Proceedings of the 8th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/2026.36380

Decentralized Coordination Architectures for Intelligent Agent Swarms

Tiancheng Ge

Department of Automation, North University of China, Taiyuan, China
3321759641@qq.com

Abstract. Large swarms of intelligent agents are useful only when coordination still works after the tidy assumptions of a laboratory test disappear. That is the practical problem behind decentralized coordination architectures. The review follows the field from graph-based modeling and consensus theory to the less tidy questions that appear in deployment: delayed messages, changing topology, faulty agents, and limited computation. The discussion treats distributed consensus, event-triggered communication, resilient control, fault-tolerant design, and cognition-inspired adaptation as parts of one architecture problem. The literature from 2020 to 2025 suggests a clear pattern. Stabilizing high-order nonlinear swarms is still difficult; robust performance under intermittent connectivity remains fragile; and adaptive decision-making is not yet fully reliable when the environment keeps changing. A combined route is needed, one in which cognitive intelligence, edge-side computing, and verification tools are developed together. Such a route is especially relevant to intelligent manufacturing, emergency response, autonomous transportation, and other settings where a centralized command chain may be too slow or too vulnerable.

Keywords: Decentralized coordination architectures, Multi-agent systems, Event-triggered control, Resilient control, Cognitive intelligence

1. Introduction

1.1. Technological context and strategic imperatives

It is easy to describe an intelligent swarm in a simulation. It is much harder to keep it useful when a link fails, a node drops out, or the task changes mid-operation. In that setting, scalability, fault tolerance, and response time are not decorative performance indicators. They decide whether the architecture can be trusted at all.

The pressure is visible in three familiar scenarios. After a disaster, UAV formations may fly through smoke, damaged buildings, GPS-denied areas, and contested radio channels. If the whole mission waits for a single command center, the formation becomes brittle; local coordination gives it a chance to continue even when part of the network is lost [1]. Urban vehicle-road collaboration creates a different bottleneck. Roadside sensors and vehicles produce data faster than a distant platform can always process, so perception and collision avoidance have to move closer to the agents [2]. Smart factories raise the same issue in another form: collaborative robots must absorb

DOI: 10.54254/2755-2721/2026.36380

line changes, local faults, and human intervention without stopping production. These cases make one point hard to avoid. Decentralization is not just an elegant control idea; in many deployments it is the condition for continued operation [3].

A centralized controller gives a clean point of coordination, but it also gives the system a clean point of failure. Decentralized designs try to build global order from local exchanges instead [4]. That trade is attractive, yet it is not free. Once agents are heterogeneous, channels are unreliable, and the environment is partly adversarial, the usual theoretical picture begins to look too polished. Figure 1 places this shift in the broader evolution of swarm coordination.

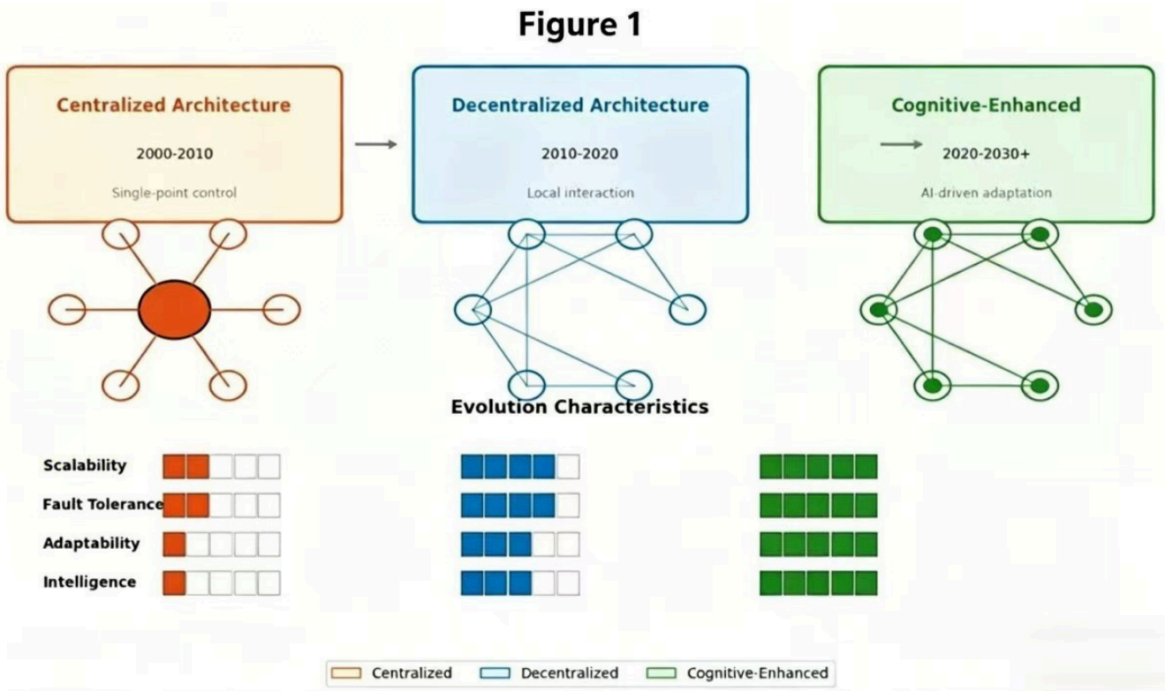

Figure 1. Swarm coordination architecture evolution and performance

1.2. Global research landscape and survey objectives

Recent work does not move in a single neat line. One stream has made intermittent connectivity, sampling constraints, and limited communication more systematic, with Ge et al.'s 2025 review offering a useful marker [5]. Another stream, including work by Chinese scholars, has pushed predictive control and fully actuated system methods [6]. A third connects coordination with resilient control, distributed learning, and privacy-aware computation [7, 8]. What ties these strands together is a difficult question: how much coordination can still be proved when information, computation, and trust are all incomplete?

A weakness in many surveys is their narrowness. A paper may cover consensus in great detail, or event-triggered control in great detail, while saying less about why the same method becomes awkward in a physical swarm. This review uses a wider lens. It asks what the available theory can actually guarantee, where deployment breaks those guarantees, and which integrations may make the gap smaller. The literature window of 2020-2025 is chosen because it captures recent work on learning-enhanced control, security-aware design, and human-agent collaboration [9].

DOI: 10.54254/2755-2721/2026.36380

2. Theoretical foundations of decentralized coordination architectures

Figure 2 gives a compact map of the methods treated here as the current backbone of decentralized coordination.

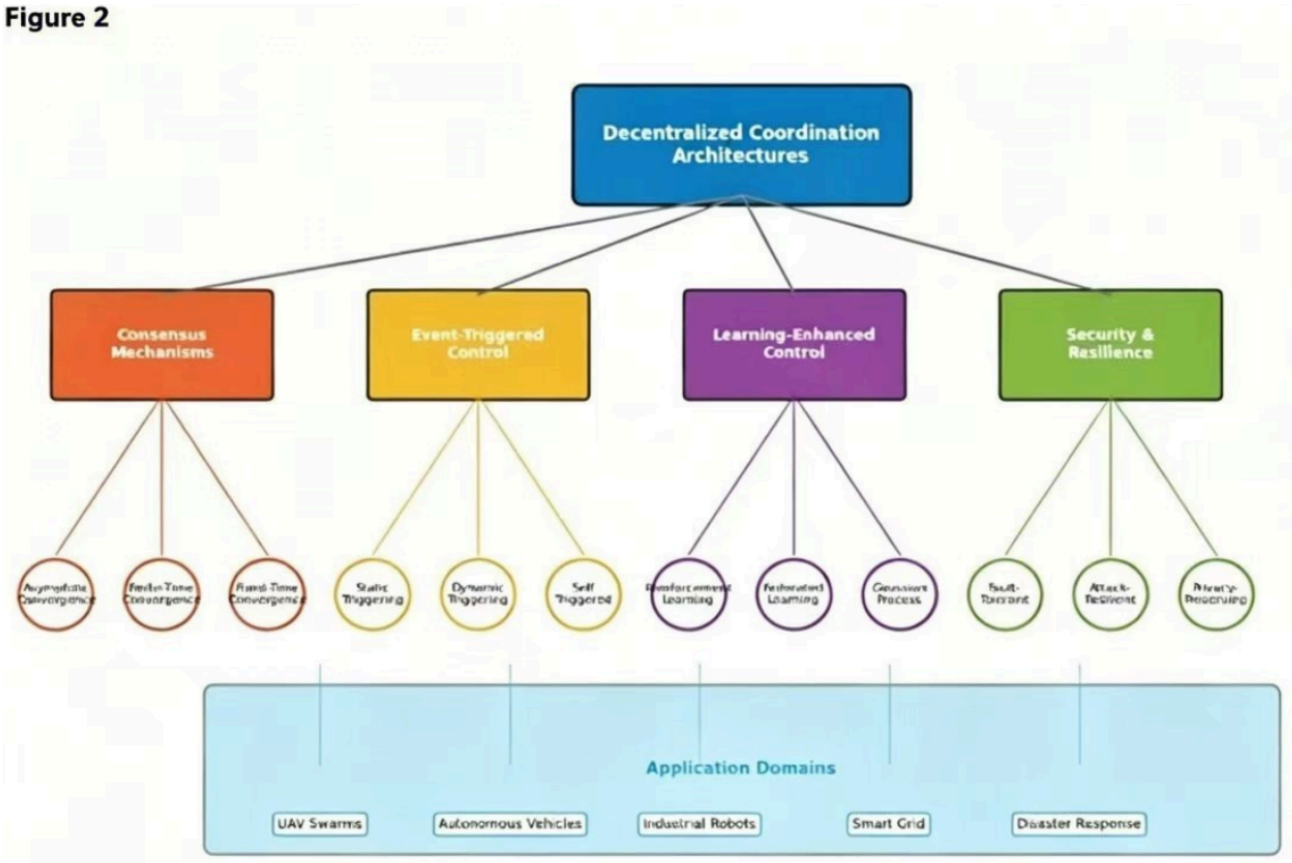

Figure 2. Core decentralized coordination architecture paradigms

2.1. Mathematical underpinnings of system modeling

Most coordination analysis begins with a useful simplification: a swarm can be read as a graph. Agents become vertices, communication links become edges, and the Laplacian spectrum tells us something about how quickly information can move through the network. When the topology changes with time, joint connectivity assumptions provide the minimal structure needed for convergence arguments [10]. The attraction of this view is obvious. It turns a messy collective behavior problem into something that can be studied with graph structure and system dynamics.

The simplification has limits. Recent models have moved beyond single- and double-integrator agents toward fully actuated systems, Euler-Lagrange dynamics, and nonholonomic platforms [3]. That move is necessary because real robots and vehicles are not point masses. Still, the analysis often keeps assumptions that field deployment will not respect: clean communication, known parameters, and topology changes that stay within a manageable class. Add packet loss, actuator uncertainty, and changing payloads, and the modeling error stops being a small nuisance. It can become part of the instability mechanism.

DOI: 10.54254/2755-2721/2026.36380

2.2. Core methodological paradigms

2.2.1. Consensus mechanisms: from asymptotic to finite-time convergence

Consensus is still the basic grammar of the field, but the question has changed. It is no longer enough to ask whether agents eventually agree. In many tasks, they must agree fast, under high-order dynamics, and with fewer messages [11]. Predictive control uses a moving horizon to handle delay. Impulsive and event-triggered designs try to save communication while keeping convergence from becoming too slow [12]. The research direction is sensible; the hard part is making these gains survive outside the assumptions used in the proof.

Here the literature is less settled than it sometimes sounds. Consensus can be proved for many carefully structured nonlinear systems, but broad nonlinear classes remain awkward. Lipschitz conditions, strict-feedback forms, and similar assumptions make analysis tractable, yet they also define a fairly narrow world. Backstepping, sliding-mode control, and adaptive compensation can stretch that world, but the resulting controllers are often difficult to tune and scale. Figure 3 compares representative convergence patterns.

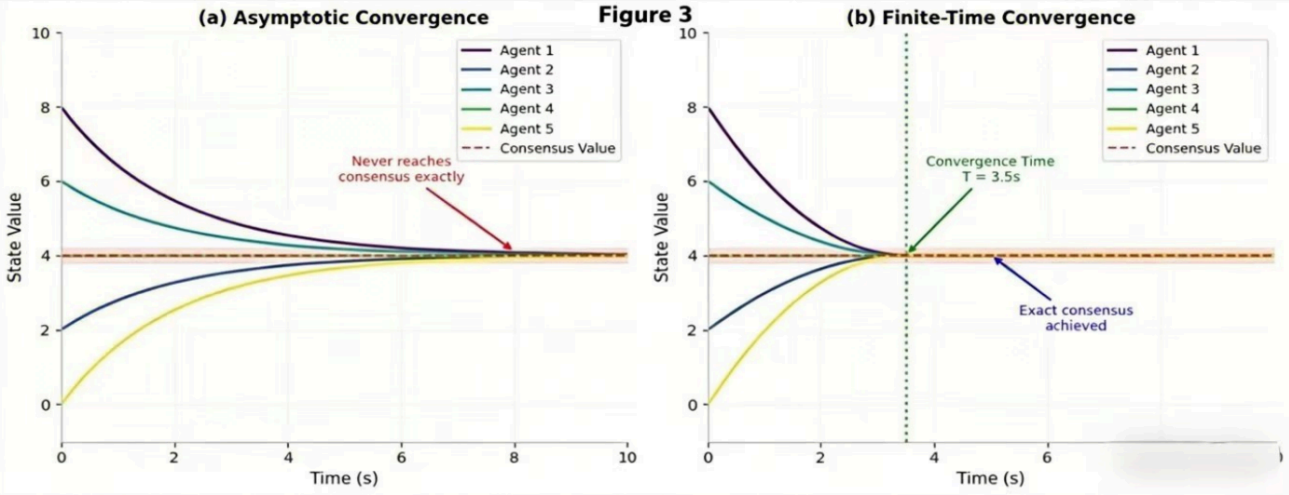

Figure 3

Figure 3. Consensus control convergence performance comparison (a) Asymptotic convergence; (b) Finite-time convergence

2.2.2. Event-triggered communication: balancing efficiency and stability

Periodic sampling is comfortable for analysis. For a large swarm, it can also be wasteful. Why should every agent transmit just because a clock says so, even when its state has barely changed? Event-triggered communication answers by tying messages to state deviation, thresholds, or prediction error. Adaptive thresholds adjust the trade-off online; dynamic triggering avoids Zeno behavior; self-triggered rules let an agent estimate its next useful transmission time without constant monitoring [5, 13]. Figure 4 shows the communication advantage of this design family.

DOI: 10.54254/2755-2721/2026.36380

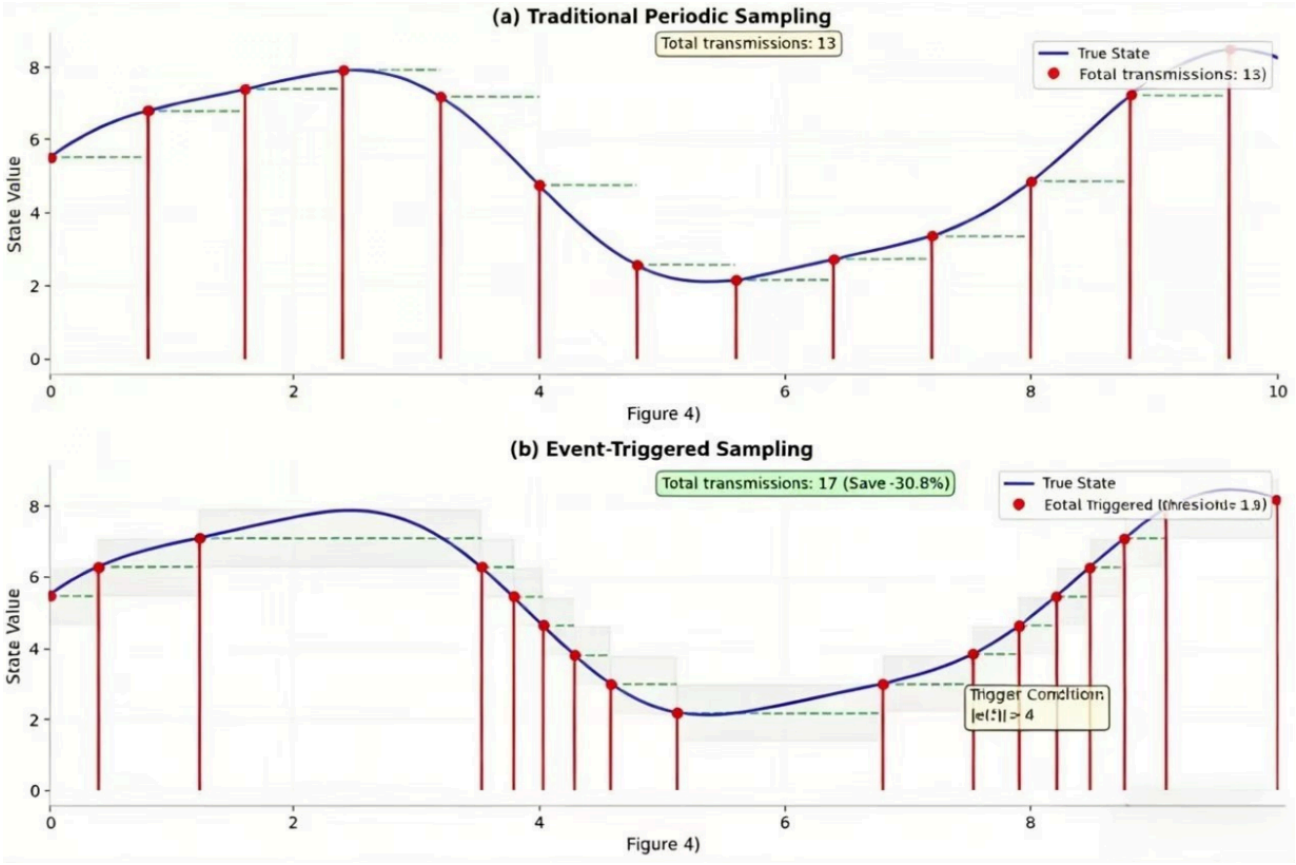

Figure 4. Periodic and event-triggered sampling comparison (a) Traditional periodic Sampling; (b) Event-triggered sampling

2.2.3. Learning-enhanced control: adaptation under uncertainty

Learning-based control is tempting because it promises adaptation when the model is incomplete. Value-function approximation and policy-gradient methods can update strategies online, and neural networks can represent nonlinear mappings that are hard to write by hand. Federated multi-agent reinforcement learning adds another benefit: agents can train together without exposing all raw data [7]. The catch is familiar but serious. Many guarantees still need stationarity, enough exploration, or stable reward structures. A fielded swarm may offer none of these for very long.

2.3. Security and reliability in decentralized coordination architectures

2.3.1. Fault-tolerant design: maintaining operation under component failures

Fault tolerance usually begins locally. A module detects an actuator or sensor anomaly, then some higher-level check prevents that local diagnosis from misleading the whole swarm. Neural adaptation can absorb part of the actuator uncertainty, and active-passive designs extend the range of faults that can be tolerated [3]. The open question is what happens when the fault is not nicely bounded, not statistically familiar, or not isolated from other disturbances. Much of the current theory still becomes cautious at that point.

DOI: 10.54254/2755-2721/2026.36380

2.3.2. Resilient control: withstanding malicious intrusions

Security is harder than ordinary robustness because the disturbance may watch and react. Predictive control and sliding-mode ideas can keep states bounded under some deception attacks, and observer reconstruction can help locate compromised sensors [13]. There is also growing work on Sybil and denial-of-service attacks in resilient consensus [8]. These results matter, but they often assume the attack model is already known. A more troubling case is an adaptive attacker that changes strategy after observing the controller. That case is not yet well closed.

2.4. Synthesis: theoretical achievements and their boundaries

So the theory is strong, but uneven. It gives useful tools by trimming the problem into analyzable shapes. Simplified dynamics hide physical constraints; asymptotic guarantees may be too slow for urgent tasks; continuous connectivity rarely matches real networks; and benign-environment assumptions make security look easier than it is. The engineering constraints in the next section are not side issues. They are where many elegant methods meet their first serious test.

3. Engineering constraints on decentralized coordination architectures

3.1. Fundamental theoretical bottlenecks

3.1.1. High-order nonlinear dynamics: beyond analytical tractability

High-order nonlinear swarms expose the gap quickly. Low-order linear models are manageable; real platforms often are not. When coupling is strong and actuators have limits, even writing a useful Lyapunov function can become difficult. Dissipativity and passivity arguments help in some cases, but they usually need structural properties that a mixed robot team may not have.

3.1.2. Intermittent connectivity: the stability–resource paradox

Continuous synchronous communication is a helpful fiction. Real networks give packet loss, clock drift, intermittent sampling, and latency that changes over time. This creates a practical dilemma: save bandwidth and energy, or transmit often enough to protect performance [5]. Too much communication exhausts resources; too little can break stability. Existing separation principles still do not fully capture this information-dynamics coupling. Figure 5 shows the kind of time-varying topology that makes the issue visible.

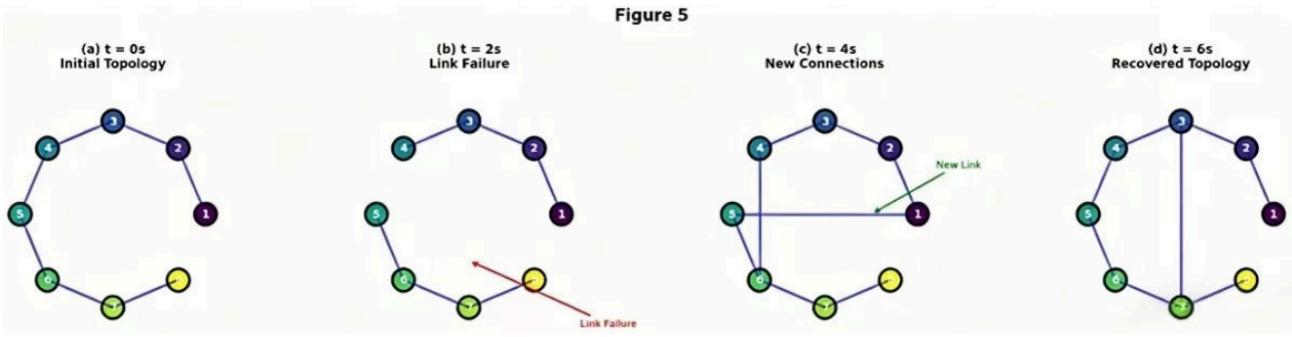

Figure 5. Time-varying communication topology evolution process

DOI: 10.54254/2755-2721/2026.36380

3.2. Structural obstacles to engineering deployment

3.2.1. Heterogeneous integration: the compositionality challenge

Heterogeneity makes deployment still less tidy. A ground vehicle, an aerial vehicle, and a manipulator may share a mission, but they do not share dynamics, sensors, or decision cycles [1]. A unified model can flatten away important differences. A hierarchy can hide coupling that becomes critical during contact, formation changes, or shared perception. In practice, designers often end up with monolithic solutions because compositional analysis tools are not mature enough.

3.2.2. Resource-constrained deployment: the edge computing imperative

Resource limits are not just an implementation detail. Quantization, noisy channels, and clock asynchrony accumulate across a swarm, and biased local estimates can distort the global decision. Moving computation to the edge helps with latency, but edge nodes also have finite memory, power, and processing capacity. A design that assumes resources will always be available is incomplete from the start. The harder question is graceful degradation: what should the swarm give up first when the edge layer is overloaded?

3.3. Cross-domain fusion challenges

3.3.1. Privacy–security tensions: intrinsic contradictions in open environments

Coordination needs information; privacy and security often restrict it. That conflict is especially sharp in transportation, industrial, and public-safety settings. Encryption may be too expensive for tight control loops. Differential privacy may blur the signal the controller needs. Federated learning avoids some direct data sharing, but it still faces slow convergence and model heterogeneity. The design problem is not simply to add a privacy tool. It is to decide what information must remain visible for coordination to remain possible.

3.3.2. Environmental adaptation: beyond stationarity assumptions

Another fragile assumption is that the world stays close to the design model. It rarely does. Goals change, obstacles move, payloads vary, and communication quality shifts during the mission. Online learning can update policies, but safe exploration in a large swarm is not a small issue. Nor is sample efficiency. A deployable architecture has to keep adapting without using the real mission as an uncontrolled experiment.

3.4. Synthesis: from barriers to transformative directions

These barriers should be read as normal operating conditions, not as afterthoughts. A swarm will face limited resources, partial information, uncertain environments, and sometimes hostile behavior. Figure 6 compares mainstream methods with that reality in mind. The design lesson is blunt: complexity and scarcity have to enter the architecture at the beginning.

DOI: 10.54254/2755-2721/2026.36380

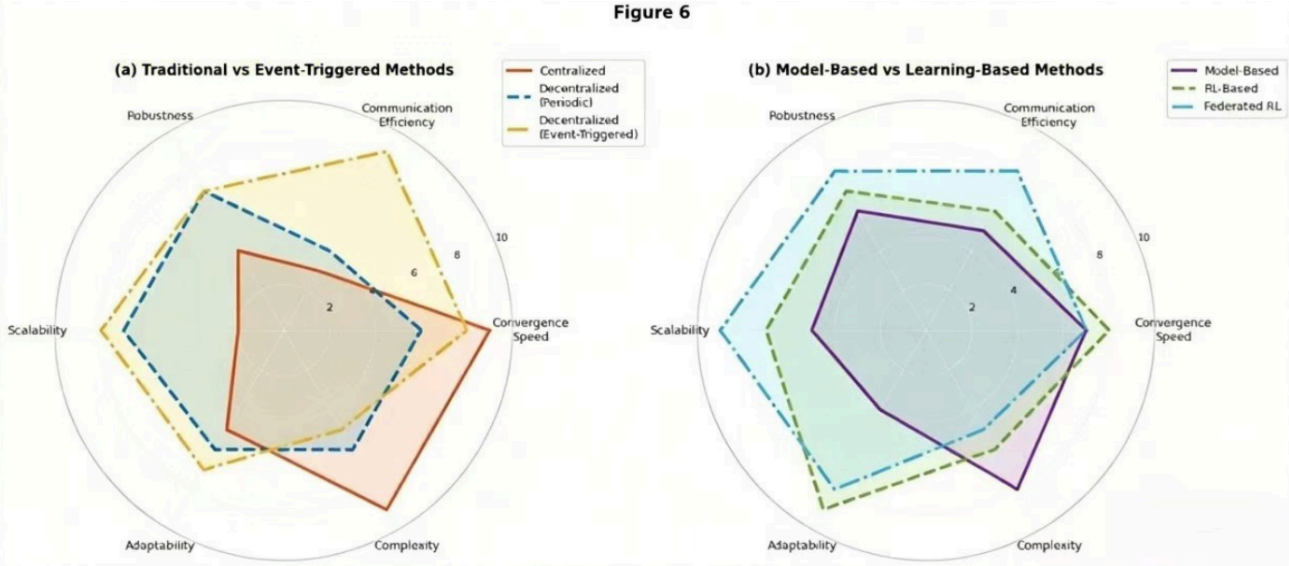

Figure 6. Mainstream decentralized coordination method comparison (a) Traditional vs event-triggered methods; (b) Model-based vs learning-based methods

4. Transformative pathways for next-generation decentralized coordination architectures

The previous sections lead to a practical conclusion. Better algorithms are needed, but algorithmic improvement alone will not carry decentralized coordination into demanding deployments. The architecture must also decide where computation happens, how uncertainty is checked, how humans remain in the loop, and how different domains communicate. The pathways below are written with that broader system problem in mind.

4.1. Cognitively enhanced control frameworks

4.1.1. Foundation model integration: from general intelligence to control guarantees

Foundation models are useful because they can read instructions, images, and context in ways that conventional controllers cannot. In a swarm, such models might translate a high-level mission description into task allocations or candidate coordination rules. But there is an uncomfortable boundary here. Fluent reasoning is not the same as control safety. Before these models can influence control decisions, their outputs need verification, physical grounding, and uncertainty estimates; otherwise they remain planning assistants rather than trustworthy control components.

4.1.2. Graph neural networks: learning relational structure for coordination

Graph neural networks match the shape of the problem better than many standard learning models. A swarm is relational: each agent acts through changing neighbors, not through a fixed grid. GNNs can learn interaction patterns from data, and reinforcement learning can use those patterns to reduce dependence on hand-designed communication structures [14]. The interesting part is dynamic graph learning. If the learned representation can change as the swarm splits, merges, or scales, the controller may become less tied to one assumed topology.

DOI: 10.54254/2755-2721/2026.36380

4.2. Architecturally innovative engineering systems

4.2.1. Cloud–edge–device hierarchies

A deployable swarm will probably not compute in one place. Cloud resources are useful for training and global optimization; edge nodes are better for low-latency inference and regional coordination; devices must still handle sensing and fast control. Split learning, federated distillation, and consensus-based distributed training are ways to divide this burden without centralizing all raw data. Figure 7 sketches the three-tier arrangement.

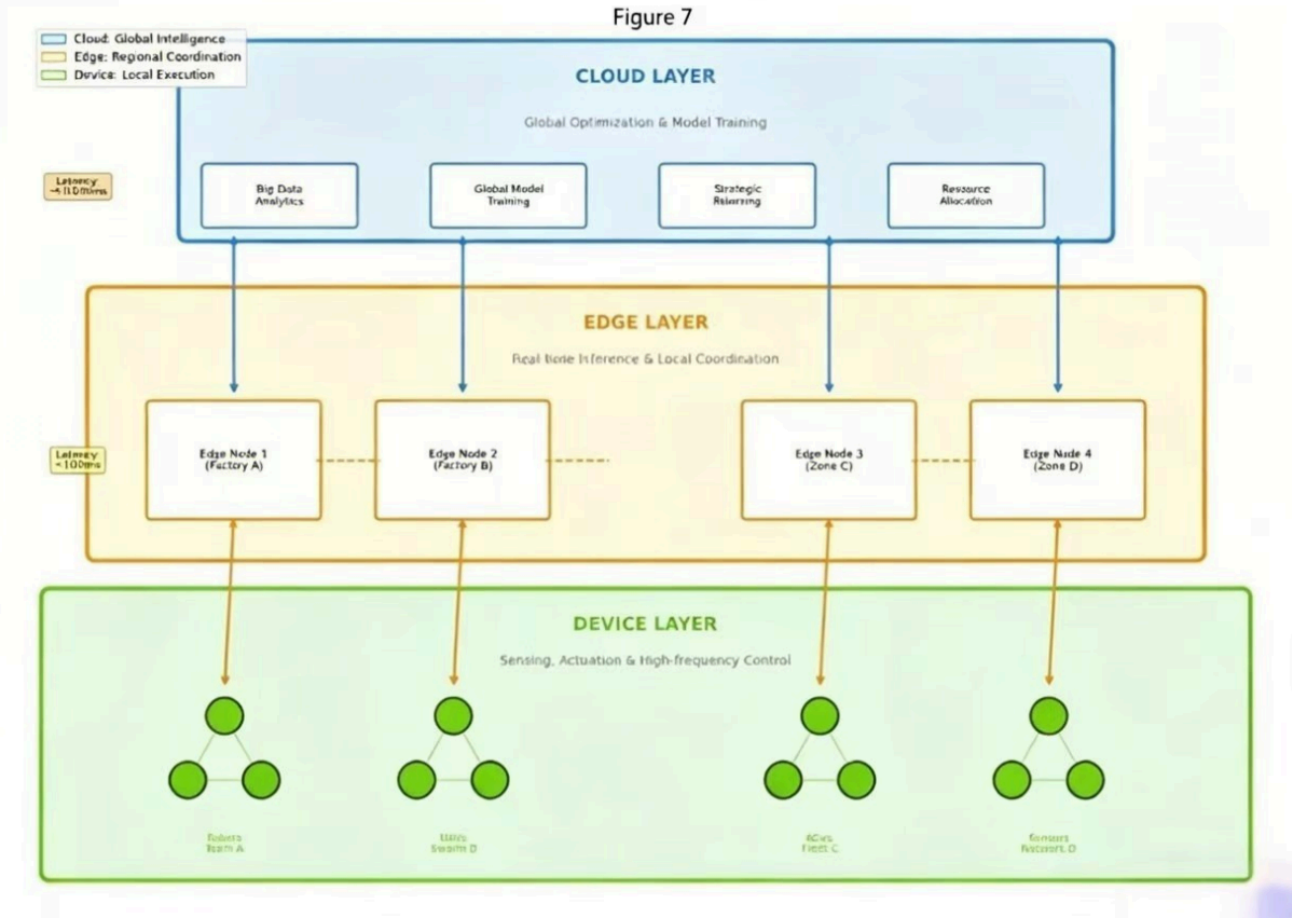

Figure 7. Three-tier architecture for agent swarm coordination

4.2.2. Digital twin ecosystems: virtual–physical co-evolution

Digital twins are attractive because real-world trial and error is expensive. A twin can expose a coordination policy to simulated faults, topology changes, and environmental disturbances before the physical swarm has to face them [15]. The risk is overbuilding the twin until it becomes too slow or too costly to use. Multi-fidelity models, online calibration, and safe exploration offer a workable middle ground: enough fidelity to catch dangerous behavior, but not so much that verification becomes impractical.

DOI: 10.54254/2755-2721/2026.36380

4.3. Cross-disciplinary fusion frontiers

4.3.1. Human–machine hybrid intelligence: neuroadaptive control architectures

Full autonomy does not remove the need for human judgment. Operators may still be better at interpreting ambiguous, ethically sensitive, or politically constrained situations, while agents are better at rapid sensing and execution [9]. Neuroadaptive control can compensate for uncertainty in human-state decoding, and autonomy allocation can shift authority when workload changes. The delicate question is timing. Human oversight must be present, but it cannot slow the swarm below the mission's response window.

4.3.2. Cross-domain system-of-systems: heterogeneous temporal orchestration

Cross-domain coordination brings scale differences that a single swarm model can easily hide. Air, ground, sea, and space systems operate with different speeds, ranges, update rates, and safety rules [1]. Singular perturbation theory gives one mathematical route for separating time scales. Time-scale-aware middleware, blockchain-supported trust, and spatiotemporal contracts may then define what each subsystem can reliably expect from the others.

4.4. Integration: toward deployable decentralized intelligence

The pathways above only make sense if they are connected. Foundation models and GNNs can improve interpretation and relational learning. Cloud-edge-device structures decide where that computation belongs. Digital twins test policies before deployment. Human-machine interfaces keep judgment available, and cross-domain frameworks keep the architecture from being trapped inside one platform class. In short, deployable decentralized intelligence is an integration problem.

5. Conclusions and outlook

5.1. Core findings

The review leaves a mixed picture. Consensus control, event-triggered coordination, and learning-based methods now give the field a shared technical vocabulary. Still, high-order nonlinear dynamics, intermittent connectivity, and safety-critical operation remain hard limits on general theory [2, 5]. Engineering practice adds its own filters: heterogeneous integration, communication adaptation, security, and environmental adaptation decide whether a method can leave simulation. AI, edge computing, and digital twins will matter most when they are tied back to control guarantees, not when they are attached as fashionable labels.

5.2. Future research priorities

The next decade should focus on four problems that are easy to name and hard to solve. Cognitive intelligence has to be fused with control theory without losing guarantees. Safety-critical swarms need formal verification that is not too slow to use. Emergent behavior must be predicted before it becomes unsafe. Large-scale applications also need governance rules, because autonomous agents increasingly operate in public and shared spaces.

DOI: 10.54254/2755-2721/2026.36380

5.3. Concluding vision

Decentralized coordination is moving toward intelligent manufacturing, smart transportation, and emergency response, but broader use will depend on more than better controllers. It will require careful links among theory, hardware, communication, security, and human oversight. The shift from centralized to decentralized, homogeneous to heterogeneous, and static to adaptive systems changes the logic of trust as much as the architecture itself. The system must not only act intelligently; it must also give people reasons to rely on it.

References

[1] Xiong, H., Deng, H., Liu, C., & Li, B. (2024). Distributed event-triggered formation control of UGV-UAV heterogeneous multi-agent systems for ground-air cooperation. *Chinese Journal of Aeronautics, 37*(12), 458–483. https://doi.org/10.1016/j.cja.2024.04.019

[2] Yan, B., Shi, P., & Chambers, J. (2024). Cooperative control for heterogeneous multi-agent systems: Progress, applications, and challenges. *Science China Information Sciences, 67*, 156201. https://doi.org/10.1007/s11432-023-1798-9

[3] Liu, D. H., Mao, Z. H., Jiang, B., & Zhang, Y. (2024). Simplified ADP-based distributed event-triggered fault-tolerant control of heterogeneous nonlinear multiagent systems with full-state constraints. *IEEE Transactions on Circuits and Systems I: Regular Papers, 71*(8), 3820–3832. https://doi.org/10.1109/TCSI.2024.3390874

[4] Olfati-Saber, R., Fax, J. A., & Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE, 95*(1), 215–233. https://doi.org/10.1109/JPROC.2006.887293

[5] Ge, X. H., Han, Q. L., Zhang, X. M., & Ding, D. (2025). Distributed coordination control of multi-agent systems under intermittent sampling and communication: A comprehensive survey. *Science China Information Sciences, 68*(5), 151201. https://doi.org/10.1007/s11432-024-4215-9

[6] Zhang, Y., Li, Y., Ran, G., & Liu, H. (2024). Dynamic event-triggered fixed-time consensus analysis of multi-agent systems under input delays and switching topologies. *IEEE 3rd Industrial Electronics Society Annual On-Line Conference*, 1–6. https://doi.org/10.1109/ONCON60544.2024.10931337

[7] Zhang, H., Wang, Y., & Liu, X. (2025). Federated multi-agent reinforcement learning: A comprehensive survey of methods, applications and challenges. *Expert Systems with Applications, 259*, 125123. https://doi.org/10.1016/j.eswa.2024.125123

[8] Dong, X., Wu, Y., Xu, M., & Li, Y. (2024). Resilient consensus for multi-agent systems in the presence of Sybil attacks. *Electronics, 13*(8), 1523. https://doi.org/10.3390/electronics13081523

[9] Xiao, W., Li, A., Cassandras, C. G., & Lygeros, J. (2024). Toward model-free safety-critical control with humans in the loop. *Annual Reviews in Control, 57*, 100944. https://doi.org/10.1016/j.arcontrol.2024.100944

[10] Ren, W., & Beard, R. W. (2008). *Distributed consensus in multi-vehicle cooperative control: Theory and applications*. Springer. https://doi.org/10.1007/978-1-84800-015-0

[11] Bhat, S. P., & Bernstein, D. S. (2000). Finite-time stability of continuous autonomous systems. *SIAM Journal on Control and Optimization, 38*(3), 751–766. https://doi.org/10.1137/S0363012997321358

[12] Wang, X., Wang, S., Liu, J., & Zhang, H. (2024). Bipartite finite-time consensus of multi-agent systems with intermittent communication via event-triggered impulsive control. *Neurocomputing, 598*, 127970. https://doi.org/10.1016/j.neucom.2024.127970

[13] Lu, J., Qin, K., Li, M., & Wang, Z. (2024). Self-triggered consensus resilient control for multi-agent systems against sensor deception attacks based on a single parameter learning method. *Chaos, Solitons & Fractals, 189*, 115513. https://doi.org/10.1016/j.chaos.2024.115513

[14] Goeckner, A., Sui, Y., Martinet, N., & Johnson, M. (2024). Graph neural network-based multi-agent reinforcement learning for resilient distributed coordination of multi-robot systems. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5732–5739. https://doi.org/10.1109/IROS59345.2024.10859672

[15] Kalyani, Y., & Collier, R. (2024). The role of multi-agents in digital twin implementation: Short survey. *ACM Computing Surveys, 57*(3), 1–15. https://doi.org/10.1145/3649531