

Reducing VR Delivery Latency in UAV-Enabled Mobile Edge Computing using PSO, GA, and Gradient-Based Approaches

Hilal H. Nuha ¹*, M. Faris Fathoni ¹, and Hassan Rizky Putra Saillellah ¹

¹School of Computing, Telkom University, Bandung, Indonesia

Abstract

Virtual reality (VR) services delivered through UAV-enabled mobile edge computing (MEC) require joint decisions on user association, caching, computing-resource allocation, and UAV placement. This study benchmarks three CVX-free placement solvers, namely Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and gradient-based L-BFGS-B, within the same alternating-optimization framework for $N \in \{15, 18, 21, 24, 27, 30\}$ VR users served by $M = 2$ UAV base stations (UBSs). The channel rate follows the outage-constrained logistic approximation of an elevation-angle-dependent Rician fading model. To account for stochastic variability, the evaluation uses 30 independent Monte Carlo system realizations for every N , paired across solvers, and includes random and centroid placement baselines. The optimized maximum aggregate UBS task-completion latency increases from approximately 225 ms at $N = 15$ to 723 ms at $N = 30$. PSO and GA achieve numerically identical latency values, while L-BFGS-B yields comparable latency at each tested user load, differing by at most 0.714 ms in any paired comparison; after Holm correction, only the $N = 15$ difference is statistically significant and its median magnitude is only 0.054 ms. Relative to random placement, optimization reduces mean aggregate latency by approximately 21–33%, whereas the gain over centroid placement is modest (about 0.26–0.47%). L-BFGS-B has the lowest average runtime over all loads, although PSO is faster at several individual load levels, and GA is consistently the slowest. The reported objective is an aggregate task-level latency metric inherited from the reference UAV-MEC model and must not be interpreted as per-frame motion-to-photon latency.

Keywords: wireless networks, virtual reality, UAV-enabled MEC, Rician fading, latency minimization, particle swarm optimization, genetic algorithm, gradient-based optimization

Received on 14 June 2026, accepted on 12 October 2026, published on 23 October 2026

Copyright © 2026 Hilal H. Nuha *et al.*, licensed to EAI. This is an open access article distributed under the terms of the [CC BY-NC-SA 4.0](#), which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/airo.13507

1. Introduction

Interactive VR imposes stringent end-to-end latency requirements. Bastug *et al.* [1] note that humans begin to notice lag after about 13 ms and evaluate VR task deadlines with a mean of 10 ms, while Chen *et al.* [2] adopt a 20 ms delay target for the wireless transmission of VR content and tracking information. In this paper, 20 ms is used as a representative target for this type of interactive, motion-to-photon (MTP) latency. However, the optimization model considered in this paper does *not* directly model headset MTP latency. Following

Nasir [3], the objective is the maximum, over all UAV base stations (UBSs), of the *sum of task-completion latencies* of the VR users (VRUs) assigned to each UBS. It is therefore an aggregate task-level scheduling metric rather than a per-frame MTP measurement. This distinction is important: the numerical values reported later are used to compare optimization strategies under the same system model, not to claim compliance with a 20 ms real-time VR requirement.

UAV-enabled MEC remains attractive for delay-sensitive applications because airborne edge servers can move closer to users and jointly provide communication, caching, and computation resources [3, 4]. In related UAV-aided MEC research, Liu *et al.* [5]

*Corresponding author. Email: hilalnuha@ieee.org

analyze outage probability under edge-node selection criteria based on received signal-to-noise ratio, computational capability, and random selection. Recent UAV systems also demonstrate increasingly demanding onboard processing in application domains such as object detection [6], precision aerial perception and geo-localization [7], gas sensing [8], and agricultural robotics [9]. These studies provide practical context for the growing importance of computationally efficient onboard algorithms, although they address different applications from the VR-MEC latency problem considered here.

The reference formulation couples four decisions: VRU–UBS association, caching, CPU allocation, and horizontal UBS placement. The placement variables affect both fronthaul and backhaul rates through an elevation-angle-dependent Rician reliability model [10]. Existing UAV-MEC work frequently uses convexification, successive convex approximation, or hybrid metaheuristics for related task-offloading and placement problems [3, 4]. In a complementary aerial-network setting, Nguyen et al. [11] use deep reinforcement learning to optimize UAV placement and user association in optical intelligent reflecting surface-assisted hybrid free-space optical/radio-frequency networks while respecting backhaul capacity constraints. Their throughput objective and communication architecture differ from those considered here, but the study provides relevant context for the coupling between placement, association, and backhaul capacity. The contribution of the present work is deliberately narrower: we hold the system model and the other optimization blocks fixed and benchmark three open-source placement solvers within the same pipeline.

The broader optimization literature continues to develop population-based and hybrid search methods for non-convex engineering problems. Recent examples include the Komodo Mlipir Algorithm and the Battlefield Optimization Algorithm, which emphasize adaptive exploration–exploitation mechanisms in metaheuristic search [12, 13]. Particle swarm optimization (PSO) and related population-based methods have also been used for hybrid prediction and hyperparameter optimization [14–17]. In UAV path planning, Chang et al. [18] combine an adaptive spherical vector-based PSO algorithm for offline global planning with an enhanced collision-avoidance vector field for local planning. Recent network-oriented studies also apply evolutionary or swarm-based optimization to delay recovery, transmission-power control, clustering, and energy-aware prediction [19–22]. These works address different applications and optimization objectives from the present UAV–MEC placement problem. They motivate

continued interest in metaheuristic solvers and reinforce the need for controlled, statistically supported solver comparisons.

The main contributions are as follows:

- We provide a CVX-free implementation of the four-block alternating framework and compare PSO, GA, and L-BFGS-B only in the UAV placement block, keeping association, caching, and frequency-allocation logic identical across solvers.
- We use an analytical subgradient for the L-BFGS-B placement objective and a KKT-based solution for the convex frequency-allocation block, removing the need for a commercial convex solver in the implemented pipeline.
- We strengthen the evaluation to 30 paired Monte Carlo system realizations for every tested user load and report mean $\pm$ standard deviation, paired Wilcoxon signed-rank tests with Holm correction, and convergence/diversity diagnostics.
- We add Random and Centroid placement baselines to quantify the absolute benefit of placement optimization and explicitly distinguish the aggregate model latency from per-frame MTP latency.

The remainder of this paper is organized as follows. Section 2 describes the system model. Section 3 formulates the optimization problem and presents the alternating optimization framework and the three placement solvers. Section 4 reports the experimental results, statistical analysis, baselines, runtime, and limitations. Conclusions are given in Section 5.

2. System Model

Consider a UAV-enabled mobile edge computing (MEC) system (Fig. 1) comprising $M = 2$ unmanned aerial vehicle base stations (UBSs), indexed by $\mathcal{M} = \{1, \dots, M\}$, and N ground users (VRUs), indexed by $\mathcal{N} = \{1, \dots, N\}$. A single ground cloud server is located at coordinates $(c_x, c_y) = (1000, 1000)$ m. The system operates over a service area of $[0, 1000] \times [0, 1000]$ m².

Each UBS $j \in \mathcal{M}$ hovers at a fixed altitude $H_j = 150$ m and is equipped with caching and computing capabilities. Let $\mathbf{q}_j = (q_{j,x}, q_{j,y}) \in \mathbb{R}^2$ denote the horizontal position of UBS j , and $\mathbf{w}_i \in \mathbb{R}^2$ the horizontal position of user i .

2.1. Distance and Channel Model

The three-dimensional Euclidean distance between user i and UBS j is

$$d_{i,j} = \sqrt{\|\mathbf{q}_j - \mathbf{w}_i\|^2 + H_j^2}, \quad (1)$$

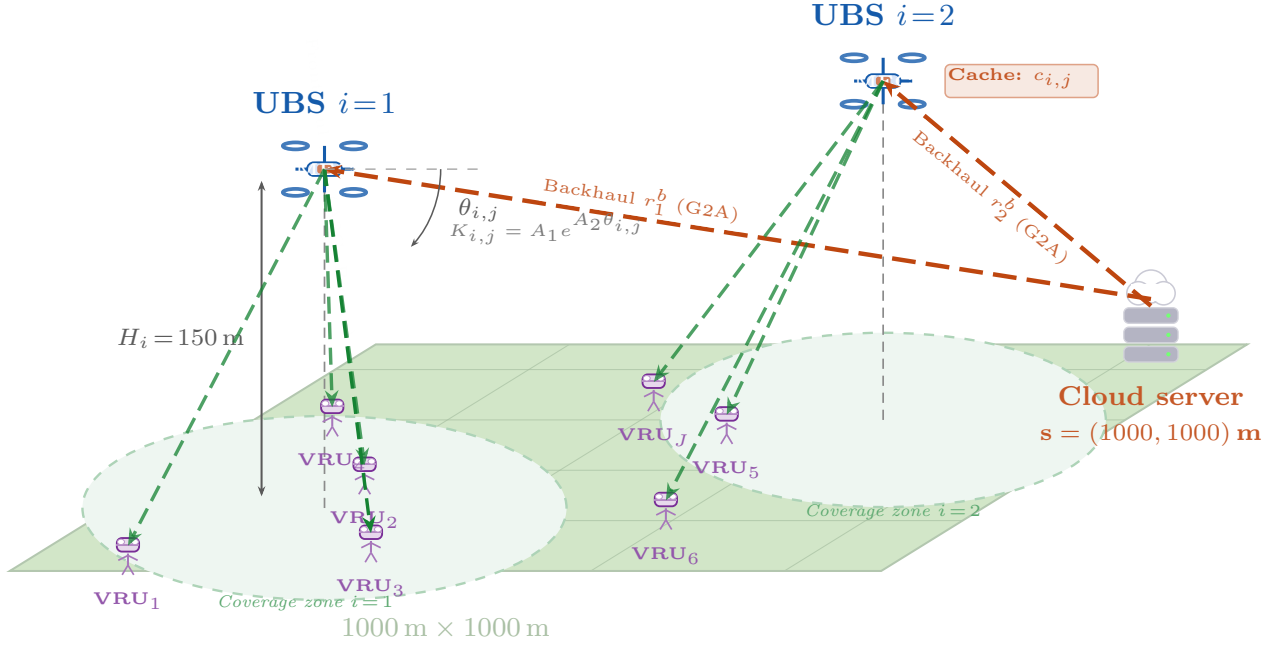


Figure 1. UAV-enabled MEC network topology for VR delivery under Rician fading channels. Two UBSs ($j = 1, 2$) serve N VRUs via A2G fronthaul links ($r_{i,j}$) while receiving VR content from the cloud server via G2A backhaul links (r_j^b).

and the distance between the cloud server and UBS j is

$$d_j^b = \sqrt{\|\mathbf{q}_j - \mathbf{c}\|^2 + H_j^2}, \quad (2)$$

where $\mathbf{c} = (c_x, c_y)$.

Following Nasir [3], which in turn adopts the regression-based treatment of You and Zhang [10], the reliable A2G rate is represented using a logistic approximation of the *effective fading power* associated with an elevation-angle-dependent Rician channel. Thus, the logistic term below should not be interpreted as an independent Bernoulli LoS-probability model; it is an approximation used to obtain a tractable outage-constrained rate under Rician fading:

$$r_{i,j}(\mathbf{q}_j) = B \log_2 \left(1 + \left(K_1 + \frac{K_2}{1 + e^{-(K_3 + K_4 v_{i,j})}} \right) \frac{\gamma}{d_{i,j}^{\alpha_1}} \right), \quad (3)$$

where $v_{i,j} = H_j/d_{i,j}$ is the sine of the elevation angle, $B = 180 \text{ kHz}$, $\gamma = p_u \beta_0 / (\sigma^2 \Gamma)$, $p_u = 0.1 \text{ W}$, $\beta_0 = 10^{-5}$ is the channel power gain at the reference distance, $\sigma^2 = N_0 B$ is the noise power with $N_0 = -174 \text{ dBm/Hz}$, and $\Gamma = 10^{8.2/10}$ is the SNR gap. The logistic constants are $K_1 = 0.01$, $K_2 = 0.99$, $K_3 = -4.7$, and $K_4 = 8.9$, with A2G path-loss exponent $\alpha_1 = 2.3$.

The cloud-to-UBS backhaul rate uses the same approximation with $v_j^b = H_j/d_j^b$, $\gamma_b = p_c \beta_0 / (\sigma^2 \Gamma)$, $p_c =$

0.1 W , and $\alpha_2 = 2.5$:

$$r_j^b(\mathbf{q}_j) = B \log_2 \left(1 + \left(K_1 + \frac{K_2}{1 + e^{-(K_3 + K_4 v_j^b)}} \right) \frac{\gamma_b}{(d_j^b)^{\alpha_2}} \right). \quad (4)$$

2.2. Task and Latency Model

Each user i has a computational task of data size D_i bits requiring F_i CPU cycles per bit. The output data size is $O_i = 2D_i$ bits (output expansion ratio $\eta = 2$).

Let $\alpha_{i,j} \in \{0, 1\}$ be the binary association variable, where $\alpha_{i,j} = 1$ indicates that user i is served by UBS j . Let $c_{i,j} \in \{0, 1\}$ be the caching indicator, where $c_{i,j} = 1$ means UBS j has pre-cached user i 's data. Let $f_{i,j} > 0$ denote the CPU frequency (cycles/s) of UBS j allocated to user i .

The total task completion latency for user i assigned to UBS j consists of three components:

1. **Upload latency** (user data to UBS via A2G link):

$$t_{i,j}^u = \frac{O_i}{r_{i,j}(\mathbf{q}_j)}, \quad (5)$$

2. **Backhaul latency** (cloud server to UBS, incurred only when data is not cached):

$$t_{i,j}^b = \frac{(1 - c_{i,j}) D_i}{r_j^b(\mathbf{q}_j)}, \quad (6)$$

3. Compute latency (VR data processing at UBS):

$$t_{i,j}^c = \frac{D_i F_i}{f_{i,j}}. \quad (7)$$

The total latency experienced by UBS j is the sum over all its assigned users:

$$\mu_j = \sum_{i=1}^N \alpha_{i,j} (t_{i,j}^u + t_{i,j}^b + t_{i,j}^c) = \sum_{i=1}^N \alpha_{i,j} \left(\frac{O_i}{r_{i,j}} + \frac{(1 - c_{i,j}) D_i}{r_j^b} + \frac{D_i F_i}{f_{i,j}} \right). \quad (8)$$

The optimization objective is the maximum aggregate UBS task-completion latency

$$\mu = \max_{j \in \mathcal{M}} \mu_j. \quad (9)$$

Because μ_j is a *sum* over all users assigned to UBS j , μ is not a per-user, per-frame, or headset MTP latency. Consequently, values above 20 ms do not by themselves indicate a violation of the mathematical model; instead, they show that the adopted parameter regime cannot be claimed to satisfy a strict 20 ms MTP target. Establishing actual MTP feasibility would require a frame/tile-level rendering and display pipeline that is outside the scope of this model.

3. Optimization Problem

This section formulates the joint optimization problem and presents its solution. The first subsection describes the alternating optimization framework and its four blocks. The second subsection presents the three placement solvers.

The joint resource allocation problem minimizes the worst-case latency μ over the association variables α , caching policy $\mathbf{c}$, frequency allocation $\mathbf{f}$, and UAV horizontal positions $\mathbf{q}$:

$$\min_{\alpha, \mathbf{c}, \mathbf{f}, \mathbf{q}} \max_{j \in \mathcal{M}} \mu_j \quad (10a)$$

$$\text{subject to } \sum_{j=1}^M \alpha_{i,j} = 1, \quad \alpha_{i,j} \in \{0, 1\}, \quad \forall i \in \mathcal{N}, j \in \mathcal{M}, \quad (10b)$$

$$\sum_{i=1}^N \alpha_{i,j} c_{i,j} D_i \leq c_{\max}, \quad \forall j \in \mathcal{M}, \quad (10c)$$

$$\sum_{i=1}^N \alpha_{i,j} f_{i,j} \leq f_{\max}, \quad \forall j \in \mathcal{M}, \quad (10d)$$

$$p_u + \kappa \sum_{i=1}^N \alpha_{i,j} f_{i,j}^3 \leq P_{\max}, \quad \forall j \in \mathcal{M}, \quad (10e)$$

$$X_{\min} \leq q_{j,x} \leq X_{\max}, \quad (10f)$$

$$Y_{\min} \leq q_{j,y} \leq Y_{\max}, \quad \forall j \in \mathcal{M}, \quad (10g)$$

$$f_{i,j} \geq 0, \quad \forall i \in \mathcal{N}, j \in \mathcal{M}, \quad (10h)$$

where the constraints are explained as follows.

- (10b): each user is served by exactly one UBS (single association).
- (10c): the total cached data at each UBS does not exceed the cache capacity $c_{\max}$.
- (10d): the total CPU frequency allocated by each UBS does not exceed $f_{\max} = 5$ GHz.
- (10e): the total power at each UBS (transmit power p_u plus computing power $\kappa \sum_i \alpha_{i,j} f_{i,j}^3$) does not exceed the budget $P_{\max} = 5$ W, where $\kappa = 10^{-27}$ is the effective switched capacitance.
- (10f)–(10g): UAV horizontal positions are constrained to the service area $[0, 1000]^2$ m².

Problem (10) is non-convex due to the integer constraints in (10b)–(10c), the non-convex rate functions $r_{i,j}(\mathbf{q}_j)$ and $r_j^b(\mathbf{q}_j)$ through their dependence on the UAV position, and the product terms $\alpha_{i,j} f_{i,j}^3$ in the power constraint (10e).

3.1. Proposed Solution: Alternating Optimization

The joint problem (10) is solved via block coordinate descent (BCD), iterating over four sub-problems until the relative change in μ falls below $\epsilon = 10^{-4}$:

$$\frac{|\mu^{(k)} - \mu^{(k-1)}|}{\mu^{(k)}} < \epsilon. \quad (11)$$

3.1.1. Sub-problem 1: User Association (P1)

With $(\mathbf{c}, \mathbf{f}, \mathbf{q})$ fixed, the implementation assigns each user to the UBS that currently gives the smallest individual task latency:

$$j^*(i) = \arg \min_{j \in \mathcal{M}} \left(\frac{O_i}{r_{i,j}} + \frac{(1 - c_{i,j})D_i}{r_j^b} + \frac{D_i F_i}{f_{i,j}} \right). \quad (12)$$

This rule has complexity $O(NM)$. Importantly, because the global objective is a maximum of *aggregate* UBS loads, this per-user greedy rule is a heuristic block update and is not claimed to solve the binary min-max association problem globally; load-balancing interactions across users can make the globally optimal association different.

3.1.2. Sub-problem 2: Caching Policy (P2)

With $(\alpha, \mathbf{f}, \mathbf{q})$ fixed, the code ranks assigned items using

$$\mu_{i,j}^c = \alpha_{i,j} \left(\frac{O_i}{r_{i,j}} + \frac{D_i F_i}{f_{i,j}} + \frac{D_i}{r_j^b} \right) \quad (13)$$

and caches items greedily while capacity remains. Since the item sizes D_i are not identical, this is a heuristic for the binary knapsack-like cache block rather than an exact optimizer. Therefore, the complete alternating scheme is treated as a practical heuristic with no global-optimality guarantee.

3.1.3. Sub-problem 3: Frequency Allocation (P3)

For fixed $(\alpha, \mathbf{c}, \mathbf{q})$, the frequency-dependent term for an assigned task is $a_i/f_{i,j}$ with $a_i = D_i F_i$. For each UBS j , the convex block is constrained by both $\sum_i \alpha_{i,j} f_{i,j} \leq f_{\max}$ and $p_u + \kappa \sum_i \alpha_{i,j} f_{i,j}^3 \leq P_{\max}$. With dual multipliers $\lambda_j \geq 0$ and $\nu_j \geq 0$, KKT stationarity gives

$$-\frac{a_i}{f_{i,j}^2} + \lambda_j + 3\nu_j \kappa f_{i,j}^2 = 0. \quad (14)$$

Letting $y = f_{i,j}^2$, the positive solution is

$$f_{i,j}^* = \sqrt{\frac{-\lambda_j + \sqrt{\lambda_j^2 + 12\nu_j \kappa a_i}}{6\nu_j \kappa}}, \quad \nu_j > 0, \quad (15)$$

while for $\nu_j = 0$, $f_{i,j}^* = \sqrt{a_i/\lambda_j}$. The dual variables are determined by nested bisection so that the active resource constraints are satisfied. Here $f_{\max} = 5$ GHz is an *aggregate modeled compute-rate budget* across tasks at a UBS, not a claim that a specific 5 GHz processor dissipates only 5 W.

3.1.4. Sub-problem 4: UAV Placement (P4)

With $(\alpha, \mathbf{c}, \mathbf{f})$ fixed, the UAV placement sub-problem minimizes

$$\min_{\mathbf{q}} \max_{j \in \mathcal{M}} \mu_j(\mathbf{q}_j), \quad (16)$$

subject to (10f)–(10g). This sub-problem is non-convex in $\mathbf{q}$ due to the log-sigmoid rate functions (3) and (4).

3.2. Proposed Solvers

We compare three solver strategies:

3.2.1. Gradient-based: L-BFGS-B with Analytical Subgradient

The subgradient of the active UAV's latency μ_{j^*} with respect to $\mathbf{q}_{j^*}$ is derived by the chain rule through the rate function:

$$\frac{\partial r_{i,j}}{\partial \mathbf{q}_j} = \frac{B}{\ln 2 (1 + \text{SNR}_{i,j})} \cdot \frac{\partial \text{SNR}_{i,j}}{\partial \mathbf{q}_j}, \quad (17)$$

where

$$\frac{\partial \text{SNR}_{i,j}}{\partial \mathbf{q}_j} = \frac{\gamma}{d_{i,j}^{\alpha_1}} \frac{\partial \ell_{i,j}}{\partial \mathbf{q}_j} + \ell_{i,j} \gamma (-\alpha_1) d_{i,j}^{-\alpha_1-1} \frac{\partial d_{i,j}}{\partial \mathbf{q}_j}, \quad (18)$$

with $\ell_{i,j} = K_1 + K_2 \sigma_{i,j}$, $\sigma_{i,j} = (1 + e^{-(K_3 + K_4 v_{i,j})})^{-1}$, $\partial d_{i,j}/\partial \mathbf{q}_j = (\mathbf{q}_j - \mathbf{w}_i)/d_{i,j}$, and

$$\frac{\partial \ell_{i,j}}{\partial \mathbf{q}_j} = K_2 \sigma_{i,j} (1 - \sigma_{i,j}) K_4 \cdot \frac{\partial v_{i,j}}{\partial \mathbf{q}_j}, \quad \frac{\partial v_{i,j}}{\partial \mathbf{q}_j} = -\frac{H_j}{d_{i,j}^2} \frac{\partial d_{i,j}}{\partial \mathbf{q}_j}. \quad (19)$$

The analytical subgradient is supplied to L-BFGS-B [23] with box constraints (10f)–(10g).

3.2.2. Particle Swarm Optimization (PSO)

A swarm of $n_p = 30$ particles evolves the UAV positions via the velocity-position update:

$$\mathbf{v}_k^{(t+1)} = w \mathbf{v}_k^{(t)} + c_1 \mathbf{r}_1 \odot (\mathbf{p}_k - \mathbf{x}_k^{(t)}) + c_2 \mathbf{r}_2 \odot (\mathbf{g} - \mathbf{x}_k^{(t)}), \quad (20)$$

$$\mathbf{x}_k^{(t+1)} = \text{clip}(\mathbf{x}_k^{(t)} + \mathbf{v}_k^{(t+1)}, \mathbf{lb}, \mathbf{ub}), \quad (21)$$

where $w = 0.7$ is the inertia weight, $c_1 = c_2 = 1.5$ are the cognitive and social coefficients, $\mathbf{r}_1, \mathbf{r}_2 \sim \mathcal{U}(0, 1)^{2M}$, $\mathbf{p}_k$ is the personal best of particle k , and $\mathbf{g}$ is the global best position. The clip operation enforces (10f)–(10g) [24].

3.2.3. Genetic Algorithm (GA)

A population of $n_g = 30$ real-valued chromosomes each encoding the concatenated UAV coordinates $[\mathbf{q}_1; \dots; \mathbf{q}_M] \in \mathbb{R}^{2M}$ evolves through:

1. **Tournament selection** (size 2): for each offspring slot, draw two candidates uniformly at random and select the fitter one.
2. **Arithmetic crossover** (probability $p_c = 0.8$): for a random blend parameter $a \sim \mathcal{U}(0, 1)^{2M}$,

$$\begin{aligned} \mathbf{x}'_1 &= a \odot \mathbf{x}_1 + (1 - a) \odot \mathbf{x}_2, \\ \mathbf{x}'_2 &= (1 - a) \odot \mathbf{x}_1 + a \odot \mathbf{x}_2. \end{aligned} \quad (22)$$

3. **Gaussian mutation** (per-gene probability $p_m = 0.15$): $\mathbf{x} \leftarrow \mathbf{x} + \delta$, where $\delta \sim \mathcal{N}(0, 0.05(\mathbf{ub} - \mathbf{lb}))$ on mutated genes.

4. **Elitism:** the best individual survives unchanged into the next generation.

Box constraints (10f)–(10g) are enforced by clipping after mutation [25].

4. Experimental Results

This section presents the experimental results. The first subsection describes the simulation protocol. The second subsection discusses latency, statistical robustness, and baselines. The third and fourth subsections analyze convergence and runtime, and the last subsection discusses limitations.

4.1. Simulation protocol and parameter interpretation

Table 1 summarizes the system parameters. Unless otherwise noted, these values are retained from the reference formulation [3] to enable comparison with that model.

Table 1. System and optimization parameters

Parameter	Symbol	Value	Unit
Number of UBSs	M	2	—
UBS altitude	H_j	150	m
Service area	—	$[0, 1000]^2$	m^2
Cloud server position	$\mathbf{c}$	(1000, 1000)	m
Bandwidth	B	180	kHz
Noise PSD	N_0	-174	dBm/Hz
Reference channel gain	β_0	10^{-5}	—
SNR gap	Γ	$10^{8.2/10}$	—
UBS / cloud transmit power	p_u, p_c	0.1, 0.1	W
Max modeled power budget	$P_{\max}$	5	W
Aggregate compute-rate limit	$f_{\max}$	5	GHz
Task size	D_i	$\mathcal{U}(10, 15)$	kbit
Computational complexity	F_i	$\mathcal{U}(500, 800)$	cycles/bit
Cache capacity	$c_{\max}$	60	kbit
Effective switched capacitance	κ	10^{-27}	—
A2G/G2A path-loss exponents	α_1, α_2	2.3, 2.5	—
Output expansion ratio	η	2	—
Logistic constants	K_1, K_2 K_3, K_4	0.01, 0.99 -4.7, 8.9	—
BCD convergence tolerance	ϵ	10^{-4}	—
PSO particles / iterations	—	30 / 100	—
PSO inertia / acceleration	w, c_1, c_2	0.7, 1.5, 1.5	—
GA population / generations	—	30 / 100	—
GA crossover / mutation prob.	p_c, p_m	0.8, 0.15	—
Independent scenarios per N	—	30	—

Following the publicly released implementation of [3], the task size and cache capacity are expressed in

bits.¹ Two parameters should be interpreted cautiously. First, $c_{\max} = 60 \text{ kbit}$ (7.5 KB) is far smaller than contemporary full-frame VR assets; it is therefore best viewed as a small, normalized cache budget in the mathematical benchmark, not as a realistic complete VR-content store. Second, $f_{\max} = 5 \text{ GHz}$ is an aggregate modeled compute-rate allocation limit, while the cubic dynamic-power equation is an abstraction rather than a hardware-validated 5 GHz/5 W processor specification.

For each $N \in \{15, 18, 21, 24, 27, 30\}$, we generated 30 independent Monte Carlo system realizations. VRU positions were sampled uniformly in the $1000 \times 1000 \text{ m}$ service area, $D_i \sim \mathcal{U}(10, 15) \text{ kbit}$, and $F_i \sim \mathcal{U}(500, 800) \text{ cycles/bit}$. The same realization was paired across PSO, GA, and L-BFGS-B. All three solvers start from the same incumbent placement, namely the K -means ($K = 2$) centroid of the VRU positions, and a candidate placement is accepted only when the full four-block pipeline does not worsen the incumbent objective. Random placement and centroid placement were evaluated on the same 30 scenarios as baselines. Results are reported as mean $\pm$ sample standard deviation.

4.2. Latency, statistical robustness, and baselines

Table 2 reports the 30-scenario results. It can be noticed that the optimized latency grows from approximately 225 ms at $N = 15$ to 723 ms at $N = 30$, as the aggregate computation and transmission load per UBS increases with N . These magnitudes are consistent with the CVX-based benchmark of [3], which reports approximately 0.45–1.08 s for the same range of N . The reported quantity is the maximum aggregate UBS task-completion latency μ , not MTP delay.

Across the six load levels, the optimized mean latency is only 0.26–0.47% below the centroid baseline (0.32–0.47% for PSO and GA, 0.26–0.45% for L-BFGS-B), but 21.0–32.6% below the random-placement mean, as depicted in Fig. 2. Therefore, the data do *not* support a claim that sophisticated placement always provides a large gain over a strong geometric heuristic. Instead, the benefit is large relative to naive random placement and modest relative to centroid placement.

A paired Wilcoxon signed-rank test was applied to each solver pair at each N , followed by Holm correction over all 18 pairwise tests, as reported in Table 3. PSO and GA are numerically identical in all scenarios (paired differences below 10^{-6} ms). L-BFGS-B is never better than PSO and GA; it either matches them or is marginally worse, by at most

¹Reference [3] reports $D_i \sim \mathcal{U}(10, 15) \text{ KB}$ and $c_{\max} = 60 \text{ KB}$, whereas its released MATLAB implementation uses these values in bits (10^4 – 1.5×10^4 bits and 6×10^4 bits). We retain the implemented values; with byte-valued sizes all latencies would increase by a factor of approximately eight.

Table 2. Maximum aggregate UBS task-completion latency, mean $\pm$ standard deviation (ms), 30 paired scenarios per N

N	PSO	GA	L-BFGS-B	Centroid	Random
15	224.76 $\pm$ 76.86	224.76 $\pm$ 76.86	224.89 $\pm$ 76.81	225.48 $\pm$ 76.95	309.47 $\pm$ 89.16
18	307.69 $\pm$ 73.50	307.69 $\pm$ 73.50	307.75 $\pm$ 73.49	309.13 $\pm$ 73.90	389.75 $\pm$ 119.11
21	444.98 $\pm$ 106.24	444.98 $\pm$ 106.24	445.02 $\pm$ 106.24	446.55 $\pm$ 106.48	581.47 $\pm$ 175.21
24	516.52 $\pm$ 118.78	516.52 $\pm$ 118.78	516.54 $\pm$ 118.77	518.77 $\pm$ 118.93	667.55 $\pm$ 266.20
27	599.16 $\pm$ 154.40	599.16 $\pm$ 154.40	599.16 $\pm$ 154.40	601.50 $\pm$ 155.31	888.96 $\pm$ 275.08
30	722.87 $\pm$ 163.86	722.87 $\pm$ 163.86	722.87 $\pm$ 163.86	725.46 $\pm$ 164.40	1018.81 $\pm$ 315.27

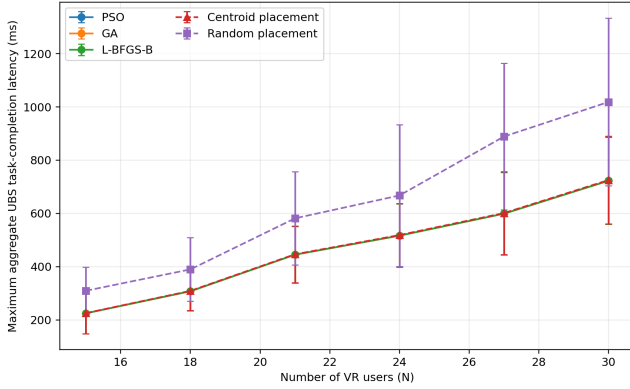


Figure 2. Thirty-scenario mean $\pm$ standard deviation of the maximum aggregate UBS task-completion latency for the three placement solvers and two baselines. Markers are offset horizontally around each N for visibility.

0.714ms over all 180 paired cases. Only $N = 15$ remains significant after Holm correction ($p_{\text{Holm}} = 0.00237$); however, the median paired difference is only 0.054ms and the maximum is 0.612ms at that load. At $N = 18, 21, 24, 27, 30$, corrected p -values are at least 0.287. Therefore, there is no evidence of a *practically meaningful or systematic* latency-quality superiority of one placement solver across the tested loads.

Table 3. Holm-adjusted paired Wilcoxon p -values

N	PSO-GA	PSO-L-BFGS-B	GA-L-BFGS-B
15	1.000	0.0024	0.0024
18	1.000	0.2874	0.2874
21	1.000	0.2874	0.2874
24	1.000	0.5060	0.5060
27	1.000	1.000	1.000
30	1.000	1.000	1.000

PSO-GA: all paired differences are zero to numerical precision, so $p = 1$ is reported by convention.

4.3. Convergence and PSO behavior

Fig. 3 shows the outer BCD trajectory for a representative scenario at $N = 30$. It can be observed that all three solvers reduce the objective from the centroid incumbent (664.8ms) to 659.5ms in the first outer

iteration and satisfy the stopping criterion (11) at the second; across all 540 runs, the BCD loop terminates after 1.6–2.0 outer iterations on average. In this scenario the three solvers reach the same objective, which indicates that the P4 landscape around the incumbent does not separate them. Fig. 4 depicts the normalized PSO positional diversity, which contracts from 0.25 to about 0.01 within 50 iterations as information about the best region is shared. In the 30-scenario evaluation, PSO does not exhibit a systematic quality degradation at any N . Therefore, stochastic behavior changes the search trajectory and runtime, but not the attained latency quality in the tested settings.

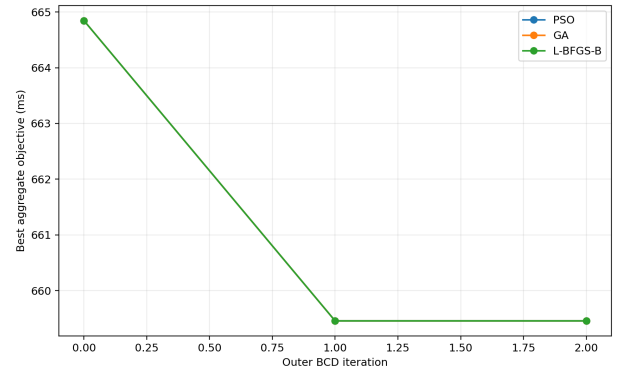


Figure 3. Outer BCD convergence for a representative scenario at $N = 30$. Iteration 0 is the common centroid incumbent; the three solver trajectories coincide.

4.4. Runtime

Table 4 reports the end-to-end optimization runtime of the implemented alternating pipeline. L-BFGS-B has the lowest runtime averaged across all six load levels (0.155s versus 0.178s for PSO and 0.720s for GA), but it is not the fastest at every individual N . PSO is faster at $N = 21, 24, 30$, while L-BFGS-B is faster at $N = 15, 18, 27$. GA is consistently slower than both, by a factor of 3.6–5.1 relative to PSO. Therefore, the runtime advantage of L-BFGS-B is moderate and load-dependent rather than universal.

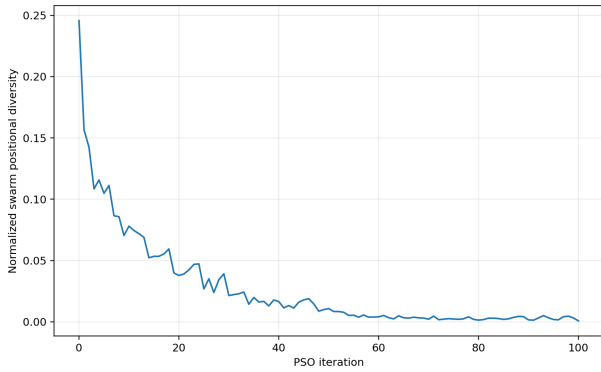


Figure 4. Normalized PSO swarm positional diversity at $N = 30$

Table 4. Optimization runtime, mean $\pm$ standard deviation (s)

N	PSO	GA	L-BFGS-B
15	$0.196 \pm 0.310^{\dagger}$	0.992 ± 0.677	0.052 ± 0.039
18	0.166 ± 0.091	0.712 ± 0.262	0.088 ± 0.044
21	0.169 ± 0.092	0.626 ± 0.231	0.177 ± 0.186
24	0.203 ± 0.156	0.756 ± 0.377	0.244 ± 0.209
27	0.163 ± 0.063	0.587 ± 0.207	0.145 ± 0.166
30	0.174 ± 0.059	0.649 ± 0.119	0.224 ± 0.193

[†]Includes one 1.82 s run (first-call overhead); the other 29 runs give 0.140 ± 0.051 s.

4.5. Limitations and practical interpretation

These results expose several limitations that are important for implementation. First, the objective is aggregate task-level latency and cannot establish 20 ms MTP feasibility. Second, the 60 kbit cache and abstract compute/power model are inherited benchmark parameters, not a hardware-validated VR platform specification. Third, P1 and P2 are greedy block heuristics, so the complete BCD procedure has no global-optimality guarantee; the centroid incumbent safeguard limits degradation but does not close the sub-optimality gap. Finally, UAV mobility, propulsion energy, time-varying users, altitude optimization, larger UAV fleets, and online control remain outside the present static benchmark.

5. Conclusion

This paper benchmarked PSO, GA, and L-BFGS-B for UAV placement within the same CVX-free alternating optimization pipeline for UAV-enabled VR/MEC, using 30 paired Monte Carlo scenarios for each user load. PSO and GA attain numerically identical latency values in the tested scenarios, and L-BFGS-B differs from them by at most 0.714 ms in any paired comparison. After Holm correction, only the $N = 15$ comparison with L-BFGS-B is statistically significant, but its median effect is only 0.054 ms; hence no solver demonstrates

a practically meaningful or systematic latency-quality advantage over the tested loads.

The baseline comparison is equally important: optimization lowers mean aggregate latency by approximately 21–33% relative to random placement, but only about 0.26–0.47% relative to centroid placement. L-BFGS-B has the lowest runtime averaged over all loads, PSO is competitive and faster at several individual loads, and GA is consistently the slowest in the current Python implementation. These findings support solver selection based on implementation constraints and runtime rather than claims of universal latency superiority.

Finally, the reported objective is the maximum aggregate UBS task-completion latency inherited from the reference model, not per-frame MTP latency. Under the adopted benchmark parameters, this study therefore does not claim compliance with the 20 ms interactive-VR target. Future work should evaluate frame/tile-level MTP, realistic cache sizes and hardware power measurements, dynamic users and UAV trajectories, propulsion energy, and larger-scale multi-UAV deployments.

Acknowledgements

This work was supported by internal research funding from Telkom University under Research Grant No 10233/LIT06/PPM-LIT/2026.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Simulation Program

The code is publicly available at https://github.com/hilalnuha/vr_latency.

AI Usage Statement

The authors used generative AI tools, including Claude AI and ChatGPT, to assist with language refinement and code debugging. All mathematical derivations, experiments, numerical results, citations, and final manuscript content were reviewed and verified by the authors.

References

- [1] Bastug E, Bennis M, Médard M, Debbah M. Toward interconnected virtual reality: Opportunities, challenges, and enablers. *IEEE Commun Mag.* 2017; 55(6):110–7. doi:10.1109/MCOM.2017.1601089.
- [2] Chen M, Saad W, Yin C, Debbah M. Data correlation-aware resource management in wireless virtual reality: An echo state transfer learning approach. *IEEE Trans Commun.* 2019;67(6):4267–80. doi:10.1109/TCOMM.2019.2900624.
- [3] Nasir AA. Latency optimization of UAV-enabled MEC system for virtual reality applications under Rician fading channels. *IEEE Wirel Commun Lett.* 2021; 10(8):1633–7. doi:10.1109/LWC.2021.3075762.

- [4] Yuan H, Wang M, Bi J, Shi S, Yang J, Zhang J, et al. Cost-efficient task offloading in mobile edge computing with layered unmanned aerial vehicles. *IEEE Internet Things J.* 2024;11(19):30496–509. doi:[10.1109/JIOT.2024.3408216](https://doi.org/10.1109/JIOT.2024.3408216).
- [5] Liu J, Zhang Y, Wang J, Cui T, Zhang L, Li C, et al. Outage probability analysis for UAV-aided mobile edge computing networks. *EAI Endorsed Trans Ind Netw Intell Syst.* 2022;9(31):e4. doi:[10.4108/eetinis.v9i31.960](https://doi.org/10.4108/eetinis.v9i31.960).
- [6] Zarei M, Moshayedi AJ, Zhong Y, Khan AS, Kolahdooz A, Andani ME. Indoor UAV object detection algorithms on three processors: Implementation test and comparison. In: *2023 3rd International Conference on Consumer Electronics and Computer Engineering (ICCECE)*. Piscataway (NJ): IEEE; 2023. p. 812–9. doi:[10.1109/ICCECE58074.2023.10135199](https://doi.org/10.1109/ICCECE58074.2023.10135199).
- [7] Mazinani A, Norouzi M, Talaeizadeh A, Alasty A, Saadat Foumani M, Kolahdooz A. Innovative real-time palm tree detection, geo-localization and counting from unmanned aerial vehicle (UAV) aerial images using deep learning. *Automation.* 2026;7(2):51. doi:[10.3390/automation7020051](https://doi.org/10.3390/automation7020051).
- [8] Wang M, Moshayedi AJ, Lu H, Guo B, Bassir D. Integration of electronic noses and UAVs for intelligent gas detection: A review of methods and challenges. In: *Smart Manufacturing, Structural Health Monitoring and Digital Twins: Proceedings of the 1st International Conference (ICSSD 2025)*; Dongguan, China. SAGE Publications; 2025. p. 82–9. doi:[10.3233/ATDE251175](https://doi.org/10.3233/ATDE251175).
- [9] Moshayedi AJ, Khan AS, Geng K, Hu J, Kolahdooz A. Advancing agricultural practices: Analyzing the role of robotics in corn farming. *Int J Eng Trans A Basics.* 2025; 38(7):1517–32. doi:[10.5829/IJE.2025.38.07A.07](https://doi.org/10.5829/IJE.2025.38.07A.07).
- [10] You C, Zhang R. 3D trajectory optimization in Rician fading for UAV-enabled data harvesting. *IEEE Trans Wirel Commun.* 2019;18(6):3192–207. doi:[10.1109/TWC.2019.2911939](https://doi.org/10.1109/TWC.2019.2911939).
- [11] Nguyen CKP, Do TA, Nguyen TV, Pham HTT, Dang NT. Optimizing UAV trajectories in optical IRS-aided hybrid FSO/RF aerial access networks using DRL technique. *EAI Endorsed Trans Ind Netw Intell Syst.* 2025;12(4). doi:[10.4108/eetinis.124.10134](https://doi.org/10.4108/eetinis.124.10134).
- [12] Suyanto S, Ariyanto AA, Ariyanto AF. Komodo mlpir algorithm. *Appl Soft Comput.* 2022;114:108043. doi:[10.1016/j.asoc.2021.108043](https://doi.org/10.1016/j.asoc.2021.108043).
- [13] Setiawan D, Suyanto S, Erfianto B, Gozali AA. Battlefield optimization algorithm. *Expert Syst Appl.* 2025;266:126097. doi:[10.1016/j.eswa.2024.126097](https://doi.org/10.1016/j.eswa.2024.126097).
- [14] Al-Shaikhi A, Nuha HH, Mohandes M, Rehman S, Adrian M. Vertical wind speed extrapolation model using long short-term memory and particle swarm optimization. *Energy Sci Eng.* 2022;10(12):4580–94. doi:[10.1002/ese3.1291](https://doi.org/10.1002/ese3.1291).
- [15] Hosseinzadeh M, Tanveer J, Rahmani AM, Gharehchopogh FS, Yusof NM, Khoshvaght P, et al. A comprehensive overview of PSO-LSTM approaches: Applications, analytical insights, and future opportunities. *Arch Comput Methods Eng.* 2026; 33(4):5081–126. doi:[10.1007/s11831-025-10445-y](https://doi.org/10.1007/s11831-025-10445-y).
- [16] Daviran M, Maghsoudi A, Ghezelbash R. Optimized AI-MPM: Application of PSO for tuning the hyperparameters of SVM and RF algorithms. *Comput Geosci.* 2025;195:105785. doi:[10.1016/j.cageo.2024.105785](https://doi.org/10.1016/j.cageo.2024.105785).
- [17] Chandola J, Pundir S, Sharma A, Song Y, Fekete G, Singh T. Recent advances in MPPT techniques for photovoltaic systems: A review of classical (P&O, IC), intelligent (ANN), optimization (PSO) and hybrid (ANN-PSO) methods. *Results Eng.* 2026;29:109395. doi:[10.1016/j.rineng.2026.109395](https://doi.org/10.1016/j.rineng.2026.109395).
- [18] Chang X, Ye L, Ma L, Chen S. Research on hybrid path planning algorithms for UAVs in complex environments. *EAI Endorsed Trans Scalable Inf Syst.* 2025;12(3). doi:[10.4108/eetsis.8974](https://doi.org/10.4108/eetsis.8974).
- [19] Deng W, Xu H, Guan Z, Sun Y, Ran X, Ma H, et al. PSO-K-means clustering-based NSGA-III for delay recovery. *IEEE Trans Consum Electron.* 2025; 71(4):10084–95. doi:[10.1109/TCE.2025.3615937](https://doi.org/10.1109/TCE.2025.3615937).
- [20] Fauzan MN, Munadi R, Sumaryo S, Nuha HH. Enhanced grey wolf optimization for efficient transmission power optimization in wireless sensor network. *Appl Syst Innov.* 2025;8(2):36. doi:[10.3390/asi8020036](https://doi.org/10.3390/asi8020036).
- [21] Fauzan MN, Munadi R, Sumaryo S, Nuha HH. Enhanced wireless sensor network lifetime using EGWO-optimized neural network approach. *Network.* 2026;6(1):5. doi:[10.3390/network6010005](https://doi.org/10.3390/network6010005).
- [22] Nuha HH, Ariyanto E, Yulita IN, Sailallah HRP. Clustered packet optimization for human-powered energy generation prediction. *Discov Internet Things.* 2026;6(1):111. doi:[10.1007/s43926-026-00381-z](https://doi.org/10.1007/s43926-026-00381-z).
- [23] Zhu C, Byrd RH, Lu P, Nocedal J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans Math Softw.* 1997;23(4):550–60. doi:[10.1145/279232.279236](https://doi.org/10.1145/279232.279236).
- [24] Kennedy J, Eberhart R. Particle swarm optimization. In: *Proceedings of the IEEE International Conference on Neural Networks (ICNN'95)*; 1995 Nov 27–Dec 1; Perth, Australia. Piscataway (NJ): IEEE; 1995. p. 1942–8. doi:[10.1109/ICNN.1995.488968](https://doi.org/10.1109/ICNN.1995.488968).
- [25] Goldberg DE. Genetic algorithms in search, optimization and machine learning. Reading (MA): Addison-Wesley; 1989.