Journal of ELECTRICAL ENGINEERING, VOL. 64, NO. 1, 2013, 3–11

VARIABLE STRUCTURE NEURAL NETWORKS FOR ADAPTIVE ROBUST CONTROL USING EVOLUTIONARY ARTIFICIAL POTENTIAL FIELDS

Hassen Mekki [*] [**] — **Mohamed Chtourou** [*]

A novel neural network architecture, is proposed and shown to be useful in approximating the unknown nonlinearities of dynamical systems. In the variable structure neural network, the number of basis functions can be either increased or decreased with time according to specified design strategies so that the network will not overfit or underfit the data set. Based on the Gaussian radial basis function (GRBF) variable neural network, an adaptive state feedback controller is presented. The location of the centers of the GRBFs is analyzed using a new method inspired from evolutionary artificial potential fields method combined with a pruning algorithm. Using this method we can guarantee a minimal number of neuron. It is in noted, that both the recruitment and the pruning is made by a single neuron. Consequently, the recruitment phase does not perturb the network and the pruning does not provoke an oscillation of the output response. The weights of neural network are adapted using a Lyapunov approach. Moreover, the stability of the system can be analyzed and guaranteed by introducing the supervisory controller and modified adaptation law with projection.

K e y w o r d s: variable structure neural network; radial basis functions, evolutionary artificial fields, robust adaptive control

1 INTRODUCTION

Neural network research has gained increasing attention in recent years. In fact, artificial neural networks are capable of learning and reconstructing complex nonlinear mappings and they have been widely studied by control researchers in identification analysis and the design of control systems [10, 13–15, 17]. The network size, often measured by number of hidden units in a single hidden layer network, reflects the capacity of the neural network to approximate an arbitrary function. A fundamental question is what size of the neural network is required to solve a specific problem. If the training starts with a small network, it is possible that the learning process cannot be achieved. On the other hand, if a large network is used, the learning process can be very slow and/or overfitting may occur. The approaches, which assume a priori the number of RBFs, usually lead to the problem of poor generalization. In addition, these approaches usually work offline, so they are not suitable for practical real-time applications where the online learning is required for the neural-network-based controller design. To remedy the aforementioned shortcomings, several growing RBF networks have been proposed in [8, 11, 12, 18, 21].

In using RBF networks, the basis function are placed on regular points of a square mesh, for example, covering a relevant region of space where the state is known to be contained [3, 4]. This region therefore is the network approximation region, which is in general known for a given system. The distance between the points affects the number of basis functions required to cover the region and hence determines the size of the neural network.

It seems well that if the size of the neural network input vector increases, it will have an excessive increases of the neural network size which will provoke oscillation in the output responses.

To remedy theses problems, a novel neural network architecture is proposed, where the location of the centers of the GRBFs is analyzed using a new method inspired from evolutionary artificial potential field's method.

Output tracking and stabilization of nonlinear systems has received considerable attention during last decades [4, 5, 24]. Feedback linearizing method has been widely used in this area. Although, linearizing feedback controllers are efficient in theory, they can not always be implemented in practice since they need complete knowledge about the dynamical model as well, as all its parameters. As solution, adaptive neural network controller is suggested.

In this paper, a method which concerns the fully-state linearizable or minimum phase nonlinear systems is proposed. The idea consists in designing a feedback controller constructed by a neural network. The weights of the neural network are updated such that the proposed control can track a predetermined input-output linearizing controller. Eventually, using the Lyapunov approach, it can be proved that the desired trajectory is asymptotically tracked by the output signal. As another contribution, we will show that under some structural assumptions the system drift is not necessary to construct the controller.

This paper is organized as follows. Section 2 provides a brief preliminary on fully-state linearizing control for trajectory tracking is given. In Section 3, the proposed adaptation law and the implementation of the controller

[*] Intelligent Control, design and Optimization of complex system (ICOS), University of Sfax, Tunisia, [**] National School of Engineering of Sousse, University of Sousse, Tunisia, hassen.mekki@eniso.rnu.tn, mohamed.chtourou@enis.rnu.tn

DOI: 10.2478/jee-2013-0001, ISSN 1335-3632

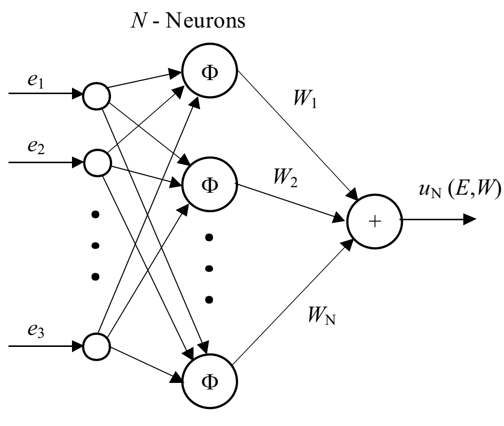

Fig. 1. Structure of the neural network

are then presented. This is done by the use of multilayered neural networks for the identification of uncertain nonlinear functions. In Section 4, we describe a novel self-organizing RBF network that can dynamically vary its structure in real time. The proposed self-organizing RBF network is capable of adding or removing RBFs to ensure the desired approximation accuracy and at the same time to keep the appropriate network complexity. The location of the centers of the GRBFs is analyzed using a new method inspired from evolutionary artificial potential fields method. To show the obtained performances of the proposed algorithm, Section 5 presents two simulation examples. The first one deals with a based variable structure network on-line identification of a nonlinear function. The second example treats the robust adaptive control of nonlinear dynamical system.

2 LINEARIZING CONTROL

In this study, we consider an nth-order nonlinear system, with the input $u \in \Re$ and the output $y \in \Re$ described as

$$\begin{aligned} x^{(n)} &= f\big(x, \dot{x}, \dots, x^{(n-1)}\big) + g\big(x, \dot{x}, \dots x^{(n-1)}\big) u\,, \\ y &= x \end{aligned} \tag{1}$$

where $f,\, g\colon \Re^{(n)} \to \Re$ are unknown nonlinear function. It will be assumed that there exist two constants $f_{\max} > 0$ and $g_{\min} > 0$ such that $|f| \leqslant f_{\max}$ and $|g| \geqslant g_{\min}$.

Let $X = [x, \dot{x}, \dots, x^{(n-1)}]^\top = [x_1, x_2, \dots, x_n]^\top \in \Re^n$ be the state vector of the system. As is well known, if $f(X)$ and $g(X)$ of the system (1) are known, then the feedback linearization technique can be employed to design a desired controller. Let $e = y_d - y$ be the error between the desired and the actual outputs. Define $Y_d = [y_d, \dot{y}_d, \dots, y_d^{(n-1)}]^\top$ and assume that $y_d, \dot{y}_d, \dots, y_d^{(n-1)}$ are all bounded. Then the error vector of the system becomes $E = Y_d - X = [e, \dot{e}, \dots, e^{(n-1)}]^\top = [e_1, e_2, \dots, e_n]^\top$. Suppose we choose a gain vector $K = [k_0, \dots, k_{n-1}]^\top$ such that all roots of $s^n + k_{n-1}s^{n-1} + \dots + k_1 s + k_0 = 0$ are in the left-half complex plane. Let the feedback control law given by

$$u^*(t) = \frac{1}{g(X)}[-f(X) + y_d^{(n)} + K^\top E]\,. \tag{2}$$

Substituting (2) into (1), we have

$$e^{(n)} + k_{n-1}e^{(n-1)} + \dots + k_1\dot{e} + k_0 e = 0\,. \tag{3}$$

Consequently, from (3), we have $e(t) \to 0$ as $t \to \infty$, so, $y \to y_d$ asymptotically. However, it is noted that $f(X)$ and $g(X)$ of the system (1) are assumed to be unknown in this study.

3 PROPOSED ADAPTIVE CONTROLLER

One of the main drawbacks of the linearizing state feedback law is the difficulty to construct the nonlinear part, in addition to a necessary exact knowledge of the system model ($f(X)$ and $g(X)$).

To solve this problem, we suggest using an adaptive neural network controller and a control law given by

$$u = u_N(E, W) + u_s\,, \tag{4}$$

where u_N is the output of a direct adaptive neural controller (5) and u_s is a supervisory control action which is achieved only when the error of the system exceeds some bound.

$$u_N = \sum_{i=1}^{N} w_i \varphi_i(e) \tag{5}$$

(5) where the activation function $\varphi(\cdot)\colon \Re \to \Re$ is a Gaussian function and N. is the number of neurons in the hidden layer and w_i are the weights between the hidden and the output layer (see Fig. 1)

Throughout the paper, the following assumption is made.

Assumption 1. Let define the constraint sets Ω_x and Ω_W for the state X and the adjustable parameter vector W as

$$\Omega_x = \big\{X \in \Re^n \colon \|X\| \le M_x\big\}\,,$$
$$\Omega_W = \big\{W \in \Re^N \colon \|W\| \le M_W\big\}\,,$$

where M_x and M_W are pre-specified parameters. Intuitively, Ω_x is the feasible set of the state X.

From equations (1) and (4), we have

$$\begin{aligned} \dot{x}_n &= f(X) + g(X)[u_N(E,W) + u_s] = \\ &f(X) + g(X)[u_N(E,W) + u_s] + g(X)u^* - g(X)u^* = \\ &f(X) + g(X)[u_N(E,W) + u_s] - f(X) + y_d^{(n)} + K^\top E - g(X)u^* \\ &= y_d^{(n)} + K^\top E - g(X)[u^* - u_N(E,W) - u_s]\,. \end{aligned} \tag{6}$$

This implies that

$$e^{(n)} = -K^\top E + g(X)[u^* - u_N(E,W) - u_s]\,. \qquad (7)$$

Let $B_c = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ g(X) \end{bmatrix}$, $A_c = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ -k_0 & -k_2 & -k_3 & \dots & -k_{n-1} \end{bmatrix}$.

We have

$$\dot E = A_c E + B_c\big[u^* - u_N(E,W) - u_s\big]. \qquad (8)$$

THEOREM. *Consider the class of nonlinear dynamical systems described by* (1) *with assumption 1. Assume that the state vector X is measurable and y_d is a smooth reference trajectory to be tracked. If the control input u is designed such that in* (4) *with* (5)*, and u_s is given by*

$$u_s = I^* \operatorname{sig}(E^\top P B_c)\big[|u_N(E,W)| + g_{\min}^{-1}(f_{\max} + |y_d^{(n)}| + |K^\top E|)\big].$$

The weight vector are adjusted using the adaptive mechanism given by

$$\dot W = \gamma E^\top P_n g_{\min} \frac{\partial u_N(E,W)}{\partial W} \text{ if } \|W\| < M_W \text{ or}$$
$$\|W\| = M_W \text{ and } E^\top P_n g_{\min} W^\top \frac{\partial u_N(E,W)}{\partial W} \ge 0\,, \qquad (9)$$

$$\dot W = \Pr\Big\{\gamma E^\top P_n g_{\min} \frac{\partial u_N(E,W)}{\partial W}\Big\} \quad \text{otherwise.} \qquad (10)$$

where the projection operator $\Pr\{\cdot\}$ *is defined as* [24]

$$\Pr\Big\{\gamma E^\top P_n g_{\min} \frac{\partial u_N(E,W)}{\partial W}\Big\} = \gamma E^\top P_n g_{\min} \frac{\partial u_N(E,W)}{\partial W} - \gamma E^\top P_n g_{\min} \frac{W}{\|W\|^2} W^\top \frac{\partial u_N(E,W)}{\partial W}$$

and P_n the n^{th} column of P (P is positive definite symmetric matrix described afterward). Then the output tracking error asymptotically converges to zero.

P r o o f . Consider the Lyapunov function candidate

$$V_e = \frac12 E^\top P E\,, \qquad (11)$$

where P is a positive-definite symmetric matrix satisfying the Lyapunov equation

$$A_c^\top P + P A_c^\top = -Q \qquad (12)$$

and Q is a given positive-definite symmetric matrix. In the following, we will choose Q such that $\lambda_{\min}(Q) > 1$, where $\lambda_{\min}(Q)$ denotes the minimum eigenvalue of Q.

Define

$$V_M = \frac12 \lambda_{\min}(P)(M_x - \|Y_d\|_\infty)^2. \qquad (13)$$

Note that if $\|X\| \ge M_x$, then, from (11), we have

$$V_e \ge \frac12 \lambda_{\min}(P)\|E\|^2 \ge \frac12 \lambda_{\min}(P)(\|X\| - \|Y_d\|)^2 \ge \frac12 \lambda_{\min}(P)(M_x - \|Y_d\|_\infty)^2 = V_M\,.$$

Hence if $V_e < V_M$, then $\|X\| < M_x$. The time derivative of V_e along the trajectories of the closed-loop system (8) satisfies

$$\dot V_e = \frac12 E^\top (A_c^\top P + P A_c) E + E^\top P B_c [u^* - u_N(E,W) - u_s]$$
$$= -\frac12 E^\top Q E + E^\top P B_c [u^* - u_N(E,W) - u_s]$$
$$\le -\frac12 E^\top Q E + |E^\top P B_c|(|u^*| + |u_N(E,W)| - E^\top P B_c u_s\,.$$

Let P_n be the n^{th} column of P. We have

$$E^\top P B_c = E^\top P_n g\,. \qquad (14)$$

From (2) and the hypothesis $|f| \le f_{\max}$ and $|g| \ge g_{\min}$ we have

$$u^* \le g_{\min}^{-1}(f_{\max} + |y_d^{(n)}| + |K^\top E|)\,.$$

Define the indicator function I^* by $I^* = 1$ if $V_e \ge V_M$ and $I^* = 0$ if $V_e < V_M$. Hence, if the supervisory controller is chosen as

$$u_s = I^* \operatorname{sig}(E^\top P B_c)\big[|u_N(E,W)| + g_{\min}^{-1}(f_{\max} + |y_d^{(n)}| + |K^\top E|)\big] \qquad (15)$$

where sig represent the sign function. Then, from (13) and (15), we can guarantee that $\dot V_e < 0$ if $V_e \ge V_M$.

On the other hand, in order to derive a proper adaptation law for the parameter vector $W \in \mathbb{R}^N$, let W^* be the optimal parameter vector such that the approximation error

$$\delta = u_N(E, W^*) - u^* \qquad (16)$$

is minimized. Notice that from (2) the u^* is a function of time, hence so is W^*.

For simplicity of analysis, we may choose Ω_W large enough such that $W^*(t) \in \Omega_W$ for all t. By incorporating (16), (8), we can write

$$\dot E = A_c E + B_c\big[u_N(E,W^*) - u_N(E,W) - u_s\big] - B_c \delta\,. \qquad (17)$$

Let consider another Lyapunov function candidate, containing the error of the system and the error between the optimal parameter W^* and the actual parameter W

$$V = \frac12 E^\top P E + (2\gamma)^{-1}(W^* - W)^\top (W^* - W) \qquad (18)$$

where γ is a positive constant determining the convergence speed. Using (17), we have

$$\dot{V} = \frac{1}{2}E^\top(A_c^\top P + PA_c)E + E^\top PB_c\big[u_N(E,W^*) - u_N(E,W) - u_s - \delta\big] + \gamma^{-1}(W^* - W)(\dot{W}^* - \dot{W}) = -\frac{1}{2}E^\top QE + E^\top PB_c\big[u_N(E,W^*) - u_N(E,W) - u_s - \delta\big] + \gamma^{-1}(W^* - W)(\dot{W}^* - \dot{W})\,. \quad (19)$$

The Taylor expansion of $u_N(E,W)$ around W^*:

$$u_N(E,W^*) - u_N(E,W) = (W^* - W)^\top \frac{\partial u_N(E,W)}{\partial W} + O(W^* - W)^2\,, \quad (20)$$

where $O(W^* - W)^2$ is a high-order term.

The equation (19) can be written as follows

$$\dot{V} = -\frac{1}{2}E^\top QE + E^\top PB_c\Big[(W^* - W)^\top \frac{\partial u_N(E,W)}{\partial W} + O(W^* - W)^2 - u_s - \delta\Big] + \gamma^{-1}(W^* - W)^\top(\dot{W}^* - \dot{W})$$

$$\dot{V} = -\frac{1}{2}E^\top QE - \gamma^{-1}(W^* - W)^\top\Big[\dot{W} - \gamma E^\top PB_c\frac{\partial u_N(E,W)}{\partial W}\Big] - E^\top PB_c\big[\delta - O(W^* - W)^2\big] - E^\top PB_c u_s + \gamma^{-1}(W^* - W)^\top\dot{W}^*\,. \quad (21)$$

Or $E^\top PB_c u_s \geq 0$ and from (14) we obtain

$$\dot{V} \leq -\frac{1}{2}E^\top QE - \gamma^{-1}(W^* - W)^\top\Big[\dot{W} - \gamma E^\top P_n g_{\min}\frac{\partial u_N(E,W)}{\partial W}\Big] - E^\top P_n g_{\min}\big[\delta - O(W^* - W)^2\big] + \gamma^{-1}(W^* - W)^\top\dot{W}^*\,.$$

In order to get a proper adaptation law and simultaneously guarantee $W \in \Omega_W$, a modified adaptation law with projection is proposed as

$$\dot{W} = \gamma E^\top P_n g_{\min}\frac{\partial u_N(E,W)}{\partial W} \text{ if } \|W\| < M_W \text{ or } \|W\| = M_W \text{ and } E^\top P_n g_{\min} W^\top \frac{\partial u_N(E,W)}{\partial W} \geq 0\,, \quad (22)$$

$$\dot{W} = \Pr\Big\{\gamma E^\top P_n g_{\min}\frac{\partial u_N(E,W)}{\partial W}\Big\} \text{ otherwise.} \quad (23)$$

where the projection operator $\Pr\{\cdot\}$ is defined as in [24] by

$$\Pr\Big\{\gamma E^\top P_n g_{\min}\frac{\partial u_N(E,W)}{\partial W}\Big\} = \gamma E^\top P_n g_{\min}\frac{\partial u_N(E,W)}{\partial W} - \gamma E^\top P_n g_{\min}\frac{W}{\|W\|^2}W^\top\frac{\partial u_N(E,W)}{\partial W}\,.$$

4 VARIABLE STRUCTURE NEURAL NETWORK

There are five parameters characterizing the RBF network approximation to be determined:

- The number of RBFs N,
- The type of the RBF,
- The location of the center $C_{(j)}$,
- The radius in each coordinate $\sigma_{i(j)}$,
- The weight vector for each output neuron ω_k.

In this section, we present a novel RBF network structure that is capable of determining the number of RBFs N, the location of the center $C_{(j)}$ and the weight vector for each output neuron ω_k by itself. The determination of ω_k is already seen in the section 2. We first show how to determine the location of the center $C_{(j)}$. The strategy of determination of the number of RBFs needed in the proposed online identification problem will be discussed in section 3. The proposed self-organizing RBF network is capable of adding or removing RBFs to ensure the desired approximation accuracy and at the same time to keep the appropriate network complexity.

4.1 Determination of the RBFs location center

In using RBF networks, the basis function are placed on regular points of a square mesh, for example, covering a relevant region of space where the state is known to be contained [3, 4]. This region therefore is the network approximation region, which is in general known for a given system. The distance between the points affects the number of basis functions required to cover the region and hence determines the size of the neural network.

It seems well that if the size of the neural network input vector increases, it will have an excessive increases of the neural network size which will provokes oscillation in output responses.

To remedy theses problems, a novel neural network architecture, is proposed, where the location of the centers of the GRBFs, is analyzed using a new method inspired from evolutionary artificial potential field method.

4.2 The Artificial Potential Field

This method is especially used in the real-time robot path planning. In the artificial potential field methods, a robot is considered as a particle under the influence of an artificial potential field U whose local variations reflect e.g. the positions of obstacles and of the goal that the robot is supposed to reach [16, 20, 23]. The potential field function is defined as the sum of an attraction field that pulls the robot towards the goal and a repulsive field that repels it forms the obstacles. The movement is executed in an iterative way, in which an artificial force is induced by

$$\vec{F}(q) = -\vec{\nabla}U(q)\,. \quad (24)$$

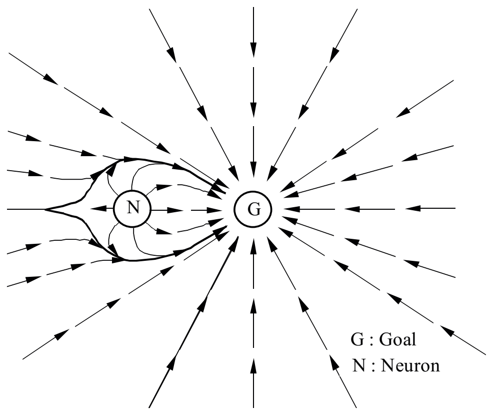

Fig. 2. The Artificial potential field

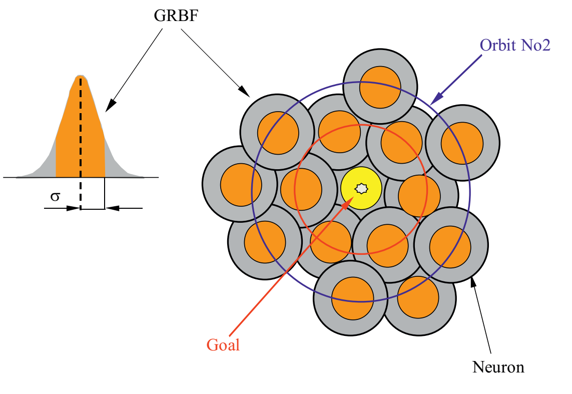

Fig. 3. Location of the GRBF in the space (dim = 2)

That forces the robot to move to the direction that the potential field decrees, where $\vec{\nabla}$ the gradient with respect to q and represents the coordinates of the robot position. The complete potential field is a superposition of contributions from obstacles, waypoint (if applicable), and the goal

$$U(q)=\sum_{j=1}^{n_0}U_j^0(q)+\sum_{j=1}^{n_w}U_j^w(q)+U^g(q) \tag{25}$$

where n_0 and n_w denote the number of obstacles and waypoints, respectively, and U_j^0 and U_j^w are their potential. U^g is the potential generated by the goal (navigation target).

In our approach, a neuron plays the role of the robot, the other neurons play the roles of the obstacles and the current true state of the system, is the goal point.

Different potential functions have been proposed in literature. The most commonly used attractive potential take the form [19, 22]:

$$U_{att}(q)=\frac{1}{2}\beta\rho^m\big(q,q_{goal}\big) \tag{26}$$

where β is a positive scaling factor, $\rho(q,q_{goal})-q$ is the distance between the neuron q and the goal q_{goal}, and $m=1$ or 2, the attractive potential is conic in shape and the resulting attractive force has constant amplitude except at the goal, where U_{att} is singular. For $m=2$, the attractive potential is parabolic in shape. The corresponding attractive force is then given by the negative gradient of the attractive potential

$$F_{att}(q)=-\nabla U_{att}(q)=\beta\frac{(q_{goal}-q)}{\|q_{goal}-q\|} \tag{27}$$

which is a constant force on the space: it does not tend to infinity with increasing distance from q_{goal} However, it is not zero at q_{goal}.

One commonly used repulsive potential function takes the following form [1]

$$U_{rep}(q)=\begin{cases}\frac{1}{2}\eta\big(\frac{1}{\rho(q,q_{neu})}-\frac{1}{\rho_0} & \text{if } \rho(q,q_{neu})\le\rho_0\,,\\ 0 & \text{if } \rho(q,q_{neu})>\rho_0\end{cases} \tag{28}$$

where η is a positive scaling factor, $\rho(q,q_{neu})$ denotes the minimal distance from the center of neuron q and the center of other neuron, q_{neu} denotes the center of the nearest neuron, and ρ_0 is a positive constant denoting the distance of influence of the neuron. The corresponding repulsive force is given by

$$F_{rep}(q)=-\nabla U_{rep}(q)=$$
$$\begin{cases}\eta\big(\frac{1}{\rho(q,q_{neu}}-\frac{1}{\rho_0}\big)\frac{1}{\rho(q,q_{neu})^2}\nabla\rho(q,q_{neu}) & \text{if } \rho(q,q_{neu})\le\rho_0,\\ 0 & \text{if } \rho(q,q_{neu})>\rho_0\,.\end{cases} \tag{29}$$

The total force applied to the neuron is the sum of the attractive force and the sum of the repulsive force

$$F_{total}=F_{att}+\sum F_{rep}\,. \tag{30}$$

This determines the motion of the neuron.

The attractive and the repulsive phenomena are given by Fig. 2.

The parameters β, η and ρ_0 are chosen so that we obtain a scenario similar to the Fig. 3. We obtain concentrations of neurons in a balls of dimension $(n+m)$ centered in the desired point. By construction we obtain several layers (or orbit) of radius $r_i\approx i\sigma$ were i is the rank of the orbit and σ is the width of the GRBF.

4.3 Determination of the number N

The proposed self-organizing RBF network is capable of adding or removing RBFs to ensure the desired approximation accuracy and at the same time to keep the appropriate network complexity.

4.3.1 Adding RBFs

As the system trajectory evolves in time, the approximation error e is measured. We first check if the Euclidean norm of the approximation error e exceeds a predetermined threshold $e_{\max}$, and the period between the two adding operations is greater than the minimum response time T_r. $e_{\max}$ and T_r are design parameters. If

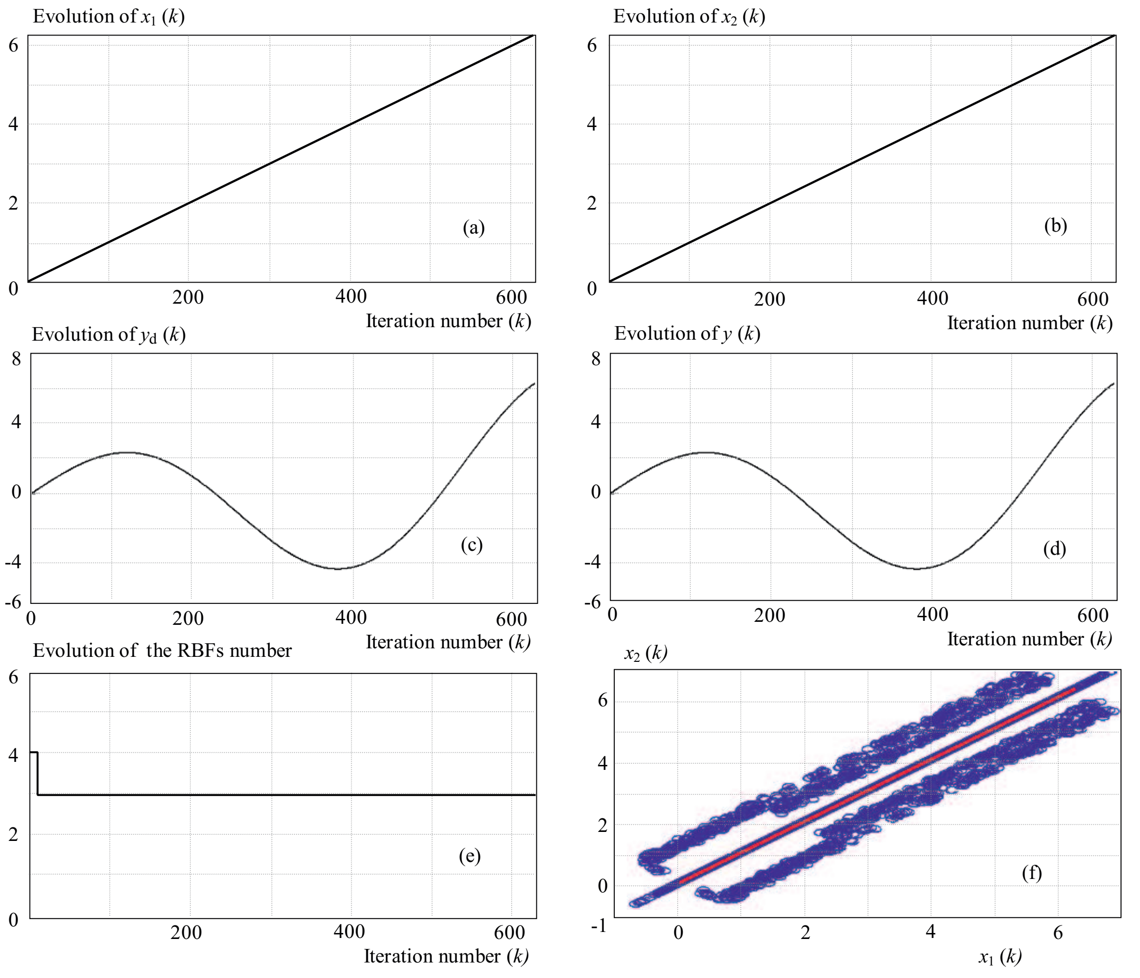

Fig. 4. Evolution of: (a) – x_1 and, (b) – x_2 respectively; (c) – the desired function and, (d) – its approximation respectively, (e) – the RBFs number, (f) – location of RBF center (o) and state (-)

these two conditions are satisfied, we recruit a new neuron which will be placed on a neighborhood of the last orbit. Its noted, that the recruitment is made by a single neuron. By consequence, the recruitment phase does not perturb the network.

4.3.2 Removing RBFs

The RBF removing operation is also implemented sequentially for all N coordinates. We first measure the approximation error e. If the Euclidean norm of the approximation error e is smaller than τe_{max}, where $\tau \in (0,1)$ is a design parameter, we remove a neuron from the last orbit. It is in noted, that the pruning is also made by a single neuron, which aims at not provoking an oscillation of the output response.

4.4 Neural network adaptive control Algorithm

Choose the design parameters A_c, Q, $f_{\max}$, $g_{\min}$, $e_{\max}$, M_x, M_w τ T_r. Initialize some GRBFs in a neighborhood of the initial condition and the weight matrix W of the initial RBF network. In each sampling period, repeat the following steps.

1) Compare the current and the desired output of the system to obtain the approximation error $e = y - y_d$.
2) If $e > e_{max}$ and the period between two adding operation is greater than T_r, go to 3); otherwise, go to 4).
3) Add a new neuron wich will be placed on a neighborhood of the last orbit. The radius of this orbit is given by: $r_{i_{last}} = (i_{last}+1).\sigma$; where i_{last} is the rank of the last layer.
4) If $e \le \tau.e_{max}$, go to 5); otherwise go to 6).
5) Remove a neuron from the last orbit
6) Update the weight matrix W using (9) and (10).
7) Calculate output of a direct adaptive neural controller using (5)
8) Determination of the motion of the GRBF in the space using (30).

5 SIMULATION RESULTS

In this section, we test our proposed real-time self-organizing RBF network approximator on two examples: the first one deal with the on-line identification of a nonlinear function using the gradient method. The second example treats the real-time approximation of nonlinear dynamical system using the proposed direct adaptive robust control given in Section 3.

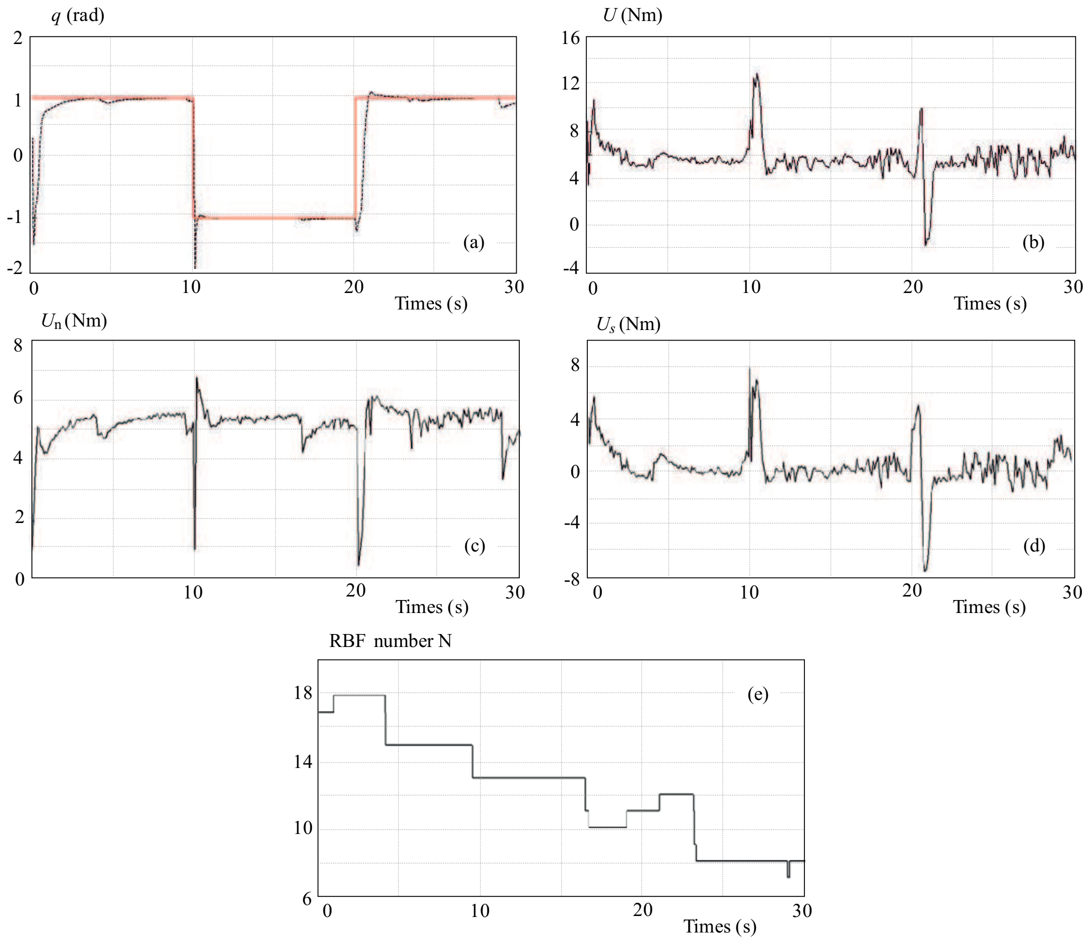

Fig. 5. (a)– output tracking of the angular position, (b) – control input signal, (c) – neural controller, (d)– supervisory controller, (e)– evolution of the RBF number

5.1 On-line identification of a nonlinear function

For this problem, the considered nonlinear function is described by the following equation

$$y(x_1, x_2) = x_2 \cos(x_1) + 2sin(x_1) . \tag{31}$$

The aim is to seek an optimal number of RBF using the proposed algorithm, to approximate adaptively the function with small error.

Based on simulation studies, the parameters of the proposed algorithm are chosen as follows

$$e_{\max} = 0.1 , \ \tau = 0.5 , \ \sigma = 0.5 , \ \rho_0 = 2\sigma , \ \eta = 1.5 .$$

The initial number of the hidden units is chosen as $N = 4$.

Figure 4 shows the simulation results of the process using the proposed algorithm. Figures 4(a) and 4(b) show respectively, the evolution of $x_1(t)$ and $x_2(t)$. The evolutions of the desired function and its estimate are given by figs. 4(c) and 4(d) respectively. It is clear that the approximation error is so acceptable. Figure 4(e) represents the evolution of the hidden units number. It is noted that we used a minimal number of neuron on this simulation. Figure 4(f) represents the localization of the centers of the used RBFs. It is obvious that the centers are joined together all around the state (x_1, x_2) to be estimated.

5.2 Neural network Adaptive control

This example illustrates a one-link rigid robotic manipulator. The dynamic equation of the one-link rigid robotic manipulator is given by [4]

$$ml^2\ddot{q} + d\dot{q} + mlg\cos(q) = u , \tag{32}$$

where the link is of length l and masse m, and q is the angular position with initial value $q(0) = 0.1$ and $\dot{q}(0) = 0$.

The parameters m, l, d and g are

$$m = 1\,\mathrm{kg}, l = 1\,\mathrm{m}, d = 0.1 \text{ and } g = 10\,\mathrm{m/s}^2.$$

The above dynamical equation can be written as the following state equation

$$\begin{aligned} \dot{x}_1 &= x_2 , \\ \dot{x}_2 &= \frac{1}{ml^2}\left[-dx_2 - m\lg\cos(x_1)\right] + \frac{1}{ml^2}u , \\ y &= x_1 . \end{aligned} \tag{33}$$

The output of the closed-loop system has to track a desired output y_d by using the proposed control scheme. From (30), we choose $f_{\max} = 1$ and $g_{\min} = 0.8$, $K =$

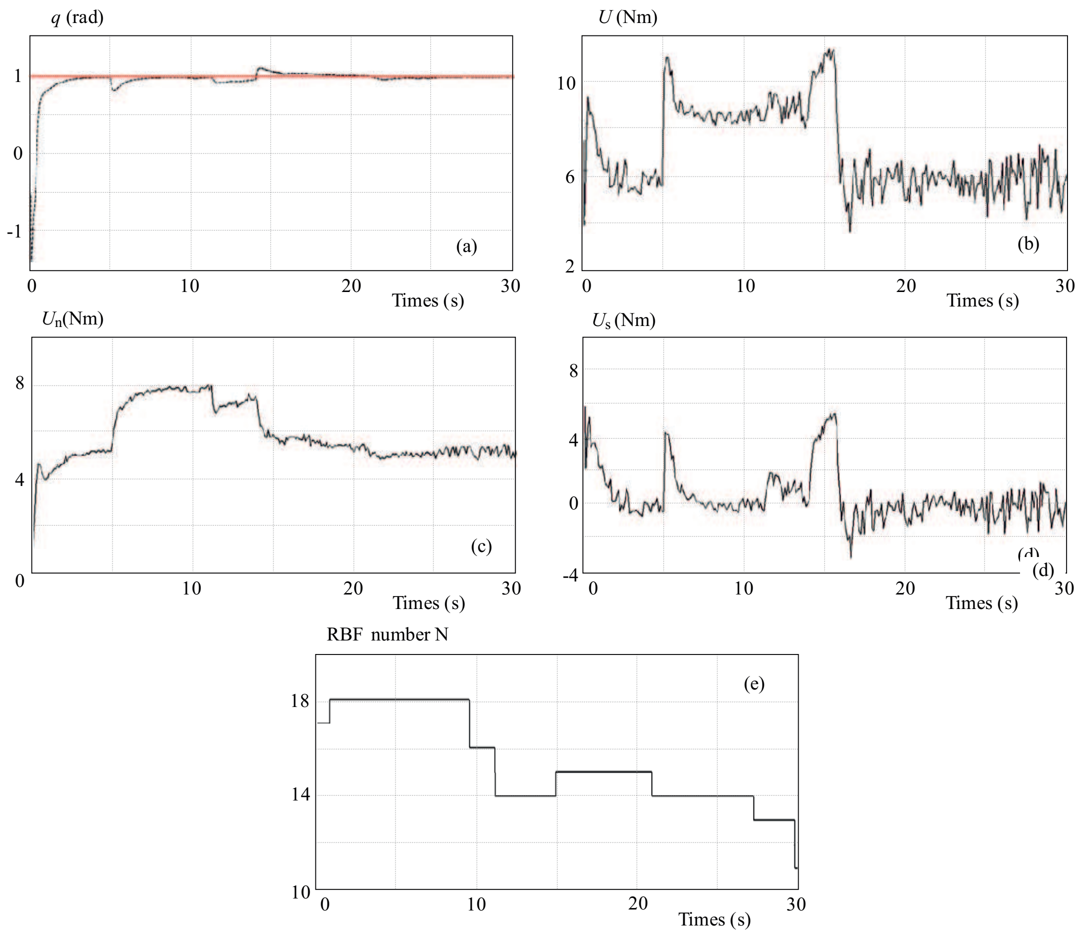

Fig. 6. (a) – output tracking of the angular position, (b) – control input signal, (c) – neural controller, (d) – supervisory controller, (e) – Evolution of the RBF number

$[1 \quad 2]$, $Q = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$, $A_c = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}$, and $P = \begin{bmatrix} 4.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix}$.

In the following simulations, the number of the hidden units is chosen as $N = 9$, the weights $w(0)$ are chosen randomly in the interval $[-0.5\,,\, 0.5]$. Moreover, we choose $M_x = 1.1$ and $M_W = 10$. Figure 5 shows the simulation results of the process using the proposed algorithm. Figures 5(a) and 5(b) show the output response of the system and the corresponding control action control based on the neural adaptive state feedback controller. It is obvious that satisfactory output tracking performances have been achieved through the proposed control scheme. Figures 5(c) and 5(d) represent respectively the corresponding neural controller and the supervisory controller outputs. The evolution of the RBFs number is presented in Fig. 5(e). To test the robustness of the proposed algorithm, some disturbances are applied to the considered system. We consider a variation on the mass of the link as follows

$$\begin{cases} m = 1\,\mathrm{kg} & \text{if } t \le 5s\,, \\ m = 1.5\,\mathrm{kg} & \text{if } 5s < t \le 14s\,, \\ m = 1\,\mathrm{kg} & \text{if } t > 14s \end{cases}$$

The initial conditions are chosen as in the first simulation, but only $f_{\max}$ and $g_{\min}$ are modified as

$$f_{\max} = 2 \quad \text{and} \quad g_{\min} = 0.6\,.$$

Figures 6(a) and 6(b) show the output response of the system and the corresponding control based on the neural adaptive state feedback controller. It is well seems that the perturbations are well rejected using the proposed control scheme. Figures 6(c) and 6(d) represent respectively the corresponding neural controller and the supervisor controller outputs. Figure 6(e) represents the evolution of the hidden units number. It is clear that we used a minimal number of neuron on this simulation.

Comparing with other works [3, 4, 6, 7], it is so clear that we used a minimal number of neuron while respecting the imposed performances. We can notice too, that the recruitment is made by a single neuron. By consequence, the recruitment phase does not perturb the network.

6 CONCLUSION

A direct adaptive neural control for a class of nonlinear system is presented. an important contribution in our proposed scheme is that the exact knowledge of the system model $(f(X))$ is not necessary. The adaptation law to adjust these parameters is proposed based on the Lyapunov approach. Moreover, the stability of the system can be also analyzed and guarantied by introducing the supervisory controller and the modified adaptation law with projection. Moreover, a novel neural network architecture, is proposed and shown to be useful in approximating the unknown nonlinearities of dynamical systems. In the variable structure neural network, the number of basis functions can be either increased or decreased with time according specified design strategies so that the network will not overfit or underfit the data set. Based on the Gaussian radial basis function (GRBF) variable neural network. The location of the centers of the GRBFs is analyzed using a new method inspired from evolutionary artificial potential fields method combined with a pruning algorithm.

References

[1] QIXIN, C.—YANWEN, H.—JINGLIANG, Z.: An Evolutionary Artificial Potential Field Algorithm for Dynamic Path Planning of Mobile Robot, IEEE/RSJ Int. conf. on Intelligent Robots and Systems, Beijing, Oct 2006, pp. 3331–3336.

[2] SHI, D.—YEUNG, D. S.—GAO, J.: Sensitivity Analysis Applied to the Construction of Radial Basis Function Networks, Neural networks **18** (2005), 951–957.

[3] LIU, G. P.—KADIRKAMANATHAN, V.—BILLINGS, S. A.: Variable Neural Networks for Adaptive Control of Nonlinear Systems, IEEE Trans. Trans. Syst. Man Cybern. B. Cybern. **398** No. 1 (Feb 1999), 34–43.

[4] MEKKI, H.—CHTOUROU, M.—DERBEL, N.: Variable Structure Neural Networks for Adaptive Control of Nonlinear Systems using the Stochastic Approximation, Simulation Modeling Practice and Theory **14** (2006), 1000–1009.

[5] MEKKI, H.—CHTOUROU, M.—DERBEL, N.: A Robust Adaptive Control using Neural Network, Int. J. Modelling, Identification and control **2** No. 1 (2007).

[6] LIAN, J.—LEE, Y.—SUDHOFF, S. D.—ZAK, S. H.: Variable Structure Neural Network Based Direct Adaptive Robust Control of Uncertain Systems, American Control Conference, Seattle, Washington, June 2008.

[7] LIAN, J.—LEE, Y.—SUDHOFF, S. D.—ZAK, S. H.: Self-Organizing Radial Basis Function Netwoek for Real-Time Approximation of Continuous-Time Dynamical Systems, IEEE Transactions on neural network, **19** No. 3 (Mar 2008).

[8] JR, J. R. B.—NICOLETTI, M. C.: A Multiclass Version of a Constructive Neural Network Algorithm Based on Linear Separability and Convex Hull, ICANN, Part II, LNCS, 2008, pp. 723–733.

[9] NORIEGA, J. R.—WANG, H.: A Direct Adaptive Neural Network Control for Unknown Nonlinear Systems and its Application, IEEE Transaction on Neural Networks **9** No. 1 (1998), 721–738.

[10] SALAHSHOOR, K.—JAFARI, M. R.: On-Line Identification of Non-Linear Systems using Adaptive RBF-Based Neural Networks, International Journal of Information Science and Technology **5** No. 2 (July/Dec 2007), 99–121.

[11] AUGUSTO, L.—MELEIRO, D. C.—ZUBEN, F. J. V.—FILHO, R. M.: Constructive Learning Neural Network Applied to Identification and Control of Fuel-Ethanol Fermentation Processjour Engineering Applications of Artificial Intelligence.

[12] MA, L.—KHORASANI, K.: New Training Strategies for Constructive Neural Networks with Application to Regression Problems, Neural networks **17** (2004), 589–609.

[13] NICOLETTI, M. D. C.—BERTINI, J. R.: An Empirical Evaluation of Constructive Neural Network Alghorithms n classification tasks, Int. J. Innovative Computing and Application **1** No. 1, 1–13 2007.

[14] NICOLETTI, M. D. C. *et al*: Constructive Neural Network Algorithms for Feedforward Architectures Suitable for Classification Tasks, Springer-Verlag Berlin Heidelberg, SCI 258.

[15] GROCHOWSKI, M.—DUCH, W.: Constructive Neural Network Algorithms that Solve Highly Non-Separable Problems, Springer-Verlag Berlin Heidelberg, SCI 258.

[16] PARK, M. G.—JEON, J. H.—LEE, M. C.: Obstacle Avoidance for Mobile Robots using Artificial Potential Field Approach with Simulated Annealing, IEEE International Symposium on Industrial Electronics, vol. 3, 2001, pp. 1530–1535.

[17] JAFARI, M. R.*et al*: On-Line Identification of Non-Linear Systems using an Adaptive RBF-based neural Network, Proceeding of WCECS, 58-3, San Francisco, Oct 2007.

[18] KARAYIANNIS, N. B.—MI, G. W.: Growing Radial basis Neural Networks: Merging Supervised and Unsupervised Learning with Network Growth Techniques, IEEE Trans. Neural Netw. **8** No. 6 (Nov 1997), 1492-1506,.

[19] AHANG, P. Y.—LÜ, T. S.—SONG, L. B.: Soccer Robot Path Planning Based on the Artificial Potential Field Approach with Simulated Annealing, Robotica **22** (2004), 563–566.

[20] VADAKKEPAT, P.—LEE, T. H.—XIN, L.: Application of Evolutionary Artificial Potential Field in Robot Soccer System, IFSA World Congress and 20th NAFIPS int. Conf., vol. 5, Vancouver Canada, july 2001, pp. 2781–2785.

[21] LEE, S.—KIL, R. M.: A Gaussian Potential Function Network with Hierachiclally Self-Organizing Learning, Neural Netw. **4** No. 2 (1996), 207–224.

[22] GE, S. S.—CUI, Y. J.: New Potential Functions for Mobile Robot Path Planning, IEEE Transactions on Robotics and automation **16** No. 5 (Oct 2000), 615–620.

[23] GE, S. S.—CUI, Y. J.: New Potential Functions for Mobile Robot Path Planning, IEEE Trans. on Robot. and Auto. **16** No. 5 (Oct 2000).

[24] CHANG, W. D.—HWANG, R. C.—HSIEH, J. G.: Stable Direct Adaptive Neural Controller of Nonlinear Systems Based on Single Auto-Tuning Neuron, Neurocomputing **48** (2002), 541–554.

Received 13 July 2012

Hassen Mekki received the BS, MS, and PhD degrees in electrical engineering, from National School of Engineers of Tunis-Tunisia, National School of Engineers of Tunis-Tunisia, and National School of Engineers of Sfax-Tunisia, respectively. He is currently a professor in the department of electrical Engineering of National School of Engineers of Sousse-Tunisia. His current research interests include visual serving, neural and fuzzy systems intelligent and adaptive control. He is author of more than five papers in international journals.

Mohamed Chtourou received his BS, MS, and PhD degrees in electrical engineering, from National School of Engineers of Sfax-Tunisia, Institut Nationale des Sciences Apliques de Toulouse-France and Institut Polytechnique de Toulouse-France, respectively. He is currently a professor in Department of Electrical Engineering of National School of Engineering of Sfax-Tunsia. His current research interests include neural and fuzzy systems, intelligent and adaptive control. He is author and co-author of more than ten papers in international journals.