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

A. De Luca, G. Paesano, G. Ulivi

Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza"
Via Eudossiana 18, 00184 Roma, Italy

Abstract

A frequency-domain approach to the analysis and design of a learning control law for linear dynamical systems is presented. In its most simple version the scheme uses two separate filters in order to achieve rapid improvements in a specified bandwidth while cutting-off – perhaps unmodeled – dynamic effects which would bar the convergence. The merit of this approach is to make explicit the trade-off between global convergence conditions and approximate learning of trajectories. The proposed learning controller can be applied also to robot manipulators for exact tracking of repetitive trajectories. The scheme has been successfully implemented for learning the actuator inputs which enable accurate reproduction of robot trajectories defined in the joint-space. Experimental results are reported which show the efficacy of this learning scheme also in avoiding the possible occurrence of an unstable behavior.

Introduction

Learning is a control technique where the required input to a given system is built iteratively from successive experiments. Performance on repetitive tasks is improved from one iteration to the other, until the desired behavior is attained.

A main advantage of learning control is that it works with limited a priori knowledge of the system. The modeling effort can be restricted to system dynamics which are known exactly or have small parametric uncertainties. 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 hypothesis about the repetitive nature of the desired behavior. No explicit parameter identification is performed and the learning process should be restarted when tackling a different task. On the other hand, the scheme can be made robust with respect to the effects of unmodeled dynamics and of a wide class of disturbances.

Another relevant aspect of learning schemes is the relative simplicity of the resulting control law, enabling a cheap implementation without heavy real-time computing burden. The learning controller may also be easily added in parallel to ("plugged-in") an already existing controller designed for the given system, so to produce an overall performance improvement. This allows to relax the requirements of a purely model-based approach.

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 which guarantee convergence of the iterative schemes were performed either in the frequency [1-3] or in the discrete [4-6] or continuous time [7,8] domain. For linear time-varying systems of dynamic order two (called acceleration systems) different simple learning laws have been studied, based on the PI, PD, or PID treatment of the velocity error; the last two require also an acceleration feedback [9].

Extensions to specific classes of nonlinear systems were also considered [9-13]. These include mechanical systems like robot arms, which possess the relevant property of being both stabilizable by a linear PD law and feedback linearizable by a nonlinear state feedback. The proof of convergence is more involved, but the resulting control schemes are 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. A feedback controller of the plant is explicitly included in the analysis of the overall convergence. The convergence conditions will be defined with respect to the closed-loop behavior, which is much more convenient and predictable than the open-loop one.
2. The method allows for different weighting of "what is to be learned" from the current trial and "what to remember" from the previous ones. A small amount of additional signal processing is needed which is used to enforce convergence by cutting off high-frequency system dynamics. In a robot arm, these could be originated by the elasticity in the joint transmissions.
3. The approach is frequency-oriented and provides deeper engineering insight during the design phase. Tools like frequency response, filtering and Nyquist plots are of direct use.

The paper is organized as follows. For the sake of simplicity, the method is presented for the special case of a SISO linear system, subject to feedback from the output. Yet, the peculiarities of the proposed approach over previous ones can be fully illustrated there. In order to extend the applicability of the method also to the multivariable case and to robot motion control, it is easy to incorporate the basic idea into other existing analyses (see e.g. [2],[6] and [9]). A discussion of the modifications needed for trajectory learning in robot manipulators is included. The major aspects of implementation of this learning control for a geared prototype arm are then described and the experimental results are reported. The performance of the proposed approach are evaluated pointing out explicitly the capability of avoiding the possible occurrence of unstable behavior in the learning process.

Learning Control Algorithm

With reference to Fig. 1, let the linear plant be described by its transfer function $p(s)$ between the Laplace transforms of the applied scalar control input $u(s)$ and of the scalar output $y(s)$. A feedback

TH0282-4/89/0000/0066$01.00 © 1989 IEEE

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. The purpose of this controller is mainly stabilization of the original system and not accurate tracking. The resulting scheme has a closed-loop transfer function $w(s)$ given by

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

The learning contribution v_k at iteration k is added to the plant input so to yield

$$u(s) = u'(s) + v_k(s) = c(s)e(s) + v_k(s) \quad (2)$$

which is the input applied on-line to the plant at iteration k. Note that u' is the control effort of the feedback controller $c(s)$ which is typically designed apart from the learning procedure. The system output can be rewritten as

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

The update law for the learning process may depend in general from one or more of the various signals which are present in the system: v_k, u, u', y, y_d, and e. Avoiding the use of redundant information, the update v_{k+1} will be chosen as

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

The frequency-dependent functions $\alpha(s)$ and $\beta(s)$ are selected to accelerate the learning process while providing convergence. The next iterate can be found manipulating (4)

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

Due to the repetitive nature of the desired output, the iterations (5) will converge for all values of $s = j\omega$ such that

$$|\beta(s) - \alpha(s)w(s)| < 1. \quad (6)$$

The above contraction condition generalizes the one found in the literature, where $\beta(s) \equiv 1$. The role of the filter $\beta(s)$ can be illustrated directly using the Nyquist frequency plot. Two typical Nyquist plots of $1 - \alpha(j\omega)w(j\omega)$ are reported in Fig. 2. In the first case, stability is guaranteed for all $\omega < \infty$; in the second, unstable behavior will occurr for $\omega > \omega_0$. In this case, (6) may be satisfied only by introducing a frequency-dependent $\beta(s)$ which will force the whole plot inside the unitary circle. Note that if the phase of $\alpha(j\omega)w(j\omega)$ asymptotically reaches -90° it is not necessary to introduce the 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 effect of introducing $\beta(s)$ is particularly interesting in presence of unmodelled 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 as $\beta(s)$. The high-frequency content (ideally at ∞) of the closed-loop signals is not harmful anymore for stability. Indeed, the price to pay is that the learning performance degradates above a certain ω^*. This is not too restrictive since in the applications also the reference signals are characterized by an useful band of frequencies, beyond which there is no practical interest in reproduction. In any case, accurate tracking of the desired trajectory will be mantained within the specified bandwidth.

The learning process (5) may be equivalently described as the update of a relationship between the desired output y_d and v_k

$$v_k(s) = m_k(s)y_d(s), \quad (7)$$

so that the following recursion is obtained:

$$m_{k+1}(s) = \frac{\alpha(s)c(s)[1 - p(s)m_k(s)]}{1 + p(s)c(s)} + \beta(s)m_k(s). \quad (8)$$

Under assumption (6), the limit value of (8) is found as

$$\lim_{k\to\infty} m_k(s) = \frac{1}{p(s)}\frac{\alpha(s)w(s)}{1 - \beta(s) + \alpha(s)w(s)}. \quad (9)$$

For all $\omega \leq \omega^*$ such that $\beta(j\omega) \equiv 1$, it is easy to see that

$$\lim_{k\to\infty} m_k(s) = \frac{1}{p(s)} \quad (10)$$

for any $\alpha(s)$. The above relation express the fact that asymptotically the learning block is inverting the plant, at least in the specified frequency range. It is expected that this process may be troublesome for plants having right half-plane zeros. However, the inversion process is done off-line, processing the data of each trial along the given trajectory only after its execution. Therefore, a non-causal inversion update can be easily devised; moreover, the scheme is intrinsically limited in frequency. As a result, the learning strategy could be used also for end-point trajectory learning of a flexible one-link arm, which displays a typical non-minimum phase behavior.

It is interesting to see that, if the choice

$$\alpha(s) = \frac{1}{w(s)} \quad (11)$$

is a feasible one, then $\beta(s)$ can be taken equal to 1 and exact tracking would be achieved after just one trial. Thus, (11) provides the theoretical optimal choice for the learning process. It also confirms the intuitive idea that including the best available model of the closed-loop system in the control-weighting part (i.e. α) of the learning update speeds up the numerical convergence. Note that the use of a feedback controller $c(s)$ is very advantageous with respect to the ease of choice of a near-optimal $\alpha(s)$. In fact, although the plant transfer function itself $p(s)$ may not be known exactly, the (known) feedback controller $c(s)$ could be designed so as to shape the overall closed-loop behavior $w(s)$. According to this, 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 extension of condition (8) 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 pre-compensator. 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

$$\|B(s) - A(s)[I + C(s)P(s)]^{-1}C(s)P(s)\| < 1 \quad (12)$$

where $\|.\|$ 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.

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

$$[J + B(q)]\ddot{q} + [D + S(q,\dot{q})]\dot{q} + g(q) = u \qquad (13)$$

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 damping term (with D diagonal), and $g(q)$ is the gravitational force. In (13), 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, e.g. harmonic drives. In this case, the linear dynamics overrides the nonlinear one and J is diagonally dominant. Thus, the previous learning technique can be applied directly to this case, provided that the compensator C is designed as a proportional-derivative controller plus a gravity compensation (see also [9]); in the time domain

$$u'(t) = K_P(q_d(t) - q(t)) + K_D(\dot{q}_d(t) - \dot{q}(t)) + g(q(t)). \quad (14)$$

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 is nearly linear and its input-output behavior in the Laplace domain can be represented by

$$q(s) = [Js^2 + (D + K_D)s + K_P]^{-1}[K_D s + K_P]q_d(s) \quad (15)$$

or, with J diagonal and using local PD controllers,

$$w_i(s) = \frac{q_i(s)}{q_{d,i}(s)} = \frac{K_{D,i}s + K_{P,i}}{J_i s^2 + (D_i + K_{D,i})s + K_{P,i}}, \qquad (16)$$

with $i = 1, \cdots, N$. These expressions are used for the design of the filters $\alpha_i(s)$ and $\beta_i(s)$ satisfying condition (6).

There are cases when the above 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 (14) asymptotically stabilizes the robot dynamics around any regulation point [14]. 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 ($k = 1$)

$$v_1 = [\hat{J} + \hat{B}(q_d)]\ddot{q}_d + [\hat{D} + \hat{S}(q_d,\dot{q}_d)]\dot{q}_d + \hat{g}(q_d) \qquad (17)$$

is used, where the hat denotes the "best" a priori model. This model differs from the correct one, both because of the inexact knowledge of the system parameters and because of the possible simplifications introduced for 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 in [12].

Implementation Aspects

A hardware/software environment suitable for real-time experimentation of robot control laws has been developed in our Laboratory. The system is constituted by a 6-dof prototype manipulator and a multimicro process computer. Fig. 4 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 [15]. 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-bit 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-bit/2π rad and, respectively, 11-bit/(rad/sec).

The multimicro architecture is composed by an IBM XT286 personal computer equipped with digital I/O ports communicating with a board based on a Texas TMS 32025 Digital Signal Processor with 4K words of RAM program memory and 2K words of data memory; the DSP computer is interfaced to the converters of the robot (see [16] 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 programmed 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. The simple linear state-feedback (14) has been used as the closed-loop controller of Fig. 1. Since the gravity effects are almost entirely compensated by the mechanical design, no nonlinear feedback $g(q)$ is present.

The closed-loop linear control is performed by the DSP with a sampling time of 400 μsec. 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 msec, 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 segment of the time-varying reference signals $y_d(t)$ and $v_k(t)$ is 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).

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 just 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. 5). Each joint moves back and forth by approximately 160°, extending the arm simmetrically around the upward stretched configuration. The maximum speed reached is 50°/sec for both joints, while the maximum accelerations are 40 and

$50°/sec^2$ for joints 2 and 3, respectively. The total traveling time is about 9 sec.

Table 1 contains the PD gains used for the feedback controller and the estimated link inertia and of friction. These coefficients should be used in the linear dynamic model (16). In the same Table, the actual values of the motor torque constants $K_{m,i}$ are included. The initialization of the learning memory is chosen as $v_1 = 0$, i.e. no feedforward is used at the first trial.

Figs. 6 and 7 show the profiles of the position errors of joints 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. 8 which shows the Root Mean Square value of the errors against the trial number. The convergence of the error to zero is quite regular and with the expected asymptotic behavior, despite the high level of static (dry) friction present at the joints of this manipulator. Note that static friction cannot be assumed as a strictly repetitive disturbance and causes also 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.

Fig. 9 displays the evolution of the learning memory v_k and of the feedback control action u' for joint 2; the same quantities are shown for joint 3 in Fig. 10. From these results it can be seen that the closed-loop effort is progressively transferred to the feedforward control. While the memory is learning the required input, the error along the trajectory is driven to zero together with the closed-loop action. At iteration 12, only high-frequency ripples are left in the feedback control u', which will not be learnt due to the presence of the 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 as filters

$$\alpha(s) = 0.25, \quad \beta(s) = \frac{1}{1 + 0.008\,s}.$$

The chosen low-pass filter $\beta(s)$ gives a cut-off frequency of 20 Hz. The selection of a constant value for $\alpha(s)$ is a conservative one, since it takes no advantage of knowledge about the closed-loop system, as would be suggested instead by eqn. (11).

In order to verify the benefit 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. 11 compares 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.

Conclusions

A new frequency-domain learning algorithm has been presented and implemented for the accurate tracking control of robot manipulators. The rationale behind this approach stands in the possibility of a clear separation in the analysis between the learning capabilities of the algorithm and the stability issues. The learning scheme can be added in parallel to an existing feedback controller to improve performance. Starting with a closed-loop system may also yield relaxed convergence conditions. The frequency shaping of the algorithm does not affect adversely the number of required trials, but only the bandwidth within which the information content of the desired behavior will be reproduced.

The positive effects gained by adding a filtering device on the learning memory were clearly illustrated by the experimental results, which also pointed out the occurrence of instabilities in the learning process when this filter is turned off. In the application to the robot, the obtained rate of convergence is quite good although no a priori knowledge on the dynamic model was stored in the learning memory.

Although the implementation of the learning control is naturally a digital one, the whole approach has been described in the continuous time. 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 could be also convenient for exploiting the possibilities of a non-causal filter design.

References

[1] S. Hara, Y. Yamamoto, T. Omata, and M. Nakano, "Repetitive Control System: A New Type Servo System for Periodic Exogenous Signals", *IEEE Trans. Autom. Control*, vol. **AC-33**, 7, pp. 659–668, 1988.

[2] T. Mita and E. Kato, "Iterative Control and Its Application to Motion Control of Robot Arm - A Direct Approach to Servo Problems -", *24th IEEE Conf. on Decision and Control*, pp. 1393–1398, Ft. Lauderdale, FL, Dec. 1985.

[3] L. M. Hideg and R. P. Judd, "Frequency Domain Analysis of Learning Systems", *27th IEEE Conf. on Decision and Control*, pp. 586–591, Austin, TX, Dec. 1988.

[4] M. Tomizuka, T. C. Tsao, and K. K. Chew, "Discrete-Time Domain Analysis and Synthesis of Repetitive Controllers", *1988 American Control Conf.*, pp. 860–866, Atlanta, GA, Jun. 1988.

[5] M. Togai and O. Yamano, "Learning Control and Its Optimality: Analysis and Its Application to Controlling Industrial Robots", *3rd IEEE Int. Conf. on Robotics and Automation*, pp. 248–253, S. Francisco, CA, Apr. 1986.

[6] M. C. Tsai, G. Anwar, and M. Tomizuka, "Discrete-Time Repetitive Control for Robotic Manipulators", *5th IEEE Int. Conf. on Robotics and Automation*, pp. 1341–1346, Philadephia, PA, Apr. 1988.

[7] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering Operation of Robots by Learning", *J. of Robotic Systems*, vol. **1**, 2, pp. 123–140, 1984.

[8] S. Arimoto, S. Kawamura, F. Miyazaki, and S. Tamaki, "Learning Control Theory for Dynamical Systems", *24th IEEE Conf. on Decision and Control*, pp. 1375–1380, Ft. Lauderdale, FL, Dec. 1985.

[9] S. Arimoto, F. Miyazaki, and S. Kawamura, "Motion Control of Robotic Manipulator Based on Motor Program Learning", *2nd IFAC Symp. on Robot Control (SYROCO'88)*, Karlsruhe, FRG, Oct. 1988.

[10] G. Casalino and L. Gambardella, "Learning of Movements in Robotic Manipulators", *3rd IEEE Int. Conf. on Robotics and Automation*, pp. 572–578, S. Francisco, CA, Apr. 1986.

[11] C. G. Atkeson and J. McIntyre, "Robot Trajectory Learning Through Practice", *3rd IEEE Int. Conf. on Robotics and Automation*, pp. 1737–1742, S. Francisco, CA, Apr. 1986.

[12] J. Hauser, "Learning Control for a Class of Nonlinear Systems", *26th IEEE Conf. on Decision and Control*, pp. 859–860, Los Angeles, CA, Dec. 1987.

[13] T. Omata, S. Hara, and M. Nakano, "Nonlinear Repetitive Control with Application to Trajectory Control of Manipulators", *J. of Robotic Systems*, vol. **4**, 5, pp. 631–652, 1987.

[14] S. Arimoto and F. Miyazaki, "Stability and Robustness of PID Feedback Control for Robot Manipulators of Sensory Capability", *1st Int. Symp. on Robotics Research*, M. Brady and R. P. Paul Eds., pp. 783–799, MIT Press, Cambridge, 1984.

[15] 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", *1987 IEEE Ind. Applic. Soc. Annual Meeting*, pp. 1288–1294, Atlanta, GA, Oct. 1987.

[16] S. Battilotti and G. Ulivi, "A Multimicro System Designed for High Performance Control Applications", *15th IEEE Ind. Electronics Soc. Annual Conf.*, Philadelphia, PA, Nov. 1989.

This work was partially supported by CNR *Progetto Finalizzato Robotica*.

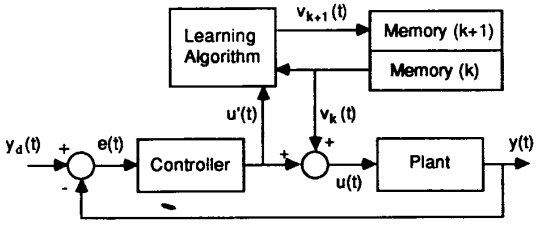

Fig. 1 – Overall block diagram of the learning control scheme

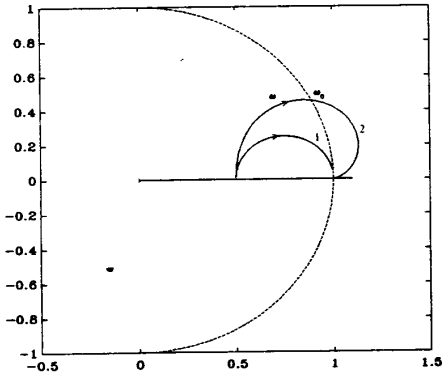

Fig. 2 – Plots of $1 - \alpha(j\omega)w(j\omega)$ for stable (1) and unstable (2) learning algorithms

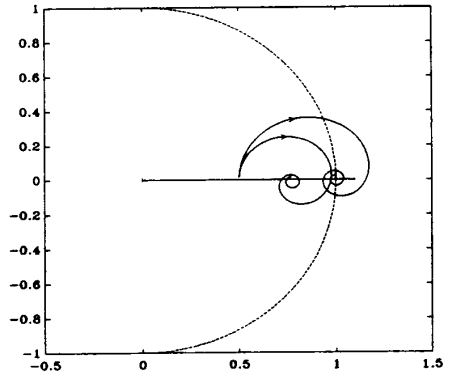

Fig. 3 – Stabilization of an unstable learning algorithm using a lag compensator as filter $\beta(j\omega)$

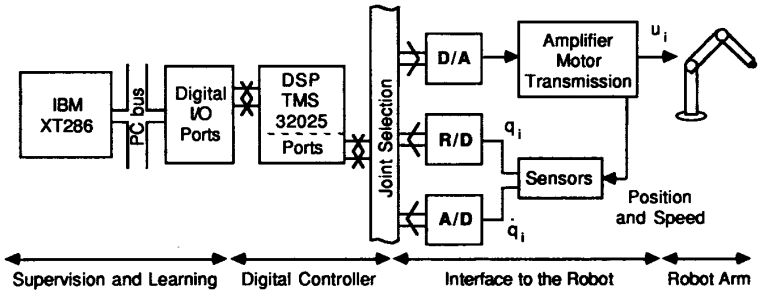

Fig. 4 – Block diagram of the experimental environment

	joint 2	joint 3	units
J_i	82.5	26	$Kg\,m^2$
D_i	130	100	Nm/(rad/sec)
$K_{P,i}$	22.7	47.2	Amp/rad
$K_{D,i}$	20.4	12.3	Amp/(rad/sec)
$K_{M,i}$	45.7	27.5	Nm/Amp

Table 1 – Values of the relevant parameters for the PD control loops

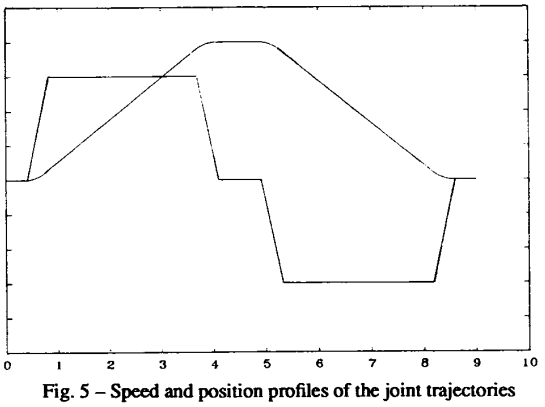

Fig. 5 – Speed and position profiles of the joint trajectories

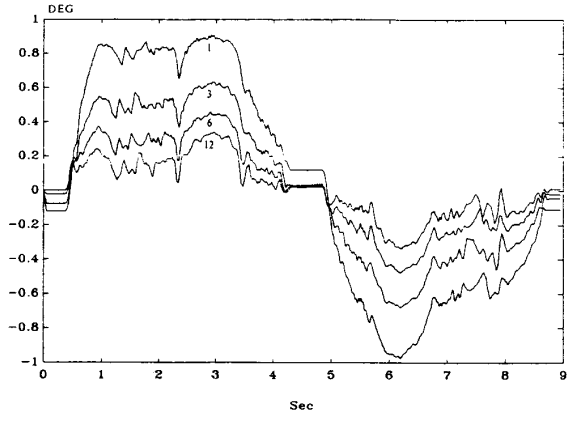

Fig. 6 – Position errors for joint 2, in trials 1, 3, 6 and 12

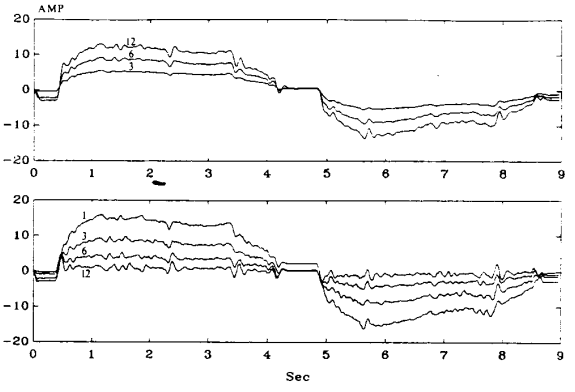

Fig. 9 – Memory control v_k (upper trace) and feedback control u' (lower trace), in trials 1, 3, 6 and 12, for joint 2

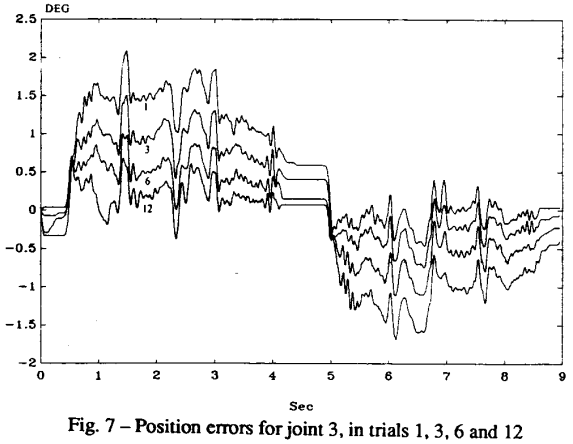

Fig. 7 – Position errors for joint 3, in trials 1, 3, 6 and 12

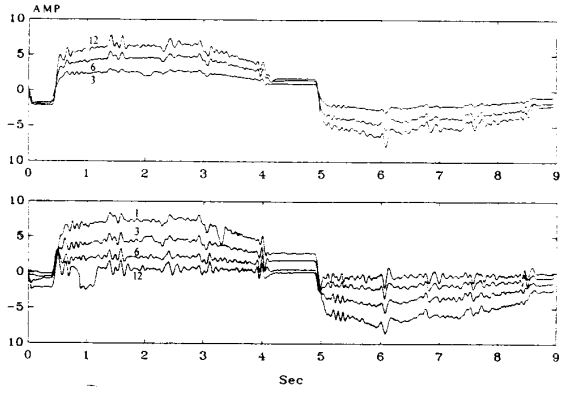

Fig. 10 – Memory control v_k (upper trace) and feedback control u' (lower trace), in trials 1, 3, 6 and 12, for joint 3

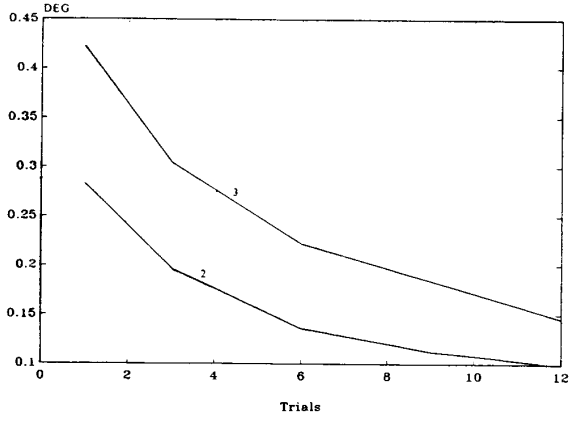

Fig. 8 – RMS values of position errors vs. trial number

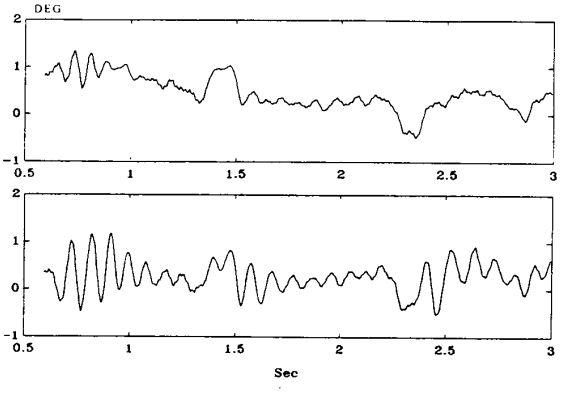

Fig. 11 – Comparison of position errors at iteration 12 for joint 3, with filter $\beta(j\omega)$ (upper trace) and without it (lower trace)