



Kalpa Publications in Computing

Volume 22, 2025, Pages 425–439

Proceedings of The Sixth International Conference on Civil and Building Engineering Informatics



# An Experimental Study for Multi-robot Coordination in Multi-story Building Construction Sites

Haojun Luo<sup>1</sup> and Yantao Yu<sup>1, 2\*</sup>

<sup>1</sup> Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology, Hong Kong 999077, China.

<sup>2</sup> HKUST Shenzhen-Hong Kong Collaborative Innovation Research Institute, Shenzhen 518045, China.

hluoaw@connect.ust.hk, ceyantao@ust.hk

## Abstract

Robots offer a promising solution to relieve workers from physically demanding tasks and improve safety and productivity in construction. It is critical that the robots on construction sites are coordinated effectively. However, most multi-robot coordination algorithms are designed for planar areas, neglecting the multi-story nature of building construction sites. It is still unclear how construction robots should be coordinated given the constraints of elevators while adhering to construction schedules. To fill the gap, this paper introduced the deployment of commonly used elevator algorithms and robot target allocation strategies in a multi-story construction simulation environment. Through a series of group experiments conducted in a simulated multi-story construction environment, we evaluated the performance of these algorithms and examined the characteristics of robot-elevator coordination. The results reveal that while existing algorithms with strong generalization capabilities are useful, they may be less effective in specialized scenarios like multi-story construction. This research contributes valuable insights into the future of automation in construction, paving the way for enhanced integration of robotic systems and elevator operations.

## 1 Introduction

Robots are increasingly recognized as a transformative force in the construction industry, offering solutions that alleviate physically demanding tasks while enhancing safety and productivity (Aghimien et al. 2020). Various construction robots have been developed for executing specific

---

\* The corresponding author of this paper

construction tasks, such as tiling (Le et al. 2020), plastering (Wang et al. 2024), painting (Seriani et al. 2015) and so on. The coordination of this heterogeneous robotic system is essential for enhancing efficiency and effectiveness in construction projects. However, most multi-robot coordination algorithms are designed for planar areas, neglecting the multi-story nature of building construction sites. It is still unclear how construction robots should be coordinated given the constraints of elevators while adhering to construction schedules.

There are two main characteristics of multi-story construction sites. In a multi-story construction project, there are several apartments on each floor, and each apartment has multiple processes that need to be completed, requiring careful management of numerous tasks to ensure that the entire project progresses smoothly. It is a dynamic and gradual process. Additionally, elevators play a critical role in transporting robots between floors in multi-story construction projects. Ineffective use of elevators can lead to delays in robot deployment across different construction sites, negatively impacting overall project timelines. Therefore, it is essential to develop a multi-robot coordination algorithm that considers elevator constraints and can complete all multi-story building construction tasks efficiently.

In multi-story construction sites, coordinating the movement of elevators and robots presents unique challenges that are not adequately addressed by existing algorithms designed for other scenarios, such as warehouses (Nielsen et al. 2016) or assembly lines (Kousi et al. 2019; Kousi et al. 2016). These algorithms typically do not account for the vertical movement requirements of robots, which is crucial in a multi-story environment. The complexity of determining where robots should go and how elevators should move complicates the coordination process, highlighting a significant gap in current research on multi-agent systems.

Therefore, we conducted an experimental study on multi-robot coordination in a multi-story building construction site. In our exploration of coordinating elevators and robots in multi-story construction sites, we try to deploy commonly used elevator algorithms, specifically the **LOOK algorithm** and the **nearest neighbor method**. These algorithms are selected for their strong applicability to our scenario due to their effectiveness in managing requests and optimizing movement. For the robot target allocation, we investigated two strategies: the **first-come-first-served method (FCFS)** and the **nearest-served method (NS)**. By combining these robot allocation strategies with the elevator algorithms, we form multiple experimental groups to evaluate their performance in the proposed multi-story environment. All algorithms are tested through a simulation of interior finishing work in a ten-story residential building. The experimental results provide valuable insights for future research on multi-agent coordination including elevators and robots, and the practical deployment of robots in multi-story construction scenarios.

## 2 Literature Review

The problem of finding the best elevator planning and robot target allocation in construction projects is similar to the Pickup and Delivery Problems (PDPs). In PDPs, goods or passengers must be transported from different departure locations to different destinations (Toth and Vigo 2014). In the proposed scenario, robots need to take the elevator from different departure floors to different destination floors depending on the task. According to the decision framework being considered and the availability of information, existing studies on these problems can be divided into two categories: static approaches and dynamic approaches. This section will first review existing related research, and then identify research gaps.

**Static approaches** to PDPs operate under the assumption that all relevant information, such as demand, vehicle availability, and travel times, is known in advance (Berbeglia et al. 2007). Existing research can be summarized into mathematical methods and heuristic methods.

Mathematical methods for solving static PDPs typically involve formulating the problem using optimization models that aim to minimize costs or travel times. (Berbeglia et al. 2007) presents a comprehensive classification of static PDPs and discusses various mathematical formulations, including mixed-integer linear programming models. These models are often used to derive optimal routes while considering constraints such as time windows and vehicle capacities. Once accurate mathematical models are established, solving these models can yield the optimal solution. Techniques (Cordeau 2006; Garaix et al. 2011; Qu and Bard 2015) for solving these models are developed mainly based on upon the concept of branch-and-bound (B&B). B&B is designed for general discrete and combinatorial optimization problems. However, in construction scenarios, the robot's demand for elevators depends on actual construction efficiency and environmental factors, which is non-deterministic. It is difficult to establish an accurate mathematical model to describe such a dynamic scenario.

Heuristic methods have been shown to be effective and efficient compared to mathematical methods (Ho et al. 2018). In order to find the optimal solution, B&B usually enumerates a large number of solutions. In the worst case, the computation time of the B&B program may increase exponentially. On the contrary, although the heuristic method cannot guarantee optimality, it can always find an acceptable solution in a shorter time. Common methods include tabu search (Cordeau and Laporte 2003), simulated annealing algorithm (Mauri, Antonio, and Lorena 2009), genetic algorithm (Jorgensen, Larsen, and Bergvinsdottir 2007), and hybrid heuristic algorithm (Berbeglia, Cordeau, and Laporte 2012). In the heuristic method, instead of building an accurate mathematical model, feasible solutions are constructed based on the constraints of the environment, and then the existing solutions are improved through heuristic rules to make the new solutions better and better. However, in order to calculate the fitness function for the heuristic algorithm to improve current solutions, the distribution of all demands and the benefits of completing the demands must be known in advance. In the proposed scenario, the demands for the robots to cross floors is dynamically updated over time.

**Dynamic approaches** typically provide a solution strategy rather than a deterministic static plan. The strategy uses the information revealed to specify what actions must be performed over time. Currently there are few studies on dynamic methods. A basic and commonly used strategy is to adapt an algorithm that solves a static version of the problem (Berbeglia, Cordeau, and Laporte 2010). There are two ways to implement this strategy. One is to treat all the information at the current time as a static problem to be solved whenever new information appears (Berbeglia, Cordeau, and Laporte 2010). The disadvantage of this strategy is that it takes too much time to solve a complete static problem every time. The other is to solve the static problem once at the beginning using the known information to get a feasible solution, and then use heuristic methods to update the feasible solution as time goes by and new information is obtained (Berbeglia, Cordeau, and Laporte 2010). The second method is more common, but still requires time and memory to maintain and update existing solutions. Most importantly, existing solutions focus on improving the path of the vehicle. Construction is an irreversible process, and constantly changing strategies during construction is not a good option.

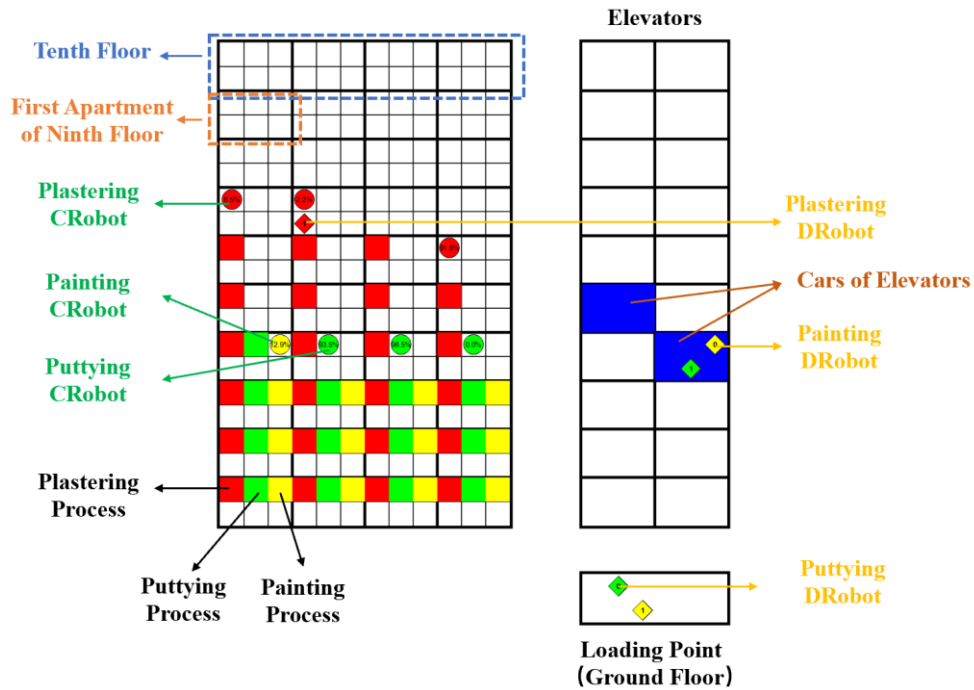
Coordinating robot teams and elevators in multi-story construction projects is challenging due to the nature of multi-story buildings. The literature review to date shows that there is a lack of an effective method to coordinate robot teams and elevators in multi-story construction sites. Therefore, we explored the combination of commonly used elevator planning algorithms and robot target allocation algorithms to see how they perform on multi-story construction sites and further explore the characteristics of multi-agent coordination on multi-story construction sites.

### 3 Method

his study focuses on the coordination of elevators and robots in multi-story construction sites, aiming to optimize their interactions to improve operational efficiency. In the proposed scenario, we deployed two types of elevator algorithms - LOOK and Nearest Neighbor. LOOK is a classic disk scanning algorithm. For the nearest neighbor method, in addition to the traditional distance-based nearest neighbor algorithm (DNN), we also improved the metric for evaluating "nearest" and proposed the improved nearest neighbor algorithm (INN). For the robot target allocation algorithm, the FCFS method and the NS method were applied. By combining these robot allocation strategies with the elevator algorithm, we formed multiple experimental groups to evaluate their performance in the proposed multi-story environment. This section is structured around the design of the simulation environment, the implementation of elevator and robot algorithms.

#### 3.1 Simulation Environment

We created a simulated multi-story construction site based on C++. The environment consists of ten floors, with two elevators and multiple robots responsible for construction and delivery. The simulation framework is designed to generate requests for elevator service and robot allocation in a construction scenario to fully evaluate the algorithm under different conditions. Details are shown in Figure 1.






**Figure 1:** The composition of the simulation environment. (CRobot means construction robot and DRobot means delivery robot.)

Here comes the workflow of the environment. We consider three construction processes, plastering, puttying and painting. They follow a flow-through construction method from bottom to top. The construction robot of the next process will enter the site after the previous process is completed.

Each process is handled by a construction robot. When the construction robot runs out of materials, it will suspend the operation and issue a material demand. After the material demand is received by the idle delivery robot, it will deliver the materials to the construction robot. After receiving the materials, the construction robot continues to work until the process is completed, and then goes to the next construction. The delivery robot should return to the loading point to replenish material after completing the delivery. It is worth noting that the construction robots may work on different floors at the same time. Material requirements may come from multiple floors. All robots rely on elevators to go to other floors. As for elevators, considering the size of construction robots in reality, each elevator is limited to carrying two robots. The real-time location of robots and elevators is known in the simulation, which can be achieved by existing technology i.e., RFID systems (Motroni, Buffi, and Nepa 2021).

In the above environment, there are three main cross-floor activities that rely on elevators. One is that the construction robot goes from the current floor to another floor to work, the second is that the delivery robot delivers materials from the loading point to the construction robot, and the last is that the delivery robot returns to the loading point after delivering materials on the current floor. The entities involved are the construction robot, the delivery robot, and the elevator. Figure 2 shows the states of these three entities. For an elevator, it can be in the state of going up, going down, and stopping. For the delivery robot, there are four places it can be located, namely the loading point, the apartment, inside the elevator, and the place waiting for the elevator. It may also be on the ways of loading point – elevator and apartment – elevator. A construction robot is similar. It could be in an elevator, in an apartment, or waiting for an elevator. Also, it could be on the way between the elevator-apartment.

|  |  |
|--|--|
| Elevators            | <ul style="list-style-type: none"> <li>• Stop</li> <li>• Going up</li> <li>• Going down</li> </ul>   |
| Delivery robot      | <ul style="list-style-type: none"> <li>• AL: at loading point</li> <li>• L2E: on the way from loading point to elevator</li> <li>• WE: waiting elevator</li> <li>• IE: in the elevator</li> <li>• E2A: on the way from elevator to apartment</li> <li>• AT: at task apartment</li> <li>• A2E: on the way from apartment to elevator</li> <li>• E2L: on the way from elevator to loading point</li> </ul> |
| Construction robot  | <ul style="list-style-type: none"> <li>• Flow construction, red→green→yellow, Bottom-up</li> <li>• WE: waiting elevator</li> <li>• IE: in the elevator</li> <li>• E2A: on the way from elevator to apartment</li> <li>• AT: at task apartment</li> <li>• A2E: on the way from apartment to elevator</li> </ul>   |

**Figure 2:** The states of Elevators, Delivery Robot, and Construction Robot

## 3.2 Algorithms

After the simulation environment is established, the algorithms are applied to it. The goal of the study is to coordinate multiple agents including robots and elevators to finish the whole construction

project. The coordination algorithm consists of two parts, one is the elevator scheduling algorithm, and the other is the robot target allocation algorithm. They will be explained in detail below.

### 3.2.1. Robot Target Allocation Algorithms

This part is to assign targets to available delivery robots. Specifically, when a delivery robot has multiple potential targets, which one should be assigned to it to determine the target floor it needs to go to. Available construction robots will cross floors from bottom to top in a flow-line construction manner according to the settings of the simulation environment, without the need to design an algorithm for allocation. Two strategies were employed:

**First-Come-First-Served:** This method allocates tasks to robots based on the order in which requests are received, ensuring a straightforward allocation process. Figure 3 shows an example of FCFS method. There is an idle robot 1 that receives a task from the 7th floor. It accepts the task and takes the elevator to the 7th floor. During this process, there is a task from the 3rd floor, but according to the FCFS principle, the robot will ignore this task and go to the 7th floor first to complete the task.

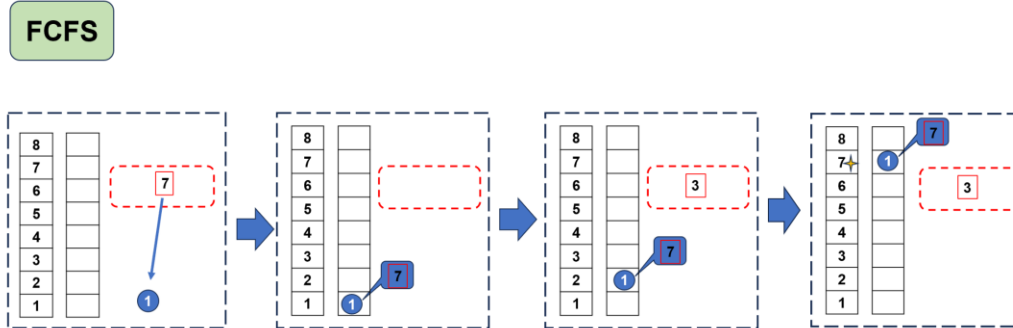


Figure 3: An example of FCFS method

**Nearest-Served Method:** In this method, delivery robots of the same type share all potential targets. The delivery robot in the elevator will put all potential targets in a target set in chronological order. Once the elevator reaches a floor in the target floor set, it will assign the delivery robot that has that floor in its target floor set to serve that floor. Then the target floor sets of all delivery robots are updated. This approach prioritizes allocating the nearest target to the delivery robot, not necessarily the earliest target. Figure 4 shows an example of NS method. The same task situation as FCFS is used here. The difference is that when a task comes from the 3rd floor, the task will be added to the robot's target floor set. The robot passes the 3rd floor first with the updated target floor set, so it will complete the task on the 3rd floor first.

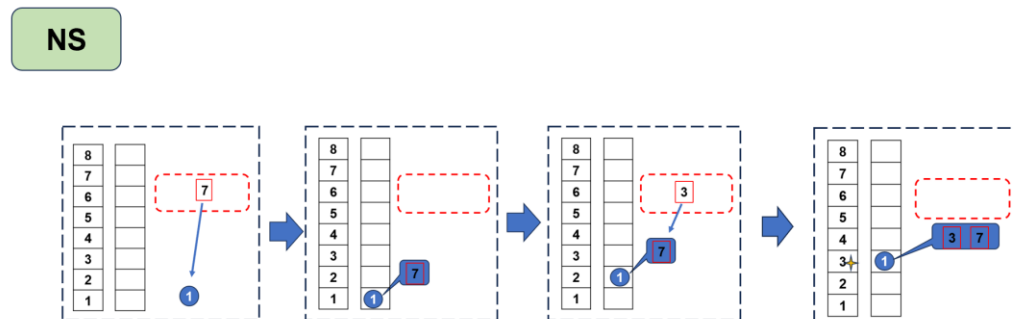


Figure 4: An example of NS method

The difference between FCFS and NS is that the former focus on solving the material needs that are called early, while the latter solves the most recent material needs.

### 3.2.2. Elevator Scheduling Algorithms

The elevator scheduling algorithm is responsible for managing elevator requests efficiently. In this study, we implemented two primary algorithms:

**LOOK Algorithm:** This classic disk scanning algorithm operates by moving the elevator in one direction, serving requests along its path before reversing direction. It effectively minimizes wait times by ensuring that all requests in one direction are addressed before changing course. Since the real-time positions of the robot and the elevator are known, two situations may occur. One is that the robot sends an elevator request when it is waiting for the elevator. In this way, the robot will get on the elevator as soon as the elevator arrives, but the robot needs to wait for the elevator to arrive. The other is that the robot sends an elevator request when it is on the way to the elevator. When the elevator arrives but the robot has not arrived yet, the elevator needs to wait for the robot to arrive and get on the elevator before leaving. Figure 5 shows the diagram of the LOOK algorithm with these two situations.

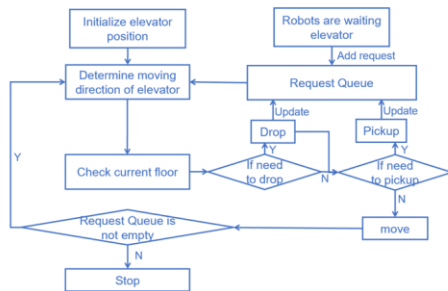


Figure 5-1. The diagram of the LOOK algorithm when the robot is waiting for the elevator

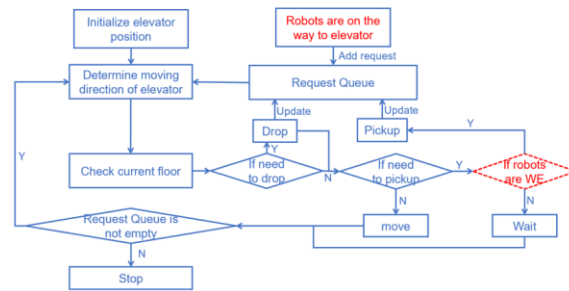


Figure 5-2. The diagram of the LOOK algorithm when the elevator is waiting for the robot

**Figure 5:** The diagram of the LOOK algorithm

**Nearest Neighbor Method:** This method focuses on fulfilling requests based on proximity. We utilized two variations:

**Distance-Based Nearest Neighbor (DNN):** This traditional approach allocates requests based on the shortest distance from the elevator's current position. Figure 6 shows the diagram of DNN.

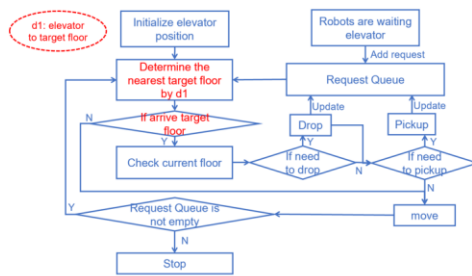


Figure 6-1. The diagram of DNN when the robot is waiting for the elevator

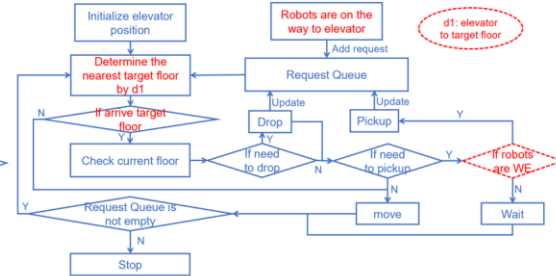


Figure 6-2. The diagram of DNN when the elevator is waiting for the robot

Figure 6.: The diagram of DNN

**Improved Nearest Neighbor (INN):** This enhanced version refines the evaluation metric for "nearest" requests by incorporating additional factors such as current wait times and the elevator's position relative to pending requests. This adjustment aims to improve responsiveness and overall service efficiency. Figure 7 shows the diagram of INN.

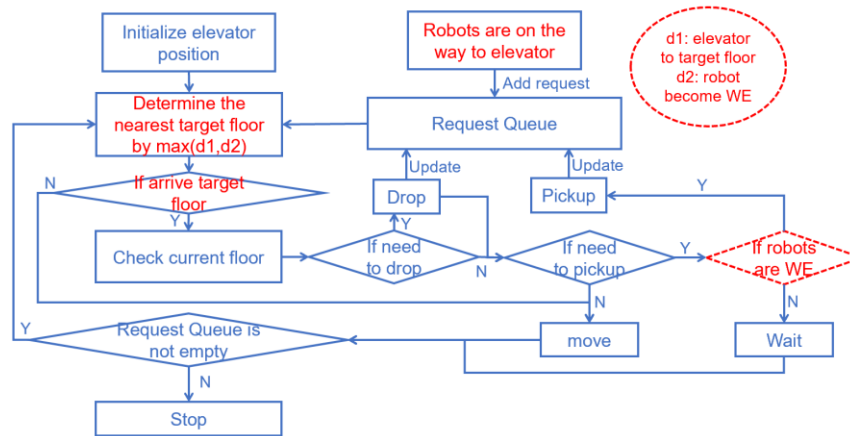


Figure 7: The diagram of INN

As can be seen from Figure 4 and Figure 5, the main difference between DNN and INN lies in the definition of "nearest". In DNN, the nearest request is selected as the target floor based on the distance (d1) between the request and the current floor of the elevator. In INN, the nearest target is evaluated based on the maximum value of the distance (d1) and the actual waiting time (d2). It is worth noting that in the case of the robot waiting for the elevator, this INN will collapse into a DNN, because the actual waiting time is equal to the distance at this time.

### 3.2.3. Multi-agent Coordination Algorithms

To evaluate the performance of the combined algorithms, multiple experimental groups were formed by pairing each robot assignment strategy with each elevator algorithm. This resulted in four distinct experimental configurations:

- LOOK Algorithm + FCFS;



- LOOK Algorithm + NS;
- Distance-based Nearest Neighbor (DNN) + FCFS;
- Distance-based Nearest Neighbor (DNN) + NS;
- Improved Nearest Neighbor (INN) + FCFS;
- Improved Nearest Neighbor (INN) + NS.

The total construction duration (TCD) is used to estimate the performance, as shown in Equation (1).

$$\pi^* = \underset{\pi}{\operatorname{argmin}}(\text{TCD}) \quad (1)$$

$\pi$  is the coordination strategy generated by the above configurations. The objective is to find the best strategy  $\pi^*$  to minimize the total construction duration.

## 4 Results

The performance of each coordination strategy was assessed using key metrics such as total construction duration. Data collected during simulation runs were analyzed to determine which combinations of algorithms yielded optimal results in coordinating elevators and robots within a multi-story environment. Taking into account the cost of the robots and the information obtained from visits to robot construction companies (Guangdong Bright Dream Robotic Co. Ltd (BDR)), we set the parameters of construction robot as shown in Table 1 and Table 2.

|   | Plastering | Puttying | Painting |
|---|------------|----------|----------|
| Work speed<br>(Percentage/unit time)              | 0.48       | 1.16     | 0.76     |
| Call material<br>frequency<br>(unit time/1 times) | 20         | 5        | 20       |

**Table 1.** The parameters of construction robot

|                                      | Plastering | Puttying | Painting |
|--------------------------------------|------------|----------|----------|
| Loading time<br>(unit time /1 times) | 3          | 3        | 3        |
| Supply time<br>(unit time/1 times)   | 5          | 5        | 5        |

**Table 2:** The parameters of delivery robot

In particular, the above settings are only for the simulation environment to make it run smoothly. These parameters are not considered as known information in all the proposed coordination algorithms.

In order to fully explore the performance of the algorithm, we set the number of robots in various ways to describe different situations, as shown in Table 3. It can be divided into three categories. The first category is groups 1-3, which is to explore the impact of increasing the number of construction robots when the number of delivery robots is very small. The second category is groups 4-6, where the number of construction robots is set inversely proportional to their construction efficiency. On this basis, we explored setting the number of delivery robots inversely proportional to the frequency of calling materials, the same number of delivery robots, and the same number of delivery robots as

construction robots. The third category is groups 7-9. Here we keep the number of construction robots and delivery robots the same, and gradually increase the number of robots.

| No. |                    | Plastering | Puttying | Painting |
|-----|--------------------|------------|----------|----------|
| 1   | Construction robot | 1          | 1        | 1        |
|     | Delivery robot     | 1          | 1        | 1        |
| 2   | Construction robot | 2          | 2        | 2        |
|     | Delivery robot     | 1          | 1        | 1        |
| 3   | Construction robot | 5          | 5        | 5        |
|     | Delivery robot     | 1          | 1        | 1        |
| 4   | Construction robot | 5          | 2        | 3        |
|     | Delivery robot     | 1          | 2        | 1        |
| 5   | Construction robot | 5          | 2        | 3        |
|     | Delivery robot     | 2          | 2        | 2        |
| 6   | Construction robot | 5          | 2        | 3        |
|     | Delivery robot     | 5          | 2        | 3        |
| 7   | Construction robot | 2          | 2        | 2        |
|     | Delivery robot     | 2          | 2        | 2        |
| 8   | Construction robot | 3          | 3        | 3        |
|     | Delivery robot     | 3          | 3        | 3        |
| 9   | Construction robot | 5          | 5        | 5        |
|     | Delivery robot     | 5          | 5        | 5        |

**Table 3.** Number settings of construction robot and delivery robot

Table 4 and Table 5 show the results of the combination of LOOK with different robot target allocation algorithms. Table 6 and Table 7 show the total construction duration by applying DNN with different robot target allocation algorithms. Table 8 and Table 9 show the total construction duration by applying INN with different robot target allocation algorithms. All the serial numbers of results are consistent with Table 3.

| No. | Total Construction Duration (unit time) |                       |
|-----|---|-----------------------|
|     | Robots wait elevators                   | Elevators wait robots |
| 1   | 23727                                   | 24214                 |
| 2   | 24330                                   | 24393                 |
| 3   | 25147                                   | 24952                 |
| 4   | 18001                                   | 17639                 |
| 5   | 14204                                   | 14212                 |
| 6   | 13728                                   | 13631                 |
| 7   | 14113                                   | 14176                 |

|   |       |       |
|---|-------|-------|
| 8 | 10939 | 10999 |
| 9 | 8084  | 7982  |

**Table 4:** Total Construction Duration by applying LOOK Algorithm + FCFS

| No. | Total Construction Duration (unit time) |                       |
|-----|---|-----------------------|
|     | Robots wait elevators                   | Elevators wait robots |
| 1   | 23535                                   | 23468                 |
| 2   | 23664                                   | 23881                 |
| 3   | 24511                                   | 24383                 |
| 4   | 16471                                   | 16165                 |
| 5   | 13326                                   | 13246                 |
| 6   | 13021                                   | 12827                 |
| 7   | 13266                                   | 13273                 |
| 8   | 9715                                    | 9954                  |
| 9   | 6732                                    | 6629                  |

**Table 5:** Total Construction Duration by applying LOOK Algorithm + NS

| No. | Total Construction Duration (unit time) |                       |
|-----|---|-----------------------|
|     | Robots wait elevators                   | Elevators wait robots |
| 1   | 24639                                   | 24420                 |
| 2   | 25298                                   | 24816                 |
| 3   | 26023                                   | 25812                 |
| 4   | 17972                                   | 17408                 |
| 5   | 14398                                   | 14041                 |
| 6   | 13643                                   | 13304                 |
| 7   | 14048                                   | 13835                 |
| 8   | 10578                                   | 10390                 |
| 9   | 7419                                    | 7240                  |

**Table 6:** Total Construction Duration by applying DNN + FCFS

| No. | Total Construction Duration (unit time) |                       |
|-----|---|-----------------------|
|     | Robots wait elevators                   | Elevators wait robots |
| 1   | 24639                                   | 24420                 |
| 2   | 25285                                   | 24975                 |
| 3   | 26400                                   | 25674                 |
| 4   | 17266                                   | 17344                 |
| 5   | 14259                                   | 13961                 |
| 6   | 13647                                   | 13252                 |
| 7   | 14024                                   | 13760                 |
| 8   | 10692                                   | 10359                 |
| 9   | 7494                                    | 7344                  |

**Table 7:** Total Construction Duration by applying DNN + NS

| No. | Total Construction Duration (unit time) |
|-----|---|
|     | Elevators wait robots                   |
| 1   | 24375                                   |
| 2   | 24784                                   |

|   |       |
|---|-------|
| 3 | 25735 |
| 4 | 17356 |
| 5 | 13986 |
| 6 | 13324 |
| 7 | 13823 |
| 8 | 10183 |
| 9 | 7205  |

**Table 8:** Total Construction Duration by applying INN + FCFS

|     | Total Construction Duration<br>(unit time) |
|-----|--|
| No. | Elevators wait robots                      |
| 1   | 24375                                      |
| 2   | 24756                                      |
| 3   | 25596                                      |
| 4   | 17222                                      |
| 5   | 13945                                      |
| 6   | 13302                                      |
| 7   | 13718                                      |
| 8   | 10070                                      |
| 9   | 7337                                       |

**Table 9:** Total Construction Duration by applying INN + NS

From the above results, we can draw the following inferences:

- The combination of LOOK algorithm + NS performed best in almost all experimental groups;
- When there is only one delivery robot for each process, increasing the number of construction robots will not improve construction efficiency but will increase the total construction duration. (Refer to the experimental results of groups 1-3);
- From the perspective of elevators waiting for robots or robots waiting for elevators, most results are that the former is better than the latter, no matter which combination it is;
- For the nearest neighbor method, INN always performs better than DNN, whether combined with FCFS or NS;

The detailed analysis will be discussed in the following section.

## 5 Discussion

As mentioned in Section 3.1, there are three main types of activities that rely on elevators to cross floors in the proposed scenario. Among them, the most frequent is the delivery robot delivering materials and returning to the loading point after delivering materials. In the simulation environment, the loading point is fixed on the ground floor. Therefore, the movement of the delivery robot is a scanning movement from the ground floor to the high floor, which explains why the LOOK algorithm performs best, because the LOOK algorithm is an efficient scanning algorithm.

When the number of delivery robots is limited, blindly increasing the number of construction robots is not a viable choice, because this will increase the proportion of construction robots moving across floors. Due to the limited capacity of elevators, more construction robots will compete with delivery robots for elevators, thereby reducing the delivery efficiency of delivery robots, prolonging

the waiting time for construction robots in need of materials, and reducing the overall construction efficiency.

The robot waiting for the elevator means that the robot will not work for a long time, and the robot's working time is directly related to the overall construction progress. The mode of waiting for the robot for the elevator can shorten the time the robot waits for the elevator, which means that the robot will be put into work faster.

In DNN, the distance between the request and the current floor of the elevator is the only factor that determines the movement of the elevator. But in reality, it is possible that when the elevator reaches the nearest requested floor, the robot is still on the way. In this case, the elevator needs to wait for the robot to arrive and pick it up before it can move. In INN, the actual time when the robot reaches the elevator and the distance from the elevator to the requested floor are taken into account, and the "closest" is evaluated more accurately in time and space. This reduces the time the elevator waits for the robot.

We also make some speculations as to why the nearest neighbor method does not perform as expected in the proposed scenario. Previous studies (Sheridan et al. 2013) have shown that the nearest neighbor is a promising method for solving the traveling salesman problem. However, in order to increase the generalization ability of the method, they relax the constraints and assumptions of the problem, such as the random generation of vehicle demand (Ghiani et al. 2003). In this study, the scenario is a specific multi-story construction site, and the construction robots follow the top-down flow construction, which greatly reduces the randomness of the demand for elevators. In this case, some special algorithms, such as the scanning algorithm LOOK, may produce better results because they match the movement mode of the majority of delivery robots. This observation suggests that while existing algorithms with strong generalization capabilities are valuable in broader contexts, they may have limited effectiveness in specialized scenarios such as robot-elevator coordination in construction. The findings indicate a significant research potential for developing multi-agent scheduling algorithms tailored specifically to unique operational environments like multi-story construction sites.

## 6 Conclusions

In conclusion, this study experimentally investigated multi-robot coordination within a multi-story construction site. We created a simulated environment consisting of ten floors, equipped with two elevators and teams of robots tasked with various cross-floor requirements. We approached the situation by implementing established elevator algorithms and robot target allocation strategies that were adapted to fit this particular context. Through group experiments, we explored the performance of these algorithms in multi-story construction settings and examined the characteristics of robot-elevator coordination. The results indicated that the combination of LOOK algorithm and NS, which operates by systematically scanning floors, is particularly well-suited for this environment. Its design aligns with the predominant motion patterns of delivery robots, facilitating more efficient elevator scheduling. By comparing our findings with conclusions from previous studies, we observed that algorithms that excel in general scenarios often perform significantly worse than those specifically tailored to the unique characteristics of multi-story construction sites. This disparity underscores the potential for further research in developing multi-agent coordination algorithms that cater to specialized operating environments. Currently, the total construction duration is the only metric considered in this study. But minimizing task completion time may increase total travel distance, leading to higher energy usage. It would provide a more comprehensive evaluation by adding additional metrics, such as resource utilization and energy consumption. Looking ahead, we plan to conduct more in-depth explorations of elevator-robot coordination in multi-story construction sites.

Our future research will focus on improving these algorithms and evaluating their applicability in more complex construction scenarios, ultimately contributing to improving operational efficiency and automation in the construction industry.

## Acknowledgements

This work was supported by Collaborative Research Fund [grant number C6044-23GF] from University Grants Committee of The Government of the Hong Kong Special Administrative Region; and 30 for 30 Scheme [grant number 3030007] funded by The Hong Kong University of Science and Technology.

## References

- Aghimien, Douglas Omoregie, Clinton Ohis Aigbavboa, Ayodeji Emmanuel Oke, and Wellington Didibhuku Thwala. (2020). 'Mapping out research focus for robotics and automation research in construction-related studies: A bibliometric approach', *Journal of Engineering, Design and Technology*, 18: 1063-79.
- Berbeglia, Gerardo, Jean-François Cordeau, Irina Gribkovskaia, and Gilbert Laporte. (2007). 'Static pickup and delivery problems: a classification scheme and survey', *Top*, 15: 1-31.
- Berbeglia, Gerardo, Jean-François Cordeau, and Gilbert Laporte. (2010). 'Dynamic pickup and delivery problems', *European journal of operational research*, 202: 8-15.
- Berbeglia, Gerardo, Jean-François Cordeau, and Gilbert Laporte. (2012). 'A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem', *INFORMS Journal on Computing*, 24: 343-55.
- Cordeau, Jean-François. (2006). 'A branch-and-cut algorithm for the dial-a-ride problem', *Operations research*, 54: 573-86.
- Cordeau, Jean-François, and Gilbert Laporte. (2003). 'A tabu search heuristic for the static multi-vehicle dial-a-ride problem', *Transportation Research Part B: Methodological*, 37: 579-94.
- Garaix, Thierry, Christian Artigues, Dominique Feillet, and Didier Josselin. (2011). 'Optimization of occupancy rate in dial-a-ride problems via linear fractional column generation', *Computers & Operations Research*, 38: 1435-42.
- Ghiani, Gianpaolo, Francesca Guerriero, Gilbert Laporte, and Roberto Musmanno. (2003). 'Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies', *European journal of operational research*, 151: 1-11.
- Ho, Sin C, Wai Yuen Szeto, Yong-Hong Kuo, Janny MY Leung, Matthew Petering, and Terence WH Tou. (2018). 'A survey of dial-a-ride problems: Literature review and recent developments', *Transportation Research Part B: Methodological*, 111: 395-421.
- Jorgensen, Rene Munk, Jesper Larsen, and Kristin Berg Bergvinsdottir. (2007). 'Solving the dial-a-ride problem using genetic algorithms', *Journal of the operational research society*, 58: 1321-31.
- Kousi, Niki, Spyridon Koukas, George Michalos, and Sotiris Makris. (2019). 'Scheduling of smart intra-factory material supply operations using mobile robots', *International Journal of Production Research*, 57: 801-14.
- Kousi, Niki, Spyridon Koukas, George Michalos, Sotiris Makris, and George Chryssoulouris. (2016). 'Service oriented architecture for dynamic scheduling of mobile robots for material supply', *Procedia CIRP*, 55: 18-22.

Le, Anh Vu, Rizuwana Parween, Phone Thiha Kyaw, Rajesh Elara Mohan, Tran Hoang Quang Minh, and Charan Satya Chandra Sairam Borusu. (2020). 'Reinforcement learning-based energy-aware area coverage for reconfigurable hRombo tiling robot', *IEEE Access*, 8: 209750-61.

Mauri, Geraldo Regis, Luiz Antonio, and Nogueira Lorena. (2009). 'Customers' satisfaction in a dial-a-ride problem', *IEEE Intelligent Transportation Systems Magazine*, 1: 6-14.

Motroni, Andrea, Alice Buffi, and Paolo Nepa. (2021). 'A survey on indoor vehicle localization through RFID technology', *IEEE Access*, 9: 17921-42.

Nielsen, Izabela, Ngoc Anh Dung Do, Peter Nielsen, and Yohanes Khosiawan. (2016). "Material Supply Scheduling for a Mobile Robot with Supply Quantity Consideration—A GA-based Approach." In, 41-52. Cham: Springer International Publishing.

Qu, Yuan, and Jonathan F Bard. (2015). 'A branch-and-price-and-cut algorithm for heterogeneous pickup and delivery problems with configurable vehicle capacity', *Transportation Science*, 49: 254-70.

Seriani, Stefano, Alessio Cortellessa, Sandro Belfio, Marco Sortino, Giovanni Totis, and Paolo Gallina. (2015). 'Automatic path-planning algorithm for realistic decorative robotic painting', *Automation in Construction*, 56: 67-75.

Sheridan, Patricia Kristine, Erich Gluck, Qi Guan, Thomas Pickles, Barış Balciog, and Beno Benhabib. (2013). 'The dynamic nearest neighbor policy for the multi-vehicle pick-up and delivery problem', *Transportation Research Part A: Policy and Practice*, 49: 178-94.

Toth, Paolo, and Daniele Vigo. 2014. *Vehicle routing: problems, methods, and applications* (2<sup>nd</sup> ed.). Society for Industrial and Applied Mathematics Philadelphia.

Wang, Yue, Liangxi Xie, Jin Chen, Mengmeng Chen, Teng Hu, Hongyu Liao, Shibin Sun, and Jian Chen. (2024). 'Mortar spraying and plastering integrated robot for wall construction', *Automation in Construction*, 165: 105533.