Data-Driven Iterative Learning Control for Nonlinear Discrete-Time MIMO Systems

Xian Yu, Zhongsheng Hou, *Fellow, IEEE*, Marios M. Polycarpou, *Fellow, IEEE*, and Li Duan

***Abstract*— This article considers the tracking control of unknown nonlinear nonaffine repetitive discrete-time multi-input multi-output systems. Two data-driven iterative learning control (ILC) schemes are designed based on two equivalent dynamic linearization data models of an unknown ideal learning controller, which exists theoretically in the iteration domain. The two control schemes provide ways of selecting learning controllers based on the complexity of the controlled nonlinear systems. The learning control gain matrixes of the two learning controllers are optimized through the steepest descent method using only the measured input–output data of the nonlinear systems. The proposed ILC approaches are pure data-driven since no model information of the controlled systems is involved. The stability and convergence of the proposed ILC approaches are rigorously analyzed under reasonable conditions. Numerical simulation and an experiment based on a Gantry-type linear motor drive system are conducted to verify the effectiveness of the proposed data-driven ILC approaches.**

***Index Terms*— Data-driven iterative learning control (ILC), dynamic linearization (DL), multi-input multi-output (MIMO) system, repetitive nonlinear discrete-time system.**

I. Introduction

MANY systems and processes, such as robots [1], [2], batch processes [3], semiconductor manufacturing processes [4], servo systems [5], digital networks [6], freeway traffic [7] and train operation systems [8]–[11], feature the execution of a given task in the repetitive operating pattern over a finite-time interval. For instance, a robotic manipulator in a manufacturing assembly line carries out a series of welds at predefined locations repeatedly over a finite-time interval; a high-speed train runs strictly from one station to another station repetitively according to the operation timetable every day for a transportation task.

For repetitive systems, iterative learning control (ILC) provides a powerful control method of improving tracking from iteration to iteration over a finite-time interval. Numerous ILC methodologies have been developed over the past three decades, such as contraction mapping based ILC [12], [13], composite energy function-based ILC [14], [15], norm-optimal ILC [11], [16], point-to-point ILC [17], and terminal ILC [18]. Many of the aforementioned control methods are based on the condition of an available affine dynamic model for the controlled plant. From this viewpoint, they are considered as model-based ILC methods. Data-driven ILC is directly designed by using the input–output (I/O) data obtained from the controlled plants or by utilizing the knowledge from data processing [19], [20]. It can handle the control problems of unknown nonlinear systems and has attracted significant attention in recent years. Since all the dynamic information of controlled plants is included in the measured I/O data, concepts, such as system modeling and unmodeled dynamics of the traditional model-based framework, are not needed under the data-driven control framework; therefore, data-driven ILC eliminates some of the key challenging issues of model-based ILC.

One challenging issue for data-driven ILC is to determine the learning controller structure for a class of unknown nonlinear systems. In the case of traditional ILC, the learning controller structure is usually designed *a priori* by experience or some knowledge of the controlled plant, such as the P-type ILC [21]. Another challenge is to design the learning control gain updating algorithm using only the measured I/O data of the controlled plants. The learning control gain in traditional ILC is usually calibrated heuristically from iteration to iteration if the model information of controlled plants is unavailable. The third challenge is to eliminate the problem of poor transient performance for the convergence of the traditional ILC schemes, such as the contraction mapping-based ILC, which is analyzed in the sense of the λ-norm [21].

To address these challenging issues, the dynamic linearization (DL) technique provides a valuable tool. The DL-based data-driven control methods were first proposed by Hou [22] and refined by Hou and Xiong [23]. They have been successfully applied to many practical systems, such as rotor aerodynamic systems [24], water tank systems [25], multi-degree-of-freedom robotic exoskeletons [26], servo

Manuscript received June 20, 2019; revised February 5, 2020; accepted March 4, 2020. Date of publication April 10, 2020; date of current version March 1, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61433002 and Grant 61833001, in part by the European Union's Horizon 2020 Research and Innovation Program (KIOS CoE) under Grant 739551, and in part by the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development. *(Corresponding author: Zhongsheng Hou.)*

Xian Yu is with the Advanced Control Systems Laboratory, School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China, and also with the KIOS Research and Innovation Center of Excellence, University of Cyprus, 1678 Nicosia, Cyprus (e-mail: yuxian@bjtu.edu.cn).

Zhongsheng Hou is with the School of Automation, Qingdao University, Qingdao 266071, China (e-mail: zhshhou@bjtu.edu.cn; zshou@qdu.edu.cn).

Marios M. Polycarpou is with the KIOS Research and Innovation Center of Excellence, University of Cyprus, 1678 Nicosia, Cyprus, and also with the Department of Electrical and Computer Engineering, University of Cyprus, 1678 Nicosia, Cyprus (e-mail: mpolycar@ucy.ac.cy).

Li Duan is with the Advanced Control Systems Laboratory, School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China (e-mail: liduan@bjtu.edu.cn).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.2980588

motor systems [27], and power systems [28]. A more detailed introduction to the DL-based data-driven control methods can be found in [29].

So far, there are no works on DL-based data-driven ILC methods dealing with the aforementioned challenges for unknown nonlinear multi-input multi-output (MIMO) systems in a repetitive operating pattern over a finite-time interval. Compared with single-input single-output (SISO) nonlinear systems, the control problem of MIMO nonlinear systems is more complex due to the couplings among various control inputs and system outputs. As a result, in practice, the dynamics of MIMO systems are generally more difficult to deal with than that of SISO systems. Furthermore, as stated in [30], sometimes it is even difficult to reach the mathematical expression of MIMO nonlinear systems in a meaningful manner. Due to these difficulties, in comparison with considerable results for SISO nonlinear systems, fewer results in the literature are available for MIMO nonlinear systems. Although some preliminary works [31], [32] of DL-based data-driven ILC methods have been developed for SISO systems, the corresponding results cannot be directly applied to MIMO systems. The reason is that the matrix operation in MIMO systems is significantly different from the scalar operation of SISO systems. This article proposes two novel data-driven ILC approaches for nonlinear nonaffine MIMO systems. Specifically, the compact form DL (CFDL) and partial form DL (PFDL) methods in the iteration domain are first used for an unknown ideal learning controller and then two ILC schemes are formulated using only the I/O data of the nonlinear systems. The two ILC schemes provide ways of alternating learning controllers based on the nonlinear complexity of the controlled plants.

Comparing with the existing ILC methods for MIMO systems, the main contributions of this article are as follows.

1) The proposed ILC approaches open a new way of constructing the learning controller structure for a class of unknown nonlinear nonaffine repetitive discrete-time MIMO systems.
2) The proposed ILC approaches are pure data-driven control methods. That is, the learning controllers are independent of the controlled nonlinear systems, and the learning control gain matrixes are automatically tuned using only the measured I/O data of the controlled systems.
3) The steepest descent method is used for designing the learning gain updating algorithm, which avoids the problem of matrix inversion that would be encountered if using the methods in [31] and [32].
4) The stability and convergence of the proposed ILC approaches are guaranteed in the sense of the generic norm under a generalized Lipschitz condition, and their effectiveness is further validated by an experiment on a Gantry-type linear motor drive system.

This article is different from the results presented in [33] and [34]. The works of [33] [34] are used to implement the feedback tracking tasks as time reaches infinity for unknown nonlinear SISO systems in the time domain, while this article designs the iterative learning feedforward control in the iteration domain for unknown nonlinear MIMO systems operating in the repetitive pattern.

The rest of this article is organized as follows. The learning controllers are constructed in Section II. Section III presents the tuning of learning control gain matrixes of the constructed learning controllers, and the stability and convergence analyses of the proposed data-driven ILC approaches. The new results are further verified through simulation and an experiment in Section IV, followed by some conclusions in Section V. In this article, $\|\cdot\|$ denotes any generic vector norm, and the corresponding matrix norm is the induced matrix norm.

II. Dynamic Linearization on Ideal Learning Controller in Iteration Domain

This article considers the repeatable nonlinear discrete-time MIMO system described by

$$\boldsymbol{y}(t+1,j)=\boldsymbol{f}(\boldsymbol{y}(t,j),\ldots,\boldsymbol{y}(t-n_y,j),\\ \boldsymbol{u}(t,j),\ldots,\boldsymbol{u}(t-n_u,j)) \quad (1)$$

where $t=\{1,2,\ldots,T\}$ is the time instant, and positive integer T indicates the terminal time of the finite-time duration, $j=1,2,\ldots$ is the iteration number, $\boldsymbol{y}(t,j)\in\mathbb{R}^m$ and $\boldsymbol{u}(t,j)\in\mathbb{R}^m$ are the system output vector and control input vector, respectively, and m is a known positive integer, the two positive integers $n_y\in\mathbb{Z}_+$ and $n_u\in\mathbb{Z}_+$ are the unknown orders of the system outputs and control inputs of (1); $\boldsymbol{f}(\cdot):\mathbb{R}^{m(n_y+n_u+2)}\mapsto\mathbb{R}^m$ is an unknown nonlinear vector-valued function.

We denote the tracking error vetor as $\boldsymbol{e}(t,j)=\boldsymbol{y_d}(t,j)-\boldsymbol{y}(t,j)$, where $\boldsymbol{y_d}(t,j)\in\mathbb{R}^m$ is the desired output vector of the nonlinear system (1). The control objective is to design a learning controller that can drive $\boldsymbol{e}(t,j)$ to approach zero when the iteration number j tends to infinity.

We assume that an ideal learning controller theoretically exists for the nonlinear system (1), which means that the system (1) controlled by the ideal learning controller will guarantee the system outputs equal to the desired outputs in one step-ahead. The ideal learning controller can be written in the following mathematical expression:

$$\boldsymbol{u}(t,j)=\boldsymbol{C}(\boldsymbol{e}(t+1,j),\ldots,\boldsymbol{e}(t+1,j-n_e),\\ \boldsymbol{u}(t,j-1),\ldots,\boldsymbol{u}(t,j-n_c)) \quad (2)$$

where $\boldsymbol{C}(\cdot):\mathbb{R}^{m(n_e+n_c+1)}\mapsto\mathbb{R}^m$ is an unknown nonlinear vector-valued function, and $n_e\in\mathbb{Z}_+$ and $n_c\in\mathbb{Z}_+$ are the unknown orders of the ideal learning controller (2) on the tracking errors and control inputs, respectively.

It is noted that the ideal learning controller (2) cannot be implemented in practice. Although it is assumed to exist, it is hard to find out the detailed form, or it might be too complex for practical plants. Thus, the main task of this article is to transform the ideal learning controller (2) into a practical learning controller, and in the meantime, to keep it equivalent to (2) in the I/O data sense. In the following, we will discuss this in detail.

A. CFDL on Ideal Learning Controller

The CFDL on the ideal learning controller (2) is based on the following two assumptions.

Assumption 1: The partial derivatives of $\boldsymbol{C}(\cdot)$ with respect to the tracking error vector $\boldsymbol{e}(t+1, j)$ are continuous.

Assumption 2: $\boldsymbol{C}(\cdot)$ satisfies the generalized Lipschitz condition in the iteration domain, that is, if $\|\Delta\boldsymbol{e}(t+1, j)\| \neq 0$, then there exists a constant $\kappa_1 > 0$ such that

$$\|\Delta\boldsymbol{u}(t, j)\| \leq \kappa_1 \|\Delta\boldsymbol{e}(t+1, j)\| \tag{3}$$

where $\Delta\boldsymbol{u}(t, j) = \boldsymbol{u}(t, j) - \boldsymbol{u}(t, j-1)$, $\Delta\boldsymbol{e}(t+1, j) = \boldsymbol{e}(t+1, j) - \boldsymbol{e}(t+1, j-1)$.

Remark 1: Assumption 1 is a common condition for the controller design [29]. Assumption 2 imposes an upper bound limitation on changes of the control inputs affected by changes of the tracking errors. In other words, the ideal learning controller (2) requires to be a physical energy-consuming unit and is stable [29]. Obviously, some linear-type ILC and nonlinear ILC, such as the P-type ILC and Newton-type ILC [35], satisfy these assumptions, where the learning controllers are assumed to have continuously bounded partial derivatives over the corresponding independent variables. Such learning controllers have been successfully applied in several practical applications, such as six-DOF (degree of freedom) industrial robots [2], high-speed trains [8], and switched reluctance motors [35].

The following result provides a parametrization for the ideal learning controller.

Theorem 1: For the ideal learning controller (2), satisfying Assumptions 1 and 2, there must exist $\boldsymbol{\Theta}(t, j)$, called pseudo Jacobian matrix (PJM), such that (2) can be transformed into the following equivalent DL learning controller by using the CFDL method:

$$\Delta\boldsymbol{u}(t, j) = \boldsymbol{\Theta}(t, j)\Delta\boldsymbol{e}(t+1, j) \tag{4}$$

where

$$\boldsymbol{\Theta}(t, j) = \begin{bmatrix} \theta_{11}(t, j) & \theta_{12}(t, j) & \cdots & \theta_{1m}(t, j) \\ \theta_{21}(t, j) & \theta_{22}(t, j) & \cdots & \theta_{2m}(t, j) \\ \vdots & \vdots & \vdots & \vdots \\ \theta_{m1}(t, j) & \theta_{m2}(t, j) & \cdots & \theta_{mm}(t, j) \end{bmatrix} \in \mathbb{R}^{m\times m}$$

$\|\boldsymbol{\Theta}(t, j)\| \leq b_1$, and $b_1 > 0$ is a constant.

Proof: According to the ideal learning controller (2), $\Delta\boldsymbol{u}(t, j)$ can be written as

$$\begin{aligned} \Delta\boldsymbol{u}(t, j) = {} & \boldsymbol{C}(\boldsymbol{e}(t+1, j), \ldots, \boldsymbol{e}(t+1, j-n_e) \\ & \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c)) \\ & - \boldsymbol{C}(\boldsymbol{e}(t+1, j-1), \boldsymbol{e}(t+1, j-1), \ldots, \\ & \boldsymbol{e}(t+1, j-n_e), \boldsymbol{u}(t, j-1), \ldots, \\ & \boldsymbol{u}(t, j-n_c)) + \boldsymbol{\Upsilon}_1(t+1, j-1) \end{aligned} \tag{5}$$

where

$$\begin{aligned} & \boldsymbol{\Upsilon}_1(t+1, j-1) \\ & = \boldsymbol{C}(\boldsymbol{e}(t+1, j-1), \boldsymbol{e}(t+1, j-1), \ldots \boldsymbol{e}(t+1, j-n_e), \\ & \quad \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c)) \\ & \quad - \boldsymbol{C}(\boldsymbol{e}(t+1, j-1), \ldots, \boldsymbol{e}(t+1, j-n_e-1) \\ & \quad \boldsymbol{u}(t, j-2), \ldots, \boldsymbol{u}(t, j-n_c-1)). \end{aligned}$$

By virtue of Assumption 1 and the differential mean value theorem, (5) gives

$$\Delta\boldsymbol{u}(t, j) = \frac{\partial\boldsymbol{C}^*}{\partial\boldsymbol{e}(t+1, j)}\Delta\boldsymbol{e}(t+1, j) + \boldsymbol{\Upsilon}_1(t+1, j-1) \tag{6}$$

where

$$\frac{\partial\boldsymbol{C}^*}{\partial\boldsymbol{e}(t+1, j)} = \begin{bmatrix} \frac{\partial C_1^*}{\partial e_1(t+1, j)} & \frac{\partial C_1^*}{\partial e_2(t+1, j)} & \cdots & \frac{\partial C_1^*}{\partial e_m(t+1, j)} \\ \frac{\partial C_2^*}{\partial e_1(t+1, j)} & \frac{\partial C_2^*}{\partial e_2(t+1, j)} & \cdots & \frac{\partial C_2^*}{\partial e_m(t+1, j)} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial C_m^*}{\partial e_1(t+1, j)} & \frac{\partial C_m^*}{\partial e_2(t+1, j)} & \cdots & \frac{\partial C_m^*}{\partial e_m(t+1, j)} \end{bmatrix}.$$

$\partial C_p^*/\partial e_q(t+1, j)$ indicates the partial derivative of function C_p with respect to $e_q(\cdot)$ at a certain point in the interval $[e_q(t+1, j), e_q(t+1, j-1)]$, where $p = 1, 2, \ldots, m$ and $q = 1, 2, \ldots, m$.

Consider the following equation with matrix $\boldsymbol{\Psi}(t, j) \in \mathbb{R}^{m\times m}$:

$$\boldsymbol{\Upsilon}_1(t+1, j-1) = \boldsymbol{\Psi}(t, j)\Delta\boldsymbol{e}(t+1, j). \tag{7}$$

Since $\|\Delta\boldsymbol{e}(t+1, j)\| \neq 0$ holds, (7) must have at least one solution $\boldsymbol{\Psi}^*(t, j)$. In fact, it has infinite number of solutions. We let

$$\boldsymbol{\Theta}(t, j) = \frac{\partial\boldsymbol{C}^*}{\partial\boldsymbol{e}(t+1, j)} + \boldsymbol{\Psi}^*(t, j). \tag{8}$$

Based on (6)–(8), (4) is obtained. Then, we can get $\|\boldsymbol{\Theta}(t, j)\| \leq b_1$ based on Assumption 2. ■

For convenient description and distinction compared with the DL learning controller in Section II-B, we label the learning controller (4) as the CFDL controller (CFDLc).

B. PFDL on Ideal Learning Controller

The CFDLc (4) indicates that all the nonlinearity of the ideal learning controller (2) is fused into the PJM $\boldsymbol{\Theta}(t, j)$. Since the dynamic model of the nonlinear system (1) is unknown, the unknown PJM $\boldsymbol{\Theta}(t, j)$ must be estimated in real time using some parameter estimation algorithms based on the measured I/O data of (1). However, it would be difficult for an estimation algorithm to track the dynamics of $\boldsymbol{\Theta}(t, j)$, which is caused by the high nonlinearities and complexities due to the time-varying parameter, order, and structure in the system (1). An effective strategy of handling this issue is to develop another type of DL data model [36]. That is, the PFDL data modeling method is required, and another two assumptions, similar to Assumptions 1 and 2, are introduced.

Assumption 3: The partial derivatives of $\boldsymbol{C}(\cdot)$ with respect to the tracking errors $\boldsymbol{e}(t+1, j), \ldots, \boldsymbol{e}(t, j-l+1)$ are continuous. $l \in \mathbb{Z}_+$ is called the linearization length constant (LLC).

Assumption 4: $\boldsymbol{C}(\cdot)$ satisfies the generalized Lipschitz condition in the iteration domain, that is, if $\|\Delta \boldsymbol{E}(t+1, j)\| \neq 0$, then there exists a constant $\kappa_2 > 0$ such that

$$\|\Delta \boldsymbol{u}(t, j)\| \leq \kappa_2 \|\Delta \boldsymbol{E}(t+1, j)\| \tag{9}$$

where $\Delta \boldsymbol{E}(t+1, j) = [\Delta \boldsymbol{e}^T(t+1, j), \ldots, \Delta \boldsymbol{e}^T(t+1, j-l+1)]^T \in \mathbb{R}^{ml}$, $\Delta \boldsymbol{e}(t+1, j-i+1) = \boldsymbol{e}(t+1, j-i+1) - \boldsymbol{e}(t+1, j-i)$, and $i = 1, 2, \ldots, l$.

The following result provides a different parametrization based on the PFDL method.

Theorem 2: For the ideal learning controller (2), satisfying Assumptions 3 and 4, there must exist $\bar{\boldsymbol{\Theta}}(t, j)$, called pseudopartitioned Jacobian matrix (PPJM), such that (2) can be transformed into the following equivalent DL learning controller by using the PFDL method:

$$\Delta \boldsymbol{u}(t, j) = \bar{\boldsymbol{\Theta}}(t, j) \Delta \boldsymbol{E}(t+1, j) \tag{10}$$

where $\bar{\boldsymbol{\Theta}}(t, j) = [\boldsymbol{\Theta}_1(t, j), \ldots, \boldsymbol{\Theta}_l(t, j)] \in \mathbb{R}^{m \times ml}$

$$\boldsymbol{\Theta}_i(t, j) = \begin{bmatrix} \theta_{11i}(t, j) & \theta_{12i}(t, j) & \cdots & \theta_{1mi}(t, j) \\ \theta_{21i}(t, j) & \theta_{22i}(t, j) & \cdots & \theta_{2mi}(t, j) \\ \vdots & \vdots & \vdots & \vdots \\ \theta_{m1i}(t, j) & \theta_{m2i}(t, j) & \cdots & \theta_{mmi}(t, j) \end{bmatrix} \in \mathbb{R}^{m \times m}$$

$\|\bar{\boldsymbol{\Theta}}(t, j)\| \leq b_2$, and $b_2 > 0$ is a constant.

Proof: From (6), we have

$$\begin{aligned} \Delta \boldsymbol{u}(t, j) = {} & \frac{\partial \boldsymbol{C}^*}{\partial \boldsymbol{e}(t+1, j)} \Delta \boldsymbol{e}(t+1, j) \\ & + \boldsymbol{\Upsilon}_1(\boldsymbol{e}(t+1, j-1), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \\ & \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c-1)) \end{aligned} \tag{11}$$

and (11) is further rewritten as

$$\begin{aligned} \Delta \boldsymbol{u}(t, j) = {} & \frac{\partial \boldsymbol{C}^*}{\partial \boldsymbol{e}(t+1, j)} \Delta \boldsymbol{e}(t+1, j) \\ & + \boldsymbol{\Upsilon}_1(\boldsymbol{e}(t+1, j-1), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \\ & \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c-1)) \\ & - \boldsymbol{\Upsilon}_1(\boldsymbol{e}(t+1, j-2), \boldsymbol{e}(t+1, j-2), \ldots, \\ & \boldsymbol{e}(t+1, j-n_e-1), \boldsymbol{u}(t, j-1), \ldots, \\ & \boldsymbol{u}(t, j-n_c-1)) \\ & + \boldsymbol{\Upsilon}_1(\boldsymbol{e}(t+1, j-2), \boldsymbol{e}(t+1, j-2), \ldots, \\ & \boldsymbol{e}(t+1, j-n_e-1), \boldsymbol{u}(t, j-1), \ldots, \\ & \boldsymbol{u}(t, j-n_c-1)). \end{aligned} \tag{12}$$

By virtue of Assumption 3 and the differential mean value theorem, (12) yields

$$\begin{aligned} \Delta \boldsymbol{u}(t, j) = {} & \frac{\partial \boldsymbol{C}^*}{\partial \boldsymbol{e}(t+1, j)} \Delta \boldsymbol{e}(t+1, j) \\ & + \frac{\partial \boldsymbol{\Upsilon}_1^*}{\partial \boldsymbol{e}(t+1, j-1)} \Delta \boldsymbol{e}(t+1, j-1) \\ & + \boldsymbol{\Upsilon}_2(\boldsymbol{e}(t+1, j-2), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \\ & \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c-1)) \end{aligned} \tag{13}$$

where

$$\begin{aligned} & \frac{\partial \boldsymbol{\Upsilon}_1^*}{\partial \boldsymbol{e}(t+1, j-1)} \\ & = \begin{bmatrix} \frac{\partial \Upsilon_{11}^*}{\partial e_1(t+1, j-1)} & \frac{\partial \Upsilon_{11}^*}{\partial e_2(t+1, j-1)} & \cdots & \frac{\partial \Upsilon_{11}^*}{\partial e_m(t+1, j-1)} \\ \frac{\partial \Upsilon_{12}^*}{\partial e_1(t+1, j-1)} & \frac{\partial \Upsilon_{12}^*}{\partial e_2(t+1, j-1)} & \cdots & \frac{\partial \Upsilon_{12}^*}{\partial e_m(t+1, j-1)} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \Upsilon_{1m}^*}{\partial e_1(t+1, j-1)} & \frac{\partial \Upsilon_{1m}^*}{\partial e_2(t+1, j-1)} & \cdots & \frac{\partial \Upsilon_{1m}^*}{\partial e_m(t+1, j-1)} \end{bmatrix}. \end{aligned}$$

$\partial \Upsilon_{1p}^* / \partial e_q(t+1, j-1)$ indicates the partial derivative value of function Υ_{1p} with respect to $e_q(\cdot)$ at a certain point in the interval $[e_q(t+1, j-1), e_q(t+1, j-2)]$, and $\boldsymbol{\Upsilon}_2(\boldsymbol{e}(t+1, j-2), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c-1)) = \boldsymbol{\Upsilon}_1(\boldsymbol{e}(t+1, j-2), \boldsymbol{e}(t+1, j-2), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c-1))$.

Similar to the process from (11)–(13), we obtain

$$\begin{aligned} \Delta \boldsymbol{u}(t, j) = {} & \left[\frac{\partial \boldsymbol{C}^*}{\partial \boldsymbol{e}(t+1, j)}, \frac{\partial \boldsymbol{\Upsilon}_1^*}{\partial \boldsymbol{e}(t+1, j-1)}, \ldots, \right. \\ & \left. \frac{\partial \boldsymbol{\Upsilon}_{l-1}^*}{\partial \boldsymbol{e}(t+1, j-l+1)} \right] \times \Delta \boldsymbol{E}(t+1, j) \\ & + \boldsymbol{\Upsilon}_l(\boldsymbol{e}(t+1, j-l), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \\ & \boldsymbol{u}(t, j-1), \ldots, \boldsymbol{u}(t, j-n_c-1)). \end{aligned} \tag{14}$$

Consider the following equation with matrix $\bar{\boldsymbol{\Psi}}(t, j) \in \mathbb{R}^{m \times ml}$:

$$\begin{aligned} & \boldsymbol{\Upsilon}_l(\boldsymbol{e}(t+1, j-l), \ldots, \boldsymbol{e}(t+1, j-n_e-1), \boldsymbol{u}(t, j-1), \ldots, \\ & \boldsymbol{u}(t, j-n_c-1)) = \bar{\boldsymbol{\Psi}}(t, j) \Delta \boldsymbol{E}(t+1, j). \end{aligned} \tag{15}$$

Then, there exists at least one solution $\bar{\boldsymbol{\Psi}}^*(t, j)$ for (15) when $\|\boldsymbol{E}(t+1, j)\| \neq 0$. We let

$$\begin{aligned} \bar{\boldsymbol{\Theta}}(t, j) = {} & \left[\frac{\partial \boldsymbol{C}^*}{\partial \boldsymbol{e}(t+1, j)}, \frac{\partial \boldsymbol{\Upsilon}_1^*}{\partial \boldsymbol{e}(t+1, j-1)}, \ldots, \right. \\ & \left. \frac{\partial \boldsymbol{\Upsilon}_{l-1}^*}{\partial \boldsymbol{e}(t+1, j-l+1)} \right] + \bar{\boldsymbol{\Psi}}^*(t, j). \end{aligned} \tag{16}$$

With (14)–(16), (10) is obtained. Then, it has $\|\bar{\boldsymbol{\Theta}}(t, j)\| \leq b_2$ according to Assumption 4. ■

Similar to (4), we label the learning controller (10) as the PFDL controller (PFDLc).

Remark 2: Since the PFDLc (10) is an equivalent expression of the ideal learning controller (2) with iteration-varying linearization structure, it can be considered theoretically as the optimal learning controller for the unknown nonlinear repetitive MIMO system (1), while a traditional learning controller is usually determined *a priori* by experience or some knowledge of the MIMO systems. In other words, the selection problem of the learning controller is addressed through the two systematic approaches as shown in Theorems 1 and 2.

C. Practical Learning Controller

Both of the CFDLc (4) and PFDLc (10) cannot be implemented in practice since the noncausal tracking error vector $\boldsymbol{e}(t+1, j)$ exists in (4) and (10). Similar to [31] and [32], the practical learning controllers corresponding to (4) and (10) are considered in this article

$$\text{CFDLc:}\quad \Delta\boldsymbol{u}(t,j) = -\boldsymbol{\Theta}(t,j)\boldsymbol{e}(t+1,j-1) \tag{17}$$

$$\text{PFDLc:}\quad \Delta\boldsymbol{u}(t,j) = \bar{\boldsymbol{\Theta}}(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1) \tag{18}$$

where $\Delta\bar{\boldsymbol{E}}(t+1,j-1) = [-\boldsymbol{e}^T(t+1,j-1), \Delta\boldsymbol{e}^T(t+1,j-1),\ldots,\Delta\boldsymbol{e}^T(t+1,j-l+1)]^T \in \mathbb{R}^{ml}$.

The CFDLc (17) and PFDLc (18) provide two ways of alternating learning controllers based on the nonlinear complexity of the controlled system (1). That is, (18) can be used by choosing the considerable LLC l if (17) is difficult to track the high nonlinearity of (1), and the large l value does not increase the structural complexity of the PFDLc (18). Besides, in the special case of $l = 1$, (18) is the same as (17). Furthermore, for high-dimensional MIMO systems, it is suggested to select the PFDLc (18) rather than the CFDLc (17) since the high dimension usually implies strong coupling interactions of the controlled plant. Theoretically speaking, the more complex the controlled plant is, the more complicated the controller structure would be. Although the two learning controllers (17) and (18) are mathematically equivalent, in practice, they have different capabilities in capturing the complex dynamics of the controlled plant. In this case, the LLC l should be appropriately selected according to the complexity of the controlled plant.

Following the method in [31] and [32], it can be easily obtained that the P-type ILC $\Delta\boldsymbol{u}(t,j) = \boldsymbol{K}_p\boldsymbol{e}(t+1,j-1)$ [37] and the high-order ILC $\boldsymbol{u}(t,j) = \sum_{p=1}^{s}\boldsymbol{M}\boldsymbol{u}(t,j-p) + \sum_{p=1}^{s}\boldsymbol{Q}\boldsymbol{e}(t+1,j-p)$ [12] can be considered as special cases of the practical CFDLc (17) and PFDLc (18), respectively, through proper choices for $\boldsymbol{\Theta}(t,j)$ and $\bar{\boldsymbol{\Theta}}(t,j)$, where $\boldsymbol{K}_p$, $\boldsymbol{M}$, and $\boldsymbol{Q}$ are the learning control gain matrixes with appropriate dimensions, and s is a positive integer. In other words, the learning control gain matrixes of the P-type ILC and high-order ILC can be designed and analyzed in the proposed data-driven ILC schemes.

III. Controller Design and Stability Analysis

With the practical CFDLc (17) and PFDLc (18), the problem of determining the learning controller structure is addressed for unknown nonlinear MIMO systems by virtue of extending the CFDL and PFDL methods to the multidimensional space in the iteration domain on the ideal learning controller (2). The remaining main task is to design the learning control gain matrix, updating algorithms for $\boldsymbol{\Theta}(t,j)$ and $\bar{\boldsymbol{\Theta}}(t,j)$ using only the measured I/O data of the nonlinear system (1).

A. CFDLc-ILC Design and Convergence Analysis

We consider the following control criterion function:

$$J = \frac{1}{2}\|\boldsymbol{e}(t+1,j)\|^2 + \frac{1}{2}\lambda_t\|\Delta\boldsymbol{u}(t,j)\|^2 \tag{19}$$

where $\lambda_t > 0$ is a weight factor. The first term on the right-hand side of (19) reflects the tracking performance, while the second term provides a control action energy minimization.

Substituting the CFDLc (17) into the control criterion function (19), the derivative of (19) over $\boldsymbol{\Theta}(t,j)$ yields

$$\begin{aligned}\frac{\partial J}{\partial\boldsymbol{\Theta}(t,j)} &= \frac{\partial\boldsymbol{y}(t+1,j)}{\partial\boldsymbol{u}(t,j)}\boldsymbol{e}(t+1,j)\boldsymbol{e}^T(t+1,j-1)\\ &\quad + \lambda_t\boldsymbol{\Theta}(t,j)\boldsymbol{e}(t+1,j-1)\boldsymbol{e}^T(t+1,j-1).\end{aligned} \tag{20}$$

With (20), an updating algorithm of $\boldsymbol{\Theta}(t,j)$ is achieved by adopting the following steepest descent method:

$$\begin{aligned}&\hat{\boldsymbol{\Theta}}(t,j+1)\\ &\quad= \hat{\boldsymbol{\Theta}}(t,j) - \eta_t\frac{\partial J}{\partial\hat{\boldsymbol{\Theta}}(t,j)}\\ &\quad= \hat{\boldsymbol{\Theta}}(t,j)(I - \eta_t\lambda_t\boldsymbol{e}(t+1,j-1)\boldsymbol{e}^T(t+1,j-1))\\ &\qquad - \eta_t\frac{\partial\boldsymbol{y}(t+1,j)}{\partial\boldsymbol{u}(t,j)}\boldsymbol{e}(t+1,j)\boldsymbol{e}^T(t+1,j-1)\end{aligned} \tag{21}$$

where $\eta_t \in (0,1]$ is a step size of $\boldsymbol{\Theta}(t,j)$, $I \in \mathbb{R}^{m\times m}$ is an identity matrix, and $\hat{\boldsymbol{\Theta}}(t,j)$ is the estimation of $\boldsymbol{\Theta}(t,j)$.

The updating algorithm (21) cannot be applied to the nonlinear system (1) because the term $\boldsymbol{e}(t+1,j)$ is noncausal and the derivative term $\partial\boldsymbol{y}(t+1,j)/\partial\boldsymbol{u}(t,j)$ cannot be computed directly since the dynamic model of (1) is completely unknown. The I/O data relationship between $\boldsymbol{y}(t+1,j)$ and $\boldsymbol{u}(t,j)$ is a key for the ILC design.

In order to obtain the relationship, the equivalent DL data model for the nonlinear system (1) is utilized by the CFDL method in the iteration domain [36], which is described by

$$\Delta\boldsymbol{y}(t+1,j) = \boldsymbol{\Phi}(t,j)\Delta\boldsymbol{u}(t,j) \tag{22}$$

where $\Delta\boldsymbol{y}(t+1,j) = \boldsymbol{y}(t+1,j) - \boldsymbol{y}(t+1,j-1)$, the unknown matrix

$$\boldsymbol{\Phi}(t,j) = \begin{bmatrix}\phi_{11}(t,j) & \phi_{12}(t,j) & \cdots & \phi_{1m}(t,j)\\ \phi_{21}(t,j) & \phi_{22}(t,j) & \cdots & \phi_{2m}(t,j)\\ \vdots & \vdots & \vdots & \vdots\\ \phi_{m1}(t,j) & \phi_{m2}(t,j) & \cdots & \phi_{mm}(t,j)\end{bmatrix} \in \mathbb{R}^{m\times m}$$

is called the PJM of the nonlinear system (1) satisfying $\|\boldsymbol{\Phi}(t,j)\| \le b_3$. $b_3 > 0$ is a constant. The issue of matrix inversion cannot be addressed if we use the same updating algorithms in [31] and [32] for $\boldsymbol{\Theta}(t,j)$ by virtue of the data model (22), while it is avoided in this article through combining the data model (22) and the updating algorithm (21).

Remark 3: The I/O data relationship of the nonlinear system (1) also can be obtained by using any predictive algorithms, such as the subspace identification [38] and multiple-step-ahead prediction [39], when the dynamic model of the nonlinear system (1) is unavailable. Actually, when the plant model is known, it can be directly used to compute the two terms $\boldsymbol{e}(t+1,j)$ and $\partial\boldsymbol{y}(t+1,j)/\partial\boldsymbol{u}(t,j)$. In other words, the tuning problem of the learning controller parameters also can be solved by virtue of the updating algorithm (21) when the plant model is known.

Remark 4: In considering with noisy sensing/control, $\boldsymbol{y}(t,j)$ should be modified as $\boldsymbol{y}_w(t,j) = \boldsymbol{y}(t,j) + \boldsymbol{w}(t,j)$, where $\boldsymbol{y}_w(t,j)$ is the noisy system output vector and $\boldsymbol{w}(t,j)$ is the sensing noise vector. In this case, the DL data model (22) needs to be rewritten as $\Delta\boldsymbol{y}_w(t+1,j) = \boldsymbol{\Phi}(t,j)\Delta\boldsymbol{u}(t,j)$, where $\Delta\boldsymbol{y}_w(t+1,j) = \boldsymbol{y}_w(t+1,j) - \boldsymbol{y}_w(t+1,j-1)$. Readers are referred to the results in [40] for further development regarding the corresponding learning controller designs and robust stability analyses. For the case of partial observation, it usually indicates the state space model of a controlled plant with partial state measurements, while this article only focuses on the input–output model as demonstrated by (1) and (22) where only the system outputs require to be measured.

The DL data model (22) demonstrates that the functional relationship between $\boldsymbol{y}(t+1, j)$ and $\boldsymbol{u}(t, j)$ is determined by a simple and dynamic structure. Equation (22) is equivalent to the nonlinear system (1) at any arbitrary operation condition [36], while the I/O data relationship built by other predictive algorithms is only an approximation to the nonlinear system (1). Besides, these predictive algorithms generally require offline training with large amounts of I/O data of the nonlinear system (1), which increases the computational complexity.

The following projection algorithm is used to automatically tune the unknown $\boldsymbol{\Phi}(t, j)$ [36]:

$$\hat{\boldsymbol{\Phi}}(t, j) = \hat{\boldsymbol{\Phi}}(t, j-1) + \frac{\rho_t \boldsymbol{e}_{\boldsymbol{\Phi}}(t+1, j-1)\Delta \boldsymbol{u}^T(t, j-1)}{\mu_t + \|\Delta \boldsymbol{u}(t, j-1)\|^2} \tag{23}$$

where $\boldsymbol{e}_{\boldsymbol{\Phi}}(t+1, j-1) = \Delta \boldsymbol{y}(t+1, j-1) - \hat{\boldsymbol{\Phi}}(t, j-1)\Delta \boldsymbol{u}(t, j-1)$, $\rho_t \in (0, 1]$ is a step size of $\boldsymbol{\Phi}(t, j)$, $\mu_t > 0$ is a weight factor, and $\hat{\boldsymbol{\Phi}}(t, j)$ is the estimation of $\boldsymbol{\Phi}(t, j)$.

With (22) and (23), the estimations of the two terms $\boldsymbol{e}(t+1, \ j)$ and $\partial \boldsymbol{y}(t+1, j)/\partial \boldsymbol{u}(t, j)$ is achieved by

$$\begin{aligned} \hat{\boldsymbol{e}}(t+1, j) &= \boldsymbol{y}_d(t+1, j) - \boldsymbol{y}(t+1, j-1) \\ &\quad - \hat{\boldsymbol{\Phi}}(t, j)\Delta \boldsymbol{u}(t, j) &(24)\\ \frac{\partial \hat{\boldsymbol{y}}(t+1, j)}{\partial \boldsymbol{u}(t, j)} &= \hat{\boldsymbol{\Phi}}(t, j). &(25) \end{aligned}$$

Therefore, the updating algorithm (21) and the CFDLc (17) are, respectively, further derived as

$$\begin{aligned} \hat{\boldsymbol{\Theta}}(t, j+1) &= \hat{\boldsymbol{\Theta}}(t, j)(I - \eta_t \lambda_t \boldsymbol{e}(t+1, j-1)\boldsymbol{e}^T(t+1, j-1)) \\ &\quad - \eta_t \hat{\boldsymbol{\Phi}}(t, j)\hat{\boldsymbol{e}}(t+1, j)\boldsymbol{e}^T(t+1, j-1) &(26)\\ \boldsymbol{u}(t, j) &= \boldsymbol{u}(t, j-1) - \hat{\boldsymbol{\Theta}}(t, j)\boldsymbol{e}(t+1, j-1). &(27) \end{aligned}$$

Based on the ILC law (27), the two updating algorithms (23) and (26), and the estimations of $\boldsymbol{e}(t+1, j)$ and $\partial \boldsymbol{y}(t+1, j)/\partial \boldsymbol{u}(t, j)$ given in (24) and (25), a data-driven ILC scheme is summarized and described in the algorithm CFDLc-ILC (Algorithm 1).

Remark 5: At step 2 in the CFDLc-ILC, the applied prespecified controller can be designed by using the model-free adaptive control [20] or any other data-driven control approaches. For simplicity, the model-free adaptive control is adopted for generating the required I/O data of the nonlinear system (1) and the PJM estimation $\hat{\boldsymbol{\Phi}}(t, j)$ before iteration proceeds. It is worth noting that $\hat{\boldsymbol{\Phi}}(t, j)$ is required to be calculated if other data-driven approaches are applied, and it can be obtained through the I/O data of the nonlinear system (1) by virtue of the virtual data model (22).

In the CFDLc-ILC, $\text{sat}(\underline{b}, \overline{b})$ denotes the projection of a variable b on $[\underline{b}, \overline{b}]$ and $j_{\max}$ is the given maximum iteration number, and $\underline{b}, \overline{b} \in \mathbb{R}$. Obviously, the norms of both $\underline{\boldsymbol{\Theta}}(t)$ and $\overline{\boldsymbol{\Theta}}(t)$ are less than b_1.

Algorithm 1 CFDLc-ILC

```
1: Given $j = 1$ and an arbitrary $\hat{\boldsymbol{\Theta}}(t, j)$.
2: Obtain $\boldsymbol{y}(t, j)$, $\boldsymbol{u}(t, j)$ and $\hat{\boldsymbol{\Phi}}(t, j)$ by applying one prespecified controller to (1).
3: Compute $\hat{\boldsymbol{\Phi}}(t, j)$ by (23) with the resetting mechanism $\hat{\boldsymbol{\Phi}}(t, j) = \text{sat}(\underline{\boldsymbol{\Phi}}(t), \overline{\boldsymbol{\Phi}}(t))$.
4: Compute $\hat{\boldsymbol{\Theta}}(t, j+1)$ by (26) with the resetting mechanism $\hat{\boldsymbol{\Theta}}(t, j+1) = \text{sat}(\underline{\boldsymbol{\Theta}}(t), \overline{\boldsymbol{\Theta}}(t))$.
5: Obtain $\boldsymbol{u}(t, j)$ by (27), and $\boldsymbol{y}(t+1, j)$ by driving (1).
6: If $j < j_{\max}$, let $j = j + 1$, and goto Step 3; else goto Step 7.
7: The iteration terminates.
```

Next, the convergence of the proposed CFDLc-ILC is analyzed based on the following definition.

Definition 1: The ideal PJM $\boldsymbol{\Theta}^*(t)$ drives $\boldsymbol{e}(t+1, j) = \boldsymbol{0}$, where $\|\boldsymbol{\Theta}^*(t)\| \leq b_1$.

Based on the Definition 1 and (17), it has

$$\Delta \boldsymbol{u}^*(t, j) = -\boldsymbol{\Theta}^*(t, j)\boldsymbol{e}(t+1, j-1). \tag{28}$$

Theorem 3: Let the nonlinear system (1) satisfying Assumptions 1 and 2 be controlled by the CFDLc-ILC. Then, the tracking errors of the nonlinear system (1) are asymptotically convergent in the iteration domain if the following condition is satisfied:

$$b_3 b_1 < \frac{1}{2}. \tag{29}$$

Proof: We define the following function:

$$V(t+1, j) = \|\boldsymbol{e}(t+1, j)\|^2. \tag{30}$$

Then, the difference of the function in the iteration domain is

$$\begin{aligned} \Delta V(t+1, j) &= V(t+1, j) - V(t+1, j-1) \\ &= \|\boldsymbol{e}(t+1, j)\|^2 - \|\boldsymbol{e}(t+1, j-1)\|^2 \\ &= \|\Delta \boldsymbol{e}(t+1, j) + \boldsymbol{e}(t+1, j-1)\|^2 \\ &\quad - \|\boldsymbol{e}(t+1, j-1)\|^2. \end{aligned} \tag{31}$$

According to (22), we have

$$\begin{aligned} \Delta \boldsymbol{e}(t+1, j) &= (\boldsymbol{y}_d(t+1, j) - \boldsymbol{y}(t+1, j)) \\ &\quad - (\boldsymbol{y}_d(t+1, j-1) - \boldsymbol{y}(t+1, j-1)) \\ &= \Delta \boldsymbol{y}_d(t+1, j) - \Delta \boldsymbol{y}(t+1, j) \\ &= \Delta \boldsymbol{y}_d(t+1, j) - \boldsymbol{\Phi}(t, j)\Delta \boldsymbol{u}(t+1, j) \end{aligned} \tag{32}$$

where $\Delta \boldsymbol{y}_d(t+1, j) = \boldsymbol{y}_d(t+1, j) - \boldsymbol{y}_d(t+1, j-1)$. Taking (32) into (31) yields

$$\begin{aligned} \Delta V(t+1, j) &= \|\Delta \boldsymbol{y}_d(t+1, j) - \boldsymbol{\Phi}(t, j)\Delta \boldsymbol{u}(t+1, j) \\ &\quad + \boldsymbol{e}(t+1, j-1)\|^2 - \|\boldsymbol{e}(t+1, j-1)\|^2. \end{aligned} \tag{33}$$

Based on (27), (33) is rewritten as

$$\begin{aligned} \Delta V(t+1, j) &= \|\Delta \boldsymbol{y}_d(t+1, j) + \boldsymbol{\Phi}(t, j)\hat{\boldsymbol{\Theta}}(t, j)\boldsymbol{e}(t+1, j-1) \\ &\quad + \boldsymbol{e}(t+1, j-1)\|^2 - \|\boldsymbol{e}(t+1, j-1)\|^2. \end{aligned} \tag{34}$$

We define the estimation error of the PJM $\boldsymbol{\Theta}(t,j)$ as $\tilde{\boldsymbol{\Theta}}(t,j) = \hat{\boldsymbol{\Theta}}(t,j) - \boldsymbol{\Theta}^*(t)$, then

$$\begin{aligned}\Delta V(t+1,j) &= \|\Delta\boldsymbol{y}_d(t+1,j) + \boldsymbol{\Phi}(t,j)\tilde{\boldsymbol{\Theta}}(t,j)\boldsymbol{e}(t+1,j-1)\\ &\quad + \boldsymbol{e}(t+1,j+1) + \boldsymbol{\Phi}(t,j)\boldsymbol{\Theta}^*(t,j)\boldsymbol{e}(t+1,j-1)\|^2\\ &\quad - \|\boldsymbol{e}(t+1,j-1)\|^2. \end{aligned}\tag{35}$$

With Definition 1, (28), and (32), (35) gives

$$\begin{aligned}\Delta V(t+1,j) &= \|\boldsymbol{\Phi}(t,j)\tilde{\boldsymbol{\Theta}}(t,i)\boldsymbol{e}(t+1,j-1)\|^2 - \|\boldsymbol{e}(t+1,j-1)\|^2\\ &\le (\|\boldsymbol{\Phi}(t,j)\|^2\|\tilde{\boldsymbol{\Theta}}(t,j)\|^2 - 1)\|\boldsymbol{e}(t+1,j-1)\|^2. \end{aligned}\tag{36}$$

Since $\|\boldsymbol{\Phi}(t,j)\| \le b_3$, $\|\hat{\boldsymbol{\Theta}}(t,j)\| \le b_1$ and $\|\boldsymbol{\Theta}^*(t)\| \le b_1$, (36) further yields

$$\Delta V(t+1,j) \le \left(4b_3^2b_1^2 - 1\right)\|\boldsymbol{e}(t+1,j-1)\|^2. \tag{37}$$

Based on condition (29), inequality (37) indicates

$$\Delta V(t+1,j) < 0. \tag{38}$$

Inequality (38) illustrates that the tracking errors of the nonlinear system (1) are asymptotically convergent in the iteration domain. ■

B. PFDLc-ILC Design and Stability Analysis

Taking the PFDLc (18) into the control criterion function(19), the updating algorithm of the PPJM $\bar{\boldsymbol{\Theta}}(t,j)$ similar to (21) is obtained using the steepest descent method

$$\begin{aligned}\hat{\bar{\boldsymbol{\Theta}}}(t,j+1) &= \hat{\bar{\boldsymbol{\Theta}}}(t,j) - \eta_t\frac{\partial J}{\partial\hat{\bar{\boldsymbol{\Theta}}}(t,j)}\\ &= \hat{\bar{\boldsymbol{\Theta}}}(t,j)(\bar{I} - \eta_t\lambda_j\Delta\bar{\boldsymbol{E}}(t+1,j-1)\Delta\bar{\boldsymbol{E}}^T(t+1,j-1))\\ &\quad + \eta_t\frac{\partial\boldsymbol{y}(t+1,j)}{\partial\boldsymbol{u}(t,j)}\boldsymbol{e}(t+1,j)\Delta\bar{\boldsymbol{E}}^T(t+1,j-1) \end{aligned}\tag{39}$$

where $\bar{I} \in \mathbb{R}^{ml\times ml}$ is an identity matrix and $\hat{\bar{\boldsymbol{\Theta}}}(t,j)$ is the estimation of $\bar{\boldsymbol{\Theta}}(t,j)$. Similar to (26) and (27), the updating algorithm (39) and the PFDLc (18) are, respectively, modified as

$$\begin{aligned}\hat{\bar{\boldsymbol{\Theta}}}(t,j+1) &= \hat{\bar{\boldsymbol{\Theta}}}(t,j)(\bar{I} - \eta_t\lambda_t\Delta\bar{\boldsymbol{E}}(t+1,j-1)\Delta\bar{\boldsymbol{E}}^T(t+1,j-1))\\ &\quad + \eta_t\hat{\boldsymbol{\Phi}}(t,j)\hat{\boldsymbol{e}}(t+1,j)\Delta\bar{\boldsymbol{E}}^T(t+1,j-1) \end{aligned}\tag{40}$$

$$\begin{aligned}\boldsymbol{u}(t,j) &= \boldsymbol{u}(t,j-1) + \hat{\bar{\boldsymbol{\Theta}}}(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1). \end{aligned}\tag{41}$$

Based on the ILC law (41), the two updating algorithms (23) and (40), and the estimations of $\boldsymbol{e}(t+1,j)$ and $\partial\boldsymbol{y}(t+1,j)/\partial\boldsymbol{u}(t,j)$ given in (24) and (25), the second data-driven ILC scheme is summarized in the algorithm PFDLc-ILC (Algorithm 2).

Remark 6: The detailed steps of the CFDLc-ILC and PFDLc-ILC illustrate that no explicit or implicit dynamic model and structural information of the nonlinear system (1) are used in the control system design. The PJMs $\boldsymbol{\Theta}(t,j)$ and $\boldsymbol{\Phi}(t,j)$, and the PPJM $\bar{\boldsymbol{\Theta}}(t,j)$ are automatically tuned using only the measured I/O data of the nonlinear system (1). Besides, the obtained CFDLc (17) and PFDLc (18) are independent of the nonlinear system (1). Therefore, the two proposed CFDLc-ILC and PFDLc-ILC are pure data-driven ILC approaches.

Algorithm 2 PFDLc-ILC

```
1: Given $j = 1$ and an arbitrary $\hat{\bar{\boldsymbol{\Theta}}}(t,1)$.
2: Obtain $\boldsymbol{y}(t,j)$, $\boldsymbol{u}(t,j)$ and $\hat{\boldsymbol{\Phi}}(t,j)$ by applying one pre-specified controller to (1).
3: Compute $\hat{\boldsymbol{\Phi}}(t,j)$ by (23) with the resetting mechanism $\hat{\boldsymbol{\Phi}}(t,j) = \mathrm{sat}(\underline{\boldsymbol{\Phi}}(t), \overline{\boldsymbol{\Phi}}(t))$.
4: Compute $\hat{\bar{\boldsymbol{\Theta}}}(t,j+1)$ by (40) with the resetting mechanism $\hat{\bar{\boldsymbol{\Theta}}}(t,j+1) = \mathrm{sat}(\underline{\bar{\boldsymbol{\Theta}}}(t), \overline{\bar{\boldsymbol{\Theta}}}(t))$.
5: Obtain $\boldsymbol{u}(t,j)$ by (41) with the resetting mechanism $\boldsymbol{u}(t,j) = \mathrm{sat}(\underline{\boldsymbol{u}}(t), \overline{\boldsymbol{u}}(t))$, and $\boldsymbol{y}(t+1,j)$ by driving (1).
6: If $j < j_{\max}$, let $j = j+1$, and goto Step 3; else goto Step 7.
7: The iteration terminates.
```

According to Assumption 4, the norms of both $\underline{\bar{\boldsymbol{\Theta}}}(t)$ and $\overline{\bar{\boldsymbol{\Theta}}}(t)$ are less than b_2. Similar to Definition 1, the following definition is required for the stability analysis of the proposed PFDLc-ILC.

Definition 2: The ideal PPJM $\bar{\boldsymbol{\Theta}}^*(t)$ drives $\boldsymbol{e}(t+1,j) = \mathbf{0}$, where $\|\bar{\boldsymbol{\Theta}}^*(t)\| \le b_2$.

Based on Definition 2 and (18), it has

$$\Delta\boldsymbol{u}^*(t,j) = \bar{\boldsymbol{\Theta}}^*(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1). \tag{42}$$

Theorem 4: Let the nonlinear system (1) satisfying Assumptions 3 and 4, be controlled by the PFDLc-ILC. Then, the tracking errors of the nonlinear system (1) are uniformly ultimately bounded in the iteration domain for all $t \in \{1,2,\ldots,T\}$ if the following two conditions are satisfied:

$$b_3b_2 < \frac{1}{2} \tag{43}$$

$$|\Delta y_d(t+1,j)| \le b_4 \tag{44}$$

where b_4 is a positive constant.

Proof: With (18) and (30), we rewrite (33) as

$$\begin{aligned}\Delta V(t+1,j) &= \|\Delta\boldsymbol{y}_d(t+1,j) - \boldsymbol{\Phi}(t,j)\hat{\bar{\boldsymbol{\Theta}}}(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1)\\ &\quad + \boldsymbol{e}(t+1,j-1)\|^2 - \|\boldsymbol{e}(t+1,j-1)\|^2. \end{aligned}\tag{45}$$

Defining the estimation error of the PJM $\bar{\boldsymbol{\Theta}}(t,j)$ as $\tilde{\bar{\boldsymbol{\Theta}}}(t,j) = \hat{\bar{\boldsymbol{\Theta}}}(t,j) - \bar{\boldsymbol{\Theta}}^*(t)$, (45) yields

$$\begin{aligned}\Delta V(t+1,j) &= \|\Delta\boldsymbol{y}_d(t+1,j) - \boldsymbol{\Phi}(t,j)\tilde{\bar{\boldsymbol{\Theta}}}(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1)\\ &\quad + \boldsymbol{e}(t+1,j-1) - \boldsymbol{\Phi}(t,j)\bar{\boldsymbol{\Theta}}^*(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1)\|^2\\ &\quad - \|\boldsymbol{e}(t+1,j-1)\|^2. \end{aligned}\tag{46}$$

Based on Definition 2 and (42), (46) further gives

$$\begin{aligned}\Delta V(t+1,j) &\\ =\|\Delta \boldsymbol{y}_d(t+1,j)-\boldsymbol{\Phi}(t,j)\tilde{\bar{\boldsymbol{\Theta}}}(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1)&\\ +\boldsymbol{e}(t+1,j-1)-\boldsymbol{\Phi}(t,j)\Delta\boldsymbol{u}^*(t,j)\|^2&\\ -\|\boldsymbol{e}(t+1,j-1)\|^2. &\quad (47)\end{aligned}$$

Equation (32) and the Definition 2 yield that

$$\begin{aligned}\Delta\boldsymbol{e}^*(t+1,j)&=-\boldsymbol{e}(t+1,j-1)\\ &=\Delta\boldsymbol{y}_d(t+1,j)-\boldsymbol{\Phi}(t,j)\Delta\boldsymbol{u}^*(t+1,j). \quad (48)\end{aligned}$$

Combining (47) and (48) together gives

$$\begin{aligned}\Delta V(t+1,j)&=\|-\boldsymbol{\Phi}(t,j)\tilde{\bar{\boldsymbol{\Theta}}}(t,j)\Delta\bar{\boldsymbol{E}}(t+1,j-1)\|^2\\ &\quad -\|\boldsymbol{e}(t+1,j-1)\|^2\\ &\le (\|\boldsymbol{\Phi}(t,j)\|\|\tilde{\bar{\boldsymbol{\Theta}}}(t,j)\|\|\Delta\bar{\boldsymbol{E}}(t+1,j-1)\|)^2\\ &\quad -\|\boldsymbol{e}(t+1,j-1)\|^2. \quad (49)\end{aligned}$$

Furthermore, $\boldsymbol{u}(t,j)=\mathrm{sat}(\underline{\boldsymbol{u}}(t),\overline{\boldsymbol{u}}(t))$ leads to

$$\|\boldsymbol{u}(t,j)\|\le 2(\|\underline{\boldsymbol{u}}(t)\|+\|\overline{\boldsymbol{u}}(t)\|). \quad (50)$$

Based on the condition (44) and $\|\boldsymbol{\Phi}(t,j)\|\le b_3$, (32) generates

$$\begin{aligned}\|\Delta\boldsymbol{e}(t+1,j)\|&\le\|\Delta\boldsymbol{y}_d(t+1,j)\|+\|\boldsymbol{\Phi}(t,j)\|\|\Delta\boldsymbol{u}(t+1,j)\|\\ &\le b_4+2b_3(\|\underline{\boldsymbol{u}}(t)\|+\|\overline{\boldsymbol{u}}(t)\|). \quad (51)\end{aligned}$$

Therefore, there exists a positive constant α such that

$$\|\Delta\bar{\boldsymbol{E}}(t+1,j)\|^2\le\|\boldsymbol{e}(t+1,j-1)\|^2+\alpha. \quad (52)$$

Besides, $\|\hat{\bar{\boldsymbol{\Theta}}}(t,j)\|\le b_2$ and $\|\bar{\boldsymbol{\Theta}}^*(t)\|\le b_2$ imply that

$$\|\tilde{\bar{\boldsymbol{\Theta}}}(t,j)\|=\|\hat{\bar{\boldsymbol{\Theta}}}(t,j)-\bar{\boldsymbol{\Theta}}^*(t)\|\le 2b_2. \quad (53)$$

With $\|\boldsymbol{\Phi}(t,j)\|\le b_3$, (52), and (53), (49) yields

$$\begin{aligned}\Delta V(t+1,j)&\le 4b_3^2b_2^2(\|\boldsymbol{e}(t+1,j-1)\|^2+\alpha)\\ &\quad -\|\boldsymbol{e}(t+1,j-1)\|^2\\ &=\left(4b_3^2b_2^2-1\right)\|\boldsymbol{e}(t+1,j-1)\|^2+4\alpha b_3^2b_2^2\\ &\qquad (54)\end{aligned}$$

which implies

$$\begin{aligned}\|\boldsymbol{e}(t+1,j)\|^2&\le\left(4b_3^2b_2^2\right)^{j-1}\|\boldsymbol{e}(t+1,1)\|^2\\ &\quad+4\alpha b_3^2b_2^2\frac{1-\left(4b_3^2b_2^2\right)^{j-2}}{1-4b_3^2b_2^2}. \quad (55)\end{aligned}$$

Based on condition (43), inequality (55) satisfies

$$\lim_{j\to\infty}\|\boldsymbol{e}(t+1,j)\|=2b_3b_2\left(\alpha/\left(1-4b_3^2b_2^2\right)\right)^{\frac{1}{2}}. \quad (56)$$

Therefore, the tracking errors of the nonlinear system (1) are uniformly ultimately bounded in the iteration domain for all $t\in\{1,2,\ldots,T\}$. ■

Remark 7: With the PFDLc-ILC, the stability of the nonlinear system (1) is guaranteed by Theorem 4. Through reasonable setting of the $\bar{\boldsymbol{\Theta}}(t,j)$, a low bound can be achieved when $\bar{\boldsymbol{\Theta}}(t,j)$ approximates $\bar{\boldsymbol{\Theta}}^*(t,j)$; that is, $\|\bar{\boldsymbol{\Theta}}(t,j)-\bar{\boldsymbol{\Theta}}^*(t,j)\|$ tends to zero. In this case, the condition (43) is easily satisfied. Consequently, the tracking errors for all $t\in\{1,2,\ldots,T\}$ are close to zero when the iteration j tends to infinity.

For the CFDLc-ILC, it can be seen that, at each time instant t in each iteration j, the I/O data required to be collected for generating the control input vector $\boldsymbol{u}(t,j)$ are $\boldsymbol{y}(t+1,j-1)$, $\boldsymbol{y}(t+1,j-2)$, $\boldsymbol{u}(t,j-1)$, and $\boldsymbol{u}(t,j-2)$; that is, the collected data size is $4m$. The I/O data for PFDLc-ILC are $\boldsymbol{y}(t+1,j-1),\ldots,\boldsymbol{y}(t+1,j-l),\boldsymbol{u}(t,j-1)$ and $\boldsymbol{u}(t,j-2)$; that is, the collected data size is $(l+2)m$. The aforementioned observations indicate that, in general, the proposed two data-driven ILC approaches have fast computation speeds to a certain extent. Furthermore, since the controlled system is repeatable and operates from identical initial conditions in each iteration, then, for the same time instant t in different iterations, the change of $\boldsymbol{\Phi}(t,j)$ is slow or remains almost the same even if the controlled plant is data sensitive or the system itself has some obvious variances [20], [36], which implies that the two proposed ILC approaches are relatively robust even with raw collected I/O data.

IV. Simulation and Experiment

To demonstrate the effectiveness of the proposed CFDLc-ILC and PFDLc-ILC, a simulation and an experiment are considered in this section. The simulation is conducted on a complicated nonlinear MIMO discrete-time system. The experiment is carried out on an LMG2A-CB6-CC8 Gantry-type linear motor drive system to further demonstrate the applicability of the proposed CFDLc-ILC and PFDLc-ILC in practice. For comparison, the P-type ILC and high-order ILC are also applied. It should be pointed out that the two systems are considered unknown and are used only to generate the I/O data for the simulation and practical application.

A. Simulation

The following nonlinear discrete-time MIMO system is considered [36]:

$$\begin{cases} x_{11}(t+1,j)=\dfrac{x_{11}^2(t,j)}{1+x_{11}^2(t,j)}+0.3x_{12}(t,j),\\ x_{12}(t+1,j)=\dfrac{x_{11}^2(t,j)}{1+x_{12}^2(t,j)+x_{21}^2(t,j)+x_{22}^2(t,j)}\\ \qquad\qquad +a(t)u_1(t,j)\\ x_{21}(t+1,j)=\dfrac{x_{21}^2(t,j)}{1+x_{21}^2(t,j)}+0.2x_{22}(t,j),\\ x_{22}(t+1,j)=\dfrac{x_{21}^2(t,j)}{1+x_{11}^2(t,j)+x_{12}^2(t,j)+x_{22}^2(t,j)}\\ \qquad\qquad +b(t)u_2(t,j)\\ y_1(t+1,j)=x_{11}(t+1,j)\\ y_2(t+1,j)=x_{21}(t+1,j)\end{cases} \quad (57)$$

where $a(t)=1+0.1\sin(2\pi t/150)$ and $b(t)=1+0.1\cos(2\pi t/150)$ are the time-varying parameters. From the definitions of the system output vector and control input vector, it is known that $\boldsymbol{u}(t,j)=[u_1(t,j),u_2(t,j)]^T$ and $\boldsymbol{y}(t,j)=[y_1(t,j),y_2(t,j)]^T$.

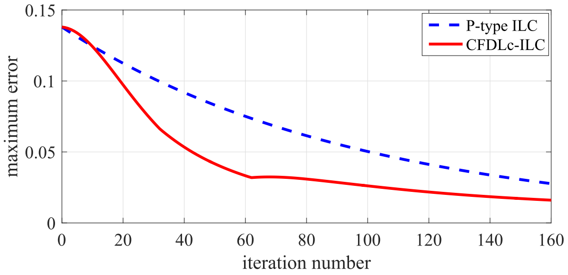


Fig. 1. Learning performances of P-type ILC and CFDLc-ILC on $e_{\max,1}$.

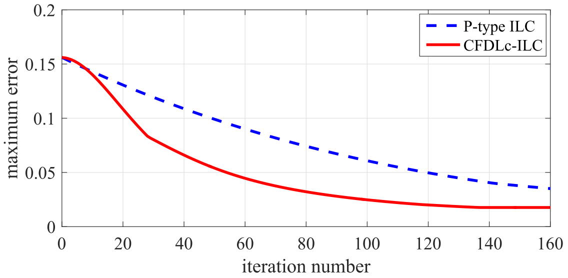


Fig. 2. Learning performances of P-type ILC and CFDLc-ILC on $e_{\max,2}$.

The desired output vector $\mathbf{y}_d(t) = [y_1(t), y_2(t)]^T$ is given as

$$\begin{cases} y_{d,1}(t) = -0.25 + 0.25\cos(0.25\pi t/100) + 0.25\sin(0.5\pi t/100) \\ y_{d,2}(t) = 0.25\sin(0.25\pi t/100) + 0.25\sin(0.5\pi t/100). \end{cases} \tag{58}$$

In the simulations, the terminal time instant is $T = 800$ and the maximum iteration number is 160. The initial system outputs and control inputs are given as $y_1(1, j) = 0.0039$, $y_2(1, j) = 0.0059$, $y_1(2, j) = 0.0078$, $y_2(2, j) = 0.0118$ and $u_1(1, j) = u_2(1, j) = u_1(2, j) = u_2(2, j) = 0$, respectively.

The simulation results are shown in Figs. 1–4, which provide the learning performances of the four ILC approaches: CFDLc-ILC, PFDLc-ILC, P-type ILC, and high-order ILC, where the vertical axis is the maximum absolute value of the tracking error $e_{\max,i_m}(j) = sup_{t\in 1,2,\ldots,800}|y_{d,i_m}(t) - y_{i_m}(t, j)|$, $i_m = 1, 2,\ j = 1, 2, \ldots, 160$.

From Figs. 1 and 2, it is obvious that the convergence speed of CFDLc-ILC is faster than that of P-type ILC on both of the tracking errors $e_{\max,1}(j)$ and $e_{\max,2}(j)$, which reduce to 0.0160 and 0.0177 for CFDLc-ILC, and 0.0277 and 0.0351 for P-type ILC, respectively, when the iteration period terminates. Furthermore, the faster convergence speed of PFDLc-ILC than that of high-order ILC is obvious, as shown in Figs. 3 and 4, where the $e_{\max,1}(j)$ and $e_{\max,2}(j)$ for high-order ILC still remain 0.0850 after 34 iterations and 0.1009 after 37 iterations, respectively.

B. Experiment

The experimental setup of the LMG2A-CB6-CC8 Gantry-type linear motor drive system is shown in Fig. 5. It mainly includes five components: host computer, links-box, terminal block, driver, and linear motor drive system. The host computer configured with the Simulink and RT-SIM software is responsible for modeling, monitoring, and managing the simulated control environment of the linear motor drive system. The links-box is used to compile the simulated control environment and communicate with the terminal block. The terminal block transfers the computed voltage signal (control input) from the links-box through Ethernet, and the x-axis and y-axis position signals of the linear motor drive system, where the x-axis and y-axis are assembled orthogonally with two linear motors. The driver serves for driving the x-axis and y-axis to perform a given control task.

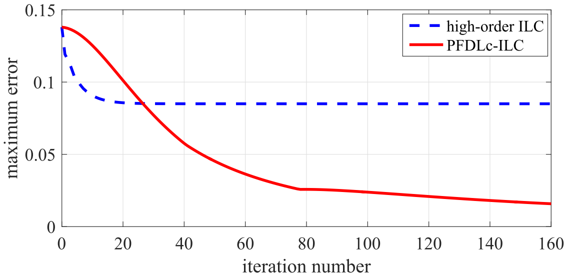


Fig. 3. Learning performances of high-order ILC and PFDLc-ILC on $e_{\max,1}$.

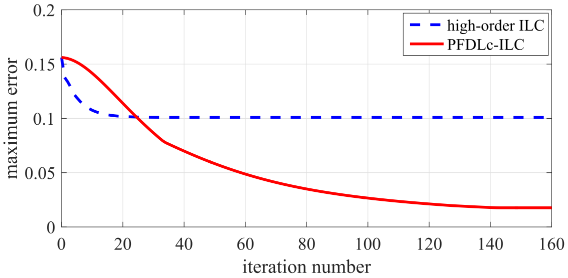


Fig. 4. Learning performances of high-order ILC and PFDLc-ILC on $e_{\max,2}$.

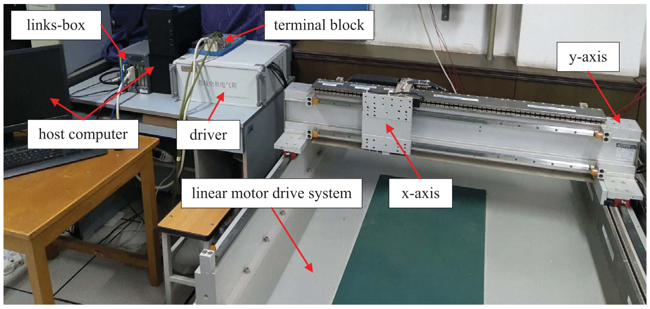


Fig. 5. Experimental setup.

In the following experiments, two tasks are considered: task 1 is to track a circle, and task 2 is to track a cardioid with noisy sensing.

1) Circle: In the experiment, two cases are considered for the linear motor drive system. In case 1, identical desired outputs are imposed, and case 2 is the iteration-varying desired outputs. For the two cases, the iteration number that the linear motor drive system runs is 6. The sampling time is 0.001 s and the experiment time is given as 12.6 s in each iteration, which means the terminal time $T = 12\,600$.

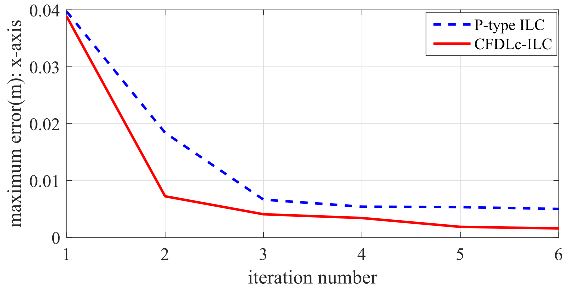


Fig. 6. Learning performances of P-type ILC and CFDLc-ILC on x-axis (Case 1).

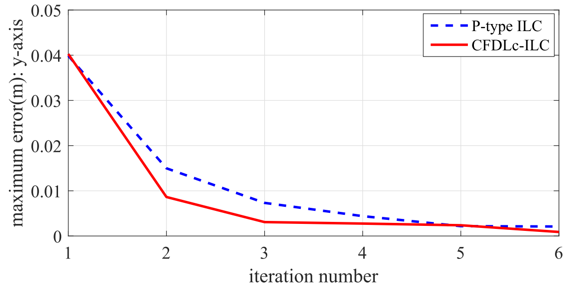


Fig. 7. Learning performances of P-type ILC and CFDLc-ILC on y-axis (Case 1).

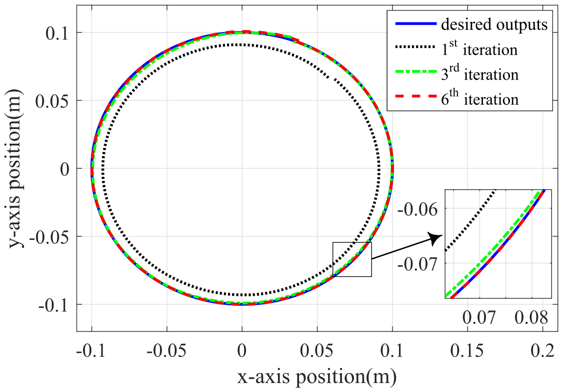


Fig. 8. Tracking trajectories of CFDLc-ILC (Case 1).

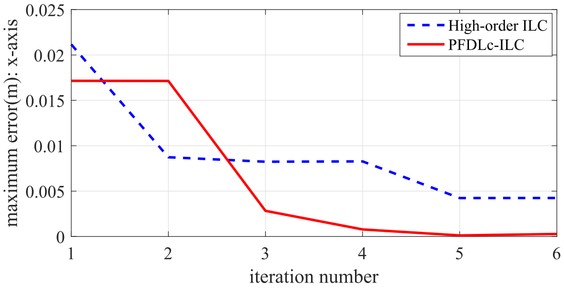


Fig. 9. Learning performances of high-order ILC and PFDLc-ILC on x-axis (Case 1).

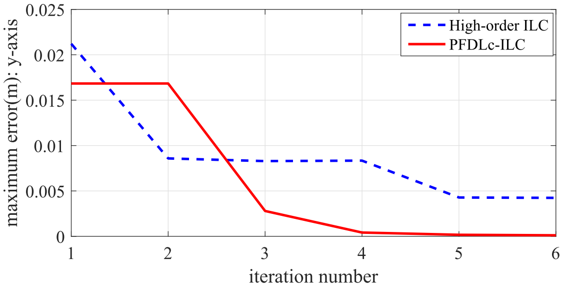


Fig. 10. Learning performances of high-order ILC and PFDLc-ILC on y-axis (Case 1).

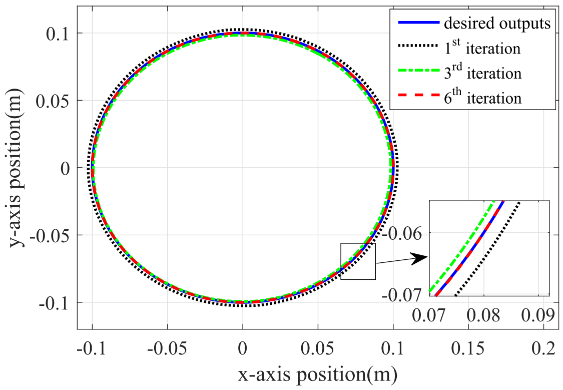


Fig. 11. Tracking trajectories of PFDLc-ILC (Case 1).

Case 1: The identical desired outputs are

$$\begin{cases} y_{d,1}(t) = 0.1\cos(0.5\pi(t+2400)/12600) \\ y_{d,2}(t) = 0.1\sin(0.5\pi(t+2400)/12600) \end{cases} \tag{59}$$

where $y_{d,1}(t)$ and $y_{d,2}(t)$ are the desired x-axis and y-axis positions, respectively. Actually, the desired x-axis and y-axis positions formulate the desired circle with radius $r = 0.1$ m in the xy coordinate system.

Figs. 6–8 illustrate the experimental results between CFDLc-ILC and P-type ILC, where Figs. 6 and 7 give the learning convergence of the tracking errors on x-axis and y-axis positions versus the iteration, and Fig. 8 shows the x-axis and y-axis positions at the first, third, and sixth iterations by applying CFDlc-ILC.

It is demonstrated from Figs. 6 and 7 that the convergence speed of CFDLc-ILC is faster than that of P-type ILC on both of the tracking errors $e_{\max,1}$ and $e_{\max,2}$, in which $e_{\max,1}$ and $e_{\max,2}$ of CFDLc-ILC are lower 0.0034 m and 0.0012 m, respectively, than those of P-type ILC when the iteration period terminates. Besides, from Fig. 8, it can be seen that the tracking circle has been much close to the desired circle after the third iteration, and the details shown in Fig. 8 demonstrate that the CFDLc-ILC can achieve an approximately perfect tracking at the sixth iteration.

The experimental results between PFDLc-ILC and high-order ILC are shown in Figs. 9–11, where Fig. 11 gives the tracking trajectories of PFDLc-ILC at the first, third, and sixth iterations. By comparing the values of $e_{\max,1}$ and $e_{\max,2}$ at the last iteration between PFDLc-ILC and high-order ILC, as shown in Figs. 9 and 10, the tracking accuracies of PFDLc-ILC raise 15.2213× and 36.8292× than that of the high-order ILC on the x-axis and y-axis, respectively, only after six iterations. From the details shown in Fig. 11, approximately perfect tracking is achieved using PFDLc-ILC after six iterations.

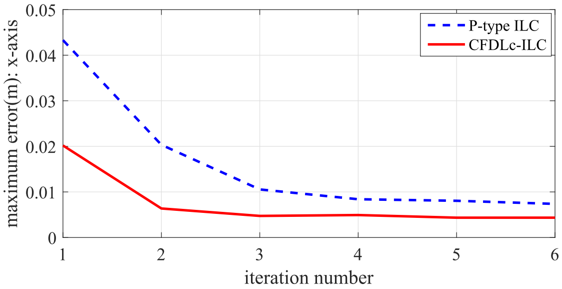


Fig. 12. Learning performances of P-type ILC and CFDLc-ILC on x-axis (Case 2).

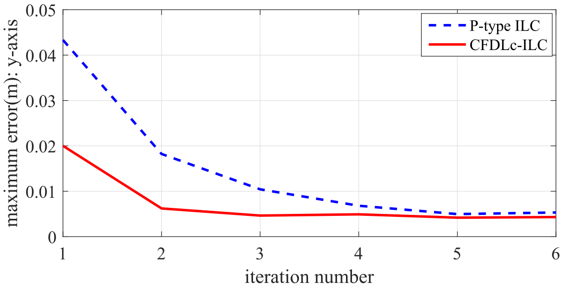


Fig. 13. Learning performances of P-type ILC and CFDLc-ILC on y-axis (Case 2).

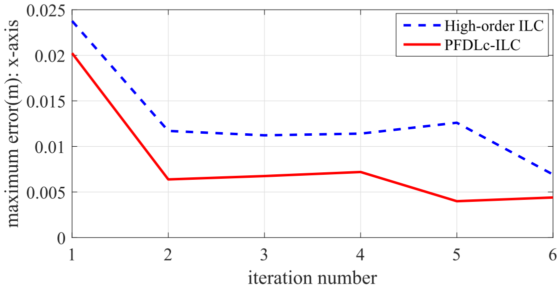


Fig. 14. Learning performances of high-order ILC and PFDLc-ILC on x-axis (Case 2).

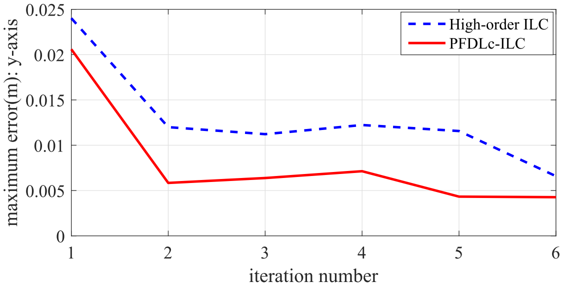


Fig. 15. Learning performances of high-order ILC and PFDLc-ILC on y-axis (Case 2).

Case 2: The iteration-varying desired outputs are $y_{d,1}(t,j) = y_{d,1}(t) + d(t,j)$ and $y_{d,2}(t,j) = y_{d,2}(t) + d(t,j)$, where $d(t,j)$ is a random number with the magnitude of 0.001 m for any time instant t and iteration number j, $t = 1, 2, \ldots, 12\,600$, $j = 1, 2, \ldots, 6$.

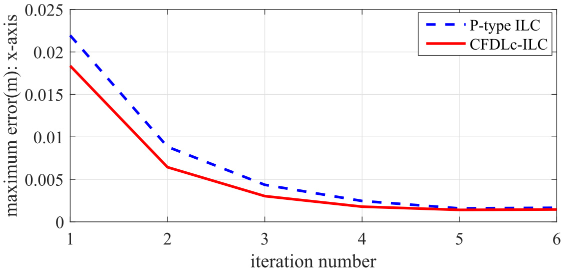


Fig. 16. Learning performances of P-type ILC and CFDLc-ILC on x-axis.

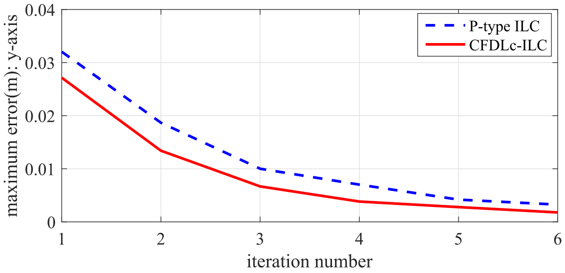


Fig. 17. Learning performances of P-type ILC and CFDLc-ILC on y-axis.

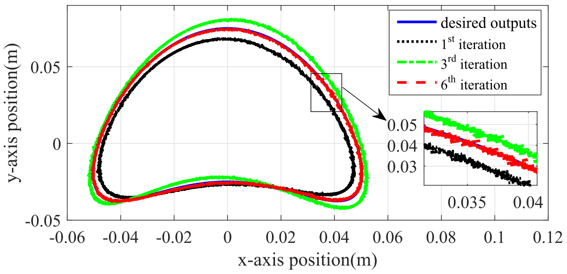


Fig. 18. Tracking trajectories of CFDLc-ILC.

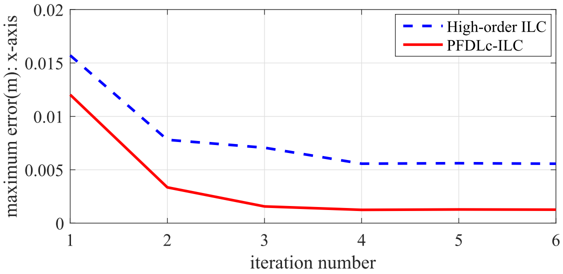


Fig. 19. Learning performances of high-order ILC and PFDLc-ILC on x-axis.

The experimental results for CFDLc-ILC and P-type ILC are presented in Figs. 12 and 13, which illustrate the better learning performance of CFDLc-ILC than P-type ILC in the case of the iteration-varying desired outputs. Figs. 14 and 15 display the better learning performance of PFDLc-ILC than high-order ILC. In general, the effectiveness of the learning performance on CFDLc-ILC and PFDLc-ILC is demonstrated from Figs. 12–15. Besides, the maximum errors shown in Figs. 12–15 would not converge to zero due to the existence of $d(t,j)$, which is demonstrated by Theorem 4.

2) Cardioid: In the experiment, the sampling time is still 0.001 s, and the running time is set as 6.3 s in each iteration (that is, $T = 6300$). The sensing noise is generated by the random number $d(t,j)$ with magnitude of ± 0.001 m.

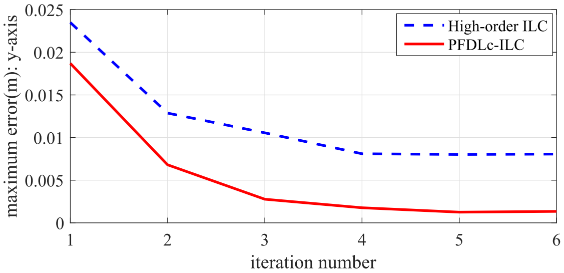


Fig. 20. Learning performances of high-order ILC and PFDLc-ILC on *y*-axis.

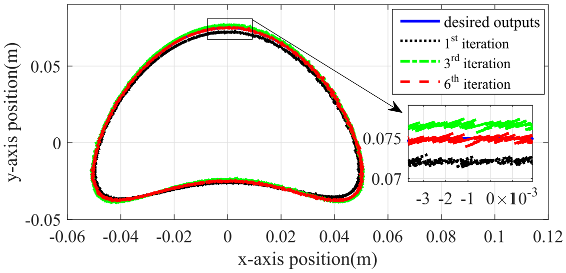


Fig. 21. Tracking trajectories of PFDLc-ILC.

The desired outputs are

$$\begin{cases} y_{d,1}(t) = 0.05\left(\sin\left(\dfrac{2\pi t}{6300}\right) + 0.5\sin\left(\dfrac{4\pi t}{6300}\right)\right) \\ y_{d,2}(t) = 0.05\left(\cos\left(\dfrac{2\pi t}{6300}\right) + 0.5\cos\left(\dfrac{4\pi t}{6300}\right)\right). \end{cases} \tag{60}$$

The experimental results for CFDLc-ILC, P-type ILC, PFDLc-ILC, and high-order ILC are presented in Figs. 16–21, where Figs. 18 and 21, respectively, show the tracking trajectories of CFDLc-ILC and PFDLc-ILC at the first, third, and sixth iterations. These results illustrate that, in the case of noisy sensing, the CFDLc-ILC and PFDLc-ILC have better learning performances than the P-type ILC and high-order ILC, respectively, and the tracking errors are reduced to within a small bound after six iterations of applying the proposed CFDLc-ILC and PFDLc-ILC.

V. Conclusion

In this article, two data-driven ILC approaches are proposed for a class of unknown nonlinear repetitive discrete-time MIMO systems. The learning controllers are constructed by extending the CFDL and PFDL methods to the multidimensional space in the iteration domain on an unknown ideal learning controller. The determined learning controllers are independent of the controlled plants, and the corresponding complexities of the learning controllers do not increase with that of the controlled plants. By virtue of the CFDL data model of the controlled nonlinear systems, the learning control gain updating algorithms are formulated using only the measured I/O data of the controlled plants through the steepest descent method. The stability and convergence of the two data-driven ILC approaches are achieved theoretically under a generalized Lipschitz condition. Finally, comparative analyses with simulated and experimental results are provided to verify the effectiveness of the two data-driven ILC approaches. In future work, it is worth investigating the design and analysis of robust data-driven iterative learning control approaches, for dealing with noisy sensing, and further investigating the case of more general DL methods for the learning controller design in order to deal with more complex nonlinear systems.

References

[1] A. Steinhauser and J. Swevers, "An efficient iterative learning approach to time-optimal path tracking for industrial robots," *IEEE Trans Ind. Informat.*, vol. 14, no. 11, pp. 5200–5207, Nov. 2018.

[2] Y. M. Zhao, Y. Lin, F. Xi, and S. Guo, "Calibration-based iterative learning control for path tracking of industrial robots," *IEEE Trans. Ind. Electron.*, vol. 62, no. 5, pp. 2921–2929, May 2015.

[3] R. Chi, Z. Hou, S. Jin, and B. Huang, "Computationally efficient data-driven higher order optimal iterative learning control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5971–5980, Dec. 2018.

[4] S. Mishra, J. Coaplen, and M. Tomizuka, "Precision positioning of wafer scanners segmented iterative learning control for nonrepetitive disturbances," *IEEE Control Syst. Mag.*, vol. 27, no. 4, pp. 20–25, Jul. 2007.

[5] M.-B. Radac, R.-E. Precup, E. M. Petriu, S. Preitl, and C.-A. Dragos, "Data-driven reference trajectory tracking algorithm and experimental validation," *IEEE Trans Ind. Informat.*, vol. 9, no. 4, pp. 2327–2336, Nov. 2013.

[6] W. Xiong, X. Yu, Y. Chen, and J. Gao, "Quantized iterative learning consensus tracking of digital networks with limited information communication," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1473–1480, Jun. 2017.

[7] Z. Hou, J. Yan, J.-X. Xu, and Z. Li, "Modified iterative-learning-control-based ramp metering strategies for freeway traffic control with iteration-dependent factors," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 606–618, Jun. 2012.

[8] H. Sun, Z. Hou, and D. Li, "Coordinated iterative learning control schemes for train trajectory tracking with overspeed protection," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 323–333, Apr. 2013.

[9] Y. Wang, Z. Hou, and X. Li, "A novel automatic train operation algorithm based on iterative learning control theory," in *Proc. IEEE Int. Conf. Service Oper. Logistics, Informat.*, Beijing, China, 2008, pp. 1766–1770.

[10] H. Ji, Z. Hou, and R. Zhang, "Adaptive iterative learning control for high-speed trains with unknown speed delays and input saturations," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 260–273, Jan. 2016.

[11] Q. Yu, Z. Hou, and J.-X. Xu, "D-type ILC based dynamic modeling and norm optimal ILC for high-speed trains," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 2, pp. 652–663, Mar. 2018.

[12] C.-J. Chien, "A discrete iterative learning control for a class of nonlinear time-varying systems," *IEEE Trans. Autom. Control*, vol. 43, no. 5, pp. 748–752, May 1998.

[13] D. Meng and K. L. Moore, "Contraction mapping-based robust convergence of iterative learning control with uncertain, locally Lipschitz nonlinearity," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 2, pp. 442–454, Feb. 2020.

[14] J.-X. Xu, Y. Tan, and T.-H. Lee, "Iterative learning control design based on composite energy function with input saturation," *Automatica*, vol. 40, no. 8, pp. 1371–1377, Aug. 2004.

[15] W. He, T. Meng, D. Huang, and X. Li, "Adaptive boundary iterative learning control for an Euler–Bernoulli beam system with input constraint," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1539–1549, May 2018.

[16] N. Amann, D. H. Owens, and E. Rogers, "Iterative learning control for discrete-time systems with exponential rate of convergence," *IEE Proc.-Control Theory Appl.*, vol. 143, no. 2, pp. 217–224, Mar. 1996.

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

[18] R. Chi, Z. Hou, S. Jin, and D. Wang, "Improved data-driven optimal TILC using time-varying input signals," *J. Process Control*, vol. 24, no. 12, pp. 78–85, Dec. 2014.

[19] R.-H. Chi and Z.-S. Hou, "Dual-stage optimal iterative learning control for nonlinear non-affine discrete-time systems," *Acta Automat. Sinica*, vol. 33, no. 10, pp. 1061–1065, Oct. 2007.

[20] Z. S. Hou and S. T. Jin, *Model Free Adaptive Control: Theory and Applications*. New York, NY, USA: CRC Press, 2013.

[21] J. X. Xu and Y. Tan, *Linear and Nonlinear Iterative Learning Control*. Berlin, Germany: Springer-Verlag, 2003.

[22] Z. S. Hou, “Parameter identication, adaptive control and model-free learning adaptive control for nonlinear systems,” Ph.D. dissertation, Dept. Autom. Control, Northeastern Univ., Shenyang, China, 1994.

[23] Z. S. Hou and S. S. Xiong, “On model free adaptive control and its stability analysis,” *IEEE Trans. Autom. Control*, vol. 64, no. 11, pp. 4555–4569, Nov. 2019.

[24] D. Xu, B. Jiang, and P. Shi, “A novel model-free adaptive control design for multivariable industrial processes,” *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6391–6398, Nov. 2014.

[25] Z. Hou, S. Liu, and T. Tian, “Lazy-learning-based data-driven model-free adaptive predictive control for a class of discrete-time nonlinear systems,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 8, pp. 1914–1928, Aug. 2017.

[26] X. Wang, X. Li, J. Wang, X. Fang, and X. Zhu, “Data-driven model-free adaptive sliding mode control for the multi degree-of-freedom robotic exoskeleton,” *Inf. Sci.*, vol. 327, pp. 246–257, Jan. 2016.

[27] Z.-H. Pang, G.-P. Liu, D. Zhou, and D. Sun, “Data-based predictive control for networked nonlinear systems with network-induced delay and packet dropout,” *IEEE Trans. Ind. Electron.*, vol. 63, no. 2, pp. 1249–1257, Feb. 2016.

[28] C. Lu, Y. Zhao, L. Tu, K. Men, and Y. Han, “Wide-area power system stabiliser based on model-free adaptive control,” *IET Control Theory Appl.*, vol. 9, no. 13, pp. 1996–2007, Aug. 2015.

[29] Z. Hou, R. Chi, and H. Gao, “An overview of dynamic-linearization-based data-driven control and applications,” *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4076–4090, May 2017.

[30] B. Yao and M. Tomizuka, “Adaptive robust control of MIMO nonlinear systems in semi-strict feedback forms,” *Automatica*, vol. 37, no. 9, pp. 1305–1321, Sep. 2001.

[31] X. Yu, Z. Hou, and C. Yin, “Iterative learning control for discrete-time nonlinear systems based on adaptive tuning of 2D learning gain,” in *Proc. 36th Chin. Control Conf. (CCC)*, Dalian, China, Jul. 2017, pp. 3581–3586.

[32] Z. Hou, X. Yu, and C. Yin, “A data-driven iterative learning control framework based on controller dynamic linearization,” in *Proc. Annu. Amer. Control Conf. (ACC)*, Milwaukee, WI, USA, Jun. 2018, pp. 5588–5593.

[33] Y. Zhu and Z. Hou, “Data-driven MFAC for a class of discrete-time nonlinear systems with RBFNN,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 1013–1020, May 2014.

[34] Z. Hou and Y. Zhu, “Controller-dynamic-linearization-based model free adaptive control for discrete-time nonlinear systems,” *IEEE Trans Ind. Informat.*, vol. 9, no. 4, pp. 2301–2309, Nov. 2013.

[35] J.-X. Xu and Y. Tan, “On the P-type and Newton-type ILC schemes for dynamic systems with non-affine-in-input factors,” *Automatica*, vol. 38, no. 7, pp. 1237–1242, Jul. 2002.

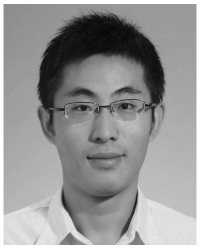
[36] Z. Hou and S. Jin, “Data-driven model-free adaptive control for a class of MIMO nonlinear discrete-time systems,” *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 2173–2188, Dec. 2011.

[37] D. Shen, W. Zhang, Y. Wang, and C.-J. Chien, “On almost sure and mean square convergence of P-type ILC under randomly varying iteration lengths,” *Automatica*, vol. 63, pp. 359–365, Jan. 2016.

[38] B. Huang and R. Kadali, *Dynamic Modeling, Predictive Control and Performance Monitoring: A Data-Driven Subspace Approach*. New York, NY, USA: Springer-Verlag, 2005.

[39] G. Bontempi and S. Ben Taieb, “Conditionally dependent strategies for multiple-step-ahead prediction in local learning,” *Int. J. Forecasting*, vol. 27, no. 3, pp. 689–699, Jul. 2011.

[40] X. Bu, Z. Hou, F. Yu, and F. Wang, “Robust model free adaptive control with measurement disturbance,” *IET Control Theory Appl.*, vol. 6, no. 9, pp. 1288–1296, 2012.



Xian Yu received the B.S. degree in mechanical engineering and automation and the M.S. degree in mechatronic engineering from Guangxi University, Nanning, China, in 2012 and 2015, respectively. He is currently pursuing the Ph.D. degree with the Advanced Control Systems Laboratory, Beijing Jiaotong University, Beijing, China.

He is currently a Visiting Researcher with the KIOS Research and Innovation Center of Excellence, University of Cyprus, Nicosia, Cyprus. His current research interests include iterative learning control and data-driven control and their industrial applications.

Zhongsheng Hou (Fellow, IEEE) received the bachelor’s and master’s degrees both in applied mathematics from the Jilin University of Technology, China, in 1983 and 1988, respectively, and the Ph.D. degree in control theory and applications from Northeastern University, Shenyang, China, in 1994.

From 1997 to 2018, he was with Beijing Jiaotong University, Beijing, China, where he was a Distinguished Professor and the Head of the Department of Automatic Control. He is currently the Chair Professor with the School of Automation, Qingdao University, Qingdao, China. He has authored or coauthored over 200 peer-reviewed journal articles and over 180 articles in prestigious conference proceedings. He has authored a monograph *Model-Free Adaptive Control: Theory and Applications* (CRC Press, 2013). His research interests are in data-driven control, model-free adaptive control, learning control, and intelligent transportation systems.

Prof. Hou is the Founding Director of the technical committee on Data-Driven Control, Learning and Optimization, Chinese Association of Automation (CAA). He is a fellow of CAA. He is an International Federation of Automatic Control (IFAC) Technical Committee Member of both Adaptive and Learning Systems and Transportation Systems. He has served as a TPC member for many conferences. He has served as an Associate Editor and a Guest Editor for a few international journals and Chinese journals, including the IEEE TRANSACTIONS ON NEURAL NETWORKS in 2011 and the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS in 2017.

Marios M. Polycarpou (Fellow, IEEE) received the B.A. degree in computer science and the B.Sc. degree in electrical engineering from Rice University, Houston, TX, USA, both in 1987, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 1992.

He is currently a Professor of electrical and computer engineering and the Director of the KIOS Research and Innovation Center of Excellence, University of Cyprus, Nicosia, Cyprus. He has authored or coauthored over 350 articles in refereed journals, edited books, and refereed conference proceedings. He has coauthored seven books and six patents. He has participated in over 70 research projects/grants, funded by several agencies and industry in Europe and the United States, including the prestigious European Research Council Advanced Grant. His teaching and research interests are in intelligent systems and networks, fault diagnosis, computational intelligence, and critical infrastructure systems.

Prof. Polycarpou is a fellow of IFAC and a Founding Member of the Cyprus Academy of Sciences, Letters, and Arts. He was a recipient of the 2016 IEEE Neural Networks Pioneer Award. He has served as the President of the IEEE Computational Intelligence Society from 2012 to 2013, the President of the European Control Association from 2017 to 2019. He has served as the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS from 2004 to 2010.

Li Duan received the M.S. degree in communication engineer from Chongqing Jiaotong University, Chongqing, China, in 2017. She is currently pursuing the B.S. degree with the Advanced Control Systems Laboratory, Beijing Jiaotong University, Beijing, China.

Her current research interests include data-driven control and iterative learning control and their industrial applications.