Available Online: 12 May 2026 DOI: 10.54254/2977-3903/2026.33548

Graph-enhanced hierarchical reinforcement learning for UAV-aided virtual reality

Zhengyu Zhao[1], Xingyu He[2*]

[1]School of Optical-Electrical & Computer Engineering, University of Shanghai for Science & Technology, Shanghai, China
[2]College of Publishing, University of Shanghai for Science & Technology, Shanghai, China

*Corresponding Author. Email: 1069035711@qq.com

Abstract. Existing Unmanned Aerial Vehicle (UAV)-aided virtual reality schemes overlook the coupling between the cost of real-time image acquisition and the freshness of rendered content, and fail to jointly optimize these two objectives. To address this issue, this paper proposes a freshness-aware hierarchical reinforcement learning method enhanced by graph attention to jointly optimize real-time image acquisition and rendering within a unified framework. We first construct a heterogeneous graph involving users, acquisition UAVs, and rendering UAVs, and then employ graph attention to model cached time, spatial distance, and UAV energy for enhanced state representation. The process is formulated as a two-layer Markov decision process: the high-level policy determines whether to perform real-time acquisition or cache-based rendering, while the low-level policy assigns optimal UAVs accordingly. Experimental results demonstrate the superiority of the proposed scheme over benchmark algorithms in terms of latency, energy consumption, and content freshness.

Keywords: UAV, virtual reality, hierarchical reinforcement learning, graph attention

1. Introduction

As Virtual Reality (VR) technology continues to advance, immersive experiences characterized by high fidelity and low latency have become critical requirements in diverse application scenarios, including entertainment and gaming, remote infrastructure inspection, disaster emergency response, and urban planning [1]. However, in wide-area dynamic environments, traditional VR service models relying on static modeling or offline pre-rendering struggle to address challenges arising from highly variable scenes, complex viewpoint demands, and content timeliness requirements, exhibiting significant limitations in real-time performance and flexibility.

In recent years, Unmanned Aerial Vehicle (UAV) swarms, distinguished by their high mobility, three-dimensional deployment, and flexible payload capacity, have been identified as a key technological enabler for overcoming the aforementioned bottlenecks [2, 3]. Through dynamic aerial deployment, UAVs can provide flexible environmental sensing capabilities and serve as mobile edge nodes participating in computational offloading and content distribution, offering novel architectural possibilities for VR services in wide-area

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/).

dynamic scenarios. Research on UAV-enabled VR primarily unfolds along two directions: (1) employing UAVs as aerial mobile sensing terminals to provide users with first-person dynamic perspectives through real-time image acquisition [4, 5]; and (2) utilizing UAVs as edge service nodes to enhance the efficiency of VR content rendering offloading through localized computation and caching capabilities [6, 7].

As real-time image acquisition and transmission devices, UAVs must address critical challenges including bandwidth fluctuations, end-to-end latency sensitivity, and real-time encoding and rendering of large-scale visual data. To tackle these issues, related research has explored multiple directions: Taleb et al. designed an immersive VR service framework for B5G networks, employing edge proxies to enhance the stability and interaction fluidity of panoramic video transmission [8]. Gao et al. proposed an onboard dynamic compression and data fragmentation mechanism for point cloud data under high data rate and link-constrained conditions, supporting real-time transmission and rendering over bandwidth-limited channels [9]. Yan et al. focused on multi-terminal collaborative scenarios, constructing a real-time video streaming system through the fusion of SLAM and AR rendering, enabling synchronous sharing and collaborative experiences of UAV-view content across multiple devices [10].

As edge caching and computational offloading nodes, UAVs must achieve efficient task allocation and content rendering under dynamic network topology changes, limited computational resources, and heterogeneous user demands. The core focuses of related research primarily encompass: Zhang and Chakareski approached from the perspective of three-dimensional deployment and content hierarchical caching, improving VR content accessibility and service efficiency through the optimization of UAV positions and caching structures [11]. Li et al. employed reinforcement learning and multi-agent collaborative mechanisms to investigate UAV hovering positions, service scheduling, and rendering mode selection strategies, achieving dynamic adaptation to multi-user interaction demands and computational resources [12]. Li et al. proposed a graph neural network-assisted deep reinforcement learning framework, leveraging GNNs to mine correlations in network state features to guide DRL action training, achieving continuous-domain flight control and scheduling decisions in unknown dynamic environments with significantly improved task offloading rates [13]. Yao et al. constructed a digital twin edge network to support policy training in virtual environments, introducing graph attention mechanisms to characterize inter-base station service request similarities and interference correlations, realizing intelligent decisions for collaborative task offloading and service caching, effectively enhancing QoE utility and system scalability [14].

Specifically, building upon the aforementioned research, existing approaches for UAV-enabled VR remote rendering primarily fall into two categories. The first category, involving real-time image acquisition followed by rendering, can effectively satisfy content timeliness requirements but typically relies on frequent UAV maneuvering and high-bandwidth transmission, resulting in significantly increased overall system latency and energy consumption overhead. The second category, performing rendering based on historical image caches from UAVs, reduces communication and energy costs to a certain extent by reusing historical cached content and edge computing power, yet struggles to cope with rapidly changing scenes and provides insufficient support for real-time constraints. Consequently, there exists a notable absence of frameworks that jointly model and optimize image acquisition UAVs and rendering UAVs within a unified architecture, with no prior awareness of the inherent correlation between real-time image acquisition costs and rendering cache timeliness.

To address this gap, we propose a graph attention-enhanced hierarchical reinforcement learning decision framework for the joint optimization of real-time image acquisition rendering and historical cache-based rendering. Specifically, the main contributions of this paper are as follows:

a) We propose a graph attention-based timeliness-aware state representation method that jointly encodes multi-source information including image cache time on rendering UAVs, spatial distances between rendering UAVs and users, and energy states of acquisition and rendering UAVs. Through the graph attention mechanism, we adaptively learn the real-time image acquisition task delay and energy consumption, historical cache-based rendering task delay and energy consumption, and content timeliness between regions and UAVs, thereby constructing structurally-aware enhanced state representations for both UAVs and grids, providing a reliable foundation for subsequent hierarchical reinforcement learning state inputs.

b) We construct a hierarchical reinforcement learning decision framework, where the high-level network classifies rendering tasks into either real-time image acquisition-based rendering or historical image cache-based rendering under the guidance of dual rewards considering timeliness and rendering costs, while the low-level network selects optimal executing UAVs for different rendering types.

2. Graph attention model

2.1. System model

System Roles. The system consists of M UAVs, N scene area nodes, and 1 user node. The k-th UAV is denoted as U_k, with constant attributes $\{v_{max}, E_{max}\}$, where v_{max} represents the maximum flight speed of UAV U_k, and E_{max} represents its maximum energy capacity. The i-th scene area node is denoted as S_i, obtained through mesh segmentation of the three-dimensional scene. The user node is denoted as U_s, with attribute position p_s.

2.2. Graph nodes and feature definitions

To characterize the spatial relationships, semantic relationships and historical interaction information among UAVs, task areas and users in the system, the heterogeneous graph constructed in this paper is shown in Figure 1. It consists of three types of nodes: UAV nodes, grid nodes and user nodes. The feature definition of each node is given as follows.

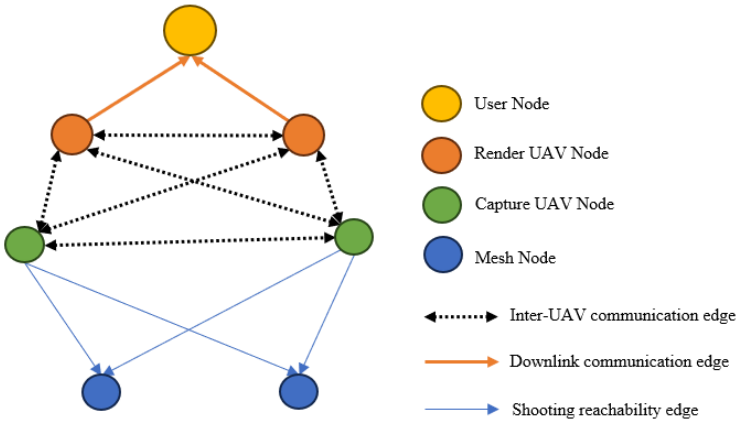


Figure 1. Heterogeneous graph

a) UAV Nodes $x_{U_k} = [p_k, v_k, E_k^{remain}, R_k^{comp}, B_k^{comm}, r_k]$. $p_k = (x_k, y_k, h_k)$ represents the position coordinates of UAV U_k, where x_k, y_k represent the horizontal coordinates of UAV U_k in the twodimensional plane, and h_k represents the flight altitude. $v_k = (v_k^x, v_k^y, v_k^h)$ represents the current three-dimensional

movement velocity of UAV $U_k \cdot E_k^{remain}$ represents the remaining available energy of UAV U_k at the current moment. R_k^{comp} represents the currently schedulable computing capability of UAV U_k . B_k^{comm} represents the available communication resources of UAV U_k at the current moment. r_k indicates whether the role of UAV U_k is acquisition or rendering.

b) Mesh Nodes $x_{S_i} = [c_i, s_i^{sem}, d_i^{render}, t_{i,k}^{last}, f_i^{vis}]$. c_i represents the geometric center position of mesh S_i in the 3D VR scene. s_i^{sem} represents the semantic label information of mesh $S_i \cdot d_i^{render}$ indicates the rendering computational load of this mesh S_i . t_i^{last} represents the time when mesh S_i was last updated by UAV U_k , i.e, the timestamp of the last update of mesh S_i by UAV U_k ,initialized to T_{max}. This implies that at the beginning of tasks, the content of this region is in a stale state and requires immediate updating. f_i^{vis} refers to the visual features extracted from the image or view corresponding to mesh S_i .

c) User Nodes. $x_{u_s} = [p_s]$. p_s represents the user position information.

2.3. Heterogeneous graph edge weight definitions

The heterogeneous graph contains four types of edges: UAV-UAV, UAV-mesh, mesh-mesh, and UAV-user. The weight of each edge is formed through spatial distance, communication quality, semantic similarity, and historical interaction relationships, with each edge type possessing unique characteristics.

a) UAV-UAV Edges. These are used to characterize the spatial proximity and link quality between UAV U_k and U_j. When constructing the edge weight between UAV U_k and U_j, this paper approaches from two aspects: spatial proximity $w_{k,j}^{spatial}$ and communication link quality $w_{k,j}^{comm}$. The spatial proximity $w_{k,j}^{spatial}$ between UAV U_k and U_j is specifically formulated as:

$$w_{k,j}^{spatial} = exp(-\frac{\|p_k - p_j\|^2}{2\sigma^2}) \cdot I(\|p_k - p_j\| \leq D_{max}) \tag{1}$$

The communication link quality $w_{k,j}^{comm}$ between UAV U_k and U_j is specifically formulated as:

$$w_{k,j}^{comm} = \frac{B}{R_{max}} log_2(1 + \frac{P_k h_{k,j}}{N_0 + I_j}) \tag{2}$$

b) UAV-Mesh Edges. These are used to characterize the acquisition relationship and historical task between UAV U_k and mesh S_i. This paper constructs UAV-mesh edge weights from two aspects: spatial distance $w_{k,i}^{dist}$ and historical interaction $w_{k,i}^{hist}$. The spatial distance $w_{k,i}^{dist}$ between UAV U_k and mesh S_i is specifically formulated as:

$$w_{k,i}^{dist} = exp(-\frac{\|p_k - c_i\|^2}{2\sigma^2}) \cdot I(\|p_k - c_i\| \leq D_{max}) \tag{3}$$

The historical interaction $w_{k,i}^{hist}$ between UAV U_k and mesh S_i is specifically formulated as:

$$w_{k,i}^{hist} = exp(-\alpha \Delta t_{k,i}) \cdot \frac{1}{2}(1 + \frac{f_{k,i}^{hist} \cdot f_i^{curr}}{\|f_{k,i}^{hist}\| \|f_i^{curr}\| + \epsilon}) \tag{4}$$

c) Mesh-Mesh Edges. $w_{i,j}^{semantic}$ is used to characterize the semantic similarity between mesh S_i and S_j, specifically formulated as:

$$w_{i,j}^{semantic} = s_i^{sem} \cdot s_j^{sem} \tag{5}$$

d) Attention Aggregation: Select the projection matrix W_{e_ϕ} according to the edge type ϕ , and calculate the attention coefficient:

$$e_{ij} = LeakyReLU(a^\top [Wx_i \| Wx_j \| W_{e_\phi} E_{ij}]) \tag{6}$$

After Softmax normalization, grid nodes aggregate the historical information of UAVs and neighbour semantic information to generate the enhanced representation z_i^S ; UAV nodes aggregate collaborative states and task requirements to generate z_k^U as the input state for HRL.

3. Graph attention-based hierarchical reinforcement learning model

The hierarchical reinforcement learning model proposed in this paper is illustrated in Figure 2. After generating the enhanced state representation z_{S_i} through the graph attention mechanism and feeding it into the hierarchical reinforcement learning framework, the high-level MDP aims to make task classification decisions for the target grid region between real-time capture and historical cached rendering based on the current environmental state. The low-level MDP is responsible for assigning appropriate UAVs to execute tasks according to the high-level task decisions, thereby achieving dual optimization of content timeliness requirements and system execution costs.

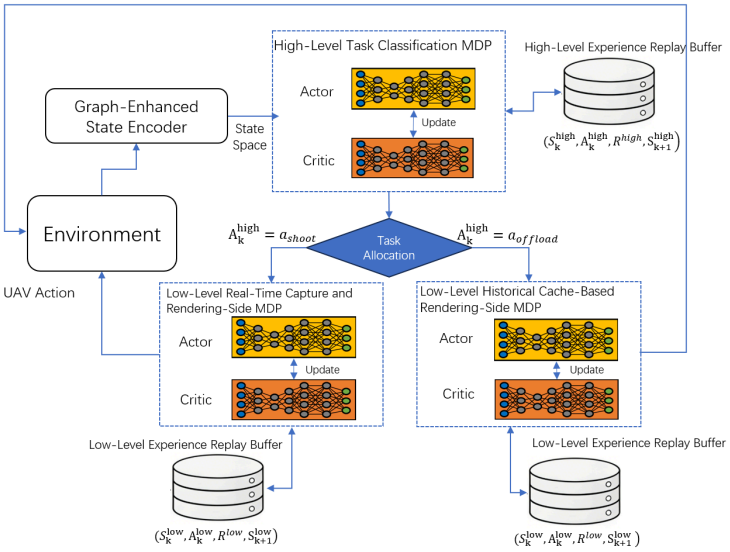


Figure 2. Hierarchical reinforcement learning model

3.1. High-level task classification MDP

a) High-Level State Space. The high-level state s_t^{high} needs to highly summarize grid features and system global capabilities. Using enhanced representations extracted by the graph attention network, it is defined as:

$$s_t^{high} = [z_{S_i} \| t_{i,k}^{last} \| d_i^{render}] \tag{7}$$

b) High-Level Action Space. The high-level policy π_{high} outputs a discrete binary action a_t^{high} to specify the service type of the current task:

$$a_t^{high} \in \{0, 1\} \tag{8}$$

$a_t^{high} = 1$ (Shoot) determines that the grid content is too stale, and the system needs to schedule physical entities to fly to the target point to acquire new data and return updates; $a_t^{high} = 0$ (Render) determines that the grid content does not require real-time acquisition, and the system needs to select a suitable UAV U_k to offload content to user u_s .

c) High-Level Reward Function. The design of high-level reward R_{high} follows the benefit-cost trade-off principle, maximizing the QoE improvement brought by content timeliness while penalizing excessively large system estimated costs caused by misclassification.

$$R_{high} = \underbrace{\mathbb{I}\left(a_t^{high}=1\right)\cdot\Phi(S_i)}_{\text{Acquisition Gain}} + \underbrace{\mathbb{I}\left(a_t^{high}=0\right)\cdot\Psi_{off}}_{\text{Render Gain}} \tag{9}$$

3.2. Low-level UAV scheduling strategy MDP

a) Low-Level State Space. The low-level state s_t^{low} needs to contain state information of all candidate UAVs U_k and attributes of the current target mesh S_i, in order to evaluate which UAV is most suitable for the task:

$$s_t^{low} = \{s_t^{high} \| U_{state} \| L_{state}\} \tag{10}$$

$U_{state} = [p_k, v_k, E_k^{rem}, R_k^{comp}]$ contains the three-dimensional position, velocity, remaining energy, and available computing power of each UAV U_k; L_{state} is the channel state information of all currently available links.

b) Low-Level Action Space. In shoot mode ($a_t^{high}=1$), both an image-capturing UAV and a rendering UAV need to be selected simultaneously, while in render mode ($a_t^{high}=0$), only the optimal rendering-serving UAV needs to be chosen.

$$a_{low} = \begin{cases} a_{low}^{shoot} = (u_{sen}, u_{rel}) & a_t^{high}=1, \\ a_{low}^{render} = u_{rel} & a_t^{high}=0. \end{cases} \tag{11}$$

c) Low-Level Reward Function. Covers the full process costs from UAV maneuvering, data acquisition to multi-hop transmission:

$$\begin{cases} R_{low}^{shoot} = -(\omega_t \cdot T_{total}^{shoot} + \omega_e \cdot E_{total}^{shoot}) a_t^{high}=1, \\ R_{low}^{render} = -(\omega_t \cdot T_{total}^{render} + \omega_e \cdot E_{total}^{render}) a_t^{high}=0. \end{cases} \tag{12}$$

4. Experiments

To verify the effectiveness of the proposed method, we constructed a simulation environment on the Python 3.8 platform, with the deep learning module implemented based on PyTorch 1.12. To comprehensively evaluate the performance of the proposed method, we designed six key evaluation metrics to characterize system performance across different dimensions.

a) Weighted Average End-to-End Latency $\overline{D}_{weighted}$. This metric comprehensively considers the impact of task priority differences on system response performance, with mathematical definition:

$$\overline{D}_{weighted} = \frac{1}{\sum_{k=1}^{K} w_k} \sum_{k=1}^{K} w_k (D_k^{fly} + D_k^{proc} + D_k^{trans}) \tag{13}$$

b) Normalized System Energy Efficiency η_{energy}, defined as the effective task throughput completed per unit energy consumption:

$$\eta_{energy} = \frac{\sum_{k=1}^{K} I(D_k \le D_k^{req})}{\sum_{M=1}^{M} (E_u^{fly} + E_u^{comp} + E_u^{comm})/E_0} \tag{14}$$

c) Peak Age of Information (PAoI), used to characterize the worst-case content timeliness:

$$PAoI = max_{i \in \mathscr{R}} \left\{ max_{t \in [0,T]} \left(t - t_i^{update}(t) \right) \right\} \tag{15}$$

d) UAV Load Balance Index (LBI), measuring the balance degree of task allocation among UAVs.

$$LBI = 1 - \frac{\sqrt{\frac{1}{M}\sum_{M=1}^{M}(L_u - \overline{L})^2}}{\overline{L}} \tag{16}$$

e) Task Success Rate (TSR), defined as the proportion of tasks successfully completed within latency constraints:

$$TSR = \frac{1}{K}\sum_{k=1}^{K} I(D_k \le D_k^{req} \wedge Q_k \ge Q_k^{min}) \tag{17}$$

f) Average Content Freshness $\overline{F}_{content}$, measuring the timeliness of VR content received from the user perspective:

$$\overline{F}_{content} = \frac{1}{K}\sum_{k=1}^{K} e^{-\beta(t_k^{recv} - t_{rk}^{capture})} \tag{18}$$

4.1. Parameters setting

The experiment simulates a 1,000 × 1,000 × 200 meter three-dimensional urban scene airspace. To simulate a heterogeneous UAV network, the system deploys 6 UAVs, including 3 acquisition UAVs and 3 rendering UAVs. Specific experimental environment configurations, UAV parameters, and network hyperparameters are shown in Table 1.

Table 1. Experiment parameters setting

Parameter	value
UAV Number M	6
UAV v_{max}	10 m/s
Layers / Heads	3, 4
α, β	0.3, 0.3
C_{comp}	5 units/s
ω_t, ω_e	0.5, 0.5
Batch size	128
Episodes	1,000
Learning rate	10-4
γ	0.97

4.2. Simulation results

4.2.1. Performance analysis

Figure 3 shows the impact of different learning rates on various performance metrics. When the learning rate is set to $\alpha = 10^{-5}$, due to the small parameter update step size, the training convergence speed is relatively slow, and the ultimately achievable performance is slightly worse than the baseline configuration. When the learning rate is increased to the baseline value $\alpha = 3 \times 10^{-4}$, the system can reach optimal performance, with balanced and stable performance across all metrics. If the learning rate is further increased to $\alpha = 10^{-3}$, performance decreases. When the learning rate is increased to $\alpha = 3 \times 10^{-3}$, performance deteriorates, mainly because the excessively large learning rate makes parameter updates too drastic, causing strong training oscillations and making it difficult to converge to the global optimum.

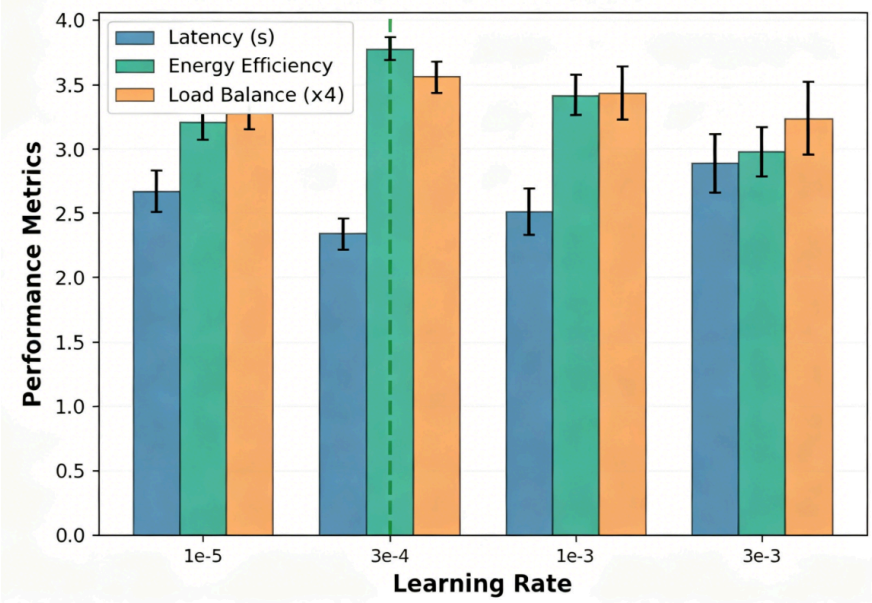


Figure 3. Impact of learning rate on various performance metrics

Figure 4 presents the impact of the discount factor on normalized performance. This parameter reflects the algorithm's trade-off between long-term and short-term rewards. When the discount factor is set to $\gamma = 0.90$, the smaller discount factor causes the policy to prioritize satisfying immediate tasks. When the discount factor is increased to the baseline value $\gamma = 0.95$, the algorithm achieves a relatively good balance between short-term response and long-term optimization, with optimal performance across all metrics. When the discount factor is further increased to $\gamma = 0.98$, performance improves, but when the discount factor increases to $\gamma = 0.99$, performance improvement begins to saturate, and some metrics even slightly decrease. This is because excessive emphasis on long-term rewards may make the policy overly conservative, unable to comprehensively consider all metrics. Therefore, discount factors in the range $[0.95, 0.98]$ can achieve the most ideal overall performance.

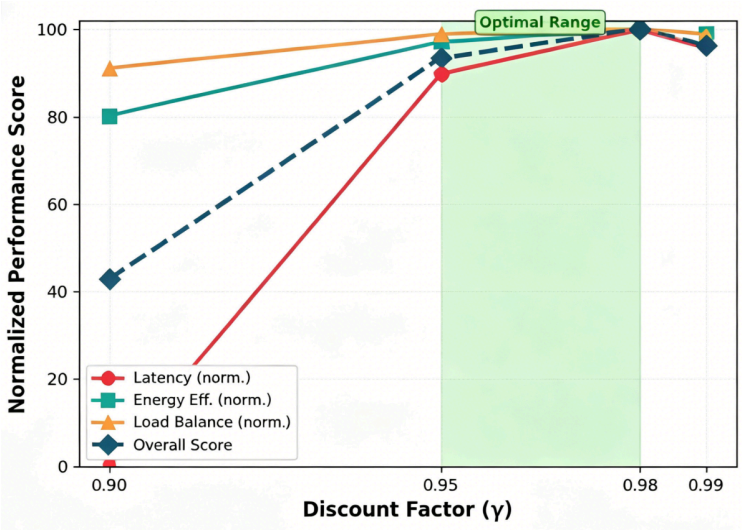


Figure 4. Impact of discount factor on normalized performance

4.2.2. Ablation performance

Figure 5 illustrates the percentage of performance degradation caused by removing individual modules.

Analysis across latency, energy efficiency, PAoI, load balance, task success rate and content freshness shows that omitting the graph attention mechanism most severely impairs freshness-related metrics, since GAT explicitly models historical capture time and content similarity for effective freshness awareness.

Removing the hierarchical structure notably degrades energy efficiency and load balancing, as the hierarchical framework excels at resource allocation and long-term planning.

Even the simplified RL-based baseline achieves limited optimization, but its convergence speed and final performance are far worse than the full model.

The version without GAT and hierarchy needs about 3.2 times more iterations to reach only 70% of the full-model performance, which validates the effectiveness and necessity of the proposed scheme.

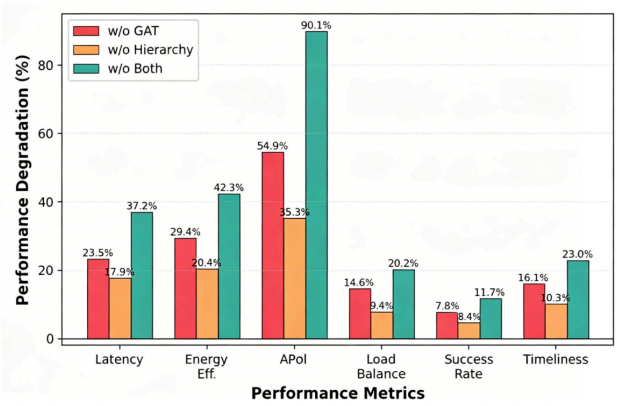


Figure 5. Performance degradation caused by the absence of each module

4.2.3. Comparative analysis

To validate the advantages of the proposed method, we select three benchmark algorithms for comparison:

The first is a Deep Reinforcement Learning (DRL)-based VR edge rendering approach [12], which employs multi-agent deep reinforcement learning to optimize UAV hovering positions and rendering decisions. However, it does not consider real-time capture tasks and merely focuses on the allocation of edge computing resources.

The second is a cache-optimization-based UAV-assisted VR service method [7], which improves service efficiency by optimizing the three-dimensional deployment positions of UAVs and content caching strategies. Nevertheless, relying on an offline pre-caching mechanism, it faces considerable difficulty in addressing content timeliness requirements under dynamic scenarios.

The third is a greedy strategy-based task scheduling method, which selects the UAV with the shortest distance to the target area and the highest residual energy to execute tasks at each decision moment. Serving as a heuristic baseline algorithm, we additionally introduce a random strategy as a lower-bound reference, where UAVs and task types are randomly selected.

As shown in Figure 6, the proposed method significantly outperforms other algorithms in terms of latency, energy utilization, and age of information performance metrics.

Regarding latency performance, DRL-VR performs reasonably well in offloading decisions, but due to the lack of photography module support, it can only rely on pre-stored data in content stale regions, resulting in high overall response latency. Cache-Opt's offline pre-caching mechanism performs poorly when scenes change frequently, with frequent cache updates generating additional latency overhead. The Greedy strategy

lacks a global optimization perspective, easily causing some UAVs to be overloaded while others remain idle, resulting in resource waste.

Regarding energy performance, Cache-Opt can use pre-caching to reduce real-time computation overhead in some cases, but its three-dimensional deployment optimization generally requires UAVs to perform large-scale maneuvering adjustments, thereby increasing flight energy consumption. As for DRL-VR, since it does not uniformly model photography tasks, it can only rely on external data sources when stale content needs updating, failing to fully utilize onboard acquisition capabilities of UAVs. The Greedy strategy has the lowest energy efficiency because its myopic decision mechanism leads to frequent invalid flights and energy waste.

Regarding PAoI performance, the main reason for DRL-VR is its lack of content timeliness perception mechanisms. During task scheduling, it does not distinguish between fresh and stale content, causing some regions to remain unupdated for long periods. Cache-Opt and Greedy methods struggle to ensure global content timeliness in dynamic scenarios, with significant service quality deficiencies in worst-case situations.

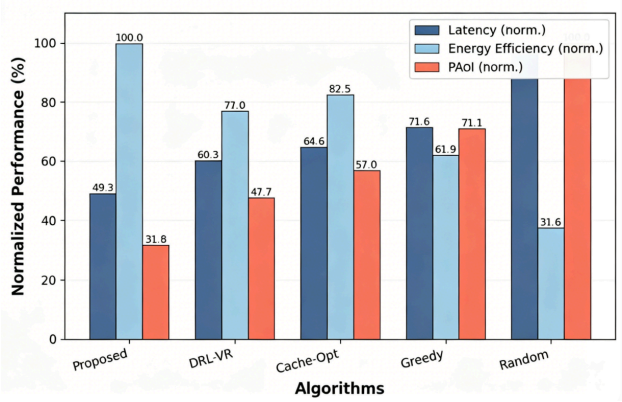


Figure 6. Performance comparison of different algorithms

5. Conclusion

This paper proposes a graph attention network-based hierarchical reinforcement learning joint decision-making method for UAV VR acquisition and edge rendering problems. By employing graph attention mechanisms to model mesh semantic features and historical correlations with UAVs, structure-aware enhanced state representations are constructed to provide reliable state inputs for subsequent decision-making. Furthermore, a high-low level collaborative Markov decision process is constructed, where the high-level policy globally balances content timeliness, system load, and execution costs, adaptively selecting acquisition or edge rendering modes, and the low-level policy, under high-level instructions, completes UAV task allocation through differentiated sub-policy networks, improving system latency and energy consumption. Simulation results show that this method possesses good stability and adaptability in various dynamic scenarios, effectively improving system performance while ensuring VR content timeliness, providing a feasible optimization solution for complex collaborative services of UAV swarms.

References

[1] Tham, J. (2024). Researching with virtual reality: Exploring the methodological affordances of VR for sociotechnical research and implications for technical and professional communication. *IEEE Transactions on Professional Communication, 67*(2), 192–210.

[2] Javed, S., Hassan, A., Ahmad, R., Ahmed, W., Ahmed, R., & Saadat, A. (2024). State-of-the-art and future research challenges in UAV swarms. *IEEE Internet of Things Journal, 11*(11), 19023–19045. https: //doi.org/10.1109/JIOT.2024.3364230

[3] Sheraz, M., Chuah, T. C., Lee, Y. L., Alam, M. M., Al-Habashna, A., & Han, Z. (2024). A comprehensive survey on revolutionizing connectivity through artificial intelligence-enabled digital twin network in 6G. *IEEE Access, 12*, 49184–49215. https: //doi.org/10.1109/ACCESS.2024.3384272

[4] Yan, L., Wang, Q., Zhao, J., Guan, Q., Tang, Z., Zhang, J., & Liu, D. (2025). Radiance field learners as UAV first-person viewers. In A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, & G. Varol (Eds.), *Computer Vision – ECCV 2024* (Lecture Notes in Computer Science, Vol. 15119, pp. 88–107). Springer. https: //doi.org/10.1007/978-3-031-73030-6_6

[5] Wang, Y., Ho, I. W. H., & Wang, Y. (2023). Real-time intersection vehicle turning movement counts from live UAV video stream using multiple object tracking. *Journal of Intelligent and Connected Vehicles, 6*(3), 149–160.

[6] Yang, X., Wang, Q., Yang, B., & Cao, X. (2025). Energy-efficient aerial STAR-RIS-aided computing offloading and content caching for wireless sensor networks. *Sensors, 25*(2), Article 393. https: //doi.org/10.3390/s25020393

[7] Yoo, S., Jeong, S., Kim, J., & Kang, J. (2024). Cache-assisted mobile-edge computing over space–air–ground integrated networks for extended reality applications. *IEEE Internet of Things Journal, 11*(10), 18306–18319. https: //doi.org/10.1109/JIOT.2024.3361907

[8] Taleb, T., Sehad, N., Nadir, Z., & Song, J. (2023). VR-based immersive service management in B5G mobile systems: A UAV command and control use case. *IEEE Internet of Things Journal, 10*(6), 5349–5363. https: //doi.org/10.1109/JIOT.2022.3222282

[9] Gao, Y., Wang, W., Chen, N., Ren, W., & Ye, Q. (2025). Real-time mobile transmission and rendering of UAV LiDAR massive point cloud. *IET Radar, Sonar & Navigation*, Article e70095. https: //doi.org/10.1049/rsn2.70095

[10] Yan, L., Fukuda, T., & Yabuki, N. (2019). Integrating UAV development technology with augmented reality toward landscape tele-simulation. In *Proceedings of the 24th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA)* (pp. 423–432). CAADRIA.

[11] Zhang, L., & Chakareski, J. (2022). UAV-assisted edge computing and streaming for wireless virtual reality: Analysis, algorithm design, and performance guarantees. *IEEE Transactions on Vehicular Technology, 71*(3), 3267–3275.

[12] Li, Z., Liang, X., Liu, J., He, X., Xie, L., & Qu, L. (2025). Optimizing mobile-edge computing for virtual reality rendering via UAVs: A multiagent deep reinforcement learning approach. *IEEE Internet of Things Journal, 12*(17), 35756–35772. https: //doi.org/10.1109/JIOT.2025.3580468

[13] Li, K., Ni, W., Yuan, X., Noor, A., & Jamalipour, A. (2023). Exploring graph neural networks for joint cruise control and task offloading in UAV-enabled mobile edge computing. In *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)* (pp. 1–6). IEEE. https: //doi.org/10.1109/VTC2023-Spring57618.2023.10200966

[14] Yao, Z., Xia, S., Li, Y., & Wu, G. (2023). Cooperative task offloading and service caching for digital twin edge networks: A graph attention multi-agent reinforcement learning approach. *IEEE Journal on Selected Areas in Communications, 41*(11), 3401–3413. https: //doi.org/10.1109/JSAC.2023.3310080