THE CONTROL OF A DC SERVO MOTOR BY USING FUZZY GAIN SCHEDULING CONTROLLER

K. Burak DALCI, Halit PASTACI, Kayhan GÜLEZ
Yıldız Technical University
Electrical & Electronics Faculty
Electrical Engineering Department
80750 Beşiktaş-İstanbul, TURKEY
dalci@yildiz.edu.tr , hpastaci@yildiz.edu.tr , gulez@yildiz.edu.tr

ABSTRACT

This paper describes the development of a fuzzy gain scheduling scheme of PI controllers for motor speed control. Fuzzy rules and reasoning are utilized on-line to determine the controller parameters based on the error signal and its first difference. Application results demonstrate that better control performance can be achieved in comparison with PI controllers.

1. INTRODUCTION

The best known controllers used in industrial control processes are proportional-integral (PI) controllers, because of their simple structure and robust performance in a wide range of operating conditions. The design of such a controller requires specification of two parameters: proportional gain and integral gain. So far, great effort has been devoted to develop methods to reduce the time spent on optimising the choice of controller parameters. The PI controllers in the literature can be divided into two main categories. In the first category, the controller parameters are fixed during control process after they have been tuned or chosen optimal way. The PI controllers of this category are simple, but can not always effectively control systems with changing parameters and may need frequent on-line retuning. The controllers of second category have a structure similar to PI controllers but their parameters are adapted on-line based on parameter estimation, which requires certain knowledge of the process. Such controllers are called adaptive PI controllers in order to differantiate them from those of this category [3].

The application of knowledge based systems in process control is growing, especially in the field of fuzzy logic control. In fuzzy control, linguistic descriptions of human expertise in controlling a process are represented as fuzzy rules or relations. This knowledge base is used by an inference mechanism, in conjunction with some knowledge of the states of the process (for example measured responce variables) in order to determine control actions. Although they do not have an appearent structure of PI controllers, fuzzy logic controllers may be considered non-linear PI controllers whose parameters can be determined on-line based on the error signal and its first derive [1], [2].

In the paper a rule-based scheme for gain scheduling of PI controllers is proposed for motor speed control. The new scheme utilises fuzzy rules and reasoning the determine the controllers parameters and the PI control generates the PWM output as the control signal. It is demonstrated in this paper that human expertise on PI gain scheduling can be represented in fuzzy rules. Furthermore, better control performance can be expected in the proposed method than that of the PI controllers with constant parameters.

2. PI CONTROLLER

The transfer function of a PI controller has the following form,

$$G_c(s) = K_p + \frac{K_i}{s} \qquad (1)$$

where K_p, K_i are proportional and integral gains, respectively. The discrete-time equivalent expression for PI control used in the paper is given as,

$$u(k) = K_p e(k) + K_i T_s \sum_{i}^{n} e(i) \qquad (2)$$

where u(k) is the control signal, e(k) is the error between the reference speed and the process speed and T_s is the sampling period of the controller.

3. FUZZY GAIN SCHEDULING

Figure 1 shows the PI control system with a fuzzy gain scheduler. The used method is to exploit fuzzy rules and reasoning to handle controller parameters. In the method controller parameters K_P and K_I are limited K_{pmin} , K_{pmax} and K_{imin} , K_{imax} respectively. Appropriate ranges are determined experimentally. K_P and K_I are normalized into the range between zero and one by the following linear transformation to use in fuzzy operations.

$$K'_p = (K_p - K_{p\min})/(K_{p\max} - K_{p\min}) \quad (3)$$

$$K'_i = (K_i - K_{i\min})/(K_{i\max} - K_{i\min}) \quad (4)$$

In the proposed scheme, PI parameters are determined based on the current error e(k) and its first difference Δ e(k). The parameters K_P and K_I are determined by a set of fuzzy rules of the form,

If e(k) is A_i and Δ e(k) is B_i,
then $K_p{}'$ is C_i, $K_i{}'$ is D_i i = 1,2,....,m. (5)

where A_i, B_i, C_i and D_i are fuzzy sets on the corresponding supporting sets. The membership functions of these fuzzy sets for e(k) and Δ e(k) are shown in figure 2. In figure 2, N represents negative, P positive and Z approximately zero, M medium, B big. Thus, NM represents negative-medium, PB represents positive-big and so on.
The fuzzy sets C_i and D_i can be described big (B) or small (S) for $K_p{}'$ and is characterized by the membership function which is shown in figure 3. The fuzzy sets can be described as little (L), small (S), normal (N), medium (M) and big (B) for $K_i{}'$ and is characterized by the membership function which is shown in figure 4. K_p is changing 4 to 23 ranges and K_i 15 to 40 ranges. We use this ranges in assembler language software in the microcontroller.

The fuzzy rules is obtained from the operator's expertise. In the system, we derive the rules experimentally based on the step response of the process. Desired time response can be shown in figure 5.

At the beginning, around a_1, we need a big control signal in order to achieve a fast rise time. To produce a big control signal, the PI controller should have a large proportional and integral gain. Thus the proportional gain $K_p{}'$ can be represented by a fuzzy set Big, and the integrale gain $K_i{}'$ by a fuzzy set PB. Therefore, the rule around a_1 is written as,

If e(k) is PB and Δ e(k) is Z, then $K_p{}'$ is Big, $K_i{}'$ is PB

around point b_1 in figure 5. We expect a small signal to avoid a large overshoot. That is, PI controller should have a small proportional and integral gain. Thus, the following fuzzy rule is written as,

If e(k) is Z and Δ e(k) is Z, then $K_p{}'$ is Small, $K_i{}'$ is N

From the rules above, a set of rules for K_p and K_i are shown in table 1 and 2 respectively.

Table 1. Rule base for K_i

If	e'(k)	NB	And	e(k)	NM	Then	Ki	Little
If	e'(k)	NM	And	e(k)	NM	Then	Ki	Little
If	e'(k)	Z	And	e(k)	NM	Then	Ki	Small
If	e'(k)	NB	And	e(k)	Z	Then	Ki	Normal
If	e'(k)	NM	And	e(k)	Z	Then	Ki	Normal
If	e'(k)	Z	And	e(k)	Z	Then	Ki	Normal
If	e'(k)	PM	And	e(k)	Z	Then	Ki	Normal
If	e'(k)	PB	And	e(k)	Z	Then	Ki	Normal
If	e'(k)	PB	And	e(k)	PM	Then	Ki	Normal
If	e'(k)	NB	And	e(k)	NB	Then	Ki	Medium
If	e'(k)	Z	And	e(k)	PM	Then	Ki	Medium
If	e'(k)	PM	And	e(k)	PM	Then	Ki	Medium
If	e'(k)	NB	And	e(k)	PM	Then	Ki	Big
If	e'(k)	NB	And	e(k)	PB	Then	Ki	Big
If	e'(k)	NM	And	e(k)	NB	Then	Ki	Big
If	e'(k)	NM	And	e(k)	PM	Then	Ki	Big
If	e'(k)	NM	And	e(k)	PB	Then	Ki	Big
If	e'(k)	Z	And	e(k)	NB	Then	Ki	Big
If	e'(k)	Z	And	e(k)	PB	Then	Ki	Big
If	e'(k)	PM	And	e(k)	NB	Then	Ki	Big
If	e'(k)	PM	And	e(k)	NM	Then	Ki	Big
If	e'(k)	PM	And	e(k)	PB	Then	Ki	Big
If	e'(k)	PB	And	e(k)	NB	Then	Ki	Big
If	e'(k)	PB	And	e(k)	NM	Then	Ki	Big
If	e'(k)	PB	And	e(k)	PB	Then	Ki	Big

Table 2. Rule base for K_p

If	e'(k)	NB	And	e(k)	NM	Then	Kp	Big
If	e'(k)	NM	And	e(k)	NM	Then	Kp	Big
If	e'(k)	Z	And	e(k)	NM	Then	Kp	Big
If	e'(k)	NB	And	e(k)	Z	Then	Kp	Small
If	e'(k)	NM	And	e(k)	Z	Then	Kp	Small
If	e'(k)	Z	And	e(k)	Z	Then	Kp	Small
If	e'(k)	PM	And	e(k)	Z	Then	Kp	Small
If	e'(k)	PB	And	e(k)	Z	Then	Kp	Small
If	e'(k)	PB	And	e(k)	PM	Then	Kp	Big
If	e'(k)	NB	And	e(k)	NB	Then	Kp	Big
If	e'(k)	Z	And	e(k)	PM	Then	Kp	Big
If	e'(k)	PM	And	e(k)	PM	Then	Kp	Big
If	e'(k)	NB	And	e(k)	PM	Then	Kp	Big
If	e'(k)	NB	And	e(k)	PB	Then	Kp	Big
If	e'(k)	NM	And	e(k)	NB	Then	Kp	Big
If	e'(k)	NM	And	e(k)	PM	Then	Kp	Big
If	e'(k)	NM	And	e(k)	PB	Then	Kp	Big
If	e'(k)	Z	And	e(k)	NB	Then	Kp	Big
If	e'(k)	Z	And	e(k)	PB	Then	Kp	Big
If	e'(k)	PM	And	e(k)	NB	Then	Kp	Big
If	e'(k)	PM	And	e(k)	NM	Then	Kp	Big
If	e'(k)	PM	And	e(k)	PB	Then	Kp	Big
If	e'(k)	PB	And	e(k)	NB	Then	Kp	Big
If	e'(k)	PB	And	e(k)	NM	Then	Kp	Big
If	e'(k)	PB	And	e(k)	PB	Then	Kp	Big

4. APPLICATION OF FUZZY PI CONTROLLER

We use motor and generator which is connected to motor with a cuplin. In this application we want to control speed of this motor. Generator is used to make disturbance effect. Figure 6 describes the topology of the motor speed control application. The 92W permanent magnet dc motor is supplied in dc source which is rectified from 240W transformer and filter with 13600 μ F. This voltage is adjusted using PWM techniques through a chopper stage composed of an insulated gate bipolar transistor (IGBT) IXTH32N60 and a freewheling diode 30N600. A RISC (Reduced Instructions Set Complex) microcontroller 16cXX with its on-board PWM timer and A/D converter, measures the speed of the motor from the analog tacho-generator and drives the IGBT through the 5/+12,-12 Volt interface.

The microcontroller manages tasks such as tacho-generator voltage measurement, fuzzy logic controller, PWM duty cycle generation and communicating with PC on RS-232 to select motor speed. Maksimum speed of motor is 3000r/min. In 3000r/min, tacho-generator's output is 5 volt (It's also full scale for A/D converter).

In spite of using 8 bit microcontroller, to get a more resolution in operations we use 16 bit subroutins in PI algorithm. The output variable is the PWM duty cycle. In the application we select the PWM frequency as 20kHz. The PWM duty cycle is calculated in PI control algorithm at every 0.5ms and applied to the IGBT gate for fine tuning the motor speed. This PWM output signal, automatically generated by the microcontroller PWM timer, ranges from %0 to %100 with a resolution of %0.4 (full range of the PWM duty cycles coded from 0 to 255).

The "fuzzyTECH PIC16cXX Explorer Edition" used to develop this application covers all the steps of a fuzzy logic design from the definition of a project, of the linguistic variables and of the rules [4]. Furthermore, this tool generates the executable code for PIC16cXX microcontroller. Fuzzy logic program can be divided in two main parts: the microcontroller environment program and the fuzzy logic application itself.

1. The environment program consists of microcontroller initialization, PI algorithm, motor speed acquisition, input variable adaptation to fuzzy logic kernel code values, fuzzy logic kernel calling, PWM duty cycle updating, acquisition of new motor speed command.
2. The fuzzy logic part consists of 16cXX executable code is generated by the development tool. This part is made of the fuzzification of the input variables, execution of the activated rules, and defuzzification producing the output variable.

5. REAL TIME TEST AND PRACTICAL RESULTS

The real time test of the application can be done using an eprom version of PIC 16cXX device. Real time recording of the application variables measuring the motor behaviour (Tachogenerator voltage, PWM variation) while the motor is running, is very helpful. This communacition can be achieved by a 38400 bauds serial link between 16c65 and a computer recording data. This data file can be used in displaying and analyzing the motor speed. Recording of tachogenerator voltage (corresponding to the motor speed) is shown following figures for PI and Fuzzy-PI control operations.

In steady state, we can notice motor speed fluctuations of $\pm$ 11r/min. These fast fluctuations (0.5ms) are due to resolution limit of speed measurement (8 bit A/D converter) and remain inaudible.

We try this controller comparison for three load. The motor permanently loaded 39, 45, 52W respectively.

6. CONCLUSION

The proposed gain scheduling scheme uses fuzzy rules and reasoning to determine the PI controller parameters. Human knowledge and experience in control system design is exploited in the tuning of a PI controller. The scheme has been tested on various load and satisfactory results are obtained. Comparison PI controller with Fuzzy-PI controller is given table 3. Figure 7, 8 and 9 show the comparison Fuzzy-PI controller and Classic-PI one for 39, 45 and 52W load respectively. The paper successfully shows the application of Fuzzy-PI dynamic controller for a permanent magnet DC servo motor drive system. The performance of the system was found to be excellent in the speed regions.

Table 3. Comparison of controllers

Process	Load (W)	Over shoot (rpm)	t_{settle} (ms)	t_{rise} (ms)	Current (A)	Voltage (V)	ITAE (%)	ISE (%)
Fuzzy-PI	52	289	177	73	1.53	33.6	4.4	8.14
PI	52	335	217	84	1.53	33.6	5.74	10.4
Fuzzy-PI	45	276	177.5	72	1.28	34.1	4.44	8.08
PI	45	312	216	80	1.28	34.1	5.697	10.14
Fuzzy-PI	39	241	174	70	1.13	34.6	4.4	7.95
PI	39	276	213	77	1.13	34.6	5.67	9.9642

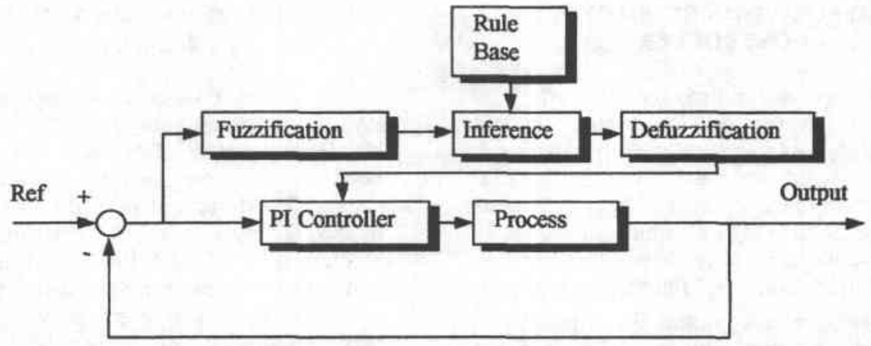

Figure 1. PI control system with a fuzzy gain scheduler

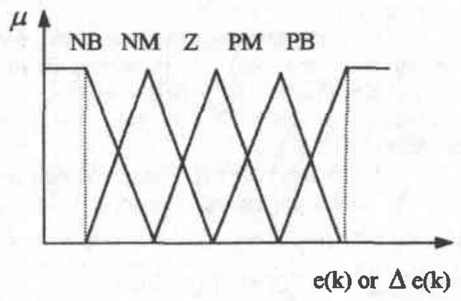

Figure 2. Membership functions for error and its first derive.

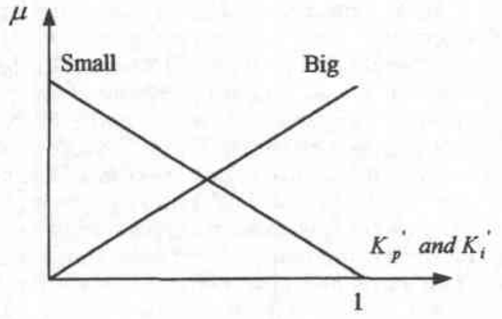

Figure 3. Membership functions for K_p'

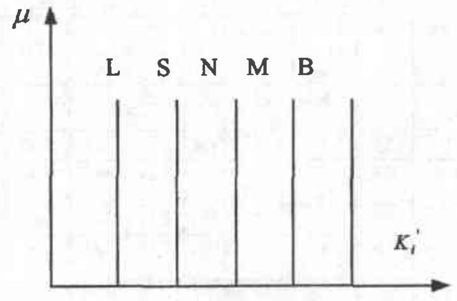

Figure 4. Membership function for K_i'

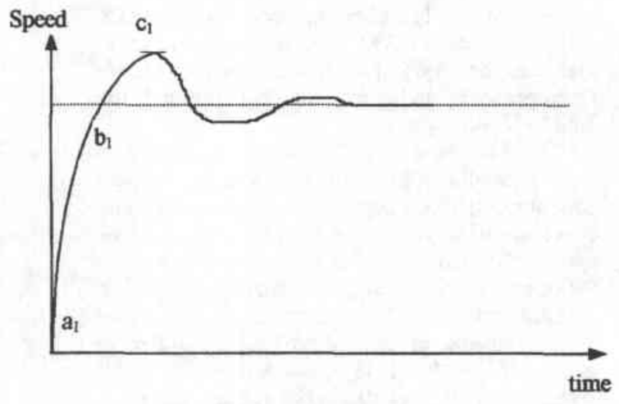

Figure 5. Process time response

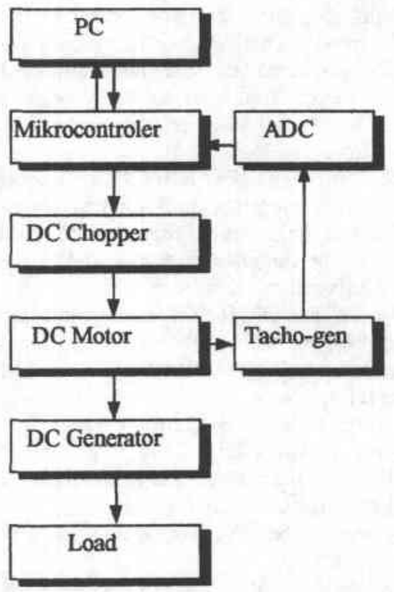

Figure 6. Block diagram of the process

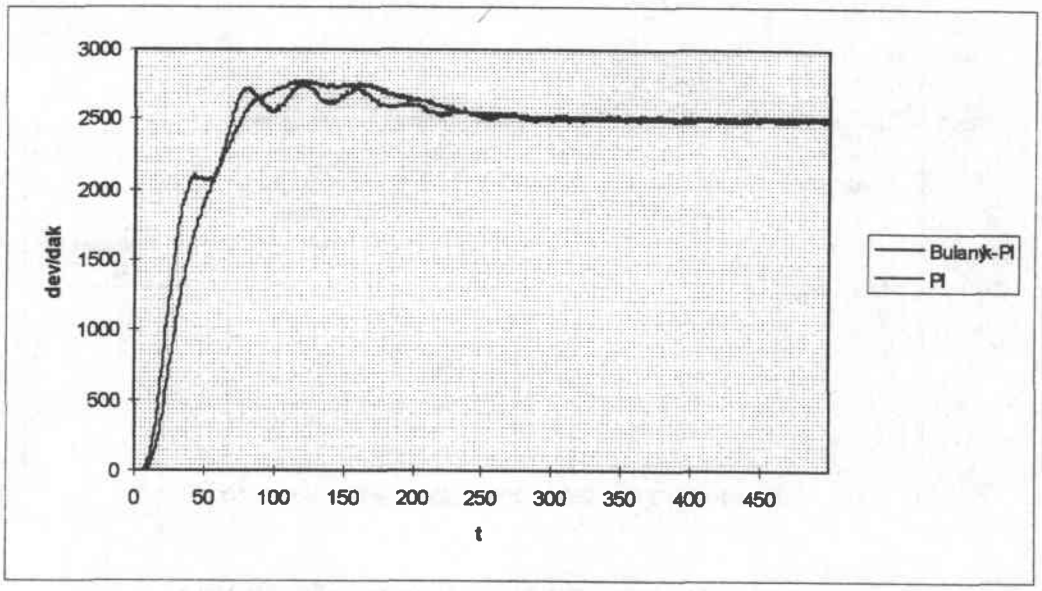

Figure 7. Comparison Fuzzy-PI and PI controller for 39W load

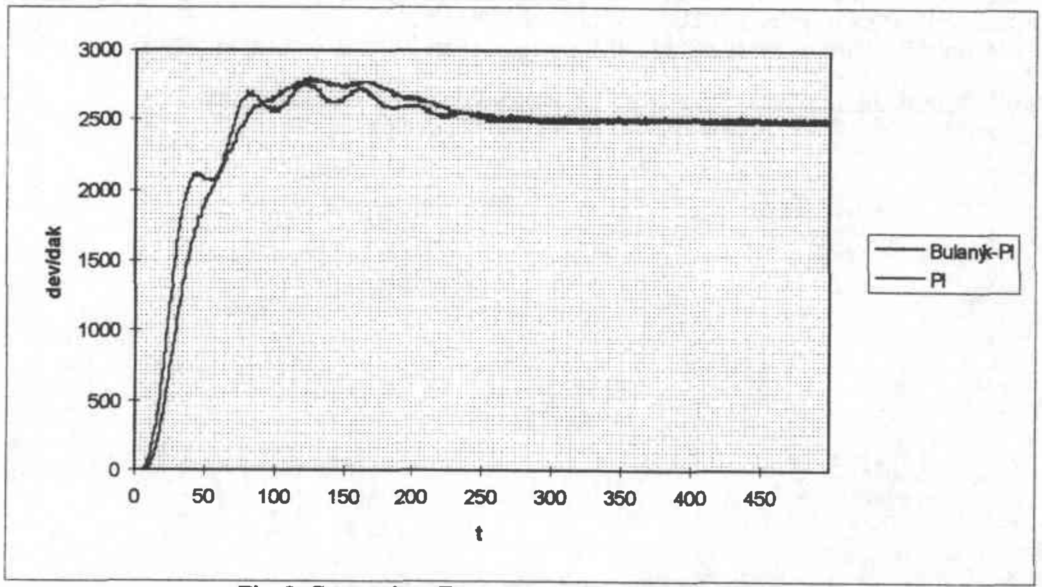

Fig. 8. Comparison Fuzzy-PI and PI controller for 45W load

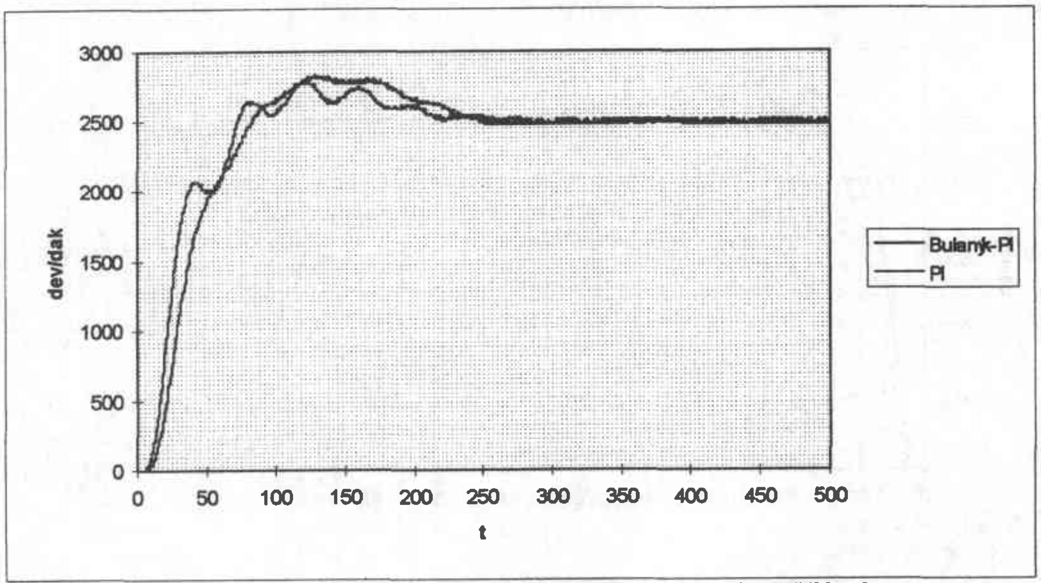

Figure 9. Comparison Fuzzy-PI and PI controller for 52W load

REFERENCES

[1] Zhao, Z.Y., Tomizuka, M. Ve Isaka, S., Fuzzy Gain Scheduling of PID controllers, IEEE Transactions on Systems, Man and Cybernetics, vol. 23, No. 5, pp. 1392-1398, 1993.
[2] Guillemin, P., Universal motor control with fuzzy logic, Fuzzy Sets and Systems, vol. 63, Iss. 3, pp. 339-348, 1994.
[3] B. C. Kuo, Automatic Control Systems, 5th. Ed. Englewood Cliffs, Prentice-Hall, 1987.
[4] FuzzyTECH PIC16cXX Explorer Edition User Manuel, Microchip.