Using Motor Speed Profile and Genetic Algorithm to Optimize the Fuzzy Logic Controller for Controlling DC Servomotor

Trung Nguyen
Shibaura Institute of Technology
307 Fukasaku, Minuma-ku, Saitama-shi,
Saitama 337-8570, Japan

Takashi Komeda
Shibaura Institute of Technology
307 Fukasaku, Minuma-ku, Saitama-shi,
Saitama 337-8570, Japan

ABSTRACT

The paper describes a new proposed algorithm to automatically tune a Fuzzy Logic Controller by using motor Speed profile and Genetic Algorithm (FLCSGA algorithm) in controlling a DC Servo Motor. In the new method, the tuning process of the Fuzzy Logic Controller (FLC) is divided into two consecutive stages which are tuning rule base and tuning Membership Functions (MFs). The tuning rule base (Fuzzy rules) is based on the motor speed profiles, and the Genetic Algorithm (GA) is used to optimize MFs. In addition, a new encoding method was suggested for the GA that reduces remarkably optimization time for the system. This is a very important thing, especially with the real experiments for optimizing system such as motors control. The experiments on a Maxon motor RE 35 273752 showed that after using FLCSGA algorithm, an optimized FLC was generated. This FLC that had better performances compared to using the conventional proportional-integral-derivative controller (PID controller) in term of settling time, rise time. Besides, the required generations and the amount of chromosomes in population of GA are reduced significantly compared to some previous studies. It means the convergence time is very fast.

General Terms:

Optimization, Algorithm, Control.

Keywords:

DC Servo Motor, Genetic Algorithm, Fuzzy Logic Controller, Motor Speed Profile.

1. INTRODUCTION

There are two main kinds of self-commutated electrical motor, that are used in most normal appliances, are AC and DC motor. DC motors have better starting moment than AC motors. In addition, DC motors have better performance than AC motors on the traction equipments. DC Servomotor which is being used in many applications in industrial and robotic machines is one type of DC motor. In the DC Servomotor, the basic continuous feedback control is PID controller. From the view of controlling, nowadays PID controllers have been used very popular because of their simplicity in structure, robust performance in a wide range of operating conditions, and easy implementation [11]. Nevertheless, The PID controllers are not enough adaptation [17] especially when the load is changed or there is noise in the system. In these cases, we should re-tune, re-design or add more filter for the PID controller to be satisfied with the new conditions.

To overcome these weak points of the PID controller, in 1965 Lotfi Zadeh first introduced the Fuzzy Logic tool. This is a mathematical method to deal with imprecise data and vague statements by providing a mechanism to present the linguistic constructs such as "many", "low", "high", etc. [25]. It uses probability theory to measure whether these events will happen or not. Besides, by imitating the rule-of-thumb thought of human beings about the system, the FLC can be replaced for the controller that is built based on precise mathematical computation about the system [20]. Until now, there are many researches have implemented to evaluate the good points of the FLC and also compare it to the PID controller [20], [14] and [16]. These researches showed some advantages of using the FLC as follows: the FLC does not need mathematical models of the systems; better possibilities compared to the PID controller in noise rejection, flexibility and sensitiveness to inertia variation.

In addition to the above advantages, the FLC also has some disadvantages: because the FLC is built based on the tedious trial and error process, the accuracy and building time depend on the operator's experience on the system. And mostly it is a time-consuming process. Refs. [24],[19], [4] and [15] represent methods of using the GAs for auto-tuning the FLC to overcome the weak points. GAs are proven to provide robust search in complex spaces [9]. They are numerical search methods that mimic the process of natural selection. Besides, when implementing optimization or auto-tuning the FLC, there are four factors that can be tuned: input scale factor, output scale factor, Membership Functions (MFs) and Control rules (rule base). Rahul Malhotra et al. in [19] employed the GA for tuning MFs and rule base of the FLC for speed control of DC motor. The simultaneous optimization of the FLC, by using the GA application, showed encouraging results [18]. In addition to using the GA to optimize the FLC, there are also other algorithms can be applied: Danilo Pelusi employed the GA [7] and then combined the GA with Neural Networks [8] to optimize the FLCs that are used in solving the problem of electrical signal frequency driving for signals acquisition experiments, second order control system, respectively. Bouallegue [21] used particle swarm optimization approach to tune PID-type FLC structure and successfully applied on an electrical

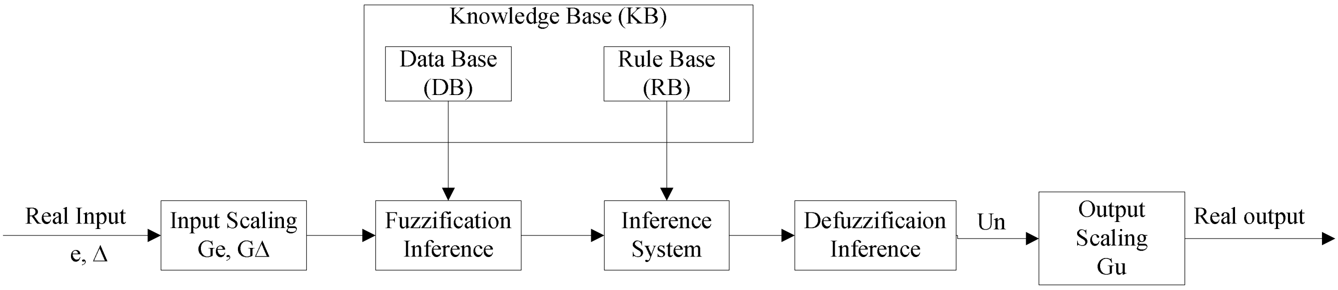


Fig. 1: Structure of a Mamdani Fuzzy Logic rule base system.

DC drive speed control. These researches got good results. However, most of them did not take tuned time into account. In fact, the number of generation and chromosomes were quite large. As a result, the required time for tuning was long. Not surprisingly, the optimization time for a complex problem maybe last for weeks. This is acceptable if the optimal process is simulation only. However, If this is a parallel process of optimizing and experimenting then it could consume a lot of time and money. As a result, it is necessary to improve the convergence of optimization process. Moreover, when implementing DC Servo motor sizing, researchers often used some speed profiles [1] such as triangular, trapezoidal profile. Therefore, tuning DC Servomotor based on its speed profile coordinating with using the GA seem to be a new reasonable approach to optimize the FLC.

This paper proposes a new method to auto-tune the FLC used for control DC Servomotor. This algorithm bases on the motor's speed profile and the GA to tune Fuzzy rule base and Membership Functions, respectively.

The remainder of this paper is organized as follows: the basic concepts of FLC and GA are described in the section 2. Section 3 explains the proposed FLCSGA. Sect.4 shows the experiment on a DC Servomotor, results and discussion. Finally, Sect.5 is some conclusions.

2. BACKGROUND KNOWLEDGE

2.1 Fuzzy Logic Controller

The FLC system uses Fuzzy logic rules to establish a control mechanism to approximate expert perception and judgment under given conditions. The system is also known as Fuzzy inference system or approximate reasoning system or expert system. The structure of a FLC can be described in the **Fig.1**.

i) ***Input Scaling or normalization:*** the current physical values of state variables i.e., which are often error (e) and change of error ($\triangle$), are mapped into normalized values by Scaling factor G_e and $G_\triangle$.

ii)***Fuzzification:*** each crisp current process state values (e and $\triangle$) is converted to a fuzzy set to make it compatible with fuzzy set of process variable in the rule-antecedent.

iii)***Inferencing (inference engine):*** The fuzzified input values are transferred to the inference engine to evaluate the control rules stores in rule base. And the result of this evaluation is a single fuzzy set or several fuzzy sets. Generally, logic rules which are the main facts to compose inference engine use AND (taking minimum value) or OR (taking maximum value) operators.

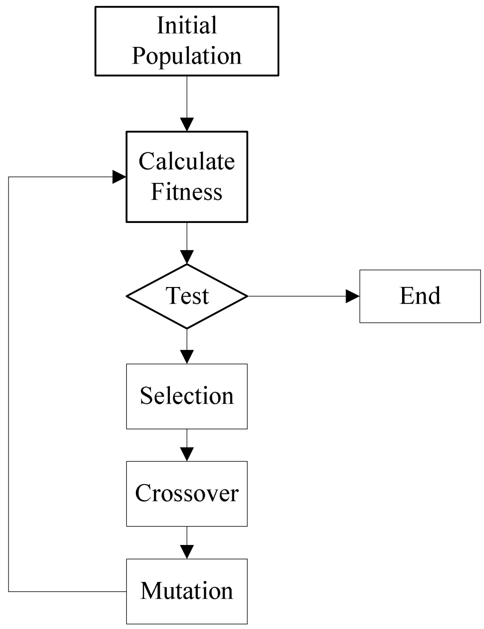


Fig. 2: The flow chart of genetic algorithm.

iv)***Deffuzzication:*** defuzzification converts inference results of all active logic rules into a single crisp value in normalized domain. Defuzzification often use the maximum membership method, center of average method or center of gravity method.

v)***Output scaling or denormalization:*** the defuzzified normalized control output (u_N) is mapped into physical value (u) by the output scaling factor G_u.

2.2 Genetic Algorithms basic concept

GAs are search algorithms which use principles inspired by natural genetics to evolve solutions to problems [5], [6]. GAs include three major operators: selection, crossover, and mutation, in addition to four control parameters: population size, selection, crossover, and mutation rate [19]. The idea is to maintain a population of chromosomes (representing candidate solutions to the concrete problem being solved) that evolves over time through a process of competition and controller variation. The flow chart of the GA as in **Fig.2** is described bellow.

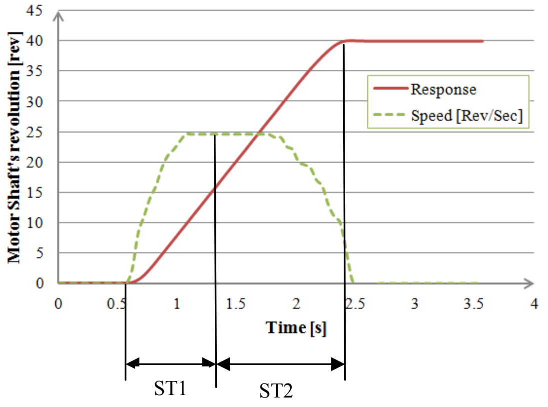


Fig. 3: Assumption step response.

Table 1. : Initial rule base.

u		Error (e)						
		NB	NM	NS	ZE	PS	PM	PB
Change of Error (ce)	NB	NB	NB	NB	NM	NS	NS	ZE
	NM	NB	NM	NM	NM	NS	ZE	ZE
	NS	NB	NM	NS	NS	ZE	PS	PM
	ZE	NB	NM	NS	ZE	PS	PM	PB
	PS	NM	NS	ZE	PS	PS	PM	PB
	PM	NS	ZE	PS	PM	PM	PM	PB
	PB	ZE	PS	PS	PM	PB	PB	PB

Table 2. : Coded value of linguistic value.

Linguistic variables	NB	NM	NS	ZE	PS	PM	PB
Coded value	0	1	2	3	4	5	6

[**Initial Population**]. GA starts with generating a random population of n chromosomes (suitable solutions for the problem). The population is composed by binary or real coded string chromosomes.

[**Calculate Fitness**]. Evaluate the fitness value of each chromosome in the population.

[**Test**]. If the end condition is satisfied, Stop, and return the best solution in the current population.

[**New population**]. If the condition is not satisfied, a population is generated by following the steps below until a new population is complete.

(1) [***Selection***]. Selects two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected).

(2) [***Crossover***]. With a crossover probability, cross over the parents to generate new offspring. If no crossover was performed, the offspring is copied exactly the parents.

(3) [***Mutation***]. with a mutation probability, mutate new offspring at each locus.

3. A NEW DC MOTOR TUNING ALGORITHM - FLCSGA ALGORITHM

The algorithm used two consecutive stages to optimize the FLC: the first one was used to tune the Fuzzy rules based on Motor's speed profile and then GA with the new coding method was applied to optimize the MF(s) of the FLC.

For the Mamdani fuzzy logic controller, the algorithm used AND operator for inference engine and the centre of gravity defuzzification formula :

$$u = \frac{\sum_{i=1}^{m} .m_k.\mu_{Bk}(u)}{\sum_{i=1}^{m} .\mu_{Bk}(u)} \qquad (1)$$

Where: m_k the centre of fuzzy output set, B_k, $k = 1, 2...m$ is the fuzzy output variable. Using the maximum inference engine:

$$\mu_{Bk} = max(\mu_{R1}, \mu_{R2}, ..., \mu_{Rr}) \qquad (2)$$

Where: r is the total of candidate rules, Ri is membership function of rule Ri.

3.1 Tuning Fuzzy rules using DC Servomotor Speed profile

In order to design an optimized FLC, the initial rules were established in **Table 1** in according to with [22]. The linguistic variables were defined as: **NB**: Negative Big, **NM**: Negative Medium, **NS**: Negative Small, **ZE**: Zero, **PS**: Positive Small, **PM**: Positive Medium, **PB**: Positive Big.

The step response was supposed to be divided into 2 stages as in **Fig.3**: the first stage was increasing stage (ST1) of velocity and the other was decreasing stage of velocity (ST2)[10]. In the first stage, the speed of the motor increased to the maximum speed that depends on the load, bearing friction etc. and then stabilized with this speed until the end of this stage. The terminative point of this stage was at around 40% of the reference value. The second stage continued to stabilize the motor's maximum speed until the point of around 60% reference value and then reduced to get the stable speed of zero. The algorithm is summarized as:

```
Begin
    GetMinimumVoltage();
    CheckConditionForST1;
    CheckConditionForST2;
End.
```

The missions of *GetMinimumVoltage()* function is to get the minimum requirement voltage of the motor to achieve the maximum speed and to count the minimal acceleration time of the motor. *CheckConditionForST1* implements to control motor and check

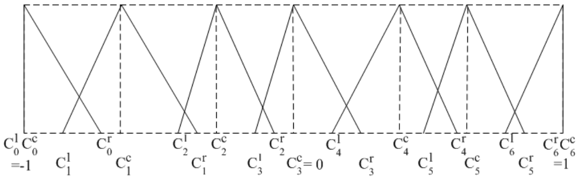


Fig. 4: Fuzzy sets on a premise.

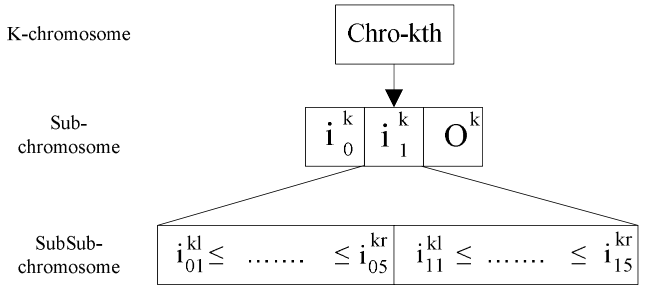


Fig. 5: Encoding method for Membership functions.

whether the condition of ST1 is sufficient or not to do next step of *CheckConditionForST2*. In each of these checking processes, there are number of movements of rule base depending on the responses of motor.

3.2 Tuning Membership Function using GA

When implementing a simple genetic algorithm a number of issues must be handled [12], these include the coding procedure, the selection , crossover, mutation.

3.2.1 A new encoding method for membership functions. The membership functions show the fuzziess degree in a premise. However, the main feature of membership function is their overlapping capability rather than their shapes precision [18]. Consider a triangle fuzzy member and its parameters are C^{kr}, C^{kc} and C^{kl}, respectively, represent the coordinates of right anchor, cortex and left anchor of k_{th} linguistic degree. In addition, as mention above, each universe of discourse was divided into 7 linguistic variables. Therefore, the algorithm needed 21 parameters to represent each Input or Output premise. Furthermore, without loss of generality, we assumed that the first and the last degrees of fuzzy number were left- and right- skewed triangles, respectively. It is noticed that the first, last and center anchor of each universe of discourse are -1,1 and 0 respectively and they need not to be tuned. As a result, to represent each premise the algorithm only needed 16 parameters. Additional constraint are imposed: $C^{kr} \geq C^{kc} \geq C^{kl}$, and the condition to prevent overlapped membership functions [2] a typical premise of input and output is expressed as **Fig.4** or as in (3).

$$\begin{aligned} &C_0^{kl} = C_0^{kc} = -1 \leq C_1^{kl} < C_0^{kr} \leq C_1^{kc} \leq C_2^{kl} < ... \\ &\leq C_3^{kc} = 0 \leq ... \leq C_6^{kl} < C_5^{kr} \leq C_6^{kr} = C_6^{kc} = 1 \end{aligned} \tag{3}$$

From the condition (3), we suggested a simple way to achieve the of parameters' values at initial time to generate the first generation is that in each current range of universe of discourse, [0,1] for example, 8 values was randomly selected and rearranged following the increasing trend. These increasing values after that were assigned to the parameters form C_1^{kl} to C_5^{kr}. The process of rearrangement was also re-performed after each generation. In addition, in the FLC to control DC Servomotor, there were 2 inputs and 1 output. Therefore, to tune the membership functions, it is necessary to had to encode 2 input premises and 1 output premise in each chromosome as in **Fig.5**. In the other side, each chromosome was further divided into sub- and subsub-chromosome. This method, which is carried out after first stage, reduced the computational time, time to converge. This is clearer when comparing the results with the encoding method suggest in Ref. [13], [3]

The **Fig.5** describes a $k-th$ chromosome structure. Each $k-th$ chromosome includes 3 Sub-Chormosomes which are 2 inputs Sub-Chromosome i_0^k, i_1^k, 1 output Sub-Chromosome o^k. For each Sub-Chromosome, it is partitioned into 2 SubSub-Choromosome that are negative portion: $i_{01}^{kl}...i_{05}^{kr}$ and positive portion $i_{11}^{kl}...i_{15}^{kr}$.

3.2.2 GA operators. Crossover and Mutation: The crossover technique used in binary genetic algorithm is a simple crossover technique [23]. The part of this reproduction mechanism was governed by an initiating probability p_1. The crossover process was implemented in each SubSub-Chromosome. In order to diversify the genotype strings, mutation operator was performed in each SubSub-Chromosome with mutation probability.

Selection: The capacity to survive of each individual was evaluated through the cost function. The Integral of Absolute Error (IAE) (4) is the cost function which was used to measure the system performance since it is known to give better all round performance indicator of a control system response [24].

$$IAE = \int_0^\infty |e(t)dt| \tag{4}$$

To evaluate the performance of the DC Servomotor, the algorithm compared the minimum value of the cost functions to get the minimal value.

4. EXPERIMENTS AND RESULTS

4.1 Experimental setup and method

In order to analyze and verify the FLCSGA algorithm, the FLCSGA was used to optimize the FLC of a DC Servomotor. The experimental model is shown in **Fig.6**.

In the **Fig.6**, the PCI 6024 is a PCI pulse counter board to measure the number of revolutions of motor shaft. And the PCI 3523A is a 12-bit analog Input/Output board that is used to perform AD/DA conversion. The dash or dash-dot lines are the power transmission form energy sources to the Encoder HEAL 5540 and the motor controller. The motor RE 35-273752, which is attached a 4.8 :1 reduction ratio Gearhead, is controlled by a Maxon Motor controller of AD 50/5. Two power supplies are PMC 18-3 which are set up based on the specification of the encoder, motor controller and motor. The continuous lines are the signal lines to get the current revolution of the motor shaft and to control the motor.

Firstly, the universe of discourses were split into 6 equal fuzzy sets and run the first stage of FLCSGA. The experiments were implemented in different reference values from 20 to 40 which represent the different fuzzy sets. Next, the second stage of optimization using FLCSGA was done to optimize the membership functions of the FLC.

4.2 Experiment results by FLCSGA

After running the first stage, the speed responses of the motor were collected as in **Fig.7** for the different reference values as well as

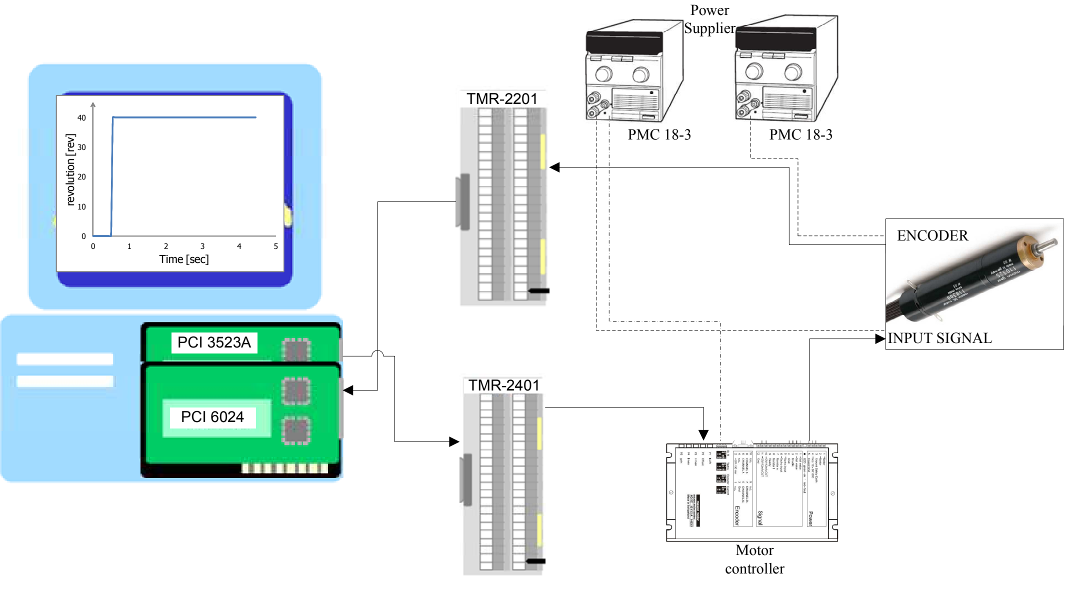


Fig. 6: Experiment model to control a DC Servomotor.

the optimized rule base as in **Table 3**. The maximum speed of the motor in this case is the no load speed:

$$n_0 = \frac{7070}{4,8.60} = 24.5[rev/sec] \tag{5}$$

where $7070[rev/min]$ is no load speed of motor RE 35-273752. After completing the first stage, the second optimal stage was performed with a population size is 16 chromosomes, number of generation is 12, gene length is 8 bits, selection rate iss 0.5, crossover rate is 0.5 and mutation rate is 0.005. The convergence speed of the proposed the FLCSGA algorithm was very fast as in **Fig.9**. More specifically, the **Table 4** is the convergence speed comparison between the proposed algorithm with some other algorithms. The **Fig.8(a)** and **Fig.8(b)** are respectively the Fuzzy rule base surface at the first stage and after second stage of using FLSGA algorithm, respectively. **Fig.10** is optimal membership functions.
Next, the **Fig.11** and **Table 5** are step response comparison between the FLCGA and PID controller. The proposed FLCSGA has better settling time, rise time compared to the PID controller.

4.3 Discussion for improving the FLCSGA algorithm

It is necessary to improve the accuracy of tunning rule base by properly following the motor speed's profile. In the current research, the 40% and 60% reference value were used to divide the response process into 2 stages. However, it may be better if the speed response was divided and tuned into 3 stages which include acceleration, constant speed and deceleration stage. To be able to do this, it is necessary to determine the times that are the end of increasing speed and starting point to decrease motor's speed.

Table 3. : Optimized rule base.

u		**Error (e)**						
		NB	NM	NS	ZE	PS	PM	PB
Change of Error (ce)	NB	NB	NB	NB	**NB**	**NM**	**PM**	**PM**
	NM	NB	NM	NM	NM	NS	**ZE**	**ZE**
	NS	NB	NM	NS	NS	ZE	PS	**PS**
	ZE	**NB**	NM	**ZE**	ZE	**NS**	PM	PB
	PS	**NS**	NS	**PS**	PS	PS	PM	PB
	PM	NS	**NB**	**ZE**	PM	PM	PM	PB
	PB	**NS**	**NS**	**ZE**	**PB**	PB	PB	PB

5. CONCLUSION

In this paper, a new FLCSGA algorithm was proposed. The algorithm is based on the speed profile of the motor and the GA with a new coding method, to tune the FLC for controlling DC Servomotor. After that, the algorithm was verified by tuning the FLC to control a DC Maxon Servomotor.
We confirmed that using FLCSGA reduces the computation cost and improve the convergence speed to the optimal value. The response by using optimized FLC also showed better settling time, rise time compared to using the PID controller. In the future, after optimizing the algorithm, the FLCSGA should be used to optimize the real controller in systems. These systems are complex, difficult to establish the mathematical models or are affected by eternal noise, disturbance.

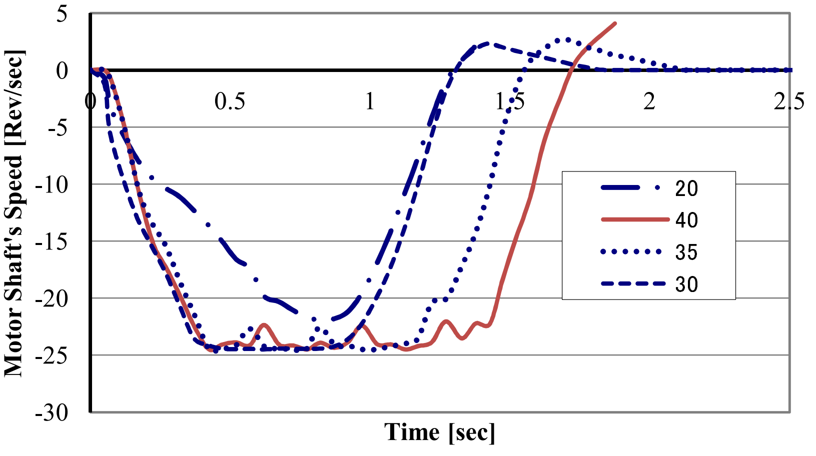
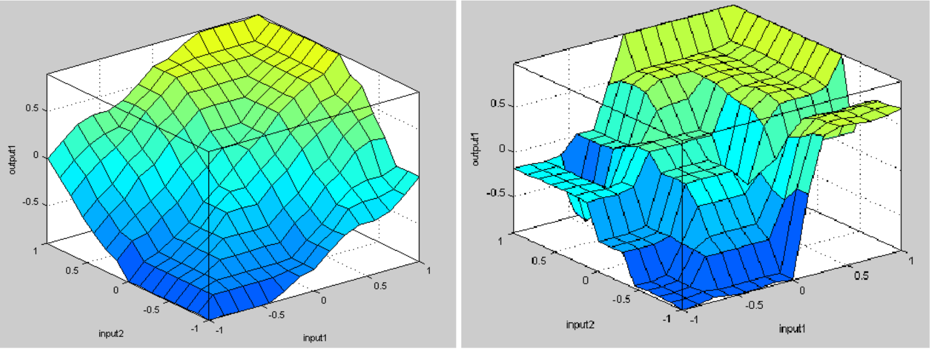


Fig. 7: Speed response after the 1st stage.



a) b)

Fig. 8: Fuzzy rule base surfaces at the first and after the second stage generated by FLCSGA.

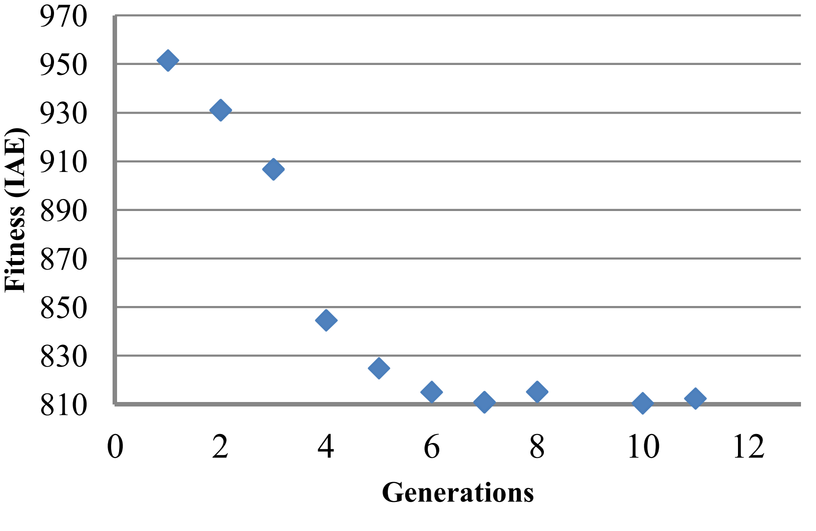


Fig. 9: Convergence properties of the proposed FLCSGA algorithm.

Table 4. : Comparison the convergence speed between the proposed algorithm with some other algorithms.

	Algorithm				
	GA [24]	GA [4]	GA [15]	PSO [21]	**FLCSGA**
Generations	300	100	100	30	**12**
Chromosomes	30	72	40		**16**

Table 5. : Comparison between PID and FLCSGA.

	PID	**FLCSGA**
Settling time [s]	2.8	2.267
Rise time [s]	1.85	1.767
Overshoot	0.0021	0.00216

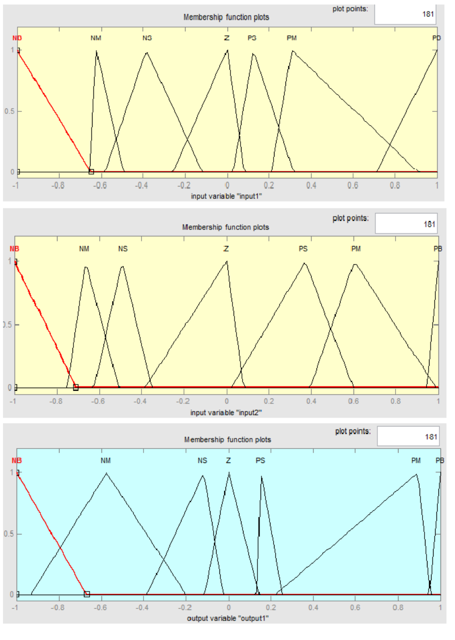


Fig. 10: The optimal membership functions.

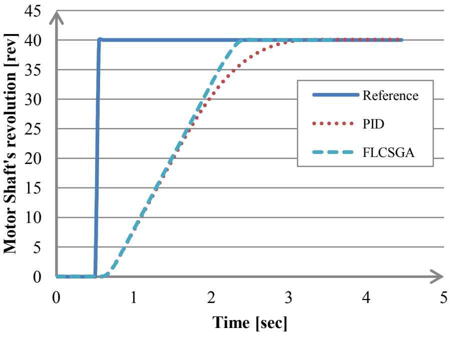


Fig. 11: Step response of PID and FLCSGA.

6. REFERENCES

[1] Copperhill Technologies Corp. "A Comprehensible Guide to Servo Motor Sizing," , May 2007.

[2] C.Chen, T.Hong, and V.S.Tseng. A comparison of different fitness functions for extracting membership functions used in fuzzy data mining. In *IEEE Symposium on foundations of computational intelligence*, pages 550–555, 2007.

[3] Y. Chiou and L.Lan. Genetic fuzzy logic controller: an iterative evolution algorithm with new encoding method. *Fuzzy Sets and Systems*, 152(3):617–635, 2005.

[4] C.Liu, B.Li, and X.Yang. Fuzzy logic controller design based on genetic algorithm for dc motor. In *IEEE Intl. Conf. On Electronics, Communications and Control*, pages 2662–2665, 2011.

[5] D.E.Goldberg. *Genetic Algorithms in search, optimization, and machine learning*. Addison-Wesley, 1989.

[6] D.E.Goldberg. *The Design of competent Genetic Algorithms: Steps towards a computational theory of innovation*. Kluwer Academic, 2002.

[7] D.Pelusi. Genetic-neuro-fuzzy controllers for second order control systems. In *5th European Symposium on Computer Modeling and Simulation*, pages 12–17, 2011.

[8] D.Pelusi. Improving settling and rise times of controllers via intelligent algorithms. In *14th Inter. Conf. On Modelling and Simulation*, pages 187–192, 2012.

[9] F.Herrera, M. Lozano, and J. L. Verdegay. Tuning fuzzy logic controllers by genetic algorithms. *Inte. Jour. Of Approximate Reasoning*, pages 299–315, 2013.

[10] J.Jamaludin, N.A. Rahim, and W.P. Hew. Development of a self-tuning fuzzy logic controller for intelligent control of elevator systems. *Engineering Applications of Artificial Intelligence*, 22(8):1167–1178, 2009.

[11] J.Zhang, N.Wang, and S.Wang. A developed method of tuning pid controllers with fuzzy rules for integrating process. In *Proc.IEEE Of American Control Conference*, pages 1109 – 1114, 2004.

[12] K.Belarbi, F. Titel, W. Bourebia, and K. Benmahammed. Design of mamdani fuzzy logic controllers with rule base minimisation using genetic algorithm. *Engineering Applications of Artificial Intelligence*, 17(7):875880, 2005.

[13] L.Baron. Fuzzy decision support system knowledge base generation using a genetic algorithm. *Inter. Journal of approximate reasoning*, 24(2-3):125–148, 2001.

[14] M.F.Rahmat and M.M.Ghazaly. Performance comparison between pid and fuzzy logic controller in position control system of dc servomotor. *Technology*, pages 1–17, 2006.

[15] N.Ozturk and E.Celik. Speed control of permanent magnet synchronous motors using fuzzy controller based on genetic algorithms. *Int. Journal of Electrical Power and Energy Systems*, 43(1):889–898, 2012.

[16] O.Gundogdu and K.Erenturk. Fuzzy control of a dc motor driven four-bar mechanism. *Mechatronics Journal*, 15:423–438, 2003.

[17] O.Wahyunggoro and N.B.Saad. Development of fuzzy-logic-based self tuning pi controller for servomotor. In *IEEE Intl. Conf. On Control, Automation, Robotics and Vision*, pages 1545–1550, 2004.

[18] P.C.Shill, K.K.Pal, M.F.Amin, and K.Murase. Genetic algorithm based fully automated and adaptive fuzzy logic controller. In *IEEE Intl. Conf. On Fuzzy Systems*, pages 1572–1579, 2011.

[19] R.Malhotra, N.Singh, and Y.Singh. Design of embedded hybrid fuzzy-ga control strategy for speed control of dc motor: A servo control case study. *Int. Journal of Computer Applications*, 6(5):37–46, 2010.

[20] R.Malhotra, T.Kaur, and G.S.Deol. Dc motor control using fuzzy logic controller. *Int. Journal. of advanced engineering sciences and technologies*, 8(2):291–296, 2011.

[21] S.Boualegue, J. Haggge, M. Ayadi, and M. Benrejeb. Pid-type fuzzy logic controller tuning based on particle swarm optimization. *Engineering Applications of Artificial Intelligence*, 25(3):483–493, 2013.

[22] S.Chopra and R.Mitra. Fuzzy controller: choosing an appropriate and smallest rule set. *Inter. Journal of computational cognition*, 3(4):73–79, 2005.

[23] S.G.Gilbert. Uniform crossover in genetic algorithms. In *Proc. Of the 3rd Inter. Conf. Genetic Algorithms*, pages 2–9, 1989.

[24] S.Khan and et al. Design and implementation of an optimal fuzzy logic controller using genetic algorithm. *Journal of Computer Siences*, 4(10):799–806, 2008.

[25] S.N.Sivanandam, S. Sumathi, and S. N. Deepa. *Introduction to Fuzzy Logic using MATLAB*. Springer, 2007.

APPENDIX: DC Servomotor RE 35-273752 Parameter

Table 6. : Data of DC Servomotor used in experiments

Motor Data	Unit	Value
Nominal Voltage	V	15
No Load Speed	rpm	7070
No Load Current	mA	245
Max.efficiency	%	81