A Frequency-Domain Approach to Learning Control: Implementation for a Robot Manipulator

Alessandro De Luca, *Member*, *IEEE*, Giorgio Paesano, and Giovanni Ulivi, *Member*, *IEEE*,

Abstract—**A new frequency-domain approach to the analysis and design of learning control laws for achieving a desired repetitive behavior in a dynamical systems is presented. The scheme uses two separate filters in order to obtain rapid improvements in a specified bandwidth while cutting off possibly destabilizing dynamic effects that would bar learning convergence. In this way, the trade-off between global convergence conditions and approximate learning of trajectories is made explicit. The synthesis is presented for single-input single-output (SISO) linear systems, but the method is of general application. The proposed learning controller has been used for exact tracking of repetitive trajectories in robot manipulators. In particular, actuator inputs that enable accurate reproduction of robot joint-space trajectories are learned in few iterations without the knowledge of the robot dynamic model. Implementation aspects are discussed and experimental results are reported that show how the proposed learning scheme avoids the occurrence of unstable phenomena over iterations.**

I. Introduction

LEARNING control is a technique in which the input signal required to achieve a given behavior as output of a dynamical system is built iteratively from successive experiments. Performance on repetitive tasks is improved from one iteration to the other until the desired goal is attained.

One major advantage of learning control is that it works even with limited a priori knowledge of the mathematical model of the system. In particular, the development of an accurate model can be restricted to that part of the system dynamics that is known exactly, has small parametric uncertainties, and can be evaluated in real time with little computational effort. Under suitable assumptions, the learning scheme will acquire from trials the additional information needed for the successful completion of the task. As compared to adaptive control schemes, learning control is limited by the assumption of a repetitive nature of the desired behavior. Moreover, no explicit parameter identification is performed and the learning process should be restarted when tackling a different task. On the other hand, the learning scheme can be made robust with respect to unmodeled dynamics, as opposed to standard adaptation schemes, while the need for iterative operation is not restrictive in many situations.

Manuscript received December 20, 1989; revised September 9, 1991. This paper is based on work sponsored by the Consiglio Nazionale delle Richerche, grant no. 91-01946 (Progetto Finalizzato Robotica).

The authors are with the Dipartimento di Informatica e Sistemistica, Università degli Studi di Roma "La Sapienza," 00184, Rome, Italy.

IEEE Log Number 9105263.

Another relevant aspect of learning schemes is the simplicity of the resulting control law, enabling a cheap implementation without heavy real-time computational burden. Therefore, high-performance microcomputers are not required as most part of the numerical processing can be done off line. Instead, a rather large amount of memory is needed for storing the time profiles of selected process variables. This is not a real drawback, as memory cost is low and continuously decreasing, whereas size is growing at a fast rate.

Iterative learning has already found wide application in the robotic field. In fact, the repetitive task of tracking a desired path is a very natural one for an industrial manipulator. However, this control technique can be applied to a broader class of systems and operative conditions, including those subject to periodic disturbances. In [4], a learning controller has been added in parallel, or "plugged-in", to an already existing controller designed for a disk drive actuator in order to compensate for structural misalignments. A similar approach can be used for rejecting the effects of eccentricity in electrical motors. As a result, the learning strategy allows to relax the requirements of a purely model-based control design. Moreover, it provides a natural way to generate the suitable feedforward action needed to modify a constant set-point regulator into a trajectory tracking controller.

In the rapidly increasing related literature, most of the proposed learning algorithms were investigated first for the case of linear systems [1]-[8]. The analysis of the conditions that guarantee convergence of the iterative scheme is performed either in the frequency [1]-[3] or in the discrete [4]-[6] or continuous time [7], [8] domains. For linear time-varying systems of dynamic order two (also called acceleration systems), different simple learning laws have been studied, based on the proportional integral (PI), proportional derivative (PD), or proportional-integral derivative (PID) treatment of the velocity error. In particular, the last two require also a direct or indirect measure of joint accelerations [9].

Extensions to specific classes of nonlinear systems were also considered [9]-[13]. These include mechanical systems —like robot arms—which possess the relevant properties of being stabilizable by a linear PD law and exactly transformed into a linear system by means of a nonlinear state feedback. Although the proof of convergence is more involved in these cases, the resulting learning control scheme is still simple.

In this paper, a new learning controller is proposed and its

convergence studied for linear time-invariant systems, using a frequency-based approach. The specific features of this scheme are:

1) Relaxed convergence conditions are obtained as the method allows for different weighting of "what to learn" from the current trial and "what to remember" from the previous ones. Only a small amount of additional signal processing is needed to enforce convergence, typically cutting off high-frequency system dynamics. In a robot arm, these could be originated by unforeseen elasticity of the joint transmissions.
2) The presence of a feedback controller, designed for the original plant, is explicitly included in the convergence analysis. Thus, the elements of the learning algorithm will be designed with respect to closed-loop characteristics, which are more convenient and predictable than open-loop ones.
3) Because the approach is frequency oriented, it provides deeper engineering insights during the design phase, where tools like frequency response, filtering, and Nyquist plots can be used directly.

The paper is organized as follows. The learning algorithm is presented for the case of a SISO linear system, subject to feedback from the output. The peculiarities of the proposed approach over previous ones can be fully illustrated already in such a simple case. It is then easy to incorporate the basic ideas into other existing analyses (see e.g. [2], [6], and [9]), in order to extend the applicability of the method to the multivariable case and to robot motion control. A discussion of the modifications needed for trajectory tracking in robot manipulators is included. The main implementation issues of the learning control law are then described and experiments are reported for a geared prototype robot arm. The obtained results will clearly point out the capability of avoiding the occurrence of unstable behavior in the learning process.

II. Learning Control Algorithm

With reference to Fig. 1, let a scalar linear plant be described by its transfer function $p(s)$ between the Laplace transforms of the applied control input u and of the output y. A feedback compensator $c(s)$ is designed for this system, based on the error $e = y_d - y$, where y_d is the reference signal for the output. Typically, the purpose of this controller is the stabilization of the original system and not the accurate tracking of trajectories. The resulting scheme has closed-loop transfer function $w(s)$, given by

$$w(s) = \frac{y(s)}{y_d(s)} = \frac{p(s)c(s)}{1 + p(s)c(s)}. \tag{1}$$

At iteration k, a learning contribution v_k is added to the controller output to yield

$$u_k(s) = u'_k(s) + v_k(s) = c(s)e_k(s) + v_k(s) \tag{2}$$

which is the input applied to the plant. Note that u'_k is the control effort of the feedback controller $c(s)$, which is designed independently from the learning part. The system

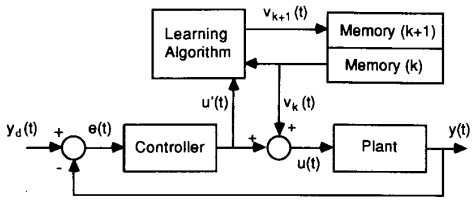

Fig. 1. Overall block diagram of the learning controller.

output can be rewritten as

$$y_k(s) = w(s)y_d(s) + \frac{p(s)}{1 + p(s)c(s)}v_k(s). \tag{3}$$

The learning algorithm is defined by an update law for v_{k+1}, which may depend in general from one or more of the various signals present in the system: v_k, u_k, u'_k, y_k, y_d, and e. Avoiding the use of redundant information, the update v_{k+1} will be chosen as

$$v_{k+1}(s) = \alpha(s)u'_k(s) + \beta(s)v_k(s). \tag{4}$$

The frequency-dependent functions $\alpha(s)$ and $\beta(s)$ will be selected to accelerate the learning process while providing convergence. Manipulating (4), the next iterate takes the form

$$v_{k+1}(s) = \frac{\alpha(s)c(s)}{1 + p(s)c(s)}y_d(s) + (\beta(s) - \alpha(s)w(s))v_k(s). \tag{5}$$

A similar iterative expression holds for the error sequence $e_k = y_d - y_k$:

$$e_{k+1}(s) = \frac{1 - \beta(s)}{1 + p(s)c(s)}y_d(s) + (\beta(s) - \alpha(s)w(s))e_k(s). \tag{6}$$

Due to the repetitive nature of the desired output y_d over successive trails, and to the assumed stable characteristics of the involved functions, sequences (5) and (6) will converge for all values of $s = j\omega$ such that

$$|\beta(s) - \alpha(s)w(s)| < 1. \tag{7}$$

The smaller is the magnitude of the left hand side of (7), the faster will be the rate of convergence. Under this condition, the limit values of the learning memory and of the error are

$$v_\infty(s) = \lim_{k\to\infty} v_k(s) = \frac{y_d(s)}{p(s)} \frac{\alpha(s)w(s)}{1 - \beta(s) + \alpha(s)w(s)} \tag{8}$$

and

$$e_\infty(s) = \lim_{k\to\infty} e_k(s) = \frac{y_d(s)}{1 + p(s)c(s)} \frac{1 - \beta(s)}{1 - \beta(s) + \alpha(s)w(s)}. \tag{9}$$

The contraction condition (7) generalizes the one usually found in the literature, where $\beta(s) \equiv 1$ [7]-[9]. If this choice

is feasible, i.e., if an $\alpha(s)$ can be found such that $|1 - \alpha(s)w(s)| < 1$, one recovers exact reproduction in the limit, or

$$v_\infty^*(s) = \frac{y_d(s)}{p(s)} \qquad e_\infty^*(s) = 0. \tag{10}$$

The role of filter $\beta(s)$ can be illustrated directly using the Nyquist frequency plot. Two typical Nyquist diagrams of $1 - \alpha(j\omega)w(j\omega)$ are reported in Fig. 2. In the first case, stability is guaranteed for all finite ω; in the second, an unstable behavior will occur due to the presence of signal components with $\omega > \omega_0$, and the learning process will diverge over iterations. In the latter case, (7) may be satisfied only by introducing a frequency-dependent $\beta(s)$, designed with the aim of forcing the whole plot inside the unitary circle. Note that if the phase of $\alpha(j\omega)w(j\omega)$ asymptotically reaches $-90°$, then it is not necessary to introduce any filtering action $\beta(s)$. This phase condition is related to the one of strict positiveness of the impulse response of the given linear system, as referred in [8].

The use of $\beta(s)$ is particularly advantageous in the presence of unmodeled destabilizing dynamics. For example, the Nyquist plot of a simple pole dynamics where a small delay is added is shown in Fig. 3, together with the plot obtained by using a lag compensator for $\beta(s)$. With the introduction of $\beta(s)$, the high-frequency content of the closed-loop signals will not be anymore harmful for stability. The price to pay is that learning performance degradates above a certain ω_0. However, this may not be particularly restrictive as the reference signals are usually characterized by useful band of frequencies, beyond which there is no practical interest of reproduction. In any case, accurate tracking of the desired trajectory will be preserved within the specified bandwidth.

The learning process (5) can be equivalently described avoiding reference to a specific trajectory y_d by introducing a "virtual" transfer function between the desired behavior and the memory signal

$$m_k(s) = \frac{v_k(s)}{y_d(s)}. \tag{11}$$

It is easy to see that the following recursion is obtained:

$$m_{k+1}(s) = \frac{\alpha(s)c(s)(1 - p(s)m_k(s))}{1 + p(s)c(s)} + \beta(s)m_k(s). \tag{12}$$

Again, under assumption (7), the limit value for (12) is found as

$$m_\infty(s) = \frac{1}{p(s)} \frac{\alpha(s)w(s)}{1 - \beta(s) + \alpha(s)w(s)}. \tag{13}$$

Using this result, a limit transfer function can be defined as

$$w_\infty(s) = \lim_{k\to\infty} \frac{y_k(s)}{y_d(s)} = w(s) \frac{1 - \beta(s) + \alpha(s)}{1 - \beta(s) + \alpha(s)w(s)}. \tag{14}$$

Equation (14) can be used to quantify—in the frequency range—the amount of introduced approximation in the reproduced reference trajectory, thus representing a useful guide for design purposes and performance evaluation.

When $\beta(s) \equiv 1$, it is easy to see that $w_\infty^*(s) = 1$. Moreover, for any choice of $\alpha(s)$ satisfying the simplified contraction condition, one has

$$m_\infty^*(s) = \frac{1}{p(s)}. \tag{15}$$

Relation (15) expresses the fact that the learning block asymptotically inverts the plant. When $\beta(s)$ is not the identity, the same holds true—at least in a specified frequency range. It is expected that this inversion process may be troublesome for plants having right half-plane zeros. However, the trajectory data from each trial are processed only after execution time and thus inversion is performed off line. In particular, an anticipative (noncausal) learning update can easily be devised by letting the new memory $v_{k+1}(t)$ depend also on error values $e_k(\tau)$ at times $\tau > t$, given the trajectory executed at the kth trial. As a result, the proposed learning strategy is applicable also to tracking problems in systems that display a nonminimum phase behavior, e.g., for the end-point trajectory learning in a one-link flexible arm [14].

Vice versa, whenever the choice

$$\alpha(s) = \frac{1}{w(s)} \tag{16}$$

is a feasible one, then $\beta(s)$ can be safely taken equal to 1 and exact tracking will be achieved after just one trial. Thus, (16) provides the optimal theoretical choice for the learning process. It also confirms the intuitive idea that including the best available model $w_e(s)$ of the closed-loop system in the control-weighting part (i.e., in α) of the learning update (4) will speed up numerical convergence. Note also that the use of a feedback controller $c(s)$ is very advantageous with respect to the ease of choosing a near-optimal $\alpha(s)$. In fact, although the plant transfer function itself $p(s)$ may not be known exactly, the feedback controller $c(s)$ could be designed so as to dominate the overall closed-loop behavior $w(s)$. Accordingly, the objective of a well-designed feedback $c(s)$ should be the robust stabilization of the original plant, together with the largest reduction of the phase lag within a wide band of frequencies.

The learning control algorithm has been introduced with reference to Fig. 1. In this scheme, the learning contribution can be described as a feedforward action relieving the control effort of the feedback controller. An equivalent scheme is obtained if the closed-loop signal extracted for learning and the injected feedforward signal (i.e., u'_k and v_k) are both moved just before the compensator in the direct path. Elementary block algebra shows that the same concept can be realized as in Fig. 4, where the tracking error e_k is used in place of u'_k for learning purposes. In this case, the signal $\hat{v}_k$ stored in the memory is related to the one in Fig. 1 by $\hat{v}_k(s) = v_k(s)/c(s)$. The convenience of such an implementa-

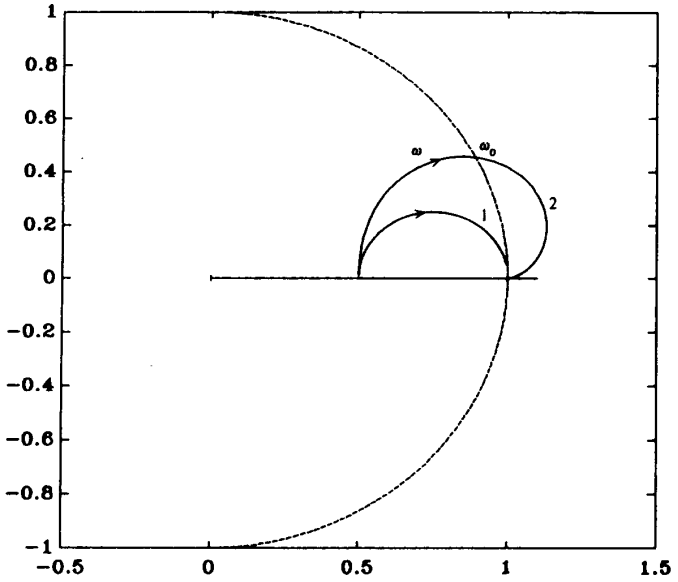

Fig. 2. Plots of $1 - \alpha(j\omega)w(j\omega)$ for stable (1) and unstable (2) learning algorithms.

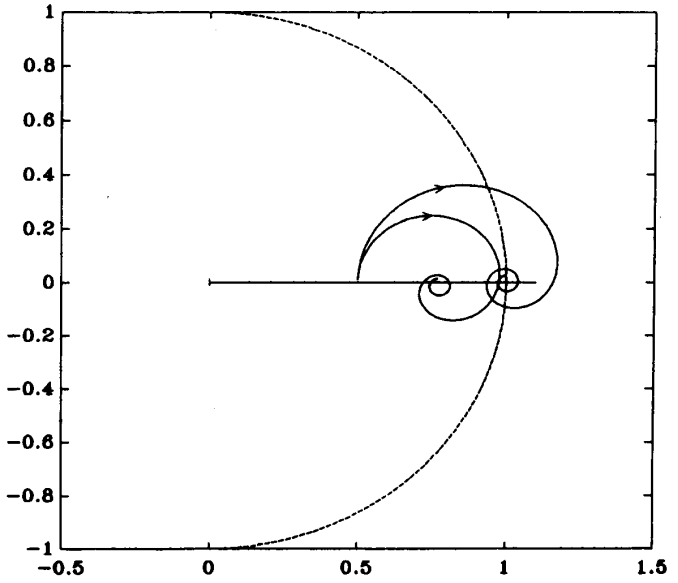

Fig. 3. Stabilization of an unstable learning algorithm using a lag compensator as filter $\beta(j\omega)$.

tion is that the learning controller can now be added without "opening" the previous loop but just by modifying the reference signal of an existing feedback control scheme. Also, Fig. 4 allows to point out a structural difference of the proposed learning scheme in relation with other ones [7], [8] working on an open-loop system or, externally, on a closed-loop one (see the dashed lines). In fact, previous schemes do not have the "direct" link from the reference signal y_d to the closed-loop plant. This further connection allows the proper operation of the system already at the first iteration,

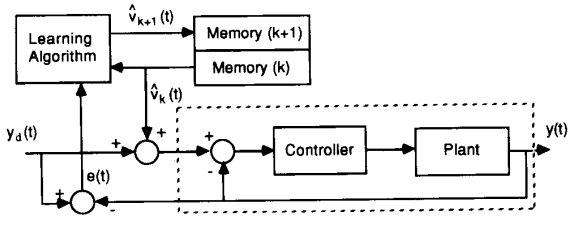

Fig. 4. Modified block diagram of the learning controller.

when the memory content is zero, and increases the robustness of the method with respect to progressive variations of y_d from one iteration to the other.

Finally, the extension of condition (7) to the multivariable case is quite straightforward and can be done following similar steps as in [2]. Let $P(s)$ be the transfer matrix of a square plant (i.e., with the same number of inputs and outputs), and let $C(s)$ be the chosen multivariable precompensator. Denoting by $A(s)$ and $B(s)$ the two learning filters, respectively on the applied on-line control and on the memory, the convergence condition is given by

$$\left\| B(s) - A(s)\left(I + C(s)P(s)\right)^{-1}C(s)P(s)\right\| < 1 \quad (17)$$

where $\|\cdot\|$ is any matrix norm. This expression can be further simplified when, as customary, $A(s)$, $B(s)$, and $C(s)$ are chosen to be diagonal.

III. Application to Robot Trajectory Control

The previous learning scheme is designed using a linear system as the model of the plant. Indeed, the dynamic model of a rigid robot arm with N joints is nonlinear and interacting and can be written as

$$\left(J + B(q)\right)\ddot{q} + \left(D + S(q, \dot{q})\right)\dot{q} + g(q) = u \quad (18)$$

where $q \in R^N$ are the generalized joint coordinates, $J + B(q)$ is the positive definite inertia matrix, $S(q, \dot{q})\dot{q}$ contains the centrifugal and Coriolis terms, $D\dot{q}$ is the viscous friction term (with D diagonal), and $g(q)$ is the gravitational force. In (18), the first two terms have been explicitly written as a linear plus a nonlinear contribution. The constant inertia term J is due both to the inertia of the motors and to the mean average link inertia, as reflected through the gearings.

It is a common practice to design industrial robots using high-geared transmission elements. In this case, the linear dynamics overrides the nonlinear one and J becomes diagonally dominant. Thus, the previous learning technique can be applied directly to this case, provided that the diagonal compensator $C(s)$ is designed as a proportional-derivative controller plus a gravity compensation (see also [9]); in the time domain

$$u'(t) = K_P(q_d - q) + K_D(\dot{q}_d - \dot{q}) + g(q) \quad (19)$$

with $K_P > 0$ and $K_D > 0$. If the joint velocity is available for measurement, the PD part of this control law is implemented as a linear feedback from the full state of the robot arm. The closed-loop system becomes then nearly linear and its input–output behavior in the Laplace domain can be represented by

$$q(s) = \left(Js^2 + (D + K_D)s + K_P\right)^{-1}(K_D s + K_P)q_d(s) \quad (20)$$

or, with diagonal J and using local (decentralized) PD controllers,

$$w_i(s) = \frac{q_i(s)}{q_{d,i}(s)} = \frac{K_{D,i}s + K_{P,i}}{J_i s^2 + (D_i + K_{D,i})s + K_{P,i}} \qquad i = 1, \cdots, N. \quad (21)$$

These expressions are used in the design of filters $\alpha_i(s)$ and $\beta_i(s)$ satisfying condition (7).

There are cases when the foregoing hypothesis of an approximately linear behavior is too unrealistic; typically, the full nonlinear dynamics becomes relevant for robots with open kinematic chains and direct-drive actuators. Even in this case, it is well-known that the feedback law (19) asymptotically stabilizes the robot dynamics around any desired regulation point [15]. Therefore, the overall control strategy will be modified only in the learning algorithm. In particular, if a possibly good approximation of the dynamic model is available, a feedforward term can be used to reduce the learning effort. This term is introduced as the initialization of the learning memory of the controller. At the first iteration ($k = 1$)

$$v_1(t) = \left(J_e + B_e(q_d)\right)\ddot{q}_d + \left(D_e + S_e(q_d, \dot{q}_d)\right)\dot{q}_d + g_e(q_d) \quad (22)$$

is used, where the subscript e denotes the "best" estimated model. This model may differ from the correct one because of the inexact knowledge of system parameters, as well as for the simplifications introduced in evaluating the dynamic terms. Note that there are no specific restrictions on this initial guess, which may even be very poor. A general convergence analysis which is applicable to this approach can be found, e.g., in [18].

IV. Implementation Aspects

A hardware/software environment suitable for real-time experimentation of robot control laws has been developed at the Robotics Laboratory in our Department. The system is constituted by 6-degree of freedom (dof) prototype manipulator and a multimicro process computer. Fig. 5 illustrates the main building blocks of the system.

The robot arm has a symmetric distribution of masses and no shoulder or elbow lateral displacements. All joints are revolute and are actuated by dc motors through harmonic drives with transmission ratios equal to 160. The inertial parameters and the dynamic model of the robot arm, with the last three joints held fixed, are given in [16]. Approximately 90% of the gravity loading is compensated by means of a system of springs. Motors are powered by current amplifiers, whose reference values are provided by 12-b D/A converters. Each joint is equipped with a resolver and a dc tachometer for velocity feedback. The analog outputs of both sensors are converted into digital values with a resolution of 16-b/2π rad and, respectively, 11-b/(rad/s).

The multimicro architecture is composed by an IBM XT286 personal computer using digital I/O ports to communicate with a board based on a Texas TMS 32025 Digital Signal Processor (DSP) with 4K words of RAM program memory and 2K words of data memory; the DSP computer is directly interfaced to the converters of the robot (see [17] for more details).

In this hierarchical structure, the XT286 performs the high-level control actions and provides the system mass memory and graphic user interfaces. All functions are pro-

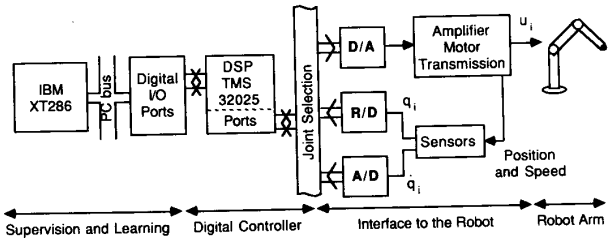

Fig. 5. Functional blocks of the experimental environment.

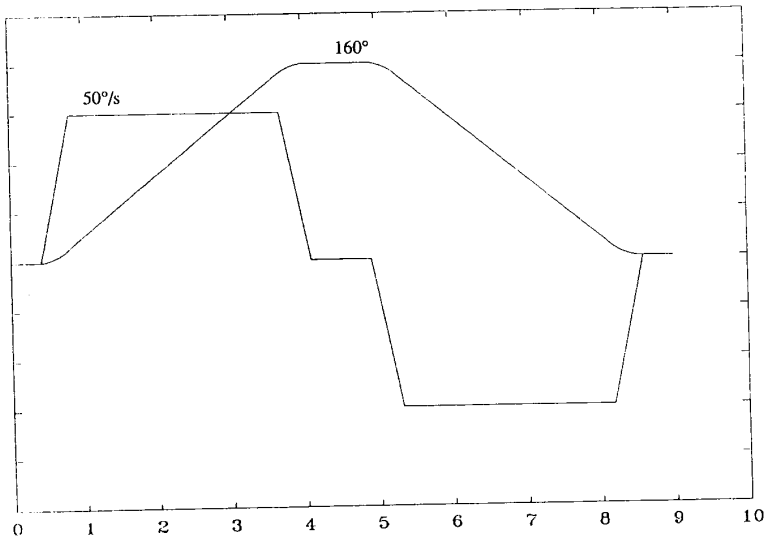

Fig. 6. Speed and position profile of the joint trajectories.

grammed in Pascal; in particular, the trajectory generation is done at this level. Moreover, a programming environment for the TMS 32025 is supported, including an editor and a C cross-compiler. Executable codes are downloaded to the DSP through the same channel used for real-time communications.

The proposed control law has been implemented on this architecture with the specific goal of testing the performance of the learning part. A simple linear state-feedback has been used as the closed-loop controller of Fig. 1, neglecting the nonlinear term $g(q)$ in (19), as gravity effects are almost entirely compensated by the mechanical design.

The closed-loop linear control is performed by the DSP at a sampling rate of 400 μs. The most relevant system variables (among the others, position and velocity errors and the output u' of the linear controller) are stored every other sampling instant in a local buffer and transferred to the XT286 every 20 ms, the sampling time of this computer. These data are progressively saved in a large buffer in the PC RAM. At the same rate, the proper segments of the time-varying reference signals $y_d(t)$ and $v_k(t)$ are downloaded. At the end of each trial, the PC buffer is processed off-line and the new learning output $v_{k+1}(t)$ is computed according to (4).

We remark that the overall control law has a natural decomposition that matches perfectly with the levels of the available computing structure. As a matter of fact, the principle of simple, fast low-level modules governed by devices devoted to more complex but slow rate tasks is typical of most industrial controllers.

V. Experimental Results

Several experiments were performed for evaluating the tracking performance of the learning controller. The results reported here refer to a trajectory specified for two of the six axes of the prototype arm. Only joints 2 and 3 are required to move, so that the desired motion of the arm is restricted to a vertical plane. The reference trajectory is defined in the joint space and is composed of two trapezoidal velocity profiles (see Fig. 6). Each joint moves back and forth by approximately 160°, extending the arm symmetrically around the upward stretched configuration. The maximum speed reached is 50°/s for both joints, while the maximum accelerations are 40 and 50°/s^2 for joints 2 and 3, respectively. The total traveling time is about 9 s. These values are close to the maximum allowable velocity and acceleration for this robot, which are specified respectively as 60°/s and 60°/s^2.

Table I contains the PD gains used for the feedback controller and the estimated link inertia and joint viscous friction. These coefficients should be used in the linear dynamic model (21). In the same table, the actual values of the motor torque constants $K_{m,i}$ are also included. In all experiments, the learning memory is initialized as $v_1 = 0$, i.e., no feedforward is used at the first trial.

Figs. 7 and 8 show the profiles of the position errors of

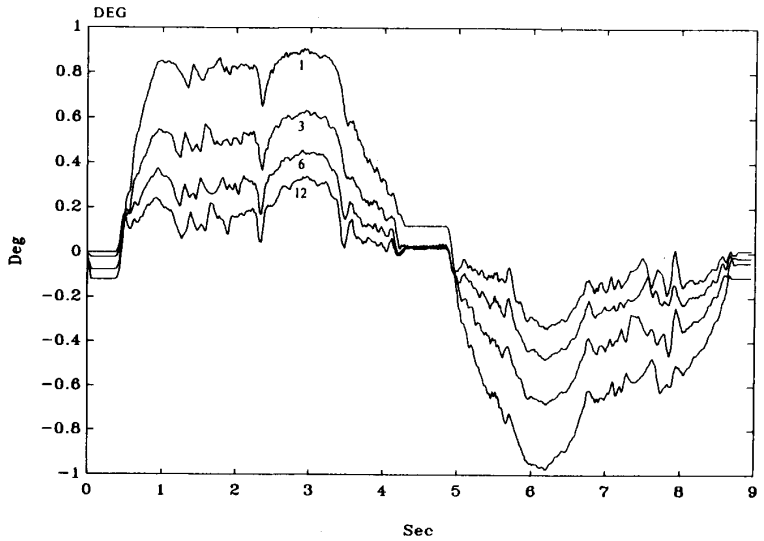

Fig. 7. Position errors for joint 2 in trials 1, 3, 6, and 12.

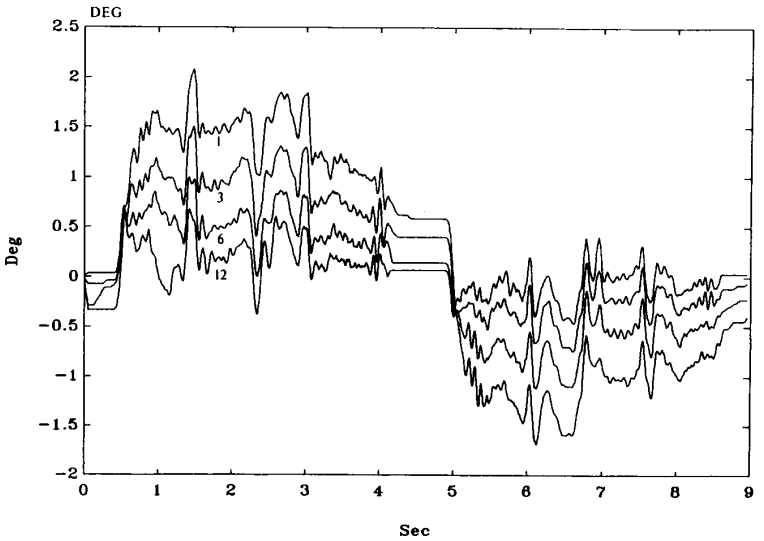

Fig. 8. Position errors for joint 3 in trials 1, 3, 6, and 12.

TABLE I
VALUES OF THE RELEVANT PARAMETERS FOR THE PD CONTROL LOOPS

Parameter	Joint 2	Joint 3	Units
J_i	82.5	26	kg m^2
D_i	130	100	N · m/(rad/s)
$K_{P,i}$	22.7	47.2	A/rad
$K_{D,i}$	20.4	12.3	A/(rad/s)
$K_{M,i}$	45.7	27.5	N · m/A

joint 2 and 3 along the given trajectory, as the learning proceeds. Data refers to trials 1, 3, 6, and 12, where it is evident that the amplitude of the error has been greatly reduced. A more quantitative information can be obtained from Fig. 9, which shows the root mean square (rms) value of the errors against the trial number. The convergence of the error to zero is quite regular and follows the expected asymptotic behavior, despite the high level of static (dry) friction present at the joints of this manipulator. The stiction phenomenon is quite relevant in our case, vanishing the efforts for an accurate modeling of robot links dynamics and limiting also the effectiveness of model-based controllers, e.g., the computed torque method [19]. Note that static friction cannot be assumed as a strictly repetitive disturbance and also causes slightly different initial conditions in successive trials. Despite this, the learning controller behaves in a quite robust way even with respect to these kind of problems. If an integral-type controller were used instead, it would result in oscillations around the final set point.

Fig. 10 displays the evolution of the learning memory v_k and of the feedback control action u'_k for joint 2; the same quantities are shown for joint 3 in Fig. 11. From these, it can be seen that the closed-loop effort is transferred to feedfor-

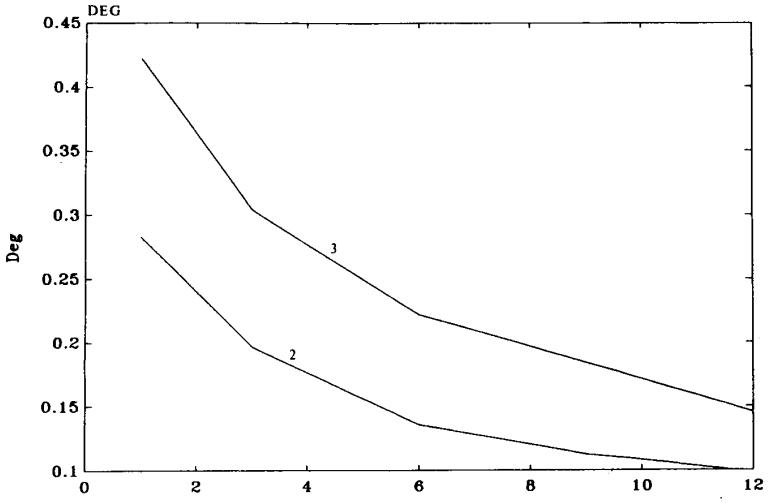

Fig. 9. RMS values of position errors vs. trial number.

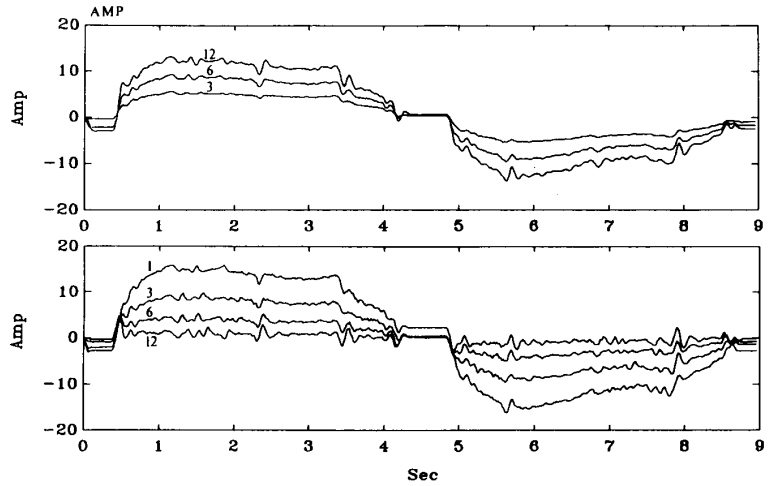

Fig. 10. Memory control v_k (upper trace) and feedback control u'_k (lower trace) in trials 1, 3, 6, and 12, for joint 2.

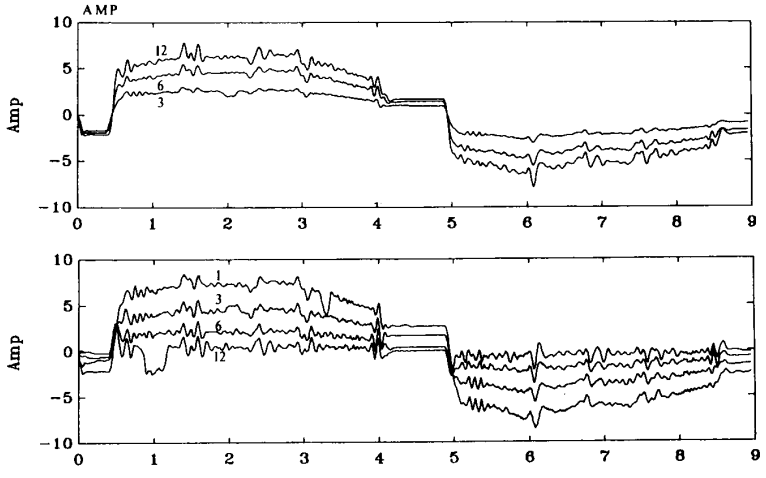

Fig. 11. Memory control v_k (upper trace) and feedback control u'_k (lower trace) in trials 1, 3, 6, and 12, for joint 3.

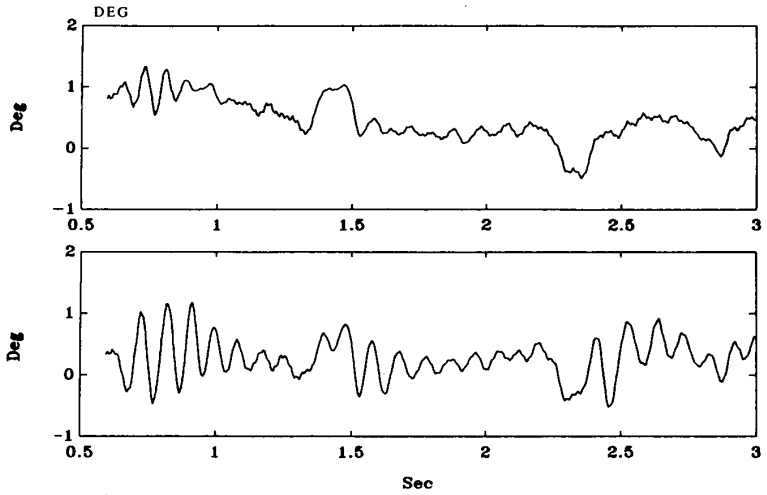

Fig. 12. Comparison of position errors at iteration 12 for joint 3, with filter $\beta(j\omega)$ (upper trace) and without it (lower trace).

ward action as the number of iterations increases. Although the memory is learning the required input, the error along the trajectory is driven to zero together with the closed-loop control. At iteration 12, only high-frequency ripples are left in the feedback control u'_{12}, which will not be learned thanks to the presence of filter $\beta(s)$. The residual errors are mainly due to the inconsistency between velocity and position measurements, since two different sensors are used. These effects are very small indeed, namely with peaks of the order of 0.5°.

All the above results were obtained using, for both joints, the learning filters

$$\alpha(s) = 0.25 \qquad \beta(s) = \frac{1}{1 + 0.008s}. \tag{23}$$

The chosen low-pass filter $\beta(s)$ gives a cutoff frequency of 20 Hz. The selection of a constant value for $\alpha(s)$ is a conservative one, as it takes no advantage of the knowledge about the closed-loop system, as would be suggested instead by (16).

In order to verify the benefits on the overall stability induced by the introduction of $\beta(s)$, a set of experiments were performed setting $\beta(s) \equiv 1$. After an initial period of learning (10 iterations) some small oscillations appear. Fig. 12 compares the position errors of joint 3 at iteration 12, obtained with and without the filter $\beta(s)$. The lower trace clearly reports a high-frequency oscillation superimposed to the useful signal. This phenomenon has been already observed experimentally in [6], where a deadband on the tracking error was used to avoid instabilities.

VI. Conclusions

A new frequency-domain learning algorithm has been presented that properly handles the presence of destabilizing dynamic signals and guarantees convergence of the iterative process. The rationale behind this approach stands in the clear separation between learning capabilities of the algorithm and convergence issues. The frequency shaping of the algorithm does not affect adversely the number of required trials but only the bandwidth where the information content of the desired behavior will be reproduced.

The learning scheme can be added in parallel to an existing feedback controller in order to improve performance. The analysis has shown that starting with a closed-loop system yields relaxed convergence conditions that can be more easily satisfied. The learning technique has been implemented for the trajectory tracking control problem in robot manipulators. The benefits gained by adding a filtering device on the learning memory, and in particular the capability of rejecting an oscillatory behavior buildup, were clearly illustrated by the experimental results. The obtained rate of convergence is quite good, even if no a priori knowledge on the dynamic model was initially stored in the learning memory. Although the reported experiments refer to robotics, this should be seen as a case study: the learning algorithm is of general application to all those processes for which high-performance but still simple controllers are needed in repetitive tasks or in the presence of periodic disturbances.

Further research includes a discrete-time analysis of the algorithm, suited for the case of truly nonlinear models of robot manipulators. The discrete-time approach is also convenient for thoroughly exploiting the possibilities of noncausal filtering design.

Acknowledgments

Comments of an anonymous referee are gratefully acknowledged.

References

[1] M. Tomizuka, T. C. Tsao, and K. K. Chew, "Discrete-time domain analysis and synthesis of repetitive controllers," in *Proc. 1988 American Control Conf.*, Atlanta, GA, 1988, pp. 860–866.

[2] S. Hara, Y. Yamamoto, T. Omata, and M. Nakano, "Repetitive control system: A new type servo system for periodic exogenous signals," *IEEE Trans. Automat. Contr.*, vol. AC-33, no. 3, pp. 659–668, 1988.

[3] T. Mita and E. Kato, "Iterative control and its application to motion

control of robot arm—A direct approach to servo problems -," in *Proc. 24th IEEE Conf. on Decision and Control*, Ft. Lauderdale, FL, 1985, pp. 1393-1398.

[4] L. M. Hideg and R. P. Judd, "Frequency domain analysis of learning systems," in *Proc. 27th IEEE Conf. on Decision and Control*, Austin, TX, 1988, pp. 586-591.

[5] M. Togai and O. Yamano, "Learning control and its optimality: Analysis and its application to controlling industrial robots," in *Proc. 1986 IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, 1986, pp. 248-253.

[6] M. C. Tsai, G. Anwar, and M. Tomizuka, "Discrete-time repetitive control for robotic manipulators," in *Proc. 1988 IEEE Int. Conf. on Robotics and Automation*, Philadelphia, PA, 1988, pp. 1341-1346.

[7] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *J. Robotic Syst.*, vol. 1, pp. 123-140, 1984. 1984.

[8] S. Arimoto, S. Kawamura, F. Miyazaki, and S. Tamaki, "Learning control theory for dynamical systems," in *Proc. 24th IEEE Conf. on Decision and Control*, Ft. Lauderdale, FL, 1985, pp. 1375-1380.

[9] S. Arimoto, F. Miyazaki, and S. Kawamura, "Motion control of robotic manipulator based on motor program learning," *Prepr. 2nd IFAC Symp. on Robot Control (SYROCO'88)*, Karlsruhe, W. Germany, Oct. 1988.

[10] P. Bondi, G. Casalino, and L. Gambardella, "On the iterative learning control theory of robotic manipulators," *IEEE Trans. Robotics and Automation*, vol. RA-4, pp. 14-22, 1988.

[11] C. G. Atkeson and J. McIntyre, "Robot trajectory learning through practice," in *Proc. 1986 IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, 1986, pp. 1737-1742.

[12] J. Hauser, "Learning control for a class of nonlinear systems," in *Proc. 26th IEEE Conf. on Decision and Control*, Los Angeles, CA, 1987, pp. 859-860.

[13] T. Omata, S. Hara, and M. Nakano, "Nonlinear repetitive control with application to trajectory control of manipulators," *J. Robotic Systems*, vol. 4, pp. 631-652, 1987.

[14] M. Poloni and G. Ulivi, "Iterative trajectory tracking for flexible arms with approximate models," *5th Int. Conf. on Advanced Robotics (ICAR'91)*, Pisa, Italy, 1991, pp. 108-113.

[15] S. Arimoto and F. Miyazaki, "Stability and robustness of PID feedback control for robot manipulators of sensory capability," in *1st Int. Symp. on Robotics Research*, M. Brady and R. P. Paul, Eds., pp. 783-799, MIT Press, Cambridge, MA, 1984.

[16] A. Bellini, G. Figalli, P. Pinello, and G. Ulivi, "Realization of a control device for a robotic manipulator based on nonlinear decoupling and sliding mode control," *IEEE Trans. Industry Applications*, vol. 25, pp. 790-799, 1989.

[17] S. Battilotti and G. Ulivi, "An architecture for high performance control using digital signal processor chips," *Control Syst. Mag.*, vol. 10, no. 6, pp. 20-23, 1990.

[18] R. Horowitz, W. Messner, and J. B. Moore, "Exponential convergence of a learning controller for robot manipulators," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 890-894, 1991.

[19] W. E. Snyder, *Industrial Robots: Computer Interfaces and Control*. Englewood Cliffs, NJ: Prentice Hall, 1985.