Journal of Advances in Engineering and Technology
ISSN(O):3005-7507
ISSN(P):3005-7493
Vol. 2 No.1 (2025)

Differential Dynamics Modeling and Simulation Analysis of Multi-Agent Cooperative Motion

Shao Qiang*
Gaston Day School Shanghai Shangde, Shanghai 201315, China
Corresponding author: Shao Qiang , sost12822@163.com

Abstract: With the widespread application of multi-agent systems (MASs) in fields such as drone formations, autonomous driving, and robotic swarms, achieving efficient collaboration and stable motion among agents has become a key research focus. This study begins by describing the vertices of agents relative to the formation centroid to enable collision avoidance and formation shape tracking control. Using the Lyapunov direct method, a heat-equation-based collective dynamics model for multi-agent systems is established, providing stability criteria for the model and a leader-follower algorithm. The model enables the transformation from continuous multi-agent systems to discrete systems, completing the cooperative motion of real multi-agent systems. Simulation analysis verifies the effectiveness of the proposed model and control strategy. In a typical simulation scenario, follower agents achieve consensus with leader agents within approximately 10 seconds, with the number of path nodes reduced to just six, zero obstacle collisions, and a computation time of only 49.6 seconds. The proposed control method significantly enhances the cooperative efficiency and motion stability of multi-agent systems under limited information exchange and complex environmental conditions, offering robust theoretical support for the collaborative control of future intelligent systems.
Keywords: Multi-agent Systems; Cooperative Motion; Differential Dynamics Model; Model Stability; Follower Algorithm
Published: Jan 22, 2025
DOI: https://doi.org/10.62177/jaet.v2i1.151

1. Introduction

In recent years, multi-agent systems (MASs) have been widely applied in various fields such as drone formations, autonomous driving, and robotic swarms [1]. Compared to single-agent systems, MASs are more efficient and reliable in task execution and can accomplish complex and challenging tasks that are difficult or even impossible for single-agent systems [2]. When multiple agents perform tasks, they typically achieve the overall objective optimally through local information sharing and mutual cooperation [3-4]. Research on the cooperative motion of multi-agent systems provides the necessary theoretical foundation for the organization and autonomous coordination control of complex MASs, while also having a profound impact on practical applications such as intelligent transportation, robotic swarms, and environmental monitoring.

However, in the control of cooperative motion for MASs, designing an efficient and robust control strategy that enables agents to collaborate effectively in dynamic environments remains a challenging problem. In particular, complex real-world scenarios introduce challenges such as interactions between agents, sensor errors, and environmental disturbances, making the control of MASs more intricate. Therefore, optimizing the cooperative motion of MASs while ensuring system stability has become a key issue in current research. The differential dynamics model, as an important continuous-time control framework, effectively describes the motion behavior of MASs and their interactions. Unlike traditional discrete control methods, the differential dynamics model transforms the agents' motion and control strategies into differential equations, allowing a more precise mathematical representation of their dynamic behavior. This model enables the design of cooperative control strategies suitable for complex environments, ensuring collaboration and motion consistency among agents in dynamic scenarios.

This study builds on the foundation of formation control and system behavior control for MASs, constructing a distributed differential dynamics model for cooperative motion control using the Lyapunov method. Based on this model, control strategies adaptable to different scenarios are designed, and the effectiveness of the proposed model and control strategies is validated through simulation analysis.

2. Cooperative Motion Formation Path Planning for Multi-Agent Systems

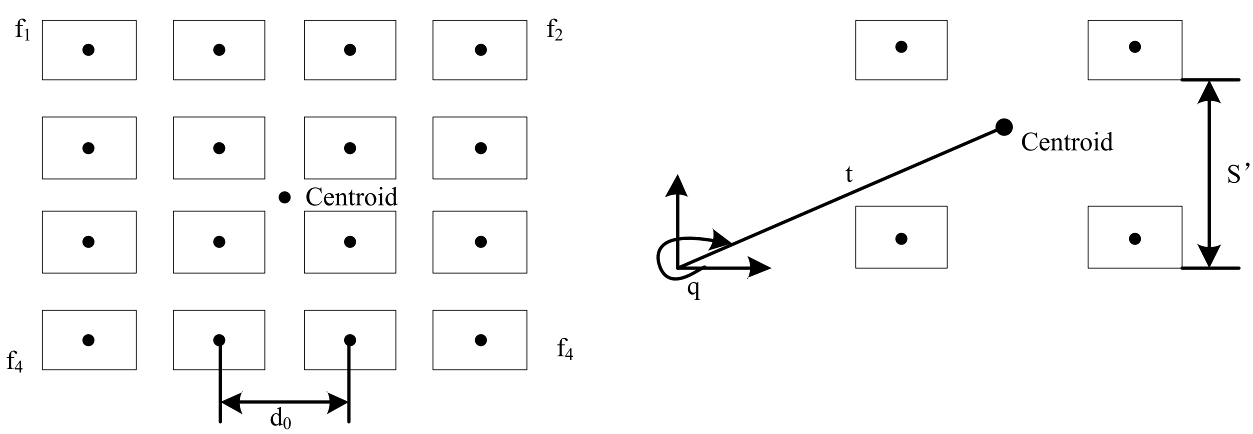
2.1 Formation Control of the Group

The control of group formation behavior is to achieve collision-free movement in multi-agent formation tasks while maintaining a specific formation shape [5]. First, multiple agents are organized into a square shape for movement. The model is based on a square formation structure f composed of multiple agents. The definition of the cooperative motion formation is shown in Figure 1. In this structure, a set $\left\{f_1^i, f_2^i, \cdots f_n^i\right\}$ is used to describe the specific position of the agent in the formation shape. In this paper, the position is the four corners of the four vertices of the square. A distance parameter d is introduced to represent the distance between any two agents in the formation. d allows the agents to maintain a certain distance in the group formation, thereby achieving the stability of the group formation.

The kinematics of each agent are assumed to be the same, and $\left\{p_1^i, p_2^i, \cdots p_n^i\right\}$ is used to represent their positions. The outer contour of the formation is represented by the intersection points formed by the positions of the agents in the formation. The smallest convex polygon that surrounds all agents at the intersection points directly displays the overall shape and size of the formation. The formation model can be constructed by combining the formation positions of the agents with the relative vertices. The formation optimization variables can be determined by $x = \left(t, s'q\right)$. The variables in this paper include the relative positions of the agents, directional accelerations, and the cumulative deviations between the agents and the target states. The positions of the agents and the vertices of the formation are represented by the above variables as follows:

$$
\begin{aligned}
p_j^i &= t + s' rot\left(q, p_j^i\right) \forall j \in \left[1, n\right] \\
f_j^i &= t + s' rot\left(q, f_j^i\right) \forall j \in \left[1, n_i\right]
\end{aligned} \tag{1}
$$

Figure 1 Definition of coordinated movement formation



(a) Convex hull position and formation vertices (b) Representation of formation motion space

All these position and orientation data combined form the configuration of motion at a given moment, representing the set of all possible positions and orientations. In a multi-agent system, each agent controls its respective position and orientation to ensure the entire formation remains consistently stable.

2.2 Behavior Control of Intelligent Group Systems

To prevent collision and obstacle avoidance issues during the cooperative motion paths of multi-agent systems, this study adopts an individual-based Lagrangian modeling framework. The behavior of the group is understood as the result of interactions, where individuals attract each other when far apart and repel each other when close. This ensures a safe distance is maintained between individuals, accounting for the physical space occupied by each individual. Specifically, individuals attract each other at longer distances and repel at shorter distances. Modeling static and dynamic obstacles in the environment effectively captures the fundamental motion characteristics of biological groups in nature, ensuring a safe distance between individuals to avoid collisions during movement. In the collective motion of intelligent groups, the trajectory generated by the dynamics model of any individual serves as a reference trajectory for other agents to follow or track. This approach ensures coordinated path tracking and maintains orderly movement within the entire intelligent group system.

3. Construction of differential dynamics model for multi-agent coordinated motion

3.1 Multi-agent distribution of Lyapunov method

In the study of multi-agent systems, stability analysis is crucial. Lyapunov indirection is to introduce Lyapunov functions with generalized energy properties, and then analyze the monotonicity of the function by means of derivation, and then judge its stability. Suppose there is a nonlinear system of equations: In a system where the position, velocity, and acceleration of a multi-agent are jointly described, so that $f\left(x_e\right)=0$ satisfies in the neighborhood δ of the equilibrium point x_e . If a continuously differentiable positive definite function $V\left(x\right)$ defined in the neighborhood δ of x_e can be found, and satisfies in the neighborhood δ of the equilibrium point x :

$$\dot{V}\left(x\right)=\frac{\partial V}{\partial x}\frac{dx}{dt}<0, x\in\delta \tag{2}$$

This equilibrium point is called a globally asymptotically stable equilibrium point, which means that no matter how far the initial state of the system is from the equilibrium point, the system will eventually tend to this equilibrium point as time evolves.

3.2 Modeling of multi-agent systems

3.2.1 Modeling principles and ordinary differential models

Taking the heat equation as an example, the position of the agent can be represented by the state variable $u\left(t,a\right)$, where the real part represents the horizontal coordinate and the imaginary part represents the vertical coordinate. There are multiple agents distributed in a two-dimensional plane. In order to achieve the formation goal of the multi-agent system in the plane, the multi-agent system constructed with an ordinary differential model usually uses $\left(x,y\right)$ to represent the position of each agent. For the agents in the plane, the consistency control law is:

$$\dot{x}_i=\sum_{j\in\delta_i}a_{ij}\left(x_j-x_i\right) \tag{3}$$

$$\dot{y}_i=\sum_{j\in\delta_i}a_{ij}\left(y_j-y_i\right) \tag{4}$$

Each agent can only exchange information with other agents within its communication range, and these interactive agents constitute δ_i. a_{ij} represents the connection weight between agents i and j, reflecting the intensity of information interaction between the two agents [6].

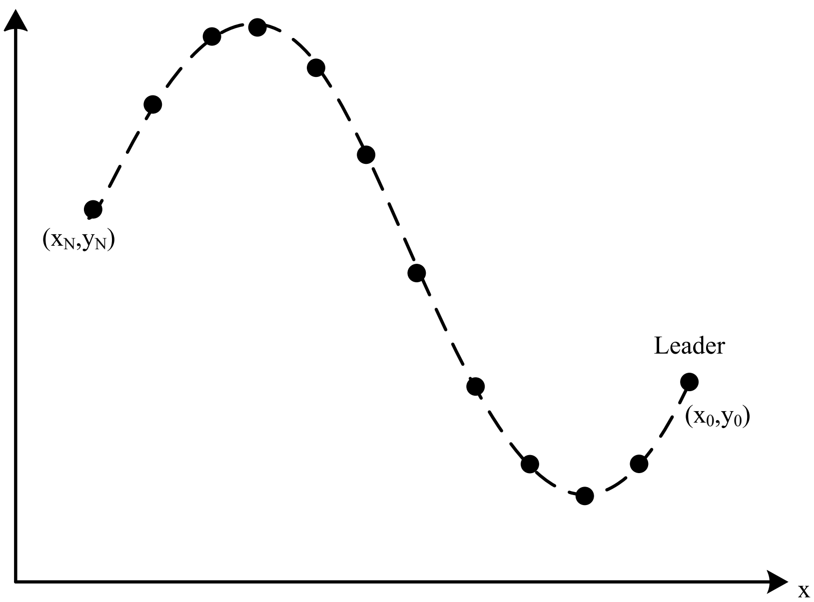
3.2.2 Partial differential equation modeling and collective dynamics model

The originally discrete multi-agents are mapped to continuous space. By introducing the state variable $u\left(t,a\right)=x\left(t,a\right)+iy\left(t,a\right)$, the real part is used to represent the horizontal coordinate position of the agent, and the imaginary part represents the vertical coordinate position[7-8]. Collective differential dynamics model of multi-agents:

$$u_t\left(t,a\right)=k_a\left[u_{aa}\left(t,a\right)-u_{aa}^d\left(t,a\right)\right] \tag{5}$$

Among them, $u_t=\frac{\partial u}{\partial t}$ reflects the change of the agent state over time; $u_{aa}=\frac{\partial^2 u}{\partial a^2}$ reflects the changing trend of the agent state in spatial distribution; k represents the control gain, which determines the response intensity of the system to the deviation; $u^d(a)$ represents the expectation function, which can be set to achieve complex formations in collective formation motion tasks. The cooperative motion of a multi-agent system is illustrated in Figure 2. By adjusting the desired function, the multi-agent system can be configured into various specific shapes. Notably, in this study, the partial differential equation modeling and collective dynamics model frequently employ a leader-follower algorithm for multi-agent formations. This algorithm enables precise positioning within the system, with predesignated leader drones equipped with advanced navigation and mapping devices to acquire global information and continuously adjust their direction, position, and speed. Other agents move solely based on the leader's information, achieving the cooperative motion of the entire multi-agent system.

Figure 2 Coordinated motion of a multi-agent system



4. Simulation analysis and result discussion

4.1 Simulation platform and experimental setting

This study used a simulation platform based on MATLAB/Simulink for experiments. Table 1 shows the experimental environment, which lists the specific information of the computer hardware configuration and software environment used for simulation.

Table 1 Experimental environment

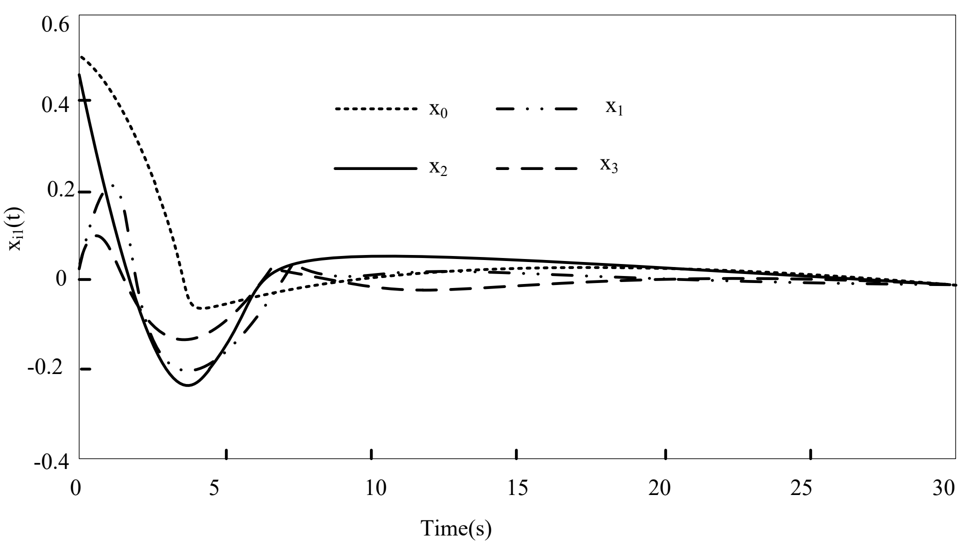
Project		Configuration and description
Hardware Configuration	Intel Core i7-10700K, 8 cores and 16 threads, 3.8 GHz	Intel Core i7-10700K, 8 cores and 16 threads, 3.8 GHz
	32 GB DDR4 3200 MHz	32 GB DDR4 3200 MHz
	NVIDIA GeForce RTX 3060 12GB	NVIDIA GeForce RTX 3060 12GB
	1TB NVMe SSD	1TB NVMe SSD
	Windows 10 Pro 64-bit	Windows 10 Pro 64-bit
Software Configuration	MATLAB R2023b, Simulink 2023b	MATLAB R2023b, Simulink 2023b
	Control System Toolbox, Optimization Toolbox, Reinforcement Learning Toolbox, ADP Toolbox	Control System Toolbox, Optimization Toolbox, Reinforcement Learning Toolbox, ADP Toolbox
	Simulink for multi-agent system modeling, Stateflow for control flow and logic design	Simulink for multi-agent system modeling, Stateflow for control flow and logic design
	MATLAB, Simulink (graphical modeling)	MATLAB, Simulink (graphical modeling)
Simulation Settings	0.01 s	0.01 s
	50 s	50 s

4.2 Analysis of multi-agent cooperative motion trajectory

It can be seen that in the differential dynamics model of cooperative motion of agents using the proposed Lyapunov function method, the system states of all agents can reach a consensus with the leader node after about 10 seconds, as shown in Figure 3. Figure 3 (a) shows the position motion state of the agent. It can be seen that at the initial moment, the x1-x0 states of agents with different numbers (including leaders and followers) have different starting values; at 4 seconds, x1 is at -0.18, x2 is at -0.20, x3 is at -0.22, and x0 is at 0.13, reflecting the initial differences in the states of the four individual agents. As time goes on, at a time node of about 9.5 seconds, the position motion states of the four individual agents tend to a common value of 0, and the state of the leader x0 has completely served as the reference target state of other agents x2-x4. Figure 3 (b) shows the directional motion state of the agent. Similarly, the directional motion state values of each agent are different at the beginning. At 3 seconds, x1 is in the direction of -0.19, x2 is in the direction of -0.31, x3 is in the direction of -0.39, and x0 is in the direction of 0.20, which also shows the initial difference in the states of the four individual agents. As time gradually increases from 0 seconds, at about 7.8 seconds, the directional motion state curves of all agents gradually converge to the same, indicating that the system states of all agents reach a consensus with the leader node at this point in time, further proving the effectiveness of collaborative tracking.

Figure 3 Trajectories of four individual agents over time

(a) Position motion state trajectory of the agent



(b) Trajectory of the directional motion state of the agent

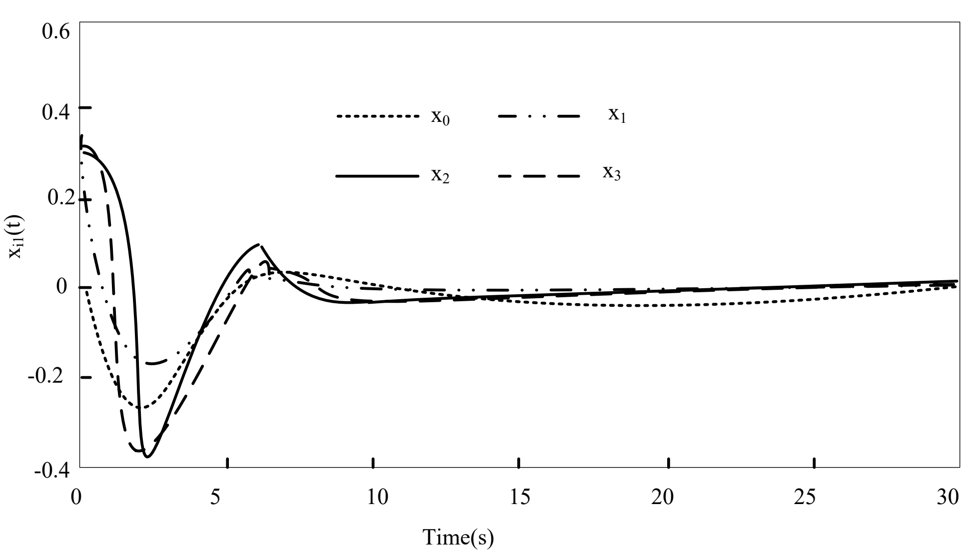
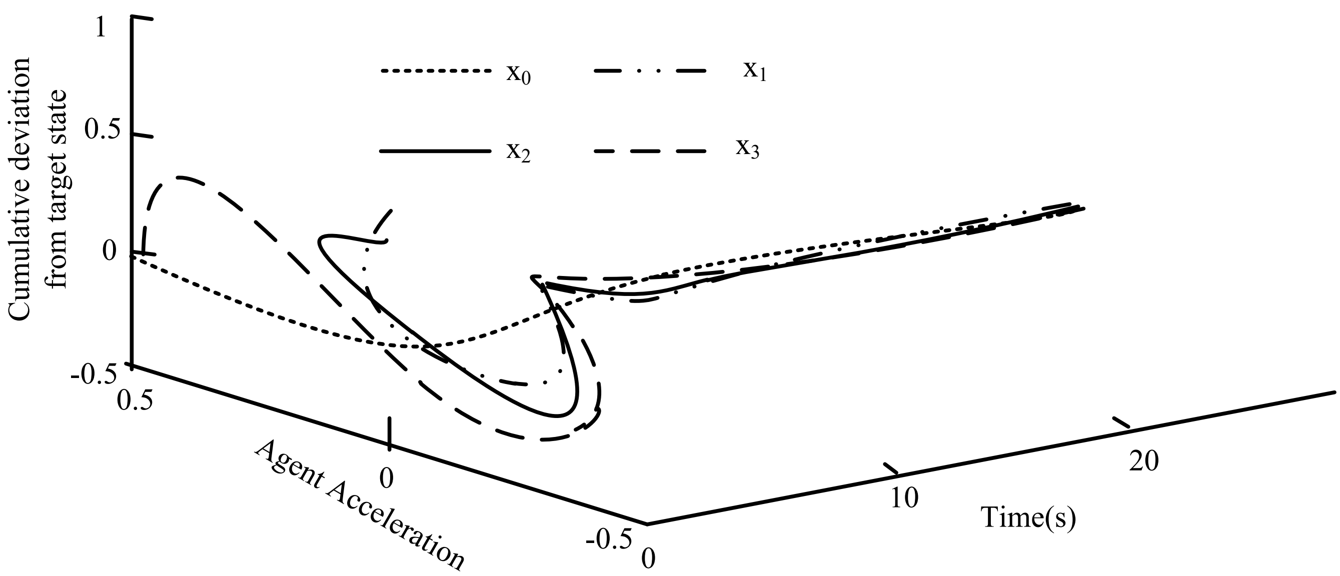


Figure 4 is a comprehensive evolution of the states of four individual agents, showing the overall situation of the agent acceleration of all agents (including leaders and followers) and the cumulative deviation of the agent from the target state. As shown in Figure 4, it can be seen that at the initial moment of the agent acceleration, the x1-x0 states of agents with different numbers (including leaders and followers) have different starting values; at 3 seconds, the acceleration change rate of each agent may be large and different. x1 is -0.16 acceleration, x2 is -0.22 acceleration, x3 is -0.25 acceleration, x0 is at 0.19 acceleration, and when it finally reaches 8.2 seconds, the acceleration change rate tends to 0, ensuring the stability and coordination of the agent's movement. It reflects the initial differences in the states of the four individual agents. Then observe the cumulative deviation between the agent and the target state. In the initial stage, the distance between each agent and the target state is different. As time goes by, the agent keeps approaching the target state. At about 10 seconds, the cumulative deviation gradually decreases, and the desired collaborative state is achieved, indicating that all agents have successfully reached a state consensus. The effectiveness and stability of the method in this paper are demonstrated.

Figure 4. The evolution of the states of four individual agents



4.3 Comparison of multi-agent collaborative motion performance

In order to further verify the motion performance of each agent in the multi-agent system of this method, this paper selects three common traditional PID control methods, consensus algorithms, and distributed control methods for comparative analysis. Table 2 shows the performance comparison of the four methods in a dynamic warehousing environment. It can be seen that the multi-agent collaborative motion performance of the traditional PID control method is poor, and it cannot cope well with the complex dynamic adjustment when the formation shape changes. The number of path nodes is 26 nodes, the number of obstacle collisions is 16 times, the number of turning points is 12 times, and the calculation time is 64.5 seconds. In contrast, the consistency algorithm and the distributed control method have improved in some indicators, but there are still shortcomings. Although the consistency algorithm reduces the number of path nodes by 18 and the number of collisions with obstacles by 3 times, the calculation time of 59.2 and the number of path turning points of 11 are still relatively high. The distributed control method is further optimized in the number of path nodes (15) and the number of collisions with obstacles (2 times). This is because the distributed control method uses graph theory to describe the communication topology between multi-agents to complete the collaborative control of multi-agents, but it is also slightly lower than the traditional method. However, the proposed method showed significant advantages in all indicators. The number of path nodes was reduced to only 6, the number of path turning points was kept at a reasonable 5, and no obstacle collisions occurred, which greatly improved the safety and efficiency of the operation. The calculation time was also shortened to 49.6 seconds, which is better than all other methods. The proposed method performed best in the dynamic environment method, with higher efficiency and safety of moving group path planning, and is a better collaborative strategy.

Table 2 Performance comparison of four methods in dynamic storage environment

Method	Path Node	Collision frequency	Turning Points	Calculation time /s
Traditional PID control method	26	8	12	64.5
Consistency algorithm	18	3	11	59.2
Distributed control method	15	2	7	58.6
This method	6	0	5	49.6

5. Conclusion

This study employs a Lyapunov indirect method to establish a multi-agent system model based on partial differential equations (PDEs). Initially, the system's dynamic obstacle avoidance and real-time response capabilities in formation tracking were not provided. Subsequently, the stability of the system was determined by analyzing the eigenvalue distribution of the linear system state equation. By mapping the originally discrete multi-agent system into a continuous space using PDEs, the control accuracy and efficiency of the cooperative motion system were improved. Simulation results verified the effectiveness of the proposed method.

In the trajectory analysis of cooperative motion, the position, direction acceleration, and cumulative deviation of four individual agents relative to the target stabilized within 9.5 seconds, 7.8 seconds, 8.2 seconds, and 10 seconds, respectively. The motion states converged toward a common value of 0, achieving overall stability through unified cooperative changes. Moreover, no obstacle collisions occurred, and the computation time of 49.6 seconds was significantly shorter compared to three other methods. This demonstrates the method's high efficiency and quality in achieving cooperative tracking, formation, and obstacle avoidance behaviors among agents.

Funding

no

Conflict of Interests

The author(s)declare(s) that there is no conflict of interest regarding the publication of this paper.

References

[1] Amirkhani A, Barshooi A H. Consensus in multi-agent systems: a review[J]. Artificial Intelligence Review, 2022, 55(5): 3897-3935.

[2] Wang J, Hong Y, Wang J, et al. Cooperative and competitive multi-agent systems: From optimization to games[J]. IEEE/CAA Journal of Automatica Sinica, 2022, 9(5): 763-783.

[3] Bao G, Ma L, Yi X. Recent advances on cooperative control of heterogeneous multi-agent systems subject to constraints: A survey[J]. Systems Science & Control Engineering, 2022, 10(1): 539-551.

[4] Ren H, Wang Y, Liu M, et al. An optimal estimation framework of multi-agent systems with random transport protocol[J]. IEEE Transactions on Signal Processing, 2022, 70: 2548-2559.

[5] Liu Y, Liu J, He Z, et al. A survey of multi-agent systems on distributed formation control[J]. Unmanned Systems, 2024, 12(05): 913-926.

[6] Yu C, Velu A, Vinitsky E, et al. The surprising effectiveness of ppo in cooperative multi-agent games[J]. Advances in Neural Information Processing Systems, 2022, 35: 24611-24624.

[7] Fang Z, Jiang D, Huang J, et al. Autonomous underwater vehicle formation control and obstacle avoidance using multi-agent generative adversarial imitation learning[J]. Ocean Engineering, 2022, 262: 112182.

[8] Li J, Shi H, Guo Y, et al. Tragcan: Trajectory prediction of heterogeneous traffic agents in iov systems[J]. IEEE Internet of Things Journal, 2022, 10(8): 7100-7113.