Robotica (2001) volume 19, pp. 41–51. Printed in the United Kingdom © 2001 Cambridge University Press

Neural hybrid control of manipulators, stability analysis

*N. Saadia, *Y. Amirat, *J. Pontnau and †N.K. M'Sirdi

(Received in Final Form: June 3, 2000)

SUMMARY

The design and implementation of adaptive control for nonlinear unknown systems is extremely difficult. The nonlinear adaptive control for assembly robots performing a peg-in-hole insertion is one such an example. The recently intensively studied neural networks brings a new stage in the development of adaptive control, particularly for unknown nonlinear systems. The aim of this paper is to propose a new approach of hybrid force position control of an assembly robot based on artificial neural networks systems. An appropriate neural network is used to model the plant and is updated online. An artificial neural network controller is then directly evaluated using the updated neuro model. Two control structures are proposed and the stability analysis of the closed-loop system is investigated using the Lyapunov method. Experimental results demonstrate that the identification and control schemes suggested in this paper are efficient in practice.

KEYWORDS: Constrained control; Neural networks; Force position control; Identification of system; Learning rules and Adaptive control.

1. INTRODUCTION

Neural network theory has been an active field of research over the last few years. Since the late 1980s, particular attention has been paid to the application of neural networks to design control systems,[1,2] including adaptive control problems.[3,4] In the past three decades, major advances have been carried out in adaptive identification and control of plants with unknown parameters. The identifier and controller structures are based on well established results in systems theory. The main goal of this paper concerns a new approach for the identification and control of a specific class of nonlinear unknown systems using neural networks. A typical example on which we shall focus is an assembly robot interacting with its environment. In fact, during an assembly task, the manipulator may interact with the parts to assemble, thus, its motion is constrained by the task. In this case, pure position control is ineffective since forces and positions in the task frame must be controlled simultaneously. This kind of control is called hybrid force/position control. A few hybrid control schemes have been proposed in the literature.[5–9] The major drawback of these methods is that a precise mathematical dynamic model of the manipulator must be known a priori and the control inputs must either be calculated from complex dynamic equations in real time or from a precalculated and stored array. Therefore, it is difficult to design a robust classical hybrid control of an unknown nonlinear system. This problem has been investigated using other techniques like classical control associated to a fuzzy supervision.[10] Generally, such a kind of approach doesn't take into account the system evolution and its parameters variation. The fuzzy inference system is static (the knowledge doesn't change according to the system evolution during the control). To solve constrained control of this kind of systems, a new approach based on neural networks is proposed.

In this context, the application of neural networks to dynamic system identification and control can be developed in quite a natural way, due to the adaptive nature of the learning process of neural network. Two basic approaches based on the use of neural networks have been proposed in the literature. In the first one, some design parameters are learned off line from the input-output signals and from the observation of the plant behavior in some key situations. The control can then be implemented from this learning.[11,12] In the second approach, an adaptive learning is implemented and the control input is determined on-line as the output of a neural network.[13]

In this paper, we adopt the adaptive learning approach mentioned above, from a direct learning of free motion forces and from the dynamic behavior of the system. More specifically, a Feed Forward Neural Network Model (FFNNM) is used for the estimation of free motion forces. This network learns off-line these forces using variable metric method combined with a one dimensional optimization in order to improve robustness and to accelerate convergence.[14,15] A second Feed Forward Neural Network System (FFNNS) which is used as an identifier learns off-line the dynamic of the system using variable metric method. The last Feed Forward Neural Network Controller (FFNNC), which is used as a controller, is first trained off-line to learn the input output relation from a classical hybrid force position controller using variable metric method. Then, in the second step, a real time parameters adjustment methodology using a quadratic output criterion is proposed. This paper aims at three main objectives. The two first ones, which are the most important, consist of identifying both involved forces in free motions and the plant and to design control structures using neural networks for the adaptive hybrid force position control of unknown nonlinear systems. While major advances have been made in the design of adaptive controllers for linear systems with unknown parameters, such controllers cannot be used for the global control of nonlinear systems. Consequently, the models suggested represent a first step in this direction. The third

* LIIA-IUT de Creteil, 122, Rue Paul Armangot, 94400 Vitry Sur Seine Cedex (France). E-mail: saadia@univ-paris12.fr
† Laboratoire de Robotique de Paris, 10–12, Avenue de l'Europe, 78140 Vélizy (France). E-mail: nacer@robo.uvsq.fr

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

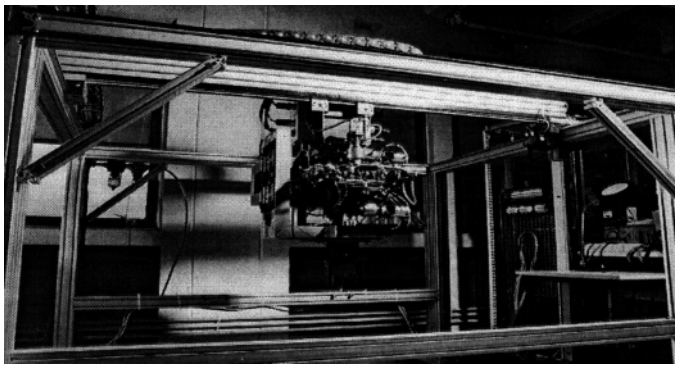

Photo A: Whole view of the flexible assembly cell.

objective is to present an adaptive method for the dynamic adjustment of the parameters based on the back-propagation algorithm. The last one concerns the local stability analysis of the two proposed structures using Lyapunov method.

The paper is organized as follows: In the first section, the experimental set up is described. After the formulation of hybrid force position control problem of an unknown non-linear system, two new hybrid structures based on neural networks are proposed in Section 3.1. Section 3.2 deals with the problem of identification of free motion forces. The initialization technique of the neural network controller is presented in Section 4.1. In Section 4.2, an on-line adjustment algorithm of the controller parameters along with the real time control architecture are presented. In Section 5, the stability conditions of the proposed FFNNC are studied. Before presenting some remarks and perspectives, experimental results are given and discussed in the last section.

2. DESCRIPTION OF THE ASSEMBLY CELL

Photo A shows the whole view of the experimental set up which consists of an assembly cell. This cell includes a 2D Cartesian robot and a six degrees of freedom parallel robot which acts as an active force controlled wrist of the Cartesian robot. In our approach, an assembly task consists of the following steps:

– Wide amplitude displacements performed by the 2D Cartesian robot in order to bring the assembly parts in a close vicinity.
– Very accurate corrective trajectory performed by the parallel robot under control of an external vision sensor in order to perform the proper location of the moving part with respect to the receptive part. The vision system measures the relative positioning of the parts to assemble.
– Assembly or final insertion phase. During this phase, contacts between parts may arise. It is therefore necessary to implement a force feedback control of the parallel robot in order to carry out this task successfully. This force feedback is needed for security constraints and to insure regularity and quality of parts.

The parallel robot presented above consists of a static part and a mobile part connected together by six actuated segments. Each segment is embedded to the static part and linked to the mobile one through a spherical joint attached to two crossed sliding plates. Theoretical studies concerning this architecture have been presented.[16–18] The C5 parallel robot is equipped with six linear actuators, each of them is driven by a DC motor. Each motor drives a ball and screw. The position measurements are done by six incremental encoders which are tied to the DC motors. In order to implement a force feedback control, the robot has been equipped with six strain gauge force sensors. Each sensor is serially displayed between the linear actuator end and the swivel (spherical joint). From the information given by the force sensors, the contact force vector is then computed. Modelisation of such a system is not obvious leading to complex models for controls. This assembly cell is intended to perform complex and various assembly tasks such as weak tolerance insertion of parts with various shapes or contour following under high dynamics. The architecture of the control system is hierarchized (see Figure 1).

3. CONTROL AND ESTIMATION-COMPENSATION

3.1 Control structure

A robotic assembly task exhibits two kinds of motion: Free motions and constrained ones. When the system works with high dynamics, transitions from one kind of motion to another lead to undesirable oscillations, thus the system may become unstable. In this case, contact forces as well as

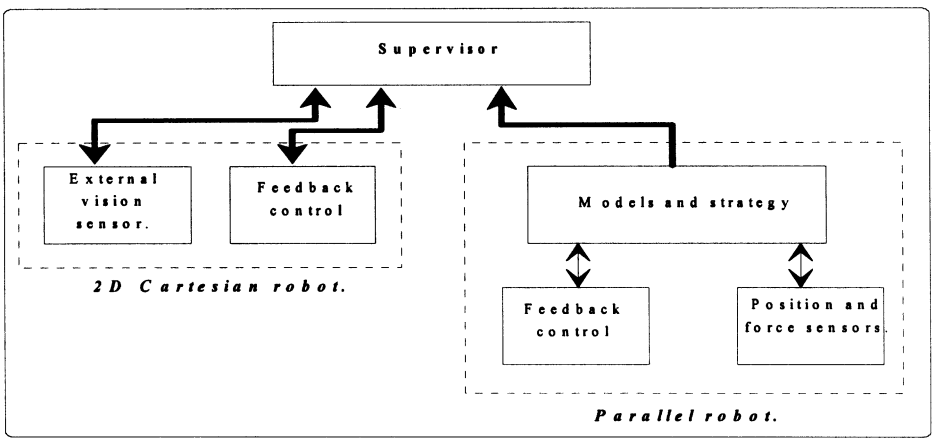

Fig. 1. Hierarchized architecture of the control system.

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

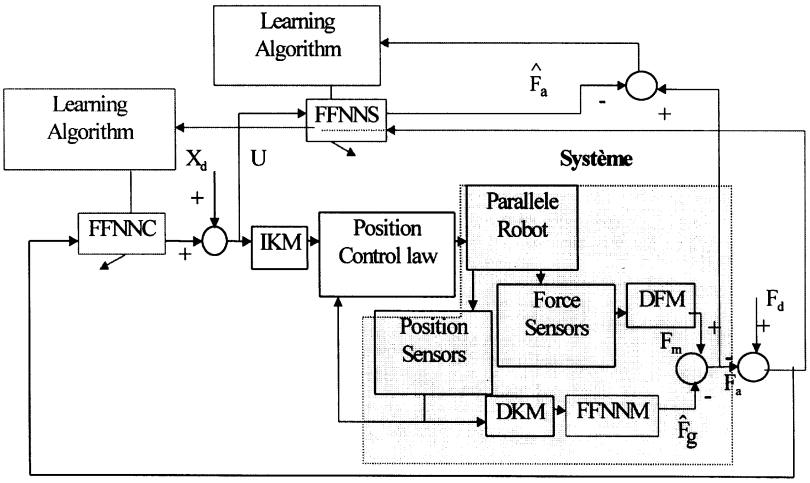

Fig. 2. Reference model Control Structure (Structure A).

position trajectories must be controlled simultaneously in order to ensure the quality of the task. Various classical structures exist, however, the implementation of most of them requires the mathematical models of plant, which are difficult to obtain in some cases. In order to satisfy all these constraints, a control structure based on an external force control scheme[19,20] is proposed. In this structure, a position trajectory (*Xd*) and a force trajectory (*Fd*) are specified where the force control loop is hierarchized with respect to the position control loop so that the force error (ΔF) is corrected from the variation of the position (ΔX) around the nominal position trajectory (*Xd*) according to the following relationship:

$$\Delta X = C(\Delta F) \tag{1}$$

where *C* represents the force control law generally called compliance model for force model. In order to control a nonlinear system, the function *C* must be nonlinear. The ability of neural networks to approximate a large class of nonlinear functions with sufficient accuracy suggests their use to compute function *C*. Figure 2 and Figure 3 illustrate the two proposed control structures.

In the first one (Structure A), three neural networks are used while in the structure B only two are needed. Due to their locations, the force sensors measure both contact and free motion forces induced by the mobile part of the parallel robot. In order to extract the contact force values from the measured ones, a FFNNM is used to estimate free motion forces. This neural network is adjusted off-line using the training data obtained from the robot's displacements in the free space. In constrained motion, a FFNNC is used as compliance controller. The implementation of this controller requires two steps (see Figure 4). First ($\alpha = 0$), the FFNNC is initialized off-line from the identification of an external force position controller. In the second step ($\alpha = 1$), the objective is to perform a real time control. In order to perform this goal, two neural networks are then implemented in the structure. The first one FFNNS acts as an identifier to learn off-line plan's dynamics. The FFNNC is implemented on-line using back-propagation method through the FFNNS. In the structure B, an associative reinforcement learning[21] to adapt on line the FFNNC's parameters is used.

In Figure 4, *Ud* is the desired output control, *Ua* the corresponding actual output, *Yd* the desired output system and *Ya* the corresponding actual output system.

3.2 Identification of free motion forces

Due to nonlinear mapping resulting from the nonlinear distribution of robot's inertia during a compliant motion, an appropriate neural network can be used to estimate the contact forces from the measured ones. This FFNNM is trained off-line using data obtained from the robot's displacements in the free space. The objective is to build a suitable model (Figure 5) which when excited by input $P(k)$, produces an output $\hat{Fg}(k)$ that approximates $Fg(k)$ such as

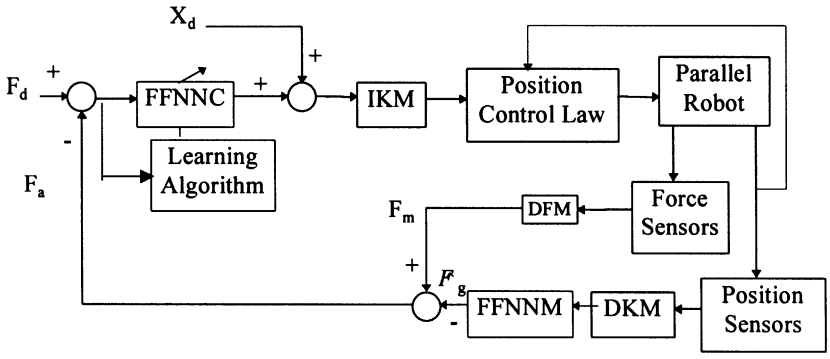

Fig. 3. Control structure with no reference model (Structure B).

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

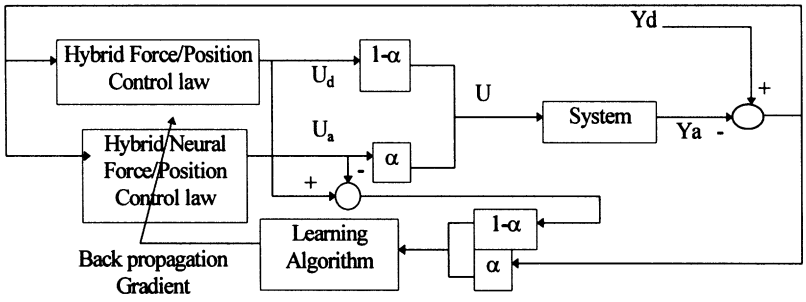

Fig. 4. Implemented architecture.

the total squared error function $E(w)$ is minimized by the network. The criterion is as follows:

$$E(w) = \frac{1}{2}(Fg - \hat{F}_g)^2 \qquad (2)$$

where Fg is the desired output network corresponding to the input pattern, $\hat{Fg}$ is the corresponding actual output. The learning process is performed off line using Quasi-Newton method combined with a one dimensional search.[22] The position and force control is performed using an external force position architecture.

4. HYBRID NEURAL NETWORK CONTROLLER

4.1 Learning hybrid force position control

The initialization of the FFNNC consists of duplicating an external hybrid force position controller. The learning structure is presented in Figure 6.

Fd, *Fa* are the desired and actual contact forces. *Fm* is the measured forces. *Fg*, $\hat{Fg}$ are the free motion forces and the estimated ones. In this structure, the force and position control laws are of PID control type.

If the weights of the networks are considered as elements of vector parameters *W*, the learning process involves the

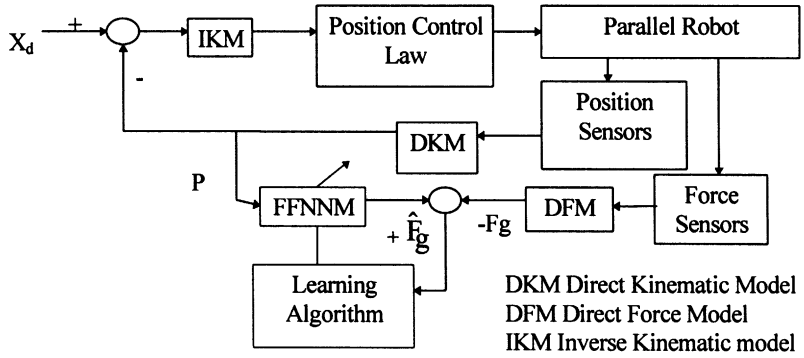

Fig. 5. FFNNM Learning a free motions forces.

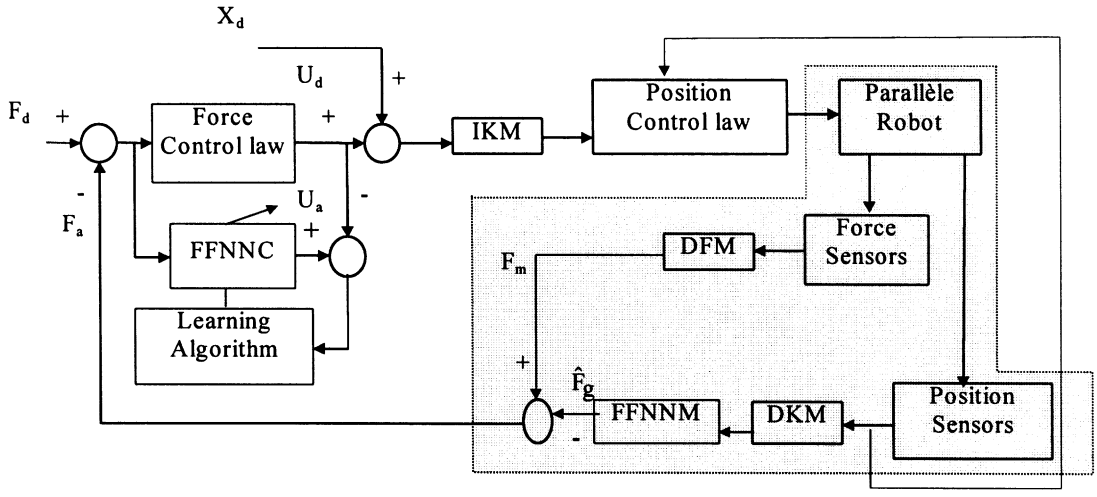

Fig. 6. FFNNC Learning Force Control.

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

determination of the vector W^* which minimises a cost function J_1 that uses all available training data. This function can be based on the output error or on state criteria for output process. In our case, the following criterion is used:

$$J_1(w) = \frac{1}{2} p = 1 \sum^{P} j = 1 \sum^{m_l} (Ud_{jp} - Ua_{jp})^2 \qquad (3)$$

where Ud_{jp} is the desired output control from the jth input, Ua_{jp} is the corresponding actual output. Quasi-Newton method[22] is applied to train the FFNNC. The parameters of the FFNNM are the constants determined in Section 3.2.

4.2 Adaptive neural network controller

In order to ensure an efficient real time force position control, the FFNNC controller is trained on-line using a dynamic back-propagation method. In dynamic control of constrained robot motion, neural networks are used, and an adaptive algorithm is then needed to adjust the parameters of the network from a given set of input output pairs. Back-propagation is the most commonly used method for this purpose in static contexts. A first dynamic back-propagation algorithm has been studied by Narendra and Parthasarathy,[24] and Williams and Zipser.[23] In a causal dynamic system, the change in parameters at time k produces a change in the output $y(t)$ for all $t \geq k$. A dynamic learning process can be formulated as follows:

$$W(k+1) = W(k) - \mu \frac{\partial J(k)}{\partial W(k)} \qquad (4)$$

where $W(k)$ is an estimation of the weight vector at time k and μ is the constant step size. The output of the network at current time k can be obtained by using only the state and input of the network at past time.

4.2.1 Structure A: Reference model Control structure. Two neural networks are used in this first structure (see Figure 2) in order to perform a dynamic real time control. The first neural network acts as an identifier (FFNNS) to learn off-line the forward dynamics of the plant. This identification step is followed by an implementation of the second neural network FFNNC which operates on-line. Dynamic back-propagation method through the FFNNS using a gradient method is employed to achieve this step.

Identification of the plant: After having learned of the classical force position control, the FFNNC is implemented in the structure presented in Figure 7. This structure includes the FFNNS which is used to identify the forward model of the plant. To train the FFNNS the following criteria is used:

$$J_s(w) = \frac{1}{2} (\hat{Fa} - Fa)^2 \qquad (5)$$

where Fa is the desired output vector and $\hat{Fa}$ the corresponding actual output. The neural network parameters are adjusted by applying Quasi-Newton method. The parameters of the FFNNM and FFNNC are the constant terms determined respectively in Section 3.2 and Section 4.1. The output of the FFNNS is the estimation of contact forces and can be written as follows:

$$\hat{F}_a = N_s(W_s, U) \qquad (6)$$

where U and W_s are respectively the input and the weight vector of the identifier FFNNS.

Real time control: In this subsection, we will consider the real time force position control of this system. So, in order to increase the performances of the control system, the FFNNC is trained on-line (Figure 2) using a dynamic back-propagation method through a FFNNS. The controller has to be updated so as to satisfy the following performance criterion:

$$J_c = \frac{e^T e}{2} \qquad (7)$$

where $e = F_d - F_a$, Fd is the desired contact force and Fa is the measured one.

The parameters of the FFNNM and FFNNS are taken constants, determined in Section 4 and Section 5.2.1, respectively. Applying the back-propagation method, the gradient of Equation (7) has to be computed:

$$\frac{\partial J(k)}{\partial W(k)} = -e^T \frac{\partial Fa(k)}{\partial U(k)} \frac{\partial U(k)}{\partial W(k)} \qquad (8)$$

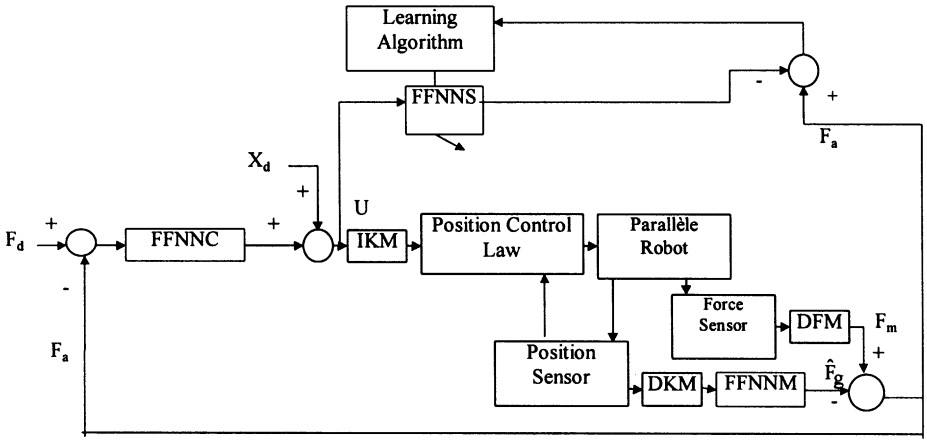

Fig. 7. Identification of the plant by FFNNS.

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

Combining Equation (6) and Equation (8), Equation (8) can be re-written to give:

$$\frac{\partial J(k)}{\partial W(k)} = -e^T \frac{\partial Ns(k)}{\partial U(k)} \frac{\partial U(k)}{\partial W(k)} \tag{9}$$

The matrix $\frac{\partial Fa(k)}{\partial U(k)}$ represents the plant dynamic effect and is used in the learning of structure A. Note that in this first structure, when the learning process is applied to the FFNNC, this information is obtained in the first learning process by the FFNNS.

4.2.2 Structure B: Control structure with no reference model: Real-time control. In this structure (Figure 3), the FFNNC is trained on-line using dynamic back-propagation with an associative reinforcement learning. In such a learning process, the learning system interacts in a closed loop with its environment. At each time step, the environment provides an input to the learning system based on which the learning system generates an action. Based on both learning systems inputs and the corresponding generated actions, the adaptive algorithm computes an evaluation or a "reinforcement". In our case, the error between the force F_a and the desired one F_d, is used as the input of the neural controller which it uses to compute a control action. The action is executed by the robot resulting in some motion of the peg. The network's output is then evaluated from the forces acting on the peg and the matrix D called "the reinforcement matrix". This matrix represents the dynamic of the interaction robot-environment. Then, the Equation (9) can be rewritten as follows:

$$\frac{\partial J(k)}{\partial W(k)} = -e^T D \frac{\partial U(k)}{\partial W(k)} \tag{10}$$

Before proceeding to the implementation of the FFNNC, its stability property must be investigated. In the next section, we will study the stability of the direct controller using the Lyapunov method for the two proposed structures.

5. STABILITY OF THE FFNNC

As shown in the first structure (Figure 2), the FFNNC is trained through the plant dynamic effect. In the second one (Figure 3), an associative reinforcement learning is implemented. We start our study with the first structure and then we will focus our interest on the case of reinforcement learning.

5.1 Stability analysis using the reference model

In this case, the back-propagation method is based on the gradient of quadratic error criterion (given by Equation (7) and is given by Equation (8).

The matrix $\frac{\partial Fa(k)}{\partial U(k)}$ which represents the plant dynamic effect, is an important factor for the stability and is needed for the back-propagation method. In our case, when the FFNNC is applied to the learning process, this information is obtained from the first learning process by the FFNNS. However, due to the adaptive nature of our control structure, this information is usually limited beforehand. Thus, it is important to determine how much information is necessary to ensure the stability. We discuss the stability from this point of view. The output U of the controller, which is used as the input of the plant, is defined by

$$U = N(W, I) \tag{11}$$

If we assume that the three-layer model and all the units have the same sigmoid function f, the Equation (6) and Equation (11) can be rewritten as follows

$$U = f(w_2 f(w_1 e + b_1) + b_2) \tag{12}$$

$$\hat{F}_a = f(w_{2s} f(w_{1s}(U + X_d) + b_{1s}) + b_{2s}) \tag{13}$$

where:

- w_1 and b_1 are the weight and threshold matrix from an input layer to a hidden layer of the FFNNC.
- w_2 and b_2 are the weight and threshold matrix from a hidden layer to the output layer of the FFNNC.
- w_{1s} and b_{1s} are the weight and threshold matrix from an input layer to a hidden layer of the FFNNS.
- w_{2s} and b_{2s} are the weight and threshold matrix from a hidden layer to the output layer of the FFNNS.
- f is the sigmoid function given by

$$f(x) = \frac{1}{(1 + \exp(-x))} \tag{14}$$

The sigmoid function of Equation (12) and Equation (13) is a vector type extension of Equation (14). The relation between the weight matrices w_1, w_2, w_1, w_{2s}, the threshold matrix b_1, b_2, b_{1s}, b_{2s} and the weight vectors W, W_s are given by:

$$w_1 = \begin{bmatrix} w_1^t 1 \\ : \\ w_1^t j \\ : \\ w_1^t n \end{bmatrix}, w_2 = \begin{bmatrix} w_2^t 1 \\ : \\ w_2^t i \\ : \\ w_2^t n \end{bmatrix}, b_1 = \begin{bmatrix} b_1^t 1 \\ : \\ b_1^t j \\ : \\ b_1^t n \end{bmatrix} \text{ and } b_2 - \begin{bmatrix} b_2^t 1 \\ : \\ b_2^t i \\ : \\ b_2^t n \end{bmatrix}$$

$$W^t = \begin{bmatrix} \ldots, & w_1^t j, & \ldots, & w_2^t i, & \ldots, & b_1^t j, & \ldots, & b_2^t i \end{bmatrix}$$

$$w_{1s} = \begin{bmatrix} w_{1s}^t 1 \\ : \\ w_{1s}^t j \\ : \\ w_{1s}^t n \end{bmatrix}, w_{2s} = \begin{bmatrix} w_{2s}^t 1 \\ : \\ w_{2s}^t i \\ : \\ w_{2s}^t n \end{bmatrix}, b_{1s} = \begin{bmatrix} b_{1s}^t 1 \\ : \\ b_{1s}^t j \\ : \\ b_{1s}^t n \end{bmatrix} \text{ and } b_{2s} = \begin{bmatrix} b_{2s}^t 1 \\ : \\ b_{2s}^t i \\ : \\ b_{2s}^t n \end{bmatrix} \tag{16}$$

$$W_s^t = \begin{bmatrix} \ldots, & w_{1s}^t j, & \ldots, & w_{2s}^t i, & \ldots, & b_{1s}^t j, & \ldots, & b_{2s}^t i \end{bmatrix}$$

As the back-propagation method is derived from the quadratic error J defined by Equation (7), we choose the following extended quadratic error $V(e)$ as Lyapunov function:

$$V(e) = J = \frac{e^T P e}{2} \tag{17}$$

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

where P = I (P Symmetric Positive definite chosen here as the identity matrix). Both stability of the dynamic system and the convergence of the FFNNC are guaranteed when $\frac{\partial V}{\partial t} \leq 0$ since the Lyapunov function candidate is a positive definite function as shown in Equation (17). This condition is investigated using Equation (17), so:

$$\frac{\partial V}{\partial t} = \frac{\partial(\frac{e^T Pe}{2})}{\partial t} = e^T P\dot{e} \tag{18}$$

where

$$\dot{e} = \frac{\partial e}{\partial t} = \frac{\partial e}{\partial U}\frac{\partial U}{\partial t} \tag{19}$$

When the back-propagation method is applied, then:

$$\Delta W = -\mu e^T \frac{\partial e}{\partial U}\frac{\partial U}{\partial W} \tag{20}$$

where μ is a positive parameter for the learning gain tuning. Since the relation given by Equation (20) obtained from the back-propagation method cannot be substituted in Equation (19), we consider a small deviation from the equilibrium point, which corresponds to local stability analysis using equation (11). The control input U is given by:

$$\Delta U = \left(\frac{\partial N}{\partial W}\right)^T \Delta W + \left(\frac{\partial N}{\partial I}\right)^T \Delta I \tag{21}$$

where I and N are respectively the input and output of the FFNNC. Using relation (21), Equation (18) can be rewritten as follows:

$$\frac{\partial V}{\partial t} = \varepsilon\left(\frac{\partial N}{\partial W}\right)^T \Delta W + \varepsilon\left(\frac{\partial N}{\partial I}\right)^T \Delta I \tag{22}$$

with

$$\varepsilon = e^T P\frac{\partial e}{\partial U} \text{ and } \Delta I \simeq e$$

(small deviation from the equilibrium point) (23)

Combining Equations (22) and (20), the following equation can be obtained:

$$\frac{\partial V}{\partial t} = e^T(-R+T)e \tag{24}$$

where

$$R = \mu\left(P\frac{\partial e}{\partial N}\right)\left(\frac{\partial N}{\partial W}\right)^T\left(\frac{\partial N}{\partial W}\right)\left(P\frac{\partial e}{\partial N}\right)^T \tag{25}$$

$$T = P\left(\frac{\partial e}{\partial N}\right)\left(\frac{\partial N}{\partial e}\right)^T \tag{26}$$

As the stability is guaranteed when $\frac{\partial V}{\partial t} \leq 0$, we investigate the sign of each term of Equation (24). The first term is always positive ($R \geq 0$). The stability is then ensured if the sign of the second term is negative ($T \leq 0$). The matrix T depends on the identifier and controller weight vector which are strongly related to both the initial value and the learning technique. However, even if the negative definiteness condition is not satisfied, the system may be stable when the norm of the positive defined matrix R is larger than the norm of the matrix T. The norm of the positive definite matrix R depends strongly on both the turning parameter μ and the norm of the weight matrices. (Equation 25). Therefore, we can choose large values for μ and a large $\frac{\partial e}{\partial N}$ and $\frac{\partial N}{\partial W}$ norms, the system can be made stable. However, if we consider a discrete type of back-propagation method, there is an upper bound of $\|\Delta W\|$ which leads to instabilities when ΔW becomes large. In our case, the learning parameter μ does not change during the process and therefore does not affect the stability of the whole system. This analysis shows that the stability of this structure depends on both the initial values of the weights vector and the gain tuning parameter.

5.2 Stability analysis using the reinforcement learning

In this second structure (Figure 3), the associative reinforcement learning is used. In our case, first, the error between the sensed peg force F_a and the desired one F_d, is used as the input to the neural controller. The action computed by the neural controller is executed by the robot resulting in some motion of the peg. The network's output is then evaluated from the forces acting on the peg and the reinforcement matrix D which represents the dynamics of the robot environment interaction. Then the Equation (24) can be rewritten as follows:

$$\frac{\partial V}{\partial t} = e^T(-\acute{R}' + T')e \tag{27}$$

where

$$R' = \mu(PD)\left(\frac{\partial N}{\partial W}\right)^T\left(\frac{\partial N}{\partial W}\right)(PD)^T \tag{28}$$

$$T' = (PD)\left(\frac{\partial N}{\partial e}\right)^T \tag{29}$$

Since the stability is guaranteed when $\frac{\partial V}{\partial t} \leq 0$, D must be chosen such that the following inequality (Equation (30)) hold:

$$(PD)\left[-\mu\left(\frac{\partial N}{\partial W}\right)^T\left(\frac{\partial N}{\partial W}\right)(PD)^T + \left(\frac{\partial N}{\partial e}\right)^T\right] \leq 0 \tag{30}$$

one solution for this condition is given by:

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

$$(PD)^T = \frac{\left(\frac{\partial N}{\partial e}\right)^T \left[\left(\frac{\partial N}{\partial W}\right)^T \left(\frac{\partial N}{\partial W}\right)\right]^{-1}}{\mu} \qquad (31)$$

As it can be seen for this learning type, by choosing appropriate values of both matrix D and tuning parameters and controllers parameters the local stability of the closed loop system can be guaranteed.

6. EXPERIMENTATION

In order to demonstrate the validity of the adaptive control approach described above, the second control structure has been implemented on the flexible cell of the LIIA laboratory described in Section 2. The aim is to perform assembly tasks with high accuracy for different shapes of parts. For each experimentation, two parameters are defined: play margin P_m and insertion velocity V_i. For free motion forces estimation, a network made up of three layers, six input-layer neurons to represent the position and the orientation of the end-effector, twenty neurons in single hidden layer and six output layer neurons for the corresponding gravity force vector. To learn the control, the same neural network architecture has been used except that the six input-layer neurons represent the difference betwen the desired and actual force vector. The six output-layer correspond to the control level. The hyperbolic tangent is used as nonlinear activation function for the single hidden layers while a linear function is chosen for the output layers. The initial weights are chosen randomly within the range of -1 to $+1$. The first step consists of the implementation of an external force position control in order to initialize the FFNNC. Figure 8 shows the convergence behavior of the FFNNC. We obtain at the 2000 epoch, a sum squared error equals to 0.001, which represents a good result. From these results, it can be seen that a substantial time saving is obtained in training by employing the quasi-Newton method.

In the second step, the FFNNC is implemented on-line. The associative reinforcement learning is used to adapt on line the neural controller parameters. Figure 9 shows the time evolution of the obtained contact force vector components (Fx, Fy, Fz, Mx, My, Mz) with an insertion velocity of

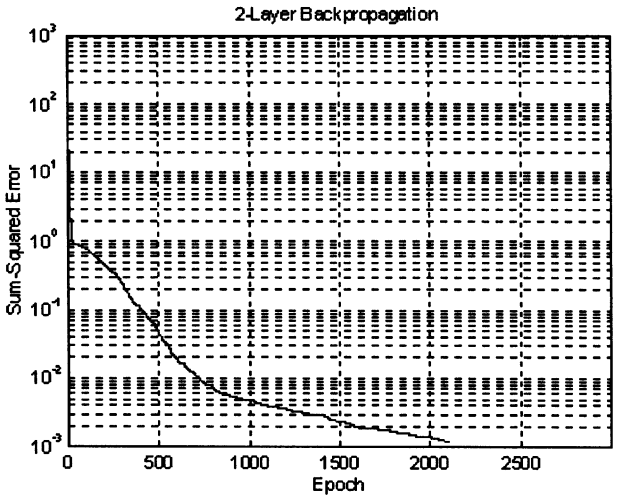

Fig. 8. Off-line training of FFNNC.

15 mm/s and two parts of cylindrical shapes. The component Fx exhibits a peak which corresponds to the first contact between the peg and the receptive part. The remaining components of the contact force vector decreases and exhibits weak amplitude oscillations around the desired force vector. Figure 10 shows the time evolution of the obtained contact force vector components with an insertion velocity of 10 mm/s and two parts of rectangular section. It can be observed that components Fy and Fz exhibit peaks during the contact and stabilize, respectively, around 10 N and 0 N. In the same way, the Mz momentum component exhibits a peak of 5 Nm during contact, then decreases to stabilize close to zero. These results show that the momentum components are well corrected by the control structure.

Similarly, Figure 11 and Figure 12 show the time evolution of the different components of the moving part trajectory. When contact forces between parts arise, along a direction, the corresponding position exhibits peak then decreases and oscillates around the desired value.

These results show that the proposed structure leads to a satisfactorily dynamic behavior from force contact minimization point of view, even if the system dynamic is unknown. Moreover, an experimental comparison between

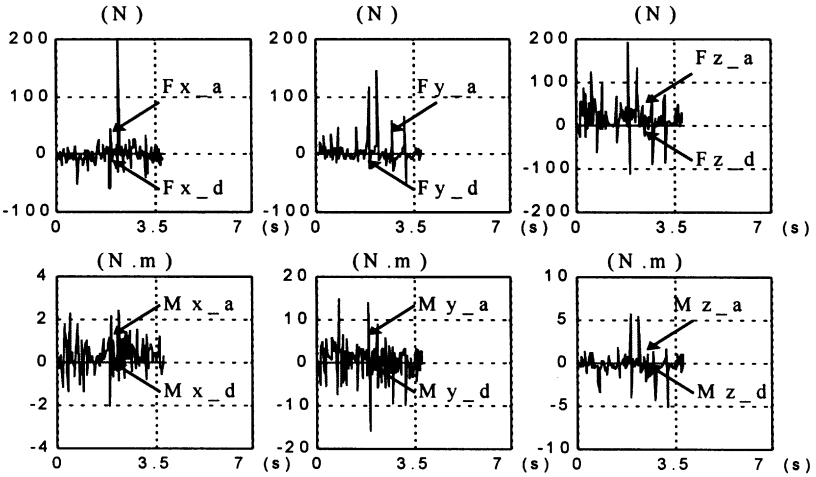

Fig. 9. Adaptive neural controller (Vi = 15 mm/s, Pm = 0.1 mm, cylindric section part).

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

external force position controller and the proposed hybrid neural controller has been done. In fact, we have performed insertion task using external force position control with insertion velocity of 10 mm/s for two parts of rectangular shapes. Figure 13 shows the time evolution of components (Fx, Fy, Fz, Mx, My, Mz) of contact force vector. The component Fx exhibits a peak which indicates occurrences of contact between the peg and the receptive part. According to the insertion velocity, instabilities may occur. In assembly task involving rectangular section part, some vector components show non zero residual values. Figure 14 illustrates the time evolution of the position vector and show the same behavior. Compared with an external force position control (Figure 13 and Figure 14), adaptive neural approach (Figure 11 and Figure 12) exhibits better performances due to its adaptive feature and reveals that the proposed structure ensures better dynamic performances than the external force position control. Due to its adaptive feature, the proposed approach is easier to adjust for various tasks parameters (insertion velocity, play margin). Indeed, according to the insertion velocity value, instability may occur when an external force position control is implemented.[25,26]

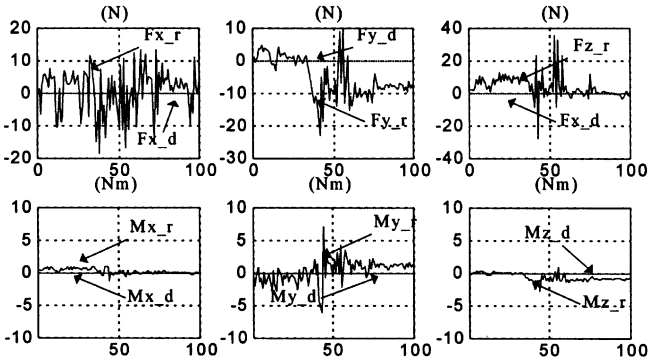

Fig. 10. Adaptive neural controller (Vi = 10 mm/s, Pm = 0.1 mm, rectangular section part).

7. CONCLUSION

In this paper, a neural approach to solve the control of constrained nonlinear dynamic systems and identification of free motion forces is proposed. Opposite to classical force position control, the architecture of the proposed structure does not need any mathematical representation of dynamic

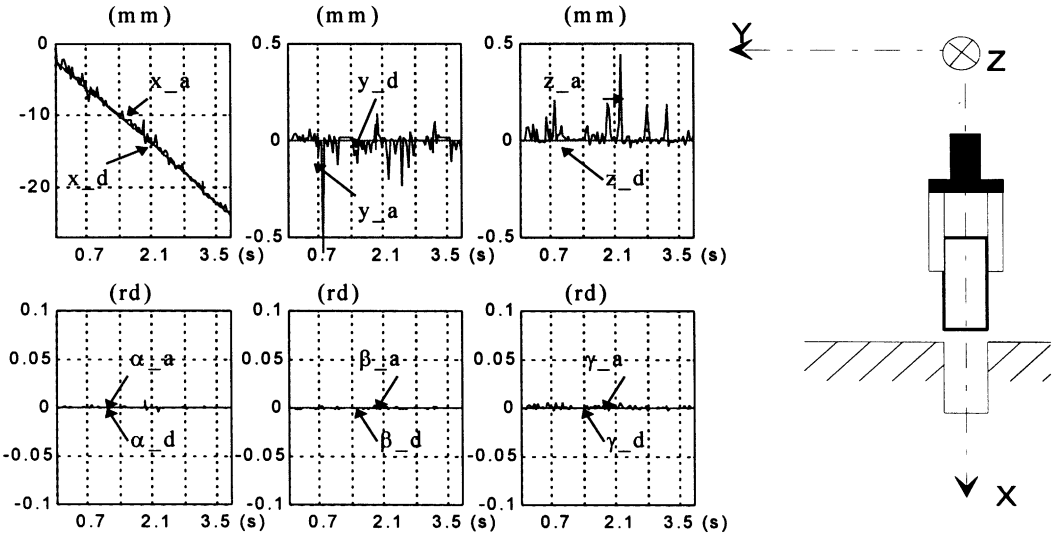

Fig. 11. Adaptive neural controller (Vi = 15 mm/s, Pm = 0.1 mm, cylindric section part).

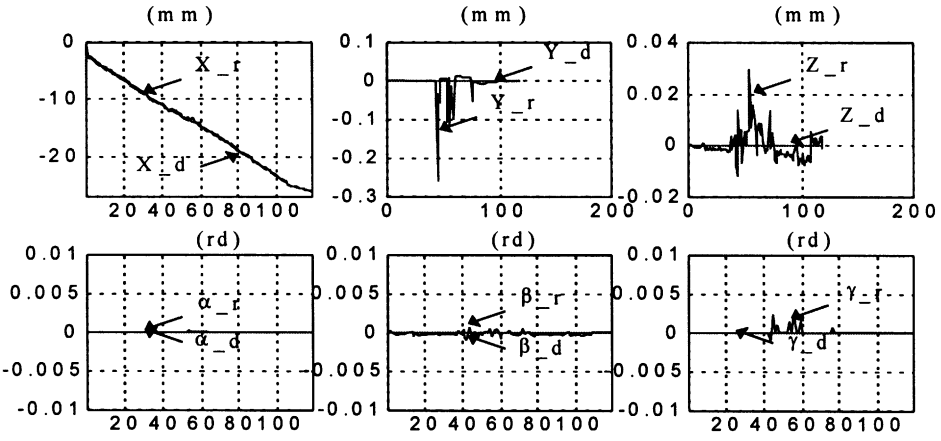

Fig. 12. Adaptive neural controller (Vi = 10 mm/s, Pm = 0.1, rectangular section part).

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

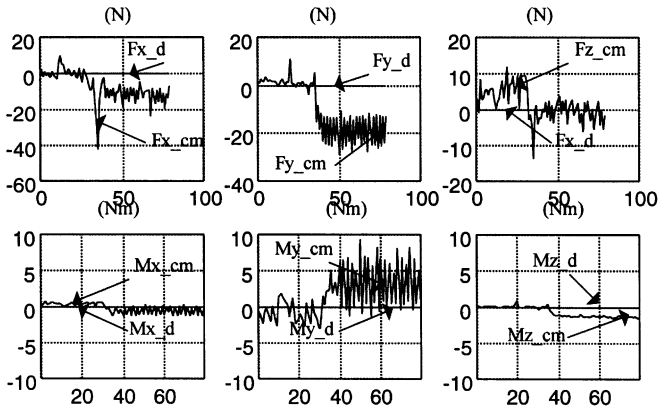

Fig. 13. External controller (Vi = 10 mm/s, Pm = 0.1 mm, rectangular section part).

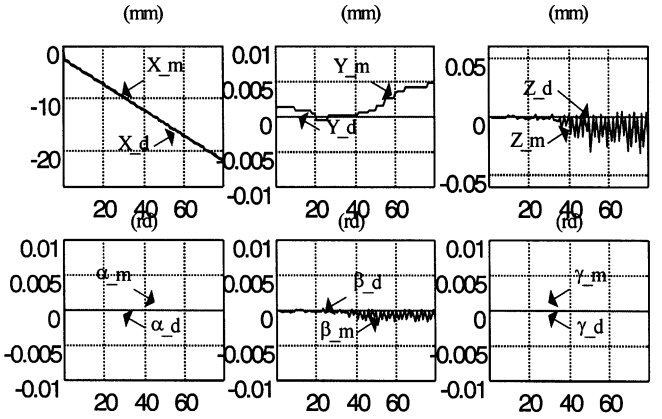

Fig. 14. External controller (Vi = 10 mm/S, Pm = 0.1 mm, rectangular section part).

behavior of the robot. In the specific models given, the difference between the measured forces and the identified ones by a FFNNM are used as inputs to FFNNC. Methods for the adjustment of parameters in generalized neural networks are treated and the concept of dynamic back-propagation is introduced in this context to generate partial derivatives of a performance index with respect to on line adjustable parameters. Two techniques of back-propagation are presented: the first one is implemented through the reference model (FFNNS) while the second one uses the reinforcement learning. The stability analysis of the proposed structures led us to keep the second one. In order to show the validity of this approach, a peg in a hole insertion with complex shapes and weak tolerances has been performed. The analysis of experimental results reveals a satisfactory dynamic behavior, from force contact minimization point of view. The proposed hybrid neural control insures better dynamic performances than the external force position control in an assembly task. These promising results lead us to increase investigations in this control approach for other kinds of tasks such as the control of flexible structures.

References

1. D. Psaltis, A. Sideris and A. Yamamura, "A multilayered network controller", *IEEE Control Syst. Mag.* 17–21 (1988).
2. P. Antsakis, "Neural Networks in Control Systems", *IEEE Control Syst. Mag.* **10**, 3–5 (1990).
3. D. H. Nguyen and B. Windrow, "Neural network for self-learning control systems", *IEEE Control Syst. Mag.* **10**, 18–23 (1990).
4. J.H. Sira-Ramirez and S.H. Zak, "The adaptation of perceptrons with application to inverse dynamics identification of unknown dynamic systems", *IEEE Trans. Syst. Man. Cybern.* **SMC-21**, 634–643 (1991).
5. M.T. Mason, "Compliance and force control for computer controlled manipulators", *IEEE Trans. Syst. Man. Cybernet.* **SMC-11**(6), 418–432 (1981).
6. R.C.P. Paul and B.E. Shimano, "Compliance and control", *Proc. Joint Automatic Control Conference*, Purdue University (July, 1976) pp. 694–699.
7. M.H. Raibert and J.J. Craig, "Hybrid position/force control of manipulators", *J. Dyn. Syst. Meas. Control* **102**, 126–133 (1982).
8. D.E. Whitney, "Force feed-back control of manipulator fine motion", *J. Dyn. Syst. Meas. Control* 91–97 (June, 1997).
9. O. Khatib, "A unified approach to motion and force control of robot manipulators", *IEEE Trans. On Robotics and Automation* **1**(3), 43–53 (1987).
10. P. Begon, "Commande des robots paralleles rapides: Application au Robot HEXA", *Master's Thesis* (Montpelier II University, France, Juin, 1995).
11. M. Tokita, T. Mitsuka, T. Fuikuda and T. Kurihara, "Force control of robot manipulator by neural network (experimental results and their evaluation of one degree-of-freedom manipulator)", *J. Robot Soc.* **8**, 52–59 (Japan, 1990).
12. M. Tokita, T. Mitsuka, T. Fukuda and T. Kurihara, "Force control of robot manipulator by neural network (control of one degree-of-freedom manipulator)", *J. Robot. Soc.* **7**, 47–51 (Japan, 1989).
13. K. Watanabe, T. Fukuda and S.J. Tzafestas, "An adaptive control for CARMA systems using linear networks", *Int. J. Control* **56**, 483–497 (1992).
14. A. Ramdane-Cherif, "Optimization schemes for learning the forward and inverse Kinematic Equations with Neural Network", *IEEE Int. Conf. on Neural Networks*, Perth, Australia (Sept., 1995) pp. 2732–2737.
15. L.E. Scales, *Introduction to Non-Linear Optimization* (Springer-Verlag, New York, 1985).
16. Y. Amirat, F. Artigue and J. Pontnau, "Six degrees of freedom parallel robots with C5 links", *Robotica* **10**, Part 1, 35–44 (1992).
17. M.Y. Amirat, "Contribution à la commande de haut niveau de processus robotisés et à l'utilisation des concepts de I'IA dans l'interaction robot-environnement", *DHDR Diploma* (Paris XII University, France, Jan., 1996).
18. E. Dafaoui, Y. Amirat, J. Pontnau and C. Francois, "Analysis and design of a six DOF parallel manipulator. Modelling singular configuration and workspace", *IEEE Trans. on Robotics and Automation* **14**(1), 78–92 (Feb. 1998).
19. V. Perderau, "Contribution à la commande hybride force-position", *Master's Thesis* (Paris VI University, France, 18 Feb., 1991).
20. P. Fraisse, "Contribution à la commande robuste position-force des robots manipulateurs à architecture complexe. Application à un robot à deux bras", *Master's Thesis* (Montpelier II University, France, 17 Feb., 1994).
21. A.G. Barto and P. Anandan, "Pattern recognizing stochastic learning automata", *IEEE Transactions on Systems, Man, and Cybernetics* **15**, 360–375 (1985).
22. N. Saadia, Y. Amirat, J. Pontnau and A. Ramdane-Cherif, "Hybrid Force Position Control of Nonlinear Systems using Neural Networks", *Sec. ECPD Int. Conf. on Adv. Rob. Int. Aut. and Act. Sys.* Vienna, Austria (September 26–28, 1996) pp. 68–73.
23. R. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks", *Neural Computation* **1**, 270–280 (1989).

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press

24. K.S. Narendra and K. Parthasarathy, "Identification and control of dynamic systems using neural networks", *IEEE Trans. Neural Networks* **NN-1**(1), 4–27 (1990).
25. N. Saadia, Y. Amirat, J. Pontnau and N.K. M'Sirdi, "Experimental comparison between External Force Position controller and New Hybrid Neural Networks controller", *Third EPCD Int. Conf. on Adv. Rob. Int. Aut. and Act. Sys.,* Bremen Germany (September 15–17, 1997) pp. 68–73.
26. N. Saadia, "Contribution à la commande hybride force-position des robots compliants selon une approche neuronale", *Master's Thesis* (Paris XII University, France, Dec. 1997).

https://doi.org/10.1017/S0263574700002885 Published online by Cambridge University Press