International Journal of Mechanical Engineering and Technology (IJMET)
Volume 9, Issue 11, November 2018, pp. 670–683, Article ID: IJMET_09_11_067
Available online at http://www.iaeme.com/ijmet/issues.asp?JType=IJMET&VType=9&IType=11
ISSN Print: 0976-6340 and ISSN Online: 0976-6359

  **Scopus** Indexed

RADIAL BASIS FUNCTION NEWRAL NETWORK AND GENETIC ALGORITHM IN TRAJECTORY TRACKING CONTROL OF THE OMNI-DIRECTIONAL MOBILE ROBOT

Van Huong Dong
Ho Chi Minh City University of Transport, Ho Chi Minh City, Vietnam

Thanh Tung Pham and Kieu Mai Le Thi
Faculty of Electrical and Electronic Engineering Technology
Vinh Long University of Technology Education, Vinh Long, Vietnam

Chi Ngon Nguyen and Chi Cuong Tran
College of Engineering, Can Tho University, Can Tho, Vietnam

ABSTRACT

The paper's aim is to combine a radial basis function (RBF) neural network and genetic algorithm in trajectory tracking control of the Omni-directional mobile robot.The radial basis function neural network is considered as an adaptive controller in the adaptive sliding mode control law. This is self-learning, selforganizing, and adaptive, possess fast training speed, and global convergenceneural network.The genetic algorithm is used to optimize the number of neurons in the hidden layer, centers, widths and initial weights of the radial basis function neural network. After optimizing, the radial basis function neural network is online trained by Quasi - Newton algorithm. The simulation results in MATLAB/SIMULINK show that the proposed controller is efficient, the response of the Omni-directional mobile robot in simulation model converge to reach the trajectory with steady-state error is about $0.001 \pm 0.0005(m)$, *and the overshoot is about* $0.15 \pm 0.05(\%)$.

Keywords: Genetic algorithm, radial basis function neural network, Quasi-Newton, Omni-directional mobile robot, optimization.

Cite this Article Van Huong Dong, Thanh Tung Pham, Kieu Mai Le Thi, Chi Ngon Nguyen and Chi Cuong Tran, Radial Basis Function Newral Network and Genetic Algorithm in Trajectory Tracking Control of the Omni-Directional Mobile Robot, International Journal of Mechanical Engineering and Technology, 9(11), 2018, pp. 670–683.
http://www.iaeme.com/IJMET/issues.asp?JType=IJMET&VType=9&IType=11

1. INTRODUCTION

RBF neurons have been studied and used in many fields such as identification, classification, approximation... becauseit hassimple structure, good efficiency, fast convergence. This is a self-learning, self-organizing, adaptive, fast-training, and global convergence network [1]. However, the difficulty in training RBF networks is to select the appropriate number of hidden layer's neurons, centers, thresholds, and connection weights [1 - 2]. The use of many hidden layer's neurons will cause long learning and increase the complexity of the network [3], this in turn caused the network don't able to fully learn the sample data [4]. This parameter is usually chosen by trial-and-error method[5]. The centersof the network also affect the performance of the RBF network. If the centersare inappropriately chosen, the efficiency of RBF neural network will be difficult to achieve. If the centers are too close, they will create relative linear correlations, otherwise they will not meet linear processing requirements. Too much of the centers will lead to overflow, if the centersare too few they are difficult to classificate [2]. In addition, the selection of threshold values and initial initialization weights also affect network efficiency.

Genetic algorithm was introduced by J. Holland based on Darwin's natural evolutionary theory [6]. This algorithm is developed from natural selection and computational mechanisms. This is a search algorithm with the characteristics of high parallel search, randomization, adaptation [2, 7, 8]. This algorithm has become a popular tool for many fields of research such as system control, control design, science and technology [9]. Specifically, research [3] presented steps to optimize RBF networks using genetic algorithms and was experimented on the UCI and Iris dataset. Research [4] used genetic algorithms to simultaneous optimize the number of hidden layer's neurons and RBF network parameters. The proposed algorithm was experimented on the UCI dataset and solved the Mackey-Glass differential equation. Research [10] used genetic algorithms to optimize the number of hidden layer's neurons and parameters of RBF networks. The proposed algorithm was experimented on three standard time models: Box-Jenkins, Wolf's sunspot data and Mackey-Glass. Research [11] used a genetic algorithm to optimize the structure and weights of RBF networks by crossbred coding, then use the Pseudo or LMS (Least Mean Square) inversion method to adjust the network. Research[12] has used genetic algorithms to optimize the structure and weight of RBF networks. The proposed algorithm is used to approximate the curve and is experimented on the Boston dataset. Research [13] has developed a nonlinear offline model for Solid Oxide Fuel Cell (SOFC) using RBF neurons and genetic algorithms. Genetic algorithms are used to optimize the initial values of RBF neurons. This network is trained by the Gradient Descent algorithm.

In this paper, the authors combine RBF neurons and genetic algorithms in multidirectional mobile robot trajectory tracking control. This is a holonomic robot capable of simultaneous and independent movement during movement and rotation [14]. RBF neurons act as adaptive controllers in adaptive sliding control laws. Genetic algorithms are used to determine the number of hidden layer's neurons, centers, thresholds, and weights of the RBF neurons. After optimization, RBF neurons are trained online by Quasi-Newton.

This paper is organized into four sections. Introduction is Section 1; Section 2 describes an RBFand genetic algorithm; Section 3 describes the adaptive sliding mode controller using GA-RBF neural networkand Section 4 is the conclusions.

2. RBF NEURAL NETWORK AND GENETIC ALGORITHM

2.1. RBF neural network

2.1.1. Structure of the RBF neural network

RBF neurons consist of three layers: the input layer, the hidden layer and the output layer. The neurons in the hidden layer are activated by radial basis function. The hidden layer include an array of computed units, it is called hidden nodes. Each hidden node contains the center vector *c*that's the same size as the input vector *x*; the Euclidean distance between the center and the input signal of the network is defined as$\left\|x(t) - c_j(t)\right\|$ [15].

The structure of the RBF neural network is shown in Figure 1.

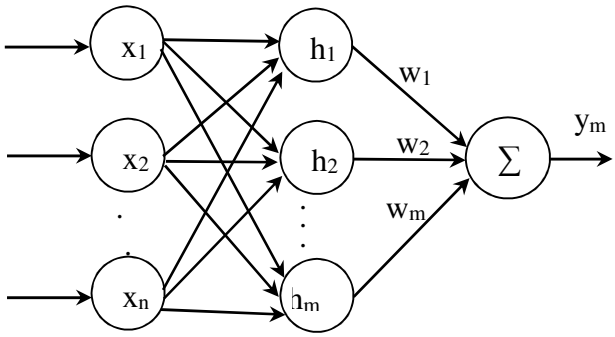


Figure 1 Structure of the RBF neural network

The output of the hidden layer is processed through the nonlinear activation function $h_j\,(t)$ as (1):

$$h_j(t) = exp\left(-\frac{\|x(t)-c_j(t)\|^2}{2b_j^2}\right), j = 1, 2, \dots, m \quad (1)$$

Whereb_j is the width of the j^{th} RBF and *m* is the number of hidden nodes.

The output signal of the network is a combination of linear weights as (2):

$$y_j(t) = \sum_{j=1}^{m} \omega_{ij} h_j(t)\,, i = 1, 2, \dots, n \quad (2)$$

Whereω is the weight of the output layer; *n* is the number of output nodes and *y* is the output signal.

2.1.2 Quasi-Newton algorithm with BFGS

BFGS (Broyden Fletcher Goldfarb Shanno) is one of the Quasi-Newton methods that useapproximations for the Hessian inverse matrix methods [16]. If it isindicated in the minimum problem as (3):

$$Min\ f(x), x \in R^n \quad (3)$$

Given $f(x^*)$as a minimum, the iterative process for searching x^*as (4):

$$X_{k+1} = X_k - \alpha_k \Xi_k \nabla f^T(x_k) \quad (4)$$

Where: $\Xi_k$is the symmetric matrix $n \times n$and $\alpha_k$is the learning speedthat are found tominimize$f(x_{k+1})$.$\Xi_k$is the Hessian matrix that the Newton method needs to find (then $\Xi_k = I$we have the Gradient Descent method).

Assume$f \in R^n$contain partial continuous second derivative, it is definedas (5):

$$g_k = \nabla f^T(x_k), p_k = X_{k+1} - X_k \quad (5)$$

And $H_k$is the Hessian approximation matrix in the $k$iteration. From each Positive constant symmetry matrixH_0, each point $x_0$and$k = 0$are implemented in three steps:

Step 1: Consider$d_k = -H_k^{-1} g_k$

Step 2: Minimal $f(x_k + \alpha d_k)$with$\alpha \geq 0$

We have$X_{k+1}, P_k = \alpha_k d_k, g_{k+1}$

Step 3: Consider $q_k = g_{k+1} - g_k$

And $H_{k+1} = H_k + \frac{p_k p_k^T}{p_k^T q_k} - \frac{H_k q_k q_k^T H_k}{q_k^T H_k q_k}$ (6)

Then update k and return to Step 1.

If $H_k$is the real value then $H_{k+1}$is the real value and the sequence of this algorithm converges. A new method for updating to H was proposed by Broyden Fletcher Goldfarb Shanno it is called BFGS as (7).

$$H_{k+1} = H_k + \left(1 + \frac{q_k^T H_k q_k}{q_k^T p_k}\right) \frac{p_k p_k^T}{p_k^T q_k} - \frac{p_k q_k^T H_k + H_k q_k p_k^T}{q_k^T p_k} \quad (7)$$

2.2. Genetic Algorithm

Genetic Algorithms (GAs) is a computer science technique for solving complex optimal problems. GAs is the evolutionary adaptation process of the biological populations that's on based Darwinian theory. It uses the principles of genetics, mutation, natural selection and hybridization. Chromosomes are made up of genes (represented by a linear array). Each gene carries several characteristics and is located in the chromosome. GAs begins with a population representing the potential solution arrays. However, the population consists of a certain number of coded individuals, which are characteristic chromosomes.

The main content in the construction of genetic algorithms is the method of coding feasible and design genetic operators. It is a repetitive procedure; in each iteration, it retains the outstanding solution and classifies them according to the quality of the solution and then selects some solutions according to some indexs and uses genetic operators to calculate, result it is a new generation of outstanding solutions. They will repeat this process until it meets some convergence index. Figure 2 illustrates the process of genetic algorithm.

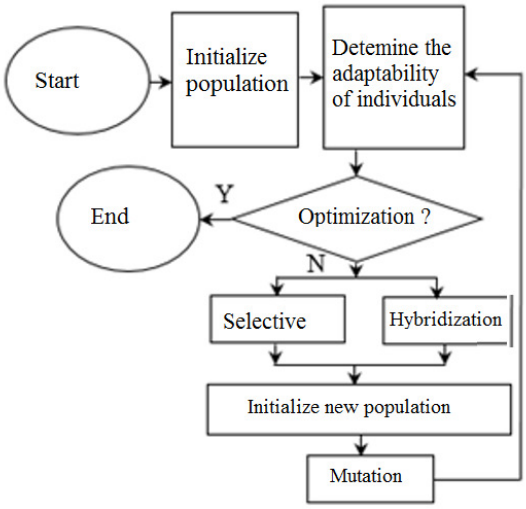


Figure 2 Diagram of genetic algorithm

2.3. Optimizing the structure and parameters of RBF neural networks using genetic algorithms

The main contents of using genetic algorithms to optimize the structure and parameters of RBF networks is to encode individuals, define objective functions and build RBF controllers with genetic algorithms. The use genetic algorithms to optimize RBF network, it can be considered as an adaptive system [2]. Control diagram is shown in Figure 3:

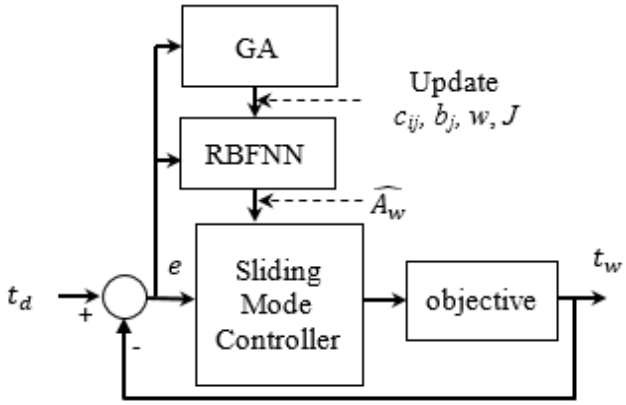


Figure 3 Diagram of GA-ASMC-RBF Control

Genetic algorithms adjust automatic the parameters of the network without humanintervention, as shown in Figure 4.

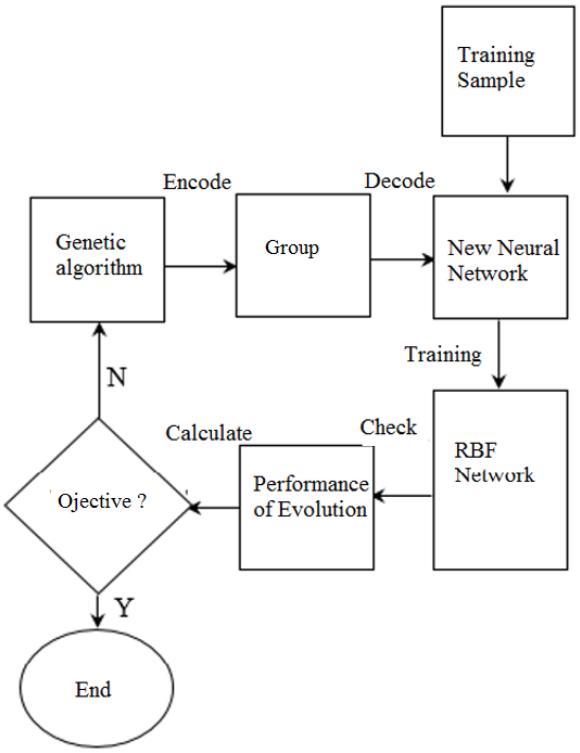


Figure 4 Diagram of structure and weights optimization of GA- RBF network

2.3.1. Chromosomal coding

Assumethat the maximum number of neurons on the hidden layer is s and the output neuron is m. Encode the hidden layers neurons in binary form such as (8) [2]:

$$j_1 j_2 \dots j_s \quad (8)$$

The number of hidden layers neurons is encoded by the binary method, represented as jvalues 0 or 1. When$j_i = 1$, it means that the neurons exist. When$j_i = 0$, the neurons do not exist and s represents the upper limit.

Connection weight is encoded as (9):

$$\omega_{11}\omega_{12} \dots \omega_{21}\omega_{22} \dots \omega_{sm} \quad (9)$$

The weights from the hidden layer to the output layer are encoded by the real-valued encoded method and ω_{ij}represent the connection weights from the input ith neurons to the jth hidden leyer neuron.

The center vector is encoded as (10):

$$c_{11}c_{12} \dots c_{21}c_{22} \dots c_{sm} \quad (10)$$

Thresholds and real-valued encoded program such as (11):

$$b_1 b_2 \dots b_m \quad (11)$$

Where, the thresholds of the output neurons are encoded by the real-valued method, $b_j$is the jth output neuron.

Thus, the coding arrays of a chromosome is the array of structural components, the connection weights, the center and the threshold, they are presented as (12):

$$j_1 j_2 \dots j_s \omega_{11}\omega_{12} \dots \omega_{21}\omega_{22} \dots \omega_{sm} b_1 b_2 \dots b_m c_{11}c_{12} \dots c_{21}c_{22} \dots c_{sm} \quad (12)$$

2.3.2. Ojective function

The ojective function is the basis to evaluategenetic algorithm, which directly enfluences the performance of the algorithm as well as the convergence speed of the network. Using the training error, actual value and output value of the network to determine the corresponding ojective function as (13):

$$E = \frac{1}{2}(t_d - t_\omega)^2 \quad (13)$$

Where: E is the training error, $t_d$is the desired input signal and $t_\omega$is the output of the object.

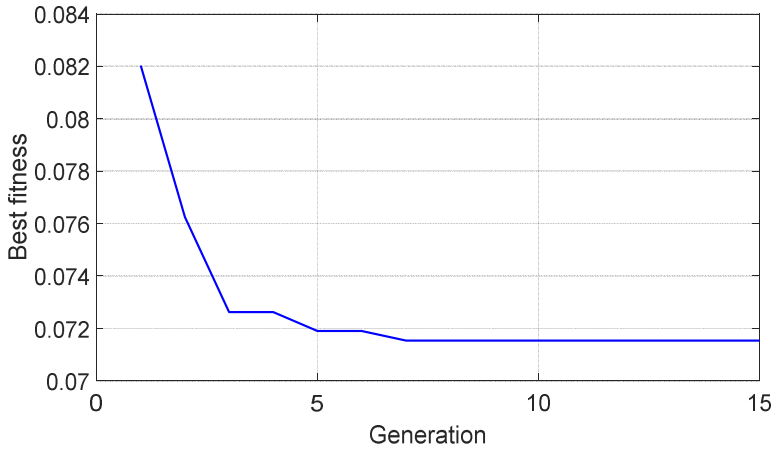
2.4. Optimal result of RBF neural network

Parameters for the complete GA algorithm are shown in Table 1. Selectivetypeis used the Roulette wheel method. The Roulette wheel method is the simplest selective type, each individual in the population occupies a slot on the roulette wheel that is proportional to the ojective function value.

Table 1 Parameters of the GA algorithm

Parameters	Optimalstructure the RBF network	Optimalparameter the RBF network
Generation number	15	15
Population size	20	30
Frequency of hybridization	0.7	0.7
Probability of mutation	0.01 – 0.001	0.01 – 0.001
Selective	Roulette	Roulette
Ojective function	$F = E + C\frac{s}{s_{max}}$	$\mathrm{F} = \mathrm{E} = \frac{1}{2}(\mathrm{t_d} - \mathrm{t_\omega})^2$

The results of the RBF neural network using GA are shown in Figures 5 and 6. Figure 5 shows the convergence across 15 generations, the most convergent point being 0.0715 (the smallest value of the objective function).



.

Figure 5 Diagram indicate the value of the objective function in structure optimal RBF network

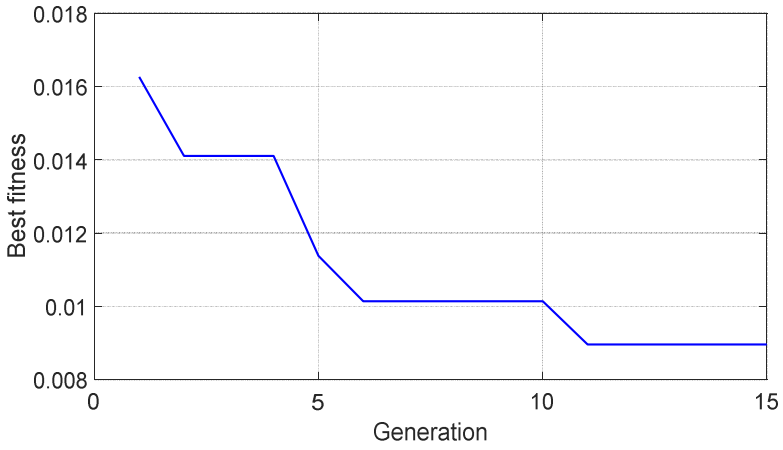


Figure 6 Diagram indicate the value of the ojective function in weights optimal RBF network

Figure 6 shows the convergence over 15 generations, the most convergent point being 0.0085 (the smallest value of the objective function). The average distance between individuals in terms of adaptability, across each generation is shorter.

The number of hidden layer neurons and parameters of RBF neurons after optimization by genetic algorithms are presented in Table 2.

Table 2 Structure and parameters of RBF network using the GA algorithm

Note	Value
b	10.6022
c_{ij}	$\begin{bmatrix} 12.41 & -0.47 & -1.73 & -12.08 & 9.89 & 1.76 & -11.01 \\ -4.39 & 1.19 & -13.67 & -2.88 & 9.83 & 10.35 & -13.91 \\ -1.65 & 12.3 & 7.07 & -9.59 & 3.01 & -3.92 & -12.82 \\ 6.25 & 5.15 & -2.06 & -2.69 & -10.11 & -2.2 & 3.98 \\ 13.58 & 4.00 & -0.88 & -3.9 & -1.98 & 4.42 & -1.46 \end{bmatrix}$
W_1	$[4.81 \quad 0.77 \quad -1.24 \quad 0.9 \quad -1.78 \quad 1.58 \quad -2.41]$
W_2	$[-3.92 \quad 0.87 \quad -1.86 \quad -1.48 \quad -2.24 \quad -2.00 \quad -2.45]$
W_3	$[3.57 \quad -2.82 \quad 2.38 \quad -1.16 \quad -4.5 \quad 4.18 \quad 1.77]$
J	7

From Table 2,it is observed that the genetic algorithm has determined the specific values of the hidden layer neuron, center, threshold, and initial weight of the RBF network.

3. ADAPTIVE SLIDING MODE CONTROLLER USING GA-RBF NEURAL NETWORK

3.1. Adaptive sliding mode control law using RBF neural network

The researchers propose a sliding-adaptive controller usingGA-RBF neural network to control multi-directional mobile robot trajectories. The mathematical model of the robot is presented as (14) [17].

$$\begin{bmatrix}\ddot{x}_\omega\\ \ddot{y}_\omega\\ \ddot{\phi}_\omega\end{bmatrix}=\begin{bmatrix}a_1 & -a_2\phi_d & 0\\ a_2\phi_d & a_1 & 0\\ 0 & 0 & a_3\end{bmatrix}\begin{bmatrix}\dot{x}_\omega\\ \dot{y}_\omega\\ \dot{\phi}_\omega\end{bmatrix}+\begin{bmatrix}b_1\gamma_1 & b_1\gamma_2 & 2b_1cos\phi\\ b_1\gamma_3 & b_1\gamma_4 & 2b_1sin\phi\\ b_2 & b_2 & b_2\end{bmatrix}\begin{bmatrix}u_1\\ u_2\\ u_3\end{bmatrix}+\begin{bmatrix}D_{fx}\\ D_{fy}\\ D_{f\phi}\end{bmatrix}$$

$$= A_\omega\beta + B_\omega U \tag{14}$$

Where, $x=[\dot{x}_\omega \quad \dot{y}_\omega \quad \dot{\phi}_\omega]^T$is the state variable vector of the robot; $U=[u_1 \quad u_2 \quad u_3]^T$is the control signal.

$$a_1=\frac{-3c}{3I_\omega+2Mr^2};\ a_2=\frac{3I_\omega}{3I_\omega+2Mr^2};\ a_3=\frac{-3cL^2}{3I_\omega L^2+I_v r^2}$$

$$b_1=\frac{kr}{3I_\omega+2Mr^2};\ b_2=\frac{krL}{3I_\omega+I_V r^2}$$

$$\gamma_1=-\sqrt{3}sin\phi-cos\phi;\ \gamma_2=\sqrt{3}sin\phi-cos\phi$$

$$\gamma_3=\sqrt{3}cos\phi-sin\phi;\ \gamma_4=-\sqrt{3}cos\phi-sin\phi$$

Sliding control law is defined as (15) [17]:

$$U=-B_\omega^{-1}\{A_\omega\dot{\beta}+(\ddot{\beta}_\omega+k\dot{e})+\eta sign(S)\} \tag{15}$$

Where,

$\eta=\begin{bmatrix}\eta_x & 0 & 0\\ 0 & \eta_y & 0\\ 0 & 0 & \eta_\phi\end{bmatrix}$is the positive determinant matrix,$k=diag[k_x \quad k_y \quad k_\phi]$is a positive constant

$S=\begin{bmatrix}\dot{e}_x+k_xe_x\\ \dot{e}_y+k_ye_y\\ \dot{e}_\phi+k_\phi e_\phi\end{bmatrix}=\begin{bmatrix}S_x\\ S_y\\ S_\phi\end{bmatrix}$is slip surface, $e=\beta-\beta_d=\begin{bmatrix}e_x\\ e_y\\ e_\phi\end{bmatrix}$is the error parameter.

The use the Quasi-Newton algorithm to approximate$A_\omega$for GA-RBF network is presented in Table 2. This matrix contains the robot's characteristic parameters as: radius of wheel (r); the inertia torque of the robot (I_v) and the inertiatorqueof of the ball (M).

The RBF network structure is used to approximate the f_i component in the $A_\omega$matrix asshown in Figure 7 [15]:

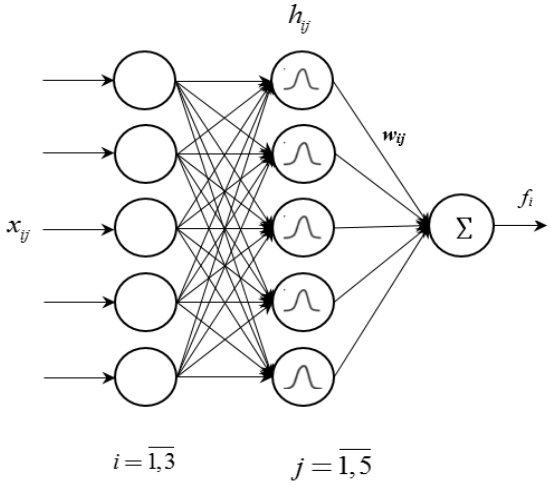


Figure 7 RBF neural network structure for approximate f_i

Where:

$$X_i = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} e(1) & \dot{e}(1) & \beta_d(1) & \dot{\beta}_d(1) & \ddot{\beta}_d(1) \\ e(2) & \dot{e}(2) & \beta_d(2) & \dot{\beta}_d(2) & \ddot{\beta}_d(2) \\ e(3) & \dot{e}(3) & \beta_d(3) & \dot{\beta}_d(3) & \ddot{\beta}_d(3) \end{bmatrix} \tag{16}$$

$$h_{ij}\big|_{i=1,2,3} = [h_{i1} \quad h_{i2} \quad h_{i3} \quad h_{i4} \quad h_{i5}] \tag{17}$$

$$\boldsymbol{\omega_{ij}}\big|_{i=1,2,3} = [\omega_{i1} \quad \omega_{i2} \quad \omega_{i3} \quad \omega_{i4} \quad \omega_{i5}] \tag{18}$$

Outputsfiias (14):$f_i = \boldsymbol{\omega}_{ij}^T h_{ij}$ (19)

The result approximates the matrix $A_\omega$as (20):

$$\widehat{A_\omega} = \begin{bmatrix} \boldsymbol{\omega}_{1j}^T h_{1j} & -\boldsymbol{\omega}_{2j}^T \phi_d h_{2j} & 0 \\ \boldsymbol{\omega}_{2j}^T h_{2j} & \boldsymbol{\omega}_{1j}^T h_{1j} & 0 \\ 0 & 0 & \boldsymbol{\omega}_{3j}^T h_{3j} \end{bmatrix} \tag{20}$$

Adaptive sliding mode controllers to RBF neurons are designed as (21):

$$U = -B_\omega^{-1}\{\widehat{A_\omega}\dot{\beta} + (\ddot{\beta}_\omega + k\dot{e}) + \eta sign(S)\} \tag{21}$$

The RBF networks are considered an adaptive controller. When the actual trajectory of the robot deviates from the reference trajectory due to the effects of actual conditions such as road surface friction, torque changes, etc. RBF networks will be automatically updated using the Quasi-Newton algorithm, resulting in the same variation so that the error is minimal. It is found that the control law (21) is adapted to the operating conditions of the robot.

To prove the stability of the control law we need to prove that the slidingmode surface converges to 0 according to Lyapunov. The Lyapunov function is defined as follows:

$$V = \frac{1}{2}S^T S > 0 \tag{22}$$

$$\dot{V} = S^T\left(-\widehat{A_\omega}\dot{\beta} - \eta sign(S)\right) = S^T\left(-\tilde{\Gamma}\dot{\beta} - \eta sign(S)\right)$$

$$= -S^T\left(\tilde{\Gamma}\dot{\beta} + \eta sign(S)\right) \leq 0 \tag{23}$$

Where, the matrix $\eta$is the positive symmetric matrix. In addition, the error will converge to zero, so that$S(t) \longrightarrow 0$when$t \longrightarrow \infty$. And $e(t), \dot{e}(t) \longrightarrow 0$when$t \longrightarrow \infty$.

3.2. Parameters and simulation results

3.2.1. Simulation parameters:

The robot parameters are shown in Table 3:

Table 3 Parameters of multi-directional mobile robot [18]

Notation	Meaning	Value	Unit
I_v	The inertia torque of the robot	11.25	kgm^2
M	The inertia torque of ball	9.4	kgm^2
L	The distance from wheel to the center of the robot	0.178	m
k	Transmission coefficient	0.448	
c	Lubrication coefficient	0.1889	kgm^2/s
I_ω	The inertia torque of the wheel	0.02108	kgm^2
r	Radius of wheel	0.0245	m

* The parameters of the adaptive sliding mode controller to the RBF neural network is initialized within the value range ofX_i,asshown in Table 4.

Table 4 Parameters of the sliding-adaptive controller

Meaning	Notation	Value
Sliding parameters	$\begin{bmatrix}\eta_x & 0 & 0\\ 0 & \eta_y & 0\\ 0 & 0 & \eta_\phi\end{bmatrix}$	$\begin{bmatrix}20 & 0 & 0\\ 0 & 20 & 0\\ 0 & 0 & 20\end{bmatrix}$
	$\begin{bmatrix}k_x & 0 & 0\\ 0 & k_y & 0\\ 0 & 0 & k_\phi\end{bmatrix}$	$\begin{bmatrix}25 & 0 & 0\\ 0 & 25 & 0\\ 0 & 0 & 58\end{bmatrix}$

The diagram in MATLAB / SIMULINK controls the multi-directional mobile robot as shown in Figure 8.

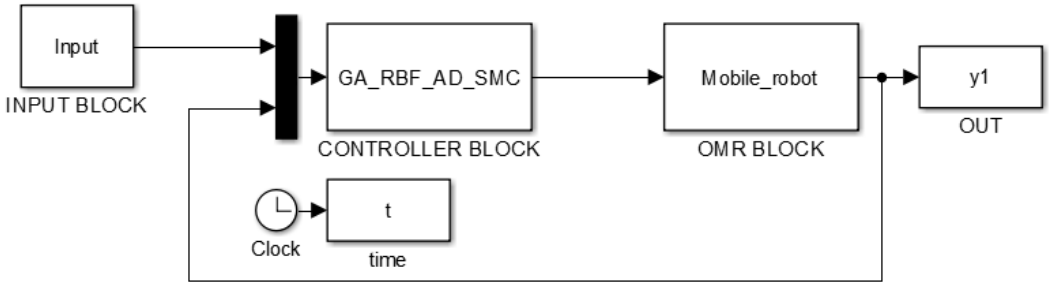


Figure 8 Simulating diagram of adaptive sliding mode controller to GA- RBF neural network

3.2.2. Simulation results

a) The input referency trajectory of the sliding-adaptive controller using RBF neurons as (24) (circular path):

$$\begin{cases} x = 5\cos(2\pi t)\ (m) \\ y = 5\sin(2\pi t)\ (m) \\ \phi = -0.2\pi t\ (rad/s) \end{cases} \quad (24)$$

Simulationresult of the adaptive sliding mode controller using RBF network in two cases:the network structure and parameters are randomandthe network structure and parameters aredetermined by genetic algorithm as Figure 9:

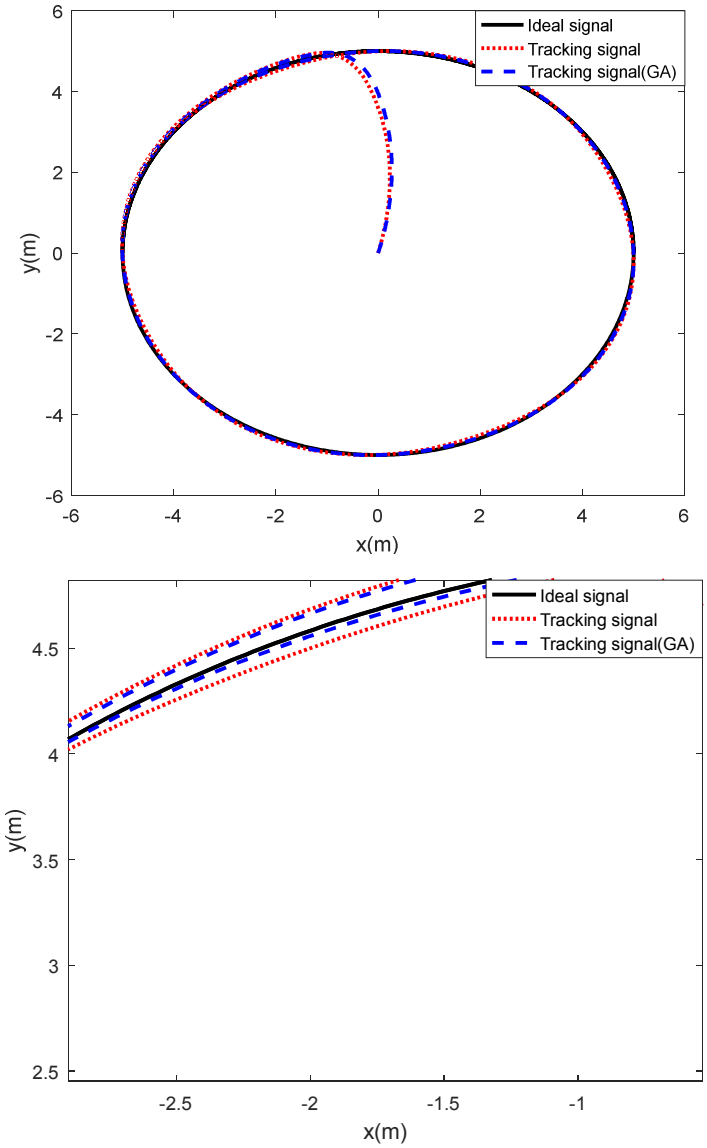


Figure 9 Respond to the circular path

The purpose of the control is to maintain stability and the ability to follow a referency trajectory on the x, yaxis. Simulation result in Figure 9 indicate that the tracking trajectory of the robot moving in a circle path, with a control cycle of 1.5 (s). The results show that the convergence capability and control quality of the adaptivesliding modecontroller to GA-RBF neural network has a better. In Table 5, it is found that the static error of RBF network is 0.002 (m) and the GA-RBF network is 0.001 (m). However, the static time of the two methods is nearly the same at 0.3 (s).

b) The input referency trajectory of the adaptive sliding mode controller using RBF neuronsas (25) (Rhodonea path):

$$\begin{cases} x = 5\cos(2\pi t)\cos(\pi t) - 5\,(m) \\ y = 5\sin(2\pi t)\sin(\pi t)\,(m) \\ \phi = -0.2\pi t \qquad\qquad (rad/s) \end{cases} \tag{25}$$

Simulation result of the adaptive sliding mode controller using RBF network in two cases: the network structure and parameters are random and the network structure and parameters are determined by genetic algorithm:

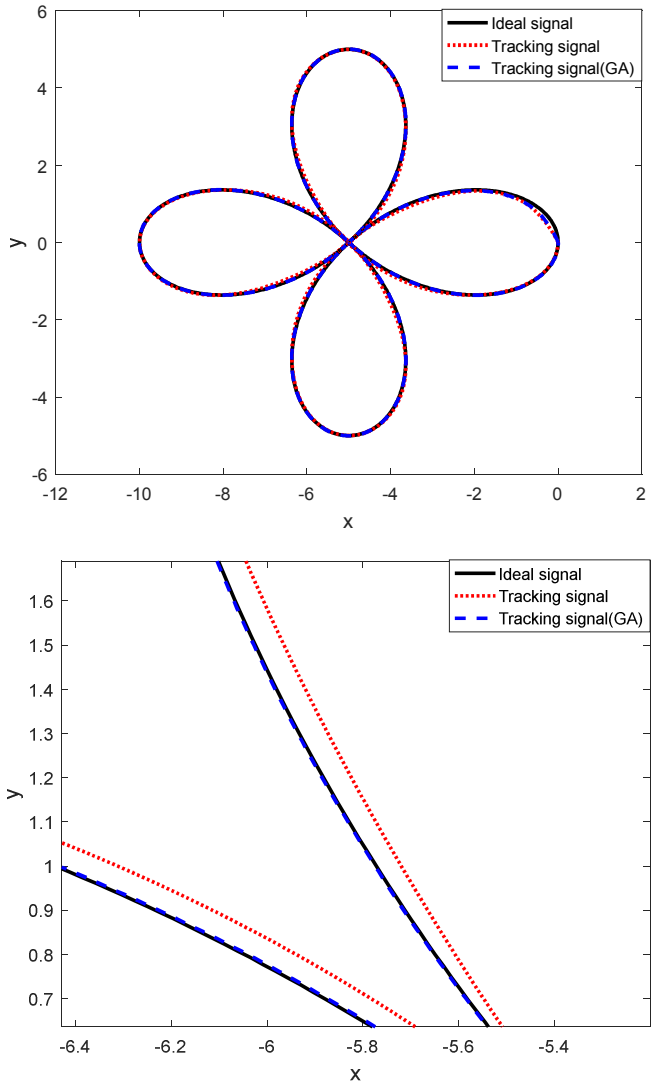


Figure 10 Respond to to the rhodoneapath

In the Rhodonea path, the quality of the tracking trajectory of the the adaptive sliding mode controller to GA-RBF neural network (static error is 0.001 (m))is better than the traditional RBF neural network (static error is0.002 (m)). In Table 5, it is found that the static error of RBF network is 0.002 (m) and the GA-RBF network is 0.001 (m). However, the static time of the two methods is nearly the same at 0.15 (s).

However, in the two case,the static time of the control of the Rhodonea pathis 0.15(s) faster than the circular path,because the starting point of the robot is from the Rhodonea path, while the circular path is from the center of the trajectory.

Table 5 Simulation results of adaptive sliding mode control to RBF neural network

Parameters	Circular path		The Rhodonea curve	
	RBF	GA-RBF	RBF	GA-RBF
Stabilitytime (s)	0.3	0.3	0.15	0.15
Stability error (m)	0.002± 0.0005	0.001± 0.0005	0.002± 0.0005	0.001± 0.0005
Overshoot (%)	0.3± 0.05	0.15± 0.05	0.3± 0.05	0.15± 0.05

From Table 5, it is found that the quality criteria of the static error and overshoot of the adaptive sliding mode controller using the GA-RBF neural network is smaller traditionalRBF

neural network. This demonstrates the effectiveness of genetic algorithms and Quasi-Newton's algorithm in online training to control multi-directional mobile robots and nonlinear systems.

4. CONCLUSION

In this paper, genetic algorithms were used to optimize the structure and parameters of RBF neurons. This network is used for adaptive sliding mode controller to control the multi-directional mobile robots. The weights of the network are trained online with the Quasi-Newton algorithm. Simulation results with MATLAB/SIMULINK show that the efficiency of the adaptive sliding mode controller to the GA-RBF neural network. Quality criterias of GA-RBF controllers are better than the traditional RBF controllers. The proposed algorithm is also suitable to control the nonlinear system.

REFERENCES

[1] Yang Jinyun, Jiang Daixin.Applying the Ant Colony RBF Neural Network to Forecast Competition Achievement.*Journal of Residuals Science & Technology,* **13**(6), 2016, pp. 226.1-226.5.

[2] Weikuan Jia, Dean Zhao, Tian Shen, Chunyang Su, Chanli Hu, and Yuyan Zhao. A New Optimized GA-RBF Neural Network Algorithm.*Computational Intelligence and Neuroscience,*2014, pp. 1-7.

[3] Ms. Tanvi Gupta, Supriya P Panda, S.S. Handa and M.V.Ramana Murthy. A Review on Optimizing Radial Basis Function Neural Network using NatureInspired Algorithm.*International Journal on Future Revolution in Computer Science & Communication Engineering*,**3**(10), 2017,pp. 49-57.

[4] Xu-Sheng Gan, Can Yang, Hai-Long Gao.Optimization Design of Structure and Parameters for RBF Neural Network Using Hybrid Hierarchy Genetic Algorithm.*Advanced Materials Research*, **951**, 2014, pp. 274-277.

[5] Min Gan, Hui Peng, Xue-ping Dong. A hybrid algorithm to optimize RBF network architecture and parameters for nonlinear time series prediction.*Applied Mathematical Modelling,***36**, ScienceDirect, 2012, pp. 2911-2919.

[6] Kinjal Jadav, Mahesh Panchal.Optimizing Weights of Artificial Neural Networks using Genetic Algorithms.*International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE)*,**1**(10), 2012,pp. 47-51.

[7] Janati Idrissi, Ramchoun Hassan, Ghanou Youssef, Ettaouil Mohamed.Genetic Algorithm for Neural Network Architecture Optimization.*International Conference on Logistics Operations Management* (GOL), IEEE, 2016, pp. 1-4.

[8] Nursel Öztürk. Useof genetic algorithm to design optimal neural network structure.*Engineering Computations*, **20**(8), 2003, pp. 979-997.

[9] Ms.Dharmistha D.Vishwakarma.Genetic Algorithm based Weights Optimization of Artificial Neural Network.*International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*,**1**(3), 2012,pp. 206-211.

[10] Min Gan, Hui Peng, Xue-ping Dong. A hybrid algorithm to optimize RBF network architecture and parameters for nonlinear time series prediction. *Applied Mathematical Modelling,***36**, ScienceDirect, 2012, pp. 2911-2919

[11] Shifei Ding, Li Xu, Chunyang Su, Fengxiang Jin. An optimizing method of RBF neural network based on genetic algorithm. *Neural Comput & Applic,***21**, 2012, pp. 333-336.

[12] Virginie Lefort, Carole Knibbe, Guillaume Beslon, and Jo¨el Favrel. Simultaneous Optimization of Weights and Structure of an RBF Neural Network.*International Conference on Artificial Evolution*, Springer, 2005, pp. 49-60.

[13] Xiao-Juan Wu, Xin-Jian Zhu, Guang-Yi Cao, Heng-Yong Tu. Modeling a SOFC stack based on GA-RBF neural networks identification.*Journal of Power Sources,***167**, 2007, pp. 145-150.

[14] S Alima Djebrani, Abderraouf Benali, Foudil Abdessemed. Modelling and control of an omnidirectional mobile manipulator.*Int. J. Appl. Math. Comput. Sci*., **22**(3), 2012, pp. 601-616.

[15] Jinkun Liu,Radial Basis Function (RBF) Neural Network Control for Mechanical Systems, Springer, 2013, pp. 374.

[16] M. Sadeghi, M. Pashaie and A. Jafarian. RBF Neural Networks Based on BFGS Optimization Method for Solving Integral Equations**.***Advances in Applied Mathematical Biosciences,* **7**(1), 2016, pp. 1-22.

[17] Ya-Chao Yang and Chi-Cheng Cheng.Robust adaptive trajectory control for an Omnidirectional vehicle with parametric uncertainty**.***Transactions of the Canadian Society for Mechanical Engineering,* **37**(3), 2013, 405- 413.

[18] Keigo Watanabe.Control of an omnidirectional Mobile Robot.*Second International Conference on Knowledge-Based Intelligent Electronic Systems*, 1998, pp. 51-60.