On GA-based Optimal Fuzzy Control

Sinn-Cheng Lin, *Student Member IEEE* and Yung-Yaw Chen, *Member IEEE*

Lab. 202, Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
email: sclin@ipmc.ee.ntu.edu.tw

Abstract

Two architectures for designing optimal fuzzy control systems were proposed in this paper. In both cases, the membership functions in the fuzzy rule-bases were tuned by the genetic algorithms. The objective was to explore a fuzzy controller by minimizing a quadratic cost function. In the first architecture, the employed controller was a conventional *fuzzy logic controller* which used the system states as input variables. Consequently, the reciprocal of the cost function to be minimized could be directly applied towards evaluating the fitness of the controller. In the second architecture, a *fuzzy sliding mode controller* was adopted. The combined information of the system states, *i.e.* the sliding function, formed a single input variable. The problem of minimizing the cost function in this case could be transformed to that of deriving an optimal sliding surface. Then, a faster hitting time and a smaller distance away from the sliding surface a controller had, a higher fitness it got. Simulations and comparisons were taken on both cases.

Keywords: fuzzy control, optimal control, fuzzy sliding mode control, genetic algorithms

I. Problem Formulation

Consider a class of nth order nonlinear systems that can be expressed by the following state equation [5]:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \quad\vdots \\ \dot{x}_n = f(x) + g(x)u \end{cases} \tag{1}$$

where $x = [x_1\ x_2\ \ldots\ x_n]^{\mathrm{T}} \in \mathbb{R}^n$ is the state vector; $u \in \mathbb{R}$ is the control input; $f(.)$ is an unknown continuous function with known upper bound, *i.e.* $|f| \le f^U$; $g(.)$ is an unknown positive definite function with known lower bound, *i.e.* $0 < g_L \le g$. Actually, (1) represents a general uncertain nonlinear dynamical system. Define a quadratic cost function [3]:

$$J = \frac{1}{2}\int_0^\infty x(t)^{\mathrm{T}} Q x(t) dt \tag{2}$$

where $Q \in \mathbb{R}^{n\times n}$ is a positive definite weighting matrix. The objective of this paper can be described as follows:

Given a system of (1), only knowing the upper bound of f(.) and the lower bound of g(.), attempt to develop a fuzzy control system such that (2) is minimized.

Genetic algorithms (GAs) are parallel and global search techniques which take the concepts from evolution theory and natural genetics. They emulate biological evolutions by means of genetic operations such as *reproduction*, *crossover* and *mutation*. Usually, GAs are used as optimization tools [2]. In this paper, GAs are used for searching the parameter space of fuzzy membership functions and for finding suitable fuzzy controllers to minimize J. Two architectures of fuzzy control for optimization are proposed:

1) GA-based optimal fuzzy logic control: Fig. 1 shows the block diagram of such a control system. In this architecture, the employed fuzzy controller is a traditional fuzzy logic controller (FLC) [1]. Consequently, the input of the GA-based FLC (GA-FLC) is the state vector x, so that the cost function (2) can be directly calculated and converted to the fitness value for genetic operations.

2) GA-based optimal fuzzy sliding mode control: Fig. 2 shows the basic configuration of such a control system. In which, the employed fuzzy controller is a fuzzy sliding mode controller (FSMC) [6]. There is a mapping mechanism which maps the state vector, x, to a sliding function, s. As a result, the input variable of the GA-based FSMC (GA-FSMC) is s instead of x . In the other word, the fitness function, or equivalently the cost function $J(x)$, for genetic operations can't be directly obtained. The primary job remained is to transform the problem of minimizing the original cost function $J(x)$ to that of minimizing an alterative cost function J_s which is a function of s, *i.e.* $J_s = J_s(s)$.

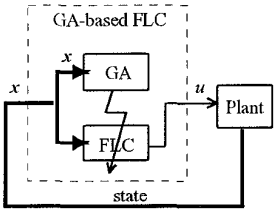

Fig. 1. The architecture of GA-based FLC

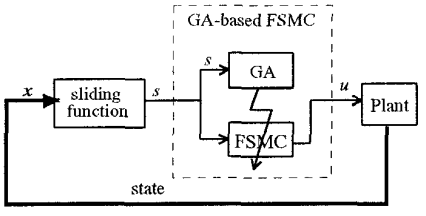

Fig. 2. The architecture of GA-based FSMC

II. Genetic Algorithms: an overview

GAs work with a set of artificial elements (parameter strings) called a population. An individual (string) in a population is referred as a chromosome, and a single element in a chromosome is called a gene. GAs generate a new population (called offsprings) by applying the genetic operators to the chromosomes in the old population (called parents). Each iteration of genetic operations are referred as a generation. A fitness function, *i.e.* the function to be maximized, is used to evaluate the fitness of an individual. One of the important purpose of GAs is to reserve the better schemata, *i.e.* the patterns of certain genes, so that the offsprings may yield higher fitness than their parents. Consequently, the fitness value increases from generation to generation. In most of GAs, *reproduction*, *crossover* and *mutation* are three basic operators. Actually, reproduction and crossover don't introduce new patterns of gene into the population, but mutation does. Mutation can be viewed as a random-work mechanism to avoid the local optimum trapping problem. As a result, GAs could always find a sub-optimal solution that approximate the global one.

Although there are a large amount of genetic algorithms had been proposed, the fundamental principle were based on the simple genetic algorithm (SGA). In general, the individual strings that the SGA works on are binary-coded (*e.g.* 01101110); hence, SGA also known as binary-coded GA. The basic operations of SGA are briefly described as follows [2]:

1) *Reproduction:* the Darwinian "survival of the fittest" is the underlying spirit of reproduction. First, a fitness value F is assigned to each individual string in a population. A higher F value indicates a better fit (or larger benefit). Next, the old individual strings are probabilistically selected and copied into a mating pool according to their fitness value. The arrangement allows the strings with a higher fitness to have a greater probability of contributing a larger amount of offsprings in the new population.

2) *Crossover:* crossover provides a mechanism for individual strings to exchange information via a probabilistic process. Once the reproduction operator is applied, the members in the mating pool are allowed to mate with one another. First, two parents are randomly selected from the mating pool. Next, a random crossover point is picked up, on which the parents will exchange their genes. Finally, the parents' genetic codes are mixed by exchanging their codes following the crossover point. For example, let a, b denote two parent strings, and a', b' be their children. If the crossover point is selected on the 3rd bit, then we have:

$$a = 10101\underline{010} \qquad a' = 10101\underline{100}$$
$$\Longrightarrow$$
$$b = 01111\underline{100} \qquad b' = 01111\underline{010}$$

This random process provides a highly efficient method to search the string space for finding a better solution.

3) *Mutation:* every gene is subject to a random change with probability of the pre-assigned mutation rate in each iteration. In the SGA, a mutation operator is nothing but just changes a random-selected bit from 0 to 1 or vice versa.

III. GA-based Optimal Fuzzy Logic Control

In general, each rule in the rule-based of an FLC can be represented as:

$$R^{(j)}: \textit{IF } x_1 \textit{ is } X_1^{(j)}(m_1^{(j)}, \sigma_1^{(j)}) \textit{ and } x_2 \textit{ is } X_2^{(j)}(m_2^{(j)}, \sigma_2^{(j)}) \textit{ and } \ldots x_n \textit{ is } X_n^{(j)}(m_n^{(j)}, \sigma_n^{(j)}) \textit{ THEN } u \textit{ is } U^{(j)}(\varphi^{(j)}) \quad (3)$$

where $j = 1, 2, \ldots, N_l$ and N_l is the number of rules; $X_i^{(j)}$ are fuzzy sets called the input linguistic labels; particularly, they are all Gaussian-typed in this paper, *i.e.* $\mu_{X_i^{(j)}}(x_i) = \exp\left[-\left(\frac{x_i - m_i^{(j)}}{\sigma_i^{(j)}}\right)^2\right]$; $U^{(j)}$ are fuzzy sets called the output linguistic labels and $\varphi^{(j)}$ are supports of $U^{(j)}$ on which the membership grade $\mu_{U^{(j)}}(\varphi^{(j)}) = 1$.

Suppose that the singleton fuzzification and the

weighted average defuzzification methods [5] are adopted, then the control output of (3) is given by:

$$u = \frac{\sum_{j=1}^{N_l} \varphi^{(j)} \bigwedge_{i=1}^{n} \exp\left[-\left(\frac{x_i - m_i^{(j)}}{\sigma_i^{(j)}}\right)^2\right]}{\sum_{j=1}^{N_l} \bigwedge_{i=1}^{n} \exp\left[-\left(\frac{x_i - m_i^{(j)}}{\sigma_i^{(j)}}\right)^2\right]} \tag{4}$$

where the *and* operator $\wedge$ can be *min*, *product* or any *T-norm* [1].

Obviously, the control output of FLC is depend on the parameters of membership functions, *i.e.* $m_i^{(j)}$, $\sigma_i^{(j)}$ and $\varphi_i^{(j)}$. Hence, the genetic algorithms are applied towards adjusting the parameter set $\{m_i^{(j)}, \sigma_i^{(j)}, \varphi_i^{(j)} \mid i = 1, 2, ..., n, j = 1, 2, ..., N_l\}$ of (3) so as to minimize $J(x)$.

To synthesis the optimal FLC by GA, each rule can be parameterized as:

$$R^{(j)} = enc(m_1^{(j)}, ..., m_n^{(j)}, \sigma_1^{(j)}, ..., \sigma_n^{(j)}, \varphi_1^{(j)}, ..., \varphi_n^{(j)}) \tag{5}$$

where $enc(.)$ denotes the encoding operator which encodes the real values into the corresponding binary codes. Then, the chromosome of an individual, which represents a particular rule-base of FLC, can be defined as:

$$R = \{R^{(1)}, R^{(2)}, ..., R^{(N_l)}\} \tag{6}$$

Since the state x is directly fed into the mechanism of GA, $J(x)$ can be calculated straightforward. Therefore, the fitness function for GA-FLC can be directly defined as:

$$F = 1/(J + \delta) \tag{7}$$

where δ is a small positive value for avoiding the numerical error of divided by zero.

IV. GA-based Optimal Fuzzy Sliding Mode Control

A. Fuzzy Sliding Mode Controller

In the FSMC [6] design, the sliding surface can be defined as:

$$s(x) = c^{\mathrm{T}} x = 0 \tag{8}$$

where $c = [c_1\ c_2\ ...\ c_n]^{\mathrm{T}} \in \mathbb{R}^n$ is a coefficient vector that has to be properly determined. Each rule in an FSMC is represented as [6]:

$$R^{(j)}: \textit{IF } s \textit{ is } S^{(j)}(m^{(j)}, \sigma^{(j)}) \textit{ THEN } u \textit{ is } U^{(j)}(\varphi^{(j)}) \tag{9}$$

where $j = 1, 2, ..., N_s$, and N_s is the number of rules; $S^{(j)}$ are the input linguistic labels; also, they are Gaussian-typed in this paper, *i.e.* $\mu_{S^{(j)}}(s) = \exp\left[-\left(\frac{s - m^{(j)}}{\sigma^{(j)}}\right)^2\right]$; $U^{(j)}$ are the output linguistic labels and $\mu_{U^{(j)}}(\varphi^{(j)}) = 1$.

Again, the singleton fuzzification and weighted-average defuzzification methods are adopted to obtain the control output of (5), as shown in following:

$$u = \frac{\sum_{j=1}^{N_s} \varphi^{(j)} \exp\left[-\left(\frac{s - m^{(j)}}{\sigma^{(j)}}\right)^2\right]}{\sum_{j=1}^{N_s} \exp\left[-\left(\frac{s - m^{(j)}}{\sigma^{(j)}}\right)^2\right]} \tag{10}$$

Clearly, the genetic algorithms can be applied towards adjusting the parameter set $\{m^{(j)}, \sigma^{(j)}, \varphi^{(j)} \mid j = 1, 2, ..., N_s\}$ to obtain a suitable u such that $J(x)$ is minimized.

Similarly, each rule of FSMC can be parameterized as:

$$R^{(j)} = enc(m^{(j)}, \sigma^{(j)}, \varphi^{(j)}) \tag{11}$$

again, the chromosome of an individual (a particular rule-base of FSMC) can be defined as:

$$R = \{R^{(1)}, R^{(2)}, ..., R\} \tag{12}$$

To define the fitness function of GA-FSMC, let's look at Fig .2. Since the input variable that fed into the GA mechanism is s instead of x, the fitness function can't be calculated from x directly. In the other word, $J(x)$ can't be calculated and converted to the fitness function straightforward. In the following, the optimal sliding surface is derived by minimizing the cost function (2). Correspondingly, the fitness function of GA-FSMC can be defined in the following intuition: a controller which can drive the state to hit the optimal sliding surface as fast as possible and keep the state to the surface as close as possible will get a higher score.

B. Optimal Sliding Surface Design

Consider again the sliding surface defined in (8). Without loss of generality, let $c_n = 1$, *i.e.*

$$s(x) = x_n + \sum_{i=1}^{n-1} c_i x_i \triangleq x_n + \bar{c}^{\mathrm{T}} \bar{x} = 0 \tag{13}$$

where $\bar{c} = [c_1\ c_2\ ...\ c_{n-1}]^{\mathrm{T}}$ and $\bar{x} = [x_1\ x_2\ ...\ x_{n-1}]^{\mathrm{T}}$. Assume that there is a control u^* which can achieve the sliding mode in a finite time, *i.e.* $s = 0$ and $\dot{s} = 0$ as $t \geq t_s$. Then, the original system (1) can be linearlized by u^*, and the equivalent system is given as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \vdots \\ \dot{x}_{n-1} \\ \hline \dot{x}_n \end{bmatrix} = \left[\begin{array}{ccccc|c} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ \hline 0 & -c_1 & -c_2 & \cdots & -c_{n-2} & -c_{n-1} \end{array}\right] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-1} \\ \hline x_n \end{bmatrix}$$

$$\stackrel{\Delta}{=} Ax \tag{14}$$

Rewriting (14) yields:

$$\dot{x} \stackrel{\Delta}{=} \begin{bmatrix} \dot{\varepsilon} \\ \dot{\xi} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \varepsilon \\ \xi \end{bmatrix} \tag{15}$$

in which the state, x, has been partitioned into two parts, $\varepsilon \in \mathbb{R}^{(n-1)\times 1}$, $\xi \in \mathbb{R}^{1\times 1}$, and the system matrix, A, has been divided into four sub-matrix, $A \in \mathbb{R}^{(n-1)\times(n-1)}$, $A_{12} \in \mathbb{R}^{(n-1)\times 1}$, $A_{21} \in \mathbb{R}^{1\times(n-1)}$ and $A \in \mathbb{R}^{1\times 1}$. Obviously, $\varepsilon = x_n$ and $\xi = \bar{x}$. Hence, from (13) we have

$$\xi + \bar{c}^{\mathrm{T}}\varepsilon = 0 \tag{16}$$

Partition the weighting matrix of $J(x)$, *i.e.* Q, in the same way of partitioning A, we get

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \tag{17}$$

Currently, it gets ready to transform the primary optimization problem to a standard LQ one. According to (2), (15), (16) and (17), we have

$$\dot{\varepsilon} = A_0\varepsilon + A_{12}\lambda \tag{18}$$

and

$$J = \frac{1}{2}\int_{t_s}^{\infty}\left(\varepsilon^{\mathrm{T}}Q_0\varepsilon + \lambda^{\mathrm{T}}Q_{22}\lambda\right)dt \tag{19}$$

where

$$\begin{cases} A_0 = A_{11} - A_{12}Q_{22}^{-1}Q_{21} \\ Q_0 = Q_{11} - Q_{12}Q_{22}^{-1}Q_{21} \\ \lambda = Q_{22}^{-1}Q_{21}\varepsilon + \xi \end{cases} \tag{20}$$

Obviously, (18) and (19) form a standard LQR system [3] with a pseudo state ε and a pseudo control λ. Therefore, the optimal control of (18) for minimizing (19), λ^*, can be solved by the famous LQR technique [3].

$$\lambda^* = -\psi^{*\mathrm{T}}\varepsilon \tag{21}$$

with the optimal gain

$$\psi^* = \left(Q_{22}^{-1}A_{12}P_0\right)^{\mathrm{T}} \tag{22}$$

in which P_0 is the solution of the following Riccati equation:

$$A_0^{\mathrm{T}}P_0 + P_0A_0 - P_0A_{12}Q_{22}^{-1}A_{12}^{\mathrm{T}}P_0 + Q_0 = 0 \tag{23}$$

Consequently, the coefficients of optimal sliding surface is given by:

$$\bar{c}^* = \left(\psi^{*\mathrm{T}} + Q_{22}^{-1}Q_{21}\right)^{\mathrm{T}} \tag{24}$$

After deriving the function $s^*(x) = [\bar{c}^{*\mathrm{T}}\ 1]x$, the primary missions the optimal GA-FSMC has to do is to drive the state to hit the optimal sliding surface ($s^* = 0$) as fast as possible and then keep the state to $s^* = 0$ as close as possible is . To achieve such goal, define an alternative cost function

$$J_s(s) = t_s + \int_{t_s}^{\infty}(s^*)^2 dt \tag{25}$$

Similar to (7), the fitness function for the optimal GA-FSMC can be defined as:

$$F = 1/(J_s + \delta) \tag{26}$$

V. Stability Consideration

Since the initial population is random selected, and the GA consists of stochastic processes. Hence, the evolution procedure of both cases described above may yield unstable control systems. In this section, the approaches of guaranteeing the stability of GA-based fuzzy control systems without changing the original structure is proposed. Consider the following control law:

$$u = u_f + \alpha u_s \tag{27}$$

where u_f is a fuzzy control law that given by (4) for GA-FLC and given by (10) for GA-FSMC; u_s denotes a stabilizing control law and α is a switching factor.

To stabilize the GA-FLC, the stabilizing control can be realized by the supervisory control proposed by Wang [5], *i.e.*

$$u_s = sgn\left(x^{\mathrm{T}}Pb\right)\left[g_L^{-1}\left(f^U + \left|k^{\mathrm{T}}x\right|\right) + \left|u_f\right|\right] \tag{28}$$

where $b = [0\ 0\ \ldots\ g]^{\mathrm{T}}$, $k = [k_1\ k_2\ \ldots\ k_n]^{\mathrm{T}}$ and P is a symmetric positive definite matrix satisfying the Lyapunov equation

$$\Lambda^{\mathrm{T}}P + P\Lambda = -Q_s \tag{29}$$

where $Q_s > 0$ and Λ is the companion matrix of Hurwitz polynomial $z^n + k_1z^{n-1} + \ldots + k^n = 0$. The switching factor in this case is selected as

$$\alpha = \begin{cases} 1, & V > \bar{V} \\ 0, & V \le \bar{V} \end{cases} \tag{30}$$

where $V = \frac{1}{2}x^{T}Px$ and $\bar{V}$ is a pre-specified boun V.

On the other hand, the hitting control propc by Lin and Chen [6] can be applied to the GA-FS as the stabilizing control, *i.e.*

$$u_s = -sign(s^*)\left[g_L^{-1}\left(f^U + |\bar{c}^{*T}\bar{x}| + \eta\right) + |u_f|\right] \quad ($$

(31) satisfies the sliding condition $s^*\dot{s}^* < -\eta|s^*|$. switching factor of this case can be selected as

$$\alpha = \begin{cases} 1, & |s^*| > \bar{s} \\ 0, & |s^*| \le \bar{s} \end{cases} \quad ($$

where $\bar{s} > 0$ is a pre-specified bound of s^*.

VI. Simulation and Discussion

Consider an underwater vehicle whose simpli model can be represented as [4]

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{d}{m}x_2|x_2| + \frac{1}{m}u \end{cases}$$

where x_1, x_2 define the position and velocity, resp tively; u is the control force; m is the mass of the hicle, and d denotes the drag coefficient. parameter values, $m = 3 + 1.5\sin(|x_2|t)$, and $d = 1.2 + .2\sin(|x_2|t)$, used in [4] are also adopted in the following simulation. We select the number of rules $N_l = 36$ and $N_s = 6$, the weighting matrix $Q = \begin{bmatrix} 2 & 0.5 \\ 0.5 & 1 \end{bmatrix}$, the population size = 10, the crossover rate = 0.8 and the mutation rate = 0.03.

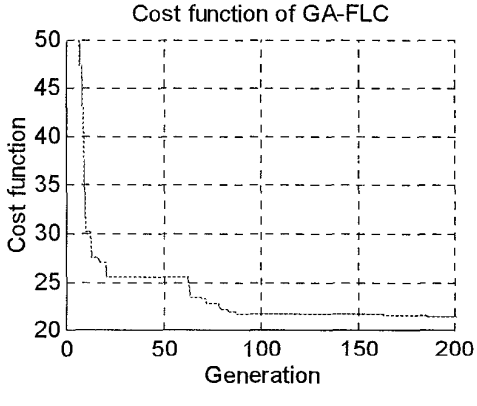

Fig. 3. Cost function of GA-FLC

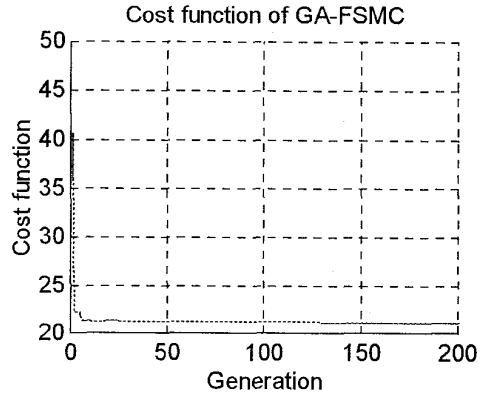

Fig. 4. Cost function of GA-FSMC

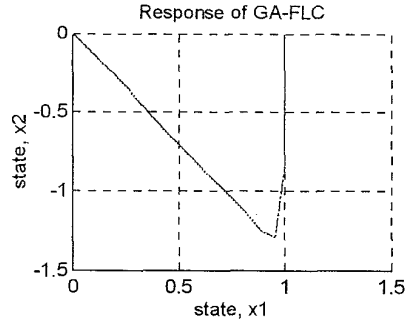

Fig. 5. State response of GA-FLC

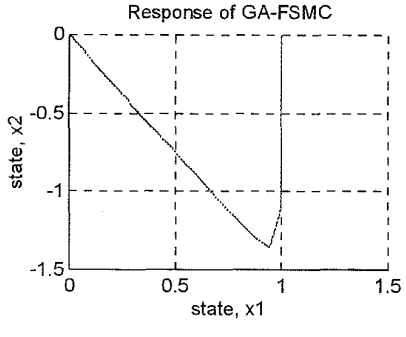

Fig. 6. State response of GA-FSMC

One of the most important advantages of FSMC is that the size of rule-base of FSMC is significantly smaller than that of FLC. As a result, the length of chromosome and the size of search space of GA-FSMC can be efficiently reduced, as we have pointed out in [6]. So, we expect that the convergent rate of GA-FSMC may be faster than that of GA-FLC. Fig. 3 and Fig. 4 show the evolution results of GA-FLC and GA-FSMC, respectively. We can see that the convergent rate agrees with our expectation. Fig. 5 and Fig.

6 show the final responses of GA-FLC and GA-FSMC, respectively. Although the architectures of both cases are so different, the state trajectories of them, however, are almost identical.

VII. Conclusions

In this paper, we developed the GA-based optimization methods for the fuzzy control systems. Two kinds of architectures, GA-FLC and GA-FSMC, are dealt with. The final results of system responses of such two different approaches are similar, as shown in the simulations. In GA-FSMC design, an optimal sliding surface was derived based on the LQ technique.

References

[1] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller, parts I and II," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 2, pp. 404-435, 1990.

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

[3] F. L. Lewis, *Applied Optimal Control and Estimation*, Englewood Cliffs, NJ: Prentice Hall, 1992.

[4] J. J. E. Slotine and W. Li, *Applied Nonlinear Control*, Englewood Cliffs, NJ: Prentice Hall, 1991.

[5] L. X. Wang, *Adaptive Fuzzy Systems and Control*, Englewood Cliffs, NJ: Printice Hall, 1994.

[6] S. C. Lin and Y. Y. Chen, "A GA-based fuzzy controller with sliding mode," *IEEE Int. Conf. on Fuzzy Systems*, Japan, 1995.