Iterative learning control (ILC) is based on the notion that the performance of a system that executes the same task multiple times can be improved by learning from previous executions (trials, iterations, passes). For instance, a basketball player shooting a free throw from a fixed position can improve his or her ability to score by practicing the shot repeatedly. During each shot, the basketball player observes the trajectory of the ball and consciously plans an alteration in the shooting motion for the next attempt. As the player continues to practice, the correct motion is learned and becomes ingrained into the muscle memory so that the shooting accuracy is iteratively improved. The converged muscle motion profile is an open-loop control generated through repetition and learning. This type of learned open-loop control strategy is the essence of ILC.

We consider learning controllers for systems that perform the same operation repeatedly and under the same operating conditions. For such systems, a nonlearning controller yields the same tracking error on each pass. Although error signals from previous iterations are information rich, they are unused by a nonlearning controller. The objective of ILC is to improve performance by incorporating error information into the control for subsequent iterations. In doing so, high performance can be achieved with low transient tracking error despite large model uncertainty and repeating disturbances.

DOUGLAS A. BRISTOW, MARINA THARAYIL, and ANDREW G. ALLEYNE

ILC differs from other learning-type control strategies, such as adaptive control, neural networks, and repetitive control (RC). Adaptive control strategies modify the controller, which is a system, whereas ILC modifies the control input, which is a signal [1]. Additionally, adaptive controllers typically do not take advantage of the information contained in repetitive command signals. Similarly, neural network learning involves the modification of controller parameters rather than a control signal; in this case, large networks of nonlinear neurons are modified. These large networks require extensive training data, and fast convergence may be difficult to

A Survey of Iterative Learning Control

A LEARNING-BASED METHOD FOR HIGH-PERFORMANCE TRACKING CONTROL

1066-033X/06/$20.00©2006IEEE

guarantee [2], whereas ILC usually converges adequately in just a few iterations.

ILC is perhaps most similar to RC [3] except that RC is intended for continuous operation, whereas ILC is intended for discontinuous operation. For example, an ILC application might be to control a robot that performs a task, returns to its home position, and comes to a rest before repeating the task. On the other hand, an RC application might be to control a hard disk drive's read/write head, in which each iteration is a full rotation of the disk, and the next iteration immediately follows the current iteration. The difference between RC and ILC is the setting of the initial conditions for each trial [4]. In ILC, the initial conditions are set to the same value on each trial. In RC, the initial conditions are set to the final conditions of the previous trial. The difference in initial-condition resetting leads to different analysis techniques and results [4].

Traditionally, the focus of ILC has been on improving the performance of systems that execute a single, repeated operation. This focus includes many practical industrial systems in manufacturing, robotics, and chemical processing, where mass production on an assembly line entails repetition. ILC has been successfully applied to industrial robots [5]–[9], computer numerical control (CNC) machine tools [10], wafer stage motion systems [11], injection-molding machines [12], [13], aluminum extruders [14], cold rolling mills [15], induction motors [16], chain conveyor systems [17], camless engine valves [18], autonomous vehicles [19], antilock braking [20], rapid thermal processing [21], [22], and semibatch chemical reactors [23].

ILC has also found application to systems that do not have identical repetition. For instance, in [24] an underwater robot uses similar motions in all of its tasks but with different task-dependent speeds. These motions are equalized by a time-scale transformation, and a single ILC is employed for all motions. ILC can also serve as a training mechanism for open-loop control. This technique is used in [25] for fast-indexed motion control of low-cost, highly nonlinear actuators. As part of an identification procedure, ILC is used in [26] to obtain the aerodynamic drag coefficient for a projectile. Finally, [27] proposes the use of ILC to develop high-peak power microwave tubes.

The basic ideas of ILC can be found in a U.S. patent [28] filed in 1967 as well as the 1978 journal publication [29] written in Japanese. However, these ideas lay dormant until a series of articles in 1984 [5], [30]–[32] sparked widespread interest. Since then, the number of publications on ILC has been growing rapidly, including two special issues [33], [34], several books [1], [35]–[37], and two surveys [38], [39], although these comparatively early surveys capture only a fraction of the results available today.

As illustrated in "Iterative Learning Control Versus Good Feedback and Feedforward Design," ILC has clear advantages for certain classes of problems but is not applicable to every control scenario. The goal of the present article is to provide a tutorial that gives a complete picture of the benefits, limitations, and open problems of ILC. This presentation is intended to provide the practicing engineer with the ability to design and analyze a simple ILC as well as provide the reader with sufficient background and understanding to enter the field. As such, the primary, but not exclusive, focus of this survey is on single-input, single-output (SISO) discrete-time linear systems. ILC results for this class of systems are accessible without extensive mathematical definitions and derivations. Additionally, ILC designs using discrete-time

Iterative Learning Control Versus Good Feedback and Feedforward Design

The goal of ILC is to generate a feedforward control that tracks a specific reference or rejects a repeating disturbance. ILC has several advantages over a well-designed feedback and feedforward controller. Foremost is that a feedback controller reacts to inputs and disturbances and, therefore, always has a lag in transient tracking. Feedforward control can eliminate this lag, but only for known or measurable signals, such as the reference, and typically not for disturbances. ILC is anticipatory and can compensate for exogenous signals, such as repeating disturbances, in advance by learning from previous iterations. ILC does not require that the exogenous signals (references or disturbances) be known or measured, only that these signals repeat from iteration to iteration.

While a feedback controller can accommodate variations or uncertainties in the system model, a feedforward controller performs well only to the extent that the system is accurately known. Friction, unmodeled nonlinear behavior, and disturbances can limit the effectiveness of feedforward control. Because ILC generates its open-loop control through practice (feedback in the iteration domain), this high-performance control is also highly robust to system uncertainties. Indeed, ILC is frequently designed assuming linear models and applied to systems with nonlinearities yielding low tracking errors, often on the order of the system resolution.

ILC cannot provide perfect tracking in every situation, however. Most notably, noise and nonrepeating disturbances hinder ILC performance. As with feedback control, observers can be used to limit noise sensitivity, although only to the extent to which the plant is known. However, unlike feedback control, the iteration-to-iteration learning of ILC provides opportunities for advanced filtering and signal processing. For instance, zero-phase filtering [43], which is noncausal, allows for high-frequency attenuation without introducing lag. Note that these operations help to alleviate the sensitivity of ILC to noise and nonrepeating disturbances. To reject nonrepeating disturbances, a feedback controller used in combination with the ILC is the best approach.

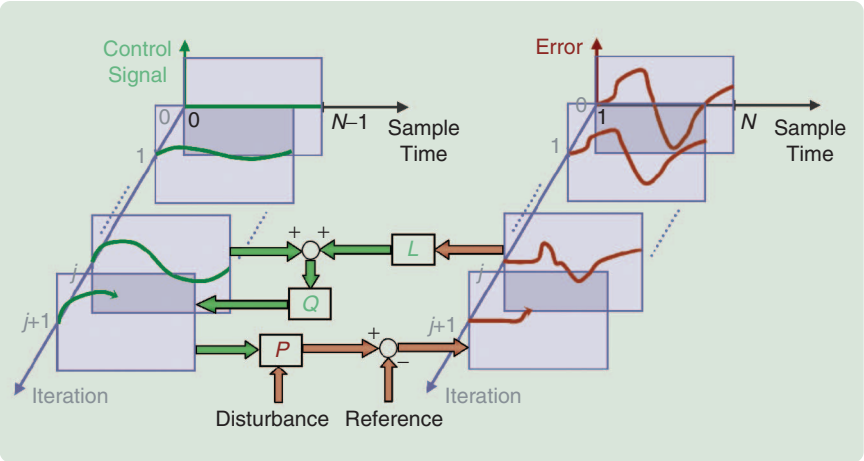

FIGURE 1 A two-dimensional, first-order ILC system. At the end of each iteration, the error is filtered through L, added to the previous control, and filtered again through Q. This updated open-loop control is applied to the plant in the next iteration.

linearizations of nonlinear systems often yield good results when applied to the nonlinear systems [10], [22], [40]–[43].

ITERATIVE LEARNING CONTROL OVERVIEW

Linear Iterative Learning Control System Description

Consider the discrete-time, linear time-invariant (LTI), SISO system

$$y_j(k) = P(\mathrm{q})u_j(k) + d(k), \tag{1}$$

where k is the time index, j is the iteration index, q is the forward time-shift operator $\mathrm{q}x(k) \equiv x(k+1)$, y_j is the output, u_j is the control input, and d is an exogenous signal that repeats each iteration. The plant $P(\mathrm{q})$ is a proper rational function of q and has a delay, or equivalently relative degree, of m. We assume that $P(\mathrm{q})$ is asymptotically stable. When $P(\mathrm{q})$ is not asymptotically stable, it can be stabilized with a feedback controller, and the ILC can be applied to the closed-loop system.

Next consider the N-sample sequence of inputs and outputs

$$u_j(k), k \in \{0, 1, \ldots, N-1\},$$
$$y_j(k), k \in \{m, m+1, \ldots, N+m-1\},$$
$$d(k), k \in \{m, m+1, \ldots, N+m-1\},$$

and the desired system output

$$y_d(k), k \in \{m, m+1, \ldots, N+m-1\}.$$

The performance or error signal is defined by $e_j(k) = y_d(k) - y_j(k)$. In practice, the time duration N of the trial is always finite, although sometimes it is useful for analysis and design to consider an infinite time duration of the trials [44]. In this work we use $N = \infty$ to denote trials with an infinite time duration. The iteration dimension indexed by j is usually considered infinite with $j \in \{0, 1, 2, \ldots\}$. For simplicity in this article, unless stated otherwise, the plant delay is assumed to be $m = 1$.

Discrete time is the natural domain for ILC because ILC explicitly requires the storage of past-iteration data, which is typically sampled. System (1) is sufficiently general to capture IIR [11] and FIR [45] plants $P(\mathrm{q})$. Repeating disturbances [44], repeated nonzero initial conditions [4], and systems augmented with feedback and feedforward control [44] can be captured in $d(k)$. For instance, to incorporate the effect of repeated nonzero initial conditions, consider the system

$$\mathbf{x}_j(k+1) = \mathbf{A}\mathbf{x}_j(k) + \mathbf{B}u_j(k) \tag{2}$$
$$y_j(k) = \mathbf{C}\mathbf{x}_j(k), \tag{3}$$

with $\mathbf{x}_j(0) = \mathbf{x}_0$ for all j. This state-space system is equivalent to

$$y_j(k) = \underbrace{\mathbf{C}(\mathrm{q}\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}}_{P(\mathrm{q})}u_j(k) + \underbrace{\mathbf{C}\mathbf{A}^k\mathbf{x}_0}_{d(k)}.$$

Here, the signal $d(k)$ is the free response of the system to the initial condition $\mathbf{x}_0$.

A widely used ILC learning algorithm [1], [35], [38] is

$$u_{j+1}(k) = Q(\mathrm{q})[u_j(k) + L(\mathrm{q})e_j(k+1)], \tag{4}$$

where the LTI dynamics $Q(\mathrm{q})$ and $L(\mathrm{q})$ are defined as the Q-filter and learning function, respectively. The two-dimensional (2-D) ILC system with plant dynamics (1) and learning dynamics (4) is shown in Figure 1.

General Iterative Learning Control Algorithms

There are several possible variations to the learning algorithm (4). Some researchers consider algorithms with linear time-varying (LTV) functions [42], [46], [47], nonlinear functions [37], and iteration-varying functions [1], [48]. Additionally, the order of the algorithm, that is, the number N_0 of previous iterations of $\mathbf{u}_i$ and $\mathbf{e}_i$, $i \in \{j - N_0 + 1, \ldots, j\}$, used to calculate $\mathbf{u}_{j+1}$ can be increased. Algorithms with $N_0 > 1$ are referred to as higher-order learning algorithms [26], [36], [37], [44], [49]–[51]. The current error $\mathbf{e}_{j+1}$ can also be used in calculating $\mathbf{u}_{j+1}$

to obtain a current-iteration learning algorithm [52]–[56]. As shown in [57] and elsewhere in this article (see "Current-Iteration Iterative Learning Control"), the current-iteration learning algorithm is equivalent to the algorithm (4) combined with a feedback controller on the plant.

Outline

The remainder of this article is divided into four major sections. These are "System Representations," "Analysis," "Design," and "Implementation Example." Time and frequency-domain representations of the ILC system are presented in the "System Representation" section. The "Analysis" section examines the four basic topics of greatest relevance to understanding ILC system behavior: 1) stability, 2) performance, 3) transient learning behavior, and 4) robustness. The "Design" section investigates four different design methods: 1) PD type, 2) plant inversion, 3) $\mathcal{H}_\infty$, and 4) quadratically optimal. The goal is to give the reader an array of tools to use and the knowledge of when each is appropriate. Finally, the design and implementation of an iterative learning controller for microscale robotic deposition manufacturing is presented in the "Implementation Example" section. This manufacturing example gives a quantitative evaluation of ILC benefits.

SYSTEM REPRESENTATIONS

Analytical results for ILC systems are developed using two system representations. Before proceeding with the analysis, we introduce these representations.

Time-Domain Analysis Using the Lifted-System Framework

To construct the lifted-system representation, the rational LTI plant (1) is first expanded as an infinite power series by dividing its denominator into its numerator, yielding

$$P(\mathrm{q}) = p_1\mathrm{q}^{-1} + p_2\mathrm{q}^{-2} + p_3\mathrm{q}^{-3} + \cdots, \tag{5}$$

where the coefficients p_k are Markov parameters [58]. The sequence $p_1, p_2, \ldots$ is the impulse response. Note that $p_1 \neq 0$ since $m = 1$ is assumed. For the state space description (2), (3), p_k is given by $p_k = \mathbf{C}\mathbf{A}^{k-1}\mathbf{B}$. Stacking the signals in vectors, the system dynamics in (1) can be written equivalently as the $N \times N$-dimensional lifted system

Current-Iteration Iterative Learning Control

Current-iteration ILC is a method for incorporating feedback with ILC [52]–[56]. The current-iteration ILC algorithm is given by

$$u_{j+1}(k) = Q(\mathrm{q})[u_j(k) + L(\mathrm{q})e_j(k+1)] + \underbrace{C(\mathrm{q})e_{j+1}(k)}_{\text{Feedback control}}$$

and shown in Figure A. This learning scheme derives its name from the addition of a learning component in the current iteration through the term $C(\mathrm{q})e_{j+1}(k)$. This algorithm, however, is identical to the algorithm (4) combined with a feedback controller in the parallel architecture. Equivalence can be found between these two forms by separating the current-iteration ILC signal into feedforward and feedback components as

$$u_{j+1}(k) = w_{j+1}(k) + C(\mathrm{q})e_{j+1}(k),$$

where

$$w_{j+1}(k) = Q(\mathrm{q})[u_j(k) + L(\mathrm{q})e_j(k+1)].$$

Then, solving for the iteration-domain dynamic equation for w yields

$$w_{j+1}(k) = Q(\mathrm{q})[w_j(k) + (L(\mathrm{q}) + q^{-1}C(\mathrm{q}))e_j(k+1)].$$

Therefore, the feedforward portion of the current-iteration ILC is identical to the algorithm (4) with the learning function $L(\mathrm{q}) + \mathrm{q}^{-1}C(\mathrm{q})$. The algorithm (4) with learning function $L(\mathrm{q}) + \mathrm{q}^{-1}C(\mathrm{q})$ combined with a feedback controller in the parallel architecture is equivalent to the complete current-iteration ILC.

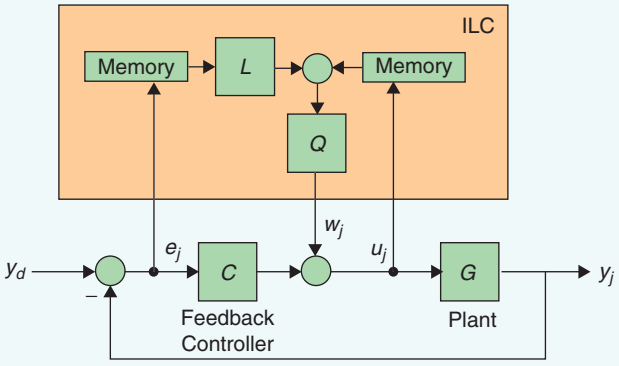

FIGURE A Current-iteration ILC architecture, which uses a control signal consisting of both feedforward and feedback in its learning algorithm. This architecture is equivalent to the algorithm (4) combined with a feedback controller in the parallel architecture.

$$\underbrace{\begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(N) \end{bmatrix}}_{\mathbf{y}_j} = \underbrace{\begin{bmatrix} p_1 & 0 & \cdots & 0 \\ p_2 & p_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_N & p_{N-1} & \cdots & p_1 \end{bmatrix}}_{\mathbf{P}} \underbrace{\begin{bmatrix} u_j(0) \\ u_j(1) \\ \vdots \\ u_j(N-1) \end{bmatrix}}_{\mathbf{u}_j} + \underbrace{\begin{bmatrix} d(1) \\ d(2) \\ \vdots \\ d(N) \end{bmatrix}}_{\mathbf{d}}, \tag{6}$$

and

$$\underbrace{\begin{bmatrix} e_j(1) \\ e_j(2) \\ \vdots \\ e_j(N) \end{bmatrix}}_{\mathbf{e}_j} = \underbrace{\begin{bmatrix} y_d(1) \\ y_d(2) \\ \vdots \\ y_d(N) \end{bmatrix}}_{\mathbf{y}_d} - \underbrace{\begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(N) \end{bmatrix}}_{\mathbf{y}_j}.$$

The components of $\mathbf{y}_j$ and $\mathbf{d}$ are shifted by one time step to accommodate the one-step delay in the plant, ensuring that the diagonal entries of P are nonzero. For a plant with m-step delay, the lifted system representation is

$$\begin{bmatrix} y_j(m) \\ y_j(m+1) \\ \vdots \\ y_j(m+N-1) \end{bmatrix} = \begin{bmatrix} p_m & 0 & \cdots & 0 \\ p_{m+1} & p_m & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_{m+N-1} & p_{m+N-2} & \cdots & p_m \end{bmatrix} \times \begin{bmatrix} u_j(0) \\ u_j(1) \\ \vdots \\ u_j(N-1) \end{bmatrix} + \begin{bmatrix} d(m) \\ d(m+1) \\ \vdots \\ d(m+N-1) \end{bmatrix},$$

$$\begin{bmatrix} e_j(m) \\ e_j(m+1) \\ \vdots \\ e_j(m+N-1) \end{bmatrix} = \begin{bmatrix} y_d(m) \\ y_d(m+1) \\ \vdots \\ y_d(m+N-1) \end{bmatrix} - \begin{bmatrix} y_j(m) \\ y_j(m+1) \\ \vdots \\ y_j(m+N-1) \end{bmatrix}.$$

The lifted form (6) allows us to write the SISO time- and iteration-domain dynamic system (1) as a multiple-input, multiple-output (MIMO) iteration-domain dynamic system. The time-domain dynamics are contained in the structure of **P**, and the time signals u_j, y_j, and d are contained in the vectors $\mathbf{u}_j$, $\mathbf{y}_j$, and $\mathbf{d}$.

Likewise, the learning algorithm (4) can be written in lifted form. The Q-filter and learning function can be noncausal functions with the impulse responses

$$Q(\mathrm{q}) = \cdots + q_{-2}\mathrm{q}^2 + q_{-1}\mathrm{q}^1 + q_0 + q_1\mathrm{q}^{-1} + q_2\mathrm{q}^{-2} + \cdots$$

and

$$L(\mathrm{q}) = \cdots + l_{-2}\mathrm{q}^2 + l_{-1}\mathrm{q}^1 + l_0 + l_1\mathrm{q}^{-1} + l_2\mathrm{q}^{-2} + \cdots,$$

respectively. In lifted form, (4) becomes

$$\underbrace{\begin{bmatrix} u_{j+1}(0) \\ u_{j+1}(1) \\ \vdots \\ u_{j+1}(N-1) \end{bmatrix}}_{\mathbf{u}_{j+1}} = \underbrace{\begin{bmatrix} q_0 & q_{-1} & \cdots & q_{-(N-1)} \\ q_1 & q_0 & \cdots & q_{-(N-2)} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N-1} & q_{N-2} & \cdots & q_0 \end{bmatrix}}_{\mathbf{Q}} \left(\underbrace{\begin{bmatrix} u_j(0) \\ u_j(1) \\ \vdots \\ u_j(N-1) \end{bmatrix}}_{\mathbf{u}_j} + \underbrace{\begin{bmatrix} l_0 & l_{-1} & \cdots & l_{-(N-1)} \\ l_1 & l_0 & \cdots & l_{-(N-2)} \\ \vdots & \vdots & \ddots & \vdots \\ l_{N-1} & l_{N-2} & \cdots & l_0 \end{bmatrix}}_{\mathbf{L}} \underbrace{\begin{bmatrix} e_j(1) \\ e_j(2) \\ \vdots \\ e_j(N) \end{bmatrix}}_{\mathbf{e}_j} \right). \tag{7}$$

When $Q(\mathrm{q})$ and $L(\mathrm{q})$ are causal functions, it follows that $q_{-1} = q_{-2} = \cdots = 0$ and $l_{-1} = l_{-2} = \cdots = 0$, and thus the matrices **Q** and **L** are lower triangular. Further distinctions on system causality can be found in "Causal and Noncausal Learning."

The matrices **P**, **Q**, and **L** are Toeplitz [59], meaning that all of the entries along each diagonal are identical. While LTI systems are assumed here, the lifted framework can easily accommodate an LTV plant, Q-filter, or learning function [42], [60]. The construction for LTV systems is the same, although **P**, **Q**, and **L** are not Toeplitz in this case.

Frequency-Domain Analysis Using the z-Domain Representation

The one-sided z-transformation of the signal $\{x(k)\}_{k=0}^{\infty}$ is $X(z) = \sum_{k=0}^{\infty} x(k)z^{-1}$, and the z-transformation of a system is obtained by replacing q with z. The frequency response of a z-domain system is given by replacing z with $e^{i\theta}$ for $\theta \in [-\pi, \pi]$. For a sampled-data system, $\theta = \pi$ maps to the Nyquist frequency. To apply the z-transformation to the ILC system (1), (4), we must have $N = \infty$ because the

z-transform requires that signals be defined over an infinite time horizon. Since all practical applications of ILC have finite trial durations, the z-domain representation is an approximation of the ILC system [44]. Therefore, for the z-domain analysis we assume $N = \infty$, and we discuss what can be inferred about the finite-duration ILC system [44], [56], [61]–[64].

The transformed representations of the system in (1) and learning algorithm in (4) are

$$Y_j(z) = P(z)U_j(z) + D(z) \tag{8}$$

and

$$U_{j+1}(z) = Q(z)[U_j(z) + zL(z)E_j(z)], \tag{9}$$

respectively, where $E_j(z) = Y_d(z) - Y_j(z)$. The z that multiplies $L(z)$ emphasizes the forward time shift used in the learning. For an m time-step plant delay, z^m is used instead of z.

ANALYSIS

Stability

The ILC system (1), (4) is *asymptotically stable* (AS) if there exists $\bar{u} \in \mathbb{R}$ such that

$$|u_j(k)| \leq \bar{u} \text{ for all } k = \{0, \ldots, N-1\} \text{ and } j = \{0, 1, \ldots, \},$$

and, for all $k \in \{0, \ldots, N-1\}$,

$$\lim_{j \to \infty} u_j(k) \text{ exists.}$$

We define the converged control as $u_\infty(k) = \lim_{j\to\infty} u_j(k)$. Time-domain and frequency-domain conditions for AS of the ILC system are presented here and developed in [44].

Substituting $\mathbf{e}_j = \mathbf{y_d} - \mathbf{y}_j$ and the system dynamics (6) into the learning algorithm (7) yields the closed-loop iteration domain dynamics

$$\mathbf{u}_{j+1} = \mathbf{Q}(\mathbf{I} - \mathbf{LP})\mathbf{u}_j + \mathbf{QL}(\mathbf{y_d} - \mathbf{d}). \tag{10}$$

Let $\rho(\mathbf{A}) = \max_i |\lambda_i(\mathbf{A})|$ be the spectral radius of the matrix $\mathbf{A}$, and $\lambda_i(\mathbf{A})$ the ith eigenvalue of $\mathbf{A}$. The following AS condition follows directly.

Theorem 1 [44]

The ILC system (1), (4) is AS if and only if

$$\rho(\mathbf{Q}(\mathbf{I} - \mathbf{LP})) < 1. \tag{11}$$

When the Q-filter and learning function are causal, the matrix $\mathbf{Q}(\mathbf{I} - \mathbf{LP})$ is lower triangular and Toeplitz with repeated eigenvalues

$$\lambda = q_0(1 - l_0 p_1). \tag{12}$$

In this case, (11) is equivalent to the scalar condition

$$|q_0(1 - l_0 p_1)| < 1. \tag{13}$$

Causal and Noncausal Learning

One advantage that ILC has over traditional feedback and feedforward control is the possibility for ILC to anticipate and preemptively respond to repeated disturbances. This ability depends on the causality of the learning algorithm.

Definition: The learning algorithm (4) is *causal* if $u_{j+1}(k)$ depends only on $u_j(h)$ and $e_j(h)$ for $h \leq k$. It is *noncausal* if $u_{j+1}(k)$ is also a function of $u_j(h)$ or $e_j(h)$ for some $h > k$.

Unlike the usual notion of noncausality, a noncausal learning algorithm is implementable in practice because the entire time sequence of data is available from all previous iterations. Consider the noncausal learning algorithm $u_{j+1}(k) = u_j(k) + k_p e_j(k+1)$ and the causal learning algorithm $u_{j+1}(k) = u_j(k) + k_p e_j(k)$. Recall that a disturbance $d(k)$ enters the error as $e_j(k) = y_d(k) - P(\mathrm{q})u_j(k) - d(k)$. Therefore, the noncausal algorithm anticipates the disturbance $d(k+1)$ and preemptively compensates with the control $u_{j+1}(k)$. The causal algorithm does not anticipate since $u_{j+1}(k)$ compensates for the disturbance $d(k)$ with the same time index k.

Causality also has implications in feedback equivalence [63], [64], [111], [112], which means that the converged control u_∞ obtained in ILC could be obtained instead by a feedback controller. The results in [63], [64], [111], [112] are for continuous-time ILC, but can be extended to discrete-time ILC. In a noise-free scenerio, [111] shows that there is feedback equivalence for causal learning algorithms, and furthermore that the equivalent feedback controller can be obtained directly from the learning algorithm. This result suggests that causal ILC algorithms have little value since the same control action can be provided by a feedback controller without the learning process. However, there are critical limitations to the equivalence that may justify the continued examination and use of causal ILC algorithms. For instance, the feedback control equivalency discussed in [111] is limited to a noise-free scenario. As the performance of the ILC increases, the equivalent feedback controller has increasing gain [111]. In a noisy environment, high-gain feedback can degrade performance and damage equipment. Moreover, this equivalent feedback controller may not be stable [63]. Therefore, causal ILC algorithms are still of significant practical value.

When the learning algorithm is noncausal, the ILC does, in general, preemptively respond to repeating disturbances. Except for special cases, there is no equivalent feedback controller that can provide the same control action as the converged control of a noncausal ILC since feedback control reacts to errors.

Using (8), (9), iteration domain dynamics for the z-domain representation are given by

$$U_{j+1}(z) = Q(z)\,[1 - zL(z)P(z)]\,U_j(z) + zQ(z)L(z)\,[Y_d(z) - D(z)]\,. \qquad (14)$$

A sufficient condition for stability of the transformed system can be obtained by requiring that $Q(z)[1 - zL(z)P(z)]$ be a contraction mapping. For a given z-domain system $T(z)$, we define $\|T(z)\|_\infty = \sup_{\theta\in[-\pi,\pi]} |T(e^{i\theta})|$.

Theorem 2 [44]

If

$$\|Q(z)[1 - zL(z)P(z)]\|_\infty < 1, \qquad (15)$$

then the ILC system (1), (4) with $N = \infty$ is AS.

When $Q(z)$ and $L(z)$ are causal functions, (15) also implies AS for the finite-duration ILC system [44], [52]. The stability condition (15) is only sufficient and, in general, much more conservative than the necessary and sufficient condition (11) [4]. Additional stability results developed in [44] can also be obtained from 2-D systems theory [61], [65] of which ILC systems are a special case [66]–[68].

Performance

The performance of an ILC system is based on the asymptotic value of the error. If the system is AS, the asymptotic error is

$$\begin{aligned} e_\infty(k) &= \lim_{j\to\infty} e_j(k) \\ &= \lim_{j\to\infty} (y_d(k) - P(\mathrm{q})u_j(k) - d(k)) \\ &= y_d(k) - P(\mathrm{q})u_\infty(k) - d(k). \end{aligned}$$

Performance is often judged by comparing the difference between the converged error $e_\infty(k)$ and the initial error $e_0(k)$ for a given reference trajectory. This comparison is done either qualitatively [5], [12], [18], [22] or quantitatively with a metric such as the root mean square (RMS) of the error [43], [46], [69].

If the ILC system is AS, then the asymptotic error is

$$\mathbf{e}_\infty = [\mathbf{I} - \mathbf{P}[\mathbf{I} - \mathbf{Q}(\mathbf{I} - \mathbf{LP})]^{-1}\mathbf{QL}](\mathbf{y_d} - \mathbf{d}) \qquad (16)$$

for the lifted system and

$$E_\infty(z) = \frac{1 - Q(z)}{1 - Q(z)[1 - zL(z)P(z)]}[Y_d(z) - D(z)] \qquad (17)$$

for the z-domain system. These results are obtained by replacing the iteration index j with ∞ in (6), (7) and (8), (9) and solving for $\mathbf{e}_\infty$ and $E_\infty(z)$, respectively [1].

Many ILC algorithms are designed to converge to zero error, $e_\infty(k) = 0$ for all k, independent of the reference or repeating disturbance. The following result gives necessary and sufficient conditions for convergence to zero error.

Theorem 3 [57]

Suppose P and L are not identically zero. Then, for the ILC system (1), (4), $e_\infty(k) = 0$ for all k and for all y_d and d, if and only if the system is AS and $Q(\mathrm{q}) = 1$.

Proof

See [57] for the time-domain case and, assuming $N = \infty$, [11] for the frequency-domain case.

Many ILC algorithms set $Q(\mathrm{q}) = 1$ and thus do not include Q-filtering. Theorem 3 substantiates that this approach is necessary for perfect tracking. Q-filtering, however, can improve transient learning behavior and robustness, as discussed later in this article.

More insight into the role of the Q-filter in performance is offered in [70], where the Q-filter is assumed to be an ideal lowpass filter with unity magnitude for low frequencies $[0, \theta_c]$ and zero magnitude for high frequencies $\theta \in (\theta_c, \pi]$. Although not realizable, the ideal lowpass filter is useful here for illustrative purposes. From (17), $E_\infty(e^{i\theta})$ for the ideal lowpass filter is equal to zero for $\theta \in [0, \theta_c]$ and equal to $Y_d(e^{i\theta}) - D(e^{i\theta})$ for $\theta \in (\theta_c, \pi]$. Thus, for frequencies at which the magnitude of the Q-filter is 1, perfect tracking is achieved; for frequencies at which the magnitude is 0, the ILC is effectively turned off. Using this approach, the Q-filter can be employed to determine which frequencies are emphasized in the learning process. Emphasizing certain frequency bands is useful for controlling the iteration domain transients associated with ILC, as discussed in the "Robustness" section.

Transient Learning Behavior

We begin our discussion of transient learning behavior with an example illustrating transient growth in ILC systems.

Example 1

Consider the ILC system (1), (4) with plant dynamics $y_j(k) = [\mathrm{q}/(\mathrm{q} - .9)^2]u_j(k)$ and learning algorithm $u_{j+1}(k) = u_j(k) + .5e_j(k+1)$. The leading Markov parameters of this system are $p_1 = 1$, $q_0 = 1$, and $l_0 = 0.5$. Since $Q(\mathrm{q})$ and $L(\mathrm{q})$ are causal, all of the eigenvalues of the lifted system are given by (12) as $\lambda = 0.5$. Therefore, the ILC system is AS by Theorem 1. The converged error is identically zero because the Q-filter is unity. The trial duration is set to $N = 50$, and the desired output is the smoothed-step function shown in Figure 2. The 2-norm of $\mathbf{e}_j$ is plotted in Figure 3 for the first 180 trials. Over the first 12 iterations, the error increases by nine orders of magnitude.

Example 1 shows the large transient growth that can occur in ILC systems. Transient growth is problematic in

practice because neither the rate nor the magnitude of the growth is closely related to stability conditions. Recall that the eigenvalues of the closed-loop iteration dynamics in Example 1 are at $\lambda = 0.5$, which is well within the unit disk. Furthermore, it is difficult to distinguish transient growth from instability in practice because the initial growth rate and magnitude are so large. Large transient growth is a fundamental topic in ILC and preventing it is an essential objective in ILC design. Insights into the cause of large transient growth in ILC systems are presented in [4], [7], [43], [71], and [72].

To avoid large learning transients, monotonic convergence is desirable. The system (1), (4) is monotonically convergent under a given norm $\| \bullet \|$ if

$$\|\mathbf{e}_\infty - \mathbf{e}_{j+1}\| \leq \gamma \|\mathbf{e}_\infty - \mathbf{e}_j\|,$$

for $j \in \{1, 2, \dots\}$, where $0 \leq \gamma < 1$ is the convergence rate. We now develop conditions for monotonic convergence.

By manipulating the system dynamics (6), (7) and the asymptotic-error result (16), we obtain

$$\mathbf{e}_\infty - \mathbf{e}_{j+1} = \mathbf{PQ}(\mathbf{I} - \mathbf{LP})\mathbf{P}^{-1}(\mathbf{e}_\infty - \mathbf{e}_j). \tag{18}$$

When $P(q)$, $Q(q)$, and $L(q)$ are causal (that is, $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{L}$ are Toeplitz and lower triangular), the matrices $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{L}$ commute, and (18) reduces to

$$(\mathbf{e}_\infty - \mathbf{e}_{j+1}) = \mathbf{Q}(\mathbf{I} - \mathbf{LP})(\mathbf{e}_\infty - \mathbf{e}_j).$$

For the z-domain system, the error dynamics can be similarly obtained as

$$[E_\infty(z) - E_{j+1}(z)] = Q(z)\,[1 - zL(z)P(z)]\,\big[E_\infty(z) - E_j(z)\big]. \tag{19}$$

Let $\bar{\sigma}(\cdot)$ be the maximum singular value and let $\| \cdot \|_2$ denote the Euclidean norm. Using (18) and (19) we obtain the following monotonic convergence conditions.

Theorem 4 [44]

If the ILC system (1), (4) satisfies

$$\gamma_1 \triangleq \bar{\sigma}\left(\mathbf{PQ}(\mathbf{I} - \mathbf{LP})\mathbf{P}^{-1}\right) < 1, \tag{20}$$

then

$$\|\mathbf{e}_\infty - \mathbf{e}_{j+1}\|_2 < \gamma_1 \|\mathbf{e}_\infty - \mathbf{e}_j\|_2$$

for all $j \in \{1, 2, \dots\}$.

Theorem 5 [11]

If the ILC system (1), (4) with $N = \infty$ satisfies

$$\gamma_2 \triangleq \|Q(z)\,[1 - zL(z)P(z)]\|_\infty < 1, \tag{21}$$

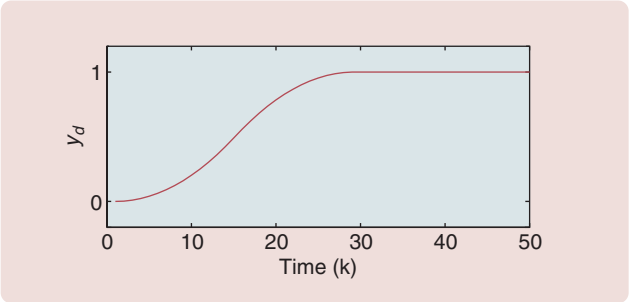

FIGURE 2 Reference command for Example 1. This smoothed step is generated from a trapezoidal velocity profile to enforce a bound on the acceleration. Industrial motion controllers often use smoothed references.

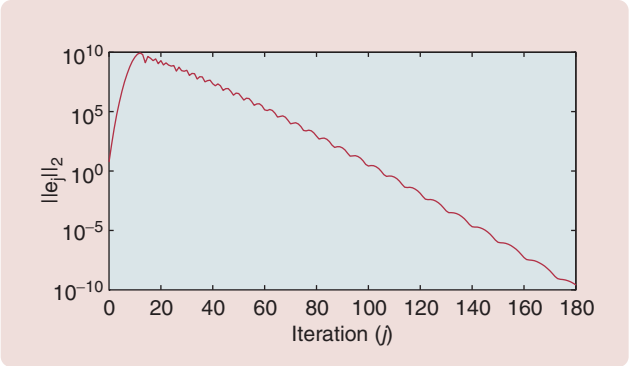

FIGURE 3 Error 2-norm for Example 1. Despite the use of a smoothed step, the tracking error grows rapidly over the first ten

then

$$\big\|E_\infty(z) - E_{j+1}(z)\big\|_\infty < \gamma_2 \big\|E_\infty(z) - E_j(z)\big\|_\infty$$

for all $j \in \{1, 2, \dots\}$.

When $Q(z)$ and $L(z)$ are causal functions, (21) also implies $\|\mathbf{e}_\infty - \mathbf{e}_{j+1}\|_2 < \gamma_2 \|\mathbf{e}_\infty - \mathbf{e}_j\|_2$ for $j \in \{1, 2, \dots\}$ for the ILC system with a finite-duration N [44]. Note that the z-domain monotonic convergence condition (21) is identical to the stability condition (15) given in Theorem 2. Thus, when $Q(z)$ and $L(z)$ are causal functions, the stability condition (15) provides stability and monotonic convergence independent of the iteration duration N. In contrast, the monotonic convergence condition of the lifted system (20) is a more stringent requirement than the stability condition (11), and both are specific to the iteration duration N under consideration.

In some cases, the learning transient behavior of an ILC system may be more important than stability. Some researchers have argued that unstable ILC algorithms can be effective if their initial behavior quickly decreases the error [52], [56], [71]. These algorithms can then be said to satisfy a "practical stability" condition because the learning can be stopped at a low error before the divergent learning transient behavior begins.

For an AS ILC system, the worst-case learning transients can be upper bounded by a decaying geometric function

$$\|\mathbf{e}_\infty - \mathbf{e}_j\|_2 < \bar{\gamma}^{\,j}\bar{\kappa}\|\mathbf{e}_\infty - \mathbf{e}_0\|_2,$$

where $|\bar{\gamma}| < 1$. This result is well known for stable LTI discrete-time systems, and the bounding function can be constructed using Lyapunov analysis [58]. This result is specific to the trial duration N under consideration. In general, altering the trial duration alters the bounding function.

Robustness

Implicit in the ILC formulation is uncertainty in the plant dynamics. If the plant were known exactly, more direct methods could be used to track a given reference. As such, robustness is a central issue in ILC. Here, we consider robustness to uncertainty as related to stability and monotonicity. This discussion provides insight into the limitations that safe operation requirements (good transients and stability) impose on the achievable performance of a learning algorithm. We then consider stability robustness to time-delay error in the system model and, finally, the effects of nonrepeating disturbances on performance.

A key question is whether or not a given AS ILC system remains AS to plant perturbations. Consider the scenario in which $Q(\mathrm{q}) = 1$, which achieves zero converged error, and $L(\mathrm{q})$ is causal [4]. Then the stability condition (13) is equivalent to $|1 - l_0 p_1| < 1$. Therefore, assuming l_0 and p_1 are both nonzero, the ILC system is AS if and only if

$$\mathrm{sgn}(p_1) = \mathrm{sgn}(l_0), \tag{22}$$

and

$$l_0 p_1 \le 2, \tag{23}$$

where sgn is the signum function.

This result shows that ILC can achieve zero converged error for a plant using only knowledge of the sign of p_1 and an upper bound on $|p_1|$. Perturbations in the parameters $p_2, p_3, \ldots$ do not destabilize the ILC system in this scenario. Since (23) can be satisfied for an arbitrarily large upper bound on $|p_1|$ by choosing $|l_0|$ sufficiently small, we conclude that ILC systems can be stably robust to all perturbations that do not change the sign of p_1. Robust stability, however, does not imply acceptable learning transients.

Consider the uncertain plant

$$P(\mathrm{q}) = \hat{P}(\mathrm{q})[1 + W(\mathrm{q})\Delta(\mathrm{q})], \tag{24}$$

where $\hat{P}(\mathrm{q})$ is the nominal plant model, $W(\mathrm{q})$ is known and stable, and $\Delta(\mathrm{q})$ is unknown and stable with $\|\Delta(z)\|_\infty < 1$. Robust monotonicity is given by the following theorem.

Theorem 6

If

$$|W(e^{i\theta})| \le \frac{\gamma^* - |Q(e^{i\theta})||1 - e^{i\theta}L(e^{i\theta})\hat{P}(e^{i\theta})|}{|Q(e^{i\theta})||e^{i\theta}L(e^{i\theta})\hat{P}(e^{i\theta})|}. \tag{25}$$

for all $\theta \in [-\pi, \pi]$, then the ILC system (1), (4), (24) with $N = \infty$ is monotonically convergent with convergence rate $\gamma^* < 1$.

Proof

From Theorem 5, the ILC system is monotonically convergent with rate $\gamma^* < 1$ if

$$\begin{aligned}\gamma^* &\ge \|Q(z)[1 - zL(z)\hat{P}(z)[1 + W(z)\Delta(z)]]\|_\infty \\ &= \max_\theta |Q(e^{i\theta})[1 - e^{i\theta}L(e^{i\theta})\hat{P}(e^{i\theta})[1 + W(e^{i\theta})\Delta(e^{i\theta})]]| \\ &= \max_\theta [|Q(e^{i\theta})[1 - e^{i\theta}L(e^{i\theta})\hat{P}(e^{i\theta})]| \\ &\quad + |(Q(e^{i\theta})e^{i\theta}L(e^{i\theta})\hat{P}(e^{i\theta})W(e^{i\theta})|],\end{aligned}$$

which is equivalent to (25).

Unlike the stability robustness condition (22), (23), which depends only on the first Markov parameters, the monotonic robustness condition (25) depends on the dynamics of $P(\mathrm{q})$, $Q(\mathrm{q})$, and $L(\mathrm{q})$. Examining (25), we see that the most direct way to increase the robustness $|W(e^{i\theta})|$ at a given θ is to decrease the Q-filter gain $|Q(e^{i\theta})|$. Recall from the "Performance" section that decreasing $|Q(e^{i\theta})|$ negatively impacts the converged performance. Therefore, when we consider a more practical measure of robustness, such as monotonic robustness, there is a tradeoff between performance and robustness, with the Q-filter acting as the tuning knob.

Uncertainty in the system time delay may also lead to stability and learning transient problems. Recall that the error is forward shifted in the learning algorithm by the system delay m. If m is not the true delay, then the entries of the matrix $\mathbf{P}$ are shifted diagonally [73]. If, instead, the plant has an unknown time-varying delay, then each row of $\mathbf{P}$ is shifted left or right according to the system delay at that time. Either case can lead to loss of stability or poor learning transients, and currently there is no complete analysis technique for determining robustness to time-delay error. However, several ILC algorithms have been developed for dealing with a constant time-delay error [50], [74]–[77].

As a robust performance issue, we consider the effects of noise, nonrepeating disturbances, and initial condition variation on performance. All of these effects can be considered together by adding an iteration-dependent exogenous signal d_j to (1). The iteration-dependent signal prevents the error from converging to e_∞. However, if d_j is bounded, then the error converges

to a ball centered around e_∞. Ideally, we do not want the ILC to attempt to learn from d_j, so we might expect that slower learning would decrease the sensitivity to d_j. However, the counter-intuitive result presented in [78] shows that if d_j is a stationary stochastic process, then learning should be fast to minimize the error spectral density. That is, we should choose Q(q), P(q), and L(q) so that $|Q(e^{i\theta})[1 - e^{i\theta}L(e^{i\theta})P(e^{i\theta})]| \ll 1$ at frequencies at which the spectral density of d_j is large. We cannot expect to find $L(e^{i\theta})$ such that $1 - e^{i\theta}L(e^{i\theta})P(e^{i\theta})$ is small for all perturbations of P, so we see once again that decreasing the Q-filter bandwidth is the best way to improve robustness. The tradeoff here is between nominal performance and robust performance.

Performance robustness to noise, nonrepeating disturbances, and initial condition variation can also be handled in an optimal stochastic ILC framework [16], [79]–[81]. Alternatively, the noise robustness problem can be addressed by adding online observers [42]. As a simple ad hoc approach to dealing with noise and nonrepeating disturbances, we recommend running several trials between updates and using the averages of the trials to update u_{j+1}. Robustness to initial condition variation is discussed in [82]–[86]. Although [82]–[86] consider continuous-time systems, parallels for many of the results contained therein can be obtained for discrete-time systems. Additional references for some of the results beyond the scope of this review can be found in "Extensions to Time-Varying, Continuous-Time, Multivariable and Nonlinear Systems."

DESIGN

From a practical perspective, the goal of ILC is to generate an open-loop signal that approximately inverts the plant's dynamics to track the reference and reject repeating disturbances. Ideally, ILC learns only the repeating disturbance and ignores noise and nonrepeating disturbances. We emphasize that ILC is an open-loop control and has no feedback mechanism to respond to unanticipated, nonrepeating disturbances. As such, a feedback controller in combination with ILC can be beneficial. "Using Feedback Control with Iterative Learning Control" illustrates the two basic forms for combining ILC with feedback algorithms.

In the following sections, we discuss four of the most popular ILC algorithms and design techniques. The PD-type learning function is a tunable design that can be applied to a system without extensive modeling and analysis. The plant inversion learning function converges quickly but relies heavily on modeling and can be sensitive to model errors. The $\mathcal{H}_\infty$ design technique can be used to design a robustly monotonically convergent ILC but at the expense of performance. The quadratically optimal (Q-ILC) designs use a quadratic performance criterion to obtain an optimal ILC. An experimental comparison of the P-type, plant inversion, and Q-ILC designs on a rotary robot are presented in [40].

PD-Type and Tunable Designs

As the name implies, the PD-type learning function consists of a proportional and derivative gain on the error. The learning function used in Arimoto's original work [5] on ILC is a continuous-time, D-type learning function. The P-, D-, and PD-type learning functions are arguably the most widely used types of learning functions, particularly for nonlinear systems [5], [12], [39], [81], [83], [86]–[92]. These learning functions rely on tuning and do not require an accurate model for implementation, similar to PID feedback control. The integrator, or I term, is rarely used for learning functions because ILC has a natural integrator action from one trial to the next. The discrete-time, PD-type learning function can be written as

$$u_{j+1}(k) = u_j(k) + k_p e_j(k+1) + k_d[e_j(k+1) - e_j(k)], \quad (26)$$

where k_p is the proportional gain and k_d is the derivative gain. Some authors [45] use the proportional gain on $e_j(k)$ rather than $e_j(k+1)$.

From Theorem 1, the ILC system with the PD-type learning algorithm is AS if and only if $|1 - (k_p + k_d)p_1| < 1$. Clearly, when p_1 is known, it is always possible to find k_d and k_p such that the ILC system is AS. Monotonic convergence, however, is not always possible using a PD-type learning algorithm. However, when the iteration is sufficiently short, monotonic convergence can be achieved using PD-type learning. Although this relationship is shown for continuous-time systems in [94], parallel results can be obtained for discrete-time systems. Also, conditions on the plant dynamics under which a P-type ($k_d = 0$) learning algorithm is monotonic are given in [93]. In [45] an optimization-based design that indirectly attempts to minimize the convergence rate is presented, but monotonic convergence is not guaranteed. The most favorable and generally applicable approach to achieving monotonic convergence is to modify the learning algorithm to include a lowpass Q-filter [4], [7], [41], [70]. As discussed earlier, the lowpass Q-filter can be used to disable learning at high frequencies, which is useful for satisfying the monotonic convergence condition in Theorem 5. The Q-filter also has the benefits of added robustness and high-frequency noise filtering.

Just as with PD feedback controllers, the most commonly employed method for selecting the gains of the PD-type learning function is by tuning [5], [6], [10], [12], [17], [27], [43]. When a Q-filter is used, the filter type (for example, Butterworth, Chebyshev, Gaussian, or FIR) and order are specified, and the bandwidth of the filter is used as the tuning variable. Despite the popularity of this approach, ILC tuning guidelines comparable to the Ziegler-Nichols [95] rules for PID feedback control are not available. In lieu of formal guidelines, we offer the following suggestions. The goals of the tuning include both good learning transients and low error. For each set of gains k_p and k_d, the learning is

reset and run for sufficient iterations to determine the transient behavior and asymptotic error. Initially, the learning gains and filter bandwidth are set low. After a stable baseline transient behavior and error performance have been obtained, the gains and bandwidth can be increased. Depending on the application, the learning gains influence the rate of convergence, whereas the Q-filter influences the converged error performance. Increasing the Q-filter

Extensions to Time-Varying, Continuous-Time, Multivariable, and Nonlinear Systems

The lifted system framework can accommodate discrete-time LTV plants, learning functions, and Q-filters [1], [42], [60]. For example, consider the LTV plant

$$P(k, \mathrm{q}) = p_1(k)\mathrm{q}^{-1} + p_2(k)\mathrm{q}^{-2} + p_3(k)\mathrm{q}^{-3} + \cdots .$$

The lifted form of $P(k, \mathrm{q})$ is

$$\underbrace{\begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(N) \end{bmatrix}}_{\mathbf{y}_j} = \underbrace{\begin{bmatrix} p_1(0) & 0 & \cdots & 0 \\ p_2(1) & p_1(1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_N(N-1) & p_{N-1}(N-1) & \cdots & p_1(N-1) \end{bmatrix}}_{\mathbf{P}_{LTV}} \times \underbrace{\begin{bmatrix} u_j(0) \\ u_j(1) \\ \vdots \\ u_j(N-1) \end{bmatrix}}_{\mathbf{u}_j} + \underbrace{\begin{bmatrix} d(1) \\ d(2) \\ \vdots \\ d(N) \end{bmatrix}}_{\mathbf{d}} .$$

Unlike $\mathbf{P}$ in (6), $\mathbf{P}_{LTV}$ is not Toeplitz, although both $\mathbf{P}$ and $\mathbf{P}_{LTV}$ are iteration-invariant. Therefore, the closed-loop dynamics in the iteration domain for $P(k, \mathrm{q})$ is given by (10) with $\mathbf{P}$ replaced by $\mathbf{P}_{LTV}$. It is easy to verify that AS and monotonic convergence for the LTV plant are given by Theorems 1 and 4, respectively, with $\mathbf{P}$ replaced by $\mathbf{P}_{LTV}$ in (11) and (20).

For continuous-time LTI systems, stability, performance, and monotonic convergence results are available in the frequency domain [1], [52], [113], [114]. Results for LTV continuous-time systems can be found in [5], [36], [115]. For square multiple-input multiple-output (MIMO) systems, extensions of the above results are generally straightforward. For nonsquare MIMO systems, see [37], [113], [116] for further detail.

Nonlinear systems have also received a lot of focus in ILC research, both in analysis and algorithm design. We can broadly separate nonlinear systems into two groups, namely, those that are affine in the control input and those that are not. Systems that are affine in the control are assumed to have the form

$$\begin{aligned} \dot{x}(t) &= f(x(t)) + B(x(t))u(t) \\ y(t) &= g(x(t)), \end{aligned}$$

where x is the system states, u is the system input, and y is the system output. This type of system is examined in [8], [36], [50], [85], [89], and [117]–[120]. A special case of this type of system is an n-link robot [39] described by

$$\mathbf{M}_\mathbf{r}(\mathbf{q})\ddot{\mathbf{q}} - \mathbf{C}_\mathbf{r}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{g}_\mathbf{r}(\mathbf{q}) - \mathbf{d}_\mathbf{r}(\mathbf{q},\dot{\mathbf{q}}) = \tau,$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are $n \times 1$ vectors of link positions, velocities, and accelerations; $\boldsymbol{\tau}$ is the $n \times 1$ vector of torque inputs for each link; $\mathbf{M}_\mathbf{r}(\mathbf{q})$ is a symmetric positive-definite $n \times n$ matrix of link inertias; $\mathbf{C}_\mathbf{r}(\mathbf{q}, \dot{\mathbf{q}})$ is the $n \times n$ Coriolis and centripetal acceleration matrix; $\mathbf{g}_\mathbf{r}(\mathbf{q})$ is the $n \times 1$ gravitational force vector; and $\mathbf{d}_\mathbf{r}(\mathbf{q}, \dot{\mathbf{q}})$ is the $n \times 1$ friction force vector. ILC for nonlinear robot dynamics is examined in [90] and [121]–[123].

While ILC for affine nonlinear systems uses a variety of learning algorithms, one of the key assumptions common to all of these algorithms is that the nonlinear system is smooth, which is often expressed as a global Lipshitz constraint on each of the functions [36], [86], [89], [117], [124].

$$\begin{aligned} |f(x_1) - f(x_2)| &\le f_o|x_1 - x_2| \\ |B(x_1) - B(x_2)| &\le b_o|x_1 - x_2| \\ |g(x_1) - g(x_2)| &\le g_o|x_1 - x_2|. \end{aligned}$$

The Lipshitz constants f_0, b_0, g_0 are used in a contraction mapping to demonstrate stability of the nonlinear ILC system. In addition to stability, research on nonlinear ILC includes performance [8], [118], [124], learning transient behavior [119], [121], and robustness to initial condition variation [36], [50], [85], [86], [120], repeating disturbances [36], [50], [86], [120], and model uncertainty [36], [50], [89].

Nonaffine systems have the form

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)), \\ y(t) &= g(x(t)), \end{aligned}$$

and ILC for this system type is examined in [36], [37], [92]. Robustness of this system type is covered in [37], [92].

ILC has also been extended to discrete-time nonlinear systems. While obtaining discrete-time models of nonlinear systems may be nontrivial, the results are often simpler than their continuous-time counterparts and are consistent with the digital implementation typical of ILC controllers. These works include [125]–[132] for affine systems and [133], [134] for nonaffine systems.

Another way to analyze and design ILC for nonlinear systems is to treat the nonlinearities as perturbations to the linearized system. When the system is under feedback control, it is likely that the trajectory remains in a neighborhood of the reference trajectory. In this case, the nonlinearities can be evaluated along the reference trajectory to yield additive LTV system and signal perturbations that are iteration independent [87]. Therefore, the nonlinearities can be accounted for in design with model uncertainty [9], [10], [12], [17], [22], [41]–[43], [56].

bandwidth decreases robustness but improves performance, whereas decreasing the bandwidth has the opposite effect. We caution that large transients can appear quickly, as demonstrated in Example 1. The signals must be monitored closely for several iterations beyond the apparent convergence for signs of high-frequency growth.

In [4] a two-step tuning method is recommended using an experimentally obtained frequency response of the system. The approach is based on satisfying the AS and monotonic convergence condition in (15), which can be rewritten as

$$|1 - e^{i\theta}L(e^{i\theta})P(e^{i\theta})| < \frac{1}{|Q(e^{i\theta})|},$$

for all $\theta \in [-\pi, \pi]$. Using a Nyquist plot of $e^{i\theta}L(e^{i\theta})P(e^{i\theta})$, the learning gains are tuned to maximize the range $\theta \in [0, \theta_c]$ over which $e^{i\theta}L(e^{i\theta})P(e^{i\theta})$ lies inside the unit circle centered at 1. The Q-filter bandwidth is selected last to satisfy the stability condition.

Plant Inversion Methods

Plant inversion methods use models of the inverted plant dynamics as the learning function. The discrete-time plant inversion ILC algorithm is given by

$$u_{j+1}(k) = u_j(k) + \hat{P}^{-1}(\mathrm{q})e_j(k).$$

By rewriting the learning algorithm as $u_{j+1}(k) = u_j(k) + q^{-1}\hat{P}^{-1}(\mathrm{q})e_j(k+1)$, we see that the plant-inversion learning function is $L(\mathrm{q}) = \mathrm{q}^{-1}\hat{P}^{-1}(\mathrm{q})$, which is causal and has zero relative degree. Assuming that $\hat{P}(\mathrm{q})$ is an exact model of the plant, it can be verified from Theorem 4 or Theorem 5 that the convergence rate is $\gamma = 0$. That is, convergence occurs in just one iteration and the converged error is $e_\infty \equiv 0$.

Using Feedback Control with Iterative Learning Control

As presented here, ILC uses open-loop control action only, which cannot compensate for nonrepeating disturbances. Thus, in most physical implementations, a well-designed feedback controller must be used in combination with ILC. In many cases, a feedback controller already exists on the system, and ILC can be implemented without modifying the feedback controller.

ILC can be combined with a feedback loop in two ways as shown in Figures B and C. In this work, we refer to the first arrangement as *serial* because the ILC control input is applied to the reference before the feedback loop and the second arrangement as *parallel* because the ILC control input and feedback control input are combined. The feedback controller modifies the input/output dynamics with regard to the ILC depending on the particular arrangement. For the serial arrangement, the dynamics are

$$y_j = \underbrace{(1 + GC)^{-1}GC}_{P}\, u_j + \underbrace{(1 + GC)^{-1}GC}_{d} y_d.$$

For the parallel arrangement, the dynamics are

$$y_j = \underbrace{(1 + GC)^{-1}G}_{P}\, u_j + \underbrace{(1 + GC)^{-1}GC}_{d} y_d.$$

Note that setting the ILC input u_j to zero in both cases results in the standard feedback-controlled response to the reference y_d. Therefore, in both of these arrangements, the ILC can be disabled whenever nonrepeating reference trajectories are used.

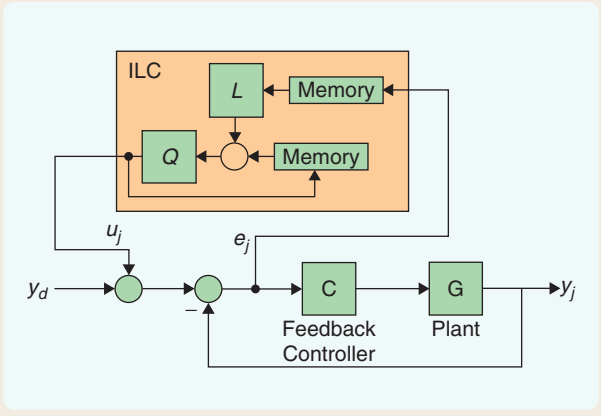

FIGURE B. Serial architecture, which alters the reference signal to the system [4], [135]. This concept is useful when applying ILC to a preexisting system that uses a commercial controller that does not allow direct access to modifying the control signal to the plant.

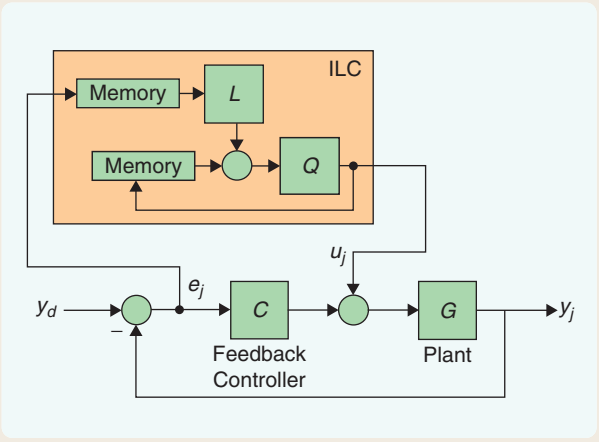

FIGURE C. Parallel architecture, which directly alters the control signal to the plant [11]. This architecture may be more intuitive to motion control designers who add feedforward signals directly to the control signal for improved tracking performance. The individual control contributions from the ILC and feedback controller are also easily separable in this architecture. As the ILC converges, the feedback controller applies less effort.

One of the immediate difficulties with the plant inversion approach occurs when dealing with nonminimum phase systems, in which direct inversion of the plant $P(q)$ results in an unstable filter. Although finite-duration iterations ensure bounded signals, the unstable filter undoubtedly generates undesirably large control signals. This problem can be avoided by using a stable inversion approach, which results in a noncausal learning function [96], [97]. For nonlinear systems, direct inversion of the dynamics may be difficult. However, in some cases, inversion of the dynamics linearized around the dominant operating conditions may be sufficient to achieve good results [7], [98].

Whether $P(q)$ is minimum phase or not, the success of the plant inversion method ultimately depends on the accuracy of the model. A mismatch between the model $\hat{P}(q)$ and the actual dynamics $P(q)$ prevents convergence from occurring in one iteration. Furthermore, mismatch can lead to poor transient behavior. Consider using the plant-inversion learning function with the uncertain system (24). From Theorem 6, the uncertain system is monotonically convergent with a rate better than γ^* if $|W(e^{i\theta})| < \gamma^*$ for $\theta \in [-\pi, \pi]$, where $W(q)$ is the uncertainty weighting function. If at a given θ_0, $|W(e^{i\theta_0})| \geq 1$, signifying an uncertainty greater than 100%, then the system will not be robustly monotonically convergent. Since model uncertainty of greater than 100% is common at high frequencies, the plant inversion algorithm is not robustly monotonically convergent. To avoid poor transients associated with model uncertainty, a lowpass Q-filter is typically employed [7], [99]. By setting the filter cutoff to a sufficiently low frequency, frequencies where uncertainty is greater than 100% can be disabled in the learning function and robust monotonicity can be achieved.

$\mathcal{H}_\infty$ Methods

$\mathcal{H}_\infty$ design methods offer a systematic approach to ILC design. One approach is summarized as follows [11], [70]. The goal is to find the learning function $L(q)$ that offers the fastest convergence rate for a given Q-filter or, equivalently, to solve the model matching problem

$$L^*(z) = \arg\min_L \|Q(z)(I - zL(z)P(z))\|_\infty .$$

Dropping the complex argument for notational convenience, this model matching problem can be written equivalently as a lower linear fractional transform,

$$Q(I - zLP) = G_{11} + G_{12}L(I - G_{22}L)^{-1}G_{21} = \mathcal{F}_\mathcal{L}(G, L),$$

where

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} = \begin{bmatrix} Q & Q \\ -zP & 0 \end{bmatrix}.$$

This system is shown pictorially in Figure 4. In this form, standard $\mathcal{H}_\infty$ synthesis tools can be used to solve the problem directly.

$\mathcal{H}_\infty$ synthesis can be further extended to include models with known uncertainty bounds. In this case, the problem becomes a robust performance problem for which μ-synthesis tools can be applied. Finding a solution for which $\mu < 1$ guarantees robust monotonic convergence. Note that solutions to these synthesis problems, with or without uncertainty, are optimal for a given Q-filter. Recall that the lowpass Q-filter affects the asymptotic error performance, and finding the best tradeoff between convergence rate and performance requires iterating over the Q-filter bandwidth [70].

Alternative $\mathcal{H}_\infty$ design methods focus on current-iteration ILC formulations that include feedback action in the learning. The approach presented in [56] restricts learning to the feedback signal. Thus, the synthesis method is to find the best feedback controller for ILC. In [52], learning is also allowed directly on the error signal as well as the feedback control. To solve this problem, a two-step minimization procedure is presented in which the learning function is designed first and the feedback controller second. In [35], [54], an indirect linear fractional transformation is used to place the current-iteration ILC in standard form for $\mathcal{H}_\infty$ synthesis. This approach allows for the simultaneous design of the feedback controller and learning function.

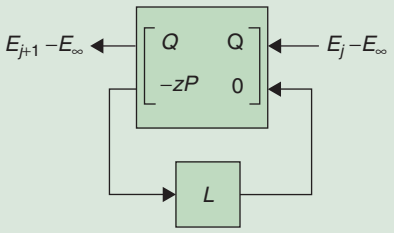

FIGURE 4 ILC iteration dynamics. By using the linear fractional transformation representation, the ILC system is suitable for $\mathcal{H}_\infty$ synthesis to minimize the error transmission from one iteration to the next. This approach is useful for fast-convergence design.

Quadratically Optimal Design

In Q-ILC, the learning functions are designed in the lifted-system representation to minimize a quadratic next-iteration cost criterion. Typically, the criteria used have the form

$$J_{j+1}(\mathbf{u}_{j+1}) = \mathbf{e}_{j+1}^T \mathbf{Q}_\mathbf{LQ} \mathbf{e}_{j+1} + \mathbf{u}_{j+1}^T \mathbf{R}_\mathbf{LQ} \mathbf{u}_{j+1} + \delta_{j+1}\mathbf{u}^T \mathbf{S}_\mathbf{LQ} \delta_{j+1}\mathbf{u},$$

where $\delta_{j+1}\mathbf{u} \triangleq \mathbf{u}_{j+1} - \mathbf{u}_j$, $\mathbf{Q}_\mathbf{LQ}$ is an $N \times N$ positive-definite matrix, and $\mathbf{R}_\mathbf{LQ}$, $\mathbf{S}_\mathbf{LQ}$ are $N \times N$ positive-semidefinite matrices. Minimizing the cost criterion with respect to $\mathbf{u}_{j+1}$ [100], [101] yields the optimal Q-filter

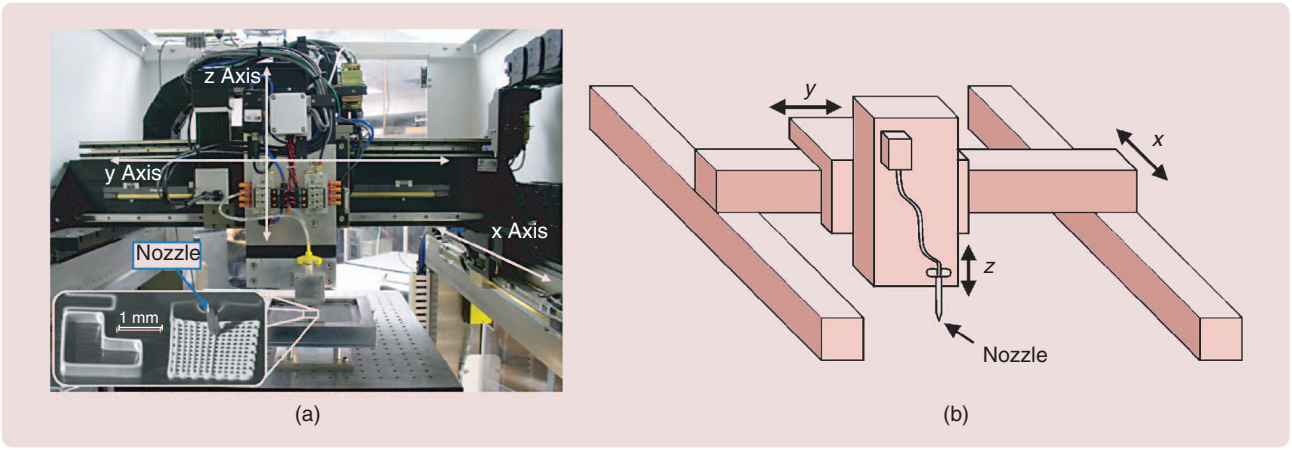

FIGURE 5 (a) Photo of μ-RD system. (b) Schematic of motion system layout. The motion system is a standard H-drive or gantry robot with linear motors for x and y travel and a rotary motor and ballscrew for z travel. An electronic pressure regulator controls the ink extrusion flow rate.

$$\mathbf{Q}_{\text{opt}} = (\mathbf{P}^T\mathbf{Q}_{\mathbf{LQ}}\mathbf{P} + \mathbf{R}_{\mathbf{LQ}} + \mathbf{S}_{\mathbf{LQ}})^{-1}(\mathbf{P}^T\mathbf{Q}_{\mathbf{LQ}}\mathbf{P} + \mathbf{S}_{\mathbf{LQ}})$$

and optimal learning function

$$\mathbf{L}_{\text{opt}} = (\mathbf{P}^T\mathbf{Q}_{\mathbf{LQ}}\mathbf{P} + \mathbf{S}_{\mathbf{LQ}})^{-1}\mathbf{P}^T\mathbf{Q}_{\mathbf{LQ}}.$$

The choice of weighting functions determines the performance, convergence rate, and robustness of the Q-ILC. For instance, substituting $\mathbf{Q}_{opt}$ and $\mathbf{L}_{opt}$ into (16) to obtain the converged error in terms of the weighting function yields

$$\mathbf{e}_{\infty,\text{opt}} = \left[\mathbf{I} - \mathbf{P}(\mathbf{P}^T\mathbf{Q}_{\mathbf{LQ}}\mathbf{P} + \mathbf{R}_{\mathbf{LQ}})^{-1}\mathbf{P}^T\mathbf{Q}_{\mathbf{LQ}}\right](\mathbf{y}_\mathbf{d} - \mathbf{d}).$$

Therefore, the weighting on the change in control $\mathbf{S}_{\mathbf{LQ}}$, which affects how quickly the ILC converges, has no effect on the asymptotic error. However, $\mathbf{R}_{\mathbf{LQ}}$, the weighting on the control, does. Note that if $\mathbf{R}_{\mathbf{LQ}} = \mathbf{0}$, then the error converges to zero for perfect tracking. $\mathbf{R}_{\mathbf{LQ}} \neq \mathbf{0}$ may be useful for limiting the control action to prevent actuator saturation, particularly for nonminimum phase systems [101]. This weighting can also be used to decrease ILC sensitivity at high frequencies to limit hardware wear and damage [101].

The above optimization problem can be solved in a different way leading to a combination of optimal state feedback control and current-iteration ILC [100], [102]. Although more complicated than the above solution, this approach simultaneously yields the ILC and a feedback controller consistent with the aims of the optimization. This approach is extended in [103] to obtain an n-iteration predictive optimal solution.

Robustness of Q-ILC is examined in [22] and [42]. In [22] a loop-shaping ILC is designed by selecting the weighting matrices to meet robust stability, controller fragility, and actuator amplitude requirements. In [42] constrained Q-ILC and robust Q-ILC algorithms are developed to optimize learning for systems with input/output constraints, repeating and nonrepeating disturbances, noise, and bounded-parameter uncertainty. Similarly, a multiobjective noise-attenuation-versus-convergence-rate design is presented in [104].

When the plant is unknown, an adaptive Q-ILC algorithm can be used to identify the system online and redesign the learning functions [60]. To reduce the memory requirements associated with the large matrix form of the Q-ILC algorithm, low-order solutions to Q-ILC can be obtained [105]. These low-order solutions are not specific to the trial duration and can be extrapolated to trial durations of arbitrary length [106]. In addition to the LTI discrete-time Q-ILC, optimal ILC solutions are available for continuous-time, nonlinear Hamiltonian systems [107].

IMPLEMENTATION EXAMPLE: MICROSCALE ROBOTIC DEPOSITION

To demonstrate the effectiveness of ILC on a real system, we consider the microscale robotic deposition (μ-RD) system shown in Figure 5 [6], [108]. μ-RD uses a solid-freeform-fabrication manufacturing technique whereby ink is extruded through a nozzle and deposited onto a substrate [109], [110]. The highly thixotropic ink solidifies quickly after exiting the nozzle, allowing the ink to span open gaps and support high aspect-ratio features. A robot is used to position the nozzle continuously in three-dimensional space to deposit the ink. We are interested in improving the tracking performance of the robot using ILC. Since the robot is Cartesian, the axes are dynamically decoupled by design, and therefore can be considered individually. Here we focus on a single axis, since similar results can be obtained for the other two.

A swept-sine frequency response of the x axis (Figure 6) is used to obtain the 1-kHz sampled dynamic model, as shown in (27) at the bottom of the page. The nominal

$$G_X(z) = \frac{0.00083315(z + 0.9604)(z^2 - 1.981z + 0.9918)(z^2 - 1.874z + 0.9747)}{(z - 0.9994)(z - 1)(z^2 - 1.978z + 0.9894)(z^2 - 1.738z + 0.8672)}. \quad (27)$$

controller uses both a feedback control and a reference feedforward control. The feedback controller (Figure 7)

$$C_{\mathrm{FBK}}(z) = \frac{12.3359(z-0.9874)(z-0.9577)(z-0.945)}{(z-0.9991)(z-0.9174)(z-0.9164)},$$

is designed using loop-shaping for fast transient response. The high-frequency gain of this controller is tuned to be as high as possible without exciting resonant modes of the motor and amplifier. The low-frequency gain is set sufficiently high to compensate for friction and cogging forces while minimizing the oscillatory response of the system to reference changes. To further improve system response, the reference feedforward controller

$$C_{\mathrm{FFD}}(z) = G_X^{-1}(z)F(z)$$

uses approximate inversion of the identified model, where $F(z)$ is the lowpass filter with 100-Hz bandwidth given by

$$F(z) = \frac{0.1568(z+0.8093)}{z^2 - 1.25z + 0.5335},$$

which makes C_{FFD} causal. As an alternative control strategy, we replace the reference feedforward controller with a parallel-architecture ILC controller. The two approaches are then compared against one another on a typical reference trajectory.

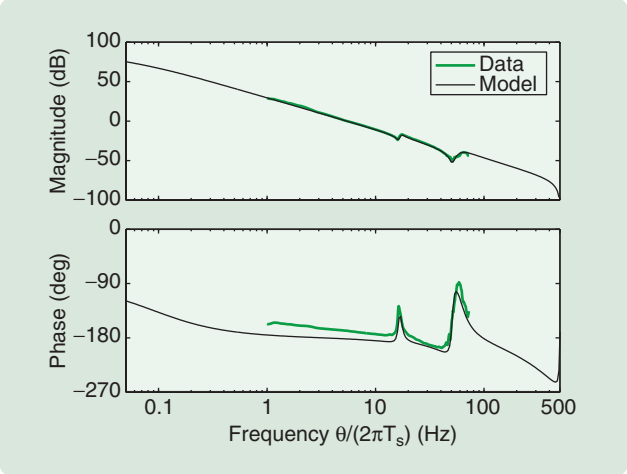

FIGURE 6 Experimental frequency response and model of the μ-RD x axis with sampling period $T_s = 0.001$ s. The experimental data are obtained from 1–70 Hz. Outside this range nonlinearities appear in the form of static friction at low frequencies and high signal-to-noise ratio at high frequencies. The frequency response shows dominant double-integrator dynamics with structural resonances at 16 Hz and 55 Hz.

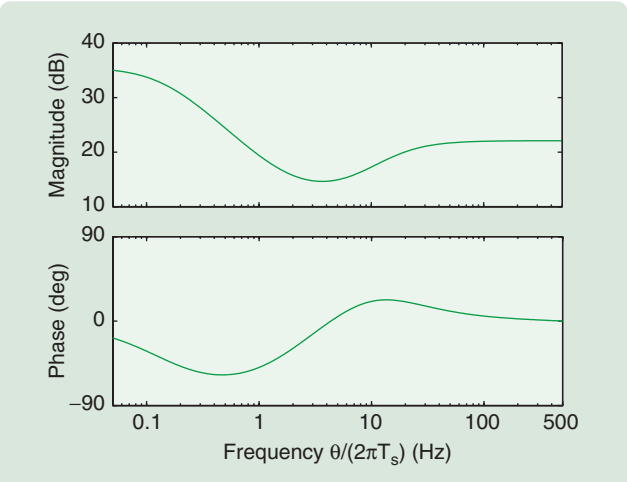

FIGURE 7 Frequency response of the feedback controller $C_{\mathrm{FBK}}(z)$ for the μ-RD x axis with sampling period $T_s = 0.001$ s. This controller is designed to provide fast transient response without exciting resonant modes of the motor and amplifier. The controller also attempts to compensate for friction and cogging forces while minimizing the oscillatory response of the system to reference changes.

Tuning

The design methods discussed in the "Design" section are applicable to this system. However, we select the simplest method, the tuned PD learning function. The rationale is that if the simplest ILC algorithm outperforms a well-tuned nominal control design, then effectiveness of the ILC approach has been demonstrated. We choose a PD learning algorithm of the form

$$u_{j+1}(k) = Q(\mathrm{q})[u_j(k) + k_p e_j(k+1) + k_d(e_j(k+1) - e_j(k))], \tag{28}$$

where $Q(\mathrm{q})$ is a zero-phase, finite-impulse-response Gaussian lowpass filter [6], [39]. Note that (28) is the PD law in (26) with a Q-filter added for robustness. A position step with trapezoidal velocity profile (see Figure 10) is used as the sample trajectory for tuning the ILC parameters. The maximum velocity and acceleration of this trajectory are 10 mm/s and 250 mm/s^2, respectively.

We begin tuning with conservative gains $k_p = 1$ and $k_d = 10$ and a Q-filter bandwidth of 20 Hz. Holding the bandwidth constant, various gains are used, and the maximum and RMS errors are plotted versus iteration. For clarity, only a subset of these gains is shown in Figure 8. From these results, the gains $k_p = 2$ and $k_d = 50$ are selected. Holding these gains fixed, a second set of trials is performed with varying Q-filter bandwidths. Representative results are shown in Figure 9, and the 40-Hz bandwidth filter is chosen as the best among the set.

Tracking Performance Results

Figure 10 shows the output trajectory of the x axis with and without the ILC after 50 learning iterations. Using ILC, the output is nearly indistinguishable from the reference. The ILC reduces the maximum error by 85% and the RMS error by 92% over the well-designed feedback and reference

feedforward controller combination. Note that the learning function was obtained with the simplest of ILC approaches.

CONCLUDING REMARKS

This article surveyed the major results in ILC analysis and design over the past two decades. Problems in stability, performance, learning transient behavior, and robustness were discussed along with four design techniques that have emerged as among the most popular. The content of this survey was selected to provide the reader with a broad perspective of the important ideas, potential, and limitations of ILC. Indeed, the maturing field of ILC includes many results and learning algorithms beyond the scope of this survey. Though beginning its third decade of active research, the field of ILC shows no signs of slowing down.

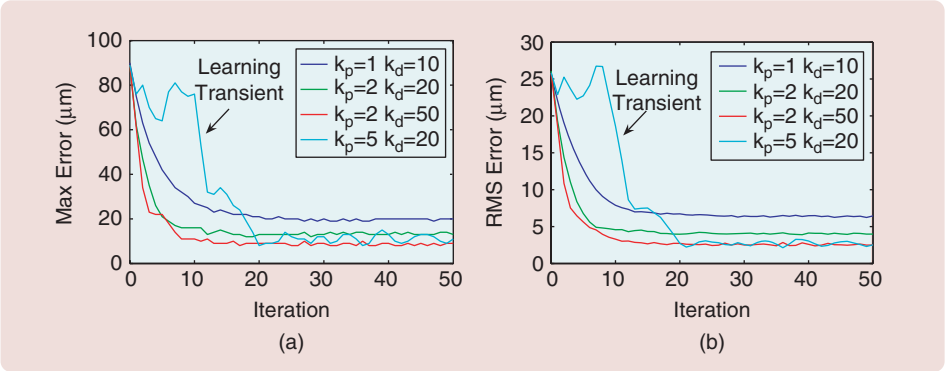

FIGURE 8 Transient behavior and asymptotic performance for PD gain tuning. While the gain combination $k_p = 5$, $k_d = 20$ has low asymptotic error, the learning transients do not converge monotonically like the others and, therefore, may not be a good choice of gains. The other three combinations have nearly monotonic transients, but $k_p = 2$, $k_d = 50$ converges fastest with the lowest final error.

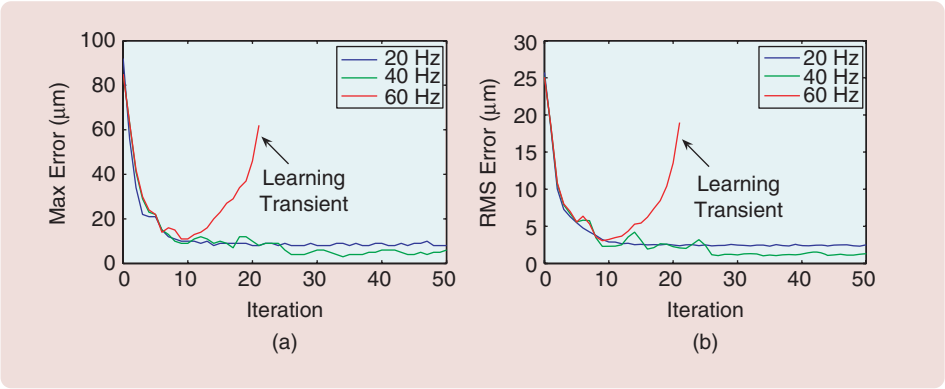

FIGURE 9 Transient behavior and asymptotic performance for Q-filter bandwidth tuning. These plots show that the Q-filter has little effect on the convergence rate of the error, but instead primarily determines the magnitude of the converged error and stability of the system. Although increased bandwidth improves the error, a bandwidth that is too high results in learning transients.

Good learning transient behavior was identified as the practical stability condition, where good transient behavior was defined as monotonic convergence. Furthermore, when robustness is considered, model uncertainty leads to limitations on the performance of LTI learning algorithms. An open research topic is the formalization of the robustness-versus-performance tradeoff. Nonrepeating disturbances and noise can be detrimental to the performance of ILC, yet few algorithms or designs consider these effects in a comprehensive manner amenable to implementation. Finally, ILC researchers have not yet developed general methods for using the information gained through ILC training to aid in nonidentical, but similar, reference signals. Therefore, many opportunities exist for advancement of theoretical and practical knowledge in the field.

The ILC field grew out of practical problems in robotics and should continue to be motivated by real-world problems. One rapidly growing area where the authors see particularly strong opportunities for ILC is in micro- and nano-manufacturing. The dynamics at these length scales are nonlinear and uncertain, while accurate, real-time feedback

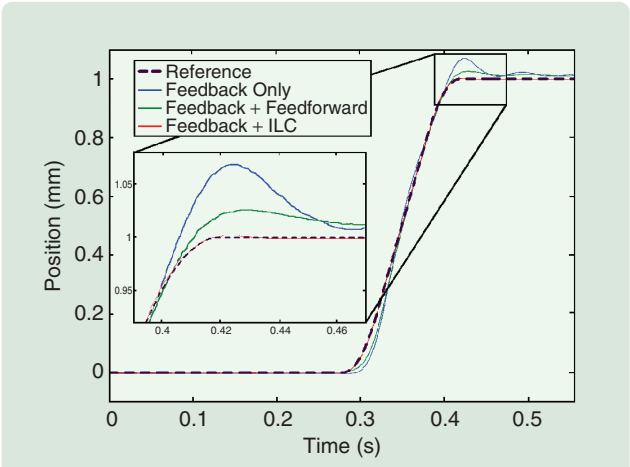

FIGURE 10 Tracking results with and without ILC on the μ-RD system. The ILC is shown after 50 learning iterations, outperforming the reference feedforward controller. Note that the ILC response to the changing velocity is immediate, whereas the feedback and reference feedforward controllers show a lagged response. The improved tracking provides lower overshoot and shorter settling time.

becomes more challenging as length scales decrease. For these applications, where precision levels are very high ($\leq$1 nm), high bandwidth feedforward control becomes necessary. The ability of ILC to use available measurements in an offline learning manner, coupled with feedback controller robustness, enables an effective tracking control solution.

ACKNOWLEDGMENTS

The authors would like to thank the National Science Foundation (NSF), for support under DMI-0140466, an NSF Graduate Research Fellowship, and the University of Illinois at Urbana-Champaign Nano-CEMMS Center NSF Award 0328162.

AUTHOR INFORMATION

Douglas A. Bristow received his B.S. from the University of Missouri at Rolla in 2001 and his M.S. from the University of Illinois at Urbana-Champaign in 2003, both in mechanical engineering. He is currently a Ph.D. candidate in mechanical engineering at the University of Illinois. In 2002 he was awarded the National Science Foundation Graduate Fellowship. His research interests include micro- and nano-manufacturing systems and iterative learning control.

Marina Tharayil received the B.S. degree in mechanical engineering from the University of Illinois, Chicago, in 1999. She received her M.S. (2001) and Ph.D. (2005) in mechanical engineering from the University of Illinois at Urbana-Champaign, where she was a National Science Foundation Graduate Fellow. She is currently employed as a research engineer at the Xerox Wilson Center for Research and Technology in Webster, New York. Her research interests include repetitive and iterative learning control.

Andrew G. Alleyne (alleyne@uiuc.edu) received his B.S.E. degree from Princeton University in mechanical and aerospace engineering in 1989. He received his M.S. and Ph.D. degrees in mechanical engineering in 1992 and 1994, respectively, from the University of California at Berkeley. He joined the Department of Mechanical and Industrial Engineering at the University of Illinois at Urbana-Champaign (UIUC) in 1994 and is also appointed in the Coordinated Science Laboratory of UIUC. He currently holds the Ralph M. and Catherine V. Fisher Professorship in the College of Engineering. His research interests are a mix of theory and implementation with a broad application focus. He was awarded the ASME Dynamics Systems and Control Division's Outstanding Young Investigator Award in 2003 and was a Fulbright Fellow to the Netherlands in 2003 where he held a visiting professorship in vehicle mechatronics at TU Delft. He is an editor of *Vehicle System Dynamics* and an associate editor of *IEEE Control Systems Magazine*. He is a Fellow of the ASME and a Senior Member of the IEEE. He can be contacted at the University of Illinois, 1206 West Green St., MC-244, Urbana, IL 61801 USA.

REFERENCES

[1] K.L. Moore, *Iterative Learning Control for Deterministic Systems*. London: Springer-Verlag, 1993.

[2] K.J. Hunt, D. Sbarbaro, R. Zbikowski, and P.J. Gawthrop, "Neural networks for control systems—A survey," *Automatica*, vol. 28, no. 6, pp. 1083–112, 1992.

[3] G. Hillerstrom and K. Walgama, "Repetitive control theory and applications—a survey," in *Proc. 13th World Congress Vol.D: Control Design II, Optimization*, 1997, pp. 1–6.

[4] R.W. Longman, "Iterative learning control and repetitive control for engineering practice," *Int. J. Contr.*, vol. 73, no. 10, pp. 930–954, 2000.

[5] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *J. Robot. Syst.*, vol. 1, pp. 123–140, 1984.

[6] D.A. Bristow and A.G. Alleyne, "A manufacturing system for microscale robotic deposition," in *Proc. Ame. Contr. Conf.*, 2003, pp. 2620–2625.

[7] H. Elci, R.W. Longman, M. Phan, J.-N. Juang, and R. Ugoletti, "Discrete frequency based learning control for precision motion control," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 1994, pp. 2767–2773.

[8] W. Messner, R. Horowitz, W.-W. Kao, and M. Boals, "A new adaptive learning rule," *IEEE Trans. Automat. Contr.*, vol. 36, no. 2, pp. 188–197, 1991.

[9] M. Norrlof, "An adaptive iterative learning control algorithm with experiments on an industrial robot," *IEEE Trans. Robot. Automat.*, vol. 18, no. 2, pp. 245–251, 2002.

[10] D.-I. Kim and S. Kim, "An iterative learning control method with application for CNC machine tools," *IEEE Trans. Ind. Applicat.*, vol. 32, no. 1, pp. 66–72, 1996.

[11] D. de Roover and O.H. Bosgra, "Synthesis of robust multivariable iterative learning controllers with application to a wafer stage motion system," *Int. J. Contr.*, vol. 73, no. 10, pp. 968–979, 2000.

[12] H. Havlicsek and A. Alleyne, "Nonlinear control of an electrohydraulic injection molding machine via iterative adaptive learning," *IEEE/ASME Trans. Mechatron.*, vol. 4, no. 3, pp. 312–323, 1999.

[13] F. Gao, Y. Yang, and C. Shao, "Robust iterative learning control with applications to injection molding process," *Chem. Eng. Sci.*, vol. 56, no. 24, pp. 7025–7034, 2001.

[14] M. Pandit and K.-H. Buchheit, "Optimizing iterative learning control of cyclic production processes with application to extruders," *IEEE Trans. Contr. Syst. Technol.*, vol. 7, no. 3, pp. 382–390, 1999.

[15] S. Garimella and K. Srinivasan, "Application of iterative learning control to coil-to-coil control in rolling," *IEEE Trans. Contr. Syst. Technol.*, vol. 6, no. 2, pp. 281–293, 1998.

[16] S.A. Saab, "A stochastic iterative learning control algorithm with application to an induction motor," *Int. J. Contr.*, vol. 77, no. 2, pp. 144–163, 2004.

[17] A.D. Barton, P.L. Lewin, and D.J. Brown, "Practical implementation of a real-time iterative learning position controller," *Int. J. Contr.*, vol. 73, no. 10, pp. 992–999, 2000.

[18] W. Hoffmann, K. Peterson, and A.G. Stefanopoulou, "Iterative learning control for soft landing of electromechanical valve actuator in camless engines," *IEEE Trans. Contr. Syst. Technol.*, vol. 11, no. 2, pp. 174–184, 2003.

[19] Y.Q. Chen and K.L. Moore, "A practical iterative learning path-following control of an omni-directional vehicle," *Asian J. Contr.*, vol. 4, no. 1, pp. 90–98, 2002.

[20] C. Mi, H. Lin, and Y. Zhang, "Iterative learning control of antilock braking of electric and hybrid vehicles," *IEEE Trans. Veh. Technol.*, vol. 54, no. 2, pp. 486–494, 2005.

[21] D.R. Yang, K.S. Lee, H.J. Ahn, and J.H. Lee, "Experimental application of a quadratic optimal iterative learning control method for control of wafer temperature uniformity in rapid thermal processing," *IEEE Trans. Semiconduct. Manufact.*, vol. 16, no. 1, pp. 36–44, 2003.

[22] D. Gorinevsky, "Loop shaping for iterative control of batch processes," *IEEE Contr. Syst. Mag.*, vol. 22, no. 6, pp. 55–65, 2002.

[23] M. Mezghani, G. Roux, M. Cabassud, M.V. Le Lann, B. Dahhou, and G. Casamatta, "Application of iterative learning control to an exothermic semibatch chemical reactor," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 6, pp. 822–834, 2002.

[24] S. Kawamura and N. Sakagami, "Analysis on dynamics of underwater robot manipulators basing on iterative learning control and time-scale transformation," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 1088–1094, 2002.

[25] C.V. Giessen, Q. Zou, and S. Devasia, "Inversion-based precision-positioning of inertial reaction devices," in *Proc. Amer. Contr. Conf.*, 2004, pp. 3788–3793.

[26] Y. Chen, C. Wen, J.-X. Xu, and M. Sun, "High-order iterative learning identification of projectile's aerodynamic drag coefficient curve from radar measured velocity data," *IEEE Tran. Contr. Syst. Technol.*, vol. 6, no. 4, pp. 563–570, 1998.
[27] C.T. Abdallah, V.S. Soulian, and E. Schamiloglu, "Toward "smart tubes" using iterative learning control," *IEEE Trans. Plasma Sci.*, vol. 26, no. 3, pp. 905–911, 1998.
[28] M. Garden. "Learning control of actuators in control systems," U.S. Patent 3555252, 1971.
[29] M. Uchiyama, "Formation of high-speed motion pattern of a mechanical arm by trial," *Trans. Soc. Instrument Contr. Engineers*, vol. 14, no. 6, pp. 706–712, 1978.
[30] J.J. Craig, "Adaptive control of manipulators through repeated trials," in *Proc. Amer. Contr. Conf*, 1984., pp. 1566–1573.
[31] G. Casalino and G. Bartolini, "A learning procedure for the control of movements of robotic manipulators," in *Proc. IASTED Symp. Robot. Automat.*, 1984, pp. 108–111.
[32] S. Kawamura, F. Miyazaki, and S. Arimoto, "Iterative learning control for robotic systems," in *Proc. Int. Conf. Ind. Electron., Contr. and Instrum*, 1984., pp. 393–398.
[33] K.L. Moore and J.-X. Xu, "Editorial: Iterative learning control," *Int. J. Contr.*, vol. 73, no. 10, 2000.
[34] "Iterative learning control" *Asian J. Contr.*, vol. 4, no. 1, 2002.
[35] Z. Bien and J.-X. Xu, *Iterative Learning Control: Analysis, Design, Integration and Applications*. Boston: Kluwer, 1998.
[36] Y. Chen and C. Wen, *Iterative Learning Control: Convergence, Robustness, and Applications*. London: Springer, 1999.
[37] J.-X. Xu and Y. Tan, *Linear and Nonlinear Iterative Learning Control*. Berlin: Springer, 2003.
[38] K.L. Moore, M. Dahleh, and S.P. Bhattacharyya, "Iterative learning control: A survey and new results," *J. Robot. Syst.*, vol. 9, no. 5, pp. 563–594, 1992.
[39] R. Horowitz, "Learning control of robot manipulators," *Trans. ASME J. Dyn. Syst. Meas. Control*, vol. 115, no. 2B, pp. 402–411, 1993.
[40] M. Norrlof and S. Gunnarsson, "Experimental comparison of some classical iterative learning control algorithms," *IEEE Trans. Robot. Automat.*, vol. 18, no. 4, pp. 636–641, 2002.
[41] T. Kavli, "Frequency domain synthesis of trajectory learning controllers for robot manipulators," *J. Robot. Syst.*, vol. 9, no. 5, pp. 663–680, 1992.
[42] J.H. Lee, K.S. Lee, and W.C. Kim, "Model-based iterative learning control with a quadratic criterion for time-varying linear systems," *Automatica*, vol. 36, no. 5, pp. 641–657, 2000.
[43] H. Elci, R.W. Longman, M.Q. Phan, J.-N. Juang, and R. Ugoletti, "Simple learning control made practical by zero-phase filtering: Applications to robotics," *IEEE Trans. Circuits Syst. I: Fundamental Theory Applicat.*, vol. 49, no. 6, pp. 753–767, 2002.
[44] M. Norrlof and S. Gunnarsson, "Time and frequency domain convergence properties in iterative learning control," *Int. J. Contr.*, vol. 75, no. 14, pp. 1114–1126, 2002.
[45] Y. Chen and K.L. Moore, "An optimal design of PD-type iterative learning control with monotonic convergence," in *Proc. IEEE Int. Symp. Intelligent Contr.*, 2002, pp. 55–60.
[46] K.L. Moore, Y. Chen, and V. Bahl, "Monotonically convergent iterative learning control for linear discrete-time systems," *Automatica*, vol. 41, no. 9, pp. 1529–1537, 2005.
[47] H.-S. Lee and Z. Bien, "A note on convergence property of iterative learning controller with respect to sup norm," *Automatica*, vol. 33, no. 8, pp. 1591–1593, 1997.
[48] D.H. Owens and G.S. Munde, "Universal adaptive iterative learning control," in *Proc. IEEE Conf. Decision Contr.*, 1998, pp. 181–185.
[49] Y. Chen, C. Wen, and M. Sun, "Robust high-order P-type iterative learning controller using current iteration tracking error," *Int. J. Contr.*, vol. 68, no. 2, pp. 331–342, 1997.
[50] Y. Chen, Z. Gong, and C. Wen, "Analysis of a high-order iterative learning control algorithm for uncertain nonlinear systems with state delays," *Automatica*, vol. 34, no. 3, pp. 345–353, 1998.
[51] D.H. Owens and K. Feng, "Parameter optimization in iterative learning control," *Int. J. Contr.*, vol. 76, no. 11, pp. 1059–1069, 2003.
[52] N. Amann, D.H. Owens, E. Rogers, and A. Wahl, "An $\mathcal{H}_\infty$ approach to linear iterative learning control design," *Int. J. Adaptive Contr. Signal Processing*, vol. 10, no. 6, pp. 767–781, 1996.
[53] J.-X. Xu, X.-W. Wang, and L.T. Heng, "Analysis of continuous iterative learning control systems using current cycle feedback," in *Proc. Amer. Contr. Conf.*, 1995, pp. 4221–4225.
[54] T.-Y. Doh, J.-H. Moon, K.B. Jin, and M.J. Chung, "Robust iterative learning control with current feedback for uncertain linear systems," *Int. J. Syst. Sci.*, vol. 30, no. 1, pp. 39–47, 1999.
[55] D.H. Owens and G. Munde, "Error convergence in an adaptive iterative learning controller," *Int. J. Contr.*, vol. 73, no. 10, pp. 851–857, 2000.
[56] C.J. Goh and W.Y. Yan, "An $\mathcal{H}_\infty$ synthesis of robust current error feedback learning control," *J. Dyn. Syst. Meas. Control*, vol. 118, no. 2, pp. 341–346, 1996.
[57] M.Q. Phan, R.W. Longman, and K.L. Moore, "Unified formulation of linear iterative learning control," *Adv. Astronautical Sci.*, vol. 105, pp. 93–111, 2000.
[58] C.-T. Chen, *Linear System Theory and Design*. New York: Oxford Univ. Press, 1999.
[59] U. Grenander and G. Szego, *Toeplitz Forms and their Applications*. Berkeley, CA: Univ. of California Press, 1958.
[60] J.A. Frueh and M.Q. Phan, "Linear quadratic optimal learning control (LQL)," *Int. J. Contr.*, vol. 73, no. 10, pp. 832–839, 2000.
[61] E. Rogers and D.H. Owens, *Stability Analysis for Linear Repetitive Processes*. Berlin: Springer-Verlag, 1992.
[62] D.H. Owens, E. Rogers, and K.L. Moore, "Analysis of linear iterative learning control schemes using repetitive process theory," *Asian J. Contr.*, vol. 4, no. 1, pp. 68–89, 2002.
[63] D.H. Owens and E. Rogers, "Comments on 'On the equivalence of causal LTI iterative learning control and feedback control'," *Automatica*, vol. 40, no. 5, pp. 895–898, 2004.
[64] P.B. Goldsmith, "Author's reply to "Comments on 'On the equivalence of causal LTI iterative learning control and feedback control'"," *Automatica*, vol. 40, no. 5, pp. 899–900, 2004.
[65] E. Rogers, K. Galkowski, A. Gramacki, J. Gramacki, and D.H. Owens, "Stability and controllability of a class of 2-D linear systems with dynamic boundary conditions," *IEEE Trans. Circuits Syst. I: Fundamental Theory and Applicat.*, vol. 49, no. 2, pp. 181–195, 2002.
[66] Z. Geng, J.D. Lee, R.L. Carroll, and L.H. Haynes, "Learning control system design based on 2-D theory—An application to parallel link manipulator," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1990, pp. 1510–1515.
[67] J.E. Kurek and M.B. Zaremba, "Iterative learning control synthesis based on 2-D system theory," *IEEE Trans. Autom. Contr.*, vol. 38, no. 1, pp. 121–125, 1993.
[68] Y. Fang and T.W.S. Chow, "2-D analysis for iterative learning controller for discrete-time systems with variable initial conditions," *IEEE Trans. on Circuits Syst. I: Fundamental Theory and Applica.t.*, vol. 50, no. 5, pp. 722–727, 2003.
[69] B. Bukkems, D. Kostic, B. de Jager, and M. Steinbuch, "Learning-based identification and iterative learning control of direct-drive robots," *IEEE Trans. Contr. Syst. Technol.*, vol. 13, no. 4, pp. 537–549, 2005.
[70] D. de Roover, "Synthesis of a robust iterative learning controller using an $\mathcal{H}_\infty$ approach," in *Proc. 35th IEEE Conf. Decision Contr.*, 1996, pp. 3044–3049.
[71] Y.-C. Huang and R.W. Longman, "Source of the often observed property of initial convergence followed by divergence in learning and repetitive control," *Advances Astronaut. Sci.*, vol. 90, no. 1, pp. 555–572, 1996.
[72] R.W. Longman and Y.-C. Huang, "The phenomenon of apparent convergence followed by divergence in learning and repetitive control," *Intell. Automat. Soft Comput.*, vol. 8, no. 2, pp. 107–128, 2002.
[73] A. Sala and P. Albertos, "Open-loop iterative learning control," in *Iterative Identification and Control*, A. Sala and P. Albertos, Eds. London: Springer, 2002.
[74] Q. Hu, J.-X. Xu, and T.H. Lee, "Iterative learning control design for smith predictor," *Syst. Contr. Lett.*, vol. 44, no. 3, pp. 201–210, 2001.
[75] J.-X. Xu, T.H. Lee, J. Xu, Q. Hu, and S. Yamamoto, "Iterative learning control with smith time delay compensator for batch processes," in *Proc. Amer. Contr. Conf.*, 2001, pp. 1972–1977.
[76] M. Sun and D. Wang, "Iterative learning control design for uncertain dynamic systems with delayed states," *Dynamics Control*, vol. 10, no. 4, pp. 341–357, 2000.
[77] K.-H. Park, Z. Bien, and D.-H. Hwang, "Design of an iterative learning controller for a class of linear dynamic systems with time delay," in *IEE Proc.: Contr. Theory Applicat.*, vol. 145, no. 6, pp. 507–512, 1998.
[78] M. Norrlof, "Iterative learning control: Analysis, design, and experiments," Ph.D. dissertation, Linkoping Studies Sci. Technol., Linkopings Universitet, Linkoping, Sweden, 2000.
[79] S.S. Saab, "A discrete-time stochastic learning control algorithm," *IEEE Trans. Automat. Contr.*, vol. 46, no. 6, pp. 877–887, 2001.

[80] S.S. Saab, "On a discrete-time stochastic learning control algorithm," *IEEE Trans. Automat. Contr.*, vol. 46, no. 8, pp. 1333–1336, 2001.
[81] S.S. Saab, "Stochastic P-type/D-type iterative learning control algorithms," *Int. J. Contr.*, vol. 76, no. 2, pp. 139–148, 2003.
[82] H.-S. Lee and Z. Bien, "Study on robustness of iterative learning control with non-zero initial error," *Int. J. Contr.*, vol. 64, no. 3, pp. 345–359, 1996.
[83] K.-H. Park, Z. Bien, and D.-H. Hwang, "Study on the robustness of a PID-type iterative learning controller against initial state error," *Int. J. Syst. Sci.*, vol. 30, no. 1, pp. 49–59, 1999.
[84] K.-H. Park and Z. Bien, "A generalized iterative learning controller against initial state error," *Int. J. Contr.*, vol. 73, no. 10, pp. 871–881, 2000.
[85] M. Sun and D. Wang, "Iterative learning control with initial rectifying action," *Automatica*, vol. 38, no. 7, pp. 1177–82, 2002.
[86] G. Heinzinger, D. Fenwick, B. Paden, and F. Miyazaki, "Stability of learning control with disturbances and uncertain initial conditions," *IEEE Trans. Automat. Contr.*, vol. 37, no. 1, pp. 110–14, 1992.
[87] S. Arimoto, "Mathematical theory of learning with applications to robot control," in *Proc. Adaptive and Learning Systems: Theory and Applications*, 1986, pp. 388–379.
[88] J.E. Hauser, "Learning control for a class of nonlinear systems," in *Proc. 26th IEEE Conf. Decision Contr.*, 1987, pp. 859–860.
[89] C.-J. Chien and J.-S. Liu, "P-type iterative learning controller for robust output tracking of nonlinear time-varying systems," *Int. J. Contr.*, vol. 64, no. 2, pp. 319–334, 1996.
[90] C.-C. Cheah and D. Wang, "Learning impedance control for robotic manipulators," *IEEE Trans. Robot. Automat.*, vol. 14, no. 3, pp. 452–465, 1998.
[91] D. Wang, "On D-type and P-type ILC designs and anticipatory approach," *Int. J. Contr.*, vol. 73, no. 10, pp. 890–901, 2000.
[92] J.-X. Xu and Y. Tan, "Robust optimal design and convergence properties analysis of iterative learning control approaches," *Automatica*, vol. 38, no. 11, pp. 1867–1880, 2002.
[93] K.L. Moore, "An observation about monotonic convergence in discrete-time, P-type iterative learning control," in *Proc. IEEE Int. Symp. Intell. Contr.*, 2001, pp. 45–49.
[94] H.-S. Lee and Z. Bien, "Robustness and convergence of a PD-type iterative learning controller," in *Iterative Learning Control: Analysis, Design, Integration and Applications*, Z. Bien and J.-X. Xu, Eds., Boston: Kluwer, 1998.
[95] G.F. Franklin, J.D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems.* New Jersey: Prentice Hall, 2002.
[96] K. Kinosita, T. Sogo, and N. Adachi, "Iterative learning control using adjoint systems and stable inversion," *Asian J. Contr.*, vol. 4, no. 1, pp. 60–67, 2002.
[97] T. Sogo, "Stable inversion for nonminimum phase sampled-data systems and its relation with the continuous-time counterpart," in *Proc. 41st IEEE Conf. Decision Contr.*, 2002, pp. 3730–3735.
[98] J. Ghosh and B. Paden, "Pseudo-inverse based iterative learning control for nonlinear plants with disturbances," in *Proc. 38th IEEE Conf. Decision Contr.*, 1999, pp. 5206–5212.
[99] K.S. Lee, S.H. Bang, and K.S. Chang, "Feedback-assisted iterative learning control based on an inverse process model," *J. Process Contr.*, vol. 4, no. 2, pp. 77–89, 1994.
[100] N. Amann, D.H. Owens, and E. Rogers, "Iterative learning control for discrete-time systems with exponential rate of convergence," *IEE Proc.: Control Theory Applicat.*, vol. 143, no. 2, pp. 217–224, 1996.
[101] S. Gunnarsson and M. Norrlof, "On the design of ILC algorithms using optimization," *Automatica*, vol. 37, no. 12, pp. 2011–2016, 2001.
[102] N. Amann, D.H. Owens, and E. Rogers, "Iterative learning control for discrete time systems using optimal feedback and feedforward actions," in *Proc. 34th IEEE Conf. Decision Contr.*, 1995, pp. 1696–1701.
[103] N. Amann, D.H. Owens, and E. Rogers, "Predictive optimal iterative learning control," *Int. J. Contr.*, vol. 69, no. 2, pp. 203–226, 1998.
[104] R. Tousain, E. Van Der Meche, and O. Bosgra, "Design strategy for iterative learning control based on optimal control," in *Proc. 40th IEEE Conf. Decision Contr.*, 2001, pp. 4463–4468.
[105] B.G. Dijkstra and O.H. Bosgra, "Convergence design considerations of low order Q-ILC for closed loop systems, implemented on a high precision wafer stage," in *Proc. 41st IEEE Conf. Decision Contr.*, 2002, pp. 2494–2499.
[106] B.G. Dijkstra and O.H. Bosgra, "Extrapolation of optimal lifted system ILC solution, with application to a waferstage," in *Proc. Amer. Contr. Conf.*, 2002, pp. 2595–2600.
[107] K. Fujimoto and T. Sugie, "Iterative learning control of Hamiltonian systems: I/O based optimal control approach," *IEEE Trans. Automat. Contr.*, vol. 48, no. 10, pp. 1756–1761, 2003.
[108] D. Bristow, A. Alleyne, and D. Zheng, "Control of a microscale deposition robot using a new adaptive time-frequency filtered iterative learning control,"in *Proc. Amer. Contr. Conf.*, 2004, pp. 5144–5149.
[109] J. Cesarano, R. Segalman, and P. Calvert, "Robocasting provides moldless fabrication from slurry deposition," *Ceram. Industry*, pp. 94–102, 1998.
[110] Q. Li and J.A. Lewis, "Nanoparticle inks for directed assembly of three-dimensional periodic structures," *Adv. Mater.*, vol. 15, no. 19, pp. 1639–1643, 2003.
[111] P.B. Goldsmith, "On the equivalence of causal LTI iterative learning control and feedback control," *Automatica*, vol. 38, no. 4, pp. 703–708, 2002.
[112] P.B. Goldsmith, "The fallacy of causal iterative learning control,"in *Proc. 40th IEEE Conf. Decision Contr.*, 2001, pp. 4475–4480.
[113] F. Padieu and R. Su, "$\mathcal{H}_\infty$ approach to learning control systems," *Int. J. Adaptive Contr. Signal Processing*, vol. 4, no. 6, pp. 465–474, 1990.
[114] C.J. Goh, "A frequency domain analysis of learning control," *Trans. ASME J. Dyn. Syst. Meas. Control*, vol. 116, no. 4, pp. 781–786, 1994.
[115] L.M. Hideg, "Stability of linear time varying multiple input multiple output continuous time learning control systems: A sufficient condition," in *Proc. 1994 IEEE Int. Symp. Intell. Contr.*, Aug. 16–18 1994, 1994, pp. 285–290.
[116] M. Togai and O. Yamano, "Analysis and design of an optimal learning control scheme for industrial robots: A discrete system approach," in *Proc. 24th IEEE Conf. Decision Contr.*, 1985, pp. 1399–1404.
[117] T.-Y. Kuc, J.S. Lee, and K. Nam, "Iterative learning control theory for a class of nonlinear dynamic systems," *Automatica*, vol. 28, no. 6, pp. 1215–1221, 1992.
[118] Y.A. Jiang, D.J. Clements, and T. Hesketh, "Betterment learning control of nonlinear systems," in *Proc. IEEE Conf. Decision Contr.*, 1995, pp. 1702–1707.
[119] T.-J. Jang, C.-H. Choi, and H.-S. Ahn, "Iterative learning control in feedback systems," *Automatica*, vol. 31, no. 2, pp. 243–248, 1995.
[120] J. Ghosh and B. Paden, "A pseudoinverse-based iterative learning control," *IEEE Trans. Automat. Contr.*, vol. 47, no. 5, pp. 831–837, 2002.
[121] R. Horowitz, W. Messner, and J.B. Moore, "Exponential convergence of a learning controller for robot manipulators," *IEEE Trans. Automat. Contr.*, vol. 36, no. 7, pp. 890–894, 1991.
[122] T.H. Lee, J.H. Nie, and W.K. Tan, "Developments in learning control enhancements for nonlinear servomechanisms," *Mechatronics*, vol. 5, no. 8, pp. 919–936, 1995.
[123] J.-X. Xu, B. Viswanathan, and Z. Qu, "Robust learning control for robotic manipulators with an extension to a class of non-linear systems," *Int. J. Contr.*, vol. 73, no. 10, pp. 858–870, 2000.
[124] T. Sugie and T. Ono, "Iterative learning control law for dynamical systems," *Automatica*, vol. 27, no. 4, pp. 729–732, 1991.
[125] T.-J. Jang, H.-S. Ahn, and C.-H. Choi, "Iterative learning control for discrete-time nonlinear systems," *Int. J. Syst. Sci.*, vol. 25, no. 7, pp. 1179–1189, 1994.
[126] S.S. Saab, "Discrete-time learning control algorithm for a class of nonlinear systems," in *Proc. Amer. Contr. Conf.*, 1995, pp. 2739–2743.
[127] J.-X. Xu, "Analysis of iterative learning control for a class of nonlinear discrete-time systems," *Automatica*, vol. 33, no. 10, pp. 1905–1907, 1997.
[128] D. Wang, "Convergence and robustness of discrete time nonlinear systems with iterative learning control," *Automatica*, vol. 34, no. 11, pp. 1445–1448, 1998.
[129] C.-J. Chien, "A discrete iterative learning control for a class of nonlinear time-varying systems," *IEEE Trans. Automat. Contr.*, vol. 43, no. 5, pp. 748–752, 1998.
[130] M. Sun and D. Wang, "Robust discrete-time iterative learning control: Initial shift problem," in *Proc. IEEE Conf. Decision Contr.*, 2001, pp. 1211–1216.
[131] M. Sun and D. Wang, "Analysis of nonlinear discrete-time systems with higher-order iterative learning control," *Dynamics Control*, vol. 11, no. 1, pp. 81–96, 2001.
[132] Y.-T. Kim, H. Lee, H.-S. Noh, and Z.Z. Bien, "Robust higher-order iterative learning control for a class of nonlinear discrete-time systems," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2003, pp. 2219–2224.
[133] D.-Y. Pi and K. Panaliappan, "Robustness of discrete nonlinear systems with open-closed-loop iterative learning control," in *Proc. 2002 Int. Conf. Machine Learning Cybern.*, 2002, pp. 1263–1266.
[134] M. Sun and D. Wang, "Initial shift issues on discrete-time iterative learning control with system relative degree," *IEEE Trans. Automat. Contr.*, vol. 48, no. 1, pp. 144–148, 2003.
[135] R. Longman, "Designing iterative learning and repetitive controllers," in *Iterative Learning Control: Analysis, Design, Integration and Applications*, Z. Bien and J.-X. Xu, Eds. Boston: Kluwer, 1998.