International Journal of Fuzzy Logic and Intelligent Systems, vol. 5, no. 4, Dec 2005 pp. 327-332

Genetically Optimized Hybrid Fuzzy Set-based Polynomial Neural Networks with Polynomial and Fuzzy Polynomial Neurons

Sung-Kwun Oh[1], Seok-Beom Roh[2], and Keon-Jun Park[1]

[1]Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea

[2]Department of Electrical Electronic and Information Engineering, Wonkwang University, 344-2, Shinyong-Dong, Iksan, Chon-Buk, 570-749, South Korea

Abstract

We investigatea new fuzzy-neural networks-Hybrid Fuzzy set based polynomial Neural Networks (HFSPNN). These networks consist of genetically optimized multi-layer with two kinds of heterogeneous neurons thatare fuzzy set based polynomial neurons (FSPNs) and polynomial neurons (PNs). We have developed a comprehensive design methodology to determine the optimal structure of networks dynamically. The augmented genetically optimized HFSPNN (namely gHFSPNN) results in a structurally optimized structure and comes with a higher level of flexibility in comparison to the one we encounter in the conventional HFPNN. The GA-based design procedure being applied at each layer of gHFSPNN leads to the selection leads to the selection of preferred nodes (FSPNs or PNs) available within the HFSPNN. In the sequel, the structural optimization is realized via GAs, whereas the ensuing detailed parametric optimization is carried out in the setting of a standard least square method-based learning. The performance of the gHFSPNN is quantified through experimentation where we use a number of modeling benchmarkssynthetic and experimental data already experimented with in fuzzy or neurofuzzy modeling.

Key Words : Hybrid Fuzzy Set-based Polynomial Neural Networks (HFSPNN), fuzzy set, Fuzzy Polynomial Neural Networks(FPNN), Polynomial Neural Networks (PNN), fuzzy polynomial neuron(FPN), polynomial neuron(PN), genetic algorithms, regression polynomial fuzzy inference

1. Introduction

A lot of researchers on systemmodeling have been interested in the multitude of challenging and conflicting objectives such as compactness, approximation ability, generalization capability and so on which they wish to satisfy. It is common practice to use various forms of neural networks and fuzzy systems in designing nonlinear system with good predictive abilities as well as approximation capabilities. In particular, when dealing with high-order nonlinear and multivariable equations of the model, we require a vast amount of data for estimating all its parameters thatis an important key to determine the model performance. The Group Method of Data Handling (GMDH)[1] introduced by A.G. Ivakhnenko is one of the approaches that help alleviate the problem. But, GMDH has some drawbacks. First, it tends to generate quite complex polynomial for relatively simple systems. Second, owing to its limited generic structure, GMDH also tends to produce an overly complex network(model) when it comes to highly nonlinear systems. In alleviating the problems of the GMDH algorithms, Polynomial Neural Networks(PNN)[2-3] was introduced as a new class of networks. Combination of neural networks and fuzzy systems (or neurofuzzy systems for short) has been recognized as a powerful alternative approach to develop fuzzy systems. We have investigated a new category of neuro-fuzzy networks, Fuzzy Set based Polynomial Neural Networks (FSPNN) and developed Hybrid Fuzzy Set based Polynomial Neural Networks (HFSPNN) composed of multi-layer with two kinds of heterogeneous neurons that are fuzzy set based polynomial neurons (FPSNs) and polynomial neurons (PNs). . Although the HFSPNN has flexible architecture whose potential can be fully utilized through a systematic design, it is difficult to obtain the structurally and parametrically optimized network because of the limited design of the nodes (viz. FSPNs and PNs) located in each layer of the network.

In this paper, we study a genetic optimization-driven new neuro-fuzzy topology, called genetically optimized Hybrid Fuzzy Set based Polynomial Neural Networks (gHFSPNN) and discuss a comprehensive design methodology supporting their development. gHFSPNN is a network resulting from the combination of fuzzy inference system and PNN algorithm driven to genetic optimization. Each node of the first layer of gHFSPNN, that is a fuzzy polynomial neuron (FSPN) operates as a compact fuzzy inference system. The networks of the second and higher layers of the gHFSPNN come with a high level of flexibility as each node (processing element forming a PN). The determination of the optimal values of the parameters available within an individual PN and FSPN (viz. the number of input variables, the order of the polynomial, a collection of preferred nodes, and the number of membership functions (MFs)) leads to a structurally and parametrically optimized network.

Manuscript received Aug. 5, 2005; revised Sep. 7, 2005.
This work has been supported by KESRI (R-2004-B-274), which is funded by MOCIE(Ministry of Commerce, Industry and Energy)

2. The architecture of the hybrid fuzzy set based polynomial neural networks

2.1 The architecture of fuzzy set-based polynomial neurons (FSPN) based layer of gHFSPNN

The FSPN encapsulates a family of nonlinear "if-then" rules. When put together, FSPNs results in a self-organizing Fuzzy Set-based Polynomial Neural Networks (FSPNN). The FSPN consists of two basic functional modules. The first one, labeled by F, is a collection of fuzzy sets (here denoted by $\{A_k\}$ and $\{B_k\}$) that form an interface between the input numeric variables and the processing part realized by the neuron. The second module (denoted here by **P**) refers to the function based nonlinear (polynomial) processing that involves some input variables This nonlinear processing involves some input variables (x_i and x_j), which are capable of being the input variables (Here, x_p and x_q), or entire system input variables. Each rule reads in the form

$$\text{if } x_p \text{ is } A_k \text{ then } z \text{ is } P_{pk}(x_i, x_j, a_{pk}) \quad (1)$$

$$\text{if } x_q \text{ is } B_k \text{ then } z \text{ is } P_{qk}(x_i, x_j, a_{qk})$$

where a_{pk} is a vector of the parameters of the conclusion part of the rule while $P(x_i, x_j, a)$ denoted the regression polynomial forming the consequence part of the fuzzy rule which uses several type of high order polynomials besides the constant function forming the simplest version of the consequence; refer Table 1.

Table 1. Different forms of the regression polynomials forming the consequence part of the fuzzy rules.

No. of inputs / Order of polynomial	1	2	3
0 (Type 1)	Constant	Constant	Constant
1 (Type 2)	linear	Bilinear	Trilinear
2 (Type 3)	Quadratic	Biquadratic-1	Triquadratic-1
2 (Type 4)		Biquadratic-2	Triquadratic-2

The activation levels of the rules contribute to the output of the FSPN being computed as a weighted average of the individual condition parts (functional transformations) $P_{(l,k)}$.

$$z = \sum_{l=1}^{total\ input} \left(\sum_{k=1}^{totalrule_l} \mu_{(l,k)} P_{(l,k)}(x_i, x_j, a_{(l,k)}) \Big/ \sum_{k=1}^{totalrule_l} \mu_{(l,k)} \right) = \sum_{l=1}^{total\ input} \left(\sum_{k=1}^{totalrule_l} \tilde{\mu}_{(l,k)} P_{(l,k)}(x_i, x_j, a_{(l,k)}) \right) \quad (2)$$

$$\tilde{\mu}_{(l,k)} = \frac{\mu_{(l,k)}}{\sum_{k=1}^{\text{total\_rules related to input } l} \mu_{(l,k)}} \quad (3)$$

When developing the FSPN-based layer, we use genetic algorithms to produce the optimized network, which is realized by selecting such parameters as the number of input variables, the order of polynomial, and choosing a specific subset of input variables. Especially for the polynomial type of the consequent part, we consider two kinds of input vector formats in the conclusion part of the fuzzy rules as shown in Table 2.

Table 2. Polynomial type according to the number of input variables in the conclusion part of fuzzy rules

Input vector / Type of the consequence polynomial	Selected input variables in the premise part	Selected input variables in the consequence part	Entire system input variables
Type T	A	A	B
Type T*	A	B	B

2.2 The architecture of the polynomial neuron(PN) based layer of gHFSPNN

As underlined, the PNN algorithm in the PN based layer of gHFSPNN is based on the GMDH method and utilizes a class of polynomials such as linear, quadratic, modified quadratic, etc. to describe basic processing realized there. Let us recall that the input-output data are given in the form

$$(X_i, y_i) = (x_{1i}, x_{2i}, \ldots, x_{Ni}, y_i), \quad i = 1, 2, 3, \ldots, n \quad (4)$$

where N is the number of input variables, i is the data number of each input and output variable, and n denotes the number of data in the dataset.

The estimated output $\hat{y}$ reads as

$$\hat{y} = c_0 + \sum_{i=1}^{N} c_i x_i + \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} x_i x_j + \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} c_{ijk} x_i x_j x_k \quad (5)$$

Where, $C(c_0, c_i, c_{ij}, c_{ijk},)(i, j, k, : 1, 2, \ldots, N)$ and $X(x_i, x_j, x_k,)(i, j, k, : 1, 2, \ldots, N)$ are vectors of the coefficients and input variables of the resulting multi-input single-output(MISO) system, respectively.

The detailed PN involving a certain regression polynomial is shown in Table 3.

Table 3. Different forms of the regression polynomial building a PN

Order / No. of inputs		1	2	3
1	(Type 1)	Linear	Bilinear	Trilinear
2	(Type 2)	Quadratic	Biquadratic-1	Triquadratic-1
	(Type 3)		Biquadratic-2	Triquadratic-2

3. Genetic optimization of gHFSPNN

Genetic algorithms (GAs) are optimization techniques based on the principles of natural evolution [5]. In essence, they are

search algorithms that use operations found in natural genetics to guide a comprehensive search over the parameter space. In this study, for the optimization of the gHFSPNN model, GA uses the serial method of binary type, roulette-wheel used in the selection process, one-point crossover in the crossover operation, and a binary inversion (complementation) operation in the mutation operator. To retain the best individual and carry it over to the next generation, we use elitist strategy [6].

As mentioned, when we construct PNs and FSPNs of each layer in the conventional HFSPNN, such parameters as the number of input variables (nodes), the order of polynomial, and input variables available within a PN and a FSPN are fixed (selected) in advance by the designer. This could have frequently contributed to the difficulties in the design of the optimal network. To overcome this apparent drawback, we resort ourselves to the genetic optimization.

4. The algorithm and design procedure of genetically optimized HFSPNN (gHFSPNN)

The genetically optimized HFSPNN (gHFSPNN) comes witha highly versatile architecture both in the flexibility of the individual nodes as well as the interconnectivity between the nodes and organization of the layers. Overall, the framework of the design procedure of the HFSPNN based on genetically optimized multi-layer perceptron architecture comprises the following steps.

[Step 1] *Determine system's input variables.*

[Step 2] *Form a training and testing data.*

The input-output data set $(X_i, y_i) = (x_{1i}, x_{2i}, , x_{Ni}, y_i)$, i=1, 2, 3, ..., n is divided into two parts, that is, a training and testing dataset.

[Step 3] *Decide initial information for constructing the gHFSPNN structure.*

a) Stopping criterion, b) the maximum number of input variables, c) the total number W of nodes, d) the depth of the gHFSPNN, e) the depth and width of the gHFSPNN to be selected, f) The decision of initial information for fuzzy inference method and fuzzy identification.

[Step 4] *Decide a structure of the PN and FSPN based layer of gHFSPNN using genetic design.*

This concerns the selection of the number of input variables, the polynomial order, and the input variables to be assigned in each node of the corresponding layer. In addition, particularly for the FSPN based layer, we should consider the number of the membership functions. That is why we divide thechromosome to be used for genetic optimization into three sub-chromosomes for PN based layer and four sub-chromosomes for FSPN based layer. The 1st sub-chromosome contains the number of input variables, the 2nd sub-chromosome involves the order of the polynomial of the node, the 3rd (which is used only for the FSPN based layer) involves the number of membership functions (MFs), and the last sub-chromosome (remaining bits) contains input variables coming to the corresponding node (PN and FSPN) and). All these elements are optimized when running the GA.

[Step 5] *Estimate the coefficient parameters of the polynomial in the selected node (PN or FSPN).*

[Step 5-1] ***In case of a PN (PN-based layer)***

The vector of coefficients $\mathbf{C}_i$ is derived by minimizing the mean squared error between y_i and z_{mi}

$$E = \frac{1}{N_{tr}} \sum_{i=0}^{N_{tr}} (y_i - z_{mi})^2 \tag{6}$$

Using the training data subset, this gives rise to the set of linear equations

$$\mathbf{Y} = \mathbf{X}_i\mathbf{C}_i \tag{7}$$

Evidently, the coefficients of the PN of nodes in each layer are expressed in the form

$$\mathbf{Ci} = (\mathbf{X}_i^{\mathrm{T}}\mathbf{X}_i)^{-1}\mathbf{X}_i^{\mathrm{T}}\mathbf{Y} \tag{8}$$

[Step 5-2] ***In case of a FSPN (FSPN-based layer)***

At this step, the regression polynomial inference is considered. The inference method deals with regression polynomial functions viewed as the consequentsof the rules. In the fuzzy inference, we consider two types of membership functions, namely triangular and Gaussian-like membership functions. The consequence parameters are produced by the standard least squares method.

[Step 6] *Select nodes (PNs or FSPNs) with the best predictive capability and construct their corresponding layer.*

The generation process can be organized as the following sequence of steps

Sub-step 1) We set up initialgenetic information necessary for generation of the gHFSPNN architecture.

Sub-step 2) The nodes (PNs or FSPNs) are generated through the genetic design.

Sub-step 3) We calculate the fitness function. The fitness function reads as

$$F(\text{fitness function}) = 1/(1+EPI) \tag{9}$$

where EPI denotes the performance index for the testing data (or validation data).

Sub-step 4) To move on tothe next generation, we carry out selection, crossover, and mutation operation using genetic initial information and the fitness values obtained via ***sub-step 3***.

Sub-step 5) We choose several nodes (PNs or FSPNs) characterized by the best fitness values. Here, we use the pre-defined number W of nodes (PNs or FSPNs) with better predictive capability that need to be preserved to assure an optimal operation at the next iteration of the HFSPNN algorithm. The outputs of the retained nodes serve as inputs to the next layer of the network. There are two cases as to the number of the retained nodes, that is

(i) If $W^* < W$, then the number of the nodes retained for the next layer is equal to z.

Here, W^* denotes thenumber of the retained nodes in each layer that nodes with the duplicated fitness values were moved.

(ii) If $W^* \geq W$, then for the next layer, the number of the retained nodes is equal to W.

Sub-step 6) For the elitist strategy, we select the node that

has the highest fitness value among the selected nodes (W).

Sub-step 7) We generate new populations of the next generation using operators of GAs obtained from ***Sub-step 4***. We use the elitist strategy. This sub-step carries out by repeating ***sub-step 2-6***.

Sub-step 8) We combine the nodes (W populations) obtained in the previous generation with the nodes (W populations) obtained in the current generation.

Sub-step 9) Until the last generation, this sub-step carries out by repeating ***sub-step 7-8***.

[Step 7] *Check the termination criterion.*

As far as the performance index is concerned (that reflects a numeric accuracy of the layers), a termination is straightforward and comes in the form,

$$F_1 \le F^* \tag{10}$$

Where, F_1 denotes a maximal fitness value occurring at the current layer whereas F^* stands for a maximal fitness value that occurred at the previous layer.

[Step 8] *Determine new input variables for the next layer.*

If (10) has not been met, the model is expanded. The outputs of the preserved nodes (z_{1i}, z_{2i}, ..., z_{Wi}) serves as new inputs to the next layer (x_{1j}, x_{2j}, ..., x_{Wj})($j=i+1$). This is captured by the expression

$$x_{1j} = z_{1i},\ x_{2j} = z_{2i},\ \ldots,\ x_{Wj} = z_{Wi} \tag{11}$$

The HFSPNN algorithm is carried out by repeating steps 4-8 of the algorithm.

5. Simulation study

We demonstrate how the gHFSPNN can be utilized to predict future values of a chaotic Mackey-Glass time series [4, 7-12]. The time series is generated by the chaotic Mackey-Glass differential delay equation [13] comes in the form

$$\dot{x}(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{12}$$

To obtain thetime series value at each integer point, we applied the fourth-order Runge-Kutta method to find the numerical solution to (12). From the Mackey-Glass time series x(t), we extracted 1000 input-output data pairs in the following format:

$$[x(t-24),\ x(t-18),\ x(t-12),\ x(t-6),\ x(t)\ ;\ x(t+6)]$$

where, t=118 to 1117. The first 500 pairs were used as the training data set while the remaining 500 pairs formed the testing data set. To come up with a quantitative evaluation of the network, we use the standard RMSE performance index as given by (13).

$$E(PI\ or\ EPI) = \sqrt{\frac{1}{N}\sum_{p=1}^{N}(y_p - \hat{y}_p)^2} \tag{13}$$

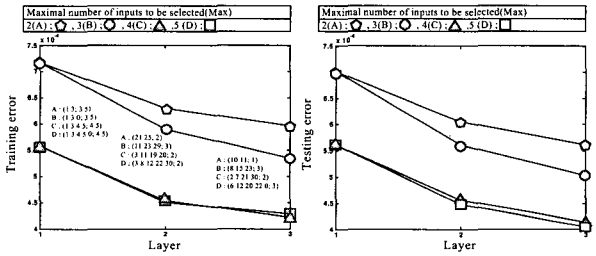
Table 4summarizes the list of parameters used in the genetic optimization of the network.

Table 4. Summary of the parameters of the genetic optimization

	Parameters	1st layer	2nd to 3th layer
GA	Maximum generation	150	150
	Total population size	60	60
	Selected population size (W)	30	30
	Crossover rate	0.65	0.65
	Mutation rate	0.1	0.1
	String length	3+3+2+30	3+3+30
HFS PNN	Maximal no.(Max) of inputs to be selected	$1 \le l \le$ Max (2~5)	$1 \le l \le$ Max (2~5)
	Polynomial type (Type T) of the consequent part of fuzzy rules	$1 \le T \le 4$	$1 \le T \le 4$
	Consequent input type to be used for Type T (*)	Type T*	Type T*
	Membership Function (MF) type	Triangular	
		Gaussian	
	No. of MFs per input	$2 \le T \le 5$	

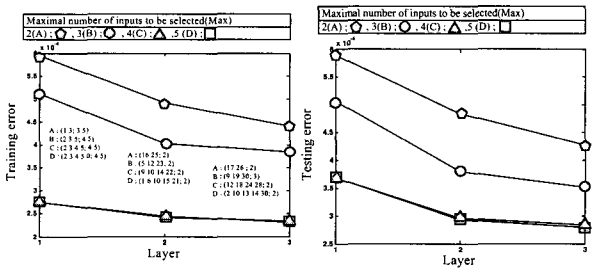
l, T, Max: integers, T* means that entire system inputs are used for the polynomial in the conclusion part of the rules.

Fig. 1 depicts the performance index of each layer (FSPN-based or PN-based layer) of gHFSPNN for each type (Type T*: that is, entire system inputs) according to the increase of maximal number of inputs to be selected.



(a-1) Training error (a-2) Testing error

(a) Triangular MF

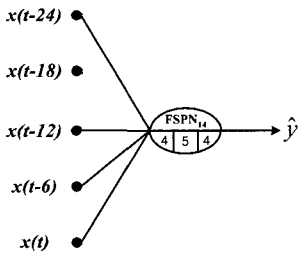


(b-1) Training error (b-2) Testing error

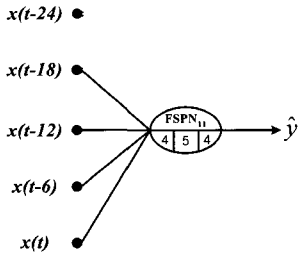
(b) Gaussian-likeMF

Fig. 1.Performance index according to the increase of number of layers (Type T*)

Fig. 2(a)-(b) illustrate the detailed optimal topologies of gHFSPNN for 1 layer and Max=5 in case of Type T*: those are quantified as PI=5.58e-4, EPI=5.61e-4 for triangular MF, and PI=2.74e-4, EPI=3.68e-4 for Gaussian-like MF.



(a) Triangular MF(1 layer and Max=5)



(b) Gaussian-like MF(1 layer and Max=5)

Fig. 2. gHFSPNNarchitecture in case of using entire system input vector format (Type T*)

Table 5 gives a comparative summary of the network with other models.

Table 5. Comparative analysis of the performance of the network; considered are models reported in the literature

Model				PI	PIs	EPIs	NDEI*
Wang's model[7]				0.044			
				0.013			
				0.010			
ANFIS[8]					0.0016	0.0015	0.007
FNN model[9]					0.014	0.009	
Recurrent neural networks[10]				0.0138			
SONN** [11]	Basic (5th layer)		Case 1		0.0011	0.0011	0.005
			Case 2		0.0027	0.0028	0.011
	Modified (5th layer)		Case 1		0.0012	0.0011	0.005
			Case 2		0.0038	0.0038	0.016
HFPNN[4]	Triangular		5th layer		7.0e-4	6.0e-4	
	Gaussian		5th layer		4.8e-5	7.1e-5	
Proposed gHFSPNN	Max=5 (Type T*)	Triangular	3rd layer		4.28e-4	4.05e-4	
	Max=5 (Type T*)	Gaussian	3rd layer		2.30e-4	2.80e-4	

*Non-dimensional error index (NDEI) as used in [12] is defined as the root mean square errors divided by the standard deviation of the target series. ** is called "conventional optimized FPNN".

6. Concluding remarks

In this study, the GA-based design procedure of Hybrid Fuzzy Set based Polynomial Neural Networks (HFSPNN) along with its architectural considerations has been investigated. Through the consecutive generation of a layer through a growth process (iteration) of the gHFSPNN, the depth (layer size) and width (node size of each layer) of the network could be flexibly selected based on a diversity of local characteristics of these preferred FSPNs and PNs (such as the number of input variables, the order of the consequent polynomial of rules/the polynomial order, a collection of specific subset of input variables, and the number of membership functions) available within HFSPNN. The design methodology comes as a hybrid structural optimization (based on GMDH method and genetic optimization) and parametric learning being viewed as two fundamental phases of the design process. The comprehensive experimental studyinvolving well-known datasets quantify a superb performance of the network in comparison to the existing fuzzy and neuro-fuzzy models. Most importantly, through the proposed framework of genetic optimization we canefficiently search for the optimal network architecture (structurally and parametrically optimized network) and this becomes crucial in improving the performance of the resulting model.

References

[1] Ivahnenko. A.G., "Polynomial theory of complex systems," *IEEE Trans. on Systems, Man and Cybernetics.* SMC-12, pp. 364-378, 1971

[2] S.-K. Oh and W. Pedrycz., "The design of self-organizing Polynomial Neural Networks," *Information Science.* Vol. 141, pp. 237-258, 2002

[3] S.-K. Oh, W. Pedrycz and B.-J. Park, "Polynomial Neural Networks Architecture: Analysis and Design," *Computers and Electrical Engineering.* Vol. 29, pp. 703-725, 2003

[4] S.-K. Oh, W. Pedrycz, and D.-W. Kim., "Hybrid Fuzzy Polynomial Neural Networks," *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems.* Vol. 10, pp. 257-280, 2002

[5] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs," *Springer-Verlag,* Berlin Heidelberg. 1996

[6] D. Jong, K. A., Are *Genetic Algorithms Function Optimizers?* Parallel Problem Solving from Nature 2, Manner, R. and Manderick, B. eds., North-Holland, Amsterdam

[7] L. X. Wang, J. M. Mendel, "Generating fuzzy rules from numerical data with applications," *IEEE Trans. Systems, Man, Cybern.* Vol. 22, pp. 1414-1427, 1992

[8] J. S. R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Trans. System, Man, and Cybern.* Vol. 23 pp. 665-685, 1993

[9] L. P. Maguire, B. Roche, T. M. McGinnity, L. J. McDaid, "Predicting a chaotic time series using a fuzzy neural network," *Information Sciences.* Vol. 112, pp. 125-136, 1998

[10] C. James Li, T. -Y. Huang, "Automatic structure and parameter training methods for modeling of mechanical systems by recurrent neural networks," *Applied Mathematical Modeling.* Vol. 23, pp. 933-944, 1999

[11] S.-K. Oh, W. Pedrycz, T.-C. Ahn, "Self-organizing neural networks with fuzzy polynomial neurons," *Applied Soft Computing.* Vol. 2, pp. 1-10, 2002

[12] A. S. Lapedes, R. Farber, *Non-linear Signal Processing Using Neural Networks: Prediction and System Modeling,* Technical Report LA-UR-87-2662, Los Alamos National Laboratory, Los Alamos, New Mexico 87545. 1987

[13] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science.* Vol. 197, pp. 287-289, 1977

[14] S-K. Oh, *Advanced Hybrid Fuzzy Inference Systems by Programming*, NAEHA Publishing Co., Nov. 2005.

[15] S-K. Oh, *Advanced HFIS-related Materials Published in International Journals*, NAEHA Publishing Co., Nov. 2005.

Sung-Kwun Oh

He received the BSc, MSc, and PhD. degrees in Electrical Engineering from Yonsei University, Seoul, Korea, in 1981, 1983, and 1993, respectively. During 1983-1989, he was a Senior Researcher of R&D Lab. of Lucky-Goldstar Industrial Systems Co., Ltd. From 1996 to 1997, He held a position of a Postdoctoral fellow in the Department of Electrical and Computer Engineering at the University of Manitoba, Canada. He is currently a Professor in the Dept. of Electrical Engineering, The University of Suwon, Korea. His research interests include fuzzy systems, fuzzy-neural networks, automation systems, advanced Computational Intelligence, and intelligent control. He is a member of IEEE. He currently serves as an Associate Editor of KIEE Transactions on Systems & Control and Journal of Control, Automation, and Systems Engineering of the ICASE, Korea.
E-mail : ohsk@suwon.ac.kr

Seok-Beom Roh

He received his BSc and MSc degrees in Control and Instrumentation Engineering from Wonkwang University, Korea in 1994 and 1996, respectively. He is currently a PhD candidate at the same institute. His rersearch interests include fuzzy and hybrid systems, neurofuzzy models, and Computational Intelligence.

Keon-Jun Park

He received the B.S degrees in electronics engineering and the M.S in control and instrumentation engineering from Wonkwang University in 2003 and 2005 respectively. He is now pursuing his Ph.D degree in electrical engineering from The University of Suwon. His research interests include FIS, Neural Networks, GAs, optimization theory, Computational Intelligence and Automation control.