Comparison of Speed Control of DC Motor in Real Time Using Conventional and Nonconventional Methods

Budidi Udaya Kumar and M. Ramesh Patnaik

Abstract This paper illustrates the design and implementation of the speed control of DC motor in real time using Matlab and Arduino Due board with conventional and nonconventional methods. In the conventional technique, the PID controller is implemented using two different modeling approaches, viz., first principles modeling and data-driven modeling. In nonconventional techniques, an inverse neural network controller is trained with three layers using backpropagation algorithm, concurrent relay-based PID controller consists of a parallel PID controller with dead band relay, and a neuro-fuzzy controller is designed and implemented using the adaptive neuro-fuzzy interface system. The responses of simulation and in real time of closed-loop speed control of DC motor are presented. A comparative study of time response characteristics analysis has been carried out and observed that the performance of the ANFIS controller performs better control than other control methods used in this paper.

Keywords PID controller · Inverse neural network · ANFIS · CRPID

1 Introduction

DC motors are used as an actuator and act as a final control element in many process control systems. In control systems, the controller manipulates the actuator based on the error signal. DC motors have been widely used in most of the applications in avionics, process plants, automobiles, rolling mills, consumer products, etc. In such

B. Udaya Kumar (✉)
Department of Instrument Technology, Andhra University, Visakhapatnam 530003, Andhra Pradesh, India
e-mail: udayakumar.budidi@gitam.edu

M. Ramesh Patnaik
Department of Electronics and Instrumentation Engineering, GITAM University, Visakhapatnam 530045, Andhra Pradesh, India
e-mail: ramesh_patnaik@yahoo.com

G. Panda et al. (eds.), *Microelectronics, Electromagnetics and Telecommunications*, Lecture Notes in Electrical Engineering 521,
https://doi.org/10.1007/978-981-13-1906-8_67

applications, speed and the position of the motor need to be controlled accurately, to improve the efficiency and optimum performance of the system. Various conventional and nonconventional techniques are currently in use to control the speed of a DC motor in literature as PID controller, fuzzy logic, neural networks, hybrid models, etc.

Traditional methods like P, PI, and PID controllers are commonly used in many applications as it is simple in design. The accuracy of the system response highly depends on the tuning parameters of the PID controller and the governing equations of the mathematical model [1]. Conventional control theory is suitable to model linear systems, and it is difficult to model the complex nonlinear systems where the mathematical model is indecisive. In the past decade, several nonconventional techniques have been developed and proven to overcome the difficulties in the modeling of complex nonlinear dynamical systems such as sliding mode control, model reference adaptive control, fuzzy logic, neural network, and hybrid models like fuzzy-PID, neuro-fuzzy, and genetic algorithms. In this paper, the design and implementation of speed control of DC motor with Matlab software and Arduino Due hardware are presented. Conventional PID method and nonconventional methods like inverse neural network, neuro-fuzzy, and concurrent relay-based PID were implemented. A comparative study is carried out in both simulations and in real time.

2 Experimental Setup

Figure 1 shows the block diagram of experimental setup considered in this paper. Transcoil 12 V DC motor with inbuilt tachometer which runs at the maximum speed of 4500 rpm is used. The sensitivity of the DC motor is 1.9 V/1000 rpm. At maximum speed, the output voltage from the tachometer gives 8.5 V; however, the Arduino Due microcontroller measures the maximum analog input voltage range of 0–3.3 V. A level shifter is required to reduce the voltage of 0–8.5 V to 0–3.3 V range is designed. The noise from the output of the tachometer is eliminated by an RC low pass filter. The voltage from the tachometer is measured with Arduino Due board with a sampling rate of 0.01 s and converted into speed [2]. The speed of the DC motor is controlled using L293D H-bridge controller. PWM signal required for the desired speed is calculated and sent to the H-bridge controller [3]. The overall circuit diagram and experimental setup are shown in Figs. 2 and 3.

2.1 Conventional Control

The block diagram of a general closed-loop feedback control system is shown in Fig. 4. The controller takes the error as an input signal and produces the output signal based on the type of the controller. The error signal is the difference between the setpoint and the measured value. The type of the controller can be conventional PID or nonconventional controllers such as fuzzy, neural or neuro-fuzzy controllers,

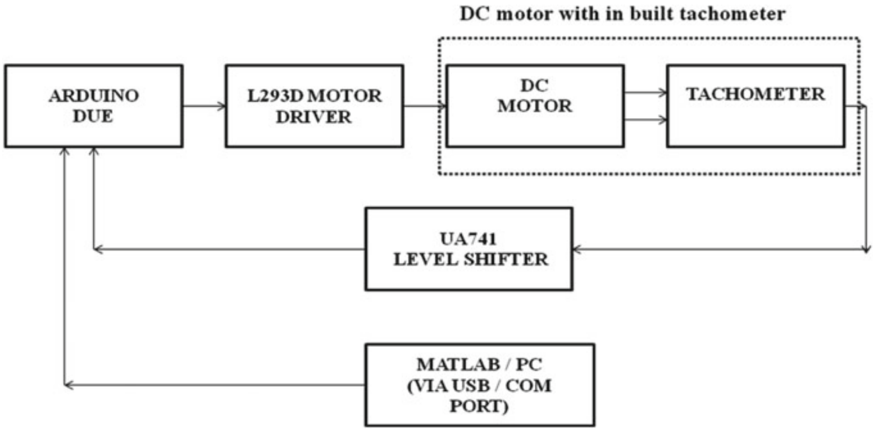


Fig. 1 Block diagram of speed control of DC motor

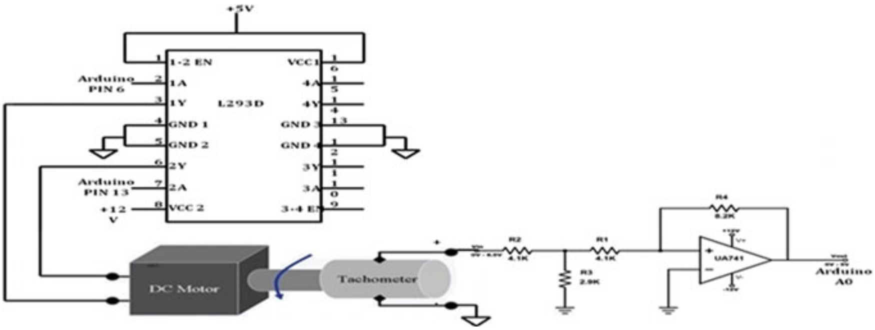


Fig. 2 Circuit diagram of speed control of DC motor

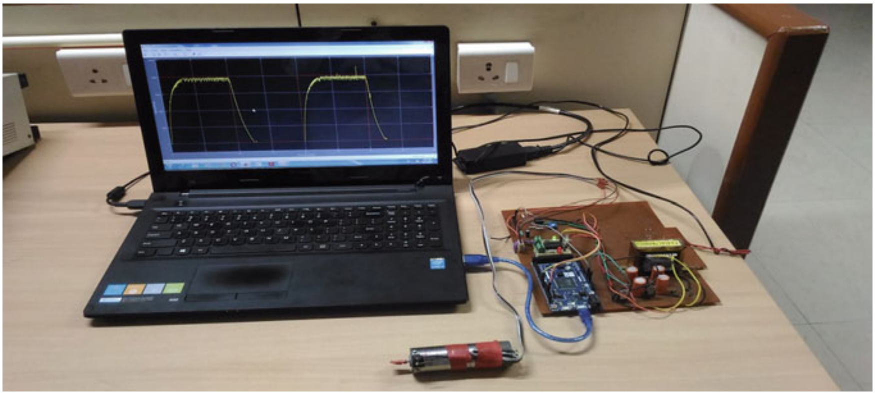


Fig. 3 Experimental setup

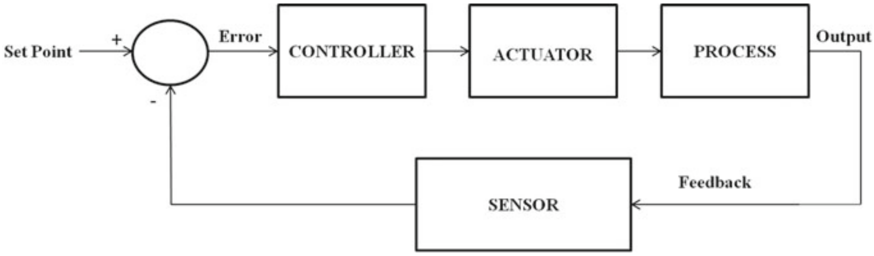
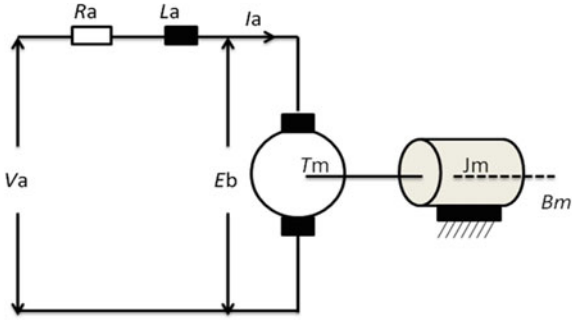


Fig. 4 Closed-loop feedback control system

Fig. 5 DC motor model



etc. In this paper, PID controller is designed using the first principles model and data-driven model.

2.1.1 First Principles Modeling

First principles model requires the knowledge of fundamental governing laws. The mathematical model of the DC motor is as shown in Fig. 5. The transfer function of the speed versus input voltage is shown Eq. (1). The model of DC motor can be designed through either Matlab Simulink toolbox or Simscape toolbox. Most of the times all the parameters may not be available from the manufacturer to design the model. Some of the parameters not specified by the manufacturer for the above motor are Bm (Damping Coefficient), Kb (Back EMF constant).

$$\frac{\mathrm{w(s)}}{\mathrm{V_a(s)}} = \frac{\mathrm{K_m}}{\mathrm{L_a} \cdot \mathrm{J_m S^2} + (\mathrm{R_a} \cdot \mathrm{J_m} + \mathrm{L_a} \cdot \mathrm{B_m})\mathrm{S^2} + (\mathrm{R_a} \cdot \mathrm{B_m} + \mathrm{K_b} \cdot \mathrm{K_T})\mathrm{S}} \tag{1}$$

Matlab provides an extension to Simulink toolbox is Simscape toolbox to develop the model of DC motor from its electrical and mechanical ratings. The Simscape model of the DC motor is shown in Fig. 6.

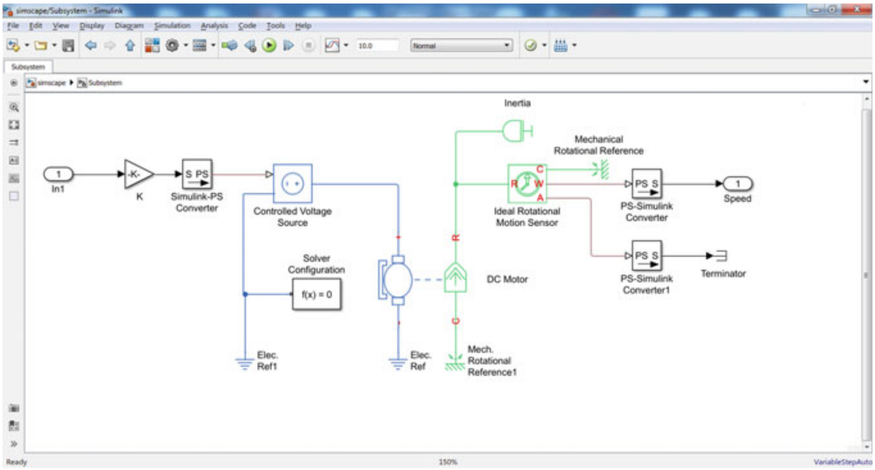
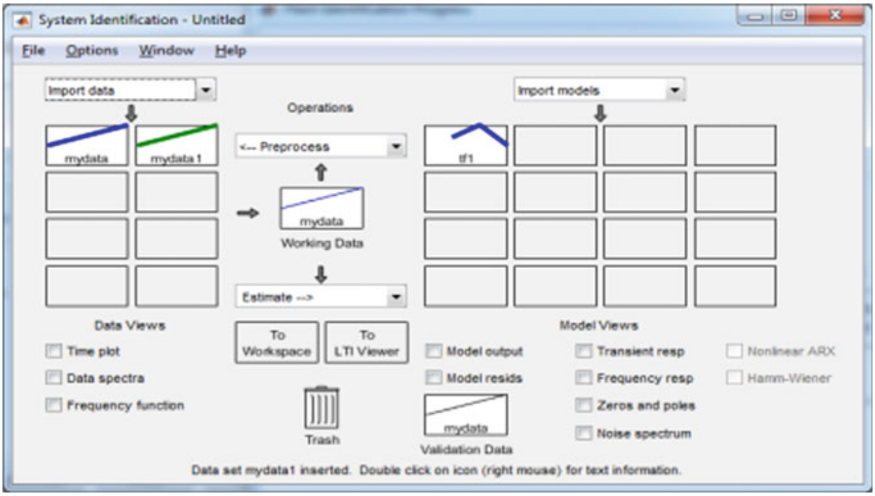


Fig. 6 Simscape model of a DC motor

2.1.2 Data-Driven Modeling

Data-driven modeling is used when the transfer function of the system cannot be derived from the mathematical modeling. System identification tools are used to estimate either a linear or nonlinear transfer functions from the acquired input–output data of the system. Figure 7 shows the linear transfer function estimation from the system identification tools.



Data set mydata1 inserted. Double click on icon (right mouse) for text information.

Fig. 7 System identification tools to obtain transfer function

The estimated linear transfer function of the DC motor is

$$\mathrm{TF} = \frac{0.0313\mathrm{Z}^{-3}}{1 - 1.974\mathrm{Z}^{-1} + 0.9767\mathrm{Z}^{-2}} \tag{2}$$

2.2 Design of PID Controller

A widely used feedback controller in process control is PID controller due to its simplicity in design [4]. The output of the controller depends on the present, past, and future error. The governing equation of the PID controller is

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} \tag{3}$$

where $\mathrm{K_p}$, $\mathrm{K_i}$, $\mathrm{K_d}$ are the constants of the controller proportional, integral, and derivative control. The main challenge of the PID controller is tuning of the PID parameters can be performed either online or offline. Several offline tuning techniques such as Ziegler Nicholas and open reaction curve method exist in the literature.

In this paper, tuning of the PID controller has done in offline with Matlab automatic tuning toolkit from the model obtained through either first principles or data-driven modeling technique. Figure 8 shows the Simulink model to tune PID constants, and Fig. 9 shows the parameters tuned by the toolkit. Figure 10 shows the Simulink model of closed-loop controller in real time which measure the speed from the Arduino Due board and generates the PWM signal by desired for the PID controller.

3 Nonconventional Modeling

Conventional techniques are still widely used in process industries due to simple in design, less cost, and not appropriate for nonlinear systems [5]. Obtaining the mathematical model for nonlinear complex systems is difficult consequently; the intelligent controlling techniques such as fuzzy logic, neural network, and hybrid models are used to achieve optimum performance in nonlinear systems.

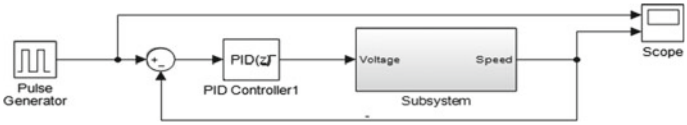
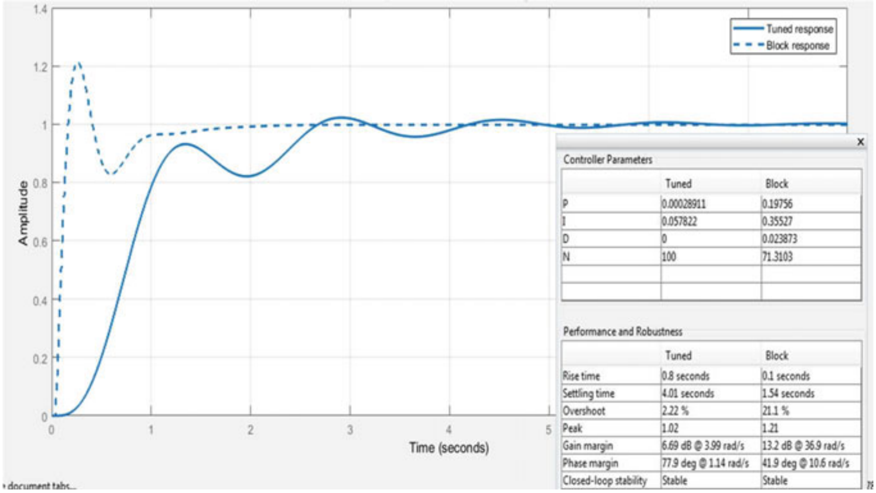


Fig. 8 Simulink model to tune PID constants



Controller Parameters

	Tuned	Block
P	0.00028911	0.19756
I	0.057822	0.35527
D	0	0.023873
N	100	71.3103

Performance and Robustness

	Tuned	Block
Rise time	0.8 seconds	0.1 seconds
Settling time	4.01 seconds	1.54 seconds
Overshoot	2.22 %	21.1 %
Peak	1.02	1.21
Gain margin	6.69 dB @ 3.99 rad/s	13.2 dB @ 36.9 rad/s
Phase margin	77.9 deg @ 1.14 rad/s	41.9 deg @ 10.6 rad/s
Closed-loop stability	Stable	Stable

Fig. 9 Tuning of PID parameters

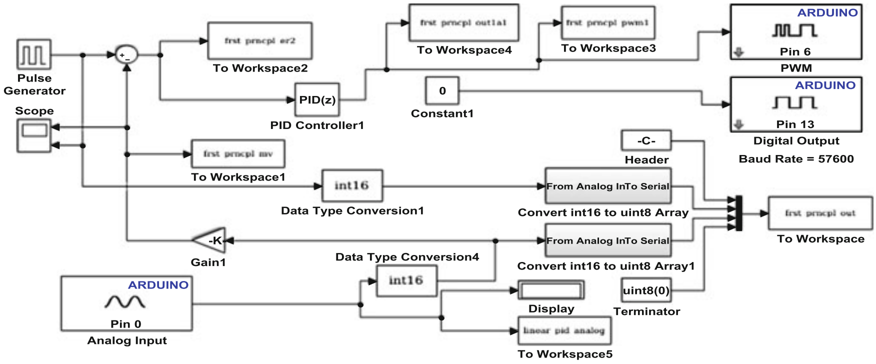


Fig. 10 Simulink model of PID controller in real time

3.1 *Inverse Neural Network Controller*

Figure 11 shows the block diagram of an inverse neural network controller. Neural networks are used in control of nonlinear systems due to its ability of learning. The network is trained to act as the inverse of the system and used as a controller [6]. In this paper, the network is trained with four inputs y(k + 1), y(k), y(k − 1), y(k − 2) and u(k) where u(k) is the input voltage of motor and y(k) is the speed of the motor as output. After the training, y(k + 1) is replaced with the set point. Equation 3 shows the output of the inverse model is

$$u(k) = f_2(y(k+1), y(k), \ldots, y(k-n-1), u(k-n), \ldots, u(k-m+1)) \quad (3)$$

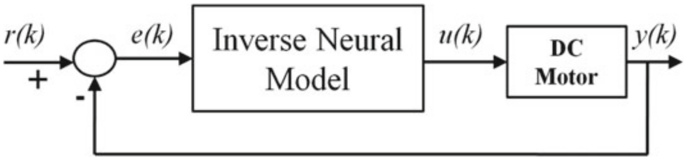


Fig. 11 Block diagram of an inverse neural network

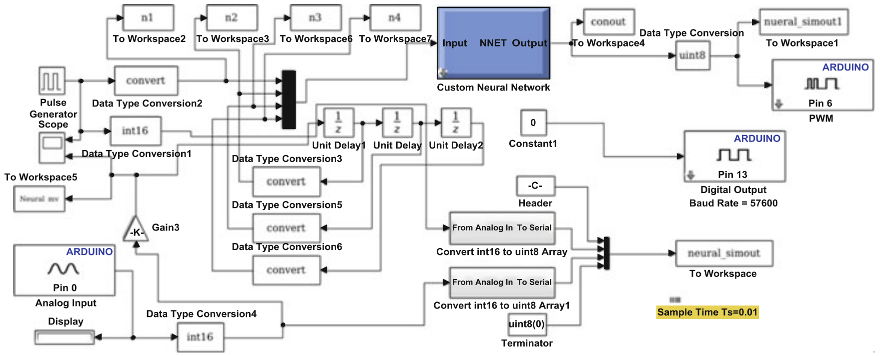


Fig. 12 Simulink model inverse neural network in real time

Figure 12 shows the implementation of neural network controller Simulink model in real time.

3.2 Design of Concurrent Relay PID Controller

Guomin Li proposed a hybrid model in 2007 by adding a dead band relay and a limited integrator to the PID controller. Dead band relay outputs a signal when the error is large and switched off when the error is small [7]. The purpose of the integrator is used to reducing the settling time. The lead compensator is to adjust the gain margin and phase margin to keep the closed loop system is stable. This hybrid model improves the system response when compared to PID alone. Figure 13 shows the block diagram of CRPID controller.

The lead compensator gain $\mathrm{Gl} = \frac{\tau s+1}{\alpha\tau s+1}$, dead band relay, and limited integrator $\mathrm{k_{ai}} = \frac{6\mathrm{k_i}}{\mathrm{d}}$ are calculated from above equations, where d is the maximum controller output and h is the maximum and T is the time constant and placed parallel to PID controller.

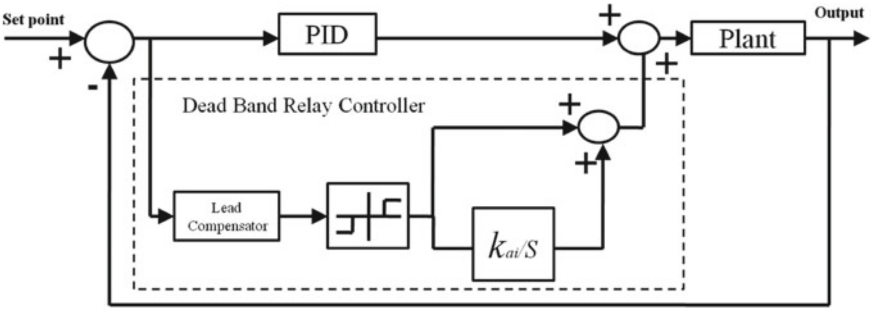
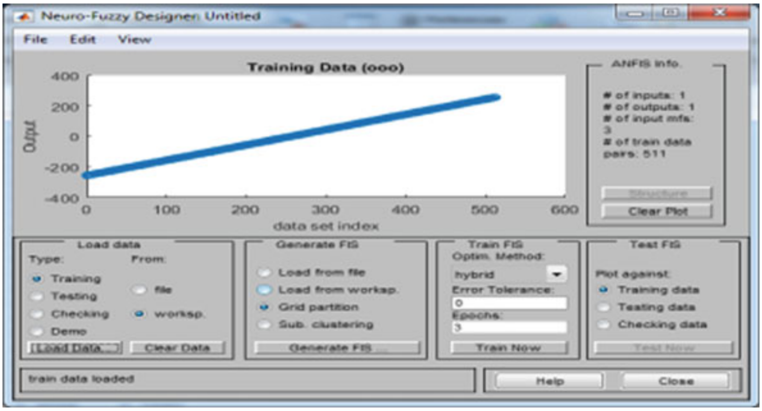


Fig. 13 Scheme of concurrent relay PID controller (CRPID)

3.3 *Design of Neuro-Fuzzy Controller*

Hybrid models such as fuzzy neural and neuro-fuzzy controllers offer the advantages of both the controllers. In 1993, Jang introduced the concept of an intelligent approach known as ANFIS which combines the merits of a fuzzy logic with optimized rules and membership functions and neural network models to solve nonlinear and complex models with good precision [8]. From the sampled input–output data, the Matlab ANFIS tool generates a fuzzy file and its member functions that are tuned with a neural network. Figure 14 shows the ANFIS tool to design a neuro-fuzzy controller. Figure 15 shows the Simulink model for implementation of ANFIS model in real time.



# of inputs: 1
# of outputs: 1
# of input mfs: 3
# of train data pairs: 511

Type: ● Training ○ Testing ○ Checking ○ Demo

From: ○ file ● worksp.

○ Load from file
○ Load from worksp.
● Grid partition
○ Sub. clustering

Optim. Method: hybrid

Error Tolerance: 0

Epochs: 3

Plot against: ● Training data ○ Testing data ○ Checking data

train data loaded

Fig. 14 Matlab ANFIS tool with input–output data

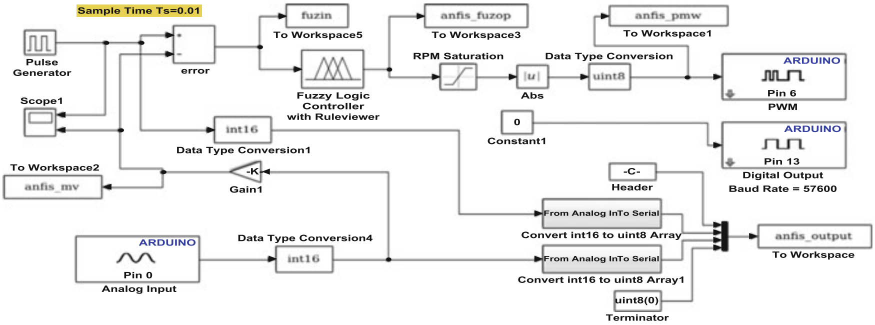


Fig. 15 Simulink model for implementation of ANFIS model in real time

4 Results and Conclusion

The experiment has been successfully conducted at different speeds and the practical values are compared with the theatrical values matched satisfactorily and observed that the speed of the motor follows the set point. Figure 16 shows the time response characteristics of all the models observed at a speed of 2000 rpm. The rise time and settling time of all the models are listed in Table 1. It is observed that the performance of ANFIS controller is better among the remaining methods used in this paper.

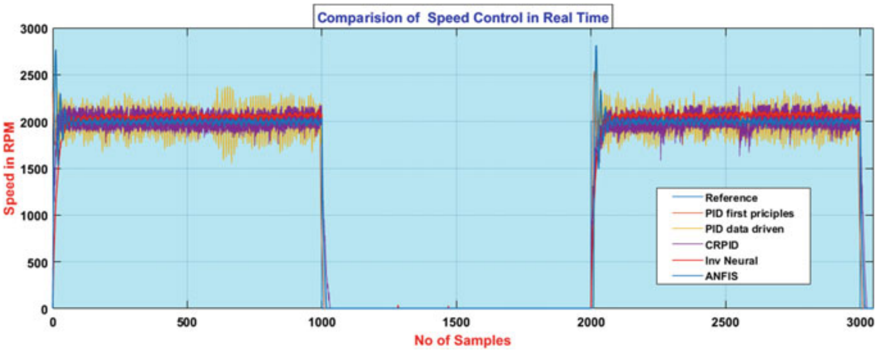


Fig. 16 Comparison of models in real time

Table 1 Comparison of real-time response parameters

Modeling technique	First principles	Data driven	CRPID	Neural network	ANFIS
Rise time (s)	3.2	2.75	2.1	1.8	0.9
Settling time (s)	3.3	2.7	1.6	1.3	0.6

References

1. Udaya Kumar B, Ramesh Patnaik M (2017) Comparative analysis of real time discrete PID controller design using first principles and data driven model. IJAREIE 6:1111–1121
2. Zaki AM, El-Bardini M, Soliman FAS, Sharaf MM (2015) Embedded two level direct adaptive fuzzy controller for DC motor speed control. Ain Shams Eng J
3. Petru L, Mazen G (2015) PWM control of a DC motor used to drive a conveyor belt. Procedia Eng 100:299–304
4. Puangdownreong D, Nawikavatan A, Thammarat C (2016) Optimal design of I-PD controller for DC motor speed control system by Cuckoo search. Int Electr Eng Congr 86:83–86
5. Aziz Khater A, El-Bardini M, El-Rabaie NM (2015) Embedded adaptive fuzzy controller based on reinforcement learning for DC motor with flexible shaft. Arab J Sci Eng 40:2389–2406
6. Buzi E, Marango P (2013) A comparison of conventional and nonconventional methods of DC motor speed control. In: 15th IFAC workshop on international stability, technology and culture, vol 46, pp 50–53
7. Li G, Tsang KM (2007) Concurrent relay-PID control for motor position servo systems. IJCAS 5:234–242
8. Jayetileke HR, de Mel WR, Ratnayake HUW (2014) Real-time fuzzy logic speed tracking controller for a DC motor using Arduino Due. In: 7th international conference on information and automation for sustainability. IEEE