Designing Fuzzy Speed Controller for DC Drives and Tuning with Genetic Algorithms

Ferenc Farkas, Sándor Halász, István Kádár

Department of Electric Power Engineering
Budapest University of Technology and Economics
Egry József u. 18, H-1111 Budapest, Hungary
ikadar@eik.bme.hu

Abstract: In this article the designing method of a fuzzy speed controller for DC drives is presented. The designing phase consists of choosing the number of membership functions for input/output and constructing the minimal rule base. The design phase is completed with tuning the input/output membership functions using genetic algorithm. The difference between the actual and the expected transfer function of the fuzzy controller is chosen as the performance function of the genetic algorithm. The expected transfer function is constructed by analyzing the behaviour of a conventional PID speed controller for different inertia, load and speed reference. Finally, the DC drive step response is presented using both a conventional PD controller and the designed fuzzy controller.

Keywords: fuzzy controller, DC drives, genetic algorithm

1 Introduction

The "conventional" PID speed controller can be successfully applied in controlling linear plants, but it is not able to cope with nonlinear plants with the same success. By conventional PID controller it is meant a controller having a proportional, derivative, and integral input and with a linear transfer function. Because even a DC drive might behave as a nonlinear system, where nonlinearities may appear due to armature current limitations, change of load and drive inertia, it might be useful to use controllers with non-linear transfer function.

One of the performance indicator of the speed controller is the system step response for a given reference speed. It is desired that the step response of the system has minimal rise time and without overshoot. However, conventional PD or PI controllers cannot be tuned in such way that the optimum step response is achieved for different inertia, load and speed reference. This is why a nonlinear controller is needed, like the fuzzy controller, even in case of DC drives.

Fuzzy controller is robust and nonlinear, the two well-known features of it. But how nonlinear a fuzzy controller is depends on its parameters, like its membership functions and rule base. One way to set these parameters is the so-called trial and error method. In this article another design method is presented. To understand the reasoning, a short analyze is presented about the drawback of a PD controller.

2 Limitations of the PID Controller

For simplifying the analyses, only PD or PI controller behaviour is presented. Moreover, as the PI controller can be constructed as a PD controller, but having the change of control signal instead of the control signal itself, the analyses is simplified even more. Thus, a PD controller behaviour is analysed having as input the E error between the reference speed ω_r and motor speed ω_m, and the output as the current reference I_r. In Fig. 1 the step response of a DC motor is presented in ideal circumstances, that is, when the armature current can rise and fall instantaneously, for example from $I_{\max}$ to I_n. E and DE represent the error and change of error, respectively, while the meaning of ε and Δt will be clarified later on. The acceleration of the DC drive is given by the following equation:

$$\frac{d\omega_m}{dt} = \frac{M_{\max} - M_l}{\Theta} \tag{1}$$

where $M_{\max}$, M_l are the torque determined by the maximal current and the load, respectively, and Θ is the drive inertia.

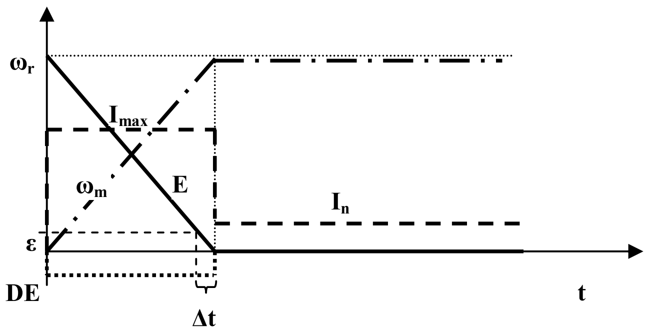

Figure 1
Step response of the DC motor

In ideal circumstances, as the motor speed reaches the reference speed and the E error drops to zero, the maximal armature current falls instantaneously to the nominal current, where the nominal current value depends on the total load of the drive. However, the armature inductance is never zero, thus, the armature current cannot fall instantaneously. This means, overshoot can be prevented only if the current reference falls to the nominal value with Δt time earlier respect to the zero error. The PD controller is described with the following equation:

$$I_r = K_P \cdot E + K_D \cdot DE \tag{2}$$

where $E = \omega_a - \omega_m$ and $DE = \dfrac{d(\omega_r - \omega_m)}{dt}$.

From equation (2) it is obvious that maximal current reference exist until the inequality (3) holds:

$$K_P \cdot | E | \geq K_D \cdot | DE | + I_{\max} \tag{3}$$

For this reason, the tuning of the PD controller has to be done with taking into account that inequality (3) must be held only for $| E | \geq \varepsilon > 0$. From the Fig. 1 it can be seen that

$$\varepsilon = \frac{d\omega}{dt} \cdot \Delta t \tag{4}$$

where Δt is determined by the electrical time constant of the armature, that is, $\Delta t \approx 3 \cdot \dfrac{L_a}{R_a}$. Taking the limit of the inequality (3), that is, when the equality holds, we have:

$$K_P \cdot | \varepsilon | = K_D \cdot \left| \frac{d\omega}{dt} \right| + I_{\max} \tag{5}$$

The K_P value is tuned in such way that the permanent error of the PD speed controller should be as less as possible, whereas the K_D value is set to a value where no overshoot occurs. However, the ratio of the K_P, K_D values depend on the change of motor speed, which in turns depends on the load and inertia. This statement is obvious when ε from equation (5) is substituted into equation (6):

$$\Delta t = \frac{\varepsilon}{\dfrac{d\omega}{dt}} = \frac{K_D}{K_P} + \frac{I_{\max}}{K_P \cdot \dfrac{d\omega}{dt}} \tag{6}$$

Let's have the K_P, K_D values tuned for a given $\frac{d\omega}{dt}$ change of speed and analyze the system step response when the speed of change is increased either because of the load or the drive inertia is decreased, that is:

$$\frac{d\omega_1}{dt} > \frac{d\omega}{dt} \tag{7}$$

By rearranging inequality (7) it is obtained:

$$\frac{1}{\frac{d\omega_1}{dt}} < \frac{1}{\frac{d\omega}{dt}} \tag{8}$$

Multiplying both side by $\frac{I_{\max}}{K_P}$ and adding to both side $\frac{K_D}{K_P}$ it is obtained:

$$\frac{K_D}{K_P} + \frac{I_{\max}}{K_P \cdot \frac{d\omega_1}{dt}} < \frac{K_D}{K_P} + \frac{I_{\max}}{K_P \cdot \frac{d\omega}{dt}} \tag{9}$$

which is equivalent with $\Delta t_1 < \Delta t$. Thus, in the latter case the current reference will fall later and overshoot will occur. It can be concluded, that the step response of the DC drive will overshoot when the change of speed is greater compared to the change of speed at which the PD controller was tuned. Similar conclusion can be drawn when the PD controller is tuned for a big speed reference and a much lower speed reference is applied to DC motor. The reasoning is the same.

3 Solution to the Drawback of the PD Controller

The DC drive has optimal step response only and only if $\Delta t_1 = \Delta t$, when the change of load increases. This means, that according to relation (9), K_{D1} must be set to a higher value when K_P is kept constant. This is also presented in Fig. 2(b). When the speed reference is changed from a higher value to a lower one, the DC drive has optimal step response only if a lower K_{P1} value is chosen, as it is shown if Fig. 2(a). This is a nonlinear relationship between the inputs and output, where inputs are the error and change of error, while the output is the current reference. One may think of an adaptive PD controller which modifies its K_P and K_D

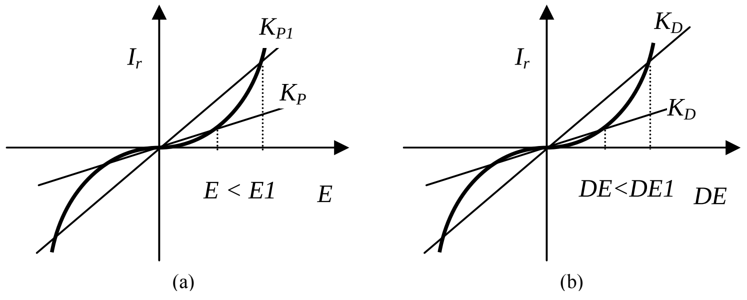

Figure 2
Current reference in function of error (a) and change of error (b)

values according to the change of load and speed reference. Such methods are already presented in the literature. However, a nonlinear controller is more suitable. The reason is, that tracking the change of load and speed reference and setting the optimal K_P and K_D value according to these changes in real time is not an easy task. Thus, a fuzzy controller is chosen to solve the problem. Why fuzzy controller? The answer is simple, fuzzy controller is robust to noise, can be made nonlinear by setting its parameters properly and easy to implement.

4 Designing the Fuzzy Controller

The proposed fuzzy controller will be a PD-like controller having two inputs, error (E), change of error (DE) and one output, the current reference (I_r). Deciding on the number of membership functions, there are two counter interest: (1) the rule base is proportional to the number of membership functions, thus in order to keep the evaluation time low the number of membership functions are set as low as possible; (2) on the other hand the number of membership functions determines the smoothness response of the fuzzy controller, thus we would like to set as high as possible.

Let's start from the lower number of the membership function. There is no reason of choosing only one membership function per input, thus having only one rule. Choosing two membership functions we have a conflict built in the fuzzy controller. In case of steady-state, that is, the error and change of error is near to zero, all the rules are firing at the same level, thus the defuzzyfication process have to resolve the conflict between the contradictory fuzzy sets suggested by the rules. In conclusion, at least 3 membership functions must be chosen for the inputs. In what follows, it must be checked whether the number of 3 chosen for the membership functions is sufficient.

When 3 membership functions are chosen per input the number of rules are 3x3=9. Let's note the sets on the error input as **E<0**, **E=0**, **E>0**, on the change of error input as **DE<0**, **DE=0**, **DE>0** and on the output as **I<0**, **I=0**, **I>0**. In Fig. 3 the error curve is presented during a transient (only the curve between the dashed lines should be considered for the moment). The rule which should be active for a given error (E) and change of error (DE) is presented on the curve with a numbered dot. From Fig. 3 it is obvious that the fuzzy controller cannot make any difference between a big error ($E_0 > 0$) caused by the change of speed reference (rule I. of the upper diagram) and a minor error caused by an overshoot (rule I. of the lower diagram). For this reason the fuzzy controller cannot act differently when having big errors instead of small ones. Thus, more then 3 membership functions are needed for the error input.

It was already pointed out that the number of membership functions should be even in order to properly handle the steady-state. Thus, 5 membership functions are chosen for the error input. That means, two sets are added to the error input **E<<0**, **E>>0** and two sets to the output **I<<0**, **I>>0**, as well. Adding two sets to the output is necessary, otherwise no use of sensing the difference in the error value. What we can see now, that we extended the rule base, and we have all the rules presented on Fig. 3.

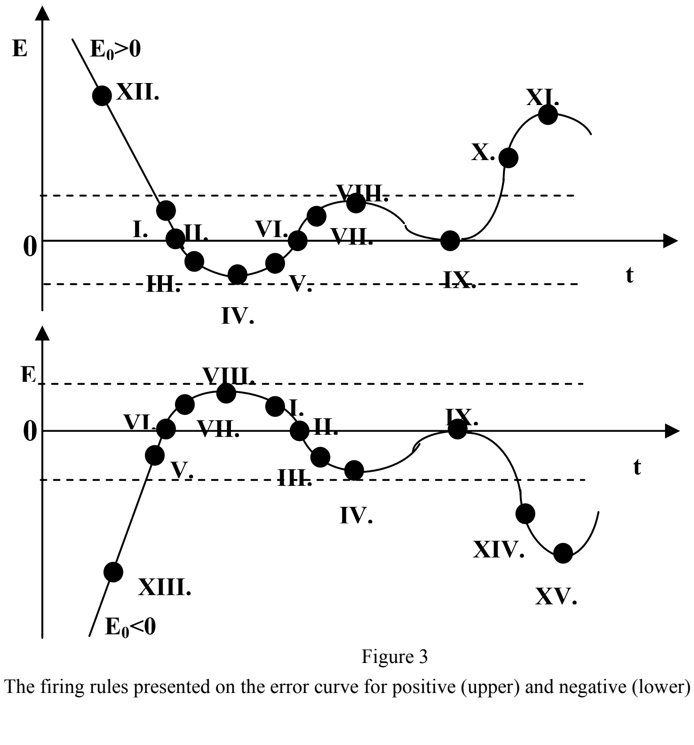

Figure 3

The firing rules presented on the error curve for positive (upper) and negative (lower) initial error

It was already pointed out, when the behaviour of the PD controller was examined, that different actions must be taken depending on the value of the change of load. The current reference should be less when the change of error is big near to zero error (the error decreases rapidly) and should be more pronounced when the change of error is small near to zero error (the error decreases slowly). Thus, there are should be 5 membership function for the change of error as well. That means, two sets are added to the change of error **DE<<0**, **DE>>0** and two sets to the current reference **I≤0**, **I≥0**. Thus we have extended the rule base again. The final version of the proposed rule base for the fuzzy controller is presented in Table 1.

Table 1

The rule base of the proposed fuzzy controller

DE \ E	E<<0	E<0	E=0	E>0	E>>0
DE<<0	I<<0 (XXV.)	I<<0 (XX.)	I<0 (XVIII.)	I≥0 (XVII.)	I>>0 (XXVI.)
DE<0	I<<0 (XIV.)	I<0 (III.)	I≤0 (II.)	I>0 (I.)	I>>0 (XII.)
DE=0	I<<0 (XV.)	I<0 (IV.)	I=0 (IX.)	I>0 (VIII.)	I>>0 (XI.)
DE>0	I<<0 (XIII.)	I<0 (V.)	I≥0 (VI.)	I>0 (VII.)	I>>0 (X.)
DE>>0	I<<0 (XXIV.)	I≤0 (XX.)	I>0 (XXI.)	I>>0 (XXII.)	I>>0 (XXIII.)

5 Tuning the Fuzzy Controller

When the tuning of the fuzzy controller is made by trial and error method, only one parameter of one membership function is changed. From the step response of the DC motor this changed is either kept or dismissed. However, taking into account the number of parameters which must be optimized and the number of step response of the DC motor which should be examined (different inertia, load and speed reference) it is clear that a better way must be find out to tune the fuzzy controller. One way of searching the optimum values of the fuzzy controller is by using genetic algorithm. Genetic algorithm has two advantages: it is easy to implement and there is no any kind of restriction on definition of the performance function of the genetic algorithm.

There are two important factors, which greatly influence the successful application of the genetic algorithm. One is the proper construction of the gene and the other is the proper choice of the performance function on which the genetic algorithm searches the parameters space. As we already know, the parameters of the membership functions have to be optimized, thus the gene (binary string) must be constructed as the chain of parameters. However, the transfer function of the fuzzy controller must be symmetric regarding to the positive and negative value of the error and change of error. Thus, the membership functions are themselves symmetric to the middle of the input/output space. That means, only half of the optimum values need to be searched, the rest will be calculated from symmetrical counterparts. In case of 5 membership functions of the error input E each having trapezoidal form with parameters $\{A_k, B_k, C_k, D_k\}$, $(k = 1,...,5)$, we need to find the optimal value of $X = \{A_1, B_1, C_1, D_1, A_2, B_2, C_2, D_2, A_3, B_3\}$ and then calculate $Y = \{D_5, C_5, B_5, A_5, D_4, C_4, B_4, A_4, D_3, C_3\}$ from X as $Y = 1 - X$ (assuming the range of input as being [0,1]).

In case of trapezoidal membership functions we need to consider that $A_k \leq B_k \leq C_k \leq D_k$, $(k = 1,...,5)$. In order to fulfil these inequalities, only the locations of the membership functions are determined in the binary string, not the actual parameters of the trapezoids. In that way, the parameters $\{A_k, B_k, C_k, D_k\}$, $(k = 1,...,5)$ can be reordered according to the inequalities before applying to the membership function and evaluating the fuzzy controller.

After the binary string is constructed the performance function has to be defined. One may think of constructing the performance function from the step response of the DC motor. However, a lot of start-ups are needed before we are able to decide about the fitness of only one individual of a population. There is a more simple way of defining the performance function for the genetic algorithm. We already got some hints about the transfer function when the behaviour of the PD controller was examined.

In what follows it is assumed that the range of fuzzy inputs and output is [0,1]. The transfer function of the fuzzy controller must have a nonlinear form as it was pointed out in Fig. 2. Thus, two constraints are defined for the transfer function of the fuzzy controller: a nonlinear curve when the error is zero **G(E=0.5)** and a nonlinear curve when the change of error is zero **G(DE=0.5)** (see Fig. 4).

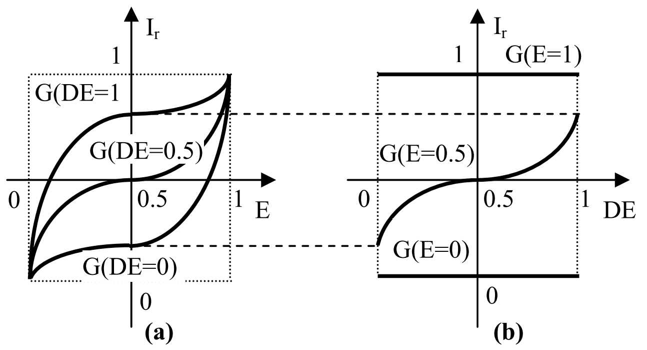

Figure 4
The border constraints for the transfer function of the change of error (a) and error (b) inputs

The shapes of these nonlinear curves are determined by the $K_P, K_{P1}, K_D.K_{D1}$ values tuned for the PD controller. The output of the fuzzy controller must have maximum value in case of big error, whatever the change of error is. Thus, we have another two constraints: **G(E=0)** and **G(E=1)**. There are two more constraints to be constructed: a nonlinear curve when the change of error is maximal, **G(DE=0)** and **G(DE=1)**. These curves intersect the I_r axes where the **G(E=0.5)** constraints has the minimal and maximal value.

Thus, the performance function of the genetic algorithm is defined as the sum of the absolute difference between the fuzzy transfer function **F(E,DE)** and the constraints **G(E,DE)** defined above:

$$\begin{aligned}\Sigma = &| F - G |_{E=0} + | F - G |_{E=0.5} + | F - G |_{E=1} + \\ &+ | F - G |_{DE=0} + | F - G |_{DE=0.5} + | F - G) |_{DE=1}\end{aligned} \tag{10}$$

6 Simulation Results

Now, that things are put together, it is time to run the genetic algorithm! The number of individuals in a population was set to 30, and the number of population during the searching was set to 500. The offspring between two populations was set to 0.9, the crossover to 0.7 and the mutation to 0.01. Both the crossover and the mutation were chosen as single-point. The transfer function of the fuzzy controller tuned by the genetic algorithm is shown in Fig. 5.

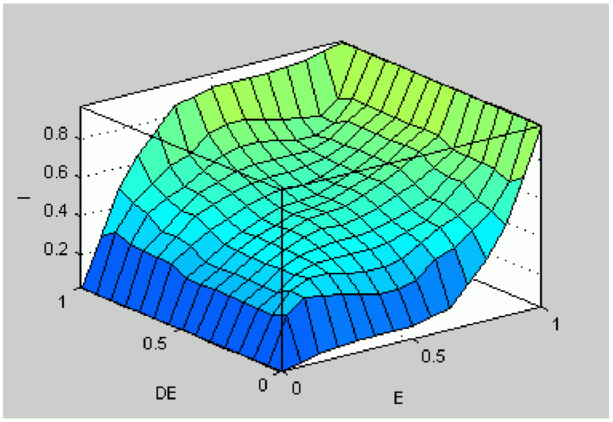

Figure 5
The transfer function of the tuned fuzzy controller

Simulation results were obtained using MATLAB environment. The parameters of the DC motors used in simulations are shown in Table 2.

Table 2
Parameters of the DC servomotor

Parameters	Notation	Value	Unit
Nominal torque	M_n	3	Nm
Nominal current	I_n	13	A
Maximal current	I_{max}	80	A
Speed domain	ω	0-2500	Rpm
Frictional torque	M_f	0.113	Nm
Rotor inertia	θ_n	0.00192	kgm^2
Torque coefficient	K_n	0.24	Nm/A
Armature inductance	L_a	1.6	mH
Armature resistance	R_a	0.49	Ω

In Fig. 6 it is shown the step response of both the PD and the fuzzy controller tuned for Θ=0.00384 kgm^2. However, when inertia is only half to which the controllers were tuned, the PD controller has on overshoot, whereas the fuzzy does not (see Fig. 7).

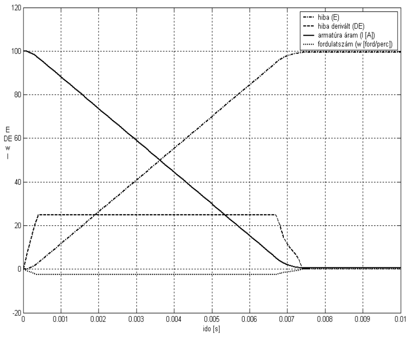

Figure 6
Step response of the PD and fuzzy controller at $\Theta=\Theta_0$

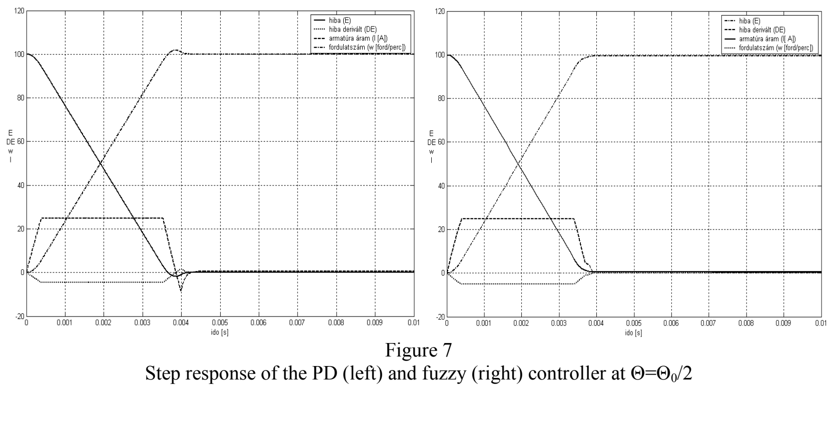

Figure 7
Step response of the PD (left) and fuzzy (right) controller at $\Theta=\Theta_0/2$

In Fig. 8 it is shown the step response of both the PD and the fuzzy controller tuned for M=–3Nm. However, when the direction of the load is reversed in respect to which the controllers were tuned, the PD controller has a more intense overshoot than the fuzzy controller (see Fig. 9).

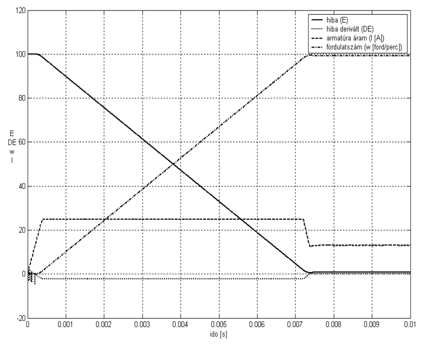

Figure 8
Step response of the PD and fuzzy controller at M_l=-M

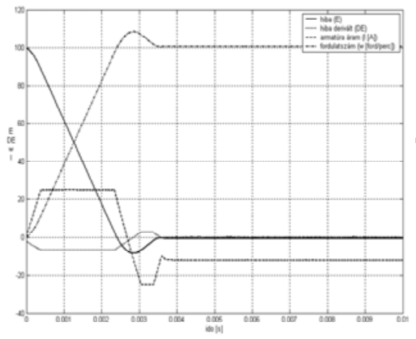
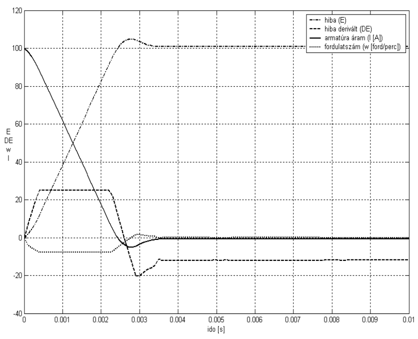

Figure 9
Step response of the PD (left) and fuzzy (right) controller at M_l=+M

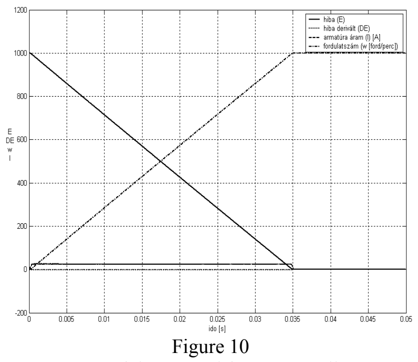

Figure 10
Step response of the PD and fuzzy controller at $\omega = \omega_0$

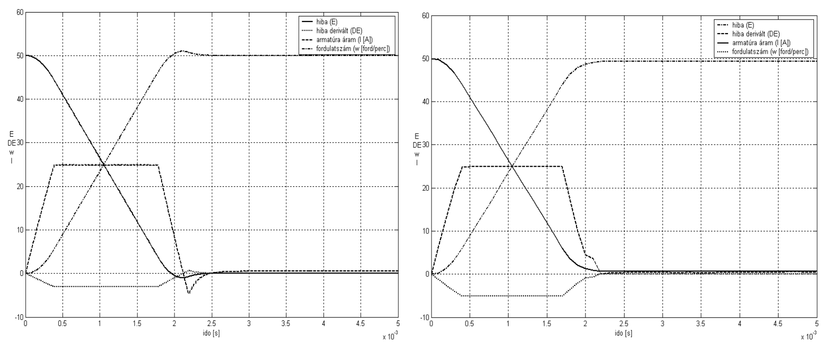

Figure 11

Step response of the PD (left) and fuzzy (right) controller at $\omega = \omega_0/20$

In Fig. 10 it is shown the step response of both the PD and the fuzzy controller tuned for ω=1000 rpm. However, when the reference speed is only one 20th of the reference speed to which the controllers were tuned, the PD controller has on overshoot, whereas the fuzzy does not (see Fig. 11).

Conclusions

It was shown in this article that in case of a DC drive the fuzzy speed controller design consists of rigorous steps leading to a well-designed and well-tuned fuzzy controller. The simulation results show that the obtained fuzzy controller outperforms the conventional PD controller when the step response of the DC drive is in focus. Although it was not presented here, the same reasoning is held when one designs a fuzzy position controller for DC drive.

Acknowledgement

This paper was supported by the Hungarian N.Sc. Fund (OTKA No. T 042866) for which the authors express their sincere gratitude.

References

[1] Mitchell Melanie: *An introduction to genetic algorithms*, MIT Press, London, 1998

[2] Farkas, F., Zakharov, A. and Varga, Sz.: “Speed and Position Controller for DC Drives Using Fuzzy Logic”, *Studies in Applied Electromagnetics and Mechanics (Vol. 16): Applied Electromagnetics and Computational Technology II*, Amsterdam, Netherlands, IOS Press, 2000

[3] Retter Gyula: *Fuzzy, neurális, genetikus, kaotikus rendszerek – Bevezetés a „lágy számítás” módszereibe*, Invest Marketing Bt., Budapest, 2003