2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Iterative Learning Control — Algorithms, Applications and Future Research Directions

Eric Rogers[a], Bing Chu[a], Kevin Moore[b], Tom Oomen[c], Ying Tan[d]

Abstract— **This paper gives a tutorial on iterative learning control nearly five decades after what is widely regarded as the first substantive paper in the literature. The focus is on algorithm development under a number of general headings (linear, optimization, frequency domain, and nonlinear), together with supporting experimental validation/industrial applications and also applications in healthcare.**

I. Introduction

Iterative learning control (ILC) applies to systems that complete the same finite duration operation over and over again, with resetting to the starting location at the end of each operation, or a stoppage time between one operation and the start of the next. In the literature various terms are used to denote each operation, e.g., trial, pass and iteration, but this paper will only use the term trial and the duration of the trial will be termed the trial length. The unique feature of ILC is that once a trial is complete all information generated over the trial length is available for use in constructing the control input for the subsequent trial, with the aim of improving performance from trial-to-trial.

An example to illustrate how ILC can be applied is a robot undertaking a pick and place task, where the operations are i) collect the payload from a specified location, ii) transfer it over a finite duration, iii) place the payload at a specified endpoint location or under synchronization onto a moving conveyor (for example), iv) return to the starting location, and v) repeat i)-iv). In application areas, such as manufacturing, there will be requirements for high accuracy and completing the maximum possible number of trials before a halt is needed because system errors cause a loss of synchronization, resulting in, e.g., failure to place the payload on the moving conveyor. To control this system, a starting point is to specify a reference trajectory between start and end that represents the ideal path for the robot to follow on each trial, and then many design methods are possible.

The ILC literature typically uses a subscript on variables representing signals or operators to index the trial number, representing each execution of the task. Thus, for continuous-time variables one notation is of the form $y_k(t)$, $0 \leq t \leq \alpha$, $k \geq 1$, where y is the vector or scalar valued variable under consideration, $\alpha < \infty$ is the trial length and the integer $k \geq 1$ denotes the trial number (or index). In the case of discrete dynamics, the form is $y_k(p)$, $0 \leq p \leq N-1$, $k \geq 1$ where the integer N denotes the number of samples along a trial (N times the sampling period gives the trial length). (Other notation will be used in parts of this paper.)

Once a trial, say k, is completed, then all values of variables generated on trial $k-1$ over the complete trial length is, at the cost of storage, available for use in constructing the control input for the subsequent trial. (More generally information from a finite number, say M, of previous trial data could be used, and the resulting ILC law is termed higher order.) The core question is: how to make use of this previous trial data?

In the literature, [1] used such data to enhance the performance of robotic motion. Let $r(t)$, $0 \leq t \leq \alpha$, denote the specified reference trajectory, most often directly specified for the application and not changing from one trial to the next. (In some work the reference trajectory is assumed to be generated as the output of a stable system and it is also possible to consider switching, i.e., complete a number of trials with one reference trajectory and then switch to another. Analysis for these cases is returned to later in the paper.) On trial k the error $e_k(t) = r(t) - y_k(t)$, $0 \leq t \leq \alpha$, $k \geq 1$, where $y_k(t)$ is the system output. Hence, the sequence of errors $\{e_k(t)\}_k$ can be formed and the convergence of this sequence in k is of critical interest (performance in t is also of interest).

The results in [1] did not directly address these critical issues. Instead, a rigorous treatment of ILC convergence analysis and design was first reported in [2] (with simultaneous and independently-derived results in [3]). This analysis assumed that i) the reference trajectory was specified, ii) resetting at the end of each trial is to the same initial state vector, iii) the system dynamics are invariant throughout the trials, iv) the error on, say, trial k is used in forming the input for trial $k+1$, and v) the system dynamics are invertible in the sense that for a specified $r(t)$ (assumed to have a piecewise continuous derivative) there exists a unique input that produces the output $r(t)$. These assumptions will be considered below after an example is given to highlight the structure of ILC laws. However, we comment that much of the research effort in ILC has been aimed at relaxation of these various assumptions, including allowing for non-uniform reset conditions, varying system dynamics (in time and in trial), trial-varying reference signals, noise and stochastic dynamics, and various forms of robustness and optimal performance analysis, and more.

[a] School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK; etar(b.chu)@ecs.soton.ac.uk
[b] Department of Electrical Engineering, Colorado School of Mines, USA; kmoore@mines.edu
[c] Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands and Department of Mechanical Engineering, Delft University of Technology, The Netherlands; T.A.E.Oomen@tue.nl.
[d] Department of Mechanical Engineering, University of Melbourne, Australia; yingt@unimelb.edu.au.

In the case of discrete dynamics Fig. 1 shows a schematic of the application of an ILC law. This figure also shows that ILC has the structure of a two-dimensional (2D) system, i.e., information propagation from trial-to-trial (in k) and along-the-trial (in p). The systems theory for 2D systems is one setting for ILC analysis and design with a follow through to experimental validation and nonlinear dynamics. In particular, the stability theory for linear and nonlinear repetitive processes is a very powerful setting for design.

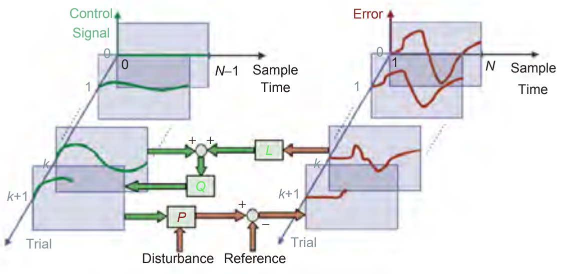


Fig. 1. The 2D structure of ILC.

Essentially, ILC design and application is critically dependent on how the previous trial information is exploited. In [2] one form of ILC law considered for continuous time dynamics has the structure

$$u_{k+1}(t) = u_k(t) + \Gamma \dot{e}_k(t) \tag{1}$$

where $u_k(t)$ is the system input, $e_k(t)$ is the error on trial k, Γ is a constant to be designed and in this paper the dot notation is used to represent differentiation with respect to time. Immediately it is seen that ILC designs the input, which is a signal, instead of a controller, which is a system. At first sight, this ILC law appears to be non-causal, which is a critical feature of ILC. The law actually acts on past data, the resetting of the initial conditions at the start of each trial allows 'non causal' processing of the errors from the previous trial. One immediate consequence is that zero-phase filtering, which is non-causal, can be applied to enable high frequency attenuation without introducing any lag.

Suppose the system to be controlled, denoted by the state space triple $\{A, B, C\}$, is defined as

$$\begin{aligned} \dot{x}_k(t) &= Ax_k(t) + Bu_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned}$$

In [2] it is shown that the ILC law (1) for such a system with the same number of inputs and outputs ensures convergence (in k) of the error sequence $\{e_k(t)\}_k$ if

$$||I - CB\Gamma|| < 1 \tag{2}$$

where I denotes the identity matrix of compatible dimensions and $||\cdot||$ denotes the norm on the underlying function space. This result shows that trial-to-trial error convergence can occur for systems with an unstable state matrix (A), due to the finite trial length. Also, convergence cannot occur for systems whose first Markov parameter (CB) is zero. In the first of these cases, feedback action must be introduced to stabilize the state dynamics, resulting in an ILC law with a feedback (in t) and feedforward (in k) structure. This structure for the ILC law can be designed and applied in a number of different ways as seen later in this paper. Similarly, the first Markov parameter problem can be overcome (see below).

For discrete dynamics, let q denote the time forward time shift operator acting on a signal, say $h(p)$, as $qh(p) \equiv h(p+1)$. Then, in the single-input, single-output (SISO) case for simplicity, the dynamics in the ILC setting can be written as

$$y_k(p) = G(q)u_k(p) + d(p) \tag{3}$$

where $G(q)$ is a proper rational function of q with relative degree m and $d(q)$ is an exogenous that repeats on each trial. In the particular case of systems described by the state space triple $\{A, B, C\}$ we have

$$G(q) = C(qI - A)^{-1}B, \;\; d(p) = CA^p x(0)$$

It is also commonly assumed that $G(q)$ is stable (if not then a stabilizing controller must first be designed and ILC applied to the resulting dynamics).

An extensively used ILC law, see, e.g., the survey papers [4], [5] for references to the early work is

$$u_{k+1}(p) = Q(q)[u_k(p) + L(q)e_k(p+1)] \tag{4}$$

where $Q(q)$ and $L(q)$ are referred to as the Q-filter and learning function, respectively. Current trial ILC extends the last law to incorporate feedback. The structure is

$$u_{k+1}(p) = Q(q)[u_k(p) + L(q)e_k(p+1)] + C(q)e_{k+1}(p) \tag{5}$$

where the last entry on the right-hand side is the current trial error feedback term. The design of an ILC law of this form is considered again in the 2D systems setting in the paper.

A particular case of (5) is

$$u_{k+1}(p) = u_k(p) + h_p e_k(p+\lambda) \tag{6}$$

where $\lambda > 0$, h_p is a proportional gain term to be selected and this law is known as phase lead ILC. It is also possible to specify a discrete time PD-type laws as

$$u_{k+1}(p) = u_k(p) + h_p e_k(p+1) + h_d[e_k(p+1) - e_k(p)] \tag{7}$$

where h_d is the derivative gain. Moreover, many other simple structure ILC laws can be defined. Indeed, in Arimoto's early work he went so far as to define a full-on proportional-integral-derivative (PID) ILC control law for continuous-time systems given by:

$$u_{k+1}(t) = u_k(t) + \Phi e_k(t) + \Gamma \dot{e}_k(t) + \Psi \int_0^\alpha e_k(\tau)d\tau$$

Such laws and other ILC algorithms have been extensively developed applied, see, e.g., the book [6], which, in turn, cites the original work.

The PD-type and related relatively simple structure ILC laws have been extensively studied from the standpoint of developing tuning rules that do not need an accurate model of the dynamics for design and implementation. A point

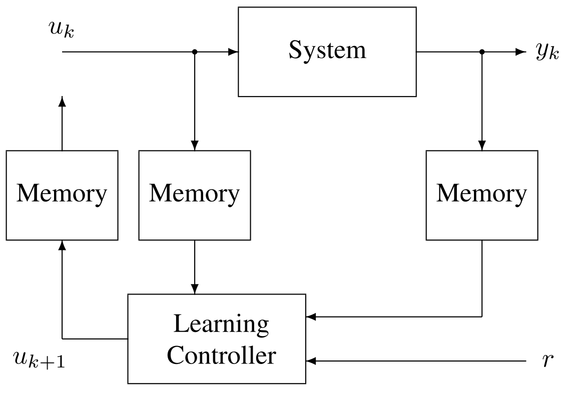


Fig. 2. Iterative learning control architecture.

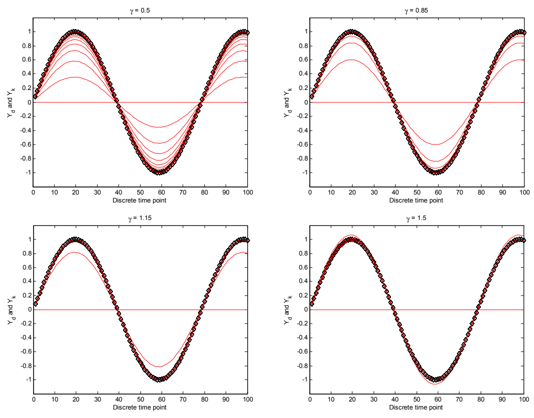


Fig. 3. Iterative learning control example.

of particular note is that there is no ILC equivalent of the Ziegler Nichols rules for three term proportional plus integral plus derivative (PID) design. Despite this, the literature gives many examples of physical processes where an ILC law designed by tuning has been applied. One approach is to use gradient methods to optimize the gain G_k in:

$$u_{k+1}(p) = u_k(t) + G_k e_k(p+1)$$

See for example [7], which introduces Gauss-Newton as well as other approaches, like steepest descent and Newton-Raphson for selecting the gain. These approaches led to the norm-optional ILC approach discussed below.

It is also useful for design to specify the learning control algorithm in the frequency domain, for example, viewing the update algorithm for continuous-time system as

$$U_{k+1}(s) = L(s)[U_k(s) + aE_k(s)]$$

We will consider this in more detail later in the paper.

The design of ILC laws requires performance specifications and algorithms for design, the subject of the rest of this paper, which focuses on model based design, again with emphasis on those that have seen experimental validation. Both linear and nonlinear dynamics are treated, along with expositions of the norm-optimal approach to ILC and frequency-domain techniques.

II. Linear Model Based Analysis and Design

From a signal flow perspective the basic idea of ILC is depicted in Fig. 2. Under certain assumptions on the system and a prescribed structure for the learning controller algorithm, which may use all past information from the system's operation, the goal is to achieve a form of convergence, e.g., zero error between the desired (reference) output $r(t)$ (or $r(p)$) and the actual output $y_k(t)$ (or $y_k(p)$).

To give a flavor of the effectiveness of ILC, consider the linear plant [8]

$$\begin{aligned} x_k(p+1) &= \begin{bmatrix} -0.8 & -0.22 \\ 1 & 0 \end{bmatrix} x_k(p) + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k(p) \\ y_k(p) &= [1, 0.5] x_k(p) \end{aligned}$$

with reference $r(p) = \sin(8.0p/100)$. Fig. 3 shows the trajectory evolution for the simple linear control update

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t+1)$$

with four different gains: $\gamma = 0.5$, $\gamma = 0.85$, $\gamma = 1.15$, $\gamma = 1.5$, assuming zero initial conditions on each trial and defining $e_k(t) = r(t) - y_k(t)$. For each gain, the ILC algorithm converges, but the convergence rate depends on γ. So, even in the absence of a model, perfect tracking is obtained through iteration.

A. An Operator-Theoretic Perspective

For the linear case, although many possibilities exist for designing and tuning ILC update algorithms, we can take a general operator-oriented perspective that contains all of these, at least for the case of using only data from the previous trial to update the input for the current trial. For instance, if we consider the system or plant to be a general linear operator (continuous or discrete-time) given by T_s, living in a specific vector space with a prescribed topology, and consider the ILC controller to be a general linear operator, then we have this result [9]:

Theorem 1: For the plant $y_k = T_s u_k$, the linear time-invariant learning control algorithm

$$u_{k+1} = T_u u_k + T_e(r - y_k)$$

converges to a fixed point $u^*(t)$ given by

$$u^*(t) = (I - T_u + T_e T_s)^{-1} T_e r(t)$$

with a final error

$$e^*(t) = \lim_{k\to\infty}(y_k - y_d) = (I - T_s(I - T_u + T_e T_s)^{-1} T_e) r(t)$$

defined on the interval (t_0, t_f) if

$$\|T_u - T_e T_s\|_i < 1$$

Further, if $T_u = I$ then $\|e^*(t)\| = 0$ for all $t \in [t_o, t_f]$, otherwise the error will be non-zero.

Such an operator-theoretic approach helps us understand the limits of performance and the nature of ILC as follows:

Theorem 2: Let T_n^* solve the problem:

$$\min_{T_n} \|(I - T_s T_n)r\|$$

Then $u^*(t)$ from Theorem 1 is given by $u^*(t) = T_n^* r(t)$ (and we can define T_u and T_c in terms of T_n^*).

This means that the essential effect of a properly designed learning controller is to produce the output of the best possible inverse of the system in the direction of y_d.

The development above is quite general, arguably covering the entire linear case (see [10] for more on the operator-theoretic perspective). In the next subsection, we restrict our attention to a more specific framework.

B. Discrete Dynamics – Lifting Setting

Considering the SISO case for ease of presentation, the finite trial length enables the values of a variable along a trial to be assembled into a finite dimensional column vector. This leads to the so-called lifted model description where the trial-to-trial dynamics are described by a standard linear systems state space model. Let Y_k, U_k, D, E_k and R denote the lifted model representations on trial k for the output, input, disturbance, and reference, respectively, where, e.g.,

$$\begin{aligned} Y_k &= \begin{bmatrix} y_k(1) & y_k(2) & \dots & y_k(N) \end{bmatrix}^T && (8) \\ U_k &= \begin{bmatrix} u_k(0) & u_k(1) & \dots & u_k(N-1) \end{bmatrix}^T && (9) \end{aligned}$$

Then the dynamics of (3) can be written as

$$\begin{aligned} Y_k &= GU_k \\ E_k &= R - Y_k \end{aligned} \qquad (10)$$

where

$$G = \begin{bmatrix} g_1 & 0 & \dots & 0 \\ g_2 & g_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_N & g_{N-1} & \dots & g_1 \end{bmatrix} \qquad (11)$$

where $g_j = CA^{j-1}B$. If, for example, the first Markov parameter $CB = 0$ then in (8) the first entry in this vectors is removed and the matrix G is adjusted accordingly, with a natural generalization to the other cases. In effect, control over the corresponding number of samples is lost. Note, however, that it is common to assume the linear plant has relative degree one, which is almost always true when working with a sampled-data system.

Consider application of the control law (5), where both Q and R can be non-causal functions. Let these functions have impulse responses

$$\begin{aligned} Q(q) &= \dots + q_{-2}q^2 + q_{-1}q + q_0 + q_1 q^{-1} + q_2 q^{-2} + \dots \\ L(q) &= \dots + q_{-2}l^2 + q_{-1}l + l_0 + l_1 q^{-1} + l_2 q^{-2} + \dots \end{aligned} \qquad (12)$$

The lifted representation of (5) is

$$\begin{aligned} U_{k+1} &= \begin{bmatrix} q_0 & q_{-1} & \dots & q_{-(N-1)} \\ q_1 & g_0 & \dots & q_{-(N-2)} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N-1} & g_{N-2} & \dots & q_0 \end{bmatrix} U_k \\ &+ \begin{bmatrix} l_0 & l_{-1} & \dots & l_{-(N-1)} \\ l_1 & l_0 & \dots & l_{-(N-2)} \\ \vdots & \vdots & \ddots & \vdots \\ l_{N-1} & g_{N-2} & \dots & l_0 \end{bmatrix} E_k \end{aligned} \qquad (13)$$

where the entries in these last two matrices are defined by the impulse responses of $Q(q)$ and $L(q)$, respectively. In the case when $Q(q)$ and $L(q)$ causal functions the matrices, denoted by Q and L respectively, in (12) are lower triangular. Moreover, the matrices G, L and Q are Toeplitz, i.e., all entries along each diagonal are the same.

The controlled system is said to be asymptotically stable if there exists a number, say β, such that

$$|u_k(p)| \leq \beta,\ 0 \leq p \leq N-1,\ k \geq 0$$

and also $\forall\, k\ \lim_{k\to\infty}$ exists. Also let $r(\cdot)$ denote the spectral radius of its matrix argument. Then the controlled system is asymptotically stable if and only if

$$r(Q(I - LG) < 1 \qquad (14)$$

In the particular case when Q and L are causal, $Q(I-LG)$ is lower triangular with repeated eigenvalues. In this case (13) is equivalent to

$$|q_0(1 - l_0 g_1)| < 1$$

Moreover, $g_1 = CB$, confirming that trial-to-trial error convergence can occur independent of the state matrix A (due to the final trial length).

A critical performance issue is the final value of the error $e_\infty(k) = \lim_{k\to\infty} e_k(p)$. If the design considered is asymptotically stable then the asymptotic error is

$$E_\infty = [I - G[I - Q(I - LG)]^{-1}QL](R - d) \qquad (15)$$

Note that this is equivalent to the expression for e^* in Theorem 1. Similar to that theorem, we can see that if convergence to zero error is required then, on the assumption that G and L are not identically zero, this property holds if and only if the controlled system is asymptotically stable and $Q(q) = 1$. The requirement of $Q(q) = 1$ is often enforced in the specification of the ILC law. Including $Q(q) \neq 1$ can help with transient learning, i.e., the response along-the-trials. One way to limit large transient growth is to enforce monotonic trial-to-trial error conference, i.e., require that for a specified norm $||\cdot||$

$$||e_\infty - e_{k+1}|| \leq \lambda ||e_\infty - e_k||,\ k \geq 0 \qquad (16)$$

where $\lambda \in (0, 1)$ is the convergence rate. There has been much study of the so-called "transient" problem. In fact, the error might grow very large in the trial direction before later converging to zero. See [11], [12] for early discussions of this problem and [13] for a more recent analysis.

An alternative approach to this last problem is to design a feedback control law and apply it to the system and then apply ILC to the controlled dynamics. The 2D systems/repetitive processes setting allows one step design for this case and is covered below. Applications of the control law considered here are widespread, see, e.g., [5], [14] and for more recent applications [6].

C. 2D systems/Repetitive Processes Analysis

As discussed in the opening section of this paper, ILC has a 2D systems structure, i.e., information from trial-to-trial and along-the-trial, respectively. The theory of 2D systems based on the well known Roesser [15] and Fornasini Marchesini [16] models has been extensively investigated since at least the mid 1970s. The first work on applying this theory to ILC design is widely credited to [17]. Since this first paper, a very large volume of literature has been produced but none of the designs have been experimentally validated.

Repetitive processes make a series of trials (also termed a pass in the repetitive process literature) through a set of dynamics defined over a finite duration known as the trial length. The novel feature of these processes is that the output produced on any trial acts as a forcing function on the subsequent trial and contributes to its dynamics. The result can be oscillations in the sequence of trial outputs whose amplitudes increase with k. A detailed treatment of these processes, focusing on the case of linear dynamics, is given in [18] (including their industrial origins). To highlight the distinction between such repetitive process and traditional dynamic systems, consider a typical model for a discrete repetitive process:

$$\begin{aligned} x_{k+1}(p+1) &= Ax_{k+1}(p) + Bu_{k+1}(p) + B_0 y_k(p) \\ y_{k+1}(p) &= Cx_{k+1}(p) + Du_{k+1}(p) + D_0 y_k(p) \end{aligned} \quad (17)$$

The distinction between (17) and conventional linear system is the introduction of the term $y_k(p)$. We emphasize that there are such physically occurring processes as detailed in [18]. Further, we point out that in such processes once can consider that the trial-to-trial memory in the system is inherent in the plant dynamics. This is in contrast to ILC, whereby the ILC controller creates a system in which the trial-to-trial memory in the system is contained in the controller. As such, ILC is as a special case of repetitive processes, as described below.

In a repetitive process, information propagation in one direction, along-the-trials, is of finite duration, which is physics based and not a mathematical assumption, and in this sense they are a better fit to ILC, and some of the designs developed using this setting have been experimentally validated, including robust designs. The focus in this section is again on design for discrete dynamics, but it is also possible to consider the case when the along-the-trial dynamics are described by linear differential equation, see, e.g., [6] (Chapter 7).

Consider again the ILC problem for discrete linear dynamics over $0 \leq p \leq N-1,\ k \geq 0$,

$$\begin{aligned} x_{k+1}(p+1) &= Ax_{k+1}(p) + Bu_{k+1}(p) \\ y_{k+1}(p) &= Cx_{k+1}(p),\ x_{k+1}(0) = 0 \end{aligned} \quad (18)$$

and write the state equation of this model in the form

$$x_k(p) = Ax_{k+1}(p-1) + Bu_k(p-1) \quad (19)$$

Also introduce

$$\eta_{k+1}(p+1) = x_{k+1}(p) - x_k(p) \quad (20)$$

and

$$u_{k+1}(p) = u_k(p) + \Delta u_{k+1}(p) \quad (21)$$

where

$$\Delta u_{k+1}(p) = K_1 \eta_{k+1}(p+1) + K_2 e_k(p+1) \quad (22)$$

The resulting controlled system dynamics are described by

$$\begin{aligned} \eta_{k+1}(p+1) &= (A+BK_1)\eta_{k+1}(p) + BK_2 e_k(p) \\ e_{k+1}(p) &= -C(A+BK_1)\eta_{k+1}(p) \\ &+ (I-CBK_2)e_k(p) \end{aligned} \quad (23)$$

This last description of the ILC dynamics is a particular case of the discrete linear repetitive process state space model [18] with zero input, current trial state vector $\eta_{k+1}(p)$ and previous trial profile vector $e_k(p)$. The stability theory for linear repetitive processes requires that a bounded initial trial profile produces a bounded sequence of trial profiles, i.e., the sequence $\{e_k\}$, where the bounded property is defined in terms of the norm on the underlying function space. The bounded property can be either over the finite and fixed trial length, or the stronger requirement that this property holds for all possible (i.e., finite) trial lengths. These properties are termed asymptotic stability and stability along-the-trial, respectively.

The stability theory for linear repetitive processes has been developed [18] based on an abstract model of the dynamics in a Banach space setting that includes all such processes as special cases. Hence application to a particular example requires interpretation of these general conditions. For the case of systems described by (23), let

$$\begin{aligned} \hat{A} &= A + BK_1,\ \hat{B}_0 = BK_2 \\ \hat{C} &= -C(A + BK_1),\ \hat{D}_0 = I - CBK_2 \end{aligned} \quad (24)$$

The following is Theorem 1 in [19].

Theorem 3: Suppose that the pair $\{\hat{A}, \hat{B}_0\}$ is controllable and the pair $\{\hat{C}, \hat{A}\}$ observable. Then the ILC dynamics described by the discrete linear repetitive process state space model (23) (with the notation of (24) has the stability along-the-trial property if and only if

a) $\rho(\hat{D}_0) < 1$
b) $\rho(\hat{A}) < 1$
c) all eigenvalues of the transfer function matrix

$$G(z) = \hat{C}(zI - \hat{A})^{-1}\hat{B}_0 + \hat{D}_0 \quad (25)$$

have modulus strictly less than unity for all $|z| = 1$.

Condition a) is the condition for the asymptotic stability property and guarantees trial-to-trial error convergence, condition b) requires that the dynamics along any trial are stable, and condition c) (in the SISO case for ease of presentation) requires that each frequency component of the initial error is attenuated from trial-to-trial at a geometric rate.

The following result is Theorem 2 in [19].

Theorem 4: The ILC dynamics described by the discrete linear repetitive process state-space model (23) has the stability along-the-trial property if there exist compatibly dimensioned matrices $X_1 \succ 0$, $X_2 \succ 0$, R_1 and R_2 such that the following LMI is feasible

$$M = \begin{bmatrix} H_1 & H_2 \end{bmatrix} \prec 0 \tag{26}$$

$$H_1 = \begin{bmatrix} -X_1 & 0 & X_1A^T + R_1^TB^T \\ 0 & -X_2 & R_2^TB^T \\ AX_1 + BR_1 & BR_2 & -X_1 \\ -CAX_1 - CBR_1 & X_2 - CBR_2 & 0 \end{bmatrix}$$

$$H_2 = \begin{bmatrix} -X_1A^TC^T - R_1^TB^TC^T \\ X_2 - R_2^TB^TC^T \\ 0 \\ -X_2 \end{bmatrix}$$

If this LMI is feasible, stabilizing control law matrices K_1 and K_2 are given by

$$K_1 = R_1X_1^{-1}, \ K_2 = R_2X_2^{-1} \tag{27}$$

The control law (21) can be written as

$$\begin{aligned} u_{k+1}(p) &= u_k(p) + K_1(x_{k+1}(p) - x_k(p)) \\ &\quad + K_2(r(p+1) - y_k(p+1)) \end{aligned} \tag{28}$$

where the last term on the right-hand side is phase-lead ILC action. Moreover, the second term on the right-hand side involves the state vectors' difference on the current and previous trials. Experimental verification of this design by application to a gantry robot completing a pick and place task is also reported in [19]. If all state vector entries cannot be measured, one option is to use an observer to estimate them. Another option is to replace the state feedback by output, i.e., trial, information, for the details see [20].

D. Robust Design and Other Applications

Design in the presence of uncertainties in the model is as relevant as in standard control design. The robustness filter approach will be considered in Section IV. A comprehensive overview of results in this area can be found in, e.g., [6], which cites the relevant literature. The designs include those developed using the lifted setting and those based on 2D systems/repetitive process theory.

In recent years, the application of ILC to healthcare has emerged. The early work was based on linear model design but substantial progress has been made using nonlinear dynamic models and control schemes and is discussed in the section of this paper on nonlinear ILC design.

III. Optimization-based Iterative Learning Control Design

This section introduces optimization-based ILC design techniques. We focus on the well-established Norm-optimal ILC (NOILC) framework to illustrate the key design principles, convergence properties, and its ability to handle constraints. We also briefly discuss alternative optimization-based approaches at the end. An extensive cover of the topics in this section and detailed proofs are found in the text [10].

A. Prologue – Gradient ILC and Parameter Optimization

To illustrate how optimization techniques can be used in ILC design, consider trial $k+1$. We have the input signal u_k and the tracking error e_k available from the previous trial k. We need to design an ILC law that updates the input signal u_{k+1} such that the tracking error e_{k+1} is as small as possible. If we use the squared 2-norm as a measure of how 'small' the error is, we then aim to minimize:

$$\min_{u_{k+1}} \|e_{k+1}\|^2, \ \text{s.t. } e_{k+1} = r - Gu_{k+1} - d$$

Here G denotes a general system operator, to be discussed further in the next subsection. This operator was called T_s in Theorem 1. In the discrete-time case it can be considered the matrix G defined in (10). The variable d represents the effect of initial conditions, as well as repeated noise and disturbances affecting the system.

Applying classical gradient optimization techniques at the currently available data pair (e_k, u_k) by using the Jacobian of $e_{k+1}^Te_{k+1}$ with respect to u_{k+1} leads to the following iterative update equation

$$u_{k+1} = u_k + \beta G^T(r - Gu_k - d) = u_k + \beta G^Te_k \tag{29}$$

where $\beta > 0$ is a step size (or 'learning gain') chosen by the user. This algorithm is called the *Gradient ILC Algorithm* (due to the use of gradient descent optimization method).

To analyse its convergence properties, we examine how the tracking error evolves from trial-to-trial. Substituting the update (29) for u_{k+1} into the tracking error definition gives

$$\begin{aligned} e_{k+1} &= r - y_{k+1} = r - Gu_{k+1} - d \\ &= r - G(u_k + \beta G^Te_k) - d = (I - \beta GG^T)e_k := L_ee_k \end{aligned}$$

where the *error evolution operator* $L_e = I - \beta GG^T$. Clearly the properties of L_e control the evolution of the error sequence, the analysis of which gives the following results:

Theorem 5: For Gradient-based ILC when the learning gain satisfies $0 < \beta\|G\|^2 < 2$, we have $\|L_e\| \leq 1$. Consequently the tracking error norm converges monotonically, i.e.,

$$\|e_{k+1}\| \leq \|e_k\|, \ \forall k \geq 0$$

Furthermore, the final converged tracking error is given by

$$\lim_{k\to\infty} e_k = P_{ker[G^T]}e_0$$

where $P_{ker[G^T]}e_0$ is the projection of the initial tracking error e_0 onto the kernel of G^T.

The learning gain β can also be trial varying (as long as it is within the above range), or it can be chosen automatically using a *Parameter Optimization* technique by minimising

$$J_{k+1}(\beta_{k+1}) = \|e_{k+1}\|^2 + \omega\beta_{k+1}^2$$

where $w > 0$ is a weighting parameter introduced to limit the value of β_{k+1} used. Solving it gives the following solution

$$\beta_{k+1} = \frac{\|G^T e_k\|^2}{\omega + \|GG^T e_k\|^2}$$

resulting in the same convergence properties as stated in Theorem 5 with the additional property that the sequence of optimal learning gains satisfies $\lim_{k\to\infty} \beta_k = 0$.

B. Norm Optimal Iterative Learning Control

The previous subsection shows how optimization techniques can be used to design ILC algorithms. In this subsection, we introduce a comprehensive optimization-based ILC design framework – the NOILC design. It can handle many different practical design situations. We first introduce a general mathematical representation of the system dynamics.

1) Operator Form Description of System Dynamics: The system is represented in the following operator form

$$y = Gu + d \tag{30}$$

where the system dynamics are assumed to be linear and described by a bounded linear operator G mapping a real Hilbert space $\mathcal{U}$ of input signals $u \in \mathcal{U}$ into a real Hilbert space $\mathcal{Y}$ of output signals $y \in \mathcal{Y}$, and $d \in \mathcal{Y}$ represents the initial condition or repeated disturbances. The inner product in $\mathcal{U}$ (resp. $\mathcal{Y}$) is denoted by $\langle u, v\rangle_{\mathcal{U}}$ (resp.$\langle y, w\rangle_{\mathcal{Y}}$). The induced norms in $\mathcal{U}$ (resp. $\mathcal{Y}$) are then given by $\|u\|_{\mathcal{U}} = \langle u, u\rangle_{\mathcal{U}}^{\frac{1}{2}}$ (resp. $\|y\|_{\mathcal{Y}} = \langle y, y\rangle_{\mathcal{Y}}^{\frac{1}{2}}$).

The above operator form description covers many situations of practical interest including convolution and state space descriptions of linear continuous time state space models and sampled data and multi-rate sampling systems plus problems where control signals have specific structure exemplified by a requirement that it is continuous and piecewise linear with discontinuities only allowed at specific time instants. In a similar manner to optimal control, the inner products and norms are design variables to the physical properties of the system, the control objective and/or the performance of the iterative algorithm to be used.

The following are two examples. Note that in this section, we use t to denote time in both continuous and discrete cases for notational simplicity, taking advantage of the generality of the operator form representation.

- A *linear discrete-time, time-invariant, ℓ-input, m-output state space system* with input vector $u(t)$ and output vector $y(t)$ on the time interval $0 \le t \le N$ can be associated with an input-output convolution relationship

$$y(t) = \sum_{t'=0}^{t} CA^{t'-1}Bu(t') + CA^t x_0$$

which has the form of (30) where G represents the convolution operator defined by the impulse response matrix $H(t) = CA^tB$, and d represents the time series $CA^t x_0$. The spaces $\mathcal{U}$ and $\mathcal{Y}$ are chosen as Cartesian product Hilbert spaces

$$\mathcal{U} = \underbrace{\mathcal{R}^\ell \times \mathcal{R}^\ell \times \cdots \times \mathcal{R}^\ell}_{(N+1)-\text{times}},\ \mathcal{Y} = \underbrace{\mathcal{R}^m \times \mathcal{R}^m \times \cdots \times \mathcal{R}^m}_{(N+1)-\text{times}}$$

with inner products defined by symmetric positive definite matrices $R(t)$ and $Q(t)$, $0 \le t \le N$, using

$$\langle u, v\rangle_{\mathcal{U}} = \sum_{t=0}^{N} u^T(t)R(t)v(t)$$

$$\langle y, w\rangle_{\mathcal{Y}} = \sum_{t=0}^{N} y^T(t)Q(t)w(t)$$

In lifted supervector form, G is the system matrix, u, y, d are supervectors introduced previously, and the inner products are just

$$\langle u, v\rangle_{\mathcal{U}} = u^T R_d v,\ \langle y, w\rangle_{\mathcal{Y}} = y^T Q_d w$$

where R_d and R_d are block diagonal matrices with diagonal elements $R(t)$, $Q(t)$ respectively.

- *Point to point tracking tasks.* The above formulation can be easily modified to describe the situation where the system is required to track a reference defined only at a finite number of intermediate time instants $t_1, t_2, \cdots, t_M$. In this case, the inner products are defined by setting $R(t)$ and $Q(t)$ to zero for all other time instants (while keeping other definitions unchanged).

The readers are referred to [10] for more examples including the linear continuous-time, time-invariant state space system.

2) The NOILC Algorithm: We are now ready to present the NOILC framework. Note NOILC now has a number of interpretations and extensions. In this paper we consider its basic and simplest form. More precisely, given data pair (e_k, u_k) on the k^{th} iteration, NOILC constructs the input u_{k+1} to be used on iteration $k + 1$ by minimizing

$$J(u) = \|e\|_{\mathcal{Y}}^2 + \|u - u_k\|_{\mathcal{U}}^2$$

subject to the constraints given by the system dynamics (30). The objective function has two parts: the first term penalizes the tracking error norm (reflecting the reference tracking requirement), and the second term limits the input change for caution and robustness and also provides an integral action (over the trial domain) that is necessary for zero tracking error to be achieved.

Solving this optimization problem gives the solution that

$$u_{k+1} = u_k + G^*(I + GG^*)^{-1}e_k \tag{31}$$

where G^* is the adjoint operator of G, i.e. the uniquely defined and bounded linear operator $\mathcal{Y} \to \mathcal{U}$ satisfying the condition that $\langle w, Gu\rangle_{\mathcal{Y}} = \langle G^*w, u\rangle_{\mathcal{U}}$ for all $w \in \mathcal{Y}, u \in \mathcal{U}$. The calculation of G^* for a number of discrete and continuous time state space problems can be found in [10]. As an example, for linear discrete time state space model introduced in the previous subsection, in lifted matrix form it is just $G^* = R_d^{-1}G^T Q_d$.

It is worth noting that if G is a state space model outlined in the previous subsection and $\mathcal{Y}$ and $\mathcal{U}$ are spaces with inner products defined therein, this problem is simply a linear quadratic optimal control problem with a familiar Riccati-style tracking solution consisting of both feedforward and feedback terms – please refer to [10] for details.

3) Convergence Properties: Substituting the NOILC updating law (31) into the tracking error definition gives the following error evolution

$$e_{k+1} = (I + GG^*)^{-1} e_k := L_e e_k$$

where the error evolution operator $L_e = (I + GG^*)^{-1}$. Analysis of it shows NOILC has the following convergence properties:

Theorem 6: The NOILC algorithm has the following properties:

1) The error signal norm sequence is monotonic, i.e.,

$$\|e_{k+1}\|_{\mathcal{Y}} < \|e_k\|_{\mathcal{Y}},\ k \geq 0$$

until input convergence has been achieved.

2) The error signal converges to the signal given as the orthogonal projection of e_0 onto the subspace of $\mathcal{Y}$ defined as the kernel of GG^*. If $\ker[GG^*] = \{0\}$, then the limit error is precisely zero.

3) The minimum value of $J(u)$ is $J(u_{k+1}) = \langle e_k, Le_k\rangle_{\mathcal{Y}} \leq J(u_k) = \|e_k\|^2_{\mathcal{Y}}$.

4) Finally, if the input sequence converges, it converges to a limit u_∞ that minimizes $\|u - u_0\|^2_{\mathcal{U}}$. That is

$$u_\infty = \arg\min\{\ \|u - u_0\|^2_{\mathcal{U}} : r = Gu + d\ \}$$

The inner product definitions in the underlying space $\mathcal{Y}$ and $\mathcal{U}$ (through the choices of $Q(t), R(t)$) determine the relative weight given to the error and change in the control signal. If more emphasis is put on the tracking error, a faster tracking error convergence is expected. Note however this may result in a 'high gain' form in feedback implementation that might introduce sensitivity and robustness problems.

C. NOILC with Constraints

Using the NOILC framework, system constraints that are widely existing in practice can be easily incorporated into the design. Taking input constraints as an example. Suppose the system input is constrained to be in a set Ω, taken to be a closed convex set in the input space $\mathcal{U}$, for example:

- $\Omega = \{u \in \mathcal{U} : |u(t)| \leq M(t)\}$
- $\Omega = \{u \in \mathcal{U} : \lambda(t) \leq |u(t)| \leq \mu(t)\}$
- $\Omega = \{u \in \mathcal{U} : 0 \leq u(t)\}$

where $M(t), \lambda(t), \mu(t)$ are known bounds on the input at time t. Then the input sequence $u_{k+1}, k = 0, 1, 2, \cdots$, defined by *Constrained NOILC Algorithm 1* as follows

$$u_{k+1} = \arg \min_{\substack{u\in\Omega \\ \text{s.t. (30)}}} \left\{\|e\|^2_{\mathcal{Y}} + \|u - u_k\|^2_{\mathcal{U}}\right\} \tag{32}$$

satisfies the constraint and iteratively solves the constrained ILC problem. The algorithm has the following properties:

Theorem 7: Constrained NOILC Algorithm 1 achieves monotonic convergence in the tracking error norm, i.e.,

$$\|e_{k+1}\|_{\mathcal{Y}} \leq \|e_k\|_{\mathcal{Y}},\ k \geq 0$$

Furthermore, if the underlying spaces $\mathcal{U}$ and $\mathcal{Y}$ are finite-dimensional, the input converges to a point u^*_s given by

$$u^*_s \in \arg \min_{\substack{u\in\Omega \\ \text{s.t. (30)}}} \|e\|^2_{\mathcal{Y}}$$

i.e., an input producing the minimum tracking error norm.

The above algorithm requires the solution of constrained quadratic programming problem (32) which in general does not admit a closed form solution. Many efficient numerical solvers are available. An alternative, computationally simpler algorithm *(Constrained NOILC Algorithm 2*) using the following input sequence $u_{k+1}, k = 0, 1, 2, \cdots$, defined by the solution of the unconstrained NOILC optimization problem

$$\tilde{u}_k = \arg \min_{\substack{u \\ \text{s.t. (30)}}} \left\{\|e\|^2_{\mathcal{Y}} + \|u - u_k\|^2_{\mathcal{U}}\right\} \tag{33}$$

followed by the simple input projection

$$u_{k+1} = \arg\min_{u\in\Omega} \|u - \tilde{u}_k\|_{\mathcal{U}} \tag{34}$$

also satisfies the constraint and iteratively solves the constrained ILC problem.

Note that the first step of Constrained NOILC Algorithm 2 is a standard NOILC step which can be implemented easily. The second step requires the solution of the problem (34). In practice the input constraint Ω is often a point-wise constraint and the solution of (34) can be computed straightforwardly, e.g., when $\Omega = \{u \in H : |u(t)| \leq M(t)\}$, the solution is simply

$$u_{k+1}(t) = \begin{cases} M(t) & : \tilde{u}_k(t) > M(t) \\ \tilde{u}_k(t) & : |\tilde{u}_k(t)| \leq M(t) \\ -M(t) & : \tilde{u}_k(t) < -M(t) \end{cases}$$

This approach is much simpler than Constrained NOILC Algorithm 1. Not surprisingly, this may sacrifice some convergence performance, e.g., monotonic convergence in the tracking error norm may no longer be preserved. As a consequence, the users may need to make a compromise based on the available computational resources and the desired convergence properties.

D. A Case Study on a Robotic Arm

The NOILC design has been tested on a six degree of freedom anthropomorphic robotic arm (shown in Figure 4). The system is required to track a point-to-point task replicating an industrial multiple 'pick and place' process in which payloads are manipulated during an assembly operation (shown in Figure 5 as stars). Frequency response tests have established that a linear model adequately represents the system dynamics, with angular input and output specified in degrees (see [21] for more details) which is used in the NOILC design.

Results are presented for the first joint which is aligned in the horizontal plane as illustrated in Figure 4. For comparison

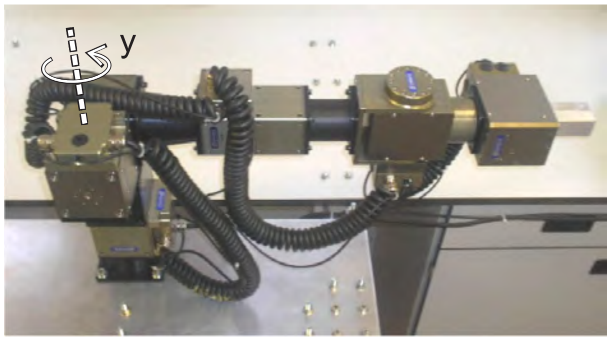


Fig. 4. Robotic manipulator system showing actuated output, y.

the predicted limiting solution is given using the nominal model. Results for the scalar, time invariant weighting choices $Q(t) = 1$ (for the intermediate tracking points, otherwise zero), and $R(t) = 0.2, 0.1$, and 0.02 are shown in Figure 5 which illustrates the effect on convergence speed of reducing the weighting on the input norm term in the cost. In all cases the algorithm enforces rapid convergence to similar low levels of input and error norm. The minimum energy property of the approach results in an input norm that does not match that of the nominal solution, due to the presence of modeling uncertainty and noise.

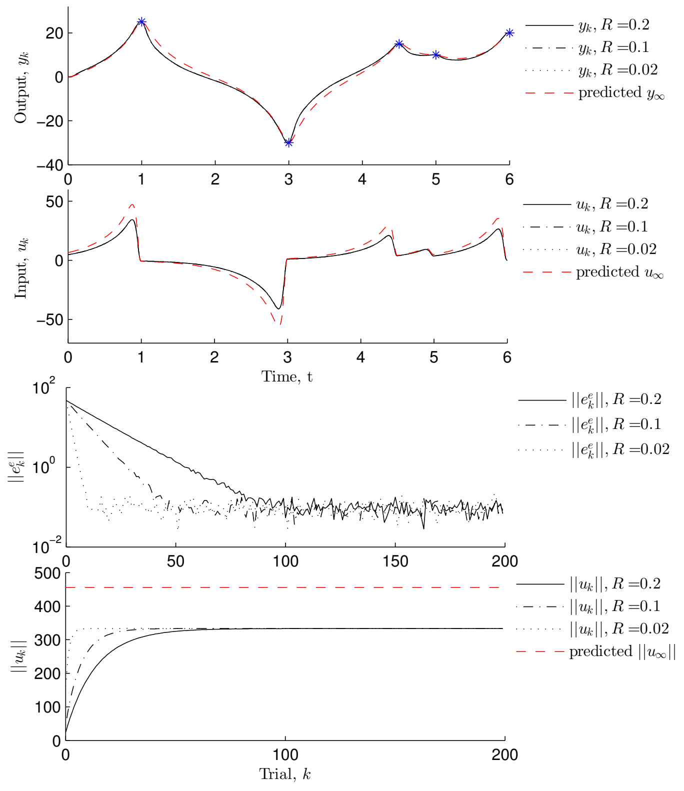


Fig. 5. Experimental final trial output and input, together with error norm and input norm results using NOILC.

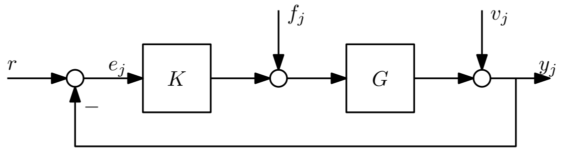


Fig. 6. Traditional motion control architecture.

E. *Discussion and Further Reading*

The use of optimization techniques in ILC design has a long history. Some earliest results on Gradient ILC can be found in [7], [22], and more recently in [23]–[25]. NOILC was first proposed in [26], [27] and further developed in [21], [28]–[34]. It now also has a number of different forms and interpretations, examples of which include introducing a penalty on the norm of the input (in addition or in place of penalizing the input change) [35], the use of basis functions [36], the use of data-driven methods [37]–[40]. Parameter optimal ILC was introduced in [41], [42]. Computational aspects are explored in [43], [44]. H_∞ based design was studied in [45], [46] and related robust designs in [47]–[49]. ℓ_0 and ℓ_1 based designs that promote sparsity are developed in [50]. Cross-coupled designs that involve time-varying weighting filters are investigated in [51]–[53], whereas spatial norms are investigated in [54] and intermittently sampled systems in [55]. The above list is by no means exhaustive – the reader is referred to the literature for more results.

Many of the optimization-based designs have found successful applications. Some examples include industrial robot [35], free electron lasers [56], [57], industrial flatbed printer [58], healthcare [59], broiler production [60], quantum control [61], nuclear fusion [62], additive manufacturing [63], [64], servo systems [65], [66]. The readers are referred to the recent text [67] for more examples.

IV. Frequency-domain ILC Design and Applications

A. *Introduction and Motivation*

Frequency domain approaches to iterative learning control are commonly applied to systems where frequency response models are used for their analysis and feedback control. Typical examples include mechatronic systems that are sufficiently linear, and where feedback designs are often made through frequency domain design methodologies [68].

A common control structure is depicted in Fig. 6, where r denotes the reference signal, G the system, v_j measurement noise, y_j the system output, K a stabilizing feedback controller, and f_j the ILC command signal. This parallel structure is most common in applications such as mechatronic systems, where f_j is the feedforward signal.

B. *Design Approach*

1) Learning filter L: The main idea in ILC is to learn from past measured error signals e_j to design the future command signal f_{j+1}. Suppose that an initial task $j = 0$ in Fig. 7, where a feedback controller K is implemented and feedforward $f_0 = 0$.

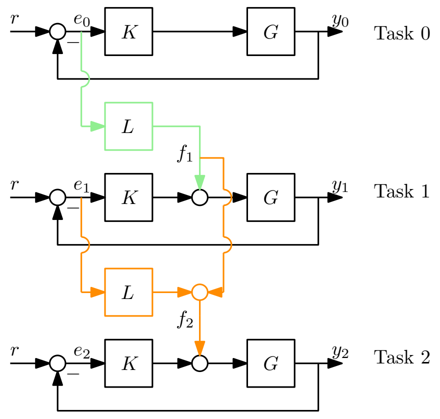


Fig. 7. Learning from past tasks.

After each task j, the ILC algorithm should generate a command signal for the next task f_{j+1}. Hence, command signal f_1 is determined prior to starting task 1, i.e., based on measured data in task 0. Assume for the moment that the same task is performed, and that only access to measured error signal e_0 is available. In addition, the disturbance v_1 is neglected to facilitate the exposition, i.e., $v_1 = 0$. The ILC problem now involves determining f_1 based on the measured signal e_0, such that e_1 is small. Note from Fig. 7 that

$$e_0 = Sr \tag{35}$$

$$e_1 = Sr - GSf_1 \tag{36}$$

with

$$S = \frac{1}{1+GK} \tag{37}$$

Frequency-domain ILC can be directly derived by following two key steps. First, substitute (35) into (36):

$$e_1 = e_0 - GSf_1 \tag{38}$$

Second, define f_1 as a filtered version of e_0, see Fig. 7, i.e.,

$$f_1 = Le_0 \tag{39}$$

In this case,

$$e_1 = (1 - GSL)e_0 \tag{40}$$

Next, $e_1 = 0$ is obtained by $(1 - GSL) = 0$. This is achieved by $L = (\widehat{GS})^{-1}$, where $\widehat{GS}$ is a model of the true closed-loop system, comprising both G and K.

In practice, model errors, i.e., $\widehat{GS} \neq GS$ may lead to a situation where f_1 leads to an error signal e_1 that is not exactly zero. This leads to the concept of iterative learning control:

$$f_{j+1} = f_j + Le_j \tag{41}$$

The intuition is as follows: if f_1 leads to $e_1 = 0$, then in the next task $j = 2$ this command input is retained, i.e., $f_2 = f_1$. Otherwise, a small correction Le_1 is added to f_1.

2) Design of the robustness filter Q: Despite that ILC generates a feedforward signal in Fig. 7, the iteration (41) in fact involves an iteration-domain feedback mechanism, and its convergence must be analyzed. In the analysis, the extended learning update

$$f_{j+1} = Q(f_j + Le_j) \tag{42}$$

is considered, where (41) is recovered by setting $Q = 1$. The freedom in the design of Q is exploited later on to improve convergence properties.

To analyze convergence, note from (36) that

$$e_j = Sr - GSf_j \tag{43}$$

By evaluating (43) for both j and $j + 1$, and using (42),

$$e_{j+1} = Q(1 - GSL)e_j + (1 - Q)Sr \tag{44}$$

The ILC convergence question reduces to whether the iteration (44) converges. The following theorem addresses this through a contraction mapping. Here, the notion of monotonic convergence is used, which is defined as follows.

Definition 1: The iteration (44) is monotonically convergent in the ℓ_2 norm if, for some $k \in [0, 1)$ and for all e_j,

$$\|e_\infty - e_{j+1}\|_2 \leq k\|e_\infty - e_j\|_2 \tag{45}$$

Theorem 8: The iteration (44) is monotonically convergent in the ℓ_2 norm to a fixed point e_∞ and corresponding f_∞ if and only if

$$\|Q(1 - GSL)\|_{\mathcal{L}_\infty} < 1 \tag{46}$$

For a proof of Theorem 8, see [50, Theorem 2]. Here, the ℓ_2 norm of a signal x is defined as $\|x\|_2 = \sqrt{\sum_{t=-\infty}^{\infty} |x(t)|^2}$.

The main point is that the condition (46) can be evaluated in the frequency domain, since it is equivalent to

$$\sup_\omega |Q(1 - GSL)| < 1 \tag{47}$$

for single-input single-output systems.

The condition (47) enables a systematic design, leading to the frequency domain design framework. The main idea is summarized in the following procedure.

Procedure 1: Frequency-domain ILC design procedure.

1. Identify a parametric model $\widehat{GS}$
2. set $L = (\widehat{GS})^{-1}$
3. check condition (46) with $Q = 1$
   - satisfied? implement (42) with $Q = 1$
   - not satisfied? next step
4. design Q such that condition (46) is satisfied
   - design $Q(\omega) \approx 1$ for performance
   - design $|Q(\omega)| < 1$ to satisfy (46)

Note that Q in Step 4 of Procedure 1 should be chosen close to 1 to ensure high performance. Indeed, $Q \neq 1$ leads to an asymptotic error, which can be directly derived from (44) and (35) and $e_\infty = \lim_{j\to\infty} e_j$, leading to $e_\infty = \frac{1-Q}{1-Q(1-GSL)}e_0$.

A key advantage of the approach outlined here is its fast convergence in conjunction with a systematic design of robustness filters. The design of the robustness filter, i.e.,

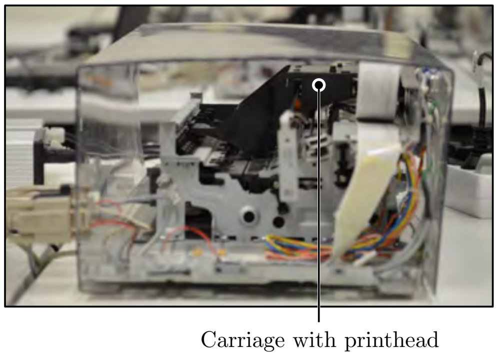


Fig. 8. Printer system with repeating tasks.

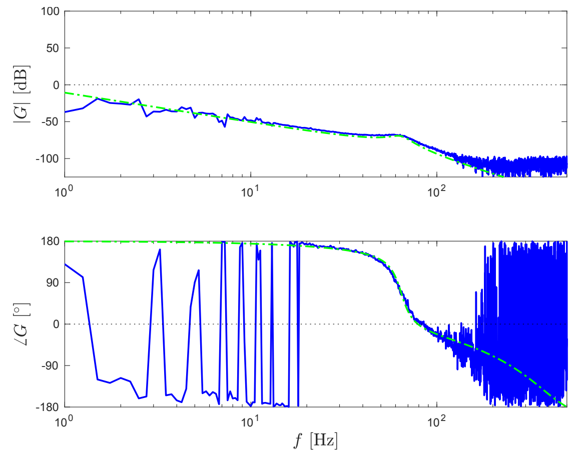


Fig. 9. Identified frequency response function of the printer system (solid blue), parametric model $\widehat{G}$ (dash-dotted green).

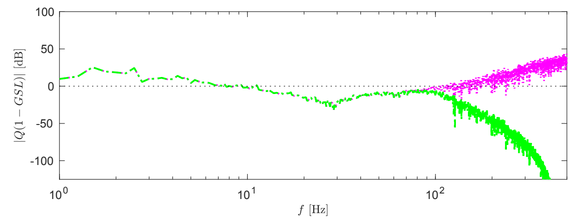


Fig. 10. Convergence condition (47) based on the identified non-parametric frequency response function of the printer system in Fig. 9 for $Q = 1$ (dotted magenta). Next, robust monotonic convergence is guaranteed by inclusion of Q, leading to $Q(1 - GSL)$ (dash-dotted green). In the latter case, the convergence condition is met for frequencies beyond 10 Hz. Below 10 Hz, the frequency response function is very noisy, and therefore discarded.

Step 4 of Procedure 1, can be further refined. In particular, the model $\widehat{GS}$ is typically less accurate compared to a nonparametric frequency response function estimate [69]. The condition (46) can even be verified for a range of nonparametric frequency response function estimates, corresponding to operating conditions [70] or populations of systems, ensuring robust monotonic convergence for the entire system set.

Remark 1: (Implementation aspects) The inversion in Step 1 in Procedure 1 may lead to a filter L that is analytic outside the unit disc (in the discrete time case), due to nonminimum phase zeros or strict delays of $\widehat{GS}$. In this case, a non-causal design of L should be implemented, see [71] for an overview of relevant algorithms. Similarly, a non-causal design of Q is advised by filtering both with Q and its adjoint to eliminate the phase, see [40]. In this respect, note that the $\mathcal{L}_\infty$ norm in (46) resembles the commonly used $\mathcal{H}_\infty$ norm [72]. The key difference is that the $\mathcal{L}_\infty$ allows for non-causal ILC algorithms [73], i.e, non-causal L and Q in (42).

C. Applications and Extensions

1) Design example: A design example is shown on the industrial printer system in Fig. 8. The task is to track a reference, i.e., moving the carriage over a sheet of paper. A linear model is identified. In particular, in Fig. 9, an identified frequency response function is depicted, as well as a parametric model $\widehat{G}$, leading to a closed-loop model $\widehat{GS} = \frac{\widehat{G}}{1+\widehat{G}K}$.

Next, Procedure 1 is invoked, leading to a filter L, and a design of Q in the frequency domain, see Fig. 10. Implementation leads to the error signal in Fig. 11, where it can clearly be seen that the error converges fast and to a small error value. The former is related to the learning filter that is essentially based on equating (40) to zero. The latter is due to the fact that $Q \approx 1$ for a large range of frequencies where the setpoint trajectory r has frequency content. For more details on the design, see [74].

2) Multivariable designs: Many systems have multiple inputs and outputs, in which case a multivariable learning filter must be designed. This requires special care, and suitable choices must be made regarding centralized or decentralized learning and robustness filters. A broad range of such algorithms is developed in [75].

3) Repetitive control: In certain applications, the error is repeating but the system is not reset after each tasks. This leads to a repetitive control setting, for which a very similar design framework is available, see [76] for an approach that is along the lines of the multivariable design in Sec. IV-C.2.

4) Noise reduction: In the analysis of Sec. IV-B, the noise v_j is set to zero. In case v_j is nonzero, the algorithm (42) may increase the error, see [50] for a detailed analysis and advanced designs to mitigate noise in ILC. Several of these approaches, including the sparse optimization in [50], directly extend the norm-optimal approach explained in the current paper.

5) Task flexibility: The basic assumption that r in Fig. 6 does not change for different tasks j is violated in many applications [77]. In [78], a frequency domain design is presented that incorporates basis functions for task flexibility.

V. Nonlinear ILC Design

From the beginning of ILC people have considered its use for nonlinear systems, as well as the use of nonlinear update laws for linear systems. Developed methods including contraction mapping (CM), composite energy functions (CEF), the gap metric, Newton's method and an extension of the 2D systems approach to nonlinear systems. Among these, CEF is often used for designing ILC algorithms for systems with

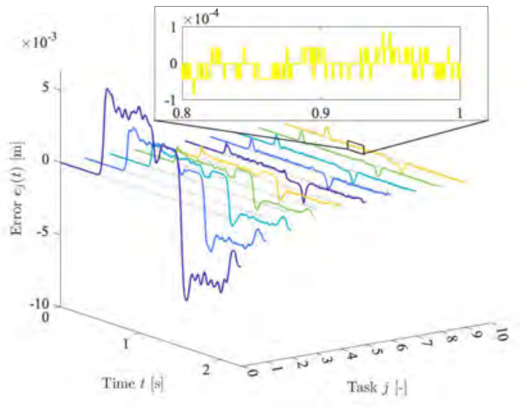


Fig. 11. Measured error signal for ten tasks j using the learning update (42) using L and Q of Fig. 10. The error reduces monotonically towards e_∞ for increasing j, achieving almost encoder resolution.

nonlinear dynamics.

In this section, we will show how CEF works to design an ILC algorithm for a simple second-order linear time-invariant (LTI) system and demonstrate how it can be used to explain human motor learning. CEF offers a structured approach to address these challenges by formulating an energy-based criterion that guides the iterative learning process. It is highlighted that although this example is an LTI system, the motivation for using the composite energy function (CEF) is to address nonlinear dynamic systems. In particular, CEF is useful when the global Lipschitz continuity condition does not hold, as discussed in [79]. Moreover, CEF can easily incorporate Lyapunov functions to design appropriate output feedback (as shown in Theorem 9). CEFs can also handle output constraints by incorporating for tools such as barrier control Lyapunov functions [80].

A. Design Approach

Let a second-order LTI dynamic system at the i^{th} iteration be given by:

$$\begin{aligned}\dot{\boldsymbol{x}}_i(t) &= A\boldsymbol{x}_i(t) + Bu_i(t)\\ y_i(t) &= C\boldsymbol{x}_i(t), \forall t \in [0, T_f]\end{aligned} \tag{48}$$

where A and B matrices are

$$A = \begin{bmatrix} 0 & 1\\ a_1 & a_2\end{bmatrix}, B = \begin{bmatrix} 0\\ b_1\end{bmatrix}, C = \begin{bmatrix} 1 & 0\end{bmatrix} \tag{49}$$

with a_1 and a_2 are unknown constants and b_1 is an unknown positive constant. Here $\boldsymbol{x}_i(\cdot) \in \mathcal{R}^2$ is the state, $u_i(\cdot) \in \mathcal{R}$ is the control input, and $y_i(\cdot) \in \mathcal{R}$ is the output at the i^{th} iteration.

The dynamic system (48) can represent a large class of controllable systems such as unmanned ground vehicles moving in a one-dimensional space. As it is in a controllable canonical form, the design principles presented here can be easily extended to high-dimensional movements, for example, robotic systems with multiple degrees of freedom [81], as well as other forms of nonlinear dynamics systems [82].

By checking the observability matrix, the system (48) is also observable. As $CB = 0$ and $CAB \neq 0$, this system has with a relative degree of 2. It is assumed that our control objective is to track a desired trajectory $y_d(t)$, which comes from the following reference model

$$\begin{aligned}\dot{\boldsymbol{x}}_d(t) &= A\boldsymbol{x}_d(t) + Bu_d(t)\\ y_d(t) &= C\boldsymbol{x}_d\end{aligned} \tag{50}$$

where matrices A, B, and C come from (49). At the i^{th} iteration, the tracking error is defined as

$$e_i(t) = y_d(t) - y_i(t), \qquad \forall t \in [0, T_f] \tag{51}$$

where y_i comes from (48).

The control objective is to find a sequence of control input $\{u_i\}_{i=1,2,\ldots,}$ such that

$$\lim_{i\to\infty} |e_i(t)| = 0, \forall t \in [0, T_f]. \tag{52}$$

Two possible ILC algorithms can be used to achieve this control objective: one is pure feed-forward, while the other incorporates both feedback and feed-forward strategies. The convergence analysis of the pure feed-forward approach is based on the CM method, whereas the CEF is used to demonstrate convergence when both feedback and feed-forward parts are employed.

By abusing the notation, we ignore the time variable t in our analysis when no confusion arises. For the convenience of notation, we define

$$\Delta\boldsymbol{x}_i = \boldsymbol{x}_d - \boldsymbol{x}_i, \qquad \delta\boldsymbol{x}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i \tag{53}$$

Feed-Forward Law

In such a situation, we use only a feed-forward control law:

$$u_i(t) = u_i^{ff}(t), \qquad u_0(t) = 0, \qquad \forall t \in [0, T_f] \tag{54}$$

As the system (48) has the relative degree of 2, the feed-forward control law takes the following form:

$$u_{i+1}^{ff}(t) = u_i^{ff}(t) + \gamma\ddot{e}_i(t), \qquad \forall t \in [0, T_f] \tag{55}$$

where $\gamma > 0$ is the learning gain and the tracking error e is defined in (51).

For a continuous function $s(t)$, its time-weighted norm is defined as $\|s\|_\lambda = \max_{t\in[0,T_f]} e^{-\lambda t}|s(t)|$ for some positive constant λ.

Using the CM method, the following proposition shows how the feed-forward ILC (54) works.

Proposition 1: Assuming that $\ddot{e}_i(0) = \dot{e}_i(0) = e_i(0), \forall i = 0, 1, \ldots,$ the proposed control law (54), where $u_i^{ff}(t)$ comes from (55), will ensure that the tracking error converges, i.e.,

$$\lim_{i\to\infty} |e_i(t)| = 0, \forall t \in [0, T_f] \tag{56}$$

if the following convergence condition holds:

$$|1 - \gamma b_1| \leq \rho < 1, \qquad \rho \in (0, 1) \tag{57}$$

Feedback with Feed-Forward Law

Now let the ILC law have both feedback and feed-forward

$$u_i = u_i^{fb} + u_i^{ff} \tag{58}$$

Assume that the state of (48) is measurable. As the system (48) is controllable, there exists a feedback gain $K \in R^{2\times 1}$ such that if the control input takes the following form

$$u_i^{fb} = -K\left(\boldsymbol{x}_d - \boldsymbol{x}_i\right) \tag{59}$$

then the matrix $A - BK$ is a Hurwitz. As $A - BK$ is a Hurwitz, for any given $Q = Q^T > 0$, $Q \in R^{2\times 2}$, there exists a symmetric positive definite matrix $P \in R^{2\times 2}$ such that following Lyapunov equation is satisfied:

$$(A - BK)^T P + P(A - BK) = -Q. \tag{60}$$

The feed-forward control law takes the following form

$$u_{i+1}^{ff} = u_i^{ff} + B^T P \Delta \boldsymbol{x}_i \tag{61}$$

where $u_0^{ff}(t) = 0, \forall t \in [0, T_f]$, B is from (49), and P is from (60). By combining (48) and (58), the closed-loop system becomes

$$\Delta \dot{\boldsymbol{x}}_i(t) = (A - BK)\Delta \boldsymbol{x}_i(t) - B\Delta u_i^{ff}(t) \tag{62}$$

where

$$\Delta u_i^{ff} = u_d - u_i^{ff}. \tag{63}$$

Theorem 9: Assuming that $\Delta \boldsymbol{x}_i(0) = \mathbf{0}_{2\times 1}, \forall i = 0, 1, \ldots$, the proposed control law (58), consisting of the feedback (59) satisfying (60), and feed-forward (61), will ensure that the tracking error converges, i.e.,

$$\lim_{i\to\infty} |\Delta \boldsymbol{x}_i(t)|^2 = 0, \forall t \in [0, T_f] \tag{64}$$

Remark 2: In the control law (58), we need to find a stabilizing feedback K in the feedback (59). This feedback K and its corresponding Lyapunov equation (60) play an important role in the feed-forward law (61). In many applications, engineering practitioners know the nominal model. The knowledge of the nominal model also can help to design the appropriate K, with its corresponding matrix P. On the other hand, when matrices A and B are unknown, designing an appropriate feedback gain K can be challenging. However, in certain cases, such as robotics, a high-gain PD control law can always stabilize the system, provided that an appropriate energy-related Lyapunov function is available.

Remark 3: CEF here consists of two parts: one related to the state tracking energy ($V_i(t) = e^{-\lambda t}\Delta \boldsymbol{x}_i^T(t) P \Delta \boldsymbol{x}_i(t)$) and the other related to the integral performance of the input tracking, i.e., the control input converging to the desired control input. The feed-forward law (61) is more like a point-wise adaptation, trying to cancel the influence of $\Delta u_i(t)$ in the dynamics in the time domain by using the updating law in the iteration domain. It is highlighted that $V_i(t)$ has the term $e^{-\lambda t}$. When it moves inside the integral part, it generates a large damping term $-\lambda \int_0^t e^{-\lambda\tau} V_i(\tau) d\tau$ by selecting a sufficiently large λ. The role of λ in the CEF is similar to that of a time-weighted norm in CM (see Proposition 1) in the sense that both of them try to ignore some dynamics in the convergence analysis.

Remark 4: The feedback is not needed in the convergence analysis. When $K = 0_{2\times 1}$, we can select $V_i(t) = \Delta \boldsymbol{x}_i^T(t)\Delta \boldsymbol{x}_i(t)$ with $P = I_{2\times 2}$. The following feed-forward law is proposed,

$$u_{i+1}^{ff} = u_i^{ff} + \Delta \boldsymbol{x}_{2,i} = u_i^{ff} + \dot{e}_i \tag{65}$$

By constructing the following CEF

$$E_i(t) \;\; = \;\; V_i(t) + \frac{1}{b_1}\int_0^t e^{-\lambda\tau}\left(\Delta u_{i+1}(\tau)\right)^2 d\tau \tag{66}$$

for the positive b_1, by following similar steps to the proof of Theorem 1, we can show that the pure feed-forward law (65) can also ensure convergence. However, using CEF, we utilize $\dot{e}_i$ instead of $\ddot{e}_i$ for a dynamic system with the relative degree of 2 as in (55). Consequently, this simplifies the resetting condition since $\ddot{e}_i(0)$ is no longer needed. Moreover, the convergence condition (57) is no longer required. This shows that the CEF provides more flexibility in designing ILC updating laws. Moreover, recent work showed that the composite energy function (CEF) [82] can be used to demonstrate the convergence of ILC algorithms when they are designed based on the contraction mapping (CM) method. These findings support the assertion that the CEF is a very powerful tool in designing and analyzing ILC algorithms.

B. Application to Healthcare

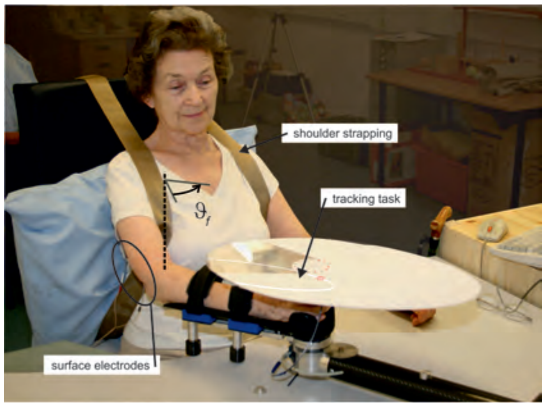
The application of ILC to healthcare problems started in mid-2004/2005 with the stroke rehabilitation application discussed next, but other applications have also emerged, e.g., see [83] for ventricular assist devices.

Stroke is a leading cause of disability worldwide, with incidence set to rise, which, in turn, will place an increasing burden on healthcare and rehabilitation resources. If the capacity of health services is to meet future demand, novel approaches to rehabilitation are needed. Enabling rehabilitation outside the hospital, supported by mobile technology, may lead to i) reduced cost, ii) increased intensity of therapy, and iii) shift the emphasis of responsibility for good health from healthcare professionals to the patients.

When people re-learn skills after a stroke, they go through the same process as you do when you learn to play tennis, but they have a problem because they can hardly move at all, so they cannot practice, which means they do not get feedback. Muscles can be made to work by Electrical Stimulation (ES).

If ES is applied to a person's muscle, electrical impulses travel along the nerves in much the same way as the electrical impulses from the brain. If the stimulation is carefully controlled, beneficial movement can be made. This effect works better if the person is attempting the movement themselves and hence needs to combine a person's effort with just enough extra electrical stimulation to achieve the movement. Upper limb impairment is very common post-stroke and limits many daily living activities, especially those requiring reach to grasp actions such as picking up a drink.

Fig 12 shows the system designed in the first work based on a reaching (2D) task where a) shows a patient using the experimental setup and b) the actions required. In essence



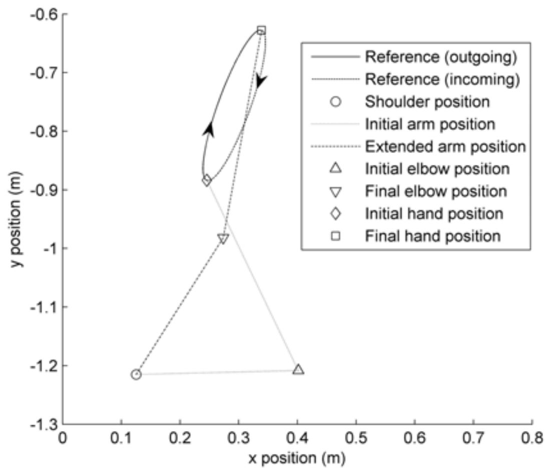


Fig. 12. a) Photograph of a patient using the robotic system, b) ILC perspective.

the patient is presented with a task and attempts to move the effected limb to follow a lighted path (the reference trajectory) with a simple structure ILC law used to adjust the stimulation applied during each trial, the development of this system together with the modeling and control law design is detailed in [84]. If a patient is improving with each attempt, the voluntary effort should increase and the applied stimulation decrease. This effect was detected in a small scale clinical trial [85].

This initial success, has led on to research for rehabilitating 2D tasks, such as collecting a garment from a table, putting it into an open drawer and then using the affected limb to reach up and close it. This requires stimulation of two muscles and linear model based ILC is ineffective. Moreover, there is a considerable degree of uncertainty present and a robust design is needed. Designs based on the Newton method, input-output feedback linearization, and the gap metric have been developed and supporting experimental results/clinical trials have been reported, see, e.g., [86]–[88].

VI. Possible Future Research Directions

There are a great many areas for possible future research, both in terms of new theory/algorithms and continuing to develop methods to assist with applications. This section gives some possible areas (based on the views of the authors). The first area is ILC for distributed parameter systems, where one approach is to continue to develop analysis as per non-ILC applications, e.g., based on semigroup theory. The nonlinear equations arising in applications, such as flow over a wind turbine blade, do not lend themselves to such analysis except under very strong assumptions.

An alternative approach is to base design on data obtained from simulations, such as computational fluid dynamics, which is used to construct finite dimensional approximate models on which to base design and then test the resulting ILC laws against, e.g., the computational fluid mechanics model. Some initial progress in this general direction is reported in [89], with supporting experimental results from application to a heating process. The potential for data-driven ILC designs is yet to be fully established, but some early results were noted above in Section III-E, see [37]–[39].

In implementations, it can be the case that a linear model based control design encounters difficulties due to, e.g., actuator saturation or backlash or windup in the actuators. Some first results for saturation, with experimental results are in [90]. More research is required in this general area.

For many applications, the basic assumptions in ILC are violated. One area where ILC is exceptionally promising is in precision mechatronics. There, systems involve position-dependent behavior, different sensors for ILC than feedback with nontraditional sampling schemes, and flexible tasks. These all require new approaches, see [91] for an overview of these challenges.

The theory of nonlinear ILC needs substantial research effort, including stochastic designs. Also there are connections between ILC and reinforcement learning, initial results are reported in [92] and [93]. Furthermore, there is also a clear link between ILC and repetitive control, also for nonlinear systems, and which approach is most applicable for the application at hand should be carefully decided, see e.g. [94] and [95] for recent results on a healthcare application where such algorithms are implemented.

Regarding these healthcare applications, the broad field is ripe for further application, especially in the ideal case of take home technology. Further research on such learning approaches and the applications are required, and we expect this will lead to a major impact for future healthcare.

References

[1] M. Uchiyama, "Formation of high speed motion pattern of mechanical arm by trial," *Society of Instrumentation and Control Engineers*, vol. 5, no. 19, pp. 706–712, 1978.

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

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

[4] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control: A learning-based method for high-performance tracking control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, 2006.

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

[6] E. Rogers, B. Chu, C. T. Freeman, and P. L. Lewin, *Iterative Learning Control Algorithms and Experimental Benchmarking*. Wiley, 2023.

[7] M. Togai and O. Yamano, “Analysis and design of an optimal learning control scheme for industrial robots: A discrete system approach,” in *Proceedings of the 24th IEEE Conference on Decision Control.* IEEE, New York, NY, USA, 1985, pp. 1399–1404.

[8] H.-S. Ahn, “Robust and adaptive learning control design in the iteration domain,” Ph.D. dissertation, Utah State University, Colorado, USA, 2006.

[9] K. L. Moore, M. Dahleh, and S. P. Bhattacharyya, “Iterative learning control: a survey and new results,” *Journal of Robotic Systems*, vol. 9, no. 5, pp. 563–594, 1992.

[10] D. Owens, *Iterative Learning Control: An Optimization Paradigm*. Springer, 2015.

[11] M. Norrlof and S. Gunnarsson, “Time and frequency domain convergence properties in iterative learning control,” *International Journal of Control*, vol. 75, no. 14, pp. 1114–1126, 2002.

[12] H. Elci, R. Longman, M. Phan, J.-N. Juang, and R. Ugoletti, “Simple learning control made practical by zero-phase filtering: Applications to robotics,” *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 49, pp. 753 – 767, 07 2002.

[13] S. Koscielniak, “Observations on causal iterative-learning-control & transients,” in *2024 American Control Conference (ACC)*. IEEE, 2024, pp. 4786–47 914.

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

[15] R. P. Roesser, “A discrete state-space model for linear image processing,” *IEEE Transactions on Automatic Control*, vol. 20, no. 1, pp. 1–10, 1975.

[16] E. Fornasini and G. Marchesini, “Two-dimensional linear doubly-indexed dynamical systems: state-space models and structural properties,” *Mathematical Systems Theory*, vol. 12, pp. 59–72, 1978.

[17] J. Kurek and M. Zaremba, “Iterative learning control synthesis based on 2-D system theory,” *IEEE Transactions on Automatic Control*, vol. 38, no. 1, pp. 121 –125, 1993.

[18] E. Rogers, K. Gałkowski, and D. H. Owens, *Control Systems Theory and Applications for Linear Repetitive Processes*, ser. Lecture Notes in Control and Information Sciences. Berlin, Germany: Springer-Verlag, 2007, vol. 349.

[19] L. Hladowski, K. Galkowski, Z. Cai, E. Rogers, C. T. Freeman, and P. L. Lewin, “Experimentally supported 2D systems based iterative learning control law design for error convergence and performance,” *Control Engineering Practice*, vol. 18, no. 4, pp. 339–348, 2010.

[20] ——, “Output information based iterative learning control law design with experimental verification,” *Journal of Dynamic Systems, Measurement, and Control Publisher: ASME*, vol. 134, no. 2, p. https://doi.org/10.1115/1.4005038, 2012.

[21] D. H. Owens, C. T. Freeman, and B. Chu, “Multivariable norm optimal iterative learning control with auxiliary optimisation,” *International Journal of Control*, vol. 86, no. 6, pp. 1026–1045, 2013.

[22] K. Furuta and M. Yamakita, “The design of learning control systems for multivariable systems,” in *Proceedings of the IEEE International Symposium on Intelligent Control*, 1987, pp. 371–376.

[23] K. Kinoshita, T. Sogo, and N. Adachi, “Adjoint-type iterative learning control for a single-link flexible arm,” *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 223–228, 2002.

[24] D. H. Owens, J. Hatonen, and S. Daley, “Robust monotone gradient-based discrete-time iterative learning control,” *International Journal of Robust and Nonlinear Control*, vol. 19, no. 6, pp. 634–661, 2009.

[25] D. H. Owens and B. Chu, “Combined inverse and gradient iterative learning control: performance, monotonicity, robustness and non-minimum-phase zeros,” *International Journal of Robust and Nonlinear Control*, vol. 24, no. 3, pp. 406–431, 2014.

[26] N. Amann, D. H. Owens, and E. Rogers, “Iterative learning control for discrete-time systems with exponential rate of convergence,” *IEE Proceedings: Control Theory and Applications*, vol. 143, no. 2, pp. 217–224, 1996.

[27] ——, “Iterative learning control using optimal feedback and feedforward actions,” *International Journal of Control*, vol. 65, no. 2, pp. 277–293, 1996.

[28] ——, “Predictive optimal iterative learning control,” *International Journal of Control*, vol. 69, no. 2, pp. 203–226, 1998.

[29] D. H. Owens, C. T. Freeman, and T. Van Dinh, “Norm-optimal iterative learning control with intermediate point weighting: theory, algorithms, and experimental evaluation,” *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 999–1007, 2012.

[30] D. H. Owens and B. Chu, “Error corrected references for accelerated convergence of low gain norm optimal iterative learning control,” *IEEE Transactions on Automatic Control*, 2024.

[31] B. Chu and D. H. Owens, “Accelerated norm-optimal iterative learning control algorithms using successive projection,” *International Journal of Control*, vol. 82, no. 8, pp. 1469–1484, 2009.

[32] ——, “Iterative learning control for constrained linear systems,” *International Journal of Control*, vol. 83, no. 7, pp. 1397–1413, 2010.

[33] B. Chu, C. T. Freeman, and D. H. Owens, “A novel design framework for point-to-point ILC using successive projection,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1156–1163, 2015.

[34] B. Chu, A. Rauh, H. Aschemann, E. Rogers, and D. H. Owens, “Constrained iterative learning control for linear time-varying systems with experimental validation on a high-speed rack feeder,” *IEEE Transactions on Control Systems Technology*, vol. 30, no. 5, pp. 1834–1846, 2021.

[35] S. Gunnarsson and M. Norrlof, “On the design of ILC algorithms using optimization,” *Automatica*, vol. 37, no. 12, pp. 2011–2016, 2001.

[36] J. Bolder and T. Oomen, “Rational basis functions in iterative learning control—with experimental verification on a motion system,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 722–729, 2015.

[37] P. Janssens, G. Pipeleers, and J. Swevers, “A data-driven constrained norm-optimal iterative learning control framework for LTI systems,” *IEEE Transactions on Control Systems Technology*, vol. 21, no. 2, pp. 546–551, 2012.

[38] Z. Jiang and B. Chu, “Norm optimal iterative learning control: A data-driven approach,” *IFAC-PapersOnLine*, vol. 55, no. 12, pp. 482–487, 2022.

[39] B. Chu and P. Rapisarda, “Data-driven iterative learning control for continuous-time systems,” in *2023 62nd IEEE Conference on Decision and Control (CDC)*. IEEE, 2023, pp. 4626–4631.

[40] J. Bolder, S. Kleinendorst, and T. Oomen, “Data-driven multivariable ILC: Enhanced performance by eliminating L and Q filters,” *International Journal of Robust and Nonlinear Control*, vol. 28, no. 12, pp. 3728–3751, 2018.

[41] D. H. Owens and K. Feng, “Parameter optimization in iterative learning control,” *International Journal of Control*, vol. 76, no. 11, pp. 1059–1069, 2003.

[42] J. Hätönen, D. H. Owens, and K. Feng, “Basis functions and parameter optimisation in high-order iterative learning control,” *Automatica*, vol. 42, no. 2, pp. 287–294, 2006.

[43] K. L. Barton, D. A. Bristow, and A. G. Alleyne, “A numerical method for determining monotonicity and convergence rate in iterative learning control,” *International Journal of Control*, vol. 83, no. 2, pp. 219–226, 2010.

[44] J. van Zundert, J. Bolder, S. Koekebakker, and T. Oomen, “Resource-efficient ILC for LTI/LTV systems through LQ tracking and stable inversion: Enabling large tasks on a position-dependent industrial printer,” *Mechatronics*, vol. 38, pp. 76–90, 2016.

[45] N. Amann, D. H. Owens, E. Rogers, and A. Wahl, “An H_∞ approach to linear iterative learning control design,” *International Journal of Adaptive Control and Signal Processing*, vol. 10, no. 6, pp. 767–781, 1996.

[46] D. De Roover and O. H. Bosgra, “Synthesis of robust multivariable iterative learning controllers with application to a wafer stage motion system,” *International Journal of Control*, vol. 73, no. 10, pp. 968–979, 2000.

[47] D. A. Bristow, “Weighting matrix design for robust monotonic convergence in norm optimal iterative learning control,” in *Proc. 2008 American Control Conference*, Seattle, WA, United States, 2008, pp. 4554–4560.

[48] J. van de Wijdeven, T. Donkers, and O. Bosgra, “Iterative learning control for uncertain systems: Robust monotonic convergence analysis,” *Automatica*, vol. 45, pp. 2383–2391, 2009.

[49] W. Paszke, E. Rogers, K. Gałkowski, and Z. Cai, “Robust finite frequency range iterative learning control design and experimental verification,” *Control Engineering Practice*, vol. 21, no. 10, pp. 1310–1320, 2013.

[50] T. Oomen and C. R. Rojas, “Sparse iterative learning control with application to a wafer stage: Achieving performance, resource efficiency, and task flexibility,” *Mechatronics*, vol. 47, pp. 134–137, 2017.

[51] K. L. Barton and A. G. Alleyne, "A cross-coupled iterative learning control design for precision mechatronics," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1218–1231, 2008.

[52] K. L. Barton, D. J. Hoelzle, A. G. Alleyne, and A. J. Wagoner Johnson, "Cross-coupled iterative learning control of systems with dissimilar dynamics: Design and implementation," *International Journal of Control*, vol. 84, no. 7, pp. 1223–1233, 2011.

[53] L. Aarnoudse, J. Kon, K. Classens, M. van Meer, M. Poot, P. Tacx, N. Strijbosch, and T. Oomen, "Cross-coupled iterative learning control: A computationally efficient approach applied to an industrial flatbed printer," *Mechatronics*, vol. 99, p. 103170, 2024.

[54] D. J. Hoelzle and K. L. Barton, "On spatial iterative learning control via 2-D convolution: Stability analysis and computational efficiency," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 4, pp. 1504–1512, 2016.

[55] N. Strijbosch and T. Oomen, "Iterative learning control for intermittently sampled data: Monotonic convergence, design, and applications," *Automatica*, vol. 139, p. 110171, 2022.

[56] E. Rogers, D. H. Owens, H. Werner, C. T. Freeman, P. L. Lewin, C. S. S. Kichhoff, and G. Lichtenberg, "Norm optimal iterative learning control with application to problems in accelerator based free electron lasers and rehabilitation robotics," *European Journal of Control*, vol. 16, no. 5, pp. 496–521, 2010.

[57] A. Rezaeizadeh and R. S. Smith, "Iterative learning control for the radio frequency subsystems of a free-electron laser," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1567–1577, 2018.

[58] L. Blanken, S. Koekebakker, and T. Oomen, "Data-driven feedforward tuning using non-causal rational basis functions: With application to an industrial flatbed printer," *Mechatronics*, vol. 71, p. 102424, 2020.

[59] C. T. Freeman, E. Rogers, A. M. Hughes, J. H. Burridge, and K. L. Meadmore, "Iterative learning control in health care: electrical stimulation and robotic-assisted upper-limb stroke rehabilitation," *IEEE Control Systems Magazine*, vol. 32, no. 1, pp. 18–53, 2012.

[60] S. V. Johansen, M. R. Jensen, B. Chu, J. D. Bendtsen, J. Mogensen, and E. Rogers, "Broiler FCR optimization using norm optimal terminal iterative learning control," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 2, pp. 580–592, 2019.

[61] R.-B. Wu, B. Chu, D. H. Owens, and H. Rabitz, "Data-driven gradient algorithm for high-precision quantum control," *Physical Review A*, vol. 97, no. 4, p. 042122, 2018.

[62] F. Felici and T. Oomen, "Enhancing current density profile control in tokamak experiments using iterative learning control," in *Proc. Conference on Decision and Control*, Osaka, Japan, 2015, pp. 5370–5377.

[63] Z. Wang, C. P. Pannier, K. Barton, and D. J. Hoelzle, "Application of robust monotonically convergent spatial iterative learning control to microscale additive manufacturing," *Mechatronics*, vol. 56, pp. 157–165, 2018.

[64] E. C. Balta, D. M. Tilbury, and K. Barton, "Iterative learning spatial height control for layerwise processes," *Automatica*, vol. 167, p. 111756, 2024.

[65] W. Paszke, E. Rogers, and K. Gałkowski, "Experimentally verified generalized KYP lemma based iterative learning control design," *Control Engineering Practice*, vol. 53, pp. 57–67, 2016.

[66] W. Paszke, R. Merry, and R. van de Molengraft, "Iterative learning control by two-dimensional system theory applied to a motion system," in *Proc. 2007 American Control Conference*, New York, NY, United States, 2007, pp. 5484–5489.

[67] E. Rogers, B. Chu, C. Freeman, and P. Lewin, *Iterative Learning Control Algorithms and Experimental Benchmarking*. John Wiley & Sons, 2023.

[68] R. Munnig Schmidt, G. Schitter, and J. van Eijk, *The Design of High Performance Mechatronics*. Delft, The Netherlands: Delft University Press, 2011.

[69] R. Pintelon and J. Schoukens, *System Identification: A Frequency Domain Approach*, 2nd ed. New York, NY, United States: IEEE Press, 2012.

[70] R. van der Maas, A. van der Maas, R. Voorhoeve, and T. Oomen, "Accurate FRF identification of LPV systems: nD-LPM with application to a medical X-ray system," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1724–1735, 2017.

[71] J. van Zundert and T. Oomen, "On inversion-based approaches for feedforward and ILC," *Mechatronics*, vol. 50, pp. 282–291, 2018.

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

[73] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control: A learning-based method for high-performance tracking control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, 2006.

[74] T. Oomen, "Learning for advanced motion control," in *Int. Workshop on Advanced Motion Contr.*, Agder, Norway, 2020, pp. 65–72.

[75] L. Blanken and T. Oomen, "Multivariable iterative learning control design procedures: From decentralized to centralized, illustrated on an industrial printer," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 4, pp. 1534–1541, 2020.

[76] L. Blanken, S. Koekebakker, and T. Oomen, "Multivariable repetitive control: Decentralized designs with application to continuous media flow printing," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 1, pp. 294–304, 2020.

[77] D. J. Hoelzle, A. G. Alleyne, and A. J. Wagoner Johnson, "Basis task approach to iterative learning control with applications to micro-robotic deposition," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 5, pp. 1138–1148, 2011.

[78] F. Boeren, A. Bareja, T. Kok, and T. Oomen, "Frequency-domain ILC approach for repeating and varying tasks: With application to semiconductor bonding equipment," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 6, pp. 2716–2727, 2016.

[79] J.-X. Xu and Y. Tan, "A composite energy function-based learning control approach for nonlinear systems with time-varying parametric uncertainties," *IEEE Transactions on Automatic Control*, vol. 47, no. 11, pp. 1940–1945, 2002.

[80] K. P. . Tee, S. S. Ge, and E. Tay, "Barrier Lyapunov functions for the control of output-constrained nonlinear systems," *Automatica*, vol. 45, pp. 918–927, 2009.

[81] G. Sebastian, Y. Tan, D. Oetomo, and I. Mareels, "On input and output constraints in iterative learning control design for robotic manipulators," *Unmanned Systems*, vol. 06, no. 03, pp. 197–208, 2018.

[82] G. Sebastian, Y. Tan, and D. Oetomo, "A unified analysis tool in iterative learning control: Composite energy function," in *2019 12th Asian Control Conference (ASCC).* IEEE, 2019, pp. 1601–1606.

[83] M. Ketelhut, S. Stemmler, J. Gesenhues, M. Hein, and D. Abel, "Iterative learning control of ventricular assist devices with variable cycle durations," *Control Engineering Practice*, vol. 83, pp. 33–44, 2019.

[84] C. T. Freeman, A.-M. Hughes, J. H. Burridge, P. H. Chappell, P. L. Lewin, and E. Rogers, "Iterative learning control of FES applied to the upper extremity after stroke," *Control Engineering Practice*, vol. 17, no. 3, pp. 368 – 381, 2009.

[85] A.-M. Hughes, C. T. Freeman, J. H. Burridge, P. H. Chappell, P. L. Lewin, and E. Rogers, "Feasibility of iterative learning control mediated by functional electrical stimulation for reaching after stroke," *Neurorehabilitation and Neural Repair*, vol. 23, no. 6, pp. 559 – 568, 2009.

[86] C. T. Freeman, "Newton-method based iterative learning control for robot-assisted rehabilitation using FES," *Mechatronics*, vol. 24, pp. 934–943, 2014.

[87] ——, "Upper limb electrical stimulation using input-output linearization and iterative learning control," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 4, pp. 1546–1554, 2015.

[88] ——, "Robust ILC with application to stroke rehabilitation," *Automatica*, vol. 81, pp. 270–278, 2017.

[89] S. Mandra, K. Galkowski, A. Rauh, H. Aschemann, and E. Rogers, "Iterative learning control for a class of multivariable distributed systems with experimental validation," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 949–960, 2019.

[90] P. Pakshin, S. Mandra, J. Emelianova, E. Rogers, ErwinskiK., and K. Galkowski, "Experimentally validated vector Lyapunov function-based iterative learning control design under input saturation," *IEEE Transactions on Control Systems Technology*, vol. 32, no. 1, pp. 189–201, 2024.

[91] T. Oomen, "Advanced motion control for precision mechatronics: Control, identification, and learning of complex systems," *IEEJ Journal of Industry Applications*, vol. 7, no. 2, pp. 127–140, 2018.

[92] Y. Zhang, "Iterative learning control: from model-based to reinforcement learning," Ph.D. dissertation, University of Southampton, 2023.

[93] M. Poot, J. Portegies, and T. Oomen, "On the role of models in learning control: Actor-critic iterative learning control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1450–1455, 2020.

[94] M. Scheel, A. Berndt, and O. Simanski, "Iterative learning control: An example for mechanical ventilated patients," *IFAC-PapersOnLine*,

vol. 48, no. 20, pp. 523–527, 2015, 9th IFAC Symposium on Biological and Medical Systems BMS 2015.

[95] J. Reinders, M. Giaccagli, B. Hunnekens, D. Astolfi, T. Oomen, and N. van de Wouw, “Repetitive control for Lur’e-type systems: Application to mechanical ventilation,” *IEEE Transactions on Control Systems Technology*, vol. 31, no. 4, pp. 1819–1829, 2023.