Linear Parameter Varying Iterative Learning Control with application to a Linear Motor System

Mark Butcher and Alireza Karimi

***Abstract*— In this paper an Iterative Learning Control (ILC) algorithm is proposed for a certain class of Linear Parameter Varying (LPV) systems whose dynamics change between iterations. Consistency of the algorithm in the presence of stochastic disturbances is shown. The proposed algorithm is tested in simulation and the obtained tracking performance is compared with that obtained using a standard Linear Time Invariant ILC algorithm. Better results are obtained using the proposed method. The method is also applied to a linear, permanent magnet synchronous motor system, which is shown to be an LPV system for a specific class of movements. Greatly improved tracking is achieved.**

***Index Terms*— Learning control systems, linear parameter-varying system, linear motors, motion control**

I. Introduction

Iterative Learning Control (ILC) is now well recognised as a methodology capable of producing very high tracking performance for systems carrying out repetitive tasks [1], [2]. ILC adjusts the system's input from one repetition/iteration to the next in order to compensate for the system's dynamics and disturbances, thus improving its output tracking performance. ILC has been successfully applied to numerous systems carrying out repetitive tasks (e.g. [3], [4], [5], [6]). One of the main drawbacks of ILC, however, is that it requires the system's dynamics and disturbances to be repetition invariant. This assumption may not always be valid in practice i.e. when the output sensor is affected by non-negligible measurement noise or if a robotic arm picks up different loads from one repetition to the next. If the variations are of a stochastic nature, they should not be compensated for and ILC algorithms that are insensitive to these variations should be used (e.g. [7]). If, however, they are deterministic, it may be possible to compensate for the changes. It is therefore of interest to investigate ILC algorithms that are applicable when the system's dynamics and disturbances change deterministically between repetitions.

To the authors' knowledge, very little work has been done on this problem. In [8] the problem of deterministically iteration varying disturbances is considered. It is shown that, by using the internal-model principle in the iteration domain, the disturbances can be rejected as the iterations tend to infinity. The problem with this approach is that it is necessary to know the form of the disturbance variation in advance in order to include its model in the ILC controller, as required by the internal-model principle.

The authors are with the Automatic Control Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland.

Corresponding author: alireza.karimi@epfl.ch

This work is supported by the Swiss National Science Foundation under Grant No. 200021-116156/1.

In this paper we develop an ILC algorithm that can lead to improved tracking for systems that can be represented by the Linear Parameter Varying (LPV) class of systems [9], and therefore whose dynamics change as a function of a measurable scheduling parameter. Development and application of control techniques for LPV systems has been active in recent years (e.g. [10], [11]). ILC for LPV systems has been considered in [12]. The variation of the system's dynamics due to the changing scheduling parameter is, however, assumed to take place during the iteration, rather than from one iteration to the next. The problem considered is, therefore, different to that studied here.

The method developed in this paper is applied to a linear, permanent magnet, synchronous motor system (LPMSM). LPMSMs are very stiff and have no mechanical transmission components. They, therefore, do not suffer from backlash and so allow very high positioning accuracy to be achieved. Typical uses for this type of motor include wafer stages, microscale robotic decomposition, electronic assembly and manufacturing, and machine tools. The precision required by all of these applications is continuously increasing. When the required movements are repetitive, ILC is an obvious choice to achieve this precision [13], [14], [15], [16], [17]. LPMSMs, however, are affected by a periodic, position-dependent force ripple disturbance. When a movement starts from the same place this disturbance will be repetitive. ILC can thus adjust the system's input to compensate for it [18]. However, if the movement starts from a different position, the disturbance will change and the learnt input will no longer produce optimal tracking. This problem has been investigated in [19]. The method proposed there is to learn the input that gives optimal tracking for a specific starting position. This input will compensate for errors due to the system's dynamics and the ripple force disturbance. It can thus be decomposed into these two components. When the movement is executed from a different starting position the component compensating the system's dynamics can be applied to the system, plus a phase-shifted version of the component that compensates the periodic, position dependent disturbance. The phase shift will be a function of the distance between starting positions. This method has the benefit that the input only needs to be learnt once, rather than for every starting position. Its disadvantage is that the learning process has to be done offline as a tuning procedure in order that the same starting position is used at each iteration.

In this paper it will be shown how, for a certain class of movements whose amplitude is negligible compared to the period of the force ripple, the LPMSM can be modelled as an LPV system with the position-dependent force ripple as the scheduling parameter. The proposed LPV ILC algorithm is therefore applicable to the LPMSM.

The paper is organised as follows. In Section II the LPV ILC algorithm is presented. The algorithm is then tested in simulation in Section III. It is then applied to the LPMSM, as detailed in Section IV. Finally some conclusions are made in Section V.

II. Linear Parameter Varying ILC

In this section the LPV ILC algorithm is described. First the class of systems considered in this paper is outlined. The ideal input for this class of systems is then given. A parameterisation for the input to be estimated is then presented, based on the structure of the ideal input. Next a recursive algorithm, similar to the standard recursive least squares but taking into account the iteration-varying aspect of LPV systems, is developed to estimate the ideal input. Finally it is shown that, under a persistency of excitation condition on the scheduling parameter, the learning algorithm converges probabilistically to the ideal input.

A. System description

We consider the finite-time tracking problem of following a repetitive, finite duration desired trajectory $y_d(t)$, defined for $t = 0, \dots, N-1$.

The output at time t of the stable, LPV SISO discrete-time system, resulting from linearising a nonlinear system about the operating point $\boldsymbol{\sigma}(k) \in \mathbb{R}^{n_\sigma}$, is given by:

$$\begin{aligned} &A(\boldsymbol{\sigma}(k), q^{-1}) y(t,k,\boldsymbol{\sigma}(k)) \\ &\quad = B(q^{-1})u(t,k) + d(t,k,\boldsymbol{\sigma}(k)) + n(t,k,\boldsymbol{\sigma}(k)), \end{aligned} \quad (1)$$

where

$$A(\boldsymbol{\sigma}(k), q^{-1}) = \sum_{j=0}^{n_a} a_j(\boldsymbol{\sigma}(k)) q^{-j}, \quad B(q^{-1}) = \sum_{j=0}^{n_b} b_j q^{-j},$$

$u(t,k)$ is the input to the system and q^{-1} is the backward-shift time operator. $d(t,k,\boldsymbol{\sigma}(k))$ and $n(t,k,\boldsymbol{\sigma}(k))$ are a deterministic and a stochastic disturbance, respectively, both possibly dependent on $\boldsymbol{\sigma}$. The operating point $\boldsymbol{\sigma}(k)$ remains constant throughout repetition k. The dependence of the coefficients a_i and the deterministic disturbance on the scheduling parameter is assumed polytopic:

$$\begin{aligned} a_i(\boldsymbol{\sigma}(k)) &= \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) a_{i,j} \\ \text{and} \qquad d(t,k,\boldsymbol{\sigma}(k)) &= \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) d_j(t), \\ 0 \le \lambda_j(\boldsymbol{\sigma}(k)) \le 1, \quad &\sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) = 1, \end{aligned} \quad (2)$$

where $\lambda_j(\boldsymbol{\sigma}(k)) : \mathbb{R}^{n_\sigma} \to \mathbb{R}$. Additionally we designate the values of $\boldsymbol{\sigma}$ at the vertices of the polytopic space as $\boldsymbol{\sigma}_j$.

As the signals are defined over the finite duration of the repetition, it is possible to express the system's input-ouput relationship by a matrix representation. Using the lifted-system representation typically used in ILC, we define, for a system with a relative degree of m, the vectors:

$$\begin{aligned} \mathbf{u}(k) &= [u(0,k), u(1,k), \dots, u(N-m-1,k)]^T \\ \text{and} \quad \mathbf{y}(k,\boldsymbol{\sigma}(k)) &= [y(m,k,\boldsymbol{\sigma}(k)), y(m+1,k,\boldsymbol{\sigma}(k)), \\ &\quad \dots, y(N-1,k,\boldsymbol{\sigma}(k))]^T, \end{aligned} \quad (3)$$

with $\mathbf{y_d}$, $\mathbf{d_j}$ and $\mathbf{n}(k,\boldsymbol{\sigma}(k))$ defined similarly to $\mathbf{y}(k,\boldsymbol{\sigma}(k))$. This representation can then be used to write (1) as:

$$\begin{aligned} \mathbf{A}(\boldsymbol{\sigma}(k))\mathbf{y}(k,\boldsymbol{\sigma}(k)) &= \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{A}_j \mathbf{y}(k,\boldsymbol{\sigma}(k)) \\ &= \mathbf{B}\mathbf{u}(k) + \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{d}_j + \mathbf{n}(k,\boldsymbol{\sigma}(k)), \end{aligned} \quad (4)$$

$$\text{where} \quad \mathbf{A}_j = \begin{bmatrix} a_{0,j} & 0 & \dots & 0 \\ a_{1,j} & a_{0,j} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{N-m-1,j} & a_{N-m-2,j} & \dots & a_{0,j} \end{bmatrix}$$

$$\text{and} \quad \mathbf{B} = \begin{bmatrix} b_m & 0 & \dots & 0 \\ b_{m+1} & b_m & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_{N-1} & b_{N-2} & \dots & b_m \end{bmatrix}. \quad (5)$$

Assumptions: The disturbance vector $\mathbf{n}(k,\boldsymbol{\sigma}(k))$ is assumed to be a zero-mean, random vector with unknown but bounded covariance matrix $\mathbf{R_n}(k,\boldsymbol{\sigma}(k))$. Additionally, realisations of $\mathbf{n}(k,\boldsymbol{\sigma}(k))$ are considered independent.

Later in the paper the following lemma will be needed:

Lemma 1: [p. 253 in [20]] Let $X(k)$ be an independent random sequence with constant mean μ_X and variance $\sigma_X^2(k)$ defined for $k \ge 1$. Define another random sequence as:

$$\hat{\mu}_X(K) = \frac{1}{K} \sum_{k=1}^{K} X(k) \quad \text{for } K \ge 1. \quad (6)$$

Then if

$$\lim_{K\to\infty} \sum_{k=1}^{K} \frac{\sigma_X^2(k)}{k^2} < \infty, \quad (7)$$

$$\hat{\mu}_X(K) \xrightarrow{P} \mu_X \text{ as } K \to \infty, \quad (8)$$

where $\xrightarrow{P}$ represents convergence in probability.

The theorem also holds true for almost sure (a.s.) convergence, making it a Strong Law.

B. Ideal input

The measured error is defined as:

$$\mathbf{e}(k,\boldsymbol{\sigma}(k)) = \mathbf{y_d} - \mathbf{y}(k,\boldsymbol{\sigma}(k)). \quad (9)$$

The ideal input vector, defined as the one that achieves zero mean error, i.e. $E\{\mathbf{e}(k,\boldsymbol{\sigma}(k))\} = \mathbf{0}\ \forall\boldsymbol{\sigma}$, where $E\{\cdot\}$ denotes the mathematical expectation operator, is given by:

$$\mathbf{u}^0(k,\boldsymbol{\sigma}(k)) = \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{B}^{-1} \left(\mathbf{A}_j \mathbf{y_d} - \mathbf{d}_j\right). \quad (10)$$

At the values of $\boldsymbol{\sigma}$ that correspond to the vertices of the polytopic dependence we have:

$$\mathbf{u}^0(k,\boldsymbol{\sigma}_j) = \mathbf{B}^{-1}\left(\mathbf{A}_j\mathbf{y_d} - \mathbf{d}_j\right). \tag{11}$$

Using this (10) can be written as:

$$\mathbf{u}^0(k,\boldsymbol{\sigma}(k)) = \sum_{j=0}^{J-1}\lambda_j(\boldsymbol{\sigma}(k))\mathbf{u}^0(\boldsymbol{\sigma}_j). \tag{12}$$

Remark: Expression (12) rationalises the class of LPV systems considered here i.e. LPV systems with dependence on the scheduling parameter solely in the denominator. Only for this system class does the ideal input depend linearly on the ideal inputs at the vertices. This linear dependence means the estimation of these inputs, as considered in the next subsection, can be done via linear least squares and there will be a global minimum. Furthermore, as will be seen in the application section, real systems exist that belong to this system class.

C. Input parameterisation

We see from (12) that the ideal input is a function of the scheduling parameter $\boldsymbol{\sigma}(k)$. The ILC algorithm should therefore estimate an input that is also a function of $\boldsymbol{\sigma}(k)$. Motivated by the form of the ideal input, and under the assumption that the functions $\lambda_j(\boldsymbol{\sigma}(k))$ are known, the input is parameterised as:

$$\mathbf{u}(k,\boldsymbol{\sigma}(k)) = \sum_{j=0}^{J-1}\lambda_j(\boldsymbol{\sigma}(k))\mathbf{u}_j. \tag{13}$$

The system output can be written as:

$$\begin{aligned}\mathbf{y}(k,\boldsymbol{\sigma}(k)) &= \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\left[\mathbf{B}\mathbf{u}(k) + \mathbf{d}(k,\boldsymbol{\sigma}(k))\right.\\ &\qquad \left.+\mathbf{n}(k,\boldsymbol{\sigma}(k))\right]\\ &= \mathbf{G}(\boldsymbol{\sigma}(k))\mathbf{u}(k) + \bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k)) + \bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k)),\end{aligned} \tag{14}$$

where $\mathbf{G}(\boldsymbol{\sigma}(k)) = \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{B}$, $\bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k)) = \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{d}(k,\boldsymbol{\sigma}(k))$ and $\bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k)) = \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{n}(k,\boldsymbol{\sigma}(k))$. It should be mentioned that for $\mathbf{G}(\boldsymbol{\sigma}(k))$, $\bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k))$ and $\bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k))$ to exist, $\mathbf{A}(\boldsymbol{\sigma}(k))$ should be nonsingular, which is always the case due to its structure. If the parameterised input (13) is applied to the system we have:

$$\begin{aligned}\mathbf{y}(k,\boldsymbol{\sigma}(k)) &= \mathbf{G}(\boldsymbol{\sigma}(k))\sum_{j=0}^{J-1}\lambda_j(\boldsymbol{\sigma}(k))\mathbf{u}_j + \bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k))\\ &\quad + \bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k))\\ &= \left[\lambda_0(\boldsymbol{\sigma}(k))\mathbf{G}(\boldsymbol{\sigma}(k)), \lambda_1(\boldsymbol{\sigma}(k))\mathbf{G}(\boldsymbol{\sigma}(k)), \ldots,\right.\\ &\quad \left.\lambda_{J-1}(\boldsymbol{\sigma}(k))\mathbf{G}(\boldsymbol{\sigma}(k))\right]\left[\mathbf{u}_0^T, \mathbf{u}_1^T, \ldots, \mathbf{u}_{J-1}^T\right]^T\\ &\quad + \bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k)) + \bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k))\\ &= \mathcal{G}(\boldsymbol{\sigma}(k))\mathbf{U} + \bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k)) + \bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k)),\end{aligned} \tag{15}$$

where $\mathcal{G}(\boldsymbol{\sigma}(k)) \in \mathbb{R}^{(N-m)\times J(N-m)}$ and $\mathbf{U} \in \mathbb{R}^{J(N-m)}$.

The ideal input (12) is achieved when:

$$\mathbf{U} = \mathbf{U}^0 = \left[[\mathbf{u}^0(\boldsymbol{\sigma}_0)]^T, [\mathbf{u}^0(\boldsymbol{\sigma}_1)]^T, \ldots, [\mathbf{u}^0(\boldsymbol{\sigma}_{J-1})]^T\right]^T. \tag{16}$$

D. The learning algorithm

The aim of the algorithm is to estimate $\mathbf{U}^0$ over the iterations. The approach proposed here to find the estimate is to minimise a quadratic cost function over all previous iterations i.e. to find the estimate that minimises:

$$J_K(\mathbf{U}) = \frac{1}{2K}\sum_{k=1}^{K}\mathbf{e}^T(k,\boldsymbol{\sigma}(k),\mathbf{U})\mathbf{e}(k,\boldsymbol{\sigma}(k),\mathbf{U}), \tag{17}$$

where K is the number of completed iterations. Via some simple calculations, the estimate can be found as:

$$\hat{\mathbf{U}}_K = \mathbf{P}(K)\sum_{k=1}^{K}\mathcal{G}^T(\boldsymbol{\sigma}(k))[\mathbf{y_d} - \bar{\mathbf{d}}(k,\boldsymbol{\sigma}(k)) - \bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k))], \tag{18}$$

where

$$\mathbf{P}(K) = \left[\sum_{k=1}^{K}\mathcal{G}^T(\boldsymbol{\sigma}(k))\mathcal{G}(\boldsymbol{\sigma}(k))\right]^{-1}.$$

Alternatively, via some standard manipulations, (18) can be written in the recursive form as:

$$\begin{aligned}\hat{\mathbf{U}}_{k+1} &= \hat{\mathbf{U}}_k\\ &\quad + \mathbf{P}(k+1)\mathcal{G}^T(\boldsymbol{\sigma}(k+1))\mathbf{e}(k+1,\boldsymbol{\sigma}(k+1),\hat{\mathbf{U}}_k).\end{aligned} \tag{19}$$

The error signal:

$$\begin{aligned}\mathbf{e}(k+1,\boldsymbol{\sigma}(k+1),\hat{\mathbf{U}}_k) &= \mathbf{y_d} - \mathcal{G}(\boldsymbol{\sigma}(k+1))\hat{\mathbf{U}}_k\\ &\quad - \bar{\mathbf{d}}(k+1,\boldsymbol{\sigma}(k+1)) - \bar{\mathbf{n}}(k+1,\boldsymbol{\sigma}(k+1))\end{aligned}$$

can be evaluated experimentally by applying the input $\mathbf{u}(k,\boldsymbol{\sigma}(k+1))$ from (13) based on $\hat{\mathbf{U}}_k$ to the real system. Therefore we see that the estimate $\hat{\mathbf{U}}$ can be evaluated recursively using data measured from the real system.

E. Consistency of estimates

Next a condition for consistent estimates is given.

Theorem 1: Under the assumptions made in Subsection II-A, the algorithm (19) is a consistent estimator, i.e. $\hat{\mathbf{U}}_K$ converges almost surely to $\mathbf{U}^0$ as $K \to \infty$, if:

$$\lim_{K\to\infty}\frac{1}{K}\mathbf{P}^{-1}(K) \tag{20}$$

is nonsingular.

Proof: The recursive algorithm (19) has the same asymptotic properties as the batch result (18) so the consistency of (18) can be considered. (18) can be rewritten as:

$$\begin{aligned}\hat{\mathbf{U}}_K &= K\mathbf{P}(K)\\ &\qquad \frac{1}{K}\sum_{k=1}^{K}\mathcal{G}^T(\boldsymbol{\sigma}(k))\left[\mathcal{G}(\boldsymbol{\sigma}(k))\mathbf{U}^0 - \bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k))\right]\\ &= \mathbf{U}^0 - K\mathbf{P}(K)\frac{1}{K}\sum_{k=1}^{K}\mathcal{G}^T(\boldsymbol{\sigma}(k))\bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k)).\end{aligned}$$

In order for the estimates to be consistent it is necessary that:

$$\lim_{K\to\infty}\frac{1}{K}\mathbf{P}^{-1}(K) \tag{21}$$

be nonsingular and

$$\mathbf{w}(K) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{G}^T(\boldsymbol{\sigma}(k))\bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k)) \to \mathbf{0} \quad \text{a.s., as } K \to \infty. \quad (22)$$

(21) is the condition in the theorem that should be satisfied. To show that (22) is true we first write:

$$\begin{aligned}\mathcal{G}^T(\boldsymbol{\sigma}(k))\bar{\mathbf{n}}(k,\boldsymbol{\sigma}(k)) &= \mathcal{G}^T(\boldsymbol{\sigma}(k))\mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{n}(k,\boldsymbol{\sigma}(k)) \\ &= \mathbf{x}(k,\boldsymbol{\sigma}(k)).\end{aligned}$$

Since $\mathbf{n}(k,\boldsymbol{\sigma}(k))$ is assumed to be zero mean and independent between iterations, and $\mathbf{A}$ is nonsingular, this expression means that $\mathbf{x}(k,\boldsymbol{\sigma}(k))$ will also be zero mean and independent between iterations. Additionally $\mathbf{x}(k,\boldsymbol{\sigma}(k))$'s covariance matrix is given by:

$$\begin{aligned}&\mathbf{R}_{\mathbf{x}}(k,\boldsymbol{\sigma}(k)) \\ &= \mathcal{G}^T(\boldsymbol{\sigma}(k))\mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{R}_{\mathbf{n}}(k,\boldsymbol{\sigma}(k))\mathbf{A}^{-T}(\boldsymbol{\sigma}(k))\mathcal{G}(\boldsymbol{\sigma}(k)).\end{aligned}$$

Since $\mathbf{R}_{\mathbf{n}}(k,\boldsymbol{\sigma}(k))$ is assumed bounded and $\mathbf{A}$ is nonsingular, $\mathbf{R}_{\mathbf{x}}(k,\boldsymbol{\sigma}(k))$ will be bounded.

The ith component of $\mathbf{w}(K)$ in (22) therefore represents the sample average of a sequence of zero-mean, independent random variables with finite, though possibly different, variances $\sigma_x^2(i,k,\boldsymbol{\sigma}(k))$. Lemma 1 implies that (22) is satisfied if:

$$\lim_{K\to\infty}\sum_{k=1}^{K}\frac{\sigma_x^2(i,k,\boldsymbol{\sigma}(k))}{k^2} < \infty. \quad (23)$$

Since

$$\sum_{k=1}^{K}\frac{\sigma_x^2(i,k,\boldsymbol{\sigma}(k))}{k^2} \le \overline{\sigma}_x^2(i)\sum_{k=1}^{K}\frac{1}{k^2}$$

where $\sigma_x^2(i,k,\boldsymbol{\sigma}(k)) \le \overline{\sigma}_x^2(i) < \infty\ \forall k$, and

$$\lim_{K\to\infty}\sum_{k=1}^{K}\frac{1}{k^2} = \frac{\pi}{6},$$

(23) is satisfied. The theorem is therefore proved. ■

Remark: The condition (20) in Theorem 1 is a persistency of excitation condition that requires the scheduling parameter trajectory to be sufficiently rich.

III. Simulation

The effectiveness of the proposed method is next shown via simulation. The LPV system used in the simulation is defined by the polynomials:

$$B(q^{-1}) = (0.0048 + 0.0047q^{-1})q^{-1} \quad (24)$$

and

$$A(\boldsymbol{\sigma}(k), q^{-1}) = 1 + a_1(\boldsymbol{\sigma}(k))\,q^{-1} + a_2(\boldsymbol{\sigma}(k))\,q^{-2} \quad (25)$$

where

$$a_1(\boldsymbol{\sigma}(k)) = -1.8953\lambda_0(\boldsymbol{\sigma}(k)) - 1.8903\lambda_1(\boldsymbol{\sigma}(k)) \quad (26)$$

and

$$a_2(\boldsymbol{\sigma}(k)) = 0.9048\lambda_0(\boldsymbol{\sigma}(k)) + 0.9098\lambda_1(\boldsymbol{\sigma}(k)). \quad (27)$$

The functions $\lambda_j(\boldsymbol{\sigma}(k))$ are given by:

$$\lambda_0(\boldsymbol{\sigma}(k)) = \frac{\overline{\boldsymbol{\sigma}} - \boldsymbol{\sigma}(k)}{\overline{\boldsymbol{\sigma}} - \underline{\boldsymbol{\sigma}}}, \quad \lambda_1(\boldsymbol{\sigma}(k)) = \frac{\boldsymbol{\sigma}(k) - \underline{\boldsymbol{\sigma}}}{\overline{\boldsymbol{\sigma}} - \underline{\boldsymbol{\sigma}}},$$

where $\underline{\boldsymbol{\sigma}}$ and $\overline{\boldsymbol{\sigma}}$ are the respective minimum and maximum values of the scheduling parameter $\boldsymbol{\sigma}(k)$, which is taken as a uniformly distributed signal in the interval (0,1). We, therefore, have that $\underline{\boldsymbol{\sigma}} = 0$ and $\overline{\boldsymbol{\sigma}} = 1$.

The desired output $y_d(t)$ is defined by:

$$y_d(t) = 1 - \cos(0.01\pi t) \qquad t = 0, 1, \ldots, 200. \quad (28)$$

The disturbances $d(t,k,\boldsymbol{\sigma}(k))$ and $n(t,k,\boldsymbol{\sigma}(k))$ are set to zero to emphasise the proposed algorithm's ability to compensate the changing dynamics.

The proposed method is applied to the system, however, in order to investigate the method's robustness to uncertainty, instead of using the matrix $\mathbf{G}(\boldsymbol{\sigma}(k))$ in the algorithm, the constant matrix $\mathbf{G}(\underline{\boldsymbol{\sigma}})$ is used, independently of the current value of $\boldsymbol{\sigma}(k)$.

In the implementation of the algorithm the matrix:

$$\mathbf{P}(k+1) = \left[\sum_{i=1}^{k+1}\mathcal{G}^T(\boldsymbol{\sigma}(k))\mathcal{G}(\boldsymbol{\sigma}(k))\right]^{-1}$$

where

$$\mathcal{G}(\boldsymbol{\sigma}(k)) = \left[\lambda_0(\boldsymbol{\sigma}(k))\mathbf{G}(\underline{\boldsymbol{\sigma}}), \lambda_1(\boldsymbol{\sigma}(k))\mathbf{G}(\underline{\boldsymbol{\sigma}})\right]$$

is required at each iteration of the algorithm. In order for the inverse to exist, it is necessary that this matrix be full rank. For this to be the case a persistency of excitation condition on the scheduling parameter trajectory should be satisfied. A necessary, but not sufficient, condition for the matrix to be full rank is that 2 different values of $\boldsymbol{\sigma}(k)$ are visited.[1] In order to satisfy this the algorithm is not used to calculate the system input until 2 different values of $\boldsymbol{\sigma}(k)$ have been visited. $\mathbf{u}(k,\boldsymbol{\sigma}(k)) = \mathbf{y_d}$ is used for $k = 1, 2$.

The results obtained using the proposed method are compared with those obtained using an ILC algorithm developed under the assumption of the system being LTI. The algorithm is given by:

$$\mathbf{u}(k+1) = \mathbf{u}(k) + \frac{1}{k}\mathbf{G}^{-1}(\underline{\boldsymbol{\sigma}})\mathbf{e}(\mathbf{u}(k)). \quad (29)$$

This algorithm can be motivated either by stochastic approximation theory [7] or as the equivalent recursive version of the least squares solution when the 2-norm of the tracking error is minimised over all iterations up to iteration $k+1$. The latter is the same motivation as that used to develop the LPV ILC algorithm so makes the comparison fair. The algorithm is tested in simulation using the same signal for $\boldsymbol{\sigma}(k)$ and $y_d(t)$.

The RMS values $\|\mathbf{e}(k)\|_2$ achieved using the LPV and LTI ILC algorithms are shown in Figure 1. It can be clearly seen that the proposed method reduces the RMS value of the error substantially over the iterations and the error converges towards zero. There is, however, no obvious trend in the error

[1]This condition arises because $\mathcal{G}(\boldsymbol{\sigma}(k)) \in \mathbb{R}^{(N-m)\times 2(N-m)}$ is of maximum rank $N-m$, and thus so is $\mathcal{G}^T(\boldsymbol{\sigma}(k))\mathcal{G}(\boldsymbol{\sigma}(k))$. $\mathbf{P}^{-1}(k+1)$ must be of rank $2(N-m)$. This is only possible after 2 different values of $\boldsymbol{\sigma}(k)$ are visited.

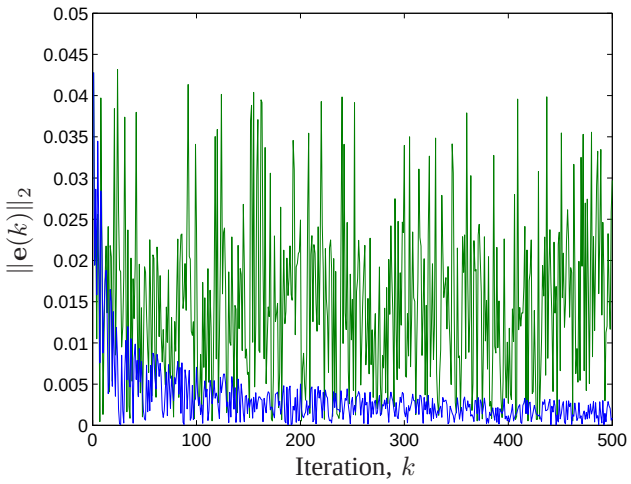

Fig. 1. RMS value of $\mathbf{e}(k)$ obtained in simulation using the proposed algorithm (blue) and the LTI algorithm (green)

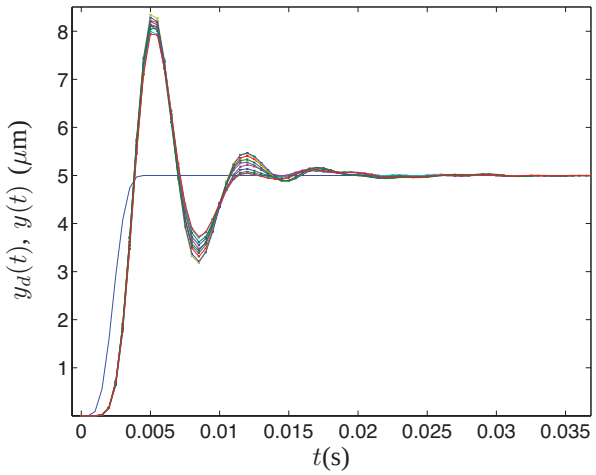

Fig. 3. Mean system response at 10 different positions to desired output $y_d(t)$ (blue) applied as closed-loop reference

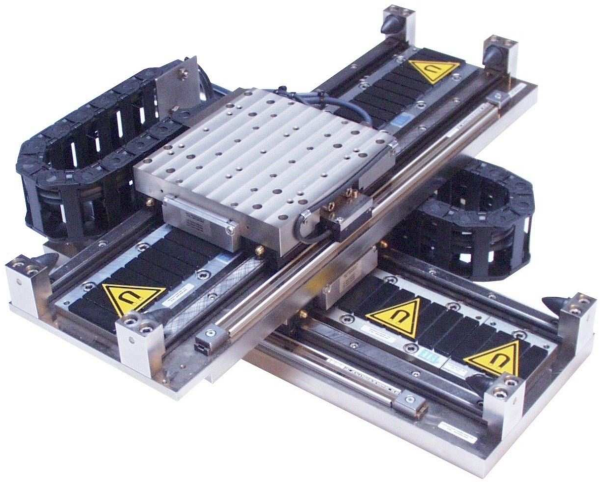

Fig. 2. Linear motor (courtesy of ETEL)

achieved using the LTI algorithm. This lack of tracking improvement is because the LTI algorithm is not able to learn the correct input as the system's dynamics change continuously between iterations.

IV. Application

As mentioned previously, the proposed method is applied to a linear, permanent magnet, synchronous motor. This LPMSM forms the upper axis of an x-y positioning table, see Fig. 2.

A PID feedback controller is used to control the motor's position. It operates at a sampling frequency of 2kHz. An analog position encoder using sinusoidal signals with periods of 2μm, which are then interpolated with 8192 intervals/period to obtain a resolution of 0.24nm, is used to measure the motor's position. However, the accuracy of this type of encoder is limited to 20nm.

The input, $\mathbf{u}(k, \boldsymbol{\sigma}(k+1))$, computed by the ILC algorithm, is used as the closed-loop system's position reference signal.

The desired output motion is a low-pass filtered step type movement with an amplitude of 5 μm in one direction followed by a similar movement back to the starting position. This movement is typical in the semiconductor industry. In this paper we consider the problem of improving the tracking performance of the LPMSM when it executes the movement repetitively but from a different starting position each repetition. This scenario is clearly possible in an industrial setting. If the system's dynamics and disturbances were position-independent, this problem could be solved with a standard LTI ILC algorithm. This, however, is not the case for the system considered as can be seen from Figures 3 and 4. In these figures the mean system's response is shown when the movement is carried out at 10 different starting positions spaced 1.6 mm apart. The movements are repeated 10 times at each position in order to be able to differentiate between the effect of stochastic disturbances on the system's response and that due to changing dynamics. It can be clearly seen that significantly different responses are obtained which differ by an amount greater than could be expected from stochastic effects.

A. System Modelling

Experimentally, a position dependence of the system's response has been shown. In order to understand where this comes from a model of the system will now be developed. The position $y(t,k)$ of the translator of the LPMSM, at time t and repetition k, obeys the following equation:

$$\frac{d^2y(t,k)}{dt^2} = f_m(t,k) - \alpha\frac{dy(t,k)}{dt} - f_r(y(t,k)) \qquad (30)$$

where $f_m(t,k)$ is the force applied to the motor, α is the viscous friction coefficient and $f_r(y(t,k))$ is the force ripple. Since the system has a current loop whose dynamics are much faster than the position dynamics, the force applied to the motor can be considered proportional to the applied current i.e. $f_m(t,k) = k_m i(t,k)$. The force ripple term contains both the

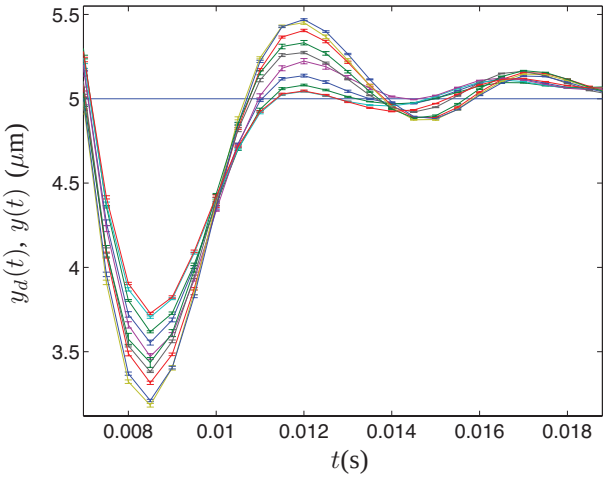

Fig. 4. Zoom of mean system response at different positions with error bars showing 1 standard deviation

cogging force and reluctance force, and is primarily a periodic function of the system's position $y(t,k)$ that can be modelled by:

$$f_r(y(t,k)) = \sum_{i=0}^{n_r-1} F_{r,i} \sin\left(\frac{2\pi y(t,k)}{T_{r,i}} + \phi_{r,i}\right), \qquad (31)$$

where the periods $T_{r,i}$ are influenced by different factors such as the average pitch of the magnets and the size of the bearings. We see that this force has a nonlinear dependence on the system's position making the system, itself, nonlinear.

The developed ILC algorithm is for the LPV system class. The system represented by (30) does not have the form of an LPV system. Nonetheless for the specific class of movements considered in this work, whose amplitude is negligible compared to the force ripple's principal period, certain manipulations can be made in order to obtain an LPV model. Under this assumption on the movement amplitude, the force ripple can be assumed to vary linearly in a small zone about a certain operating point corresponding to a specific starting position $\bar{y}(k)$. Using this idea we can obtain an LPV model, whose dynamics change as a function of the starting position, by linearising the force ripple term about this operating point, giving:

$$\frac{d^2}{dt^2}\delta y(t,k) = \delta f_m(t,k) - \alpha\frac{d}{dt}\delta y(t,k) - \boldsymbol{\sigma}(\bar{y}(k))\delta y(t,k), \qquad (32)$$

where $\delta y(t,k)$ is a small position deviation about $\bar{y}(k)$, $\delta f_m(t,k)$ is a small force deviation about the force required to maintain the system at the operating point and

$$\begin{aligned}\boldsymbol{\sigma}(\bar{y}(k)) &= \frac{df_r(y(t,k))}{dy(t,k)}\Big|_{y(t,k)=\bar{y}(k)} \\ &= \sum_{i=0}^{n_r-1}\left(\frac{2\pi F_{r,i}}{T_{r,i}}\right)\cos\left(\frac{2\pi\bar{y}(k)}{T_{r,i}} + \phi_{r,i}\right). \qquad (33)\end{aligned}$$

This model can be seen to represent an LPV system with $\boldsymbol{\sigma}(\bar{y}(k))$ as the scheduling parameter. For simplicity of notation from now on $\delta y(t,k)$ and $\delta f_m(t,k)$ will be replaced by $y(t,k)$ and $f_m(t,k)$, respectively, though it should be remembered that they represent small deviations about the operating point values.

Taking the Laplace transform of (32), for a fixed $\boldsymbol{\sigma}(\bar{y}(k))$, gives:

$$\frac{Y(s,\boldsymbol{\sigma}(\bar{y}(k)))}{F_m(s,k)} = P(s,\boldsymbol{\sigma}(\bar{y}(k))) = \frac{1}{s^2+\alpha s+\boldsymbol{\sigma}(\bar{y}(k))}. \qquad (34)$$

In practice, the ILC algorithm is applied to the reference signal $r(t)$ of the system operating in closed-loop with a PID feedback controller:

$$K(s) = K_p\left(1 + \frac{1}{T_i s} + T_d s\right). \qquad (35)$$

The closed-loop transfer function between the reference signal and the system output is:

$$\frac{Y(s,\boldsymbol{\sigma}(\bar{y}(k)))}{R(s)} = \frac{K(s)P(s,\boldsymbol{\sigma}(\bar{y}(k)))}{1+K(s)P(s,\boldsymbol{\sigma}(\bar{y}(k)))}. \qquad (36)$$

The ILC algorithm is applied in discrete-time. It is therefore necessary to discretise the transfer function (36). This discretisation is done using Euler's second method i.e. by substituting s by $(1-z^{-1})/h$, where h is the sampling period. Additionally the discrete-time system has a two sampling period input delay. We therefore have the discrete-time closed-loop LPV system transfer function given by:

$$\begin{aligned}G(z^{-1},\boldsymbol{\sigma}(\bar{y}(k))) &= \frac{B(z^{-1})}{A(\boldsymbol{\sigma}(\bar{y}(k)),z^{-1})} \\ &= \frac{z^{-2}\left(b_2+b_3z^{-1}+b_4z^{-2}\right)}{a_0\left(\boldsymbol{\sigma}(\bar{y}(k))\right)+a_1\left(\boldsymbol{\sigma}(\bar{y}(k))\right)z^{-1}+a_2z^{-2}+a_3z^{-3}}, \qquad (37)\end{aligned}$$

where

$$\begin{aligned}a_0\left(\boldsymbol{\sigma}(\bar{y}(k))\right) &= a_0^0\lambda_0(\boldsymbol{\sigma}(\bar{y}(k))) + a_0^1\lambda_1(\boldsymbol{\sigma}(\bar{y}(k))) \\ a_1\left(\boldsymbol{\sigma}(\bar{y}(k))\right) &= a_1^0\lambda_0(\boldsymbol{\sigma}(\bar{y}(k))) + a_1^1\lambda_1(\boldsymbol{\sigma}(\bar{y}(k)))\end{aligned}$$

and

$$\lambda_0(\boldsymbol{\sigma}(\bar{y}(k))) = \frac{\overline{\boldsymbol{\sigma}}-\boldsymbol{\sigma}(\bar{y}(k))}{\overline{\boldsymbol{\sigma}}-\underline{\boldsymbol{\sigma}}}, \quad \lambda_1(\boldsymbol{\sigma}(\bar{y}(k))) = \frac{\boldsymbol{\sigma}(\bar{y}(k))-\underline{\boldsymbol{\sigma}}}{\overline{\boldsymbol{\sigma}}-\underline{\boldsymbol{\sigma}}}.$$

It has, therefore, been shown that the LPMSM can be represented by an LPV model for the class of movements considered and thus the developed algorithm is applicable.

B. System Identification

The developed ILC algorithm uses the matrix $\mathcal{G}(\boldsymbol{\sigma}(k))$, as seen in (19). This matrix is formed from the system matrix $\mathbf{G}(\boldsymbol{\sigma}(k))$ and the functions $\lambda_i(\boldsymbol{\sigma}(\bar{y}(k)))$. A model of $G(z^{-1},\boldsymbol{\sigma}(\bar{y}(k)))$ needs to be identified in order to construct the matrix $\mathbf{G}(\boldsymbol{\sigma}(k))$. This identification will be considered first, followed by the identification of $\lambda_i(\boldsymbol{\sigma}(\bar{y}(k)))$.

The LPV model of $G(z^{-1},\boldsymbol{\sigma}(\bar{y}(k)))$ could be found by first identifying LTI models, each with the same structure (order and delay) as (37) at different positions. These models' parameters could then be used to find the parameters of the overall LPV model (37). This method is, however, very time

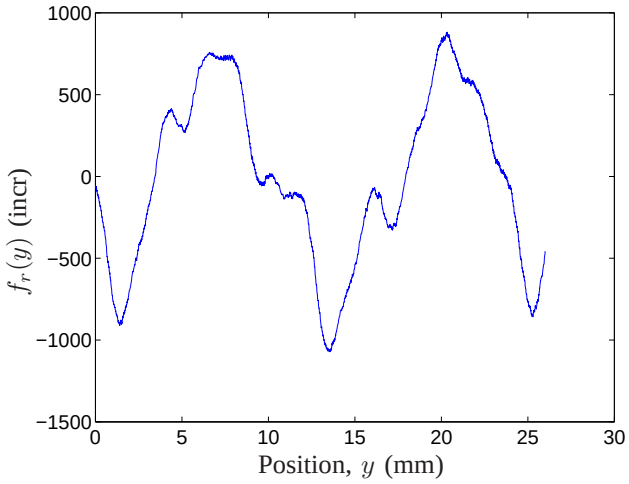

Fig. 5. Measured force ripple ('incr' are the LPMSM's unit of force)

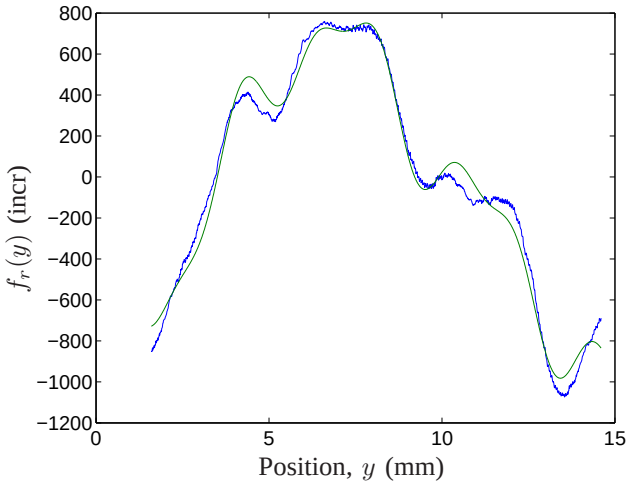

Fig. 6. Measured force ripple (blue) and model estimate (green)

consuming and calls into question the practical usefulness of the ILC method. As was seen in the simulation, the proposed algorithm is robust to a certain amount of system uncertainty. A single LTI model $G(q)$ is therefore identified to be used in the place of the LPV model in (19).

In order to excite the system correctly for the identification of the LTI model, a PRBS signal is chosen as the system's reference signal. It uses a shift register of 10 bits and a divider of 7 giving a signal length of 7161 points. These values are chosen to sufficiently excite the system at low frequencies, and are calculated from an estimate of the system's settling time obtained via a step response test on the real system. The amplitude of the signal is selected large enough to give a good signal-to-noise ratio and reduce the effects of static friction, but small enough to remain in the zone of linearisation and avoid saturation. Two experiments are carried out at an arbitrarily chosen position, one set for parameter estimation and the other for validation. An Output Error model, with the same order and number of input delays as (37), is found to give a good fit to the validation data in simulation.

Next the identification of the functions $\lambda_i(\boldsymbol{\sigma}(\bar{y}(k)))$ is undertaken. These functions are based on the scheduling parameter $\boldsymbol{\sigma}(\bar{y}(k))$. $\boldsymbol{\sigma}(\bar{y}(k))$ cannot be measured directly, only $\bar{y}(k)$. In order to calculate $\boldsymbol{\sigma}(\bar{y}(k))$ it is necessary to estimate the values of $F_{r,i}$, $T_{r,i}$ and $\phi_{r,i}$ for $i = 0, \ldots, n_r - 1$. This can be done using measurements from the motor. The force applied by the feedback controller is measured whilst the motor moves at constant velocity. Since the acceleration is approximately equal to zero and the friction force is assumed constant, any variation in the applied force is to compensate the force ripple. By doing several experiments in the positive and negative directions, removing the offset due to the constant friction force and taking the average we can obtain an estimate of the force ripple, see Fig. 5. The spectrum of the measured force ripple is calculated in order to estimate the periods $T_{r,i}$. Significant peaks occur at spatial frequencies corresponding to periods of 2 mm, 3.7 mm, 8 mm, 13 mm and 21 mm. These values are thus used for $T_{r,i}$ for $i = 0, \ldots, 4$, respectively. With these values the parametric model of the force ripple is:

$$\hat{f}_r(y(t,k)) = \sum_{i=0}^{4} F_{r,i} \sin\left(\frac{2\pi y(t,k)}{T_{r,i}} + \phi_{r,i}\right)$$
$$= \begin{bmatrix} \sin\left(\frac{2\pi y(t,k)}{T_{r,0}}\right) \\ \cos\left(\frac{2\pi y(t,k)}{T_{r,0}}\right) \\ \sin\left(\frac{2\pi y(t,k)}{T_{r,1}}\right) \\ \vdots \\ \cos\left(\frac{2\pi y(t,k)}{T_{r,4}}\right) \end{bmatrix}^T \begin{bmatrix} f^1_{r,0} \\ f^2_{r,0} \\ f^1_{r,1} \\ \vdots \\ f^2_{r,4} \end{bmatrix}.$$

$f^1_{r,0}, f^2_{r,0}, \ldots$ are estimated by a least squares procedure using data from a 13 mm section of the total measured data. This length is chosen as it corresponds to the period of the largest component in the force ripple spectrum. The values of $F_{r,1}, \phi_{r,1}, \ldots$ are then calculated from these estimates. This procedure gives $F_{r,0} = 92.03$ incr, $\phi_{r,0} = 0.74$ rad, $F_{r,1} = 179.07$ incr, $\phi_{r,1} = 1.20$ rad, $F_{r,2} = -253.36$ incr, $\phi_{r,2} = 0.67$ rad, $F_{r,3} = -754.19$ incr, $\phi_{r,3} = 0.62$ rad, $F_{r,4} = 435.79$ incr and $\phi_{r,4} = 0.86$ rad. A comparison of the output of the model formed with these parameters and the force estimated from the measurements is shown in Figure 6. We see that the model fits the measurement estimates reasonably well.

Using this model, the scheduling parameter $\boldsymbol{\sigma}(\bar{y}(k))$ can now be calculated from (33). The maximum and minimum values of $\boldsymbol{\sigma}(\bar{y}(k))$, $\overline{\boldsymbol{\sigma}}$ and $\underline{\boldsymbol{\sigma}}$ are also needed. As the dependence of the derivative of $\boldsymbol{\sigma}(\bar{y}(k))$ is nonlinear with respect to $\bar{y}(k)$, it is not possible to evaluate these values analytically. They are, therefore, estimated by gridding $\boldsymbol{\sigma}(\bar{y}(k))$ over the range considered.

C. Application of proposed algorithm to the real system

With the required model identified, the proposed algorithm is applied to the LPMSM. The desired motion's length is such that $N - m = 24$. Similarly to the simulation, in order for

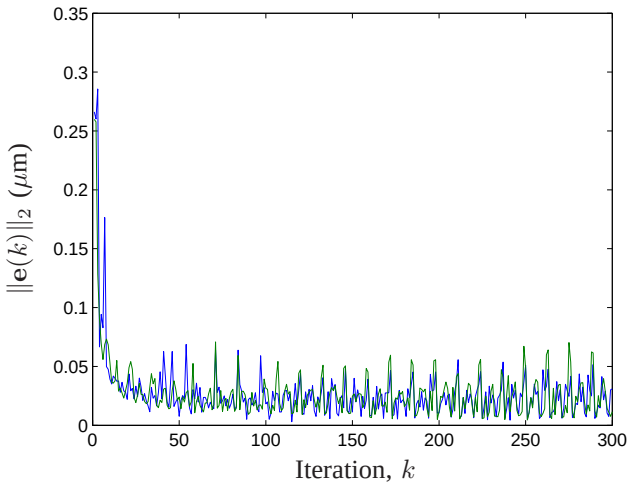

Fig. 7. RMS value of $\mathbf{e}(k)$ obtained experimentally using the proposed algorithm (blue) and the LTI algorithm (green)

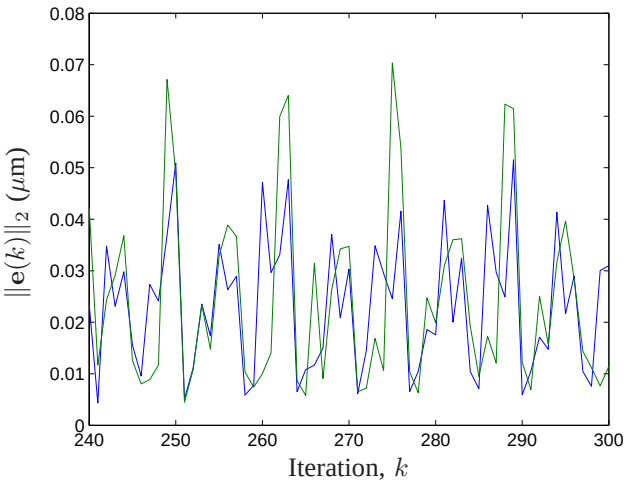

Fig. 8. RMS value of $\mathbf{e}(k)$ obtained experimentally using the proposed algorithm (blue) and the LTI algorithm (green)

the matrix $\mathbf{P}(k+1)$ to exist, the algorithm is not used to calculate the system input until 2 different values of $\bar{y}(k)$ have been visited. Nonetheless the matrix $\mathbf{P}^{-1}(k+1)$ is still ill-conditioned. This is because the identified model $G(q)$ used to produce the matrix $\mathbf{G}$ has an unstable zero at $q = -7$. The $\mathbf{G}$ used to generate $\mathbf{P}^{-1}(k+1)$ thus uses $G(q)$ with this zero stabilised by replacing it with a zero at $q = -1/7$. This stabilisation process is motivated by the fact that it gives a system with a frequency response that has the same magnitude as the identified $G(q)$, though the phase is different.

The movement's starting positions are chosen as a sawtooth waveform that periodically visits 13 equally spaced positions in the 13 mm range considered.

As in the simulation section, the results obtained using the proposed method are compared with those obtained using an LTI ILC algorithm.

The RMS values $\|\mathbf{e}(k)\|_2$ achieved using the LPV and LTI ILC algorithms are shown in Figures 7 and 8. It can be seen that the proposed LPV algorithm considerably reduces the RMS value of the tracking error. The LPV algorithm gives slightly better converged tracking than that achieved with the LTI algorithm; the mean and maximum RMS error values over the last 26 iterations (i.e. the last two periods of starting positions) are 0.0234 μm and 0.0515 μm, respectively, for the LPV algorithm and 0.0254 μm and 0.0703 μm, respectively, for the LTI algorithm. The improvement achieved with the proposed LPV method is not, however, as great as that obtained in the simulation. This is probably because the force ripple affecting the LPMSM used in the experiments is not sufficient to make the system strongly LPV and therefore the system does not truly benefit from the LPV algorithm.

V. Conclusions

An ILC algorithm has been proposed for systems that can be represented by the discrete-time, LPV class of systems. Consistency of the algorithm in the presence of nonstationary stochastic disturbances has been shown when the scheduling parameter trajectory is sufficiently exciting.

The algorithm was tested in simulation and shown to give improved tracking performance over a standard LTI ILC algorithm.

The algorithm has also been applied to a linear, permanent magnet, synchronous motor. Its use lead to a considerable tracking improvement.

Monotonic convergence of a norm of the error signal is of practical interest, and much attention has been given to this issue in LTI ILC. Unfortunately, it seems unlikely that this property can be incorporated into LPV ILC algorithms as the system's dynamics change from one iteration to the next and so, depending on how they change, it is always possible that the error will increase slightly, though the overall trend should be to decrease.

Experimentally the algorithm has been shown to be robust to a certain amount of model uncertainty. Nevertheless, a theoretical result quantifying the acceptable uncertainty should be the subject of future work.

Furthermore, the proposed method is restricted to LPV systems with a dependence on the scheduling parameter only in the denominator of the system's transfer function. It is clear that a more general algorithm that works for systems with scheduling parameter dependence in the numerator as well would be of interest. A different approach to that presented in this paper would be necessary, nonetheless, as the ideal input could not be represented as a linear combination of the ideal inputs at the polytopic vertices.

References

[1] D. Bristow, M. Tharayil, and A. Alleyne, "A survey of iterative learning control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, June 2006.

[2] H. S. Ahn, Y. Chen, and K. Moore, "Iterative learning control: Brief survey and categorization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1099–1121, Nov. 2007.

[3] H. Havlicsek and A. Alleyne, “Nonlinear Control of an Electrohydraulic Injection Molding Machine via Iterative Adaptive Learning,” *IEEE/ASME Transactions on Mechatronics*, vol. 4, no. 3, pp. 312–323, September 1999.
[4] D. Wang and Y. Ye, “Design and Experiments of Anticipatory Learning Control: Frequency-Domain Approach,” *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 3, pp. 305–313, June 2005.
[5] J. Lee and K. Lee, “Iterative learning control applied to batch processes: An overview,” *Control Engineering Practice*, vol. 15, no. 10, pp. 1306–1318, 2007.
[6] A. Tayebi, S. Abdul, M. B. Zaremba, and Y. Ye, “Robust Iterative Learning Control Design: Application to a Robot Manipulator,” *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 5, pp. 608–613, October 2008.
[7] M. Butcher, A. Karimi, and R. Longchamp, “Iterative learning control based on stochastic approximation,” in *17th IFAC World Congress*, Seoul, Korea, July 2008.
[8] K. Moore and Y. Chen, “A separative high-order framework for monotonic convergent iterative learning controller design,” in *IEEE American Control Conference*, Denver, Colorado USA, 2003, pp. 3644–3649 vol.4.
[9] J. Shamma and M. Athans, “Guaranteed properties of gain scheduled control for linear parameter-varying plants,” *Automatica*, vol. 27, no. 3, pp. 559–564, 1991.
[10] S. Sivrioglu and K. Nonami, “Sliding Mode Control With Time-Varying Hyperplane for AMB Systems,” *IEEE/ASME Transactions on Mechatronics*, vol. 3, no. 1, pp. 51–59, March 1998.
[11] M. da Silva, W. Desmet, and H. V. Brussel, “Design of Mechatronic Systems With Conguration-Dependent Dynamics: Simulation and Optimization,” *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 6, pp. 638–646, December 2008.
[12] Z. Li, C. Wen, Y. Soh, and Y. Chen, “Iterative learning control of linear parameterized varying uncertain systems,” in *6th International Conference on Control, Automation, Robotics and Vision*, Singapore, December 2000.
[13] B. G. Dijkstra, “Iterative Learning Control with applications to a wafer stage,” Ph.D. dissertation, Delft University of Technology, Delft, The Netherlands, 2003.
[14] S. Mishra, J. Coaplen, and M. Tomizuka, “Precision positioning of wafer scanners segmented iterative learning control for nonrepetitive disturbances [applications of control],” *IEEE Control Systems Magazine*, vol. 27, no. 4, pp. 20–25, Aug. 2007.
[15] D. Bristow and A. Alleyne, “A high precision motion control system with application to microscale robotic deposition,” *IEEE Transactions on Control Systems Technology*, vol. 14, no. 6, pp. 1008–1020, Nov. 2006.
[16] H. Ding and Z. Xiong, “Motion stages for electronic packaging design and control,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 51–61, Dec. 2006.
[17] D.-I. Kim and S. Kim, “A Iterative Learning Control method with application for CNC machine tools,” *IEEE Transactions on Industry Applications*, vol. 32, no. 1, pp. 66–72, Jan./Feb. 1996.
[18] G. Otten, T. A. de Vries, J. van Amerongen, A. Rankers, and E. Gaal, “Linear Motor Motion Control Using a Learning Feedforward Controller,” *IEEE/ASME Transactions on Mechatronics*, vol. 2, no. 3, pp. 179–187, September 1997.
[19] K. van Berkel, I. Rotariu, and M. Steinbuch, “Cogging Compensating Piecewise Iterative Learning Control with application to a motion system,” in *IEEE American Control Conference*, Denver, Colorado USA, July 2007, pp. 1275–1280.
[20] H. Stark and J. W. Woods, *Probability, Random Processes and Estimation Theory for Engineers*. New Jersey, U.S.A.: Prentice-Hall, 1986.