Proceedings of CONF-SEML 2026 Symposium: Computational Analysis and Modeling in Complex Intelligent Systems
DOI: 10.54254/2755-2721/2026.GU32392

A Survey of Multi-Agent Systems for Cooperative Control

Xuehan Zhang

Ocean University of China, Qingdao, China
1580392247@qq.com

Abstract. This paper reviews the latest research progress in multi-agent systems (MAS) for distributed control. We investigate four types of multi-agent systems: continuous-time MAS, discrete-time MAS, linear MAS, and nonlinear MAS. We also summarize the main results about consensus problem, event-triggered control problem, and distributed optimization. Based on the existing research, we also give some research directions that may need to be investigated such as machine learning, edge computing, and adaptive control. Simulations and industrial examples from robotics and manufacturing demonstrate the feasibility and effectiveness of these methods.

Keywords: Multi-agent systems, consensus, distributed control, event-triggered control

1. Introduction

Multi-agent systems (MAS) are a way to use computers and control systems. In these systems, many independent things like robots, vehicles, or software agents work together. They use local rules and communication to reach shared goals together [1]. These systems come from distributed artificial intelligence and control theory. People have paid much attention to MAS and used them in many areas, such as autonomous transportation systems, smart grids, industrial automation, and cooperative robotics. One main research problem is to design distributed control rules. These rules must make sure the system is stable, works well, and is strong against uncertainties, even with problems like limited communication, changing environments, and possible attacks [2].

This survey shows recent progress in MAS in a clear way. It divides the research into two main parts: time (continuous-time or discrete-time) and system dynamics (linear or nonlinear). We look at important methods, such as event-triggered control, scale-free protocol design, Lyapunov-based stability analysis, and intelligent approximation techniques. By using both theory and real examples, this review can help researchers and engineers who work on new autonomous systems.

2. Preliminaries

Let $\mathrm{G} = (\mathscr{V}, \mathscr{E}, \mathscr{W})$ be a weighted undirected graph with node set $\mathscr{V} = \{1, \ldots, \mathrm{N}\}$, edge set ϵ and symmetric weight matrix $\mathrm{W} = [\mathrm{w_{ij}}] \in \mathbb{R}^{\mathrm{N}\times\mathrm{N}}$ where $\mathrm{w_{ij}} > 0$ if $(\mathrm{i}, \mathrm{j}) \in \mathscr{E}$ and $\mathrm{w_{ij}} = 0$ otherwise. The adjacency matrix is $\mathrm{A} = \mathrm{W}$ and the corresponding Laplacian is

$$L = D - A \tag{1}$$

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.GU32392

where $D = \mathrm{diag}(d_1, \ldots, d_N)$ is the degree matrix with $d_i = \sum_j w_{ij}$.

3. Multi-agent system

3.1. Continuous-time multi-agent systems

Continuous-time multi-agent systems (MAS) have been extensively studied due to their natural alignment with physical systems governed by differential equations, such as robotic networks, autonomous vehicles, and power systems. The design and analysis of continuous-time controllers focus on achieving collective behaviors such as consensus, formation, and optimization through local interactions and distributed protocols.

Continuous-time multi-agent systems are often described by a differential equation:

$$\dot{x}_i(t) = Ax_i(t) + Bu_i(t) \tag{2}$$

Here, $x_i(t)$ is the state of agent i, $u_i(t)$ is its control input. The matrices A and B define the internal dynamics of each agent.

(Continuous-time consensus) If the graph g is undirected and connected, and the solution of (2) satisfies

$$\lim_{t\to\infty} \left\| x_i\left(t\right) - x_j\left(t\right) \right\| = 0 \tag{3}$$

Then system (2) can reach consensus.

3.1.1. Resilient consensus under adversarial attacks

Some agents in a network may be malicious. Bai and Wang [3] designed a protocol to achieve consensus despite these attacks. Their method simplifies the system by transforming the state, $x_i(t)$, into a new coordinate system, $\zeta_i(t) = Px_i(t)$. It then uses a scalar value, $\omega_i\left(t\right) = \widehat{\Gamma}\zeta_i\left(t\right)$, for decision-making.

The control law is:

$$u_i\left(t\right) = \Omega x_i\left(t\right) + c\sum_{j\in N_i\left(t\right)\backslash \bar{N}_i\left(t\right)} a_{ij}\left(t\right) YP\left(x_j\left(t\right) - x_i\left(t\right)\right) \tag{4}$$

In this law, Ω is a feedback gain. The constant c controls the interaction strength. The set $R_i(t)$ contains neighbors whose information is ignored because their values are extreme. This is known as the MSR strategy. Consensus is guaranteed if the network's communication graph is sufficiently well-connected, a property called $(2f+1)$ -robustness.

3.1.2. Event-triggered and scaled consensus

Constant communication uses too many resources. Lu and Jia [4] made an event-triggered method to cut down communication. Agents do not send information all the time. They only send new information when a local error becomes too big.

The control input is shown as:

DOI: 10.54254/2755-2721/2026.GU32392

$$u_i\left(t\right) = c\sum_{j\in N_i(t)} a_{ij}\left(y_j\left(t^j\right) - y_i\left(t^i\right)\right) \tag{5}$$

Agents send a hidden state, $y_i(t)$, to keep their information private [5]. The trigger condition is:

$$\left|\left|e_i\left(t\right)\right|\right| = \lambda\left|\left|\sum_{j=1}^{N} l_{ij}y_j\left(t_k^i\right)\right|\right| \tag{6}$$

Here, $e_i(t)$ is the measurement error. λ_i is a threshold value set by designers. t_H^i is the latest time when agent i sent an update. This method cuts communication a lot and stops Zeno behavior, so triggers do not happen infinitely fast.

3.1.3. Distributed optimization under edge agreements

Lu and Mou [6] expanded consensus to solve optimization problems. The goal is to make the total cost of all agents as small as possible, while following local rules between neighbors.

The problem is:

$$min\sum_{i=1}^{N} f_i(x_i)\ \ s.t.\ x_i = x_j, \forall(i,j)\in\mathscr{E} \tag{7}$$

Here, $f_i(x_i)$ is the local cost for agent i . The constraint $x_i = x_j$ means connected agents must agree with each other. Their algorithm uses saddle-point dynamics to find the best solution. This is helpful for jobs like formation control, where agents must keep correct positions relative to each other.

3.1.4. Key techniques and applications

Studies in this area use several important tools. Lyapunov theory is used to prove the system is stable. Graph robustness shows how well the network can resist attacks. Event-triggered control saves communication resources.

These methods are used in real systems. Examples include controlling aircraft flying in groups, controlling robot teams, and managing microgrids. They work well with less communication.

3.2. Discrete-time multi-agent systems

Discrete-time multi-agent systems work very well on digital systems. Control actions happen at fixed, separate times. These systems are easier to use on modern digital platforms for networked robots and automation. Research tries to solve important problems. These problems include synchronization with input limits, communication delays, and real industrial use.

3.2.1. Handling unknown input delays in discrete-time MAS

Liu et al. [7] studied systems with unknown and different input delays. The agent dynamics with input delay are shown as:

$$x_i(k+1) = Ax_i(k) + Bu_i(k-\tau_i) \tag{8}$$

where $x_i(t)$ is the state vector of agent at time step k , $u_i(k-\tau_i)\in R^m$ is the control input vector subject to an unknown delay τ_i , and matrices A and B show the agent's own dynamics.

DOI: 10.54254/2755-2721/2026.GU32392

The study found a delay limit that only depends on agent dynamics. The authors proposed a scale-free protocol that uses local information exchange. The protocol is written as:

$$\hat{z}_i\left(k\right) = \sum_{j=1}^{N} a_{ij}\left(\xi_i\left(k\right) - \xi_j\left(k\right)\right) \tag{9}$$

where $\hat{z}_i(k)$ is the local information exchanged, α_{ij} represents the communication weights, and $\xi_j(k)$ is an internal signal generated by agent j. This method makes the system strong without needing to know the network structure in advance [7].

3.2.2. Global synchronization under input saturation constraints

Liu et al. [7] solved global regulated state synchronization with input saturation. The agent model with saturation is:

$$x_i(k+1) = Ax_i(k) + B\sigma(u_i(k)) \tag{10}$$

$$y_i(k) = Cx_i(k) \tag{11}$$

Here, $\sigma(\cdot)$ is a standard saturation function, and $y_i(k) \in R^p$ is the output vector. The control goal is regulated state synchronization, which means:

$$\lim_{k\to\infty}\left(x_i\left(k\right) - x_r\left(k\right)\right) = 0 \tag{12}$$

where $x_y(k)$ is the reference path from an exosystem. The authors made a nonlinear protocol with a dynamic scheduling parameter ϵ:

$$\varepsilon\left(\chi_i\left(k\right)\right) = \max\{\rho \in \left(0, \rho^*\right] : 2\chi_T\left(k\right)P_\rho\chi_i\left(k\right)tr\left(P_\rho\right)\|B\|^2 \le 1\} \tag{13}$$

In this formulation, $\chi_i(k)$ is an internal protocol state, and p_ρ is the only positive-definite solution to a discrete-time algebraic Riccati equation. This method works well with actuator saturation limits [7].

3.2.3. Industrial applications in job shop scheduling

Sanogo et al. [8] used these systems in real job shop scheduling. The main problem is to order jobs on machines in the best way. Autonomous Intelligent Vehicles (AIVs) carry materials. The goal is to minimize makespan, which is:

$$\min C_{max} = \max\{C_i\} \tag{14}$$

where $c_{\max}$ is the total makespan and c_i is the finish time of i.

Human worker behavior in the workshop is modeled using a Discrete-Time Markov Chain (DTMC). The transition probability matrix p is:

DOI: 10.54254/2755-2721/2026.GU32392

$$P = \begin{pmatrix} p_{1,1} & \cdots & p_{1,\lambda} \\ \vdots & & \vdots \\ p_{\lambda,1} & \cdots & p_{\lambda,\lambda} \end{pmatrix}, \sum_{y=1}^{\lambda} p_{x,y} = 1 \tag{15}$$

where $p_{x,y}$ is the probability that a worker moves from place x to place y. Path planning for AIVs considers human workers and optimizes this cost:

$$Minimize\, TT + (ENH \times AVG_w) \tag{16}$$

TT is travel time, ENH is the estimated number of humans on the path, and AVG_{wt} is the average time cost of avoiding a human. Tests show that AIVs reduce makespan more than traditional AGVs. This shows the method works well [8].

3.2.4. Conclusion and future trajectory

Research on discrete-time multi-agent systems has made big progress in making strong control methods. Scale-free protocols can be used for large systems without needing special network information. Good ways to handle input saturation and communication delays make these systems more useful in real life. Successful industrial uses connect theoretical design with real needs of cyber-physical systems.

3.3. Linear multi-agent systems

Linear multi-agent systems are very important in cooperative control. Their models are easy to study. They are also used in many areas [7]. The main goal is consensus or synchronization. Agents reach the same state or output. They do this by communicating with nearby agents. Research has developed a lot. Early work only studied simple consensus algorithms. Now, methods are more advanced. They can solve problems such as digital use and limited resources.

3.3.1. Continuous-time linear mass and event-triggered control

A common continuous-time linear agent model is:

$$\dot{x}_i(t) = Ax_i(t) + Bu_i(t), \quad y_i(t) = Cx_i(t) \tag{17}$$

Here, $x_i(t)$ is the state vector of agent i. It shows the agent's internal condition. $\mathrm{u_i(t)}$ is the control input vector. This is the command given to the agent. $\mathrm{y_i(t)}$ is the output vector. This is the measurable signal from the agent. The matrices A, B, and C define the agent's dynamics.

Early studies assumed continuous communication. This is not practical. It uses too much communication bandwidth. Event-Triggered Control (ETC) is a better solution. It reduces communication. For example, Yang et al. [9] studied bipartite consensus. This works with signed graphs. Signed graphs can show friendly and hostile interactions. They proposed a distributed ETC protocol.

$$u_i\left(t\right) = K\sum_{j=1}^{N} a_{ij}\left(x_j\left(t_k^i\right) - sgn\left(a_{ij}\right)x_i\left(t_k^i\right)\right) \tag{18}$$

DOI: 10.54254/2755-2721/2026.GU32392

Here, $\mathscr{L}_{\mathrm{H}}^{\mathrm{i}}$ is the last event time for agent i. The control signal updates only at these times. It uses sampled state data from neighbors. A special triggering condition is used. This condition relies on local errors. It ensures bipartite consensus is achieved. Agents converge to opposite values across groups. Controller updates are reduced. Stability is proven with Lyapunov theory. The method also avoids Zeno behavior [9].

3.3.2. Discrete-time systems and scale-free design

Discrete-time models are crucial for digital systems. They are implemented on computers. The model is similar to the continuous one.

$$x_i(t+1) = Ax_i(t) + Bu_i(t) \tag{19}$$

$$y_i(t) = Cx_i(t) \tag{20}$$

The variables mean the same. But time t is now a discrete step.

A major challenge is scale-free design. Protocols must work without network information. They should not need to know the number of agents. Liu et al. [7] solved this for heterogeneous systems. Their systems had unknown, variable communication delays. Their method has two steps. First, a pre-compensator changes each agent. It makes different agents act the same. Second, a collaborative protocol is designed. This protocol only needs the model of the transformed agents. It does not need network details. This is a scale-free design. It guarantees output synchronization. It works for any connected network. It handles any communication delays. This property is particularly beneficial for large-scale real-world applications [7].

3.3.3. Output feedback and finite-time control

Sometimes, the full state $\mathrm{x_i(t)}$ cannot be measured. Only the output $\mathrm{y_i(t)}$ is available. Output feedback control is needed. An observer estimates the missing state. Li et al. [10] designed a finite-time observer.

$$\dot{\hat{x}}_i\left(t\right) = A\hat{x}_i\left(t\right) + Bu_i\left(t\right) - g_{FT}\left(\sigma_i\left(t\right)\right) \tag{21}$$

Here, $\mathrm{\hat{x}_i(t)}$ is the estimated state. $\mathrm{\sigma_i(t) = y_i(t) - C\hat{x}_i(t)}$ is the output error. The function $\mathrm{g_{FT}(\bullet)}$ is special. It makes the estimation error go to zero in finite time. This is faster than older, asymptotic observers.

This observer was combined with Dynamic Event-Triggered Control (DETC). The goal was finite-time consensus tracking. The control law has a special term. This term uses a fractional power. It forces the system to agree in finite time. The triggering rule uses an internal variable $\eta_i(t)$. This variable changes the threshold dynamically. This method greatly reduces triggers. It also prevents Zeno behavior [10].

3.3.4. Summary of linear multi-agent control

Research on linear MASs has advanced from basic consensus algorithms to sophisticated strategies handling discrete-time dynamics, scale-free design, output feedback, and finite-time convergence. The integration of ETC, especially DETC, with advanced observers enables resource-efficient, high-

DOI: 10.54254/2755-2721/2026.GU32392

performance control [7-8], forming a robust foundation for practical applications with limited resources and demanding performance requirements [7].

3.4. Nonlinear multi-agent systems

3.4.1. Fuzzy approximation of unknown nonlinearities

Research on nonlinear multi-agent systems (MAS) has advanced toward addressing unknown dynamics, input constraints, and efficient communication. Early work focused on approximating nonlinear functions using intelligent models. Li et al. [11] studied singular MAS with dynamics

$$Ex_i(t) = Ax_i(t) + f(x_i) + Bu_i(t) \tag{22}$$

where the unknown nonlinear term $f(x_i)$ is approximated by a fuzzy logic system (FLS). The control law is

$$u_i = F\sum_{j=1}^{N} \omega_i a_{ij}\left(x_i - x_j\right) \tag{23}$$

An adaptive rule tunes the coupling gains ω_j . Their LMI-based analysis ensured stability and impulse-free behavior and demonstrated that all agents reach consensus.

Practical nonlinearities such as dead zones were later integrated into MAS control design. Jiang and Sheng [12] modeled the dead zone as

$$\Psi_i(u_i) = m_i u_i + d_i(t) \tag{24}$$

where m_i is a linear coefficient and $d_i(t)$ is a bounded disturbance. They designed an adaptive fuzzy event-triggered controller with input

$$u_i(t) = \omega_i(t_k), \quad t \in [t_k,\ t_{k+1}) \tag{25}$$

and the triggering condition

$$t_{k+1} = inf\{t > t_k \mid ||Z_i(t) \geq hu_i(t) + \eta\} \tag{26}$$

This strategy achieved leader–follower synchronization, avoided Zeno behavior, and preserved system stability.

To reduce resource usage, event-triggered control (ETC) has been widely applied. ETC updates signals only when required, lowering communication loads while maintaining performance. Using Lyapunov methods, Jiang and Sheng [12] showed that all closed-loop signals remain bounded and tracking errors converge even with dead zones and unknown nonlinearities.

Parallel progress was made in navigation tasks for multi-robot systems. Rousseas et al. [13] introduced an Artificial Harmonic Potential Field (AHPF) for safe exploration. The potential field ϕ is derived from

$$\nabla^2\phi\left(p_i; k_i\right) = 0 \tag{27}$$

DOI: 10.54254/2755-2721/2026.GU32392

A low-level tracking controller controlled the nonlinear dynamics. The heading angle followed this rule:

$$\theta = -K_\theta(1 - cos(\alpha))sin(\alpha) \tag{28}$$

This method avoided local minima. It also allowed full exploration when sensing was limited.

These nonlinear MAS methods have been tested with simulations and SITL experiments. Li et al. [11] and Jiang & Sheng [12] showed reliable consensus and tracking when there were nonlinearities and constraints. Rousseas et al. [13] showed strong exploration and map-building. All these studies together prove that modern nonlinear MAS control is practical and well-developed.

4. Future study for MAS

The future of multi-agent systems (MAS) will depend on how well they deal with uncertainty, work well in large networks, and stay safe in bad environments. A growing trend is to combine learning with distributed control. Learning methods let agents adapt to changes in real time, but they can also make the system unstable. Online updates may break consensus and create non-stationary behavior, especially when many agents learn at the same time [4]. Future research needs to put learning methods together with stable control frameworks so that agents can adapt and still keep reliable system behavior.

Scalability will still be a main challenge. Recent progress in scale-free and topology-independent protocol design has reduced the need for global network information [7]. However, it is still hard to check global performance using only local interactions. As networks become larger, delays, link failures, and heterogeneous dynamics make traditional analysis tools not enough. New hierarchical or modular approaches may be needed to analyze and guarantee performance in very large MAS such as swarm robotic systems and distributed sensor networks.

Security and resilience are also becoming very important. MAS used in open environments must resist malicious agents, false information, and privacy risks. Resilient algorithms provide protection, but they often need more communication or become less efficient [3]. Designing controllers that work well under bad conditions while keeping good performance will need new methods that combine trust evaluation, secure consensus, and privacy-preserving optimization.

The integration of MAS with edge computing and 5G/6G networks makes things more complex. High-speed communication allows real-time coordination, but asynchronous updates and changing network loads can cause instability. Making sure the system behaves well under these conditions needs joint communication–control design, where scheduling, bandwidth allocation, and control rules work together to support distributed decision making [8].

Another big problem is the lack of standardized benchmarks. Many existing studies use custom simulation environments, so it is hard to compare methods or check results. Without shared testing platforms, progress is not well connected. Establishing unified benchmarks will help reproduce results and speed up the transition from theoretical research to real-world applications [11-13].

In summary, future MAS research must solve several basic problems: keeping stability in learning-based systems, making sure scalability works under uncertain network conditions, improving resilience against bad agents, coordinating communication with control, and developing standardized evaluation tools. Progress in these areas will let MAS work safely and efficiently in complex cyber-physical environments.

DOI: 10.54254/2755-2721/2026.GU32392

5. Conclusion

This survey has looked at the best and newest control methods in multi-agent systems. It shows clear progress in both theory and real-world use. Continuous-time models now have strong coordination and efficient event-triggered work. Discrete-time systems support digital control that works on a large scale and can handle delays. Linear MAS still provide easy-to-use tools for consensus and synchronization. Nonlinear methods, such as fuzzy logic, adaptive control, and potential-field methods, deal with complex dynamics and real-world limits. All these developments share common goals: better use of resources, robustness under uncertainty, and scalability. These goals are supported by progress in triggering mechanisms, adaptive learning, and graph-theoretic design.

References

[1] Luzolo, P. H., Elrawashdeh, Z., Tchappi, I., Galland, S., & Outay, F. (2024). Combining Multi-Agent Systems and Artificial Intelligence of Things: Technical challenges and gains. Internet of Things, 28, 101364. https: //doi.org/10.1016/j.iot.2024.101364

[2] Han, Y., Zhang, K., Li, H., Coelho, E. a. A., & Guerrero, J. M. (2017). MAS-Based Distributed Coordinated Control and Optimization in Microgrid and Microgrid Clusters: A Comprehensive Overview. IEEE Transactions on Power Electronics, 33(8), 6488–6508. https: //doi.org/10.1109/tpel.2017.2761438

[3] Bai, Y., & Wang, J. (2022). Resilient consensus of Continuous-Time Linear networked systems. IEEE Transactions on Circuits & Systems II Express Briefs, 69(8), 3500–3504. https: //doi.org/10.1109/tcsii.2022.3161369

[4] Lu, X., & Jia, Y. (2022). Scaled Event-Triggered resilient consensus control of Continuous-Time Multi-Agent systems under Byzantine agents. IEEE Transactions on Network Science and Engineering, 10(2), 1157–1174. https: //doi.org/10.1109/tnse.2022.3229820

[5] Wang, A., Liu, Y., & Huang, T. (2022). Event-Triggered privacy-preserving average consensus for continuous-time multi-agent network systems. Journal of the Franklin Institute, 359(10), 4959–4975. https: //doi.org/10.1016/j.jfranklin.2022.04.028

[6] Lu, Z., & Mou, S. (2023). Distributed optimization under edge agreements: A continuous-time algorithm. Systems & Control Letters, 183, 105698. https: //doi.org/10.1016/j.sysconle.2023.105698

[7] Liu, Z., Nojavanzadeh, D., Saberi, A., & Stoorvogel, A. A. (2022). Scale-Free collaborative protocol design for output synchronization of heterogeneous Multi-Agent systems with nonuniform communication delays. IEEE Transactions on Network Science and Engineering, 9(4), 2882–2894. https: //doi.org/10.1109/tnse.2022.3172567

[8] Sanogo, K., Benhafssa, A. M., & Sahnoun, M. (2024). A multi-agent system simulation of job shop scheduling with human consideration: A comparative analysis of AGVs and AIVs. Simulation Modelling Practice and Theory, 139, 103060. https: //doi.org/10.1016/j.simpat.2024.103060

[9] Yang, R., Peng, L., Yang, Y., & Zhu, F. (2021). Bipartite consensus of Linear Multi-Agent Systems by Distributed Event-Triggered Control. Journal of Systems Science and Complexity, 34(3), 955–974. https: //doi.org/10.1007/s11424-020-9293-7

[10] Li, Z., Hu, Y., Ning, X., & Chen, Y. (2025). Finite-Time Output Consensus Tracking for General Linear Multi-Agent Systems via Dynamic Event-Triggered Mechanism. Systems Science and Mathematics, 1-24. https: //link.cnki.net/urlid/11.2019.o1.20250708.1516.050.

[11] Li, J., Zhang, Y., & Jin, Z. (2023). Distributed cooperative control of singular multi-agent systems based on fuzzy logic approach. International Journal of Fuzzy Systems, 26(1), 390–401. https: //doi.org/10.1007/s40815-023-01607-w

[12] Jiang, T., & Sheng, N. (2024). Event Triggered Cooperative Control of Multi-agent System with Input Dead Zone. Complex Systems and Complexity Science, 21(4), 58-64. https: //doi.org/10.13306/j.1672-3813.2024.04.010.

[13] Rousseas, P., Karras, G. C., Bechlioulis, C. P., & Kyriakopoulos, K. J. (2022). Indoor Visual Exploration with Multi-Rotor Aerial Robotic Vehicles. Sensors, 22(14), 5194. https: //doi.org/10.3390/s22145194