Letters

A Method for Design of a Hybrid Neuro-Fuzzy Control System Based on Behavior Modeling

Wei Li

***Abstract*—It is known that control signals from a fuzzy logic controller are determined by a response behavior of a controlled object rather than its analytical models. That implies that the fuzzy controller could yield a similar control result for a set of plants with a similar dynamic behavior. This idea leads to modeling of a plant with unknown structure by defining several types of dynamic behavior, such as "oscillation," "overdamping," "underdamping," and so forth. On the basis of dynamic behavior classification, a new method is presented for design of a neuro-fuzzy control system in two steps: First, we model a plant with unknown structure by choosing a set of simplified systems with equivalent behavior as "templates" to optimize their fuzzy controllers off-line. Second, we use an algorithm for system identification to perceive dynamic behavior and a neural network to adapt fuzzy logic controllers by matching the "templates" on-line. The main advantage of this method is that convergence problem can be avoided during adaptation process. Finally, the proposed method is used to design neuro-fuzzy controllers for a two-link manipulator.**

***Index Terms*—Neuro, controller, loop response.**

I. Introduction

DESIGN of a fuzzy logic controller for a controlled object with unknown structure mainly suffers from both aspects of the controller itself and the controlled object. First, it is very difficult to mathematically analyze the characters of the fuzzy logic controllers and, second, a detailed analytical model of the controlled object is not achieved due to nonlinearity or uncertainty.

One of the widely used design method for fuzzy controllers is to define membership functions of linguistic variables and to formulate fuzzy rules by control engineers [1]–[3]. Unfortunately, there might be no generally applicable principle on determining membership functions and rules for each particular system with specific requirements. Another approach is to adapt rule-base and/or membership functions by self-organizing algorithms or neural network according to previous responses until a desired control performance is achieved [4]–[8]. In particular, an adaptive fuzzy system based on *softmin* operator (proposed by Berenji and Khedkar [6]) is very interesting in control of a plant with uncertainty. However, because the basis of the adaptive strategy is a trial-and-error idea this strategy suffers mainly from its convergence, and the convergence not only depends on the characteristics of a given system, but also on different input references. To get a good convergence during an adaptive process, some systems may be started more times to adjust and test some initial values and learning coefficients. A system might especially be out of control if these coefficients were inadequate.

To deal with the convergence problem, this paper presents a new paradigm for designing a neuro-fuzzy controller based on behavior modeling. It is known that control signals from a fuzzy logic controller are determined by response behavior of a controlled object rather than its analytical models. That means that for two plants with a similar dynamic behavior, a fuzzy logic controller can yield a similar control response although these two plants have different dynamic equations. This leads to the idea of description of a plant with unknown structure by defining several types of dynamic behavior, such as "oscillation," "overdamping," "underdamping," and so forth. On the basis of dynamic behavior classification, we design a neuro-fuzzy system for this plant with unknown structure in following steps. First, we model the plant by choosing a set of simplified systems with equivalent behavior as "templates" to optimize their fuzzy logic controllers off-line. Second, we use an algorithm for system identification to perceive a dynamic behavior and a neural network to adapt fuzzy logic controllers by matching the "templates" on-line. It should be noted that any two plants with a similar dynamic behavior under a given input such as overshot, rise time, and so on do not mean that their analytical models are equivalent; but, if the dynamic equations of two plants are identical, they must have a same dynamic response under a given input. Therefore, description of controlled objectives using dynamic behavior models is more general than that using analytical models.

The differences between the proposed method and traditional adaptation strategies [4]–[8] are addressed as follows: 1) the strategy of the proposed neuro-fuzzy system is to adapt fuzzy controller by matching "templates" instead of the trial-and-error idea, 2) in the trial-and-error idea only current response values, e.g., the k-step error between a reference and an actual value $e(k)$ and its error-in-change $\dot{e}(k)$ are used to adapt the controllers, however, by using the proposed method much more historical information on responses is used to classify dynamic behavior, 3) one of key problems with the trial-and-error idea is convergence. In control based on

Manuscript received May 17, 1995; revised April 15, 1996. This work was supported by the National Natural Science Foundation of China under Grant 69 585 004 and by the Chinese "863" High Technology Project under Grant 863-512-28-20.

The author is with the Department of Computer Science and Technology Tsinghua University, Beijing 100084, P. R. China.

Publisher Item Identifier S 1063-6706(97)00035-0.

behavior modeling, a tuning of a fuzzy logic controller is divided into off-line and on-line stages, and the parameters of the fuzzy logic controller are adapted based on "templates." Therefore, convergence can be greatly improved.

This paper is organized into six sections. In Section II, we discuss general characters of a fuzzy logic controller. In Section III, we propose an approach to behavior modeling, and we use an identification algorithm to perceive dynamic behavior. In Section IV, we present a neuro-fuzzy control system based on behavior modeling, and we optimize the parameters of the fuzzy logic controller using the Nelder and Mead's simplex algorithm [11]. In Section V, we use a neural network on-line to adapt fuzzy controller based on "templates." In Section VI, we use the proposed method to design fuzzy controllers for a two-link manipulator.

II. Characters of a Fuzzy Logic Controller

The principle of designing a fuzzy logic controller is to integrate empirical knowledge and operator experience into controllers by using fuzzy sets and fuzzy rules. To acquire such knowledge, response process of a controlled object must be observed, and its decision and control strategy are expressed by fuzzy logic. Since this design strategy is only depends on response behavior, a fuzzy logic controller can yield a similar control result for a set of plants with a similar dynamic behavior regardless of their mathematical models. We use the following example to demonstrate this character of fuzzy logic controller. First, we choose a second-order system as a controlled object; its dynamic equation is expressed as follows:

$$\ddot{y} + 2.0\omega\xi\dot{y} + \omega^2 = \omega^2 u \tag{1}$$

where $\omega = 1.4$ and $\xi = 0.575$. Then, according to this controlled object we optimize its fuzzy logic controller to achieve a desired control performance. Fig. 1 shows open and closed-loop responses to step change of the controlled object. Now we replace the second-order system with a nonlinear system with its dynamic equation

$$\ddot{y} + 2.0\omega\xi\dot{y}y + \omega^2 y^2 = \omega^2 u \tag{2}$$

where $\omega = 1.0$ and $\xi = 1.0$, i.e., we use the optimized fuzzy controller to control the nonlinear system. Although the parameters of the nonlinear system are different from those of the second-order system, by using the same fuzzy logic controller its closed-loop time response is as good as that of the second-order system (a very small maximum overshoot and a fast settling time), as shown in Fig. 2. This is because that the open-loop dynamic behavior of the nonlinear system is very similar to that of the second-order system. It should be noted that, due to the nonlinearity of (2) its dynamic behavior is quite differ from that of (1) if both of the equations have the same parameters $\omega = 1.0$ and $\xi = 1.0$. When the parameter in (2) changes, its open loop response also changes. Nevertheless, it is possible to find out another second-order system with a similar response to that of this nonlinear system. For example, the dynamic behavior of (2) with $\omega = 1000.0$ and $\xi = 1.0$ is similar to that of a second-order system with $\omega = 1400$ and $\xi = 0.578$. Therefore, we can design a same fuzzy logic

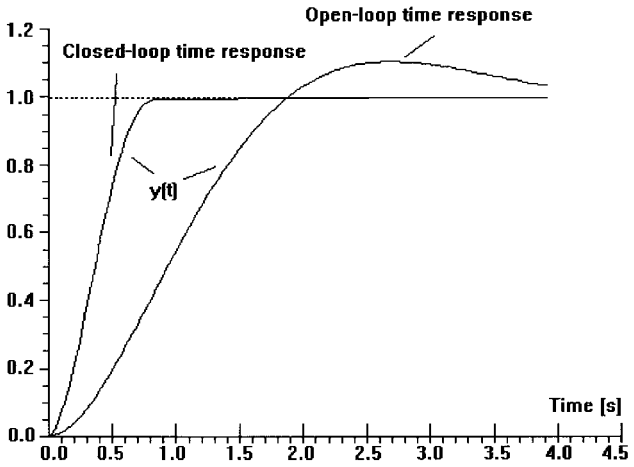


Fig. 1. Open and closed-loop responses of a second-order system.

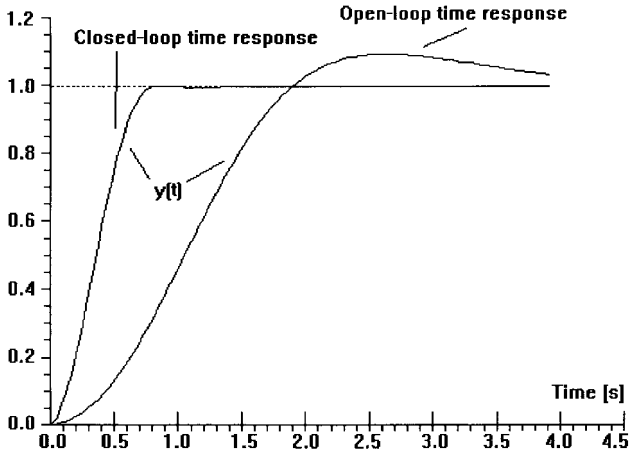


Fig. 2. Open and closed-loop time responses of a nonlinear system.

controller for both of the controlled objects. Another example is to choose a set of higher order system (from third order to sixth-order) with a similar response loop behavior to that of (1). Then using the same fuzzy logic controller, we can get a similar closed-loop response. This leads to idea of modeling of a plant with unknown structure by defining several types of dynamic behavior.

The next well-known character of a fuzzy logic controller is its robustness to parameter changes. When we change the parameters of the second-order system (1) from $\omega = 1.0$ and $\xi = 0.7$ to $\omega = 1.0$ and $\xi = 1.0$, we can observe the change of the open-loop responses of the second-order system, as shown in Fig. 3. At first, we optimize a fuzzy controller related to the second-order system with $\omega = 1.0$ and $\xi = 0.7$ to achieve a good control performance shown in Fig. 4. Then, we use the fuzzy logic controller to control the second-order systems with $\omega = 1.0$ and $\xi = 1.0$. Fig. 4 shows that the performance of closed-loop response is also quite good. For system modeling, this character greatly reduces the types of dynamic behavior to be defined.

III. Behavior Modeling of a Controlled Object

A dynamic behavior of a system is defined as its dynamic response under a given input signal. According to the first

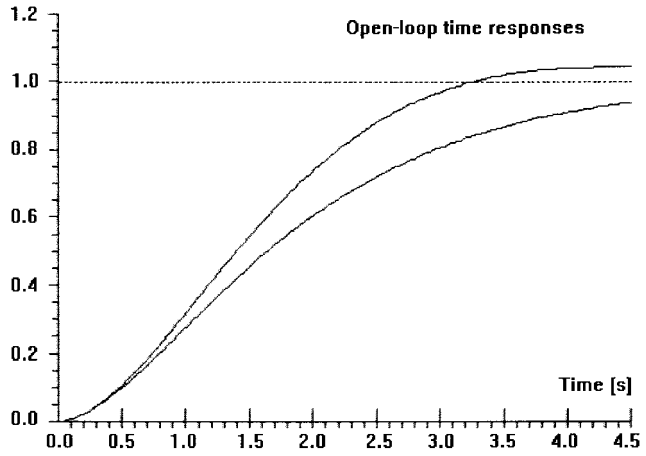


Fig. 3. Open-loop time responses of a second-order system while changing its parameters.

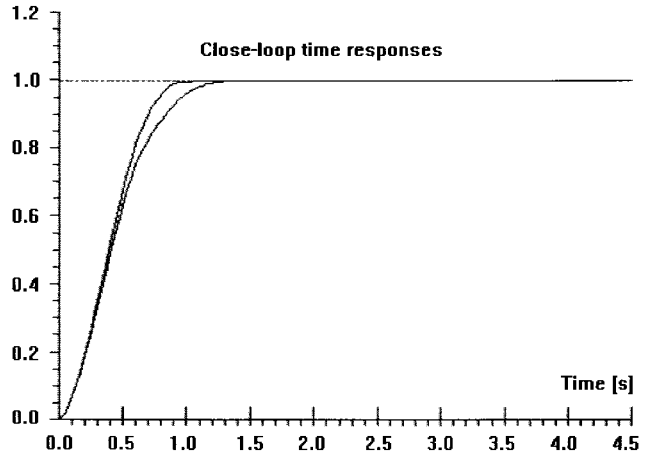


Fig. 4. Time responses of a second-order system while changing its parameters under fuzzy control.

character of a fuzzy logic controller, if a cluster of systems has a similar dynamic behavior we can model all these systems by choosing a system with simple dynamics. Moreover, based on this system, with simple dynamics we can design a fuzzy logic controller for this cluster of systems.

In terms of the second character, we can define very few types of dynamic behavior to describe all response phenomenon. For example, we can use a set of second-order systems to define the following types of dynamic behavior:

1) "oscillation" $(0.0 \leq \zeta \leq 0.2)$;
2) "strong overdamping" $(0.2 < \S \leq 0.4)$;
3) "overdamping" $(0.4 < \S \leq 0.6)$;
4) "appropriate" $(0.6 < \S \leq 0.8)$;
5) "underdamping" $(0.8 < \S \leq 1.3)$;
6) "strong underdamping" $(1.3 < \S \leq 3.0)$, as shown in Fig. 5.

For generalization, in this paper, these qualitative dynamic behaviors are described by using the coefficients a_1, a_2, b_1, and b_2 of a discrete-time equation as follows:

$$y(k)+a_1y(k-1)+a_2y(k-2) = b_1u(k-1)+b_2u(k-2). \quad (3)$$

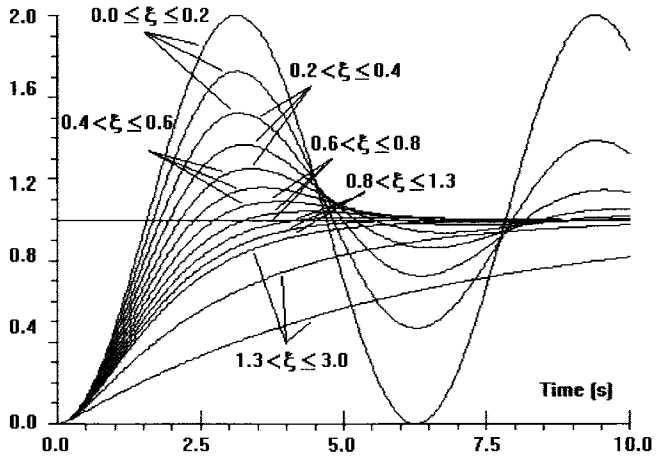


Fig. 5. Definition of dynamic behavior by a set of second-order systems.

For each type of defined behavior, we can optimize its fuzzy logic controller in advance.

A key problem in behavior modeling is how to use the defined types of dynamic behavior to model any dynamics of a system with unknown structure. Obviously, for dynamic responses of a complicated nonlinear plant with unknown structure it is difficult to model them only by using a second-order system. To deal with this problem we divide a whole response process into many segments and to model each segment based behavior models. The idea is to use an algorithm to identify the coefficients a_1, a_2, b_1, and b_2 of a second-order system based on the measured input and output data $u(k)$ and $y(k)(k = n+1, n+2, \cdots, N)$ shown in Fig. 6, i.e., to recognize the dynamic behavior of each segment of the response based on historical information on inputs and outputs. When we model the nonlinear plant by the coefficients a_1, a_2, b_1, and b_2, there exists an error between a second-order system and the nonlinear plant, which can be expressed by

$$\begin{aligned} &y(k) + a_1y(k-1) + a_2y(k-2) \\ &\quad = b_1u(k-1) + b_2u(k-2) + e(k) \end{aligned} \quad (4)$$

where $e(k)$ depends on the coefficients, a_1, a_2, b_1, and b_2, and its evaluating criteria is expressed by

$$J = \sum_{k=n+1}^{N} e^2(k). \quad (5)$$

In doing this, we define the following vectors: $\boldsymbol{u}, \boldsymbol{y}, \boldsymbol{e}, \theta$, and a matrix x:

$$\boldsymbol{u}^T = [u(n+1), u(n+2) \cdots u(N)]^T \quad (6)$$

$$\boldsymbol{y}^T = [y(n+1), y(n+2) \cdots y(N)]^T \quad (7)$$

$$\boldsymbol{e}^T = [e(n+1), e(n+2) \cdots e(N)]^T \quad (8)$$

$$\theta^T = [a_1, \cdots a_n, b_0, \cdots b_n]^T \quad (9)$$

$$\boldsymbol{x} = \begin{bmatrix} -y(n) & -y(n-1) & u(n) & u(n-1) \\ -y(n+1) & -y(n) & u(n+1) & u(n) \\ \cdots & \cdots & \cdots & \cdots \\ -y(N-1) & -y(N-2) & u(N-1) & u(N-2) \end{bmatrix}. \quad (10)$$

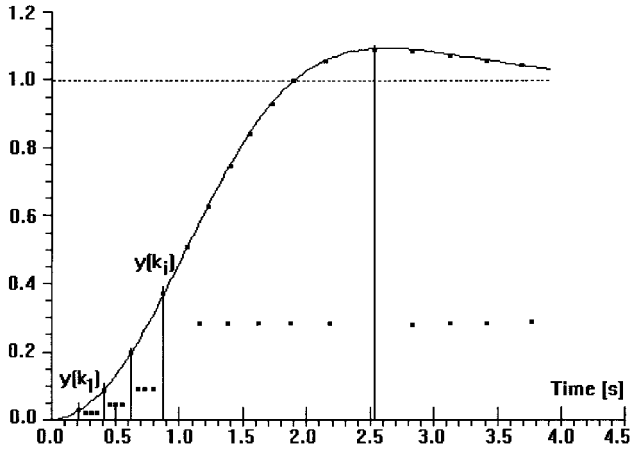


Fig. 6. Dynamic behavior perception.

In behavior modeling, we should compute the coefficient vector $\hat{\theta}$ so that (5) can be minimized. For convenience, (4) is represented by a vector form

$$\boldsymbol{y} = \boldsymbol{x}\theta + \boldsymbol{e} \tag{11}$$

$$J = \boldsymbol{e}^T\boldsymbol{e} = (\boldsymbol{y} - \boldsymbol{x}\theta)^T(\boldsymbol{y} - \boldsymbol{x}\theta) = \boldsymbol{y}^T\boldsymbol{y} - \theta^T\boldsymbol{x}^T\boldsymbol{y} - \boldsymbol{y}^T\boldsymbol{x}\theta + \theta^T\boldsymbol{x}^T\boldsymbol{x}\theta \tag{12}$$

$$\frac{\partial J}{\partial \theta} = -2\boldsymbol{x}^T\boldsymbol{y} + 2\boldsymbol{x}^T\boldsymbol{x}\theta. \tag{13}$$

Let $\partial J/\partial\theta = 0$; we get

$$\boldsymbol{x}^T\boldsymbol{x}\theta = \boldsymbol{x}^T\boldsymbol{y}. \tag{14}$$

By defining matrix $\boldsymbol{A} = \boldsymbol{x}^T\boldsymbol{x}$ there then exits an orthogonal matrix $\boldsymbol{U}$ and an elementary matrix $\boldsymbol{V}, \exists \boldsymbol{U}, \boldsymbol{V}, \partial$: $\boldsymbol{UAV} =$ Diag($\boldsymbol{D}$), and

$$\boldsymbol{D} = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_{(N-n)} \end{bmatrix}. \tag{15}$$

Then (14) can be transformed as

$$\boldsymbol{U^T D V^{-1}}\hat{\theta} = \boldsymbol{x^T y} \tag{16}$$

$$\boldsymbol{DV^{-1}}\hat{\theta} = \boldsymbol{Ux^T y}. \tag{17}$$

Let $\beta = \boldsymbol{Ux}^T\boldsymbol{y}, \gamma = \boldsymbol{V^{-1}}\hat{\theta}$, and $\gamma^T = [h_1 \cdots h_{(N-n)}]$, then $\boldsymbol{D}\gamma = \beta$. We get

$$h_i = \begin{cases} \dfrac{k_i}{d_i}, & \text{if } d_i \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

$$\hat{\theta} = \boldsymbol{V}\gamma. \tag{19}$$

In this paper, $b_0 = 0, n = 2$ and $N = 6$.

For a nonlinear controlled object with uncertainty, there might be two types of practical ways to get its dynamic response behavior. First, it is only possible by using test data during system operation such as boiler control. Second, it can be obtained by simulating a rough analytical dynamics such as a dynamic equation of a manipulator. Therefore, a dynamic response of the controlled object can be expressed by an array of inputs and outputs with a length K_0. On the basis of behavior classification, we can off-line model the nonlinear system by a set of second-order systems in following steps.

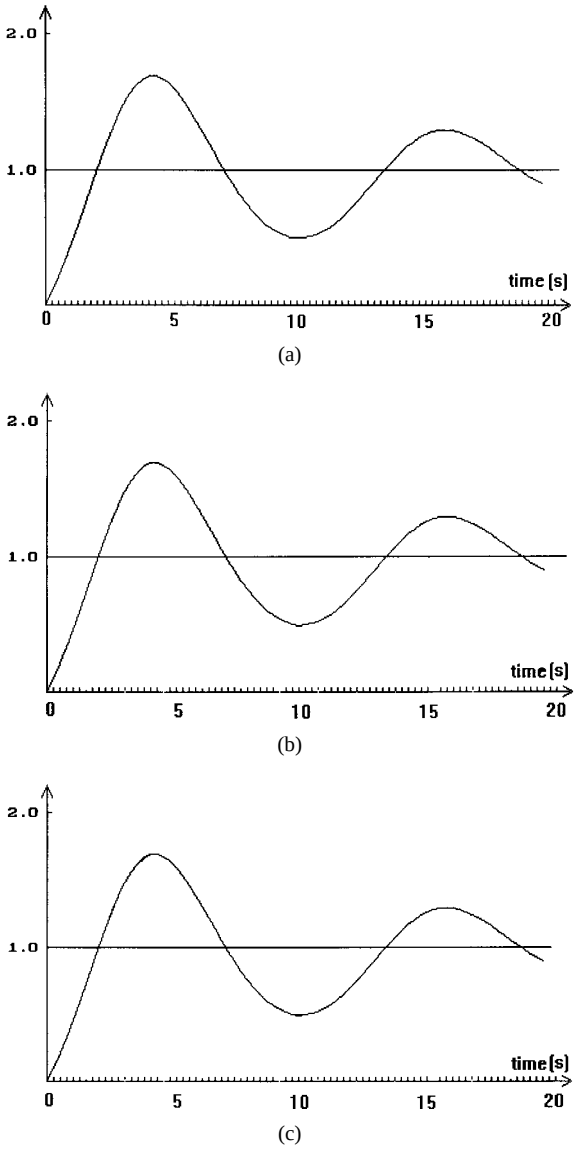


Fig. 7. Identification of dynamic behavior of a nonlinear system. (a) Time response of a nonlinear system. (b) Dynamic response yielded by behavior models. (c) Time responses of both the nonlinear system and the behavior models.

Step 1) Initialize $k_0 = 0$.

Step 2) Get inputs $u(k)$ and outputs $y(k)$ $(k = k_0 + 1, \cdots, k_0 + 6)$ from the test data that represent dynamic behaviors of the controlled objects.

Step 3) Compute the coefficients, a_1, a_2, b_1, and b_2 by using (15)–(19).

Step 4) If $k < K_0$, increase $k_0 = k_0 + 6$ go to Step 1); otherwise go to Step 5).

Step 5) Classify all coefficients, a_1, a_2, b_1, and b_2 for each segment of dynamic response according to the defined types of dynamic behaviors.

In some cases, it may be difficult to model a dynamic response of a nonlinear system by a set of second-order systems due to its complexity. To deal with this problem, we

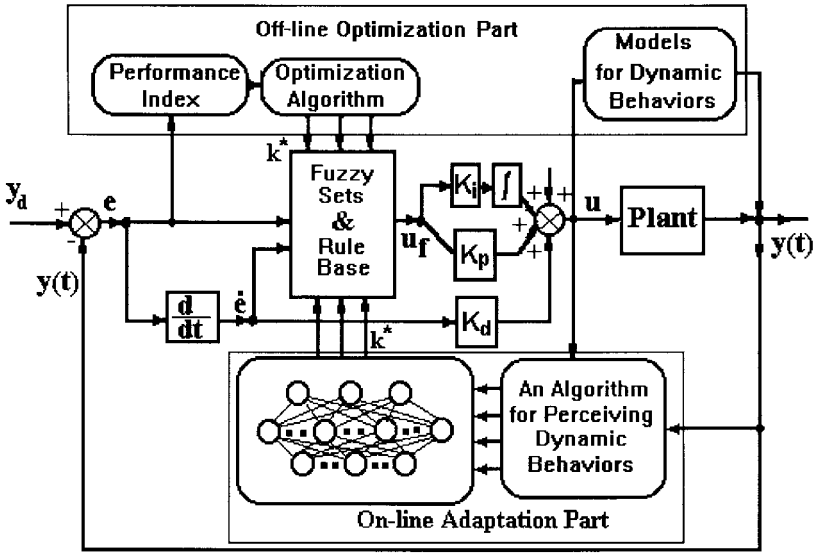


Fig. 8. A neuro-fuzzy control scheme based on behavior modeling.

recommend the use of a set of higher order linear system because it is always possible to approximate a nonlinear system by linearization. In fact, it is not necessary to identify coefficients very exactly by fuzzy logic control.

Fig. 7(a) shows a dynamic response of the nonlinear plant. Fig. 7(b) shows behavior modeling of the nonlinear plant by a set of second-order systems. In Fig. 7(c), both the dynamic responses of the nonlinear plant and the behavior models are plotted together. If we can identify the type of behavior for a segment of response, we can fire its correspondent fuzzy logic controller.

IV. Optimization of Membership Functions of Cubic Splines

Fig. 8 shows a neuro-fuzzy control scheme based on behavior modeling, which consists of four parts: 1) a fuzzy proportional integral (PI) controller and a conventional derivative D controller, 2) an algorithm for off-line optimization of the fuzzy logic controller, 3) a unit for behavior modeling which consists of a set of simplified models, and 4) a unit for behavior perception which consists of an identification algorithm discussed above and a standard back-propagation neural network.

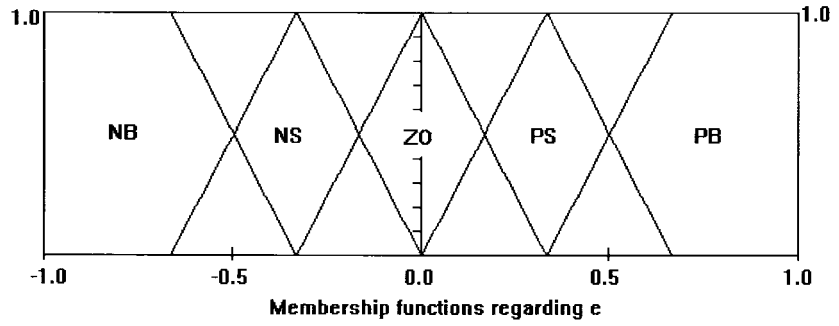
By combining the fuzzy PI and D controllers, static and transient behaviors of a system can be improved [3]. A control signal u for a controlled object is computed as follows:

$$u(k) = K_p u_f(k) + K_i \Delta T \sum_{j=1}^{k} u_f(j) + \frac{K_d}{\Delta T}(e(k) - e(k-1)) \tag{20}$$

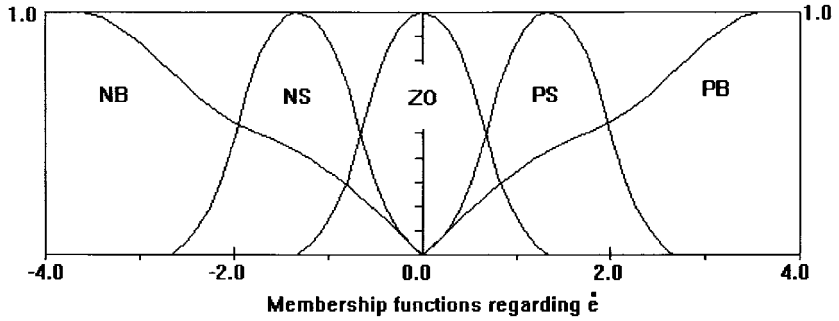
where ΔT is a sampling time and $u_f(k)$ is an output of the fuzzy controller and is a control signal to a plant.

This hybrid neuro-fuzzy control system is operated in off-line and on-line two steps. In the off-line optimization step, we use the behavior models to replace a controlled object with unknown structure, and we optimize parameters of a fuzzy logic controller according to the defined types of dynamic behavior. For optimization of a fuzzy logic controller, of course, many optimization techniques can be used such as a genetic algorithm. On the basis of our previous work [9], in this paper, the Nelder and Mead's simplex algorithm is used to optimize membership functions of fuzzy controllers.

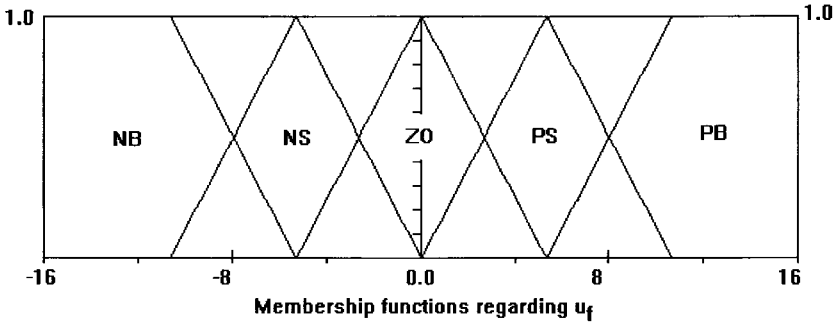
Optimization of a fuzzy logic controller is to refine its membership functions and rule base. We can understand that a fuzzy rule base represents human knowledge qualitatively, whereas membership functions change qualitative human knowledge into quantitative computation. For some problems, a fuzzy rule base can be clearly obtained by using human knowledge. In this case, a key problem is how to determine membership functions to realize human knowledge efficiently. For the control problem discussed in this paper, the fuzzy rule bases could be fixed, as shown in Table I, since human knowledge about the problem is clear. There are different strategies to choose membership functions, but we believe that, in practice, one should avoid changing all membership functions for optimization of a fuzzy controller. To reduce the number of membership functions to be optimized, therefore, it is necessary to investigate effect of membership functions on control performance. The study in [10] shows that the membership functions regarding change-in-error $\dot{e}(k)$ affect dynamic responses very strongly because they represent the feedback of velocity. We use a very simple example to explain this argument. If we remove all rules and membership functions regarding change-in-error $\dot{e}(k)$ in a traditional fuzzy controller, the fuzzy logic controller becomes a nonlinear proportional controller. Using such a fuzzy controller it is very difficult to control an unstable system, e.g., an inverse pendulum. Moreover, optimization of membership functions is to search for reasonable combination of different types of membership functions regarding $e(k), \dot{e}(k)$, and $u_f(k)$. Therefore, it is possible that a fine tuning of the fuzzy logic controller can be achieved by optimizing the membership functions regarding change-in-error $\dot{e}(k)$ after the membership functions regarding $e(k)$ and $u_f(k)$ are roughly defined according to human experience.



(a)



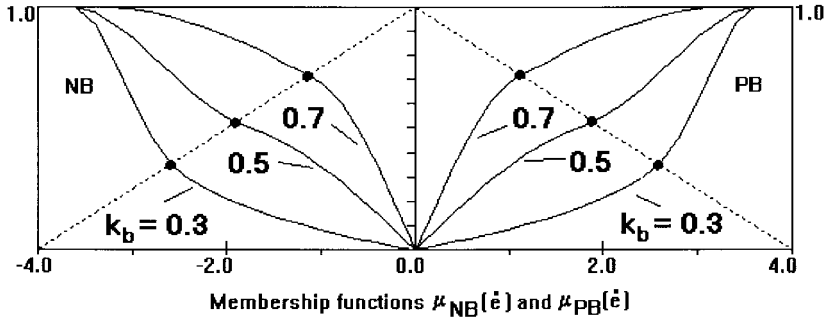
(b)



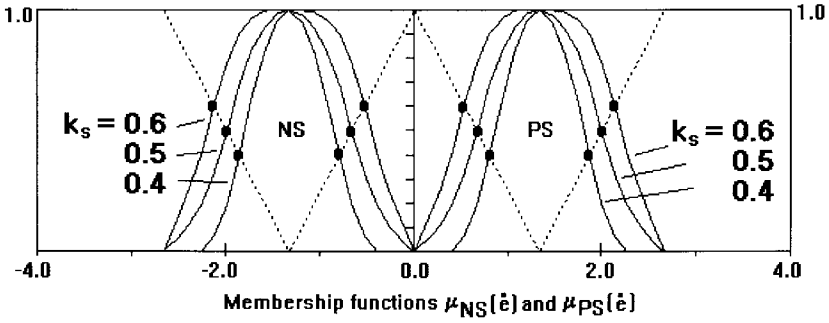
(c)

Fig. 9. Membership functions of linguistic variables.

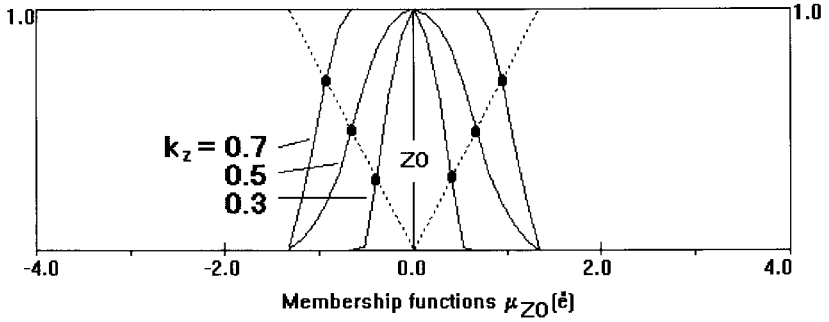
In [8], we have used cubic splines to define the membership functions regarding change-in-error $\dot{e}(k)$, as shown in Fig. 9(b). Their shapes can be modified by shifting the "moving" points along the dashed lines by their parameters shown in Fig. 10. The parameters k_b, k_s, and k_z, denoted by a vector $\boldsymbol{k}$, are used to adjust the membership functions $\mu_{NB}(\dot{e})$ and $\mu_{PB}(\dot{e})$, $\mu_{NS}(\dot{e})$, and $\mu_{PS}(\dot{e})$, and $\mu_{ZO}(\dot{e})$, as shown in Fig. 10(a)–(c), respectively. The vector $\boldsymbol{k}$ can be changed in the range [0.15, 0.85] $\times$ [0.15, 0.85] $\times$ [0.15, 0.85]. On the contrary, the membership functions regarding $e(k)$ and $u_f(k)$ are chosen to be the triangular type and remain unchanged during operations, as shown in Fig. 9(a) and 9(c), respectively. To investigate the effect of membership functions regarding $\dot{e}(k)$ on control performance, we systematically perform the following numerical simulations. Here, the nonlinear system in (2) with $\omega = 1.0$ and $\xi = 1.0$ is chosen as the controlled object. First, we change $\mu_{NB}(\dot{e})$ and $\mu_{PB}(\dot{e})$ by increasing k_b from 0.15 to 0.85 when $k_s = 0.5$ and $k_z = 0.5$. At $k_b = 0.15$, the time response exhibits an underdamped behavior, as shown in Fig. 11(a). Since, in this case, the areas under $\mu_{NB}(\dot{e})$ and $\mu_{PB}(\dot{e})$ are very small, the rules associated with NB and PB have a weak effect on $\mu_f(k)$. Consequently, the negative feedback becomes weak. By increasing k_b, the time response becomes more damped. At $k_b = 0.85$, the time response exhibits an overdamped behavior shown in Fig. 11a. Since, in the case, the areas $\mu_{NB}(\dot{e})$ and $\mu_{PB}(\dot{e})$ are very large, the rules associated with NB and PB have a strong effect on $u_f(k)$. Consequently, the negative feedback becomes strong. In similar manner, we change $\mu_{NS}(\dot{e})$ and $\mu_{PS}(\dot{e})$ by increasing k_s from 0.15 to 0.85 when $k_b = 0.5$ and $k_z = 0.5$. At $k_s = 0.15$, the time response is more overdamped than that at $k_s = 0.85$, as shown in Fig. 11(b). Because the areas under $\mu_{NS}(\dot{e})$ and $\mu_{PS}(\dot{e})$ are very small at $k_s = 0.15$, the rules associated with NS and PS have a weak effect on $u_f(k)$, whereas the effects of the rules associated with NB and PB get relative stronger. Finally, we change $\mu_{ZO}(\dot{e})$ by increasing k_z from 0.15 to 0.85 when $k_b = 0.5$ and $k_s = 0.5$. Fig. 11(c) shows that the difference between the responses at $k_z = 0.15$



(a)



(b)



(c)

Fig. 10. Membership functions of cubic splines.

and $k_z = 0.85$ is very little. The studies show that the effects of $\mu_{NB}(\dot{e})$ and $\mu_{PB}(\dot{e})$ on control performance are strongest, whereas that of $\mu_{ZO}(\dot{e})$ is weakest.

To optimize the membership functions of cubic splines, the integral-of-time-multiplied absolute-error (ITAE) criterion

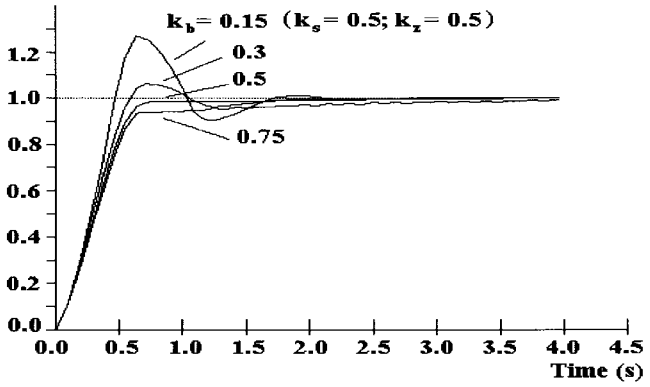
$$H = \int_0^0 t|e(t)|\, dt \tag{21}$$

is used to describe control performance. Since $H = f(k_z, k_s, k_b)$ is a function of the parameter vector $\boldsymbol{k} = (kz, k_s, k_b)$, the optimization of the membership functions of cubic splines is the computation of the minimum value, H^*, by searching for the corresponding vector $\boldsymbol{k}^*$ of the membership functions. Computing the minimum value $f(\boldsymbol{k}^*)$ by the simplex algorithm is briefly described as follows [11].

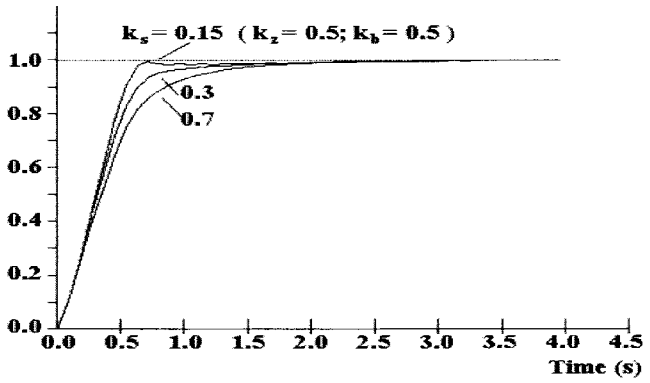
1) Start with three points $\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3$ and compute $H_1 = f(\boldsymbol{k}_1), H_2 = f(\boldsymbol{k}_2)$, and $H_3 = f(\boldsymbol{k}_3)$.
2) Find the maximum value H_h, the next maximum value H_g, and the minimum value H_l and the corresponding points $\boldsymbol{k}_h, \boldsymbol{k}_g, \boldsymbol{k}_l$.
3) Find the center point of the points $\boldsymbol{k}_g$ and $\boldsymbol{k}_l$: $\boldsymbol{k}_f = 0.5(\boldsymbol{k}_g + \boldsymbol{k}_l)$ and evaluate $H_f = f(\boldsymbol{k}_f)$.
4) Reflect $\boldsymbol{k}_h$ in $\boldsymbol{k}_f$ to find $\boldsymbol{k}_r$ and $H_r = f(\boldsymbol{k}_r)$.
5) If $H_r \geq H_g$, proceed to the contraction and compute $\boldsymbol{k}_c = \beta \boldsymbol{k}_h + (1 - \beta)\boldsymbol{k}_f$ and $H_c = f(\boldsymbol{k}_c)$ where $\beta (0 < \beta < 1)$ is the contraction coefficient. If $h_r < H_g$, compute $\boldsymbol{k}_c = \beta \boldsymbol{k}_r + (1 - \beta)\boldsymbol{k}_f$ and $H_c = f(\boldsymbol{k}_c)$.
6) If $H_c < H_h$, replace $\boldsymbol{k}_h$ by $\boldsymbol{k}_c$, check convergence, and, if not, return to 2). If $H_c \geq H_h$, move to next step.
7) Reduce the size of the simplex by $\boldsymbol{k}_h = \boldsymbol{k}_h + 0.5(\boldsymbol{k}_h - \boldsymbol{k}_l)$ and $\boldsymbol{k}_g = \boldsymbol{k}_g + 0.5(\boldsymbol{k}_g - \boldsymbol{k}_l)$, and calculate H_h and H_g, test for convergence, and, if not, return to 2).

V. Neural Network Training for On-Line Adaptation of Fuzzy Controller

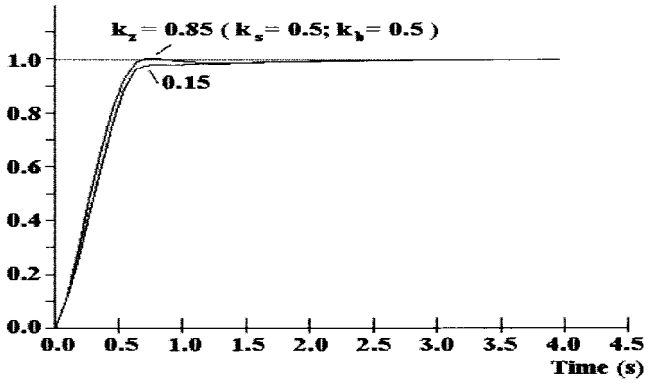
In the on-line adaptive step, the unit for behavior modeling is switched off. During system operation, the identification algorithm is used to identify dynamic behavior of a controlled object according to the last segment of dynamic response, and the neural network is used to determine the parameters of the



(a)



(b)



(c)

Fig. 11. Effect of membership functions regarding change-in-error on control performance. (a) Time responses caused by changing. (b) Time responses caused by changing. (c) Time responses caused by changing.

fuzzy logic controller based on a the type of dynamic behavior. To achieve this objective, a three-layer BP neural network is used to build a mapping relationship between the types of dynamic behavior and parameters of their optimized fuzzy logic controllers. The input patterns of the neural network are coefficients a_1, a_2, b_1, and b_2 that are used to describe different types of dynamic behavior. The output patterns from the neural network are the optimized vector $\boldsymbol{k}^* = (k_z^*, k_s^*, k_b^*)$ of the membership functions for control of each type of dynamic behavior. The output $q_j^{[s]}$ of the jth neuron on the sth hidden layer is calculated as

$$q_j^{[s]} = f(\mathrm{Net}_j^{[s]}) = f\left(\sum_i (w_{ji}^{[s]} * q_i^{[s-1]})\right) \tag{22}$$

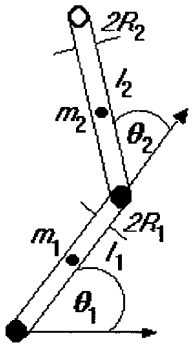


Fig. 12. A two-link manipulator with rotary joints.

where $q_{ji}^{[s]}$ is the weight on connection joining ith neuron in layer $(s-1)$ to jth neuron in layer s, and $f(x)$ is a sigmoid logistic function

$$f(x) = \frac{1}{1+e^{-x}}. \tag{23}$$

The Widrow–Hoff δ learning rule is used to modify the weight $w_{ji}^{[s]}$ as follows:

$$\begin{aligned}
\delta_z &= f'(w_{zi}^{[3]} * q_{zi}^{[3]}) \sum_k (t_{zk}^* - \hat{t}_{zk}) \\
\delta_s &= f'(w_{si}^{[3]} * q_{si}^{[3]}) \sum_k^k (t_{sk}^* - \hat{t}_{sk}) \\
\delta_b &= f'(w_{bi}^{[3]} * q_{bi}^{[3]}) \sum_k^k (t_{bk}^* - \hat{t}_{bk})
\end{aligned} \tag{24}$$

for the output layer and

$$\delta_j^{[s]} = f'(\mathrm{Net}_j^{[s]}) \sum_k (\delta_k^{[s+1]} \cdot w_{kj}^{[s+1]}) \qquad s = 1, 2, 3 \tag{25}$$

for any other layers. The connection weights $w_{ji}^{[s]}$ of the neural network are updated by

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t+1) \tag{26}$$

$$\Delta w_{ij}(t+1) = \eta \delta_j^{[s]}\, \mathrm{Net}_i^{[s-1]} + \alpha \Delta w_{ij}(t) \tag{27}$$

where η is a learning coefficient, and α is a momentum constant. After patterns training, this hybrid neuro-fuzzy control system can be used in on-line control of a nonlinear controlled object.

VI. Control of a Manipulator

In this section, the proposed method is used to design a neuro-fuzzy control system for control of a two-link manipulator in Fig. 12 whose dynamic equations can be found in [3]. Because the dynamic equations, in absence of friction, represent an unstable controlled object without controllers, we design its controllers as follows.

1) Use proportional-integral-derivative (PID) controllers to control each joint of the manipulator. The idea is a rough design of PID controllers to make manipulator operation stable. In this case, control performance could not be

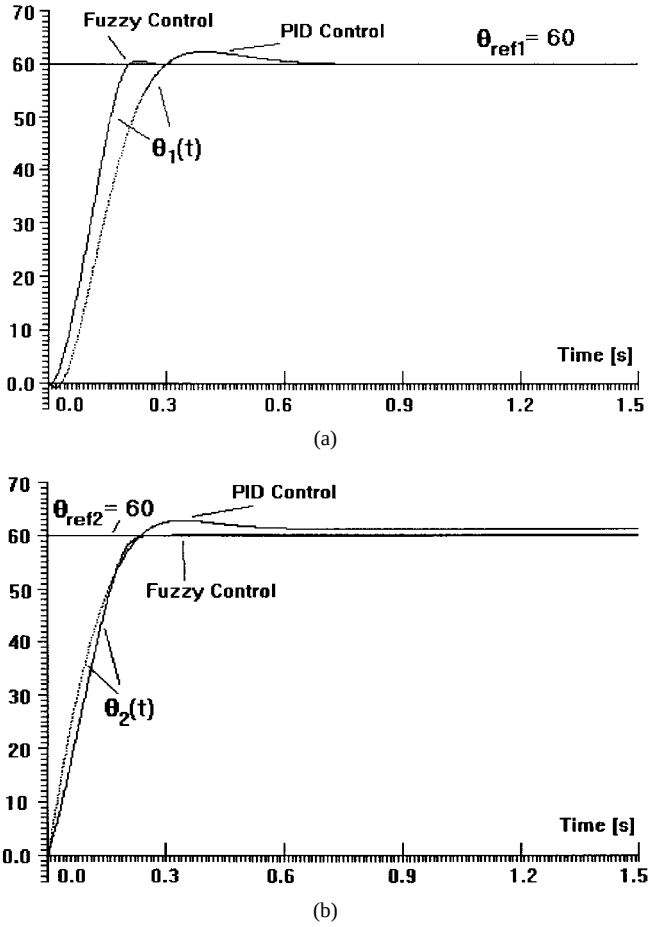


Fig. 13. Manipulator time responses for step control. (a) Joint one. (b) Joint two.

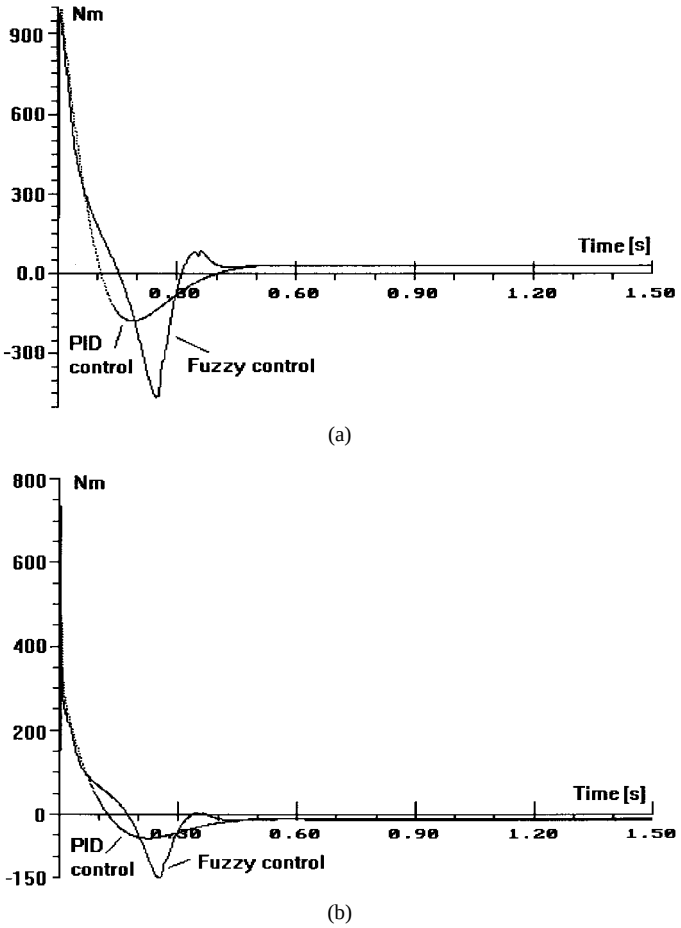


Fig. 14. The applied torques to joints one and two computed by PID and fuzzy control. (a) Joint one. (b) Joint two.

desired usually, but it is acceptable. PID control parameters k_{pj}, K_{ij} and K_{dj} are determined by Ziegler–Nichols technique [12].

2) Obtain dynamic responses of the manipulator under PID control by simulating its rough dynamic equations or by operating the real manipulator.
3) Build behavior models according to dynamic responses of the manipulator under PID control.
4) Optimize the vector $\boldsymbol{k}^* = (k_z^*, k_s^*, k_b^*)$ of the membership functions as discussed in Section IV, based on behavior models of the manipulator.
5) Train the neural network to establish the map relationship between each type of dynamic behavior and the optimized parameters $\boldsymbol{k}^* = (k_z^*, k_s^*, k_b^*)$ of its fuzzy logic controller.

Fig. 13(a) and (b) shows the step responses to the joints one and two of the manipulator, respectively. In these simulation studies, ΔT was chosen to be 2 m's; the initial anglers $\theta_1(0)$ and $\theta_2(0)$ were set to be zero degree, and the reference values θ_{ref1} and θ_{ref2} were chosen as 60°. It can be observed that the time responses under PID control (rough control) exhibit much larger overshoots, slower settling time, and larger steady-state errors. Of course, control performance can be improved by carefully adjusting the PID control parameters. To do this, however, it is not very easy because the dynamic equations

TABLE I
FUZZY RULE BASE

$\dot{e}$ \ e	NB	NS	ZO	PS	PB
NB	PB	PB	PB	PS	ZO
NS	PB	PB	PS	ZO	NS
ZO	PB	PS	ZO	NS	NB
PS	PS	ZO	NS	NB	NB
PB	ZO	NS	NB	NB	NB

of the manipulator are highly nonlinear. When the fuzzy logic controller of each joint, whose parameter vector $\boldsymbol{k}^* = (k_z^*, k_x^*, k_b^*)$ is determined by neural network, are switched on to control the manipulator, control performance becomes much better, e.g., smaller overshoots, faster settling time, and higher precision. Fig. 14 shows the applied torques to each joint, computed by PID and fuzzy control, respectively. It should be noted that the maximal applied torques to joint one and two, computed by PID control, are $\tau_{1\max} = 996$ Nm and $\tau_{1\max} = 733$ Nm. However, the applied torques to joint one and two, computed by fuzzy control, are $\tau_{1\max} = 975$ Nm and $\tau_{2\max} = 604$ Nm. This example illustrates that the proposed neuro-fuzzy controllers not only need lower energy,

but also achieve much better control performance than the PID controllers do those. Therefore, the maximum torque of each joint motor can be reduced by the proposed neuro-fuzzy controllers.

Here, the strategy for design of the neuro-fuzzy control system for the manipulator is based on rough design of PID controllers and fine design of fuzzy controllers. Since by using behavior modeling we can describe dynamic responses of the manipulator under PID control without any difficulty, fuzzy logic controllers are used to improve control performance yielded by PID control. In a conservative view, this ensures that control performance under fuzzy control is better than one under PID control. In [14], we report control results of a real PUMA 560 robot by using this hybrid neuro-fuzzy control system.

VII. Conclusions

This paper presents a new method for designing a neuro-fuzzy control system based on behavior modeling. Using behavior modeling, a complicated control problem is transformed to off-line to model a controlled object and on-line to adapt its controller.

J. C. Bezdek points out that fuzzy models seem particularly well suited for a smooth transition from numerical (quantitative) to semantic (qualitative) information [13]. Therefore, for a fuzzy controller, we have to design its semantic information and to determine its numerical information. For some control problems, its semantic information (rule base) can be directly defined by human-knowledge. For example, a rule base for position control of a manipulator in this paper. In this case, a key problem is how to determine its quantitative information (fuzzy sets or membership functions). Since for any given plant quantitative information is not uniquely represented by fuzzy sets, it is desired to adjust a few of membership functions to achieve a good control performance. Therefore, we propose for optimizing the membership functions related to change-in-error. If both numerical and semantic information are not available, a rule base and membership functions should be optimized synchronously.

On the basis of behavior modeling, we can optimize fuzzy logic controllers using different optimization strategy. Because of our previous work, the simplex algorithm is implemented in this control system. We would like to recommend to use a genetic algorithm for off-line optimization.

Here, we would like to emphasize the strategy for design of the neuro-fuzzy control system in two steps: rough design of PID controllers and fine design of fuzzy controllers. This strategy is practical because, in most control applications, PID controllers have been widely adopted. Since in using this design strategy fuzzy logic controllers are used to improve control performance yielded by PID controller, this strategy can ensure that control performance is better than one yielded by PID control.

This study is based on the assumption that design of a fuzzy controller depends on a response behavior of a controlled plant rather than its analytical model. Although this view is often adopted in designing a fuzzy logic controller in practice, we would try to provide some rigorous proof or limitation conditions in our further research.

Acknowledgment

The author would like to thank the referees for detailed and helpful comments.

References

[1] M. Braae and D. A. Rutherford, "Theoretical and linguistic aspects of the fuzzy logic controller," *Automatica*, vol. 15, pp. 553–577, 1979.
[2] S. Tzafestas and N. Papanikolopoulos, "Incremental fuzzy expert PID control," *IEEE Trans. Ind. Electron.*, vol. 37, pp. 365–371, Oct. 1990.
[3] W. Li and B. Zhang, "Fuzzy control of robotic manipulators in the presence of joint friction and loads changes," in *Proc. ASME Int. Comput. Eng. Conf.*, San Diego, CA, Aug. 1993.
[4] T. J. Procyk and E. H. Mamdani, "A linguistic self-organizing process controller," *Automatica*, vol. 15, pp. 15–30, 1979.
[5] E. M. Scharf and N. J. Mandic, "The application of a fuzzy controller to the control of a multi-degree-freedom robot arm," *Industry Applications of Fuzzy Control*, M. Sugeno, Ed. Amsterdam, The Netherlands: North Holland, 1985, pp. 41–61.
[6] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Trans. Neural Networks*, vol. 3, pp. 724–740, June 1992.
[7] J. S. R. Jang, "Self-Learning fuzzy controllers based on temporal back propagation," *IEEE Trans. Neural Networks*, vol. 3, pp. 714–723, June 1992.
[8] W. Li, "Optimization of a fuzzy logic controller using neural network," in *Proc. (FUZZ-IEEE '94) IEEE World Congress Computat. Intell.*, Orlando, FL, June 1994, vol. 1, pp. 223–227.
[9] W. Li, Z. Q. Sun, and H. Janocha, "An approach to automatic tuning of a fuzzy logic controller for manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Munich, Germany, Sept. 1994, pp. 634–640.
[10] W. Li, B. Zhang, P. F. Jin, and K. Z. He, "Effect of membership functions of linguistic variables on control performance," in *Proc. 1st Chinese World Congress Intell. Contr. Intell. Automat.*, Beijing, Aug. 1993, pp. 866–876.
[11] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comp. J.*, vol. 7, pp. 308–313, 1965.
[12] J. G. Ziegler and N. B. Nichols, "Optimum setting for automatic controllers," *Trans. Amer. Soc. Mech. Eng.*, vol. 8, pp. 759–768, 1942.
[13] J. C. Bezdek, "What is computational intelligence?," in *Computational Intelligence*, J. M. Zurada, R. J. Marks II, and C. J. Robinson, Eds. Piscataway, NJ: IEEE Press, 1994, pp. 1–12.
[14] W. Li and Q. H. Tan, "A method for design of a neuro-fuzzy controller based on behavior classification," in *Proc. IEEE Int. Conf. Neural Networks*, Perth, Australia, Nov. 1995, pp. 2275–2280.