# DESIGN OF DC MOTOR FUZZY-PI CONTROLLER USING GENETIC ALGORITHMS

Mehmet Bulut — *bulutm@hotmail.com*
Galip Cansever — *cansever@yildiz.edu.tr*
S.Vakkas Ustun — *sustun@yildiz.edu.tr*
*Yidiz Techical University, Faculty of Engineering, Department of Electrical Engineering, Istanbul, Turkey*

*Key words: FuzzyLogic,Genetic Algorithms, Learning Rule base, DcMotor*

## ABSTRACT

**In this study, genetic algorithms are used to learn fuzzy PI control rules for PC based direct current motor speed control . It is developed in three stages: the first one is a fuzzy rule genetic generating process based on a rule learning iterative approach, the second applying rules to process which is best rule base in each generation, and the third one is ended learning process if the best rule have desired fitness value. The three components of the learning process are developed to formulate suitable genetic algorithms.**

## I. INTRODUCTION

(FLCs) are now considered as one of the most important applications of the fuzzy-rule-based systems. The experience of skilled operators and the knowledge of control engineers are expressed qualitatively by a set of fuzzy control rules. The construction of fuzzy rules has been mainly based on the operator's control experience or actions. However, converting the experts know-how into if- then rules is difficult and often results are incomplete, unnecessary and include conflicting knowledge, since operators and control engineers are not capable of specific details or cannot express all their knowledge including intuition and inspiration [1] .

Genetic algorithms (GAs) are search algorithms based on the mechanics of natural selection and natural genetics. GAs have the properties that make them a powerful technique for selecting high-performance parameters for FLCs. For example, Lee and Takagi [2] design a fuzzy system using a genetic algorithm for the inverted pendulum. Karr and Gentry [3] control the pH of the acid base system with fuzzy system's input membership functions manipulated by a GA. Park et al. [4] optimise a fuzzy reasoning model by genetic algorithm to control a direct current series motor.

Previous works are focused only on optimising the FLC parameters and on reducing the number of the rules, in this paper.Genetic algorithms are used to search for an extracting rules with high performance. The ill-defined+ and conflicting rules are also eliminated through the search process. The rest of the work is divided as follows.In Section II-III, we give the review of fuzzy logic control, genetic algorithms and dc motor system. In Section IV-V-VI, we demonstrate the implementation of GAs for the design of the fuzzy-PI controller, and the effectiveness of the controller based on an intensive simulation and experimental study.

## II. FUZZY LOGIC CONTROLLER (FLC)

In the design of a fuzzy logic controller, the rules may have multiple input and output variables. However, it has been observed that the majority of the applications in the literature are of the multiple input, single output configuration [5]. For example, in the case of a PD-like fuzzy logic controller, a rule may appear as:

IF *error* is **PB** AND *change of error* is **SZ**
THEN *control* is **PS**

where *error*, *change of error* and *control* are linguistic variables; and PB (Positive Big), SZ (Small around Zero) and PS (Positive Small) are linguistic values taken by these variables. A rule in the rule base of the system may have the following generic form:

IF $X1$ is $\mu_{n1}(x_1)$ $X2$ is $\mu_{n1}(x_2)$ AND $\mu_{n1}(x1)$ AND .... $X\ m$ is $\mu_{nm}(x_m)$ THEN $Y$ is $\lambda_q(y)$

**$\mu_{n1}(x_1)$** : fuzzy sets as linguistic values of $X_J$ (e.g. PM (positive medium), NB (negative big), etc.)
**$\lambda_q(y)$** : fuzzy sets as linguistic values of output. $q$ =1, 2, 2 , $r$ and $r$ is the number of possible fuzzy sets for the consequent linguistic variable.
***Xj*** : $j$th linguistic variable of the antecedent part of a rule ( $j$=1, 2, 2 , $m$).
***Y*** : linguistic variable of the consequent part of a rule.

The deterministic output of the fuzzy system is computed by using the *product-operation* inference mechanism and the centroid method as follows:

$$y = \frac{\int \left[\bigcup_{i=1}^{p} w_i \lambda_q(y)\right] y dy}{\int \left[\bigcup_{i=1}^{p} w_i \lambda_q(y)\right] dy} \tag{2}$$

where $\cup$ stands for union (MAX operation).

## III. GENETIC ALGORITHMS

GAs are search algorithms based on the mechanics of natural selection and natural genetics. Unlike many classical optimization techniques, genetic algorithms do not rely on computing local derivatives to guide the search process. GAs generally consist of three fundamental operators: reproduction, crossover and mutation. The following explains procedure of genetic algorithms with the details [2]. It has become a common way to translate the parameters into binary bit or real number strings. Several parameters are coded into one long string

**a. Initial generation.** It always begins by randomly generating an initial population of *N* strings, each of length *m* bits. The population size *N* is a compromising factor.

**b. Fitness evaluation.** In the current generation, each of the strings is decoded to be its corresponding actual parameter. Then, these parameters are sent to a *fitness function* which yields a measure of the solution's quality, evaluated with some objective functions and assigned individually with fitness values.

**c. Reproduction.** Reproduction which is arranged with larger fitness values can produce accordingly with higher probabilities large number of their copies in the new generation. The most common method is used which is weighted roulette selection shown in the Fig. 1.

**d. Crossover**. Crossover is a process by which the systematic information exchange between two strings is implemented using probabilistic decisions. In a crossover process, two newly reproduced strings are chosen from the mating pool and arranged to exchange their corresponding portions of binary strings at a randomly selected partitioning position along them.

This process can combine better qualities among the preferred good strings. Crossover example.

Before crossover

A = 110111101 11011101011
B = 000100010 00010010010 (3)

After crossover

A = 110111101 *j* 00011010010
B = 000100010 *j* 11011101011 (4)

**e. Mutation.** Mutation is a process by which the chance for the GA to reach the optimal point is reinforced through just an occasional alteration of a value at a randomly selected bit position. Mutation may suddenly spoil the opportunity of the current appropriate generation, so, this process usually occurs with a small probability and is complementary to reproduction and crossover. In Eq. (4), Because of mutation the underline bit is changed from 0 to 1.

The GA runs iteratively (b-e) until it arrives at a predetermined ending condition. The speed of iteration depends not only on the population size *N* and the string length m but also selection of probabilities. Finally, the acceptable solution is obtained and decoded into its original pattern from the resulting binary strings.

## IV. GA BASED LEARNING FUZZY SYSTEMS

In the few years many different approaches have been presented taking the genetic algorithms (GAs) as a base of the learning process. GAs have demonstrated to be a tool powerful for automating the definition of the fuzzy control rule knowledge base [8]. These approaches called the general name of genetic fuzzy systems (GFSs)

We propose a GFS methodology based on three stages. The first stage is a fuzzy-PI rule genetic generating process based on a rule learning iterative approach, the second applying rules to process which best rule base in each generation, and the third one is ended learning process if best rule have desired fitness value. The three components of the learning process are developed formulating suitable genetic algorithms. This methodology as is shown in Fig 5.

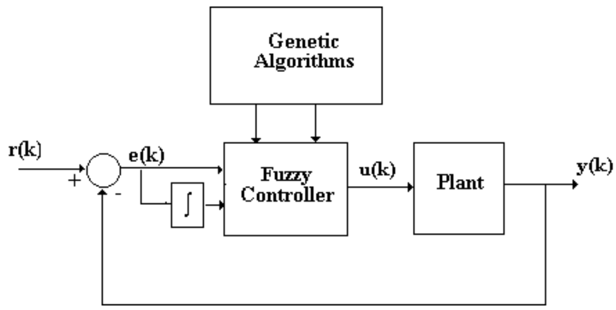



Fig. 5. Genetic based Learning Fuzzy-PI Control System

Genetic generating process is based on an iterative rule learning approach and consists of a fuzzy rule generating method together with an iterative covering method. The fuzzy rule generating method is developed by means of a real coding GA that codes as all fuzzy rule in each chromosome. The GA finds the best rule base in every run over the application to system according to the features included in its fitness function. The fitness function used in this work that reflects small steady-state errors, a short rise time, low oscillations and overshoots with a good stability is given by eqn 10.

$$fitness = \exp\left(-\frac{a}{T}\sum_{t=0}^{T} t*e^2 + t*(\Delta e)^2\right) \qquad (10)$$

where T is the the duration of the simulation in evaluating the design; a is a positive number used to scale the maximum fitness up or down, t is the time index in simulation, $e_i$ is the error between measured signal and the desired signal at simulation step t; $\Delta$e is the change of error. This statement of the fitness yields 0 through 1. Higher values corresponds to better controller performance.

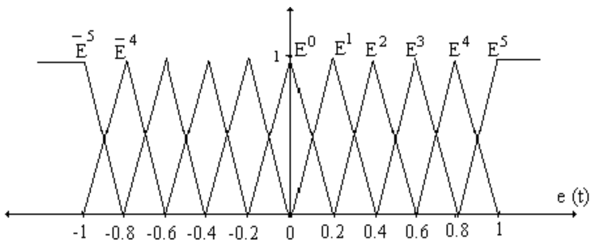



Fig 6. Used triangle membership function

Each gene in chromosome represents one membership function for control action; an example 1 is shown $E_1$ (Fig. 7).

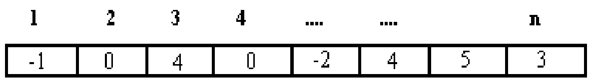



Fig 7. Chromosome with encoded Rule Base have Fuzzy sets

The fuzzy rule base focuses 11x11 possible control actions corresponding to values in input error and integral of error, therefore 121 bits are used in chromosome to form the look-up table, where a single bit represents each control action. This is illustrated in Table 1.

| | | **e** | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **-5** | **-4** | **-3** | **-2** | **-1** | **0** | **1** | **2** | **3** | **4** | **5** |
| **Δe** | **-5** | -1 | -1 | -1 | -1 | -1 | -1 | -0.8 | -0.6 | -0.4 | -0.2 | 0 |
| | **-4** | -1 | -1 | -1 | -1 | -1 | -0.8 | -0.6 | -0.4 | -0.2 | 0 | 0.2 |
| | **-3** | -1 | -1 | -1 | -1 | -0.8 | -0.6 | -0.4 | -0.2 | 0 | 0.2 | 0.4 |
| | **-2** | -1 | -1 | -1 | -0.8 | -0.6 | -0.4 | -0.2 | 0 | 0.2 | 0.4 | 0.6 |
| | **-1** | -1 | -1 | -0.8 | -0.6 | -0.4 | -0.2 | 0 | 0.2 | 0.4 | 0.6 | 0.8 |
| | **0** | -1 | -0.8 | -0.6 | -0.4 | -0.2 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 |
| | **1** | -0.8 | -0.6 | -0.4 | -0.2 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1 |
| | **2** | -0.6 | -0.4 | -0.2 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1 | 1 |
| | **3** | -0.4 | -0.2 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1 | 1 | 1 |
| | **4** | -0.2 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1 | 1 | 1 | 1 |
| | **5** | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 | 1 | 1 | 1 | 1 | 1 |

Table 1. Rule base for PD-like Fuzzy Controller

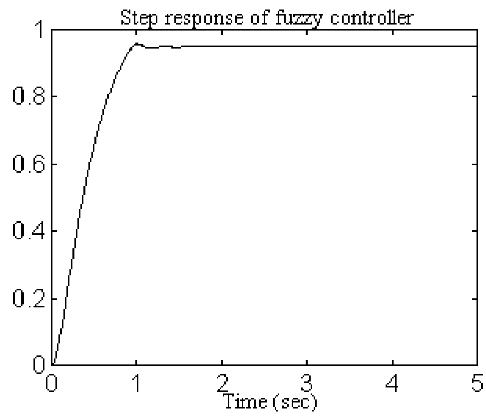



Fig 8. Step Response of PD-like Fuzzy Controller for used DC motor

## V. SIMULATION RESULTS

In this paper a dc motor from Sinano Co. is used and parameters is given by company are;

Moment of inertia ( J ) : 0.225 g-cm-s$^2$
Damping ratio (*B*) : 1.75 Kg-cm
Armature resistance (Ra) : 12.6 ohm
Armature inductance (La) : 9 mH
Ki : 1.5 Kg-cm/A
Kb : 15.4 mV/rpm

Under this parameters values motor transfer function is

$$G(s) = \frac{0.7407}{s^2 + 9.178\,s + 22.3} \tag{10}$$

For comparasion of the results and measure performance GFS a PD-like FLC with 11 rules was designed. The rule base of the system is given in Table 1. The fuzzy sets used in the controller are represented by triangle membership functions For dc motor system that we used this work step response of PD-like fuzzy controller is shown in Fig.8. It seems that rule base at Table 1 is not convenient for near zero steady state error.

We have taken results from designed genetic fuzzy system for dc motor are given in Table 2. and Fig 9, 10,11. Although learning process is executed for 10 generation take good results. During all generations have taken all best fitness values of chromosome that represents rule base is shown in Table 2. Best fitness in 9$^{th}$ generations have maximum fitness value.

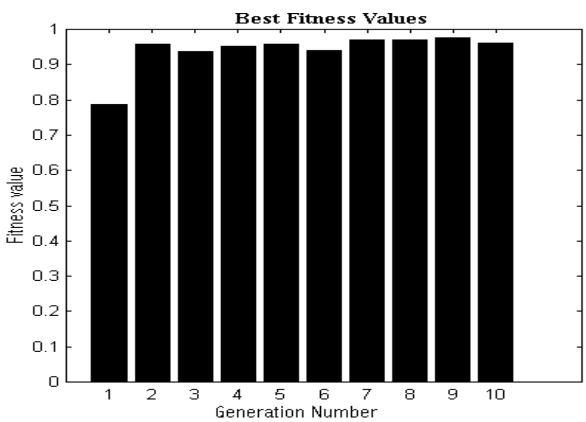



Table 2. Best fitness values in all generations

We designed GFS learning system is executed for 10 generations and taken step response outputs in fig 9.

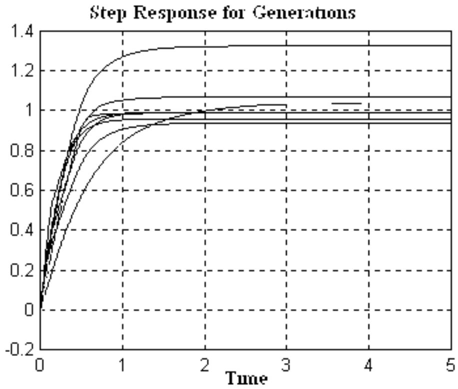



Fig 9. Step responses of dc motor system during generations for best fitness values

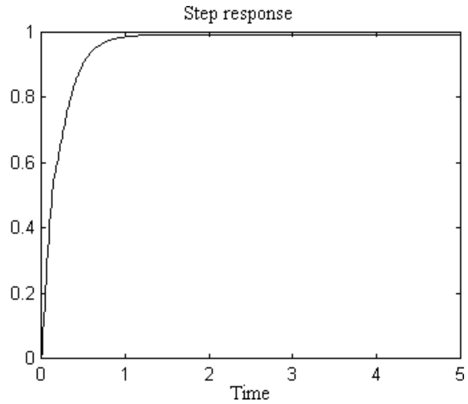



Fig 10. Step response of learned best rule base in the all generations for fuzzy-PI controller using GA

Step response of dc motor system by rule base have best Fitness is shown in Fig 10. i.e step response designed learning rule base of Fuzzy PI Controller for dc motor. When compared with PD-like Fuzzy Controller, it is seen that, the designed Fuzzy-PI Controller using genetic algorithms have better results. Surface of Rule base for designed Fuzzy-PI controller is shown in Fig 11.

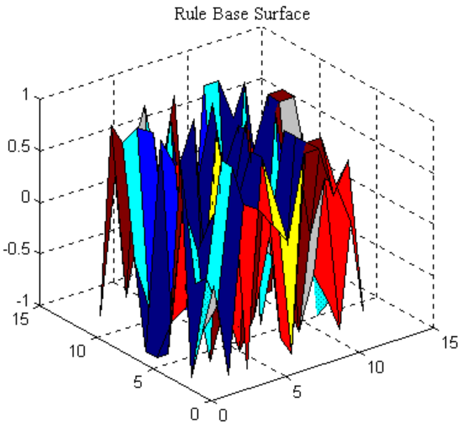



Fig 11. Learning Rule base surface for best fitness value

It seems that learned rule base disperses unregulerly in figure. Because of GA has random process, learned rule base is complex that contains most situations.

## VI. EXPERIMENTAL RESULTS

In this work, genetic algorithms which is a popular research technique are used for generating the DC generator fuzzy-PI control rule base. For this purpose a fuzzy control software runs on a PC is used to control the generator via the PC's parallel port. General diagram of this control system is shown at the figure 12.

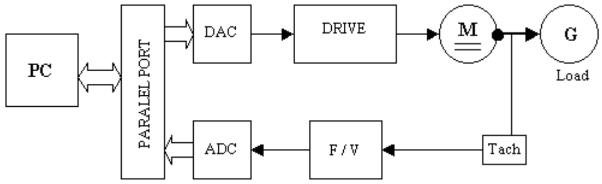
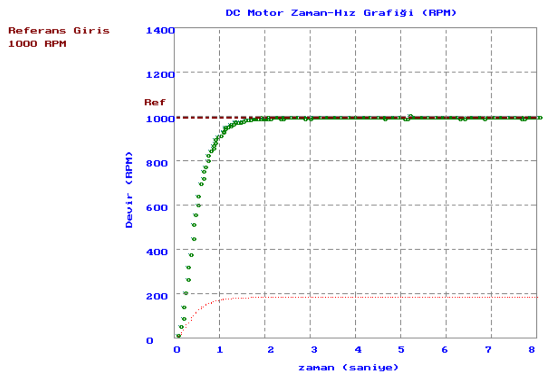
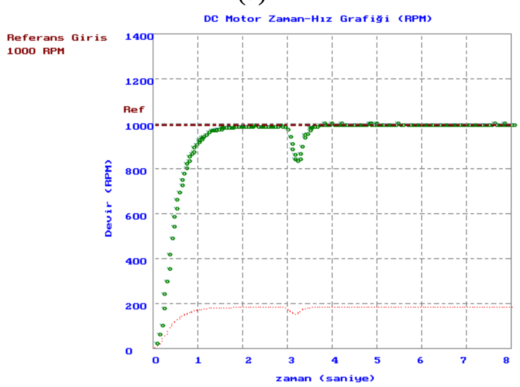



Fig 12. General diagram of targeted control system

Fuzzy PI controller using the rule base obtained from 30 generations, run of GFS algorithm designed in this work is applied to a unloaded DC generator system and the results are given in figures 13 a,b. A disturbing effect is given to this system to experiment system response. As it can be seen in figure 13-b, system settles to given reference point in a short time when the disturbing effect is removed.

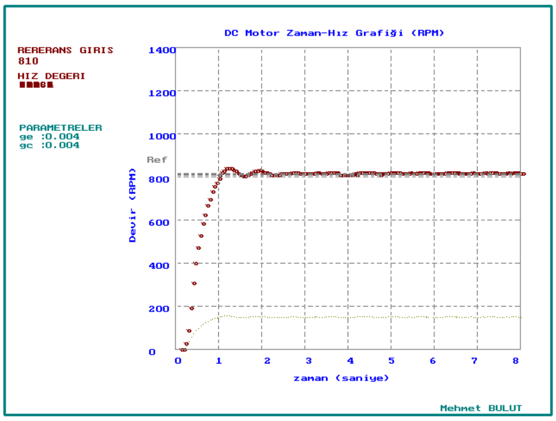



(a)

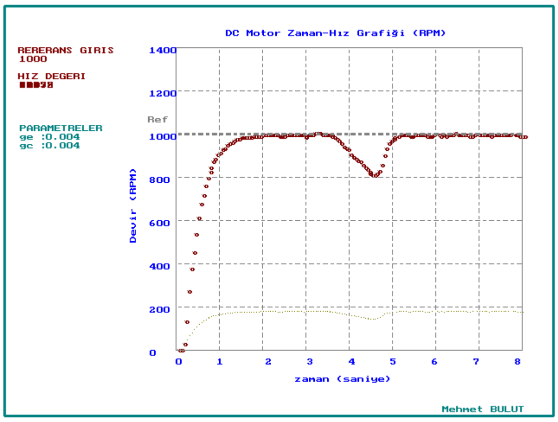



(b)

Fig 13. DC motor responses for 1000 rpm at reference value without load and with disturbance




(a)




(b)

Fig 14. DC motor responses for 810 rpm at reference value with load and for 1000 rpm with plus disturbance

Furthermore, this fuzzy PI controller is experimented with load given in various reference input points and obtained results are given in figure 14 a,b. When the loaded system is applied at 810 rpm (revolution per minute) and 1000 rpm reference values; System reaches reference point below Tr <1 sec overshooting time and settles to steady-state at Ts=1.5 sec.

Experimental results are shows that the implemented genetic fuzzy system algorithm is applicable to DC generator, Also it has flexible structure and high performance for fuzzy-PI controller. As a conclusion, the proposed approach in this study realizes a fuzzy-PI controller having low steady-state error and low settling time for a DC generator control.

## VII. CONCLUSION

Recently, this work has been studied in developing well-performing fuzzy rule-base without help of human expertise. Learning rule-base requires human experience with a long time. Genetic Fuzzy System (GFS) provides a method to automatize design of the knowledge base a direct fuzzy controller.

In this work, we are interested in automatically learning rule-base for a fuzzy logic based dc motor controller. In this article we provide an analysis of Genetic Fuzzy-PI System (GFS) for dc motor control system. It seems that the designed GFS is a good method for control of dc motor and gives well performance. At the work. Figures are drawed by using Pentium 120 MHz computer and Turbo C program. Hence, it provides an approach to computer aided design automation for dc motor control system.

As a result, this approach can provide a low-cost and robust means of design for the fuzzy-PI rule-based controller. From the results obtained, the proposed genetic algorithm model has been demonstrated its capabilities in term so flexibility and reliability by improving the rule base of the fuzzy logic controller.

## REFERENCES

1. Bimal K. Bose, “Expert system, Fuzzy Logic and Neyral Networks Applications in Power Electronics and Motion Control”, Proceedings of The IEEE, vol 82. No.8, 1303-1324, August, 1994.
2. M.A. Lee and H. Takagi, Integrating design stages of fuzzy systems using genetic algorithms, Proc. IEEE Internat. Conf. on Fuzzy Systems, 612 617, 1993.
3. C.Karr and E.Gentry, Fuzzy control of pH using genetic algorithms, IEEE Trans. Fuzzy Systems 1, 46-53, 1993.
4. D.Park, A. Kandel and G. Langholz, Genetic-based new fuzzy reasoning models with application to fuzzy control, IEEE Trans. Systems Man Cybernet. 24, 1994.
5. Timothy J. Ross, “ Fuzzy ogic with Engineering Applications”, McGraw-Hill Inc.,469-514, 1995
6. D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning (Addison-Wesley, Reading, MA, 1989.
7. Gopal K. Dubey, C. Rao Kasarabada, “Power Electronics and Drivers”, Tata McGraw-Hill Pub, 261-281, New Delhi, India, 1993.
8. F. Herrara, M.Lozano, J.L.Verdegay," Generating Fuzy Rules from Examples Using Genetic Algorithms", Department of Computer Science and Artificial Intelligence. University of Granada, Spain, 1995.