Auto-tuning of PID controller based on fuzzy logic

Łukasz Niewiara, Krzysztof Zawirski
Poznań University of Technology
60-965 Poznań, ul. Piotrowo 3a, e-mail: krzysztof.zawirski@put.poznan.pl

Issues related to the automatic selection of the PID controller settings have been known for several years. This article describes the concept of self-tuning using fuzzy logic block (FLB). The FLB represents expert knowledge and in the process serves as the master. It describes the advantages of the designed structure of fuzzy logic simulation system application. The paper contains block diagram descriptions of designed controller and control loop. The designed tuning algorithm bases on quality indicators and rule bases which was described as membership functions. The readjustment of the controller settings takes place in subsequent iterations of the tuning process. Simulation tests result of the proposed tuning system topology were presented. As the simulation environment was used Matlab Simulink.

KEYWORDS: PID controller, auto-tuning, fuzzy logic.

1. Introduction

The auto-tuning process of PID controllers allows to minimize influence of the human factor in the selection of the settings. Influence is limited to run the process and possibly supervision it. In view of the speed of technological development and industrial use of self-tuning algorithms makes it possible to accelerate the implementation of projects. The conventional approach to the controller tuning process is often a tedious and time-consuming process, in addition, this process is constrained by the criteria availability. Self-tuning processes are realized by a specific algorithm, during subsequent iterative cycles. The process is direct by a driver that much faster analyzes the process data, than a man. This results in a significant acceleration of the tuning process, which allows for faster development of industrial applications.

The use of fuzzy logic to perform the tuning process led to his more general description, where it becomes partially independent from strictly mathematical description of the algorithm. This approach takes advantage of an imprecise description of quality indicators of the control process by appropriate linguistic variables and rule bases. The fuzzy logic block (FLB) in the designed structure does not directly affect the regulation process, it accounts only for updating the controller settings. This approach allowed the use of the traditional PID control algorithm, where FLB is a parent block corresponding expert knowledge.

2. Simulated system description

The self-tuning was simulated with the help of MatLab environment by using of m-scripts created for this purpose. The model of the control system was built in Simulink.

In the auto-tuning simulation process of this system (Fig. 1) as the actuator was used a cascade connected inverter and controller model. The controlled object model output closes the negative feedback loop.

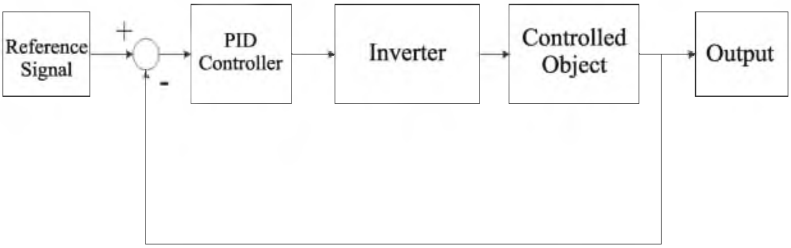

Fig. 1. Designed control system block diagram

The PID controller was able to be adjusted in one of three modes, depending of the type of controlled plant model. There was possible to select one of the following configurations: PD, PI or PID. The structure is shown below (Fig. 2).

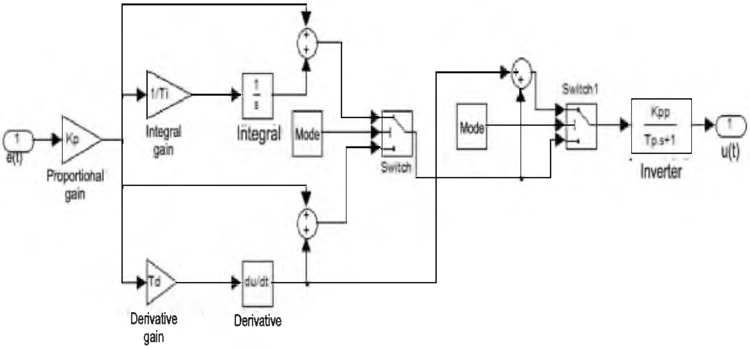

Fig. 2. Designed controller structure

The tuning process carried out for two models of control objects: second order inertia and integration with inertia (static and astatic model). The tuning process is carried out according to the designed algorithm (Fig. 3). Before the beginning of the self-tuning process, there must specify initial values of the control parameters, maximum allowed overshot and normalized regulation time.

As the quality indicators were chosen maximum response overshot - ε and normalized regulation time – T_N.

$$\varepsilon = \frac{y_{max} - y_{ref}}{y_{ref}} \tag{1}$$

where: y_{max}– maximum response value, y_{ref} – reference signal.

$$T_w = \frac{T_r}{100T_p} \tag{2}$$

where: T_r – regulation time for 2% regulation track, T_p – inverter time constant.

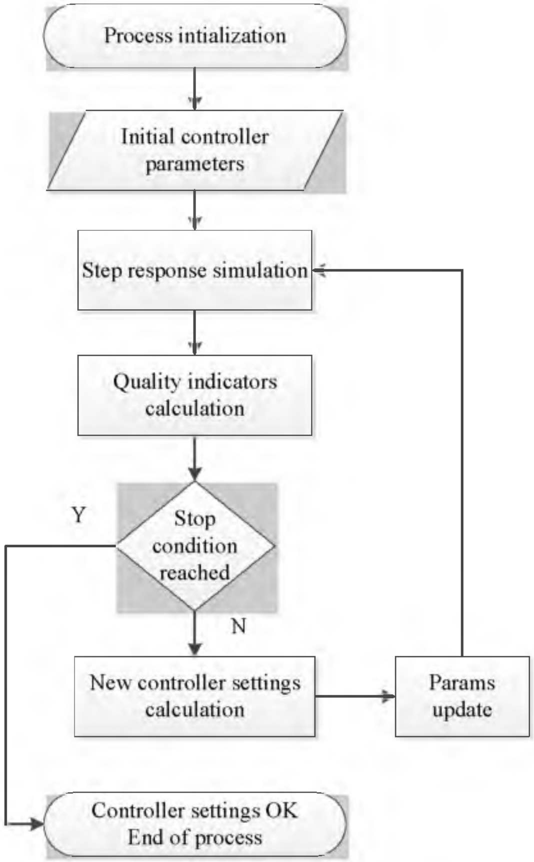

Fig. 3. Tuning algorithm

The stop condition was defined as maximum number of tuning loops or response meets the quality criteria (not greater than maximum allowed overshot and regulation time).

3. Fuzzy Logic Block

The fuzzy logic bloc is responsible for the calculation of value changes of the controller settings. They are calculated based on the value of quality indicators (Fig. 4).

Fig. 4. Fuzzy Logic Block – input and output data

Controller settings changes realized according to the following formula:

$$K_p(i+1) = K_p(i)C^{\Delta K_P} \tag{3}$$

$$T_I(i+1) = T_I(i)C^{\Delta T_I} \tag{4}$$

$$T_D(i+1) = T_{DI}(i)C^{\Delta T_D} \tag{5}$$

where: C – power base that determines the speed of settings change, $K_p(i+1)$ – new value of proportional gain, $T_I(i+1)$ – new value of integral time constant, $T_D(i+1)$ – new value of derivative time constant, ΔK_P, ΔT_I, ΔT_D – the calculated values of controller settings growth for the i-th iteration.

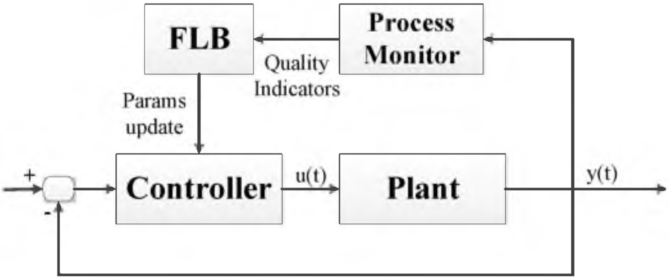

Fig. 5. Designed self-tuning system structure

The FLB is a very important part of the designed system (Fig. 5), together with the process monitor they realize controller settings changes. Process monitor calculates the quality indicators based on the plant response. Then operates the fuzzy rule bases, so that sets the value of the adjustment changes. This involved designing a membership function.

For the input parameters were defined membership functions shown in (Fig. 6 and 7), they represents following set of linguistic variables: {Z, S, M, B}, where:

Z – zero, S – small, M – medium, B – big. They shapes and ranges were appointed empirically.

For the output parameters were defined membership functions shown in (Fig. 8 and 9), they represents following set of linguistic variables: {NB, NS, Z, PS, PB}, where: NB – negative big, NS – negative small, Z – zero, PS – positive small, PB – positive big. They shapes and ranges were appointed empirically.

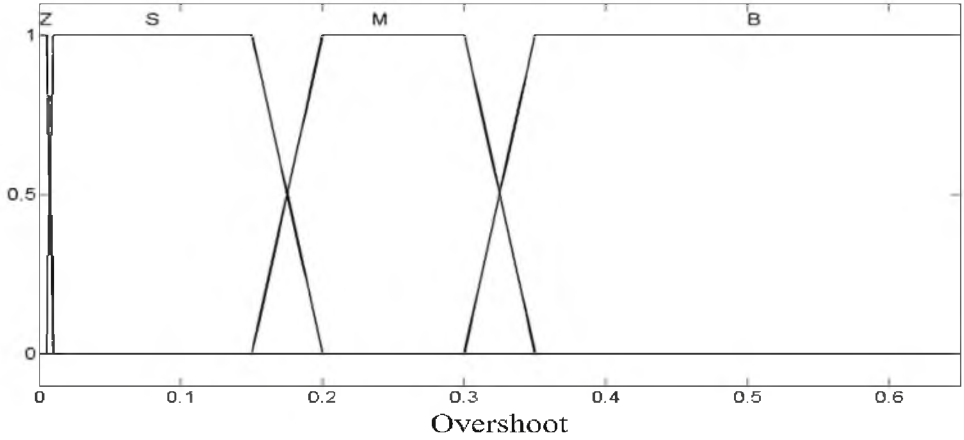

Fig. 6. Proposed overshoot Membership Function

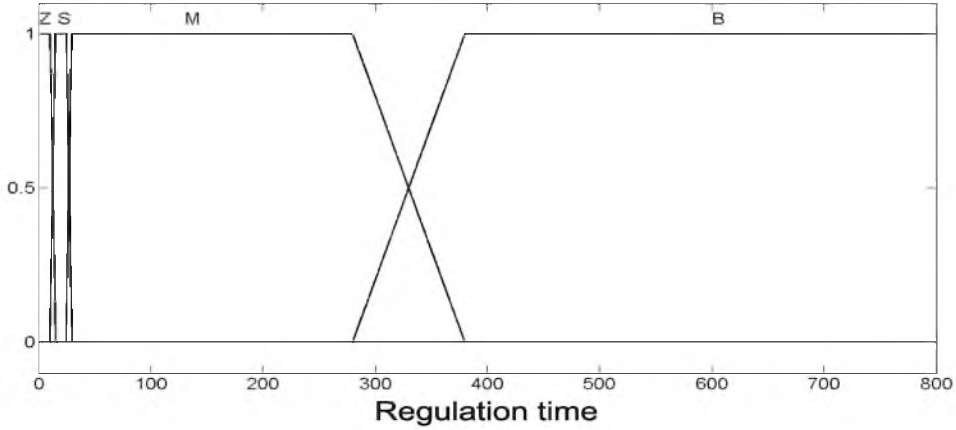

Fig. 7. Proposed regulation time Membership Function

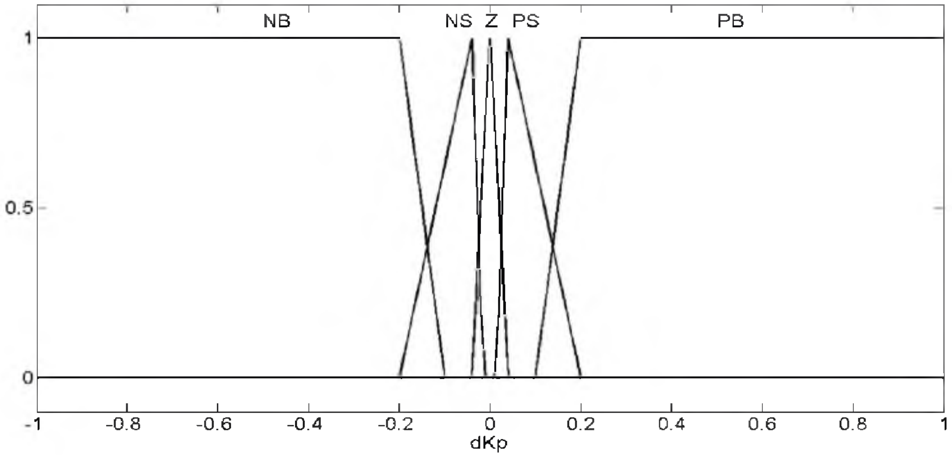

Fig. 8. Proposed ΔK_P Membership Function

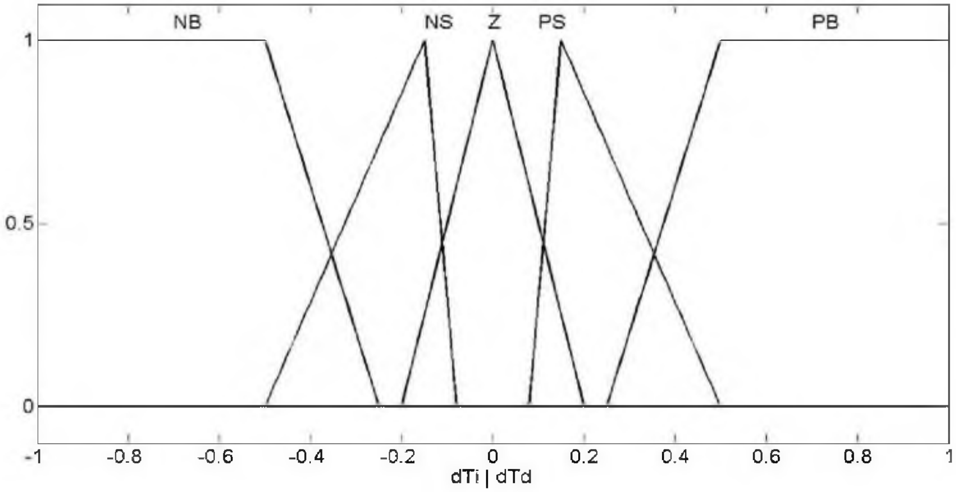

Fig. 9. Proposed ΔT_I and ΔT_D Membership Functions

One of the most important steps during the designing of FLB was to define the rule bases. There were defined different bases for each controller type. Two for the PD mode, two for the PI mode for astatic model, two for PI mode for static model, and three for PID mode. The proposed solution was shown below (Table 1-9).

Table 1. Proposed ΔK_P rule base for PD mode

ΔK_P	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PS	NS	NS	NB
	S	Z	NS	NS	NS
	M	PS	NS	NS	NB
	B	PS	PS	NB	PB

Table 2. Proposed ΔT_D rule base for PD mode

ΔT_D	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PB	NS	Z	NB
	S	NS	NB	NS	PS
	M	NS	PS	PB	PS
	B	NB	PS	NB	PB

Table 3. Proposed ΔK_P rule base for PI mode – astatic model

ΔK_P	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PB	NS	NS	NB
	S	PS	NS	NS	NS
	M	PB	PS	NS	NS
	B	PB	Z	Z	NS

Table 4. Proposed ΔT_I rule base for PI mode – astatic model

ΔT_I	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	Z	Z	Z	Z
	S	Z	Z	Z	PS
	M	PS	PB	PS	PS
	B	PS	NB	PB	PB

Table 5. Proposed ΔK_P rule base for PI mode – static model

ΔK_P	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PS	NS	NS	NB
	S	NB	NS	NS	NS
	M	PB	NS	NS	NB
	B	PB	PS	NS	NS

Table 6. Proposed ΔT_I rule base for PI mode – static model

ΔT_I	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PB	NS	NS	NB
	S	NS	NB	NB	Z
	M	NS	PS	PB	Z
	B	NB	PS	NB	NB

Table 7. Proposed ΔK_P rule base for PID mode

ΔK_P	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PS	NS	NS	NB
	S	NB	NS	NS	NS
	M	PB	NS	NS	NB
	B	PB	PS	NS	NS

Table 8. Proposed ΔT_D rule base for PID mode

ΔT_D	**Overshoot**				
Regulation time		**Z**	**S**	**M**	**B**
	Z	PS	PS	Z	NS
	S	NS	PS	Z	Z
	M	NS	NS	Z	NS
	B	NB	NS	NS	NB

Table 9. Proposed ΔT_I rule base for PID mode

ΔT_I		Overshoot			
Regulation time		**Z**	**S**	**M**	**B**
	Z	PB	PS	Z	Z
	S	NS	NB	NB	Z
	M	NS	PS	PB	Z
	B	NB	PS	NB	NB

4. Simulation results

The reference signal was given to the input as a step signal with a amplitude equal 100. Then the system response using the process monitor, was analyzed after which the quality indicators were calculated. Halting followed after meeting the criteria by the system response or after reaching the maximum loops number. Selected simulation results was shown below (Fig. 10-13). Figures illustrate the process of self-tuning for four different control objects. The illustration shows that after a few or several steps the tuning process a correct control process, i.e. without overshoot and the relatively short time of regulation is achieved.

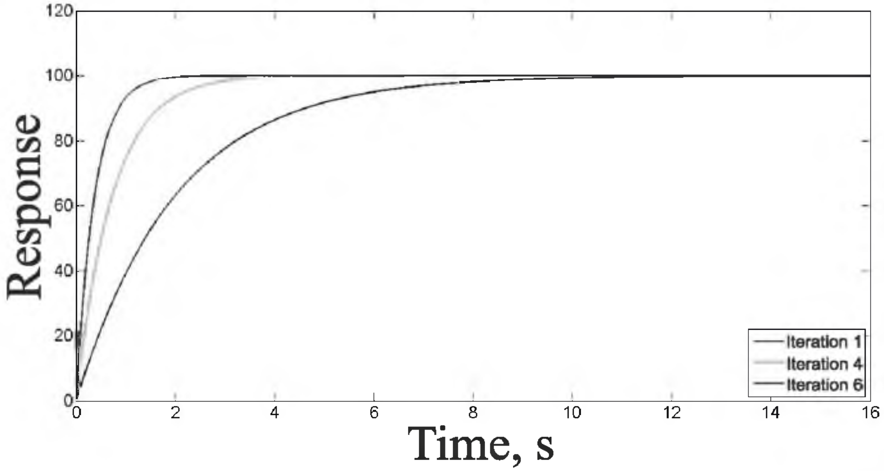

Fig. 10. Tuning process 1 – PD mode

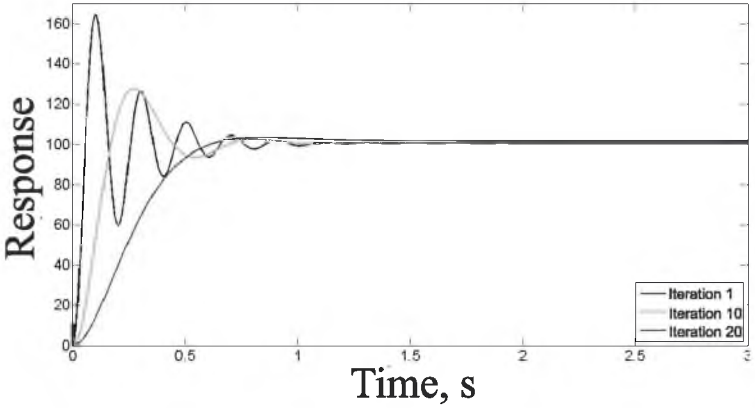

Fig. 11. Tuning process 2 – PI astatic mode

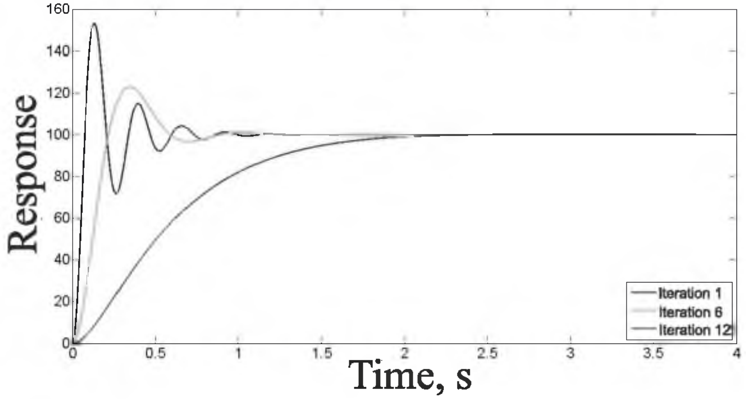

Fig. 12. Tuning process 3 – PI static mode

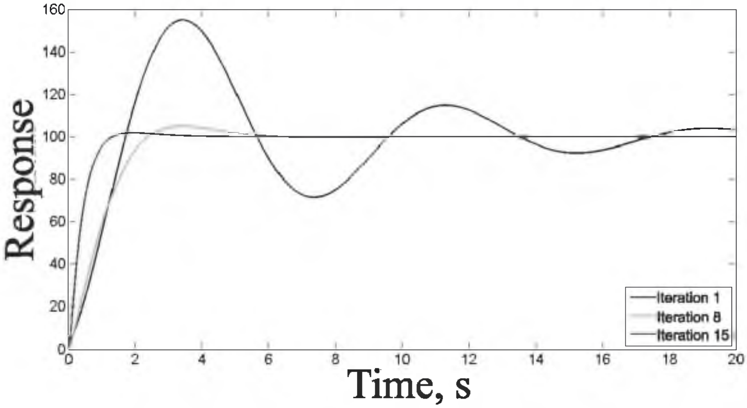

Fig. 13. Tuning process 4 – PID mode

5. Conclusions

Simulation tests of the designed auto-tuning system with defined membership functions and rules bases proved that it is successful and consistent with the objectives. The auto-tuning process starting with initial controller parameters, modify them until the response meets the criteria put to it. The simulation results shows that it is successful for different class types of control object models. After several iterations response oscillations are reduced finally giving a benign course signal.

References

[1] R. Kaula „Podstawy automatyki” Wydawnictwo Politechniki Śląskiej, 2005.

[2] A. Piegat „Modelowanie i sterowanie rozmyte” Akademicka Oficyna Wydawnicza EXIT, Warszawa 1999.

[3] Ł. Niewiara „Samostrojenie regulatora PID dla obiektu o nieznanych parametrach z wykorzystaniem logiki rozmytej”, Politechnika Poznańska, Praca magisterska, 216/Z2/2012.

[4] Ł. Niewiara, K. Zawirski, „Auto-strojenie regulatora typu PID z wykorzystaniem logiki rozmytej”, Poznan University of Technology Academic Journals, Electrical Engineering Issue 75, Poznan University of Technology, Poznań 2013, ISSN: 1897-0737, 79-84.