

# Improvement of DC Motor Velocity Estimation Using a Feedforward Neural Network

**Miroslav Milovanović, Dragan Antić, Miodrag Spasić, Saša S. Nikolić, Staniša Perić, Marko Milojković**

Department of Control Systems, Faculty of Electronic Engineering  
University of Niš

Aleksandra Medvedeva 14, 18000 Niš, Republic of Serbia

E-mail: miroslav.b.milovanovic@elfak.ni.ac.rs, dragan.antic@elfak.ni.ac.rs,  
miodrag.spasic@elfak.ni.ac.rs, sasa.s.nikolic@elfak.ni.ac.rs,  
stanisa.peric@elfak.ni.ac.rs, marko.milojkovic@elfak.ni.ac.rs

---

*Abstract: This paper proposes a method for improving the DC motor velocity estimations and the estimations obtained from the state observer, when the system operates with large moments of inertia. First, the state observer for estimating velocity and DC motor position, is designed. Then, the variable structure controller is formed using estimated position and velocity values. State observer and designed controller are implemented in default system control logic. Dependences between estimated velocities and moments of inertia are established and presented by experimental results. It is noted that velocity time responses of the designed controller are not as expected when the system operates with large moments of inertia on the motor shaft. The feedforward neural network is empirically designed and implemented in control logic with purpose to solve poor velocity estimations and to improve overall system performances. It is experimentally shown that an artificial network improves estimation quality of the observer and overall control of the system for different input signals.*

*Keywords: variable structure controller; neural network; state observer; servo system; DC motor; moment of inertia*

---

## 1 Introduction

Artificial neural networks are increasingly represented in the field of power systems control [1-4], because of their ability to operate with a large number of data. A quality training procedure is a precondition for successful neural network usage. Conventional controllers in the presence of disturbances eventually do not provide a system robust enough. Both stability and robustness can be increased by introducing neural network into the default control logic of a system. An artificial neural network could be used as a compensator, whose assignment is to bring

system dynamics to desired states [5]. The combination of neural network and fuzzy logic in the form of hybrid control can also be used for the purpose of providing better system performances [6, 7]. Inteco model of a servo system based on a DC motor will be used for experimental purposes in this paper. The artificial network for DC motor velocity control is presented in [8], where neural network control logic is formed in two parts: for estimating motor velocity and for generating control signal. Another example of well-formed neural network, which successfully controls a DC motor, is shown in [9]. A state observer will be used in this paper for estimating motor velocity. The velocity is estimated by the state observer in [10], where nonlinear control input for control of serially coupled DC motors is used. Neural networks, used for motor velocity estimations, are presented in [11-13].

The starting point of this paper is a servo system, based on a DC motor and brass inertia load. For the purpose of experimental research, the state observer is designed in Section 2, as well as a variable structure controller in Section 3. Observer design procedures are presented in [14, 15]. Poor velocity estimations of a servo system are experimentally obtained in this paper and presented in Section 4. The servo system possesses control limitations while working in sliding mode with large loads attached to the motor shaft. Those limitations directly affect unsatisfied observer velocity estimations, which are also presented in Section 4. The problem is solved by introducing neural network into the default control logic of the system. The artificial network is formed and trained with real experimental data in Section 5. Significant improvements of estimated velocities for different input signals are experimentally obtained in Section 6. Both velocity offset elimination and estimated error minimization justify the neural network implementation into default system control logic.

## 2 State Observer Design Procedure

State space coordinates of a motor are necessary for the practical implementation of the variable structure controller. A state observer represents an additional system for the state space coordinates estimation of the controlled object. The state space coordinates can be obtained at any time for a known input of the object. Often it is not possible to form an ideal model, whereas unknown and immeasurable disturbances appear on a real system. The Luenberger model is used in this paper in order to solve this problem. Comprehensive details and formulation of the Luenberger model can be found in [16]. The servo system, manufactured by Inteco, Poland [17], is powered by the Bühler 1.13.044.236 DC motor, whose characteristics and parameters are given in [18]. A mathematical model of the observer can be represented by the form:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + B_0(c(t) - \hat{c}(t)); \quad \hat{c}(t) = D\hat{x}(t), \quad (1)$$

where  $\hat{x}(t)$  and  $\hat{c}(t)$  are the state space vector and the observer output, respectively. The estimated error is defined as:

$$e(t) = x(t) - \hat{x}(t) \Rightarrow \dot{e}(t) = \dot{x}(t) - \dot{\hat{x}}(t). \quad (2)$$

It is possible to neglect viscous friction and the inductance of the rotor circuit during the design procedure of the observer, because Bühler DC motor used in this paper is a small power motor. On the basis of this possibility, a differential state equation and an output equation can be represented as:

$$\begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{T_s} \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{K_s}{T_s} \end{bmatrix} u_r(t); \quad c(t) = [1 \quad 0] \begin{bmatrix} \theta \\ \omega \end{bmatrix}, \quad (3)$$

where  $K_s$  and  $T_s$  can be calculated as:  $K_s = K/\beta R + K^2$  and  $T_s = RJ/\beta R + K^2$ . Desired parameters can be calculated using the data from the engine specifications:  $a = 1/T_s = 10,526$  and  $b = K_s/T_s = 2273,68$ . The characteristic equation of the motor model is:

$$\det[SI - A] = \det \begin{bmatrix} s & 0 \\ 0 & s + a \end{bmatrix} = s(s + a) = s^2 + sa. \quad (4)$$

The poles of this system are  $s_1 = 0$  and  $s_2 = -a$  [19]. A desired range of the observer poles is such that the pole at zero will be moved to the new position  $s_1 = -20$ , while the pole at  $-a$  will be moved to  $s_2 = -22a$ . The characteristic equation of the motor now takes the form:  $(s + 20)(s + 22a) = 0$ .

The observer matrix  $A_0$  is designed using a rule:

$$\det[SI - A_0] = (s + 20)(s + 22a) = 0. \quad (5)$$

It is known that:  $A_0 = \begin{bmatrix} 0 & 1 \\ 0 & -a \end{bmatrix}$ ,  $B_0 = \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$ ,  $D = [1 \quad 0]$ , so the characteristic observer equation can be found from:

$$\det[SI - A_0] = \begin{bmatrix} s + l_1 & -1 \\ l_2 & s + a \end{bmatrix} = (s + l_1)(s + a) - (-l_2) = 0. \quad (6)$$

Values  $l_1 = 236$  and  $l_2 = 9,0385$  are calculated by equalizing coefficients from (5) and (6).

The final form of the designed state observer can be presented as:

$$\begin{aligned} \dot{\hat{\theta}}(t) &= \hat{\omega}(t) + 236(\theta(t) - \hat{\theta}(t)), \\ \dot{\hat{\omega}} &= -10,56\hat{\omega}(t) + 2273,68u_r(t) + 9,0385(\theta(t) - \hat{\theta}(t)). \end{aligned} \quad (7)$$

The experimental results obtained from the servo system are shown in Figs. 1 and 2. The estimated time response of the angular position from observer compared to the angular position of the motor is shown in Fig. 1. The time response of the estimated angular velocity of the observer is compared to the real angular motor velocity (Fig. 2). The presence of disturbances and noises, and the inability of their filtering are disadvantages of a standard linear observer. Those disturbances are present because some of them are slowly varying parameters which are not measurable. Position estimation is not significantly sensitive to the effects of disturbances and it is therefore accurate (Fig. 1). The velocity is sensitive to these disturbances, therefore, the estimation is not completely accurate. In practice, rotor current is introducing as a disturbance signal in the observer for the purpose of reducing a velocity estimation error. In our case, the motor current is not measurable parameter. As a result, it is not possible to introduce rotor current as a disturbance signal in the observer. As a result, we have velocity estimation error, which can be seen in Fig. 2.

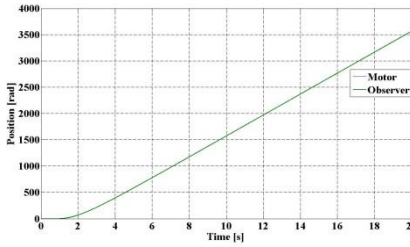


Figure 1

Time responses of estimated angular positions from observer and motor

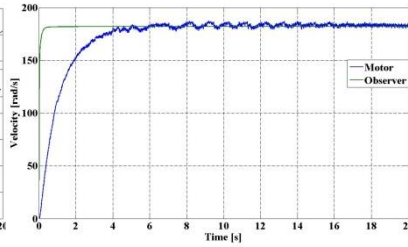


Figure 2

Time responses of estimated angular velocities from observer and motor

### 3 Variable Structure Controller

Variable Structure Control (VSC) is a control algorithm frequently used within nonlinear control systems. The main advantage of this approach is low sensitivity to parameter perturbations and disturbances, which makes it a robust control method [20, 21]. The dynamics of the second order system is represented by the following differential equations:

$$\begin{aligned}\dot{x}_1 &= x_2, \\ \dot{x}_2 &= -ax_2 + bu, \quad a > 0, b > b_0 > 0.\end{aligned}\tag{8}$$

The design procedure of VSC logic consists of two steps. The first one is establishing a reaching motion within the system trajectory, which will move towards the sliding manifold and reach it in a finite time; the second step is to keep the motion of the trajectory on the manifold as  $t$  tends to infinity.

If we choose the switching function as:

$$g = cx_1 + x_2, \quad (9)$$

the sliding manifold will be defined as  $g = 0$ .

Therefore, the motion of the system trajectory is governed by:

$$\dot{x}_1 = -cx_1. \quad (10)$$

From (10) it can be seen that the order of the sliding mode equation is less than the order of the original system, and the dynamics are determined by the parameter  $c$ . That means the dynamics during the sliding mode do not depend on the original system dynamics.

In order to provide stability for the system, the Lyapunov stability theory is used:

$$V = \frac{1}{2}g^2, \quad (11)$$

as a Lyapunov function candidate. The derivative of  $V$  is:

$$\dot{V} = g\dot{g} = g(c\dot{x}_1 + \dot{x}_2) = g(cx_2 - ax_2 + bu) = g(c-a)x_2 + bgu. \quad (12)$$

If it is assumed that the derivative of the switching function  $g$  satisfies the inequality:

$$\left| \frac{(c-a)}{x_2} \right| < \rho(x), \quad (13)$$

for some known  $\rho(x)$  functions, from (12) and (13) the following inequality is obtained:

$$\dot{V} = g\dot{g} \leq b|g|\rho(x) + bgu. \quad (14)$$

By choosing the control input signal  $u$  as:

$$u = -\psi \operatorname{sign}(g), \quad \psi \geq \rho(x) + \psi_0 > 0, \quad (15)$$

where:

$$\operatorname{sgn}(g) = \begin{cases} 1, & \text{if } g > 0 \\ 0, & \text{if } g = 0 \\ -1, & \text{if } g < 0, \end{cases} \quad (16)$$

and substituting (15) in (14), the reaching and existence condition can be expressed as:

$$\dot{V} \leq -b_0\psi_0. \quad (17)$$

We can calculate the parameter  $c$  if condition (17) is fulfilled.

If the control input is defined as a quasi-relay sliding mode control:

$$u = \psi |x_2| \text{sign}(g), \quad (18)$$

the stability, reaching, and existing conditions are calculated from:

$$\dot{V} = g\dot{g} \leq 0,$$

$$g(c\dot{x}_1 + \dot{x}_2) < g(cx_2 - ax_2 + bu) < g[(c-a)x_2] + bgu < 0. \quad (19)$$

If we substitute (18) in (19), the following inequality is obtained:

$$g[(c-a)x_2] + b\psi |x_2| |s| < 0, \quad (20)$$

and it is correct for all  $c < a$ .

The block diagram of a servo system with the observer and VSC is shown in the following Figure.

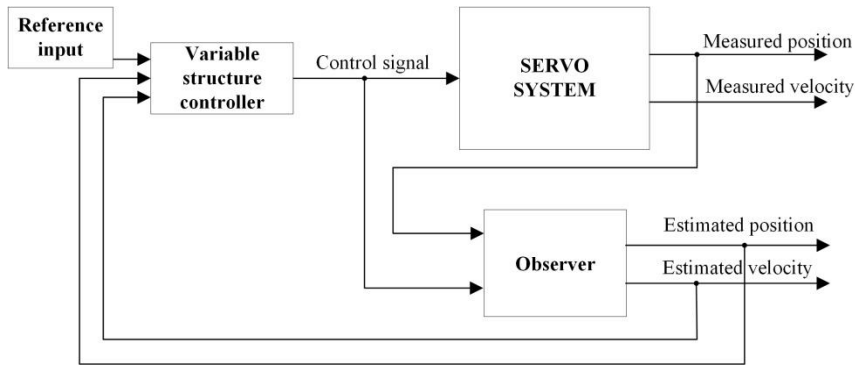


Figure 3

System block diagram with implemented state observer and variable structure controller

## 4 Servo System with Implemented VSC – Experiments and Poor Estimation Analysis

A graphic representation of the servo system used in all the experiments is shown in Fig. 4. The brass load weighing 2,030 kg, with the moment of inertia  $J_{bi} = 0,001105 \text{kgm}^2$ , is connected in Series to the shaft of DC motor.

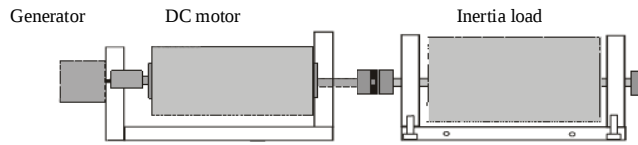


Figure 4

A servo system with brass load attached to the motor shaft

Six different step referent input values are used for experiments (Table 1). The main task is to find the system responses for different values of the proposed motor positions and to check the estimation quality. Experimental results of the servo system with implemented variable structure controller are shown in Figs. 5 to 10. From Figs. 5, 6, and 7 it can be seen that the observer precisely follows the angular position of the motor for all six input values.

Table 1  
Referent inputs

| Signal type | Specified positions — Final values of a referent input signal |                  |                 |                |                 |                 |
|-------------|---------------------------------------------------------------|------------------|-----------------|----------------|-----------------|-----------------|
| STEP        | $-25 \cdot 2\pi$                                              | $-15 \cdot 2\pi$ | $-5 \cdot 2\pi$ | $5 \cdot 2\pi$ | $15 \cdot 2\pi$ | $25 \cdot 2\pi$ |

A big deficiency of the estimated velocity from observer, is the offset appearance when the motor reaches desired positions. When the desired position is reached, Figs. 5-7, velocity should converge to zero. Analyzing Figs. 8, 9, and 10, it can be concluded that for all the input values, the observer estimates non-zero velocities after reaching the desired positions. These results are not satisfactory. The offset appearance cannot be tolerated, because it decreases observer reliability and related estimation accuracy. The second notable problem, from Figs. 8-10, is the poor estimation performance of transient processes. Requirements for optimal transient responses (13) are determined and based on [22]. The duration of transient conditions is a time period of starting, braking, and transition from one velocity to another. The time required to change the speed of the drive from  $\omega_1$  to  $\omega_2$ , when all parameters have constant values, can be calculated as:

$$t_{1,2} = J \frac{\omega_2 - \omega_1}{T - T_L}, \quad (13)$$

where  $T$  is the motor torque, and  $T_L$  is load torque.

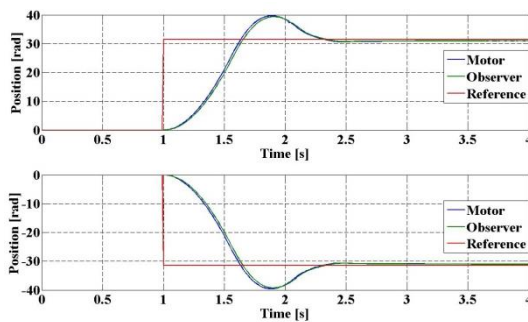


Figure 5

Estimated angular positions of observer and motor for specified motor positions:  $5 \cdot 2\pi$  and  $-5 \cdot 2\pi$

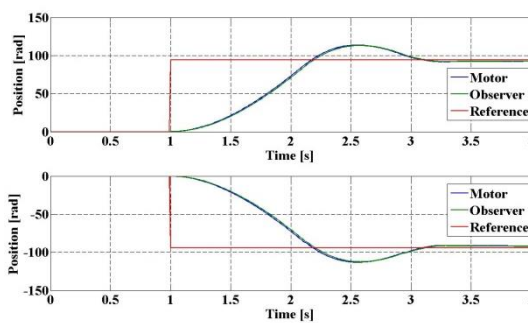


Figure 6

Estimated angular positions of observer and motor for specified motor positions:  $15 \cdot 2\pi$  and  $-15 \cdot 2\pi$

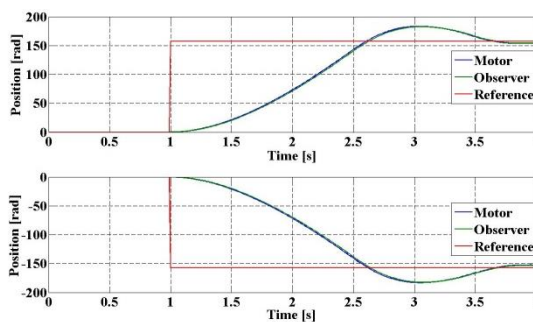


Figure 7

Estimated angular positions of observer and motor for specified motor positions:  $25 \cdot 2\pi$  and  $-25 \cdot 2\pi$

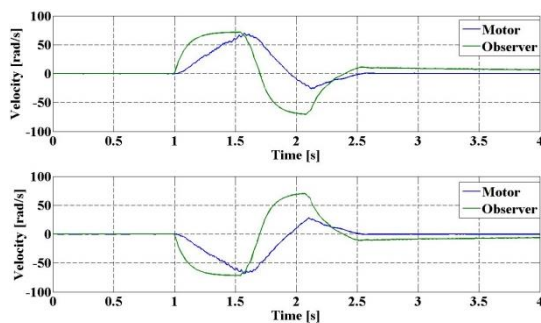


Figure 8

Estimated angular velocities for observer and motor for specified motor positions:  $5 \cdot 2\pi$  and  $-5 \cdot 2\pi$

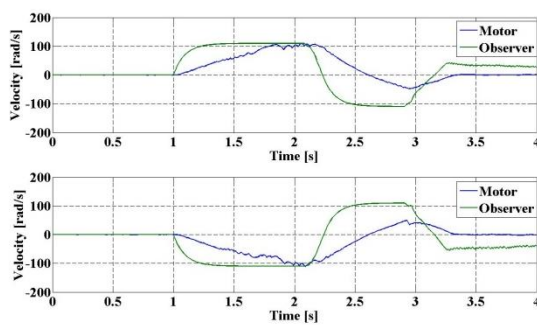


Figure 9

Estimated angular velocities for observer and motor for specified motor positions:  $15 \cdot 2\pi$  and  $-15 \cdot 2\pi$

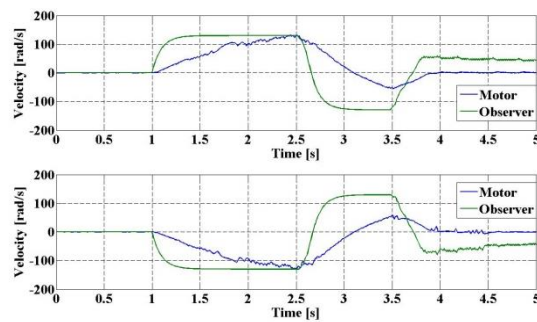


Figure 10

Estimated angular velocities for observer and motor for specified motor positions:  $25 \cdot 2\pi$  and  $-25 \cdot 2\pi$

Servo system time period  $t_{1,2}$  varies according to the desired motor position from Table 1. It can be concluded from Figs. 8, 9, and 10 that estimated observer offset is getting larger with the increase of referent input signal. The relation between the moment of inertia of motor  $J_{mot}$  and brass load inertia  $J_{bi}$  can be calculated as follows:

$$\frac{J_{bi}}{J_{mot}} = \frac{0.001105 \text{kgm}^2}{0.000018 \text{kgm}^2} \approx 61. \quad (14)$$

Large load inertia compared to the motor inertia (14), and transition from one velocity to another are two main reasons for significant changes of time period  $t_{1,2}$  (13). The transient process time period  $t_{1,2}$  increases with the increase of the attached load on the motor shaft.

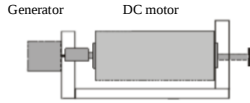


Figure 11

Servo system without load on the motor shaft

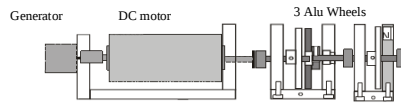


Figure 12

Servo system with aluminum wheels load

Two experiments are performed in order to show the validity of the previous analysis. In these experiments, only velocity time responses will be taken into consideration. The first experiment is based on the servo system from Fig. 11. The system is formed without any load attached to the motor shaft ( $J_l = 0$ ). VSC is included in the control logic and recorded angular velocity is presented in Fig. 13. In the second experiment, 3 aluminum wheels are attached to the motor shaft (Fig. 12). The total weight of the wheels is 0.15 kg and the moment of inertia is  $J_{aw} = 0.00008 \text{kgm}^2$ . Time response from this experiment is shown in Fig. 14. The value of referent input signal is  $25 \cdot 2\pi$  in both experiments.

The relation between moment of inertia of the DC motor ( $J_{mot}$ ) and total moment of inertia of aluminum wheels ( $J_{aw}$ ) is:

$$\frac{J_{aw}}{J_{mot}} = \frac{0.00008 \text{kgm}^2}{0.000018 \text{kgm}^2} \approx 5. \quad (15)$$

The designed observer estimates velocity in a satisfactory manner while the system operates without any load attached. The observer ability to estimate the motor velocity decreases when the load moment of inertia is attached and increased. Velocity offset is notable between 2<sup>nd</sup> and 4<sup>th</sup> second on Fig. 14, but it converges to zero after 5 seconds. In the next chapter, a neural network will be designed with the purpose to improve observer estimation performances and overall system control.

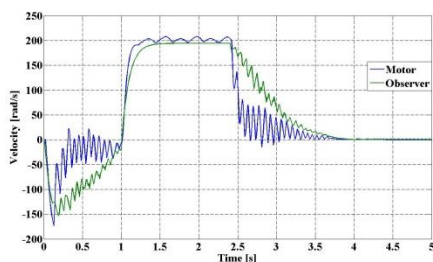


Figure 13

Experiment 1 — Estimated velocity without load torque on the motor shaft

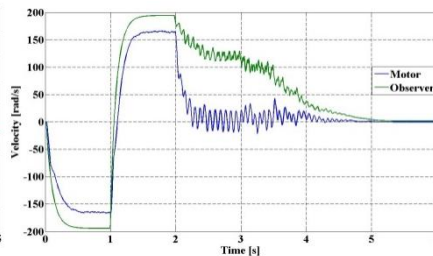


Figure 14

Experiment 2 — Estimated velocity with attached aluminum wheels to the motor shaft

## 5 Estimated Velocity Compensator Based on the Feedforward Neural Network

The block diagram of the servo system with the integrated neural network is shown in Fig. 15. A standard feedforward network is used for neural network realization. Real values from experimental model are imported for training purposes. Velocity test data from the observer and motor are used for network inputs and outputs, respectively. Four different signal types are used as referent input signals for training purposes: step, sinus, sawtooth, and square signal.

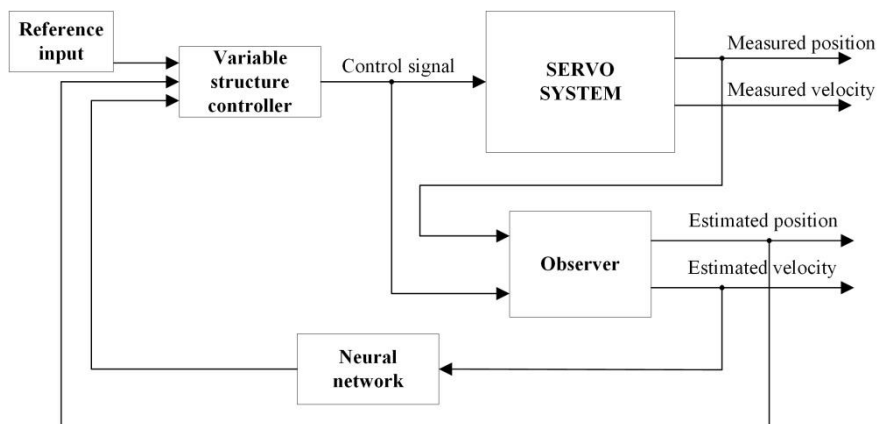


Figure 15

Modified servo system block diagram

Every input and output vector at the start of the process contained 1000 elements. Each input/output pair is obtained by performing experiments with different referent input values. Sixteen different referent input signals are used for this

purpose and they are presented in Table 2. Experiments are performed on the system with default control logic, which is presented in Fig. 3. The total number of elements of all vectors was 36000, which made the entire database large. The initial training procedures showed poor results with this database and the problem is solved with database reduction procedure.

Table 2  
Referent inputs

| Signal type | $Is_{fv}$ — Final value of a referent input signal (Specified position) |                  |                  |                  |                  |                  |
|-------------|-------------------------------------------------------------------------|------------------|------------------|------------------|------------------|------------------|
|             | $1 \cdot 2\pi$                                                          | $5 \cdot 2\pi$   | $10 \cdot 2\pi$  | $15 \cdot 2\pi$  | $20 \cdot 2\pi$  | $25 \cdot 2\pi$  |
| STEP        | $-1 \cdot 2\pi$                                                         | $-5 \cdot 2\pi$  | $-10 \cdot 2\pi$ | $-15 \cdot 2\pi$ | $-20 \cdot 2\pi$ | $-25 \cdot 2\pi$ |
| SINUS       | $25 \cdot 2\pi$                                                         | $-25 \cdot 2\pi$ |                  |                  |                  |                  |
| SAWTOOTH    | $25 \cdot 2\pi$                                                         | $-25 \cdot 2\pi$ |                  |                  |                  |                  |
| SQUARE      | $25 \cdot 2\pi$                                                         | $-25 \cdot 2\pi$ |                  |                  |                  |                  |

A reduction of elements inside every vector was performed in order to optimize the training procedure. Each new formed vector included 100 elements instead of 1000, which made the database significantly smaller. The database reduction did not make a bad influence on neural network learning procedure. The main condition for neural network implementation into control logic is to design one-input/one-output network. The real time experimental environment has only processed vectors whose length is  $m \times 1$ . Input and output vectors are formed by merging all reduced input and output vectors respectively. The number of iterations, the type of training procedure, and the number of neurons in the hidden layer are determined experimentally. Neural network activation function is selected to be default hyperbolic tangent sigmoid transfer function (tansig). Sixteen different types of training processes are used for the initial testing phase, and the training results are shown in Table 3. Fields in Table 3 that are labeled with “x” indicate unsatisfactory training procedures, due to performance divergence, extremely poor results or too slow convergence of performance factors. Those data will not be analyzed. Standard training types integrated in Matlab software [23] are applied in this paper. The decision parameter that is used for selecting training process is total error which is made during the training. The total error is presented as “Performance” category in Tables 3 and 4.

The perfect result would be the one where a training procedure has Performance index equal to 0. Seven best training types from Table 3 are selected on the basis of performance results. Those 7 training types were analyzed further, the neuron number zones which could potentially give better results are experimentally selected, and new training procedures are performed.

Table 3  
Initial training/testing phase

| Neurons:      | 5                         | 10                        | 20                        | 50                       | 100                      | 200     | 300     |
|---------------|---------------------------|---------------------------|---------------------------|--------------------------|--------------------------|---------|---------|
| Iterations:   | 1000                      |                           |                           |                          |                          |         |         |
| Training type | Performance               |                           |                           |                          |                          |         |         |
| trainbfg      | 2030.81                   | 2121.26                   | 2045.74                   | 2539.57                  | 3610.38                  | 8231    | 13293.6 |
| trainbr       | 2.35<br>*10 <sup>6</sup>  | 2.33<br>*10 <sup>6</sup>  | 2.02<br>*10 <sup>6</sup>  | 1.87<br>*10 <sup>6</sup> | 1.71<br>*10 <sup>6</sup> | x       | x       |
| trainbuwb     | x                         | x                         | x                         | x                        | x                        | x       | x       |
| trainc        | x                         | 10013.1                   | 11378.3                   | 8266.13                  | x                        | x       | x       |
| traincgb      | 2144                      | 2140.42                   | 2031.82                   | 2054.61                  | 1903.4                   | 1744.45 | 3280.25 |
| traincgf      | 2332.31                   | 2166.54                   | 1969.6                    | 2009.24                  | 2447.35                  | 3909.72 | x       |
| traincgp      | 2048.78                   | 2262.55                   | 2020.01                   | 2069.71                  | 1937.8                   | 2959    | 2698.41 |
| traingd       | 2.7<br>*10 <sup>44</sup>  | 2.7<br>*10 <sup>48</sup>  | 2.17<br>*10 <sup>51</sup> | 2.2<br>*10 <sup>57</sup> | x                        | x       | X       |
| traingda      | 2486.53                   | 2935.52                   | 4174.84                   | 7837.16                  | 16314.6                  | x       | x       |
| traingdm      | 3.66<br>*10 <sup>32</sup> | 3.01<br>*10 <sup>36</sup> | 1.17<br>*10 <sup>42</sup> | 4.7<br>*10 <sup>47</sup> | x                        | x       | x       |
| traingdx      | 2278.56                   | 2161.79                   | 15606.5                   | 3500.76                  | 8335.5                   | 12280.2 | x       |
| trainlm       | 2300.2                    | 2114.77                   | 2111.07                   | 1946.85                  | 1578.29                  | 992.037 | 946.069 |
| trainoss      | 2266.92                   | 2240.2                    | 1964.89                   | 2368.82                  | 2470.58                  | 4092.24 | x       |
| trainr        | x                         | 4782.69                   | 5338.76                   | 6632.04                  | x                        | x       | x       |
| trainrp       | 2191.89                   | 2179.37                   | 2124.28                   | 1929.01                  | 2068.66                  | 1678.16 | 2074.26 |
| trainscg      | 2100.9                    | 2102.76                   | 2015.77                   | 2185.61                  | 2093.41                  | 1748.67 | 2061.14 |

Table 4  
Final training/testing phase

| Training type | trainscg | trainrp | trainlm      | traingdx | traincgp | traincgb | trainbfg |
|---------------|----------|---------|--------------|----------|----------|----------|----------|
| Neurons       | 30       | 40      | <b>600</b>   | 7        | 30       | 15       | 3        |
| Performance   | 2106.3   | 2078.4  | <b>617.1</b> | 2549.7   | 2594.4   | 2101.1   | 2342.9   |
| Neurons       | 40       | 60      | <b>700</b>   | 12       | 40       | 30       | 8        |
| Performance   | 1995.1   | 2346.8  | <b>549.3</b> | 2151.6   | 2118.3   | 2005.2   | 2399.2   |
| Neurons       | 125      | 150     | <b>800</b>   | 16       | 70       | 40       | 15       |
| Performance   | 2277.7   | 1914.4  | <b>429.5</b> | 2309     | 2721.1   | 2108.6   | 2064     |
| Neurons       | 150      | 170     | <b>900</b>   | 18       | 80       | 60       | 20       |
| Performance   | 1733.7   | 1886.3  | <b>665.6</b> | 2282.6   | 2176.7   | 1884.1   | 2148.9   |
| Neurons       | 170      | 190     | <b>1000</b>  | *        | 90       | 80       | 100      |
| Performance   | 2398.7   | 1805.1  | <b>572</b>   | *        | 2034.6   | 1899     | 1892.9   |
| Neurons       | 190      | 230     | <b>1100</b>  | *        | *        | 120      | *        |
| Performance   | 2375.4   | 1715.1  | <b>534.3</b> | *        | *        | 2314.2   | *        |
| Neurons       | *        | 350     | <b>1200</b>  | *        | *        | 150      | *        |
| Performance   | *        | 1688.2  | <b>339.4</b> | *        | *        | 1947.7   | *        |

The results of those additional analyses are presented in Table 4. *Trainlm* is a network training function that updates weight and bias values according to the Levenberg–Marquardt optimization method. It is considered to be one of the fastest backpropagation algorithms. A deficiency of this function is the requirement of much more computer memory for its realization, in comparison to the other algorithms. The quantity of neurons in the hidden layer, the memory that is used, and the elapsed time for training processes are not considered while making the network choice. The final decision was to use 1200 neurons in the hidden layer and to use the Levenberg–Marquardt optimization training method on the basis of the results in Table 4.

## 6 Modified Servo System with Implemented VSC – Experiments

Figures 16-21 present time responses of controlled modified servo system from Fig. 15. Six step referent inputs, shown in Table 2, are used, and the system responses for each input signal are determined experimentally. The performance evaluation will be based on four parameters: a quality of the estimated angular position, an offset appearance degree, a speed of transient process, and a quality of the estimated angular velocity during the transient process.

The observer kept the estimation quality of angular positions for all input signals, as it is shown in Figs. 16, 17, 18. Figures 19, 20, and 21 show comparisons between estimated angular velocities from observer and motor. The observer velocity estimations in the steady states improved in comparison to the system responses presented in Figs. 8, 9, 10. Offset appearances problem is resolved and the errors for all the referent input signals from Table 1 were removed.

Table 5 is formed on the basis of the analysis of the transient processes before and after neural network implementation. The servo system transient process time duration is labeled as  $t_{tp\_ss}$ , the modified servo system (with implemented neural network) transient process time duration is labeled as  $t_{tp\_ss+nn}$ ,  $t_i$  represents increased time duration of transient process of the servo system after neural network implementation,  $I_{s_{fv}}$  is a symbol for the final value of referent input signal, and  $I_t(\%)$  represents increased time duration comparing systems with and without a neural network. It can be concluded that all transient process speeds decrease in the range between 25.6% and 46.2% after neural network implementation. That implies that overall system responses get slower after artificial network implementation. The influence on speed performances can be classified as a deficiency, if there is a need for obtaining faster responses.

Table 6 represents observer estimation quality comparisons between experimental results of the system before and after neural network implementation.  $Err_{ss}$  and  $Err_{ss+nn}$  represent absolute errors of the systems without and with neural network, respectively. Each error is formed as an absolute value of the difference between the velocity estimated by observer and the actual motor velocity. It must be noted that only transient process parts of signals are analyzed. Further,  $Erpr_{ss}$  represents the estimated error per second for the default servo system,  $Erpr_{ss+nn}$  the estimated error per second in the modified servo system,  $EQI$  represents the Estimation Quality Improvement after neural network implementation, and  $I_{OE}(\%)$  represents an Improvement of Observer Estimation in percentage — comparing the results before and after neural network implementation.

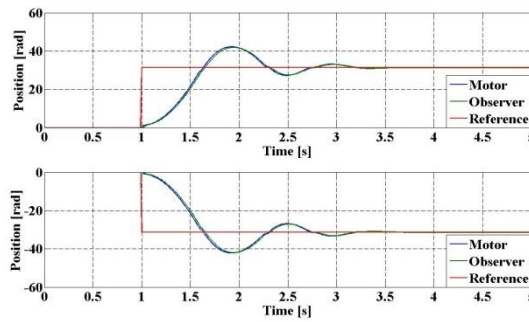


Figure 16

Estimated angular positions from observer and motor for specified motor positions:  $5 \cdot 2\pi$  and  $-5 \cdot 2\pi$ , after neural network implementation

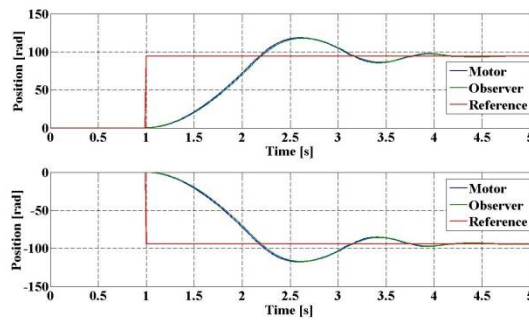


Figure 17

Estimated angular positions for observer and motor for specified motor positions:  $15 \cdot 2\pi$  and  $-15 \cdot 2\pi$ , after neural network implementation

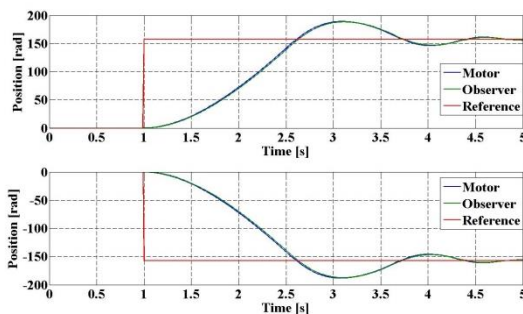


Figure 18

Estimated angular positions for observer and motor for specified motor positions:  $25 \cdot 2\pi$  and  $-25 \cdot 2\pi$ , after neural network implementation

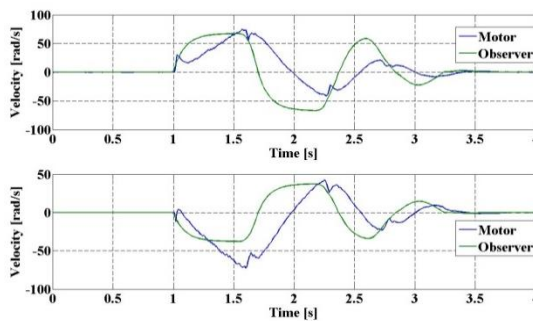


Figure 19

Estimated angular velocities for observer and motor for specified motor positions:  $5 \cdot 2\pi$  and  $-5 \cdot 2\pi$ , after neural network implementation

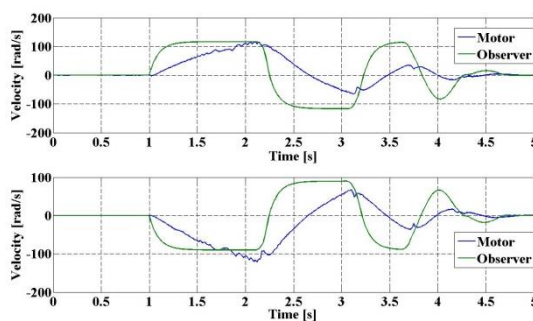


Figure 20

Estimated angular velocities for observer and motor for specified motor positions:  $15 \cdot 2\pi$  and  $-15 \cdot 2\pi$ , after neural network implementation

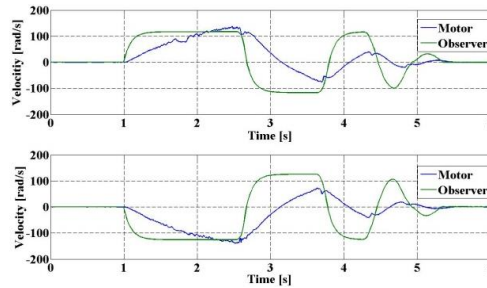


Figure 21

Estimated angular velocities for observer and motor for specified motor positions:  $25 \cdot 2\pi$  and  $-25 \cdot 2\pi$ , after neural network implementation

Table 5

Servo system transient processes time durations

| $I_{S_{fv}}$     | $t_{p_{ss}}$ (s) | $t_{p_{ss+nn}}$ (s) | $t_i$ (s) | $I_t$ (%) |
|------------------|------------------|---------------------|-----------|-----------|
| $-25 \cdot 2\pi$ | 4.45             | 5.59                | 1.14      | 25.6      |
| $-15 \cdot 2\pi$ | 3.35             | 4.90                | 1.55      | 46.2      |
| $-5 \cdot 2\pi$  | 2.55             | 3.55                | 1.00      | 39.2      |
| $5 \cdot 2\pi$   | 2.60             | 3.71                | 1.11      | 42.7      |
| $15 \cdot 2\pi$  | 3.40             | 4.91                | 1.51      | 44.4      |
| $25 \cdot 2\pi$  | 4.01             | 5.56                | 1.55      | 38.6      |

It can be concluded, from Table 6 (column  $I_{OE}(\%)$ ), that the precision of the estimation is improved in 5 experiments. A neural network improves observer performances for all negative input signal values. For positive selection of input signals, a deficiency occurs when the final value of referent input signal is  $15 \cdot 2\pi$ . For all the other experiments from Table 5, the observer improvement is in the range between 2.2% and 28.2% after implementation of the neural network. The disadvantage of the modified servo system is increased duration of the time responses, which is a regular occurrence after artificial network induction. The overall conclusion is that implementation of the standard feedforward neural network can be a satisfactory solution to improve velocity estimations of the state observer. The network can compensate an error that occurs as a result of variable structure control logic and effects of large load moments of inertia.

Table 6

Observation quality comparisons before and after neural network implementation

| $I_{S_{fv}}$     | $Err_{ss}$          | $Err_{ss+nn}$       | $Erpr_{ss}$        | $Erpr_{ss+nn}$     | <b>EQI</b> | $I_{OE}(\%)$ |
|------------------|---------------------|---------------------|--------------------|--------------------|------------|--------------|
| $-25 \cdot 2\pi$ | $23.666 \cdot 10^3$ | $28.655 \cdot 10^3$ | $5.318 \cdot 10^3$ | $5.126 \cdot 10^3$ | <b>yes</b> | <b>3.6</b>   |
| $-15 \cdot 2\pi$ | $13.826 \cdot 10^3$ | $16.990 \cdot 10^3$ | $4.127 \cdot 10^3$ | $3.467 \cdot 10^3$ | <b>yes</b> | <b>16.0</b>  |
| $-5 \cdot 2\pi$  | $4.937 \cdot 10^3$  | $4.935 \cdot 10^3$  | $1.936 \cdot 10^3$ | $1.390 \cdot 10^3$ | <b>yes</b> | <b>28.2</b>  |
| $5 \cdot 2\pi$   | $5.128 \cdot 10^3$  | $7.151 \cdot 10^3$  | $1.972 \cdot 10^3$ | $1.927 \cdot 10^3$ | <b>yes</b> | <b>2.2</b>   |
| $15 \cdot 2\pi$  | $13.566 \cdot 10^3$ | $22.459 \cdot 10^3$ | $3.990 \cdot 10^3$ | $4.574 \cdot 10^3$ | <b>no</b>  | <b>-12.8</b> |
| $25 \cdot 2\pi$  | $19.982 \cdot 10^3$ | $26.938 \cdot 10^3$ | $4.983 \cdot 10^3$ | $4.844 \cdot 10^3$ | <b>yes</b> | <b>2.8</b>   |

## Conclusions

This paper presents a method of neural network use, in the servo system control logic, for the purpose of obtaining better estimation performance. The servo system that is used is based on a separately excited DC motor. System control includes Variable Structure Control (VSC) logic. Position and velocity are estimated by the Luenberger state observer. Poor observer velocity estimations are noted and it is experimentally shown what causes poor performance. The feedforward neural network is designed to properly compensate the control signal and solve those problems. The training type and number of neurons in the hidden layer are empirically determined by comprehensive simulation procedures. Modified control logic is tested by a series of experiments.

Velocity offsets in steady states are eliminated for each experiment. Further, modified control logic significantly reduced overall velocity estimation errors. The main disadvantage of the neural network implementation is the increased time duration of the servo system processes, in comparison to default control logic performance. In general, control logic should be modified in the way demonstrated in this work, if there is need for more reliable and accurate observer velocity estimations. On the other hand, it is better to avoid feedforward neural network implementation if the speed of a system time responses is of greater importance.

## Acknowledgement

This paper was realized as a part of the projects III 43007, III 44006, and TR 35005, financed by the Ministry of Education, Science and Technological Development of the Republic of Serbia within the framework of integrated and interdisciplinary research.

## References

- [1] A. Bretas, A. Phadke: Artificial Neural Networks in Power System Restoration, IEEE Transactions on Power Delivery, Vol. 18, No. 4, pp. 1181-1186, 2003
- [2] T. Haque, M. Kashtiban: Application of Neural Networks in Power Systems, World Academy of Science, Engineering and Technology, Vol. 1, No. 6, pp. 917-927, 2005
- [3] A. Wahab, A. Mohamed: Transient Stability Assessment of Power Systems using Probabilistic Neural Network with Enhanced Feature Selection and Extraction, International Journal on Electrical Engineering and Informatics Vol. 1, No. 2, pp. 103-114, 2009
- [4] V. Vankayala, N. Rao: Artificial Neural Networks and Their Applications to Power Systems - A Bibliographical Survey, Vol. 28, No. 1, pp. 67-79, 1993

- [5] A. Ishiguro, T. Furuhashi, S. Okuma: A Neural Network Compensator for Uncertainties of Robotics Manipulators, IEEE Transactions on Industrial Electronics, Vol. 39, No. 6, pp. 565-570, 1992
- [6] A. Kusagur, S. F. Kodad, B. V. Sankar: Speed Control of Separately Excited DC Motor using Neuro Fuzzy Technique, International Journal of Computer Applications, Vol. 6, No. 12, pp. 29-44, 2010
- [7] R. E. Precup, S. Preitl: Stability and Sensitivity Analysis of Fuzzy Control Systems, Mechatronics Applications, Acta Polytechnica Hungarica, Vol. 3, No. 1, pp. 61-76, 2006
- [8] G. MadhusudhanaRao, B. SankerRam: A Neural Network Based Speed Control for DC Motor, International Journal of Recent Trends in Engineering, Vol. 2, No. 6, pp. 121-124, 2009
- [9] A. Atri, M. Ilyas: Speed Control of DC Motor using Neural Network Configuration, International Journal of Advanced Research Computer Science and Software Engineering, Vol. 2, No. 5, pp. 209-212, 2012
- [10] S. Mehta, J. Chiasson: Nonlinear Control of a Series DC Motor: Theory and Experiment, IEEE Transactions on Industrial Electronics, Vol. 45, No. 1, pp. 134-141, 1998
- [11] P. Dzung, L. Phuong: Control System DC Motor with Speed Estimator by Neural Networks, Power Electronics and Drives Systems, Vol. 2, pp. 1030-1035, 2005
- [12] N. Rai, B. Rai: Neural Network Based Closed Loop Speed Control of DC Motor using Arduino Uno, International Journal of Engineering Trends and Technology, Vol. 4, No. 2, pp. 137-140, 2013
- [13] B. Mouna, S. Lassaad: Neural Network Speed Controller for Direct Vector Control of Induction Motors, International Journal of Engineering Science and Technology, Vol. 2, No. 12, pp. 7470-7480, 2010
- [14] F. Farhani, A. Zaafouri, A. Chaari: Gain-scheduled Adaptive Observer for Induction Motors: An LMI Approach, Acta Polytechnica Hungarica, Vol. 11, No. 1, pp. 49-61, 2014
- [15] S. Yogesh, M. Gupta, M. Garg: DC motor Speed Control using Artificial Neural Network, International Journal of Modern Communication Technologies & Research, Vol. 2, No. 2, pp. 19-24, 2014
- [16] D. G. Luenberger, Introduction to Dynamic Systems: Theory, Models, and Applications, New York: John Wiley & Sons, 1979
- [17] Inteco, Krakow, Poland, Modular Servo System - User's Manual, [http://sharif.edu/~namvar/index\\_files/servo\\_um.pdf](http://sharif.edu/~namvar/index_files/servo_um.pdf)
- [18] [www.dc-motorshop.com/Motoren/DC+Motoren/artikel/1.13.044.236+-+24+V+DC+Motor.html](http://www.dc-motorshop.com/Motoren/DC+Motoren/artikel/1.13.044.236+-+24+V+DC+Motor.html)

- [19] R. E. Precup, R. C. David, E. M. Petriu, St. Preitl, M. B. Rădac: Fuzzy Control Systems with Reduced Parametric Sensitivity Based on Simulated Annealing, *IEEE Transactions on Industrial Electronics*, Vol. 59, No. 8, pp. 3049-3061, 2012
- [20] V. I. Utkin, Sliding Mode Control, in *Variable structure systems from principles to implementation*, Eds. A. Sabanovic, L. M. Fridman, S. Spurgeon, pp. 3-18, 2004
- [21] H. Khalil, *Nonlinear Systems*, New Jersey: Prentice Hall, 2002
- [22] M. Chilikin: *Electric Drive*, MIR Publishers, Moscow, 1976
- [23] H. Demuth, M. Beale, M. Hagan: *Neural Network Toolbox™ 6 - User's Guide*, 2013