Point-to-Point Iterative Learning Control with Optimal Tracking Time Allocation

Yiyang Chen, Bing Chu* and Christopher T. Freeman

***Abstract*—Iterative learning control is a high performance tracking control design method for systems operating in a repetitive manner. This paper proposes a novel design methodology that extends the recently developed point-to-point iterative learning control framework to allow automatic via-point time allocation within a given point-to-point tracking task, leading to significant performance improvements, e.g. energy reduction. The problem is formulated into an optimization framework with via-point temporal constraints and a reference tracking requirement, for which a two stage design approach is developed. This yields an algorithmic solution which minimizes input energy based on norm optimal iterative learning control and gradient minimization. The algorithm is further expanded to incorporate system constraints into the design, prior to experimental validation on a gantry robot test platform to confirm its feasibility in practical applications.**

***Index Terms*—point to point iterative learning control, constraint handling.**

I. Introduction

ITERATIVE learning control (ILC) is a high performance control technique applicable to systems which perform repeated tasks. Unlike modifying the controller as in adaptive control, ILC directly updates the input based on information from previous experimental attempts (termed trials) to improve tracking performance. Each trial has the same finite duration, and the system states are reset to the same value at the end of each trial. The tracking error can theoretically be reduced to zero after sufficient trials. This appealing property has led to ILC being applied to various industrial high performance systems, such as robotics [1], [2], chemical batch processing [3], [4] and stroke rehabilitation [5]. See [6], [7] and [8] for a detailed overview of ILC.

In the classic ILC setting, the output of the system is required to track a given reference defined on the whole trial interval. However, in a large subset of control tasks such as robotic pick-and-place manipulation, the system output is only critical at a finite number of time-points along the trial duration. To address this design problem, the ILC framework can be modified to update the input by using only the error information recorded at these time-points. Significant design freedom can hence be exploited to incorporate additional performance objectives by eliminating the unnecessary noncritical tracking constraints. This novel control concept is termed point-to-point ILC [9], and has attracted significant interest.

A number of point-to-point ILC algorithms have been proposed in the literature. Terminal ILC, a special case of the point-to-point ILC problem where only tracking performance at the end of the trial is required, is studied in [10]–[13]. General point-to-point ILC design by employing a 'complete' reference that passes through all the desired intermediate points is studied in [14]–[16]. These methods, however, do not fully exploit the extra freedom provided by the point-to-point tracking requirements. As such, the overall system performance could be limited. This drawback is addressed recently in [17]–[20] where the intermediate point tracking requirements are directly handled by optimizing a quadratic performance index characterizing the tracking performance at these intermediate points. Results containing convergence properties of these algorithms are also available. More recently, system constraints in point-to-point ILC have been considered [21]–[23].

All the aforementioned point-to-point ILC problem formulations in [9]–[23] have assumed that the critical tracking time-points are known *a priori*, and this information is generally embedded within a performance cost function whose optimization is implemented in the ILC framework. Hence the performance cost function is highly dependent on the tracking time allocation within the point-to-point ILC tracking problem. If this framework can be expanded to enable the tracking time allocation to be embedded as an optimized variable, significant practical benefit can be realized, such as reducing the energy used, reducing the damage to machine components and increasing the efficiency of production (i.e. throughput). This hence motivates the expansion of the point-to-point ILC framework to allow flexibility in the selection of the temporal tracking subset, with its input also updated to achieve the overall point-to-point control objective. Note that existing research into optimal tracking time allocation of point-to-point robotic motion [24] addresses a series of independent motions, but these are not coupled together and the approach does not take advantage of ILC to enable precise tracking.

To clearly illustrate the above motivation, consider the robotic pick-and-place task shown in Figure 1. In this problem, the robotic arm is required to start from the resting position (shown as green dot) at the beginning of a trial ($t = 0$), move to the 'pick' position (shown as yellow dot) at the specified time t_1 and then move to the 'place' position (shown as red dot) at a specified time t_2, before finally moving back (resetting) to the resting position at the end of the trial ($t = T$). Note that in this problem, the pre-specified tracking time allocation, i.e. the 'pick' and 'place' time instants t_1 and t_2, plays a key role in the system performance. As an

Y. Chen, B. Chu and C. T. Freeman are with the Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (yc12u12@soton.ac.uk; b.chu@soton.ac.uk; cf@ecs.soton.ac.uk).

*The corresponding author (e-mail: b.chu@soton.ac.uk).

The material in this paper was partially presented at the 54th IEEE Conference on Decision and Control, Osaka, Japan, December 2015.

intuitive example, if the two tracking time instants are chosen close to each other, the robotic arm will have to move from the desired 'pick' location to the 'place' position within a very short time, therefore requiring fast moving and thus high power consumption (as an example see the difference in the the two trajectories shown in the figure). In all existing point-to-point ILC designs, these critical tracking time instants are assumed to be known as *a priori*. The extra flexibilities in choosing the tracking time instants (which affect the system performance) have not been explored.

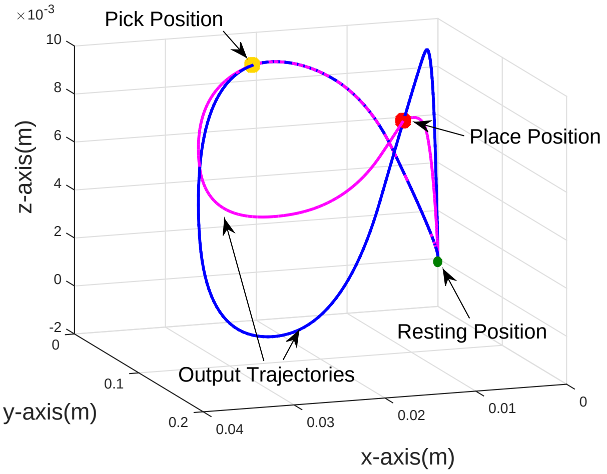

Fig. 1. A Robotic 'Pick-and-Place' Tracking Example.

This paper develops a comprehensive optimal tracking time allocation framework in point-to-point ILC to allow automatic selection of the tracking time to optimize some performance of interest and, at the same time, achieving high performance reference tracking at the chosen intermediate points based on our preliminary idea in [25]. The main contributions of the paper are as follows:

- *Rigorous formulation of point-to-point ILC problem with optimal tracking time allocation (Section II).* The design problem is formulated into an optimization framework where the flexibility in tracking time allocation is exploited to simultaneously optimize some performance index of interest, e.g. energy consumption to ensure accuracy tracking. The problem formulation is based on an abstract operator form in some Hilbert space, which allows the essence of the results to be generalized to other system models without difficulty.
- *Derivation of a two stage design framework (Sections III and IV).* A two stage design framework is proposed to solve the optimal tracking time allocation optimization problem. The two stage design involves the alternating use of a well known norm optimal point-to-point ILC algorithm [17] ensuring accurate tracking, and a gradient based minimization to update the tracking time allocation to optimize the performance index. Under certain conditions, the proposed algorithm converges to the 'best' solution that can be achieved. Implementation procedures of the proposed algorithm are discussed in detail.
- *Incorporation of system constraints into the design (Section V).* The two stage design framework is further extended to incorporate system constraints that exist widely in practice. In particular, it is shown that the input constraints can be incorporated into the design using a modified two stage design framework and the resulting algorithm guarantees that the constraint is satisfied, as well as improved performance.
- *Experimental verification on a gantry robot test platform (Section VI).* The proposed design methods are verified experimentally on a gantry robot test platform. The results show that by exploiting the flexibility in choosing the tracking time allocation in point-to-point ILC, significant benefit can be obtained in terms of the input energy reduction compared to *a priori* tracking time allocation, at the same time maintaining high tracking performance. The results also show that the proposed algorithm exhibits a degree of robustness against modelling mismatch/error due to the use of previous data which is clearly desirable in practical applications.

The notation used in this paper is standard: $\mathbb{N}$ is the set of non-negative integers; $\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ denote the sets of n dimensional real vectors and $n\times m$ real matrices respectively; $\mathbb{S}^n_{++}$ is the set of all $n\times n$ real positive definite matrices; $L_2^\ell[0,\ T]$ denotes the space of functions defined on $[0,\ T]$ whose function value belongs to $\mathbb{R}^\ell$ and 2 power is Lebesgue integrable; $\langle x,\ y\rangle$ is the inner product of x and y in some Hilbert space; $\mathbb{X}\times\mathbb{Y}$ is the Cartesian product of two spaces $\mathbb{X}$ and $\mathbb{Y}$; $P_\Theta(x)$ denotes the projection x to the set Θ in some Hilbert space. Other notation will be introduced as needed.

II. Formulation of the Problem

In this section, the design problem is formulated rigorously into an optimization problem using an abstract operator form representation of system dynamics in some Hilbert space. In this paper, a continuous time linear time-invariant state space model is considered. The general abstract form problem formulation allows the results to be extended to more general models, e.g. time varying systems, differential delay systems and switched system.

A. Point-to-Point ILC Framework

Consider the following m-output, ℓ-input linear continuous time-invariant system in state space form $S(A,B,C)$

$$\begin{aligned} \dot{x}_k(t) &= Ax_k(t)+Bu_k(t), & x_k(0)&=x_0,\\ y_k(t) &= Cx_k(t), & t&\in[0,\ T], \end{aligned} \tag{1}$$

where t is the time; $k\in\mathbb{N}$ denotes the trial number; $x_k(t)\in\mathbb{R}^n, u_k(t)\in\mathbb{R}^\ell$ and $y_k(t)\in\mathbb{R}^m$ are the state, input and output respectively on trial k; A,B,C are system matrices of compatible dimensions; $0<T<\infty$ is the trial length. The initial conditions are identical for all trials, i.e. $x_k(0)=x_0,\ \forall k$.

To facilitate the problem formulation, an abstract description of the system dynamics is introduced first. Note that system (1) can be represented in an equivalent operator form

$$y_k = Gu_k + d. \tag{2}$$

In this operator form, $u_k \in L_2^\ell[0,T]$, $y_k \in L_2^m[0,T]$, are the system input and output, in which $L_2^\ell[0,T]$ and $L_2^m[0,T]$ are the input and output Hilbert spaces equipped with the inner products and associated induced norms

$$\langle u,\ v\rangle_R = \int_0^T (u(t))^\top Rv(t)\mathrm{d}t,\ \ \|u\|_R = \sqrt{\langle u,\ u\rangle_R} \quad (3)$$

$$\langle x,\ y\rangle_S = \int_0^T (x(t))^\top Sy(t)\mathrm{d}t,\ \ \|y\|_S = \sqrt{\langle y,\ y\rangle_S} \quad (4)$$

respectively where $R \in \mathbb{S}_{++}^\ell$ and $S \in \mathbb{S}_{++}^m$ are positive definite matrices with appropriate dimensions; $G : L_2^\ell[0,T] \rightarrow L_2^m[0,T]$ is the system operator and $d \in L_2^m[0,T]$ represents the effect of initial condition, taking the following forms

$$(Gu_k)(t) = \int_0^t Ce^{A(t-s)}Bu_k(s)\mathrm{d}s,\ d(t) = Ce^{At}x_0. \quad (5)$$

For notational simplicity and without loss of generality, it is assumed that $x_0 = 0$ and thus $d = 0$.

In the point-to-point ILC framework, we are only interested in the system output $y(t)$ at a finite number of pre-specified time instants (of interest), $t_i,\ i = 1,\dots,M$ in vector form as

$$\Lambda = [t_1,\ t_2,\dots,\ t_M]^\top \quad (6)$$

Also denote the desired tracking reference at these time instants as

$$r^p = [r_1,\ r_2,\dots,r_M]^\top \in \mathbb{R}^{Mm},$$

where r_i, $i = 1,\dots,M$ are the desired tracking requirements for $i^{th}, i = 1,\dots,M$ intermediate points, respectively. To extract the system output value at Λ, introduce the linear mapping $\zeta \in L_2^m[0,\ T] \mapsto \zeta^p \in H$ defined as

$$\zeta^p = [\zeta(t_1),\ \zeta(t_2),\dots,\zeta(t_M)]^\top \quad (7)$$

where H is a Hilbert space

$$H = \underbrace{\mathbb{R}^m \times \cdots \times \mathbb{R}^m}_{\text{M times}} \quad (8)$$

with respective inner product and induced norm

$$\langle \hat{x},\ \hat{y}\rangle_{[Q]} = \sum_{i=1}^M \hat{x}_i^\top Q_i \hat{y}_i,\ \ \|\hat{y}\|_{[Q]} = \sqrt{\langle \hat{y},\ \hat{y}\rangle_{[Q]}} \quad (9)$$

in which

$$\hat{x} = [\hat{x}_1,\ \hat{x}_2,\dots,\ \hat{x}_M]^\top \in H,\ \hat{y} = [\hat{y}_1,\ \hat{y}_2,\dots,\ \hat{y}_M]^\top \in H$$

and $[Q]$ denotes the data set $\{Q_1,\ Q_2,\dots,\ Q_M\}$ where for $i = 1,\dots,M$ each $Q_i \in \mathbb{S}_{++}^m$ is a positive definite matrix. Using this notation, the plant output corresponding to tracking time allocation Λ is given by

$$y^p = (Gu)^p.$$

Since G is linear, this can be further written as

$$y^p = G_\Lambda^p u = \begin{bmatrix} G_1 u \\ \vdots \\ G_M u \end{bmatrix} \quad (10)$$

where $G_\Lambda^p : L_2^\ell[0,\ T] \rightarrow H$ is a linear operator with each component $G_i : L_2^\ell[0,\ T] \rightarrow \mathbb{R}^m$ defined by

$$G_i u = \int_0^{t_i} Ce^{A(t_i - t)}Bu(t)\mathrm{d}t. \quad (11)$$

The tracking error at tracking time allocation Λ is therefore

$$e^p = r^p - y^p.$$

B. Point-to-Point ILC with Optimal Tracking Time Allocation

We are now ready to formulate the problem of point-to-point ILC with optimal tracking time allocation: Firstly, we consider the tracking time allocation as a variable, i.e. $\Lambda \in \Theta$, where Θ is the admissible set of tracking time allocation

$$\Theta = \{\Lambda \in \mathbb{R}^M : 0 < t_1^- \leqslant t_1 \leqslant t_1^+ \leqslant t_2^- \leqslant t_2 \leqslant t_2^+ \leqslant \dots \leqslant t_M^+ = T\} \quad (12)$$

in which $[t_i^-,\ t_i^+]$ defines the (allowed) allocation interval for t_i representing the requirements on enforcing process timing and synchronization constraints necessary to complete the task.

The **Point-to-Point ILC with Optimal Tracking Time Allocation Problem** can then be defined as iteratively finding a tracking time allocation Λ_k and an input u_k with the asymptotic property that the output values at these time instants, i.e. y_k^p, accurately pass through a set of desired points r^p with

$$\lim_{k\to\infty} y_k^p = r^p,$$

at the same time minimizing a target cost function $f(u, y)$ as a function of the system input u and output y, i.e.

$$\lim_{k\to\infty} (u_k,\ y_k,\ \Lambda_k) = (u_k^*,\ y_k^*,\ \Lambda_k^*)$$

where u_k^*, y_k^* and Λ_k^* are optimal solutions of the problem

$$\begin{aligned} \underset{u,y,\Lambda}{\text{minimize}} \quad & f(u,y) \\ \text{subject to} \quad & r^p = G_\Lambda^p u, \\ & y = Gu, \\ & \Lambda \in \Theta. \end{aligned} \quad (13)$$

Note that this problem formulation comprises a significant expansion of the current point-to-point ILC framework by exploiting the flexibilities in choosing the tracking time allocation Λ to optimize some performance of interest in addition to the tracking requirement. This however, as will be seen later, introduces substantial difficulties in algorithm design, which will be addressed in the following sections.

Remark 1. The index $f(u, y)$ represents our requirements on the performance and should be chosen according to the specific application. As an example, if we would like to minimize the control input energy, $f(u, y)$ can be chosen as

$$f(u, y) = \|u\|_R^2;$$

if we would like the system to minimize a function of the output, e.g. acceleration of a robotic movement, $f(u, y)$ can be chosen as

$$f(u, y) = \|g(y)\|_S$$

where the function $g(y)$ computes the output acceleration. In this paper for simplicity, the performance index $f(u, y)$ is chosen as a convex function of u and y. Note that this encompasses many real life applications.

Remark 2. Note that the optimization problem (13) does not necessary have a unique solution. When there are more than one solutions, they all produce the same performance value and therefore are considered equally 'good'. If a certain solution is regarded as better than the others, then an extra term has to be included in the performance index to ensure this particular solution will be chosen when solving the optimization problem.

Remark 3. It is worth pointing out that the general problem formulation in Hilbert space makes it possible for the techniques used in this paper to be further extended to other systems, e.g. linear discrete time systems and switched linear systems, the details of which however will differ and are not described in this paper.

III. A TWO STAGE DESIGN FRAMEWORK

In this section, a two stage design framework is developed to solve the above point-to-point ILC design with optimal tracking time allocation problem. Note that while the tracking time allocation Λ does not explicitly appear in the performance index $f(u, y)$, they are connected by the tracking requirement $G^p_\Lambda u = r^p$ in a nonlinear manner. Furthermore, the input u lies in an infinite dimensional space $L^\ell_2[0, T]$ and the tracking time allocation Λ lies in the finite dimensional space Θ. All these make the problem (13) non-trivial.

A. Framework Description

Optimization problem (13) can be written equivalently as

$$\min_{\Lambda\in\Theta}\left\{\min_{u} f(u,y),\ \text{subject to } G^p_\Lambda u = r^p,\ y = Gu\right\} \tag{14}$$

by optimizing over u first and then optimizing over Λ. Define the function $\tilde{f}$ of Λ by

$$\tilde{f}(\Lambda) = \min_{u}\left\{f(u,y),\ \text{subject to } G^p_\Lambda u = r^p,\ y = Gu\right\},$$

and denote a global minimizer for u of the inner optimization problem (14) as $u_\infty(\Lambda) : \Theta \to L^\ell_2[0, T]$, the optimization problem (14) is then equivalent to

$$\min_{\Lambda\in\Theta}\{\tilde{f}(\Lambda) := f(u_\infty(\Lambda), Gu_\infty(\Lambda))\}. \tag{15}$$

It follows that the point-to-point ILC with optimal tracking time allocation problem can be solved using the following two stage design framework:

- *Stage One:* Fix the tracking time allocation Λ and solve the inner optimal problem (14), i.e.

$$\begin{aligned} \underset{u,y}{\text{minimize}} \quad & f(u,y) \\ \text{subject to} \quad & r^p = G^p_\Lambda u, \\ & y = Gu. \end{aligned} \tag{16}$$

- *Stage Two:* Substitute the solution $u_\infty(\Lambda)$ of the problem (16) into the original optimization problem (14) and then solve the resulting optimization problem (15) to compute the optimal tracking time allocation, i.e.

$$\min_{\Lambda\in\Theta}\{\tilde{f}(\Lambda) := f(u_\infty(\Lambda), Gu_\infty(\Lambda))\}. \tag{17}$$

To exemplify the approach, the input energy consumption is selected to be the target performance index in this paper, so that $f(u, y) = \|u\|^2_R$. This guarantees the existence of a unique global minimizer for the inner optimization problem within (14), and the resulting optimization problems in Stage One and Two become

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & \|u\|^2_R \\ \text{subject to} \quad & r^p = G^p_\Lambda u, \end{aligned} \tag{18}$$

and

$$\min_{\Lambda\in\Theta}\{\tilde{f}(\Lambda) := \|u_\infty(\Lambda)\|^2_R\} \tag{19}$$

respectively. Note that as the output y does not appear in the performance index, therefore the second constraint in problem (16), i.e. $y = Gu$, is not needed in the optimization problem (18). It is worth pointing out that other (convex) performance indices rather than the input energy can also be used with no changes in the form of the two stage design framework - the implementation of the resulting algorithms however will differ from those described in subsequent sections of this paper.

As dictated by the ILC framework, the two stages must be achieved using experimental data in order to embed robustness against model uncertainties. Before this is discussed in detail in next section, the solution of this two stage design framework is given below.

B. Solution of the Proposed Framework

1) Solution of Stage One Optimization Problem: For a given tracking time allocation Λ, the Stage One optimization problem is in fact a point-to-point ILC design problem with a minimum control energy requirement. This can be solved efficiently using the point-to-point norm optimal ILC algorithm with a special initial input choice, as shown next.

Theorem 1. If the system $S(A, B, C)$ is controllable and C has full row rank, the Stage One optimization problem (18) for a given tracking time allocation Λ can be solved by the norm-optimal point-to-point ILC algorithm

$$u_{k+1} = \arg\min_{u}\{\|e^p\|^2_{[Q]} + \|u - u_k\|^2_R\} \tag{20}$$

proposed in [17] with initial input $u_0 = 0$, such that

$$u_\infty = \lim_{k\to\infty} u_k.$$

The iterative solution is given by

$$u_{k+1} = u_k + G^{p*}_\Lambda(I + G^p_\Lambda G^{p*}_\Lambda)^{-1}e^p_k \tag{21}$$

where $G^{p*}_\Lambda : (\omega_1, ..., \omega_M) \in H \to u \in L^\ell_2[0, T]$ is the Hilbert adjoint operator of G^p_Λ defined by

$$\begin{aligned} & u(t) = R^{-1}B^\top p(t),\ \dot{p}(t) = -A^\top p(t), \\ & p(T) = 0,\ p(t_i-) = p(t_i+) + C^\top Q_i\omega_i,\ i = 1, \ldots, M \end{aligned} \tag{22}$$

with p denoting the costate and R, Q_i denoting the weighting matrices, and the $Mm \times Mm$ matrix $G^p_\Lambda G^{p*}_\Lambda$ has a block structure with $(i,j)^{th}$ block

$$G_i G_j^* = \int_0^{min(t_i,t_j)} Ce^{A(t_i-t)}BR^{-1}B^\top e^{A^\top(t_i-t)}C^\top Q_j \mathrm{d}t. \tag{23}$$

Furthermore, an analytic solution can be obtained for $u_\infty(\Lambda)$ as follows

$$u_\infty(\Lambda) = G^{p*}_\Lambda (G^p_\Lambda G^{p*}_\Lambda)^{-1} r^p. \tag{24}$$

Proof. See Appendix A for the proof of Theorem 1. □

Note that the system controllability condition can be satisfied without difficulty as a controllable state space model can always be constructed for a given system and the requirement C has full row rank is not restrictive either as this simply implies no output component can be constructed from others, i.e. there is no redundant output, and is therefore assumed to hold for the rest of the paper.

2) Solution of Stage Two Optimization Problem: With the analytical solution of Stage One optimization problem, the Stage Two optimization problem (19) can be further reduced as shown in the following lemma.

Lemma 1. Based on the analytical solution (24) of Stage One optimization problem, the Stage Two optimization problem (19) can be expressed as

$$\min_{\Lambda\in\Theta} \|u_\infty(\Lambda)\|^2_R = \min_{\Lambda\in\Theta} \ \left\langle r^p,\ (G^p_\Lambda G^{p*}_\Lambda)^{-1} r^p \right\rangle_{[Q]}. \tag{25}$$

Proof. See Appendix B for the proof of Lemma 1. □

Solving the above Stage Two optimization problem, however, is non-trivial except for the special case of $M = 1$, i.e. there is only one tracking point, where the solution can be obtained analytically, as shown in the following theorem.

Theorem 2. When there is only one tracking point, i.e. $M = 1$, the solution of Stage Two optimization problem (25) is

$$\Lambda^* = T.$$

The corresponding minimum energy is

$$\min_{\Lambda\in\Theta} \|u_\infty(\Lambda)\|^2_R = \left\langle r^p,\ \Psi_T^{-1} r^p \right\rangle_{[Q]}$$

where

$$\Psi_t = \int_0^t Ce^{A(t-s)}BR^{-1}(Ce^{A(t-s)}B)^\top \mathrm{d}s.$$

Proof. See Appendix C for the proof of Theorem 2. □

Theorem 2 shows that when $M = 1$, $\Lambda^* = T$ is always an optimal choice in terms of minimizing the control input energy - this is not surprising as this allows the system output to change gradually to the desired position and thus less control energy could be expected. However, when $M > 1$, the performance index is generally non-linear and non-convex with respect to the tracking time allocation Λ leading to significant difficulties in solving Stage Two optimization problem (25). This is addressed in the following theorem using a gradient based algorithm.

Theorem 3. For $M \geqslant 2$, Stage Two optimization problem (25) can be solved using the gradient based iterative method

$$\Lambda_{j+1} = P_\Theta(\Lambda_j - \gamma_j \cdot \nabla \tilde{f}(\Lambda_j)) \tag{26}$$

where $j \in \mathbb{N}$ denotes the updating iteration (loop) number, $\nabla \tilde{f}(\Lambda_j) \in \mathbb{R}^M$ is the gradient of the function $\tilde{f}$, $P_\Theta(\cdot)$ denotes the projection operator, i.e.

$$P_\Theta(x) = \arg\min_{z\in\Theta} \|x - z\|^2,$$

and $\gamma_j > 0$ is a step size chosen by the generalized Armijo rule, i.e.

$$\gamma_j = \beta^{m_k} \cdot \gamma \tag{27}$$

where m_k is the smallest non-negative integer such that

$$\tilde{f}(\Lambda_{j+1}) - \tilde{f}(\Lambda_j) \leqslant \sigma(\nabla \tilde{f}(\Lambda_j))^\top (\Lambda_{j+1} - \Lambda_j) \tag{28}$$

and σ, β, γ are constant scalars with $0 < \sigma < 1$, $0 < \beta < 1$, $\gamma > 0$. The resulting sequence $\{\tilde{f}(\Lambda_j)\}$ converges downward to a limit $\tilde{f}^*$, i.e.

$$\tilde{f}(\Lambda_{j+1}) \leqslant \tilde{f}(\Lambda_j), \text{ and } \lim_{j\to\infty} \{\tilde{f}(\Lambda_j)\} = \tilde{f}^* \tag{29}$$

and the sequence $\{\Lambda_j\}$ satisfies

$$\lim_{j\to\infty} \|\Lambda_j - \Lambda_{j+1}\| = 0 \tag{30}$$

with every limit point z of the sequence $\{\Lambda_k\}$ is a stationary point for problem (25), i.e.

$$z = P_\Theta(z - \nabla \tilde{f}(z)). \tag{31}$$

Proof. See Appendix D for the proof of Theorem 3. □

It is noted that being a stationary point satisfying (31) is a necessary condition of a (possibly locally) minimum point. Note that the function $\tilde{f}(\Lambda)$ is bounded below and Θ is a compact set. Therefore, the (global) minimum of the optimization problem exists and is a stationary point. When the problem only has one such point, it must be the minimum. In this case, the above algorithm converges to the global minimum solution following results in [26], i.e. the best result that can be achieved.

Remark 4. It is worth pointing out that other step size choices are also possible, e.g. constant step size [26]

$$0 < \mu \leqslant \gamma \leqslant \frac{2(1-\mu)}{L}, \tag{32}$$

where $L > 0$ is the Lipschitz constant of $\tilde{f}(\Lambda)$ on Θ and $\mu \in (0,\ 2/(2+L)]$ is a positive scalar, and the projected Barzilai-Borwein step size [27]

$$\gamma_j = \frac{\langle \Delta x_j,\ \Delta g_j \rangle}{\langle \Delta g_j,\ \Delta g_j \rangle}, \text{ or } \gamma_j = \frac{\langle \Delta x_j,\ \Delta x_j \rangle}{\langle \Delta x_j,\ \Delta g_j \rangle} \tag{33}$$

where $\Delta x_j = \Lambda_j - \Lambda_{j-1}$, $\Delta g_k = \nabla \tilde{f}(\Lambda_j) - \nabla \tilde{f}(\Lambda_{j-1})$. Using these step size choices, the convergence properties will be different from those stated in the above theorem and are omitted here for brevity.

IV. Implementation of the Design Approach

In previous section, a two stage design framework was proposed. Its implementation is now discussed in detail.

A. Implementation of Stage One Design

The general update (21) of Stage One design can be either computed directly using the analytical solution (24), or implemented experimentally using the following feedback plus feedforward algorithm.

Proposition 1. The Stage One update (21) can be implemented using the feedforward plus feedback implementation

$$u_{k+1}(t) = u_k(t) + R^{-1}B^\top p_k(t) \tag{34}$$

with

$$p_k(t) = -K(t)(x_{k+1}(t) - x_k(t)) + \xi_{k+1}(t) \tag{35}$$

where with p_k denotes the costate, $K(t)$ denotes the Riccati feedback matrix

$$\begin{aligned} &K(T) = 0,\ \ K(t_i-) = K(t_i+) + C^\top Q_i C,\ \ i = 1, \dots, M,\\ &0 = \dot{K}(t) + (A^\top - K(t)BR^{-1}B^\top)K(t) + K(t)A \end{aligned} \tag{36}$$

and $\xi_{k+1}(t)$ denotes the predictive feedforward term given at the $(k+1)^{th}$ trial by

$$\begin{aligned} &0 = \dot{\xi}_{k+1}(t) + (A^\top - K(t)BR^{-1}B^\top)\xi_{k+1}(t),\ \ \xi_{k+1}(T) = 0,\\ &\xi_{k+1}(t_i-) = \xi_{k+1}(t_i+) + C^\top Q_i e_k(t_i),\ \ i = 1, \dots, M. \end{aligned} \tag{37}$$

Proof. See Appendix E for the proof of Proposition 1. □

It it worth pointing out that although both implementation methods can solve the Stage One optimization problem, the feedback plus feedforward implementation embeds robust performance (due to the introduction of state feedback) when applied to the true plant (more details can be found in [17]), and therefore is preferable in practice.

B. Implementation of Stage Two Design

The Stage Two gradient based design method (26) involves two steps: a gradient update step and a projection step. The gradient update step is

$$\tilde{\Lambda}_{j+1} = \Lambda_j - \gamma_j \cdot \nabla \tilde{f}(\Lambda_j)$$

where the selection of γ_j is dictated by (27) and $\tilde{\Lambda}_{j+1}$ denotes the intermediate solution obtained for the gradient update step at the j^{th} loop. The gradient can either be computed analytically using (25), or using a computationally efficient estimation

$$\left.\frac{\partial \tilde{f}}{\partial t_i}\right|_{\Lambda_j} = \frac{\tilde{f}(\Lambda_j^{i+}) - \tilde{f}(\Lambda_j^{i-})}{2\Delta T} \tag{38}$$

where $\Lambda_j^{i+} = [t_1^j,\ t_2^j, \dots,\ t_i^j + \Delta T, \dots,\ t_M^j]^\top$ and $\Lambda_j^{i-} = [t_1^j,\ t_2^j, \dots,\ t_i^j - \Delta T, \dots,\ t_M^j]^\top$, and $\Delta T \in \mathbb{R}$ is a sufficiently small number. It should be noted that both analytic calculation and experimental testing can be used to compute the fixed tracking time allocation's optimal energy $\tilde{f}(\Lambda_j^{i+})$ and $\tilde{f}(\Lambda_j^{i-})$ in (38). The projection step is performed to obtain the tracking time allocation Λ_{j+1} for the next loop based on $\tilde{\Lambda}_{j+1}$, i.e.

$$\Lambda_{j+1} = P_\Theta(\tilde{\Lambda}_{j+1}) = \arg\min_{\Lambda \in \Theta} \|\Lambda - \tilde{\Lambda}_{j+1}\|.$$

Algorithm 1 Greedy initial tracking time allocation

```
Input: system state space model $S(A, B, C)$, desired tracking reference $r^p$, central initial tracking time allocation $\Lambda_0^c$ and admissible set of tracking time allocation $\Theta$
Output: Greedy initial tracking time allocation $\Lambda_0^g$
for $i = 1$ to $M$ do
    Let $\Lambda_i = [t_1^*,\ t_2^*, \dots,\ t_i, \dots,\ t_M^c]^\top$.
    Solve the following problem using Theorem 3
        $t_i^* = \arg\min_{t_i} \tilde{f}(\Lambda_i).$   (41)
end for
return $\Lambda_0^g = [t_1^*,\ t_2^*, \dots,\ t_M^*]^\top$
```

Note that this can be formulated into the following quadratic programming (QP) problem

$$\begin{aligned} &\underset{\Lambda}{\text{minimize}} && \|\Lambda - \tilde{\Lambda}_{j+1}\|^2\\ &\text{subject to} && \hat{A}\Lambda - b \preceq 0 \end{aligned}$$

where $\hat{A} = [I, -I]^\top$, $b = [t_1^+, ..., t_M^+, -t_1^-, ..., -t_M^-]^\top$ and the symbol $\preceq$ denotes the component-wise inequality. This QP problem can be solved efficiency using standard QP solvers, e.g. using Matlab function `quadprob`.

As an iterative algorithm, the choice of initial tracking time allocation Λ_0 may affect the algorithm's convergence performance, as in most non-linear and non-convex optimization problems. Therefore, three methods are now proposed to provide an appropriate initial tracking time allocation for the algorithm to get better system performance.

1) Central initial tracking time allocation: In this method, all the initial tracking points are specified in the center of their time intervals, and the initial tracking time allocation is hence

$$\Lambda_0^c = [t_1^c,\ t_2^c, \dots,\ t_M^c]^\top \tag{39}$$

where $t_i^c = (t_i^- + t_i^+)/2$.

2) Greedy initial tracking time allocation: This method is defined by Algorithm 1 which takes M 'greedy' steps to obtain the greedy initial tracking time allocation

$$\Lambda_0^g = [t_1^g,\ t_2^g, \dots,\ t_M^g]^\top \tag{40}$$

based on the central initial tracking time allocation Λ_0^c. Each 'greedy' step only computes a single optimal time-point t_i^*, and the other time-points are known as constants, i.e. $t_1^*, \dots, t_{i-1}^*$ and $t_{i+1}^c, \dots, t_M^c$. Algorithm 1 requires some computations but tries to provide a better initial tracking time allocation Λ_0^g rather than Λ_0^c.

3) Low resolution initial tracking time allocation: Low resolution initial tracking time allocation can be implemented by using Algorithm 2, which involves performing a grid search in order to approximate the optimal tracking time allocation based on the nominal plant model. The solution is denoted as

$$\Lambda_0^l = [t_1^l,\ t_2^l, \dots,\ t_M^l]^\top \tag{42}$$

Algorithm 2 Low resolution initial tracking time allocation

```
Input: system state space model $S(A,B,C)$, desired tracking reference $r^p$, admissible set of tracking time allocation $\Theta$ and sample time $T_s$
Output: Low resolution initial tracking time allocation $\Lambda_0^l$
Discretize the infinite set $\Theta$ at a sample rate of $T_s$ to obtain
    $\tilde{\Theta} = \{\Lambda \in \mathbb{R}^M : 0 < t_1^- \leqslant t_1 \leqslant t_1^+ \leqslant t_2^- \leqslant t_2 \leqslant t_2^+ \leqslant \ldots \leqslant t_M^+ = T,\ t_i = n_i T_s,\ n_i \in \mathbb{N},\ i = 1,\ldots,M\}$
    which is a finite subset of $\Theta$.
Solve the optimization problem below using blind search
    $\Lambda^* = \arg\min_{\Lambda \in \tilde{\Theta}} \tilde{f}(\Lambda). \quad (43)$
return $\Lambda_0^l = \Lambda^*$
```

Algorithm 3 Point-to-point ILC with optimal tracking time allocation

```
Input: initial tracking time allocation $\Lambda_0$, system state space model $S(A,B,C)$, desired tracking reference $r^p$, admissible set of tracking time allocation $\Theta$, weighting matrices $R$ and $Q_i$
Output: Optimal tracking time allocation $\Lambda_{opt}$ and corresponding input $u_{opt}$
1: initialization: Loop number $j = 0$
2: Repeatedly implement Stage One update (21) with $\Lambda = \Lambda_0$ experimentally using feedback plus feedforward update (34) until convergence, i.e. $\|e_k^p\| < \epsilon\|r^p\|$; record converged input $u_\infty^{ex}(\Lambda_0)$ and input energy $\tilde{f}(\Lambda_0)$.
3: repeat
4:     Compute the gradient using (25) or (38) with $r^p = G_{\Lambda_j}^p u_\infty^{ex}(\Lambda_j)$; implement Stage Two update (26) .
5:     Set $j \to j+1$.
6:     Repeatedly implement Stage One update (21) with $\Lambda = \Lambda_j$ experimentally using feedback plus feedforward update (34) until convergence, i.e. $\|e_k^p\| < \epsilon\|r^p\|$; record converged input $u_\infty^{ex}(\Lambda_j)$ and input energy $\tilde{f}(\Lambda_j)$.
7: until $\left|\tilde{f}(\Lambda_j) - \tilde{f}(\Lambda_{j-1})\right| < \delta \left|\tilde{f}(\Lambda_{j-1})\right|$
8: return $\Lambda_{opt} = \Lambda_j$ and $u_{opt} = u_\infty^{ex}(\Lambda_j)$
```

which minimizes the performance index. The term low resolution implies that the sampling time T_s is suitably large, and hence the total number of time-point combinations, i.e. number of elements in $\tilde{\Theta}$, should not be excessive. Therefore this method can balance computation time and accuracy in approximating the global solution. However, this method may require a significant amount of time to carry out the grid search procedure when the number of time-points is large.

C. An Iterative Implementation Algorithm

Combining the implementation of Stage One and Stage Two designs leads to an iterative implementation of the two stage design framework - Algorithm 3. Note that Λ_0 a suitably chosen initial tracking time allocation, and $\epsilon > 0$, $\delta > 0$ are small scalars which depend on the tracking precision requirement and performance requirement, respectively.

It is essential to note that in Algorithm 3, we require that Step 2 and 6 (i.e. the norm-optimal point-to-point ILC algorithm) are implemented experimentally and Step 4 uses experimental data $u_\infty^{ex}(\Lambda_j)$. These requirements are not necessary when an accurate system model is known. However when there exists model mismatch/uncertainty, the proposed algorithm embeds appealing robustness properties as the algorithm 'learns' information concerning the real plant dynamics through exploitation of experimental data. This will be further demonstrated in subsequent experimental results.

V. CONSTRAINED INPUT CONDITION HANDLING

The previous sections propose a two stage design approach for point-to-point ILC with optimal tracking time allocation. This section further extends the proposed method to incorporate system constraints into the design.

A. Optimal Tracking Time Allocation with System Constraints

In practice, constraints exist widely in control systems due to physical limitations or performance requirements. For example, input constraints typically assume the forms:

- *Input saturation constraint:*
$$\Omega = \{u(t) \in \mathbb{R}^\ell : |u(t)| \preceq M(t),\ t \in [0,T]\}. \tag{44}$$
- *Input amplitude constraint:*
$$\Omega = \{u(t) \in \mathbb{R}^\ell : \lambda(t) \preceq u(t) \preceq \mu(t),\ t \in [0,T]\}.$$
- *Input sign constraint:*
$$\Omega = \{u(t) \in \mathbb{R}^\ell : 0 \preceq u(t),\ t \in [0,T]\}.$$
- *Input energy constraint:*
$$\Omega = \{u(t) \in \mathbb{R}^\ell : \int_0^T (u(t))^\top u(t)\mathrm{d}t \leqslant M,\ t \in [0,T]\}.$$

With constraints, the optimal problem (13) becomes

$$\begin{aligned} \underset{u,y,\Lambda}{\text{minimize}} \quad & f(u,y) \\ \text{subject to} \quad & r^p = G_\Lambda^p u, \\ & y = Gu, \\ & \Lambda \in \Theta,\ \ u \in \Omega. \end{aligned} \tag{45}$$

As will be seen later, the constraints add significant difficulties into the algorithm design. In this paper, only input constrains are considered. Note that in principle, the design developed in the following section can handle output constraints as well, but the details will be different and are omitted here for brevity.

B. A Modified Two Stage Design Framework with Input Constraints

Following a similar procedure to that in Section III, the constrained optimization problem (45) becomes

$$\min_{\Lambda\in\Theta} \left\{ \min_u f(u,y),\ \text{s.t. } G_\Lambda^p u = r^p,\ \ y = Gu,\ \ u \in \Omega \right\} \tag{46}$$

suggesting a possible two stage design framework:

- *Stage One:*
$$\begin{aligned} \underset{u\in\Omega}{\text{minimize}} \quad & \|u\|_R^2 \\ \text{subject to} \quad & r^p = G_\Lambda^p u \end{aligned} \tag{47}$$

whose the solution is denoted as $\hat{u}_\infty(\Lambda)$.

- *Stage Two:*

$$\min_{\Lambda \in \Theta} \{\tilde{f}(\Lambda) := \|\hat{u}_\infty(\Lambda)\|_R^2\}. \tag{48}$$

With the presence of the input constraints, the problem becomes significantly more difficult as the Stage One inner optimization problem needs to solve a constrained optimization problem, which unfortunately is inherently challenging and does not admit an analytical solution that is essential to the Stage Two optimization problem. To address this difficulty, a modified two stage design is proposed as follows.

Note that now Stage One does not have a direct analytical solution, but the norm-optimal ILC algorithm with successive projection proposed in [23] can be applied to solve the modified Stage One optimization problem (47). The update (21) is accordingly replaced by two alternative update methods.

- *Method 1:* Solve the constrained input norm-optimal point-to-point ILC optimization problem

$$u_{k+1} = \arg\min_{u \in \Omega} \{\|e^p\|_{[Q]}^2 + \|u - u_k\|_R^2\}. \tag{49}$$

This algorithm converges to the minimum error norm. The constrained QP problem (49) becomes computationally demanding which might introduce problems in some applications, when the trial length is large. A number of methods has been proposed to address this problem, see [23], [28] for more information.

- *Method 2:* Solve the unconstrained input norm-optimal point-to-point ILC optimization problem

$$\tilde{u}_{k+1} = \arg\min_{u} \{\|e^p\|_{[Q]}^2 + \|u - u_k\|_R^2\} \tag{50}$$

and then perform a simple input projection

$$u_{k+1} = \arg\min_{u \in \Omega} \|u - \tilde{u}_{k+1}\|. \tag{51}$$

It is clear that the first step has an analytical solution and the solution of the second step is straightforward as the input constraint Ω is usually a pointwise constraint in practice. This method is computationally simpler than Method 1 and can be carried out for large scale applications. Its convergence performance property, however, is different from that of Method 1.

The Stage Two optimization problem (48), i.e. Step 6 in Algorithm 3, is modified as

$$\Lambda_{j+1} = \arg\min_{\Lambda \in \Theta} \tilde{f}_j(\Lambda) \tag{52}$$

where

$$\tilde{f}_j(\Lambda) = \|u_\infty(\Lambda)\|_R^2 + \rho\,\|u_\infty(\Lambda) - \hat{u}_\infty(\Lambda_j)\|_R^2\,,\ \rho \geqslant 0, \tag{53}$$

and $\hat{u}_\infty(\Lambda_j)$ is the converged input in Stage One design for tracking time allocation Λ_j. Note that in this modified stage two design, the input constraint is decoupled from the optimization problem and thus can be solved analytically (using the algorithm in Theorem 3).

These new solution forms combine to generate Algorithm 4 for the optimal tracking time allocation problem in point to point ILC with system constraints.

Algorithm 4 Constrained input point-to-point ILC with optimal tracking time allocation

```
Input: initial tracking time allocation $\Lambda_0$, system state space model $S(A,B,C)$, desired tracking reference $r^p$, admissible set of tracking time allocation $\Theta$, input constraint set $\Omega$, weighting matrices $R$ and $Q_i$
Output: Optimal tracking time allocation $\Lambda_{opt}$ and corresponding input $u_{opt}$
initialization: Loop number $j = 0$
Repeatedly implement Stage One update (49) or (50)-(51) with $\Lambda = \Lambda_0$ experimentally using the update (34) until convergence, i.e. $\|e_k^p\| < \epsilon\|r^p\|$; record converged input $\hat{u}_\infty^{ex}(\Lambda_0)$, input energy $\tilde{f}(\Lambda_0)$.
repeat
    Compute the gradient using (24) or (38) with $r^p = G_{\Lambda_j}^p \hat{u}_\infty^{ex}(\Lambda_j)$; implement Stage Two update (52).
    Set $j \to j + 1$.
    Repeatedly implement Stage One update (49) or (50)-(51) with $\Lambda = \Lambda_j$ experimentally using the update (34) until convergence, i.e. $\|e_k^p\| < \epsilon\|r^p\|$; record converged input $\hat{u}_\infty^{ex}(\Lambda_j)$, input energy $\tilde{f}(\Lambda_j)$.
until $\left|\tilde{f}_j(\Lambda_j) - \tilde{f}(\Lambda_{j-1})\right| < \delta \left|\tilde{f}(\Lambda_{j-1})\right|$
return $\Lambda_{opt} = \Lambda_j$ and $u_{opt} = \hat{u}_\infty^{ex}(\Lambda_j)$
```

Remark 5. It is also possible to estimate the gradient $\nabla\tilde{f}(\Lambda)$ in (48) experimentally following similar procedures as those discussed in Section IV-B rather than compute it analytically using the above modified stage two design, the details of which are omitted here for brevity.

C. Convergence Properties of the Algorithm

Algorithm 4 has the following convergence properties:

Theorem 4. Suppose perfect tracking is achievable and $\rho \leqslant 1$, the analytical input energy resulting from (52) satisfies

$$\|u_\infty(\Lambda_{j+1})\|_R^2 \leqslant \|\hat{u}_\infty(\Lambda_j)\|_R^2\,. \tag{54}$$

Proof. See Appendix F for the proof of Theorem 4. □

Although Theorem 4 only states that the next loop's unconstrained minimum energy is no larger than the constrained minimum energy of the current loop, it still provides useful information about the convergence properties of proposed algorithm. We have undertaken a series of simulations using different models and input constraints to examine the convergence properties of Stage Two update (52). The results demonstrate that although not proved the proposed algorithm achieves monotonic convergence of the constrained minimum input energy, which is very appealing in practice. The simulation results are however omitted here for space reason.

VI. EXPERIMENTAL VERIFICATION ON A GANTRY ROBOT

The proposed design framework is now validated experimentally on a three-axis gantry robot test facility to demonstrate its effectiveness on a widely used industrial platform.

A. *Test Platform Specification*

The multi-axis gantry robot shown in Figure 2 comprises three perpendicular axes placed above a moving conveyor. The x-axis and the y-axis are designed to move in the horizontal plane and driven by linear brush-less dc motors. The vertical z-axis is placed above the other two axes, and has a linear ball-screw stage driven by a rotary brushless dc motor. The axis displacement data is measured using optical incremental encoders. The gantry robot uses an electromagnet to pick the payloads from a dispenser and place them onto the moving conveyor. Because the three axes are orthogonal and controlled separately, the gantry robot is considered to comprise three separate single-input single-output (SISO) systems.

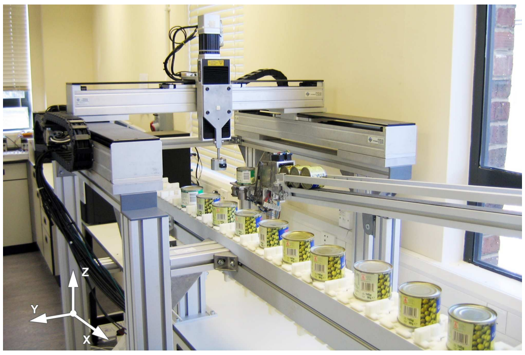

Fig. 2. Multi-axis Gantry Robot Test Platform.

The control design objective is to perform a pick-and-place task with two special tracking points ($M = 2$), which correspond to the 'pick' position and the 'place' position as shown in Figure 1 from an exemplary tracking trajectory. For simplicity, we only consider the z-axis in this paper with system model $G_z(s)$ and take the time index to two decimal place accuracy. The total trial length is $T = 1.99s$ and the reference for z-axis, r^p, is $[0.01, 0.008]^\top$. The parameter choice $t_1^+ = 0.99s$, $t_2^+ = 1.99s$, $t_1^- = 0.01s$, and $t_2^- = 1.01s$ ensures that the robot moves to the pick position first and then to the place position. To provide baseline tracking and disturbance rejection, a proportional controller with gain K, is applied around the system, yielding

$$G(s) = \frac{G_z(s)}{1 + KG_z(s)}. \tag{55}$$

The transfer function system model $G(s)$ can be equivalently written in state space form $S(A, B, C)$. Note that for this problem, previous studies used a predefined (*a priori*) tracking time allocation $\Lambda_r = [0.5, 1.35]^\top$, with a corresponding control input energy obtained by implementing the Stage One update only. The central initial tracking time allocation $\Lambda_0 = [0.5, 1.5]^\top$ is used in both Algorithm 3 and 4, the weighting matrices are taken as $Q_i = qI$ for $i = 1, \ldots, M$, $R = rI$ where q and r are positive scalars, and the gradient is obtained by the estimation (38) via analytical calculation. Furthermore, we choose the appropriate weighting matrices according to the theoretical predictions in [17] to balance convergence speed and robust performance, i.e. $q/r = 500,000$

B. *Experimental Results*

First assume that only an approximate system model of the z-axis is available as follows:

$$\hat{G}_z(s) = \frac{0.03}{s} \tag{56}$$

with a feedback gain $K = 30$. A 60 loop updating procedure of Algorithm 3 is performed using the gantry robot. In Step 4, the generalized Armijo step size (27) is applied with $\sigma = 0.1$, $\beta = 0.8$ and $\gamma = 0.03, 0.04, 0.05$.

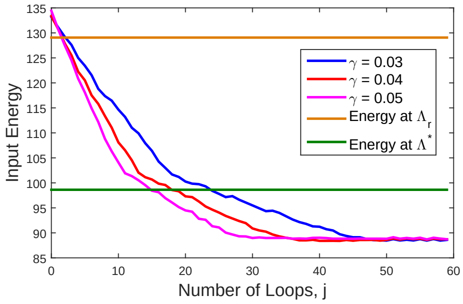

Fig. 3. Input Energy Results using an Inaccurate Model without Input Constraints.

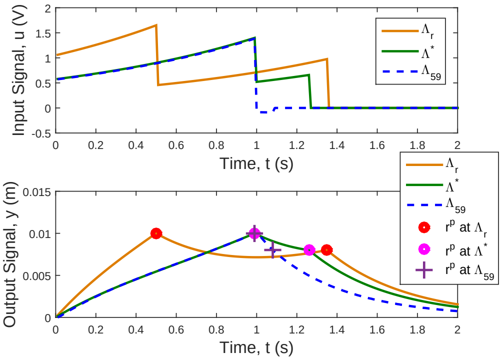

Fig. 4. Converged Input and Output Trajectory Comparison using an Inaccurate Model without Input Constraints.

The experimental optimal input energy $\tilde{f}(\Lambda_k)$ at each loop is plotted in Figure 3 for step size chosen under different values of γ. The optimal energy $\tilde{f}(\Lambda_r) = 129.06$ required for the gantry robot to track at the *a priori* tracking time allocation is also plotted for comparison. It should be noted that the proposed algorithm provides an experimental final converged energy of 88.76, which is a 31% reduction in input energy compared to the operating energy $\tilde{f}(\Lambda_r)$. This is further compared with normal practice by computing the optimal tracking time allocation in simulation using the nominal model, and then using Stage One update alone to track them experimentally. This yields $\Lambda^* = [0.99, 1.26]^\top$, and $\tilde{f}(\Lambda^*) = 98.62$. It is clear that the experimental final converged energy is approximately 10% less. This means that experimental implementation of Algorithm 4 is far superior to optimization using the nominal model in simulation. This confirms that the algorithm displays satisfactory robustness against model uncertainty. Furthermore, the converged input and output trajectories at the original (*a priori*), theoretical optimal and experimental optimal allocations are plotted in

Figure 4 and illustrate how the experimentally obtained optimal allocation outperforms the other two allocations.

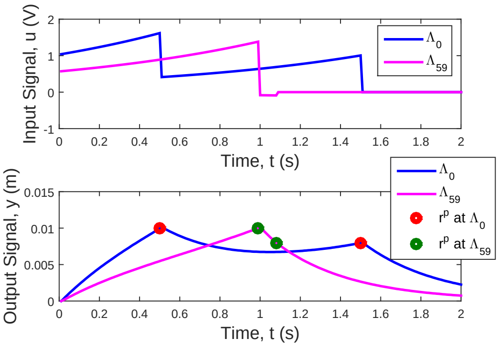

Fig. 5. Experimental Converged Input and Output Trajectories for Initial and Final Loops using an Inaccurate Model without Input Constraints.

The experimental final converged input and output trajectories for the initial and final loops of the algorithm are compared in Figure 5, and it is clear that the input signal immediately becomes zero after finishing tracking the last point in both figures as there is no tracking requirement along the remaining finite time interval. The reference at the special tracking time is marked with red and green circles. It is clear that the final converged output accurately tracks the special tracking points, so the algorithm not only optimizes the input energy but also maintains high tracking performance even with significant high model uncertainty. For example, while using the experimental estimation with step size chosen under $\gamma = 0.03$, the final mean square error is 0.00032mm^2 at Step 6 of the 60^{th} loop.

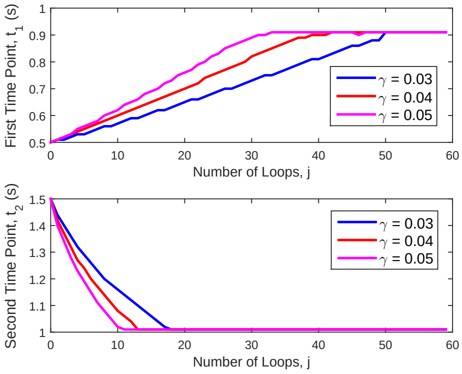

Fig. 6. Experimental Time-Point Position Results at Each Loop using an Inaccurate Model without Input Constraints.

To further illustrate the performance of Algorithm 3, the convergence of the tracking time allocation is shown in Figure 6. For each value of γ, the tracking time allocation at each loop are plotted in this figure, and all converge to identical tracking time allocation is $[0.91, 1.01]^\top$. The automatic tracking time allocation adjustment has meant that the speed of the gantry is slower and the distance the gantry moves is shorter (as can be seen from Figure 3), and thus leads to a lower input energy consumption. Experiments using the other initial tracking time allocations, e.g. the low resolution initial tracking time allocation, yield similar levels of performance.

TABLE I
SUMMARIZED EXPERIMENTAL WITHOUT CONSTRAINTS.

	$\gamma = 0.03$	$\gamma = 0.04$	$\gamma = 0.05$
Λ_{49}	$[0.99, 1.09]^\top$	$[0.99, 1.09]^\top$	$[0.99, 1.09]^\top$
$\tilde{f}(\Lambda_{49})$	78.69	78.67	78.66
Reduction from $\tilde{f}(\Lambda_r)$	37.10%	37.11%	37.12%
~~Reduction from~~ $\tilde{f}(\Lambda^*)$	0.63%	0.66%	0.67%
$\|e^p\|^2/2$ at Λ_{49}	0.00051mm^2	0.00006mm^2	0.00026mm^2

TABLE II
SUMMARIZED EXPERIMENTAL RESULTS WITH CONSTRAINTS.

	$\gamma = 0.05$	$\gamma = 0.06$	$\gamma = 0.07$
Λ_{24}	$[0.99, 1.08]^\top$	$[0.99, 1.08]^\top$	$[0.99, 1.08]^\top$
$\tilde{f}(\Lambda_{24})$	78.78	79.10	78.17
Reduction from $\tilde{f}(\Lambda_r)$	37.03%	36.77%	37.51%
~~Reduction from~~ $\tilde{f}(\Lambda^*)$	0.52%	0.11%	1.29%
$\|e^p\|^2/2$ at Λ_{24}	0.00013mm^2	0.00002mm^2	0.00003mm^2

The proposed algorithm with input saturation constraint (44) with $M(t) = 1.8$ has also been tested using different system models and parameter choices. A representative result on the input energy convergence is shown in Figure 7. From this figure, the optimal energy 89.53 obtained by the algorithm is 28% less than the energy $\tilde{f}(\Lambda_r) = 129.12$ at the *a priori* allocation and 10% less than the energy $\tilde{f}(\Lambda^*) = 98.51$ at the theoretically obtained optimal allocation using the inaccurate model, which again confirms the superiority of the proposed design method. Furthermore, inspection on the converged input shows that input constraint satisfaction is guaranteed by the proposed algorithm, a typical result of which is show in Figure 8 that clearly demonstrates this.

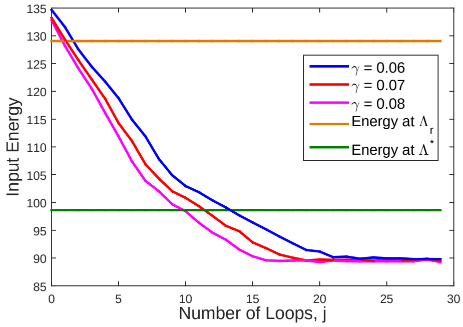

Fig. 7. Exemplary Input Energy Convergence with Input Constraints.

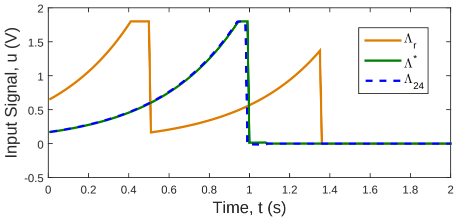

Fig. 8. Exemplary Converged Input Trajectories with Input Constraints.

The above tests have also been repeated using a relatively accurate system model obtained in [29] using system identification techniques, i.e.

$$G_z(s) = \frac{15.8869(s + 850.3)}{s(s^2 + 707.6s + 3.377 \times 10^5)} \tag{57}$$

with feedback gain $K = 100$. The results are summarized in Table I and Table II for the unconstrained and constrained cases. From the tables, the tracking time allocations obtained by the proposed algorithms all converge, and are close to the corresponding theoretical obtained one Λ^* as the system model is relatively accurate. It is clear that obtained input energy results at each case are approximately 37% less than the *a priori* one $\tilde{f}(\Lambda_r)$, and are almost the same as the theoretical one $\tilde{f}(\Lambda^*)$. Also, the tracking errors are within the practical tolerance, confirming that the algorithm not only optimizes the input energy, but also maintains satisfactory tracking performance. Due to space reason, the detailed results are omitted here for brevity.

VII. Conclusion

Tracking time allocation plays an important role in point-to-point ILC and can significantly affect the system performance. This paper develops an optimization framework to fully exploit the flexibility in choosing tracking time allocation to optimize some performance index of interest, in addition to high accuracy reference tracking. The problem is formulated into an optimization problem in some abstract Hilbert space and a two stage design framework is developed. Solution to the Stage One design problem is derived using a well-known norm-optimal point to point ILC algorithm. For the Stage Two design problem there are no direct analytical solutions of the optimization problem, and an iterative algorithm based on the gradient projection method is proposed. The implementation procedures are discussed in detail and the proposed design framework is further extended to embed system constraints into the design.

The proposed algorithm is verified experimentally on a gantry robot test platform. When the system model is inaccurate, significant reduction of the input energy can be achieved. When an accurate system model is available, the input energy converges to the theoretical optimal solution. In both scenarios, the proposed algorithm guarantees high performance tracking.

Although the experimental implementation of the algorithm has demonstrated that the proposed design has certain degree of robustness against model uncertainties, a rigorous analysis of the algorithm's robustness properties will be undertaken. In addition, other methods, e.g. the projected Newton method [30], can also be used to solve the Stage Two optimization problem, the convergence properties of which will be different from those described in this paper. Furthermore, in this paper the performance index is considered to be a convex function. In principle, the proposed design framework can be extended to non-convex cost functions as well. The above forms part of our future research and will be reported separately.

Appendix

A. Proof of Theorem 1

The proof follows the ideas in [17] and is described in detail below.

On the $(k+1)^{th}$ trial, the norm-optimal point-to-point ILC algorithm solves the optimization problem

$$\min_u \{\|e^p\|^2_{[Q]} + \|u - u_k\|^2_R : \; e^p = r^p - y^p, \; y^p = G^p_\Lambda u\} \quad (58)$$

to get the control input u_{k+1}. The problem (58) has an identical structure to the norm-optimal ILC problem described in [31], with the only difference being the definitions of the operators, signals and underlying Hilbert spaces. Therefore, the iterative solution can be expressed as

$$u_{k+1} = u_k + G^{p*}_\Lambda e^p_{k+1} \Rightarrow e^p_{k+1} = (I + G^p_\Lambda G^{p*}_\Lambda)^{-1} e^p_k \quad (59)$$

which gives rise to (21).

It is proved in [17] that if a system is controllable and C has full row rank, the reference r^p can be tracked exactly and the limit of the sequence $\{u_k\}$ exists, i.e.

$$\lim_{k\to\infty} e^p_k = 0, \quad \lim_{k\to\infty} u_k = u_\infty.$$

The algorithm converges to the minimum control energy that achieves perfect tracking requirement if $u_0 = 0$. Hence the Stage One optimization problem (18) can be solved by the norm-optimal point-to-point ILC algorithm.

The relevant adjoint operator G^{p*}_Λ is obtained in [17] from the definition

$$\langle(\omega_1, ..., \omega_M), \; G^p_\Lambda u\rangle_{[Q]} = \langle G^{p*}_\Lambda(\omega_1, ..., \omega_M), \; u\rangle_R \quad (60)$$

which gives rise to

$$(G^*_i \omega_i)(t) = \begin{cases} R^{-1}B^\top e^{A^\top(t_i - t)} C^\top Q_i \omega_i, & 0 \leqslant t \leqslant t_i, \\ 0, \; t > t_i. \end{cases} \quad (61)$$

The equation (61) can be further written as

$$(G^*_i \omega_i)(t) = R^{-1}B^\top p_i(t) \quad (62)$$

where $p_i(t) = 0$ on $(t_i, \; T]$, and on $[0, \; t_i)$

$$\dot{p}_i(t) = -A^\top p_i(t), \; p_i(t_i-) = C^\top Q_i \omega_i. \quad (63)$$

Adjoint operator G^{p*}_Λ is the map $(\omega_1, ..., \omega_M) \mapsto u$ defined by

$$u(t) = \sum_{i=1}^M (G^*_i \omega_i)(t) = R^{-1}B^\top p(t) \quad (64)$$

where $p(t) = \sum_{i=1}^M p_i(t)$. Due to the linearity, these equations yield the costate equation modified by 'jump conditions' at time t_i, which generates the definition (22) of the adjoint operator G^{p*}_Λ. Therefore, the $(i,j)^{th}$ block of the matrix $G^p_\Lambda G^{p*}_\Lambda$ can be computed as the equation (23).

In order to solve the optimization problem (18) analytically, the associated Lagrangian expression is given

$$\mathcal{L}(u) = \|u\|^2_R + 2\langle\lambda, \; G^p_\Lambda u - r^p\rangle_{[Q]}. \quad (65)$$

Follow the method introduced in [32], let u_∞ be the global optimal u that minimize $\mathcal{L}$ and there exists

$$\mathcal{L}(u_\infty) \leqslant \mathcal{L}(u_\infty + \tau), \; \forall \; \tau \in L^\ell_2[0, \; T] \quad (66)$$

which can be equivalently written as

$$\begin{aligned} \mathcal{L}(u_\infty + \tau) - \mathcal{L}(u_\infty) &= \|\tau\|^2_R + 2\langle u_\infty, \; \tau\rangle_R + 2\langle\lambda, G^p_\Lambda \tau\rangle_{[Q]} \\ &= \|\tau\|^2_R + 2\left\langle u_\infty + G^{p*}_\Lambda \lambda, \; \tau\right\rangle_R \geqslant 0, \forall \; \tau \in L^\ell_2[0, \; T]. \quad (67) \end{aligned}$$

Substitute $\tau = -(u_\infty + G^{p*}_\Lambda \lambda)$ into (67) to obtain

$$-\left\|u_\infty + G^{p*}_\Lambda \lambda\right\|^2_R \geqslant 0. \quad (68)$$

It follows that the inequality condition in (68) only holds when $u_\infty = -G_\Lambda^{p*}\lambda$ which satisfy the condition (66) for all $\tau \in L_2^\ell[0,\ T]$. Then recall the tracking requirement to build

$$\begin{cases} G_\Lambda^p u_\infty = r^p \\ u_\infty = -G_\Lambda^{p*}\lambda \end{cases} \tag{69}$$

which yields $G_\Lambda^p G_\Lambda^{p*}\lambda = -r^p$. As the matrix $G_\Lambda^p G_\Lambda^{p*}$ is invertible, it follows that $\lambda = -(G_\Lambda^p G_\Lambda^{p*})^{-1} r^p$, which together with $u_\infty = -G_\Lambda^{p*}\lambda$ give rise to the analytical solution (24).

B. Proof of Lemma 1

Substituting the analytical solution (24) into the problem (19) and using the property of adjoint operator gives

$$\begin{aligned} \min_{\Lambda\in\Theta} \|u_\infty(\Lambda)\|_R^2 &= \min_{\Lambda\in\Theta}\ \langle(u_\infty(\Lambda),\ u_\infty(\Lambda)\rangle_R \\ &= \min_{\Lambda\in\Theta}\ \left\langle(G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1}r^p, G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_R \\ &= \min_{\Lambda\in\Theta}\ \left\langle G_\Lambda^p G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1}r^p, (G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_{[Q]} \\ &= \min_{\Lambda\in\Theta}\ \left\langle r^p,\ (G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_{[Q]} \end{aligned}$$

which completes the proof.

C. Proof of Theorem 2

For $M = 1$, there is only one time-point and thus $\Lambda = t_1 \in \mathbb{R}$. Denote $\Psi_{t_1} = G_{t_1}^p G_{t_1}^{p*}$ which can be explicitly written as

$$\Psi_{t_1} = \int_0^{t_1} Ce^{A(t_1-t)}BR^{-1}(Ce^{A(t_1-t)}B)^\top \mathrm{d}t. \tag{70}$$

The Stage Two optimization problem becomes

$$\min_{\Lambda\in\Theta} \|u_\infty(\Lambda)\|_R^2 = \min_{\Lambda\in\Theta}\ \left\langle r^p,\ \Psi_{t_1}^{-1} r^p\right\rangle_{[Q]}. \tag{71}$$

Note that Ψ_{t_1} is a positive operator, and furthermore,

$$\Psi_{t_1} \leqslant \Psi_T,\ \forall\ t_1^- \leqslant t_1 \leqslant t_1^+ = T,$$

as for any x, it can be shown

$$\begin{aligned} &\langle x,\ (\Psi_T - \Psi_{t_1})x\rangle_{[Q]} = \\ &\qquad \langle x,\ \int_{t_1}^T Ce^{A(t_1-t)}BR^{-1}(Ce^{A(t_1-t)}B)^\top \mathrm{d}tx\rangle_{[Q]} \geqslant 0. \end{aligned}$$

The above properties of positive operators yield $\Psi_{t_1}^{-1} \geqslant \Psi_T^{-1}$, and therefore $\left\langle r^p,\ \Psi_{t_1}^{-1} r^p\right\rangle_{[Q]} \geqslant \left\langle r^p,\ \Psi_T^{-1} r^p\right\rangle_{[Q]}$. It follows that $t_1 = t_1^+ = T$ is an optimum of the Stage Two optimization problem, which completes the proof.

D. Proof of Theorem 3

To prove Theorem 3, the following lemma is needed.

Lemma 2. [33] Let $\{\Lambda_k\}$ be a sequence generated by $\Lambda_{j+1} = P_\Theta(\Lambda_j - \gamma_j \cdot \nabla f(\Lambda_j))$ where

$$\Theta = \{\Lambda \in \mathbb{R}^M : \lambda_i \leqslant t_i \leqslant \mu_i,\ i = 1, \ldots, M\}, \tag{72}$$

and γ_j is chosen according to the generalized Armijo step size (27). Then every limit point of the sequence $\{\Lambda_k\}$ is a stationary point for problem (25).

In this paper, the admissible set Θ satisfies the constraint set requirement (72), and the gradient projection method (26) with generalized Armijo step size (27) is used in Theorem 3. Using this theorem, all the assumptions in Lemma 2 are satisfied, and hence the sequence $\{\Lambda_k\}$ converges to a stationary point of the problem (25).

E. Proof of Proposition 1

The ILC update (21) is equivalent to

$$u_{k+1}(t) = u_k(t) + (G_\Lambda^{p*} e_{k+1}^p)(t), \tag{73}$$

and $(G_\Lambda^{p*} e_{k+1}^p)(t)$ can be written as

$$\begin{aligned} &(G_\Lambda^{p*} e_{k+1}^p)(t) = R^{-1}B^\top p_k(t),\ \dot{p}_k(t) = -A^\top p_k(t), \\ &p_k(T) = 0,\ p_k(t_i-) = p_k(t_i+) + C^\top Q_i e_{k+1}(t_i), \end{aligned} \tag{74}$$

according to the costate equation (22). Hence (73) becomes

$$u_{k+1}(t) = u_k(t) + R^{-1}B^\top p_k(t). \tag{75}$$

Then substitute the equation

$$p_k(t) = -K(t)(x_{k+1}(t) - x_k(t)) + \xi_{k+1}(t) \tag{76}$$

into the jump condition at t_i of costate equation (74) to give

$$\begin{aligned} &-(K(t_i+) - K(t_i-))(x_{k+1}(t_i) - x_k(t_i)) \\ &\qquad + (\xi_{k+1}(t_i+) - \xi_{k+1}(t_i-)) = C^\top Q_i e_{k+1}(t_i) \end{aligned} \tag{77}$$

and the error $e_{k+1}(t_i)$ can be further equivalently written as

$$\begin{aligned} e_{k+1}(t_i) &= r_i - y_{k+1}(t_i) \\ &= e_k(t_i) - C(x_{k+1}(t_i) - x_k(t_i)). \end{aligned} \tag{78}$$

Hence (77) and (78) suggest the jump conditions at t_i in (36) and (37). Then use the method proposed in [32] to differentiate (35) at any point t not in Λ and substitute for $\dot{x}_k$ and $\dot{x}_{k+1}$. These provide the Riccati and predictive differential equations in (36) and (37).

F. Proof of Theorem 4

As Λ_{j+1} is the solution of the gradient projection, it is clear from Theorem 3 that the inequality $\tilde{f}_j(\Lambda_{j+1}) \leqslant \tilde{f}_j(\Lambda_j)$ holds and hence it follows that

$$\begin{aligned} &\|u_\infty(\Lambda_{j+1})\|_R^2 \leqslant \rho\,\|u_\infty(\Lambda_{j+1}) - \hat{u}_\infty(\Lambda_j)\|_R^2 + \|u_\infty(\Lambda_{j+1})\|_R^2 \\ &\leqslant \|u_\infty(\Lambda_j)\|_R^2 + \rho\,\|u_\infty(\Lambda_j) - \hat{u}_\infty(\Lambda_j)\|_R^2 = \rho\,\|\hat{u}_\infty(\Lambda_j)\|_R^2 \\ &+ (1+\rho)\,\|u_\infty(\Lambda_j)\|_R^2 - 2\rho\,\langle\hat{u}_\infty(\Lambda_j),\ u_\infty(\Lambda_j)\rangle_R\,. \end{aligned} \tag{79}$$

Then, recall the analytical solution (24) for $u_\infty(\Lambda)$ and the perfect tracking assumption $G_\Lambda^p \hat{u}_\infty(\Lambda) = r^p$ to give

$$\begin{aligned} &\langle\hat{u}_\infty(\Lambda),\ u_\infty(\Lambda)\rangle_R = \left\langle\hat{u}_\infty(\Lambda),\ G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_R \\ &= \left\langle G_\Lambda^p \hat{u}_\infty(\Lambda), (G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_{[Q]} = \left\langle r^p, (G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_{[Q]} \\ &= \left\langle G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1}r^p,\ G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1}r^p\right\rangle_R \\ &= \langle u_\infty(\Lambda),\ u_\infty(\Lambda)\rangle_R\,. \end{aligned} \tag{80}$$

Substitute (80) into (79) to give

$$\|u_\infty(\Lambda_{j+1})\|_R^2 \leqslant (1-\rho)\,\|u_\infty(\Lambda_j)\|_R^2 + \rho\,\|\hat{u}_\infty(\Lambda_j)\|_R^2\,. \tag{81}$$

It is clear that the unconstrained converged input energy is no larger than the constrained converged input energy i.e.

$$\|u_\infty(\Lambda_j)\|_R^2 \leqslant \|\hat{u}_\infty(\Lambda_j)\|_R^2,$$

and it follows that

$$(1-\rho)\,\|u_\infty(\Lambda_j)\|_R^2 \leqslant (1-\rho)\,\|\hat{u}_\infty(\Lambda_j)\|_R^2 \qquad (82)$$

as $(1-\rho)$ is non-negative. Hence combine (82) and (81) together to generate the inequality (54).

Acknowledgment

This research project was partially funded by the China Scholarship Council (CSC), China NSFC (No. 61473265), Science and Technology Innovation Team Project of Henan Province China (No. 17IRTSTHN013), and ZZU-Southampton collaborative research project (Ref:16306/01).

References

[1] 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, pp. 339–348, 2010.

[2] M. Norrlof, "An adaptive iterative learning control algorithm with experiments on an industrial robot," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 2, pp. 245–251, 2002.

[3] X. Yu, Z. Xiong, D. Huang, and Y. Jiang, "Model-based iterative learning control for batch processes using generalized hinging hyperplanes," *Industrial & Engineering Chemistry Research*, vol. 52, no. 4, pp. 1627–1634, 2013.

[4] J. H. Lee and K. S. Lee, "Iterative learning control applied to batch processes: An overview," *Control Engineering Practice*, vol. 15, pp. 1306–1318, 2007.

[5] C. T. Freeman, *Control System Design for Electrical Stimulation in Upper Limb Rehabilitation*. Springer International Publishing, 2016.

[6] D. Bristow, M. Tharayil, and A. 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.

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

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

[9] C. T. Freeman, Z. Cai, E. Rogers, and P. L. Lewin, "Iterative learning control for multiple point-to-point tracking application," *Ieee Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 590–600, 2011.

[10] G. Gauthier and B. Boulet, "Robust design of terminal ILC with mixed sensitivity approach for a thermoforming oven," *Journal of Control Science and Engineering*, vol. 2008, 2008, article ID 289391.

[11] Y.Wang and Z. Hou, "Terminal iterative learning control based station stop control of a train," *International Journal of Control*, vol. 84, no. 7, pp. 1263–1277, 2011.

[12] J.-X. Xu, Y. Chen, T. Lee, and S. Yamamoto, "Terminal iterative learning control with an application to RTPCVD thickness control," *Automatica*, vol. 35, no. 9, pp. 1535–1542, 1999.

[13] J.-X. Xu and D. Huang, "Initial state iterative learning for final state control in motion systems," *Automatica*, vol. 44, no. 12, pp. 3162–3169, 2008.

[14] H. Ding and J. Wu, "Point-to-point control for a high-acceleration positioning table via cascaded learning schemes," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 5, pp. 2735–2744, 2007.

[15] J. Park, P. H. Chang, H. S. Park, and E. Lee, "Design of learning input shaping technique for residual vibration suppression in an industrial robot," *IEEE/ASME Transactions on Mechatronics*, vol. 11, no. 1, pp. 55–65, 2006.

[16] J. van de Wijdeven and O. Bosgra, "Residual vibration suppression using hankel iterative learning control," *International Journal of Robust Nonlinear Control*, vol. 18, no. 10, pp. 1034–1051, 2008.

[17] D. H. Owens, C. T. Freeman, and T. V. 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, 2013.

[18] 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, 2013.

[19] D. Shen and Y. Wang, "Survey on stochastic iterative learning control," *Journal of Process Control*, vol. 24, no. 12, pp. 64–77, 2014.

[20] T. D. Son, H. S. Ahn, and K. L. Moore, "Iterative learning control in optimal tracking problems with specified data points," *Automatica*, vol. 49, no. 5, pp. 1465–1472, 2013.

[21] C. T. Freeman, "Constrained point-to-point iterative learning control with experimental verification," *Control Engineering Practice*, vol. 20, no. 5, pp. 489–498, 2012.

[22] C. T. Freeman and Y. Tan, "Iterative learning control with mixed constraints for point-to-point tracking," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 604–616, 2013.

[23] 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.

[24] P. Janssens, G. Pipeleers, M. Diehl, and J. Swevers, "Energy optimal time allocation of a series of point-to-point motions," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 6, pp. 2432–2435, 2014.

[25] Y. Chen, B. Chu, and C. T. Freeman, "Point-to-point iterative learning control with optimal tracking time allocation," in *54th IEEE Conference on Decision and Control*, Osaka, Japan, 2015, pp. 6089–6094.

[26] A. A. Goldstein, *Constructive Real Analysis*. Dover Publications, 2012.

[27] E. G. Birgin, J. M. Martinez, and M. Raydan, "Nonmonotone spectral projected gradient methods on convex sets," *SIAM Journal on Optimization*, vol. 10, no. 4, pp. 1196–1211, 2000.

[28] 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.

[29] L. Hladowski, K. Galkowski, Z. Cai, E. Rogers, C. T. Freeman, and P. L. Lewin, "A 2D systems approach to iterative learning control for discrete linear processes with zero markov parameters," *International Journal of Control*, vol. 84, no. 7, pp. 1246–1262, 2011.

[30] D. P. Bertsekas, "Projected newton methods for optimization problems with simple constraints," *SIAM Journal on Control and Optimization*, vol. 20, no. 2, pp. 221–246, 1982.

[31] N. Amann, D. Owens, and E. Rogers, "Iterative learning control using optimal feedback and feedforward actions," *International Journal of Control*, vol. 65, no. 2, pp. 277–293, 1996.

[32] N. Amann, "Optinal algorithms for iterative learning control," Ph.D. dissertation, University of Exeter, Exeter, 1996.

[33] D. P. Bertsekas, "On the Goldstein - Levitin - Polyak gradient projection method," *IEEE Transactions on Automatic Control*, vol. 21, no. 2, pp. 174–184, 1976.