Coarse-to-Fine Grounded Memory for LLM Agent Planning

Wei Yang[1], Jinwei Xiao[1], Hongming Zhang[1]†, Qingyang Zhang[1], Yanna Wang[1], Bo Xu[1]†
[1]National Key Laboratory of Cognition and Decision Intelligence for Complex Systems,
Institution of Automation, Chinese Academy of Sciences
{yangwei2023,xiaojinwei2024,hongming.zhang}@ia.ac.cn
{zhangqingyang2019,wangyanna2013,boxu}@ia.ac.cn
†**Correspondence:** hongming.zhang@ia.ac.cn, boxu@ia.ac.cn

Abstract

Recent advancements in Large Language Models (LLMs) have driven growing interest in LLM-based agents for complex planning tasks. To avoid costly agent training, many studies adopted memory mechanism that enhances LLM with offline experiences or online trajectory analysis. However, existing works focus on single-granularity memory derived from dynamic environmental interactions, which are inherently constrained by the quality of the collected experiences. This limitation, in turn, constrain the diversity of knowledge and the flexibility of planning. We propose Coarse-to-Fine Grounded Memory (CFGM), a novel framework that grounds coarse-to-fine memories with LLM, thereby fully leverage them for flexible adaptation to diverse scenarios. CFGM grounds environmental information into coarse-grained focus points to guide experience collection in training tasks, followed by grounding of actionable hybrid-grained tips from each experience. At inference, CFGM retrieves task-relevant experiences and tips to support planning. When facing environmental anomalies, the LLM grounds the current situation into fine-grained key information, enabling flexible self-QA reflection and plan correction. Extensive experiments on AlfWorld, Webshop and ScienceWorld demonstrate that CFGM significantly outperforms competitive baselines and comprehensively optimizes memory-enhanced LLM Agent system.

1 Introduction

The remarkable progress of LLMs has ignited intense interest in their deployment as autonomous agents capable of tackling complex, long-horizon tasks within interactive environments (Wang et al., 2023; Xi et al., 2023; Gur et al., 2024; Wang et al., 2024a). These LLM-based agents leverage extensive world knowledge (Zhao et al., 2023) and reasoning capabilities to formulate policies (Yao et al., 2023a). However, significant challenges hinder their practical effectiveness. Agents often suffer from inefficient exploration, yielding sparse or irrelevant data (Arumugam and Griffiths, 2025; Zhang et al., 2025a,c). They also struggle with reasoning derailment in partially observable environments, where incomplete observations cause cascading errors (Kim et al., 2024). Existing methods attempt to mitigate these issues using memory mechanisms, such as storing offline experiences and insights (Zhao et al., 2024a), extracting context-aware guidelines (Fu et al., 2024b), or checking belief states with fixed-question reflection (Kim et al., 2024).

However, recent studies depend on single-granularity memory derived from dynamic environmental interactions, whose effectiveness is limited by the quality of the collected experiences. This constraint consequently reduces the variety of knowledge and limits the adaptability of planning. These limitations lead to explicit outcomes manifested in three aspects: (1) When confronted with challenging planning tasks, the initial exploration efficiency of LLMs is often suboptimal, constraining the quality of memory. (2) Current approaches extract knowledge either at a holistic level or a scenario-specific level, both of which lead to homogenized granularity in the acquired knowledge. (3) Although directly using manually designed questions to reflect on trajectory information in short-term memory can yield decent results, it lacks generalizability across environments.

To bridge this gap, we introduce **C**oarse-to-**F**ine **G**rounded **M**emory (CFGM), a novel agent framework that leverages the internal knowledge of LLM to guide the memory collection and utilization, thereby enhancing their planning capabilities in complex environments. As illustrated in Figure 1, CFGM operates through three progressively memory grounding stages to ensure that the agent's planning and actions remain continuously anchored in LLM's prior understanding of memory. Guided by

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 13041–13068
November 4-9, 2025

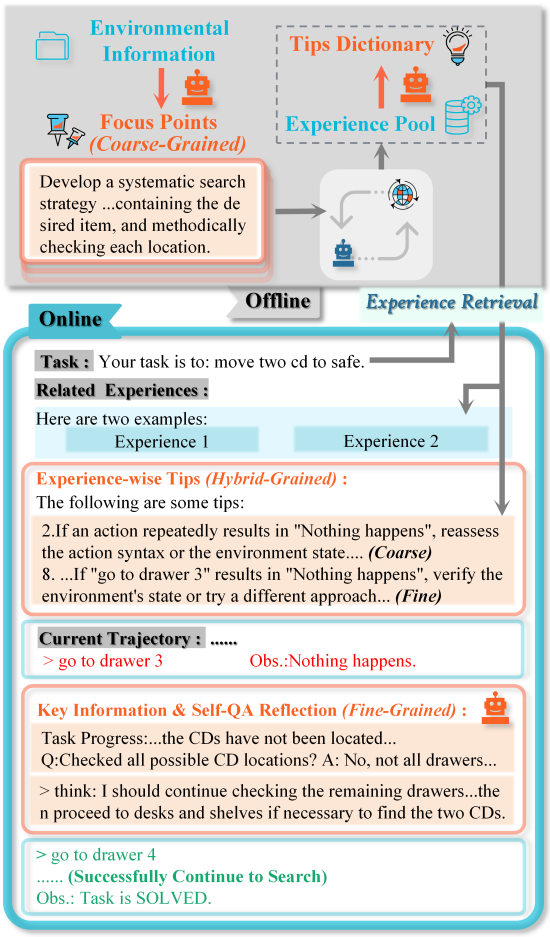

Figure 1: Conceptual overview of CFGM. During the offline training, our method first extracts coarse-grained focus points to guide experience collection, then distills hybrid-grained tips from these experiences. At the online inference time, it retrieves relevant experiences and tips for planning. When encountering anomalies, the system identifies fine-grained key details for adaptive self-QA and plan adjustment.

environmental information (initial memory), the LLM first identifies coarse-grained focus points for task planning. This preliminary understanding enables the collection of higher-quality experiences. For each one of the offline experiences (long-term memory), LLM grounds the knowledge it contains into high-level principles and detailed techniques, thereby constructing an experience-wise hybrid-grained tips dictionary. During online interactions, the most relevant successful experiences and tips are retrieved to enhance LLM planning as context. Upon facing environmental anomalies, the agent conducts a fine-grained key information analysis of the current state (short-term memory) , followed by a self-QA reflection grounded in both past successes and the present situation to effectively correct planning. By grounding explicit coarse-to-fine memories with LLM's internal knowledge, CFGM enables the LLM to autonomously consider diverse information in complex planning environments and optimizes the collection and utilization of memory based on its reasoning outcomes. Because of this, CFGM is able to enhance the quality of collected memories, diversify experiential knowledge, and provide more flexible error-correction support during planning. Finally, this enables LLMs to tackle complex tasks more effectively, even preserving compatibility with powerful closed-source models. We evaluate CFGM on three diverse interactive planning benchmarks. Our results demonstrate that CFGM significantly outperforms existing memory-enhanced agent systems, showcasing improved performance and robustness.

Our three key contributions are as follows: (i) We propose Coarse-to-Fine Grounded Memory (CFGM), a novel framework that enhances LLM agents by systematically grounding memory with LLM's internal knowledge during experiences collection, tips extraction and adaptive planning. (ii) We validate the effectiveness of CFGM across multiple challenging benchmarks, achieving state-of-the-art performance compared to strong memory-enhanced LLM agent system baselines. (iii) We perform comprehensive ablation studies and analyses on the framework's core components, which show coarse-to-fine grounded memory enhances the quality of collected memory experiences, constructs hybrid-grained tips with more useful information in offline phase, and enables flexible and effective reflection during online planning.

2 Related Work

2.1 LLM-based Agents

As an emerging technology, LLM-based agents employ LLM which provides language understanding and generation capabilities as the core of planning (Yue, 2025). Compared to traditional AI agents, LLM-based agents demonstrate higher levels of autonomy and reasoning capabilities (Luo et al., 2025). LLM-based agents have been applied in various domains, including software engineering (Jin et al., 2024), question answering (Yue, 2025), scientific research (Xiao et al., 2024), web navigation (Deng et al., 2023), mathematical reasoning (Ahn et al., 2024), embodied intelligence (Zhao et al., 2024b), and search and recommendation systems (Zhang et al., 2025b). In

this paper, we aim to construct high-performance agents targeting diverse planning task with coarse-to-fine grounded memory.

2.2 Experience-based Optimization for Agents

Experience-based optimization focuses on improving the performance of agents by leveraging historical data and accumulated knowledge (Li et al., 2024b,a; Zhao et al., 2024a; Bai et al., 2024a; Chen et al., 2024; Fu et al., 2024b; Qian et al., 2024; Fu et al., 2024a; Zhang et al.). This optimization provides a mechanism for LLM agents to learn and adapt over time without adjusting the model's parameters (Du et al., 2025; Zheng et al., 2025; Song et al., 2024). ExpeL (Zhao et al., 2024a) exemplifies this approach through prompt-based learning with experience recall and insight extraction. AutoGuide (Fu et al., 2024b) extracts and applies context-aware guidelines from experiences. However, these methods often struggle with inefficient exploration in complex planning tasks due to limited prior task understanding, leading to suboptimal memory collection. Furthermore, when extracting knowledge from memories, they typically rely on single-grained information, restricting the agent's holistic comprehension. To overcome these issues, CFGM leverages LLM-generated focus points to guide exploration toward critical dynamics, reducing redundancy. Additionally, it extracts hybrid-grained tips from experiences to enhance LLMs' comprehensive memory understanding.

2.3 Feedback-based Optimization for Agents

Feedback-based optimization involves using various types of online feedback to guide the learning and improvement of agents. This feedback can come from external models (Bo et al., 2025; Huang et al., 2023; Yao et al., 2023b; Zhang et al., 2024a; Bai et al., 2024b; Sun et al., 2025), or the agent itself (Huang et al., 2024; Wang et al., 2024b; Liang et al., 2024; Zhang et al., 2024b; Zhou et al., 2024; Shinn et al., 2023; Cui et al., 2025). Self-reflection and self-correction enable agents to iteratively improve their outputs by identifying and addressing errors (Luo et al., 2025). Reflexion (Shinn et al., 2023) preforms trial-and-error attempts on online inference trajectories, reflects on the mistakes, and applies the lessons learned to subsequent attempts. QuBE (Kim et al., 2024) conducts belief-state reflection with fixed-question templates, which addresses errors in tasks that allow only a single attempt. To enable more flexible short-term memory reflection in single-attempt planning compared to existing approaches, CFGM employs adaptive Self-QA reflection that dynamically leverage key information extracted from online trajectory information with LLM.

3 Method

To tackle the current challenges that memory-augmented agents encounter during planning, we introduce an LLM-based agent system characterized by a coarse-to-fine grounding of the model's internal knowledge into explicit memories. This system leverages inherent LLM knowledge to enhance the quality of collected memories, diversify high-level experiential knowledge, and provide more flexible error-correction support during planning, ultimately enabling LLMs to handle complex decision-making tasks more effectively. The framework is depicted in Figure 2 and comprises three components: coarse-grained focus-driven experience collection, hybrid-grained experience-wise tips extraction and fine-grained key information adaptive planning. The prompt template related to memory grounding in the entire framework are presented in Appendix C.

Algorithm 1 Coarse-Grained Focus-Driven Experience Collection.

$N = len(\mathcal{T}_{\text{train}}), n = 0$
$FP = \text{LLM}_{\text{Focus}}(Desc_{\text{env}}, F_{\text{manual}})$
while task $n < N$ **do**
  $t_n \leftarrow \mathcal{T}_{\text{train}}[n], \nu_{n,0} \leftarrow$ ""
  for trial $z = 0$ to Z **do**
    Initialize Trajectory $\tau_{n,z} \leftarrow$ env.reset(t_n)
    for timestep $i = 0$ to H **do**
      $a_i \leftarrow \text{LLM}_{\text{ReAct}}(a_i \mid \tau_{n,z}, F_{\text{manual}}, \nu_{n,z}, FP)$
      o_{i+1}, done $\leftarrow$ env.step(a_i)
      $\tau_{n,z} \leftarrow \tau_{n,z} \cup \{(a_i, o_{i+1})\}$
      if done **then**
        break
      end if
    end for
    $\mathcal{B} \leftarrow \mathcal{B} \cup (t_n, \tau_{n,z})$
    if done or $z = Z$ **then**
      $n \leftarrow n + 1$
      break
    else
      $\nu_{n,z+1} \leftarrow \text{concat}(\nu_{n,z} + \text{LLM}_{\text{Reflect}}(\tau_{n,z}))$
    end if
  end for
end while
return $\mathcal{B}$

3.1 Coarse-Grained Focus-Driven Experience Collection

To collect usable and diverse experiential memories, we adopted a synergistic approach Reflex-

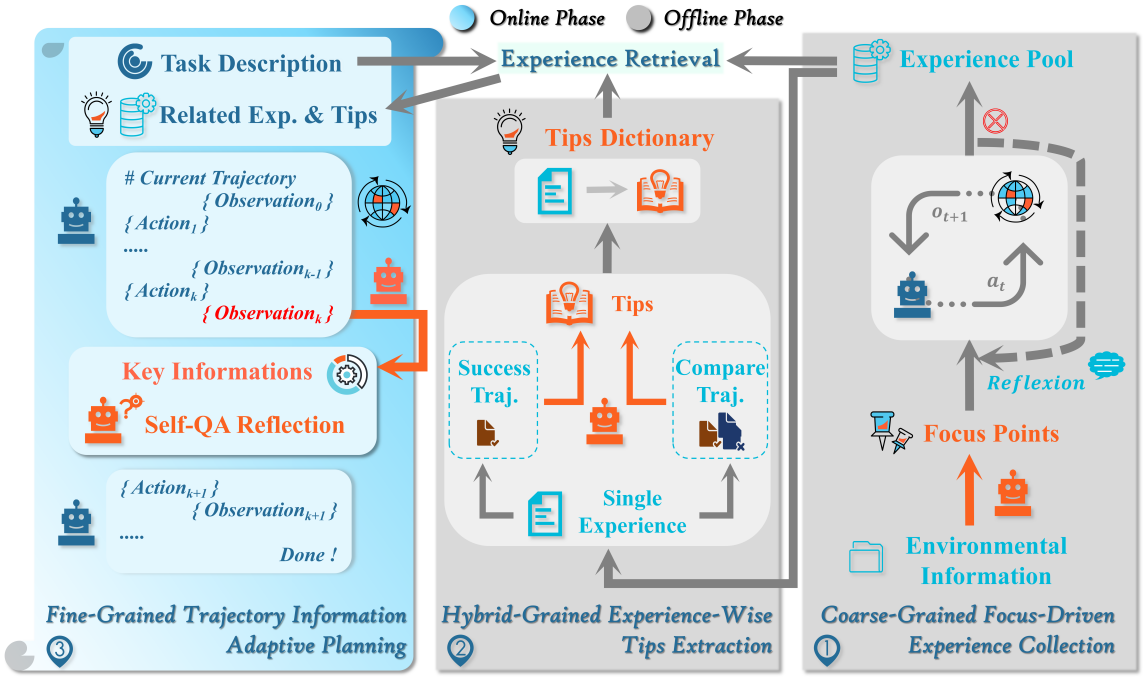

Figure 2: Framework of CFGM. CFGM collect experiences offline with coarse-grained focus points grounded from environmental information, followed by extracting the hybrid-grained tips grounded from trajectories of each experience to construct tips dictionary. Then, agent's online planning will be enhanced by retrieved experiences and tips, in which the fine-grained self-QA reflection is activated by the key information grounded from current situation and relevant history when anomaly observed. The orange arrow represents the memory grounding process.

ion (Shinn et al., 2023) for collaborative thinking and retry-reflection to collect diverse experiences. Additionally, we enabled the LLM to ground environmental foundational information to obtain focus points that guide LLM with their own preliminary understanding of the environment to collect higher-quality experience. The workflow of this component is illustrated in the pseudo-code of Algorithm 1. Specifically, at the initiation of experience collection, the agent $\text{LLM}_{\text{Focus}}$ conducts a comprehensive analysis of the environmental description $Desc_{\text{env}}$ and manual trajectories few-shot F_{manual} using its internal knowledge. Through coarse-grained examination of task-related foundational information, the agent can generate some preliminary thoughts for handling the environmental tasks. Guided by these focus points FP on training task sets $\mathcal{T}_{\text{train}}$, the agent $\text{LLM}_{\text{ReAct}}$ executes up to $Z+1$ rounds of trial attempts involving think-act iterations. During this process, failures trigger introspective reflection $\text{LLM}_{\text{Reflect}}(\tau_{n,z})$, with updates incorporated into task-specific reflections $\nu_{n,z+1}$ to assist subsequent attempts until task t_n completion. High-quality trajectory information $\tau_{n,z}$ collected under focus points guidance is stored in the experience pool $\mathcal{B}$ with the corresponding task t_n. Both successful and failed trajectories in success task will subsequently support tip generation and key information reflection phases.

Algorithm 2 Hybrid-Grained Experience-Wise Tips Extraction.

$N = len(\mathcal{T}_{\text{train}})$
Construct fail/success tuples of the same tasks in $\mathcal{B}$:
$C_{\text{compare}} = \{t_1 : (\tau_1^{\text{success}}, \tau_{1,0}^{\text{fail}}, ...), ...,$
$\quad t_i : (\tau_i^{\text{success}}, \tau_{i,0}^{\text{fail}}, ...), ...\}$
Construct success trajectories in $\mathcal{B}$:
$C_{\text{success}} = \{t_1 : \tau_1^{\text{success}}, ..., t_i : \tau_i^{\text{success}}, ...\}$
for task $m = 0$ to $N-1$ **do**
  $t_n \leftarrow \mathcal{T}_{\text{train}}[n]$
  if $t_n \in C_{\text{compare}}$ **then**
    $TD[t_n] = \text{LLM}_{\text{Tips}}(C_{\text{compare}}[t_n])$
    $TD[t_n] \leftarrow (TD[t_n],$
      $\text{LLM}_{\text{Tips}}(C_{\text{success}}[t_n], TD[t_n]))$
    continue
  end if
  if $t_n \in C_{\text{success}}$ **then**
    $TD[t_n] = \text{LLM}_{\text{Tips}}(C_{\text{success}}[t_n]))$
  end if
end for
return TD

3.2 Hybrid-Grained Experience-Wise Tips Extraction

Because the experience collection stage follows a repeated trial-and-error paradigm (Sutton and

Barto, 2018; Dong et al., 2020), many tasks stored in the experience pool are accompanied by both successful and failed trajectories. These trajectories contain not only fine-grained scene-specific details but also more high-level coarse-grained insights that are applicable across similar tasks. We refer to such hybrid-grained knowledge as tips. To help the LLM make better use of this hybrid-grained information, the current module leverages the model's internal knowledge offline to convert the multiple trajectories of a given task experience into a set of tips that blend hybrid-grained perspectives.

The complete workflow of this module is illustrated in Algorithm 2. We begin by re-indexing all trajectory data in the experience pool $\mathcal{B}$ according to their associated task t_n. Trajectories that contain both successful and failed attempts are aggregated into the compare experience dictionary C_{compare}, while those consisting a successful attempt are placed in the success experience dictionary C_{success}. Next, we iterate over every training task t_n. If t_n appears in the C_{compare}, we prompt the LLM_{Tips} to compare the task's successful and failed trajectories and, drawing on its internal knowledge, extract tips that help prevent the observed errors. We then ask the LLM to focus exclusively on successful trajectories to refine additional success-specific factors to expand tips. The resulting tips are stored under the corresponding task t_n in the Tips Dictionary TD. For tasks that the LLM can solve directly in a successful attempt, we perform the same reflective analysis on their successful trajectories, aiming to derive a moderate number of tips that remain applicable in similar scenarios.

3.3 Fine-Grained Trajectory Information Adaptive planning

After completing the offline collection of experiences and extraction of relevant hybrid-grained knowledge, we need to dynamically utilize these auxiliary information online during evaluation based on fine-grained current status information. In addition to leveraging auxiliary information, the agent model inevitably encounters unexpected scenarios during evaluation and planning. To mitigate the resulting planning derailment, we employ LLM to perform adaptive grounding of corrective planning by analyzing the fine-grained current trajectory status and analogous successful cases.

The detailed workflow of the aforementioned evaluation reasoning is presented in Algorithm 3. For task t_m from evaluation set $\mathcal{T}_{\text{evaluation}}$, we conduct a prior analysis based on task description retrieval before inference. Using the Faiss library (Johnson et al., 2019), we retrieve the top-k most similar experiences E_{sim} from the Experience Pool $\mathcal{B}$ with textual embedder $\mathcal{E}$. These experiences' successful trajectory contents τ_h^{success} and corresponding tips knowledge $TD[t_h]$ are integrated into Similar Trajectories content ST and relevant Experience Tips ET , which serve as contextual assistance for the decision-making of policy model LLM_{CFGM} during inference. During the reasoning process for task t_m, we implement real-time monitoring of observed information o_{i+1} and trigger alerts for specific abnormal observations. When triggered, we employ the Key Information Extraction model LLM_{KIE} to perform a structured, fine-grained analysis of the current trajectory content τ_m, thereby obtaining grounded current trajectory content as key information KI. Subsequently, by synthesizing KI, τ_m, and ST as contextual elements, we further instruct the Key Information Reflection model LLM_{KIR} to conduct self-reflective question-answer qa_i, ultimately deriving a final corrective plan ref_i. The ref_i is then integrated into the current trajectory content to guide LLM_{CFGM} in implementing adaptive responses to exceptional situations.

Algorithm 3 Fine-Grained Trajectory Information Adaptive Planning.

$M = len(\mathcal{T}_{\text{evaluation}})$
for task $m = 1$ to M **do**
  $t_m \leftarrow \mathcal{T}_{\text{evaluation}}[m]$
  Initialize trajectory $\tau_m \leftarrow$ env.reset(t_m)
  $E_{\text{sim}} \leftarrow \texttt{Faiss}(t_m, \mathcal{B}, \mathcal{E}, k)$
  $ET \leftarrow$ "", $ST \leftarrow$ ""
  for experience task $(t_h, \tau_h^{\text{success}})$ in $E_{\text{similar tasks}}$ **do**
    $ET \leftarrow \text{concat}(ET + TD[t_h])$
    $ST \leftarrow \text{concat}(ST + \tau_h^{\text{success}})$
  end for
  for timestep $i = 1$ to H **do**
    $a_i \leftarrow \text{LLM}_{\text{CFGM}}(a_i \mid \tau_m, ET, ST)$
    o_{i+1}, done $\leftarrow$ env.step(a_i)
    current step $s_i \leftarrow (a_i, o_{i+1})$
    if $trigger(o_{i+1})$ **then**
      $KI \leftarrow \text{LLM}_{\text{KIE}}(\tau_m)$
      $(qa_i, ref_i) \leftarrow \text{LLM}_{\text{KIR}}(\tau_m, KI, ST)$
      $s_i \leftarrow \text{concat}(s_i + ref_i)$
    end if
    $\tau_m \leftarrow \tau_m \cup \{s_i\}$
    if done **then**
      break
    end if
  end for
end for

4 Experiments

4.1 Experimental Settings

We evaluate CFGM on three challenging interactive environment to assess its effectiveness and capabilities, particularly how its coarse-to-fine grounding mechanism enhance LLM agent performance. We refer Appendix A for detailed information of main environments and WebArena (Zhou et al., 2023) used in ablation study.

Environments. The basic introduction of environment is as follows: (i) AlfWorld is an virtual household agent benchmark where the agent performs household tasks based on natural language instructions. It requires navigation and object manipulation (Shridhar et al., 2020). (ii) WebShop is an interactive environment simulating online shopping, where the agent needs to find and purchase a product matching specific criteria by interacting with a website (Yao et al., 2022). (iii) ScienceWorld is a text-based environment focusing on elementary science tasks, requiring complex reasoning and interaction with the simulated world (Wang et al., 2022).

Evaluation. Following standard practices in interactive agent evaluation, we use Success Rate (SR) as the primary metric across all environments. Specifically, SR measures completing the task within time limits for AlfWorld, purchasing the item that matches all attributes for WebShop, and successful task completion for ScienceWorld. For WebShop, we also report the mean reward, which quantifies the quality of the purchased product based on the matching of required attributes.

Baselines. We compare CFGM against some mainstream LLM agent methods, most of which are memory-augmented agents: (i) ReAct is a foundational method that interleaves reasoning steps with actions in the environment (Yao et al., 2023a). (ii) ExpeL learns from past experiences by extracting insights and retrieving relevant trajectories (Zhao et al., 2024a). (iii) AutoGuide generates context-aware guidelines from offline experiences to guide the agent at test time (Fu et al., 2024b). (iiii) QuBE enhances agent reasoning by constructing belief state reflection with human-curated questions to alleviate reasoning derailment (Kim et al., 2024).

Implementation Details. Unless otherwise specified (e.g., in model generalization studies), we use GPT-4-Turbo as the agent model and GPT-4o as the knowledge extraction model (like insight extraction model in ExpeL) for all baselines and CFGM. Additional implementation details are presented in Appendix B.

4.2 Main Results

Table 1 presents the main performance comparison of CFGM (Ours) against the baselines for the three benchmarks. Based on these results, we conduct separate analyses focusing on the framework's effectiveness and environmental generalizability.

Effectiveness of CFGM. As shown in Table 1, CFGM outperforms ReAct across all environments: AlfWorld SR (+10.40%), WebShop SR (+20%), and ScienceWorld SR (+31%). This performance boost is due to CFGM's coarse-to-fine grounding approach, which actively integrates LLM knowledge into structured memories (Zhang et al., 2023). Compared to other memory-augmented agent baselines, our method also achieves superior task performance. This improvement stems from two key aspects: firstly, our approach leverages both offline experiences from long-term memory to assist online planning and incorporates short-term memory reflection on current trajectories; secondly, we utilize the internal knowledge of LLMs to ground the memories, enabling the agent to better comprehend and utilize past experiences. Additionally, CFGM consistently and significantly outperforms the ExpeL + QuBE baseline. This demonstrates CFGM's superiority is not merely an artifact of combining two orthogonal techniques.Furthermore, we wish to re-emphasize that CFGM is not a simple combination of existing ideas. Its components are deeply integrated and synergistic with coarse-to-fine grounded memory. Specifically, for example, the coarse-grained Focus Points are not just an independent pre-processing step; they fundamentally improve the quality of the memories collected. This, in turn, enhances the effectiveness of both the hybrid-grained Experience-wise Tips and the fine-grained Key Information Reflection. This synergistic "1+1>2" effect is a core contribution of our work and distinguishes CFGM from a simple integrated pipeline of unrelated modules.

Robustness in Diverse Environments. CFGM shows strong scalability and robustness, achieving significant gains in diverse environments. Specifically, it reaches 91.00% SR in AlfWorld, 0.85 Reward and 57% SR in WebShop, and 74% SR in ScienceWorld. These tasks range from long action sequences planning to dynamic web navigation and scientific experiments, where agents typically struggle with planning derailment and inefficient

Method	Features			AlfWorld	WebShop		ScienceWorld
	Off. Exp.	Traj. Ref.	Gro. Mem.	Success Rate (SR)↑	Reward↑	SR↑	SR↑
ReAct	✗	✗	✗	80.60% ± 0.68%	58.6 ± 1.0	37% ± 2%	43% ± 1%
ExpeL	✓	✗	✗	81.34% ± 0.85%	62.2 ± 1.3	42% ± 3%	57% ± 2%
AutoGuide	✓	✗	✗	83.58% ± 0.77%	73.3 ± 1.4	47% ± 2%	N/A
QuBE	✗	✓	✗	84.33% ± 0.73%	N/A	N/A	N/A
ExpeL + QuBE	✓	✓	✗	<u>85.07% ± 1.03%</u>	<u>75.6 ± 1.3</u>	<u>49% ± 3%</u>	<u>65% ± 2%</u>
CFGM	✓	✓	✓	**91.00% ± 0.82%**	**85.0 ± 1.3**	**57% ± 3%**	**74% ± 2%**

Table 1: Main results across AlfWorld, WebShop, and ScienceWorld benchmarks. Off. Exp. refers to collecting offline experiences, Traj. Ref. refers to reflection based on trajectories during online inference, and Gro. Mem. refers to leveraging grounded memory with LLM's knowledge. Best in bold, second-best underlined. CFGM consistently outperforms the baselines on all domains, highlighting the importance of comprehensive memory grounding and the scalability of CFGM in diverse environments.

exploration. These results demonstrate the generalization capability of CFGM's memory grounding approach across different environments. Detailed CFGM's experimental trajectory examples for each environment are provided in the Appendix D with ReAct's comparative trajectories.

4.3 Ablation Studies

After evaluating the performance of the framework, we conduct extensive experiments to assess the framework's effectiveness and the design of memory grounding process of each component.

Effectiveness of Different Components. Fistly, we evaluate the impact of each core component of CFGM—coarse-grained focus points (FP), hybrid-grained experience tips (ET), and fine-grained key information reflection (KIR)—by adding them individually or in combination to a ReAct baseline on AlfWorld. As shown in Table 2, adding coarse-to-fine grounded memory individually provides consistent gains over the ReAct baseline. This demonstrates that grounded memory based on the LLM's internal knowledge enhances the agent's planning capability across all granularity levels. By integrating all three components, CFGM achieves the highest SR of 91.00%, demonstrating the enhancement effects of grounded memory at different granularities on LLM planning are cumulative and validating the rationality of our coarse-to-fine framework design. It is worth noting that removing FP causes SR to drop to 88.06 %, showing FP enhances the positive effects of ET and KIR on LLM planning performance by improving the quality of collected memories.

Generalizability across Different Models. Having examined the detailed mechanisms of individual component, we proceed to evaluate the framework's overall generalizability across diverse agent models. For closed-source GPT-based models up to GPT-4o, we choose GPT-4o as knowledge extraction model. For other models (Qwen, GPT-4.1), the component rels on the model itself, allowing evaluation of the framework's self-sufficiency with different base LLMs. As presented in Figure 3, from smaller open-source ones (Qwen2.5-7B-Instruct) to powerful closed-source ones (GPT-4.1), CFGM consistently outperforms both ReAct and ExpeL by a significant margin, which proves that the grounded memory in CFGM exhibits universal effectiveness in enhancing planning capabilities across different LLM models. Having analyzed the effectiveness and generalizability of grounded memory, we further explore the design and capability of grounded memory at key component.

FP	✗	✓	✗	✗	✗	✓
ET	✗	✗	✓	✗	✓	✓
KIR	✗	✗	✗	✓	✓	✓
SR%	80.60	85.82	86.57	85.82	88.06	91.00

Table 2: Module ablation experiment results on AlfWorld. FP refers to using focus points to guide experience collection, ET refers to extracting experience-wise tips, and KIR refers to question-answer self-reflection based on key information during online planning.

Impact of Experience Retrieval Scope on Tips Quality. For the hybrid-grained tips used online, we investigate how the number of retrieved experiences (top-k) affects the tips' quality. Specifically,

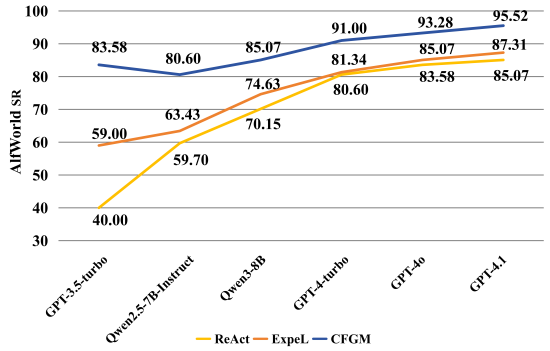

Figure 3: The SR achieved by different methods using various models on AlfWorld. CFGM demonstrates strong generalization across different models and consistently outperforms the baselines.

we evaluate the impact by the SR on AlfWorld using a ReAct + ET setup. As shown in 3, the baseline ReAct ($k = 0$) has an SR of 80.60%. Introducing just one relevant experience's tips ($k = 1$) improves SR to 82.09%. Performance peaks at $k = 2$ and $k = 3$, both achieving 86.57% SR. Increasing k further to 5 leads to a drop in performance (81.34%), due to the inclusion of less relevant or potentially conflicting hybrid-grained tips. This suggests that retrieving a moderate, focused set of the most relevant experiences ($k = 2$ or 3) provides the highest quality hybrid-grained tips to guide the LLM agent.

k	0	1	2	3	5
SR	80.60%	82.09%	86.57%	86.57%	81.34%

Table 3: The impact of different Top-k experience retrieval scope on the quality of tips on AlfWorld. The retrieved top-2 experiences yield tip sets with optimal quality and retrieval efficiency.

Experience Tips' Generalizability to Out-of-Distribution Environments. To test the out-of-domain capability of tips across different domains but relevant tasks, we conduct a transfer learning experiment from WebShop to WebArena-Shopping that is a distinct domain but contain relevant tasks. We extract hybrid-grained tips from WebShop experiences and apply them to WebArena-Shopping. For the domain adaptation, we align the tips to WebArena-Shopping based on GPT-4-Turbo. As presented in Table 4, the transferred tips achieves 25.1% SR, significantly outperforming all the baselines, which demonstrates that the hybrid-grained tips grounded from experiences can capture generalizable knowledge, enabling effective transfer to related but distinct out-of-domain environments.

Method	**WebArena–Shopping SR**
ReAct	10.2% ± 0.5%
ExpeL	18.5% ± 0.9%
AutoGuide	20.4% ± 0.7%
CFGM	25.1% ± 0.8%

Table 4: Out-of-distribution generalization of different method's extracted experience knowledge from WebShop on the 98 WebArena–Shopping tasks. hybrid-grained tips of CFGM perform superior out-of-domain generalization compared to existing knowledge extraction agents after transfer.

Comparative Analysis of Online Reflection. To explore the rationality of KIR's memory grounding design, we analyze the effectiveness of different online reflection mechanisms on AlfWorld. Table 5 compares KIR against two other reflection variants: a template-free Self-QA reflection without key information and a QA reflection via fixed questioning template like QuBE (Kim et al., 2024)), both applied on top of a ReAct agent. As shown in Table 5, KIR demonstrates superior corrective assistance for agent planning compared to QA Reflection (Fixed Question), which indicates that fine-grained grounded memory can achieve more effective reflection outcomes with greater flexibility than manually human-crafted question-answering reflection approaches. The performance improvement of KIR over Self-QA Reflection further demonstrates that prior analysis of short-term memory can significantly enhance subsequent comprehension and reasoning effectiveness when utilizing grounded memory.

Method	**AlfWorld SR**
Self-QA Reflection	84.33%
QA Reflection(Fixed Question)	84.33%
Key Information Reflection	85.82%

Table 5: The comparative analysis of different reflection. Fine-grained grounded memory enables more flexible and effective reflection than manual QA approaches. Prior short-term memory analysis (key information) further enhances comprehension and reasoning when leveraging grounded memory.

The Overall Efficiency of CFGM. To assess the practical efficiency of CFGM compared against the baseline methods, we analyzed the token consumption and interaction turns of the framework

on the AlfWorld benchmark. As shown in Table 6, during offline processing, CFGM's token cost is comparable to ExpeL and notably more efficient than AutoGuide, which show the efficient offline traininng process of CFGM. Additionally, in the online phase, CFGM is the most efficient framework among the memory-augmented baselines in terms of both tokens per step and average interaction turns. This reduction in steps can lead to lower overall latency and cost, demonstrating that the comprehensive memory grounding leads to more efficient, less redundant agent behaviors.

Method	Off. Tokens	On. Tokens	On. Turns
ReAct	-	1734.6	19.01
ExpeL	3888.2	5125.5	17.32
AutoGuide	4873.4	4809.4	16.55
QuBE	-	4752.3	17.21
CFGM	4068.5	4681.4	14.32

Table 6: Average token consumption for processing a single training data sample during the offline phase on AlfWorld (Off. Tokens). Average token consumption per step and average interaction turns per complete trajectory during online inference on AlfWorld (On. Tokens and On. Turns).

5 Conclusion

In this paper, we propose an LLM agent system enhanced with coarse-to-fine grounded memory. CFGM relies on the LLM's intrinsic knowledge to analyze coarse-grained environmental information, hybrid-grained experiences' trajectories, and fine-grained current trajectory. This process yields knowledge that enhances the LLM's comprehension and interactive performance in complex planning scenarios. Experimental results across three challenging benchmarks demonstrate the superiority of CFGM over existing memory-enhanced LLM agent systems. Through comprehensive optimization of the agent's framework, CFGM obtains higher-quality memory, more comprehensive experiential knowledge, and more flexible online reflection compared to other baselines. Additional experiments further validate the rationality and effectiveness of each component of CFGM. These findings offer valuable insights into the full utilization of memory knowledge in LLM agent systems.

Limitations

As a memory-augmented LLM Agent system, our work focuses primarily on memory optimization utilization through LLM internal knowledge grounding, yet still presents several limitations: 1) Although the collection efficiency for training tasks has improved, it remains challenging to generate sufficient effective experiences when confronted with extremely small training sets. Therefore, how to effectively expand limited training task sets constitutes an important research direction for future investigation. 2) While leveraging experiences from similar tasks during online inference demonstrates strong auxiliary effects under moderate task similarity, their effectiveness may diminish if the trajectory of experience is very long, potentially introducing redundancy or interference in LLM planning. Future work can explore dynamic filtering mechanisms to adaptively retain the most relevant planning information from the long trajectory, further enhancing the robustness of the approach across diverse scenarios.

Acknowledgments

This work is supported by the National Key RD Program of China (No.2022ZD0116405).

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237, St. Julian's, Malta. Association for Computational Linguistics.

Dilip Arumugam and Thomas L. Griffiths. 2025. Toward efficient exploration by large language model agents. *arXiv preprint*. ArXiv:2504.20997.

Fengshuo Bai, Mingzhi Wang, Zhaowei Zhang, Boyuan Chen, Yinda Xu, Ying Wen, and Yaodong Yang. 2024a. Efficient model-agnostic alignment via bayesian persuasion. *arXiv preprint arXiv:2405.18718*.

Fengshuo Bai, Rui Zhao, Hongming Zhang, Sijia Cui, Ying Wen, Yaodong Yang, Bo Xu, and Lei Han. 2024b. Efficient preference-based reinforcement learning via aligned experience estimation. *arXiv preprint arXiv:2405.18688*.

Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. 2025. Reflective multi-agent collaboration based on large language models. *Advances in Neural Information Processing Systems*, 37:138595–138631.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. 2024. Automanual:

Constructing instruction manuals by LLM agents via interactive environmental learning. *Advances in Neural Information Processing Systems*, 37:589–631.

Sijia Cui, Aiyao He, Shuai Xu, Hongming Zhang, Yanna Wang, Qingyang Zhang, Yajing Wang, and Bo Xu. 2025. Self-guided function calling in large language models via stepwise experience recall. *arXiv preprint arXiv:2508.15214*.

Xiang Deng, Yu Gu, Boyuan Zheng, and 1 others. 2023. Mind2Web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114.

Hao Dong, Zihan Ding, and Shanghang Zhang. 2020. *Deep Reinforcement Learning: Fundamentals, Research and Applications*. Springer Nature.

Shuai Du, Jiahao Zhao, Jie Shi, and 1 others. 2025. A survey on the optimization of large language model-based agents. *arXiv preprint*. ArXiv:2503.12434.

Dayuan Fu, Biqing Qi, Yihuai Gao, Che Jiang, Guanting Dong, and Bowen Zhou. 2024a. MSI-agent: Incorporating multi-scale insight into embodied agents for superior planning and decision-making. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 643–659, Miami, Florida, USA. Association for Computational Linguistics.

Yao Fu, Dong-Ki Kim, Joonhee Kim, and 1 others. 2024b. Autoguide: Automated generation and selection of state-aware guidelines for large language model agents. *arXiv e-prints*. ArXiv:2403.08978.

Izzeddin Gur, Hiroki Furuta, Austin V. Huang, and 1 others. 2024. A real-world WebAgent with planning, long context understanding, and program synthesis. In *International Conference on Learning Representations (ICLR)*.

Xiang Huang, Sitao Cheng, Shanshan Huang, Jiayu Shen, Yong Xu, Chaoyun Zhang, and Yuzhong Qu. 2024. QueryAgent: A reliable and efficient reasoning framework with environmental feedback based self-correction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5014–5035, Bangkok, Thailand. Association for Computational Linguistics.

Xiao Huang, Junjie Lian, Yaochen Lei, and 1 others. 2023. Recommender ai agent: Integrating large language models for interactive recommendations. *arXiv preprint*. ArXiv:2308.16505.

Hao Jin, Lili Huang, Haotian Cai, and 1 others. 2024. From LLMs to LLM-based agents for software engineering: A survey of current, challenges and future. *arXiv preprint*. ArXiv:2408.02479.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Minsoo Kim, Jongyoon Kim, Jihyuk Kim, and Seungwon Hwang. 2024. QuBE: Question-based belief enhancement for agentic LLM reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21403–21423, Miami, Florida, USA. Association for Computational Linguistics.

Junkai Li, Yunghwei Lai, Weitao Li, Jingyi Ren, Meng Zhang, Xinhui Kang, Siyu Wang, Peng Li, Ya-Qin Zhang, Weizhi Ma, and 1 others. 2024a. Agent hospital: A simulacrum of hospital with evolvable medical agents. *arXiv preprint*. ArXiv:2405.02957.

Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. 2024b. Optimus-1: Hybrid multimodal memory empowered agents excel in long-horizon tasks. *Advances in Neural Information Processing Systems*, 37:49881–49913.

Xuechen Liang, Meiling Tao, Yinghui Xia, Tianyu Shi, Jun Wang, and Jingsong Yang. 2024. Self-evolving agents with reflective and memory-augmented abilities. *arXiv preprint*. ArXiv:2409.00872.

Jian Luo, Wei Zhang, Yuxuan Yuan, and 1 others. 2025. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint*. ArXiv:2503.21460.

Chen Qian, Yufan Dang, Jiahao Li, Wei Liu, Zihao Xie, YiFei Wang, Weize Chen, Cheng Yang, Xin Cong, Xiaoyin Che, Zhiyuan Liu, and Maosong Sun. 2024. Experiential co-learning of software-developing agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5628–5640, Bangkok, Thailand. Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Ashwin Gopinath, and 1 others. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, and 1 others. 2020. ALFWorld: Aligning text and embodied environments for interactive learning. *arXiv preprint*. ArXiv:2010.03768.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7584–7600, Bangkok, Thailand. Association for Computational Linguistics.

Chuanneng Sun, Songjun Huang, Haiqiao Liu, Jie Gong, and Dario Pompili. 2025. Retrieval-augmented hierarchical in-context reinforcement learning and hindsight modular reflections for task planning with llms. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1217–1224. IEEE.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2023. A survey on large language model based autonomous agents. *arXiv preprint*. ArXiv:2308.11432.

Lei Wang, Chen Ma, Xueyang Feng, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

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

Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. 2024b. Boosting LLM agents with recursive contemplation for effective deception handling. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9909–9953, Bangkok, Thailand. Association for Computational Linguistics.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2023. The rise and potential of large language model based agents: A survey. *arXiv preprint*. ArXiv:2309.07864.

Yuanchun Xiao, Jieyu Liu, Yizhen Zheng, and 1 others. 2024. CellAgent: An LLM-driven multi-agent framework for automated single-cell data analysis. *arXiv preprint*. ArXiv:2407.09811.

Shunyu Yao, Howard Chen, John Yang, and 1 others. 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. 2023a. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Wenhao Yao, Sebastian Heinecke, Juan Carlos Niebles, and 1 others. 2023b. Retroformer: Retrospective large language agents with policy gradient optimization. *arXiv preprint*. ArXiv:2308.02151.

Ming Yue. 2025. A survey of large language model agents for question answering. *arXiv preprint*. ArXiv:2503.19213.

Hongming Zhang, Fengshuo Bai, Chenjun Xiao, Chao Gao, Bo Xu, and Martin Müller. 2025a. β-dqn: Improving deep q-learning by evolving the behavior. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pages 2317–2326.

Hongming Zhang, Chenjun Xiao, Chao Gao, Han Wang, Martin Müller, and 1 others. 2024a. Exploiting the replay memory before exploring the environment: enhancing reinforcement learning through empirical mdp iteration. *Advances in Neural Information Processing Systems*, 37:85658–85692.

Hongming Zhang, Chenjun Xiao, Han Wang, Jun Jin, bo xu, and Martin Müller. 2023. Replay memory as an empirical MDP: Combining conservative estimation with experience replay. In *The Eleventh International Conference on Learning Representations*.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. 2024b. Agentpro: Learning to evolve via policy-level reflection and optimization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5348–5375, Bangkok, Thailand. Association for Computational Linguistics.

Yichi Zhang, Shuai Qiao, Junyi Zhang, and 1 others. 2025b. A survey of large language model empowered agents for recommendation and search: Towards next-generation information retrieval. *arXiv preprint*. ArXiv:2503.05659.

Yichi Zhang, Ge Xiong, Honghao Li, and 1 others. 2025c. EDGE: Efficient data selection for LLM agents via guideline effectiveness. *arXiv preprint*. ArXiv:2502.12494.

Zhaowei Zhang, Fengshuo Bai, Qizhi Chen, Chengdong Ma, Mingzhi Wang, Haoran Sun, Zilong Zheng, and Yaodong Yang. Amulet: Realignment during test time for personalized preference adaptation of llms. In *The Thirteenth International Conference on Learning Representations*.

Aojun Zhao, Dahu Shi, Quan Xu, and 1 others. 2024a. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642.

Qi Zhao, Haotian Fu, Chen Sun, and George Konidaris. 2024b. EPO: Hierarchical LLM agents with environment preference optimization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6401–6415, Miami, Florida, USA. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, and 1 others. 2023. A survey of large language models. *arXiv preprint*, 1(2). ArXiv:2303.18223.

Jie Zheng, Chenhui Shi, Xiaodong Cai, and 1 others. 2025. Lifelong learning of large language model based agents: A roadmap. *arXiv preprint*. ArXiv:2501.07278.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Wangchunshu Zhou, Yixin Ou, Shengwei Ding, Long Li, Jialong Wu, Tiannan Wang, Jiamin Chen, Shuai Wang, Xiaohua Xu, Ningyu Zhang, and 1 others. 2024. Symbolic learning enables self-evolving agents. *arXiv preprint*. ArXiv:2406.18532.

Appendix

A Environment Details

A.1 AlfWorld (Shridhar et al., 2020)

AlfWorld is a simulated environment that bridges the gap between language-based reasoning and embodied action by aligning tasks in a text-based interactive fiction environment with corresponding tasks in a visually rich, embodied 3D simulator. This unique structure allows for the study of agents capable of both high-level abstract planning and low-level grounded execution.

AlfWorld encompasses a diverse set of household tasks, requiring agents to perform a sequence of actions to achieve a specified goal. These tasks are categorized into six main types:

- Pick & Place (e.g., "move a plate to the coffee table") - the agent must find an object of the desired type, pick it up, find the correct location to place it, and move it there.
- Examine in Light (e.g., "examine a book under the lamp") - the agent must find an object of the desired type, locate and turn on a light source with the desired object in-hand.
- Clean & Place (e.g., "clean the knife and move it to the drawer") - the agent must find an object of the desired type, pick it up, go to a sink or a basin, wash the object by turning on the faucet, then find the correct location to place it, and move it there.
- Heat & Place (e.g., "heat a mug and move it to the coffee table") - the agent must find an object of the desired type, pick it up, go to a microwave, heat the object turning on the microwave, then find the correct location to place it, and move it there.
- Cool & Place (e.g., "move a cool bottle to the countertop") - the agent must find an object of the desired type, pick it up, go to a fridge, put the object inside the fridge and cool it, then find the correct location to place it, and move it there.
- Pick Two & Place (e.g., "move two pencils to the drawer") - the agent must find an object of the desired type, pick it up, find the correct location to place it, move it there, then look for another object of the desired type, pick it up, return to previous location, and move it there with the other object.

A.2 WebShop (Yao et al., 2022)

WebShop is an interactive environment that simulates a real-world e-commerce website, designed as a benchmark for evaluating interactive planning and grounded language understanding in agents. The environment features a substantial catalog of 1.18 million real-world products, each with various attributes and options. Agents are tasked with completing shopping missions based on 12,087 diverse, crowd-sourced natural language instructions. The WebShop benchmark also includes over 1,600 human demonstration trajectories, which can be utilized for validating the task and supporting imitation learning.

The core task for an agent in WebShop is to navigate the simulated website, search for products, identify items that match the criteria given in the instruction, potentially customize options (e.g., color, size), and finally purchase the correct item. The agent interacts with the environment by issuing a variety of actions, such as typing search queries, clicking on product links or buttons, selecting product options, and executing a purchase command. Success is measured by whether the purchased item exactly matches the instruction's requirements, yielding a binary success rate (SR).

A task episode concludes when the agent issues a purchase action or a maximum step limit is reached. The agent receives a reward based on the degree to which the purchased item matches the specified attributes and options in the initial instruction, including price constraints. The reward function is designed to reflect the quality of the purchased item against the user's requirements. Specifically, the reward r is calculated as:

$$r = \frac{|U_{\text{att}} \cap Y_{\text{att}}| + |U_{\text{opt}} \cap Y_{\text{opt}}| + \mathbb{I}[y_{\text{price}} \leq u_{\text{price}}]}{|U_{\text{att}}| + |U_{\text{opt}}| + 1} \cdot r_{\text{type}} \tag{1}$$

where U_{att} and Y_{att} are the sets of attribute names from the user query and the purchased product, respectively; U_{opt} and Y_{opt} are the sets of optional attribute values; $\mathbb{I}[y_{\text{price}} \leq u_{\text{price}}]$ is an indicator function which is 1 if the purchased product's price is less than or equal to the user's preferred price, and 0 otherwise. r_{type} is a term related to textual matching between the selected product's title and the target product's title, defined as:

Retrieval Parameters	
Vectorstore	Faiss
Retriever Type	kNN
Embedder	all-mpnet-base-v2
Experience Retrieval Scope	2
Top-k Experience's Tips Selection	2
Agent Parameters	
Max Reflection Retries	3
Focus Points Generation Model	gpt-4-turbo-2024-04-09
Policy Model	gpt-4-turbo-2024-04-09
Reflection Model	gpt-4-turbo-2024-04-09
Tips Generation Model	gpt-4o-2024-08-06
Key Information Extraction Model	gpt-4-turbo-2024-04-09
Key Information Reflection Model	gpt-4-turbo-2024-04-09
Max Tips from Compare Traj.	5
Max Tips from Success Traj.	3
AlfWorld-specific Parameters	
Max Number of Environment Steps	20
Number of Experiment Tasks	134
WebShop-specific Parameters	
Max Number of Environment Steps	15
Number of Experiment Tasks	134
ScienceWorld-specific Parameters	
Max Number of Environment Steps	80
Number of Experiment Tasks	100
WebArena-Shopping-specific Parameters	
Max Number of Environment Steps	20
Number of Experiment Tasks	90
Tips Alignment Model	gpt-4-turbo-2024-04-09

Table 7: Retrieval, Agent and Environment Parameters.

$$r_{\text{type}} = \begin{cases} 0, & \text{if TextMatch} = 0 \\ 0.1, & \text{if TextMatch} < 0.1 \\ 0.5, & \text{if TextMatch} \leq 0.2 \\ 1, & \text{otherwise.} \end{cases} \quad (2)$$

The term "TextMatch" denotes the textual overlap of pronouns, nouns, and proper nouns between the selected product's title and the target product's title.

A.3 ScienceWorld (Wang et al., 2022)

ScienceWorld is a text-based virtual environment designed to evaluate an agent's scientific reasoning and procedural task completion abilities, benchmarked at the level of a standard elementary school science curriculum.

ScienceWorld features a rich, dynamic, and interactive simulated world. This includes multiple interconnected locations (e.g., kitchen, workshop, laboratory, greenhouse), populated with over 200 common household and science-related objects. These objects possess various properties (e.g., temperature, conductivity, state of matter) and afford a wide range of actions. The underlying simulation engines model simplified thermodynamics, electrical circuits, chemical reactions, and biological processes, allowing for complex interactions and state changes within the environment.

The benchmark comprises 30 diverse task types, spanning 10 distinct topics from the elementary science curriculum. Examples include tasks related to changing states of matter (e.g., melting ice, boiling water), understanding life cycles (e.g., growing a plant), exploring basic physics (e.g., building simple electrical circuits), and conducting simple chemistry experiments. For each task type, multiple variations are procedurally generated, altering object properties, initial locations, and specific sub-goals to test for generalization. Agents interact with the environment by issuing text commands from a predefined set of action templates (e.g., "open door", "take apple from table", "use thermometer on beaker"), and receive textual observations describing the current state of their surroundings and the outcomes of their actions. The average optimal decision depth for these tasks can be significant, often requiring a long sequence of correct actions to achieve the goal.

A.4 WebArena-Shopping (Zhou et al., 2023)

WebArena is a suite of realistic and reproducible web environments designed for the development and evaluation of autonomous agents. The platform features fully operational, self-hostable web applications that emulate real-world websites across various domains.The WebArena–Shopping environment is implemented as part of the WebArena suite, providing a realistic and reproducible e-commerce setting for interactive planning agents.

B More Implementation Details

To fairly compare with ExpeL, we employ four-fold validation for all experiments. We train on one half of the dataset and evaluate on the other half, and vice versa. All results include the mean and standard error of the results across the folds. The parameter information for the experiments conducted in each environment is shown in Table 7.

We adopted different experimental task selection strategies for different environments. In the case of ALFWorld (Shridhar et al., 2020), we utilized the 134 solvable tasks that ReAct and ExpeL used, which comprised of 6 different task types in a virtual household environment: Pick (24), Clean (18), Heat (31), Cool (23), Examine (21), and Pick two (17). Similarly, for WebShop (Yao et al., 2022), we evaluated using the same 100 tasks used by ReAct and ExpeL. For ScienceWorld (Wang et al., 2022), we assess performance using 100 tasks sampled from test set, which comprised of diverse task types with environmental steps up to 80: task-10-use-thermometer (10), task-2-power-component (5), task-2a-test-conductivity(10), task-3-find-animal (10), task-3-find-living-thing (10), task-3-find-non-living-thing (10), task-3-find-plant (10), task-4-grow (8), task-5-chemistry (11), task-6-lifespan (3), task-7-identify-life-stage (9), task-1-change-the-state-of-matter (4).

C Prompt Templates

C.1 Focus Points Generation Model

We present our prompt template of focus points generation model in the ALFWorld, WebShop and ScienceWorld in Figure 4.

C.2 Experience-Wise Tips Generation Model

Figure 5, 6 and 7 detail our prompt template of experience-wise tips generation model in the ALFWorld, WebShop and ScienceWorld.

C.3 Key Information Extraction Model

Figure 8 shows our prompt template for key information extraction to perform a structured, fine-

grained analysis of the current trajectory content.

C.4 Key Information Reflection Model

When specific abnormal observations happen during evaluation, we use the prompt with Figure 9 for key information reflection.

D Example Trajectories

D.1 AlfWorld, CFGM & ReAct

Figure 10 and Figure 11 show the cases of tips and key information reflection in CFGM compared to ReAct, respectively, to help the agent avoid errors or recover from derailed planning in AlfWorld.

D.2 WebShop, CFGM & ReAct

Figure 12 and Figure 13 show the cases of tips and key information reflection in CFGM compared to ReAct, respectively, to help the agent avoid errors or recover from derailed planning in WebShop.

D.3 ScienceWorld, CFGM & ReAct

Figure 14 and Figure 15 show the tips and key information reflection in CFGM, respectively, to help the agent avoid errors or recover from derailed planning in ScienceWorld.

Focus Points Generation Model Prompt

You are a Task Learning Guidance Agent whose mission is to analyze a given task and its examples in detail. Your objective is to derive a comprehensive list of actionable and specific focus points that will be critical for learning and executing the task. When producing your analysis, please adhere to the following requirements:
- Your points must be concrete and directly linked to the task's environment and execution process. Avoid abstract, generic, or vague statements. The maximum number of these points is 10.
- Each focus point should provide clear guidance and actionable insights that assist the development of further learning and experience-based summaries.
- The analysis should enable thorough, multi-faceted reasoning. It must help identify key challenges, actionable steps, and potential pitfalls the agent may encounter during task execution.

For each focus point you extract, follow the structure below:
First, explain your reasoning process that led you to identify this focus point. This reasoning should be based on a detailed analysis of the provided task environment, execution steps, constraints, and relevant specifics. Ensure that your reasoning is clear and directly connected with the task's context.
Next, present the concrete focus point as a moderately detailed sentence that provides clear guidance for learning and implementation. The focus point must be specific and closely linked to the task, but do not use 'such as' or 'e.g.' to explain it.
Finally, Ensure that each pair of reasoning and focus point is numbered sequentially. For example:
Reasoning 1: [Detailed explanation...]
Focus Point 1: [Specific guidance sentence ...]
Repeat this structure for each distinct focus point needed to support comprehensive task execution.

Input the necessary task details below:
Task Environment Description:
{task_description}

Task Examples:
{few_shots}

Figure 4: The prompt template of focus points generation model in the ALFWorld, WebShop and ScienceWorld.

Experience-Wise Tips Generation Model Prompt

Tips Generation from Compare Trajectories Template

By examining the successful and failed trials of the same task, you need to think about the difference and the causes that make one trajectory failed and the other successful. The failed trials may be multiple. Based on your thinking process and the above suggestions about the tips, generate tips to avoid the same mistakes and help the successful completion in the similar task. Strictly follow what the SUCCESSFUL TRIAL does to summerize tips. When the generated Tip involves actions, you must use the action syntax from the SUCCESSFUL TRIAL to describe the corresponding action for the correctness.
The tips you come up with don't need to be limited to the above suggestions, but you need to ensure that there are no contradictions or duplications between them.
You should make your tips specific and concise, of which number is should not exceed 5, and the quality of the tips is more important than the quantity, so there is no need to force 5 tips.
You should put them in this format:
Thinking n: ...
Tip n: ...
where n indicate the index of thinking and related tip.

Here are the previous success and failed trials of the same task to compare and critique:

SUCCESSFUL TRIAL:

{success_history}

(END OF SUCCESSFUL TRIAL)

FAILED TRIALS:

{fail_history}

(END OF FAILED TRIALS)

Now, based on the given SUCCESSFUL TRIAL and FAILED TRIALS about the same TRIAL TASK, provide your thinkings and tips accordingly.

Figure 5: The prompt template of tips generation from compare trajectories.

 # Extra Tips Generation from Single Successful Trajectory Template

By examining a successful trial and some existing tips of a task, you need to think about the factors that lead to success in addition to existing tips in this successful trajectory. Based on your thinking process and the above suggestions about the tips, generate new tips to help the successful completion in the similar task. Strictly follow what the SUCCESSFUL TRIAL does to summerize tips. When the generated Tip involves actions, you must use the action syntax from the SUCCESSFUL TRIAL to describe the corresponding action for the correctness.
The tips you come up with don't need to be limited to the above suggestions, but you need to ensure that there are no contradictions or duplications between them.
The content of your proposed tips MUST NOT be similar to the existing tips, and you must ensure that the tips you propose are innovative.
You should make your new tips specific and concise, of which number is should not exceed 3, and the quality of the new tips is more important than the quantity, so there is no need to force 3 tips.
You should put them in this format:
Thinking n: ...
Tip n: ...
where n indicate the index of thinking and related tip.

Here is the SUCCESSFUL TRIAL:

{success_history}

(END OF SUCCESSFUL TRIAL)

Here is the EXISTING TIPS:

{existing_tips}

(END OF EXISTING TIPS)

Now, based on the given SUCCESSFUL TRIAL and EXISTING TIPS, provide your thinkings and tips accordingly.

Figure 6: The prompt template of extra tips generation from single success trajectory. The extraction is performed after generating tips from the compare trajectories as a supplement to get the factors leading to task success.

 # Tips Generation from Single Successful Trajectory Template

By examining a successful trial of a task, you need to think about the factors that lead to success in this successful trajectory. Based on your thinking process and the above suggestions about the tips, generate tips to help the successful completion in the similar task. Strictly follow what the SUCCESSFUL TRIAL does to summerize tips. When the generated Tip involves actions, you must use the action syntax from the SUCCESSFUL TRIAL to describe the corresponding action for the correctness.
The tips you come up with don't need to be limited to the above suggestions, but you need to ensure that there are no contradictions or duplications between all tips.
You should make your tips specific and concise, of which number is should not exceed 3, and the quality of the tips is more important than the quantity, so there is no need to force 3 tips.
You should put them in this format:
Thinking n: ...
Tip n: ...
where n indicate the index of thinking and related tip.

Here is the SUCCESSFUL TRIAL:

{success_history}

(END OF SUCCESSFUL TRIAL)

Now, based on the given SUCCESSFUL TRIAL, provide your thinkings and tips accordingly.

Figure 7: The prompt template of tips generation from single success trajectory. The extraction targets the tasks which are successfully completed at the first try.For these tasks, there is only single success trajectory.

Key Information Extraction Model Prompt

Now you need to play the role of an trajectory information extractor. You are provided with the current trajectory information. The current trajectory information have errors occurred in the latest process.
Analyze current trajectory to extract its **trajectory information** that capture essential patterns and knowledge for guiding agent reflection.
Output Format:
- Task Objective: [The primary goal or purpose of the task.]
- Initial Environment: [Initial information about the Task Environment without losing any detailed information.]
- Task Progress: [A summary of the current state of the task, including a natural and coherent description of actions and observations.]

Your output must capture task-relevant concepts about current trajectory information. Be clear, concise, and analytical—your goal is to abstract knowledge that supports improvement for the current trajectory. The **trajectory information** will be used to guide the agent's self-reflection and reasoning enhancement.

The current trajectory information is as follow:

{current_history}

(END OF CURRENT TRAJECTORY)

Now, based on the above information, please provide the trajectory information of current trajectory in the specified format:

Figure 8: The prompt template of key information extraction model in the ALFWorld, WebShop and ScienceWorld.

Key Information Reflection Model Prompt

Now you need to play the role of an self-reflector. You will be provided with: current trajectory, the extracted key information about current trajectory and some successful examples for reference.
Based on some successful examples and key information about current trajectory, Your task is to analyze and reflect on the current trajectory through self-questioning and answering, then reason about the plan that can help current trajectory to success.
You need to consider the current status of the current trajectory (current planning, task progress and action format), and then reflect on and correct the current trajectory without retrying similar errors.
If the successful trajectory corresponds to a task similar to the current one, you can examine the overlapping segment between the successful trajectory and the current trajectory and mimic the planning adopted by the successful trajectory beyond that overlap. It is important to avoid repeatedly trying planning approaches that do not modify the current trajectory.
It is important to avoid repeatedly trying planning approaches that do not modify the current trajectory.
Here are some successful examples for reference:

{success_history}

(END OF EXAMPLES)

Here is the current trajectory to reflect:

{current_history}

(END OF CURRENT HISTORY)

Here is the key information extracted from the current trajectory, which you can use to reckeck the current trajectory information:

{key_info}

(END OF KEY INFO)

You must output your reasoning strictly in the following format to ensure consistency and ease of parsing:
> Self-QA: [Ask some critical questions about current planning, task progress and action format and answer them to reflect. The questions and answers should be concise.]
> Think: [Based on the content of Self-QA, infer the next step that needs to be done for the current trajectory to succeed. The content must be concise and clear without explanations]

Now, based on the above information, please provide your Self-QA and Think in the specified format:

Figure 9: The prompt template of key information reflection model in the ALFWorld, WebShop and ScienceWorld.

You are in the middle of a room. Looking quickly around you, you see a bed 1, a desk 2, a desk 1, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a laundryhamper 1, a safe 1, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, and a shelf 1.
Your task is to: examine the bowl with the desklamp.

ReAct

Your task is to: examine the bowl with the desklamp.
> think: To solve the task, I need to find and take a bowl, then find and use a desklamp. First, I need to find a bowl. A bowl is more likely to appear in drawer (1-6), desk (1-2), shelf (1-6), garbagecan (1). I can check one by one, starting with drawer 1.
> think: OK.
> go to drawer 1
.....
> go to drawer 3
Nothing happens.
> think: You arrive at drawer 3. The drawer 3 is closed.
> open drawer 3
Nothing happens.
> go to drawer 3
Nothing happens.

..... **(Repeated Action 'go to drawer 3')**

Nothing happens. (Run out of maxsteps!)

Potential Influential Tips

Tip 4: Prioritize checking desks first when searching for items like bowls, as they are more likely to be found there compared to drawers or shelves. ***(Fine-Grained)***

CFGM

> think: To solve the task, I need to find and take a bowl, then find and use a desklamp to examine it. I will start by checking the desks, as they are common places for such items.
> go to desk 1
You arrive at desk 1. On the desk 1, you see a alarmclock 2, a alarmclock 1, a desklamp 1, a mug 1, and a pencil 1.
> go to desk 2
You arrive at desk 2. On the desk 2, you see a bowl 3, a bowl 2, and a mug 2.
> take bowl 2 from desk 2
You pick up the bowl 2 from the desk 2.
> go to desk 1
You arrive at desk 1. On the desk 1, you see a alarmclock 2, a alarmclock 1, a desklamp 1, a mug 1, and a pencil 1.
> use desklamp 1

Task is SOLVED.

Figure 10: Example Trajectories in the ALFWorld. The tip says, 'Prioritize checking the desk,' so the CFGM can quickly find the bowl instead of obsessing over drawer 3 and overlooking the more accessible solution on the desk—unlike ReAct, which ignored this efficient path.

You are in the middle of a room. Looking quickly around you, you see a bed 1, a desk 2, a desk 1, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a laundryhamper 1, a safe 1, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, and a shelf 1.
Your task is to: move two cd to safe.

ReAct

> think: To solve the task, I need to find and take the first cd, then move it to the safe, then find and take the second cd, then move it to the safe.
> think: OK.
> think: First I need to find the first cd. A cd is more likely to appear in drawer (1-6), desk (1-2), or shelf (1-6). I can check one by one, starting with drawer 1.
You are thinking too many times without taking action.
> go to drawer 1
.....
> go to drawer 3
Nothing happens.
> go to drawer 3
Nothing happens..]

..... **(Repeated Action'go to drawer 3')**

Ran out of steps! TASK FAILED

CFGM

> think: To solve the task, I need to find and take the first cd, then move it to the safe, then find and take the second cd, then move it to the safe.
> think: OK. I will start by checking the drawers for cds, then move to desks, and finally check shelves if necessary.
> go to drawer 1
.....
> go to drawer 3
Nothing happens.

Potential Influential Key Info & Self-QA

> Key Info:... Task Progress:... The task has not yet been successful as the CDs have not been located...
>Self-QA: ....Have I checked all possible locations where CDs might be found based on the successful examples?I have not yet checked all drawers, desks, or shelves where CDs might be located.

> think: I should continue checking the remaining drawers, starting with drawer 3, and then proceed to desks and shelves if necessary to find the two CDs.
> go to drawer 4
.....
> take cd 2 from desk 2
.....
> move cd 1 to safe 1
Task is SOLVED.

Figure 11: Example Trajectories in the ALFWorld. When interacting with drawer 3 and the environment returns "Nothing happens", the CFGM realizes that it has not checked other locations by KIR. Instead of repeating the operation like ReAct, CFGM skips drawer 3 and continues to check other drawers.

Instruction: i'm looking for hair treatments that are sulfate and paraben free and are of high quality too. i need it in bottle for with 60 capsules, and price lower than 50.00 dollars

ReAct

Action: search[hair treatment sulfate free paraben free 60 capsules]
Observation:
i'm looking for hair treatments that are sulfate and paraben free and are of high quality too. i need it in bottle for with 60 capsules, and price lower than 50.00 dollars
[Back to Search] .....
Action: think[None of the listed products mention being in a bottle form with 60 capsules. I should check the next page to see if there are any relevant products.]
Observation: OK.
Action: click[Next >]
.....
Action: think[The search results still do not show any hair treatment products in a bottle form with 60 capsules. I should check the next page to see if there are any relevant products.]

..... **(Multiple ineffective click attempts and failed to successfully find.)**

Ran out of steps! TASK FAILED

Potential Influential Tips

Tip 4: Use a detailed search query that includes specific attributes of the product you are looking for.... ***(Coarse-Grained)***
Tip 5: Analyze the search results to identify products that match your criteria, such as price and description, and select one to investigate further by clicking on it. ***(Fine-Grained)***

CFGM

Action: search[hair treatments sulfate paraben free high quality 60 capsules bottle]
Observation:
i'm looking for hair treatments that are sulfate and paraben free and are of high quality too. i need it in bottle for with 60 capsules, and price lower than 50.00 dollars
[Back to Search]
Page 1 (Total results: 50)
.....
[B09HWS67T8] (Although the product does not come in the form of "60 capsules," in our subsequent operations, you can select the "1 bottle + 60 capsules" option and place an order.)
.....
Action: click[1 bottle + 60 capsules]
Observation: You have clicked 1 bottle + 60 capsules.
Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 1.0
Task is SOLVED.

Figure 12: Example Trajectories in the WebShop. The tips suggest using a detailed search query and investigating products by clicking on them, ensuring that CFGM focuses more precisely on target-matching products rather than relying on fuzzy searches like ReAct. Once a suitable product is found, CFGM proceeds to click on it and selects the '1 bottle, 60 capsules' option.

Instruction: i need a futon and chaise set that is made with faux leather. and i would prefer the navy linen color, and price lower than 130.00 dollars

ReAct

Action: search[futon chaise set faux leather]
Observation:
i need a futon and chaise set that is made with faux leather. and i would prefer the navy linen color, and price lower than 140.00 dollars
[Back to Search] .....
Action: think[None of the options on the fifth page match the criteria of being under $130.00 and navy linen color. I should check the next page.]
Action: click[Next >]
Observation: Invalid action!
..... **(Multiple ineffective click attempts and failed to successfully find.)**

Ran out of steps! TASK FAILED

CFGM

Action: search[futon and chaise set faux leather navy linen]
Observation:
i need a futon and chaise set that is made with faux leather. and i would prefer the navy linen color, and price lower than 130.00 dollars
.....
Action: click[Next >]
Observation: Invalid action!

Potential Influential Key Info & Self-QA

> Key Info: ...Task Progress:...The closest matches either did not meet the color specification or were priced above the desired range....resulted in an "Invalid action!" , indicating a possible issue with the search functionality or user input error at that stage.
>Self-QA: ...Q3:What should be the next step after realizing there are no matching products on the first page? A3: The next step should be to refine the search criteria or adjust expectations, such as broadening the search terms or including alternative color options.

Action: think[[To succeed in the current trajectory, refine the search query to either broaden the search terms or include alternative color options. Additionally, consider checking each product's details by clicking on them to see if any meet the criteria even if the color isn't exactly as specified but close to navy linen.]]
Action: search[futon and chaise set faux leather]
.....
Action: click[navy linen]
Observation: You have clicked navy linen.
Action: click[futon and chaise set]
Observation: You have clicked futon and chaise set.
Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 1.0
Task is SOLVED.

Figure 13: Example Trajectories in the WebShop. Through KIR, CFGM recognized that the closest match in the current task progress failed to locate products with the desired color, prompting it to refine the search query by broadening the search terms. This approach avoided unnecessary multiple searches and page navigation—unlike ReAct, which exhausted its steps and ultimately failed.

Task Description: Your task is to find a(n) plant. First, focus on the thing. Then, move it to the blue box in the bedroom.

ReAct

Thought: To find a plant, I should start by looking in the greenhouse, as that is the most likely place to find plants.
Action: teleport to greenhouse
.....
Thought: I found the blue box in the bedroom. Now I need to move the peach tree into this box.
Action: move peach tree in flower pot 4 to blue box
Observation: No known action matches that input.
.....
Thought: The flower pot contains a peach tree and soil, but there's no indication that it's designed to be easily movable. I need to find a way to transport the peach tree to the bedroom.
Action: pick up flower pot 4 containing peach tree
Observation: No known action matches that input.
Thought: I am unable to pick up the flower pot directly. I should look for a tool or device in the greenhouse that could help me move the flower pot.

..... **(Attempted to move the flowerpot but couldn't find the necessary tools)**

Task Truncated. Your score (min -1.0, max 1.0): -1.0.

Potential Influential Tips

Tip 2: If you cannot pick up the plant directly, pick up the entire flower pot containing the plant to move it to the desired location. ***(Fine-Grained)***
Tip 3: Once you have the flower pot in your inventory, teleport to the target location and move the flower pot to the specified container to successfully complete the task. ***(Fine-Grained)***

CFGM

Thought: To find a plant, the greenhouse is the most likely place where plants are kept.
Action: teleport to greenhouse
Observation: You teleport to the greenhouse.
.....
Thought: I see several plants in the greenhouse. I will focus on one of the plants to complete the first part of the task.
Action: focus on apple tree in flower pot 5
Observation: You focus on the apple tree.
.....
Thought: Since I cannot pick up the apple tree directly, I will try moving the entire flower pot containing the apple tree to the blue box in the bedroom.
Action: pick up flower pot 5
.....
Observation: You move the flower pot 5 to the blue box. Task Success. Your score (min -1.0, max 1.0): 1.0.
Task is SOLVED.

Figure 14: Example Trajectories in the ScienceWorld. When CFGM realizes that it can't pick up the plant directly, tips reminds that it can move the whole pot and teleport to the target location immediately after moving it, avoiding the pointless search for a carrying tool in ReAct.

Task Description: Your task is to determine if unknown substance U is electrically conductive. The unknown substance U is located around the workshop. First, focus on the unknown substance U. If it is electrically conductive, place it in the yellow box. If it is electrically nonconductive, place it in the purple box.

ReAct

Thought: To determine if the unknown substance U is electrically conductive, I need to find it in the workshop and then test its conductivity.
Action: teleport to workshop
.....
Action: connect red wire to battery's positive terminal
Observation: No known action matches that input.
.....
Action: connect battery's positive terminal to red wire's one end
Observation: No known action matches that input.
..... **(Unable to correctly use the "connect" action format, repeated attempts with invalid commands)**
Observation: Connections must specify specific points of contact on polarized electrical components (e.g. anode, cathode) **Task Truncated. Your score (min -1.0, max 1.0): -1.0.**

CFGM

Action: teleport to workshop
Observation: You teleport to the workshop.
Action: look around
.....
Action: connect black wire to battery
Observation: Connections must specify specific points of contact on unpolarized electrical components (e.g. terminal 1, terminal 2)
Action: connect black wire terminal 1 to battery terminal 1
Observation: No known action matches that input.

Potential Influential Key Info & Self-QA

> Key Info: ...Task Progress: ...The agent attempted to specify a terminal but faced an input mismatch error, indicating a possible misunderstanding or incorrect command format for making connections...
>Self-QA: Q1:Why did the action to...fail? A1: The action failed because the command format was incorrect. The system requires specifying connections with exact terms, such as "terminal 1 on black wire to anode on battery."

Thought: Examine the battery and black wire to identify their specific connection points. Then, attempt to connect them using the correct terminology based on the examination results...
.....
Action: connect black wire terminal 1 to battery anode
Observation: terminal 1 on black wire is now connected to anode on battery
.....
Observation: You move the unknown substance U to the purple box.
Task Success. Your score (min -1.0, max 1.0): 1.0.

Figure 15: Example Trajectories in the ScienceWorld. CFGM finds out that it is currently facing a mismatch error between the battery and the wire through the KIR which prompt a possible exact term to connect. Then CFGM successfully finds the correct way to connect by checking the connection points and avoids the invalid operation due to the formatting error in the ReAct.