Actor-Q Based Active Perception Learning System

Katsunari Shibata†, Tetsuo Nishino‡& Yoichi Okabe‡

†Dept. of Electrical & Electronics Engineering, Oita Univ., 700 Dannoharu, Oita 870-1192, JAPAN
‡Res. Ctr. for Advanced Sci. & Tech., The Univ. of Tokyo, 5-3-1 Komaba, Meguro-ku, Tokyo 153-0041, Japan
shibata@cc.oita-u.ac.jp, okabe@okabe.rcast.u-tokyo.ac.jp

Abstract

An active perception learning system based on reinforcement learning is proposed. A novel reinforcement architecture, called Actor-Q, is employed in which Q-learning and Actor-Critic are combined. The system decides its actions according to Q-values. One of the actions is to move its sensor, and the others are to make an answer of its recognition result, each of which corresponds to each pattern. When the sensor motion is selected, the sensor moves according to the actor's output signals. The Q-value for the sensor motion is trained by Q-learning, and the Actor is trained by the Q-value for the sensor motion on behalf of the critic. When one of the other actions is selected, the system outputs the recognition result. When the recognition answer is correct, the Q-value is trained to be the upper limit of the Q-value, and when the answer is not correct, it is trained to be 0.0. The module to compute Q-value and the actor module are both consisted of a neural network, and are trained by Error Back Propagation. The training signals are generated based on the above reinforcement learning.

It was confirmed by some simulations using a visual sensor with non-uniform visual cells that the system moves its sensor to the place where it can recognize the presented pattern correctly. Even though the Q-value surface as a function of the sensor location has some local peaks, the sensor was not trapped and moved to the appropriate direction because the Q-value for the sensor motion becomes larger.

Key Words: Actor-Q Architecture, Reinforcement Learning, Neural Network, Active Perception, Visual Sensor

1 Introduction

Our living creatures obtain a variety of informations about the environment through our sensors, and utilize the information to generate our appropriate actions. However, the information of the environment is too huge, it is very inefficient to obtain all the detailed information. To solve this problem, we can move our sensors actively and obtain necessary information effectively. It is called "active perception".

When our visual sensor, which obtains the largest amount of information among our sensors, is observed, the distribution of the visual cells is non-uniform on the retina. We take a general view of the environment or target object by using whole the sensor, then move the dense part of the sensor to the appropriate place, and finally recognize the target correctly. The knowledge that tells us which part of the target should we focus on, cannot be thought inherited, but it is obtained by learning after our birth.

Actually, the system, in which a visual sensor moves, has been developed[1]. However, the system does not learn where its attention should be moved for appropriate recognition, but the main purpose is the pursuit of a moving object, and the captured image is processed by a given way.

On the other hand, reinforcement learning has been focused recently by its autonomous, adaptive, purposive learning. Mainly it is utilized to learn the action planning, but it is expected to be utilized to learn the whole process from sensors to motors including recognition, attention, and so on by employing neural networks[2]. The sensory signals are put into the neural network directly and the output signals are dealt as the motion commands.

It has been tried that the reinforcement learning is applied to active perception systems. In the system by Whitehead et al.[3], the block relocation task is employed, and "which block the attention frame should be moved to" is trained by Q-learning[4], Though the target block of the attention is selected using the Q-values, the motion of the sensor was not considered. Further, the recognition is not dealt with explicitly.

In the system by Shibata et al., visual sensory signals are put into a layered neural network directly, and the neural network generates the motion commands for the visual sensor and the recognition results[5]. However, there are three problems. At first, the reinforcement signal, which is a continuous scalar signal representing how the recognition outputs are close to the ideal ones, has to be given at every time step when the visual sensor makes a step motion. Secondly, the sensor is sometimes trapped at the place where the value function has a local peak, and the system makes a mistake. Third one is that the system does not make a recognition answer at the moment when it becomes to be able to recognize, but the timing when the system makes a recognition answer is fixed.

In this paper, an architecture is proposed not to

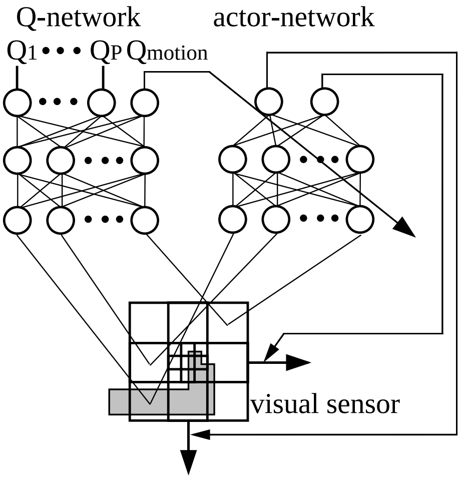


Figure 1: The active perception learning system based on Actor-Q architecture proposed in this paper.

be trapped at a local maximum on the value function surface, and to reach the global maximum. By this architecture, the timing to output the recognition result is also obtained through learning. The evaluation of the recognition result is not required at every time step, but the binary reinforcement signal, a correct (reward) or mistake (no reward), is given to the system only when the system outputs the result. That is similar to the experiments to examine the recognition ability of monkeys.

2 Actor-Q Architecture

Fig. 1 shows the active perception learning system based on Actor-Q architecture that is proposed in this paper. There are two layered neural networks, Q-network and actor-network, and the input of both networks are visual sensory signals. Because the input signals are the same, it is possible that only one neural network makes a role of the both. The outputs of the first network are used as Q-values. One of them is Q-value for the sensor motion and each of the others is Q-value for recognition of each pattern. This means that making a recognition output is considered as one action as well as moving the sensor, and one action is selected using these Q-values.

When it is selected to move the visual sensor, the sensor is moved according to the outputs of the actor network. The two outputs are used as the velocity of the sensor in the direction of x and y respectively. After the sensor motion, a new image caught by the visual sensor is put into the neural networks again, and an action is selected again according to the new Q-values. The Q-value for the sensor motion is trained by popular one-step Q-learning[4]. The training signal is computed as

$$Q_{training}(s(t), motion) = \gamma \max_a Q(s(t+1), a), \quad (1)$$

where γ: a discount factor. The neural network is trained by Error Back Propagation, but the other outputs of the Q-network are not trained. Note that the transformation between the output of the network and Q-value is necessary.

The velocity of the sensor along each of x and y axis is decided by the sum of the output $\mathbf{o}_m$ and the random number **rnd** as

$$\mathbf{m} = \beta(\mathbf{o}_m + \mathbf{rnd}), \quad (2)$$

where β: a constant. The actor-network is trained by the training signal as

$$O_{m,training} = O_m + \mathbf{rnd}(\gamma \max_a Q(s(t+1), a) - \max_a Q(s(t), a). \quad (3)$$

While in the actor-critic architecture[6], it is trained according to the change of the critic output.

When one of the other actions, that means one of the recognition outputs is selected, the recognition output is evaluated whether it is correct or not, and the trial finishes. If the output is correct, the corresponding output of the Q-network is trained to be $Q = 1.0$, and if not correct, it is trained to be 0.0. This corresponds that some juice is given to the monkey when it makes a correct answer, and some penalty is given when it makes a mistake. The flow chart of this learning is shown in Fig. 2.

3 Simulation

3.1 Task Setting

The visual sensor employed in this paper is as shown in Fig. 3. The sizes of the sensory cells are not uniform, such that it is small around the center of the sensor, and it is large at the fringe. The size of the small one is 0.5×0.5, while that of the large one is 1.5×1.5. The sensor has 9 small cells and 8 large cells, and totally 17 cells. The output of each visual sensory cell is the area ratio occupied by the projected pattern against its receptive field. When the signals are put into the neural network, they are linearly expanded from -1.0 to 1.0. The initial location of the sensor is chosen randomly under the condition that the sum of all the sensory signals is larger than 0.5.

The sets of presented patterns are shown in Fig. 4. In the first set, the difference among the 4 patterns exists around the upper-left corner not depending on the presented pattern. The smallest square in the pattern is just the same size as the small sensory cell. When the sensor catches the center of the presented patterns, it is difficult to identify the pattern. So the sensor is required to move its center at the upper-left corner of the pattern.

In the second set, in order to identify the pattern 1 or 2, the sensor should move its center to the upper-right corner of the pattern, while it should moved the

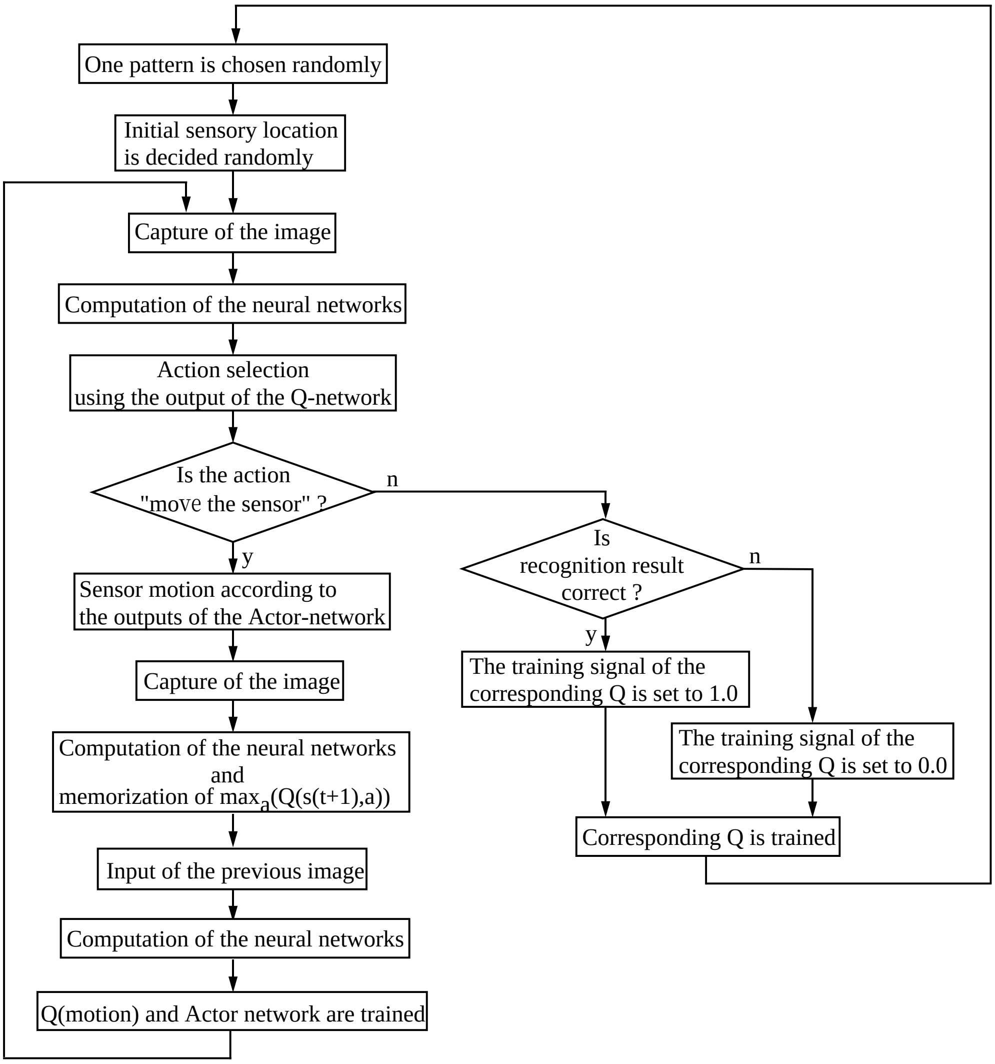

One pattern is chosen randomly
Initial sensory location is decided randomly
Capture of the image
Computation of the neural networks
Action selection using the output of the Q-network
Is the action "move the sensor" ?
n
y
Sensor motion according to the outputs of the Actor-network
Capture of the image
Computation of the neural networks and memorization of max_a(Q(s(t+1),a))
Input of the previous image
Computation of the neural networks
Q(motion) and Actor network are trained
Is recognition result correct ?
n
y
The training signal of the corresponding Q is set to 1.0
The training signal of the corresponding Q is set to 0.0
Corresponding Q is trained

Figure 2: The flow chart of the proposed learning.

center to the upper-left corner to identify the pattern 3 or 4. So the visual sensor is required at first to know which group the presented pattern belongs to using whole the sensor, and then to move its center to the appropriate location.

Here the both neural networks, the Q-network and the actor-network have three layers. The number of hidden neurons is 30 in the Q-network, and 10 in the actor-network. The bias value is not introduced to the output layer in the both networks. That is because the bias sometimes leads to instability of learning or generates a constant flow of the sensor motion not depending on the sensor location.

Since the value range of the each neuron's output function is from -0.5 to 0.5, the training signal for the neural network is obtained from $Q_{training}$ in Eq. (1) and Eq. (3)by the transformation as

$$O_{training} = \alpha(Q_{training} - 0.5) \tag{4}$$

where α: a constant. While Q-value is transformed from the output of the network as

$$Q = O/\alpha + 0.5. \tag{5}$$

Here 0.8 is employed as α to avoid the saturation range of the output function. If the Q becomes less than 0.0, Q is set to be 0.0. As a discount factor γ in Eq. (1), 0.99 is employed.

In Eq. (2), 0.4 is employed as β. So the maximum motion step is 0.2 for each axis, while the minimum square of the pattern and the smallest sensor cell is 0.5×0.5. The random numbers **rnd** are the uniform random number powered by 3.0. The range is from -1.0 to 1.0, while the output range of the network is from -0.5 to 0.5.

One action is selected according to Boltzmann Distribution in the learning phase, and is selected according to the greedy method in the execution phase using the Q-values. The temperature is reduced gradually from 1.0 to 0.01 according to the progress of the learning as shown in Fig. 5. The number of the trial

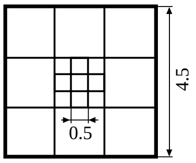
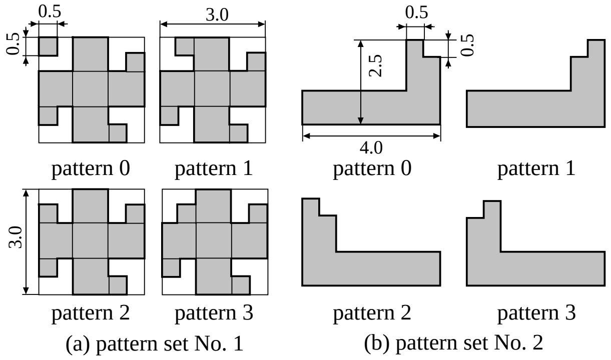


Figure 3: The visual sensor with non-uniform visual cells employed in this paper.



(a) pattern set No. 1

(b) pattern set No. 2

Figure 4: The presented pattern sets.

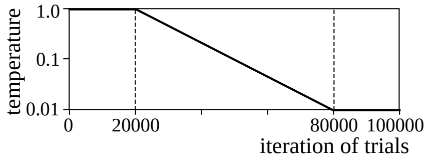


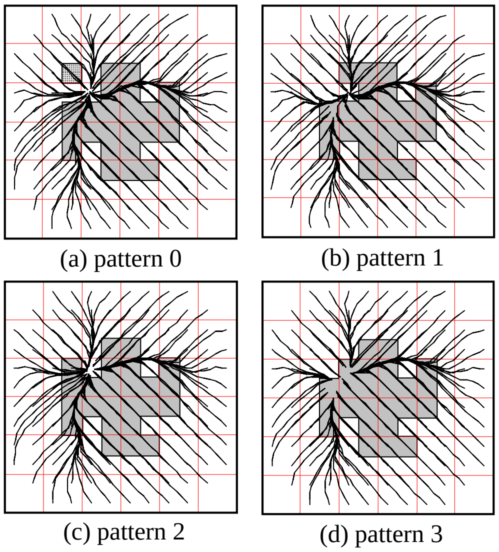
Figure 5: Temperature cooling schedule that is used in the action selection (log scale).

iterations is 100000.

3.2 Result

5 simulation runs are done for each pattern sets varying the initial weight values in the neural network, the initial sensor location, and the order of the presented patterns. In all the simulation runs, the system could identify the presented pattern after some motions. Fig. 6 shows an example of the trajectories of the visual sensor when the pattern set No. 1 is presented. It is seen that the system moves the center of its sensor to the upper-left corner of the presented pattern from given 132 initial sensor locations those are on the grid with 0.25 width. It is noticed that when the pattern 0, 1, or 2 is presented, the sensor moves and converges to almost one point, and the small square that is the difference from the other pattern is caught just by one of the small size sensory cells. This seems an effective and sure way of identification. While when the pattern 3 is presented, the convergence area is wider. That is the general property observed over the 5 simulation runs.

Fig. 7 shows an example of the trajectories of the visual sensor when the pattern set No. 2 is presented.



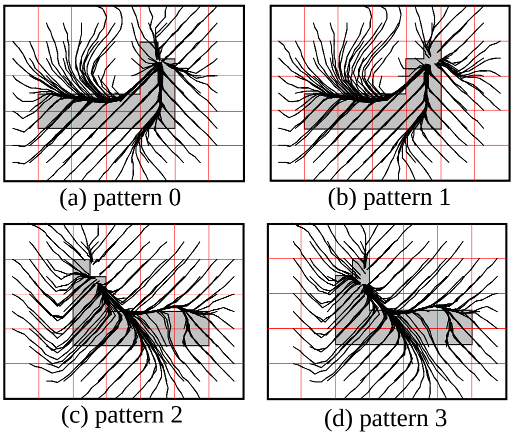
(a) pattern 0

(b) pattern 1

(c) pattern 2

(d) pattern 3

Figure 6: The trajectories of the visual sensor when the pattern set No. 1 is presented.



(a) pattern 0

(b) pattern 1

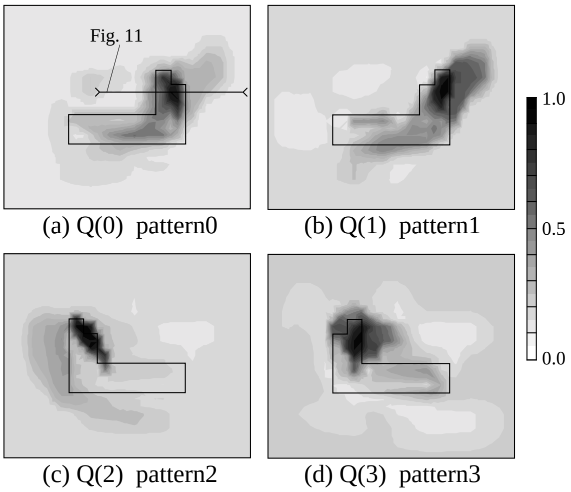
(c) pattern 2

(d) pattern 3

Figure 7: The trajectories of the visual sensor when the pattern set No. 2 is presented.

It can be seen that the system moves its sensor to the appropriate direction depending on the presented pattern. Concretely when the presented pattern is 0 or 1, the sensor moves to the upper-right corner, while when the pattern is 2 or 3, it moves to the upper-left corner. The sensor catches the small difference between the patterns on the center after a series of motions. Finally it could make a correct answer for each of the about 132 initial locations of the sensor.

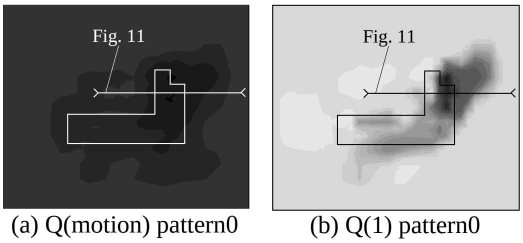
Fig. 8 shows the distribution of the Q-values corresponding to each presented pattern. It is seen that the Q-value is large around the small difference area, and the area is the same as the location where the visual sensor converges as shown in Fig. 7. Fig. 9 shows the distribution of the Q-value for the sensor motion and the Q-value for the pattern 1 when the pattern 0 is presented. It is seen that the Q-value for the sensor



(a) Q(0) pattern0 (b) Q(1) pattern1

(c) Q(2) pattern2 (d) Q(3) pattern3

Figure 8: The distribution of the Q-values that is corresponding to the presented pattern.



(a) Q(motion) pattern0 (b) Q(1) pattern0

Figure 9: The distribution of the Q-value for the sensor motion and Q-value for the pattern 1 when the pattern 0 in the pattern set No. 2 is presented.

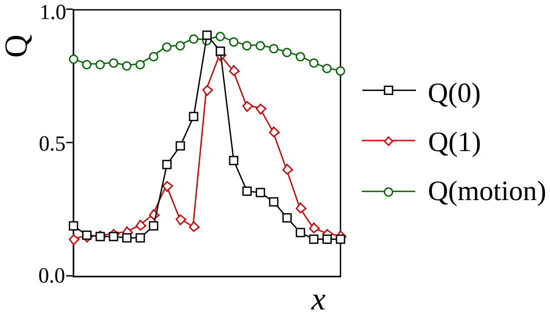


Figure 10: The one-dimension of distribution of the Q-values when the sections of the Q-value surfaces, Fig. 9(a), Fig. 10(a), and (b), are observed.

motion is always large not depending on the sensor location because the discount factor is close to 1.0. The Q-value for the pattern 1 is large around the upper-right corner.

Since it is difficult to see which value is larger at one sensor location, the section of the Q-value surface as shown in Fig. 8(a), Fig. 9(a),(b), is observed. Fig. 10 shows the distribution of the Q-values along the section. It can be seen that only at one point, the Q-value for the pattern 0 is slightly larger than Q-value for the sensor motion, while the Q-value for the pattern 1 is always smaller than the Q-value for the motion. The Q-value for the sensor motion is reduced gradually from the maximum value, while the other Q value reduced suddenly. So even if the Q-value surface of one pattern has a local maximum, the Q-value for the sensor motion is larger, and the system selects to move the sensor. Accordingly the sensor is not trapped at the local maxima.

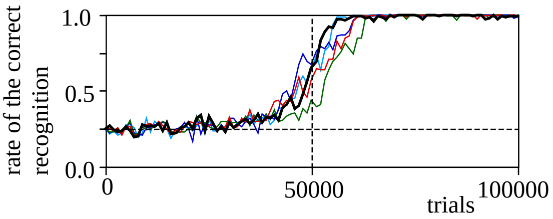
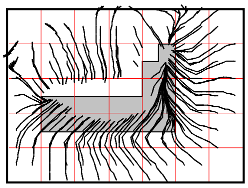


Figure 11: Learning curve when the pattern set No. 2 is presented. The y axis indicates the probability of the successful recognition.



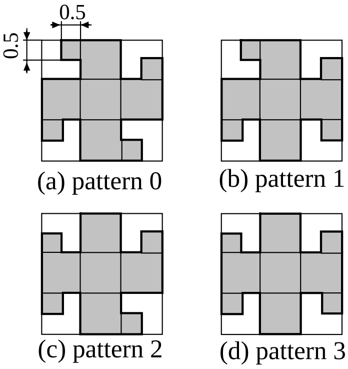
pattern 1

Figure 12: The trajectories of the visual sensor when the pattern 1 is presented after 50000 trials of learning.

Fig. 11 shows the learning curve of 5 simulation runs when the pattern set No. 2 is presented. All the learning curve is similar, and the goal probability becomes large around 50000 trials. Fig. 12 shows the sensor trajectories after 50000 trials. The system makes an answer when the center of the sensor arrives on the pattern. In this case, the system makes the answer that the presented pattern is 0 even when the presented pattern is 1. When the sensor motion is selected, the Q-values for recognition are not trained. So the area of the sensor location where the Q-value for recognition is trained becomes smaller according to the progress of the learning. Then the Q-value surface for recognition becomes to have a strong peak. As above, the learning of Q-value and the learning of the motion make progress giving an effect with each other.

3.3 Simulation of Context Inputs

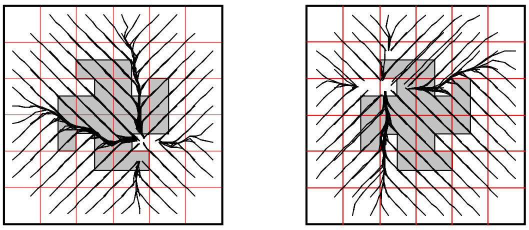
Next, it is examined that the system can generate the different series of sensor motions depending on the context inputs. The pattern set as shown in Fig. 13 is given. Each of the patterns cannot be identified even if the sensor goes to one of the corners of the pattern. For example, the difference between the pattern



(a) pattern 0 (b) pattern 1

(c) pattern 2 (d) pattern 3

Figure 13: The pattern set in which the system requires the context inputs to identify each presented pattern.

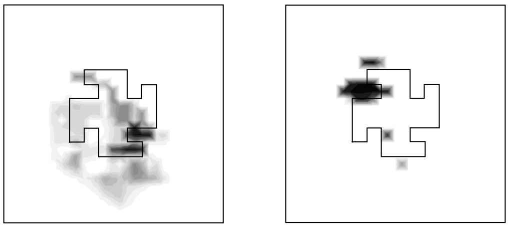


(a) pattern 0 (context: 0 or 1) (b) pattern 0 (context: 0 or 2)

Figure 14: The difference of the sensor trajectories depending on the context inputs.

0 and 1 exists at the bottom-right corner of the pattern, while the difference between the pattern 0 and 2 exists at the upper-left corner. So when the sensor reaches the upper-left corner, it cannot identify whether the pattern is 0 or 1, while when the sensor reaches the bottom-right corner, it cannot identify whether 0 or 2. The context input that consists of 4 signals indicating the possibility that the presented pattern can be each of the 4 patterns, is also given to the neural network. In this case, only two of the signals are 2, which means a possible pattern, and the other two are -2. In this simulation, the number of the trial iterations is 2000000, and the number of the hidden neurons is 50 in the Q-network, and 20 in the actor-network. The discount factor γ is set to be 0.96. The temperature is reduced as Fig. 5, but the x axis is expanded linearly.

The difference of the sensor trajectories depending on the context inputs when the pattern 0 is presented is shown in Fig. 14. It can be seen that when the context inputs shows the possibility 0 or 1, the system moves its sensor to the bottom-right corner, while when the context shows the possibility 0 or 2, it moves its sensor to the upper-left corner. However, it is far more difficult to learn these motions than the previous simulation runs, and it needs 2000000 trials for learning. Fig. 15 shows the distribution of the Q-value for the pattern 0. It can be seen that the distribution is perfectly different from each other even if the visual sensory signals are completely the same.



(a) Q(0) pattern 0
(context: 0 or 1)

(b) Q(0) pattern 0
(context: 0 or 2)

Figure 15: The difference in the Q-value distribution depending on the context inputs.

4 Conclusion

Q-actor architecture and the learning algorithm have been proposed for the active perception system based on reinforcement learning. Through the simulation using a visual sensor with non-uniform sensory cells, it was confirmed that the system becomes to move its sensor to the place where the difference between the patterns exists, and then to output the correct recognition result.

Acknowledgments A part of this research was done under the Grant-in-Aid for Scientific Research on Priority Area No. 264 (System Theory of Function Emergence) supported by the Scientific Research Foundation of the Ministry of Education, Science, Sports and Culture of Japan.

REFERENCES

[1] S. Rougeaux and Y. Kuniyoshi, "Robust Real-Time Tracking on an Active Vision Head", *Proc. of IROS'97* (1997)

[2] Shibata, K., Okabe, Y. & Ito, K, “Direct-Vision-Based Reinforcement Learning in “Going to an Target” Task with an Obstacle and with a Variety of Target Sizes”, *Proc. of NEURAP'98*, pp. 95–102 (1998)

[3] S.D.Whitehead & D.H.Ballard, "Learning to Perceive and Act by Trial and Error", *Machine Learning*, **7**, pp. 45–83 (1991)

[4] Watkins, C. J. C. H. and Dayan, P., "Q-Learning", *Machine Learning*, **8**, pp. 279–292 (1992)

[5] K. Shibata, T. Nishino & Y. Okabe, “Active Perception based on Reinforced Learning”, *Proc. of WCNN '95*, **II**, pp. 170–173 (1995)

[6] Barto, A. G., Sutton, R. S. & Anderson C. W., “Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems,” *IEEE Trans. of SMC*, **13**, pp. 835–846 (1983)