# **Collaborative Tree Search for Enhancing Embodied Multi-Agent Collaboration**

Lizheng Zu<sup>1, 2</sup>, Lin Lin<sup>1</sup>, Song Fu<sup>1</sup>, Na Zhao<sup>3</sup>, Pan Zhou<sup>4</sup> <sup>1</sup>Harbin Institute of Technology, <sup>2</sup>Nanyang Technological University, <sup>3</sup>Singapore University of Technology and Design, <sup>4</sup>Singapore Management University

zuli\_edu@outlook.com, waiwaiyl@hit.edu.cn, fusong@hit.edu.cn

na\_zhao@sutd.edu.sg, panzhou@smu.edu.sg

## Abstract

Embodied agents based on large language models (LLMs) face significant challenges in collaborative tasks, requiring effective communication and reasonable division of labor to ensure efficient and correct task completion. Previous approaches with simple communication patterns carry erroneous or incoherent agent actions, which can lead to additional risks. To address these problems, we propose Cooperative Tree Search (CoTS), a framework designed to significantly improve collaborative planning and task execution efficiency among embodied agents. CoTS guides multi-agents to discuss long-term strategic plans within a modified Monte Carlo tree, searching along LLMdriven reward functions to provide a more thoughtful and promising approach to cooperation. Another key feature of our method is the introduction of a plan evaluation module, which not only prevents agent action confusion caused by frequent plan updates but also ensures plan updates when the current plan becomes unsuitable. Experimental results show that the proposed method performs excellently in planning, communication, and collaboration on embodied environments (CWAH and TDW-MAT), efficiently completing long-term, complex tasks and significantly outperforming existing methods.

# 1. Introduction

Embodied agents are increasingly capable of perceiving and interacting with their environments like humans, autonomously managing planning, decision-making, actions, and execution tasks [23, 36]. Recent advances have generated substantial interest in these agents, especially for applications requiring complex task execution in dynamic environments, such as robotic operations in warehouses or domestic settings. In particular, collaborative task completion among multiple agents enhances both efficiency and adaptability, especially in dynamic scenarios [5, 10]. However, collaboration among embodied agents introduces significant challenges: agents must not only perceive and understand their environment but also communicate, share information, divide tasks, and coordinate actions responsively. For example, consider autonomous robots in a home responding to a request: "Bring me the iPad and apple, and put the milk in the refrigerator." Achieving this requires multi-level coordination, including optimal search strategies, task prioritization, and efficient movement planning.

Large language models (LLMs) have recently provided embodied agents with advanced natural language understanding, dialogue, and reasoning abilities [1, 3, 38, 40]. These capabilities allow LLMs to decompose complex and long-term tasks into a sequence of manageable sub-goals, making LLM-driven agents a promising alternative to traditional reinforcement learning models [14, 34] which are difficult to train and often generalize poorly [7, 21, 39].

However, enabling embodied agents to work together in decentralized environments remains a significant and underexplored challenge, as it requires long-term planning and coherent decision-making to coordinate actions efficiently. Early attempts at multi-agent collaboration, such as CoELA [43] and RoCo [24], demonstrate progress but also reveal limitations. For instance, as shown in Fig. 1 (b), CoELA facilitates collaboration by sharing updates through natural language when sub-tasks are completed, yet each agent's decision-making process remains independent, resulting in suboptimal coordination. In contrast, as illustrated in Fig. 1 (c), RoCo develops multi-agent work plans through agent discussions and environmental interactions. However, RoCo relies on a single reasoning path, which is susceptible to the randomness and unpredictability of LLM outputs. This can lead to inefficient or incorrect planning, particularly in applications requiring precision, where such errors can have serious consequences.

**Contributions.** To address these limitations, we propose *Cooperative Tree Search* (CoTS), a framework designed to significantly improve collaborative planning and task exe-

<sup>\*</sup>Corresponding author.

Task: transport 2 oranges, 3 apples, 1 banana, 3 breads, 1 burger to the bed.

**Background**: Agents randomly appear in a home environment with a total of six rooms and need to discover objects through exploration. Each agent can carry up to two objects. Finding and using containers can improve efficiency.



Figure 1. The task execution processes of CoELA, RoCo, and our CoTS. (1) In CoELA, agents update each other on sub-task completion and individually determine their next actions without long-term strategic planning. (2) RoCo uses multi-round dialogues and summarizes the plan at the end, but its single-dialogue chain can lead to suboptimal plans and even errors due to the randomness and unpredictability of LLMs. (3) Within a modified Monte Carlo Tree Search framework, CoTS allows agents to evaluate multiple planning paths, enabling them to collaboratively deliberate and identify the most effective course of action based on task progress and environmental context.

cution efficiency among embodied agents. Inspired by human decision-making, CoTS allows agents to search multiple planning paths by combining exploration and backtracking, enabling them to collaboratively deliberate and identify the most effective course of action based on task progress and environmental context.

As illustrated in Fig. 1 (d), CoTS advances agent collaboration in embodied environments through two core innovations. First, we introduce a collaborative planning module based on a modified Monte Carlo Tree Search (MCTS) framework, which enables agents to explore multiple planning paths and select the most coherent and efficient long-term plan. Unlike traditional MCTS which relies on simulation-based rewards, CoTS incorporates a nonsimulated LLM-based reward function suited for high-cost physical movement interactions in embodied environments. In this way, CoTS transforms the traditional linear sequence of action steps into a search process, exploring potential paths within a shared reasoning and action space to develop robust collaborative strategies. Second, CoTS features a plan evaluation module, that actively monitors the relevance and feasibility of the selected plan during execution. This mechanism allows agents to adapt plans as needed, reducing unnecessary disruptions from frequent plan updates while still accommodating environmental changes.

Our experiments on two long-term multi-objective planning tasks — TDW-MAT [43] and C-WAH [8] — demonstrate that CoTS significantly outperforms current state-ofthe-art methods, *e.g.*, CoELA [43] and RoCo [24], in terms of task completion rate and efficiency. These results highlight the effectiveness and superiority of CoTS, underscoring its potential for high-stakes applications that demand reliable multi-agent collaboration.

## 2. Related Work

LLM-based Agents. Recent advances in LLMs have unlocked near-human reasoning and planning capabilities [1, 27, 37]. Researchers have leveraged LLMs as the "brain" or controller of agents, expanding their perceptual and action capacities through multimodal perception and tool use [6, 25, 32, 38, 40, 45]. Multi-agent systems [16, 35] show particular promise, enabling agents to collaborate on planning, discussion, and decision-making-similar to how humans solve complex problems in teams [26, 33]. Works such as MetaGPT [13], CAMEL [19], and ChatDev [31] decompose complex tasks into simpler, manageable subtasks to reduce hallucinations and improve task-solving capabilities. However, embodied multi-agent systems [20, 44] require advanced communication and face significant challenges in navigating a vast action search space in complex, dynamic environments, especially for long-term tasks.

**Embodied Multi-Agent Cooperation.** Research on embodied multi-agent systems is still emerging. Habitat [30] explores agents' social awareness, enabling adaptive assistance strategies in collaboration. Co-NavGPT [41] uses LLMs for multi-robot collaborative goal navigation, encoding environmental data to improve scene understanding. Organized-LLM-Agents [11] reduces redundancy in multi-agent collaboration through structured prompts, enhancing coordination. CoELA [43] enables collaborative agents to communicate and plan through natural language and chain-of-thought processes, yet lacks strategic long-term planning. RoCo [24] and CaPo [22] advance multi-agent collaboration with high-level communication and path planning, but their single reasoning path often leads to suboptimal plans and even errors. Although some existing work



Figure 2. The framework of CoTS. Upon receiving a new task in the embodied environment, each embodied agent initially employs the Perception Module to gather raw observational data, which is then stored in its Memory Module. In the collaborative planning module, agents guide the exploration of collaborative plans toward more promising directions through the collaborative tree search. The explored plans are retained within the memory module as a special 'extended memory' and are periodically updated based on assessments from the planning evaluation module. Finally, agents perform actions aligned with the collaborative plan and tasks.

on LLMs involves multi-branch reasoning [2, 17, 40, 42], they are inadequate for complex embodied environments and agent collaboration. Our goal is to develop embodied agents capable of searching long-term strategic planning.

# 3. Preliminaries

**Task Assumption.** As introduced by previous work [9, 24], multi-agent collaborative tasks can be viewed as a Markov game, which is defined by a tuple  $(\mathcal{N}, \mathcal{S}, \mathcal{A}, P, r, \gamma)$ , in which  $\mathcal{N}$  is the set of agents (*e.g.* Alice and Bob),  $\mathcal{S}$  is the state space,  $\mathcal{A} = \prod_{i=1}^{n} \mathcal{A}^{i}$  is the combination of all action spaces,  $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$  is the transition probability function,  $r : S \times A \rightarrow \mathbb{R}$  is the reward score, and  $\gamma \in [0,1)$  is the discount factor. In the task, each agent at time step  $t \in \mathbb{N}$  obtains the information by perception module from the embodied environment  $s_t \in S$ , chooses and executes an action  $a_t^i \in \mathcal{A}^i$  from its available policy  $\pi^i(\cdot|s_t)$ . This will along with the actions of other agents, constitutes a joint action  $\mathbf{a}_t = (a_t^1, a_t^2, ..., a_t^n) \in \mathcal{A}$  derived from the joint policy  $\pi(\cdot|s_t) = \prod_{i=1}^n \pi^i(\cdot|s_t)$ . Then agents perceive a new state  $s_{t+1}$  with probability  $P(s_{t+1}|s_t, \mathbf{a}_t)$ . We seek to determine a collaborative plan that maximizes the  $J(\pi) \triangleq \mathbb{E}_{s_0,\infty \sim P, a_0,\infty \sim \pi} [\sum_{i=0}^{\infty} \gamma^i r_i]$ . For this paper, the goal is to coordinate the distribution of sub-tasks among agents so as to complete tasks more efficiently.

**CoELA** [43]. It contains five modules: 1) Perception, 2) Memory, 3) Communication, 4) Planning and 5) Execution. The Perception module leverages a trained Mask-RCNN [12] to segment RGB images and detect objects and obstacles, creating 3D point clouds from RGB-D data and extracting essential high-level information like key object states. The Memory module manages short-term and longterm memory, with short-term memory capturing immediate details such as semantic maps and agent states, and long-term memory storing persistent information like task descriptions. In the Communication module, CoELA retrieves relevant information from memory, transforming it into template-based textual descriptions for agents to aid in action selection. However, each agent plans independently, with no unified action coordination. In the Planning and Execution modules, CoELA retrieves agent-related information from memory, generates potential actions for the LLM to consider, and applies chain-of-thought [38] reasoning for more in-depth decision-making. Each agent then executes its chosen action accordingly.

# 4. Collaborative Tree Search for Multi-Agents

## 4.1. Overview of CoTS

Now we introduce an overview of our Collaborative Tree Search (CoTS), which is designed for multi-agent cooperation in embodied environments. As shown in Fig. 2, CoTS contains four key modules: 1) Perception Module, 2) Memory Module, 3) Collaborative Planning Module, and 4) Plan Parsing and Execution Module. This work primarily enhances CoELA in its planning by designing the Collaborative Planning Module. The other three modules are largely borrowed from CoELA and introduced in Sec. 3. Here we briefly introduce our enhancements to CoEAL below.

**Perception Module.** Unlike CoELA, we introduce distance as a key perceptual factor, leveraging fixed spatial layouts in real-world settings to enhance agents' room-based understanding and improve planning efficiency by using LLM to access the moving distance of agents.



Figure 3. Overview of the tree search in CoTS. Each node in the tree contains content generated by Alice and Bob. Alice generates the collaborative plans and messages to Bob, while Bob generates messages to Alice and rewards for the plans. These rewards are used for backpropagation to guide the tree search toward more promising collaborative plans. The figure also shows examples of three nodes, illustrating the stepwise optimization of collaborative planning and the dialogue process between agents.

**Memory Module.** Compared with CoELA, CoTS introduces "extended memory" (Fig. 2) to manage collaboration plans—information that needs to be retained for a moderate period and updated periodically. This is because collaboration plans must balance retention for a certain period and timely updates when they become unsuitable for the current embodied environment. This extended memory mimics the medium-term storage of the human memory system, bridging short-term recall and long-term consolidation.

**Collaborative Planning Module** is the core of CoTS, where MCTS is modified and employed to develop longterm strategic collaboration plans. Inspired by human problem-solving through iterative discussion and cooperation, it avoids error accumulation caused by a single path approach. This module represents the first integration of embodied agent collaboration with MCTS, enhancing agents' capability to tackle complex, long-term tasks through tree-based search and backtracking. Details of this module are provided in Sec. 4.2.

**Plan Parsing and Execution Module** translates collaborative plans into executable actions, enabling agents to complete tasks in a coordinated manner. See details in Sec. 4.3.

Below we will delve deeper into two core modules, Collaborative Planning Module and Plan Parsing and Execution Module. Additionally, we will highlight CoTS's integration of spatial awareness and extended memory to enhance overall functionality and efficiency.

### 4.2. Collaborative Planning Module

Our Collaborative Planning Module, central to our CoTS framework, is designed to search for long-term, efficient

and coherent collaborative plans for multi-agent tasks in dynamic environments. It comprises two main sub-modules: a collaborative tree search module for generating collaborative plans (Sec. 4.2.1) and a plan evaluation module to analyze when to update the plan (Sec. 4.2.2).

## 4.2.1. Monte Carlo Tree Search for Collaborative Plan

We adapt MCTS to convert multi-agent collaborative planning into a tree search, identifying an optimal pathway from multiple reasoning paths for cohesive strategies. As shown in Fig. 3, each node in this tree stores a proposed plan, agent feedback, and a performance reward. In this setup, one agent (like Alice) functions as the "planner" to propose plans, while other agents (like Bob) act as "critics". Alice drafts several initial collaborative plans or extends existing plan based on task goals and current task progress. Then Bob evaluates them, considering factors like task allocation and movement efficiency, to generate a reward for the current plan. Once feedback is provided, each plan's reward is propagated back through the collaboration tree, strengthening successful strategies and adjusting future planning pathways. Accordingly, they will select the most efficient collaborative plan for further reflections and refinements. As shown in Fig. 3, this process, including plan extension (drafting), evaluation, backpropagation, and selection, will repeat until agents explore all potential collaborative plans and select the most promising one. In the following, for these four steps, we will elaborate on each of them.

**Selection** Similar to traditional MCTS [4, 15], CoTS starts from the root node and traverses down the tree by selecting nodes with the highest UCT values. This process is repeated

until a leaf node is reached, and is illustrated in Fig. 3 where we mark the selected nodes with gold cups. For UCT of each node, it is computed as follows:

$$UCT(s_i) = V(s_i) + w\sqrt{\ln N(p_i)/N(s_i)}, \qquad (1)$$

Where  $N(s_i)$  is visiting number of node  $s_i$ ,  $V(s_i)$  is the reward of the subtree rooted at  $s_i$ , w is the exploration coefficient, and  $p_i$  is the parent node of  $s_i$ . After an episode, backpropagation updates each node's  $V(s_i)$  along the path:  $V(s_i) = \frac{V(s_{i-1}) \cdot (N(s_i) - 1) + r}{N(s_i)}$ , where  $V(s_{i-1})$  is the previous reward, and r is the reward used to update each  $V(s_i)$ . **Expansion.** Once a node is selected, we expand it by sampling n new discussion paths, each building on the plan (from "planner", Alice) and suggestions (from "critic", Bob) in the parent node. This generates n new child nodes, representing potential planning routes. The tree, along with its nodes, is stored in an external long-term memory to retain valuable information and facilitate consistent planning across episodes.

Evaluation. Each newly created child node receives a reward r to quantify the quality of the collaboration plan and to guide the search algorithm towards the most promising areas of the tree. Here CoTS incorporates a non-simulated LLM-based reward function to estimate the reward r of a plan in a node. In traditional MCTS, reward estimation requires agents to execute their actions for receiving environment feedback, like completing a subtask, and thus is highcost. So CoTS introduces an LLM-based reward function based on two components: (1) an allocation reward assessing coherence and efficiency, and (2) a cost reward evaluating the spatial movement cost of agents. To ensure consistency, we standardize the evaluation criteria to provide clear guidelines to minimize ambiguity. Each criterion is scored on a scale of 1 to 5, with the final reward normalized between 0 and 1. See examples in Appendix B.1.

**Backpropagation.** Each evaluation reward is propagated back along the path to update node rewards, with formulas ensuring that parent nodes integrate the feedback, guiding future selections. Specifically, the back-propagation updates node rewards  $N(s_i) = N(s_{i-1}) + 1$  and  $V(s_i) = \frac{V(s_{i-1})N(s_{i-1})+r}{N(s_i)}$  for each node  $s_0, s_1, \ldots, s_p$  along the path from the root (initial state  $s_0$ ) to leaf (terminal state  $s_p$ ). Accordingly these estimated rewards and visiting number are used in "Expansion" step for node selection, directing the search toward the most promising areas of the tree.

## 4.2.2. Collaborative Plan Evaluation and Adaptation

In this section, we will explain why we introduced the planning evaluation module (a sub-module in the collaborative planning module, as shown in Fig. 2) and its function. The collaborative plan provides high-level embodied agent action guidance, but as the task is executed and agents act, the embodied environment gradually changes (*e.g.* the agents' positions, the agents' progress, and the current status, etc.). Therefore, the collaborative plan needs to be updated based on the latest environmental information.

At this point, two scenarios arise: If the plan is updated too slowly or not updated at all, the current plan becomes outdated and cannot effectively guide the embodied agents in executing tasks. However, in a multi-agent system, instantly updating the plan with every progress can be inefficient and counterproductive. It may lead to excessive queries to the LLM and environments. Furthermore, frequent plan revisions may introduce conflicting strategies due to minor environmental shifts, potentially disrupting agent actions and lowering overall efficiency. This is illustrated in Fig. 4 (c): if the updated plan conflicts with the action that Bob is currently performing, it may interrupt Bob's action which even almost finishes the subtask.

To address this, we introduce a plan evaluation module within CoTS to continuously evaluate the viability of the current collaborative plan. When an agent makes progress, the module determines if the existing plan remains effective based on the status of task execution. If the evaluation metric meets a defined threshold, indicating that the plan is still viable, it is retained. Otherwise, CoTS will call the collaborative planning module to re-search and update the plan. By focusing on the feasibility and long-term stability of the plan rather than continuous recalibration, this module supports efficient, coherent collaboration, enabling agents to complete tasks reliably in a dynamic environment.

## **4.3. Plan Parsing and Execution Module**

To transform the collaborative plan generated in Sec. 4.2 into executable actions, CoTS incorporates an action parsing module and a plan decomposition module. Action parsing module bridges high-level planning with real-world actions, generating a set of feasible actions for each agent based on current memory. For example, if an agent's hands are free, it will list objects that can be grabbed; if the agent is in an unknown room, it will list movement options. This module ensures that all sub-plans are formatted into natural language and returned for subsequent process.

Plan decomposition module understands and breaks down high-level tasks from the collaboration plan into specific and prioritized actions. The agent interprets the task arrangement, selects actions from the parsed options that best advance the collaborative plan's goals, and finally translates each chosen action into real-world execution.

# 5. Experiments

**Benchmarks.** Following CoELA [43], we evaluate the performance of CoTS in two highly challenging embodiedworld multi-agent environments: (1) TDW-MAT (ThreeD World Multi-Agent Transport) [8] extends the ThreeD World Transport Challenge by simulating collaborative

Table 1. Quantitative results on TDW-MAT. Consistent with CoELA [43], TDW-MAT has two environment settings: with oracle perception and without oracle perception. The results marked with \* are from CoELA [43]. MAT uses central observation and oracle perception.  $91_{+9.64\%}$  means the relative improvement when comparing with CoELA is 9.64\%.

Transport Rate (%)	Bas	e Agent	RL Agent			LLM	Agent				
	RHD*	CoRHP*	D* MAT*	GPT 3.5			GPT 4				
	Cokin		CoELA	RoCo	CaPo	CoTS <sub>(ours)</sub>	CoELA*	RoCo	CaPo	CoTS <sub>(ours)</sub>	
			TDW	-MAT w/ 0	Oracle Percep	tion					
Stuff(†)	49	74	17	73	80+9.68%	$84_{+15.07\%}$	$81_{+10.96\%}$	83	$85_{+2.41\%}$	87+4.82%	91 <sub>+9.64%</sub>
Food(↑)	52	76	13	72	$74_{+2.78\%}$	$85_{+18.06\%}$	$83_{+15.28\%}$	87	$89_{+2.30\%}$	$90_{+3.45\%}$	$94_{+8.05\%}$
Average(↑)	50	75	15	73	$77_{+5.48\%}$	$84_{+15.07\%}$	$82_{+12.33\%}$	85	$87_{+2.35\%}$	$89_{+4.71\%}$	$93_{\mathbf{+9.41\%}}$
			TDW	MAT w/o	Oracle Percep	otion					
Stuff( <sup>†</sup> )	36	54	-	39	$47_{+20.51\%}$	$45_{+15.38\%}$	$50_{+28,21\%}$	61	$64_{+4.92\%}$	$64_{+4.92\%}$	$77_{+26.22\%}$
Food( <sup>†</sup> )	49	67	-	67	$64_{-4.48\%}$	$70_{+4.48\%}$	$67_{\pm 0.00\%}$	82	$80_{-2.44\%}$	$85_{+3.65\%}$	88+7.32%
Average( <sup>†</sup> )	43	61	-	52	$55_{+5.77\%}$	$57_{+9.62\%}$	$\mathbf{59_{+13.46\%}}$	71	$72_{+1.41\%}$	$74_{+4.23\%}$	$83_{+16.90\%}$

Table 2. Quantitative results on C-WAH. LLM-based agents are driven by GPT 4. Results marked with \* are from CoELA [43].

Average Step		Visual $Obs(\downarrow)$	Symbolic $Obs(\downarrow)$	
Base Agent	MHP* CoMHP*	141 103	111 75	
LLM Agent	CoELA* RoCo CaPo <i>CoTS</i> (ours)	$92 \\ 89_{+3.26\%} \\ 83_{+9.78\%} \\ 81_{+11.96\%}$	$57 \\ 57_{+0.00\%} \\ 51_{+10.53\%} \\ 49_{+14.04\%}$	

transportation tasks in a physics-based household environment. We select 24 tasks, split evenly between food and stuff transportation. (2) C-WAH (Communicative Watch-And-Help) [29] extends the Watch-And-Help challenge within the VirtualHome-Social platform [28]. It features five common household scenarios. Our test set includes 10 tasks, with two tasks per scenario. The objective is to complete all sub-goals, prioritizing faster completion.

**Evaluation Metrics.** On TDW-MAT, transport rate is evaluated by the percentage of target objects successfully delivered to their designated locations within a 3,000-frame timeframe. For C-WAH, efficiency is measured by the number of time steps needed to complete all assigned tasks, where fewer steps indicate higher efficiency.

**Baseline Methods.** The baselines include three types of embodied agents: (1) Base Agent: RHP [8] uses heuristic rules for navigation and A\* for path planning. MHP [18] employs Monte Carlo Tree Search (MCTS) for high-level planning coupled with regression planning for task execution (2) RL-Based Agent: MAT [39] is a multi-agent reinforcement learning framework, featuring a centralized decision transformer that synthesizes actions from shared observations. (3) LLM-Based Agent: CoELA [43] is a leading collaboration framework for embodied agents that integrates the reasoning capabilities of LLMs. It facilitates communication and interaction between agents through natural language. RoCo [24] and CaPo [22] are multi-agent collaboration methods that enhance collaboration by generating coordinated plans through structured agent dialogues. **Setup.** We conduct all experiments on an Intel Core i9-10980XE CPU and an NVIDIA GeForce RTX 3090 GPU. We utilize both GPT-4 and GPT-3.5-Turbo [1], configuring them with a temperature setting of 0.7 and top-1 sampling to produce outputs up to 512 tokens.

## 5.1. Main Results

**CoTS Outperforms the Strong Baselines.** Following the experimental frameworks of CoELA [43] and RoCo [24], we configure two-agent cooperation, excluding RHP. As demonstrated in Table 1, CoTS achieves best performance on the TDW-MAT task. Specifically, when powered by GPT-4 within the Oracle Perception environment, CoTS surpasses CoELA, achieving a 9.41% average relative improvement in task completion, and a 16.9% improvement in the absence of Oracle Perception. The primary strength of CoTS is its ability to generate superior long-term strategies through collaborative tree search.

Conversely, CoELA focuses on short-term actions derived from immediate information, which limits its strategic depth and adaptability to future scenarios and environmental shifts. RoCo and CaPo are similar. Although they are capable of developing plans, they rely on a single-branch discussion method, which may result in suboptimal or inaccurate strategies. Moreover, Their frequent plan revisions can interfere with agent coordination. Therefore, while RoCo and CaPo surpass CoELA, they are inferior to CoTS. When driven by GPT-3.5, CaPo slightly outperforms CoTS on some metrics. The main reason for this is that GPT-3.5 has weaker evaluation and scoring capabilities. However, when powered by GPT-4, CoTS significantly outperforms CaPo. Our findings underscore that multi-agent collaboration generally surpasses single-agent strategies, exemplified by RHP. The performance of reinforcement learning-based MAT is poor, mainly because TDW-MAT is too difficult for it with long time domain and sparse reward signals.

The performance of LLM-based agents primarily de-

# (a) CoELA leads to short-sighted action plans that ignore precious action steps

**Background**: Alice is in living room holding a container. Bob finds two target objects in living room<1000>.



**CoELA's plan** : Alice goes to the living room to help Bob, because Alice can take these objects.

**Our plan** : Alice explores the current room, find objects, then transport them to the bedroom. Then go to the living room <1000>, no actions are wasted.

## (b) RoCo's single-chain dialogue leads to suboptimal collaboration plan

**Background**: Alice and Bob appear in the same living room and now need to explore some rooms to find objects.



**RoCo's plan** : Alice explore the current room. Bob explore the current room.

**Our plan** : Alice goes to the nearest room, Bedroom <2000>, while Bob explores the current room. Exploring rooms separately would improve efficiency.

## (c) Frequent plan updates may hinder completion of current actions

**Background**: Bob was transporting objects to the bed, Alice found a bread in the living room<6000>



**RoCo's Plan:** Bob goes to the living room <6000> to help Alice grasp the bread.

**Our plan**: The current plan is still valid. Bob is close to the bedroom. He can first deliver the objects in his hand to the bed, and then help Alice.

Figure 4. Some interesting example cases demonstrate that our CoTS achieves better collaboration than RoCo and CoELA. These scenes are taken from the experiments in TDW-MAT.



There seems to be no plan. Both Alice and Bob are not holding anything. Bob in bedroom and has found an container, while Alice is in the. A collaboration plan needs to be updated. **Reward: 1** 



Alice have found an apple in the fridge and Bob is holding a cupcake. Current progress is consistent with the collaborative plan and future actions are outlined in the plan. There is no need to update the plan. **Reward: 4** 

Figure 5. Plan Evaluation and Update. The plan evaluation module gives a lower reward when it determines that the current plan needs to be updated, and a higher reward when it believes the current plan is still valid. The reward range is from 1 to 5.

pends on their reasoning abilities and the depth of embedded world knowledge. Agents powered by GPT-4, as demonstrated by CoTS, significantly surpass those based on GPT-3.5, highlighting the advancements in LLM technologies and their impact on agent proficiency. As shown in Table 2, CoTS achieves superior efficiency, improving by 11.96% and 14.04% over CoELA in Visual Obs and Symbolic Obs settings, respectively. These results underscore CoTS's robust adaptability across various embodied environments and its potential for real-world applications.

# 5.2. Qualitative Analysis

**CoTS Can Develop Long-term Strategic Plans.** Unlike CoELA, which is adept at responding to immediate tasks



Figure 6. Comparison of transportation rates for GPT-4-based CoTS and CoELA at different time steps on TDW-MAT.

but falters in sustained coordination, CoTS excels in longterm strategic planning. As illustrated in Fig. 4 (a), when Bob discovers a new target that requires assistance, Alice, under CoTS's direction, continues to thoroughly explore her current room before moving the item to the bed. This systematic approach not only supports Bob but also adheres to an overarching collaborative strategy, optimizing both immediate and future task alignments.

**CoTS Generates More Optimal Collaboration Plans.** Unlike RoCo to generate plans sequentially through a single-chain dialogue, CoTS makes more optimized collaboration plans. Due to the randomness of LLMs and the embodied environment's complexity, RoCo's plans may sometimes be suboptimal. For example, in Fig. 4 (b), RoCo instructs both Alice and Bob to explore their respective rooms. However, they are currently in the same room, making joint exploration redundant. CoTS effectively identifies this and assigns Alice to explore the nearest room instead.



Figure 7. Comparison of transportation rates with different branching factors. More branching factors means more new nodes are generated each time in the expansion phase.

**CoTS Ensures Smooth Completion of Agent Actions.** In RoCo, new progress often prompt discussions and lead to new plans that can override existing ones. As illustrated in Fig. 4 (c), Bob, tasked with transporting items to the bed, must halt his activity to help Alice following a revised plan. Conversely, CoTS incorporates a plan evaluation module that validates the feasibility of plans before any adjustments are made. In Fig. 4 (c), CoTS allows Bob to finish transporting before next action. This approach avoids chaotic agent actions and ensures timely plan updates when necessary.

**CoTS Dynamically Manages Collaborative Plans.** As shown in Fig. 5, the assessment module continuously evaluates the viability of existing plans. It assigns a lower reward (e.g. 1) if collaboration necessitates devising a new plan or modifying an existing one, prompting agents to adapt their strategies accordingly. Conversely, when the current progress aligns with planned tasks and their distribution among agents is deemed efficient, the module rewards this alignment with a higher value (e.g. 4). This incentivizes agents to persist with the established collaborative plan, avoiding unnecessary replanning.

## 5.3. Ablation Study and Additional Analysis

The Core Modules of CoTS Have Important Impacts. The Table 3 shows the ablation results of CoTS. To examine the impact of tree search, we modify it to a singlebranch structure, where agents can only communicate along one path. The experimental results indicate a 5.38% relative decrease in the average transportation rate, mainly because the limited communication does not consistently yield correct conclusions and diverges from typical human discussion patterns. Furthermore, eliminating the reward function and plan update evaluation components from CoTS leads to variable performance declines, highlighting the critical roles these elements play in the system's efficacy.

Analysis of Agent Collaboration Progress at Different Time Steps. We analyze the transportation rates of CoELA and CoTS on TDW-MAT at various time steps, as depicted in Fig. 6. An interesting observation emerges at 500 steps, where CoTS exhibits a lower performance compared to

Table 3. Effects of the reward function, collaboration tree search, and plan evaluation module in GPT-4-driven CoTS on TDW-MAT.

Transport Rate	Stuff(†)	$Food(\uparrow)$	Average(†)
CoTS w/o Reward	$87_{-4.40\%}$	$90_{-4.26\%}$	89-4.30%
CoTS w/o MCTS Plan	$85_{-6.59\%}$	$91_{-3.19\%}$	$88_{-5.38\%}$
CoTS w/o E&U	$88_{-3.30\%}$	$92_{-2.13\%}$	$90_{-3.23\%}$
CoTS	91	94	93

CoELA, attributed to CoTS's strategy of prioritizing item collection early to enhance transport capacity. However, this initial disadvantage is counterbalanced by a significant improvement at 2000 steps, where CoTS markedly surpasses CoELA. This shift underscores CoTS's effectiveness in optimizing agent collaboration for task execution.

**Impact of Different Branching Factors on Results.** During the expansion phase of MCTS, we define the number of new candidate nodes generated at each step as the "branching factors." As shown in Fig. 7, increasing the number of branching factors expands the selection space, leading to a better plan. Adjusting the branching factor allows for a balance between performance and efficiency.

Besides, the human-agent collaboration and the cost of CoTS are discussed in Appendix A.1 and Appendix A.2, respectively. CoTS also demonstrates significant advantages.

# 6. Conclusions

This paper introduces CoTS, a collaborative framework that leverages large language models with a modified Monte Carlo tree search to enhance multi-agent task planning in complex and dynamic environments. By enabling agents to evaluate multiple potential plans and refine strategies based on progress, CoTS significantly advances over prior methods like CoELA and RoCo. The addition of a plan feasibility module ensures smooth task execution by adapting plans only when necessary. Experimental results on multiobjective tasks confirm CoTS's superior performance.

**Limitation.** CoTS relies on LLMs' reasoning abilities. We find that GPT-4 surpasses GPT-3.5-turbo in instructionfollowing and response quality, while GPT-3.5-turbo can be verbose and occasionally inaccurate. Despite this, even for GPT-3.5, CoTS improves agent collaboration through collaborative planning search in the collaboration tree.

# 7. Acknowledgments

This work is supported by National Natural Science Foundation of China (U2133202, 52305570), China Postdoctoral Science Foundation (2022M720955), and Outstanding Doctoral Dissertation Funding Project of Heilongjiang Province (LJYXL2022-011). Lizheng Zu also acknowledges the support of the China Scholarship Council program (202406120111).

## References

- [1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023. 1, 2, 6
- [2] David Brandfonbrener, Simon Henniger, Sibi Raja, Tarun Prasad, Chloe R Loughridge, Federico Cassano, Sabrina Ruixin Hu, Jianang Yang, William E Byrd, Robert Zinkov, et al. Vermcts: Synthesizing multi-step programs using a verifier, a large language model, and tree search. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*, 2024. 3
- [3] Tom B Brown. Language models are few-shot learners. arXiv preprint arXiv:2005.14165, 2020. 1
- [4] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in* games, 4(1):1–43, 2012. 4, 3
- [5] Abhinav Dahiya, Alexander M Aroyo, Kerstin Dautenhahn, and Stephen L Smith. A survey of multi-agent human–robot interaction systems. *Robotics and Autonomous Systems*, 161: 104335, 2023. 1
- [6] Norman Di Palo, Arunkumar Byravan, Leonard Hasenclever, Markus Wulfmeier, Nicolas Heess, and Martin Riedmiller. Towards a unified agent with foundation models. arXiv preprint arXiv:2307.09668, 2023. 2
- [7] Amal Feriani and Ekram Hossain. Single and multi-agent deep reinforcement learning for ai-enabled wireless networks: A tutorial. *IEEE Communications Surveys & Tutorials*, 23(2):1226–1252, 2021. 1
- [8] Chuang Gan, Siyuan Zhou, Jeremy Schwartz, Seth Alter, Abhishek Bhandwaldar, Dan Gutfreund, Daniel LK Yamins, James J DiCarlo, Josh McDermott, Antonio Torralba, et al. The threedworld transport challenge: A visually guided taskand-motion planning benchmark towards physically realistic embodied ai. In 2022 International conference on robotics and automation (ICRA), pages 8847–8854. IEEE, 2022. 2, 5, 6
- [9] Ran Gong, Qiuyuan Huang, Xiaojian Ma, Yusuke Noda, Zane Durante, Zilong Zheng, Demetri Terzopoulos, Li Fei-Fei, Jianfeng Gao, and Hoi Vo. Mindagent: Emergent gaming interaction. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3154–3183, 2024. 3
- [10] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. arXiv preprint arXiv:2402.01680, 2024. 1
- [11] Xudong Guo, Kaixuan Huang, Jiale Liu, Wenhui Fan, Natalia Vélez, Qingyun Wu, Huazheng Wang, Thomas L Griffiths, and Mengdi Wang. Embodied llm agents learn to cooperate in organized teams. arXiv preprint arXiv:2403.12482, 2024. 2

- [12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 3
- [13] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. Metagpt: Meta programming for multi-agent collaborative framework. arXiv preprint arXiv:2308.00352, 2023. 2
- [14] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR, 2022. 1
- [15] Steven James, George Konidaris, and Benjamin Rosman. An analysis of monte carlo tree search. In *Proceedings of the* AAAI Conference on Artificial Intelligence, 2017. 4
- [16] Feibo Jiang, Yubo Peng, Li Dong, Kezhi Wang, Kun Yang, Cunhua Pan, Dusit Niyato, and Octavia A Dobre. Large language model enhanced multi-agent systems for 6g communications. *IEEE Wireless Communications*, 2024. 2
- [17] Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, et al. Graph chain-of-thought: Augmenting large language models by reasoning on graphs. arXiv preprint arXiv:2404.07103, 2024. 3
- [18] Richard E Korf. Planning as search: A quantitative approach. *Artificial intelligence*, 33(1):65–88, 1987. 6, 5
- [19] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36: 51991–52008, 2023. 2
- [20] Xin Li. Positioning is all you need. *arXiv preprint arXiv:2404.01183*, 2024. 2
- [21] Iou-Jen Liu, Unnat Jain, Raymond A Yeh, and Alexander Schwing. Cooperative exploration for multi-agent deep reinforcement learning. In *International conference on machine learning*, pages 6826–6836. PMLR, 2021. 1
- [22] Jie Liu, Pan Zhou, Yingjun Du, Ah-Hwee Tan, Cees GM Snoek, Jan-Jakob Sonke, and Efstratios Gavves. Capo: Cooperative plan optimization for efficient embodied multiagent cooperation. arXiv preprint arXiv:2411.04679, 2024. 2, 6, 5
- [23] Yang Liu, Weixing Chen, Yongjie Bai, Jingzhou Luo, Xinshuai Song, Kaixuan Jiang, Zhida Li, Ganlong Zhao, Junyi Lin, Guanbin Li, et al. Aligning cyber space with physical world: A comprehensive survey on embodied ai. arXiv preprint arXiv:2407.06886, 2024. 1
- [24] Zhao Mandi, Shreeya Jain, and Shuran Song. Roco: Dialectic multi-robot collaboration with large language models. In 2024 IEEE International Conference on Robotics and Automation (ICRA), pages 286–299. IEEE, 2024. 1, 2, 3, 6, 5
- [25] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. arXiv preprint arXiv:2112.09332, 2021. 2

- [26] Bo Ni and Markus J Buehler. Mechagents: Large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge. *Extreme Mechanics Letters*, 67:102131, 2024. 2
- [27] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. Advances in neural information processing systems, 35:27730–27744, 2022. 2
- [28] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8494–8502, 2018. 6
- [29] Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Yuan-Hong Liao, Joshua B Tenenbaum, Sanja Fidler, and Antonio Torralba. Watch-and-help: A challenge for social perception and human-ai collaboration. arXiv preprint arXiv:2010.09890, 2020. 6
- [30] Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, et al. Habitat 3.0: A co-habitat for humans, avatars and robots. arXiv preprint arXiv:2310.13724, 2023. 2
- [31] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, 2024. 2
- [32] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. Advances in Neural Information Processing Systems, 36, 2024. 2
- [33] Paul E Smaldino, Cody Moser, Alejandro Perez Velilla, and Mikkel Werling. Maintaining transient diversity is a general principle for improving collective problem solving. *Perspectives on Psychological Science*, 19(2):454–464, 2024. 2
- [34] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009, 2023. 1
- [35] Wiebe Van der Hoek and Michael Wooldridge. Multi-agent systems. *Foundations of Artificial Intelligence*, 3:887–928, 2008. 2
- [36] Tai Wang, Xiaohan Mao, Chenming Zhu, Runsen Xu, Ruiyuan Lyu, Peisen Li, Xiao Chen, Wenwei Zhang, Kai Chen, Tianfan Xue, et al. Embodiedscan: A holistic multimodal 3d perception suite towards embodied ai. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19757–19767, 2024. 1
- [37] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of

large language models. *arXiv preprint arXiv:2206.07682*, 2022. 2

- [38] 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. Advances in neural information processing systems, 35:24824–24837, 2022. 1, 2, 3
- [39] Muning Wen, Jakub Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. *Advances in Neural Information Processing Systems*, 35:16509–16521, 2022. 1, 6, 5
- [40] 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. Advances in Neural Information Processing Systems, 36, 2024. 1, 2, 3
- [41] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. Co-navgpt: Multi-robot cooperative visual semantic navigation using large language models. arXiv preprint arXiv:2310.07937, 2023. 2
- [42] Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b: A technical report. arXiv preprint arXiv:2406.07394, 2024. 3
- [43] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. arXiv preprint arXiv:2307.02485, 2023. 1, 2, 3, 5, 6
- [44] Zhonghan Zhao, Ke Ma, Wenhao Chai, Xuan Wang, Kewei Chen, Dongxu Guo, Yanting Zhang, Hongwei Wang, and Gaoang Wang. Do we really need a complex agent system? distill embodied agent into a single model. arXiv preprint arXiv:2404.04619, 2024. 2
- [45] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. arXiv preprint arXiv:2310.04406, 2023. 2

# **Collaborative Tree Search for Enhancing Embodied Multi-Agent Collaboration**

Supplementary Material

The supplementary material is organized as follows:

- Appendix A: Additional Experiments
- Appendix B: Additional Details about CoTS
- Appendix C: Embodied Environments and Baselines
- Appendix D: Planning and Evaluating Examples
- Appendix E: Prompt Templates

# **A. Additional Experiments**

## A.1. Collaborating with Humans

Following CoELA, we evaluate human-agent cooperation in C-WAH (symbolic observations). Since CoTS requires one collaborator to act as a planner, this role can be challenging and time-consuming for humans in complex embodied environments. However, LLMs excel in such tasks. Therefore, in this experiment, Alice (the Planner) is controlled by GPT-4, while Bob (the Reflector and Evaluator) is a human. This setup leverages the reasoning capabilities of the language model alongside human experience and precise evaluation.

To assess efficiency, we measure the average number of steps taken. Additionally, eight participants evaluate their teammate based on three key criteria—communication effectiveness, helpfulness, and trust—using a 7-point Likert scale. The results are summarized in the Table 4.

Table 4. Human experiments results.

Method	Ave steps	Comm Effectiveness	Helpfulness	Trust
CoELA	48	5.7/7	5.4/7	6.0/7
CoTS	43	<b>6.3</b> / 7	<b>6.1</b> / 7	<b>6.4</b> /7

**Communication Effectiveness** is evaluated based on the following aspects: 1) Clarity of Information Transmission – Whether the provided prompts or explanations are easy to understand, and whether the language is accurate, concise, and well-organized. 2) Understanding User Needs – Whether the Agent can correctly interpret user instructions, questions, or intentions and provide appropriate responses. 3) Overall Interaction Experience – Whether the conversation flows naturally and whether the communication frequency is reasonable. From the experimental results, CoTS demonstrates better "communication effectiveness" mainly because it establishes a detailed long-term collaboration plan, making communication more logical and persuasive while reducing unnecessary exchanges and improving overall communication efficiency.

Helpfulness measures whether the Agent can provide targeted, highly feasible, and productive dialogue content, effectively reducing the user's cognitive and reasoning burden, thereby making task completion easier. This metric focuses on the Agent's actual contribution to user goals and whether users perceive value from the interaction. According to the experimental results, CoTS is primarily responsible for integrating large amounts of information and formulating complex plans, requiring humans only to provide feedback and evaluations, which significantly reduces the user's workload. Additionally, CoTS exhibits higher task execution efficiency, making it easier for users to obtain assistance and improve their productivity.

**Trust** assesses the degree to which participants trust the Agent's response, including whether the answers provided are accurate and reliable, whether the decision-making process is coherent and well-founded, and whether the information is professional and specific. During the collaboration process, users demonstrated a higher level of trust in CoTS, mainly due to its proactive incorporation of human input during planning and its collaborative approach in multi-round discussions to develop long-term plans. Moreover, because CoTS has well-defined functions for each, its responses are more professional and targeted, making it easier for humans to accept and trust its decisions.

Overall, CoTS outperforms CoELA in communication effectiveness, helpfulness, and trust. Experimental participants generally found CoTS to be more fluid in communication, more helpful, and more trustworthy. The efficiency of experiments conducted with CoTS was also significantly higher than with CoELA, further demonstrating CoTS's potential in the field of human-AI collaboration.

## A.2. Cost Discussion

The primary costs for embodied agents come from communication and movement, with movement costs often being higher, as current robots lack the agility of humans, especially in complex and large-scale environments like urban areas. However, through communication, CoTS enables more effective collaborative planning and reduces errors that could lead to critical outcomes in sensitive applications like healthcare or surgery. While this incurs added communication costs (mainly from LLM inference), these are essential to enhance system intelligence, and as LLM technology advances, these costs are likely to diminish. Besides, enhanced planning in CoTS minimizes redundant or erroneous actions by agents, thereby reducing overall task completion time.

Regarding communication cost, we compare CoTS with the traditional tree search method ToT (DFS) on C-WAH (symbolic obs), and the results are shown in Table 5. The communication cost represents the average number of char-

Table 5. Cost of each plan update.

Method	Ave steps	Communication Cost	Time Cost
CoT	57	-	-
ТоТ	52	10568 characters	32.4s
CoTS	49	7923 characters	26.8s

acters consumed per round of plan updates. Results show that 1) ToT and CoTS outperform CoT because of their plan search and exploration, but are more costly. 2) CoTS surpasses ToT by 5.8% relative improvement and a big efficiency improvement including a 17% reduction in time cost. 3) Unlike CoTS, ToT lacks selection and backpropagation steps, weakening its value evaluation ability, increasing exploration costs and consuming more memory due to maintaining more nodes. In contrast, CoTS ensures precise long-term planning, minimizing errors and redundant actions, which ultimately reduces overall costs.

# **B.** Additional Details about CoTS

## **B.1. Plan Reward Based on LLMs**

In the Collaboration Tree Search Module and Plan Evaluation Module of CoTS, assessments based on large language models (LLMs) are required to quantify rewards for collaboration plans. (1) In the tree search, the evaluation aims to assign a reward to each newly generated node, guiding the tree toward exploring more promising directions. (2) The evaluation module assesses the viability of current plans based on the progress and actions of embodied agents. If a plan remains viable, it is retained; otherwise, a new action plan is sought.

Utilizing LLMs to assess collaborative plans leverages their formidable reasoning capabilities and comprehensive world knowledge. In the context of Collaborative Tree Search, LLM-based evaluation bypasses the need for intricate simulations by directly providing reward values within the Monte Carlo tree, offering distinct advantages for embodied environments. However, the straightforward application of LLMs for evaluating collaboration plans often yields outcomes that are vague and exhibit a significant lack of consistency with human subjective judgments, marked by considerable randomness.

Specifically, we enhance our prompts with labels to refine the assessment process. For example, here we present three different prompt templates in the planning evaluation module. The label-enhanced version is applied to the plan evaluation module in CoTS (as shown in Fig. 8). If the final reward exceeds the set threshold (*e.g.*, 3), there is no need to update the plan.

**Basic Version.** Please analyze whether the current collaboration plan requires immediate adjustments. If the current plan is satisfactory, respond with "satisfied"; otherwise, respond with "unsatisfied."

**Reward Version.** Please analyze whether the current collaboration plan requires immediate adjustments and assign a reward. The higher the reward, the better the current plan and the less it needs updating. The reward should be an integer between 1 and 5.

**Our Label Enhanced Version in Plan Evaluation.** Please analyze whether the current collaboration plan requires immediate adjustments. Your response should include both your reasoning and a reward. The reward should be assigned based on the following criteria: Plan\_Reward: [the reward between 0 and 5: 1, The current plan is completely unreasonable and needs to be revised 2, The current plan requires significant adjustments 3, The current plan needs improvement, its core structure is sound, though details are unclear 4, The current plan requires only minor adjustments 5, The current plan is fully reasonable and needs no changes. Only output the reward.]

The evaluation of plans in collaborative tree search follows a similar pattern. It assesses the rationality of collaborative plans by considering the allocation of agent actions and whether the action distances between agents are accounted for. After evaluation, the rewards from these two aspects are aggregated and normalized to a range between 0 and 1, serving as the final reward for the node.

**Dis\_Reward.** [the reward between 0 and 5: 1, No consideration of distance 2, Minimal consideration of distance, overlooking key factors 3, Distance is considered but not entirely accurate 4, Distance is sufficiently considered, with only minor oversights 5, Comprehensive and accurate consideration of distance. Only output the.]

**Task\_Reward.** [the reward between 0 and 5: 1, No consideration of work distribution between Alice and Bob 2, Minimal consideration of work distribution, leading to unreasonable allocation 3, Work distribution is considered but not entirely accurate 4, Work distribution is reasonable, with only minor oversights 5, Work distribution is highly effective, making full use of Alice and Bob's abilities. Only output the reward.]

## **B.2.** Monte Carlo Tree Search

MCTS is a technique for identifying optimal decisions within a defined decision space by randomly sampling and constructing a search tree based on these outcomes. It has profoundly influenced artificial intelligence (AI), particularly in game strategies and planning challenges. Monte Carlo trees are constructed incrementally and asymmetrically, with each node in the tree representing a specific state. During each iteration of the algorithm, the child state with the highest UCT (Upper Confidence Bound for Trees) value is selected for further expansion, balancing exploration (searching under-sampled areas) and exploitation (focusing on promising areas). Upon the conclusion of



**Input**: [Goal] [Current plan] [Dialogue history] [Previous action] [Progress desc] Please help me analyze whether the current collaboration plan needs to be adjusted immediately. Your response needs to include your thought process and rewarding.

[the reward between 0 and 5: 1, The current plan is completely unreasonable and needs to be revised 2, The current plan requires significant adjustments 3, The current plan needs improvement, its core structure is sound, though details are unclear 4, The current plan requires only minor adjustments 5, The current plan is fully reasonable and needs no changes. Only output the reward.]

LLM Output: 3

Figure 8. An example of using LLMs to assess whether the collaboration plan need to be updated.



Figure 9. Monte Carlo Tree Search [4].

an episode, backpropagation is performed, as illustrated in Fig. 9. MCTS consists of four essential stages:

(1) Selection: Starting from the root node, the algorithm identifies the most promising node according to the tree policy, continuing this process until a leaf node is reached.

(2) Expansion: At the selected leaf node, if it is not a terminal state, new child nodes are added to represent potential future states.

(3) Simulation: From the newly added nodes, the algorithm conducts random simulations until reaching a terminal state, then evaluates the node's potential based on the outcomes.

(4) Backpropagation: Simulation results are propagated back to the root, updating the scores of all visited nodes along the way.

The effectiveness of MCTS depends heavily on its ability to balance exploring unknown areas with exploiting known information, making it particularly valuable in domains requiring precise and adaptive decision-making.

# C. Supplementary Introduction to Embodied Environments and Contrastive Methods

## **C.1. Embodied Environments.**

Following CoELA [43], this paper considers two extremely challenging embodied-world multi-agent environments to test the performance of EMCT, including a subset of 24 tasks in TDW-MAT and a subset of 10 tasks in C-WAH.

1) **TDW-MAT** (**ThreeD World Multi-Agent Transport**), an extension of the ThreeD World Transport Challenge, TDW-MAT simulates collaborative transportation tasks in a physics-enabled household environment, built on the TDW platform. In this environment, agents must explore a house to locate dispersed objects, pick them up, and deliver them to designated locations. Each agent can carry only one item per hand, but they can strategically use containers to enhance transportation efficiency.

This study selects 24 representative subtasks, evenly split between food transportation (12 tasks) and item transportation (12 tasks). Each subtask environment contains 10 target objects and 2–5 containers, with a maximum duration of 3,000 action frames. Agents are tasked with transporting as many target objects as possible to the goal position, utilizing containers as tools. While a container can hold up to three objects, an agent without a container is limited to carrying two objects at a time.

Agents retain access to their ego-centric visual observations and action spaces throughout the tasks. Communication among agents, mediated by the LLM, occurs exclusively through natural language, leveraging the LLM's advanced reasoning and comprehension capabilities.

The food-transporting task involves six types of targets (apple, banana, orange, bread, loaf bread, and burger) and three types of containers (bowl, plate, and tea tray). In contrast, the object-transporting task includes six different targets (calculator, mouse, pen, lighter, purse, and iPhone) with three container types (plastic basket, wooden basket, and wicker basket). In each task, there are 10 target objects and 2 to 5 containers in total. Objects are distributed across four types of rooms—living room, office, kitchen, and bedroom—arranged in alignment with common-sense placement.

**Observation Space** The embodied agent primarily receives an egocentric RGB image and a depth image, along with several auxiliary observations. The observation space details are as follows:

- **RGB Image**: An egocentric image captured by a forward-facing camera, with a resolution of  $512 \times 512$  and a 90-degree field of view.
- **Depth Image**: A depth map captured using the same camera parameters as the RGB image.
- Oracle Perception (optional): A color-mapped image where each object ID corresponds to a unique color, using the same intrinsic camera parameters as the RGB image.
- Agent Position and Rotation: The agent's position and orientation within the simulated environment.
- Held Objects: Information about objects currently in the agent's possession.
- **Opponent Held Objects**: Enumerates the objects held by another agent, provided that this agent is visible.
- Messages: Communications exchanged among agents.

Action Space In TDW-MAT, agents can perform seven types of actions for environmental interaction or inter-agent communication. Each action spans multiple frames, with specifics listed below:

- Move Forward: Advances the agent by 0.5 meters.
- Turn Left: Rotates the agent 15 degrees to the left.
- Turn Right: Rotates the agent 15 degrees to the right.
- **Grasp**: Allows the agent to grasp an object if it is within close proximity. The object can be a target or container.
- **Put In**: Enables the agent to place a target object into a container, provided the agent is holding the target and container simultaneously.
- Drop: Releases any objects held by the agent.
- Send Message: Transmits a message to other agents, with a limit of 500 characters per frame.

We compile a comprehensive Action List encompassing all available actions relevant to the current state, providing LLMs with a structured framework for action selection. These selectable actions can be conveniently executed by embodied agents. For example, the TDW-MAT (Three-Dimensional World Multi-Agent Task) includes the following sub-plans:

- · explore current room
- go to the xx room
- go grasp the xxx (target object or container)
- put holding objects into the holding container
- transport holding objects to the bed

2) C-WAH (Communicative Watch-And-Help) builds

upon the Watch-And-Help challenge and is implemented on the VirtualHome-Social multi-agent simulation platform, introducing inter-agent communication. The environment encompasses five common household scenarios: afternoon tea preparation, dish washing, meal preparation, grocery organization, and table setting. Each task comprises multiple subgoals defined by predicates, such as "ON/IN(x,y)" denoting the placement of object x on/in location y. Detailed task descriptions and subgoal listings are provided in Table 6. Our test set consists of 10 tasks, with two tasks selected from each of the five scenario types. The task goal is to meet all given sub-goals within 250 time steps, the faster the better.

C-WAH encompasses a diverse range of household tasks, each with specific objectives: preparing afternoon tea, dishwashing, meal preparation, grocery storing, and dinner table arrangement.

**Objective.** Agents are required to complete all specified subgoals within a 250-timestep limit. Tasks consist of three to five subgoals, challenging agents to demonstrate effective strategic planning and inter-agent communication.

**Observation Space.** CWAH encompasses two modalities: *Symbolic Observation* and *Visual Observation*. In the *Symbolic Observation* mode, aligned with the setup in the foundational Watch-And-Help challenge, an agent accesses extensive object-related data within its vicinity, encompassing attributes such as location, status, name, and relational context. In contrast, *Visual Observation* permits agents to capture egocentric RGB and depth visuals, alongside supplementary observational data. Specifics of the observation space are as follows:

- **RGB Image**: Captured through a forward-oriented camera, presenting a resolution of  $256 \times 512$  and a 60-degree viewing angle.
- **Depth Image**: Utilizes the same intrinsic parameters as the RGB camera to provide depth cues.
- **Oracle Perception**: This visualization assigns unique colors to each object identifier, consistent with the RGB camera's intrinsic settings.
- Agent Position: Reflects the agent's location within the simulated environment.
- Messages: Includes all communications disseminated by the agents.

Action Space. Expanding on the established Watch-And-Help Challenge, the action repertoire includes a novel messaging capability. Enumerated actions are:

- Walk Towards: Navigate towards an object or a different room.
- Turn Left: Execute a 30-degree left turn.
- Turn Right: Execute a 30-degree right turn.
- **Grasp**: Ability to grasp an object, contingent on proximity.
- Open: Capability to open a nearby closed container.

Task Name	Predicate Set	
Prepare afternoon tea	ON(cupcake,coffeetable), ON(pudding,coffeetable),	
	ON(apple,coffeetable), ON(juice,coffeetable), ON(wine,coffeetable)	
Wash dishes	IN(plate,dishwasher), IN(fork,dishwasher)	
Prepare a meal	ON(coffeepot,dinnertable), ON(cupcake,dinnertable),	
	ON(pancake,dinnertable), ON(pudding,dinnertable),	
	ON(apple,dinnertable), ON(juice,dinnertable), ON(wine,dinnertable)	
Put groceries	IN(cupcake,fridge), IN(pancake,fridge),	
	IN(pudding,fridge), IN(apple,fridge), IN(juice,fridge),	
	IN(wine,fridge)	
Set up a dinner table	ON(plate,dinnertable), ON(fork,dinnertable)	

Table 6. Task description in C-WAH. The tasks are divided into five types, each containing several predicates.

- Close: Ability to close an open container when in proximity.
- **Put**: Place objects into an open container or onto a surface if nearby.
- Send Message: Transmit a message to peers, limited to 500 characters per instance.

## C.2. Baselines

Here we introduce in detail the baseline methods included in the experiment:

1) RHP (Rule-based Hierarchical Planner) [8], which builds upon the existing robust baseline from the ThreeD-World traffic challenge. RHP introduces a Rule-Based Hierarchical Planning framework enhanced by a Frontier Exploration technique. It consists of a rule-driven high-level planner that selects among multiple strategic options based on predefined rules. Navigation is handled by an A\*-based pathfinding planner, utilizing both occupancy and semantic maps that are continuously updated based on visual inputs.

2) MHP (MCTS-based Hierarchical Planner) [18], which represents the strongest baseline in the Watch-and-Help Challenge. It features a Hierarchical Planner combining a high-level planner powered by Monte Carlo Tree Search (MCTS) with a low-level planner based on Regression Planning (RP). This Modular Hierarchical Planner dynamically infers other agents' intentions and adjusts its subgoals accordingly, based on their observed actions.

3) MAT (Multi-Agent Transformer) [39], which is a multi-agent reinforcement learning model. We follows the MAT model trained in CoELA. It conceptualizes MARL as a sequence modeling challenge, employing a centralized decision transformer to orchestrate action generation. The input for the MAT is bifurcated. The primary segment encompasses a top-down semantic map of dimensions (12, 24), derived from oracle perception. The secondary input segment incorporates agent-specific information, such as container possession status and the count of held objects.

4) CoELA (Cooperative Embodied Language Agent) [43] is an advanced collaborative embodied agent that seamlessly integrates perception, memory, and execution for planning, communication, and collaboration. By harnessing the vast world knowledge and superior reasoning capabilities of LLMs, along with their expertise in natural language processing, CoELA effectively coordinates with other agents to tackle complex, real-world tasks.

5) RoCo (Dialectic Multi-Robot Collaboration) [24] is an embodied collaborative agent that utilizes LLMs for both high-level communication and low-level path planning. It facilitates agent collaboration through natural language dialogues. Using multi-round interactions, agents share mutual information and provide feedback. We implement RoCo in TDW-MAT and CWAH environments, where agents determine a unified collaboration plan through dialogue. Based on this plan, agents take actions, and any new progress triggers further discussions to refine the plan.

6) CaPo (Cooperative Plan Optimization) [22] is similar to RoCo, as both discuss cooperative plans through dialogue before executing them. CaPo consists of two key phases: meta-plan generation and meta-plan updating. During the meta-plan generation phase, agents exchange information to determine the plan, while in the meta-plan updating phase, the plan is revised when new items are discovered or sub-tasks are completed.

# **D.** Examples

In this section, we present several examples of CoTS in experiments. First, Fig. 10 provides detailed node information for a specific path within the collaborative tree. Then, Fig. 11 presents two concrete examples from the plan evaluation module, representing cases where the current plan needs to be updated and where it does not.

# **E. Prompt Templates**

We have listed the following templates: Alice Prompt (Fig. 12), Bob Prompt (Fig. 13), Plan Evaluation Prompt (Fig. 14), and Plan Parsing Prompt (Fig. 15).

Task: transport 2 iphones, 2 purses, 2 mouses, 2 lighters, 1 pen, 1 calculator to the bed

The Process in Each Node: Alice makes plans, Bob provides suggestions and rewards.

**Plan from Alice:** Step 1: Alice grasps the <mouse> (9068700). Bob explore his current room. Step 2: Alice explore the <Livingroom> (2000). If they find any target items, transport them to the bed.

**Message to Bob:** Hi Bob, I am currently in <Livingroom> (4000) where I found a <mouse> (9068700) which I plan to grasp. As I am not aware of your location or progress, please let me know if you have any suggestions or find any target objects or containers. Let's focus on exploring the rooms we are currently in and grasp any found target objects.

**Message to Alice:** Hi Alice, I am currently in the <Livingroom> (4000) and have found an <iphone> (14360913). It would be more efficient if I continue to explore the <Livingroom> (4000) while you explore a new room. Let's avoid exploring the same room together and utilize our abilities more efficiently. Can we adjust the plan accordingly?**Reward: 0.4** 

**Plan from Alice:** Step 1: Alice moves to the <Bedroom> (8000). Bob grasps the <iphone> (14360913). Step 2: Alice explore the <Bedroom> (8000). Bob explores the <Livingroom> (4000). Step 3: If they find any target items, transport them to the bed.

**Message to Bob:** Hi Bob, thanks for the update. I will now shift to the <Bedroom> (8000) to avoid duplication of our efforts. Please continue exploring <Livingroom> (4000) and grasp the iphone> (14360913) you found. Let's keep each other updated on our progress.

**Message to Alice:** Hi Alice, I think it would be more efficient to split up and explore different rooms. While I continue to explore the <Livingroom> (4000) and retrieve the <iphone> (14360913), you could head to the nearest room, the <Livingroom> (2000), to explore and look for more target objects. I suggest you grasp the <mouse> (9068700) in your current room before moving to <Livingroom> (2000). This way, we can cover more ground and find the target objects more quickly? **Reward: 0.7** 

**Plan from Alice:** Step 1: Alice grasps the <mouse> (9068700). Bob grasps the iphone> (14360913).Step 2: Alice
moves to the <Livingroom> (2000). Bob explores the <Livingroom> (4000).Step 3: Alice explores the <Livingroom> (2000). If they find any target items, transport them to the bed.

**Message to Bob:** Hi Bob, I agree with you. I will move to the <Livingroom> (2000) to explore while you continue in the <Livingroom> (4000). This should help us cover more ground quickly. Please keep me updated on your progress.

**Message to Alice:** Hi Alice, your plan looks good. I will grasp the *iphone* (14360913) in my current room and continue exploring for other target objects or containers. If I find another object or container, I will grasp it and transport to thebed if I am holding two objects. **Reward: 0.9** 

Figure 10. An example of node information in a path of the collaboration tree, selected from the experiment on TDW-MAT.

## An Example of the Current Plan That Do Not Needs Updating

## **Current Plan**

Step 1: Alice go grasp <orange> (14249767) and the container <tea\_tray> (16224318). Bob continues to explore the Kitchen (5000).

Step 2: Alice put <orange> (14249767) into the container <tea\_tray> (16224318). If Bob finds any target objects in the Kitchen (5000), he should go grasp up to two of them.

Step 3: Alice transport the container <tea\_tray> (16224318) with the <orange> to the bed. If Bob is holding any target objects, he should transport them to the bed. If not, Bob should continue to explore the Kitchen (5000).

#### **Plan Evaluation Reasoning**

According to the latest progress, Alice and Bob are following the action plan. Alice has found the target object <orange> (14249767) and containers <tea\_tray> (16224318), <plate> (1457820) in the <Livingroom> (4000) as planned. Bob is in the <Kitchen> (5000) and exploring. As per the plan, Alice needs to grasp the <orange> and one container, put the <orange> into the container, and then transport them to the bed. Bob is supposed to continue exploring the Kitchen and grasp target objects if found. No new target objects have been found by Bob and Alice is yet to execute the planned action. Therefore, there is no need to change the action plan immediately. Reward: [5].

#### Current Plan

# An Example of the Current Plan That Needs Updating

Step 1: Alice go to the <Kitchen> (2000), Bob continues to explore the <Livingroom> (3000). Step 2: Alice explore the <Kitchen> (2000), Bob go grasp <orange> (7175626). Step 3: Alice go grasp up to two objects if found, Bob transport holding object to the bed.

## **Plan Evaluation Reasoning**

According to the latest progress, Alice is in the Bedroom (6000) and has found a loaf of bread (13469629), which is one of the target objects, but she hasn't acted to grasp it yet. Meanwhile, Bob is in the Livingroom (3000) and has found an orange (7175626), a target object, but he hasn't acted to grasp it either. The current action plan is for Alice to proceed to the Kitchen (2000) and Bob to grasp the orange (7175626). However, since Alice has already found a loaf of bread (13469629) in Bedroom (6000), it would be more efficient for her to grasp that first before moving to another room. Similarly, Bob should also act to grasp the orange (7175626) he has found before continuing to explore. Therefore, the current action plan could be improved to be more efficient. Reward: [2].

Figure 11. Examples in the plan evaluation module.

I am Alice. My teammate Bob and I want to transport as many target objects as possible to the bed with the help of containers within 3000 steps. Remember Alice or Bob each can hold two things at a time and the 2 objects can be objects or containers. Alice or Bob can grasp containers and put objects into them to hold more objects at a time. Assume Alice is an expert in designing plan outlines. Given our shared goal, previous plan, dialogue history, latest progress, Bob's suggestion, please help me generate/refine the global plan for Bob and me during task execution, guiding us to achieve the goal collaboratively as soon as possible. Note that a container can hold three objects and will be lost once transported to the bed. I can only place objects into a container I am holding after grasping it. All objects are denoted as <name>(id), such as (712). Actions take several steps to complete. Note that it may be costly to go to another room or transport to the bed, use these actions sparingly.

The generated collaboration plan should strictly meet following requirements:

1. You should make full use of the agent's transport capabilities (up to two objects or containers per agent) to transport as much as possible efficiently. The collaboration plan should reasonably arrange the division of action between Alice and Bob in order to achieve the goal as soon as possible. Please be as detailed as possible in assigning actions to each agent, specific to a room or item.

2. Please keep your reasoning process, but the final collaboration plan should be brief, reliable, authentic, and consistent with the latest progress of Alice and Bob. Don't make random and meaningless plans.

3. There are only 5 allowed actions you can use to construct the collaboration plan. 1) 'go to': move to a specified room. 2) 'explore': explore a room for underlying target objects. 3) ' go grasp': go to grasp a specified target object. 4) 'put': Place an object into a specified container. 5) 'transport': Transport holding objects or containers to the bed and drop them on the bed.

4. The collaboration plan should be detailed to each Agent. The collaboration plan only needs to consider three steps at most at current time. When there is not much known information or content to be planned, the collaboration plan can have only one or two steps. The collaboration plan must be structured strictly in the format: {Action Plan: Step 1: Alice xxx, Bob xxx; Step 2: Alice xxx, Bob xxx; Step 3: Alice xxx, Bob xxx}. 'xxx' represents one or multiple allowed actions.

5. In order to let Bob know about Alice's situation, you need to generate a short message to Bob. The message has to be concise, reliable, and helpful for assisting Bob and Alice to make an efficient and consistent collaboration plan, and transport as many objects to the bed as possible. Don't generate repetitive messages.

6. Alice and Bob act separately and can only exchange information and they cannot exchange items.

Here is an example for Alice:

{Goal: [Transport 2 oranges, 3 apples, 1 banana, 3 breads, 1 burger to the bed.]

Reasoning: [According to Bob's suggestions and progress, the current plan is partially reasonable and needs minor adjustments. Bob found <orange> (5345043) and <br/>banana> (6381322) in <Livingroom> (4000), but he can't take more things in his hand. He needs to transport them to the bed first. Alice currently is not far from <Livingroom> (4000), so Alice can go to <Livingroom> (4000) to get <orange> (5345043) and <br/><br/>banana> (6381322).]

Collaboration plan: [Step 1: Alice go to the <Livingroom> (4000). Bob transport holding objects <bread> (547795) and <br/> <br/> (521494) to the bed and drop them on the bed.

Step 2: Alice go grasp <orange> (5345043) and <br/>shanana> (6381322). Bob goes to the <Kitchen> (5000) to explore. Step 3: Alice transports holding objects to the bed. If any target objects or containers are found by Bob, Bob go grasp objects.]

Message: [Hi Bob, I have received your message and will adjust our collaboration plan. You continue to transport the target items in your hand to the bed. I will transport the <orange> (5345043) and <br/>banana> (6381322) in <Livingroom> (4000) you found. Do you have any new suggestions for the updated plan?]

Following are provided information for Alice: Goal: \$GOAL\$ Previous Collaboration Plan: \$PREVIOUS\_PLAN\$ Dialogue History: \$DIALOGUE\_HISTORY\$ Alice's Progress: \$ALICE\_PROGRESS\$, it's need to consider Alice's latest action progress.

Think step by step, and generate the response:

Figure 12. Alice Prompt.

I am Bob. My teammate Alice and I want to transport as many target objects as possible to the bed with the help of containers within 3000 steps. Remember Alice or Bob can hold two things at a time, and they can be objects or containers. Alice or Bob can grasp containers and put objects into them to hold more objects at a time. Because Alice may not understand Bob's current progress and information, and may not consider the plan comprehensively and perfectly, which wastes our action time. Given our shared goal, collaboration plan, dialogue history, progress, and my previous actions, please help me analyze and reward Alice's proposed collaboration plan, point out the shortcomings of Alice's plan and reflect on it and finally generate a message to send to Alice, at the beginning of the message, I should first explain my findings. Alice and Bob actseparately and they cannot exchange items. It may be costly to go to another room or transport to the bed, use these actions sparingly.

The content Bob generate mainly consists of two parts: reasoning and message sent to Alice. Please strictly follow the following format:

Reasoning: [the reasoning process, analyze the unreasonableness of the current plan and consider how to make it more efficient]

Dis\_Reward: [the reward between 0 and 5: 1, No consideration of distance 2, Minimal consideration of distance, overlooking key factors 3, Distance is considered but not entirely accurate 4, Distance is sufficiently considered, with only minor oversights 5, Comprehensive and accurate consideration of distance. Only output the reward.] Task\_Reward: [the reward between 0 and 5: 1, No consideration of work distribution between Alice and Bob 2, Minimal consideration of work distribution, leading to unreasonable allocation 3, Work distribution is considered but not entirely accurate 4, Work distribution is reasonable, with only minor oversights 5, Work distribution is highly effective, making full use of Alice and Bob's abilities. Only output the reward.] Message: [the message sent to Alice, you need to first tell Alice about your findings]

Here is an example for your reference: :

Reasoning: [In the current plan, Alice may not know our progress, so some of Bob's plans are vague, which needs to be improved. In addition, it is a waste of time for Alice and Bob to explore the same room together.] Dis\_Reward: [4]

Task\_Reward: [2]

Message: [Hi, Alice, I dont find any objects and we are in the same room, it is not efficient for us to explore the same room together. Your plan needs to be adjusted and describe my actions as detailed as possible.]

The following is the information of Bob currently:

Bob's Previous Action: \$ACTION\_HISTORY\$ Bob's Progress: \$BOB\_PROGRESS\$, it's need to consider Bob's latestprogress and the distance is a dimensionless relative measure.

The following is the relevant information when Alice is planning her collaboration plan, which can be used as a reference for Bob:

User: {user input}\n

Alice's response: {candidate\_content}

Think step by step, and generate the content sent to Alice:

Figure 13. Bob Prompt.

I am Bob. My teammate Alice and I want to transport as many target objects as possible to the bed with the help of containers within 3000 steps. Important: Alice or Bob can hold two things at a time, and they can be objects or containers. I can grasp containers and put objects into them to hold more objects at a time. Note that a container can contain three objects, and will be lost once transported to the bed. I can only put objects into the container I hold after grasping it. All objects are denoted as <name> (id), such as (712). Actions take several steps to finish. Before, Alice has generated an collaboration plan based on our progress and discussion to coordinate Alice and Bob to complete the task efficiently. Now some steps have passed since the last discussion of the collaboration plan, given our shared goal, collaboration plan, dialogue history, progress, and our progress can only reflect whether the action has been completed, but not whether the action is being taken.

Your response needs to include your thought process and scoring. Scoring is mainly based on the following criteria: PA\_Reward: [the reward between 0 and 5: 1, The current plan is completely unreasonable and needs to be revised 2, The current plan requires significant adjustments 3, The current plan needs improvement, its core structure is sound, though details are unclear 4, The current plan requires only minor adjustments 5, The current plan is fully reasonable and needs no changes. Only output the reward.]

Here are two examples for you:

Reasoning: According to the latest progress, There is currently no plan to execute, so a plan needs to be generated. Plan\_Reward: [1]

Reasoning: According to the latest progress, Bob and Alice are both acting according to plan. Alice is in the process of taking action and Bob found new target items <ipod> (2189332) in the room <Office> (3000), which was not considered in the previous collaboration plan. We should update our collaboration plan to take them into account. Plan\_Reward: [3]

Following are provided information for you:

Goal: \$GOAL\$ Collaboration Plan: \$ACTION\_PLAN\$ Dialogue History: \$DIALOGUE\_HISTORY\$ Previous Action: \$ACTION\_HISTORY\$ Alice and Bob's Progress Desc: \$PROGRESS\$, it's important to consider Alice and Bob's latest action progress and the distance is a dimensionless relative measure.

Think step by step, and generate the message:

Figure 14. Plan Evaluation Prompt.

I am \$AGENT\_NAME\$. My teammate \$OPP\_NAME\$ and I want to transport as many target objects as possible to the bed with the help of containers within 3000 steps. I can hold two things at a time, and they can be objects or containers. I can grasp containers and put objects into them to hold more objects at a time. All objects and rooms are denoted as <name>(id), <Living room> (3000) and <Living room> (1000) are not the same room.

Assume that you are an expert decision maker. Given our shared goal, collaboration plan, my progress, and previous actions, please help \$AGENT\_NAME\$ choose adjustments and select my next available action. Note that a container can contain three objects, and will be lost once transported to the bed. I can only put objects into the container I hold after grasping it. Actions take several steps to finish. It may be costly to go to another room or transport to the bed, use these actions sparingly.

The collaboration plan has the highest priority, if the number of steps is close to 3000, it is most important to get the things in hand or the nearest things to the bed as soon as possible, rather than continue to look for.

Because I need to complete the task step by step, you only need to help me to choose the next step that needs to be taken. Your response must be one of the Available actions.

Goal: \$GOAL\$ Collaboration Plan: \$ACTION\_PLAN\$ Available Actions: \$AVAILABLE\_ACTIONS\$ \$AGENT\_NAME\$ Previous Action: \$ACTION\_HISTORY1\$ \$OPPO\_NAME\$ Previous Action: \$ACTION\_HISTORY2\$ Progress: \$PROGRESS\$

Think step by step, and choose the next step of \$AGENT\_NAME\$:

Figure 15. Plan Parsing Prompt.