International Journal of
Science, Architecture, Technology and Environment

Design and Implementation of Control Systems Using Artificial Intelligence

Geku Diton[1*]

[1]*Department of Electrical and Electronic Engineering, Federal university otuoke*

[*]*Corresponding author*

DOI: https://doi.org/10.63680/ijsate0525163.55

Abstract

This research presents a comparative analysis of artificial intelligent speed controllers and traditional dc motor speed controllers, highlighting the benefits of the latter over the former. Practical implementations of all standard controllers use data acquisition cards. Simulation findings for traditional controllers are contrasted with practical outcomes. To obtain a quick and precise speed controller, this study presents the design of an artificial neural network controller that can make a DC motor follow any arbitrarily chosen speed.

Keywords: Speed Controllers, Artificial Neural Network, Data Acquisition Card, Choppers, PWM

1. Introduction

An essential component of contemporary industrial civilisation are control systems. There are various applications all around us, and in many cases, the plant's outer mathematical model is unclear or undefined, which makes the control method's design more complicated. According to certain theories, intelligent control systems perform better in these situations. Intelligent actuators are based on artificial intelligence (AI) instead of a plant model, in contrast to traditional control methods [1].

Several basic, highly linked processing components make up an artificial neural network (ANN), a computer system that analyses data based on its dynamic state reaction to outside inputs. Because ANN models have the potential to provide answers to some of the computer-related issues that conventional serial computers have been unable to solve, their research has become more and more important in recent years [2].

This paper's proposed ANN controller demonstrates that ANNs are highly robust, capable of learning, ready to handle unexpected and incomplete input data, fast because of massive parallelism, and train rather than program; as a result, their performance may get better with practice. High-level functions like adaptation and learning, with or without supervision, are possible with ANNs [1,2].

2. DC Motor Model

Because of its straightforward and steady properties, the dc motor is the ideal test bed for sophisticated control algorithms in electric drives. Additionally, it is perfect for applications involving trajectory control. The dc motor may be seen as a SISO plant from the perspective of control systems, which removes the issues related to multi-input drive systems [3,4].

2.1 Motor dynamics

The dc motor dynamics are given by the following two equations [5]

$$v_a = R_a i_a + L_a \frac{di_a}{dt} + K_b \omega \qquad (1)$$

$$T_D = K_m . i_a = J \frac{d\omega}{dt} + F.\omega + T_L \qquad (2)$$

R_a = armature resistance Ω
L_a = armature inductance H
v_a = armature input Source Voltage v
J = rotor inertia Nm^2
F = damping constant Nm
K_m, K_b = torque & back emf constants NmA^{-1}
T_D = developed torque Nm
T_L = load torque Nm

A laboratory dc motor is utilised in this work, and all motor parameters are measured experimentally via testing; the usual values are shown below.

Ra = 10 Ω
La =0.06mH
Va = 50v.
F =0.000128 Nm
J =0.000192 Nm^2
K_m =0.153 NmA^{-1}
K_b =0.153 NmA^{-1}

2.2 Discrete time dc motor model

In order to obtain the discrete time The DC motor model Equations (1) and (2) need first undergo some modification. The discrete time domain model is as follows, and the sample period for the altered equations is (T=1ms). [1, 6].

$$\omega(k+1) = 1.844\omega(k) - 0.846\omega(k-1) + 12.198 v_a(k) \qquad (3)$$

Figure 1 displays the dc motor model's no load step response. The MATLAB software is used to produce all simulation results, with a desired speed of 1500 rpm.

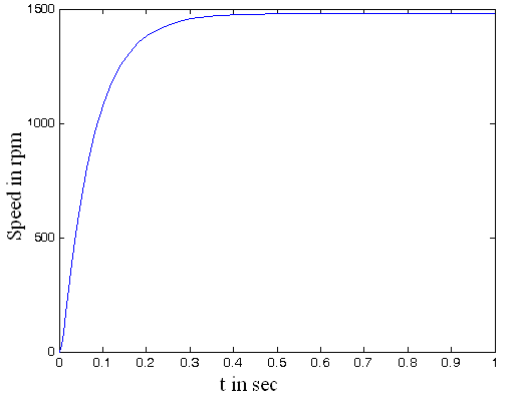


Figure 1: Simulation results of dc motor step response

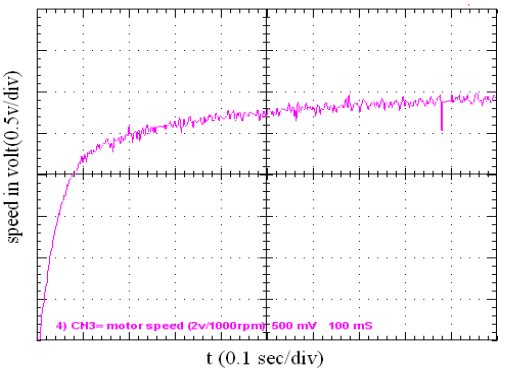


Figure 2: Practical results of dc motor step response

Figure 2 depicts the open loop system response; all practical results are obtained using a digital storage oscilloscope; all oscilloscope figures are stored while the horizontal setting is 0.1sec/div and the vertical setting is 0.5v/div; and the rotor speed sensor is a tachometer with a constant of 2v/1000rpm.

2.3 Conventional controller design

In this study, controllers are designed using the root locus method, which is considered the heart of conventional control algorithms. Simulation findings are shown alongside real outcomes to compare theoretical and experimental results. This article presents four conventional controllers: P-controller, PD-controller, PI-controller, and PID controller.

The proposed system block diagram is shown in figure 3.

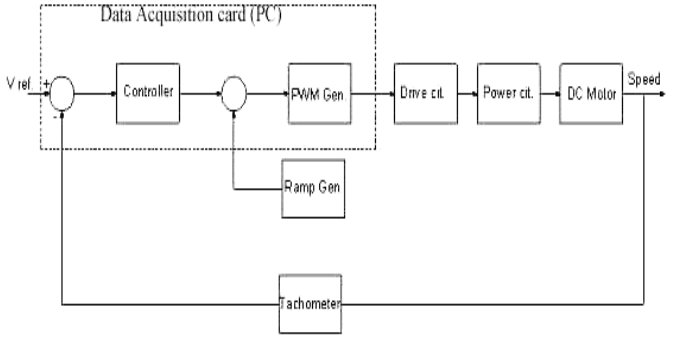


Figure 3: Proposed system block diagram

The recommended system's digital controller is the data collection card placed in the PC and developed in C++. It detects the tachometer's reference voltage and feedback signal before sending PWM pulses to the power circuit, a two-quadrant dc chopper for speed control and regeneration.

2.4 Conventional controllers Results discussion

For the compensated system with P-controller the simulation result is shown in figure 4, and the practical result is shown in figure 5.

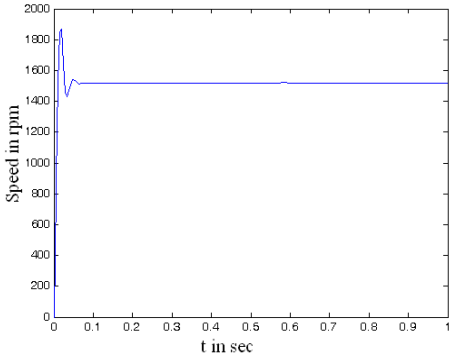


Figure 4: Simulation results for compensated system with P-controller

The simulation result shows an overshoot of 26.67%, settling time 0.05sec settling time, and a steady state error of 5%.

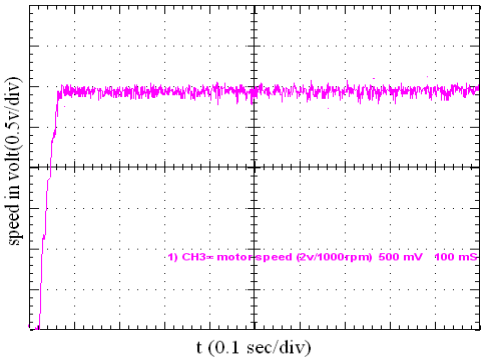


Figure 5: Practical results for compensated system with P-controller

In order to prevent a current increase from harming the armature coils, the suggested saturation limit for the controller output value provides a minimal overreach in the real system. The findings of the task has been completed and theory coincide well, as seen in Figure 6.

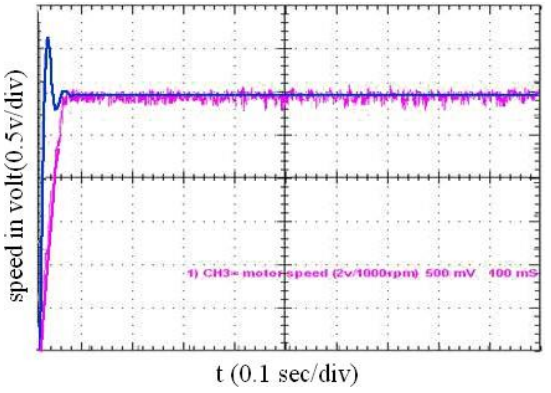


Figure 6: Response with P-controller for both simulated and actual systems

Figure 7 displays the simulation result for the compensatory system using PD-controller, whereas Figure 8 displays the real outcome.

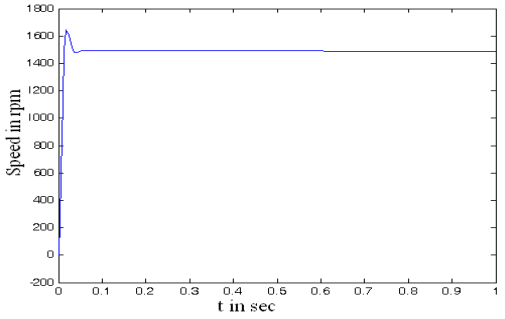


Figure 7: Simulation results for compensated system with PD-controller

The simulation result shows an overshoot of 10%, settling time 0.03sec settling time, and a steady state error of 3%.

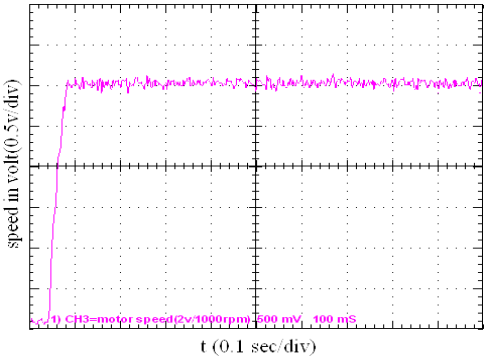


Figure 8: Practical results for compensated system with PD-controller

The practical system presents a minimum overshoot due to the proposed saturation limit. Figure 9 presents a good agreement between theoretical and experimental results.

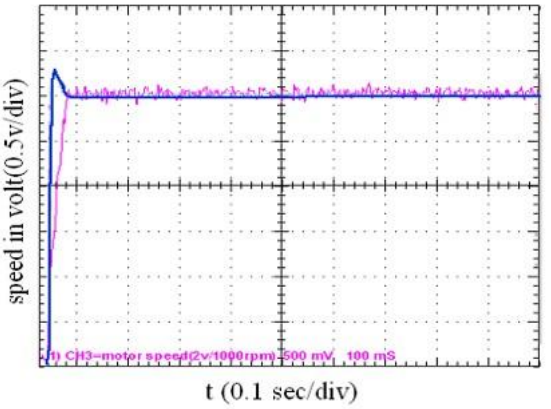


Figure 9: Response with PD-controller for both simulated and actual systems

Figure 10 displays the simulation result for the compensatory system using PI-controller, whereas Figure 11 displays the real outcome.

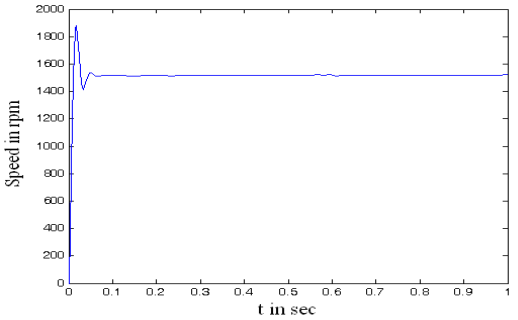


Figure 10: Simulation results for compensated system with PI-controller

The simulation result shows an overshoot of 26.7%, settling time 0.055sec settling time, and no steady state error.

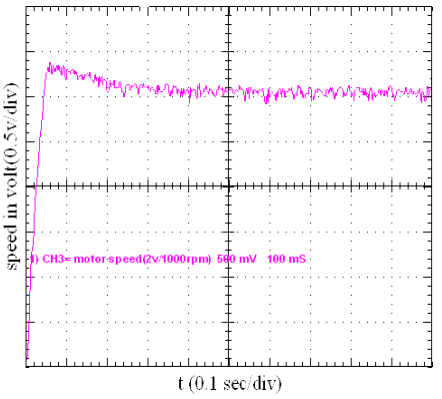


Figure 11: Practical results for compensated system with PI-controller

The practical system presents an overshoot of 11.67%. Figure 12 presents a good agreement between theoretical and experimental results.

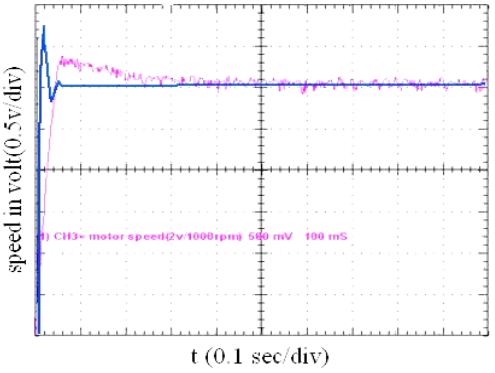


Figure 12: Response with PI-controller for both simulated and actual systems

For the compensated system with PID-controller the simulation result is shown in figure 13, and the practical result is shown in figure 14.

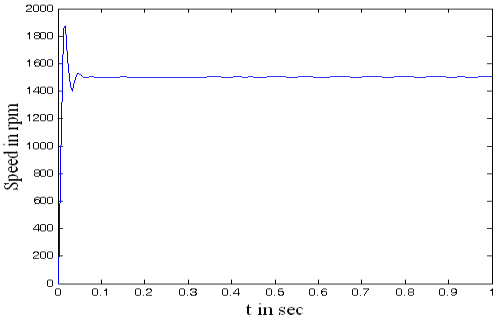


Figure 13: Response with PID-controller for both simulated and actual systems

The simulation result shows an overshoot of 23.3%, settling time 0.05sec settling time, and no steady state error.

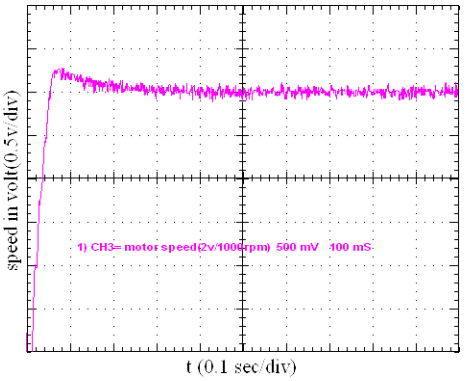


Figure 14: Response with PID-controller for both simulated and actual systems

The practical system presents an overshoot of 6.67%. Figure 15 presents a good agreement between theoretical and experimental results.

2.5 Design of Artificial Neural Network Controller

A significant amount of training data is needed to create an artificial neural network controller with the necessary accuracy and adaption speed; this training data may be obtained via the inverse model control method shown below in figure 16.

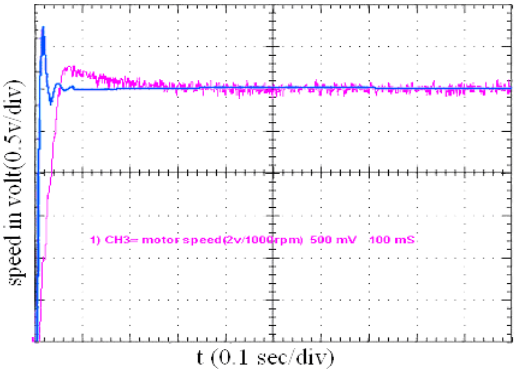


Figure 15: Response with PID-controller for both simulated and actual systems

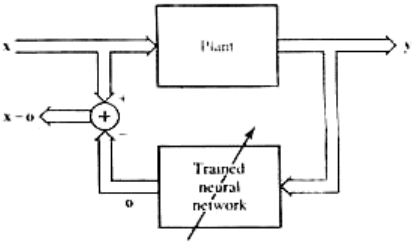


Figure 16: Inverse model control method

Figure 17 shows the suggested neural controller with three points of input and one output. The dc motor intermittent model is described by equation 5, which allows for the achievement of this input and output quantity.



Figure 17: the proposed ANN controller

By implementing an input signal x to the plant approach, the data for training can be obtained. The plant output is then used as the input data for a suggested neural network, and the amount of what is actually given to the plant is used as the intended result of the newly developed neural network. This model is depicted in figure 16.

In this paper the selected input to the neural network is

$$\omega(k+1) = 50\sin(2\pi kT/7) + 45(2\pi kT/3)$$

And the desired response is obtained after rearranging equation 3 as follows

$$v(k+1)=0.08203\omega(k+2)-0.1513\omega(k+1)+0.0694\omega(k) \quad (5)$$

Several feedforward ANN models were designed and tested in this paper. These are combination of one learning algorithm, two transfer functions and many different structures selected all others because, when compared to all other combinations that were attempted, they had the highest generalising capacity. These learning algorithms were Levenberg-Marquardt, and the pure-line and logarithmic sigmoid were the transfer functions. Table 1 displays the ANN configurations suggested in this article. With MATLAB simulation, the ANN may now be trained in accordance with the block diagram shown in figure 16. Figure 18 displays the neural network's progress during training using the Sum of the Squared Error (SSE) graph, while Figure 19 displays the error between the real and goal.

Table (1) ANN controller arrangements (T=1ms)

Layers	**Layer 1**	**Layer 2**	**Layer 3**
No. of neurons	8	6	2
Transfer function	Logistic sigmoid	Pure line	Pure line
No. of training data	30,000		
Error goal	0.00015		
Learning rate	0.04		
Momentum	0.05		
Learning algorithm	Levenberg-Marquardt		

Figure 21 displays the ANN's real reaction, whereas Figure 20 displays the targeted output's responses and a graph plotted against the quantity of epochs. Additionally, figure 22 shows the actual and goal data; it is evident from this figure that there is a significant degree of consensus between the two.

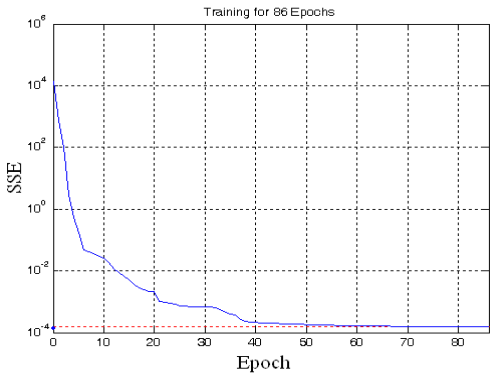


Figure 18: the sum squared error (SSE)

Figure 23 shows the proposed ANN controller after fitting it to the dc motor reference model. Figure 24 shows the response of the actual ANN to a desired speed of (Wd =150 rad/s), which is shown by the dashed line in figure 24, the actual speed

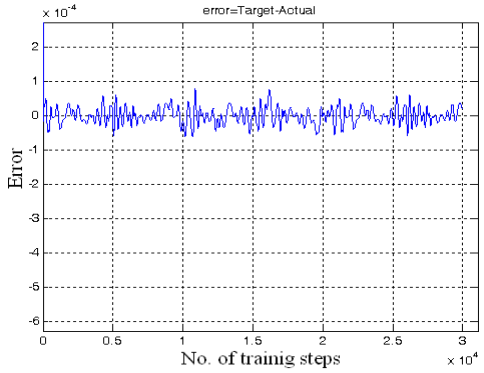


Figure 19: the error (target-actual)

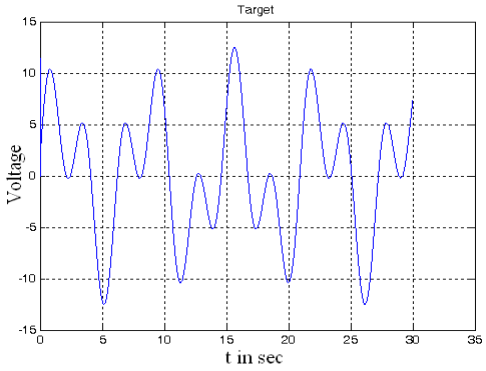


Figure 20: the target signal

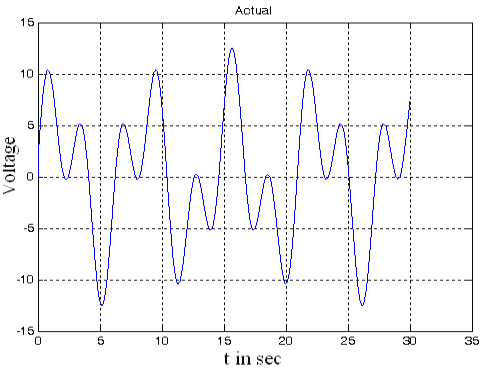


Figure 21: the actual signal

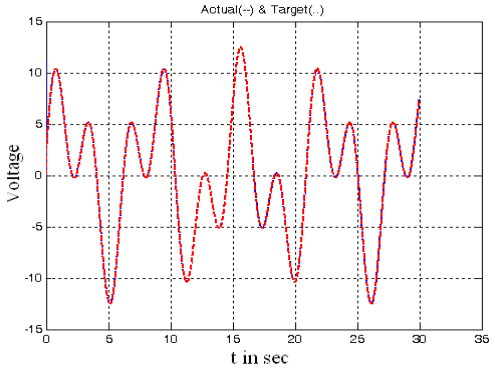


Figure 22: target signal, and actual

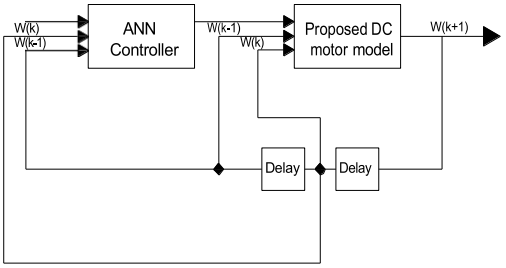


Figure 23: the proposed ANN controller configuration

achieves the target speed with a 20% overestimation, zero errors in the steady state, and a settling period of 0.18 seconds (for ±5% of the intended output).

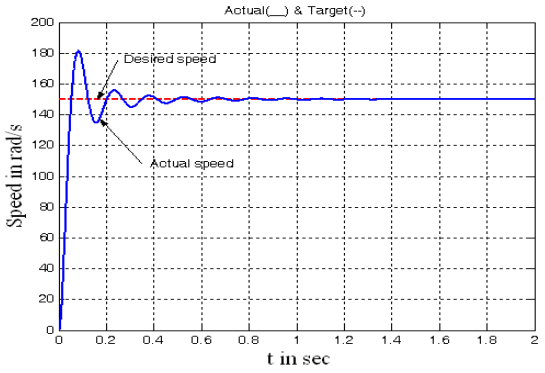


Figure 24: Overall system response

Table 2 compares the results of the inverted model the controller for ANN figure 24 with the results of the conventional techniques.

Table (2) Comparison between conventional and intelligent controllers

	Open loop	**P**	**PD**	**PI**	**PID**	**ANN**
Settling time(s)	0.7	0.02	0.01	0.07	0.05	0.18
Over shoot(%)	-	15.3	18	20.7	23.3	20
Rise time(s)	-	0.02	0.05	0.02	0.02	0.025
e_{ss} (%) for 50% load	large	5	5	0	0	0

Conclusion

1. This research compares and contrasts artificial neural network controllers with traditional controllers.
2. The simulation findings make use of a real-world DC motor. Comparing experimental data demonstrates the excellent correspondences between experimental and theoretical findings.
3. Using the suggested dc motor model, the ANN controller presented in this research has been tested to follow any randomly chosen speed.

Declaration of Conflicting Interests

The authors declare no potential conflicts of interest with respect to the research, authorship and publication of this article.

Funding

The author received no financial support for the research, authorship and publication of this article.

References

[1] Omar M.A. Salim, 'Analysis and Design of Control Systems Using Artificial Intelligence', M.Sc Thesis, Banha University 2006.
[2] Ying Wu, Zhiqiang Lv, Dabin Hu, Lin He, Yingyun Huang, Shijian Zhu, and Dingfang Chen, 'The BP Neural Network Intelligent Evaluation System Of Raft Vibration-isolating Unit Design', Proceeding of the 2002 International conference on control and automation, Xiamen, China, June 2002.
[3] Siri Weerasooria, and M. A. El-Sharkawi 'Identification And Control Of DC Motor Using Back Propagation Neural

Networks', IEEE Trans. On Energy Conversion, vol. 6, No. 4, December 1991.

[4] Carmadi Machbub, Ary Setijadi Prihatmanto, Yoseph Dwi Cahaya: 'Design And Implementation Of Adaptive Neural Networks Algorithms For DC Motor Speed Control System Using Simple Microcontroller', IEEE PEDS 2001-Indonesia.

[5] M. D. Minkova, D. Minkov *, J. L. Rodgerson, R.G. Harly , 'Adaptive neural speed controller of a DC motor', ELSEVIER, Electric Power Systems Research 47(1998) 123-132.

[6] Hassanzadah, S. Khanmohammadi, M.B.B. Sharifian, and J. Jiang, 'A SISO Discrete Control System Containing Neural Estimator And Neural Controller', IEEE 2000