

TUNING OF A NEURO-FUZZY CONTROLLER BY GENETIC ALGORITHM

Teo Lian Seng, Marzuki Khalid*, and Rubiyah Yusof

Centre for Artificial Intelligence and Robotics,
University Teknologi Malaysia,
Jalan Semarak, 54100 Kuala Lumpur, Malaysia.
Email address: marzuki@klred.utm.my
Fax number: 603-2904892
(All correspondence should be sent to *)

ABSTRACT

Due to their powerful optimization property, genetic algorithms (GAs) are currently being investigated for the development of adaptive or self-tuning fuzzy logic control systems. This paper presents a neuro-fuzzy logic controller (NFLC) where all of its parameters can be tuned simultaneously by GA. The structure of the controller is based on the Radial Basis Function neural network (RBF) with Gaussian membership functions. The NFLC tuned by GA can somewhat eliminate laborious design steps such as manual tuning of the membership functions and selection of the fuzzy rules. The GA implementation incorporates dynamic crossover and mutation probabilistic rates for faster convergence. A flexible position coding strategy of the NFLC parameters is also implemented to obtain near optimal solutions. The performance of the proposed controller is compared with a conventional fuzzy controller and a PID controller tuned by GA. Simulation results show that the proposed controller offers encouraging advantages and has better performance.

1. INTRODUCTION

Fuzzy logic control systems, which have the capability of transforming linguistic information and expert knowledge into control signals [1-2], are currently being used in a wide variety of engineering applications [3-7]. The simplicity of designing these fuzzy logic systems has been the main advantage of their successful implementation over traditional approaches such as optimal and adaptive control techniques. Despite the advantages of the conventional fuzzy logic controller (FLC) over traditional approaches, there remain a number of drawbacks in the design stages. Even though rules can be developed for many control applications, they need to be set up through expert observation of the process. The complexity in developing these rules increases with the complexity of the process. FLCs also consist of a number of parameters that are needed to be selected and configured in prior, such as selection of scaling factors,

configuration of the center and width of the membership functions, and selection of the appropriate fuzzy control rules.

Due to their learning capability, artificial neural networks are being sought in the development of neuro-fuzzy controllers or adaptive FLCs. Berenji [8] developed a FLC that is capable of learning as well as tuning of its parameters by using neural networks trained through a reinforcement learning method. Jang [9] developed a self-learning FLC based on a neural network trained by temporal back-propagation. Lee et.al. [10] proposed a self-organizing fuzzified basis function based on the competitive learning scheme.

A more recent technique in implementing adaptive or self-tuning FLCs is by using genetic algorithms (GAs). Karr and Gentry [11,12] applied GA in the tuning of fuzzy membership functions which was applied to a pH control process and a cart-pole balancing system. Kim et.al. [13] used a similar method, however, with different shapes of fuzzy membership functions applied to different processes. Varsek et.al.[14] used GAs to tune FLC in three phases: learning of basic control rules, rules compression and fine tuning. Hwang and Thomson [15] used GAs to search for optimal fuzzy control rules with prior fixed membership functions. Hu et.al. [16] showed how cell-map information can be incorporated with GA for tuning output variable parameters of the Takagi-Sugeno type of FLC. While Homaifar and McCormick [17] optimized a FLC which is applied to a number of applications, however, the width of the membership functions and output fuzzy variables are already predefined.

In the above applications, there are just too many parameters involved in the development of the FLCs, which if all of them are encoded, will result in rather long strings, and thus will increase the complexity of the problem. A number of the above applications uses GA in tuning of the fuzzy controller parameters on a stage by stage basis. However, partial or stage by stage optimization of the FLC parameters and control rules restrict the searching spaces of GA, thus, causing higher possibility of partial or sub-optimal solutions. As each of the design stages of the FLC may not be independent, it is important to consider and optimize them simultaneously.

There have been efforts where tuning of the FLC parameters are being done simultaneously by GA. Lee and Takagi [18] proposed a method of determining the parameters of a Takagi-Sugeno type of FLC, which used chromosomes of 2880-bit length strings. In another development, Shimojima et.al. [19] used GA to tune a type of RBF based fuzzy model, with only three fuzzy memberships for each fuzzy variable. However, the fuzzy system was used to model a mathematical function, and was not implemented as a FLC.

By using a FLC based on the RBF neural networks, which we labeled as NFLC, this paper proposed a simultaneous tuning strategy of all of its parameters by GA, without the need

to perform partial or sequential optimization. The RBF neural network which forms the basis of the NFLC [20-22], is used as the fuzzy reasoning mechanism. All the Gaussian input membership functions and the weights of the NFLC are tuned by the GA. By using this NFLC structure, no fuzzy output membership functions and fuzzy control rules are needed to be defined. Instead, the weights are being optimized to determine the appropriate control action for each of the fuzzy control conditions.

Unlike in [19], where each fuzzy membership function consists of 3 parameters, in this paper, the Gaussian functions are chosen as they are characterized by only two parameters, thus, enabling more fuzzy memberships to be encoded in a specific length of chromosome. In tuning the NFLC for function learning or modelling purposes, the GA objective function is rather straight forward, i.e., to minimize the error between the actual data and fuzzy model output [19]. However, if the fuzzy system is applied as a controller, the GA objective function needs to be defined differently. Several forms of objective functions are formulated in this paper which are applied to three different control systems. The proposed NFLC structure takes less parameters as compared to a conventional FLC [2] or the Takagi-Sugeno type of FLC [18]. This resulted in a shorter coded string which allows GA to search more efficiently.

The GA is implemented using dynamic crossover and mutation probability rates for better exploitation of the optimal NFLC parameters [23,24]. Furthermore, a flexible position coding strategy for the fuzzy membership functions is introduced to configure the fuzzy membership partitions in the corresponding universes of discourse. The proposed methodology is then applied and tested on three different plants: an open loop unstable with non-minimum phase high-order plant, a nonlinear plant and a car parking simulation. All of the initial GA populations are randomized, which implies that minimum heuristic control knowledge is used. The appropriate NFLC parameters evolved accordingly and are tuned gradually throughout the GA iterations.

This paper has been organized as follows, first, the implementation of a simplified fuzzy logic control algorithm based on the RBF neural network structure, which forms the basis of the proposed NFLC, is discussed. This is followed by a brief introduction to GA. The implementation of the proposed algorithm which involves coding of the controller parameters, the optimization process and selection of fitness function are next discussed. The proposed methodology which is applied to three different systems are detailed in the sections that follow. The performance of the proposed NFLC is then compared to a PID controller tuned by GA and also a conventional FLC.

2. DESCRIPTION OF THE NEURO-FUZZY CONTROLLER

This section discusses the formulation of the NFLC, which implements a simplified fuzzy logic control algorithm based on the radial basis function neural network [10,20,25]. The RBF neural network is usually used to approximate a continuous linear or nonlinear function mapping. Its structural and computational detail can be referred in [26, 27]. The structure of the multi-input and multi-output NFLC is shown as in Fig. 1. The input layer accepts the system state feedback $(x_1, x_2, \dots, x_n, \dots, x_N)$ (input vector), and the fuzzy inferencing is processed at the hidden layer. The strength of the control action for each of the fuzzy rules is given by the interconnected weights between the hidden and the output layers. The output layer implements the normalization operation to produce the control signals $(y_1, y_2, \dots, y_m, \dots, y_M)$. The RBF structure can be used to implement the fuzzy control rules which are written as:

$$\begin{aligned} & \text{IF } (X_1^i) \text{ and } \dots (X_n^i) \dots \text{and } (X_N^i) \\ & \text{THEN } (w_{i1}) \text{ and } \dots (w_{im}) \dots \text{and } (w_{iM}) \end{aligned} \quad (2.1)$$

where w_{im} is the singleton defined control action for the i^{th} control rule of the m^{th} output variable.

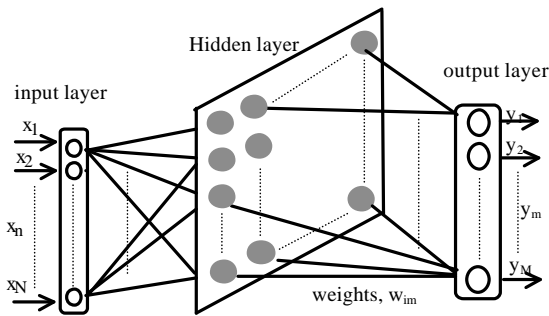


Fig. 1 The architecture of the NFLC based on the RBF neural network.

In order to further visualize this concept, consider a NFLC implemented upon this structure, which has two input variables, namely the error (e) and the change of the error (Δe). Each of these variables takes five Gaussian type of fuzzy membership functions that are labeled as positive big (PB), positive small (PS), zero (Z), negative small (NS), negative big (NB). Each of the membership functions has two parameters, i.e., the center and width of the Gaussian functions. The multi-variate Gaussian can also be viewed as the product of a single-variate Gaussian function. It performs a conjunctive operation in the ‘premise’ part of the fuzzy rules in the hidden layer. Figure 2 shows the rule base matrix of the corresponding fuzzy basis units at the hidden layer of the controller. Each of the kernel squares represents one control rule condition. Thus, the number of the hidden nodes for this network is exactly equal to the number of fuzzy control rules. The output from these units is the matching degree or inferred result (h_i) of the particular fuzzy control rules.

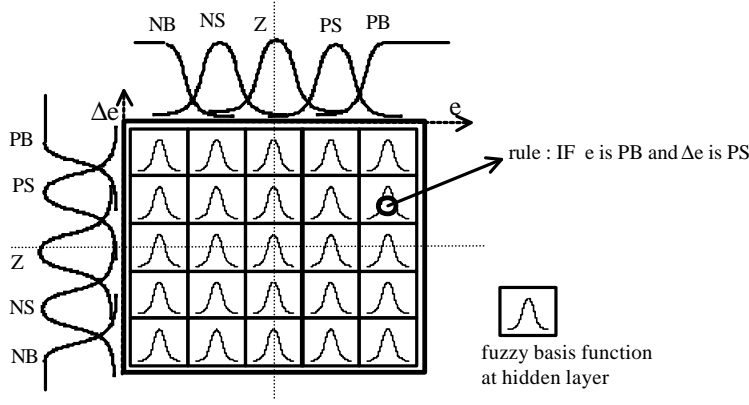


Fig. 2 The fuzzy basis function at the hidden layer.

Basically, fuzzy logic control involves three main stages: fuzzification, inferencing, and defuzzification. This fuzzy inference mechanism can be further simplified to as only pattern matching and weights averaging, thereby, eliminating the procedures of fuzzification and defuzzification [20,25]. The first operation deals with the *IF* part of the fuzzy control rules; it determines the matching degree of the current input to the condition of each of the fuzzy control rules. Some examples of the matching degree computation methods are proposed in [20,25], however, a different approach is used in this paper. By characterizing the fuzzy input membership functions with only two parameters (C_x and D_x), and using the Gaussian membership functions, the matching formula can be written as follows:

$$h_i = \exp \left(- \left\| \frac{C_{x,n}^i - x_n}{D_{x,n}^i} \right\|^2 \right) \quad \text{for } i = 1 \text{ to } T \quad (2.2)$$

where T is the total number of fuzzy rules. $C_{x,n}^i$ and $D_{x,n}^i$ denote the center and the width of n^{th} input variable's membership assigned to the i^{th} control rules, respectively. While $\|\cdot\|$ is the norm operator presented as either Euclidean, Hamming, Maximum, etc.. The matching degree process is simply an operation that returns the matching level, $h_i \in [0,1]$ between the inputs and the rule pattern for the i^{th} rule. A matching degree of '1' means that a full match occurs to that rule, while a small h_i indicates poor matching between the input pattern and the particular rule pattern [25].

The weights are then averaged to obtain the control action of each output variable. Thus for the m^{th} controller output, y_m can be computed by normalizing the weights as follows :

$$y_m = \frac{\sum_{i=1}^P (h_i \cdot w_{im})}{\sum_{i=1}^T (h_i)} \quad (2.3)$$

Figure 3 shows graphically how such computation is carried out. It can be viewed as the modified center of gravity defuzzification strategy. The algorithm, can be understood as a modification of the maximum membership decision scheme, where the global center is

calculated by the center of gravity algorithm. The controller output y_m is a crisp value that can be readily applied to the system. GA is then implemented as an optimization algorithm to tune all the parameters of this NFLC, which is discussed in the next section.

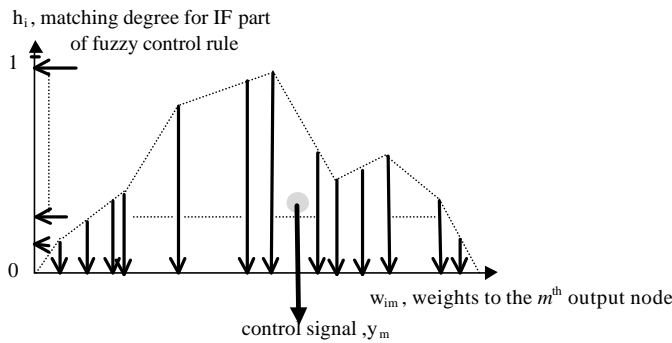


Fig. 3 Computation of the output value of the NFLC.

3. DESIGN OF THE NEURO-FUZZY CONTROLLER BY GA

3.1 Genetic Algorithm

Genetic Algorithm (GA) is a random search technique that imitates natural evolution with Darwinian survival of the fittest approach. GAs perform on the coding of the parameters and not on the exact parameters, therefore, it does not depend on the continuity of the parameter nor the existence of derivatives of the functions as needed in some conventional optimization algorithms. The coding method allows GAs to handle multi-parameters or multi-model type of optimization problems easily, which is rather difficult or impossible to be treated by classical optimization methods.

The population strategy enables GA to search the near optimal solutions from various parts and directions within a search space simultaneously. Therefore, it can avoid converging to the local minimum or maximum points better. GA processes each chromosome independently and make it highly adaptable for parallel processing. It needs no more than only the relative fitness of the chromosomes, thus, it is rather suitable to be applied to systems that are ill-defined. GAs can also work well for non-deterministic systems or systems that can only be partially modelled. GAs use random choice and probabilistic decision to guide the search, where the population improves toward near optimal points from generation to generation.

GAs consist of three basic operations: reproduction, crossover, and mutation. Reproduction is the process where members of the population reproduced according to the relative fitness of the individuals, where the chromosomes with higher fitness have higher probabilities of having more copies in the coming generation. There are a number of selection schemes available for reproduction, such as 'roulette wheel', 'tournament scheme', 'ranking scheme', etc. [28,29]. Crossover in GA occurs when the selected chromosomes exchange

partially their information of the genes, i.e., part of the string is interchanged within two selected candidates. Mutation is the occasionally alteration of states at a particular string position. Mutation is essentially needed in some cases where reproduction and crossover alone are unable to offer the global optimal solution. It serves as an insurance policy which would recover the loss of a particular piece of information. Further discussion on GAs can be obtained in [28 -29].

3.2 Tuning of the NFLC Parameters by GA

This section discusses how the proposed NFLC is formulated by using the GA approach, where all the parameters of the NFLC are initially randomized, then being tuned and optimized simultaneously by GA.

A. Coding strategy of the NFLC parameters

In this paper, the NFLC as shown in Fig.1 is configured to have two inputs (X_1, X_2) and one output (y), which is the controlled variable. Each of the Gaussian membership functions has a center $C_{x_1}^i$ ($C_{x_2}^i$) and the width $D_{x_1}^i$ ($D_{x_2}^i$) for the inputs X_1 and X_2 respectively. In the following experiments, each of the input fuzzy variables is quantified into five membership functions, therefore, resulting in 20 parameters. Our initial investigations showed that increasing the number of membership functions do not significantly improve the experimental results. Furthermore, it increases the complexity of the GA searching process. On the other hand, reducing the number of membership functions, however, does have an effect on the accuracy of the problems. Thus, the membership functions were chosen judiciously. With these fuzzy input membership functions, there are 25 fuzzy radial units (5x5) at the hidden layer, where, 25 weights (w_{ij}) are needed to connect the hidden units to the output node, given as $w_{ij}=\{w_{11}, w_{12}, w_{13}, \dots, w_{21}, w_{22}, \dots, w_{54}, w_{55}\}$. Thus, a total of 45 parameters ($5_{\text{membership_functions}} \times 2_{\text{parameters}} \times 2_{\text{variables}} + 25_{\text{weights}}$) are needed to be tuned by the GA, which is much less than the parameters of a conventional FLC and also the Takagi-Sugeno FLC when configured in the same manner.

As GAs deal with coded parameters, all the NFLC parameters that need to be tuned must be encoded into a finite length of string. The Linear Mapping Method [28,29] is used for this purpose, which can be expressed as follows :

$$g_q = G_{qmin} + (G_{qmax} - G_{qmin}) \cdot A_q / (2^N - 1) \quad (3.1)$$

where g_q is the actual value of the q^{th} parameter, A_q is the integer represented by a N -bit string gene. G_{qmax} and G_{qmin} are user-defined upper and lower limits of the gene, respectively. The encoded genes are concatenated to form a complete *chromosome*. Each of the parameters is

encoded into 8-bit strings, resulted in a complete chromosome of 360 bits. The coded parameters of the NFLC are arranged as shown in the following equation to form the chromosome of the population:

$$\begin{array}{l}
 \text{gene} \quad \quad \quad | 1 | 2 | \dots | 9 | 10 | 11 | 12 | \dots | 19 | 20 | 21 | 22 | \dots | 45 | \\
 \text{chromosome} | \text{sub-chromosome of } X_1 | \quad | \text{sub-chromosome of } X_2 | \text{sub-chromosome of weights} | \\
 \text{parameter} \quad | \dots (C_{x_1}^i, D_{x_1}^i) \dots | \dots (C_{x_2}^l, D_{x_2}^l) \dots | w_{11} | w_{12} | \dots | w_{il} | \dots | w_{55} | \quad (3.2)
 \end{array}$$

It can be observed that Gene 1 to Gene 10 is allocated to the sub-chromosome of the first controller input (X_1) with the centers of membership functions (Gene 1,3,5,7,9) and the corresponding membership widths (Gene 2,4,6,8,10) at the antecedent position. Gene 11 to Gene 20 are assigned in a similar way for the second controller input (X_2).

In this paper, the flexible position coding strategy is applied to improve the diversity of possible input spaces partition. The order of the center-width pair genes in the sub-chromosome is not fixed following the order of the corresponding membership functions in the universe of discourse. Each time when the genes are decoded for fitness evaluation, the fuzzy memberships are rearranged in an ascending manner in its universe of discourse based on the center of the membership functions. The respective weight (w_{il}) is then assigned to the respective control rule condition of “IF ($C_{x_1}^i, D_{x_1}^i$) AND ($C_{x_2}^l, D_{x_2}^l$)” as given in (3.2).

B. Optimization by GA

To describe the GA optimization process, consider the functional block diagram as shown in Fig.4. At the beginning of the process, the initial populations comprise a set of chromosomes that are scattered all over the search space. The initial population may be randomly generated or may be partly supplied by the user. However, in all our experiments, the population consists of 200 chromosomes which are all randomized initially. Thus, the use of heuristic knowledge of the controller is minimized.

The assignment of the fitness in GA serves as a guidance to lead the search towards the optimal solution. After each of the chromosomes is evaluated and associated with a fitness, the current population undergoes the reproduction process to create the next generation of population. The ‘Roulette wheel’ selection scheme is used to determine the members of the new generation population [28]. After the new group of population is built, the mating pool is formed and the crossover is carried out. This is then followed by the mutation operation. Generally, after these three operations, the overall fitness of the population improves. Each of the population generated then goes through a series of evaluation, reproduction, crossover and mutation, the procedure is repeated until the termination condition is reached. After the evolution process, the final generation of population consists of highly fit strings that provide optimal or near optimal solutions.

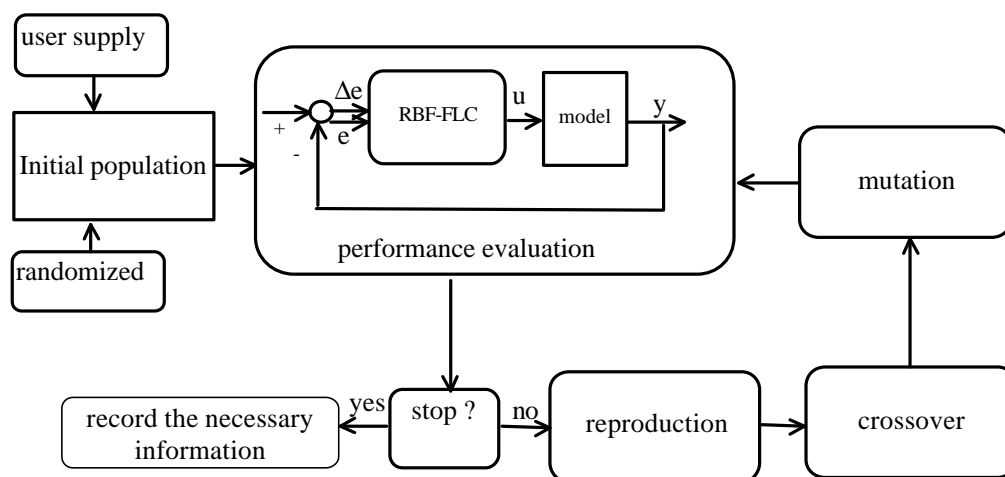


Fig. 4 A functional block diagram showing the GA optimization process.

In the evaluation routine for control purposes, one chromosome is taken and decoded to the actual value of the parameters. These sets of controller parameters are then used to control the system where it undergoes a series of tracking response of multi-step reference setpoints. The use of a multi-step reference signal is to excite the different states of the system, to enable the evaluation to cover a wider system operating range. Based on the various states of the

control system, the performance of the controller is calculated by using a predefined cost function. GA is then used to tune the controller parameters to minimize the cost function.

C. Initialization of the GA parameters

Dynamic crossover and mutation probability rates are used in the GA operation, as they provide faster convergence when compared to constant probability rates [23]. Figure 5 shows the crossover and mutation rates that are changed dynamically in the evolution process. The crossover rate is set high at the beginning of the generation and decreases exponentially during the generations. At the beginning of the GA iterations, the randomized initial GA population is diverse, i.e. pieces of good solution are scattered throughout the search space. As the crossover operator can put together the small pieces [29], it is set to be relatively high at the beginning of the iteration. Over the iterations, these pieces would then be assembled, i.e., the population converged to smaller sections in the search space. Mutation is the operation used to further exploit the improved solution in the established region of the current best solution. This accounts for the increment of binary mutation as the iterations proceeded. Note that, in early generations as the members of the population are very distinct, mutation is not really needed, and kept almost to zero. This technique is applied as it helps the convergence of the GA without much loss of solution optimality as reported in [23,24], such that the consistency of obtaining the final solution is always maintained.

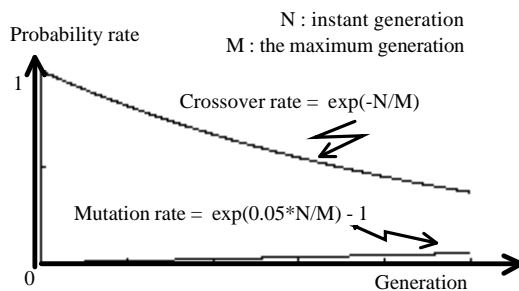


Fig.5 The dynamic crossover and mutation probability rates.

The proposed tuning of the NFLC involved 200 chromosomes which are all initially randomized. The Gray-Code transformation method is applied as it can enhance the GA searching engine [29]. Apart from that, the elitist strategy is employed to the selection scheme, which means the fittest chromosome has one copy directly to the new generation. In addition, a generation gap of 0.9 is used during the reproduction operation which means 90% of the members in the new population are determined by the selection scheme employed, and the remaining 10% is selected uniformly from the previous generation. This strategy helps to

prevent premature convergence of the population. A two-point crossover is applied in exchanging the gene information.

The proposed NFLC tuned by GA is tested on three different plants: a non-minimum and open loop unstable plant, a non-linear plant, and a car parking mechanism. Each of the experiments used different GA objective functions or performance indices. The performance index (F) is related to fitness (f) using the following relationship :

$$f=A/(1+F)^g. \quad (3.3)$$

where f is the fitness for the parameter set, F is the performance index, and g is the constant that affects the performance curve. A is a non-negative constant and appropriately chosen so that f will not be too small, i.e., becomes insignificant due to the large value of F . The experiments are simulated on a 133MHz Pentium IBM-Compatible Personal Computer with 16 MB of RAM. The GA software is mainly based on the GENESIS Version 5.0 package with some improved features as described. It runs under Borland C++ Version 3.1 environment.

4. SIMULATION RESULTS

4.1 Application To An Unstable Plant

In this application, consider a non-minimum phase plant having an open loop unstable pole with the following transfer function :

$$G_h(s) = \frac{(-0.67s^2 + 5.52s - 9.437)10}{(s - 0.559)(s^2 + 27.388s + 12.6244)} \quad (4.1)$$

The transfer function is discretized with a sampling period of 0.01 second. The discrete transfer function resulted in having two non-minimum zeros and one unstable pole.

The choice of the fitness function or performance index is dependent on the type of response that are desired for the particular plant. Since the central objective of the control system is to minimize the error between the actual plant response and the set-point, the performance index, F is chosen as follows:

$$F = \sum_{i=1}^L \left[\sum_{k=1}^{N_i} \{e^2(k).k^4\} \right] \quad (4.2)$$

where $e(k)$ is the system's error at the k^{th} sampling instant. The error is taken as the difference between the setpoint or reference signal and the actual system response.

All of the parameters of the controller were normalized in the range of -1 to +1. Furthermore, the controller output is limited within -5 volt to +5 volt. At the beginning of the GA iterations, the overall fitness is improved drastically from one generation to another. Figure 6(a) shows the membership functions of the input fuzzy variables tuned by the GA after

convergence. The weights tuned by the GA are given in the table in Fig.6(b). It performed well as shown in Fig.7.

The performance of the NFLC is compared to a positional-type PID controller. As the plant is rather complex, initially, the Ziegler-Nichols tuning method [30] is used, and this is followed by manual fine-tuning. However, it is still difficult to obtain satisfactory performance. In order to improve the performance, GA is used to tune the PID controller, which is performed under the same control environment as applied in the GA tuning of NFLC. Furthermore, the performance index in Eq(4.2) is also used in tuning the PID controller. The three discrete PID parameters, proportional gain K_p , integration gain K_i and derivative gain K_d given by GA are 1.2, 0.0013 and 4.1 respectively, in which its response is shown in Fig.8. Comparing both of these results, even though the same performance index is used for both cases, it can be observed that the NFLC has shorter settling time and lower overshoot when compared to the GA tuned PID controller.

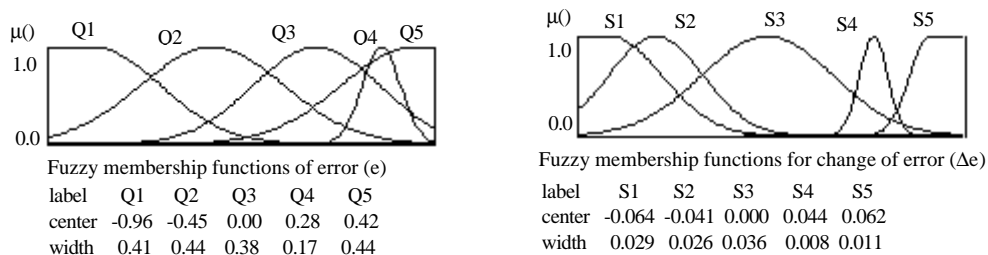


Fig.6(a) The fuzzy inputs membership functions of the NFLC tuned by GA for the unstable plant.

	Q1	Q2	Q3	Q4	Q5
S1	-0.019	0.660	-0.043	-0.036	-0.379
S2	0.812	-0.687	-0.415	-0.344	0.345
S3	-0.015	0.042	0.000	0.382	0.269
S4	-0.360	0.121	-0.342	-0.912	-0.650
S5	0.437	0.261	-0.256	0.255	0.173

Fig. 6(b) The values of the NFLC's weights generated by GA for the unstable plant.

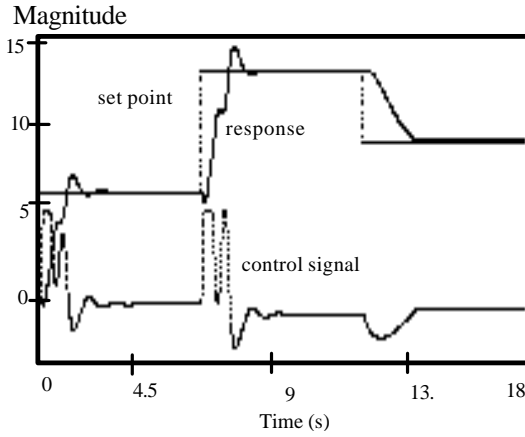


Fig.7 Response of the open-loop unstable with non-minimum phase plant using the GA tuned NFLC.

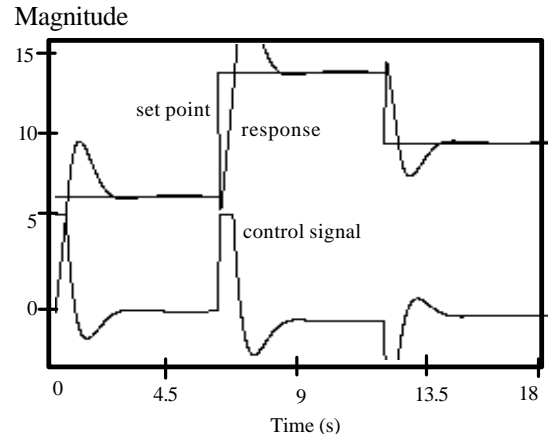


Fig.8 Response of the open-loop unstable with non-minimum phase plant using the GA tuned PID controller.

4.2 Application to a Non-linear Plant

Due to their nonlinear properties, fuzzy controllers are known to be capable of regulating non-linear processes, however, it is rather difficult to construct and tune the fuzzy membership functions and rules for such purpose even by using expert knowledge. In this experiment, the NFLC is configured to control a non-linear plant which is given by the following discrete-time plant dynamics :

$$y(k) = 0.79y(k-1) + 0.012y(k-1)y(k-2) - 0.005y^2(k-2) + 0.15u(k) - 0.8u(k-1) \quad (4.3)$$

The nonlinearity and non-minimum phase dynamics of the plant make control rather difficult and challenging. The effect of the non-linear behavior of the plant is less if the magnitude of the reference signal is small, however, in our experiments, the reference signal is set to a large value (near to a magnitude of 25), which significantly increases the non-linearity of the plant. The control signal of the system is limited between -5v to +5v.

In this experiment, the performance index, F is chosen differently from the previous experiments which is given as follows :

$$F = \sum_{i=1}^L \left[s_i \sum_{k=1}^{N_i} \{ e^2(k).k^4 \} \right] \quad (4.4)$$

where S is the number of oscillations encountered during the tracking process. This is due to the fact that upon initial investigation, the response of this nonlinear plant is rather oscillatory. This situation occurred at the beginning of the GA iterations where the NFLC was still badly tuned. It can be observed that by adding such a term in the performance index, the parameters of the

controller were tuned in such a way to suppress these oscillations, where gradually a smoother output response was achieved through the GA generations.

After convergence of the GA, the resulted NFLC membership functions parameters and the corresponding weights are shown in Fig.9(a) and Fig.9(b) respectively. Figure 11 shows the tracking ability by the controller in following a number of setpoints. It can be observed that the NFLC performed well in controlling the system for a wide range of setpoints. Experiments to control the same system using a classical PID controller appeared futile. Thus, a conventional manually-tuned FLC is setup for comparison purposes. The conventional FLC used the MIN-MAX inference mechanism and center of gravity defuzzification strategy [2,3]. Figure 10(a)-(c) and Fig.12 show the normalized memberships and fuzzy control rules, and the response of the conventional FLC in controlling the same plant, respectively.

Comparing the two controllers, the conventional FLC gives better transient response, i.e. faster rise time and lower overshoot, at the lower range of setpoint as indicated in Fig.12. However, when the setpoint is changed to a higher value, where the nonlinear characteristic of the plant became more significant, the response of conventional FLC deteriorated and became oscillatory. The oscillation of system response can be reduced by re-tuning the input and output scaling factors. However, when the sensitivity of the controller is reduced, the system transient response was quite badly affected. On the other hand, the NFLC resulted in moderate transient responses within the operating points. It can be observed that GA tuned NFLC in such a way to compensate between the fast transient response and nonlinear oscillatory effects over a wide range of operating points based on the performance index defined.

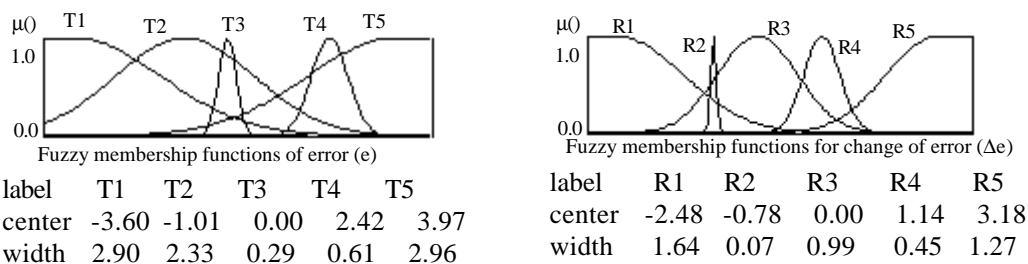


Fig.9(a) The fuzzy inputs membership functions of the NFLC tuned by GA for the nonlinear plant.

	T1	T2	T3	T4	T5
R1	-1.36	-1.26	-1.64	2.35	-2.39
R2	-1.18	-1.91	-0.23	-1.69	-2.14
R3	-0.36	-0.17	0.00	0.28	2.37
R4	2.40	-1.96	-0.53	0.49	1.98
R5	1.91	2.50	-0.50	2.35	2.65

Fig. 9(b) The weights of the NFLC generated by GA for the nonlinear plant.

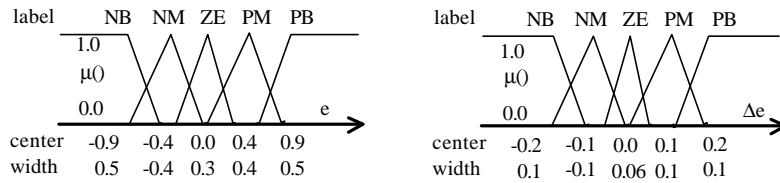


Fig.10(a) Fuzzy inputs memberships of the conventional FLC.

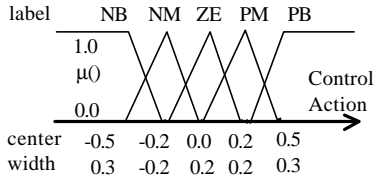


Fig.10(b) Fuzzy output memberships of the conventional FLC.

e	NB	NM	ZE	PM	PB
Δe					
PB	ZE	ZE	PM	PB	PB
PM	NB	ZE	PM	PB	PB
ZE	NB	NM	ZE	PM	PB
NM	NB	NM	NM	ZE	PB
NB	NB	NB	NM	ZE	PM

Fig.10(c) Fuzzy control rules table of the conventional FLC.

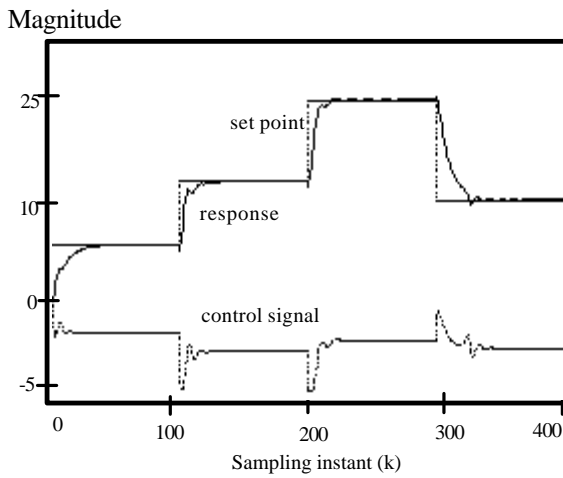


Fig.11 Response of the non-linear system using the NFLC tuned by GA.

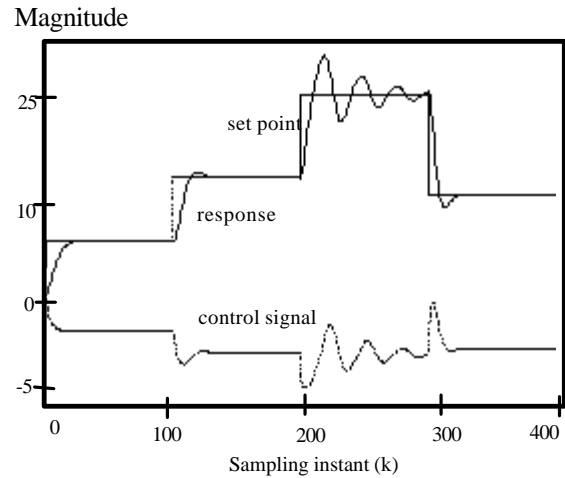
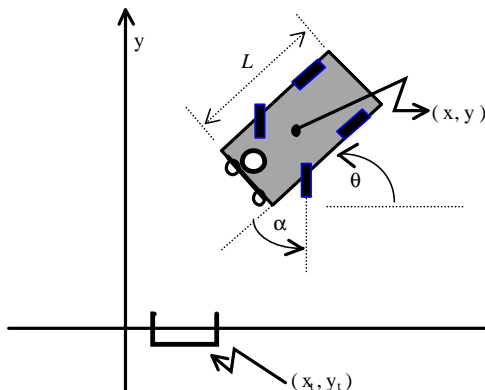


Fig.12 Response of the non-linear system using the conventional FLC.

4.3 Application to an Automatic Car Parking System

The next application describes an experiment to automate a car parking system using the proposed methodology, where the controller is used to decide the steering angle of the car in the parking process. A car model in Cartesian coordinates are shown as in Fig.13. The car movement dynamics which consist of non-linear characteristics [31] can be described as follows:



$$\begin{aligned} \theta(k+1) &= \theta(k) - v.T.\tan(\alpha)/L \\ y(k+1) &= y(k) - S.v.T.\sin(\theta_{k+1}) \\ x(k+1) &= x(k) - S.v.T.\cos(\theta_{k+1}) \end{aligned}$$

Fig.13 The car parking mechanism and its system dynamics.

where the car length, $L = 2.6\text{m}$, the constant velocity of the car, $v = 0.5\text{m/s}$; and sampling period, $T=1\text{s}$; the Cartesian parking space is defined as $-75 < (x, y) < 75\text{m}$; car angle θ , with $-\pi$

$\leq \theta \leq \pi$; steering angle α , with $-\pi/3 \leq \alpha \leq \pi/3$; with $S = 1$ for forward movement and $S = -1$ for backward movement. The parking trials are performed in a normalized parking space.

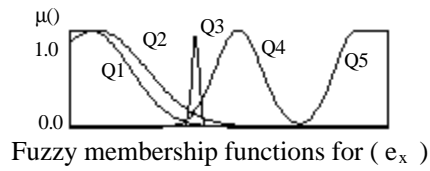
The position of the car on the plane is indicated by an (x,y) coordinate system. The goal of the experiment is to park the car at a specified parking lot (x_t, y_t) with desired car angle θ_t . The NFLC is configured to accept two inputs, i.e. the error of x-position, e_x , and the car angle, θ , and to produce the steering angle, α as the controller output. Each of the fuzzy input variables, e_x and θ , has five fuzzy membership functions in their respective universes of discourse. The centers and widths of all the fuzzy membership functions are determined by GA. In these experiments, enough clearance between the car and the target parking lot is assumed, thus, the constraint of the y-position coordinate can be ignored.

In the GA evaluation routine, a set of controller parameters is evaluated by starting the car from several initial positions, (x_o, y_o) with the initial car angle, θ_o . In this experiment, each set of the parameters went through 12 parking trials from 12 set of initial states of the car. Each of the parking trials took 300 sampling instants. The performance index of the evaluation criteria was formulated as :

$$F = \sum_{i=1}^L \left[\sum_{k=1}^{N_i} \left(\left[e_x^2(k) + 1 \right] \cdot \left[e_\theta^2(k) + 1 \right] \cdot k \right) \right] \quad (4.5)$$

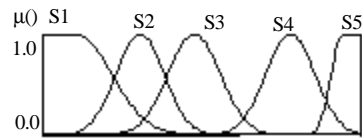
where $e_x(k)$ is the error of x-position and $e_\theta(k)$ is the error of car angle at the k^{th} sampling instant, N_i is the total number of iterations for the i^{th} trial ($N_i=300$), L is the total number of tests carried out ($L=12$). Throughout the GA iterative learning process, the target parking lot (x_t, y_t) was set at (0,0) with target backward parking angle of θ_t equals $-\pi/2$. In all of these experiments, only backing up of the car, i.e. backward movements is considered.

Figure 14(a) shows the Gaussian fuzzy input membership functions for error of x-position (e_x) and the car angle (θ) constructed by the GA. The tuned weights which connect the fuzzy basis units (fuzzy rules) at hidden layer to the output layer is given as Fig.14(b). Figures 15(a)-(d) show some parking capabilities of the NFLC optimized by GA. In all the parking trials, the car parked precisely into the designated parking lot. The results of this proposed NFLC in parking the same car are more superior than that shown in [31].



Fuzzy membership functions for (e_x)

label	Q1	Q2	Q3	Q4	Q5
center	-0.68	-0.64	0.00	0.28	1.00
width	0.36	0.45	0.03	0.21	0.20



Fuzzy membership functions for (θ)

label	S1	S2	S3	S4	S5
center	-0.78	-0.50	0.25	0.20	0.44
width	0.23	0.16	0.17	0.16	0.07

Fig.14(a) The fuzzy inputs membership functions of the NFLC constructed by GA for car parking simulation.

	Q1	Q2	Q3	Q4	Q5
S1	-1.15	-1.15	-1.18	-1.07	-1.10
S2	0.08	0.81	0.86	0.75	-1.03
S3	-0.41	0.48	0.94	1.10	1.08
S4	-1.16	-0.69	0.00	1.11	1.09
S5	-1.17	-1.16	-0.27	0.03	0.36

Fig. 14(b) The weights of the NFLC generated by GA for car parking simulation.

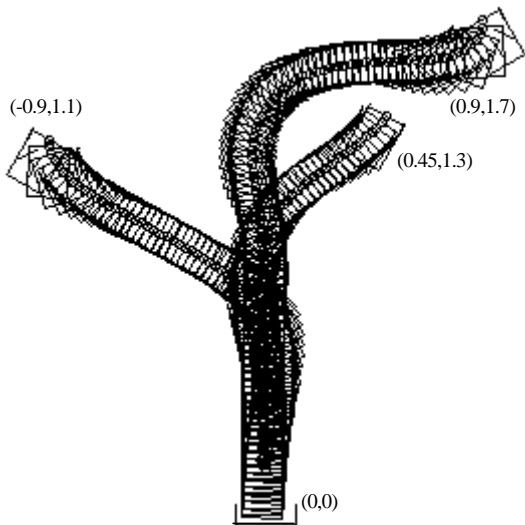


Fig.15(a) Parking trajectories for cases of initial car body angle, $-\pi \leq \theta_0 \leq 0$.

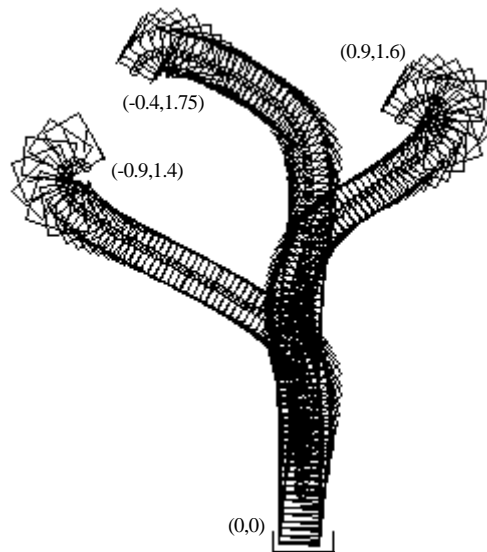


Fig.15(b) Parking trajectories for cases of initial car angle of $0 \leq \theta_0 \leq \pi$.

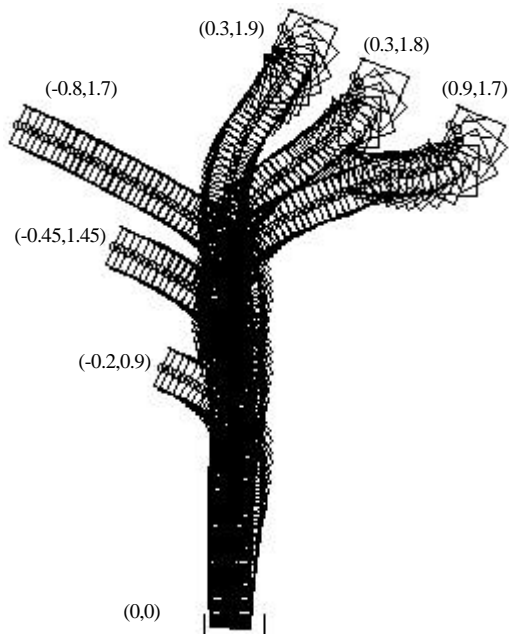


Fig.15(c) Parking trajectories for cases of fixed initial car angle of $\theta_0 = \pi/4$.

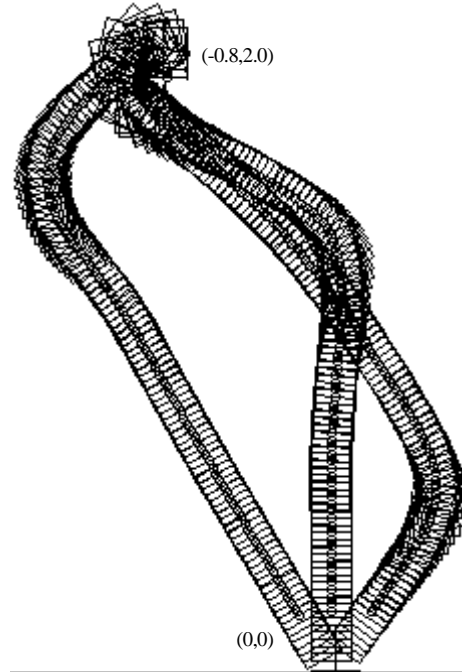


Fig.15(d) Parking trajectories from initial states $(x_0 = -0.8, y_0 = 2.0, \theta_0 = \pi/2)$ to target position $(x_t = 0.0, y_t = 0)$ with parking angle $(\theta_0 = \pi/2, \theta_t = \pi/2)$ respectively.

5. CONCLUSION

This paper has presented a neuro-fuzzy controller where all its parameters can be simultaneously tuned by GA. The controller is based on the Gaussian type RBF neural network. By appropriate coding of the NFLC parameters, it can achieve self-tuning properties from an initial random state. By employing dynamic crossover and mutation probability rates, the tuning process by GA was further improved. The proposed NFLC tuned by GA has also been tested on three different systems, i.e., an unstable and non-minimum phase plant, a non-linear plant, and in an automated car parking system. In the experiments, the control performance has been compared to a GA tuned PID controller and also the conventional FLC. It was observed that the applied NFLC performed relatively better than the other two controllers. Compared to the conventional FLC, the NFLC can somewhat eliminate laborious design steps such as manual tuning of the scaling factors and membership functions, and also selection of the fuzzy rules. Though it can be argued that in the proposed NFLC, before GA can be used to optimize its parameters, initial encoding and settings are required, however, such procedures are somewhat relatively simpler, and more systematic than heuristic.

In addition, this paper also shows the flexibility of the proposed methodology in applying the different types of performance indexes for various types of control systems. Each of the performance indexes used was rather straightforward and simple, yet resulted in satisfactory system responses. An on-line learning algorithm utilizing the GA approach and a

General Regression Neural Network which learns the plant characteristic is currently being developed to provide an on-line self-tuning of the NFLC.

References

1. Zadeh L.A., Fuzzy Sets, *Information and Control*, Vol.8, pp338-353, June 1965.
2. Mamdani,E.H. and Assilian,S., An experiment in linguistic synthesis with a fuzzy logic controller, *Int. J. Man Mach. Studies*, Vol. 7, No. 1, pp1-13, 1975.
3. King,P.J. and Mamdani,E.H., The Application of Fuzzy Control System to Industrial Process., *Automatica*, Vol.13., pp235-242, 1977.
4. Qin,S.H. and Borders,G., A Multiregion Fuzzy Logic Controller of Nonlinear Process Control, *IEEE Transactions on Fuzzy Systems*,Vol 2, No 1, pp74-81, Feb.1994.
5. Wegmann,H., Fuzzy Control and Neural Networks Industrial Applicaitons in the World of PLCs. *Proc. of the Third IEEE Conf. on Control Applications*, Glasgow, UK, 24-28 Aug.1994, Vol.2, pp1245-1249.
6. Chiu S. and Chand. S., Self-organizing Traffic Control via Fuzzy Logic, *Proceedings of the 32nd Conference on Decision and Control*, Dec.1993, pp1897-1902.
7. Oshima,H., Yasunobu,S. and Sekino,S., Aotomatic Train Operation System Based on Predictive Fuzzy Control, *International Workshop on Artificial Intelligence for Industrial Applications 1988*. pp485-489.
8. Berenji,H.R., Learning and Tuning Fuzzy Logic Controllers Through Reinforcements, *IEEE Transactions on Neural Networks*, Vol.3, No.5, pp724-740, Sept 1992.
9. Jang, J.R., Self-learning Fuzzy Controllers Based on Temporal Back Propagation, *IEEE Transaction on Neural Networks*, Vol.3, No.5, pp714-721,Sept 1992.
10. Lee,T.H., Nie,J.H. and Tan,W.K., A self-organizing fuzzified basis function network control system applicable to nonlinear servo mechanisms, *Mechatronics* Vol.5, No.6, pp695-713, 1995.
11. Karr,C.L. and Gentry,E.J. Fuzzy Control of pH Using Genetic Algorithms, *IEEE Trans.on Fuzzy Systems*, Vol.1, No.1. pp46-53, Feb.1993
12. Karr,C.L., Weck,B., Massart,D.L. and Vankeerberghen,P. Least Median Squares Curve Fitting Using a Genetic Algorithm, *Engng.Applic.Artif.Intell.* Vol.8.No.2, pp177-189,1995.
13. Kim,J., Moon,Y. and Zeigler,B.P., Designing Fuzzy Net Controllers Using Genetic Algorithms. *IEEE Control System*, Vol.15, No.3, pp66-72.
14. Varsek,A., Urbancic,T. and Filipic,B., Genetic Algorithms in Controller Design and Tuning, *IEEE Transaction on Sys., Man and Cybernetics*, Vol.23, No.5, Sept./Oct. 1993.
15. Hwang,W.R. and Thompson,W.E., An Intelligent Controller Design Based on Genetic Algorithm. *Proc. of the 32nd IEEE Conf. on Decision & Control*, 15-17 Dec. 1993, USA, pp1266-1267.
16. Hu,H.T., Tai,H.M. and Shenoi,S. Incorporating Cell Map Information in Fuzzy Controller Design, *Proc. of the 3rd IEEE Conf. on Fuzzy Systems*, USA, 26-29 June 1994, pp394-399.
17. Homaifar,A. and McCormick,E.. Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms. *IEEE Transaction on Fuzzy Systems*.Vol.3, No.2, pp129-138, May 1995.
18. Lee,M.A. and Takagi,H., Integrated design stages of fuzzy systems using genetic algorithms, *Proc. of FUZZ-IEEE'93* , San Francisco, USA., 1993, pp612-617.
19. Shimojima, K., Fukuda,T. and Hasegawa,Y., Self-tuning fuzzyt modeling with adaptive membership function, rules, and hierarchical structure based on genetic algorithm, *Fuzzy Sets and Systems 71*, pp295-309,1995.
20. Linkens,D.A. and Nie,J. A Unified Real Time Approximate Reasoning Approach for Use in Intelligent Control. Part I: theoretical development. *Int. J. Control*, Vol.56, pp347-364,1992.

21. Katayama,R., Kajitani,Y., Kuwata,K. and Nishida,Y., Self Generating Radial Basis Function as Neuro-Fuzzy Model and its application to Nonlinear Prediction of Chaotic Time Series. *Proc. of 2nd IEEE Int. Conf. on Fuzzy Systems*, USA, 28 March - 1 Apr. 1993, pp407-414.
22. Tao,M.K. A Closer Look At The Radial Basis Function(RBF) Networks. *Conf. Record of the 27th Conf. on Signals, Systems and Computers*, USA, 1-3 Nov.1993, pp401-405.
23. Sheble,G.B and Britting,K., Refined Genetic Algorithm-Economic Dispatch Example, *IEEE Trans. on Power systems*, 10(1), pp117-124, Feb.1995.
24. Sheble,G.B. and Maifeld,T.T., Unit commitment by genetic algorithm and expert system, *Electric Power Systems Research* 30 pp115-121, 1994.
25. Linkens,D.A. and Nie,J. Fuzzified RBF Network-based Learning Control: Structure and Self-construction. *Proc. of 1993 IEEE Int. Conf. on Neural Networks*, USA, 28 March-1 Apr. 1993, pp1016-1021.
26. Moody,J. and Darken,C.. Fast-learning in Networks of Locally-tuned Processing Units. *Neural Comput.*, Vol 1, pp281-294,1994.
27. Leonard,J.A. and Kramer,M.A., Radial Basis Function Networks for Classifying Process Faults, *IEEE Control System*, pp31-38, Apr.1991.
28. Davis, L., *Handbook of Genetic Algorithms*. NY:Van Nostrand Reinhold, 1991.
29. Goldberg, D.E.. *Genetic Algorithms in Search, Optimization and Machine Learning*. US:Addison-Wesley, 1989.
30. Ogata,K. *Discrete-Time Control Systems*. Prentice-Hall Inc. NJ 1987.
31. Tanaka,K. and Sano,M., Trajectory Stabilization of a Model Car Via Fuzzy Control, *Fuzzy Sets and Systems*, Vol.70, pp155-170, 1995.
32. Driankov,D., Hellendoorn,H. and Reinfrank, *An Introduction to Fuzzy Control*, 2nd-revised Ed., Springer-Verlag, Berlin-Heidelberg, 1996.
33. Lee,C.C., Fuzzy logic in Control Systems: Fuzzy Logic Controller - I & II, *IEEE Trans. on Sys., Man, and Cybernetics*, Vol.20, No.2, . pp404-435, 1990.

Figure Captions

- Fig. 1 The architecture of the NFLC based on the RBF neural network.
- Fig. 2 The fuzzy basis function at the hidden layer.
- Fig. 3 Computation of the output value of the NFLC.
- Fig. 4 A functional block diagram showing the GA optimization process.
- Fig. 5 The dynamic crossover and mutation probability rates.
- Fig. 6(a) The fuzzy inputs membership functions of the NFLC tuned by GA for the unstable plant.
- Fig. 6(b) The values of the NFLC's weights generated by GA for the unstable plant.
- Fig. 7 Response of the open-loop unstable with non-minimum phase plant using the GA tuned NFLC.
- Fig. 8 Response of the open-loop unstable with non-minimum phase plant using the GA tuned PID controller.
- Fig.9(a) The fuzzy inputs membership functions of the NFLC tuned by GA for the non-linear plant.
- Fig. 9(b) The weights of the NFLC generated by GA for the non-linear plant.
- Fig.10(a) Fuzzy inputs memberships of the conventional FLC.
- Fig.10(b) Fuzzy output memberships of the conventional FLC.
- Fig.10(c) Fuzzy control rules table of the conventional FLC.
- Fig.11 Response of the non-linear system using the NFLC tuned by GA.
- Fig.12 Response of the non-linear system using the conventional FLC.
- Fig.13 The car parking mechanism and its system dynamics.
- Fig.14(a) The fuzzy inputs membership functions of the NFLC constructed by GA for car parking simulation.
- Fig. 14(b) The weights of the NFLC generated by GA for car parking simulation.
- Fig.15(a) Parking trajectories for cases of initial car body angle, $-\pi \leq \theta_0 \leq 0$.
- Fig.15(b) Parking trajectories for cases of initial car angle of $0 \leq \theta_0 \leq \pi$.
- Fig.15(c) Parking trajectories for cases of fixed initial car angle of $\theta_0 = \pi/4$.
- Fig.15(d) Parking trajectories from initial states $(x_0 = -0.8, y_0 = 2.0, \theta_0 = \pi/2)$ to target position $(x_t = 0.0, y_t = 0)$ with parking angle $(\theta_0 = \pi/2, \theta_0 = \pi/2, \theta_0 = \pi/2)$ respectively.