Iterative Learning Control for Path Following Tasks with Performance Optimization

Yiyang Chen, Bing Chu* and Christopher T. Freeman

***Abstract*—The classical problem setup of iterative learning control (ILC) is to enforce tracking of a reference profile specified at all time points in the fixed task duration. The removal of the time specification releases significant design freedom in how the path is followed, but has not been fully exploited in the literature. This paper unlocks this extra design freedom by formulating the ILC task description to handle repeated path following tasks, e.g. welding and laser cutting, which aim at following a given "spatial" path defined in the output space without any temporal information. The general ILC problem is reformulated for ILC design with the inclusion of an additional performance index, and the class of piecewise linear paths is characterized for the reformulated problem setup. A Two Stage design framework is proposed to solve the characterized problem, and yields a comprehensive algorithm based on an ILC update and a gradient projection update. This algorithm is verified on a gantry robot experimental platform to demonstrate its practical efficacy and robustness against model uncertainty.**

***Index Terms*—iterative learning control; optimization; path following**

I. INTRODUCTION

In a broad class of automation tasks, e.g. assembling, welding and drilling, the task design objective is to follow a path, which is specified as a set of spatial points without temporal information. The concept of "path following" differs from that of "reference tracking" as it removes all temporal tracking information. In other words, it does not specify how fast the system moves along the given path. In path following problems the speed profile (as a function of time) along the path is considered as a time-varying variable due to the elimination of the temporal constraints. This feature releases significant control design freedom compared to the reference tracking problem setup with a predefined speed profile, but also leads to substantial difficulties in the control design phase.

The class of path following problems has been considered in existing research, e.g. [1]–[4], in the application of autonomous air or surface vehicles. They implicitly or explicitly design the speed profile using the measured data via a feedback control setup, such that the desired path is followed asymptotically even in the presence of model uncertainty (e.g. disturbances and parametric model mismatches). However, these techniques are all specific to vehicle steering problems and no performance optimization rather than path following is considered, which has not fully exploited the design freedom. In the area of robotic motion planning, existing research, e.g. [5]–[8], has capitalized on this design freedom to optimize an additional benefit while maintaining path following accuracy. Since they do not incorporate a learning procedure based on previous executions of the task, their along the trial performance is inevitably limited for repeated path following tasks.

Conversely, a framework termed Iterative Learning Control (ILC) has been formulated to learn repeated tracking tasks, but has not yet exploited the design freedom of path following. ILC is suitable for a large class of path following applications, e.g. gantry robot [9], mobile robot [10], wafer stage [11] and stroke rehabilitation [12]. It updates the system input signal at the end of each trial based on the measured data, e.g. the tracking error, from the previous trials, which theoretically reduces the error to zero. See [13]–[17] for detailed overviews of ILC. Note that most existing ILC research, e.g. [18]–[20], focuses on tracking a fixed reference profile specified at all points over a predefined time horizon.

In the last decade, ILC research has started to remove the postulate of a fixed reference profile. Vehicle path following was considered in [21], and a feedforward ILC update law was developed employing the local symmetrical double-integral of the feedback control signal obtained from the previous trial. The seminal work in [22] used a P-type ILC algorithm to solve the 2D corner tracking problem using the distance between the output trajectory and the desired path for motion control applications. The research in [23], [24] aimed at reducing the periodic varying torque ripple and improving the additive manufacturing quality respectively. They redefined the system input and output signals using spatial coordinates, and reformulated existing ILC algorithms into the spatial formats. In [25], [26], the minimum time path following problem was considered for a 2D path by explicitly updating the nominal system model using the measured data and then solving the problem using the algorithms in [5], [6], which however has not inherited the input update procedure of ILC.

The aforementioned ILC approaches are either application specific or not concerned with path following. In addition, they only consider path following accuracy, and have not fully harnessed the available design freedom in the temporal domain to optimize an extra performance index. The optimization of

This work is partially funded by the China Scholarship Council (CSC), the Excellent Young Scholar Program of Soochow University under Grant Q411700520, the ZZU-Southampton Collaborative Research Project under Grant 16306/01, the Royal Society International Exchanges Award under Grant IE161369 and Natural Science Foundation of China under Grant 61773232.

Y. Chen is with the School of Mechanical and Electrical Engineering, Soochow University, Suzhou, 215137, China (e-mail: yychen90@suda.edu.cn). The research reported in this work was performed when Y. Chen was at the School of Electronics and Computer Science, University of Southampton, Southampton, UK.

B. Chu and C. T. Freeman are with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (e-mail: b.chu@soton.ac.uk; cf@ecs.soton.ac.uk).

*The corresponding author.

such a performance index brings significant practical benefits, such as control effort reduction, machine damage avoidance and increased manufacturing efficiency. Note that the authors' previous work in [27]–[30] has considered optimization of the intermediate time instant within point-to-point tracking problems and the path following problem with fixed intermediate time instants. However, they have not formulated a general ILC framework to address optimal path following problems in a principled manner.

To address the above limitations and need, an ILC framework is developed in this paper to solve the path following problems for a broad class of systems with high tracking accuracy, and simultaneously minimizes a specified performance index of interest, e.g. control effort. The path following ILC problem is first defined in Section II, and then reformulated into an equivalent problem for control design in Section III. The class of piecewise linear paths is characterized for the reformulated problem setup, which comprises a substantial extension to existing ILC scope involving constraint handling, embedded projections and time variable updates. An expanded version of the Two Stage design framework proposed in [27], [28] is applied to solve the equivalent ILC problem. In Section IV an iterative algorithm is obtained with desired robust convergence properties, which minimizes the control effort and ensures path following accuracy. This algorithm is verified experimentally in Section V on a gantry robot test platform to demonstrate its practical efficacy and robustness against model uncertainty. Note that some initial results in this paper have been reported in [31], but do not embed system constraint handling or provide the robust convergence properties.

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; $L^\ell_2[0,\ T]$ 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 of x onto the set Θ in some Hilbert space; $\mathrm{Im}\,M$ and $\mathrm{Ker}\,M$ are the image and kernel of an operator M respectively.

II. PROBLEM FORMULATION

This section introduces the system dynamics as well as the path following task design objective, which together form a general path following ILC problem. To further exploit the design freedom, an additional performance index is embedded as a cost function to be optimized within this problem setup.

A. System Dynamics

Consider a multi-input, multi-output linear time-invariant system given in a state space form

$$\begin{aligned}\dot{x}_k(t) &= Ax_k(t)+Bu_k(t),\\ y_k(t) &= Cx_k(t),\end{aligned} \tag{1}$$

where $t\in[0,\ T]$ is the time instant; $0<T<\infty$ denotes the ILC trial length; the subscript $k\in\mathbb{N}$ denotes the trial number;

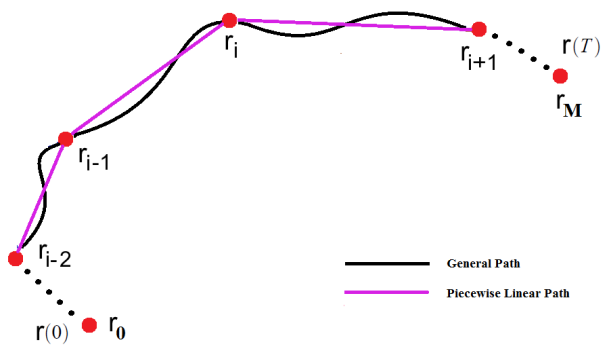

Fig. 1. An example of paths as set of spatial points in $\mathbb{R}^m$.

$x_k(t)\in\mathbb{R}^n$, $u_k(t)\in\mathbb{R}^\ell$ and $y_k(t)\in\mathbb{R}^m$ are the state, input and output respectively; A, B and C are system matrices of compatible dimensions. At the end of each trial, the state is reset to an identical initial value, i.e. $x_k(0)=x_0$. The system has an equivalent operator form

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

In the above operator form, $y_k\in L^m_2[0,\ T]$, $u_k\in L^\ell_2[0,\ T]$ are the system input and output signals. The output Hilbert space $L^m_2[0,\ T]$ is defined with inner product and associated induced norm

$$\langle x,\ y\rangle_S = \int_0^T x^\top(t)Sy(t)\mathrm{d}t,\ \ \|y\|_S = \sqrt{\langle y,\ y\rangle_S}, \tag{3}$$

in which $S\in\mathbb{S}^m_{++}$ is a weight matrix. The input Hilbert space $L^\ell_2[0,\ T]$ is defined in a similar fashion to (3) by replacing y and S with u and $R\in\mathbb{S}^\ell_{++}$. The system operator $G: L^\ell_2[0,\ T]\to L^m_2[0,\ T]$ and the effect of initial conditions $d\in L^m_2[0,\ T]$ have analytic forms

$$(Gu_k)(t) = \int_0^t Ce^{A(t-s)}Bu_k(s)\mathrm{d}s,\ \ d(t)=Ce^{At}x_0, \tag{4}$$

which are obtained from the state space form (1). Without loss of generality, zero initial conditions are assumed, i.e. $x_0=0$ and therefore $d=0$. The results in this paper hold for non-zero initial conditions, simply by subtracting d from both the output, y_k, and reference.

B. Path Following Design Objective

Any continuous path can be defined in the output space by introducing a continuous function $r(s)$ parametrized by spatial variable $s\in[0,\ 1]$ with each point $r(s)\in\mathbb{R}^m$. An example of such a path is shown in Figure 1 as the black curve. Defining a continuous map g from interval $[0,\ T]$ to interval $[0,\ 1]$ characterizing the speed profile along the path, i.e.

$$s=g(t),\ \ t\in[0,\ T],\ \ s\in[0,\ 1], \tag{5}$$

gives rise to a specific temporal trajectory defined by the function $r(g(t))$. As explained in [32], s is termed the progression rate, denoting the 0% to 100% completion of the whole path. An example is $g(t)=t/T$ corresponding to the case of a

constant speed profile. Let $\mathcal{G}_T$ be defined as the set of all such continuous maps

$$g : t \in [0,\ T] \mapsto s \in [0,\ 1],\ g(0) = 0 \text{ and } g(T) = 1, \quad (6)$$

then the set of all admissible trajectories along the same path $r(s)$ is given by

$$\mathcal{R}_r = \{\tilde{r} \in L_2^m[0,\ T] :\ \tilde{r}(t) = r(g(t)),\ t \in [0,\ T],\ g \in \mathcal{G}_T\}. \quad (7)$$

The path following design objective is to follow the given path at any admissible speed profile, i.e.

$$y = \tilde{r},\ \tilde{r} \in \mathcal{R}_r. \quad (8)$$

C. Path Following ILC Problem

In trajectory tracking tasks, the speed profile $g(t)$ is specified *a priori*, which is not the case in path following tasks. Given the infinite speed profile choices, significant design freedom can be unlocked in the control design of path following tasks. For example, the input u and output y can be stipulated to optimize an addition performance index $f(u, y)$ while maintaining the path following accuracy. The optimization of such performance index brings practical benefits. For example, if $f(u, y) = \|u\|_R^2$, the control effort can be minimized while performing the path following task, which significantly reduces manufacturing costs or overheads.

In addition, design constraints generally exist in practice for robotic path following tasks. For example, consider input constraints representing the allowed or safe input load range, an example of which is the set:

$$\Omega = \{u(t) \in \mathbb{R}^\ell : \lambda(t) \preceq u(t) \preceq \mu(t),\ t \in [0,\ T]\}. \quad (9)$$

where $\preceq$ denotes elementwise inequality.

Considering an extra input constraint, the path following ILC problem is then stated as follows:

Definition 1. The **Path Following ILC Problem** is to iteratively update an input sequence $\{u_k\}$ from an initial input $u_0 \in \Omega$ with the asymptotic property that the output y_k accurately passes through all points along the path r, i.e.

$$\lim_{k\to\infty} u_k = u^*, \quad (10)$$

where u^* is the solution of the optimization problem

$$\min_u f(u, y) \text{ subject to } y = Gu,\ y = \tilde{r},\ \tilde{r} \in \mathcal{R}_r,\ u \in \Omega. \quad (11)$$

III. Problem Reformulation and Solution

In this section the ILC problem stated in Definition 1 is reformulated into an equivalent form for ILC design, and exemplified through application to the class of piecewise linear paths. A Two Stage design framework is proposed to enable a tractable solution for the problem.

A. General Problem Reformulation

The set $\mathcal{R}_r$ can be equivalently defined using a projection operator P_r and a output constraint $y \in \Phi_r$ to yield a restatement of problem (11) that is suitable for ILC design as shown below.

Proposition 1. If there exist a projection operator $P_r : L_2^m[0,\ T] \mapsto L_2^{\hat{m}}[0,\ T]$ and a set Φ_r such that for any $\tilde{r} \in \mathcal{R}_r$ and $y \in L_2^m[0,\ T]$

$$P_r y = P_r \tilde{r} \text{ and } y \in \Phi_r \Leftrightarrow y \in \mathcal{R}_r, \quad (12)$$

the optimization problem (11) within the general path following ILC problem of Definition 1 is equivalent to

$$\begin{aligned} \min_u \quad & f(u, y) \\ \text{subject to} \quad & y = Gu,\ P_r y = P_r \tilde{r},\ y \in \Phi_r,\ u \in \Omega, \end{aligned} \quad (13)$$

where $\hat{m}$ is the dimension of the projected space.

Proof. Note that it is first assumed that $\Phi_r \subset \mathcal{R}_r$, otherwise the projection constraint $P_r y = P_r \tilde{r}$ may simply be omitted from both (12) and (13). Since the sufficient and necessary condition (12) holds, the general optimization problem (11) can be equivalently written as (13) by replacing $y = \tilde{r}$ and $\tilde{r} \in \mathcal{R}_r$ with $P_r y = P_r \tilde{r}$ and $y \in \Phi_r$. □

B. Characterization of Piecewise Linear Paths

In general, the projection operator P_r can be discontinuous and non-linear. However, there are classes of paths which give rise to a linear projection operator P_r that satisfies condition (12) in Proposition 1. An example of such a class is shown in the next definition.

Definition 2. The class of **Piecewise Linear Paths** is defined by the following function

$$r(s) = r_i + \left(\frac{s - s_i}{s_{i+1} - s_i}\right)(r_{i+1} - r_i), \quad (14)$$

for $s \in [s_i,\ s_{i+1}]$, $i = 0, \ldots, M$, with transition vertices

$$r_i = [r_i^1,\ r_i^2, \ldots,\ r_i^m]^\top,\ i = 0, \ldots, M + 1, \quad (15)$$

in Cartesian space and a set of spatial variables satisfying

$$0 = s_0 < s_1 < \ldots < s_M < s_{M+1} = 1. \quad (16)$$

The piecewise linear path following problem specifies that the i^{th} segment of the path is completed before the $(i + 1)^{th}$ segment is started. While staying within the exact line segment, the end-effector can move backwards to save energy. Such piecewise linear paths are common in manufacturing/construction, e.g. applications such as welding panels in ships and buildings, and typically cannot be modified by the designer. The following theorem characterizes the class of piecewise linear paths to obtain the corresponding operator P_r and set Φ_r required to satisfy Proposition 1.

Theorem 1. For the class of piecewise linear paths of Definition 2, operator P_r is defined by

$$(P_r y)(t) = P_i y(t),\ t \in [t_i,\ t_{i+1}],\ i = 0, ..., M, \quad (17)$$

and the output constraint set Φ_r be defined by

$$\Phi_r = \{y \in L_2^m[0,T] : a_i^\top r_i \leqslant a_i^\top y(t) \leqslant a_i^\top r_{i+1}, \quad t \in [t_i,\ t_{i+1}],\ i = 0,\ldots,M\}, \tag{18}$$

where $P_i \in R^{(m-1)\times m}$ is a full rank matrix satisfying

$$\operatorname{Ker} P_i = \operatorname{Im} a_i,\ i = 0,\ldots,M, \tag{19}$$

t_i is the time instant of the i^{th} transition vertex r_i, i.e.

$$y(t_i) = r_i,\ i = 0,...,M+1, \tag{20}$$

and $a_i = r_{i+1} - r_i$, $i = 0,...,M$. The ILC problem (13) is equivalent to

$$\begin{array}{ll} \min\limits_{u} & f(u,y) \\ \text{subject to} & y = Gu,\ u \in \Omega,\ y \in \Phi_r, \\ & P_i y(t) = P_i r_i, t \in [t_i, t_{i+1}], i = 0,...,M, \\ & y(t_i) = r_i,\ i = 0,...,M+1. \end{array} \tag{21}$$

Proof. See Appendix A. □

Although the general optimization problem (13) is not convex, the above theorem characterizes the class of piecewise linear paths by introducing a linear and continuous projection operator P_r and an output constraint $y \in \Phi_r$, which make the piecewise linear path following problem (21) convex. To simplify the notation in (21), the projection $P_i y(t) = P_i r_i$ and $y(t_i) = r_i$ in (21) can be equivalently expressed as

$$y^e = r^e, \tag{22}$$

where the mapping $\zeta \in L_2^m[0,\ T] \mapsto \zeta^e \in H$ is defined as

$$\zeta^e = [F\zeta,\ P\zeta]^\top, \tag{23}$$

and the reference $r \in L_2^m[0,\ T]$ is defined as $r(T) = r_{M+1}$, and $r(t) = r_i$, $t \in [t_i,\ t_{i+1})$, for $i = 0,\ldots,M$.

In (23), the operator F extracts the elements of ζ at the intermediate time instants t_i, $i = 1,\ldots,M$ as well as the start/terminal time instants t_0 and t_{M+1}, and is defined as

$$F\zeta = [\zeta(t_0),\ldots,\zeta(t_{M+1})]^\top,\ \zeta(t_i) \in \mathbb{R}^m. \tag{24}$$

The operator P extracts the elements of ζ along $[t_i,\ t_{i+1}]$, $i = 0,\ldots,M$, and is defined as

$$P\zeta = [(P\zeta)_0,\ldots,(P\zeta)_M]^\top, \tag{25}$$

where $(P\zeta)_i \in L_2^{m-1}[t_i,\ t_{i+1}]$ is defined as

$$(P\zeta)_i(t) = P_i\zeta(t),\ t \in [t_i,\ t_{i+1}].$$

The Hilbert space H is defined as

$$H = \underbrace{\mathbb{R}^m \times \cdots \times \mathbb{R}^m}_{\text{M+2}} \times L_2^{m-1}[t_0,\ t_1] \times \cdots \times L_2^{m-1}[t_M,\ t_{M+1}]$$

with the inner product and associated induced norm

$$\begin{aligned} \langle(\omega,\nu),(\mu,\lambda)\rangle_{\tilde{Q}} &= \sum_{i=0}^{M+1} \omega_i^\top Q_i \mu_i + \sum_{i=0}^{M} \int_{t_i}^{t_{i+1}} \nu_i^\top(t)\hat{Q}_i\lambda_i(t)\mathrm{d}t, \\ \|(\omega,\nu)\|_{\tilde{Q}} &= \sqrt{\langle(\omega,\ \nu),\ (\omega,\ \nu)\rangle_{\tilde{Q}}}, \end{aligned} \tag{26}$$

where $(\omega,\nu),\ (\mu,\lambda) \in H$ is defined by the following form

$$\begin{aligned} &\omega = [\omega_0,\ \omega_1,\ldots,\ \omega_{M+1}]^\top,\ \mu = [\mu_0,\ \mu_1,\ldots,\ \mu_{M+1}]^\top, \\ &\nu = [\nu_0,\ \nu_1,\ldots,\ \nu_M]^\top,\ \lambda = [\lambda_0,\ \lambda_1,\ldots,\ \lambda_M]^\top. \end{aligned}$$

Here $\omega_i,\ \mu_i \in \mathbb{R}^m$, $\nu_i,\ \lambda_i \in L_2^{m-1}[t_{i-1},\ t_i]$, $\tilde{Q}$ denotes the set $\{Q_0,\ldots,Q_{M+1},\ \hat{Q}_0,\ldots,\hat{Q}_M\}$, $Q_i \in \mathbb{S}_{++}^m$ and $\hat{Q}_i \in \mathbb{S}_{++}^{m-1}$.

The mapping (23) gives an "extended signal" ζ^e defined to be a "partial projected" signal $(P\zeta)(t)$ plus values $\zeta(t_i)$ at transition time instants. Using this notation, the "extended" system dynamics are modelled as

$$y_k^e = G_\Lambda^e u_k = (Gu_k)^e = [FGu_k,\ PGu_k]^\top, \tag{27}$$

where y^e is the extended system output. The subscript Λ of the extended system operator $G_\Lambda^e : L_2^\ell[0,\ T] \to H$ denotes the dependence on the unknown transition time instants defined as

$$\Lambda = [t_1,\ldots,\ t_M]^\top \in \Theta, \tag{28}$$

where Θ is the admissible set for Λ defined as

$$\Theta = \{\Lambda \in \mathbb{R}^M : 0 < t_1 < \ldots < t_M < 0\}. \tag{29}$$

Using the equivalent design constraint (22), the path following ILC problem stated in Definition 1 is reformulated for the class of piecewise linear paths as follows.

The **Piecewise Linear Path Following ILC Problem** is to iteratively update the transition time allocation Λ_k and the input signal u_k from initial values $\Lambda_0 \in \Theta$ and $u_0 \in \Omega$ with an asymptotic property such that the extended output y_k^e accurately tracks the extended reference r^e, i.e.

$$\lim_{k\to\infty}(\Lambda_k, u_k) = (\Lambda^*, u^*), \tag{30}$$

and (Λ^*, u^*) are solutions of the optimization problem

$$\begin{array}{ll} \min\limits_{\Lambda,u} & f(u,y) \\ \text{subject to} & y = Gu, \\ & y^e = r^e, \\ & \Lambda \in \Theta,\ u \in \Omega,\ y \in \Phi_r. \end{array} \tag{31}$$

C. A Two Stage Design Framework

The optimization problem (31) requires simultaneous selection of Λ and u to optimize the performance index $f(u,y)$ subject to the tracking requirements. To solve this problem, several extensions will be added to existing ILC frameworks reported in the literature, i.e. temporal optimization, and hard constraint handling.

The optimization problem (31) is equivalently stated as

$$\min_{\Lambda\in\Theta}\left\{\min_u f(u,y), \text{s.t. } y^e = r^e, y = Gu, u \in \Omega, y \in \Phi_r\right\}, \tag{32}$$

which first optimizes the problem over u and then over Λ. Define a function $\tilde{f}(\Lambda)$ as the minimum value of the inner optimization problem in (32), i.e.

$$\tilde{f}(\Lambda) = \min_u \{f(u,y), \text{s.t. } y^e = r^e, y = Gu, u \in \Omega, y \in \Phi_r\},$$

and denote a global minimizer for the inner optimization problem as $u_\infty(\Lambda) : \Theta \to L_2^\ell[0,\ T]$. Combining the above notations, the optimization problem (32) is equivalent to

$$\min_{\Lambda\in\Theta} f(u_\infty(\Lambda), Gu_\infty(\Lambda)). \tag{33}$$

To solve the original optimization problem (31), a Two Stage design framework inspired by the block coordinate descent methods of [33] is employed in this paper, which reformulates the problem into two stages. It alternates between optimizing the variables u and Λ using experimentally obtained data to provide a local minimizer of (33). This framework is described as follows:

- *Stage One:* Solve the inner optimization problem (32) with a fixed transition time allocation Λ, i.e.

$$\min_u\ f(u,y), \text{s.t. } y^e = r^e, y = Gu, u \in \Omega, y \in \Phi_r. \tag{34}$$

- *Stage Two:* Compute the solution of the resulting problem (33), i.e.

$$\min_{\Lambda\in\Theta}\{\tilde{f}(\Lambda) := f(u_\infty(\Lambda), Gu_\infty(\Lambda))\}. \tag{35}$$

where the signal $u_\infty(\Lambda)$ is the solution of problem (34).

To exemplify the design framework, the control effort $f(u,y) = \|u\|_R^2$ (which is a convex function) is chosen. Note that the projection operator P_r is linear for the class of piecewise linear paths, so there exists of a unique global minimizer for Stage One Problem (34), and hence (35) becomes $\min_{\Lambda\in\Theta} \|u_\infty(\Lambda)\|_R^2$.

IV. Comprehensive Solution of the Two Stage Design Framework

In this section, the solutions of the Two Stage framework are derived to yield an iterative algorithm with robust convergence performance as well as implementation instructions.

A. Stage One Solution

Stage One problem (34) involves transition position tracking, sub-interval tracking and hard output constraints, plus a minimum control effort requirement. It does not have a direct analytic solution, but can be solved using an ILC update approach. To solve this problem, the following notations are first introduced. A linear operator G_Λ^s is defined by

$$G_\Lambda^s u = [G_\Lambda^e u,\ Gu]^\top : L_2^\ell[0,\ T] \to \tilde{H} \tag{36}$$

and its Hilbert adjoint operator by G_Λ^{s*}. The Hilbert space $\tilde{H}$ is denoted by

$$\tilde{H} = \mathbb{R}^m \times \cdots \times \mathbb{R}^m \times L_2^{m-1}[t_0,\ t_1] \times \cdots \times L_2^{m-1}[t_M,\ t_{M+1}] \times L_2^m[0,\ T],$$

with inner product and associated induced norm naturally derived from (3) and (26). The projection operators P_Ω and P_{Φ_r} are defined as

$$P_\Omega(u) = \arg\inf_{z\in\Omega} \|z-u\|_R^2, \tag{37}$$

$$P_{\Phi_r}(y) = \arg\inf_{z\in\Phi_r} \|z-y\|_S^2, \tag{38}$$

which project the signals onto the sets Ω and Φ_r respectively. The linear operator $L : L_2^1[0,\ T] \to L_2^m[0,\ T]$ is defined as

$$\begin{aligned}(Lz)(t_i) &= [1,...,1]^\top z(t_i),\ i = 0,...,M+1,\\ (Lz)(t) &= P_i^\perp z(t),\ t \in (t_i,\ t_{i+1}),\ i = 0,...,M,\end{aligned} \tag{39}$$

where the column of $P_i^\perp \in R^{m\times 1}$ forms the orthogonal basis of the null space of P_i. Also, the error e_k^s is defined as

$$e_k^s = [e_k^e,\ e_k^c]^\top,\ e_k^e = r^e - y_k^e,\ e_k^c = \tilde{r}_k - y_k. \tag{40}$$

Using the above notation, the ILC update approach is illustrated in the next theorem.

Theorem 2. If Stage One problem (34) has a feasible solution, the system (1) is controllable and C is full row rank, the problem (34) is iteratively addressed by the ILC update

$$\tilde{u}_{k+1} = u_k + G_\Lambda^{s*}(I + G_\Lambda^s G_\Lambda^{s*})^{-1} e_k^s, \tag{41}$$

$$u_{k+1} = P_\Omega(\tilde{u}_{k+1}),\ \tilde{r}_{k+1} = P_{\Phi_r}(\tilde{y}_{k+1}), \tag{42}$$

with initial values $u_0 = 0$ and $\tilde{r}_0 \in \Phi_r$, and the solution is

$$u_\infty(\Lambda) = u^* + G^{-1}Lz^*, \tag{43}$$

where u^* is the converged value of the sequence $\{u_k\}$ generated by the ILC update (41)-(42), i.e. $\lim_{k\to\infty} u_k = u^*$, and the continuous signal $z^* \in L_2^1[0,\ T]$ is the solution of the optimization problem

$$\begin{aligned}&\min_z\ f(u,y)\\ &\text{s.t. } u = u^* + \Delta u,\ \Delta u = G^{-1}Lz,\ Fz = 0,\ y = Gu,\\ &\quad u \in \Omega,\ y \in \Phi_r.\end{aligned} \tag{44}$$

Proof. See Appendix B. □

Theorem 2 requires Stage One problem (34) to have a feasible solution, i.e. the path following task should be performed within the given system constraints. This scenario is guaranteed by task specifications as normal practice to ensure that the task is possible to perform. The assumption that the system (1) is controllable with C full row rank ensures the reference is in the range of G. It is not restrictive, since a controllable model can always be constructed for a given system and redundant output components can be removed to make C full row rank.

Remark 1. To solve Stage One problem (34), the problem (44) employs the converged input u^* and the one dimensional signal z with $z(t_i) = 0$ to remove the constraint $y^e = r^e$ in (34) by constructing the ℓ dimensional input signal $u = u^* + G^{-1}Lz$ without violating the tracking accuracy. Therefore, its search space of z is $L_2^1[0,\ T]$ instead of $L_2^\ell[0,\ T]$, which is the search space of u in (34). Since the system (1) is multi-input and multi-output, i.e. $\ell \geqslant 2$, (44) can cut down the search space of (34) significantly (e.g. for $\ell = 3$, the search space is reduced by 66.7%, and for $\ell = 4$,the search space is reduced by 75%) by at least 50%, which reduces the computational complexity.

Remark 2. The update (41) can be either implemented in a feedforward manner by direct calculation, or using a causal

feedback plus feedforward structure in [17] to further embed potential robust performance in practice. Meanwhile, the convex optimization problem (44) can be solved via various methods, such as the barrier method [34] and the primal-dual interior-point method [35].

The input constraint $u \in \Omega$ represents the allowed range for the input signal, and the output constraint $y \in \Phi_r$ in problem (34) has the effect of preventing potential overshoot, i.e. the end effector moving beyond the acceptable region. In the next corollary, an alternative "direct" ILC update is proposed to solve problem (34) when the constraints do not engage.

Corollary 1. If the constraints $u \in \Omega$ and $y \in \Phi_r$ do not engage, the system (1) is controllable and C is full row rank, Stage One problem (34) is solved by the ILC update

$$u_{k+1} = u_k + G_\Lambda^{e*}(I + G_\Lambda^e G_\Lambda^{e*})^{-1} e_k^e, \tag{45}$$

which provides an analytic global solution to (34) as follows:

$$u_\infty(\Lambda) = G_\Lambda^{e*}(G_\Lambda^e G_\Lambda^{e*})^{-1} r^e. \tag{46}$$

Proof. See Appendix C. □

Corollary 1 provides a significantly simpler ILC update to that in Theorem 2. The next proposition defines an explicit example of certain systems, e.g. industrial robots with velocity control, which are compatible with Corollary 1.

Proposition 2. If the system is a chain of integrators ($A = 0$) without the input constraint $u \in \Omega$ specified, the solution $u_\infty(\Lambda)$ satisfies $Gu_\infty(\Lambda) \in \Phi_r$, and the simplified ILC update (45) is directly applicable to Stage One problem (34).

Proof. See Appendix D. □

B. Stage Two Solution

Stage Two problem (35) is generally non-linear and non-convex with respect to the transition time allocation Λ, which leads to significant difficulties in obtaining an analytic solution. To solve this problem, the following lemma from [36] is needed.

Lemma 1. Let $\{\Lambda_k\}$ be a sequence generated by $\Lambda_{j+1} = P_\Theta(\Lambda_j - \gamma_j \nabla \tilde{f}(\Lambda_j))$ where Θ is a closed convex set and γ_j is chosen according to the generalized Armijo step size

$$\gamma_j = \alpha^\beta \gamma, \tag{47}$$

where β is the smallest non-negative integer such that

$$\tilde{f}(\Lambda_{j+1}) - \tilde{f}(\Lambda_j) \leqslant \sigma(\nabla \tilde{f}(\Lambda_j))^\top (\Lambda_{j+1} - \Lambda_j) \tag{48}$$

and $0 < \sigma < 1$, $0 < \alpha < 1$ and $\gamma > 0$ are constant scalars.

If there exists some $L > 0$ such that

$$\left|\nabla \tilde{f}(x) - \nabla \tilde{f}(y)\right| \leqslant L\,|x - y|\,,\ \forall x, y \in \Theta, \tag{49}$$

then every limit point of the sequence $\{\Lambda_k\}$ is a stationary point for problem (35).

Proof. The proof of this lemma is derived in [36]. □

Based on the above lemma, an iterative approach is developed in the next theorem to solve this problem.

Theorem 3. If the function $\tilde{f}(\Lambda)$ is twice continuously differentiable, the gradient projection update

$$\Lambda_{j+1} = P_\Theta\big(\Lambda_j - \gamma_j \nabla \tilde{f}(\Lambda_j)\big) \tag{50}$$

solves the Stage Two optimization problem (35) such that the sequence $\{\tilde{f}(\Lambda_j)\}$ converges downward to a limit η and every limit point of the sequence $\{\Lambda_j\}$ is a stationary point for the problem (35), i.e.

$$z = P_\Theta(z - \nabla \tilde{f}(z)). \tag{51}$$

Here $j \in \mathbb{N}$ denotes the gradient projection trial number, the gradient $\nabla \tilde{f}(\Lambda_j)$ is obtained using experimental measured data, P_Θ denotes the gradient projection optimal solution

$$P_\Theta(x) = \arg\min_z \{\|z - x\| : z \in \Theta\}, \tag{52}$$

and $\gamma_j > 0$ is the generalized Armijo step size defined in (47).

Proof. Note that all assumptions in Lemma 1 are satisfied, so that the gradient projection update (50) can be applied to the problem considered in this paper. According to Lemma 1, the generalized Armijo step size guarantees the convergence of the gradient projection method to a stationary point. □

Since the function $\tilde{f}(\Lambda)$ is bounded below and Θ is a compact set, the (global) minimum of the optimization problem exists. In addition, being a stationary point satisfying (51) is a necessary condition of being a (possibly local) minimum point. Therefore, the converged stationary point must be the minimum if there exists only one stationary point in problem (35). As a consequence, the above algorithm is guaranteed to converge to the global minimum solution in this case.

In general, the gradient within (50) does not admit an analytic realization, and can be computed using a computationally efficient estimation

$$\left.\frac{\partial \tilde{f}}{\partial t_i}\right|_{\Lambda_j} = \frac{\tilde{f}(\Lambda_j^{i+}) - \tilde{f}(\Lambda_j^{i-})}{2\Delta T}, \tag{53}$$

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 sufficiently small. The values of $\tilde{f}(\Lambda_j^{i+})$ and $\tilde{f}(\Lambda_j^{i-})$ can be obtained using the solution (41) via experimental implementation or simulation.

Furthermore, if the system constraints do not engage, the problem (35) can be expressed analytically as shown in the following corollary.

Corollary 2. In the absence of the system constraints $u \in \Omega$ and $y \in \Phi_r$, Stage Two problem (35) is transformed as

$$\min_{\Lambda \in \Theta} \|u_\infty(\Lambda)\|_R^2 = \min_{\Lambda \in \Theta}\ \left\langle r^e,\ (G_\Lambda^e G_\Lambda^{e*})^{-1} r^e \right\rangle_{\tilde{Q}}. \tag{54}$$

Proof. See Appendix E. □

According to Corollary 2, in the absence of the system constraints, the gradient within (50) can be computed analytically

by performing differentiation of the cost function in (54) and calculating its partial derivatives.

Remark 3. The choice of initial allocation Λ_0 affects the convergence performance of (50). It can be chosen arbitrarily from the admissible set Θ if no information is available. Alternatively, it can be chosen using the methods, e.g. low resolution initial allocation, proposed in [27], which approximate the optimal allocation.

C. A Comprehensive Implementational Algorithm

Algorithm 1 is obtained by combining the aforementioned implementations of the design framework, which are summarized in the following steps:

- Choose the constraint sets Θ and Ω as well as the weighting parameters R, S, Q_i and $\hat{Q}_i$ according to the application; specify the required tracking accuracy and performance improvement threshold as the sufficiently small positive scalars ϵ and δ, for example, $\epsilon = 0.01$ and $\delta = 0.001$; choose a suitable initial transition time allocation Λ_0, by $t_i = iT/M$, $i = 1, ..., M$, and an extended reference r^e; set the gradient projection trial number j to 0.
- Then, apply the generalized ILC update law (41)-(42) together with (43), experimentally to the current input signal u_k and error e_k^s until the accuracy ϵ is reached (Stage One solution), which computes an optimal input signal $u_\infty^{ex}(\Lambda_0)$ with respect to Λ_0. The solution minimizes the optimal performance index $\tilde{f}(\Lambda_0)$ in practice, whose value is recorded.
- After that, apply the gradient projection procedure (50) experimentally to the transition time allocation Λ_j to update its value to Λ_{j+1} until the convergence condition in Step 7 is satisfied. Meanwhile, the same Stage One solution as explained in the above bullet point is performed to compute the optimal input signal $u_\infty^{ex}(\Lambda_j)$ at each gradient projection trial, and the values of $\tilde{f}(\Lambda_j)$ are recorded.
- The final values of Λ_j and $u_\infty^{ex}(\Lambda_j)$ are taken as the output of the algorithm.

Note that the experimental tests in Step 2 and 6 of the algorithm (i.e. generalized ILC update) can be replaced with simulations, however this will significantly reduce robustness to model uncertainty and this replacement is not suggested in practice. The gradient $\nabla\tilde{f}(\Lambda_j)$ in (50) is estimated by (53), in which the values of $\tilde{f}(\Lambda_j^{i+})$ and $\tilde{f}(\Lambda_j^{i-})$ using the Stage One solution. In addition, the generalized Armijo step size (47) in (50) is determined by (48).

D. Robustness Analysis

In practice, the nominal system model G will not exactly match the real plant model $\hat{G}$. Therefore, it is necessary to perform robustness analysis to verify the convergence performance of Algorithm 1 in practice. Note that Stage One update (41)-(42) in Step 2 and 6 of Algorithm 1 involves the participation of measured output trajectory y_k in practical implementation. Meanwhile, the experimental obtained input

Algorithm 1 Iterative learning control for path following tasks

Input: Initial transition time allocation Λ_0, system operator G, extended reference r^e, admissible set Θ, input constraint set Ω, weighting matrices R, S, Q_i and $\hat{Q}_i$ and parameter values σ, β and γ.

Output: Optimal solutions Λ_{opt} and u_{opt}

1: **Initialization:** Gradient projection trial number $j = 0$
2: Perform (41)-(42) with $\Lambda = \Lambda_0$ experimentally until $\|e_k^s\| < \epsilon$, and then solve the problem (44) to obtain $u_\infty(\Lambda_0)$ using (43); record $u_\infty(\Lambda_0)$ and $\tilde{f}(\Lambda_0)$.
3: **while**
4: Apply (50) with (53) using $r_i = G_i u_\infty(\Lambda_j)$.
5: Set $j = j + 1$.
6: Perform (41)-(42) with $\Lambda = \Lambda_j$ experimentally until $\|e_k^s\| < \epsilon$, and then solve the problem (44) to obtain $u_\infty(\Lambda_j)$ using (43); record $u_\infty(\Lambda_j)$ and $\tilde{f}(\Lambda_j)$.
7: **if** $\left|\tilde{f}(\Lambda_j) - \tilde{f}(\Lambda_{j-1})\right| < \delta\left|\tilde{f}(\Lambda_{j-1})\right|$, **break**.
8: **end while**
9: **return** $\Lambda_{opt} = \Lambda_j$ and $u_{opt} = u_\infty^{ex}(\Lambda_j)$.

data $u_\infty(\Lambda_j)$ is applied in Step 4 to generate the function of the desired performance index. Therefore, this algorithm learns by exploiting real data to reduce the effect of modelling inaccuracy. Robustness analysis is shown in the next theorem.

Theorem 4. Suppose the system constraints $u \in \Omega$ and $y \in \Phi_r$ do not engage, the extended operator G_Λ^e is right invertible and the system operator $\hat{G}$ is denoted as

$$\hat{G}_k = (I + \Delta)G, \tag{55}$$

where the unknown operator $\Delta : L_2^m[0,\ T] \to L_2^m[0,\ T]$ represents the model uncertainty. If operator Δ satisfies

$$\left\| I - \begin{bmatrix} F \\ P \end{bmatrix} \Delta G G_\Lambda^{e*} \right\| \leqslant 1, \tag{56}$$

then the sequence $\{e_k^e\}$ generated by the ILC update (45), monotonically converges to zero, i.e.

$$\left\| e_{k+1}^e \right\| \leqslant \rho \left\| e_k^e \right\|, \ \forall\ k \geqslant 0, \tag{57}$$

where $\rho < 1$ is the spectral radius of $(I + G_\Lambda^e G_\Lambda^{e*})^{-1}$.

If the operator $G_\Lambda^{e*}(\hat{G}_\Lambda^e G_\Lambda^{e*})^{-1}$ is upper bounded as

$$\left\| G_\Lambda^{e*}(\hat{G}_\Lambda^e G_\Lambda^{e*})^{-1} r^e \right\| / \left\| r^e \right\| \leqslant \overline{\sigma},\ \forall\ r^e \in H, \tag{58}$$

the function $\tilde{f}(\Lambda)$ generated by the gradient projection update (50) is bounded above as

$$\tilde{f}(\Lambda) \leqslant \overline{\sigma}^2 \left\| r^e \right\|^2, \tag{59}$$

where the upper bound $\overline{\sigma}$ is a positive scalar.

Proof. See Appendix F. □

From the above theorem, the Stage One ILC update still achieves perfect path following even with the existence of model uncertainty, while (56) is satisfied. Moreover, the sequence $\{\tilde{f}(\Lambda_j)\}$ generated by the Stage Two gradient projection update is proved to be bounded above. Therefore,

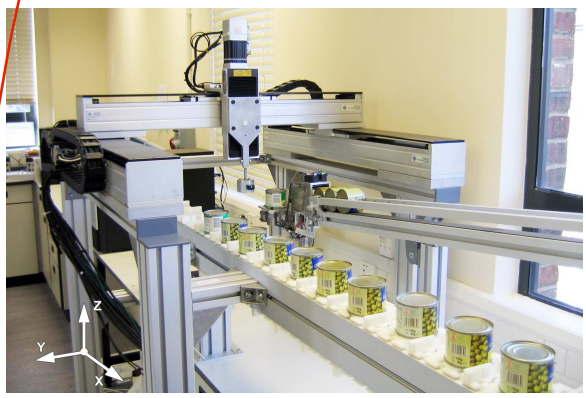

Fig. 2. A multi-axis gantry robot test platform.

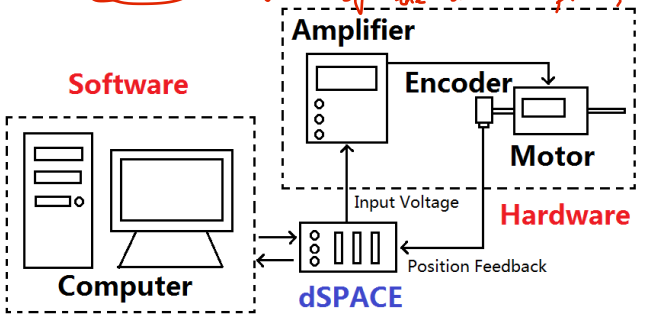

Fig. 3. Overall structure of the gantry robot test platform.

Algorithm 1 has attractive robust performance against model uncertainty.

Furthermore, the sequence $\{\tilde{f}(\Lambda_j)\}$ has strong robust convergence properties for certain cases. This means that the proposed algorithm then solves the optimization problem (31) with high tracking accuracy as well as optimizing an additional performance index for these special cases. See the following corollary for an exemplary case.

Corollary 3. If the model uncertainty is in a constant scalar form, i.e. $\Delta = \kappa$, where κ is a constant scalar, then the sequence $\{\tilde{f}(\Lambda_j)\}$ generated by the gradient projection update (50) strictly monotonically converges downward to a limit η, i.e.

$$\tilde{f}(\Lambda_{j+1}) \leqslant \tilde{f}(\Lambda_j), \ \forall\ j \geqslant 0. \tag{60}$$

Proof. See Appendix G. □

Corollary 3 considers a special case of the model uncertainty, and the actual control effort is proportional to the numerically computed value $\tilde{f}(\Lambda)$ at each gradient projection trial. Therefore, the monotonic convergence of control effort to a local minimum value is indeed demonstrated in this corollary.

Remark 4. Although the true optimal solution is guaranteed for this special case of model uncertainty in a scalar form, the tracking error norm converges to zero and the performance index can be only shown bounded for the more general cases of model uncertainties.

V. EXPERIMENTAL VERIFICATION

In this section, a three-axis gantry robot is employed as a test platform to validate the performance of Algorithm 1 and demonstrate its effectiveness in practice.

A. Test Platform Specifications

The multi-axis gantry robot shown in Figure 2 has three axes, which are perpendicular to each other. The x-axis and the y-axis move in the horizontal plane and are driven by linear brush-less DC motors. The z-axis with vertical moving direction is placed on the top of the other two axes, and comprises a linear ball-screw stage driven by a rotary brushless DC motor. For each axis, a BNC channel sends the input voltage signal to the motor amplifier, and the displacement of each axis is measured as the output trajectory signal by an optical incremental encoder installed on the same axis. The hybrid motion of the three axes gives rise to a path of the gantry robot end-effector in 3D space.

This test platform consists of a dSPACE DS1103 microcontroller, Aerotech model BA10 linear amplifiers, and Renishaw RGH22 linear encoders with a resolution of 1 μm for x-axis and y-axis, and Aerotech rotary encoder for z-axis with a resolution of 0.5 μm. See Figure 3 for the overall structure of the gantry robot test platform.

B. Task Design Objective

The control design objective is to use all three axes ($m = 3$) to perform a piecewise linear path following task composed of five line segments ($M = 5$) with

$$r_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, r_1 = \begin{bmatrix} 0.00476 \\ 0.00345 \\ 0.008 \end{bmatrix}, r_2 = \begin{bmatrix} 0.00294 \\ 0.00905 \\ 0.01 \end{bmatrix},$$
$$r_3 = \begin{bmatrix} -0.00294 \\ 0.00905 \\ 0.01 \end{bmatrix}, r_4 = \begin{bmatrix} -0.00476 \\ 0.00345 \\ 0.008 \end{bmatrix}, r_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

as shown in Figure 6 (the blue path), during the given tracking time horizon $[0,\ 2]$, i.e. $T = 2s$. In addition, the minimum time spent between two adjecent transition vertices is $0.01s$.

For implementation, the weighting matrices are chosen as $Q_i = q_i I$, $\hat{Q}_i = \hat{q}_i I$ and $R = \hat{r} I$ where q_i, $\hat{q}_i$ and $\hat{r}$ are positive scalars. Furthermore, appropriate weighting matrices are chosen according to the theoretical predictions in [37] to balance convergence speed and robust performance, i.e. $q_i/\hat{r} = 30,000$ and $\hat{q}_i/\hat{r} = 3,000,000$.

C. Experimental Performance of Algorithm 1

The robust performance of Algorithm 1 is tested with significant model uncertainty to replicate an industrial environment. Assume that only three inaccurate nominal models (for x-axis, y-axis and z-axis) are available for control design as

$$G_x(s) = \frac{0.04}{s},\ G_y(s) = \frac{0.04}{s} \text{ and } G_z(s) = \frac{0.04}{s}. \tag{61}$$

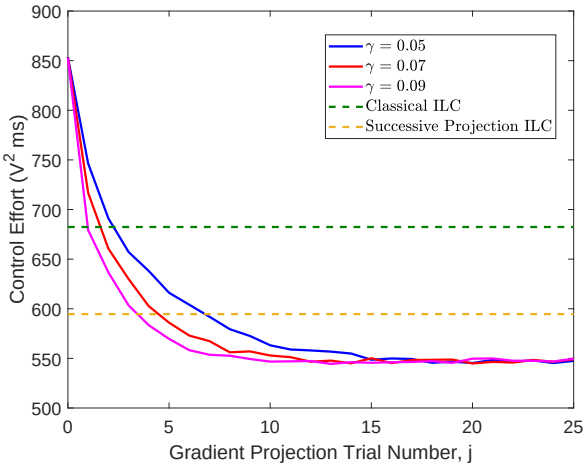

Fig. 4. A comparison of control effort along the gradient projection trials for Algorithm 1, classical ILC and successive projection ILC.

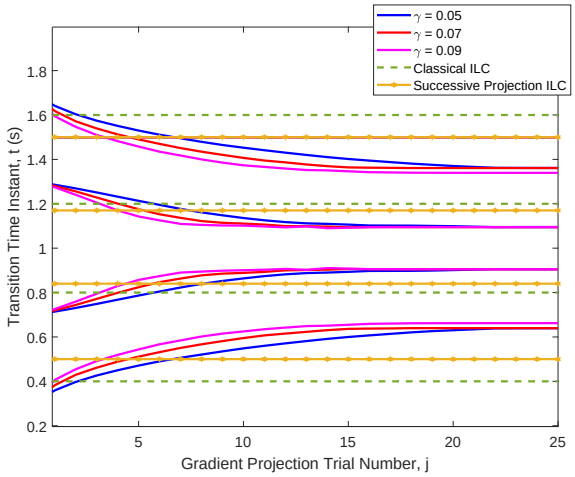

Fig. 5. A comparison of transition time allocation along the gradient projection trials for Algorithm 1, classical ILC and successive projection ILC.

Very accurate high order transfer function models of the same gantry robot are given in [38] and allow the the model uncertainty operator Δ to then be computed. This satisfies the condition (56) of Theorem 4. Algorithm 1 is applied with the initial transition time allocation $\Lambda_0 = [0.3, 0.7, 1.3, 1.7]^\top$, and the gradient projection (50) chooses the parameters values to be $\sigma = 0.1$, $\alpha = 0.5$ and $\gamma = 0.05, 0.07, 0.09$ respectively. The results of control effort and transition time instants over 25 gradient projection trials are shown in Figure 4 and 5. Note that the attempt number of Stage One solution required to obtain the generalized Armijo step size satisfying (48) increases from 0 to 7 along the gradient projection trials, and it requires a total number of 8 attempts of Stage One solution to compute the values of $\tilde{f}(\Lambda_j^{i+})$ and $\tilde{f}(\Lambda_j^{i-})$ to hence estimate $\nabla\tilde{f}(\Lambda_j)$ using (53) at each trial.

For comparison, the performance of classical norm optimal ILC algorithm in [37] is also evaluated using a pre-planned speed profile with the same path. As a normal practice, the pre-planned trajectory is defined by a fixed transition time $[0.4, 0.8, 1.2, 1.6]^\top$ and the moving speed of the end-effector along each line segments is considered as a constant value. The final case implemented is to solve optimization problem (31) offline using the nominal plant model (61), yielding the solution plan $[0.50, 0.84, 1.17, 1.50]^\top$, and then to use this as a fixed reference in the standard successive projection ILC algorithm of [30]. The corresponding results of classical ILC and successive projection ILC are plotted in the same figures.

In Figure 4, the blue, red, and magenta curves denote the minimum control effort $\tilde{f}(\Lambda_k)$ at each gradient projection trial for $\gamma = 0.05, 0.07, 0.09$ respectively. Using the proposed algorithm, the control effort of the path following task converges over the trials, and the converged minimum control effort of 546.9 is independent of the algorithm's parameter choice of γ. Compared to the standard practice of using a pre-planned speed profile with a required control effort of 682.4 (shown as the green dashed line in the figure), the proposed algorithm achieves approximately 20% control effort reduction. In addition, it outperforms successive projection ILC (shown as the yellow dashed line in the figure with value 594.6) by around 8% in terms of control effort reduction. These features demonstrate the advantage of the proposed algorithm over other ILC algorithms. This advantage will increase further as the degree of model uncertainty increases. For further information, the transition time instants for these approaches are shown in Figure 5, where the blue, red, and magenta curves denote the transition time allocation Λ_j at each trial for $\gamma = 0.05, 0.07, 0.09$ respectively with the final converged value $\Lambda_{opt} = [0.66, 0.90, 1.09, 1.36]^\top$ and the green and yellow dashed lines represent the constant transition time instants of classical ILC and successive projection ILC.

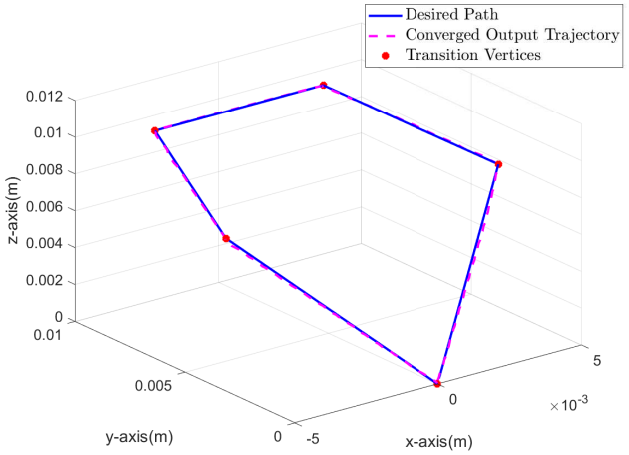

Fig. 6. Desired path and converged output trajectory at the final gradient projection trial for robust test.

Note that the above control effort reduction is achieved without compromising the path following accuracy. The experimental converged output trajectory for Λ_{opt} is shown in Figure 6 with the red dots denoting the transition vertices, and it is clear that the converged output trajectory accurately follows the given piecewise linear path. The final converged input and output trajectories of each axis are shown in Figure 8

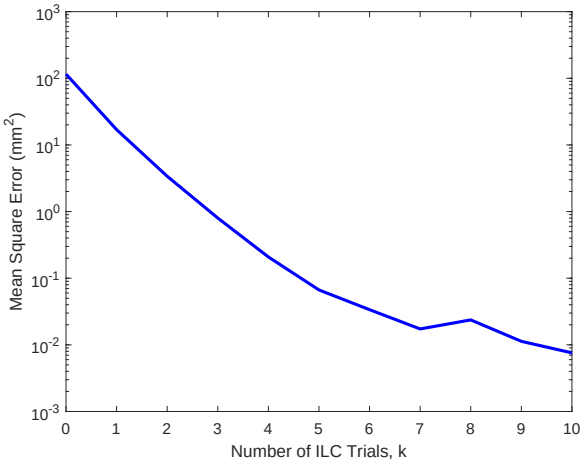

Fig. 7. The mean square path following error along the ILC trials at the final gradient projection trial in the robustness test.

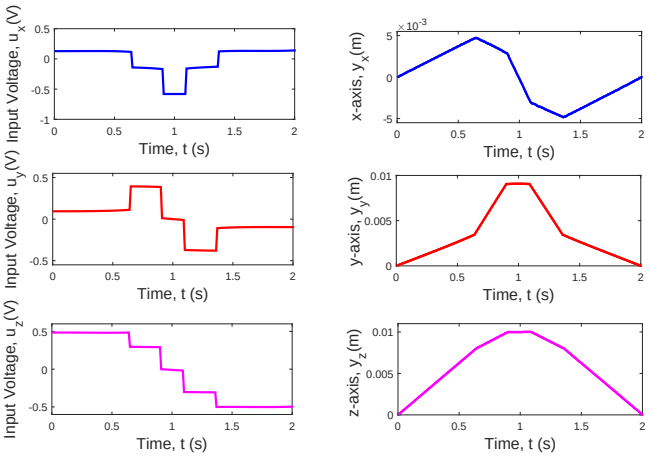

Fig. 8. Converged input signal (left) and output trajectory (right) at the final gradient projection trial in the robustness test.

for more information. To examine convergence in more detail, the mean squared path following errors at the ILC trials for the final gradient projection trial are plotted in Figure 7 for the $\gamma = 0.09$ case. This figure shows that it requires 10 ILC trials for the path following error to decline to a value below the accuracy value $\epsilon = 10^{-2} mm^2$. In the experiments, it generally requires $8 - 12$ ILC trials to achieve the desired accuracy. It is also worth mentioning that the above results are obtained using relatively inaccurate nominal models, illustrating that the proposed algorithm has a certain degree of robustness against disturbance and model uncertainty, which is appealing in practice.

Furthermore, experiments with different values of Q_i, $\hat{Q}_i$ and R have also been carried out using the parameters $\sigma = 0.1$, $\beta = 0.5$ and $\gamma = 0.07$. The corresponding results are summarized in Table I. From this table, the optimal transition time allocation obtained by Algorithm 1 all converge, and the minimum control effort results of these cases are all approximately 20% less than the value obtained from classical ILC. Experiments using other initial transition allocation, e.g. the low resolution one in [27], have also been performed and produced similar performance.

TABLE I
SUMMARY OF EXPERIMENTAL RESULTS WITH DIFFERENT Q_i, $\hat{Q}_i$ AND R.

	Λ_{opt}	$\tilde{f}(\Lambda_{opt})$	Reduction
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 3,000,000$	$[0.65, 0.90, 1.09, 1.36]^\top$	545.1	20.12 %
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 5,000,000$	$[0.65, 0.90, 1.09, 1.36]^\top$	546.8	19.87 %
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 8,000,000$	$[0.65, 0.90, 1.09, 1.35]^\top$	547.3	19.80 %
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 10,000,000$	$[0.66, 0.90, 1.09, 1.35]^\top$	546.5	19.92 %
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 3,000,000$	$[0.65, 0.90, 1.09, 1.35]^\top$	545.9	20.00 %
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 3,000,000$	$[0.65, 0.90, 1.09, 1.36]^\top$	544.7	20.18 %
$q_i/\hat{r} = 30,000$, $\hat{q}_i/\hat{r} = 3,000,000$	$[0.66, 0.90, 1.09, 1.36]^\top$	546.6	19.90 %

VI. CONCLUSION AND FUTURE WORK

This paper develops a general ILC design framework for repeated path following tasks. It not only guarantees the path following accuracy, but also minimizes an additional performance index of interest by fully exploiting the control design freedom. The general path following problem is reformulated into an equivalent optimization problem using a projection operator. The class of piecewise linear paths is characterized for the reformulated problem setup, and a Two Stage design framework is proposed to solve this non-trivial problem. This framework yields an iterative algorithm with robust convergence properties, which incorporates an ILC update and a gradient projection update. This algorithm is verified experimentally on a gantry robot test platform to demonstrate its significant practical benefits of control effort reduction and robustness against model uncertainty.

Although experimental verification of the proposed algorithm has revealed practical efficacy, this paper focuses on piecewise linear paths and minimum control effort. In general, the optimization problem (31) can be extended to handle other types of paths, e.g. piecewise arc paths. In this case, the projection operator P_r becomes non-linear, and non-linear ILC approaches using gradient or Newton method can be applied to solve the problem. However, a global solution might not be guaranteed due to the non-linearity. In principle, the ILC design framework can be implement to optimize an alternative performance index, such as the path following time. Meanwhile, a rigorous analysis of the robust convergence properties of the proposed algorithm with respect to iteration varying noise or disturbances is needed. These aspects constitute the focus of future research and will be reported separately.

APPENDIX

A. Proof of Theorem 1

If $y \in \mathcal{R}_r$, there exists $y(t) = r(g(t))$ and

$$y(t_i) = r(g(t_i)) = r(s_i) = r_i, \ i = 0, \ldots, M+1, \qquad (62)$$

by definition. Due to the tracking order, it follows that

$$0 = t_0 < t_1 < \ldots < t_{M+1} = T. \tag{63}$$

Therefore, $s_i \leqslant g(t) \leqslant s_{i+1}$ for $t \in [t_i,\ t_{i+1}]$, and all the points $y(t)$ on this sub-interval belong to the line segment set

$$\mathcal{R}_i = \{y \in \mathbb{R}^m : y = r_{i+1} - \gamma(r_{i+1} - r_i),\ \gamma \in [0,\ 1]\}, \tag{64}$$

and satisfy the relationship $P_i y(t) = P_i r_i$. It follows that

$$P_i(r_{i+1} - \gamma(r_{i+1} - r_i)) = P_i r_i, \tag{65}$$

which yields $P_i a_i = 0$, and further equates to the projection operator P_i in (19). Also, as $\mathcal{R}_i$ is a line segment set, it follows that a hard output constraint exists

$$a_i^\top r_i \leqslant a_i^\top y(t) \leqslant a_i^\top r_{i+1},\ t \in [t_i,\ t_{i+1}], \tag{66}$$

to prevent overshoot, which yields the set Φ_r defined in (18). The above derivations imply the operator P_r defined in (17) and the output constraint $y \in \Phi_r$.

If P_r is defined by (17) and $y \in \Phi_r$, it follows that

$$y(t_i) = r(g(t_i)),\ i = 0, \ldots, M+1. \tag{67}$$

Since $(P_r y)(t) = P_i y(t) = P_i r_i$, $t \in [t_i,\ t_{i+1}]$ and $y \in \Phi_r$ are satisfied, the above steps can be reversed to obtain

$$y(t) \in \mathcal{R}_i,\ t \in [t_i,\ t_{i+1}],\ i = 0, \ldots, M. \tag{68}$$

Therefore, (67) and (68) combine to give

$$y(t) = r(g(t)),\ t \in [0,\ T], \tag{69}$$

for $g(t_i) = s_i$ and $s_i \leqslant g(t) \leqslant s_{i+1}$, $t \in [t_i,\ t_{i+1}]$, which implies $y \in \mathcal{R}_r$.

Therefore, the ILC problem (13) is equivalent to the form (21) by substituting the definition (17) of P_r and the extra output constraint $y \in \Phi_r$ to replace $P_r y = P_r \tilde{r}$.

B. Proof of Theorem 2

With the assumptions made in the theorem, the path following design objective of the problem (34) is within the range of the ILC problem discussed in [30]. Therefore, the ILC algorithm proposed in that paper can be applied with minor modifications to achieve the design objective with desirable convergence properties by setting the appropriate parameters, which give rise to the ILC update law (41)-(42). However, it only guarantees y_k^e to converge to r^e, i.e. $\lim_{k\to\infty} y_k^e = y^{e*} = r^r$, rather than an optimal performance index.

The system is invertible since the system (1) is controllable. Hence, it follows from the definition of the operator L in (39) and the signal z is continuous that the constraints $\Delta u = G^{-1} L z$ and $F z = 0$ give rise to

$$\begin{aligned} &(G\Delta u)(t_i) = 0,\ i = 0, ..., M+1, \\ &P_i(G\Delta u)(t) = 0,\ t \in (t_i,\ t_{i+1}),\ i = 0, ..., M. \end{aligned} \tag{70}$$

The above equations are denoted as $G_\Lambda^e \Delta u = 0$. Therefore, the problem (44) is equivalent to

$$\begin{aligned} &\min_u\ f(u, y) \\ &\text{s.t. } y^e = y^{e*},\ y = Gu,\ u \in \Omega,\ y \in \Phi_r, \end{aligned} \tag{71}$$

which is identical to Stage One problem (34). Therefore, its solution (43) added an offset input Δu to the input signal u^* obtained by the ILC update to optimize the performance index $f(u, y)$ without violating the perfect tracking error of $r^e - y^{e*} = 0$ or the system constraints $u^* \in \Omega,\ y^* \in \Phi_r$, which solves Stage One problem (34).

C. Proof of Corollary 1

In the absence of the constraints, the ILC update (41)-(42) collapses to the ILC update (45). According to [39], the ILC update (45) ultimately converges to provide the optimal solution (46) to problem (34) without the constraints.

D. Proof of Proposition 2

The proof by contradiction is used to confirm that the overshoot does not happen when $A = 0$, so it is assumed that the output $Gu_\infty(\Lambda)$ has overshoot at the i^{th} line segment. It is necessary (but not sufficient) that there must exist a time $t_i^* \in (t_{i-1},\ t_i)$ such that

$$(Gu^*)(t_i^*) = (Gu^*)(t_i) = r_i. \tag{72}$$

Since the input constraint is not specified, an input

$$u^*(t) = \begin{cases} 0, & t \in [t_i^*,\ t_i], \\ (u_\infty(\Lambda))(t), & \text{else}, \end{cases} \tag{73}$$

can be always constructed on $[0,\ T]$ based on the assumption of overshoot to provide a lower control effort than $u_\infty(\Lambda)$, i.e. $\|u^*\|_R^2 < \|u_\infty(\Lambda)\|_R^2$. From (73), there exists

$$(Gu^*)(t) = (Gu_\infty(\Lambda))(t),\ t \in [0,\ t_i^*]. \tag{74}$$

Since $A = 0$, it follows that

$$\begin{aligned} (Gu^*)(t) &= \int_0^{t_i^*} CBu^*(s)\mathrm{d}s + \int_{t_i^*}^{t} CBu^*(s)\mathrm{d}s \\ &= \int_0^{t_i^*} CB(u_\infty(\Lambda))(s)\mathrm{d}s \\ &= (Gu_\infty(\Lambda))(t_i^*) = r_i,\ \forall\ t \in [t_i^*,\ t_i], \end{aligned} \tag{75}$$

$$\begin{aligned} (Gu^*)(t) &= \int_0^{t_i} CBu^*(s)\mathrm{d}s + \int_{t_i}^{t} CBu^*(s)\mathrm{d}s \\ &= (Gu_\infty(\Lambda))(t_i) + \int_{t_i}^{t} CB(u_\infty(\Lambda))(s)\mathrm{d}s \\ &= (Gu_\infty(\Lambda))(t),\ \forall\ t \in [t_i,\ T]. \end{aligned} \tag{76}$$

Then, it follows that

$$\begin{aligned} &(Gu^*)(t_i) = r_i,\ i = 0, ..., M, \\ &(P_i G u^*)(t) = P_i r_i,\ t \in [t_{i-1},\ t_i],\ i = 1, ..., M. \end{aligned} \tag{77}$$

Hence the input u^* not only provides a lower control effort than the optimal value $u_\infty(\Lambda)$, but also satisfies the tracking requirement $y^e = r^e$. This obviously contradicts the assumption that the optimal input $u_\infty(\Lambda)$ generates overshoot. Therefore, the simplified ILC update (45) provides a solution satisfying the hard constraint $y \in \Phi_r$.

E. Proof of Theorem 2

Since the system constraints do not engage, the analytic solution (46) is available. Substitute (46) into the problem (35) and use the property of adjoint operator to yield

$$\begin{aligned}\min_{\Lambda\in\Theta}\|u_\infty(\Lambda)\|_R^2 &= \min_{\Lambda\in\Theta}\ \langle (u_\infty(\Lambda),\ u_\infty(\Lambda))\rangle_R \\ &= \min_{\Lambda\in\Theta}\ \left\langle (G_\Lambda^{e*}(G_\Lambda^e G_\Lambda^{e*})^{-1}r^e, G_\Lambda^{e*}(G_\Lambda^e G_\Lambda^{e*})^{-1}r^e\right\rangle_R \\ &= \min_{\Lambda\in\Theta}\ \left\langle G_\Lambda^e G_\Lambda^{e*}(G_\Lambda^e G_\Lambda^{e*})^{-1}r^e, (G_\Lambda^e G_\Lambda^{e*})^{-1}r^e\right\rangle_{\tilde{Q}} \\ &= \min_{\Lambda\in\Theta}\ \left\langle r^e,\ (G_\Lambda^e G_\Lambda^{e*})^{-1}r^e\right\rangle_{\tilde{Q}}, \end{aligned} \tag{78}$$

which completes the proof.

F. Proof of Theorem 4

Since the real plant $\hat{G}$ is in a multiplicative form (55), the measured extended error e_k^e is

$$e_k^e = r^e - \begin{bmatrix} F\hat{G} \\ P\hat{G}\end{bmatrix} u_k = r^e - \begin{bmatrix} F(I+\Delta)G \\ P(I+\Delta)G\end{bmatrix} u_k. \tag{79}$$

Based on equation (79) and update (45), it follows that

$$\begin{aligned} e_{k+1}^e &= r^e - \begin{bmatrix} F(I+\Delta)G \\ P(I+\Delta)G\end{bmatrix}(u_k + \underbrace{G_\Lambda^{e*}(I+G_\Lambda^e G_\Lambda^{e*})^{-1}}_{L} e_k^e) \\ &= \underbrace{(I - \begin{bmatrix} F \\ P\end{bmatrix}\Delta G G_\Lambda^{e*})(I+G_\Lambda^e G_\Lambda^{e*})^{-1}}_{E} e_k^e. \end{aligned} \tag{80}$$

Since G_Λ^e is right invertible, it is clear that

$$\left\|(I+G_\Lambda^e G_\Lambda^{e*})^{-1}\right\| \leqslant \rho < 1, \tag{81}$$

which with the condition (56) give rise to the monotonic convergence condition (57) as follows:

$$\begin{aligned}\left\|e_{k+1}^e\right\| &\leqslant \left\|I - \begin{bmatrix} F \\ P\end{bmatrix}\Delta G G_\Lambda^{e*}\right\| \left\|(I+G_\Lambda^e G_\Lambda^{e*})^{-1}\right\| \left\|e_k^e\right\| \\ &\leqslant \left\|(I+G_\Lambda^e G_\Lambda^{e*})^{-1}\right\| \left\|e_k^e\right\| \leqslant \rho \left\|e_k^e\right\| < \left\|e_k^e\right\|. \end{aligned} \tag{82}$$

The error norm is lower bounded by zero, so the sequence $\{e_k^e\}$ converges to zero and the task design objective is achieved as

$$r^e = \hat{G}_\Lambda^e u_\infty^{ex}(\Lambda) \tag{83}$$

for each gradient projection trial.

It follows from (45) and (80) that the input signal can be represented as

$$\begin{aligned} u_1 &= u_0 + Le_0^e \\ u_2 &= u_1 + Le_1^e = u_0 + Le_0^e + LEe_0^e \\ &\cdots \\ u_k &= u_0 + L\sum_{i=0}^{k-1} E^i e_0^e. \end{aligned} \tag{84}$$

Since E satisfies $\|E\| < 1$, there exists

$$\lim_{k\to\infty} u_k = u_0 + L(I-E)^{-1}e_0^e. \tag{85}$$

Note that $u_0 = 0$, hence the solution is

$$u_\infty^{ex}(\Lambda) = L(I-E)^{-1}r^e = G_\Lambda^{e*}(\hat{G}_\Lambda^e G_\Lambda^{e*})^{-1}r^e. \tag{86}$$

which together with the condition (58) yields the upper bound of $\tilde{f}(\Lambda)$ in (59).

G. Proof of Corollary 3

It follows from (83), $\hat{r}_\Lambda^e = G_\Lambda^e u_\infty^{ex}(\Lambda)$ and $\hat{G} = (1+\kappa)G$ such that the relationship between the constant value r^e and the numerical computed value $\hat{r}_\Lambda^e$ is obtained as

$$r^e = \hat{G}_\Lambda^e u_\infty^{ex}(\Lambda) = (1+\kappa)G_\Lambda^e u_\infty^{ex}(\Lambda) = (1+\kappa)\hat{r}_\Lambda^e, \tag{87}$$

which means that the value $\hat{r}_\Lambda^e$ at each gradient projection trial is proportional to the value r^e. Therefore, the performance index $\left\langle \hat{r}_\Lambda^e,\ (G_\Lambda^e G_\Lambda^{e*})^{-1}\hat{r}_\Lambda^e\right\rangle$ optimized at each trial is proportional to $\tilde{f}(\Lambda)$ as

$$\left\langle \hat{r}_\Lambda^e,\ (G_\Lambda^e G_\Lambda^{e*})^{-1}\hat{r}_\Lambda^e\right\rangle = \frac{1}{(1+\kappa)^2}\tilde{f}(\Lambda), \tag{88}$$

with respects to a constant ratio. In this sense, the sequence $\{\tilde{f}(\Lambda_j)\}$ monotonically converges to a limit η. According to the condition (48), the equal sign in (60) only holds when the sequence converges. Therefore, the sequence strictly monotonically decreases.

REFERENCES

[1] K. D. Do, Z. P. Jiang, and J. Pan, "Robust adaptive path following of underactuated ships," *Automatica*, vol. 40, pp. 929–944, 2004.

[2] A. P. Aguiar and J. P. Hespanha, "Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1362–1379, 2007.

[3] D. R. Nelson, D. B. Barber, T. W. McLain, and R. W. Beard, "Vector field path following for miniature air vehicles," *IEEE Transactions on Robotics*, vol. 23, no. 3, pp. 519–529, 2007.

[4] T. I. Fossen, K. Y. Pettersen, and R. Galeazzi, "Line-of-sight path following for dubins paths with adaptive sideslip compensation of drift forces," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 820–827, 2015.

[5] D. Verscheure, B. Demeulenaere, J. Swevers, J. de Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2318–2327, 2009.

[6] T. Lippa and S. Boyd, "Minimum-time speed optimisation over a fixed path," *International Journal of Control*, vol. 87, no. 6, pp. 1297–1311, 2014.

[7] Q.-C. Pham, S. Caron, P. Lertkultanon, and Y. Nakamura, "Admissible velocity propagation: Beyond quasi-static path planning for high-dimensional robots," *International Journal of Robotics Research*, vol. 36, no. 1, pp. 44–67, 2017.

[8] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.

[9] 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.

[10] X. Jin, "Fault-tolerant iterative learning control for mobile robots non-repetitive trajectory tracking with output constraints," *Automatica*, vol. 94, pp. 63–71, 2018.

[11] T. Oomen and C. R. Rojas, "Sparse iterative learning control with application to a wafer stage: Achieving performance, resource efficiency, and task flexibility," *Mechatronics*, vol. 47, pp. 134–147, 2017.

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

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

[14] 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.

[15] 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.

[16] J.-X. Xu, "A survey on iterative learning control for nonlinear systems," *International Journal of Control*, vol. 84, no. 7, pp. 1275–1294, 2011.

[17] D. Owens, *Iterative Learning Control: An Optimization Paradigm*. Springer, 2015.
[18] R. W. Longman, "Iterative learning control and repetitive control for engineering practice," *International Journal of Control*, vol. 73, no. 10, pp. 930–954, 2000.
[19] A. Tayebi and M. B. Zaremba, "Robust iterative learning control design is straightforward for uncertain LTI systems satisfying the robust performance condition," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 101–106, 2003.
[20] K. L. Moore, Y. Q. Chen, and V. Bahl, "Monotonically convergent iterative learning control for linear discrete-time systems," *Automatica*, vol. 41, no. 9, pp. 1529–1537, 2005.
[21] Y. Q. Chen and K. L. Moore, "A practical iterative learning path-following control of an omni-directional vehicle," *Asian Journal of Control*, vol. 4, no. 1, pp. 90–98, 2002.
[22] K. L. Moore, M. Ghosh, and Y. Q. Chen, "Spatial-based iterative learning control for motion control applications," *Meccanica*, vol. 42, pp. 167–175, 2007.
[23] S. K. Sahoo, S. K. Panda, and J.-X. Xu, "Application of spatial iterative learning control for direct torque control of switched reluctance motor drive," in *IEEE Power Engineering Society General Meeting*, Tampa, FL, 2007, pp. 1–7.
[24] D. J. Hoelzle and K. L. Barton, "On spatial iterative learning control via 2-D convolution: Stability analysis and computational efficiency," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 4, pp. 1504–1512, 2016.
[25] P. Janssens, W. V. Loock, G. Pipeleers, F. Debrouwere, and J. Swevers, "Iterative learning control for optimal path following problems," in *52nd IEEE Conference on Decision and Control*, Florence, Italy, 2013, pp. 6670 – 6675.
[26] A. Steinhauser and J. Swevers, "An efficient iterative learning approach to time-optimal path tracking for industrial robots," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 5200–5207, 2018.
[27] Y. Chen, B. Chu, and C. T. Freeman, "Point-to-point iterative learning control with optimal tracking time allocation," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1685–1698, 2018.
[28] ——, "A coordinate descent approach to optimal tracking time allocation in point-to-point ILC," *Mechatronics*, vol. 59, pp. 25–34, 2019.
[29] ——, "Generalized iterative learning control using successive projection: Algorithm, convergence and experimental verification," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2079–2091, 2020.
[30] Y. Chen, B. Chu, C. T. Freeman, and Y. Liu, "Generalized iterative learning control with mixed system constraints: A gantry robot based verification," *Control Engineering Practice*, vol. 95, 2020, 104260.
[31] Y. Chen, B. Chu, and C. T. Freeman, "Spatial path tracking using iterative learning control," in *55th IEEE Conference on Decision and Control*, Las Vegas, US, 2016, pp. 7189–7194.
[32] M. Davidson and V. Bahl, "The scalar ϵ-controller: A spatial path tracking approach for ODV, ackerman, and differentially-steered autonomous wheeled mobile robots," in *the 2001 IEEE International Conference on Robotics and Automation*, 2001, pp. 175–180.
[33] A. Beck and L. Tetruashvili, "On the convergence of block coordinate descent type methods," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2037–2060, 2013.
[34] A. Ben-Tal and M. Zibulevsky, "Penalty/barrier multiplier methods for convex programming problems," *SIAM Journal on Optimization*, vol. 7, no. 2, pp. 347–366, 1997.
[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
[36] D. P. Bertsekas, "On the Goldstein - Levitin - Polyak gradient projection method," *IEEE Transactions on Automatic Control*, vol. 21, no. 2, pp. 174–184, 1976.
[37] 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.
[38] J. D. Ratcliffe, "Iterative learning control implemented on a multi-axis system," Ph.D. dissertation, University of Southampton, Sothampton, June 2005.
[39] D. H. Owens, C. T. Freeman, and B. Chu, "Generalized norm optimal iterative learning control with intermediate point and sub-interval tracking," *International Journal of Automation and Computing*, vol. 12, no. 3, pp. 243–253, 2015.