ITERATIVE LEARNING CONTROLLER DESIGN FOR MULTIVARIABLE SYSTEMS

Manuel Olivares [*,1] **Pedro Albertos** [**] **Antonio Sala** [**]

[*] *Dept. of Electronics, Univ. Técnica F. Santa María*
P.O. Box 110-V, Valparaíso, Chile
mos@elo.utfsm.cl
[**] *Dept. of Syst. Eng. and Control, Univ. Politécnica de Valencia*
P.O. Box 22012, E-46071 Valencia, Spain
pedro@aii.upv.es *asala@isa.upv.es*

Abstract: In this paper, a novel expression for the convergence of an iterative learning control algorithm for sampled linear multivariable systems is stated. The convergence analysis shows that, applying this algorithm, the input sequence converges to the system output inverse sequence, specified as a finite-time output trajectory, with zero tracking error on all the sampled points. Also, it gives insight on the learning gain matrix selection to act on the convergence speed or the decoupling of inputs, allowing for an easy tuning using methods from modern control theory. The results are illustrated by some examples, showing a number of options to be investigated. *Copyright ©2002 IFAC*

Keywords: Iterative learning control, multivariable systems, feedforward control, decoupling.

1. INTRODUCTION

Iterative learning is a well known methodology allowing to incorporate the acquainted experience in the use of preliminary designs to get the final one. Since the seminal work of Arimoto, (Arimoto *et al.*, 1984), this has been successfully applied in the field of control, mainly for repetitive working conditions, like it is the case in many manufacturing applications, with a finite time control horizon.

The iterative feature is very attractive from the user viewpoint but a number of formal problems arise, the main one being the existence of a final design, in some sense defined as *optimal*, to which the iterative solution converges. Some sufficient conditions have been proved, (Moore, 1993; Chen and Wen, 1999), for special cases. Another important problem, if this approach is intended to be applied in a more general range of working conditions, is the need of generalization, that is, to get a final design able to control the plant under operating conditions differing to those in which the learning process was carried out. This can be partially solved in the case of affine in control systems, where some sort of scaling can be applied.

One of the drawbacks of the approach is that the controller tends to implement the inverse model of the plant, restricting its application to stable and inverse stable open loop plants. Again, the *a priori* knowledge of the plant model is a usual requirement that also limits the range of applications.

The way the iterative learning approach is implemented, usually leads to a *feedforward* controller. In any control application, an additional control loop would be required to guarantee some kind of disturbance rejection (Amann *et al.*, 1996*b*), bounded steady state error and stability margin, as basic controlled system features.

In the literature, like in (Shoureshi, 1991; Amann *et al.*, 1996*a*), there are many proposed algorithms

[1] Partially supported by projects USM 23.01.11 and GV00-100-14

and learning rules to perform the iterative learning. They consider an unknown model of the plant, as in (Lee *et al.*, 2000), and the convergence is usually proved under restrictive cases (Owens, 1992; Chen and Wen, 1999). The purpose of this paper is to analyse a learning algorithm which is suitable for linear multiple-input-multiple-output (MIMO) systems, denoted as multivariable systems. The learning rule introduces some tunable parameters that determine important features such as the convergence speed.

To introduce the idea of getting the inverse model for a MIMO plant, assume the sampled one described by

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{1}$$

where $x \in \mathscr{R}^n$ is the discrete state vector, $u \in \mathscr{R}^v$ is the input vector and $y \in \mathscr{R}^p$ is the output vector. In this paper, as it is usual, D is assumed to be square, i.e., $v = p$. The input-output matrix transfer function is given by

$$\begin{aligned} y(z) &= G(z)u(z) \\ G(z) &= C(Iz-A)^{-1}B + D \end{aligned} \tag{2}$$

and the pulse response $H(i)$ allows for the equivalent representation

$$\begin{aligned} y(t) &= \sum_{i=0}^{t} H(t-i)u(i), \quad t \geq 0 \\ H(0) &= D; \quad H(i) = CA^{i-1}B, \quad i > 0 \end{aligned} \tag{3}$$

The inverse system, that is the system having $y(t)$ as input and $u(t)$ as output, exists if the following equivalent conditions are satisfied

(1) D is full rank
(2) $G(1)$ is full rank
(3) $H(0)$ is full rank

Plants with delay. In general, the above conditions are not satisfied, as usually $D = 0$, i.e., there is no direct input/output coupling. Moreover, a delay of $m \geq 1$ sampling time units is present, yielding

$$H(i) = 0, \quad i < m \tag{4}$$

In that case, the system output expression (3) can be reduced to show the input-output causality explicitly. Also, for convenience, the cumulated past inputs can be grouped in $\eta(t)$ and considered as disturbances to the present output

$$\begin{aligned} y(t+m) &= H(m)u(t) + \eta(t) \\ \eta(t) &= \sum_{i=0}^{t-1} H(t+m-i)u(i) \end{aligned} \tag{5}$$

Let us consider a finite time horizon N. The following block vectors $U, Y \in \mathscr{R}^{p(N-m+1)}$ can be defined:

$$\begin{aligned} U &= \left[u^T(0)\ u^T(1)\ \cdots\ u^T(N-m)\right]^T \\ Y &= \left[y^T(m)\ y^T(m+1)\ \cdots\ y^T(N)\right]^T \end{aligned} \tag{6}$$

and the matrix relationship $Y = HU$ holds, where

$$H = \begin{bmatrix} H(m) & 0 & \cdots & 0 \\ H(m+1) & H(m) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ H(N) & H(N-1) & \cdots & H(m) \end{bmatrix} \tag{7}$$

$$H(i) = \begin{cases} 0 & i < m \\ CA^{i-1}B & m \leq i \leq N \end{cases} \tag{8}$$

Matrix H in (7) is block lower triangular Toeplitz, referenced here by its leading matrix elements as $H = Toeplitz\{H(m), H(m+1), \dots\}$ for short. Thus, the inverse system representation in finite time exists if $rank[H(m)] = p$

$$U = H^{-1}Y = MY \tag{9}$$

where, due to the triangular structure in H, $M = Toeplitz\{M(m), M(m+1), \dots\}$ with $M(m) = H^{-1}(m)$. From (9), the system input can be expressed as the output of the non causal system

$$u(t) = \sum_{i=0}^{t} M(m+i)y(m+t-i), \quad t \geq 0 \tag{10}$$

2. MIMO ITERATIVE LEARNING

According to (9), the ideal control sequence to track a desired output Y_d is given by $U^* = H^{-1}Y_d$. Let us analyse an iterative scheme to get this control sequence from an initial estimate, denoting as U_k the control action vector at iteration k. Each element $u_k(t)$ in the block vector U_k can be learnt iteratively by means of the general setting shown in figure 1, where the next sequence of inputs U_{k+1} is determined from the current one, U_k, plus a corrective term delivered by the learning algorithm based on Y_d and the actual output sequence Y_k. The corrective term should consider

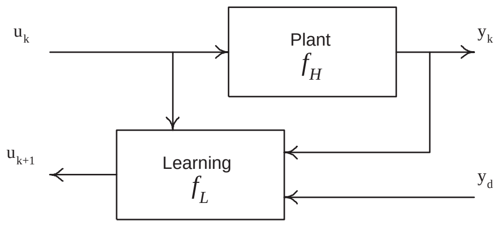


Fig. 1. Iterative learning setting

the plant delay m to meet the causality relationship (10) between the input-output sequence elements. The learning process is repeated for each k (iteration axis), applying a full set of precalculated inputs sequentially, until a specified maximum tracking error is achieved.

The general linear learning algorithm considers the plant and the iterative learning blocks, f_H and f_L, as linear operators, and can be expressed by

$$\begin{aligned} u_{k+1}(t) &= \sum_{j=0}^{n} \alpha_j u_{k-j}(t) + \Gamma e_k(t+m) \\ e_k(t+m) &= y_d(t+m) - y_k(t+m) \end{aligned} \tag{11}$$

where $e_k(t)$ represents the output error vector at time t in the iteration k, α_j are the past control factors and $\Gamma_{p\times p}$ is the learning gain matrix. The non causality of (11) is avoided because of the iterative calculation of the next input sequence. Additional terms involving $e_{k-j}(t)$ could be also considered in (11). For $\alpha_0 = 1$, $\alpha_j = 0\ \forall j > 0$, and accordingly to (5), the whole iterative learning process for $t \in [0, N-m]$ can be represented by a set of $N-m+1$ closed loop subsystems, as shown in figure 2. This setting originates the following theorem.

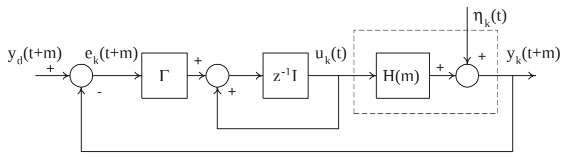


Fig. 2. Plant subsystems with learning

Theorem 1. (Learning Algorithm Convergence). For the multivariable system plus delay (1), (2), (5), and $D = 0$, the first order iterative learning control algorithm designed to track a desired finite time trajectory sequence Y_d

$$u_{k+1}(t) = u_k(t) + \Gamma(y_d(t+m) - y_k(t+m)) \qquad (12)$$

i. Converges if and only if
$$F = I - H(m)\Gamma \quad \text{is Hurwitz} \qquad (13)$$
ii. the maximum steady state tracking error is
$$\lim_{k\to\infty}\left(\max_{m\le i\le N}|y_d(i) - y_k(i)|\right) = 0 \qquad (14)$$
iii. the input sequence U^* converges to
$$U^* = \lim_{k\to\infty} U_k = H^{-1}Y_d \qquad (15)$$
iv. and the maximum convergence speed is achieved for
$$\Gamma = \left(CA^{m-1}B\right)^{-1} = H^{-1}(m) \qquad (16)$$

Proof: The learning law (12) may be rewritten using (10), that yields

$$\begin{aligned}&\sum_{i=0}^{t} M(m+i)y_{k+1}(m+t-i) = \\ &\quad \sum_{i=0}^{t} M(m+i)y_k(m+t-i) + \Gamma e_k(t+m)\end{aligned} \qquad (17)$$

Subtracting from both sides the ideal input, $\sum_{i=0}^{t} M(m+i)y_d(m+t-i)$, the following tracking error equation is obtained

$$\begin{aligned}&\sum_{i=0}^{t} M(m+i)e_{k+1}(m+t-i) = \\ &\quad \sum_{i=0}^{t} M(m+i)e_k(m+t-i) - \Gamma e_k(t+m)\end{aligned} \qquad (18)$$

Now, the $\mathscr{Z}$-transform in the iteration axis k is applied to (18), obtaining the following relationship among the temporal tracking errors

$$\begin{aligned}&[(z-1)I\cdot M(m) + \Gamma]e_z(t+m) = \\ &-(z-1)I\sum_{i=1}^{t} M(m+i)e_z(t+m-i)\end{aligned} \qquad (19)$$

Initial condition terms have been removed as they do not affect stability. The characteristic equation $\det[(z-1)I\cdot M(m) + \Gamma] = 0$ must have its roots (poles of the iterative learning algorithm) inside the unit circle for stable learning. That can be easily shown to be equivalent to the condition

$$|eig[M^{-1}(m)(M(m) - \Gamma)]| < 1 \qquad (20)$$

which proves i. Once Γ is properly chosen, ii is proved applying the final value theorem to (19), that gives zero steady state tracking error over all the sampled points. Part iii is consequence of ii, as the inverse problem solution is unique. Part iv just shows that for the special gain matrix selection $\Gamma = H^{-1}(m)$, the poles of the characteristic equation are at the origin. $\square$

Note that strong control actions could be applied if the first non null $H(i)$ matrix has a rather small minimum singular value $\underline{\sigma}(H(m))$. In that case, it could be better to consider a new delay, $\bar{m} > m$, showing a stronger dependency of the output with respect to the input applied $\bar{m}$ time instants of time before. Both, $H(m)$ and m, can be estimated through the open-loop system pulse response, if available.

Going for the maximum convergence speed with (minimum time algorithm, high learning rates) can be a bad solution in a noisy environment or if there is a model mismatch.

After the learning process, a *feedforward* control is applied. This implies the requirement of stable and inverse stable open-loop plants, or the use of feedback stabilizing controllers or filters. In general, non minimum phase systems yield unfeasible U^* input sequences for big values of task length N.

Remark 1. To compute a learning gain matrix for MIMO systems, an estimate of $H(m)$ must be available. That is considerably more difficult than the choice of a learning gain in the SISO case, where a small learning rate with the right sign would be enough for convergence; in the MIMO case, directionality effects arise. The task is considerably simpler if good input-output pairings are chosen, so that $H(m)$ is diagonal dominant.

Remark 2. High order learning laws, taking into account more previous terms, will lead to similar results. This can be also a good option to filter the effect of measurement noise.

3. EXTENSIONS

In the previous section a number of assumptions have been taken. Some of them are easily removed because it is just a matter of ease of notation, like it is the case on the order of the learning law. Some other assumptions can be also removed providing additional insight in the proposed analysis.

Plant delay. The selection of the plant delay, m, in the learning algorithm, is crucial. If the delay is underestimated, the computed control action would be influenced by the presence of noise and $\underline{\sigma}(H(m))$ will be very small or null. Thus, a very strong action will result. This can also happen if, at time $t = m$ some outputs are excited, but others have longer delays or negligible response at that time.

If the delay is overestimated, that is, if there is some influence on the output for $t < m$, the previous control actions will act as disturbances in the learning process. As a result, the learnt control sequence, U, if it converges, is no more the one theoretically obtained by the inverse model.

Learning gain matrix. As pointed out in the theorem, the selection of Γ determines the convergence and the velocity of the learning process. The computation of this gain matrix can be solved like a pole placement problem.

Assuming a system matrix I, and an input matrix $H(m)$, if the pair $(I, H(m))$ is controllable, the poles of the matrix F in (13) can be assigned to the required values. Given the special value of the system matrix, I, the necessary condition is $H(m)$ to be full rank. Thus, the learning gain matrix can be designed from a given tracking error dynamics matrix, F, which should be Hurwitz

$$\Gamma = H^{-1}(m)(I-F) \tag{21}$$

Generalization. The proposed approach has been proved for linear systems. Thus, the linear systems properties can be used to apply the learnt control sequence U^* to any combination of previously learnt desired output trajectories Y_d. Notice that the learnt control sequence is only valid for the Y_d used in the training phase.

In principle, for multivariable systems, changes in the references can be considered separately. In this way, once the control inputs are determined for each one, assuming constant the remaining references, any tracking scenario can be solved by combining the corresponding control actions. Also, it is allowed to scale the control action if a desired Y_d sequence is proportional to the one for which the system has been trained.

4. EXAMPLES

In order to illustrate the application of theorem 1, three simple examples are shown. In the first case a two-input two-output open loop stable plant, G_1, is controlled. Then, an unstable plant, G_2, is considered and a stabilizer controller is previously computed. Finally, some discussion about the selection of the gain matrix Γ, will point out the convergence issue in the learning algorithm.

Example 1. (Stable plant). Given the external representation of the plant

$$G_1(s) = \begin{bmatrix} \dfrac{2}{4s^2+1} & \dfrac{0.5}{3s+1} \\ \dfrac{-(s+1)}{s^2+3s+1} & \dfrac{2}{2s^2+2s+3} \end{bmatrix}$$

the target is to follow two independent finite time references, in the interval $[m, N]$, with $N = 20$. A step change, y_{d1}, for the first output and a triangular reference, y_{d2}, for the second one.

First, a sampled data model of the plant, with sampling period $T = 1sec$, is derived. Then, the learning rule (12) is applied. As there is no delay, $m = 1$ is chosen. Initially, the fastest learning gain is selected, being

$$\Gamma_1 = \left(CA^{m-1}B\right)^{-1} = \begin{bmatrix} 2.1961 & -0.9514 \\ 3.2619 & 1.6434 \end{bmatrix}$$

From iteration 19 onwards, the maximum tracking error is lower than 10^{-9} on both channels. The final control and output signals are shown in the figure 3. ■

Example 2. (Unstable plant). Again, the input-output representation of the plant is initially considered, being

$$G_2(s) = \begin{bmatrix} \dfrac{1}{2s^2+2s+3} & \dfrac{-2}{2s+1} \\ \dfrac{4}{(2s+1)(4s-1)} & \dfrac{1}{4s^2+3s+1} \end{bmatrix}$$

and similar references as before are required to be tracked. As the subsystem $G_{21}(s)$ is unstable, it is necessary to stabilize the system. The global controller will be composed of a feedback part, $u_{fb}(t)$, (stabilizing) and a feedforward part, $u_{ff}(t)$, (tracking). In this case, a state feedback is initially designed. An optimal LQR, with $Q = I_{7\times7}$ and $R = 20I_{2\times2}$ is assumed, taking the same sampling period as before, $T = 1sec$.

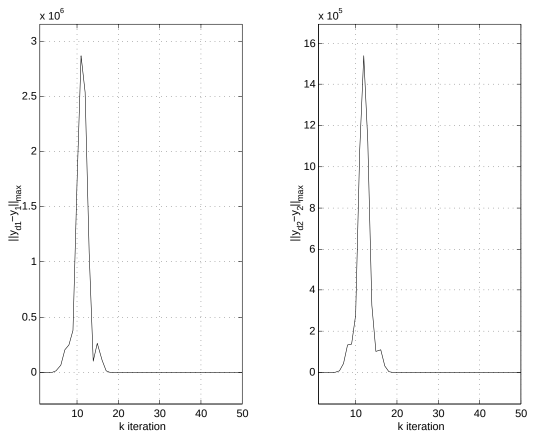
$$u(t) = u_{fb}(t) + u_{ff}(t); \quad u_{fb} = -Kx$$

$$K = \begin{bmatrix} -0.19 & 0.63 & -1.1 & 0.99 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.4 & -0.55 & 0.61 \end{bmatrix}$$

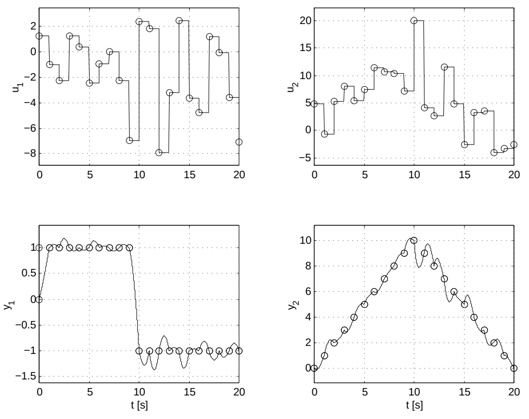
The iterative learning law (12) is applied to get u_{ff}, with the fastest learning gain, being

$$\Gamma_2 = \left(CA^{m-1}B\right)^{-1} = \begin{bmatrix} 0.4868 & 3.9539 \\ -1.1695 & 0.8219 \end{bmatrix}$$

Again, from $k = 19$, the maximum tracking error is lower than 10^{-5} on both channels. The results are



(a) Maximum tracking error vs. k



(b) Tracking result, at $k = 50$

Fig. 3. Control of plant $G_1(s)$

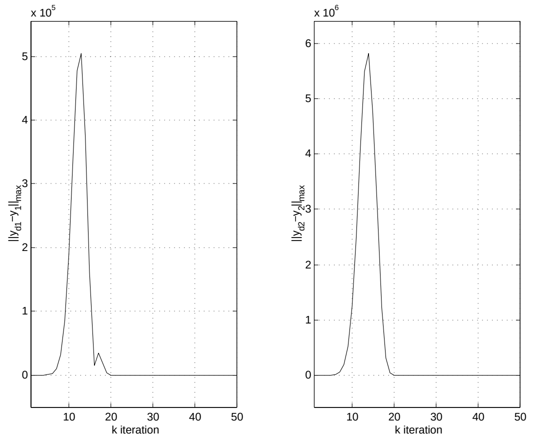
shown in the figure 4, where the learnt control inputs u_{1ff} and u_{2ff} are shown. ■

Example 3. (Convergence rate). Now, for the first plant, G_1, different learning gain matrix Γ are applied. The learning gain is selected using (21), with $F = diag\{f_1, f_2\}$, whose elements, determine the rate of convergence. This allows for input learning decoupling.

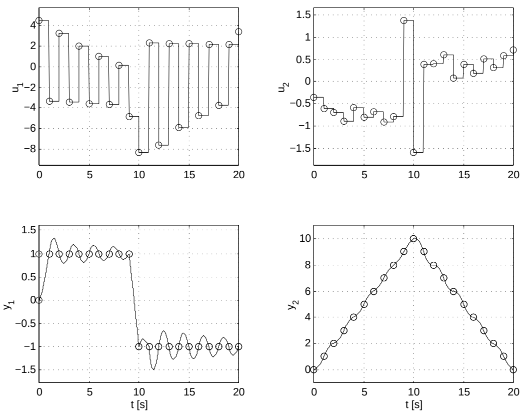
Firstly $f_1 = 0.1$ and $f_2 = 0.3$ are chosen. The results, at $k = 50$ iterations, are shown in fig. 5(a). In this case, a good convergence is obtained for the control sequences $u^*(t)$, although a big transient tracking error appears.

Now if $f_1 = e^{-0.1}$ and $f_2 = e^{-10}$ are chosen, the learning process is slower, since 300 iterations are required to get convergence, but the maximum transient tracking error is lower. Results are shown in figure 5(b). ■

Note that, as stated in theorem 1, the convergence and its speed is determined by the learning law (by



(a) Maximum tracking error vs. k



(b) Tracking result, at $k = 50$

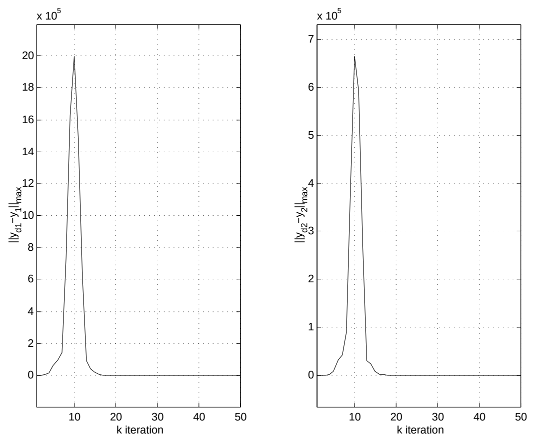
Fig. 4. Tracking for the unstable plant $G_2(s)$

Γ among other factors), but the final "optimal" control sequences are the same. The sampling period choice and the Y_d selection affect the intersampling behavior, as hidden oscillations shown in the previous figures. To avoid hidden oscillations the reference model output must satisfy some algebraic constraints, but they can only be evaluated if the plant transfer function matrix is known (and that is not the case in ILC). Also, as the minimum time learning gain is referred to the adopted sampling period, its choice could present a high sensitivity to disturbances, reflected as big transient tracking errors.

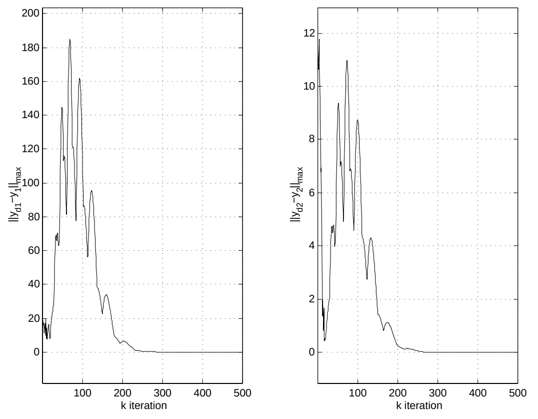
5. CONCLUSIONS

A convergence analysis method has been proposed for an iterative learning control algorithm, suitable for MIMO systems. The resulting design approach has been tested with a number of examples and the parameter selection has been discussed.

The main result of this paper is a convergence theorem giving the feasibility of applying pole assign-



(a) Maximum tracking error vs. k with F_1



(b) Maximum tracking error vs. k with F_2

Fig. 5. Results with diagonal matrix F

ment techniques in the iteration axis, rather than the time axis. Other techniques could be also applied. When convergence is assured, zero tracking error on all sampling points is achieved in steady state. Other approaches deal with the convergence problem, getting learning convergence as a sufficient condition.

Many questions remain open. The generalization issue is not addressed in this paper, as well as the possibility of implementing fuzzy logic controllers using the same technique. In (Albertos and Olivares, 2000) an attempt to design iterative learning fuzzy controllers has been presented. The main idea being to learn the controller structure and parameters instead of a control sequence. As a result, some options for generalizing the learnt controller to different operating scenarios could be foreseen. A full analysis of these properties, for MIMO and SISO systems using fuzzy logic controllers is the subject of (Olivares, 2001). This is currently a topic of active research.

6. ACKNOWLEDGEMENTS

Manuel Olivares would like to acknowledge gratefully the support of the Agencia Española de Cooperación Internacional (AECI) during his stay at the Systems Engineering and Control Department of the Universidad Politécnica de Valencia.

7. REFERENCES

Albertos, P. and M. Olivares (2000). On line learning of a gantry crane. In: *Proc. Of the 15th Int. Symposium on Intelligent Control ISIC'00*. pp. 157–162.

Amann, N., D.H. Owens and E. Rogers (1996*a*). An H_∞ approach to linear iterative learning control design. *Int. J. of Adaptive Control and Signal Processing* **10**, 767–781.

Amann, N., D.H. Owens and E. Rogers (1996*b*). Iterative learning control using optimal feedback and feedforward actions. *Int. Journal of Control* **65**(2), 277–293.

Arimoto, S., S. Kawamura and F. Miyazaki (1984). Bettering operations of robots by learning. *J. of Robotic Systems* **1**, 123–140.

Chen, Y. and C. Wen (1999). *Iterative Learning Control: Convergence, Robustness and Applications*. Vol. 248 of *LNCIS*. Springer-Verlag.

Lee, T.H., H. Dou, K.K. Tan and Y. Chen (2000). Experimental studies on high precision tracking control of linear motor using noncausal filtering based iterative learning control. In: *Asian Control Conference*. pp. 1–6, WC 1–4.

Moore, K.L. (1993). *Iterative Learning Control for Deterministic Systems*. Springer-Verlag.

Olivares, M. (2001). Fuzzy Control with Iterative Learning. PhD thesis. Dept. of Systems Engineering and Control, Universidad Politécnica de Valencia. Spain. In spanish.

Owens, D.H. (1992). Iterative learning control – convergence using high gain feedback. In: *Proc. Of the 31st IEEE Conference on Decision and Control*. pp. 2545–2549.

Shoureshi, R. (1991). Learning and decision making for intelligent control systems. *IEEE Control Systems* **11**(1), 34–37.