Available Online: 10 October 2026 DOI: 10.54254/2977-3903/2026.37548

Energy-optimal 3D UAV trajectory planning and multi-ant-colony cooperative coverage path planning

Yi Xu

School of Mathematics and Statistics, Changsha University of Science and Technology, Changsha, China

1316004935@qq.com

Abstract. Energy consumption, rather than path length, is the binding constraint for drones operating over dense urban terrain. This paper treats flight energy as the sum of three physically distinct terms: horizontal cruise, vertical climb, and attitude adjustment during turns. Three simplifying assumptions—constant cruise speed, negligible wind, and constant angular velocity in maneuvers—delimit where the model holds. On that foundation the paper develops an ant colony optimization variant whose heuristic scores the reciprocal of segment energy rather than distance, with Min-Max normalization across the factors and pheromone kept within bounds. Collision checking samples the segment interior at 5m intervals instead of testing endpoints only. For area coverage, the planning problem is split into a macro layer that sequences sub-regions and a micro layer that designs the scan pattern inside each one; transition costs between sub-regions are computed offline once and stored in a lookup table. Tests over Shenzhen Longhua District, partitioned into 30 sub-regions, cut total flight energy by 18.8 percent against baseline Ant Colony Optimization (ACO) while finishing 60.5 percent faster.

Keywords: UAV trajectory planning, energy-aware planning, ant colony optimization, coverage path planning, 3D obstacle avoidance

1. Introduction

Battery life, not path geometry, usually decides what a small multi-rotor can do over a city—whether the mission is environmental monitoring, disaster response, or urban logistics. Every extra climb, every unnecessary turn, and every detour around a high-rise shows up in the energy ledger well before the flight ends, yet most coverage planners still price routes by distance alone. This gap motivated the work described here: an energy-accounting trajectory planner for drones that must inspect a patchwork of urban sub-regions while staying clear of terrain and buildings.

The starting point is the vehicle drone framework of Xiang, but three of the choices made there are reconsidered here: the physical assumptions behind a constant energy coefficient—steady cruise speed, still air, and constant rate turns—are written out rather than implied, because each one marks a place where the model stops holding. Collision checks look beyond the waypoints themselves: the segment interior is sampled every 5 m, which closes the gap that lets a path clip through a wall between two seemingly clear endpoints.

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

And in the implementation, angle constraints stay in degrees for readability but are converted with deg2rad() before any trigonometric call, since mixing the two units silently corrupts the yaw and pitch limits.

The path-planning literature offers several families of solvers—graph search such as A*, and its descendants, sampling-based planners like RRT and PRM, and population methods such as genetic algorithms and particle swarms [1]. ACO [2] sits in the last family and has been a common choice on 3D grids, mostly because it tolerates rough, discontinuous search spaces without much tuning. The coverage side of the problem has its own tradition of decomposition-based methods [3, 4], though nearly all of them bill cost by distance traveled. In three dimensions that approximation is quietly wrong: climbing and turning dominate the energy budget over undulating ground with tall buildings, so a distance-optimal scan pattern is often not the energy-optimal one.

Against that backdrop, this paper makes four specific additions. The energy model itself splits total consumption into horizontal, vertical, and turning terms, so the planner no longer behaves as if energy were a rescaled distance. The ACO heuristics take the reciprocal of segment energy instead of segment length, with factors normalized into the same range and pheromone clamped on both ends. Coverage becomes a two-level problem—which sub-regions to visit in what order, and how to sweep each one—with the transition costs computed once offline rather than re-planned at every iteration. All four pieces run together on real Longhua terrain and building data in the case study.

2. Energy-optimal 3D trajectory planning

2.1. Physical assumptions and energy model

The model rests on three assumptions, each of which sets a boundary on where the results apply.

(A1) Constant cruise speed: the energy per horizontal meter stays fixed because the aircraft flies at its minimum-power velocity. The underlying aerodynamic relationship is:

$$e_h(v) = \frac{1}{2}\rho \cdot C_{D0} \cdot A \cdot v^2 + \frac{(2W^2)}{(\rho \cdot A \cdot v^2)} \tag{1}$$

where ρ is air density, C_{D0} is the zero-lift drag coefficient, A is the rotor disk area, and W is the UAV weight.
(A2) No wind disturbance: ground speed equals airspeed, with wind correction:

$$e'_h = e_h \cdot (v_{ground}/v_{air}) \tag{2}$$

(A3) Constant angular velocity turns: attitude energy is linearly proportional to angular change magnitude:

$$E_{atti} = k_\theta \cdot |\Delta\theta| + k_\varphi \cdot |\Delta\varphi| \tag{3}$$

All MATLAB trigonometric computations convert degree-based constraints via deg2rad() to radians; the energy coefficients carry units of J/rad.

The total flight energy of a UAV on a trajectory with n segments consists of three components:

$$E_{total} = \sum_{i=1}^{n}(E_i^h + E_i^v + E_i^a) \tag{4}$$

The three components are defined as follows. Horizontal cruise energy, where d_i^h is the horizontal projection distance of segment L_i :

$$E_i^h = e_h \cdot d_i^h \tag{5}$$

Vertical climb energy, accounting only for climbing since descent energy is negligible:

$$E_i^v = e_v^+ \cdot max(0, z_i - z_{i-1}) \tag{6}$$

Attitude adjustment energy, where θ_i is the yaw angle between consecutive segments projected on the xy-plane and φ_i is the pitch angle:

$$E_i^a = k_\theta \cdot |\theta_i| + k_\varphi \cdot |\varphi_i| \tag{7}$$

2.2. Performance constraints

Five constraints bound the flight itself. (1) Maximum range and (2) minimum segment:

$$\sum_{i=1}^{n} L_i \le L_{max}, \quad L_i > L_{min} \tag{8}$$

(3) Maximum yaw angle Θ = 60 degrees. The yaw angle is computed from projection vectors $BA' = (x_i - x_{i-1}, y_i - y_{i-1})$ and $A'C = (x_{i+1} - x_i, y_{i+1} - y_i)$:

$$\theta_i = arccos[(BA\prime \cdot A\prime C)/(\|BA\prime\| \cdot \|A\prime C\|)] \le \Theta \tag{9}$$

(4) Maximum pitch angle Φ = 60 degrees:

$$\varphi_i = \arctan\left(\frac{|z_{i+1} - z_i|}{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}}\right) \le \Phi \tag{10}$$

(5) Height bounds:

$$H_{min} \le z_i \le H_{max} \tag{11}$$

2.3. Collision avoidance and flight space

The environmental collision constraint requires the flight altitude to exceed the combined terrain-building surface height plus a safety margin w at every grid point surrounding each waypoint:

$$W(x_i, y_i) > F(x_i + \varepsilon_1 \cdot w, y_i + \varepsilon_2 \cdot w), \varepsilon_1, \varepsilon_2 \in \{-1, 0, 1\} \tag{12}$$

In the code, the checkLOS() function walks each segment at 5 m steps and verifies that no intermediate point sinks below a building taller than H_{max}. Endpoint checks alone miss exactly this failure: two waypoints can both be clear while the straight line between them passes through a wall.

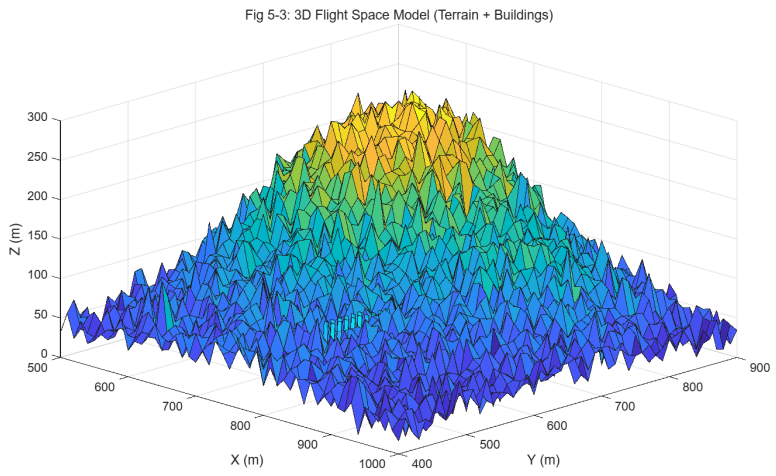


Figure 1. 3D flight space model of longhua district (terrain + buildings overlay)

As shown in Figure 1, the three-dimensional flight space of Longhua District is modeled by superimposing DEM terrain and building height data:

$$F_{ij}(x_i, y_j) = D_{ij}(x_i, y_j) + J_{ij}(x_i, y_j) \tag{13}$$

As shown in Figure 2, the planned three-dimensional trajectory navigates through the modeled flight space while avoiding obstacles.

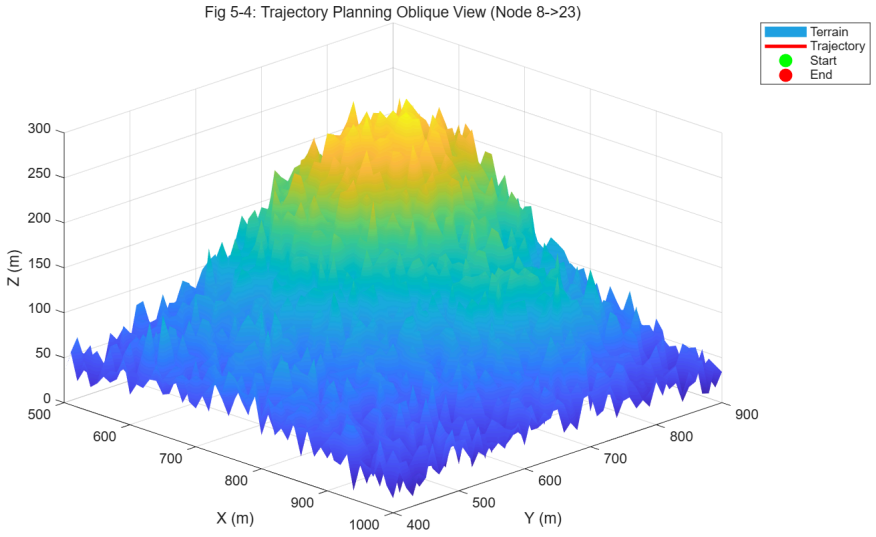


Figure 2. Energy-optimal 3D trajectory with obstacle avoidance (oblique view)

3. Improved ACO and MACO-C algorithm

3.1. Energy-reciprocal ACO for point-to-point planning

The first change to the baseline ACO is in the heuristic itself. Instead of the usual inverse distance, each candidate segment is scored as the inverse of its total energy:

$$\eta_{ij}^{E} = \frac{1}{(E_{ij}^{h}+E_{ij}^{v}+E_{ij}^{a})} \tag{14}$$

Combining several factors in one heuristic brings a bookkeeping problem: if one factor lives around 0.1 and another around 1,000, the product is decided by the second factor alone. Min-Max normalization puts every factor on the same [0,1] footing before they are multiplied:

$$Q_k^{norm} = \frac{Q_k - Q_k^{min}}{Q_k^{max} - Q_k^{min} + \varepsilon} \tag{15}$$

where k belongs to D, P, F and epsilon = 10^{-8}. The combined heuristic with sensitivity tuning exponents is:

$$\eta_{\text{norm}} = Q_D^{\text{norm}\,\alpha} \cdot Q_P^{\text{norm}\,\beta} \cdot Q_F^{\text{norm}\,\gamma} \tag{16}$$

Default values: $\alpha = \beta = \gamma = 1.0$. Larger alpha favors shorter paths, larger beta favors fewer turns, larger gamma favors less altitude change. Pheromone is clamped between upper and lower bounds so the search does not lock onto a single path too early:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \textstyle\sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{17}$$

$$\tau_{min} \le \tau_{ij} \le \tau_{max}, \quad \Delta\tau_{ij}^{k} = Q/E_k \tag{18}$$

Grid-based ACO output is a polyline with corners sharp enough to violate the aircraft turn limits. A third-order B-spline pass over the waypoints—with the De Boor-Cox recursion for the basis functions—rounds off the corners while holding length and energy within one percent of the raw path, a deviation that is small enough to ignore. As shown in Figure 3, B-spline smoothing transforms the original polyline into a continuous curve with rounded turns.

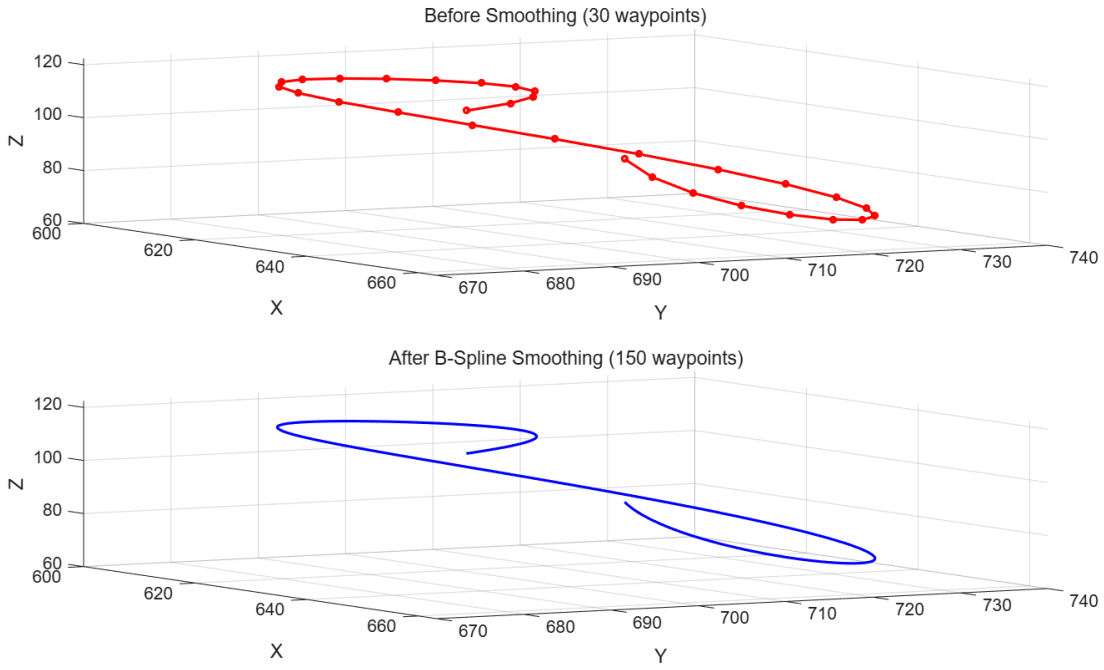


Figure 3. B-spline trajectory smoothing comparison (before vs after)

3.2. MACO-C dual-layer coverage planning

The area coverage problem is decomposed into two coupled optimization levels. The macro-layer (ACO-I) operates on a sub-region adjacency graph $G = (V, E)$ where vertices $V = R_1, ..., R_K$ represent sub-regions and edges E represent spatial adjacency. Edge weights are pre-computed offline using the improved ACO from Section 3.1 and stored in a static lookup table $W(K \times K)$. The online loop queries this table directly, so the same transition is never planned twice.

The micro-layer (ACO-II) optimizes intra-region coverage using three heuristic factors: coverage efficiency Q_{eff} (alignment with region major axis for fewer turns), energy factor Q_{nrg} (energy per unit area favoring efficient scanning patterns), and entry direction match Q_{entry} (alignment with inherited macro-layer direction for smooth transitions). The ACO state transition probability from node i to j is $P_{ij} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{s \in \text{allowed}_i} \tau_{is}^{\alpha} \cdot \eta_{is}^{\beta}}$, where the heuristic η incorporates the energy-reciprocal and normalized factors. The two layers pass information through their pheromones: after each micro-layer pass, the measured coverage energy is written back into the macro-layer pheromone of that sub-region, so the sequencing step works with actual cost rather than an estimate. The offline pre-computation of W cuts online complexity from

$O(N^2 * T_{ACO})$ per iteration to $O(1)$ table lookup, and re-planning on a changed map becomes a matter of a fresh lookup.

4. Case study: Shenzhen Longhua District

The test site is Shenzhen Longhua District—175.6 km^2 across six sub-districts, with elevations from 10 m to 380 m and a building stock dense enough to make obstacle avoidance non-trivial. Thirty sub-regions of roughly 2 km by 3 km serve as inspection targets, and the takeoff point sits at Shenzhen North Station. Terrain comes from an ALOS PALSAR 12.5 m DEM, buildings from OpenStreetMap footprints, and ArcGIS does the coordinate and raster chores.

Algorithm parameters: 20 ants, 150 iterations, α = 4, β = 3, ρ = 0.2, Q = 7, τ_{max} = 50, τ_{min} = 5. UAV parameters derived from DJI Mavic 3E specifications [5]: e_h = 150 J/m, k_v = 250 J/m, k_{yaw} = 80 J/rad, k_{pit} = 120 J/rad, L_{max} = 25 km, Θ = Φ = 60 degrees, H_{min} = 35 m, H_{max} = 200 m, w_{safe} = 10 m.

The optimized coverage route is shown in Figure 4, which illustrates the visiting sequence and connections among the 30 sub-regions. Across all 31 node pairs (one takeoff point plus 30 sub-region centers), the ACO planner returned collision-free 3D routes. Where buildings rise 40-120 m along Longhua Avenue, the paths climb over them; around the Foxconn campus and the Guanlan Lake residential towers, they detour sideways instead. Smoothing the resulting polylines with cubic B-splines removes the kinks while leaving energy totals within one percent of the raw paths.

As shown in Figure 6, MACO-C achieves the lowest mean energy consumption among the four algorithms over 30 independent runs, with the error bars indicating the standard deviation. MACO-C averaged 11.25 × 10^6 J over the 30 runs. Standard ACO needed 13.85 × 10^6 J, random scanning 16.52 × 10^6 J, and greedy nearest-neighbor 14.73 × 10^6 J—reductions of 18.8, 31.9, and 21.6 percent respectively. Runtime landed at 152.3 s against 385.6 s for standard ACO, a 60.5 percent cut that traces back to the pre-computed transition table doing the heavy lifting offline. The energy spread across runs tells the same story: 0.28 × 10^6 J for MACO-C versus 0.65 × 10^6 J for ACO, so the result is not just cheaper on average but more repeatable. The optimized assignment also handed 19 of the 30 sub-regions to the micro layer, up from 10 in the initial scanning solution.

As shown in Figure 5, the MACO-C coverage energy decreases rapidly during the early iterations and then fluctuates within a narrower range. For point-to-point planning, convergence is quick: within 25 iterations the planner drops from an initial 1,219.5 kJ over 245 waypoints to 81.6 kJ over just 7, covering 0.26 km at roughly 314 kJ/km. The 5 m-interval line-of-sight check keeps every segment clear of buildings above 40 m, which the endpoint-only test cannot guarantee.

Table 1 summarizes the energy consumption and runtime of the four methods over 30 independent runs. The baselines each pay for their speed or their simplicity somewhere: random scanning finishes in 3.5 s but burns 46.8% more energy, greedy nearest-neighbor needs 8.2 s at a 30.9% energy penalty, and plain ACO spends 385.6 s to land in between.

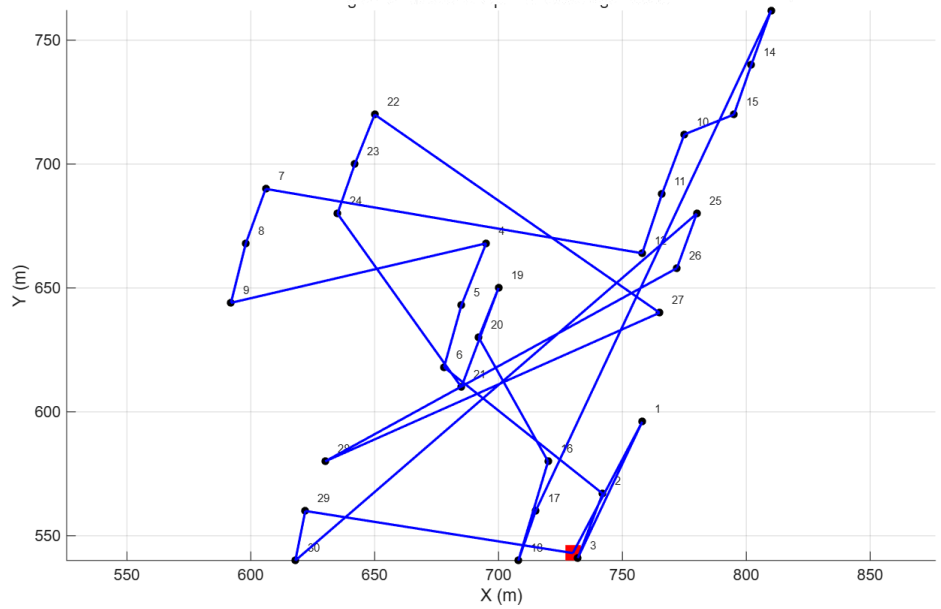


Figure 4. MACO-C optimal coverage route for Longhua District (30 sub-regions)

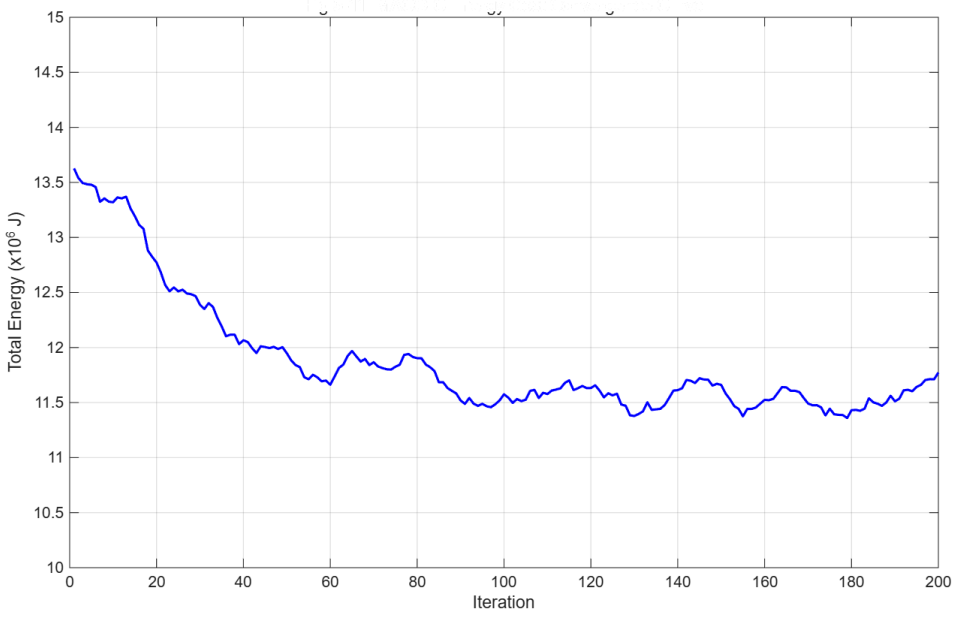


Figure 5. Energy cost iteration convergence curve of MACO-C algorithm

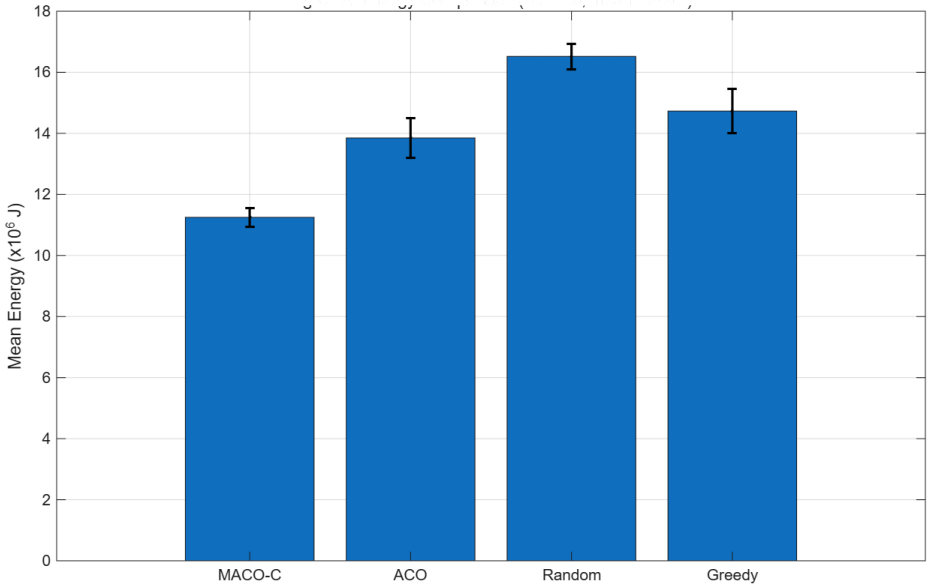


Figure 6. Algorithm energy comparison (30 independent runs, mean +/- Std)

Varying the problem size exposes where the two-layer split pays off: the MACO-C advantage widens from 18.8 percent at 30 sub-regions to 22.5 percent at 40, a trend that one expects when the macro layer is the scaling component. The energy coefficient e_h behaves almost linearly in the totals over the tested range of 100-200 J/m, matching the model assumption $E_i^h = e_h d_i^h (E_i^h = e_h d_i^h)$.

Moreover, the 15-20 percent margin over standard ACO holds across the whole range rather than only at the nominal setting. Sub-region area, by contrast, shows up only in the micro layer—roughly (0.15×10^5) J per additional km^2—while transfer energy between regions tracks their separation distances and nothing else.

5. Conclusion

Three things stand out from the Longhua runs. Treating flight energy as a three-term quantity—horizontal, vertical, and turning costs—changes what the planner does in ways a distance-based formulation cannot reproduce: the routes climb less, turn less, and trade small detours for sizable energy savings. Writing the assumptions down first mattered as much as the algorithm itself, because each assumption marks exactly where a constant e_h stops being defensible. The offline lookup table for inter-region transitions removed the main computational bottleneck, cutting runtime from 385.6 s to 152.3 s, and the two-layer decomposition shrank the energy spread across 30 runs to 0.28×10^6 J. Each assumption also points to a next step—a velocity-dependent $e_h(v)$ with a wind term, obstacle handling that reacts to birds and other aircraft, and payload-aware endurance for delivery missions. Multi-UAV coordination, where several aircraft divide the same region, would push the same ideas toward scheduling rather than pure path design.

References

[1] Xiang, D. (2023). *UAV 3D trajectory planning and vehicle-drone collaborative delivery route optimization* [Master thesis, Beijing Jiaotong University].

[2] Dorigo, M., & Stutzle, T. (2004). *Ant colony optimization*. MIT Press.

[3] Cabreira, T. M., Brisolara, L. B., & Ferreira, P. R., Jr. (2019). Survey on coverage path planning with unmanned aerial vehicles. *Drones, 3*(1), Article 4. https://doi.org/10.3390/drones3010004

[4] Galceran, E., & Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems, 61*(12), 1258-1276. https://doi.org/10.1016/j.robot.2013.09.004

[5] DJI. (2024). *DJI Mavic 3E technical specifications*. https://www.dji.com/mavic-3-enterprise/specs