AIMS Mathematics, 11(5): 12687–12704.
DOI: 10.3934/math.2026522
Received: 13 February 2026
Revised: 09 April 2026
Accepted: 15 April 2026
Published: 08 May 2026

https://www.aimspress.com/journal/Math

Research article

Data-driven iterative learning consensus control for linear parabolic distributed parameter multi-agent systems

Lanlan Liu[1] **and Xisheng Dai**[2,3]*

[1] School of Business, Chongqing College of Finance and Economics, Chongqing 402160, China
[2] School of Computer Science, Guangdong University of Finance, Guangzhou 510520, China
[3] School of Automation, Guangxi University of Science and Technology, Liuzhou 545616, China

* **Correspondence:** Email: mathdxs@163.com.

Abstract: To address the consensus control challenge in model-unknown linear parabolic distributed parameter multi-agent systems, this paper proposes a novel data-driven iterative learning control (DDILC) method. A forward difference scheme is adopted to discretize the spatiotemporal dynamics, thereby establishing a data-driven input-output model that eliminates reliance on precise mathematical formulations. Neural network approximates the unknown linear operators, and an error-driven iterative learning law updates the network weights online to compensate for unmodeled dynamics and uncertainties. Rigorous Lyapunov-based analysis verifies the boundedness of the weight errors and the exponential convergence of the consensus tracking errors. Numerical simulations with one virtual leader and four followers confirm that all agents achieve precise consensus tracking within 20 iterations, with errors converging to within 0.01. This method overcomes the limitations of traditional model-based control, offering an efficient solution for consensus control in engineering applications such as UAV swarms and intelligent industrial collaboration.

Keywords: data-driven iterative learning control; distributed parameter multi-agent systems; neural network; consensus control
Mathematics Subject Classification: 93A16, 93C20, 68T07

1. Introduction

The concept of multi-agent systems comes from the observation of activities such as bird flocks, fish schools, and ant colonies migrating, avoiding enemies, and foraging [1–3]. Inspired by nature, multi-agent systems composed of multiple agents have received extensive attention and application in fields such as swarms [4], intelligent transportation [5], industrial collaboration [6], and environmental monitoring [7]. In such application contexts, individual agents need to collaborate to

complete complex tasks, requiring them to move a common goal, which has given rise to research on multi-agent cooperative control [8, 9]. Among them, consensus control is one of the core directions [2].

Reference [10] investigated multi-agent systems affected by model uncertainties and proposed a robust control-based consensus protocol design method, where controller parameters were solved using a cone complementarity linearization algorithm. In [11], a model-free Q-learning algorithm used fuzzy logic systems to learn the optimal switching policy for discrete-time switched nonlinear systems while considering switching costs. Reference [12] proposed a data-driven method for constructing a new protocol by introducing a one-step super expected signal to solve the tracking consistency problem of a class of unknown nonlinear multi-agent systems. In [13], an effective distributed control algorithm with general state set constraints was proposed to address the state consensus problem of multi-agent systems.

At present, research on consensus control of multi-agent systems mainly focuses on systems described by ordinary differential equations, considering their convergence in the time domain [14–17]. Specifically, reference [18] proposed a phased algorithm that combines symbiotic organisms search with an improved multi-agent consensus algorithm to optimize energy dispatch in islanded multi-microgrids, achieving high renewable energy utilization. Reference [19] proposed a distributed model predictive control algorithm for linear quadratic optimal consensus of discrete-time multi-agent systems, establishing a consensus condition that depends only on local parameters. In [20], a dynamic event-triggered adaptive control strategy was developed for output consensus of nonlinear multi-agent systems, employing recursive sliding-modes and nonlinear gain functions to improve control performance. However, multi-agent systems have temporal and spatial evolution characteristics during their movement, requiring simultaneous consideration of agent states in both domains. In view of this issue, several scholars have conducted relevant explorations. In [21], a consensus-based iterative learning control protocol was proposed for a class of multi-agent systems with distributed parameter models, leveraging a network topology framework and nearest neighbor knowledge. Reference [22] investigated the consensus control problem for multi-agent systems with time-delayed distributed parameter models. By leveraging neighboring agent interactions and explicitly addressing time delays, a distributed P-type iterative learning control protocol was proposed. Reference [23] investigated iterative learning consensus control schemes for a class of multi-agent systems characterized by time-delayed distributed parameter models [24].

However, existing research predominantly designs control algorithms based on precise model assumptions. Reference [25, 26] provided a comprehensive overview of dynamic-linearization-based data-driven control methods, highlighting their ability to handle unknown nonlinear systems without requiring precise mathematical models. Reference [27] investigated iterative learning consensus control for distributed parameter multi-agent systems with time delays, demonstrating the effectiveness of model-based approaches under specific structural assumptions. In reality, multi-agent systems are inherently complex systems due to their structural and operational characteristics, and they often face challenges in practical applications such as unknown model parameters, complex dynamics, and agent heterogeneity [28]. Traditional model-based control methods struggle with precise modeling and face challenges in high-cost, low-reliability global information acquisition within distributed environments [29, 30]. In contrast, data-driven iterative learning control methods, renowned for their ability to control systems without requiring precise mathematical models,

effectively address the challenges of complex multi-agent environments—specifically overcoming difficult modeling, information constraints, and intricate dynamics [31]. Therefore, this paper proposes a data-driven control method for distributed parameter multi-agent systems with unknown parametric models [32–34]. A data-driven model capturing the input-output relationship of the system is established, and a mapping from desired output to desired input is constructed based on neural network. An optimal iterative learning algorithm for weight updating is derived to adjust the system input and address the challenge of unknown parameters. The efficacy of the proposed strategy is substantiated through simulations, confirming that each agent attains precise trajectory following over a finite time window.

The remainder of this paper is organized as follows. Section 2 presents the problem formulation and preliminaries, including the discrete distributed parameter system model and the control objective. Section 3 details the design of the scheme, incorporating RBF neural networks and the weight update law, followed by theoretical analysis of the boundedness of weight estimation errors and the convergence of tracking errors. Section 4 provides numerical simulations to validate the effectiveness of the proposed method. Section 5 discusses the results and practical implications. Finally, Section 6 concludes the paper and outlines future research directions.

The main contributions and innovations of this article are:

(1) The proposed approach considers the scenario where individuals are distributed parameter systems in multi-agent systems, and based on this, considers the convergence control of multi-agent systems. This extends the application of consensus control from ordinary differential equation models to distributed parameter models, enabling simultaneous consideration of temporal and spatial evolution characteristics.
(2) Considering the problem of unknown model parameters, applying data-driven control methods to multi-agent distributed parameter systems has a broader application background. Compared with existing methods [21–23], the proposed approach does not require precise mathematical models of the system, making it more applicable to real-world scenarios where accurate modeling is difficult or impossible.
(3) In designing the algorithm, a time-varying neural network is introduced to update the input of the weight adjustment system for the hidden layer and output layer, and an iterative learning algorithm is designed for the weights, which enhances the system's adaptability to dynamic uncertainties while ensuring control accuracy. The data-driven nature allows for online adaptation to changing system dynamics, and the iterative learning framework ensures convergence even with unknown parameters and disturbances. This aspect also represents a key challenge addressed in this research.

Notations: In this paper, required mathematical notations and knowledge are as follows.

(1) Mathematical notations. $\mathbb{R}^X$ denotes the X-dimensional Euclidean space, and $\mathbb{R}^{X\times X}$ denotes the set of all $X \times X$ real matrices. Here, $\mathbb{R}$ represents the set of real numbers. The main symbols are defined as follows:
(2) Communication topology. Let the weighted directed graph among agents be denoted as $\mathcal{G} = (\bar{V}, \bar{E}, \bar{A})$, where the node set is $\bar{V} = \{1, 2, 3, \ldots, N\}$ and N indicates that the multi-agent system consists of N agents. The edge set is $\bar{E} \subseteq \bar{V} \times \bar{V}$, and the matrix $\bar{A}$ is the adjacency matrix of $\mathcal{G}$. Nodes j and i correspond to the j-th and i-th agents, respectively. If there exists an edge from

node j to node i, i.e., the ordered pair $(j,i) \in \bar{E}$, then agent i can receive information from agent j; in this case, node j is considered a parent node of node i, and node i is a child node of node j. The set of neighboring agents of agent i is defined as $\mathcal{N}_i = \{j \in \bar{V} \mid (j,i) \in \bar{E}\}$. The weighted adjacency matrix of $\mathcal{G}$ is $\bar{A} = (a_{j,i}) \in \mathbb{R}^{N\times N}$, where $a_{j,i} = 1$ if $(j,i) \in \bar{E}$, and $a_{j,i} = 0$ otherwise. The diagonal entries satisfy $a_{i,i} = 0$ for all i.

(3) Neural network notation. An RBF neural network is adopted for approximation, with X neurons in the hidden layer. $\phi(\chi) = [\phi_1(\chi), \ldots, \phi_X(\chi)]^T \in \mathbb{R}^X$ is the hidden layer output vector, where $\phi_p(\chi) = \exp\left(-\frac{\|\chi - c_p\|^2}{2\sigma_p^2}\right)$ is the Gaussian activation output of the p-th neuron. $W^k \in \mathbb{R}^{X\times X}$ denotes the output layer weight matrix at the k-th iteration. W^d denotes the ideal weight matrix that satisfies the optimal approximation condition.

(4) Error and performance index. The relative formation tracking error is defined as $e_{ij}^k(t) = y_{ij}^d(t) - y_{ij}^k(t)$. The weight increment is denoted by $\Delta W^k = W^k - W^{k-1}$. During the optimization process, the trace operation $\mathrm{tr}(\cdot)$ is employed to convert the matrix weight into a scalar energy, i.e., $\frac{\beta}{2}\sum_{m,n}(\Delta W_{kmn})^2 = \frac{\beta}{2}\|\Delta W^k\|_F^2$, where $\|\cdot\|_F$ denotes the Frobenius norm.

2. Preliminaries and problem statement

Consider a discrete linear distributed parameter multi-agent system that is composed of N agents of the same class based on spatiotemporal discretization and repeatedly runs within a time interval. The dynamic equation of the i-th agent is

$$\begin{cases} \Delta_2 w_i(x,t) = a\Delta_1^2 w_i(x-1,t) + b w_i(x,t) + c u_i(x,t), \\ y_i(x,t) = d w_i(x,t) + u_i(x,t). \end{cases} \tag{2.1}$$

Them, x and t are the discrete variables of space and time, respectively. $w_i(t) \in \mathbb{R}^X$ and $y_i(t) \in \mathbb{R}^T$ are spatial and temporal variables, respectively. a, b, c and d are system parameters that are unknown but bounded.

Assumption: All agents belong to the same class, i.e., they share identical system structure parameters.

To discretize the continuous distributed parameter system, we adopt the forward difference scheme. The difference scheme is defined as

$$\begin{cases} \Delta_2 w_i(x,t) = w_i(x,t+1) - w_i(x,t), \\ \Delta_1^2 w_i(x,t) = w_i(x+1,t) - 2w_i(x,t) + w_i(x-1,t). \end{cases} \tag{2.2}$$

The most widely used numerical method for solving definite solution problems for partial differential equations is the finite difference method. Its basic idea is a discrete numerical analysis approximation approach for differential equations and definite solution conditions with continuous variables [22, 32]. In this paper, the forward Euler scheme is adopted to discretize System (2.1). This scheme is conditionally stable, and the stability condition is expressed in terms of the step-size ratio as

$$\frac{a\Delta t}{(\Delta x)^2} \le \frac{1}{2},$$

where a is a parameter of the differential equation, Δt is the time step, and Δx is the spatial step. Under this condition, the discretization is numerically stable. Substituting the difference scheme into

the system, we can obtain

$$w_i(x, t+1) = aw_i(x+1, t) + (1 - 2a + b)w_i(x, t) + aw_i(x-1, t) + cu_i(x, t).$$

When $x = 1$,

$$w_i(1, t+1) = aw_i(2, t) + (1 - 2a + b)w_i(1, t) + aw_i(0, t) + cu_i(1, t)$$

When $x = 2$,

$$w_i(2, t+1) = aw_i(3, t) + (1 - 2a + b)w_i(2, t) + aw_i(1, t) + cu_i(2, t)$$

When $x = 3$,

$$w_i(3, t+1) = aw_i(4, t) + (1 - 2a + b)w_i(3, t) + aw_i(2, t) + cu_i(3, t)$$

...

When $x = X - 1$,

$$w_i(X-1, t+1) = aw_i(X, t) + (1 - 2a + b)w_i(X-1, t) + aw_i(X-2, t) + cu_i(X-1, t)$$

When $x = X$,

$$w_i(X, t+1) = aw_i(X+1, t) + (1 - 2a + b)w_i(X, t) + aw_i(X-1, t) + cu_i(X, t).$$

$$\begin{bmatrix} w_i(1,t+1) \\ w_i(2,t+1) \\ \vdots \\ w_i(X-1,t+1) \\ w_i(X,t+1) \end{bmatrix} = \begin{bmatrix} 1-2a+b & a & 0 & 0 & 0 \\ a & 1-2a+b & a & 0 & 0 \\ 0 & a & 1-2a+b & a & 0 \\ 0 & 0 & a & 1-2a+b & a \\ 0 & 0 & 0 & a & 1-2a+b \end{bmatrix} \cdot \begin{bmatrix} w_i(1,t) \\ w_i(2,t) \\ \vdots \\ w_i(X-1,t) \\ w_i(X,t) \end{bmatrix}$$

$$+ \begin{bmatrix} aw_i(0,t) \\ 0 \\ \vdots \\ 0 \\ aw_i(X+1,t) \end{bmatrix} + c \begin{bmatrix} u_i(1,t) \\ u_i(2,t) \\ \vdots \\ u_i(X-1,t) \\ u_i(X,t) \end{bmatrix}.$$

The initial and boundary values are 0, i.e., $w_i(0, t) = w_i(X + 1, t) = 0$. Physically, this Dirichlet boundary condition fixes the agent's state to zero at the spatial boundaries. This is a common constraint that confines the system to a bounded spatial domain. For example, in UAV formation control, it ensures that agents maintain a stationary state when approaching the mission boundary. Consequently, the system matrix takes the following tridiagonal form, and the system can be written in vector form as

$$\begin{cases} x_i(t+1) = Ax_i(t) + cu_i(t), \\ y_i(t) = dx_i(t) + u_i(t), \end{cases} \tag{2.3}$$

where

$$x_i(t+1) = [w_i(1,t+1), w_i(2,t+1), \ldots, w_i(X,t+1)]^T \in \mathbb{R}^X,$$
$$y_i(t+1) = [y_i(1,t+1), y_i(2,t+1), \ldots, y_i(X,t+1)]^T \in \mathbb{R}^X,$$
$$u_i(t+1) = [u_i(1,t+1), u_i(2,t+1), \ldots, u_i(X,t+1)]^T \in \mathbb{R}^X,$$

and

$$A = \begin{bmatrix} 1-2a+b & a & & & & \\ a & 1-2a+b & a & & & \\ & \ddots & \ddots & \ddots & & \\ & & a & 1-2a+b & a & \\ & & & a & 1-2a+b \end{bmatrix} \in \mathbb{R}^{X\times X}.$$

The system equation in the k-th iteration process can be expressed as

$$\begin{cases} x_i^k(t+1) = Ax_i^k(t) + cu_i^k(t), \\ y_i^k(t) = dx_i^k(t) + u_i^k(t). \end{cases} \tag{2.4}$$

Here, $x_i^k(t) \in \mathbb{R}^X$ denotes the X-dimensional state vector of the i-th agent at the k-th iteration; $u_i^k(t) \in \mathbb{R}^X$ represents the X-dimensional control input of the i-th agent at the k-th iteration; $y_i^k(t) \in \mathbb{R}^X$ is the system output, $A \in \mathbb{R}^{X\times X}$ is the state transition matrix and $c, d \in \mathbb{R}$ are system constants.

By induction, the output of the i-th agent after the k-th iteration is derived as

$$y_i^k(t) = dA_i^t x_i^0 + d_i c_i \sum_{\tau=0}^{t-1} A_i^{t-1-\tau} u_i^k(\tau) + u_i^k(t). \tag{2.5}$$

It can be abbreviated as

$$y_i^k(t) = dA_i^t x_i^0 + Gu_i^k(t), \tag{2.6}$$

where G is an unknown but bounded linear operator.

The output of the j-th agent after the k-th iteration can be expressed as

$$y_j^k(t) = dA_j^t x_j^0 + Gu_j^k(t). \tag{2.7}$$

In the multi-agent system topology, assume that the i-th agent is a child and the j-th agent is a parent, meaning information flows from j to i. If the unknown system parameters are identical for all agents and all agents share the same initial state, then it follows that

$$y_i^k(t) - y_j^k(t) = Gu_i^k(t) - Gu_j^k(t). \tag{2.8}$$

By defining

$$y_{ij}(t) = y_i(t) - y_j(t), \quad u_{ij}(t) = u_i(t) - u_j(t),$$

the above equation can be written as

$$y_{ij}^k(t) = Gu_{ij}^k(t). \tag{2.9}$$

The desired relative formation error is defined as $y_{ij}^d(t)$. The control objective is to regulate the system inputs so that the parent and child agents follow the desired formation trajectory within a finite time. The formation error $e_{ij}^k(t)$ is then defined as the difference between the desired $y_{ij}^d(t)$ and the actual relative formation: $\lim_{k\to\infty} e_{ij}^k(t) = 0$.

3. Data-driven iterative learning control design

An RBF neural network is introduced to learn the unknown parameter matrix G, thereby establishing a mapping from the desired output $y_{ij}^d(t)$ to the desired input. The number of hidden layer neurons is defined as X. After training from the input layer to the hidden layer, the output of the hidden layer $\phi(\chi)$ can be written as $\phi(\chi) = [\phi_1(\chi), \phi_2(\chi), \cdots, \phi_X(\chi)]^{\mathrm{T}} \in \mathbb{R}^X$. The variation of $\phi(\chi)$ depends on the input layer. Here, the unknown parameter matrix G and the system input $u_{ij}^k(t)$ are treated as an integrated entity $\phi(\chi)$, which serves as the basis function vector. This integrated entity can be obtained through neural network training.

$$y_{ij}^k(t) = Gu_{ij}^k(t) = (W_{ij}^k(t))^T \phi(\chi). \tag{3.1}$$

Define the ideal weight as W_{ij}^d, satisfying $y_{ij}^d(t) = (W_{ij}^d(t))^T \phi(\chi)$. The error equation can then be written as

$$e_{ij}^k(t) = y_{ij}^d(t) - y_{ij}^k(t) = (\widetilde{W}_{ij}^k(t))^T \phi(\chi), \tag{3.2}$$

where $\widetilde{W}_{ij}^k(t) = W_{ij}^d(t) - W_{ij}^k(t)$ represents the weight estimation error.

The iterative error of adjustable weights can be expressed as $\Delta W_{ij}^k(t) = W_{ij}^k(t) - W_{ij}^{k-1}(t)$.

Define the performance index function for the weights as

$$J(W_{ij}^k(t)) = \frac{\lambda}{2}(e_{ij}^k(t))^T(e_{ij}^k(t)) + \frac{\beta}{2}\mathrm{tr}(\Delta W_{ij}^k(t))^T(\Delta W_{ij}^k(t)), \tag{3.3}$$

where $\lambda, \beta > 0$. Since W_{ij}^k is a matrix, the trace operation $\mathrm{tr}(\cdot)$ is utilized to scalarize it, resulting in a scalar quantity equivalent to $\frac{\beta}{2}\sum_{m,n}(\Delta W_{k,mn})^2$.

By taking the partial derivative of the performance index W_{ij}^k with respect to the weights and simplifying, we obtain the error-driven iterative learning update law for the weights, which serves as the controller of the system

$$W_{ij}^k(t) = W_{ij}^{k-1}(t) + \frac{\lambda\phi(\chi)(e_{ij}^{k-1}(t))^T}{\beta + \|\phi(\chi)\|^2}. \tag{3.4}$$

3.1. Boundedness of neural network weight estimates

Using the ideal weight value W_{ij}^d both sides of (3.4), we have

$$\widetilde{W}_{ij}^k(t) = \widetilde{W}_{ij}^{k-1}(t) - \frac{\lambda\phi(\chi)(e_{ij}^{k-1}(t))^T}{\beta + \|\phi(\chi)\|^2}. \tag{3.5}$$

Then

$$\widetilde{W}_{ij}^k(t) - \widetilde{W}_{ij}^{k-1}(t) = \widetilde{W}_{ij}^{k-1}(t) - \frac{\lambda\phi(\chi)(e_{ij}^{k-1}(t))^T}{\beta + \|\phi(\chi)\|^2}. \tag{3.6}$$

Define the energy function

$$V_k = \frac{1}{2}\|\widetilde{W}_{ij}^k\|_F^2, \tag{3.7}$$

where $\|\cdot\|_F$ is the Frobenius norm.

Then we have

$$V_k = \frac{1}{2}\|\widetilde{W}_{ij}^{k-1} - \Delta W_{ij}^k\|_F^2 = \frac{1}{2}\|\widetilde{W}_{ij}^{k-1}\|_F^2 + \frac{1}{2}\|\Delta W_{ij}^k\|_F^2 - \mathrm{tr}((\widetilde{W}_{ij}^{k-1})^T \Delta W_{ij}^k). \tag{3.8}$$

Define $\Delta V_k = V_k - V_{k-1}$, and we have

$$\Delta V_k = \frac{1}{2}(\|\widetilde{W}_{ij}^k\|_F^2 - \|\widetilde{W}_{ij}^{k-1}\|_F^2). \tag{3.9}$$

Substituting (3.8) into (3.9),

$$\Delta V_k = \frac{1}{2}\|\Delta W_{ij}^k\|_F^2 - \mathrm{tr}((\widetilde{W}_{ij}^{k-1})^T \Delta W_{ij}^k). \tag{3.10}$$

According to (3.4), we have

$$\Delta W_{ij}^k = \frac{\lambda \phi(\chi)(e_{ij}^{k-1})^T}{\beta + \|\phi(\chi)\|^2}. \tag{3.11}$$

Remark 1. *The numerical error caused by the forward difference scheme is uniformly bounded. According to the Lyapunov stability analysis and Frobenius norm derivation, such bounded errors affect only the final steady-state error magnitude but do not influence the exponential convergence property of the consensus tracking error along the iteration axis. Thus, the tracking error still converges exponentially to a small neighborhood of zero, whose radius is determined by the upper bound of the discretization error.*

Substituting (3.2) and (3.11) into (3.10)

$$\mathrm{tr}((\widetilde{W}_{ij}^{k-1})^T \Delta W_{ij}^k) = \frac{\lambda}{\beta + \|\phi(\chi)\|^2}\mathrm{tr}((\widetilde{W}_{ij}^{k-1})^T \phi(\chi)(e_{ij}^{k-1})^T) = \frac{\lambda}{\beta + \|\phi(\chi)\|^2}\|e_{ij}^{k-1}\|^2. \tag{3.12}$$

Since $\|\Delta W_{ij}^k\|_F^2 = \frac{\lambda^2\|\phi(\chi)\|^2\|e_{ij}^{k-1}\|^2}{(\beta+\|\phi(\chi)\|^2)^2}$, substituting into (3.10) and simplifying, we have

$$\Delta V_k = \frac{\lambda^2\|\phi(\chi)\|^2}{2(\beta + \|\phi(\chi)\|^2)^2}\|e_{ij}^{k-1}\|^2 - \frac{\lambda}{\beta + \|\phi(\chi)\|^2}\|e_{ij}^{k-1}\|^2. \tag{3.13}$$

From Eq (3.2), the norm of the error at the $(k-1)$-th iteration can be expressed as

$$\|e_{ij}^{k-1}\|^2 = [(\widetilde{W}_{ij}^{k-1})^T \phi(\chi)]^T (\widetilde{W}_{ij}^{k-1})^T \phi(\chi) \le \|\phi(\chi)\|^2 \cdot \lambda_{\max}(\widetilde{W}_{ij}^{k-1}(\widetilde{W}_{ij}^{k-1})^T), \tag{3.14}$$

where $\lambda_{\max}$ is the maximum eigenvalue of the matrix. According to the properties of eigenvalues, we have

$$\lambda_{\max}(\widetilde{W}_{ij}^{k-1}(\widetilde{W}_{ij}^{k-1})^T) \le \|\widetilde{W}_{ij}^{k-1}\|_F^2. \tag{3.15}$$

Therefore,

$$\|e_{ij}^{k-1}\|^2 \le \|\phi(\chi)\|^2 \cdot \|\widetilde{W}_{ij}^{k-1}\|_F^2. \tag{3.16}$$

By substituting (3.16) into (3.15)) and simplifying it, we can obtain

$$\Delta V_k \le \frac{\lambda^2 \|\phi(\chi)\|^4 \left\|\tilde{W}_{ij}^{k-1}\right\|_F^2}{2(\beta + \|\phi(\chi)\|^2)^2} - \frac{\lambda \|\phi(\chi)\|^2 \left\|\tilde{W}_{ij}^{k-1}\right\|_F^2}{\beta + \|\phi(\chi)\|^2} \le \|\widetilde{W}_{ij}^{k-1}\|_F^2 \cdot \left[\frac{\lambda^2\|\phi(\chi)\|^4}{2(\beta + \|\phi(\chi)\|^2)^2} - \frac{\lambda\|\phi(\chi)\|^2}{\beta + \|\phi(\chi)\|^2}\right]. \tag{3.17}$$

Define

$$C = \frac{\lambda^2\|\phi(\chi)\|^4}{2(\beta+\|\phi(\chi)\|^2)^2} - \frac{\lambda\|\phi(\chi)\|^2}{\beta+\|\phi(\chi)\|^2} = \frac{\lambda\|\phi(\chi)\|^2}{\beta+\|\phi(\chi)\|^2}[\frac{\lambda\|\phi(\chi)\|^2}{2(\beta+\|\phi(\chi)\|^2)} - 1]. \tag{3.18}$$

Then (3.17) can be expressed as

$$\Delta V_k \leq C \cdot \|\widetilde{W}_{ij}^{k-1}\|_F^2. \tag{3.19}$$

According to the parameter definition, it can be inferred that when $\lambda < 2\left(1+\frac{\beta}{\|\phi(\chi)\|^2}\right)$, we have $C < 0$, which means the energy function decreases and the neural network weight estimation is bounded.

3.2. Convergence of error

The error at the k-th iteration is

$$e_{ij}^k(t) = y_d(t) - (W_{ij}^k(t))^T\phi(\chi). \tag{3.20}$$

The error at the $(k-1)$-th iteration is

$$e_{ij}^{k-1}(t) = y_d(t) - (W_{ij}^{k-1}(t))^T\phi(\chi). \tag{3.21}$$

Subtracting (3.21) from (3.20),

$$e_{ij}^k(t) = e_{ij}^{k-1}(t) + (W_{ij}^{k-1}(t))^T\phi(\chi) - (W_{ij}^k(t))^T\phi(\chi). \tag{3.22}$$

Extracting the total error ΔW_{ij}^k,

$$e_{ij}^k(t) = e_{ij}^{k-1}(t) - (W_{ij}^k(t) - W_{ij}^{k-1}(t))^T\phi(\chi). \tag{3.23}$$

According to the learning law (3.4), the above equation can be rewritten as

$$e_{ij}^k(t) = e_{ij}^{k-1}(t) - \frac{\lambda e_{ij}^{k-1}(t)\phi^T(\chi)}{\beta+\|\phi(\chi)\|^2}\phi(\chi) = \left(1 - \frac{\lambda\|\phi(\chi)\|^2}{\beta+\|\phi(\chi)\|^2}\right)e_{ij}^{k-1}(t). \tag{3.24}$$

Taking the norm of both sides of Eq (3.24)

$$\|e_{ij}^k(t)\| = \left|1 - \frac{\lambda\|\phi(\chi)\|^2}{\beta+\|\phi(\chi)\|^2}\right|\|e_{ij}^{k-1}(t)\|. \tag{3.25}$$

Given $\lambda, \beta > 0$, $\|\phi(\chi)\|^2 > 0$ and $\frac{\lambda\|\phi(\chi)\|^2}{\beta+\|\phi(\chi)\|^2} < \lambda$, if $\lambda < 2\left(1+\frac{\beta}{\|\phi(\chi)\|^2}\right)$ holds, then $e_{ij}^k(t)$ will exponentially converge to the zero vector along the iteration axis, i.e., $\lim_{k\to\infty} e_{ij}^k(t) = 0$.

From the above design and analysis of the controller, the main theorem of this paper are obtained.

Theorem 1. *For the discrete linear distributed parameter multi-agent system described in (2.1), assume that it satisfies the given assumption. Let the RBF neural network basis function vector be $\phi(\chi) \in \mathbb{R}^X$ with $\|\phi(\chi)\|^2 > 0$, and let the learning gains satisfy $\lambda > 0$, $\beta > 0$. If $\lambda < 2\left(1+\frac{\beta}{\|\phi(\chi)\|^2}\right)$, then, for any initial weight matrix $W_{ij}^0(t)$ and any initial state, the consensus tracking error $e_{ij}^k(t)$ defined in (3.2) converges to zero as iteration times k tend to infinity, i.e.*

$$\lim_{k\to\infty} e_{ij}^k(t) = 0. \tag{3.26}$$

4. Numerical simulation

To verify the effectiveness of the proposed algorithm, this paper considers a discrete-time linear distributed parameter multi-agent system with disturbances, consisting of one virtual leader and four followers. The communication topology of the multi-agent system is illustrated in Figure 1, where "0" denotes the parent agent and "1" the child agent. A chain-type directed topology $0 \to 1 \to 2 \to 3 \to 4$ is adopted, where only follower 1 directly receives information from the leader, and the remaining followers only interact with their immediate predecessor agents.

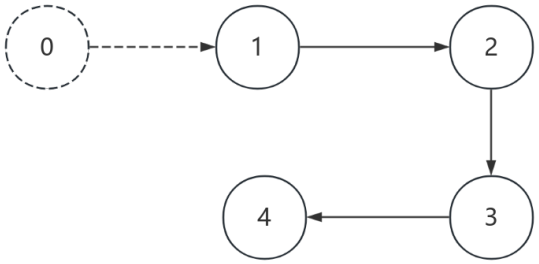

Figure 1. Topology structure of multi-agent system.

Following the common simulation methodology in related literature [21–23], a small-scale topology composed of a few agents from a large-scale multi-agent system is selected for verification. This approach is both reasonable and generalizable, because a complex multi-agent system with N agents can be decomposed into interconnected subsystems of arbitrary scale. Furthermore, the performance observed on small-scale subsystems can effectively reflect the algorithm's performance when applied to the full large-scale system.

The dynamic model of the i-th intelligent agent is

$$\begin{cases} \Delta_2 w_i(x,t) = 0.12\Delta_1^2 w_i(x-1,t) + 0.25 w_i(x,t) + 0.4 u_i(x,t), \\ y_i(x,t) = 0.7 w_i(x,t) + u_i(x,t). \end{cases} \tag{4.1}$$

Them, $w_i(x,t)$ is the system state of the i-th agent determined by position x and time t. $u_i(x,t)$ is the system input, and $y_i(x,t)$ is the system output, which is also influenced by position x and time t. Let a=0.12, b=0.25, c=0.4, and d=0.7.

For the spatiotemporal evolution characteristics of distributed parameter systems, the forward difference scheme is adopted to discretize the continuous partial differential equation. According to the stability criterion $\Delta x, /\Delta t^2 \leq 1/2$, the parameters are designed such that the spatial domain $x \in [0,1]$ is equally divided into 10 segments, with $X = 11$ spatial nodes and step size $\Delta t = 0.1$, and the time domain of total length 200 is equally divided into 200 steps, with time step $\tau = 1$, yielding a step ratio $\Delta x/\Delta t^2 = 0.5$ that satisfies the critical stability condition. The spatiotemporal nodes are partitioned into a regular discrete grid of size 11×200.

4.1. Simulation results under conventional trajectory

The output trajectory of the virtual leader is

$$y_0(x,t) = 2 \cdot \sin(\pi x) \cdot e^{-(x-0.5)^2} \cdot \cos\left(2\pi \frac{t-1}{T}\right) - 1.$$

Based on this trajectory, the four followers design differentiated relative formation trajectories that cover typical offset patterns. $y_{d1}(x,t) = y_0(x,t) + 0.15$, $y_{d2}(x,t) = y_0(x,t) + 0.1 \cdot \sin(\pi x)$, $y_{d3}(x,t) = y_0(x,t) + 0.1 \cdot \cos\left(2\pi\frac{t-1}{T}\right)$, and $y_{d4}(x,t) = y_0(x,t) + 0.1 \cdot \sin(\pi x) \cdot \cos\left(2\pi\frac{t-1}{T}\right)$.

Figure 2 shows three-dimensional surface plots of the desired relative formation trajectories for the four followers. The horizontal axis represents the spatial position x, the vertical axis represents the time step t, and the vertical axis represents the trajectory output value y. The figure presents four typical formation requirements: constant offset, spatially correlated offset, time-correlated offset, and spatiotemporal coupled offset, which cover the common spatiotemporal evolution formation patterns in distributed parameter systems.

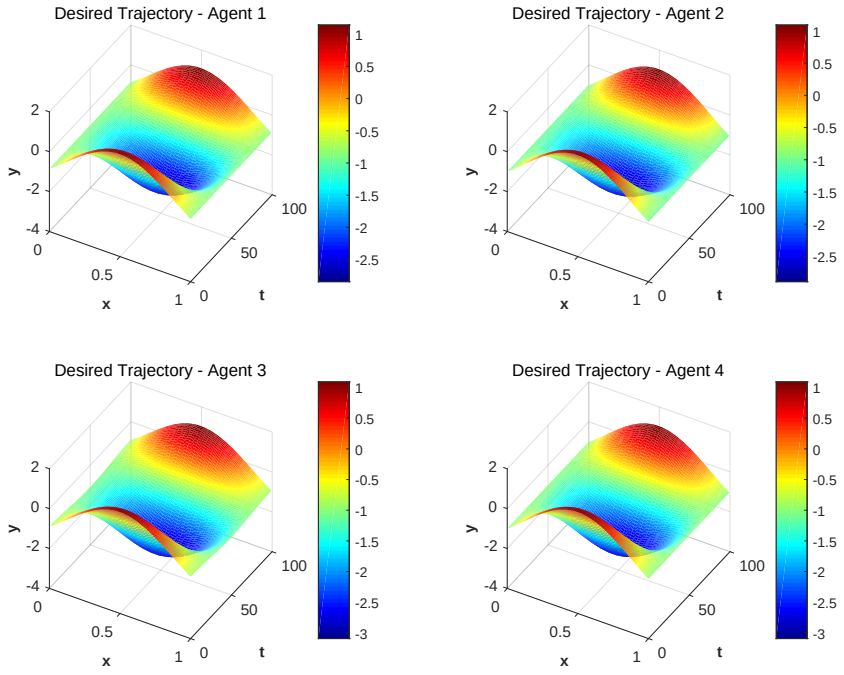

Figure 2. Actual outputs of four followers.

An RBF neural network approximates the unknown linear operator of the system and is combined with an error-driven iterative learning control algorithm. The key parameters include the number of iterations $K = 50$, the learning gain $\lambda = 0.6$, and the regularization parameter $\beta = 0.05$. This parameter set satisfies the convergence constraint $\lambda < 2\left(1 + \frac{\beta}{\|\varphi(\chi)\|^2}\right)$. The RBF neural network contains 25 hidden layer nodes and a kernel width $\sigma = \frac{1.8(T-1)}{p-1}$.

Figure 3 shows the actual output trajectories of the followers after 50 iterations, with the axes consistent with those in Figure 2. Figure 4 shows the consensus tracking errors between the four agents and their respective followers. These errors represent the deviation between the actual output and the desired relative formation trajectories, i.e., $y_{di}(x,t) - y_i(x,t)$, $i = 1,2,3,4$. All errors converge to within 0.01.

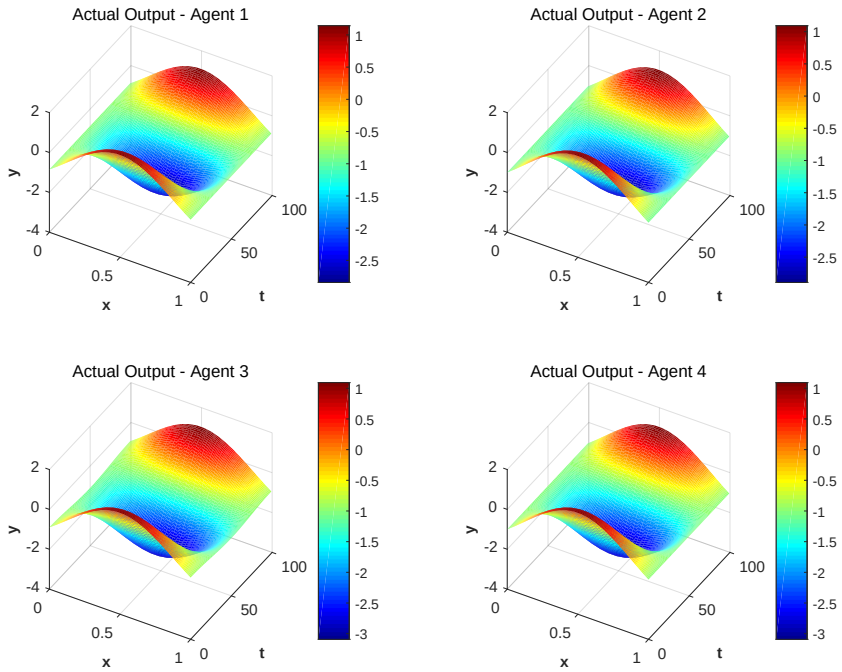

Figure 3. Actual outputs of four followers.

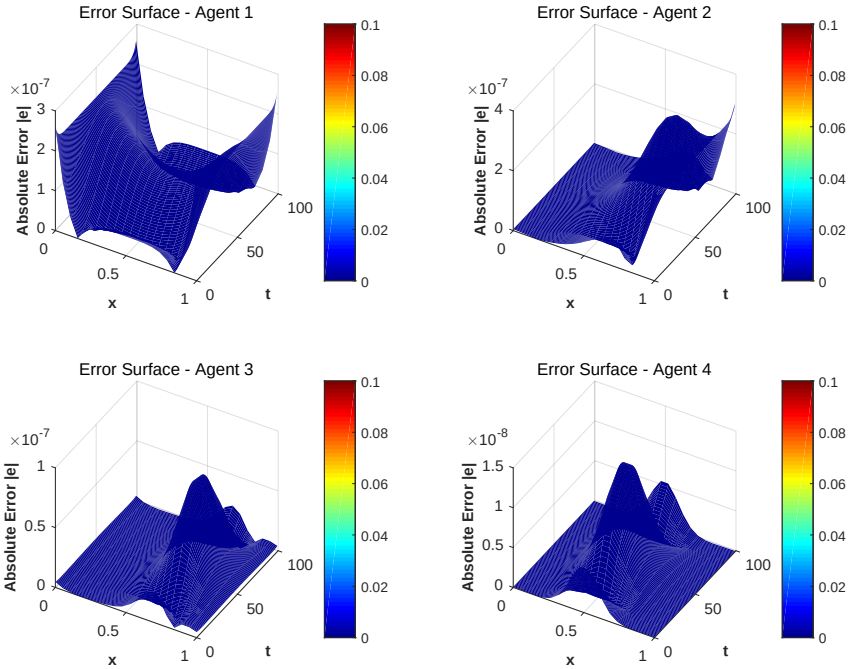

Figure 4. Convergence tracking error of four followers.

Figure 5 shows that as the number of iterations increases, the Frobenius norm of the errors for all

four agents gradually converges. Around the 20th iteration, the error norms of all four agents converge to zero.

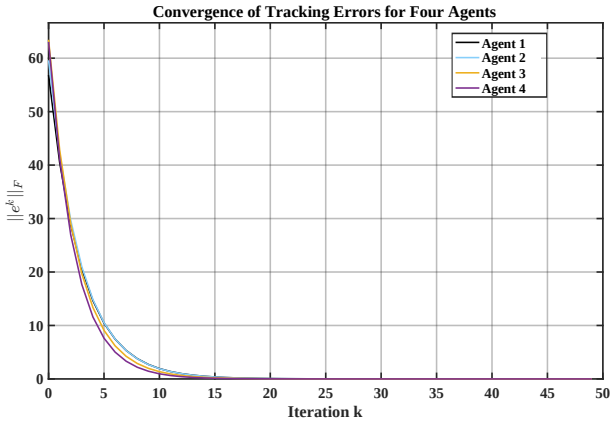

Figure 5. Convergence of tracking errors between four intelligent agents and their respective leaders.

4.2. Simulation results under a fast time-varying trajectory

To evaluate the robustness of the proposed method under challenging operating conditions, we consider a rapidly varying virtual leader trajectory defined as

$$y_0(x,t) = 2\sin(\pi x)\cdot\exp\left(-(x-0.5)^2\right)\cdot\cos\left(\frac{20\pi(t-1)}{T}\right)-1,$$

where $x \in [0,1]$ is discretized into 50 spatial points and $t \in [1,T]$ with $T = 100$ time steps. This trajectory incorporates a spatial Gaussian window $\exp(-(x-0.5)^2)$, a spatial sinusoidal distribution $\sin(\pi x)$, and a high-frequency temporal oscillation $\cos(20\pi(t-1)/T)$, resulting in a rapidly time-varying and spatiotemporally coupled reference.

The four following agents are required to track the same formation pattern described in Section 4. A comparison of the convergence performance between the proposed DDILC method and the traditional P-type method conducted under this rapidly varying reference trajectory.

Figure 6 illustrates the convergence process of tracking errors for the four agents under the traditional ILC and the proposed DDILC algorithm. The vertical axis adopts a logarithmic scale to clearly show the decreasing trend of the errors. Different colors distinguish the four agents, while solid and dashed lines correspond to the traditional ILC and DDILC algorithms, respectively. The results demonstrate that the DDILC algorithm not only converges much faster than the traditional ILC, but also reduces the final steady-state error by one to two orders of magnitude, fully validating the effectiveness of the proposed algorithm in fast time-varying trajectory tracking tasks.

Figure 7 presents the spatiotemporal distribution surfaces of tracking errors for the four agents at the final iteration, intuitively illustrating the difference in error distribution between the traditional ILC and the proposed DDILC algorithm. The horizontal axis represents the time step, the vertical axis represents the spatial position, and the vertical axis represents the absolute tracking error. Surfaces with different transparencies correspond to the two algorithms. It can be observed that the traditional ILC algorithm exhibits significant tracking errors across the entire spatiotemporal domain, whereas the

error surface of the proposed DDILC algorithm nearly coincides with the zero plane, confirming that the DDILC algorithm achieves high-precision tracking in both the temporal and spatial dimensions.

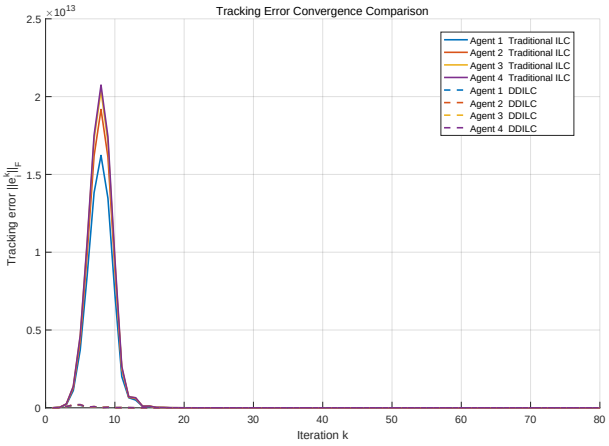

Figure 6. Comparison of iterative convergence curves of tracking errors for four agents.

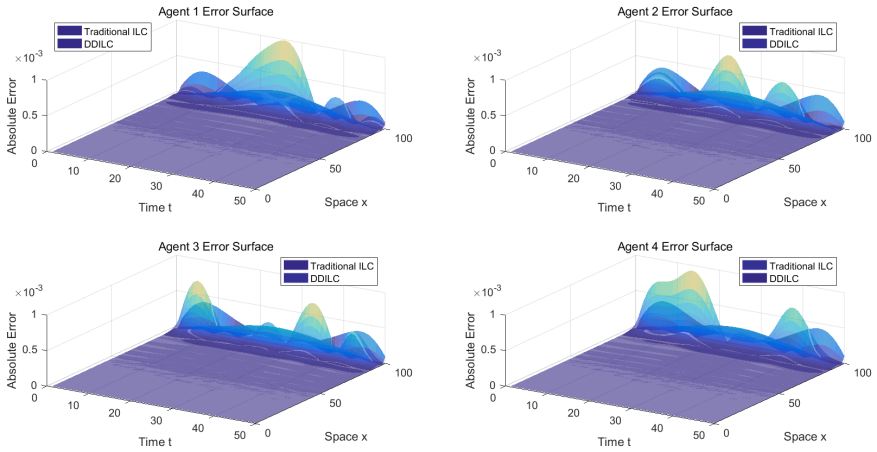

Figure 7. Comparison of spatiotemporal distribution surfaces of tracking errors in the final iteration for four agents.

5. Discussion

The proposed method demonstrates significant advantages over traditional model-based approaches. By leveraging neural networks to approximate unknown linear operators of the system effectively addresses the consensus control challenge in linear parabolic distributed parameter multi-agent systems with unknown parameters. This eliminates reliance on precise mathematical modeling an inherent limitation of traditional control strategies. Rigorous theoretical analysis proves the boundedness of the neural network weight estimation errors and the exponential convergence of the consensus tracking

errors along the iteration axis. Numerical simulations in Section 4 further validate these theoretical conclusions.

Simulation results from both the conventional smooth reference trajectory (Section 4.1) and the rapidly varying non-smooth trajectory (Section 4.2) confirm that the proposed algorithm achieves high-precision consensus tracking within approximately 20 iterations, with all tracking errors converging to within 0.01. In particular, the comparative simulation results in Section 4.2 clearly show that the DDILC algorithm outperforms the traditional P-type ILC algorithm in both convergence speed and steady-state accuracy under the fast time-varying trajectory scenario. This fully reflects the good robustness and adaptability of the proposed method in handling complex spatiotemporal coupling reference trajectory tasks.

Notably, the method features linear computational complexity with respect to the number of agents. Each agent independently updates its neural network weights using only local input-output data and neighbor information, without requiring global precise system information. This inherent parallelism not only adapts to the information interaction characteristics of distributed multi-agent systems but also makes the algorithm easy to extend to large-scale agent cluster scenarios. Therefore, the DDILC method has important practical application value in real-world engineering fields such as UAV swarm formation, robotic team collaboration, and intelligent industrial synergy, where accurate system modeling is often difficult or even impossible to achieve.

6. Conclusions

This study presents a novel consensus control framework based on data-driven iterative learning for distributed parameter multi-agent systems with parametric uncertainties. The approach constructs a data-oriented model that captures the input-output mapping of the system, employing neural networks to approximate unmodeled dynamics. A weight update mechanism governed by iterative learning principles is formulated. Theoretical guarantees regarding the boundedness of neural network parameters and the asymptotic convergence of tracking errors are rigorously established through Lyapunov analysis and norm-based arguments. Numerical simulation shows that the virtual leader and four following intelligent agents achieve precise consensus tracking within a finite number of iterations, and the tracking error converges exponentially along the iteration axis, verifying the effectiveness of the proposed algorithm. This method overcomes the reliance on precise models in traditional control approaches and offers a new strategy for addressing control challenges in multi-agent systems characterized by difficult model establishment, limited information, and complex dynamics.

Future research directions include extending the method to nonlinear distributed parameter multi-agent systems, investigating the effects of communication delays and packet losses, and applying the method to practical engineering applications such as multi-robot systems and smart grid control. The proposed method exhibits linear computational complexity with respect to the number of agents, as the update law (3.4) is applied independently to each agent based on its local data. This property makes the algorithm well suited for large-scale agent clusters. In future work, we will explore the nonlinear extension problem and verify its performance in multi-leader and large-scale scenarios through more extensive simulations.

Author contributions

Lanlan Liu: Conceptualization, Methodology, Writing–original draft, Validation, Writing–review and editing; Xisheng Dai: Supervision. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work is supported by National Natural Science Foundation of China (NNSFC) under Grant 62363002

Conflict of interest

The authors declare no conflict of interest.

References

1. M. Minsky, The society of mind, *Personalist Forum*, **3** (1987), 19–32. https://doi.org/10.2307/20708493
2. C. L. Liu, S. Liu, Y. Zhang, Y. Y. Chen, Consensus seeking of multi-agent systems with intermittent communication: a persistent-hold control strategy, *Int. J. Control*, **93** (2020), 2161–2167. https://doi.org/10.1080/00207179.2018.1548784
3. C. D'Apice, R. Manzo, B. Piccoli, Differential inclusions for measures and lyapunov stability, *J. Optim. Theory Appl.*, **208** (2026), 13. https://doi.org/10.1007/s10957-025-02828-9
4. Y. Hou, J. Zhao, R. Zhang, X. Cheng, L. Yang, UAV swarm cooperative target search: A multi-agent reinforcement learning approach, *IEEE Trans. Intell. Veh.*, **9** (2023), 568–578. https://doi.org/10.1109/TIV.2023.3316196
5. M. Li, M. Ma, L. Wang, Z. Pei, J. Ren, B. Yang, Multiagent deep reinforcement learning based incentive mechanism for mobile crowdsensing in intelligent transportation systems, *IEEE Syst. J.*, **18** (2024), 527–538. https://doi.org/10.1109/JSYST.2024.3351310
6. Z. Luo, D. Li, J. Wan, S. Wang, G. Wang, M. Cheng, et al., Multi-agent collaboration mechanisms based on distributed online meta-learning for mass personalization, *J. Indust. Inform. Integr.*, **46** (2025), 100852. https://doi.org/10.1016/j.jii.2025.100852
7. Y. Huang, W. Cheng, C. Tang, C. Wang, Study of multi-agent-based coal mine environmental monitoring system, *Ecol. Indic.*, **51** (2015), 79–86. https://doi.org/10.1016/j.ecolind.2014.09.047
8. H. Y. Li, Q. L. Wei, Data-driven optimal output cluster synchronization control of heterogeneous multi-agent systems, *IEEE Trans. Autom. Sci. Eng.*, **21** (2023), 3910–3920. https://doi.org/10.1109/TASE.2023.3289950

9. Y. Zhang, Z. Xiang, Prescribed-time optimal control for a class of switched nonlinear systems, *IEEE Trans. Autom. Sci. Eng.*, **22** (2024), 3033–3043. https://doi.org/10.1109/TASE.2024.3388456
10. Z. X. Li, H. B. Ji, Robust H_∞ Convergence control for multi agent systems with input delay, *J. Autom.*, **40** (2014), 2556–2562. https://doi.org/10.1016/S1874-1029(14)60401-8
11. Z. Xiang, P. Li, M. Chadli, W. Zou, Fuzzy optimal control for a class of discrete-time switched nonlinear systems, *IEEE Trans. Fuzzy Syst.*, **32** (2024), 2297–2306. https://doi.org/10.1109/TFUZZ.2023.3348535
12. J. Wu, N. Liu, W. Tang, Data-driven tracking consensus for a class of unknown nonlinear multi-agent systems, *J. Vibr. Control*, **28** (2022), 3559–3574. https://doi.org/10.1177/10775463211034049
13. Y. Chen, F. Zhang, J. Li, Anti-disturbance fault-tolerant constrained consensus for time-delay faulty multi-agent systems with semi-markov switching topology, *Mathematics*, **10** (2022), 4581–4592. https://doi.org/10.3390/math10234581
14. C. Liu, D. Shen, J. R. Wang, Iterative learning control of multi-agent systems with random noises and measurement range limitations, *Taylor Francis*, **50** (2019), 1465–1482. https://doi.org/10.1080/00207721.2019.1616127
15. C. D'Apice, R. Manzo, B. Piccoli, Existence of solutions to Cauchy problems for a mixed continuum-discrete model for supply chains and networks, *J. Math. Anal. Appl.*, **362** (2010), 374–386. https://doi.org/10.1016/j.jmaa.2009.07.058
16. C. D'Apice, R. Manzo, L. Rarità, B. Piccoli, Relaxed controls and measure controls, In: *2024 IEEE 63rd Conference on Decision and Control (CDC), Milan, Italy*, 2024, 1949–1954. https://doi.org/10.1109/CDC56724.2024.10886291
17. P. Gong, K. Wang, W. Y. Lan, Fully distributed robust consensus control of multi-agent systems with heterogeneous unknown fractional-order dynamics, *Int. J. Syst. Sci.*, **50** (2019), 1902–1919. https://doi.org/10.1080/00207721.2019.1645913
18. K. Yang, C. Li, X. Jing, Z. Zhu, Y. Wang, H. Ma, et al., Energy dispatch optimization of islanded multi-microgrids based on symbiotic organisms search and improved multi-agent consensus algorithm, *Energy*, **239** (2022), 122105. https://doi.org/10.1016/j.energy.2021.122105
19. Q. Wang, Z. Duan, Y. Lv, Q. Wang, G. Chen, Linear quadratic optimal consensus of discrete-time multi-agent systems with optimal steady state: A distributed model predictive control approach, *Automatica*, **127** (2021), 109505. https://doi.org/10.1016/j.automatica.2021.109505
20. Y. Yang, Y. Qian, Dynamic event-triggered mechanism-based output consensus of nonlinear multi-agent systems via improved dynamic surface control approach, *Int. J. Control*, **95** (2022), 3392–3403. https://doi.org/10.1080/00207179.2021.1974095
21. Q. Fu, L. L. Du, G. Z. Xu, J. Wu, P. Yu, Consensus control for multi-agent systems with distributed parameter models, *Neurocomputing*, **308** (2018), 58–64. https://doi.org/10.1016/j.neucom.2018.04.051
22. X. S. Dai, C. Wang, S. P. Tian, Q. Huang, Consensus control via iterative learning for distributed parameter models multi-agent systems with time-delay, *J. Franklin Inst.*, **356** (2019), 5240–5259. https://doi.org/10.1016/j.jfranklin.2019.05.015

23. Y. H. Lan, B. Wu, Y. X. Shi, Y. P. Luo, Iterative learning based consensus control for distributed parameter multi-agent systems with time-delay, *Neurocomputing*, **357** (2019), 77–85. https://doi.org/10.1016/j.neucom.2019.04.064
24. R. H. Chi, Z. S. Hou, S. T. Jin, B. Huang, Computationally efficient data-driven higher order optimal iterative learning control, *IEEE Trans. Neur. Networks Learning Syst.*, **29** (2018), 5971–5980. https://doi.org/10.1109/TNNLS.2018.2814628
25. Z. S. Hou, R. H. Chi, H. J. Gao, An overview of dynamic-linearization-based data-driven control and applications, *IEEE Trans. Indust. Elect.*, **64** (2016), 4076–4090. https://doi.org/10.1109/TIE.2016.2636126
26. Z. S. Hou, S. T. Jin, Data-driven model-free adaptive control for a class of MIMO nonlinear discrete-time systems, *IEEE Trans. Neur. Networks*, **22** (2011), 2173–2188. https://doi.org/10.1109/TNN.2011.2176141
27. X. S. Dai, X. Y. Zhou, Mixed PD-type iterative learning control algorithm for a class of parabolic singular distributed parameter systems, *IEEE Access*, **9** (2021), 12180–12190. https://doi.org/10.1109/ACCESS.2021.3050486
28. Y. Qi, X. Zhao, J. Huang, Data-driven event-triggered control for switched systems based on neural network disturbance compensation, *Neurocomputing*, **490** (2022), 370–379. https://doi.org/10.1016/j.neucom.2021.11.103
29. B. Chu, P. Rapisarda, Data-driven iterative learning control for continuous-time systems, In: *2023 62nd IEEE Conference on Decision and Control (CDC)*, 2023, 4626–4631. https://doi.org/10.1109/CDC49753.2023.10384133
30. Y. Zhang, Z. Xiang, Nash equilibrium solutions for switched nonlinear systems: A fuzzy-based dynamic game method, *IEEE Trans. Fuzzy Syst.*, **33** (2025), 2006–2015. https://doi.org/10.1109/TFUZZ.2025.3549879
31. X. Wang, D. Ding, X. Ge, H. Dong, Neural-network-based control with dynamic event-triggered mechanisms under DoS attacks and applications in load frequency control, *IEEE Trans. Cir. Syst. I: Regular Papers*, **69** (2022), 5312–5324. https://doi.org/10.1109/TCSI.2022.3206370
32. X. S. Dai, L. L. Liu, Z. P. Deng, Optimised data-driven terminal iterative learning control based on neural network for distributed parameter systems, *Int. J. Autom. Control*, **15** (2021), 463–481. https://doi.org/10.1504/ijaac.2021.116422
33. Z. Hu, S. Liu, W. Luo, L. Wu, Resilient distributed fuzzy load frequency regulation for power systems under cross-layer random denial-of-service attacks, *IEEE Trans. Cyber.*, **52** (2020), 2396–2406. https://doi.org/10.1109/TCYB.2020.3005283
34. A. D. Shakibjoo, M. Moradzadeh, S. Z. Moussavi, A. Mohammadzadeh, L. Vandevelde, Load frequency control for multi-area power systems: A new type-2 fuzzy approach based on levenberg marquardt algorithm, *ISA Trans.*, **121** (2022), 40–52. https://doi.org/10.1016/j.isatra.2021.03.044