An Acquisition of the Relation between Vision and Action using Self-Organizing Map and Reinforcement Learning

Kazunori Terada, Hideaki Takeda and Toyoaki Nishida

Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-01, Japan
E-mail : kazuno-t@is.aist-nara.ac.jp

Abstract

An agent must acquire internal representation appropriate for its task, environment, sensors. As a learning algorithm, reinforcement learning is often utilized to acquire the relation between sensory input and action. Learning agents in the real world using visual sensors is often confronted with critical problem how to build necessary and sufficient state space for the agent to execute the task. In this paper, we propose acquisition of relation between vision and action using Visual State-Action Map (VSAM). VSAM is the application of Self-Organizing Map (SOM). Input image data is mapped on the node of learned VSAM. Then VSAM outputs the appropriate action for the state. We applied VSAM to real robot. The experimental result shows that a real robot avoids the wall while moving around the environment.

1 Introduction

A sensor based agent performing a task in an environment must acquire internal representation about its environment. In order to adapt uncertain and unforgiving environments, an agent should acquire the internal representation by interacting with its environment. Evaluating current sensory input using internal representation, an agent can output an appropriate action for the situation which the agent faces.

An agent gets information about its environment through various sensors, e.g., visual sensor, tactile sensor and sonar sensor. Tactile sensor differs with the other sensors about the point that it outputs a signal when the body of the agent physically touches an object. A crucial state for agent's task such as colliding with a obstacle or grasping an object is perceived through only tactile sensors. On the other hand, using a visual sensor or a sonar sensor, an agent can estimate what happen in the future result from performing selected action. However, this is correct only under following assumption. Visual input represent 3 dimensional space. In other words, an agent can reconstruct 3 dimensional space from visual input. A newborn agent should not know even this assumption. Therefore, an agent should acquire relation visual input and physical world by interacting with its environment.

Many works have addressed on acquiring a behavior or constructing internal representation through interaction between an agent and its environment. In the field of computer vision, *purposive vision* have proposed [1] [2]. In simple terms, this approach suggests that vision has a purpose, and Perception Action system should be organized by behavior. Pfeifer [8] proposed the concept of *sensory-motor coordination*. It means that all interaction with the environment is to be conceived as a sensory-motor coordination.

As a learning algorithm, reinforcement learning is often utilized. Reinforcement provides agents with the capability of learning to act optimally by experiencing the consequences of actions, without any domain knowledge. The interaction between an agent and the environment is only focused upon. Learning agents in the real world using visual sensors is often confronted with critical problem how to build necessary and sufficient state space for the agent to execute the task. This problem includes two problems.

1. How to reduce the size of the state space into proper size with which reinforcement learning can be performed. An abundance of visual sensors has resulted in a state space so large that it overwhelms the agent's limited

resources for computation, storage and learning experience.

2. How to select the appropriate axes of state space. For a given tasks, an agent must focus its attention upon only necessary features in enormous visual input data. Axes of the state space are often selected depending on human intuition.

This two problems is often handled as combined problem. Nakamura and Asada [7] have proposed a method using a *motion sketch* which closely relates robot actions to sensor patterns for reducing the size of the state space. This approach uses features selected by humans such as optical flows. MaCallum [6] has proposed an approach using *U-Tree*, a reinforcement learning algorithm that uses selective attention and short term memory. This approach uses features previously selected human and some features appropriate for tasks are automatically selected. Ishiguro and Sato [3] have proposed an approach using *Empirically Obtained Perceivers* which classifies sensor patterns by statistically analyzing the actions, sensor patterns, and rewards given as results of task executions. However, they performed their experiments in simulated world.

In recent researches of learning agent performing in the real world, raw image data was not used. Instead of using raw image data, many researchers have used processed image. This paper proposes an approach to construct a state space using *Visual State-Action Map (VSAM)* which is the application of Self-Organizing Map (SOM)[4]. VSAM is internal representation of the agent. Input image data is mapped on the node of VSAM. Then VSAM outputs the appropriate action for the state. Off-line reinforcement learning is performed on VSAM and appropriate relation between vision and action is acquired.

A similar works to our research approach have been already reported. Kröse [5] proposed a sensor based navigation scheme which makes use of a global representation of environment by means of a self-organizing map. They performed their work in simulation and used 16 range sensors. We, rather focus upon real world agent and acquiring relation between vision and action.

2 Architecture

The purpose of this system is to output appropriate action for the state. For this purpose the system have two features. First, we use the nodes of SOM as state space, and each node of SOM is the cluster of image data. Second, to decide the appropriate outputs for each state, we use reinforcement learning.

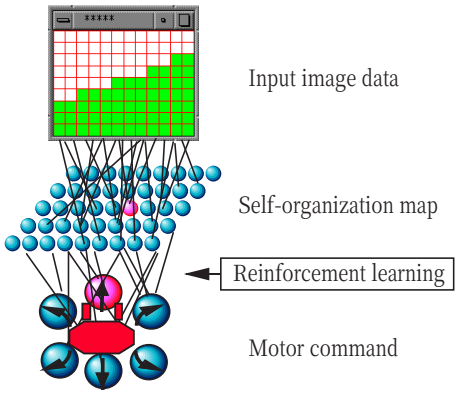


Figure 1: Architecture

Furthermore, to construct necessary and sufficient state space for the agent, we used following assumption. If the state space consists of n pixels of binary image data, the number of whole state space becomes 2^n. However, the state space contains unsuitable states which does not really exist. We consider one image pattern as one state.

The input of this system is the vector of the image data and the output is an action. The input vector is clustered and mapped to a unit of self-organizing map (SOM) which is used as internal state space. To perform off line learning, each units of SOM has a set of possible action and next state resulted from performing a selected action. Then relation between the states and actions is computed using reinforcement learning. We call this architecture *Visual State-Action Map* (VSAM).

Figure 1 shows the architecture of VSAM. The process of constructing VSAM is decomposed into three phases.

1. Clustering of image data previously collected using self-organizing map.

2. Exploration of the environment. During exploration, the image data was mapped on the winning node of SOM. Then, the winning node and corresponding action was stored, and estimation of the state transition probability $p_{ij}(a)$ is made. $p_{ij}(a)$ is defined as the probability that, given an initial state i, the agent is in the state j after action a.

3. Learning. Since transition of each state was calculated, reinforcement learning in SOM was performed.

3 Clustering of images

3.1 Self-Organizing Maps

The SOM defines a mapping from the input data space $\boldsymbol{R}^n$ onto several dimensional array of nodes. With every node i, a parametric reference vector $\boldsymbol{m}_i = [\mu_{i1}, \ldots, \mu_{1n}]^T \in \boldsymbol{R}^n$ is associated. An input vector $\boldsymbol{X} = [x_1, \ldots, x_n]^T \in \boldsymbol{R}^n$ is connected to all nodes in parallel via variable scalar weights μ_{ij} and be compared with all the $\boldsymbol{m}_i$. The smallest of Euclidean distances $\|\boldsymbol{X} - \boldsymbol{m}_i\|$ can be made to define the winner node, signified by the subscript c.

$$\|\boldsymbol{X} - \boldsymbol{m}_c\| = \min_i \{\|\boldsymbol{X} - \boldsymbol{m}_i\|\} \tag{1}$$

The output y_i of the node i is

$$y_i = \begin{cases} 1 & i = c \\ 0 & \text{otherwise} \end{cases}$$

Learning of SOM is performed by renewing the weight of winner node $\boldsymbol{m}_i$ and a neighborhood set of array points around winner node.

$$\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + h_{ci}(t)[\boldsymbol{X}(t) - \boldsymbol{m}_i(t)] \tag{2}$$

Where $t = 0, 1, 2, \ldots$ is an integer, the discrete-time coordinate. The function $h_{ci}(t)$ is called neighborhood function.

$$h_{ci}(t) = h(\|r_c - r_i\|, t) = \alpha(t) \cdot \frac{(N_c + 1 - \|r_c - r_i\|)}{N_c} \tag{3}$$

Where $r_c \in \boldsymbol{R}^2$ and $r_i \in \boldsymbol{R}^2$ are the location vector of nodes c and i, respectively, in the array. The value of $\alpha(t)$ is identified with a learning-rate factor $(0 < \alpha(t) < 1)$. N_c is a neighborhood set of array points around node c. Both $\alpha(t)$ and the radius of N_c are usually decreasing monotonically in time (during the learning process).

$$\alpha(t) = \alpha_0(1 - \frac{t}{T}) \tag{4}$$

$$N_c = (d_0 - 1)(1 - \frac{t}{T}) + 1 \tag{5}$$

Where α_0 is constant value. d_0 is the initial value of N_c. T is whole learning steps.

3.2 Preprocessing of Optic Patterns

SOM algorithms compare input pattern vector with weight vectors using Euclidean differences, whereby it is not meant to tolerate even insignificant amounts of transformation in elementary operations such as translation, rotation, or scale changes of patterns.

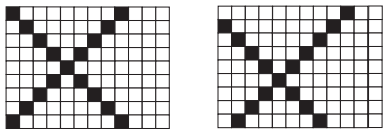


Figure 2: Two line figures that are orthogonal

Consider Figure 2; the two binary patterns are completely orthogonal, although looking very similar to us, and would be interpreted as totally dissimilar by SOM. To avoid this problem, we use blurring which is one of the simplest preprocessing method. If the value of a cell (i, j) is 1, the value of neighborhood cells follows the equations described below.

$$C(d) = \begin{cases} 1 & d = 0 \\ 0.66 & d = 1 \\ 0.33 & d = 2 \\ 0 & d > 2 \end{cases}$$

Where d is the distance from cell (i, j).

4 Reinforcement learning

Reinforcement learning is an attractive framework for the unsupervised learning of action policies for autonomous agents. The tasks are given by a human as rewards embodied in the external world.

Q-learning [9] is based on estimating the value of $Q(s, a)$, which is the expected future discounted reward for taking an action a in the input state s and continuing with the optimal policy. The discounted reward is the sum of all future reward weighted by how close they are. The value of $Q(s, a)$ is renewed.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{b \in A} Q(s_{t+1}, b) - Q(s_t, a_t)) \tag{6}$$

Where t is the present time, γ is a discount factor $(0 < \gamma < 1)$, and r_t is the reward received at time t.

The following is a Q-learning algorithm we used here.

1. $s \leftarrow$ the current state
2. $a \leftarrow$ Select an action randomly
3. $s' \leftarrow$ Next state, $r \leftarrow$ Immediate reward in s'
4. Renew $Q(s, a)$

4.1 Acquisition of Visual State-Action Map

Many applications of reinforcement learning were implemented by simulation, since enormous computational time is required to converge the learning.

To accomplish off-line learning with real data we propose Visual State-Action Map (VSAM) which is the application of SOM. Each node of VSAM has 6 sub-nodes as selectable action and each node has a link to the next state from performed selected action(Figure 3). VSAM also has a slot of reward which is received when the robot touches the wall. The following is the algorithm to acquire the data set of VSAM.

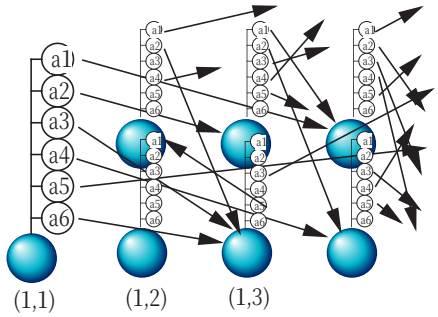
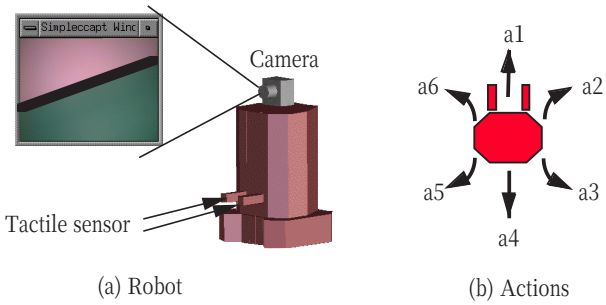


Figure 3: VSAM

1. $a_i \leftarrow$ select an action $(i = 1, 2, \ldots, 6)$
2. $U_t \leftarrow$ define winner node
3. $r \leftarrow$ reward
4. Memorize U_t and r to a_i of U_{t-1}

5 Experimental Results

5.1 Robot and Environment



(a) Robot (b) Actions

Figure 4: Robot

The robot used in this experiment is shown in Figure 4(a). The robot is driven by two motors to which we can send a motor command $-1, 0, +1$ independently. Selecting a combination of each command;

$$a_1(1,1), a_2(1,0), a_3(-1,0)$$
$$a_4(-1,-1), a5(0,-1), a_6(0,1)$$

the robot can move 6 directions (Figure 4(b)). The robot has a CCD camera which outputs a picture of 160×120 pixels at each time step and two tactile sensors in front of the hand.

The environment used this experiment is $90cm \times 180cm$ rectangular space surrounded by $40cm$ height wall. The boundary line between the floor and the wall is marked black. The width of boundary line is $5cm$.

5.2 Generation of SOM

Input vector of SOM used in this experiment is 12×9 cells binary picture which resolution is decreased from 160×120 pixels. If 20% of the pixels in a cell is black, the value of input vector is 1, else 0. With 201 piece of image data set collected by random movements in the environment, SOM as a internal state space has been constructed (Figure 5). Whole learning times was 200 steps. 70 nodes of SOM is mapped from input vector.

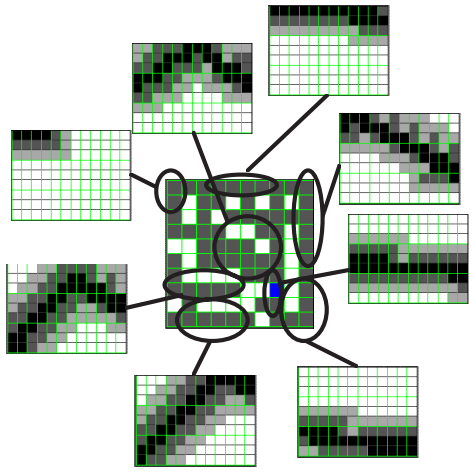


Figure 5: SOM

The nodes in circles drawn in Figure 5 have similar patterns from a view of human intuition. The average of Euclidean distances between neighborhood nodes is 1.4.

5.3 Construction of VSAM

The transition of the state of VSAM is collected by random movements.

Result from executing 600 actions, 317 sets of relation between state and action in 420 possible sets were required. Table 1 is the examples of acquired action sets. For example, if the current state is $(1, 9)$ and a selected action is a4, the next state is $(3, 3)$. Reward is also required, when the tactile sensor in front of the hand of robot touches the wall.

Table 1: Example of VSAM

	a_1	a_2	a_3	a_4	a_5	a_6	reward
1,9	3,3	1,7	0,4	3,3		0,6	0
2,9	2,2	0,6	6,9	3,3	2,5	5,5	0
3,9	3,6	0,5	6,8	3,4	3,5	5,6	0
4,9		0,5	7,9	2,5		7,6	0
5,9	3,9	8,9	8,7	2,6	5,6	7,5	0
6,9				2,9	3,9	,6,8	-1

The value of reward is -1. The blank columns of the table means that the relation was not required.

5.4 Learning

Using the required transition of the state of VSAM, off-line Q-learning was performed. Figure 6 shows the examples of learned Q value which was sufficiently converged. The shaded arrow means the Q value, black arrow means high Q value and white arrow means low Q value. We can see that the Q value of the action toward the wall is low. Under a policy by which a robot selects an action of which Q value is maximum, it seems like that a robot avoid the wall.

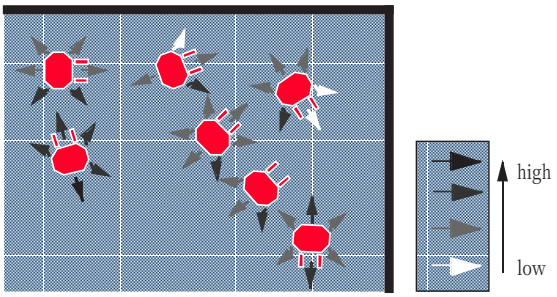


Figure 6: Learned Q value: When agent is going along the wall, Q value of the action toward wall is low and Q value of the action leaving from the wall is high.

6 Experimental result for a real mobile robot

Experiment with real mobile robot ware performed with R3 robot (IS Robotics, Inc.). Figure 7 shows a configuration of system used in this experiment. The motor commands is sent to R3 robot via wireless modem. The image taken by a CCD camera mounted on the robot is transmitted to a

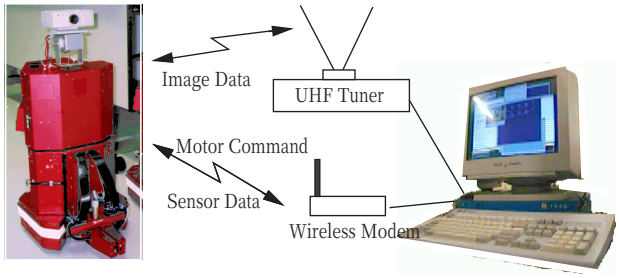


Figure 7: Configuration of system

UHF receiver and sent to SGI workstation through video port.

We applied the VSAM to the real mobile robot. Figure 8 shows the result of this experiment

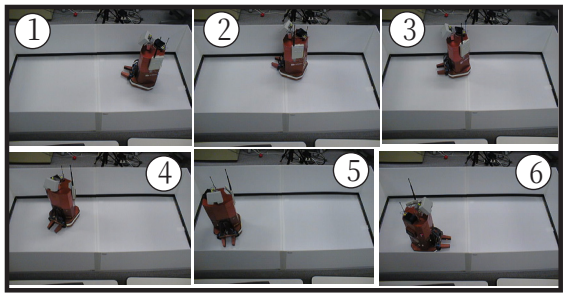


Figure 8: The action avoiding the wall. 1,4,6: The agent turns the corner.; 2,3,5: The agent goes along the wall.

In Figure 8, we can see that the robot avoids the wall. When it comes to the corner, it changes the direction, and run along the wall.

The advantage of having visual sensors is to estimate what happen in the future result from performing selected action. As a consequence of performing reinforcement learning, value of action have calculated. When agent is going along the wall, Q value of the action toward wall is low and Q value of the action leaving from the wall is high. This means that agent previously know the negative reward which is given agent when it collide with wall.

7 Discussion and Conclusion

In this paper, we proposed an approach to the problem which concerns the construction of the state space. Clustering of input image vector using self-organizing map, we have constructed a state space which does not depend upon human intuition. To perform off-line learning with real data, we have used Visual State-Action Map as a model of an environment.

Using VSAM, reinforcement learning which deal with high dimensional state space in the real world was performed. However, in order to apply the proposed method to more complex environment and tasks, the state space constructed on VSAM must be clustered based on certain viewpoint of feature.

To define the axes of state space explicitly, an agent must focus its attention on the changes appeared in the image data. In our work, the agent did not explicitly select the axes of state space after learning, nevertheless, from the view of input generalization, an agent must decide axes which is the feature of action of object.

To recognize the objects in the real world, an agent must know various visual feature. Since visual feature strongly concerns physical feature, acquiring the relation between the tactile sensor inputs and visual input is very important. In our work, agent has a few tactile sensors. An agent must have many tactile sensors to recognize various object of which feature is unique. Finally, we discuss the problem about dynamic environment. Applying the method of VSAM to dynamic environment, state space will multiply. Because one node of VSAM represent one state, and trivial change of image according to change of environment will generate new state. Therefore an agent must consider an object as independent object (not a pattern), and observe critical objects for agent's action.

Acknowledgments

The author thanks Takayuki Nakamura and Atsushi Ueno for discussions about learning algorithm .

References

[1] Yiannis Aloimonos. What i have learned. In *CVGIP: Image Understanding*, pages 60:74–85, 1994.

[2] R. James Firby, Roger E. Kahn, Peter N . Prokopowicz, and Michael J. Swain. An architecture for vision and action. In *International Joint Conference on Artificial Intelligence, IJCAI-95*, pages 72–79, 1995.

[3] R. Sato H. Ishiguro and T. Ishida. Robot oriented state space construction. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems 1996 (IROS96)*, pages 1496–1501, 1996.

[4] T. Kohonen. *Self-Organizing Maps*. Springer, 1997.

[5] B.J.A. Krose and M. Eecen. A self-organizing representation of sensor space for mobile robot navigation. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, pages 9–14, 1994.

[6] A.K. McCallum. Learning to use selective attention and short-term memory in sequential tasks. In *Proc. of The Fourth International Conference on Simulation of Adaptive Behavior (SAB96)*, 1996.

[7] T. Nakamura and M. Asada. Motion sketch: Aquisition of visual motion guided behaviors. In *Proc. of Int. Joint Conference on Aritificial Intelligence*, pages 126–132, 1995.

[8] Rolf Pfeifer and Christian Scheier. Sensory-motor coordination: the metaphor and beyond. In R. Pfeifer and R.A. Brooks, editors, *Robotics and Autonomous Systems. Special Issue on Practice and future of autonomous agents.*, 1996.

[9] C.J.C.H. Watkins and P. Dayan. Technical note: Q-learning. *Machin Learning*, 8:39–46, 1992.