CRaFT: Causal Reasoning Augmented Reflection for Long-Horizon Robotic Planning

***Abstract*— Recent developments suggest that Large Language Models (LLMs) can identify and correct errors in their generated responses using reflection mechanisms. However, when applied to long-horizon task planning, these methods reveal significant limitations. Reflection methods may neglect the causes of errors in earlier planning, producing results that contain inaccurate information, thereby leading to further mistakes in subsequent planning. This paper explores the reflection framework suitable for long-horizon task planning. Inspired by human causal cognitive processes, we introduce the Causal Reasoning augmented reFlecTion framework (CRaFT). CRaFT employs systematic causal reasoning to accurately identify the root causes of errors and to generate effective action plan revisions by integrating association information. We conducted household task experiments in ALFWorld and VirtualHome, and the results show that CRaFT significantly improves success rates by 34% and 27%, respectively, on complex long-horizon tasks. Furthermore, we applied CRaFT to a real-world robotic system, highlighting its potential for practical applications. Website at https://causalraft.github.io/**

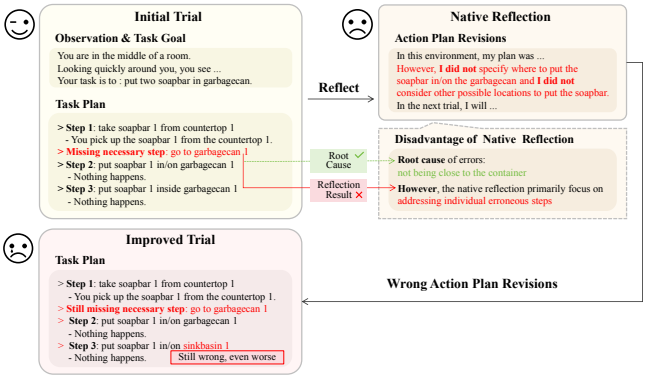

Fig. 1. Illustrates the limitation in using reflection methods for long-horizon task planning. These methods merely identify and correct one or a few erroneous steps, overlooking the root causes of errors. However, correcting only these steps may lead to greater errors in subsequent trials.

I. INTRODUCTION

Task planning is a critical process widely applied in fields such as robotics navigation [1], manipulation [2], [3], and everyday household tasks [4]–[8]. While research on short-term decision-making has progressed significantly, the complexity and error accumulation inherent in long-horizon task planning continue to present substantial challenges. Long-horizon task planning requires the management of extended action sequences and more complex environmental states. Additionally, in long-horizon planning, the issue of error accumulation becomes more evident, as initial mistakes may cause progressively greater deviations over time.

Recently, large language models (LLMs) have been incorporated into task planning due to their strong commonsense reasoning and logical capabilities, as seen in methods like ReAct [9] and ADAPT [10]. However, these LLM-based methods can face challenges like hallucinations and contextually unfaithful responses [11]–[13]. To address these issues, recent studies have introduced reflection mechanisms into task planning, employing either human or automatic feedback to refine erroneous outputs. This transition from an open-loop to a closed-loop system is exemplified by methods like RETROFORMER [14] and Reflexion [15]. Therefore, we pose the question: Can the LLM-based reflection framework be applied to long-horizon task planning?

Although the above methods have achieved some initial success, a key challenge in applying reflection methods to long-horizon task planning lies in their inability to identify the root causes of errors (RCE), which is essential for the subsequent generation of action plan revisions. Presently, these reflection methods do not concentrate on understanding the reasons behind errors in long-horizon task planning. Instead, they focus solely on revising the action plan by addressing only part of the erroneous steps [14]–[16]. For example, in Figure 1, reflection methods focus on testing different placement strategies for a soapbar instead of investigating why the failure occurred in the first place. This neglect of identifying the RCE leads to subsequent trials that are even more erroneous. In some cases, it can even result in dangerous actions that compromise the robot's reliability.

Motivated by the above challenges, this paper investigates the analysis of RCE in complex long-horizon task planning. Such planning typically involves extended action sequences and accumulates errors over time. Therefore, simply identifying and correcting one or a few erroneous steps does not fully rectify the errors in the planning. Instead, we frame the problem as identifying a minimal subset of actions whose repair simultaneously ensures causal coverage of all downstream errors and guarantees that the resulting trajectory satisfies the task goal. Only by accurately identifying the RCE can one generate action plan revisions that directly address them, thereby resolving accumulating errors and enabling successful task completion in subsequent trials. However, current reflection methods predominantly focus on addressing individual erroneous steps to generate action plan revisions. In contrast, human intelligence is characterized by the ability to perform causal reasoning across three levels: Counterfactuals, Association, and Intervention [17]–[19]. These levels range from simple observations to complex imaginations, enabling humans to systematically analyze

Manuscript 1293 submitted to 2026 IEEE International Conference on Robotics and Automation (ICRA). Received September 15, 2025.

the RCE and propose effective action plan revisions, thus enhancing the success rate of subsequent trials.

Inspired by the process outlined above, we propose CRaFT, a general reflection framework that harnesses the causal reasoning capabilities of LLMs to analyze the RCE and generate effective action plan revisions for future trials. First, we employ LLMs to infer associated information from environmental observations, corresponding to the first level of causal reasoning. Next, based on the task trajectory, associated information, and the RCE in the previous action plan, we formulate action plan revisions aimed at addressing the root causes and successfully complete the next trial, corresponding to the second level of causal reasoning. Finally, for the failed action plan, we utilize counterfactual reasoning to identify RCE by performing dependency-aware backtracking over unmet preconditions. Each candidate failure is validated through LLM-based inference under the assumption that the corresponding action succeeded—aligning with third-level causal reasoning. To further enhance the reflection optimization process, we integrate memory management to improve the LLM's performance in subsequent trials.

To validate the effectiveness of our framework, we first evaluate it in the ALFWorld environment across three task types, where it achieves a 34% higher success rate than the Planning-only baseline. We further test CRaFT in the VirtualHome environment, showing a 27% improvement on the NovelTask dataset. Finally, a CRaFT-based planning system evaluated on real-world long-horizon rearrangement tasks demonstrates its practical applicability. In summary, the contributions of this paper are as follows:

1) To our knowledge, this is the first attempt to study RCE analysis in the context of long-horizon task planning.
2) We have introduced a framework named CRaFT. Our framework effectively enhances the application of reflection frameworks in long-horizon task planning by identifying the RCE, obtaining association information, and then generating action plan revisions.
3) We conducted experiments in the virtual household environments ALFWorld and VirtualHome, as well as in real-world tasks. The results demonstrate that our framework could effectively enhances the success rate of complex long-horizon task planning.

II. Related works

A. LLMs for Task Planning

In recent years, the general performance exhibited by LLMs has sparked interest among researchers in exploring the potential of LLMs for task planning in open-ended environments. For instance, LID [20] uses GPT-2 as a policy network, predicting the next action based on encoded tasks information. ReAct [9] introduces a new prompting paradigm that combines reasoning and action to enhance the ability of LLMs to solve general tasks. CLIN [21] features persistent dynamic text memory centered on causal abstraction, enabling LLM-based agents to continually learn and adapt in new environments. RAP [22] combines a retrieval-based mechanism with a generative model, boosting multimodal LLM agents' planning capabilities. SYNAPSE [23] employs a trajectory-based prompting mechanism that selects and uses similar past trajectories from memory.

B. LLMs for Reflection

To address the challenges of hallucinations or context unfaithfulness in outputs from LLMs, researchers have investigated the application of reflection mechanisms to improve the quality of outputs. For example, SELF-REFINE [24] is an iterative self-improvement framework that has shown significant improvements in single-step reasoning tasks by providing feedback on the current output and refining it based on the feedback. Reflexion [15] introduces a framework that uses language feedback as a semantic gradient signal. CRITIC [25] interacts with specific tools to assess and correct errors and flaws in the generated text. SelfCheck [26] corrects mistakes by regenerating specific steps in the reasoning process and comparing them with the original steps.

C. LLMs for Causal Reasoning

One potential solution to the aforementioned problem is to leverage causal reasoning to identify the RCE. In complex long-horizon tasks, errors accumulate over time, and causal reasoning can pinpoint the RCE, enabling the development of effective action plan revisions. Recent research has shown that LLMs have potential in causal reasoning, as they can outperform existing causal reasoning algorithms by utilizing common sense and domain knowledge in tasks such as counterfactual reasoning and real-world causal relationships [17], [27]–[29]. Furthermore, even in the absence of specific domain knowledge, LLMs can still perform limited causal reasoning by leveraging the available data [30].

For complex long-horizon tasks, if reflection frameworks fail to identify RCE, it can lead to ineffective or incorrect plan revisions, which in turn may cause future trials to fail again. Applying reflection frameworks to long-horizon task planning remains a challenging and unresolved issue.

III. Preliminaries

A. Planning Framework

A task can be defined as a tuple $\langle S, O, A, \mathrm{T} \rangle$, where S represents the set of states, O is the set of observations, A denotes the set of actions, and T describes the stochastic transitions [31]. The transition function, formalized as $\mathrm{T}: S \times A \to S$, represents the environmental state changes resulting from actions taken. The objective is to determine a sequence of actions that transitions from the initial state to the target state, thereby forming a plan. Although there is no strict definition for long-horizon tasks, they generally involve 5 to 15 or more steps and necessitate extensive interactions with objects and the environment, in contrast to short-term tasks.

In the reflection method, the sequence of actions is generated by the strategy function. It can be defined as $\Phi(a_i^t | g^t, h_i^t, o_i^t, C^t)$, where Φ determines the next action $a_i^t \in A$, based on the task goal g^t, the historical information $h_i^t = \{o_1^t, a_1^t, \ldots, o_{i-1}^t, a_{i-1}^t\}$, the observation $o_i^t \in O$, and the

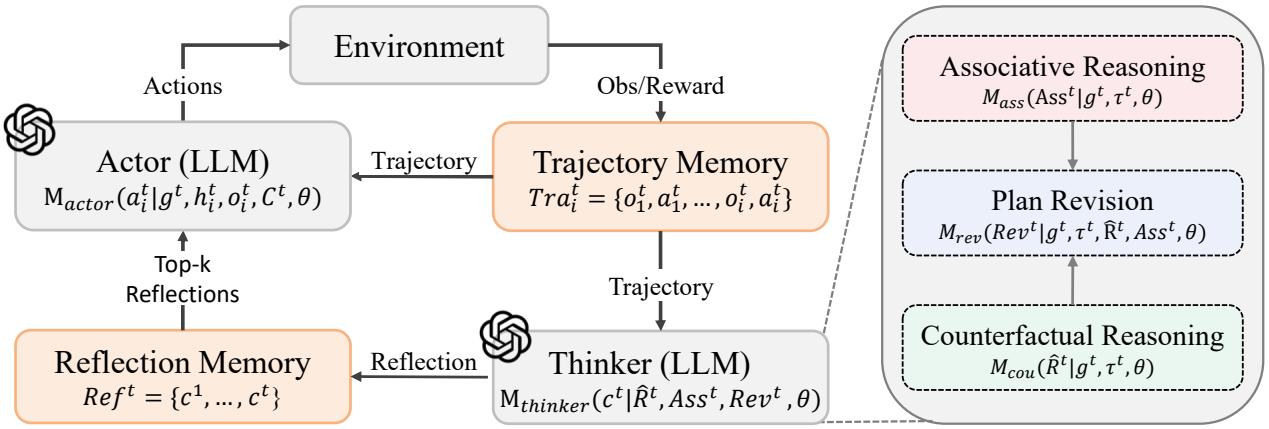

Fig. 2. The framework of CRaFT. Our framework operates by taking historical task information and environmental feedback as input and producing reflection results as output. The framework comprises three stages: 1) Associative Reasoning is tasked with inferring about the associations. 2) Plan Revision is charged with generating the action plan revisions for subsequent trials. 3) Counterfactual Reasoning is responsible for identifying and repairing error actions and analyzing the RCE. Additionally, we have introduced a Memory Management module to manage the historical results generated.

historical reflection information set C^t. Here, t is the index of the trial, and i is the index of the step within the t-th trial.

B. Action Representation and Error Modeling

Each action a_i^t in a trajectory τ^t is characterized by its preconditions $pre(a_i^t)$ and effects $post(a_i^t)$, representing the required and resulting environmental states, respectively.

The *dependency degree* of an action measures its structural complexity by the number of required preconditions:

$$d(a_i^t) = |pre(a_i^t)|. \tag{1}$$

For example, *cool* $\{obj\}$ *with fridge* requires holding the object and being near the fridge, resulting in $d = 2$.

For a failed trajectory τ^t, the error actions is defined as:

$$E_\tau^t = \left\{ a_i^t \;\middle|\; \left(pre(a_i^t) \wedge \neg post(a_i^t)\right) \vee \neg pre(a_i^t) \right\}, \tag{2}$$

where the first case captures actions whose preconditions were met but effects failed, and the second includes those executed with unsatisfied preconditions.

Causal dependencies between actions are denoted by $a_i \rightsquigarrow a_j$, indicating that a_i influences the success of a_j. We write $Repair(R,\tau) \models g$ to indicate that repairing actions in set R yields a trajectory that achieves goal g.

IV. Methodology

In this section, we will provide a detailed introduction to our proposed framework, CRaFT, and its interactions with external systems, as shown in Figure 2. The external system primarily includes the strategy model and the environment. We define the strategy model as the Actor, which uses task goal, historical information, environmental observations, and reflection results as inputs to generate actions for interacting with the environment. Our reflection framework is defined as the Thinker, which integrates task information and environmental feedback to conduct causal reasoning. First, the Thinker applies associative reasoning to infer association information from environmental observations. Then, it integrates the association information with the RCE identified in the previous failed plan to generate action plan revisions. Finally, if the revised plan still fails, the Thinker performs counterfactual reasoning to identify potential RCE by evaluating hypothetical corrections to error actions. Additionally, we have introduced a Memory Management module to manage the historical results generated.

A. Root Cause of Error via Repairable Action Sets

Given a failed trajectory τ^t and error actions E_τ^t, we define the **RCE** as the minimal subset $R^t \subseteq \tau^t$ that satisfies:

- **Causal Coverage:** For every error action $a_{\text{err}}^t \in E_\tau^t$, there exists an action $a_r^t \in R^t$ such that $a_r^t \rightsquigarrow a_{\text{err}}^t$.
- **Minimal Repairability:** R^t has the smallest cardinality for which the repaired trajectory satisfies the task goal:

$$Repair(R^t,\tau^t) \models g^t. \tag{3}$$

Because exhaustive search is intractable in long horizons, we instead compute a feasible repair set $\hat{R}^t \subseteq \tau^t$ that ensures $Repair(\hat{R}^t,\tau^t) \models g^t$ and covers a dynamically identified subset $E_{critical}^t \subseteq E_\tau^t$ via $\forall a_{err}^t \in E_{critical}^t, \exists a_r^t \in \hat{R}^t : a_r^t \rightsquigarrow a_{err}^t$. This relaxation trades global minimality for efficient identification of high-impact intervention points.

B. LLMs as Actor

Following [15], we use the React method [9] as the Actor module *Actor*, due to its outstanding performance in task planning. As an implementation of the strategy function, M_{actor} takes information about the task and environment as inputs and outputs the next action. It can be represented as:

$$a_i^t = M_{actor}(g^t,h_i^t,o_i^t,C^t,\theta), \tag{4}$$

where a_i^t denotes the next action output, g^t represents the task goal, h_i^t encapsulates the historical information, o_i^t indicates the environmental observations, $C^t = \{c^{t-k},\ldots,c^{t-1}\}$ represents the set of reflection outcomes from the past k trials, and θ represents the parameters of LLMs. Additionally, the action and observation at each step will be collected by the memory management module.

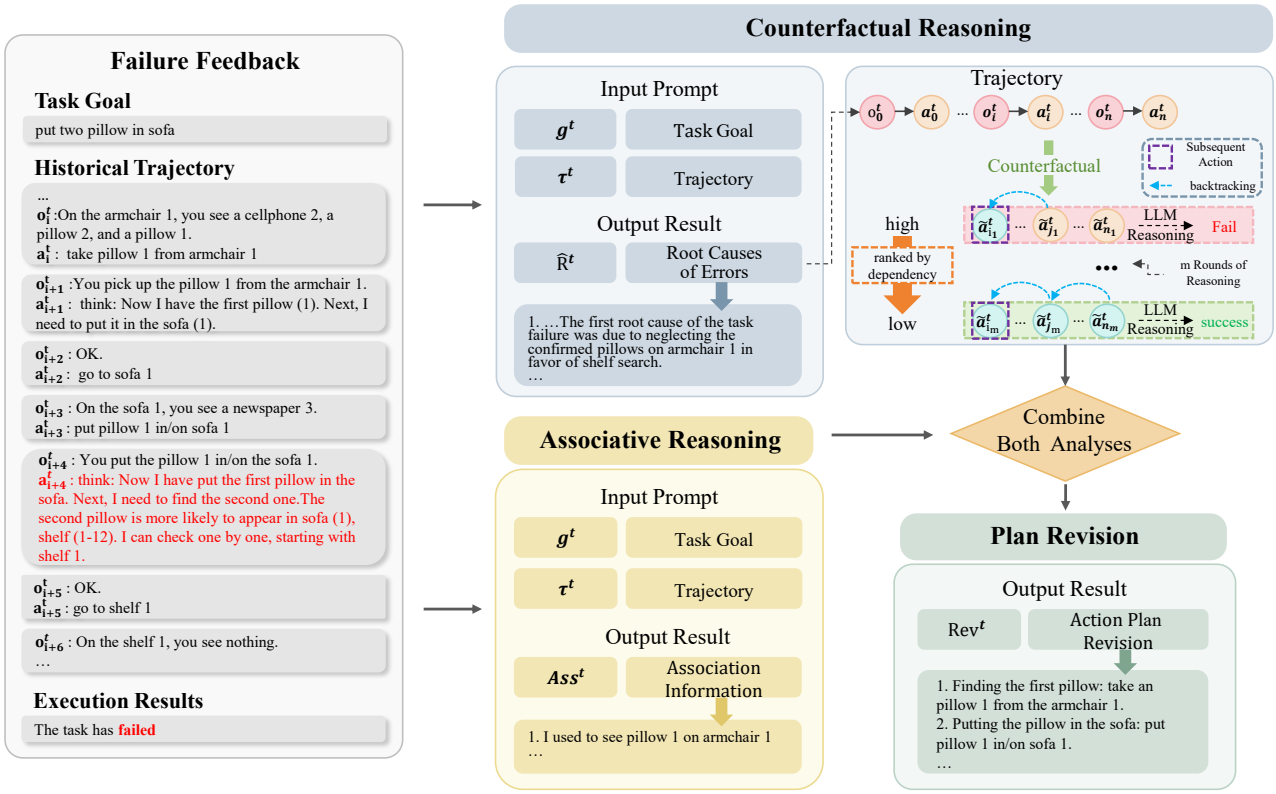

Fig. 3. The detailed illustration of CRaFT. This framework integrates task information and environmental feedback as input and outputs the RCE in the planning process, the association information, and the action plan revisions

C. LLMs as Thinker

To identify the RCE in previous trials and to generate action plan revisions, we designed the Thinker module, $M_{thinker}$. As illustrated in Figure 3, the Thinker consists of three submodules: the Associative Reasoning submodule, tasked with inferring the relationships between objects and containers Ass^t; and the Plan Revision submodule, which generates actionable revisions Rev^t based on inferred associations and identified RCE in the previous action plan—unlike the hypothetical assumptions in counterfactual reasoning; the Counterfactual Reasoning submodule, responsible for identifying error actions and analyzing the RCE $\hat{R}^t$ by simulating counterfactual scenarios—assuming each candidate action had been executed successfully—rather than directly modifying the behavior. It can be formalized as follows:

$$c^t = M_{thinker}(Ass^t, Rev^t, \hat{R}^t, \theta), \tag{5}$$

where c^t represents the collection of outputs from each submodule, Ass^t denotes the object-container association information specified by the task goal, Rev^t denotes the revised action plan, and $\hat{R}^t$ denotes the RCE. Furthermore, the results from the Thinker module will be collected by the memory management module.

Associative Reasoning M_{ass} infers associations between objects and containers based on observations within the trajectory. During trial t, M_{ass} takes the task goal and trajectory as inputs and outputs the associations between objects and containers. To minimize redundancy and shorten the context length, we focus only on objects directly related to the task goal. It can be expressed as:

$$Ass^t = M_{ass}(g^t, \tau^t, \theta). \tag{6}$$

This submodule leverages associative reasoning to accurately infer object locations, thereby improving search efficiency and assisting the subsequent planning Revision module in formulating effective action revisions.

Plan Revision M_{rev} proposes action plan revisions based on the results of counterfactual and associative reasoning. During trial t, M_{rev} takes the identified RCE, the association information between objects and containers, and the task information as inputs, and outputs the action plan revisions. The process is outlined as follows:

$$Rev^t = M_{rev}(g^t, \tau^t, \hat{R}^t, Ass^t, \theta). \tag{7}$$

This submodule utilizes the object-container association information generate correct repairs for the RCE set $\hat{R}^t$ in the previous failed plan, resulting in an optimized action plan revision that significantly improves task success rates.

Counterfactual Reasoning M_{cou} identifies RCE by tracing error actions in the trajectory, prioritizing those with higher dependency degrees, and recursively backtracking to uncover upstream errors that led to unmet preconditions.

To determine whether an action's preconditions were satisfied at the point of failure, we employ a LLM-assisted

precondition evaluation method. For each action, explicit natural language prompts define its preconditions. The LLM is provided with the sequence of prior actions and environmental states from the failed trajectory and is tasked with inferring whether the relevant preconditions were met. If satisfied, the failure is attributed to the action itself and added to the candidate set $\hat{R}^t$; otherwise, backtracking continues to identify responsible upstream actions.

Once the candidate set $\hat{R}^t$ is constructed, we perform counterfactual inference by assuming each action in $\hat{R}^t$ was successfully executed. The LLM then evaluates whether, under these assumptions, the subsequent actions can proceed with their preconditions satisfied, ultimately satisfying the task goal. If the repaired trajectory satisfies the task goal, as verified via LLM-based reasoning, the RCE identification is considered valid. Otherwise, the candidate set is refined through further backtracking and reevaluation. This process can be formally expressed as:

$$\hat{R}^t = M_{cou}(g^t, \tau^t, \theta), \tag{8}$$

where τ^t represents the trajectory. This submodule, through counterfactual reasoning, can efficiently identify the RCE.

The Thinker consists of three submodules, each corresponding to a level of causal reasoning. This structure can efficiently identify RCE, infers association information, and then generates effective action plan revisions, thereby increasing the success rate of long-horizon tasks.

D. Memory Management

For long-horizon tasks, maintaining contextual memory is essential for LLMs. Integrating all necessary information into the prompt may lead to overly long inputs, straining the model's memory capacity. Inspired by the long-short-term memory design in Reflexion, we introduce a memory management module *mem* to manage both execution trajectories and analysis results during planning. The short-term memory component, Tra, stores trajectory information to support subsequent action generation. The long-term memory component, Ref, stores reflection results from the Thinker module to guide future trials, defined as $Ref^t = \{c^1, \ldots, c^t\}$, where Ref^t represents the accumulated reflections up to trial t. This memory structure helps reduce prompt length and supports more efficient planning across trials.

V. EXPERIMENTS

In this section, our framework is evaluated through experiments conducted in a virtual household environment.

A. Experimental Setup

Environment. we conducted five trials to assess different methods in ALFWorld, a semantically-based virtual environment used for training and assessing intelligence. ALFWorld consists of 120 rooms, each dynamically populated with a set of portable objects and static containers. Intelligent agents can interact with these objects and containers to perform everyday tasks such as cooling, cleaning, and heating. The diversity of task environments and goals provides a comprehensive test for intelligent agents.

Dataset. We conducted experiments on 100 tasks from the unseen dataset provided by [32], which includes six task types: Pick & Place, Examine in Light, Clean & Place, Heat & Place, Cool & Place, and Pick Two & Place. To reduce bias from task distribution and support comparative analysis, we grouped them into three categories: Pick Task (Pick & Place), Pick & Act Task (Clean & Place, Heat & Place, Cool & Place, Pick Two & Place), and Examine Task (Examine in Light), with each category averaging around ten steps.

Compare methods. We compared our method with three previous approaches: 1) **Planning-Only**: ProgPrompt [33], which directly generates the next action by inputting a simple context containing the task description into the LLM; 2) **React-Only**: React [9], which incorporates a reasoning process during planning to derive the next action based on reasoning; 3) **React-Reflexion**: Reflexion [15], which integrates a reflection process during multiple trials in React, obtaining the next action based on the reflection from the previous trial. Among all the comparison methods, we use the GPT-4 series as the backbone LLM.

Metrics. We evaluated the performance from three aspects, with the first two metrics specifically targeting the reflection method: **Identification(ID)** assesses whether the reflection can identify the RCE by counting whether each task's reflection identifies the RCE; **Revision(Rev)** evaluates whether the reflection revises errors from previous plans by counting whether each task's reflection has revised these errors; **Success Rate(SR)** measures the effectiveness of the reflection and planning by counting whether each task is successfully completed [20], [34].

B. Results

The primary results are presented in Table I. We can observe that: 1) CRaFT consistently outperforms other methods across all metrics and task categories. This indicates that our framework can efficiently identify the RCE, thereby generating effective action plan revisions and enhancing the success rate. Figure 4 exemplifies the application of CRaFT to long-horizon tasks. 2) A noticeable trend is that a low success rate corresponds to a low rate of identifying the RCE. This highlights a major issue in long-horizon task planning: a low identification of the root causes leads to inaccuracies in action plan revisions, severely impacting the successful execution of subsequent plans. 3) The **React-Only** method and the **React-Reflexion** method exhibit similar performance across various tasks, indicating that reflection methods that only revises action plans for one or a few erroneous steps are not effectively resolving errors encountered in long-horizon task planning. 4) The **React-CRaFT** method demonstrates superior performance compared to methods like **React-Reflexion**, indicating that CRaFT can pinpoint the RCE, thereby effectively enhancing task success rate. Furthermore, the differences in identification, revision, and success rate across different task categories suggest that the effectiveness of reflection methods may be influenced by task complexity.

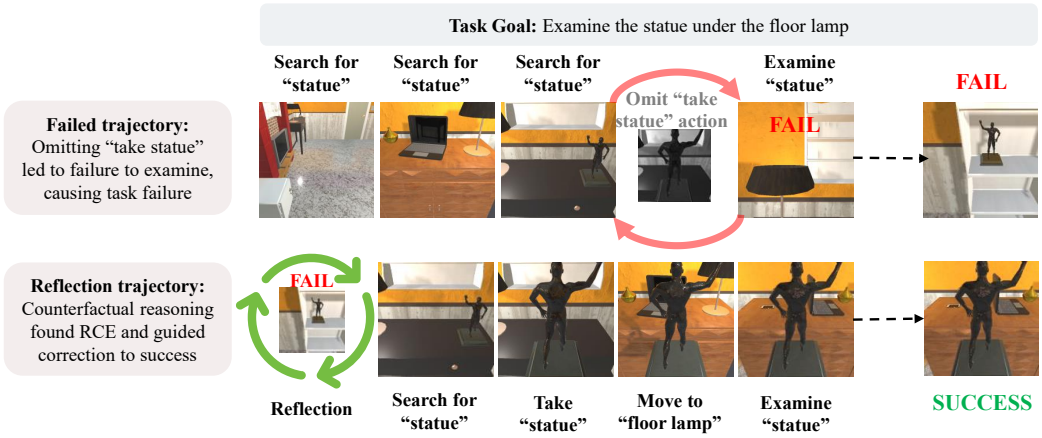

Fig. 4. Example of a failed task successfully replanned using our framework. In the first trial, the agent entered an error loop, repeatedly failing to examine the statue and then searching for it to correct the error, while neglecting to pick it up. In the second trial, our framework identified the root cause of error and generated an optimized action plan revisions, successfully completing the task.

TABLE I

OVERALL PERFORMANCE CRAFT AND BASELINES ACROSS VARIOUS TASKS. OUR FRAMEWORK OUTPERFORMS ALL BASELINES. IDENTIFICATION AND REVISION ARE SOLELY USED TO EVALUATE THE REFLECTION METHOD.

Category	Method	Pick Task			Pick & Act Task			Examine Task		
		ID	Rev	SR	ID	Rev	SR	ID	Rev	SR
w/o reflection	Planning-Only	-	-	56	-	-	54	-	-	18
	React-Only	-	-	81	-	-	73	-	-	82
With reflection	React-Reflexion	68	62	84	68	61	80	40	60	73
	React-CRaFT	**73**	**70**	**85**	**77**	**74**	**85**	**100**	**100**	**100**

VI. ANALYSIS AND DISCUSSION

A. Episode Analysis

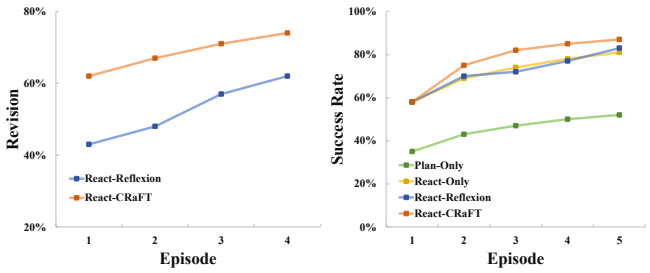

Fig. 5. Result of revision and success rate with different episodes. CRaFT effectively improves the performance of long-horizon tasks.

To analyze the effectiveness of methods, we conducted an episode analysis, with results displayed in Figure 5. Our observations are as follows: 1) Our framework consistently outperformed the baseline across all episodes, demonstrating its effectiveness in improving revision and success rate for long-horizon tasks. 2) Across all episodes, the React-Only method consistently outperforms the Planning-only approach. However, the React-Reflexion method does not exceed the performance of the React-Only method. This further confirms our previous assertion that without identifying the RCE in planning, reflection methods that merely revises action plans for one or a few erroneous steps cannot achieve the goal of optimizing subsequent trials. These results highlight the importance of integrating causal reasoning into reflection, as our framework's ability to accurately identify RCE and generate effective plan revisions significantly enhances success rate in long-horizon task planning.

B. Method Performance Across Different LLMs

To further examine the universality and adaptability of our framework, we conducted experiments with CRaFT using multiple LLMs of varying scales and architectures. As shown in Figure 7, CRaFT achieves comparable performance across all tested models, demonstrating its robustness and generalizability. While overall success rates remain similar, differences in learning trends across episodes suggest that our causality-driven prompt design effectively guides diverse LLMs through the reflection and revision process in long-horizon task planning. These results highlight the LLM-agnostic nature of CRaFT and its ability to maintain stable performance by adapting to the reasoning dynamics of each model through structured causal prompting.

C. Method Performance on the VirtualHome Benchmark

We conducted additional evaluations in VirtualHome, a 3D residential environment featuring various homes and objects. In this setting, the agent must make multi-step decisions and manipulate objects to complete household tasks. To assess performance, we performed comparative experiments on 100 tasks from the NovelTask dataset provided by LID [20].

Manuscript 1293 submitted to 2026 IEEE International Conference on Robotics and Automation (ICRA). Received September 15, 2025.

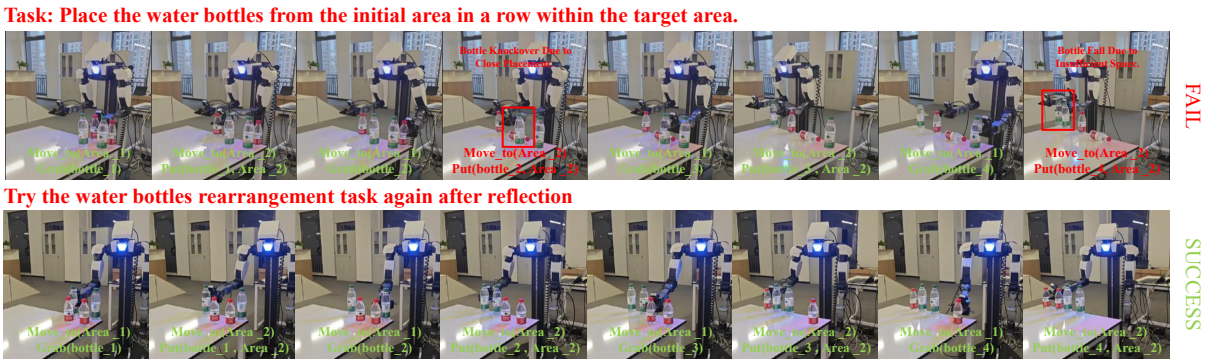

Fig. 6. Example of Successful Replanning After Initial Failure in Water Bottles Rearranging. In the first trial, two errors occurred: the second bottle was placed too close to the first, causing it to topple, and the third bottle was positioned too far from the second, resulting in the fourth bottle falling. In the second trial, our framework identified and revised these errors, successfully completing the bottle rearrangement task.

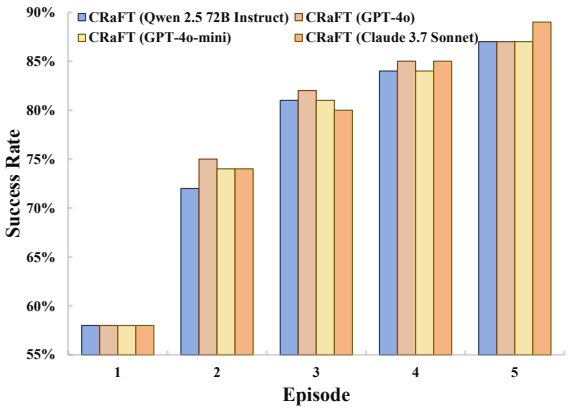

Fig. 7. Result of success rate with different LLMs. CRaFT possesses outstanding universality and robustness.

TABLE II

PERFORMANCE COMPARISON IN VIRTUALHOME. OUR FRAMEWORK OUTPERFORMS ALL BASELINE MODELS.

Method	SR
Planning-Only	50
React-Only	53
React-Reflexion	70
React-CRaFT	**77**

To further verify the generalizability of our method across different environments, we conducted experiments in VirtualHome. The results, presented in Table II, reveal the following key findings: 1) Our method achieves a task success rate of 77% in the VirtualHome environment, significantly outperforming all baseline models. 2) Compared to the React-Only method, the React-Reflexion method increases the success rate by 17%, validating the effectiveness of reflection in task execution. 3) Compared to the React-Reflexion method, our approach further optimizes the reflection process, achieving a 24% higher success rate than the React-Only method, highlighting its advantage in long-horizon task planning.

D. Ablation Study

To assess the contribution of each submodule, we conducted ablation studies on the same 100 tasks from the unseen dataset provided by [32]. The results are detailed in Table III. We report the overall success rate across all tasks. In the **w/o Associative** model, action plan revisions are generated without association information. Compared to our complete model, the three metrics respectively decreased by 16%, 7%, and 9%. The results indicate that association information contributes to improving identification, which in turn enhances success rate. This is linked to the key role of association information in forming effective action plan revisions, which bring trajectories closer to the correct paths and reduce the difficulty of identification. In the **w/o Counterfactual** model, action plan revisions are generated without identifying the RCE. Compared to our complete model, the w/o Counterfactual shows a 7% decrease in revision, leading to an 8% decline in success rate. This suggests that utilizing the counterfactual reasoning submodule to identify the RCE can significantly enhance the revision and thereby increase success rate. The **w/o Both Analyses** model directly generates action plan revisions. Compared to our complete model, this variant's revision decreased by 22%, and the success rate dropped by 12%. These results highlight the critical roles of the associative and counterfactual reasoning submodules in improving revision and success rate.

TABLE III

ABLATION OF VARIOUS SUBMODULES OF CRAFT. ALL DESIGN SUBMODULES CONTRIBUTE TO THE PERFORMANCE. WITHOUT THE COUNTERFACTUAL SUBMODULE, IDENTIFICATION CANNOT BE MEASURED.

Method	ID	Rev	SR
Ours full	**76**	**74**	**87**
w/o Associative	60	67	78
w/o Counterfactual	-	67	79
w/o Both Analyses	-	52	75

VII. REAL-WORLD APPLICATION

To illustrate the practical potential of CRaFT, we developed a CRaFT-based planning system and evaluated it on real-world long-horizon tasks, such as repositioning water bottles. Experiments under varying task complexities showed that the system could identify and correct single-step failures

in simple scenarios, and trace multi-step dependencies to resolve upstream causes in more complex ones. As illustrated in Figure 6, the system effectively handled both isolated and compound errors, showcasing its robustness and applicability in real-world robotic planning.

VIII. CONCLUSIONS

To our knowledge, this is the first attempt to study RCE analysis in the context of long-horizon task planning. Inspired by human intelligence, we propose CRaFT, a framework that enhances reflection through identifying the RCE and generating effective action plan revisions by integrating three causally aligned submodules corresponding to distinct levels of causal reasoning. Additionally, we introduce a memory management module to efficiently handle the context of LLMs. Experimental results from both virtual household environments and real-world scenarios demonstrate that our framework significantly enhances error identification, action plan revisions, and success rates in long-horizon task planning, highlighting its strong generalizability and robustness.

REFERENCES

[1] S. Vemprala *et al.*, "ChatGPT for robotics: Design principles and model abilities," *Microsoft Autonomous Systems and Robotics Research*, vol. 2, p. 20, 2023.

[2] P.-L. Guhur *et al.*, "Instruction-driven history-aware policies for robotic manipulations," in *Conference on Robot Learning*, 2023, pp. 175–187.

[3] S. Reed *et al.*, "A generalist agent," *arXiv preprint arXiv:2205.06175*, 2022.

[4] A. Brohan *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," in *Conference on Robot Learning*, 2023, pp. 287–318.

[5] M. Zhuge *et al.*, "Mindstorms in natural language-based societies of mind," *arXiv preprint arXiv:2305.17066*, 2023.

[6] J. Wu *et al.*, "Tidybot: Personalized robot assistance with large language models," *Autonomous Robots*, vol. 47, no. 8, pp. 1087–1102, 2023.

[7] A. Zhao *et al.*, "Expel: LLM agents are experiential learners," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 17, 2024, pp. 19 632–19 642.

[8] C. Sun, S. Huang, and D. Pompili, "Hierarchical in-context reinforcement learning with hindsight modular reflections for planning," *arXiv e-prints*, pp. arXiv–2408, 2024.

[9] S. Yao *et al.*, "React: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations (ICLR)*, 2023.

[10] A. Prasad *et al.*, "Adapt: As-needed decomposition and planning with language models," in *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024, pp. 4226–4252.

[11] Q. Lyu *et al.*, "Faithful chain-of-thought reasoning," in *The 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (IJCNLP-AACL 2023)*, 2023.

[12] J. Zhang *et al.*, "Fltrnn: Faithful long-horizon task planning for robotics with large language models," in *IEEE International Conference on Robotics and Automation*, 2024, pp. 6680–6686.

[13] S. Longpre *et al.*, "Entity-based knowledge conflicts in question answering," in *Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 7052–7063.

[14] W. Yao *et al.*, "Retroformer: Retrospective large language agents with policy gradient optimization," *arXiv preprint arXiv:2308.02151*, 2023.

[15] N. Shinn *et al.*, "Reflexion: Language agents with verbal reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 36, 2024, pp. 8634–8652.

[16] S. Min *et al.*, "Rethinking the role of demonstrations: What makes in-context learning work?" in *Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 11 048–11 064.

[17] Z. Jin *et al.*, "Cladder: A benchmark to assess causal reasoning capabilities of language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[18] J. Pearl and D. Mackenzie, *The book of why: the new science of cause and effect*. Basic books, 2018.

[19] J. Pearl, "Theoretical impediments to machine learning with seven sparks from the causal revolution," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, ser. WSDM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 3. [Online]. Available: https://doi.org/10.1145/3159652.3176182

[20] S. Li *et al.*, "Pre-trained language models for interactive decision-making," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 31 199–31 212.

[21] B. P. Majumder *et al.*, "Clin: A continually learning language agent for rapid task adaptation and generalization," *arXiv preprint arXiv:2310.10134*, 2023.

[22] T. Kagaya *et al.*, "Rap: Retrieval-augmented planning with contextual memory for multimodal LLM agents," *arXiv preprint arXiv:2402.03610*, 2024.

[23] L. Zheng *et al.*, "Synapse: Trajectory-as-exemplar prompting with memory for computer control," in *The Twelfth International Conference on Learning Representations*, 2023.

[24] A. Madaan *et al.*, "Self-refine: Iterative refinement with self-feedback," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[25] Z. Gou *et al.*, "Critic: Large language models can self-correct with tool-interactive critiquing," *arXiv preprint arXiv:2305.11738*, 2023.

[26] N. Miao, Y. W. Teh, and T. Rainforth, "Selfcheck: Using LLMs to zero-shot check their own step-by-step reasoning," *arXiv preprint arXiv:2308.00436*, 2023.

[27] H. Cai, S. Liu, and R. Song, "Is knowledge all large language models needed for causal reasoning?" *arXiv preprint arXiv:2401.00139*, 2023.

[28] Z. Wang, "Causalbench: A comprehensive benchmark for evaluating causal reasoning capabilities of large language models," in *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, 2024, pp. 143–151.

[29] Y. Zhou *et al.*, "Causalbench: A comprehensive benchmark for causal learning capability of llms," *arXiv preprint arXiv:2404.06349*, 2024.

[30] E. Kıcıman *et al.*, "Causal reasoning and large language models: Opening a new frontier for causality," *arXiv preprint arXiv:2305.00050*, 2023.

[31] M. Lauri, D. Hsu, and J. Pajarinen, "Partially observable markov decision processes in robotics: A survey," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 21–40, 2022.

[32] M. Shridhar *et al.*, "ALFWorld: Aligning Text and Embodied Environments for Interactive Learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. [Online]. Available: https://arxiv.org/abs/2010.03768

[33] I. Singh *et al.*, "Progprompt: Generating situated robot task plans using large language models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530.

[34] W. Huang *et al.*, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International Conference on Machine Learning*, 2022, pp. 9118–9147.