Adaptive Neural Control of a Rotating Flexible Manipulator

A. R. Maouche*, and M. Attari**

*,** Laboratory of Instrumentation, Faculty of Electronic and Computer Science, Houari Boumedien University, Algiers, Algeria

* armaoouche@umbb.dz, ** attari.mo@gmail.com

Abstract—The motion control of a planar manipulator with a flexible arm is studied. Dynamics are developed in Lagrange’s formulation. A novel control system structure is proposed to control the joint position and velocity as well as deflection of the tip of the arm. First, a neural network controller based on the equations of motion of the robot is elaborated. Its aim is to produce a fast and stable control. Then, an adaptive neural controller is added to compensate structured and unstructured uncertainties. Efficiency of the new controller obtained by combining the two control laws is tested facing an important variation of the dynamic parameters of the flexible manipulator and compared to a classical nonlinear controller. Simulation results show the effectiveness of the control strategy proposed.

Index Terms—Adaptive neural control, nonlinear control, flexible manipulator.

I. INTRODUCTION

This article presents a novel control system structure composed of two neural controllers. The first neural controller uses feed forward neural networks to approximate the dynamic model of the robot. The use of knowledge-based modeling, whereby mathematical equations are derived in order to describe a process, based on a physical analysis, is important to elaborate effective controllers. However this can lead to a complex controller if the model of the system to be controlled is huge and its computation can be time consuming. We propose then, to construct a controller based on artificial neural network (ANN) that approximates the dynamic model of the robot. The use of ANN in the place of the nonlinear model permits to simplify the structure of the controller reducing its computation time and enhancing its reactivity, without a loss in the precision of the tracking control. This is important when real time control is needed. The main feature that makes ANN ideal technology for controller systems is that they are nonlinear regression algorithms that can model high-dimensional systems and have the extreme flexibility due to their learning ability.

We propose to add to the first controller an adaptive neural controller trained online, to compensate for errors due to structured and unstructured uncertainty, increasing the precision of the overall controller [1].

The demand for increased productivity in Industry has led to the use of lighter robots with faster response and lower energy consumption. Flexible manipulator systems have relatively smaller actuators, higher payload to weight ratio and, in general, less overall cost. The drawbacks are a reduction in the stiffness of the manipulator structure which results in an increase in robot deflection and poor performance due to the effect of mechanical vibration in the links.

The control of non-rigid link manipulators motion has attracted attention of researchers for almost three decades. However, most of the control techniques for flexible manipulators are inspired by classical controls [2]-[7]. On other hand, much research effort has been put into the design of neural network application for robotic control [8]-[10]. With recent developments in sensor/actuator technologies, many researchers have concentrated on control methods for suppressing vibrations of flexible structures using smart materials such as Shape Memory Alloys (SMA) [11], Magnetorheological (MR) materials, Electrorheological (ER) materials, Piezoelectric transducers (PZT) [12] etc.

The control law presented has several distinguished advantages. It is easy to compute since it is based on ANN. This robust control design method maximizes the control performance guaranteeing good precision when regulating the tip position of the flexible manipulator in the presence of large time-varying payload and parameter uncertainties.

A flexible manipulator is a system with distributed parameters and is governed by complex equations with partial derivative. A dynamic model of such a system, to be used in control design, is by nature an approximate model. Thus, the modeling error introduced by this approximation influences the performances of the control. Choosing an adaptive control, allows dealing with modeling errors and makes it possible to compensate, until a certain level, physical phenomena such as friction, which representation is difficult to achieve.

The reminder of this paper is organized as follows. In Section 2, a planar flexible link manipulator is modeled according to Euler-Lagrange’s formulation and the finite element method for the discretisation. Section 3 presents the nonlinear controller. Stability and robustness analysis of this control method is carried out which ends in the development of the adaptive neural controller presented in Section 4. Section 5 describes simulation results. Tests have been carried out for the adaptive neural controller and compared to those obtained with the conventional nonlinear controller. Results show the efficiency of the control strategy proposed facing an important variation of the dynamic parameters. Finally, conclusion is given in Section 6.
II. DYNAMIC MODELING

The system considered here consists of a one link flexible manipulator constrained to move in the horizontal plane. The link is composed of a flexible beam cantilevered onto a rigid rotating joint.

The manipulator is actuated by a DC motor with a torque input at the hub as shown in Fig. 1. It is assumed that the link can be bent freely in the horizontal plane but is stiff in the vertical bending and torsion (no gravity effect). Thus, Euler-Bernoulli beam theory is sufficient to develop the robot arm dynamics.

As shown in Fig. 1, \( \{O_0 \hat{x}_0 \hat{y}_0 \} \) represents the stationary frame, \( \{O_1 \hat{x}_1 \hat{y}_1 \} \) is the moving coordinate frame with origin at the hub of the link. \( \theta \) is the revolving angle at the hub of the link. \( f \) and \( \alpha \) are the elastic displacements, they describe the deflection and the section rotation at the tip of the link.

Motion of the manipulator’s arm is described by one rigid and two elastic variables:

\[
\mathbf{q} = [\mathbf{q}_r \mathbf{q}_e]^T
\]

where \( \mathbf{q}_r = [\theta]^T \) and \( \mathbf{q}_e = [f \; \alpha]^T \).

Torque applied to the manipulator’s joint is given by:

\[
\Gamma = [\Gamma]
\]

(2)

Let us consider an arbitrary point \( M \) on the link. The kinetic energy of the link is then, given by:

\[
T = \frac{1}{2} \rho \int S \{V(M)\}^2 \, ds \, dx
\]

(3)

where, \( V(M) \) is the velocity of \( M \) on the flexible link. \( L, S \) and \( \rho \) are length, section and mass density of the link, respectively.

Now, the total kinetic energy \( T \) can be written as [13]:

\[
T = \frac{1}{2} J_A \dot{\theta}^2 + \frac{1}{2} J_B (\dot{\theta} + \dot{\alpha})^2 + \frac{1}{2} m_C V(C)^2 + T
\]

(4)

where, \( J_A \) and \( J_B \) are, mass moment of inertia at the origin and at the end of the link, respectively.

Note that the first term on the right-hand side in (4) is due to moment of inertia of the portion of the mass of the actuator relative to the link. The second term is due to moment of inertia of the mass at \( C \) (payload). The third term is the kinetic energy of the link. The forth term is the kinetic energy of the flexible link.

The potential energy \( U \) can be written as:

\[
U = \frac{1}{2} \mathbf{q}^T \mathbf{K} \mathbf{q}
\]

(5)

where, \( \mathbf{K} \) is the stiffness matrix. The first row and column of \( \mathbf{K} \) is zero as \( U \) does not depend on \( \mathbf{q}_r \). \( E \) is the Young modulus and \( I \) the quadratic moment of section of the link.

Dynamic motion equation of the flexible manipulator can be derived in terms of Lagrange-Euler formulation:

\[
\frac{d}{dt} \left[ \frac{\partial \ell}{\partial \dot{q}_r} \right] - \left[ \frac{\partial \ell}{\partial \dot{q}_e(i)} \right] = \Gamma
\]

(6a)

\[
\frac{d}{dt} \left[ \frac{\partial \ell}{\partial \dot{q}_e(i)} \right] - \left[ \frac{\partial \ell}{\partial \dot{q}_e(i)} \right] = 0 \quad (i = 1, 2)
\]

(6b)

where \( \ell \) is the Lagrangian function and \( \ell = T - U \).

Substituting (4) and (5) into (6a) and (6b) yields to:

\[
\mathbf{L}_r \Gamma = \mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{B}(\mathbf{q})[\mathbf{q}\dot{\mathbf{q}}] + \mathbf{C}(\mathbf{q})[\dot{\mathbf{q}}^2] + \mathbf{K} \mathbf{q}
\]

(7)

where \( \mathbf{A}(\mathbf{q}) \) is the \( (n \times n) \) inertia matrix, \( \mathbf{B}(\mathbf{q}) \) is the \( (n \times (n^2 - n) / 2) \) matrix of Coriolis terms and \( \mathbf{C}(\mathbf{q}) \) is an \( ((n^2 - n)/2 \times 1) \) vector of joint velocity products given by: \( \left[ \dot{q}_1 \dot{q}_2, \; \dot{q}_1 \dot{q}_3, \; \dot{q}_1 \dot{q}_4, \ldots, \; \dot{q}_{n-1} \dot{q}_n \right]^T \). \( \mathbf{C}(\mathbf{q}) \) is the \( (n \times n) \) matrix of centrifugal terms and \( \dot{\mathbf{q}}^2 \) is an \( (n \times 1) \) vector given by: \( \left[ \dot{q}_1^2, \; \dot{q}_2^2, \; \ldots, \; \dot{q}_n^2 \right]^T \). \( \mathbf{K} \) is the \( (n \times n) \) stiffness matrix and \( \mathbf{L}_r \Gamma \) is the \( n \) torque vector \( \left[ \Gamma_1, \Gamma_2, \ldots, \Gamma_n \right]^T \) applied to the joints. \( n \) is the total number of variables: \( n_r + n_e \) (rigid and elastic, respectively) of the system, in our case, \( n = 3, \; n_r = 1 \) and \( n_e = 2 \).

If we suppose that the length \( L \) of the link is known, we define [13]:

\[
\mathbf{X} = [J_A + J_B, J_B, m_C, \rho I, M, EI]^T
\]

(8)

where, \( \mathbf{X} \) is the vector of dynamic parameters of the robot.
III. NONLINEAR CONTROL

This control is a generalization of the classically known 'computed torque' used to control rigid manipulator [14]. It consists of a proportional and derived (PD) part completed by a reduced model which contains only the rigid part of the whole nonlinear dynamic model of the flexible manipulator [15]. Let:

\[ h(q, \dot{q}) = B(q) \dot{q} + C(q) \dot{q}^2 \]  

(9)

then, the model can be reduced to:

\[ L_r \Gamma = A(q) \ddot{q} + h(q, \dot{q}) \dot{q} + K_q \]  

(10)

or even,

\[
\begin{bmatrix}
\dot{q} \\
\dot{q} \\
\dot{q}
\end{bmatrix} =
\begin{bmatrix}
A_r & A_e & \dot{q} \\
A_e & A_e & \dot{q} \\
0 & 0 & K_e
\end{bmatrix}
\begin{bmatrix}
q \\
\dot{q} \\
\dot{q}
\end{bmatrix}
+ 
\begin{bmatrix}
h_r \\
h_e \\
0
\end{bmatrix}
\begin{bmatrix}
\dot{q} \\
\dot{q} \\
\dot{q}
\end{bmatrix}
+ 
K_q
\begin{bmatrix}
0 \\
0 \\
\dot{q}
\end{bmatrix}
\]  

(11)

We deduce from (11) that:

\[ \Gamma = A_r \ddot{q} + h_r \dot{q} + A_e \dot{q} + h_e \dot{q} \]  

(11a)

\[ 0 = A_e \ddot{q} + h_e \dot{q} + A_e \dot{q} + h_e \dot{q} + K_e \dot{q} \]  

(11b)

We can then use the following control law:

\[ \Gamma_{NL} = A_r (q_r, \dot{q}_r) \ddot{q}_r + h_r (q_r, \dot{q}_r, \dot{q}_r, \dot{q}_r) \dot{q}_r^d + K_{pr} \ddot{q}_r + K_{vr} \dot{q}_r \]  

(12)

where \( q_r^d, \dot{q}_r^d \) and \( \dot{q}_r^d \) define the desired angular trajectory. \( \ddot{q}_r = \ddot{q}_r - \ddot{q}_r \), \( \dot{q}_r = \dot{q}_r - \dot{q}_r \) are angular position and velocity errors. \( K_{vr} \) and \( K_{pr} \) are positive definite matrices of gain.

If we consider the ideal case where no errors are made while evaluating the dynamic parameters \( X \), a Lyapunov stability analysis of the control law is presented on Appendix.

IV. ADAPTIVE NEURAL CONTROL

The nonlinear law presented has some major advantages as it uses information extracted from the dynamic motion equations of the system to control the manipulator. Physical characteristics like the passivity of the system can be used to elaborate a stable controller [16].

The drawback is that, using the model of the system in the construction of the controller can lead to a complex controller. Computing such a controller can be time consuming. This is mainly the case with flexible manipulators as they are governed by complex equations which lead generally to a huge model. Using such a model can be incompatible with real time control.

To avoid this problem we propose to approximate parts of the model (which will be used in the controller) with neural networks. The main feature that makes neural network ideal technology for controller systems is that they are nonlinear regression algorithms that can model high-dimensional systems and have the extreme flexibility due to their learning ability. In addition their computation is very fast.

The functions \( A_r \) and \( h_r \) are approximated with the artificial neural networks \( A_{r,NN} \) and \( h_{r,NN} \). We will then use their output in addition to the PD part of (12) to elaborate the first controller,

\[ \Gamma_{NN} = A_{r,NN}(q_r, \dot{q}_r) \ddot{q}_r^d + h_{r,NN}(q_r, \dot{q}_r, \dot{q}_r, \dot{q}_r, \dot{q}_r) \dot{q}_r^d + K_{pr} \ddot{q}_r + K_{vr} \dot{q}_r \]  

(13)

In the neural network design scheme of \( A_{r,NN} \) and \( h_{r,NN} \), there are three-layered networks consisting of input, hidden and output layers. We used a sigmoid function in the hidden layer and a linear function in the output layer. The back-propagation algorithm is adopted to perform supervised learning [17].

The two distinct phases to the operation of back-propagation learning include the forward phase and the backward phase.

In the forward phase the input signal propagate through the network layer by layer, producing a response at the output of the network. In this control scheme, the input signals of the input layer for \( A_{r,NN} \) are the rigid and elastic position: \([\theta, f, \alpha]^T\). For \( h_{r,NN} \) the inputs are the rigid and the elastic position and velocity: \([\theta, f, \alpha, \dot{\theta}, f, \dot{\alpha}]^T\).

The actual response of \( A_{r,NN} \) and \( h_{r,NN} \) so produced are then compared with the desired response of \( A_r \) and \( h_r \) respectively, generating error signals that are then propagated in a backward direction through the network.

In the backward phase, the delta rule learning makes the output error between the output value and the desired output value change the weights and reduce the error. The training is made off line so that it does not disturb the real time control.

Using dynamic equations of the system in the place of the system itself to get the data subset to train the network presents many advantages.

The data (inputs/outputs set) are easily and rapidly obtained from simulation. They are not tainted with noise. They can be generated in sufficient number to have a good approximation of the model. We can generate data that have better representation of the model of the system. This is important and will ensure that the neural network will have a better capacity of generalization (better results with unseen data).

To reduce the modeling error between the actual system and its representation, we propose to add an adaptive neural controller. Here, the neural network is trained online, to compensate for errors due to structured and unstructured uncertainty, increasing the precision of the over all controller.
The basic concept of the adaptive neural network used in the second controller is to produce an output that forms a part of the control torque that is used to drive the manipulator joint to track the desired trajectory. The errors between the joint’s desired and actual position/velocity values are then used to train online the neural controller [1].

In the adaptive neural network design scheme there are also three layers. Sigmoid and linear functions are used in the hidden and the output layer respectively.

The input signals of the input layer are angular position and velocity: \([\theta, \dot{\theta}]^T\), and the output signal \(\mathbf{Y}\) of the output layer is the torque \(\Gamma_{AN}\).

Training is made online and the parameters of the network are adjusted so as to minimize the following error function:

\[
E = \frac{1}{2} (\mathbf{Y}^d - \mathbf{Y})^2 = \frac{1}{2} (K_{pn} \ddot{q}_r + K_{vn} \dot{q}_r)^2
\]

where \(\mathbf{Y}^d\) and \(\mathbf{Y}\) are the desired and actual output of the neural network. \(K_{pn}\) and \(K_{vn}\) are positive definite matrices of gain.

The overall robotic manipulator control system obtained is shown in Fig. 2. It can be written:

\[
\Gamma = \Gamma_{NN} + \Gamma_{AN}
\]

where, \(\Gamma\) is the overall controller output (torque); \(\Gamma_{NN}\) is the first controller output based on the neural model of the robot, as defined in (13); \(\Gamma_{AN}\) is the second controller output based on the adaptive neural network.

![Fig. 2. Overall control system](image)

To test the robustness of the control strategy proposed we consider the extreme case where the estimation error on the dynamic parameters \(\mathbf{X}\) is such that:

\[
\hat{\mathbf{X}} = \mathbf{X} / 100
\]

Then, we use these values (\(\hat{\mathbf{X}}\)) in the training of \(\mathbf{A}_p, \mathbf{NN}\) and \(\mathbf{h}_p, \mathbf{NN}\). This will drive the first controller to produce an incorrect torque. We will see how the second controller deals with this error and how it will correct it.

In order to better appreciate the effectiveness of the overall adaptive neural controller we will compare its results with the nonlinear controller given in (12).

Fig. 3 to Fig. 7 illustrate the results obtained with the adaptive neural controller and the nonlinear controller applied to the one-link flexible manipulator. These figures describe: angular position, error on position, deflection, angular velocity and error on the angular velocity, respectively.

The desired trajectory (target) is reported on Fig. 3 and Fig. 6, in dotted line. We have reported on Fig. 3 to Fig. 7 the results of the nonlinear control in dashed line for comparison. Table II presents the maximum error and the Root Mean Square error (RMS) of the angular position and velocity obtained with the two types of control strategy used.

The desired trajectory imposes a fast change of acceleration, as the acceleration is also of a sinusoidal form. This kind of trajectory stresses the control. We can see the impact of the change in the acceleration from negative to positive at \(t = 30\) s on the velocity tracking (see Fig. 6 and Fig. 7).

The velocity tracking with the adaptive neural controller is good and the error induced is acceptable, whereas the nonlinear controller strongly deviates from its target.

The controller proposed deals well with the elasticity of the link as the deflection is lessen (see Fig. 5).

For the position control (see Fig. 3 and Fig. 4) we notice that the angular trajectory obtained with the adaptive neural controller matches perfectly the target, with an error lower than 0.0033 rad, whereas it reaches 0.19 rad with the nonlinear controller (see Table II).

This demonstrates the good performance of the adaptive neural control proposed.

### Table I: Manipulator Characteristics

<table>
<thead>
<tr>
<th>Physical parameters</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (m)</td>
<td>(L = 0.80)</td>
</tr>
<tr>
<td>Moment of inertia at the origin of the link (kg m²)</td>
<td>(J_s = 1.80 \times 10^{-3})</td>
</tr>
<tr>
<td>Moment of inertia at the end of the link (kg m²)</td>
<td>(J_u = 4.70 \times 10^{-2})</td>
</tr>
<tr>
<td>Mass of the link (kg)</td>
<td>(m = 1.89)</td>
</tr>
<tr>
<td>Mass at the tip (kg)</td>
<td>(m_t = 4.0)</td>
</tr>
<tr>
<td>Mass density (kg/m³)</td>
<td>(\rho = 7860)</td>
</tr>
<tr>
<td>Young modulus</td>
<td>(E = 1.98 \times 10^9)</td>
</tr>
<tr>
<td>Quadratic moment of section (m⁴)</td>
<td>(I = 7.85 \times 10^{-9})</td>
</tr>
</tbody>
</table>
Fig. 3. Evolution of the angular position $\theta$ (rad)

Fig. 4. Evolution of the angular position error $\dot{\theta}$ (rad)

Fig. 5. Evolution of the deflection $f$ (m)

Fig. 6. Evolution of the angular velocity $\dot{\theta}$ (rad/s)

Fig. 7. Evolution of the angular velocity error $\ddot{\theta}$ (rad/s)

### TABLE II
Trajectory Error

<table>
<thead>
<tr>
<th>Variable</th>
<th>$\theta$ (rad)</th>
<th>$\dot{\theta}$ (rad/s)</th>
<th>$\ddot{\theta}$ (rad)</th>
<th>$\dddot{\theta}$ (rad/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive Neural Control</td>
<td>$3.29 \times 10^{-3}$</td>
<td>$1.59 \times 10^{-3}$</td>
<td>$8.89 \times 10^{-4}$</td>
<td>$6.04 \times 10^{-4}$</td>
</tr>
<tr>
<td>Nonlinear Control</td>
<td>$1.90 \times 10^{-1}$</td>
<td>$5.27 \times 10^{-2}$</td>
<td>$1.51 \times 10^{-1}$</td>
<td>$1.37 \times 10^{-2}$</td>
</tr>
</tbody>
</table>
VI. CONCLUSION

In this paper, a new adaptive neural control strategy was presented. This controller is a combination of two controllers. A static feed forward based controller and an adaptive neural network based controller.

The first controller is based on the approximation with the neural network of the dynamical equations of motion. Its aim is to provide a stable and fast control based on the dynamic model of the system.

The second controller is based on neural networks which are trained online. Its objective is to ensure that the actual trajectory matches the desired one by compensating errors due to structured and unstructured uncertainty, increasing the precision of the control.

Simulation results have shown the robustness in performance of this control design scheme against adverse effects such as model parameter variations.

In summary, this article provides a novel control structure, to over come the robotic manipulator control difficulties faced by conventional control schemes when uncertainties (e.g., friction, changing payload, time-varying friction, disturbances) cannot be ignored. Using artificial neural network in the place of the nonlinear model allow to simplify the structure of the controller reducing the computation burden and enhancing the reactivity of the control.

APPENDIX

By subtracting (12) from (11a), we obtain the error equation:

\[ A_r \ddot{q}_r + A_{re} \dot{q}_e + h_r \dot{q}_r + h_{re} \dot{q}_e + K_{pr} \dot{q}_r + K_{vr} \ddot{q}_r = 0 \]  

(A.1)

with, \( \ddot{q}_r = 0 - q_e = -q_e \) and \( \ddot{q}_e = 0 - \dot{q}_e = -\dot{q}_e \) representing the elastic stabilization errors.

In addition, rewriting the coupling equation (11b) according to the trajectory and the elastic stabilization error variables \( (q_e, \dot{q}_e) \) gives:

\[
A_r \ddot{q}_r + A_{re} \dot{q}_e + h_r \dot{q}_r + h_{re} \dot{q}_e + K_p \ddot{q}_r + K_v \dot{q}_e = 0
\]  

(A.2)

Using (A.1) and (A.2), the global error equation becomes:

\[
A \ddot{q} + h \dot{q} + K_p \ddot{q} + K_v \dot{q} + s_1 = 0
\]  

(A.3)

where the positive constant matrices \( K_P, K_v \) are

\[
\begin{bmatrix}
K_{pr} & 0 \\
0 & K_e
\end{bmatrix}, \quad \begin{bmatrix}
K_{vr} & 0 \\
0 & 0
\end{bmatrix}
\], respectively, and

\[
s_1 = \begin{bmatrix}
0 \\
A_{er} \dot{q}_{rd} + h_{er} \dot{q}_{rd}
\end{bmatrix}
\].

To study the stability of the global system, the following Lyapunov function is considered:

\[
V = (1/2) \dot{q}^T (K_p - h) \ddot{q} - (1/2) \ddot{q}^T (K_v \ddot{q} + s_1)
\]  

(A.4)

By differentiating \( V \) and using (A.3), we obtain:

\[
\dot{V} = \ddot{q}^T (1/2) A - h) \ddot{q} - \dot{q}^T (K_v \ddot{q} + s_1)
\]  

(A.5)

The property of passivity of the flexible robot implies that \((1/2) A - h) \ddot{q} \) is skew symmetric, finally we have:

\[
\dot{V} = -\ddot{q}^T K_v \ddot{q} + \dot{q}^T (A_{er} \dot{q}_{rd} + h_{er} \dot{q}_{rd})
\]  

(A.6)

The Lyapunov second method provides that the asymptotic stability of the control is assured if the following conditions are met. \( V \) is strictly positive everywhere except in \( \ddot{q} = 0 \) where it is 0 and \( \dot{V} \) is strictly negative everywhere except in \( \ddot{q} = 0 \) where it is 0. These conditions are always met if the desired angular velocity and acceleration are not too significant for a given tuning of \( K_v \), so that \( V \) remains essentially negative to ensure the control stability.

REFERENCES


