Solving Continuous Action/State Problem in Q-Learning Using Extended Rule Based Fuzzy Inference Systems

Min-Soeng Kim and Ju-Jang Lee

Abstract: Q-learning is a kind of reinforcement learning where the agent solves the given task based on rewards received from the environment. Most research done in the field of Q-learning has focused on discrete domains, although the environment with which the agent must interact is generally continuous. Thus we need to devise some methods that enable Q-learning to be applicable to the continuous problem domain.

In this paper, an extended fuzzy rule is proposed so that it can incorporate Q-learning. The interpolation technique, which is widely used in memory-based learning, is adopted to represent the appropriate Q value for current state and action pair in each extended fuzzy rule. The resulting structure based on the fuzzy inference system has the capability of solving the continuous state and action problem in Q-learning. Also it can generate fuzzy rules via interacting with the environment without a priori knowledge about the environment. The effectiveness of the proposed structure is shown through simulation on the cart-pole system.

Keywords: Q-Learning, reinforcement learning, fuzzy logic controller

I. Introduction

Q-learning is a kind of reinforcement learning where the agent solves the given task based on rewards received from the environment. In Q-learning[1], Q value which can be seen as a quality value for a certain action in a certain state is generated and an agent can learn to select its proper action in each state based on these Q values. However, most research done in the field of reinforcement learning has focused on discrete domains, although the environment with which the agent must interact is usually continuous. Thus we need to devise some methods that enable Q-learning to be applicable to the continuous problem domain. Moreover, from the control perspective, it is natural that the agent with continuous action can perform better than the one with limited discrete actions. There has been much research to make Q-learning deal with the continuous state/action spaces.[2]-[4] In these methods, however, the action selection method is based on a step search in action space. That is, the Q value for all actions in the possible action range is calculated and the action of which the Q value is the maximum is selected to be performed. The continuous property of action is dependent on the step size. Moreover, the sequence of generated actions does not change continuously, or smoothly. The continuity of action is important from the practical view because the control input cannot change rapidly by an actuator. Meanwhile, in the fuzzy inference system (FIS), both the continuous state and continuous action can be described by a finite set of fuzzy variables. Thus, the FIS is promising as an effective model to overcome the limitations of conventional Q-learning.[5]-[7]

In this paper, the FIS and Q-learning are combined to resolve the continuous state and action problem in Q-learning. A basic fuzzy rule is extended so that Q-learning can be incorporated and the interpolation technique, which is widely used in memory-based learning, is adopted to represent the appropriate Q value for current state and current action.

The remainder of this paper is organized as follows. In Section 2, the basic fuzzy rule and the conventional Q-learning algorithm are briefly reviewed. In Section 3, the concept of extended rule is addressed. Also Q-value representation scheme using kernel function based interpolation is developed, which enable the introduction of Q-learning in FIS. The proposed structure and the learning algorithm for this structure are shown in Section 4. In Section 5, the simulation results are presented to show the effectiveness of the proposed methods. The conclusion about the characteristics of proposed methods is presented in Section 6.

Manuscript received: Nov. 24, 2000., Accepted: May. 25, 2001.

Min-Soeng Kim: Department of Electrical Engineering and Computer Science, KAIST (mibella@cais.kaist.ac.kr)

Ju-Jang Lee: Department of Computer Science and Electrical Engineering, KAIST (jjlee@ee.kaist.ac.kr)

II. Preliminaries

1. Fuzzy inference system

Fuzzy Inference Systems (FIS) are expert systems based on if-then rules, in which premises and conclusions are expressed by means of linguistic terms. The most important features of FIS are that they can incorporate experts' prior knowledge into their parameter and that these parameters have clear physical meaning. The FIS rule base is made of N different rules of the following general form:

$$R_i : \text{if } s_1 \text{ is } A_1^i \text{ and } s_2 \text{ is } A_2^i \cdots \text{ and } s_k \text{ is } A_M^i, \text{ then } y^i \text{ is } B^i \tag{1}$$

where $s = [s_1\, s_2 \cdots s_k] \in R^k$ is input state variables, A_m^i (where $m = 1, \cdots, M$) and B^i are fuzzy memberships for each i-th rule and characterized by linguistic label (e.g., small, medium, etc). These memberships are represented by a function $\mu_A : s \rightarrow [0,1]$ and $\mu_B : s \rightarrow [0,1]$, respectively.

A FIS are based on the fuzzy rule base that consists of several fuzzy rules. A FIS usually has a fuzzifier that translates real-valued input into fuzzy values and a defuzzifier that translates fuzzy output values into real-valued output. Thus continuous states or actions can be described by a finite set of fuzzy variables. Thus, a FIS seems to be an effective model to overcome the limitations of conventional Q-learning, that is, continuous state/action problem. By combining a FIS and Q-learning, we can make Q-learning able to deal with continuous states while generating continuous actions.

2. Q-learning

In Q-learning, the agent selects an appropriate action for the current state using the corresponding Q value of the current state and action. The Q value is the immediate reward by executing action from state plus the value of sum of expected future reward following the optimal policy thereafter. The basic Q-learning algorithm is as follows:

- Initialize all $Q(s,a)$ values to 0 and observe the current state s
- Do forever
  - Select an action a and execute it
  - Receive immediate reward r
  - Observe new state s'
  - Update $Q(s,a)$ value using the following equation

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{b \in A} Q(s',b) - Q(s,a)] \qquad (2)$$

  - Replace s by s'

A detailed explanation can be found in [8].

III. Extended rule

1. Extended rule

To incorporate Q-learning in FIS, an extended rule is proposed in this paper. The proposed extended rule is different from the basic fuzzy rule in that it has many candidates that can be selected to be an actual consequent part of the rule and has several characteristics as follows:

- For one antecedent part of each rule, several candidate consequent parts(actions) exist.
- Each candidate action has its own Q value.
- A rule action is selected among candidate actions by using policy and each candidate's Q-value.

This rule action becomes an actual consequent part of the rule and final action is obtained by defuzzification.

This extended rule can be described as follows:

$$\begin{aligned} R_i : &\text{if } s_1 \text{ is } L_1^i \text{ and } s_2 \text{ is } L_2^i \cdots \\ &\text{and } s_k \text{ is } L_M^i, \text{ then } a^i \text{ is } \pi(\mathrm{u}_\mathrm{j}^\mathrm{i}, Q_j^i) \end{aligned} \qquad (3)$$

where $s = [s_1\, s_2 \cdots s_k] \in R^k$ is input state variables, L_m^i (where $m = 1, \cdots, M$) is fuzzy memberships related to each input state. a^i is a i-th rule action which corresponds to the actual consequent part of i-th rule, R_i. $\pi(\cdot,\cdot)$ is a policy that selects a rule action from discrete action set A_i. A_i is a collection of an action, u_j^i (where $j = 1, \cdots, p$), which is a candidate action for the consequent part of i-th rule. Every u_j^i has its own Q value, Q_j^i. The superscript i represents rule number and $i = 1, \cdots, N$.

In each rule, one of the candidate actions is selected based on their Q value and the policy. Therefore the Q-learning process must be incorporated. A Rule's candidate action is currently selected action, among candidate consequent parts, to be an actual consequent part. These selected rule actions generate final action, a_f, by defuzzification. The final action is, however, generally different from each rule's rule action. Because Q value is the functional value of state and action, different value of action cannot be used in obtaining Q-value for current state and the final action. That is, we must have Q-value for the final action in each rule to obtain $Q(s,a_f)$ by defuzzification. Thus some kinds of technique to obtain the Q value corresponding to final action in each rule have to be devised, the interpolation technique is used to obtain a Q value for the final action in each rule in this paper.

2. Q value representation in the extended rule

Once the final action, a_f, is determined from the fuzzy rule base, the Q(s,a) value for a certain action a, which is not included in the discrete action set, can be calculated using interpolation technique. In other words, Q value for the final action in R_i can be obtained using the following equation:

$$Q(R_i, a_f) = \sum_{j=1}^{p} \frac{1}{\sum_{h=1}^{p} K(u_h^i, a_f)} K(u_j^i, a_f) Q_j^i \qquad (4)$$

where $K(\cdot,\cdot)$ is a kernel function which determines the degree of how each (u_j^i, Q_j^i) pair in R_i contributes to calculating $Q(R_i, a_f)$ and has the following form:

$$K(u_j^i, a_f) = \exp\left(\frac{-(a_f - u_j^i)^2}{\sigma^2}\right) \qquad (5)$$

IV. Self-organizing fuzzy inference system

1. The structure of SOFIS-Q

In this section, the self-organizing fuzzy inference system based on Q-learning(SOFIS-Q) is introduced. The term, self-organizing, means that the consequent part of the fuzzy rule is automatically selected from the discrete action set based on the policy and on the Q value for each candidate action. The whole structure is shown in Fig. 1. This structure performs the fuzzy inference based on the fuzzy rule base that consists of extended rules in (3). The inputs for the network are the states of the environment or a plant. Outputs are the final action and the Q value for the final action and the current state (i.e., $Q(s,a_f)$). The SOFIS-Q network consists of total 5 layers. Layer 1 to layer 3 achieves fuzzification in the input state that resolves continuous state representation problem in Q-learning Layer 4 to layer 5 generates continuous action by fuzzy inference and also generates the associated Q value for that inferred action by fuzzy inference mixed with the interpolation technique.

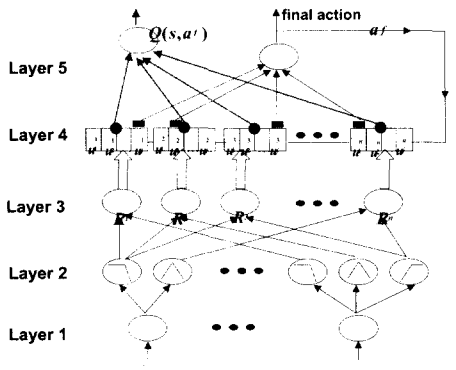


Fig. 1. The structure of SOFIS-Q.

• Layer 1

The function of the first layer is to receive the value of the state and transmit this value to the next layer. Each node in this layer corresponds to one input state variable $s_i(i=1,\cdots,k)$

• Layer 2

This layer consists of several linguistic labels for each input variable. Linguistic label L_m^i in (3) has its own membership function that is denoted by $\mu_{L_m}^i(\cdot)$. Triangular and trapezoidal membership functions are used in the paper. The output value from layer 2 indicates a membership degree of the current state.

• Layer 3

One link from a node in layer 1 to the node in layer 3 corresponds to one antecedent part of FIS. That is, it represents the '**if**' part of each rule R_i in (3). The value of each node in layer 3 simply represents how much current state s belongs to the rule R_i and can be obtained by *fuzzy and* operator. This value represents the *firing strength* of R_i for the state s and denoted as α_{R_i}. α_{R_i} is obtained using the following equation.

$$\alpha_{R_i}(s)=\prod_{m=1}^{M}\mu_{L_m}^i(s) \tag{6}$$

where the *fuzzy and* operator is implemented by using product operation.

• Layer 4

Layer 4 corresponds to the '**then**' part of each fuzzy rule. Based on Q values and policy, one candidate action from the discrete action set is selected as a rule action a^i in (3) and these rule actions from each rule are combined to generate the final action.

$$a^i=\pi(u_j^i,Q_j^i) \tag{7}$$

where j is the number of candidate actions and $\pi(\cdot,\cdot)$ represents the policy used to select the action. In this paper, ε-greedy policy [8] is used. This policy selects an action of which Q value is the maximum with the probability of $(1-\varepsilon)$. Besides the selection of the rule action, the Q value for the final action is calculated by (4) after the final action is calculated in layer 5.

• Layer 5

In layer 5, the final action is obtained by using a weighted sum of each rule action with their firing strength as follows.

$$a_f=\sum_{i=1}^{N}\frac{1}{\sum_{h=1}^{N}\alpha_{R_h}}\alpha_{R_i}a^i \tag{8}$$

Once a_f is obtained, value of $Q(R_i,a_f)$ is calculated for each rule using (4). Then, $Q(s,a_f)$ value can be obtained using the following equation.

$$Q(s,a_f)=\sum_{i=1}^{N}\frac{1}{\sum_{h=1}^{N}\alpha_{R_h}}\alpha_{R_i}Q(R_i,a_f) \tag{9}$$

2. Learning algorithm for SOFIS-Q

The update of Q-value for each rule in SOFIS-Q is performed through the change in the Q value of each candidate action for each rule. To speed up the learning, a replacing eligibility [9] is adopted. The eligibility is obtained using the following equation.

$$e_{ij}=\frac{\alpha_{R_i}}{\sum_{k=1}^{N}\alpha_{R_k}}\cdot\frac{K(u_j^i,a_f)}{\sum_{h=1}^{p}K(u_h^i,a_f)} \quad \text{if } R_i/u_j^i \text{ pair is selected.}$$
$$e_{ij}=\lambda\gamma e_{ij} \quad \text{otherwise} \tag{10}$$

where λ is an *eligibility rate* which is used to weight each (rule,rule action)-pair according to their proximity to the occurring time step from the current state.

The learning procedure is shown below:

1. Initialize all Q values of all candidate actions in each rule to zero
2. Perceive the current state s_{curr} and generate final action $a_{f_{curr}}$ by (8)
3. Calculate $Q(s_{curr},a_{f_{curr}})$ by (9) and calculate eligibility traces by (10)
4. Perform $a_{f_{curr}}$, receive reward r_{next} and transition to the next state s_{next}.
5. Obtain $Q(s_{next},a_{f_{next}})$ by (9) and calculate temporal error ε.

$$\varepsilon=r_{next}+\gamma Q(s_{next},a_{f_{next}})-Q(s_{curr},a_{f_{curr}})$$

6. For each candidate action of every rule, update the Q value as

$$Q_{j_{curr}}^i=Q_{j_{curr}}^i+\alpha\varepsilon e_{ij}$$

7. If trial is not ended, set $a_{f_{curr}}=a_{f_{next}}$ and goto step 2.

V. Simulation

1. Cart-pole balancing problem

The cart-pole balancing problem is concerned with how to balance an upright pole. The pole has only one degree of freedom and the primary control task is to keep the pole vertically balanced while keeping the cart within the rail track boundaries. Four state variables are used in describing this system and one variable represents the force applied to the cart. These are:

θ : the angle of the pole from upright position(radian);
$\dot{\theta}$: the angular velocity of the pole(radian/seconds);
x : the position of the cart's center (meters);
$\dot{x}$: the velocity of the cart (meter/seconds);
f : force applied to the cart (Newton)

The model dynamics and corresponding parameters can be found in [10]. Euler method is used for simulation using a time step of 0.02s. The constraints on the variables are

$$|\theta|\le 12^\circ, |x|\le 2.4m, \text{ and } |f|\le 10N.$$

It is assumed that the dynamics of the system is unknown to the controller. The controller can be informed of only the values of the state variables and the reward signal at each time step. The only available feedback for SOFIS-Q is failure signal when $|\theta|>12^\circ$ or $|x|>2.4m$. The reward signal given to

the controller is as follows:

$$r = \begin{cases} -1 & |\theta| > 12^\circ \text{ or } |x| > 2.4m \\ 0 & \text{otherwise} \end{cases}$$

The simulations were done with various numbers of candidate actions. The parameter values used for Q-learning are as follows: γ is 0.9, α is 0.3 and λ is 0.7.

2. Simulation results

Figure 2 and Fig. 3 show learning curves for the case where two candidate actions and five candidate actions, with zero initial condition, are used, respectively. The curve consists of ten consecutive runs where a run consists of several trials. Each trial ends in two cases. First, if the pole has fallen or the cart position exceeds the limits of the rail track. Second, if the time step exceeds 10000. 10000-time step corresponds to 200 seconds in simulation time.

Finally, from Fig. 4 to Fig. 6, shown are the resulting trajectories of θ, x and f for learned controller with 5 discrete candidate actions. These trajectories are selected as a sample from many results. The proposed SOFIS-Q can learn to achieve the given task only with the binary reward signal.

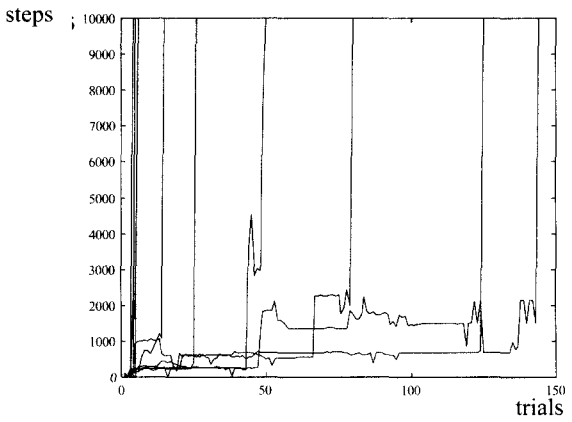


Fig. 2. Learning curve with 2 candidate actions.

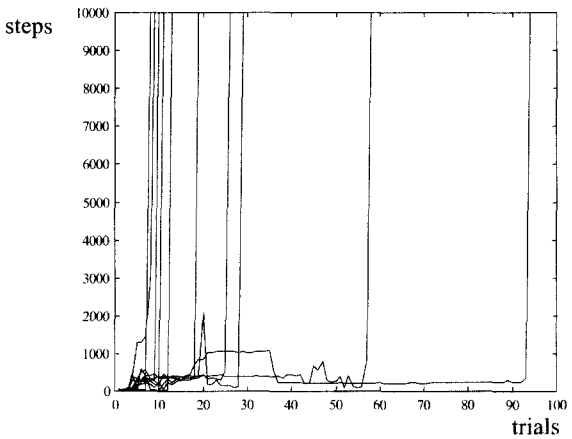


Fig. 3. Learning curve with 5 candidate actions.

Table 1. Average learning speed.

Method	Average trial
Conventional Q-learning	1530
Fuzzy interpolation based Q-learning	926
Self-organizing modular neural network	500
The proposed method with 2 candidate actions	47
The proposed method with 5 candidate actions	29

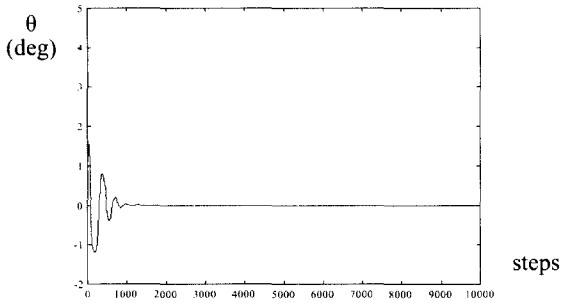


Fig. 4. Trajectory of pole angle.

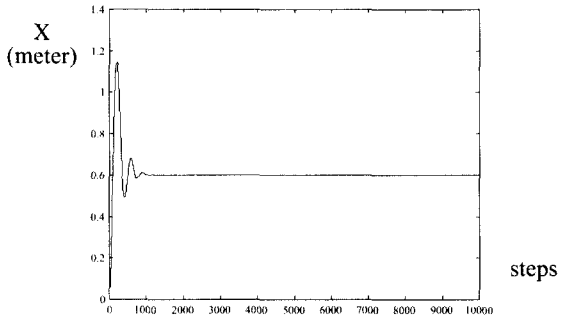


Fig. 5. Trajectory of cart position.

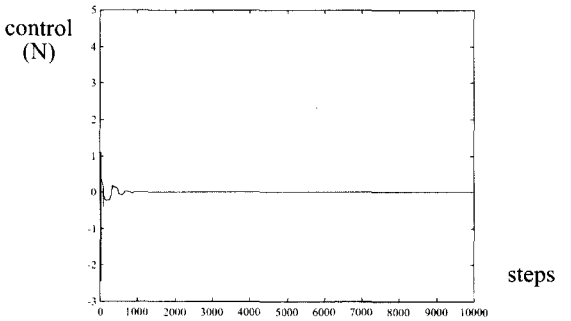


Fig. 6. Generated control force.

3. Comparison study

There were few approaches that can resolve continuous state and action problem simultaneously with Q-learning. Since the focus of this paper is on whether the agent can generate continuous action or not. Thus, we compare the results in the view of exerted control effort. Fig. 8 shows the results from the obtained data by the one-step search based approach, in which the control is selected by increasing control value with a finite step size and comparing the resulting Q value for each control value. Although the approaches based on the one-

step search might be different according to each method, it can be said that, because selection of control action is based upon the one-step search, the resulting control effort has similar characteristics for the most algorithm. Thus we here adopt the one-step search algorithm which uses the fuzzy network as an approximator of Q(s,a) values and used the same learning parameters and input space partition. From the Figures, we can directly see the advantage of ability that can generate continuous action. The control trajectory of Fig. 8 is very different from that of Fig. 7. For the one-step search method, we can make a smoother control by reducing the step size used in searching. This reducing in step size, however, means more computation time. Thus we must trade off between the continuity and the computation time with one-step search based algorithm. More importantly, the resulting control by the one-step search method is not truly *continuous* in that its value can jump from one action value to another value discontinuously. However, in SOFIS-Q, the generated control is inherently continuous.

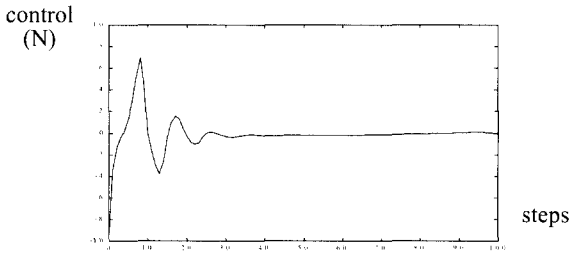


Fig. 7. Generated control force with the proposed method.

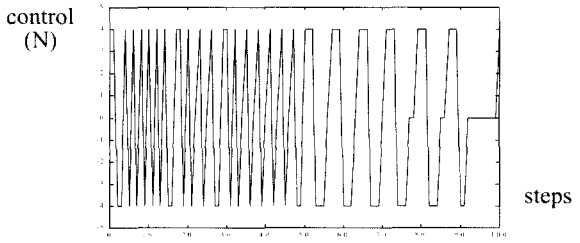


Fig. 8. Generated control force with one-step search method.

VI. Conclusion

In this paper, the self-organizing fuzzy inference system by Q-learning is proposed. The purpose of this SOFIS-Q is to cope with continuous states while generating continuous actions in Q-learning. By methods based on fuzzification, the continuous input state problem is easily resolved. By Using defuzzification, the continuous action problem is also resolved. By the extended rule and the interpolation technique, the SOFIS-Q behaves as a fuzzy inference system while approximating the Q(s,a) values. Several simulations for a pole-balancing problem were conducted to show the effectiveness of the proposed structure. To show the advantage of the continuous action, the comparison with the one step search based method was shown and the learning speed was compared. By incorporating the fuzzy inference system with Q-learning, without the model and knowledge about the plant, the SOFIS-Q can adjust descendant part of a FIS automatically.

In SOFIS-Q, there are several candidate actions in each rule, but what number is the optimal one for the problem can not be determined. There should be some pruning or growing methods that automatically increase or decrease the number of candidate actions. Similarly where to locate the candidate action in action space can be a problem according to the given task. Thus research on self-locating the candidate action should be made.

References

[1] C. J. C. H. Watkins and P. Dayan, "Technical note: Q-learning," *Machine Learning*, vol. 8, pp. 279-292, 1992.

[2] T. Horiuchi, A. Fujino, O. Katai, and T. Sawaragi, "Fuzzy interpolation based Q-learning with continuous states and action", *Proc. FUZZ-IEEE'96, 5th Int. Conf. Fuzzy Systems.*, New Orleans, pp. 594-600, 1996.

[3] J. H. Kim, I. H. Suh, S. R. Oh, Y. J. Cho, and Y. K. Chung, "Region-based Q-learning using convex clustering approach", *Proc. IROS 97*, pp. 601-607, 1997.

[4] J. C. Santamaria, R. S. Sutton, and A. Ram, "Experiments with reinforcement learning in problems with continuous state and action spaces," *CONIS Technical Report 96-088*, University of Massachusetts, December, 1996.

[5] A. Bonarini, "Delayed reinforcement, fuzzy {Q}-learning and fuzzy logic controllers," *Proc. Herrera* , pp. 447-466, 1996.

[6] P. Y. Glorennec, "Fuzzy Q-learning and dynamical fuzzy Q-learning," *Proc. FUZZ-IEEE'94, 3rd Int. Conf. Fuzzy Systems.*, Orlando, FL, vol. 1, pp. 474-479, 1994.

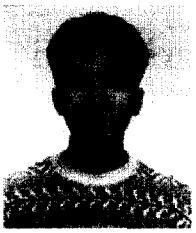
[7] K.B. Sim, "Fuzzy inference-based reinforcement learning of dynamic recurrent neural networks," *SICE Annual Conference of Japan(International session)*, Tokushima, Japan, pp. 1083-1088, 1997.

[8] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

[9] S. P. Singh, R. S. Sutton, "Reinforcement learning with replacing eligibility traces," *Machine Learning*, vol. 22, pp. 123-158, 1996.

[10] C. T. Lin and C. S. G. Lee, "Reinforcement structure / parameter learning for neural-network-based fuzzy logic control systems," *IEEE Transactions on Fuzzy Systems*, vol. 2, no. 1, pp. 46-63, February, 1994.

[11] S. G. Hong, M. S. Kim, W. Kim, and J. J. Lee, "Self-organizing modular neural network for reinforcement learning problem," *Proceedings of the ICMT' 99*, Oct. 21-23, pp. 678-683. 1999, Pusan, Korea.



Min-Soeng Kim

He received the B.S. degree from Hanyang university in 1997, and M. S. degree from KAIST in 1999. He is currently working toward the Ph.D. degree in electrical engineering at KAIST. His research interests are artificial intelligence, reinforcement learning and mobile robots.

Ju-Jang Lee

He was born in Seoul, Korea, on November 14, 1948. He received the B.S. and M.S. degrees in electrical engineering from Seoul National University in 1973 and 1977, respectively, and the Ph.D. degree in electrical engineering from the university of Wisconsin in 1984. He is a professor in the department of electrical engineering, KAIST, which he joined 1984. His research interests are in the areas of space robotics, variable structure systems, intelligent mobile robot control, ITS and robust control theory.