Fuzzy Model Reference Learning Control*

Jeffery R. Layne† and Kevin M. Passino‡

Department of Electrical Engineering
The Ohio State University
2015 Neil Avenue
Columbus, OH 43210

Abstract

A "learning system" possesses the capability to improve its performance over time by interaction 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. In this brief paper, we introduce a learning controller that is developed by synthesizing several basic ideas from fuzzy set and control theory, self-organizing control, and conventional adaptive control. We utilize a learning mechanism which observes the plant outputs and adjusts the membership functions of the rules in a direct fuzzy controller so that the overall system behaves like a "reference model". The effectiveness of this "fuzzy model reference learning controller" (FMRLC) is illustrated by showing that it can achieve high performance learning control for a nonlinear time-varying rocket velocity control problem and a multi-input multi-output (MIMO) two degree-of-freedom robot manipulator.

I Introduction

Over recent years, fuzzy control has emerged as a practical alternative to classical control schemes when one is interested in controlling certain time-varying, non-linear, and ill-defined processes. There have in fact been several successful commercial and industrial applications of fuzzy control [1] - [5]. Despite this success, there exist several significant drawbacks of this approach:

1. The design of fuzzy controllers is usually performed in an *ad hoc* manner; hence, it is often not clear exactly how to justify the choices for many parameters in the fuzzy controller (e.g., the membership functions, defuzzification strategy, and fuzzy inference strategy).

2. The fuzzy controller constructed for the nominal plant may later perform inadequately if significant and unpredictable plant parameter variations, structural changes, or environmental disturbances occur.

In this paper a "learning" control algorithm is presented which helps to resolve some of these fuzzy controller design issues. This algorithm employs a reference model (a model of how you would like the plant to behave) to provide closed-loop performance feedback for synthesizing and tuning a fuzzy controller's

*Bibliographic information for this paper: Layne J.R., Passino K.M., "Fuzzy Model Reference Learning Control," Journal of Intelligent and Fuzzy Systems, Vol. 4, No. 1, pp. 33–47, 1996.

†J. Layne gratefully acknowledges the support of the U.S. Air Force Palace Knight Program. J. Layne has recently joined Wright Laboratories, WL/AAAS-3, Wright Patterson AFB, Ohio 45433-6543

‡K. Passino was supported in part by an Engineering Foundation Research Initiation Grant and in part by National Science Foundation Grant IRI-9210332. Please address all correspondence to K. Passino (614-292-5716, email: passino@ee.eng.ohio-state.edu).

knowledge-base. Consequently, this algorithm is referred to as a "fuzzy model reference learning controller" (FMRLC). The FMRLC grew from research on how to improve Procyk and Mamdani's linguistic self-organizing controller (SOC) [6] by utilizing certain general ideas in conventional adaptive control [7, 8]. The first advantage that the FMRLC has over the SOC is that it does not rely on the specification of an explicit inverse model of the process (which can be difficult/impossible to determine for many applications). In addition, the performance criteria for the linguistic SOC can only characterize what is essentially a compromise between rise-time and overshoot (and not the relative importance of each) and hence it provides little flexibility in specifying what performance is to be achieved/maintained (this is the case even if the "optimized" fuzzy performance evaluator introduced in [9, 10, 11] is used). Via the use of a reference model, in the FMRLC framework we incorporate a capability for accurately quantifying virtually any form of desired performance. Next, note that the knowledge-base modification algorithm of Procyk and Mamdani [6] relies on modification of a fuzzy relation table which describes the relationship between the fuzzy controller inputs and outputs. Often, this automatically implies that all input and output universes of discourse must be quantized into discrete levels to implement the fuzzy relation in a computer. Unfortunately, this will generally result in large memory requirements and computational demands since a fuzzy relation table often contains many entries for real world applications (some progress has been made at addressing the computational complexity of knowledge-base modification for the SOC in [9]). In this article, we use a knowledge-base modification algorithm (similar to the one in [11]) which reduces computation time and memory requirements by utilizing a rule base array table rather than a fuzzy relation table. The knowledge-base modification approach is flexible enough to be used in both the conventional SOC approach and the FMRLC (this is shown in [12]). Finally, we note that the linguistic SOC has been used in robotics applications [9, 10], motor and temperature control [13], blood pressure control [14], and in satellite control [15, 16, 17]. While in this paper we describe the application of the FMRLC to robotics and a rocket velocity control problem (where there is a significant underlying process variation resulting from fuel consumption as the rocket launches), the FMRLC has also recently been used for (i) control of a cart-pendulum system where certain improvements over SOC were illustrated [12], (ii) anti-skid brake system control to enhance performance when there are significant variations in the road conditions [18, 19], (iii) cargo ship steering where in [20, 21] it is shown to have certain advantages over conventional model reference adaptive control, (iv) vibration damping in a two-link flexible robot where in [22] the authors develop a fuzzy controller and show how its performance can be enhanced if it is tuned with the FMRLC (experimental results are also provided for both the direct fuzzy controller and the FMRLC to illustrate its ability to compensate for the effects of a payload variation), and (v) for aircraft control law reconfiguration in case of failures [23].

Using conventional adaptive control terminology, the FMRLC and SOC are "direct" adaptive control schemes since they directly update the parameters of the controller without explicit identification of the plant parameters. Other relevant literature that focuses on direct adaptive fuzzy control includes the work in [24] where an adaptive fuzzy system is developed for a continuous casting plant, and the approach in [25] where a fuzzy system adapts itself to driver characteristics for an automotive speed control device.

The use of fuzzy systems for estimation/identification [26, 27, 28, 29, 30, 31, 32] is relevant, especially if "indirect adaptive" [7, 8] fuzzy control techniques (i.e., ones where plant parameters are identified and used to tune the parameters of the controller) such as those in [33, 34, 35] are used. Also, it is interesting to note that in [30, 31, 34] there are inherent uses of inverse dynamics of the plant; however, our use of the fuzzy inverse model is significantly different. Finally, the authors note that since the initial results in FMRLC have been introduced in [21] some other relevant new adaptive/learning techniques have been developed [36, 37, 38, 39] where neural approaches are used to tune fuzzy systems and one fuzzy adaptive control scheme is shown to be stable.

In Section II, the detailed description of the FMRLC algorithm is presented. Then in Section III we study the performance of the FMRLC for single stage rocket velocity control where there is a significant variation in the process dynamics due to the change in mass of the rocket as fuel is expended. Moreover, the FMRLC will be used as a learning controller for a two degree-of-freedom robot manipulator to illustrate the application of FMRLC for a multi-input, multi-output (MIMO) process. Finally, in the concluding remarks in Section IV we will discuss the advantages and disadvantages of FMRLC and highlight some important future research directions.

II Fuzzy Model Reference Learning Control

In this Section, we present a novel learning control technique that was developed by extending some of the linguistic self-organizing control concepts presented by Procyk and Mamdani in [6] and by utilizing ideas from conventional "model reference adaptive control" (MRAC) [8, 7]. The learning control technique, which is shown in Figure 1, utilizes a learning mechanism that: (i) observes data from a fuzzy control system, (ii) characterizes its current performance, and (iii) automatically synthesizes and/or adjusts the fuzzy controller so that some pre-specified performance objectives are met. These performance objectives are characterized via the *reference model* shown in Figure 1. In an analogous manner to conventional MRAC [8, 7] where conventional (often linear) controllers are adjusted, the learning mechanism seeks to adjust the fuzzy controller (a nonlinear controller) so that the closed-loop system (the map from $\underline{y}_r(kT)$ to $\underline{y}(kT)$) acts like a pre-specified reference model (the map from $\underline{y}_r(kT)$ to $\underline{y}_m(kT)$). We have named the new learning control technique "fuzzy model reference learning control" (FMRLC) due to its similarities to MRAC, its unique approach to remembering the adjustments it makes, and according to the prevailing definition of "learning" and "adaptive" [40] (we avoid the term "self-organizing" due to the differences between our approach and the work in [6]). For a much more detailed discussion of the key issues encountered when studying the comparison between adaptive and learning systems see [40, 21]. Next we describe each component of the FMRLC in more detail.

A. The Fuzzy Controller

The process in Figure 1 is assumed to have r inputs denoted by the r - dimensional vector $\underline{u}(kT) = [u_1(kT) \ldots u_r(kT)]^t$ (T is the sample period) and s outputs denoted by the s - dimensional vector $\underline{y}(kT) =$

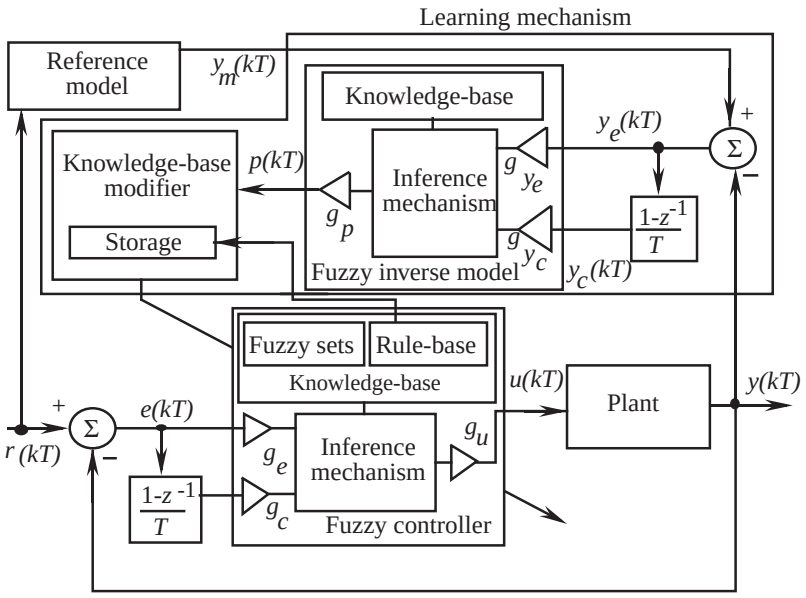

Figure 1: Functional Architecture for the FMRLC.

$[y_1(kT)\ ...\ y_s(kT)]^t$. Most often the inputs to the fuzzy controller are generated via some linear function of the plant output $\underline{y}(kT)$ and reference input $\underline{y}_r(kT)$. Figure 1 shows a special case of such a linear map that was found to be useful in many applications. The inputs to the fuzzy controller are the error $\underline{e}(kT) = [e_1(kT)\ ...\ e_s(kT)]^t$ and change in error $\underline{c}(kT) = [c_1(kT)\ ...\ c_s(kT)]^t$ defined as

$$\underline{e}(kT) = \underline{y}_r(kT) - \underline{y}(kT), \tag{1}$$

$$\underline{c}(kT) = \frac{\underline{e}(kT) - \underline{e}(kT-T)}{T}, \tag{2}$$

respectively, where $\underline{y}_r(kT) = [y_{r_1}(kT)\ ...\ y_{r_s}(kT)]^t$ denotes the desired process output.

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 $\underline{g}_e$, $\underline{g}_c$, and $\underline{g}_u$ were employed to normalize the universe of discourse for the error $\underline{e}(kT)$, change in error $\underline{c}(kT)$, and controller output $\underline{u}(kT)$, respectively (e.g., $\underline{g}_e = [g_{e_1}, ..., g_{e_s}]^t$ so that $g_{e_i} e_i(kT)$ is a scaled input to the fuzzy controller).

For convenience we utilize r multi-input single output (MISO) fuzzy controllers (one for each process input u_n) as it is equivalent to using one s input r output MIMO fuzzy controller. The knowledge-base for the fuzzy controller associated with the n^{th} process input is generated from **IF-THEN** control rules of the form:

$$\textbf{If } \tilde{e}_1 \text{ is } \tilde{E}_1^j \textbf{ and } ... \textbf{ and } \tilde{e}_s \text{ is } \tilde{E}_s^k \textbf{ and } \tilde{c}_1 \text{ is } \tilde{C}_1^l \textbf{ and } ... \textbf{ and } \tilde{c}_s \text{ is } \tilde{C}_s^m \textbf{ Then } \tilde{u}_n \text{ is } \tilde{U}_n^{j,...,k,l,...,m},$$

where $\tilde{e}_a$ and $\tilde{c}_a$ denote the *linguistic variables* associated with controller inputs e_a and c_a, respectively, $\tilde{u}_n$ denotes the linguistic variable associated with the controller output u_n, $\tilde{E}_a^b$ and $\tilde{C}_a^b$ denote the b^{th} *linguistic values* associated with $\tilde{e}_a$ and $\tilde{c}_a$, respectively, and $\tilde{U}_n^{j,...,k,l,...,m}$ denotes the *consequent linguistic*

value associated with $\tilde{u}_n$ for the rule listed above. The above control rule may be quantified by utilizing fuzzy set theory to obtain a fuzzy implication of the form:

$$\textbf{If } E_1^j \textbf{ and } ... \textbf{ and } E_s^k \textbf{ and } C_1^l \textbf{ and } ... \textbf{ and } C_s^m \textbf{ Then } U_n^{j,...,k,l,...,m},$$

where E_a^b, C_a^b, and $U_n^{j,...,k,l,...,m}$ denote the fuzzy sets that quantify the *linguistic statements* "$\tilde{e}_a$ is $\tilde{E}_a^b$", "$\tilde{c}_s$ is $\tilde{C}_s^m$", and "$\tilde{u}_n$ is $\tilde{U}_n^{j,...,k,l,...,m}$", respectively. This fuzzy implication can be represented by a fuzzy relation

$$R_n^{j,...,k,l,...,m} = (E_1^j \times ... \times E_s^k) \times (C_1^l \times ... \times C_s^m) \times U_n^{j,...,k,l,...,m}. \tag{3}$$

The fuzzy controller decision mechanism for this control rule may be expressed by

$$\begin{aligned} \hat{U}_n^{j,...,k,l,...,m}(kT) = ((\hat{E}_1(kT) \times \hat{E}_2(kT) \times ... \times \hat{E}_s(kT)) \times \\ (\hat{C}_1(kT) \times \hat{C}_2(kT) \times ... \times \hat{C}_s(kT))) \circ R_n^{j,...,k,l,...,m} \end{aligned} \tag{4}$$

where $\hat{E}_j(kT)$ and $\hat{C}_j(kT)$ denote the fuzzified error and change in error, respectively, associated with the j^{th} element of $\underline{e}(kT)$ or $\underline{c}(kT)$, $\hat{U}_n^{j,...,k,l,...,m}(kT)$ denotes the implied fuzzy set, and "$\circ$" denotes Zadeh's Composition. See [41] for a more detailed mathematical explanation of Equation 4. Typically in fuzzy system design, a fuzzy implication exists for every possible combination of fuzzy sets describing the inputs to the fuzzy system. Therefore, the fuzzy controller is made up of many fuzzy implications whose overall control action may be computed by the "center of gravity" (COG) method expressed as

$$u_n(kT) = \frac{\sum_{j,...,k,l,...,m} \hat{A}_n^{j,...,k,l,...,m}(kT)\ \hat{c}_n^{j,...,k,l,...,m}(kT)}{\sum_{j,...,k,l,...,m} \hat{A}_n^{j,...,k,l,...,m}(kT)}, \tag{5}$$

where $\hat{A}_n^{j,...,k,l,...,m}(kT)$ and $\hat{c}_n^{j,...,k,l,...,m}(kT)$ are the area and center of area, respectively, of the membership function associated with $\hat{U}_n^{j,...,k,l,...,m}(kT)$.

B. The Reference Model

The reference model provides a capability for quantifying the desired performance of the process. In general, the reference model may be any type of dynamical system (linear or non-linear, time-invariant or time-varying, discrete or continuous time, etc.). The performance of the overall system is computed with respect to the reference model by generating an error signal $\underline{y}_e(kT) = [y_{e_1}\ ...\ y_{e_s}]^t$ where

$$\underline{y}_e(kT) = \underline{y}_m(kT) - \underline{y}(kT). \tag{6}$$

Given that the reference model characterizes design criteria such as stability, rise time, overshoot, settling time, etc. and the input to the reference model is the reference input $\underline{y}_r(kT)$, the desired performance of the controlled process is met if the learning mechanism forces $\underline{y}_e(kT)$ to remain very small for all time. Hence, the error $\underline{y}_e(kT)$ provides a characterization of the extent to which the desired performance is met at time $t = kT$. If the performance is met ($\underline{y}_e(kT) \approx 0$) then the learning mechanism will not make significant modifications to the fuzzy controller. On the other hand if $\underline{y}_e(kT)$ is big, the desired performance is not achieved and the learning mechanism must adjust the fuzzy controller.

C. 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 based on 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 necessary changes in the process output, as expressed by $\underline{y}_e(kT)$, to the relative changes in to process inputs (denoted by $\underline{p} = [p_1 \ ... \ p_r]^t$) necessary to achieve these process output changes. The knowledge-base modifier performs the function of modifying the fuzzy controller's knowledge-base to affect the needed changes in the process inputs. More details of this process are discussed next.

The Fuzzy Inverse Model

The *fuzzy inverse model* was developed by investigating methods to alleviate the problems with using the inverse process model in the linguistic SOC framework of Procyk and Mamdani [6]. Procyk and Mamdani's use of the inverse process model depended on the use of an explicit mathematical model of the process and ultimately assumptions about the underlying physical process. This dependence on a mathematical model of the process often causes significant difficulties in applying their approach (e.g., they are often forced to assume that the plant will act like a constant gain (matrix) and hope that the adaptation mechanism can compensate for this inaccuracy).

Using the fact that most often a control engineer will know how to roughly characterize the inverse model of the plant, we introduce the idea of using a fuzzy system to represent the inverse plant dynamics. We emphasize that it is not necessary to accurately characterize the inverse dynamics; only an approximate representation is needed. This "fuzzy inverse model" as it is shown in Figure 1, simply maps $\underline{y}_e(kT)$, and possibly other parameters such as the functions of $\underline{y}_e(kT)$ and the process operating conditions, to the necessary changes in the process inputs. Hence, the fuzzy inverse model *is used to characterize how to change the plant inputs to force the plant output* $\underline{y}(kT)$ *to be as close as possible to* $\underline{y}_m(kT)$ (i.e., to make $\underline{y}_e(kT)$ small). Again, we use r MISO fuzzy inverse models. While there exist numerous possible combinations of inputs to the fuzzy inverse model, in Figure 1 only error $\underline{y}_e(kT)$ and the change in error $\underline{y}_c(kT)$ are shown for the sake of brevity (e.g. delayed versions and functions of the variables could also be used). In this paper, we will assume $\underline{y}_e(kT)$ and $\underline{y}_c(kT)$ are always employed as inputs to the fuzzy inverse model. The reasons for using the change in the desired output change is to provide some "damping" in the learning mechanism. In other words, since we have information about the rate of change of the desired output changes, we may quantify that a small value of y_{e_i} with a small change in y_{e_i} is more desirable than a small value of y_{e_i} with a large change in y_{e_i} due to the fact that overshoot is likely to occur. (Using reasoning along similar lines the authors in [22] explain how to view the fuzzy inverse model as a controller in the adaptation loop and show how this perspective can be used in FMRLC design.)

Note that similar to the fuzzy controller, the fuzzy inverse model shown in Figure 1 contains normalizing scaling factors, namely $\underline{g}_{y_e}$, $\underline{g}_{y_c}$, and $\underline{g}_p$, for each universe of discourse. Selection of the normalizing gains

can impact the overall performance of the system and a gain selection procedure is given below in Section D.

The knowledge-base for the fuzzy inverse model associated with the n^{th} process input is generated from fuzzy implications of the form: **If** $Y_{e_1}^j$ **and** ... **and** $Y_{e_s}^k$ **and** $Y_{c_1}^l$ **and** ... **and** $Y_{c_s}^m$ **Then** $P_n^{j,\ldots,k,l,\ldots,m}$, where $Y_{e_a}^b$ and $Y_{c_a}^b$ denote the b^{th} fuzzy set associated with the error y_{e_a} and change in error y_{c_a}, respectively, and associated with the a^{th} process output and $P_n^{j,\ldots,k,l,\ldots,m}$ denotes the consequent fuzzy set for this rule describing the necessary change in the n^{th} process input. This fuzzy implication can be represented by a fuzzy relation $S_n^{j,\ldots,k,l,\ldots,m} = (Y_{e_1}^j \times \ldots \times Y_{e_s}^k) \times (Y_{c_1}^l \times \ldots \times Y_{c_s}^m) \times P_n^{j,\ldots,k,l,\ldots,m}$. The fuzzy inverse model decision mechanism for this fuzzy implication may be expressed by $\hat{P}_n^{j,\ldots,k,l,\ldots,m}(kT) = ((\hat{Y}_{e_1}(kT) \times \hat{Y}_{e_2}(kT) \times \ldots \times \hat{Y}_{e_s}(kT)) \times (\hat{Y}_{c_1}(kT) \times \hat{Y}_{c_2}(kT) \times \ldots \times \hat{Y}_{c_s}(kT))) \circ S_n^{j,\ldots,k,l,\ldots,m}$ where $\hat{Y}_{e_p}(kT)$ and $\hat{Y}_{c_p}(kT)$ denote the fuzzified error and change in error, respectively associated with the p^{th} element of $\underline{y}_e$ and $\underline{y}_c$, $\hat{P}_n^{j,\ldots,k,l,\ldots,m}(kT)$ denotes the implied fuzzy set for this fuzzy implication describing input changes for the n^{th} process input (actually the n^{th} direct fuzzy controller). As was the case for the direct fuzzy controller, the overall input changes for the n^{th} direct fuzzy controller $p_n(kT)$ are determined from the COG defuzzification method.

A typical rule base array which may be employed in a fuzzy inverse model for a SISO process is shown in Table 1 below (this and other fuzzy inverse models are used in the applications) where Y_e^j and Y_c^k denote the fuzzy sets associated with $y_e(kT)$ and $y_c(kT)$, respectively, and the $P_i^{j,k}$ denote the fuzzy sets quantifying the desired process input change $p_i(kT)$. Note that the body of Table 1 lists the center values for convex, symmetric, and normal membership functions that are defined on universes of discourse normalized to $[-1, 1]$.

Table 1: Typical rule base array table for the fuzzy inverse model.

$P_i^{j,k}$		Y_c^k										
		-5	-4	-3	-2	-1	$+0$	$+1$	$+2$	$+3$	$+4$	$+5$
Y_e^j	-5	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-0.8	-0.6	-0.4	-0.2	0.0
	-4	-1.0	-1.0	-1.0	-1.0	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	$+0.2$
	-3	-1.0	-1.0	-1.0	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	$+0.2$	$+0.4$
	-2	-1.0	-1.0	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$
	-1	-1.0	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$
	0	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$	$+1.0$
	$+1$	-0.8	-0.6	-0.4	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$	$+1.0$	$+1.0$
	$+2$	-0.6	-0.4	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$	$+1.0$	$+1.0$	$+1.0$
	$+3$	-0.4	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$	$+1.0$	$+1.0$	$+1.0$	$+1.0$
	$+4$	-0.2	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+1.0$
	$+5$	0.0	$+0.2$	$+0.4$	$+0.6$	$+0.8$	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+1.0$

The fuzzy inverse model rule base array shown in Table 1 was designed to take advantage of the "damping" feature described above. For example, consider the case where $y_e(kT) = 0$ which is best characterized by fuzzy set Y_e^j where $j = 0$ since it characterizes the case where $y_e(kT)$ is small. The best change in $y_e(kT)$, is $y_c(kT) = 0$ which is characterized in a similar way by Y_c^k where $k = 0$. This zero point (i.e., the center of Table 1, $j = k = 0$) represents a point where the fuzzy inverse model indicates that no change in the input is required to force $\underline{y}(kT) = \underline{y}_m(kT)$ since this equality is already achieved. If for some time k,

we have $j = 0$ but $k = -2$ (i.e., currently $\underline{y}(kT)$ is close to $\underline{y}_m(kT)$ but $\underline{y}(kT)$ is increasing above $\underline{y}_m(kT)$) then Table 1 indicates that for $P_i^{j,k}$, the center of the fuzzy set is at -0.4 which characterizes the fact that a negative increment should be added to the process input to ensure that $\underline{y}$ will not continue to increase (i.e., so that we maintain a small y_{e_i}). Similar statements hold for the remaining elements in Table 1.

It is important to note that: (i) development of the fuzzy inverse model does not depend on the existence and specification of the mathematical model of the plant or its inverse (i.e., the plant inverse need not exist), (ii) the fuzzy inverse model should not be confused with the mathematical model of the inverse of the plant that is sometimes used in fixed (i.e., non-adaptive) control where the controller has no ability to synthesize itself or auto-tune in response to plant parameter changes, and (iii) while the above discussion provides some general guidelines for the construction of the fuzzy inverse model, and the applications in Section III show how to construct it for a rocket velocity control problem with time-varying parameters and a multi-input multi-output robot control problem, if the plant is very complex then it can sometimes be difficult to specify the fuzzy inverse model. To gain further insight into how to specify the fuzzy inverse model see [22] where a fuzzy inverse model is developed for a FMRLC which is implemented on a complex flexible robotic mechanism (the perspective used there is that the fuzzy inverse model acts as a controller in the adaptation loop). Moreover, see [20, 18] for the details on how to specify the fuzzy inverse model for the FMRLC for a cargo ship steering application and a anti-skid braking problem. Overall, the FMRLC's performance depends on the engineer's ability to specify a fuzzy inverse model. For the applications listed above we have found that the fuzzy inverse model is relatively easy to specify and that it does not need to be extremely accurate since the learning mechanism tends to compensate for the inaccuracies. The general guidelines given above, coupled with the applications studied in this paper and in [22, 20, 18], provide significant insights into choosing the fuzzy inverse model so that the FMRLC will be useful for other applications. Finally, it is interesting to note that other inherent uses of inverse dynamics of the plant in adaptive fuzzy control schemes can be found in [30, 31, 34].

The Knowledge-Base Modifier

The knowledge-base modifier presented next grew from work that focused on improving the knowledge-base modification approach for the linguistic SOC [6]. In the linguistic SOC framework, knowledge-base modification was performed on the overall fuzzy relation ($R_n = \bigcup_{j,\ldots,k,l,\ldots,m} R_n^{j,\ldots,k,l,\ldots,m}$) used to implement the fuzzy controller. However, this method of knowledge-base modification can be computationally complex due to the fact that R_n is generally a very large array. Here we use a knowledge-base modification algorithm (similar to the one in [11]) which increases computational efficiency by modifying the membership functions of consequent fuzzy sets $U_n^{j,\ldots,k,l,\ldots,m}$ rather than the fuzzy relation array R_n.

The knowledge-base modifier performs the function of modifying the fuzzy controller so that better performance is achieved. Given the information about the necessary changes in the input as expressed by the vector $\underline{p}(kT)$ from the fuzzy inverse model, the knowledge-base modifier changes the knowledge-base of the fuzzy controller so that the previously applied control action will be modified by the amount $\underline{p}(kT)$. Therefore, consider the previously computed control action, which contributed to the present good/bad system performance. Note that $\underline{e}(kT-T)$ and $\underline{c}(kT-T)$ would have been the process error and change in error, respectively, at that time. Likewise, $\underline{u}(kT-T)$ would have been the controller output at that time.

The controller output which would have been desired is expressed by

$$\bar{\underline{u}}(kT-T) = \underline{u}(kT-T) + \underline{p}(kT). \tag{7}$$

Next we will show that by modifying the fuzzy controller's knowledge-base we may force the fuzzy controller to produce this desired output given similar controller inputs.

Assume that only symmetric membership functions are defined for the fuzzy controller's output so that $c_n^{j,\ldots,k,l,\ldots,m}$ denotes the center value of the membership function associated with the fuzzy set $U_n^{j,\ldots,k,l,\ldots,m}$. Knowledge-base modification is performed by shifting centers of the membership functions of the fuzzy sets $U_n^{j,\ldots,k,l,\ldots,m}$ which are associated with the fuzzy implications that contributed to the previous control action $\underline{u}(kT-T)$. This modification involves shifting these membership functions by an amount specified by $\underline{p}(kT) = [p_1(kT)\ \ldots\ p_r(kT)]^t$ so that

$$c_n^{j,\ldots,k,l,\ldots,m}(kT) = c_n^{j,\ldots,k,l,\ldots,m}(kT-T) + p_n(kT). \tag{8}$$

The degree of contribution for a particular fuzzy implication in the fuzzy controller whose fuzzy relation is denoted $R_n^{j,\ldots,k,l,\ldots,m}$ is determined by its "activation level", defined

$$\begin{aligned}\delta_n^{j,\ldots,k,l,\ldots,m}(t) = \min\{&\mu_{E_1^j}(e_1(t)), \ldots, \mu_{E_s^k}(e_s(t)),\\ &\mu_{C_1^l}(c_1(t)), \ldots, \mu_{C_s^m}(c_s(t))\},\end{aligned} \tag{9}$$

where μ_A denotes the membership function of the fuzzy set A and t denotes the current time. Only those fuzzy implications $R_n^{j,\ldots,k,l,\ldots,m}(kT-T)$ whose activation level $\delta_n^{j,\ldots,k,l,\ldots,m}(kT-T) > 0$ are modified. All others remain unchanged (this allows for local learning and hence memory for our learning controller [40, 21]).

Consider the effect that the above knowledge-base modification has on the COG defuzzification (for the direct fuzzy controller) expressed in Equation 5. Notice that since the area of the implied fuzzy sets is proportional to the "activation level" of the fuzzy relation (i.e., $A_n^{j,\ldots,k,l,\ldots,m}(kT-T) \propto \delta_n^{j,\ldots,k,l,\ldots,m}(kT-T)$) only those fuzzy relations whose activation levels are greater than zero affect the center of gravity, or control action. Furthermore, notice that since symmetric membership functions are utilized, a shift in the membership function associated with fuzzy set $U_n^{j,\ldots,k,l,\ldots,m}(kT)$ will also shift, by the same amount, the centers of the membership functions associated with the previous implied fuzzy sets $\hat{U}_n^{j,\ldots,k,l,\ldots,m}(kT-T)$. Therefore, given the previous controller inputs $\underline{e}(kT-T)$ and $\underline{c}(kT-T)$ and the new fuzzy relation $R_n^{j,\ldots,k,l,\ldots,m}(kT)$ obtained after shifting the consequent fuzzy set, the new center value of the membership function associated with the implied fuzzy set $\hat{U}_n^{j,\ldots,k,l,\ldots,m}(kT-T)$ is expressed as

$$\bar{\hat{c}}_n^{j,\ldots,k,l,\ldots,m}(kT-T) = \hat{c}_n^{j,\ldots,k,l,\ldots,m}(kT-T) + p_n(kT). \tag{10}$$

Substituting this new center value into Equation 5 we obtain

$$\bar{u}_n(kT-T) = \frac{\sum_{j,\ldots,k,l,\ldots,m} A_n^{j,\ldots,k,l,\ldots,m}(kT-T)\ \bar{\hat{c}}_n^{j,\ldots,k,l,\ldots,m}(kT-T)}{\sum_{j,\ldots,k,l,\ldots,m} A_n^{j,\ldots,k,l,\ldots,m}(kT-T)} \tag{11}$$

where $\bar{u}_n(kT-T)$ is the new control action that is obtained. Simplifying, it is easy to see that

$$\bar{u}_n(kT-T) = u_n(kT-T) + p_n(kT), \tag{12}$$

which is the desired effect. Notice that this approach also achieves "generalization" as it is called in learning theory [40] since it will at the same time learn how to deal with values that are near to those considered (i.e., near to $\underline{e}(kT)$, $\underline{c}(kT)$, and $\underline{y}(kT)$). Along these lines it is interesting to note that our knowledge-base modification procedure implements a form of *local adaptation* and hence utilizes memory. Different parts of the rule-base are "filled in" based on different operating conditions of the system, and when one area of the rule-base is updated, the other rules are not affected. Hence, the controller adapts to new situations and also remembers how it has adapted to past situations (this is why the term "learning" is used).

As an example of the knowledge-base modification procedure, Table 2 shows a knowledge-base array table where its entries represent the center values of symmetric membership functions associated with the implied fuzzy sets $U^{j,k}$ defined on a normalized universe of discourse (normalized to $[-1, 1]$). Given that the fuzzy controller employs a knowledge-base array table similar to Table 2, the process of knowledge-base modification reduces to a simple two step algorithm which is expressed below.

1. Determine which fuzzy implications in the knowledge-base array table contributed to the previously applied input. In other words, determine the fuzzy implications whose premise element has an activation level above zero (i.e., the rule is "on" - its implied fuzzy set is not null).
2. Modify the entries in the knowledge-base array for those fuzzy implications.

Table 2: Typical knowledge-base array table.

$U^{j,k}$		C^k										
		-5	-4	-3	-2	-1	$+0$	$+1$	$+2$	$+3$	$+4$	$+5$
E^j	-5	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$
	-4	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2
	-3	$+1.0$	$+1.0$	$+1.0$	$+1.0$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2	-0.4
	-2	$+1.0$	$+1.0$	$+1.0$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2	-0.4	-0.6
	-1	$+1.0$	$+1.0$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2	-0.4	-0.6	-0.8
	0	$+1.0$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2	-0.4	-0.6	-0.8	-1.0
	$+1$	$+0.8$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2	-0.4	-0.6	-0.8	-1.0	-1.0
	$+2$	$+0.6$	$+0.4$	$+0.2$	$+0.0$	-0.2	-0.4	-0.6	-0.8	-1.0	-1.0	-1.0
	$+3$	$\boxed{+0.4}$	$\boxed{+0.2}$	$+0.0$	-0.2	-0.4	-0.6	-0.8	-1.0	-1.0	-1.0	-1.0
	$+4$	$\boxed{+0.2}$	$\boxed{+0.0}$	-0.2	-0.4	-0.6	-0.8	-1.0	-1.0	-1.0	-1.0	-1.0
	$+5$	$+0.0$	-0.2	-0.4	-0.6	-0.8	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0

For example, assume that the previous process error $e(kT-T)$ took on a value such that the membership functions associated with sets E^{+3} and E^{+4} shown in Table 2 evaluated to be greater than zero. Similarly, assume that the previous change in the process error $c(kT - T)$ was best characterized by the fuzzy sets C^{-4} and C^{-5}. The fuzzy implications which contributed (i.e., had $\delta > 0$) to the previously applied process input are illustrated by the boxed entries in Table 2 (i.e., they are all the implications which contain as left-hand-side elements the boxed entries – 4 rules in this case). Suppose that $p_i(kT) = g_{u_i} p'_i(kT)$ so that $p'_i(kT)$ is the normalized desired change in the process input. For our example, assume that $p'_i(kT) = 0.1$, then after knowledge-base modification for the boxed values 0.4, 0.2, 0.0, and 0.2 in Table 2 we get 0.5, 0.3, 0.1 and 0.3, respectively (all other entries in the Table remain unchanged at this time).

D. FMRLC Design Procedure and Implementation Issues

The design procedure for the FMRLC involves: (i) the specification of a direct fuzzy controller with consequent membership functions that can be tuned (this fuzzy controller can be chosen via conventional (heuristic) fuzzy control design techniques for the nominal plant), (ii) specifying the reference model which characterizes the desired system performance, (iii) specifying the fuzzy inverse model which characterizes how the inputs to the plant should be changed so that the desired performance is achieved, and (iv) selection of the normalizing gains for the fuzzy controller and the fuzzy inverse model. As selection of the normalizing gains for both the fuzzy controller and the fuzzy inverse model can impact the overall performance, next we will provide a procedure to choose these parameters. It is important to note that although it is often not highlighted, most learning/adaptive control approaches require some type of initial choice of controller structure and parameters (e.g., the choice of an adaptation gain or initial controller parameters in a conventional adaptive controller). The gain selection procedure to be presented next provides a systematic methodology to select such initial parameters for the FMRLC.

Due to physical constraints for a given system, the range of values for the process inputs and outputs is generally known from a qualitative analysis of the process. As a result, we can determine the range of values or the universe of discourse for $\underline{e}(kT)$, $\underline{u}(kT)$, $\underline{y}_e(kT)$, and $\underline{p}(kT)$. Consequently, $\underline{g}_e$, $\underline{g}_u$, $\underline{g}_{y_e}$, and $\underline{g}_p$ are chosen so that the appropriate universes of discourse are mapped to $[-1, +1]$. To determine $\underline{g}_c$ we disconnect the adaptive mechanism and pick it using standard fuzzy control system design techniques or by iteratively applying inputs to $\underline{y}_r$, observing $\underline{c}(kT)$, and finding a scaling factors to map the universes of discourse to $[-1, 1]$.

The vector $\underline{g}_{y_c}$ is left as a vector of tuning parameters for the FMRLC. Recall that the scaling factors $\underline{g}_{y_c}$ associated with the change in the desired output changes has the effect of providing "damping" to the controller modifications. Moreover, the "damping" effect is increased as the elements of the scaling factor $\underline{g}_{y_c}$ are increased. A suitable selection of $\underline{g}_{y_c}$ may be obtained by monitoring the response of the overall process with respect to the reference model response. If undesirable oscillations exist between a given process and the associated reference model output response, it is likely that the element of $\underline{g}_{y_c}$ associated with this ouput is too small and should be increased. Likewise, if a given element of $\underline{g}_{y_c}$ is too large, the process will be unable to keep up with the reference model due to the resulting damping.

Below a simple procedure is presented for selecting the gains:

1. Select the controller gains $\underline{g}_e$, $\underline{g}_u$, and $\underline{g}_{y_e}$ so that each universe of discourse is mapped to the interval $[-1, 1]$.

2. Choose the controller gains g_{p_i} to be the same as for the fuzzy controller output gain g_{u_i}. This will allow the $p_i(kT)$ to take on values as large as the largest possible inputs $u_i(kT)$.

3. Using standard fuzzy control design techniques (i.e., ones that use human expertise) or simple experiments choose $\underline{g}_c$ to map the universes of discourse of $\underline{c}(kT)$ to $[-1, 1]$.

4. Assign the numerical value 0 to the scaling factors associated with the changes in the desired output changes (i.e., all elements of $\underline{g}_{y_c}$ are set equal to 0).

5. Apply a step input to the process which is of a magnitude that may be typical for the process during normal operation. Observe the process response and the reference model response.

6. Three cases:

   (a) If there exist unacceptable oscillations in a given process output response about the reference model response, then increase the associated element of $\underline{g}_{y_c}$. Go to step 5.

   (b) If a given process output response is unable to "keep up" with the reference model response, then decrease the associated element of $\underline{g}_{y_c}$. Go to step 5.

   (c) If the process response is acceptable with respect to the reference model response, then the controller design is completed.

For the applications presented in this paper, the above gain selection procedure has proven very successful. However, given that the procedure is a result of practical experience with the FMRLC rather than strict mathematical analysis, it is likely that it will not work for all processes. For some applications (none of the ones we've studied in [21, 12, 19, 20, 18, 22]), the procedure may result in an unstable process. In such situations, it may be necessary to modify other controller parameters such as the controller sampling period T or the number of fuzzy controller rules. Clearly, the stability analysis of the FMRLC is an important research direction.

Note that when implementing (and simulating) the FMRLC one must be concerned with the "curse of dimensionality". Particularly, assume that: (i) the fuzzy controller has s' inputs (where $s' = \alpha s$ with $\alpha = 2$ for the case shown in Figure 1), r outputs, and N membership functions on each of its input universes of discourse; (ii) the fuzzy inverse model has s'' inputs (where $s'' = \beta s$ with $\beta = 2$ for the case shown in Figure 1), r outputs, and M membership functions on each of its input universes of discourse; and (iii) that both the fuzzy controller and the fuzzy inverse model use the maximum number of rules possible (for completeness). In this case there are $r(N^{s'} + M^{s''})$ rules in the FMRLC. As is the case with standard fuzzy control, increasing the number of inputs causes an exponential increase in the number of rules. It is important to note that if one assumes that the membership functions are uniformly distributed across the input universes of discourse so that at most two overlap for any point (this is in fact what we do in all the applications in this paper), then at most $r(2^{s'} + 2^{s''})$ rules will be on at one time and hence the code implementing the FMRLC is much less complex than one might think at first glance. It is in fact this characteristic that we exploit when we implemented the FMRLC for the flexible robotic system in [22] where the FMRLC had 1150 rules and operated with a sampling interval of 15 milliseconds on a 386-based computer.

III Applications

A. Rocket Velocity Control

In this section we illustrate the performance of a FMRLC which is employed to control the velocity of a single stage rocket. A mathematical model for this process is presented by Barrére *et al.* in [42] and Mandell *et al.* in [43] and is expressed by the following differential equation:

$$\frac{d\ v(t)}{dt} = c(t)\left(\frac{m}{M - m\ t}\right) - g_o\left(\frac{R}{R + y(t)}\right) - 0.5\ v^2(t)\left(\frac{\rho_a\ A\ C_d}{M - m\ t}\right), \tag{13}$$

where $v(t)$ is the rocket velocity at time t, $y(t)$ is the altitude of the rocket (above sea level), $c(t)$ is the velocity of the exhaust gases, and for our simulation: (i) $M = 15000.0\ kg$ - initial mass of the rocket and

fuel, (ii) $m = 100.0\ \frac{kg}{s}$ - exhaust gases mass flow rate (approximately constant for some solid propellant rockets), (iii) $A = 1.0\ meter^2$ - maximum cross sectional area of the rocket, (iv) $g_o = 9.8\ \frac{meters}{s^2}$ - the acceleration due to gravity at sea level, (v) $R = 6.37 \times 10^6\ meters$ - radius of the earth, (vi) $\rho_a = 1.21\ \frac{kg}{m^3}$ - density of air, and (vii) $C_d = 0.3$ - drag coefficient for the rocket.

The mathematical model in Equation 13 was developed based on the simple dynamics of a point mass. However, in general, rockets dynamics are studied in the realm of exterior ballistics. This type of analysis often tends to be very complex and falls outside the scope of this paper. However, even in this restricted context the modeled dynamics have characteristics which make for difficult control. For example, due to the loss of fuel resulting from combustion and exhaust the rocket has a time-varying mass. Furthermore, it can be determined by inspection of Equation 13 that the system is a non-linear process. Indeed, the primary purpose for considering this control application is to investigate the capability of the FMRLC algorithm for controlling non-linear time-varying processes.

As stated before, the control objective is to control the velocity of the rocket. To accomplish this task the rocket is assumed to have one input, namely the velocity of the exhaust gases $c(t)$. In general, the exhaust gas velocity is proportional to the cross-sectional area of the nozzle. Consequently, the exhaust gases may be controlled by changing this cross sectional area. However, for this controller implementation, we assume that the dynamics of the actuators which change the nozzle area and the dynamics of the exhaust gases are fast relative the rocket velocity dynamics and therefore may be eliminated from the model.

1. FMRLC Design

The inputs to the fuzzy controller are the velocity error and change in error and the controller output is the velocity of the exhaust gases. In this fuzzy controller design, 11 fuzzy sets are defined for each controller input (using the structure of Figure 1) such that the membership functions are triangular shaped and evenly distributed on the appropriate universe of discourse (of course the outer-most membership functions are trapezoidal). The normalizing controller gains for the error, change error, and the controller output are chosen to be $g_e = \frac{1}{4000}$, $g_c = \frac{1}{2000}$, and $g_u = 10000$, respectively. The fuzzy sets for the fuzzy controller output are also assumed to be triangular shaped with a width of 0.4 on the normalized universe of discourse. The knowledge-base array was initially chosen with all zero entries. The fuzzy controller sampling period was chosen to be $T = 100$ milliseconds.

The reference model for this process was chosen to represent somewhat realistic performance specifications and is expressed by the following differential equation

$$\frac{d\ y_m(t)}{dt} = -0.2\ y_m(t) + 0.2\ y_r(t), \tag{14}$$

where $y_m(t)$ specifies the desired system performance for the rocket velocity $v(t)$ and the input to the reference model $y_r(t)$ is equal to the desired rocket velocity.

The inputs to the fuzzy inverse model include the error and change in error between the reference model and the rocket velocity expressed as $y_e(kT) = y_m(kT) - v(kT)$, and $y_c(kT) = \frac{y_e(kT) - y_e(kT-T)}{T}$, respectively. For these inputs, 11 fuzzy sets are defined with triangular shaped membership functions which are evenly distributed on the appropriate universes of discourse. The normalizing controller gains associated with $y_e(kT)$, $y_c(kT)$, and $p(kT)$ are chosen to be $g_{y_e} = \frac{1}{4000}$, $g_{y_c} = \frac{1}{2000}$, and $g_p = 10000$, respectively. For

the rocket process, for an increase in the exhaust gas velocity we would generally expect an increase in the process output. Consequently, the knowledge-base array shown in Table 1 was employed for the fuzzy inverse model.

2. Simulation Results

The simulation results for the FMRLC of the rocket are shown below in Figure 2. Note that the system

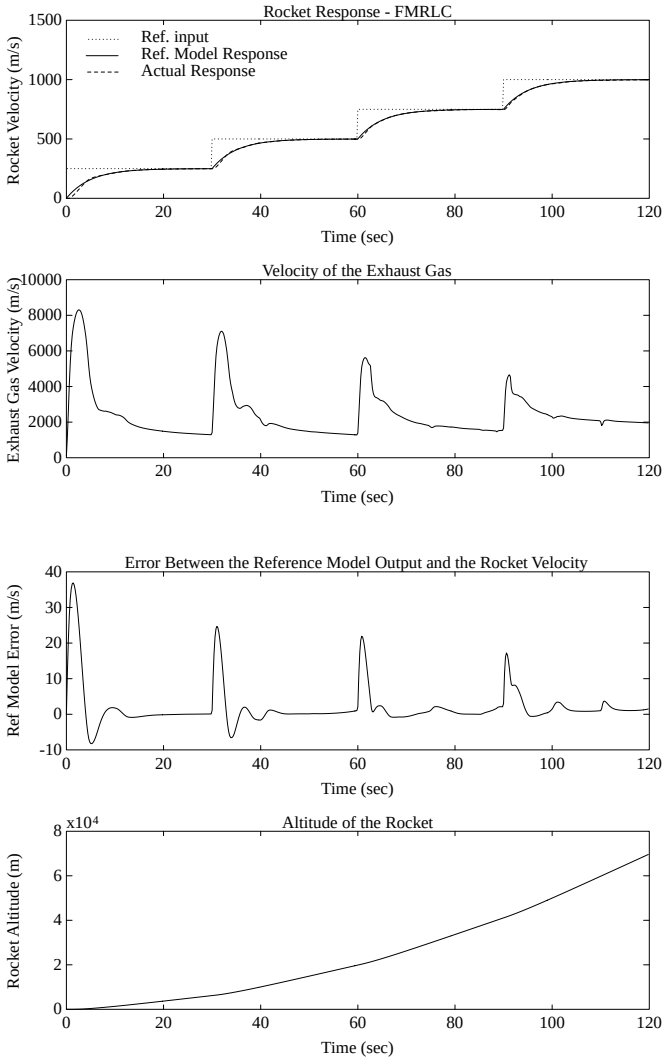

Figure 2: Simulation results for FMRLC control of the rocket system.

exhibits "good" tracking with the reference model even after the mass of the rocket has been reduced significantly from the initial mass due to fuel loss (note that lower amounts of exhaust gas velocity, the control input, are needed as more fuel is used). This application clearly illustrates the effectiveness of the FMRLC algorithm for controlling a nonlinear time varying process.

B. Two-Degree of Freedom Robot Manipulator

Figure 3 illustrates the physical model of a two degree of freedom manipulator. It consists of two links where link #1 is mounted on a rigid base by means frictionless hinge and link #2 is mounted at the end of link #1 by means of a frictionless ball bearing. This control problem is provided to illustrate the

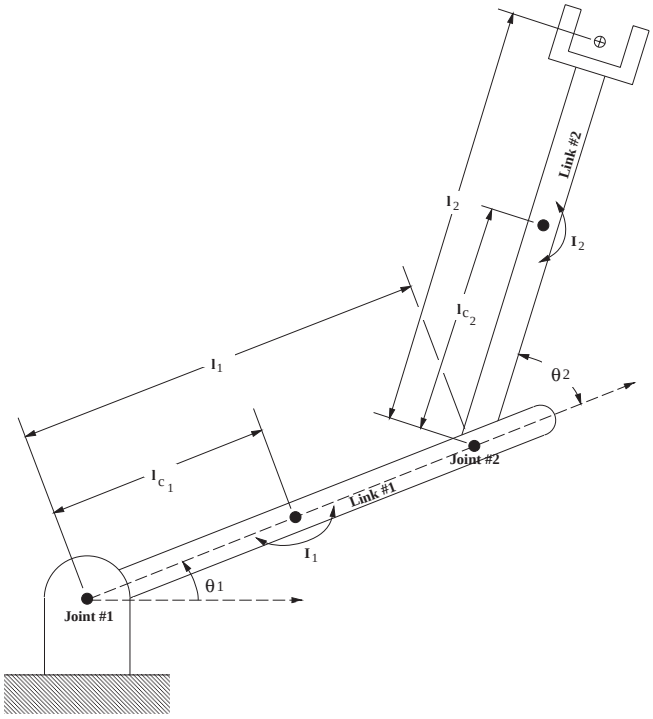

Figure 3: Graphical representation of a 2-link robot.

application of the FMRLC to a nonlinear MIMO system. The inputs to the process are the torques τ_1 and τ_2 which are applied to the links at joints #1 and #2. The outputs are the joint positions θ_1 and θ_2. The model for the robotic system was developed using the well-known Lagrangian equations in classical dynamics and is expressed by the following matrix differential equation [44, 8]:

$$\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -h\dot{\theta}_2 & -h\dot{\theta}_1 - h\dot{\theta}_2 \\ h\dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{15}$$

where

$$H_{11} = m_1 l_{c_1}^2 + I_1 + m_2[l_1^2 + l_{c_2}^2 + 2l_1 l_{c_2}\cos(\theta_2)] + I_2 \tag{16}$$

$$H_{22} = m_2 l_{c_2}^2 + I_2 \tag{17}$$

$$H_{12} = H_{21} = m_2 l_1 l_{c_2}\cos(\theta_2) + m_2 l_{c_2}^2 + I_2 \tag{18}$$

$$h = m_2 l_1 l_{c_2}\sin(\theta_2) \tag{19}$$

$$g_1 = m_1 l_{c_1} g\cos(\theta_1) + m_2 g[l_{c_2}\cos(\theta_1+\theta_2) + l_1\cos(\theta_1)] \tag{20}$$

$$g_2 = m_2 l_{c_2} g\cos(\theta_1+\theta_2), \tag{21}$$

and where $\theta = [\theta_1\ \theta_2]^T$ are the two joint angles, $\tau = [\tau_1\ \tau_2]^T$ are the input joint torques. For purpose of our simulation the robot parameters are given by: (i) $m_1 = 1.0\ kg$ - mass of link #1, (ii) $m_2 = 1.0\ kg$ - mass of link #2, (iii) $l_1 = 1.0\ meters$ - length of link #1, (iv) $l_2 = 1.0\ meters$ - length of link #2, (v) $l_{c_1} = 0.5\ meters$ - distance from joint #1 to the center of gravity of link #1, (vi) $l_{c_2} = 0.5\ meters$ - distance from joint #2 to the center of gravity of link #2, (vi) $I_1 = 0.2\ kg - m^2$ - lengthwise centroidal inertia of link #1, and (vii) $I_2 = 0.2\ kg - m^2$ - lengthwise centroidal inertia of link #2.

1. FMRLC Design

For this application, the process contains two inputs, namely τ_1 and τ_2. Consequently, two MISO fuzzy controllers are needed for this process (one for each process input). The inputs to the fuzzy controller are the robot joint position error $\underline{e} = [e_1\ e_2]^T$ and change in error $\underline{c} = [c_1\ c_2]^T$. The fuzzy controllers have outputs τ_1 for the first controller and τ_2 for the second controller. For both fuzzy controller designs, 11 fuzzy sets are defined for each controller input such that the membership functions are triangular shaped (with base widths of 0.4) and evenly distributed on appropriate universes of discourse (the outer-most membership functions are trapezoidal). Also, the normalizing controller gains for the error, change error, and the controller output are chosen to be $\underline{g}_e = [\frac{1}{2\pi}\ \frac{1}{2\pi}]^T$, $\underline{g}_c = [\frac{1}{20}\ \frac{1}{20}]^T$, and $\underline{g}_u = [100\ 25]^T$, respectively. The knowledge-base array for both fuzzy controllers was initially chosen with all zero entries. The fuzzy controller sampling period was chosen to be $T = 5$ milliseconds.

The reference model for this FMRLC design is given by the following differential equation

$$\begin{bmatrix} \dot{y}_{m_1}(t) \\ \dot{y}_{m_2}(t) \end{bmatrix} = \begin{bmatrix} -0.75 & 0.0 \\ 0.0 & -1.5 \end{bmatrix} \begin{bmatrix} y_{m_1}(t) \\ y_{m_2}(t) \end{bmatrix} + \begin{bmatrix} +0.75 & 0.0 \\ 0.0 & +1.5 \end{bmatrix} \begin{bmatrix} y_{r_1}(t) \\ y_{r_2}(t) \end{bmatrix}, \tag{22}$$

where y_{m_1} and y_{m_2} specify the system performance for θ_1 and θ_2, respectively. For FMRLC implementation, the inputs to the reference model y_{r_1} and y_{r_2} are equal to the desired position of joints #1 and #2, respectively.

For this FMRLC design, two fuzzy inverse models are needed, one for each fuzzy controller. In general, both process inputs will affect both process outputs. However, for this fuzzy inverse model design we will assume that the cross-coupling between the inputs is negligible (i.e., τ_1 affects only θ_1 and τ_2 affects only θ_2). As a result, the input to a given fuzzy inverse model includes the error and change in error between the associated reference model output and robot position. Therefore, for the i^{th} fuzzy inverse model, these inputs may expressed as $y_{e_i}(kT) = y_{m_i}(kT) - \theta_i(kT)$ and $y_{c_i}(kT) = \frac{y_{e_i}(kT) - y_{e_i}(kT-T)}{T}$ respectively. For these inputs, 11 fuzzy sets are defined with triangular shaped membership functions which are evenly distributed on the appropriate universe of discourse. The normalizing fuzzy system gains associated with $\underline{y}_e(kT)$, $\underline{y}_c(kT)$, and $\underline{p}(kT)$ are chosen to be $\underline{g}_{y_e} = [\frac{1}{2\pi}\ \frac{1}{2\pi}]^T$, $\underline{g}_{y_c} = [1\ \frac{1}{2}]^T$, and $\underline{g}_p = [100\ 25]^T$, respectively. For the robot process for an increase in the torque τ_1 we would generally expect an increase in the process output θ_1. Likewise, for an increase in the torque τ_2 we would generally expect an increase in the process output θ_2. Consequently, the knowledge-base array shown in Table 1 was employed for both fuzzy inverse models.

2. Simulation Results

The simulation results for the FMRLC of the two degree-of-freedom robot manipulator are shown below in Figure 4 for joint #1 and Figure 5 for joint #2. Once again the FMRLC provided good

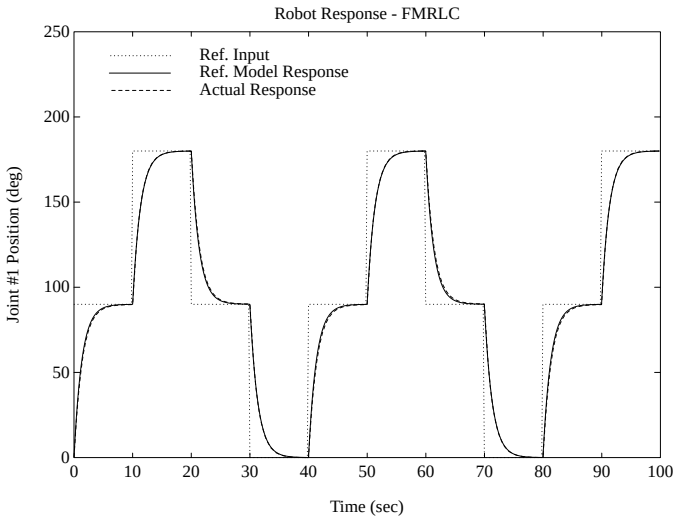

Figure 4: Simulation results for joint #1 of FMRLC controlled robot system.

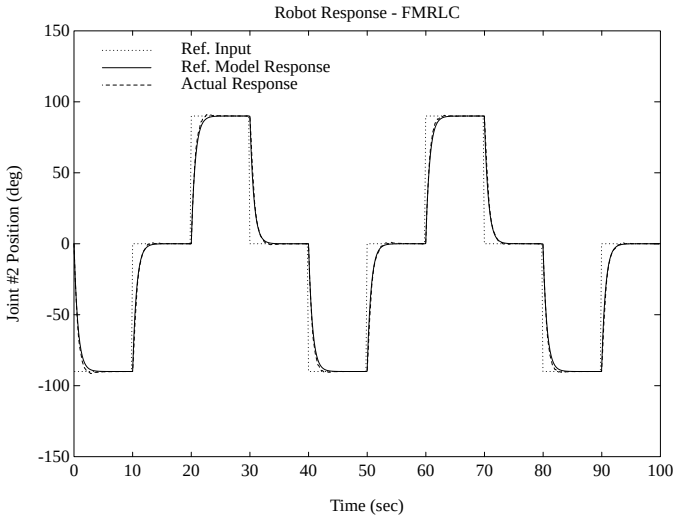

Figure 5: Simulation results for joint #2 of FMRLC controlled robot system.

system tracking with respect to the reference model. As a result, the system exhibits good steady state and transient response. In fact, the response for joint #1 in Figure 4 was so close to the response of the reference model that the two almost perfectly overlap.

IV Concluding Remarks

The principal objectives of this paper were to: (i) introduce the FMRLC, (ii) provide a *design methodology* for the FMRLC, (iii) design a FMRLC for a nonlinear time-varying rocket velocity control problem,

and (iv) develop a MIMO FMRLC for a nonlinear two degree-of-freedom robot manipulator. The key advantages that the FMRLC seems to offer as a learning controller may be summarized as follows:

- A detailed analytical model of the process is not needed to develop the FMRLC.
- The FMRLC provides an automatic method to synthesize a portion of the knowledge-base (specifically, the right-hand-sides of the rules) for the direct fuzzy controller while at the same time it ensures that the system will behave in a desirable fashion (in particular, there is no need to "learn from drastic failures" as is often the case for other learning control techniques - e.g., as is often done for the inverted pendulum).
- The learning/adaptation mechanism in the FMRLC dynamically and continually updates the rule-base in the direct fuzzy controller in response to process parameter variations and/or disturbances (e.g., see the rocket velocity control application). In this way if unpredictable changes occur within the plant, the FMRLC can make on-line adjustments to a direct fuzzy controller to maintain adequate performance levels.

Basically, by combining learning/adaptive control concepts with fuzzy system theory, we have developed a control scheme which often has a fast rate of convergence and often provides an appropriate nonlinear mapping between controller inputs and outputs (i.e., it automatically performs "function approximation" [40] to achieve learning control).

Despite these apparant advantages of the FMRLC algorithm, several drawbacks do exist: (1) the design procedure (e.g., selection of the normalizing gains) tends to be somewhat *ad hoc*, (2) there have been no investigations for the FMRLC (or any other fuzzy adaptive technique) to theoretically show that the fuzzy controller can in fact be tuned so that the performance specified in the reference model can be achieved (this problem is very well studied in conventional adaptive control where linear controllers are tuned so that performance specified in linear reference models is achieved), (3) conditions for stability and convergence of the FMRLC algorithm are yet to be found, (4) *persistent excitation* [7, 8] issues for the FMRLC need to be mathematically investigated (since the reference input affects the ability of the fuzzy controller parameters to converge to values that result in the reference model behavior being achieved), and (5) although it provides certain improvements over SOC (as shown in [12]), the FMRLC algorithm is still computationally intensive. These disadvantages provide several future research directions. For example, future research involving the FMRLC algorithm should include a mathematical analysis of the controller to better quantify the effect of controller design parameters. Perhaps the most important research direction is to perform stability and convergence analysis in the spirit of the extensive and significant contributions in stability analysis of conventional adaptive control [7, 8] and the important recent (actually yet to appear) results in [38]. Such stability analysis can be quite involved as the learning mechanism for the FMRLC adjusts a nonlinear (fuzzy) controller as opposed to the conventional adaptive control case where often linear controllers are adjusted (there is, however, a growing body of literature on adapting nonlinear controllers). Finally, research must be directed towards developing faster algorithms for FMRLC computations to ensure that the FMRLC can be employed in real world applications (along the same lines as was done in [22]).

Acknowledgement: The authors would like to thank the reviewers for their helpful comments.

References

[1] W. Kickert and H. V. N. Lemke, "Application of a fuzzy controller in a warm water plant," *Automatica*, vol. 12, no. 4, pp. 301–308, 1976.

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

[3] J. Bernard, "Use of a rule-based system for process control," *IEEE Control Systems Magazine*, vol. 8, pp. 3–13, October 1988.

[4] Y. Li and C. Lau, "Development of fuzzy algorithms for servo systems," *IEEE Control Systems Magazine*, vol. 9, pp. 65–72, April 1989.

[5] K. Self, "Designing with fuzzy logic," *IEEE Spectrum*, pp. 42–105 and 105, November 1990.

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

[7] K. Åström and B. Wittenmark, *Adaptive Control*. Reading, Massachusetts: Addison-Wesley Publishing Company, 1989.

[8] K. Narendra and A. Annaswamy, *Stable Adaptive Systems*. Englewood Cliffs, New Jersey: Prentice Hall, 1989.

[9] 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*, pp. 41–62, Amsterdam, the Netherlands: M. Sugeno (ed.), 1985.

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

[11] T. Yamazaki, *An improved algorithm for a self-organizing controller and its experimental analysis*. PhD thesis, London University, 1982.

[12] J. Layne and K. Passino, "Fuzzy model reference learning control," *Proceedings of the 1st IEEE Conference on Control Applications*, pp. 686–691, Dayton, Ohio, September 1992.

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

[14] S. Isaka, A. Sebald, A. Karimi, N. Smith, and M. Quinn, "On the design and performance evaluation of adaptive fuzzy controllers," *Proceedings, 1988 IEEE Conference on Decision and Control*, pp. 1068–1069, Austin, Texas, December 1988.

[15] S.Daley and K. F. Gill, "Comparison of a fuzzy logic controller with a P+D control law," *Journal of Dynamical System, Measurement, and Control*, vol. 111, pp. 128–137, June 1989.

[16] S.Daley and K. F. Gill, "Altitude control of a spacecraft using an extended self-organizing fuzzy logic controller," *Proc. I. Mech. E.*, vol. 201, no. 2, pp. 97–106, 1987.

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

[18] J. Layne, K. Passino, and S. Yurkovich, "Fuzzy learning control for anti-skid braking systems," *IEEE Trans. on Control System Technology*, vol. 1, pp. 122–129, June 1993.

[19] J. Layne, K. Passino, and S. Yurkovich, "Fuzzy learning control for anti-skid braking systems," *Proc. IEEE Conf. on Decision and Control*, pp. 2523–2528, Tucson, AZ, December 1992.

[20] J. Layne and K. Passino, "Fuzzy model reference learning control for cargo ship steering," *IEEE Control Systems*, vol. 13, no. 6, pp. 23–34, 1993.

[21] J. Layne, "Fuzzy model reference learning control," Master's thesis, Department of Electrical Engineering, The Ohio State University, March 3 1992.

[22] V. Moudgal, W. Kwong, K. Passino, and S. Yurkovich, "Learning control for a two-link flexible mechanism," *Proc. of the American Control Conference*, pp. Baltimore, MD, June 1994.

[23] W. Kwong and K. Passino, "Fuzzy learning systems for aircraft control law reconfiguration," *Proceedings of the IEEE Int. Symp. on Intelligent Control*, pp. Columbus, Ohio, Aug. 16-18 1994.

[24] 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," *Industrial Application of Fuzzy Control*, pp. 73–86, 1985.

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

[26] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[27] L. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *Proceedings, 1991 IEEE International Symposium on Intelligent Control*, pp. 263–268, Arlington, Virginia, August 1991.

[28] R. M. Tong, "Some properties of fuzzy feedback systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 10, pp. 327–330, June 1980.

[29] A. Cumani, "On a possibilistic approach to the analysis of fuzzy feedback systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 12, pp. 417–422, May/June 1982.

[30] 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.

[31] E. Czogała and W. Pedrycz, "Control problems in fuzzy systems," *Fuzzy Sets and Systems*, vol. 7, pp. 257–273, 1982.

[32] C. Batur, A. Srinivasan, and C. Chan, "Automatic rule based model generation for uncertain complex dynamical systems," *Proceedings, 1991 IEEE International Symposium on Intelligent Control*, pp. 275–279, 1991.

[33] F. V. D. Rhee, H. V. N. Lemke, and J. Dijkman, "Knowledge based fuzzy control of systems," *IEEE Transactions on Automatic Control*, vol. 35, pp. 148–155, February 1990.

[34] P. Graham and R. Newell, "Fuzzy adaptive control of a first-order process," *Fuzzy Sets and Systems*, vol. 31, pp. 47–65, 1989.

[35] P. Graham and R. Newell, "Fuzzy identification and control of a liquid level rig," *Fuzzy Sets and Systems*, vol. 8, pp. 255–273, 1988.

[36] "IEEE Int. Conf. on Fuzzy Systems, San Diego," 1992.

[37] R. Langari and H. Berenji, "Fuzzy logic in control engineering," in *Handbook of Intelligent Control*, pp. 93–140, Van Nostrand Reinhold, New York: D. White and D. Sofge (eds.), 1992.

[38] L. Wang, "Stable adaptive fuzzy control of nonlinear systems," *Proceedings of the IEEE Conf. on Decision and Control*, December 1992.

[39] H. Berenji, "Fuzzy and neural control," in *An Introduction to Intelligent and Autonomous Control Systems* (P. Antsaklis and K. Passino, eds.), Kluwer Academic Publishers; Norwell MA, 1993.

[40] J. Farrell and W. Baker, "Learning control systems," in *An Introduction to Intelligent and Autonomous Control Systems* (P. Antsaklis and K. Passino, eds.), Kluwer Academic Publishers; Norwell MA, 1993.

[41] C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller-part I," *IEEE Trans. on Systems, Man. and Cybernetics*, vol. 20, pp. 404–418, March/April 1990.

[42] M. Barrére, A. Jaumotte, B. Veubeke, and J. Vandenkerckhove, eds., *Rocket Propulsion.* New York, New York: Elsevier Publishing Company, 1960.

[43] G. Mandell, G. Caporaso, and W. Bengen, eds., *Topics in Advanced Model Rocketry.* Cambridge, Massachusetts: The MIT Press, 1973.

[44] J. Slotine and W. Li, *Applied Nonlinear Control.* Englewood Cliffs, New Jersey: Prentice Hall, 1991.