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 [1]. Unlike modifying the controller as in adaptive control, ILC directly updates the input based on information from previous experimental attempts (named trials) to improve tracking performance [2]. Each trial has the same finite trial length, and the system states are reset to the same value at the start of each trial. The tracking error over each trial 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 [3], [4], chemical batch processing [5], [6] and stroke rehabilitation [7]. See [1] 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 [9]. 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 [10]. This novel control concept is termed point-to-point ILC, and has attracted significant interest.

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.

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 [11]–[14]. General point-to-point ILC design by employing a 'complete' reference that passes through all the desired intermediate points is studied in [15]–[17]. These methods, however, do not fully use 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 [18]–[21] 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 [9], [10].

All the aforementioned point-to-point ILC problem formulations 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 benefits can be realized, such as reducing the energy use in industry, 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 [22], 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.

This paper develops a comprehensive optimal tracking time allocation framework in point-to-point ILC to allow automatic choosing 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 [23]. 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 ex-

ploited to optimize some performance index of interest, e.g. energy consumption, at the same time, 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 [18] ensuring accurate tracking, and a gradient based minimization to update the tracking time allocation to optimize the performance index. It can be shown that 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 into the design. 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: $\mathbb{N}^n$ is the set of n dimensional vectors containing 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; $\langle\cdot,\ \cdot\rangle$ is the inner product; R, S and Q are the product spaces; P_Θ is the projection operator to the set Θ. Other notation will be introduced as needed.

II. FORMULATION OF THE PROBLEM

In this section, the design problem of point-to-point ILC with optimal tracking time allocation is illustrated firstly using a robotic 'pick and place' example and then 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 and differential delay systems, etc.

A. Point-to-Point ILC

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

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

where t is the time index; $k\in\mathbb{N}$ 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. $x_k(0)=x_0,\ \forall k$.

The design objective is to find an input such that the system output follows a given reference defined on the trial interval $[0,T]$ as accurately as possible. Note that the system operates in a repetitive manner, i.e. at the end of each trial ($t=T$), the system state is reset to the same initial condition x_0, another trial begins and the system is required to track the same reference again. This class of systems have many applications in robotics [24], manufacturing [25], etc.

In the point-to-point ILC framework, only the output values $y(t)$ at a finite number of pre-specified time instants are of interest. As an example, the output trajectories for a robotic pick-and-place tasks are 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, we are only interested in the tracking positions at the 'pick' and 'place' time instants, i.e. t_1 and t_2 - the movements beyond them are not of interest. Therefore, although the two output trajectories in the figure differ significantly, both satisfy the control design requirement. These two trajectories will clearly lead to different system performance in terms of energy use, smoothness of the movements, etc. The problem of finding a suitable trajectory with satisfactory performance has been considered in the literature - please refer to [26], [27] for more details.

Another look at the problem reveals that 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 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. 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. This paper aims to address this problem by proposing a design framework to automatically choose the tracking time allocation to optimize some performance of interest, and at the same time to ensure high performance tracking at these chosen time instants.

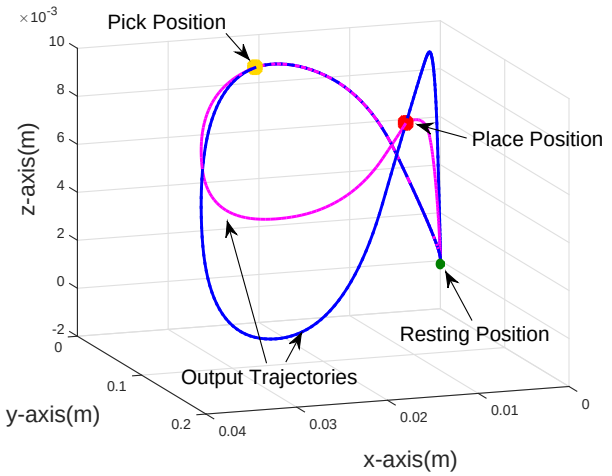

Fig. 1. A Robotic 'Pick-and-Place' Tracking Example.

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

To formulate the optimal tracking time allocation problem, denote the M tracking time instants (of interest), $t_i,\ i = 1,\ldots,M$ in vector form as

$$\Lambda = [t_1,\ t_2,\ldots,\ t_M]^\top \in \Theta \tag{2}$$

where Θ is the admissible set of tracking time allocation

$$\Theta = \{\Lambda \in \mathbb{R}^M : 0 < t_1^- \leqslant t_1 \leqslant t_1^+ \leqslant t_2^- \leqslant t_2 \leqslant t_2^+ \leqslant \ldots \leqslant t_M^+ = T\} \tag{3}$$

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. Also denote the desired tracking reference at these time instants as

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

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

$$y_k = Gu_k + d. \tag{4}$$

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

$$\langle u,\ v\rangle_R = \int_0^T (u(t))^\top Rv(t)\mathrm{d}t,\ \ \|u\|_R^2 = \langle u,\ u\rangle_R \tag{5}$$

$$\langle x,\ y\rangle_S = \int_0^T (x(t))^\top Sy(t)\mathrm{d}t,\ \ \|y\|_S^2 = \langle y,\ y\rangle_S \tag{6}$$

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

$$(Gu_k)(t) = \int_0^t Ce^{A(t-s)}Bu_k(s)\mathrm{d}s,\ \ d(t) = Ce^{At}x_0. \tag{7}$$

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

In point-to-point tracking tasks, we are only interested in the system output at a particular tracking time allocation Λ. To extract the system output value at Λ, introduce the linear mapping $f \in L_2^m[0,T] \mapsto f^p \in H$ defined as

$$f^p = \begin{bmatrix} f(t_1) \\ f(t_2) \\ \vdots \\ f(t_M) \end{bmatrix} \tag{8}$$

where H is a Hilbert space

$$H = \underbrace{\mathbb{R}^m \times \cdots \times \mathbb{R}^m}_{\text{M times}} \tag{9}$$

with respective inner product and induced norm

$$\langle \hat{x},\ \hat{y}\rangle_Q = \sum_{i=1}^M \hat{x}_i^\top Q\hat{y}_i,\ \ \|\hat{y}\|_Q^2 = \langle \hat{y},\ \hat{y}\rangle_Q \tag{10}$$

in which $Q \in \mathbb{S}_{++}^m$ 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 a linear operator, this can be further written as

$$y^p = G_\Lambda^p u = \begin{bmatrix} G_1 u \\ G_2 u \\ \vdots \\ G_M u \end{bmatrix} \tag{11}$$

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

$$G_i u = \int_0^{t_i} Ce^{A(t_i-t)}Bu(t)\mathrm{d}t. \tag{12}$$

The tracking error at tracking time allocation Λ is therefore

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

We are now ready to formulate the problem of point-to-point ILC with optimal tracking time allocation:

The **Point-to-Point ILC with Optimal Tracking Time Allocation Problem** can now be defined as iteratively finding a tracking time allocation Λ_k and an input u_k such that the output values at these intermediate time instants, i.e. y_k^p, accurately pass through a set of desired points r^p, i.e. $\lim_{k\to\infty} y_k^p = r^p$, at the same time minimizing a target performance index $f(u,\ y)$ as a function of the system input u and output y with an asymptotic property that is $(u_k,\ y_k,\ \Lambda_k) \to (u_k^*,\ y_k^*,\ \Lambda_k^*)$ where u_k^*, y_k^* and Λ_k^* are optimal solutions of the following problem

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

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.

Remark 2. 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 $G_\Lambda^p u = r^p$ in a nonlinear manner. Furthermore, the input u lies in an infinite dimensional space $L_2^\ell[0,T]$ 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

$$\min_{\Lambda\in\Theta}\Big\{\min_u f(u,\ y),\ \text{subject to } G_\Lambda^p u = r^p,\ y = Gu\Big\} \tag{14}$$

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

$$\tilde{f}(\Lambda) = \min_u \big\{ f(u,\ y),\ \text{subject to } G_\Lambda^p u = r^p,\ y = Gu\big\},$$

and denote a global minimizer for u of the inner optimization problem as $u_\infty(\Lambda) : \Theta \to L_2^\ell[0,T]$, the optimization problem (14) is then equivalent to

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

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 optimal input selection problem

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

- *Stage Two:* Substitute the fixed time optimal solution $u_\infty(\Lambda)$ into the original optimization problem (14) and compute the optimal tracking time allocation

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

To exemplify the approach in this paper, the input energy consumption is selected to be the target performance index, so that $f(u,\ y) = \|u\|_R^2$. 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

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

and

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

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 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 uncertainty. 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

$$u_{k+1} = \arg\min_u\{\|e^p\|_Q^2 + \|u - u_k\|_R^2\} \tag{20}$$

proposed in [18] with initial input $u_0 = 0$, such that

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

The iterative solution is given by

$$u_{k+1} = u_k + G_\Lambda^{p*}(I + G_\Lambda^p G_\Lambda^{p*})^{-1} e_k^p \tag{21}$$

where $G_\Lambda^{p*} : (\omega_1,...,\omega_M) \in H \to u \in L_2^\ell[0,T]$ is the Hilbert adjoint operator of G_Λ^p defined by

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

and the $Mm \times Mm$ matrix $G_\Lambda^p G_\Lambda^{p*}$ has a block structure with $(i,j)^{th}$ block

$$G_i G_j^* = \int_0^{min(t_i,t_j)} Ce^{A(t_i-t)} BR^{-1} B^\top e^{A^\top (t_i-t)} C^\top Q \; dt. \quad (23)$$

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

$$u_\infty(\Lambda) = G_\Lambda^{p*} (G_\Lambda^p G_\Lambda^{p*})^{-1} r^p. \quad (24)$$

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

$$\min_{\Lambda\in\Theta} \|u_\infty(\Lambda)\|_R^2 = \min_{\Lambda\in\Theta} \; \left\langle r^p, \; (G_\Lambda^p G_\Lambda^{p*})^{-1} r^p \right\rangle_Q. \quad (25)$$

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

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

where

$$\Psi_T = \int_0^T Ce^{A(t_1-t)} BR^{-1} (Ce^{A(t_1-t)} B)^\top \; dt.$$

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

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

where $j \in \mathbb{N}$ denotes the updating 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 [28], i.e.

$$\gamma_j = \beta^{m_k} \cdot \gamma \quad (27)$$

where m_k is the smallest non-negative integer such that

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

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

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

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

$$\lim_{j\to\infty} \left\|\Lambda_j - \Lambda_{j+1}\right\| = 0 \quad (30)$$

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

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

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

Remark 3. It is worth pointing out that other step size choices are also possible, e.g. constant step size [29]

$$0 < \mu \leqslant \gamma_j \leqslant \frac{2(1-\mu)}{L}, \quad \forall j \quad (31)$$

where $L > 0$ is the Lipschitz constant of $\tilde{f}(\Lambda)$ on Θ and $\mu \in (0, \; 2/(2+L)]$ is a positive scalar, projected Barzilai-Borwein step size [30]

$$\gamma_j = \frac{\langle \Delta x_j, \; \Delta g_j \rangle}{\langle \Delta g_j, \; \Delta g_j \rangle}, \text{ or } \gamma_j = \frac{\langle \Delta x_j, \; \Delta x_j \rangle}{\langle \Delta x_j, \; \Delta g_j \rangle} \quad (32)$$

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.

C. A Numerical Example

In this subsection, a numerical example is presented to illustrate the results in Theorem 2 for one tracking point, and design difficulties for more than one tracking point.

Example 1. Consider the following system model

$$G_z(s) = \frac{15.8869(s + 850.3)}{s(s^2 + 707.6s + 3.377 \times 10^5)} \quad (33)$$

which is used in [31] with a proportional feedback gain of 100 to accurately model the gantry robot system employed in Section VI. Firstly, we suppose $M = 1$, i.e. there is only one tracking point, the trial length is $T = 2s$, and the tracking

reference is $r^p = 0.01$. The admissible set of tracking time allocation Θ in (3) is defined by the parameters $t_1^+ = 2$ and $t_1^- = 0.01$. The input energy at a particular time allocation t_1 can be computed analytically using equations (69) and (70), and the result is plotted in Figure 2. It is clear from this figure that the minimum input energy is achieved at $t_1 = T$ verifying the theoretical prediction in Theorem 2.

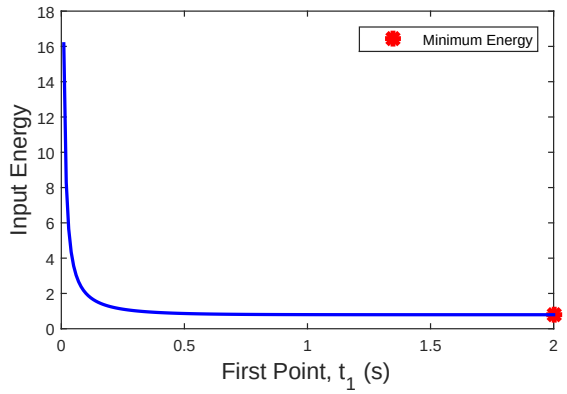

Fig. 2. Input Energy $\tilde{f}(\Lambda)$ at Single Point Case ($M = 1$) using Example 1.

Now suppose $M = 2$, the trial length stays the same as $T = 2s$, and the tracking reference is given by $r^p = [0.01,\ 0.008]^\top$. Furthermore, the admissible set of tracking time allocation Θ in (3) is defined by parameters $t_1^+ = 1$, $t_2^+ = 2$, $t_1^- = 0.01$, and $t_2^- = 1.01$. The required input energy to achieve the design tracking task as a function of tracking time instants t_1 and t_2 is plotted in Figure 3. It is clear from this figure that $\tilde{f}(\Lambda)$ is non-linear and non-convex with respect to Λ indicating the difficulty in solving the Stage Two problem. In this case the algorithm in Theorem 3 should be applied to solve the Stage Two optimization problem (25), the results of which will be verified experimentally later in section VI.

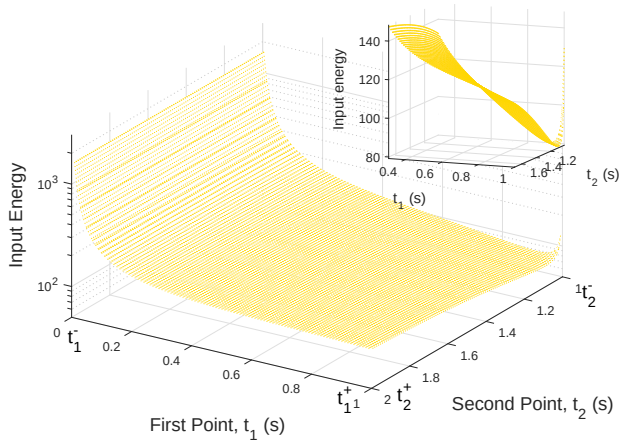

Fig. 3. Input Energy $\tilde{f}(\Lambda)$ at Multiple Point Case ($M = 2$) using Example 1.

IV. IMPLEMENTATION OF THE DESIGN APPROACH

In the 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 $K(t)$ denotes the Riccati feedback matrix

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

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

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

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 [18]), 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). 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 $\Lambda_j^{i+} = [t_1^j, t_2^j, ..., t_i^j + \Delta T, ..., t_M^j]^\top$ and $\Lambda_j^{i-} = [t_1^j, t_2^j, ..., t_i^j - \Delta T, ..., t_M^j]^\top$, 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 given by

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

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

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

Algorithm 1 Greedy initial tracking time allocation

Input: $S(A,B,C)$, r^p, Λ_0^c and Θ

Output: Greedy initial tracking time allocation Λ_0^g

1: **for** $i = 1$ to M **do**
2: Let $\Lambda_i = [t_1^*,\ t_2^*,\ldots,\ t_i,\ldots,\ t_M^c]^\top$.
3: Solve the optimization problem below using Theorem 3

$$t_i^* = \arg\min_{t_i} \tilde{f}(\Lambda_i). \tag{41}$$

4: **end for**
5: **return** $\Lambda_0^g = [t_1^*,\ t_2^*,\ldots,\ t_M^*]^\top$

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. Three methods are now proposed to provide an appropriate initial tracking time allocation for the algorithm.

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

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

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

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

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. $t_1^*,\ldots,\ t_{i-1}^*$ and $t_{i+1}^c,\ldots,\ t_M^c$. Algorithm 1 uses some computation time, but tries to provide a better initial tracking time allocation Λ_0^g rather than the central initial tracking time allocation 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

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

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 $\varepsilon > 0$, $\delta > 0$ are small scalars which depend on the tracking precision requirement and performance requirement, respectively.

Algorithm 2 Low resolution initial tracking time allocation

Input: $S(A,B,C)$, r^p, Θ and T_s

Output: Low resolution initial tracking time allocation Λ_0^l

1: Discretize the infinite set Θ at a sample rate of T_s to obtain $\tilde{\Theta}$ which is a finite subset of Θ.
2: Solve the optimization problem below using blind search

$$\Lambda^* = \arg\min_{\Lambda\in\tilde{\Theta}} \tilde{f}(\Lambda). \tag{43}$$

3: **return** $\Lambda_0^l = \Lambda^*$

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

Input: Λ_0, $S(A,B,C)$, r^p and Θ

Output: Optimal tracking time allocation Λ_{opt} and corresponding input u_{opt}

1: **Initialization:** Loop number $j = 0$
2: Implement Stage One update (21) with $\Lambda = \Lambda_0$ experimentally using feedback plus feedforward update (34) until the convergence is achieved, i.e. $\|e_k^p\| < \varepsilon\|r^p\|$; record converged input $u_\infty^{ex}(\Lambda_0)$ and input energy $\tilde{f}(\Lambda_0)$.
3: **repeat**
4: Implement Stage Two update (26) with $r^p = G_{\Lambda_j}^p u_\infty^{ex}(\Lambda_j)$ in (25) or (38) while computing the gradient.
5: Set $j \to j+1$.
6: Implement Stage One update (21) with $\Lambda = \Lambda_j$ experimentally using feedback plus feedforward update (34) until the convergence is achieved, i.e. $\|e_k^p\| < \varepsilon\|r^p\|$; 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 $u_{opt} = u_\infty^{ex}(\Lambda_j)$

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

$$\Omega = \{u(t) \in \mathbb{R}^\ell : \|u(t)\| \leqslant M(t),\ t \in [0,T]\}. \tag{44}$$

Input amplitude constraint

$$\Omega = \{u(t) \in \mathbb{R}^\ell : \lambda(t) \preceq u(t) \preceq \mu(t),\ t \in [0,T]\}. \quad (45)$$

Input sign constraint

$$\Omega = \{u(t) \in \mathbb{R}^\ell : 0 \preceq u(t),\ t \in [0,T]\}. \quad (46)$$

Input energy constraint

$$\Omega = \{u(t) \in \mathbb{R}^\ell : \int_0^T (u(t))^\top u(t)\mathrm{d}t \leqslant M,\ t \in [0,T]\}. \quad (47)$$

With constraints, the optimal tracking time allocation problem becomes

$$\begin{aligned} \underset{u,\ y,\ \Lambda}{\text{minimize}} \quad & f(u,\ y) \\ \text{subject to} \quad & r^p = G^p_\Lambda u, \\ & y = Gu, \\ & \Lambda \in \Theta,\ u \in \Omega. \end{aligned} \quad (48)$$

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 within Section III, the constrained optimization problem (48) becomes

$$\min_{\Lambda\in\Theta}\left\{\min_u f(u,y),\ \text{subject to } G^p_\Lambda u = r^p,\ y = Gu,\ u \in \Omega\right\} \quad (49)$$

suggesting a modified two stage design framework as:

- *Stage One:*

$$\begin{aligned} \underset{u\in\Omega}{\text{minimize}} \quad & \|u\|^2_R \\ \text{subject to} \quad & r^p = G^p_\Lambda u \end{aligned} \quad (50)$$

  whose the solution is denoted as $\hat{u}_\infty(\Lambda)$.
- *Stage Two:*

$$\min_{\Lambda\in\Theta}\{\tilde{f}(\Lambda) := \|\hat{u}_\infty(\Lambda)\|^2_R\}. \quad (51)$$

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.

Note that now Stage One does not have a direct analytical solution, but the norm-optimal ILC algorithm with successive projection proposed in [32] can be applied to solve the modified Stage One optimization problem (50). 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

$$u_{k+1} = \arg\min_{u\in\Omega}\{\|e^p\|^2_Q + \|u-u_k\|^2_R\}. \quad (52)$$

  This algorithm converges to the minimum error norm. The constrained QP problem (52) 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 [32], [33] for more information.
- *Method 2:* Solve the unconstrained input norm-optimal point-to-point ILC optimization problem

$$\tilde{u}_{k+1} = \arg\min_u\{\|e^p\|^2_Q + \|u-u_k\|^2_R\} \quad (53)$$

  and then perform a simple input projection

$$u_{k+1} = \arg\min_{u\in\Omega}\|u-\tilde{u}_{k+1}\|. \quad (54)$$

  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 rationale behind Stage Two is that for each tracking time allocation Λ on Θ there exists an optimal input energy $\tilde{f}(\Lambda)$ and we can obtain an optimal $\Lambda^* \in \Theta$ to minimize the input energy $\tilde{f}(\Lambda)$. The Stage Two optimization problem (51) can be equivalently solved using

$$\Lambda_{j+1} = \arg\min_{\Lambda\in\Theta}\tilde{f}_j(\Lambda) \quad (55)$$

which can be solved by Theorem 3 where

$$\tilde{f}_j(\Lambda) = \|u_\infty(\Lambda)\|^2_R + \rho\left\|u_\infty(\Lambda) - \hat{u}_\infty(\Lambda_j)\right\|^2_R,\ \rho \geqslant 0. \quad (56)$$

Note that in the modified stage two design, the input constraint is decoupled from the optimization problem and thus can be solved analytically. In practical implementation, the modified Stage One solution $\hat{u}_\infty(\Lambda_j)$ in the update (55) should be considered to be the converged input $\hat{u}^{ex}_\infty(\Lambda_j)$ obtained experimentally.

These new solution forms combine to generate Algorithm 4 which solves the input constrained optimization problem (49).

Remark 4. The modified Stage Two optimization problem can be also solved by

$$\Lambda_{j+1} = P_\Theta(\Lambda_j - \gamma_j \cdot \nabla\tilde{f}(\Lambda_j)). \quad (57)$$

However, we can only use the estimation method (38) to compute the gradient $\nabla\tilde{f}(\Lambda)$ rather than compute it analytically, as the modified Stage One does not have an analytical solution.

C. Convergence Properties of the Algorithm

Algorithm 4 has the following convergence properties: The convergence properties of the gradient projection update (55) are described in the next theorem.

Theorem 4. Suppose perfect tracking is achievable and $\rho \leqslant 1$, the analytical input energy resulting from (55) satisfies

$$\left\|u_\infty(\Lambda_{j+1})\right\|^2_R \leqslant \left\|\hat{u}_\infty(\Lambda_j)\right\|^2_R. \quad (58)$$

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

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

Input: Λ_0, $S(A,B,C)$, r^p, Θ and Ω

Output: Optimal tracking time allocation Λ_{opt} and corresponding input u_{opt}

1: **Initialization:** Loop number $j = 0$
2: Implement Stage One update (52) or (53)-(54) with $\Lambda = \Lambda_0$ experimentally using feedback plus feedforward update (34) until the convergence is achieved, i.e. $\|e_k^p\| < \varepsilon\|r^p\|$; record converged input $\hat{u}_\infty^{ex}(\Lambda_0)$ and input energy $\tilde{f}(\Lambda_0)$.
3: **repeat**
4: Implement Stage Two update (55) with $r^p = G_{\Lambda_j}^p \hat{u}_\infty^{ex}(\Lambda_j)$ in (24) or (38) while computing the gradient.
5: Set $j \to j+1$.
6: Implement Stage One update (52) or (53)-(54) with $\Lambda = \Lambda_j$ experimentally using feedback plus feedforward update (34) until the convergence is achieved, i.e. $\|e_k^p\| < \varepsilon\|r^p\|$; record converged input $\hat{u}_\infty^{ex}(\Lambda_j)$ and input energy $\tilde{f}(\Lambda_j)$.
7: **until** $\left|\tilde{f}_j(\Lambda_j) - \tilde{f}(\Lambda_{j-1})\right| < \delta\left|\tilde{f}(\Lambda_{j-1})\right|$
8: **return** $\Lambda_{opt} = \Lambda_j$ and $u_{opt} = \hat{u}_\infty^{ex}(\Lambda_j)$

Theorem 4 states only that the next loop's unconstrained minimum energy is no larger than the constrained minimum energy of the current loop, but it still provides useful information about the convergence performance of the constrained input condition. We have undertaken a series of simulations using different models and input constraints to examine the convergence performance of Stage Two update (55), and a representative example is now given.

Example 2. Consider the same design objectives as the multiple case ($M = 2$) example of Section III and assume input saturation constraint (44) with $M(t) = 1.5$. Perform Algorithm 4 for a total number of 30 loops with the Stage Two update (55) in simulation assuming $\sigma = 0.1$, $\beta = 0.8$ and $\gamma = 0.08$. Plot the corresponding constrained and unconstrained minimum input energy $\left\|\hat{u}_\infty(\Lambda_j)\right\|_R^2$ and $\left\|u_\infty(\Lambda_j)\right\|_R^2$ for each loop in Figure 4. The results in Figure 4 not only verify (58), but also show monotonic convergence of the constrained minimum input energy.

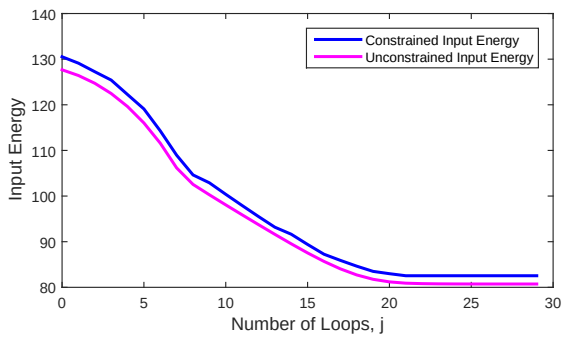

Fig. 4. Convergence Performance Comparison between Constrained and Unconstrained Minimum Input Energy at Each Loop.

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 5 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 are both 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, and uses the reference trajectory shown in Figure 6. Because the three axes are orthogonal and controlled separately, the gantry robot can be considered to comprise three separate single-input single-output (SISO) systems.

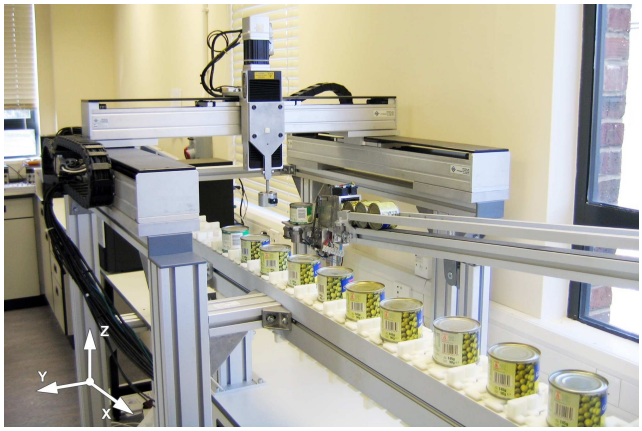

Fig. 5. Multi-axis Gantry Robot Test Platform.

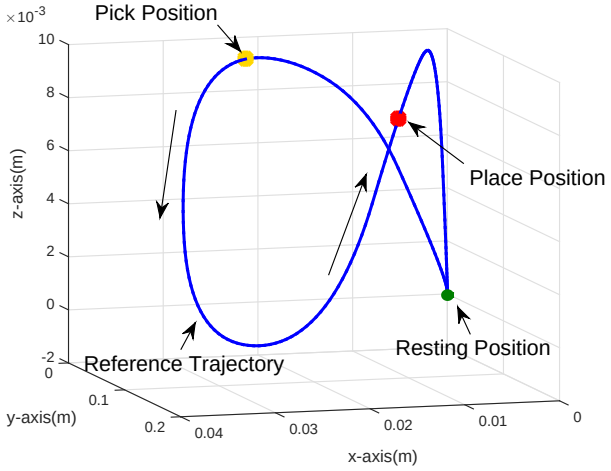

Fig. 6. 3D Reference Trajectory of Gantry Robot

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. 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 $t_1^+ = 0.99$, $t_2^+ = 1.99$, $t_1^- = 0.01$, and $t_2^- = 1.01$ 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

$$G(s) = \frac{G_z(s)}{1 + KG_z(s)}. \tag{59}$$

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 $Q = qI$, $R = rI$ where q and r are positive scalars which satisfy $q/r = 500,000$, and the gradient is obtained by the estimation (38) with analytical calculation.

B. Experimental Results using an Inaccurate Model without Input Constraints

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

$$\hat{G}_z(s) = \frac{0.03}{s} \tag{60}$$

with a feedback gain $K = 30$, which hence provides the state space form $S(A,B,C)$

$$A = -0.9,\ B = 0.125,\ C = 0.24.$$

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 $\gamma = 0.03, 0.04, 0.05$.

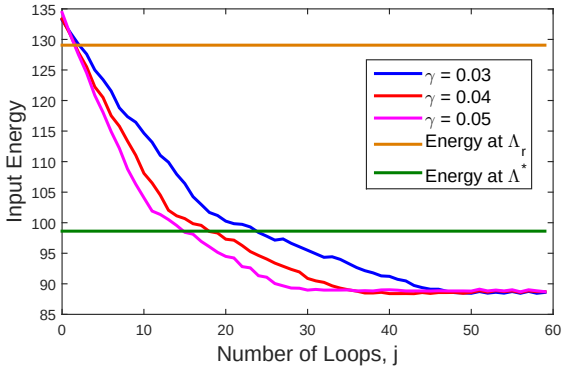

Fig. 7. Experimental Input Energy Results using Inaccurate Model at Unconstrained Condition.

The experimental optimal input energy $\tilde{f}(\Lambda_k)$ at each loop is plotted in Figure 7 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 $\tilde{f}(\Lambda_r)$.

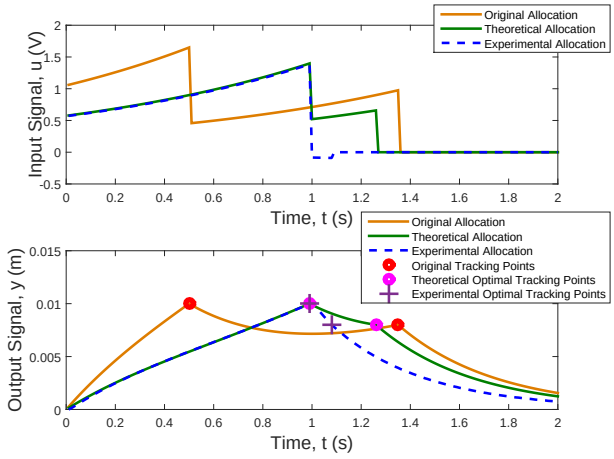

Fig. 8. Converged Input and Output Trajectory Comparison using Inaccurate Model at Unconstrained Condition.

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, theoretical optimal and experimental optimal allocations are plotted in Figure 8 and illustrate how the experimentally obtained optimal allocation takes advantage of the other two allocations.

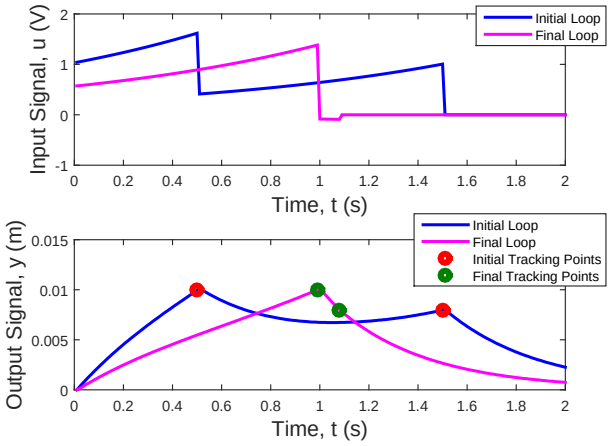

Fig. 9. Experimental Converged Input and Output Trajectories for Initial and Final Loops using Inaccurate Model at Unconstrained Condition.

The experimental final converged input trajectory for the initial and final loops of the algorithm are compared in Figure 9, 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 experimental final converged output trajectory for the initial and final loops of the algorithm are also compared in Figure 9, and 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 0.00032 at Step 6 of the 60^{th} loop.

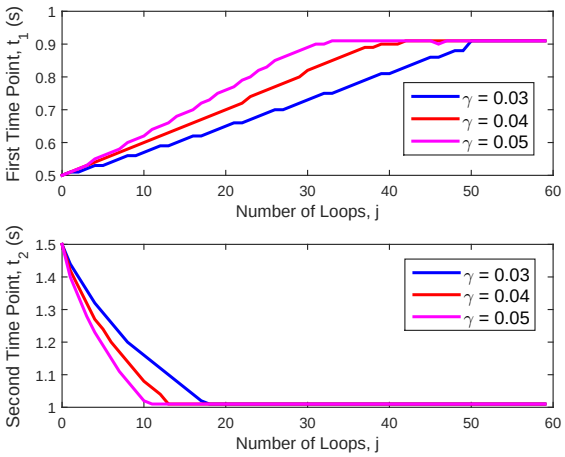

Fig. 10. Experimental Time-Point Position Results at Each Loop using Inaccurate Model at Unconstrained Condition.

To further illustrate the performance of the proposed algorithm, the convergence of the tracking time allocation is shown in Figure 10. For each different value of γ, the tracking time allocation at each loop are plotted in this figure, and all converge to identical values. It should be noted that the 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, which contributes to a lower input energy. Experiments using the other initial tracking time allocations, e.g. the low resolution initial tracking time allocation, yield similar levels of performance.

C. Experimental Results using an Accurate Model without Input Constraints

Next the accurate model (33) is employed within (59) with feedback gain $K = 100$ to assess the performance of the proposed algorithm. Now the state space form $S(A,B,C)$ is

$$A = \begin{bmatrix} -707.6 & -662.824 & -41.225 \\ 512 & 0 & 0 \\ 0 & 64 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0.5 \\ 0 \\ 0 \end{bmatrix},$$
$$C = \begin{bmatrix} 0 & 0.062 & 0.825 \end{bmatrix}.$$

Algorithm 3 is again applied for 25 updating loops. In Step 4, the generalized Armijo step size (27) is applied with the same parameters as in the previous subsection.

For step size chosen under different values of γ, the optimal input energy $\tilde{f}(\Lambda_k)$ at each loop is plotted in Figure 11. The optimal energy $\tilde{f}(\Lambda_r) = 125.13$ required for the gantry robot to track the *a priori* tracking time allocation is also plotted for comparison. The algorithm provides an optimal energy 78.67, and reduces the input energy by more than 30% compared with the operating energy $\tilde{f}(\Lambda_r)$. Blind search has been performed

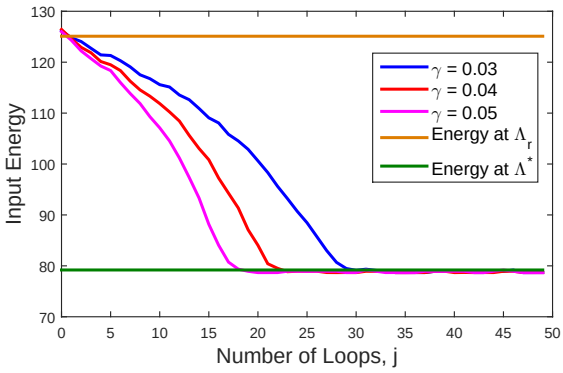

Fig. 11. Experimental Input Energy Results using an Accurate Model at Unconstrained Condition.

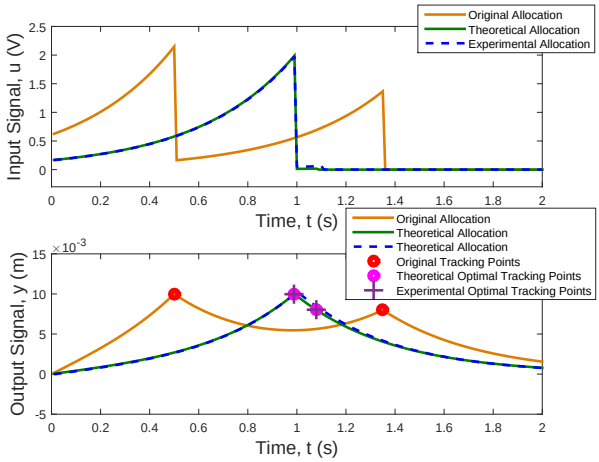

Fig. 12. Converged Input and Output Trajectory Comparison using an Accurate Model at Unconstrained Condition.

in simulation with the accurate model, and the theoretical optimal tracking time allocation is $\Lambda^* = [0.99, 1.08]^\top$, which yields the theoretical optimal energy $\tilde{f}(\Lambda^*) = 79.19$. As can be seen from the figure, the converged input is very close to the theoretical one $\tilde{f}(\Lambda^*)$. Also, the converged input and output trajectories at the original, theoretical optimal and experimental optimal allocations have been plotted in Figure 12 and it can be shown that the converged trajectories of the theoretical optimal and experimental optimal allocations are almost the same.

The experimental final converged input trajectory for the initial and final loops of the algorithm are shown in Figure 13. For comparison, the experimental final converged output trajectory of the initial and final loops of the algorithm are also shown in Figure 13, and the reference at the special tracking time is marked with red and green circles. It is obvious that the output converges to the special tracking points, confirming that the algorithm not only optimizes the input energy, but also maintains satisfactory tracking performance.

To further illustrate the performance of the proposed algorithm with accurate model, the corresponding convergence of the tracking time allocation is shown in Figure 14. For each different value of γ, the tracking time allocation at each loop is plotted in this figure, and it is clear that the final converged tracking time allocation is $[0.99, 1.09]^\top$, which is close to the

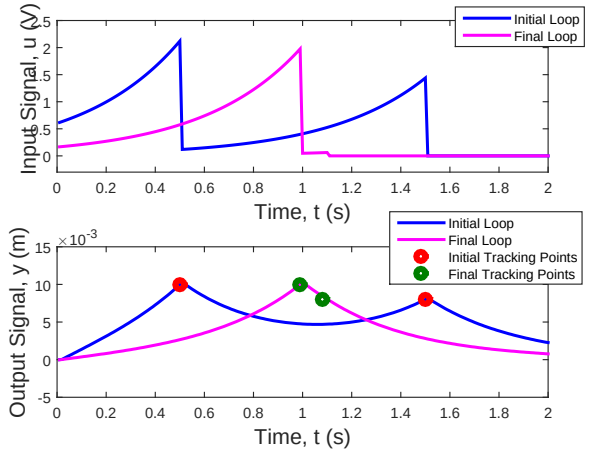

Fig. 13. Experimental Converged Input and Output Trajectories for Initial and Final Loops using an Accurate Model at Unconstrained Condition.

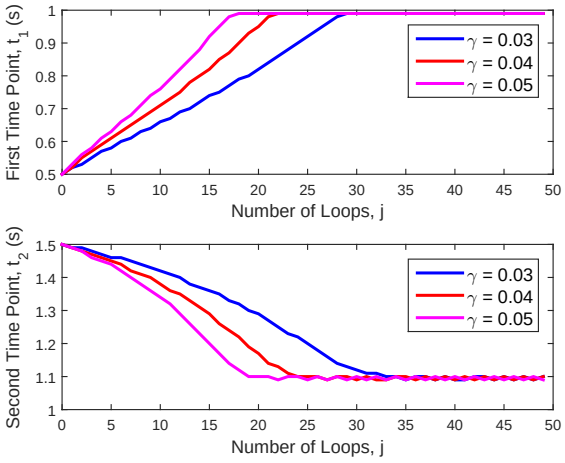

Fig. 14. Experimental Time-Point Position Results at Each Loop using an Accurate Model at Unconstrained Condition.

theoretical one Λ^*. A range of different initial tracking time allocations have been used to perform the experiments, and their results are similar to those presented in Figure 11.

D. Experimental Results using an Accurate Model with Input Constraints

Now assume the gantry robot has an input saturation constraint (44) with $M(t) = 1.8$, and the accurate model $S(A,B,C)$ obtained in the previous subsection is used. Algorithm 4 is performed with 25 updating loops. For implementational simplicity, we use solution (53)-(54) during Step 2 and 6, and solution (57) with generalized Armijo step size (27) using $\sigma = 0.1$, $\beta = 0.8$ and $\gamma = 0.05, 0.06, 0.07$ during Step 4.

Similar data process method has been used to plot the optimal input energy $\tilde{f}(\Lambda_k)$ under each value of γ at each loop in Figure 15 together with the optimal energy $\tilde{f}(\Lambda_r) = 125.3$ at the *a priori* tracking time allocation. It can be seen from the figure that the optimal energy obtained by the algorithm is 78.72, which is 30% less than the operating energy $\tilde{f}(\Lambda_r)$. Blind search is then performed in simulation with the accurate model, which provides the theoretical optimal tracking time allocation $\Lambda^* = [0.99, 1.08]^\top$ and the theoretical optimal

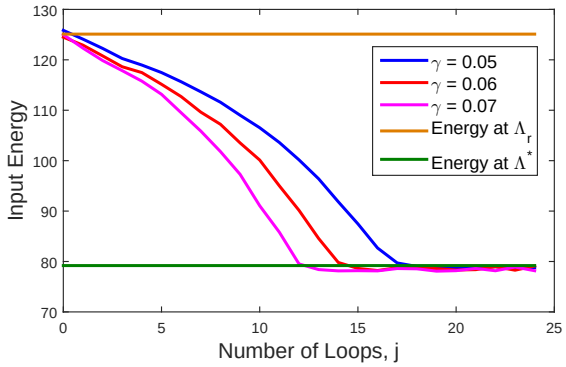

Fig. 15. Experimental Input Energy Results using an Accurate Model at Constrained Condition.

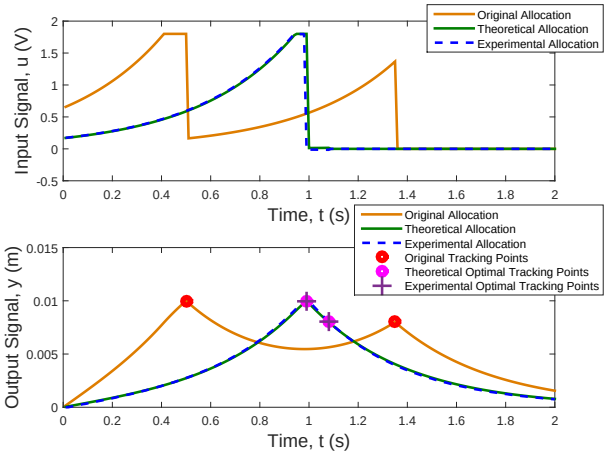

Fig. 16. Converged Input and Output Trajectory Comparison using an Accurate Model at Constrained Condition.

energy $\tilde{f}(\Lambda^*) = 79.32$. In this figure, it is obvious that the converged input is close to the theoretical one $\tilde{f}(\Lambda^*)$ as the model is accurate. For further comparison, the converged input and output trajectories at the original, theoretical optimal and experimental optimal allocations have been plotted in Figure 16, which shows that the converged trajectories of the theoretical optimal and experimental optimal allocations are almost the same.

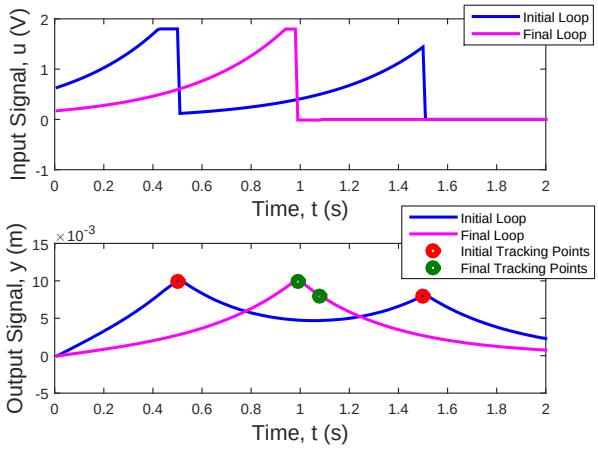

Fig. 17. Experimental Converged Input and Output Trajectories for Initial and Final Loops using an Accurate Model at Constrained Condition.

The experimental final converged input trajectory for the initial and final loops of the algorithm are shown in Figure 17, and it is clear that the input signal meets the input saturation constraint. Furthermore, the experimental final converged output trajectory of the initial and final loops of the algorithm are also shown in Figure 17 with red and green circles standing for the reference at the special tracking time. It shows that the converged output trajectories pass through the special tracking points, which verifies that the algorithm not only optimizes the input energy, but also maintains satisfactory tracking performance.

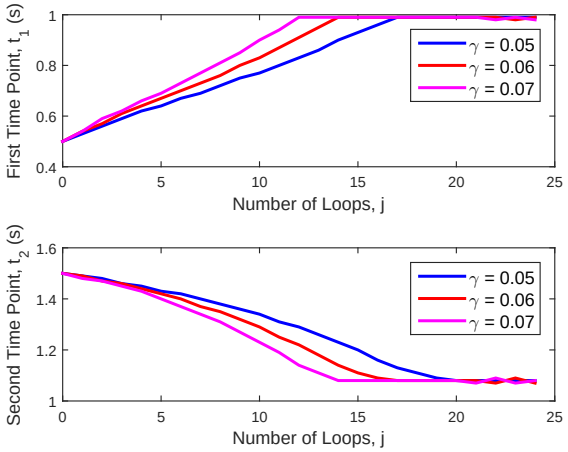

Fig. 18. Experimental Time-Point Position Results at Each Loop using an Accurate Model at Constrained Condition.

In terms of the algorithm's performance, the tracking time allocation at each value of γ and each loop is shown in Figure 18. From the figure, the final converged tracking time allocation is $[0.99, 1.08]^\top$, which is exactly the same as the theoretical one Λ^* to two decimal place accuracy. Different initial tracking time allocations have also been used to perform the experiments under constrained condition, whose results are similar to those presented in Figure 15.

VII. Conclusion

Tracking time allocation plays an important role in point-to-point ILC and can significantly affect the system performance. This paper developed 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 has been developed. Global solutions to Stage One are derived for efficient implementation. For Stage Two there are no direct analytical solutions of the optimization problem, and hence an iterative algorithm based on the gradient method has been proposed. The implementation procedures have been 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 improvement 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 its certain effectiveness in practice, a rigorous analysis of the algorithm's robustness properties will be undertaken in future research to show the degree of robustness against model uncertainty. Furthermore, the design framework will be expanded to minimize total input energy of all three axes while performing a pick-and-place task. In addition, other methods, e.g. the projected Newton method [34], will also be used to solve the Stage Two optimization problem.

Appendix

A. Proof of Theorem 1

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

$$\min_u \{\|e^p\|_Q^2 + \|u-u_k\|_R^2 :\ e^p = r^p - y^p,\ y^p = G_\Lambda^p u\} \tag{61}$$

to get the control input u_{k+1}. The problem (61) has an identical structure to the norm-optimal ILC problem described in [35], with the only difference being the definitions of the operators, signals and underlying Hilbert spaces. Therefore, the iterative solution can be expressed as

$$u_{k+1} = u_k + G_\Lambda^{p*} e_{k+1}^p \Rightarrow e_{k+1}^p = (I + G_\Lambda^p G_\Lambda^{p*})^{-1} e_k^p \tag{62}$$

which gives rise to (21).

It is proved in [18] 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.

$$\lim_{k\to\infty} e_k^p = 0,\ \lim_{k\to\infty} u_k = u_\infty.$$

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_Λ^{p*} is obtained in [18] from the definition

$$\left\langle (\omega_1, ..., \omega_M),\ G_\Lambda^p u \right\rangle_Q = \left\langle G_\Lambda^{p*}(\omega_1, ..., \omega_M),\ u \right\rangle_R \tag{63}$$

which gives rise to

$$(G_i^* \omega_i)(t) = \begin{cases} R^{-1} B^\top e^{A^\top (t_i - t)} C^\top Q \omega_i,\ 0 \leqslant t \leqslant t_i, \\ 0,\ t > t_i. \end{cases} \tag{64}$$

The equation (64) can be further written as

$$(G_i^* \omega_i)(t) = R^{-1} B^\top p_i(t) \tag{65}$$

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

$$\dot{p}_i(t) = -A^\top p_i(t),\ p_i(t_i-) = C^\top Q \omega_i. \tag{66}$$

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

$$u(t) = \sum_{i=1}^{M} (G_i^* \omega_i)(t) = R^{-1} B^\top p(t) \tag{67}$$

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_Λ^{p*}. Therefore, the $(i,j)^{th}$ block of the matrix $G_\Lambda^p G_\Lambda^{p*}$ can be computed as the equation (23).

It has also been proved in [18] that the limit u_∞ exists with

$$u_\infty - u_0 = G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1} e_0^p. \tag{68}$$

As the initial input $u_0=0$ which makes $e_0^p = r^p$, the analytical solution (24) for $u_\infty(\Lambda)$ is obtained.

B. Proof of Lemma 1

Substituting the analytical solution (24) into the optimization 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 tracking 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

$$\Psi_{t_1} = \int_0^{t_1} Ce^{A(t_1-t)} BR^{-1} (Ce^{A(t_1-t)}B)^\top\ dt. \tag{69}$$

The Stage Two optimization problem becomes

$$\min_{\Lambda\in\Theta}\|u_\infty(\Lambda)\|_R^2 = \min_{\Lambda\in\Theta}\ \left\langle r^p,\ \Psi_{t_1}^{-1} r^p \right\rangle_Q. \tag{70}$$

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

$$\begin{aligned}&\langle x,\ (\Psi_T - \Psi_{t_1})x\rangle_Q = \\ &\quad \langle x,\ \int_{t_1}^{T} Ce^{A(t_1-t)} BR^{-1} (Ce^{A(t_1-t)}B)^\top\ dt x\rangle_Q \geqslant 0.\end{aligned} \tag{71}$$

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. [28] 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

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

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

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

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

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

Hence (77) and (78) suggest the jump conditions

$$\begin{aligned}&K(t_i-) = K(t_i+) + C^\top QC,\ 1 \leqslant i \leqslant M, \\ &\xi_{k+1}(t_i-) = \xi_{k+1}(t_i+) + C^\top Q e_k(t_i),\ 1 \leqslant i \leqslant M\end{aligned} \tag{79}$$

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

$$\begin{aligned}&0 = \dot{K}(t) + A^\top K(t) + K(t)A - K(t)BR^{-1}B^\top K(t),\ K(T)=0, \\ &0 = \dot{\xi}_{k+1}(t) + (A^\top - K(t)BR^{-1}B^\top)\xi_{k+1}(t), \xi_{k+1}(T) = 0\end{aligned} \tag{80}$$

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

$$\tilde{f}_j(\Lambda_{j+1}) \leqslant \tilde{f}_j(\Lambda_j) \tag{81}$$

holds and hence it follows that

$$\begin{aligned}\left\|u_\infty(\Lambda_{j+1})\right\|_R^2 &\leqslant \left\|u_\infty(\Lambda_{j+1})\right\|_R^2 + \rho\left\|u_\infty(\Lambda_{j+1}) - \hat{u}_\infty(\Lambda_j)\right\|_R^2 \\ &\leqslant \left\|u_\infty(\Lambda_j)\right\|_R^2 + \rho\left\|u_\infty(\Lambda_j) - \hat{u}_\infty(\Lambda_j)\right\|_R^2 \\ &= \left\|u_\infty(\Lambda_j)\right\|_R^2 + \rho\left\|u_\infty(\Lambda_j)\right\|_R^2 + \rho\left\|\hat{u}_\infty(\Lambda_j)\right\|_R^2 \\ &\quad - 2\rho\left\langle \hat{u}_\infty(\Lambda_j),\ u_\infty(\Lambda_j)\right\rangle_R. \end{aligned} \tag{82}$$

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

$$\begin{aligned}\left\langle \hat{u}_\infty(\Lambda),\ u_\infty(\Lambda)\right\rangle_R &= \left\langle \hat{u}_\infty(\Lambda),\ G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1} r^p\right\rangle_R \\ &= \left\langle G_\Lambda^p \hat{u}_\infty(\Lambda),\ (G_\Lambda^p G_\Lambda^{p*})^{-1} r^p\right\rangle_Q \\ &= \left\langle r^p,\ (G_\Lambda^p G_\Lambda^{p*})^{-1} r^p\right\rangle_Q \\ &= \left\langle (G_\Lambda^p G_\Lambda^{p*})(G_\Lambda^p G_\Lambda^{p*})^{-1} r^p,\ (G_\Lambda^p G_\Lambda^{p*})^{-1} r^p\right\rangle_Q \\ &= \left\langle G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1} r^p,\ G_\Lambda^{p*}(G_\Lambda^p G_\Lambda^{p*})^{-1} r^p\right\rangle_R \\ &= \left\langle u_\infty(\Lambda),\ u_\infty(\Lambda)\right\rangle_R. \end{aligned} \tag{83}$$

Substitute (83) into (82) to give

$$\left\|u_\infty(\Lambda_{j+1})\right\|_R^2 \leqslant (1-\rho)\left\|u_\infty(\Lambda_j)\right\|_R^2 + \rho\left\|\hat{u}_\infty(\Lambda_j)\right\|_R^2. \tag{84}$$

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

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

and it follows that

$$(1-\rho)\left\|u_\infty(\Lambda_j)\right\|_R^2 \leqslant (1-\rho)\left\|\hat{u}_\infty(\Lambda_j)\right\|_R^2 \tag{85}$$

as $(1-\rho)$ is non-negative. Hence combine (85) and (84) together to generate the inequality (58).

ACKNOWLEDGMENT

This research project is funded by the China Scholarship Council (CSC).

REFERENCES

[1] 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.

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] C. T. Freeman, E. Rogers, J. H. Burridge, A.-M. Hughes, and K. L. Meadmore, *Iterative Learning Control for Electrical Stimulation and Stroke Rehabilitation*, 1st ed. Springer-Verlag London, 2015.

[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 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.

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

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

[21] 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.

[22] 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.

[23] 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.

[24] K. Hamamoto and T. Sugie, "Iterative learning control for robot manipulators using the finite dimensional input subspace," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 4, pp. 632–635, 2002.

[25] P. Hagqvist, A. Heralic, A.-K. Christiansson, and B. Lennartson, "Resistance based iterative learning control of additive manufacturing with wire," *Mechatronics*, vol. 31, pp. 116–123, 2015.

[26] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous UAV guidance," *Journal of Intelligent and Robotic Systems*, pp. 65–100, 2010.

[27] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

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

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

[30] 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.

[31] 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.

[32] 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.
[33] 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.
[34] 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.
[35] 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.
[36] N. Amann, "Optinal algorithms for iterative learning control," Ph.D. dissertation, University of Exeter, Exeter, 1996.