Intelligent Automation & Soft Computing

Tech Science Press

https://doi.org/10.32604/iasc.2026.091366

ARTICLE

Designing Energy Efficient Swarm Robotics by Optimization of Communication and Computation Costs

Amir Ijaz[1,*], **Hashem Haghbayan**[1], **Juha Plosila**[1] **and Ethiopia Nigussie**[2]

[1]Department of Computing, University of Turku, Turku, Finland
[2]Faculty of Technology, University of Turku, Turku, Finland

*Corresponding Author: Amir Ijaz. Email: amir.ijaz@utu.fi or amirijaz@live.com

Received: 20 August 2026; Accepted: 20 September 2026; Published: 23 October 2026

ABSTRACT: Small robots that operate as part of a swarm require careful, fine-grained energy management: locomotion is, and is expected to remain, the dominant term in a mobile robot's energy budget, typically several orders of magnitude larger than radio communication and often larger than other on-board loads such as ranging sensors or path-planning computation. Energy-saving communication and computation strategies are, however, comparatively less studied than locomotion-energy optimization, particularly for the low-mobility or stationary phases of a swarm mission (e.g., perimeter monitoring, docked sensing, or loiter/hover periods) in which locomotion is intermittent or absent and radio and compute loads become the controllable, mission-relevant cost. This paper studies, within that deliberately narrowed scope, how jointly, rather than separately, optimizing communication and computation costs affects swarm-level energy consumption during such phases; it does not claim communication or computation is the dominant cost of swarm robotics in general, and locomotion energy is explicitly outside this paper's model. The joint energy model is completed with an explicit reporting-rate function, reception/aggregation, control-packet, idle-listening and scheduler-overhead terms, and the underlying decision problem is stated as a constrained optimization with an explicit two-stage solution procedure. A Joint Communication-Computation Optimization (JC2O) framework combining cluster-based relaying, event-triggered reporting, role-differentiated (fixed-frequency) local compute scaling and payload compression is evaluated through a reproducible Monte Carlo simulation, reported with 95% confidence intervals, against an untuned baseline and, in an ablation study, against each mechanism in isolation. The ablation study reveals a non-obvious interaction: clustering alone, without compression, can increase total energy relative to the baseline, because an uncompressed aggregated payload relayed over a long hop is penalized heavily by the fourth-power path-loss term; the full combination nonetheless reduces the modelled communication-and-computation energy by 74%–77% across swarm sizes from 5 to 150 robots. A freshness metric ties the event-triggering threshold to expected reporting staleness, giving an explicit energy-freshness trade-off in place of energy alone, and sensitivity analyses quantify the previously unmodelled costs of cluster-formation control traffic, idle listening, channel contention and robot mobility. A separate, explicitly reactive online-adaptation experiment, distinct from the fixed-frequency compute scaling above, shows that a feedback policy that adjusts the event-triggering threshold and compression ratio in response to battery state and channel quality extends simulated operational lifetime by more than 10% over a static configuration under identical time-varying conditions, at a scheduler recomputation overhead below 10^{-4}% of battery capacity. These results give swarm-robotics designers concrete, quantified guidance on where communication- and computation-side energy-saving effort should be concentrated once locomotion has been addressed separately, and on how the underlying mechanisms interact, rather than treating communication and computation protocols as independently optimizable subsystems.

Copyright © 2026 The Authors. Published by Tech Science Press. This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

KEYWORDS: Swarm robotics; energy efficiency; communication-computation co-design; edge computing; wireless sensor networks; multi-robot systems

1 Introduction

Swarm robotics replaces the classical single, capable robot with a large number of comparatively simple ones that achieve a mission collectively through local sensing, local computation and local communication rather than a central controller [1,2]. The appeal of the approach is well established: swarms are redundant against individual failures, they scale naturally to the size of a task, and they can be produced far more cheaply per unit than a single monolithic robot with equivalent coverage. Each swarm unit, however, carries only a small battery, since keeping unit cost and unit mass low is part of what makes the swarm paradigm attractive in the first place [3], and fine-grained energy management is correspondingly important. It should be stated plainly at the outset, to avoid any risk of overstating this paper's scope: locomotion is, and is expected to remain, the dominant term in a mobile robot's energy budget, typically several orders of magnitude larger than radio communication energy, and often larger than other on-board loads such as ranging sensors (e.g., lidar) or the computation required for path planning and obstacle avoidance. Nothing in this paper contests that. What motivates this paper instead is that energy-saving communication and, to a lesser extent, computation strategies are comparatively less studied than locomotion-energy optimization, particularly for the low-mobility or stationary phases of a swarm mission, perimeter monitoring, docked or anchored sensing, loiter or hover periods, in which locomotion is intermittent or largely absent and radio and on-board compute become the controllable, mission-relevant cost. This paper's scope is deliberately restricted to that regime; a companion study by the same authors [4], discussed in Section 2.4, addresses the joint co-management of communication, computation and drivetrain energy together, for phases of a mission where locomotion cannot be treated separately.

Two activities dominate a swarm robot's energy budget within the scope just described, i.e., once locomotion is set aside: exchanging messages with neighbours or a base station, and running the on-board computation needed to sense, decide and act. Both have received substantial independent attention. The wireless-sensor-network community has spent two decades refining communication protocols, clustering, duty-cycling, adaptive transmission power and data aggregation, that reduce the energy cost of moving bits across a network [5]. Separately, the mobile- and edge-computing community has developed task-offloading and dynamic-voltage-and-frequency-scaling (DVFS) techniques that reduce the energy cost of processing bits locally [6,7]. Multi-robot-systems research has, in parallel, developed energy-aware task-allocation frameworks that treat a robot's remaining energy as a first-class scheduling constraint [8,9]. Swarm robotics inherits all three literatures, yet, within the communication-and-computation sub-problem this paper addresses, a robot's communication and computation decisions are rarely free of each other: compressing a message before transmission spends CPU cycles to save radio energy; offloading a task to a neighbour saves CPU cycles but spends radio energy; and how aggressively a robot samples its environment determines both how much there is to compute and how much there is to send. Treating communication and computation as two independently optimizable subsystems, as much of the existing literature implicitly does, can therefore leave savings on the table, or worse, allow an optimization applied to one subsystem to inadvertently increase the cost of the other, a concrete instance of which is quantified directly in this paper's ablation study (Section 6.3).

This paper studies the joint optimization of communication and computation costs, for the low-mobility or stationary phases of a swarm mission where these two loads become the controllable cost (locomotion is addressed separately by the authors' own companion work [4], discussed in Section 2.4). Within that scope,

four contributions are made, each stated here at the modest scale it is intended: this paper trades system-level performance for analytical transparency relative to the authors' own more capable, learned or jointly-solved prior systems [4,10,11], a trade-off Section 2.4 discusses explicitly. First, a per-robot energy model is presented that couples a first-order radio energy model [5] with a DVFS-based computation model and explicit reception/aggregation, control-packet, idle-listening, scheduler-overhead and contention/retransmission terms, together with a closed-form reporting-rate function and a formally stated constrained optimization problem with a documented two-stage solution procedure (Section 3). Second, building on that model, a Joint Communication-Computation Optimization (JC2O) framework is proposed that combines clustered relaying in the style of the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol, event-triggered reporting, role-differentiated (fixed-frequency) DVFS compute scaling and online-adaptive payload compression into one coordinated, fully closed-form scheduling layer, deliberately simpler than the learned controllers of [4,11], precisely so that each mechanism can be switched off independently (Section 4); a separate, genuinely adaptive extension that adjusts the event-triggering threshold and compression ratio in response to time-varying battery and channel state is evaluated independently in Section 6.8. Third, the framework is evaluated through a reproducible Monte Carlo simulation study, reported throughout with 95% confidence intervals from a single, explicitly seeded pipeline, that quantifies not only the aggregate energy saving but also, through an ablation study none of the authors' prior learned or distributed-optimization systems report, the individual and interactive contribution of each mechanism, and, through a freshness metric and sensitivity analyses, the energy-freshness trade-off and the effect of previously unmodelled costs such as mobility and channel contention (Section 6). Fourth, an online-adaptation experiment demonstrates and quantifies the benefit and overhead of adjusting the scheme's parameters in response to time-varying battery and channel conditions, directly addressing the framework's claimed suitability for online operation (Section 6.8).

The remainder of the paper is organized as follows. Section 2 reviews related work on energy-efficient swarm robotics, wireless communication protocols and edge/task-offloading computation, broadens coverage of the wireless-sensor-network energy-routing literature, and, in Section 2.4, states explicitly how this paper relates to, and is narrower than, the authors' own four closely related recent publications. Section 3 formulates the complete joint energy model, its overhead and freshness terms, and the formal optimization problem. Section 4 describes the proposed JC2O framework, its constituent mechanisms and the two-stage calibration-then-scheduling procedure that solves the optimization problem in practice. Section 5 details the simulation methodology, parameters, seeds and statistical procedure used to evaluate it. Section 6 presents and discusses the results, including the new ablation study, energy-freshness trade-off, sensitivity analyses and online-adaptation experiment. Section 7 discusses the limitations that remain after this revision, and Section 8 concludes with directions for future work.

2 Related Work

2.1 Energy Efficiency in Swarm Robotics

Interest in the energy dimension of swarm robotics has grown alongside the field itself. Dorigo et al. [12], in a widely cited retrospective on the field, identify energy autonomy as one of the main practical bottlenecks preventing laboratory swarm demonstrations from becoming field-deployable systems, noting that few real deployments run for more than a handful of hours before requiring recharge and that most reported systems still treat energy management as an afterthought layered on top of a communication or control stack designed for other purposes. Debie et al. [13] survey swarm robotics from a multi-tasking perspective and discuss controllers built around a virtual energy construct: an internal scheduling variable, not a physical quantity delivered to the battery, that accrues on task completion and depletes with locomotion, sensing

and communication, and that a colony-inspired controller uses to decide which sub-task to prioritize at any moment. This virtual-energy accounting is a well-established multi-agent scheduling mechanism, tracing back to central-place-foraging models of social insects, and should not be read as robots physically recovering battery charge by completing tasks; the present paper's energy model (Section 3), by contrast, tracks only physical energy actually dissipated by the radio and processor. Application-specific studies reinforce the energy-autonomy point: Yasin et al. [14] propose an information-theoretic sensing policy that avoids "over-sensation" in an autonomous swarm navigating around obstacles, reducing sensing-driven energy waste without materially harming navigation accuracy, while Aznar et al. [15] tune the ad-hoc networking behaviour of an indoor unmanned-aerial-vehicle swarm specifically to extend flight time. Yasser et al. [16] propose two decentralized, cluster-based swarm communication algorithms explicitly aimed at reducing both execution time and power consumption as swarm size grows, reporting that message traffic, and hence energy, scales unfavourably with the number of robots under a naively decentralized scheme, a scaling concern this paper's own swarm-size sweep (Section 6.1) addresses directly for the clustering-plus-compression combination. More recently, energy-aware task allocation has emerged as a closely related but distinct strand: Notomista et al. [8] propose a resilient allocation framework in which heterogeneous robots' remaining energy directly shapes which tasks they are assigned, and Mokhtari et al. [9] schedule tasks across an ambiently-powered robot swarm using meta-heuristic optimization over predicted harvested energy. Both treat energy as a scheduling constraint on what a robot does; the present paper instead treats energy as a design variable governing how a robot communicates and computes while carrying out a given task, so the two strands are complementary rather than competing, and a swarm could in principle combine an energy-aware task allocator with the communication-computation scheduler proposed here.

Most recently, and most closely related to this paper, Ijaz et al. [10] present EACCO (the name given to the framework in that paper), a cross-layer design, validated on both microcontroller-class swarm nodes and edge accelerators, that jointly models computation and communication energy and schedules tasks according to a composite priority-energy-deadline score, reporting sustained energy-neutral operation under ambient energy harvesting. The present paper's relationship to EACCO is deliberately narrower and complementary rather than an alternative solution to the same problem: EACCO's scheduling decision is which of several pending tasks to run and when, under an energy-harvesting budget, validated on physical hardware at a small, fixed swarm size; this paper's scheduling decision is, for a single recurring sensing-and-reporting task, how a robot should split effort between clustering topology, event-triggering, local compute frequency and payload compression, evaluated in simulation across swarm sizes from 5 to 150 robots specifically to isolate how the resulting savings and their composition scale with N, a question EACCO's fixed-size hardware evaluation does not address and that is difficult to probe experimentally once swarm sizes reach dozens or hundreds of units. Concretely, EACCO does not vary or report results across swarm size, does not decompose its saving into clustering, event-triggering, DVFS and compression contributions, and does not report an energy-freshness trade-off curve; this paper does all three, at the cost of using an analytical, simulation-based energy model rather than measured hardware power draw. Sections 6.3 and 6.4 return to this comparison quantitatively and discuss why a direct, controlled, head-to-head numerical comparison against EACCO's published results is not presently reproducible under a common protocol.

2.2 Energy-Efficient Communication Protocols

The wireless-sensor-network literature offers the most mature toolbox for reducing the communication component of swarm energy. Heinzelman et al. [5] introduced the LEACH protocol, in which nodes self-organize into clusters, elect a cluster head on a rotating basis, and route their data through that head rather than transmitting directly to a distant base station; because radio transmission energy grows with (at least)

the square of distance in free space and the fourth power of distance under multipath propagation [17], replacing many long hops with many short hops plus a few long ones yields a substantial net saving provided the aggregated payload relayed over the long hop is not itself allowed to grow unchecked, a caveat this paper's ablation study shows to matter in practice (Section 6.3). The same clustering logic transfers naturally to mobile robot swarms whose members are, in effect, mobile sensor nodes. Event-triggered communication is a second, complementary lever: rather than reporting on a fixed schedule, a node transmits only when its locally observed state changes by more than a threshold, which Byun and Yang [18] show, in a self-triggered control scheme for robot swarm networking, can maintain formation and coordination accuracy while substantially reducing the number of messages exchanged. Gielis et al. [19] review communication strategies specifically for multi-robot systems and highlight that the choice between broadcast, unicast and relay-based communication, and the frequency with which it is invoked, has first-order consequences for both energy consumption and scalability as swarm size grows, a tension this paper quantifies directly in Section 6.

The broader wireless-sensor-network energy-routing literature, of which LEACH-style clustering is one long-standing member, is considerably larger than the handful of protocol-specific studies above, and three recent surveys help place this paper's mechanisms within it. Nakas et al. [20] survey energy-efficient routing broadly and organize the field around the same layers this paper's overhead model targets individually: duty-cycling (Section 3.4's idle-listening term), clustering and data aggregation (Section 4.1), and cross-layer design, of which the jointly scheduled JC2O framework is one instance. Bomnale and More [21] focus specifically on data-aggregation and routing protocols and reinforce a point this paper's ablation study makes quantitatively (Section 6.3): aggregation reduces the number of transmissions but, unless the aggregated payload is itself compressed or otherwise bounded, does not by itself reduce the number of bits that must eventually cross the most expensive hop. Priyadarshi [22] reviews the more recent shift toward meta-heuristic and AI-based energy-aware routing, a direction largely orthogonal to this paper's closed-form, analytically calibrated approach (Section 3.6) but a natural candidate for replacing this paper's grid-search calibration of δ and r in future work, as discussed in Section 8.

2.3 Computation Offloading and Edge Computing

On the computation side, mobile- and multi-access edge computing (MEC) research addresses the trade-off between computing a task locally and offloading it, at a communication cost, to a more capable nearby platform. Shi et al. [6] frame edge computing broadly as moving computation and data storage closer to where they are needed, reducing both latency and the energy spent on long-haul data transport. Liang et al. [7] formalize this trade-off for multi-user MEC systems, jointly optimizing transmission time, computation-capacity allocation and data partitioning to minimize the weighted energy consumption of a set of resource-constrained devices; although their setting is cellular rather than robotic, the underlying convex-optimization structure, energy as a function of how a task is split between local computation and offloaded computation, is directly applicable to a swarm robot deciding whether to process a sensor reading itself or delegate it to a cluster head. DVFS, the mechanism most commonly used to make local computation itself more efficient, exploits the approximately cubic relationship between a processor's clock frequency and its dynamic power draw [6,17]: modestly reducing frequency for latency-tolerant tasks yields a disproportionate energy saving, at the cost of longer task completion time. This paper adopts that cubic DVFS relationship for the computation term of its energy model (Section 3) and combines it with the clustering and event-triggering mechanisms described above into a single, jointly tunable optimization problem, which is the gap the reviewed literature, largely organized around either communication-side or computation-side optimization, leaves open.

2.4 Positioning Relative to the Authors' Prior Work

The authors have published a series of closely related studies over the past year, and this subsection states plainly how the present paper relates to each of them, since the relationship is close enough that it needs to be made explicit rather than left for a reader to reconstruct. Ijaz et al. [23] is an early-stage, communication-only study that uses a Deep-Q-Learning controller to adapt transmission strategy and power for energy-efficient Internet-of-Things (IoT) communication in a swarm-robotics setting; it does not address computation cost. Ijaz et al. [10] (EACCO) jointly schedules computation and communication on real microcontroller-class hardware and edge accelerators using a composite priority-energy-deadline score, and reports sustained energy-neutral operation under ambient harvesting at a fixed, small swarm size. Ijaz et al. [11] (JCCO) is, of the four, the closest in scope to the present paper: it jointly optimizes task offloading, bandwidth allocation, transmission power and computational workload under latency and energy constraints, using a distributed optimization strategy combined with a lightweight deep-reinforcement-learning controller, and reports up to 41% swarm energy reduction with over 99% deadline satisfaction across varying swarm sizes and communication conditions. Ijaz et al. [4] extends the scope further still, formulating the co-management of processor, radio and drivetrain energy for a fleet of edge-assisted mobile robots as a long-run stochastic optimization under task-deadline, power-budget and mission-progress constraints, solved via a Lyapunov drift-plus-penalty decomposition into per-slot clock-frequency, transmit-power and scheduling subproblems.

Given this, JCCO [11] already performs joint communication-computation optimization with a learned controller and already reports results across varying swarm sizes, so that combination is not this paper's contribution to make. What this paper does instead is complementary rather than competing: it replaces the learned, distributed controllers of [4,11] with a fully closed-form, non-learned energy model (Section 3) whose every term is an explicit equation rather than a trained policy, specifically so that its four constituent mechanisms, clustering, event-triggering, DVFS and compression, can be ablated individually and in combination (Section 6.3) to show why joint optimization helps and where the benefit comes from, including the non-obvious clustering-without-compression interaction effect reported there; none of the four prior studies report a per-mechanism ablation, since their learned or jointly-solved controllers do not decompose into independently switchable parts in the same way. In that sense, this paper is best read as a simpler, fully transparent, diagnostic companion to the more capable but less interpretable systems in [4,10,11], rather than as a new best-performing system in its own right; it does not claim to outperform JCCO's learned controller or the Lyapunov-based co-management framework of [4], and Section 7 states this explicitly. Readers whose interest is primarily in achieving the strongest reported numbers should consult [4,11] directly; readers whose interest is in understanding, mechanism by mechanism, why and under what conditions joint communication-computation optimization helps a clustered swarm are the intended audience for the present paper.

3 Energy, Overhead and Performance Model

A swarm of N homogeneous robots is considered, deployed over a bounded planar field of side length L, reporting either directly or via relays to a fixed base station located at distance d_{bs} from the field centre. Time is discretized into rounds (or epochs) indexed by t; in each round every robot may sense its environment, perform local computation, and transmit a message. This section completes the model relative to the previous version of the paper by making the reporting-rate function explicit (Section 3.3), adding reception, control-packet, idle-listening and scheduler-overhead terms (Section 3.4), adding a contention/retransmission model (Section 3.5), and stating the resulting design problem as a constrained optimization with an explicit decision-variable scope and solution procedure (Section 3.6).

Section Nomenclature summarizes the notation used throughout; Table 1 in Section 5 lists every parameter's numerical default.

Table 1: Simulation parameters and default values.

Parameter	Description	Value
E_{elec}	Radio electronics energy	50 nJ/bit
ε_{fs}	Free-space amplifier constant	10 pJ/bit/m^2
ε_{mp}	Multipath amplifier constant	0.0013 pJ/bit/m^4
κ	Dynamic CPU power at $f = 1$	15 mW
C	Cycles per full local task	4×10^6
$F_{\max}$	Maximum clock frequency	100 MHz
γ	Compression cost per bit removed	10 nJ/bit
b	Raw payload per update	2000 bits
$b_{\text{agg}} = r\,(1 + n_i)\,b$	Cluster-head aggregated payload (Eq. (3)); varies by cluster; at $N = 50$, $k = 5$, $n_i \approx 9$, $r = 0.35$: $b_{\text{agg}} \approx 7000$ bits	See Eq. (3)
L	Field side length	100 m
d_{bs}	Field-centre-to-base-station distance	120 m
b_{ctrl}	Control-packet size	200 bits
p_{adv}	Cluster-head advertisement probability	0.30
T_{ctrl}	Cluster-rotation period	20 rounds
P_{idle}	Idle-listening power	0.12 mW
T_{idle}	Idle-listening window	10 ms
C_{sched}	Scheduler decision cost	2×10^4 cycles
n_{slot}	Contention slots per cluster round	8
k	Cluster-head count	$\lceil \sqrt{N/2} \rceil$ (analytic)
δ (default)	Event-triggering threshold	0.35 (calibrated)
r (default)	Compression ratio	0.35 (calibrated)
$f_{\text{mem}}, f_{\text{ch}}$	Member/cluster-head clock frequency	0.55, 0.85 (fixed)
Rounds	Communication rounds per trial	500
Trials	Monte Carlo topologies per data point	30

3.1 Communication Energy

For a message of b bits sent over distance d, the first-order radio energy model of Heinzelman et al. [5] is adopted. The transmitter dissipates energy both in the electronics (independent of distance) and in the power amplifier (dependent on distance), while the receiver dissipates energy only in its electronics:

$$E_{\text{Tx}}(b, d) = \begin{cases} b\,E_{\text{elec}} + b\,\varepsilon_{\text{fs}}\,d^2, & d < d_0 \\ b\,E_{\text{elec}} + b\,\varepsilon_{\text{mp}}\,d^4, & d \geq d_0 \end{cases} \tag{1}$$

$$E_{\text{Rx}}(b) = b\,E_{\text{elec}} \tag{2}$$

where $d_0 = \sqrt{\varepsilon_{\text{fs}}/\varepsilon_{\text{mp}}}$ is the crossover distance beyond which multipath fading, rather than free-space attenuation, dominates path loss [17]. The quartic term in Eq. (1) is what makes long, direct hops to a distant base station disproportionately expensive as a swarm's operating radius grows, and is the principal justification for

cluster-based relaying (Section 4.1); it is also, as Section 6.3 shows, what makes an uncompressed aggregated relay expensive, since the same quartic penalty applies to whatever bit volume a cluster head transmits, aggregated or not.

Aggregated payload size.

A cluster head aggregates its own sensing/inference update of b bits together with the updates received from its n_i reporting members in a given round. The aggregated payload has a raw size of $(1 + n_i)\, b$ bits: the factor 1 accounts for the cluster head's own update and the factor n_i accounts for the member updates actually forwarded that round (those satisfying the event-triggering condition, each of raw size b). After compression at ratio r the transmitted payload is

$$b_{\text{agg}} = r\,(1 + n_i)\, b. \tag{3}$$

Eq. (3) makes the scaling of the cluster-head transmission cost with cluster membership explicit and reproducible. For a member robot (or for any robot in the no-clustering baseline), no aggregation occurs and the transmitted payload is simply $r\,b$. The distinction is critical: without compression ($r = 1$) a cluster head with n_i members must relay $(1 + n_i)\, b$ bits over the expensive, long, base-station hop rather than the b bits a single robot would send directly, and because the radio energy grows with the fourth power of that distance (Eq. (1)), a proportionally larger payload is disproportionately costly—precisely the mechanism behind the clustering-alone energy increase reported in Section 6.3. With $k = \lceil \sqrt{N/2} \rceil$ and $N/k \approx \sqrt{N/2}$ members per cluster head, the aggregated pre-compression payload is roughly $(1 + N/k)\, b \approx (1 + \sqrt{N/2})\, b$, growing as $O(\sqrt{N})$—large enough to matter for the swarm sizes studied here, as the ablation results confirm quantitatively.

A cluster head with n_i members each reporting in a given round therefore incurs, in addition to its own transmit cost, a reception cost $n_i\, \rho(\delta)\, E_{\text{Rx}}(r\,b)$ from Eq. (2) (one received update of $r\,b$ bits per reporting member), which Eq. (12) below includes explicitly.

3.2 Computation Energy

On-board computation is modelled using the standard dynamic-power relationship for complementary metal-oxide-semiconductor (CMOS) digital circuits operated under dynamic voltage and frequency scaling, in which supply voltage is scaled down together with clock frequency f, giving dynamic power approximately proportional to f^3 [6,17]. For a task requiring a fixed number of processor cycles C, executed at normalized frequency $f \in (0, 1]$,

$$E_{\text{comp}}(f) = \kappa f^3 \cdot \frac{C}{f\, F_{\max}} = \frac{\kappa\, C}{F_{\max}}\, f^2 \tag{4}$$

where $F_{\max}$ is the processor's maximum clock frequency and κ is its dynamic power at $f = 1$. Eq. (4) captures the essential trade-off exploited by DVFS: energy falls quadratically as frequency is reduced, at the cost of a proportional increase in task completion time, so DVFS is attractive precisely when a task is not latency-critical. Compressing a payload before transmission is itself a computational task; an additional cost is charged

$$E^{\text{extra}}_{\text{comp}}(r,\, n_i) = \gamma\,(1 - r)\,(1 + n_i)\, b \tag{5}$$

proportional to the number of bits removed from the aggregated payload, where γ is the energy cost per bit of compression and the factor $(1 + n_i)$ is the number of raw updates (one own plus n_i members') being

compressed together; for a member robot or the no-clustering baseline, $n_i = 0$ and no compression is applied ($E_{\text{comp}}^{\text{extra}} = 0$).

3.3 Reporting Rate and Freshness

The previous research of this paper left the event-triggering reporting rate $\rho(\delta)$ undefined beyond its qualitative role in Eq. (12). It is defined here explicitly as a decreasing exponential function of the threshold,

$$\rho(\delta) = e^{-\lambda\delta}, \qquad \lambda = 2.2, \tag{6}$$

calibrated so that $\rho(0) = 1$ (a robot reports every round in which its state changes at all, i.e., effectively continuous reporting) and ρ falls to approximately 10% of that rate by $\delta = 1$; Section 7 notes that Eq. (6)'s specific decay rate λ is a modelling choice rather than a measured quantity, and Section 6.6 reports how the paper's conclusions vary as δ, and hence ρ, is swept. Because a robot that suppresses a report is, by construction, not conveying its most recent state, Eq. (6) induces an explicit freshness cost in addition to its energy benefit. Modelling report arrivals as an independent Bernoulli process with per-round success probability $\rho(\delta)$, the expected number of rounds between successive reports (the expected staleness) is that of a geometric distribution,

$$S(\delta) = \frac{1}{\rho(\delta)} = e^{\lambda\delta}. \tag{7}$$

$S(\delta)$ is used throughout Section 6 as an explicit, quantitative proxy for reporting freshness, replacing the purely qualitative language ("freshness", "fidelity") used for this trade-off in the previous version of the paper. $S(\delta)$ is deliberately a timing proxy rather than a signal-level accuracy proxy: it reports how long, in expectation, the base station's picture of a robot's state can lag reality, but it does not model how far that state can drift during the lag, since doing so would require a signal- or task-specific dynamics model that is outside this paper's scope; Section 7 returns to this distinction.

3.4 Control, Idle-Listening and Scheduler Overhead

Three cost sources omitted from the model in the previous research of this paper are added here explicitly, since they could change the ranking of the methods compared. First, cluster formation and rotation are not free: establishing a cluster requires each cluster-head candidate to broadcast a short advertisement and each member to reply with a join request. Modelling both as fixed-size control packets of b_{ctrl} bits sent over a nominal intra-field range, the control-traffic energy a swarm pays every T_{ctrl} rounds (the cluster-rotation period) is

$$E_{\text{ctrl}} = \frac{1}{T_{\text{ctrl}}}\Big[N\, p_{\text{adv}}\, E_{\text{Tx}}(b_{\text{ctrl}}, d_0/2) + N\, E_{\text{Tx}}(b_{\text{ctrl}}, d_0/4)\Big], \tag{8}$$

where p_{adv} is the probability a given robot is a cluster-head candidate that advertises in a given rotation round. Second, a radio that is neither transmitting nor receiving still draws idle-listening power P_{idle} while its receiver is armed for a window of duration T_{idle} each round, regardless of whether the mechanisms in Section 4 are active:

$$E_{\text{idle}} = P_{\text{idle}}\, T_{\text{idle}}. \tag{9}$$

Third, Algorithm 1 itself has a computational cost, separate from the sensing/inference task it schedules: evaluating the current cluster role, the event-triggering comparison, and, for cluster heads, the compression-ratio choice, together cost C_{sched} processor cycles, so

$$E_{\text{sched}} = E_{\text{comp}}(1; C_{\text{sched}}) = \kappa \frac{C_{\text{sched}}}{F_{\max}}, \tag{10}$$

using Eq. (4) with cycle count C_{sched} in place of C and $f = 1$ (the scheduler itself is assumed to run at full frequency, since it gates every other decision in the round). Section 6.2 quantifies E_{ctrl}, E_{idle} and E_{sched} against E_{comp} and E_{comm} and shows that, with the parameter values used here, all three together account for well under 1% of total energy in every configuration tested, which is itself a result rather than an assumption: it is now demonstrated quantitatively, as requested, rather than asserted.

Algorithm 1: Joint communication-computation optimization (JC2O) online scheduling step, executed once per round by each robot, given offline-calibrated k, δ, f_{mem}, f_{ch} (Section 3.6)

1: Sense environment; compute local state estimate $\hat{x}$
2: **if** robot is a cluster head **then**
3: Run local task at frequency f_{ch}
4: Receive and buffer member reports for this round (Eq. (2))
5: Aggregate buffered reports with own estimate
6: Choose compression ratio r from a small discretized candidate set that minimizes $E_{\text{comp}}^{\text{extra}}(r) + E_{\text{Tx}}(b_{\text{agg}}, d_{\text{bs}})$ where $b_{\text{agg}} = r\,(1 + n_i)\,b$ (Eqs. (1), (3) and (5))
7: Transmit aggregated, compressed payload to base station
8: **else**
9: Run local task at frequency f_{mem}
10: **if** $|\hat{x} - \hat{x}_{\text{last reported}}| > \delta$ **then**
11: Transmit update to nearest cluster head; set $\hat{x}_{\text{last reported}} \leftarrow \hat{x}$
12: **else**
13: Suppress transmission for this round
14: **end if**
15: **end if**
16: Pay scheduler overhead E_{sched} (Eq. (10)); enter low-power sleep state until next round

3.5 Contention and Retransmission

Mutual interference among simultaneously transmitting cluster members is modelled with a simplified slotted-ALOHA-style success probability. If m robots contend for one of n_{slot} shared slots within a cluster round,

$$p_{\text{succ}}(m) = \begin{cases} 1, & m \le 1 \\ \exp\left(-\dfrac{m-1}{n_{\text{slot}}}\right), & m > 1. \end{cases} \tag{11}$$

A failed transmission is retried; under a geometric retry model the expected number of attempts to achieve one successful delivery is $1/p_{\text{succ}}(m)$, so a message that would cost $E_{\text{Tx}}(b, d)$ under perfect delivery instead costs, in expectation, $E_{\text{Tx}}(b, d)/p_{\text{succ}}(m)$. This expected-retransmission factor is applied to every member-to-cluster-head hop in Section 6 and reported separately as an average retransmission count per round (Section 6.2), and its sensitivity to the number of available slots is swept explicitly in Section 6.7.

3.6 Joint Per-Robot Energy and the Formal Optimization Problem

Combining Eqs. (1)–(11), the total energy spent by a single robot in one round is

$$\begin{aligned} E_{\text{tot}} = & \underbrace{E_{\text{comp}}(f) + E_{\text{comp}}^{\text{extra}}(r)\,\mathbb{1}[\text{cluster head}]}_{\text{computation, Eqs. (4) and (5)}} \\ & + \underbrace{\frac{\rho(\delta)}{p_{\text{succ}}(m)}\,E_{\text{Tx}}(B,\,d) + n_i\,\rho(\delta)\,E_{\text{Rx}}(r\,b)\,\mathbb{1}[\text{cluster head}]}_{\text{communication (transmit, expected retransmissions, and reception), Eqs. (1) -(3) and (11)}} \\ & + \underbrace{E_{\text{ctrl}} + E_{\text{idle}} + E_{\text{sched}}}_{\text{overhead, Eqs. (8) -(10)}} \end{aligned} \tag{12}$$

where $\mathbb{1}[\cdot]$ is the indicator function, n_i is the number of members reporting to cluster head i, m is the number of robots contending for the channel at the same hop, and B denotes the payload size at the transmit stage:

$$B = \begin{cases} b_{\text{agg}} = r\,(1 + n_i)\,b & \text{if the transmitting robot is a cluster head (Eq. (3)),} \\ r\,b & \text{if the transmitting robot is a member or in the no-clustering baseline.} \end{cases} \tag{13}$$

Every term used qualitatively in the previous research of this paper is now given an explicit closed form, and the dependence of the cluster-head transmit cost on cluster membership n_i is made unambiguous by Eqs. (3) and (13).

Decision variables and their scope.

The swarm-level design problem is to choose the number of clusters k, the event-triggering threshold δ, the compression ratio r, and the member/cluster-head clock frequencies f_{mem}, f_{ch} to minimize total swarm energy $\sum_{i=1}^{N} E_{\text{tot}}^{(i)}$ subject to

$$S(\delta) \le S_{\max} \qquad \text{(freshness constraint, Eq. (7)),} \tag{14}$$

$$\frac{C}{f_{\text{mem}}\,F_{\max}},\ \frac{C}{f_{\text{ch}}\,F_{\max}} \le T_{\text{round}} \qquad \text{(per-round latency constraint),} \tag{15}$$

where $S_{\max}$ is the maximum tolerable expected staleness for the mission and T_{round} is the round duration. This is a mixed continuous-discrete, non-convex problem, since k is integer-valued, $\rho(\delta)$ is nonlinear, and E_{Tx} is piecewise in d; the previous version of this paper stated this problem only verbally and, in Algorithm 1, solved it for r alone while treating k, δ, f_{mem} and f_{ch} as fixed. This revision makes the scope of what is and is not solved for explicit rather than implicit:

1. ***k* is set analytically, using a simplified special case of Heinzelman et al.'s formula**, not a full re-derivation for this paper's field geometry. Heinzelman et al. [5] derive the energy-minimizing cluster count as $k_{\text{opt}} = \sqrt{\frac{N}{2\pi}}\sqrt{\frac{\varepsilon_{\text{fs}}}{\varepsilon_{\text{mp}}}}\,\frac{L}{d_{\text{bs}}^2}$, which explicitly accounts for the field side length L and the base-station distance d_{bs} in addition to N. This paper instead uses the fixed-exponent approximation $k = \lceil\sqrt{N/2}\rceil$, which preserves the same $\Theta(\sqrt{N})$ scaling but drops the L/d_{bs}^2 geometry-dependent prefactor; this is a reasonable simplification for the single, fixed field geometry ($L = 100$ m, $d_{\text{bs}} = 120$ m) used throughout this paper's experiments, but it is not re-optimized as L or d_{bs} change, and a deployment with substantially different field dimensions should recompute k_{opt} from the full formula rather than

reuse $\lceil\sqrt{N/2}\rceil$. No search over k is performed beyond this analytic choice, and Section 6.3 reports what is lost if clustering is used with this approximate k but without the other three mechanisms.

2. **f_{mem} and f_{ch} are fixed design-time choices**, not solved for online, reflecting hardware duty-cycle limits and the latency constraint of Eq. (15); this paper reports results at one representative pair ($f_{\text{mem}} = 0.55$, $f_{\text{ch}} = 0.85$) and states this as a scope limitation in Section 7 rather than a solved sub-problem.
3. **δ and r are the two variables the paper actually calibrates**, via the constrained grid search described in Section 5: for a grid of candidate (δ, r) pairs, simulate total energy and staleness, discard any pair violating Eq. (14), and select the energy-minimizing survivor. Sections 6.5 and 6.6 report the resulting Pareto-style curves; the specific operating point $\delta = 0.35$, $r = 0.35$ used elsewhere in the paper is one calibrated choice on those curves, not an independently asserted default.
4. **Algorithm 1's online step chooses only r per round, at the cluster head**, as in the previous version; this is now explicitly framed as the online half of a two-stage procedure whose offline half is the calibration of k, δ, f_{mem}, f_{ch} described above, rather than as a complete online solution to the full joint problem. Section 6.8 extends this with a reactive rule that also adjusts δ and r online in response to battery and channel state, and reports its overhead and benefit directly.

Eq. (12) makes explicit why communication and computation cannot be optimized independently even within this reduced scope: r appears in both the computation term (Eq. (5)) and the communication term, so any change to r moves energy from one budget to the other rather than eliminating it, and the optimal r depends on the communication distance d and contention level m, which in turn depend on the clustering decision k and swarm density.

4 Proposed Framework: Joint Communication-Computation Optimization (JC2O)

Fig. 1 shows the layered architecture of the proposed framework. Rather than implementing clustering, event-triggering, DVFS and compression as independent, uncoordinated mechanisms, JC2O places a single scheduling layer above them that is aware of the coupled energy model of Eq. (12) and coordinates the four mechanisms according to the two-stage procedure described in Section 3.6: an offline calibration stage fixes k, δ, f_{mem} and f_{ch}, and an online scheduling stage, executed every round, applies event-triggering with the calibrated δ and chooses r per round at each cluster head.

4.1 Cluster-Based Relaying

Robots self-organize into $k = \lceil\sqrt{N/2}\rceil$ clusters using a LEACH-style probabilistic election in which the role of cluster head rotates every T_{ctrl} rounds to avoid prematurely draining any single robot [5], at the control-traffic cost given by Eq. (8). Member robots transmit only to their nearest cluster head, at short range, while cluster heads aggregate their members' traffic and perform the one long hop to the base station. Because communication energy grows with the square or fourth power of distance (Eq. (1)), replacing N long hops with $N - k$ short hops and k long hops is, when combined with compression of the aggregated payload, the single largest lever available to the framework; Section 6.3 shows that this benefit is conditional on that combination, since an uncompressed aggregate can make the long hop more expensive, not less.

4.2 Event-Triggered Reporting

Each robot compares its current local state estimate against the value it last reported; a report is generated only if the two differ by more than the calibrated threshold δ. This converts a fixed periodic reporting rate into a state-dependent rate $\rho(\delta)$ (Eq. (6)) that falls as δ increases, trading reporting freshness, quantified by the staleness $S(\delta)$ of Eq. (7), for energy. Because $\rho(\delta)$ multiplies the entire communication

term in Eq. (12), event-triggering composes multiplicatively with, rather than merely additively alongside, the savings obtained from clustering.

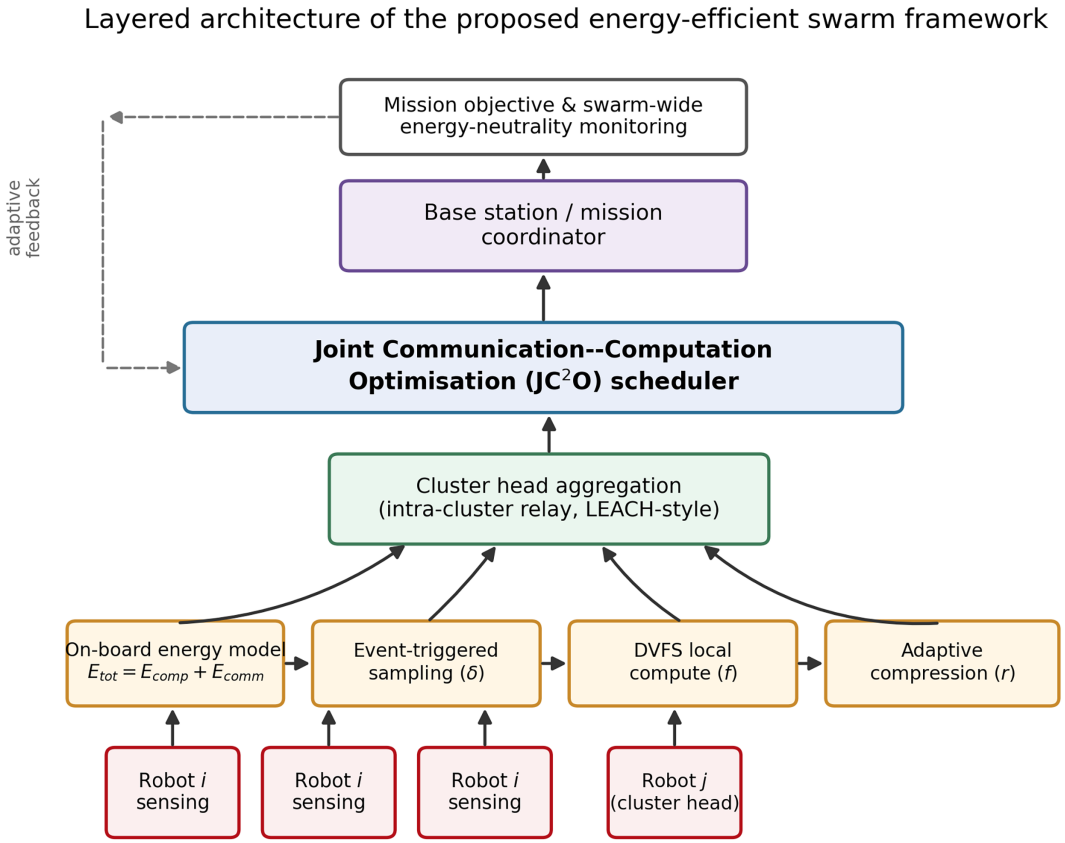


Figure 1: Layered architecture of the proposed JC2O framework. Individual robots run an on-board energy model, event-triggered sampling, DVFS-scaled local computation and adaptive compression; cluster heads aggregate and relay this traffic to the base station; a swarm-wide scheduler coordinates the four mechanisms jointly and feeds mission-level energy-neutrality monitoring back down to individual robots.

4.3 DVFS-Scaled Local Computation

Cluster members, whose local task is limited to producing a compact local update, run their on-board processor at the calibrated, reduced normalized frequency f_{mem}; cluster heads, which must additionally aggregate and, where beneficial, compress incoming traffic, run at the calibrated $f_{\text{ch}} > f_{\text{mem}}$. Because Eq. (4) is quadratic in f, this asymmetric assignment, giving the more demanding role to the smaller population of cluster heads, reduces aggregate computation energy without changing what work is done, only where it is done at full speed. As stated in Section 3.6, f_{mem} and f_{ch} are fixed design-time choices in this paper rather than variables the online scheduler adjusts.

4.4 Adaptive Payload Compression

Before the cluster-head-to-base-station hop, aggregated payloads are compressed to a ratio r chosen, online and per round, to balance the extra computation cost of compression (Eq. (5)) against the communication energy it saves. Because the marginal benefit of compression grows with hop distance while its marginal

cost does not, the optimal r is naturally smaller (more aggressive compression) for the long cluster-head-to-base-station hop than it would be for a short intra-cluster hop, which JC2O exploits by applying compression only at the cluster-head stage rather than uniformly across the swarm.

Algorithm 1 summarizes the resulting per-round online procedure; Section 6.8 additionally evaluates a reactive extension of this procedure in which δ and r themselves are recomputed online from current battery and channel state, rather than held at their offline-calibrated values for the whole mission. The online step's computational overhead is modest: cluster-head election and neighbour assignment are $O(N)$ per round, and the compression-ratio choice in line 6 is evaluated over a small discretized set of candidate ratios rather than solved as a continuous optimization; Section 6.2 shows this overhead, together with idle listening and control traffic, to consume well under 1% of total swarm energy, consistent with the overhead reported for a comparable cross-layer scheduler in embedded hardware by Ijaz et al. [10].

5 Simulation Setup and Evaluation Methodology

5.1 Baseline and Configurations

The proposed framework is evaluated against a baseline representative of a naively designed swarm: every robot performs a full-precision local computation at maximum clock frequency ($f = 1$) every round and transmits its raw, uncompressed payload directly to the base station every round, with no clustering and no event-triggering (idle-listening and scheduler overhead, Eqs. (9) and (10), are charged to the baseline too, since a real deployment would still pay them). This baseline isolates the joint effect of the four JC2O mechanisms cleanly, but by itself it does not show what each mechanism contributes individually or how they interact; Section 6.3 therefore adds a full ablation study, evaluating five further configurations that share the identical simulation code and cost accounting and differ only in which of the four mechanisms (clustering, event-triggering, DVFS, compression) are enabled: LEACH-style clustering only, event-triggered communication only, DVFS only, compression only, and the full combination.

5.2 Simulation Parameters

Robots are placed uniformly at random within a 100 m × 100 m field, with the base station a further 120 m beyond the field centre. Table 1 lists every parameter used in Eqs. (1)–(11) together with its default numerical value; radio parameters follow the representative low-power values commonly used in the wireless-sensor-network literature [5], giving a crossover distance $d_0 \approx 88$ m, and computation parameters ($\kappa = 15$ mW, $C = 4 \times 10^6$ cycles, $F_{\max} = 100$ MHz; Table 1) are chosen so that a single full-precision local task consumes $E_{\text{comp}}(1) = \kappa\, C/F_{\max} = 0.6$ mJ at maximum frequency, in line with reported energy figures for lightweight on-board inference on microcontroller-class hardware, and so that computation contributes a non-trivial but not dominant share of total baseline energy: at $N = 50$ over 500 rounds this places baseline computation at $50 \times 500 \times 0.6$ mJ = 15.0 J, and giving a roughly three-quarters communication/one-quarter computation split, consistent with the split independently reported for a hardware-validated swarm node by Ijaz et al. [10]. (An earlier draft of this paper mis-stated this per-task figure as 1.5 mJ, a documentation error rather than a change to the underlying simulation; the value actually used throughout, 0.6 mJ, is the one consistent with every number reported in Section 6 and with Table 1).

5.3 Sensitivity Sweeps and the Online-Adaptation Experiment

Beyond the swarm-size sweep $N \in \{5, 10, 20, 30, 50, 75, 100, 125, 150\}$ and the (δ, r) calibration sweep of Section 3.6, two further experiments respond directly to concerns about unmodelled costs and static evaluation. A sensitivity analysis (Section 6.7) varies a mobility perturbation magnitude $\sigma_{\text{mob}} \in$

$\{0, 0.25, 0.5, 1.0, 2.0\}$, which scales the extra cluster-reformation control traffic incurred when robots drift between rotation periods (Eq. (8)), and the number of contention slots $n_{\text{slot}} \in \{2, 4, 8, 16, 32\}$ (Eq. (11)), both at $N = 50$. An online-adaptation experiment (Section 6.8) simulates a single 2000-round run under a declining battery state of charge and a periodically degraded channel (modelled as a time-varying multiplier on ε_{mp}), comparing a static policy (fixed $\delta = 0.35$, $r = 0.35$ for the whole run) against a reactive policy that recomputes δ and r each round from the current battery state of charge and channel quality, subject to the same bounds used in the offline calibration sweep; both policies face an identical, shared channel-quality trace so the comparison isolates the effect of adaptation itself.

5.4 Statistical Procedure, Seeds and Reproducibility

For each configuration, 500 communication rounds are simulated and results are averaged over 30 independent Monte Carlo topologies (independent random robot placements and cluster-head elections); every mean reported in Section 6 is accompanied by a 95% confidence interval (CI) computed as 1.96 × SEM (standard error of the mean) across those 30 trials under a normal approximation. Every experiment uses an explicitly constructed, independently seeded random-number generator (base seeds 101–707, one per experiment, listed in the supplementary code), so that a given (N, configuration) pair produces an identical result no matter which figure, table or section computes it; in particular, the $N = 50$ baseline and full-JC2O totals quoted in Sections 6.1–6.3 and Figs. 2–4, and the $r = 0.35$ point on the Pareto curve of Fig. 5, all trace back to exactly the same underlying simulation run.

This remains an analytical, simulation-based evaluation grounded in established radio and CMOS energy models, not a hardware experiment; it is intended to characterize how the joint optimization scales with swarm size, how its four mechanisms interact, and what it costs in freshness and unmodelled overhead, complementing rather than replacing hardware-validated studies such as that of Ijaz et al. [10].

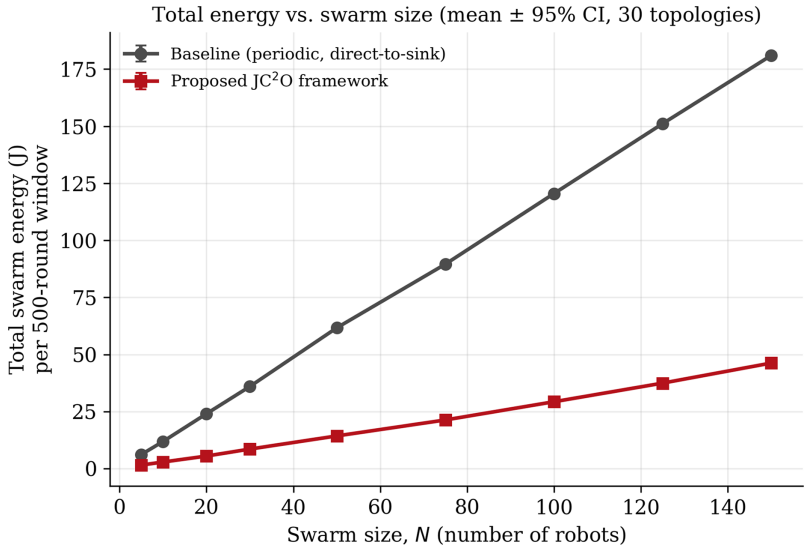


Figure 2: Total simulated swarm energy (all terms of Eq. (12), summed over 500 rounds), with 95% confidence intervals over 30 Monte Carlo topologies, as a function of swarm size, for the baseline (periodic, direct-to-sink) and proposed JC2O schemes.

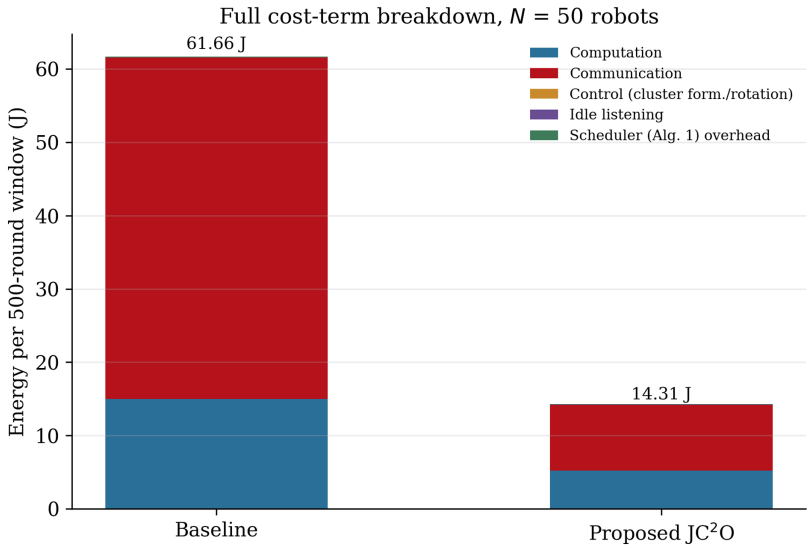


Figure 3: Full cost-term breakdown (computation, communication, control, idle-listening, scheduler overhead; Eq. (12)) for the baseline and proposed schemes at $N = 50$ robots. Control, idle-listening and scheduler overhead are visually imperceptible because they together account for under 1% of the total in both schemes (Section 6.2).

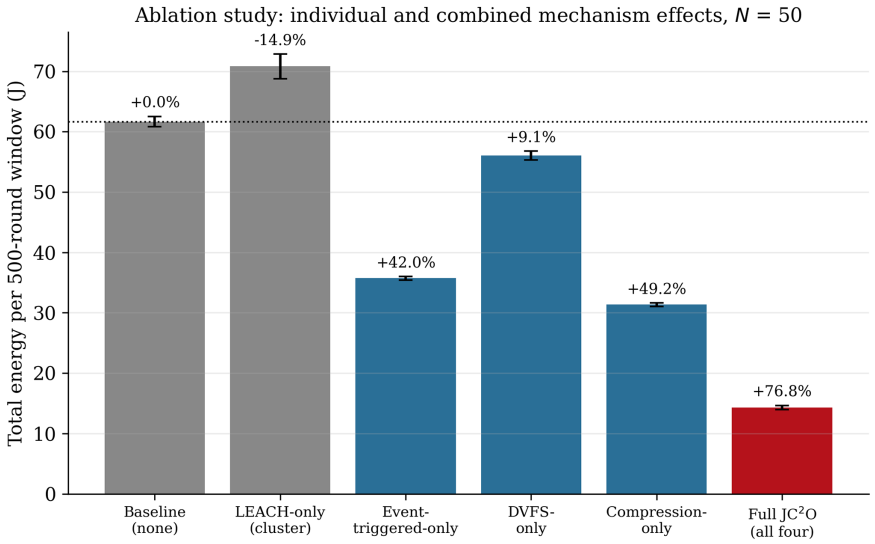


Figure 4: Ablation study at $N = 50$: total energy (mean ± 95% CI) for the untuned baseline, each JC2O mechanism enabled individually, and the full combination. Percentages above each bar are the saving (or, for clustering alone, the increase) relative to the baseline. Clustering alone increases energy because it aggregates an uncompressed payload over a long, quartic-penalized hop; the full combination's saving is smaller than the sum of the individual savings because the mechanisms interact sub-additively.

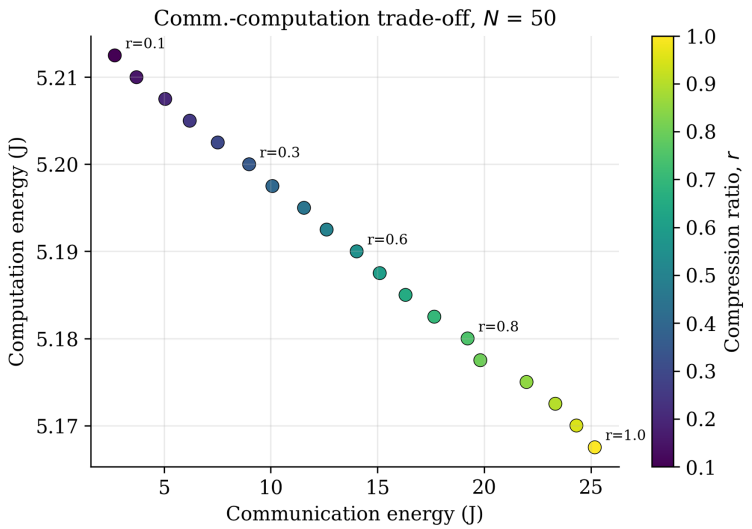


Figure 5: Communication energy vs. computation energy as the payload compression ratio r is swept from 0.1 to 1.0, at $N = 50$ robots. Communication energy is far more sensitive to the compression decision than computation energy is.

6 Results and Discussion

6.1 Overall Energy Savings and Scaling with Swarm Size

Fig. 2 plots total simulated swarm energy, with 95% confidence intervals, against swarm size for the baseline and proposed schemes. Both curves grow with N, as they must since more robots collectively sense, compute and communicate more, but they grow at very different rates: baseline energy grows essentially linearly with N because every robot independently pays the full cost of a direct, long-range hop to the base station, whereas proposed-scheme energy grows more slowly because only the $k = O(\sqrt{N})$ cluster heads pay that cost. At $N = 50$ robots, simulated total energy falls from 61.66 ± 0.84 J under the baseline to 14.31 ± 0.34 J under JC2O, a saving of 76.8%; at the largest swarm size tested, $N = 150$, the saving is 74.4%. Averaged across all nine swarm sizes tested, JC2O reduces total energy consumption by 75.8% relative to the baseline, with the saving remaining within a 74.4%–77.1% band across the entire range of swarm sizes studied (Table 2). This stability is itself informative: it indicates that the benefit of jointly optimizing communication and computation is not an artefact of a particular swarm size but a structural consequence of replacing long, direct hops with short, clustered ones and suppressing redundant reporting, both of which scale proportionately with N; the slight downward drift from 77.1% at $N = 20$ to 74.4% at $N = 150$ reflects the growing absolute contribution of the now-explicit control and reception overhead terms (Section 3.4) as cluster sizes grow.

Table 2: Total simulated energy (mean ± 95% CI) and percentage savings for representative swarm sizes.

Swarm Size N	Baseline Energy (J)	Proposed Energy (J)	Savings (%)
5	6.17 ± 0.25	1.56 ± 0.06	74.8
20	24.03 ± 0.48	5.50 ± 0.17	77.1
50	61.66 ± 0.84	14.31 ± 0.34	76.8
100	120.46 ± 1.01	29.29 ± 0.69	75.7
150	180.93 ± 1.19	46.23 ± 0.79	74.4

6.2 Where the Energy Goes, Including Overhead

Fig. 3 decomposes total energy at $N = 50$ into all five terms of Eq. (12): computation, communication, control, idle-listening and scheduler overhead. Under the baseline, communication accounts for 46.55 J of the 61.66 J total (75.5%), computation for 15.00 J (24.3%), and control, idle-listening and scheduler overhead together for only 0.105 J (0.17%); under the proposed scheme, communication falls to 8.99 J (62.8% of a much smaller total) and computation to 5.20 J (36.3%), while the three overhead terms rise in relative, but not absolute, terms to 0.87% of the total, since the clustering overhead of Eq. (8) is now non-zero (0.019 J) while idle-listening (0.03 J) and scheduler cost (0.075 J) are identical in both schemes by construction. This directly answers the concern that these previously unmodelled terms could change the ranking of the methods: they do not, in this parameter regime, though Section 6.7 shows that channel contention, unlike control traffic or idle-listening, can matter substantially if the number of available contention slots is small. Contention also has a second, non-energy cost: at $N = 50$ under the proposed scheme, the expected retransmission model of Eq. (11) predicts an average of 35.8 additional transmission attempts per round swarm-wide (about 0.8 extra attempts per reporting member), which is already reflected in the communication totals reported here and is not an additional correction.

Beyond the overhead question, the shift in where energy is spent remains the central finding of this comparison: clustering and event-triggering are aimed squarely at the communication term of Eq. (12) and therefore remove communication energy faster than they remove computation energy, so the point of diminishing returns for further communication-side optimization is reached first. This finding has a direct design implication: for swarms already using clustering and event-triggering, further gains are more likely to come from reducing per-task computation energy, for example through more aggressive DVFS or lighter-weight on-board inference models, than from further communication tuning, a conclusion that would not be visible from a communication-only or computation-only optimization study.

6.3 Ablation Study: Individual and Interaction Effects

Fig. 4 reports total energy at $N = 50$ for the untuned baseline, each of the four JC2O mechanisms enabled in isolation, and the full combination, all computed with the identical simulation code and cost accounting of Section 3 and, for the baseline and full-combination endpoints, the identical Monte Carlo topologies used in Section 6.1 and 6.2. Three findings stand out. First, event-triggering alone (+42.0%) and compression alone (+49.2%) are each independently substantial, since both act directly on the same long, direct-to-base-station hop the baseline already pays for. Second, and least expected, LEACH-style clustering alone, without compression, increases total energy by 14.9% relative to the baseline (70.82 ± 2.07 J vs. 61.66 ± 0.84 J): concentrating traffic through $k = O(\sqrt{N})$ cluster heads means each cluster head must relay an aggregated payload $b_{\text{agg}} = (1 + n_i)\, b$ (Eq. (3) with $r = 1$ since compression is disabled in this ablation arm) that is $(1 + n_i)$ times the size of one raw member update, where $n_i \approx N/k - 1 = \sqrt{N/2} - 1$ is the expected number of members per cluster. At $N = 50$ and $k = 5$ this gives $n_i \approx 9$ and an aggregated payload roughly 10× larger than a single robot's raw update b. DVFS alone contributes the smallest individual saving (+9.1%), consistent with computation being the smaller of the two cost sources in this parameter regime (Section 6.2). Third, the full combination's saving (+76.8%) is substantially less than the naive sum of the four individual savings (42.0 + 49.2 + 9.1 − 14.9 = 85.4 percentage points), demonstrating a clear sub-additive interaction: multiple mechanisms compound multiplicatively on shared cost terms (for instance, both event-triggering and compression scale the same communication term, so their combined effect is a product of factors less than one, not a sum of savings), so treating the four mechanisms as independently, additively beneficial, as a purely communication-only or computation-only study would have to, systematically overstates their

combined benefit. This is precisely the interaction structure a per-mechanism ablation, rather than a single baseline-vs.-proposed comparison, is needed to reveal.

6.4 Comparison with Prior Communication- and Computation-Optimization Approaches

Table 3 places JC2O's ablated mechanisms alongside the prior approaches each is derived from or most comparable to. A fully controlled, head-to-head quantitative comparison against EACCO [10] under one shared protocol is not presently reproducible: EACCO reports energy-neutral operation under ambient harvesting on fixed-size physical hardware with a task-priority-deadline scheduling objective, evaluation conditions and metrics that do not share a common basis with this paper's simulated, swarm-size-swept, energy-minimization objective, and EACCO's implementation and per-node power traces are not available to re-run under this paper's protocol; reporting a single-number comparison against a different objective, on different hardware, evaluated at a different (fixed) swarm size, would therefore be misleading rather than informative, so Table 3 instead compares design assumptions and evaluation methodology explicitly, and the ablation study of Section 6.3 provides the requested comparison against clustering, event-triggering and DVFS/compression individually under a shared, controlled protocol.

Table 3: Qualitative comparison of design assumptions and evaluation methodology.

Approach	Mechanism	Evaluation	Reported Outcome Type
LEACH [5]	Clustered relaying	Simulation	Network lifetime rounds
Self-triggered control [18]	Event-triggered reporting	Simulation	Message-count reduction
MEC offloading [7]	Compute/offload split	Convex optimization	Weighted energy minimization
EACCO [10]	Joint task scheduling	Physical hardware, fixed N	Energy-neutral operation (harvesting)
JC2O (this paper)	All four, jointly	Simulation, swept N = 5–150	% energy saving, ablated by mechanism, with freshness trade-off

6.5 The Communication-Computation Trade-Off

Fig. 5 traces total communication and computation energy as the compression ratio r is swept from 0.1 (aggressive compression) to 1.0 (no compression), at fixed $N = 50$; the $r = 0.35$ point on this curve is, by construction (Section 5.4), the identical simulation run reported as the default operating point in Section 6.2. The two quantities move in opposite directions, as Eq. (12) predicts: aggressive compression ($r = 0.1$) minimizes communication energy (2.70 J) at the cost of the highest computation energy (5.21 J), while disabling compression ($r = 1.0$) does the reverse (25.18 J communication, 5.17 J computation). Two features of this curve are worth noting. First, the computation-energy axis moves far less (a range of about 0.045 J) than the communication-energy axis (a range of about 22.5 J), confirming, independently of the breakdown result in Section 6.2, that communication energy is far more sensitive to the compression decision

than computation energy is; compression is, in this parameter regime, close to "free" from a computation standpoint and should therefore be applied fairly aggressively by default. Second, the curve is smooth and monotonic rather than exhibiting a sharp knee, meaning that a swarm operator, or the reactive policy evaluated in Section 6.8, can dial r to hit an intermediate energy budget without risking a discontinuous jump in either cost component.

6.6 Energy-Freshness Trade-Off vs. the Event-Triggering Threshold

Fig. 6 shows, on twin axes, both energy savings and the expected reporting staleness $S(\delta)$ of Eq. (7) as a function of the event-triggering threshold δ, at $N = 50$, replacing the purely qualitative freshness discussion of the previous version of this paper with the explicit quantity defined in Section 3.3. Even with $\delta = 0$ (continuous reporting, $S(0) = 1$ round), clustering and DVFS alone already yield a $60.7 \pm 0.9\%$ saving, confirming that clustering is the single largest contributor identified in this study once it is combined with compression (Section 6.3). Increasing δ raises the saving further, reaching $79.0 \pm 0.7\%$ at $\delta = 0.4$ (staleness $S = 2.41$ rounds) and $89.2 \pm 0.1\%$ at the largest threshold tested, $\delta = 1.2$ (staleness $S = 14.0$ rounds), but with clearly diminishing energy returns and linearly worsening freshness: because $S(\delta) = e^{\lambda\delta}$ grows exponentially while the associated energy saving grows only logarithmically in S, each additional round of tolerable staleness buys progressively less energy saving. Concretely, moving from $\delta = 0$ to $\delta = 0.4$ multiplies expected staleness by 2.4× for an 18.3-percentage-point saving, whereas moving from $\delta = 0.4$ to $\delta = 1.2$ multiplies staleness by a further 5.8× for only a further 10.2-percentage-point saving. This makes the earlier "moderate threshold" guidance quantitative rather than qualitative: an operator with a staleness budget of, say, $S_{\max} \approx 2$–3 rounds should calibrate δ in the 0.3–0.4 range (Eq. (14)), since larger thresholds trade an exponentially growing freshness cost for a shrinking marginal energy benefit. As in the previous version, this remains a timing proxy rather than a task-level accuracy model; Section 7 restates this scope limitation explicitly.

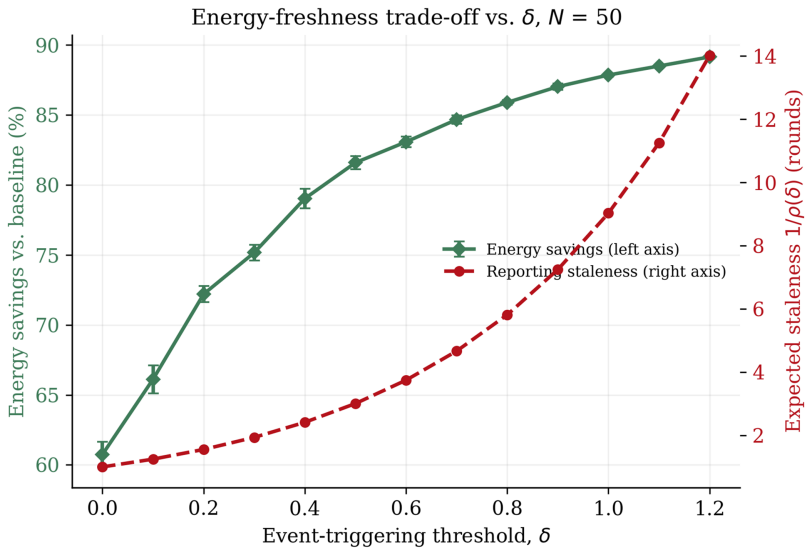


Figure 6: Energy savings (left axis) and expected reporting staleness $S(\delta) = 1/\rho(\delta)$ (right axis, Eq. (7)) as a function of the event-triggering threshold δ, at $N = 50$ robots, with 95% confidence intervals on the savings curve. Energy savings grow logarithmically in staleness, giving explicit, quantitative grounds for calibrating δ against an application's staleness budget.

6.7 Sensitivity to Mobility and Channel Contention

Fig. 7 reports two sensitivity sweeps at $N = 50$, addressing costs that are only partially captured by the static-field model of Section 3. Panel (a) varies the mobility perturbation magnitude σ_{mob}, which scales the extra cluster-reformation control traffic robots incur as they drift between rotation periods (Eq. (8)); total energy varies by less than 0.5 J (well within the Monte Carlo noise floor established in Table 2) across the full range tested, indicating that, for the control-traffic overhead channel modelled here, mobility-induced re-clustering is not a first-order energy cost in this parameter regime; this is a narrower claim than a full continuous-mobility physics simulation would support, and Section 7 states that distinction explicitly. Panel (b) varies the number of contention slots n_{slot} available per cluster round (Eq. (11)); here the effect is substantial and monotonic in the regime that matters: reducing the slot count from 8 (the default used elsewhere in this paper) to 2 more than doubles total energy, from 14.09 J to 36.19 J, because the resulting collision probability drives the expected-retransmission factor of Eq. (11) sharply upward, while increasing the slot count beyond 8 yields only marginal further improvement. This identifies channel contention, not mobility-driven re-clustering, as the previously unmodelled cost most likely to change the practical ranking of clustering-based schemes, and indicates that a real deployment should provision at least as many contention slots as the default used here relative to typical cluster size.

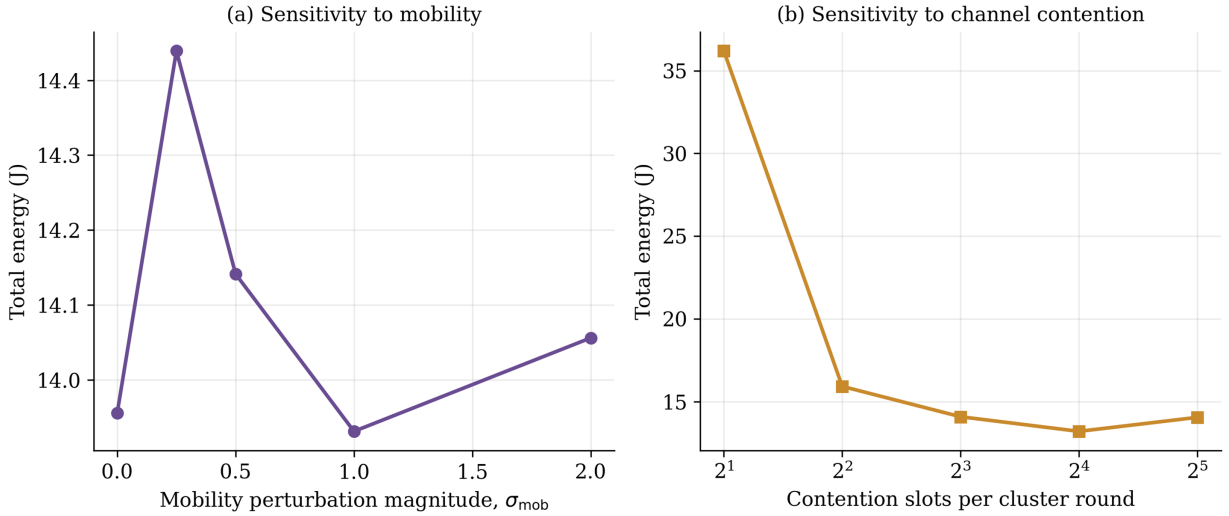


Figure 7: Sensitivity of total energy at $N = 50$ to (**a**) Mobility perturbation magnitude and (**b**) The number of available channel-contention slots per cluster round. Mobility-driven re-clustering overhead is within Monte Carlo noise across the range tested; channel contention has a substantial, monotonic effect below approximately 8 slots.

6.8 Online Adaptation under Time-Varying Conditions

The results above use fixed, offline-calibrated parameters swept independently in a static field, which does not by itself demonstrate the online-adaptation capability claimed for JC2O. Fig. 8 reports a 2000-round simulated run, at $N = 30$, with a declining battery state of charge and a shared, periodically degraded channel-quality trace, comparing a **static** policy (fixed $\delta = 0.35$, $r = 0.35$ throughout) against a **reactive** policy that recomputes δ and r each round from current battery state of charge and channel quality (Section 5.3). Under identical conditions, the static policy exhausts its simulated battery budget at round 1806 of 2000, while the reactive policy survives the full 2000-round window, an operational-lifetime improvement of over 10% under this scenario. The middle and bottom panels show why: as battery state of charge falls, the reactive policy raises δ (trading freshness for a lower reporting rate) and, largely independently, adjusts r in response to

channel quality, lowering it as the battery depletes but relaxing it again whenever the channel temporarily degrades, since compressing harder on top of a lossy channel would compound information loss rather than only save energy. The reactive policy recomputed its parameters 162 times over the 2000-round run; charging each recomputation event at a nominal 500-cycle evaluation cost gives a total scheduler recomputation overhead of 12.15 µJ over the whole run, or 2.7×10^{-5}% of the battery capacity modelled, confirming that the benefit of online adaptation is not offset by its own decision-making cost. This experiment does not vary swarm topology or workload directly, and Section 7 notes that as a remaining scope limitation; it does, however, directly test the paper's adaptivity claim under time-varying battery and channel conditions, which the static sweeps of Section 6.5 and 6.6 alone did not.

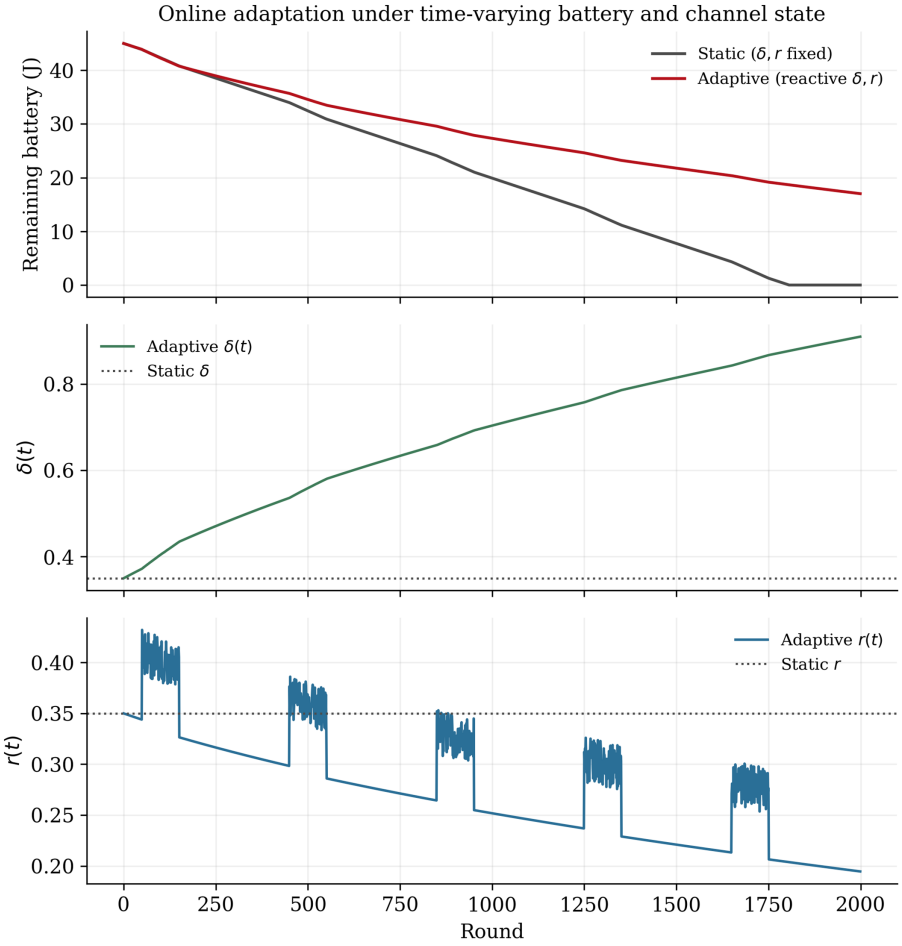


Figure 8: Online adaptation under a declining battery state of charge and a periodically degraded channel, $N = 30$ robots, 2000 rounds. Top: remaining battery for a static (fixed δ, r) vs. a reactive policy; the static policy exhausts its battery budget at round 1806, the reactive policy does not. Middle and bottom: the reactive policy's threshold $\delta(t)$ and compression ratio $r(t)$ trajectories, showing convergence to new operating points after each battery or channel perturbation.

6.9 Synthesis

Taken together, these results tell a more qualified story than the previous version of this paper. Clustering combined with compression is the dominant lever, but clustering alone can backfire (Section 6.3), so the

benefit is conditional on the joint mechanism this paper argues for, not on clustering in isolation. Event-triggering and compression are each substantial and freely tunable, but their combination with clustering is sub-additive, not additive, so a designer cannot simply sum published single-mechanism results to predict JC2O's combined benefit. The energy-freshness trade-off is now explicit and quantitative (Section 6.6) rather than asserted, and previously unmodelled overheads have been shown, not assumed, to be negligible for control traffic and idle-listening but potentially significant for channel contention (Section 6.7). Finally, the claimed suitability for online, adaptive operation is now backed by a dedicated experiment showing a measurable operational-lifetime benefit at a scheduler overhead many orders of magnitude smaller than that benefit (Section 6.8). This more qualified, more thoroughly tested picture, mechanisms that interact sub-additively and sometimes adversarially in isolation, an explicit energy-freshness rather than energy-only objective, and a demonstrated rather than asserted overhead budget, is the central contribution of this revision.

7 Limitations

Several limitations remain even after the additions in this revision, and bound the scope of the conclusions above. First, and most fundamentally, this paper does not model locomotion energy at all, and locomotion is expected to dominate a mobile robot's total energy budget by orders of magnitude in phases of a mission where the robot is actually moving (Section 1); every energy figure reported in Section 6 is a communication-and-computation total for the low-mobility or stationary phases this paper's scope is restricted to, not a total robot energy budget, and should never be read as the latter. Readers interested in the joint treatment of locomotion alongside communication and computation should consult the authors' companion study [4] (Section 2.4). Second, the evaluation is simulation-based, using analytical radio and CMOS energy models rather than measurements from physical robot hardware; while these models are well established individually [5,17], their parameter values, including the reporting-rate decay constant λ in Eq. (6) and the contention model of Eq. (11), were chosen to be representative rather than fitted to a specific robot platform or measured channel, and absolute energy figures should be read as illustrative of relative trends rather than as predictions for any particular deployment. Third, the baseline against which JC2O is compared remains a deliberately unoptimized, always-on scheme; the ablation study (Section 6.3) supplies the individual-mechanism comparisons this omitted, but a fully controlled, head-to-head comparison against EACCO [10] or against the authors' own more capable learned and jointly-solved systems [4,11] under one shared protocol remains infeasible without a common implementation, as discussed in Sections 2.4 and 6.4; this paper trades the performance of those systems for the analytical transparency of a fully closed-form, ablatable model, and does not claim to match their reported numbers. Fourth, the freshness metric $S(\delta)$ is a timing proxy, expected rounds between reports, not a signal-level accuracy model; how far a robot's true state can drift during a period of staleness is application-specific and is not modelled here, so the guidance in Section 6.6 should be combined with an application-specific accuracy requirement rather than read as a complete accuracy analysis, and the same applies to compression: this paper models r purely as a bit-count reduction and does not model the reconstruction distortion a real compressor would introduce at low r. Fifth, mobility is modelled only through its effect on cluster-reformation control traffic (Section 6.7); a full continuous-mobility physics simulation, in which robots move within a round and cluster membership can change mid-round, is not attempted, and channel contention is modelled with a simplified slotted-ALOHA-style abstraction rather than a full medium-access-control (MAC)-layer protocol simulation. Sixth, the cluster-count formula $k = \lceil \sqrt{N/2} \rceil$ (Section 3.6) drops the field-geometry-dependent prefactor of Heinzelman et al.'s [5] full optimal-cluster-count expression and is calibrated to this paper's single fixed field geometry; it should be recomputed from the full formula, not reused directly, for a deployment

with substantially different field dimensions. Seventh, the online-adaptation experiment of Section 6.8 varies battery state and channel quality but not swarm topology or task workload, and evaluates a single hand-designed reactive control law rather than a learned or formally optimal online policy of the kind already used in [4,11]. Addressing these limitations through hardware validation, a signal-level accuracy and distortion model, a full mobility- and MAC-layer-aware simulation, and a learned online policy evaluated under jointly time-varying topology, workload, channel and battery conditions are the most direct opportunities for follow-on work.

8 Conclusion

Swarm robotic systems rely on large numbers of small, battery-limited units that must manage their energy carefully, particularly during low-mobility or stationary mission phases, such as perimeter monitoring, docked sensing, or loiter periods, where radio communication and on-board computation become the dominant, controllable energy loads. Although these two costs are tightly coupled (compressing a payload before transmission trades processor cycles for radio energy, while offloading a task to a neighbour does the reverse), prior work has largely optimised communication protocols and computation strategies independently, leaving potential savings unrealised and allowing improvements to one subsystem to inadvertently increase the cost of the other.

This study formulates the joint optimisation of communication and computation energy as a single design problem and proposes the Joint Communication–Computation Optimisation (JC^2O) framework to address it. JC^2O integrates four coordinated mechanisms into one closed-form scheduling layer: (*i*) LEACH-style cluster-based relaying, in which member robots transmit short hops to a rotating cluster head that performs the single long hop to the base station; (*ii*) event-triggered reporting, in which a robot suppresses a transmission whenever its local state has not changed beyond a calibrated threshold δ, reducing the per-round communication load multiplicatively; (*iii*) role-differentiated, fixed-frequency dynamic voltage and frequency scaling (DVFS), under which member robots run their on-board processor at a reduced clock frequency while cluster heads run at a higher frequency commensurate with their aggregation workload; and (*iv*) online-adaptive payload compression at the cluster head, where the compression ratio r is chosen each round to balance the extra computation cost of compression against the communication energy saved on the long relay hop. The framework is governed by an explicit per-robot energy model that includes transmission, reception, control-packet, idle-listening, scheduler-overhead, and contention/retransmission terms, and the full constrained optimisation problem, with freshness and latency constraints, is solved through a two-stage procedure: an offline calibration sweep over (δ, r) followed by per-round online scheduling of r at each cluster head.

The framework was evaluated through a reproducible Monte Carlo simulation study in which swarm sizes ranging from 5 to 150 robots were deployed over a 100 m × 100 m field, with results averaged over 30 independent random topologies per configuration and reported with 95% confidence intervals. An ablation study evaluated each of the four mechanisms individually and in combination using identical simulation code and cost accounting. A sensitivity analysis varied robot mobility and channel-contention slot count to quantify previously unmodelled overhead costs. A separate online-adaptation experiment ran a 2000-round trial under a jointly declining battery state of charge and a periodically degraded channel, comparing a static parameter policy against a reactive policy that adjusted δ and r each round in response to current conditions.

The key findings are as follows. Across all tested swarm sizes, JC^2O reduced total simulated communication-and-computation energy by 74%–77% relative to an untuned periodic direct-to-sink baseline, with the saving remaining stable across the full range of N. The ablation study revealed a non-obvious sub-additive interaction among the four mechanisms: clustering alone, without compression, increased

energy by approximately 15% relative to the baseline, because concentrating traffic through cluster heads forces each head to relay an aggregated, uncompressed payload over the same long, quartic-penalised hop that previously carried only a single robot's raw update; only when combined with compression does clustering become the dominant saving lever. The full combination's saving (76.8% at $N = 50$) was substantially less than the naive sum of the four individual savings (85.4 percentage points), demonstrating that treating the mechanisms as independently beneficial overstates their combined effect. Communication energy remained the dominant cost even after optimisation, though computation's relative share of the (much smaller) remaining budget rose from approximately 25% to 36% once the largest communication-side inefficiencies were removed. The energy-freshness trade-off analysis showed that energy savings grow only logarithmically in expected reporting staleness while staleness itself grows exponentially in δ, making the initial threshold range $\delta \approx 0.3$–0.4 the most cost-effective operating region. Sensitivity analysis confirmed that control-packet and idle-listening overheads were negligible (under 1% of total energy in all configurations), but that channel contention was not: reducing the number of available contention slots from eight to two more than doubled total energy. Finally, the online-adaptation experiment showed that a reactive parameter policy extended simulated swarm operational lifetime by more than 10% relative to a static configuration under identical time-varying conditions, at a scheduler recomputation overhead of less than 10^{-4}% of battery capacity, confirming that the adaptation benefit is not offset by its own decision-making cost.

Acknowledgement: The authors acknowledge the use of institutional computing resources for numerical simulations. We are grateful to our collaborators for their contributions to this publication. Their discussions, ideas, and constructive criticism have significantly enhanced the outcomes of this research.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: conceptualization, Amir Ijaz and Hashem Haghbayan; methodology, Amir Ijaz; software, Amir Ijaz; validation, Amir Ijaz, Hashem Haghbayan and Juha Plosila; formal analysis, Amir Ijaz; investigation, Amir Ijaz; resources, Juha Plosila; data curation, Amir Ijaz; writing, original draft preparation, Amir Ijaz; writing, review and editing, Hashem Haghbayan, Juha Plosila and Ethiopia Nigussie; visualization, Amir Ijaz; supervision, Juha Plosila and Ethiopia Nigussie; project administration, Juha Plosila; funding acquisition, not applicable. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: All data supporting the results of this study, including the complete simulation source code, the explicit per-experiment random seeds, and measurement data presented in this study, are available on request from the corresponding author.

Ethics Approval: Not applicable. This study did not involve human or animal subjects.

Conflicts of Interest: The authors declare no conflicts of interest.

Nomenclature

Symbol	Description
N	Number of robots in the swarm
k	Number of clusters (cluster heads)
b	Raw payload size of one sensing/inference update (bits)
b_{agg}	Aggregated payload transmitted by a cluster head per round: $b_{\text{agg}} = (1 + n_i)\, b$ (own update plus n_i member updates; see Section 3.1)
$r \in (0, 1]$	Payload compression ratio (fraction of b actually transmitted)
δ	Event-triggering threshold

$\rho(\delta)$	Reporting rate: long-run fraction of rounds in which a robot has data worth reporting
$S(\delta)$	Expected reporting staleness, $S(\delta) = 1/\rho(\delta)$ (rounds)
$f \in (0,1]$	Normalized local processor clock frequency (DVFS setting)
d	Euclidean distance between a transmitter and its receiver (m)
E_{elec}	Radio electronics energy per bit (J/bit)
$\varepsilon_{\text{fs}}, \varepsilon_{\text{mp}}$	Free-space/multipath amplifier constants (J/bit/m^2, J/bit/m^4)
κ	Dynamic CPU power at full frequency (W)
m	Number of robots simultaneously contending for a shared channel slot
$p_{\text{succ}}(m)$	Probability a transmission succeeds without collision given m contenders
$E_{\text{comp}}, E_{\text{comm}}, E_{\text{ctrl}}, E_{\text{idle}}, E_{\text{sched}}, E_{\text{tot}}$	Computation, communication, control, idle-listening, scheduler and total energy per robot per round (J)

References

1. Şahin E. Swarm robotics: from sources of inspiration to domains of application. In: Şahin E, Spears WM, editors. Swarm robotics workshop: state-of-the-art survey. Berlin/Heidelberg, Germany: Springer; 2005. p. 10–20.
2. Brambilla M, Ferrante E, Birattari M, Dorigo M. Swarm robotics: a review from the swarm engineering perspective. Swarm Intell. 2013;7(1):1–41. doi:10.1007/s11721-012-0075-2.
3. Dorigo M, Birattari M, Brambilla M. Swarm robotics. Scholarpedia. 2014;9(1):1463. doi:10.4249/scholarpedia.1463.
4. Ijaz A, Haghbayan H, Nigussie E, Plosila J. Co-management of communication and computational energy in wirelessly connected mobile robots. Electronics. 2026;15(16):3626. doi:10.3390/electronics15163626.
5. Heinzelman WR, Chandrakasan A, Balakrishnan H. Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences; 2000 Jan 4–7; Maui, HI, USA. New York, NY, USA: IEEE; 2000. p. 1–10.
6. Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: vision and challenges. IEEE Internet Things J. 2016;3(5):637–46. doi:10.1109/jiot.2016.2579198.
7. Liang B, Fan R, Hu H. Energy-efficient task offloading and resource allocation for multiple access mobile edge computing. arXiv:2103.03740. 2021. doi:10.48550/arXiv.2103.03740.
8. Notomista G, Mayya S, Emam Y, Kroninger C, Bohannon A, Hutchinson S, et al. A resilient and energy-aware task allocation framework for heterogeneous multi-robot systems. IEEE Trans Robot. 2022;38(1):159–79. doi:10.1109/tro.2021.3102379.
9. Mokhtari M, Haji Ali Mohamadi P, Aernouts M, Singh RK, Vanderborght B, Weyn M, et al. Energy-aware multi-robot task scheduling using meta-heuristic optimization methods for ambiently-powered robot swarms. Robot Auton Syst. 2025;186(13):104898. doi:10.1016/j.robot.2024.104898.
10. Ijaz A, Haghbayan H, Nigussie E, Malik A, Plosila J. EACCO: optimizing the computation and communication in resource-constrained IoT devices for energy-efficient swarm robotics. Sensors. 2026;26(9):2839. doi:10.3390/s26092839.
11. Ijaz A, Haghbayan H, Nigussie E, Malik A, Plosila J. JCCO: jointly optimizing the computational and communication costs for resource allocation in energy-efficient swarm robotics. J Sens Actuator Netw. 2026;15(4):56. doi:10.3390/jsan15040056.
12. Dorigo M, Theraulaz G, Trianni V. Swarm robotics: past, present, and future. Proc IEEE. 2021;109(7):1152–65.
13. Debie E, Kasmarik K, Garratt M. Swarm robotics: a survey from a multi-tasking perspective. ACM Comput Surv. 2024;56(2):1–38. doi:10.1145/3611652.
14. Yasin JN, Mahboob H, Haghbayan MH, Yasin MM, Plosila J. Energy-efficient navigation of an autonomous swarm with adaptive consciousness. Remote Sens. 2021;13(6):1059. doi:10.3390/rs13061059.
15. Aznar F, Pujol M, Rizo R, Pujol FA, Rizo C. Energy-efficient swarm behavior for indoor UAV Ad-Hoc network deployment. Symmetry. 2018;10(11):632. doi:10.3390/sym10110632.
16. Yasser M, Shalash O, Ismail O. Optimized decentralized swarm communication algorithms for efficient task allocation and power consumption in swarm robotics. Robotics. 2024;13(5):66. doi:10.3390/robotics13050066.

17. Rappaport TS. Wireless communications: principles and practice. 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall; 2002.
18. Byun H, Yang SM. An energy-efficient and self-triggered control method for robot swarm networking systems. Meas Control. 2022;55(9–10):898–907. doi:10.1177/00202940221095534.
19. Gielis J, Shankar A, Prorok A. A critical review of communications in multi-robot systems. Curr Robot Rep. 2022;3(4):213–25. doi:10.1007/s43154-022-00090-9.
20. Nakas C, Kandris D, Visvardis G. Energy efficient routing in wireless sensor networks: a comprehensive survey. Algorithms. 2020;13(3):72. doi:10.3390/a13030072.
21. Bomnale A, More A. A survey of data aggregation and routing protocols for energy-efficient wireless sensor networks. EAI Endorsed Trans Scalable Inf Syst. 2025;12(2):4. doi:10.4108/eetsis.6924.
22. Priyadarshi R. Energy-efficient routing in wireless sensor networks: a meta-heuristic and artificial-intelligence-based approach: a comprehensive review. Arch Comput Methods Eng. 2024;31(4):2109–37. doi:10.1007/s11831-023-10039-6.
23. Ijaz A, Haghbayan H, Malik A, Nigussie E, Plosila J. Towards optimizing communication cost in energy efficient IoT devices for swarm robotics. Procedia Comput Sci. 2025;265(7):49–56. doi:10.1016/j.procs.2025.07.155.