Iterative Identification of Feedforward Controllers for Iterative Learning Control

Hoday Stearns, Benjamin Fine, Masayoshi Tomizuka

* *University of California, Berkeley, Berkeley, CA 94720 USA (e-mail: hodayx@berkeley.edu).*

Abstract: One the the main disadvantages to the standard iterative learning control (ILC) problem is that the learning filter is static and does not update based on the current iteration information. This means that while the learning control is updating the feedforward control input, the learning controller itself is not updating. In this paper, we present an iteration varying learning filter which is based on identification techniques. Every iteration, this learning filter is computed to be the closed loop system inverse based on a least squares approximation. We show that the iteration varying learning filter successfully reduces the 2-norm error as fast as a static, model-based learning filter and much better than a P-Type learning filter. This results suggests the benefit of extensive preliminary system identification efforts for iterative systems is limited at best.

Keywords: Iterative Learning Control, Identification

1. INTRODUCTION

Iterative Learning Control (ILC) is a popular feedforward control method used on systems undergoing a repetitive batch process. Since the control objectives of each batch are the same, previous runs can conceivably be used to improve the next run. For motion control applications, an iterative process is one where the same reference trajectory is used for tracking control. Here, the objective of ILC is to improve the performance of the system by using tracking error from previous trials to update the feedforward signal in the current trial. ILC was first popularized by Arimoto et al. (1984) and several recent surveys of learning control suggests its recent surge in interest (see Ahn et al. (2007), Bristow et al. (2006), Horowitz (1993)).

The biggest advantage of ILC is that performance of unknown or uncertain systems can be improved with little dependence on the system model. Early methods of ILC focused on proportional (P-) type and proportional plus derivative (PD-) type learning algorithms similar to the feedback controller equivalents. Recently, model based learning algorithms have gained popularity (see deRoover and Bosgra (2000), Gunnarsson and Norrlöf (2001)). These are based on the fact that the ideal learning law is based on the inverse system dynamics. These model based learning laws are limited by how well the system is modeled which begs the following question: is it necessary to spend effort obtaining a better system model before running the iterative control system?

One major disadvantage of standard ILC algorithms is that they use fixed learning laws which are limited to a-priori system knowledge. This means that even though more system knowledge may be gained each iteration, the learning law remains static. One way to address this limitation is to adaptively adjust the learning law after each iteration. In Messner et al. (1990), an adaptive learning law is used based on integral transforms, though in a repetitive control environment. Also, Ashraf et al. (2008) show a gradient based search method is used to adaptively adjust learning gains. Owens and Feng (2003) adaptively adjust a proportional type learning law but either require a plant model or additional experiments for the update law.

In this paper, we combine two issues, system modeling and iteration varying learning filters, with application to a wafer stage system (see Mishra et al. (2007)). ILC performance is important in semiconductor manufacturing systems where the cost of time not manufacturing has a substantial dollar amount. As shown in Owens and Feng (2003), single parameter adaptation can result in a slow ILC convergence rate. Rather than adaptively updating the learning algorithm, we present here a model based ILC algorithm which is updated each iteration based on system identification techniques. We choose a least squares system identification approach (see Ljung (1987)) over gradient based search methods as shown in Ashraf et al. (2008). Rather than directly modeling the system, this approach attempts to identify a fictitious feedforward controller which results in the same output as the ILC algorithm. This controller, based on the plant inversion, is then used to update the ILC learning filter. A recursive method is proposed to identify the plant model which minimizes the difference between the output of the fictitious feedforward controller and ILC algorithm.

The outline of this paper is as follows. In Section 2, we briefly introduce ILC. In Section 3, we present the experimental wafer stage. Section 4 presents the iteration varying ILC algorithm based on the fictitious feedforward controller. The effectiveness of the proposed method is verified in Section 4 with simulation and in Section 5 with experimental results.

2. ILC INTRODUCTION

The ILC problem is formulated as follows. Consider a Single Input Single Output (SISO), discrete time, linear time invariant (LTI) system $G(q)$

$$y_j(k) = G(q)u_j^{ILC}(k) + d(k) \tag{1}$$

where j denotes the iteration (trial) number. The disturbance $d(k)$ is assumed to be iteration invariant. The objective of ILC is to choose the control input $u_j^{ILC}(k)$ to minimize the tracking error

$$e_j(k) = y_d(k) - y_j(k). \tag{2}$$

Based on a common ILC update law (see Bristow et al. (2006)), we introduce

$$u_{j+1}^{ILC}(k) = Q(q)(u_j^{ILC}(k) + L_j(q)e_j(k + t_d)) \tag{3}$$

where $L_j(q)$ is the iteration varying learning filter and $Q(q)$ is a low-pass filter. Because this control algorithm is computed off-line, the system delay t_d can be accounted for. Thus, the controller design problem is to find $L_j(q)$ and $Q(q)$ which minimizes $e_j(k)$. In this paper, we present a model based learning filter based on the system being learned, $L_j(q) = G^{-1}(q)$. The system $G(q)$ depends on the location of the ILC injection point with respect to the feedback controller will be further defined in section 4. Convergence proofs for iteration varying learning filters can be found in Norrlöf and Gunnarsson (2002).

3. EXPERIMENTAL SETUP

The proposed algorithm will be implemented on a prototype of an industrial wafer stage. A picture of the wafer stage prototype is shown in Fig. 1. A wafer stage is part of a device used for manufacturing integrated circuits in the process known as photolithography. In photolithography, a light source is used to transfer a pattern from a mask ("reticle") onto a chemical photoresist covering the silicon wafer. The positioning accuracy of the stage is very important because higher precision in the positioning will enable smaller feature sizes to be written. Additionally, the stage must be able to position with high accuracy while moving at high speeds to produce many chips in a short amount of time. The operation of a wafer stage is a repetitive step and scan motion. The repetitiveness of the task can be exploited to further increase tracking accuracy by learning from past iterations of the same task.

The stage is actuated by a linear permanent magnet motor. The permanent magnet rail is attached to a countermass which is moveable with respect to the base and is actuated by another linear motor coil. The purpose of the countermass setup is to prevent vibrations from being transmitted to the base when the wafer stage accelerates. The position of the stage is measured by a laser interferometer. The stage is controlled by a LabVIEW realtime system with FPGA with sample time of 0.0004 s.

The wafer stage is modeled with the transfer function

$$P(s) = \frac{c}{ms^2 + bs} \tag{4}$$

where m is the stage mass, b is viscous friction coefficient, and c is hardware constant. In this transfer function, the input u is considered to be the voltage sent to the amplifier and the output y is the linear displacement (in meters) of the stage relative to the base. We considered the effects of non-viscous friction (such as Coulomb friction) to be negligible since the stage is sliding on an air bearing.

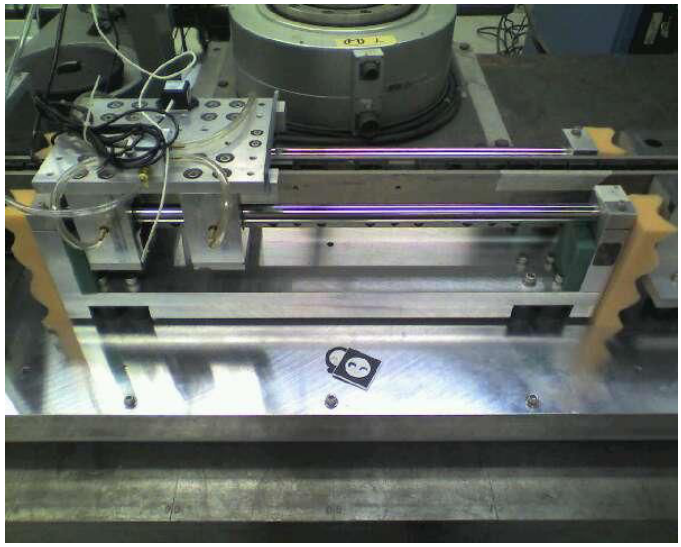

Fig. 1. Wafer stage prototype experimental setup.

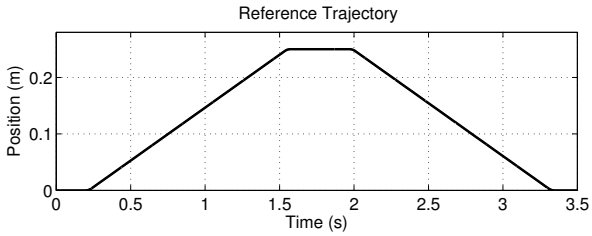

Fig. 2. Reference trajectory for wafer stage

For the duration of the ILC updates, the wafer stage is in close-loop operation. The feedback controller is a tuned PID controller which is held fixed through all iterations. The reference trajectory during tuning is shown in Fig. 2.

4. ALGORITHM

In this section, we propose a new iteration varying ILC algorithm which is based on estimating an equivalent feedforward controller to the ILC control input. This feedforward controller is then used to update the model based learning filter. The block diagram is shown in Figure 3 shows the learning filter being injected in parallel to the feedback controller $C(q)$. The closed loop system output is given by

$$y = \frac{P(q)}{1 + P(q)C(q)}(u^{ILC}(k) + d(k)) + \frac{P(q)C(q)}{1 + P(q)C(q)}r(k) \tag{5}$$

where y is the measured output, u^{ILC} is the ILC input, d is an iteration-invariant disturbance, r is an iteration-invariant reference trajectory, q is a time domain shift operator, P is the plant transfer function operator and k is the discrete time domain index.

By defining $G_1 := \frac{P(q)}{1+P(q)C(q)}$ and $G_2 := \frac{P(q)C(q)}{1+P(q)C(q)}$ and substituting into eqn. (5), the output equation can be expressed

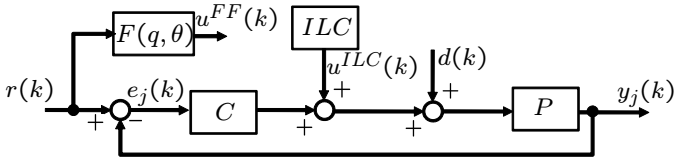

Fig. 3. Block diagram

$$y_j(k) = G_1 u_j^{ILC}(k) + G_1 d(k) + G_2 r(k). \tag{6}$$

The plant estimator is shown in detail in fig. 4. The plant is estimated based on $r(k)$ and $e_j(k)$ after each iteration by using a least-squares optimization of a cost function. The estimate is then used to update the learning filter for the next iteration learning control. The update in the inverse plant model results in an updated learning filter in each iteration.

We define an ILC law the same as in Eq. (3). A well-known result from ILC theory states that in the absence of noise, the tracking error will converge to zero in one iteration if learning filter $L(q)$ is set equal to the inverse of the $G_1(q)$ (see Norrlöf and Gunnarsson (2002)), or

$$\begin{aligned} L &= G_1^{-1}(q) \\ &= \frac{1 + P(q)C(q)}{P(q)} \\ &= P^{-1}(q) + C(q). \end{aligned} \tag{7}$$

The plant inverse model will be identified through estimating a hypothetical feedforward controller with similar output as u^{ILC}, as explained in the following. Consider a hypothetical fixed-structure feedforward controller $F(q,\theta)$ parameterized by unknown parameters θ. This ideal controller shown in fig. (3) is

$$F(q,\theta) = P(q)^{-1} \tag{8}$$

and results in perfect tracking if the plant is perfectly known. So, identifying the plant inverse can be thought of as identifying the ideal feedforward controller. The feedforward controller will be updated in each iteration by finding the values of parameters θ such as to minimize a chosen cost function. The cost function we wish to minimize is the difference between the next iteration ILC input $u_{j+1}^{ILC}(k)$ and the ideal feedforward control signal $u_{j+1}^{FF}(k)$

$$J(q) = \left|\left| u_{j+1}^{ILC}(k) - u_{j+1}^{FF}(k) \right|\right|_2 \tag{9}$$

where

$$u_j^{FF}(\theta_j) = F(q,\theta_j) r(k) \tag{10}$$

is the output of the feedforward controller. In other words, the objective is to determine a feedforward controller $F(q,\theta)$ whose output matches the ILC input $u_{j+1}^{ILC}(k)$ as closely as possible. Unfortunately, the ideal feedforward controller is not available. Thus, we utilize eqn. (9) to obtain the feedforward control $F(q,\theta)$ such that the feedforward output matches the a-priori ILC input

$$u_{j+1}^{ILC\circ} = Q(q)\left(u_j^{ILC} + L_j(q) e_j(k)\right). \tag{11}$$

The a-priori ILC learning filter is set to the previous iteration plant model: $L_j(q) = \hat{P}_j^{-1} + C$. Now we can minimize the cost function

$$J = \left|\left| u_{j+1}^{ILC\circ} - F(q,\theta_{j+1}) r(k) \right|\right|_2 \tag{12}$$

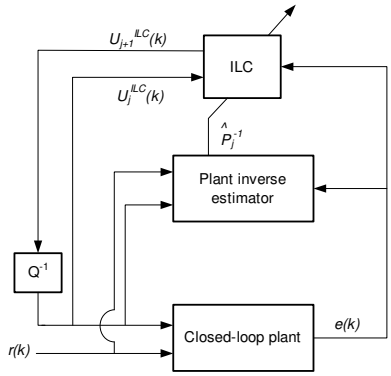

Fig. 4. ILC Scheme

with respect to θ_{j+1} to compute the a-posteriori estimate $L_{j+1} = \hat{P}_{j+1}^{-1} + C$. Then, the a-posteriori ILC input can now be computed

$$u_{j+1}^{ILC} = Q(q)\left(u_j^{ILC} + L_{j+1}(q) e_j(k)\right). \tag{13}$$

We can recursively compute the ILC algorithm and the minimization until $\hat{P}_j^{-1}$ has converged satisfactorily. This algorithm is summarized below.

(1) Initialize $\hat{P}_j^{-1}$, $u_j^{ILC\circ}$
(2) Run iteration j, collect data e_j.
(3) Estimate $\hat{P}_{j+1}^{-1}$
  (a) Compute future ILC input, $u_{j+1}^{ILC\circ}(k)$ using $\hat{P}_j^{-1}$.
  (b) Identify $\hat{P}_{j+1}^{-1}$ by minimizing eqn. (12).
  (c) Recompute $u_{j+1}^{ILC}(k)$ using the updated plant estimation.
  (d) Reiterate until $\hat{P}_{j+1}$ has converged.
(4) Return to 2

The objective of this algorithm is to find a hypothetical feedforward controller with a similar output as u^{ILC}. Part 4 of the algorithm above is now explained in detail.

The minimization problem in Eq. (9) can be approximated as a linear least-squares problem if it is reformulated as shown below. Let

$$(u_j^{ILC})^T = \left[\, u_j^{ILC}(1)\ u_j^{ILC}(2)\ \ldots\ u_j^{ILC}(N) \,\right]$$

Let the FF controller F be given as

$$F(q,\theta) := \frac{B(q^{-1})}{A(q^{-1})} = \frac{b^0 + b^1 q^{-1} + \ldots + b^n q^{-n}}{1 + a^1 q^{-1} + \ldots + a^m q^{-m}} \tag{14}$$

where F is parameterized by parameters θ written as

$$\theta_j^T := \left[\, a_j^1\ \ldots\ a_j^m\ b_j^0\ b_j^1\ \ldots\ b_j^n \,\right]. \tag{15}$$

Then the ILC input can be written as the sum of the feedforward controller output plus an error term

$$u_j^{ILC}(k) = F(q,\theta_j) r(k) + \varepsilon(k)$$

which becomes

$$u_j^{ILC}(k) = \frac{B(q^{-1})}{A(q^{-1})} r(k) + \varepsilon(k).$$

or

$$A(q^{-1}) u_j^{ILC}(k) = B(q^{-1}) r(k) + A(q^{-1}) \varepsilon(k).$$

We define an auto-regressive model for the purpose of feedforward controller identification

$$u_j^{ILC}(k) = -a_j^1 u_j^{ILC}(k-1) - a_j^2 u_j^{ILC}(k-2) - \dots$$
$$\dots - a_j^m u_j^{ILC}(k-m) + b_j^0 r(k) + \dots$$
$$\dots + b_j^n r(k-n) + A(q^{-1})\varepsilon_j(k)$$

Then u_j^{ILC} can be written

$$u_j^{ILC} = X_j\theta_j + A(q^{-1})\varepsilon_j$$

where

$$X_j := \left[-(u_j^{ILC})\ q^{-1}(u_j^{ILC})\ \dots \right.$$
$$\left. \dots\ q^{-m}(u_j^{ILC})\ q^{-1}(r)\ \dots\ q^{-n}(r) \right] \quad (16)$$

Instead of considering the minimization of norm of ε_j, if we consider the minimization of filtered prediction error $A(q^{-1})\varepsilon_j$, we will have the following cost-function:

$$\min \left\| u_{j+1}^{ILC} - X\theta_{j+1} \right\|_2$$

which is a standard linear-least squares problem.

Then the solution is given by

$$(\theta_{j+1}^*)^T = (X^T X)^{-1} X^T u_j^{ILC}. \quad (17)$$

In this way, as the ILC input converges, the inverse plant model becomes closer to the true plant. The inverse plant model $\hat{P}_{j+1}$ is set to

$$\hat{P}_{j+1}^{-1} = F(q, \theta_{j+1}^*) \quad (18)$$

and used to compute L_{j+1}.

In the algorithm just proposed, the identification of the plant inverse is done in a recursive manner to minimize eqn. (12). The process is repeated until the update in plant model becomes satisfactorily small. However, if we assume that the recursive identification of $\hat{P}_j^{-1}$ converges to some model, then we can solve directly for the limit. Starting from Eq. 12, we have

$$J = \left|\left| Q(u_j + (\hat{P}_j^{-1} + C)e_j(k) - \hat{P}_{j+1}^{-1} r(k) \right|\right|_2. \quad (19)$$

Substituting $\hat{P}_{\text{inf}}^{-1}$ for $\hat{P}_j^{-1}$and $\hat{P}_{j+1}^{-1}$, we have

$$J = \left|\left| Q(u_j + (\hat{P}_{\text{inf}}^{-1} + C)e_j(k) - \hat{P}_{\text{inf}}^{-1} r(k) \right|\right|_2. \quad (20)$$

Then the cost function becomes

$$J = \left|\left| Q(u_j + Ce_j(k)) + \hat{P}_{\text{inf}}^{-1}(Qe_j(k) - r(k)) \right|\right|_2. \quad (21)$$

This is also a linear least squares problem that can be solved to find $\hat{P}_{\text{inf}}^{-1}$.

In summary, at each iteration, a new hypothetical feedforward controller is computed to match the ILC input of the next iteration. The feedforward controller gives a model of the plant inverse which is a plant model for the current iteration. The learning filter is updated based on the most recent plant model and the ILC input for the next iteration is recalculated. This process is iterated until both the ILC control input and the plant model converge.

5. SIMULATION

We apply the proposed ILC scheme to improve trajectory tracking accuracy for a wafer stage in simulation and in experiment. The model structure of the wafer stage is given in Eqn. 4, so we have chosen a feedforward controller structure

$$F^{cont}(s) = \theta^2 s^2 + \theta^1 s. \quad (22)$$

Since the controller must be implemented in discrete-time, we can chose to approximate the continuous-time controller with the forward-difference discretization

$$F^{disc}(z) = \theta^2 \left(\frac{1 - z^{-1}}{T_s} \right)^2 + \theta^1 \left(\frac{1 - z^{-1}}{T_s} \right).$$

However, an easier way to implement this learning controller is in a fixed-step-size simulation of the zero-order-hold discretization of the continuous plant on discrete data-sequence $e_k, k = 1 \dots K$. The filter is implemented with the MATLAB command `lsim`. The use of the discrete-time approximation is justified because the filters are most different in the high-frequency range but we have used a sufficiently high sampling frequency. Also, the bandwidth of the plant and Q-filter are low in comparison to the Nyquist frequency which means the high frequency inaccuracies will be insignificant.

In both simulation and experiment, a P-type ILC law was used, in which the ILC signal was injected at the plant input point. The ILC-delay is chosen to be 2. The Q-filter is constant throughout the iterations and is a zero-phase low-pass filter with cutoff frequency 140 Hz. The initial learning filter L_0 is chosen to be $0.6C$ where C is the feedback controller (so that the ILC law is equivalent a constant-gain P-type ILC injected at reference point). The performance of the proposed algorithm is compared to the case of an ILC with non-varying learning filter set to this same initial L throughout all the iterations.

The performance of the proposed scheme is first verified in simulation. The results of simulation are shown in Figs. 5 through 8. Figure 5 shows that this scheme reduced error due to both changes in the reference setpoint (during the acceleration phase) and force ripple disturbances (during the constant scan phase). The ILC input signal after 5 iterations is shown in Figs. 6. The ILC signal amplitude appears to be within reasonable bounds. Next, Fig. 7 shows that the tuned parameters in the feedforward controller converged. Furthermore, the converged values are close to the values expected if θ^1 and θ^2 were to be designed such that the feedforward controller corresponds to the inverse of the plant model used in the simulation, $\theta^1 = 0.6000$ and $\theta^2 = 0.4417$. Finally, Fig. 8 shows that the new tuning scheme results in faster convergence of the L_2-norm of error than a standard ILC scheme in which the learning matrix L is not updated with each iteration. Therefore, it is seen that this scheme successfully applies plant identification to improving ILC convergence speed.

In Section 4 it was explained that the plant model may be identified recursively or the limit directly identified. Fig. 9 shows a comparison of the ILC convergence rates for both methods. It is seen that the direct identification has the fastest convergence rate, but that even doing one iteration achieves a convergence rate that is almost as fast. The recursive method is used in this paper because the computation of the solution for θ^* is simpler.

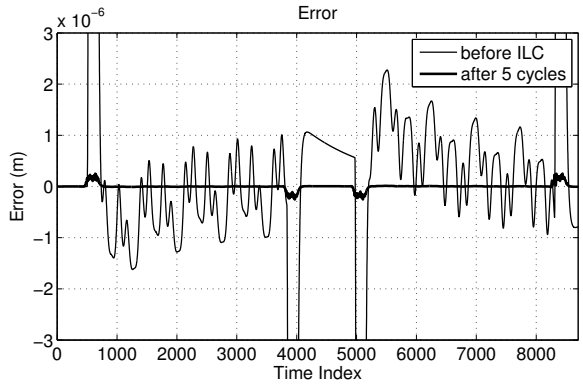

Fig. 5. Reduction of error after ILC tuning in simulation

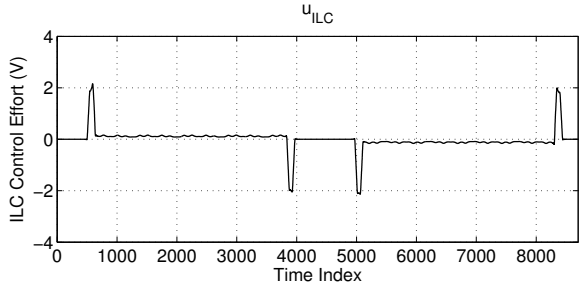

Fig. 6. ILC feedforward signal after 5 iterations of tuning in simulation

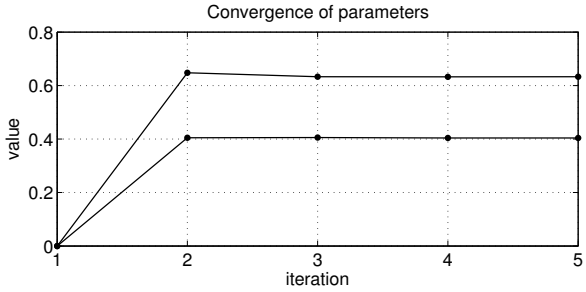

Fig. 7. Convergence of parameters θ_j^* in simulation

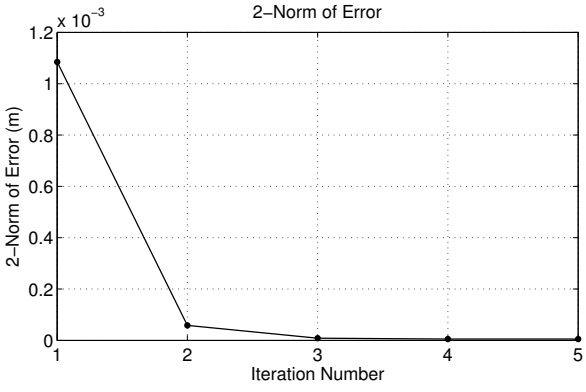

Fig. 8. Reduction of L_2 norm of error in simulation

6. EXPERIMENT

The previous section's results were experimentally verified on the wafer stage system. The results are shown in Figs. 11 through 13. In Fig. 11, we can see that the norm of the error was greatly reduced, and especially the effects of

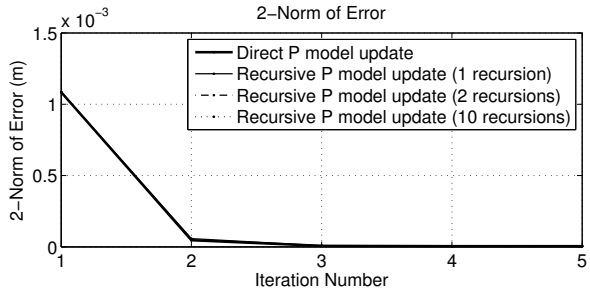

Fig. 9. Reduction of L_2 norm of error in simulation. Comparison of different methods of finding plant inverse.

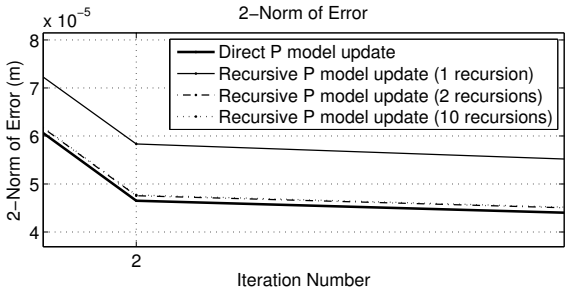

Fig. 10. Magnification of above graph.

force ripple and error caused by acceleration are reduced. The peak error was also greatly reduced. Fig. 12 shows the ILC input after 5 iterations of tuning. Fig. 13 shows that the feedforward controller parameters appear to converge. The converged values appear to be close to those identified in previous experimental tuning of feedforward control through iterative feedforward tuning (IFT), which were $\theta_2 = 0.47$ and $\theta_1 = 0.52$ (see Stearns et al. (2008)).

Fig. 14 is a comparison of the convergence rate of the L_2 norm of error over the ILC iterations for different ILC schemes. We compared the proposed scheme (with iteration-varying learning filter) with three other ILC update schemes of iteration-invariant learning filters: P-type ILC with a constant learning gain (0.6, the same filter used for the initial L in our algorithm), ILC with a learning filter based on plant-model inverse where the model was identified through IFT, and ILC with a learning filter based on plant-model inverse where the model was identified by direct measurement of the physical parameters. The figure shows that the proposed scheme (second line) has significant improvement of error convergence rate over a constant fixed-gain ILC scheme (first line), which is one common choise of L in cases where the plant model is not known well. The proposed scheme achieves performance that is almost identical to model-based schemes (third and fourth lines) where the model is known previously. However, it must be taken into consideration that in order to identify these models in the first place, many experiments or IFT iterations were necessary. The proposed scheme achieves similar performance without the need for the time-consuming preliminary identification step.

7. CONCLUSION

A new algorithm for implementing ILC was proposed. This new method takes advantage of information gained from multiple runs of a repetitive process to identify a

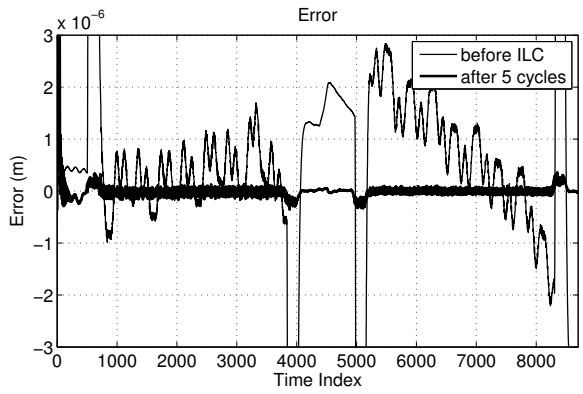

Fig. 11. Reduction of error after ILC tuning in experiment

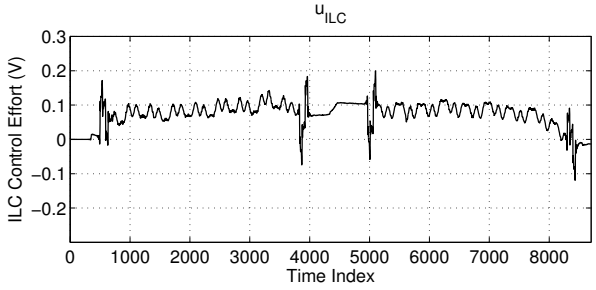

Fig. 12. ILC feedforward signal after 5 iterations of tuning in experiment

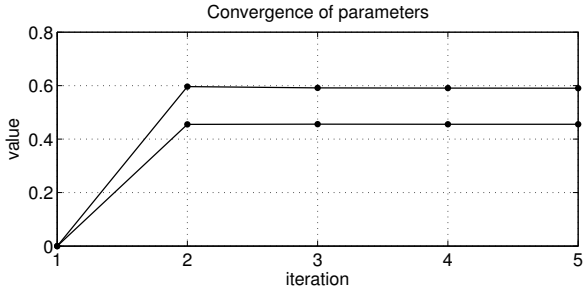

Fig. 13. Convergence of parameters θ_j^* in experiment

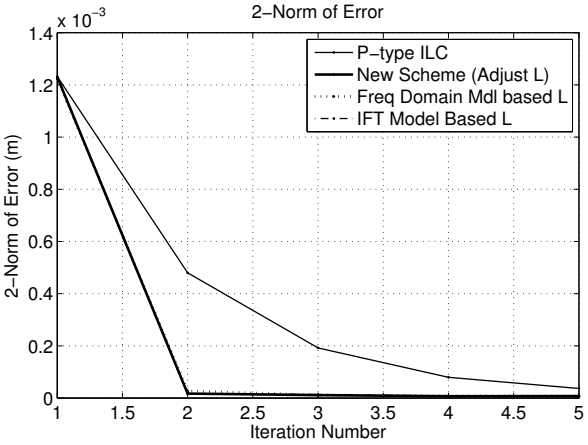

Fig. 14. Reduction of L_2 norm of Error - Comparison with other ILC schemes in experiment

plant inverse model and uses this model in the learning filter of the ILC scheme to further speed up convergence. Experiments verified that the new method results in faster convergence of error norm than ILC with fixed learning gain.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support of Nikon Research Corporation of North America, National Instruments,and the University of California Discovery Grant.

REFERENCES

Ahn, H.S., Chen, Y., and Moore, K.L. (2007). Iterative learning control: Brief survey and categorization. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6), 1099–1121.

Arimoto, S., Kawamura, S., and Miyazaki, F. (1984). Bettering operation of robots by learning. *Journal of Robotic Systems*, 1(2), 123–140. 10.1002/rob.4620010203.

Ashraf, S., Muhammad, E., and Al-Habaibeh, A. (2008). Self-learning control systems using identification-based adaptive iterative learning controller. *Proceedings of the I MECH E Part C: Journal of Mechanical Engineering Science*, 222, 1177–1187.

Bristow, D.A., Tharayil, M., and Alleyne, A.G. (2006). A survey of iterative learning control. *Control Systems Magazine, IEEE*, 26(3), 96–114.

deRoover, D. and Bosgra, O.H. (2000). Synthesis of robust multivariable iterative learning controllers with application to a wafer stage motion system. *International Journal of Control*, 73(10), 968–979. Doi:10.1080/002071700405923.

Gunnarsson, S. and Norrlöf, M. (2001). On the design of ilc algorithms using optimization. *Automatica*, 37(12), 2011–2016.

Horowitz, R. (1993). Learning control of robot manipulators. *ASME Journal of Dynamic Systems, Measurement and Control*, 115, 402–411.

Ljung, L. (1987). *System identification : theory for the user*. Prentice-Hall information and system sciences series. Prentice-Hall, Englewood Cliffs, NJ.

Messner, W., Horowitz, R., Kao, W.W., and Boals, M. (1990). A new adaptive learning rule. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, 1522–1527 vol.3.

Mishra, S., Coaplen, J., and Tomizuka, M. (2007). Precision positioning of wafer scanners segmented iterative learning control for nonrepetitive disturbances. *Control Systems Magazine, IEEE*, 27(4), 20–25.

Norrlöf, M. and Gunnarsson, S. (2002). Time and frequency domain convergence properties in iterative learning control. *International Journal of Control*, 75, 1114–1126.

Owens, D.H. and Feng, K. (2003). Parameter optimization in iterative learning control. *International Journal of Control*, 76, 1059–1069.

Stearns, H., Mishra, S., and Tomizuka, M. (2008). Iterative tuning of feedforward controller with force ripple compensation for wafer stage. In *Proceedings of 10th International Conference on Advanced Motion Control*. Trento, Italy.