A Study on Genetic Algorithms for Automatic Fuzzy Rule Generation

Hyun-Joon Cho and Bo-Hyeun Wang
LG Electronics Research Center
16 Woomyeon Dong, Seocho-Gu, Seoul, 137-140, Korea.
Email: loxias@erc.goldstar.co.kr

Abstract

The application of genetic algorithms to fuzzy rule generation holds a great deal of promise in overcoming difficult problems in fuzzy systems design. There are some aspects to be considered when genetic algorithms are used for generating fuzzy rules. In this paper, we will present an aspect about the control surface constructed by the resultant rules. In the extensive simulations, an important observation that the rules searched by genetic algorithms are randomly scattered is made and a solution to this problem is provided by including a smoothness cost in the objective function. We apply the fuzzy rules generated by genetic algorithms to the fuzzy truck backer-upper control system and compare them with the rules made by an expert.

1. Introduction

Mamdani and his co-workers pioneered a method of fuzzy logic control [1]. The key idea behind their approach was to replace the operator's complex tasks with machines in order to control a class of complex systems. In this case, a control designer captures operators' knowledge and converts it into a set of fuzzy control rules. The benefit of the simple design procedure of a fuzzy controller leads to the successful applications of a variety of engineering systems [2].

On the contrary to the successes, there is still some reluctance in industry to adopt the fuzzy control approach. One reason for industry reluctance is that there has been little practical guidance on the design of fuzzy control rules [3]. Most reported applications have resorted to heuristic methods for constructing the rule base. But it is difficult to develop a control strategy and to calibrate control rules when complex processes of higher dimensions are involved. Since fuzzy rule generation is difficult and time consuming procedure, it is required to have a systematic method for constructing appropriate rules.

Automatic rule generation is required to overcome the difficulty. Learning capability of neural networks and optimization techniques such as genetic algorithms play the central role [4,5].

Genetic algorithms (GAs) developed by Holland [6] incorporate the features of natural evolution in computer algorithms. In contrast that traditional optimization techniques deal with a single candidates, GAs operate on a population of candidates. This makes it possible for GAs to search several areas of a solution space. GAs work through function evaluation, not through differentiation or other such means, and aim to optimize a user-defined function irrespective of the form. Because of this trait, GAs do not care what type of problem it is asked to optimize, only that it be properly coded. Thus GAs are able to solve a wide range of problems; linear, nonlinear, discontinuous, discrete, etc.

Several researches that use those characteristics of GAs in fuzzy rule generation, have been reported in [7,8]. They used the genetic-based learning as a general approach to synthesizing fuzzy control strategies, and demonstrated that GAs can be used effectively for generating fuzzy rules in fuzzy control structures. Nowadays the use of GAs for optimizing fuzzy controllers is more and more increasing.

Investigating the rules generated by GAs, we often observe peculiar phenomena in which the rules searched by GAs are randomly scattered. The control surface that is constructed by such rules is rough and irregular. However, structured knowledge of a human expert usually provides a quite smooth surface. Moreover the irregularity of control surface may result in negative effects on the control inputs.

In this paper, we briefly describe a general method for automatic fuzzy rule generation using GAs, and a solution to the problem, that is, the irregularity of the resultant control surface, is provided by including a smoothness cost in the objective function. Finally, an interesting phenomenon related to the contributions of the rules to control is introduced.

2. Automatic Rule Generation of Fuzzy Controllers Using Genetic Algorithms

In this section, we will describe automatic rule generation using GAs. The application of GAs to fuzzy rule generation holds a great deal of promise in overcoming difficult problems in fuzzy systems design, that is, design optimality. Robustness of GAs enables them to cover a complex search space in a relatively short period of time, while ensuring an optimal or near optimal solution. The rules generated by GAs are applied to the truck backer-upper control system.

2.1 Fuzzy truck Backer-Upper System

Truck backer-upper control is to make the truck arrive at the loading dock at a right angle ($\phi_f = 90°$) and to align the rear center of the truck (x, y) with the desired loading dock (x_f, y_f). The truck moves backward by some fixed distance at every stage. Fig. 1 shows a simulated truck and loading zone.

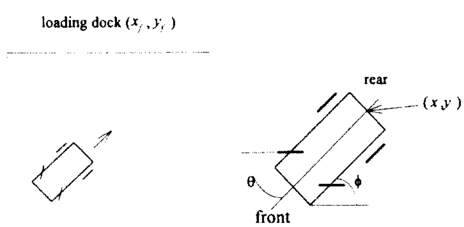

Fig. 1. Diagram of simulated truck and loading zone.

The controller's input variables are the truck angel ϕ and the x-position coordinate x. The output variable is the steering-angle signal θ. Although y-location can be considered a variable, it is assumed in this paper that the truck is sufficiently far from the loading dock in the y-direction so that the y-distance can be ignored.

The equations of motion for the truck are given as

$$x' = x + r\cos(\phi'),$$
$$y' = y + r\sin(\phi'),$$
$$\phi' = \phi + \theta,$$

where r is a fixed distance that the truck backs at each time step (we use $r = 2.$ in this paper), and ϕ', x', and y' are the new truck angle, x-location, and y-location, respectively. The variable ranges are as follows:

$$0 \le x \le 100$$
$$-90 \le \phi \le 270$$
$$-30 \le \theta \le 30$$

The structure of a control rule is written as:

If x is L_x and ϕ is L_ϕ, then θ is L_θ,

where L_x, L_ϕ, and L_θ are linguistic values of x, ϕ, and θ, respectively. Each fuzzy variable takes the following linguistic values:

L_x = {LE, LC, CE, RC, RI},

L_ϕ = {RB, RU, RV, VE, LV, LU, LB},

L_θ = {NB, NM, NS, ZE, PS, PM, PB}.

This leads us to have 35 fuzzy rules, if the rule base is complete.

We chose the same linguistic values of the input and output fuzzy variables as Kosko [9] for the comparison of the rules made by GAs and the expert rules.

2.2 Algorithm Description

The basis for the software used in this paper is the Simple Genetic Algorithm (SGA) program developed by Goldberg [10]. We used integer coding scheme instead of SGA's binary coding, roulette wheel selection scheme, one-point crossover, random mutation, and elitism.

First the number of genes are determined from the size of the rule set. Suppose that we have a two input and single output system. Furthermore we assume that the first input space, the second input space, and the output space are partitioned to l, m, and n sets, respectively. So the rule set consists of ($l \times m$) fuzzy rules. GAs search for such ($l \times m$) fuzzy rules so that they optimize a given objective function. For this, we represent a fuzzy rule base as a string which consists of ($l \times m$) genes. Fig. 2 shows a string example in the case of $l = 5$, $m = 7$, and $n = 7$. The integer in a gene indicates a specific linguistic value.

Along with encoding, another important task is to construct objective function to be optimized. In general, the objective function may be of the following form:

$$J = \sum_i a_i C_i,$$

where a_i's are correction factors that adjust the dimensions or orders of the functions C_i's. In this section, we have employed the following cost functions:

$$C_1 = \sqrt{(\phi_f - \phi)^2 + (x_f - x)^2},$$

$$C_2 = (\textit{total time}),$$

$$C_3 = (\textit{time from initial position to the position}, x = 50, \phi = 90^\circ).$$

Each cost function is designed for a particular purpose. C_1 minimizes docking error, C_2 minimizes total time from the initial positions to loading dock, and C_3 minimizes the time from the initial positions to docking conditional position. Although C_3 seems to be redundant, it is necessary to the stability of the controller when the truck is located initially in the position of which y-location is upper than that of the initial conditions used for genetic learning.

2.3 Simulation Description

The loading zone corresponds to the plane [0, 100] × [0, 100], and (x_f, y_f) equals (50, 100). The initial conditions used are shown in Table 1. These position-symmetric initial conditions include 17 points made intentionally for the completion of the fuzzy control rules.

We have tested the controller whether it is able to control the truck under any initial condition. The input space $x \times \phi$, [10, 90] × [-90, 270] was partitioned into 29241 grids. For all the conditions the truck was successfully docked to the desired loading dock.

TABLE 1. Initial conditions for truck backer-upper control

	Initial Conditions				
x	10	40	50	60	90
ϕ	-45, 22.5, 90, 157.5, 225	22.5, 157.5	-45, 90, 225	22.5, 157.5	-45, 22.5, 90, 157.5, 225

2.4 Control Results

The fuzzy controller with the rules generated by GAs produced successful truck backing-up trajectories starting from any initial position. Fig. 3 shows some examples of the fuzzy-controlled truck trajectories from different initial positions. Table 2 summarizes a fuzzy control rules generated by GAs.

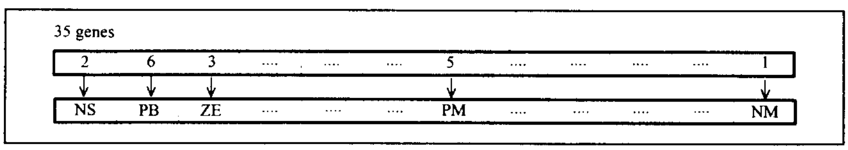

Fig. 2. Encoding for GA.

Fig. 3. Truck trajectories of the fuzzy controller for initial positions (x, y, ϕ): (a) (20, 20, 30), (b) (80, 30, -80).

TABLE 2. Fuzzy logic rule table for the fuzzy truck backer-upper controller

		x				
		LE	*LC*	*CE*	*RC*	*RI*
ϕ	*RB*	PB	PB	PM	PM	NB
	RU	ZE	PS	PB	PB	PB
	RV	NB	NS	PS	PM	PB
	VE	NB	NB	ZE	PB	PB
	LV	NB	NB	NM	PS	PB
	LU	NB	NM	PS	NS	NS
	LB	PB	NM	NM	NS	NM

3. Smoothness Cost

In the rule generation using GAs, we consistently observe peculiar phenomena in which the rules searched by GAs are randomly scattered. The control surface that is constructed by such rules is rough and irregular. However, structured knowledge of a human expert usually provides a quite smooth surface. To avoid the irregular control surface, we apply the following cost function:

$$C_4 = \sum_{i=1}^{7} \sum_{\substack{j=1 \\ (i+j)\neq odd}}^{5} \left(\sum_{k=i-1}^{i+1} (R_{ij} - R_{kj}) + \sum_{m=j-1}^{j+1} (R_{ij} - R_{im}) \right)$$

To compute C_4, each linguistic value is first converted into an integer, i.e., $\{NB \to 0,\ NM \to 1,\ \ldots,\ PB \to 6\}$. R_{ij} represents the integer that is assigned to the ijth cell of the rule table. As shown in Fig. 4, the discrepancies between R_{ij} and its four neighboring values are computed and the summation of them is denoted as P_{ij}. To avoid a repetitive computation, we perform the P_{ij} calculation for every other cell that is represented by the gray cell in Fig. 4. A smaller $\sum P_{ij}$ provides a smoother control surface.

As was mentioned above, we have observed that the fuzzy rules optimized by GAs form an irregular or a rough control surface as shown in Fig 5.(a). In order to resolve this problem, we have added smoothness cost given as C_4 to the objective function used in Section 2. The fuzzy rules generated by this objective function are shown in Table 3. Comparing Fig. 5.(a) with Fig. 5.(b), it is easy to see that the control surface achieved by this method is smoother than that of the previous rulebase.

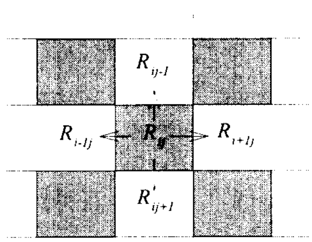

Fig. 4. Computation of smoothness cost.

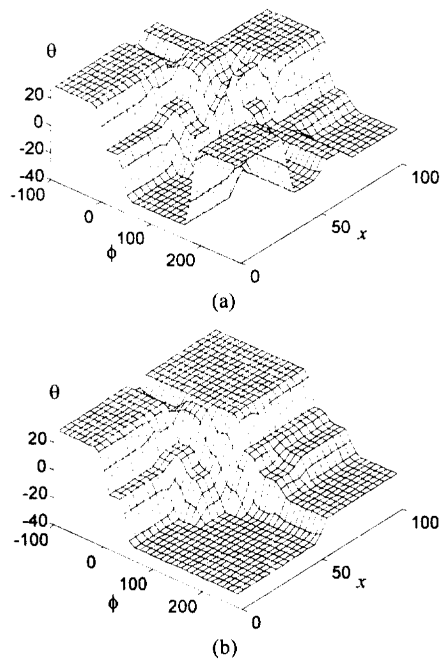

Fig. 5. Control surfaces of (a) SGA, (b) GAiSC

TABLE 3. Fuzzy logic rule table generated by the GA including smoothness cost

		x				
		LE	*LC*	*CE*	*RC*	*RI*
ϕ	*RB*	PB	PB	PM	PB	PB
	RU	ZE	PS	PB	PB	PB
	RV	NB	NS	PS	PB	PB
	VE	NB	NM	ZE	PB	PB
	LV	NB	NB	NS	PS	PS
	LU	NB	NB	NB	NS	ZE
	LB	NB	NB	NB	NM	NM

To understand how the objective function that includes the smoothness cost affects the performance of controller, we computed the total time steps required to dock the truck under all the test initial conditions mentioned in section 2.3. As shown in Table 4, the total time steps of the GAs including the smoothness cost (GAiSC) are the smallest of all. The rules searched by GAs are optimized for the given initial conditions mentioned in section 2.3, but only the 17 initial positions cannot cover all the possible positions where the truck could possibly be. So the rules might not be optimized for the positions that are not trained in genetic learning. However, the rules generated by the GAiSC backed up the truck better for such untrained positions due to the influence of the smoothness cost on the neighboring rules.

TABLE 4. Comparison of the control performances

	Smoothness cost included	SGA	Kosko designed
total time steps	1,353,954	1,405,495	1,413,340
docking failure no.	0	0	0

Another interesting phenomenon was observed related to the smoothness cost. We applied the 17 position-symmetric initial conditions to the three controllers, and watched over the contribution of each rule. Fig. 6 shows the accumulated numbers of the contributions by the rules of the two GAs and those of Kosko. We can see that the contributions of Kosko's and the GAiSC's rules are almost symmetric.

There are some systems whose control actions are symmetric for a center point, such as the inverted pendulum. The truck backer-upper system might be symmetric one, so the control rules would be symmetric if a human expert designed them. As can be seen in [9], the control surface of the rules designed by an expert is symmetric and the contributions of the rules for the position-symmetric initial conditions are almost symmetric.

We did not design the smoothness cost to obtain such symmetric rules, but it is an interesting phenomenon come from the use of the smoothness cost. More theoretical details on this issue is now under investigation.

		x				
		LE	*LC*	*CE*	*RC*	*RI*
ϕ	*RB*	5	0	3	2	2
	RU	67	74	13	5	2
	RV	3	89	71	27	2
	VE	2	1	446	447	2
	LV	2	4	459	510	23
	LU	2	7	24	89	71
	LB	2	3	3	0	10

(a)

		x				
		LE	*LC*	*CE*	*RC*	*RI*
ϕ	*RB*	50	14	3	2	2
	RU	64	74	24	5	4
	RV	8	201	278	97	4
	VE	2	227	490	266	2
	LV	4	97	316	239	8
	LU	4	5	24	74	64
	LB	2	2	3	14	50

(b)

		x				
		LE	*LC*	*CE*	*RC*	*RI*
ϕ	*RB*	2	0	3	2	2
	RU	74	69	12	5	4
	RV	5	229	260	72	4
	VE	2	227	406	179	2
	LV	4	92	252	210	54
	LU	4	4	15	73	77
	LB	2	1	2	0	8

(c)

Fig. 6. Contributions of the rules to control, (a) SGA , (b) Kosko, (c) GAiSC.

4. Conclusions

In this paper, methods to generate rules for fuzzy controllers using GAs have been described, and an aspect about the control surface constructed by the resultant rules was investigated.

Based on our extensive simulations, we found that the irregular property of the resultant control surface is common in GAs' rule generation. To resolve this problem, we have introduced a smoothness cost function and have been able to construct a fuzzy rule base that has a smoother rule surface with a good control performance. Also we observed an interesting phenomenon related to the symmetry of the control rules' contributions.

Future work will attempt to develop GAs which are more tractable for fuzzy rule generation. Since it is conjectured that a rule surface extracted from structured knowledge of human experts is also of a quite smooth rule surface, it is essential that efforts be directed toward inventing genetic operators for smoothening the rule surface.

References

[1] S. Assilian and E. H. Mamdani, "An experiment in linguistic synthesis with fuzzy logic controllers," *Int. J. Man-Machine Studies*, vol. 7, pp. 1-13, 1974.

[2] D. E. Thomas and B. A. Helouvry, "Fuzzy logic control: A taxonomy of demonstrated benefits," *Proceedings of IEEE*, vol. 83, no. 3, pp. 341-492, 1995.

[3] R. M. Tong, "A retrospective view of fuzzy control systems," *Fuzzy Sets and Systems*, vol. 14, pp. 199-210, 1984.

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

[5] Cho, K. B. and Wang, B. H., "Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction," accepted to *Fuzzy Sets and Systems*, 1995.

[6] J. H. Holland, *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor, Mich., 1975.

[7] A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms," *IEEE Trans. Fuzzy Systems*, vol. 3, no. 2, pp. 129-139, 1995.

[8] P. Thrift, "Fuzzy logic synthesis with genetic algorithms," *Proceedings of the Fourth Int. Conference on Genetic Algorithms*, San Diego, pp. 509-513, 1991.

[9] B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice-Hall, 1992.

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