(IJNCAA)

ISSN 2220-9085(ONLINE)
ISSN 2412-3587(PRINT)

INTERNATIONAL JOURNAL OF NEW COMPUTER ARCHITECTURES AND THEIR APPLICATIONS

Volume 10, Issue 3
2020

SDIWC
www.sdiwc.net

International Journal of
NEW COMPUTER ARCHITECTURES AND THEIR APPLICATIONS

Overview

The SDIWC International Journal of New Computer Architectures and Their Applications (IJNCAA) is a refereed online journal designed to address the following topics: new computer architectures, digital resources, and mobile devices, including cell phones. In our opinion, cell phones in their current state are really computers, and the gap between these devices and the capabilities of the computers will soon disappear. Original unpublished manuscripts are solicited in the areas such as computer architectures, parallel and distributed systems, microprocessors and microsystems, storage management, communications management, reliability, and VLSI.

One of the most important aims of this journal is to increase the usage and impact of knowledge as well as increasing the visibility and ease of use of scientific materials, IJNCAA does NOT CHARGE authors for any publication fee for online publishing of their materials in the journal and does NOT CHARGE readers or their institutions for accessing the published materials.

Publisher

The Society of Digital Information and Wireless Communications
20/F, Tower 5, China Hong Kong City, 33 Canton Road, Tsim Sha Tsui, Kowloon, Hong Kong

Further Information

Website: http://sdiwc.net/ijncaa, Email: ijncaa@sdiwc.net,
Tel.: (202)-657-4603 - Inside USA; 001(202)-657-4603 - Outside USA.

The issue date is Sept. 2020.

ISSN 2220-9085 (Online)
ISSN 2412-3587 (Print)

IJNCAA

Volume 10, Issue No. 3

2020

CONTENTS

ORIGINAL ARTICLES

International Journal of
NEW COMPUTER ARCHITECTURES AND THEIR APPLICATIONS

The *International Journal of New Computer Architectures and Their Applications* aims to provide a forum for scientists, engineers, and practitioners to present their latest research results, ideas, developments and applications in the field of computer architectures, information technology, and mobile technologies. The IJNCAA is published four times a year and accepts three types of papers as follows:

1. **Research papers**: that are presenting and discussing the latest, and the most profound research results in the scope of IJNCAA. Papers should describe new contributions in the scope of IJNCAA and support claims of novelty with citations to the relevant literature.
2. **Technical papers**: that are establishing meaningful forum between practitioners and researchers with useful solutions in various fields of digital security and forensics. It includes all kinds of practical applications, which covers principles, projects, missions, techniques, tools, methods, processes etc.
3. **Review papers**: that are critically analyzing past and current research trends in the field.

Manuscripts submitted to IJNCAA **should not be previously published or be under review** by any other publication. Plagiarism is a serious academic offense and will not be tolerated in any sort! Any case of plagiarism would lead to life-time abundance of all authors for publishing in any of our journals or conferences.

Original unpublished manuscripts are solicited in the following areas including but not limited to:

- Computer Architectures
- Parallel and Distributed Systems
- Storage Management
- Microprocessors and Microsystems
- Communications Management
- Reliability
- VLSI

International Journal of New Computer Architectures and their Applications (IJNCAA) 10(3): 32-38
The Society of Digital Information and Wireless Communications, 2020 ISSN 2220-9085 (Online); ISSN 2412-3587 (Print)

An Experimental Study for Tracking Ability of Deep Q-Network

Masashi SUGIMOTO[1*] Ryunosuke UCHIDA[1]
Kentarou KURASHIGE[2] Shinji TSUZUKI[1]
[1]Communication Systems Engineering
Department of Electrical and Electronic Engineering,
Graduate School of Science and Engineering, Ehime University
3 Bunkyo, Matsuyama C., Ehime Pref. 790-8577 Japan
E-mail: sugimoto.masashi.du@ehime-u.ac.jp
(∗ : *Corresponding author*)
[2]Division of Information and Electronic Engineering,
Department of Sciences and Informatics,
Muroran Institute of Technology
27-1 Mizumoto, Muroran C., Hokkaido 050-8585 Japan

ABSTRACT

Reinforcement Learning (RL) had been attracting attention for a long time that because it can be easily applied to real robots. On the other hand, in Q-Learning, since the Q-table is updated, a large amount of Q-tables are required to express continuous"states," such as smooth movements of the robot arm. There was a disadvantage that calculation could not be performed real-time. Deep Q-Network (DQN), on the other hand, uses convolutional neural network to estimate the Q-value itself, so that it can obtain an approximate function of the Q-value. From this characteristics of calculation, this method has been attracting attention, in recent. On the other hand, it seems to the following of multitasking and moving goal point that Q-Learning was not good at has been inherited by DQN. In this paper, to confirm the weak points of DQN by changing the exploration ratio as known as epsilon dynamically, has been tried.

KEYWORDS

Reinforcement Learning, Deep Q-Network, Exploration Ratio, Object Tracking Ability, Maze Problem.

1 INTRODUCTION

Over the years, many studies have been conducted with the objective of facilitating the working of robots in dynamic environments [1, 2, 3]. Various robots have been developed to assist humans in workspaces, such as a house or factory [4]. In general, robots are required to work effectively and safely in a dynamic environment to achieve their tasks. In addition, the robots should recognize state as similar as Human. However, it is not easy to make a robot behave like a human in dynamic environments [5, 6]. When they are working in a certain environment, humans select an appropriate course of action through subconsciously predicting all the changes in the environment and their next state. For achievement these problems, in recent years, various machine learning methods have been suggested. In reinforcement learning, it attracts attention as the technique that often use in the actual robot [7, 8, 9, 10]. Reinforcement Learning (RL) had been attracting attention for a long time that because it can be easily applied to real robots. On the other hand, in Q-Learning, it has some problems; since the Q-table is updated, a large amount of Q-tables are required to express continuous"states," such as smooth movements of the robot arm. From the table amount, there was a disadvantage that calculation could not be performed real-time. Another one of the problems, a robot does not cope with changing purpose in RL. RL has been demanded to achieve various purposes, because what request to robot is diversifying and to achieve various purposes in robot have been wanting, as mentioned above.

Deep Q-Network (DQN), on the other hand, uses Convolutional Neural Network (CNN) to estimate the Q-value itself, so that it can obtain an approximate function of the Q-value[11]. From this characteristics of calculation, this method has been attracting attention, in recent. On the other hand, it seems to the following of multitasking and moving goals that Q-Learning was not good at has been inherited by DQN. In this paper, to confirm the weak points of DQN by changing the exploration ratio as known as ϵ dynamically, has been tried.

Rest of this paper is organized as follows: In section 2, we explain the how to obtaining or deciding the optimal action for agent in situation of utilization DQN. In parallel, we provide details about the proposed method. In Section 3, we explain about the setting for the experiment. Finally, in Section 4, we present the conclusions of this study.

2 THE FLOW OF LEARNING OF DQN

Deep Q-Network is based on Q-Learning of an ordinal Reinforcement Learning, where problems are typically stated as Markov Decision Processes (MDP). The MDP consists of a pair: state s_t and action a_t. Transitions between states are performed with transition probability p, reward r and a discount rate γ. Probability transition p shows the number of transitions and rewards occurrence from one state to the other, where the sequential state and reward depend only on the state s_t and action a_t taken at the previous time step $(t-1)$. Reinforcement Learning defines environment for the agent to perform certain actions that according to policy, to maximize the reward. The basis of optimal behavior of the agent is defined by Bellman equation, that is a widely used method for solving practical optimization problems.

Reinforcement Learning can be sufficiently applicable to the environment in case of the all achievable states can be manged and stored in Random Access Memory (RAM) of a computer. However, the environment where the number of states overwhelms the capacity of computational environments ordinary RL approach is not very applicable. Furthermore, in real environment, the agent has to face with continuous states and continuous variables and continuous action problems[9]. It will be needed to consider that the complexity of environment has to operate in the standard well defined RL, that Q-space will be built based on states and actions. Network architecture using CNN, choice of network hyper parameters and learning is performed during training phase. In DQN, it allows the agent to explore unknown environment and acquire knowledge which over time makes them possible for imitating human behavior.

The main concept of DQN was depicted on below fig. 1, where Q-network proceeds as a as nonlinear approximation which maps both state into an action value.

During the training process, the agent, interacts with the environment and receives data, which is used during the learning the Q-network. The agent explores the environment to build a complete picture of transitions and action outcomes. At the beginning the agent decides about the actions randomly which over time becomes insufficient. While exploring the environment the agent tries to look on Q-network that approximated in order to decide how to act. It called that this approach that combination of random behavior and according to Q-network, as an ϵ-greedy method, which just means changing between random and Q-policy using the probability hyper parameter ϵ.

The core of presented Q-Learning algorithm is derived from the supervised learning. Here, as it was mention above, the goal is to approximate a complex, nonlinear function $Q(s_t, a_t)$ with a CNN. Similarly, to supervised learning, in DQN, we can define the loss function $E(\theta_t)$ as the squared difference between the target and predicted value, and we will also try to minimize the loss by updating the weights (assuming that the agent performs a transition from one state s_t to the next state s_{t+1} by performing some action a_t and receive a reward r).

$$E(\theta_t) = \{r + \gamma \max_{a_{t+1}} Q\left(s_{t+1}, a_{t+1}; \theta_{t+1}\right) \\ -Q\left(s_t, a_t; \theta_t\right)\}^2 \tag{1}$$

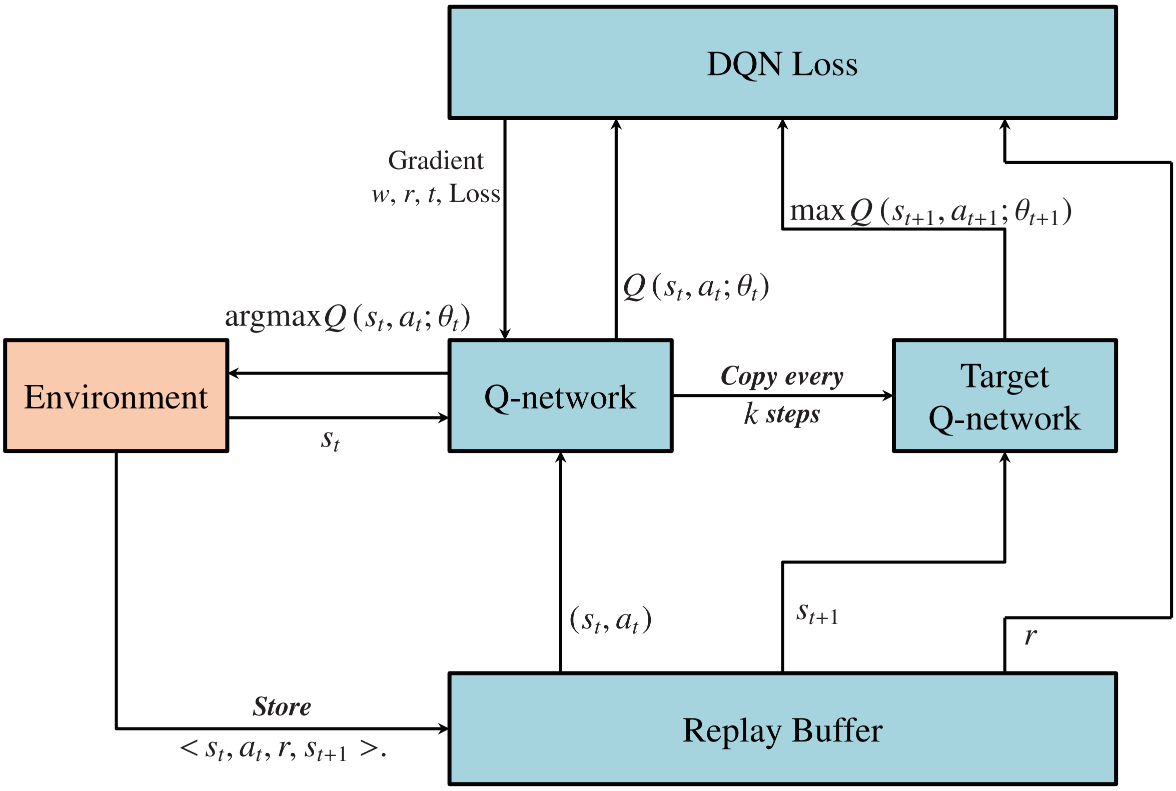

Figure 1. A Concept of DQN Algorithm[12].

During the learning process we use two separate Q-networks to calculate the predicted value (weights θ_t) and target value (weights θ_{t+1}). The target network is frozen for several time steps and then the target network weights are updated by copying the weights from the actual Q-network. Freezing the target Q-network for a while and then updating its weights with the actual Q-network weights stabilizes the training. In order to make training process more stable, this method would like to avoid learning network on data which is relatively correlated, which can happen if we perform learning on last transition. To achieve this, it had applied replay buffer which memorizes experiences of the Agent behavior. Then, training is performed on random samples from the replay buffer. From this viewpoint, that reduces the correlation between the agent's experience and helps the agent to learn better from a wide range of experiences. The DQN algorithm can be describes as follows:

(1) Initialize replay buffer $\mathcal{D}$ to capacity $\mathcal{N}$.

(2) Pre-process, the environment, and feed state s to DQN, which will return the Q-values of all possible actions in the state.

(3) Select an action using the ϵ-greedy with the probability ϵ, we select a random action a and with probability $1-\epsilon$. Select an action that has a maximum Q-value, such as $a = \text{argmax} Q(s, a, \theta)$.

(4) After selecting the action a, the agent performs chosen action in a state s and move to a new state s_{t+1} and receive a reward r.

(5) Store transition in replay buffer as $< s_t, a_t, r, s_{t+1} >$.

(6) Next, sample some random batches of transitions from the replay buffer and calculate the loss using the formula (1).

(7) Perform gradient descent with respect to actual network parameters in order to minimize this loss.

(8) After every k steps, copy our actual network weights to the target network weights.

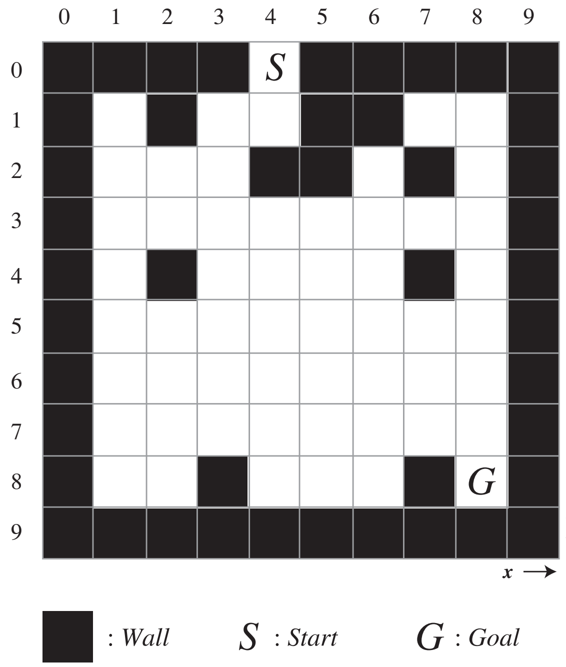

Figure 2. Simulational Environment for Experiment 1.

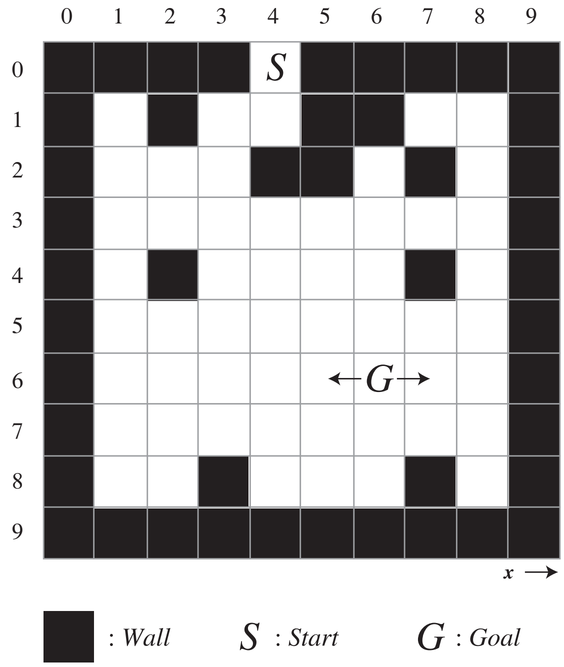

Figure 3. Simulational Environment for Experiment 2.

(9) Repeat these steps for M number of episodes.

3 VERIFICATION EXPERIMENT

3.1 The Outline of Experiment

We verify the characteristics up to the previous section by computer simulation. The characteristics are evaluated by comparing the difference of the convergence speed of Deep Q-Network with Reinforcement Learning. At this time, each techniques are to learn the shortest path that reaches the goal while avoiding walls through trial and error. The behavior will be selected according to facing the state. Also consider the maze environment with walls and pitfalls consisting of a grid of 10×10 shown in fig. 2 or 3 as the experimental environment.

Moreover, the agents implemented two techniques will be affected by transition of goal grid during task execution. In figures 2 and 3, the black grid is the wall, 'S'-marked grid and 'G'-marked grid are each start grid and goal grid. The agents that applied two techniques are perfect perception and can move up, down, left and right of the grid. The agent will be getting a reward 100 when agent reaches the goal grid.

3.2 Simulation Conditions

In this experiment, we mainly deal with episodic tasks: an agent is an agent that operates with RL and another agent is an agent that operates with DQN. Treat the following as one episode: when each agent reaches the goal grid from the start grid, the reward is obtained and the process returns to the start grid. In this experiment, 20,000 episodes have been operated each techniques. Setting of experimental parameters is as shown in the following tables 1 through 3.

Table 1. Learning Parameters of RL for Verification Experiment

Property	Value
Discount rate γ	0.9
Learning rate α	0.2
Explore ratio ϵ	0.2
Initial Q-value	0.0

Table 2. Network Layer of DQN for Verification Experiment

Layer	Output
Dense	128
Flatten	256
Dense	128
Dense	128
Dense	1

Table 3. Learning Parameters of DQN for Verification Experiment

Property	Value
Discount rate γ	0.9
Learning rate α	0.0001
Initial explore ratio ϵ_{init}	1.0
Final explore ratio ϵ_{fin}	0.0

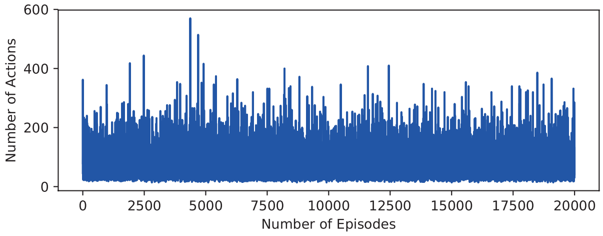

Figure 4. Number of Action per Episodes using RL (1).

3.3 Discussion on Simulated Results

3.3.1 Verification Experiment 1 – In Case of Goal Grid is Fixed

In this experiment, the goal grid is fixed during episodes.

Figures 4 and 5 are the transition of the behavior in each episode by two techniques. The initial value of learning is the number of the behaviors. From these results we can confirm that almost converging to minimum steps. However, the behavior that applied RL seems randomly than DQN. This symptom is caused that RL had fixed exploration ratio during the simulation. On the other hand, exploration ratio of DQN will be decreasing to zero during proceeds of episodes. From the above, the behavior was realized and affected from exploration ratio, will be confirmed.

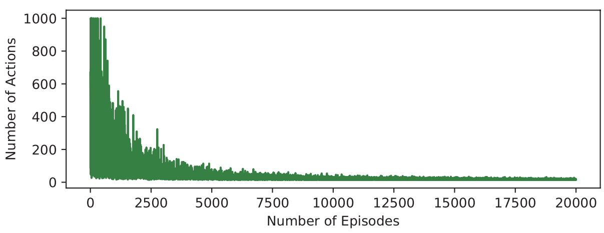

Figure 5. Number of Action per Episodes using DQN (1).

3.3.2 Verification Experiment 2 – In Case of Goal Grid is Changed per Episode

In this experiment, the goal grid will be changed per episodes. In detail, initial position of goal is on (6, 6). Next episode, the goal position will be moved to (5, 6) or (7, 6). Then, the goal will be moved from (5, 6) to (6, 6) or from (7, 6) to (6, 6).

Figures 6 and 7 are the transition of the behavior in each episode by two techniques. The initial value of learning is the number of the behaviors. From these results we can confirm that steps of each agents are divergent or oscillate. However, the behavior that applied DQN seems explosion in latter half of episodes than RL. This symptom is caused that RL had fixed exploration ratio during the simulation in strong contrast to fixed goal grid. In exploration ratio of DQN will be decreasing to zero during proceeds of episodes. In the situation, the agent applied DQN will be lost sight the goal grid transition in latter half of episodes. From the results of each goal grid position, it will be demand that a method to dynamically adjust the action-decision strategy based on behavioral results. In detail, it will be needed the evaluation mechanism that the exploration ratio will be increase or decrease for adjust to the current situation.

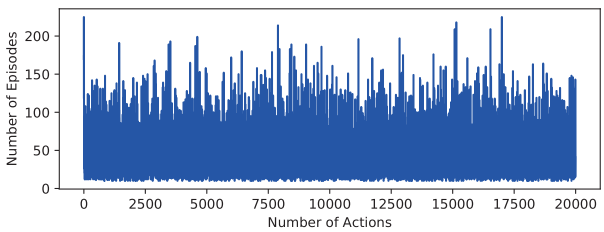

Figure 6. Number of Action per Episodes using Q-Learning (2).

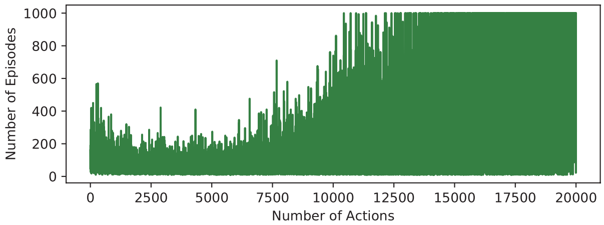

Figure 7. Number of Action per Episodes using DQN (2).

4 CONCLUSION

This paper has focused on the goal tracking ability of Deep Q-Network. In conventional Reinforcement Learning, the exploration ratio is fixed during certain episodes. On the other hands, in case of DQN, the exploration ratio will be decreased per episodes in certain step, have been defined. From these methods, RL is better in case of moving goal grid, is confirmed by fig.6. On the other hands, DQN is optimize the shortest path finding per episodes by decreasing an exploration ratio during progress of episodes. Thus, this method is better in case of fixed goal grid, is confirmed by fig.5.

As the result of verification simulation, Deep Q-Network had been not good at tracking the goal transition, have been confirmed. Therefore, it will be needed that a method for the exploration ratio will be increase or decrease dynamically adjust the action-decision strategy based on the agent's behavioral results.

As the future works, it is necessary to investigate the following points:

- Improvement of the learning speed by exchanging information technique between each agents applied DQN, and consideration an exploration ratio's correction mechanism[13].
- Evaluation of task execution result under imperfect perception, whether task execution is fast by information exchange when incomplete perception and complete perceived agents coexist[14].
- Verification experiment on actual environment when the implementing DQN on mobile robots mounted with LiDAR and Jetson Nano.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series), *The MIT Press*, 2005.

[2] S. Asaka and S. Ishikawa, "Behavior Control of an Autonomous Mobile Robot in Dynamically Changing Environment," *Journal of the Robotics Society of Japan*, Vol.12 No.4, pp.583-589, 1994.

[3] T. Kanda, H. Ishiguro, T. Ono, *et al.*, "Development of "Robovie" as Platform of Everyday-Robot Research," *IEICE Transactions on Information and Systems, Pt.1 (Japanese Edition)*, Vol.J85-D-1 No.4, pp.380-389, 2002.

[4] International Federation of Robotics, "All-time-high for industrial robots Substantial increase of industrial robot installations is continuing," 2011.

[5] T. Sogo, K. Kimoto, H. Ishiguro, and T. Ishida, "Mobile Robot Navigation by a Distributed Vision System," *Journal of the Robotics Society of Japan*, Vol.17 No.7, pp.1-7, 1999.

[6] J. J. Park, C. Johnson, and B. Kuipers, "Robot Navigation with MPEPC in Dynamic and Uncertain Environments: From Theory to Practice," *IEEE IROS 2012 Workshop on Progress, Challenges and Future Perspectives in Navigation and Manipulation Assistance for Robotic Wheelchairs*, 2012.

[7] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. *The MIT Press*, 1998.

[8] N. Sugimoto, K. Samejima, K. Doya, and M. Kawato, "Reinforcement Learning and Goal Estimation by Multiple Forward and Reward Models," *IEICE Transactions on Information and Systems, Pt.2 (Japanese Edition)*, Vol.J87-D-2 No.2, pp.683-694, 2004.

[9] Y. Takahashi and M. Asada, "Incremental State Space Segmentation for Behavior Learning by Real Robot," *Journal of the Robotics Society of Japan*, Vol.17 No.1, pp.118-124, 1999.

[10] A. Agogino and K. Tumer, "Reinforcement Learning in Large Multi-agent Systems." *In Proc. of AAMAS-05 Workshop on Coordination of Large Scale Multiagent Systems*, 2005.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Playing Atari With Deep Reinforcement Learning," *NIPS Deep Learning Workshop*, 2013.

[12] A. Nair, P. Srinivasan, S. Blackwell, *et al.*, “Massively Parallel Methods for Deep Reinforcement Learning,” *In Proc. of ICML Deep Learning Workshop*, 2015.

[13] M. Sugimoto, “A Study for Dynamically Adjustmentation for Exploitation Rate using Evaluation of Task Achievement,” *International Journal of New Computer Architectures and their Applications*, Vol.8 No.2, pp.53-60, 2018.

[14] M. SUGIMOTO, H. YASHIRO, K. NISHIMURA, *et al.*, “A Study for Improvement for Reinforcement Learning based on Knowledge Sharing Method -Adaptability to a situation of intermingled of complete and incomplete perception under an maze-,” *International Journal of New Computer Architectures and their Applications*, Vol.9 No.2, pp.60-67, 2020.