Dynamically Focused Fuzzy Learning Control

Waihon A. Kwong and Kevin M. Passino, *Member, IEEE*

Abstract— **A "learning system" possesses the capability to improve its performance over time by interacting with its environment. A learning control system is designed so that its "learning controller" has the ability to improve the performance of the closed-loop system by generating command inputs to the plant and utilizing feedback information from the plant. Learning controllers are often designed to mimic the manner in which a human in the control loop would learn how to control a system while it operates. Some characteristics of this human learning process may include: (i) a natural tendency for the human to focus their learning by paying particular attention to the current operating conditions of the system since these may be most relevant to determining how to enhance performance; (ii) after learning how to control the plant for some operating condition, if the operating conditions change, then the best way to control the system may have to be relearned; and (iii) a human with a significant amount of experience at controlling the system in one operating region should not forget this experience if the operating condition changes. To mimic these types of human learning behavior, we introduce three strategies that can be used to dynamically focus a learning controller onto the current operating region of the system. We show how the subsequent "dynamically focused learning" (DFL) can be used to enhance the performance of the "fuzzy model reference learning controller" (FMRLC) [1]–[5] and furthermore we perform comparative analysis with a conventional adaptive control technique. A magnetic ball suspension system is used throughout the paper to perform the comparative analyses, and to illustrate the concept of dynamically focused fuzzy learning control.**

I. Introduction

In recent years, there has been a significant growth in the commercial and industrial use of fuzzy logic systems. Subway systems, automobile transmissions, air-conditioners, washing machines, autofocus cameras and camcorders have employed fuzzy control [6]–[8], and these products have been advertised to have "intelligence" as compared to their "nonfuzzy" counterparts. While there is significant marketing hype about fuzzy systems being intelligent, in reality the fuzzy controller is no more than a nonlinear controller. Hence, there exists the same type of problems in design and implementation of fuzzy controllers as exist in conventional control. For instance, there is the need to specify a set of performance objectives (e.g., stability, rise-time, overshoot etc.) and show that these objectives are achieved. In addition, as with conventional fixed (i.e., nonadaptive) control, (i) it can be difficult to tune the parameters of the controller and (ii) there always exists the possibility that upon implementation plant parameter variations will result in performance degradation. A multitude of approaches to adaptive control have been introduced to address these two problems. For instance, there exist many approaches to model reference adaptive control (MRAC) that seek to tune the parameters of a linear controller in response to plant variations so that the behavior specified in a "reference model" is achieved [9]. In an analogous fashion to MRAC, the work in [1]–[5] shows how a fuzzy model reference learning controller (FMRLC) can achieve and maintain the performance specified in a reference model by synthesizing and tuning a fuzzy controller. In this paper, we investigate the possibility of enhancing the FMRLC learning capabilities via various "dynamically focused learning" (DFL) strategies that seek to optimally allocate the fuzzy controller rules to the operating region of the system. In particular, we show that the DFL-enhanced FMRLC can outperform the standard FMRLC and an MRAC technique for a magnetic ball suspension system (this paper is an expanded version of [10]).

Fuzzy fuzzy set theory was originated by Lotfi A. Zadeh [11], [12] at the University of California, Berkeley. Zadeh proposed the possibility of using fuzzy set theory as a means of analyzing some very complex real world dynamical systems. However, it was the pioneering research by E. H. Mamdani and his colleagues at Queen Mary College in England [13]–[15] that introduced how fuzzy set theory could be employed in control applications. Since then, most fuzzy controllers have basically been based on the suggested model by Mamdani, where the fuzzy logic rules are obtained from expert knowledge, the laws of physics, and similar *a priori* information. Regardless of the information used in their construction, fuzzy controllers turn out to be nonlinear controllers. Design procedures for fuzzy controllers often involve ad hoc tuning of the fuzzy controller parameters. To address this problem, Procyk and Mamdani [16] introduced the linguistic self-organizing controller (SOC) where the fuzzy control laws are automatically improved by modifying the knowledge-base so that a given performance measure is minimized. The modification basically uses a performance evaluation fuzzy system and an inverse of the plant dynamics to change the fuzzy relations in the controller knowledge-base so that the controller can assess its own performance. The linguistic SOC framework of Procyk and Mamdani was studied further by Shao in [17] where a rule-base modification algorithm is designed to reduce computation time and memory. Shao also demonstrated the practical application of the linguistic SOC by employing it in two applications which contained nonlinearities and large time

Manuscript received June 26, 1994; revised February 20, 1995. This work was supported in part by National Science Foundation Grants IRI-9210332 and EEC-9315257.

W. A. Kwong was with the Department of Electrical Engineering, The Ohio State University, Columbus OH 43210-1272 USA. He is now with Epsilon Lammbda Electronics Corp., Chicago, IL.

K. M. Passino is with the Department of Electrical Engineering, The Ohio State University, Columbus OH 43210-1272 USA.

Publisher Item Identifier S 1083-4419(96)00419-0.

lags. Other applications of Procyk and Mamdani's linguistic SOC framework were implemented by Scharf and Mandic [18] and again by Tanscheit and Scharf [19] on a multiple degree-of-freedom robot arm where the fuzzy performance evaluator is optimized. Their design employs an improved performance measure decision table which was proposed by Yamazaki of London University [20]. Isaka *et al.* in [21] demonstrate a practical application of SOC by employing it as a blood pressure controller where the dynamic responses of the human body vary greatly between various persons. Daley and Gill in [22]–[24] describe the application of the linguistic SOC algorithm proposed by Procyk and Mamdani for a complex multi-variable process involving the attitude control of a flexible satellite. In [25]–[27], the SOC approach is further illustrated in various experiments with nonlinear plants.

Despite the many advantages and successes of the linguistic SOC algorithm, several drawbacks do exist. For example, the performance measure employed in the linguistic SOC only characterizes some compromise between rise time and overshoot. For some control problems, the fastest possible rise time with some overshoot may not be a desirable process characteristic. Another potential problem involves the assumption that incremental relationships between process inputs and outputs are monotonic when generating the inverse process model. In general this assumption does not always hold true.

These problems are avoided via the fuzzy model reference learning control (FMRLC) introduced by Layne and Passino in [1]–[5]. The FMRLC framework adopts ideas from conventional MRAC [28] by introducing a reference model for improved performance feedback and definition of the desired process characteristic. The problem associated with the inverse of the plant is solved by the use of a fuzzy inverse model, which contains qualitative information about desired changes in process outputs and maps this to a necessary change in the process inputs for improvements in the control laws. The FMRLC is later successfully implemented in [29], [30] for endpoint positioning of a flexible-link robot and has been studied in simulation for a cargo ship steering problem [2], an inverted pendulum [1], anti-skid braking systems [3], [4], a rocket velocity control problem and a rigid robot [5], and reconfigurable control for aircraft [31].

Other alternatives to FMRLC and SOC are contained in [26], [32], [33]; however, all of the adaptive fuzzy control techniques discussed up to this point fit into the class of adaptive systems often referred to as "direct adaptive control," where the controller is directly updated without first identifying the plant parameters. Alternatively, information about the process may be obtained indirectly by utilizing system identification, and then the controller can be updated based on the identification results; this is often referred to as "indirect adaptive control." Rhee *et al.* [34] present a knowledge-based fuzzy control system, which is constructed off-line. Another example of indirect adaptive fuzzy control presented by Graham and Newell in [35], [36] uses a fuzzy identification algorithm developed by Czogala and Pedrycz [37], [38] to identify a "fuzzy process model" that is then used to determine the control actions. Batur and Kasparian [39] present a methodology to adapt the initial knowledge-base of a fuzzy controller to changing operating conditions. The output membership functions of their fuzzy controller are adjusted in response to the future or past performance of the overall system, where the prediction is obtained through a linear process model updated by on-line identification. In addition to using the direct SOC framework or the indirect approach, there are many other adaptive fuzzy system applications, to name a few, that chose to use neural network for identification and reinforcement learning [40]–[42], or genetic algorithms for natural selection of controller parameters [43]–[46] as the learning mechanism.

Recent work on adaptive fuzzy systems has focused on merging concepts and techniques from conventional adaptive systems into a fuzzy systems framework. Most notable is the work of Wang that is gathered in [47] where he shows how to construct: (i) fuzzy estimators/identifiers using, for example, least squares, back-propagation, and clustering techniques; (ii) stable (direct and indirect) adaptive fuzzy controllers; and (iii) fuzzy adaptive filters. Our approach is significantly different from Wang's since we seek to characterize general ways in which humans might learn to control a process and utilize these in adaptive control; his focus is on the design and nonlinear analysis of adaptive fuzzy systems. Next, we explain the organization of this paper.

In Section II, we describe the nonlinear model of the magnetic ball suspension system which will be used to illustrate the concepts and techniques in the paper. Then we develop a conventional adaptive controller and a standard FMRLC for the magnetic ball suspension system and demonstrate how the FMRLC fails to achieve the control objectives. Via the failure of the FMRLC, we motivate the need for the dynamically focused learning (DFL). In Section III, we introduce three types of DFL strategies and evaluate their performance relative to MRAC and FMRLC. Section IV contains some concluding remarks and future research directions.

II. Conventional Adaptive Control and FMRLC for a Magnetic Ball Suspension System

In this section we develop a conventional adaptive controller and FMRLC for a magnetic ball suspension system and perform a comparative analysis to assess the advantages and disadvantages of each approach. At the end of this section, we highlight certain problems that can arise with the FMRLC and use these as motivation for the dynamically focused learning enhancement to the FMRLC.

A. Magnetic Ball Suspension System

The model of the magnetic ball suspension system[1] shown in Fig. 1 is given by

$$M\frac{d^2y(t)}{dt^2} = Mg - \frac{i^2(t)}{y(t)}$$
$$v(t) = Ri(t) + L\frac{di(t)}{dt} \tag{1}$$

[1] An experimental magnetic ball suspension system is described in [48]. It is interesting to note that the system parameters of their experimental setup are quite similar to those used in our model.

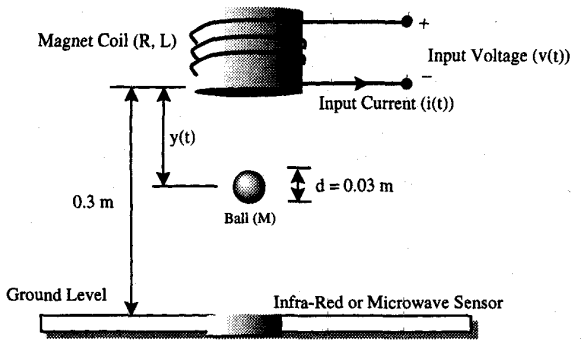


Fig. 1. Magnetic ball suspension system.

where $y(t)$ is the ball position in meters, $M = 0.1$ kg is the ball mass, $g = 9.8$ m/s^2 is the gravitational acceleration, $R = 50\ \Omega$ is the winding resistance, $L = 0.5$ H is the winding inductance, $v(t)$ is the input voltage, and $i(t)$ is the winding current. The position of the ball is detected by a position sensor (e.g. an infra-red or microwave sensor) and is assumed to be fully detectable over the entire range between the magnetic coil and the ground level. In state-space form (1) becomes

$$\begin{aligned}\frac{dx_1(t)}{dt} &= x_2(t)\\ \frac{dx_2(t)}{dt} &= g - \frac{x_3^2(t)}{Mx_1(t)}\\ \frac{dx_3(t)}{dt} &= -\frac{R}{L}x_3(t) + \frac{1}{L}v(t)\end{aligned} \tag{2}$$

where $[x_1(t)\ x_2(t)\ x_3(t)]' = [y(t)\ \frac{dy(t)}{dt}\ i(t)]'$ (where "$'$" denotes matrix transpose). Notice that the nonlinearities are induced by the $x_3^2(t)$ and $\frac{1}{x_1(t)}$ terms in the $\frac{dx_2(t)}{dt}$ equation. By linearizing the plant model in (2), assuming that the ball is initially located at $x_1(0) = y(0)$, a linear system can be found by calculating the Jacobian matrix at $y(0)$. The linear state-space form of the magnetic ball suspension system is given as

$$\begin{aligned}\frac{dx_1(t)}{dt} &= x_2(t)\\ \frac{dx_2(t)}{dt} &= \frac{g}{y(0)}x_1(t) - 2\sqrt{\frac{g}{My(0)}}x_3(t)\\ \frac{dx_3(t)}{dt} &= -\frac{R}{L}x_3(t) + \frac{1}{L}v(t).\end{aligned} \tag{3}$$

Since the ball position $y(t)$ is the only physical output of the plant, by assuming all initial conditions are zero, the model can be rewritten as a transfer function

$$\frac{\hat{y}(s)}{\hat{v}(s)} = \frac{-\frac{2}{L}\sqrt{\frac{g}{My(0)}}}{\left(s^2 - \frac{g}{y(0)}\right)\left(s + \frac{R}{L}\right)} \tag{4}$$

(in this section, we adopt the convention that if $z(t)$ is a time function, $\hat{z}(s)$ is its Laplace transformation). Note that there are three poles (two stable and one unstable) and no zeros in the transfer function in (4). Two poles (one stable and one unstable) and the dc gain change based on the initial position of the ball (i.e., the system dynamics will vary significantly depending on the location of the ball). From Fig. 1, the total distance between the magnetic coil and the ground level is 0.3 m, and the diameter of the ball is 0.03 m. Thus, the total length of the suspension system is 0.27 m, and the initial position of the ball $y(0)$ can be anywhere between 0.015 m (touching the coil) and 0.285 m (touching the ground). For this range the numerator of the transfer function $-\frac{2}{L}\sqrt{\frac{g}{My(0)}}$ varies from -323.3 (ball at 0.015 m) to -74.17 (ball at 0.285 m), while the two poles move from ± 25.56 to ± 5.864 as shown in the pole-zero map in Fig. 2. Clearly then the position of the ball will affect our ability to control it. If it is close to the coil it may be difficult to control since the unstable pole moves further out into the right half plane, while if it is near the ground level it is easier to control. The effect of the ball position on the plant dynamics can cause problems with the application of fixed linear controllers (e.g. ones designed with root locus or Bode techniques that assume the plant parameters are fixed). It is for this reason that we investigate the use of a conventional adaptive controller and the FMRLC for this control problem. We emphasize, however, that our primary concern is *not* with the determination of the best control approach for the magnetic ball suspension system; we simply use this system as an example to compare control approaches and to illustrate the ideas in this paper.

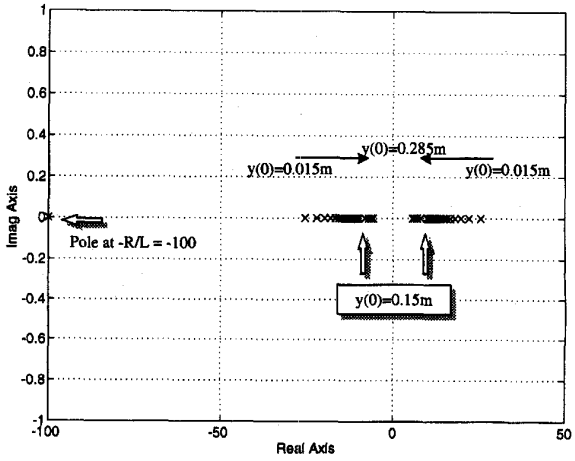


Fig. 2. Pole-zero map of the magnetic ball suspension system (third order linear model with all possible initial conditions).

B. Conventional Adaptive Control

In this section a model reference adaptive controller (MRAC) is designed for the magnetic ball suspension system. The particular type of MRAC we use is described in [9] (on p. 125) and it uses the so called "indirect" approach to adaptive control where the updates to the controller are made by first identifying the plant parameters. The MRAC controller structure is shown in Fig. 3, for which every component is discussed next (it is assumed that the reader is familiar with the concepts and techniques in conventional adaptive control).

To design the MRAC, a linear model is required. To make the linear model most representative of the range of dynamics of the nonlinear plant we assume that the ball is initialized

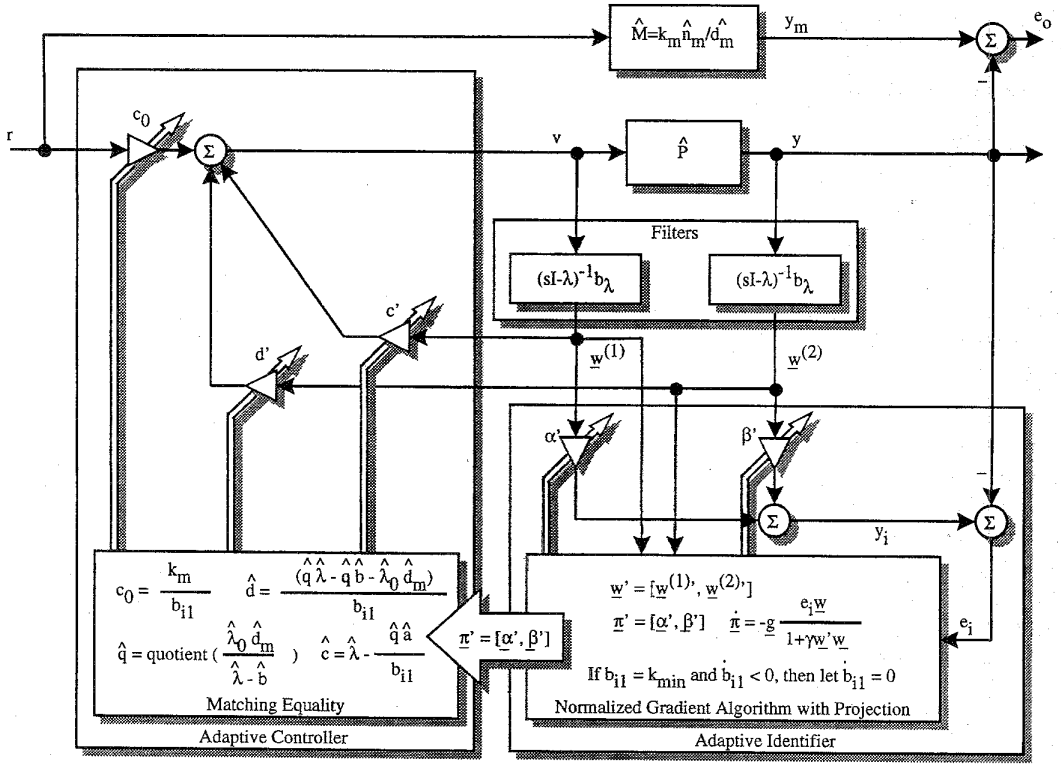


Fig. 3. Model reference adaptive controller (MRAC) structure.

at the middle between the magnetic coil and the ground level where $y(0) = 0.15$ m to perform our linearization. In order to simplify the MRAC design, we will assume the plant is second order by neglecting the pole at -100 since its dynamics are much faster then the remaining roots in the plant (see Fig. 2). We found via simulation that the use of this second order linear model has no significant affect on the low frequency responses compared to the original third order linear model. Hence, the transfer function of the system is rewritten as (note that the dc gain term k_p is changed accordingly)

$$\hat{P}(s) = \frac{\hat{y}(s)}{\hat{v}(s)} = \frac{k_p}{s^2 + a_{p2}s + a_{p1}} = \frac{k_p \hat{n}_p(s)}{\hat{d}_p(s)} \tag{5}$$

where $k_p = -1.022$, $\hat{n}_p(s) = 1$, $\hat{d}_p = s^2 + a_{p2}s + a_{p1} = s^2 - 65.33$. The reference model $\hat{M}(s)$ in Fig. 3 is used to specify the desired closed-loop system behavior. Here, the reference model is chosen to be

$$\hat{M}(s) = \frac{k_m}{s^2 + a_{m2}s + a_{m1}} = \frac{k_m \hat{n}_m(s)}{\hat{d}_m(s)} \tag{6}$$

where $k_m = -25$, $\hat{n}_m(s) = 1$, $\hat{d}_m = s^2 + a_{m2}s + a_{m1} = s^2 + 10s + 25$ (i.e., there are two poles at -5). This choice reflects our desire to have the closed-loop response with minimal overshoot, zero steady-state error, and yet a stable, fast response to a reference input. Moreover, to ensure that the "matching equality" is achieved (i.e., that there will exist a set of controller parameters that can achieve the behavior specified in $\hat{M}$) [9] we choose the order of the reference model to be the same as that of the plant.

The control has the form,

$$v = c_0 r + \frac{\hat{c}(s)}{\hat{\lambda}(s)} v + \frac{\hat{d}(s)}{\hat{\lambda}(s)} y \tag{7}$$

where $\hat{\lambda}(s) = \hat{\lambda}_0(s)\hat{n}_m(s) = s^2 + \lambda_2 s + \lambda_1 = (s + 20)^2$ is a monic function (where the value 20 is chosen since $\frac{1}{\hat{\lambda}(s)}$ filters v and y and a cut-off of 20 rad/sec will not attenuate most frequencies of interest). From the matching equality in [9] the values of the controller parameter c_0 and the polynomials $\hat{c}(s)$ and $\hat{d}(s)$ that result in the reference model response being achieved are c_0^*, $\hat{c}^*(s)$, and $\hat{d}^*(s)$ where

$$\begin{aligned} c_0^* &= \frac{k_m}{k_p}, \\ \hat{c}^* &= \hat{\lambda} - \hat{q}\hat{n}_p, \\ \hat{d}^* &= \frac{1}{k_p}(\hat{q}\hat{d}_p - \hat{\lambda}_0 \hat{d}_m) \end{aligned} \tag{8}$$

where $\hat{q}$ is the quotient of $\frac{\hat{\lambda}_0 \hat{d}_m}{\hat{d}_p}$, and the nominal controller parameters are (based on the matching equalities)

$$\begin{aligned} \hat{q}^* &= s^2 + 50s + 890.33, \\ c_0^* &= 24.45, \\ \hat{c}^* &= c_3^* s^2 + c_2^* s + c_1^*, \\ \hat{d}^* &= d_3^* s^2 + d_2^* s + d_1^* \end{aligned} \tag{9}$$

where $[c_1^* \; c_2^* \; d_1^* \; d_2^*] = [-490.33 \; -10 \; 66673.9 \; 8085.43] := \underline{\theta}^*$ and $\underline{\theta}$ denotes the controller parameters that are tuned.

Using the "certainty equivalence principle" (i.e., that the estimates of the plant parameters should be taken as the true values of the plant parameters and used to specify the controller) [9], the plant parameters are used, along with (8) to compute the controller parameters (see Fig. 3). Assume that the "identifier model" (i.e., the model that is adjusted so that it behaves like the plant) is

$$\hat{M}_i(s) = \frac{b_{i2}s + b_{i1}}{s^2 + a_{i2}s + a_{i1}} = \frac{\hat{n}_i(s)}{\hat{d}_i(s)}. \tag{10}$$

In order to share the signals between the identifier and the controller, the identifier model can be re-written as (see [9], Section II-A for more details)

$$\hat{y}_i(s) = \frac{b_{i2}s + b_{i1}}{\hat{\lambda}(s)}\hat{v}(s) + \frac{(\lambda_2 - a_{i2})s + (\lambda_1 - a_{i1})}{\hat{\lambda}(s)}\hat{y}(s). \tag{11}$$

Notice that $\hat{v}(s)$ is the input to the plant and $\hat{y}(s)$ is the output of the plant. The identifier structure is

$$\hat{y}_i(s) = \underline{\beta}'\underline{\hat{w}}^{(1)}(s) + \underline{\alpha}'\underline{\hat{w}}^{(2)}(s) \tag{12}$$

where $\underline{\beta} = [b_{i1}\ b_{i2}]'$, $\underline{\alpha} = [(\lambda_1 - a_{i1})\ (\lambda_2 - a_{i2})]'$, $\underline{\hat{w}}^{(1)}(s) = [(sI - \Lambda)^{-1}b_\lambda]\hat{r}(s)$, $\underline{\hat{w}}^{(2)}(s) = [(sI - \Lambda)^{-1}b_\lambda]\hat{y}(s)$, and $[(sI - \Lambda)^{-1}b_\lambda] = [\frac{1}{\hat{\lambda}(s)}\ \frac{s}{\hat{\lambda}(s)}]'$ in which $\Lambda = \begin{bmatrix} 0 & 1 \\ -\lambda_1 & -\lambda_2 \end{bmatrix}$ and $b_\lambda = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Further simplification will give the identifier output as

$$\hat{y}_i(s) = \underline{\pi}'\underline{\hat{w}}(s) \tag{13}$$

where $\underline{\pi}' = [\underline{\alpha}'\ \underline{\beta}']$, and $\underline{\hat{w}}'(s) = [\underline{\hat{w}}^{(1)'}(s)\ \underline{\hat{w}}^{(2)'(s)}]$. It is assumed that the initial conditions of the "observer" (i.e., the box labeled "filter" in Fig. 3) are chosen to be zero. Note that $\underline{\pi}$ contains the parameters that will be determined through the identifier, and based on the plant model it is expected that the unknown parameters $\underline{\pi}$ converge to $\underline{\pi}^* = [k_p\ 0\ (\lambda_1 - a_{p1})\ (\lambda_2 - a_{p2})] = [24.45\ 0\ 465.33\ 40]$. The adaptation mechanism will use the identifier output error $e_i = y_i - y$ in the "normalized gradient algorithm with projection" to update the parameter $\underline{\pi}$ so that

$$\dot{\underline{\pi}} = -g\frac{e_i\underline{w}}{1 + \gamma\underline{w}'\underline{w}} \tag{14}$$

where $g > 0$ is an adaptation gain, $\gamma > 0$ is a scaling factor, and the projection rule is

If $b_{i1} = k_{\min}$ **and** $\dot{b}_{i1} > 0$, **then** let $\dot{b}_{i1} = 0$

where $k_{\min}$ is chosen to be -0.001. For each new estimate of the plant parameters π, the controller parameter $\underline{\theta}$ is then updated by using the matching equality as shown in Fig. 3. The parameter error is denoted by $\underline{\phi} = \underline{\pi} - \underline{\pi}^*$ and it is expected to approach zero as $t \to \infty$. As the parameter error $\underline{\phi}$ goes to zero, it is expected the output error e_o approaches zero provided that the plant is exactly the second order linear plant[2], as proven in [9].

[2]Note that when this MRAC is used on a nonlinear model, there is no guarantee of convergence.

Since the plant is assumed to be second order, based on the theory of persistency of excitation [9], the identifier parameters will converge to their true values if an input which is "sufficiently rich" of at least to the order of twice the order of the system. Therefore, an input composed as the sum of two sinusoids will be used to obtain richness of order four according to the theory. In order to pick the two sinusoids as the input, it would be beneficial to study the frequency response of the plant model. The way to pick the inputs is that the frequency selected should be able to excite most of the frequency range we are interested in. A Bode plot of the third order linear system suggested that the cut-off frequency (3 dB cut-off) of the plant is about 6 $\frac{\text{rad}}{\text{s}}$. Hence, we picked two sinusoids (1 $\frac{\text{rad}}{\text{s}}$ and 10 $\frac{\text{rad}}{\text{s}}$) to cover the most critical frequency range. The amplitude of the input is chosen to force the system, as well as the reference model, to swing approximately between ± 0.05 m around the initial ball position (i.e., at 0.15 m, where the total length of the system is 0.3 m); hence, we choose $r(t) = 0.05(\sin(1t) + \sin(10t))$. Note that this input will drive the system into different operating conditions where the plant behavior will change due to the nonlinearities.

Next, the adaptive controller will be simulated with two different plant models to demonstrate the closed-loop performance. The two plant models used are: (i) the second order linear model, and (ii) the original nonlinear system (i.e., (2)).

Second Order Linear Plant: In this section, the adaptive controller will be simulated with the second-order linear model of the ball suspension system which was used to design the MRAC. The initial position of the ball is at 0.15 m. In order to speed up the adaptation, the initial condition of the identifier is chosen to be $\underline{\pi} = [-1\ 1\ 460\ 40]$ (i.e., we assume that a_{p2} is known to be zero, that we have a good guess of the value of a_{p1}, but that we have no good idea of the value of k_p). With these initial parameters, after some tuning we chose the adaptation gain $g = 10000$ and $\gamma = 1$. The time required for the adaptive mechanism to track the reference model is approximately 100 s; hence we did not include plots showing the results. This time lag is definitely not acceptable for such a fast dynamical system (i.e., it is expected that the adaptive system should be able to change in at most a few seconds). But this slow adaptation only happens on one of the parameters a_{i1} (where $a_{i1}^* = 465.33$). Therefore, a vector adaptation gain is used to force the parameter a_{i1} to change faster. After some simulation-based investigations, it was found that a vector gain $\underline{g}' = [10000\ 10000\ 100000000\ 10000]'$ (such that $\underline{g}'e_i\underline{w}/(1 + \gamma\underline{w}'\underline{w})$ is used in (14)) would significantly enhance the speed of the adaptive system. As shown in Fig. 4, the identifier error e_i is approaching zero in 0.5 seconds, while the plant output error e_o is still slowly converging after 20 s (i.e., swinging between ± 0.005 m) but $y(t)$ is capable of matching the response specified by the reference model in about 15 s. Notice that there is a fairly large transient period in the first 2 s when both the identifier and the controller parameters are varying widely. We see that the system response is approaching the one specified by the reference model, but the convergence rate is quite slow even with relatively large adaptation gains in the vector $\underline{g}$. Note

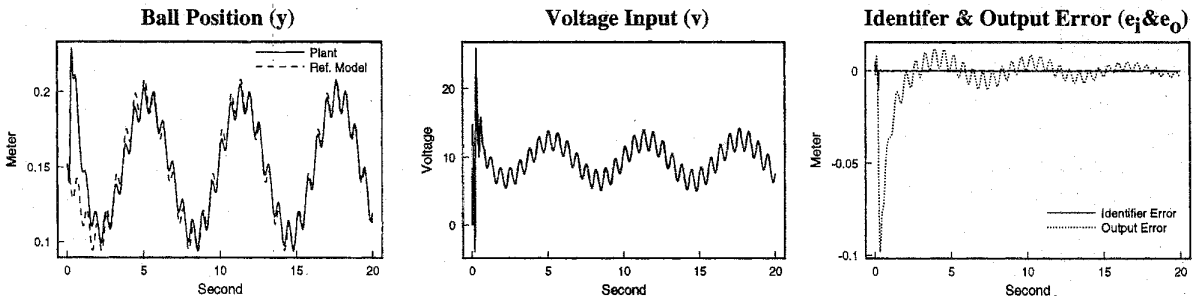


Fig. 4. Responses using MRAC design (reduced order linear model, sinusoidal input sequence).

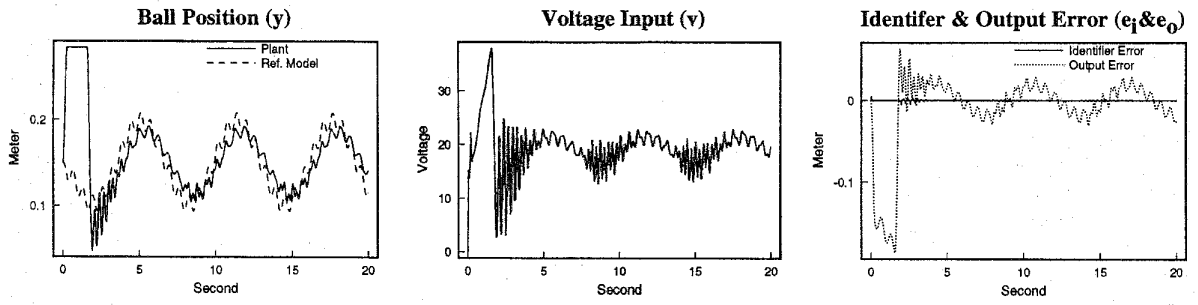


Fig. 5. Responses for MRAC design (nonlinear model, sinusoidal input sequence).

that the voltage input v in Fig. 4 is of acceptable magnitude compared with the implementation in [48]; in fact all control strategies studied in this paper produced acceptable voltage control inputs to the plant compared to [48].

Non-Linear System: In this section, the adaptive controller will be simulated with the nonlinear model of the ball suspension system with the same controller and initial conditions so that the ball starts at 0.15 m. Fig. 5 shows the responses for the nonlinear model. It is observed that the ball first drops to the ground level since the adaptation mechanism is slow and it cannot keep up with the fast-moving system. After about 2.5 s, the system starts to recover and tries to keep up with the plant. The identifier error e_i dies down after about 5 s where it swings between ± 0.001 m; however, the plant output error swings between ± 0.03 m and appears to maintain at the same level (i.e., the plant output never perfectly matches that of the reference model). The plant output is not capable of matching the one specified by the reference model mainly because the indirect adaptive controller is not designed for the nonlinear model. It is also observed that the ball position reacts better in the range where the ball is close to the ground level (0.3 m), whereas the response gets worse in the range where the ball is close to the magnetic coil (0 m) (i.e., the nonzero identifier error is found and the control input is more oscillatory in the instant when the ball position is closer to 0 m). This behavior is due to the nature of the nonlinear plant, where the system dynamics vary significantly with the ball position, and the adaptive mechanism is not fast enough to adapt the controller parameters with respect to the system dynamics.

In order to keep the ball from falling to the ground level or lifting up to the coil, one approach is to apply the previously adapted controller parameters to initialize the adaptive controller. It is hoped that this initialization process would help the adaptation mechanism to keep up with the plant dynamics at the beginning of the simulation. As shown in Fig. 6 when this approach is employed, the ball does not fall to the ground level (compared to Fig. 5). Despite the fact that the system appears to be stable, the identifier error does not approach zero and swings between ± 0.001 m and the plant output error swings between ± 0.03 m (i.e., the closed-loop response of the plant is still not matching that of the reference model).

Note that from all the simulations, due to the use of the gradient type update the MRAC seems to be slow in general[3]. Although it is faster with the use of very large adaptation gains, it is obvious that large adaptation gains will be problematic because of sensitivity to noise. In addition, the MRAC designed with a linear, second order model does not perform adequately with the nonlinear plant, which is due to the variation in plant dynamics. If other input sequences are used, such as those with higher order of richness (more sinusoidal components at different frequencies), the convergence of the control may be improved. However, a complicated reference input for a nonlinear plant may result in excess plant variations such that it is even harder for the plant to follow a reference model. If a step input sequence is used as the reference input to the nonlinear plant as shown in Fig. 7, the MRAC does a very poor job of following the reference model.

[3] We also investigated MRAC with different adaptation algorithms, such as normalized least squares algorithm with convariance resetting [9]. Although the simulation results for this least squares type MRAC show a slightly faster adaptation speed for the linear plant model, the simulation results appear to be less satisfactory than those of the gradient update method when the nonlinear plant model is used. It is for this reason, and due to space constraints, that we did not include our results for the MRAC with least squares update mechanism.

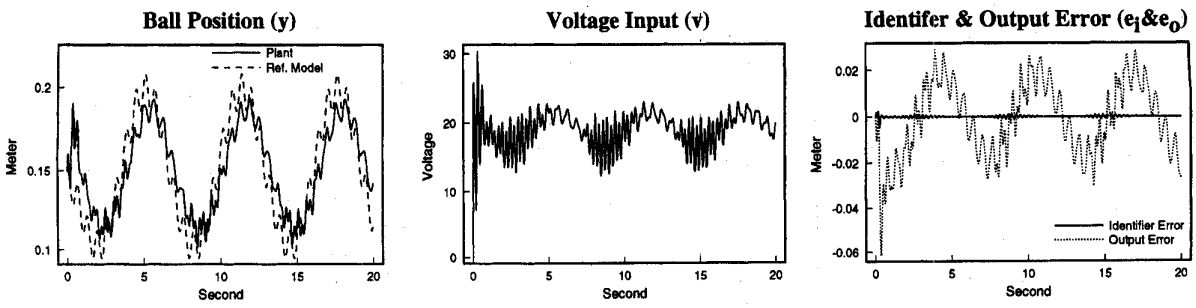


Fig. 6. Responses for MRAC design after "training" (nonlinear model, sinusoidal input sequence).

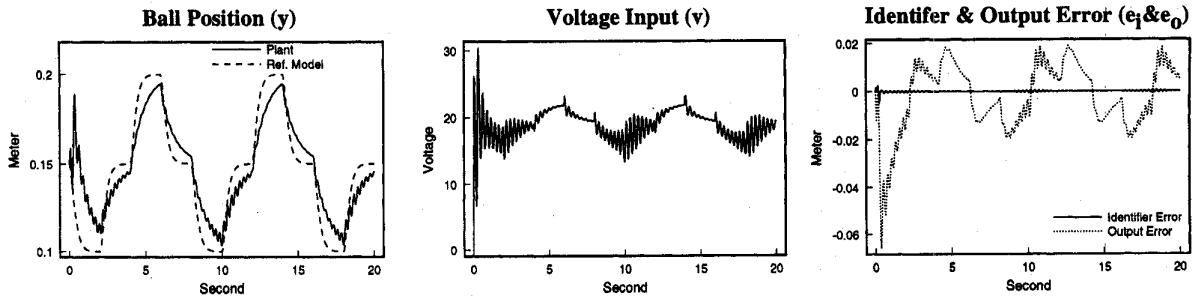


Fig. 7. Responses for MRAC (nonlinear model, step input sequence).

Also note that since the magnetic ball suspension system is feedback linearizable, it is possible to design a stable adaptive controller for this nonlinear system (see Ch. 7 in [9]). However, since the nonlinear model of the magnetic ball suspension system has a "relative degree" of three, using the approach in Section 7.3 of [9] results in an adaptive controller of significant complexity (with a high dimension regressor and parameter vectors). In addition, the approach in [9] only works for a very special structure for the nonlinear plant that restricts the unknown parameters to enter linearly. It is for these reasons that we investigate the use of fuzzy model reference learning control (FMRLC) for the magnetic ball suspension system.

C. Fuzzy Model Reference Learning Control

In this section, the FMRLC shown in Fig. 8, which was introduced in [1]–[5], will be designed for the magnetic ball suspension system. Note that the design of FMRLC does not require the use of a linear plant model, and thus from now on we will always use the nonlinear model of the magnetic ball suspension system. The fuzzy controller in Fig. 8 uses the error signal[4] $e(kT) = r(kT) - y(kT)$ and the change in error of the ball position $c(kT) = \frac{e(kT)-e(kT-T)}{T}$ to decide what voltage to apply so that $y(kT) \to r(kT)$ as $k \to \infty$. The learning mechanism in Fig. 8 is used to (i) observe data from the fuzzy control system, (ii) characterize its current performance, and (iii) automatically synthesize and/or adjust the fuzzy controller so that some pre-specified performance objectives are met. These performance objectives are characterized via the *reference model* shown in Fig. 8. In a manner analogous to conventional adaptive control, where conventional controllers are adjusted, the learning mechanism seeks to adjust the fuzzy controller so that the closed-loop system (the map from $r(kT)$ to $y(kT)$ where T is the sampling period) acts like a pre-specified reference model (the map from $r(kT)$ to $y_m(kT)$). Next we describe each component of the FMRLC in Fig. 8.

The Fuzzy Controller: In fuzzy control theory, the range of values for a given controller input or output is often called the "universe of discourse" [47], [49]. Often, for greater flexibility in fuzzy controller implementation, the universes of discourse for each process input are "normalized" to the interval $[-1, 1]$ by means of constant scaling factors. For our fuzzy controller design, the gains g_e, g_c, and g_v were employed to normalize the universe of discourse for the error $e(kT)$, change in error $c(kT)$, and controller output $v(kT)$, respectively. The gain g_e is chosen so that the range of values of $g_e e(kT)$ lie on $[-1, 1]$ and g_v is chosen by using the allowed range of inputs to the plant in a similar way. The gain g_c is determined by experimenting with various inputs to the system to determine the normal range of values that $c(kT)$ will take on; then g_c is chosen so that this range of values is scaled to $[-1, 1]$. According to this procedure, the universes of discourse of the inputs to the fuzzy controller $e(t)$ and $c(t)$ are chosen to be $[-0.275, 0.275]$ and $[-2.0, 2.0]$ respectively. This choice is made based on the distance between the coil and ground level of the magnetic ball suspension system and an estimate of the maximum attainable velocity of the ball that we obtain via simulations. Thus, the gains g_e and g_c are $\frac{1}{0.275}$ and $\frac{1}{2}$, respectively. The output gain g_v is then chosen to be 30, which is the maximum voltage we typically would like to apply to the plant.

We utilize one multiple-input, single-output (MISO) fuzzy controller, which has a knowledge-base of **IF-THEN** control

[4]Notice that we use sampled versions of all signals as the operation of the FMRLC is easier to explain and visualize in discrete-time.

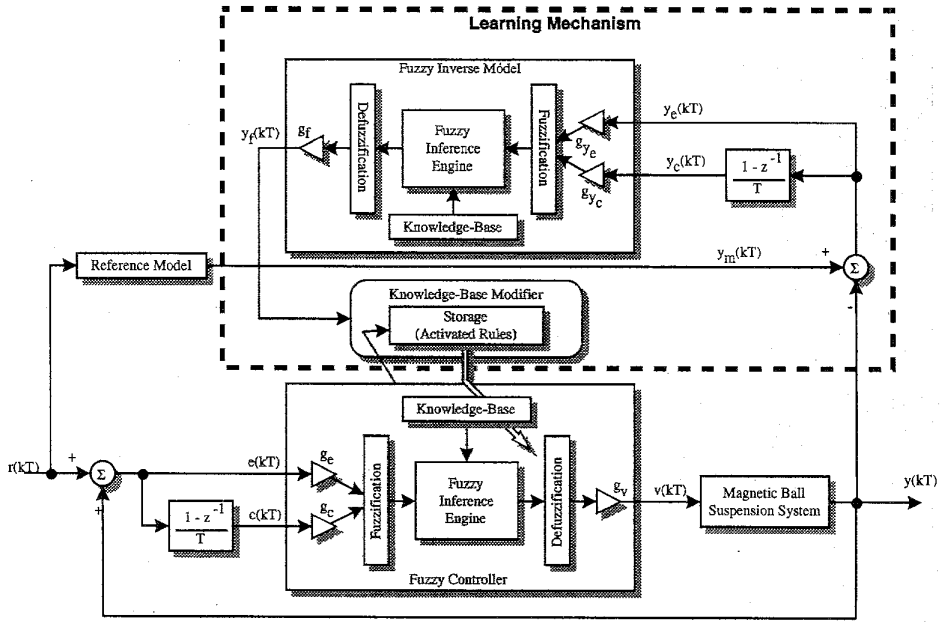


Fig. 8. An FMRLC for the magnetic ball suspension system.

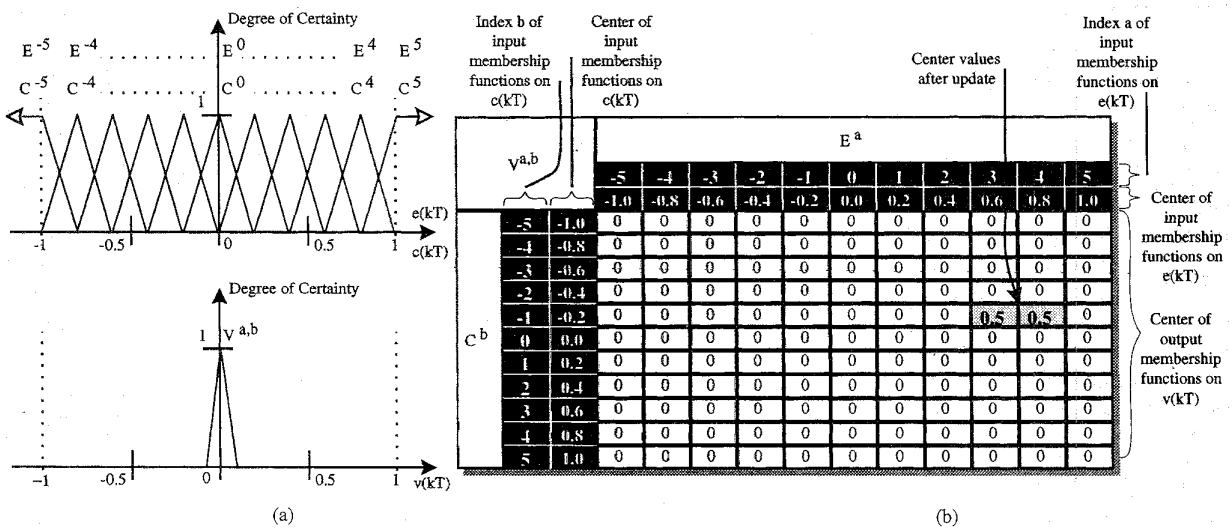
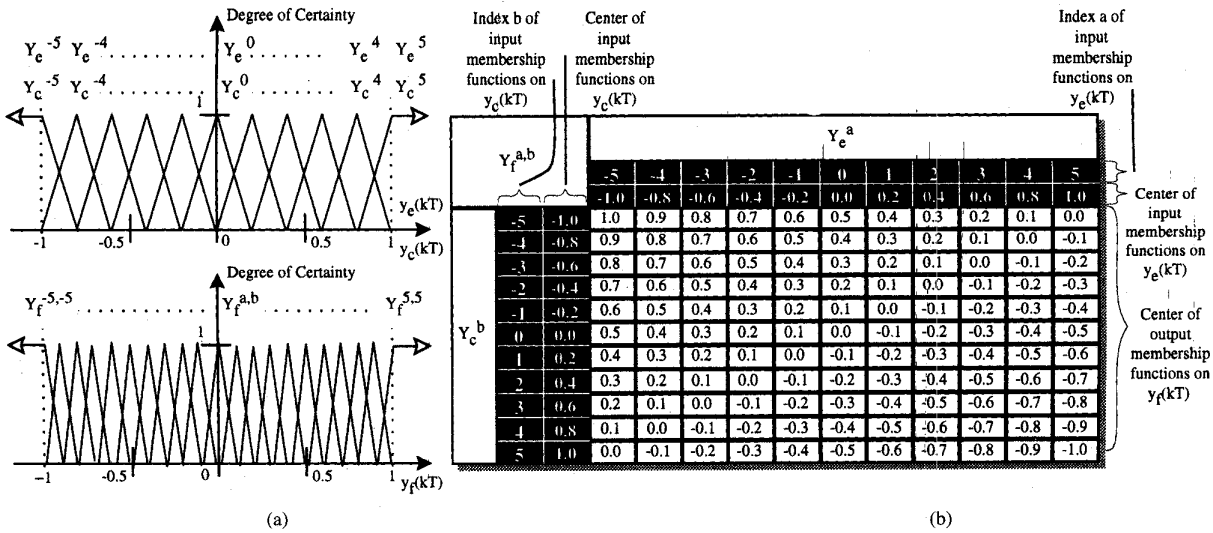


Fig. 9. Input-output universes of discourse and rule-base for the fuzzy controller.

rules of the form **If** $\tilde{e}$ is $\tilde{E}^a$ **and** $\tilde{c}$ is $\tilde{C}^b$ **Then** $\tilde{v}$ is $\tilde{V}^{a,b}$, where $\tilde{e}$ and $\tilde{c}$ denote the *linguistic variables* associated with controller inputs $e(kT)$ and $c(kT)$, respectively, $\tilde{v}$ denotes the linguistic variable associated with the controller output v, $\tilde{E}^a$ denotes the ath *linguistic value* associated with $\tilde{e}$ and $\tilde{C}^b$ denotes the bth *linguistic value* associated with $\tilde{c}$, respectively, and $\tilde{V}^{a,b}$ denotes the *consequent linguistic value* associated with $\tilde{v}$. For example, one fuzzy control rule could be: **If** *Error* is *PositiveLarge* **and** *ChangeInError* is *NegativeSmall* **Then** *PlantInput is PositiveBig*, (in this case $\tilde{e}$ = "*Error*," $\tilde{E}^4$ = "*PositiveLarge*," etc.). A set of such rules forms the "rule-base" which characterizes how to control a dynamical system. The above control rule may be quantified by utilizing fuzzy set theory to obtain a fuzzy implication of the form: **If** E^a **and** C^b **Then** $V^{a,b}$, where E^a, C^b, and $V^{a,b}$ denote the fuzzy sets that quantify the *linguistic statements* "$\tilde{e}$ is $\tilde{E}^a$," "$\tilde{c}$ is $\tilde{C}^b$," and "$\tilde{v}$ is $\tilde{V}^{a,b}$," respectively. We chose to use 11 fuzzy sets (triangular membership function with base widths of 0.4) on the normalized universes of discourse for $e(kT)$ and $c(kT)$ as shown in Fig. 9(a).

Assume that we use the same fuzzy sets on the $c(kT)$ normalized universes of discourse (i.e., $C^b = E^a$). The membership functions on the output universe of discourse are assumed to be unknown; they are what the FMRLC will automatically synthesize. As shown in Fig. 9(a), we initialize the fuzzy controller knowledge-base with 121 rules (using all possible combinations of rules) where all the right-hand-side membership functions are triangular with base widths



$Y_f^{a,b}$	Y_e^a -5	-4	-3	-2	-1	0	1	2	3	4	5
Y_c^b	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	0.2	0.4	0.6	0.8	1.0
-5 -1.0	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0
-4 -0.8	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0	-0.1
-3 -0.6	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0	-0.1	-0.2
-2 -0.4	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0	-0.1	-0.2	-0.3
-1 -0.2	0.6	0.5	0.4	0.3	0.2	0.1	0.0	-0.1	-0.2	-0.3	-0.4
0 0.0	0.5	0.4	0.3	0.2	0.1	0.0	-0.1	-0.2	-0.3	-0.4	-0.5
1 0.2	0.4	0.3	0.2	0.1	0.0	-0.1	-0.2	-0.3	-0.4	-0.5	-0.6
2 0.4	0.3	0.2	0.1	0.0	-0.1	-0.2	-0.3	-0.4	-0.5	-0.6	-0.7
3 0.6	0.2	0.1	0.0	-0.1	-0.2	-0.3	-0.4	-0.5	-0.6	-0.7	-0.8
4 0.8	0.1	0.0	-0.1	-0.2	-0.3	-0.4	-0.5	-0.6	-0.7	-0.8	-0.9
5 1.0	0.0	-0.1	-0.2	-0.3	-0.4	-0.5	-0.6	-0.7	-0.8	-0.9	-1.0

Fig. 10. Input-output universes of discourse and the rule-base for the fuzzy inverse model.

of 0.2 and centers at zero. This is done to model the fact that the fuzzy controller initially knows nothing about how to control the plant. In conventional direct fuzzy controller development the designer specifies a set of such control rules where $V^{a,b}$ are also specified *a priori*; for the FMRLC, the system will automatically specify and/or modify the fuzzy sets $V^{a,b}$ to improve/maintain performance as is explained next. Note that we use singleton fuzzification, minimum to quantify the premise and implication, and the standard center-of-gravity (COG) defuzzification technique [47], [49].

The Reference Model: The reference model provides a means for quantifying the desired performance. In general, the reference model may be any type of dynamical system (linear or nonlinear, time-invariant or time-varying, discrete or continuous time, etc.). Here, we use a discretized version of the same reference model as was used for the MRAC. The performance of the overall system is computed with respect to the reference model by generating error signals $y_e(kT) = y_m(kT) - y(kT)$ and $y_c(kT) = \frac{y_e(kT) - y_e(kT-T)}{T}$ shown in Fig. 8. Given that the reference model characterizes design criteria such as rise time and overshoot, and that the input to the reference model is the reference input $r(kT)$, the desired performance of the controlled process is met if the learning mechanism forces $y_e(kT)$ and $y_c(kT)$ to remain very small for all time; hence, $y_e(kT)$ and $y_c(kT)$ provide a characterization of the extent to which the desired performance is met at time kT. If the performance is met $(y(kT) \approx 0)$ then the learning mechanism will not make significant modifications to the fuzzy controller. On the other hand if $y_e(kT)$ and $y_c(kT)$ are big, the desired performance is not achieved and the learning mechanism must adjust the fuzzy controller. Next we describe the operation of the learning mechanism.

The Learning Mechanism: As previously mentioned, the learning mechanism performs the function of modifying the knowledge-base of a direct fuzzy controller so that the closed-loop system behaves like the reference model. These knowledge-base modifications are made by observing data from the controlled process, the reference model, and the fuzzy controller. The learning mechanism consists of two parts: a fuzzy inverse model and a knowledge-base modifier. The fuzzy inverse model performs the function of mapping $y_e(kT)$ and $y_c(kT)$ (representing the deviation from the desired behavior), to changes in the process input $y_f(kT)$ that are necessary to force $y_e(kT)$ and $y_c(kT)$ to zero. The knowledge-base modifier performs the function of modifying the fuzzy controller's knowledge-base to affect the necessary changes in the process inputs.

The authors in [1]–[5] introduced the idea of using a fuzzy system to map $y_e(kT)$ and $y_c(kT)$ (and possibly functions of $y_e(kT)$, or process operating conditions), to the necessary changes in the process inputs $y_f(kT)$. This map is called the *fuzzy inverse model* since information about the plant inverse dynamics is used in its specification. Note that similar to the fuzzy controller, the fuzzy inverse model shown in Fig. 8 contains normalizing scaling factors, namely g_{y_e}, g_{y_c}, and g_f, for each universe of discourse. Given that $g_{y_e} y_e$ and $g_{y_c} y_c$ are inputs to the fuzzy inverse model, the knowledge-base for the fuzzy inverse model associated with the process input is generated from fuzzy implications of the form **If** Y_e^a and Y_c^b **Then** $Y_f^{a,b}$, where Y_e^a denotes the ath fuzzy set for the error y_e and Y_c^b denotes the bth fuzzy set for the change in error y_c, respectively, and $Y_f^{a,b}$ denotes the consequent fuzzy set for this rule describing the necessary change in the process input. As with the fuzzy controller, we often utilize membership functions for the normalized input universes of discourse as shown in Fig. 10 (we use $Y_c^b = Y_e^a$), triangular membership functions for the output universe of discourse, singleton fuzzification, minimum to quantify the premise and implication, and COG defuzzification. The fuzzy inverse model is then set up similar to the fuzzy controller except that the membership function of the output is initialized as shown in Fig. 10 to represent the inverse dynamics of the plant. The rule-base is chosen so that it represents the

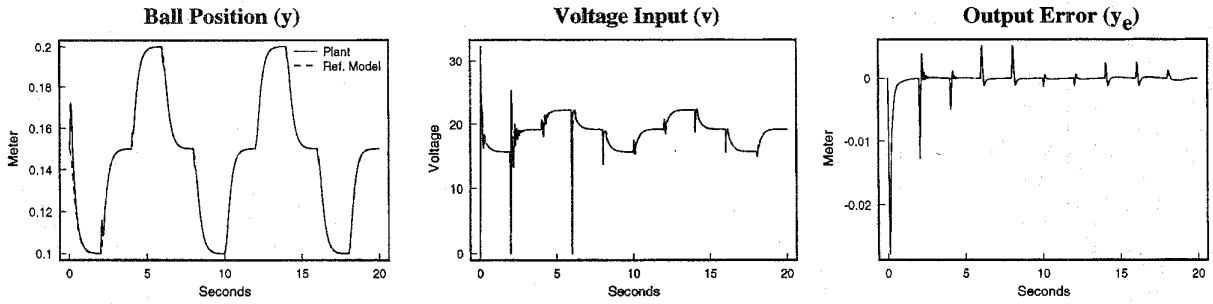


Fig. 11. Responses for FMRLC (step input sequence).

knowledge of how to update the controller when the error, change of error between the reference model, and the plant output is given. The gains of the fuzzy inverse model are then initially chosen to be $g_{y_e} = \frac{1}{0.275}$, $g_{y_c} = 0.5$, and $g_f = 30$. Note that all the gains are chosen based on the physical properties of the plant, so that $g_{y_e} = g_e$, $g_{y_c} = g_c$, and $g_f = g_v$ (more details on the rationale and justification for this choice for the gains is provided in [1]–[5]). Successful design of the fuzzy inverse model has been performed for many applications including a cargo ship steering problem [2], an inverted pendulum [1], anti-skid braking systems [3], [4], a rocket velocity control problem and a rigid robot [5], aircraft control [31], and a flexible robot [29], [30].

Given the information about the necessary changes in the input as expressed by $y_f(kT)$, the *knowledge-base modifier* (as shown in Fig. 8) changes the knowledge-base of the fuzzy controller so that the previously applied control action will be modified by the amount $y_f(kT)$. Therefore, consider the previously computed control action $v(kT - T)$, which contributed to the present good/bad system performance. Note that $e(kT - T)$ and $c(kT - T)$ would have been the process error and change in error, respectively, at that time. By modifying the fuzzy controller's knowledge-base we may force the fuzzy controller to produce a desired output $v(kT - T) + y_f(kT)$. Assume that only symmetric membership functions are defined for the fuzzy controller's output so that $v_c^{a,b}(kT)$ denotes the center value of the membership function at time kT associated with the fuzzy set $V^{a,b}$ (initially, all centers are at zero, $v_c^{a,b}(0) = 0$). Knowledge-base modification is performed by shifting centers of the membership functions of the fuzzy sets $V^{a,b}$ which are associated with the fuzzy implications that contributed to the previous control action $v(kT-T)$. The degree of contribution for a particular fuzzy implication whose fuzzy relation is denoted $R^{a,b}$ is determined by its "activation level," defined as $\bar{\delta}^{a,b}(t) = \min\{\mu_{E^a}(e(t)), \mu_{C^b}(c(t))\}$, where μ_A denotes the membership function of the fuzzy set A and $t = kT$ is the current time. Only those rules with nonzero activation level are modified; all others remain unchanged. This modification involves shifting these membership functions by an amount specified by $y_f(kT)$ so that

$$v_c^{a,b}(kT) = v_c^{a,b}(kT - T) + y_f(kT). \tag{15}$$

It is important to note that our rule-base modification procedure implements a form of *local* learning and hence utilizes memory. In other words, different parts of the rule-base are "filled in" based on different operating conditions for the system, and when one area of the rule-base is updated, other rules are not affected. Hence, the controller adapts to new situations and also remembers how it has adapted to past situations [1], [2], [50].

Continuing with our example above, assume that all the normalizing gains for both the direct fuzzy controller and the fuzzy inverse model are unity and that the fuzzy inverse model produces an output $y_f(kT) = 0.5$ indicating that the value of the output to the plant at time $kT - T$ should have been $v(kT - T) + 0.5$ to improve performance (i.e., to force $y_e \approx 0$). Next, suppose that $e(kT - T) = 0.75$ and $c(kT - T) = -0.2$. Then, the rules **If** E^3 **and** C^{-1} **Then** $V^{3,-1}$ and **If** E^4 **and** C^{-1} **Then** $V^{4,-1}$ are the only rules with nonzero activation levels ($\bar{\delta}^{3,-1} = 0.25$ and $\bar{\delta}^{4,-1} = 0.75$). Hence, these are the only rules that have their consequent fuzzy sets ($V^{3,-1}$, $V^{4,-1}$) modified (see Fig. 9(b)). To modify these fuzzy sets we simply shift their centers according to (15). Next, we apply the FMRLC to the ball-suspension control problem.

According to the design procedure in [1]–[5], a step input can be used to tune the gains g_c and g_{y_c} of the FMRLC. Here, we chose a step response sequence. Notice in the ball position plot in Fig. 11 that the FMRLC design was quite successful in generating the control rules such that the ball position tracks the reference model almost perfectly. It is important to note that the FMRLC design here required no iteration on the design process. This is not necessarily true in general and some tuning is often needed for different applications (see the applications in [1]–[5] and [29], [30]).

Up to this point, the FMRLC seems to be a very efficient control algorithm for a wide variety of nonlinear systems (see, [1]–[5]). However, there currently exists no mathematical evaluation of the robustness and stability properties of the FMRLC. It is possible that there exists an input sequence which will cause the FMRLC to fail since stability of the FMRLC depends on the input (as it does for all nonlinear systems). For example, if the sinusoidal input sequence $r(t) = 0.05(\sin(1t) + \sin(10t))$ is used (as it was used in the adaptive controller design), the plant response is unstable as shown in Fig. 12. Although exhaustive tuning of the gains (except g_e, g_v, g_{y_e}, and g_f since we consider these to be set by the physical

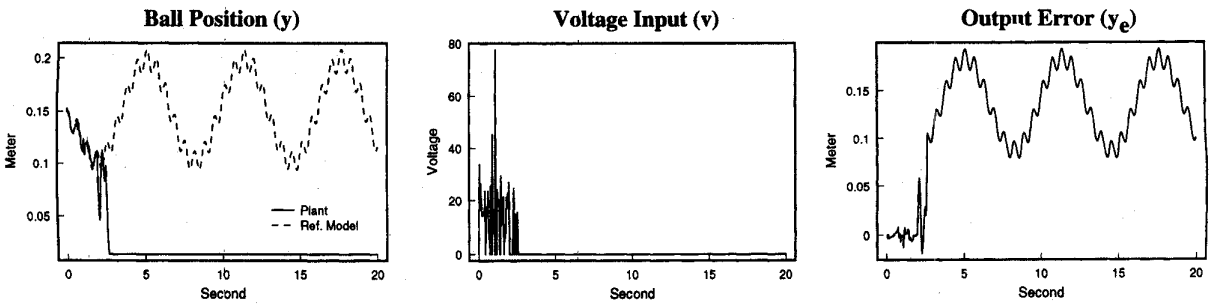
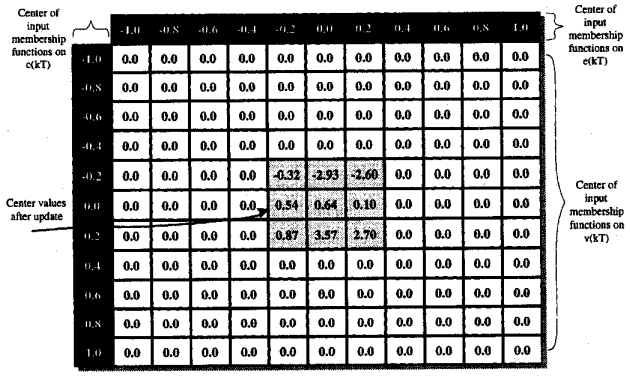


Fig. 12. Responses for FMRLC (sinusoidal input sequence).



	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	0.2	0.4	0.6	0.8	1.0
-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-0.2	0.0	0.0	0.0	0.0	-0.32	-2.93	-2.60	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.54	0.64	0.10	0.0	0.0	0.0	0.0
0.2	0.0	0.0	0.0	0.0	0.87	3.57	2.70	0.0	0.0	0.0	0.0
0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Fig. 13. Rule-base of the learned fuzzy controller (step input sequence).

system) are performed to improve the FMRLC, Fig. 12 indeed shows one of the best responses we can obtain.

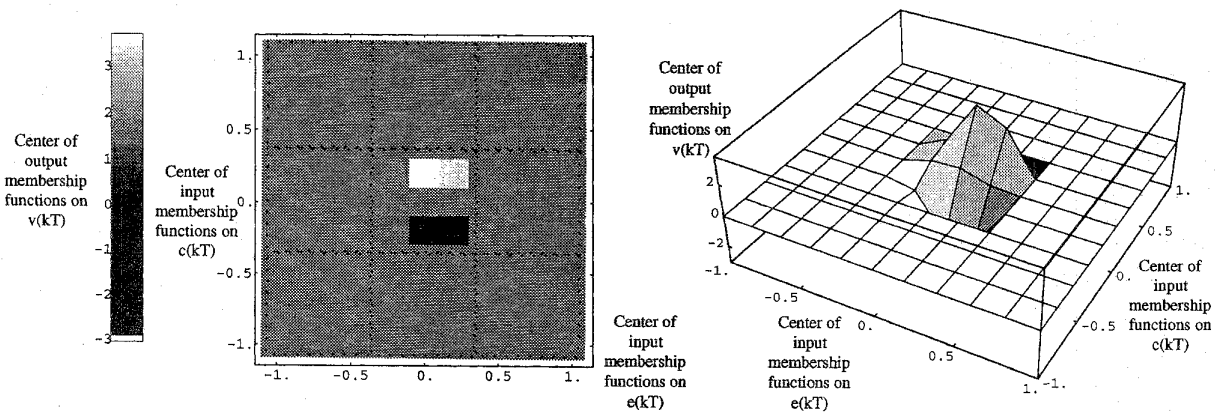
D. Motivation for Dynamically Focused Learning (DFL)

With the results as shown in Fig. 12, one would ask: *What are the effects of different reference inputs*? In the conventional adaptive controller design for linear plants with unknown but constant coefficients, the theory of persistence of excitation provides some guidelines for selecting an input with "sufficient richness" so that the adaptive controller will be fully excited and hence capable to identify the appropriate parameters. There is no such theory established for the FMRLC. As was found in [1]–[5] the quality of the design of the FMRLC depends on the successful tuning of the gains g_e, g_c and g_v for the controller and g_{y_e}, g_{y_c} and g_f for the fuzzy inverse model. Moreover, there is an underlying assumption that the premises of the rules totally cover the operating range of the system and have a sufficient number and appropriate distribution of membership functions on the input universes of discourse. As was shown in [5], performance of the FMRLC depends on the number of rules used in the fuzzy controller where if too few rules are used then oscillatory behavior can result. To gain better insight into why the FMRLC fails, in Fig. 13 we show the learned rule-base of the fuzzy controller in the FMRLC (after the step input sequence[5] in Fig. 11 is applied to the system for 20 seconds). The rule-base is zero everywhere except the center nine rules. For better visualization of the rule-base, Fig. 14(a) and (b) show two different graphical representations of the rule-base in Fig. 13. Fig. 14(a) is a "density plot" of the rule-base, where the shade of gray indicates the value of the center of the output membership functions. This density plot or "rule-base map" shows that the fuzzy controller actually only utilized 9 of the 121 possible rules. In fact, if the rule-base is sub-divided into 9 sections as shown in Fig. 14(a), the 9 rules that are learned lie within the center section. With such a small number of rules, the learning mechanism of the FMRLC performed inadequately because the resulting control surface can only capture very approximate control actions. In the other words, for more complicated control actions, such a rule-base may not be able to force the plant to follow the reference model closely.

Fig. 14(b) is a 3-D surface plot of the same rule-base (which is similar to the nonlinear "control surface" of the fuzzy controller), since it shows the values of the centers of the output membership functions for fuzzy sets $V^{a,b}$, plotted versus the centers of the input membership functions of E^a and C^b on the x- and y-axes. Note that if fuzzy inference and defuzzification were used, Fig. 14(b) would show exactly how the fuzzy system interpolates to produce the true nonlinear control surface. The control surface as shown in Fig. 14(b) is nonsymmetric (i.e., the maximum of the output is 3.57 and the minimum is -2.93). This is because the control laws for moving the ball upward and downward are different as they vary with the ball position (i.e., the input $e(kT)$ to the fuzzy controller).

To improve FMRLC performance, one possible solution is to redesign the controller so that the rule-base has enough membership functions at the center where the most learning is needed. Yet, this approach will not be considered because the resulting controller will then be limited to a specific range of the inputs that happen to have been generated for the particular reference input sequence. Another possible solution is to increase the number of rules (by increasing the number of membership functions on each input universe of discourse) used by the fuzzy controller. Therefore, the total number of rules (for all combinations) is also increased, and we enhance the capability of the rule-base to memorize more distinct control actions (i.e., to achieve "fine control"). For instance, if we increase the number of membership functions on each input universe of discourse from 11 to, say 101 (but keeping

[5]The rule-base of the fuzzy controller for the sinusoidal input sequence is not used since it is filled with extreme values as the system is unstable.



(a) (b)

Fig. 14. (a) Control surface and (b) rule-base map of the fuzzy controller (step input sequence).

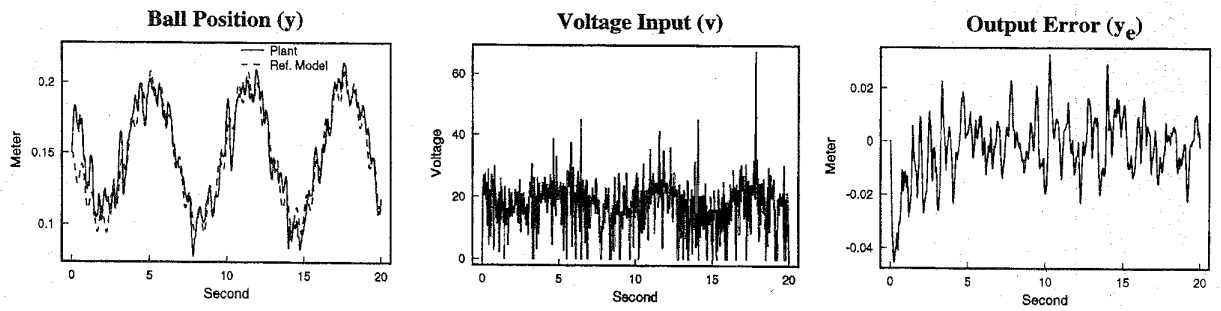


Fig. 15. Responses for FMRLC (nonlinear model, sinusoidal input sequence).

all other parameters, such as the scaling gains, the same), the total number of rules will increase from 121 to 10201 (i.e, there are two orders of magnitude increase in the number of rules)[6], and we get the responses shown in Fig. 15 for the FMRLC. Clearly as compared to Fig. 12, we have drastically improved the performance of the FMRLC to the extent that it performs similar to the MRAC for the nonlinear model (see Fig. 6). Notice that in Fig. 15 the output error swings between ± 0.027 even after 15 s of simulation, and the plant output is oscillatory. Longer simulations have shown that this FMRLC appears to be stable but the plant cannot perfectly follow the response of the reference model.

Even though we were able to significantly improve performance, enlarging the rule-base has many disadvantages: (i) the number of rules increases exponentially for an increase in membership functions and increases even faster with more inputs to the fuzzy controller[7], (ii) the computational efficiency drastically decreases as the number of rules increases, and (iii) a rule-base with a large number of rules will require a long time period for the learning mechanism to fill in the correct control laws since smaller portions of the rule-base map in Fig. 14(b) will be updated by the FMRLC for a higher granularity rule-base. Hence, the advantages of increasing the number of rules will soon be offset by practical implementation considerations and possible degradations in performance.

This motivates the need for special enhancements to the FMRLC so that: (i) we can minimize the number of membership functions and therefore rules used, and (ii) at the same time maximize the granularity of the rule-base near the point where the system is operating (e.g., the center region of the rule-base map in Fig. 14(a)) so that very effective learning can take place. In the next section we introduce the idea of "dynamically focused learning" that seeks to allocate rules to the learning process in an efficient manner.

III. Fuzzy Model Reference Learning Control with DFL

In order to avoid an excessive number of rules, this section first discusses an alternative view of a fuzzy rule-base and then presents three alternative approaches to perform "dynamically focused learning" (DFL) for the FMRLC, where the rule-base is "focused" onto the current region of operation so that a smaller rule-base can be used.

A. FMRLC Learning Dynamics

To begin we clarify several issues in FMRLC learning dynamics including: (i) the effects of gains on linguistics,

[6] We chose this number of membership functions by trial and error and found that further increases in the number of membership functions had very little effect on performance.

[7] The maximum number of rules for a MISO fuzzy system can be found as $\prod_{i=1}^{n} N_i$ where n is the total number of inputs and N_i is the number of membership functions on the ith input universe of discourse.

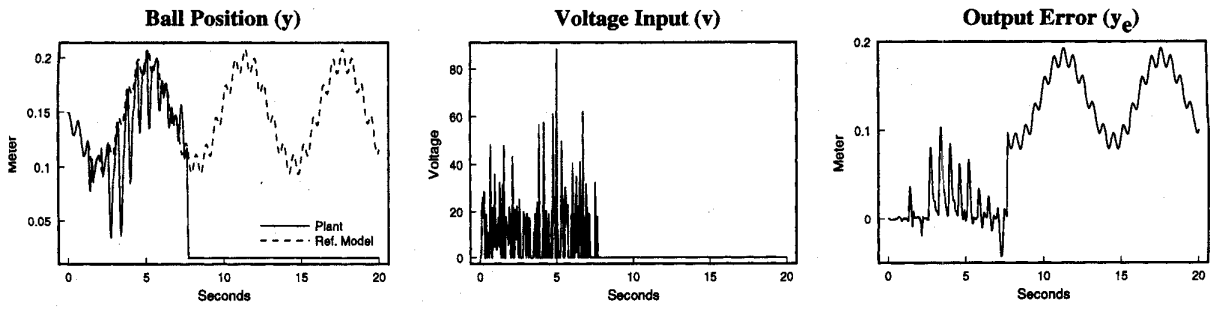


Fig. 16. Responses for FMRLC with reduced rule-base and no DFL (sinusoidal input sequence).

and (ii) characteristics of the rule-base such as granularity, coverage, and the control surface. The fuzzy controller in the FMRLC (see Fig. 8) used for the magnetic ball suspension system has 11 membership functions for each process input ($e(kT)$ and $c(kT)$). There are a total of 121 rules (i.e., 121 output membership functions), with all the output membership function centers initialized at zero. The universes of discourse for each process input are "normalized" to the interval $[-1, 1]$ by means of constant scaling factors. For our fuzzy controller design, the gains g_e, g_c, and g_v were employed to normalize the universe of discourse for the error $e(kT)$, change in error $c(kT)$, and controller output $v(kT)$, respectively. The gains g_e and g_c then act as the scaling factors of the physical range of the inputs. By changing these gains, the meanings of the premises of the linguistic rules will also be changed. An off-line tuning procedure for selecting these gains (such as the one described in [1]–[5]) is essentially picking the appropriate meaning for each of the linguistic variables. For instance, one of the membership functions E^4 on $e(kT)$ is defined as "PositiveBig" (see Fig. 9) and it covers the region [0.6 1.0] on $e(kT)$. With the gain $g_e = \frac{1}{0.275}$, the linguistic term "PositiveBig" quantifies the position errors in the interval [0.165 0.275]. If the gain is increased to $g_e = \frac{1}{0.05}$ (i.e., reducing the domain interval of the universe of discourse from $[-0.275\ 0.275]$ to $[-0.05\ 0.05]$), then the linguistic term "PositiveBig" quantifies position errors in the interval [0.03 0.05]. Note that the range covered by the linguistic term is reduced by increasing the scaling factor (decreasing the domain interval of the universe of discourse), and thus the true meanings of a membership function can be varied by the gains applied.

In addition, the fuzzy controller rule-base can be seen as a control surface. Then, a two-input, single-output fuzzy controller can be viewed as a functional map which maps the inputs to the output of the fuzzy controller. Therefore, the FMRLC algorithm that constructs the fuzzy controller is essentially identifying this control surface for the specified reference model. With the "granularity" chosen by the number of membership functions and the gain, this control surface is most effective on the domain interval of the input universes of discourse (at the outer edges the inputs and output of the fuzzy controller saturate). For example, the gain $g_e = \frac{1}{0.275}$ is chosen to scale the input $e(kT)$ onto a normalized universe of discourse $[-1\ 1]$. The domain interval of the input universe of discourse on $e(kT)$ is then bounded on $[-0.275\ 0.275]$. Hence, a tuning procedure that changes the gains g_e and g_c is altering the "coverage" of the control surface. Note that for a rule-base with a fixed number of rules, when the domain interval of the input universes of discourse are large (i.e., small g_e and g_c), it represents a "coarse control" action; and when the input universes of discourse are small (i.e., large g_e and g_c), it represents a "fine control" action. Hence, we can vary the "granularity" of a control surface by varying the gains g_e and g_c.

Based on the above intuition about the gains and the resulting fuzzy controller, it is possible to develop different strategies to adjust the gains g_e and g_c so that a smaller rule-base can be used at the input range needed the most. This is done by adjusting the meaning of the linguistic values based on the most recent input signals to the fuzzy controller so that the control surface is properly *focused* on the region that describes the system activity. In the next section, we will give details on three techniques that we will be able to scale (i.e., "auto-tune"), to move (i.e., "auto-attentive"), and to move and remember (i.e., "auto-attentive with memory") the rule-base to achieve dynamically focused learning for FMRLC. For comparison purposes, all the fuzzy controllers in the following sections have 121 rules, where each of the input universes of discourse have 11 uniformly spaced membership functions (the same ones that were used in Fig. 9 of Section II-C). The initial gains g_e and g_c are chosen to be $\frac{1}{0.05}$ and $\frac{1}{0.5}$ respectively[8] in order to ensure various DFL approaches for FMRLC are activated so that we can study their behavior. It is interesting to note that with this choice of gains, the FMRLC (without dynamically focused learning) will produce the unstable responses shown in Fig. 16. In the next three sections we will introduce techniques that will focus the rule-base so that such poor behavior (i.e., where the ball is lifted to hit the coil) will be avoided.

B. DFL Strategy I—Auto-Tuning Mechanism

In the standard FMRLC design for the magnetic ball suspension system, the input sequence does not excite the whole range of the designated input universes of discourse (see Fig. 14). Instead, the rule-base learned for the input sequence only covered the center part of the rule-base. Hence, to achieve an adequate number of rules to enhance the granularity of the rule-base near the center, it would be necessary to design the

[8]This choice will make the initial rule-base of the fuzzy controller much smaller than the center learned region as shown in a dashed box in Fig. 14(a).

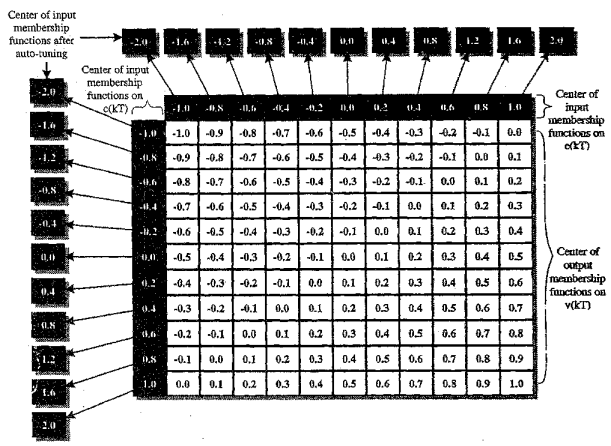


Fig. 17. DFL I: Dynamics of auto-tuning for FMRLC.

rule-base so that it is located at exactly where most of the rules are needed. However, we would like to ensure that we can adapt the fuzzy rule-base should a different input sequence drive the operation of the system out of this center region.

Based on our experience in tuning the FMRLC, it is often observed that the gains g_e and g_c are chosen as bounds on the inputs to the controller so that the rule-base represents the active region of the control actions. Hence, our approach to scale each input universe of discourse is that we chose the maximum of each input over a time interval (window) of the last T_A seconds ($\max_{T_A}\{e(kT)\}$ and $\max_{T_A}\{c(kT)\}$). Then this maximum value is defined as the gain of each input $e(kT)$ and $c(kT)$ so that $g_e = \frac{1}{\max_{T_A}\{e(kT)\}}$ and $g_c = \frac{1}{\max_{T_A}\{c(kT)\}}$. After some experimentation, we chose $T_A = 0.1$ s[9]. Longer time windows tend to slow down the auto-tuning action; while a shorter window often speeds up the auto-tuning but the resulting control is more oscillatory. Once the gains are changed, it is expected that the learning mechanism of the FMRLC will adjust the rules accordingly when they are re-activated, because the scaling will alter all the rules in the rule-base. Note that the learning process now involves two individual, distinct components: (i) the FMRLC learning mechanism that fills in the appropriate consequents for the rules, and (ii) the auto-tuning mechanism (i.e., an adaptation mechanism) that scales the gains which actually re-define the premise membership functions. Normally, we make the learning mechanism operate at a higher rate than the auto-tuning mechanism for the premise membership functions in order to try to assure stability. If the adaptation mechanism is designed to be faster than the learning mechanism, the learning mechanism will not be able to keep up with the changes made by the auto-tuning mechanism so that it will never be able to learn the rule-base correctly. The different rates in learning and adaptation can be achieved by adjusting the sampling period T of the FMRLC and the window length T_A of the auto-tuning mechanism.

Fig. 17 illustrates how the gain scaling implemented by auto-tuning affects the input membership functions. Note that the center of the output membership functions on $v(kT)$ in Fig. 17 are filled with a "standard" set of rules such that they represent a typical choice (for illustration purposes) from a control engineer's experience for the fuzzy controller. For example, at the beginning the centers of each input membership functions are shown in the rule-base shown in Fig. 17. In the next time instant if the values $\max_{T_A}\{e(kT)\}$ and $\max_{T_A}\{c(kT)\}$ are halved, the gains $g_e = \frac{1}{\max_{T_A}\{e(kT)\}}$ and $g_c = \frac{1}{\max_{T_A}\{c(kT)\}}$ are now doubled. Then, the overall effect is that each of membership functions in the input universes of discourse is given a new linguistic meaning and the domain of the control surface is expanded as shown by the centers of each input membership function after the auto-tuning action (see Fig. 17).

Notice that we will require a maximum gain value; otherwise each input universe of discourse for the fuzzy system may be reduced to zero (where the gains g_e and g_c go to infinity) so that controller stability is not maintained. For the magnetic ball suspension system, the maximum gain is chosen to be the same as the initial value (i.e., $g_e = \frac{1}{0.05}$ and $g_c = \frac{1}{0.5}$). Other gains g_v, g_{y_e}, g_{y_c} and g_f are the same as the one used in the standard FMRLC in Section II-C.

For FMRLC with auto-tuning, Fig. 18 shows that the ball position can follow the sinusoidal input sequence very closely, although perfect tracking of the reference response cannot be achieved. However this result is better than the case where conventional adaptive control is used (see Fig. 6), and definitely better than the standard FMRLC design (see Fig. 12). Notice that the results shown in Fig. 18 are similar to Fig. 15 where 10201 rules are used; however, the auto-tuning approach used only 121 rules (and there are extra computations needed to implement the auto-tuning strategy—issues in computational complexity for the DFL strategies are discussed in more detail in Sections III-E and III-F below). Fig. 19 shows excellent responses for the same auto-tuned FMRLC with the step input sequence where the ball position follows the reference model without noticeable difference (compare to Figs. 7 and 11 for the MRAC and FMRLC respectively).

C. DFL Strategy II—Auto-Attentive Mechanism

The auto-tuning mechanism seems to work well, but the performance can still be improved. One of the major disadvantages of auto-tuning the FMRLC is that all the rules in the rule-base are changed by the scaling of the gains, which may cause distortions in the rule-base and requires the learning mechanism to re-learn the appropriate control laws. Hence, instead of scaling, we will consider moving the entire rule-base will respect to a fixed coordinate system so that the fuzzy controller can "pay attention" to the current inputs.

To explain the auto-attentive mechanism it is convenient to define some new terms that are depicted in Fig. 20. First of all, the rule-base of the fuzzy controller is considered to be a single cell called the "auto-attentive active region," and it represents a fixed size rule-base which is chosen by the initial scaling gains (i.e., g_e and g_c must be selected *a priori*). The outermost shaded region of the rule-base is defined as the "attention

[9]It was found via simulations that any $T_A \in [0.05, 0.3]$ s can be used equally effectively.

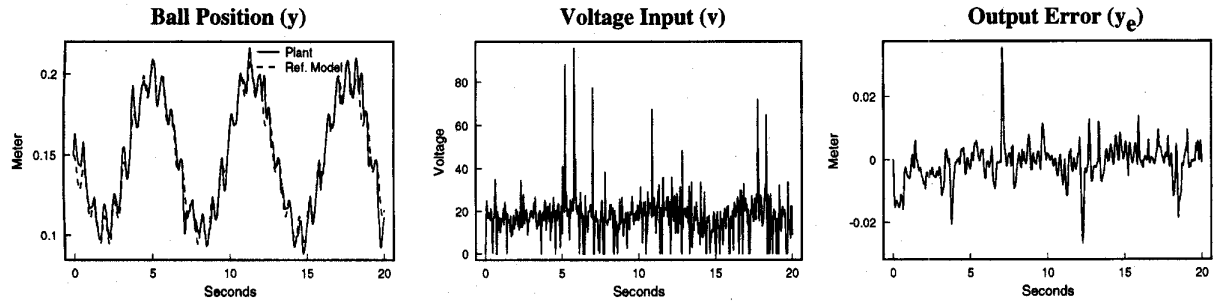


Fig. 18. Responses for FMRLC with auto-tuning (sinusoidal input sequence).

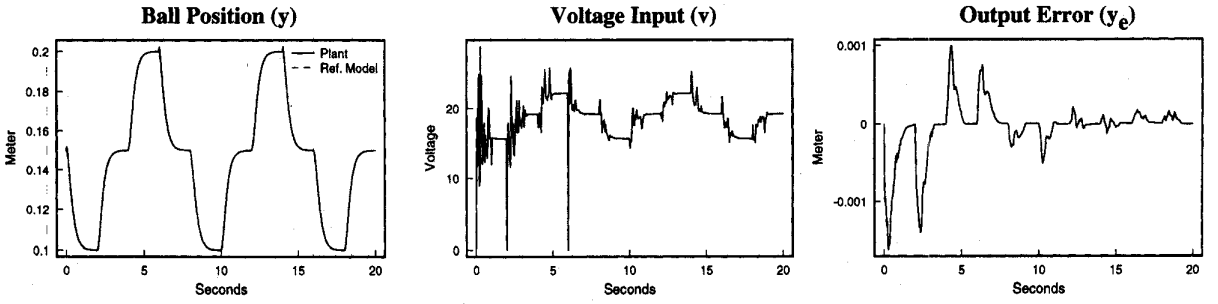


Fig. 19. Responses for FMRLC with auto-tuning (step input sequence).

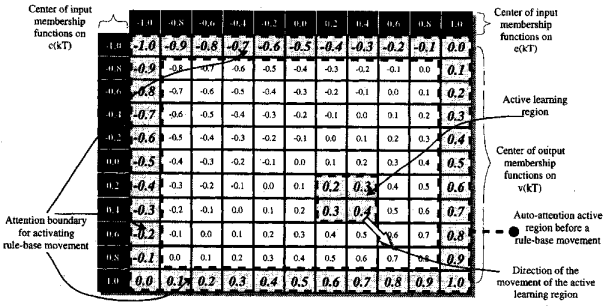


Fig. 20. DFL II: Auto-attentive mechanism for FMRLC (before shifting).

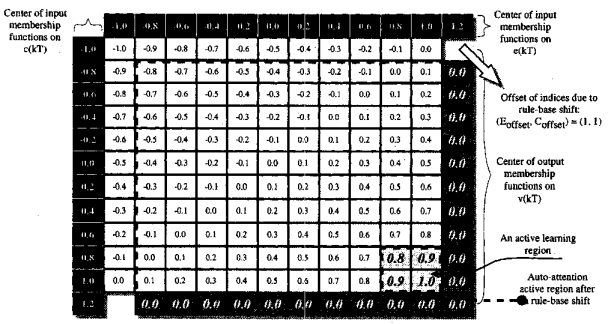


Fig. 21. DFL II: Auto-attentive mechanism for FMRLC (after shifting).

boundary." The four shaded rules[10] in the lower right portion of the rule-base are referred as the FMRLC "active learning region" where the rules are updated by the learning mechanism of the FMRLC. Finally, the white arrow in Fig. 20 indicates the direction of movement of the active learning region.

For the auto-attentive mechanism, if the active learning region moves to be adjacent to the attention boundary, a "rule-base shift" is activated. For example, if the active learning region hits the lower right attention boundary as shown in Fig. 21, the result is that the rule-base will be shifted down one unit and to the right one unit (i.e., the width of a membership function). The shift in the rule-base is represented by the "offset" of the rule-base from its initial position[11], which is $(E_{\text{offset}}, C_{\text{offset}}) = (1, 1)$ as shown in Fig. 21 for this example. With the offset values, the shift of the rule-base is simply obtained by adding the offset values to each of the premise membership functions. After the rule-base is shifted, the active attention region is moved to the region in the large dash box in Fig. 21. In the new un-explored region, the consequent of the rules will be filled with zeros since this represents that there is no knowledge of how to control in the new region. Conceptually, the rule-base is moving and following the active learning region. We emphasize, however, that if the active learning region never hits the attention boundary, there will never be a rule-base shift and the controller will behave exactly the same as the standard FMRLC. Overall, we see that the auto-attentive mechanism seeks to keep the controller rule-base focused on the region where the FMRLC is learning how to control the system (one could think of this as we did with the auto-tuning mechanism as adapting the meaning of the linguistics). If the rule-base shifts frequently the system will "forget" how to control in the regions where it used to be, yet

[10] Note that there are at most four rules "on" at one time due to our choice for membership functions shown in Fig. 9.

[11] We chose the convention that shifting the rule-base to the right and downward to be a positive offset and shifting the rule-base to the left and upward to be a negative one. This choice is made to be compatible with the convention used in the input universes of discourse in the rule-base (as shown in Figs. 20 and 21).

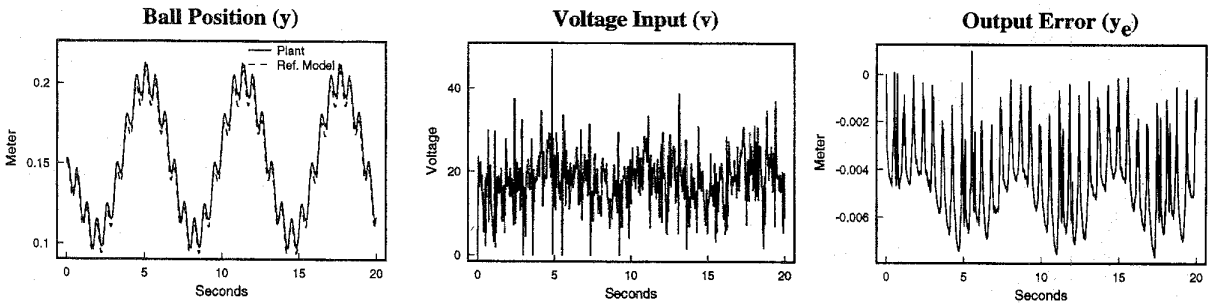


Fig. 22. Responses for FMRLC with auto-attentive mechanism (sinusoidal input sequence).

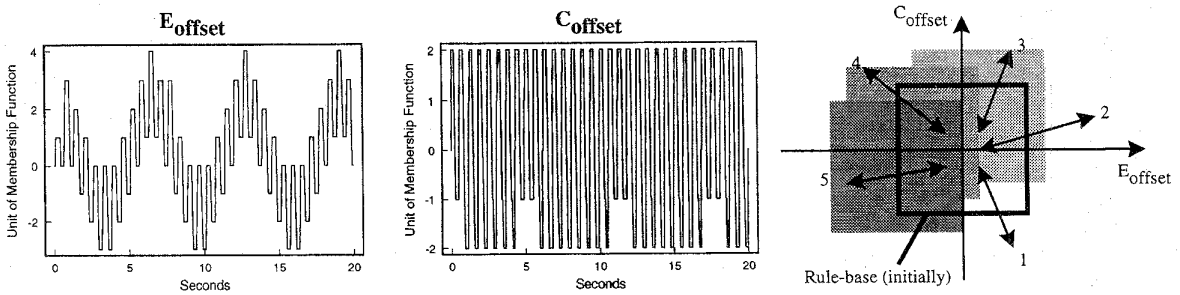


Fig. 23. (a)–(c) Movement of the rule-base for the auto-attentive mechanism (sinusoidal input sequence).

learn how to control in the new regions where adaptation is needed most.

For the magnetic ball suspension system, the input universes of discourse are chosen as $[-0.05, 0.05]$ and $[-0.5, 0.5]$ (i.e., the gain g_e and g_c are $\frac{1}{0.05}$ and $\frac{1}{0.5}$, respectively), while all the other gains are the same as the ones used in the standard FMRLC design in Section II-C. Note that we can consider the width of the attention boundary to be a design parameter, but we found that it is the best to set the attention boundary as shown in Fig. 20 since this choice minimizes oscillations and unnecessary shifting of the rule-base for this example.

Similar to the auto-tuning DFL strategy, there are two distinct processes: (i) the FMRLC learning mechanism that fills in appropriate consequents for the rules and (ii) the auto-attentive mechanism (i.e., an adaptation mechanism) that moves the entire rule-base. Moreover, the learning mechanism is running at a higher rate compared to the auto-attentive mechanism (in order to try to assure stability), since we only allow a shift of the entire rule-base by a single unit in any direction in any time instant. The rate of adaptation can be controlled by using a different attention boundary to activate the rule-base movement. For example, if the attention boundary shown in Fig. 20 is in the inner part of the rule-base (say the second outer-most region of the rule-base instead of the outer-most region), then the rule-base will be shifted more often and thus increase the adaptation rate of the auto-attentive mechanism.

Fig. 22 illustrates the performance of the FMRLC with the auto-attentive mechanism. We see that the ball position can follow the input sequence very closely, although perfect tracking of the reference response cannot be achieved (with maximum output error y_e within ± 0.0078 m), but this result is better than the case where the conventional adaptive controller (see Fig. 6), the standard FMRLC with 10201 rules (see Fig. 15) and the auto-tuning FMRLC (see Fig. 18), and definitely better then the unstable standard FMRLC (see Fig. 12 where the ball is lifted to the coil).

To gain insight into the dynamics of the auto-attentive mechanism, Fig. 23(a) and (b) show the E_{offset} and C_{offset} values throughout the simulation, and Fig. 23(c) depicts the first five movements of the rule-base. The double arrows in Fig. 23(c) denote the movement of the rule-base from the initial position (shown as a empty box) to an outer region (shown as a shaded box), while the number next to the shaded box is the rule-base at the next time instant where the rule-base moved (the shades also change to deeper gray as time progress). Hence, the rule-base is actually moving closer and further to the $(E_{\text{offset}}, C_{\text{offset}})$ origin as time progresses, and it also moves around the initial position in a counter-clockwise circular motion (this motion is induced by the sinusoids that the rule-base is trying to track). Note that we have done simulation tests for different sizes of the active attention region for improving the responses from the auto-attentive FMRLC. However, we found that smaller active attention regions result in excessive motion for the rule-base, while larger auto-attention active regions will have the low rule-base "granularity" problem as the standard FMRLC. Fig. 24 shows excellent responses for the same auto-attentive FMRLC design with a step input sequence, which is basically the same as in the case of standard FMRLC (see Fig. 11).

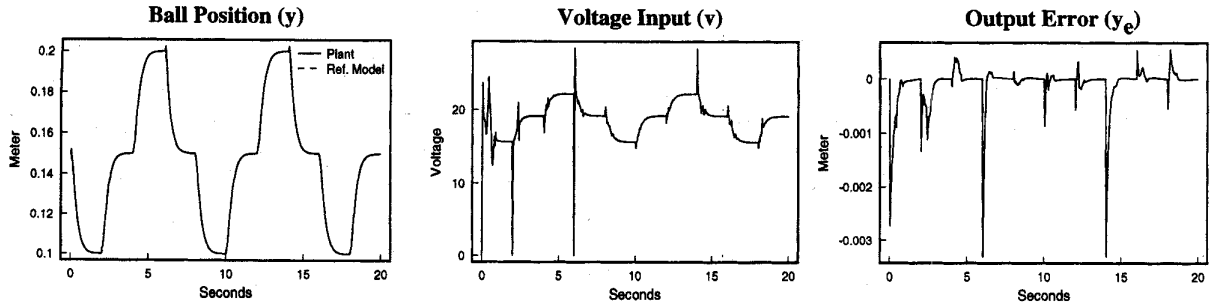


Fig. 24. Responses for FMRLC with auto-attentive mechanism (step input sequence).

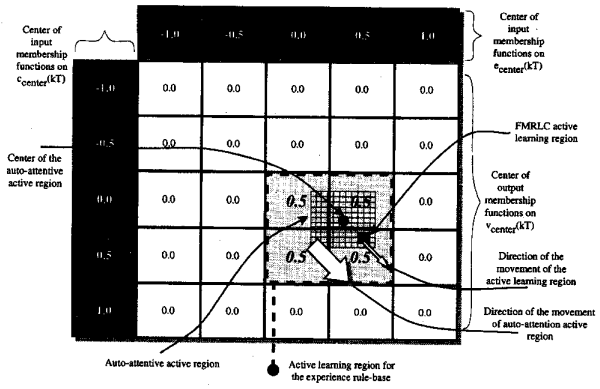


Fig. 25. DFL III: The fuzzy experience model for the auto-attentive mechanism with memory for FMRLC.

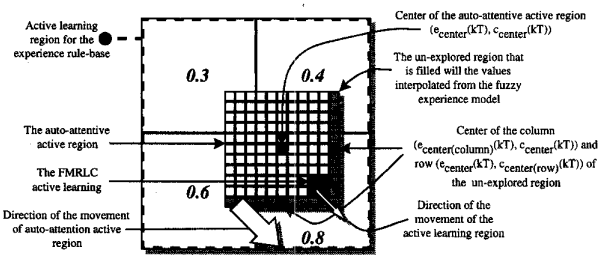


Fig. 26. DFL III: The enlargement of the active learning region for the experience rule-base.

D. DFL Strategy III—Auto-Attentive Mechanism with Memory

Note that in the auto-attentive DFL strategy, every shift of the rule-base will create a new un-explored region. This region will be filled with zeros since this represents that we have no knowledge of how to control when we move into a new operating condition. Having to learn the new regions from scratch after every movement of the rule-base can cause degradations in the performance of the auto-attentive FMRLC since it will require the learning mechanism to fill in the unknown rules (i.e., additional time for learning will be needed). For example, if an auto-attentive FMRLC has been operating for a long time on an input sequence, then at some time instant a disturbance affected the controller inputs and forced the rule-base to leave its current position, some of the rules are lost and replaced by new rules that will accommodate the disturbance. When the temporary disturbance is stopped and the rule-base returns to its initial position again, its previous experience is lost and it is required to "re-learn" everything about how to control in a region where it really has gained a significant amount of experience.

Fuzzy Experience Model: To better reflect the "experience" that a controller gathers, we will introduce a third fuzzy system which we call the "fuzzy experience model" for the FMRLC (the first one is the fuzzy controller and the second one is the fuzzy inverse model) as the memory to record an abstraction of the control laws which are in the region previously reached through the auto-attentive mechanism. The rule-base of this fuzzy experience model (i.e., the "experience rule-base") is used to represent the "global knowledge" of the fuzzy controller. In this case, no matter how far off the auto-attentive mechanism has offset the rule-base, there is a rough knowledge of how to control in any region where the controller has visited before. In other words, this fuzzy controller not only possesses learning capabilities from the learning mechanism and adaptation abilities from the auto-attentive algorithm, it also maintains a representation of the "experience" it has gathered on how to control in an additional fuzzy system (an added level of memory and hence learning[12]).

As shown in Fig. 25, the fuzzy experience model has two inputs $e_{\text{center}}(kT)$ and $c_{\text{center}}(kT)$, which represent the center of the auto-attentive active region that is defined on $e(kT)$ and $c(kT)$. For our example, these inputs have five symmetric, uniformly spaced membership functions, and there are a total of 25 rules (i.e., 25 output membership functions which are initialized at zero at the beginning). The universes of discourse for each of these inputs are normalized to the interval $[-1, 1]$ by means of constant scaling factors. To represent the global knowledge, the gains $g_{e_{\text{center}}} = \frac{1}{0.275}$ and $g_{c_{\text{center}}} = \frac{1}{2.0}$ were employed to normalize the universe of discourse for the error $e_{\text{center}}(kT)$ and change in error $c_{\text{center}}(kT)$. The same gains used in the standard FMRLC design are employed here since these are assumed to represent the complete universes of discourse (determined by the physical limits) for the magnetic ball suspension system. The output universe of discourse is selected to be $[-1, 1]$ with gain $g_{v_{\text{center}}} = 1$, which preserves the original information from the fuzzy experience model without scaling.

[12]One can easily envision how to add successive nested learning/auto-attentive mechanisms and memory models for the FMRLC.

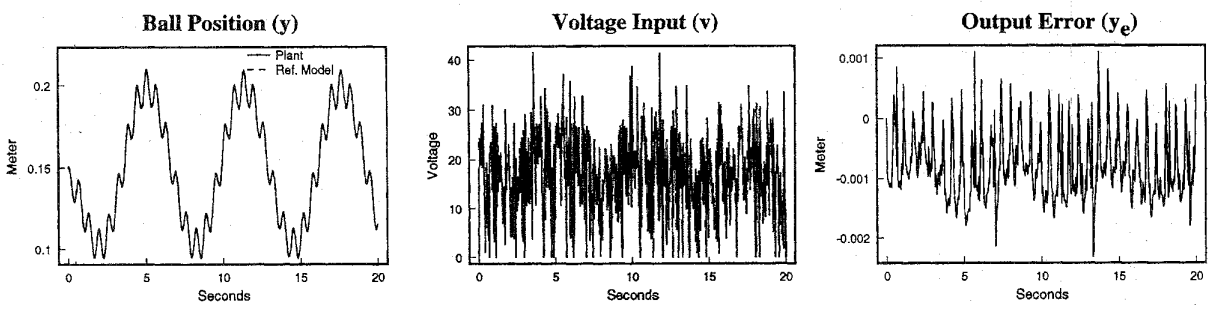


Fig. 27. Responses for FMRLC with auto-attentive mechanism with memory (sinusoidal input sequence).

Learning Mechanism for Fuzzy Experience Model: The learning mechanism for this fuzzy experience model is similar to the learning mechanism for the FMRLC except that the fuzzy inverse model is not needed. The two inputs $e_{\text{center}}(kT)$ and $c_{\text{center}}(kT)$ (i.e., the center of the auto-attentive active region) are used to calculate the "experience" $v_{\text{center}}(kT)$ for the current auto-attentive active region, and the "activation level" of all the rules, while only the rules with activation levels larger than zero will be updated (i.e., the same as the method used in the FMRLC learning mechanism). Each time after the fuzzy controller rule-base (i.e., the auto-attentive active region) is updated, the numerical average value of the auto-attentive rule-base consequent centers $v_{\text{center(avg)}}(kT)$ will be taken for the corresponding fuzzy experience model. Hence, the change of the consequent fuzzy sets of the experience rule-base, that have premises with nonzero activation levels, can be computed as $v_{\text{center(chg)}} = v_{\text{center(avg)}}(kT) - v_{\text{center}}(kT)$ and $v_{\text{center(chg)}}$ is used to update the fuzzy experience model exactly the same way as the fuzzy controller is updated by the y_f in Section II-C. For example, the shaded area in Fig. 25 (the active learning region for the experience rule-base) is activated by the inputs $e_{\text{center}}(kT)$ and $c_{\text{center}}(kT)$ (i.e., these are the rules that have nonzero activation level). First, assume that the centers of all membership functions on $v_{\text{center}}(kT)$ are zero at the beginning, and thus the output of the fuzzy experience model $v_{\text{center}}(kT)$ is zero. Then, assume we found $v_{\text{center(avg)}}(kT) = 0.5$ to be the average value of the control surface (i.e., average value of the centers of the output member functions) for the auto-attentive active region; hence we the update of the fuzzy experience model $v_{\text{center(chg)}} = 0.5$ can be found. Hence, the consequent membership functions of the fuzzy experience model will be shifted to 0.5 as shown in the shaded region of Fig. 25. It is obvious that there are numerous other methods to obtain an abstract representation of the rule-base in the auto-attentive active region besides using the average[13]. Our approach uses a simple method to represent experience and hence provides a rough estimate of the unknown control laws (so that we can do better than simply filling in the unknown part with zeros as we did for DFL II). Then, it is hoped that the learning mechanism of the FMRLC will properly update the rules that were filled in by the experience model if they are activated.

With this approach the new un-explored region of the fuzzy controller (i.e., the shaded region at the boundary of the auto-attentive active region in Fig. 21) can then be interpolated using the information recorded in the fuzzy experience model, instead of filling with zeros in the consequent of the rules. The interpolation is achieved by finding the consequent fuzzy sets for the unexplored region (see the shaded region in Fig. 26) given the centers of each of the premise fuzzy sets. The enlarged active learning region for the experience rule-base shown in Fig. 26 illustrates that there are 21 un-explored rules in the auto-attentive active region needed to be estimated. With the interpolation approach described above, we will be computing the output of 21 different locations in the experience rule-base. These computations are expensive for obtaining the guesses for the un-explored region, and thus we choose to only compute the consequent fuzzy sets for the center of the column (i.e., with input at $(e_{\text{center(column)}}(kT), c_{\text{center}}(kT))$ as shown in Fig. 26) and the row (i.e., with input at $(e_{\text{center}}(kT), c_{\text{center(row)}}(kT))$ as shown in Fig. 26) of the un-explored region, and then fill the entire column or row with their center values. Note that the auto-attentive mechanism that uses the fuzzy experience model for memory essentially performs a multi-dimensional interpolation, where a coarse rule-base is used to store the general shape of the global control surface and this information is used to fill in guesses for the auto-attentive active region as it shifts into regions that it has visited before.

As shown in Fig. 27 when the auto-attentive mechanism with memory is used, the ball position can follow the input sequence almost perfectly with maximum output error y_e within ± 0.0022 m (i.e., about 3.5 times smaller than the auto-attentive FMRLC without memory in Fig. 22). Fig. 28 shows the results for the same technique when we use a step input sequence. Notice that in terms of output error these are the best results that we obtained (compared to the results from MRAC and the two other dynamic focusing techniques).

E. Computational Issues

Note that when different DFL strategies are applied to the standard FMRLC to minimize memory usage in the rule-base and to allow the rule-base to "focus" there are additional computations for these operations. In DFL I we need to save

[13] We have tried other more complicated methods such as using least squares to find a linear surface that best fits the control surface, but we found that such a method significantly increases the computational complexity without major performance improvements.

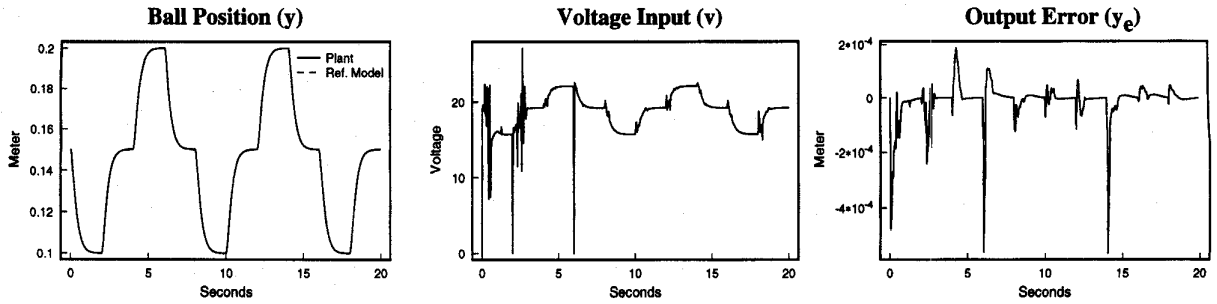


Fig. 28. Responses for FMRLC with auto-attentive mechanism with memory (step input sequence).

a window of input data, where the maximum value over the entire window length is computed for each input, so that the input gains of the fuzzy controller can be calculated in the next time step. The memory used for the windowed data and the elapsed time for computing the maximum in DFL I is relatively large when these functions were implemented in software (we more carefully quantify what computing resources are needed for all the DFL strategies in the next section).

For DFL II where the rule-base is allowed to shift, we used a specially structured indexing system to incorporate the movement of the rule-base so that the extra computations are minimal. To illustrate this indexing system, a two-input, one-output fuzzy system shown in Fig. 29(a) is discussed here. The fuzzy rule-base contains a set of **If** ... **Then** rules which can be represented in the table as shown in Fig. 29(a) (note that all numbers shown in Fig. 29(a) are indices, *not* the centers of the membership functions as in some previous figures). Assuming that there are two membership functions in each of the two input universes of discourse (named by "0" and "1"), and there four membership functions in the output universe of discourse (named by "0," "1," "2," and "3"), then all the rules can be listed out as shown in Fig. 29(b) where the premise and consequent of a rule are listed together in a row. Note that with this indexing scheme for the membership functions, the indices of the input membership functions of the **If** ... **Then** rules appear to be a number system (in this example, it is a binary number system).

Normally, the most computationally intensive part of simulating the fuzzy system is to identify which rules are activated since this has in the past often been done by checking whether each and every rule is activated. If we employ the numbering scheme in Fig. 29(b) in order to locate the activated rules, all that is needed is to identify which premises become active. For example, in the case with triangular membership functions where at most two membership functions can be activated at once, instead of checking all $\Pi_{i=1}^{n} N_i$ rules (where n is the total number of inputs and N_i is the number of membership functions on the ith input universe of discourse), we can pinpoint the 2^n possible rules that can be activated very fast since the premise membership functions that are "on" directly identify which rules are activated (hence an exhaustive search is not necessary). While for the case with two membership functions in each input universe of discourse the binary indexing scheme is not particularly innovative, it

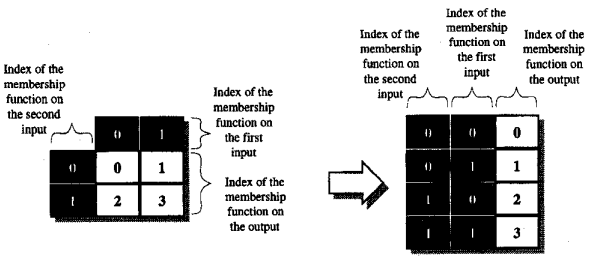


Fig. 29. (a)–(b) Structured fuzzy rule-base indexing system.

is important to note that the exact same scheme works for n membership functions in each input universe of discourse by simply using a base-n indexing scheme. It is the use of such a scheme for a higher number of inputs that will provide significant computational efficiency.

Next, assume that the rule-base is shifted via DFL II as shown in Fig. 30(a) where the movement can be indicated by adding one unit (i.e., the offset value to the rule-base with respect to its initial position) to the indices of each input universes of discourse. Hence, before the rule-base shift each of the input universes of discourse has the membership functions named "0" and "1," and they are shifted to "1" and "2" as shown in the gray filled input indices in Fig. 30(a). This adjustment is further illustrated in Fig. 30(b) and Fig. 30(c), where the indices with the lowest value (i.e., "0" in our example) are simply replaced by the highest value plus the offset of the rule-base (in our case the highest value is "1" and the offset is "1" so we replace the index "0" by "2"). It is important to note that the rule-base before and after the shift still form a binary number system except that the digits used are no longer "0" and "1" but "1" and "2." This indexing scheme works in a similar way for n inputs to the fuzzy system by employing a base-n indexing scheme as it is discussed above. Also notice that this adjustment scheme ensures that we do not need additional memory for the fuzzy system to operate in the region where the rule-base has been shifted. There is no real movement is the memory except that the rule-base indices of some premises are renamed. Hence, this scheme significantly reduces the computational complexity of DFL II so that it is at a level similar to that of the standard FMRLC.

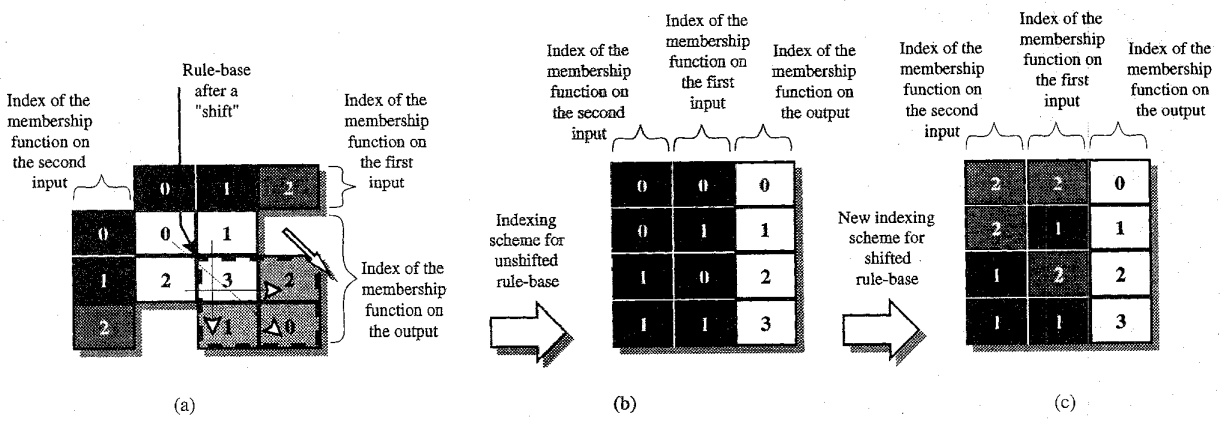


Fig. 30. (a)–(c) Structured fuzzy rule-base indexing system with rule-base movement.

TABLE I
COMPARISON OF DIFFERENT CONTROL STRATEGIES[14,15]

Controller Type	(Step Input) $\Sigma_k y_e^2(kT)$	(Step Input) $\Sigma_k v^2(kT)$	(Sinusoidal Input) $\Sigma_k y_e^2(kT)$	(Sinusoidal Input) $\Sigma_k v^2(kT)$	Elapsed time[14] (in ms)	Mem. usage[15] (in kb)	See Figures
MRAC	1.05×10^{-1}	1.81×10^5	1.34×10^{-1}	1.86×10^5	0.12	0.16	7, 6
Standard FMRLC	4.45×10^{-5}	1.77×10^5	unstable		0.53	1.25	11, 12
FMRLC with 10201 rules	not tested		7.53×10^{-2}	1.86×10^5	0.96	43.5	15
FMRLC without DFL	not tested		unstable		0.53	1.25	16
Auto-tuned FMRLC	1.26×10^{-4}	1.81×10^5	5.23×10^{-2}	1.84×10^5	1.18	5.43	19, 18
Auto-attentive FMRLC	8.70×10^{-5}	1.79×10^5	1.13×10^{-2}	1.81×10^5	0.55	1.25	24, 22
Auto-attentive FMRLC with memory	3.06×10^{-6}	1.79×10^5	4.80×10^{-4}	1.79×10^5	1.01	1.86	28, 27

Finally, the DFL III is constructed by augmenting DFL II with the fuzzy experience model. The additional fuzzy system in DFL III not only requires much more memory and thus more computations (since it adds another fuzzy system), it also requires us to calculate the average value of the center values of the output membership functions of fuzzy controller rule-base (which is expensive for a large rule-base) in order to update the fuzzy experience model. In summary, DFL I and III require significant additional memory and calculations, but DFL II is nearly as efficient as the standard FMRLC.

F. Summary Evaluation

As shown in the results for the magnetic ball suspension system, FMRLC with DFL outperforms the original FMRLC control strategies and the conventional adaptive control techniques. The memory usage in the fuzzy controller rule-base is minimized at the expense of slightly increasing the computational complexity due to the addition of the DFL strategies. To clarify these points, next we will summarize the results of the entire paper. Recall that we ran all our simulations in this paper for 20 seconds and that our sampling interval was $T = 0.004$ s. Let $\sum_k y_e^2(kT)$ and $\sum_k v^2(kT)$ denote the sum of the squares of the signals $y_e(kT)$ and $v(kT)$ over the entire simulation interval. Table I summarizes the results from all simulations in this paper.

First, note that while the computational requirements for the MRAC are the lowest, it achieves the worse performance. As Table I shows, there are stability problems with the standard FMRLC for certain inputs. Notice that the amount of control energy $\sum_k v^2(kT)$ used is roughly the same for all the controllers. In addition, although the auto-tuned FMRLC achieved moderate performance with a small rule-base, it is relatively slow (in terms of elapsed time) due to the use of a window-based approach to tune the controller input gains g_e and g_c. While the auto-attentive FMRLC with memory achieved the best performance, its elapsed time is the second worst one. If we were interested in implementation of the

[14]The elapsed time is calculated from a simulation program run on a NeXTStation with a Motorola 68040 33Mhz CPU, DSP56001, and 32Mb RAM. Due to the limited resolution of 1/60 sec used to calculate time intervals, we simulated the controller 50,000 times and calculated the average elapsed time to estimate the length of time used by the controller to compute a single control value.

[15]The memory usage includes the amount of storage needed for the rule-bases and all supporting variables for the controller.

controllers[16], it is observed from Table I that the best strategy is perhaps the auto-attentive FMRLC because it is only slightly slower that the standard FMRLC and yet it achieved low output error.

In case of a large number of inputs and rules, DFL will be a particularly useful technique for the FMRLC. However, it must be emphasized that the DFL strategies discussed in this paper introduce more parameters for tuning. While this can complicate the tuning process, it also increases the flexibility, as well as the learning capabilities of the FMRLC design. Finally, note that all the DFL techniques can also be applied to the fuzzy inverse model if needed (for example, we have found it quite useful to auto-tune the gains of the fuzzy inverse model).

IV. Conclusion

In this paper we used a magnetic ball suspension system as a testbed to: (i) introduce and evaluate three approaches to dynamically focused learning control (auto-tuning, auto-attentive, and auto-attentive with memory); and (ii) compare the performance of the FMRLC with the DFL enhancement to conventional model reference adaptive control and the original FMRLC. We found that while the FMRLC with DFL is more computationally intensive than conventional MRAC and the FMRLC, it can provide enhanced performance over both of these techniques.

We must emphasize that while we have only shown how to use the concept of dynamically focused learning for the FMRLC it is a general concept that could be applied to other control strategies (e.g., in neural control). Moreover, while the concept of dynamically focused learning may extend to other control paradigms (indeed, the auto-tuning approach is used in [51], [52]), the performance realized in this case study may not. It is therefore important to study the following issues in future work:

i) stability, convergence, and robustness issues for the FMRLC with DFL,
ii) more extensive comparisons with conventional adaptive control techniques (e.g., adaptive variable structure control), and
iii) application to a plant with more complex and challenging dynamics.

Furthermore, there is a significant need to perform experimental evaluation of the DFL strategies. For example, it would be interesting to determine if the DFL strategies can enhance the performance of the FMRLC that was implemented for the two-link flexible robot in [29, 30].

References

[1] J. R. Layne and K. M. Passino, "Fuzzy model reference learning control," in *Proc. IEEE Conf. Control Applications*, Dayton, OH, Sept. 1992, pp. 686–691.

[2] ——, "Fuzzy model reference learning control for cargo ship steering," *IEEE Contr. Syst. Mag.*, vol. 13, no. 6, pp. 23–34, 1993.

[16] An implementation of several control strategies for the magnetic ball suspension system is described in [48]. It is interesting to note that the resulting characteristics of their experimental closed-loop responses (e.g., in terms of control inputs) are quite similar to our simulated responses.

[3] J. R. Layne, K. M. Passino and S. Yurkovich, "Fuzzy learning control for antiskid braking systems," in *Proc. IEEE Conf. Decision Control*, Tuscon, AZ, Dec. 1992, pp. 2523–2528.

[4] ——, "Fuzzy learning control for antiskid braking systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 1, no. 2, pp. 122–129, June 1993.

[5] J. R. Layne and K. M. Passino, "Fuzzy model reference learning control," to appear in *Journal of Intelligent and Fuzzy Systems*, 1996.

[6] R. M. Tong, "A control engineering review of fuzzy systems," *Automatica*, vol. 13, pp. 559–569, Nov. 1977.

[7] M. Sugeno, "An introductory survey of fuzzy control," *Inform. Sci.*, vol. 36, no. 1, pp. 59–83, 1985.

[8] K. Self, "Designing with fuzzy logic," *IEEE Spectrum*, pp. 42–105, Nov. 1990.

[9] S. Sastry and M. Bodson, *Adaptive Control: Stability, Convergence and Robustness*. Englewood Cliffs, NJ: Prentice Hall, 1989.

[10] W. A. Kwong and K. M. Passino, "Dynamically focused fuzzy learning control," in *Proc. American Control Conf.*, Seattle, WA, June 21–23, 1995, pp. 3755–3759.

[11] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. Syst., Man, Cybern.*, vol. 3, no. 1, pp. 28–44, 1973.

[12] ——, "Fuzzy logic," *IEEE Computer*, pp. 83–93, Apr. 1988.

[13] E. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *Int. J. Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975.

[14] E. Mamdani, "Advances in the linguistic synthesis of fuzzy controllers," *Int. J. Man-Machine Studies*, vol. 8, no. 6, pp. 669–678, 1976.

[15] W. Kickert and E. Mamdani, "Analysis of a fuzzy logic controller," *Fuzzy Sets and Systems*, vol. 1, pp. 29–44, 1978.

[16] T. Procyk and E. Mamdani, "A linguistic self-organizing process controller," *Automatica*, vol. 15, no. 1, pp. 15–30, 1979.

[17] S. Shao, "Fuzzy self-organizing controller and its application for dynamic processes," *Fuzzy Sets and Systems*, vol. 26, pp. 151–164, 1988.

[18] E. Scharf and N. Mandic, "The application of a fuzzy controller to the control of a multi-degree-of-freedom robot arm," in *Industrial Applications of Fuzzy Control*, M. Sugeno, Ed. Amsterdam: North-Holland, 1985, pp. 41–62.

[19] R. Tanscheit and E. Scharf, "Experiments with the use of a rule-based self-organizing controller for robotics applications," *Fuzzy Sets and Systems*, vol. 26, pp. 195–214, 1988.

[20] T. Yamazaki, "An improved algorithm for a self-organizing controller and its experimental analysis," *Ph.D. Thesis*, London University, 1982.

[21] S. Isaka, A. Sebald, A. Karimi, N. Smith and M. Quinn, "On the design and performance evaluation of adaptive fuzzy controllers," in *Proc. 1988 IEEE Conf. Decision and Control*, Austin, TX, Dec. 1988, pp. 1068–1069.

[22] S. Daley and K. F. Gill, "A design study of a self-organizing fuzzy logic controller," *Proc. I. Mech. E.*, 1986, vol. 200, pp. 59–69.

[23] ——, "Altitude control of a spacecraft using an extended self-organizing fuzzy logic controller," in *Proc. I. Mech. E.*, vol. 201, no. 2, pp. 97–106, 1987.

[24] ——, "Comparison of a fuzzy logic controller with a $P + D$ control law," *J. Dynamical Syst. Meas. Contr.*, vol. 111, pp. 128–137, June 1989.

[25] D. A. Linkens and J. S. Shieh, "Self-organizing fuzzy modeling for nonlinear system control," in *Proc. IEEE Int. Conf. Fuzzy Systems*, San Diego, CA, Aug. 1992, vol. 1, pp. 210–215.

[26] C. Y. Shieh and S. S. Nair, "A new self tuning fuzzy controller design and experiements," *Second IEEE Int. Conf. Fuzzy Systems*, San Francisco, CA, Mar./Apr. 1993, vol. 1, pp. 309–314.

[27] N. Vijeh, "Self organizing fuzzy logic control of a level control rig," *Second IEEE Int. Conf. Fuzzy Systems*, San Francisco, CA, Mar./Apr. 1993, vol. 1, pp. 303–308.

[28] K. Åström and B. Wittenmark, Eds., *Adaptive Control*. Reading, MA: Addison-Wesley, 1989.

[29] V. G. Moudgal, W. A. Kwong, K. M. Passino and S. Yurkovich, "Fuzzy learning control for a flexible-link robot," in *Proc. American Control Conf.*, Baltimore, MD, June 1994, pp. 563–567.

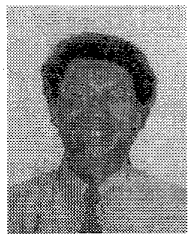
[30] ——, "Fuzzy learning control for a flexible-link robot," to appear in the *IEEE Trans. on Fuzzy Systems*, vol. 3, no. 2, pp. 199–210, May 1995.

[31] W. A. Kwong, K. M. Passino, E. G. Laukonen and S. Yurkovich, "Expert supervision of fuzzy learning systems for fault tolerant aircraft control," *Proc. IEEE*, vol. 83, no. 3, pp. 466–483, Mar. 1995.

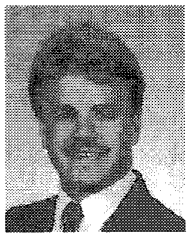
[32] G. Bartolini, G. Casalino, F. Davoli, R. M. M. Mastretta and E. Morten, "Development of performance adaptive fuzzy controllers with applications to continuous casting plants," *Indus. Applicat. Fuzzy Control*, Amsterdam, the Netherlands, pp. 73–86, 1985.

[33] H. Takahashi, "Automatic speed control device using self-tuning fuzzy logic," *1988 IEEE Workshop on Automotive Applications of Electronics*, Dearborn, MI, Oct., 1988, pp. 65–71.

[34] F. V. D. Rhee, H. V. N. Lemke and J. Dijkman, "Knowledge based fuzzy control of systems," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 148–155, Feb. 1990.
[35] P. Graham and R. Newell, "Fuzzy adaptive control of a first-order process," *Fuzzy Sets and Systems*, vol. 31, pp. 47–65, 1989.
[36] ——, "Fuzzy identification and control of a liquid level rig," *Fuzzy Sets and Systems*, vol. 26, pp. 255–273, 1988.
[37] E. Czogała and W. Pedrycz, "On identification in fuzzy systems and its applications in control problems," *Fuzzy Sets and Systems*, vol. 6, pp. 73–83, 1981.
[38] ——, "Control problems in fuzzy systems," *Fuzzy Sets and Systems*, vol. 7, pp. 257–273, 1982.
[39] C. Batur, A. Srinivasan and C. C. Chan, "Fuzzy model based fuzzy predictive control," in *Proc. First Int. Conf. Fuzzy Theory and Technology*, 1992, pp. 176–180.
[40] R. J. Williams, "A class of gradient-estimating algorithms for reinforcement learning in neural networks," in *Proc. Int. Joint Conf. Neural Networks*, San Diego, CA, 1987, pp. 601–608.
[41] C. T. Lin and C. S. G. Lee, "Neural network-based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, no. 12, pp. 1320–1336, Dec. 1991.
[42] R. Ichikawa, K. Nishimura, M. Kunugi and K. Shimada, "Auto-tuning method of fuzzy membership functions using neutral network learning algorithm," *Proc. Second Int. Conf. on Fuzzy Logic and Neural Networks*, 1992, pp. 345–348.
[43] C. Karr, L. Freeman and D. Meredith, "Improved fuzzy process control of spacecraft autonomous rendervous using a genetic algorithm," in *Proc. SPIE Conf. on Intelligent Control and Adaptive Systems*, Orlando, FL, 1989, pp. 274–283.
[44] C. Karr, "Design of an adaptive fuzzy logic controller using a genetic algorithm," in *Proc. Int. Conf. of Genetic Algorithm*, 1992, pp. 450–457.
[45] T. Nishiyama, T. Takagi, R. Yager and S. Nakanishi, "Automatic generation of fuzzy inference rules by genetic algorithm," *8th Fuzzy System Symp.*, pp. 237–240, 1992. (in Japanese).
[46] H. Nomura, L. Hayashi and N. Wakami, "A self-tuning method of fuzzy reasoning by genetic algorithm," *4th IFSA Congress*, Louisville, KY, 1992, pp. 236–245.
[47] L.-X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, NJ: Prentice Hall, 1994.
[48] W. G. Barie, "Design and implementation of a nonlinear state-space controller for a magnetic levitation system," *Master's Thesis*, Department of Electrical Engineering, University of Pittsburgh, 1994.
[49] K. M. Passino and S. Yurkovich, "Fuzzy control," in *The Control Handbook*, W. Levine, Ed. Boca Raton, FL: CRC Press, 1996.
[50] H. Farrell and W. Baker, "Learning control systems," in *An Introduction to Intelligent and Autonomous Control Systems*, P. J. Antsaklis and K. M. Passino, Eds. Norwell, MA: Kluwer, ch. 10, pp. 237–262, 1993.
[51] A. Angsana and K. M. Passino, "Distributed fuzzy control of flexible manufacturing systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 2, no. 4, pp. 423–435, Dec. 1994.
[52] M. Widjaja and S. Yurkovich, "Intelligent control for swing up and balancing of an inverted pendulum system," in *Proc. IEEE Conf. Control Applications*, Albany, NY, Sept. 1995.



Waihon A. Kwong received the B.S. and M.S. degrees in electrical engineering from The Ohio State University, Columbus, in 1992 and 1994, respectively. He has worked as a project engineer at Sun's Engineering, Inc., designing quality control systems in paper and pulp industry. He is presently employed by Epsilon Lambda Electronics Corp., Chicago, IL, as a design engineer where he is using DSP's for a radar detection and identification system for automotive applications.



Kevin M. Passino (S'79–M'90) received the the B.S.E.E. from Tri-State University, Angola, IN, in 1983, and the M.S. and Ph.D. degrees in electrical engineering from the University of Notre Dame, South Bend, IN, in 1985 and 1989, respectively.

He has worked in the control systems group at Magnavox Electronic Systems Co., Ft. Wayne, IN, on research in missile control and at McDonnell Aircraft Co., St. Louis, MO, on research in flight control. He spent a year at Notre Dame as a Visiting Assistant Professor, and is currently an Associate Professor, Department of Electrical Engineering, The Ohio State University. His research interests include intelligent and autonomous control techniques, nonlinear analysis of intelligent control systems, failure detection and identification systems, and scheduling and stability analysis of flexible manufacturing systems.

Dr. Passino is an Associate Editor for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL; served as the Guest Editor for the 1993 IEEE *Control Systems Magazine* Special Issue on Intelligent Control; he is currently a Guest Editor for a special track of papers on Intelligent Control for *IEEE Expert Magazine*, and is on the Editorial Board of the *International Journal for Engineering Applications of Artificial Intelligence*. He is the Publicity Co-Chair for the IEEE Conference on Decision and Control in Japan in 1996; was a Program Chairman for the *8th IEEE Int. Symp. on Intelligent Control*, 1993; he served as the Finance Chair for the *9th IEEE Int. Symp. n Intelligent Control*; and he is serving as the General Chair for the *11th IEEE Int. Symp. on Intelligent Control*. He is co-editor (with P. J. Antsaklis) of the book *An Introduction to Intelligent and Autonomous Control*, Kluwer Academic Press, 1993. He is a member of the IEEE Control Systems Society Board of Governors.