	4.0	2.0	25.7	21.8	16.4	11.5
GLM-4-32B-0414	83.0	57.5	67.0	49.9	55.0	34.4	37.9	29.0	25.4	17.2
Mistral-7B-Instruct	7.0	5.0	6.0	4.6	1.0	0.4	8.6	7.6	6.6	4.4
Ministral-3-14B-Instruct	72.0	39.0	56.0	35.8	18.0	13.3	22.1	15.8	15.2	5.4
Phi-4	56.0	41.2	66.0	50.8	33.0	21.4	25.0	18.6	18.8	12.2
DeepSeek-V3.2	98.0	71.1	97.0	73.1	93.0	56.8	80.7	59.0	41.0	24.8
Gemini 2.5 Flash	99.0	72.0	97.0	70.7	97.0	60.4	78.6	58.1	35.0	21.6
Thinking Model										
Qwen3-4B-Thinking	77.0	47.9	91.0	66.7	55.0	30.7	57.1	40.9	12.2	8.2
Qwen3-30B-A3B-Thinking	98.0	69.8	93.0	68.0	95.0	55.6	66.4	50.7	21.0	14.6
Phi-4-reasoning	68.0	51.3	62.0	48.8	51.0	37.4	15.7	9.8	22.6	14.3
gpt-oss-120b	100.0	73.7	99.0	73.8	97.0	60.2	50.7	29.0	26.2	11.1
DeepSeek-R1	100.0	74.9	96.0	72.6	95.0	59.4	83.6	64.1	38.4	24.1
Gemini 2.5 Pro	100.0	75.2	100.0	75.0	99.0	60.2	80.7	62.9	43.2	27.1

Table 1: We report success rate (SR) and area under variation (AUV). ↑ means a higher value is better. Colored model name represents proprietary models. Best values are in bold.

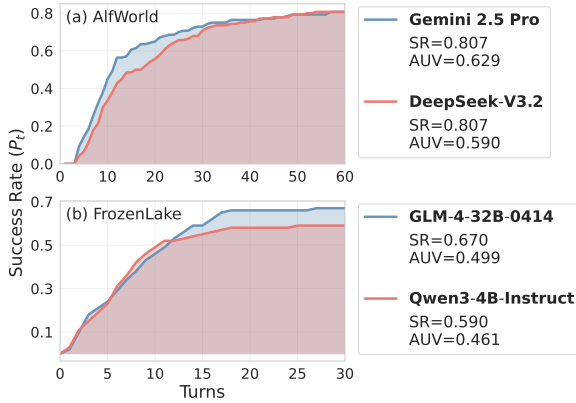


Figure 2: SR curves on two environments. We report both AUV and final SR. We reveal that SR ignores critical information of TTI, and evaluation within limited turns is insufficient.

3 Breakdown Test-Time Improvement

In this section, we analyze TTI through three diagnostic indicators with different input requirements: AUV uses step-wise success outcomes, LR additionally requires a task-appropriate state/action equivalence function, and MI requires a simple memory-removal input intervention.

3.1 Optimization Efficiency Diagnostic: Modeling Temporal Dynamics

To quantify the optimization efficiency of an agent’s performance evolution, we model agent

performance as a discrete-time evolution process over interaction steps. Instead of assessing an interaction trajectory through a terminal outcome, we characterize TTI progress with a variation curve P_t , which denotes the cumulative proportion of tasks successfully solved within the first t interaction turns. Tracking P_t over t explicitly captures how quickly and steadily an agent converts additional interaction budget into task success, enabling a temporal analysis of the task completion dynamics.

Building upon P_t , we introduce a holistic evaluation metric, denoted as Area Under Variation (AUV), which is defined as the integral area under the performance variation curve. Formally, let $[0, t_{max}]$ denote the domain-specific evaluation window, where t_{max} is the experimental evaluation horizon in each environment. Detailed configuration for t_{max} is in Appendix C.4. AUV is calculated within this evaluation window as follows:

$$\text{AUV} = \frac{1}{t_{max}} \sum_{t=0}^{t_{max}-1} \frac{P_t + P_{t+1}}{2}, \quad (1)$$

where the trapezoidal sum $\frac{P_t + P_{t+1}}{2}$ accounts for the incremental performance gain between successive steps. The denominator t_{max} serves as a normalization constant, ensuring that the metric is bounded between 0 and 1.

Unlike previous works (Taylor and Stone, 2009; Brys et al., 2015) measuring training-phase conver-

gence, AUV specifically quantifies the optimization efficiency at test-time. Additionally, theoretical proofs in Appendix A indicate that AUV complements final SR by distinguishing trajectories with identical or similar terminal success but different temporal efficiency. For comparison, we report AUV along with the corresponding SR in Table 1, and reveal the following findings:

AUV Quantifies Temporal Efficiency. As illustrated in Figure 2(a), while *DeepSeek-V3.2* and *Gemini 2.5 Pro* converge to nearly identical final success rates 0.807 in AlfWorld, they exhibit a notable divergence in AUV. Specifically, *Gemini 2.5 Pro* achieves a higher AUV 0.629, indicating early-stage TTI efficiency. This indicates that AUV successfully captures the distinguishing temporal efficiency of different agents. Notably, in simple environments where SR saturates at 1 for different models, such as BlocksWorld, AUV continuously demonstrates unique value. AUV captures a fine-grained distinction of optimization efficiency, which cannot be discovered by SR. Moreover, to better illustrate how success rates evolve with the number of interaction turns, we further report the turns required for each model to achieve a 50% success rate across all environments in Appendix B.3.

AUV Rewards Sustained Convergence Dynamics. As illustrated in Figure 2(b), although *GLM-4-32B-0414* exhibits a comparable or even lower SR in the early stages compared to *Qwen3-4B-Instruct*, it achieves significantly higher performance in later turns, demonstrating a superior TTI capability. Crucially, the substantial marginal gains realized in the later stages result in *GLM-4-32B-0414* achieving a superior overall AUV 0.499, confirming that AUV effectively rewards agents that possess the continuous improvement capability. Moreover, although SR and AUV reveal similar evaluation results in this case (*GLM-4-32B-0414* exhibits both high AUV and SR), we argue that AUV is not equivalent to SR. More experimental details to validate this can be found in Appendix B.2.

TTI Success Depends on Agent-Environment Match. TTI efficacy is not a universal capability but is subject to agent-environment constraints. For instance, *Llama3.3-70B-Instruct* surpasses *Qwen3-30B-A3B-Instruct* in BlocksWorld, a trend that sharply reverses in FrozenLake and Sudoku. This highlights the imperative for holistic evaluation to accurately verify a model’s generalized TTI capa-

Model	BW	FL	Su	Alf	WS
Non-thinking Model					
Qwen3-4B-Instruct	15.8	32.0	22.5	0.3	36.7
Qwen3-30B-A3B-Instruct	1.0	37.7	4.2	0.5	5.7
Llama-3.1-8B-Instruct	3.2	5.4	14.7	0.9	2.1
Llama-3.3-70B-Instruct	0.0	9.5	2.7	0.0	1.6
Glm-4-9B-Chat	7.6	16.7	9.5	0.8	1.5
Glm-4-32B-0414	1.2	5.1	5.8	0.0	0.1
Mistral-7B-Instruct	51.0	63.3	15.7	10.4	17.5
Ministral-3-14B-Instruct	6.0	4.3	6.9	2.7	4.0
Phi-4	12.3	24.3	21.0	0.2	2.6
DeepSeek-V3.2	0.0	0.0	0.1	0.0	0.0
Gemini 2.5 Flash	0.0	1.0	0.6	0.3	0.2
Thinking Model					
Qwen3-4B-Thinking	28.5	9.8	34.3	4.4	4.5
Qwen3-30B-A3B-Thinking	2.0	5.3	2.2	1.3	0.7
Phi-4-reasoning	5.2	1.4	1.0	1.4	35.7
gpt-oss-120b	1.0	0.7	0.0	9.5	9.2
DeepSeek-R1	0.0	0.0	0.0	0.0	0.1
Gemini 2.5 Pro	0.0	0.2	0.1	0.0	0.0

Table 2: Loop Ratios, which are shown in %. **Colored model name** represents proprietary models. BW, FL, Su, Alf, WS refer to BlocksWorld, FrozenLake, Sudoku, Alfworld, WebShop, respectively.

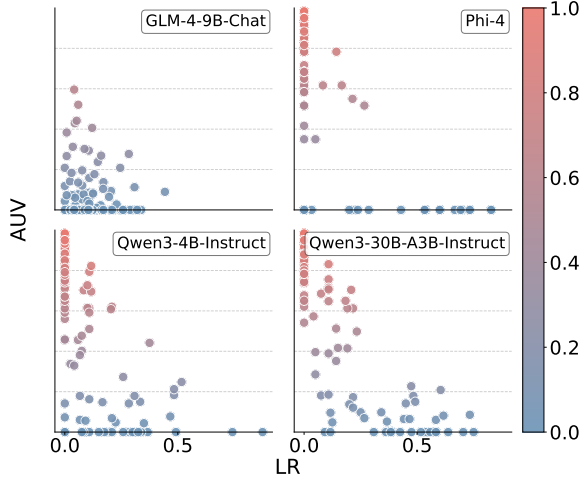
bility, rather than its specialized performance.

Takeaway 1: Current metrics overlook the temporal dynamics of TTI. Moreover, TTI efficacy depends on agent-environment match rather than being purely intrinsic.

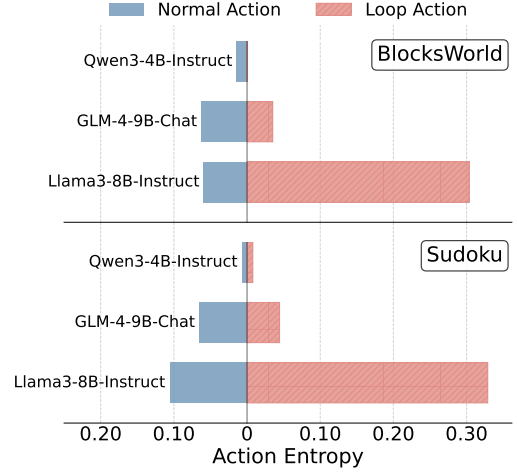
3.2 Behavior Adaptation Diagnostic: Characterizing Recursive Loops

To formalize the boundary between behavior adaptation and recursive failure, we introduce an auto-detection algorithm through interpreting an agent’s interaction trajectory as a path over the latent environment state space (described in Sec 2). From this perspective, actions correspond to directed transitions between states, allowing the trajectory to be analyzed as a graph-structured process. Then we rigorously distinguish adaptive exploration from degenerate repetition by identifying recursive cycle structures in the trajectory-based graph, which serve as a signal of adaptation failure.

We define the cycle as $l_{ij} = [s_i, a_i, \dots, a_{j-1}, s_j]$, where $s_i = s_j$ and $i \neq j$. The environment state departs from a state node s_i and eventually returns to the exact same state s_j , yielding no goal progress. The set of these cycle units is denoted as $\mathcal{L}_{cycle}$. Notably, we impose a non-recursive constraint that each cycle does not contain any nested sub-cycles to avoid ambiguous decomposition. Moreover, a single action resulting in no change of the environment state is also denoted as a cycle:



(a) AUV against LR



(b) Average action entropy

Figure 3: (a) AUV against LR across 4 models, revealing the relationship between LR and AUV for each task in FrozenLake. Different colors represent different values of AUV. (b) Average action entropy, in which a lower value indicates that the model is more confident.

$$\mathcal{L}_{cycle} = \{l_{ij} \mid s_j = s_i, i < j, \text{ and } \forall i \leq p < q < j, s_p \neq s_q\}. \quad (2)$$

Once encountering a cycle, the agent can utilize it to traverse a new path or repeat the previous cycle without task progress. The latter loop behavior significantly diminishes exploration without task progress. We define a subset of $\mathcal{L}_{cycle}$ as $\mathcal{L}_{loop}$, where the agent repeats the previous cycle and the two cycles should occur consecutively:

$$\mathcal{L}_{loop} = \{l_{jk} \in \mathcal{L}_{cycle} \mid l_{ij} = l_{jk}, i < j < k, l_{ij} \in \mathcal{L}_{cycle}\}. \quad (3)$$

We then quantify this loop behavior by Loop Ratio (LR) as the proportion of these redundant loop actions relative to the total actions:

$$LR = \frac{\sum_{l_{ij} \in \mathcal{L}_{loop}} j - i}{\text{Total Actions}}, \quad (4)$$

where a lower LR suggests less stagnation, signifying that the agent actively attempts to alter its strategy upon failure rather than succumbing to recursive failures.

More implementation details and the auto-detection algorithm can be found in Appendix C.3. We demonstrate the LR results in Table 2 and reveal the following findings:

Loop is Widely Observed in Current Models. A majority of the evaluated LLM agents exhibit

remarkably high LR. This indicates that instead of adapting their behaviors based on negative environment feedback, the models tend to persist in a state of recursive failure, repeatedly attempting the same erroneous actions. This vulnerability is particularly pronounced in smaller or less aligned architectures. For instance, *Qwen3-4B-Instruct* records a LR of 32.0% in the FrozenLake environment. Such rigidity suggests that these agents suffer from a deficiency in valid behavior adaptation. Notably, in Appendix B.10, we further demonstrate that agents exhibit significant loops even in open-ended action spaces (such as GUI environments), where the probability of re-executing identical failed action sequences is intuitively much lower.

High LR Indicates Suboptimal TTI. To investigate the extent to which the loop phenomenon constrains performance, we visualize the relationship between LR and AUV in Figure 3(a). Every data point represents a single task in each environment, which is repeated 8 times to generate a reasonable AUV value. We observe a statistically significant inverse relationship across four models, where a high LR is associated with a low AUV. This suggests that loops are strongly associated with suboptimal TTI. Notably, as indicated by the data points in the lower-left corner of each subfigure, a low Loop Ratio does not strictly guarantee a high AUV. Thus, we conclude that minimizing loops is strongly associated with high TTI capacity.

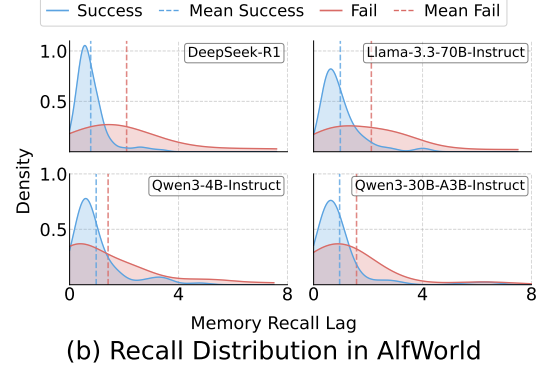
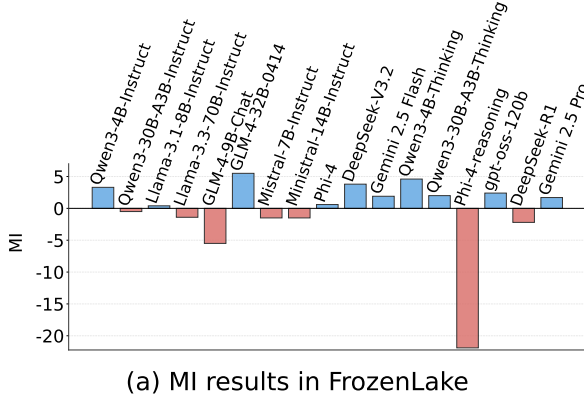


Figure 4: (a) MI results in FrozenLake, revealing frequent performance degradation when accumulated working memory is available under our ablation protocol. More MI results can be found in Appendix B.5. (b) Recall Distribution in AlfWorld. We categorize trajectories into successful and failed group.

Loop Phenomenon Analysis. To understand the inner cause of loops, we further analyze the entropy on loop actions against non-loop actions. In Figure 3(b), *Qwen3-4B-Instruct* and *GLM-4-9B-Chat*, the remarkably low entropy during stagnation phases indicates that the agent is confidently fixed on erroneous paths. Conversely, *Llama3-8B-Instruct* demonstrate low entropy along with low LR. This comparison indicates that most loops stem from over-confidence. Moreover, from Table 2, we can observe that within the same model family, larger models consistently demonstrate significantly lower LR, providing a straightforward way to mitigate loop phenomenon. Furthermore, we provide manual qualitative analysis of fail trajectories with loop phenomenon in Appendix B.4.

Takeaway 2: Loop-induced stagnation appears frequently in evaluated models, which is associated with overconfidence. High LR is consistently associated with lower AUV, making LR a useful diagnostic signal for one common source of inefficient TTI.

3.3 Memory Utility Diagnostic: Assessing Working Memory Contribution

To quantify and analyze the utility of interaction memories to agent performance, we adopt an ablation evaluation protocol that compares agent performance w/ and w/o access to the accumulated interaction trajectory. By keeping the agent and environment fixed and varying only the availability of past trajectories, this setup enables a direct quantification of the specific contribution of working memory to optimization dynamics.

Building on this setup, we define Memory Index (MI) as a quantitative measure of this effect, computed as the performance gap between w/ memory and w/o memory configurations, thereby capturing the specific contribution of the expanding interaction memory:

$$MI = AUV_{w/memory} - AUV_{w/o memory}, \quad (5)$$

where $AUV_{w/memory}$ denotes the performance using complete working memory, and $AUV_{w/o memory}$ represents the performance when the agent is restricted to only the task description and immediate observation, without access to earlier trajectory. A positive MI reflects a scenario in which working memory relates positively to TTI, while a negative MI indicates a harmful influence.

We emphasize that MI is designed to measure the efficacy of working memory rather than to depict the agent’s competence in performing memory management. For instance, an agent with both $AUV_{w/memory} = 1$ and $AUV_{w/o memory} = 1$ produces a zero MI, but this does not imply a failure in memory management. It suggests that the memory information plays a negligible role between these settings. We demonstrate the MI results in the FrozenLake Environment in Figure 4(a) and reveal the following findings:

Negative Memory Influence is Widely Observed. As illustrated in Figure 4(a), contrary to the prevailing assumption that expanding context promotes agent performance, our MI data reveal a counter-intuitive phenomenon: as evidenced by negative memory influence of multiple models in FrozenLake, simply leveraging working memory without

further management often acts as a cognitive burden, especially for open-source models. This may be because excessive irrelevant details in memory hinder reasoning, suggesting that a simple scaling context length is not a universally valid strategy for performance enhancement, highlighting the need for active memory management.

Moreover, we provide the agent with different window sizes of interaction history and reveal that simply scaling the history buffer yields negligible improvements once the essential context is captured, which can be found in Appendix B.6. We additionally conduct another experiment that the agent automatically summarize the history and continue to interact based on the summary. Results in Appendix B.7 indicates that a simple summary is still a suboptimal mechanism.

Memory Mechanism Analysis. To investigate the underlying cause of the memory limitation, in AlfWorld, we define Memory Recall Lag as the temporal interval between acquiring a critical information and its utilization. For example, if the agent takes a fork at the first step and use it at the third step, then the interval is 2. We firstly select the trajectories, in which the agent acquires the critical information, then divide them by whether the agent finishes the task. In Figure 4(b), we reveal a sharp divergence: successful trajectories cluster tightly around low lags, while failures exhibit a long-tail distribution of high lags. This confirms that although the agent retrieves critical information, it fails reasoning across long-range information. Implementation details can be found in Appendix C.5.

Takeaway 3: Simply leveraging working memory without deliberate management is often harmful in several scenarios, which can be caused by the inability of reasoning across long horizon interactions.

4 Application of TIDE Framework

In this section, we synthesize the three metrics to present a holistic TTI analysis of the SOTA LLM agents in the five environments mentioned above. We further demonstrate that the AUV and LR components of *TIDE* can be applied post-hoc to external trajectories.

4.1 Comprehensive Diagnostic

TIDE serves as a lightweight, agent-agnostic and environment-agnostic tool to deconstruct the underlying mechanisms of TTI into three distinct dimensions. Figure 5 depicts a comparative analysis. More results across various models can be found in Appendix B.8.

Extremely Large-scale Models Stand Out as a Robust Baseline. *Gemini* and *DeepSeek* series consistently achieve superior AUV and LR across all environments. Notably, in tasks like BlocksWorld, these models exhibit relatively low MI. The lower MI does not imply a deficiency in the model’s memory utilization capability. This suggests that their robust reasoning capability reduces reliance on interaction memory.

Adaptive Models Derive TTI Capacity from Working Memory With Minimum Loop. We identify agents like *GLM-32B-0414* with high memory sensitivity, characterized by a small gap between AUV and MI. This suggests that their TTI benefits substantially from access to interaction history under our ablation protocol. Crucially, this success corresponds to a low LR, indicating that memory can be more effectively translated into behavior adaptation when agents avoid recurrent loops. We highlight that agents with limited prior task competence can attain strong performance with TTI through environmental feedback.

Merely Test-Time Scaling is Insufficient for TTI. We challenge the view that test-time scaling inherently guarantees better behavior adaptation. In our experiments results, reasoning-enhanced models, such as *gpt-oss-120b*, often fail to translate internal chain-of-thought into effective external actions in tasks like AlfWorld. This reveals a critical decoupling between internal cognitive capacity and external interactive efficacy. Consequently, agent design should shift from solely maximizing static reasoning depth to explicitly optimizing dynamic, interaction-driven evolution.

4.2 Generalize to Extended Environments

Our proposed lightweight framework is deliberately designed to be both agent-agnostic and environment-agnostic, ensuring broad compatibility across diverse evaluation settings. Leveraging this property, we extend *TIDE* to GUI environments through a secondary analysis of existing interaction logs (Xie et al., 2024; Bonatti

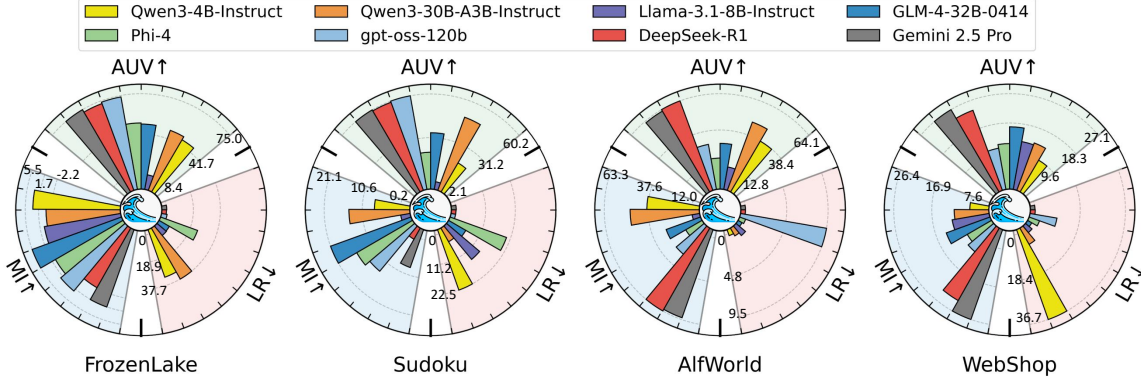


Figure 5: Comprehensive evaluation based on *TIDE*. Bar height stands for the normalized performance on each metric. Uparrow $\uparrow$ indicates that a higher AUV is better. Downarrow $\downarrow$ indicates that a lower LR is better. For MI, larger values indicate stronger positive dependence on working memory under the ablation protocol, not necessarily better overall agent quality. We present 8 models across 4 environments. More results can be found in Appendix B.8.

Model	AW		OSW		WAA	
	AUV	LR	AUV	LR	AUV	LR
InternVL3.5-30B-A3B	12.4	6.8	—	—	6.7	4.4
UI-TARS-1.5-7B	25.0	44.8	15.4	21.3	—	—
UI-TARS-72B-DPO	34.8	16.9	20.3	16.1	—	—
Qwen2.5-VL-72B-Instruct	—	—	5.9	4.6	6.0	8.1
Qwen3-VL-235B-A22B-Instruct	43.2	32.6	—	—	9.5	19.1
Claude3.7-Sonnet-20250219	—	—	8.4	1.4	0.0	21.6
GPT-4o + ScaleCUA-7B	—	—	15.8	36.8	—	—

Table 3: Evaluation results in AndroidWorld (AW), OS-World (OSW), WindowsAgentArena (WAA). For *GPT-4o + ScaleCUA-7B*, the former acts as a planner and the latter acts as a grounder for low-level execution. **Colored model name** contains proprietary model.

et al., 2025; Liu et al., 2025b; Rawles et al., 2025; Sun et al., 2025b). Notably, *TIDE* can operate solely on recorded interaction trajectories without re-executing the experiments. Implementation details of the code are provided in Appendix C.7, which can generalize to various environments.

We use the screenshot to define the current state, which is processed into an embedding through CLIP model. Two state are considered the same if their cosine similarity is above a pre-defined threshold. The results provided in Table 3 reveal a ubiquitous looping phenomenon, where models become trapped in repetitive, unproductive actions in GUI settings. We additionally provide the LR variation along with different thresholds in Appendix B.9, revealing that our proposed LR is a robust metric under different threshold settings. Moreover, we argue that the grounding capability is still a bottleneck of existing GUI agents in Appendix B.10.

5 Related Works

Multi-turn Interaction Agent. While LLMs exhibit strong reasoning capabilities (Xu et al., 2024;

DeepSeek-AI et al., 2025; Xu et al., 2025a; Yang et al., 2025; Wu et al., 2026a,c; Yan et al., 2026), relying purely on internal estimates lacks environmental grounding, which is prone to hallucination (Du et al., 2024; Yan et al., 2025; Huang et al., 2025a) and suboptimal performance in partially observable or non-static scenarios (Parisi et al., 2024; Wei et al., 2025). Consequently, effective problem-solving requires self-refinement (Madaan et al., 2023; Cheng et al., 2025; Wu et al., 2026b) to adapt action from environmental feedback. Recent works (Gao et al., 2025; Dou et al., 2025; Shen et al., 2025; Huang et al., 2025b; Acikgoz et al., 2025; Yang et al., 2026) are shifting their focus to multi-turn interaction agents, which adopt inner reasoning to dynamic interaction contexts. We formalize these evolving mechanisms as TTI.

LLM Agent Evaluation. Outcome-oriented frameworks (Liu et al., 2025a; Zhou et al., 2024; Sun et al., 2025a) primarily benchmark agents based on the final Success Rate, ignoring the interaction quality. Beyond the LLM setting, studies in reinforcement learning via sample-efficiency curves (Taylor and Stone, 2009; Brys et al., 2015) and cycle-detection analyses (Bellemare et al., 2016) assume access to the value function that black-box LLM agents do not expose. Several works have shifted towards fine-grained analysis. For example, refining SR into Progress Rate (PR) (Chang et al., 2024; Gioacchini et al., 2024) or decompose performance into specific capabilities (Jian et al., 2024; Chang et al., 2025; Yuanzhe et al., 2025; Yu et al., 2025). However, they overlook the temporal dynamics of how an agent adapts

behavior during interaction. In contrast, *TIDE* is deliberately designed to be content-aware, assessing the interaction quality beyond final success.

6 Conclusion

We formalize TTI as a dynamic improvement process. To address the limitations of existing evaluations in multi-turn agent tasks, we introduce *TIDE*, an agent-agnostic and environment-agnostic diagnostic framework that characterizes the dynamics of TTI through three complementary metrics: AUV, LR, and MI. Experiments reveals that agent performance hinges not only on final success but also on how efficiently, adaptively, and effectively agents improve through interaction.

Limitations

(1) Our study mainly focuses on test-time improvement in single-agent systems built upon foundation models. Extending the proposed analysis to multi-agent settings, where improvement may emerge from inter-agent interaction, is left for future work. (2) Moreover, while *TIDE* applies to a wide range of agentic tasks, certain domains with irreversible dynamics (e.g., board games) challenge the interpretation of loop-based diagnostics. In such cases, Loop Ratio may not strictly indicate suboptimal behavior, motivating domain-aware extensions.

Ethics Statement

In this submission, we use AI to polish our writing and code. Moreover, we agree to release our code and data under the MIT License to facilitate future research. For all the environments and models used in this work, we strictly adhere to their original usage terms and licenses. We confirm that all existing artifacts are used strictly for their intended purpose of academic research and agent evaluation. Several icons in our figures are from FLATICON website.

Acknowledgments

This work was supported by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM116), National Natural Science Foundation of China (No.62137002, 62293553, 62450005, 62477036, 62192781).

References

- Emre Can Acikgoz, Cheng Qian, Heng Ji, Dilek Hakkani-Tür, and Gokhan Tur. 2025. [Self-improving llm agents at test-time](#). *Preprint*, arXiv:2510.07841.
- Huan ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, and 8 others. 2025. [A survey of self-evolving agents: On path to artificial super intelligence](#). *Preprint*, arXiv:2507.21046.
- Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29.
- Rogério Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, Lawrence Keunho Jang, and Zheng Hui. 2025. [Windows agent arena: Evaluating multi-modal OS agents at scale](#). In *Forty-second International Conference on Machine Learning*.
- Tim Brys, Anna Harutyunyan, Matthew E Taylor, and Ann Nowé. 2015. Policy transfer using reward shaping. In *AAMAS*, pages 181–188.
- Edward Chang, Longling Geng, and Edward Chang. 2025. [Realm-bench: A real-world planning benchmark for llms and multi-agent systems](#). <https://doi.org/10.48550/arxiv.2502.18836>, abs/2502.18836.
- Ma Chang, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. 2024. [Agentboard: An analytical evaluation board of multi-turn llm agents](#). *Advances in neural information processing systems*, 37:74325–74362.
- Zhaoling Chen, Xiangru Tang, Gangda Deng, Fang Wu, Jialong Wu, Zhiwei Jiang, Viktor Prasanna, Arman Cohan, and Xingyao Wang. 2025. [Locagent: Graph-guided llm agents for code localization](#). *Preprint*, arXiv:2503.09089.
- Kanzhi Cheng, Li YanTao, Fangzhi Xu, Jianbing Zhang, Hao Zhou, and Yang Liu. 2025. [Vision-language models can self-improve reasoning via reflection](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8876–8892.
- Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. [Mem0: Building production-ready ai agents with scalable long-term memory](#). *arXiv preprint arXiv:2504.19413*.

- DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *Preprint*, arXiv:2501.12948.
- Shihan Dou, Ming Zhang, Chenhao Huang, Jiayi Chen, Feng Chen, Shichun Liu, Yan Liu, Chenxiao Liu, Cheng Zhong, Zongzhang Zhang, Tao Gui, Chao Xin, Chengzhi Wei, Lin Yan, Yonghui Wu, Qi Zhang, and Xuanjing Huang. 2025. [Evalearn: Quantifying the learning capability and efficiency of llms via sequential problem solving](#). *Preprint*, arXiv:2506.02672.
- Xuefeng Du, Chaowei Xiao, and Sharon Li. 2024. [Halo-scope: Harnessing unlabeled llm generations for hallucination detection](#). *Advances in Neural Information Processing Systems*, 37:102948–102972.
- Luca Gioacchini, Giuseppe Siracusano, Davide Sanvito, Kiril Gashteovski, David Friede, Roberto Bifulco, Carolin Lawrence, et al. 2024. [Agentquest: A modular benchmark framework to measure progress and improve llm agents](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: System Demonstrations)*, volume 3, pages 185–193. Association for Computational Linguistics.
- Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2025a. [A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions](#). *ACM Transactions on Information Systems*, 43(2):1–55.
- Yuchen Huang, Sijia Li, Minghao Liu, Wei Liu, Shijue Huang, Zhiyuan Fan, Hou Pong Chan, and Yi R. Fung. 2025b. [Environment scaling for interactive agentic experience collection: A survey](#). *Preprint*, arXiv:2511.09586.
- Xie Jian, Jian Xie, Zhang Kai, Kai Zhang, Jian Xie, Jiangjie Chen, Jiangjie Chen, Kai Zhang, Zhu Tinghui, Tinghui Zhu, Jiangjie Chen, Renze Lou, Renze Lou, Tinghui Zhu, Yuandong Tian, Yuandong Tian, Renze Lou, Xiao Yanghua, Yanghua Xiao, and 5 others. 2024. [Travelplanner: A benchmark for real-world planning with language agents](#). <http://arxiv.org/abs/2402.01622>.
- Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. [Swe-bench: Can language models resolve real-world github issues?](#) *Preprint*, arXiv:2310.06770.
- Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, and 3 others. 2025a. [Agentbench: Evaluating llms as agents](#). *Preprint*, arXiv:2308.03688.
- Zhaoyang Liu, Jingjing Xie, Zichen Ding, Zehao Li, Bowen Yang, Zhenyu Wu, Xuehui Wang, Qiushi Sun, Shi Liu, Weiyun Wang, Shenglong Ye, Qingyun Li, Xuan Dong, Yue Yu, Chenyu Lu, YunXiang Mo, Yao Yan, Zeyue Tian, Xiao Zhang, and 11 others. 2025b. [Scalecua: Scaling open-source computer use agents with cross-platform data](#). *Preprint*, arXiv:2509.15221.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. [Self-refine: Iterative refinement with self-feedback](#). *Advances in Neural Information Processing Systems*, 36:46534–46594.
- Simone Parisi, Alireza Kazemipour, and Michael Bowling. 2024. [Beyond optimism: Exploration with partially observable rewards](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Priya Pitre, Naren Ramakrishnan, and Xuan Wang. 2025. [CONSENSAGENT: Towards efficient and effective consensus in multi-agent LLM interactions through sycophancy mitigation](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 22112–22133, Vienna, Austria. Association for Computational Linguistics.
- Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjuan Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, and 16 others. 2025. [Ui-tars: Pioneering automated gui interaction with native agents](#). *Preprint*, arXiv:2501.12326.
- Jielin Qiu, Zuxin Liu, Zhiwei Liu, Rithesh Murthy, Jiangguo Zhang, Haolin Chen, Shiyu Wang, Ming Zhu, Liangwei Yang, Juntao Tan, Roshan Ram, Akshara Prabhakar, Tulika Awalgaonkar, Zixiang Chen, Zhepeng Cen, Cheng Qian, Shelby Heinecke, Weiran Yao, Silvio Savarese, and 2 others. 2025. [Locobench-agent: An interactive benchmark for llm agents in long-context software engineering](#). *Preprint*, arXiv:2511.13998.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. [Learning transferable visual models from natural language supervision](#). In *International conference on machine learning*, pages 8748–8763. PmLR.
- Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William E Bishop, Wei Li, Folawiyi Campbell-Ajala, Daniel Kenji Toyama, Robert James Berry, Divya Tyamagundlu, Timothy P Lillicrap, and Oriana Riva. 2025. [Androidworld: A dynamic benchmarking environment for autonomous agents](#). In

- The Thirteenth International Conference on Learning Representations.*
- Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amrith Setlur, Shengbang Tong, Diego Caples, Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. 2025. [Thinking vs. doing: Agents that reason by scaling test-time interaction](#). *Preprint*, arXiv:2506.07976.
- Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. [{ALFW}orld: Aligning text and embodied environments for interactive learning](#). In *International Conference on Learning Representations*.
- Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, et al. 2025a. [Os-genesis: Automating gui agent trajectory construction via reverse task synthesis](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5555–5579.
- Qiushi Sun, Zhoumianze Liu, Chang Ma, Zichen Ding, Fangzhi Xu, Zhangyue Yin, Haiteng Zhao, Zhenyu Wu, Kanzhi Cheng, Zhaoyang Liu, Jianing Wang, Qintong Li, Robert Tang, Tianbao Xie, Xiachong Feng, Xiang Li, Ben Kao, Wenhai Wang, Biqing Qi, and 2 others. 2025b. [Scienceboard: Evaluating multimodal autonomous agents in realistic scientific workflows](#). In *ICML 2025 Workshop on Computer Use Agents*.
- Matthew E Taylor and Peter Stone. 2009. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7).
- Mauro Vallati, Lukas Chrpá, Marek Grzes, Thomas Leo McCluskey, Mark Roberts, Scott Sanner, et al. 2015. [The 2014 international planning competition: Progress and trends](#). *Ai Magazine*, 36(3):90–98.
- Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024. [Executable code actions elicit better LLM agents](#). In *Forty-first International Conference on Machine Learning*.
- Xuehui Wang, Zhenyu Wu, JingJing Xie, Zichen Ding, Bowen Yang, Zehao Li, Zhaoyang Liu, Qingyun Li, Xuan Dong, Zhe Chen, Weiyun Wang, Xiangyu Zhao, Jixuan Chen, Haodong Duan, Tianbao Xie, Chenyu Yang, Shiqian Su, Yue Yu, Yuan Huang, and 9 others. 2025a. [Mmbench-gui: Hierarchical multi-platform evaluation framework for gui agents](#). *Preprint*, arXiv:2507.19478.
- Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. 2025b. [Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning](#). *Preprint*, arXiv:2504.20073.
- Chenxing Wei, Hong Wang, Ying He, Fei Yu, and Yao Shu. 2025. [Test-time policy adaptation for enhanced multi-turn interactions with llms](#). *Preprint*, arXiv:2509.23166.
- Jinyang Wu, Mingkuan Feng, Shuai Zhang, Feihu Che, Zhengqi Wen, Chonghua Liao, Ling Yang, Haoran Luo, Zheng Lian, and Jianhua Tao. 2026a. Beyond examples: Towards automated thought-level in-context reasoning for large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2955–2995, San Diego, California, United States. Association for Computational Linguistics.
- Jinyang Wu, Shuo Yang, Zhengxi Lu, Fan Zhang, Yuhao Shen, Lang Feng, Haoran Luo, Zheng Lian, Shuai Zhang, Zhengqi Wen, et al. 2026b. [Seed: Self-evolving on-policy distillation for agentic reinforcement learning](#). *arXiv preprint arXiv:2607.14777*.
- Jinyang Wu, Shuo Yang, Yuhao Shen, Shuai Zhang, Zhengqi Wen, and Jianhua Tao. 2026c. [SPARK: Strategic policy-aware exploration via dynamic branching for long-horizon agentic learning](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23981–24004, San Diego, California, United States. Association for Computational Linguistics.
- Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. 2024. [Os-atlas: A foundation action model for generalist gui agents](#). *Preprint*, arXiv:2410.23218.
- Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2025. [The rise and potential of large language model based agents: A survey](#). *Science China Information Sciences*, 68(2):121101.
- Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. 2024. [Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments](#). *Advances in Neural Information Processing Systems*, 37:52040–52094.
- Fangzhi Xu, Jun Liu, Qika Lin, Tianzhe Zhao, Jian Zhang, and Lingling Zhang. 2024. [Mind reasoning manners: Enhancing type perception for generalized zero-shot logical reasoning over text](#). *IEEE Transactions on Neural Networks and Learning Systems*, 35(12):18499–18511.
- Fangzhi Xu, Hang Yan, Chang Ma, Haiteng Zhao, Jun Liu, Qika Lin, and Zhiyong Wu. 2025a. [phi-decoding: Adaptive foresight sampling for balanced inference-time exploration and exploitation](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13214–13227.

Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Z. Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Maben, Raj Mehta, Wayne Chi, Lawrence Jang, Yiqing Xie, and 2 others. 2025b. [Theagentcompany: Benchmarking llm agents on consequential real world tasks](#). *Preprint*, arXiv:2412.14161.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. 2025. [Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning](#). *arXiv preprint arXiv:2509.02479*.

Hang Yan, Fangzhi Xu, Rongman Xu, Yifei Li, Jian Zhang, Haoran Luo, Xiaobao Wu, Anh Tuan Luu, Haiteng Zhao, Qika Lin, and Jun Liu. 2026. [MUR: Momentum uncertainty guided reasoning for large language models](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23078–23103, San Diego, California, United States. Association for Computational Linguistics.

Hang Yan, Fangzhi Xu, Rongman Xu, Yifei Li, Jian Zhang, Haoran Luo, Xiaobao Wu, Luu Anh Tuan, Haiteng Zhao, Qika Lin, and Jun Liu. 2025. [Mur: Momentum uncertainty guided reasoning for large language models](#). *Preprint*, arXiv:2507.14958.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. 2025. [Dynamic early exit in reasoning models](#). *Preprint*, arXiv:2504.15895.

Shuo Yang, Jinyang Wu, Zhengxi Lu, Yuhao Shen, Fan Zhang, Lang Feng, Shuai Zhang, Haoran Luo, Zheng Lian, Zhengqi Wen, et al. 2026. [Opid: On-policy skill distillation for agentic reinforcement learning](#). *arXiv preprint arXiv:2606.26790*.

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 Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). *Preprint*, arXiv:2210.03629.

Wang Yu, Chen Xi, Yu Wang, and Xi Chen. 2025. [Mirix: Multi-agent memory system for llm-based agents](#). <https://doi.org/10.48550/arxiv.2507.07957>, abs/2507.07957.

Hu Yuanzhe, Wang Yu, Yuanzhe Hu, Julian McAuley, Yu Wang, and Julian McAuley. 2025. [Evaluating memory in llm agents via incremental multi-turn interactions](#). <http://arxiv.org/abs/2507.05257>, abs/2507.05257.

Yang Zhang, Shixin Yang, Chenjia Bai, Fei Wu, Xiu Li, Zhen Wang, and Xuelong Li. 2025. [Towards efficient](#)

[llm grounding for embodied multi-agent collaboration](#). *Preprint*, arXiv:2405.14314.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. [Webarena: A realistic web environment for building autonomous agents](#). *Preprint*, arXiv:2307.13854.

A Theoretical Analysis of AUV Properties

In this section, we provide a rigorous theoretical framework to analyze the mathematical properties of the area under variation (AUV) metric. We argue that **AUV is not merely a redundant correlate of final success rate, but a distinct metric that captures convergence rate and marginal interaction gain**. Moreover, we provide proof of statistical consistency and convergence for AUV.

A.1 Preliminaries and The Weighted-Increment Lemma

Let the interaction horizon be defined as $\mathcal{T} = \{t \in \mathbb{Z} \mid 0 \leq t < t_{max}\}$. Let P_t denote the cumulative success rate at turn t , which is monotonically non-decreasing. Notably, $P_0 = 0$. We define the marginal performance gain at step k as δ_k , representing the incremental probability of discovering a solution at the k -th interaction turn:

$$\delta_k = P_{k+1} - P_k, \quad \text{where } \delta_k \geq 0. \quad (6)$$

Consequently, the performance at any step t can be reconstructed as:

$$P_t = \sum_{k=t_{min}}^{t-1} \delta_k \quad (7)$$

Recall the definition of AUV, calculated via the discrete trapezoidal integration rule over the normalized horizon $H = t_{max}$:

$$\text{AUV} = \frac{1}{H} \sum_{t=0}^{t_{max}-1} \frac{P_t + P_{t+1}}{2} \quad (8)$$

To facilitate our proofs, we first derive **Lemma 1** that reformulates AUV as a weighted sum of marginal gains.

Lemma 1: Weighted-Increment Representation
The AUV is mathematically equivalent to a linear combination of all marginal gains δ_k , weighted by a time-decaying coefficient w_k .

Proof. First, substitute the recursive expansion $P_t = \sum_{k=t_{min}}^{t-1} \delta_k$ and $P_{t+1} = P_t + \delta_t$ into Eq. (8).

Let $\mathcal{A} = H \cdot \text{AUV}$ denote the area without normalization:

$$\begin{aligned}\mathcal{A} &= \sum_{t=0}^{t_{\max}-1} \left(P_t + \frac{1}{2} \delta_t \right) \\ &= \sum_{t=0}^{t_{\max}-1} \left(\sum_{k=0}^{t-1} \delta_k + \frac{1}{2} \delta_t \right)\end{aligned}\quad (9)$$

We analyze how many times a specific δ_k generated at step k contributes to the summation over t :

- The term $\frac{1}{2} \delta_t$ appears exactly once when $t = k$.
- The accumulated term $\sum_{k=0}^{t-1} \delta_k$ includes δ_k for all subsequent time steps $t > k$. Specifically, δ_k contributes a weight of 1.0 for each step from $t = k + 1$ to $t_{\max} - 1$.

Summing these contributions, the total weight coefficient w_k for δ_k is:

$$\begin{aligned}w(k) &= \underbrace{0.5}_{\text{contribution at } t=k} + \underbrace{\sum_{t=k+1}^{t_{\max}-1} 1}_{\text{contribution at } t>k} \\ &= 0.5 + ((t_{\max} - 1) - (k + 1) + 1) \\ &= t_{\max} - k - 0.5\end{aligned}\quad (10)$$

Thus, AUV can be rewritten as follows:

$$\text{AUV} = \frac{1}{H} \sum_{k=0}^{t_{\max}-1} w(k) \delta_k \quad (11)$$

□

A.2 Differs from Final Success Rate

Proposition 1: *AUV resolves the ambiguity within the equivalence class of trajectories yielding the same final metric SR_{final} , strictly distinguishing efficient paths from inefficient ones.*

Proof. The final success rate is a path-agnostic metric that performs a compression of the interaction trajectory. In contrast, AUV is a path-dependent metric.

Let the trajectory of marginal gains be represented by a vector $\delta = [\delta_0, \dots, \delta_{t_{\max}-1}]^T$. The standard metric SR_{final} is mathematically equivalent to the unweighted L_1 norm shifted by the baseline:

$$SR_{\text{final}}(\delta) = \|\delta\|_1 = \sum_k \delta_k \quad (12)$$

This functional is permutation-invariant that any temporal rearrangement of the gain sequence $\{\delta_k\}$ yields the exact same SR_{final} . Thus, SR_{final} defines an set $\mathcal{C}_\Omega = \{\delta \mid \sum \delta_k = \Omega\}$, where all trajectories achieving total gain Ω are indistinguishable.

In contrast, AUV is formulated as the inner product:

$$\text{AUV}(\delta) = \frac{1}{H} \langle \mathbf{w}, \delta \rangle = \frac{1}{H} \sum_k w(k) \delta_k \quad (13)$$

Since $\mathbf{w}$ is not vector $\mathbf{1}$ that elements are all ones, the value defined by AUV is totally different from the value defined by SR.

To demonstrate the resolving power of AUV, consider a temporal perturbation within the set $\mathcal{C}_\Omega$. Assume an agent originally achieves a marginal gain ϵ at a late step t_{late} . We construct a variation where this specific gain is shifted to an earlier step t_{early} , forming a new marginal gain trajectory δ' . So that $\delta'_{t_{\text{early}}} = \delta_{t_{\text{early}}} + \epsilon$ and $\delta'_{t_{\text{late}}} = \delta_{t_{\text{late}}} - \epsilon$.

This perturbation shows keeps that $\Delta SR = 0$. And $\Delta \text{AUV} \propto w(t_{\text{early}}) \cdot \epsilon - w(t_{\text{late}}) \cdot \epsilon > 0$.

This proves that while SR_{final} treats performance as a static state variable, AUV treats it as a dynamic process variable. AUV strictly rewards the temporal optimization of the solution path, retaining structural information lost by the standard metric. □

A.3 Statistical Consistency and Convergence

In real-world evaluations, the true performance of an agent is an unknown population parameter. We estimate it using a finite dataset of N tasks. Let δ_k^* denote the true expected marginal increment $\in \mathbb{R}$ that the agent achieves exactly at step k . The empirical marginal increment observed in the experiment is a random variable, denoted as $\hat{\delta}_k$.

Proposition 2: Consistency and Variance Decay *Let $\widehat{\text{AUV}}_N$ be the empirical estimator calculated from N i.i.d. tasks. As N increases, $\widehat{\text{AUV}}_N$ converges in probability to the true population value AUV^* . The estimator is unbiased, and its variance decays at a rate of $O(1/N)$.*

Proof. For a dataset of N tasks, let $x_k^{(i)}$ be the marginal metric increment observed for the i -th task at step k . Here, $x_k^{(i)} \in [0, 1]$ represents the value added at this specific step. The empirical marginal gain $\hat{\delta}_k$ is the sample mean of these individual increments:

$$\hat{\delta}_k = \frac{1}{N} \sum_{i=1}^N x_k^{(i)} \quad (14)$$

Consequently, the cumulative performance P_t is the summation of these empirical marginal gains:

$$P_t = \sum_{j=0}^{t-1} \hat{\delta}_j \quad (15)$$

Based on **Lemma 1**, the empirical AUV is the weighted sum of these marginal gains:

$$\widehat{\text{AUV}}_N = \frac{1}{H} \sum_{k=0}^{t_{\max}-1} w_k \hat{\delta}_k \quad (16)$$

Substituting the sample definition of $\hat{\delta}_k$:

$$\widehat{\text{AUV}}_N = \frac{1}{H} \sum_{k=0}^{t_{\max}-1} w_k \left(\frac{1}{N} \sum_{i=1}^N x_k^{(i)} \right) \quad (17)$$

We rearrange the summation order to isolate individual samples:

$$\widehat{\text{AUV}}_N = \frac{1}{N} \sum_{i=1}^N \underbrace{\left(\frac{1}{H} \sum_{k=0}^{t_{\max}-1} w_k x_k^{(i)} \right)}_{Z^{(i)}} \quad (18)$$

Here, $Z^{(i)}$ represents a single-task AUV score for the i -th task. Since the tasks are i.i.d., the variables $Z^{(1)}, \dots, Z^{(N)}$ are i.i.d. random variables with finite variance σ_Z^2 .

Assuming the true expected increment is $\mathbb{E}[x_k^{(i)}] = \delta_k^*$:

$$\begin{aligned} \mathbb{E}[\widehat{\text{AUV}}_N] &= \frac{1}{H} \sum_k w_k \mathbb{E}[\hat{\delta}_k] \\ &= \frac{1}{H} \sum_k w_k \delta_k^* \\ &= \text{AUV}^* \end{aligned}$$

The variance of the sample mean becomes:

$$\begin{aligned} \text{Var}(\widehat{\text{AUV}}_N) &= \text{Var} \left(\frac{1}{N} \sum_{i=1}^N Z^{(i)} \right) \\ &= \frac{1}{N^2} \sum_{i=1}^N \text{Var}(Z^{(i)}) \\ &= \frac{\sigma_Z^2}{N} \end{aligned}$$

Since the variance tends to 0 as N increases, and the estimator is unbiased, $\widehat{\text{AUV}}_N$ converges in probability to AUV^* . This confirms that AUV is a statistically consistent estimator. $\square$

B Supplementary Analysis

B.1 Flexible&Extensible Framework

Our proposed metric is highly flexible and extensible. Operationally, metrics such as AUV and LR are derived directly from standard interaction logs, including step-wise success rates and raw trajectory sequences without architectural modifications. Notably, our proposed metric AUV is not only designed for binary success rate. It can also be applied on progress rate or other metrics. Furthermore, the computation of MRI necessitates only a straightforward control experiment, restricting the agent's input to the immediate state. Detailed coding help for adopting other logs into our framework can be found in Appendix C.7.

B.2 More Results of SR and AUV

To strongly demonstrate that AUV is a new metrics that captures more information than SR, we illustrate some scenarios that a higher SR does not result in a high AUV in Table 4.

B.3 Time to 50% success

To provide more information on success rate against interaction turns, we further provide time to 50% success of each model across all environments as a time-weighted success integral to enrich the evaluative dimension of the AUV metric: it enables fine-grained differentiation between early and late task success, capturing nuanced differences in success trajectories (e.g., early plateau versus late surge) that are not fully characterized by a single AUV value alone, which can be found in Table 5.

B.4 Qualitative Analysis of Loop Phenomenon

We manually inspect 100 randomly sampled failure trajectories on AlfWorld and WebShop, focusing on trajectories containing loop actions.

On AlfWorld, we find that 42% of loops stem from planning failures, in which the agent repeatedly executes a syntactically valid but strategically wrong sub-plan. For example, re-opening the same drawer after already inspecting it, instead of moving on to the next receptacle. 40% are memory-recall failures, where the agent revisits a location whose contents were observed earlier in the trajectory but absent from its working reasoning context. 18% are grounding failures, where the agent issues actions referring to objects or receptacles that do not exist in the current observation. For instance,

Benchmark	ModelA	ModelB	SR _A	SR _B	AUV _A	AUV _B
BlocksWorld	Qwen3-4B-Thinking	Phi-4-reasoning	77.0	68.0	47.9	51.3
	Qwen3-30B-A3B-Thinking	DeepSeek-V3.2	98.0	98.0	69.8	71.1
	gpt-oss-120b	DeepSeek-R1	100.0	100.0	73.7	74.9
	gpt-oss-120b	Gemini 2.5 Pro	100.0	100.0	73.7	75.2
	DeepSeek-R1	Gemini 2.5 Pro	100.0	100.0	74.9	75.2
FrozenLake	Gemini 2.5 Flash	DeepSeek-V3.2	97.0	97.0	70.7	73.1
	Gemini 2.5 Flash	DeepSeek-R1	97.0	96.0	70.7	72.6
	Phi-4-reasoning	Qwen3-30B-A3B-Instruct	62.0	61.0	48.8	49.0
Sudoku	GLM-4-9B-Chat	Llama-3.1-8B-Instruct	4.0	4.0	2.0	2.1
	GLM-4-32B-0414	Phi-4-reasoning	55.0	51.0	34.4	37.4
	Qwen3-4B-Thinking	GLM-4-32B-0414	55.0	55.0	30.7	34.4
	Qwen3-4B-Thinking	Phi-4-reasoning	55.0	51.0	30.7	37.4
	Qwen3-30B-A3B-Thinking	DeepSeek-V3.2	95.0	93.0	55.6	56.8
AlfWorld	DeepSeek-V3.2	Gemini 2.5 Pro	80.7	80.7	59.0	62.9
	gpt-oss-120b	GLM-4-32B-0414	50.7	37.9	29.0	29.0
WebShop	Minstral-3-14B-Instruct	Qwen3-4B-Instruct	15.2	15.0	5.4	9.6
	Minstral-3-14B-Instruct	Qwen3-4B-Thinking	15.2	12.2	5.4	8.2
	Llama-3.1-8B-Instruct	Qwen3-30B-A3B-Instruct	20.8	20.8	13.1	13.7
	Phi-4-reasoning	Qwen3-30B-A3B-Thinking	22.6	21.0	14.3	14.6
	gpt-oss-120b	Qwen3-30B-A3B-Instruct	26.2	20.8	11.1	13.7
	Llama-3.3-70B-Instruct	gpt-oss-120b	26.8	26.2	17.6	11.1
	gpt-oss-120b	GLM-4-9B-Chat	26.2	16.4	11.1	11.5
	gpt-oss-120b	GLM-4-32B-0414	26.2	25.4	11.1	17.2
	gpt-oss-120b	Phi-4	26.2	18.8	11.1	12.2
	gpt-oss-120b	Phi-4-reasoning	26.2	22.6	11.1	14.3

Table 4: All model pairs (A, B) satisfying $SR(A) \geq SR(B)$ and $AUV(A) \leq AUV(B)$. Three key scenarios of the complementary nature of SR and AUV metrics in model evaluation are illustrated: equal SR with divergent AUV, equal AUV with divergent SR, lower AUV but higher SR, and instances of lower SR but higher AUV between two models.

taking apple 1 from countertop 2 when no apple was ever observed.

On WebShop, the distribution shifts noticeably: 51% of loops are planning failures (e.g., oscillating between two product-listing pages without committing to a purchase), 26% are grounding failures (e.g., clicking buttons or filters whose textual labels were hallucinated), 23% are memory-recall failures (e.g., re-issuing a search query that had already been issued earlier with the same keywords).

Across both environments, planning failures dominate, suggesting that loop stagnation in current LLM agents is primarily a deliberation deficit rather than a perception or memory deficit. Memory-recall failures are notably more frequent in AlfWorld, consistent with its stronger partial observability and longer dependency horizons. Grounding failures are correspondingly more frequent in WebShop, whose action space is tightly coupled to dynamically rendered page elements. This qualitative breakdown complements our quantitative LR analysis and confirms that LR captures heterogeneous failure modes that can be further

decomposed by domain-specific annotation.

B.5 More Results of MI

We report detail MI data in Table 7, from which we can observe that a negative MI is widely observed in reasoning-bound tasks.

B.6 Expanding Memory Demonstrates Saturation.

Given the limited or even negative influence of working memory, we further evaluate the impact of fine-grained working memory length. As illustrated in Figure 6, we restrict the agent’s access to a limited prefix of recent interaction history. Only several of the most recent interaction turns are visible for the agent at each step, which is denoted as the window size. We can observe that performance gains exhibit diminishing marginal returns. Benefits are confined primarily to the first 5 window sizes, after which the curve rapidly plateaus. This indicates that simply scaling the history buffer yields negligible improvements once the essential context is captured.

Model	Blocksworld		FrozenLake		Sudoku		AlfWorld		WebShop	
	T50	AUV	T50	AUV	T50	AUV	T50	AUV	T50	AUV
Qwen3-4B-Instruct	3.0	30.8	6.0	46.1	7.0	19.3	11.0	38.3	5.0	9.6
Qwen3-30B-A3B-Instruct	5.0	45.4	6.0	49.0	7.0	48.8	11.0	48.1	5.0	13.7
Llama-3.1-8B	4.0	21.9	8.0	8.4	8.0	2.1	11.0	12.8	5.0	13.1
Llama-3.3-70B	6.0	68.2	6.0	34.7	7.0	24.9	11.0	46.7	5.0	17.6
GLM-9B-Chat	5.0	18.2	7.0	7.2	10.0	2.0	8.0	21.8	5.0	11.5
GLM-4-32B	6.0	57.5	7.0	49.9	7.0	34.4	12.0	29.0	5.0	17.2
Mistral-7B	3.0	5.0	4.0	4.6	13.0	0.4	5.0	7.6	5.0	4.4
Ministral-3-14B-Instruct	10.0	39.0	10.0	35.8	4.0	13.3	15.0	15.8	10.0	5.4
Phi-4	5.0	41.2	6.0	50.8	6.0	21.4	13.0	18.6	5.0	12.2
DeepSeek-V3.2	6.0	71.1	6.0	73.1	8.0	56.8	12.0	59.0	5.0	24.8
Gemini 2.5 Flash	6.0	72.0	7.0	70.7	7.0	60.4	12.0	58.1	5.0	21.6
Qwen3-4B-Thinking	7.0	47.9	6.0	66.7	7.0	30.7	13.0	40.9	5.0	8.2
Qwen3-30B-A3B-Thinking	6.0	69.8	7.0	68.0	8.0	55.6	12.0	50.7	5.0	14.6
Phi-4-Reasoning	5.0	51.3	6.0	48.8	5.0	37.4	19.0	9.8	5.0	14.3
gpt-oss-120b	6.0	73.7	6.0	73.8	7.0	60.2	24.0	29.0	8.0	11.1
Deepseek-R1	5.0	74.9	6.0	72.6	7.0	59.4	11.0	64.1	5.0	24.1
Gemini 2.5 Pro	5.0	75.2	6.0	75.0	8.0	60.2	10.0	62.9	5.0	27.1

Table 5: BW represents BlocksWorld, FL represents FronzenLake, Su represents Sudoku, AW represents AlfWorld, WS represents WebShop. T50 represents the turn usage for the agent to reach its half relative success rate (For example, if the final success rate is 20%, then half relative success rate is 10%).

Model	BlocksWorld		FrozenLake		Sudoku		AlfWorld		WebShop	
	Naive	Summary	Naive	Summary	Naive	Summary	Naive	Summary	Naive	Summary
Qwen3-4B-Instruct	30.8	23.5	46.1	45.7	19.3	11.9	38.3	25.5	9.6	4.8
Llama3.1-8B-Instruct	21.9	13.5	8.4	5.3	2.1	0.7	12.8	4.4	13.1	7.9
Glm4-9B-Chat	18.2	8.1	7.2	2.7	2.0	0.0	21.8	7.6	11.5	11.6
Mistral-7B-Instruct	5.0	4.8	4.6	0.0	0.4	0.0	7.6	0.0	4.4	0.0

Table 6: Memory summary performance comparison across five environments. We report AUV in two settings: **Origin** (LLM agents interact with the environment without summary) and **Summary** (LLM agents interact with the environment with summary).

B.7 Memory Summary Influence

To investigate memory utility under alternative management strategies, we instruct the LLM agent to summarize its interaction trajectory and reasoning process. The results in Table 6 demonstrate that even with summarization, the accumulated working memory continues to impose a cognitive burden on LLM agents.

B.8 More Results of *TIDE* Comprehensive Analysis

We demonstrate all comparative data in Figure 7 to provide a comprehensive diagnose of LLM agents in our paper.

B.9 LR with Different CLIP Threshold

We report the sensitivity of similarity threshold in the following table to demonstrate the stability of LR metric. Results in Table 8 indicate that providing the threshold is sufficiently high, the fluctuation

in LR is minimal and the comparative conclusions hold.

B.10 More Evaluation Results of Existing GUI Agent Trajectories

As shown in Table 9, our analysis in OSWorld reveals a stark contrast in robustness. Click Ratio quantifies the proportion of click action loops within all loops. While most agents suffer catastrophic performance degradation in looping trajectories, *Claude3.7-Sonnet-20250219* stands out as the sole exception, maintaining a loop AUV 6.9% comparable to its non-loop performance 9.0%. This resilience correlates directly with its significantly lower Click Ratio 31.1%. Since high click ratios in loops typically indicate repeated failed attempts to manipulate GUI elements, Claude’s performance pinpoints grounding precision as the critical bottleneck causing exploration failure in GUI environments.

Model	BlocksWorld		FrozenLake		Sudoku		AlfWorld		WebShop	
	AUV↑	MI	AUV↑	MI	AUV↑	MI	AUV↑	MI	AUV↑	MI
Non-thinking Model										
Qwen3-4B-Instruct	30.8	7.0	46.1	3.3	19.3	7.3	38.3	38.3	9.6	7.6
Qwen3-30B-A3B-Instruct	45.4	15.8	49.0	-0.5	48.8	14.3	48.1	48.1	13.7	11.4
Llama-3.1-8B-Instruct	21.9	5.4	8.4	0.4	2.1	0.2	12.8	12.7	13.1	12.1
Llama-3.3-70B-Instruct	68.2	35.5	34.7	-1.4	24.9	12.7	46.7	46.7	17.6	16.6
Glm-4-9B-Chat	18.2	4.7	7.2	-5.5	2.0	-2.4	21.8	20.6	11.5	8.7
Glm-4-32B-0414	57.5	30.5	49.9	5.5	34.4	21.1	29.0	29.0	17.2	14.3
Mistral-7B-Instruct	5.0	-7.7	4.6	-1.5	0.4	-0.6	7.6	7.6	4.4	3.0
Mistral-3-14B-Instruct	39.0	35.5	35.8	-1.5	13.3	8.4	15.8	15.8	5.4	5.4
Phi-4	41.2	16.8	50.8	0.6	21.4	15.3	18.6	18.6	12.2	9.8
DeepSeek-V3.2	71.1	13.8	73.1	3.8	56.8	23.4	59.0	59.0	24.8	24.0
Gemini 2.5 Flash	72.0	21.7	70.7	1.9	60.4	11.4	58.1	58.1	21.6	20.6
Thinking Model										
Qwen3-4B-Thinking	47.9	15.8	66.7	4.6	30.7	12.7	40.9	40.9	8.2	6.1
Qwen3-30B-A3B-Thinking	69.8	9.3	68.0	2.0	55.6	23.7	50.7	50.7	14.6	12.2
Phi-4-reasoning	51.3	-11.7	48.8	-21.9	37.4	8.6	9.8	9.8	14.3	13.4
gpt-oss-120b	73.7	4.0	73.8	2.4	60.2	13.7	29.0	29.0	11.1	11.1
DeepSeek-R1	74.9	7.7	72.6	-2.2	59.4	1.1	64.1	63.3	24.1	23.2
Gemini 2.5 Pro	75.2	0.3	75.0	1.7	60.2	8.6	62.9	62.2	27.1	26.1

Table 7: Detailed MI data. Colored model name represents proprietary models.

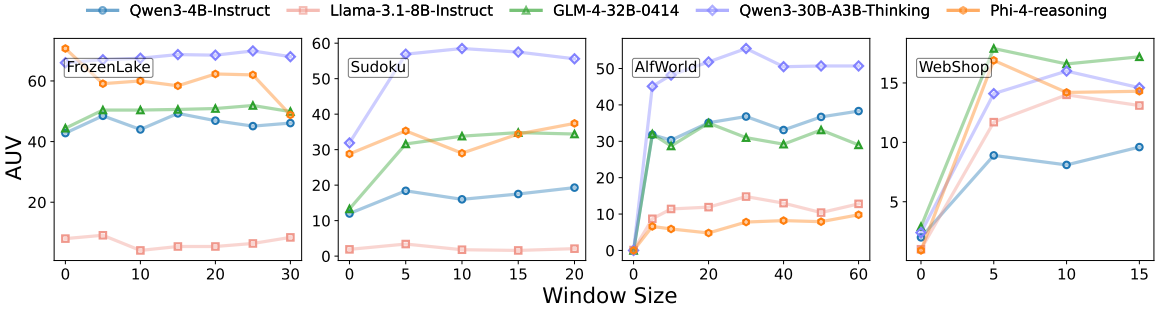


Figure 6: Window size denotes the number of most recent interaction turns retained in memory.

C Implementation Details

C.1 Experiment Environment

We select 5 widely used environments, dividing them into two categories based on the main character of environment feedback in TTI: (1) reasoning-bound (MDP), including BlocksWorld, FrozenLake, Sudoku. (2) information-bound (POMDP), including AlfWorld, WebShop.

Additionally, we apply our framework to 3 domains of existing POMDP agent interaction trajectories in AndroidWorld, OSWorld and WindowsAgentArena.

BlocksWorld is a PDDL (Vallati et al., 2015) environment in which an agent must rearrange stacks of blocks to satisfy goal configurations. In our

experiments, it contains 100 data generated by ourselves, which are included in our data file.

FrozenLake is a grid-world environment in which an agent must navigate toward a goal while avoiding holes. We follow the implementation in RAGEN (Wang et al., 2025b). In our experiments, it contains 100 data generated by ourselves, which are included in our data file.

Sudoku is a constraint-satisfaction puzzle in which agent must fill a partially specified grid so that each row, column, and designated subregion contains all required symbols exactly once. In our experiments, it contains 100 data generated by ourselves, which are included in our data file.

AlfWorld (Shridhar et al., 2021) consists of household manipulation tasks that require agents

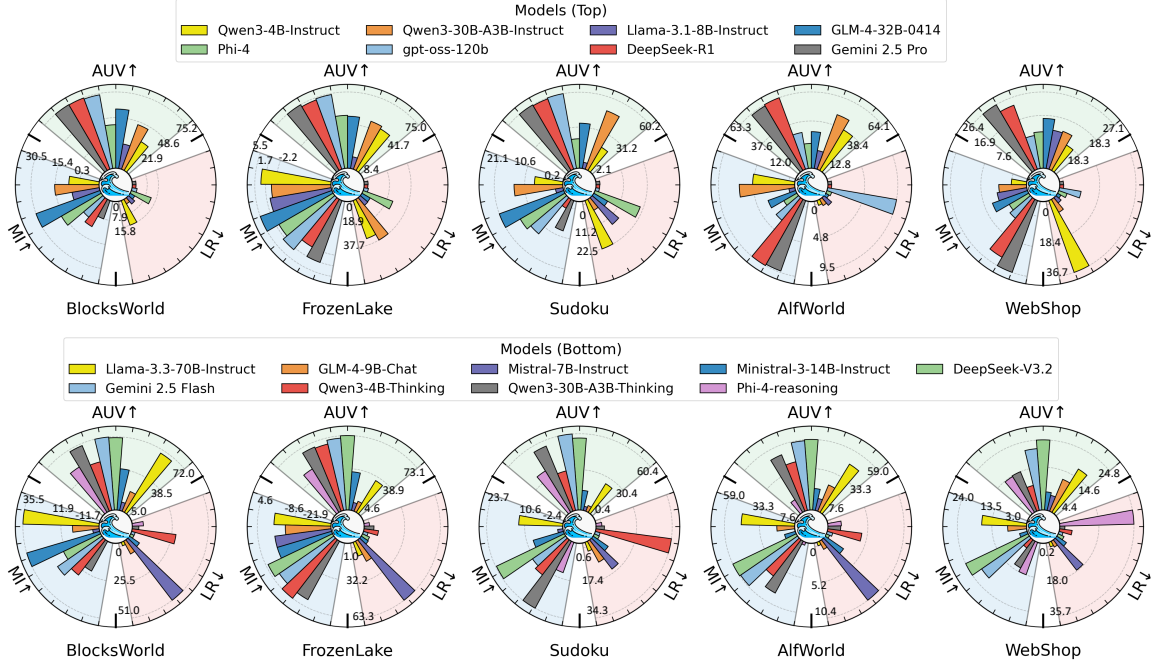


Figure 7: TIDE radar plot results.

Benchmark	Model	$\tau = 0.999$	$\tau = 0.99$	$\tau = 0.95$	$\tau = 0.90$
AndroidWorld	Qwen3-VL-235B-A22B-Instruct	32.6	33.0	33.4	33.5
	InternVL3.5-30B-A3B	6.8	7.3	7.5	7.5
	UI-TARS-72B-DPO	16.9	17.3	17.9	18.2
	UI-TARS-1.5-7B	44.8	47.1	48.3	48.7
WindowsAgentAreana	Claude3.7-Sonnet-20250219	21.6	22.8	23.6	24.2
	InternVL3.5-30B-A3B	4.4	4.5	4.9	5.0
	Qwen2.5-VL-72B-Instruct	8.1	9.2	9.5	9.6

Table 8: LR under different CLIP thresholds, which is denoted as τ .

to navigate their environment and execute actions grounded in commonsense reasoning. We use evaluation in distribution, which contains 140 data entries.

WebShop (Yao et al., 2022) serves as a network-based simulation environment that enables controlled experiments on e-commerce tasks. We use the full 500 data entries.

AndroidWorld (Rawles et al., 2025) is a real-world Android environment and benchmark designed to evaluate autonomous AI agents on their ability to execute complex, cross-application daily tasks on a mobile operating system.

OSWorld (Xie et al., 2024) is a benchmark and realistic computer environment designed to evaluate multimodal agents on their ability to execute open-ended, cross-application tasks within real desktop operating systems.

WindowsAgentArena (Bonatti et al., 2025) is a scalable benchmark and realistic environment

developed by Microsoft to evaluate multimodal AI agents on their ability to execute diverse tasks within a native Windows operating system.

C.2 Model Selection

Following recent work (Dou et al., 2025), we divide models into thinking and non-thinking models. Additionally, we divide models into proprietary models and open-source models. For *Qwen* series, we use a single base model and change between instruct mode and thinking mode, which can be simply implemented by a single parameter *enable_thinking* in function *apply_chat_template*.

C.3 Loop Ratio Implementation Details

We provide detailed pseudo code of LR calculation in Algorithm 1. Moreover, for BlocksWorld, FrozenLake, Sudoku, AlfWorld and WebShop, we use exact text match to represent the state and the

Model	Trajectories w/ Loop Phenomenon		Trajectories w/o Loop Phenomenon
	AUV	Click Ratio	AUV
UI-TARS-1.5-7B	3.2	57.2	28.0
UI-TARS-72B-DPO	5.3	50.7	26.3
Qwen2.5-VL-72B-Instruct	0.4	50.4	8.3
Claude3.7-Sonnet	6.9	31.1	9.0
GPT-4o + ScaleCUA-7B	6.6	47.4	29.5

Table 9: Results on OSWorld. Trajectories are divided into two categories based on the presence of loops. Click Ratio is defined as the proportion of *Click* actions among all loop actions. *GPT-4o + ScaleCUA-7B* denotes GPT-4o as planner and ScaleCUA as GUI grounder. **Colored model name** contains proprietary model.

action. For GUI environments, we use *clip-vit-base-patch32* (Radford et al., 2021) model for picture embedding and set 0.999 as the threshold for identifying two pictures as the same state. We use exact match for same action identification.

C.4 t_{max} Configuration

The AUV metric is specifically designed to quantify the efficacy of agent environment interaction. Regarding the upper boundary t_{max} , we determine it empirically based on the performance saturation point observed across agents.

Guided by the cumulative success rate trajectories in Figure 8, we set domain-specific t_{max} values of 20, 30, 20, 60, and 15 for BlocksWorld, FrozenLake, Sudoku, AlfWorld, and WebShop, respectively. Additionally, we set t_{max} to 50 for GUI tasks following the implementation of original data.

Notably, in environments that involve multi-step tasks which cannot be solved within a single action, the AUV metric is inherently bounded away from 1. Nevertheless, an empirical upper bound of AUV can be estimated through extensive rollouts of SOTA models under the same evaluation protocol. An observed AUV value substantially lower than this empirical upper bound indicates untapped potential for improving TTI capacity, rather than a limitation imposed by the task structure itself.

C.5 Memory Recall Distance in AlfWorld

To quantitatively evaluate an agent’s ability to maintain and utilize memory of object locations across extended action sequences, we propose the Memory Recall Distance metric. This metric measures the temporal gap between observing a task-relevant object and later interacting with it.

To compute the memory recall distance in a trajectory τ with T timesteps, we first identify the set of task-relevant objects $\mathcal{G}_{task}$ by extracting target required objects from the instruction based on

the task categorization defined in the AlfWorld benchmark (Shridhar et al., 2021): $\mathcal{G}_{task} = \{obj \mid obj \text{ is required for task}\}$

Throughout the trajectory, we track when each object appears in the agent’s observations. Let $\mathcal{E}(o_t)$ denote the extraction function that returns all objects visible in observation o_t . For any object obj , we maintain a record of its most recent observation timestep prior to t :

$$LastSeen(obj, t) = \max\{k \mid k < t, obj \in \mathcal{E}(o_k)\} \quad (19)$$

where $LastSeen(obj, t)$ is undefined if obj has not been observed before timestep t .

The key measurement occurs when the agent interacts with a task-relevant object. Specifically, at timestep t , if action a_t involves interaction with object obj where $obj \in \mathcal{G}_{task}$ and $LastSeen(obj, t)$ is defined, we compute the recall distance as: $d_t(obj) = t - LastSeen(obj, t)$

This distance captures how many timesteps have passed since the agent last saw this object before deciding to interact with it. Finally, we aggregate these individual recall distances across the entire trajectory to obtain the overall memory recall distance:

$$D_{recall}(\tau) = \frac{1}{|\mathcal{I}|} \sum_{(t, obj) \in \mathcal{I}} d_t(obj) \quad (20)$$

where $\mathcal{I}$ represents the set of all valid (timestep, object) pairs in the trajectory: $\mathcal{I} = \{(t, obj) \mid obj \in \mathcal{G}_{task} \wedge \exists LastSeen(obj, t)\}$

C.6 Radar Plot Details

Specifically, we generate distinct radar profiles for each task environment. To facilitate cross-model comparison, all metrics are normalized to the unit interval $[0, 1]$ via min-max scaling across all models. Notably, regarding that LR serves as a negative

Algorithm 1 Finite State Machine-based Loop Ratio (LR) Calculation

Input: Trajectories set $\{\tau^{(1)}, \tau^{(2)}, \dots, \tau^{(N)}\}$, where each $\tau^{(k)}$ is a trajectory.

Output: LR

```
1: Initialize Global Counters:
2:    $S_{\text{total}} \leftarrow 0$  // Total interaction steps across all samples
3:    $L_{\text{total}} \leftarrow 0$  // Total loop steps across all samples
4: for  $k = 1$  to  $N$  do
5:    $\tau \leftarrow \tau^{(k)}$ 
6:    $S_{\text{total}} \leftarrow S_{\text{total}} + \text{Length}(\tau)$ 
7:   // Reset local tracking variables for the new trajectory
8:    $\mathcal{H} \leftarrow \emptyset, l_{\text{prev}} \leftarrow \emptyset, t_{\text{end}} \leftarrow -1$ 
9:   for  $t = 0$  to  $\text{Length}(\tau)$  do
10:     $h_t \leftarrow \text{Hash}(s_t)$  // Compute hash of the current state
11:    if  $h_t \in \mathcal{H}$  then
12:       $i \leftarrow \mathcal{H}[h_t]$  // Retrieve start index of potential cycle
13:       $l_{\text{curr}} \leftarrow \tau[i : t]$ 
14:      // Condition 1: Check Non-Recursive
15:      if not  $\text{HasNested}(l_{\text{curr}})$  then
16:        // Condition 2: Check Consecutive Repetition
17:        if  $(i == t_{\text{end}})$  and  $(l_{\text{curr}} == l_{\text{prev}})$  then
18:           $L_{\text{total}} \leftarrow L_{\text{total}} + (t - i)$  // Accumulate loop length into global counter
19:        end if
20:         $l_{\text{prev}} \leftarrow l_{\text{curr}}$ 
21:         $t_{\text{end}} \leftarrow t$ 
22:      end if
23:       $\mathcal{H}[h_t] \leftarrow t$  // Update the latest visit time for this state
24:    else
25:       $\mathcal{H}[h_t] \leftarrow t$  // Record the first visit time
26:    end if
27:  end for
28: end for
29: Return  $\text{LR} = L_{\text{total}} / S_{\text{total}}$ 
```

indicator of TTI performance, we apply an inversion transformation 1 - LR to ensure a consistent orientation across all axes, so that a larger value indicates a better TTI. More over, we add a minimum baseline value to make each bar visible even its corresponding indicator is zero. We also limit the maximum hight so that the highest bar does not reach the boundary of the figure. Notably, these shifts do not influence the diagnose of each model.

C.7 Framework Design

Our framework adopts a highly modular architecture with clear separation of concerns. Complex concurrency control and inference logic are encapsulated within abstract base classes, allowing researchers to focus solely on task-specific adaptations. This design significantly reduces the barrier

to entry for incorporating new evaluation benchmarks.

Task Execution The TaskRunner module implements a template method pattern providing flexible lifecycle management while minimizing the required interface surface. Researchers can leverage two complementary environment management strategies: (1) *Resource pooling* for stateful, expensive-to-initialize environments (e.g., WebShop) through the EnvPool mechanism, eliminating redundant instantiation overhead; (2) *On-demand instantiation* for environments requiring complete teardown between episodes, where researchers simply define construction logic in `_initialize_trajectory` and the framework handles per-trajectory environment generation automatically.

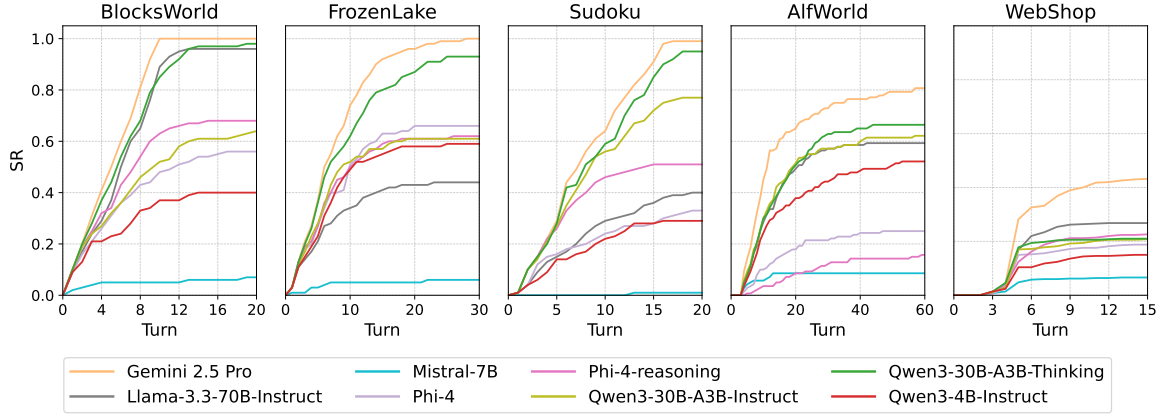


Figure 8: Cumulative success rate curves across five environments. The trajectories illustrate the performance saturation points, which serve as the empirical basis for determining the upper boundary t_{max} in our AUV metric.

Context Management The ContextManager module achieves complete separation between prompt engineering and control flow logic. Interaction histories are abstracted as structured StepMemory objects, enabling seamless switching between experimental configurations (e.g., inclusion of chain-of-thought reasoning, history truncation) without modifying underlying implementation code.

Unified Agent Interface To support fair comparison between open-weight models and proprietary APIs, we provide a unified BaseAgent interface with ClientAgent implementation wrapping OpenAI-format API calls. This abstraction allows seamless backend switching through configuration changes alone—transitioning from local *Qwen3-4B-Instruct* to remote *Gemini 2.5 Pro* requires no task code modifications. The framework incorporates thread-pooled concurrent requests with back-off retry mechanisms, ensuring robust performance despite network latency.

Parallel Execution The framework implements multi-level parallelism for large-scale evaluation on limited computational resources. At the GPU level, automatic dataset sharding distributes workloads across devices with file-lock mechanisms ensuring safe concurrent result aggregation. At the batch level, TaskRunner maintains a global trajectory pool, automatically collecting contexts from all active trajectories into unified batches for efficient inference.

Furthermore, we present the primary class structures and function interfaces below, covering: core data structures, task runner framework, agent interface, context manager, and environment interface. Notably, we are thankful to valuable works (Wang

et al., 2025b; Xue et al., 2025) for their open-sourced code.

C.8 Hyper-Parameter and Hardware Settings

We set temperature to 0.7 and top- $p = 1.0$ for all generation. For models with configurable reasoning effort (*gpt-oss-120b*, *Gemini 2.5 Pro*), we use the *medium* setting. We conduct our experiments on Nvidia-A100 GPUs.

C.9 Prompt Details

We provide our detailed prompt implementation. First, we provide the template of interaction format. Detailed system prompt can also be found following the interaction template.

```

@dataclass
class StepMemory:
    '''Stores information for a single reasoning step in the trajectory.

    Attributes:
        observation: Current environment observation
        true_state: Ground truth state from environment
        input_state: State representation input to the model
        analysis: Model's reasoning/thinking for this step
        action: Action taken by the model
        is_valid: Whether the action was valid
        feedback: Environment feedback message
        previous_memory: Link to previous step
    '''
    observation: str
    true_state: str
    input_state: str
    analysis: str
    action: str
    is_valid: bool
    feedback: str
    previous_memory: Optional['StepMemory']

@dataclass
class TrajectoryInfo:
    '''Encapsulates complete trajectory information for parallel processing.

    Attributes:
        idx_in_batch: Index within the current batch
        traj_rollout_idx: Rollout index for multiple trajectories
        env: Environment instance for this trajectory
        env_idx: Index in the environment pool
        ctx_manager: Context manager for prompt formatting
        steps: List of step dictionaries
        done: Whether trajectory is completed
        success: Whether task was successfully solved
        stop_right: Whether agent stopped at correct time
    '''
    idx_in_batch: int
    traj_rollout_idx: int
    env: BaseEnv
    env_idx: int
    ctx_manager: ContextManager
    steps: List[Dict[str, Any]]
    done: bool
    success: bool

```

Code 1: Core Data Structures

```

class TaskRunnerBase:
    '''Base class for task execution and evaluation management.'''

    def __init__(self, config: DictConfig, agent: BaseAgent,
                  dataset: List[Dict]) -> None:
        '''Initialize task runner with configuration, agent, and dataset.

        Args:
            config: Configuration object containing task and agent settings
            agent: Agent instance for generating actions
            dataset: List of task instances to evaluate
        '''

    def _prepare_step_history(self, traj: TrajectoryInfo) -> None:
        '''Prepare history and state information for current step.

        Args:
            traj: Trajectory to prepare step for
        '''

    def _process_step(self, traj: TrajectoryInfo,
                     step_info: Dict[str, Any]) -> None:
        '''Execute action in environment and update trajectory state.

        Args:
            traj: Trajectory to process
            step_info: Dictionary containing action and analysis
        '''

    def _initialize_trajectory(self, data_idx: int, data: dict,
                              traj_rollout_idx: int) -> TrajectoryInfo:
        '''Initialize single trajectory with environment and context.

        Args:
            data_idx: Index in dataset
            data: Data instance dictionary
            traj_rollout_idx: Rollout index for this trajectory

        Returns:
            TrajectoryInfo: Initialized trajectory object
        '''

    def run(self, dp_idx: int = 0,
            lock: Optional[Lock] = None) -> None:
        '''Main entry point for running evaluation on dataset.

        Args:
            dp_idx: Starting index in dataset
            lock: Optional lock for thread synchronization
        '''

    def _run_stepwise_episode_batch(self,
                                    batch: List[Dict]) -> List[Dict]:
        '''Run batch of episodes with step-by-step execution.

        Args:
            batch: List of data instances

        Returns:
            List of trajectory results with multiple rollouts per instance
        '''

```

Code 2: Task Runner Framework

```

class BaseAgent:
    '''Abstract base class for all agent implementations.'''
    def __init__(self, config: DictConfig):
        '''Initializes the agent with task-specific configurations.

        Args:
            config: Configuration object containing model and generation parameters.
        '''

    def get_next_step_parallel(self,
                              trajectories: List[TrajectoryInfo]
                              ) -> List[Dict[str, Any]]:
        '''Generate actions for multiple trajectories in parallel.

        Args:
            trajectories: List of trajectories to process

        Returns:
            List of action dictionaries for each trajectory
        '''

    def close(self) -> None:
        '''Clean up agent resources and connections.'''

```

Code 3: Agent Interface

```

class ContextManager:
    '''Manages conversation history and prompt formatting.'''

    def __init__(self, system_prompt: str, instruction_prompt: str,
                 tokenizer: AutoTokenizer, config: DictConfig) -> None:
        '''Initialize with prompts and formatting configuration.

        Args:
            system_prompt: System-level instruction prompt
            instruction_prompt: Task-specific instruction
            tokenizer: Tokenizer for applying chat templates
            config: Configuration for formatting options
        '''

    def format_prompt(self) -> str:
        '''Format full prompt string based on chat format and history.

        Returns:
            Formatted prompt string including conversation history

        Supported formats:
            - 'default_format': Standard prompt format
            - 'user_assistant_format': Dialogue-based format
            - 'user_assistant_format_part': Partial dialogue format
        '''

    def format_messages(self) -> List[Dict[str, str]]:
        '''Format as message list for API-based agents.

        Returns:
            List of message dictionaries with 'role' and 'content' keys
        '''

```

Code 4: Context Manager

```

class BaseEnv:
    '''Abstract base class for task environments.'''

    def reset(self, seed: int, mode: str) -> None:
        '''Reset environment to initial state.

        Args:
            seed: Random seed for reproducibility
            mode: Evaluation mode ('easy', 'medium', 'hard')
        '''

    def step(self, action: str) -> Tuple[Any, float, bool, Dict]:
        '''Execute action and return environment response.

        Args:
            action: Action string to execute

        Returns:
            Tuple containing:
            - observation: New environment observation
            - reward: Reward signal
            - done: Whether episode is terminated
            - info: Additional information dictionary
        '''

    def render(self) -> str:
        '''Generate string representation of current state.

        Returns:
            Human-readable string of environment state
        '''

    @property
    def instruction_text(self) -> str:
        '''Return task instruction or query text.

        Returns:
            Task-specific instruction string
        '''

```

Code 5: Environment Interface