Fifth IFSA World Congress (1993), 881-886

Self-Organization of Fuzzy Rule Base Using Genetic Algorithm

* Sae-Hie Park*, Yong-Ho Kim**, Young-Keel Choi**, Hyun-Chan Cho***, Hong-Tae Jeon**

*Samsung Electronic Co.
**Dept. of Electronic Eng. ChungAng Univ., Seoul, Korea
***Dept. of Electronic Eng. KITE, Chonan, Korea

Abstract

Fuzzy logic rule-based controller has many desirable advantages, which are simple to implement on the real time and need not the information of structure and dynamic characteristics of the system. Thus, nowadays, the scope of the application of the fuzzy logic controller becomes enlarged. But, if the controlled plant is a time-varying and nonlinear system, it is not easy to construct the fuzzy logic rules which usually need the knowledge of an expert. In this paper, an approach in which the logic control rules can be self-organized using genetic algorithm will be proposed and the effectiveness of the proposed method will be verified by computer simulation of the 2 d.o.f. planar robot manipulator.

1. Introduction

Recently, the scope of application of fuzzy logic has been widely extended to many fields because of its effectiveness in the uncertain information processing. Especially, in designing the control system, fuzzy logic rule-based controller draws lots of attention from the following desirable aspects; (1) it needs not the mathematical modelling of controlled plant, and (2) the fuzzy logic can make the linguistic control and parallel process computing possible.

However, there exist some difficulties in constucting the FLC. One of them is to construct the appropriate fuzzy rule base, which is usually acquried from the knowledge of experts or the experimental results. Futhermore if the controlled plant has the highly nonlinear dynamic characteristics, its implementation becomes more difficult and, even though rule base had been made, it could not be judged whether the rule base is proper or not. Also, it is tedious to verify the effectiveness of the rule base. Thus, recently many researches are going on to leap over these difficulities. Most of the researchs are classified into; 1) fusion with neural networking and 2) self-tuning using look-up table.

This paper presents a novel method that can self-organize the effective and fine rules for a controlled plant. The proposed approach is based on the genetic algorithm of "natural selection and evolution". The genetic algorithm can find the optimal values of scaling factors, which transform the input variables into that of in universe of discourse of fuzzy variables. Also the genetic algorithm can determine the effective membership function of the fuzzy variable.

The organization of this paper is as follows. In section 2, the conventional genetic algorithm will be explained and the method about the self-organization of fuzzy control rules is proposed using the genetic algorithm in section 3. Then robot in section 4. Finally the conclustions are deferred to section 5.

2. Genetic Algorithm

The genetic algorithm proposed by J.H. Holland[4][5][6] comes from "Survival of the fittest" in the nature. The algorithm is a sort of simulated evolution search algorithm to find the optimal solution of the undefined function, Y=G(X). And has three major operators; reporduction, crossover and mutation. The flow of genetic algorithm can be simply explained as follows;

[STEP 1] Determine the string length to represent the variable X in the form of binarycode 1 and 0, and establish an initial string.

[STEP 2] Construct an initial population using the initial string in STEP 1.

[STEP 3] Transform each string of population into decimal code and then find the fitness value of function, G(x).

[STEP 4] Select the proper strings according to the fitness value to form a gene pool.

[STEP 5] Obtain a new population through the evolution process of cossover and mutation between the genes of STEP 4.

[STEP 6] Repeat STEP 3 ~ STEP 5 until some measure is satisfied.

Fig.1(a) shows the string representation of decimal factor as in STEP 1 and Fig.1(b) shows the crossover of two 8-bit strings, X and Y in STEP 5. At the crossover, crossing site is usually selected by random numbers. Meanwhile mutation is simultaneously accomplished with crossover and is done

by a simple bit-transition with a selected random bits. For example, when x = 1001 and the third bit is arbitrarily selected, a new mutated X is 1011.

parameter	string		cross site	
0.750000 ←→	01100000	X = xxxx	xxxx	→ X' = xxxxyyyy
		Y = yyyy	yyyy	→ Y' = yyyyxxxx

(a) String representation (b) Crossover between strings X and Y

Fig 1. Representation of parameter into string and its crossover

The above genetic algorithm has some distinct

1. In the genetic algorithm,the population of solution space is utilized.
2. Genetic algorithm is blind. That is, it needs not the mathematical information of optimizing functions, such as the possibility of derivative and continuity, etc.
3. Genetic algorithm is able to find the global optimum solution.

3. Self-Organization of fuzzy logic control rule.

The conventional FLC is shown in Fig 2. In Fig 2, GC and GE are the scaling factors of error and change of error, respectively.

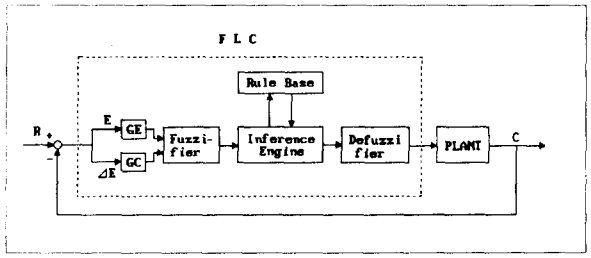


Fig 2. The Conventional FLC

And fuzzy logic rule base consists of "If - Then" rules as,

$$\text{If } X \text{ is } A_i \text{ and } Y \text{ is } B_i \text{ then } U \text{ is } C_i \quad (i=1,\ldots,N) \qquad (1)$$

Antecedence Consequence

where X, Y and U are the input and output variables, respectively, and Ai, Bi and Ci are the fuzzy variables possessing membership functions shown in the Figs. 3 and 4. But it is, as above mentioned, not easy to implement the adquate rules to the controlled plant. Especially, if the controlled plant is a time-varying and nonlinear system it becomes more difficult.

One way to overcome the above diffuculties and self-tune the rule-base according to the controlled plant may be to modify membership functions of fuzzy variables (A_i, B_i and C_i) and adjust the scaling factors(GE and GC) automatically. Thus, in this paper, the membership function of consequent,(Ci, i= 1,..N) of each rule and the scaling factors (GE and GC) are selected as main object to adjust(or optimization factors) to construct the proper rule base fitted in the control environment. The procedure to tune C_i, GE and GC is explained in the followings.

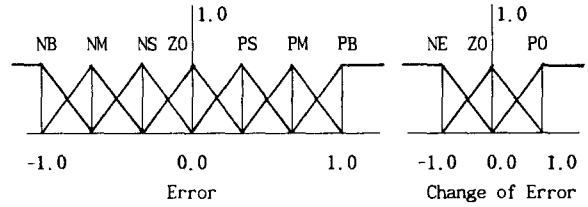


Fig 3. Membership function of antecedent

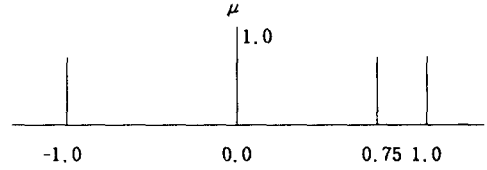


Fig 4. Membership funciton of consequent

3-1. Determination of string and fitness

As above mentioned, since the scaling factors (GE and GC) and membership functions affect massively on the performance of FLC, these parameters are chosen as the optimizatrion objects.

In order to apply the genetic algorithm, the scaling factors must be represented into the binary code. This can be easily accomplished by coding them as the unsigned binary.

But, it is not simple to represent the membership functions of the consequents as binary code. Generally, to characterize the isoceles triangle membership function on the universe of discourse, three vertices of the trianglar must be specified. However, if the length of base line of the trianglar function is assumed to be constant, the position of the upper vertex of triangular is enough to specify the membership function.

Finally one string can be formulated by combining two sets of binary code. Fuzzy rule base and scaling factors can be adjusted by changing the contents of the string.

Next the fitness value must be determined to measure whether the the present fuzzy rule base and scaling factors (or the string) is proper or not. The fitness can be defined as follows.

$$\text{Fitness} = \frac{K_1}{\text{Error}} + \frac{K_2}{\text{Change of error}} + \frac{K_3}{\text{Energy}} \qquad (2)$$

Here k1,k2 and k3 are constant values and energy is input to the plant.

In Eq. 2. the string of high fitness means that the corresponding rule base and scaling factors becomes more adequatre to the control purpose and yields the small error and energy.

3-2. Generation of new rule base by evolution

After determining the form of string and fitness, the next procedure to find the string of the highest fitness value is explained as follows.

At first, an initial string which implies the initial rule base and scaling factors is formulated by generating binary numbers at random. With the initial string, an initial population can be constructed from M strings which are randomly generated. Fig.5 showes an example of the initially generated population. Each row of the population represents a fuzzy rule base and a set of scaling factors, and the initial population contains M sets of fuzzy rule base and scaling factor.

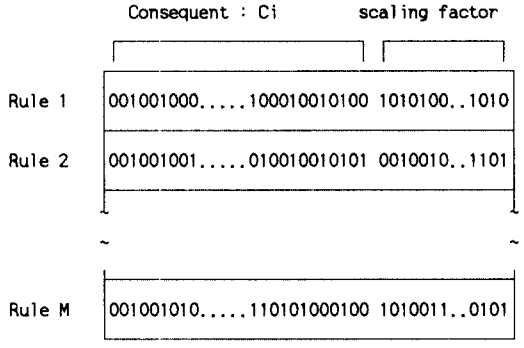


Fig 5. Initial population of strings

After determining the initial population, the fitness value of each rows of population are obtained by the following procedure.

[STEP 1] Find the membership function of consequent of each rule and the scaling factors by transforming each string into decimal value.

[STEP 2] Construct fuzzy logic controller using the rule base and scaling factors obtained from step 1.

[STEP 3] Find error, change of error and energy and then the fitness value

Finding the fitness values of all the strings, next population (or generation) can be constructed by using the good strings, (or more fitted strings) of the initial population. This procedure is done by giving more selection probability to the string of higher fitness value[4] and forming gene pool.

With a gene pool, next generation of population is constructed through crossover between each strings and mutation. Crossover can be done by exchanging arbitrary strings. And mutation can be occured by flipping some bits in the string. For example, if the mutation probability is 0.033 and the length of the string is 184 bits, then only 6 bits of all 184 bits are flipped(0→1 or 1→0) for mutation of the string. Mutation plays a role of preventing the evoluation procedure from falling into the local extrimal.

As each generation are pregressed, the string with the highest fitness value increases and sets to a steady value. The string of the highest value is selected to be the rule base of the FLC, which yields to minial error, and change of error and energy to the plant.

4. Fuzzy logic controller of Robot manipulator

Robot manipulator is a highly nonlinear and time-varying system, and it is not quite easy to construct a optimal controller using fuzzy logic. In this section a fuzzy logic contoller using genetic algorithm for the robot manipulator will be explained.

In FLC, fuzzifier is accomplished by singleton method where the crisp input value of joint error angle $\theta_e(=\theta_d-\theta_a)$, and joint velocity error θ_e $(=\theta_d-\theta_a)$ are transformed into the corresponding fuzzy values. Mamdani's direct inference method is chosen for the inference, and fuzzy rule base consists of the following rules.

$$\text{If } \theta e \text{ is } A_i \text{ and } \theta e \text{ is } B_j \text{ Then U is } C_k \quad (i=1\ldots7,\ j=1\ldots3,\ k=1\ldots21) \tag{3}$$

Here A_i, B_j, C_k are fuzzy variables with membership functions in Fig. 3 and 4.

Finally defuzzification is accomplished with center of gravity method in Eq.4.

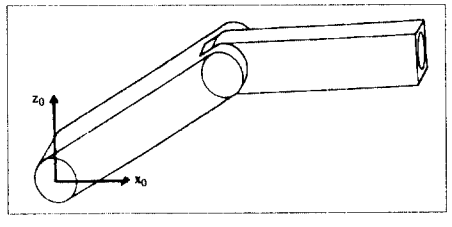
$$U^*=\frac{\int \mu_U \times U}{\int \mu_U} \tag{4}$$

where U^* is a defuzzified torque(input to the robot) and μ_U is a inferred value.

5. Computer Simulation

5-1. Command trajectory and input

To show its effectiveness of the proposed method, two link planar robot manipulator (cf. Fig.6) is chosen for computer simulation.



Mass of link 1 : 2.5Kg
Mass of link 2 : 2.0Kg
Length of link 1: 0.53M
Length of Link 2: 0.47M

Fig 6. two link planar manipulator

The trajectory of the robot end-effector is given as a square in the Cartesian x-z plane. Each length of the square trajectory are 0.25M with the following vertices position.

Position of each vertex is ;

(0.40, 0.00, 0.20), (0.65, 0.00, 0.20),
(0.65, 0.00, 0.45), (0.40, 0.00, 0.45)

Also the string has a 368 bit length which consist of the 32 bit of for the scaling factors and 336 bits for the membership functions of the consequent sentences. Mutation probability is set to 0.033 and the total number of strings in the population is 250.

Furthermore, to verify the effectiveness of the FLC constructed from the square trajectory, another circular trajectory is given as follows,

$$\text{Position X} = 0.6 + 0.20\cos(\theta t) \quad (5a)$$

$$\text{Position Y} = 0.0 \quad (5b)$$

$$\text{Position Z} = 0.3 + 0.15\sin(\theta t) \quad (5c)$$

5-2. Results

Table 1 shows the optimal rule base after 16 generations. And in Table 1 each number represents the tip of vertex of triangle membership function. This show that the FLC consisting only 7~9 rules is sufficient to control the position of two link robot. Also it can be shown that the optimal scaling factors of error and change of error for the joint 1 are 0.034 and 0.353, respectively, and the values of joint 2 are 0.059 and 0.377, respectively.

Change of Error \ Error	NS	ZO	PS
NE	0.42	0.71	
ZO	-0.80	0.0	0.87
PO	-0.28	0.72	

Scaling factor of Error : 0.034
Scaling factor of Change of Error : 0.353

(a) Optimal rule base of joint 1

Change of Error \ Error	NS	ZO	PS
NE	0.18	0.48	
ZO	-0.87	0.0	0.26
PO	0.06	-0.33	

Scaling factor of Error : 0.059
Scaling factor of Change of Error : 0.337

(b) Optimal rule base of joint 2

Table 1. Optimal rule-base for the robot manipulator

Meanwhile, computer simulation results are showed from Fig.7 to Fig.10. Fig.7 is a result of the highest fitness value in the initial population. Fig.8 shows the trajectory along which the robot travelled by the FLC obtained after 16 generations. Comparing these two results and considering the results shown in Fig.10 where the mass-parameters of the robot are varied, the FLC after 16 generations can be verified as the optimal one. Fig.11 shows the variation of the fitness value at every generation.

After obtaining the optimal FLC, another circular trajectory is commanded. Fig.10 shows the trajectory travelled by the two-link planar robot at the initial travelling. From this results, it can be easily concluded that FLC after 16 generations can effectively execute any kinds of trajectory without further generation.

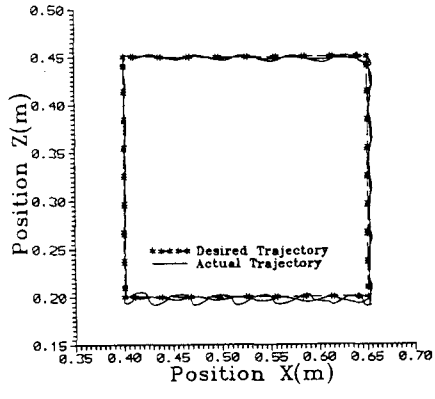


Fig 7. The square trajectory travelled with the highest fitness string of initial control rule

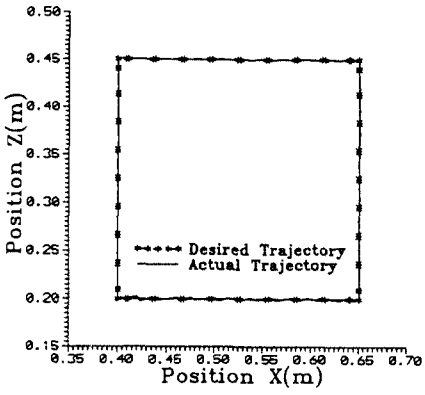


Fig 8. The trajectory travelled with the FLC obtained after 16 generations

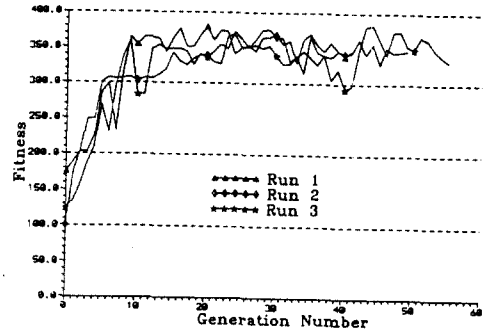


Fig 11. Fitness value plot at each generation

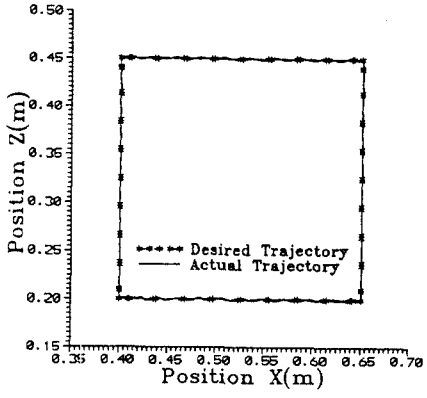


Fig 9. The trajectory with parameter variation (Mass of each links are increased in 30%)

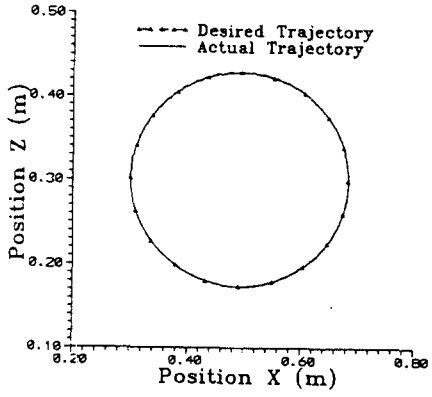


Fig 10. The circular trajectory travelled with the FLC obtained after 16 generations

6. Conclusion

It is not easy to construct the efficient rule base of the FLC. Futhermore, if the controlled plant is highly nonlinear and time varying, it becomes much harder and tedious job to obtain the proper FLC.

In this paper an approach for the self-organization of fuzzy rule base has been proposed using the genetic algorithm, which is a powerful optimization search algorithm and recently attracts many attention. To demonstrate the effectiveness of the proposed method, the FLC of two-link planar is utilized. From the computer simulations, it can be concluded that the FLC using genetic algorithms can self-organize the optimal fuzzy rules and scaling factors.

Reference

[1] Shin-ichi Horikawa et al , "A Fuzzy Controller Using A Neural Network And Its Capability To Learn Control Rules," Proceedings of the International Conference on Fuzzy Logic & Neural Networks , pp.103-106, 1990

[2] Hideyuki TAKAGI, "Fusion Technology of Fuaay Theory and Neural NetWorks - Survey and Future Directions - ," Proceedings of International Conference on Fuzzy Logic & Neural Networks, pp. 13 - 26, 1990

[3] Shihuang SHAO, "Fuzzy Self-Organizing Controller And Its Application for Dynamic Processes," Fuzzy Sets and Systems 26, pp. 151-164, North-Holland , 1988

[4] David E. Goldberg , Genetic Algorithms in Search, Optimization & Machine Learning, Addison-Wesley Publishing Company, 1989.

[5] Yuval DAVIDOR, Genetic Algorithms and Robotics, A Heuristic Strateghy for Optimization, World Scientific Publishing Co. 1991.

[6] Joey K.Paker, Ahmad R. Khoogar, David E. Goldberg, "Inverse Kinematics of Redundant Robots using Genetic Algorithms, " Proceedings 1989 IEEE International Conference on Robotics And Automation, pp.271-276, 1989

[7] Abraham Kandel, Fuzzy Mathematical Techniques with Applications, Addison-Wesley Publishing Company, 1986
[8] K.S. Fu, R.C. Gonzalez, C.S.G. Lee , Robotics control, Sensing, Vision and Intelligence, McGraw-Hill