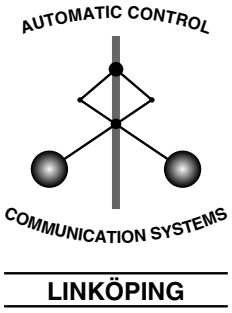
Experimental comparison of some classical Iterative Learning Control algorithms

Mikael Norrlöf, Svante Gunnarsson

Division of Automatic Control
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
WWW: http://www.control.isy.liu.se
E-mail: mino@isy.liu.se, svante@isy.liu.se

17th June 2002



Report no.: LiTH-ISY-R-2433
Submitted to IEEE Transactions on Robotics and Automation, 2002

Technical reports from the Control & Communication group in Linköping are available at http://www.control.isy.liu.se/publications.

Abstract

This paper gives an overview of classical Iterative Learning Control algorithms. The presented algorithms are also evaluated on a commercial industrial robot from ABB. The presentation covers implicit to explicit model based algorithms. The result from the evaluation of the algorithms is that performance can be achieved by having more system knowledge.

Keywords: Iterative learning control, design, experiment, industrial robot

Experimental comparison of some classical Iterative Learning Control algorithms

Mikael Norrlöf, *Member, IEEE*, and Svante Gunnarsson

Abstract— **This paper gives an overview of classical Iterative Learning Control algorithms. The presented algorithms are also evaluated on a commercial industrial robot from ABB. The presentation covers implicit to explicit model based algorithms. The result from the evaluation of the algorithms is that performance can be achieved by having more system knowledge.**

Keywords— **Iterative learning control, design, experiment, industrial robot**

I. Introduction

The purpose of this paper is to give an overview of some classical *Iterative Learning Control (ILC)* algorithms. This includes describing how to design and implement the ILC algorithms and to compare the resulting designs in experiments. The system that will be used throughout the paper is an industrial robot (IRB 1400) from ABB. The control system is based on the commercially available S4C, modified to make it possible to implement and evaluate ILC on the joint level. A thorough description of the experimental setup is found in e.g., [1]. The paper is organized as follows. In Section II ILC is introduced and the history of ILC is briefly reviewed. Section III gives the theoretical background to ILC, this background is necessary for the presentation of the design algorithms. Section IV describes the design steps in the different ILC algorithms and in Section V the proposed algorithms are evaluated on the ABB industrial robot. Finally some conclusions are given in Section VI.

II. A brief history on ILC

The idea of using an iterative method to compensate for a repetitive error is not new. When letting a machine do the same task repeatedly it is, at least from an engineering point of view, very sound to use knowledge from previous iterations of the same task to try to reduce the error next time the task is performed. The first academic contribution to what today is called ILC appears to be a paper by Uchiyama [2]. What is a bit remarkable is however that an application for a US patent on "Learning control of actuators in control systems" [3] was done already in 1967 and it was accepted as a patent in 1971. The idea in the patent was to store a "command signal" in a computer memory and iteratively update the command signal using the error between the actual response and the desired response of the actuator. This is clearly an implementation of ILC (see also [4]). From an academic perspective it was not until 1984 that ILC started to become an active research area. In 1984 [5], [6], and [7], were independently published describing a method that iteratively could compensate for model errors and disturbances. The development of ILC stems originally from the robotics area where repetitive motions show up naturally in many applications. Examples of contributions where ILC is applied in robotics are [5], [8], [9], [10], and [11]. Examples of surveys on ILC can be found in, [12], [13], [14], [15] and [16]. [14] contains a very good overview of the ILC research and a categorization of many of the publications on ILC up to 1998.

Mikael Norrlöf (corresponding author) and Svante Gunnarsson are with the Department of Electrical Engineering, Linköpings universitet, SE-581 83 Linköping, Sweden. Phone: +46 13 28{2704,1747} Fax: +46 13 282622 Email: {mino, svante}@isy.liu.se

III. Problem Description

A. System description

This paper deals with ILC applied to SISO systems working in discrete time. The general system description is

$$\boldsymbol{y}_k = \boldsymbol{T}_r \boldsymbol{r} + \boldsymbol{T}_u \boldsymbol{u}_k \tag{1}$$

with

$$\boldsymbol{y}_k = \big(y_k(0), \dots, y_k(n-1)\big)^T \tag{2}$$

and $\boldsymbol{r}$, $\boldsymbol{u}_k$ defined accordingly. As a special case the linear time invariant case follows

$$y_k(t) = T_r(q)r(t) + T_u(q)u_k(t). \tag{3}$$

The system in (1) gives a more general description since it captures also time variant systems.

A linear time invariant and causal system, $T_r(q)$, is in matrix form described by a Toeplitz matrix

$$\boldsymbol{T}_r = \begin{pmatrix} g_{T_r}(0) & 0 & \dots & 0 \\ g_{T_r}(1) & g_{T_r}(0) & & \vdots \\ \vdots & & \ddots & 0 \\ g_{T_r}(n-1) & g_{T_r}(n-2) & \dots & g_{T_r}(0) \end{pmatrix} \tag{4}$$

where $g_{T_r}(t)$, $t \in [0, n-1]$ are the impulse response coefficients of T_r, the sampling time is assumed to be 1, and n is the number of samples. If the system is linear time variant, the matrix $\boldsymbol{T}_r$ does not become a lower triangular Toeplitz matrix but instead a general lower triangular matrix. The matrix $\boldsymbol{T}_u$ is given in the same way. The symbols describing vectors and matrices in the matrix description are given in bold face to make it easier to distinguish between the representation in (3) and the matrix description in (1). The system description can be made even more general by including system and measurement disturbances. This is covered in [1].

For the frequency domain analysis the system is assumed to be linear time invariant as in (3) with $T_r(q)$ and $T_u(q)$ stable. The corresponding frequency domain representation is found using the Fourier transform. Given $T_r(q)$, the frequency domain representation can also be found by simply replacing q with $e^{i\omega}$ in $T_r(q)$. The frequency domain representation of $T_u(q)$ is found in the same way.

The signals $y_k(t)$, $r(t)$, and $u_k(t)$ are transformed to the frequency domain by $X(\omega) = \sum_{l=0}^{\infty} x(l)e^{-i\omega l}$ and it is assumed that this sum exists and is finite for all ω. This gives the resulting frequency domain representation,

$$Y_k(\omega) = T_r(e^{i\omega})R(\omega) + T_u(e^{i\omega})U_k(\omega). \tag{5}$$

For iterative systems this is an approximation since in the computation of the Fourier transform it is assumed that the time horizon is infinite. In the next section (as a result of Theorem 2) it is shown that this is in fact no restriction from the stability point of view.

B. Classical ILC

The interpretation of *classical ILC* might differ among researchers but the one adopted here is that classical ILC is first order ILC with iteration independent operators. This means that the updating equation for the ILC can be written

$$\boldsymbol{u}_{k+1} = \boldsymbol{Q}(\boldsymbol{u}_k + \boldsymbol{L}\boldsymbol{e}_k) \tag{6}$$

where the matrix form description is used. The error $\boldsymbol{e}_k$ is defined as $\boldsymbol{e}_k = r - \boldsymbol{y}_k$. In this framework $\boldsymbol{Q}$ and $\boldsymbol{L}$ are matrices in $\mathbb{R}^{n\times n}$, $\boldsymbol{u}_k$ and $\boldsymbol{e}_k$ are vectors in $\mathbb{R}^n$, compare also with equations (1), (2), and (4). Sometimes it is also useful to have a filter description,

$$u_{k+1}(t) = Q(q)(u_k(t) + L(q)e_k(t)) \tag{7}$$

compare also the system representation in (3).

Among the works that have been addressing design of ILC algorithms, an important part is based on linear quadratic optimal ILC. Within this framework Prof. Owens' group has made contributions in e.g., [17] and [18], while other contributions are for example [19], [20], [21], and [22]. An early contribution in this direction is also [23]. ILC synthesis based on H_∞ methods has been covered in e.g., [24], [25], and [26] but this work is not covered here.

C. Some results on stability

In this section some stability results for ILC systems are reviewed. Two different measures of the size of a matrix will be used. The first is the spectral radius which is defined as

$$\rho(\boldsymbol{F}) = \max_{i=1,\dots,n} |\lambda_i(\boldsymbol{F})| \tag{8}$$

where $\lambda_i(\boldsymbol{F})$ is the ith eigenvalue of the matrix $\boldsymbol{F} \in \mathbb{R}^{n\times n}$. The second is the maximum singular value

$$\bar{\sigma}(\boldsymbol{F}) = \sqrt{\rho(\boldsymbol{F}^T\boldsymbol{F})}. \tag{9}$$

The maximum singular value gives a bound on the gain of a matrix by the fact that $\|\boldsymbol{F}\boldsymbol{x}\| \leq \bar{\sigma}(\boldsymbol{F})\|\boldsymbol{x}\|$. From (1) and the ILC updating equation in (6) it follows that

$$\boldsymbol{u}_{k+1} = \boldsymbol{Q}(I - \boldsymbol{L}\boldsymbol{T}_u)\boldsymbol{u}_k + \boldsymbol{Q}\boldsymbol{L}\boldsymbol{r}. \tag{10}$$

The next theorem comes as a natural result of (10).

Theorem 1 (Stability condition) *The system in* (1) *controlled using the ILC updating equation* $\boldsymbol{u}_{k+1} = \boldsymbol{Q}(\boldsymbol{u}_k + \boldsymbol{L}\boldsymbol{e}_k)$ *is stable iff* $\rho(\boldsymbol{Q}(I - \boldsymbol{L}\boldsymbol{T}_u)) < 1$.

The formal proof is given in [1] and it is based on standard results from linear systems theory. The following result is important for many of the design algorithms.

Theorem 2 (Monotone exponential convergence) *If the system in* (1) *is controlled using the ILC updating equation* $\boldsymbol{u}_{k+1} = \boldsymbol{Q}(\boldsymbol{u}_k + \boldsymbol{L}\boldsymbol{e}_k)$ *and* $\bar{\sigma}(\boldsymbol{Q}(I - \boldsymbol{L}\boldsymbol{T}_u)) < 1$ *then the ILC system is stable and*

$$\|\boldsymbol{u}_\infty - \boldsymbol{u}_k\| \leq \lambda^k \|\boldsymbol{u}_\infty - \boldsymbol{u}_0\|$$

with $\boldsymbol{u}_\infty = \lim_{k\to\infty} \boldsymbol{u}_k$.

The proof is found in [1]. The condition on the maximum singular value can be replaced by a frequency domain condition according to

$$|1 - L(e^{i\omega})T_u(e^{i\omega})| < |Q^{-1}(e^{i\omega})|, \quad \forall\omega \tag{11}$$

if the system is linear time invariant. This coincides with the very common frequency domain condition given in the ILC literature. In Theorem 2 it is shown that when this condition is fulfilled u_k will converge to the limit value, u_∞, exponentially and without overshoot. Note that the results are all formulated in the disturbance free case. For a generalization to the case with disturbances see e.g., [1].

IV. Some Methods for Classical ILC Synthesis

To design a stable and efficient ILC algorithm it is necessary to have a model of the plant to be controlled. The level of detail of the model will differ between the different design algorithms but it is always true that some knowledge of the controlled system is needed in order to carry out the ILC design. The knowledge might be replaced by experiments where the ILC algorithm is adjusted according to the result from the experiments.

A. A heuristic approach

The first design algorithm uses a system model to check if the stability criterion is fulfilled. The knowledge about the system could also be reduced to only the time delay of the system and, to be sure of the stability, the size of the first Markov parameter of the controlled system. This might however give very poor performance. The Q-filter in (7) is applied to robustify the ILC algorithm, cf. (11) where it is clear that by choosing $|Q(e^{i\omega})|$ small the stability region of the algorithm can be enlarged. The price paid for this action is that the algorithm will not converge to zero error, even in the noise free case. The algorithm is here called "heuristic" but sometimes it has also been referred to as P-type in the literature.

Algorithm 1 (A heuristic design procedure)
1. Choose the Q filter as a low-pass filter with cut-off frequency such that the band-width of the learning algorithm is sufficient.
2. Let $L(q) = \kappa q^\delta$. *Choose* κ *and* δ *such that the stability criterion, formulated in the frequency domain* $|1 - L(e^{i\omega})T_u(e^{i\omega})| < |Q^{-1}(e^{i\omega})|$, *is fulfilled. Normally it is sufficient to choose* δ *as the time delay and* $0 < \kappa \leq 1$ *to get a stable ILC system.*

If in step 2, the necessary and sufficient condition for stability is used (from Theorem 1) then the resulting ILC algorithm is still stable but the transient response can be bad. In the next example it is shown that the performance is not so good if the time delay of the system is uncertain.

Example 1 (A numerical example) *Assume that the system* $T_u(q)$ *is given by*

$$T_u(q) = \frac{0.09516q^{-1}}{1 - 0.9048q^{-1}} \tag{12}$$

and the filter $L(q)$ *is chosen according to* $L(q) = q$ *and* $L(q) = 1$. *The first choice corresponds to the true system delay while the second choice comes as a result when the system model is incorrect and it is assumed that there is no delay in the system. With* $Q(q) = 1$ *the bandwidth of the ILC algorithm is not limited and in order to fulfill the stability condition for the given system,* κ *has to be chosen such that* $|1 - \kappa 0.09516| < 1$ *in the first case. This means that* κ *must lie in the interval* $0 < \kappa < 21$. *In the second case the necessary and sufficient stability condition in Theorem 1 is not fulfilled since* $\rho(I - \boldsymbol{T}_u) = 1$. *This implies that convergence will not be achieved for any* κ.

If the system delay is over-estimated instead of under-estimated as in the example above, then it is not true that with $Q = 1$ the system will be guaranteed to be stable. Introducing a Q as a low pass filter can however make the system stable.

B. A model-based approach

The design procedure presented in this section has also been discussed in [27], [28] and [1]. The idea is similar to the approach

in [26] but there a model matching approach based on H_∞ methods is used while here an algebraic approach is adopted.

Algorithm 2 (A model-based design procedure)
1. Build a model of the relation between the ILC input and the resulting correction on the output, i.e., find a model $\widehat{T}_u$ of T_u.
2. Choose a filter $H_B(q)$ such that it represents the desired convergence rate for each frequency. Normally this means a high-pass filter.
3. Compute L as $L(q) = \widehat{T}_u^{-1}(q)(1 - H_B(q))$.
4. Choose the filter $Q(e^{i\omega})$ as a low-pass filter with cut-off frequency such that the band-width of the resulting ILC algorithm is high enough and the desired robustness is achieved.

To explain the use of the filter $H_B(q)$ in step 2, consider the updating equation for the error, $e_{k+1}(t) = (1 - T_u(q)L(q))e_k(t)$. Clearly the choice of $H_B(q)$ will decide the nominal convergence rate of the error. In the frequency domain the filter H_B can be adjusted to give, e.g., a slow but more robust convergence for some frequencies. The choice of H_B must be realizable. It is clearly not possible to choose H_B small for frequencies where the model is very uncertain since this will most likely lead to a divergent behavior of the resulting ILC algorithm. The choice of H_B has, therefore, also to include robustness considerations, although robustness is achieved with the Q-filter.

The resulting L-filter might have an unnecessary high degree, therefore it can be possible to make a model reduction of L using model reduction techniques, e.g., balanced truncation or L_2 model reduction.

C. Design based on optimization

Previous contributions to the optimization based approach to ILC can be found in e.g., [29], [30], [19], and [20]. Some approaches based on ideas from unconstrained optimization and minimization techniques are presented in [23]. For a general discussion on unconstrained minimization see, e.g., the book [31].

C.1. Algorithm derivation. Assume that the system is in matrix form as in (1). Let the quadratic criterion be

$$J_{k+1} = \boldsymbol{e}_{k+1}^T \boldsymbol{W}_e \boldsymbol{e}_{k+1} + \boldsymbol{u}_{k+1} \boldsymbol{W}_u \boldsymbol{u}_{k+1}$$

where $\boldsymbol{e}_{k+1} = \boldsymbol{r} - \boldsymbol{y}_{k+1}$. The idea is to determine $\boldsymbol{u}_{k+1}$ in such a way that the error $\boldsymbol{e}_{k+1}$ becomes as small as possible with respect to the criterion. The weighting matrices decide the trade off between performance and input energy and the matrices can be used for both frequency as well as time weighting. The criterion is minimized subject to the constraint

$$(\boldsymbol{u}_{k+1} - \boldsymbol{u}_k)^T (\boldsymbol{u}_{k+1} - \boldsymbol{u}_k) \leq \delta.$$

Introducing the Lagrange multiplier yields the criterion

$$\begin{aligned} \bar{J}_{k+1} = \boldsymbol{e}_{k+1}^T \boldsymbol{W}_e \boldsymbol{e}_{k+1} + \boldsymbol{u}_{k+1}^T \boldsymbol{W}_u \boldsymbol{u}_{k+1} \\ + \lambda((\boldsymbol{u}_{k+1} - \boldsymbol{u}_k)^T (\boldsymbol{u}_{k+1} - \boldsymbol{u}_k) - \delta). \end{aligned} \tag{13}$$

From (1) it follows that $\boldsymbol{e}_{k+1}$ is given by

$$\boldsymbol{e}_{k+1} = (I - \boldsymbol{T}_r)\boldsymbol{r} - \boldsymbol{T}_u \boldsymbol{u}_{k+1}. \tag{14}$$

Using this result together with (13) makes it possible to do a straightforward differentiation of $\bar{J}_{k+1}$ with respect to $\boldsymbol{u}_{k+1}$. This gives

$$-\boldsymbol{T}_u^T \boldsymbol{W}_e \boldsymbol{e}_{k+1} + \boldsymbol{W}_u \boldsymbol{u}_{k+1} + \lambda(\boldsymbol{u}_{k+1} - \boldsymbol{u}_k) = 0 \tag{15}$$

where the optimum is achieved when the derivative equals zero. Using (14) in (15) gives

$$\begin{aligned} \boldsymbol{u}_{k+1} = (\boldsymbol{W}_u + \lambda \cdot I + \boldsymbol{T}_u^T \boldsymbol{W}_e \boldsymbol{T}_u)^{-1} \\ (\lambda \boldsymbol{u}_k + \boldsymbol{T}_u^T \boldsymbol{W}_e (I - \boldsymbol{T}_r)\boldsymbol{r}). \end{aligned} \tag{16}$$

From (16), (10), and Theorem 1 it is possible to establish the convergence criterion for the proposed method. If the system model corresponds to the true system and $\lambda > 0$, then the proposed ILC algorithm is always stable. This follows from the fact that $\boldsymbol{W}_u + \boldsymbol{T}_u^T \boldsymbol{W}_e \boldsymbol{T}_u$ is symmetric and positive definite and $\lambda > 0$. See [20] for more details. From (14) it follows that

$$(I - \boldsymbol{T}_r)\boldsymbol{r} = \boldsymbol{e}_k + \boldsymbol{T}_u \boldsymbol{u}_k.$$

The result in (16) can therefore be reformulated into

$$\begin{aligned} \boldsymbol{u}_{k+1} = (\boldsymbol{W}_u + \lambda \cdot I + \boldsymbol{T}_u^T \boldsymbol{W}_e \boldsymbol{T}_u)^{-1} \\ ((\lambda \cdot I + \boldsymbol{T}_u^T \boldsymbol{W}_e \boldsymbol{T}_u)\boldsymbol{u}_k + \boldsymbol{T}_u^T \boldsymbol{W}_e \boldsymbol{e}_k). \end{aligned} \tag{17}$$

Interpreted as in (6) and emphasizing that the system model, $\widehat{\boldsymbol{T}}_u$, has to be used in the final algorithm gives

$$\boldsymbol{Q} = (\boldsymbol{W}_u + \lambda \cdot I + \widehat{\boldsymbol{T}}_u^T \boldsymbol{W}_e \widehat{\boldsymbol{T}}_u)^{-1} (\lambda \cdot I + \widehat{\boldsymbol{T}}_u^T \boldsymbol{W}_e \widehat{\boldsymbol{T}}_u) \tag{18}$$

and

$$\boldsymbol{L} = (\lambda \cdot I + \widehat{\boldsymbol{T}}_u^T \boldsymbol{W}_e \widehat{\boldsymbol{T}}_u)^{-1} \widehat{\boldsymbol{T}}_u^T \boldsymbol{W}_e. \tag{19}$$

The updating matrices $\boldsymbol{Q}$ and $\boldsymbol{L}$ hence depend on the nominal model $\widehat{\boldsymbol{T}}_u$ and the weighting matrices $\boldsymbol{W}_u$ and $\boldsymbol{W}_e$. The Lagrange multiplier λ is not computed explicitly but instead used as a design variable.

Algorithm 3 (Optimization based ILC design)
1. Build a model of the relation between the ILC input and the resulting correction on the output, i.e., find a model $\widehat{\boldsymbol{T}}_u$ of $\boldsymbol{T}_u$.
2. Choose the weights $\boldsymbol{W}_e$ and $\boldsymbol{W}_u$ and the Lagrange multiplier λ in the criterion.
3. Calculate the matrices $\boldsymbol{Q}$ and $\boldsymbol{L}$ according to (18) *and* (19).
4. Use the ILC updating equation according to (6) *with $\boldsymbol{u}_0$ for example chosen as $\boldsymbol{u}_0 = 0$.*

In the next sections the different design parameters and how different choices effect the resulting ILC system will be discussed more.

V. Experiments

Next the three presented algorithms are implemented and evaluated on the industrial robot. The models that are used for the design are found by applying system identification [32] to the plant. This step is described in more detail in [1]. The experiment is an example of a multiple joint motion where ILC is applied to three of the joints of the robot.

A. Description of the test case

In this test case ILC is applied to 3 of the 6 joints of the IRB 1400, see Fig. 1. Each of the 3 joints is modeled as a transfer function description from the ILC input to the measured motor position of the robot. The conventional feedback controller, implemented by ABB in the S4C control system, works in parallel with the ILC algorithms. Since the controller is working well, the closed loop from reference angular position to measured angular position can be described using a low order linear discrete time model.

In Fig. 2 the program used in the experiment is shown together with the desired trajectory on the arm-side of the robot. The instruction `moveL p2,v100,z1` refers to an instruction that produces a straight line on the arm-side of the robot. The line starts from the current position, not explicitly stated, and ends in `p2`. The speed along the path is in this case programmed to be 100 mm/s. The last parameter, `z1`, indicates that the point `p2` is a zone point. This means that the robot will pass in a neighborhood of the point with a distance not more than 1 mm. This can also be seen in Figure 2. The actual position of `p1` in the base coordinate system is x = 1300 mm, y = 100 mm, and z = 660 mm. The configuration of the robot is also shown in Fig. 1.

As a first step in the design of the ILC schemes an identification experiment is performed. The result from this step is three models, one for each joint. The models are calculated using *System Identification Toolbox* [33] and the models are of ARX type with $na = 1$, $nb = 1$, and $nk = 1$,

$$\widehat{T}_{u,1}(q) = \widehat{T}_{u,2}(q) = \frac{0.1q^{-1}}{1 - 0.9q^{-1}}, \quad (20)$$

$$\widehat{T}_{u,3}(q) = \frac{0.13q^{-1}}{1 - 0.87q^{-1}}. \quad (21)$$

The models are now utilized in order to design the different ILC algorithms.

B. Design 1: Heuristic design, Algorithm 1

The design follows the steps in the heuristic design, Algorithm 1.
1. The Q-filter is chosen as a zero-phase low-pass filter, $Q(q) = \bar{Q}(q)\bar{Q}(\frac{1}{q})$. $\bar{Q}(q)$ is a second order Butterworth filter with cut-off frequency at 20 % of the Nyquist frequency.
2. The L-filter has been chosen the same for the three joints, $L(q) = 0.9q^4$, i.e., $\kappa = 0.9$ and $\delta = 4$. This choice can be explained by calculating

$$\sup_{\omega \in [0, \pi/t_s]} |Q(e^{i\omega})(1 - L(e^{i\omega})\widehat{T}_u(e^{i\omega}))|$$

for different choices of δ. For robustness reasons this should be as low as possible and it has a minimum for $\delta = 5$. The choice is here $\delta = 4$ which gives about the same value.

C. Design 2: Model-based design, Algorithm 2

1. The models are given by (20) and (21).
2. An aggressive approach is employed here and H_B is chosen as $H_B(q) = 0$. Robustness is achieved using the Q-filter.
3. With the choice of H_B the L-filters become

$$L_i(q) = T_{u,i}^{-1}(q), \quad i = 1, 2, 3$$

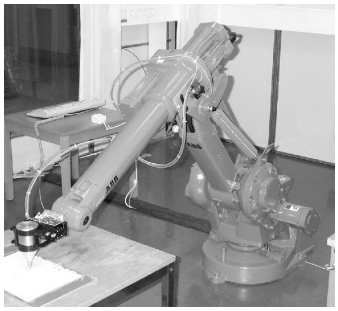


Fig. 1. The ABB IRB1400 manipulator.

```
%% starting at p1
moveL p2,v100,z1;
moveL p3,v100,z1;
moveL p4,v100,z1;
moveL p5,v100,z1;
moveL p6,v100,z1;
moveL p1,v100,fine;
```

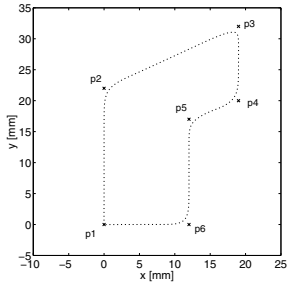


Fig. 2. The program that produces the trajectory used in the example (left) and the resulting trajectory on the arm-side translated such that p1 is the origin (right).

4. The Q-filter is chosen as in design 1 above, i.e., as a zero-phase low-pass filter.

D. Design 3: Optimization based design, Algorithm 3

Here two different values of the parameters in the design of the optimization based ILC algorithm have been chosen. The resulting algorithms will be referred to as, design 3a, and design 3b, respectively.
1. The models of the three joints are given in (20) and (21) and the matrices $\widehat{\boldsymbol{T}}_{u,1}$, $\widehat{\boldsymbol{T}}_{u,2}$, and $\widehat{\boldsymbol{T}}_{u,3}$ are found lower triangular Toeplitz matrices created from the impulse response of (20) and (21).
2. The weight matrices are chosen as $\boldsymbol{W}_e = I$, and $\boldsymbol{W}_u = \rho \cdot I$ with $\rho_a = 0.01$ and $\rho_b = 0.1$ in design 3a and design 3b, respectively. The Lagrange multiplier is chosen as $\lambda = 0.1$.
3. The matrices in the ILC updating equations $\boldsymbol{Q}_i$ and $\boldsymbol{L}_i$, for $i = 1, 2, 3$, are calculated according to

$$\boldsymbol{Q}_i = ((\rho + \lambda) \cdot I + \widehat{\boldsymbol{T}}_{u,i}^T \widehat{\boldsymbol{T}}_{u,i})^{-1} (\lambda \cdot I + \widehat{\boldsymbol{T}}_{u,i}^T \widehat{\boldsymbol{T}}_{u,i})$$
$$\boldsymbol{L}_i = (\lambda \cdot I + \widehat{\boldsymbol{T}}_{u,i}^T \widehat{\boldsymbol{T}}_{u,i})^{-1} \widehat{\boldsymbol{T}}_{u,i}$$

where ρ_a and ρ_b are used. For further aspects of the choice of λ and ρ see [20] or [1].

Note that the choice of ρ only has an effect on the Q-filter, i.e., the robustness of the ILC system. Clearly an increased ρ gives a more robust algorithm but also a lower bandwidth as is shown in [20].

E. Results from the multiple joint motion experiments

The four different ILC algorithms resulting from the three design algorithms have been running for a total of 11 iterations. The resulting normalized maximum errors for design 1 and 2 are shown in Fig. 3 and the corresponding results for the designs 3a and 3b are shown in Fig. 4. In Fig. 3 and Fig. 4 the normalized 2-norm of the error is also presented. The two measures are calculated as

$$V_{k,i,j} = \frac{\|e_{k,i,j}\|}{\max\limits_{l=1,2,3} \max\limits_{m=1,2,3} \|e_{0,l,m}\|} \quad i = 1, 2, 3 \quad j = 1, 2, 3, 4 \quad (22)$$

where i is motor number and j is the design number. The norms are the normalized ∞-norm (the maximum value) in the first case and the normalized 2-norm in the second case.

From the upper row of diagrams in Fig. 3 it can, for example, be seen that the maximum value of the error of joint 1 is about 50 % of what is achieved by joint 2. The initial value of the maximum error as well as the energy in the different experiments is not exactly the same but the behavior of the different algorithms can still be evaluated.

From Fig. 3 and Fig. 4 it is clear that the best result is achieved with the two (explicitly) model based ILC algorithms, i.e., design 2 and design 3a. Clearly, by adjusting the design parameters in design 3 it is possible to get a slower (even slower than design 1) and more robust scheme, as in design 3b, or a faster and less robust one, as in design 3a. Also in this more complicated motion the resulting behavior after 5-6 iterations is very similar for the different ILC algorithms. If it is acceptable to run 5-6 iterations then any of the ILC algorithms can be chosen. The algorithm in design 3b has, however, the disadvantage that there is a large steady state error. This is caused by the fact that the gain of the Q-filter is less than 1.

Another way of evaluating the result of applying ILC to the robot is to transform the measured motor angles to the arm-side using the kinematic model of the robot. In Fig. 5 and Fig. 6 the result from this operation is depicted for designs 1 and 2, and designs 3a and 3b, respectively. It is clear that the error when doing a transformation to the arm-side is not very big but it is important to stress that the transformation has been carried out under the assumption that the robot is stiff which is not true in practice.

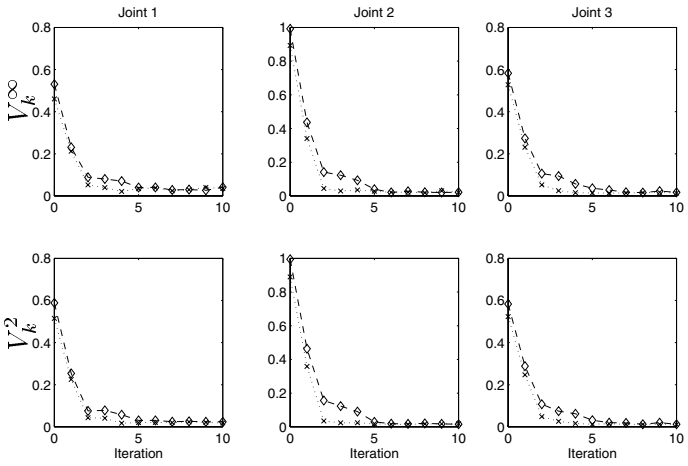


Fig. 3. Normalized maximum error (upper) and normalized energy of the error (lower) for joint 1 to joint 3 from left to right, design 1 ($\diamond$) and design 2 ($\times$).

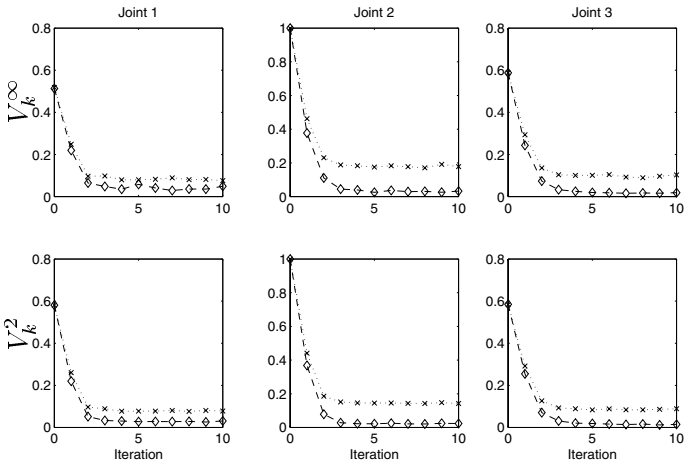


Fig. 4. Normalized maximum error (upper) and normalized energy of the error (lower) for joint 1 to joint 3 from left to right, design 3a ($\diamond$) and design 3b ($\times$).

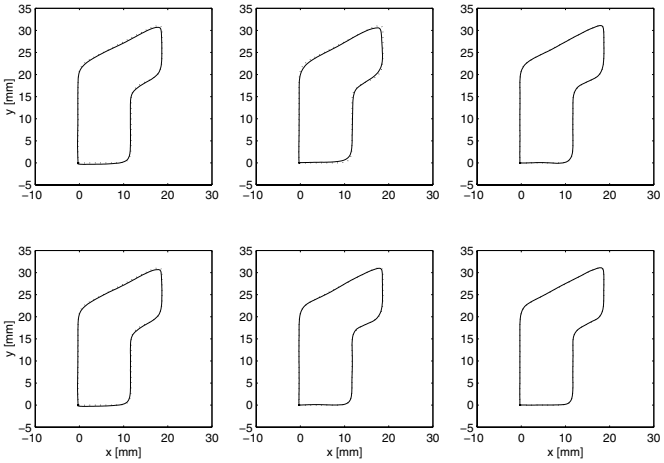


Fig. 5. Resulting trajectory transformed to the arm-side using the forward kinematics (solid) and the reference trajectory (dotted). Upper row, design 1, and lower row, design 2, iteration 0, 1, and 5, from left to right.

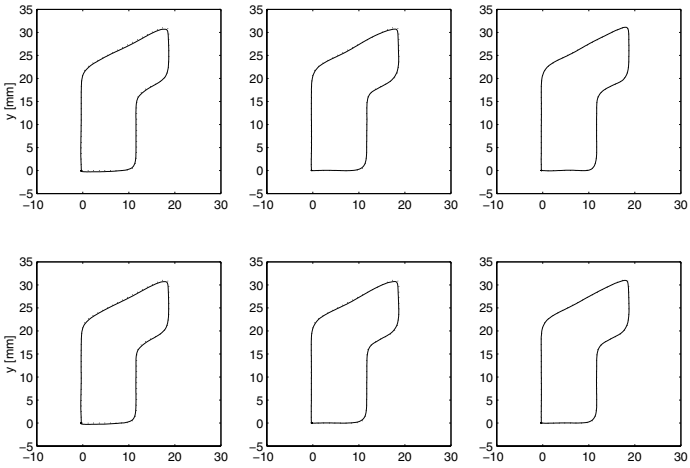


Fig. 6. Resulting trajectory transformed to the arm-side using the forward kinematics (solid) and the reference trajectory (dotted). Upper row, design 3a, and lower row, design 3b, iteration 0, 1, and 5, from left to right.

VI. Summary and Conclusions

Three different design strategies have been presented in detail and they have also been implemented on the industrial robot in a multiple joint motion. Two of the three design methods use explicitly a model of the system which can be considered a bit in contradiction with the original ILC idea. ILC has often been presented as a non model based approach which, as it has been shown here, is not the case. Also the first approach, the *heuristic approach*, uses a model to assure stability of the ILC system.

To decide the best choice of design strategy is not so easy, but there are two general comments that can be made. First, *"try simple things first"*, which means that if Algorithm 1 gives sufficient performance this is the algorithm that should be used. If the performance is not enough a model based approach has to be chosen. The optimization based approach, Algorithm 3, has only a few design parameters to tune the performance/robustness of the algorithm. The design based on Algorithm 2 is also straightforward to apply, at least as it has been done in the two experiments described in this paper. What can be a risk is, however, that the inverse system model solution is

too aggressive and might lead to instability. The Q-filter makes the algorithm more robust but it also limits the bandwidth of the ILC system. The second, and final, comment is, *"use all information available"*. This means that if a good model is available this model should be used for the analysis and also for simulations to evaluate the resulting design.

Acknowledgments

The authors would like to thank VINNOVA's competence center ISIS and CENIIT, both at Linköpings universitet, for the financial support.

References

[1] M. Norrlöf, *Iterative Learning Control: Analysis, Design, and Experiments*, Ph.D. thesis, Linköpings universitet, Linköping, Sweden, 2000, Linköping Studies in Science and Technology. Dissertations; 653. Download from http://www.control.isy.liu.se/publications/.

[2] M. Uchiyama, "Formulation of high-speed motion pattern of a mechanical arm by trial," *Trans. SICE (Soc. Instrum. Contr. Eng.)*, vol. 14, no. 6, pp. 706–712, 1978, Published in Japanese.

[3] Murray Garden, "Learning control of actuators in control systems," US Patent, US03555252, Jan 1971, Leeds & Northrup Company, Philadelphia, USA.

[4] Chen YuangQuang and Kevin.L Moore, "Comments on us patent 3555252: Learning control of actuators in control systems," in *ICARV'00 CD-ROM Proceedings*, 2000, The Sixth International Conference on Control, Automation, Robotics and Vision.

[5] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *Journal of Robotic Systems*, vol. 1, no. 2, pp. 123–140, 1984.

[6] G. Casalino and G. Bartolini, "A learning procedure for the control of movements of robotic manipulators," in *IASTED Symposium on Robotics and Automation*, 1984, pp. 108–111.

[7] J.J. Craig, "Adaptive control of manipulators through repeated trials," in *Proc. of ACC*, San Diego, CA, June 1984.

[8] P. Bondi, G. Casalino, and L. Gambardella, "On the iterative learning control theory for robotic manipulators," *IEEE Journal of Robotics and Automation*, vol. 4, pp. 14–22, Feb 1988.

[9] K. Guglielmo and N. Sadegh, "Theory and implementation of a repetetive robot controller with cartesian trajectory description," *Journal of Dynamic Systems, Measurement, and Control*, vol. 118, pp. 15–21, March 1996.

[10] R. Horowitz, W. Messner, and J. B. Moore, "Exponential convergence of a learning controller for robot manipulators," *IEEE Transactions on Automatic Control*, vol. 36, no. 7, pp. 890–894, July 1991.

[11] F. Lange and G. Hirzinger, "Learning accurate path control of industrial robots with joint elasticity," in *Proc. IEEE Conference on Robotics and Automation*, Detriot, MI, USA, 1999, pp. 2084–2089.

[12] R. Horowitz, "Learning control of robot manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, pp. 402–411, June 1993.

[13] K. L. Moore, *Iterative Learning Control for Deterministic Systems*, Advances in Industrial Control. Springer-Verlag, 1993.

[14] K. L. Moore, "Iterative learning control - an expository overview," *Applied and Computational Controls, Signal Processing and Circuits*, vol. 1, 1998.

[15] Y. Chen and C. Wen, *Iterative Learning Control: Convergence, Robustness and Applications*, vol. 248 of *Lecture Notes in Control and Information Sciences*, Springer-Verlag, 1999.

[16] Z. Bien and J.-X. Xu, *Iterative Learning Control: Analysis, Design, Integration and Application*, Kluwer Academic Publishers, 1998.

[17] N. Amann and D. H. Owens, "Non-minimum phase plants in iterative learning control," in *Second International Conference on Intelligent Systems Engineering*, Technical University of Hamburg, Germany, 1994, pp. 107–112.

[18] N. Amann, D. H. Owens, and E. Rogers, "Iterative learning control using prediction with arbitrary horizon," in *Proceedings of the European Control Conference 1997*, Brussels, Belgium, July 1997.

[19] K.S. Lee and J.H. Lee, *Design of Quadratic Criterion-Based Iterative Learning Control. In Iterative Learning Control: Analysis, Design, Integration and Applications. Z. Bien and J.X. Xu., eds.*, Kluwer Academic Publishers, 1998.

[20] S. Gunnarsson and M. Norrlöf, "On the design of ILC algorithms using optimization," *Automatica*, vol. 37, pp. 2011–2016, 2001.

[21] M. Norrlöf, "An adaptive iterative learning control algorithm with experiments on an industrial robot," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 245–251, April 2002.

[22] Jay. H. Lee, Kwang S. Lee, and Won C. Kim, "Model-based iterative learning control with a quadratic criterion for time-varying linear systems," *Automatica*, vol. 36, no. 5, pp. 641–657, May 2000.

[23] M. Togai and O. Yamano, "Analysis and design of an optimal learning control scheme for industrial robots: a discrete system approach," in *Proc. of the 24th IEEE Conf. on Decision and Control*, Ft. Lauderdale, FL., Dec 1985, pp. 1399–1404.

[24] J. S. Park and T. Hesketh, "Model reference learning control for robot manipulators," in *Proceedings of the IFAC 12th Triennial World Congress*, Sydney, Australia, 1993, pp. 341–344.

[25] Y.-J. Liang and D. P. Looze, "Performance and robustness issues in iterative learning control," in *Proc. of the 32nd Conf. on Decision and Control*, San Antonio, Texas, USA, Dec 1993, pp. 1990–1995.

[26] D. de Roover, "Synthesis of a robust iterative learning controller using an H_∞ approach," in *Proceedings of 35th Conference on Decision and Control*, Kobe, Japan, 1996, pp. 3044–3049.

[27] S. Gunnarsson and M. Norrlöf, "On the use of learning control for improved performance in robot control systems," in *Proceedings of the European Control Conference 1997*, Brussels, Belgium, July 1997.

[28] S. Gunnarsson and M. Norrlöf, "Some experiences of the use of iterative learning control for performance improvement in robot control systems," in *Preprints of the 5th IFAC symposium on robot control*, Nantes, France, Sep 1997, vol. 2.

[29] D.M. Gorinevsky, D. Torfs, and A.A. Goldenberg, "Learning approximation of feedforward dependence on the task parameters: Experiments in direct-drive manipulator tracking," in *Proc. ACC*, Seattle, Washington, 1995, pp. 883–887.

[30] J.A Frueh M.Q. Phan, "Linear quadratic optimal learning control (LQL)," in *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, Florida, USA, 1998, pp. 678–683.

[31] JR. J.E. Dennis and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632, 1983.

[32] L. Ljung, *System Identification: Theory for the User*, Prentice-Hall, second edition, 1999.

[33] L. Ljung, *System Identification Toolbox - For Use with Matlab*, The MathWorks Inc., 1995.