
Improving LLM Reasoning under Uncertainty with Coach-Player Multi-agent

Heewon Park

Soongsil University
Seoul, South Korea

heewon012@soongsil.ac.kr

Minhae Kwon *

Soongsil University
Seoul, South Korea

minhae@ssu.ac.kr

Abstract

Large language models (LLM) are increasingly deployed as interactive agents that reason, plan, and act across multiple turns. While reasoning-based paradigms improve problem solving, and reflection-based methods enhance robustness, single-agent designs remain limited: they often reinforce their own biases and intervene only after errors have accumulated, leading to instability in long-horizon tasks. We introduce a Player–Coach multi-agent LLM system that enables conditional, real-time self-correction during multi-turn decision making. A Player agent engages with the environment through step-wise reasoning and actions, while a Coach agent is invoked only when the Player’s uncertainty is high. The Coach provides targeted metacognitive feedback, such as clarifying objectives or resolving confusion. Feedback is immediately added to the Player’s context. To trigger interventions precisely when needed, we develop a composite uncertainty measure combining normalized entropy and top-two margin. Experiments on ALFWorld show that our approach improves success rates and reduces trajectory length compared to single-agent baselines, without requiring additional training or fine-tuning.

1 Introduction

The rapid progress of LLMs has revealed their capacity to serve as autonomous agents for reasoning, planning, and interaction [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]. Yet when tasks unfold over multiple turns, new difficulties arise. Agents must not only interpret ambiguous observations and adapt to evolving contexts, but also avoid cascading errors that accumulate across long sequences of reasoning and action [14, 15, 16, 17, 18, 19]. Ensuring that LLM agents remain robust, reliable, and efficient in sequential decision making is therefore an essential research question.

Recent advances demonstrate the promise of augmenting LLMs with explicit reasoning. Chain-of-Thought [1], ReAct [2], and Tree-of-Thought [20] improve problem solving by integrating structured reasoning traces with environment interaction. LAC [17] further extends this line with an Actor–Critic formulation, while reflection-based methods such as Reflexion [3] enable post-hoc trajectory revision. Despite their strengths, these approaches mostly operate within a single-agent paradigm, where the model reflects only on its own reasoning history. This design exhibits two key limitations: (i) the agent tends to reinforce its own biases and confusions (e.g., confusing semantically similar objects), and (ii) corrections typically occur after errors have already accumulated, limiting their effectiveness in real-time multi-turn scenarios.

To address these limitations, we propose a Player–Coach multi-agent system for conditional, real-time self-correction in multi-turn environments. Player engages with the environment through a reasoning and acting loop as in ReAct [2], while Coach is invoked only under uncertainty to analyze the Player’s

*Corresponding author: Minhae Kwon (minhae@ssu.ac.kr)

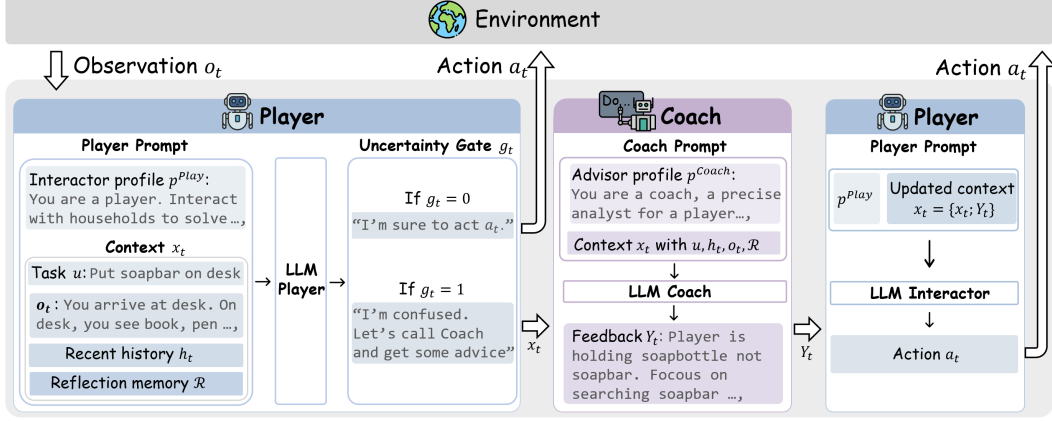


Figure 1: Overview of the proposed Player–Coach system. At each step t , the environment provides an observation o_t , which the Player encodes into a contextual prompt x_t including the task u , recent history h_t , and reflection memory \mathcal{R} . The Player proposes an action a_t with reasoning as ReAct, while an uncertainty gate g_t monitors confidence. If the Player is confident ($g_t = 0$), the action is executed directly. If uncertainty is high ($g_t = 1$), the Player queries the Coach, which analyzes x_t and produces feedback Y_t containing rationale, reminders, or corrective suggestions. This feedback is appended to the Player’s context, prompting re-scoring of admissible actions before executing the final action. By separating fast interaction from conditional metacognitive intervention, the framework enables real-time error correction and yields more reliable trajectories.

trajectory from an external vantage point. The Coach provides structured metacognitive feedback by clarifying objectives, surfacing forgotten observations, and resolving confusions. This feedback is immediately incorporated into the Player’s decision process, ensuring that fast interaction and deliberate correction are explicitly separated. As a result, the architecture mitigates self-reinforcing loops and yields more accurate, stable, and efficient multi-turn behavior. Detailed description of realted works is provided in Appendix A.

2 Method

We propose a *Player–Coach* multi-agent LLM system that performs conditional, real-time intervention during sequential decision making. Figure 1 shows the overview of the proposed Player–Coach system. The system comprises two distinct agents, each instantiated by a role-specific prompt (profile): a *Player* that directly engages with the environment by alternating reasoning and acting, and a *Coach* that is invoked only under high uncertainty to supply targeted, metacognitive feedback. By separating interaction from externalized analysis, the Coach can scrutinize the Player’s trajectory from an independent vantage point, mitigating self-reinforcing errors and producing trajectories that are both more accurate and more efficient. The overall algorithmic flow of the proposed system is provided as pseudocode in Appendix B.

2.1 Player for Reasoning and Acting

The Player is the agent that directly engages with the environment. At each step t , it follows a ReAct-style loop, alternating between a short natural-language *reason* and a concrete *act*. Given the task instruction and a bounded history of recent steps, the Player constructs a contextual prompt and scores the admissible actions to produce a fast proposal.

Formally, the environment \mathcal{E} provides the current observation o_t and the set of admissible actions $\mathcal{A}_t = \{a^{(1)}, \dots, a^{(K_t)}\}$, where $K_t = |\mathcal{A}_t|$ is the number of admissible actions at step t . The Player maintains the most recent n steps of history

$$h_t = ((o_{t-n}, a_{t-n}), \dots, (o_{t-1}, a_{t-1})),$$

and builds a textual context

$$x_t = \text{Concat}(u, h_t, o_t, \mathcal{R}),$$

where u is the task instruction and \mathcal{R} is a short memory of prior reflections.

Each admissible action string $a = (w_1, \dots, w_{|a|})$ is assigned a sequence log-likelihood under the Player profile p^{Play} as follows.

$$\ell(a \mid x_t) = \sum_{j=1}^{|a|} \log f_{\theta}(p^{Play}, x_t, w_{<j})[w_j] \quad (1)$$

Here, f_{θ} denotes the base LLM and $[\cdot]$ indexes the probability of token w_j . These scores are normalized into a preference distribution as follows.

$$q(a \mid x_t) = \frac{\exp(\ell(a \mid x_t))}{\sum_{a' \in \mathcal{A}_t} \exp(\ell(a' \mid x_t))}, \quad a \in \mathcal{A}_t \quad (2)$$

The Player then outputs a fast proposal via greedy selection $a_t \in \arg \max_{a \in \mathcal{A}_t} q(a \mid x_t)$.

2.2 Uncertainty-Guided Intervention Trigger

The Player invokes the Coach only when its own action preference distribution $q(a \mid x_t)$ in (2) indicates high uncertainty. Specifically, we monitor two complementary signals: (i) normalized entropy, which captures diffuse uncertainty spread over many actions, and (ii) the top-two margin, which captures local ambiguity between the most likely candidates. By combining these signals, the system reduces both spurious interventions and missed opportunities.

First, the normalized entropy is defined as follows.

$$\bar{H}_t = -\frac{1}{\log K_t} \sum_{a \in \mathcal{A}_t} q(a \mid x_t) \log q(a \mid x_t) \quad (3)$$

Here, high \bar{H}_t indicates uncertainty dispersed across many admissible actions.

Second, we define the top-two margin

$$m_t = q_{(1)} - q_{(2)}, \quad (4)$$

where $q_{(1)} \geq q_{(2)}$ are the two largest probabilities. A small m_t signals indecision between the most likely actions.

The uncertainty gate g_t is activated whenever either condition holds

$$g_t = \mathbb{I}[\bar{H}_t \geq \tau_H \vee m_t \leq \tau_m], \quad (5)$$

where τ_H and τ_m are hyperparameters chosen in $(0, 1)$ and tuned on a held-out validation set. This composite rule balances coverage and precision, ensuring the Coach intervenes only when the Player is genuinely uncertain.

2.3 Coach as Externalized Metacognition and Fusion

The Coach complements the Player by providing targeted metacognitive feedback from an externalized perspective. Unlike the Player, which focuses on rapid reasoning and acting, the Coach is invoked only when the uncertainty gate g_t in (5) is triggered. This allows the Coach to critically examine the Player’s context and trajectory, surfacing forgotten facts, resolving object confusions, or re-stating the task objective.

Formally, when $g_t = 1$, the Player queries the Coach with the current context as follows.

$$Y_t = f_{\theta}(p^{Coach}, x_t) \quad (6)$$

Here, p^{Coach} is the Coach profile. The Coach generates a structured feedback Y_t consisting of a rationale, key facts, and either a recommended admissible action or a minimal probe.

The final action at step t is chosen as

$$a_t = \begin{cases} \arg \max_{a \in \mathcal{A}_t} q(a \mid x_t), & g_t = 0, \\ \arg \max_{a \in \mathcal{A}_t} q(a \mid \text{Concat}(x_t, Y_t)), & g_t = 1, \end{cases} \quad (7)$$

ensuring that Coach feedback directly influences the Player’s decision while remaining within the admissible action set R^{trial} . At the end of each trajectory, the Coach also provides a trial-level reflection that is stored in \mathcal{R} for future steps.

Table 1: Performance comparison on ALFWorld with LLaMA-3-8B. We report success rate (%) and average trajectory length for successful episodes (mean \pm std). Best results are **bolded**.

Method	Seen		Unseen	
	Success rate (%)	Trajectory length	Success rate (%)	Trajectory length
ReAct [2]	64.9	13.4 \pm 6.5	59.0	15.1 \pm 7.2
LAC [17]	67.1	14.2 \pm 6.7	61.1	15.3 \pm 6.9
Ours (Fixed-Step)	72.4	12.7 \pm 5.2	62.4	13.3 \pm 6.8
Ours (Uncertainty-Gated)	77.6	11.3 \pm 4.1	66.9	12.5 \pm 5.8

3 Simulation Results

In this section, we present empirical evaluations of the proposed Player–Coach framework. We assess its effectiveness on long-horizon, multi-turn decision making tasks and compare against strong single-agent baselines. Example prompts and detailed trajectory cases are provided in Appendix C.

3.1 Experimental Setup

We evaluate our approach on ALFWorld benchmark [21], a widely used text-based household environment that requires agents to complete everyday tasks through sequences of high-level actions. The benchmark contains 140 seen and 134 unseen evaluation tasks. An agent receives a reward of 1 if the goal is achieved and 0 otherwise. The main challenge of ALFWorld lies in locating target objects and executing household tasks that demand commonsense reasoning over long action sequences.

As baselines, we include ReAct [2], which alternates reasoning and acting under a single-agent design, and LAC [17], an actor–critic style single-agent framework. Both represent strong single-agent paradigms. Our method is implemented with the LLaMA-3-8B model [22] as the pretrained backbone, and trajectories are capped at a maximum of 50 steps.

We also compare two variants of our Player–Coach system: (i) Fixed-Step Coach, where the Coach is invoked deterministically every 20 steps, and (ii) Uncertainty-Gated Coach, where interventions are adaptively triggered by the proposed uncertainty gate g_t in (5). This ablation highlights the effectiveness of uncertainty-aware conditional intervention.

3.2 Performance Comparison

Table 1 shows success rates and average trajectory lengths on ALFWorld benchmark. Both variants of our Player–Coach framework outperform the single-agent baselines, demonstrating the benefit of externalized metacognitive intervention. The Fixed-Step Coach yields clear improvements over ReAct and LAC, showing that periodic external feedback can mitigate self-reinforcing errors. However, this approach still wastes interventions when the Player is confident and may miss critical corrections between scheduled steps.

In contrast, the Uncertainty-Gated Coach achieves the best overall performance, improving success rates by over 10% in the seen split and 5.8% in the unseen split compared to ReAct, while also reducing average trajectory length. These results highlight that conditional interventions, triggered precisely when uncertainty is high, are more effective and efficient than fixed scheduling. The improvements on the unseen tasks further indicate that our uncertainty-driven mechanism generalizes well beyond familiar environments.

4 Conclusion

We proposed a Player–Coach multi-agent framework for LLM-based decision making under uncertainty. The Player engages in reasoning and acting, while the Coach provides targeted feedback only when the Player’s uncertainty is high. This separation enables conditional, real-time correction that mitigates self-reinforcing errors and improves both success rate and efficiency. Experiments on ALFWorld show that our approach consistently outperforms single-agent baselines without requiring additional training. We believe this highlights the potential of role-separated, uncertainty-aware architectures as a simple yet powerful direction for robust multi-turn LLM agents.

References

- [1] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in neural information processing systems*, volume 35, pages 24824–24837, 2022.
- [2] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.
- [3] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 8634–8652, 2023.
- [4] Zhihan Liu, Hao Hu, Shenao Zhang, Hongyi Guo, Shuqi Ke, Boyi Liu, and Zhaoran Wang. Reason for future, act for now: A principled framework for autonomous llm agents with provable sample efficiency. In *International Conference on Machine Learning*, 2023.
- [5] Ye Bai, Minghan Wang, and Thuy-Trang Vu. MAPLE: Multi-agent adaptive planning with long-term memory for table reasoning. *arXiv preprint arXiv:2506.05813*, 2025.
- [6] Ruiwen Zhou, Yingxuan Yang, Muning Wen, Ying Wen, Wenhao Wang, Chunling Xi, Guoqiang Xu, Yong Yu, and Weinan Zhang. TRAD: Enhancing llm agents with step-wise thought retrieval and aligned decision. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.
- [7] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2Web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, volume 36, pages 28091–28114, 2023.
- [8] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690, 2024.
- [9] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.
- [10] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594, 2023.
- [11] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation framework. In *Conference on Language Modeling*, 2024.
- [12] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for “mind” exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, pages 51991–52008, 2023.
- [13] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Conference on Empirical Methods in Natural Language Processing*, 2024.
- [14] Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, et al. Mt-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. In *Association for Computational Linguistics*, 2024.
- [15] Ma Chang, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems*, volume 37, pages 74325–74362, 2024.

- [16] Kaustubh Deshpande, Ved Sirdeshmukh, Johannes Baptist Mols, Lifeng Jin, Ed-Yeremai Hernandez-Cardona, Dean Lee, Jeremy Kritz, Willow E Primack, Summer Yue, and Chen Xing. Multichallenge: A realistic multi-turn conversation evaluation benchmark challenging to frontier LLMs. In *Findings of the Association for Computational Linguistics*, pages 18632–18702, 2025.
- [17] Heng Dong, Kefei Duan, and Chongjie Zhang. Enhancing decision-making of large language models via actor-critic. In *International Conference on Machine Learning*, 2025.
- [18] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Advances in neural information processing systems*, 2023.
- [19] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. VOYAGER: An open-ended embodied agent with large language models. *arXiv preprint arXiv: Arxiv-2305.16291*, 2023.
- [20] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 11809–11822, 2023.
- [21] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.
- [22] AI Meta. Llama 3: Open foundation and instruction models, 2024.

A Related Works

A.1 LLM-Based Sequential Reasoning and Acting

Large language models (LLMs) have been increasingly applied as agents that perform sequential reasoning and acting in interactive environments. Early work such as Chain-of-Thought (CoT) [1, 2, 17, 20, 17, 18, 19] shows that prompting with explicit intermediate reasoning steps significantly improves problem solving. Tree-of-Thought (ToT) [20] extends this idea by enabling structured search over reasoning paths, while ReAct [2] integrates reasoning and acting in a loop to guide decision making. Beyond prompting strategies, LAC [17] introduces an Actor–Critic style framework for sequential decision making without gradient-based optimization, and Voyager [19] demonstrates open-ended embodied agents in Minecraft driven by LLMs. These approaches highlight the promise of augmenting LLMs with explicit reasoning traces and interaction loops, but they typically rely on a single-agent paradigm.

A.2 Reflection and Collaborative Correction

A substantial body of work enhances LLM robustness through either self-reflection or multi-agent collaboration. Reflection-based methods focus on enabling agents to revise their own trajectories: Reflexion [3] introduces verbal reinforcement learning to improve future behavior, Constitutional AI [9] leverages normative principles for self-feedback, and Self-Refine [10] and RAFA [4] iteratively refine outputs using self-generated or adaptive signals. Complementary to this, multi-agent frameworks externalize correction by introducing role separation and inter-agent communication. AutoGen [11] and CAMEL [12] employ structured role-based workflows, MAD [13] fosters divergent thinking through debate, and MAPLE [5] adopts an actor–critic formulation to coordinate multiple agents. These approaches highlight the promise of reflective or collaborative correction, yet many rely on post-hoc revision, fixed protocols, or additional training. Our Player–Coach framework advances this direction by providing real-time, uncertainty-triggered metacognitive correction without extra training overhead.

B Pseudocode

Algorithm 1 Player–Coach Framework

- 1: **Input:** Task instruction u , environment \mathcal{E} , Player profile p^{Play} , Coach profile p^{Coach}
 - 2: **Output:** Action sequence $\{a_1, a_2, \dots\}$
 - 3: **for** each step $t = 1, 2, \dots$ until termination **do**
 - 4: Receive observation o_t and admissible actions \mathcal{A}_t from \mathcal{E}
 - 5: Build context $x_t = \text{Concat}(u, h_t, o_t, \mathcal{R})$
 - 6: Compute log-likelihood scores $\ell(a | x_t)$ for $a \in \mathcal{A}_t$ under p^{Play}
 - 7: Normalize into preference distribution $q(a | x_t)$
 - 8: Evaluate uncertainty signals: entropy \bar{H}_t and margin m_t
 - 9: **if** $\bar{H}_t \geq \tau_H$ **or** $m_t \leq \tau_m$ **then**
 - 10: Query Coach: $Y_t = f_\theta(p^{Coach}, x_t)$
 - 11: Update context: $x_t \leftarrow \text{Concat}(x_t, Y_t)$
 - 12: **end if**
 - 13: Select action $a_t = \arg \max_{a \in \mathcal{A}_t} q(a | x_t)$
 - 14: Execute a_t in \mathcal{E} and update history h_{t+1}
 - 15: **end for**
 - 16: Receive final trial-level reflection R^{trial} from Coach
 - 17: Update reflection memory $\mathcal{R} \leftarrow \mathcal{R} \cup \{R^{\text{trial}}\}$
-

C Prompts

Table 1: An example of trajectory for ALFWorld pick_two_obj_and_place.

You are in the middle of a room. Looking quickly around you, you see an armchair 1, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a dresser 1, a garbagecan 1, a safe 1, a shelf 12, a shelf 11, a shelf 10, a shelf 9, a shelf 8, a shelf 7, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, a shelf 1, a sidetable 1, and a sofa 1. Your task is to: put two pillow in sofa.

> [Player] Act 1: think: To solve the task, I need to find and take the first pillow, then put it in sofa, then find and take the second pillow, then put it in sofa.

OK.

> [Player] Act 2: think: First I need to find the first pillow. A pillow is more likely to appear in armchair (1), cabinet (1-4), drawer (1-5), dresser (1), sidetable (1), sofa (1). I can check one by one, starting with armchair 1.

OK.

> [Player] Act3: go to armchair 1

You arrive at armchair 1. On the armchair 1, you see a cellphone 2, a pillow 2, and a pillow 1.

> [Player] Act 4: take pillow 1 from armchair 1

You pick up the pillow 1 from the armchair 1.

> [Player] Act 5: think: Now I find the first pillow (1). Next, I need to put it in sofa 1.

OK.

> [Player] Act 6: go to sofa 1

You arrive at sofa 1. On the sofa 1, you see a newspaper 3.

> [Uncertainty] H=2.054, margin=0.060, $g_t=1 \rightarrow$ Coach invoked.

> [Coach] Rationale: You are close, but you forgot to put the first pillow in sofa. You should put the first pillow in sofa before finding the second pillow.

> [Coach] Plan: ['put pillow 1 in sofa 1', 'go to armchair 2', 'take second pillow', 'put second pillow in sofa 1']

> [Coach] Advice: Put the first pillow in sofa before finding the second one.

> [Player] Act 7: move pillow 1 to sofa 1

On the sofa 1, you see a newspaper 3, and a pillow 1.

> [Player] Act 8: go to armchair 1

You arrive at armchair 1. On the armchair 1, you see a cellphone 2, and a pillow 2.

> [Player] Act 9: take pillow 2 from armchair 1

You pick up the pillow 2 from the armchair 1.

> [Player] Act 10: think: Now I find the second pillow (2). Next, I need to put it in sofa 1.

OK.

> [Player] Act 11: go to sofa 1

You arrive at sofa 1. On the sofa 1, you see a newspaper 3, and a pillow 1.

> [Player] Act 12: move pillow 2 to sofa 1

You move the pillow 2 to the sofa 1.

> [Coach] Trial reflection: Consider the possibility of multiple objects in a location before taking an action.

Outcome: success

Table 2: Example trajectory on ALFWorld pick_clean_then_place_in_recep.

You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1. Your task is to: put a clean cloth in countertop.

> [Player] Act 1: think: To solve the task, I need to find and take a cloth, then clean it with sinkbasin, then put it in countertop.
OK.

> [Player] Act 2: think: First I need to find a cloth. I will check the cabinets starting from cabinet 4.
OK.

> [Player] Act 3: go to cabinet 4
You arrive at cabinet 4. The cabinet 4 is closed.

> [Player] Act 4: open cabinet 4
You open the cabinet 4. The cabinet 4 is open. In it, you see a soapbottle 2.

> [Player] Act 5: go to cabinet 3
You arrive at cabinet 3. The cabinet 3 is closed.

> [Player] Act 6: open cabinet 3
You open the cabinet 3. The cabinet 3 is open. In it, you see a cloth 2.

> [Player] Act 7: take cloth 2 from cabinet 3
You pick up the cloth 2 from the cabinet 3.

> [Player] Act 8: go to sinkbasin 1
You arrive at sinkbasin 1. On the sinkbasin 1, you see a soapbar 1.

> [Player] Act 9: clean cloth 2 with sinkbasin 1
You clean the cloth 2 using the sinkbasin 1.

> [Player] Act 10: go to countertop 1
You arrive at countertop 1. On the countertop 1, you see a candle 1, a cloth 3, and a spraybottle 1.

> [Player] Act 11: think: I might still need soap from cabinet 4 to clean a cloth before placing it on the countertop. I should go back and fetch soap first.
OK.

> [Uncertainty] H=1.87, margin=0.78, $g_t = 1 \rightarrow$ Coach invoked.

> [Coach] Rationale: You already cleaned cloth 2. The goal is to place a clean cloth on the countertop, not to fetch soap now. Also note there is an unclean cloth 3 on the countertop--do not switch targets.

> [Coach] Plan: (1) Verify the scene, (2) Place cloth 2 on countertop 1.

> [Coach] Advice: Skip fetching soap; execute the placement with the cleaned cloth you hold.

> [Player] Act 12: examine countertop 1
On the countertop 1, you see a candle 1, a cloth 3, and a spraybottle 1.

> [Player] Act 13: move cloth 2 to countertop 1
You move the cloth 2 to the countertop 1.

> [Coach] Trial reflection: After cleaning an item, prioritize placing it to meet the goal rather than restarting the cleaning process.

Outcome: success

Table 3: Example trajectory on ALFWorld pick_heat_then_place_in_recep.

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: put a hot apple in fridge.

> [Player] Act 1: think: First I need to find an apple. I will start with the fridge.

OK.

> [Player] Act 2: go to fridge 1

You arrive at fridge 1. The fridge 1 is closed.

> [Player] Act 3: open fridge 1

You open the fridge 1. In it, you see a bowl 1, a cup 2, a cup 1, a lettuce 1, a mug 2, a potato 1, and a tomato 1.

> [Player] Act 4: think: No apple here. Maybe I should check cabinets one by one.

OK.

> [Uncertainty] H=2.05, margin=0.09, $g_t = 1 \rightarrow$ Coach invoked.

> [Coach] Rationale: Apples are more likely on countertops.

> [Coach] Plan: (1) Go to countertop 1, (2) take apple, (3) heat it with microwave, (4) place in fridge.

> [Coach] Advice: Check the countertop before searching every cabinet.

> [Player] Act 5: go to countertop 1

You arrive at countertop 1. On the countertop 1, you see an apple 1, an apple 2, and a mug 3.

> [Player] Act 6: take apple 1 from countertop 1

You pick up the apple 1 from the countertop 1.

> [Player] Act 7: go to microwave 1

You arrive at microwave 1. The microwave 1 is closed.

> [Player] Act 8: heat apple 1 with microwave 1

You heat the apple 1 using the microwave 1.

> [Player] Act 9: think: The apple is now hot. Next, I should place it in fridge 1.

OK.

> [Player] Act 10: go to fridge 1

You arrive at fridge 1. The fridge 1 is open.

> [Player] Act 11: move apple 1 to fridge 1

You move the apple 1 to the fridge 1.

> [Coach] Trial reflection: Prioritize likely locations (countertops) before exploring less relevant places.

Outcome: success
