A Self-Organizing Fuzzy Logic Controller for Dynamic Systems Using a Fuzzy Auto-Regressive Moving Average (FARMA) Model

Young-Moon Park, *Senior Member, IEEE*, Un-chul Moon, and Kwang Y. Lee, *Senior Member, IEEE*

***Abstract*—This paper proposes a complete design method for an on-line self-organizing fuzzy logic controller without using any plant model. By mimicking the human learning process, the control algorithm finds control rules of a system for which little knowledge has been known. In a conventional fuzzy logic control, knowledge on the system supplied by an expert is required in developing control rules, however, the proposed new fuzzy logic controller needs no expert in making control rules. Instead, rules are generated using the history of input-output pairs, and new inference and defuzzification methods are developed. The generated rules are stored in the fuzzy rule space and updated on-line by a self-organizing procedure. The validity of the proposed fuzzy logic control method has been demonstrated numerically in controlling an inverted pendulum.**

I. Introduction

TRADITIONALLY, controllers are designed on the basis of a mathematical description and its linearized model. Therefore, it is difficult to implement these model-based controllers to a real system, especially to a system which is complex and nonlinear [1]. As an alternative to these model-based controls, the concept of fuzzy logic was introduced by Zadeh [2], and since its introduction by Mamdani [3], [4], the fuzzy logic control method has been successfully applied to various control problems [5]–[8].

In general, the Mamdani's fuzzy logic control method controls the plant using fuzzy inference with rules preconstructed by an expert. Although there are some nonlinear effects, the fuzzy inference procedure can be roughly interpreted as an interpolation of the rules [9], [10]. Therefore, in Mamdani's method, the most important task is to form the rule base which represents the experience and intuition of human experts. When the rule base of human experts is not available, an efficient control can not be expected.

The self-organizing fuzzy controller (SOFC) is a rule-based type of controller which learns how to control on-line while being applied to a system, and it has been used successfully for a wide variety of processes [11]–[13]. The SOFC combines system identification and control based on experience. The ability of SOFC in carrying out the system identification makes it unnecessary to have a good understanding of the environment; therefore, only a minimal amount of information about the environment needs to be provided. As a step toward self-organization Ramaswamy, *et al.* proposed an automatic tuning method for a fuzzy logic controller and applied it to control a nuclear reactor [14], [15]. In this method, the rules were parameterized as functions of fuzzy input variables and the parameters were tuned off-line through experiments. Recently, Jang proposed a generalized control strategy that enhances the fuzzy controllers with self-learning capability [16], where he implemented fuzzy inference system into a neural network and applied the back-propagation-type gradient descent method to propagate the error signals through different times stages. Karr and Gentry used genetic algorithms for high-performance of a fuzzy control, and successfully applied it to a pH control problem [17].

The main purpose of this paper is to minimize the role of human experts in designing a fuzzy logic controller. In general, human beings learn about an unknown object through experience, which is replaced with a new experience whenever it is regarded better. For this reason, the Fuzzy Auto-Regressive Moving Average (FARMA) controller is proposed, which not only uses output history but also input history in its rules. The FARMA controller has no rule at its initial stage, but forms rules by defining membership functions using the plant input–output data as singletones and stores them in a rule base. The rule base is updated as experience is accumulated using a self-organizing procedure. Another contribution of this paper is the development of new method for the inference and defuzzification. Unlike the conventional inference scheme where the truth value reflects only one input variable of the least similarity, the new inference method incorporates all input variables using Euclidean distance in computing the truth value. A new method for defuzzification is also developed by adding a predictive capability using a trend model.

In Section II, we introduce the FARMA rule. The generation of the FARMA rule using plant input–output pairs is presented in Section III-A. The developments of inference and defuzzification are presented in Sections III-B and C, respectively. The self-organization of the rule base is presented in Section III-D. Afterwards, in Section IV, simulation results are presented. And, finally, the conclusions are drawn in Section V.

Manuscript received August 12, 1993; revised February 9, 1994. This work was supported in part by Korea Science and Engineering Foundation (KOSEF) and the National Science Foundation (NSF) under Grants INT-9223030 and EID-921232 and by NSF and the Electric Power Research Institute (EPRI) under Grant ECS-9216504/RP-8030-4.

Y.-M. Park and U.-C. Moon are with the Department of Electrical Engineering, Seoul National University, Seoul 151-742, Korea.

K. Y. Lee is with the Department of Electrical Engineering, The Pennsylvania State University, University Park, PA 16802 USA.

IEEE Log Number 9406651.

II. The FARMA Fuzzy Logic Controller

A. Formulation of the Problem

In conventional fuzzy logic control (FLC), the dynamic behavior of a system is characterized by a set of linguistic descriptions of rules based on an expert's knowledge. The expert's knowledge is usually of the form

R_1: If x_1 is A_1, and x_2 is B_1, then z is C_1,
R_2: If x_1 is A_2, and x_2 is B_2, then z is C_2,
... ...
... ...
R_n: If x_1 is A_n, and x_2 is B_n, then z is C_n

where x_1, x_2, and z are linguistic variables representing two process variables and one control variable, respectively; A_i, B_i, and C_i are linguistic values or fuzzy sets of the linguistic variables x_1, x_2, and z, respectively. Typically, the linguistic variables of a FLC are the error and the error derivative of the system output. The linguistic values are usually defined as fuzzy sets with appropriate membership functions.

In general, the output of a system can be described with a function or a mapping of the plant input-output history. For a single-input single-output (SISO) discrete-time system, the mapping can be written in the form of a nonlinear function as follows

$$y(k+1) = f(y(k), y(k-1), y(k-2), \cdots, u(k), u(k-1), u(k-2), \cdots) \quad (1)$$

where $y(k)$ and $u(k)$ are, respectively, the output and input variables at the kth time step.

The objective of the control problem is to find a control input sequence which will drive the system to an arbitrary reference set point y_{ref}. Rearranging (1) for control purposes, the value of the input u at the k-th step that is required to yield the reference output y_{ref} can be written as follows

$$u(k) = g(y_{\text{ref}}, y(k), y(k-1), \cdots, u(k-1), u(k-2), \cdots) \quad (2)$$

which is viewed as an inverse mapping of (1).

B. Definition of the FARMA Rule

It should be noted that the control input in (2) depends on both input and output history. Therefore, it is proposed to include the input terms, $u(i)$, in a fuzzy rule, since the output history, $y(i)$, alone cannot describe the system. From this observation, a rule is made from both input and output history; while a typical conventional fuzzy rule uses the "error" and the "change of error" of the output history alone.

The proposed controller does not use rules preconstructed by experts, but forms rules with input and output history at every sampling step. The rules generated at every sampling step are stored in a rule base and updated as experience is accumulated using a self-organizing procedure.

System (1) yields the last output value $y(k+1)$ when the output and input values, $y(k), y(k-1), y(k-2), \cdots, u(k), u(k-1), u(k-2), \cdots$, are given. This implies that $u(k)$ is the input to be applied when the desired output is y_{ref} as indicated explicitly in (2). Therefore, a new rule with the input and output history can be defined as follows

IF y_{ref} is $A_{1i}, y(k)$ is $A_{2i}, y(k-1)$ is $A_{3i}, \cdots,$
$y(k-n+1)$ is $A_{(n+1)i}$,
AND $u(k-1)$ is $B_{1i}, u(k-2)$ is $B_{2i}, \cdots, u(k-m)$ is B_{mi},
THEN $u(k)$ is C_i, (for the ith rule) (3)

where

n, m: number of output and input variables
A_{ij}, B_{ij}: antecedent linguistic values for the i-th rule
C_i: consequent linguistic values for the i-th rule.

The rule (3) resembles the auto-regressive moving average model (ARMA) in time-series; and hence in this paper, it will be called the FARMA rule.

As an input to the FLC, crisp plant conditions are changed to fuzzy sets, i.e., input fuzzy sets, corresponding to the antecedent part of the FARMA rule. The value y_{ref} and the given plant condition, $y(k), y(k-1), \cdots, y(k-n+1), u(k-1), u(k-2), \cdots, u(k-m)$, at the kth step are used to form input fuzzy sets of the FLC. The fuzzification for input fuzzy sets is done by fuzzy singletons using the crisp values, y_{ref}, $y(k), y(k-1), \cdots, y(k-n+1), u(k-1), u(k-2), \cdots$, and $u(k-m)$, in the next section.

III. Design of the FARMA Fuzzy Logic Controller

A. Generation of the Rule Base

In a conventional FLC, an expert usually determines the linguistic values A_{ij}, B_{ij}, and C_i by partitioning each universe of discourse, and the formulation of fuzzy logic control rules is achieved on the basis of the expert's experience and knowledge. In this paper, however, these linguistic values are determined from the crisp values of the input and output history at every sampling step. Therefore, at the initial stage, the assigned $u(k)$ may not be good control, but over time, the rule base is updated using the proposed self-organizing procedure, and better controls are applied.

A fuzzification procedure for fuzzy values is developed to determine $A_{1i}, A_{2i}, \cdots, A_{(n+1)i}, B_{1i}, B_{2i}, \cdots, B_{mi}$, and C_i from the crisp $y(k+1), y(k), y(k-1), \cdots, y(k-n+1)$, $u(k-1), u(k-2), \cdots, u(k-m)$, and $u(k)$, respectively. The fuzzification is done with its base on a reasonably assumed input or output ranges. When the assumed input or output range is [a, b], the membership function for crisp x_1 is determined in a triangular shape as follows

$$\mu_{A_1} = \begin{cases} 1 + (x - x_1)/(b-a) & \text{if } a \le x < x_1, \\ 1 - (x - x_1)/(b-a) & \text{if } x_1 \le x < b. \\ 0 & \text{else} \end{cases} \quad (4)$$

If the crisp value is "a" (lower bound), then the membership function is a straight line, with the membership degree one at the lower bound and zero at the upper bound. The slope of the line is "$-1/(b-a)$". Similarly, for the case of the crisp value "b" (upper bound), the membership function is also a straight line, with the membership degree zero at the lower bound and one at the upper bound. The slope of the line is "$1/(b-a)$." For a crisp value in the interior of the range $[a, b]$, the fuzzification is done in a triangular shape, which has the

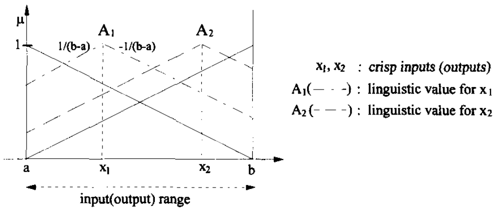

Fig. 1. The fuzzification procedure for A_{ij}, B_{ij}, or C_i.

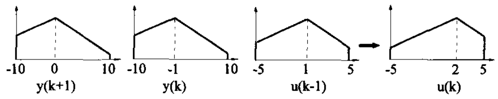

Fig. 2. The generation of a FARMA rule.

slope "$-1/(b-a)$" and "$1/(b-a)$" for the right and left lines, respectively.

Fig. 1 shows the fuzzification procedure for crisp variables x_1 and x_2, where A_1 and A_2 are the corresponding linguistic values (fuzzy sets) with membership functions defined on the range $[a, b]$.

Note that all linguistic values overlap on the entire range $[a, b]$, and furthermore, every crisp value uniquely defines the membership function with the unity vertex value and identical slopes. Thus, this fuzzification procedure requires only the minimal information in forming the membership functions, i.e., the crisp value; and moreover, each rule can be uniquely represented as a point in the $(n+m+1)$-dimensional rule space. For practical purposes, the rule space is partitioned into a finite number of domains and only one rule, i.e., a point, is stored in each domain (see Section III-D).

The above fuzzification procedure is used to form a FARMA rule. For example, let output range be $[-10, 10]$, and input range be $[-5, 5]$, and the input output history be $y(k+1) = 0, y(k) = -1$, and $u(k-1) = 1$ then, a FARMA rule generated at the kth step is shown in Fig. 2.

A FARMA rule is generated at each sampling step and stored in a rule base. This means that every experience is regarded initially as a fuzzy logic control rule. As the run continues, the experience will be accumulated and the FARMA rule is updated for each domain in the rule space. The updating procedure will be explained later in Section III-D.

B. Inference With Similarity Function

To attain the output fuzzy set in a conventional method, it is necessary to determine a "truth value" of the input fuzzy set with respect to each rule [19], [20], [28], [29]. If input fuzzy variables are considered as fuzzy singletons, the truth value of the input fuzzy variables for each rule may be calculated by using the *min-and* operation as follows

$$\omega_i = \min[(A_{1i} \wedge X_1), (A_{2i} \wedge X_2), \cdots, (A_{(n+1)i} \wedge X_{(n+1)}), \cdots] \quad (5)$$

where

ω_i: truth value for the ith rule
X_i: input fuzzy variable (singleton)
$\wedge$: AND operation.

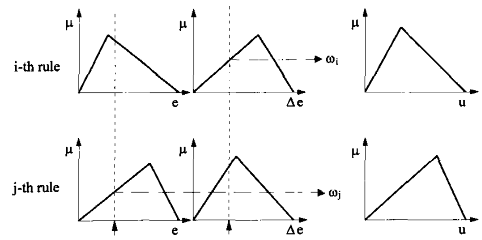

Fig. 3. The truth values in a conventional FLC method.

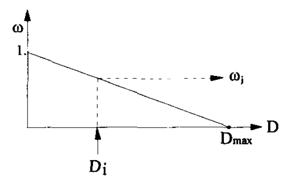

Fig. 4. The truth value with the similarity function.

The truth value for the ith rule can be considered as "the degree of similarity" between the input fuzzy variables and the antecedent linguistic values of the ith rule.

This conventional method has a shortcoming in that it considers only the minimum intersection degree between input fuzzy variables and the antecedent linguistic values as shown in Fig. 3. This weakness may become severe as the number of input variables increases. Since the minimum value alone is considered in this case, many other relevant input variables will be ignored.

From the interpolation point of view, this paper proposes a new method to deduce the truth value by compounding all the input variables with the l_2-norm or the Euclidean distance. First, the Euclidean distance between the newly measured crisp input variables and the values corresponding to the vertices of the membership functions of the existing linguistic values is defined as follows

$$D_i = \sqrt{(x_{1i}-x_1)^2+(x_{2i}-x_2)^2+\cdots+(x_{(n+m+1)i}-x_{(n+m+1)})^2} \quad \text{(for the } i\text{th rule)} \quad (6)$$

where

$x_1, x_2, \cdots,$: crisp input variables
$x_{1i}, x_{2i}, \cdots$: vertices of the membership functions for $A_{1i}, A_{2i}, \cdots, B_{1i}, B_{2i}, \cdots$.

Next, using Euclidean distance, a similarity function is introduced as shown in Fig. 4. The similarity value, ω, plays the role of the truth value. For example, if the crisp input variables perfectly match the vertices of the corresponding membership functions, the Euclidean distance D_i is zero. Hence ω_i is one, which means the input condition perfectly matches the antecedent linguistic variables of the ith rule. The point $D_{\max}$ in Fig. 4 explains the distance for which the truth value becomes zero. Although a tuning procedure may be necessary to determine $D_{\max}$, it is determined by trial and error in this paper.

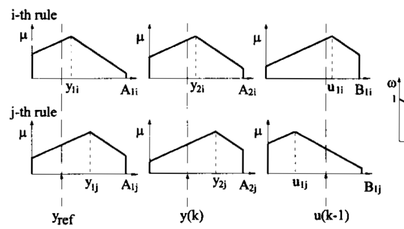

Fig. 5. Inference with the similarity function.

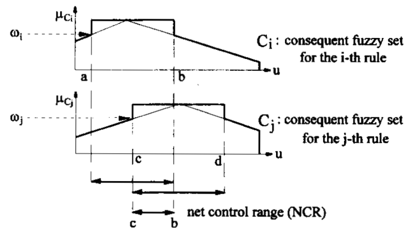

Fig. 6. The net control range (NCR) by the φ-opertion with two rules.

The similarity value thus defined reflects the contribution of all input variables. The evaluation of the proposed truth value ω with three fuzzy input variables, $y_{\text{ref}}, y(k)$, and $u(k-1)$, is shown in Fig. 5, where the i-th rule is closer to the input variables than the jth rule and thus $\omega_i > \omega_j$.

The consequent linguistic value, that is the net linguistic control action, C_n, is deduced with the φ-operation [28] as follows

$$C_n = \bigcap_i (\omega_i \varphi \mu_{C_i}), \tag{7}$$

$$\omega_i \varphi \mu_{C_i} = \begin{cases} 1 & \text{if } \omega_i \leq \mu_{C_i} \\ \mu_{C_i} & \text{if } \omega_i > \mu_{C_i} \end{cases} \tag{8}$$

where

C_n: net linguistic control action

ω_i: truth value from the similarity function for the i-th rule

μ_{C_i}: membership degree of the consequence linguistic value C_i in the ith rule.

With the C_n, we take the α-cut of the C_n where $\alpha = \max \mu(C_n)$, to find a control range for the highest possibility. The use of the φ-operation and the inference procedure with two rules are shown in Fig. 6. Each operation forms the consequent fuzzy set, and the range with its membership degree one is deduced as a control range for each rule, i.e., $[a, b]$ for the ith rule, and $[c, d]$ for the jth rule as the respective ranges with the highest possibility. As a result of this inference, the net control range (NCR), which is the "intersection" of all control ranges, is determined, i.e., $[c, b]$ in Fig. 6.

For a large $D_{\max}$ the NCR may be empty since the large $D_{\max}$ value in the similarity function causes a high truth value for each rule. In this case, however, the α-cut with $\alpha < 1$ always guarantees an NCR with a singleton.

C. Defuzzification with Extrapolation

Defuzzification is a procedure to determine a crisp value from a consequent fuzzy set. That is, the defuzzification selects a representative crisp output of the FLC from the possibility distribution over the output space. The often used methods are the center of area (COA) and the mean of maxima (MOM) [19]–[21]. In this paper, defuzzification is to determine a crisp value from the net control range (NCR) resulting from the inference. Any control value within the NCR has a potential as a control, however, some controls may cause overshoot while others may be too slow.

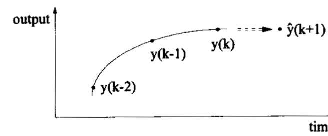

Fig. 7. Estimation of $y(k+1)$ by the second extrapolation.

This problem can be avoided by adding a predictive capability in the defuzzification. A method is presented which modifies the NCR to compute a crisp value by using the prediction or "trend" of the output response. The series of the last outputs is extrapolated in time domain to estimate $y(k+1)$ by the Newton backward-difference formula [31]. If the extrapolation order is l, using the binomial-coefficient notation

$$\binom{s}{k} = \frac{s(s-1)\cdots(s-k+1)}{k!} \tag{9}$$

the estimate $\hat{y}(k+1)$ is calculated as follows

$$\hat{y}(k+1) = \sum_{i=0}^{l} (-1)^i \binom{-1}{i} \nabla^i y(k) \tag{10}$$

where

$$\nabla^i y(k) \triangleq \nabla(\nabla^{i-1} y(k)) \text{ for } i \geq 2$$
$$\nabla y(k) \triangleq y(k) - y(k-1).$$

Fig. 7 shows an example of the second order extrapolation from the last three points, $y(k-2), y(k-1)$, and $y(k)$, to attain the estimate of $y(k+1)$, denoted by $\hat{y}(k+1)$. The estimation using extrapolation is based on the hypothesis that the output will not have any sudden change, but will follow the latest output trend.

Defuzzification is performed by comparing the two values, the estimate $\hat{y}(k+1)$ and the reference output y_{ref} or the temporary target $y_r(k+1)$, generated by

$$y_r(k+1) = y(k) + \alpha(y_{\text{ref}} - y(k)) \tag{11}$$

where $y_r(k+1)$ is the reference output or the temporary target and α is the target ratio constant $(0 < \alpha \leq 1)$. The value of α describes the rate with which the present output $y(k)$ approaches the reference output value, and thus has a positive value between zero and one. The value of α is chosen by the user to obtain a desirable response. The selection of α has similar effect as the selection of a reference model in the model reference adaptive control (MRAC).

When the estimate exceeds the reference output, the control has to slow down. On the other hand, when the estimate has not reached the reference, the control should speed up. Two possible cases will therefore be considered

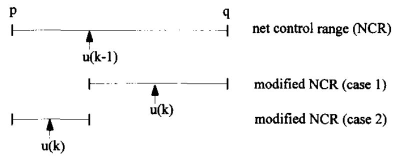

Fig. 8. The modification of the net control range and defuzzification.

Case 1: $\hat{y}(k+1) < y_r(k+1)$

Case 2: $\hat{y}(k+1) > y_r(k+1)$.

To modify the control range, the sign of $\nabla u(k)(= u(k) - u(k-1))$ is assumed to be the same as the sign of $(y_r(k+1) - \hat{y}(k+1))$. This is based on the ARMA model representation of a plant

$$y(k+1) = a_1 y(k) + a_2 y(k-1) + a_3 y(k-2) + \cdots + b_1 u(k) + b_2 u(k-1) + b_3 u(k-3) + \cdots \tag{12}$$

where the sign of "b_1" determines whether $u(k)$ has to be increased or decreased in controlling the value $y(k+1)$.

The sign of b_1 can be easily determined in a physical system. For example, in a boiler system, if u is fuel and y is temperature then the output y increases (decreases) as u increases (decreases). Thus, the sign of b_1 is positive. Therefore, without the loss of generality, the sign of b_1 is assumed to be positive in this paper; implying that the input has to be increased to increase the output.

Thus, for Case 1 the sign of $(y_r(k+1) - \hat{y}(k+1))$, hence the sign of $\nabla u(k)$, is positive, implying that $u(k)$ has to be increased from the previous input $u(k-1)$. On the other hand, for Case 2, the sign of $\nabla u(k)$ is negative and $u(k)$ has to be decreased from the previous input $u(k-1)$. The final crisp control value $u(k)$ is then selected as one of the mid-points of the modified net control ranges as shown in Fig. 8

$$u(k) = \begin{cases} (u(k-1) + q)/2 & \text{for Case 1,} \\ (p + u(k-1))/2 & \text{for Case 2} \end{cases} \tag{13}$$

where p and q are the respective lower and upper limits of the net control range (NCR) resulting from the inference (Section III-B).

When the sign of b_1 is negative, $u(k)$ is $(p + u(k-1))/2$ for Case 1, and $(u(k-1) + q)/2$ for Case 2, respectively, in (13); and in Fig. 8, the two cases have to be switched.

D. Self-Organization of the Rule Base

The FARMA rule defined in Section II-B is generated at every sampling time. Each rule can be represented as a point in the $(n + m + 1)$-dimensional rule space, i.e., $(x_{1i}, x_{2i}, \cdots, x_{(n+m+1)i})$ as in (6). If every rule is stored in the rule base, two problems will occur. First, the memory will be exhausted. And second, the rules which are formed improperly during initial stages also affect the later inference.

For this reason, the fuzzy rule space is partitioned into a finite number of domains of different sizes and only one rule, i.e., a point, is stored in each domain. The variables around the set point (target) are partitioned densely to reduce the steady-state error. This uneven partition, however, is not done for input variables, $u(k-1), u(k-2), \cdots$, since the steady state value of u is not known. Fig. 9 shows an example of the division of a rule space for one output and one input variables, $y(k)$ and $u(k-1)$.

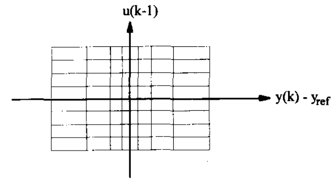

Fig. 9. Division of a 2-dimensional rule space.

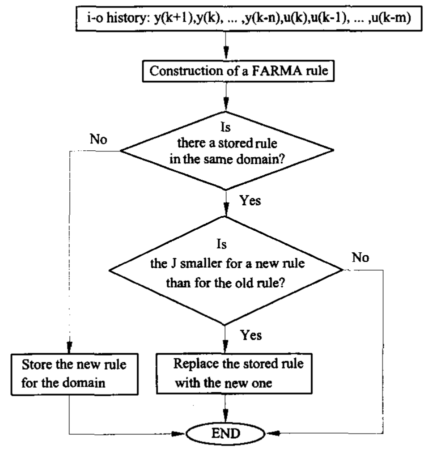

Fig. 10. The self-organization of the rule base.

To overcome the second problem, the following performance index is defined in updating the rule base

$$J = |y_r(k+1) - y(k+1)| \tag{14}$$

where $y(k+1)$ is real plant output, and $y_r(k+1)$ is the reference output. Therefore, at the $(k+1)$-th step, the performance index J is calculated with the real plant output $y(k+1)$ resulting from the kth step control. Fig. 10 shows the rule base updating procedure. If there are two rules in a given domain, the selection of a rule is based on J. That is, if there is a new rule which has the output closer to the reference output in a given domain, the old rule is replaced by the new one. This updating procedure of the rule base makes the proposed fuzzy logic controller capable of learning the object plant and self-organizing the rule base. The number of rules increases monotonically as new input-output data is experienced. It converges to a finite number in steady state, however, and never exceeds the maximum number of domains partitioned in the $(n + m + 1)$-dimensional rule space.

This approach compares well with the Cell State Space Algorithm [18], in that the proposed method makes a reference trajectory or reference cell in the state space by (14) and remembers the favorable results as the FARMA rules.

Fig. 11 shows the architecture of the proposed FLC system. Initially, since there is no control rule in the rule base, the control input $u(k)$ for the first step is the medium value of

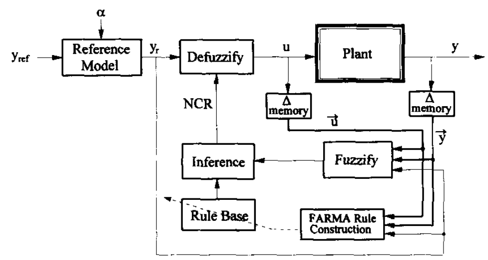

Fig. 11. The self-organizing fuzzy logic control system architecture.

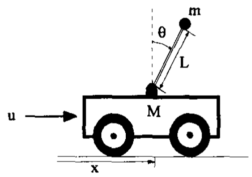

Fig. 12. An inverted pendulum of cart.

the entire input range. As time increases, the defuzzification procedure begins to determine whether the input has to be increased or decreased depending on the trend of the output. The sign of $\nabla u(k)$ and the magnitude of $u(k)$ are determined in the defuzzification procedure. The self-organization of the rule base, in other words "learning" of the object system, is performed at each sampling time.

IV. Simulation Result

A case study considered is the problem of balancing an upright pole, i.e., an inverted pendulum. The bottom of a pole is attained by a pivot to a cart that travels along a track as shown in Fig. 12. The movement of the cart and pole is constrained in the vertical plane.

The model of an inverted pendulum is described by the following differential equations

$$(M+m)\ddot{x} + mL\cos(\theta)\ddot{\theta} - mL\cos(\theta)\dot{\theta}^2 = u \qquad (15)$$
$$m\ddot{x}\cos\theta + mL\ddot{\theta} = mg\sin\theta$$

where

M: mass of cart
m: mass of pole
L: distance from pivot to pole's center of mass
g: acceleration of gravity ($g = 9.8$ m/s^2).

This model is a fourth-order nonlinear system with input force u in Newton (N) and output angle $y = \theta$ in degree (°). The system is inherently unstable, and has severe nonlinearity when the angle deviates much from zero. Thus, it is very difficult to solve this control problem by traditional analytical methods.

The control force u is limited to 40[N] as the maximum available force. The parameters of the system are $M = 1$[kg], $m = 1$[kg], and $L = 1$[m], and the sampling time is 0.01[sec]. The fourth-order Runge–Kutta method is applied to solve the given differential equations. In the simulation, $y_{\text{ref}}, y(k), y(k-1)$, and $u(k-1), u(k-2)$ were used as input variables to the fuzzy logic controller and each variable was divided into six segments to partition the rule space. The third-order extrapolation was performed to estimate $y(k+1)$. Output range is $-70° \sim 70°$, input range is $-40 \sim 40$[N], and the target ratio constant α was determined by trial and error. In Fig. 12, it is noted that the input u has to be increased (decreased) to decrease (increase) the output angle. Therefore, the sign of b_1 in the ARMA model (12) is negative.

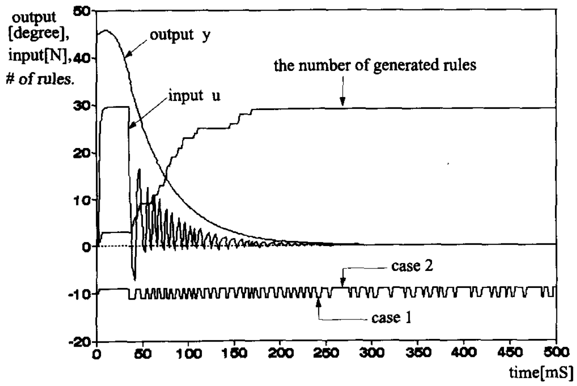

Fig. 13. Simulation result 1: initial angle 45 degrees, $\alpha = 0.15$.

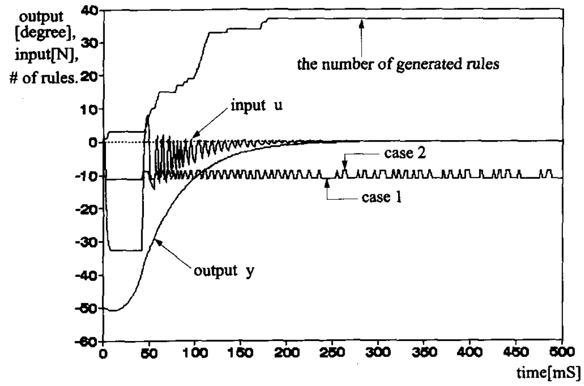

Fig. 14. Simulation result 2: initial angle -50 degrees, $\alpha = 0.15$.

Fig. 13 shows the result of a simulation with the proposed fuzzy logic controller when the pole is with an initial angle of $y = 45$ degrees and, the reference is zero (degree). The used target ratio constant α is 0.15 and the number of generated rules is 29 after convergence. Case 1 and Case 2 for the defuzzification procedure are shown in Fig. 13 using a function, sign $(\nabla u(k)) - 10$. The function has the value -11 for Case 1 and -9 for Case 2, respectively. If the plant condition from the output trend is classified as Case 1 (Case 2), the input $u(k)$ is decreased (increased) from $u(k-1)$.

In Fig. 14, the initial angle is $y = -50$ degrees, the reference is zero, and the α is 0.15. The number of generated rules is 37. Case 1 and Case 2 for defuzzification are also shown. From Fig. 13 and Fig. 14, it can be observed that the number of rules generated increases monotonically as runs continue and more rules are stored for larger initial condition.

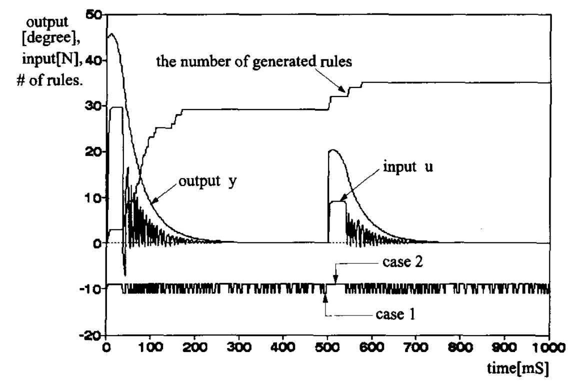

Fig. 15. Simulation result 3: initial angle 20 degrees, $\alpha = 0.15$.

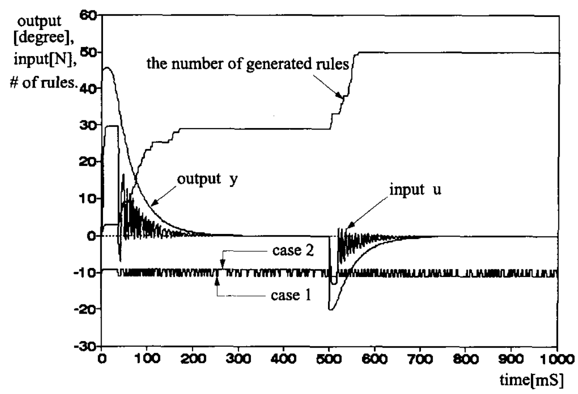

Fig. 16. Simulation result 4: disturbance −20 degrees, $\alpha = 0.15$.

In Fig. 15 and Fig. 16, an external disturbance is suddenly applied at 500 step (5 sec.). The magnitudes of the disturbance are 20 degrees and −20 degrees, respectively, and the number of generated rules is increased to 35 and 50, respectively. The number of newly-generated rules is larger in simulation 4 than in simulation 3. This is because the sign of the disturbance (−20 degrees) and the initial condition (45 degrees) are different in simulation 4, and thus more new plant conditions are experienced by the FLC.

Fig. 17 shows the influence of different α. The smaller the α was, the slower the convergence was. The output diverged when α was larger than 0.2, which is similar to the case when system diverges with improperly chosen reference model in the MRAC.

V. CONCLUSION

This paper proposed a fuzzy logic control algorithm which controls an unknown plant without a modeling procedure or preconstructed rules of an expert. It mimics the human learning process with only a minimal information on the environment. That is, the proposed algorithm starts with no initial rule, but controls the plant on-line by trial and error, and by remembering results that are considered favorable. To realize this, a concept of the FARMA rule was introduced using the plant input–output history, and new inference and

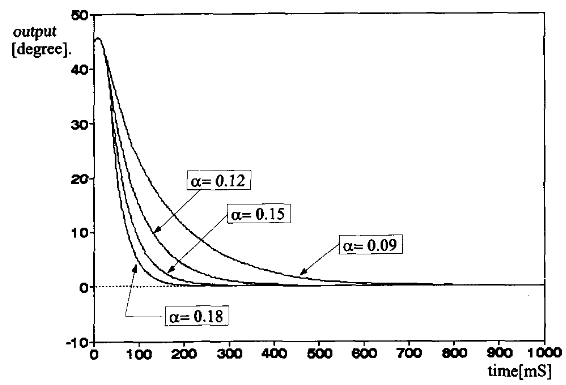

Fig. 17. The influence of the target ratio constant α.

defuzzification methods were developed. The truth value for inference is defined by using the Euclidean distance and a similarity function to better represents the closeness of the input variables to the antecedent linguistic variables. In the defuzzification procedure, the sign of $\nabla u(k)$ is determined with the hypothesis that the system output will not have any sudden change, but will follow the latest trend. An updating procedure of the rule base is developed, which makes the proposed fuzzy logic controller capable of learning the system and self-organizing the controller.

Computer simulations for an inverted pendulum have demonstrated the effectiveness of the proposed fuzzy logic controller, and showed satisfactory results without modeling or preconstructed rules of human experts. The FARMA rule and the inference using the similarity function developed here can also be applied to other conventional fuzzy logic control.

REFERENCES

[1] B. D. O. Anderson, "Adaptive system, lack of persistency of excitation and bursting phenomena," *Automatica*, vol. 21, pp. 247–258, 1985.
[2] L. A. Zadeh, "Fuzzy sets," *Inform. Contr.*, vol. 8, pp. 338–353, 1965.
[3] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *Int. J. Man Mach. Studies*, vol. 7, no. 1, pp. 1–13, 1975.
[4] E. H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *IEEE Trans. Computer*, vol. C-26, no. 12, pp. 1182–1191, 1977.
[5] J. A. Bernard, "Use of rule base system for process control," *IEEE Contr. Syst. Mag.*, vol. 8, no. 5, pp. 3–13, 1988.
[6] S. Yasunobu and S. Miyamoto, "Automatic train operation by predictive fuzzy control," *Industrial Applications of Fuzzy Control.* New York: North-Holland, 1985, pp. 1–18.
[7] T. Tagkagi and S. Sugeno, "Fuzzy identification of system and its applications to modeling and control," *IEEE Trans. Man, Cybern.*, vol. SMC-15, no. 1, pp. 116–132, 1985.
[8] O. Yagishita, O. Itoh, and M. Sugeno, "An application of fuzzy reasoning to water purification process," *Industrial Applications of Fuzzy Control.* New York: North-Holland, 1985, pp. 19–40.
[9] K. L. Tang and R. J. Mulholland, "Comparing fuzzy logic with classical controller designs," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-17, no. 1, pp. 1085–1088, 1985.
[10] H. Ying, W. Siler, and J. J. Buckley, "Fuzzy control theory: A nonlinear case," *Automatica*, vol. 26, no. 3, pp. 513–520, 1990.
[11] T. J. Procyk and E. H. Mamdani, "A linguistic self-organizing process controller," *Automatica*, vol. 15, pp. 15–30, 1979.
[12] R. Tanscheit and E. M. Scharf, "Experiments with the use of a rule-based self organizing controller for robotics applications," *Fuzzy Set and System*, vol. 26, pp. 195–214, 1988.

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

[14] P. Ramaswamy, R. M. Edwards, and K. Y. Lee, "An automatic tuning method of a fuzzy logic controller for nuclear reactors," *IEEE Trans. Nuclear Sci.*, vol. 40, no. 4, pp. 1253–1262, Aug. 1993.

[15] ——, "Wide range nuclear reactor temperature control using automatically tuned fuzzy logic controller," *Trans. Amer. Nucl. Soc.*, vol. 66, pp. 101–103, Nov. 1992.

[16] J. S. Jang, "Self-learning fuzzy controllers based on temporal back propagation," *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 714–723, 1992.

[17] C. L. Karr and E. J. Gentry, "Fuzzy control of pH using genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 1, pp. 46–53, 1993.

[18] Y. Y. Chen and T. C. Tsao, "A description of the dynamical behavior of fuzzy system," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-19, no. 4, pp. 745–755, 1989.

[19] C. C. Lee, "Fuzzy logic in control system: fuzzy logic controller—Part I," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-20, no. 2, pp. 404–418, 1990.

[20] ——, "Fuzzy logic in control system: fuzzy logic controller—Part II," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-20, no. 2, pp. 419-435, 1990.

[21] M. Sugeno, "An introductory survey of fuzzy control," *Inform. Sci.*, vol. 36, pp. 59–83, 1985.

[22] G. M. Abdelour, C. H. Chang, F. H. Huang, and J. Y. Cheung, "Design of fuzzy controller using input and output mapping factors," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC.-21, no. 5, pp. 952–960, 1991.

[23] T. R. Anderson and S. B. Nielson, "An efficient single output fuzzy control algorithm for adaptive application," *Automatica*, vol. 21, no. 5, pp. 539–545, 1983.

[24] Z. Y. Zhao, M. Tomizuka, and S. Sagara, "A fuzzy tuner for fuzzy logic controller," *Proc. Am. Contr. Conf.*, pp. 2268–2272, 1992.

[25] L. J. Huang and M. Tomizuka, "A self-pased fuzzy tracking controller for two-dimensional motion control," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-20, no. 5, pp. 1115–1124, 1990.

[26] D. Benlahcen and M. Lamotte, "A fuzzy automation synthesis method," *Automatica*, vol. 17, no. 2, pp. 297-306, 1981.

[27] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive System.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[28] W. Pedrycz, *Fuzzy Control and Fuzzy Systems.* New York: Wiley, 1989.

[29] H. J. Zimmermann, *Fuzzy Set Theory and Its Application.* Norwell, MA: Kluwer, 1986.

[30] B. Kosco, *Neural Network and Fuzzy Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1992.

[31] R. L. Burden and J. D. Faires, *Numerical Analysis.* PWS-KENT, 1989.

Young-Moon Park (M'84–SM'91) received the B.S., the M.S., and the Ph.D. degrees from Seoul National University, Seoul, Korea, in 1956, 1959, and 1971, respectively.

He has been with Seoul National University since 1959, where he is currently a Professor of Electrical Engineering. He was a Visiting Professor at Tokyo University in 1981, University of Houston in 1983, and the Pennsylvania State University in 1987. His current research interests include power system control, operation and planning, and application of fuzzy logic control, artificial neural networks, and expert systems to power systems.

Dr. Park is a member of Korea Institute of Electrical Engineering, where he was the past President from 1989 to 1990. He was the chairman of the Organizing Committee and a co-chairman of the International Federation of Automatic Control Symposium on Control of Power Plants and Power Systems, which was held in Seoul in 1989. He is the current Director of Korea Electrical Engineering and Science Research Institute.

Un-Chul Moon received the B.S. and the M.S. degrees in electrical engineering from Seoul National University, Seoul, Korea, in 1991 and 1993, respectively. He is currently a Ph.D. candidate in Power Systems Laboratory, Electrical Engineering, Seoul National University.

His current research interests include fuzzy logic systems, artificial neural networks, and their applications to control, operation, and planning of power systems.

Kwang Y. Lee (S'70–M'72–SM'86) received the B.S. degree from Seoul National University, Seoul, Korea, in 1964, the M.S. degree from North Dakota State University, Fargo, ND, in 1967 and the Ph.D. degree in systems science from Michigan State University, East Lansing, MI, in 1971.

He has been on the faculties of Michigan State, Oregon State, University of Houston, and Pennsylvania State University, University Park, PA, where he is a Professor of Electrical Engineering. He was a Visiting Professor at Seoul National University in 1993, where he conducted a US-Korea Cooperative Research on Intelligent Distributed Control of Power Plants and Power Systems. His current research interests include control theory, artificial neural networks, fuzzy logic systems, and computational intelligence and their applications to power systems and robotics.

Dr. Lee is currently the Director of Power Systems Control Laboratory and co-director of Intelligent Distributed Controls Research Laboratory at Penn State.