Direct Adaptive Control of DC Motor Using Radial Basis Function Neural Network

A. Shebani
College of Computer Technology – Zawya
amerelshibani@yahoo.com

Abstract

A variety of a non-linear complex system such as robotics and many other electrical systems require modeling and control. A direct adaptive control method is considered in this work for controlling of a DC motor system. The proposed direct adaptive control scheme is implemented in MATLAB and the simulated results show that the design requirements are achieved by using 10 radial basis functions with the optimum number of centres being determined by using K-means clustering algorithm. The optimum value for the learning factor is determined through simulation tests. Least mean square method is used to calculate the weight values for output layer for adaptive control scheme with radial basis function network. The direct adaptive control scheme provides a systematic approach for the automatic adjustment of controllers in real time, therefore, it can be used to maintain a desired level of control system performance when the system parameters are changing in time.

Keywords direct adaptive control, DC motor, radial basis function neural network, feedforward control.

1.INTRODUCTION

Radial basis function neural network (RBFNN) has a simple architecture consisting of one input layer, one output layer and only one array of hidden nodes called centers [1]. The training of the RBFNN is very fast because the learning in the hidden layer means selection of centers and widths and the learning in the output layer means selection of the weights [2]. The K-means algorithm is reliable, simple, offers fast convergence and can handle large data sets. But it is sensitive to the initial cluster centers meaning that average values of all objects in the same class are considered as cluster centers. The least mean squares (LMS) algorithm is the common algorithm which can be used to adapt the weights of RBFNN because of its simplicity and reasonable performance [1]. The RBFNN has good nonlinear function approximation ability so it is widely used to solve many problems related to modeling and control of nonlinear systems. The Euclidean distance method could be employed to measure the width of the Gaussian function, which is a positive constant that represents the standard deviation of the function [2].

The control of nonlinear systems using RBFNN studied, where the RBFNN has a good generalization, strong tolerance to input noise, and online learning ability. The properties of RBFNN make it very suitable to design flexible control systems. The main advantage of a direct adaptive control scheme over an indirect adaptive control scheme is that in a direct adaptive control scheme there is no need for explicit system identification. In indirect adaptive control scheme, the system is generally identified off-line from its input-output data and the controller is designed based on the identified system model [3].

The mathematical model of the speed control for the DC motor is developed and implemented in MATLAB. Then the proposed direct adaptive control scheme is implemented in MATLAB and simulations are used to determine the optimum values for the learning factor and number of RBFs which should be employed to achieve the design requirements [4-9]

2.RADIAL BASIS FUNCTION NEURAL NETWORK (RBFNN)

A. *Radial Basis Function neural Network Structure*

The RBFNN has a feed forward structure consisting of three layers as shown in Figure 1.

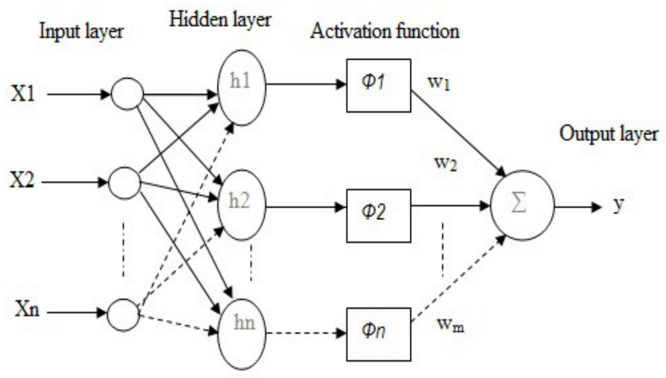

Figure 1: Radial Basis Function Neural Network Structure

The input layer passes the input data to the hidden layer. The hidden layer consists of an array of nodes and each node contains a parameter vector called a center. The output layer is only a set of weighted linear combinations of the activation function. The only adjustable parameters in the output layer are the weights.

The output vector y is given by:

$$y = \sum_{j=1}^{N} W_j \emptyset_j \qquad (1)$$

Where (W_j) is the weight of the (j^{th}) node and $(\emptyset_j)$is the activation function. The hidden node calculates the distance between the center and the RBFNN input vector, and then passes the result through the nonlinear activation function *($\emptyset$)* to the output layer.

The Gaussian activation function can be written as

$$\emptyset_j(x) = exp\frac{-\sum_{j=1}^{n}(x_i - c_j)^2}{\sigma_j^2} \qquad (2)$$

i = 1,2,3,....,n , j = 1,2,3,....,m

Where $(\emptyset_j)$ is the output of the *(j^{th})* unit in the hidden layer, (X_i) is the input data to the network, (C_j) is the center of the j^{th} unit in the input space, (σ_j) is the width of the Gaussian function, (*m*) is the number of centers and (n) the dimension of the input space. The major requirement is that the function must tend to zero quite rapidly as the distance increases between the input *X* and center *C*. This can be assured by using the Gaussian exponential function.

The Radial Basis Function Network consists of three important parameters, centers (C), widths (σ) and weights (W). The value of these parameters is generally unknown and may be found during the learning process of the network. There are a variety of methods to allow the RBF network to learn. These processes are generally divided into two stages, as each layer of the RBF perform a different task: The first learning stage involves selecting the centers and the widths in the hidden layer, and the second stage is to adjust the weights in the output layer [1].

The major problem is how to select an appropriate set of RBFNN centers. To overcome this problem, the network requires some strategy for selecting the adequate set of centers, hence clustering algorithm have been used extensively. The objective of clustering algorithm is to categorize or cluster the data. The classes must be found from the correlation of an input data set. So, the clustering is a way of grouping similar patterns and separating dissimilar (different) ones. Assume there is a set of data to be used as input to an RBFNN and no information is known about the number of classes that may be present in this set. Clustering in such a case involving identifying the number of classes and assigning individual datum membership of these classes. The vectors in the same cluster are similar which means that they are close to each other in the input space. There are many clustering algorithms, but the most popular algorithm is K-means clustering algorithm [1]. The choice of subsets of data as centers for the radial basis function is a very important task, since network performance relies upon good generalization [1, 10]. The K-means clustering algorithm is used for selecting the number of the centers; this algorithm has been used in this work because of its simplicity and ability to produce good results, the basic steps of K-means algorithm shown in Fig. 2.

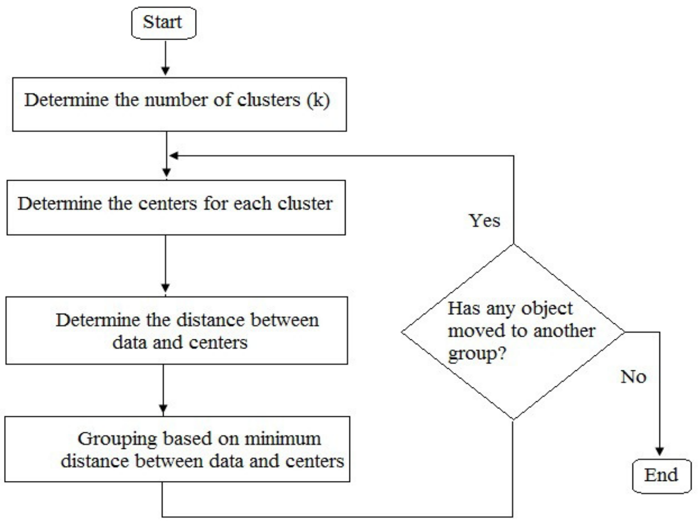
Start
Determine the number of clusters (k)
Determine the centers for each cluster
Determine the distance between data and centers
Grouping based on minimum distance between data and centers
Has any object moved to another group?
Yes
No
End

Figure 2: Classical K-means clustering algorithm

The next task would be the proper selection of the width between the centers. The width of the Gaussian function is a positive constant that represents the standard deviation of the function. RBFNN with the same σ in each hidden unit have the capability of universal approximation. The most common method used is the Euclidean distance measure [2]. This method is widely used because it is simple to calculate and more reliable. The shortest distance between vector *X* and vector *C* is the Euclidean distance which is defined as:

$$E_{dist} = \sqrt{\sum_{i=1}^{n} (Xi - cj)^2} \qquad (3)$$

Where n the vector dimension, and Edist isthe Euclidean distance.

In this work the feed-forward neural networks was considered in direct adaptive controller by RBFNN, the network uses the input vector to produce the estimate actual RBFNN output $y(t)$ compared with the true desired output of the network $y_{d(t)}$to generate an error vector e(t). If there is no difference, no learning takes place; otherwise, the weights are adjusted to reduce the error [6]. The Least Mean Squares rule for adapting the weights can be written as:

$$W_j(n+1) = W_j(n) + \mu\left(y_d(t) - y(t)\right)g_j(n) \qquad (4)$$

$$j = 1,2,3,\ldots,H, \text{ and } n = 1,2,3,\ldots,N.$$

Where H is the number of centres, *N* is the number of inputs to the network,$y(t)$ is the output of the RBFNN,$y_d(t)$ is the desired output of the network, $g(n)$ are the Gaussian output, $W_j(n)$ are the previous weights originally set to zero, $W_j(n+1)$ are the updated weights, and μ is the learning rate, where $0 < \mu \leq 1$ is a positive gain factor term that controls the adaptation rate of the algorithm.

The adaptation algorithm described by equation (4) is illustrated in Figure 3.

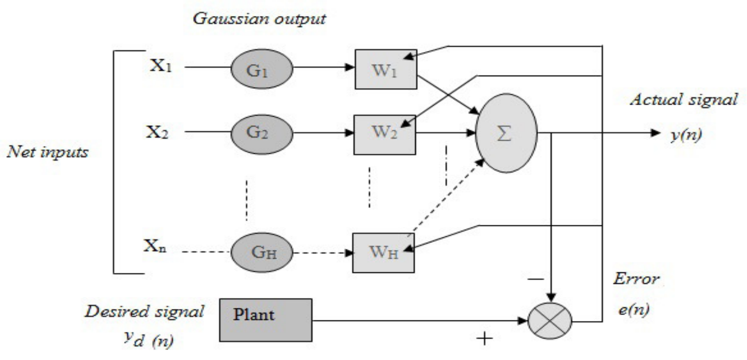

Figure 3: The adaptive LMS algorithm [6]

The training process of LMS algorithm consists of the following steps:

- Apply the input vector *(X)* from the training data set to the input layer.
- Compute the output of the hidden layer.
- Compute the RBF network output vector *(y)*. Compare this to the desired vector *(y_d)*, and then adjust the vector W so as to reduce the difference.
- Repeat steps 1 to 3 for each vector in the training set.
- Repeat steps 1 to 4 until *($y - y_d$)* tends to zero or other terminating conditions occur (in our work until the speed of DC motor reached to 1 rad/sec).

K-means clustering algorithm is used to resolve cluster problems because this algorithm is simple and fast for large data collection. Fig.4 shows the steps of chose the initial cluster centers by using K-Means [11, 12, 13].

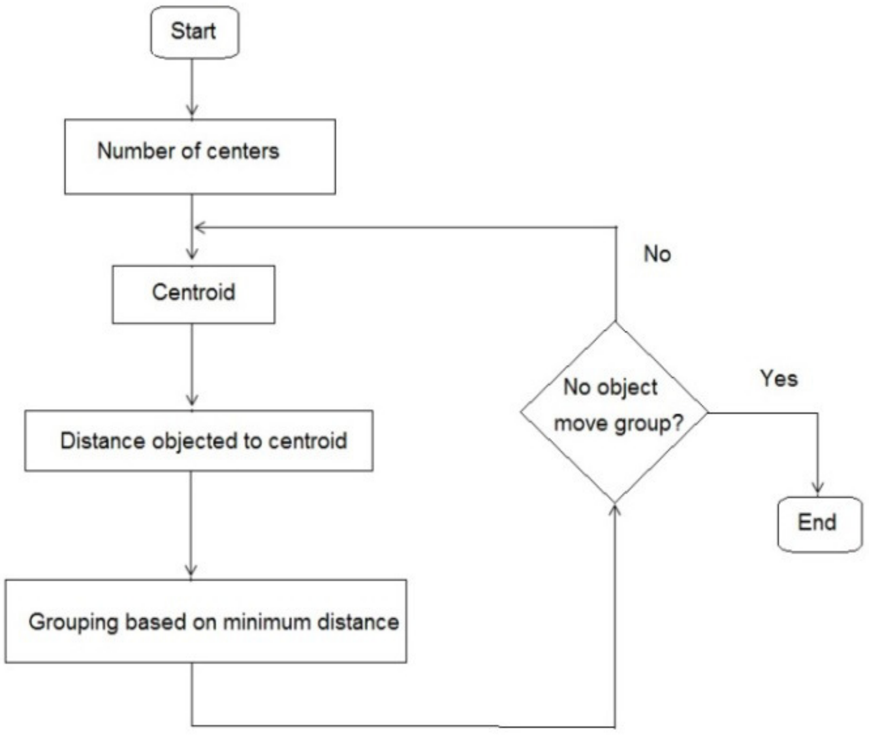
Start
Number of centers
Centroid
Distance objected to centroid
Grouping based on minimum distance
No object move group?
No
Yes
End

Figure 4:K-means clustering algorithm

3.DIRECT ADAPTIVE CONTROL USING RBFNN

A direct adaptive control (or the on-line) method is used to overcome the problems associated with the off-line approach. In this method, the neural network learns during the on-line feed forward control period [1, 2, 14]. As shown in Fig. 5, the controller network is placed in front of the plant, whereby the net output control signal $Uc(t)$ is an input to the plant.

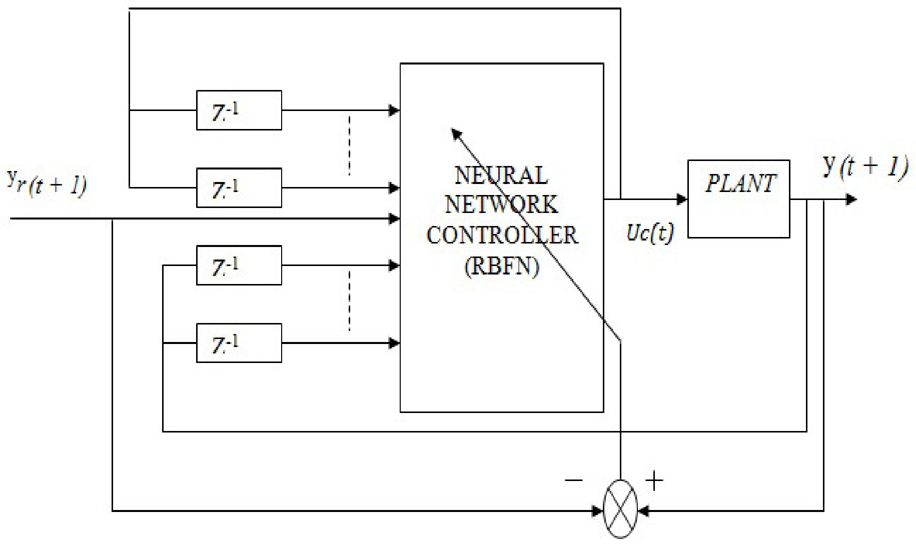

Figure 5: Direct adaptive control configuration using RBFNN

The control signal can be realized as:

$Uc(t) = RBFNC[X(t)](5)$

Where:

The desired reference signal y_r *(t + 1)* was used instead of the unknown y *(t + 1)*, this can be rewritten as:

$$X(t) = [y_r\,(t+1), y(t), \ldots.., y(t-n+1), U(t-1), ., U(t-m+1)]^T (6)$$

During the training of the network, the centers are assumed selected, presumed to have been prior to training and are given as:

$$C_{ji} = [C_{11}C_{12}\ \ldots\ldots.C_{1n}]^T \qquad (7)$$

i = 1, 2… n & j = 1, 2… m

Where n is the network input space dimension and m is the number of centers (hidden units) with $m \le n.$ It is recommended that the width between the centers is also selected, remembering that the same number of centers and the width value obtained during the modeling are used for the purpose of controlling the system. Therefore, only the weights of the networks are adapted using the LMS algorithm to decrease the error between the reference signaly_r *(t + 1)* and the actual output *y (t + 1)* in every iteration step. After the learning process is finished, the connection weights between the output layer and the hidden units will have been adjusted so that $y_r\,(t + 1) \cong y\,(t + 1).$

The learnt inverse model may then be able to take a desired response and calculate an appropriate control signal $U_c(t)$, equation (6) can be rewritten as:

$$X(t) = [y_r\,(t+1), y_r\,(t), ., y_r\,(t-n+1), U(t-1), ., U(t-m+1)]^T (8)$$

The direct adaptive control method is considered in our work for controlling the non-linear dynamic models due its simplicity and robustness, especially when the proper parameters are selected. One limitation which might affect the performance of the controller is the choice of the learning factor (μ) which controls the convergence of the LMS algorithm. Care must be taken by the user when selecting this value; the correct value can be selected according to the experience or by a trial and error procedure.

4.CASE STUDY- DC MOTOR SPEED CONTROL

A common actuator in control systems is the DC motor. It directly provides rotary motion and, coupled with wheels or drums and cables, can provide translational motion [8,9,15]. The electric equivalent circuit of the armature and the free-body diagram of the rotor are shown in the following Fig. 6.

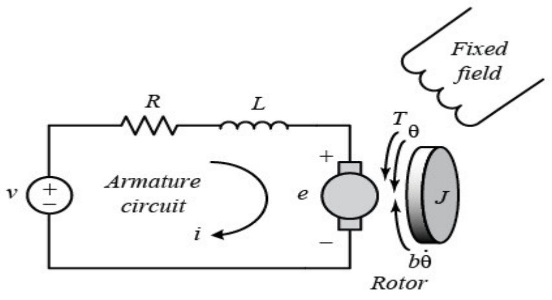

Figure6: Electric circuit of DC motor speed control

The DC motor speed control system can described sung the following transfer function:

$$\frac{\dot{\theta}(s)}{V(s)} = \frac{K}{(J\,S+b)(L\,S+R)+K^2}\left(\frac{rad/sec}{v}\right) \qquad (9)$$

Where:

J moment of inertia of the rotor (*J* = 0.01 kg.m^2);

K motor torque constant (*K*= 0.01 N.m/Amp);

R electric resistance (*R* = 1 Ohm);

L Electric inductance (*L*= 0.5 H).

Convert the equation (9) to z domain, and then, the difference equation to the DC motor model found as:

$$\dot{\theta}(k) = 1.885\,\dot{\theta}(k-1) - 0.8868\,\dot{\theta}(k-2) + 0.00009\,V(k-1) + 0.0000092\,V(k-2) \qquad (10)$$

Where: $\dot{\theta}$ is the rotational speed (output) and V is the armature voltage (input).

The design requirements are:

1. The speed of DC motor steady at 1 rad/sec.
2. Settling time 0.2 Sec
3. Overshoot less than 2%.

SIMULATION RESULTS

A. Selection of centers for proposed adaptive control scheme with RBFNN

This section covers the investigation of centers selection for a direct adaptive controller with RBFNN. Fig. 7 shows the simulation results for the applied cases on the DC motor model.

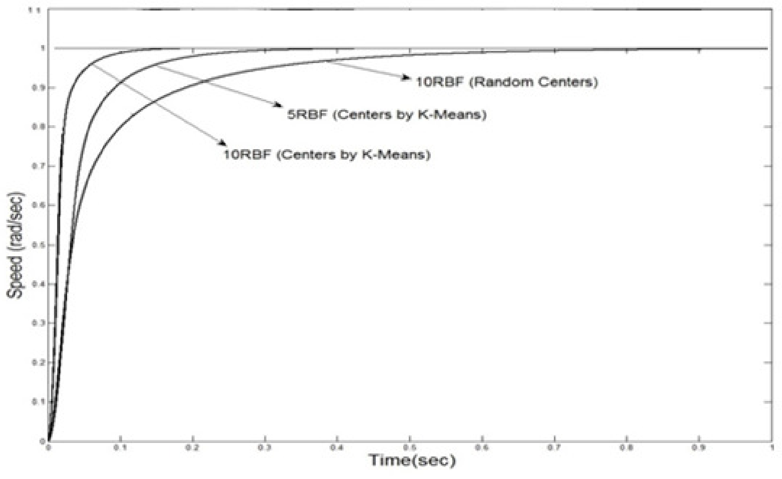

Figure7: Simulation results using RBFNN

When 10 random centers are considered for the RBFNs controlling the DC motor model, the system response is slow, settling time is large (0.8 Sec) and the design requirements are not satisfied. The second case refers to five centers determined with K-Means clustering algorithm and the system response is faster and the settling time is 0.3 Sec. The third case considers 10 centers determined with K-means clustering algorithm and the response becomes faster and settling time is 0.15 Sec, the steady value for the DC motor speed is 1 rad/sec and the overshoot is zero therefore the design requirements are satisfied. So this is the best method which should be used for the selection of centers. Simulation results show that the value of overshoot is zero in studied cases.

B. Selection of the learning factor for proposed RBFNN

This section explains the selection of the learning factor on the convergence algorithm. The values of learning factor are μ = 0.01 and 0.001 (Figure 8) and the simulated results show that any increase in the controller learning rate is leading to an increase in the desired response and faster convergence. On the other hand it causes an overshoot on the control signal itself. However, a small value ofμ causes a slow convergence rate. Therefore a careful selection of the learning rate values is requested.

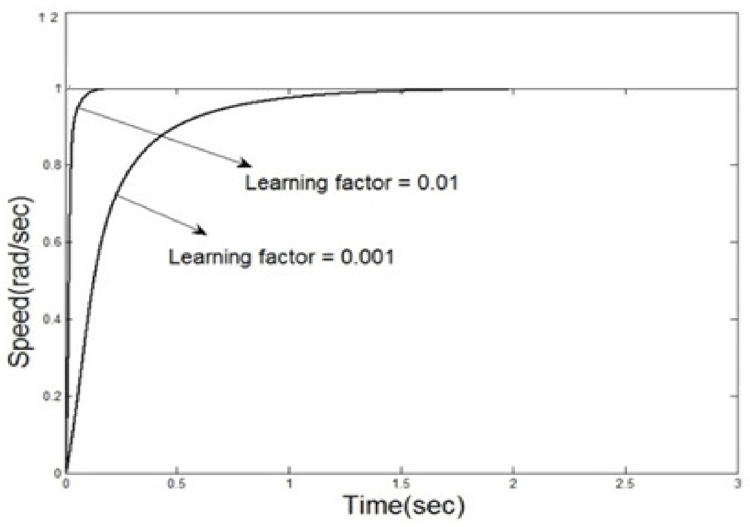

Figure 8: Effects of learning factor μ on simulated DC motor response

C. Weight adaptation for proposed RBFNN

This section presents the adaptation of the weights of the RBFN output layer by using LMS algorithm (Figure 9). The value of the weight is changed until the speed of the system response of the DC motor model reaches the steady value of 1 rad/sec. So the final value for the weight of the RBFNN output layer is 1.58 and it is obtained by applying the LMS algorithm for weight adaptation.

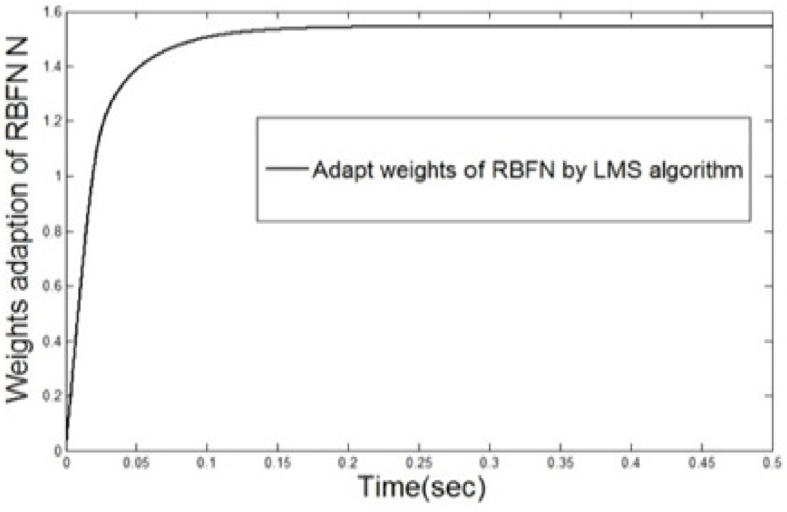

Figure 9: LMS weights adaptation of RBFNN used for the direct adaptive control of DC motor speed

5. CONCLUSION

The proposed direct adaptive control scheme is implemented in MATLAB and the simulated results show the choice of learning factor is very important in order to obtain faster convergence and high performance of the adaptive controller.It is observed that the larger the learning rate, the faster the algorithm will adapt. However, if the learning rate is too large, the output may not be robust. Finally, the simulation results show that the adaptation of the weights of the RBFNN output layerhas significant effects on the performance of the direct adaptive controller. The direct adaptive control scheme using radial basis function neural network can used to control of nonlinear system (DC motor) effectively. The results have that the on-line RBFNN controller is an effective and robust algorithm; one of the advantages is the adaptivety of the algorithm i.e., the quick modification in the behavior of the controller when there are changes in the dynamics of the process.

REFERENCES

[1] L. Jinkun, “Radial Basis Function (RBF) Neural Network control for Mechanical Systems”, Springer Verlag, 2013.

[2] Y. Hu and J. Hwang, “Hand book of Neural Network Signal Processing”, by CRC press LLC, 2002.

[3] A. Tamer, “Direct Adaptive Control of Unknown Nonlinear Systems Using Radial Basis Function Networks with Gradient Descent and K-means”, International Journal of Computer Theory and Engineering, Vol. 3, No.6, pp. 775-784, December 2011.

[4] J. Fathalla and A. Gumaidh, “The controlling of nonlinear systems using Radial Basis Function”, 10th International conference on Sciences and Techniques of Automatic control & computer engineering December 20-22, pp. 1430-1433 , Hammamet, Tunisia, 2009.

[5] Y. Hasegawa, “Control Problems of Discrete-Time Dynamical Systems”, Springer, 2013.

[6] P. Petko, K. Mihail and G. Da-Wei, “Robust Control Design with Matlab, Mixed media product”, Springer, 2013.

[7] D. Joseph, “Schaum's Outline of Feedback and Control Systems”, McGraw Hill, 2013.

[8] D. Joseph, “Modern Control Systems”, Pearson, 2010.

[9] C. Jongeun, “Control systems, modeling of DC motors, Department of mechnical engineering, University of British Columbia ”, Lecture 8, ME451, pp. 1-15, Fall 2008.

[10] T. Kardi, “K-Means Clustering Tutorial”, http:\\people.revoledu.com\kardi\tutorial\kMean, 2007.

[11] M. Wang and S. Yin, “Improved K-Means clustering based on Genetic Algorithm International Conference on Computer Application and System Modelling (ICCASM), IEEE

Catalogue Number: CFP1004K-PRTT, ISBN: 978-1-4244-7235-2 Taiyuan, China, 22 – 24 October, pp. 1-737, 2010.

[12] A. Bashar and M. Sung, “Initializing K-Mean using Genetic algorithms”, World academy of science, Engineering and technology, 2009.

[13] Z. Chunf and F. Zhiyi, An improve K-means clustering Algorithm, College of computer since and technology, Jilin University, China, Journal of computational science, Vol. 10, pp. 193–199, 2013.

[14] S. Haykin, “Neural Networks, a comprehensive foundation”, Prentice- Hall Inc., second edition, 1999.

[15] S. George and L. Feng, “Adaptive neural network control by adaptive Interaction”, Farmington Hills, Michigan 48335, Detroit, Michigan 48202, USA, 2000.