Fuzzy Logic Control With Predictive Neural Network

Sung Hoon Jung
School of Information and Computer Engineering, Hansung Univ.
389 SamsunDong-2-Ga, SungBook-Gu, Seoul, 136-792, Korea
Tel: 02-760-4344
Email: shjung@ice.hansung.ac.kr

Abstract

Fuzzy logic controllers have been shown better performance than conventional ones especially in highly nonlinear plants. These results are caused by the nonlinear inference capability of the controllers with nonlinear fuzzy rules. In some applications, however, the fuzzy rules were not sufficient to cope with significant uncertainty of the plants and environment. Moreover, it is hard to make fuzzy rules consistent and complete. In this paper, we employed a predictive neural network to enhance the nonlinear inference capability. The predictive neural network generates predictive outputs of a controlled plant using the current and past outputs and current inputs. These predictive outputs are used in terms of fuzzy rules in fuzzy inferencing. From experiments, we found that the predictive term of fuzzy rules enhanced the inference capability of the controller. This predictive neural network can also help the controller cope with uncertainty of plants or environment by on-line learning.

1 Introduction

Fuzzy logic control (FLC) methods have been widely employed to control highly nonlinear plants from the late 1980s [1, 2, 3]. This is because the performance of FLC methods is superior to that of conventional control methods especially in highly nonlinear plants. Although FLC shows good performance, some problems still remain. One of the problems is that the static fuzzy rules are insufficient to cope with significant uncertainty of the plants and environment. Moreover, it is very hard to make fuzzy rules without inconsistency and incompletion. To solve these problems, some approaches—self-organizing fuzzy control, neural-network-based fuzzy logic control, and so on—were introduced [4, 5, 6, 7, 8]. Most self-organizing fuzzy control methods automatically modify their fuzzy rules using adaptation machines. With a performance metric—often an expert system or simply an algorithm—, the adaptation machines change the fuzzy rules for adapting uncertain environment. In neural-network-based fuzzy logic control, fuzzy rules are represented by the weights of connections and artificial neurons of a neural network. This makes it possible for the fuzzy logic controller to automatically acquire and modify fuzzy rules.

This paper introduces a fuzzy logic control scheme with a predictive neural network. In this scheme, a neural network is employed to predict the outputs of a controlled plant. The predictive outputs of a plant are used in terms of fuzzy rules. Thus, the fuzzy rules consist of three terms—errors, change errors, and predictive errors. If the predictive neural network is well trained, the term of predictive errors in fuzzy rule set will greatly enhance the performance of a fuzzy logic controller. Moreover, on-line learning of the neural network enables the fuzzy logic controller to have adaptability for the uncertainty and the change of environment. One of the main advantages of our control scheme is that our control scheme can be easily incorporated with an existing fuzzy logic controller without large modification.

We measured the performance of our control scheme with a DC servo-motor system. Experimental results show that the predictive term of fuzzy rules enhanced the performance of a fuzzy logic controller.

2 Proposed Fuzzy Logic Control Scheme

Figure 1 shows proposed fuzzy logic control scheme. As you can see in figure 1, the proposed scheme is

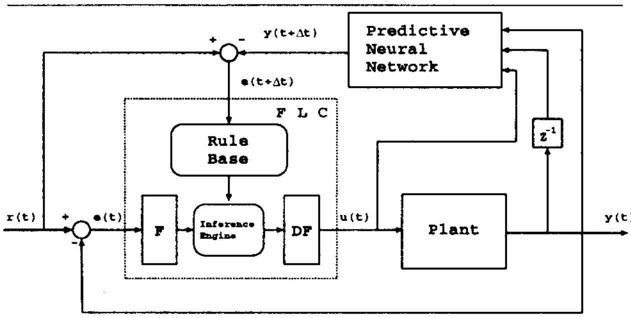

Figure 1: Control Structure of Our Control System

composed of three modules—a fuzzy logic controller, a controlled plant, and a predictive neural network. Excepting the predictive neural network, it is a normal fuzzy logic control structure. We add a neural network to the normal structure for prediction outputs of the plant with plant inputs $u(t)$, plant outputs $y(t)$, and delayed plant outputs $y(t-\Delta t)$. With these three inputs, the neural network estimates the predictive outputs of the plant. These predictive outputs $y(t+\Delta t)$, in other words outputs of neural network, are employed to measure predictive error terms with reference inputs $r(t)$. The predictive error terms $e(t+\Delta t)$ are used in terms of fuzzy rules. If the predictive neural network is sufficiently trained in operating ranges of the plant, thus, the predictive outputs are nearly equal to real outputs in next time, then the predictive error terms will greatly contribute to enhancing the performance of the fuzzy logic controller. These predictive terms of fuzzy rules will considerably reduce the effect caused by the change of environment and parameters of controller itself without human intervention. In other words, adaptability of the controller will be greatly enhanced by on-line learning.

On the other hand, if the neural network is insufficiently trained, the performance of the controller may be degraded. This phenomenon can be diminished by adopting a degree of training of the neural network in the process of inference. Assume that we use Mamdani's inference method. Consider a fuzzy rule : *If e(t) is PB, ce(t) is PS, and pe(t) is NS then u(t) is NS* where *e(t), ce(t), pe(t), u(t)* are an error, a change of error, a predictive error, and an output at time t, respectively; where *PB, PS, and NS* are positive big, positive small, and negative small, respectively; where the predictive error is $pe(t) = r(t) - y(t+\Delta t)$. In the process of fuzzy logic control, the relation $R()$ is given as:

$$R(e(t), ce(t), pe(t), u(t)) = max_{1 \leq i \leq N} min \{E_i(e(t)), CE_i(ce(t)), PE_i(pe(t)), U_i(u(t))\} \quad (1)$$

where $R()$ is a fuzzy relation of each element, N is the number of fuzzy rules, and $E_i(), CE_i(), PE_i(), U_i()$ are membership functions of $i'th$ fuzzy rule, respectively.

To solve the above phenomenon, we first define the degree of training and next a degree of rule effect.

Definition 1: Degree of training
Let $TSSE(t)$ be a total sum square error at time t. Then the degree of training $dg\_t(t)$ at time t is defined as:

$$dg\_t(t) = \frac{1}{1 + TSSE(t) \times \beta} \quad (2)$$

where β is a positive real constant value that controller designers select.

The β factor indicates the degree of effect of the predictive error term as explained later.

Definition 2: Degree of rule effect
The degree of rule effect $dg\_e(t)$ is given as:

$$dg\_e(t) = 1 - dg\_t(t) = 1 - \frac{1}{1 + TSSE(t) \times \beta} \quad (3)$$

With the definitions we can modify the predictive term $PE_i(pe(t))$ as follows.

$$PE_i^*(pe(t)) = max\{PE_i(pe(t)), dg\_e(t)\} \quad (4)$$

Finally, the equation 1 is modified as:

$$R(e(t), ce(t), pe(t), u(t)) = max_{1\leq i\leq N} min \{E_i(e(t)), CE_i(ce(t)), PE_i^*(pe(t)), U_i(u(t))\} \quad (5)$$

If the predictive neural network is insufficiently trained, then $TSSE(t)$ has large value. This makes the degree of training get near to zero and the degree of rule effect get close to one. Let the $TSSE(t)$,for simple explanation, be positive infinite, then the $dg\_t(t)$ is equal to zero and the $dg\_e(t)$ one. Finally, the value of new predictive term $PE_i^*(pe(t))$ becomes one and the $PE_i^*(pe(t))$ term will be canceled. This result indicates that the effect of predictive term is vanished. If the neural network is perfectly trained, then $TSSE(t)$ and $dg\_e(t)$ becomes zero, consequently, the $PE_i^*(pe(t))$ becomes $PE_i(pe(t))$. The higher the degree of training, the larger the effect of predictive term. If a controlled plant is highly nonlinear, then a controller designer would better select the large value of β. This makes the effect of predictive term reduce. If β is infinite, the produce output of proposed controller is equal to that of the normal fuzzy logic controller. A proper value of β selected by a designer will enhance the performance of the controller. This type of solution is also adequate to on-line learning scheme because the degree of training dynamically affects the outputs of inference.

In the prediction, the Δt is another important factor for improving the performance. It can be selected intuitively. If a plant has small inertia, then the Δt may be small. Otherwise, the Δt should be large. More researches about decision of Δt are necessary.

3 Control System Setup

A DC servo-motor system is employed to experiment our control scheme. The transfer function of the system is as follows.

$$G(s) = \frac{\Theta(s)}{V_a(s)} = \frac{K_m}{s[(R_a + L_a s)(J_m s + B_m) + K_b K_m]} \quad (6)$$

where $v_a(t) = \mathcal{L}^{-1}\{V_a(s)\}$ is the applied motor input voltage and $\theta(t) = \mathcal{L}^{-1}\{\Theta(s)\}$ is the angle of the motor shaft.

Table 1: DC servo-motor spec.

R_a	$3.9\ \Omega$
L_a	$5.27\ mH$
K_b	$0.215\ V \cdot sec$
t_m	$14\ msec$
K_m	$2.2\ kgf \cdot cm/A$
J_m	$0.0017\ kgf \cdot cm \cdot sec^2$
B_m	$0.121\ kgf \cdot cm \cdot \mathrm{sec}$

By substituting each parameter values with those in table 1 and letting $Y(s) = \Theta(s)$, $U(s) = V_a(s)$, it can be rewritten as follows.

$$G(s) = \frac{Y(s)}{U(s)} = \frac{2.2}{s(8.959 \times 10^{-6}s^2 + 7.268 \times 10^{-3}s + 0.9449)} \quad (7)$$

This plant is simulated by 4'th order Runge-Kutta method with $1ms$ time step.

We used a back-propagation neural network with 2-30-1 network structure. The learning rate η is 0.02. For training, 1000 training patterns are gathered before starting real control.

4 Experimental Results

In this paper, we experiment only two cases, i.e., β is infinite and β is zero. First we show the training patterns and the predictive output of the neural network. Next, the two experimental results follow. Figure 2 shows initial training patterns of neural network. After 5,000 training iterations, the predictive output of neural network is shown in figure 3.

Figure 4 and 5 show two experimental results in case that β is infinite and zero, respectively.

As you can see in figure 3, although the training of neural network is not sufficient, the performance of the controller as shown in figure 5 is considerably improved. After 500 sec, the result shows worse performance than the case of infinite β. This is caused by the insufficient training near the set point. This can be reduced by taking appropriate β with on-line training fashion. Instead of adopting

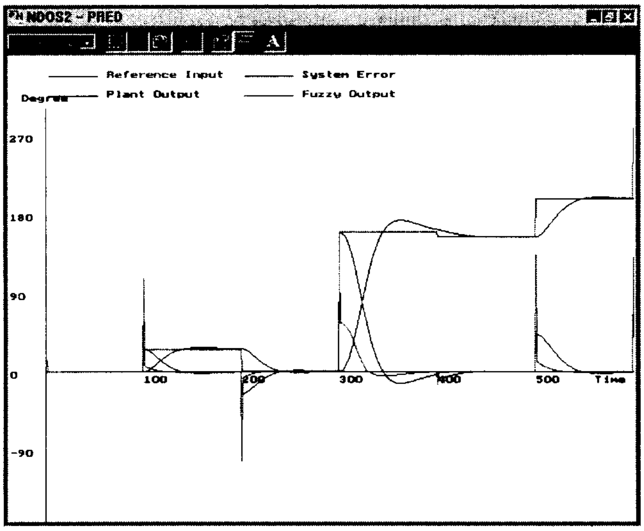

Figure 2: Initial Training Patterns

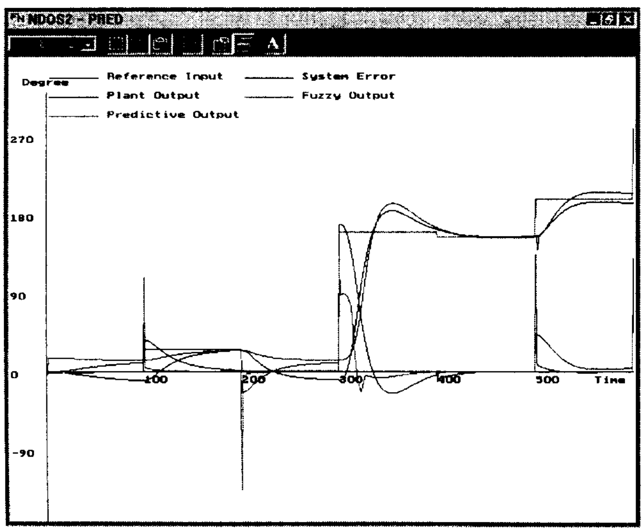

Figure 3: Prediction for Training Patterns

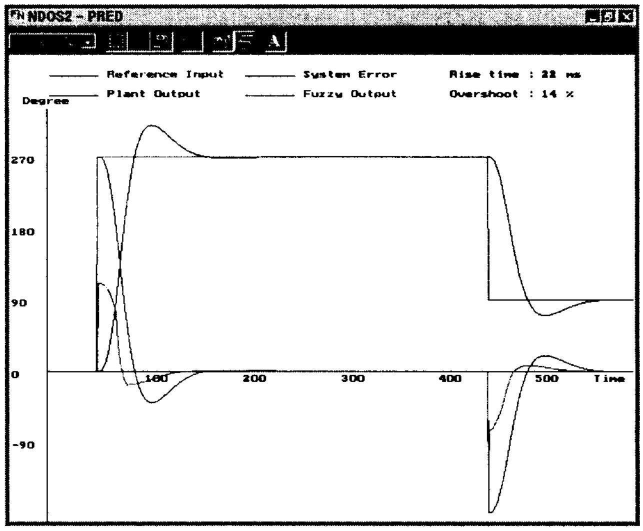

Figure 4: Experimental result with $\beta = \infty$

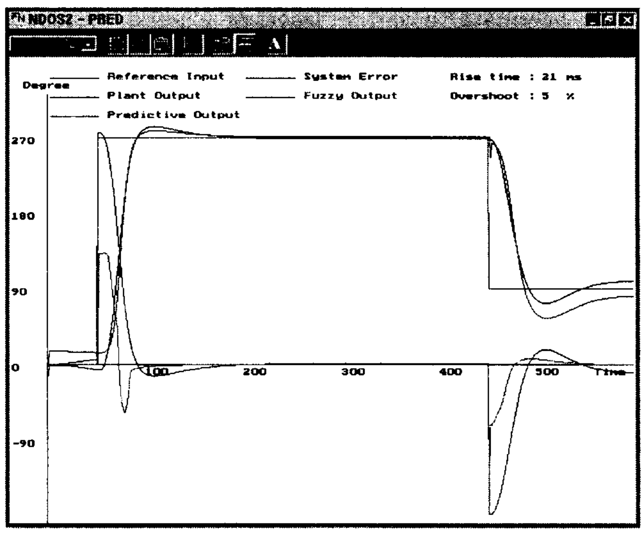

Figure 5: Experimental result with $\beta = 0.0$

$TSSE(t)$, a training error about a pattern mostly near the set point can be used as a degree of training. This method will reduce the performance degradation where the set points insufficiently trained

We used 5 time steps, namely $5ms$ as Δt. That is, neural network generates a predictive output $y(t + 5 \times t_s)$ where t_s is a time step. If this increase, then the rising time will be reduced. The increasing of Δt, however, may affect in the direction of disadvantage. Thus it should be changed carefully.

5 Conclusion

This paper introduces a new fusing method of a neural network and a fuzzy logic controller. In contrast to previous works, this can be shown as a loosely coupled method. Thus, proposed control scheme can be more easily incorporated into existing normal fuzzy logic controllers than previous schemes. The adopting of degree of training and degree of rule effect can also be effectively used without considering a disadvantage caused by insufficient training of the neural network. Experimental results with a DC-servo motor system show that proposed control scheme enhances the performance of the controller. More extensive experiments and more researches for systematic decision of β and Δt are needed.

References

[1] C. C. Lee, "Fuzzy Logic in Control Systems: Fuzzy Logic Controller-Part I/II," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 20, pp. 404–435, MARCH/APRIL 1990.

[2] H. Hellendoorn and C. Thomas, "Defuzzification in Fuzzy Controllers," *Journal of Intelligent and Fuzzy Systems*, vol. 1, pp. 109–123, 1993.

[3] W. Pedrycz, *Fuzzy Control and Fuzzy Systems.* Research Studies Press, 1989.

[4] E. Cox, "Adative fuzzy systems," *IEEE Spectrum*, pp. 27 –31, feb 1993.

[5] P.-Y. Glorennec, "Associating a neural network and fuzzy rules for dynamic process control," in *Neuro-Nimes'90. 3rd International Workshop on Neural Networks and Their Applications*, pp. 211–225, nov 1990.

[6] C.-T. Lin and C. G. Lee, "Neural-Network-Based Fuzzy Logic Control and Decision System," *IEEE Trans. on Computers*, vol. 40, pp. 1320–1336, Dec. 1991.

[7] I. Hayashi, H. Nomura, H. Yamasaki, and N. Wakami, "Construction of fuzzy inference rules by neural network driven fuzzy reasoning and neural network driven fuzzy reasoning with learning functions," *International Journal of Approximate Reasoning*, vol. 6, pp. 241–266, feb 1992.

[8] A. Patrikar and J. Provence, "A self-organizing controller for dynamic processes using neural networks," in *International Joint Conference on Neural Networks 1990, Vol.3*, pp. 359–364, jun 1990.