SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 8, No. 3, November 2011, 307-323

UDK: 681.513.6 **DOI: 10.2298/SJEE1103307B**

Neural Feedback Linearization Adaptive Control for Affine Nonlinear Systems Based on Neural Network Estimator

Mohamed Bahita[1], Khaled Belarbi[2]

Abstract: In this work, we introduce an adaptive neural network controller for a class of nonlinear systems. The approach uses two Radial Basis Functions, RBF networks. The first RBF network is used to approximate the ideal control law which cannot be implemented since the dynamics of the system are unknown. The second RBF network is used for on-line estimating the control gain which is a nonlinear and unknown function of the states. The updating laws for the combined estimator and controller are derived through Lyapunov analysis. Asymptotic stability is established with the tracking errors converging to a neighborhood of the origin. Finally, the proposed method is applied to control and stabilize the inverted pendulum system.

Keywords: Adaptive control, Control gain estimation, Feedback linearization, Radial basis function network.

1 Introduction

Adaptive control seems today a natural strategy to tackle the stabilization and tracking of highly uncertain nonlinear dynamical systems. Although linear systems are very well understood and controlled, linear control is not enough to guarantee stability and performance of nonlinear systems.

A milestone in the extension of linear control techniques to nonlinear systems has been the development of nonlinear geometric control. Recent research involving differential geometric methods [12, 13, 17] has rendered the design of controllers for a class of nonlinear systems somewhat systematic. This nonlinear control theory, called feedback linearization, is based on coordinate transformations by which a class of nonlinear systems can be transformed into linear systems through feedback. With the advent of feedback linearization, adaptive control found its way into nonlinear control. The combination of adaptive control and feedback linearization applied to flight control can be found in [16]. In most of the classical adaptive control literature it is common to

[1]Faculty of Hydrocarbons and Chemistry (FHC), University of Boumerdes, Boumerdes 35000, Algeria; E-mail mbahita@yahoo.fr
[2]Faculty of Engineering, University of Constantine, Constantine 25000, Algeria; E-mail: kbelarbi@yahoo.com

assume the unknown dynamics to have a known structure with unknown parameters entering linearly in the dynamics. The linear parameterization of unknown dynamics poses serious obstacles in adopting adaptive control algorithms in practical applications, because it is difficult to fix the structure of the unknown nonlinearities. This fact has been the motivating factor behind the interest in on-line function approximators to estimate the unknown function. The most common function approximators used in adaptive control are artificial neural network and fuzzy logic systems.

The ability of neural networks to approximate uniformly continuous functions has been proven in several articles [6, 7, 10, 11]. Neural networks (NNs) for identification and control were first proposed by Narendra and Parthasarathy [14]. Results in this initial research were limited to simulation and no proofs of stability in the closed loop were provided. Tzirkel and Fallside proposed the use of neural network control providing proof of stability [19].

In the majority of proposed adaptive controller a complementary term, called supervisory term, is added to the output of the fuzzy inference system or of the neural network system controller in order to guarantee global stability using Lyapunov theory [20]. When the system is operating within a prescribed range, the supervisory controller is turned off. The difference between the various methods lies in the structure of the model and in the form of the supervisory term, the approximated control law is based on the certainty equivalence approach while the supervisory term can take different forms, but most often, it is based on the sliding mode technique. Sanner and Slotine first proposed Radial Basis Function (RBF) NNs for control of affine systems and provided rigorous analysis of stability. They introduced a sliding mode component in the control law for keeping the evolving dynamics within the predetermined compact set of interest [15].

In this work, we introduce an alternative adaptive controller for affine nonlinear systems without the use of the supervisory control term. The architecture employs two RBF networks: the first RBF network is used to approximate the ideal control law which cannot be computed, because the system dynamics are unknown, and the second network is used to estimate the virtual control gain which is an unknown nonlinear function of the states.

From the point of view of possible drawbacks compared to the MLP (Multilayered Perceptrons) which have only the connection weights to adjust, the main drawback of the RBF networks is that they are characterised by two sets of adjustable parameters: the centres of the radial basis functions and the connections weights. On the other hand, the main advantage of RBF is that their output depends linearly on the connection weights and thus the training becomes a linear optimisation problem. However this is not possible for the centres. Usually, in RBF based adaptive control, the online adaptation is

concerned only with the connections weights and the centres of the basis functions are fixed offline. In this work, we propose to online adjust both the centres of the basis functions and the connections weights.

The centres of the RBF networks are adapted on line using the k-means algorithm [5, 21]. The global stability of the resulting closed loop system is established and the updating laws for the parameters are derived using Lyapunov stability theory.

This paper is organised as follows: in Section 2, the problem formulation is introduced, in Section 3, the stability analysis is developed and the adaptive laws are derived, in Section 4, the indirect adaptive RBF-RBF controller is used in simulation for controlling the inverted pendulum system. Section 5 concludes the paper.

2 Problem Formulation

Consider the following nonlinear system

$$x^{(n)} = f(x, \dot{x}, ..., x^{(n-1)}) + b(x)u, \quad y = x, \tag{1}$$

where $u \in R$ and $y \in R$ are the input and output of the system respectively, $f(\underline{x})$ and $b(x)$ are unknown nonlinear functions. We assume that the state vector $\underline{x} = (x_1, x_2, ..., x_n)^T = (x, \dot{x}, ...x^{(n-1)})^T \in R^n$ is available for measurement. The control objective is to force $y(t)$ to follow a given bounded reference signal $y_m(t)$, under the constraints that all involved signals must be bounded. Define the error vector as

$$\underline{e} = (e, \dot{e}, ..., e^{(n-1)})^T \in R^n. \tag{2}$$

In order to design the controller we consider the following two steps:

Step 1: We choose u to cancel the nonlinearities in the nonlinear system so that the closed-loop dynamics is in a linear form, and guarantees tracking convergence, as we have mentioned previously, this is called feedback linearization [12, 13, 17]. If the functions $f(\underline{x})$ and $b(\underline{x})$ are known and assuming $b(\underline{x})$ to be non zero (this is an usual assumption) then from (1), the optimal control law is:

$$u^* = \frac{1}{b(\underline{x})}(v - f(\underline{x})). \tag{3}$$

Substituting (3) into (1), we can cancel the nonlinearities and obtain the simple input-state relation:

$$x^{(n)} = v. \tag{4}$$

Step 2: We choose the artificial input v (an equivalent input) as a simple linear pole-placement controller $v = y_m^{(n)} - K^T \underline{e}$ that guarantees the stability of the overall system, with

$$K = (k_0, k_1, ..., k_{n-1})^T \in R^n \tag{5}$$

chosen so that the polynomial:

$$s^n + k_{n-1}s^{n-1} + \cdots + k_0 = 0 \tag{6}$$

has all its roots strictly in the left-half complex plane. Then the control law is:

$$u^\bullet = \frac{1}{b(\underline{x})}(y_m^{(n)} - K^T \underline{e} - f(\underline{x})) . \tag{7}$$

Since $e = y - y_m$, then:

$$e^{(n)} = y^{(n)} - y_m^{(n)} . \tag{8}$$

Substituting (7) into (1), using (8) we have:

$$e^{(n)} + k_{n-1}e^{(n-1)} + \cdots + k_0 e = 0 , \tag{9}$$

which implies that $\lim e(t) \to 0$ as $t \to \infty$, i.e., exponentially stable dynamics [18], which is the main objective of control. However, since $f(\underline{x})$ and $b(\underline{x})$ are unknown, the ideal control $u^\bullet$ of (7) cannot be implemented. Our purpose is to design an RBF network $u(\underline{x}, \underline{\theta})$ to approximate this ideal control law $u^\bullet$ and based on this approximation we design a second RBF network to approximate the unknown function $b(\underline{x})$ (the virtual control gain). The following sections describe and analyze the proposed controller.

3 The Neural Adaptive Controller

As mentioned above, the proposed adaptive controller is composed of two RBF networks. The RBF network can be considered as a two-layer network with one hidden layer. The output depends linearly on the weights, the training is thus a linear optimization problem [8]. More explicitly, the RBF network performs the transformation:

$$f_r : R^n \to R ,$$

with:

$$b(\underline{x}, \underline{\theta}_b) = \sum_{i=1}^{nr} \xi_i \theta_{bi} = \underline{\theta}_b^T \xi_2(\underline{x}), \quad \xi_i = \psi\left(\left\|\underline{x} - c_i\right\|_2\right), \tag{10}$$

$\underline{x}$ is the input vector, ψ is a non linear function, called radial basis function, θ_i are connection weights (parameters) between the hidden layer and the output

layer, c_i are centres of the basis functions, nr is the number of basis functions. The most common basis function is the Gaussian function:

$$\psi(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right), \tag{11}$$

with $r = \left\|\underline{x} - c_i\right\|_2$, σ is an associated constant to the function $\psi(r)$ and represents the width of the Gaussian function. Although the RBF network is usually considered linear parameterized, by adjusting the centres and the widths this type of neural network structure becomes nonlinearly parameterized. In this work, we propose to online adjust both the centres of the basis functions and the connection weights. The connection weights of the two RBF networks: the controller and the estimator are to be adapted and will be represented respectively by the vectors $\underline{\theta}$ and $\underline{\theta}_b$. The *k*-means algorithm is used for the centres adjustment. The *k*-means algorithm [5, 21] is an unsupervised training method for data clustering. It consists in dividing the input space into nr classes as follows:

1. Choose a number of classes (n_r basis functions in our case).
2. Initialise the centres of the basis functions.
3. Compute the Euclidean distances between the centres of each basis function and the input vector $\underline{x}$, i.e.

$$dist(i) = \left\|\underline{x} - c_i\right\|_2, \quad i = 1, 2, \ldots, n_r, \tag{12}$$

then adjust the vector of centres c_i which corresponds to the minimum distance $dist(j) = \min\left\|\underline{x} - c_i\right\|_2$ using the following adaptation law [5]:

$$c_j(t) = c_j(t-1) + \delta(t)\left(\underline{x}(t) - c_j(t-1)\right) \tag{13}$$

where j is the index of the basis function which corresponds to the minimum Euclidean distance $dist(j)$ and $\delta(t)$ is a gain belonging to the interval [0 1], and which tends to zero as $t \to \infty$. One adaptation law for this parameter as given in [2] is the following:

$$\delta(k) = \frac{\delta(t-1)}{\sqrt{1 + \text{int}(t/n_r)}}, \tag{14}$$

where t is the iteration, n_r is the number of basis functions, and *int* is the integer part of (t/n_r).

3. 1 The RBF adaptive controller

Now, supposing that the control u in (1) is the output of the first RBF controller $u(\underline{x},\underline{\theta})$, i.e., $u = u(\underline{x},\underline{\theta})$, then (1) becomes:

$$x^{(n)} = f(\underline{x}) + b(\underline{x})u(\underline{x},\underline{\theta}), \tag{15}$$

now adding and subtracting $b(\underline{x})u^*$ to (15) we will have:

$$x^{(n)} = f(\underline{x}) + b(\underline{x})u(\underline{x},\underline{\theta}) + b(\underline{x})u^\bullet - b(\underline{x})u^\bullet. \tag{16}$$

Substituting (7) into (16), we obtain:

$$x^{(n)} = f(\underline{x}) + b(\underline{x})u(\underline{x},\underline{\theta}) - b(\underline{x})u^\bullet + y_m^{(n)} - K^T\underline{e} - f(\underline{x}), \tag{17}$$

thus:

$$x^{(n)} - y_m^{(n)} + K^T\underline{e} = b(\underline{x})(u(\underline{x},\underline{\theta}) - u^\bullet). \tag{18}$$

Based on $y = x$ in (1), using (2) and (8), equation (18) leads to the error system:

$$\dot{\underline{e}} = A_c\underline{e} + b_c(u(\underline{x},\underline{\theta}) - u^\bullet), \tag{19}$$

with:

$$A_c = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -k_0 & -k_1 & -k_2 & \cdots & -k_{n-2} & -k_{n-1} \end{bmatrix}, \quad b_c = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ b(\underline{x}) \end{bmatrix}. \tag{20}$$

We now study the stability of the system in order to develop an adaptive law to adjust the parameter vector $\underline{\theta}$ of the first RBF controller. Define the parameter vector $\underline{\theta}^\bullet$ as the optimal parameter vector which corresponds to the ideal approximator control signal $u(\underline{x},\underline{\theta}^\bullet)$ of the ideal control signal u^* of (7). It has been proven that (10), the output of a RBF network , can approximate over a compact set Ω_Z, any smooth function up to a given degree of accuracy [4, 15]. It can thus be used to approximate the ideal control law u^* as given in (7). It follows that: $u^* \approx u(\underline{x},\underline{\theta}^\bullet)$. Thus, the error equation (19) can be rewritten as

$$\dot{\underline{e}} = A_c\underline{e} + b_c(u(\underline{x},\underline{\theta}) - u(\underline{x},\underline{\theta}^\bullet)). \tag{21}$$

Based on (10) we have:

$$u(\underline{x},\underline{\theta}) = \underline{\theta}^T\xi_l(\underline{x}), \tag{22}$$

$$u(\underline{x},\underline{\theta}^{\bullet}) = \underline{\theta}^{*T}\xi_1(\underline{x}) \tag{23}$$

let $\phi = \underline{\theta} - \underline{\theta}^{*}$ and using (22) and (23), (21) becomes:

$$\dot{\underline{e}} = A_c\underline{e} + b_c\phi^T\xi_1(\underline{x})\,. \tag{24}$$

Define the Lyapunov function candidate:

$$V = \frac{1}{2}\underline{e}^T P\underline{e} + \frac{1}{2\gamma}\phi^T\phi\,, \tag{25}$$

where γ is a positive constant and P is a solution of the Lyapunov equation:

$$A_c^T P + PA_c = -Q \quad \text{with} \quad Q > 0\,. \tag{26}$$

Differentiate V with respect to time:

$$\dot{V} = \frac{1}{2}\dot{\underline{e}}^T P\underline{e} + \frac{1}{2}\underline{e}^T P\dot{\underline{e}} + \frac{1}{2\gamma}\dot{\phi}^T\phi + \frac{1}{2\gamma}\phi^T\dot{\phi}\,, \tag{27}$$

using (24) and (26), we have:

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} + \underline{e}^T Pb_c\phi^T\xi_1(\underline{x}) + \frac{1}{\gamma}\phi^T\dot{\phi}\,. \tag{28}$$

Let P_n be the last column of P, and using (20) we obtain:

$$\underline{e}^T Pb_c = \underline{e}^T P_n b(\underline{x}) \tag{29}$$

Substituting (29) into (28), we have:

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} + \frac{1}{\gamma}\phi^T(b(\underline{x})\gamma\underline{e}^T P_n\xi_1(\underline{x}) + \dot{\phi})\,, \tag{30}$$

in order to make $\dot{V} \le 0$ (according to Lyapunov stability theory to guarantee the stability convergence), we set the second term of $\dot{V}$ in (30) equal to zero, i.e.,

$$\frac{1}{\gamma}\phi^T(b(\underline{x})\gamma\underline{e}^T P_n\xi_1(\underline{x}) + \dot{\phi}) = 0\,, \tag{31}$$

recalling that $\dot{\phi} = \dot{\underline{\theta}} - \dot{\underline{\theta}}^{*} = \dot{\underline{\theta}}$, because the optimal parameter vector $\underline{\theta}^{\bullet}$ is constant $\dot{\underline{\theta}}^{*} = 0$, thus $\dot{\phi} = \dot{\underline{\theta}}$ and from (31) we obtain the adaptation law:

$$\dot{\underline{\theta}} = -\gamma b(\underline{x})\underline{e}^T P_n\xi_1(\underline{x})\,. \tag{32}$$

From (30), it follows that:

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} \le 0\,. \tag{33}$$

Now, from (32), we see that $\dot{\underline{\theta}} = -\gamma b(\underline{x})\underline{e}^T P_n \xi_1(\underline{x})$ is a function of the unknown function $b(\underline{x})$, then it remains to develop an estimation of $b(\underline{x})$ using a second RBF network in order to be able to adapt the parameters vector $\underline{\theta}$ of the first RBF neural controller. Based on a first initialized value (different from zero) of $b(\underline{x})$, we can use the adaptive law (32) to adjust the parameters vector $\underline{\theta}$ of the controller in the first iteration, then the control input u is given a first value using the first RBF neural system with output $u(\underline{x},\underline{\theta}) = \underline{\theta}^T \xi_1(\underline{x})$, i.e., the first value of $u(\underline{x},\underline{\theta})$ exists. This is used to compute a first value of $b(\underline{x},\underline{\theta}_b)$, and in general to adjust the parameter vector $\underline{\theta}_b$ of the second RBF network with output $b(\underline{x},\underline{\theta}_b) = \underline{\theta}_b^T \xi_2(\underline{x})$ using an adaptive law as we will see below.

3.2 The second RBF neural adaptive estimator

From (15) we have:

$$x^{(n)} = f(\underline{x}) + b(\underline{x})u(\underline{x},\underline{\theta}). \tag{34}$$

Supposing that $f(\underline{x})$ is known and as the first value of $u(\underline{x},\underline{\theta})$ exists (as mentioned previously), then from (34):

$$b^{\bullet}(\underline{x}) = \frac{1}{u(\underline{x},\underline{\theta})}[v_b - f(\underline{x})], \tag{35}$$

with $u(\underline{x},\underline{\theta}) \neq 0$. The case when $u(\underline{x},\underline{\theta}) = 0$ will be discussed later in Remark 1. We chose an artificial input v_b as in Step 2 of Section 2, i.e.,

$$v_b = y_m^{(n)} - K_b^T \underline{e}, \tag{36}$$

with the vector K_b chosen so that the polynomial $s^n + k_{bn-1}s^{n-1} + \cdots + k_{b0} = 0$ has all its roots strictly in the left-half complex plane. Then equation (35) can be written as

$$b^{\bullet}(\underline{x}) = \frac{1}{u(\underline{x},\underline{\theta})}[y_m^{(n)} - K_b^T \underline{e} - f(\underline{x})]. \tag{37}$$

Substituting (37) into (34), and using (8), we will have:

$$e^{(n)} + K_b^T \underline{e} = 0. \tag{38}$$

Or equivalently

$$e^{(n)} + k_{bn-1}e^{(n-1)} + \cdots + k_{b0}e = 0, \tag{39}$$

this implies that $\lim e(t) \to 0$ as $t \to \infty$ (exponentially stable dynamics). From (37), $u(\underline{x},\underline{\theta})$ is known (the first value is computed), i.e., it is the first RBF

neural approximation of the ideal control, but $f(\underline{x})$ is not known, then $b^{\bullet}(\underline{x})$ in (37) cannot be implemented. Our second purpose is to construct a second RBF system to approximate the optimal expression $b^{\bullet}(\underline{x})$ of $b(\underline{x})$. In the equation (34), we can then replace $b(\underline{x})$ by its estimation $\hat{b}(\underline{x})$ which will be computed, thus (34) becomes:

$$x^{(n)} = f(\underline{x}) + \hat{b}(\underline{x})u(\underline{x},\underline{\theta}). \tag{40}$$

Now adding and subtracting $b^{\bullet}(\underline{x})u(\underline{x},\underline{\theta})$ to (40) we obtain:

$$x^{(n)} = f(\underline{x}) + \hat{b}(\underline{x})u(\underline{x},\underline{\theta}) + b^{\bullet}(\underline{x})u(\underline{x},\underline{\theta}) - b^{\bullet}(\underline{x})u(\underline{x},\underline{\theta}), \tag{41}$$

or equivalently:

$$x^{(n)} = f(\underline{x}) + (\hat{b}(\underline{x}) - b^{\bullet}(\underline{x}))u(\underline{x},\underline{\theta}) + b^{\bullet}(\underline{x})u(\underline{x},\underline{\theta}), \tag{42}$$

substituting (37) into (42), we can obtain:

$$\begin{aligned} x^{(n)} &= f(\underline{x}) + \left(\hat{b}(\underline{x}) - b^{\bullet}(\underline{x})\right)u(\underline{x},\underline{\theta}) + \\ &+ \frac{1}{u(\underline{x},\underline{\theta})}\left[y_m^{(n)} - K_b^T \underline{e} - f(\underline{x})\right]u(\underline{x},\underline{\theta}), \end{aligned} \tag{43}$$

or

$$x^{(n)} = y_m^{(n)} - K_b^T \underline{e} + \left(\hat{b}(\underline{x}) - b^{\bullet}(\underline{x})\right)u(\underline{x},\underline{\theta}). \tag{44}$$

Now define the parameter vector $\underline{\theta}_b^{\bullet}$ as the optimal parameter vector which guarantees the optimal estimation of $b^{\bullet}(\underline{x})$, We recall that it has been proven that an artificial neural network can approximate any nonlinear function to any desired degree [4, 15], the output of the RBF system can thus be used to approximate the optimal term $b^{\bullet}(\underline{x})$ of (37), then we can write:

$$b^{\bullet}(\underline{x}) \approx \hat{b}(\underline{x},\underline{\theta}_b^{\bullet}). \tag{45}$$

Based on (10), we have:

$$\hat{b}(\underline{x},\underline{\theta}_b) = \underline{\theta}_b{}^T \xi_2(\underline{x}) \quad \text{and} \quad \hat{b}(\underline{x},\underline{\theta}_b^{\bullet}) = \underline{\theta}_b^{\bullet T} \xi_2(\underline{x}), \tag{46}$$

if we define:

$$\varphi = \underline{\theta}_b - \underline{\theta}_b{}^{\bullet}. \tag{47}$$

Based on $y = x$ in (1), and using (8), (45), (46) and (47), equation (44) leads to the error system:

$$e^{(n)} + K_b^T \underline{e} = \varphi^T \xi_2(\underline{x})u(\underline{x},\underline{\theta}), \tag{48}$$

or equivalently:

$$\dot{\underline{e}} = A_b \underline{e} + \varphi^T \xi_2(\underline{x}) \underline{u}_b \,, \tag{49}$$

where:

$$A_b = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ -k_{b0} & -k_{b1} & -k_{b2} & \dots & -k_{bn-2} & -k_{bn-1} \end{bmatrix} \quad \text{and} \quad \underline{u}_b = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ u(\underline{x}, \underline{\theta}) \end{bmatrix}. \tag{50}$$

Let's now develop an adaptive law to adjust the parameter $\underline{\theta}_b$ of the second RBF network, $\hat{b}(\underline{x}, \underline{\theta}_b) = \underline{\theta}_b{}^T \xi_2(\underline{x})$. We choose a candidate Lyapunov function:

$$V_b = \frac{1}{2} \underline{e}^T P_b \underline{e} + \frac{1}{2\alpha} \varphi^T \varphi \,, \tag{51}$$

where α is a positive constant and P_b is a solution of the Lyapunov equation:

$$A_b^T P_b + P_b A_b = -Q_b \quad \text{with} \quad Q_b > 0 \,. \tag{52}$$

Differentiate V_b with respect to time, gives:

$$\dot{V}_b = \frac{1}{2} \dot{\underline{e}}^T P_b \underline{e} + \frac{1}{2} \underline{e}^T P_b \dot{\underline{e}} + \frac{1}{2\alpha} \dot{\varphi}^T \varphi + \frac{1}{2\alpha} \varphi^T \dot{\varphi} \,. \tag{53}$$

Using (49) and (52), then (53) becomes:

$$\dot{V}_b = -\frac{1}{2} \underline{e}^T Q_b \underline{e} + \frac{1}{\alpha} \varphi^T \left(\alpha \underline{e}^T P_b \underline{u}_b \xi_2(\underline{x}) + \dot{\varphi} \right). \tag{54}$$

Let P_{bn} be the last column of P_b, using part 2 of (50) we have:

$$\underline{e}^T P_b \underline{u}_b = \underline{e}^T P_{bn} u(\underline{x}, \underline{\theta}) \,, \tag{55}$$

substituting (55) into (54), we obtain:

$$\dot{V}_b = -\frac{1}{2} \underline{e}^T Q_b \underline{e} + \frac{1}{\alpha} \varphi^T (\alpha \underline{e}^T P_{bn} u(\underline{x}, \underline{\theta}) \xi_2(\underline{x}) + \dot{\varphi}) \,. \tag{56}$$

In order to make $\dot{V}_b \le 0$, we set the second term of $\dot{V}_b$ equal to zero, i.e.:

$$\frac{1}{\alpha} \varphi^T (\alpha \underline{e}^T P_{bn} u(\underline{x}, \underline{\theta}) \xi_2(\underline{x}) + \dot{\varphi}) = 0 \,. \tag{57}$$

Using (47), we obtain:

$$\dot{\underline{\theta}}_b = -\alpha \underline{e}^T P_{bn} u(\underline{x}, \underline{\theta}) \xi_2(\underline{x}) \,. \tag{58}$$

Remark 1: As we can see from (58) the adaptation of $\underline{\theta}_b$ is a function of $u(\underline{x},\underline{\theta})$, then when $u(\underline{x},\underline{\theta})=0$, the parameters vector $\underline{\theta}_b$ will not be adapted using equation (58) and will only keep its previous value before the term $u(\underline{x},\underline{\theta})$ has become zero. Besides, the case $u(\underline{x},\underline{\theta})=0$ in (35 or 37) will not cause any problem, because the optimal value $b^{\bullet}$ is computed with the output of the second RBF $b(\underline{x},\underline{\theta}_b)=\underline{\theta}_b^T\underline{\xi}_2(\underline{x})$.

4 Simulation Results

In this section, we test the performance of the proposed adaptive controller on the inverted pendulum system depicted in Fig 1. The dynamic equations of the inverted pendulum system are:

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= f(\underline{x})+b(\underline{x})u(t),\\ y &= x_1,\end{aligned} \tag{59}$$

where

$$f(\underline{x})=\frac{g\sin(x_1)-\dfrac{mlx_2^2\cos(x_1)\sin(x_1)}{M+m}}{l\left(\dfrac{4}{3}-\dfrac{m\cos^2(x_1)}{M+m}\right)},\quad b(\underline{x})=\frac{\dfrac{\cos(x_1)}{M+m}}{l\left(\dfrac{4}{3}-\dfrac{m\cos^2(x_1)}{M+m}\right)}, \tag{60}$$

$x_1=\theta$ is the angular position of the pendulum (see Fig. 1), $x_2=\dot{\theta}$ is the angular velocity of the pendulum. We use $g=9.8\,\mathrm{m/s^2}$, $M=1\,\mathrm{kg}$ is the mass of the cart, $m=0.1\,\mathrm{kg}$ is the mass of the pole and $l=0.5\,\mathrm{m}$ is the half length of the pole. The control objective is to make the pole of the pendulum track a sine wave trajectory $\theta_m=y_m=AM\sin(t)$ with amplitudes $AM=\pi/30$ during the time interval $t\in[0\quad 12.5]\,\mathrm{s}$, $AM=\pi/15$ during the time interval $t\in[12.5\quad 25]\,\mathrm{s}$, and finally with amplitude $AM=0$ as in [1, 3] during the remaining time interval Clearly, the derivatives of the reference y_m exist and are bounded.

The two RBF networks (the first RBF controller and the second RBF estimator) have five radial basis functions. The parameters $\underline{\theta}$ are initialised to 0, and $\underline{\theta}_b$ are all initialised to 0.5. Other choices have been tried, these last values have given a satisfactory transient performance. The centres of the basis functions of the two RBF systems are uniformly distributed in the interval

$[-2/3 \quad 2/3]$. K is chosen as $K=[k_0 \quad k_1]^T=[0.9 \quad 5]^T$, and Q in (26) is chosen as $Q=diag(127,127)>0$. Then by solving (26) we can obtain:

$$P=\begin{bmatrix} 376.9078 & 70.5556 \\ 70.5556 & 26.8111 \end{bmatrix}. \tag{61}$$

For the first RBF controller, we used $\gamma=10$, the input vector $\underline{x}$ is composed of two inputs $\underline{x}=[x_1 \; x_2]=[\theta \quad (K^T \underline{e}-\theta_m^{(2)})]$, the radial basis functions are chosen as Gaussian functions as given by (11), where the corresponding widths for every function is σ=1.32.

For the second RBF estimator, we used $\alpha=0.005$, the input vector $\underline{x}_b$ is composed of two inputs the error e and the variation of error $\mathrm{d}e$, i.e., $\underline{x}_b=[\theta-\theta_m \quad \dot{\theta}-\dot{\theta}_m]$, the radial basis functions are chosen as Gaussian functions as given by (11), where the corresponding widths for every function is $\sigma_b=0.54$.

The initial conditions are $(x_1(0), x_2(0))^T=(-0.2 \text{ rad}, 0 \text{ rad/sec})$ and the step size is set to $\mathrm{d}t=0.01$. The results of the simulation are shown in Figs. 2 to 6. The system output $y(t)$ (pole angle) is in continuous while the reference signal $y_m(t)$ is in dotted.

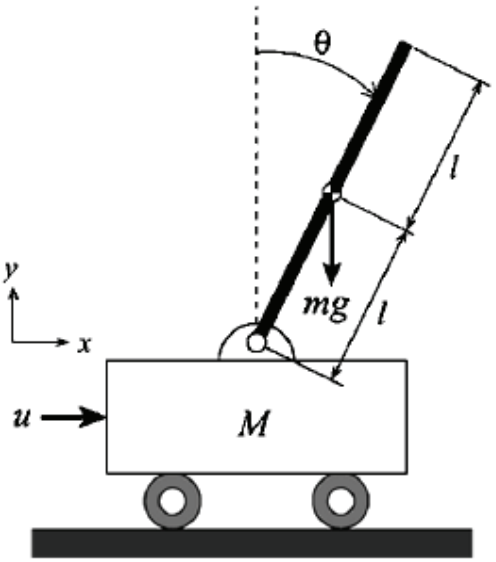

Fig. 1 – *The inverted pendulum system.*

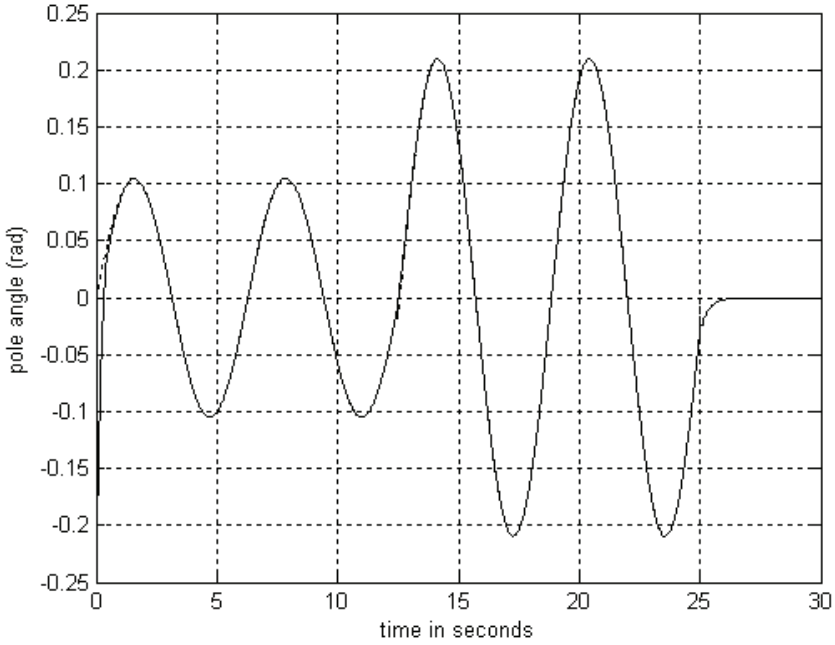

Fig. 2 – *The pendulum angle.*

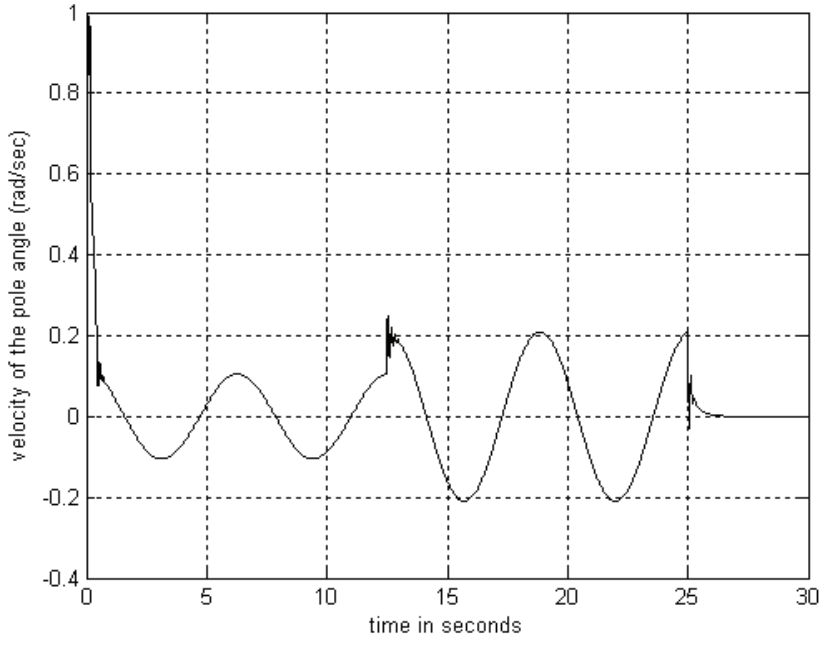

Fig. 3 – *The velocity of the pole.*

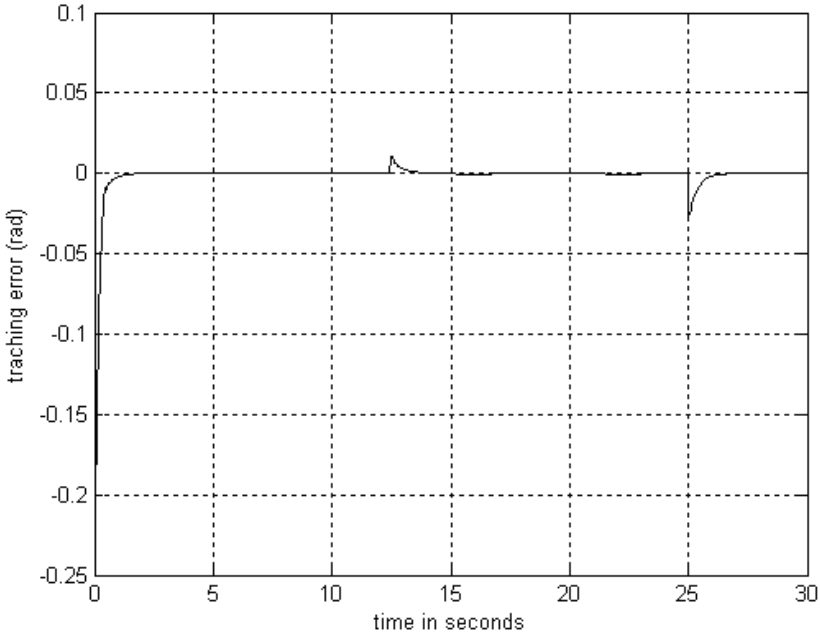

Fig. 4 – *The tracking error* $e = \theta - \theta_m$.

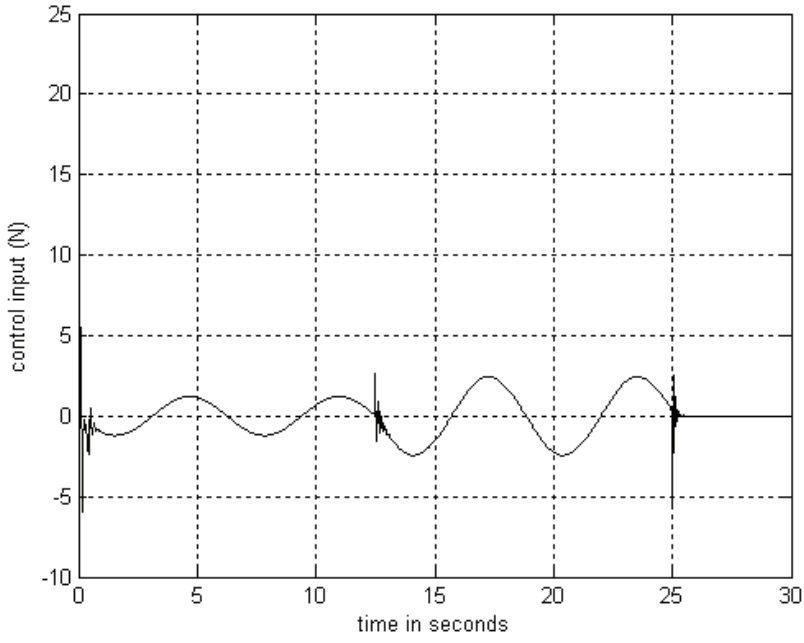

Fig. 5 – *The control input.*

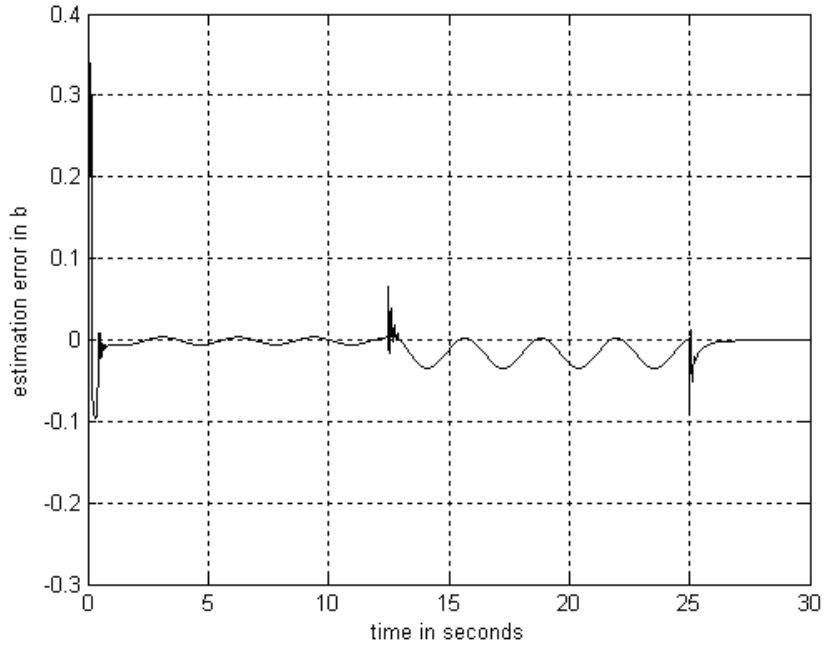

Fig. 6 – *The estimation error when estimating* $b(\underline{x})$ *by the second RBF system.*

Fig. 2 shows the response of the pole angle from the initial position $(-0.2, 0)$. We can see that the system state $x_1(t) = \theta$ tracks the desired trajectory $y_m = \theta_m$ very well. Fig. 3 shows the velocity of the pole. Fig. 4. shows that the tracking error is converging very rapidly to a value close to zero. Fig. 5 represents the corresponding control input which peaks at $t = 12.5\,\text{s}$ (because of the first amplitude variation from $AM = \pi/30$ to $AM = \pi/15$), and at $t = 25\,\text{s}$ (because of the second amplitude variation from $AM = \pi/15$ to $AM = 0$). Fig. 6 shows that the estimation error (between the real value of the virtual control gain $b(\underline{x})$ in the pendulum system and the one provided by the second RBF network estimator $b(\underline{x}, \underline{\theta}_b)$) is smooth, confirming the smoothing property of radial basis function network (RBF) systems. It also remains bounded and converges quite rapidly to a value close to zero. From these figures, we can see that the controlled system behaves well in tracking and regulation. As a comparison concerning this last case, our controller behaves better than the controller in [1, 3].

5 Conclusion

In this paper, we introduced an adaptive controller for a class of unknown nonlinear systems. It is based on two RBF networks: the first network approximates the ideal control law which cannot be implemented because the system dynamics are unknown, the second network estimates the unknown virtual control gain. The centres of the basis functions and the connection weights in the two RBF networks were adjusted on line. The k-means algorithm was used for the centres adjustment. The connection weights of the two RBF networks are adapted according to a law derived using Lyapunov stability theory. The proposed method guarantees the stability of the resulting closed-loop system in the sense that all signals involved are bounded. Finally, we used the proposed approach to control the inverted pendulum system in both tracking and regulation cases. Simulation results showed the good performances of the algorithm.

6 References

[1] X. Ban, X.Z. Gao, X. Huang, A.V. Vasilakos: Stability Analysis of the Simplest Takagi-Sugeno Fuzzy Control System using Circle Criterion, Information Sciences, Vol. 177, No. 20, Oct. 2007, pp. 4387 – 4409.

[2] S. Chen, S.A. Billings, P.M. Grant: Recursive Hybrid Algorithm for Nonlinear Systems Identification using Radial Basis Function Networks, International Journal of Control, Vol. 55, No. 5, May 1992, pp. 1051 – 1070.

[3] P.C. Chen, C.W. Chen, W.L. Chiang: GA-based Modified Adaptive Fuzzy Sliding Mode Controller for Nonlinear Systems, Expert Systems with Applications, Vol. 36, No. 3, April 2009, pp. 5872 – 5879.

[4] T.P. Chen, H. Chen: Approximation Capability to Functions of Several Variables, Nonlinear Functionals and Operators by Radial Basis Function Neural Networks, IEEE Transactions on Neural Networks, Vol. 6, No. 4, July 1995, pp. 904 – 910.

[5] C. Darken, J. Moody: Fast Adaptive k-means Clustering: Some Empirical Results, International Joint Conference on Neural Networks, San Diego, CA , USA, Vol. 2, June 1990, pp. 233 – 238.

[6] K. Funahashi: On the Approximate Realization of Continuous Mappings by Neural Networks, Neural Networks Vol. 2, No. 3, 1989, pp. 183 – 192.

[7] K. Funahashi, Y. Nakamura: Approximation of Dynamical Systems by Continuous Time Recurrent Neural Networks, Neural Networks, Vol. 6, No. 6, 1993, pp. 801 – 806.

[8] S. Haykin: Neural Networks, A Comprehensive Foundation, Prentice Hall, Delhi, India, 1998.

[9] M. Hojati, S. Gazor: Hybrid Adaptive Fuzzy Identification and Control of Nonlinear Systems, IEEE Transactions on Fuzzy Systems, Vol. 10, No. 2, April 2002, pp. 198 – 210.

[10] K. Hornik, M. Stinchcombe, H. White: Multilayer Feedforward Networks are Universal Approximators, Neural Networks, Vol. 2, No. 5, 1989, pp. 359 – 366.

[11] D.R. Hush, B. Horne: Efficient Algorithms for Function Approximation with Piecewise Linear Sigmoidal Networks, IEEE Transactions on Neural Networks, Vol. 9, No. 6, Nov. 1998, pp. 1129 – 1141.

[12] A. Isidori: Nonlinear Control Systems, Springer, Berlin, Germany, 1995.

[13] R. Marino, P. Tomei: Nonlinear Control Design: Geometric, Adaptive and Robust, Prentice Hall, NJ, USA, 1995.

[14] K.S. Narendra, K. Parthasarathy: Identification and Control of Dynamical Systems using Neural Networks, IEEE Transactions on Neural Networks, Vol. 1, No. 1, March 1990, pp. 4 – 27.

[15] R.M. Sanner, J.E. Slotine: Gaussian Networks for Direct Adaptive Control, IEEE Transactions on Neural Networks, Vol. 3, No. 6, June 1992, pp. 837 – 863.

[16] S.N. Singh, M. Steinberg: Adaptive Control of Feedback Linearizable Nonlinear Systems with Application to Flight Control, Journal of Guidance, Control, and Dynamics, Vol. 19, No. 4, July-Aug. 1996, pp. 871 – 877.

[17] J.E. Slotine, L. Weiping: Applied Nonlinear Control, Prentice Hall, NJ, USA, 1991.

[18] S.B. Stojanovic, D.Lj. Debeljkovic, I. Mladenovic: Simple Exponential Stability Criteria of Linear Discrete Time-Delay Systems, Serbian Journal of Electrical Engineering,Vol. 5, No. 2, Nov. 2008, pp. 191 – 198.

[19] E. Tzirkel-Hancock, F. Fallside: Stable Control of Nonlinear Systems using Neural Networks, International Journal of Robust and Nonlinear Control, Vol. 2, No. 1, May 1992, pp. 63 – 86.

[20] C.H. Wang, H. Liu, T. Lin: Direct Adaptive Fuzzy-neural Control with State Observer and Supervisory Controller for Unknown Nonlinear Dynamical Systems, IEEE Transactions on Fuzzy Systems, Vol. 10, No. 1, Feb. 2002, pp. 39 – 49.

[21] C. Zheru, H. Yan: Image Segmentation using Fuzzy Rules Derived from k-means Clusters, Journal of Electronic Imaging, Vol. 4, No. 2, April 1995, pp. 199 – 206.