

# We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900

Open access books available

186,000

International authors and editors

200M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index  
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?  
Contact [book.department@intechopen.com](mailto:book.department@intechopen.com)

Numbers displayed above are based on latest data collected.  
For more information visit [www.intechopen.com](http://www.intechopen.com)



# Reinforcement Learning using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution

Yuko Osana  
Tokyo University of Technology  
Japan

## 1. Introduction

The reinforcement learning is a sub-area of machine learning concerned with how an agent ought to take actions in an environment so as to maximize some notion of long-term reward (Sutton & Barto, 1998). Reinforcement learning algorithms attempt to find a policy that maps states of the world to the actions the agent ought to take in those states.

Temporal Difference (TD) learning is one of the reinforcement learning algorithm. The TD learning is a combination of Monte Carlo ideas and dynamic programming (DP) ideas. TD resembles a Monte Carlo method because it learns by sampling the environment according to some policy. TD is related to dynamic programming techniques because it approximates its current estimate based on previously learned estimates. The actor-critic method (Witten, 1977) is the method based on the TD learning, and consists of two parts; (1) actor which selects the action and (2) critic which evaluate the action and the state.

On the other hand, neural networks are drawing much attention as a method to realize flexible information processing. Neural networks consider neuron groups of the brain in the creature, and imitate these neurons technologically. Neural networks have some features, especially one of the important features is that the networks can learn to acquire the ability of information processing. The flexible information processing ability of the neural network and the adaptive learning ability of the reinforcement learning are combined, some reinforcement learning method using neural networks are proposed (Shibata et al., 2001; Ishii et al., 2005; Shimizu and Osana, 2008).

In this research, we propose the reinforcement learning method using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD) (Osana, 2009). The proposed method is based on the actor-critic method, and the actor is realized by the KFMPAM-WD. The KFMPAM-WD is based on the self-organizing feature map (Kohonen, 1994), and it can realize successive learning and one-to-many associations. The proposed method makes use of this property in order to realize the learning during the practice of task.

## 2. Kohonen feature map probabilistic associative memory based on weights distribution

Here, we explain the Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD) (Koike and Osana, 2010) which is used in the proposed

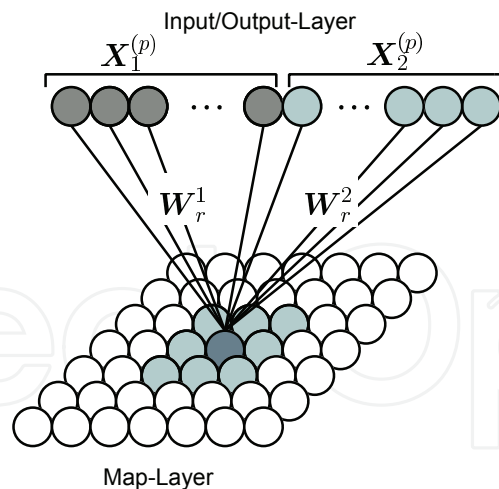


Fig. 1. Structure of KFMPAM-WD.

method.

## 2.1 Structure

Figure 1 shows the structure of the KFMPAM-WD. As shown in Fig.1, the KFMPAM-WD has two layers; (1) Input/Output(I/O)-Layer and (2) Map-Layer, and the I/O-Layer is divided into some parts.

## 2.2 Learning process

In the learning algorithm of the KFMPAM-WD, the connection weights are learned as follows:

- (1) The initial values of weights are chosen randomly.
- (2) The Euclidean distance between the learning vector  $X^{(p)}$  and the connection weights vector  $W_i$ ,  $d(X^{(p)}, W_i)$  is calculated.

$$d(X^{(p)}, W_i) = \sqrt{\sum_{k=1}^M (X_k^{(p)} - W_{ik})^2} \quad (1)$$

- (3) If  $d(X^{(p)}, W_i) > \theta^t$  is satisfied for all neurons, the input pattern  $X^{(p)}$  is regarded as an unknown pattern. If the input pattern is regarded as a known pattern, go to (8).
- (4) The neuron which is the center of the learning area  $r$  is determined as follows:

$$r = \underset{i: D_{iz} + D_{zi} < d_{iz} \text{ (for } \forall z \in F)}{\operatorname{argmin}} d(X^{(p)}, W_i) \quad (2)$$

where  $F$  is the set of the neurons whose connection weights are fixed.  $d_{iz}$  is the distance between the neuron  $i$  and the neuron  $z$  whose connection weights are fixed. In the KFMPAM-WD, the Map-Layer is treated as torus, so the distance between the neurons  $i$  and  $j$   $d_{ij}$  is given by

$$d_{ij} = \sqrt{(d_{ij}^x)^2 + (d_{ij}^y)^2} \quad (3)$$

$$d_{ij}^x = \begin{cases} x_j - x_i, & (|x_j - x_i| \leq x_{max}/2) \\ -\text{sgn}(x_j - x_i)(x_{max} - |x_j - x_i|), & (\text{otherwise}) \end{cases} \quad (4)$$

$$d_{ij}^y = \begin{cases} y_j - y_i, & (|y_j - y_i| \leq y_{max}/2) \\ -\text{sgn}(y_j - y_i)(y_{max} - |y_j - y_i|), & (\text{otherwise}) \end{cases} \quad (5)$$

where  $x_i$  and  $y_i$  are the coordinates of the neuron  $i$  in the Map-Layer,  $x_j$  and  $y_j$  are the coordinates of the neuron  $j$  in the Map-Layer, and  $x_{max}$  and  $y_{max}$  are width and height of the Map-Layer. In Eq.(2),  $D_{ij}$  is the radius of the ellipse area whose center is the neuron  $i$  for the direction to the neuron  $j$ , and is given by

$$D_{ij} = \begin{cases} \sqrt{\frac{a_i^2 b_i^2}{b_i^2 + m_{ij}^2 a_i^2}} (m_{ij}^2 + 1), & (d_{ij}^x \neq 0 \text{ and } d_{ij}^y \neq 0) \\ a_i, & (d_{ij}^y = 0) \\ b_i, & (d_{ij}^x = 0) \end{cases} \quad (6)$$

where  $a_i$  is the long radius of the ellipse area whose center is the neuron  $i$  and  $b_i$  is the short radius of the ellipse area whose center is the neuron  $i$ . In the KFMPAM-WD,  $a_i$  and  $b_i$  can be set for each training pattern.  $m_{ij}$  is the slope of the line through the neurons  $i$  and  $j$ , and is given by

$$m_{ij} = \frac{d_{ij}^y}{d_{ij}^x} \quad (d_{ij}^x \neq 0). \quad (7)$$

In Eq.(2), the neuron whose Euclidean distance between its connection weights and the learning vector is minimum in the neurons which can be take areas without overlaps to the areas corresponding to the patterns which are already trained. In Eq.(2), the size of the area for the learning vector are used as  $a_i$  and  $b_i$ .

- (5) If  $d(X^{(p)}, W_r) > \theta^t$  is satisfied, the connection weights of the neurons in the ellipse whose center is the neuron  $r$  are updated as follows:

$$W_i(t+1) = \begin{cases} W_i(t) + \alpha(t)(X^{(p)} - W_i(t)), & (d_{ri} \leq D_{ri}) \\ W_i(t), & (\text{otherwise}) \end{cases} \quad (8)$$

where  $\alpha(t)$  is the learning rate and is given by

$$\alpha(t) = \frac{-\alpha_0(t - T)}{T}. \quad (9)$$

$\alpha_0$  is the initial value of  $\alpha(t)$  and  $T$  is the upper limit of the learning iterations.

- (6) (5) is iterated until  $d(X^{(p)}, W_r) \leq \theta^t$  is satisfied.  
(7) The connection weights of the neuron  $r$   $W_r$  are fixed.  
(8) (2)~(7) are iterated when a new pattern set is given.

### 2.3 Recall process

In the recall process of the KFMPAM-WD, when the pattern  $X$  is given to the I/O-Layer, the output of the neuron  $i$  in the Map-Layer,  $x_i^{map}$  is calculated by

$$x_i^{map} = \begin{cases} 1, & (i = r) \\ 0, & (\text{otherwise}) \end{cases} \quad (10)$$

where  $r$  is selected randomly from the neurons which satisfy

$$\frac{1}{N^{in}} \sum_{k \in C} g(X_k - W_{ik}) > \theta^{map} \quad (11)$$

where  $\theta^{map}$  is the threshold of the neuron in the Map-Layer, and  $g(\cdot)$  is given by

$$g(b) = \begin{cases} 1, & (|b| < \theta^d) \\ 0, & (\text{otherwise}). \end{cases} \quad (12)$$

In the KFMPAM-WD, one of the neurons whose connection weights are similar to the input pattern are selected randomly as the winner neuron. So, the probabilistic association can be realized based on the weights distribution. For example, if the training patterns including the common term such as  $\{X, Y_1\}$ ,  $\{X, Y_2\}$  are memorized, and the number of the neurons whose connection weights are similar to the pattern pair  $\{X, Y_1\}$  is larger than the number of the neurons whose connection weights are similar to the pattern pair  $\{X, Y_2\}$ , then the probability that the pattern pair  $\{X, Y_1\}$  is recalled is higher than the probability that the pattern pair  $\{X, Y_2\}$  is recalled.

When the binary pattern  $X$  is given to the I/O-Layer, the output of the neuron  $k$  in the I/O-Layer  $x_k^{io}$  is given by

$$x_k^{io} = \begin{cases} 1, & (W_{rk} \geq \theta_b^{io}) \\ 0, & (\text{otherwise}) \end{cases} \quad (13)$$

where  $\theta_b^{io}$  is the threshold of the neurons in the I/O-Layer.

When the analog pattern  $X$  is given to the I/O-Layer, the output of the neuron  $k$  in the I/O-Layer  $x_k^{io}$  is given by

$$x_k^{io} = W_{rk}. \quad (14)$$

## 3. Reinforcement learning using Kohonen feature map probabilistic associative memory based on weights distribution

Here, we explain the proposed reinforcement learning method using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD)(Osana, 2009).

### 3.1 Outline

In the proposed method, the actor in the Actor-Critic(Witten, 1977) is realized by the KFMPAM-WD. In this research, the I/O-Layer in the KFMPAM-WD is divided into two parts corresponding to the state  $s$  and the action  $a$ , and the actions for the states are memorized.

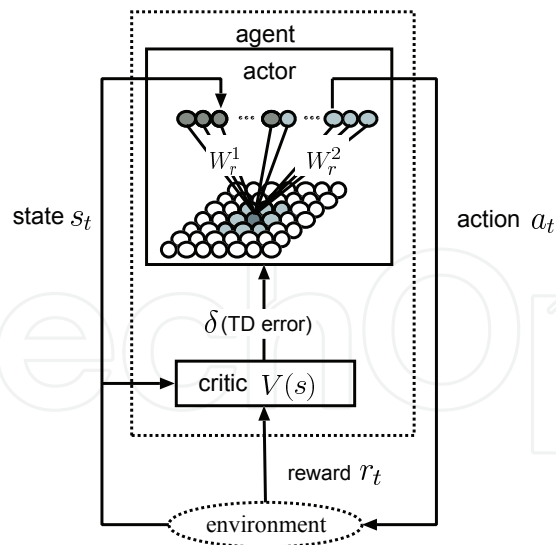


Fig. 2. Flow of Proposed Method.

In this method, the critic receives the states which are obtained from the environment, the state is estimated and the value function is updated. Moreover, the critic outputs Temporal Difference (TD) error to the actor. The KFMPAM-WD which behaves as the actor (we call this “actor network”) is trained based on the TD error, and selects the action from the state of environment. Figure 2 shows the flow of the proposed method.

3.2 Actor network

In the proposed method, the actor in the Actor-Critic(Witten, 1977) is realized by the KFMPAM-WD.

3.2.1 Dynamics

In the actor network, when the state  $s$  is given to the I/O-Layer, the corresponding action  $a$  is recalled. In the proposed method, the other action is also selected randomly (random selection), and the more desirable action from the recalled action and the action selected in the random selection is chosen as the action finally.

When the pattern  $X$  is given to the network, the output of the neuron  $i$  in the Map-Layer at the time  $t$   $x_i^{map}(t)$  is given by Eq.(10), and the output of the neuron  $k$  in the I/O-Layer at the time  $t$   $x_k^{io}(t)$  is given by Eq.(13) or Eq.(14). In the actor network, only the state information is given, so the input pattern is given by

$$X = (s(t), 0)^T$$

(15)

where  $s(t)$  is the state at the time  $t$ .

3.2.2 Learning

The actor network is trained based on the TD error from the critic. The learning vector at the time  $t$   $X^{(t)}$  is given by the state  $s(t)$  and the corresponding action  $a(t)$  as follows.

$$X^{(t)} = (s(t), a(t))^T$$

(16)

(1) When action is recalled by actor network

When the pair of the state and the selected action are memorized in the actor network, the area size corresponding to the pair is updated. If the TD error is larger than 0, the area is expanded. If the TD error is smaller than 0, the area is reduced.

**(1-1) When state and action are stored**

**(a) When TD error is larger than 0**

When the TD error is larger than 0, the area including the fired neuron whose center is the neuron  $z$  is expanded.

$$a_z^{(new)} \leftarrow \begin{cases} a_z^{(old)} + \Delta a^+, & (a_z^{(old)} + \Delta a^+ \leq a^{max}) \\ a_z^{(old)}, & (\text{otherwise}) \end{cases} \quad (17)$$

$$b_z^{(new)} \leftarrow \begin{cases} b_z^{(old)} + \Delta b^+, & (b_z^{(old)} + \Delta b^+ \leq b^{max}) \\ b_z^{(old)}, & (\text{otherwise}) \end{cases} \quad (18)$$

where  $\Delta a^+$ ,  $\Delta b^+$  are the increment of  $a_z$  and  $b_z$ , and  $a^{max}$ ,  $b^{max}$  are the maximum of  $a_z$  and  $b_z$ . The connection weights are updated as follows.

$$W_i(t+1) = \begin{cases} W_i(t) + \alpha(t)(X^{(tr)}(t) - W_i(t)), & (d_{zi} < D_{zi}) \\ W_i(t), & (\text{otherwise}) \end{cases} \quad (19)$$

where  $d_{zi}$  is the distance between the neuron  $i$  and the neuron  $z$ , and  $D_{zi}$  is the radius of the ellipse area whose center is the neuron  $z$  for the direction to the neuron  $i$ .

**(b) When TD error is smaller than 0**

When the TD error is smaller than 0, the area including the fired neuron whose center is the neuron  $z$  is reduced.

$$a_z^{(new)} \leftarrow \begin{cases} 0, & (a_z^{(new)} < 0 \text{ or } b_z^{(new)} < 0) \\ a_z^{(old)} - \Delta a^-, & (\text{otherwise}) \end{cases} \quad (20)$$

$$b_z^{(new)} \leftarrow \begin{cases} 0, & (a_z^{(new)} < 0 \text{ or } b_z^{(new)} < 0) \\ b_z^{(old)} - \Delta b^-, & (\text{otherwise}) \end{cases} \quad (21)$$

where  $\Delta a^-$ ,  $\Delta b^-$  are the decrement of  $a_z$  and  $b_z$ . If  $a_z^{(new)}$  or  $b_z^{(new)}$  becomes smaller than 0, the connection weights of neuron  $z$  are unlocked and  $a_z^{(new)}$  and  $b_z^{(new)}$  are set to 0. The connection weights are updated as follows.

$$W_i(t+1) = \begin{cases} R, & (D_{zi}^{after} < d_{zi} \leq D_{zi}^{before}) \\ W_i(t), & (\text{otherwise}) \end{cases} \quad (22)$$

where  $R$  is random value.  $D_{zi}^{before}$  is the radius of the ellipse area whose center is the neuron  $z$  for the direction to the neuron  $i$  before the area update, and  $D_{zi}^{after}$  is the radius of the ellipse area whose center is the neuron  $z$  for the direction to the neuron  $i$  after the area update.

**(1-2) When state and action are not stored**

When the fired neuron is not in the areas corresponding to the stored pairs of state and action and the TD error is larger than 0, the recalled pair of state and action is regarded as an unstored data and is memorized as a new pattern.



The connection weights are updated as follows.

$$W_i(t+1) = \begin{cases} W_i(t) + \alpha(t)(X^{(tr)}(t) - W_i(t)), & (d_{ri} \leq D_{ri}) \\ W_i(t), & (\text{otherwise}) \end{cases} \quad (23)$$

where  $r$  is the center neuron of the new area, and  $a^{ini}, b^{ini}$  are the initial radius of ellipse area.

## (2) When action is selected by random selection and TD error is larger than 0

When the pair of the state and the selected action are not memorized in the actor network and the TD error is larger than 0, the pair is trained as new pattern.

### 3.3 Reinforcement learning using KFMPAM-WD

The flow of the proposed reinforcement learning method using KFMPAM-WD is as follows:

- (1) The initial values of weights in the actor network are chosen randomly.
- (2) The agent observes the environment  $s(t)$ , and the actor  $a(t)$  is selected by the actor network or the random selection.
- (3) The state  $s(t)$  transits to the  $s(t+1)$  by action  $a(t)$ .
- (4) The critic receives the reward  $r(s(t+1))$  from the environment  $s(t+1)$ , and outputs the TD error  $\delta$  to the actor.

$$\delta = r(s(t+1)) + \gamma V(s(t+1)) - V(s(t)) \quad (24)$$

where  $\gamma$  ( $0 \leq \gamma \leq 1$ ) is the decay parameter, and  $V(s(t))$  is the value function for the state  $s(t)$ .

- (5) The eligibility  $e_t(s)$  is updated.

$$e(s) \leftarrow \begin{cases} \gamma \lambda e(s) & (\text{ifs} \neq s(t+1)) \\ \gamma \lambda e(s) + 1 & (\text{ifs} = s(t+1)) \end{cases} \quad (25)$$

where  $\gamma$  ( $0 \leq \gamma \leq 1$ ) is the decay parameter, and  $\lambda$  is the trace decay parameter.

- (6) All values for states  $V(s)$  are updated based on the eligibility  $e_t(s)$  ( $s \in S$ ).

$$V(s) \leftarrow V(s) + \xi \delta e_t(s) \quad (26)$$

where  $\xi$  ( $0 \leq \xi \leq 1$ ) is the learning rate.

- (7) The connection weights in the actor network are updated based on the TD error (See 3.2.2).
- (8) Back to (2).

## 4. Computer experiment results

Here, we show the computer experiment results to demonstrate the effectiveness of the proposed method.

### 4.1 Probabilistic association ability of KFMPAM-WD

Here, we examined the probabilistic association ability of the Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD) (Koike and Osana, 2010) which is used in the proposed method. The experiments were carried out in the KFMPAM-WD which has 800 neurons in the I/O-Layer and 400 neurons in the Map-Layer.



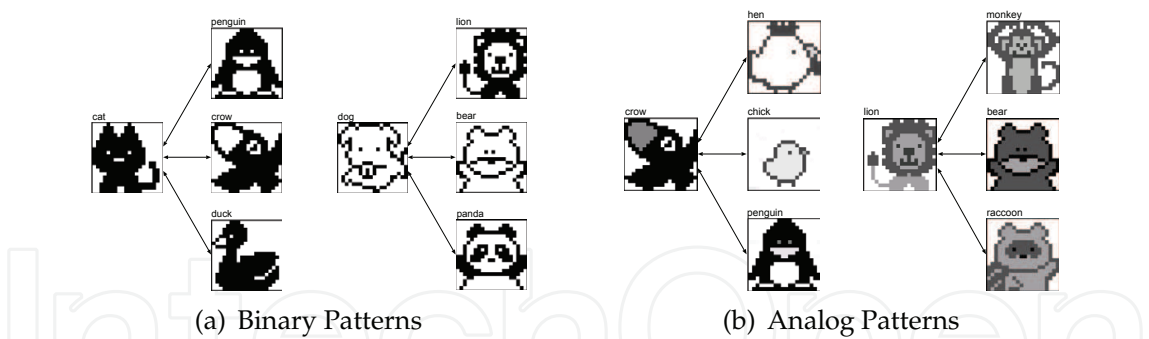


Fig. 3. Training Pattern Pairs.

Here, we show the association result of the KFMPAM-WD for binary and analog patterns. Figure 3 shows examples of stored pattern pairs. Figure 4 (a)~(c) show a part of the association result of the KFMPAM-WD when “cat” was given during  $t=1\sim500$ . As shown in this figure, the KFMPAM-WD could recall the corresponding patterns (“duck” ( $t=1$ ), “penguin” ( $t=3$ ), “crow” ( $t=4$ )). Figure 4 (d)~(f) show a part of the association result of the KFMPAM-WD when “dog” was given during  $t=501\sim1000$ . As shown in this figure, the proposed model could recall the corresponding patterns (“panda” ( $t=501$ ), “lion” ( $t=502$ ), “bear” ( $t=505$ )).

Figure 5 shows the same association result by the direction cosine between the output pattern and each stored pattern.

Figure 6 (a)~(c) show a part of the association result of the KFMPAM-WD when “crow” was given during  $t=1\sim500$ . As shown in this figure, the KFMPAM-WD could recall the corresponding patterns (“hen” ( $t=1$ ), “penguin” ( $t=2$ ), “chick” ( $t=3$ )). Figure 6 (d)~(f) show a part of the association result of the KFMPAM-WD when “lion” was given during  $t=501\sim1000$ . As shown in this figure, the proposed model could recall the corresponding patterns (“raccoon dog” ( $t=501$ ), “bear” ( $t=503$ ), “monkey” ( $t=504$ )).

Figure 7 shows the same association result by the direction cosine between the output pattern and each stored pattern.

Figure 8 shows an example of the area representation in the Map-Layer for the training set shown in Fig.3. In this figure, light blue or green areas show area representation for each training pattern, and the red neurons show the weight-fixed neurons.

Tables 1 and 2 show the relation between the area size and the number of recall time. As shown in these tables, the KFMPAM-WD can realize probabilistic association based on the area size (that is, weights distribution).

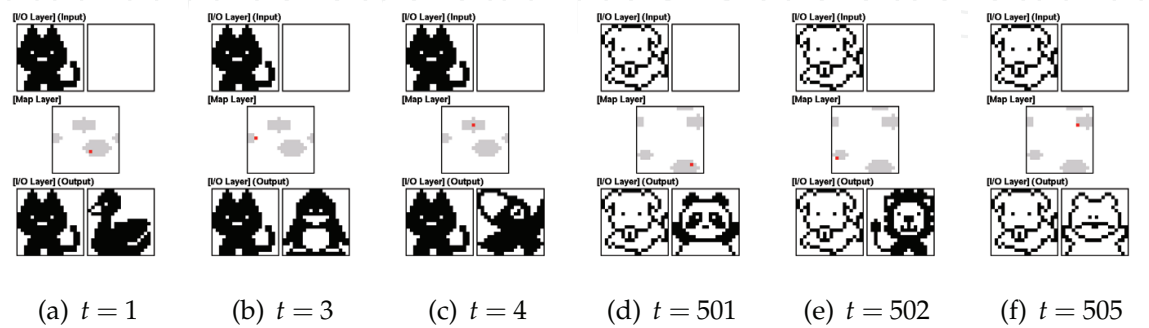


Fig. 4. Association Result (Binary Pattern).

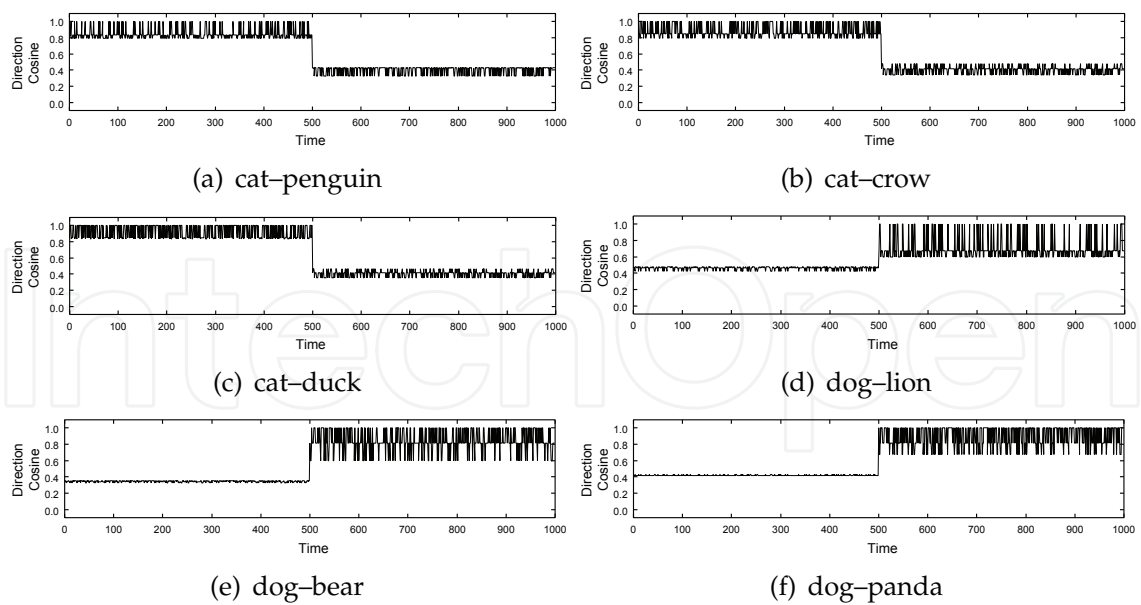


Fig. 5. Association Result (Direction Cosine).

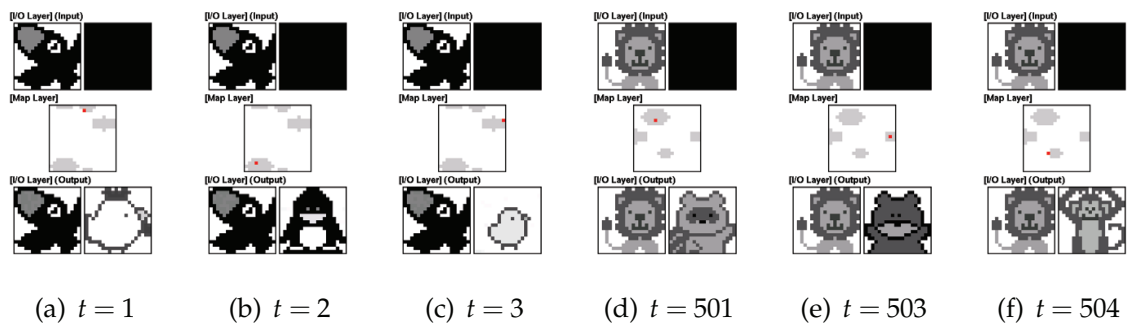


Fig. 6. Association Result (Analog Pattern).

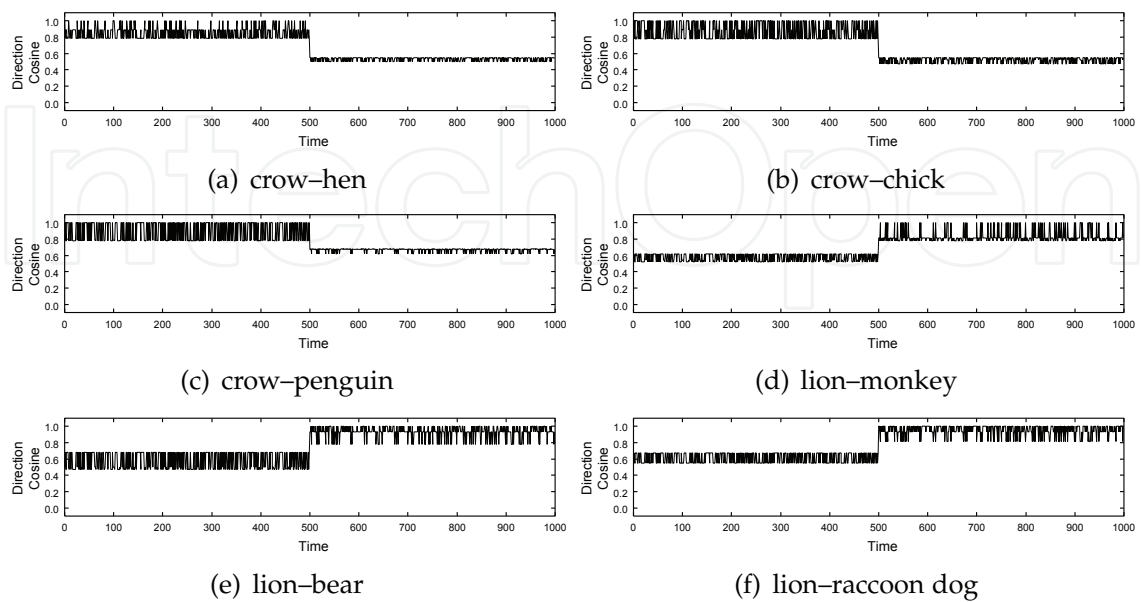


Fig. 7. Association Result (Direction Cosine).

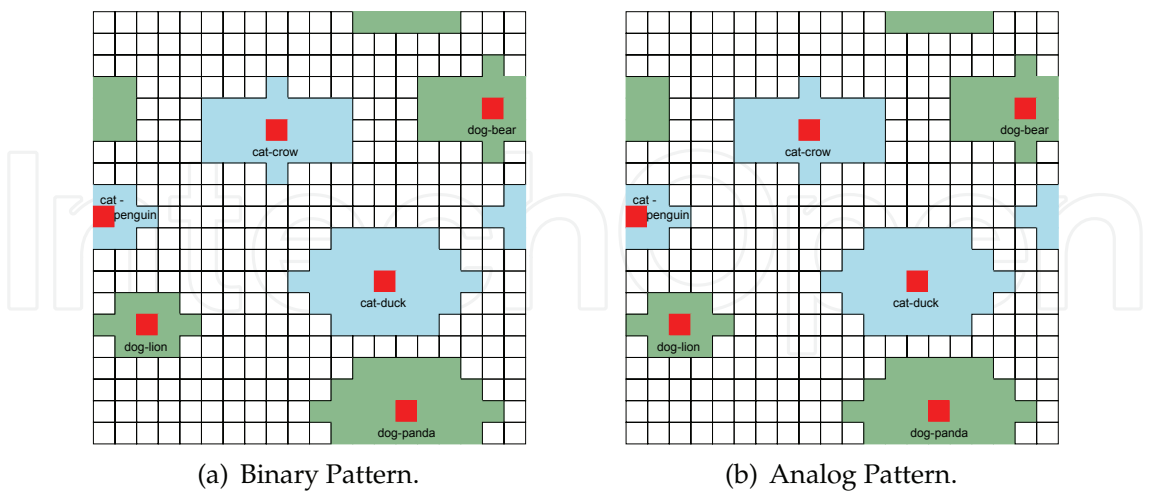


Fig. 8. Area Representation for Training Set in Fig.3.

Input Pattern	Output Pattern	Area Size	Recall Time
cat	penguin	11	85
	crow	23	157
	duck	33	258
dog	lion	11	80
	bear	23	166
	panda	33	254

Table 1. Relation between Area Size and Recall Time (Binary Pattern).

Input Pattern	Output Pattern	Area Size	Recall Time
crow	hen	11	67
	chick	23	179
	penguin	33	254
lion	monkey	11	82
	bear	23	161
	raccoon dog	33	257

Table 2. Relation between Area Size and Recall Time (Analog Pattern).

4.2 Path-finding problem

We applied the proposed method to the path-finding problem. In this experiment, a agent moves from the start point (S) to the goal point (G). The agent can observe the states of three cells in the lattice, and can move forward/left/right. As the positive reward, we gave 3 when the agent arrives at the goal and 2 when the agent moves. And as the negative reward, we gave  $-1$  when the agent hits against the wall. Table 3 shows experimental conditions. Figure 9 shows an example of maps (and the trained route (arrow)).

Parameters for Reinforcement Learning		
Decay Parameter	$\gamma$	0.7
Trace Decay Parameter	$\lambda$	0.33
Learning Rate	$\zeta$	0.33
Parameters for Actor Network (Learning)		
Random Value	$R$	$0.0 < R < 1.0$
Initial Long Radius	$a_z^{ini}$	2.5
Initial Short Radius	$b_z^{ini}$	1.5
Increment of Area Size	$\Delta a_z^+$	0.01
Decrement of Area Size	$\Delta a_z^-$	0.1
Lower Limit of Long Radius	$a_z^{max}$	4.0
Lower Limit of Short Radius	$b_z^{max}$	3.0
Lower Limit of Long Radius	$a_z^{min}$	0.0
Lower Limit of Short Radius	$b_z^{min}$	0.0
Weight Update Number	$T^{max}$	200
Threshold for Learning	$\theta^l$	$10^{-7}$
Parameters for Actor Network (Recall)		
Threshold of Neurons in Map-Layer	$\theta_b^{map}$	0.01
Threshold of Neurons in I/O-Layer	$\theta_b^{in}$	0.5

Table 3. Experimental Conditions.

4.2.1 Transition of number of steps

Figure 10 shows the transition of number of steps from the start to the goal. As shown in these figures, the agent can learn the route from the start to the goal by the proposed method.

4.2.2 Trained relation between state and action

Figure 11 shows an example of the trained relation between the state and the action. As shown these figures, the agent can learn the relation between state and action by the proposed method.

4.2.3 Variation of action selection method

Figure 12 shows the variation of the action selection method in the proposed method. As shown in these figures, at the beginning of the learning, the random selection is used frequently. After the learning, the action which is selected by the actor network is used frequently.

4.2.4 Use of learning information in similar environment

Here, we examined in the actor network that learns in the Map 2. Figure 13 (a) shows the transition of steps in the Map 3. As shown in this figure, the agent learn to reach the goal in few steps when the actor network that learns in the environment of the Map 2 in advance. Figure 13 (b) shows the variation of the action selection method in this experiment. Figure 14 shows the an example of the trained relation between the state and the action in this experiment.

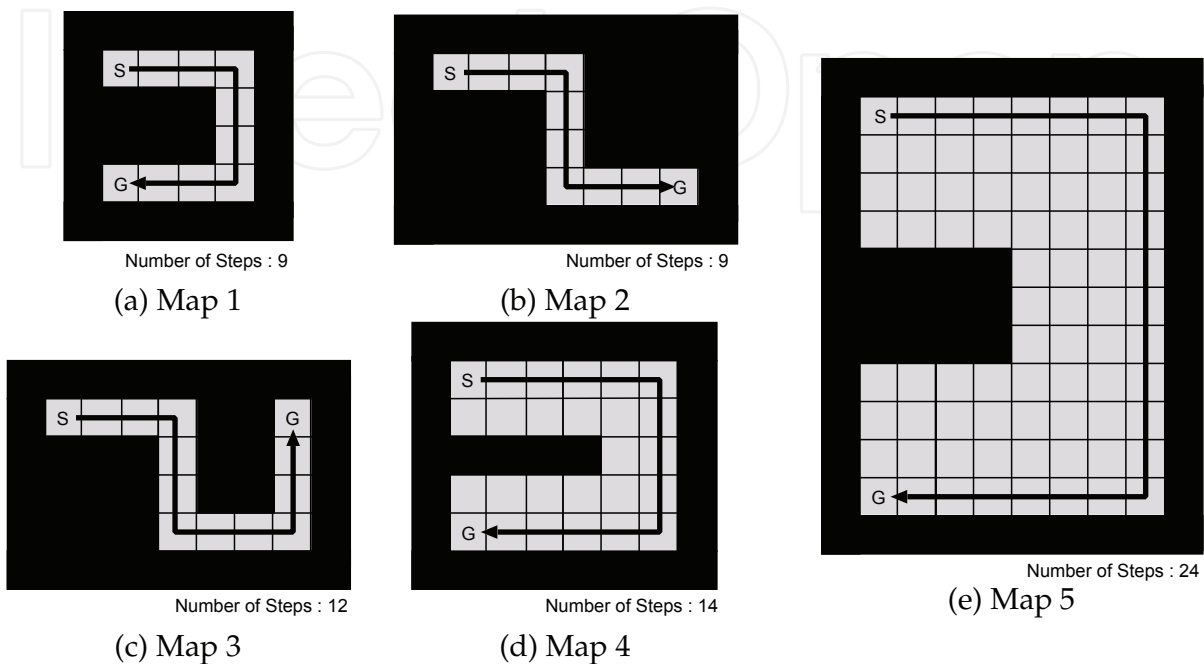


Fig. 9. Map and Trained Route.

5. Conclusion

In this research, we have proposed the reinforcement learning method using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution. The proposed method is based on the actor-critic method, and the actor is realized by the Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution. We carried out a series of computer experiments, and confirmed the effectiveness of the proposed method in path-finding problem.

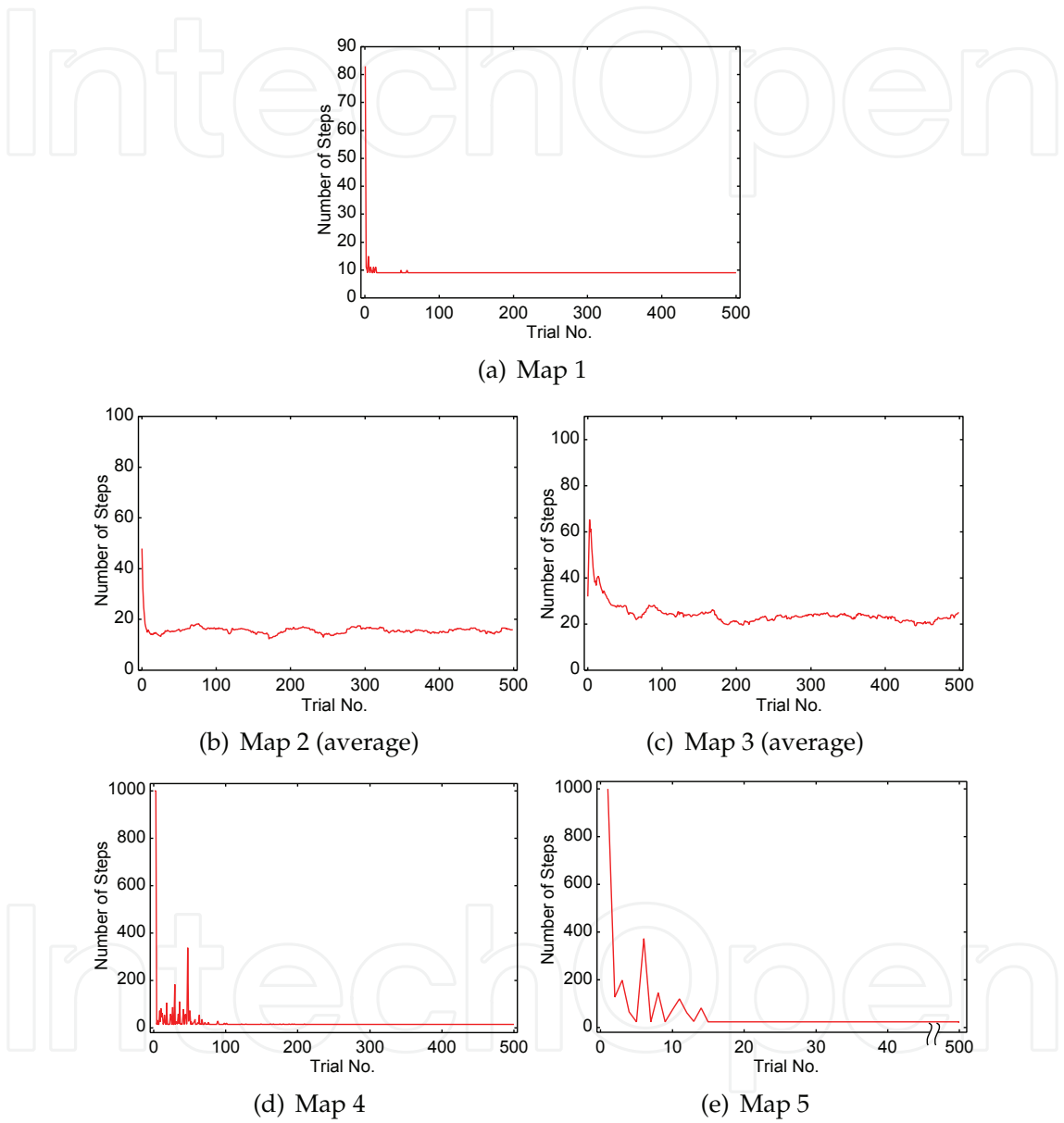


Fig. 10. Transition of Steps.

forward	forward	left	forward	right
35 $a_i : 4.00$ $b_i : 3.00$	11 $a_i : 2.56$ $b_i : 1.56$	3 $a_i : 1.90$ $b_i : 0.90$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$

(a) Map 1

right	forward	forward	forward	left	right
11 $a_i : 2.51$ $b_i : 1.51$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	11 $a_i : 2.23$ $b_i : 1.23$	35 $a_i : 4.00$ $b_i : 3.00$

(b) Map 2

forward	forward	left	right	left	forward
35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	11 $a_i : 2.51$ $b_i : 1.51$	3 $a_i : 1.04$ $b_i : 0.04$	11 $a_i : 2.53$ $b_i : 1.53$	35 $a_i : 4.00$ $b_i : 3.00$

(c) Map 3

left	forward	right
35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$

(d) Map 4

left	right	forward	forward
35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$

(e) Map 5

Fig. 11. An example of Trained Relation between State and Action.



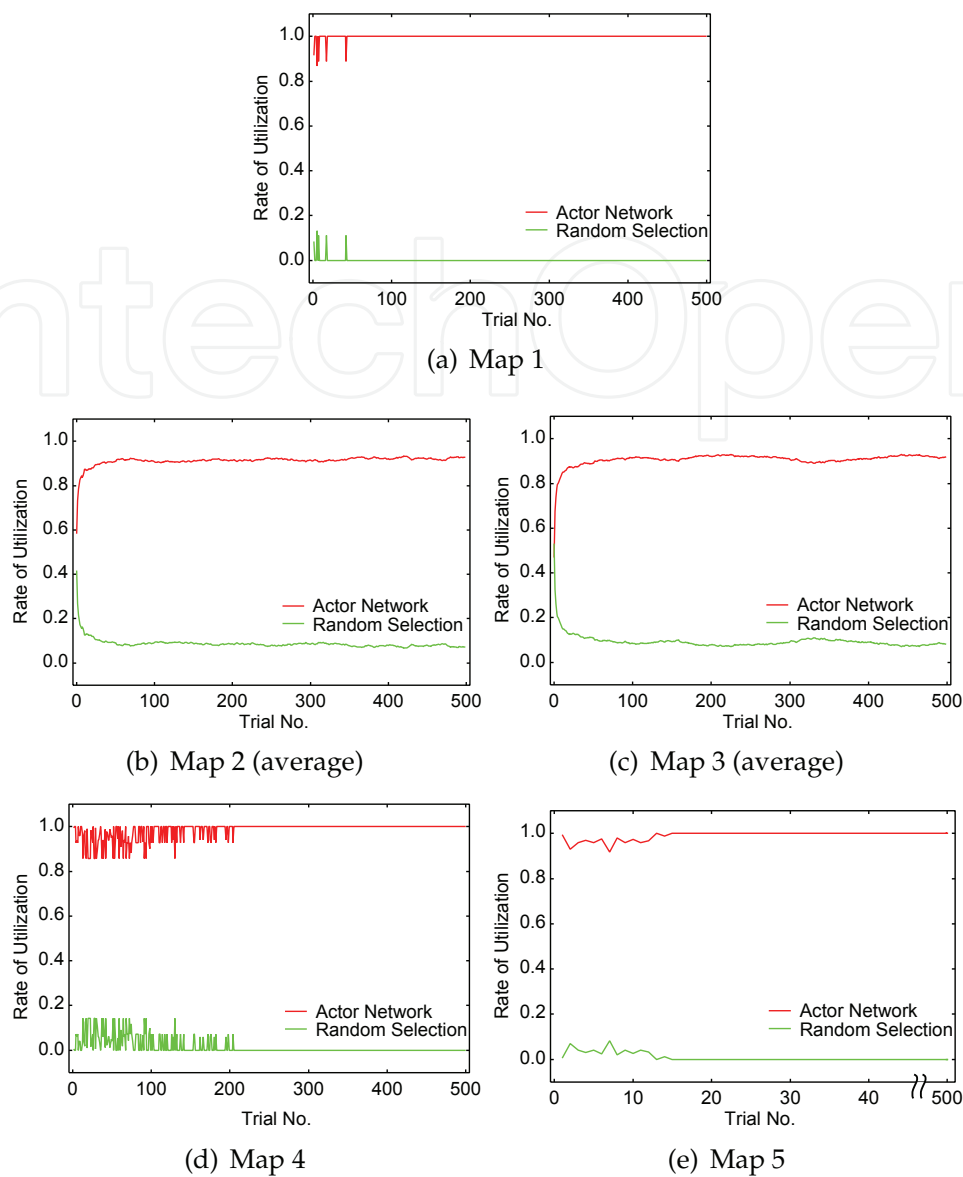


Fig. 12. Variation of Action Selection Method.

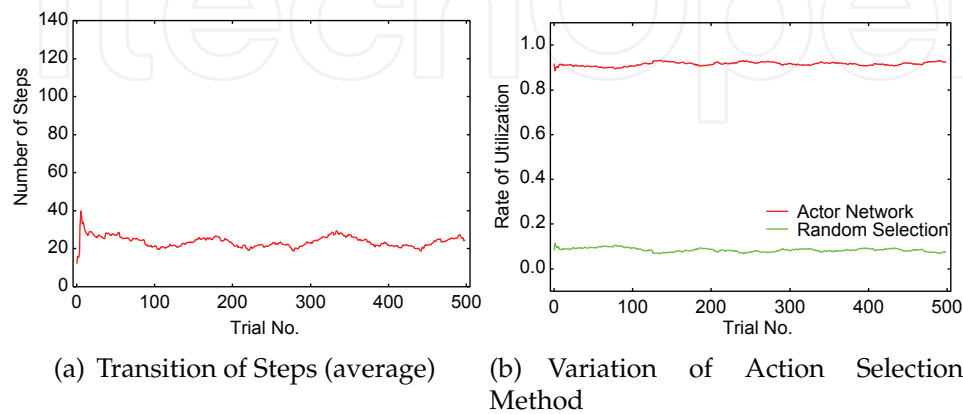


Fig. 13. Use of Learning Information in Similar Environment.







					
right	forward	left	forward	forward	right
11 $a_i : 2.51$ $b_i : 1.51$	35 $a_i : 4.00$ $b_i : 3.00$	7 $a_i : 2.00$ $b_i : 1.00$	35 $a_i : 4.00$ $b_i : 3.00$	35 $a_i : 4.00$ $b_i : 3.00$	3 $a_i : 1.04$ $b_i : 0.04$

Fig. 14. An example of Trained Relation between State and Action (Map 2 → Map 3).

6. References

Ishii, S., Shidara, M. & Shibata, K. (2005) "A model of emergence of reward expectancy neurons by reinforcement learning," *Proceedings of the 10th International Symposium on Artificial Life and Robotics*, GS21–5.

Kohonen, T. (1994) *Self-Organizing Maps*, Springer.

Koike, M. & Osana, Y. (2010) "Kohonen feature map probabilistic associative memory based on weights distribution," *Proceedings of IASTED Artificial Intelligence and Applications*, Innsbruck.

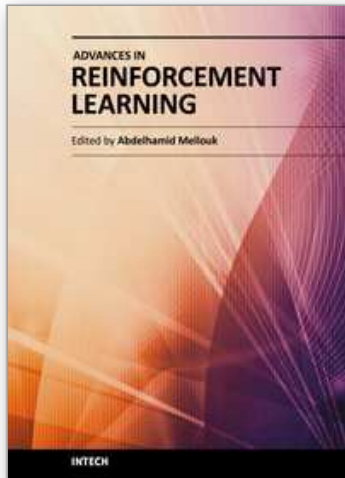
Osana, Y. (2009) "Reinforcement learning using Kohonen feature map probabilistic associative Memory based on weights distribution," *Proceedings of International Symposium on Nonlinear Theory and its Applications*, Sapporo.

Shibata, K., Sugisaka, M. & Ito, K. (2001) "Fast and stable learning in direct-vision-based reinforcement learning," *Proceedings of the 6th International Symposium on Artificial Life and Robotics*, Vol.1, pp.200–203.

Shimizu, A. & Osana, Y. (2008) "Reinforcement learning using Kohonen feature map assocaitive memory with refractoriness based on area representation," *Proceedings of International Conference on Neural Information Processing*, Auckland.

Sutton, R. S. & Barto A. G. (1998). *Reinforcement Learning, An Introduction*, The MIT Press.

Witten, I. H. (1977). "An adaptive optimal controller for discrete-time Markov environments," *Information and Control*, Vol.34, pp. 286–295.



## **Advances in Reinforcement Learning**

Edited by Prof. Abdelhamid Mellouk

ISBN 978-953-307-369-9

Hard cover, 470 pages

**Publisher** InTech

**Published online** 14, January, 2011

**Published in print edition** January, 2011

Reinforcement Learning (RL) is a very dynamic area in terms of theory and application. This book brings together many different aspects of the current research on several fields associated to RL which has been growing rapidly, producing a wide variety of learning algorithms for different applications. Based on 24 Chapters, it covers a very broad variety of topics in RL and their application in autonomous systems. A set of chapters in this book provide a general overview of RL while other chapters focus mostly on the applications of RL paradigms: Game Theory, Multi-Agent Theory, Robotic, Networking Technologies, Vehicular Navigation, Medicine and Industrial Logistic.

### **How to reference**

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Yuko Osana (2011). Reinforcement Learning Using Kohonen Feature Map Probabilistic Associative Memory Based on Weights Distribution, *Advances in Reinforcement Learning*, Prof. Abdelhamid Mellouk (Ed.), ISBN: 978-953-307-369-9, InTech, Available from: <http://www.intechopen.com/books/advances-in-reinforcement-learning/reinforcement-learning-using-kohonen-feature-map-probabilistic-associative-memory-based-on-weights-d>

**INTech**  
open science | open minds

### **InTech Europe**

University Campus STeP Ri  
Slavka Krautzeka 83/A  
51000 Rijeka, Croatia  
Phone: +385 (51) 770 447  
Fax: +385 (51) 686 166  
[www.intechopen.com](http://www.intechopen.com)

### **InTech China**

Unit 405, Office Block, Hotel Equatorial Shanghai  
No.65, Yan An Road (West), Shanghai, 200040, China  
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元  
Phone: +86-21-62489820  
Fax: +86-21-62489821

© 2011 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the [Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License](https://creativecommons.org/licenses/by-nc-sa/3.0/), which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen