Robust fuzzy control of electrical drive

Bogumiła Mrozek
Department of Computer Modelling, Cracow University of Technology
24 Warszawska Str. PL 31-155 Kraków
e-mail: bmrozek@usk.pk.edu.pl

Abstract

Mamdani and Takagi-Sugeno fuzzy controllers were prepared. The controllers were designed using *Fuzzy Logic Toolbox*. The membership functions and rules were used as design tools that give opportunity to model a control surface and controller properties.

The control system of d.c. (*direct current*) motor drive with two feedback loops is considered. First loop is used for stabilization of armature current (proportional to electrical torque), second loop is used for speed stabilization. *Simulink* model was used for simulation of d.c. motor drive and mechanical system. The moment of inertia and disturbance load were changed independently several times.

Time response of armature current and motor speed against disturbances of load torque and moment of inertia was tested for the above system. Drive systems using fuzzy controllers (Mamdani and Sugeno) were compared with drive using conventional PI controllers. *Nonlinear Control Design (NCD) Blockset* (an extension of *Simulink*) was used for tuning PI controller parameters.

Keywords: fuzzy logic control, Mamdani-type fuzzy system, Sugeno-type fuzzy system, d.c. drive

1. Introduction

The conventional control design is based on mathematical model of a plant. Real plant is always nonlinear. Electrical parameters of d.c. motor are nonlinear due to saturation and to temperature variations. Gain of converter is nonlinear too. Mechanical load parameters and moment of inertia may vary within limits, which are known or determined separately.

In a fuzzy control system there are many possibilities for adaptation to large unknown variation of plant parameters, without using a mathematical model. This is discussed below.

2. Simulation model of electrical drive

A linear model of d.c. drive will be used. The model of electrical drive consists of following parts: electrical motor with load (mechanical system) and the thyristor converter, which supplies the d.c. motor. A model of d.c. drive with classical control system was build using *Simulink* blocks (Fig. 2). Its parameters are computed by custom block **Run inidcdr22kW** (shown in Fig.2) from rated catalogue data: motor power 22[kW], supplied voltage 440[V], current 56.2[A], speed 1500 r.p.m., etc.

2.1. Model of d.c. motor and mechanical load

A linear model of d.c. motor was built in *Simulink* as custom subsystem (**MotorDC_I** block shown in Fig. 2) [3]. It assumed that the d.c. machine is separately exited with constant field current. The speed may be controlled from zero up to the rated value. The driven mechanism can be considered as a linear system. The system equations are then

$$L_a \frac{dI_a}{dt} = -R_a I_a - k_m \omega_s + U_{s\alpha}(t)$$

$$J_{obl} \frac{d\omega_s}{dt} = k_m I_a - M_{st}(t)$$

where:

I_a, $U_{s\alpha}$ - armature current and supplied voltage,
R_a, L_a, k_m - armature resistance, inductance, d.c. motor constant,
J_{obl}, M_{st}, ω_s - total inertia, load torque, angular motor velocity.

There are two inputs (armature supplied voltage and load torque) and two outputs (angular velocity and armature current).

2.2. Model of converter/rectifier

The three-phase bridge converter is often used in motor-control systems. Two of six thyristors conduct at any time instant. Gating of each thyristor initiates a pulse of load current; therefore this is a six-pulse controlled rectifier (converter).

The dead time may vary from zero to one-six the period of an a.c. source ($0 \div 3.33$ ms for 50 Hz). It is assumed that the mean dead time is *Tmip* =1.67[ms]. The model of converter is described as first order inertia with transfer function [6]:

$$G_{conv}(s) = \frac{U_{s\alpha}(s)}{u_{st}(s)} = \frac{k_p}{T_{mip} \cdot s + 1}$$

where:

kp - voltage amplification of converter (rectifier),
Tmip - mean dead time of converter (rectifier)

The output voltage of converter is [6]:

$$U_{s\alpha} = U_{s0} \cos \alpha = U_{s0} \cos\left(\frac{\pi}{2}\left(1 - \frac{u_{st}}{u_{st\,max}}\right)\right)$$

where:

U_{s0} , α - ideal no-load d.c. voltage,. delay angle,
u_{st} , $u_{st\,max}$ - control voltage of converter.

The voltage amplification of converter is non-linear, as shown in Fig. 1. The output voltage of armature current controller (u_{st}) is used to control the firing angle of thyristors in the three-phase full-wave converter which supplies the d.c. motor.

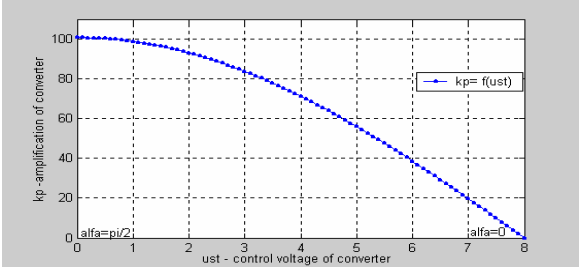

Fig.1 Output voltage and voltage amplification versus control voltage of converter.

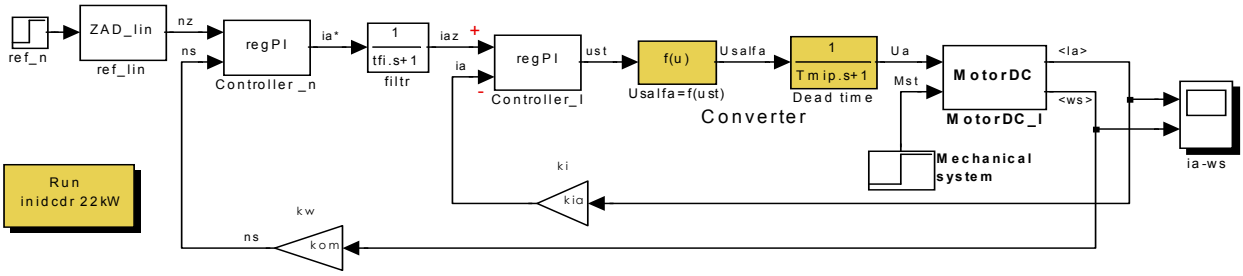

Fig. 2. Model of d.c. drive with two classical current and speed PI controllers. *Simulink* `s block is used to build the model drive.

2.3. Control system of d.c. motor drive

The control system of d.c. motor drive with two feedback loops is considered. First loop is used for stabilization of the armature current (electrical torque). Second loop is used for stabilization speed of d.c. motor. The control system of d.c. motor drive with two classical PI controllers is presented in Fig. 2.

The armature current error is the input signal to the armature controller, and it is the difference of the reference armature current and the actual armature current ($i_{az} - i_a$), where the maximum value of armature current is limited by setting of reference armature current value.

The speed controller and the armature current controller form a cascade control system. The difference of reference speed and actual speed ($n_z - n_s$) produces the speed error, which acts as an input signal to the speed controller. An output of the speed controller is the reference signal of the armature current controller, i_{az}. It should be noted that each controller has been limited, and the armature current is well controlled.

3. Tuning parameters of classical PI controllers

The *Nonlinear Control Design Blockset (NCD)* was used for tuning of the controllers' parameters, to inspect the design criteria for system with a unit step input. The *NCD Blockset* automatically converts the constraint bound data and tenable variable information into a constrained optimisation problem. It then invokes the *Optimization Toolbox* routine `constr`. The routine `constr` solves constrained optimisation problem using a sequential quadratic programming (SQP) algorithm and quasi-Newton gradient search techniques [4],[5].

The tuneable variables should be initialised. A initial values for parameters of the current and speed controllers were calculated using respectively rules of module and symmetry [6].

The *Simulink* model of non-linear plant and controller should be designed and the **NDC block** must be connected to its output signals, which to be constrained.

3.1. Tuning parameters for current controller

A plant model in current loop circuit was build as series connections of transfer functions of the converter and of armature-circuit current, as shown in Fig. 3.

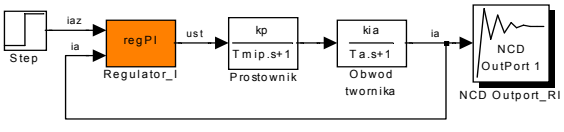

Fig. 3. *Simulink* model for tuning parameters in current loop.

The transfer function of the armature-circuit current is:

$$G_a(s) = \frac{i_a(s)}{U_{s\alpha}(s)} = \frac{k_{ia}}{T_a \cdot s + 1}$$

where: k_{ia} - gain of armature current circuit,
$T_a = L_a/R_a$ - armature-circuit time constant.

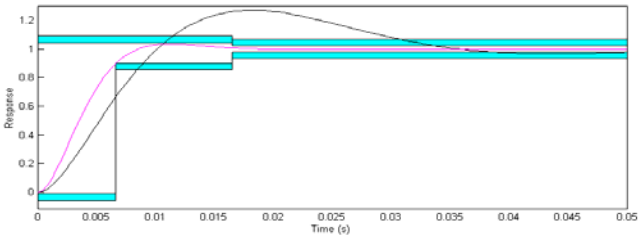

Fig. 4. *NCD Blockset* constraint window with the initial and an final response for tuning parameters of the current PI controller

The rise time, setting time and the overshoot of current were adjusted in *NCD Blockset* constraint window for the step time input, as shown in Fig. 4. Two uncertain variables were incorporated into the optimisation [3]. There are armature-circuit time constant T_a and voltage amplification of converter kp. Their ranges are: T_a varies 10% and kp varies between 34 and 101.

3.2. Tuning parameters for speed controller

A plant model in speed loop circuit was build as series connections of transfer functions of the already designed current loop and of the mechanical system, as shown in Fig. 5. The transfer function of the motor and its load (driven mechanism) is:

$$G_M(s) = \frac{\omega_s(s)}{i_a(s)} = \frac{k_{om} R_z}{k_m} \cdot \frac{1}{T_m \cdot s}$$

where: k_{om} , R_z - gain of speed circuit, total circuit resistance,
$T_m = J_{obl} R_z / k_m^2$ - electromechanical time constant.

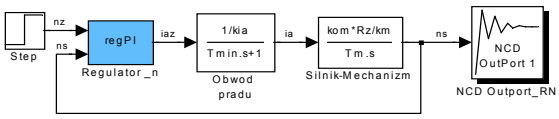

Fig. 5. Simulink model for tuning parameters in speed loop.

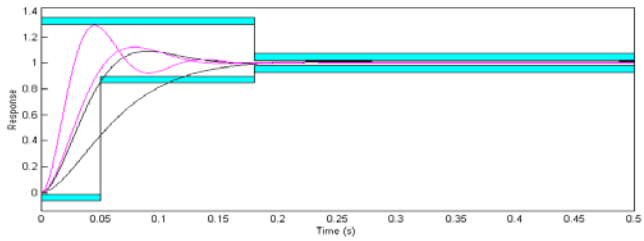

Fig. 6. *NCD Blockset* constraint window with the initial and an final response for tuning parameters of the speed motor controller.

The rise time, setting time and overshoot of motor speed for the step time response are adjusted in *NCD* constraint window (Fig. 6). The uncertain variables and their ranges are: total circuit resistance R_z varies 10%, electromechanical time constant T_m varies between 0.58[s] and 1.71[s] when total inertia J_{obl} varies respectively between 2.7[kgm^2] and 6[kgm^2].

When tuning with *NCD Blockset* is finished, the transient response is within limits adjusted in *NCD* window. The transient was initially well outside the adjusted constraints.

4. DC drive with fuzzy speed and armature current controller

Fuzzy controllers replace the speed and armature current controllers shown in Fig.2 [7]. In the first type of implementation, both the speed and armature current controllers are Mamdani-type PI fuzzy logic controllers [2]. In the second implementation, two fuzzy logic Sugeno-type PI controllers are used.

Both fuzzy controllers: Mamdani- and Sugeno-type, are implemented with the *Fuzzy Logic Toolbox*. It provides a graphical user interface (FIS Editor) [1], [3] for defining the fuzzy controller. The controller fuzzy designed may be hidden in a fuzzy block to be included in *Simulink* block diagram (Fig. 9).

4.1. Fuzzy controllers as the Mamdani system

The fuzzy logic controllers of Mamdani-type use two fuzzy variables *E*(error) and *CE* (change of error) as input, and one output fuzzy variable *CU* (change of output). The linguistic rules of the fuzzy logic controller take the form:

If *E* is *A* and *CE* is *B* then *CU* is *C*

where: A, B and C denote fuzzy set as linguistic variables (e.g. *Small Positive*, etc.)

Seven linguistic variables (from *Big Negative* to *Big Positive*) are used. The belongingness to fuzzy set is described by the membership functions. The input and output membership functions are simple Gaussian curves. There are altogether 22 rules considered. The same rules are used for both the speed and the current fuzzy controllers. The input/output block diagram of a fuzzy controller and the membership functions for fuzzy variable *error* are shown in Fig. 7

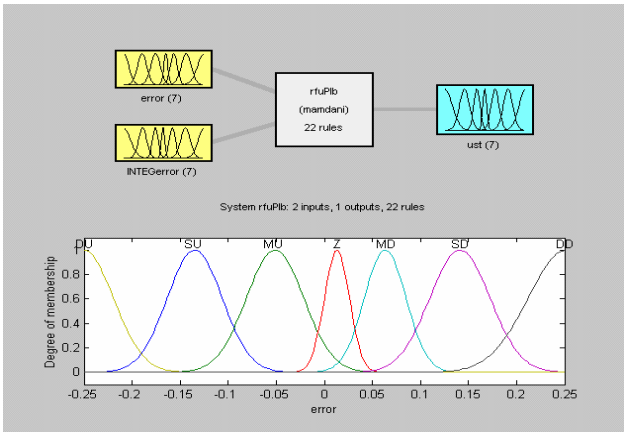

Fig.7 Block diagram of inputs/output of a fuzzy control system and the membership functionsof the fuzzy variable *error*.

The application of fuzzy PI-type controllers (for armature current and speed) give satisfactory performance when Gaussian membership functions are appropriately tuned [2].

Membership functions and rule base are design tools that give opportunity to model a control surface and controller properties. It is obvious that using this attributes one can more precisely fulfil a quality criterion in full operational range.

4.2. Fuzzy controllers as the Sugeno system

In the Sugeno-type fuzzy system, the rules take such a form that the antecedent is a function of the input variables, i.e. a typical rule in the Sugeno-type fuzzy system with two inputs is

If x is A and y is B then $z = f(x,y)$

where: x and y are input variables, A and B are fuzzy sets, and $f(x,y)$ is a function and z is the crisp output. When the function $f(x,y)$ is a first-order polynomial, the resulting fuzzy interference system is called a first-order Sugeno fuzzy model. In this case, in general, an *i*-th rule results in an output:

$z_i = f(x,y) = p_i x + q_i y + r_i$ where: p_i, q_i and r_i are constant.

The armature current fuzzy controller of Sugeno-type uses one fuzzy variable U_a (supplied voltage), crisp set *prop* (current error) and crisp set *int* (for change of current error) as inputs. One output fuzzy variable u_{st} (change of output) is used. This tree-inputs-single-output system (Fig. 9) has only tree rules with tree linguistic input/output variables:

If U_a is small then $u_{st} = k_{a1}\, prop + t_{a1}\, int$
If U_a is medium then $u_{st} = k_{a2}\, prop + t_{a2}\, int$
If U_a is large then $u_{st} = k_{a3}\, prop + t_{a3}\, int$

where: k_{a1} , t_{a1}, k_{a2} , t_{a2}, k_{a3} , t_{a3} are proportional and integral gains for classical current PI controller. All gains are tuned with *NCD Blockset* respectively for the voltage amplification $kp \in [101\ \ 93]$ (small), $kp \in [93\ \ 71]$ (medium), $kp \in [71\ \ 34]$ (large).

The speed motor fuzzy controller of Sugeno-type uses one fuzzy variable J_{obl} (total inertia), crisp set *prop* (for speed error) and crisp set *int* (for change of speed error) as inputs. One output fuzzy variable i_{az} (change of output) is used. This tree-inputs-single-output system (Fig. 9) has four rules with four linguistic input/output variables:

If J_{obl} is small then $i_{az} = k_{n1}\, prop + t_{n1}\, int$
If J_{obl} is nominal then $i_{az} = k_{n2}\, prop + t_{n2}\, int$
If J_{obl} is medium then $i_{az} = k_{n3}\, prop + t_{n3}\, int$
If J_{obl} is large then $i_{az} = k_{n4}\, prop + t_{n4}\, int$

where: k_{n1}, t_{n1}, k_{n2}, t_{n2}, k_{n3} , t_{n3}, k_{n4}, t_{n4} are proportional and integral gains for classical speed PI controller. All gains are tuned with *NCD Blockset* for total inertia of the motor and the mechanism coupled to it respectively $J_{obl} \in [1.6\ \ 2.7]$ (small), $J_{obl} \in [2.0\ \ 2.7]$ (nominal), $J_{obl} \in [2.7\ \ 4.7]$ (medium), $J_{obl} \in [4.7\ \ 6.0]$ (large).

Simple Gaussian curves are used as input and output membership functions for both designed fuzzy Sugeno-type controllers.

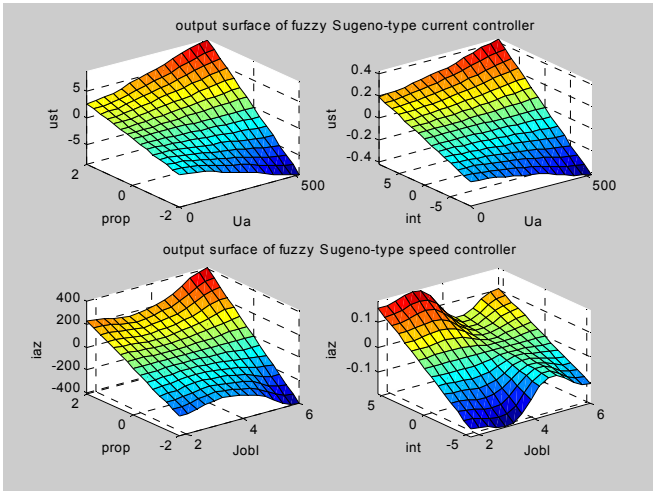

Fig. 8. Tree-dimensional view on the output surfaces of fuzzy Sugeno-type current and speed motor controllers.

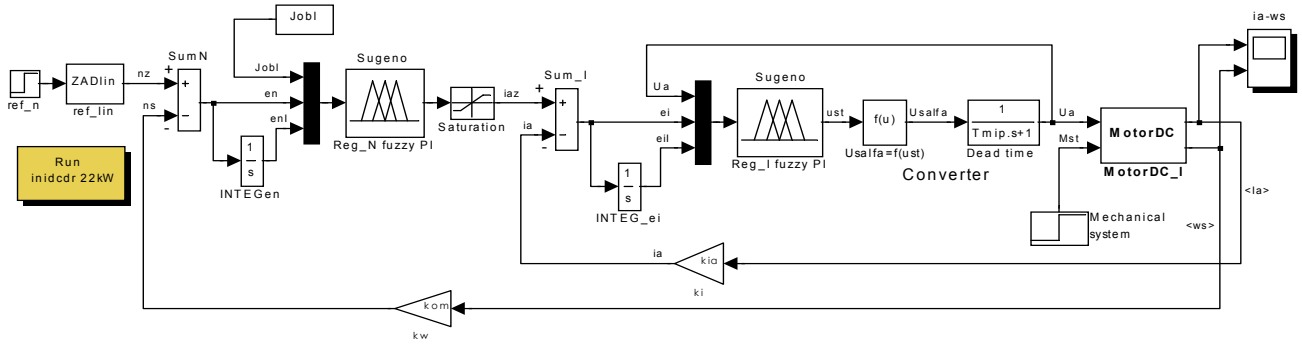

Fig.9 Model of d.c. drive with two fuzzy Sugeno-type PI controllers for armature current and motor speed

Membership functions, rule base and coefficients of the polynomial in the rule consequent are design tools that form an adaptive control system for large unknown variation of plant parameters (Fig.8).

5. Simulation results

The start of d.c. drive from n_z=0 to the reference speed n_z=114[rd/s] (e.g. 0.75n_N), with load torque equal the haft of rated value of electrical torque (M_{st}=0.5M_{elN}) was chosen for tests. At time t=1.5[s], the load torque was step increased to rated value (M_{st}=M_{elN}). This simulation was done for two values of total inertia of the mechanical system: J_{obl}=2.7 [kgm^2] and J_{obl}=6 [kgm^2].

Figure 10 shows the current and speed time responses for the drive scheme using two classical PI controllers (current and speed), as shown in Fig. 2.

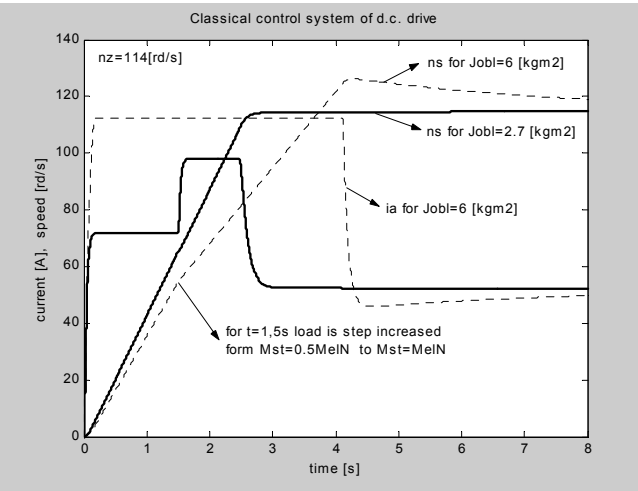

Fig. 10. Time responses for d.c. drive with two classical PI controllers

Figure 11 shows the current and speed time responses in the drive scheme using two fuzzy Sugeno-type PI controllers (current and speed), as shown in Fig. 9.

For total inertia J_{obl}=2.7 [kgm^2], the current and speed time responses in both the d.c. drives scheme (with classical and fuzzy controllers) were practically the same.

For total inertia J_{obl}=6 [kgm^2], only the fuzzy Sugeno-type PI-controlled system for d.c. drive has a satisfactory performance. This drive responded faster and more precisely than the drive using the classical PI controllers. The similar results for total inertia J_{obl}=6 [kgm^2] can be obtained using two fuzzy Mamdani-type PI controllers (current and speed) [2].

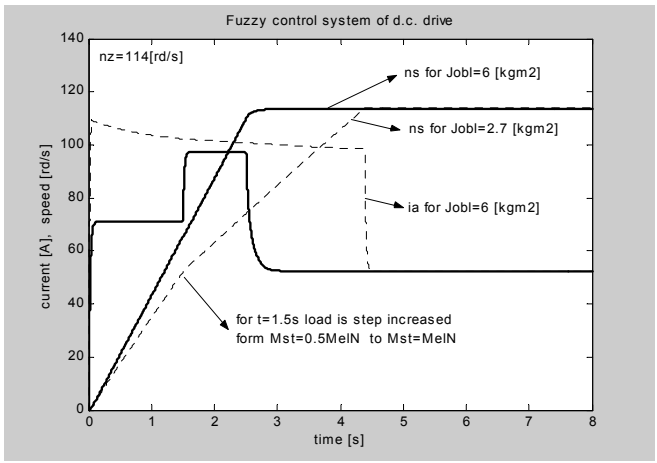

Fig.11. Time responses for d.c. drive with two Sugeno-type fuzzy PI controllers

6. Final remarks

Tuning of the fuzzy Sugeno-type PI controller is faster and easier then the fuzzy Mamdani-type PI controller. It is a great advantage of the fuzzy Sugeno-type controller that it gives the possibility to apply classical control design for the fuzzy controller.

The MATAB/*Simulink* with *NCD Blockset* and *Fuzzy Logic Toolbox* is a effective and user-friendly tool for designing and testing of a fuzzy controller of Mamdani- and Sugeno-type.

References

[1] *Fuzzy Logic Toolbox User's Guide*, The MathWorks, Inc.

[2] B. Mrozek, *Projektowanie regulatorów napędu elektrycznego w środowisku MATLAB/Simulink* III Krajowa Konferencja nt. „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim" Kraków ,19-21.11.2001 r.

[3] B. Mrozek, Z. Mrozek, *MATLAB 5.x SIMULINK 2.x poradnik użytkownika;* Wydawnictwo PLJ Warszawa 1998

[4] B. Mrozek, Z. Mrozek, *MATLAB 6 poradnik użytkownika;* Wydawnictwo PLJ Warszawa 2001

[5] *Nonlinear Control Design Blockset User's Guide* The Math Works, Inc.

[6] H. Tunia, M. Kaźmierkowski, *Automatyka napędu przekształtnikowego,* PWN 1987

[7] P. Vas *Artificial Intelligence Based Electrical Machines and Drives*, OXFORD University Press, 1999