A Human-in-the-Loop DAG Refinement Framework for LLM-Driven Multi-Robot Construction Task Planning

Yongdong Wang[1†], Runze Xiao[1], Jun Younes Louhi Kasahara[1], Shota Chikushi[1,2], Keiji Nagatani[1,3], Atsushi Yamashita[1] and Hajime Asama[1]

[1]The University of Tokyo, Tokyo, Japan
(Tel: +81-03-5841-0359; E-mail: wangyongdong@robot.t.u-tokyo.ac.jp)
[2]Kindai University, Hiroshima, Japan
[3]University of Tsukuba, Ibaraki, Japan

Abstract: We propose a human-in-the-loop refinement framework that addresses the critical challenge of ensuring safe and reliable LLM-driven multi-robot construction task planning through visual DAG-based verification and iterative refinement. The framework incorporates a Question-Answering (QA) LLM module to translate the operator's natural language instructions into a structured DAG representing atomic task dependencies. A Task Interface module presents the generated task graph to the operator for inspection and feedback. Through interactive dialogue, the operator can iteratively refine the task plan and intervene to prevent the execution of potentially unsafe actions. Experimental results demonstrate that the framework enables real-time responsiveness and effectively blocks incorrect DAG task plans from being dispatched to the robot team. In error refinement evaluations using a public dataset, the Llama3.1-8B model required an average of 1.42 dialogue turns to correct task dependency errors, whereas the GPT-4.1 model achieved error correction in just 1.17 turns. Through the iterative refinement mechanism, the framework is capable of transforming initially erroneous sub-tasks into safe and executable plans, enabling the reliable deployment of LLMs in multi-robot construction sites.

Keywords: Safe Multi-Robot Construction; Human-in-the-Loop Task Planning; Directed Acyclic Graphs (DAG); Large Language Model (LLM); Task Verification

1. INTRODUCTION

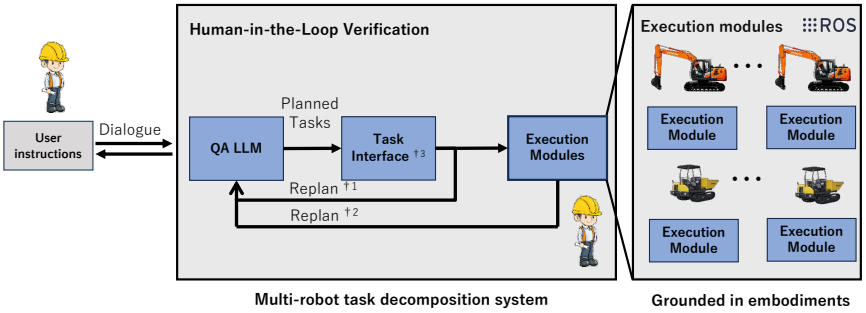
The construction industry is confronting a critical labor shortage. Construction sites typically deploy numerous machines, with traditional operational practices adhering to a "one operator per machine" paradigm, leading to a linear increase in labor demand with the number of machines. In Japan, for instance, the proportion of construction workers aged 55 and above has continued to grow, while the number of younger workers under 30 has declined significantly [1]. To address these challenges, enabling "single-operator, multi-machine" control for groups of construction robots has become a key technological imperative to mitigate labor shortages and enhance operational efficiency.

In the domain of multi-robot construction collaboration, Saboia et al. proposed a distributed reactive control framework for heterogeneous robot teams [2]; Hartmann et al. developed a long-horizon multi-robot assembly planning algorithm that integrates logic-geometric programming with bidirectional spatiotemporal Rapidly-exploring Random Tree (RRT) [3]; and Asani et al. implemented a centralized task scheduling system for synchronized transport of large components [4]. However, these methods rely heavily on formalized task descriptions and precise geometric parameters, rendering them incapable of interpreting natural language commands and limiting their responsiveness to dynamic or unexpected situations.

The emergence of large language models (LLMs) offers a promising direction for overcoming these limitations due to their powerful natural language understanding capabilities. Zhang et al. developed a modular framework that employs LLMs to construct collaborative embodied agents, aiming to enable flexible and composable multi-agent cooperation [5]. Chen et al. introduced a task and motion planning approach based on an autoregressive framework, where LLMs function as both translators and validators to enhance planning accuracy and robustness [6]. Additionally, Chen et al. proposed a scalable architecture for multi-robot collaboration and compared centralized and decentralized systems under LLM guidance, finding that none of the configurations could consistently guarantee the success of LLM-generated task plans [7]. Kannan et al. explored direct code generation for robot swarms using LLMs, but the resulting code frequently exhibited syntactic and logical errors, undermining its reliability [8]. Wang et al. proposed a safe task planning method that relies on LLM confidence thresholds to trigger assistance in a passive manner. While this approach improved task success rates, it did not allow the operator to proactively and intuitively review, modify, or adjust task plans in real time [9].

While previous studies have developed various LLM-driven robotic planning methods, these systems lack comprehensive safety validation mechanisms. In contrast, this work introduces a human-in-the-loop Directed Acyclic Graph (DAG) refinement framework for large language model (LLM)-driven multi-robot construction task planning. As illustrated in Fig. 1, the proposed framework decomposes natural language instruc-

† Yongdong Wang is the presenter of this paper.



†1: **Replanning** occurs before the execution of tasks by the Execution Modules.
†2: **Replanning** occurs after the execution of tasks by the Execution Modules.
†3: **Task Interface** is based on Directed Acyclic Graph (DAG).

Fig. 1 Overview of the Human-in-the-Loop Multi-Robot Task Decomposition Framework. The left panel shows the task decomposition system with LLM processing and human verification using DAG-based interfaces. The right panel illustrates the ground embodiments executing approved tasks through ROS2 execution modules. The operator can replan either before the Execution Modules execute their assigned tasks or afterward.

tions provided by the operator into a structured DAG-based task dependency graph. A visual interface allows the operator to inspect, validate, and refine task plans, thereby ensuring their accuracy and executability. When unexpected environmental changes occur, the operator can dynamically adjust and refine task nodes through a dialogue-based interface, enabling real-time adaptation. The main contributions of this work are as follows:

1. A DAG-based method for LLM-driven decomposition and visual verification of multi-robot construction tasks, enabling structured modeling and safety validation of natural language instructions.
2. A human-machine interaction mechanism that supports dynamic adjustment when task plans require refinement or correction, enabling real-time DAG reconstruction through iterative dialogue-based feedback, thereby enhancing planning flexibility and execution reliability.

2. DAG-BASED TASK DECOMPOSITION AND VERIFICATION

2.1. System Workflow Overview

Fig. 2 illustrates the temporal workflow of the system's task refinement procedure. The QA LLM module initially receives high-level natural language instructions from the operator and, incorporating environmental information, the robot capability library, and few-shot examples, decomposes them into a sequence of atomic tasks structured as a DAG. The Task Interface module subsequently transforms this list into a visualized task dependency graph, allowing the operator to review task logic, interdependencies, and execution order. If errors are detected or modifications are required, the operator may interact with the QA LLM module again, enabling the system to iteratively refine the task decomposition based on the dialogue history. Once confirmed, the operator transmits the DAG-structured atomic task plan to the robot swarm for execution via the Task Interface. It should be noted that this iterative refinement process can be conducted multiple times at any stage, both prior to and following task execution by the Execution Modules.

2.2. Task Decomposition in QA LLM Module

The operator provides high-level natural language instructions to the QA LLM module, which synthesizes environmental context (e.g., object names and locations), the robot's primitive capabilities, and few-shot learning examples to decompose the instruction into a DAG of executable atomic sub-tasks. These few-shot examples, structured in JSON format, explicitly encode dependency relationships, enabling the LLM to understand and replicate the DAG-based task decomposition paradigm. As illustrated in Fig. 3, the dependencies field specifies the directed edges among sub-tasks, constructing a coherent execution graph that is forwarded to the Task Interface module for operator verification and safety confirmation.

2.3. Visual Verification in Task Interface Module

To ensure the safety and executability of task plans generated by the LLM, this paper presents a visual verification interface, as illustrated in Fig. 4. The interface employs a graphical DAG representation that converts complex task dependencies into an intuitive node-edge structure. Task nodes are color-coded based on their roles: red for start tasks, orange for intermediate tasks, and purple for terminal tasks, with directed edges clearly indicating the dependency relationships. Each node is accompanied by a rectangular information panel that details the specific execution steps, the types of robots involved, and the objects being manipulated. The operator can inspect task details through the generated graphical DAG and refine illogical tasks in real time via interactive dialogue.

3. EXPERIMENTAL EVALUATION

3.1. Validation of Dynamic Task Adjustment Capabilities

A simulation experiment was conducted to evaluate the dynamic task adjustment capabilities of the proposed human-machine interaction framework. The simulation environment was developed using Unity, and communication between the QA LLM module and the Task Interface module was established via the ROS2 topic mech-

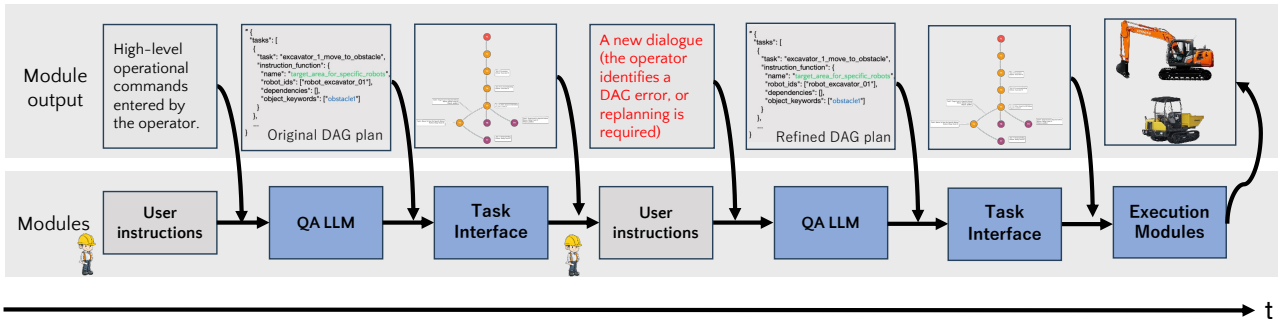


Fig. 2 Temporal workflow of the human-in-the-loop DAG task refinement framework. The QA LLM module processes natural language instructions and decomposes them into atomic tasks structured as a DAG, which is then visualized by the Task Interface module for operator verification and iterative refinement through interactive dialogue.

```
{
    "instruction_function": {
        "name": "<breakdown function 1>",
        "dependencies":["<dep 1>", "<dep 2>",
            …, "<dep n>"]
    },
    "object_keywords": ["<key 1>", "<key 2>",
        …, "<key n>"],

    "instruction_function": {
        "name": "<breakdown function 2>",
        "dependencies":["<dep 1>", "<dep 2>",
            …, "<dep n>"]
    },
    "object_keywords": ["<key 1>", "<key 2>",
        …, "<key n>"],

    ...

    "instruction_function": {
        "name": "<breakdown function m>",
        "dependencies":["<dep 1>", "<dep 2>",
            …, "<dep n>"]
    },
    "object_keywords": ["<key 1>", "<key 2>",
        …, "<key n>"]
}
```

Fig. 3 JSON format specification for DAG-structured task decomposition with dependency relationships and robot-specific execution parameters generated by the QA LLM module.

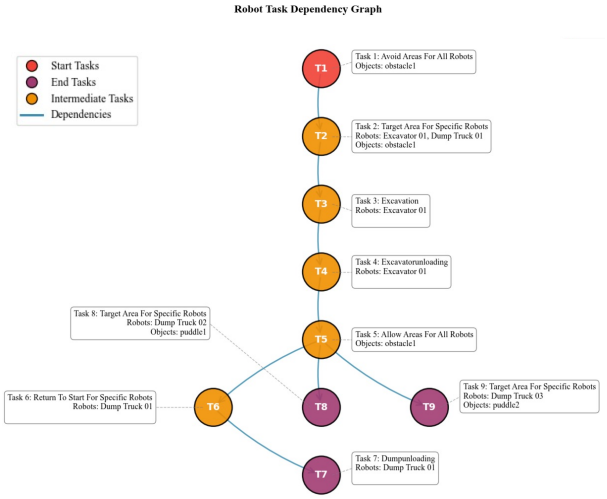


Fig. 4 DAG visualization interface featuring color-coded task nodes, dependency arrows, and real-time dialogue panel for human-in-the-loop task plan verification and refinement.

anism. As shown in Fig. 5, the experimental scenario includes an obstacle area referred to as a "puddle", and the dump truck is required to perform an inspection task.

At the beginning of the experiment, the operator issued the instruction "dump truck go inspect the puddle." The QA LLM module decomposed this instruction into the atomic task T1 (navigate to the puddle) and generated a corresponding DAG. As the dump truck approached the puddle, the operator noticed that the current path would cause the dump truck to drive directly through the puddle. Therefore, approximately 60 seconds into the experiment, the operator issued a new command—"avoid the puddle before executing all tasks" to the QA LLM module. Through the Task Interface module, the operator confirmed the task modification: the system restructured the task dependencies by introducing a new avoidance task (T1) and making the original inspection task (T2) dependent on the successful completion of the avoidance task. The system then refined the task dependency graph accordingly, updating the original navigation task to execute only after the successful completion of the new avoidance task.

Table 1 Performance Comparison of Llama3.1-8B and GPT-4.1 Models in DAG Task Dependency Error Correction

Model	Avg. Number of Correction Rounds	Success Rate (%)
Llama3.1-8B	1.42	100
GPT-4.1	1.17	100

3.2. Evaluation of DAG Validation and Error Correction Performance

Using the dataset provided by Wang et al. [10], 12 test cases containing dependency logic errors were selected to evaluate the performance of different LLMs in DAG generation and error correction. Two models, Llama3.1-8B and GPT-4.1, were evaluated. Through the visual validation interface of the Task Interface module, the operator was required to interactively generate a valid DAG-based task decomposition within a maximum of three rounds of dialogue.

Table 1 presents a performance comparison between the two models in the visual DAG validation and error correction tasks. Experimental results demonstrate that the proposed human-in-the-loop DAG refinement framework exhibits strong performance in error correc-

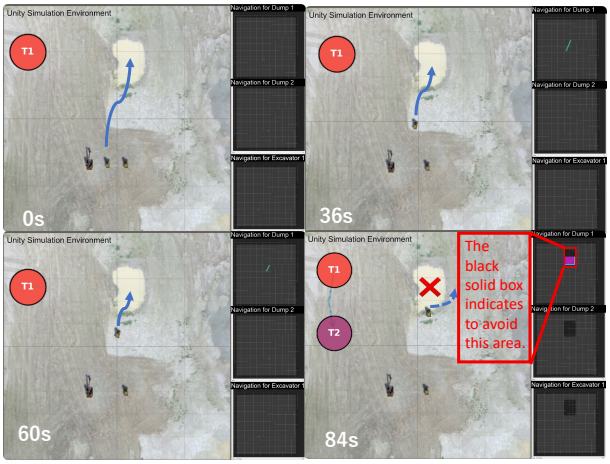


Fig. 5 Real-time DAG refinement simulation experiment of the proposed method. At 0s, the dump truck begins executing task T1 (inspect puddle) following a direct path to the puddle. At 60s, as the dump truck approaches the puddle, the operator recognizes that the current execution path would cause the robot to traverse through the puddle and issues a refinement command. The system activates the DAG refinement mechanism, restructuring the task dependencies: a new avoidance task becomes T1, while the original inspection task becomes T2 and now depends on successful puddle avoidance.

tion tasks. Evaluated on 12 dependency logic error cases from standard datasets, the Llama3.1-8B model required an average of 1.42 dialogue rounds to complete the corrections, while the GPT-4.1 model achieved the same with an average of just 1.17 rounds. Both models attained a 100% correction success rate, indicating that all test cases could be successfully transformed into correct DAG task decompositions within three dialogue rounds. GPT-4.1 demonstrated higher efficiency in task comprehension and refinement, reducing the number of interaction rounds by 17% compared to Llama3.1-8B. These findings confirm the effectiveness of the DAG-based verification method and highlight performance differences among large language models in understanding complex task dependencies and executing error correction.

4. CONCLUSION AND FUTURE WORK

This paper presented a human-in-the-loop DAG refinement framework that tightly integrated task decomposition by LLM with refinement guided by detailed feedback from the human operator. Using a visualization-based verification interface and an interactive dialogue mechanism, the framework demonstrated the ability to dynamically adapt to unexpected environments in both simulation experiments and evaluations on standard datasets, while enabling effective error correction. On average, the Llama3.1-8B model required 1.42 dialogue rounds to successfully resolve task dependency errors, whereas the GPT-4.1 model required only 1.17 rounds to achieve successful refinement. Future work includes developing safety mechanisms to handle human judgment errors and communication failures where human input cannot be transmitted to the robot system.

ACKNOWLEDGMENTS

This work is supported by JST [Moonshot Research and Development], Grant Number [JPMJMS2032].

REFERENCES

[1] Policy Bureau, Ministry of Land, Infrastructure, Transport and Tourism (MLIT), "Summary of the white paper on land, infrastructure, transport and tourism in japan, 2024," tech. rep., Ministry of Land, Infrastructure, Transport and Tourism, Japan, 2024.

[2] M. Saboia, V. Thangavelu, and N. Napp, "Autonomous multi-material construction with a heterogeneous robot team," *Robotics and Autonomous Systems*, vol. 121, p. 103239, 2019.

[3] V. N. Hartmann, A. Orthey, D. Driess, O. S. Oguz, and M. Toussaint, "Long-horizon multi-robot rearrangement planning for construction assembly," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 239–252, 2022.

[4] Z. Asani, M. Ambrosino, B. Vanderborght, and E. Garone, "Implementation of a heterogeneous multi-robot system for a construction task," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 3373–3378, 2023.

[5] H. Zhang, W. Du, J. Shan, Q. Zhou, Y. Du, J. B. Tenenbaum, T. Shu, and C. Gan, "Building cooperative embodied agents modularly with large language models," *arXiv preprint arXiv:2307.02485*, 2023.

[6] Y. Chen, J. Arkin, C. Dawson, Y. Zhang, N. Roy, and C. Fan, "Autotamp: Autoregressive task and motion planning with llms as translators and checkers," in *2024 IEEE International conference on robotics and automation (ICRA)*, pp. 6695–6702, IEEE, 2024.

[7] Y. Chen, J. Arkin, Y. Zhang, N. Roy, and C. Fan, "Scalable multi-robot collaboration with large language models: Centralized or decentralized systems," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4311–4317, IEEE, 2024.

[8] S. S. Kannan, V. L. Venkatesh, and B.-C. Min, "Smart-llm: Smart multi-agent robot task planning using large language models," *arXiv preprint arXiv:2309.10062*, 2023.

[9] J. Wang, G. He, and Y. Kantaros, "Safe task planning for language-instructed multi-robot systems using conformal prediction," *CoRR*, 2024.

[10] Y. Wang, R. Xiao, J. Younes, L. Kasahara, K. Nagatani, A. Yamashita, and H. Asama, "Construction of a dataset for i-construction robots using directed acyclic graph–based task decomposition," in *Proceedings of the 7th i-Construction Promotion Symposium*, pp. 137–140, Japan Society of Civil Engineers (JSCE), 2025.