Iterative Learning Control for Robotic Path Following with Trial-Varying Motion Profiles

Yiyang Chen, Bing Chu* and Christopher T. Freeman

***Abstract*—Iterative learning control (ILC) aims to maximize performance of systems performing repeated tracking tasks. However, in most existing applications, the motion profile is inherently specified *a priori*, which has restricted both its application range and scope of performance improvement. For example, the typical repeated path following task in robotics only requires a spatial path profile rather than a temporal trajectory profile, for which most existing ILC designs are unsuitable. To handle this requirement, this paper extends the ILC task description by relaxing this postulate to enable a trial-varying motion profile and formulate an ILC path following problem with system constraints. Under this extended problem setup, a spatial ILC algorithm is proposed with efficient implementation and robust convergence analysis, which updates the input signal and motion profile at the end of each trial to reduce the tracking error. This algorithm is implemented experimentally on a gantry robot test platform to verify performance, practical feasibility and reliability. Comparisons with other control methods are also made to clarify its advantages, such as error reduction, control effort reduction and constraint handling.**

***Index Terms*—iterative learning control, path following, gantry robot**

I. Introduction

ILC is a technique formulated to improve the performance of robotic tasks that repeatedly execute a task over multiple trials. ~~It~~ employs the input signal u_k and tracking error e_k of the current trial k to update the input signal u_{k+1} for the next trial. Based on the recorded information of past trials, ILC has been shown to significantly reduce the repeated errors caused by model mismatch, and for many system classes guarantees error convergence to zero after a sufficient number of trials. In addition, the ILC implementation procedure is relatively simple and straightforward, since it does not require change to internal parameters or controller structure. Moreover, it does not require a high computational load due to its off-line input update procedure. Because of these benefits, ILC has been applied to various robotic systems, e.g. stroke rehabilitation [1]–[3], robotic manipulator [4]–[6], inkjet printer [7], gantry robot [8] and motor system [9]. See [10] for a detailed overview.

This work was supported by the Excellent Young Scholar Program of Soochow University, the ZZU-Southampton Collaborative Research Project under Grant 16306/01, National Natural Science Foundation of China under Grant 62103293, 61773232, Natural Science Foundation of Jiangsu Province under Grant BK20210709 and the Royal Society International Exchanges Award under Grant IE161369.

Y. Chen is with the School of Mechanical and Electrical Engineering, Soochow University, Suzhou, 215137, China (yychen90@suda.edu.cn).

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

*The corresponding author.

The vast majority of ILC work, e.g. [11]–[16], considers the concept of trajectory tracking as the design objective, i.e. the output trajectory has an *a priori* motion profile specified over a finite time interval $[0,\ T]$, where T is the duration of the task. However, the concept of path following is the principal design objective within a wide range of robotic tasks, such as manufacturing assembling [17], virtual fixture [18] and packaging. Compared to trajectory tracking, path following only requires the robotic end-effector to follow a given path defined in space without any temporal constraints. Therefore, the motion profile is unlocked as a free variable to release substantial control design freedom. The work in [19]–[21] exploited this design freedom to improve the robustness in vehicle steering problems based on the feedback of measured information. This design freedom is further considered in parallel work [22]–[25] to develop path planning strategies aiming at minimizing the total robotic path following time. However, the tasks described in the above research were not repeated, which prevented the application of ILC to further improve their path following performance. The research conducted in [26] utilized ILC to handle a time-optimal path tracking task, but employed a different ILC setting using a system model update law. The recent work in [27], [28] exploited the trial-varying initial conditions as well as multiple time-delays to design robust ILC algorithm for tracking tasks with disturbances, modeling errors, and various uncertainties.

A number of ILC attempts have been made to relax the assumption of an *a priori* specified motion trajectory. The first work in [29] proposed an ILC update law to increase the path following accuracy at corner points. This update law measured the closest distance to the path for each point on the output trajectory, and employed this information to decide whether to switch on and off the motors of a two-axis gimbal system. Subsequent work in [30] addressed the torque ripple reduction problem for switched reluctance motors with spatially defined physical variables, which updated the input voltage at each rotor position using the spatial torque error. Moreover, several well-established ILC algorithms were reformulated in [31] with spatial coordinates via 2D convolution reconstruction for systems with spatial steady state coupling output, such as additive manufacturing. Although the above work developed ILC algorithms to handle robotic tasks without a unique motion profile, these algorithms were ad hoc, applicable only to specific robotic applications and only a minority focused on path following tasks. Also, none exploited the control design freedom released by the freedom of a trial-varying motion profile to achieve practical benefits other than the task accuracy. An exception is the authors' previous work in

[32]–[36], which proposed several ILC algorithms to minimize control effort while maintaining high path following accuracy. However, these algorithms were restricted to limited classes of tasks, i.e. point-to-point or piecewise linear path following tasks.

The contribution of this work in relation to the existing solutions in the literature is listed as follows:

- *ILC Task Extension (Section II).* This paper extends the classical ILC task description in [11]–[16] to exploit the design freedom of a trial-varying motion profile to improve task performance in terms of path following accuracy.
- *General Problem Formulation (Section II).* This problem formulation is the first attempt to solve general spatial path following problem with constraints on micro scale manufacturing tasks, which outperforms the existing spatial ILC frameworks in [29]–[36] in terms of generality.
- *A Comprehensive ILC Algorithm (Section III).* A powerful and comprehensive spatial ILC algorithm is proposed with convergence analysis to update the input signal and motion profile at the end of each trial, which solves the proposed general path following problem.
- *Experimental Verification (Section IV).* The practical performance of this algorithm is verified on a gantry robot test platform, and comparisons with other control methods are made to illustrate its advantages to solve the constrained path following problems, which cannot be addressed by classical norm optimal ILC in [37].

The notation used in this paper is standard: $\mathbb{N}$ is the set of non-negative integers; $\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ denote the sets of n dimensional real vectors and $n\times m$ real matrices respectively; $\mathbb{S}^n_{++}$ is the set of all $n\times n$ real positive definite matrices; $L_2^\ell[a,\ b]$ denotes the space of $\mathbb{R}^\ell$ valued Lebesgue square-integrable sequences defined on an interval $[a,\ b]$; $\langle x, y\rangle$ 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 $\mathbb{Y}$, respectively; any signal with the symbol $\hat{}$ denotes a physically measured signal rather than a numerically computed signal using a nominal system model. Other notations will be introduced as needed.

II. Problem Formulation

In this section, the system dynamics are introduced, and the path following task is described using the concepts of a path and a motion profile. The ILC task description is then extended to formulate an ILC path following problem with a trial-varying motion profile as well as embedding system constraints.

A. System Dynamics

This paper considers an ℓ-input, m-output continuous linear time-invariant system in 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 $k\in\mathbb{N}$ denotes the trial number; $t\in[0,\ T]$ is the time index with $T<\infty$ being the trial length; $x_k(t)\in\mathbb{R}^n$, $u_k(t)\in\mathbb{R}^\ell$ and $y_k(t)\in\mathbb{R}^m$ are the state, input and output on the k^{th} trial 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. $x_k(0)=x_0$. This system has an equivalent operator form

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

Since the signal d is constant with respect to the identical initial state x_0, it can be absorbed by the reference trajectory $r(t)$ without loss of generality by assuming $x_0=0$ to yield $d(t)=0$.

The input signal u_k and output signal y_k belongs to the input and output Hilbert spaces $L_2^\ell[0,\ T]$ and $L_2^m[0,\ T]$ defined with inner products and associated induced norms

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

$$\langle x,\ y\rangle_Q = \int_0^T x^\top(t)Qy(t)\mathrm{d}t,\ \ \|y\|_Q = \sqrt{\langle y,\ y\rangle_Q}, \tag{4}$$

in which $R\in\mathbb{S}^\ell_{++}$ and $Q\in\mathbb{S}^m_{++}$; $G: L_2^\ell[0,\ T]\to L_2^m[0,\ T]$ is the system operator and $d\in L_2^m[0,\ T]$ denotes the effect of initial conditions with respective forms

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

Note that G has an adjoint operator G^* defined by

$$\langle u,\ G^*y\rangle_R = \langle Gu,\ y\rangle_Q, \tag{6}$$

which will be used later in algorithm development.

B. Path Following Task Requirements

It is now necessary to introduce the concepts of a path and a motion profile, where the former is defined as follows:

Definition 1. A path is a sequence of points in output space $\mathbb{R}^m$ defined by a continuous function

$$\tilde{r}: s\in[0,\ 1]\mapsto \tilde{r}(s)\in\mathbb{R}^m, \tag{7}$$

mapping spatial coordinate s to output point $\tilde{r}(s)$.

The path profile is defined by the *a priori* function $\tilde{r}$ with initial and terminal positions, $\tilde{r}(0)$ and $\tilde{r}(1)$, and independent of any temporal information. To determine how fast the robotic end-effector moves along the given path, a continuous motion profile

$$\theta: t\in[0,\ T]\mapsto s\in[0,\ 1] \tag{8}$$

is introduced to map between time index t and spatial coordinate s, i.e. $s=\theta(t)$. Based on the above definitions of path profile and motion profile, a specific reference profile $r\in L_2^m[0,\ T]$ can be written as follows:

$$r(t) = \tilde{r}(\theta(t)),\ t\in[0,\ T]. \tag{9}$$

Therefore, the path following task is described as: to choose an input signal u such that the output trajectory $y=Gu$ accurately follows the reference profile r defined by (9), i.e.

$$y = Gu = r,\ r(t) = \tilde{r}(\theta(t)). \tag{10}$$

In practice, system constraints have to be considered in the control design procedure to guarantee the obtained solution is allowed and implementable. An input saturation constraint

$$\Omega = \{u \in l_2^\ell[0,\ T] : |u(t)| \preceq \mathcal{M}(t), t \in [0,\ T]\} \tag{11}$$

is taken into account to denote the restrictions on system input limit for an exemplary case. In addition, the backward motion of the system is prohibited to give the constraint $\dot{\theta}(t) \geqslant 0$, and the extra constraints $\theta(0) = 0$ and $\theta(N) = 1$ are used to guarantee the task starting at the initial position and ending at the terminal position. Meanwhile, considering the servo speed and acceleration constraints, a motion profile constraint is obtained as follows:

$$\begin{aligned}\Theta =&\{\theta \in L_2^1[0,\ T] :\ 0 \leqslant \dot{\theta}(t) \leqslant v_{max},\ \left|\ddot{\theta}(t)\right| \leqslant a_{max},\\ &\dot{\theta}(0) = \dot{\theta}(T) = 0\},\end{aligned} \tag{12}$$

where v_{max} and a_{max} are maximum velocity and acceleration values respectively.

According to the task description and the system constraints, a path following problem is defined as below.

Definition 2. The **Path Following Problem** aims at choosing an appropriate input signal u in the input constraint set Ω to reduce the path following error

$$e = r - y \tag{13}$$

to zero, for $r(t)$ defined in (9) and any admissible θ in the motion profile set Θ defined in (12).

C. ILC Path Following Problem

To improve the performance of repeated path following tasks in robotics, the classical ILC task description is extended in this paper to exploit the choice of motion profile as a free variable, which gives rise to the following problem.

Definition 3. The **ILC Path Following Problem** iteratively updates u_k and θ_k using

$$(u_{k+1}, \theta_{k+1}) = \mathcal{F}(u_k, \theta_k, e_k), \tag{14}$$

such that the two variables converge, the system constraints are met and the path following error $e_k = r_k - y_k$ converges to zero, i.e.

$$\lim_{k\to\infty} u_k = u^* \in \Omega,\ \lim_{k\to\infty} \theta_k = \theta^* \in \Theta,\ \lim_{k\to\infty} e_k = 0, \tag{15}$$

where $r_k(t) = \tilde{r}(\theta_k(t))$ is the reference profile at trial k.

Since the motion profile is a free variable in the above problem setup, it is also incorporated into the ILC update law (14) as an updated term. Note that the system constraint sets Ω and Θ denote the physical limits discussed in Section II-B. To achieve the design objectives of the extended problem, a spatial ILC algorithm is proposed in Section III.

Remark 1. The extended ILC problem setup in Definition 3 is more general than that of classical ILC in terms of constraint handling. It is capable of changing input signals and motion profiles to alternative values to satisfy the system constraints, while the unique solutions of classical ILC might not be admissible. See the comparisons in Section IV for more detail.

III. A Spatial ILC Algorithm

In this section, the design objectives of the ILC path following problem are listed, and a spatial ILC algorithm is proposed with robust convergence performance and implementation instructions, which solves the problem in Definition 3.

A. ILC Algorithm Description

To achieve the design objectives in Definition 3, the authors consider the norm optimal ILC update law proposed in [37], which iteratively updates the input signal u_k by minimizing a cost function consisting of the error and the input signal change. This ILC update law is extended in this paper by embedding the motion profile θ as a free variable and incorporating the system constraints as well. To handle the problem, the following lemma is needed.

Lemma 1. Let S_1 and S_2 be two closed convex sets in a Hilbert space X. Define the projection operators $P_{S_1}(\cdot)$ and $P_{S_2}(\cdot)$ as

$$\begin{aligned}P_{S_1}(x) &= \arg\min_{\hat{x}\in S_1} \|\hat{x} - x\|_X^2,\\ P_{S_2}(x) &= \arg\min_{\hat{x}\in S_2} \|\hat{x} - x\|_X^2,\end{aligned}$$

where $\|\cdot\|$ is the induced norm in X. If $S_1 \cap S_2 \neq \varnothing$, the following convergence condition is satisfied as

$$\|x^{**} - x\|_X^2 \leqslant \|x^* - x\|_X^2,\ \forall\ x \in S_1 \cap S_2, \tag{16}$$

where $x^* \in X$ and x^{**} is computed as $\tilde{x} = P_{S_1}(x^*)$ and $x^{**} = P_{S_2}(\tilde{x})$.

Proof. The proof is derived in [38]. □

Using the condition (16) in above lemma, the extended ILC update law and its convergence property are hence described in the following theorem.

Theorem 1. If the system (1) is controllable and C has full row rank, define the ILC update law as follows: with initial values $u_0 \in \Omega$ and $\theta_0 \in \Theta$, at each trial, an ILC update law updates the input signal using

$$\tilde{u}_{k+1} = u_k + G^*(I + GG^*)^{-1}(r_k - Gu_k) \tag{17}$$

followed by the projection

$$u_{k+1} = P_\Omega(\tilde{u}_{k+1}) = \arg\min_{u\in\Omega} \|u - \tilde{u}_{k+1}\|_R, \tag{18}$$

and then updates the motion profile using

$$\theta_{k+1} = \arg\min_{\theta\in\Theta} \|r - y_{k+1}\|_Q, s.t.\ r(t) = \tilde{r}(\theta(t)). \tag{19}$$

If $S_1 \cap S_2 \neq \varnothing$, the monotonic conditions hold as follows:

$$\|r_k - y_{k+1}\|_Y \leqslant \|e_k\|_Y, \tag{20}$$

$$\|e_{k+1}\|_Q \leqslant \|r_k - y_{k+1}\|_Q, \tag{21}$$

where $G^\dagger = (G^*G)^{-1}G^*$ is the pseudo inverse operator of G, $S_1 = \{(y,u) \in H : y = Gu\}$, $S_2 = \{(y,u) \in H : y = r_k, u \in \Omega\}$, H is a Hilbert space defined as $H = L_2^m[0,\ T] \times L_2^\ell[0,\ T]$ and

$$\|y\|_Y = \sqrt{\int_0^T [(G^\dagger y(t))^\top R G^\dagger y(t) + y^\top(t) Q y(t)]\mathrm{d}t}. \tag{22}$$

Proof. See Appendix A. □

The above theorem states how the extended ILC update law modifies the input signal u_{k+1} and motion profile θ_{k+1} at the end of each trial respectively, while preserving their constrained requirements $u \in \Omega$ and $\theta \in \Theta$. Moreover, it appealing properties on monotonic convergence and smaller error upper bound have been shown in the next corollary at the specific case without input constraint.

Corollary 1. If the input constraint does not engage, the path following error monotonically converges, i.e.

$$\|e_{k+1}\|_Q \leqslant \rho \|e_k\|_Q , \tag{23}$$

and the upper bound of error norm ε_k at each trial is equal or smaller than the value ε_k^{noilc} of norm optimal ILC using the same values of Q and R, i.e.

$$\varepsilon_k \leqslant \varepsilon_k^{noilc}, \tag{24}$$

where $0 < \rho < 1$ is the spectral radius of $(I + GG^*)^{-1}$.

Proof. See Appendix B. □

Based on this update law, a spatial ILC algorithm (Algorithm 1) is then proposed to update the input signal and motion profile at the end of each trial. In this algorithm, u_0 and θ_0 are appropriate initial input signal and motion profile. At the end of each trial, the input signal u_{k+1} is updated by (39) with weighting matrices Q and R, and followed by a projection mechanism. Then, the updated input signal is sent to the plant to obtain the corresponding measured output trajectory $\hat{y}_{k+1}$. After that, the motion profile θ_{k+1} is updated by performing another projection mechanism to yield the measured path following error $\hat{e}_k = r_k - \hat{y}_k$ in practice. The training procedure terminates when the error norm becomes smaller than ϵ, which denotes a sufficiently small positive scalar specifying the accepted practical accuracy level for the given task. Moreover, the computational complexity of this algorithm is $3n^2$.

Algorithm 1 Spatial ILC Algorithm

Input: System operator G, path profile $\tilde{r}$, initial input signal $u_0 \in \Omega$, initial motion profile $\theta_0 \in \Theta$, weighting matrices Q and R

Output: Optimal input signal u^* and motion profile θ^*

1: **initialization:** Trial number $k = 0$
2: Send the initial input signal u_0 to the robot, measure the initial output trajectory $\hat{y}_0$, and obtain the initial error $\hat{e}_0$.
3: Record the initial input signal u_0 and motion profile θ_0.
4: **while not** $\|\hat{e}_k\| < \epsilon$ **do**
5: Update the input signal u_{k+1} by solving (39) and performing projection $u_{k+1} = P_\Omega(\tilde{u}_{k+1})$.
6: Set $k = k + 1$ to perform the next ILC trial.
7: Send the input signal u_k to the robot, measure the output trajectory $\hat{y}_k$.
8: Update the motion profile θ_k by performing projection (19) and obtain the error $\hat{e}_k$.
9: Record the input signal u_k and motion profile θ_k.
10: **end while**
11: **return** $u^* = u_k$ and $\theta^* = \theta_k$

B. Robust Convergence Performance

In this paper, a multiplicative form of model uncertainty is considered as

$$\hat{G} = (I + \Delta)G, \tag{25}$$

where the operator $\Delta : l_2^m[0,\ N] \to l_2^m[0,\ N]$ denotes the model uncertainty. Therefore, the measured output trajectory $\hat{y}_k = \hat{G}u_k$ differs from the the numerically computed output trajectory $y_k = Gu_k$. Although the ILC update procedure in Algorithm 1 is similar to that in norm optimal ILC, the employment of the trial-varying motion profile and system constraints leads to extra effort in the robust performance analysis as shown in the following theorem.

Theorem 2. If the input constraint does not engage and

$$\|I - \Delta GG^*\| \leqslant 1, \tag{26}$$

the measured path following error sequence $\{\hat{e}_k\}_{k\geqslant 0}$ of Algorithm 1 monotonically converges to zero, i.e.

$$\|\hat{e}_{k+1}\|_Q \leqslant \rho \|\hat{e}_k\|_Q , \tag{27}$$

where $0 < \rho < 1$ is the spectral radius of $(I + GG^*)^{-1}$.

Proof. See Appendix C. □

Theorem 2 provides theoretical support for the robust monotonic convergence property of Algorithm 1 with the existence of model uncertainty. This appealing feature is of much practical concern for robotic path following task, as it guarantees the monotonic reduction of the path following error to zero even there is a certain level of model uncertainty.

C. Implementation Instructions

The ILC update law (17) can be directly applied as a feedforward implementation. Moreover, the alternative feedback plus feedforward implementation in [37] is suggested to embed real-time measured system state within the ILC update. Since this method has instantaneous interaction with system plant to learn state information during the task, it can further improve the robust performance of the algorithm.

The operator P_Ω projects the unconstrained input $\tilde{u}_{k+1}$ into the input constraint set Ω. This step is straightforward as the input constraint set Ω is usually pointwise in practice. For example, if the input constraint set Ω is represented as the saturation form (11), the solution of $u_{k+1} = P_\Omega(\tilde{u}_{k+1})$ is given by

$$u_{k+1}(t) = \begin{cases} \mathcal{M}(t), & \tilde{u}_{k+1}(t) \succ \mathcal{M}(t), \\ \tilde{u}_{k+1}(t), & -\mathcal{M}(t) \preceq \tilde{u}_{k+1}(t) \preceq \mathcal{M}(t), \\ -\mathcal{M}(t), & \tilde{u}_{k+1}(t) \prec -\mathcal{M}(t), \end{cases} \tag{28}$$

for $0 \leqslant t \leqslant T$. The projection (19) chooses an element from the motion profile constraint set Θ by solving the problem

$$\begin{aligned} &\min_{\theta} \|r - \hat{y}_{k+1}\|_Q^2, \\ &\text{subject to } \theta \in \Theta,\ r(t) = \tilde{r}(\theta(t)). \end{aligned} \tag{29}$$

The cost function of problem (29) is convex, the set Θ is also convex by its definition in (12). However, the function

$$r(t) = \tilde{r}(\theta(t)), \tag{30}$$

may be non-linear since a general path is usually not composed of straight line segments. In this sense, this problem is a large scale non-convex optimization problem, as the motion profile needs to be discretized at a small sample time into a high dimensional vector. To reduce the computational load, the following assumption is considered to add an additional constraint to problem (29).

Assumption 1. The motion profile θ in problem (29) belongs to the set

$$\Theta_k = \{\theta \in L_2^1[0,\ T] :\ \theta_k(t-\Delta) \leqslant \theta(t) \leqslant \theta_k(t+\Delta)\}, \tag{31}$$

where Δ denotes a small time interval, e.g. $\Delta = 0.01 \cdot T$.

The extra constraint $\theta \in \Theta_k$ reduces the search space of the variable θ, so each element $\theta(t)$ is then optimized over the local sub-intervals $[\theta_k(t-\Delta),\ \theta_k(t+\Delta)]$. Since the lengths of these sub-intervals are much smaller than the whole spatial interval $[0,\ 1]$, the path profile $\tilde{r}$ can be approximated as a line segment from $r_k(t-\Delta)$ to $r_k(t+\Delta)$ via local linearization within these relatively small sub-intervals. Therefore, the constraint (30) on this sub-interval can be approximated by the linear function

$$r(t) = r_k(t) + \frac{\theta(t) - \theta_k(t-\Delta)}{\theta_k(t+\Delta) - \theta_k(t-\Delta)}(r_k(t+\Delta) - r_k(t-\Delta)). \tag{32}$$

Using the above linearized approximation, the problem (29) is reformulated into the following convex optimization problem

$$\begin{aligned} &\min_{\theta} \quad \|r - \hat{y}_k\|_Q^2 \\ &\text{s.t.} \quad \theta \in \Theta \cap \Theta_k,\ r(1) = \tilde{r}(0),\ r(T) = \tilde{r}(1), \\ &r(t) = r_k(t) + \frac{\theta(t) - \theta_k(t-\Delta)}{\theta_k(t+\Delta) - \theta_k(t-\Delta)}(r_k(t+\Delta) - r_k(t-\Delta)), \end{aligned} \tag{33}$$

with a global optimal solution.

Remark 2. Assumption 1 does not affect the convergence performance of Algorithm 1, since the proofs in Theorem 1 and 2 still hold with respect to the combined motion profile constraint $\Theta \cap \Theta_k$.

Remark 3. The proposed algorithm has three steps during an identical trial, which are the input signal update step (17), the input signal projection step (18) and the motion profile update step (19). As explained in the above text, the first two steps are relatively straightforward to implement, but the computational burden mainly lies on the last step, as it needs to solve a high dimensional convex optimization problem (33). This can be efficiently solved using standard solvers and parsers in **MATLAB**, such as **SeDuMi** [39] and **YALMIP** [40].

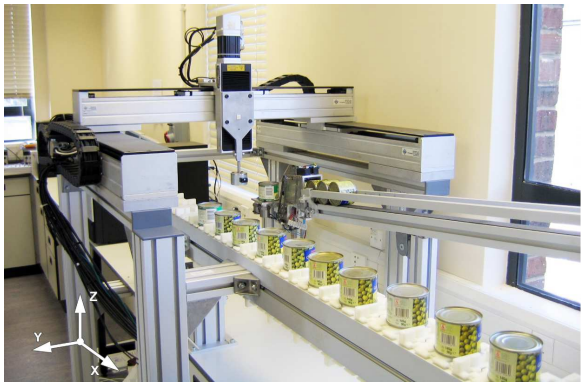

Fig. 1. Three-axis gantry robot test platform.

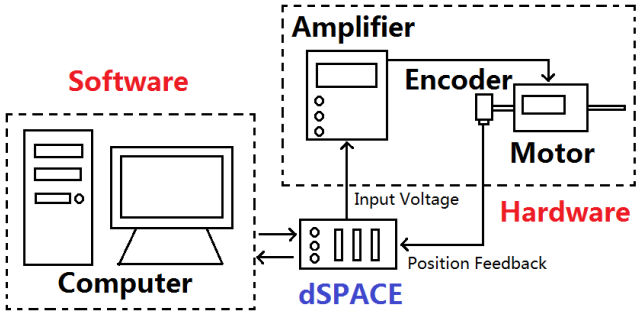

Fig. 2. Structure block diagram of gantry robot platform.

Remark 4. In practical applications, the controller should be equipped with a large enough computation load to ensure the ILC update procedure to be completed before the start of the next trial of the repetitive task. In this sense, the implementation of the proposed algorithm will not violate the planned task execution time in practice.

IV. Verification on a Gantry Robot

In this section, the performance of Algorithm 1 is verified on a three-axis gantry robot test platform, which fully replicates the practical environment with model uncertainty and random disturbance. Also, comparisons with other control methods are made to clarify its advantages .

A. Test platform description

The gantry robot shown in Figure 1 is employed as the test platform. This gantry robot comprises three axes marked as x, y and z, which are perpendicular to each other. The individual motions of the three axes together give rise to the hybrid motion of end-effector in a given 3D space. Therefore, this gantry robot structure allows the implementation of certain robotic tasks, such as pickup a payload from the dispenser and then place it down on the moving convey below.

The input of the gantry robot is in unit of voltage, and the displacement of each axis is measured by an incremental encoder as the output. The block structure overview of the gantry robot test platform is shown in Figure 2. To build up the test platform, a powerful dSPACE DS1103 micro-controller is considered to serve as the interface between the

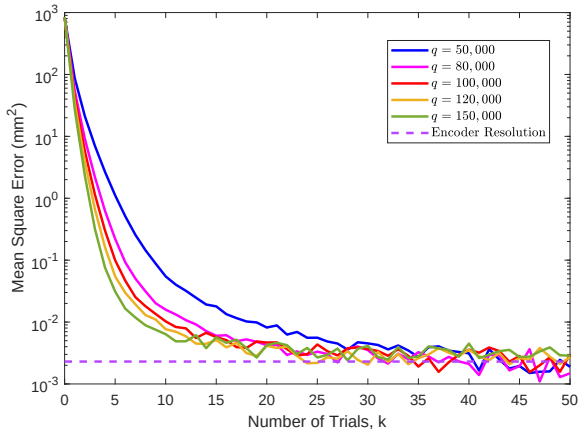

Fig. 3. Mean square path following errors of Algorithm 1 with different values of q.

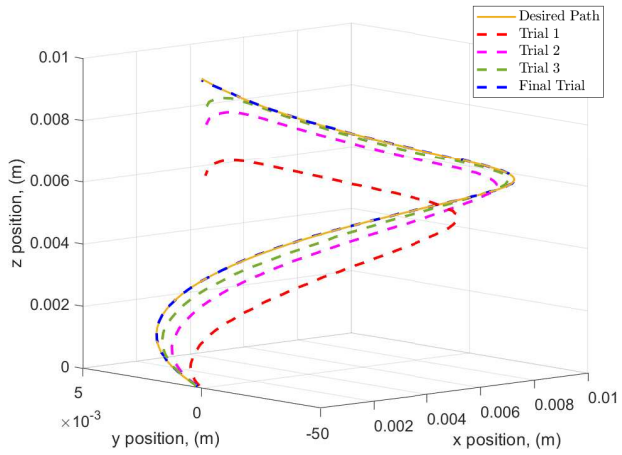

Fig. 4. Hybrid output trajectories of Algorithm 1 for the initial few trials and the final trial with $q = 100,000$.

software (host computer), and the hardware (gantry robot). For each axis, a BNC channel is used to send the input voltage ~~signal~~ to an Aerotech model BA10 linear amplifier. Renishaw RGH22 linear encoders with a resolution of $32\mu m$ for x-axis and y-axis, and Aerotech rotary encoder for z-axis with a resolution of $16\mu m$ are used to receive the position feedback of corresponding motors.

B. Path Following Task Specifications

To evaluate the performance of the proposed algorithms, the path following task is specified as: using all three axes ($m = 3$) to follow a path defined by

$$\tilde{r}(s) = \begin{bmatrix} 0.005cos(-2\pi s + \pi) + 0.005 \\ 0.005sin(-2\pi s + \pi) \\ 0.01s \end{bmatrix}, \ s \in [0,\ 1], \quad (34)$$

within the given time $T = 2s$. The nominal system models of the three axes are considered as

$$G_x(s) = \frac{0.05}{s},\ G_y(s) = \frac{0.05}{s},\ G_z(s) = \frac{0.03}{s}. \quad (35)$$

To provide baseline tracking, disturbance rejection and smooth edge tracking, a feedback proportional controller is connected in parallel with the axis to give

$$\frac{G_x(s)}{1 + K_x G_x(s)},\ \frac{G_y(s)}{1 + K_y G_y(s)},\ \frac{G_z(s)}{1 + K_z G_z(s)}, \quad (36)$$

where $K_x = 100$, $K_y = 300$ and $K_z = 100$. The input saturation constraint (11) is considered with

$$\mathcal{M}(t) = [1.10,\ 1.55,\ 1.10]^\top,\ t \in [0,\ T], \quad (37)$$

and the speed and the acceleration constraints in (12) are defined as $v_{max} = 2.00$ and $a_{max} = 1.00$ to prevent the system damage to the gantry robot. For simplicity, the weighting matrices are denoted as $Q = qI$, and $R = \hat{r}I$, where q and $\hat{r}$ are positive scalars.

C. Practical Performance of Algorithm 1

Algorithm 1 is applied to perform the path following task specified in Section IV-B. The initial input signal and motion profile are chosen as $u_0 = 0$ and $\theta_0(t) = t/T$ respectively. To compare the influence of weighting parameters on the convergence performance of this algorithm, the authors set $\hat{r} = 1$ as a constant value, and choose the values of q as $50,000$, $80,000$, $100,000$, $120,000$, and $150,000$ respectively.

To evaluate the accuracy of the algorithm in a fair manner, the concept of mean square error is used in this paper to quantify to the average squared error value. Algorithm 1 is repeatedly performed with different values of q, and the mean square path following errors at each trial are plotted in Figure 3 as the coloured curves to show their monotonic convergence properties. Also, a larger weighting parameter q contributes to a faster convergence rate. The mean square errors converge to the accuracy level of 10^{-3} approximately and fluctuate around certain values due to the measuring limit of the incremental encoders. These fluctuations after convergence cannot be avoided in practice, and can be only reduced by applying more precise sensors.

To further evaluate the performance of Algorithm 1, the experiment with $q = 100,000$ is explained in detail. The hybrid output trajectories for the initial few trials and the final trial are plotted as the coloured curves in Figure 4, which shows the path following performance from a direct point of view. The hybrid output trajectory at each trial gradually follows the desired path (the yellow curve), and the hybrid trajectory at the final trial accurately follows the path. Therefore, the information in Figure 3 and 4 gives rise to the conclusion that Algorithm 1 guarantees high path following accuracy of the robotic tasks.

The comparison between the initial and final motion profiles are made in Figure 5, whose speed and acceleration satisfy the motion profile constraint (12) with $v_{max} = 2.00$ and $a_{max} = 1.00$. This figure further indicates the evolution trend of motion profile along the trial axis, such that there gradually forms 4 acceleration-deceleration sub-intervals along the time

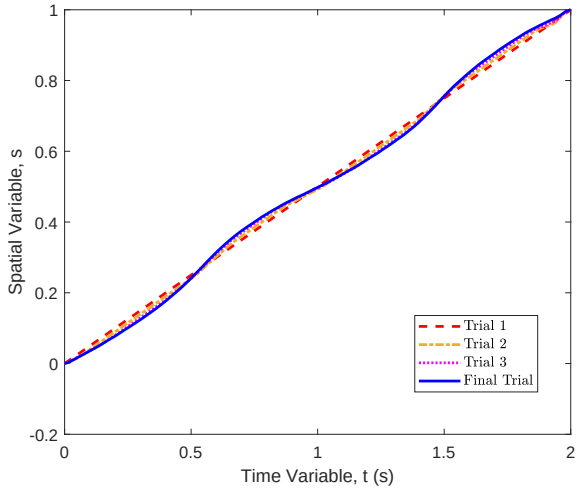

Fig. 5. Motion profiles of Algorithm 1 for the initial few trials and the final trial with $q = 100,000$.

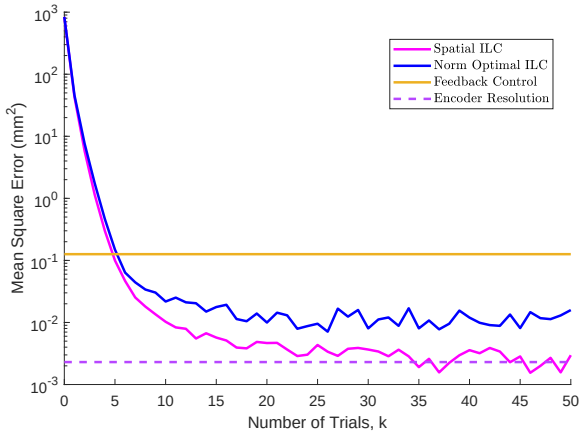

Fig. 6. Comparison between Algorithm 1, norm optimal ILC and feedback control in terms of path following error with $q = 100,000$.

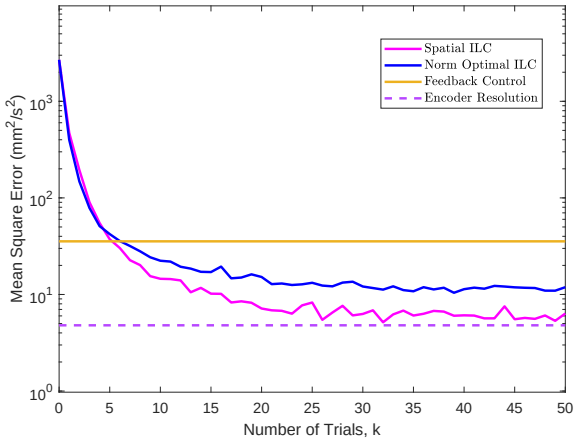

Fig. 7. Comparison between Algorithm 1, norm optimal ILC and feedback control in terms of path following error derivatives with $q = 100,000$.

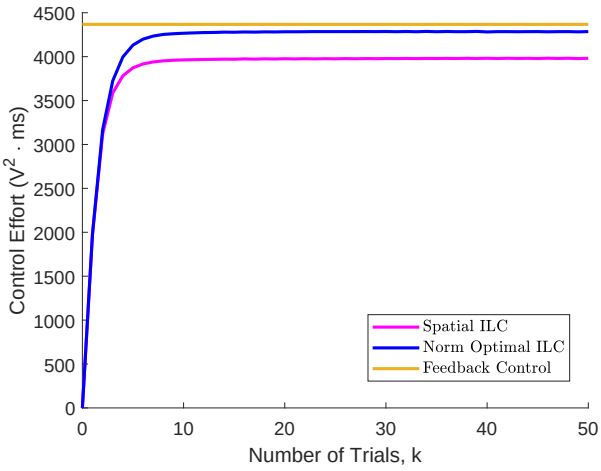

Fig. 8. Comparison between Algorithm 1, norm optimal ILC and feedback control in terms of control effort with $q = 100,000$.

axis. Therefore, the results of Algorithm 1 meet the design requirement (15) of the ILC path following problem.

D. Comparison with Other Control Methods

Furthermore, the algorithm performance is compared with that of norm optimal ILC proposed in [37] and feedback control on the same task specified in Section IV-B. The parameters of norm optimal ILC are similar to those used for the algorithm, i.e. $r = 1$, $q = 100,000$, $u_0 = 0$ and $\theta(t) = t/T$. The feedback control uses a 300 feedback gain to follow the path with motion profile $\theta(t) = t/T$.

The mean square path following error of the four control methods as well as their derivatives are plotted in Figure 6 and 7 for comparison. The mean square path following error of the proposed algorithm converges to the accuracy level of 10^{-3}, and the the derivative converges to a value below 10. However, the corresponding accuracy level of norm optimal ILC is 10^{-2} and above 10 and the accuracy levels of feedback control are 10^{-1} and 35.5 respectively, which is relatively much higher than the values provided by the algorithm. This phenomenon can be explained, since the algorithm iteratively updates the motion profile to reduce the path following error by making the reference trajectory more easily to be followed.

To illustrate other benefits of the algorithm, the concept of control effort is used to quantify to the energy consumption while performing the given task. A comparison is made in Figure 8 on the control effort between these control methods. Although the algorithm design phase does not contain the objective of control effort optimization, its control design freedom brings a positive side effect to reduce around 15% of the control effort required by existing control methods while performing the same path following task.

To evaluate the constraint handling performance, norm optimal ILC is again conducted on the same task specified in Section IV-B with an alternative motion profile

$$\theta(t) = \begin{cases} 0.5t^2, & \text{if } t \leqslant T/2, \\ 1 - 0.5(T-t)^2, & \text{otherwise.} \end{cases} \tag{38}$$

To compare the constraint handling performance, the input

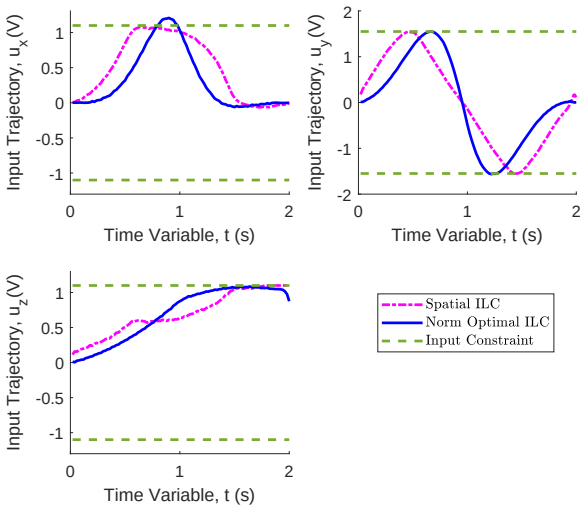

Fig. 9. Comparison between Algorithm 1, and norm optimal ILC in terms of input trajectory with $q = 100,000$.

signal of the three axes at the final trial with $q = 100,000$ are plotted in Figure 9, which reveals that the converged input signal satisfy its input saturation constraint (11). Also, the converged input signals using norm optimal ILC are plotted separately in the same figure. As this motion profile of norm optimal ILC contradicts with the given input constraint, the converged x-axis input signal of norm optimal ILC is beyond the input saturation constraint, which must be avoided. However, the proposed algorithm is capable of handling the system constraints, as its problem setup enables an alternative choice of motion profiles, which agree with the input constraint.

The above comparisons illustrate that the proposed spatial ILC algorithm has better performance than other control methods for robotic path following tasks in terms of error reduction, control effort reduction and constraint handling. Experiments with other R and Q values provide varying convergence speed but similar monotonic convergence performance to the accuracy level of $10^{-3}mm^2$. For brevity, these results are omitted.

V. Conclusion and Future Work

This paper is the first attempt to solve general spatial path following problem with a trial-varying motion profile and system constraints on micro scale manufacturing tasks, which outperforms the existing classical ILC and spatial ILC frameworks with respect to generality as well as tracking accuracy. A powerful and comprehensive spatial ILC algorithm is proposed with robust monotonic convergence analysis to solve this problem, which simultaneously update the input signal and motion profile at each trial. The practical performance of this algorithm is verified on a gantry robot test platform, and comparisons with other control methods are also conducted to show its benefits in terms of error reduction, control effort reduction and constraint handling.

The experimental results demonstrate the feasibility of the algorithms to achieve the design objectives of path following tasks. However, they use linearized approximation to convexify the problems, and other efficient linearized methods with lower computational burden will be examined and evaluated. To further exploit the design freedom of motion profile, other performance index, e.g. control effort, can be added into the cost function $J_k(u,\theta)$ to yield additional practical benefits. Last but not least, the robust convergence performance at general case with input constraint will be investigated. All these points constitute the authors' future research directions, and will be reported separately.

Appendix A
Proof of Theorem 1

The projection operator $P_{S_1}(\cdot)$ solves the problem

$$\begin{aligned} \min_{u} \quad & \|e\|_Q^2 + \|u - u_k\|_R^2 \\ \text{s.t.} \quad & e = r_k - y,\ y = Gu,\ r_k(t) = \tilde{r}(\theta_k(t)), \end{aligned} \tag{39}$$

with an identical structure to the norm optimal problem described in [37], and the only difference is the relaxation of reference profile $r_k(t) = \tilde{r}(\theta_k(t))$ to be trial-varying. In this sense, the analytic solution (17) is derived. The projection operator $P_{S_2}(\cdot)$ yields (18). Since $S_1 \cap S_2 \neq \varnothing$, there must exist $u^* \in \Omega$ and hence the point $(u^*, r_k) \in S_1 \cap S_2$. Thus, substitute $x^{**} = (u_{k+1}, y_{k+1})$, $x^* = (u_k, y_k)$ and $x = (u^*, r_k)$ into the convergence condition (16) to give

$$\begin{aligned} &\|u^* - u_{k+1}\|_R^2 + \|r_k - y_{k+1}\|_Q^2 \\ &\leqslant \|u^* - u_k\|_R^2 + \|r_k - y_k\|_Q^2 . \end{aligned} \tag{40}$$

Since the system (1) is controllable and C has full row rank, the pseudo inverse operator $G^\dagger$ exists, which makes the above inequality equivalent to

$$\begin{aligned} &\left\|G^\dagger(r_k - y_{k+1})\right\|_R^2 + \|r_k - y_{k+1}\|_Q^2 \\ &\leqslant \left\|G^\dagger(r_k - y_k)\right\|_R^2 + \|r_k - y_k\|_Q^2 , \end{aligned} \tag{41}$$

and gives rise to (20). In addition, it follows from the definition of (19) such that

$$\|e_{k+1}\|_Q = \|r_{k+1} - y_{k+1}\|_Q \leqslant \|r_k - y_{k+1}\|_Q , \tag{42}$$

which yields (21).

Appendix B
Proof of Corollary 1

Since the input constraint does not engage, it follows that $u_{k+1} = \tilde{u}_{k+1}$ and thus

$$\|r_k - y_{k+1}\|_Q \leqslant \left\|(I + GG^*)^{-1} e_k\right\|_Q \leqslant \rho \|e_k\|_Q . \tag{43}$$

From the projection (19), the following inequality holds as

$$\|e_{k+1}\|_Q \leqslant \varrho_{k+1} \|r_k - y_{k+1}\|_Q , \ \varrho_{k+1} \leqslant 1, \tag{44}$$

which gives rise to

$$\|e_{k+1}\|_Q \leqslant \rho\varrho_{k+1} \|e_k\|_Q \leqslant \rho \|e_k\|_Q . \tag{45}$$

Therefore, the upper bound of error norm at each trial is $\varepsilon_k = \rho^k(\prod_{i=1}^k \varrho_i) \|e_0\|_Q$ and this value is clearly smaller than the value $\varepsilon_k^{noilc} = \rho^k \|e_0\|_Q$ of norm optimal ILC.

APPENDIX C
PROOF OF THEOREM 2

Since the input constraint does not engage, the equation $u_{k+1} = \tilde{u}_{k+1}$ holds and there exists

$$u_{k+1} = u_k + G^*(I + GG^*)^{-1}\hat{e}_k, \tag{46}$$

and it follows that

$$\begin{aligned} r_k - \hat{y}_{k+1} &= \hat{e}_k - (I + \Delta)GG^*(I + GG^*)^{-1}\hat{e}_k \\ &= (I - \Delta GG^*)(I + GG^*)^{-1}\hat{e}_k. \end{aligned} \tag{47}$$

As $\|I - \Delta GG^*\| \leqslant 1$ and $\left\|(I + GG^*)^{-1}\right\| \leqslant \rho$, thus

$$\begin{aligned} \|r_k - \hat{y}_{k+1}\|_Q &\leqslant \|I - \Delta GG^*\| \left\|(I + GG^*)^{-1}\right\| \|\hat{e}_k\|_Q \\ &\leqslant \rho \|\hat{e}_k\|_Q . \end{aligned} \tag{48}$$

According to the definition of projection (19), the following condition holds as

$$\|\hat{e}_{k+1}\|_Q \leqslant \|r_k - \hat{y}_{k+1}\|_Q . \tag{49}$$

Therefore, the inequalities conditions (48) and (49) together give rise to the robust monotonic convergence condition (27).

REFERENCES

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

[2] X. Zhu and J. Wang, "Double iterative compensation learning control for active training of upper limb rehabilitation robot," *International Journal of Control, Automation and Systems*, vol. 16, no. 3, pp. 1312–1322, apr 2018.

[3] B. Huo, C. T. Freeman, and Y. Liu, "Data-driven gradient-based point-to-point iterative learning control for nonlinear systems," *Nonlinear Dynamics*, vol. 102, no. 1, pp. 269–283, 2020.

[4] X. Jin and J.-X. Xu, "A barrier composite energy function approach for robot manipulators under alignment condition with position constraints," *International Journal of Robust and Nonlinear Control*, vol. 24, no. 17, pp. 2840–2851, 2014.

[5] C. Wang, M. Zheng, Z. Wang, C. Peng, and M. Tomizuka, "Robust iterative learning control for vibration suppression of industrial robot manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 140, no. 1, aug 2017.

[6] H. Chi, X. Li, W. Liang, J. Cao, and Q. Ren, "Iterative learning control for motion trajectory tracking of a circular soft crawling robot," *Frontiers in Robotics and AI*, vol. 6, nov 2019.

[7] J. Bolder, T. Oomen, S. Koekebakker, and M. Steinbuch, "Using iterative learning control with basis functions to compensate medium deformation in a wide-format inkjet printer," *Mechatronics*, vol. 24, no. 8, pp. 944–953, 2014.

[8] J. D. Ratcliffe, P. L. Lewin, E. Rogers, J. J. Hatonen, and D. H. Owens, "Norm-optimal iterative learning control applied to gantry robots for automation applications," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1303–1307, 2006.

[9] C. T. Freeman and T. V. Dinh, "Experimentally verified point-to-point iterative learning control for highly coupled systems," *International Journal of Adaptive Control and Signal Processing*, vol. 29, pp. 302–324, 2015.

[10] D. Bristow, M. Tharayil, and A. Alleyne, "A survey of iterative learning control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–144, 2006.

[11] R. W. Longman, "Iterative learning control and repetitive control for engineering practice," *International Journal of Control*, vol. 73, no. 10, pp. 930–954, 2000.

[12] 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.

[13] 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.

[14] S. Mishra, U. Topcu, and M. Tomizuka, "Optimization-based constrained iterative learning control," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 6, pp. 1613–1621, 2011.

[15] 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.

[16] J. Bolder and T. Oomen, "Inferential iterative learning control: A 2D-system approach," *Automatica*, vol. 71, pp. 247–253, 2016.

[17] P. Sivasankaran and P. Shahabudeen, "Literature review of assembly line balancing problems," *The International Journal of Advanced Manufacturing Technology*, vol. 73, no. 9-12, pp. 1665–1694, 2014.

[18] B. Bischof, T. Gluck, M. Bock, and A. Kugi, "A path/surface following control approach to generate virtual fixtures," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1577–1592, 2018.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] W. van Loock, G. Pipeleers, M. Diehl, J. de Schutter, and J. Swevers, "Optimal path following for differentially flat robotic systems through a geometric problem formulation," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 980–985, 2014.

[24] 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.

[25] 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.

[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] H. Tao, J. Li, Y. Chen, V. Stojanovic, and H. Yang, "Robust point-to-point iterative learning control with trial-varying initial conditions," *IET Control Theory and Applications*, vol. 14, no. 19, pp. 3344–3350, dec 2020.

[28] H. Tao, X. Li, W. Paszke, V. Stojanovic, and H. Yang, "Robust PD-type iterative learning control for discrete systems with multiple time-delays subjected to polytopic uncertainty and restricted frequency-domain," *Multidimensional Systems and Signal Processing*, vol. 32, no. 2, pp. 671–692, jan 2021.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] ——, "A coordinate descent approach to optimal tracking time allocation in point-to-point ILC," *Mechatronics*, vol. 59, pp. 25–34, 2019.

[34] ——, "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.

[35] 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. 59, 2020, 104260.

[36] Y. Chen, B. Chu, and C. T. Freeman, "Iterative learning control for path-following tasks with performance optimization," *IEEE Transactions on Control Systems Technology*, pp. 1–13, 2021.
[37] 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.
[38] D. H. Owens and R. P. Jones, "Iterative solution of constrained differential/algebraic systems," *International Journal of Control*, vol. 27, no. 6, pp. 957–964, 1978.
[39] J. F. Sturm, "Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 625–653, 1999.
[40] J. Löfberg, "YALMIP : A Toolbox for Modeling and Optimization in MATLAB," in *In Proceedings of the CACSD Conference*, 2004.