Accuracy and Generality of Trained Models for Lift Planning Using Deep Reinforcement Learning

- Optimization of the Crane Hook Movement Between Two Points -

A. Tarutani[a] and K. Ishida[b]

[a]Department of Architecture, Graduate School of Creative Science and Engineering, University of Waseda, Japan
[b]Department of Architecture, Assistant Professor, Doctor of Engineering, Waseda University, Tokyo, Japan
E-mail: taru4boo@fuji.waseda.jp, k_ishida@waseda.jp

Abstract –

An optimization system of a lifting plan must have generality to manage different design conditions and real-time changes at a construction site. Furthermore, it must optimize the construction planning and scheduling. In a previous study, we trained a two-point locomotion model for crane hook movement using deep reinforcement learning to generate and optimize lifting plans automatically. However, we did not test the accuracy and generality of the model. In this study, we test (1) the accuracy and (2) the generality of the trained model using a new environment. To evaluate the accuracy of the optimal solution, we examined the locus of the movement of each frame between two points. To verify the generality of the trained model, we solved an optimization problem of the crane hook movement under different conditions of the crane's learning environment using the trained model. From the results, we found that the movement path was 3.6 times the shortest path and the crane hook initially moved vertical. Furthermore, the agent solved the optimization problem of the crane hook movement when the size of the crane changed. Therefore, the corresponding range increased with increasing size of the crane. However, the agent did not solve the problem when the slewing angle in the target position was larger than the slewing angle in training. Based on these results, we believe that the limited vertical movement range and rotation range of the crane reduces the accuracy and generality of the trained model.

Keywords –

Deep Reinforcement Learning; Crane Lifting Plan; Optimization; Generality; Virtual Space; Trained Model

1 Introduction

Improving efficiency in building construction is a principal research topic in the construction field, where the shortage of skilled workers is increasing. Therefore, optimization and automation of construction planning and scheduling is important to improve construction [1]. However, optimization of construction planning and scheduling is a complicated task. Therefore, it is generally prudent to optimize each task [2].

The following two aspects are considered necessary to optimize construction planning and scheduling: (1) solving complex combinatorial optimization problems and (2) providing general versatility to manage design and real-time situation changes during construction. Considering (1), problems such as those concerning work interference, route planning, placement planning, and quantity planning can be replaced with typical problems, and optimization research has been conducted [2]. In case of (2), solving problems can be difficult or impossible when changes not existing in the optimization simulation, such as work delay and obstacle interference, occur in the real space. Furthermore, adding all necessary factors in the optimization simulation to provide generality is challenging.

Therefore, we focused on using deep reinforcement learning (RL) to ensure generality. In a previous study, we developed a crane lifting plan using deep RL. Reasons for optimizing lifting tasks are as follows:

1. lifting task is a cooperative task involving several sub-tasks, and
2. lifting task is a complex optimization problem because it includes factors such as the lifting route, layout, model, quantity, and building order.

In Section 2, we review some related work on lifting planning and deep RL.

In Section 3, we first explain the trained model for two-point movement developed in the previous study, and then we present the methods to test the accuracy and generality of the trained model of the previous study.

In Section 4, we discuss the results of the verifications, and in Section 5, we propose a method for creating the environment.

2 Related Work

2.1 Previous Research on the Construction Planning Optimization Problem

In general, construction planning and scheduling can be considered as an optimization problem. Therefore, research is conducted from optimization perspectives in work interference, construction equipment arrangements, and route planning. The construction planning optimization problem considers factors, such as safety, environment, cost, time, and work cooperation.

Wo et al. indicated that a lifting plan requires a planning scheme that considers the numbers, layouts, models, and operating times of cranes. They established a mathematical model for the spatio-temporal planning of tower cranes that reduces the total cost compared to the initial solution, indicating that it provided the optimal solution for all construction projects [2].

2.2 Previous Research on Deep RL

We focused on RL to ensure the generality of the optimization and/or automation system. RL is a machine learning method; it is different from supervised learning because it acts by itself and collects environmental information regardless of the existence of accurate or ground truth data. Therefore, it responds to unknown events for which the correct answer is unknown.

RL is mainly used for autonomous control of machines and problems with no accumulated data. When creating an autonomous control system for a robot, it is difficult to establish rules for sensor systems, control values, etc. Therefore, the robot learns the surrounding situation through RL and controls itself [3]. In addition, RL is used to generate data when there is no accumulated training data, such as building vibration control, to derive optimal vibration control values [4].

There are several phenomena where the correct answers are unknown because data are not accumulated. Construction planning and scheduling are among them.

Moreover, we can perform complex information processing by combining deep learning and RL (called deep RL). This combination aids in generating an optimal solution. The solution is used to generate long-term strategies, such as artificial intelligence for gaming.

This study combines the aforementioned features to develop a crane lifting plan.

The crane lifting plan optimization problem (based on the features of deep RL) is classified as follows.

1. A path creation function for lifting a target object by controlling movements such as slewing, derricking, and lifting and lowering.
2. A function to provide a strategy for deciding the order of construction that is the most efficient.

By developing a model with these functions, we aim to automatically develop a lifting plan for an unknown condition in a simulation. We partially performed this task in a previous study and developed a trained model. This outline is presented in Section 3.

In addition, research is being conducted to investigate methods to generalize trained models developed using RL. According to Miyashita et al., a trained model for car collision prevention using deep RL prevented collisions with cars not learned during training [4].

When training a lifting task, a construction site with several factors, is difficult to reproduce. However, if the inference model can be provided with a general versatility as described earlier, the need to describe each element in the field can be minimized. In this study, the versatility is verified using a trained model [5] for moving a crane hook between two points, as developed in the previous study. Furthermore, we propose a method for creating a learning environment for general purposes based on the inference result.

3 Method

3.1 Research Aim

In this study, we perform a simulation to investigate the following.

- Accuracy of the trained model.
- Generality of the trained model.
- Creation of a learning environment to improve the accuracy and versatility of the trained model at the learning stage.

3.2 Outline of the Research Method

In this section, we present the development environment for deep RL and the structure of the study using deep RL.

We use Unity ML-Agents [6] for deep RL. Unity ML-Agents is a framework for building "environment" for RL on Unity and for "training" and "inference" agents. The proximal policy optimization algorithm [7] is used for the RL algorithm.

As shown in Figure 1, deep RL is categorized into

two processes: learning and inference. In the learning stage, an agent learns an action to maximize the cumulative reward in some environment (learning environment). A model (a formula/method of calculation) developed by this process is called a trained model. In inference, a trained model is applied to an unknown data to provide an answer. At the inference stage, we create an unknown environment for 4D construction simulation and scheduling with several changes.

In Section 3.3, we first explain the trained model developed in the previous study. In Section 3.4, we describe the methods to test the accuracy and generality of the trained model developed in the previous study.

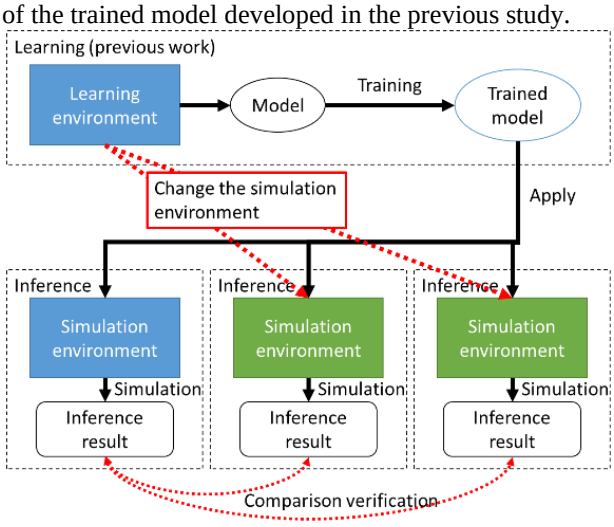


Figure 1. Verification of inference results with changed simulation environment

3.3 Training Environment

3.3.1 Simplification of Learning Content

The optimization problem of the crane lifting plan using deep RL based on the features of deep RL is categorized into the following problems.

- Optimizing hook route: lifting a target object by controlling movements such as slewing, derricking, and lifting and lowering.
- Optimizing assembly order: providing efficiency order to assemble building components.

By developing a model with these functions, we aim to automatically create a lifting plan for an unknown site in a simulation. To learn these functions, we divide the route creation function into moving between two points and collision prevention, as shown in Figure 2. This approach simplifies the environmental information and aims to converge learning.

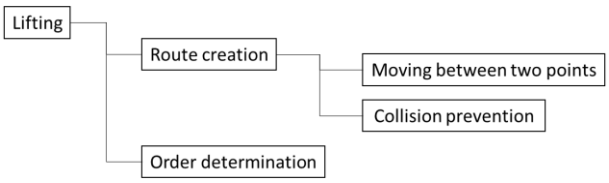


Figure 2. Simplification of learning content

3.3.2 Environment

This section presents an overview of RL, the environment, and inference results under the same conditions as during training.

RL is a machine learning algorithm that learns from the interaction between agents and the environment. Agents are learners and decision-makers. The environment is a non-agent factor on which the agent operates. As shown in Figure 3, the agent refers to the crane; the environment refers to the target, floor, coordinate space, reward, status, and other information. The agent (crane) operates on the environment and receives information such as coordinates, vector, and speed from the environment to determine new actions.

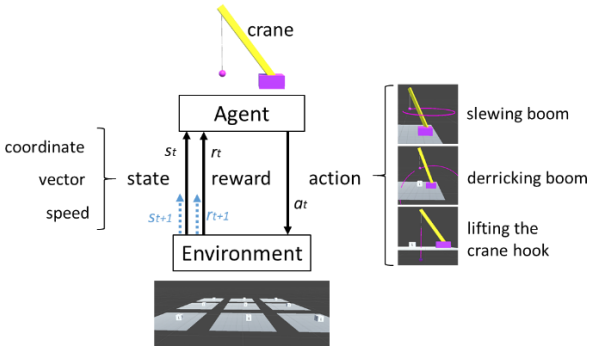


Figure 3. Interaction between agent and environment in reinforcement learning

Figure 4 (a) shows the start and end points when the crane moves between two points. Figure 4 (b) depicts the installation range of the target (box). The center coordinate of this box is the end point. During training, a target (end point) randomly appears in this range (on the horizontal plane). Figure 5 presents the inference result from the trained model for movement between the two points. The crane instantly moves between the two points, and when the end point is reached, the end-point position is initialized.

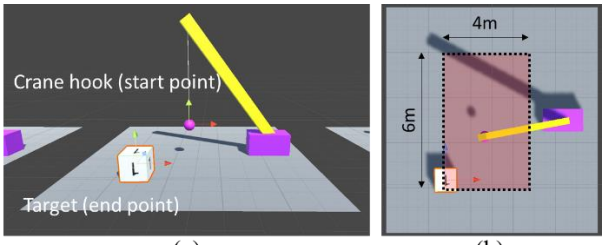


Figure 4. Start and end points (a); target (end point) installation range (b)

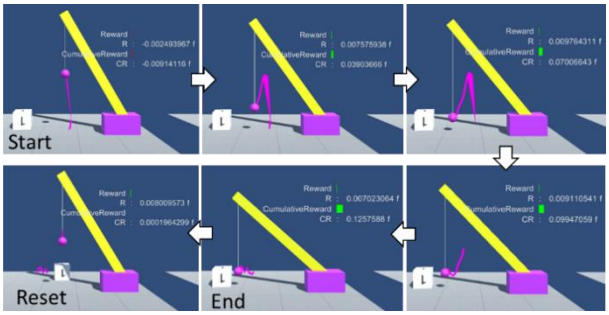


Figure 5. Trained model for moving between two points

3.3.3 Agent Action and Crane Models

In this section, we explain the operations of agents (crane) on environments in deep RL.

As shown in Figure 6, we developed a crane with three operations: slewing the boom, derricking the boom, and lifting and lowering the crane hook. The agent learns these control values by training, without limiting the rotation angle of the slewing and derricking. When lifting the crane hook, the vertical direction is restricted such that it does not exceed the boom tip. By simplifying the model in the initial stages of training, unnecessary information is deleted from the search, and the search converges easily, as shown in Figure 7.

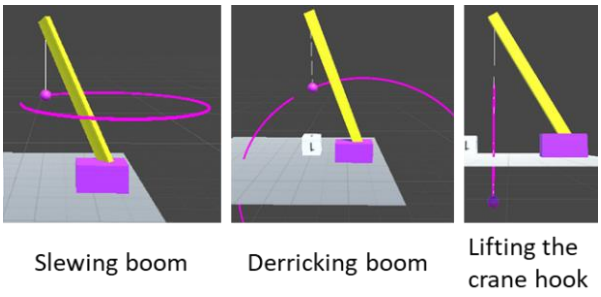


Figure 6. Movement of the crane

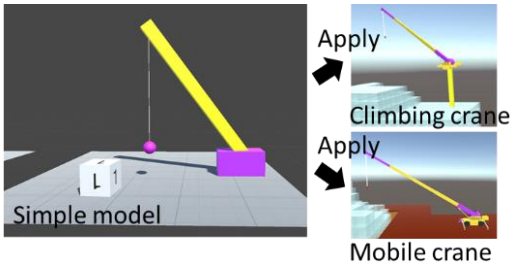


Figure 7. Simplified crane model

3.3.4 Training Method

Figure 8 shows the situation during training. The model training time was reduced by 87% by parallelizing and training nine models simultaneously. The number of iterations was 500,000.

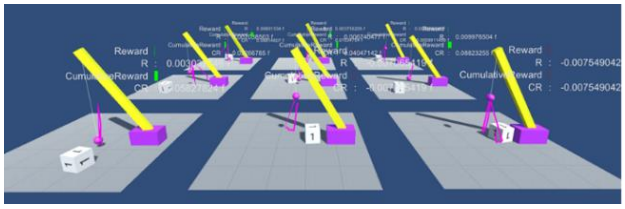


Figure 8. Parallel training of nine models

3.3.5 State and Reward

In this section, we detail a remarkable method to develop a route for a crane hook to move between two points. In deep RL, the environment rewards agents based on their behavior; agents learn behaviors that maximize their cumulative rewards. We created an environment to reward the agents as follows.

- Reward when the crane hook reaches the end point.
- Reward the movement of the crane hook towards the end point.

For a route search between two points in the horizontal direction, we can create a movement route between two points by only giving a reward when the crane hook reaches the end point. However, the optimization of the crane hook route is a path search in a three-dimensional space; hence, we created an environment to reward the operation of the crane hook approaching the end point. We used the inner product to reward the action of the crane hook approaching the end point. We multiply the inner product of the two vectors by the reward and assign it as the reward. As shown in Figure 9, the two vectors are crane hook to target direction vector and crane hook speed vector. If the angle between these two vectors is small (i.e., the inner product is close to 1), the reward is close to 1×reward.

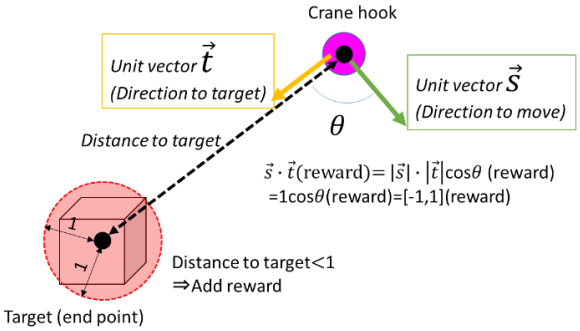


Figure 9. Simplification of rewards

3.4 Method of Verification

3.4.1 Verification of the Optimal Solution

This section presents a method investigate the accuracy of the optimal solution of the trained model.

We verify the accuracy of the optimization of the movement between two points by comparing "the length of the trajectory of the crane hook" and "the length of the shortest path between the two points."

To obtain the length of the crane hook trajectory and the shortest path between the two points, we record the coordinates of the crane hook and the start point for each frame, as listed in Table 1, and track the length and the shortest distance between the two points.

Furthermore, to visualize the crane hook's trajectory, we draw the trajectory of the crane hook, as shown in Figure 10.

Table 1. Recorded coordinates for each frame

start			end			now			last			len
0.00	-6.17	0.00	-1.50	0.50	-2.70	-1.90E-7	-6.54	7.19E-9	0.00	-6.17	0.00	0.00
0.00	-6.17	0.00	-1.50	0.50	-2.70	6.85E-7	-7.66	1.55E-8	-1.90E-7	-6.54	7.19E-9	0.36
0.00	-6.17	0.00	-1.50	0.50	-2.70	1.59E-6	-9.26	1.35E-8	6.85E-7	-7.66	1.55E-8	1.49
0.00	-6.17	0.00	-1.50	0.50	-2.70	1.29E-6	-10.9	1.06E-8	1.59E-6	-9.26	1.35E-8	3.08
0.00	-6.17	0.00	-1.50	0.50	-2.70	2.35E-6	-12.2	-2.91E-8	1.29E-6	-10.9	1.06E-8	4.70
0.00	-6.17	0.00	-1.50	0.50	-2.70	3.22E-6	-13.7	-1.12E-7	2.35E-6	-12.2	-2.91E-8	6.02
0.00	-6.17	0.00	-1.50	0.50	-2.70	3.40E-6	-14.8	-2.01E-7	3.22E-6	-13.7	-1.12E-7	7.52
0.00	-6.17	0.00	-1.50	0.50	-2.70	4.40E-6	-16.2	-2.22E-7	3.40E-6	-14.8	-2.01E-7	8.64
0.00	-6.17	0.00	-1.50	0.50	-2.70	5.35E-6	-17.2	-2.57E-7	4.40E-6	-16.2	-2.22E-7	10.07
0.00	-6.17	0.00	-1.50	0.50	-2.70	4.74E-6	-18.4	-2.62E-7	5.35E-6	-17.2	-2.57E-7	10.98

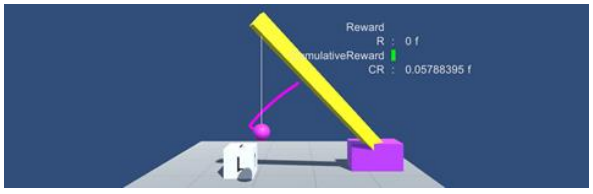


Figure 10. Visualization of the crane trajectory

3.4.2 Verification of the Generality of the Trained Model

This section presents methods to investigate the generality when altering the environment in which the trained model is adapted.

As discussed in Section 2.2, the models trained by deep RL can perform in situations different from the learning environment. However, its adaptability is unknown. Therefore, we make the following changes to the simulation environment during inference.

- Placement: Change the positional relationship between the crane and the end point. We set the end points inside and outside the "installation range of the end point during training." As shown in Figure 11, we examine the inference results while installing the end points in sequence on the horizontal plane. We do not change the position of the crane and the position of the start point.
- Type: We use a default crane for training and a large crane with a different boom length and vertical height (Figure 12).

From the inference results, we examine the generality of the inference model and propose a method to create a learning environment to improve generality.

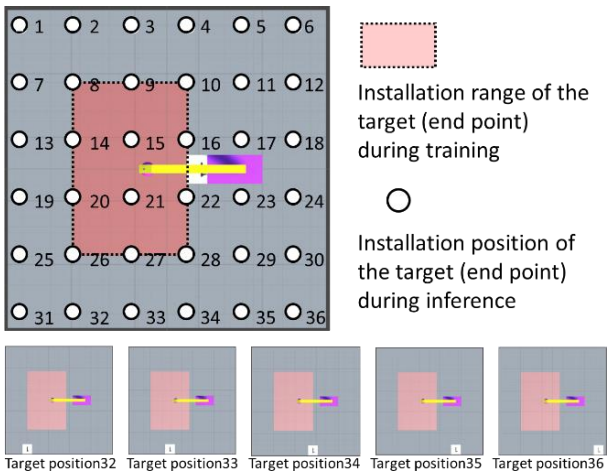


Figure 11. Change in the installation position of the end point

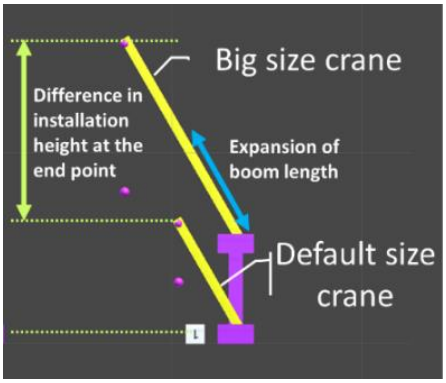


Figure 12. Change in crane type

4 Results and Discussion

4.1 Verification of the Accuracy of the Trained Model

In this section, we present the results of the accuracy verification of movement between two points using deep RL and propose a method for creating an environment to improve the accuracy of this movement.

Based on the coordinates of the start and end points, trajectory length, and the visualization of the crane hook trajectory, the following were observed.

- The movement path was 3.6 times the shortest path
- The trajectory length was almost constant, regardless of the change in the distance between the two points.
- Vertical movement was rapid, whereas horizontal movement was slow.

Figure 13 shows the transition of the locus length for each frame, i.e., the locus of movement between the two points for six iterations, with different lengths between the two points. In addition, as shown in Figures 13 and 14, the length of the trajectory is constant regardless of the distance between the two points. Figure 14 compares the shortest distance between the two points and the actual trajectory length. The length of the trajectory is approximately 3.6 times longer.

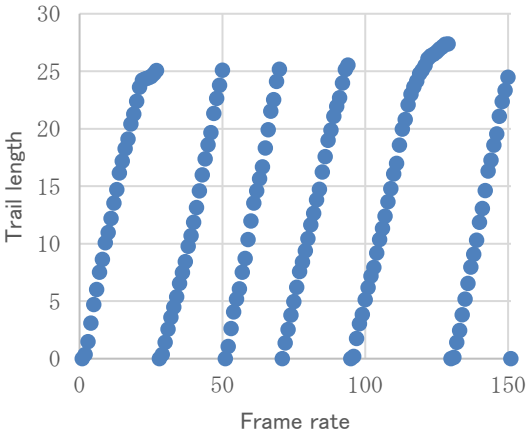


Figure 13. Change in the crane hook trail length for each frame

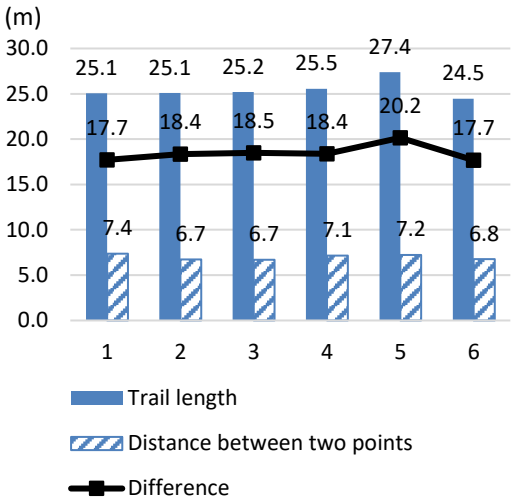


Figure 14. Comparison of shortest path length and crane hook movement path length

As shown in Figure 15, the crane hooks initially moves in the vertical direction and then in the x and z horizontal directions. This occurs because although the position of the end point is rearranged in the horizontal direction during training, it does not move in the vertical direction. Therefore, it is essential to lower the crane hook, and a reward can be obtained based on the movement toward the target direction. Therefore, when the end point is set on the plane, as shown in Figure 16 (a), the agent learns to move the crane hook and then move in the horizontal direction.

To prevent the agent from moving excessively by prioritizing the movement of the lowering of the crane hook, we propose to create an environment, as shown in Figure 16 (b). We randomly set the installation position of the end point in the vertical direction such that the crane hook lifts in the vertical direction above the start point.

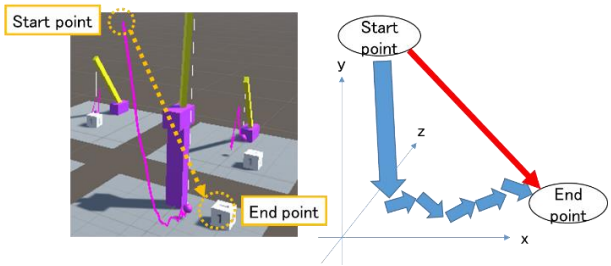
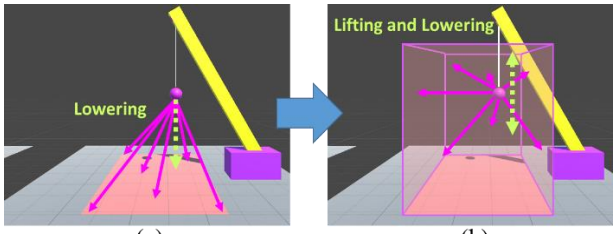


Figure 15. Trail of movement between the two points of the crane hook



(a) (b)

Figure 16. Environment that causes wasted movement of lowering the crane hook (a); environment for preventing wasted movement (b)

4.2 Verification of the Generality of the Trained Model

In this section, we describe the inference results that were examined based on the method discussed in Section 3.4.2 and propose a method to create an environment that improves the generality from inference results.

For the simulation environment, in inference, we set the end points inside and outside the "installation range of the end point during training" and let the agent infer. Consequently, as shown in Figure 17, the inference result is classified into three cases: reaching the end point, reaching the end point and performing unnecessary movements, and not reaching the end point. Figures 18 (a) and 19 (a) show the inference results corresponding to the location of the end point. Figures 18 (b) and 19 (b) depict the list of images of each inference result with the end-point position changed. Figure 18 shows the inference results using the default size crane. The inference result of installing the end point on the front of the crane includes the result that the boom is not sufficiently long. Figure 19 illustrates the inference result from the movement between two points using the larger crane with different boom lengths and crane heights.

These results show that changing the crane size does not affect the agent performance. When the size of the crane is changed, the positional relationship with the end point changes vertically. However, we confirmed that the agent can perform the movement between two points even if the positional relationship between the crane and the end point is changed. In addition, the range of movement between two points is expanded according to the size of the crane.

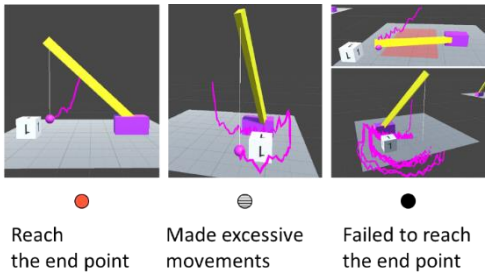
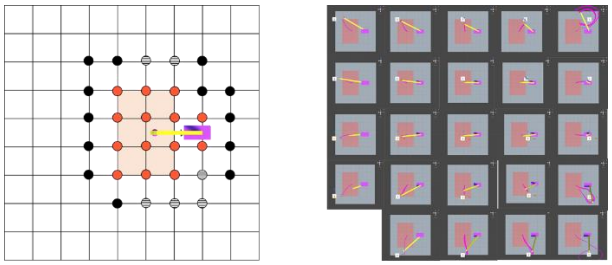
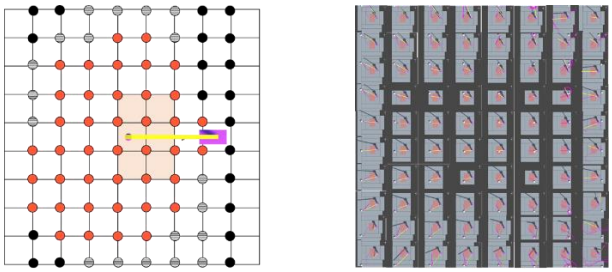


Figure 17. Three types of inference results



(a) (b)

Figure 18. Result of reaching the end point by the default size crane (a); simulation of movement between two points with the end point placed at each point (b)



(a) (b)

Figure 19. Result of reaching the end point by the big size crane (a); simulation of movement between two points with the end point placed at each point (b)

Based on Figures 18 (a) and 19 (a), we hypothesized that the range of end-point positions where the agent can perform a movement between two points is proportional to the size of the crane. Therefore, we expanded the mapping of the default size crane (Figure 18 (a)) and verified the similarity of the range of movement between two points by superimposing it on the mapping of the big crane (Figure 19 (a)), as shown in Figure 20 (a). Consequently, the range of movement between the two points was the same.

Furthermore, as shown in Figure 20 (b), the agent did not perform the movement between two points when the slewing angle of the crane increased. The slewing

angle remains within the range to reach the installation position of the end point during learning. This is probably because the slewing angle of the crane is narrow with respect to the installation range at the end point during training. Therefore, to obtain a trained model that moves in a wide range in the horizontal direction, we propose setting the end-point position such that the slewing angle becomes large, as shown in Figure 21. Additionally, when training with a small crane in a small range during training, the process can be adapted to a large crane by inference.

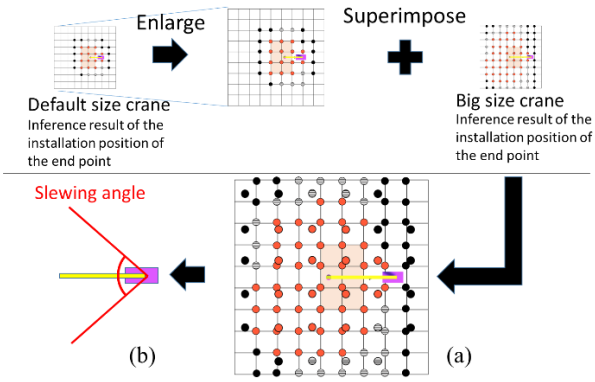


Figure 20. Similarity of the two ranges (a); limit of slewing angle for trained model (b)

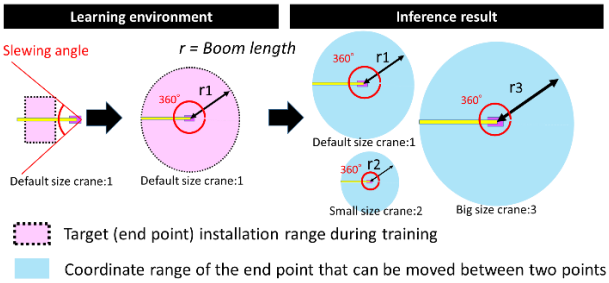


Figure 21. Environment that improves model slewing

5 Conclusion

In this study, we evaluated the accuracy of the trained model and the generality of an environment using a trained model that learned the route creation for a crane hook in a lifting task using deep RL.

We observed the following.

- The trained model's movement between two points did not traverse the shortest path. Although, it was not the optimum solution, it was inferred that the accuracy of movement between two points can be improved by creating an environment in which the crane hook is vertically lifted and lowered.
- Changing the crane size does not affect the model performance.
- The agent did not perform the movement between two points when the turning angle of the crane increased. To obtain a trained model that traverses a wide range in the horizontal direction, it is necessary to position the end points for large slewing angles.

Therefore, we conclude that the method for improving the accuracy and generality of the model for moving the crane hook between two points involves the creation of an environment that moves the crane hook vertically and horizontally (Figure 22).

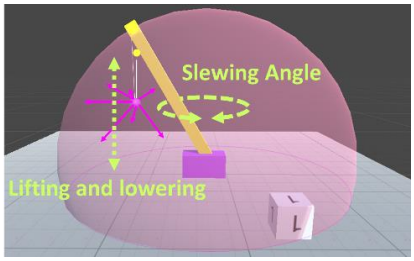


Figure 22. Environment that improves model accuracy and generality

References

[1] Zhiqian Z. and Wei P., Lift planning and optimization in construction: A thirty-year review, In *Automation in Construction*, Volume 118, 2020.10

[2] Keyi W., Borja G S. and Feilian Z., Spatio-temporal planning for tower cranes in construction projects with simulated annealing, In *Automation in Construction*, Volume 111, 2020.3

[3] Megumi M., Shirou M. Yasuhiro F., Mitsuru K., Tobias P., Eiichi M., Ryosuke O. and Daisuke O. Toward onboard control system for mobile robots via deep RL. In *32nd Conference on Neural Information Processing Systems (NIPS 2018)*, 1–12, 2018.

[4] Mikio S., Masateru H., Daisuke C and Keisuke W. Development of Active Response Control System Using AI (Part1. Outline of System). In *Summaries of technical papers of annual meeting (Tohoku Chapter, Architectural Institute of Japan)*,397-298, 2018.9

[5] Aoi T. and Kosei I., Crane Lifting Simulation Using Reinforcement Learning. In *Proceeding of the architectural research meetings II (Kanto Chapter, Architectural Institute of Japan)*, 455-458,

2020.3
[6] Arthur J., Vincent P.B., Ervin T., Andrew C., Jonathan H., Chris E., Chris G., Yuan G., Hunter H., Marwan M. and Danny L. Unity: A General Platform for Intelligent Agents. On-line: https://arxiv.org/pdf/1809.02627.pdf, Accessed: 15/06/2020.
[7] John S., Filip W., Prafulla D., Alec R. and Oleg K., Proximal Policy Optimization Algorithms. On-line: https://arxiv.org/pdf/1707.06347.pdf, Accessed: 15/06/2020.