Making Uncoordinated Autonomous Machines Cooperate: A Game Theory Approach

Kohta Sakurai and Masahide Horita

School of Engineering, University of Tokyo, Japan
sakurai-kohta655@g.ecc.u-tokyo.ac.jp

Abstract –

The construction industry in Japan is facing an increasing shortage of labor that has resulted viable demands for automation and robotics. This research study applies an approach based on game theory to model cooperation in an uncoordinated autonomous environment. A set of construction machines have been designed to complete a joint task that needs cooperation without a priori knowledge of how willing or likely other agents are to cooperate. Our simulation results show that the crisped strategy model best-promoted cooperation and successfully executed the model construction task by using the robot operating system 2.

Keywords –

Dynamic collaboration; game theory; deep reinforcement learning; multiagent determination model

1 Introduction

Automatic construction is being promoted strongly in the construction field in Japan. This field is suffering from the aging population problem that mainly comprises the construction industry and the shortage of skilled labor who can operate construction machines. To deal with these problems, efforts are currently expended to take advantage of the rapidly improving digital technology to automate constructions in ways in which operations can be conducted remotely and autonomously [1].

In the construction field, it is crucial for machines to cooperate with each other. This is because many types of machines are involved and they must achieve one common goal by associating various operations and tasks. For example, a machine should conduct its task in a rush if it is the bottleneck in the entire construction, or it should yield its resources and its flow line to others if it has some freedom to do so. To achieve the autonomous construction goal, it is important to adopt task-management approaches and cooperation mechanisms in the construction field.

Therefore, this study aims to design a dynamic collaboration mechanism that enables construction machines to engage in their assigned tasks and cooperate with each other by applying the mathematical cooperation game theory models to the construction field. This study is based the concept that cooperative construction can be realized through individual efforts by deriving the optimal decision that considers interactions among multiple agents (i.e., without the coordination through central control) and established models that enable machines to make individual judgments. We focused on borrowing tasks conducted by more than two machines as an example of construction tasks and solves the borrowing flow-line designing problem in an actual working environment.

2 Related and Previous Research

2.1 Mathematical Approach of Cooperation Using a Game Theory Approach

Rational choice theory is a field in mathematics that is based on model cooperation. Based on this theory, the components of society are modeled as the agents or mathematical expressions of the environment, and the interaction of each component is described using a specific equation. For example, Bonabeau [2] describes complex social systems to enable the expression of emergent phenomena based on the interactions of all individual agents with others. Dynamic collaboration is considered an emergent phenomenon that appears as the result of the interactions of agents involved in task execution (e.g., making a concession effort not to prevent the flow line of other machines reduces the completion time of the overall task); accordingly, machine cooperations can be reproduced by describing interactions in agent-based models and with the use of mathematics.

Gambetta [3] studied the relationship between cooperation and risk of choice. In addition, they defined the trust of agents in other components from a prediction viewpoint; they suggested that the observation of other components would transit in accordance with their subjective intention, as a means of expression in the

subjective provability, and discussed the relationship between the freedom of choice and subjective trust probability. Additionally, Vives and Feldman [4] analyzed prosocial behavior, i.e., cooperation, from the viewpoint of absolute risk and ambiguity, and observed the relationship between one's patience to ambiguity and the emergence of prosocial behavior. The term ambiguity used in the study by Vives and Feldman [4] can be reworded to the term "subjective volatility to future events." Therefore, it is mentioned [4] that the degree of preference toward the volatility to future events may have a strong affection on the emergence of prosocial behavior.

This study considers the emergence of cooperation in the context of optimization in agent-based simulation. Generally, in agent-based simulation that uses game theory approach, a certain pay-off game and the space of actions for it in the environment are proposed (such as those proposed by Chen et al. [5]) and the relationship between cooperation and action determination way are discussed. However, this study focuses on task optimization in the dynamic programing designed by Bellman equation and evaluates the degree of system optimization. In other words, this study contributes to the total system optimization by enabling the agents to cooperate in their action space by game theory approach without coordinating with each other. This approach helps the agents to determine their actions more flexibly from the environment with cooperation in the optimization context, and this enables the efficient globally-optimized autonomous construction.

2.2 Deep Reinforcement Learning

One of the major deep reinforcement learning approaches is achieved by the Deep Q-Network (DQN) [6]. This approach adopts an optimum solution that assimilates the method of deep machine learning into value-based reinforcement learning.

In the value-based reinforcement learning method represented by DQN, Q-value, the totality of the present value of future rewards, and the assembly of the action strategy of the agent itself $\{a_s\}$ (or the probability of action choice of the agent itself, $\pi(a,s)$) are being explored. This method enables the exploration of maximized rewards from a prospective viewpoint (by solving this equation, system optimization will be completed).

At some time point t_0, the totality of the net present value of rewards can be written as (1) using the action a and the state s:

$$E(s_{t_0}|a) = r(s_{t_0},a) + \gamma\left[\max_{\{a_{t_i}\}_{i=1}^{\infty}} \sum_{i=1}^{\infty} \gamma^{i-1} r(s_{t_i},a_{t_i})\right] \quad (1)$$

where $r(s,a)$ is a rewarding function that decides the amount of reward from states s and the agent's actions a, and γ is the time-discount rate (hereinafter, the subscripts of time are abbreviated.) Furthermore, $\{s_t\}$, the assembly of transition s is dependent on action a. The indentation of t on the action a and the state s indicates that the value is for time t.

The agent makes its choice that maximizes its total net present value of rewards; in other words, the action that maximizes equation (1) at each time t. From this perspective, the total net value of rewards in this circumstance is called the value (or value function) of the state s_t, $Q(s_t)$, at time t (however, $Q(s_t)$ can be determined only by the state s_t and not from $\{a_s\}$ or $\pi(a,s)$). This is because the agent can select its choice only by observing the state at each time point.

Thus, the value function $Q(s_t)$ can be expressed by equation (2),

$$Q(s_t) = \max_a \left\{ r(s_t,a) + \gamma\left[\max_{\{a_t\}_{t=1}^{\infty}} \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t)\right]\right\} \quad (2)$$

Focusing on the recursive part on the right side, equation (2) can be rewritten as follows:

$$Q(s_t) = \max_a \{ r(s_t,a) + \gamma Q(s_{t+1}) \} \quad (3)$$

Equation (3) is called the Bellman equation and is extensively used to solve the optimization problem in dynamic programming, such as reinforcement learning. The estimation of $Q(s_t)$ in this equation is equivalent to the explosion of optimized action a_t at an optional time t (this is because the optimized action a_t can be expressed according to equation (4):

$$a_t = argmax(r(s_t,a) + \gamma Q(s_{t+1})) \quad (4)$$

2.3 Deterministic Policy Gradient Algorithms (DDPG)

DDPG is a deep-learning algorithm that was presented by Silver et al. [7]. This algorithm consists of two models, namely, the actor-critic. Because the application of the concept of dividing estimation into these two models was very convenient for estimating the value function in multiagent tasks, this study established a multiagent, deep-learning model which was based on the DDPG network.

DDPG is the model used for estimating the value function $Q(s_t)$ in equation (3), as well as the DQN model. However, this model can be divided into two networks. The first one is the actor network which determines the action of the agent depending on the state s_t, and the second one is the critic network which estimates the structure of the task. The combination of these two networks enables the estimation of $Q(s_t)$ (the value where the action and the state are given). Figure 1 illustrates the concept of DDPG.

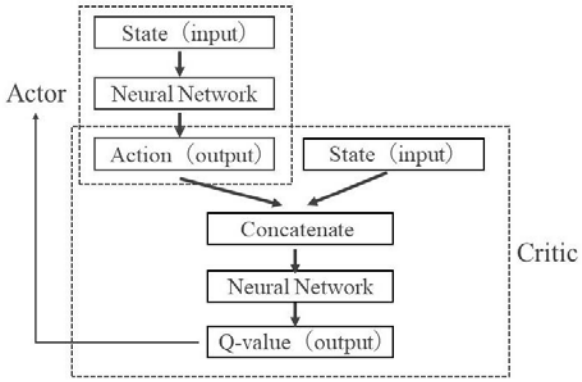


Figure 1. Actor–Critic Networks in Deterministic Policy Gradient Algorithms (DDPG)

When the function value is estimated in the DDPG model, the assembly of the action in response to the state $\{a(s_t)\}$ and the Q-value function $Q(s_t)$ are respectively replaced by the action policy [depending on the state $\mu(s_t)$] and the estimated conditional total net present value of rewards $E(s, \mu(s))$. This enables the division of the Q-value function into the actor and critic networks. The same as the DDPG model, this study adopts the actor and critic network system which estimates the agent's action policy (actor) and the relationship between the agent's action and rewards (critic) in the effort to estimate of the Q-value function.

2.4 Robot Operating System 2 (ROS2)

ROS2 is an integrative software platform that includes the tools and libraries that are necessary to develop robots. This study established the task execution environment of the real borrowing tasks of machines that was conducted by multiple agents who were already trained in the simulation by using the commands and instructions in ROS2.

3 Models and Simulation

3.1 Derivation of Multiagent Q-value Function

First, the Bellman equation, which was presented in Subsection 2-1, is extended to the model which can be applied to the multiagent system.

In a multiagent system, the competitiveness and altruism of tasks should be considered. This is because it is assumed that one's choice will have some effect (reduce or increase total net present rewards) on other agents in the case in which multiple agents engage in their tasks collaboratively at the same time. For example, if two agents engage in a major task together, they can complete it within a short time. Alternatively, if one's flow line disturbs other machines, this reduces the entire total net present value of rewards the agent can receive). Specifically, the reward r that the agent can get in the state s is determined by both the action of the agent itself a_s and the action of another agent a_p if the number of engaging agents is equal to two (the same is true for the multi-agent systems that consist of more than three agents.) Hereafter, the action of the agent conducting the task itself is represented as a_s, and the action of another agent, who can be recognized as a partner from the perspective of the agent conducting the task itself, is represented as a_p. This difference is very important to understand which action can be controlled or not by the focused agent.

Considering the following points, the total net present value of the agent itself depending on the state s_t, the assemble of the action of the agent itself a_s, and that of a_p (equation (1) in the single-agent system case) is represented as follows (5),

$$E\left(s_t \middle| a_s = a_s, a_p = a_p\right) = r\left(s_t, a_s, a_p\right) + \gamma \int_{a_{p,t} \in \{a_{p,t}\}} da_{p,t} w(a_{p,t}) \left[\max_{\{a_{s,t}\}_{t=1}^{\infty}} \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_{s,t}, a_{p,t}) \right] \quad (5)$$

Regarding the rewarding function, the reward that the agent can receive at time t depends on s_t, a_s, and a_p; therefore, the argument of the reward function is extended to these three elements.

However, attention should be paid so that the argument of the left side is only s_t because all $\{a_{s,t}\}$ and $\{a_{p,t}\}$ values that emerge after the time point t are determined uniquely at time point t, including the probabilistic representation. This determination is because the partner agent is assumed that makes repeatedly the best judgment for itself (or the one which seemed to be the best at some specific time point); this helps the pruning of the transition tree. In addition, as the agent repeatedly makes the best choice for itself, this enables the pruning of the transition tree. Therefore, the total net present reward value can be determined by s_t at any time point; this is the departure from the estimated multiagent Q-value function in this study.

Meanwhile, the first term on the right side of equation (5) $r\left(s_t, a_s, a_p\right)$ is a function that depends only on s_t because a_s and a_p are constant in this situation as the left side is set to the conditional total net present value where the actions a_s and a_p are selected at time point t. However, regarding the second term $\max_{\{a_{s,t}\}_{t=1}^{\infty}} \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_{s,t}, a_{p,t})$ is the function that depends on state s and the assembly of its partner's choices $\{a_{p,t}\}$ because $a_{p,t}$ remains in the equation (the

argument $a_{s,t}$ is deleted by the maximization function). Thus, the arguments are not unified if the right and left sides are integrated directly into the equation.

Consequently, the integral of a function in a definite interval should be calculated to unify the arguments by multiplying it with the voluntary weighting function $w(a_{p,t})$ (when the maximization function is considered as the integration of a function multiplied by the delta function $\delta(x)$, this can be regarded as the same variation as the single-agent Q-value derivation). In this situation, the range of integration is the entire range of $\{a_{p,t}\}$.

The weighting function is an appropriate function based on the partner's action a_p. This weighting function corresponds to the idea of "belief" in game theory. Agents have some rational beliefs about the actions of others that exist outside their own optimization space (this means that they perform their own optimization by multiplying some of their appropriate weights by elements), and they determine their own choices. From this perspective, the belief is usually represented by the expression of the estimated choice probability of the partner's action $p(a_p)$, so the weighting function $w(a_{p,t})$ in equation (5) can be replaced into $w\left(p(a_p)\right)$. In this case, $w\left(p(a_p)\right)$ is determined voluntary such that $\int_{a_p\in\{a_p\}} da_{p,t} w(p(a_p)) = 1$ is required following the definition of probability.

Finally, like the situation of the single-agent version, the Q-value function in the multiagent system is represented by equation (6) by replacing the conditional total net present value of rewards with the Q-value function.

$$Q(s_t) = \int da_p \max_{a_s} w(p(a_p))\{r(s, a_s, a_p) + \gamma Q(s_{t+1})\} \quad (6)$$

The next key point is to identify a way to determine the weights $w\left(p(a_p)\right)$, i.e., the way to determine one's belief according to assumptions for optimized actions of others.

One way that seems to be rational from the perspective of game theory is to determine the weights simply based on the probability density function $w\left(p(a_p)\right) = p(a_p)$. In this case, the right side of equation (6) is regarded as the expected value of the total net present value at time point t depending on the partner's choice a_p. This study refers to the determination weighting approach as the *expected action choice pattern* (this determination way is the expression that allows the volatility to future events caused by other choices).

Conversely, if the action choice probability of other agents' $p(a_p)$ is fixed at some time point t, it can be rationally assumed that the partner agent must just select the unique choice that maximizes the choice probability (and the Q-value for the partner) because a) the partner is exploring the best actions for itself during the training and b) the partner can select just one action in the transition from s_t to s_{t+1}. (The probabilistic expression can represent one's choice; however, its action is not probabilistic but definitive). Thus, by using the delta function,

$$w\left(p(a_p)\right) = \delta\left(a_p - argmax\left(p(a_p)\right)\right) \quad (7)$$

where,

$$\int_{x\in\{a_p\}} dx \cdot \delta\left(x - argmax\left(p(a_p)\right)\right) f(x) = f\left(argmax\left(p(a_p)\right)\right) \quad (8)$$

This study refers to this determination weighting approach as the *maximum action choice pattern*. Herein, two ways of choice determination were set, namely, the expectative and the maximum action choice patterns; in other words, the determination ways of agents' beliefs. Additionally, this study discusses the relationship between the determination belief and the behavior of task execution represented by the swarms which were generated by the models.

3.2 Multiagent Actor–Critic Estimation Network (MA-DDPG Model)

As was presented by Silver et al. [7], the Q-value function was first replaced by the expression that depended on the self-action determination policy $\mu_s(s_t)$ and the partner's action determination policy $\mu_p(s_t)$, as follows (10):

$$(s_t) \cong Q(s_t, \mu_s(s_t), \mu_p(s_t)) \quad (9)$$

According to equation (7),

$$\mu_s(s_t) = a_s|_{s=s_t} = argmax\left(Q(s_t)\right) \quad (10)$$
$$\mu_p(s_t) = f(a_p)|_{s=s_t}$$

As mentioned before, this is because the self-action (which depends on s_t) can be determined definitively as it maximizes the Q-value of the agent itself. However, the partner's action will be estimated by determining rationally the weighting parameters in the model in some way (this paper used the estimations performed according to the observations of the others during training based on past experiences and periodically substituted the outcomes to the model). In the situation presented in this study, agents could only predict some types of laws for the action choice depended on s_t based

on deep-learning networks.

The multiagent actor–critic model that enabled the dynamic collaborated task execution in this study is shown in Figure 2. In the function estimation used in this study, the weighting function $w(p(a_p))$ was set out of the actor's network (partner-action policy estimation network) because setting the output of this network as the estimated value of the partner's action choice probability was more convenient for making the training labels based on past experiences, and the weighting function was multiplied during the transition from the actor's network to the critic network.

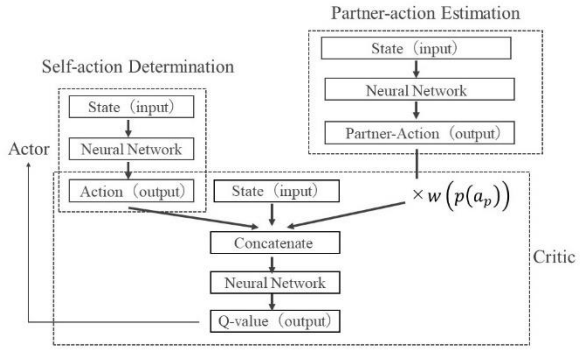


Figure 2. Multiagent Actor–Critic Estimation Network

The appropriate training of these networks enables the estimation of the Q-values of given tasks and the entire output of the model. The training targets of the three networks (self-action determination network, partner-action estimation network, and critic network) were respectively the action which maximized the Q-value at the time point t, the states-actions corresponding vectors which were stocked in the replay buffer (this enabled the estimation of the action choice probability, that is, the action policy), and the Q-value which was calculated based on the target networks. The adopted loss function of the critic model was the mean squared error (MSE) and the previous research findings, and the adopted loss functions of two actor networks were categorical cross-entropy functions because the actions were discretized in the conducted simulations.

Adopting the model shown in Section 3.2, the multiagent actor–critic estimation networks enabled the construction task execution based on dynamic collaborated cooperation between the agents.

3.3 Cooperation Fee

Changing the determination mechanism and the cooperation fee are ways to encourage the agents to cooperate.

The cooperation fee is a way of distributing its rewards to other agents as compensation for the commitment of others. For example, the study by Miyakzaki [7] focused on the application of reward allocation methods to other agents as in indirect compensation for the commitment of others. This study also focused on the way of indirect rewards as a way of encouraging the agents to cooperate, and considered the effects associated with the use of the cooperation fee by distributing a certain proportion of rewards to the other agents at the time one agent received rewards (in cases in which an agent received a chunk of the soil or discharged it, or in the case of the borrowing task on which this study was focused on). The cooperation fee was considered as the distribution of the rewards in the task, and is defined based on equation (11),

$$r_{cooperation\ fee} = r_{direct} \times \delta \tag{11}$$

where γ_{direct} represents the rewards that will be received directly by the task, and δ is a dummy binary variable which takes the value of one in the case in which the cooperation fee becomes available and the value of zero in the case the cooperation fee is not available. This study considered the effect of changing the availability of cooperation fee on the emergence of cooperative behaviors (the result is described in Section 4.1.2).

3.4 Construction Task Simulation

This study set a borrowing task of machine dump cars as an example of simulated and simplified construction cooperation tasks. A dump car received a clump of beads from an excavator and the dump carried it to a particular filling area in the borrowing task. This effort was completed by two dump machines, and the clump of beads was alternatively substituted with clumps of sand. The task was finished when a specific quantity of beads was carried. The training task (for machine learning) was pursued and completed based on computer simulations, and the actual execution environment was designed by using ROS2 commands. The overhead view of the site is shown in Figure 3. Solid lines in Figure 3 represent flow lines through which the machines can pass.

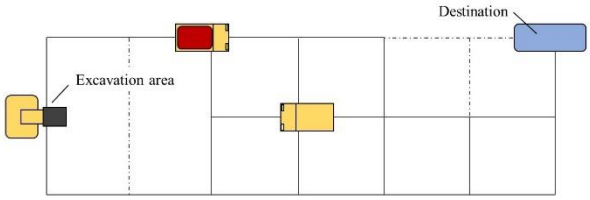


Figure 3. Overhead view of task execution area

3.5 Simulation and Learning Flow

Figure 4 depicts the learning flow diagram of agents.

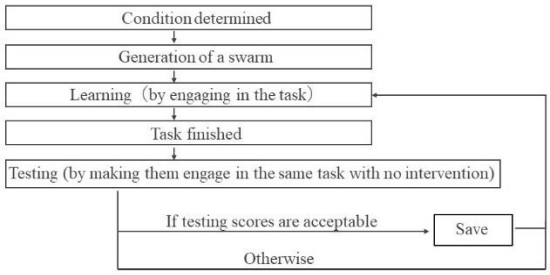


Figure 4. Flow Diagram of Training Agents and Simulations

Based on this flow diagram, the swarms of trained agents were generated, and their behaviors were observed at each point. The scores of the swarms that could finish the test task in 200 time steps were saved and the transition of their scores was observed. The score of the task execution was defined according to equation (12) by considering that the earlier agents could finish the task and that they had high scores; negative scores were not recorded.

$$\begin{cases} Learning\ score = 10000 - TL \\ Test\ score = 200 - TS \end{cases} \quad (12)$$

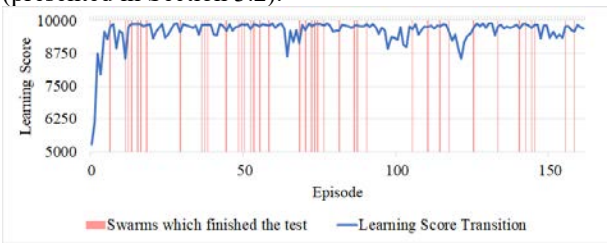
where TL is the time of the learning task execution and TS is the testing task execution.

4 Results and Discussion

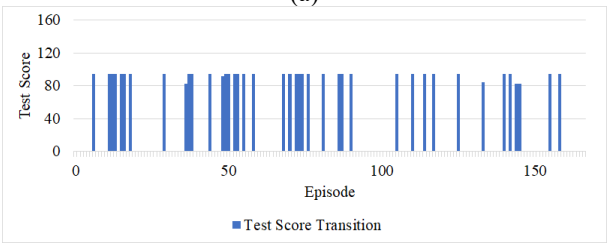
4.1 Simulation Results

4.1.1 Choice theory differences

Figure 5 describes the learning and testing results of the performances which were observed by the swarms generated by the *expectative action choice* model (presented in Section 3.2).



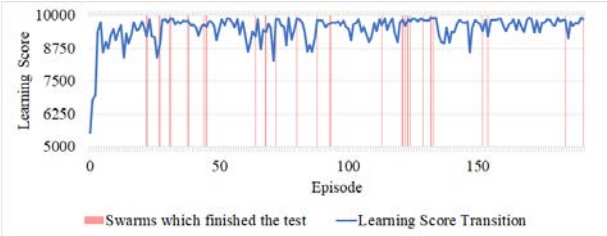
(a)



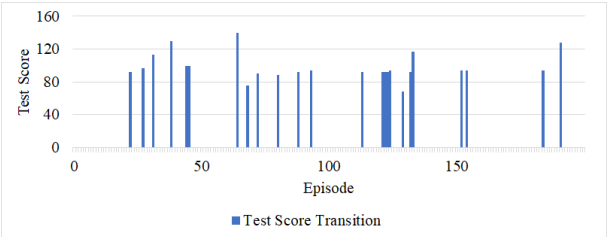
(b)

Figure 5. a) Learning Score Transition and b) Test Score Transition in the Expectative Action Choice Model

Figure 6 describes the learning and test results of the performances which were observed by the swarms generated by the *maximum action choice* model which was presented in Section 3.2 (the cooperation fees of models used in Figures 5 and 6 were both 60% of their maximum values to eliminate the effects of cooperation fee changing; $\delta = 0.6$ was thus used in equation (11)).



(a)



(b)

Figure 6. a) Learning Score Transition and b) Test Score Transition in the Maximum Action Choice Model

Table 1. Comparison of the simulation results of the expectative action choice and maximum models

Model	Expectative model	Maximum model
Number of episodes	161	191
Number of swarms which finished the test	41 (25.5% of all swarms)	24 (12.6% of all swarms)
Number of swarms with high score (>100)	0 (0.0% of tested swarms)	5 (20.8% of tested swarms)

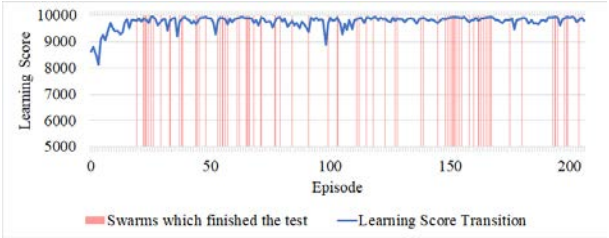
By comparing Figures 5 and 6, the behavioral differences of agents that were attributed to the weighting function $w\left(p(a_p)\right)$ differences were considered. First, regarding the score that was observed in learning, the fluctuation of the scores was less in the expectative action choice model (Figure 5a) than that in the maximum

action choice model (Figure 6a); thus, the learning scores were more stable in the executive action choice model. These differences were considered to be attributed to the facts that a) the executive action choice was more redundant to the exploration randomness compared with the maximum action choice and b) the probability that the swarms could execute the borrowing task without being affected by the external elements was higher.

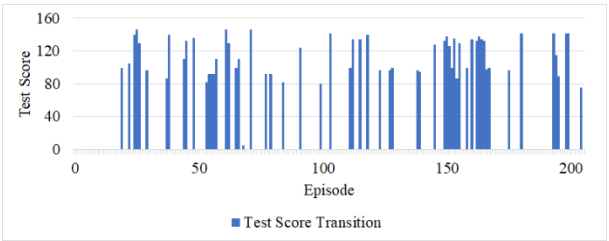
Conversely, a comparison of Figures 5b and 6b shows the probability that the generated swarms had higher test scores in the case of the maximum action choice model than those in the case of the expectative action choice model. The borrowing task could be finished in a shorter period if two agents were cooperative (if they could concede their flow lines to each other) because they could execute the task in parallel without disturbing each other. Therefore, it was considered that in the case of the maximum action choice model, the probability of generating more cooperative swarms was higher compared with that associated with the expectative action choice model. This was because the weighting function of the maximum action choice model enabled the agents to expect the choices of others to make the high-risk and high-return choices. Additionally, this point was considered to contribute to the generation of agents that could finish the task within a shorter period.

4.1.2 Cooperation Fee Differences

The effect of the cooperation fee which was discussed in Section 3.3 was also analyzed. Figure 7 shows the agents' outcomes in the case in which the cooperation fee was not available ($\delta = 0$ in equation (11)). The corresponding agents' outcomes in the case in which the cooperation fee was available ($\delta = 1$) is shown in Figure 8.

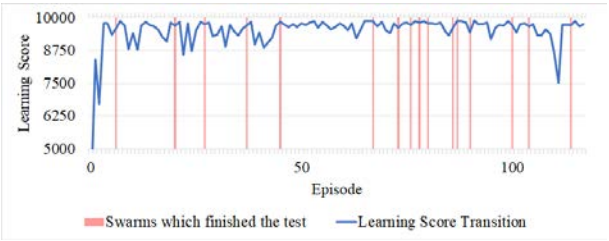


(a)

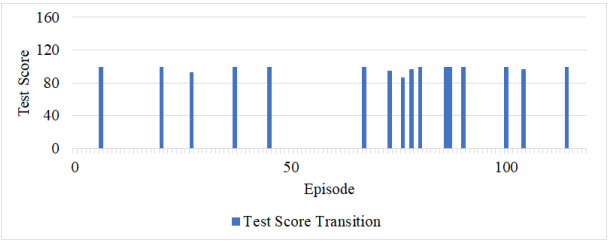


(b)

Figure 7. a) Learning Score Transition and b) Test Score Transition in the Case Wherein the Cooperation Fee was Unavailable ($\delta = 0$)



(a)



(b)

Figure 8. a) Transition of Learning Score Transition and b) Test Score Transition in the Case Wherein the Cooperation Fee was Available ($\delta = 1$)

Table 2. Comparison of the simulation results of the expectative action choice and maximum models

Cooperation fee availability	Not available ($\delta = 0$)	Available ($\delta = 1$)
Number of episodes	206	117
Number of swarms which finished the test	64 (31.1% of all swarms)	16 (13.7% of all swarms)
Number of high-score swarms (>100)	34 (53.1% of tested swarms)	0 (0.0% of tested swarms)

In the case in which the cooperation fee was available, the task was not executed cooperatively compared with the case in which the cooperative fee was unavailable. This seemed to be because the cooperation fee may affect positively the degree of sabotage. When the cooperation fee was available, the agents could obtain the rewards without engaging in the task; accordingly, this may lead to the sabotage of the agents. Conversely, in the task design of this study, the situation in which all the agents engaged enthusiastically in their tasks made the total borrowing task time shorter. Therefore, the dedication to its task seemed to assign a cooperative effect to the task.

Based on the experiments in Subsections 4.1.1 and 4.1.2, the maximum action choice model and the unavailability of the cooperation fee were the primary contributory factors in the achievement of increased machine cooperation in borrowing tasks.

4.2 Experiments in an Actual Environment

Using the conclusion that was reached in Section 4.1 and simulation results, a swarm was generated based on the learning flow diagram and it was adequately trained. The borrowing outcomes using radio-controlled machinery in an actual environment are shown in Figure 9.

Figure 9. Task Execution in an Actual Environment

5 Conclusion

This study structured a determination algorithm that enabled the autonomous operation of machines in the construction field, focused on a method of reinforcement learning that encouraged agents to obtain an autonomous determination model by repeating training to achieve dynamic collaboration in the construction field, and used game theory to represent cooperation mathematically. In the borrowing task set as a virtual step of the construction task in this study, agents could determine cooperative choices following the model of the maximum action choice in the case in which the cooperation fee was unavailable. We verified that agents could complete set tasks within a shorter time by structuring their determination models cooperatively.

In this study, we developed a globally optimizing method of realizing an optimized multiagent system that enabled agents to determine each action efficiently in an environment that changes depending on the actions of others without coordination and performed a virtual step of an autonomous construction task. The findings of this study indicate that an agent can be encouraged to acquire optimized decisions by applying game theory to a deep-learning model in an event that the interaction of multiple agents is considered. This approach will enable cooperative agents to easily achieve a single construction project.

However, regarding the volatility of a future event, the uncertain transition of an environment caused by the uncertainty of the observation can also be considered in addition to the uncertainty caused by the actions of others. Such uncertainty can be addressed by creating an elaborate building information modeling (BIM) model and setting appropriate weightings to the uncertainty of the environment. In the latter method, the Bellman equation, which describes a game with complete information, must be extended to an equation that can represent a game with incomplete and uncertain information. Further, improvement is required, as the game theory can contribute to achieving a sufficiently autonomous construction.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number JP22H01561 and Japan Science and Technology Agency (JST) - Moonshot Research and Development Program, Grant Number: JPMJMS2032.

References

[1] Ministry of Land, Infrastructure, Transport and Tourism, Japan. Automated and autonomous technology for construction equipment installation. Online: https://www.mlit.go.jp/sogoseisaku/constplan/sosei_constplan_tk_000049.html, Accessed: 01/12/2022.

[2] Bonabeau, E. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(Suppl 3):7280–7287, 2002.

[3] Gambetta, D. Can we trust. *Trust: Making and breaking cooperative relations*, 13:213–237, 2000.

[4] Vives, M. L., & Feldman Hall, O. Tolerance to ambiguous uncertainty predicts prosocial behavior. *Nature Communications*, 9(1):1–9, 2018.

[5] Chen, Y. S., Yang, H. X., Guo, W. Z., & Liu, G. G.. Promotion of cooperation based on swarm intelligence in spatial public goods games. *Applied mathematics and computation*, 320:614–620, 2018.

[6] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602,* 2013.

[7] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, pages 387–395, Beijing, China, 2014.

[8] Miyakzaki, K., Arai, S., & Kobayashi, S. A theory of profit sharing in multi-agent reinforcement learning. *Jinkouchinou (in Japanese)*, 14(6):1156–1164, 1999.