MODEL REFERENCE ADAPTIVE CONTROL (MRAC) SCHEME FOR ELIMINATING OVER SHOOT IN DC SERVOMOTOR

Okafor Patrick U., Department of Electrical and Electronic Engineering Department, Faculty of Engineering, Enugu State University of Science and Technology (ESUT), Enugu State, Nigeria

Eneh Princewill Chigozie, Department of Electrical and Electronic Engineering Department, Faculty of Engineering, Enugu State University of Science and Technology (ESUT), Enugu State, Nigeria

Arinze S. N., Department of Electrical and Electronic Engineering Department, Faculty of Engineering, Enugu State University of Science and Technology (ESUT), Enugu State, Nigeria

Abstract: *This paper presents a way of eliminating overshoot in a DC servomotor using a model reference adaptive control system, which is a controller in the Neural Network (NN) toolbox. A Speed and Position based MRAC system and a Proportional Integral Derivative (PID) controller were used to control a physical DC servomotor with known parameters. This was achieved through physical measurements by configuration of the servo driver, the computer system and the DC motor using, Simulink toolbox and Microsoft Windows Software Development kits (SDKs) 7. The reference input to the system was a step function input. The purpose is to determine the time-response of the developed system also, the stability of the system and its ability to reach one stationary state when starting from another. Results from physical measurements show that MRAC system was able to accommodate nonlinearities associated with DC motor and yet, maintain good control of the motor without voltage overshoot as against the PID controller.*

Keywords: *Artificial Neural Network (ANN), ANN Inverse Model (AIM), DC Motor , Model Reference Adaptive Control (MRAC), Overshoot, Proportional Integral Derivative (PID)*

1.0 INTRODUCTION

The most common actuator in control systems is a Direct current (DC) motor, apparently the choice for implementation of advanced control algorithms in most electric drives. Lately, developments in magnetic materials, microprocessors, semiconductor technology and mechatronics etc. had provided a wide range of applications for high performance electric motors in various industrial and automated processes. For high performance drive applications such as robotics, machine tools, conveyors, rolling mills, etc., high precision in speed and position control is paramount. DC servomotors are mostly the choice; they are widely deployed in these applications due to their reliability and ease of control because of the decoupled nature of the field and the armature magneto motive force (Sheel, Chandkishor, and Gupta, 2010).

DC Servomotor systems have two outputs that can be controlled, angular speed and angular position. Improving the system's efficiency comes from the proper control of both outputs together, or by controlling one of them at a time and, hence, two identification plants can be derived for each one. For some applications, such as disk drives and robotics, position control is more important than speed control (Makableh, 2011). One of the parameters that negatively affect the efficient control of DC servomotor is overshoot. In the context of control theory, overshoot can be regarded as an output exceeding its final steady state value. Overshoot could be seen as a form of distortion that affects the rise time, settling time etc. of an operating plant under step input response. Reviews show that conventional controllers such as Proportional Integral Derivative (PID) are not that capable of handling nonlinearities associated with DC motors and at the same time, mitigate the effects of overshoot.

Thus, one of the drawbacks of conventional tracking controllers for electric drives is that they are unable to capture the unknown load characteristics over a widely ranging operating point. Apparently, this makes tuning of controller parameters very difficult. There are many ways to overcome these difficulties but, generally there are four basic way that are common to adaptive controller; (1) Model reference adaptive control (MRAC), (2) Self tuning, (3) Dual control and (4) Gain scheduling. Usually load torque is a nonlinear function of a combination of variables such as speed and position of the rotor. Therefore, identifying the overall nonlinear system through a linearized model around a widely varying or changing operating point, under fast switching frequencies, can introduce errors which can lead to unstable or inaccurate performance of the system (Astron and Wittenmark, 1989).

2.0 THEORY

2.1 Basic Operation of DC Motors

The dc motor is a torque transducer that converts electric energy into mechanical energy. It has the electrical and the mechanical representation. The torque developed on the motor shaft is directly proportional to the field flux and the armature current. To substantiate that, assume a current-carrying conductor is established in a magnetic field with fluxϕ, and the conductor is located at a distance r from the center of rotation. The relationship among the developed torque, flux ϕ, and current i_a is

$$T_m = k_m \phi i_a. \qquad (1a)$$

Where;

T_m= the motor torque (in N-M), ϕ, = the magnetic flux in (in webers), i_a = the armature current (in amperes), k_m= a proportional constant. In addition to the torque developed, when the conductor moves in the magnetic field, a voltage is generated across its terminals. This voltage is known as the *back emf***,** which is proportional to the shaft velocity, and tends to oppose the current flow. The relationship between the back emf and the shaft velocity is:

$$e_b = k_m \phi w_m. \qquad (1b)$$

Where;

e_b= the back emf (in volts), w_m= the shaft velocity of the motor (in rad/sec). Equations 1a and 1b form the fundamentals of the dc-motor operation.

2.1 Model Reference Adaptive Control (MRAC)

MRAC is an adaptive servo system in which the desired performance is expressed in terms of a reference model, which gives desired response to the reference signal. Generally speaking, MRAC is composed of four parts namely; the plant containing unknown parameters, a reference model for compactly specifying the desired output of the control system, a feedback control law containing adjustable parameters. The adaptation law of MRAC systems extracts parameter information from the tracking errors. The online computation of this controller, like NARMA-L2, is minimal. However, unlike NARMA-L2, the model reference architecture requires that a separate neural network controller be trained offline, in addition to the neural network plant model. The controller training is computationally expensive, because it requires the use of dynamic backpropagation (Beale, Hagan, and Demuth, 2015). On the positive side, model reference control applies to a larger class of plant than does NARMA-L2 control. Fig. 1 shows an illustration of a neural network (NN) MRAC system.

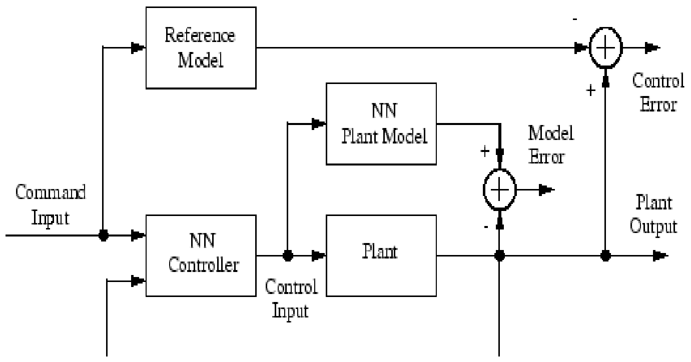

Fig 1 Block diagram of Model Reference Adaptive Control

Recently, artificial neural networks represent an effective alternative for industrial process modeling and control mainly because they can learn nonlinear input/output mapping from process data (Narendra and Parthasarathy, 1990). Perhaps, it has been emphasized that complete controllability and observability of the process must be assumed for successful neural network modeling and control (Saerens and Soquet,1991). More so Narendra and Parthasarathy (1990) indicated that considerable progress in nonlinear control theory is still needed to obtain rigorous solutions to identification and control problems using neural networks.

2.2 PID Control Systems

The PID controller is a form of close loop system i.e. with a feedback mechanism. The PID control system is achieved by tuning and calculating the error signal between an output measured value and a reference value (input), the controller works to minimize the error signal or the difference between the output signal and the reference signal to a minimum value; such that the output measured value will be as close as possible to the input reference signal.

The mathematical model of the PID controller has been proposed by many authors and is represented by:

$$U(t) = Kp.e(t) + Ki\int_0^t e(\tau)d\tau + Kd\frac{d}{dt}e(t) \qquad (2)$$

Where:

$U(t)$ is the controller output signal, $e(t)$ is the error signal, Kp is the proportional gain, *Ki* is the integral gain and *Kd* is the derivative gain.

3.1 Modeling the Permanent DC Motor

DC motors are used extensively in control systems especially in industrial actuators so, it is paramount to establish mathematical model for analytical purposes for efficient control application of dc motors. The DC motor takes in single input in the form of an input voltage and generates a single output parameter in the form of output speed. It is a single-input, single-output system (SISO). Figure 2 is the electromechanical representation of a DC motor, the diagram is used to develop the system level transfer function that characterize the operation or behavior of a DC motor.

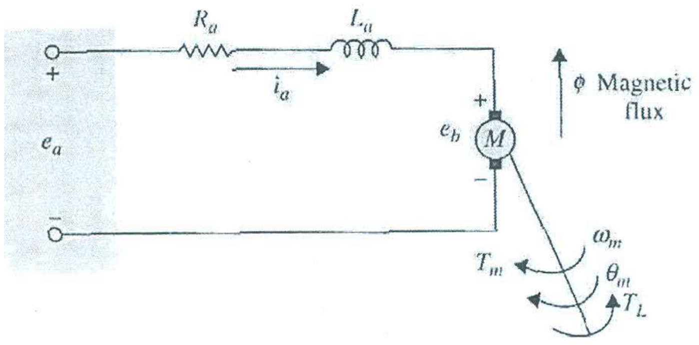

Fig 2 Electrical Model of DC Motor

The armature is modeled as a circuit with resistance R_a connected in series with an inductance L_a and a voltage source e_a, and e_b representing the back electromotive force (emf) in the armature when the rotor rotates. Looking at the diagram of fig 1, it can be seen that the control of the dc motor is applied at the armature terminals in the form of applied voltage e_a (t). It can be deduced that the torque developed in the motor is proportional to the air-gap flux and the armature current. The equations that describes the DC servomotor behavior as stated by (Kuo and Golnaraghi (2002) thus,

$T_m(t) = K_m(t)\,\phi\, i_a(t)$ (3)

Since ϕ is constant, equation 3 is in form

$T_m(t) = K_i\, i_a(t)$

$K_i\, i_a(t) = i_m\omega_m + b\omega_m + T_L$ (4)

Where:

$T_m(t)$=motor torque. $i_a(t)$ = armature current. $T_L(t)$ = load torque. Φ= magnetic flux in the air gap. k_m = proportionality constant. K_i = torque constant in N-m/A. $\omega_m(t)$=rotor angular velocity i_m = equivalent moment of inertia reflected at the motor shaft . Putting the control input voltage e_a (t) into consideration, the cause and effect equations for the motor circuit in same fig 4 are:

$$\frac{di_a(t)}{dt} = \frac{1}{L_a}\, e_a(t) - \frac{R_a}{L_a}\, i_a(t) - \frac{1}{L_a}\, e_b\,(t). \quad (5)$$

$$e_b(t) = k_b\, \frac{d\theta_m\,(t)}{dt} = K_b\omega_m(t). \; (6)$$

$$\frac{d^2\theta_m(t)}{dt^2} = \frac{1}{J_m} T_m(t) - \frac{1}{J_m}\, T_L(t) - \frac{B_m}{J_m}\, \frac{d\theta_m\,(t)}{dt}. (7)$$

Where:

$l_a(t)$ = armature inductance

R_a = armature resistance. *$e_a(t)$* = applied voltage

$e_b(t)$ = back emf. *K_b* = back emf constant

$\omega_m(t)$= rotor angular velocity. *B_m* = viscous-friction coefficient. θ_m (t)= rotor displacement

J_m=rotor inertia. From equations 3 through 6, the applied voltage *$e_a(t)$* is considered as the cause and Equation 5 considers that $\frac{di_a(t)}{dt}$ the immediate effect due to the applied voltage. From Equation 3, armature current *$i_a(t)$* causes the motor torque $T_m(t)$, while in Equation 6 the back emf $e_b\,(t)$ was defined. It can be seen also from Equation 7 that the motor torque produced causes the angular velocity $\omega_m\,(t)$ and displacement θ_m (t) of the rotor respectively.

The state variables are of the system can be define as

- Armature current = *$i_a(t)$*
- Rotor angular velocity = $\omega_m\,(t)$
- Rotor angular displacement =θ_m (t)

It is possible to eliminate all the non-state variables from Equation 3 through 7 by direct substitution then present the dc state equation in vector-matrix form as follows:

$$\begin{bmatrix} \frac{di_a}{dt}\,(t) \\ \frac{d\omega_m\,(t)}{dt} \\ \frac{d\theta_m\,(t)}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} - & \frac{K_b}{L_a} & 0 \\ \frac{K_i}{J_m} - & \frac{B_m}{J_m} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} i_a\,(t) \\ \omega_m\,(t) \\ \theta_m\,(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_a} \\ 0 \\ 0 \end{bmatrix} e_a\,(t) + \begin{bmatrix} 0 \\ -\frac{1}{J_m} \\ 0 \end{bmatrix} T_L\,(t)^{\cdot} \quad (8)$$

Note that in the case of Equation 8 above, that $T_L(t)$ is handled as a second input to the state equations. The transfer function between the motor displacement and the input voltage is obtained as thus;

$$\frac{\Theta_m(s)}{E_a(s)} = \frac{k_i}{L_a J_m s^3 + (R_a J_m + B_m L_a)s^2 + (K_b K_i + R_a B_m)s} \qquad (9)$$

Note that T_L has been set to zero in Equation 9. Fig 2 shows a block diagram of the DC motor system for speed control. From the diagram, one can see clearly how the transfer function is related to each block. It can be seen from Equation 9 that *s* can be factored out of the denominator and the significance of the transfer function $\frac{\Theta_m(s)}{E_a(s)}$ is that the dc motor is an integrating device between these two variables. $\Theta_m(s)$ is the rotor angular displacement Laplace transfer function, $E_a(s)$ is the input voltage Laplace transfer function and $\Omega_m(s)$ is the transform of angular velocity respectively. From fig 3 also, it can be seen that the motor has a built-in feedback loop caused by the back emf Eb.

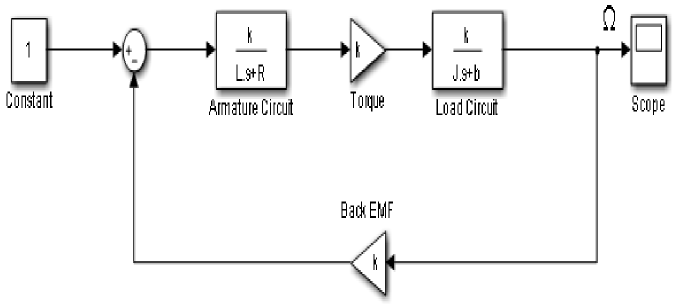

Fig 3 Simulink Model of a DC Servomotor in terms of speed

The back-emf physical represents the feedback of a signal that is proportional to the negative of the speed of the motor. From equation 9, it can be noted that back emf constant K_b represents an added term to the resistance R_a and the viscous-friction coefficient B_m . Effectively, the back-emf effect is equivalent to an electric friction which tends to improve the stability of the motor and apparently the stability of the system.

3.2. DC Motor Equivalent Circuit In Discrete Form

Recall that ANN is the modeling tool. So simulation can be performed on the control of DC motor using ANN model, there is need to construct an equivalent DC motor to a discrete time model. Effectively, the load torque is assumed as:

$T_L = \mu\omega^2 m(t)[sgn(\omega_m(t))]$ (10)

Where μ = a constant

It is obvious that from equation (10) that load torque is always opposes the direction of motion. Note that the choice of load torque here is arbitral because considering load torque as one of the functions of a DC motor; it is a common characteristic for most propeller driven loads.

Alternatively,

Direct substitution and substitute for position in equations (4), (5) and (10) i.e. rewriting the angular velocity $\omega_m(k)$, which is the speed in terms of angular displacement θ_m which is the position as shown in figure 4.

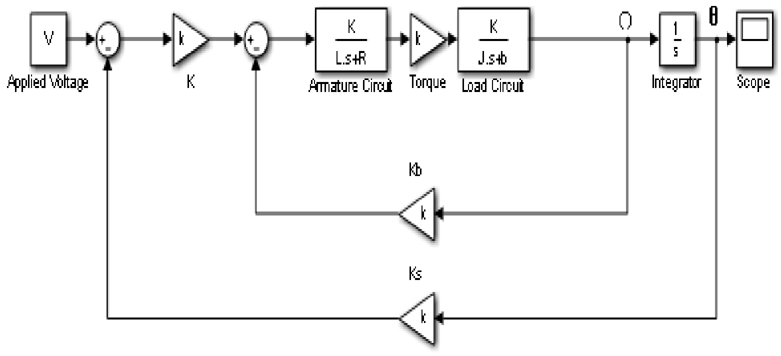

Fig 4 Simulink Model of a DC Servomotor in terms of Speed and Position

Deploying direct substitution and substitute for position in equations (4), (5) and (10) i.e. rewriting the angular velocity $\omega_m(k)$, which is the speed in terms of angular displacement θ_m which is the position as shown in figure 3. Then the equations yields;

$$k_e\theta_m = -R_a i_a - L_a\frac{di_a}{dt} + v_a. \qquad (11)$$

$$k_e i_a = I_m\theta_m + b\theta_m + T_L \qquad (12)$$

$$T_L = \mu\left(\frac{d\theta_m}{dt}\right)^2[sgn(\theta_m)]. \qquad (13)$$

Next is to estimate the derivatives of position and current in discrete form using a sampling interval of ΔT and forward difference.

$$\frac{d\theta_m(k)}{dt} = \frac{\theta_m(k+1) - \theta_m(k)}{\Delta T}. \qquad (14)$$

$$\frac{d^2\theta_m(k)}{dt^2} = \frac{\frac{d\theta_m(k+1)}{dt} - \frac{d\theta_m(k)}{dt}}{\Delta T}. \quad (15)$$

$$\frac{d^3\theta_m(k)}{dt^3} = \frac{\frac{d^2\theta_m(k+1)}{dt^2} - \frac{d^2\theta_m(k)}{dt^2}}{\Delta T} \quad (16)$$

$$\frac{di_a}{dt} = \frac{i_a(k+1) - i_a(k)}{\Delta T}. \quad (17)$$

$$T_L(k) = \mu \left(\frac{d\theta_m(k)}{dt}\right)^2 [sgn(\theta_m(k)]. \quad (18)$$

$$\frac{dT_L(k)}{dt} = \frac{T_L(k+1) - T_L(k)}{\Delta T}. \quad (19)$$

Next is to evaluate the armature current i_a in terms angular displacement θ_m which is the position here. Then substituting i_a in terms of θ_m using equations (11) into the following equation $\frac{K_e{}^2(\Delta T)^2 + R_a(b\Delta)^2}{(L_a I_m + R_a I_m \Delta T)}$ from the work of Weerasooriya and El-Sharkawi (1991), to determine the function governing the speed control of a DC motor, gives

$$k_e\dot{\theta}_m = -\frac{R_a}{k_e}\left[I_m\ddot{\theta}_m + b\dot{\theta}_m + T_L\right] - \frac{L_a}{k_e}\left[I_m\dddot{\theta}_m + b\ddot{\theta}_m + T_L\right] + v_a. \quad (20)$$

Or

$$v_a = \frac{L_a I_m}{k_e}\dddot{\theta}_m + \left[\frac{R_a I_m}{k_e} + \frac{L_a b}{k_e}\right]\ddot{\theta}_m + \left[\frac{R_a b}{k_e} + k_e\right]\dot{\theta}_m + \left[\frac{R_a}{k_e} T_L + \frac{L_a}{k_e} T_L\right] \quad (21)$$

Integrating equations (12), (13), (14), (15), (16) and (17) into equation (19) then the input voltage in equation (19) can be written as a function of:

$\theta_m(k+1)$.

$\theta_m(k)$.

$\theta_m(k-1)$.

$\theta_m(k-2)$.

$$v_a(k) = g[\theta_m(k+1), \theta_m(k), \theta_m(k-1), \theta_m(k-2)] \quad \textbf{(22)}$$

Effectively, equation (22) forms the relationship between the input voltage v_a and the motor position θ_m at four successive sampling instances. Assume that the term $\theta_m(k+1)$ is replaced in equation (22) with desired reference motor position at next instance as $\theta_d(k+1)$, and compute the control voltage (the input voltage) $v_a(k)$ with the following equation, then

$$v_a(k) = g[\theta_d(k+1), \theta_m(k), \theta_m(k-1), \theta_m(k-2)]. \quad \textbf{(23)}$$

So, if the computed voltage $v_a(k)$ at sampling instance *k* is applied, then the resulting motor position at instant $(k+1)$ will be equal to:

$\theta_m(k+1) = \theta_d(k+1)$, i.e. the desired motor position, it effectively takes the following forms of input to the ANN

$\theta_d(k+1)$. = the reference position

$\theta_m(k)$. = Position at first instance

$\theta_m(k-1)$. = Position at second instance

$\theta_m(k-2)$. Position at third instance

4.1 Structure of the ANN controller

The non-linear controller for this work is the Artificial Neural Network (ANN) Controller. Here, the design of ANN incorporated a Feed forward Neural Network (FFNN), which is made up of one input layer and one hidden layers with an output layer. The layers respectively consist of number of neurons. Each neuron has two functions as:

- Summing up all the outputs from the previous layers multiply by the corresponding weights
- Performing the nonlinear sigmoidal or linear function on the sum

During training, errors are back propagated and also minimized using least mean square algorithm. The basis for weights connection between the input and hidden layers are based on the fact that errors in the output determine the measures of the hidden layer output errors. This adjustment of weights between the layers and recalculating the output in an iterative process is continued till the error falls below a tolerable level.

4.2 Training the ANN Controller

The training data was chosen taking into consideration the mechanical and hardware limitations of the motor. And that lead to the following constrained operating range as defined below (Weerasooriya and El-Sharkawi,1991):

$\omega(k)-\omega(k-1)<0.1$

$\omega(k)<30.0$ rad/s

$v(k)<100.0$v

The corresponding target voltage was then generated by using these speed values. The offline training of the controller is performed with this training data using the back-propagation algorithm. The back-propagation training algorithm is based on the principle of minimization of a cost function of the outputs and the target of the FFNN.

The input and output of this network is guided by some basic equations, the net input of the j^th neuron of the hidden layer at the time instant n is given as follows (Haykins, 1999):

$$S_j^h(n) = \sum_{I=1}^{N} W_{ij}^h(n) I_i(n). \qquad (24)$$

Where W_{ij}^h is the connecting weight between the ith neuron at the input layer and the jth neuron at the hidden layer. The I_i is the ith input, and *N* is the number of inputs. Then, the output from the jth neuron from the hidden layer at nth instant is given by:

$$O_j^h(n) = f^h[S_j^h(n) + B_j^h(n)] \qquad (25)$$

From equation 25, B_j^h is the bias of the jth neuron and f^h is the activation function acting on each neuron at the hidden layer. The activation function can be tan sigmoidal, log sigmoidal or linear. The functions are described as follows (Beale, Hagan and Demuth, 2015):

$$tansig(x) = \frac{1-e^{-2x}}{1+e^{-x}} \qquad (26)$$

$$logsig(x) = \frac{1}{1+e^{-x}} \qquad (27)$$

$$linear(x) = x \qquad (28)$$

In the above equations x represents the input to the activation function. It follows that the net input of the *k*th neuron of the output layer at time instant n is given by:

$$S_k^o(n) = \sum_{j=1}^{M} w_{jk}^o(n)\, O_j^h(n) \qquad (29)$$

Where *M* is the number of neurons in the hidden layer and $w_{jk}^o(n)$ is the weight between the *j*th neuron at the hidden layer and *k*th neuron at the output layer respectively. It therefore also followed that output from the k^{th} neuron at the output layer at time instant n can be presented in the form:

$$O_k^o(n) = f^o[S_k^o(n) + B_k^o(n)] \qquad (30)$$

Where f^o = the activation function of the output layer and

$B_k^o(n)$ = the bias of the k^{th} neuron at the output layer.

It is also important to put into consideration how the weight is updated at various levels during the network training. To do that, there is a basic equation that describes the updating of the weight through the error signal at the output of the neuron *k* at the iteration and it is given as follow:

$$e_k(n) = d_k(n) - O_k^o(n) \qquad (31)$$

Where $d_k(n)$ represents the desired output for neuron *k*.

5.0 ANN MODEL OF DC MOTOR

Before ANN can be used to control the operation of the DC motor, the DC motor being the plant must also be modeled using ANN tool. From equation 23 where $v_a(k)$ is a function of speed at successive time intervals k+1, k and k-1 for any required trajectory, what happens is that the ANN Inverse Model (AIM) generates an output that is proportional to the voltage required at the input of the DC motor to produce these speed at the time intervals. Here, the output-input mapping is many to one perhaps, disturbances and other uncertainties may lead to the input-output mapping to become one-to-many leading to degradation in the control performance. Though, the AIM relies on the accuracy of the model used for the controller design so, the research will not worry about the degradation. The block diagram of the speed based AIM is shown in fig 5.

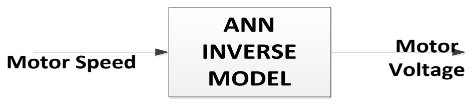

Fig 5 Block Diagram of The AIM

5.1 Structure of The AIM

The AIM for this research work is made up of three inputs and a single output structure for the three successive speed instances. Based on equation 23, the three inputs are $\omega_m(k+1)$; Speed at first instance $\omega_m(k)$; speed at second instance, $\omega_m(k-1)$;speed at third instance and the output is the $V_a(k)$ which is the motor terminal output voltage $V_t(k)$ from fig 6.

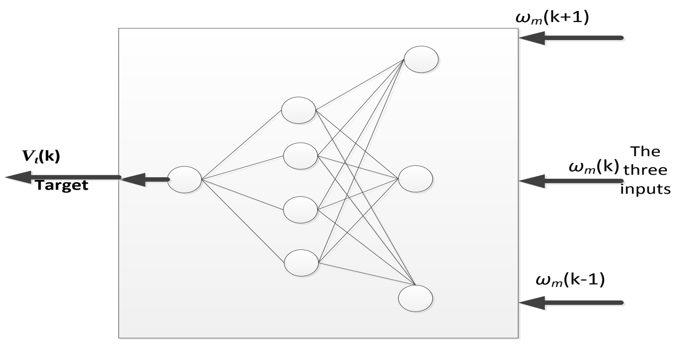

Fig 6 the structure of AIM

So based on the same equation 23, the nonlinear function (f.) can be presented in the following form:

$$f(\omega_m(k+1), \omega_m(k), \omega_m(k-1) = \frac{(\omega_m(k+1) - \alpha\omega_m(k) - \beta\omega_m(k-1) - \gamma\, sgn(\omega_m(k))\omega_m{}^2(k) - \delta sgn(\omega_m(k-1))\omega_m{}^2(k-1)}{\zeta} \quad (32)$$

The values of $(\omega_m(k+1), \omega_m(k), and\ \omega_m(k-1)$ apparently form the independent inputs of the ANN and the corresponding output as well is generated from equation 28.

5.2 Evaluating The Performance of The AIM

The generated $(\omega_m(k+1), \omega_m(k), \omega_m(k-1)$ inputs and the corresponding targets *$V_a(k)$* are used for offline training of the AIM to represent any DC servomotor with unknown parameters. From figure 6, it could be seen that the performance error is represented by *$e_i(k)$*. In evaluating the AIM performance, the value of $[\ e_i(k)]^2$ for all *kT* that are elements of time from 0 to t_f is minimized, that is;

$$[e_i(k)]^2 \forall kT \in [0, t_f] \quad (33)$$

Where T is the sampling period and t_f is the time for which simulation is performed. The terminal voltage (estimated) is given by:

$$\hat{V}_t = N[\omega_m(k), \omega_m(k-1), \omega_m(k-2)] \quad (34)$$

Once the DC motor is excited by an input signal, the output from the DC motor which is speed in this context is fed into the AIM as an input. The terminal voltage i.e. $\hat{V}_t$(k-1) is compared with the actual motor output *$e_i(k)$* for a common excitation signal. Then the mean square value of the error *$e_i(k)$* between the actual motor input and the estimated output voltage yields the performance error of the AIM.

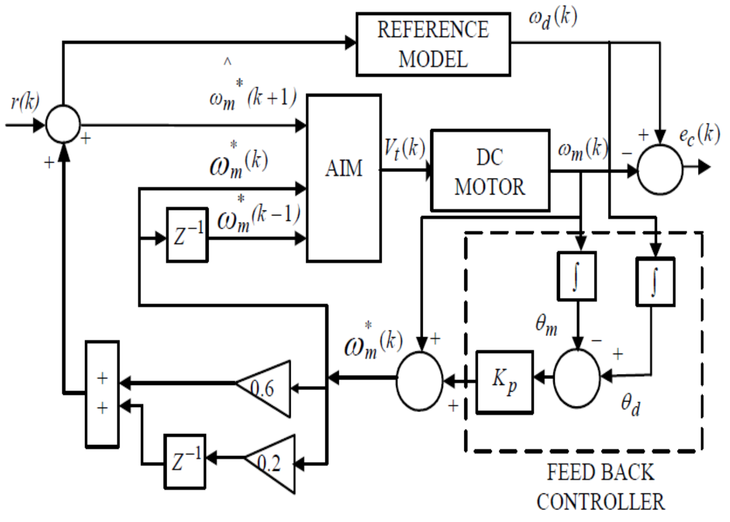

Fig 7 Structure of Speed and Position based MRAC Control System

6.0 THE PID CONTROLLER MODEL

The PID controller was achieved using equation 2 to get the best possible gains. Table 1 shows the DC motor parameters.

Table 1: DC Motor Parameter Values (Weerasooriya and El-Sharkawi, 1991)

Parameters	Values
Moment of Inertia J	0.064kg-m^2
Damping Coefficient b	0.03475Nm/s
Torque Constant Kt	3.475Nm A^{-1}
Electromotive force Constant Ke	3.40NmA^{-1}
Electrical Resistance R$_a$	7.56Ω
Electrical Inductance L$_a$	0.055H

The neural network controller is a speed and position based MRAC system as shown in fig 7. Figure 8 shows the construction of the PID controller in Simulink.

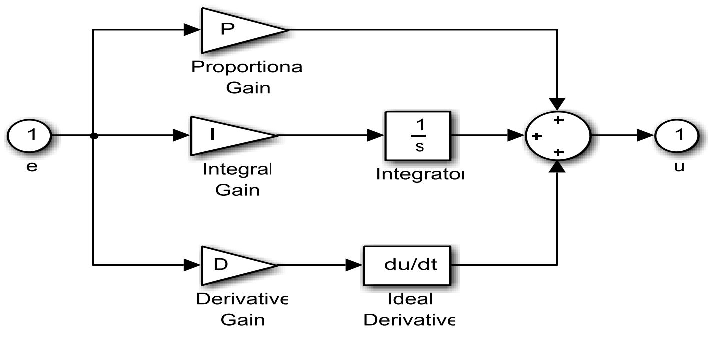

Fig 8 The PID Simulink Model

The PID controller is incorporated in the DC motor Simulink model to determine the effect of the PID controller on the system as shown in figure 9.

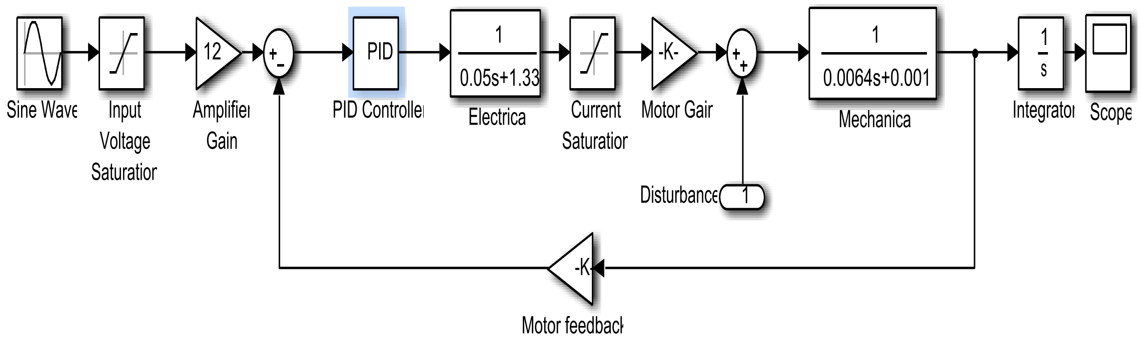

Fig 9: DC Motor Simulink Model with PID Controller

6.2 Measurement and Results

The Speed and Position based MRAC system and the PID controller system were used to drive a physical DC motor. The reference input to the system is a step function input. The essence of this is to determine the stability of the systems and their ability to reach one stationary state when starting from another. Step input value of 180 (degrees) were chosen arbitrary and used as input to both MRAC control system and the PID controller. The output time responses were graphed for both controllers to determine the existence of overshoot in voltage as shown in figures 9 and 10.

It could be observed from figure 9, that the Speed and Position based MRAC system was able to achieved steady state at about 0.56 sec without overshoot while the PID controller from figure 10, achieved steady state at about 2.41sec with voltage overshoot.

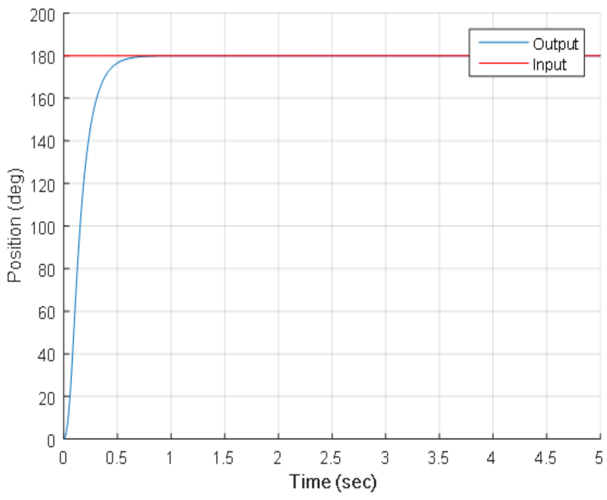

Fig 10 Step Input Response of Speed And Position Based MRAC System (180 Degrees)

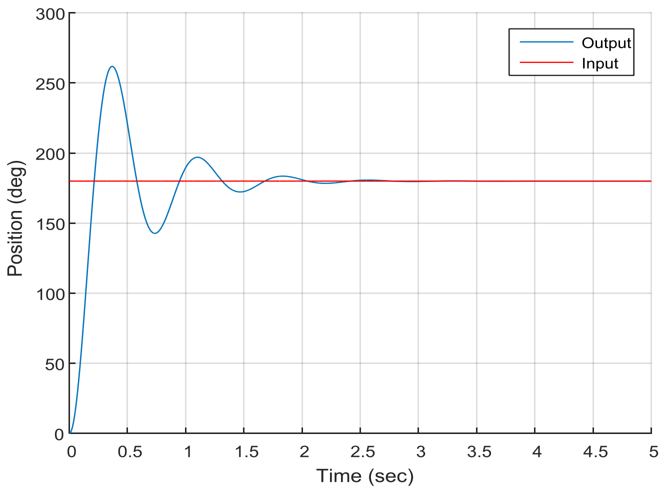

Fig 11 Step Input Response of PID Controller (180 Degrees)

CONCLUSION

Results from measurements show that the conventional PID controller was not able to accommodate the nonlinearities associated with the DC servomotor which lead to the distortion in the input voltage to cause overshoot between rise time and settling time. Perhaps, the speed and position based controller which is adaptive in nature, was able to accommodate those nonlinearities and yet maintain good control of the DC motor without voltage overshoot.

REFERENCES

1. Astrom K.J. and Wittenmark B., (1998). Computer Controlled Systems, Prentice-Hall.
2. Beale M.H, Hagan M.T. and Demuth H.B. (2015) *Neural Network Toolbox User'*
3. Haykins S. (1999) "Neural Networks– A comprehensive foundation", Second Edition, Prentice Hall, 1999
4. Kuo B. C and Golnaraghi F. (2002). *Automatic Control Systems, Eight Edition*. Pulisher: John Wiley & Sons, Inc. ISBN-13 978-0470-04896-2,
5. Makableh Y. (2011). *Efficient Control of DC Servomotor Systems Using Backpropagation Neural Networks.Electronic Theses & Dissertations*. 771. http://digitalcommons.georgiasouthern.edu/etd/771
6. Narendra K. S. and Parthasarathy K. (1990). "Identification and control of dynamical systems using neural networks", *IEEE Transactions on Neural Networks*, 1, (1), pp. 4-27.

7. Saerens M. and Soquet A. (1991). “Neural Controller Based on Back Propagation Algorithm”, IEEE Proceedings on Radar and Signal Processing, 138, (1), pp. 55 – 62.
8. Sheel S., Chandkishor R., and Gupta O. (2010). *Speed Control of DC Drives Using MRAC Technique.*International Conference on Mechanical and Electrical Technology:*EEE.*
9. Weerasooriya S. and El-Sharkawi M.A. (1991) “Identification and control of a DC motor using Back-propagation Neural Networks”, IEEE Transactions on Energy Conversion, Vol.6, No.4, pp. 663-669.