ELSEVIER

Neurocomputing 17 (1997) 141–157

NEUROCOMPUTING

Neural-networks-based adaptive control of flexible robotic arms

Jorge I. Arciniegas*, Adel H. Eltimsahy, Krzysztof J. Cios

Department of Electrical Engineering and Computer Science, University of Toledo, 2801 W. Bancroft St., Toledo, OH 43606, USA

Received 4 February 1996; accepted 21 April 1997

Abstract

This paper studies the use of RBF neural networks as key components of adaptive controllers aimed at controlling flexible robotic arms. The RBF networks in the proposed controllers are in charge of approximating the inverse dynamics of the arm. To this end, an efficient control law that exploits the approximation capabilities of RBF networks and the stabilizing properties of a servo loop is developed. A stability analysis for the proposed system is carried out and a series of results are discussed in detail. These results are encouraging as they serve to analyze the learning and adaptation behavior of RBF networks and also show the effectiveness of the proposed system.

Keywords: Robotic manipulators; Adaptive control; Neural networks; Position error

1. Introduction

New trends in automation are continuously challenging researchers and practitioners of robotics into developing techniques that would have a positive impact in real world applications of robots. For example, the demand for increased productivity can be partly met by the use of lighter robots operating at higher speeds and consuming less energy. This might imply a reduction in the stiffness of the manipulator structure, which would result in an increase in robot deflections and poor performance due to the effect of mechanical vibrations in the links. Structural flexibility can play a role for a number of reasons. For example, the flexibility problem

* Corresponding author. E-mail: aeltims@uoft02.utoledo.edu.

PII S0925-2312(97)00037-4

can be inherent in situations like the one mentioned above, requiring manipulators with high speed, precision, and increasing payload-handling capabilities. In other cases, structural flexibility might be unavoidable such as when there is a need for light-weight manipulators, like in space applications. Yet, link flexibility might even be desirable as in the case in which a robotic manipulator interacts with a delicate surface which could be damaged if excessive force is applied to it, or in household applications which call for reduced inertia and safer operation.

A control system aimed at controlling this type of manipulators must be able to effectively cope with the adverse effects of transient vibrations and link deflections due to external loads. The imposed requirements can become even more demanding for novel applications such as those sought in nuclear plants, deep sea, hospitals and space, where advanced sensory information may need to be processed to enable a robot to intelligently maneuver in unstructured environments. This constitutes a challenging control problem whose solution depends largely on progress made in a broad range of related areas such as control theory, artificial intelligence, microelectronics, sensor technology, neural networks, and fuzzy systems, to name just a few.

This paper concentrates on one of those fields, namely, artificial neural networks. Artificial neural networks have been enjoying considerable attention by the research community in the past few years. Such attention has resulted in many practical applications. For instance, artificial neural networks have been applied to the recognition of speech patterns, to the estimation of risks in issuing a loan, to the detection of explosive devices, and to the processing of bank checks. In this paper we will use them as tools to investigate the solution of the control problem of robotic manipulators having flexible links.

In terms of control systems theory, the use of artificial neural networks results in systems which are categorized by some as "intelligent controllers" [17]. These systems are called intelligent because they are capable of dealing with a degree of uncertainty which might be substantially greater than that which is tolerated by, say, traditional techniques of adaptive control. This is not to say that artificial neural networks are the sole contributors to a controller having some degree of intelligence. Other approaches such as the use of expert systems and fuzzy logic can also provide a control system with high degrees of adaptivity to a changing environment.

2. Artificial neural networks

The name artificial neural networks is used to refer to a number of systems made up of highly interconnected, simple processing elements referred to as artificial neurons, processing elements, or simply nodes. These types of networks offer an alternative to traditional approaches of computing. In the last few years, they have been extensively studied and have created a great deal of enthusiasm in several research areas which at first do not seem to share many common objectives. Nevertheless, neural networks are being scrutinized by researchers in the fields of psychology, neuroscience, physics, mathematics, electrical engineering, computer science, and biophysics to name some. Whatever the reason for their individual interest in the study of artificial neural

networks, they all seem to be fascinated by the biological basis of this type of systems and the computational capabilities that they possess.

The name artificial neural networks should be taken loosely since these types of networks are far from resembling biological neural systems. They do, however, possess some remarkable properties which evoke comparisons with certain functions of the brain. For instance, artificial neural networks can learn from experience, they can generalize knowledge, and they are capable of abstracting essential information from inputs containing irrelevant data. A key property of artificial neural networks is their ability to generate input–output maps which under mild assumptions can approximate any function to any degree of accuracy. This property has been exploited by a number of researchers to propose controllers and/or control strategies for a variety of applications [1, 7–9, 17, 18].

Several types of artificial neural networks can be found in the literature. Some of them can be applied to a wide variety of problems, while others are targetted for special applications. This paper considers the type of networks known as radial basis functions (RBF) neural networks.

3. Radial basis functions neural networks

Radial basis functions neural networks belong to the class of multilayer feedforward neural networks. They are characterized by a hidden layer made up of a collection of locally tuned processing units. These units can be seen as independent kernel nodes which compute the distance between their corresponding centroid and the input they receive from the input units. The output of these kernel nodes is characterized by a nonlinear, radially symmetric activation function of that distance. Normally, the closer an input is to the center of the receptive field of one of these units, the stronger the response of the unit. The output layer is composed of linear units which are fully connected to the units in the hidden layer. In other words, each output unit performs a weighted sum of the responses it receives from each hidden unit. RBF neural networks are modeled by the following relation:

$$y(\boldsymbol{x}) = \sum_{j=1}^{m} \omega_j g_j(\|\boldsymbol{x} - \boldsymbol{c}_j\|). \tag{1}$$

In this equation, gj corresponds to the jth hidden unit, ω_j is the weight associated with the jth unit, $\boldsymbol{x}$ represents the input vector and $\boldsymbol{c}_j$ is the receptive-field center of the jth unit.

Radial basis functions were first brought into the neural networks literature by Broomhead and Lowe [3] who used them in the prediction of chaotic time series. Their work was influenced by previous theoretical developments on multivariable interpolation reported by Powell [22] and Micchelli [14]. Since then, many other researchers have studied the learning ability and representational capacity of RBF neural networks, e.g. see [2, 16, 19].

4. RBF neural-networks-based adaptive control

It can be shown that the dynamic equations describing the behavior of flexible robotic manipulators can be written in the following matrix form:

$$\boldsymbol{f}(t) = \boldsymbol{M}(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + \boldsymbol{C}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) + \boldsymbol{G}(\boldsymbol{q}(t)), \tag{2}$$

where $\boldsymbol{f}(t)$ is the vector of generalized (non-conservative) forces, $\boldsymbol{q}(t)$ the vector of generalized coordinates, $\boldsymbol{M}(\boldsymbol{q}(t))$ the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))$ the Coriolis and centrifugal force vector, and $\boldsymbol{G}(\boldsymbol{q}(t))$ the vector of potential energy terms representing the contributions to the generalized forces from the conservative forces acting on the system.

Formally speaking, Eq. (2) represents the inverse dynamics of the arm: it is a mapping from link variables to input variables. The direct dynamics can easily be obtained from Eq. (2) considering that matrix $\boldsymbol{M}$ is always invertible. Therefore,

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{f} - \boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{G}, \tag{3}$$

$$\ddot{\boldsymbol{q}} = [\boldsymbol{D}](\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{f}). \tag{4}$$

Eq. (4) represents the direct dynamics of the arm. $[\boldsymbol{D}]$ represents a nonlinear transformation or a mapping from input variables to link variables. From this point of view, the inverse dynamics can be represented by the following relation:

$$\boldsymbol{f} = [\boldsymbol{D}]^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{5}$$

To emphasize this, note that $[\boldsymbol{D}]^{-1}$ denotes an inverse mapping, i.e. an inverse transformation of the direct dynamics. In this sense, $[\boldsymbol{D}]^{-1}$ represents a nonlinear function (transformation) from link variables to the input variables. For convenience, let us make use of the following state-variable notation:

$$\boldsymbol{x}_1 = \boldsymbol{q}, \quad \boldsymbol{x}_2 = \dot{\boldsymbol{q}}, \quad \boldsymbol{u} = \boldsymbol{f}. \tag{6}$$

Then, $\boldsymbol{u}$ can be expressed in the following way:

$$\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{x}_1)\dot{\boldsymbol{x}}_2 + \boldsymbol{C}(\boldsymbol{x}_1, \boldsymbol{x}_2) + \boldsymbol{G}, \tag{7}$$

$$\boldsymbol{u} = [\boldsymbol{D}]^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2, \dot{\boldsymbol{x}}_2), \tag{8}$$

where $\dot{\boldsymbol{x}}_2$ denotes the response of the system under the influence of the input $\boldsymbol{u}$.

5. Controller design

Let $\dot{\boldsymbol{x}}_{2d}$ denote the desired response of the system, i.e. the desired acceleration. Thus, a control signal can be formed by using the arm's inverse dynamics.

$$\boldsymbol{u} = [\boldsymbol{D}]^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2, \dot{\boldsymbol{x}}_{2d}), \tag{9}$$

$$\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{x}_1)\dot{\boldsymbol{x}}_{2d} + \boldsymbol{C}(\boldsymbol{x}_1, \boldsymbol{x}_2) + \boldsymbol{G}. \tag{10}$$

Then, equating Eqs. (7) and (10), it follows that

$$\boldsymbol{M}(\boldsymbol{x}_1)\,[\dot{\boldsymbol{x}}_{2d} - \dot{\boldsymbol{x}}_2] = 0. \tag{11}$$

This leads to the following equality:

$$\dot{\boldsymbol{x}}_2 = \dot{\boldsymbol{x}}_{2d}. \tag{12}$$

Hence, the response of the system satisfies the desired performance. The important aspect to highlight here is that this error equation is a satisfied only if the inverse dynamics of the system are known *precisely*.

In this paper, we propose the use of RBF neural networks to *approximate* the inverse mapping $[\boldsymbol{D}]^{-1}$ as closely as possible. From what has been said so far, let us use the following control law:

$$\boldsymbol{u} = [\hat{\boldsymbol{D}}]^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}_d) + \boldsymbol{K}_v\dot{\boldsymbol{e}} + \boldsymbol{K}_p\boldsymbol{e}, \tag{13}$$

where $[\hat{\boldsymbol{D}}]^{-1}$ is the neural network approximation of the actual inverse dynamics of the system. The last two terms on the right-hand side represent a servo feedback which is introduced to stabilize the system. $\boldsymbol{K}_v$ and $\boldsymbol{K}_p$ are constant gain matrices and $\boldsymbol{e} = \boldsymbol{x}_{1d} - \boldsymbol{x}_1$ represents the tracking error.

Now, making use of Eqs. (13) and (5), it can be shown that

$$\boldsymbol{K}_v\dot{\boldsymbol{e}} + \boldsymbol{K}_p\boldsymbol{e} = [\tilde{\boldsymbol{D}}]^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{14}$$

Eq. (14) characterizes a linear decoupled system driven by the nonlinear vector function $[\tilde{\boldsymbol{D}}]^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$. This function represents the error in the neural network approximation of the manipulator's inverse dynamics.

It makes intuitive sense that instead of using one network to approximate the inverse dynamics of the whole arm, one ought to use a separate network for each joint of the manipulator. With this in mind and by using Eq. (14), the error equation for the ith joint of the manipulator is expressed as follows:

$$k_v\dot{e}_i + k_p e_i = \tilde{d}_i^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{15}$$

In this last equation, k_v and k_p are constant gains and $\tilde{d}_i^{-1}(\cdot)$ the local approximation error of the RBF neural network assigned to the ith joint. Now, letting $k_v\dot{e} + k_p e = \varepsilon_i$, Eq. (15) can be written as follows:

$$\varepsilon_i = \tilde{d}_i^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{16}$$

Noting that $\varepsilon_i = \varepsilon_i(t)$, one concludes that $\varepsilon_i(t)$ constitutes a measure of the tracking error that reflects the mismatch between the actual inverse dynamics of the system and its local neural network approximation.

The question that remains to be answered at this point is how to update, or adjust, the network parameters on-line, so that the error measure $\varepsilon_i(t)$ converges to 0 as $t \to \infty$. A gradient descent approach comes immediately to mind. To do that, let us define the following cost function:

$$J = \tfrac{1}{2}\varepsilon_i^2, \tag{17}$$

J must be minimized over the parameter space of the network. In doing so, let us derive the update law.

$$\dot{\omega}_j = -\alpha \nabla J, \tag{18}$$

$$\dot{\omega}_j = -\alpha \frac{\partial \varepsilon_i}{\partial \omega_j}, \tag{19}$$

$$\dot{\omega}_j = \alpha \varepsilon_i \frac{\partial \hat{d}_i^{-1}}{\partial \omega_j}. \tag{20}$$

Recalling that the output of the network is given by Eq. (1), the update law takes on the following form:

$$\dot{\omega}_j = \alpha \varepsilon_i g_j(\cdot) \tag{21}$$

In this update law, $\alpha > 0$ is the adaptation or learning parameter. The subscript j denotes the jth weight of the RBF network. Therefore, the network's weights are updated according to a simple first-order differential equation involving the parameter α, the current value obtained from a servo feedback loop, and the local response of the corresponding radial unit. It is important to point out that the above development depends on the underlying assumption that RBF networks are capable of approximating the inverse dynamics of the controlled system. It is known that RBF networks are capable of approximating any resonable function. However, as it will be made evident shortly, that the fact by itself is not sufficient for stability. Stability imposes the additional requirement that the error in the approximation of the inverse dynamics has to remain bounded. Satisfying this requirement with carefully designed networks constitutes a nontrivial problem which still needs further research. A particularly good example of research on this subject is the work by Sanner [27] who makes use of principles from sampling theory and Fourier analysis to develop networks that, under mild assumptions, are capable of uniformly approximating smooth functions on specified compact sets. Not surprisingly, it turns out that the condition on the boundedness of the approximation error can be satisfied by carefully choosing the RBF network parameters, so as to guarantee a desired uniform approximation in a target set. This translates into selecting an appropriate variance, when the units in the networks are gaussian units, and precisely placing the network units in the region of interest. For the purposes of this paper, it will be assumed that the RBF networks in the proposed control system have been designed so as to guarantee that the error in the approximation of the inverse dynamics remains bounded.

6. Stability analysis

Consider the following proposition:

Proposition. *Given that the error in the approximation of the inverse dynamics is bounded, all states of the system will also remain bounded.*

Proof. Realizing that Eq. (14) represents an asymptotically stable linear system driven by the nonlinearity $[\tilde{\boldsymbol{D}}]^{-1}$, the above proposition can be proven using the direct method of Lyapunov. To do that, let us use the following scalar Lyapunov function:

$$V(\boldsymbol{e}) = \tfrac{1}{2}\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e}. \tag{22}$$

The system itself is represented by the expression shown below.

$$\boldsymbol{K}_v\dot{\boldsymbol{e}} + \boldsymbol{K}_p\boldsymbol{e} = [\tilde{\boldsymbol{D}}]^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}), \tag{23}$$

$$\dot{\boldsymbol{e}} + \boldsymbol{K}_v^{-1}\boldsymbol{K}_p\boldsymbol{e} = \boldsymbol{K}_v^{-1}[\tilde{\boldsymbol{D}}]^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{24}$$

Obviously, the function of Eq. (22) is positive definite. This satisfies the first condition of the Lyapunov theorem. To satisfy the second condition we must determine the circumstances under which $V(\boldsymbol{e})$ is monotonically decreasing:

$$\dot{V}(\boldsymbol{e}) = \boldsymbol{e}^{\mathrm{T}}\dot{\boldsymbol{e}}, \tag{25}$$

$$\dot{V}(\boldsymbol{e}) = \boldsymbol{e}^{\mathrm{T}}[\boldsymbol{K}_v^{-1}[\tilde{\boldsymbol{D}}]^{-1}(\cdot) - \boldsymbol{K}_v^{-1}\boldsymbol{K}_p\boldsymbol{e}], \tag{26}$$

$\dot{V}(\boldsymbol{e})$ must satisfy the condition that $\dot{V}(\boldsymbol{e}) \leq 0$. Hence,

$$\boldsymbol{e}^{\mathrm{T}}\boldsymbol{K}_v^{-1}[\tilde{\boldsymbol{D}}]^{-1}(\cdot) - \boldsymbol{e}^{\mathrm{T}}\boldsymbol{K}_v^{-1}\boldsymbol{K}_p\boldsymbol{e} \leq 0, \tag{27}$$

$$\|\boldsymbol{K}_v^{-1}\|\,\|[\tilde{\boldsymbol{D}}]^{-1}(\cdot)\| \geq \|\boldsymbol{K}_v^{-1}\|\,\|\boldsymbol{K}_p\|\,\|\boldsymbol{e}\|. \tag{28}$$

But, $\|[\tilde{\boldsymbol{D}}]^{-1}\| \leq \xi$; therefore,

$$\|\boldsymbol{e}\| \leq \frac{\xi}{\lambda_p}. \tag{29}$$

In this last expression, λ_p represents the largest eigenvalue of $\boldsymbol{K}_p$. Furthermore, from Eq. (24) and using Eq. (31) we can determine the bounds on $\dot{\boldsymbol{e}}$. This is shown below.

$$\|\dot{\boldsymbol{e}}\| \leq \|\boldsymbol{K}_v^{-1}\|\,\|[\tilde{\boldsymbol{D}}]^{-1}(\cdot)\| + \|-\boldsymbol{K}_v^{-1}\|\,\|\boldsymbol{K}_p\|\,\|\boldsymbol{e}\|, \tag{30}$$

$$\|\dot{\boldsymbol{e}}\| \leq 2\lambda_v\xi, \tag{31}$$

where λ_v represents the maximum eigenvalue of $\boldsymbol{K}_v^{-1}$.

7. Results and discussion

We begin describing a series of tests designed to study the learning and adaptive behavior of RBF neural networks and their effectiveness in controlling flexible robotic arms. These tests were performed on models of one-link and two-link flexible manipulators. We will focus our attention on the tracking accuracy of the manipulator and the amount of deflection experienced by the links.

Fig. 1 shows a block diagram representation of the overall system. Note that, as explained before, the effective control signal driving the manipulator is made up of two components: one due to RBF neural networks and the other due to the PD stabilizer. We begin testing this system with a test designed to study the learning and

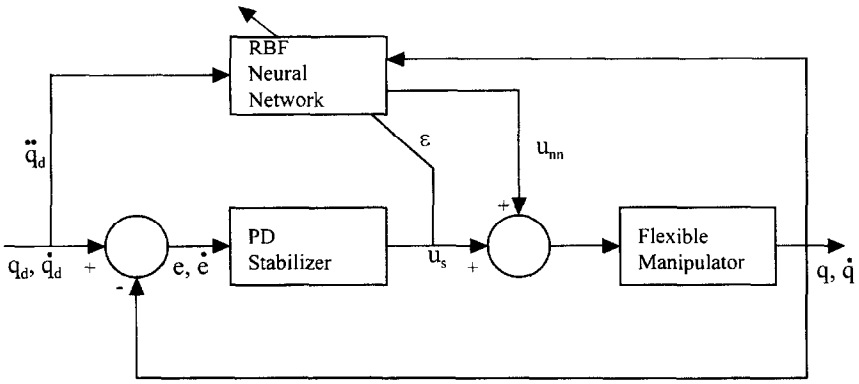


Fig. 1. Control system diagram.

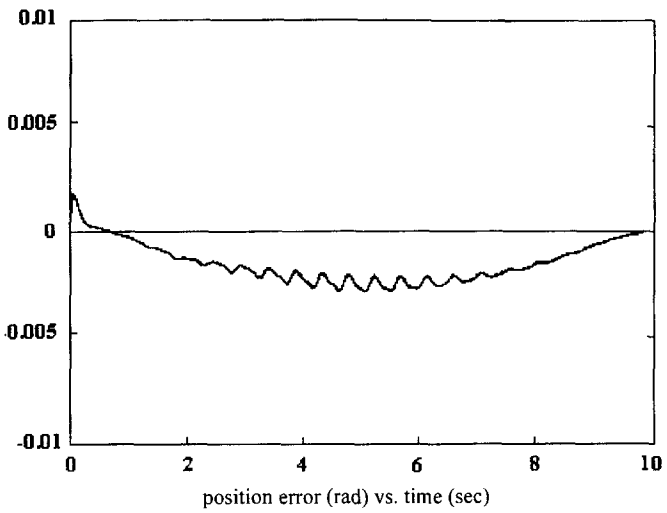


Fig. 2. Position error (case 1).

adaptation capacity of RBF networks. Even though it would be pleasing to have the system behave reasonably well under all conditions, in this test we are not particularly interested in the overall performance of the system, but rather in the learning ability of its RBF networks. We do this by using a model of a single-link flexible arm whose flexibility has been intentionally increased. In other words, we try to control a flexible link with a very large length-to-height ratio. In the figures that follow this discussion, we refer to this test as "case 1".

As depicted by Fig. 2, we observe that the system exhibits good performance in terms of position tracking. However, as mentioned above, we wish to know how the

system performed in terms of *learning*. This question can be answered by looking at the control signal components and by observing the amount of control authority exercised by the neural network. To clarify this point, it is important to point out that the RBF networks in the system are assumed to have no initial knowledge at the beginning of a control trial. This is equivalent to saying that all the weights in the network are initially set to zero. This differs with the traditional alternative in which the network weights are initialized to random numbers in some preset interval. The choice of initializing all network weights to zero makes intuitive sense, since it is known that the network does not know anything about the dynamics of the controlled system until it gets exposed to it. Therefore, it is expected that initially, the control signal be derived entirely from the servo loop, and that the amount of control authority will be transferred progressively to the neural network as learning takes place. As verified by the results portrayed by the next three figures, this is indeed what takes place. Fig. 3 shows the control component due to the RBF neural network in the system, while Fig. 4 depicts the component of the control signal due to the PD stabilizer. It is seen that the latter signal is very strong at the beginning of the trial and that it gets substantially diminished as the time passes. It is interesting to point out the fact that this component of the control signal decreases rather rapidly, and that it remains low throughout the duration of the test. In contrast, the RBF NN component starts out at zero and it increases very rapidly. Furthermore, it is also observed that this component stays active until the end of the trial.

The previous test speaks well of the learning behavior of the system RBF networks under the proposed control law. We now discuss a test in which we look at the performance of the system in more detail. In the figures that follow we refer to this test as "case 2". For this test we consider a two-link flexible manipulator. In this case, each

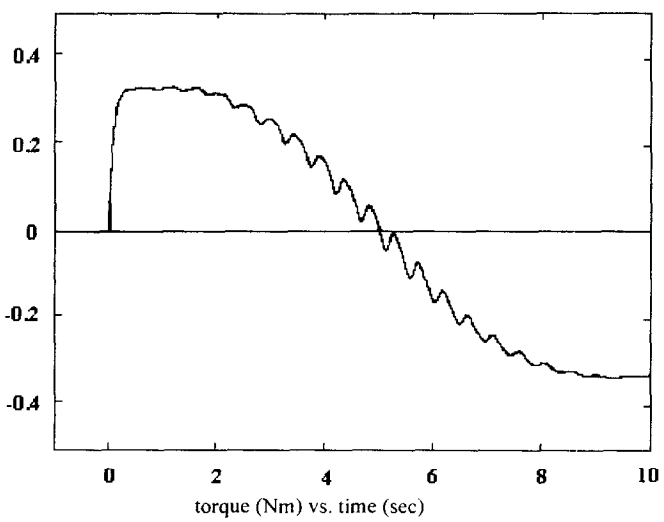


Fig. 3. RBF NN control component (case 1).

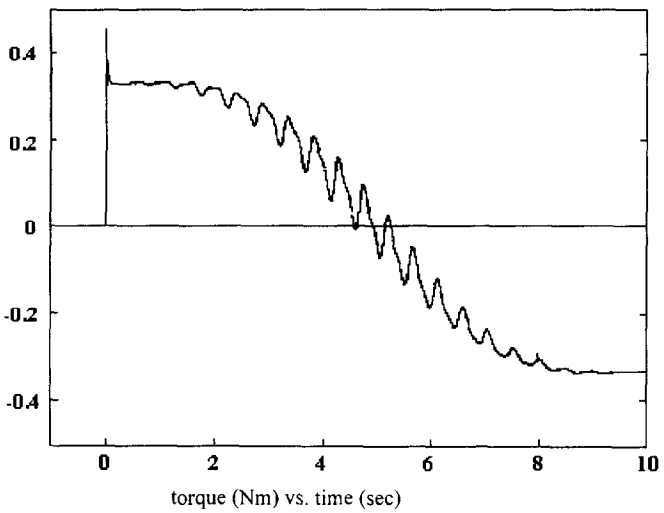


Fig. 4. Effective control signal (case 1).

link of the manipulator is controlled by a separate controller, each with its own RBF network. For brevity, we will concentrate our attention on the performance of the manipulator's distal link and the arm's tip. The RBF network selected for this controller consisted of 675 processing units uniformly distributed in the input space. The response of the radial units was governed by a Gaussian nonlinearity with a variance equal to 0.6. The learning parameter was set to 25 for this example. The servo gains were set to 5.0 and 0.5 for the proportional and derivative gains, respectively. The sampling and adaptation rates were chosen equal to 5 ms.

With this setup, the system exhibited satisfactory performance in terms of joint position, velocity and acceleration. The next three figures help us to qualify the last sentence. Fig. 5 contains a plot of the error in the joint position of the flexible, distal link. In addition, Fig. 6 shows the joint velocity error, while Fig. 7 contains a plot of the joint acceleration error. It is observed that these errors remain bounded throughout the test and, more importantly, they are small. In fact, the maximum absolute error in the joint position of the flexible link is 0.0035 rad. The maximum absolute values of the joint velocity and acceleration errors remain below 0.003 and 0.004 rad/s^2, respectively.

It is important to emphasize the fact that the above results are encouraging as the system behaves well in terms of tracking. It is worth mentioning here that the control action exerted by the system resulted in smooth acceleration profiles. This is important when dealing with flexible links since it will undoubtedly decrease the amount of vibrations experienced by the manipulator. For the case considered here, the deflection experienced by the tip of the manipulator is portrayed in Fig. 8. It is seen that the deflection of the manipulator's tip is indeed quite low throughout the duration of the test. This should reflect positively on the positioning accuracy of the manipulator.

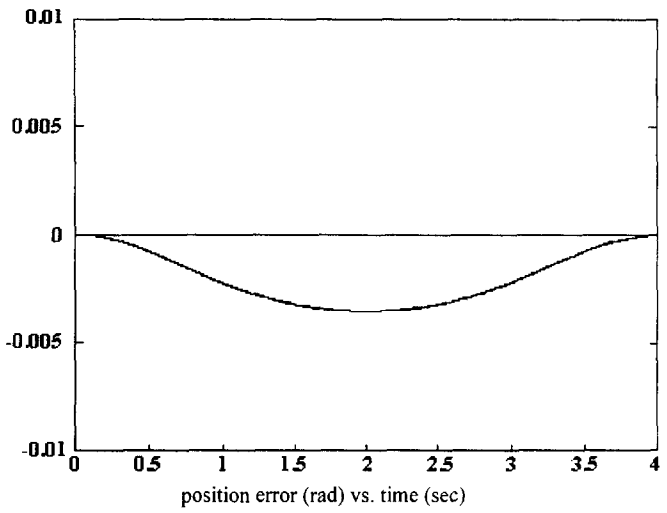


Fig. 5. Position error (case 2).

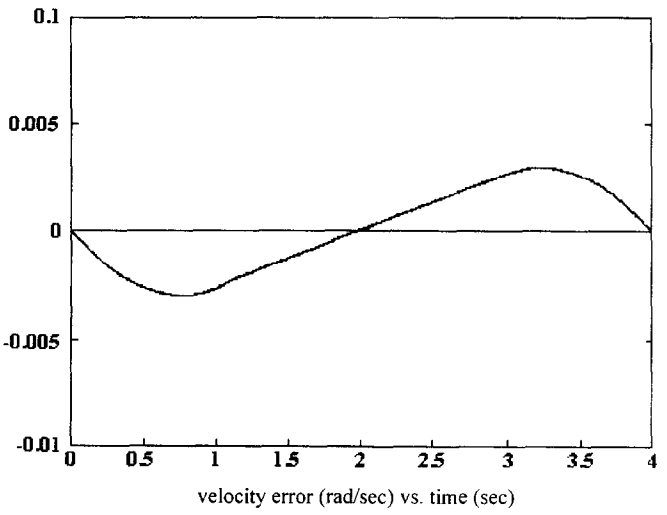


Fig. 6. Velocity error (case 2).

Finally, as we did in the previous test, we examine the control signal produced by the controller, and pay particular attention to its two components. The control components due to the RBF network and servo loop are depicted in Figs. 9 and 10, respectively. Fig. 11 contains a plot of the effective control signal driving the flexible link of the manipulator. It is seen that, as expected, a learning process similar to the

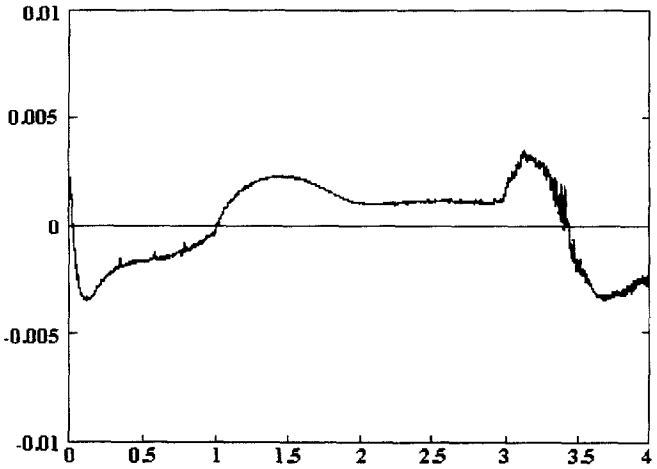


Fig. 7. Acceleration error (case 2).

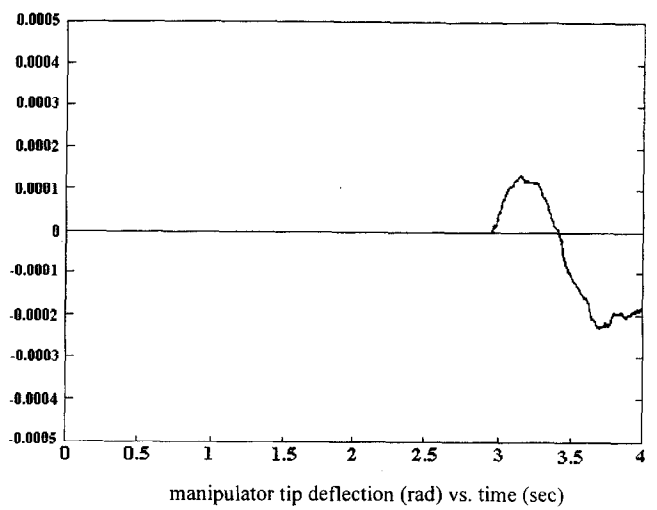


Fig. 8. Tip deflection (case 2).

one seen before has also taken place in this case, i.e. while the RBF network learns to approximate the required function, most of the control authority resides in the stabilizing servo component of the controller. As the RBF networks learns more about the dynamics of the manipulator, as switching effect takes place in the sense that the control authority gets transferred from the servo component to the neural

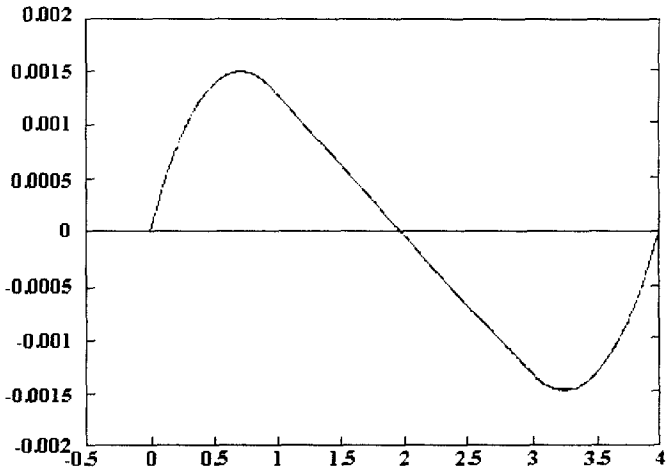


Fig. 9. NN control component (case 2).

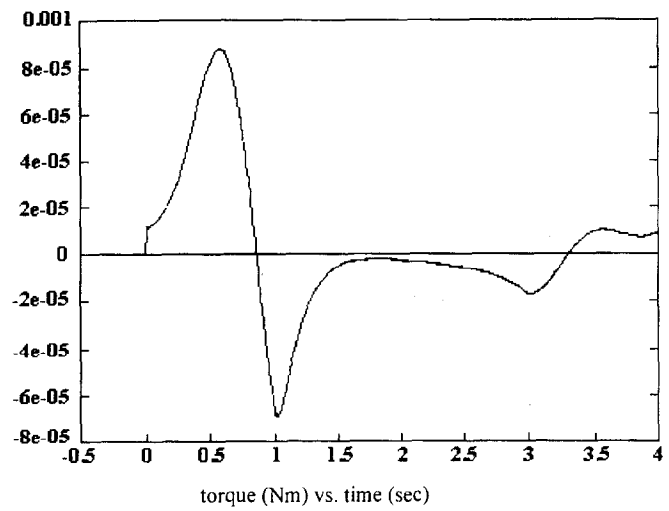


Fig. 10. PD stabilizer control component (case 2).

network. This is clearly observed in the corresponding figures which show that, as the test progresses, the net effect of the control component generated by the RBF network is more significant than the component generated by its servo counterpart.

The above comments speak very well of the learning and adaptation capabilities of RBF networks to control the kind of systems that we are dealing with, i.e. flexible

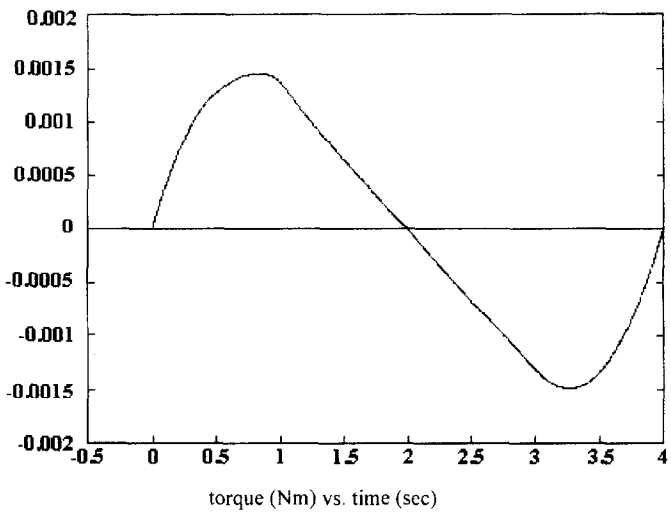


Fig. 11. Effective control signal (case 2).

robotic manipulators. In addition, we ought to also point out the importance of the servo component of the proposed control system. One can theorize that ideally, the gradual shifting of control authority from the servo component to the RBF component of the controller would take place until the latter component assumes full control of the system. In other words, in ideal conditions, the overall system would undergo a gradual transformation from being a closed-loop system to become an open-loop system. That would happen because, as the RBF network gets trained on-line, the contribution of the stabilizing servo loop to the overall control signal would diminish slowly, until it would actually disappear. The above behavior, however, is an ideal behavior on the part of the system. In reality, the contribution of the servo loop to the control signal delivered by the controller, never dies down completely. It does decrease as time passes, but it is always present, even if its net effect is substantially smaller than that of the RBF network. In fact, in spite of its diminishing contribution to the overall control signal, the servo loop portion of the system is very important for two reasons. First, it is essential to the adaptive behavior of the RBF network in the system. Second, as evidenced by the above tests, the servo loop has the important effect of stabilizing the system.

8. Conclusions

This paper has studied the problem of using RBF neural networks to perform adaptive control of flexible robotic manipulators. The paper has shown how RBF networks can be incorporated as key components of adaptive controllers. The role of RBF networks in these controllers is the approximation of the inverse dynamics of the

arm. This lead to the development of an efficient control law that exploits the approximating capabilities of RBF networks, as well as the stabilizing properties of a servo loop. A stability analysis was also carried out for the proposed system.

The results obtained from using the RBF neural-network-based adaptive controller were encouraging. Two tests were presented, and they showed that using the proposed controller one can effectively control the kind of systems that we are dealing with. These tests also served to analyze the learning and adaptation behavior of the RBF neural networks in the proposed control system. It was shown that a switching phenomenon takes place between the servo portion and the neural-network portion of the controller. However, it was also shown that in spite of becoming very small, the servo component is vital in the system and remains operational throughout a control trial.

Acknowledgements

The authors would like to thank the anonymous reviewers whose comments helped greatly in improving both the style and the substance of this paper.

References

[1] A. Barto, Connectionist learning for control, in: T. Miller, R. Sutton, P. Werbos (Eds.), Neural Networks for Control, MIT Press, Cambridge, MA, 1990, pp. 1–58.

[2] S. Botros, C. Atkeson, Generalization properties of radial basis functions, in: R. Lippman, J. Moody, D. Touretzky (Eds.), Neural Information Processing, vol. 3, Morgan Kaufmann, Los Altos, CA, 1991, pp. 707–713.

[3] D. Broomhead, D. Lowe, Multivariable interpolation and adaptive networks, Complex Systems 2, (1988) 321–355.

[4] F.-C. Chen, H.K. Khalil, Adaptive control of nonlinear systems using neural networks, Int. J. Control, 55 (6) (1992) 1299–1317.

[5] F.-C. Chen, C.-C. Liu, Adaptively controlling nonlinear continuous-time systems using neural networks, IEEE Trans. Automat. Control 39 (6) (1994) 1306–1310.

[6] X. Cui, K.G. Shin, Direct control and co-ordination using neural networks, IEEE Trans. Systems Man Cybernet. 23 (3) (1993).

[7] R. Hampo, K. Marko, Neural network architectures for active suspension control, Proc. Int. Joint Conf. Neural Networks 2 (1991) 765–770.

[8] M. Jordan, D. Rumelhart, Forward models: supervised learning with a distal teacher, Tech. Report MIT Center for Cognitive Science, Occasional Paper No. 40, 1989.

[9] M. Kawato, Computational schemes and neural network models for formation and control of multijoint arm trajectory, in: T. Miller, R. Sutton, P. Werbos (Eds.), Neural Networks for Control, MIT Press, Cambridge, MA, 1990.

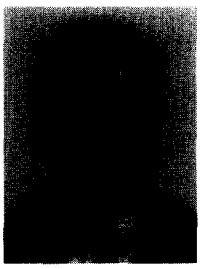
[10] F.L. Lewis, K. Liu, A. Ye, Neural net robot controller with guaranteed tracking performance, Proc. IEEE Int. Symp. Intelligent Control, August 1993, pp. 225–231.

[11] F.L. Lewis, K. Liu, A. Ye, Neural net robot controller: structure and stability proofs, Proc. IEEE Conf. Decision and Control, San Antonio, December 1993, pp. 2785–2791.

[12] F.L. Lewis, K. Liu, A. Ye, Neural net robot controller with guaranteed tracking performance, IEEE Trans. Neural Networks 6 (3) (1995) 703.

[13] C.-C. Liu, F.-C. Chen, Adaptive control of non-linear continuous-time systems using neural networks: general relative degree and MIMO cases, Int. J. Control 58 (2) (1993) 317–335.

[14] C. Micchelli, Interpolation of scattered data: distance matrices and conditionally positive definite functions, Constr. Approx. 2 (1986) 11–22.
[15] W.T. Miller, R.S. Sutton, P.J. Werbos, Neural Networks for Control, MIT Press, Cambridge, MA, 1991.
[16] J. Moody, C. Darken, Fast learning in networks of locally-tuned processing units, Neural Comput. 1 (1989) 281–294.
[17] K. Narendra, S. Mukhopadhyay, Intelligent control using neural networks, IEEE Control Systems Magn. (1992) 11–18.
[18] D. Nguyen, B. Widrow, Neural networks for self-learning control systems, Int. J. Control 54 (6) (1991) 1439–1451.
[19] T. Poggio, F. Girosi, Regularization algorithms for learning that are equivalent to multilayer networks, Science 247 (1990) 978–982.
[20] M.M. Polycarpou, P.A. Ioannou, Idenfication and control using neural models: design and stability anaysis, Tech. Report 91-09-01. Dept. Elect. Eng. Sys., University of South Carolina, September 1991.
[21] M.M. Polycarpou, P.A. Ioannou, Neural networks as on-line approximator of nonlinear systems, IEEE Conf. Decision and Control, Tucson, December 1992, pp. 7–12.
[22] M. Powell, Radial basis functions for multivariable interpolation: a review, in: J. Mason, M. Cox (Eds.), Algorithms for Approximation, Clarendon Press, Oxford, 1987.
[23] G.A. Rovithakis, M.A. Christodoulou, Adaptive control of unknown plants using dynamical neural networks, IEEE Trans. Systems Man Cybernet., 1994, to appear.
[24] N. Sadegh, Nonlinear identification and control via neural networks, Control Systems with Inexact Dynamics Models, DSC-vol. 33, ASME Winter Annual Meeting, 1991.
[25] N. Sadegh, A perceptron network for functional identification and control of nonlinear systems, IEEE Trans. Neural Networks 4 (6) (1993) 982–988.
[26] R.M. Sanner, J.-J. E. Slotine, Stable adaptive control and recursive identification using radial gaussian networks, Proc. IEEE Conf. Decision and Control, Brighton, 1991.
[27] R. Sanner, J. Slotine, Gaussian networks for direct adaptive control, IEEE Trans. Neural Networks 3 (4) (1992) 837–863.
[28] D.A. White, D.A. Sofge, Handbook of Intelligent Control, Van Nostrand Reinhold, New York, 1992.
[29] M. Vandergrift, F.L. Lewis, S. Zhu, Flexible-link robot arm control by a feedback linearization/singular perturbation approach, J. Robotic Systems 11 (7) (1994) 591–603.
[30] A. Ye, F.L. Lewis, A neural network controller for feedback linearization, Proc. IEEE Conf. Decision and Control, December 1994, pp. 2494–2499.
[31] A. Ye, M. Vandergrift, F.L. Lewis, A neural network controller for flexible-link robots, Proc. IEEE Int. Symp. Intelligent Control, 1994, pp. 63–68.



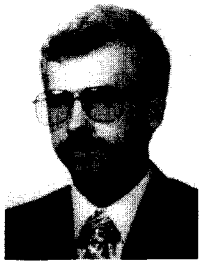
Jorge Arciniegas received the Ph.D. degree from the University of Toledo, OH in 1994. He served as an assistant professor in the Departments of Computer Science and Electrical Engineering at the Monterrey Institute of Technology in Mexico. He has also been with the Control Strategies Engineering Group of ITT Automotive where he participated in the development of an advanced vehicle stability control system. Currently, he is a senior engineer with TRW Automotive Electronics Group. His interests lie in the areas of intelligent control, adaptive control, software development, software quality and reliability, and embedded control systems.

Dr. Arciniegas is a member of the Institute of Electrical and Electronics Engineers and a member of Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi.

Dr. Eltimsahy received the M.S. and Ph.D. degrees in Electrical Engineering from the University of Michigan in 1961 and 1967, respectively. He also received the B.S. degree in Electrical Engineering from Cairo University in 1958. He joined the EE Department of the University of Toledo in 1968 as an assistant professor where he is presently a professor. During 1958, 1959 and 1962, he worked as a researcher at the National Research Center in Cairo and also as an adjunct instructor at Cairo University. During 1967–1968, he also worked as an assistant professor of electrical engineering at the University of Tennessee.

Dr. Eltimsahy's current research interests are directed towards the identification and control of flexible manipulators using tools from the fields of artificial neural networks and fuzzy systems. Some of his current research projects are (1) radial basis functions neural networks, fuzzy inference systems and their application in off-line system identification and adaptive control of flexible robotic manipulators, (2) adaptive control of two flexible robot manipulators coordinating an object, (3) stability of flexible manipulators and (4) stability of flexible manipulators utilizing fuzzy controllers.

Dr. Eltimsahy is a senior member of IEEE, a member of Eta Kappa Nu and Sigma Xi.



Krzysztof J. Cios received the M.S. degree in electrical engineering and a Ph.D. in computer science, both from AGH Technical University, Krakow, and an MBA degree from University of Toledo. He is a Professor of Bioengineering and of Electrical Engineering and Computer Science, Department of Bioengineering, University of Toledo, Toledo, OH, USA. His research interests are in the area of intelligent systems and knowledge discovery and data mining. His research was funded among others by NASA, National Science Foundation, NATO, and the American Heart Association. He has published extensively in journals, conference proceedings, and wrote several book chapters. His paper on inductive machine learning received The Norbert Wiener 1995 Outstanding Paper Award from *Kybernetes*. Dr. Cios consults for several companies and gives tutorials in his areas of expertise. He is a senior member of the IEEE, and serves on editorial boards of *Neurocomputing* and the Handbook of Neural Computation.