PERSPECTIVE

Why Large Spacecraft Swarms Will Be Maintained Primarily by Design, Not Solely by Control

Chenglong Xu[1,*]

[1] School of Aeronautics and Astronautics, Sun Yat-sen University, Shenzhen 518107, China

Abstract

As orbital swarms grow toward populations of tens to hundreds, long-term maintenance changes character. We argue that it is distinct from reconfiguration or trajectory planning, defined by two requirements: it is *perpetual*, and it must be *low-frequency*, because every active maneuver arc is stolen from the payload's working time. Against these requirements the three dominant paradigms: centralized planning, distributed reactive control, and configuration design-then-maintain, fail in qualitatively different ways. Two of these failures are *structural*, properties of how the problem is posed; only the third is *fixable*. We therefore contend that scalable swarm maintenance will be achieved primarily by design rather than by control alone, provided the design phase is reformulated to treat deployment cost and extensibility as first-class objectives. We close on the resulting open question: how to lift safety from a pairwise to a collective property so that its cost stops growing with the number of pairs.

Keywords: spacecraft swarms, maintenance, passive safety, configuration design, formation flying.

Submitted: 28 May 2026
Accepted: 22 June 2026
Published: 25 June 2026

Vol. 1, **No.** 2, 2026.
10.62762/AEC.2026.392649

***Corresponding author:**
✉ Chenglong Xu
xu_cheng_long@163.com

1 Maintenance is not reconfiguration

Much of the machinery built for distributed space systems—impulsive maneuver sequences, convex trajectory planning, primer-vector solutions—was designed for *transitions*: move a formation from one configuration to another, once, fuel-optimally, without collisions on the way [1, 2]. Maintenance is a different animal. Perturbations, dominantly J_2 and differential atmospheric drag in low Earth orbit, continuously deform the geometry a deployment was meant to hold; left alone, a swarm drifts out of its safe envelope. Maintenance is the task of intervening, for the whole mission, to keep the relative motion bounded and collision-free.

Two requirements follow that reconfiguration does not share. First, maintenance is *perpetual*: there is no terminal time, only a steady-state cost rate the mission must afford for years. A useful way to see the stakes is the crude ledger

$$J_{\text{total}} \approx c_{\text{man}}\, f\, T_{\text{mission}}, \tag{1}$$

where c_{man} is the cost of a typical correction, f the maneuver frequency, and T_{mission} the lifetime.

Second, and more consequentially, f is constrained for a reason that has nothing to do with fuel: *availability*.

Citation
Xu, C. (2026). Why Large Spacecraft Swarms Will Be Maintained Primarily by Design, Not Solely by Control. *Aerospace Engineering Communications*, 1(2), 87–92.

© 2026 by the Author. Published by Institute of Central Computation and Knowledge. This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/).

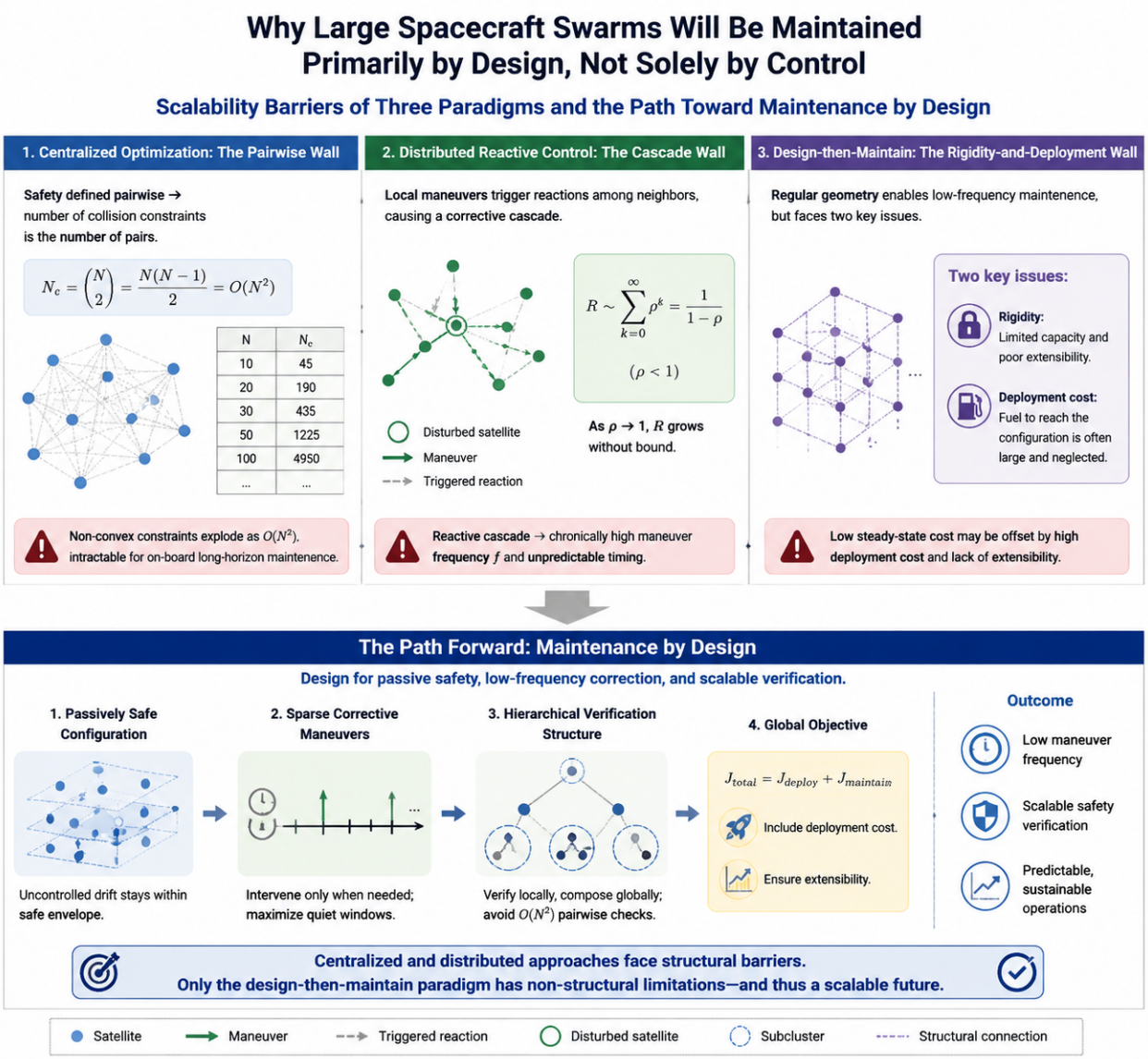


Figure 1. The figure contrasts the three dominant approaches to long-term swarm maintenance—centralized optimization, distributed reactive control, and design-then-maintain—and highlights their respective scalability barriers: the pairwise constraint explosion, the reactive maneuver cascade, and the rigidity/deployment-cost tradeoff. The lower panel illustrates the proposed path toward maintenance by design, where passively safe configurations, sparse corrective maneuvers, and hierarchical verification structures enable low-frequency, scalable, and predictable swarm operations. The central insight is that only the design-then-maintain paradigm possesses non-structural limitations and therefore remains extensible to future ultra-large spacecraft swarms.

A swarm exists to perform interferometry, distributed sensing, or inspection, and every maneuver disturbs the platform and interrupts those operations. If a correction occupies a window τ_{man}, the fraction of mission time available to the payload is

$$\eta \;=\; 1 - f\,\tau_{\mathrm{man}}. \tag{2}$$

where c_{man} is treated as independent of f, implying that each correction has a fixed cost regardless of how frequently corrections are made—an approximation that holds when the dominant cost is the correction itself rather than the setup overhead.

Equations (1)–(2) make the tension explicit: both fuel and availability push f down, yet perturbations push it up. The central design question of maintenance is therefore not "what is the optimal maneuver?" but "how rarely can we afford to maneuver at all?"

This is what makes *passive safety* the natural foundation for maintenance. Rather than continuously enforcing separation, one engineers the uncontrolled relative motion so that the minimum inter-satellite distance over a drift window already satisfies

$$d_{\min}(t) \;\geq\; \epsilon, \qquad t \in [\,t_0,\; t_0 + \Delta t_{\mathrm{drift}}\,], \tag{3}$$

for a keep-out radius ϵ, *without any control input* on that interval [3, 4]. A passively safe configuration buys a control-free window; maintenance, properly posed, is the problem of *renewing* condition (3) as rarely as possible. With that lens, the paradigms below should be judged not by how cleverly they compute a maneuver, but by how the renewal of safety scales with swarm size.

Figure 1 summarizes these paradigms and their scalability barriers, foreshadowing the design-oriented path developed in Sections 2 and 3.

It is worth stating plainly what maintenance can and cannot do. Because perturbations never stop, no scheme drives deviations to zero; the best one can hope for is to trap the swarm inside a safe shell and pay a perpetual residual to keep it there. The generic Lyapunov picture captures this: for a maintenance error x with energy $V(x)$, a well-posed scheme yields

$$\dot{V} \;\le\; -a\,V + b \quad\Longrightarrow\quad \limsup_{t\to\infty} V \le \frac{b}{a}, \tag{4}$$

a bounded but nonzero residual b/a that is the true recurring price of maintenance. Everything that follows is about keeping that price—and the frequency that pays it—from exploding as N grows.

We stress that Eqs. (1)–(4) and (6) below are order-of-magnitude conceptual models, intended to expose the governing tradeoffs rather than to serve as validated analytical results; the parameters (c_{man}, f, τ_{man}, ρ, a, b) are representative rather than measured, and the relations assume a single dominant correction cost and weak, slowly varying coupling.

For precision, we use the following terms throughout. *Maintenance* is the perpetual, low-frequency intervention that keeps the relative motion bounded and collision-free over the whole mission, as distinct from a one-time *reconfiguration*. *Passive safety* is the property (3) that the uncontrolled relative motion satisfies the keep-out radius over a finite drift window. A *quiet window* is such a control-free interval $\Delta t_{\mathrm{drift}}$, during which the payload operates undisturbed. *Deployment cost* is the fuel required to transport an arbitrary post-launch state onto the prescribed configuration—the entry term routinely omitted from (1). *Extensibility* is the ability to add satellites without the safety check reverting to the pairwise count (5). *Collective safety*, made precise in Section 4, denotes a safety constraint posed on the swarm's spatial distribution rather than on individual pairs.

2 Three paradigms, three barriers

2.1 Centralized optimization: the pairwise wall

The most direct approach poses maintenance as one optimal control problem: minimize total impulse subject to a keep-out constraint between every pair of satellites and the swarm dynamics. The obstacle is intrinsic and appears before any solver is chosen. Safety is defined *pairwise*, so the number of collision constraints is the number of pairs,

$$N_c \;=\; \binom{N}{2} \;=\; \frac{N(N-1)}{2} \;=\; \mathcal{O}(N^2). \tag{5}$$

Each such constraint is non-convex, and over a long planning horizon the discretized problem suffers a combinatorial blow-up that is intractable on board. Concretely, N_c rises from 45 at $N = 10$ to 1225 at $N = 50$ and 4950 at $N = 100$.

The literature mitigates this by imposing *regularity*—for example, organizing the swarm into structured subclusters so that verification touches only a structured subset rather than all pairs, scaling the check linearly rather than quadratically [5]. Such tricks shrink the constant in front of (5), but they do not remove the object: safety is still checked two satellites at a time. The wall is not the size of N; it is the pairwise definition itself. This is the essential cost of centralized maintenance.

2.2 Distributed reactive control: the cascade wall

The natural escape from $\mathcal{O}(N^2)$ central computation is to push decisions to the edge: let each satellite regulate its own neighbors through consensus rules or artificial potential fields [6–8]. These rules are cheap and robust to arbitrary initial states.

But distribution trades a computational problem for a *behavioral* one, and the behavioral one is fatal to the low-frequency requirement of Section 1. Without global state, a satellite that maneuvers to fix one separation perturbs its neighbors, who react in turn, who perturb *their* neighbors. If a single correction provokes on average ρ further corrections, the total number of corrections triggered by one disturbance behaves like a geometric cascade,

$$R \;\sim\; \sum_{k=0}^{\infty} \rho^{k} \;=\; \frac{1}{1-\rho} \quad (\rho < 1), \tag{6}$$

which grows without bound as the inter-satellite coupling $\rho \to 1$.

This is precisely where maintenance diverges from trajectory planning: a planner computes once, over a known horizon, with a fixed endpoint; a reactive maintainer responds forever, and its responses trigger further responses. The consequence is chronically high f in (1)–(2) and, because the cascade is unpredictable, maneuvers that fire at unpredictable times—platform jitter exactly when a payload needs stillness. Distributed control can keep a swarm safe and bounded; what it cannot do, structurally, is keep it *quiet*. And quiet is the whole point.

This critique targets purely local, reactive schemes that act on instantaneous neighbor states. We do not claim that every distributed method incurs the cascade: event-triggered and self-triggered control, distributed model-predictive control, and hierarchical or hybrid coordination can substantially suppress f and partly tame the cascade by maneuvering only when a predicted violation is imminent or by injecting limited global structure. The bound (6) should therefore be read as the worst-case reactive regime ($\rho \to 1$): these refinements lower the effective coupling ρ, but they do not change its origin—decisions made without global state—and whether ρ can be held uniformly low across an entire swarm and an entire mission remains open.

2.3 Design-then-maintain: the rigidity-and-deployment wall

The third paradigm front-loads the intelligence into the configuration: design a relative geometry whose uncontrolled evolution is benign, and pair it with a sparse control law that periodically restores it [9, 10]. Because the geometry is regular, safety verification scales gracefully; because it is engineered for control-free drift, it naturally delivers the long quiet windows the payload wants.

Its problems are of a different kind. The geometry tends to be *fixed too rigidly*: a configuration tuned for a particular swarm size and separation budget has a fixed accommodation capacity and does not straightforwardly extend to different swarm sizes [10, 11]. Worse for the ledger (1), *entering* such a configuration is not free—transporting an arbitrary post-launch state onto the prescribed geometry can cost substantial fuel, a *deployment cost* that is routinely left out of analyses advertising steady-state efficiency. A scheme cheap to maintain but expensive to enter can lose its advantage entirely on missions that re-acquire often.

3 The asymmetry that decides the question

Lining up the three barriers reveals an asymmetry that, to us, settles which paradigm large swarms will use. The centralized wall (the pairwise count, Eq. (5)) and the distributed wall (the cascade, Eq. (6)) are *structural*: they follow from how the problem is posed—safety checked two-at-a-time, or decisions made without global state—and they do not yield to a better solver or better gains.

The design-then-maintain barrier is *not* structural. Rigidity and unaccounted deployment cost are not inherent to designing a configuration; they are symptoms of designing the configuration for the wrong objective. A configuration chosen only for steady-state drift will of course be rigid, and will of course ignore the cost of reaching it—because those terms were never in the objective. The remedy is correspondingly direct: put them in.

This is the load-bearing claim of this Perspective. At scale, only the design-then-maintain paradigm has a future, because only its failures are fixable. The path forward is not to abandon configuration design but to *reformulate* it so that the deployment term of (1) and the swarm's extensibility are designed in from the start, rather than discovered afterward.

4 Toward maintenance by design

What, then, should a scalable design-then-maintain scheme optimize? Two requirements that the current paradigm omits.

First, *global cost*, not steady-state cost. The honest objective spans the whole mission—the fuel to deploy onto the configuration *plus* the fuel to hold it—and a configuration should be chosen partly for how cheaply an arbitrary initial state can reach it. Minimizing steady-state cost in isolation can invert the apparent ranking of methods the moment deployment is counted.

Second, *extensibility*. The design must keep safety verification from collapsing back into the pairwise count (5) as satellites are added. Structured, hierarchical geometries are the most promising route, because regularity is exactly what converts a global $\mathcal{O}(N^2)$ check into a sum of small, local ones [11]. A configuration that cannot grow without re-incurring the centralized wall is not a solution to the scaling problem; it is the centralized wall in disguise.

These two requirements make the paradigm scalable in cost and computation. They do not yet make it *provably*

safe at scale, which is the deepest open problem, and it has a clean diagnosis. The cleanest safety guarantees in the field are closed-form precisely because they treat the *binary* case—two satellites and a single forbidden region in the relevant geometric plane—together with an exact maneuver to stay out of it [11, 12]. These primitives are elegant and cheap, but they do not compose: for three or more satellites the pairwise forbidden regions overlap into a fragmented, irregular safe set, and the single-region analytical solution is no longer valid [11]. The very feature that makes the binary problem solvable disappears as the swarm grows.

We read this as a symptom of a single underlying mismatch: safety is currently a *pairwise* property imposed on an *individual* description of each satellite, while the system it must protect is *collective*. The two structural walls of Section 2 are two faces of this mismatch—the pairwise count and the local cascade. The most valuable open direction, in our view, is therefore to lift safety from a pairwise to a collective property: a constraint on the *distribution* of the swarm in the relevant geometric space, so that its cost scales with the number of occupied regions rather than the number of pairs. Concretely, in place of the $N(N-1)/2$ pairwise keep-out constraints, collective safety would constrain the swarm's empirical distribution in the relevant relative eccentricity/inclination plane—for instance, bounding the occupancy density of any cell of that plane so that no cell packs satellites closer than the keep-out radius ϵ permits—reducing the constraint count to the number of occupied cells. Whether this is approached through provably safe hierarchical decomposition—binary primitives inside subclusters with guaranteed interfaces between them—or through a genuinely distribution-level reformulation, the target is the same: make safety an object whose price does not grow like (5).

5 Outlook

Read this way, the field's trajectory is coherent: it has learned to design safe configurations and bound their capacity, to optimize them under structure, and to maintain the binary case analytically and efficiently. Each step has chipped at the same conclusion—at scale, control is a poor substitute for design.

Real challenges remain. Hierarchical decomposition needs provably safe interfaces between subclusters, not merely empirically safe ones. The inner optimization loops remain a computational bottleneck, and replacing them with learned surrogates or reinforcement learning policies is an attractive but unproven route to on-board autonomy. The configurations safest and cheapest to hold may be the most awkward to control during the transitions themselves. And as electric propulsion displaces impulsive maneuvering, the whole analysis must be re-derived for continuous, low-thrust transitions.

None of these reopens the central question. Centralized maintenance is walled in by pairwise checking (5); distributed maintenance is walled in by the cascade (6) and the high frequency it forces through (2); both walls are properties of the formulation and will not fall to better engineering. The configuration-based paradigm alone has remediable flaws—and the remedy, designing for deployment cost and extensibility rather than steady-state efficiency alone, is the direction worth pursuing. Large spacecraft swarms, we conclude, will be kept safe not by controlling their way out of danger but by being arranged so that danger rarely arrives, and so that restoring safety, when it must be done, is done by design: predictably, cheaply, and rarely.

Data Availability Statement

Not applicable.

Funding

This work was supported without any funding.

Conflicts of Interest

The author declares no conflicts of interest.

AI Use Statement

The author declares that AI tools (ChatGPT and Claude Opus 4.7) were used solely for language editing and polishing of the manuscript text. The author has carefully reviewed and verified all AI-assisted content and takes full responsibility for the accuracy, integrity, and originality of the manuscript.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Gaias, G., & D'Amico, S. (2015). Impulsive maneuvers for formation reconfiguration using relative orbital elements. *Journal of Guidance, Control, and Dynamics, 38*(6), 1036-1049. [CrossRef]

[2] Chernick, M., & D'Amico, S. (2018). New closed-form solutions for optimal impulsive control of spacecraft relative motion. *Journal of Guidance, Control, and Dynamics, 41*(2), 301-319. [CrossRef]

[3] D'Amico, S., & Montenbruck, O. (2006). Proximity operations of formation-flying spacecraft using an eccentricity/inclination vector separation. *Journal of Guidance, Control, and Dynamics, 29*(3), 554-563. [CrossRef]

[4] Guffanti, T., & D'Amico, S. (2023). Passively safe and robust multi-agent optimal control with application to distributed space systems. *Journal of Guidance, Control, and Dynamics, 46*(8), 1448-1469. [CrossRef]

[5] Chenglong, X. U., Zhang, C., & Jihe, W. A. N. G. (2025). Passively safe configuration design for spacecraft swarm flying with boundary constraints. *Chinese Journal of Aeronautics, 38*(8), 103595. [CrossRef]

[6] Wang, Z., Xu, Y., Jiang, C., & Zhang, Y. (2019). Self-organizing control for satellite clusters using artificial potential function in terms of relative orbital elements. *Aerospace Science and Technology, 84*, 799-811. [CrossRef]

[7] Chen, Q., Meng, Y., Liao, Y., & Wei, C. (2023). Intersatellite distance-keeping control based on relative motion geometry. *Journal of Guidance, Control, and Dynamics, 46*(1), 177-185. [CrossRef]

[8] Chen, Q., Jin, H., Wei, C., & Liao, Y. (2025). Distributed impulsive satellite swarm-keeping based on connectivity and relative Cartesian states. *Journal of Guidance, Control, and Dynamics, 48*(2), 269-281. [CrossRef]

[9] Morgan, D., Chung, S. J., Blackmore, L., Acikmese, B., Bayard, D., & Hadaegh, F. Y. (2012). Swarm-keeping strategies for spacecraft under J2 and atmospheric drag perturbations. *Journal of Guidance, Control, and Dynamics, 35*(5), 1492-1506. [CrossRef]

[10] Koenig, A. W., & D'Amico, S. (2018). Robust and safe N-spacecraft swarming in perturbed near-circular orbits. *Journal of Guidance, Control, and Dynamics, 41*(8), 1643-1662. [CrossRef]

[11] Xu, C., Zhang, C., & Wang, J. (2025). Passive safety maintenance for long-term spacecraft cluster flying utilizing relative motion perturbed evolution law. *Aerospace Science and Technology, 164*, 110402. [CrossRef]

[12] Xu, C., Zhang, C., & Wang, J. (2025). Analytic solution for combined in-plane and out-of-plane spacecraft formation reconfiguration with passive collision avoidance. *Acta Astronautica, 226*, 48-59. [CrossRef]



Chenglong Xu received the B.S. and M.S. degree in College of Aerospace Engineering from Harbin Engineering University, Harbin 150009, China, in 2020. (Email: xu_cheng_long@163.com)