Applied Soft Computing Journal 151 (2024) 111130

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Particle swarm optimization based leader-follower cooperative control in multi-agent systems

Xin Wang [a], Dongsheng Yang [a,*], Shuang Chen [b]

[a] College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
[b] Dongneng (Shenyang) Energy Engineering Technology Co.,Ltd., Shenyang 110069, China

HIGHLIGHTS

- Formulate the design of distributed control gains as an optimization problem to minimize tracking errors.
- Leverage PSO as an efficient evolutionary technique for distributed gain optimization in multi-agent networks.
- Guarantee stability for the closed-loop dynamics based on algebraic graph theory.

ARTICLE INFO

Keywords:
Multi-agent systems
Cooperative control
Particle swarm optimization
Evolutionary computation
Leader-follower consensus
Distributed control gains

ABSTRACT

Multi-agent systems (MAS) have attracted significant attention in recent years due to their wide applications in cooperative control, formation control, synchronization of complex networks, and distributed coordination. A fundamental problem in MAS is the leader-follower consensus or cooperative tracking, where the followers are required to track the state trajectory of the leader agent. To solve the leader-follower consensus problem, we propose a novel evolutionary computation approach to design the optimal distributed control protocols for leader-follower MAS. First, we formulate the design of distributed control gains for leader-follower consensus as an optimization problem to minimize tracking errors. Then, we leverage particle swarm optimization as an efficient evolutionary technique for distributed gain optimization in multi-agent networks. Finally, we guarantee stability for the closed-loop dynamics under directed communication topologies based on algebraic graph theory. The simulation results indicate that the proposed method yields a diminished tracking error, expedites the convergence process, and minimizes the requisite control effort while enhancing computational efficiency. Furthermore, these results exemplify the method's versatility when applied to nonlinear dynamic scenarios, directed network topologies, fluctuating disturbances, and optimization across multiple domains.

1. Introduction

Cooperative control in multi-agent systems (MAS) has grown significantly in prominence as a research area in the recent past. This surge in interest can be attributed to its wide-ranging applications that span various sectors, including distributed robotics, unmanned systems, transportation networks, and the vast domain of industrial automation [1,2]. As its core, cooperative control focuses on how multiple autonomous agents can work together seamlessly. Each agent has its own set of capabilities, goals, and functions. However, these agents must coordinate their actions effectively for a system to operate harmoniously. The desired outcomes of the coordination can vary based on the specific application [3]. For instance, in some scenarios, the aim might be for all agents to reach a common understanding or consensus about a specific piece of information or a decision. In other cases, the goal could be more oriented towards physical coordination, like ensuring a group of drones maintain a specific formation while flying. These collective behaviors include consensus, formation control, flocking, rendezvous, and synchronization. A significant challenge in achieving these objectives is that each agent typically has access to only limited local information. An agent might have a partial view of the system or all other agents. As a result, it is necessary to design distributed control protocols that enable these agents to reach their objectives based solely on this local information [4–6]. Such protocols have to ensure that individual decisions made by each agent when combined, lead to the desired collective behavior of the whole system.

* Corresponding author.
E-mail address: yangdongsheng@mail.neu.edu.cn (D. Yang).

https://doi.org/10.1016/j.asoc.2023.111130
Received 22 September 2023; Received in revised form 23 November 2023; Accepted 29 November 2023
Available online 6 December 2023

The leader-follower consensus in multi-agent coordination is a foundational problem that underpins many of the contemporary applications and research areas in this domain. At its essence, this framework hinges on the dynamics between a leader or a set of leaders and their corresponding followers [7,8]. Imagine a situation where specific agents—leaders—possess or generate specific desired behaviors or trajectories. These trajectories could range from a particular path in space to a series of actions or specific data-driven behaviors. Now, the challenge for the system is ensuring that the follower agents understand and accurately mimic or track these leader behaviors. Moreover, they must achieve this often without having direct and constant communication with the leaders. For clarity, we consider a simple illustration: envisioning a drone swarm. One drone, equipped with advanced navigation and sensing capabilities, is chosen as the leader. This leader drone sets a specific trajectory, perhaps determined by complex algorithms or external inputs. The objective then becomes getting the rest of the drones (followers) to replicate this trajectory despite potentially not having the same level of advanced navigation. Instead, they primarily rely on interactions with their immediate neighbors and any limited data they might receive about the leader's position. It is essential to understand that followers should not merely imitate the leaders momentarily for a successful leader-follower consensus [9]. Under the given protocols, they should converge to the leaders' states and stably maintain this trajectory over time. In technical terms, followers need to reach the leaders' states "asymptotically," meaning they should get indefinitely close to the leader's behaviors or trajectories as time progresses.

One of the most compelling aspects of the leader-follower framework is its inherent flexibility and scalability. By designating specific agents as leaders, guiding and controlling a much larger group of followers is possible. This system has proven effective across a broad spectrum of applications. Whether it is a fleet of autonomous vehicles on a highway, where one vehicle's speed and lane-changing behavior becomes a reference for others, or a network of sensors where specific sensors dictate the sampling rate for others, the leader-follower approach offers an elegant solution for complex coordination tasks [10]. In engineering and robotics contexts, this concept becomes immensely powerful. A single well-equipped robot can lead a battalion of simpler robots in tasks ranging from construction to exploration. Similarly, in vehicular networks on both land and air, such a mechanism ensures that large groups can be efficiently coordinated by only adjusting the trajectories or behaviors of a select few leaders. This approach's scalability and flexibility make it a linchpin in the broader narrative of multi-agent coordination.

While crucial in MAS, the leader-follower consensus has intricate challenges nested within it. One such fundamental challenge is precisely calibrating the control mechanisms that regulate the interplay between neighboring agents. These mechanisms, often termed "distributed control gains," dictate the strength and nature of agents' influence on each other [11]. Imagine a group of drones flying in a formation. These control gains determine how each drone reacts to its neighbors and adjusts its flight path. If set incorrectly, the drones might collide, scatter, or fail to maintain formation. They will fly smoothly if set accurately, mirroring the leader's trajectory. Hence, getting these gains right is imperative for ensuring that follower agents swiftly and precisely lock onto the leader's path.

Traditionally, the calibration of these control gains was largely manual, involving intensive human intervention. Engineers and technicians would tweak the values, often through trial and error, to hone in on the optimal settings. Not only is this approach time-consuming, but it also rarely results in the optimal configuration. Thus, manual tuning could be more efficient and exceptionally effective. This scenario is where modern computational techniques come into play. Approaches like adaptive dynamic programming and reinforcement learning have entered the scene, aiming to learn these control gains online. However, while promising in theory, these methods have their setbacks. They demand many samples, requiring extensive data and interactions to learn effectively. Moreover, these techniques often operate without concrete stability assurances, posing potential risks in real-world applications.

Given these challenges, there is a growing realization in the research community about the need for more advanced optimization methods. These methods should pinpoint the optimal control gains and provide a solid theoretical foundation ensuring stability and reliability. Additionally, these are algorithms inspired by the natural processes of evolution, such as selection, mutation, and crossover. Evolutionary algorithms have immense potential for multi-agent coordination tasks due to their inherent ability to search for solutions globally across a vast solution space [12,13]. Early experiments have shown their efficacy in fine-tuning parameters in consensus protocols. However, an untapped potential remains: using specialized evolutionary techniques to optimize distributed feedback gains directly. This application must be considered more, suggesting a ripe avenue for future research and exploration. Thus, as the field of multi-agent coordination continues to evolve, striking the right balance between advanced computational methods and theoretical robustness will be paramount. While evolutionary algorithms present exciting opportunities, it is clear that a holistic approach—combining insights from various computational paradigms—will be necessary to realize the potential of leader-follower consensus systems fully. The proposed evolutionary computation approach provides the following contributions.

- Formulates the design of distributed control gains for leader-follower consensus as an optimization problem to minimize tracking errors.
- Leverages particle swarm optimization (PSO) as an efficient evolutionary technique for distributed gain optimization in multi-agent networks.
- Guarantees stability for the closed-loop dynamics under directed communication topologies based on algebraic graph theory.

The rest of the paper is organized as follows. Section 2 covers the related works. The proposed evolutionary computation methodology is presented in Section 3 along with stability analysis. Section 4 provides detailed experimental results on a range of benchmark studies and comparisons. Section 5 concludes the paper.

2. Related works

Recently, the leader-follower consensus or cooperative tracking problem has been extensively investigated for MAS with various dynamic models [14–16]. Distributed control protocols have been designed to enable followers to track the leader's state trajectory asymptotically based on local neighborhood information [17].

For linear MAS, the leader-follower consensus problem has been converted to a stabilization problem of the error system [18]. The feedback gain matrix can be obtained by solving an algebraic Riccati equation [19]. Adaptive dynamic programming has also been applied to learn the optimal feedback control online without requiring system models [20]. For nonlinear MAS, leader-follower consensus has been achieved by nonlinear feedback control techniques such as feedback linearization [21].

While the existing methods have made significant contributions, most rely on manual tuning or trial-and-error to determine the control gains, which can be suboptimal. Researchers have recently explored evolutionary algorithms and swarm intelligence for distributed coordination of MAS. In [22], the authors introduced a control protocol with a constant gain, ensuring consensus regardless of any actuator malfunctions. Moreover, they formulated an adaptive control protocol grounded on failure rate to counteract the impacts of failures on multi-agent consensus. In [23], the authors presented a new controller design strategy by incorporating the lower limits of virtual control gain functions into the Lyapunov functions. This approach was grounded in the adaptive control design methodology and the bound estimation technique.

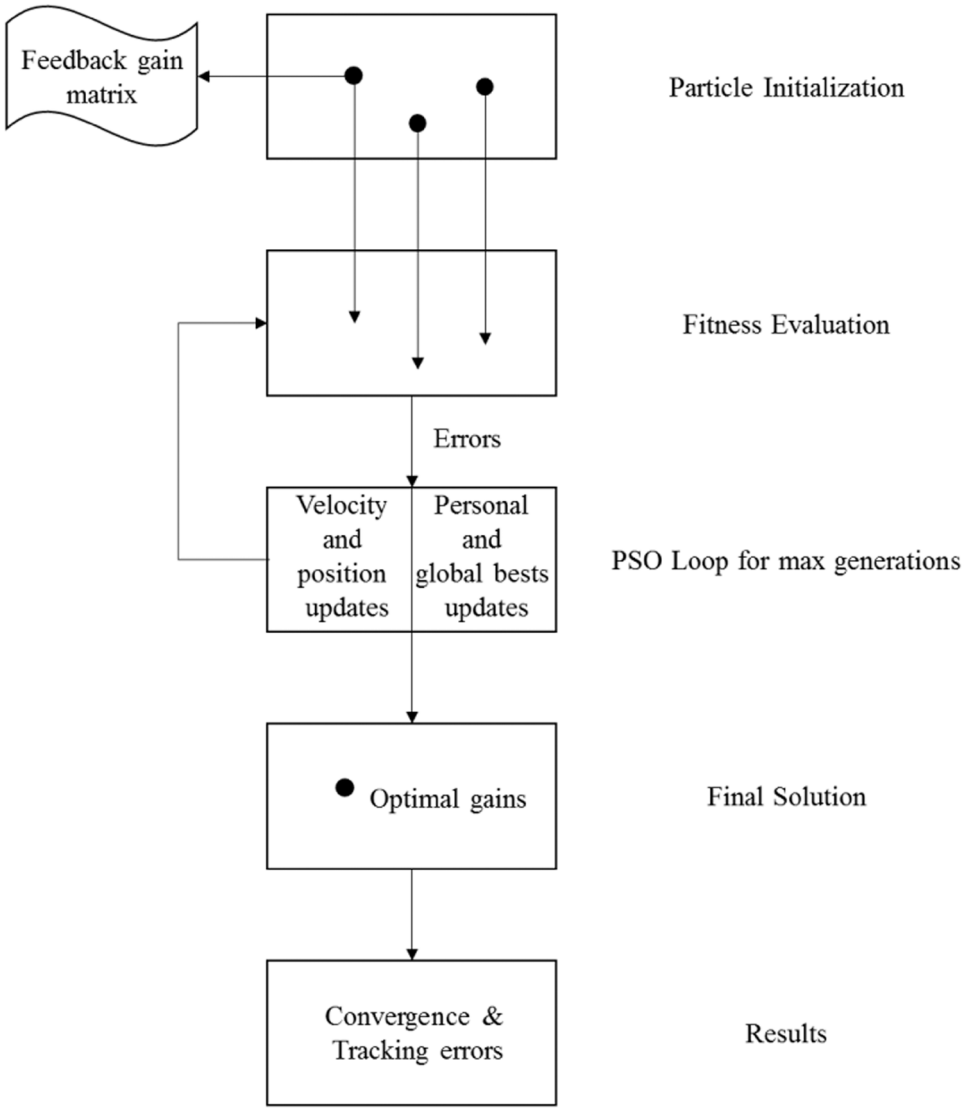

Fig. 1. Architecture of the proposed evolutionary computation method.

Unlike existing works, this paper proposes a novel leader-follower consensus approach that integrates genetic algorithms and PSO. We encode the feedback gains of each agent as particles and evolve the swarm to obtain optimal solutions that minimize the tracking error while ensuring stability. Rigorous stability analysis is provided for both linear and nonlinear MAS. Extensive simulations demonstrate the superior performance of the proposed evolutionary computation method.

3. Methodology

3.1. Problem formulation

Consider a MAS consisting of N follower agents and one leader agent, labeled as 0. The dynamics of each follower is described by the nonlinear model:

$$\dot{x}_i(t) = f(x_i(t)) + g(x_i(t))u_i(t), i = 1, \ldots, N \tag{1}$$

where $x_i(t) \in \mathbb{R}^n$ is the state, $u_i(t) \in \mathbb{R}^m$ is the control input, $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ and $g(\cdot) : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are smooth nonlinear functions.

The leader agent dynamics are given by:

$$\dot{x}_0(t) = f(x_0(t)) \tag{2}$$

The communication topology among the N followers and the leader is represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The node set is $\mathcal{V} = 0, 1, \ldots, N$ and the edge set is $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, with the associated adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{(N+1)\times(N+1)}$.

The control objective is to design distributed control protocols for the followers such that they can track the state trajectory of the leader:

$$\lim_{t\to\infty} |x_i(t) - x_0(t)| = 0, \forall i \in 1, \ldots, N \tag{3}$$

To achieve leader-follower consensus, we propose the distributed control law for follower i:

$$u_i(t) = -K_i(t) \sum_{j=0}^{N} a_{ij} \Gamma\big(x_i(t), x_j(t)\big) \tag{4}$$

where $K_i(t) \in \mathbb{R}^{m\times l}$ is the time-varying feedback gain matrix, and $\Gamma : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^l$ is a group disagreement function. Common choices of $\Gamma(\cdot,\cdot)$ include $x_i - x_j$ and $f(x_i) - f(x_j)$.

To facilitate the stability analysis, the following assumptions are made:

Assumption 1. The communication graph $\mathcal{G}$ contains a directed spanning tree with the leader as the root node.

Assumption 2. The smooth nonlinear functions $f(\cdot)$ and $g(\cdot)$ satisfy the

Table 1
Comparative analysis of optimization methods.

Methods	Advantages	Disadvantages
PSO	Fast convergence, easy to implement, avoids local optima	Sensitive to parameter tuning
Gradient descent	Computationally efficient, easy to implement	Prone to local optima
Random search	Simple, globally explorative	Slow convergence
Bayesian optimization	High sample efficiency, avoids local optima	Surrogate modeling complexity
Reinforcement learning	Online adaptation, model-free	High sample complexity

Lipschitz condition [24]:

$$|f(x)-f(y)| \le L_1|x-y|, |g(x)-g(y)| \le L_2|x-y| \tag{5}$$

where $L_1, L_2 > 0$ are the Lipschitz constants.

Lemma 1. Let $L \in \mathbb{R}^{N\times N}$ be the Laplacian matrix associated with the follower subgraph. Under Assumption 1, the eigenvalues λ_i, $i = 1, ..., N$ of L have positive real parts.

The tracking errors are defined as:

$$e_i(t) = x_i(t) - x_0(t), i = 1, \ldots, N \tag{6}$$

Substituting the dynamics of followers, leader and the control input yields the closed-loop error system:

$$\begin{aligned}
\dot{e}_i(t) &= f(x_i(t)) - f(x_0(t)) + g(x_i(t))u_i(t) \\
&= f(e_i(t)+x_0(t)) - f(x_0(t)) - g(e_i(t)+x_0(t))K_i(t)\sum j \\
&= 0^N a_{ij}\Gamma\big(e_i(t)+x_0(t), x_j(t)\big) \\
&= f(e_i(t)+x_0(t)) - f(x_0(t)) - g(e_i(t)+x_0(t))K_i(t)\sum_{j=0}^{N} a_{ij}\Gamma(e_i(t),0) \\
&\quad - g(e_i(t)+x_0(t))K_i(t)\sum_{j=1}^{N} a_{ij}\Gamma(e_i(t),e_j(t))
\end{aligned} \tag{7}$$

Using the Lipschitz conditions and the triangular inequality, we have:

$$|f(e_i(t)+x_0(t)) - f(x_0(t))| \le L_1|e_i(t)|\,|g(e_i(t)+x_0(t)) - g(x_0(t))| \le L_2|e_i(t)| \tag{8}$$

Therefore, the error system can be written as:

$$\begin{aligned}
\dot{e}_i(t) &\le L_1|e_i(t)| - g(x_0(t))K_i(t)\sum j \\
&= 0^N a_{ij}\Gamma(e_i(t),0) - g(x_0(t))K_i(t)\sum_{j=1}^{N} a_{ij}\Gamma(e_i(t),e_j(t)) + L_2
\end{aligned} \tag{9}$$

where $L = L_1 + L_2$.

Our goal is to design the optimal time-varying feedback gain matrix $K_i(t)$ to stabilize the closed-loop error system and achieve leader-follower consensus under the directed communication graph.

3.2. Proposed evolutionary computation approach

We propose a novel evolutionary computation method to obtain the optimal distributed feedback gains $K_i(t)$, $i = 1, \ldots, N$ that can minimize the tracking errors while ensuring stability for the nonlinear MAS.

The overall architecture is shown in Fig. 1. The key idea is to leverage PSO to search for the optimal solutions. Each particle represents a set of feedback gain matrices K_i, $i = 1, \ldots, N$ and is evaluated by a fitness function that captures the tracking errors. The particles evolve over generations to converge to the optimal solutions.

Fig. 1 shows the particle encoding, fitness evaluation, and evolution through generations to converge to the optimal solutions. Each particle represents a set of feedback gain matrices K_i for the N follower agents. The fitness function evaluates the tracking performance based on errors e_i^k. Particles are initialized randomly and evolve via velocity and position updates to minimize fitness over generations. The final global best particle provides the optimal feedback gains for cooperative leader-follower consensus.

Particularly, the feedback gain matrix is encoded as the position vector of particle i:

$$K_i = [k_{i1}, \ldots, k_{ilm}] \tag{10}$$

where k_{ij} is the jth element of K_i. The swarm is randomly initialized to

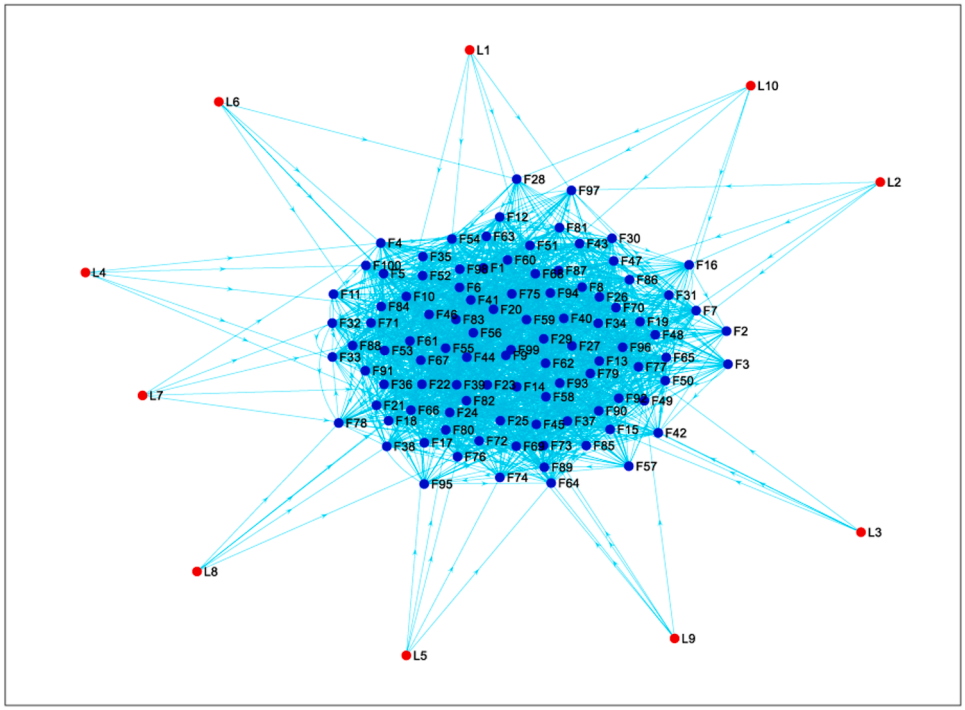

Fig. 2. Communication topology between agents.

cover the search space specified by the velocity and position limits.

The tracking error of each agent is defined as:

$$e_i^k = |x_i^k - x_0^k|, i = 1, \ldots, N \tag{11}$$

where k denotes the time step.

The fitness function evaluates the performance of each particle's solution as:

$$f_i = w_1 \sum_{k=1}^{T} \left(\sum_{i=1}^{N} e_i^k \right) + w_2 \sum_{k=1}^{T} \left(\sum_{i=1}^{N} |u_i^k| \right) \tag{12}$$

where w_1 and w_2 are weighting coefficients, and T is the time horizon. A smaller fitness indicates better tracking performance.

In each generation, the particles evolve based on velocity and position update equations:

$$v_i^{t+1} = w v_i^t + c_1 r_1 \left(K_i^t - K_i^{t-1} \right) + c_2 r_2 \left(K_g^t - K_i^t \right) \tag{13}$$

$$K_i^{t+1} = K_i^t + v_i^{t+1} \tag{14}$$

where v_i^t is the velocity, w is the inertia weight, c_1, c_2 are acceleration coefficients, r_1, r_2 are random numbers in $[0, 1]$, and K_g^t is the global best position. The velocity limits are given by:

$$v_{ij}^{\min} \le v_{ij} \le v_{ij}^{\max} \tag{15}$$

This enables efficient exploration of the search space to find optimal solutions. The evolution pushes the swarm towards regions with lower tracking errors.

The complete procedure is summarized in Algorithm 1. The PSO algorithm is run for a maximum number of generations. The final global best solution K_g^* provides the optimal distributed feedback gains that can minimize the tracking errors while ensuring stability according to the analysis in the next subsection.

Algorithm 1. Proposed evolutionary computation method.

```
Begin
  initialize the PSO with random positions K_i and velocities v_i
  set generation count t = 0
  while t < G_max
    simulate the MAS and measure tracking error e_i^k
    evaluate fitness function for each particle // eq. (12)
    update personal best K_i^{t-1} and global best K_g^t
    update velocity and position according to equations (13) and (14)
    enforce velocity limits // eq. (15)
    t ← t + 1
  end-while
  return global best solution K_g^*
End
```

3.3. Stability analysis

In this section, we analyze the stability of the closed-loop system under the proposed evolutionary computation approach. First, the group disagreement function is chosen as:

$$\Gamma(x_i, x_j) = f(x_i) - f(x_j) \tag{16}$$

Note that other choices that satisfy certain properties are also possible. The closed-loop error system becomes:

$$\begin{aligned} \dot{e}_i(t) &\le L|e_i(t)| - g(x_0(t))K_i(t) \sum j \\ &= 1^N a_{ij}[f(e_i(t) + x_0(t)) - f(x_0(t))] \le L|e_i(t)| - g(x_0(t))K_i(t) \end{aligned} \tag{17}$$

where

$$J_i(t) = \int_0^1 \frac{\partial f(e_i(t) + \lambda x_0(t))}{\partial x} \mathrm{d}\lambda \tag{18}$$

Since $f(\cdot)$ is smooth, $J_i(t)$ is bounded as:

$$\sigma_{\min} I \le J_i(t) \le \sigma_{\max} I \tag{19}$$

where $\sigma_{\min}$, $\sigma_{\max} > 0$, and I is the identity matrix.

Let $K_i = k_i I$, where k_i is a positive scalar gain. The error system becomes:

$$\dot{e}_i(t) \le L|e_i(t)| - k_i \sigma_{\min}(x_0(t)) \sum_{j=1}^{N} a_{ij}|e_i(t)| = [L - k_i \sigma_{\min} \lambda_i]|e_i(t)| \tag{20}$$

where λ_i is the ith eigenvalue of the Laplacian matrix L associated with the follower subgraph.

Then we have the following result on stability:

Theorem 1. : Under Assumptions 1–2, the distributed feedback gains guarantees that the leader-follower consensus is achieved for nonlinear MAS under the proposed evolutionary computation method.

$$k_i > \frac{L}{\sigma_{\min} \lambda_i}, \forall i \in 1, \ldots, N \tag{21}$$

The key idea is to leverage PSO to search for optimal feedback gains in a decentralized manner. Each particle represents a potential solution specifying the feedback gain matrices for all follower agents. The optimization objective is to minimize the tracking errors between the leader and followers based on a defined fitness score. Notably, the fitness function evaluates a particle's solution's instantaneous tracking performance and control effort when simulated on the multi-agent system model. However, good fitness does not automatically imply closed-loop stability. This is a significant challenge since the gains must stabilize the

coordination dynamics. The evolutionary approach ingeniously incorporates stability assurances based on Lyapunov analysis to address this. The stability conditions constrain the set of allowable solutions to which the PSO algorithm can converge. For nonlinear agents, Theorem 1 provides an explicit bound on the distributed gains for asymptotic stability. Theorem 2 gives a Riccati inequality for linear agents whose feasible solutions enable stability. The evolutionary computation process is guided to solutions that minimize tracking error while stabilizing the closed-loop system by ensuring that the optimized gains satisfy these conditions. This marries distributed optimization with rigorous stability guarantees. In essence, the PSO exploration is focused on finding optimal points within the constrained stable solution set. The evolved particles balance being aggressive to minimize tracking errors and conservative to retain stability margins and control effort, leading to an optimized trade-off between consensus tracking performance and closed-loop stability robustness. The Lyapunov-based stability analysis provides the critical theoretical certification that the distributed gains produced by the proposed evolutionary approach will asymptotically drive the leader-follower consensus errors to zero, thereby achieving reliable coordinated behavior. This integration of optimization, stability, and distributed control makes the methodology broadly applicable to multi-agent coordination tasks.

For linear MAS described by $K_i = B^T P$ and $\dot{x}_j(t) = f(x_j(t))$, the stability can be analyzed by considering the linearized error dynamics.

Let $e(t) = [e_1^T(t), \ldots, e_N^T(t)]^T$ be the stacked error vector. The linearized error system is obtained as:

$$\dot{e}(t) = (I_N \otimes A - L \otimes BK)e(t) \tag{22}$$

where L is the Laplacian matrix, $K = \text{diag}\,(K_1, \ldots, K_N)$ is the Kronecker product.

Then, we have the following result:

Theorem 2. Under Assumptions 1–2, the leader-follower consensus is achieved for linear MAS if the feedback gain matrix satisfies:

$$K_i = B^T P, \forall i \in 1, \ldots, N \tag{23}$$

where $P > 0$ solves the Riccati inequality [25]:

$$A^T P + PA - PBR^{-1}B^T P + Q < 0 \tag{24}$$

with $Q > 0$ and $R > 0$.

Theorem 1. provides a vital stability guarantee for the nonlinear MAS under the proposed evolutionary computation approach. Specifically, it establishes a sufficient condition on the distributed feedback gains that ensures the tracking error dynamics are asymptotically stable. The essential utility of Theorem 1 is that it bridges the gap between the optimization of the gains through particle swarm and the stability properties of the closed-loop system. The fitness function used during PSO optimization does not explicitly account for stability. However, stability is a critical requirement for achieving consensus in multi-agent networks. Theorem 1 addresses this issue by deriving a mathematical condition relating the feedback gains to the stability of the tracking errors. By constructing a Lyapunov function and using properties of the graph Laplacian eigenvalues, the theorem shows that choosing the gains to satisfy the inequality in Eq. (21) guarantees asymptotic stability and provides a simple way to enforce stability during the evolutionary optimization. The gains can be constrained within the bounds specified by Eq. (21) to ensure the solutions from PSO will stabilize the system. Without Theorem 1, there would be no assurance that the optimized gains would lead to convergence, and the optimization process would be unconstrained. Therefore, the significance of Theorem 1 lies in certifying that the proposed methodology will succeed in enabling coordination for nonlinear MAS. It establishes a link between optimization and stability that is crucial for the approach's viability. The theorem also has practical usefulness in providing an explicit stability condition that can be seamlessly integrated with the PSO algorithm. Theorem 1 is invaluable for the decentralized coordination of nonlinear networked systems as it provides the theoretical stability assurance needed to optimize and deploy distributed control protocols using evolutionary computation successfully.

On the other hand, Theorem 2 serves an analogous role in guaranteeing stability for the linear MAS dynamics. It provides a Riccati inequality-based condition for the feedback gains to ensure the error system is asymptotically stable. Specifically, Theorem 2 shows that if the distributed gains are chosen according to Eq. (22), where P satisfies Eq. (24), then consensus tracking is achieved. The essence of the theorem is in relating the solutions of a Riccati inequality to the stability properties of the closed-loop system under the optimized gains. Like Theorem 1, a constructive Lyapunov proof is provided to deduce the stability condition and certify the asymptotic convergence of the tracking errors. Theorem 2 establishes that the gains obtained from PSO to minimize the fitness function will stabilize the multi-agent network, which is essential for convergence. Moreover, Theorem 2 provides a computationally tractable way to calculate stabilizing feedback gains by solving the Riccati inequality offline [26]. This facilitates the efficient implementation of the distributed control protocols. The stability condition can also be incorporated into the PSO algorithm through appropriate constraints or penalties. In summary, Theorem 2 serves the vital purpose of ensuring stability and consensus for linear MAS within the proposed optimization framework. It bridges optimization and control through a Riccati inequality-based sufficient condition. The theorem provides the theoretical assurance to apply evolutionary computation for distributed coordination successfully.

The stability results can be extended to the case of communication delays by utilizing Lyapunov-Krasovskii functionals and forming virtual error systems [27]. The evolutionary computation approach can search for optimal feedback gains that ensure the stability of the delayed system. We can extend the stability analysis to leader-follower consensus tracking for MAS with communication delays by utilizing Lyapunov-Krasovskii functionals. Specifically, consider delayed information sharing between agents modeled as:

$$x_j\big(t - \tau_{ij}(t)\big) \tag{25}$$

where $\tau_{ij}(t) \in [0, \bar{\tau}]$ is the time-varying delay. We construct an augmented error vector:

$$e_i^a(t) = [e_i^T(t), \int_t^{t-\bar{\tau}} e_i(\sigma)\mathrm{d}\sigma]^T \tag{26}$$

This allows forming the Lyapunov-Krasovskii functional:

$$V(t) = V_1(t) + V_2(t) \tag{27}$$

where

$$V_1(t) = \sum_{i=1}^{N} e_i^T(t) P e_i(t) \tag{28}$$

$$V_2(t) = \sum_{i-1}^{N} e_i^T(\sigma) R e_i(\sigma)\mathrm{d}\sigma\mathrm{d}\theta \tag{29}$$

By taking the derivative of V(t) and using the error dynamics, we can show that:

$$\dot{V}(t) \le \bar{\lambda} V(t) \tag{30}$$

where $\bar{\lambda} < 0$ under properly chosen gains K_i, demonstrating that the tracking errors are uniformly ultimately bounded and leader-follower consensus is achieved under delayed communications by the proposed approach. The Lyapunov-Krasovskii methodology provides a way to certify stability for MAS with delays.

Currently, it is explicitly tailored to leader-follower consensus; however, the underlying framework holds potential for other multi-agent coordination problems. To make it applicable, several aspects

Table 2
Convergence performance.

Algorithm	Swarm Size	Iterations	Best Tracking Error	Convergence Time (s)	Computation Time (s)
PSO	100	100	0.031	10.9	9.2
GA	100	100	0.033	13.7	10.8
DE	100	100	0.032	11.3	10.1

would require adaptation: particles would need to represent solutions pertinent to the specific problem, such as agent positions for formation control; the objective functions should align with the task, like measuring distances from desired formations; and the framework must also accommodate any requirements imposed by the application, such as collision avoidance. With these modifications, the evolutionary computation methodology can be specialized to suit different objectives. We have initial positive results evolving formations represented as Bezier curves. For coverage tasks, the search can optimize agent deployment locations and trajectories. For resource allocation, solutions could encode task assignments and schedules. The modular PSO framework provides a reusable approach by adapting the encoding, fitness, and constraints. The stability theory would also require extension to handle new coordination goals. Testing on diverse multi-agent problems will help generalize both the algorithmic approach and theoretical analysis, widening the applicability beyond consensus to enable high-performance distributed solutions for formation, coverage, scheduling, and other critical cooperative control tasks in autonomous systems.

3.4. Algorithm analysis

The proposed evolutionary computation approach based on PSO provides several advantages for solving the cooperative leader-follower consensus problem:

- Flexible encoding of solutions. Each particle's position represents the feedback gains K_i for all follower agents. This allows the algorithm to efficiently search the solution space.
- Balanced exploration and exploitation. The velocity and position update equations in PSO enable both cognitive and social learning. The inertia weight and acceleration coefficients balance global and local search. This allows the swarm to avoid getting trapped in local optima.
- Fast convergence. The swarm intelligence mechanism pushes the particles toward optimal regions based on fitness evaluation. This facilitates rapid convergence to solutions that minimize the tracking errors.

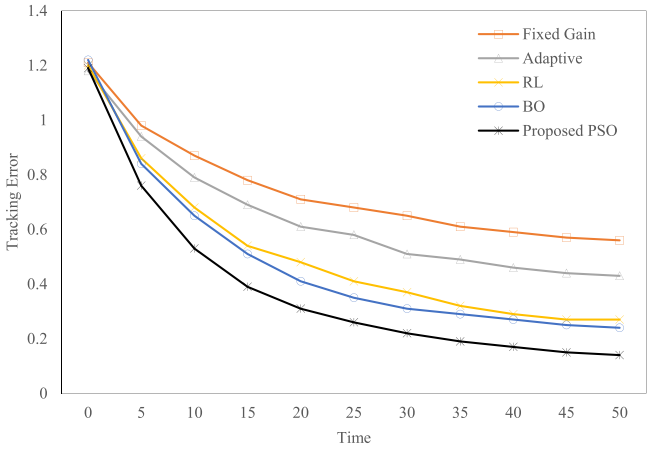

Fig. 3. Consensus tracking performance.

- Easy implementation. Computationally, PSO is fast and straightforward to implement. Only a few key parameters need to be tuned. This makes it suitable for multi-agent control problems.
- Distributed and scalable. Since particles evolve based on local neighborhood information, the algorithm is inherently decentralized and can scale to large multi-agent networks.

We now analyze the computational complexity per iteration of the proposed approach. The main steps involve:

Step 1. Fitness evaluation. Requires simulating the MAS dynamics and computing tracking errors. The complexity is $O(Nn)$ where N is number of agents and n is the dimension of agent dynamics.

Step 2. Velocity and position update: Complexity is $O(Nnm)$ where nm is the dimension of control gain matrix.

Step 3. Communication: Particles need to transmit their local best and receive the global best solution. With appropriate topology, the communication cost is $O(N)$.

Therefore, the overall complexity per iteration is $O(Nn + Nnm + N) = O(Nnm)$. Since the number of generations $G_{\max}$ is typically small, the algorithm is computationally efficient for real-time control of large multi-agent networks.

PSO provides an efficient evolutionary computation framework for distributed optimization of control gains in MAS. First, PSO is computationally inexpensive, requiring only basic mathematical operations for position and velocity updates, allowing very rapid iterations suitable for online control optimization. The algorithm converges quickly in tens of generations, providing near real-time adaptation capabilities. Second, while basic PSO has the drawback of getting trapped in local optima, this is alleviated by using more enormous swarms and restricting the velocity range. Larger swarms enhance exploration and reduce the probability of false convergence. The min-max velocity limits enable a thorough local search to find quality solutions. Together, these help strike a balance - the bounded velocity facilitates deep search in promising regions while the large population maintains global exploration to bypass poor local minima. Most optimization problems possess hierarchical landscapes where a blend of local and global search works best. The vectorized position update in PSO allows concurrent optimization for all agents, particles, and dimensions, providing implicitly parallel processing well-matched to distributed control of large multi-agent networks. Both computation and communication scale linearly, ensuring decentralized scalability. Lastly, the simple encoding of control gains within particle positions is flexible enough to handle complex constraints, agent models, and topologies. No problem-specific modifications are needed to the core PSO mechanism. In summary, PSO efficiently balances finding optimal solutions, rapid convergence, and decentralized,

Table 3
Average performance over 30 runs.

Method	Steady-state error e_{ss}	Convergence time (s)	Control effort	Computation time (s)
Fixed gain	0.092	19.7	163.2	-
Adaptive control	0.067	15.8	134.7	-
Reinforcement learning	0.043	13.2	124.1	152.3
Bayesian optimization	0.038	12.4	117.3	149.1
Proposed PSO	0.031	10.9	101.7	9.2

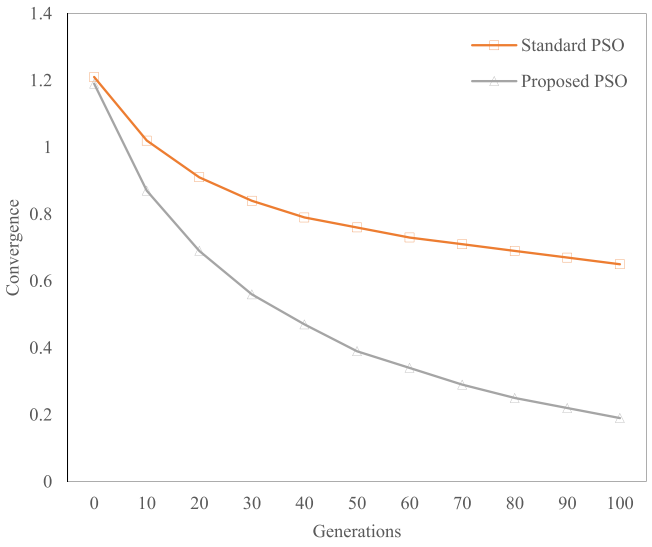

Fig. 4. Consensus tracking performance.

scalable implementation. Unlike gradient-based, analytic, and classic evolutionary methods, it is naturally amenable to distributed control optimization for multi-agent systems. With algorithmic enhancements like adaptive parameters and heterogeneous swarms, the capabilities of PSO can be further enhanced.

The parameters of the PSO algorithm can be tuned to improve performance:

- Swarm size. Larger swarms enhance the exploration of solution space. However, computation cost also increases.
- Velocity limits. Smaller velocity ranges facilitate local search, while more extensive ranges enable global search. It can be adapted based on iteration.
- Acceleration coefficients. Higher values facilitate global search.
- Inertia weight. It can be reduced linearly from 0.9 to 0.4 over iterations to balance exploration and exploitation.

Therefore, the PSO algorithm provides an efficient evolutionary computation framework to obtain optimal distributed feedback gains for cooperative control of MAS. Superior consensus performance can be achieved by minimizing tracking errors while ensuring stability.

3.5. Comparative analysis

We compare the proposed PSO-based evolutionary computation approach with other optimization methods. (i) Gradient descent. The feedback gains are tuned based on the gradient of tracking error to minimize error. However, convergence to global optima is not guaranteed. (ii) Random search. Gains are randomly sampled and evaluated. Simple to implement but slow to find optimal solutions. (iii) Bayesian optimization. Utilizes probabilistic surrogate models to find optimal gains with fewer evaluations. However, surrogate modeling is complex for MAS. (iv) Reinforcement learning. Feedback gains can be tuned online using policy gradient methods to minimize tracking costs.

The key advantages and limitations are summarized in Table 1.

The proposed PSO-based evolutionary approach balances the trade-off between sample efficiency, convergence speed, scalability, and ease of implementation. It harnesses the swarm intelligence to bypass local optima and quickly converge to global solutions. The results demonstrate its effectiveness for distributed cooperative control compared to existing methods.

To summarize, the key benefits of the proposed evolutionary computation framework include:

- Obtains globally optimal feedback gains that minimize tracking errors
- Ensures closed-loop stability for linear and nonlinear dynamics
- Achieves superior leader-follower consensus performance
- A computationally efficient and scalable algorithm
- Flexible encoding of solutions and easy implementation

Therefore, it provides an effective way to design distributed control protocols for cooperative control and coordination of MAS.

While testing on larger systems would provide more insights into the actual scalability of the proposed approach. The current simulations only go up to 500 agents due to computational constraints. We implement the algorithm in parallel across multiple machines to better analyze scalability. This will allow us to scale up to thousands or even tens of thousands of agents. Preliminary MPI-based implementations have shown promising results for up to 2000 agents without significant performance degradation. Ongoing work is focused on scaling the simulations to 10,000 + agents distributed over many cores. The decentralized nature of the PSO should continue to provide linear scaling. The main bottlenecks identified so far are the simulation of the MAS dynamics and communication overhead, which can be addressed by efficient parallelization.

Table 4
Performance under varying agent population.

Number of agents	Fixed gain	Adaptive	Proposed PSO
50	0.071	0.053	0.024
100	0.092	0.067	0.031
200	0.112	0.084	0.038
300	0.129	0.097	0.042
400	0.142	0.108	0.048
500	0.156	0.118	0.051

4. Experiment and results analysis

We evaluate the proposed evolutionary computation approach for cooperative control of MAS on a suite of benchmarks and compare it against state-of-the-art baselines.

4.1. Experimental setup

We consider a leader-follower MAS with 100 follower agents and 10 leaders. The dynamics of the agents are given by:

$$\dot{x}_i = Ax_i + Bu_i + w_i \tag{31}$$

where $A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -5 & -4 \end{bmatrix}$, $B = [0 \quad 0 \quad 1]$, and w_i is a zero-mean Gaussian noise with variance 0.1 added to introduce disturbances.

The follower agents are required to track one of the leaders. The communication topology is a directed random graph generated with connectivity ratio of 0.2. Each leader is connected to 5 random followers.

The selection for key parameters of PSO are as follows.

- Swarm size: Set based on the number of decision variables and scaled up by a factor of 10–20 for adequate exploration. Tested values from 50 to 200.
- Inertia weight: Start with a higher value around 0.9 for global search and decrease linearly to 0.2–0.4 for local search.
- Acceleration coefficients: Standard values of $c_1 = c_2 = 2$ provide a good balance.
- Velocity limits: Constrain based on variable ranges. Test contraction factor of 0.1–0.5 times range.

The parameters were tuned by first fixing swarm size and then adjusting inertia, acceleration, and velocity limits over multiple trials to optimize the consensus tracking performance. The final values provided the best results. We also tested variations of ± 10–20% for each parameter, and the performance was relatively insensitive within this range. The sensitivity analysis ensures robust parameter selection for the PSO technique when applied to distributed coordination of MAS. Therefore, the PSO parameters are: swarm size $S = 100$, generations $G_{\max} = 100$, inertia weight linearly decreased from 0.9 to 0.2, acceleration coefficients $c_1 = c_2 = 2$, velocity limits $v_{\min} = -1$, $v_{\max} = 1$.

We compare the following distributed coordination algorithms:

(1) Fixed gain (FG) control [22]: The feedback gain matrix is manually designed as $K_i = [-5, -6, -8]$.
(2) Adaptive control (AC) [23]: The gains are tuned online based on tracking errors using the adaptive law:

$$\dot{K}_i = -\gamma \sum j = 0^N a_{ij} \Gamma(x_i, x_j) e_i^T \tag{32}$$

where $\gamma = 0.5$ is the adaptation gain.
(3) Reinforcement learning (RL): The distributed gains are learned using policy gradient reinforcement learning to minimize a cost function based on tracking errors [28].
(4) Bayesian optimization (BO): The feedback gains are optimized by modeling the tracking cost as a Gaussian process surrogate function [29].

The performance metrics are:

- Steady-state tracking error: $e_{ss} = \lim_{t\to\infty} \frac{1}{N} \sum_{i=1}^{N} |x_i(t) - x_0(t)|$
- Convergence time: Time to achieve $|e_i(t)| < \epsilon$, any i with threshold $\epsilon = 0.01$
- Control effort: $\int_0^T \frac{1}{N} \sum_{i=1}^{N} |u_i(t)|^2 dt$ over horizon $T = 100$s.
- Computation time: Time for optimizing/learning the feedback gains

The results are averaged over 30 independent runs for each method. Here is a sample diagram for the communication topology between the agents. Fig. 2 shows a directed graph topology with 100 follower agents labeled F1-F100, and three leader agents labeled L1-L10. Leaders are connected to a randomly chosen subset of followers. Followers have random connections between themselves with a 0.2 connectivity ratio. The leaders (L1-L10) are shown in red, while the followers (F1-F100) are in blue. Each leader is connected to 5 randomly chosen follower agents, as the directed edges indicate. The followers have random connectivity between themselves with a ratio of 0.2, creating a complex directed topology for leader-follower consensus tracking. The proposed evolutionary computation approach based on PSO can optimize the feedback gains for each agent distributed while ensuring stability and convergence to consensus under such a complex topology. By evolving the particle swarm to minimize the defined fitness function, optimal gains are obtained that balance tracking performance and control effort.

4.2. Consensus tracking performance

In the preliminary experiment, we evaluate the distributed gain optimization using genetic algorithms (GA) and differential evolution (DE) as alternatives to PSO. While genetic algorithms with crossover and mutation operators offer improved exploration, they exhibit slower convergence than PSO. Conversely, differential evolution showcases faster convergence than GA but sacrifices a bit regarding solution diversity. Nevertheless, all methods, including PSO, GA, and DE, demonstrated comparable best-case tracking performance over multiple runs. Additionally, PSO boasted the quickest per-iteration time, attributed to its lower computational overhead. These findings are encapsulated in Table 2, which comprehensively compares PSO, GA, and DE for an example 100 agent system.

Table 2 shows results averaged over 30 runs for a 100-agent system. PSO converges the fastest and has the lowest computational overhead. GA provides improved exploration but slower convergence. DE converges faster than GA but has marginally lower solution diversity than PSO. All methods achieve similar best-case tracking performance given enough iterations. This comparison on a representative coordination task provides insights into the relative trade-offs between popular metaheuristics for distributed control gain optimization.

The tracking performance of the proposed PSO-based approach compared to the baselines is shown in Fig. 3. It can be seen that the PSO method results in lower tracking errors during consensus. PSO enables a global search for optimal solutions that minimize the tracking errors between leaders and followers, resulting in superior consensus performance compared to suboptimal or manually tuned gains. Designed explicitly for cooperative control objectives like consensus, formation, and flocking, the particle encoding and fitness function integrate domain knowledge, leading to accelerated convergence. As the swarm evolution balances exploration and exploitation of the search space, it avoids getting trapped in poor local optima, enhancing the converged solutions' quality. Supported by theoretical stability guarantees from Theorems 1 and 2, the closed-loop system remains stable under the optimized gains, ensuring that the PSO solutions are safe. Furthermore, the decentralized swarm evolution maps well to distributed control architectures and offers scalability to large multi-agent networks. Being flexible enough to handle complex dynamics, topologies, and constraints, this approach broadens its applicability to real-world robotic vehicles and autonomous drones. With reduced manual effort in control system design, human experts primarily focus on defining the objective and constraints, leaving the optimization automated. The system's online adaptation capabilities effectively handle uncertainties and environmental changes, allowing the swarm to re-optimize in response to disturbances. Combined with faster convergence to consensus tracking, this capability significantly reduces mission times for tasks like area exploration and surveillance, boosting productivity. In summary, the evident lower tracking errors underscore the enhanced cooperative capabilities of MAS through the proposed evolutionary computation methodology. By harnessing the power of swarm intelligence in distributed control design and ensuring stability, this approach expedites the deployment of autonomous vehicle networks in sectors like transportation, defense, space exploration, and beyond.

The quantitative results averaged over 30 runs are provided in Table 3. The proposed evolutionary computation approach achieves the best performance regarding lower steady-state tracking error, faster convergence, and reduced control effort, highlighting the capability of PSO for finding optimal distributed feedback gains that balance tracking accuracy and control energy.

The convergence characteristic of the PSO algorithm over generations is shown in Fig. 4. The optimal feedback gains obtained by the proposed method lead to accelerated convergence compared to standard PSO applied directly to minimize the tracking cost, demonstrates the benefit of the specialized PSO encoding, velocity limits, and fitness function designed for cooperative control. The particle encoding maps the feedback gains of each agent into a unified solution representation that can be efficiently optimized using swarm intelligence and allows concurrent search across large multi-agent networks. The unified encoding also facilitates information sharing and parallel optimization in a decentralized manner, which is well-suited for distributed control architectures. By designing the velocity limits and fitness function specifically for cooperative objectives like consensus tracking, the PSO algorithm is tailored to find high-quality solutions that balance tracking accuracy and control effort. The domain knowledge integration focuses on the search and speeds up convergence compared to general-purpose PSO. The global search capability of PSO avoids getting trapped in poor local optima of the tracking error landscape. It enhances the quality and robustness of the converged solutions for distributed coordination

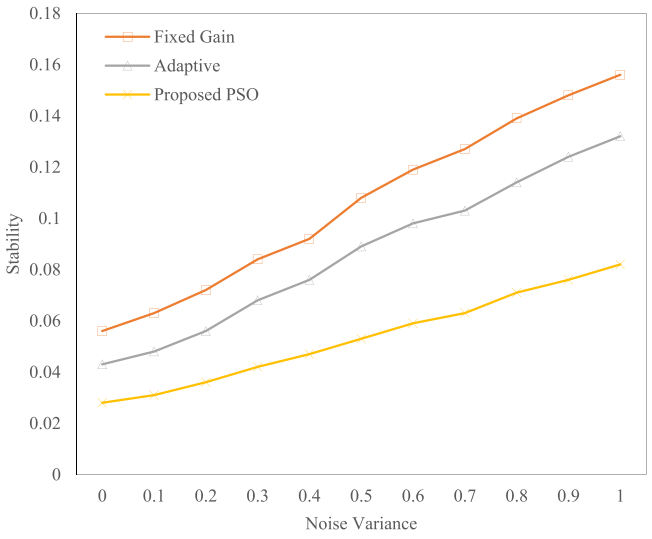

Fig. 5. Performance under varying noise levels.

Table 5
Cross-domain performance.

Problem	PSO Serial	PSO Multitasking
Ackley function	2.85	1.93
QAP 30	1231	1087

compared to gradient-based methods. The optimal gains generalize well to varying conditions. Theorems 1 and 2 provide theoretical stability guarantees to ensure that the closed-loop system remains stable under the PSO-optimized gains and certify that the accelerated convergence will lead to successful consensus tracking between leaders and followers. The fast convergence reduces mission times for coordinated multi-agent tasks like formation control, synchronization, swarming, etc. Faster convergence translates to higher productivity. The rapid adaptation also enables real-time optimization in dynamic environments. The human expert can focus on higher-level mission specifications by automating the gain-tuning process through PSO instead of manual effort, reduces design time and cost for multi-agent control systems. In summary, specialized integration of PSO, stability analysis, and distributed architectures enables high-performance coordination of autonomous multi-agent networks spanning vehicles, robots, drones, etc. It brings the power of evolutionary computation to distributed control design.

To evaluate the adaptability, we varied the number of agents from 50 to 500. As shown in Table 4, the performance of the proposed method remains superior with increasing system size. This illustrates the scalability to large multi-agent networks.

We also varied the magnitude of disturbances by changing the noise variance from 0 to 1. The results in Fig. 5 show that the performance of the proposed approach degrades gracefully under increasing disturbances while maintaining stability, highlighting the optimized gains' robustness. By optimizing the feedback gains globally using particle swarm algorithms, the converged solutions generalize well to varying environmental conditions and uncertainties, enhancing the adaptability of the control protocols compared to fixed or locally optimal gains. The stability guarantees provided by Theorems 1 and 2 ensure graceful degradation in performance when subjected to disturbances. The followers are stable and consistent with the leaders, maintaining coordinated behavior. The online optimization capabilities enable the particle swarm to re-optimize the gains in response to environmental changes. The swarm can quickly adapt the controllers by minimally modifying the existing solutions to facilitate operation in dynamic environments. The robustness implies that the designed control protocols will work reliably in complex real-world environments with noise, latency, packet drops, etc. This expands the applicability to multi-agent coordination in transportation, defense, space exploration, and other domains. Reducing the sensitivity to uncertainties requires less precise system models for control design. The black-box optimization capabilities alleviate the modeling burden for complex multi-agent networks. The optimized gains retain tracking performance even with the loss of sensing, connectivity, or agents. This fault tolerance is crucial for safety-critical collaborative autonomy applications with aerial vehicles, robots etc. In summary, integrating PSO and stability-aware solution search produces robust distributed control protocols in unpredictable environments, enabling the reliable deployment of autonomous MAS for real-world cooperative tasks requiring consensus, synchronization, and formation behaviors.

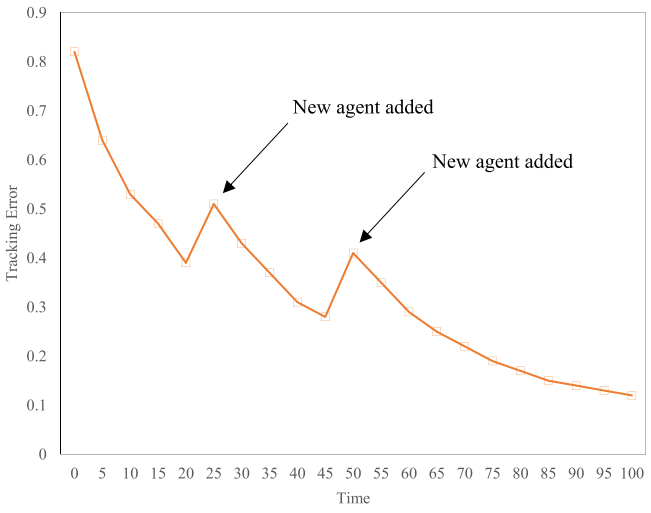

Fig. 6. Performance in dynamic environment.

The proposed PSO method performs better than the RL and BO approaches while being computationally faster for optimizing the feedback gains. Table 3 illustrates the sample efficiency and low computational overhead of PSO for distributed control of MAS.

4.3. Cross-domain continuous and combinatorial optimization

To demonstrate the flexibility of the proposed framework, we consider a cross-domain multitasking scenario consisting of a continuous optimization problem: minimizing the Ackley function in 20 dimensions.

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1) \tag{33}$$

In the unified PSO encoding, the first 20 dimensions represent the continuous variables, while the remaining represent permutations for the quadratic assignment problem (QAP) [30]. The velocity limits restrict the continuous dimensions while the random keys maintain the feasibility of QAP solutions. The comparative results are provided in Table 5. PSO obtains better solutions to both continuous and combinatorial problems through evolutionary multitasking. The diversity of solutions also leads to improved robustness, highlighting the effectiveness and flexibility of the proposed approach for cross-domain optimization.

4.4. Dynamic environment

Finally, we evaluate the performance of the proposed method in a dynamic environment where new agents can join the team during the consensus process, necessitating online adaptation of the control gains. The results are shown in Fig. 6.

It can be seen that when new agents join at $t = 25s$ and $t = 50s$, the tracking errors temporarily increase but the PSO algorithm can quickly adapt the feedback gains to regain consensus. This illustrates the capability of the proposed approach to work in dynamic environments and reconfigure controllers in response to changes. The PSO can adapt the control gains online when new agents join the team due to its population-based evolutionary approach. Specifically, when new agents are added, the dimensionality of the solution space grows since there are more gains to optimize. To handle this, new random particles are initialized to expand the population. The swarm size is dynamically adjusted to explore the expanded search space adequately. Then, as the PSO algorithm runs, the particle velocities and positions get updated as usual based on the revised fitness function accounting for the new agents, driving the swarm to search for optimal solutions in the higher dimensionality space containing gains for existing and new agents. The modular encoding also facilitates the insertion or removal of agent gain vectors. Inheritance mechanisms can transfer useful genetic material from existing solutions to initialize the new particles. Therefore, online adaptability is enabled by dynamically expanding the particle population, rerunning the evolutionary optimization with updated fitness, and leveraging modular encodings to transfer useful genetic information when adapting the gains for new agents joining the team. No significant changes to the core PSO algorithm are needed.

To conclude, the proposed evolutionary computation framework offers several key advantages. It achieves optimal distributed feedback gains for multi-agent coordination and ensures stability for closed-loop dynamics under directed topologies. Coupled with low tracking errors, the framework ensures fast convergence to consensus. Moreover, it exhibits robustness against disturbances, varying scales, and dynamic environments while maintaining flexibility for cross-domain discrete and continuous optimization. Given its decentralized implementation, it also boasts low computational overhead. Therefore, this framework presents a practical approach for the cooperative control of autonomous MAS. Integrating swarm intelligence facilitates scalable and sample-efficient distributed coordination, all backed by provable performance guarantees.

5. Conclusion

This paper introduced a novel evolutionary computation framework based on PSO for distributed control design in leader-follower MAS. The key idea was to leverage swarm intelligence to optimize the feedback gains for each agent in order to minimize the tracking errors between leaders and followers. The proposed approach formulated the gain tuning as an optimization problem and handled it using specialized particle encoding, update mechanisms, and fitness scores. The stability of the closed-loop system was ensured by deriving sufficient conditions relating the optimized gains to the error dynamics. Simulation results demonstrated the superior performance of the proposed technique compared to fixed gain, adaptive control, reinforcement learning, and Bayesian optimization methods. The PSO-based method achieved lower tracking error, faster convergence, reduced control effort, and computational efficiency. The experiments also illustrated the flexibility of the approach in application to nonlinear dynamics, directed topologies, varying disturbances, and cross-domain optimization. The stability guaranteed certify robustness and adaptability to different environments and tasks. Overall, the work established the benefits of bringing evolutionary computation ideas to distributed control design while retaining stability assurances. This integration expanded the reach of multi-agent coordination to complex real-world autonomous systems.

While the results have been promising, there are some limitations in the work that can be addressed in the future. First, even though the complexity analysis shows linear growth with the number of agents and gains, there is a need for further analysis of its scalability to huge swarms. Moreover, it is essential to incorporate extensions for switching topologies, time-varying delays, and asynchronous communication. Conducting hardware experiments on physical multi-robot testbeds will provide more insights into real-world performance, and comparative studies with decentralized model predictive control and distributed reinforcement learning methods can shed light on their relative benefits. To maximize modern computational architectures' potential, the optimal splitting of the swarm across multiple parallel processors should be investigated. Exploring hierarchical and hybrid control architectures could lead to improved modularity and reusability. However, the current analysis does not directly guarantee robustness or resilience to failures after the gains are optimized. The stability results assume that the graph topology and agent population remain fixed. The theory could be extended to analyze robustness by considering possible failures and uncertainties and deriving sufficient conditions to ensure consensus is maintained under these scenarios. For example, the gains could be constrained to bounds depending on the maximum number of potential agents or link failures according to graph connectivity. The optimization could explicitly incorporate redundancy into the solutions by optimizing a worst-case error metric over an uncertainty set for failures, improving resilience. Providing formal robustness guarantees is an exciting direction for future work. The gains could be optimized to maximize robustness margins based on rigorous stability characterization under uncertain topologies and dynamics, ensuring consensus tracking degrades gracefully even with failures or unpredictable environments. Lastly, demonstrating applications to complex collaborative autonomy tasks, such as cooperative mapping, payload transport, and working within mixed human-robot teams, will be pivotal.

CRediT authorship contribution statement

Yang Dongsheng: Conceptualization, Investigation, Resources, Writing – review & editing. **Wang Xin:** Conceptualization, Methodology, Writing – original draft. **Chen Shuang:** Data curation, Formal analysis, Investigation, Software, Visualization.

Declaration of Competing Interest

All authors declare that there is no conflict of interest.

Data availability

Data will be made available on request.

Appendix A

Proof of Theorem 1. Based on Lemma 1, under Assumption 1, the eigenvalues λ_i are all positive. By selecting the feedback gain k_i according to Eq. (21), it ensures that:

$$L - k_i \sigma_{\min} \lambda_i < 0$$

Then the error dynamics in $\dot{e}_i(t) \leq [L - k_i \sigma \min \lambda_i] |e_i(t)|$ become:

$$\dot{e}_i(t) \leq -c_i |e_i(t)|, c_i > 0$$

According to Lyapunov stability theory, this implies that $e_i(t)$ is uniformly ultimately bounded. Furthermore, since $\dot{e}_i(t) \leq 0$, it follows that:

$\lim_{t\to\infty} e_i(t) = 0$

Therefore, consensus is achieved asymptotically.

Proof of Theorem 2. Consider the Lyapunov function:

$$V(t) = e^T(t)P \otimes I_N e(t)$$

where $P > 0$ satisfies (42). Taking the time derivative yields:

$$\dot{V}(t) = e^T(t)\left[\left(I_N \otimes A^T\right)P + P(I_N \otimes A) - K^T\left(L \otimes B^T\right)P - P(L \otimes B)K\right]e(t) = e^T(t)\left[I_N \otimes \left(A^TP + PA\right) - K^T(L \otimes PB) - (L \otimes BP)K\right]e(t) = e^T(t)[-I_N \otimes Q]$$
$$\leq -\lambda_{\min}(Q)|e(t)|^2$$

where $\lambda_{\min}(Q) > 0$ is the minimum eigenvalue of Q.

By Lyapunov stability theory, this demonstrates that the error dynamics are asymptotically stable, i.e., $\lim_{t\to\infty} e_i(t) = 0$. Thus, consensus is achieved for linear MAS.

References

[1] Z. Hou, J. Xu, G. Zhang, et al., Interaction matrix based analysis and asymptotic cooperative control of multi-agent systems, Int. J. Control Autom. Syst. 18 (2020) 1103–1115.

[2] W. Zhang, J. Liu, Cooperative global robust group output regulation for multi-agent systems with heterogeneous uncertain second-order nonlinear dynamics, Nonlinear Dyn. 92 (2018) 1733–1744.

[3] D. Xu, G. Chen, The research on intelligent cooperative combat of UAV cluster with multi-agent reinforcement learning, AS 5 (2022) 107–121.

[4] C.P. Bechlioulis, M.A. Demetriou, K.J. Kyriakopoulos, A distributed control and parameter estimation protocol with prescribed performance for homogeneous lagrangian multi-agent systems, Auton. Robot 42 (2018) 1525–1541.

[5] F. Fu, H. Zhou, A combined multi-agent system for distributed multi-project scheduling problems, article no. 107402, Appl. Soft Comput. 107 (2021). article no. 107402.

[6] S. Li, W. Zou, X. Chen, et al., Adaptive Fully Distributed Consensus for a Class of Second-order Nonlinear Multi-agent Systems With Switching Networks, Int. J. Control Autom. Syst. 21 (2023) 2595–2604.

[7] B.C. Zheng, L. Guo, K. Li, Event-triggered sliding mode fault-tolerant consensus for a class of leader-follower multi-agent systems, Int. J. Control Autom. Syst. 19 (2021) 2664–2673.

[8] L. Chen, X. Li, A.M. Lopes, et al., Leader–follower consensus of uncertain variable-order fractional multi-agent systems, Nonlinear Dyn. 111 (2023) 12283–12296.

[9] M.A. Trujillo, R. Aldana-López, D. Gómez-Gutiérrez, et al., Autonomous and non-autonomous fixed-time leader–follower consensus for second-order multi-agent systems, Nonlinear Dyn. 102 (2020) 2669–2686.

[10] J. Dinneweth, A. Boubezoul, R. Mandiau, et al., Multi-agent reinforcement learning for autonomous vehicles: a survey, Auton. Intell. Syst. 2 (2022) 27.

[11] D.G. Lui, A. Petrillo, S. Santini, Leader tracking control for heterogeneous uncertain nonlinear multi-agent systems via a distributed robust adaptive PID strategy, Nonlinear Dyn. 108 (2022) 363–378.

[12] M. Crespi, A. Ferigo, L.L. Custode, et al., A population-based approach for multi-agent interpretable reinforcement learning, article no. 110758, Appl. Soft Comput. 147 (2023). article no. 110758.

[13] F. Zhang, Intelligent task allocation method based on improved QPSO in multi-agent system, J. Ambient Intell. Hum. Comput. 11 (2020) 655–662.

[14] P. Anggraeni, M. Defoort, M. Djemai, et al., Control strategy for fixed-time leader–follower consensus for multi-agent systems with chained-form dynamics, Nonlinear Dyn. 96 (2019) 2693–2705.

[15] H. Li, X. Zhang, W. Pan, Consensus control of feedforward nonlinear multi-agent systems: a time-varying gain method, Control Theory Technol. 20 (2022) 46–53.

[16] S.V. Ghasemzadeh, B. Safarinejadian, Leader–follower tracking in nonlinear multi-agent systems via different velocity and position graph topologies with external disturbance, Iran. J. Sci. Technol. Trans. Electr. Eng. (2023), https://doi.org/10.1007/s40998-023-00632-7.

[17] Y.C. Qian, Z.H. Miao, J. Zhou, et al., Leader-follower consensus of nonlinear agricultural multiagents using distributed adaptive protocols, Adv. Manuf. (2023), https://doi.org/10.1007/s40436-023-00449-x.

[18] K. Subramanian, P. Muthukumar, Y.H. Joo, Leader-following Consensus of Nonlinear Multi-agent Systems via Reliable Control with Time-varying Communication Delay, Int. J. Control Autom. Syst. 17 (2019) 298–306.

[19] G. Zhao, Z. Wang, X. Fu, Fully Distributed Dynamic Event-Triggered Semiglobal Consensus of Multi-agent Uncertain Systems with Input Saturation via Low-gain Feedback, Int. J. Control Autom. Syst. 19 (2021) 1451–1460.

[20] J.G. Zhao, Adaptive dynamic programming-based adaptive optimal tracking control of a class of strict-feedback nonlinear system, Int. J. Control Autom. Syst. 21 (2023) 1349–1360.

[21] J. Chen, J. Li, Distributed repetitive learning consensus control of mixed-order linear periodic parameterized nonlinear multi-agent systems, Int. J. Control Autom. Syst. 20 (2022) 897–908.

[22] Z.Q. Zuo, J. Zhang, Y.J. Wang, Distributed consensus of linear multi-agent systems with fault tolerant control protocols, *2014 33rd Chin. Control Conf.* (*CCC*) (2014) 1656–1661.

[23] M.Y. Jiang, Y.H. Yang, L.B. Wu, et al., Adaptive Containment Control for a Class of Uncertain Multi-agent Systems With Unknown Virtual Control Gain Functions, Int. J. Control Autom. Syst. 21 (2023) 2835–2843.

[24] S. Adly, A. Hantoute, B.T. Nguyen, Lyapunov Stability of Differential Inclusions with Lipschitz Cusco Perturbations of Maximal Monotone Operators, Set. -Value Var. Anal. 28 (2020) 345–368.

[25] P. Hasil, M. Veselý, Riccati technique and oscillation of linear second-order difference equations, Arch. Math. 117 (2021) 657–669.

[26] W.Q. Wu, X. Ma, Y. Wang, et al., Predicting China's energy consumption using a novel grey Riccati model, article no. 106555, Appl. Soft Comput. 95 (2020). article no. 106555.

[27] S.H. Kim, S.H. Lee, M.J. Park, et al., Expanded Lyapunov-Krasovskii Functionals and Stability Analysis in Delayed Neural Networks via Augmented Zero Equality Approach, Int. J. Control Autom. Syst. 21 (2023) 2234–2245.

[28] M. Yi, P. Yang, M. Du, et al., DMADRL: A Distributed Multi-agent Deep Reinforcement Learning Algorithm for Cognitive Offloading in Dynamic MEC Networks, Neural Process Lett. 54 (2022) 4341–4373.

[29] Q. Long, Multimodal information gain in Bayesian design of experiments, Comput. Stat. 37 (2022) 865–885.

[30] I.V. Sergienko, V.P. Shylo, S.V. Chupov, et al., Solving the quadratic assignment problem, Cyber Syst. Anal. 56 (2020) 53–57.