한국퍼지 및 지능 시스템학회 논문지 1997, Vol. 7, No. 5. (1997) 60~66

Fuzzy Inference-based Reinforcement Learning of Dynamic Recurrent Neural Networks

Hyo Byung Jun and Kwee Bo Sim

Robotics and Intelligent Information System Lab.
Dept. of Control and Instrumentation Eng., Chung-Ang University
221, Huksuk-Dong, Dongjak-Ku, Seoul 156-756, Korea

ABSTRACT

This paper presents a fuzzy inference-based reinforcement learning algorithm of dynamic recurrent neural networks, which is very similar to the psychological learning method of higher animals. By using the fuzzy inference technique the linguistic and conceptional expressions have an effect on the controller's action indirectly, which is shown in human's behavior. The intervals of fuzzy membership functions are found optimally by genetic algorithms. And using recurrent neural networks composed of dynamic neurons as action-generation networks, past state as well as current state is considered to make an action in dynamical environment. We show the validity of the proposed learning algorithm by applying it to the inverted pendulum control problem.

1. Introduction

Although the term of reinforcement comes from studies of higher animal learning in the experimental psychology, recently it becomes fascinating in the engineering especially as an artificial neural network's learning algorithm in the artificial intelligence.

In general, the machine learning can be classified into two categories, which are supervised and unsupervised learning, by whether the teaching signal is needed or not. In supervised learning, teaching signals from the exact modeling of the environment are needed. On the other hand, in reinforcement learning the exact model is not required, so generally it belongs to unsupervised learning algorithm. And its objective is finding the state-action rule or action generating strategy maximizing reward for the controller or agent's action under dynamically changing environment.

But as is often the case with real world, there is no immediate reinforcement until a goal state is reached. This requires improving long-term conse-quences of an action or of a strategy for performing actions, in addition to short-term consequences. This problem is known as temporal credit assignment problem. A widely studied approach to this problem is to learn an internal evaluation function that is more informative than the evaluation function implemented by the external critic. The representative methods to this problem are actor-critic architecture by Sutton's temporal difference(TD) method[1] and Watkin's Q-learning[2].

In actor-critic architecture, the critic assigns the temporal credit or blame as internal reinforcement using states and external reinforcement. This internal reinforcement is used for the learning of the action network. The critic network is trained by TD-method, and its output is set by the expected discounted sum of future reinforcement as follows[3]:

$$p(t) = E\left\{\sum_{k=0}^{\infty} \gamma^k \cdot r(t+k)\right\} = \sum_{i=1}^{n} \omega_i(t) \cdot x_i(t) \quad (1)$$

where γ is a discounting factor, $r(t)$ is the reinforcement received at time t, $x_i(t)$ is an input measure, and $\omega_i(t)$ is a connection weight. Because the correct predictions must satisfy a consistency condition relating predictions at adjacent time steps, the weight update equation is derived from the TD error back-propagation learning rule:

※ This research was supported by the Chung-Ang University Research Grants in 1997

$$\Delta\omega(t)=\eta\left[r(t)+\gamma\cdot p(t+1)-p(t)\right]\cdot x(t) \quad (2)$$

where η is learning constant.

But it is a difficulty in TD-method that $p(t+1)$ should be calculated using $\omega(t+1)$, not $\omega(t)$. In other words, the reinforcement in TD-method has been predicted by the approximation under the consumption of Markovian environment[4].

And Q-learning needs discrete state and action space so that the large state and action space problem occur. Futhermore it can not cope with continuous state and action problems[5][6].

In this paper, therefore, we propose a fuzzy inference-based reinforcement learning(FIRL) algorithm of dynamic recurrent neural networks. It is thought that human can learn unexperienced task by his reasoning and adaption ability. He does an action just by perceiving the states and then think whether his action is proper or not. After that he estimate how much or less action needed. By doing that, he can act more properly on the next state. In this process, linguistic and conceptional expressions have an effect on his action indirectly. And from the state-reward relations he can reinforce or inhibit his action to a certain situation. This is the reason we use the fuzzy inference as an evaluation function.

In our algorithm, the environment does not need to be modeled. Just by sensing the input states and outputs the agent or controller can adapt the dynamically changing, even though violating the Markov properties, environment. The overview of our learning algorithm is illustrated in Fig. 1[7].

In section 2, fuzzy inference as critic for evaluation of states and actions is discussed and in section 3, the learning algorithm for the dynamic recurrent neural networks made up of associative search units is explained. And then we evaluate the proposed algorithm by applying it to the nonlinear control problem in section 4, then conclusion and discussion follow.

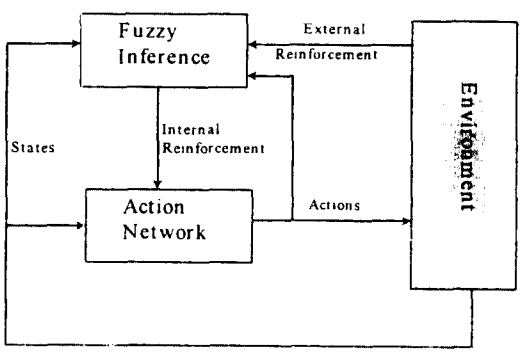


Fig. 1. Block diagram of FIRL

2. Fuzzy Inference

2.1 Generating Reinforcement Signal

In this section, we show the internal reinforcement generating method by fuzzy inference engine. Generally feedforward neural networks are used to generate the internal reinforcement in actor-critic architecture. But we introduce the fuzzy inference engine as critic, whose input variables are composed of state variables, the output of the action network and external reinforcement. And internal reinforcement will be its output.

In fuzzy logic, there are three types of fuzzy reasoning, the first is Mamdani's minimum fuzzy implication rule, the second is Tsukamoto's method with linguistic terms as monotonic membership functions, and the third is that the consequent of a rule is a function of input linguistic variables[8].

In this paper, we use Mamdani's fuzzy implication rule, that is max-min compositional rule of inference. The rules are expressed qualitatively and linguistically by fuzzy IF-THEN rules. If there are m input variables, single output and n fuzzy rules, then general fuzzy production rule is as follows:

$$R_i: IF\ x_1\ is\ A_{i1},\ x_2\ is\ A_{i2},\ \cdots,\ x_m\ is\ A_{im}$$
$$THEN\ y\ is\ B_i \quad (3)$$

Then using the notation of fuzzy relation R, equation (3) can be rewritten as

$$R=R_1\cup R_2\cup\cdots\cup R_n=\bigcup_{i=1}^{n}R_i \quad (4)$$

where $R_i=(A_{i1}\times A_{i2}\times\cdots\times A_{im})\times B_i$.

Generally the antecedent $x_k(k=1,\cdots,m)$ is measured as a crisp value x_k^0. If we have the current inputs x_1^0, x_2^0, $\cdots$, x_m^0, the consequent $B^0(y)$ can be expressed by fuzzy relation R such that

$$B^0(y)=R(x_1^0, x_2^0, \cdots, x_m^0, y). \quad (5)$$

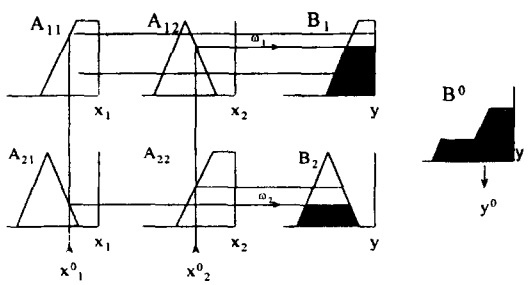


Fig. 2. An example of center of area defuzzification method

Therefore reasoning value y is as follows:

$$B^0(y) = \bigvee_{i=1}^{n} \left[\omega_i \wedge B_i(y) \right]$$

$$\omega_i = A_{i1}(x_1^0) \wedge A_{i2}(x_2^0) \wedge \cdots \wedge A_{im}(x_m^0) \qquad (6)$$

And using the center of area defuzzification method, the final inferred consequent y_0 is given by

$$y^0 = \frac{\int B^0(y) \cdot y \, dy}{\int B^0(y) \, dy}. \qquad (7)$$

Fig. 2 shows an example of the center of area defuzzification method when there are two antecedents.

2.2 Self-Partitioning Membership Function by Genetic Algorithm

As shown in Fig. 3, we use the normalized membership function partitioned with five terms. And the shape of each term is triangular except the two marginal terms. Proper fuzzy partitioning of input and output spaces plays an essential role in achieving a successful fuzzy logic inference engine design. But unfortunately, it is not deterministic and has no unique solution. So we use the Genetic Algorithms (GA) to find the optimal partitions[9]. GA proposed by Hollands in 1975 is one of the derivative-free stochastic optimization methods based on the concepts of natural selection and evolutionary processes.

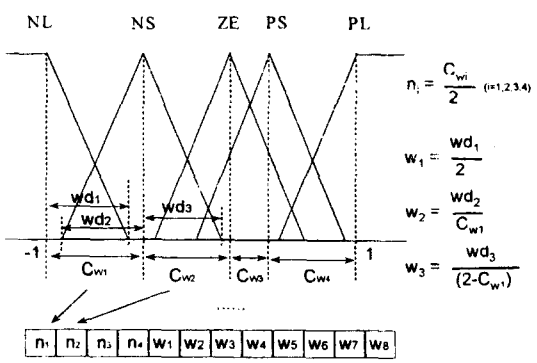


Fig. 3. Membership function and encoding scheme
NL: Negative Large, NS: Negative Smal, ZE: Zero, PS: Positive Small. PL: Positive Large

To apply GA to a problem, at first the solution spaces should be represented by the chromosome. For our case, the encoding method is illustrated in Fig. 3. The triangular membership function's shape is determined by the three points that are a center point and left/right width points. We assume that the NL and PL terms have fixed center points and the other three center points could be placed any position from -1 to 1 and all the left/right width of each terms could be from 0 to the maximum value from its center point to the margin.

For a variable the chromosome is consist of 12 bits real-valued string, where the first 4 bits represent the width proportion between the neighbor center points and the last 8 bits represent the width ratio of each term's left and right margin from its center point. For example, w_3 representing NS term's right width ratio is current right width(wd_3) over its possible maximum width($2\text{-}C_{w1}$). If there are N terms, N_i input variables and N_o output variables, then the whole length of one chromosome becomes $3\times(N\text{-}1)\times(N_i+N_o)$ bits.

This encoding method guarantee the completeness, soundness, and nonredundancy between the solution and the genotype spaces. And using the simple GA process, we can obtain the optimally partitioned membership functions.

3. Stochastic Neuron and Dynamic Recurrent Neural Networks

3.1 Associative Search Unit

Associative search unit was proposed by Klopf as a neuron used in an associative reinforcement learning scheme[1]. Basically this is an extension of the Hebbian correlation learning rule, where the output is a random variable depending on the activation level to exhibit variety in its behavior.

$$s(t)=\sum_{i=1}^{n} w_i(t)x_i(t)$$
$$w_i(t): i-th \ \ weight \ \ vector$$
$$x_i(t): i-th \ \ input \ \ vector \qquad (8)$$

Then the output is as follows:

$$y(t)=\begin{cases}1 \ \ with \ \ p(t)\\ 0 \ \ with \ \ 1-p(t)\end{cases} \qquad (9)$$

where $p(t)$ is an increasing probability function of $s(t)$ ranging from 0 to 1. If critic takes time τ to evaluate an action, then the weight update equation becomes

$$\Delta w(t)=\eta \cdot r(t) \cdot y(t-\tau) \cdot x(t-\tau) \qquad (10)$$

where η is a learning constant, τ is the delayed time and $r(t)$ is the reinforcement at time t.

In the next section, we construct dynamic recurrent neural networks with this associative search unit, and derive the weights update rule.

3.2 Dynamic Recurrent Neural Network

Dynamic recurrent neural network(DRNN) has internal state feedback and self-feedback loops. And DRNN deals with input data nonlinearly, so it shows dynamic characteristic and is useful to the problem having sequential data. The structure of fully connected recurrent neural networks is shown in Fig. 4, made up of asymmetrically interconnected neurons.

The output of the i-th neuron is

$$y_i(t)=f(h_i(t-1))+\Lambda_i(\sigma)$$
$$h_i(t)=\left(\sum_j w_{ij}y_j(t)+x_i(t)\right) \qquad (11)$$

where $h_i(t\text{-}1)$ is the net-input to th i-th neuron at time t-1, $x_i(t)$ is an external input at time t and f(·) is a nonlinear derivative activation function:

$$f(x)=\frac{2}{1+\exp\left(-\frac{2x}{u_0}\right)}-1=\tanh\left(\frac{x}{u_0}\right) \qquad (12)$$

And $\Lambda(\sigma)$ is a gaussian random number with 0 mean and σ as a standard deviation. This standard deviation is set properly as a function of reinforcement signal r whit a proportional constant α as expressed in equation (13). The constant α controls the degree of random search and plays an important role in escaping local minimum.

$$\sigma=\begin{cases}\dfrac{\alpha}{\sum r} & r>0\\ 1 & r=1\\ \alpha\sum|r| & r<0\end{cases} \qquad (13)$$

Then the total cost function to be minimized is

$$E(t)=\frac{1}{2}\sum_k\left(E_k(t)\right)^2 \qquad (14)$$

$$E_k(t)=\begin{cases}\left(1-r(t)\right)\cdot y_k(t), & r(t)\geq 0\\ \left(r(t)-1\right)\cdot y_k(t), & r(t)<0\end{cases} \qquad (15)$$

Equation (15) is an appropriate error measure for the output node k using the reinforcement $r(t)$ created by fuzzy inference. By the gradient-descent method, the change in weights is

$$\Delta w_{pq}(t)=-\eta\frac{\partial E(t)}{\partial w_{pq}}=\eta\sum_k E_k(t)\frac{\partial y_k(t)}{\partial w_{pq}} \qquad (16)$$

The last derivative in equation (16) can be found by differentiating the dynamical rule in equation (11).

$$\frac{\partial y_k(t)}{\partial w_{pq}}=f'(h_k(t-1))\left[\delta_{kp}\,y_q(t-1)+\sum_j w_{kj}\frac{\partial y_j(t-1)}{\partial w_{pq}}\right] \qquad (17)$$

where δ_{kp} is kronecker delta function.

Consequently, from equation (10), (11), (15), (16) and (17) the weight changes to be applied to each weight ω_{pq} in the networks is

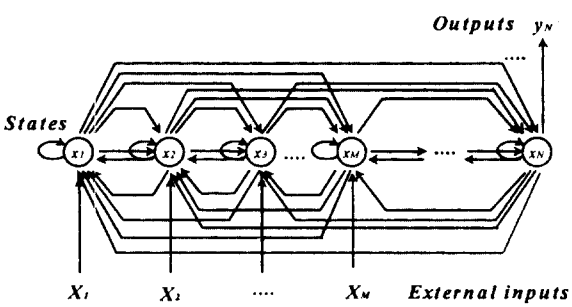


Fig. 4. Dynamic recurrent neural networks

$$\Delta w_{pq}(t) = \eta \cdot r(t) \cdot \sum_k E_k(t) \cdot z_{pq}^k(t) \qquad (18)$$

where $z_{pq}^{k}(t) \triangleq \dfrac{\partial y_k(t)}{\partial w_{pq}}$ for every time step t and all appropriate indices j and k.

This DRNN composed of associative search unit is used as action network and its weights are updated by the equation (18) using reinforcement generated by fuzzy inference engine.

4. Computer Simulation

We verify the effectiveness of the proposed learning algorithm by applying it to the inverted pendulum control simulation. It is assumed that the mathematical model of the system is not known to the control system, and the plant is treated as a black box, but the states of the system are available at instants of time. And it is also assumed that the allowed range of the cart position x and the pole's inclination angle θ are -2.5 m $< x <$ 2.5 m and -12° $< \theta <$ 12°, respectively. The objective is to keep the pole balanced while the cart is constrained to move in a prescribed range.

To generate internal reinforcement for the action networks, as shown in Fig. 5, three dimensional fuzzy rule base is constructed. If membership functions are partitioned into five terms(NL, NS, ZE, PS, PL) and there are n preconditions, then the maximum number of IF-THEN rule is 5^n. In this simulation we made 63 fuzzy rules using two states of inverted pendulum(θ and $\dot{\theta}$) and action network's output(F) as preconditions and internal reinforcement r as a consequent.

We simulated two different cases, one is stabilization control only with regular type membership functions and the other is both stabilization and position control with self- partitioned membership functions. For the first case, the regular type membership functions mean that five terms are separated uniformly with 25% overlapping. If the values of one or more variables are out of the allowed range, FAILURE is given as external reinforcement, and then that trial is terminated.

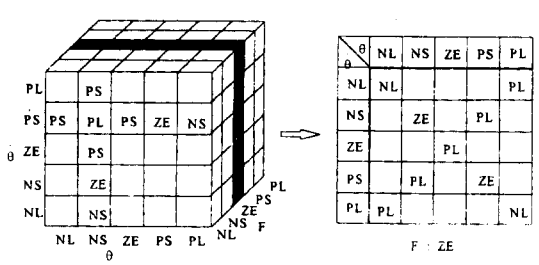


Fig. 5. Fuzzy rule base

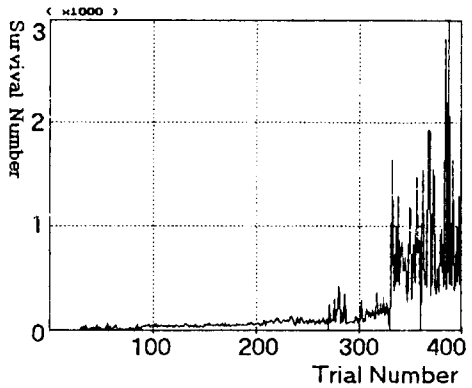


Fig. 6. Course of learning

The simulation result is illustrated in Fig. 6. We plotted the survival number, that is the number of sustaining pole vertically within the allowed region, versus trial number. This result means that the action networks are adapting unknown environment by on-line learning.

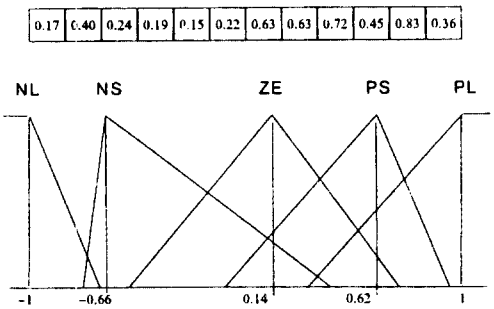
In order to find self-partitioned membership functions, as mentioned in section 2.2, each input and output variables should be expressed as genotype. We found the control parameters of GA heuristically:

- Number of population : 50
- Crossover probability : 0.85
- Mutation probability : 0.02
- Maximum number of generation : 100

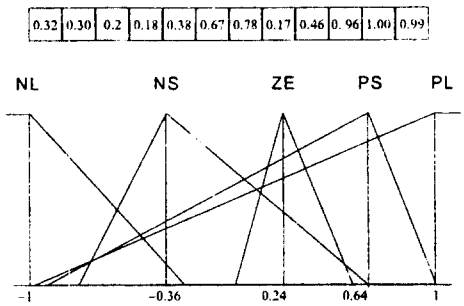
In order to ensure the character preservingness we used multi-point crossover method. The appropriate fitness measure is the sum of survival number and the reciprocal value of position deviation. About the position control, we used the same rule base which is prepared for stabilization control without any additional fuzzy rule. And we set the weighted reinforcement as

$$r(t) = \beta \cdot r_\theta(t) + (1-\beta) \cdot r_x(t) \qquad (19)$$

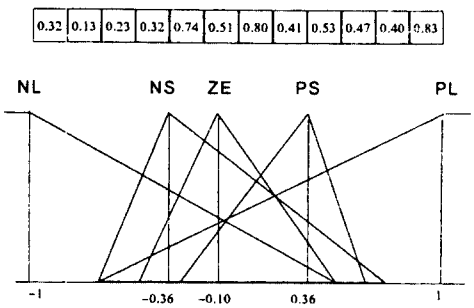
where β is weighting constant, r_θ(t) is reinforcement for the stabilization and $r_x(t)$ is rein-



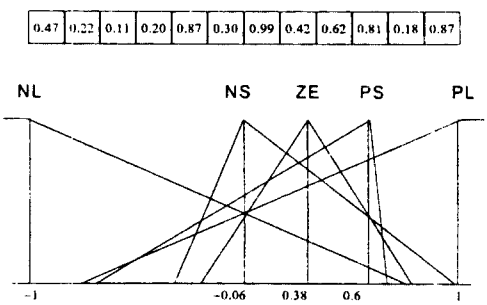
(a) Precondition $X_1(\theta)$



(b) Precondition X_2 ($\dot{\theta}$)



(c) Precondition X_3 (F)



(d) Consequent Y (r)

Fig. 7. Membership functions after self-partitioning

forcement for the position control.

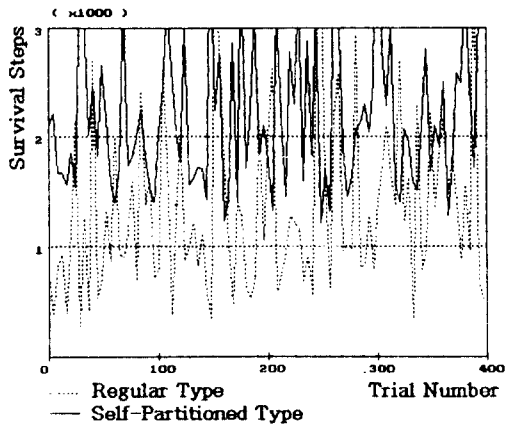
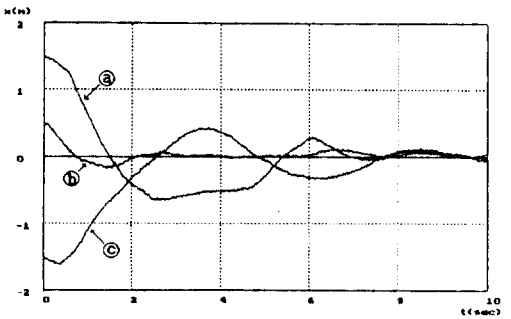


Fig. 8. Performance of regular type and self-partitioned type membership function



Initial position: ⓐ 1.5(m) ⓑ 0.5(m) ⓒ -1.5(m)

Fig. 9. Cart position change

Fig. 7 shows the membership function and best chromosome found by GA. For the simple representation we calculated only down to two places of decimals.

Fig. 8 shows the performance comparison of regular type with the self-partitioned membership function(Fig. 7). The solid line is the survival number change versus trial number when the self-partitioned membership functions are used. As shown in Fig. 8, the solid line stays above 1200 steps, that means the time of keeping the pole balanced is 24 seconds or more, in contrast to the dotted line. Fig. 9 shows the cart position converges toward zero for the different initial positions without additional fuzzy inference rules for position control.

5. Conclusions

In this paper, a reinforcement learning algorithm for dynamic recurrent neural networks has been pro-

posed. The proposed method is unsupervised and on-line learning algorithm, in which the fuzzy inference engine is combined as a critic of the neural networks. And we showed how the fuzzy membership functions are encoded into genotype for self-partitioning by GA.

The proposed learning algorithm basically mimicked human's reasoning and adapting capability, so it is model free and robust to the unexpected noise. The simulation results show the controller can adapt and learn the dynamically changing environment without any prior knowledge of the environment.

References

[1] R.S. Sutton, "Learning to Predict by the Methods of Temporal Differences," *Machine Learning*, vol. 8, pp. 9-44, 1992.

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

[3] A.G. Barto, *The handbook of Brain Theory and Neural Network*, The MIT Press, pp. 804-809, 1995.

[4] P. Martin and J.R. Millan, "Reinforcement Learning of Sensor-based Reaching Strategies for a Two-Link Manipulator," *Proc. IROS 96*, pp. 1345-1352, 1996.

[5] E. Uchibe, M. Asada, K. Hosoda, "Behavior Coordination for a Mobile Robot Using Modular Reinforcement Learning," *Proc. IROS 96*, pp. 1329-1336, 1996.

[6] T. Sawaragi, H. Sawada, O. Katai, "Reinforcement Learning for Autonomous Mobile Robots by Forming Approximate Classificatory Concepts," *Proc. IROS 96*, pp. 1 337-1344, 1996.

[7] H.B. Jun, D.W. Lee, D.J. Kim, K.B. Sim, "Fuzzy Inference-based Reinforcement Learning of Dynamic Recurrent Neural Networks," *SICE Annual Conference of Japan(International Session)*, pp. 1083-1088, 1997.

[8] C.T. Lin, C.S. George Lee, Neural Fuzzy Systems, Prentice Hall PTR, 1996.

[9] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

Hyo-Byung Jun 준회원

Hyo-Byung Jun received the B.S. degree in Department of Control and Instrumentation Engineering from Chung-Ang University, Seoul, Korea, in 1997. He is currently working towards the M.S. degree in Chung-ang University. His research interests are Neural networks, Neuro-Fuzzy and Soft Computing, Evolutionary Computation, Artificial Life and Robot Vision ets.

Kwee-Bo Sim 종신회원

Kwee-Bo Sim received the B.S. and M.S. degrees in Electronic Engineering from Chung-Aug University, Seoul, Korea, in 1984 and 1986 respectively and Ph. D. degree in Electronic Emgineering from the University of Tokyo, Japan, in 1990. From 1987 to 1990, he joined the project of Intelligent Robot System and MEMS at the Institute of Industrial Science(IIS), the University of Tokyo. Since 1991, he has been a faculty member of the School of Electrical and Electronic Engineering at the Chung-Aug University, where he is currently an Associate Professor. His research interests include Artificial Life, Neuro-Fuzzy and Soft Computing, Learning and Evolutionary Algorithms, Autonomous Decentralized System, Intelligent Robot System, Intelligent Control System and MEMS etc. He is a member of IEEE, SICE, RSJ, KITE, KIEE, ICASE and KFIS.