Optimal Design for Fuzzy Controllers by Genetic Algorithms

Yi-Sheng Zhou and Lin-Ying Lai

***Abstract*—Fuzzy control has been applied to various industrial processes, however, its control rules and membership functions are usually obtained by trial and error. Proposed in this paper is an optimal design for membership functions and control rules simultaneously by a genetic algorithm (GA). GA's are search algorithms based on the mechanics of natural selection and natural genetics. They are easy to implement and efficient for multivariable optimization problems, such as fuzzy controller design. The simulation result shows that the fuzzy controller thus designed can achieve good performance merely by using a few fuzzy variables.**

***Index Terms*—Fuzzy control, genetic algorithms, optimal design.**

I. Introduction

SINCE 1974, when the first fuzzy logic controller (FLC) was proposed by Mandani [1], many FLC applications, such as [2] and [3], have been studied. FLC's use rules in the form "IF [condition] THEN [action]" to linguistically describe the input/output relationship. The membership functions convert linguistic terms into precise numeric values. The control method of modeling human language has many advantages, such as simple calculation, as well high robustness, lack of a need to find the transfer function of the system, suitability for nonlinear systems, etc. The human-friendly controls are extensively implemented by people. In particular, fuzzy control relative to classical control or modern control has a better control effect in the cases of nonlinear, time-varying, uncertain transfer functions of a system.

Most FLC's are designed based on the experience or knowledge of experts. However, it is often the case that no expert is available. Therefore, the trial-and-error method is usually used to find fuzzy control rules and membership functions. For efficiency, an optimal design of control rules and membership functions is desired.

The first genetic algorithm (GA) was developed by Holland in 1975 [4]. Many studies have extended the application of GA's in searching, optimizing, and machine learning [5], [6]. GA's are both global and robust over a wide range of problems. The search procedures rely upon the mechanics of natural genetics. That all natural species can survive by adaptation is the underlying power of GA's. GA's combine a Darwinian survival-of-the-fittest strategy to eliminate unfit components and use random information exchange, with an exploitation of knowledge contained in old solutions, to effect a search mechanism with surprising power and speed. GA's employ multiple concurrent search points called "chromosomes" which process through three genetic operations, reproduction, crossover, and mutations, to generate new search points called "offspring" for the next iterations. Such operations ensure the discovery of an optimal solution to the problem in an appropriate manner.

Recently, there have been some studies using GA's to design membership functions [7], [8], while other studies have used GA's to design control rules for FLC's [9]. However, these designs of FLC's still require the use of an expert's experience, for example, to design control rules for the former or membership functions for the latter. In this paper, to design FLC's more efficiently, a strategy based on GA's is presented to optimally choose both membership functions and control rules simultaneously for the FLC's. The proposed procedure makes the design of FLC's simpler and more efficient.

II. Genetic Algorithms

GA's are search algorithms modeled after the mechanics of natural genetics. They are useful approaches to problems requiring effective and efficient searching, and their use is widespread in applications to business, scientific, and engineering fields. In an optimally designed application, GA's can be used to obtain an approximate solution for single variable or multivariable optimal problems. Before a GA is applied, the optimization problem should be converted to a suitably described function. The corresponding function is called "fitness function." It represents a performance of the problem. The higher the fitness value, the better the system's performance. The objective of a GA is to imitate the genetic operation process, e.g., reproduction, crossover, or mutation, to obtain a solution corresponding to the fitness value.

Recently, many GA's have been presented. The basic construction of a GA can be simply described as follows.

1) *Define the String of a Chromosome:* The string of searching parameters for the optimization problem should be defined first. These parameters are genes in a chromosome, which can be binary coded or real coded and termed "chromosome." Different chromosomes represent different possible solutions.

Paper MSDAD 97–49, presented at the 1995 Industrial Automation and Control Conference: Emerging Technology Applications, Taipei, Taiwan, R.O.C., May 22–27, and approved for publication in the IEEE TRANSACTIONS ON INDUSTRY APPLICATIONS by the Industrial Automation and Control Committee of the IEEE Industry Applications Society. Manuscript submitted for review May 7, 1995 and released for publication August 31, 1999.

Y.-S. Zhou was with the Department of Electrical Engineering, Chung Yuan Christian University, Chung-Li 32023, Taiwan, R.O.C. He is now with Mosel Vitelic Inc., Hsing-Chu, Taiwan, R.O.C. (e-mail: y.s.-chou@mail.mosel.com.tw)

L.-Y. Lai is with the Department of Electrical Engineering, Chung Yuan Christian University, Chung-Li 32023, Taiwan, R.O.C. (e-mail: g8778027@cycu.edu.tw).

Publisher Item Identifier S 0093-9994(00)00051-7.

TABLE I
EXAMPLE OF THE REPRODUCTION OF A GA

Old chromosome	Fitness value	New chromosome
[1, 3, 2, 3, 2, 3, 1]	80	[1, 3, 2, 3, 2, 3, 1]
[2, 3, 1, 1, 3, 2, 2]	67	
[3, 1, 1, 3, 2, 3, 3]	56	
[1, 2, 3, 2, 2, 2, 1]	33	

2) *Define the Fitness Function:* The fitness function is the performance index of a GA to resolve the viability of each chromosome. The design of the fitness function is according to the performance requirements of the problem, e.g., convergence value, error, rise time, etc.
3) *Generate an Initial Population:* N sets of chromosomes should be randomly generated before using a GA operation. These chromosomes are called the initial population. The size of the population, N, is chosen according to the sophistication of the optimization problem. Generally speaking, the larger values of N require fewer generations to come to a convergent solution. However, the total computation effect depends on N times the generation numbers.
4) *Generate the Next Generation or Stop:* GA's use the operations of reproduction, crossover, and mutation to generate the next generation. From generation to generation, the maximum value of the fitness value is achieved for each generation.
   a) *Reproduction:* Reproduction is the operator carrying old strings through into a new population, depending on the fitness value. Strings with high fitness values obtain a larger number of copies in the next generation. An example of such an operation is shown in Table I.
   b) *Crossover:* Crossover is a recombination operator incorporated with reproduction. It is an effective way of exchanging information and recombining segments from high-fitness individuals. The crossover procedure is to randomly select a pair of strings from a mating pool, then randomly determine the crossover position. An example of the operation is shown in Table II.
   c) *Mutation:* The mutation operator is used to avoid the possibility of mistaking a local optimum for a global one. It is an occasional random change at some string position based on the mutation probability. An example of the operation is shown in Table III.

TABLE II
EXAMPLE OF THE CROSSOVER OF A GA

Old chromosome	Fitness value	New chromosome
[1, 3, 2, 3, 2, 3, 1]	80	
[2, 3, 1, 1, 3, 2, 2]	67	[2, 1, 1, 3, 2, 2, 2]
↑ ↑ ↑		
↓ ↓ ↓		
[3, 1, 1, 3, 2, 3, 3]	56	[3, 3, 1, 1, 3, 3, 3]
[1, 2, 3, 2, 2, 2, 1]	33	

TABLE III
EXAMPLE OF THE MUTATION OF A GA

Old chromosome	Fitness value	New chromosome
[1, 3, 2, 3, 2, 3, 1]	80	
[2, 3, 1, 1, 3, 2, 2]	67	
[3, 1, 1, 3, 2, 3, 3]	56	
[1, 2, 3, 2, 2, 2, 1]	33	[1, 2, 1, 2, 3, 2, 2]
↑ ↑ ↑		

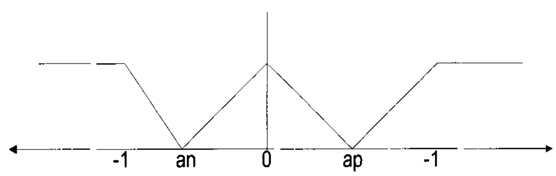

Fig. 1. Triangular-shaped membership functions.

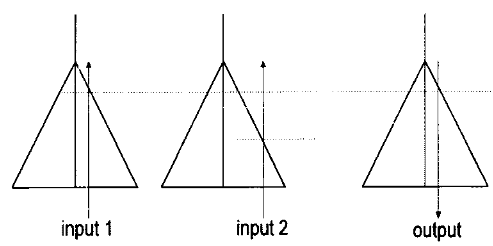

Fig. 2. Maximum-corresponding reasoning algorithms.

III. DESIGNING FLC'S USING GA'S

The design of FLC's using GA's is briefly described as follows. In an FLC design, the emphases are placed on the design of membership functions in the fuzzification procedure and the consequent variables in the fuzzy control rules. The optimal FLC design is initiated by using three fuzzy variables, i.e., the linguistic values *NB, ZO,* and *PB.* If the performance of an FLC thus designed cannot satisfy the user's requirement, the number of fuzzy variables will automatically increase by one until the requirement is satisfied. The membership functions used are triangular shaped, as shown in Fig. 1. A triangular-shaped membership function can be parameterized by the two vertexes at the base, *ap* and *an.* The defuzzification algorithm used is the simple maximum-corresponding method, as shown in Fig. 2.

The chromosome of the GA includes two parts, the $n \times n$ consequent variables on the fuzzy control table and the parameters of the membership functions. To reduce the number of genes in the chromosome, the discrete real-coded genes are used. An example of the collocation of the genes in the chromosome is shown in the following:

$$[1, 1, 2, 1, 2, 3, 3, 3, 2, 0.5625, -0.4375, 0.5, -0.125, 0.9375, -0.75].$$

TABLE IV
CONTROL RULE TABLE FOR THE CHROMOSOME EXAMPLE

e \ er	PL	ZR	NL
PL	1	1	3
ZR	1	2	3
NL	2	3	2

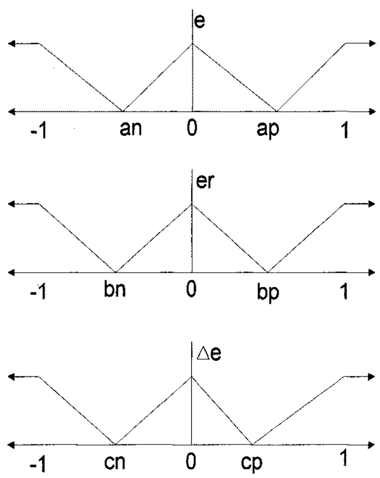

Fig.3. The membership function for the chromosome.

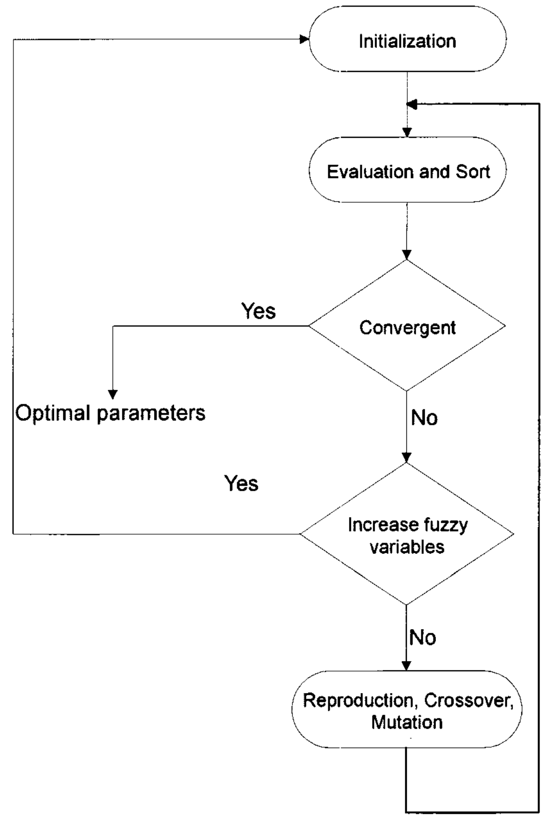

Fig. 4. Evolution procedure of the GA.

The first through ninth genes in the chromosome are the elements of the control rule table, as shown in Table IV. The numbers 1, 2, and 3 on the fuzzy control rule table represent the linguistic values *PB, ZO,* and *NB,* respectively. The tenth through the fifteenth genes are the parameters of the membership *ap,an,bp,bn,cp,* and *cn,* as shown in Fig. 3. Sixteen sections are demarcated from 0 to 1 and from 0 to –1, respectively. The parameter of each optimal membership function is searched from within these discrete points. At the left vertex (*ap, bp,* and *cp*), *PL* is assigned to be the linguistic value, and at the right vertex (*an, bn,* and *cn*), *ZR* is assigned.

In this paper, the fitness function is composed of three performance indexes in the system's step response: the maximum overshoot, the rise time, and the accumulated error. The maximum overshoot is the percentage of the maximum excess value versus the final value. The rise time is the time for the step response to reach from 0 to its final value. The accumulated error is the summation of the absolute error of the step response at the sampling instants. Each performance index is transformed to a scale from 0 to 100. The average value is used as the fitness value. Different weight can be put on each index according to system requirements. For example, for a system in which it is desired to shorten the rise time, the weighting for the rise time can be changed to 1.3 and other weightings to 0.9 and 0.8. The population is seriated by the fitness function. The first chromosome has the highest fitness function. If the performance of the FLC satisfies the design requirement, then the operation of the GA will stop; otherwise, it will continue to generate the next generation or increase the number of fuzzy variables.

The evolution procedure for the GA is shown in Fig. 4. N chromosomes of an initial population are randomly generated in the initialization segment. Then, the fitness function of each chromosome is evaluated. The first chromosome has the highest fitness value, i.e., the chromosome has the best step response of the system in this generation. If the requirement is not achieved, chromosomes of the current generation will go through three genetic operations, reproduce, crossover, and mutate, to generation the next generation. The GA operation will repeat the procedure until the requirement is achieved. If the fitness value remains constant for a certain number of generations, say, M generations, the GA will increase the number of fuzzy variables by one automatically and reinitialize the procedure.

IV. SIMULATION AND DISCUSSION

To test the proposed method, we use the fuzzy proportional-integral-derivative (PID) controller structure [10], as shown in Fig. 5. The PID control is the master controller and the fuzzy control is the slave control to enhance the master one. The antecedent variables of the fuzzy control rule are the error (e) and the error rate (er) of the system's step response. The e and the er are defined as follows:

$$e(k) = y(k) - y_r(k) \quad (1)$$
$$er(k) = (e(k) - e(k-1))/T \quad (2)$$

where y_r is the reference output of the system, and T is the sampling period. The consequent variable is the error variation (Δe) in the FLC system. The FLC uses the variation to tune the errors of the system, thus, it can ameliorate the performance

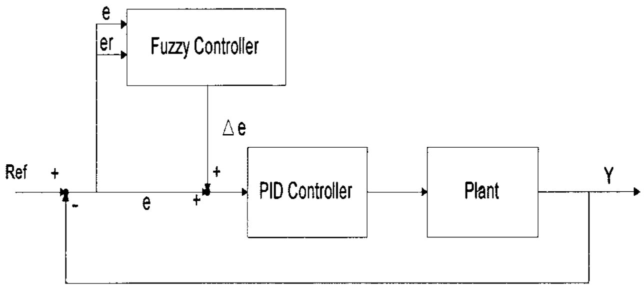

Fig. 5. Fuzzy PID controller structure.

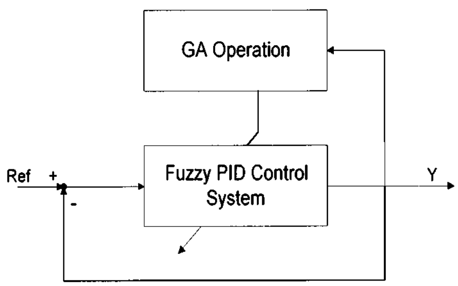

Fig. 6. GA-tuned fuzzy PID system.

of system's step response. The structure of the GA-tuned fuzzy PID control is shown in Fig. 6.

The transfer function of the simulation plant is

$$\frac{1}{(s+0.1)(s+0.2)(s+0.7)}. \tag{3}$$

The PID controller parameters are chosen initially according to Ziegler–Nichols' rule [11]. The resulting values of *Kp, Ki,* and *Kd* are 9.257, 8.6, and 1.45, respectively. The analog PID control plant system is discretized by using MATLAB. The population number N is set at 20. The weightings are 1.2, 1.2, and 0.6 for the maximum overshoot, the accumulated error, and the rise time, respectively. The conditions for stopping are a maximum overshoot under 1.5%, an accumulated error under 12, and a rise time under 0.25 s, i.e., the fitness value must exceed 87.27. The number of generations for reinitialization, M, is set at 50. The chromosome with the highest fitness value is reproduced, the second through the eleventh are crossed and the twelfth through the twentieth are mutated.

Two techniques are used to accelerate the search speed: 1) changing the mutation rate and 2) constructing a data bank of fitness values. When the highest fitness value of the chromosomes remains the same over three generations, the GA operation will increase the rate of mutation to increase the searching speed. The data bank stores the foregone value of each already generated chromosome. The system will search for the fitness value of the same chromosome before calculating the fitness function. Fig. 7 shows the highest fitness value of each generation. Fig. 8 shows the step responses of the chromosomes of the highest fitness from the first, fifth, tenth, twentieth, and twenty-fourth generations. The result shows that a better fitness value is achieved from generation to generation. A comparison of the step responses between the GA-tuned fuzzy PID control

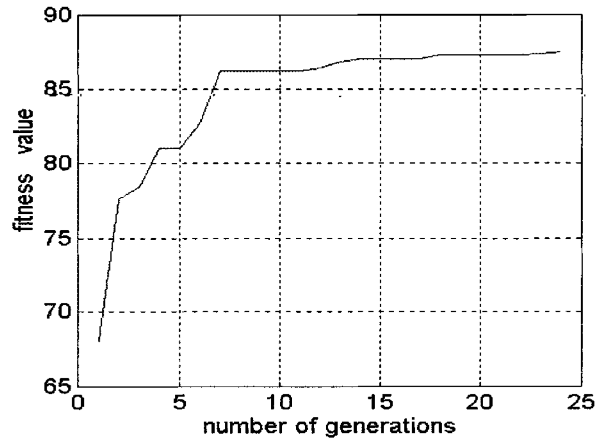

Fig. 7. The highest fitness value of each generation.

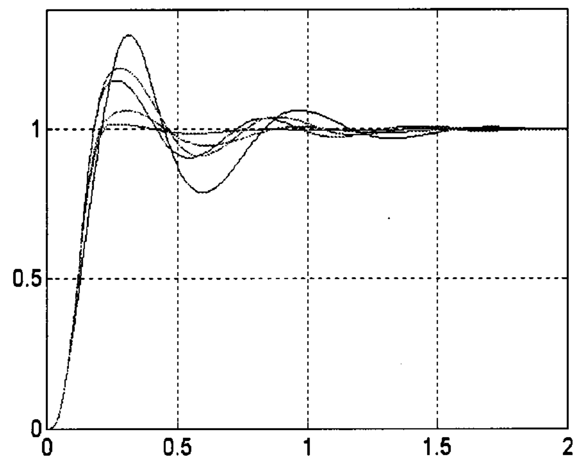

Fig. 8. Step responses of the best chromosomes from some generations.

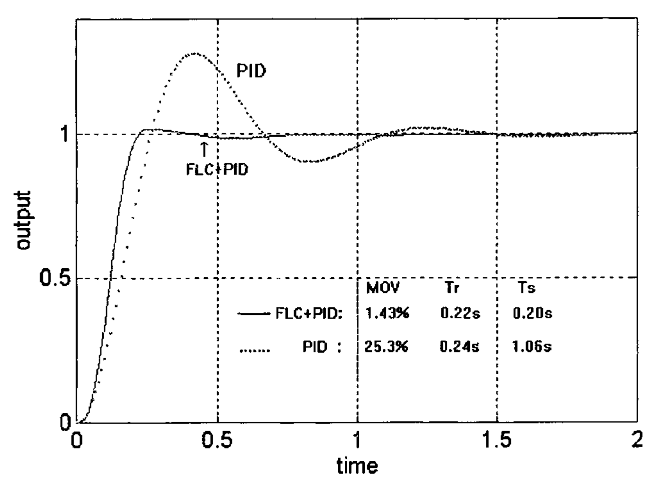

Fig. 9. Step responses of the PID controller and the optimal fuzzy PID controller systems.

system and the PID control system is shown in Fig. 9. The optimal chromosome in the FLC is found in the twenty-fourth generation. The control rule table and membership functions of the optimal FLC are shown in Fig. 10. Note that only the third, the fifth, and the sixth control rules in the FLC system are actually

e \ er	PL	ZR	NL
PL	1	1	3
ZR	1	2	3
NL	2	3	2

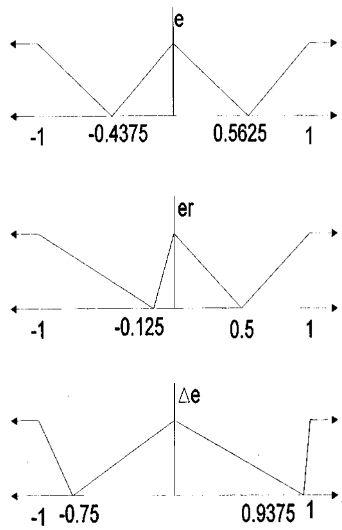

Fig. 10. The fuzzy control rules and membership functions of the optimal FLC.

used. The simulation result shows that the GA approach is efficient and effective for obtaining an optimal FLC.

V. Conclusion

In this paper, a GA has been used for developing an optimal fuzzy controller. The simulation result shows that the proposed method is effective and efficient. This technique can save time when compared to a conventional trial-and-error design procedure. The optimal fuzzy controller through a systematic search requires only a few fuzzy variables. It does not require extra professional expertise or mathematical analysis for the plant's model.

For future study, it will be worthwhile to implement FLC's using GA's for different applications.

References

[1] E. H. Mandani, "Applications of fuzzy algorithms for control of simple dynamic plants," *Proc. Inst. Elect. Eng.*, vol. 121, no. 12, pp. 1585–1588, 1974.

[2] P. J. King and E. H. Mandani, "The application of fuzzy control systems to industrial processes," *Automatica*, vol. 13, no. 3, pp. 235–242, 1977.

[3] M. Sugeno, *Industrial Applications of Fuzzy Control*. New York: Elsevier, 1985.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[5] D. E. Goldberg and M. P. Samtani, "Engineering optimization via genetic algorithm," in *Proc. 9th Conf. Electronic Computation*, 1986, pp. 471–482.

[6] A. C. Englander, "Machine learning of visual recognition using genetic algorithms," in *Proc. 1st Int. Conf. Genetic Algorithms*, 1985, pp. 197–202.

[7] C. Karr, "Genetic algorithms for a fuzzy controller, learning from examples," *AI Expert*, vol. 6, no. 1, pp. 26–33, 1991.

[8] ——, "Applying genetics to fuzzy logic," *AI Expert*, vol. 6, no. 3, pp. 8–43, 1991.

[9] J. C. Hsu and C. Yeh, "A GA-based self-learning fuzzy control for wheel-lift maneuver," in *Proc. 1994 R.O.C. Automatic Control Conf.*, pp. 441–446.

[10] L. Y. Lai and M. Y. Lee, "Fuzzy tuning of integrator outputs of PID controllers for a DC motor system," *Chung Yuan J.*, vol. XXII, pp. 126–137, Dec. 1993.

[11] J. B. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 64, pp. 759–768, 1942.

Yi-Sheng Zhou was born in Kaoshiung, Taiwan, R.O.C., in 1969. He received the B.S. and M.S. degrees in electrical engineering from Chung Yuan Christian University, Chung-Li, Taiwan, R.O.C., in 1993 and 1995, respectively.

Following two years of military service, he joined Mosel Vitelic Inc., Hsing-Chu, Taiwan, R.O.C., in 1997. His research interests are fuzzy control and genetic algorithms.

Lin-Ying Lai was born in Taipei, Taiwan, R.O.C., in 1961. She received the B.S. and M.S. degrees in power mechanical engineering from National Tsing Hua University, Hsing Chu, Taiwan, R.O.C., in 1983 and 1985, respectively, and the Ph.D. degree in aerospace engineering from Virginia Polytechnic Institute and State University, Blacksburg, in 1988.

She has been with the Electrical Engineering Department, Chung Yuan Christian University, Chung-Li, Taiwan, R.O.C., since August 1989. Her research interests are fuzzy control, genetic algorithms, neural networks, system identification, optimal estimation, and optimal control.