electronics

Article

Semantic Knowledge-Based Hierarchical Planning Approach for Multi-Robot Systems

Sanghyeon Bae, Sunghyeon Joo, Junhyeon Choi, Jungwon Pyo, Hyunjin Park and Taeyong Kuc *

Department of Electrical and Computer Engineering, College of Information and Communication Engineering, Sungkyunkwan University, Suwon 16419, Republic of Korea; shbae.skku@skku.edu (S.B.); sh.joo@skku.edu (S.J.); choijunhyeon@skku.edu (J.C.); jungwon900@skku.edu (J.P.); hyunjinp@skku.edu (H.P.)
* Correspondence: tykuc@skku.edu

Abstract: Multi-robot systems have been used in many fields by utilizing parallel working robots to perform missions by allocating tasks and cooperating. For task planning, multi-robot systems need to solve complex problems that simultaneously consider the movement of the robots and the influence of each robot. For this purpose, researchers have proposed various methods for modeling and planning multi-robot missions. In particular, some approaches have been presented for high-level task planning by introducing semantic knowledge, such as relationships and domain rules, for environmental factors. This paper proposes a semantic knowledge-based hierarchical planning approach for multi-robot systems. We extend the semantic knowledge by considering the influence and interaction between environmental elements in multi-robot systems. Relationship knowledge represents the space occupancy of each environmental element and the possession of objects. Additionally, the knowledge property is defined to express the hierarchical information of each space. Based on the suggested semantic knowledge, the task planner utilizes spatial hierarchy knowledge to group the robots and generate optimal task plans for each group. With this approach, our method efficiently plans complex missions while handling overlap and deadlock problems among the robots. The experiments verified the feasibility of the suggested semantic knowledge and demonstrated that the task planner could reduce the planning time in simulation environments.

Keywords: multi-robot system; semantic navigation; hierarchical planning; semantic knowledge; robotics

Citation: Bae, S.; Joo, S.; Choi, J.; Pyo, J.; Park, H.; Kuc, T. Semantic Knowledge-Based Hierarchical Planning Approach for Multi-Robot Systems. *Electronics* **2023**, *12*, 2131. https://doi.org/10.3390/electronics12092131

Academic Editor: Cecilio Angulo

Received: 27 March 2023
Revised: 30 April 2023
Accepted: 3 May 2023
Published: 6 May 2023

1. Introduction

In recent years, significant advancements in robotics have led to increasing robot applications in various fields, such as manufacturing and service industries. Robots are being used to replace human workers in many industries because they can operate continuously and require a lower cost than labor. As the use of robots grows, the use of multiple robots together is also increasingly in focus. The multi-robot system has the advantage of controlling each robot in parallel, thus performing more complex tasks than a single robot [1]. In addition, when a specific robot has a failure during a mission, another robot can be replaced to perform the task [2]. With these advantages, multi-robot systems are being utilized for various missions, such as delivery, guidance, and security [3–5]. However, because multi-robot systems need to consider not only the driving area and obstacles but also the paths of all robots, there have been numerous kinds of research for multi-robot systems, such as collision avoidance and path planning [6–8].

For a multi-robot system to successfully carry out its mission, it requires detailed knowledge about its operating environment. Most multi-robot systems operate in environments shared with humans, and robots need semantic knowledge to interact with these environments in a human-like manner. This semantic knowledge encompasses information about concepts, properties, purposes, and relationships acquired through perception and judgment. For example, humans recognize traffic signals on the road, cross the crosswalk, and also wait to use the elevator. Similarly, robots need to utilize semantic information, such as the connections, rules, and properties of the place, and utilize them to plan their missions. If not, the robot in the previous example could cause many problems, such as trespassing on the road without permission or planning the wrong path. Thus, several studies have been proposed to integrate semantic knowledge, such as the geometric position relationships of environmental elements and special rules for places and behaviors, into a robot's map [9–11].

Semantic maps allow robots to understand their environment in detail based on much information, such as knowledge of environmental elements and their relationships. Planning robotic missions with heavy semantic maps that contain a huge amount of information requires an efficient modeling approach. One method that has been applied to overcome this complexity is the Hierarchical Task Network (HTN) [12]. The HTN is an approach to effectively representing problems by dividing complex tasks into simplified tasks and has been applied in various methods to solve complex problems [13]. It can be used for robotic missions that require a high level of knowledge and reasoning with semantic knowledge [14]. The constraint model is also helpful for modeling tasks in complex domains. The constraint model defines the execution and finish conditions for a task to explore the changing state performing the task and generates a plan to complete the final goal. This study uses the HTN and the constraint model to define the multi-robot tasks with semantic knowledge. Hierarchical task structures are usually separated by the scale of the task, such as the mission, task, behavior, and action. However, our proposed approach separates the task layer into more detailed layers based on hierarchical relationship information in space. In addition, semantic knowledge such as occupancy and possession are integrated as constraints when modeling unit tasks at each hierarchical layer to represent multi-robot high-level tasks.

Several studies proposed using semantic knowledge to perform high-level tasks in indoor and outdoor environments [15,16]. These studies used semantic knowledge to define the task in detail and predict potential problems. As an extension, some studies that utilize semantic knowledge for multi-robot systems with a detailed description of the unique environment were proposed. Joo et al. [17] planned missions by defining semantic information about the time of opening and closing doors, the relationship of places, and the robot's capabilities in an indoor and outdoor environment. This method selected an adequate robot to perform a mission based on semantic knowledge and performed a high-level task plan using environmental elements, such as elevators and automatic doors. This study planned missions in an environment with multiple robots but did not consider the situations in which the robots operated in parallel. In another study, Moon et al. [18] performed a cooperative task for multiple AUVs and AGVs based on semantic knowledge. This method used semantic knowledge of objects and geometric relationships to understand the driving environment better and determine the mission failure situation by obstacles existing in the driving area. However, their method did not consider the influence of multiple robots on each other when planning missions. Task planners in multi-robot systems need to consider traffic problems, such as collisions and deadlocks between robots, when planning missions. To accomplish this, we extend semantic information that considers all the objects in the environment.

This paper proposes a task planner that utilizes semantic knowledge to plan multi-robot tasks quickly. Based on the semantic navigation framework [17], we extend the semantic knowledge and planning structure by considering the relationship knowledge between environmental elements in the multi-robot system. The hierarchical task planner utilizes layered spatial knowledge and a multi-level task model to plan complex multi-robot missions efficiently. We also introduce the re-planning approach for multi-robot mission failure based on the task planning structure. Simulation experiments were conducted to verify the improved performance of the proposed method over the previous method [17]. The experiments consisted of three parts: First, the failure-free driving of multiple robots by the proposed semantic knowledge was observed, and second, the time taken on task planning was measured. Finally, the re-planning time for mission failure situations was calculated. By comparing the experimental results, we validated the feasibility of the proposed semantic knowledge and verified the performance of multi-robot task planning and re-planning.

This paper is organized as follows: Section 2 compares studies related to the proposed research. Section 3 presents the extended knowledge and methods for multi-robot task planning and re-planning. Section 4 analyzes the suitability of our proposed approach through the experimental results. Finally, Section 5 summarizes the literature and discusses various implications and future work.

2. Related Work

Among the multi-robot task planning approaches, some methods have been proposed to generate the sequence of tasks to achieve the goal based on the constraints of each task and the system's state. Table 1 compares different constraint-based robot task planning methods. It represents the approach of the planning and modeling information for each robotic system.

Table 1. Comparison for constraints-based task planning approaches.

Name	System	Task Planner			Semantic Constraint Modeling	
		Planning Method	Hierarchical Structure	Re-Planning	Property and Relationship	Duplication
Hwang et al. [19]	Multi	CCBBA	O	X	-	-
Thomas et al. [20]	Multi	POPF	O	X	-	-
Schillinger et al. [21]	Multi	LTL	O	X	-	-
Buehler et al. [22]	Multi	TFD	X	X	√	-
Galindo et al. [23]	Single	Metric-FF	X	X	-	-
OMRKF [24]	Single	ABPLAN	O	X	√	-
Joo et al. [25]	Single	POPF	O	X	-	-
Hanheide et al. [26]	Multi	POMDP	X	X	√	-
Moon et al. [18]	Multi	POPF	X	X	√	-
TOSMNav [17]	Multi	POPF	O	O	√	-
Our	Multi	POPF	O	O	√	√

O: condition satisfied, X: condition unsatisfied, - : not considered, √ : considered

Most methods do not consider re-planning to handle problematic situations during the robot's task planning. Semantic knowledge, such as the property and relationship of objects and places, is used for constraint modeling. However, no study has utilized the knowledge of duplication for the multi-robot system. We explain the mentioned papers and our proposed method in detail for further understanding.

Hwang et al. proceeded with multi-robot task planning based on the constraint table and ordering rules for robot task planning [19]. In another way, Schillinger et al. proposed the algorithm for cooperative task planning by considering time constraints within defined rules [21]. In the multi-robot domain represented by constraints, it has also been proposed to use a classical planner to generate a task plan. Classical planners plan a robot's mission by exploring state changes to achieve a goal based on a defined working environment, robot task, and state. To represent various information, such as the initial state, goals, tasks, and constraints, the Planning Domain Definition Language (PDDL) [27] is used. PDDL is a standard language that supports several functions suitable for planning problems. It has been used in many different ways to represent multi-robot domains. Thomas et al. introduced a PDDL-based multi-robot cooperative navigation approach for TMP under motion and sensing uncertainty. A task–motion interaction is facilitated by means of semantic attachments that return motion costs to the task planner [20]. Buehler et al. defined the capabilities of robots and the tasks in PDDL for implementing parallel tasks among agents in a heterogeneous multi-robot system [22].

Adopting semantic knowledge has enabled robots to understand rules and knowledge to enable high-level tasks. Several studies have proposed utilizing ontology-based knowledge to represent robotic domains in various environments. Galindo et al. introduced the method to organize domains based on semantic knowledge to provide robots with a higher level of autonomy and intelligence [23]. They defined a semantic map integrating hierarchical spatial information and semantic knowledge to represent the ontology-based relationship knowledge of objects and places in indoor environments. Suh et al. proposed the Ontology-based Multi-layered Robot Knowledge Framework (OMRKF), which represents the robot domain as multiple layers of four knowledge classes: model, context, perception, and activity [24]. They represented each element in the multiple layers based on ontology and integrated robot knowledge with information, such as sensor data, primitive behaviors, and contextual information, through rule reasoning to realize an efficient delivery mission for robots. Joo et al. proposed the robot framework that introduced semantic knowledge to enable human-like behavior [25]. The framework built a semantic model of the environment to represent the semantic knowledge-based environment database and integrated it into the robot's perception and navigation modules. They used semantic knowledge-based environment information to perform object detection, localization, and inferred relationship knowledge for the robot. The inferred semantic knowledge was applied to a hierarchical planner that combines a classical planner and a reinforcement learning algorithm to implement high-level task planning.

These semantic knowledge-based robot planning methods have been extended to multiple robots. Hanheide et al. defined the ontology-based semantic knowledge categorized into three layers to represent the domain of a multi-robot system [26]. Among the knowledge in each separated layer, the higher knowledge is used to modify the lower knowledge to complement the knowledge of failure situations. The defined knowledge represented uncertain information in the robot's behavior to realize efficient multi-robot task planning and execution. Moon et al. used natural language-based scene understanding to represent and utilize semantic knowledge-based environmental information for a heterogeneous multi-robot system [18]. They proposed the task planning method that organizes the information of the environment obtained from scene understanding into ontology knowledge and utilizes it to respond to the changing state of the driving area. Joo et al. planned the mission for the multi-robot system based on a detailed semantic knowledge representation of indoor and outdoor environments [17]. They represented the semantic knowledge of the driving capabilities of the robots in the system, the opening and closing times of places, and the position relationships of objects based on ontology. The semantic knowledge of the defined environmental elements is used to build the on-demand database for mission planning by using Semantic Web Rule Language (SWRL)-based knowledge reasoning. Finally, the on-demand database information is converted into the PDDL problem to generate a robot mission plan based on a classical planner.

We propose a semantic knowledge-based multi-robot task planning method. We extend the semantic knowledge and planning structure suitable for the multi-robot based on the semantic navigation framework [17]. The extended semantic knowledge expresses relationships, such as possession and occupancy, to consider the effects and interactions between environmental elements in multi-robot systems. The proposed semantic knowledge is integrated into a hierarchical task and utilized for high-level mission planning. We also propose a hierarchical task structure that separates task levels in detail based on spatial-layer information. The hierarchical task is used to represent the multi-robot domain based on PDDL and the Partial-Order Planning Forwards (POPF) [28] planner to generate the task sequence in the defined multi-robot domain. We also introduce a re-planning approach that adapts to robot mission failure cases based on a hierarchical task planning structure.

3. Methods

This chapter describes the extended semantic knowledge and improved hierarchical task planning method for effective task planning in semantic knowledge-based multi-robot systems. The proposed approach extends the semantic navigation framework [17] to be suitable for multi-robot planning.

3.1. Semantic Knowledge-Based Multi-Robot System Modeling

The Triplet Ontology Semantic Model (TOSM) [25] is used to represent the semantic knowledge-based multi-robot domain. The TOSM combines numerical information and semantic knowledge about environmental elements. Based on the TOSM, we construct the semantic database by defining the relationship knowledge between environmental elements and data properties of places. Based on the extended semantic knowledge, tasks for multi-robot missions are defined based on the HTN structure and expressed based on the PDDL.

3.1.1. Semantic Knowledge Modeling

Based on the TOSM, we express the concepts of possession and usage to prevent deadlock and duplication problems in multi-robot systems. For this purpose, the relationship knowledge between robots and other robots, places, or objects in the environment is extended using the object properties. The object properties express the relationship knowledge between environmental elements, such as *Robot*, *Place*, and *Object* as the domain and range. Table 2 provides information about the extended object properties. We represent the class name of the environmental elements in italics to avoid word confusion.

Table 2. Extended object properties for multi-robot systems.

Object Property Hierarchy	Domains	Ranges
isOccupiedBy	*Place*	*Robot*
isNotOccupiedBy	*Place*	*Robot*
isUsedBy	*Object*	*Robot*
isNotUsedBy	*Object*	*Robot*
canUse	*Robot*	*Object*
isDifferent	*Robot*	*Robot*

First, the "isOccupiedBy" property represents a place that is occupied by the robot's movement and contains *Place* as the domain and *Robot* as the range. This object property is created for the robot's starting and destination places. The difference between this property and the "isLocatedAt" property, which represents the robot's location, is when it becomes active/inactive, as shown in Figure 1a. By defining the "isOccupiedBy" property, we can determine where the robot is working and utilize this knowledge to manage the robot's traffic. The "isNotOccupiedBy" property is defined to represent the negation of the "isOccupiedBy" property, which is created as opposed to the active/inactive of the "isOccupiedBy" property. Second, the "isUsedBy" property with the object is defined to represent the robot's possession knowledge. This property contains *Object* as the domain and *Robot* as the range and is used to avoid the redundant use of objects when planning tasks for the multi-robot. To represent the negation of this property, we define the "isNotUsedBy" property. In addition, the "canUse" property is defined to express whether a specific object can be used and is used as a constraint when planning multi-robot works. Finally, the "isDifferent" property is defined to describe the relationship between different robots and is used to ensure that all robots are considered in the detailed planning.

Additionally, low-resolution place information in the proposed hierarchical task planning structure is used to generate simplified plans. We introduce a new data property for place environment elements to achieve this. In the previous study by Joo et al. [17], the lowest-level place in the "isInsideOf" relationship graph is represented as a leaf place and used for robot planning. In contrast, we define the top-level place as a stem place and extend the data property with the "isStemPlace" property to represent it.

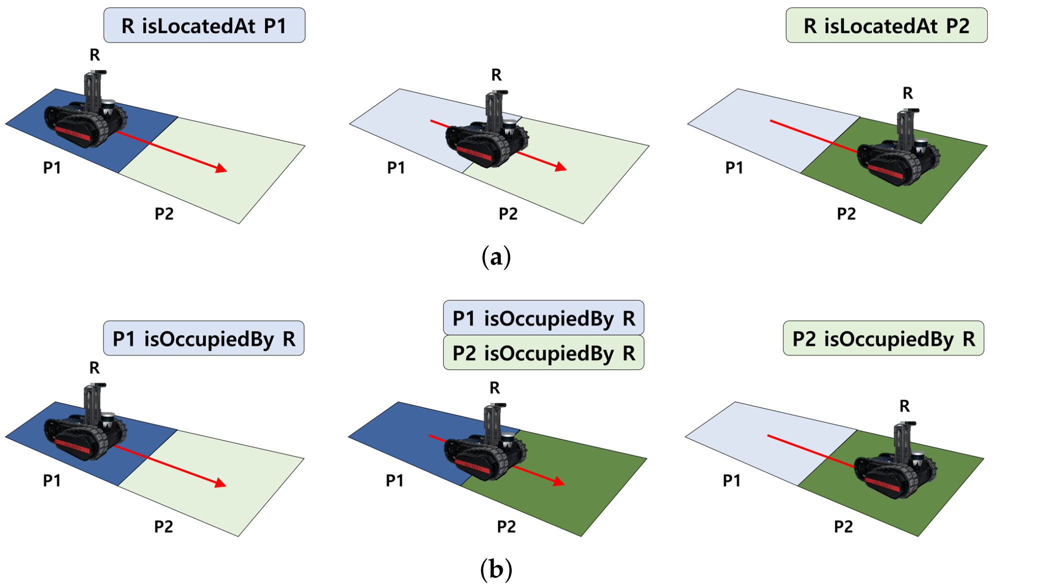


Figure 1. Comparison of when object properties activate and deactivate during the navigation. (**a**) When "isLocatedAt" property is activated/deactivated by the robot moving from place to place. (**b**) Activation/deactivation of "isOccupiedBy" property during navigation.

Figure 2 shows the relationship graph of the *Place* environmental elements in the semantic database, including the proposed stem place. The stem place is defined as the top-level *Place* element in the "isInsideOf" relationship graph that is not involved in any "isInsideOf" relationship with any other *Place* element. We can query the stem places in the semantic database using the following Semantic Query-Enhanced Web Rule Language (SQWRL) expression:

- Place(?p1) ∧ Place(?p2) ∧ isInsideOf(?p1, ?p2). sqwrl:makeSet(?s1, ?p1) ∧ sqwrl:makeSet(?s2, ?p2). sqwrl:difference(?s3, ?s2, ?s1) ∧ sqwrl:element(?p, ?s3) $\longrightarrow$ sqwrl:select(?p).

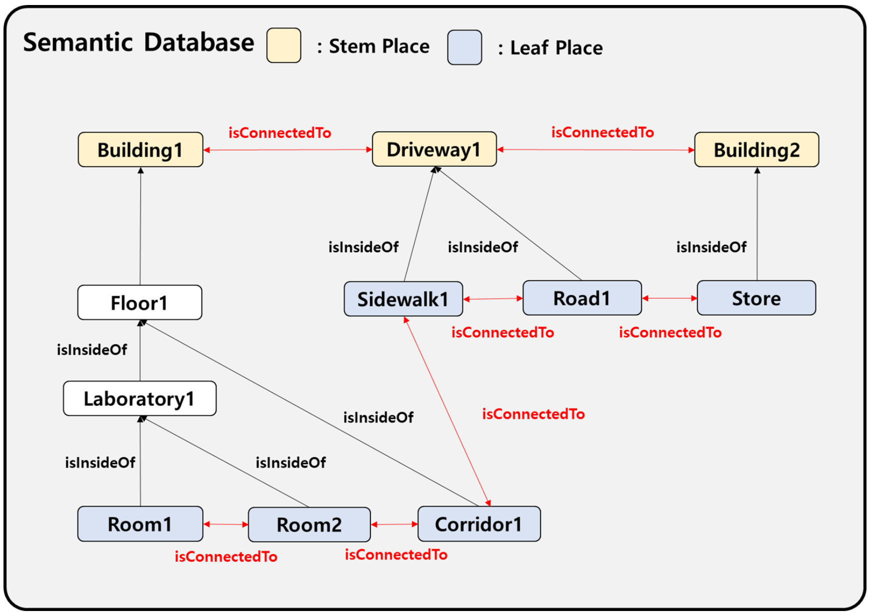


Figure 2. Hierarchical relationships of places in the semantic database.

This expression retrieves the top-level *Place* environmental element by comparing the places in the semantic database that are involved in the "isInsideOf" property with the group corresponding to the domain and the group corresponding to the range. As shown in Figure 2, we can utilize this expression to query Building1, which includes rooms, a corridor, and a floor in the "isInsideOf" property.

3.1.2. Semantic Knowledge-Based Task Modeling

The HTN structure is used to decompose tasks for efficient task planning in the semantic knowledge-based multi-robot system. The unit tasks are organized to accomplish given missions into a 3-level hierarchy. The tasks in each level are defined in a domain file using the PDDL. Table 3 lists each level of the proposed multi-robot task and its representative tasks. We represent the task name in italics to avoid word confusion.

Table 3. An example of the task list defined by the hierarchy.

Task Level Hierarchy	List of Tasks			
Mission level	Delivery	Guidance	Surveillance	Patrol
Coarse level	Move	MoveSame	Take	Give
Fine level	GoToPlace	Charge	PickUp	Dropoff

First, the mission level in the proposed hierarchical task modeling structure contains tasks that correspond to missions given to the multi-robot system. As shown in Table 3, the missions assigned to robots commonly include *Delivery*, *Guidance*, *Surveillance*, and *Patrol*.

Second, the coarse level consists of lower-level actions designed to accomplish the tasks of the mission level. Typically, missions assigned to multi-robots involve movement and interaction with objects. As a result, we define tasks for *Move* and *MoveSame* in the coarse level to describe movement and tasks for *Take* and *Give* to describe the interactions with objects. The planner utilizes these tasks to plan the multi-robot mission in the stem place, identifying the areas where the robots will perform their missions and the dependent robots in the multi-robot system.

Finally, the fine level comprises the detailed unit tasks that robots execute to carry out the coarse plan, composed of coarse-level tasks. There are numerous fine-level tasks, depending on the robot's characteristics and the interactive environment. As representative fine-level tasks, we introduce the following: *GoToPlace*, *Charge*, *PickUp*, and *DropOff*.

The tasks defined in the mission level correspond to the tasks given to the robot system and accomplish the goal defined in the problem file. For instance, the *Deliver* task aims to transport a specific object to a destination. This task is implemented in the PDDL to achieve the goal "delivery ?o ?p" when a robot with the specific object present at the destination satisfies the required preconditions, as specified in Listing 1.

Listing 1. Defined mission-level action *Delivery* in the domain.

```
(:action delivery
:parameters (?r - robot ?o - object ?p - place)
:precondition (and
    (is_used_by ?o ?r)
    (is_located_at ?r ?to))
:effect (and
    (delivery_complete ?o ?p)))
```

Two actions are defined to represent movement in the coarse-level tasks. This is to identify and group dependencies between robots that work in parallel in the multi-robot system. Therefore, we add a time element to the movement task to determine which robots are in the same place at the same time. For this purpose, move actions are implemented as "duration-actions" in the PDDL, as described in Listings 2 and 3. The time taken for both move actions is calculated as the distance divided by the speed, as shown in the 3rd line of each listing, and expressed as a duration item. This defined action implements an action at the coarse level that operates on a stem place by checking the "isStemPlace" property of the place as a condition to be executed.

Listing 2. Defined coarse-level action *Move* in the domain.

```
(:durative-action move
:parameters (?r - robot ?from ?to - place)
:duration ( = ?duration (/ (distance ?from ?to)(speed ?r)))
:condition (and
    (over all (is_connected_to ?from ?to))
    (over all (can_go_through ?r ?to))
    (over all (is_stem_place ?from))
    (over all (is_stem_place ?to))
    (at start (is_located_at ?r ?from))
    (at start (forall (?r2 - robot)
        (imply (is_different ?r ?r2)(is_not_occupied_by ?to ?r2)))))
:effect (and
    (at start (not (is_located_at ?r ?from)))
    (at end (is_located_at ?r ?to))
    (at start (is_occupied_by ?to ?r))
    (at start (not (is_not_occupied_by ?to ?r)))
    (at end (not (is_occupied_by ?from ?r)))))
```

Listing 3. Defined coarse-level action *MoveSame* in the domain.

```
(:durative-action move_same
:parameters (?r - robot ?from ?to - place)
:duration ( = ?duration (/ (distance ?from ?to)(speed ?r)))
:condition (and
    (over all (is_connected_to ?from ?to))
    (over all (can_go_through ?r ?to))
    (over all (is_stem_place ?from))
    (over all (is_stem_place ?to))
    (at start (is_located_at ?r ?from)))
:effect (and
    (at start (not (is_located_at ?r ?from)))
    (at end (is_located_at ?r ?to))
    (at start (is_occupied_by ?to ?r))
    (at start (not (is_not_occupied_by ?to ?r)))
    (at end (not (is_occupied_by ?from ?r)))
    (at end (is_not_occupied_by ?from ?r))
    (at start (is_occupied_by ?to ?r))))
```

The *Move* task is defined for a robot to navigate to a location alone. It utilizes the "isOccupiedBy" and the "isNotOccupiedBy" properties to check for overlapping multi-robot moves, as shown in lines 9 and 10 of Listing 2. It utilizes the "forall" function to check the knowledge that "all other robots are not occupying the target place" for the place where it wants to move. On the other hand, the *MoveSame* task implements a robot's movement despite the robot's presence at the target place. This is the same as the *Move* action with some conditions removed, such as Listing 3. We prioritize the *Move* task over the *MoveSame* task to ensure efficient mission planning. The *Move* task is planned when a robot moves alone. However, if another robot occupies the target place, the task planner plans the *MoveSame* task instead. This *MoveSame* task is used in the planning to identify and group robots working in the same place at the same time. These two defined domain actions activate/deactivate the "isOccupiedBy" and the "isLocatedAt" properties.

In the coarse-level tasks of a multi-robot system, we have defined *Take* and *Give* tasks for a robot to interact with an object. Listing 4 represents the PDDL action that defines the *Take* task. This action creates knowledge of the interaction between the robot and the object when both exist in the same place and satisfy the property of "canUse ?r ?o." In contrast, the *Give* task in Listing 5 releases the robot from its relationship with the object it previously interacted with. These two tasks enable the robot and the objects in the place to interact.

Listing 4. Defined coarse-level action *Take* in the domain.

```
(:action take
:parameters (?r - robot ?o - object ?p - place)
:duration (= ?duration (take-time ?r))
:condition (and
    (over all (is_stem_place ?p))
    (over all (is_located_at ?r ?p))
    (at start (is_inside_of ?o ?p))
    (over all (can_use ?r ?o)))
:effect (and
    (at end (not (is_inside_of ?o ?p)))
    (at end (is_used_by ?o ?r))))
```

Listing 5. Defined coarse-level action *Give* in the domain.

```
(:action give
:parameters (?r - robot ?o - object ?p - place)
:duration (= ?duration (give-time ?r))
:condition (and
    (over all (is_stem_place ?p))
    (over all (is_located_at ?r ?p))
    (at start (is_used_by ?o ?r)))
:effect (and
    (at end (not (is_used_by ?o ?r)))
    (at end (is_inside_of ?o ?p))))
```

We use the fine-level tasks to refine a coarse plan that consists of coarse-level tasks for each robot. These tasks are executed individually at the leaf place, using the extended "isOccupiedBy" and "isUsedBy" properties. All tasks include a customizable duration item. For example, the *GoToPlace* task, shown in Listing 6, has its duration determined by the distance and speed. The task is to perform the behavior for a specified duration of time if the conditions are satisfied. At this time, we check whether other robots are at the location with the connection information of the place through the forall function to prevent the duplication and deadlock of multiple robots.

Listing 6. Defined fine-level action *GoToPlace* in the domain.

```
(:durative-action goto_place
:parameters (?r - robot ?from ?to - place)
:duration ( = ?duration (/ (distance ?from ?to)(speed ?r)))
:condition (and
    (over all (is_connected_to ?from ?to))
    (over all (can_go_through ?r ?to))
    (over all (is_leafplace ?from))
    (over all (is_leafplace ?to))
    (at start (is_located_at ?r ?from))
    (at start (forall (?r2 - robot)
        (imply (is_different ?r ?r2)(is_not_occupied_by ?to ?r2)))))
:effect (and
    (at start (not (is_located_at ?r ?from)))
    (at end (is_located_at ?r ?to))
    (at start (is_occupied_by ?to ?r))
    (at start (not (is_not_occupied_by ?to ?r)))
    (at end (not (is_occupied_by ?from ?r)))
    (at end (is_not_occupied_by ?from ?r))))
```

The time for each task can also be fixed to a constant value. For example, the *PickUp* tasks defined in Listing 7 use a robot-specific constant value to represent the time required to perform the task. This task checks to ensure that objects existing at the target location are used by other robots to avoid duplication. This helps to avoid creating a waiting task when planning it in the multi-robot planner. By defining these fine-level tasks, we can generate a plan that prevents deadlocks and overlapping situations in the robot's mission planning.

Based on the fine-level tasks defined above, various derived tasks can be designed depending on the type of robot and the task details. For example, if a new fine-level task is added to plan a mission involving an elevator, we can define a unit task called "GoToPlace-WithElevator" by adding the condition "can_use ?robot ?elevator" to the implemented *GoToPlace* task representation. Similarly, tasks can be implemented for crossing a crosswalk and going through an automatic door. Furthermore, various interaction actions can be derived from the *PickUp* and *DropOff* .

Listing 7. Defined fine-level action *PickUp* in the domain.

```
(:durative-action pick_up
:parameters (?r - robot ?o - object ?p - place)
:duration (= ?duration (pickup-time ?r))
:condition (and
    (over all (is_leafplace ?p))
    (at start (is_located_at ?r ?p))
    (at start (is_inside_of ?o ?p))
    (over all (can_use ?r ?o))
    (at start (forall (?r2 - robot)
        (imply (is_different ?r ?r2)(is_not_used_by ?o ?r2)))))
:effect (and
    (at start (not (is_inside_of ?o ?p)))
    (at start (is_used_by ?o ?r))
    (at start (not (is_not_used_by ?o ?r)))))
```

3.2. Semantic Knowledge-Based Task Planner for Multi-Robots

We describe a method for utilizing semantic knowledge to plan tasks for multi-robot systems. The proposed approach extends the task planner from the semantic navigation framework [17] to be suitable for multi-robots. Our task planner transforms a given task into a simplified problem and then generates a coarse plan of the area and robots required for its implementation. Then, the coarse plan is used to group the robots that affect each other's navigation. Finally, we generate a fine plan that utilizes the grouped robots. Through this process, complex missions given to the multi-robot system are efficiently planned. In addition, a re-planning method is presented that effectively handles mission failure situations using our proposed task planner.

3.2.1. Hierarchical Task Planning Approach for Multi-Robots

The structure of the semantic knowledge-based multi-robot task planning methods is shown in Figure 3. Figure 3a shows the structure of multi-robot task planning based on the task planner described in [17]. In this structure, the tasks are planned using a classical planner, a behavior sequence is delivered to each robot, and feedback is received according to the performance of the robot's behavior. However, this approach requires much computational power for multi-robot task planning because it utilizes all robots to perform the tasks. Therefore, we introduce a hierarchical task planner to overcome this limitation.

Our proposed hierarchical task planner is divided into four steps, goal generating, coarse planning, plan reconstructing, and fine planning, as shown in Figure 3b. Each step utilizes hierarchical tasks with integrated semantic knowledge to generate a plan. The key idea of this planning structure is how to group dependent robots by generating a coarse plan utilizing coarse-level tasks. We explain our task planning approach in detail by considering an example environment, as shown in Figure 4.

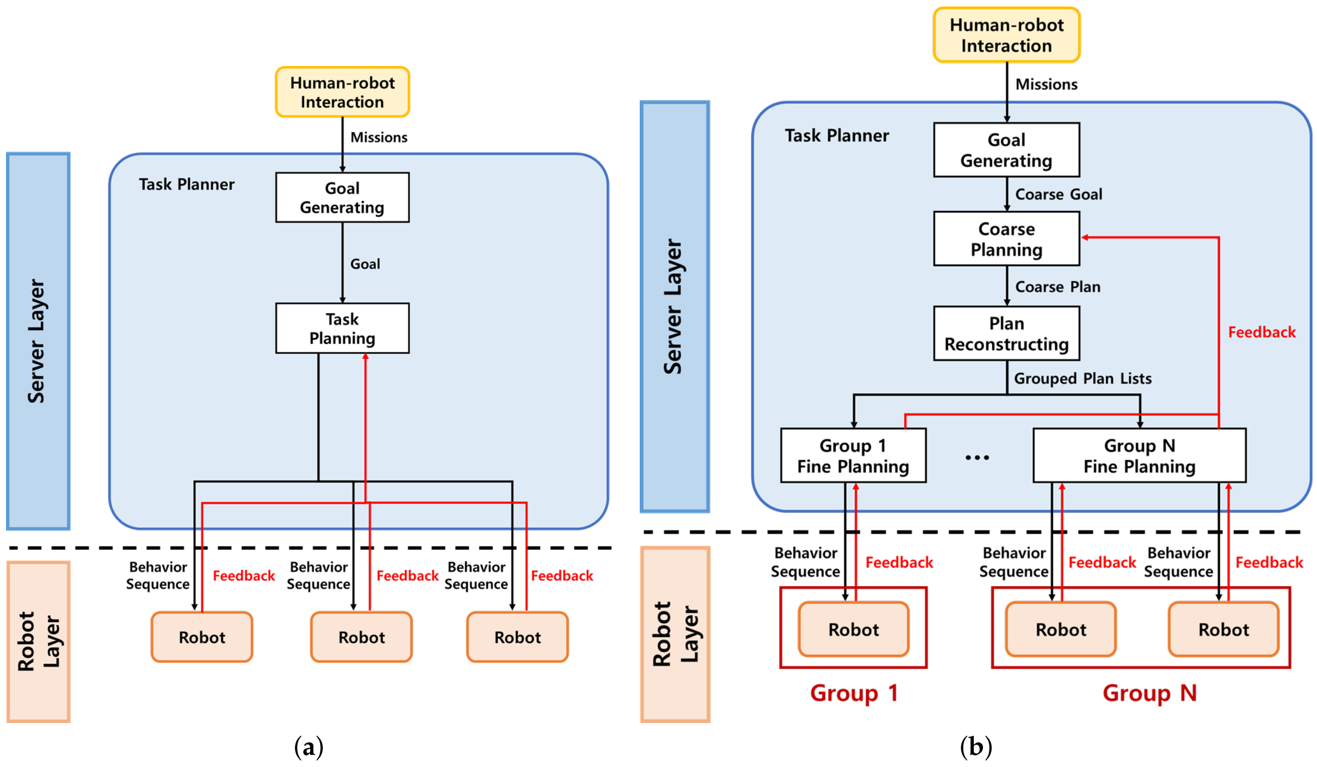


Figure 3. Comparison of semantic knowledge-based task planning structures for multi-robot. (**a**) Presented hierarchical task planning approach in [17]. (**b**) Proposed hierarchical task planning approach.

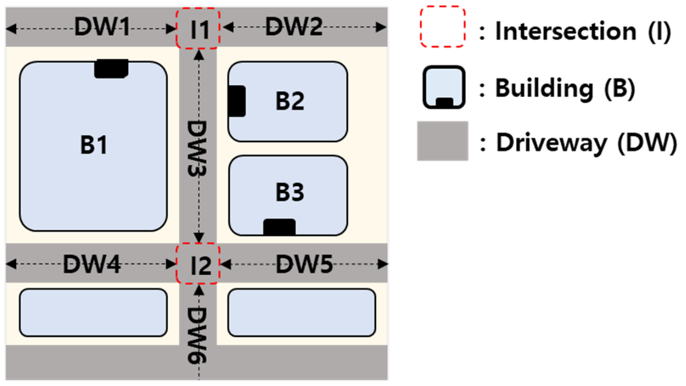
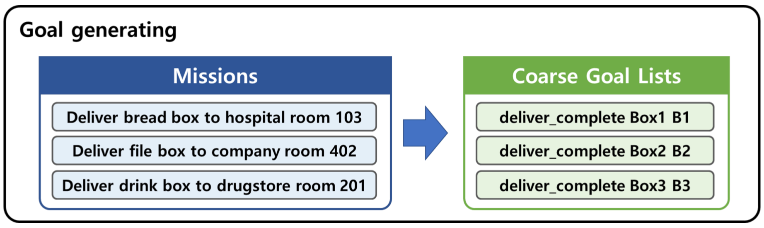


Figure 4. Example of multi-robot working outdoor environments.

First, the goal-generating step converts the knowledge generated by the effects of the mission-level tasks into problem goals so that the robot can understand human commands. At this time, we replace the *Place* parameter of the task with the stem place. Therefore, the goal-generating step queries the stem place information, including the target place, from the semantic database using the following SQWRL expression.

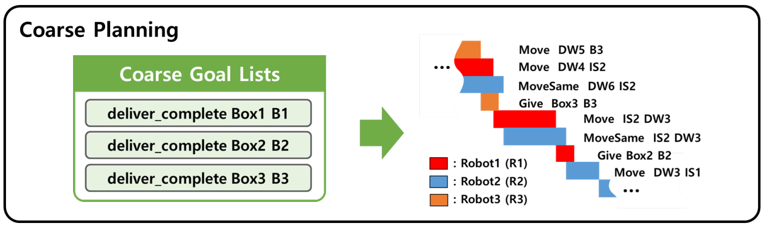
- Place(?p1) ∧ name(?p1, “Target Place”) ∧ isInsideOf(?p1, ?p2) ∧ isStemPlace(?p2, true) ⟶ sqwrl:select(?p2).

As shown in Figure 5a, if the delivery missions are given to the multi-robot task planner, the goal lists are created based on the corresponding mission-level delivery task. The example mission “Deliver bread box to hospital room 103” generates the coarse goal “delivery_complete Box1 B1” based on the effect of the *Delivery* task, “delivery_complete ?o ?p,” with a bread box (Box1) in the *Object* parameter and a hospital (B1), a stem place of the target place room 103, in the *Place* parameter. Finally, the goal-generating step utilizes all the given missions with the stem place and mission-level task to generate the coarse goal lists based on the PDDL.

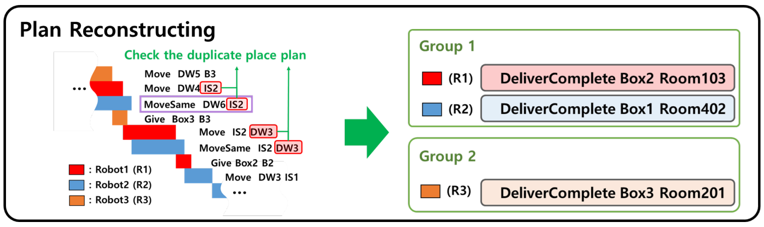


Goal generating
Missions
Deliver bread box to hospital room 103
Deliver file box to company room 402
Deliver drink box to drugstore room 201
Coarse Goal Lists
deliver_complete Box1 B1
deliver_complete Box2 B2
deliver_complete Box3 B3

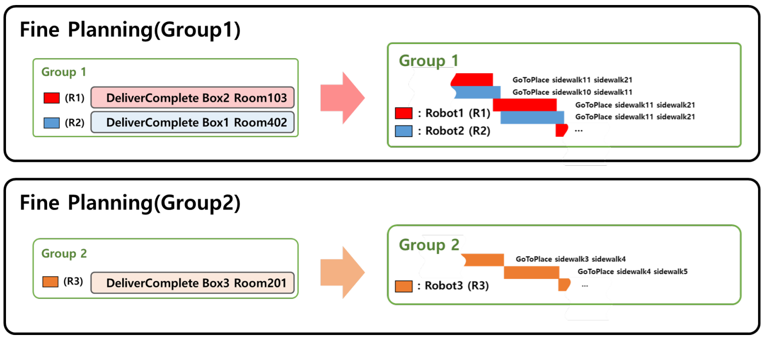
(**a**)



(**b**)



(**c**)



Fine Planning(Group1)
Group 1
(R1) DeliverComplete Box2 Room103
(R2) DeliverComplete Box1 Room402
GoToPlace sidewalk11 sidewalk21
GoToPlace sidewalk10 sidewalk11
GoToPlace sidewalk11 sidewalk21
GoToPlace sidewalk11 sidewalk21
: Robot1 (R1)
: Robot2 (R2)
Fine Planning(Group2)
Group 2
(R3) DeliverComplete Box3 Room201
GoToPlace sidewalk3 sidewalk4
GoToPlace sidewalk4 sidewalk5
: Robot3 (R3)

(**d**)

Figure 5. Four steps of the proposed hierarchical task planning approach. (**a**) Goal-generating step, which generates coarse goal lists with given missions. (**b**) Coarse planning step, which creates coarse plan lists utilizing all robots with coarse goal lists. (**c**) Plan reconstructing step, which groups overlapping robots in the coarse plan lists. (**d**) Fine planning step, which creates detailed behavior sequences for each group of robots.

Second, the coarse planning step generates the coarse-level task sequence by planning coarse goal lists using the POPF planner. For planning, the task planner queries the environment information from the semantic database to configure the on-demand DB. Importantly, because this step utilizes the coarse-level tasks, the planner only queries the stem places among the *Place* environment elements to construct the on-demand DB. Finally, the coarse planning step generates coarse task sequences based on the stem places, such as the driveway, intersection, and building, as shown in Figure 5b.

Third, the plan reconstructing step groups the robots that affect each other's navigation. In a multi-robot system, there may be overlapping situations between different robots moving in the same place at the same time. Therefore, we group the places where the *MoveSame* task occurred and the robots existing in the same place. In Figure 5c, the blue robot (R2) exists in the same place as the red robot (R1) when it executes the *MoveSame* operation indicated by the purple box. Therefore, the red and blue robots are organized into one group. In contrast, the orange robot (R3) does not overlap with any other robot, so we separate it into a single independent group. Finally, this step separates the robots with multi-robot missions into two groups.

Finally, the fine planning step plans the corresponding tasks per group using the POPF planner based on the fine-level tasks. For planning, each group's goal reconfigures the *Place* parameter to the leaf place, such as "delivery_complete Box1 room103." The idea of this step is to organize the on-demand database by querying only the information of the stem places where the tasks are performed. So, we only query the leaf places that are included as the "inInsideOf" property to the stem places used in the coarse plan result generated for each group. Then, as shown in Figure 5d, the task planner generates a fine-level task sequence to accomplish the assigned task for each group. This generates a detailed task plan based on leaf places, such as a sidewalk, and a crosswalk, as shown in Figure 6.

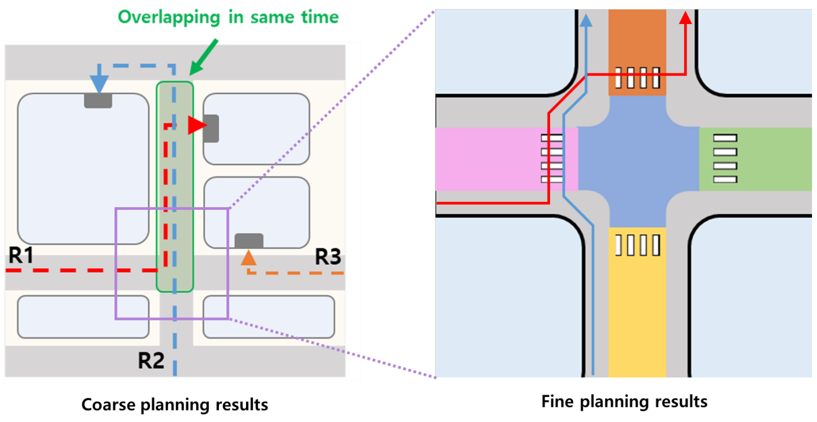


Figure 6. Comparison of coarse and fine planning results.

3.2.2. Hierarchical Task Re-Planning Approach for Multi-Robots

Multi-robot systems can respond efficiently to mission failure situations. The re-planning method using semantic knowledge introduced in [17] reconstructs a plan after updating the current situation information if an error occurs during the task execution. However, as shown in Figure 7, this scheme reconstructs the task plan utilizing all robots when a problem occurs in a multi-robot system. If one robot's path is blocked and creates a simple detour, the planner will re-plan using all robots. This may be an inefficient re-planning approach because it includes other robots that are executing their tasks well. Therefore, based on the proposed task planning structure, we propose the multi-robot re-planning approach by problem situation.

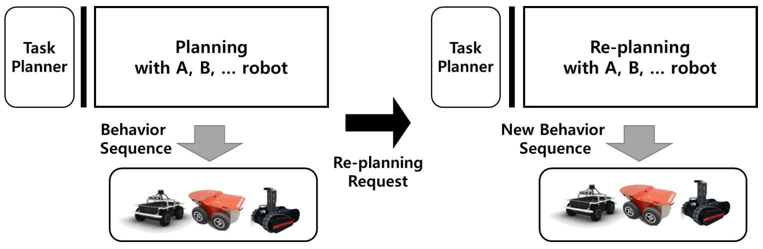
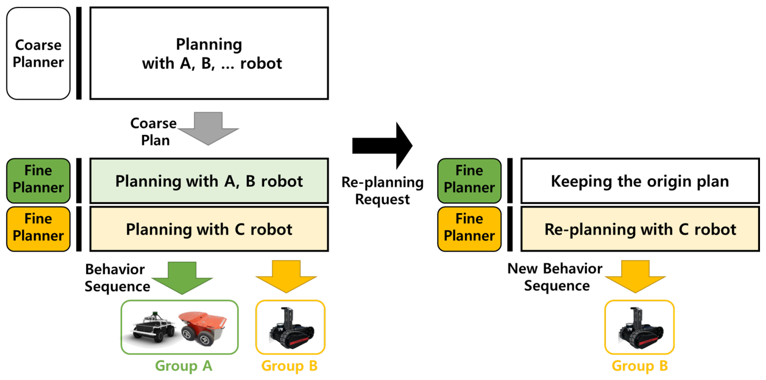
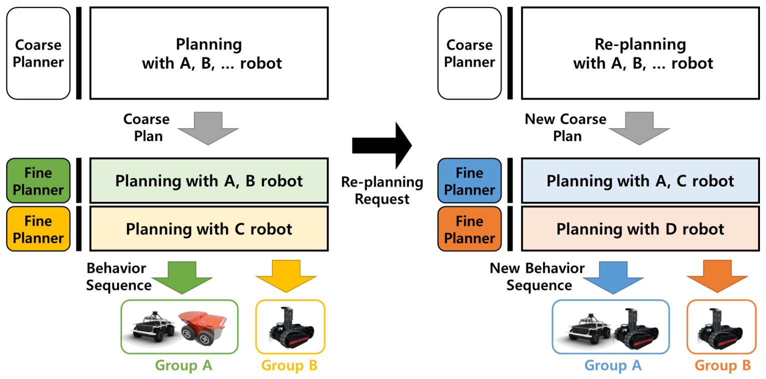


Figure 7. Re-planning method in previous method [17].

We obtain the results of each robot's execution feedback to the server. This allows the planner to check whether the robot performs successfully or if something goes wrong. If our task planner handles a simple problem, only grouped robots are re-planned in the plan reconstruction step, as shown in Figure 8a. Therefore, the planner keeps the other robot groups and re-plans only the problematic group to respond to the problem quickly. If a serious problem occurs, such as a robot breakdown, we consider the state of the multi-robot system and the remaining tasks, as shown in Figure 8b. In this way, the proposed task planner appropriately responds according to the criticality of the problem situation.



(**a**)



(**b**)

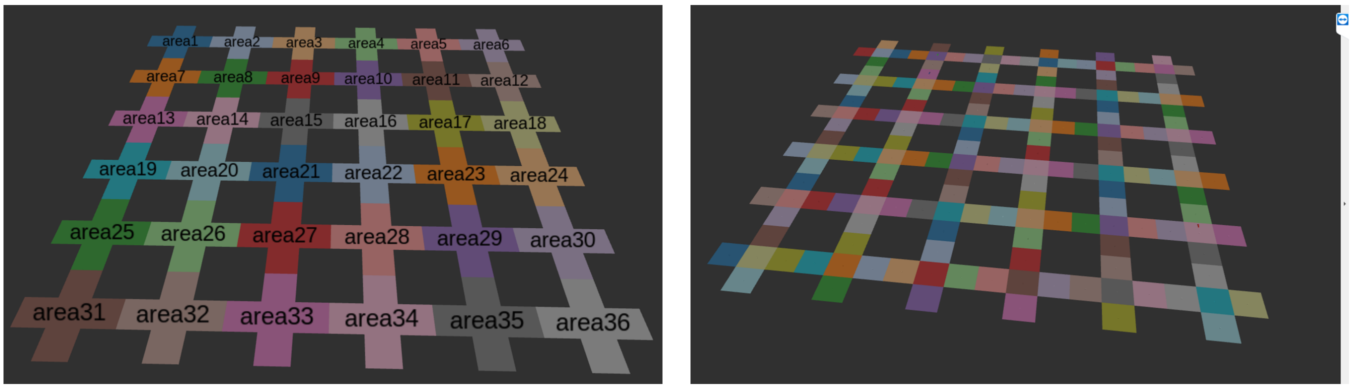
Figure 8. Re-planning method in the proposed approach. (**a**) Case of specific robot group re-planning. (**b**) Case of all robot re-planning.

4. Results

In this chapter, we validate the performance of our proposed method for multi-robot task planning by comparing it with the semantic navigation framework [17] method. For the comparison, three kinds of experiments were conducted in a simulation environment. The first experiment validates the suitability of the extended semantic knowledge by comparing the planning results for multi-robot intersection or overlap situations. The second experiment verifies the efficiency of the proposed task planning method by measuring the number of environmental factors utilized in task planning and the speed of task planning. The last experiment identifies the re-planning performance in response to problematic situations during the mission execution.

4.1. Experimental Environment

We verified the performance by implementing a simulation environment with various environmental elements and semantic knowledge, as shown in Figure 9. The simulation environment was organized in a 6 × 6 layout with a cross-shaped step place, as shown in Figure 9a. Each stem place in our environment consisted of five leaf places, as shown in Figure 9b. Therefore, the environment contained 252 place environment elements: 36 stem places and 180 leaf places. Every *Place* environment element was connected to its neighboring places with the "isConnectedTo" property, and each leaf place was connected to the stem place with the "isInsideOf" property.



(**a**) (**b**)

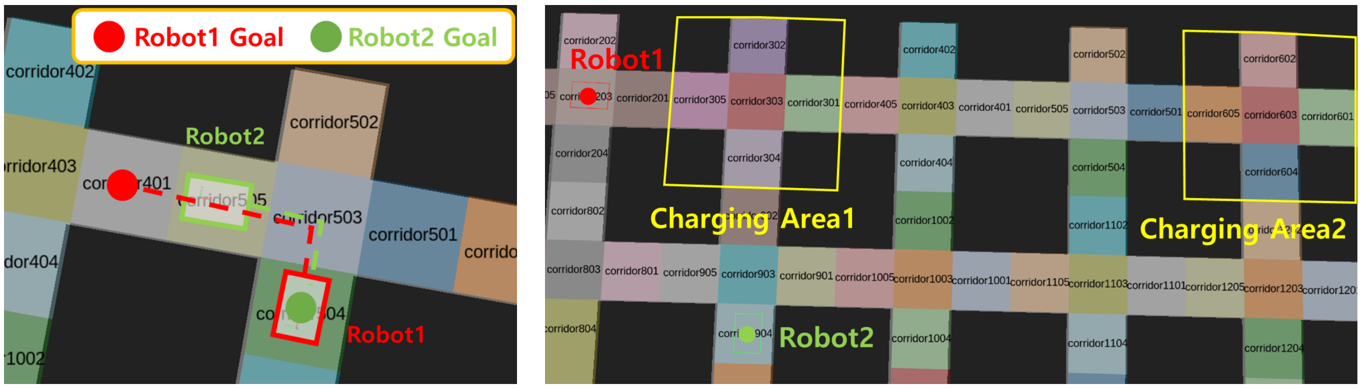
Figure 9. Simulationexperimental environment. (**a**) Cross-shaped stem place "area". (**b**) All 180 leaf places in the environment.

All the experiments were implemented utilizing the gazebo simulator on a computer equipped with an Intel(R) Core(TM) i9-9900KF CPU @ 3.60 GHz.

4.2. Experimental Scenario

4.2.1. Semantic Knowledge-Based Multi-Robot Planning

Two situations were considered to demonstrate what differences the extended semantic knowledge for multiple robots make to task planning, as shown in Figure 10. First, we compared the generated plans based on the proposed occupancy knowledge by implementing a situation where the robots cross each other. For comparison, a scenario was constructed where the target places of the robots intersected, as shown in Figure 10a. The generated sequence of tasks was compared to reach the destination and the plan execution results. Second, we compared the generated plans based on the proposed possession knowledge by implementing a situation where several robots use the charging station. For comparison, the environment was constructed with two robots and two charging areas, as shown in Figure 10b. The generated task plans were compared when short and long charging times were required in this environment.



(**a**) (**b**)

Figure 10. Semantic knowledge test scenarios. (**a**) Multi-robot intersection situation. (**b**) Multi-robot charging situation.

4.2.2. Semantic Knowledge-Based Multi-Robot Task Planning

The performance of the multi-robot mission planning methods was compared in a robot delivery mission situation under different conditions. The experiments were conducted for delivering three objects using two to four robots, as shown in Table 4. The number of robots and objects and the delivery locations in the table were randomly picked. We measured and compared the generation time of the plan to deliver the objects and the number of environmental elements used in the plan to verify the performance of the proposed method.

Table 4. Multi-robot delivery task conditions.

		Located Corridor ID									
		Robot				**Object**			**Target Place**		
		1	**2**	**3**	**4**	**1**	**2**	**3**	**1**	**2**	**3**
Case 1	**1**	3405	2004	-	-	2802	2004	3005	403	3502	1201
	2	804	1902	-	-	1104	2805	3303	2002	3504	1105
	3	1003	805	-	-	1204	305	2401	603	1901	1702
	4	2504	401	-	-	2804	103	3302	3001	601	803
	5	2802	2303	-	-	3104	2301	1803	2403	1802	3302
	6	3203	3604	-	-	105	805	3101	1801	2601	1302
	7	3005	3602	-	-	2902	2304	102	1002	3102	1104
	8	2404	1405	-	-	3201	602	802	303	3505	103
Case 2	1	3003	203	802	-	2001	3302	604	2101	2403	1202
	2	3501	303	2103	-	2701	3601	104	3105	703	605
	3	904	605	1703	-	1804	503	2902	904	802	2602
	4	1905	803	1702	-	1202	2205	2101	1002	3002	1102
	5	2205	3603	604	-	2301	1005	2201	2602	602	705
	6	2305	1804	3401	-	1502	1103	2203	1705	101	1904
	7	1904	2304	3203	-	3403	405	3102	2105	901	1104
	8	502	805	3105	-	2401	2101	603	2901	1002	1704
Case 3	1	3205	1504	1005	2404	1201	1505	2801	801	2902	2402
	2	3203	2004	305	202	3303	301	2601	502	2103	204
	3	2201	1501	1305	2101	3601	802	903	2802	1602	2403
	4	2702	202	102	302	3404	1604	703	2201	3204	3601
	5	504	2903	2005	3105	2204	1104	1405	3105	604	2205
	6	3201	1105	605	1303	1102	1205	2304	2502	904	1403
	7	2902	804	701	3403	1105	2402	1004	1604	804	2101
	8	1702	2703	305	701	3501	2305	3002	405	1202	2704

Figure 11 shows an example of three robots delivering three items. The circle shapes are the locations of the items, and the squares are the delivery locations. In this case, different colors were used for the *Object* location and the target place of each *Object* in the environment to make it easier to understand the tasks. In the simulation, we assumed that the *PickUp* task is completed after a while when the robot arrives at the *Object* location, and the *DropOff* task is completed after a while when the robot reaches the delivery location.

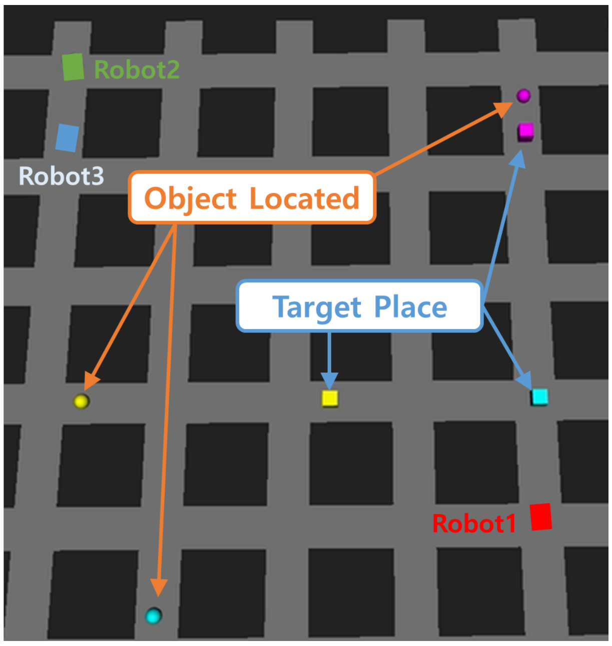


Figure 11. Deliver task environments.

4.2.3. Semantic Knowledge-Based Multi-Robot Task Re-Planning

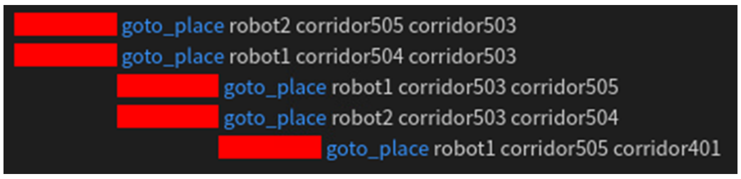
We verified the mission re-planning performance by comparing the re-planning time for robot failure. For this, the mission failure situations of a specific robot were assumed during a delivery mission with three robots. The mission failure situation included a path-blocked situation and the breakdown of a robot. The mission failure scenarios applied in this experiment are when the robot's path is blocked and when the robot breaks down. This situation was made by forcing a mission failure signal on a specific robot during a multi-robot mission.

4.3. Experimental Results

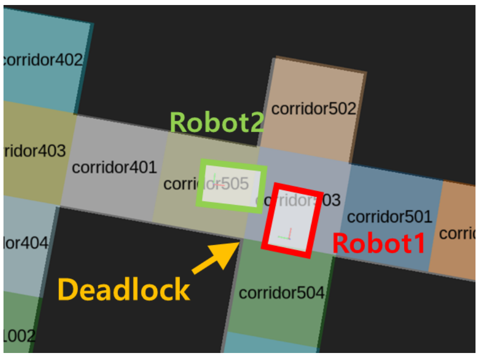
4.3.1. Semantic Knowledge-Based Multi-Robot Planning

Figure 12a shows the planned task sequence without the occupancy knowledge in a multi-robot cross-driving situation. The planner without semantic knowledge generated parallel robot tasks with the shortest execution time for each robot to complete its goal. However, it did not consider the positions of other robots, so when the plan was executed the robots got into a deadlock with each other, as shown in Figure 12b.

However, our proposed semantic planner produced a more complex plan for the same task, as shown in Figure 13a, because our method utilizes semantic knowledge to represent the occupancy of a space. Thus, when we executed the plan result, green *robot2* moved aside and red *robot1* passed, avoiding the deadlock between the robots, as shown in Figure 13b.

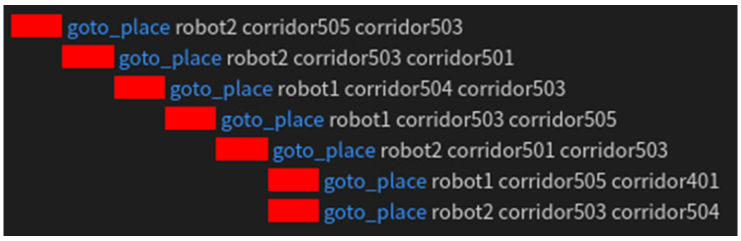

```
goto_place robot2 corridor505 corridor503
goto_place robot1 corridor504 corridor503
goto_place robot1 corridor503 corridor505
goto_place robot2 corridor503 corridor504
goto_place robot1 corridor505 corridor401
```

(**a**)

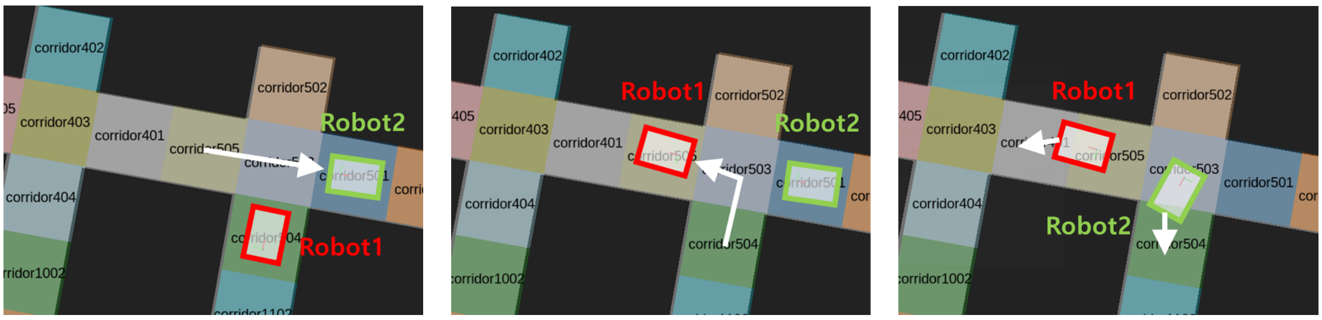


(**b**)

Figure 12. Multi-robot planning results without knowledge of "occupancy". (**a**) Planning result for multi-robot. (**b**) Task implementing result with multi-robot.


```
goto_place robot2 corridor505 corridor503
goto_place robot2 corridor503 corridor501
goto_place robot1 corridor504 corridor503
goto_place robot1 corridor503 corridor505
goto_place robot2 corridor501 corridor503
goto_place robot1 corridor505 corridor401
goto_place robot2 corridor503 corridor504
```

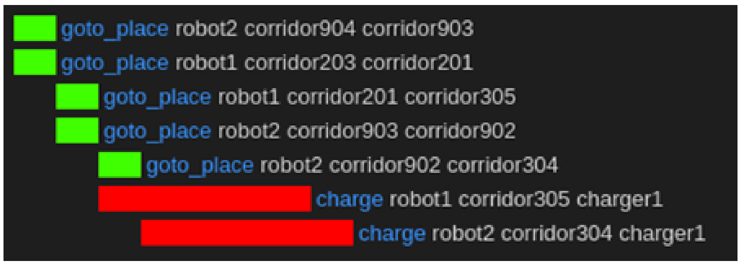
(**a**)



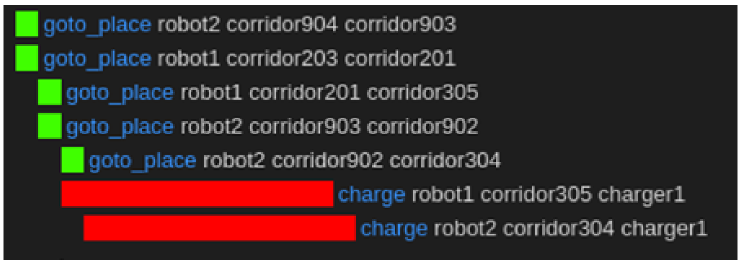
(**b**)

Figure 13. Multi-robot planning results with knowledge of "occupancy". (**a**) Planning result for multi-robot. (**b**) Task implementing result with multi-robot.

In the case of the charging task, the result of the task planning without the semantic knowledge of "usage" is shown in Figure 14. In the previous method, when a short time is required for charging, the task sequence was planned for each robot to navigate to the nearest charging station and charge, as shown in Figure 14a. This plan did not consider that the other robots had finished charging. Therefore, the late robots waited when this plan was executed at the charging station. If a longer time is required for charging, the task sequence for charging is planned similarly, as shown in Figure 14b.


goto_place robot2 corridor904 corridor903
goto_place robot1 corridor203 corridor201
goto_place robot1 corridor201 corridor305
goto_place robot2 corridor903 corridor902
goto_place robot2 corridor902 corridor304
charge robot1 corridor305 charger1
charge robot2 corridor304 charger1

(**a**)

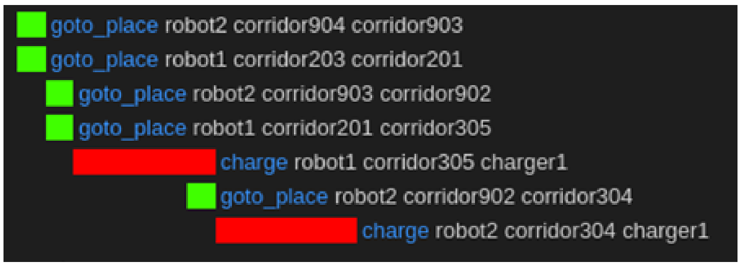

goto_place robot2 corridor904 corridor903
goto_place robot1 corridor203 corridor201
goto_place robot1 corridor201 corridor305
goto_place robot2 corridor903 corridor902
goto_place robot2 corridor902 corridor304
charge robot1 corridor305 charger1
charge robot2 corridor304 charger1

(**b**)

Figure 14. Planned behavior sequences without knowledge of "usage". (**a**) Planned sequence with short charging time. (**b**) Planned sequence with long charging time.

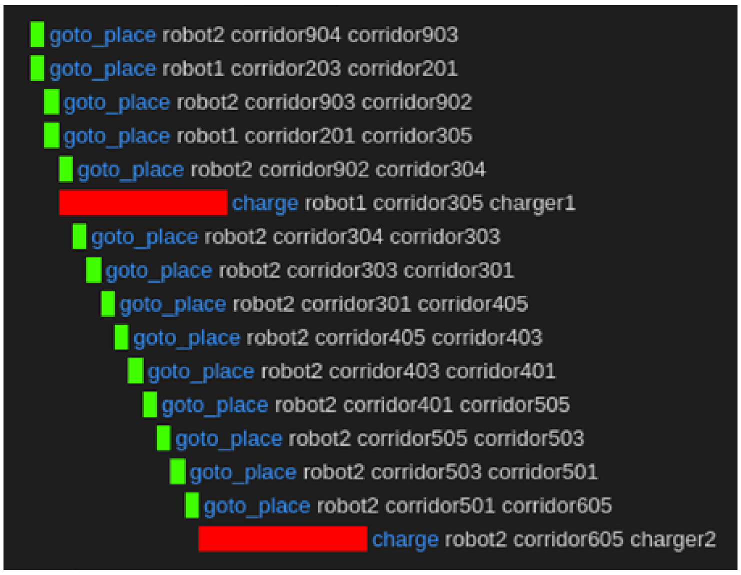
However, the proposed method utilized "usage" knowledge to plan, as shown in Figure 15. The planner generates a plan when charging takes a short time, as shown in Figure 15a. This plan consisted of a sequence in which robot2 starts charging when robot1 finishes using the charger. Because of the "isUsedBy" property of the charger, the robot took over the permission by planning the action of waiting. Therefore, when charging took a long time, the proposed method showed different results, as shown in Figure 15b. When charging at another charging station takes less time than a plan that includes waiting at a close charging station, our method generated a plan to visit the other charging station even if the distance was far. As a result, our task planner generated a plan where robot1 uses Charger1 and robot2 uses Charger2.

Through this experiment, we observed that utilizing the extended object properties detected the deadlock and redundancy situations of the robots and generated plans to prevent them. The generated plan sequence had a longer execution time, but it was organized in a sequence that ensured a trouble-free operation of the multi-robot system. Thus, our results showed that the proposed semantic knowledge helps to organize multi-robot plans in detail and avoid problematic situations.


goto_place robot2 corridor904 corridor903
goto_place robot1 corridor203 corridor201
goto_place robot2 corridor903 corridor902
goto_place robot1 corridor201 corridor305
charge robot1 corridor305 charger1
goto_place robot2 corridor902 corridor304
charge robot2 corridor304 charger1

(**a**)

Figure 15. *Cont.*



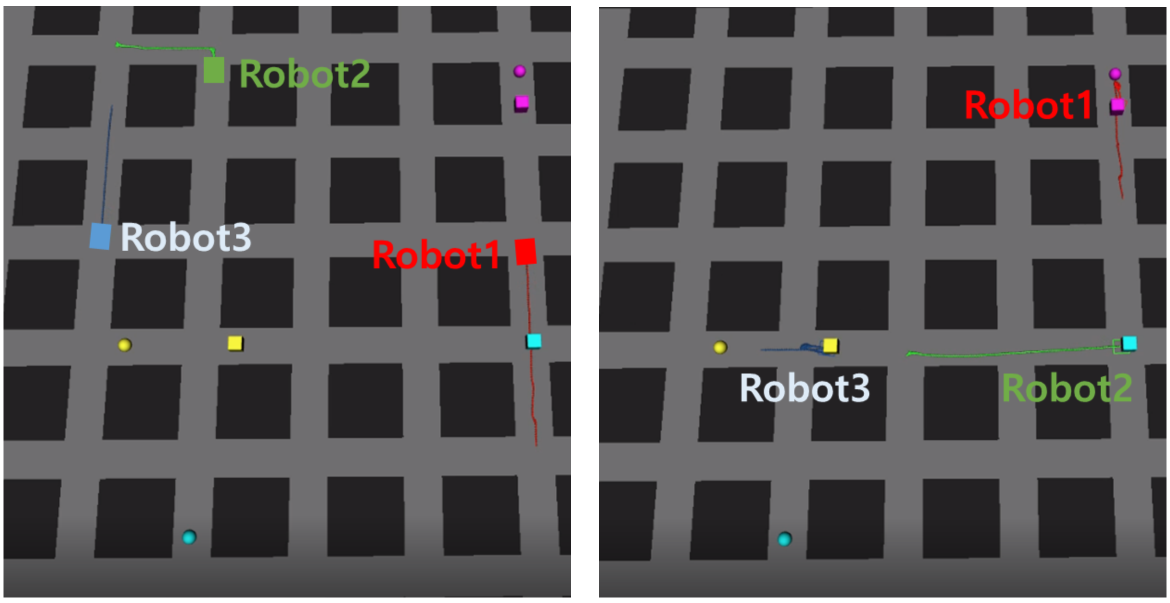
```
goto_place robot2 corridor904 corridor903
goto_place robot1 corridor203 corridor201
goto_place robot2 corridor903 corridor902
goto_place robot1 corridor201 corridor305
goto_place robot2 corridor902 corridor304
charge robot1 corridor305 charger1
goto_place robot2 corridor304 corridor303
goto_place robot2 corridor303 corridor301
goto_place robot2 corridor301 corridor405
goto_place robot2 corridor405 corridor403
goto_place robot2 corridor403 corridor401
goto_place robot2 corridor401 corridor505
goto_place robot2 corridor505 corridor503
goto_place robot2 corridor503 corridor501
goto_place robot2 corridor501 corridor605
charge robot2 corridor605 charger2
```

(**b**)

Figure 15. Planned behavior sequences with knowledge of "usage". (**a**) Planned sequence with short charging time. (**b**) Planned sequence with long charging time.

4.3.2. Semantic Knowledge-Based Multi-Robot Task Planning

Figure 16 is an example of a task plan with three robots generated using the previous and proposed methods. As shown in Figure 16a, each robot moves at the same time from the assigned corridor to deliver the items allocated to each robot. After navigating to the place where each item is located and waiting for a period of time, the robot moves to the target place. Thus, as shown in Figure 16b, the robot's mission ends when each robot has completed its assigned task.



(**a**) (**b**)

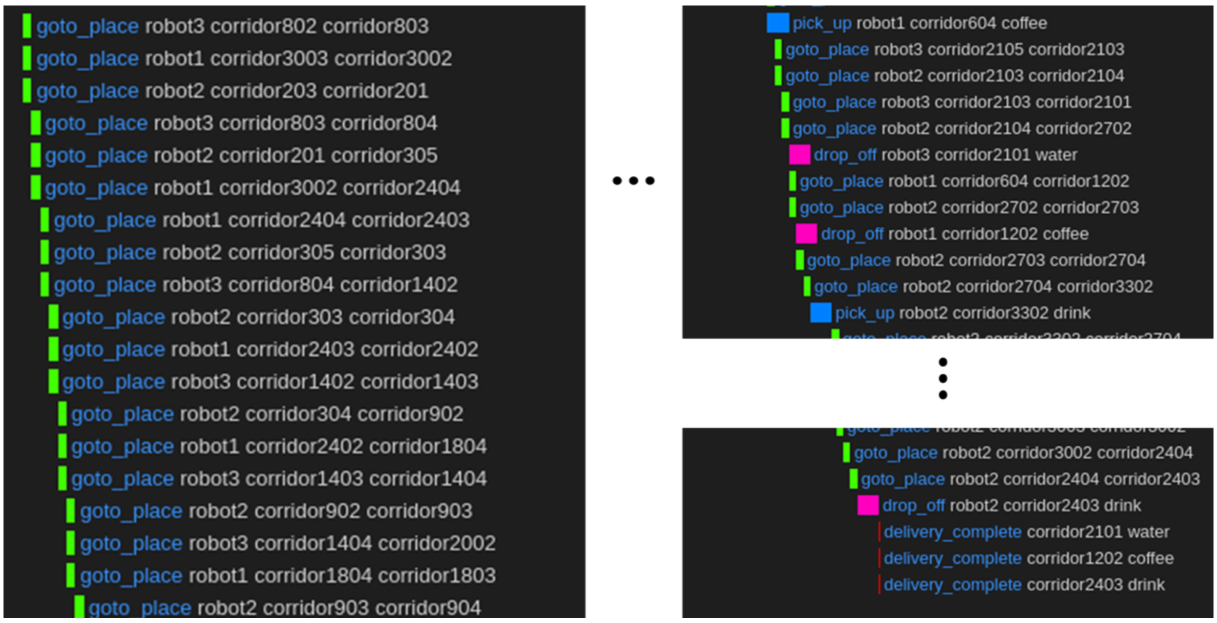
Figure 16. Multi-robot navigation based on planned task sequence. (**a**) Executing multi-robot delivery missions. (**b**) Completing multi-robot delivery missions.

When planning a multi-robot task, the previous task planner used all three robots to generate a task plan, as shown in Figure 17. However, the proposed task planner generates a task plan differently than before, as shown in Figure 18. Our task planner generates a coarse task plan based on the coarse-level hierarchy tasks that work in the stem place, as shown in Figure 18a. Then, the planner generated a fine plan for each robot group based on the generated coarse plan, as shown in Figure 18b,c. Both methods have the same object and destination delivered by the robots in the plan. However, there were differences in the methods used to generate those plans. The table below compares the number of place instances used in each method and the time to plan the task.

The proposed method measured the number of place elements used by summing the number of stem places and leaf places in the coarse planning step and the fine planning step, and the previous method measured the number of leaf places used. As shown in Table 5, the previous method planned using all 180 leaf places in the environment because it uses all places where the robot can explore. However, the proposed method only uses leaf places belonging to stem places, where the tasks are performed. So, each task used a different number of place elements. We can observe that the proposed method uses fewer place environmental elements than the previous method, as shown in Figure 19.

In addition, we compared the planning time of the previous method by summing the time spent in the coarse and fine steps. In this case, the plan reconstructing step of the proposed task planner was ignored in Table 5 because the time spent was less than 1 ms. We observed that the proposed method generates task plans faster than the previous method, as shown in the graph in Figure 20. When planning a task for multiple robots, the time spent by the previous method increased dramatically as the number of robots increased. Especially if the number of robots was increased to four, the planner failed to generate a plan because it exceeded the planning limit time of 3 minutes. However, the proposed method did not significantly increase the planning time even if the number of robots increased. Our method also successfully generated a plan when using four robots.

These results show the improved performance of our task planning approach. Although the proposed method produces the same mission plan result, the extended task planning structure uses fewer environmental elements. It also planned the multi-robot mission in less time than the previous method. If the task does not require referring to all the places, the proposed method will use less data to generate the plan than the previous method. These results prove that the proposed planning structure is an efficient approach to planning multi-robot system tasks even in environments with many environmental factors, such as urban centers and campuses.

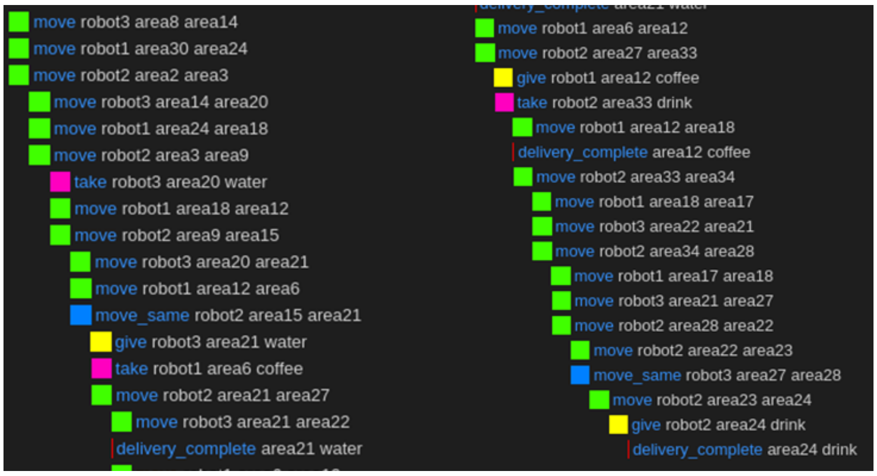


```
goto_place robot3 corridor802 corridor803
goto_place robot1 corridor3003 corridor3002
goto_place robot2 corridor203 corridor201
goto_place robot3 corridor803 corridor804
goto_place robot2 corridor201 corridor305
goto_place robot1 corridor3002 corridor2404
goto_place robot1 corridor2404 corridor2403
goto_place robot2 corridor305 corridor303
goto_place robot3 corridor804 corridor1402
goto_place robot2 corridor303 corridor304
goto_place robot1 corridor2403 corridor2402
goto_place robot3 corridor1402 corridor1403
goto_place robot2 corridor304 corridor902
goto_place robot1 corridor2402 corridor1804
goto_place robot3 corridor1403 corridor1404
goto_place robot2 corridor902 corridor903
goto_place robot3 corridor1404 corridor2002
goto_place robot1 corridor1804 corridor1803
goto_place robot2 corridor903 corridor904
```

```
pick_up robot1 corridor604 coffee
goto_place robot3 corridor2105 corridor2103
goto_place robot2 corridor2103 corridor2104
goto_place robot3 corridor2103 corridor2101
goto_place robot2 corridor2104 corridor2702
drop_off robot3 corridor2101 water
goto_place robot1 corridor604 corridor1202
goto_place robot2 corridor2702 corridor2703
drop_off robot1 corridor1202 coffee
goto_place robot2 corridor2703 corridor2704
goto_place robot2 corridor2704 corridor3302
pick_up robot2 corridor3302 drink
```

```
goto_place robot2 corridor3002 corridor2404
goto_place robot2 corridor2404 corridor2403
drop_off robot2 corridor2403 drink
delivery_complete corridor2101 water
delivery_complete corridor1202 coffee
delivery_complete corridor2403 drink
```

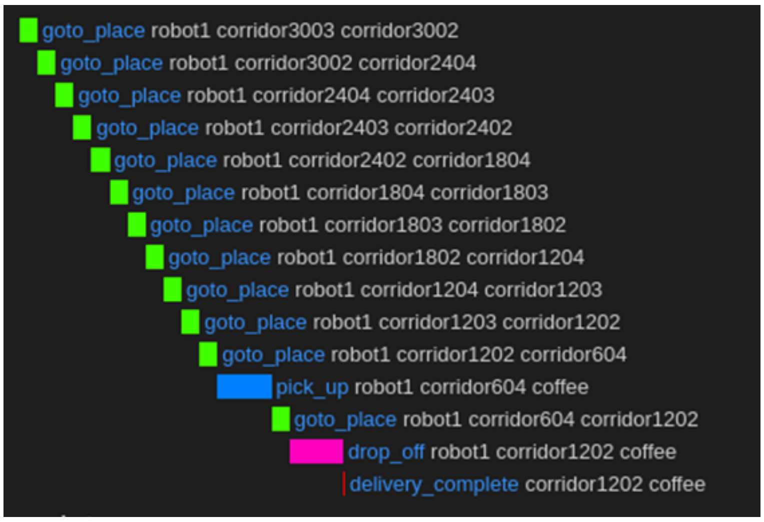
Figure 17. Planned task sequence with the previous method [17].



```
move robot3 area8 area14
move robot1 area30 area24
move robot2 area2 area3
  move robot3 area14 area20
  move robot1 area24 area18
  move robot2 area3 area9
    take robot3 area20 water
    move robot1 area18 area12
    move robot2 area9 area15
      move robot3 area20 area21
      move robot1 area12 area6
      move_same robot2 area15 area21
        give robot3 area21 water
        take robot1 area6 coffee
        move robot2 area21 area27
          move robot3 area21 area22
          delivery_complete area21 water
```

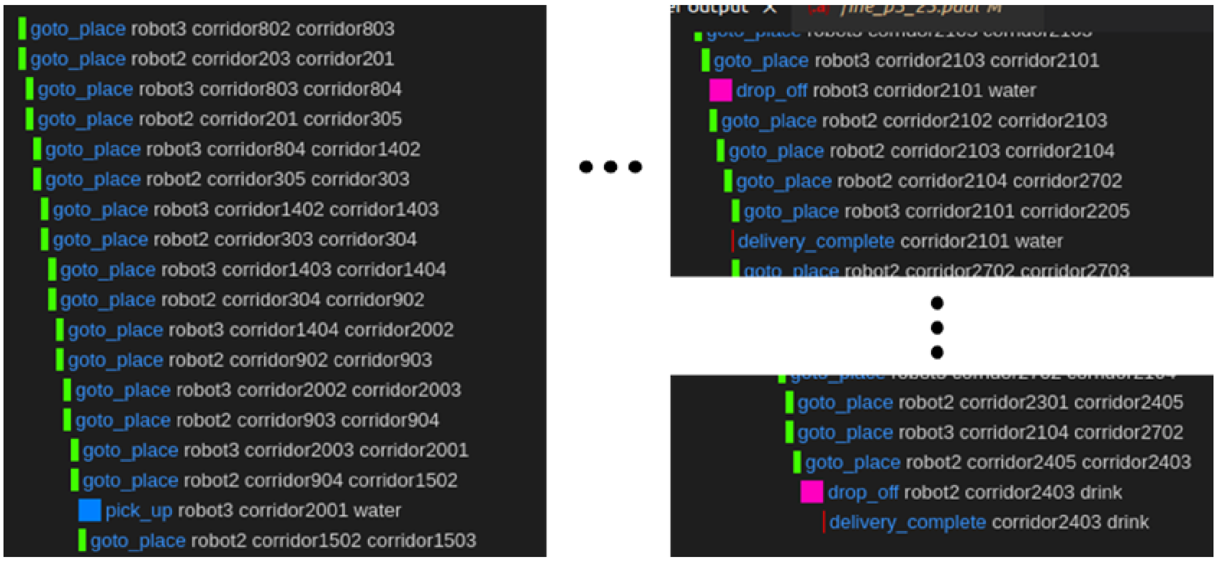
```
delivery_complete area21 water
move robot1 area6 area12
move robot2 area27 area33
  give robot1 area12 coffee
  take robot2 area33 drink
    move robot1 area12 area18
    delivery_complete area12 coffee
    move robot2 area33 area34
      move robot1 area18 area17
      move robot3 area22 area21
      move robot2 area34 area28
        move robot1 area17 area18
        move robot3 area21 area27
        move robot2 area28 area22
          move robot2 area22 area23
          move_same robot3 area27 area28
            move robot2 area23 area24
              give robot2 area24 drink
                delivery_complete area24 drink
```

(**a**)



```
goto_place robot1 corridor3003 corridor3002
 goto_place robot1 corridor3002 corridor2404
  goto_place robot1 corridor2404 corridor2403
   goto_place robot1 corridor2403 corridor2402
    goto_place robot1 corridor2402 corridor1804
     goto_place robot1 corridor1804 corridor1803
      goto_place robot1 corridor1803 corridor1802
       goto_place robot1 corridor1802 corridor1204
        goto_place robot1 corridor1204 corridor1203
         goto_place robot1 corridor1203 corridor1202
          goto_place robot1 corridor1202 corridor604
           pick_up robot1 corridor604 coffee
              goto_place robot1 corridor604 corridor1202
                 drop_off robot1 corridor1202 coffee
                    delivery_complete corridor1202 coffee
```

(**b**)



```
goto_place robot3 corridor802 corridor803
goto_place robot2 corridor203 corridor201
 goto_place robot3 corridor803 corridor804
 goto_place robot2 corridor201 corridor305
  goto_place robot3 corridor804 corridor1402
  goto_place robot2 corridor305 corridor303
   goto_place robot3 corridor1402 corridor1403
   goto_place robot2 corridor303 corridor304
    goto_place robot3 corridor1403 corridor1404
    goto_place robot2 corridor304 corridor902
     goto_place robot3 corridor1404 corridor2002
     goto_place robot2 corridor902 corridor903
      goto_place robot3 corridor2002 corridor2003
      goto_place robot2 corridor903 corridor904
       goto_place robot3 corridor2003 corridor2001
       goto_place robot2 corridor904 corridor1502
        pick_up robot3 corridor2001 water
        goto_place robot2 corridor1502 corridor1503
```

```
goto_place robot3 corridor2103 corridor2101
 drop_off robot3 corridor2101 water
 goto_place robot2 corridor2102 corridor2103
  goto_place robot2 corridor2103 corridor2104
   goto_place robot2 corridor2104 corridor2702
    goto_place robot3 corridor2101 corridor2205
    delivery_complete corridor2101 water
    goto_place robot2 corridor2702 corridor2703
```

```
goto_place robot2 corridor2301 corridor2405
goto_place robot3 corridor2104 corridor2702
 goto_place robot2 corridor2405 corridor2403
  drop_off robot2 corridor2403 drink
   delivery_complete corridor2403 drink
```

(**c**)

Figure 18. Planned task sequence with the proposed method. (**a**) Planned task sequence in coarse planning step. (**b**) Planned group1 task sequence in fine planning step. (**c**) Planned group2 task sequence in fine planning step.

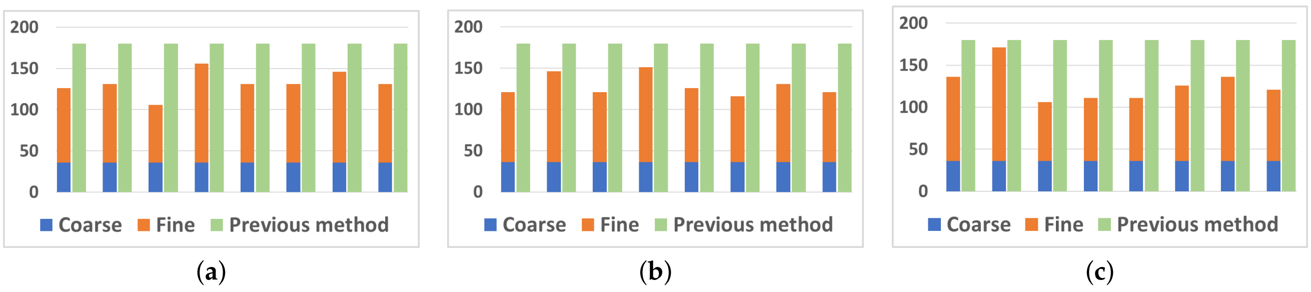


Figure 19. Graph of used *Place* instance quantity. (**a**) Test results with 2 robots. (**b**) Test results with 3 robots. (**c**) Test results with 4 robots.

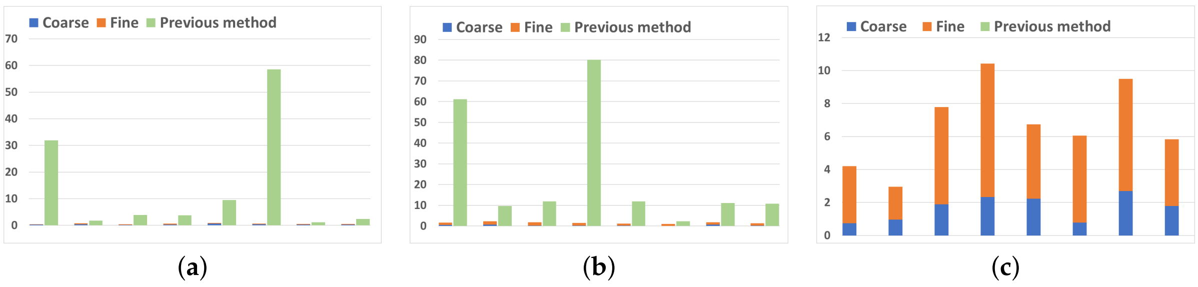


Figure 20. Graph of multi-robot task planning time. (**a**) Test results with 2 robots. (**b**) Test results with 3 robots. (**c**) Test results with 4 robots.

Table 5. Experimental results of multi-robot task planning.

		Quantity of Used Place Instance				Planning Time			
		Proposed Method			Previous Method	Proposed Method			Previous Method
		Coarse	Fine	Total		Coarse	Fine	Total	
Case 1	1	36	90	126	180	0.22	0.23	0.45	31.88
	2	36	95	131	180	0.34	0.48	0.82	1.8
	3	36	70	106	180	0.09	0.32	0.41	3.89
	4	36	120	156	180	0.33	0.36	0.69	3.7
	5	36	95	131	180	0.65	0.28	0.93	9.51
	6	36	95	131	180	0.37	0.3	0.67	58.52
	7	36	110	146	180	0.23	0.31	0.54	1.12
	8	36	95	131	180	0.24	0.27	0.51	2.42
Case 2	1	36	100	136	180	0.54	1.14	1.68	61.25
	2	36	135	171	180	0.74	1.51	2.25	9.71
	3	36	70	106	180	0.18	1.57	1.75	11.93
	4	36	75	111	180	0.16	1.25	1.41	80.23
	5	36	75	111	180	0.14	0.98	1.12	11.89
	6	36	90	126	180	0.11	0.91	1.02	2.34
	7	36	100	136	180	0.73	1.04	1.77	11.07
	8	36	85	121	180	0.24	1.12	1.36	10.7
Case 3	1	36	85	121	180	0.74	3.46	4.2	Time over
	2	36	110	146	180	0.95	2.01	2.96	Time over
	3	36	85	121	180	1.89	5.9	7.79	Time over
	4	36	115	151	180	2.32	8.1	10.42	Time over
	5	36	90	126	180	2.22	4.51	6.73	Time over
	6	36	80	116	180	0.77	5.29	6.06	Time over
	7	36	95	131	180	2.69	6.81	9.5	Time over
	8	36	85	121	180	1.79	4.05	5.84	Time over

4.3.3. Semantic Knowledge-Based Multi-Robot Task Re-Planning

Figure 21 shows an example of semantic knowledge-based multi-robot re-planning. As shown in Figure 21a, when the robot fails to execute a task due to external factors, the robot updates the current situation to the server and generates a plan to execute the remaining mission. In this case, the task planner re-plans and executes a path that avoids the blocked area, as shown in Figure 21b. This finally generates a plan to reach the goal, achieving the mission, as shown in Figure 21c.

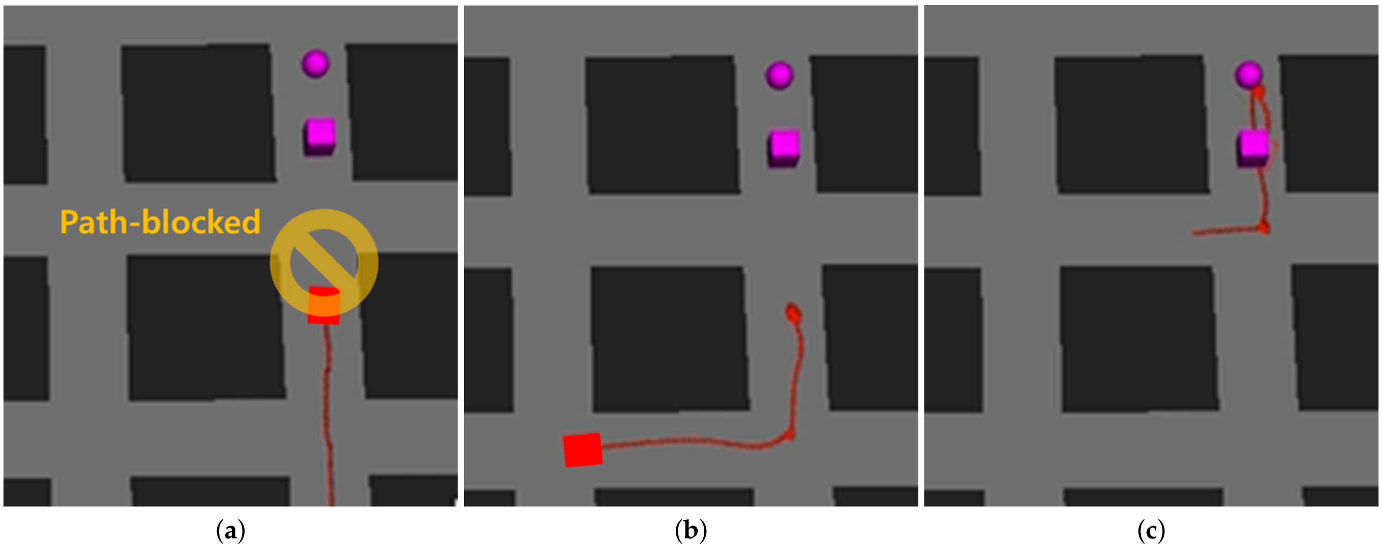


Figure 21. Re-planning in a mission failure case. (**a**) Path-blocked problem situation. (**b**) Driving a detour route. (**c**) Completing a delivery mission.

The previous method and our proposed method use different numbers of robots to generate these re-plans. The previous method considers the current state and remaining tasks of all three robots in the environment to generate a re-plan, as shown in Listing 8. However, the proposed method proceeds with re-planning only for the group currently in trouble, as shown in Listing 9. So, in this case, only one robot (robot1) belonging to the separated group is re-planned. Table 6 below shows the robots used in each re-planning method and their time for the two mission failure cases.

Table 6. Experimental results of multi-robot task re-planning.

	Trouble Case	Troubled Robot ID	Re-Planned Robot ID		Re-Planning Time			
			Proposed Method	Previous Method	Proposed Method			Previous Method
					Coarse	Fine	Total	
Case 1	Block path	1	1	1, 2, 3	-	0.31	0.31	38.93
		2	2, 3	1, 2, 3	-	0.46	0.46	32.21
		3	2, 3	1, 2, 3	-	0.54	0.54	31.32
	Robot breakdown	1	2, 3	2, 3	0.35	1.84	2.19	16.24
		2	1, 3	1, 3	0.18	2.20	2.38	25.71
		3	1, 2	1, 2	0.16	1.45	1.61	18.95
Case 2	Block path	1	1, 2, 3	1, 2, 3	0.65	0.62	1.27	8.65
		2	1, 2, 3	1, 2, 3	0.57	0.65	1.22	9.48
		3	1, 2, 3	1, 2, 3	0.61	0.48	1.09	8.41
	Robot breakdown	1	2, 3	2, 3	0.34	0.22	0.56	3.71
		2	1, 3	1, 3	0.25	0.37	0.62	2.80
		3	1, 2	1, 2	0.47	0.30	0.77	5.12

Table 6 compares the robots used and times taken for re-planning in the previous experiment with three robots, where each robot is given a mission failure situation. The previous method utilizes all the robots in the multi-robot system in every re-planning situation to construct a re-plan. As shown in Table 6, this approach utilizes all the robots, even for simple path-blocking problems. However, our proposed method only uses the group of robots that have failed in a simple path-blocking problem. In case 1, if robot 1 fails, only that robot will be re-planned, and if robots 2 or 3 fail, only that group of robots will be re-planned. In a more severe situation where a robot has failed, we re-plan using all remaining robots. In this way, our task planner distinguishes between multi-robot re-planning approaches based on the problem situation. If all robots need to be planned simultaneously, as in case 2 in Table 6, our task planner generates a plan that responds to the problem situation faster due to its efficient step-by-step task planning structure, even though the same robots are being re-planned as in previous methods. From the re-planning time shown in Table 6, we can see that the proposed task re-planning method outperforms the previous method by generating plans faster.

Listing 8. Re-planning result with the previous method.

```
0.000: (goto_place robot2 corridor902 corridor903) [3.340]
0.000: (goto_place robot3 corridor1404 corridor2002) [3.340]
0.000: (goto_place robot1 corridor1802 corridor1803) [3.340]
3.341: (goto_place robot2 corridor903 corridor904) [3.340]
3.341: (goto_place robot3 corridor2002 corridor2003) [3.340]
3.341: (goto_place robot1 corridor1803 corridor1805) [3.340]
6.682: (goto_place robot3 corridor2003 corridor2001) [3.340]
6.682: (goto_place robot1 corridor1805 corridor1701) [3.340]
6.682: (goto_place robot2 corridor904 corridor1502) [3.340]
10.023: (pick_up robot3 corridor2001 water) [10.000]
...
30.045: (drop_off robot3 corridor2101 water) [10.000]
30.069: (goto_place robot1 corridor1205 corridor1203) [3.340]
30.069: (goto_place robot2 corridor2702 corridor2703) [3.340]
33.410: (goto_place robot2 corridor2703 corridor2704) [3.340]
33.410: (goto_place robot1 corridor1203 corridor1202) [3.340]
36.751: (goto_place robot1 corridor1202 corridor604) [3.340]
36.751: (goto_place robot2 corridor2704 corridor3302) [3.340]
40.092: (pick_up robot1 corridor604 coffee) [10.000]
40.092: (pick_up robot2 corridor3302 drink) [10.000]
50.092: (goto_place robot1 corridor604 corridor1202) [3.340]
50.092: (goto_place robot2 corridor3302 corridor2704) [3.340]
53.432: (drop_off robot1 corridor1202 coffee) [10.000]
...
93.525: (goto_place robot2 corridor2404 corridor2403) [3.340]
96.865: (drop_off robot2 corridor2403 drink) [10.000]
106.866: (delivery_complete corridor2101 water) [0.001]
106.866: (delivery_complete corridor1202 coffee) [0.001]
106.866: (delivery_complete corridor2403 drink) [0.001]
```

Listing 9. Re-planning result with the proposed method.

```
0.000: (goto_place robot1 corridor1802 corridor1803) [3.340]
3.341: (goto_place robot1 corridor1803 corridor1805) [3.340]
...
33.410: (goto_place robot1 corridor1203 corridor1202) [3.340]
36.751: (goto_place robot1 corridor1202 corridor604) [3.340]
40.092: (pick_up robot1 corridor604 coffee) [10.000]
50.092: (goto_place robot1 corridor604 corridor1202) [3.340]
53.432: (drop_off robot1 corridor1202 coffee) [10.000]
63.433: (delivery_complete corridor1202 coffee) [0.001]
```

5. Conclusions and Implications

5.1. Summary of the Literature

This paper presented an extended semantic knowledge and hierarchical planning method for multi-robot systems. The proposed method defined the knowledge of possession and occupancy for multi-robot systems based on the TOSM; it also integrated the knowledge into the task model to define a high-level constraint task model; based on the defined content, the planner with a step-by-step planning structure generated efficient multi-robot task plans in complex environments; and the re-planning strategy for task failure situations effectively coped with the problem situations. The proposed method verified the validity of the semantic knowledge and the efficient task planning and re-planning performance through simulation experiments.

5.2. Theoretical Implications

The advancement of robots in the future will require the use of rules and cognitive information. The proposed method generates a task plan to perform given missions by understanding the semantic representation. It solves the problem that may occur in the multi-robot environment based on semantic representation rather than numerical representation so that the system does not generate incorrect plans. Our approach was the only one that understood knowledge, such as the influence between robots, the occupancy of places, and the possession of objects to generate task plans without problems. We believe that our approach will lead to future semantic knowledge-based robotic autonomous systems, which can be extended to humanoids that can embody human behavior or robots in complex outdoor urban environments.

5.3. Managerial Implications

Information such as cognition and rules are necessary to utilize robots precisely. However, the multi-robot system based on much semantic information has slow task planning performance. In that case, it is difficult to utilize robots due to the overall delay of the robot system and slow cyclical plan generation. From this viewpoint, the proposed task planning method has fast performance even when including the semantic knowledge of multiple robots, enabling the robot system to operate quickly and respond quickly to problem situations. Therefore, we think that the proposed task planning structure can be widely used in systems that manage multiple robots based on semantic knowledge.

5.4. Future Research Points

The proposed method has a few weaknesses in working in a real-world environment. Real-world environments may have irregularities and robots with different specifications, and task planning that responds to the changing environment in real time is also required. Nevertheless, our task planning structure can be extended in various ways, including semantic knowledge-based constraints. Thus, we can respond to this by adding additional definitions of environment and robot characteristics to the multi-robot mission model. Therefore, our future work focuses on extending the proposed multi-robot planning method to perform missions by cooperating with heterogeneous robots in a real-world environment.

Author Contributions: Conceptualization, S.B., S.J., and T.K.; methodology, S.B., S.J., and T.K.; software, S.B. and J.C.; validation, S.B., S.J., J.C., J.P., H.P., and T.K.; formal analysis, S.B., H.P., and T.K.; investigation, S.B.; resources, S.B., S.J., J.P., and T.K.; data curation, S.B.; writing—original draft preparation, S.B.; writing—review and editing, S.B.; visualization, S.B.; supervision, H.P. and T.K.; project administration, T.K.; funding acquisition, T.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Korea Evaluation Institute of Industrial Technology (KEIT), funded by the Ministry of Trade, Industry and Energy (MOTIE) (No. 1415178566).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Crosby, M.; Jonsson, A.; Rovatsos, M. A Single-Agent Approach to Multiagent Planning. In Proceedings of the ECAI, Prague, Czech Republic, 18–22 August 2014; pp. 237–242.
2. Parker, L.E. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Trans. Robot. Autom.* **1998**, *14*, 220–240. [CrossRef]
3. Mathew, N.; Smith, S.L.; Waslander, S.L. Planning paths for package delivery in heterogeneous multirobot teams. *IEEE Trans. Autom. Sci. Eng.* **2015**, *12*, 1298–1308. [CrossRef]
4. Khandelwal, P.; Barrett, S.; Stone, P. Leading the way: An efficient multi-robot guidance system. In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, Istanbul, Turkey, 4–8 May 2015; pp. 1625–1633.
5. Liao, Y.L.; Su, K.L. Multi-robot-based intelligent security system. *Artif. Life Robot.* **2011**, *16*, 137–141. [CrossRef]
6. Madridano, Á.; Al-Kaff, A.; Martín, D.; de la Escalera, A. Trajectory planning for multi-robot systems: Methods and applications. *Expert Syst. Appl.* **2021**, *173*, 114660. [CrossRef]
7. Wang, Q.; Li, J.; Yang, L.; Yang, Z.; Li, P.; Xia, G. Distributed Multi-Mobile Robot Path Planning and Obstacle Avoidance Based on ACO–DWA in Unknown Complex Terrain. *Electronics* **2022**, *11*, 2144. [CrossRef]
8. Sun, D.; Kleiner, A.; Nebel, B. Behavior-based multi-robot collision avoidance. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 1668–1673.
9. Rosinol, A.; Gupta, A.; Abate, M.; Shi, J.; Carlone, L. 3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans. *arXiv* **2020**, arXiv:2002.06289.
10. Dube, R.; Cramariuc, A.; Dugas, D.; Sommer, H.; Dymczyk, M.; Nieto, J.; Siegwart, R.; Cadena, C. SegMap: Segment-based mapping and localization using data-driven descriptors. *Int. J. Robot. Res.* **2020**, *39*, 339–355. [CrossRef]
11. Chen, X.; Milioto, A.; Palazzolo, E.; Giguere, P.; Behley, J.; Stachniss, C. Suma++: Efficient lidar-based semantic slam. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 4530–4537.
12. Tate, A. Generating project networks. In Proceedings of the 5th International Joint Conference on Artificial Intelligence, Cambridge, MA, USA, 22–25 August 1977; Volume 2, pp. 888–893.
13. Georgievski, I.; Aiello, M. An overview of hierarchical task network planning. *arXiv* **2014**, arXiv:1403.7426.
14. Erol, K.; Hendler, J.A.; Nau, D.S. *Semantics for Hierarchical Task-Network Planning*; Technical report; Maryland University College Park Institute for Systems Research: College Park, MD, USA, 1995.
15. Crespo, J.; Castillo, J.C.; Mozos, O.M.; Barber, R. Semantic information for robot navigation: A survey. *Appl. Sci.* **2020**, *10*, 497. [CrossRef]
16. Manzoor, S.; Rocha, Y.G.; Joo, S.H.; Bae, S.H.; Kim, E.J.; Joo, K.J.; Kuc, T.Y. Ontology-based knowledge representation in robotic systems: A survey oriented toward applications. *Appl. Sci.* **2021**, *11*, 4324. [CrossRef]
17. Joo, S.; Bae, S.; Choi, J.; Park, H.; Lee, S.; You, S.; Uhm, T.; Moon, J.; Kuc, T. A Flexible Semantic Ontological Model Framework and Its Application to Robotic Navigation in Large Dynamic Environments. *Electronics* **2022**, *11*, 2420. [CrossRef]
18. Moon, J.; Lee, B.H. PDDL planning with natural language-based scene understanding for uav-ugv cooperation. *Appl. Sci.* **2019**, *9*, 3789. [CrossRef]
19. Hwang, N.E.; Kim, H.J.; Kim, J.G. Centralized Task Allocation and Alignment Based on Constraint Table and Alignment Rules. *Appl. Sci.* **2022**, *12*, 6780. [CrossRef]
20. Thomas, A.; Mastrogiovanni, F.; Baglietto, M. Towards multi-robot task-motion planning for navigation in belief space. *arXiv* **2020**, arXiv:2010.00780.
21. Schillinger, P.; Bürger, M.; Dimarogonas, D.V. Simultaneous task allocation and planning for temporal logic goals in heterogeneous multi-robot systems. *Int. J. Robot. Res.* **2018**, *37*, 818–838. [CrossRef]
22. Buehler, J.; Pagnucco, M. A framework for task planning in heterogeneous multi robot systems based on robot capabilities. In Proceedings of the AAAI Conference on Artificial Intelligence, Quebec City, QC, Canada, 27–31 July 2014; Volume 28.
23. Galindo, C.; Fernández-Madrigal, J.A.; González, J.; Saffiotti, A. Robot task planning using semantic maps. *Robot. Auton. Syst.* **2008**, *56*, 955–966. [CrossRef]
24. Suh, I.H.; Lim, G.H.; Hwang, W.; Suh, H.; Choi, J.H.; Park, Y.T. Ontology-based multi-layered robot knowledge framework (OMRKF) for robot intelligence. In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October 2007–2 November 2007; pp. 429–436.
25. Joo, S.H.; Manzoor, S.; Rocha, Y.G.; Bae, S.H.; Lee, K.H.; Kuc, T.Y.; Kim, M. Autonomous navigation framework for intelligent robots based on a semantic environment modeling. *Appl. Sci.* **2020**, *10*, 3219. [CrossRef]
26. Hanheide, M.; Göbelbecker, M.; Horn, G.S.; Pronobis, A.; Sjöö, K.; Aydemir, A.; Jensfelt, P.; Gretton, C.; Dearden, R.; Janicek, M.; et al. Robot task planning and explanation in open and uncertain worlds. *Artif. Intell.* **2017**, *247*, 119–150. [CrossRef]

27. Fox, M.; Long, D. PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *J. Artif. Intell. Res.* **2003**, *20*, 61–124. [CrossRef]
28. Coles, A.; Coles, A.; Fox, M.; Long, D. Forward-chaining partial-order planning. In Proceedings of the International Conference on Automated Planning and Scheduling, Toronto, ON, Canada, 12–16 May 2010; Volume 20, pp. 42–49.