Chapter

Wavelet Neural Networks for Speed Control of BLDC Motor

Ameer L. Saleh, Adel A. Obed, Hamza H. Qasim, Waleed I.H. Breesam, Yasir I.A. Al-Yasir, Naser Ojaroudi Parchin and Raed A. Abd-Alhameed

Abstract

In the recent years, researchers have sophisticated the synthesis of neural networks depending on the wavelet functions to build the wavelet neural networks (WNNs), where the wavelet function is utilized in the hidden layer as a sigmoid function instead of conventional sigmoid function that is utilized in artificial neural network. The WNN inherits the features of the wavelet function and the neural network (NN), such as self-learning, self-adapting, time-frequency location, robustness, and nonlinearity. Besides, the wavelet function theory guarantees that the WNN can simulate the nonlinear system precisely and rapidly. In this chapter, the WNN is used with PID controller to make a developed controller named WNN-PID controller. This controller will be utilized to control the speed of Brushless DC (BLDC) motor to get preferable performance than the traditional controller techniques. Besides, the particle swarm optimization (PSO) algorithm is utilized to optimize the parameters of the WNN-PID controller. The modification for this method of the WNN such as the recurrent wavelet neural network (RWNN) was included in this chapter. Simulation results for all the above methods are given and compared.

Keywords: BLDC motor, particle swarm optimization (PSO), wavelet neural network (WNN), speed control

1. Introduction

Brushless DC (BLDC) motors have a wide application in our life due to their high-power density and high dynamic response. In addition, the BLDC motor is utilized with constant loads, varying loads, and position applications with high accuracy. This motor is generally controlled utilizing electronically commutation by three-phase power semiconductor bridge inverter with rotor position sensors that are required for starting and providing proper firing sequence to turn on the power devices in the inverter bridge. Based on the rotor position, the power devices are commutated sequentially every 60° [1, 2]. The mathematical model and the Simulink model of BLDC motor to control the speed of a BLDC by using conventional methods are introduced in Refs. [2–6]. The DC-DC converter technique is utilized to control the speed of the motor [5, 6].

In the past decade, artificial intelligence techniques such as neural networks, fuzzy-neural networks, and wavelet neural networks control have been utilized to control the speed of the BLDC motor [7–10]. Since BLDC motor is a multivariable and nonlinear system, it is complex to obtain high performance by applying classical PID control. The main objective of this chapter is to develop wavelet neural networks (WNNs) to control the speed of the BLDC motor, and the recurrent wavelet neural network (RWNN). These methods lead to an enhanced dynamic performance of the system of motor drive and are resistant to load perturbations. The learning strategy for the wavelet neural network and PID controller is developed based on PSO algorithm.

2. Wavelet networks

The combination of wavelet principle and neural networks has led to producing new representing network of wavelet neural network (WNN). Wavelet networks are feedforward networks utilizing wavelets as activation functions. Wavelet networks substitute the sigmoid activation components of the classical feedforward artificial neural networks (ANNs) with wavelets transform function. In wavelet neural networks, both the translation (position) and the dilation are tuning besides weights. The utilization of wavelet node outcomes in efficient networks are optimally approximated and estimated for nonlinear and nonstationary functions [11, 12]. There are two main types to construct the wavelet neural network:

- **Wavenet (fixed grid WNs):** in this type, the neural network and the wavelet processing are accomplished separately. The input signal is first decomposed utilizing some wavelet bases by neurons in the hidden layer with fixed wavelet bases; positions and dilations of the wavelets are preset and only the weights have to be adjusted by learning the network. The main problem is the choice of wavelet frames/bases [12, 13].

- **Wavelet network (adaptive WNs):** this type merges the two theories, which are the dilation and the translation of wavelets along with the summer weights that are adjusted in conformity with some learning procedures. Generally, the modeling of the wavelet network involves two steps: determining the network construction (the number of neurons in each layer, the number of layers, and the type of activations function (wavelet transform)) and modifying the wavelet network parameters by some optimizing algorithm method [12, 13].

3. Structure of wavelet neural network (WNN)

The arrangement of WNN is similar to that of the neural network. The hidden layer contains neurons, whose activation functions are driven from a wavelet basis. These wavelet neurons are generally referred to as wavelons, whose parameters of the inputs contain the wavelet dilation and translation elements [12, 14]. Wavelet networks can be categorized into recurrent and nonrecurrent (feedforward) types.

3.1 Feedforward wavelet neural network (FFWNN)

The FFWNN has no feedback connection. That is, the output is calculated straightly from the input with feedforward connections [12]. There are two arrangements of feedforward wavelet networks:

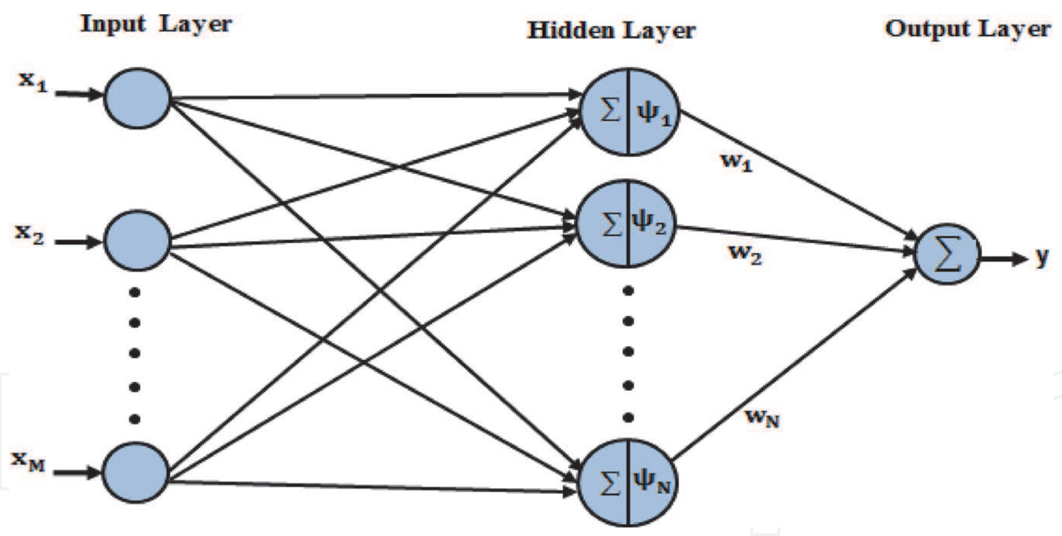

Figure 1.
The building of radial basis wavelet neural network (RBWNN).

3.1.1 Radial basis wavelet neural network (RBWNN)

Radial basis wavelet neural network (RBWNN) is the simplest form of the wavelet network [15, 16]. The arrangement of radial basis wavelet neural network (RBWNN) is shown in **Figure 1**. This network approaches any required signal f(t) by simplifying a linear combination of a group of daughter wavelets $\psi_{a,b}$, where $\psi_{a,b}$ are created by dilation a and translation b from mother wavelet ψ [11, 17].

$$\psi_{a,b} = \psi\left(\frac{x-b}{a}\right) \tag{1}$$

The network output is specified as follows [10, 18]:

$$y = \sum_{n=1}^{N} w_N \psi_{a_N,b_N} \tag{2}$$

where x is the input signal, N is the number of neuron in the hidden layer, and w_N is the weights of the output. The network parameters w_N, a_N, and b_N can be training and optimizing by any optimization technique. In this chapter, the PSO algorithm is used to minimize the error according to the fitness function as will be demonstrated later.

3.1.2 Conventional wavelet neural network

The conventional WNN is a general form of radial basis wavelet neural network [19]. **Figure 2** depicts the building of the conventional wavelet network, the number of hidden layers and neurons that are selected to create an appropriate WNN, and the parameters that are optimized by the PSO algorithm. The input layer can be represented by a vector $x = [x_1, x_2, \ldots, x_M]$, the output layer represented by a vector $y = [y_1, y_2, \ldots, y_K]$, and the activation function of hidden layer is the wavelet basis function. The output Y_j can be given as follows [11, 19]:

$$Y_j = \sigma(u_j) = \sigma\left[\sum_{n=1}^{N} w_{j,n}\ \psi_{a_n b_n}\left(\sum_{m=1}^{M} v_{n,m}\ x_m\right) + g\right] \tag{3}$$

where, $j = 1, 2, 3, \ldots, K$; M is the number of inputs; K is the number of output layers; N is the number of hidden layers; g is the bias; and $\sigma(u)$ is the activation function of the output layer, the most common formula of activation function being sigmoid function which can be illustrated as follows [12]:

$$\sigma(u) = \frac{1}{1 + e^{-u}} \tag{4}$$

3.2 Recurrent wavelet neural networks (RWNNs)

In recurrent WNNs, the output depends not only on the present inputs of the network but also on the prior outputs or conditions of the network [12, 15]. Recurrent networks have feedback and are also known as feedback networks. There are several types of recurrent networks that depend on the feedback connection [12–22].

In the recurrent wavelet network structures, the wavelet network input involves delayed samples of the system output y(k). The number of inputs increases with the order of the system actuality demonstrated. **Figure 3** depicts the structure of recurrent wavelet network. Hence, the output for each layer can be calculated as [20, 23, 24]:

$$\psi_N = \psi\left(\frac{u_N - b_N}{a_N}\right) \tag{5}$$

where a_N and b_N are translation and dilation parameters of wavelets. The inputs of this layer for time n can be denoted as:

$$u_N(n) = x_N(n) + \psi_N(n-1) \times \varnothing_N \tag{6}$$

where $\varnothing_N$ represents the weight of the self-feedback loop. The output of the network is given as follow:

$$y = \sum_{N=1}^{N} w_N \psi\left(\frac{u_N - b_N}{a_N}\right) \tag{7}$$

$$u(n) = x(n - D_i) + y(n - D_0) \times r_N \tag{8}$$

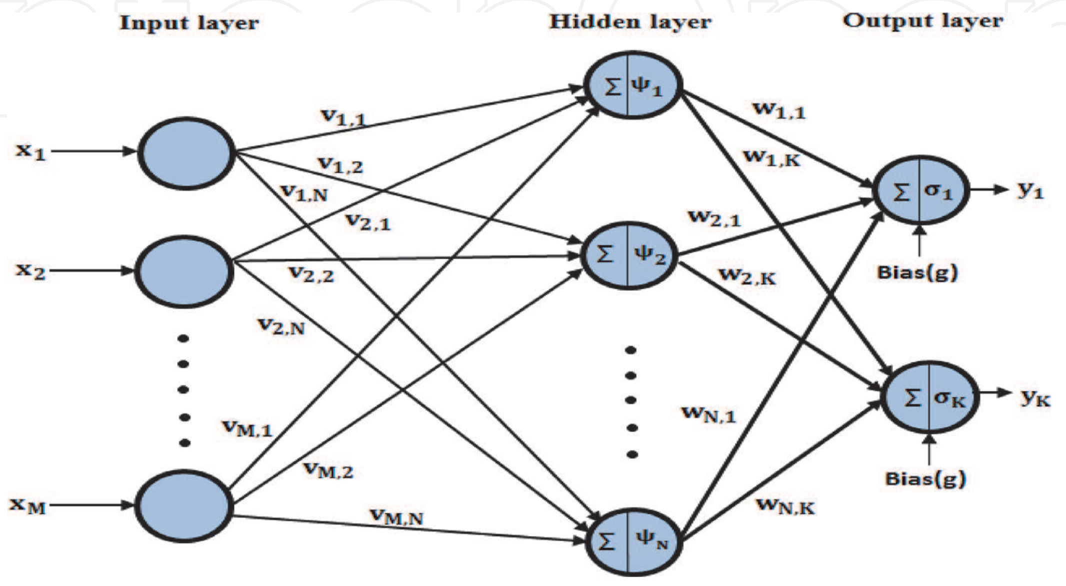

Figure 2.
The building of conventional wavelet neural network.

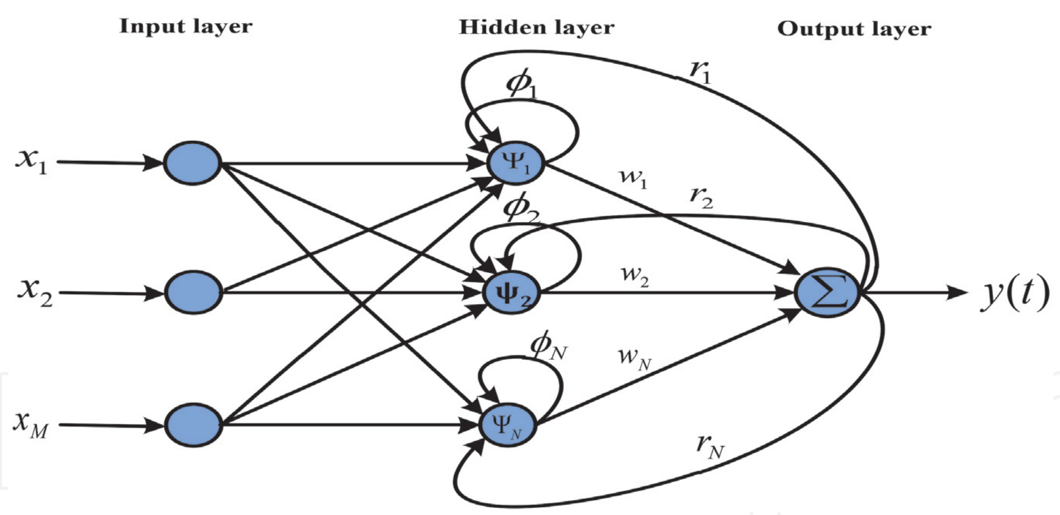

Figure 3.
The recurrent wavelet neural network.

where x is the input signal, N is the number of neuron in the hidden layer, w_N is the output weight, D_i, D_0 is the number of delay for the input and output network, and r_n is the weight of the output feedback loop.

4. Particle swarm optimization

Particle swarm optimization is an inhabitance-based computational procedure motivated from the simulation of gregarious behaviors (social-psychological): fish schooling, bird flocking, and swarm theory. PSO was firstly invented and established by Eberhart and Kennedy [25, 26]. In the PSO algorithm, in place of utilizing evolutionary operators such as mutation and crossover to operate algorithms, the population dynamics emulates a "bird flock" behavior, where social sharing of information takes place and individuals can yield from the finds and prior experience of all the other companions through the search for food. Therefore, each companion, called particle, in the population, which is called swarm intelligence as shown in **Figure 4**, is assumed to fly in several directions over the search space to meet the request fitness function [27, 28].

4.1 Particle swarm optimization algorithm

The PSO algorithm is one of the evolutionary computation techniques to solve optimization troubles. In this algorithm, a swarm of individuals or entities called particles flies over the exploration space [29, 30]. Each particle acts as a probable

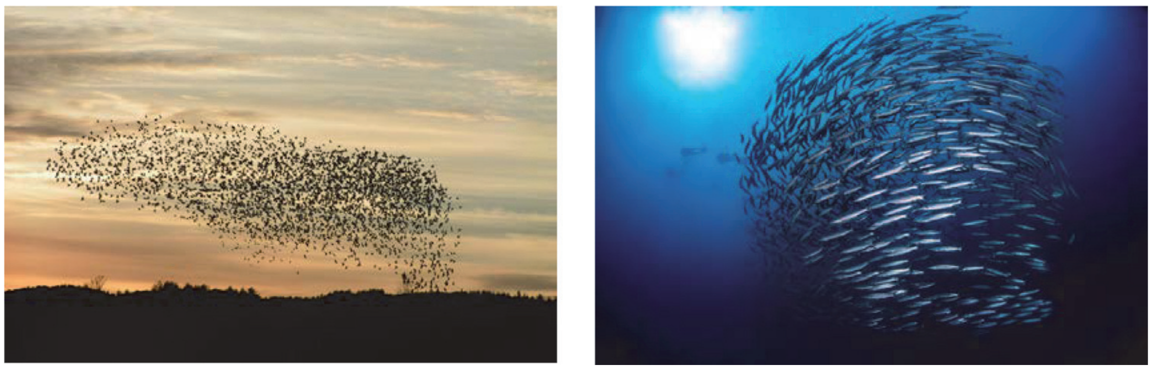

Figure 4.
Swarm intelligence.

solution to the optimization troubles. The position of a particle is influenced by the best position visited by itself, i.e., its own knowledge or experience and the position of the best particle in its knowledge of neighboring particles. When the neighborhood is the entire swarm, the best position in the neighborhood of the particle is denoted as the global best position and the resulting algorithm is referred to as the global best position PSO, where the finest prior position that gives the minimum fitness value of any particle is called local best position (lbest). The index of the best particle of all particles in the population is called global best position (gbest). The algorithm is generally referred to as the lbest PSO when smaller neighborhoods are used. For each particle, the performance is measured utilizing an objective function that differs depending on the optimization challenge. The basic PSO algorithm is given below according to the flow chart which is shown in **Figure 5** [31–35].

Step 1. Generation of population particles

Create particles regularly distributed over x, then choose the number of particles, number of iterations, modification accelerating coefficients c_1 and c_2, the inertia weight (w) and random numbers R_1, and R_2 to start the optimum searching.

Step 2. The initialization for each particle

Initialize the present position $x_i(t)$ and the velocitiy $v_i(t)$ for each particle. The particles are randomly produced among the minimum and maximum limits of parameter values. Each particle is treated as a point in a D-dimensional space. The ith particle is denoted as $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$. The velocity for the particle i is

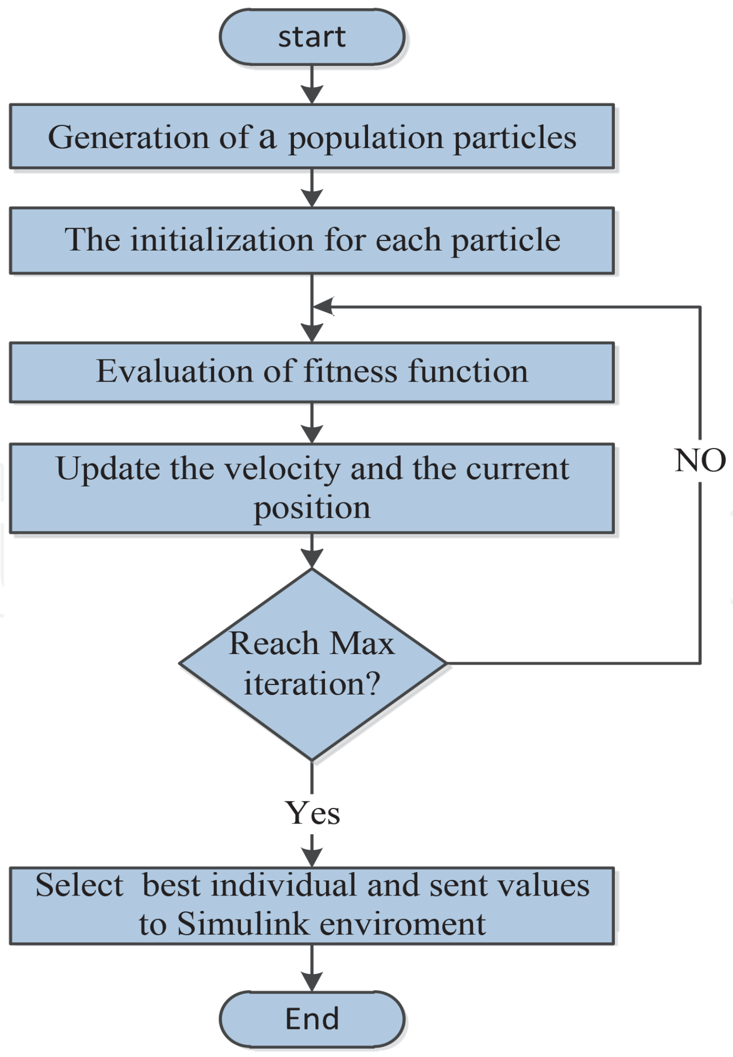
start
Generation of a population particles
The initialization for each particle
Evaluation of fitness function
Update the velocity and the current position
Reach Max iteration?
NO
Yes
Select best individual and sent values to Simulink enviroment
End

Figure 5.
General flow chart of PSO.

represented as $v_i = (v_{i1}, v_{i2}, \dots, v_{iD})$, then the local best position (lbest) and the global best position (gbest) are initialization.

Step 3. Evaluation of fitness function

The overall performance (speed of convergence, efficiency, and optimization accuracy) of the PSO algorithm counts on the objective function that observes the optimization search. The objective function is chosen to minimize the reference constraints. The popular performance standards based on the error condition are integrated absolute error (IAE), integrated of time weight square error (ITSE), and integrated of square error (ISE) that can be estimated theoretically in the frequency domain [31, 32, 36]. In this chapter, multiobjective functions are utilized based on the integral of the squared error (ISE) criterion and overshoot (M_p) criterion as follow [37, 38]:

$$\text{fitness function} = \min(\text{ISE}) + \min(M_p) \tag{9}$$

where

$$\text{ISE} = \int e^2(t)dt \tag{10}$$

$$M_p = \max(n) - (n_{ref}) \tag{11}$$

$$e(i) = D(i) - y(i) \tag{12}$$

where $y(i)$ is the system output and $D(i)$ is the desired output, while n is the actual speed and n_{ref} is the desired speed.

Step 4. Update the swarm

The updating of the velocity $v_i(t)$ and the present position $x_i(t)$ for each particle in the swarm is done according to Eqs. (13) and (14). Then the main loop and the objective function are calculated for updating positions of particles. If the new value is improved than the previous lbest, the new value is fixed to lbest. Similarly, gbest value is also updated as the best lbest. Velocity of each agent can be modification by the following:

$$v_i^{k+1} = w * v_i^k + c_1 * R_1 * (\text{lbest}_i - x_i^k) + c_2 * R_2 * (\text{gbest}_i - x_i^k) \tag{13}$$

And, the present position can be modification by the following:

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{14}$$

where x_i^k is the present position of particle i at iteration k, v_i^k is the velocity of particle i at iteration k, w is the inertia weight which can be represented in Eq. (15), c_1, c_2 represent positive acceleration constants and R_1, R_2 are random variables uniformly distributed in the range [0; 1].

$$w = w_{max} - \frac{(w_{max} - w_{min})}{iter_{max}} \tag{15}$$

where, w_{min} is the inital weight, w_{max} is the final weight, $iter_{max}$ is the maximum iteration number.

Step 5. Stopping criteria

If the current iteration number reaches the predetermined maximum iteration number, then exit. Otherwise, execute another initialization for each particle and reiterate the process.

5. Speed control of BLDC motor based on wavelet neural network

The WNN-PID controller based on PSO is proposed in this section, which combines the ability of the artificial neural networks for learning with the ability of wavelet for identification, control of dynamic system, and also having the capability of self-learning and adapting [10, 11, 19, 37]. Two types of wavelet network are modified in this section, feedforward WNN and proposed recurrent WNN with online tuning optimization using PSO algorithm [22–24].

5.1 WNN-PID controller based on PSO

In this type of controller, the WNN is utilized with PID controller based on PSO algorithm. WNN-PID controller utilizes online learning by PSO algorithm, where the PSO learning algorithm is used to train the translation parameters a_k and b_k, weights connection in the WNN, and the parameter (k_p, k_i, k_d) of PID controller on-line with the model of BLDC motor to control the speed at the desired value. There are two major issues to implement any wavelet neural networks. First, the network architecture is used and second, the algorithm is used to learn the network by the PSO algorithm. **Figure 6** depicts the block diagram of the BLDC motor with WNN-PID based on PSO algorithm. The structure and the design of the WNN-PID controller will be given in the next subsection.

5.2 Design of the structure of WNN-PID controller based on PSO training algorithm

1. **Design of PSO algorithm:** the PSO algorithm is discussed in Section 5, where each particle parameters are initiated to make a population and then the algorithm is accomplished according to the flow chart given in **Figure 6**, which includes training the parameters of this controller to guarantee the minimization of an objective function. The objective fitness is evaluated as follows:

$$\text{fitness function} = \min(\text{ISE}) + \min(M_p) \tag{16}$$

where ISE is the integrated of square error and M_p is the maximum peak overshoot.

2. **Design of WNN-PID controller**: to design the WNN-PID controller, the type of WNN must be selected as shown in Section 3 and also the number of layers and neurons and the wavelet function type must be selected [16]. In this chapter, the input layer has two inputs: the speed error and the change of this error. One hidden layer with four neurons is used. Three types of mother wavelet functions are used and they are: the Mexican hat function is [10, 11, 22, 37, 39]

$$\psi(x) = \left(1 - x^2\right)e^{\frac{-x^2}{2}} \tag{17}$$

The first partial derivative Mexican hat is

$$\psi(x) = (-x)e^{\frac{-x^2}{2}} \tag{18}$$

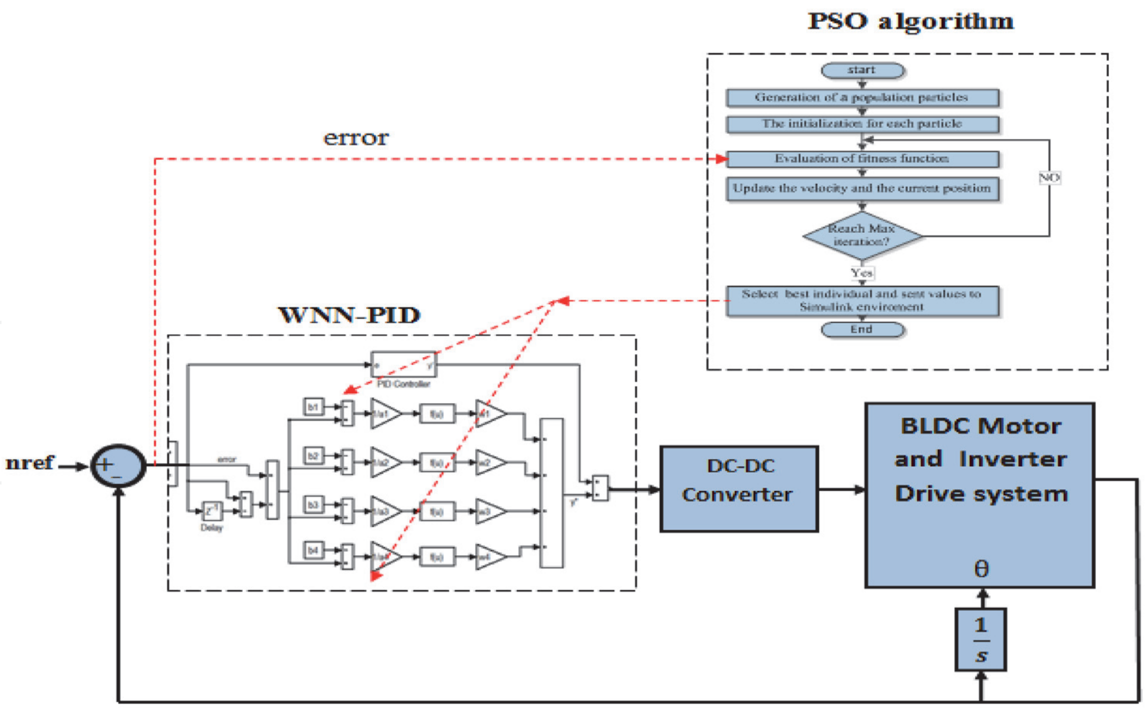
PSO algorithm
start
Generation of a population particles
The initialization for each particle
Evaluation of fitness function
Update the velocity and the current position
Reach Max iteration?
NO
Yes
Select best individual and sent values to Simulink environment
End
error
WNN-PID
PID Controller
nref
Delay
DC-DC Converter
BLDC Motor and Inverter Drive system
θ
1/s

Figure 6.
Block diagram of the BLDC motor with WNN-PID controller based on PSO algorithm.

The Morlet's basic wavelet function is

$$\psi(x) = \mathrm{Cos}(w\, x)e^{\frac{-x^2}{2}} \tag{19}$$

where x is the desired signal and w is a variable value, which was adopted to satisfy the admissibility condition. w = 5 is chosen. The output layer contains one output which is the sum of PID controller and WNN outputs. The parameters values of the WNN-PID controller, such as the dilation factors $a_{k's}$ and the translation factors $b_{k's}$ of the mother wavelet function, the weights connection $w_{k's}$ of the WNN, and PID parameters (k_p, k_i, k_d), are optimized online in PSO algorithm. The results given in this chapter are for Mexican hat function only. The results for the rest functions are similar to that in Mexican hat and are not given.

6. Simulink implementation and results for a BLDC motor drive based on WNN-PID controller

6.1 Speed control based on feedforward WNN-PID controller

The feedforward WNN with PID controller (FWNN-PID) is utilized to control the speed of the BLDC motor as shown in the Simulink model in **Figure 7**. The inputs of the WNN are the speed error and the change of this error, while the hidden layer has four neurons and one output in the output layer. The translation and dilation factors, weights connection for WNN, and PID parameters are learning on-line in PSO algorithm. The output of WNN is given by Eq. (2).

The PSO parameters are given in **Table 1**. These parameters are chosen to get optimal parameters for the PID controller and the wavelet neural network; when it is tuned on-line in PSO algorithm and BLDC motor drive, the optimal values for the PID controller parameters and the WNN parameters (a's, b's, w's) are given in **Tables 2** and **3**, respectively.

The BLDC motor drive is implemented in Simulink/Matlab program as shown in **Figure 6** with the optimal values of PID controller parameters and the optimal

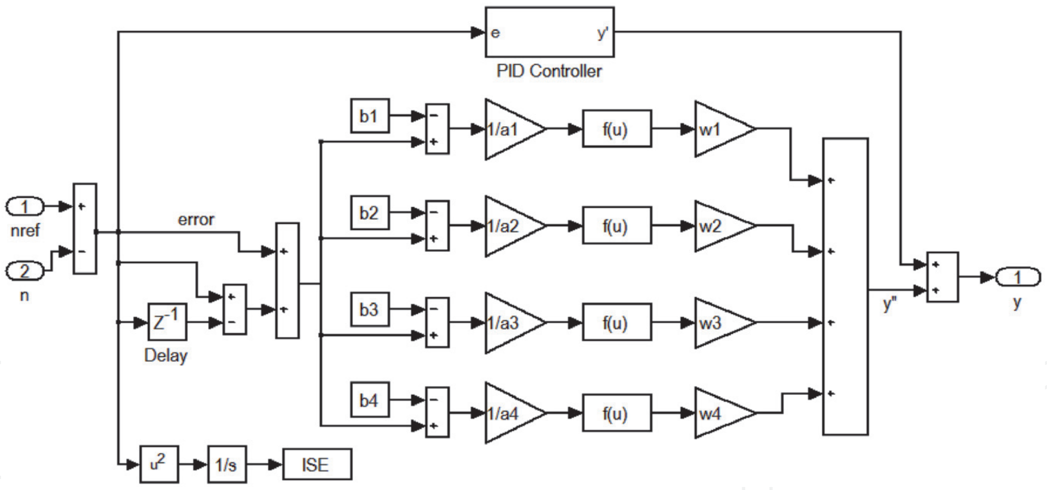

Figure 7.
Simulink model of WNN-PID controller.

PSO_Parameters	**Value**
Size of the swarm "no of birds"	50
Maximum iteration number	50
Dimension	15
PSO parameter c_1	1.2
PSO parameter c_2	1.2
W_{max}	0.9
W_{min}	0.3

Table 1.
PSO parameter values.

Parameters	K_p	K_i	K_d
Values	6.958	3.241	0.006274

Table 2.
PID parameters tuned using PSO for WNN-PID controller.

WNN dilation parameters				**WNN translation parameters**				**WNN weights parameters**			
a_1	a_2	a_3	a_4	b_1	b_2	b_3	b_4	w_1	w_2	w_3	w_4
4.535	1.227	3.013	3.141	1.248	0.314	2.576	1.352	3.632	3.321	2.2843	2.4722

Table 3.
WNN parameters tuned using PSO.

values of WNN parameters. **Figure 8** shows the speed response of the BLDC motor due to change in reference speed. The motor is started at a speed of 500 rpm and then is changed in step to a speed of 500 rpm for every 0.2 s. The actual speed of the motor is tracking the desired speed with a good response. The system starts at no load and suddenly a torque 2 N m (full load) is added at t = 0.4 s. **Figure 9** shows the speed response of the BLDC motor at 2000 rpm during no load and load condition. The developed torque during no load and load condition is shown in **Figure 10**. The position signal, the torque-speed characteristics, the phase current i_a, Phase Back-emf e_a voltage, and line voltage v_{ab} are given in **Figures 11–15**, respectively.

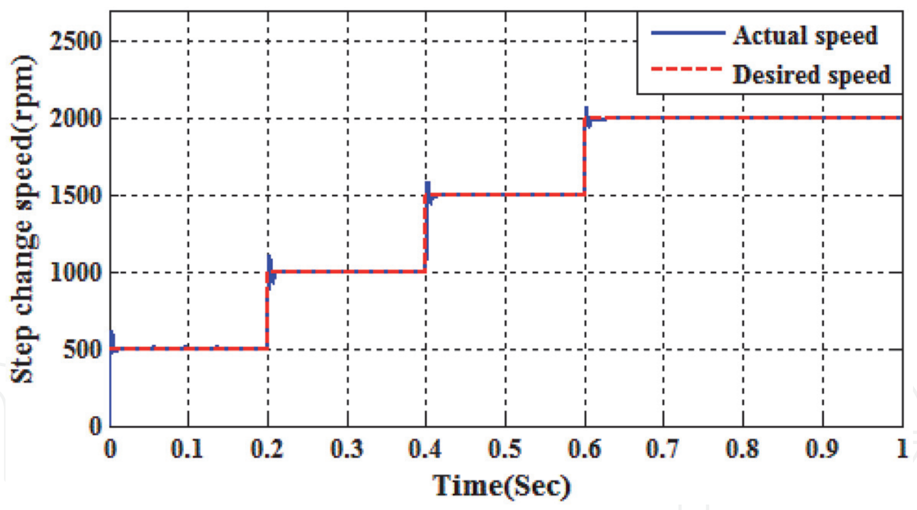

Figure 8.
Step change in speed response with feedforward WNN-PID controller.

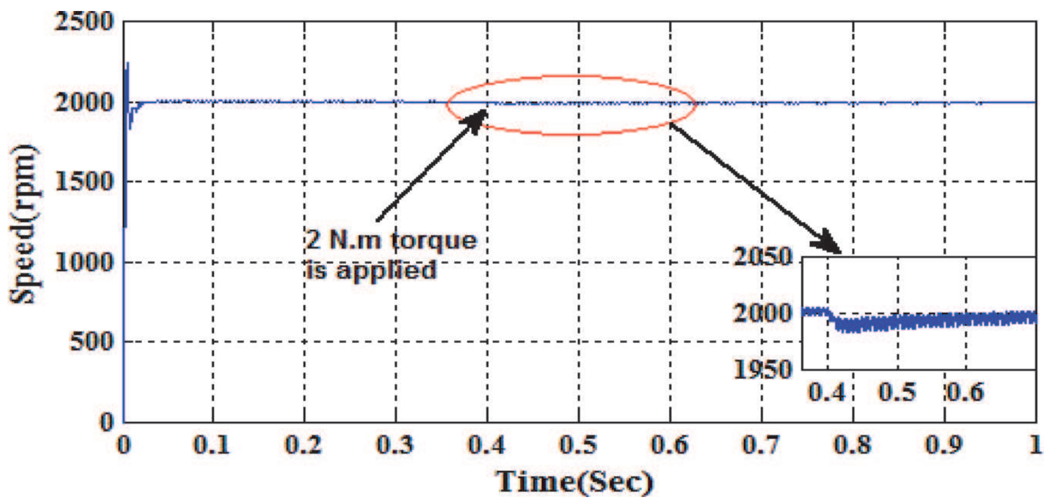

Figure 9.
Speed response of the BLDC motor with WNN-PID controller.

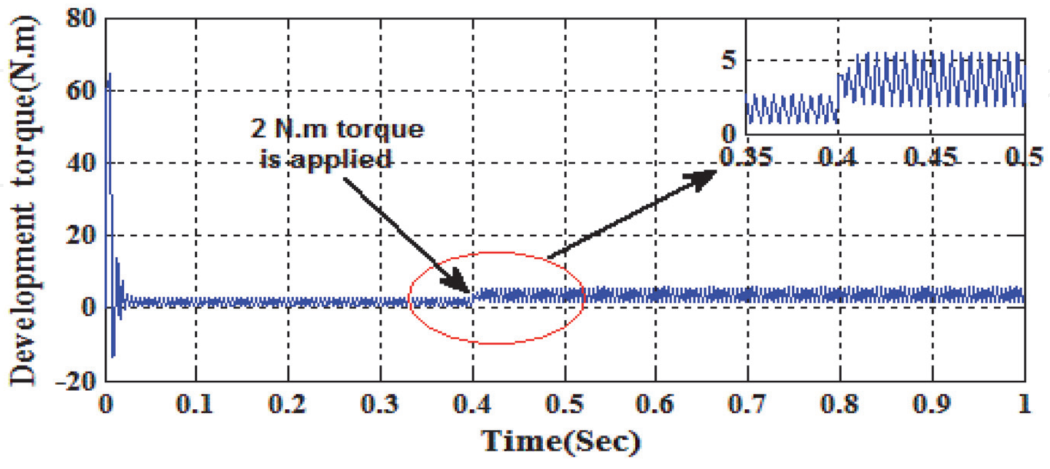

Figure 10.
Developed torque of BLDC motor with feedforward WNN-PID controller.

6.2 Speed control based on proposed recurrent WNN-PID controller

The RWNN that is proposed here is similar to that of feedforward WNN with feedback connections. The RWNN consists of three layers, with two inputs in the

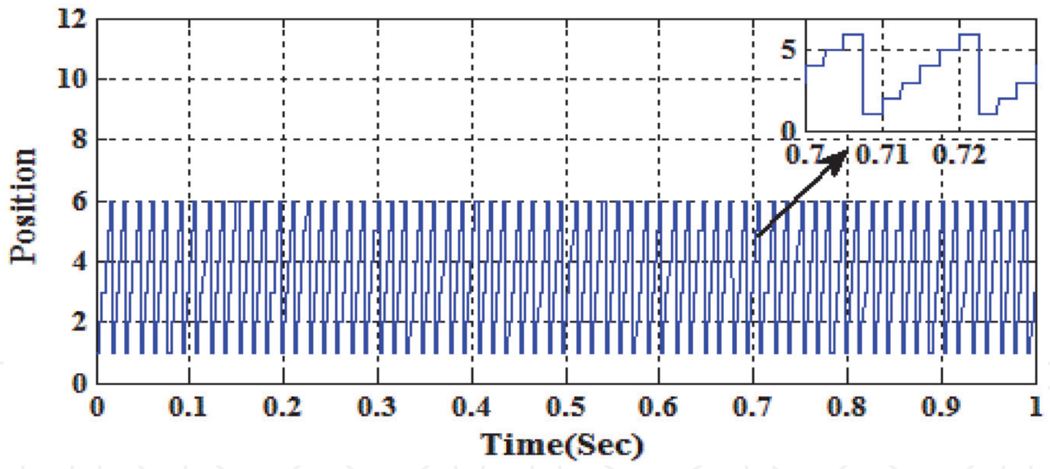

Figure 11.
Position stair signal of BLDC motor with WNN-PID controller.

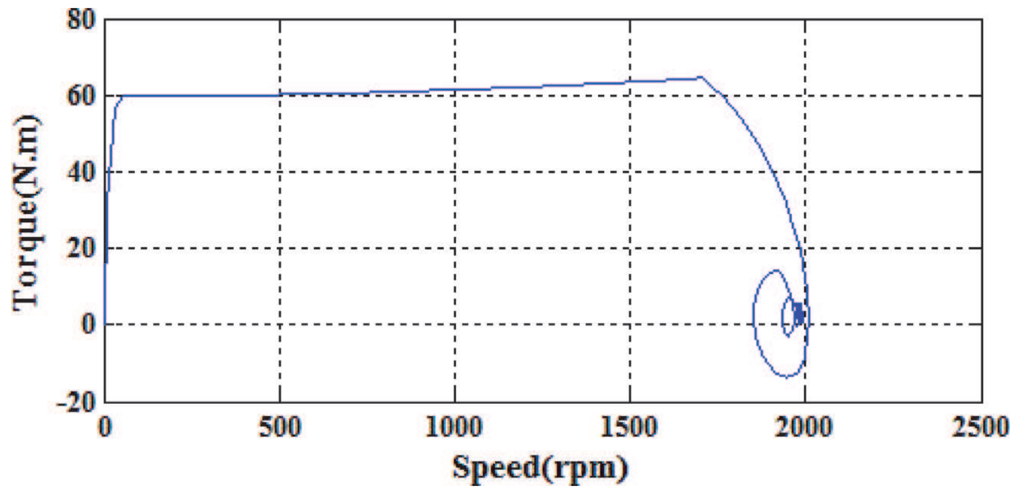

Figure 12.
Torque-speed characteristics of BLDC motor with feedforward WNN-PID controller.

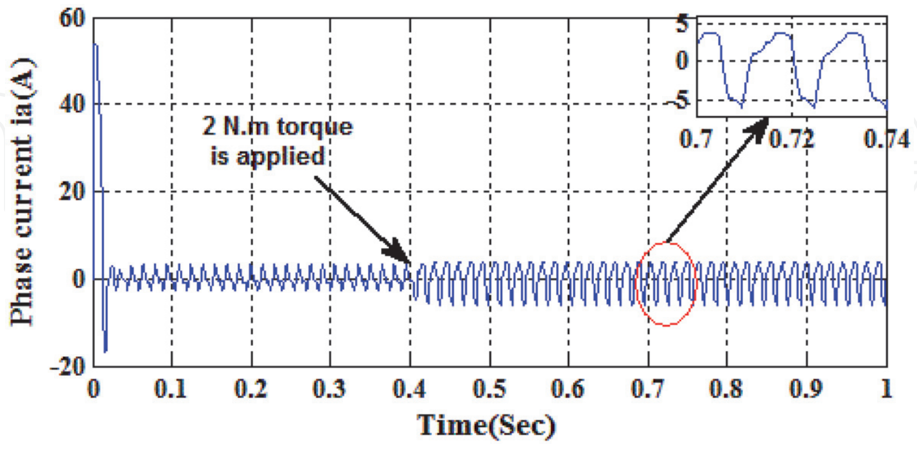

Figure 13.
Phase current i_a of BLDC motor with feedforward WNN-PID controller.

input layer, the hidden layer has four neurons, with one output in the output layer and feedback connection for each layer. In this section, the feedback connection is called "Fully feedback." Besides, the RWNN contains a number of delay samples in the input and output layers as shown in **Figure 16**. The translation and dilation factors, weights and PID parameters are learning on-line to utilize PSO method in

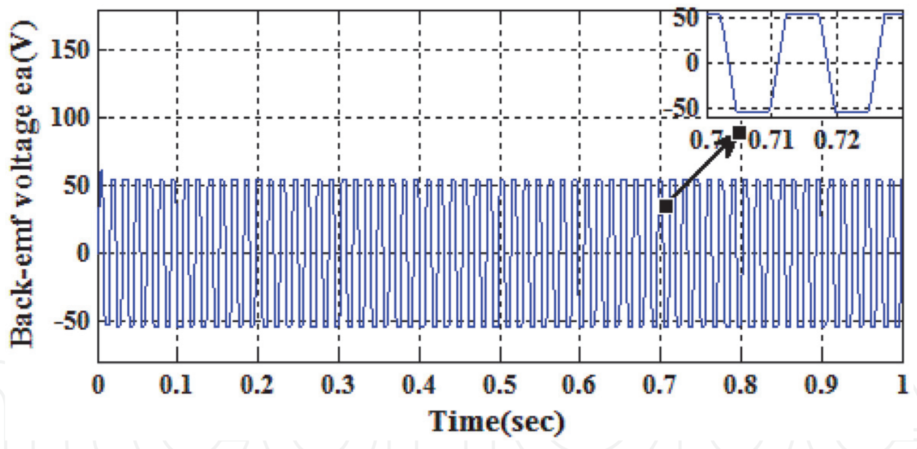

Figure 14.
Phase back-emf e_a voltage for BLDC motor with WNN-PID controller.

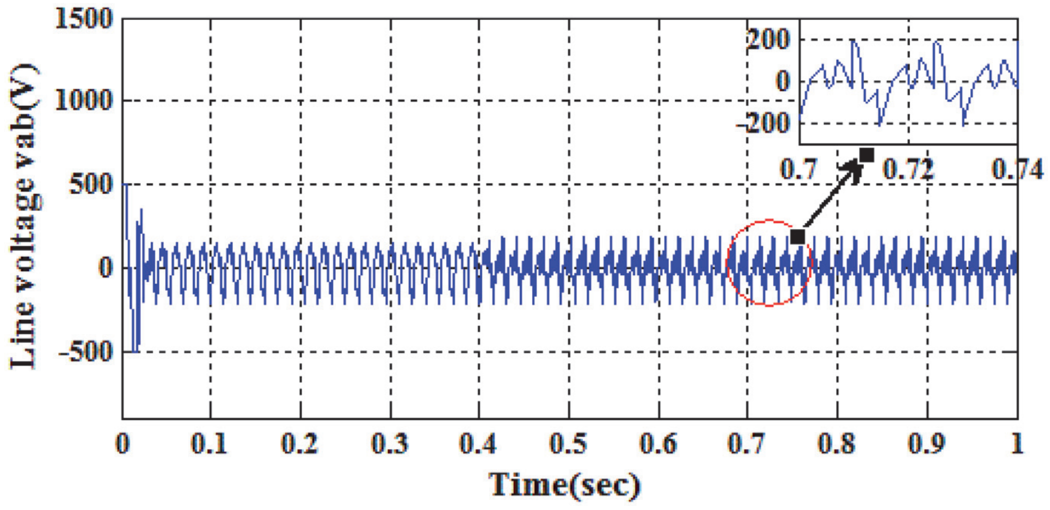

Figure 15.
Line voltage v_{ab} for BLDC motor with feedforward WNN-PID controller.

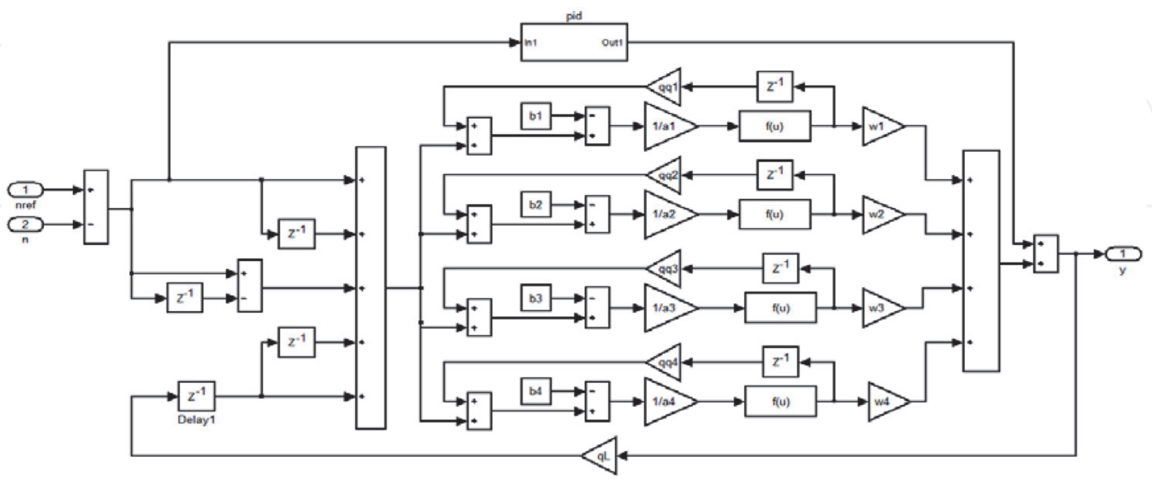

Figure 16.
Simulink model for a proposed recurrent WNN-PID controller.

the same manner used in the previous subsection and the results are given in **Tables 4–6**. The output of WNN is described by Eqs. (7) and (8).

The BLDC motor drive system with RWNN-PID controller is simulated in Matlab/Simulink program as shown in **Figure 6**. The time period that is assumed in

Parameters	K_p	K_i	K_d
Values	4.8256	2.6003	0.0105

Table 4.
PID parameters tuned using PSO for RWNN-PID controller.

WNN dilation parameters				WNN translation parameters				WNN weights parameters			
a_1	a_2	a_3	a_4	b_1	b_2	b_3	b_4	w_1	w_2	w_3	w_4
2.1781	3.0484	3.0181	1.9509	0.1266	5.3694	1.8668	3.2328	0.8769	3.2828	2.7151	0.04124

Table 5.
RWNN parameters tuned using PSO.

Feedback parameters of RWNN				
qq1	qq2	qq3	qq4	qL
0.2957	0.4738	1.0581	4.2460	3.6623

Table 6.
Feedback parameters of RWNN tuned using PSO.

this model is 1 s. The WNN-PID controller can be utilized for speed control in a wide range between 0 and the rated value, with better performance and more flexibility in the controller. **Figure 17** depicts the step change in speed of the BLDC. The motor is started at a speed of 500 rpm and then is changed in step to 500 rpm each 0.2 s. The actual speed of the motor is tracking the desired speed with a good response. The system starts at no load and suddenly a torque 2 N m (full load) is added at t = 0.4 s. **Figure 18** shows the speed response of the BLDC motor at 2000 rpm during no load and load conditions. The developed torque during no load and load conditions is shown in **Figure 19**. The position signal, the torque-speed characteristics, the phase current i_a, Phase Back-emf e_a voltage and line voltage v_{ab} are given in **Figures 20–24**, respectively.

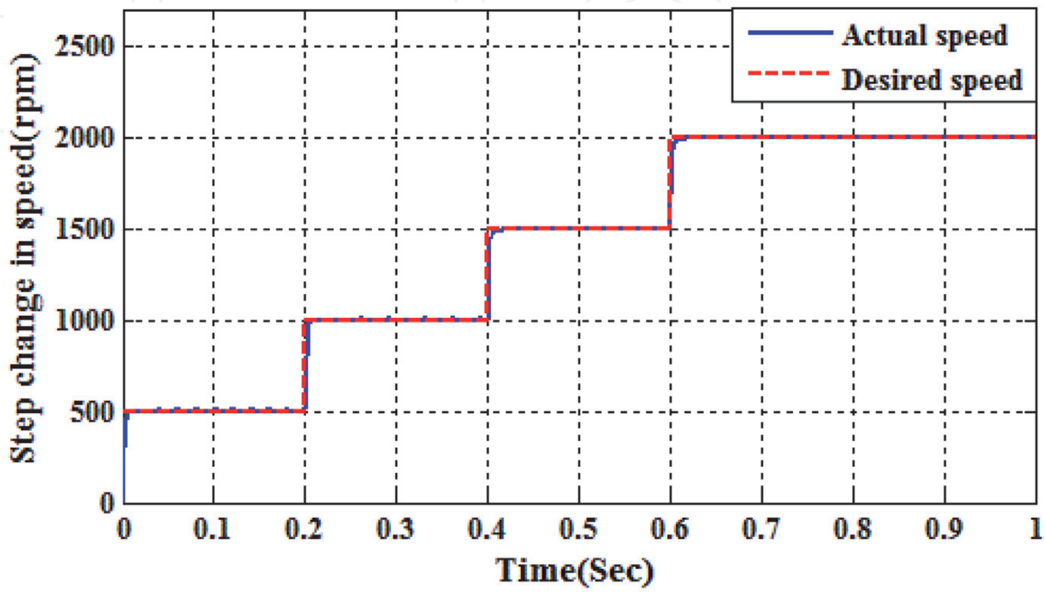

Figure 17.
Step change in speed of BLDC motor with RWNN-PID controller.

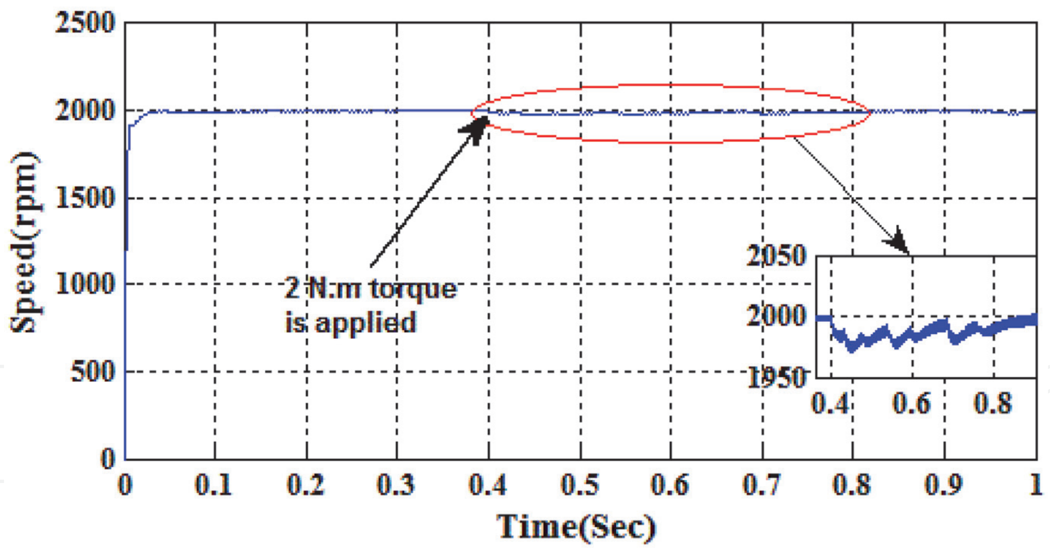

Figure 18.
Speed response of the BLDC motor with RWNN-PID controller.

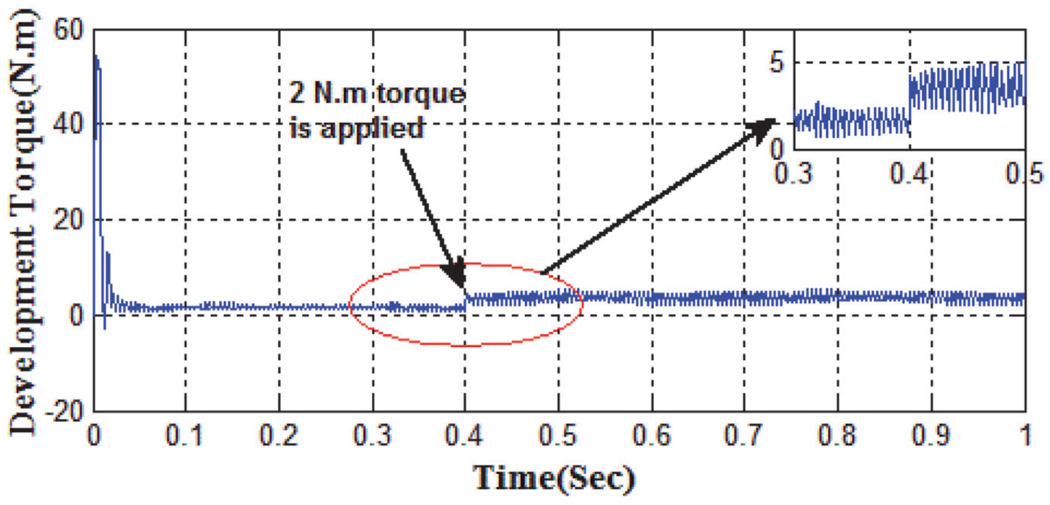

Figure 19.
Development torque of BLDC motor with RWNN-PID controller.

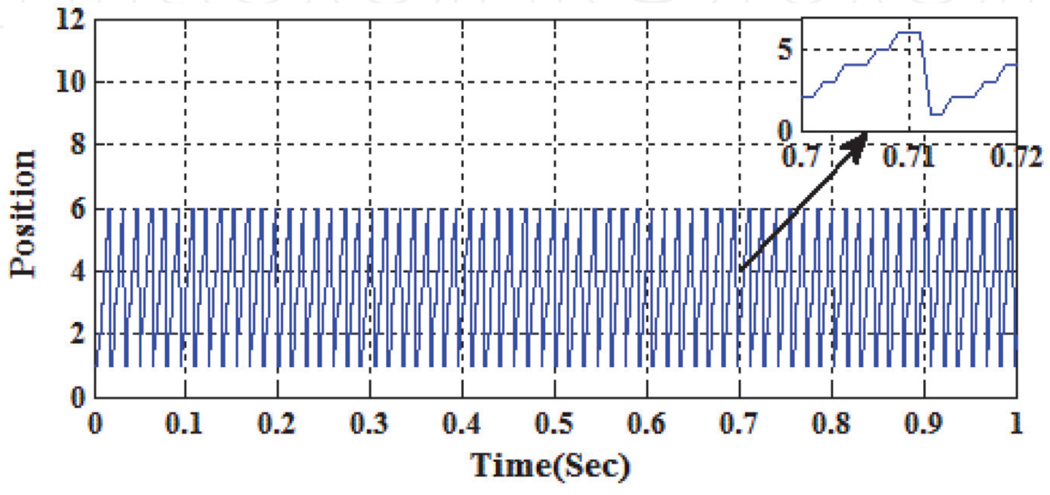

Figure 20.
Position stair signal of BLDC motor with RWNN-PID controller.

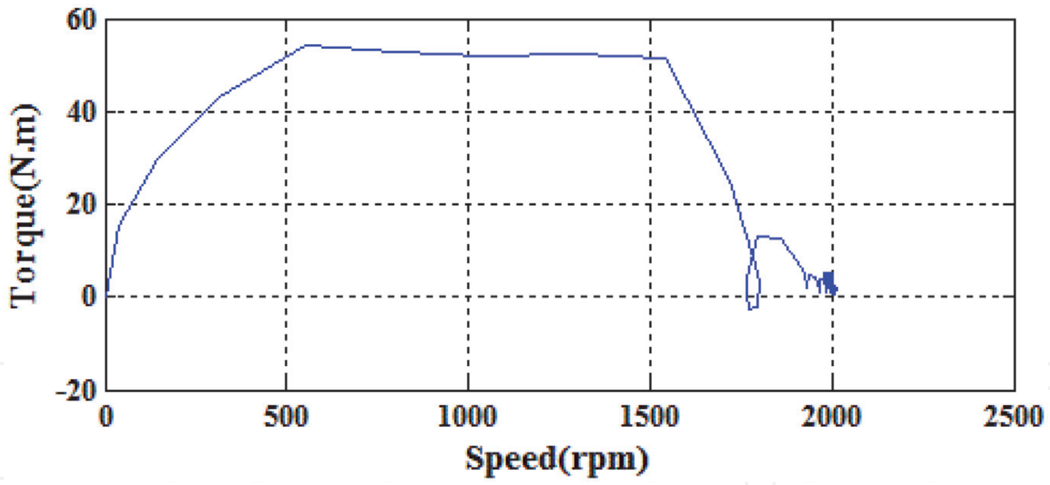

Figure 21.
Torque-speed characteristics of BLDC motor with RWNN-PID controller.

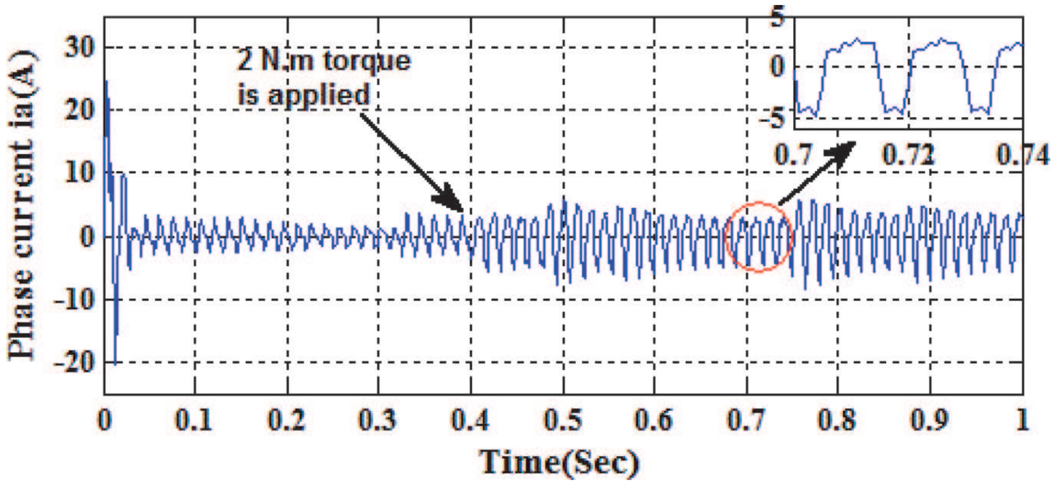

Figure 22.
Phase current i_a of BLDC motor with RWNN-PID controller.

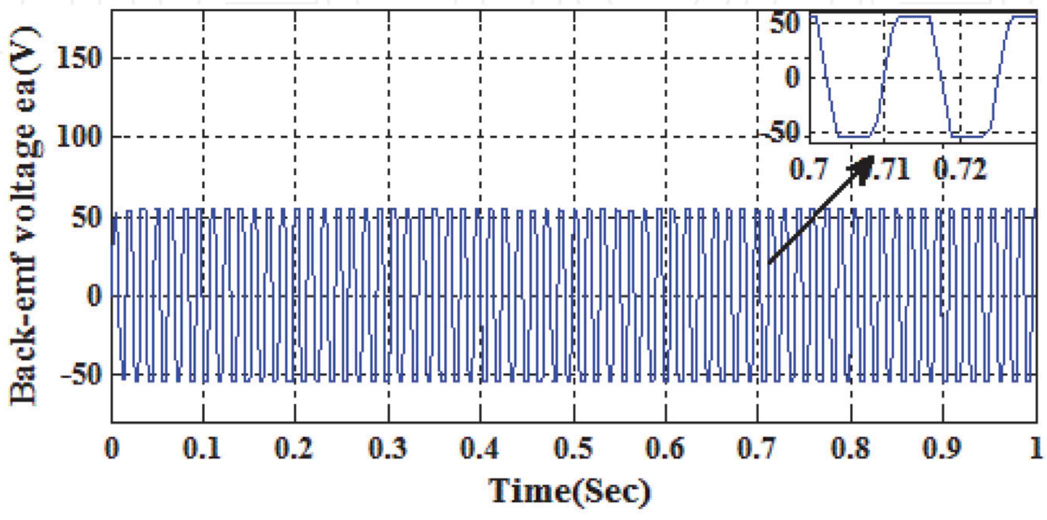

Figure 23.
Phase back-emf e_a voltage of BLDC motor with RWNN-PID controller.

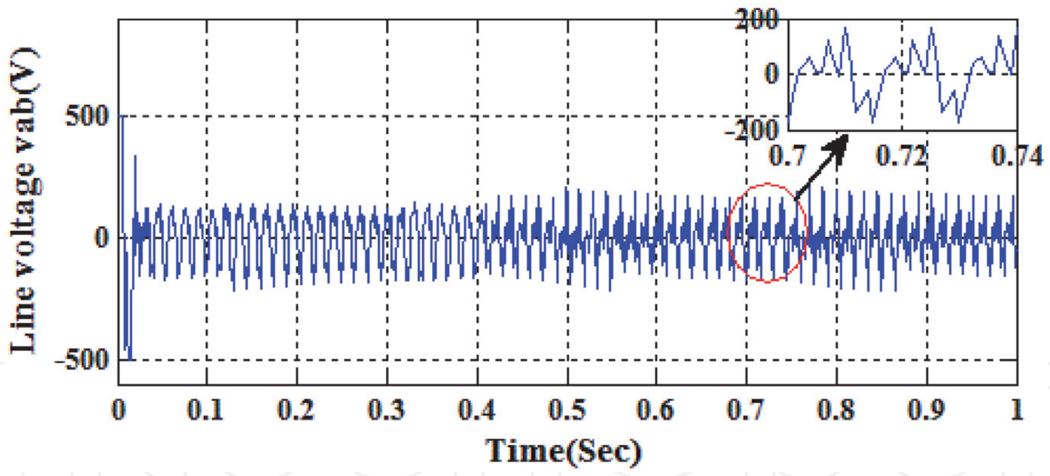

Figure 24.
Line voltage v_{ab} for BLDC motor with RWNN-PID controller.

7. Comparison of two methods for speed control of BLDC motor drive

Comparing among various wavelet neural network schemes shows that the WNN-PID is a preferable method to overcome the nonlinearity in this model with high reliability, more robustness and being good with better performance than the RWNN method as shown in **Table 7**. In addition, the performance of the WNN controller in the real application tends to make the system more robust and less sensitive as well as high precision and excellent flexibility.

Performance	WNN-PID	RWNN-PID
Rise time (s)	0.0035	0.0038
Settling time (s)	0.03	0.04
Steady state error	3×10^{-3}%	2×10^{-3}%
Overshoot	0.12%	Approximately 0%

Table 7.
Performance of speed control of BLDC motor in each methods.

8. Conclusion

In this chapter, the WNN is used with the PID controller to make an adapted controller named as the WNN-PID controller. This controller is utilized to control the speed of BLDC motor in an extensive range and can stock preferable performance than a traditional controller. Two schemes of wavelet neural network are modified for speed control of BLDC motor such as WNN and RWNN. PSO algorithm is utilized for tuning and learning the parameters of the two controllers. The two methods are implemented and tested for different conditions and the performance is compared as shown in **Table 7**. From the simulation results, one can conclude that the proposed WNN controller with PID controller is the best scheme in performance and stability. In addition, using the proposed WNN controller to control the speed of BLDC motor gives better results compared to traditional methods.

Author details

Ameer L. Saleh[1], Adel A. Obed[2], Hamza H. Qasim[3], Waleed I.H. Breesam[4], Yasir I.A. Al-Yasir[5]*, Naser Ojaroudi Parchin[5] and Raed A. Abd-Alhameed[5]

1 Department of Electrical Engineering, College of Engineering, University of Misan, Misan, Iraq

2 Department of Electrical Engineering Technical College, Middle Technical University, Baghdad, Iraq

3 Department of Communication Engineering, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

4 Electrical Department, Basra Oil Training Institute, Basra, Iraq

5 Faculty of Engineering and Informatics, University of Bradford, Bradford, UK

*Address all correspondence to: y.i.a.al-yasir@bradford.ac.uk

References

[1] Padmaraja Y. Brushless DC (BLDC) Motor Fundamentals. Microchip Technology Inc.; 2003

[2] Stefan B. BLDC motor modeling and control—A MATLAB/SIMULINK implementation [master thesis]. Gothenburg, Sweden: Electrical Power Engineering, Chalmers University of Technology; 2005

[3] Rambabu S. Modeling and control of a brushless DC motor [master thesis]. In: Power Control and Drives Technology. Rourkela: National Institute of Technology; 2007

[4] Ulasyar A, Zad HS, Zohaib A. Intelligent speed controller design for brushless DC motor. In: 2018 International Conference on Frontiers of Information Technology (FIT); January; Islamabad, Pakistan. 2019

[5] Saleh AL, Obed AA. Speed control of brushless DC motor based on fractional order PID controller. International Journal of Computer Applications. 2014; **95**:1-6

[6] Obed AA, Kadhim AK. Speed and current limiting control strategies for BLDC motor drive system: A comparative study. International Journal of Advanced Engineering Research and Science. 2018;**5**(2):119-130

[7] Niasar AH, Vahedi A, Moghbelli H. Speed control of a brushless DC motor drive via adaptive neuro-fuzzy controller based on emotional learning algorithm. In: IEEE Proceedings of the Eighth International Conference on Elec trical Machines and Systems; vol. 1. 2005. pp. 230-234

[8] Awadallah MA, Bayoumi EHE, Soliman HM. Adaptive deadbeat controllers for brushless DC drives using PSO and ANFIS techniques. Journal of Electrical Engineering. 2009;**60**(1):3-11

[9] Kumar NS, Kumar CS. Design and implementation of adaptive fuzzy controller for speed control of brushless DC motors. International Journal of Computer Applications. 2010;**1**(27): 36-41

[10] Maohua Z, Changliang X, Yang T, Dan L, Zhiqiang L. Speed control of brushless DC motor based on single neuron PID and wavelet neural network. In: IEEE International Conference on Control and Automation Guangzhou, China; June. 2007. pp. 617-620

[11] Hameed WI, Sawadi BA, Al-Kamil SJ, Al-Radhi MS, Al-Yasir YIA, Saleh AL, et al. Prediction of solar irradiance based on artificial neural networks. Inventions. 2019;**4**:45

[12] David V. Wavelet neural networks [dissertation submitted for the Master Science]. In: Data Analysis, Networks and Nonlinear Dynamics. Department of Mathematics, University of York; 2005

[13] Billings SA, Hua-Liang W. A new class of wavelet networks for nonlinear system identification. IEEE Transactions on Neural Networks. 2005;**16**:862-874

[14] Dhiraj A, Sunil K. Adaptive fuzzy wavelet network control design for nonlinear systems. International Journal of Advanced Technology & Engineering Research (IJATER). Jan. 2013;**3**:148-157

[15] Cheng-Jian L, Hung-Ming T. FPGA implementation of a wavelet neural network with particle swarm optimization learning. Mathematical and Computer Modelling. 2008;**47**:982-996

[16] Gaviphat L. Adaptive self-tuning neuro wavelet network controllers [doctor of philosophy thesis in Electrical Engineering]. Virginia Polytechnic Institute and State University; March 1997

[17] Bhowmik PS, Pradhan S, Prakash M, Roy S. Investigation of wavelets and radial basis function neural network for incipient fault diagnosis in induction motors. In: IEEE International Conference on Circuits, Controls and Communications (CCUBE); December. 2013. pp. 1-5

[18] Shouxin R, Ling G. Application of a wavelet packet transform based radial basis function neural network to analyze overlapping spectra. In: IEEE Congress on Image and Signal Processing; China; May. 2008. pp. 228-232

[19] Hamza MI. A wavelet network control scheme for path tracking of mobile robot [master thesis in Electrical Engineering]. University of Basrah, Aug. 2013

[20] Sung Jin Y, Park JB, Choi YH. Direct adaptive control using self recurrent wavelet neural network via adaptive learning rates for stable path tracking of mobile robots. In: IEEE American Control Conference, Vol. 1. 2005. pp. 288-293

[21] Hameed WI, Saleh AL, Sawadi BA, Al-Yasir YIA, Abd-Alhameed RA. Maximum power point tracking for photovoltaic system by using fuzzy neural network. Inventions. 2019;**4**:33

[22] Obed AA, Saleh AL. Speed control of BLDC motor based on recurrent wavelet neural network. Iraqi Journal of Electrical and Electronic Engineering. 2014;**10**(2):118-129

[23] Saleh AL, Obaid BA, Obed AA. Motion control of linear induction motor based on optimal recurrent wavelet neural network-PID controller. International Journal of Engineering & Technology. 2018;**7**(4):2028-2034

[24] Obed AA, Saleh AL, Kadhim AK. Speed performance evaluation of BLDC motor based on dynamic wavelet neural network and PSO algorithm. International Journal of Power Electronics and Drive System (IJPEDS). 2019;**10**(4):1742-1750

[25] Mehdi N, Hossein N, Malihe M. A PSO-based optimum design of PID controller for a linear brushless DC motor. Proceedings of World Academy of Science, Engineering and Technology. 2007;**20**:211-215

[26] Reddy MB, Obulesh YP, Raju SS. Particle swarm optimization based optimal power flow for volt-var control. ARPN Journal of Engineering and Applied Sciences. 2012;7:20-25

[27] Effatnejad R, Bagheri S, Farsijani M, Talebi R. Economic dispatch with particle swarm optimization and optimal power flow. International Journal on "Technical and Physical Problems of Engineering" (IJTPE). 2013;**5**:9-16

[28] Mohammed HJ, Abdullah AS, Ali RS, Abd-Alhameed RA, Abdulraheem YI, Noras JM. Design of a uniplanar printed triple band-rejected ultra-wideband antenna using particle swarm optimisation and the firefly algorithm. IET Microwaves, Antennas and Propagation. 2016;**10**(1):31-37

[29] Portillo AA, Michael F, Chunjiang Q. Particle swarm optimization for PID tuning of a BLDC motor. IEEE International Conference on Systems, Man, and Cybernetics; October. 2009. pp. 3917-3922

[30] Sivanandam SN, Visalakshi P, Bhuvaneswari A. Multiprocessor scheduling using hybrid particle swarm optimization with dynamically varying inertia. International Journal of Computer Science & Applications. 2007;**4**:95-106

[31] Soni YK, Rajesh B. BF-PSO optimized PID controller design using ISE, IAE, IATE and MSE error criteria. International Journal of Advanced

Research in Computer Engineering & Technology (IJARCET). 2013;**2**: 2333-2336

[32] Mohammed HJ, Abdullah AS, Ali RS, Abdulraheem YI, Abd-Alhameed RA. Performance comparison of particle swarm optimization, and genetic algorithm in the design of UWB antenna. Journal of Telecommunications. 2014;**27**(2):22-26

[33] Taeib A, Ltaeif A, Chaari A. A PSO approach for optimum design of multivariable PID controller for nonlinear systems. In: International Conference on Control, Engineering & Information Technology (CEIT'13) Proceedings Engineering & Technology. Vol. 2. 2013. pp. 206-210

[34] Mohammed HJ et al. Evaluation of genetic algorithms, particle swarm optimization, and firefly algorithms in antenna design. In: 2016 13th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Lisbon. 2016. pp. 1-4

[35] Sharaf AM, El-Gammal AAA. A novel particle swarm optimization PSO tuning scheme for PMDC motor drives controllers. In: IEEE International Conference on Power Engineering, Energy and Electrical Drives; March. 2009. pp. 134-139

[36] Xie W, Wang J-S, Wang H-B. PI controller of speed regulation of brushless DC motor based on particle swarm optimization algorithm with improved inertia weights. In: Mathematical Problems in Engineering. 2019. pp. 1-12

[37] Saleh AL, Mohammed MJ, Kadhim AS, Raadthy HM, Mohammed HJ. Design fuzzy neural petri net controller for trajectory tracking control of mobile robot. International Journal of Engineering & Technology. 2018;**7**(4):2256-2262

[38] Saleh AL, Obed AA, Al-Yasir YIA, Elfergani ITE, Rodriguez J, Clarke RW, et al. Anti-windup scheme based on 2DOF-PIλDμ controller for velocity tracking of linear induction motor. International Transactions on Electrical Energy Systems. 2019;**40**:1-17

[39] Obed AA, Kadhim AK. Multi-resolution wavelet PID speed and current controllers of BLDC motor based on invasive weed optimization technique. International Journal of Applied Engineering Research. 2018; **13**(8):6234-6243