Robotica (1998) volume 16, pp. 59–73. Printed in the United Kingdom © 1998 Cambridge University Press

Evaluation of parametric and nonparametric nonlinear adaptive controllers

E. Burdet*† and A. Codourey*

(Received in Final Form: April 26, 1997)

SUMMARY

In this paper, nine adaptive control algorithms are compared. The best two of them are tested experimentally. It is shown that the Adaptive FeedForward Controller (AFFC) is well suited for learning the parameters of the dynamic equation, even in the presence of friction and noise. The resulting control performance is better than with measured parameters for any trajectory in the workspace. When the task consists of repeating the same trajectory, an adaptive look-up-table MEMory, introduced and analyzed in this paper, is simpler to implement and results in even better control performance.

KEYWORDS: Adaptive controllers; Nonlinear; Control algorithms; Trajectories; Parameters.

1. INTRODUCTION

Many new manipulators, for example fast and light parallel manipulators,[1,2] have highly nonlinear dynamics. These dynamics have to be compensated in order to increase the precision of such robots. The rigid body model equation is mostly used as compensator. One generally determines its parameters by calculating them from the CAD map. If a higher precision is required, one normally disassembles the manipulator and measures the lengths, masses and moments of inertia of the different mechanical parts.

For about 10 years, many schemes have been proposed for learning the nonlinear dynamics of manipulators during the movements in a non-invasive way, and compensating for it.[3–13] These schemes arose not only from robotics, but also from the field of neural networks, motivated by the adaptive properties of human movements.[14] In these schemes, the robot is controlled using linear joint feedback controllers, and the error during motion is used to determine a model of the dynamics. The mathematical properties of these adaptive controllers have been extensively studied.[15,16]

Despite this intense activity, only a few implementations have been performed, and to our knowledge there exists no industrial application. However, the results of some implementations are very promising.[13,17] Our aim is to compare different nonlinear adaptive controllers developed so far and to evaluate their practical implications.

An adaptive control scheme is worth using instead of the usual dynamic compensation technique (i.e. using the rigid body model equation and measuring its parameters) only if

- it is simple to implement,
- the control is stable and robust to noise,
- it is able to identify parameters varying over time, e.g. friction,
- the learning is fast, and
- the resulting control performance is similar or better than with the conventional technique.

In this paper, we study how representative adaptive control schemes proposed in robotics and neural network fulfill these conditons. Two tasks will be considered: (i) improving the control of arbitrary trajectories; and (ii) improving the control along a repeated trajectory. It is obvious that the first task is useful, but the second is also of great practical interest: most manipulators used for assembly or paletisation repeat a single movement.

As many effects must be neglected theoretically or in simulations, experiments will be performed on the two best schemes. In particular, their ability for identifying the friction and their robustness to noise will be examined. The experiments are performed using a parallel manipulator (Figure 1) with highly nonlinear dynamics and coupling between the axes.

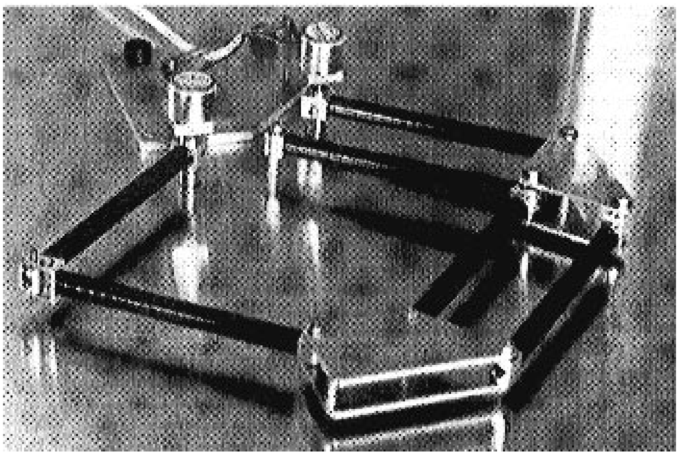

Fig. 1. Photograph of the manipulator used to test the AFFC and MEMory controllers. Each of the 6 limbs is 0.12 [m] long.

* Institute of Robotics, ETH-Zurich (Switzerland).
† School of Kinesiology, Simon Fraser University, Burnaby, B.C. V5A 1S6 (Canada), Email: e.burdet@ieee.org, WWW: http://www.ifr.mavt.ethz.ch

The paper is organized as follows: in section 2, nine control algorithms will be briefly presented, compared and evaluated, based on results from the literature. The best algorithm for compensating the dynamics for arbitrary trajectories, the Adaptive FeedForward Controller (AFFC), will then be tested experimentally in section 3. The best algorithm for compensating the dynamics along a repeated trajectory, the MEMory, will be examined in section 4. Its mathematical properties will first be shown, and experiments will be performed. Combining MEMory and AFFC will also be investigated.

2 COMPARISON OF CONTROL SCHEMES

Most industrial manipulators are controlled using linear feedback controllers. The simplest possibility is to use a proportional derivative controller at each joint. This gives the control law

$$\tau \equiv \tau_{FB} \equiv Ds, \quad s \equiv \dot{e} + \Lambda e, \tag{1}$$

where $e \equiv q_d - q$ is the error vector between the desired and actual joint positions, D and Λ are positive definite quadratic matrices (Figure 2).

In order to improve the trajectory tracking, the manipulator dynamics have to be compensated. In the adaptive control paradigm, this compensation is learned during the motion.

Most adaptive controllers are determined by the three following features:

- The *model of the inverse dynamics* used in order to compensate for the dynamics.
- The *control structure*, i.e. how the input, feedback and output are related together.
- The *adaptation criterion* used to modify the model and improve the control.

In the subsection 2.1, the different models, control structures and adaptation criteria will be described. The controllers will be defined in subsection 2.2 and evaluated in subsection 2.3.

2.1. Characteristics of nonlinear adaptive control schemes

2.1.1. Possible models of the inverse dynamics. All the models are determined by a *parameters vector* or *weights vector* w, and can be modified by changing the value of its coefficients. The *learning law* is

$$\hat{w}_{\text{new}} \equiv \hat{w}_{\text{old}} + \Delta w, \tag{2}$$

where Δw is the *correction term.*

There are principally two kinds of models: the *parametric* one, established using physical knowledge and defined by a few parameters which have to be identified,

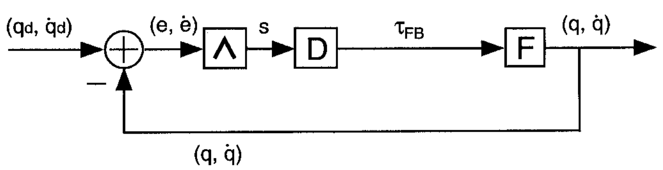

Fig. 2. Functional scheme of a *feedback controller.* $e = q_d - q$ is the error between desired and actual joint positions, $s \equiv \dot{e} + \Lambda e$, F is the dynamics of the plant, τ the torque, D and Λ are positive quadratic definite matrices.

and the *nonparametric* one, in which the function from the state $(q, \dot{q}, \ddot{q})$ to the corresponding torque, or the torque along a trajectory, is stored in a memory.

Parametric models: The most obvious parametric model of the dynamics of robot manipulators is the rigid body model equation. The most important features of this model are presented in this paragraph.

The dynamic equation of a system of rogid bodies can be described by the equation

$$\tau = H(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q), \tag{3}$$

where q is the vector of joint angles.[16] In this equation, $H(q)\ddot{q}$ is the inertia term, $C(q, \dot{q})\dot{q}$ the torque corresponding to Coriolis- and centrifugal forces, and $G(q)$ the gravity term. If one considers that each joint i has a friction $b_i(\dot{q}_i)$ modeled by

$$b_i(\dot{q}_i) \equiv b_{i1}\dot{q}_i + b_{i2}\,\text{sign}\,(\dot{q}_i), \tag{4}$$

then the system's dynamics can be described by

$$\begin{aligned} \tau &= H(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + B(\dot{q}), \\ B(\dot{q}) &\equiv (b_1(\dot{q}_1), \ldots, b_N(\dot{q}_N))^T. \end{aligned} \tag{5}$$

The first term of equation (4) is named *viscous friction* and the second term *kinetic friction.*[18]

Equation (5) can be written as a linear function

$$\tau \equiv \Psi(q, \dot{q}, \ddot{q})w \tag{6}$$

of a *parameters vector* or *weights vector* w.[10,19,20] The coefficients of this vector are (nonlinear) combinations of the lengths, masses and moments of inertia of the different links and joints. The coefficients of the *dynamic matrix* $\Psi(q, \dot{q}, \ddot{q})$ are (nonlinear) functions of q, $\dot{q}$ and $\ddot{q}$. As will be seen below, this linear representation of the dynamic equation permits to identify its parameters on a simple way.

Nonparametric models: For storing the function from a state $(q, \dot{q}, \ddot{q})$ to the corresponding torque or the torques along a trajectory, any memory with continuous input and output can be used. Possible choices are then a simple look-up-table,[4,6,21] radial basis functions networks,[22] a CMAC neural network,[3] a feedforward neural network.[23]

2.1.2. Control structures. Amongst the many nonlinear control structures proposed so far, two typical and successful structures are the feedforward and the computed torque.[4,24] We will now describe these structures.

The *feedforward structure* is depicted in Figure 3. In this structure, the model of the inverse dynamics is used in order to compute the torques necessary to realize the desired trajectory. The corresponding control law is

$$\tau = \hat{F}^{-1}(q_d, \dot{q}_d, \ddot{q}_d) + \tau_{FB}, \tag{7}$$

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

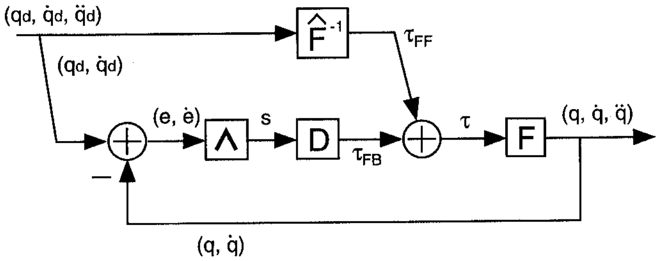

Fig. 3. Functional scheme of a *feedforward controller*. $e = q_d - q$ is the error between desired and actual joint positions, $s \equiv \dot{e} + \Lambda e$, F the dynamics of the plant, $\hat{F}^{-1}$ a model of the inverse dynamics, τ is the torque, D and Λ are positive definite matrices.

where $\hat{F}^{-1}$ is a model of the inverse dynamics.* In this scheme, the linear feedback τ_{FB} is used only to correct for unmodeled dynamics.

In the *computed torque control structure*, not the torque, but the acceleration is corrected by linear feedback controllers (Figure 4). The control law is then:

$$\tau = \hat{F}^{-1}(q, \dot{q}, \ddot{q}^*), \quad \ddot{q}^* \equiv \ddot{q}_d + Ds, \qquad s \equiv \dot{e} + \Lambda e, \quad e \equiv q_d - q. \tag{8}$$

This scheme presents the advantage that the feedback control is performed homogeneously, i.e. does not depend where in the space $(q, \dot{q}, \ddot{q})$ the control is performed.

Several variations of these two control structures will be encountered in the description of the nine controllers below.

2.1.3. Different adaptation criteria. The model of the inverse dynamics can be modified in at least two different ways: directly, by improving the model identification or indirectly, by reducing the tracking error. Thus, the adaptation criterion is either the minimization of the identification error (corresponding to the "Model-Reference Adaptive Control"[15] or to the "inverse dynamic learning"[25]) or the minimization of the tracking error (corresponding to the "Self-Tuning Control"[15] or to the "feedback error learning"[25]).

Let us now give the equations for the correction terms in the case that the dynamic equation is used as a

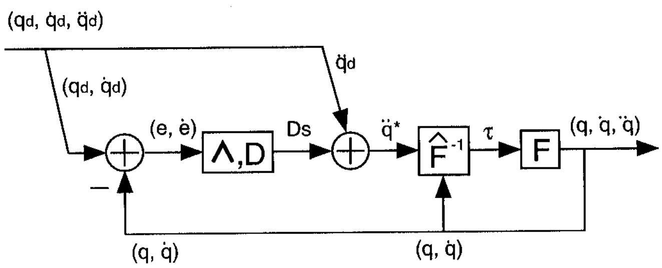

Fig. 4. Functional scheme of the computed torque control. $e = q_d - q$ is the error between desired and actual joint positions, $s \equiv \dot{e} + \Lambda e$, F represents the plant dynamics, $\hat{F}^{-1}$ a model of its inverse dynamics, D and Λ are symmetrical positive quadratic definite matrices, and τ the motor torques.

* Over a variable indicates that this variable is an estimation of the real one.

(parametric) model. In this case, the identification with a least-square estimator corresponds to the minimization of

$$\int_0^t |e|^2 \, dr, \quad e \equiv \tau - \Psi \hat{w}, \tag{9}$$

where τ is the actual torque. In order to approximate τ, the trajectory is driven using only a linear controller and the corresponding feedback is used as τ. Minimization of equation (9) leads to the correction term

$$\Delta w \equiv -P(t)\Psi^T(t)\tau_{FB}, \tag{10}$$

with

$$P(t)^{-1} \equiv \int_0^t \Psi^T \Psi \, dr,$$

$$\tau_{FB} \equiv Ds, \quad s \equiv \dot{e} + \Lambda e.$$

When the dynamic equation is used as model, the minimization of tracking error is realized using the gradient descent of the tracking error function

$$E = (\tau - \tau_{FF})^2, \tag{11}$$

where τ is in the linear form (6). This gives the correction term

$$\Delta w \equiv \Gamma \Psi^T s, \tag{12}$$

where Γ is a positive definite matrix composed of the *learning factors*. Obviously equations (10) and (12) are similar, but the calculation of equation (12) requires much less computation than those of equation (10).

2.2. *Definition and properties of the controllers*

The different controllers are defined in Table I. These are, in our opinion, representative parametric and

Table I. Description of the different controllers compared in this section. The first six controllers on the top use the rigid body model as (parametric) model of the inverse dynamics, the others a nonparametric one

	Inverse dynamic model	Control structure	Adaptation criterion minimization of the error by:
L-S[4,10]	rigid body equation	feedforward	model
Cr88[6]	ridig body equation	computed torque	tracking
Sl87[12]	rigid body equation	modified feedforward	tracking
AFFC[11,25]	rigid body equation	feedforward	tracking
Sl91[15]	rigid body equation	modified comp. torque	tracking
CAC[15]	rigid body equation	modified comp. torque	tracking & model
CMAC[3]	CMAC neural net	feedforward	model
J88[30]	feedforward neural net	feedforward	tracking & model
MEM	table	feedforward	tracking

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

nonparametric nonlinear adaptive controllers. We describe them below in more detail:

2.2.1. Parametric controllers. LS The least-square scheme is the first non-invasive scheme that was proposed for identifying the parameters of the dynamics equation.[4,10] It corresponds to the equations (9) and (10). Both the feedforward control structure (equation (7)) and computed torque control structure (equation (8)) can be used in this scheme. It has been shown theoretically that this scheme is robust to noise.[15]

Cr88: Craig[6] introduced the first adaptive controller based on the minimization of the tracking error. The control structure used is the computed torque, which gives the control law

$$\tau = \hat{F}^{-1}(q, \dot{q}, \ddot{q}^*) \equiv H(q)\ddot{q}^* + C(q, \dot{q})\dot{q} + G(q) + B(\dot{q}) \equiv \Psi(q, \dot{q}, \ddot{q}^*)w, \quad \ddot{q}^* \equiv \ddot{q}_d + Ds. \tag{13}$$

The correction term is

$$\Delta w \equiv \Gamma \Psi^T(q, \dot{q}, \ddot{q}^*)H^{-1}\tau_{FB}. \tag{14}$$

The additional H^{-1} (additional relative to equation (12)) comes from the fact that in the computed torque structure, the linear feedback providing stability acts at the acceleration level and not at the torque level. The calculation of H^{-1} requires that the corresponding starting parameters be close to the real one. Exponential stability during learning has been proved for this controller.[6]

Sl87: Slotine and Li designed the controller Sl87[12] in order to deal with the two drawbacks of Cr88: Sl87 can use arbitrary starting values for the parameters, and the actual acceleration is not required. The control law is given by

$$\tau = \hat{H}(q)\ddot{q}_r + \hat{C}(q, \dot{q})\dot{q}_r + \hat{G}(q) + \hat{B}(\dot{q}) + Ds \equiv \Psi(q, \dot{q}, \dot{q}_r, \ddot{q}_r)\hat{w} + Ds, \tag{15}$$

with the *reference position*

$$q_r \equiv q_d + \Lambda \int_0^t e \, d\tau \tag{16}$$

used to reduce the steady state error. The corresponding correction term is

$$\Delta w \equiv \Gamma \Psi^T(q, \dot{q}, \dot{q}_r, \ddot{q}_r)s. \tag{17}$$

We note that equation (17) is simpler than equation (14): the calculation of H^{-1} is not needed.

AFFC: Parallel to the developments of LS, Cr88 and Sl87, Kawato[25,26] proposed a neural network which corresponds to the Adaptive FeedForward Controller (AFFC) of reference 11. It used the control law

$$\begin{aligned} \tau &= \hat{F}^{-1}(q_d, \dot{q}_d, \ddot{q}_d) + \tau_{FB} + \alpha f \\ &\equiv \Psi(q_d, \dot{q}_d, \ddot{q}_d)w + Ds + \alpha f \\ &\equiv \hat{H}(q_d)\ddot{q}_d + \hat{C}(q_d, \dot{q}_d)\dot{q}_d + \hat{G}(q_d) + \hat{B}(\dot{q}_d) + Ds + \alpha f, \\ & f(e, \dot{e}) \equiv e^2 s, \quad \alpha > 0. \end{aligned} \tag{18}$$

The correction term is given by

$$\Delta w \equiv \Gamma \Psi^T(q_d, \dot{q}_d, \ddot{q}_d)\tau_{FB}. \tag{19}$$

The stability during learning and the robustness to noise have been shown analytically.[11]

Sl91: Slotine and Li proposed another scheme which is similar to Sl87, but whose control structure is close to the computed torque structure. The control law is

$$\tau = H(q)(\ddot{q}_r + \lambda s) + C(q, \dot{q})\dot{q}_r + G(q) + B(\dot{q}) + Ds \equiv \Psi(q, \dot{q}, \dot{q}_r, \ddot{q}_r, s)w, \quad \lambda \equiv \lambda I, \quad s\dot{e} + \lambda e. \tag{20}$$

i.e.

$$\tau = H(q)\ddot{q}^* + C(q, \dot{q})\dot{q}_r + G(q) + B(\dot{q}), \quad \dot{q}_r \equiv \dot{q}_d + \lambda e, \quad \ddot{q}^* = \ddot{q}_d + 2\lambda\dot{e} + \lambda^2 e = \ddot{q}_d + Ds. \tag{21}$$

The correction term is given by

$$\Delta w \equiv \Gamma \Psi^T(q, \dot{q}, \dot{q}_r, \ddot{q}_r, s)s. \tag{22}$$

As for Sl87, the drawbacks of Cr88 are corrected in this scheme (this is also true for the controllers introduced in references 13, 27). However, as appears in equation (21), the control structure is very similar to computed torque, and in fact exponential stability can also be proved for this scheme.[15]

CAC: The Composite Adaptive Controller of Slotine and Li[15] combines the minimization of tracking error with the minimization of identification error, and possess the theoretical advantages of both methods, i.e. the stability during learning and the robustness to noise. The control law is equation (15) and the correction term is given by

$$\Delta w \equiv P(t)(\Psi^T s + \Psi^T R(t)e), \tag{23}$$

where e is defined in equation (9) and $R(t)$ is the factor controling the relative importance of the tracking error to identification error in the minimization process. The calculation of this correction term is computationally intensive and must generally be performed off-line.

2.2.2. Nonparametric controllers

CMAC: The CMAC neural network[28] is formed by a preprocessing and a linear network. The preprocessing consists of a geometrical partition corresponding to virtual memories, which are projected to a smaller table of physical memories using a hash function. These memories in which are stored the weights of the linear network, are adapted using the "perception rule".[29] Albus used the CMAC neural network for storing the dynamics of manipulators.[3] In his controller, the input is $(q, \dot{q}, \ddot{q})$ and the teacher for the output $\tau = \tau_{FF} + \tau_{FB}$. A CMAC model of this inverse dynamics can also be adapted by minimizing the tracking error.[9]

J88: Jordan[30] proposed to store the dynamics in a "feedforward neural network" or "multilayers Perceptron" using the "backpropagation learning rule".[23] In

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

a first step, the dynamics are stored, thus the input is the motor torque vector τ and the teacher for the output the resulting trajectory $(q, \dot{q}, \ddot{q})$. In a second step, the inverse dynamics function can be stored, i.e. $(q_d, \dot{q}_d, \ddot{q}_d)$ is used as input and τ as teacher for the output.

MEMory (MEM): To store the dynamics of a manipulator in a table generally requires a too large table, and a too long time for learning. However, when the task consists of repeating a single trajectory, it is possible to correct for the dynamics only along this trajectory.[21] In designing the MEM, we tried to realize this in the most simple way. We used a feedforward control scheme and the correction term

$$\Delta w \equiv \Gamma \tau_{FB}. \quad (24)$$

This scheme will be described in more details in section 4.

We note that all the nonparametric models were used so far only with a feedforward control structure. They could however be used with computed torque structure also.

2.3. Comparison and evaluation of the controllers

2.3.1. Compensation of arbitrary trajectories. We first discuss the performances of the controllers for learning the dynamics in the whole space $(q, \dot{q}, \ddot{q})$.

Nonparametric versus parametric schemes: All the controllers defined above except MEMory can in principle be used for this task. The nonparametric schemes had the advantage that they can be used for arbitrary dynamics, i.e. also when the plant is difficult to model by a dynamic equation. However, we showed by simulating a two-link planar (SCARA) manipulator that CMAC and J88 have much worse performances than methods using the rigid body model equation.[5]

From the theoretical point of view, if one can prove the convergence of nonparametric model based control schemes (see section 4.2), no result exists concerning their stability. For the parametric schemes, strict stability and robustness results have been proved.[6,11,15,16]

In summary nonparametric methods should be avoided for storing the dynamics for arbitrary trajectories. This means that they should be used only if the parametric methods fail.

Minimization of the model error/of the tracking error: From the parametric methods, those based on the minimization of the model error (i.e. LS, CAC) require an intensive computation and must generally be performed off-line. In contrast, by the methods based on the minimization of the tracking error (AFFC, Sl87, Sl91), the learning requires only a few additional computation (see equation (12)) because the dynamic matrix Ψ must be computed anyway for compensating the dynamics. Cr88 requires more computation than these three. AFFC, Sl87 and Sl91 are thus the schemes which have the largest chance to be used on-line, which is critical for identifying parameters varying over time. We note that compact formulations have been found for these algorithms.[17,20]

Concerning the convergence against the true model, the schemes based on the minimization of the model error (Cr88, CAC) are by nature superior than those based on the minimization of the tracking error. However, the mathematical conditions for the convergence in the latter schemes are similar to those on the first schemes, and not much stronger:[15,31] they all depend on the "excitation level" of the trajectory, i.e. on the condition number of the matrix $\int \Psi^T \Psi$ along the trajectory. The very good theoretical properties of CAC, which combines the advantages of the minimization of the tracking and model errors, do not justify the intensive computation required.

Comparison between the schemes minimizing the tracking error: Finally, Whitcomb et al.[13] compared experimentally AFFC, Sl87 and an improved version of Cr88. They showed that nonlinear controllers clearly out perform linear controllers, and adaptive nonlinear controllers out perform their nonadaptive counter-parts. They also found that the simplest one, the Adaptive FeedForward Controller (AFFC), has the best control performances. We verified these results in a preliminary experiment.[32] This is probably due to the fact that with the AFFC the dynamics are computed using the desired trajectory and not the (noisy) actual trajectory.

Conclusion: The nonparametric schemes seem ill suited for storing the dynamics over the whole space $(q, \dot{q}, \ddot{q})$. Amongst the parametric schemes, the simplest one (AFFC) has as strong theoretical results regarding the stability as the other methods, and experimentally better performances.

2.3.2. Compensation along a repeated trajectory. Parametric controllers require finding the dynamic equation. This can be a rather complex task, particularly for parallel manipulators.[20] Parametric methods also require an "exciting" trajectory. When the task consists of driving a repeated trajectory, it is possible that this trajectory does not excite the whole dynamics, and that the dynamics must be learned along another trajectory. In contrast, nonparametric schemes simply store the motor torques along the repeated trajectory. They are therefore ideal candidates for this task.

We showed in simulations that, among the nonparametric methods, MEMory is more robust to noise than CMAC and J88 and about so robust as the AFFC.[5] MEMory also is the most simple of the three nonparametric methods studied.

2.3.3. Additional experimental tests required. The analysis of previous subsections suggests that: (i) The AFFC is the most suitable existing adaptive controller for learning and compensating the dynamics in arbitrary trajectories; (ii) MEMory is the simplest controller for compensating the dynamics along a repeated trajectory.

In order to complete the analysis of Whitcomb et al.[13] and test the practical utility of the AFFC for identifying the dynamics of manipulators, several additional experiments have to be performed:

- The control performance after learning must be tested on trajectories different than the trajectory used for learning.
- It must be examined if the AFFC is able to identify the friction (the friction is varying over time and difficult to identify).
- The robustness to noise must be experimentally investigated. Most of the real plants have noise, but this is difficult to simulate realistically.

We performed these tests and will report the corresponding results in section 3.

In section 4, we will investigate the mathematical properties of MEMory and test experimentally its ability to identify a plant with friction, the resulting control performance and its robustness to noise. We will also examine how MEMory and AFFC can be combined.

3. EXPERIMENTS WITH THE AFFC

3.1. Experimental apparatus

The manipulator used for the experiment is the planar parallel manipulator shown in Figure 1. This manipulator is driven by two direct drive DC-motors and can move with high speeds, inducing highly non-linear dynamic effects and coupling. The joint positions are measured by two optical encoders. The robot is controlled by a VME-68040 board enabling a sampling frequency of 333.3 Hz.

If the velocities are differentiated from the encoder position, this causes a large noise (Figures 5 and 13). In most of our experiments, the joint velocities were instead determined with velocity sensors using potentiometers and an analog differentiator.[32] In order to test experimentally the robustness to noise of the learning algorithm, some trials used the velocities differentiated from the encoder position.

3.2. Model of the manipulator

The manipulator is schematized in Figure 7. Its dynamic equation is given by equation (5) with 32

$$
\begin{aligned}
H(q) &\equiv \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}, \\
h_{11} &\equiv J_{\text{mot}} + 3J + l^2 m_C + \tfrac{11}{4} l^2 m, \\
h_{12} &= h_{21} \equiv -l^2 c_{12}(m_C + \tfrac{3}{2} m), \\
h_{22} &\equiv J_{\text{mot}} + 3J + l^2(m_C + m_A) + \tfrac{7}{4} l^2 m, \\
s_{12} &\equiv \sin(q_1 + q_2), \quad c_{12} \equiv \cos(q_1 + q_2), \\
C(q, \dot{q}) &\equiv \begin{bmatrix} 0 & C\dot{q}_2 \\ C\dot{q}_1 & 0 \end{bmatrix}, \quad C \equiv l^2 s_{12}(m_C + \tfrac{3}{2} m), \\
G(q) &\equiv 0, \quad B(\dot{q}) \equiv \begin{bmatrix} b_{11}\dot{q}_1 + b_{12}\,\text{sign}(\dot{q}_1) \\ b_{21}\dot{q}_2 + b_{22}\,\text{sign}(\dot{q}_2) \end{bmatrix}.
\end{aligned}
\tag{25}
$$

l, m and J are the length, mass and moment of inertia of each of the 6 links which are considered to be equal (Figure 7). m_C is the mass of the end effector, J_{mot} the moment of inertia of each of the 2 motors, and m_A is the mass of the plate located at the right elbow. $B(\dot{q})$ corresponds to the friction at the motors, modelled as kinetic and viscous friction. The physical parameters can be found in Table II.

By measuring the velocity obtained with different constant motors torques (Figure 6), it was verified that

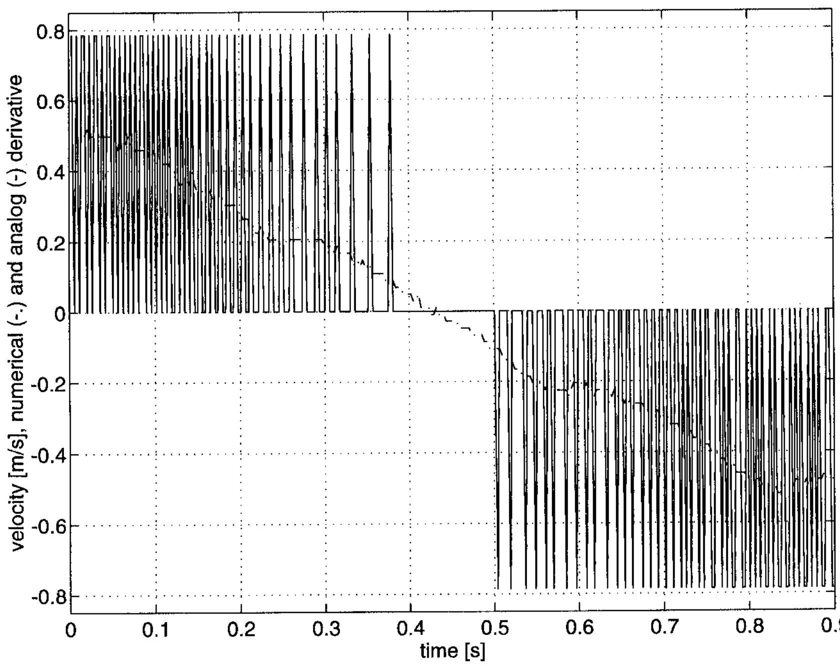

Fig. 5. Comparison between the joint velocity obtained with the velocity sensor (dashed-dotted line) and by differentiating numerically the position measured by the encoder (continuous line).

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

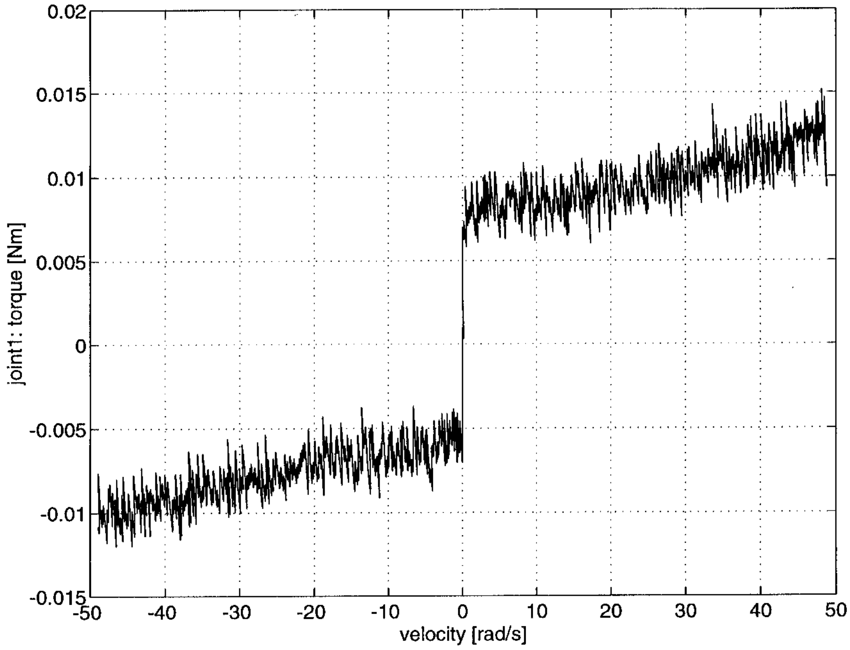

Fig. 6. Friction torque of one motor of the manipulator measured using a velocity controller. The other motor has a similar friction.

the modeled friction is similar to real friction. However, the friction parameters b_{ij} vary greatly, depending on the moment the measurement is performed, i.e. if the motor was running a long time or not.

The linear representation of our manipulator used in the experiments is given by equation (6) by

$$\Psi^T \equiv \begin{bmatrix} s_{12}\dot{q}_2^2 - c_{12}\ddot{q}_2 + \ddot{q}_1 & s_{12}\dot{q}_1^2 - c_{12}\ddot{q}_1 + \ddot{q}_2 \\ \frac{3}{2}(s_{12}\dot{q}_2^2 - c_{12}\ddot{q}_2) + \frac{11}{4}\ddot{q}_1 & \frac{3}{2}(s_{12}\dot{q}_1^2 - c_{12}q_1) + \frac{7}{4}\ddot{q}_2 \\ \ddot{q}_1 & \ddot{q}_2 \\ \ddot{q}_1 & 0 \\ \dot{q}_1 & 0 \\ \text{sign}\,(\dot{q}_1) & 0 \\ 0 & \dot{q}_2 \\ 0 & \text{sign}\,(\dot{q}_2) \end{bmatrix} \tag{26}$$

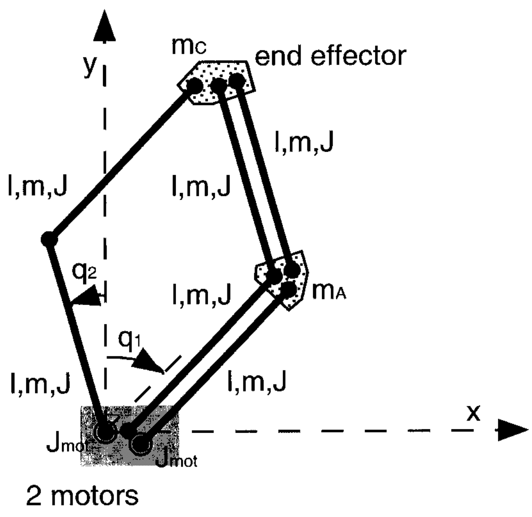

Fig. 7. A top view of the manipulator used to test the nonlinear adaptive controllers.

and

$$w \equiv (l^2 m_C,\, l^2 m,\, 3J + J_{\text{mot}},\, l^2 m_A,\, b_{11},\, b_{12},\, b_{21},\, b_{22})^T.$$

3.3. *Learning strategy and experiments*

A periodical learning trajectory was used. It is formed of six smoothly joined fifth order polynomials. The corresponding path is shown in Figure 8. The maximal velocity is about 4.5 rad/s and the maximal acceleration about 45 rad/s^2. The mean condition number along this trajectory is relatively high.[32] Thus, this trajectory is likely to be favorable for the identification of the dynamics.[33,34]

We use as matrix of learning factors

$$\Gamma \equiv \varepsilon \cdot \text{diag}\,(\gamma_1, \ldots, \gamma_8). \tag{27}$$

In this equation, the $\gamma_i > 0$ are chosen so that all the parameters w_i converge approximately at the same time, and

$$\varepsilon \equiv \varepsilon_1 + \varepsilon_2, \tag{28}$$

Table II. Comparison between the measured and learned parameters. The measured length of each limb is 0.132 m. All the units are SI units: mass in kilograms, lengths in meters, ... The 0?'s indicate that the value of the dynamic friction could not clearly be identified using the measurement (see Figure 6)

Weight	Corr. phys. par.	Measured	Learned
$w_1 10^3$	$l^2 m_C$	1.21	1.15
$w_2 10^4$	$l^2 m$	1.50	1.03
$w_3 10^5$	$3J + J_{\text{mot}}$	8.30	1.32
$w_4 10^4$	$l^2 m_A$	2.50	3.33
$w_5 10^8$	b_{11}	0?	8.28
$w_6 10^3$	b_{12}	6.50	8.62
$w_7 10^8$	b_{21}	0?	2.88
$w_8 10^3$	b_{22}	10.00	5.65

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

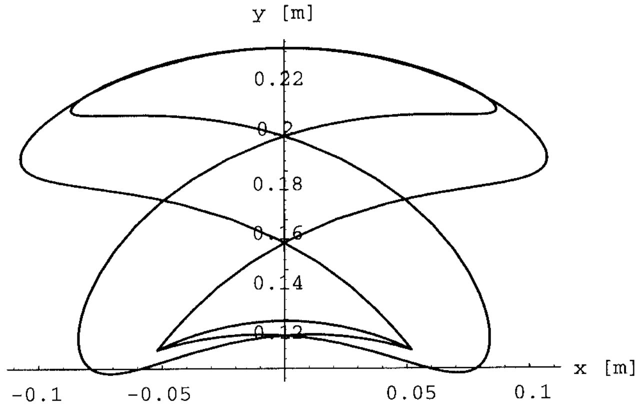

Fig. 8. The trajectory used for identifying the dynamics.

with $\varepsilon_1 > 0$ decreasing exponentially and $\varepsilon_2 > 0$ a small constant. An alternative but computationaally more requiring choice is

$$\Gamma \equiv \varepsilon \cdot P, \quad P^{-1} \equiv \int_0^T \Psi^T \Psi \, dr. \tag{29}$$

This choice should be good, as it corresponds to the least-square solution (see equations (10) and (12)).

We showed that the last term $\alpha f(e, \dot{e})$ of the control law (18) is not necessary for assuring the stability and convergence provided that $D \gg C$, i.e. if the matrix D multiplicating the feedback is much higher (in norm) than the matrix of the velocity dependent forces C.[32] In this case, the term αf is only adding noise. In preliminary experiments, we tested both with and without this term, and found a better performance without it. We therefore chose to neglect it.

Beginning without knowledge of their value ($w \equiv 0$), the parameters are learned along the trajectory of Figure 8. The control performance without and with noise are then tested along the learning trajectory and other (test) trajectories. As adaptive control is particularly attractive for parameters varying over time, we finally measure the time required for learning, when only the friction or the mass of the end-effector are identified.

3.4. *Results*

- The position and velocity errors decrease to the minimal value in about 5 sec (Figure 9). The 8 weights convege to their true value in about 1 min (Figure 10). The learned parameters are of the same order than the measured one, but have different values (Table II).
- With the measured and learned dynamics, most of the feedback error is corrected by the feedforward prediction (Figure 11). The errors in position and velocity are slightly smaller with the learned parameters than with the measured ones (Table III). This holds true for the learning trajectories and for arbitrary test trajectories. This implies in particular that the friction has been identified well.
- The open-loop contol (i.e. without the feedback) is good (Figure 12). The open-loop performances are better with the learned than with the measured weights,[32] for the learned trajectory as well as for the test trajectories.
- The noise does not disturb learning. The position and velocity errors decrease as quickly without noise. The identification of the parameters needs a slightly longer time. The resulting control is also good (Figure 13).
- When only the friction is identified, the convergence time is about 40 sec, i.e. 2/3 of the time needed to learn the whole dynamics.[32]
- Identifying the mass of the end effector only needs

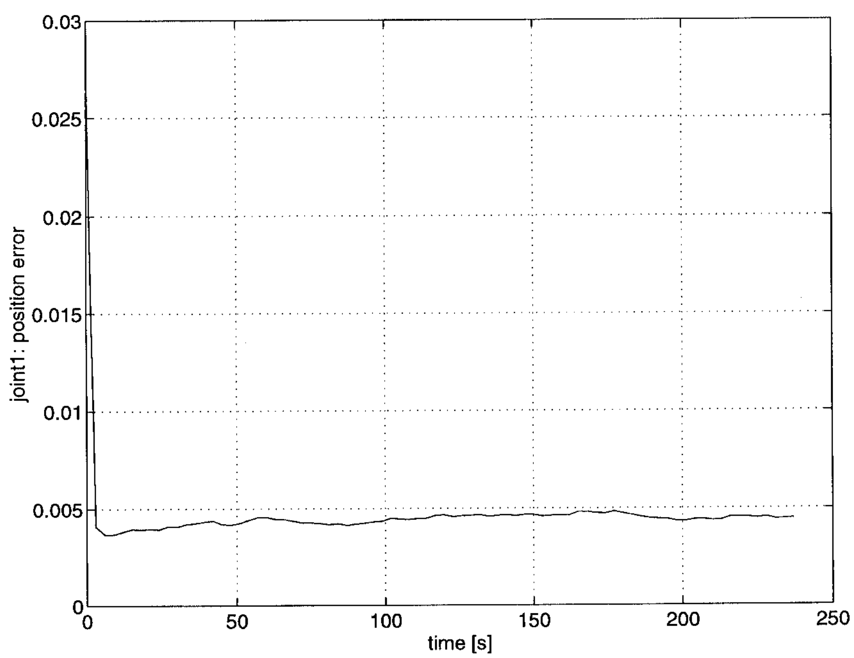

Fig. 9. AFFC: Evaluation of the position error $\int_{\text{traj}} e_1^2$ of joint 1 during learning. The curve corresponding to joint 2 is similar.

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

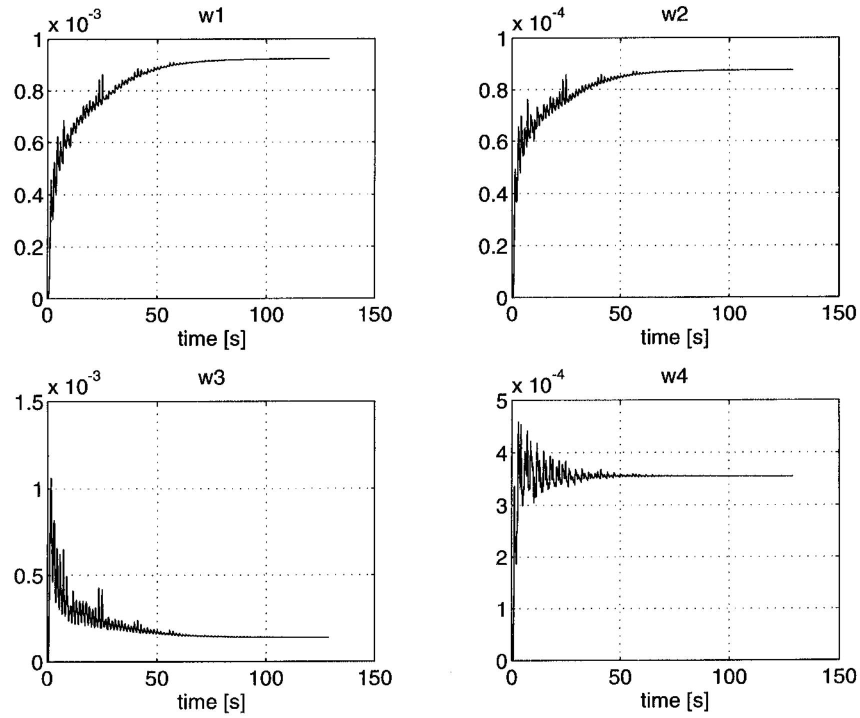

Fig. 10. AFFC: Evolution of the four first weights during learning (the other curves are similar). The identification is completed in about 1 min. Note that it requires about 10 times longer to complete the adaptation of the weights than to reduce thr tracking error.

about 0.7 sec, i.e. about 100 times faster than for the whole dynamics (this time could probably still be reduced).[32]

In conclusion, the AFFC was able to learn the dynamics of our parallel manipulator fast, even in presence of a large noise. The friction could be identified well, and the learned dynamics compensated the real dynamics for arbitrary trajectories better than when the parameters were measured, i.e. the technique commonly used.

4. MEMORY: STORING THE DYNAMICS ALONG A REPEATED TRAJECTORY

If a manipulator is well controlled by a linear feedback controller, it approximately follows the desired trajec-

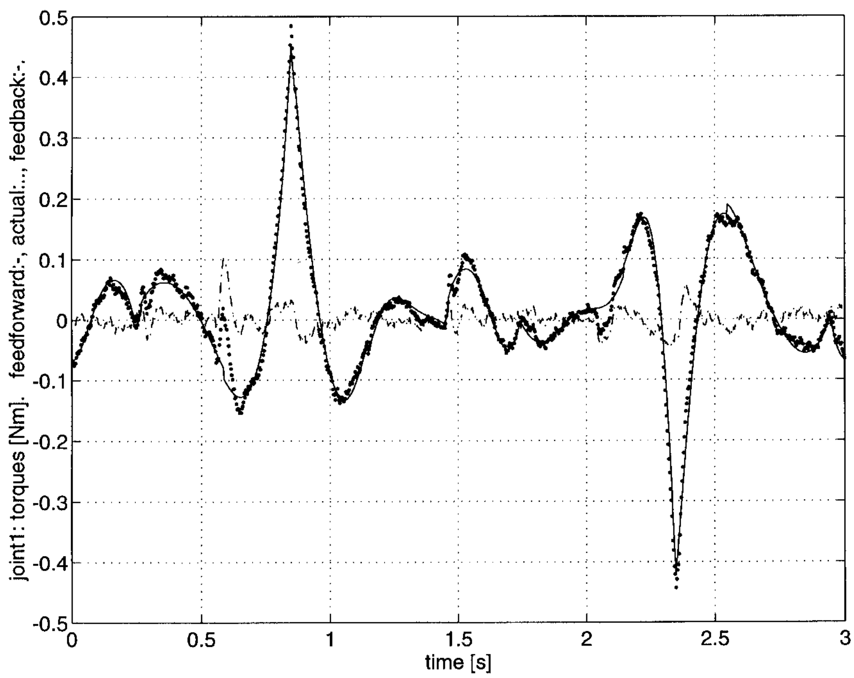

Fig. 11. AFFC: Resulting torques after learning along the learning trajectory. Continuous line: learned feedforward torque; dashed dotted line: feedback torque; dots: total torque. The results are similar for the other joint.

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

Table III. Errors along the learning trajectory and along a test trajectory with the merasured and the learned dynamics. The test trajectory used here is depicted in figure 12.

	Learning trajectory		Test trajectory	
Error	Measured parameters	Learned parameters	Measured parameters	Learned parameters
$10^2 \int e_1^2$	4.0	3.8	1.5	1.4
$10^2 \int e_2^2$	2.8	2.7	1.0	1.0
$10^{-1} \int \dot{e}_1^2$	7.5	7.1	3.1	3.0
$10^{-1} \int \dot{e}_2^2$	5.5	5.4	2.1	2.1
$10^2 \int \left(\frac{\tau_{FB,1}}{\tau_{FF,1}}\right)^2$	2.0	1.9	4.5	3.7
$10^2 \int \left(\frac{\tau_{FB,2}}{\tau_{FF,2}}\right)^2$	2.7	2.2	3.9	3.6

tory. This means that the feedback torque approximately correspond to the manipulator dynamics. *The idea of the MEMory controller is, by a repeated movement, to store the feedback torque and use it as feedforward torque for the next run.* Because the feedback torque corresponds to the manipulator dynamics, the new feedback contribution should be smaller.

As feedback always incorporates a delay, it does not correspond exactly to the manipulator dynamics. To avoid unstability,[35] only a part of this feedback signal is memorized at each step (Figure 14).

4.1. Description

Let us now describe the algorithm mathematically. We model the irreproducible part of the manipulator dynamics by a random noise term

$$N(k),\ k = 1 \cdots K, \quad \text{so that} \quad \exists M, \sum_{m=k-M}^{k+M} N(m) = 0\ \forall k. \tag{30}$$

The total torque τ is then given by

$$\tau(k) = \tau_P(k) + N(k) \quad \forall k, \tag{31}$$

where $\tau_P(k)$, $k = 1 \cdots K$, is the reproducible part of the torque.

The *control law* for the i-th *run* is

$$\tau^{(i)}(k) = \tau_{FF}^{(i)}(k) + \tau_{FB}^{(i)}(k) \quad \forall k, \tag{32}$$

where

$$\tau_{FB}^{(i)}(k), \quad k = 1 \cdots K \tag{33}$$

is the torque function for the i-th run of the trajectory and k the discrete time index. From the equations (31) and (32), it follows that

$$\tau_{FB}^{(i)}(k) = \tau_P(k) + N(k) - \tau_{FF}^{(i)}(k) \quad \forall k. \tag{34}$$

We define the *learning rule* as follows:

$$\begin{aligned} &\forall k \geq 1, \quad \tau_{FF}^{(0)}(k) \equiv 0, \\ &\tau_{FF}^{(i+1)}(k) \equiv \tau_{FF}^{(i)}(k) + \frac{\lambda}{2M+1} \sum_{k-M}^{k+M} \tau_{FB}^{(i)}(m), \end{aligned} \tag{35}$$

where λ, $0 \leq \lambda \leq 2$, is the *learning factor*. The sum in (35)

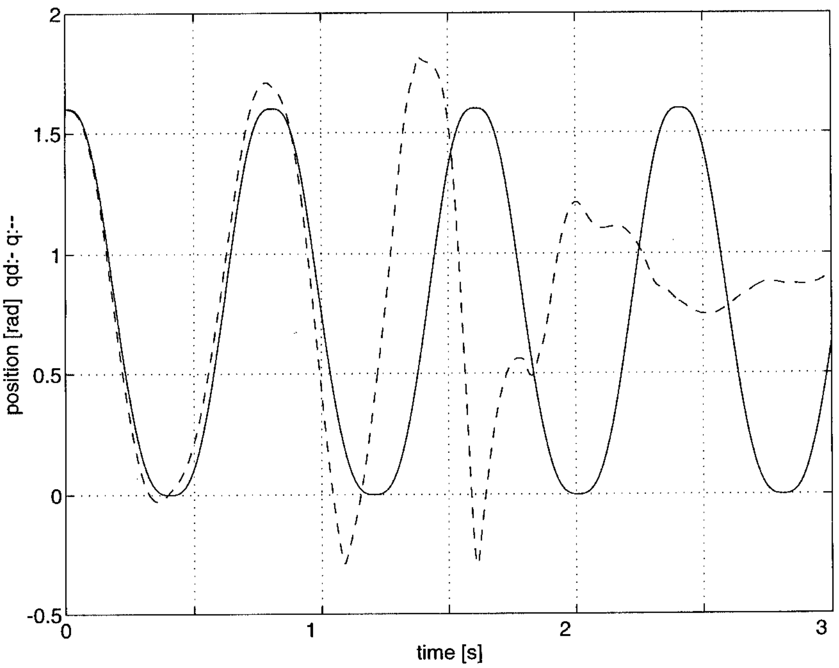

Fig. 12. AFFC: Open-loop movement along a trajectory different from the learning trajectory. Continuous line: desired trajectory; dashed line: actual trajectory.

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

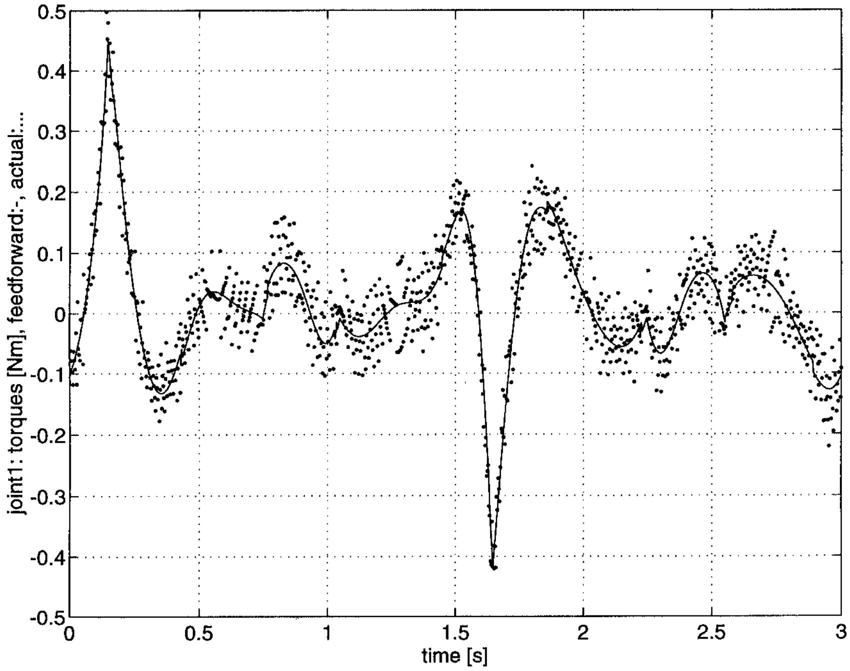

Fig. 13. AFFC: Resulting torques after learning along the learning trajectory, when the system was disturbed by noise. Continuous line: learned feedforward torque; dots: total torque. The feedforward is in the middle of the noisy total torque curve. The results of the other joint are similar.

smoothes the feedback signal and eliminates the noise (Figure 15):

$$\frac{\lambda}{2M+1}\sum_{k-M}^{k+M}\tau_{FB}^{(i)}(m)$$
$$=\frac{\lambda}{2M+1}\left(\sum_{k-M}^{k+M}\tau_P(m)+\sum_{k-M}^{k+M}N(m)-\sum_{k-M}^{k+M}\tau_{FF}^{(i)}(m)\right)$$
$$\cong\lambda(\tau_P(k)-\tau_{FF}^{(i)}(k)).$$

The first equality results from equation (34), the second from equation (30) and the fact that $\tau_P(k)$ and $\tau_{FF}^{(i)}(k)$ are almost linear over the time interval $[k-M, k+M]$.*

Note that the sum in equation (35) constitutes a noncausal filter (Figure 15). We did not use a common (causal) filter (as for example a filter first order [6] or second order, see further reference 21), because this would induce a delay which can cause instability.[35]

* This assumption is obviously not verified in the isolated points where τ_P is not continuous. At these points, the sum over the m points smooths the torque signal ρ_P.

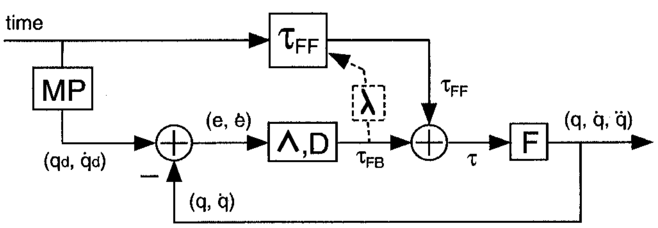

Fig. 14. The control scheme of the MEMory controller. *MP* stands for motion planner, $e=q_d-q$ is the error between desired and actual joint positions, F represents the plant dynamics, τ_{FF} the memorized torques stored in the table, D and Λ are symmetrical positive definite matrices, and τ the motor torques. λ, $0<\lambda<2$, is a multiplicative learning factor.

4.2 Mathematical properties

From equations (35) and (36) an iterative equation for the feedforward torque can be derived:

$$\tau_{FF}^{(i+1)}(k)\cong\tau_{FF}^{(i)}(k)+\lambda(\tau_P(k)-\tau_{FF}^{(i)}(k)). \quad (37)$$

By induction, we arrive at the result

$$\tau_{FF}^{(i)}(k)\cong\lambda\sum_{j=0}^{i-1}(1-\lambda)^j\tau_P(k)=(1-(1-\lambda)^i)\tau_P(k). \quad (38)$$

It follows that

$$\tau_{FF}^{(i)}\rightarrow\tau_P,\quad i\rightarrow\infty \quad (39)$$

i.e. the memorized torque converges to the reproducible part of the dynamics when the movement is repeated, and the feedback controller then corrects only for the irreproducible part of the dynamics.

We note that the above proof of convergence can also be used with slight changes as convergence proof for the CMAC.[5]

4.3. Experiments and results

The above algorithm has been implemented with the same manipulator as in section 3 and along the same trajectories. The robustness to noise has been again tested using the numerically differentiated velocity.

The algorithm is so trivial that the implementation was realised very quickly. The convergence of the error to its minimum value is about as fast as with AFFC (about 30 sec). The resulting position and velocity errors were approximately twice smaller. We do not show here many figures, because they are similar to these for the AFFC. We only show in Figure 16 the astonishing open-loop performance, which prove that a better model of the inverse dynamics has been learned (compare Figure 16

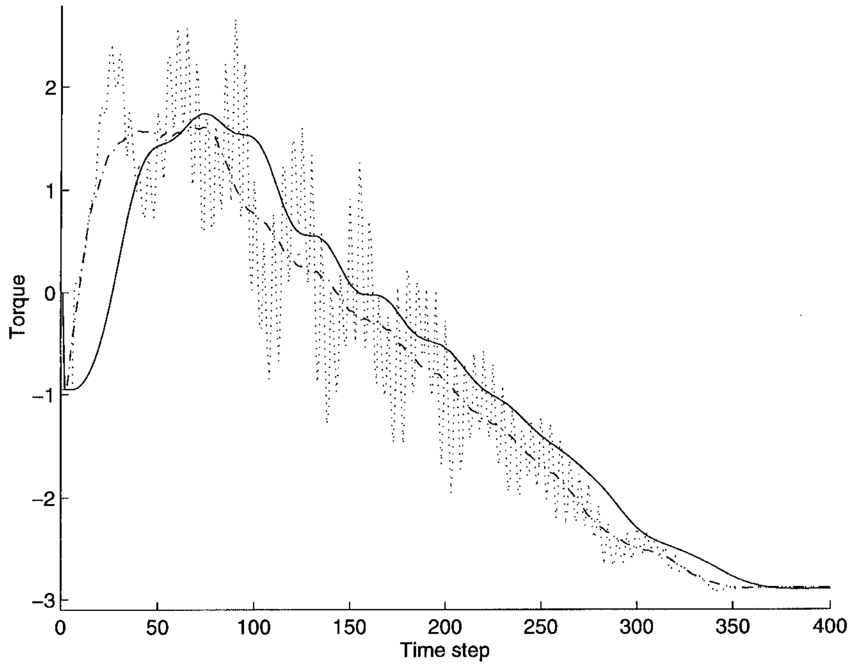

Fig. 15. MEM: Torque of one joint measured along a movement (dotted line), and two filtered versions: with a second order Butterworth filter (continuous line) and a noncausal filter like in our algorithm. The measurement has been performed with a manipulator similar than those of Figure 1, but bigger (limb length: 80 cm) and with higher compliance.

with Figure 12). Further, the learning is not disturbed by the noise (Figure 17). In the presence of noise, the adaptation is about twice slower.

4.4. Hybrid controller

Because the MEMory controller provides a better tracking than the AFF controller, there probably is a reproducible part of the dynamics which was identified by MEMory but not modelled in the rigid body equation. Is it yet possible to identify this part using MEMory after an AFFC has been used to learn the dynamics? Would it lead to better results compared with the MEMory only? In the resulting *hybrid controller,* the feedforward compensator is composed of a model incorporating the rigid body equations and a look-up-table, and the feedback part is given by the feedback controller. This corresponds simply to use MEMory when the dynamics are already compensated with the rigid body equation.

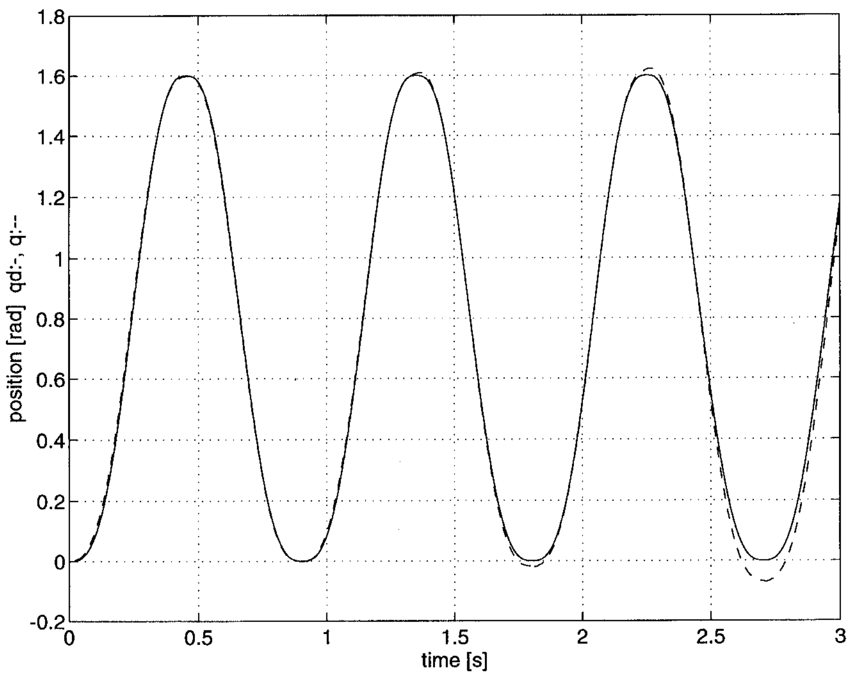

Fig. 16. MEM: Position function of one joint for a movement performed open-loop (the other joint shows a similar function). Note that contrary to the corresponding figure of last section, the error increases very slowly. This shows that the MEMory has stored a better model of the dynamics than the AFFC.

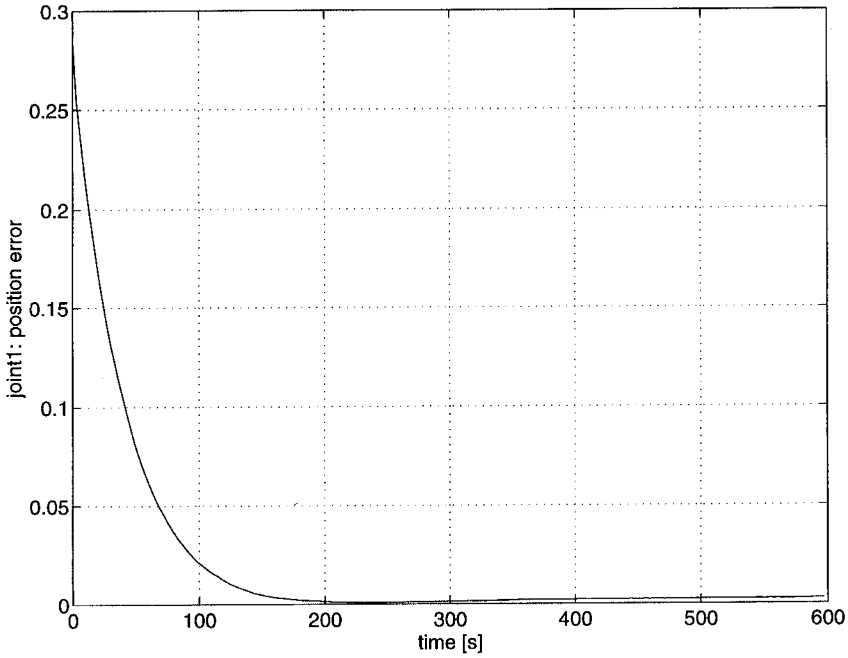

Fig. 17. MEM: Position error $\int_{traj} e_1^2$ during learning with a noisy velocity signal.

At the beginning the manipulator is controlled uniquely by the feedback controller. Then the parameters of the rigid body model are learned along an exciting trajectory. Finally, for a repeated movement, the remaining reproducible dynamics are stored in the adaptive look-up-table.

A similar hybrid controller was proposed and simulated in reference 6, but to our knowledge no implementation of this method was done. Another adaptive controller using a model and a look-up-table was proposed in reference 4.

We implemented and tested the hybrid controller. It resulted that:

- The additional MEMory is able to improve the control relative to the learned rigid body model. This means that the reproducible part of the dynamics not modelled by the rigid body dynamic equation has been identified.
- Most of the systematic errors have been eliminated (compare for example Figure 11 at time 0.6 sec with Figure 18 at time 2.8 sec).

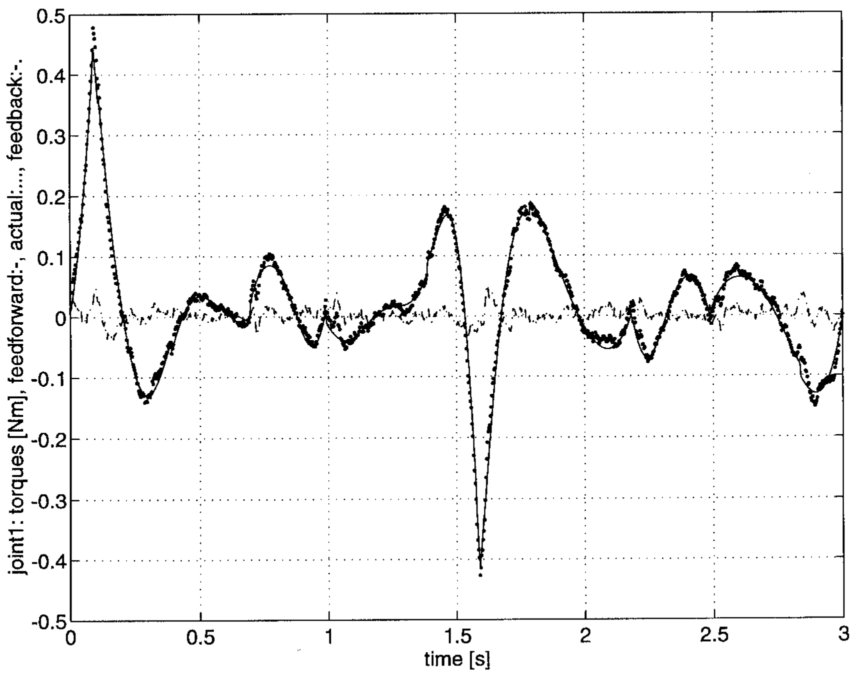

Fig. 18. MEM: Resulting torques after learning with the hybrid controller along the trajectory of Figure 8. Continuous line: learned feedforward torque, dashed dotted line: feedback torque, points: total torque. The results are similar for the other joint.

https://doi.org/10.1017/S0263574798000150 Published online by Cambridge University Press

- The remaining errors are of the same order of magnitude than when using MEMory only. The control performances are also similar. Thus it is not necessary to identify the dynamics in a first step or even to use a rigid body dynamic model, contradicting the claim of reference 4.

5. CONCLUSIONS

In this paper we analyzed practical issues critical for the application of nonlinear adaptive control in robotics. Based on results found in the literature, we selected the best algorithms, which we then tested experimentally with a 2-dof planar parallel robot.

The Adaptive FeedForward Controller (AFFC)[11,25] was selected as the best algorithm for learning and compensating the dynamics in arbitrary trajectories. It was shown that the AFFC is stable and robust to noise, enables identification of the friction, and results in better control performance than when the parameters of the dynamic equation are measured. In addition, the AFFC requires a very few computation and is non-invasive. It is thus currently the most practical way to perform the dynamic calibration of robots.

If the task consists in repeating a single movement, it is not necessary to derive the rigid body dynamic model and identify its parameters. It is simpler to use the adaptive look-up table MEMory introduced in this paper for learning and compensating for the dynamics along the repeated trajectory. An advantage of MEMory is that it can also be used when the dynamics are difficult to model with a dynamic equation. It was shown that this controller is robust to noise and results in an almost perfect compensation of the dynamics.

We have recently extended the above methods to arbitrary manipulators,[20] and successfully applied them to parallel 3-dof[36] and 6-dof[2] manipulators. Together with the above results, this suggests that AFFC and MEMory are mature enough to be used in industrial applications.

ACKNOWLEDGMENTS

Many thanks to B. Gruehl, J. Luthiger, M. Nuttin, A. Schmid and B. Sprenger for their valuable contributions.

References

1. A. Codourey, "Contribution à la commande des robots rapides et précis, application au robot Delta à entrainement direct" *PhD thesis* (Ecole Polytechnique Fédérale de Lausanne, 1991).
2. M. Honegger, A. Codourey and E. Burdet, "Adaptive control of the Hexaglide, a 6 dof parallel manipulator" *IEEE International Conference on Robotics and Automation* (1997) pp. 543–548.
3. J.S. Albus. A new approach to manipulator control: the cerebellar model articulation controller. *J. Dynamical Systems, Measurements, and Control* **97,** 220–227 (1975).
4. C.H. An, C.G. Atkeson and J.M. Hollerbach, *Model-Based Control of a Robot Manipulator* (MIT Press, Cambridge, Mass., 1988).
5. E. Burdet and J. Luthiger, "Three learning architectures to improve robot control: a comparison" *3rd European Workshop on Learning Robots, 8th European Conference on Machine Learning* (1995).
6. J.J. Craig, *Adaptive Control of Mechanical Manipulators* (Addison-Wesley, Reading, Mass., 1988).
7. D.M. Gorinevsky, "Modelling of direct motor program learning in fast human arm motions" *Biological Cybernetics* **69,** 219–228 (1993).
8. M.I. Jordan, T. Flash and Y. Arnon, "A model of the learning of arm trajectories from spatial deviations" *J. Cognitive Neuroscience* **6**(4), 359–376 (1994).
9. H. Kano and K. Takayama, "Learning control of robotic manipulators based on neurological model CMAC" *11th International Federation of Automatic control Congress* (1990) pp. 268–273.
10. P.K. Khosla, "Real-time Control and Identification of Direct-drive Manipulators" *PhD thesis* (Carnegie-Mellon University, 1986).
11. N. Sadegh and R. Horowitz, "Stability and robustness analysis of a class of adaptive controller for robotic manipulators" *Int. J. Robotics Research* **9**(3) 74–92 (1990).
12. J.-J.E. Slotine and W. Li, "Adaptive manipulator control, a case study" *IEEE International Conference on Robotics and Automation* (1987) pp. 1392–1400.
13. L.L. Whitcomb, A.A. Rizzi and D.E. Koditschek, "Comparative experiments with a new adaptive controller for robot arms" *IEEE Transactions on Robotics and Automation* **9**(1) 59–70 (1993).
14. R. Shadmehr and F. A. Mussa-Ivaldi, "Adaptive representation of dynamics during learning of a motor task" *J. Neuroscience* **14**(5) 3208–3224 (1994).
15. J.-J.E. Slotine and W. Li, *Applied Nonlinear Control* (Prentice-Hall International Editions, Inc., 1991).
16. F.L. Lewis, C.T. Abdallah and D.M. Dawson, *Control of Robot Manipulators* (Macmillan, London, 1993).
17. G. Niemeyer and J.-J.E. Slotine, "Performance in adaptive manipulator control" *Int. J. Robotic Research* **10**(2) 149–161 (1991).
18. B. Armstrong, *Control of Machines with Friction* (Kluwer Academic, Holland, 1991).
19. W. Khalil and J.F. Kleinfinger, "Minimum operations and minimum parameters of the dynamic models of tree structure robots" *IEEE Transactions on Robotics and Automation,* **3**(6) 517–526 (1987).
20. A. Codourey and E. Burdet, "A body-oriented method for finding a linear form of the dynamic equaton of fully parallel robots" *IEEE International Conference on robotics and Automation* (1997) pp. 1612–1618.
21. A. Arimoto, Learning control. In M.W. Spong, F.L. Lewis and C.T. Abdallah (editors) *Robot Control* (IEEE Press, 1993) pp. 185–188.
22. R. Sanner and J.-J.E. Slotine, "Gaussian networks for direct adaptive control" *IEEE Transactions on Neural Networks* **3**(6) 837–863 (1992).
23. D.E. Rumelhart, J.L. McClelland and the PDP Research Group, *Parallel Distributed Processing* (MIT Press, Cambridge, Mass., 1986).
24. J.J. Craig, *Introduction to Robotics, Mechanics and Control* (Addison-Wesley, Reading, Mass., 1989).
25. H. Miyamoto and M. Kawato, "Feedback-error-learning neural networks for trajectory control of a robotic manipulator" *Neural Networks* **1,** 251–265 (1988).
26. M. Kawato, K. Furukawa and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement" *Biological Cybernetics* **57,** 169–185 (1988).
27. K.A. ElSerafi, "Contributions à la commande Adaptative des Robots Manipulateurs" *PhD thesis* (Université de Nantes, 1991).
28. J.S. Albus, "Data storage in the cerebellar model articulation controller" *J. Dynamical Systems, Measurement, and Control* **97**(3) 228–233 (1975).
29. H. Ritter, T. Martinez and K. Schulten, *Neuronale Netze* (Addison-Wesley, Reading, Mass., 1991).
30. M.I. Jordan, "Supervised learning and systems with excess degrees of freedom" *Technical Report* (Department of Computer Science and Information, University of Massachusetts, 1988).

31. B.D.O. Anderson, "Exponential stability of linear systems arising from adaptive identification" *IEEE Transactions on Automatic Control* **22**(1) 83–88 (1977).
32. A. Schmid and M. Zaugg, "Balancieren von invertiertem Einfach- und Doppelpendel mit einem 2D-Deltaroboter" *Technical Report* (Institut fuer Robotik, ETH-Zuerich, 1995).
33. B. Armstrong, "On finding "exciting" trajectories for identification experiments involving systems with nonlinear dynamics" *IEEE International Conference on Robotics and Automation* (1987) pp. 1131–1139.
34. A.P. Morgan and K.S. Narendra, "On the uniform asymptotic stability of certain linear nonautonomous differential equations" *S.I.A.M. Journal of Control and Optimization* **15**(1) 5–24 (1977).
35. E. Burdet, J. Luthiger, M. Nuttin, G. Schweitzer and H. Van Brussel "A comparison of adaptive and learning control schemes for trajectory control of robot manipulators" *Technical Report* (Institut fuer Robotik, ETH-Zuerich, 1995).
36. E. Burdet, L. Rey and A. Codourey, "A trivial method of learning control", IFAC Symposium on Robot Control (1997).