Adaptive Robot Control - An Experimental Comparison

Regular Paper

Francesco Alonge¹ and Filippo D’Ippolito¹,*

¹ University of Palermo, Palermo, Italy
* Corresponding author E-mail: filippo.dippolito@unipa.it

Received 31 May 2012; Accepted 24 Sep 2012

DOI: 10.5772/53731

© 2012 Alonge and D’Ippolito; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract This paper deals with experimental comparison between stable adaptive controllers of robotic manipulators based on Model Based Adaptive, Neural Network and Wavelet-Based control. The above control methods were compared with each other in terms of computational efficiency, need for accurate mathematical model of the manipulator and tracking performances. An original management algorithm of the Wavelet Network control scheme has been designed, with the aim of constructing the net automatically during the trajectory tracking, without the need to tune it to the trajectory itself. Experimental tests, carried out on a planar two link manipulator, show that the Wavelet-Based control scheme, with the new management algorithm, outperforms the conventional Model-Based schemes in the presence of structural uncertainties in the mathematical model of the robot, without pre-training and more efficiently than the Neural Network approach.

Keywords Robot control, Model-Based Adaptive control, Wavelet based control

1. Introduction

Model Based Adaptive (MBA) control of robotic manipulators has been widely investigated due to its capability in coping with those system modifications which can be modelled as parameter variations, also of high entities, of the nonlinear mathematical model of the manipulators themselves [1-5] such as, for example, phenomena of wear and ageing. The main assumption in the synthesis of MBA controllers is the knowledge of the structure of the mathematical model of the system, whereas the parameters appearing in this model can be unknown.

Unfortunately, the manipulators can rarely be perfectly modelled. For example, manipulators which grasp certain unknown mechanical characteristics, for example, can be modelled with uncertainties both in the structure and in the parameters of the mathematical model. Moreover, it is difficult to model stiction components of friction torques and, consequently, it is not convenient to take them into account explicitly in the model used for the MBA controller design.

Obviously, the presence of structural uncertainties in the mathematical model of the plant can deteriorate the performance of the above MBA controller, depending on their entity, resulting in unsatisfactory results or instability [6-8]. To overcome this drawback, some authors propose the addition of additional loops to the MBA controller (cf. for example [6]), whereas other
authors propose non-parametric adaptive control structures, such as Neural Networks (NN) [8]-[19]. The NN is usually designed in order to identify the inverse dynamics of the system to be controlled and then implemented online as a feedforward controller [8]-[12]. An optimal structure of the NN can be obtained using Genetic Algorithms (GA) [12]-[14]. Furthermore, if parameter variations occur during operation, additional features can be introduced consisting of online adaptation sto the weights of the net, the so-called “specialised learning phase”, which allows one to obtain a neural controller whose performance is optimised for the considered reference trajectories [9], [11].

The gradient descent optimisation method, employed for adjustment of the parameters of the net, is effective in practice in the presence of structural modifications to the system [7], [12]; however, currently there is no systematic way to ensure these methods will be successfully employed. Moreover, offline pre-training of the net is often necessary. The analysis of the system becomes further complicated when learning and control are attempted simultaneously.

In two studies [19] and [20], NN controllers for robot manipulators are designed with enhanced performances while retaining the stability and convergence characteristics of MBA controllers. These algorithms treat the inverse dynamics of the manipulator as an unknown function and try to approximate this function. Practical applications of these techniques crucially depend on both the accuracy of the function approximation and efficiency of the computational structure of the approximate function.

The structure of a neural net is the topological representation of a weighted superposition of functions, particularly in the case of the single-hidden-layer perceptron and, consequently, function decomposition over a given functional basis can be represented as a neural net. There are two main topics related to function approximation via NN; the first topic regards the network structure, i.e., the choice of the topology, the family of activation functions of the nonlinear nodes, and the number of these nodes. The second topic concerns the choice of an updating law for the output weights, so that the net output could approximate the unknown function in a context of adaptive nonlinear approximation.

Within the context of the function approximation by means of NNs, the kind of activation function holds great importance. In three studies [21]-[23], fuzzy membership functions have been used. In this context, wavelets are potentially a good choice as activation functions because they have a practically-bounded support both in space and frequency [24], [26]. These characteristics allow the dynamic management of the net, adding and removing nodes from its structure without interfering with the weights of the other nodes.

Another advantage of wavelets over other basis functions, like splines or even sinusoids, is the multiresolution property which allows more efficient representation of local details of the approximated function. The multiresolution capabilities make the wavelets a very efficient tool, since by retaining only the coefficients which are above a certain threshold, it is possible to obtain a “sparse” representation which does not imply a significant loss of details [26].

An interesting control structure is that illustrated in Cannon and Slotine’s study [27], which uses a dynamic wavelet network, the structure of which is updated according to the hard-threshold algorithm [26]. In Sanner and Slotine’s study [28], the approach used in [27] was extended to the adaptive control of robotic manipulators (WN) and a simple algorithm was proposed to manage the network nodes: for each active neuron the nearest neurons can be activated if the weight of the active neuron is greater than a given threshold. A drawback of this mechanism is that the behaviour of the system is greatly influenced by the choice of the above threshold, which implies that the net can have a number of nodes either too low or too high for the task to be executed.

In Alonge et al.’s study [29], a new activation mechanism is proposed, based on the observation that in order to ensure a good tracking of a given trajectory, the active nodes of the net must cover at least all the points of the net input space, which are actually part of the trajectory itself. To this end, the following procedure is employed: a) during the evolution of the system, a new node at a maximum scale is inserted each time the trajectory of the system reaches an interval of the net input space not already covered by other nodes; b) insertion and removal of nodes with activation function at minor scales is carried out so that local details of the new unknown dynamics can be better approximated.

Step b) is executed according to a routine based on the threshold algorithm previously described, using local thresholds instead of global ones. In particular, for each maximum scale node inserted in the network structure, a couple of thresholds are considered and updated starting from the local tracking error. When the weight of a node exceeds the insertion threshold, the nodes which are adjacent in the wavelet index space are inserted; in the same way, nodes are removed when their weights go below the removal threshold. Since the thresholds are locally updated on the basis of the tracking error, insertion of new nodes is more probable in those areas of the input space in which the dynamics are insufficiently modelled. In the paper, the previously described algorithm for basis function selection is considered. The
resulting adaptive algorithm, named WN+, was experimentally compared with the MBA and NN approach. Moreover, the complexity of the resulting network is compared with the complexity of the WN controller illustrated in Sanner and Slotine’s study [28].

In Section 2 the MBA approach we consider in the paper is first reviewed then the WN+ approach is introduced with the new network management algorithm. In Section 3, experimental results on a planar two-link manipulator are displayed with the aim of comparing the WN approach with the MBA approach in two studies, [2] and [5] and NN based control laws [7], [12] and [28]. Finally, Section 4 deals with some conclusions.

2. The adaptive controllers

2.1 Model-Based Adaptive Controller

A robotic manipulator is a dynamic structure consisting of a number of links joined together by joints typically actuated by motors coupled to the joints themselves. For a manipulator with m joints, a typical mathematical representation of the dynamics of the system is the Euler model given by:

\[ \mathbf{B}(q) \ddot{q} + \mathbf{C}(q, \dot{q}) \dot{q} + \mathbf{E}(q, \dot{q}) = \mathbf{u} \]  

where \( \mathbf{u} \in \mathbb{R}^m \) is the vector of the torques generated by the motors connected to the joints at the generic instant \( t \) and \( q \in \mathbb{R}^m \) is the vector of the generalized joint coordinates at the same instant. The matrix \( \mathbf{B}(q) \) is the inertia matrix and collects terms relative to the inertia effects at the joints, the matrix \( \mathbf{C}(q, \dot{q}) \) collects Coriolis and centripetal torques and the vector \( \mathbf{E}(q, \dot{q}) \) represents every torque acting on the joints due, for example, to environmental forces, such as gravity forces, dynamic and static friction.

Given an assigned desired trajectory in the joint space, specified by means of a couple of position and velocity functions \( q_d(t), \dot{q}_d(t) \), it is possible to force the system to track asymptotically this trajectory by generating at the joints the control torque vector given by [2]:

\[ \mathbf{u} = \mathbf{B}(\dot{q}) \ddot{q}_d + \mathbf{C}(q, \dot{q}) \dot{q}_d + \mathbf{E}(q, \dot{q}) - \mathbf{K}_D \ddot{s} \]  

where \( \mathbf{B}(\ddot{q}), \mathbf{C}(q, \dot{q}) \) and \( \mathbf{E}(q, \dot{q}) \) are estimates of the corresponding matrices and \( e = q - q_d, \dot{e} = \dot{q} - \dot{q}_d + \dot{\mathbf{A}} e, \quad \ddot{e} = \ddot{q} - \ddot{q}_d + \dot{\mathbf{A}} \dot{e}, \quad s = e + \dot{\mathbf{A}} e \).

The control law (2) is based on an inverse dynamic model of the system used to compensate the nonlinearity of the model (1), so that the resulting closed-loop system, decoupled and linear, can be driven by the PD action \(-\mathbf{K}_D \ddot{s}\). This kind of approach, called “inverse dynamic control”, allows an asymptotically exact tracking of the desired trajectory, assuming that exact knowledge of the inverse dynamics model is assured. This led to the introduction of various techniques used to make the control structure robust against uncertainties in the model, like the use of a sliding controller in the control loop (cf. for example [6]), which allows it to cope with unmodelled dynamics such as, for example, some components of friction. By applying such robust techniques, the control structure can be used when an approximated modelling of the inverse dynamics of the manipulator is available.

If model (1) admits the linear parameterisation:

\[ \mathbf{B}(q) \ddot{q} + \mathbf{C}(q, \dot{q}) \dot{q} + \mathbf{E}(q, \dot{q}) = \mathbf{Y}(q, q, \dot{q}, \dot{q}) \mathbf{p} = \mathbf{u} \]  

where \( \mathbf{Y} \) is a matrix of nonlinear known functions and \( \mathbf{p} \) is a vector of unknown constant parameters, an adaptive version of (2) is given by:

\[ \mathbf{u} = \mathbf{Y}(q, q, \dot{q}, \dot{q}) \mathbf{\hat{p}} - \mathbf{K}_D \mathbf{s} \]  

\[ \mathbf{\hat{p}} = -\Gamma \mathbf{Y}^T (q, q, \dot{q}, \dot{q}) \mathbf{s} \]  

where \( \Gamma \) and \( \mathbf{K}_D \) are positive definite matrices.

However, some nonlinear systems which can be modelled using (1) cannot be parameterized as in (3) and, consequently, the adaptive approach (4) and (5) cannot be applied even if the structure of the matrices \( \mathbf{B}(q), \mathbf{C}(q, \dot{q}) \) and \( \mathbf{E}(q, \dot{q}) \) is known.

2.2 Wavelet Network Controllers

Following [25], let us assume that the matrices \( \mathbf{B}(q), \mathbf{C}(q, \dot{q}) \) and \( \mathbf{E}(q, \dot{q}) \) are unknown. In order to design an adaptive control structure following (2), it is convenient to express the dynamic model of the robotic manipulator in terms of a matrix \( \mathbf{M}(x) \) of unknown functions, as:

\[ \mathbf{M}(x) x_d = \mathbf{u} \]  

where

\[ \mathbf{M}(x) = [\mathbf{B}(q), \mathbf{C}(q, \dot{q}), \mathbf{E}(q, \dot{q})] \]  

and \( x = [q^T \quad \dot{q}^T] \), \( x_d = [q_d^T \quad \dot{q}_d^T \quad 1]^T \).

With previous positions, the control law (2) can be written as:

\[ \mathbf{u} = \mathbf{N}(x) \mathbf{v}(x, t) - \mathbf{K}_D \mathbf{s} \]  

where \( \mathbf{v} = [\dot{q}^T, \dot{q}_d^T, 1]^T \) and \( \mathbf{N}(x) \) is an estimate of \( \mathbf{M}(x) \). Each element of the matrix \( \mathbf{N}(x) \) can be approximated by means of a linear combination of functions from a
suitable set of basic functions consisting of wavelets. The weights of this linear combination can be updated online, therefore realizing a kind of nonlinear adaptive control.

A wavelet network is a family of functions which present some useful properties in the field of function approximation, resulting in a powerful tool to perform signal analysis and synthesis. As mentioned previously, the main advantages of the wavelet analysis over other methods of function decomposition, like splines or even Fourier analysis, are the multi-resolution property and the compact support.

A wavelet basis is a family of functions having a double index each function of the set has a translation index, which is a vector if the set is defined over a multidimensional input space, and a scaling index which is often scalar. Multidimensional scale indices are also used in the so-called multiscale wavelet frame. A wavelet basis is thus defined as an infinite collection of translated and scaled versions of a single motherwavelet. Since the mother wavelet is a function with a compact support, the multiple scales of the wavelets allows representation of the analysed function with various levels of detail, maintaining the localization of the information about the characteristics of the analysed function. The multi-resolution analysis also makes the wavelets a very efficient tool, since retaining only the basis function whose coefficients are above a certain threshold, it is possible to obtain a "sparse" representation of the function which does not involve a significant loss of details.

Within the context of function approximation by dynamic neural networks, wavelets are potentially a good choice as activation functions; in fact, while the compact support allows the management of the network dynamically, adding and removing nodes from the net structure without interfering with the weights of the other nodes, the multi-resolution property allows efficient representation of the local details of the approximated function.

The wavelet expansion used for function approximation can be considered as a neural net with a single layer the nodes of which have wavelets as activation functions.

In order to approximate the function matrix \( \mathbf{M}(\mathbf{x}) \) it is necessary to use a multidimensional wavelet representation; this has been obtained as a multidimensional radial wavelet frame based on the Mexican hat scalar mother wavelet given by:

\[
\psi(x) = (1 - x^2) \cdot e^{-\frac{x^2}{2}} \tag{9}
\]

For an n-dimensional input space, the corresponding wavelet family is obtained as follows:

\[
\psi_{j,k}(x) = (n - 2^j x - k[\frac{1}{2}]) \cdot e^{-\frac{x^2}{2}} \tag{10}
\]

where \( j \in \mathbb{Z}, \ k = [k_1, \ldots, k_n]^T \in \mathbb{Z}^n \) are the scaling and translation indices.

Wavelets of the form (10) satisfy the so-called “frame” properties [13], which ensure that a function \( f \in L^2(\mathbb{R}^n) \) having a compact support can be approximated within a given error by a superposition of a finite number of elements in the form (10). The above error depends on the need for truncation of the frame in order to obtain a neural network with a finite number of nodes and, consequently, physically realizable or with acceptable computational cost, so that its implementation on a digital system can be efficient.

In order to affect the frame truncation it is convenient, first of all, to choose a dominion of the space \( \mathbb{R}^n \) of \( \mathbf{x} \), where \( n = 2m \), in which a good approximation of \( \mathbf{M}(\mathbf{x}) \) has to be obtained. This dominion denoted by \( I_d \) is that containing the desired trajectories of the manipulator. Moreover, the elements of \( \mathbf{M}(\mathbf{x}) \) are continuous function but do not belong to \( L^2(\mathbb{R}^n) \); consequently, the conditions for their approximation, with a given error, by superposition of functions of a frame, are not satisfied. To overcome this problem, the following procedure is employed.

A dominion \( I_{ex} \supset I_d \) and a modulating function \( m(\mathbf{x}) \) are defined so that:

\[
m(\mathbf{x}) = \begin{cases} 1 & \mathbf{x} \in I_d, \\
0 < m(\mathbf{x}) < 1 & \mathbf{x} \in I_{ex}, \mathbf{x} \notin I_d, \\
0 & \mathbf{x} \notin I_{ex}. \end{cases} \tag{11}
\]

A new matrix \( \mathbf{M}(\mathbf{x}) = \mathbf{M}(\mathbf{x})m(\mathbf{x}) \) is defined which coincides with \( \mathbf{M}(\mathbf{x}) \) inside \( I_d \) and is zero outside \( I_{ex} \). Obviously, \( \mathbf{M}(\mathbf{x}) \in L^2(\mathbb{R}^n) \) and consequently its elements can be approximated at assigned extent by means of a superposition of a finite number of elements of a frame. Let \( \mathbf{M}(\mathbf{x}) \) be the estimate of \( \mathbf{M}(\mathbf{x}) \). The matrix \( \mathbf{M}(\mathbf{x}) \) approximates well \( \mathbf{M}(\mathbf{x}) \) only inside of \( I_d \). It follows that the component of the control law based on the estimate of \( \mathbf{M}(\mathbf{x}) \), i.e., the neural control component, should be activated only when the state \( \mathbf{x} \in I_d \). Outside \( I_{ex} \), \( \mathbf{M}(\mathbf{x}) \) is zero and the neural control component is useless. The presence of unknown dynamics outside \( I_{ex} \) suggests the use of a robust control component which stabilizes the whole system. Finally, when the state is inside \( I_{ex} \) but outside \( I_d \) it is convenient to modulate the robust and the neural components.
The previous considerations suggest the following architecture of the control law:

$$ u = -K_D s + m(x) \hat{N}(x)^v + (1-m(x))u_{sl} \quad (12) $$

where $K_D$ is a diagonal matrix and $u_{sl}$ is the sliding mode type robust control component given by:

$$ u_{sl} = K_s \text{sgn}(s) \quad (13) $$
in which $K_s$ is a positive definite diagonal matrix whose elements satisfy the conditions:

$$ k_{ij} = \sum_{j=1}^{2m+1} M_{ij}(x) v_j \quad (14) $$

With reference to the structure of the net, it is convenient to observe that $x$ and $M(x)$, and consequently $\hat{N}(x)$, are, respectively, $2m \times 1$ and $n \times (2m+1)$ matrices; consequently, the output layer consists of $m \times (2m+1)$ nodes, whereas the $2m$ inputs are directly connected to the nodes of the hidden layer. The output layer is connected to the hidden layer by means of weighted connections whose weights are the coefficients of the wavelet expansion of the elements of $M(x)$ to be estimated of $\hat{N}(x)$. These weights are updated according to an adaptive law obtained from the stability analysis of the whole system.

Regarding the choice of the number of neurons of the hidden layer, truncation operations of the frame are obtained by defining limits on the wavelet indices $j$ and $k_j$, $i=1,\ldots,n$. To this end, it is convenient to observe that the domain of the vector $x$ is $[0, 2\pi]$ for rotational joints and $[0, L]$ for translational joints. Moreover, the velocities at the joints are also bounded due to the performances of the actuators.

The superior limit on $j$ is chosen so that the support of the wavelet has the minimum dimension of $L_j$, whereas the inferior limit is chosen according to the spectral content of the unknown function to approximate. The limits on $k_j$ for assigned $j$ are given by:

$$ \text{cell}(2^j x_{j, \text{min}}) \leq k_j(j) \leq \text{floor}(2^j x_{j, \text{max}}), $$

where $\text{floor}()$ and $\text{cell}()$ denote the round-off to the nearest integer number superior or inferior, respectively.

Obviously, the above limits imply a finite number of nodes of the net. Let $K_{\text{nd}} = \{k = [j_1, \ldots, j_n]^{T} \in \mathbb{Z}^{n+1} \}$ denote the finite subset of $\mathbb{Z}^{n+1}$ consisting of the indices $j_1, \ldots, j_n$, which satisfy the scale and translation limits. This subset $K_{\text{nd}}$ contains the potential wavelets and consequently, the potential nodes of the net which can be activated to participate in the reconstruction of the unknown dynamics. Since the number of these nodes is usually high, it is convenient to use a dynamical structure of the net, i.e., an adaptive structure in which the neurons are either added or removed according to the actual trajectory to track. It follows that the set of active neurons at a given instant of time, denoted by $K_{\text{nact}}$, is a subset of $K_{\text{nd}}$. Moreover, in order to avoid discontinuities in the control signals, the weights of the nodes to be activated start from zero and are updated gradually using the adaptive updating law, whereas the weights of the nodes to be removed are gradually brought to zero using another updating law.

The generic element $\bar{\eta}_{ij}(x)$ of $\hat{N}(x)$ is computed online as follows.

$$ \bar{\eta}_{ij}(x) = \sum_{k \in K_{\text{nact}}} \bar{\omega}_{ij,k} \psi_{j,k}(x) \quad (14) $$

where the quantities $\bar{\omega}_{ij,k}$ are the output weights of the net corresponding to the active node whose activation function is $\psi_{j,k}$. As shown in [10], choosing the weights of the network according to the updating law:

$$ \bar{\omega}_{ij,k} = \alpha_i s_i \psi_{j,k}(x) \quad (15) $$

for the nodes considered active, where $s_i$ and $\alpha_i$ are respectively the $i$-th and the $l$-th components of $s$ and $\nu$ and the updating law:

$$ \bar{\omega}_{ij,k} = -\eta \text{sgn}(\bar{\omega}_{ij,k}) \quad (16) $$

for those nodes which are selected for deletion, the overall system results are stable and converge asymptotically when the filtered tracking error is greater than a given threshold given by:

$$ ||e|| \geq \|e_s\| + \sqrt{\|e_s\|^2 + 4\eta k_{d,\text{min}}^2} \quad (18) $$

where $k_{d,\text{min}}$ is the minimum value of the elements of $K_D$, $e_s = 2\eta n \hat{N}_{\text{com}}^n (n+1) w_{\text{max}} / \min(i_k)$ and

$$ e = m(x)|E_r + E_d + E_f| \nu $$

where $a) E_r$, $E_d$ and $E_f$ are, respectively, the error due to the gradual updating of the weights of the nodes to be removed, the error due to the dynamical management of the net which implies that some nodes are removed and consequently the wavelet expansion involves a reduced number of wavelets of the chosen frame, and the errors due to the truncation of the frame; b) $N_{\text{com}}$ and $w_{\text{max}}$ are the number of the nodes to be removed and the maximum absolute value of the weights of the net. The proof is similar to that shown in a study by Gomi and Kawato [10].
Note that (17) implies that the $s$ and, consequently, the tracking error $e$ do not converge to zero. To increase the tracking performance of the whole system it needs to reduce $\|e_{i}\|$ and $e_{i}$. The quantity $e_{i}$ can be reduced by reducing the number of nodes to be removed, whereas $\|e_{i}\|$ can be reduced by reducing $E_r$, $E_d$ and $E_f$, which implies the increasing of the complexity of the net.

A great importance is assigned to the algorithm which causes the insertion and removal of the nodes of the net. In a study by Gomi and Kawato [10], for a similar control system, an algorithm based on a simple threshold has been proposed. In this solution, whenever the weights of a node active in the net overcome a certain threshold $\theta$, all the nodes which have adjacent wavelet indexes are inserted in the net structure. In a similar way, all the nodes whose weights go under a given threshold $\mu$ are removed from the structure, after their weights have been gradually brought to zero. A drawback of this solution is that the behaviour of the system is greatly influenced by the choice of the numerical values of the thresholds, so that the result may be either insensitivity to the weight growth, thus keeping too low the number of nodes of the network, or oversensitivity generating a network with an excessive complexity. Within a context of adaptive control, it seems unrealistic that the algorithm managing the net needs to be tuned to the problem in order to give good results.

Moreover, in order to bring an efficient structure to the net, the algorithm managing the net must correlate the subset of the input space effectively covered by current trajectories with the position of the active neurons and the number of wavelets with minor scale index, with tracking error, which is a measure of the wellness of the local identification. Following the above consideration, in order to ensure a good approximation of the unknown dynamics of the manipulator, the active nodes in the net must, at least, cover all the points of the net input space which are actually part of the trajectory of the system. Therefore, in the proposed algorithm, since each activation function has a bounded support and centres are located in an m-dimensional grid, during the evolution of the system new nodes are inserted each time the vector $\begin{bmatrix} q^T & q^T \end{bmatrix}$ reaches an interval in the grid which is not already covered by another node.

Besides this major policy of node insertion, which ensures at least the minimum degree of coverage of the input space, a second routine causes the insertion and removal of nodes with activation function at minor scales, so that local details of the unknown dynamics can be better approximated. This routine is based on the threshold algorithm described previously but uses local thresholds instead of global ones. For each maximum scale node inserted in the network structure, a couple of thresholds are considered, the values of which are updated during the control action on the base of the local tracking error. The approach described before concurs to the insertion of wavelets in the frame, which gives a valuable contribution to the error tracking reduction, taking advantage, in such a way, of space localisation properties of such functions.

The update algorithm of local thresholds is obtained as a discrete time task, the period of which is $T$. Each time this task is activated, it checks the maximum value of the filtered tracking error $s$, which has been measured during the previous time interval, and compares it with a given global threshold (Error Threshold). If the filtered error is greater than the Error Threshold then local thresholds of the nodes whose centre is nearest to the actual value of the network input, $x$, are modified as

$$
\begin{align*}
\dot{\theta}_kT & = \dot{\theta}_{kT-1} - \alpha \\
\dot{\mu}_kT & = \mu_{kT-1} - \beta
\end{align*}
$$

(18)

where $\alpha < 1$, $\beta > 1$, $\alpha, \beta \in \mathbb{R}$, making more probable, in such a way, the local insertion of wavelets at different scales.

The nodes which are not interested by the (18) have their threshold brought back to their nominal values, $\dot{\theta}_k, \dot{\mu}_k$, by a similar formula

$$
\begin{align*}
\dot{\theta}_kT & = \dot{\theta}_{kT-1} - \alpha & \text{if } \dot{\theta}_{kT-1} < \dot{\theta}_k \\
\dot{\mu}_kT & = \mu_{kT-1} - \beta & \text{if } \mu_{kT-1} < \mu_k
\end{align*}
$$

(19)

After this threshold adjustment, the algorithm is applied as in the study by Gomi and Kawato [10]; thus, when a weight of a node exceeds the insertion threshold, nodes which are adjacent in the wavelet index space are inserted; in the same way, nodes are removed when their weights go below the reduction threshold. Since the thresholds are locally updated on the base of the tracking error, insertion of new nodes is more probable in those areas where the dynamics are insufficiently modelled, with the presence of nodes at multiple resolutions. The described procedure can alleviate the problem of the ingestible increasing of the number of basis function with the dimensionality of the input space. In fact, as a result of the algorithm, the wavelet allocation is near the actual trajectories on the net input space by making the problem, in such a way, nearly always 1-dimensional.

3. Experimental tests

Experimental tests were carried out in order to prove the practical application of the WN control law and compare it with both the NN controller described in [7] and [12] and MBA control described in [5] and [6]. In this section, the results of the above experiments are discussed.
The experimental equipment is illustrated in figure 1 and consists of:

- a two link planar robotic manipulator directly driven by variable reluctance motors (NSK Megatorque: stall torque 250Nm for link 1 and 40Nm for link 2), equipped with two resolvers and resolver-to-digital converter boards equivalent to a 19,200ppr incremental encoder;
- industrial controllers designed for torque control loops, with ±10 volt torque command signals;
- dSpace DS1103 PPC Controller Board;
- the Real-Time Interface software from dSpace, which automatically generates real-time code from Simulink models and implements this code on dSPACE real-time hardware.

The manipulator to be controlled is a two-DOF SCARA type whose structure is illustrated in Fig. 2. The mathematical model used for the implementation of MBA control law is given by:

\[ B(q)\dot{q} + C(q, \dot{q})\dot{q} = Y(q, \dot{q}, \ddot{q})p = u \]  

where matrices and parameters are given in Tables 1-3. The elements of vector \( p \) are considered unknown and were updated online with an MBA approach.

An application of MBA control law is also considered based on the same model as above, but without load (i.e., \( M_p = 0, I_p = 0, \delta = 0 \)). The corresponding control law, named MBA+, was considered in this paper in order to test the robustness properties of MBA controllers to the manipulator structure uncertainties. Note that the presence of an asymmetric load implies both structure and parameter variations in the mathematical model, because both matrix \( Y \) and nominal values of the parameters \( p \) changes.

Finally, a NN controller was considered in the paper because for its implementation, as for WN, the knowledge of the mathematical model of the manipulator is not necessary. The considered approach is that described by Alonge et al.[12], in which a multi-layer sigmoidal perceptron was used to control the manipulator sketched in figure 2 but without load. The final structure of the net was obtained from an offline training phase with input-output data acquired from a closed loop scheme with a PD controller, followed by an optimization of the net structure based on Genetic Algorithm, consisting in the determination of an optimal net architecture, number of hidden layers, number of neurons, connectivity percentage for each layer and percentage of connections between layers. The resulting neural network has the following characteristics: 7 input variable \( (q_2, \dot{q}_1, \dot{q}_2, sgn(q_1), sgn(q_2), \dot{q}_1, \dot{q}_2) \), 14 neurons in the hidden layer, 57% as connectivity percentage towards the hidden layer and 91% towards the output layer, 2 output neuron. Online learning takes place starting from an ANN which gives good initial inverse modelling after offline batch training process but use of multilayer sigmoidal perceptron implies a number of about one hundred parameters to adapt in the online application.

In all tests, the desired trajectory is:

\[
\begin{align*}
q_1(t) &= \frac{\pi}{4}(1 - \cos(\pi t)) \\
q_2(t) &= \frac{\pi}{4}(1 - \cos(1.4\pi t))
\end{align*}
\]

and the parameters of the PD control component, for all the controllers, are chosen as \( K_p = K_d = diag(700, 80) \), \( K_d = diag(70, 8) \).

In order to show the capabilities of the proposed network management algorithm, two experiments were performed using the same control scheme with two different values of the maximum allowed error. The results displayed in Table 4 show that improving the tracking performances, by decreasing of the maximum allowed error, produces increasing of the complexity of the net.

The experimental results are displayed in Figures 3–9. Figures 3–8 show the results obtained applying WN+ control law with the Error Threshold chosen as in the first column of the Table 4. In particular, Figures 3 and 4 show joint tracking errors, Figures 5 and 6 show the applied control torques and Figure 6 shows the shape of the end-point tracking error. Finally, Figure 8 shows the number of nodes at different scales automatically obtained by the node allocation mechanism.

Figure 9 shows the displacement of the filtered end-point tracking errors for WN, NN, MBA and MBA- control laws, which were obtained from MBA without modelling the grasped load. Filtering was necessary in order to extract the average displacement in a suited temporal window, from end-point tracking errors. The chosen temporal window was 5 seconds. The comparative tests show that MBA is the best control approach if a sufficiently accurate model of the manipulator is known. But if the model does not take into account, for example, the grasped load, the performance deteriorates (MBA-). NN can learn the structural modification as illustrated by the corresponding shape of the filtered tracking error but this approach necessitates a long pre-training phase and optimisation phase. A comparison between WN+ and NN controllers shows that the application of NN controller causes the tracking error to start from lower than the average value because of its pre-trained structure, but...
application of WN+ outperforms that of NN in a short time. Moreover, the final structure of WN has 14 neurons, whereas NN has 100.

Finally, a third experiment has been performed in order to compare performances of the WB and WB+ algorithms in term of complexity of the net in both cases. The results illustrated in Figure 10 show that WB+ bring in a simpler structure than that using WB, as theoretically expected.

![Functional scheme of the experimental setup.](image)

**Figure 1.** Functional scheme of the experimental setup.

![Schematic structure of the 2-DOF SCARA manipulator with asymmetric load.](image)

**Figure 2.** Schematic structure of the 2-DOF SCARA manipulator with asymmetric load.

$$D(q) = \begin{bmatrix} p_1 + 2p_2q_2 & p_2 + p_3q_2 + p_4q_2 \\ p_2 + p_3q_2 + p_4q_2 & p_2 \end{bmatrix}$$

$$C(q, \dot{q}) = \begin{bmatrix} -p_3q_2q_2 & -p_3q_2 (q_1 + q_2) + p_4q_2q_2 \\ p_3q_2q_1 & 0 \end{bmatrix}$$

**Table 1.** Matrices of the 2-DOF SCARA manipulator model.

| $M_1$, $I_1$ | mass and inertia coefficient of the link 1 |
|-------------|------------------------------------------|
| $M_2$, $I_2$ | mass and inertia coefficient of the link 2 |
| $M_p$, $I_p$ | mass and inertia coefficient of the payload |
| $I_{r1}$ | inertia coefficient of the motor 1 |
| $M_{m2}$, $I_{r2}$ | mass, inertia coefficient of the stator and inertia coefficient of the rotor of the motor 2 |
| $L_1$ | length of link 1 |
| $L_2$ | length of link 2 |
| $L_{c1}$ | distance of the centre of gravity of the link 1 and the axis of the joint 1 |
| $L_{c2}$ | distance of the centre of gravity of the link 2 and the axis of the joint 2 |
| $\delta$ | angle between axis of link 2 and the line from centre of gravity of the payload to axis of joint 2 |

$$p_1 = I_{r1} + I_1 + I_2 + I_s + I_2 + I_p + (M_{m2} + M_2 + M_p) L_1^2 + M_1 L_{c1}^2 + M_2 L_{c2}^2 + M_p L_p^2$$

$$p_2 = I_{r2} + I_2 + I_p + M_2 L_{c2}^2 + M_p L_p^2$$

$$p_3 = M_2 L_{c2} + M_p L_p \cos \delta$$

$$p_4 = M_p L_p L_p \sin \delta$$

**Table 2.** Mechanical parameters.
Mechanical parameters

\[ M_1 = 9.78 \text{ Kgf} \]
\[ M_2 = 4.45 \text{ Kgf} \]
\[ M_{m2} = 14 \text{ Kgf} \]
\[ I_1 = .934 \text{ Kgf} \cdot \text{m/s}^2 \]
\[ I_2 = .063 \text{ Kgf} \cdot \text{m/s}^2 \]
\[ I_{m2} = .267 \text{ Kgf} \cdot \text{m/s}^2 \]
\[ I_{m2} = .0075 \text{ Kgf} \cdot \text{m/s}^2 \]

\[ L_1 = .24 \text{ m} \]
\[ L_2 = .102 \text{ m} \]

Parameter vector \( \mathbf{p} = \begin{bmatrix} p_1 & p_2 & p_3 \end{bmatrix}^T \)

\[ p_1 = 3.3165 \text{ Kgf} \cdot \text{m/s}^2 \]
\[ p_2 = .1168 \text{ Kgf} \cdot \text{m/s}^2 \]
\[ p_3 = .1630 \text{ Kgf} \cdot \text{m/s}^2 \]

Table 3. Nominal values of both the mechanical parameters and the vector \( \mathbf{p} \).

| Error threshold [rad] | \( 10^{-2} \) | \( 5 \times 10^{-2} \) |
|-----------------------|---------------|----------------------|
| Nodes at scale -1    | 9             | 9                    |
| Nodes at scale 0     | 5             | 28                   |
| Nodes at scale 1     | -             | 5                    |
| Total                | 14            | 42                   |

Table 4. Final number of nodes of the wavelet controller for different choice of the Error Threshold.
In this paper an experimental comparison between Model Based Adaptive (MBA), Neural Network (NN) and Wavelet Network (WN) control has been considered. The comparative tests show that MBA is the best control approach if a sufficiently accurate model of the manipulator is known. But if the model does not take into account, for example, the grasped load, the performance deteriorates. Moreover, NN control application causes the tracking error to start from lower average value with respect to WN approach because of its pre-trained structure but application of WN outperforms that of NN in a short time. Furthermore, the final structure of WN has 14 neurons whereas NN has 100. Wavelet Networks controllers, with the new network management strategy, can learn the complex dynamics of manipulators without a pre-training phase, more efficiently and more accurately than NN and the MBA- approach.

5. Acknowledgments

This work was supported by the Italian Ministry of Education, University and Research - Research Project of National Interest “ROCOCO” (PRIN 2009).

6. References

[1] J.J. Craig, P. Hsu, S. Sastry, “Adaptive Control of Mechanical Manipulator”. In Proceedings of IEEE Int. Conf. on Robotics and Automation, San Francisco, California, 1986
[2] J. E. Slotine, W. Li, “On the Adaptive Control of Robot Manipulators”. In Proceedings of Winter Annual Meeting of the American Society of Mechanical Engineers, Anaheim, CA, USA, 1986
[3] R. Ortega, M. W. Spong, “Adaptive Motion Control of Rigid Robots: A Tutorial”. In Automatica, Vol. 25, n.6, pp. 877-888, 1989
[4] H. Berghuis, R. Ortega, “A Robust Adaptive Robot Controller”. IEEE Transactions on Robotics and Automation, Vol. 9, n.6, pp. 825 - 830, Dec. 1993
[5] F. Alonge, T. Raimondi, F. D’Ippolito “Adaptive Control Strategies for Rigid Robots to Reduce Harmonic Content of Driving Torque and Compensate Coulomb Friction”. In Proceedings of International Conference on Industrial Electronics, Control and Instrumentation, Taipei, Taiwan, August 5-9, 1996, pp. 202-207
[6] F. Alonge, F. D’Ippolito, F.M. Raimondi, “Globally convergent adaptive and robust control of robotic manipulators for trajectory tracking.” Control Engineering Practice, 12 (2004), pp. 1091–1100
[7] F. Alonge, F. D’Ippolito, E. Morgavi, F. M. Raimondi, A. Urso, “On line neuromorphic control of robotic manipulator,” Proc. of the IEEE Int. Conf. on Intelligent Engineering Systems, Vienna, Austria, September 17-19, 1998, pp.313-317
[8] J. Fu, N. K. Sinha, “An Iterative Learning Scheme for Motion Control of Robots Using Neural Networks: A Case Study”. In Journal of Intelligent and Robotic Systems, Vol. 8, pp. 375-398, 1993
[9] S.P. Chan, “A Neural Network Compensator for Uncertainness in Robotic Assembly”. In Journal of Intelligent and Robotic Systems, Vol. 13, pp. 127-141, 1995
[10] H. Gomi, M. Kawato, “Neural Network Control for a Closed-Loop-System Using Feedback-Error-Learning”. In Neural Networks, Vol. 6, pp. 933-946, 1993
[11] S.M. Prabhu, D. D. Garg, “Artificial Neural Network Based Robot Control: An Overview”. In Journal of Intelligent and Robotic Systems, Vol. 15, pp.333 - 365, 1996
[12] F. Alonge, F. D’Ippolito, E. Morgavi, F.M. Raimondi, A. Urso, “Structural Optimization Via Genetic Algorithms of Neural Networks to Control Robotic
Manipulator”. International Journal of Knowledge-Based Intelligent Engineering Systems, Vol. 2, n. 1, pp.32-41, January 1998

[13] S.A. Harp, T. Samad, A. Guha, “Designing Application-Specific Neural Networks Using the Genetic Algorithms”. In D.S. Touretzky (editor), Proceedings of IEEE Conference on Neural Information Processing Systems, Vol. 2, pp. 446-454, Morgan Kaufmann, San Mateo, 1990

[14] G.F. Miller, P. M. Todd, S. U. Hedge, “Designing Neural Networks Using Genetic Algorithms”. In J. D. Schaffer (editor), Proceedings of IEEE Conference on Neural Information Processing Systems, Vol. 2, Pag. 379-384, Morgan Kaufmann, San Mateo, 1990

[15] M. Kawato, K. Furukawa, and R. Suzuki, “A hierarchical neural-network model for control and learning of voluntary movement,” Biological Cybernetics, vol. 57, pp. 169-185, 1987

[16] W. Li and J. J. E. Slotine, “Neural network control of unknown non-linear systems,” in Proc. of American Control Conference, pp. 1136-1141, 1989

[17] W.T. Miller, F. H. Glanz and L. G. Kraft, “Application of a general learning algorithm to the control of robotic manipulators,” Int. J. of Robotic Res., vol. 6, pp.84-98, 1987

[18] K. S. Narendra, and K. Parthasarathy, “Identification and control of dynamical systems using neural network,” IEEE Trans. on Neural Networks, vol.1, no. 1, pp. 4-27, 1990

[19] F. L. Lewis, K. Liu, and A. Yesildirek, “Multilayer neural net robot controller with guaranteed tracking performances,” in Proc. of IEEE CDC, San Antonio, TX, pp. 2785-2791, 1993

[20] R.M.Sanner, J.J.Slotine, "Function Approximation, "Neural" Networks, and Adaptive Nonlinear control", IEEE, TA-7-1, 1994

[21] S.L. Cardenas-Maciel, O. Castillo, L.T. Aguilar, J.R. Castro: A T-S Fuzzy Logic Controller for biped robot walking based on adaptive network fuzzy inference system. IJCNN 2010: 1-8

[22] P. Melin, O. Castillo: Intelligent control of a stepping motor drive using an adaptive neuro-fuzzy inference system. Inf. Sci. 170(2-4): 133-151 (2005)

[23] O. Castillo, P. Melin: Intelligent adaptive model-based control of robotic dynamic systems with a hybrid fuzzy-neural approach. Appl. Soft Comput. 3(4): 363-378 (2003)

[24] Q. Zhang, and A. Benveniste. “Wavelet networks”, IEEE Trans. on Neural Networks, vol.3, pp. 889-898, 1992

[25] Q.Zhang, “Using Wavelet Network in Nonparametric Estimation,” IEEE Trans. on Neural Networks, vol.8, no.2, pp. 227-235, 1997

[26] D.L., Donoho and I.M. Johnstone, “Ideal spatial adaptation by wavelet shrinkage”, Biometrika, 81(3), 425-455, 1994

[27] M. R. Cannon, J. J. E. Slotine, “Space-frequency localized basis function networks for nonlinear system estimation and control,” Neurocomputing, 9(3), 1995

[28] R.M.Sanner, J.J.Slotine, "Structurally Dynamic Wavelet Networks for the Adaptive Control of Uncertain Robotic Systems", Proc. of 34th Conference on Decision and Control, New Orleans, LA, USA, December 1995

[29] F. Alonge, F. D’Ippolito, F. Gambino, “Control of robotic manipulator by wavelet networks,” 17th IMACS World Congress, Scientific Computation, Applied Mathematics and Simulation, Paris, France July 11 - 15, 2005