Journal of Network and Computer Applications 241 (2025) 104213

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Research paper

Optimizing disaster response with UAV-mounted RIS and HAP-enabled edge computing in 6G networks

Jamal Alotaibi [a,*], Omar Sami Oubbati [b,1], Mohammed Atiquzzaman [c,2], Fares Alromithy [d], Mohammad Rashed Altimania [d]

[a] *Department of computer engineering, College of Computer, Qassim University, Buraydah, Saudi Arabia*
[b] *LIGM, University Gustave Eiffel, Marne-la-Vallée, France*
[c] *University of Oklahoma, Norman, OK, USA*
[d] *Electrical engineering department, University of Tabuk, Tabuk, Saudi Arabia*

ARTICLE INFO

Keywords:
6G
Edge computing
Reconfigurable intelligent surfaces (RIS)
UAV
Wireless energy transfer (WET)
Disaster response

ABSTRACT

In the context of disaster response and recovery within 6th Generation (6G) networks, achieving both low-latency and energy-efficient communication under compromised infrastructure remains a critical challenge. This paper introduces a novel framework that integrates a solar-powered High-Altitude Platform (HAP) with multiple Unmanned Aerial Vehicles (UAVs) equipped with Reconfigurable Intelligent Surfaces (RISs), significantly enhancing disaster response capabilities. A hybrid approach combining game theory and multi-agent reinforcement learning (MARL) is employed to optimize UAV energy management, RIS control, and the offloading data rates of ground devices (GDs). Specifically, game theory is used to determine optimal task offloading decisions, balancing energy consumption, latency, and computational efficiency, while MARL dynamically guides UAV trajectories and RIS configurations to maintain robust communication links. A key innovation is the RIS ON/OFF mechanism, which conserves energy by switching OFF RISs when not needed, allowing UAVs to recharge during inactive periods and extending operational lifetimes. The proposed framework also demonstrates superior performance in optimizing offloading data rates and minimizing task offloading costs, ensuring efficient resource utilization. Extensive simulations validate the effectiveness of this approach, showing significant improvements in energy efficiency, data processing performance, and overall network reliability compared to traditional methods. These advancements contribute to more reliable and energy-efficient disaster response operations within 6G networks.

1. Introduction

In disaster scenarios, ensuring reliable and energy-efficient communication within 6G networks is challenging, especially when ground infrastructure is disrupted by natural disasters like earthquakes or hurricanes. Mobile devices, such as smartphones and smart bracelets, are essential for real-time data processing and emergency communication (Dai et al., 2022), but their limited processing power and battery life hinder their effectiveness in compromised environments. Edge computing, which minimizes latency by processing data closer to the source, is vital in these cases (Liu et al., 2022b). When ground networks are unavailable, High Altitude Platforms (HAPs) and Unmanned Aerial Vehicles (UAVs) play a key role in restoring connectivity and supporting rescue efforts (He et al., 2022). UAVs provide flexible coverage, while HAPs offer sustained, wide-area communication and serve as aerial edge computing platforms. For instance, in 2020, Loon used HAPs to deliver internet to underserved areas in Kenya (Caswell, 2015), and in 2021, Airbus showcased the solar-powered Zephyr HAP for direct-to-device connectivity in rural Arizona (Gavan et al., 2009). While HAPs enhance network resilience and resource management, further optimization is required to ensure energy-efficient communication over extended periods.

Power-constrained mobile devices, such as smartphones and tablets, used by rescue workers or victims in disaster situations, face significant limitations in executing computationally intensive tasks (Liwang and

[*] Corresponding author.
E-mail addresses: j.alotabi@qu.edu.sa (J. Alotaibi), omar-sami.oubbati@univ-eiffel.fr (O.S. Oubbati), atiq@ou.edu (M. Atiquzzaman), falromithy@ut.edu.sa (F. Alromithy), moh-doshan@ut.edu.sa (M.R. Altimania).
[1] Member, IEEE.
[2] Senior Member, IEEE.

https://doi.org/10.1016/j.jnca.2025.104213
Received 11 December 2024; Received in revised form 2 April 2025; Accepted 5 May 2025
Available online 21 May 2025

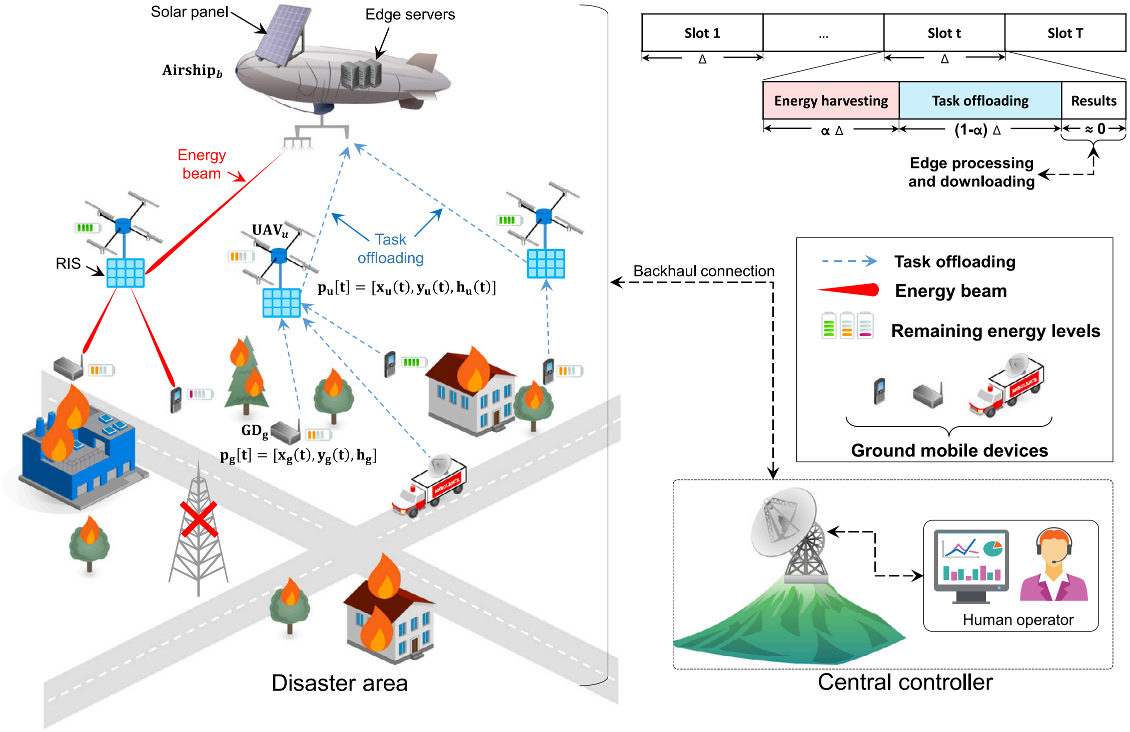


Fig. 1. Motivating scenario of our framework.

Wang, 2022). Tasks like hazard detection, geospatial analysis, multilingual processing, and machine learning for predictive analytics are crucial in disaster response (Ramakrishnan et al., 2022). However, due to their limited processing power, memory, and energy resources, it is impractical for these devices to handle such tasks locally, especially under time-sensitive conditions. To overcome this, HAPs have been proposed as aerial edge computing servers, offloading complex tasks from mobile devices to more powerful remote resources (Zhao et al., 2023). While HAPs enhance computational capabilities and ensure communication during infrastructure failures, challenges remain concerning energy consumption and latency experienced by mobile ground devices (GDs) when transmitting data over long distances to HAPs. UAVs play a crucial role in disaster recovery communication, as emphasized by Esposito et al. (2019), who use game theory to optimize user association in emergency networks, balancing communication quality and energy efficiency. However, their approach primarily focuses on communication quality and user association, without fully addressing energy inefficiency in UAV networks, leading to limited operational duration and reduced network reliability. Similarly, Wang et al. (2022) tackle task offloading for post-disaster rescue through fog computing, addressing UAV computational limitations. However, their approach does not sufficiently mitigate latency, particularly in task transmission between UAVs and GDs, resulting in delayed decision-making that can hinder the timely execution of disaster recovery operations. Both works highlight UAVs' importance but fail to fully address the twin challenges of energy inefficiency and latency. These gaps motivate our proposed system, which integrates UAVs and HAPs to optimize communication and task management. By incorporating technologies like Reconfigurable Intelligent Surfaces (RISs), our system reduces both energy consumption and latency, ensuring more efficient resource utilization and faster communication in disaster recovery operations.

In this work, we propose a novel framework that leverages a HAP deployed as a solar-powered airship operating at high altitudes, serving as a central communication hub with integrated edge computing capabilities (*c.f.,* Fig. 1). The HAP is equipped with solar panels to harvest solar energy, enabling it to remain aloft and operational for several weeks (Alam et al., 2021). In this framework, multiple UAVs are mounted with RISs, which act as dynamic nodes between the HAP and GDs. The UAV-mounted RISs relay offloaded tasks from GDs to the HAP and processed tasks back to GDs, while also redistributing energy through the HAP's Wireless Energy Transfer (WET) capability to ensure continuous operation. The HAP coordinates task aggregation, using game theory to optimize task offloading for GDs. A Multi-agent Reinforcement Learning (MARL) strategy based on a Deep Reinforcement Learning (DRL) method is also employed to manage UAV movements and ensure optimal coverage. To tackle energy inefficiency and latency, our framework integrates UAVs, HAPs, and RISs to optimize both energy usage and communication. UAV-mounted RISs dynamically adjust to minimize energy consumption when not covering GDs, and leverage the HAP's WET for efficient recharging. MARL and game theory further optimize UAV movements and task offloading, reducing communication delays between UAVs, HAPs, and GDs. This integrated system addresses both energy inefficiency and latency, ensuring faster, more efficient disaster recovery. The primary contributions of this work are:

- Developing an innovative system architecture that integrates UAVs and HAPs for efficient edge computing in disaster scenarios. The framework uses game theory to optimize GD task offloading decisions.
- Proposing a novel RIS ON/OFF mechanism, enabling UAVs to switch OFF their RIS components and recharge their batteries when not covering GDs or when their energy levels fall below a specific threshold, significantly enhancing the energy efficiency and operational longevity of the UAV network. This mechanism is further optimized by employing MARL to intelligently guide UAV movements and manage RIS states. DRL enables UAVs to autonomously optimize movements and task offloading in dynamic disaster recovery scenarios. By adapting to real-time conditions, DRL balances objectives like energy efficiency, task completion, and communication quality. This approach eliminates the need for pre-programmed rules, providing an adaptive solution that continuously improves energy inefficiency and latency challenges.

- Conducting extensive simulations to validate the proposed framework, demonstrating substantial improvements in both energy efficiency and task processing speed compared to existing approaches.

The structure of this paper is organized as follows: Section 2 presents a thorough review of related works. Section 3 details the system's core elements. Section 4 outlines the formulation of problems and the various optimization solutions adopted. Section 5 showcases the simulation outcomes of our solution. Finally, Section 6 concludes our work.

2. Related work

The growing reliance on UAVs and HAPs for edge computing, data collection, and communication in challenging environments, particularly during disaster response, has spurred significant research activities (McEnroe et al., 2022). These activities can be categorized into three main areas: (i) HAP-based edge computing and energy transfer, (ii) UAV-mounted RIS for data collection, and (iii) Reinforcement Learning (RL)-based techniques applied to UAV-assisted networks. This section reviews notable contributions in each of these areas, highlighting key advancements and remaining challenges.

2.1. HAP-based edge computing and energy transfer

Combining HAPs with edge computing capabilities has proven to be an effective solution for overcoming the computational and power limitations of mobile GDs. In Ding et al. (2021), the authors introduced a satellite–aerial integrated edge computing network (SAIECN) that leverages both LEO satellites and HAPs to provide edge computing services to ground user equipment (GUE). They minimized energy consumption by optimizing GUE association, computation task assignment, and resource allocation. In Wang et al. (2024), an aerial hierarchical Mobile Edge Computing (MEC) system with HAPs and UAVs is proposed to address the needs of IoT connectivity and computing. They formulated a task offloading problem to minimize long-term processing costs while managing task queueing. Kim (2024) deployed an aerial network for providing edge computing services to IoT devices in underserved areas, proposing a computation resource allocation algorithm using game theory. This work aims to balance service requirements through interactive learning. In Zhang et al. (2022b), the authors explored integrating laser-beamed WET into HAP-aided MEC systems for aerial user equipment, proposing a multitier tile grid-based spatial structure to partition the 3-D coverage area of the HAP for laser charging, aiming to enhance overall system utility. The work in Li et al. (2024) proposed a joint optimization approach for computation offloading and resource allocation in air–ground integrated vehicular edge computing (VEC) networks involving HAPs. Their framework minimizes task offloading delays by optimizing computation equipment selection, UAV trajectory, resource allocation, and HAP usage. Using a block coordinate descent (BCD) method, the authors demonstrated significant reductions in task offloading delays through simulations, while efficiently utilizing HAPs for aerial edge computing and communications.

2.2. UAV-mounted RIS for data collection

The UAV-mounted RIS design has garnered interest in enhancing data collection efficiency within IoT networks. In Tyrovolas et al. (2022), the authors proposed a UAV-mounted RIS-based approach for data collection in IoT networks and evaluated its performance in terms of coverage and average data collected. To improve data collection, they introduced an innovative MAC-layer protocol incorporating slotted ALOHA and code combining. Qi et al. (2024) proposed a UAV-RIS-assisted vehicular system, where a UAV acts as a mobile relay and RIS carrier. They sought to optimize the RIS phase shifts, spectrum allocation, and UAV trajectory to minimize the average Age of Information (AoI) while accounting for UAV energy constraints. A UAV-mounted RIS is deployed in Eskandari et al. (2022) to establish uninterrupted Line-of-Sight (LoS) communication links for intelligent vehicles in dense urban areas. The research tackled the NP-hardness and nonconvexity involved in planning an optimal UAV trajectory, taking into account UAV motion and LoS constraints. In Samir et al. (2021), a UAV embedded with RIS is adopted to relay data from IoT devices to a base station. The authors developed an optimization problem aimed at minimizing the expected sum AoI, focusing on optimizing UAV altitude, communication scheduling, and RIS phase shifts. The authors of Li et al. (2023) developed a flying-RIS-assisted MEC system to assist with computation offloading for IoT devices. The work focuses on the optimization of UAV trajectory planning, phase shift design for RIS, and IoT devices' association to minimize energy consumption and latency. The authors employed a deep deterministic policy gradient (DDPG)-based algorithm to solve the complex non-convex optimization problem, demonstrating the superiority of their approach in simulation results, where the proposed method significantly outperformed other benchmark schemes.

2.3. RL-based techniques for UAV-assisted networks

The application of RL techniques in UAV-assisted networks has shown promising results in addressing energy efficiency and adaptive communication strategies and being quickly adapted to unknown environments. The authors of Yang et al. (2024) proposed a UAV-RIS-assisted maritime communication system to improve energy efficiency and service quality against jamming attacks. They also proposed an adaptive energy harvesting scheme to enhance UAV endurance. Yang et al. (2023) proposed a UAV-RIS-assisted communication system to maximize the secrecy rate and ensure Quality of Service (QoS) against eavesdroppers and jammers, optimizing transmit beamforming, artificial noise, UAV-RIS placement, and RIS passive beamforming. In Duo et al. (2023), the researchers introduced an aerial RIS-supported computing approach to overcome the limitations of ground-based RIS-assisted MEC networks. To optimize energy efficiency and service quality, they jointly optimized UAV trajectories, RIS phase shift, offloading strategy, and resource allocation. The authors of Zhang et al. (2022a) analyzed the downlink of UAV networks utilizing RIS-equipped non-orthogonal multiple access (NOMA). This work aims to optimize UAV paths and RIS phase adjustments to maximize system capacity within energy limitations. In Lee and Kim (2023), the authors explored a MARL framework to manage offloading ratios and UAV trajectories in a multi-UAV mobile-edge computing network. The authors aimed to minimize task offloading delay and maximize the number of offloaded tasks through intelligent coordination of UAVs using MARL. Their work highlighted how multiple UAVs could dynamically adjust their offloading ratios and trajectories to enhance network efficiency, and the proposed method achieved a reduction in offloading delays, demonstrating the potential of MARL in UAV-assisted MEC networks.

Despite the significant advancements in integrating HAPs, UAV-mounted RIS, and RL techniques for assisting terrestrial networks, several critical challenges remain. Firstly, existing HAP-based edge computing and energy transfer solutions often fail to optimize deployment strategies of HAPs to minimize round-trip times (*i.e.,* latency) between GDs and HAP-based edge computing nodes. This latency issue can severely impact performance, particularly in time-sensitive applications like disaster response. Secondly, while UAV-mounted RIS approaches have potential in enhancing data collection, these solutions are tailored to small-scale deployments and do not address scalability for large networks with high moving device densities. Thirdly, RL-based techniques, although effective in improving network performance, typically use single-agent frameworks and discretized action spaces, leading to higher policy errors and reduced adaptability in dynamic real-world

Table 1
Comparative analysis of contributions.

		Lat.	Scal.	OT	Objective	Advantage	Drawback
HAP-based Edge Computing	Ding et al. (2021)	×	√	Multiple optimization techniques	Minimizing energy consumption by optimizing GUE association, computation task assignment, and resource allocation.	Efficiently integrates LEO satellites and HAPs to provide edge computing services.	Does not fully optimize mobility strategies to minimize latency.
	Zhang et al. (2022b)	×	√	Game theory	Improving system-wide expected total utilities by integrating laser-beamed WET into HAP-aided MEC systems.	Enhances the efficiency of energy transfer and edge computing capabilities.	High complexity and potential latency issues.
	Li et al. (2024)	×	√	BCD (Block Coordinate Descent)	Joint optimization of computation offloading and resource allocation in air–ground integrated vehicular edge computing with HAPs.	Efficient resource allocation and offloading delay minimization using HAPs for aerial edge computing.	High complexity in solving non-convex optimization problems.
UAV-mounted RIS	Tyrovolas et al. (2022)	√	×	MAC-layer Protocol	Analyzing performance in terms of coverage and average collected data to enhance data collection.	Improves energy efficiency and data collection performance.	Limited scalability for large-scale IoT networks.
	Qi et al. (2024)	√	×	Multistep Dueling Double Deep Q Network (DQN)	Minimizing the average AoI while considering UAV energy constraints.	Reduces AoI and enhances data freshness.	Does not address scalability and large-scale deployments.
	Li et al. (2023)	√	×	Multistep Deep Q Learning	Optimizing UAV trajectory planning and phase shift design for RIS-assisted MEC networks.	Enhances system performance by optimizing UAV movement and RIS phase adjustments.	Limited scalability and high complexity in large-scale deployments.
RL-based Techniques	Yang et al. (2024)	×	√	Softmax Deep Double Deterministic Policy Gradients	Improving energy efficiency and service quality against jamming attacks.	Enhances UAV endurance through adaptive energy harvesting.	Complexity in adapting to dynamic and unknown environments.
	Duo et al. (2023)	×	√	Double DQN	Addressing limitations of the system by optimizing UAV trajectories and RIS phase shifts.	Optimizes energy efficiency and service quality.	High complexity and potential latency issues.
	Lee and Kim (2023)	√	√	Multi-Agent Reinforcement Learning (MARL)	Optimizing UAV trajectories and task offloading in multi-UAV systems for edge computing.	Efficient multi-agent coordination, task allocation, and communication management in large-scale networks.	Complexity in multi-agent coordination and implementation in large networks.
Our proposed solution		√	√	Multi-Agent RL	Optimizing the trajectories and actions of UAVs and HAPs to minimize latency and improve scalability.	Efficiently integrates HAPs and UAVs for enhanced edge computing and computing task collection in large-scale IoT environments.	Complexity in implementation and system architecture.

Lat.: Measures how the solution addresses latencies, Scal.: Capability to handle scalability, OT: The used optimization technique.

scenarios. As a result, our work aims to bridge these gaps by proposing a system that integrates HAP-based edge computing with energy transfer capabilities and optimizes UAV-mounted RIS configurations for dynamic and unknown GDs deployed in disaster-prone areas and employs MARL strategies to enhance adaptability and performance. Furthermore, our work employs MARL and game theory approaches that optimize UAV movements and task offloading decisions, minimizing round-trip times by ensuring that data transmission paths are dynamically adjusted for optimal communication. These decentralized approaches enable UAVs to collaborate in real-time to reduce delays and improve the overall efficiency of task offloading, ensuring timely data transmission between GDs and HAPs.

A comparative analysis of our proposed system against the most relevant existing solutions is presented in Table 1, highlighting the uniqueness of our approach.

3. System model

As shown in Fig. 1, we design an edge computing system comprising three main components: (i) a solar-powered HAP, denoted as Airship$_b$, (ii) a set of dynamic UAVs, denoted as $\mathcal{U} \triangleq \{u = 1,2,\dots,U\}$, and (iii) various mobile GDs, denoted as $\mathcal{G} \triangleq \{g = 1,2,\dots,G\}$. These components work together to support a mobile edge computing environment within a square disaster area of interest of width $\mathbb{D}_{max}$. Airship$_b$ is equipped with ultra-high performance computing servers and statically positioned over a disaster area at the location $p_b = [x_b, y_b, h_b]$ for collecting tasks and processing. It is also endowed with energy-beam transmitters to supply UAVs and GDs with sustainable energy. GDs, such as smartphones and wearable devices, are mobile and are vital in disaster scenarios, enabling communication, navigation, real-time updates, health monitoring, and continuous data processing and transmission to Airship$_b$ for analysis and decision-making. UAVs are equipped with RISs to facilitate the relay of computing tasks and energy between GDs and Airship$_b$. An RIS can switch between ON and OFF states based on the UAV's battery level. For example, UAV$_u$ operates in relay mode (*i.e.,* with the RIS in the ON state) when its batteries are sufficiently charged. Otherwise, the RIS switches to the OFF state to conserve energy and UAV$_u$ operates in energy reception mode to replenish its batteries from the energy beams transmitted by Airship$_b$. The whole system operates over a finite interval $t \in [0, \Gamma]$, discretized into T time-slots of duration $\Delta = \frac{\Gamma}{T}$, ensuring UAVs and GDs are roughly stationary within each time-slot t, where $t \in \mathcal{T} \triangleq \{t = 1,2,\dots,T\}$. The time-varying location of a given GD$_g$ denoted as $p_g[t] = [x_g(t), y_g(t), h_g]$, where $h_g = 0$ since we suppose that GDs are moving on flat terrain. UAVs dynamically adjust their locations to serve prioritized GDs requiring energy and task processing, and thus the real-time location of a given UAV$_u$ is $p_u[t] = [x_u(t), y_u(t), h_u(t)]$. The distance between any pair of devices is given by:

$$d_i^j[t] = \sqrt{\|p_i[t] - p_j[t]\|^2}, \tag{1}$$

where $i, j \in \mathcal{U} \cup \mathcal{G} \cup \{b\}$. It is assumed that the UAVs' trajectories and energy management, as well as the edge computing operations of Airship$_b$, are fully controlled via a backhaul connection to a functional ground station that centralizes the optimization of each device's process (*see* Section 4 and subsections therein). This centralized optimization outcome improves the LoS between UAVs and GDs, synchronizes the movements of UAVs and their RIS ON/OFF states, optimizes the edge computing processing and energy transmission of Airship$_b$, fulfills the energy requirements of UAVs, and efficiently supplies GDs with energy. Moreover, at the beginning of the mission, the batteries of UAVs and GDs are considered to be fully charged. The edge computing process of our system is divided into three phases: (i) Energy Harvesting, where the HAP transfers wireless energy to GDs with RIS assistance, and GDs harvest energy from the received signals; (ii) Task Offloading, where GDs offload computational tasks to the HAP's MEC servers, facilitated by the RIS; and (iii) Result Transmission, where the MEC server processes the tasks and sends results back to GDs. To enhance clarity, we introduce Fig. 2 illustrating the interactions among HAP, UAVs, and GDs. The flowchart outlines deployment, task offloading, energy transfer, RIS activation, and mission continuity decisions such

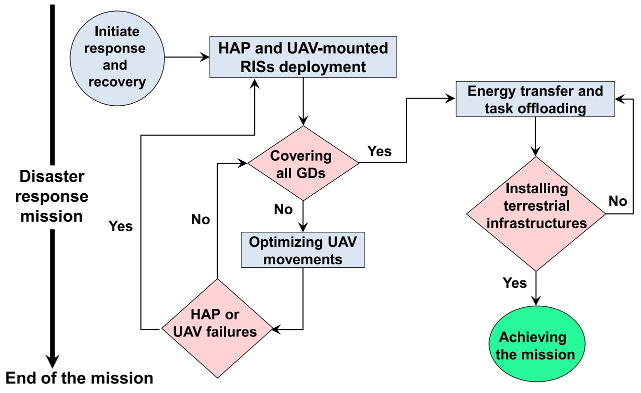

Initiate response and recovery
HAP and UAV-mounted RISs deployment
Energy transfer and task offloading
Disaster response mission
Covering all GDs
Yes
No
Yes
No
Optimizing UAV movements
Installing terrestrial infrastructures
No
Yes
HAP or UAV failures
Achieving the mission
End of the mission

Fig. 2. System-level flowchart.

Table 2
List of notations.

Notation	Description
u, U, $\mathcal{U}$	Index, Number, Set of UAVs
g, G, $\mathcal{G}$	Index, Number, Set of GDs
t, T, $\mathcal{T}$	Index, Number, Set of time-slots
b	Index of the airship (HAP)
$d_i^j[t]$	Distance between devices i and $j \in \mathcal{U} \cup \mathcal{G} \cup \{b\}$
$p_u[t]$	Position of UAV$_u$, $u \in \mathcal{U}$
$p_g[t]$	Position of GD$_g$, $g \in \mathcal{G}$
$p_b[t]$	Position of Airship$_b$
$E_i[t]$	Residual energy level of device $i \in \mathcal{U} \cup \mathcal{G}$
$E_i^b[t]$	Harvested energy received by device i from Airship$_b$
$R_g[t]$	Offloading data rate of GD$_g$ $g \in \mathcal{G}$
$H_u^i[t]$	Channel between Airship$_b$ and device i, $i \in \mathcal{U} \cup \mathcal{G}$
$Z_g[t]$	Processing decision of GD$_g$ $g \in \mathcal{G}$
$\theta_u[t]$	RIS status of UAV$_u$
$r_u[t]$	Reward obtained by UAV$_u$, $u \in \mathcal{U}$
$o_u[t]$	Environment observation of UAV$_u$, $u \in \mathcal{U}$
$a_u[t]$	Action made by UAV$_u$, $u \in \mathcal{U}$
$\pi_u(.)$	Policy function of agent $u \in \mathcal{U}$
$Q_u(.)$	Network Q's value of agent $u \in \mathcal{U}$
V_i^{max}	Maximum velocity of device $i \in \mathcal{U} \cup \mathcal{G}$

as UAV/HAP failure handling and fallback infrastructure installation. This visual flowchart simplifies complex system operations.

For the readers' convenience, Table 2 defines the key abbreviations employed in this work.

3.1. Transmission model

As illustrated in Fig. 1, each kind of device is equipped with specialized antennas optimized for its roles in disaster response and edge computing. Airship$_b$ acts as a central hub equipped with two key antenna types: (i) high-gain $K \times L$ Uniform Planar Array (UPA) antenna for focused energy transmission to UAVs and GDs, and (ii) an omnidirectional antenna serving as the primary receiver for computational tasks sent from GDs. UAVs are equipped with two types of antennas: (i) an omnidirectional antenna, dedicated to harvesting energy from Airship$_b$, ensuring UAVs maintain a continuous power supply, and (ii) a RIS equipped with UPA consisting of $M \times N$ passive reflecting elements, which assists in redistributing energy from Airship$_b$ to GDs and relaying computational tasks back to Airship$_b$. GDs are equipped with a single omnidirectional antenna that performs dual functions: receiving energy during the Energy Harvesting phase and transmitting computational tasks during the Task Offloading phase via NOMA. Therefore, Successive Interference Cancellation (SIC) is adopted at the primary receiver (*i.e.*, the single antenna at Airship$_b$) to remove the multiple-access interference.

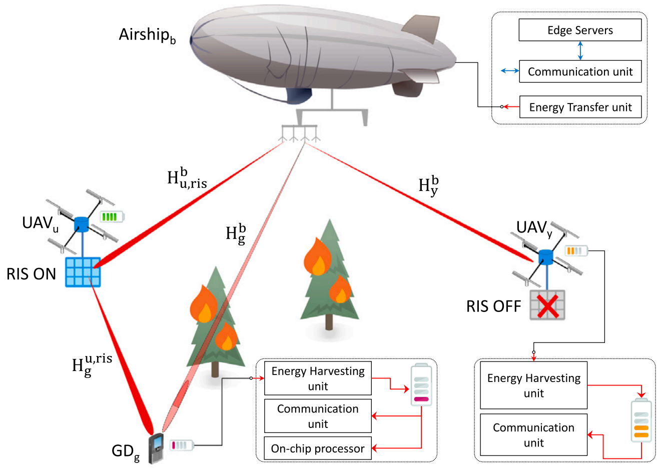


Fig. 3. Energy harvesting model.

The RIS can be dynamically switched between ON and OFF states to optimize energy efficiency and data transmission. The RIS embedded in a specific UAV$_u$ is activated when the residual energy of UAV$_u$ exceeds a certain threshold, allowing efficient energy and data reflection and transmission. Otherwise, the RIS remains OFF to allow UAV$_u$ only to harvest energy from Airship$_b$ to supply its battery with energy. We introduce a binary variable $\theta_u[t]$ to represent the state of the RIS as follows:

$$\theta_u[t] = \begin{cases} 1, & \text{If RIS is ON,} \\ 0, & \text{If RIS is OFF,} \end{cases} \tag{2}$$

It should be stressed that each time-slot is divided into two main durations: (i) the Energy Harvesting phase has a duration of $\alpha\Delta$ and (ii) the Task Offloading phase has a duration of $(1-\alpha)\Delta$, where α is the proportion of the time-slot dedicated to energy harvesting. Given the superior computational capabilities of Airship$_b$ and the relatively small size of the computed outcomes, the computation and downloading phase is considered negligible and is thus omitted from the time-slot division (*c.f.,* Fig. 1).

3.1.1. Energy harvesting phase

During the energy harvesting phase, GDs and UAVs intercept multiple narrow beams transmitted from Airship$_b$. These intercepted Radio Frequency (RF) signals are converted into energy, which powers the GDs to supply their computing task transmission to Airship$_b$ through or not UAV-mounted RIS (*c.f.,* Fig. 3). For simplicity, we assume static beam angles without considering dynamic changes due to mechanical vibrations or airflow. Moreover, we suppose that the HAP has perfect knowledge of the CSI (Channel State Information). While perfect CSI is assumed for analytical simplicity, we acknowledge that imperfect CSI can lead to suboptimal energy transfer, which can reduce the energy harvesting efficiency and affect system performance. In future work, we intend to investigate the impact of imperfect CSI and propose CSI acquisition and prediction methods that improve the accuracy of channel estimates. Furthermore, we assume static beam angles for simplicity, but dynamic beam steering, which accounts for mechanical vibrations or airflow, may impact energy transfer efficiency. We plan to explore adaptive beamforming techniques in future research to account for dynamic changes in beam angles. Studies on CSI acquisition are out of the scope of this article but can be found in Zhang et al. (2020), Wang et al. (2020).

This work uses simplified Rician fading models for communication between HAPs, UAVs, and GDs to focus on the core contributions of the framework. However, real-world conditions involve multipath fading, interference, and Doppler effects, which can degrade performance. Advanced techniques like beamforming and interference cancellation can mitigate these effects. Doppler shifts are assumed to be perfectly compensated using GPS-based velocity and position tracking, as discussed in Bai et al. (2020). Future work will explore the impact of these factors on system performance and evaluate more complex channel models to improve real-world applicability. Therefore, the time-varying channel between Aiship$_b$ and UAV$_u$, when RIS is ON, is assumed to experience the Rician channel fading, as in Li et al. (2022), with only a LoS component and can be modeled as:

$$H^b_{u,ris}[t] = \sqrt{\zeta_0 d^b_u[t]^{-\eta}}\left(\sqrt{\frac{\hat{K}}{1+\hat{K}}}H^{b,LoS}_{u,ris}[t]\right), \tag{3}$$

where ζ_0 is the channel gain at the reference distance $d_0 = 1$ m, $\eta \geq 2$ is the path loss exponent, and $\hat{K}$ denotes the Rician factor. $H^{b,LoS}_{u,ris}[t]$ represents the deterministic LoS component between Airship$_b$ and UAV$_u$ with RIS in ON state. In turn, the RIS embedded in UAV$_u$ redistributes energy to GDs through both LoS and NLoS propagation, based on the following time-varying channel between UAV$_u$ and a given GD$_g$:

$$H^{u,ris}_g[t] = \sqrt{\zeta_0 d^u_g[t]^{-\eta}}\left(\sqrt{\frac{\hat{K}}{1+\hat{K}}}H^{u,ris,LoS}_g[t] + \sqrt{\frac{1}{1+\hat{K}}}H^{u,ris,NLoS}_g[t]\right), \tag{4}$$

where $H^{u,ris,LoS}_g[t]$ and $H^{u,ris,NLoS}_g[t]$ are the deterministic LoS and NLoS components between UAV$_u$ and GD$_g$. Nevertheless, in the case when an RIS of UAV$_u$ is in the OFF state, UAV$_u$ only harvests energy from Airship$_b$ and the time-varying LoS channel between them can be expressed as:

$$H^b_u[t] = \sqrt{\zeta_0 d^b_u[t]^{-\eta}}\left(\sqrt{\frac{\hat{K}}{1+\hat{K}}}H^{b,LoS}_u[t]\right), \tag{5}$$

where $H^{b,LoS}_u[t]$ is the deterministic LoS component between Airship$_b$ and UAV$_u$ with RIS in OFF state. In the case when a given GD$_g$ is not covered by any UAVs with RIS ON, it could have LoS or NLoS link with Airship$_b$. Therefore, the adopted time-varying channel is given by:

$$H^b_g[t] = \sqrt{\zeta_0 d^b_g[t]^{-\eta}}\left(\sqrt{\frac{\hat{K}}{1+\hat{K}}}H^{b,LoS}_g[t] + \sqrt{\frac{1}{1+\hat{K}}}H^{b,NLoS}_g[t]\right), \tag{6}$$

where $H^{b,LoS}_g[t]$ and $H^{b,NLoS}_g[t]$ are the deterministic LoS and NLoS components between GD$_g$ and Airship$_b$. Therefore, the harvested energy at GD$_g$ from Airship$_b$ can be expressed as:

$$E^b_g[t] = \xi\left(|H^b_g[t]|^2 P_b + \sum_{u=1}^{U}\theta_u[t]\left|H^b_{u,ris}[t]\Theta_{ris}a_{HAP}(\omega,\phi) \times H^{u,ris}_g[t]\right|^2 P_b\right)\alpha\Delta, \tag{7}$$

where Θ_{ris} is the phase-shift matrix of the RIS and can be calculated as $\Theta_{ris} = \text{diag}\left(e^{j\varrho_1}, e^{j\varrho_2}, \dots, e^{j\varrho_{MN}}\right)$ and ϱ_i the phase shift applied by the ith element of the RIS. P_b is the power transmission of Airship$_b$. ξ is the energy conversion efficiency. $a_{HAP}(\omega,\phi)$ represents the steering vector of the UPA antennas at the HAP for an angle pair (ω,ϕ) can be expressed as:

$$a_{\text{HAP}}(\omega,\phi) = \frac{1}{\sqrt{KL}}\left[1, e^{j\frac{2\pi}{\lambda}d(\sin(\omega)\cos(\phi))}, \dots, e^{j\frac{2\pi}{\lambda}d((K-1)\sin(\omega)\cos(\phi)+(L-1)\sin(\omega)\sin(\phi))}\right]^T. \tag{8}$$

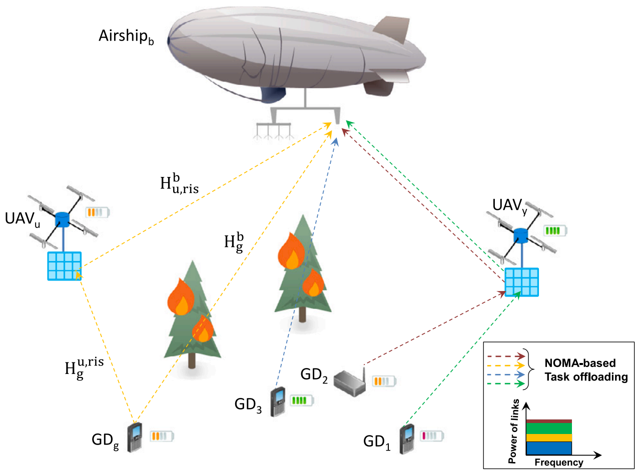


Fig. 4. NOMA Task offloading model.

UAVs could also harvest energy from Airship$_b$ during the energy harvesting phase only when their embedded RISs are OFF. Therefore, the harvested energy $E^b_u[t]$ at time-slot t is expressed as:

$$E^b_u[t] = \xi \times P_b \times |H^b_u[t]|^2 \times \alpha\Delta \tag{9}$$

3.1.2. Task offloading phase

In the task offloading phase, GDs offload their computational tasks to the HAP using the NOMA protocol to reduce latency (*c.f.,* Fig. 4).

The HAP ranks GDs based on their channel gains in ascending order and decodes their signals using the SIC method. When processing the signal from a specific GD$_g$, the signals received from other GDs are considered interference. Consequently, the total offloading data rate $R_g[t]$ for a given GD$_g$ is calculated as a sum between the individual data rates for direct transmission and relayed transmission via UAV-mounted RIS as in Nguyen and Le (2022), Wu et al. (2023). The direct transmission data rate can be expressed as follows:

$$R^{\text{direct}}_g[t] = B\log_2\left(1 + \frac{P_g(t)|H^b_g[t]|^2}{\sum_{k>g}P_k(t)|H^b_k[t]|^2 + \sigma^2_b}\right), \tag{10}$$

where $P_g(t)$ is the time-varying transmit power of GD$_g$. $|H^b_g[t]|^2$ is the channel gain between GD$_g$ and the HAP, σ^2_b is the noise power at the HAP, and B is the bandwidth allocated to the channel. When RIS is ON, the effective data rate of the relayed transmission includes the UAV-to-HAP link and the RIS reflection. Therefore, the offloading data rate $R^{\text{relayed}}_g[t]$ for transmission via the UAV-mounted RIS is:

$$R^{\text{relayed}}_g[t] = B\log_2\left(1 + \frac{P_g(t)\sum_{u=1}^{U}\theta_u[t]\left|H^{u,ris}_g[t]\Theta_{ris}H^b_{u,ris}[t]\right|^2}{\sum_{k>g}P_k(t)\left|H^{u,ris}_k[t]\Theta_{ris}H^b_{u,ris}[t]\right|^2 + \sigma^2_u}\right), \tag{11}$$

where $H^{u,ris}_g[t]$ is the channel between GD$_g$ and UAV$_u$ with RIS ON, $H^b_{u,ris}[t]$ is the channel between UAV$_u$ and the HAP, Θ_{ris} is the RIS's phase-shift matrix, and σ^2_u is the noise power at UAV$_u$. To ensure that a given GD$_g$ is served by at least one UAV-mounted RIS while meeting certain QoS criteria, a binary variable $\aleph_g$ is introduced to represent the coverage status of GD$_g$:

$$\aleph_g[t] = \begin{cases} 1, & \text{If } R^{\text{relayed}}_g[t] + R^{\text{direct}}_g[t] \geq \tau, \\ 0, & \text{Otherwise,} \end{cases} \tag{12}$$

where τ is defined as the threshold to meet the required QoS according to the kind of application. Consequently, the total offloading data rate

$R_g(t)$ for GD_g is the sum of the direct and relayed data rates:

$$R_g[t] = R_g^{\text{direct}}[t) + R_g^{\text{relayed}}[t]. \tag{13}$$

Another binary variable $\beth_g^u$ is introduced to define if a given UAV_u with RIS in active state is covering a given GD_g or not.

$$\beth_g^u[t] = \begin{cases} 1, & \text{If } B\log_2\left(1 + \frac{P_g(t)\left|\theta_u[t]H_g^{u,ris}[t]\Theta_{ris}\right|^2}{\sum_{k>g} P_k(t)\left|H_k^{u,ris}[t]\Theta_{ris}\right|^2 + \sigma_u^2}\right), \geq \tau, \\ 0, & \text{Otherwise,} \end{cases} \tag{14}$$

3.2. Energy consumption

In disaster response scenarios, energy management is critical for maintaining the functionality and operational efficiency of both GDs and UAVs. Efficient energy utilization is vital, especially given the limited power resources available in these devices during disaster response operations. Understanding and modeling the energy consumption helps in optimizing the performance and extending the operational lifespan of the system components. This section delves into the energy consumption aspects of these components within our proposed edge computing framework.

3.2.1. Computation model and energy consumption of GDs

In our disaster response system, GDs such as smartphones and IoT devices are critical for real-time data processing and task offloading. Efficiently managing the computational load and energy consumption of GDs is essential for maintaining operational effectiveness in resource-constrained environments. We analyze the energy consumption of GDs by examining two primary elements: (i) computation energy and (ii) communication energy. Therefore, to estimate the energy consumed during the computation process of GDs, we first need to characterize the computing tasks. Let us define $D_g[t]$ the size of the task offloaded by GD_g at time-slot t, and C_g denotes the CPU cycles's number required to process one bit of data for GD_g. When GD_g considers it possible to make the processing locally, the total CPU cycles's number needed to complete the task $T_g[t]$ at time-slot t is given by:

$$T_g[t] = D_g[t] \times C_g \tag{15}$$

The processing time $\Psi_g[t]$ for the task is determined by the frequency of computation $f_g[t]$ (cycles per second):

$$\Psi_g[t] = \frac{T_g[t]}{f_g[t]} \tag{16}$$

Consequently, the energy consumed by GD_g for local computation $E_g^{\text{local}}[t]$ is influenced by the processing frequency $f_g[t]$ and can be modeled as in Abkenar et al. (2022) as follows:

$$E_g^{\text{local}}[t] = \kappa_g \times T_g[t] \times \left(f_g[t]\right)^{\beta-1}, \tag{17}$$

where κ_g is a device-specific constant reflecting the energy efficiency of the GD's processor. β is the dynamic power consumption exponent, typically ranging between 2 and 3 for most processors. In the case when a given GD_g needs to process its computing tasks through Airship_b according to the adopted strategy (*see* Section 4.1), the energy consumed for transmitting computing tasks $E_g^{\text{remote}}[t]$ from GD_g to Airship_b or via the UAV-mounted RIS is given by:

$$E_g^{\text{remote}}(t) = \int_0^t Z_g[t]P_g[t] \times T_g^{\text{trans}}[t]dt, \tag{18}$$

where $T_g^{\text{trans}}[t] = \frac{D_g[t]}{R_g[t]}$ is the duration of the transmission period. $Z_g[t]$ is a binary variable that positioned to 1 when GD_g decides to offload its computing tasks to Airship_b and 0 otherwise. Therefore, the residual energy level of GD_g at each time-slot t can be calculated as follows:

$$E_g[t] = E_g[t-1] + (E_g^b[t] - (E_g^{\text{remote}}[t] + E_g^{\text{local}}[t])) \tag{19}$$

3.2.2. Energy consumption of UAVs

Given the extensive operational roles of UAVs, it is vital to manage their energy consumption effectively. This section explores the detailed energy consumption of UAVs, focusing on the propulsion energy required for movement and hovering, the operational energy of RISs, and the calculation of residual energy levels at each time slot. In our work, we adopted special quadcopters to carry RISs and cover the disaster area of interest. Therefore, according to Cabuk et al. (2024), the propulsion energy is dominated by the energy needed for hovering and the additional energy required for movements. The energy required for a given UAV_u to hover is primarily determined by the need to counteract its weight. The power required for hovering P_u^{hover} for UAV_u is modeled as:

$$P_u^{\text{hover}} = \frac{I_p \cdot W_u^{3/2}}{\sqrt{2 \cdot \rho \cdot A_r}}, \tag{20}$$

where I_p, ρ, and A_r, are the induced power coefficient, the air density (in kg/m^3), and the rotor disk area (in m^2), respectively. W_u is the weight of UAV_u along with its carried RIS. When UAV_u moves, additional power is required to overcome air resistance and generate the thrust needed for movement. The power for movement $P_u^{\text{move}}[t]$ can be expressed as:

$$P_u^{\text{move}}[t] = \frac{1}{2} \cdot DG_c \cdot \rho \cdot FR_a \cdot V_u(t)^3, \tag{21}$$

where DG_c and FR_a are the drag coefficient of the UAV and the frontal area of the UAV facing the direction of movement. $V_u(t)$ is the velocity of UAV_u at time-slot t. Consequently, the total propulsion energy consumption $E_u^{\text{prop}}[t]$ for UAV_u over the duration Δ of time-slot t is:

$$E_u^{\text{prop}}[t] = \int_0^t \left(P_u^{\text{hover}} + P_u^{\text{move}}[t]\right) dt. \tag{22}$$

The energy consumption of RISs embedded in UAVs is significant when RISs are actively reflecting signals. The energy consumption $E_u^{\text{ris}}[t]$ for the RIS on UAV_u when it is ON at time-slot t is given by:

$$E_u^{ris}[t] = \theta_u[t] \times P_u^{ris} \times M \times N \times \Delta, \tag{23}$$

where P_u^{ris} is the power consumption per reflecting element. Consequently, the residual energy level $E_u[t]$ of UAV_u at time-slot t is given by:

$$E_u[t] = E_u[t-1] + (E_u^b[t] - (E_u^{\text{prop}}[t] + E_u^{ris}[t])). \tag{24}$$

4. Problem formulation and solutions

In this section, we address the optimization challenges of (i) task offloading and (ii) UAV movement with RIS control, aiming to minimize overall operational costs, including energy consumption, offloading costs, and latency. For task offloading, we determine the optimal decisions for GDs on whether to process tasks locally or offload them to Airship_b, incorporating the energy, latency, and offloading costs into the cost function. Simultaneously, we optimize UAV movement and RIS control to enhance communication link quality by dynamically adjusting the UAV's position and RIS configurations. We employ game theory and advanced optimization techniques to solve these problems: a potential game framework for task offloading to find the Nash Equilibrium, ensuring globally optimized decisions, and a reinforcement learning scheme based on a multi-agent architecture for UAV movement and RIS control to maximize system performance by improving QoS-based coverage through strategic adjustments. These integrated solutions aim to balance cost minimization and communication efficiency.

4.1. Task offloading optimization

We consider a scenario where GDs can either offload their computing tasks to Airship$_b$ or process them locally. To optimize this process, we consider a game modeled as $\Omega(G, Z_g, Cost_g)_{g\in\mathcal{G}}$, where G is the number of GDs/players, $Z_g \in \{0,1\}$ denotes the offloading decision for GD$_g$, where $Z_g[t] = 1$ denotes that the task needs to be offloaded to Airship$_b$ and $Z_g[t] = 0$ indicates local processing and $Cost_g$ is the cost function for GD$_g$. The objective is to minimize the total cost, which includes the energy consumption for local processing and latency. The time-varying cost function for GD$_g$ can be expressed as:

$$Cost_g(Z_g[t]) = \begin{cases} E_g^{\text{local}}[t] + \Psi_g[t], & \text{if } Z_g[t] = 0, \\ E_g^{\text{remote}}[t] + T_g^{\text{trans}}[t], & \text{if } Z_g[t] = 1. \end{cases} \tag{25}$$

The global cost function is then formulated as:

$$\min_{\{Z_g[t]\}} \sum_{g=1}^{G} Cost_g(Z_g[t]) \tag{26}$$

s.t.

$$\textbf{C1}: \quad Z_g[t] \in \{0,1\}, \quad \forall g \in \mathcal{G}.$$

To solve this optimization problem, we utilize a potential game framework (Liu et al., 2022a). A potential game is characterized by a potential function $\Phi(Z_g[t])$ such that any unilateral change by a single player results in a change in the potential function that reflects the change in the player's cost. The potential function for our game can be defined as:

$$\begin{aligned}\Phi(Z_g[t]) = \sum_{g=1}^{G} \Big(& Z_g[t] \left(E_g^{\text{remote}}[t] + T_g^{\text{trans}}[t] \right) \\ & + (1 - Z_g[t]) \left(E_g^{\text{local}}[t] + \Psi_g[t] \right) \Big)\end{aligned} \tag{27}$$

Algorithm 1 iteratively adjusts GDs' offloading decisions to minimize costs. A Nash Equilibrium is reached when no GD can reduce its cost by changing its decision, ensuring a locally optimized solution.

Algorithm 1: Algorithm to Find Nash Equilibrium

```
Initialize the offloading decisions $Z_g[0] = 0, \forall g \in \mathcal{G}$.
repeat
    for each GD$_g \in \mathcal{G}$ do
        Compute current cost $Cost_g(Z_g[t])$.
        Compute current potential function $\Phi(\{Z_g[t]\})$.
        Evaluate potential function for alternative decision
        $\Phi(\{Z_g[t] \mid Z_g[t] \leftarrow 1 - Z_g[t]\})$.
        if Alternative decision reduces potential function then
            Update decision: $Z_g[t] \leftarrow 1 - Z_g[t]$.
        end if
    end for
until No changes in decisions for all GDs.
```

4.2. UAV movement and RIS control optimization

This section focuses on optimizing UAV movement and RIS control to ensure each GD is supported by a UAV-mounted RIS for local or remote task processing. The goal is to enhance system performance while meeting energy efficiency, coverage, and latency constraints. The framework ensures that each GD with tasks or energy needs is served by an active RIS-equipped UAV, prevents UAV energy depletion, avoids collisions, maintains GD coverage, and reduces latency. These optimizations consider the unpredictable mobility of GDs. To clarify, let $P_U[t] = \{p_u[t], \forall u \in \mathcal{U}\}$ and $\theta = \{\theta_u[t], \forall u \in \mathcal{U}\}$ be the UAV locations and RIS states at time-slot t. The optimization problem is formulated as follows:

$$\max_{\{P_U[t],\theta\}} \mathbb{E}\left[\left(\frac{\sum_{t=1}^{T}\sum_{g=1}^{G} R_g[t]}{\sum_{t=1}^{T}\sum_{g=1}^{G}(\Psi_g[t] + Z_g[t]T_g^{\text{trans}}[t]) + 1}\right)\left(\frac{\sum_{t=1}^{T}\bar{E}[t]}{T}\right)\right] \tag{28}$$

s.t.

$$\begin{aligned}
&\textbf{C1}: && \|p_u[t] - p_u[t-1]\|^2 \le (V_u^{max}\Delta)^2, \forall u \in \mathcal{U},\\
&\textbf{C2}: && \sum_{g=1}^{G} \aleph_g[t] \ge G, \quad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T},\\
&\textbf{C3}: && \sum_{u=1}^{U} \beth_g^u[t] > 0, \quad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T},\\
&\textbf{C4}: && d_u^y[t] \ge d_{min}, \quad \forall u, y \in \mathcal{U},\ \forall t \in \mathcal{T},\\
&\textbf{C5}: && E_u[t] > 0, \quad \forall u \in \mathcal{U},\ \forall t \in \mathcal{T},\\
&\textbf{C6}: && 0 \le x_u(t) \le \mathbb{D}_{max}, \forall u \in \mathcal{U},\ \forall t \in \mathcal{T},\\
&\textbf{C7}: && 0 \le y_u(t) \le \mathbb{D}_{max}, \forall u \in \mathcal{U},\ \forall t \in \mathcal{T},
\end{aligned}$$

where $\bar{E}[t]$ represents the average residual energy level of all devices involved in the mission (*i.e.,* UAVs and GDs). $\mathbb{E}$ is evaluated by considering the random movements of GDs. **C1** shows the traveled distance covered by UAV$_u$ during time-slot t. **C2** indicates that all GDs should be linked to Airship$_b$ with a certain level of QoS. **C3** ensures that each UAV$_u$, where $u \in \mathcal{U}$, should at least cover one GD. **C4** allows us to maintain a safety distance d_{min} between each pair of UAVs. **C5** guarantees that all UAVs remain active during the whole mission. **C6** and **C7** impose to UAVs to stay within the disaster area of interest to serve GDs. Our optimization strategy aims to develop an efficient control policy $\pi(.)$ to guide the UAVs' movements and RIS operations in a way that balances three main objectives: (i) Reducing the energy consumption of UAVs while powering sufficiently GDs to allow them performing their tasks, (ii) Maximizing the coverage of GDs by UAVs by optimizing RISs' states and ensure that GDs maintain QoS level sufficient for their offloaded computational tasks, and (iii) Reducing the latency in task processing and data transmission by optimizing the positioning of UAVs to minimize communication delays. Achieving these goals involves handling the inherent trade-offs and complexities in the system. For instance, while moving UAVs to better positions may improve coverage and reduce latency, it also increases energy consumption. Given the complexity and dynamic nature of the environment, we employ a MARL approach to derive the control policy.

To solve the complex optimization problem defined in (28), involving the coordination of UAV movements and RIS control, we employ a Multi-Agent Deep Deterministic Policy Gradient (DDPG) method (Lillicrap et al., 2015). Our RL-based framework, called DIRECT, combines the strengths of RL and deep learning to efficiently manage continuous action spaces and the interactions of multiple agents in a dynamic environment. Conventional RL is framed as a Markov Decision Process (MDP), defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$. Here, $\mathcal{S}$ signifies the set of states, $\mathcal{A}$ the set of possible actions, $\mathcal{R}$ the reward function, and $\mathcal{P}$ the state transition probability. At every time-slot t, an agent observes the present state $s[t] \in \mathcal{S}$, selects an action $a[t] \in \mathcal{A}$ based on the policy $\pi(s[t], a[t])$, receives a reward $r[t] \in \mathcal{R}$, and moves to a new state $s[t+1] \in \mathcal{S}$ following the state transition probability $\mathcal{P}(s[t+1]|s[t], a[t])$. The goal is to maximize the expected cumulative rewards over time by finding the optimal policy π. The DDPG algorithm is a Deep RL method suitable for continuous control problems. It maintains two neural networks: an actor network $\pi(s[t]|\varpi^{\pi})$ that generates optimal actions $a[t]$ based on the current state $s[t]$, and a critic network $Q(s[t], a[t]|\varpi^{Q})$ that estimates the value of actions using the Bellman equation. The actor network is trained using the chain rule to optimize the expected reward J with respect to its weights ϖ^{π}, expressed as:

$$\nabla_{\varpi^{\pi}} J(\varpi^{\pi}) \approx \mathbb{E}\left[\nabla_{\varpi^{\pi}}\pi(s|\varpi^{\pi})|_{s=s_t}\nabla_a Q(s,a|\varpi^{Q})|_{s=s_t,a=\pi(s_t)}\right]. \tag{29}$$

As depicted in Fig. 5, each UAV is considered an individual agent that observes its local environment, takes actions to adjust its position and RIS state, and receives rewards based on the overall system performance. In our system, each UAV is treated as an individual agent that observes its local environment, takes actions to adjust its position and RIS state, and receives rewards based on the overall

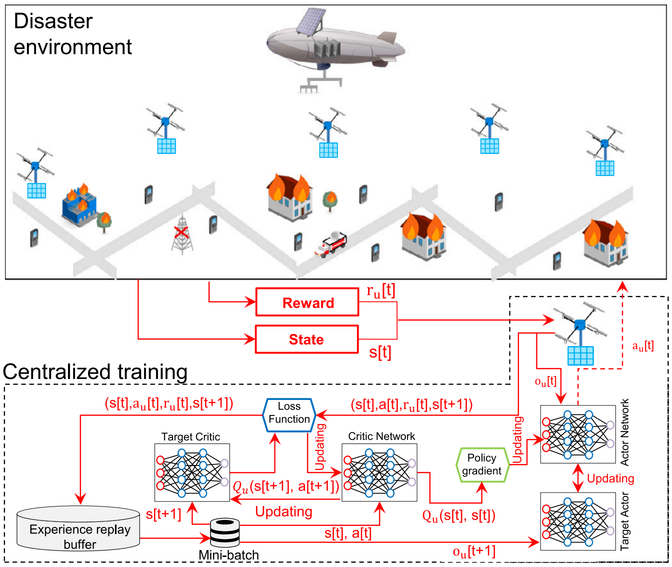


Fig. 5. DIRECT framework.

system performance. These agents interact with a dynamic environment defined by states $\mathcal{S} \triangleq \{s[t], t \in \mathcal{T}\}$. Each state $s[t] = \{o_u[t], t \in \mathcal{T}, u \in \mathcal{U}\}$ represents the observations of the UAVs, where $o_u[t]$ is the private observation of UAV$_u$. The set of actions $\mathcal{A} \triangleq \{a[t], t \in \mathcal{T}\}$ represents the feasible continuous actions that the UAVs can take. Each action $a_t = \{a_u[t], t \in \mathcal{T}, u \in \mathcal{U}\}$ denotes the action executed by UAV$_u$. At each time-slot t, UAV$_u$ observes its private observation $o_u[t]$, executes its action $a_u[t]$, and obtains its reward $r_u[t]$. The environment then transitions to a new state $s[t+1]$, incorporating the effects of all agents' actions. Each UAV maintains its own buffer $\mathbb{B}_u$, a critic network $Q_u(s[t], a[t]|\varpi^{Q_u})$, an actor network $a_u[t] = \pi^u(o_u[t]|\varpi^{\pi_u})$, and their respective target networks $Q_{u'}(s[t+1], a[t+1]|\varpi^{Q_{u'}})$ and $\pi_{u'}(o_u[t+1]|\varpi^{\pi_{u'}})$. The optimization is carried out through centralized training with a distributed execution strategy. Each UAV calculates and executes its own action $a_u[t]$ and receives its own reward $r_u[t]$. The environment state $s[t]$ includes the private observations of all UAVs. This shared information allows each UAV to optimize its actions while considering the actions of other UAVs. The main goal of each UAV is to maximize its total accumulated rewards, which leads to coordinated and efficient operations across all UAVs and RISs.

4.2.1. State space

In DIRECT framework, each agent $u \in \mathcal{U}$ observes the environment state $s[t] = \{o_u[t], t \in \mathcal{T}, u \in \mathcal{U}\}$, where $o_u[t]$ comprises the following components:

- $p_u[t], u \in \mathcal{U}$: the current position of UAV$_u$.
- $E_u[t], u \in \mathcal{U}$: the current residual energy level of UAV$_u$.
- $\theta_u[t], u \in \mathcal{U}$: the RIS's current state of UAV$_u$.
- $p_g[t], \forall g \in \mathcal{G}$: the time-varying positions of all GDs.
- $P_g[t], \forall g \in \mathcal{G}$: the time-varying transmission power of all GDs.
- $Z_g[t], \forall g \in \mathcal{G}$: the time-varying processing decisions of all GDs.

At each time-slot t, the observation $o_u[t]$ for UAV$_u$ is formally defined as: $o_u[t] = [p_u[t], E_u[t], \theta_u[t], p_1[t], \ldots, p_G[t], P_1[t], \ldots, P_G[t], Z_1[t], \ldots, Z_G[t]]$. This observation vector has a cardinality of 3+3G, encompassing the positions, processing decisions, and transmission powers of all GDs. It also includes the position, energy level, and RIS's current state of UAV$_u$. In the online phase, the control system collects these observations at each time-slot. Since Airship$_b$ and all UAVs and GDs are backhaul connected to a central controller station, the collected data is utilized to make informed decisions for each UAV. These observations are crucial inputs for our Multi-Agent DDPG framework, guiding the UAVs in optimizing their movements and RIS configurations. To expedite the learning process, all elements in the observation vector $o_u[t]$ are normalized. This normalization involves scaling each parameter to a range between 0 and 1 by dividing by their respective maximum values. This step ensures that the learning algorithm converges more quickly and effectively handles the variations in the observed data.

4.2.2. Action space

The action space $\mathcal{A}$ of UAVs in our system is denoted as $\mathcal{A} = \{\mathcal{A}_{\text{RIS-ON}}[t], \mathcal{A}_{\text{RIS-OFF}}[t]\}$. This distinction into two sub-action spaces reflects the different operational modes depending on whether the RIS is active (ON) or inactive (OFF). For $\mathcal{A}_{\text{RIS-ON}}[t]$, where the RIS is in the ON state, the action space consists of the following components:

- $\psi_u[t] \in [0, 2\pi$: The flying horizontal direction of UAV$_u$, which represents the angle of movement in the horizontal plane.
- $h_u[t] \in [A1, A2]$: The interval altitudes of UAV$_u$, indicating the range within which the UAV can adjust its altitude.
- $d_u[t] \in [0, d_{max}]$: The flying distance of UAV$_u$, specifying how far the UAV can move in the given time-slot.
- $\theta_u[t] = 1$: The state of the RIS, which is ON in this sub-action space.

When the RIS state is OFF $\theta_u[t] = 0$, represented by $\mathcal{A}_{\text{RIS-OFF}}[t]$ in which $\psi_u[t] = 0$, $d_u[t] = 0$, and the altitude remains unchanged $h_u[t] = h_u[t-1]$. This means that UAV$_u$ remains stationary until the charging process of its battery is completed to ensure optimal energy transfer scheduling. Formally, the action $a_u[t] \in \tilde{\mathcal{A}}[t]$ of each UAV$_u$ at time-slot t can be expressed based on the RIS state as follows:

$$\tilde{\mathcal{A}}[t] = \begin{cases} \mathcal{A}_{\text{RIS-ON}}[t], & \text{if } \theta_u[t] = 1, \\ \mathcal{A}_{\text{RIS-OFF}}[t], & \text{if } \theta_u[t] = 0. \end{cases} \tag{30}$$

This dual action space approach ensures that when the RIS is active, UAVs can dynamically adjust their positions and optimize their operations for better coverage and communication. Conversely, when the RIS is inactive, UAVs minimize their movements to conserve energy. This balance between dynamic positioning and energy efficiency is crucial for the overall performance of the system.

4.2.3. Reward function

At each time-slot t, the reward function $r_u[t]$ for each UAV$_u \in \mathcal{U}$ is designed to optimize the system's objectives while considering penalties for undesired behaviors. The reward function encourages actions that improve coverage, conserve energy, and penalize actions that lead to collisions, inefficient energy use, or deviations from the desired operational area. It should be stressed that the whole latency of the system is optimized through the adopted game theory method. The reward function can be expressed as the following:

$$r_u[t] = \frac{E_u[t] \cdot \theta_u[t] \cdot \sum_{g=1}^{G} \beth_g^u[t] R_g[t]}{1 + \rho_u[t]}. \tag{31}$$

The numerator includes factors that reward positive behaviors: $E_u[t]$ (the residual energy of UAV$_u$), $\theta_u[t]$ (the state of the RIS, where $\theta_u[t] = 1$ for ON), and $\sum_{g=1}^{G} \beth_g^u[t] R_g[t]$ (the cumulative data rate for all GDs served by UAV$_u$). The denominator incorporates the penalties $\rho_u[t]$, which deter undesirable actions. Three penalties are defined to discourage actions that negatively impact the mission objectives. The first penalty addresses UAVs deviating from the disaster area of interest or colliding with other UAVs. It is expressed as:

$$\rho_u^1[t] = \begin{cases} \Lambda_1, & \text{if } x_u[t] \notin [0, \mathbb{D}_{max}] \wedge y_u[t] \notin [0, \mathbb{D}_{max}], \\ \Lambda_2, & \text{if } d_u^v[t] \leq d_{min}, \forall v \neq u \in \mathcal{U}, \\ 0, & \text{Otherwise.} \end{cases} \tag{32}$$

The second penalty is applied when UAV$_u$ with sufficient battery power higher than a certain threshold Υ unnecessarily turns off its RIS. It is expressed as:

$$\rho_u^2[t] = \begin{cases} \Lambda_3, & \text{if } \theta_u[t] = 0 \wedge E_u[t] > \Upsilon, \\ 0, & \text{Otherwise.} \end{cases} \tag{33}$$

It is worthy to note that $\theta_u[t]$ is set to -1 when UAV$_u$ starts the charging process of its battery. When the battery of UAV$_u$ is full, $\theta_u[t]$ will be automatically positioned to 1. This technique aims to validate the correctness of (33). A third penalty is imposed when UAV$_u$ with an active RIS fails to cover any GDs or provides poor coverage. Moreover, we penalize each UAV$_u$ when a given GD$_g$ has a computing task to offload and it is not covered by at most one UAV. It is expressed as:

$$\rho_u^3[t] = \begin{cases} \Lambda_4, & \text{if } \sum_{g=1}^{G} \aleph_g < G, \\ \Lambda_5 + \Lambda_4, & \text{if } \sum_{u=1}^{U} \beth_g^u = 0 \wedge Z_g[t] = 1, \exists g \in \mathcal{G}, \\ 0, & \text{Otherwise.} \end{cases} \tag{34}$$

Here, Λ_1, Λ_2, Λ_3, Λ_4, and Λ_5 are predefined penalty factors, which can be adjusted based on the application context and the severity of the violations. The total penalty $\rho_u[t]$ for each UAV$_u$ at time-slot t is calculated by summing the individual penalties:

$$\rho_u[t] = \sum_{k=1}^{3} \rho_k^u[t]. \tag{35}$$

This reward function balances the system's performance goals, such as high offloading data rates and low energy consumption, with the need to maintain safe, efficient operations across the UAV network. By incorporating these penalties, the system discourages behaviors that could lead to inefficiencies or risks, ensuring that UAVs operate effectively within the constraints of the disaster response scenario.

4.2.4. DIRECT algorithm

As illustrated in Fig. 5, the training process is centralized at the central controller, while the execution of actions is decentralized, allowing each UAV to act based on its local observations. The DIRECT algorithm aims to learn an optimal policy $\pi_u^*(.)$ for each UAV$_u$, balancing objectives such as energy efficiency, coverage, and latency. The algorithm, formally described in Algorithm 2, shows that each UAV$_u \in \mathcal{U}$ operates as an independent agent and executes the DIRECT algorithm to make decisions about its movements and RIS settings.

In the initial phase (Lines 1–6), each UAV$_u$ initializes its Actor network $\pi_u(s[t]|\varpi^{\pi_u})$ and Critic network $Q_u(s[t], a_u[t]|\varpi^{Q_u})$ with random weights ϖ^{π_u} and ϖ^{Q_u}, respectively (*see* Line 1). The target networks $\pi_u'(s[t]|\varpi^{\pi_u'})$ and $Q_u'(s[t], a_u[t]|\varpi^{Q_u'})$ are also initialized with the same weights as the original networks (*see* Line 2). The replay buffer $\mathbb{B}_u$ for each UAV is set up with a capacity B to store the experience tuples (*see* Line 3). Exploration noise ε is introduced to encourage exploration of the action space (*see* Line 4). The number of episodes EP is set to M, and the number of time steps per episode is set to T (*see* Lines 5–6). During the learning phase (Lines 7–24), the algorithm iterates over M episodes, each consisting of T time steps. At the beginning of each episode, the initial position and energy level of each UAV are set, and the initial observation state $o_u[0]$ is received (*see* Lines 8–9). For each time step t, each UAV$_u$ selects an action $a_u[t]$ based on its current policy $\pi_u(o_u[t]|\varpi^{\pi_u})$ and adds stochastic noise to promote exploration (*see* Line 11). The selected action, consisting of horizontal direction $\psi_u[t]$, altitude $h_u[t]$, distance $d_u[t]$, and RIS state $\theta_u[t]$, is then executed (*see* Line 12).

If the action results in the RIS being turned off unnecessarily, or if it violates operational constraints such as flying outside the designated area or colliding with other UAVs, the action is canceled, penalties are applied, and the state is updated accordingly (*see* Lines 13–16). Otherwise, the reward $r_u[t]$ is calculated based on the system's objectives and the UAV transitions to the next state $s[t+1]$ (*see* Lines 18–19). The transition tuple $(s[t], a_u[t], r_u[t], s[t+1])$ is stored in the replay buffer $\mathbb{B}_u$ (*see* Line 20). After collecting a mini-batch of $\mathcal{M}$ experiences from the replay buffer, the algorithm updates the neural networks (Lines 21–24). The Critic network Q_u is updated by minimizing the mean-squared Bellman error between the predicted and target Q-values (*see* Line 22). The Actor network π_u is updated using the policy gradient approach to maximize the expected return (*see* Line 23). To ensure stability, the weights of the target networks are softly updated towards the weights of the main networks using a parameter ν (*see* Line 24). This approach allows each UAV to learn and optimize its policy independently while taking into account the actions and states of other UAVs in the system, leading to coordinated and efficient UAV operations.

Algorithm 2: DIRECT Algorithm

```
1:  Initialize Actor network $\pi_u(o_u[t]|\varpi^{\pi_u})$ and Critic network
    $Q_u(s[t], a_u[t]|\varpi^{Q_u})$ with random weights $\varpi^{\pi_u}$ and $\varpi^{Q_u}$
2:  Initialize target networks $\pi_u'(o_u[t]|\varpi^{\pi_u'})$ and $Q_u'(s[t], a_u[t]|\varpi^{Q_u'})$ with
    weights $\varpi^{\pi_u'} \leftarrow \varpi^{\pi_u}$ and $\varpi^{Q_u'} \leftarrow \varpi^{Q_u}$
3:  Initialize replay buffer $\mathbb{B}_u = \emptyset$ with capacity $B$
4:  Initialize exploration noise $\varepsilon$
5:  Set the number of episodes $EP$ to $M$
6:  Set the number of time steps per episode $T$
7:  **for** $EP = 1$ **to** $M$ **do**
8:      Initialize the position and residual energy of each UAV $u \in \mathcal{U}$
9:      Receive initial observation state $o_u[0]$
10:     **for** $t = 1$ **to** $T$ **do**
11:         **Select** an action $a_u[t] = \pi_u(o_u[t]|\varpi^{\pi_u}) + \varepsilon$
12:         **Execute** the action $a_u[t] = [\psi_u[t], h_u[t], d_u[t], \theta_u[t]]$
13:         **if** $\rho_u[t] > 0$ **then**
14:             **Cancel** the action $a_u[t]$
15:             **Update** $s[t]$ to $s[t+1]$ with penalties applied
16:         **else**
17:             Calculate rewards $r_u[t]$ using (31)
18:             Transition to next state $s[t+1]$
19:             **Store** transition $(s[t], a_u[t], r_u[t], s[t+1])$ in replay buffer $\mathbb{B}_u$
20:         **end if**
21:         **Sample** a random mini-batch of $\mathcal{M}$ transitions
            $(s[m], a_u[m], r_u[m], s[m+1])$ from $\mathbb{B}_u$
22:         **Set** target value
            $\mathcal{TV}_u[m] = r_u[m] + \gamma Q_u'(s[m+1], \pi_u'(o_u[m+1]|\varpi^{\pi_u'})|\varpi^{Q_u'})$
23:         **Update** the Critic network $Q_u(s[t], a_u[t]|\varpi^{Q_u})$ by minimizing
            the loss:
24:
            $Loss(\varpi^{Q_u}) = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} (Q_u(s[m], a_u[m]|\varpi^{Q_u}) - \mathcal{TV}_u[m])^2$
25:         **Update** the Actor network $\pi_u(o_u[t]|\varpi^{\pi_u})$ using the sampled
            policy gradient:
26:         $\nabla_{\varpi^{\pi_u}} J(\varpi^{\pi_u}) \approx \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \nabla_{a_u} Q_u(s[t], a_u[t]|\varpi^{Q_u})|_{a_u=\pi_u(o_u[m])}$
            $\nabla_{\varpi^{\pi_u}} \pi_u(o_u[m]|\varpi^{\pi_u})$
27:         **Update** the target networks:
28:                 $\varpi^{Q_u'} \leftarrow \nu\varpi^{Q_u} + (1-\nu)\varpi^{Q_u'}$
29:                 $\varpi^{\pi_u'} \leftarrow \nu\varpi^{\pi_u} + (1-\nu)\varpi^{\pi_u'}$
30:     **end for**
31: **end for**
```

4.3. Complexity analysis

In this section, we analyze the computational complexity of our proposed optimization algorithms in two main parts: (i) Task Offloading Optimization and (ii) UAV Movement and RIS Control Optimization.

4.3.1. Task offloading complexity

The complexity of the task offloading optimization algorithm is evaluated with respect to three primary parameters: (i) computational complexity per iteration, (ii) memory complexity, and (iii) convergence rate. This task offloading problem is formulated as a potential game, which is solved iteratively to find the Nash Equilibrium. The computational complexity per iteration is influenced by the number of GDs G. For each GD, the algorithm evaluates the offloading decision and updates the potential function. The main operations in each iteration include the calculation of the cost function and the potential function, as well as updating the offloading decision. The computational complexity per iteration, denoted as Cto, can be approximated as:

$$C_{to} = \mathcal{O}(G) \tag{36}$$

This complexity accounts for the evaluation of offloading decisions and the calculation of associated costs and potential functions for each GD. The convergence rate depends on the specific potential game framework employed. The algorithm iteratively adjusts the offloading decisions of GDs until a Nash Equilibrium is reached, where no GD can further reduce its cost by unilaterally changing its decision. The number of iterations to reach convergence is generally polynomial in the number of GDs. Therefore, the overall complexity in terms of convergence is:

$$C_{con} = \mathcal{O}(\text{iterations to converge} \times Cto) \tag{37}$$

4.3.2. UAV control optimization complexity

The DIRECT algorithm's complexity is evaluated with respect to three primary parameters: (i) training time complexity, (ii) dynamic space complexity, and (iii) neural network complexity in terms of computing. Firstly, the training time complexity of the actor and critic networks mainly depends on several factors: the number of steps T per episode, the number of episodes M, and the total number of UAVs $|\mathcal{U}|$. In each episode, every UAV executes a series of T steps. At each of these steps, the critic network is refined by minimizing the mean-squared Bellman error $Loss(\varpi^{Q_u})$. The actor network is refined using the policy gradient $\nabla_{\varpi^{\pi_u}} J(\varpi^{\pi_u})$. Hence, we can provide an approximate expression for the training time complexity per episode, denoted as $\hbar_{ttc}$, as follows:

$$\hbar_{ttc} = \mathcal{O}(T \times |\mathcal{U}| \times (|\mathcal{S}| \times |\mathcal{A}|)), \tag{38}$$

where $|\mathcal{S}|$ denotes the size of the state space, and $|\mathcal{A}|$ indicates the size of the action space. Additionally, the dynamic space complexity of the DIRECT algorithm depends on the memory needed for the replay buffer, actor networks, critic networks, and target networks. The replay buffer $\mathbb{B}_u$ retains transitions and its size is denoted as $|\mathbb{B}_u|$, relying on the number of steps per episode T and the total number of UAVs $|\mathcal{U}|$. Thus, we can provide an approximate representation of the dynamic space complexity of the replay buffer per episode, denoted as $\hbar_{dsc}$, as follows:

$$\hbar_{dsc} = \mathcal{O}(T \times |\mathcal{U}| \times (|\mathcal{S}| \times |\mathcal{A}|) \times |\mathbb{B}_u|). \tag{39}$$

The configuration of the neural networks in the DIRECT framework can significantly impact the performance and learning capabilities of the algorithm. Therefore, the neural network's complexity in storing the actor networks, critic networks, and target networks is contingent on the number of parameters in each network. This number is often influenced by factors like the size of the state space ($|\mathcal{S}|$) and action space ($|\mathcal{A}|$), as well as the number of hidden layers and neurons. To assess this complexity, let us define ι_{ac_z} as the zth layer's count of neurons of the actor network. Similarly, ι_{cr_w} represents the wth layer's number of neurons of the critic network. Therefore, the neural networks' complexity $\hbar_{nnc}$ in terms of computing per episode is expressed as follows:

$$\hbar_{nnc} = \mathcal{O}\Bigg(T \times M \times \mathcal{N} \times |\mathcal{U}| \times (|\mathcal{S}| \times |\mathcal{A}|) \times \left(\sum_{z=1}^{\mathfrak{I}} \iota_{ac_y} \iota_{ac_{y+1}} \times \sum_{w=1}^{\mathfrak{R}} \iota_{cr_w} \iota_{cr_{w+1}} \right) \Bigg) \tag{40}$$

where $\mathcal{N}$ represents the process of retrieving experiences from the replay buffer $\mathbb{B}_u$. $\mathfrak{I}$ denotes the quantity of layers in the actor network, whereas $\mathfrak{R}$ denotes the layer count in the critic network.

5. Results analysis

In this section, we validate the performance of the DIRECT framework through a detailed analysis of the numerical results obtained from our simulations. The process is divided into two main stages: the learning phase and the testing phase. In the learning phase, we focus on the offline centralized training and convergence of DIRECT, comparing it with other Deep RL variants such as single-agent Double DQN (S-DDQN) (Van Hasselt et al., 2016) and multi-agent DQN (MADQN) (Egorov, 2016). Once convergence is achieved, the parameters of the neural networks are stored and utilized during the testing phase. Here, we compare the performance of our framework with the DRL variants and baseline methods like random and greedy approaches. This section discusses our framework's simulation and training settings, followed by a detailed analysis of the results from both phases.

5.1. Environment configurations

The performance evaluation of our framework is carried out through extensive simulations using Python 3.8.10 and TensorFlow 2.8.0. The training phase comprises 1000 episodes, each consisting of 100 time steps. The actor and critic networks each have four fully connected hidden layers. Each layer has a specified number of neurons, utilizing the Rectified Linear Unit (ReLU) activation function. The output layer of the actor network employs the hyperbolic tangent (tanh) activation function to constrain the UAVs' flight paths to their maximum allowable travel distance d_{max}. The tanh activation function is used to provide continuous outputs in the range $[-1, 1]$, enabling flexible modeling of both positive and negative decision values essential for dynamic UAV control. The critic network takes in a combination of observations and actions, producing a scalar output that evaluates the states according to the overall policy. Table 3 details the parameters of the neural networks used during the learning phase.

Our simulations are conducted over a square disaster area of maximum width $\mathbb{D}_{max} = 10$ km, covering an area of 100 km^2. Within this region, 100 GDs move randomly and are covered by 16 UAVs. Airship$_b$ is deployed over this region at an altitude of 10 km. Table 4 details the main simulation parameters utilized in our work.

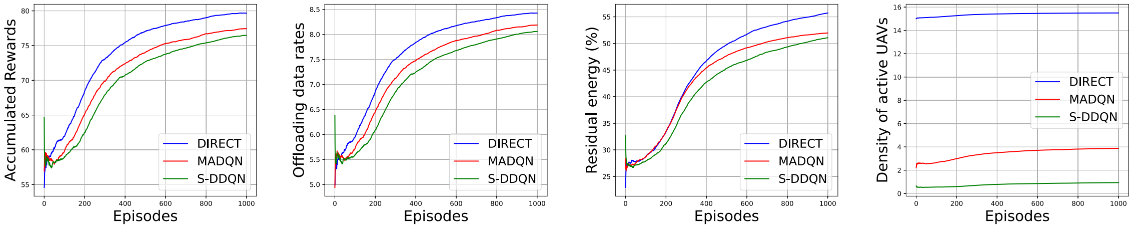
5.2. Learning phase

In the learning phase, the system leverages random experiences to adapt and optimize its performance. As depicted in Fig. 7, we plot the rewards obtained by UAVs at each episode. The rewards increase consistently until convergence around the 200th episode. This trend indicates that the DIRECT framework effectively leverages the random experiences gathered during the exploration phase, allowing the system to adapt swiftly to the dynamic movements of GDs and avoid penalties after approximately 200 episodes. It is essential to highlight that while rewards exhibit fluctuations around their average values in a constantly changing environment, there is an overall upward trend as learning progresses. This behavior reflects the system's ability to improve its performance and decision-making capabilities over time, resulting in more efficient energy management and better task offloading strategies.

The convergence results of the DIRECT framework in terms of rewards, offloading rates, and energy, are illustrated on the curves of Fig. 6. As depicted in Fig. 6(a), the accumulated rewards for the DIRECT framework show a clear upward trend, indicating effective learning and adaptation over time. DIRECT achieves higher reward levels more quickly and stabilizes faster than MADQN and S-DDQN, reflecting its superior ability to optimize task offloading and energy management

Table 3
Parameters of the Neural Networks.

Neural Actor Network Parameters			
Layers	**Number**	**Size**	**Activation Function**
Input	1	$3 + 3G$	–
Hidden	4	400, 400, 300, 300	ReLU
Output	1	4	Tanh
Neural Critic Network Parameters			
Layers	**Number**	**Size**	**Activation Function**
Input	1	$U \times (3 + 3G)$	–
Hidden	4	400, 400, 300, 300	ReLU
Output	1	1	–
Training Stage Parameters			
Parameter		**Value**	
Size of memory $\mathbb{B}_u$		10^5	
Size of mini-batch $\mathcal{M}$		256	
Learning rate of actor		0.0002	
Learning rate of critic		0.0001	
Optimizer method		Adam	
Updating steps		1000	
Reward discount factor γ		0.99	
Λ_1, Λ_2, Λ_3, Λ_4 and Λ_5		50	
Comparative DRL methods		MADQN, S-DDQN	



(a) Accumulated rewards. (b) Offloading rates. (c) Residual energy of UAVs (%). (d) Density of active UAVs.

Fig. 6. Convergence results in terms of rewards, offloading rates, average residual energy, and density of active UAVs.

Table 4
Simulation Parameters.

Environment Parameters			
Parameter	**Description**	**Value**	**Units**
Area	Disaster area surface	100	km^2
$\mathbb{D}_{max}$	Surface width	10	km
h_b	Height of Airship$_b$	10	km
P_g	Transmit Power of GD$_g$	−20–20	dBm
V_u^{max}	Maximum velocity of UAV$_u$	15	m/s
V_g^{max}	Maximum velocity of GD$_g$	1.5	m/s
η	Path loss exponent	2	Dimensionless
ξ	Energy conversion efficiency of devices	0.2	Dimensionless
$M \times N$	Number of reflecting elements	100 × 100	Elements
ϱ_i	Phase shift of the ith element	$[0, 2\pi)$	Radians
Θ_{ris}	Phase shift matrix of RIS	10 × 10	Dimensionless
P_{ris}	Power consumption per reflecting element	0.1	W
σ^2	Noise power	−100	dBm
Edge Computing Parameters			
Parameter	**Description**	**Value**	**Units**
$D_g[t]$	Computing task size of GD$_g$	5–150	MB
C_g	CPU Cycles number of GD$_g$	500–4000	Cycles/bit
$f_g[t]$	Computation frequency of GD$_g$	2.0	GHz
κ_g	Processor efficiency of GD$_g$	10^{-11}	$\text{J/Hz}^{\beta-1}$
β	Power consumption exponent	2.0	Dimensionless

processes. This trend is attributed to the efficient exploitation of experiences collected during exploration and the superior coordination of UAVs and RIS control in the DIRECT framework. The curve clearly shows that the DIRECT algorithm converges faster and achieves higher stability in reward accumulation, proving its effectiveness in handling complex tasks. Fig. 6(b) demonstrates the offloading rates for different frameworks. DIRECT consistently achieves higher offloading rates compared to its competitors, indicating its efficiency in managing and

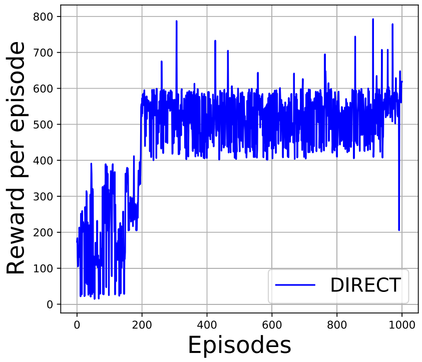


Fig. 7. Reward per episode of DIRECT framework.

processing the computational tasks of GDs. The slow convergence of MADQN and the single-agent architecture of S-DDQN delay appropriate action execution, leading to lower offloading rates. DIRECT's performance highlights its ability to maintain high offloading efficiency even as task complexity increases. In Fig. 6(c), the residual energy levels of UAVs in the DIRECT framework remain higher compared to MADQN and S-DDQN, indicating more efficient energy usage and better energy harvesting strategies. This efficient energy management ensures that UAVs can sustain longer operational periods without frequent recharging, contributing to overall system reliability. As for Fig. 6(d), it shows the density of active UAVs. DIRECT maintains a consistently high density of active UAVs, ensuring robust coverage and support for GDs throughout the mission. The slower convergence and less effective coordination in MADQN and S-DDQN result in lower densities of active UAVs. DIRECT's ability to dynamically adjust UAV positions and RIS states ensures continuous and optimal coverage.

5.3. Testing phase

In Fig. 8, we evaluate the performance and robustness of the DIRECT framework under real-world scenarios by leveraging the trained neural network parameters. Figs. 8(a) and 8(e) illustrate the offloading rates over time and their statistical distribution, respectively. In fact, we clearly observe that the offloading rates for the DIRECT framework consistently increase and stabilize over time, indicating the system's efficient adaptation and learning in dynamic environments. This behavior is significantly better compared to MADQN and S-DDQN, which exhibit slower and less stable increases due to their single-agent architecture and inherent limitations in handling continuous actions, respectively. Moreover, the two figures show that DIRECT achieves higher median offloading rates with lower variability, demonstrating its robustness and reliability in maintaining efficient offloading performance. The superior performance of DIRECT can be attributed to its multi-agent reinforcement learning strategy that optimally coordinates UAV movements and RIS control, leading to improved coverage and data transmission efficiency.

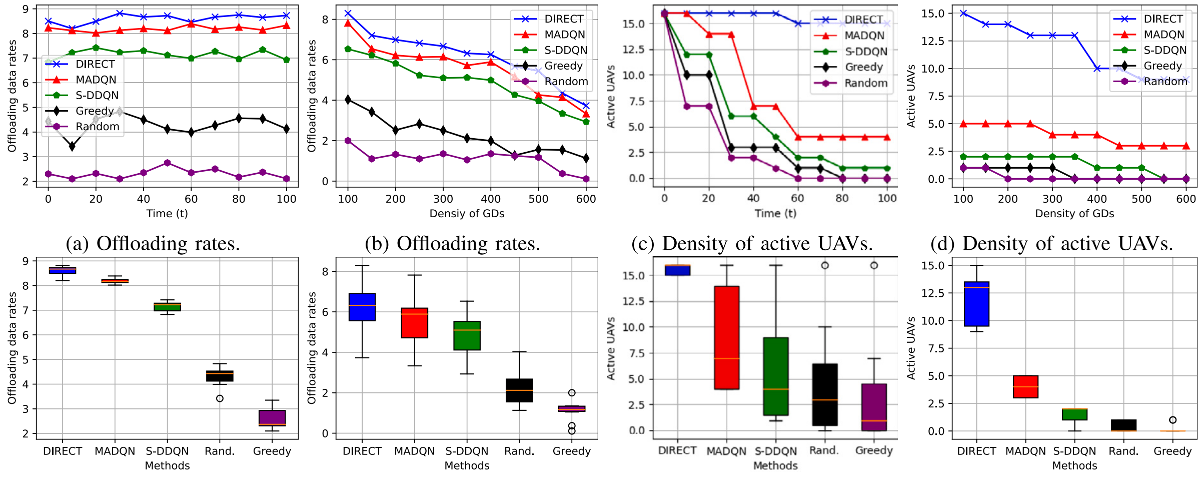
In Figs. 8(b) and 8(f), DIRECT demonstrates a higher and more consistent offloading rate across all GDs compared to MADQN and S-DDQN. This indicates that DIRECT effectively manages resource allocation and GD coverage, ensuring that each GD can offload its computational tasks efficiently. Moreover, DIRECT not only achieves higher median offloading rates but also exhibits less variability among GDs, highlighting its equitable distribution of resources and consistent performance. The performance disparity between DIRECT and its competitors can be explained by the slow convergence of MADQN due to its inability to handle continuous actions effectively, and the delayed action execution in S-DDQN and DQN due to their single-agent architectures.

Figs. 8(c) and 8(g) show the density of active UAVs over time and their statistical distribution, respectively. We clearly see that the density of active UAVs under the DIRECT framework quickly stabilizes, maintaining a high level of activity throughout the simulation. This stability is a testament to DIRECT's effective management of UAV resources, ensuring continuous operation and optimal coverage of the disaster area. Moreover, the results show a high median density of active UAVs with low variability, indicating consistent and reliable UAV deployment. In contrast, MADQN and S-DDQN struggle to maintain a stable density of active UAVs due to their less efficient coordination and resource management. The superior performance of DIRECT can be attributed to its advanced control policy that dynamically adjusts UAV positions and RIS states, ensuring optimal energy usage and coverage.

As for Figs. 8(d) and 8(h), they represent the density of active UAVs in relation to the number of GDs and their statistical distribution, respectively. Indeed, the DIRECT framework consistently maintains a high density of active UAVs relative to the number of GDs, ensuring that all GDs are effectively covered and supported. Moreover, DIRECT shows a high median density with low variability, highlighting its robustness in various GD density scenarios. The inefficiency of MADQN and S-DDQN in maintaining a high density of active UAVs can be attributed to their slower convergence rates and less effective action coordination. DIRECT's superior performance stems from its ability to dynamically optimize UAV deployment and RIS operation, ensuring efficient energy distribution and continuous GD coverage.

Figs. 9(a) and 9(b) compare latency and energy consumption as functions of GD density per km^2, showcasing our system's superior performance over Esposito et al. (2019) and Wang et al. (2022). Our system consistently achieves lower latency and energy consumption, particularly in high-density environments, due to dynamic task offloading and RIS-based energy distribution. In contrast, Esposito et al. (2019) focuses on communication quality, leading to higher energy use, while Wang et al. (2022) struggles with large-scale deployments, resulting in increased latency and energy consumption due to its static task offloading model. Our system's dynamic UAV movement and RIS optimization ensure efficient resource use and reduced delays, making it more scalable for large-scale disaster recovery scenarios.

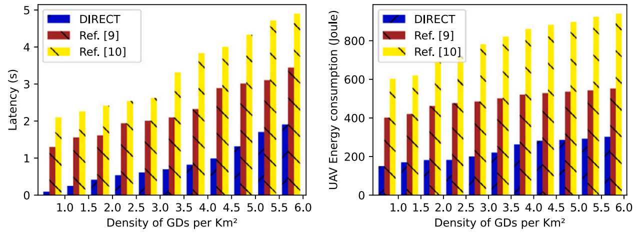
In Fig. 10, a set of results related to the optimization of the edge computing part of our system. Indeed, we compare the overall efficiency of different task processing strategies, namely (i) GT-NE (Game Theory-based Nash Equilibrium) adopted by our system, (ii) local processing, and (iii) remote processing. In Fig. 10(a), the results clearly indicate that the GT-NE strategy results in the lowest average system cost, showcasing its superior optimization in balancing offloading and local task processing. Local processing exhibits higher system costs due to significant energy consumption and latency associated with high computational complexity, while remote processing incurs initial offloading costs but remains more advantageous for computationally intensive tasks as it manages to keep system costs lower compared to local processing. This trend underscores the efficacy of GT-NE in minimizing system costs by efficiently distributing tasks between local and remote resources. Fig. 10(b) depicts the average system cost as a function of the CPU cycles for the three task processing strategies. Our game theory-based strategy shows an optimized decision-making process where GDs balance offloading tasks to Airship$_b$ and local processing, leading to a slower initial cost increase. However, the local processing curve rises steeply as C_g increases, highlighting significant energy consumption and latency with higher computational complexity, making local processing inefficient. As for the remote processing, the curve starts higher due to initial offloading costs but increases more slowly with higher C_g, suggesting offloading becomes more advantageous for computationally intensive tasks despite initial communication costs. Fig. 10(c) shows the system cost as a function of data sizes. The GT-NE strategy consistently maintains the lowest system costs, demonstrating its effectiveness in balancing task offloading and local processing even as data sizes increase. Local processing shows a steep cost increase with larger data



(a) Offloading rates. (b) Offloading rates. (c) Density of active UAVs. (d) Density of active UAVs.

(e) Offloading rates. (f) Offloading rates. (g) Density of active UAVs. (h) Density of active UAVs.

Fig. 8. Performance of DIRECT in terms of offloading rates and density of active UAVs according to time and density of GDs.



(a) Latency. (b) UAV energy consumption.

Fig. 9. Performance of DIRECT vs. Esposito et al. (2019) and Wang et al. (2022).

sizes, reflecting the high energy consumption and latency associated with processing large datasets locally. Conversely, remote processing, while incurring higher initial costs due to offloading, increases more gradually, making it a better option for handling large data sizes. These results highlight GT-NE's superior capability in optimizing system costs across different data sizes, ensuring efficient resource utilization and cost-effectiveness.

6. Conclusion

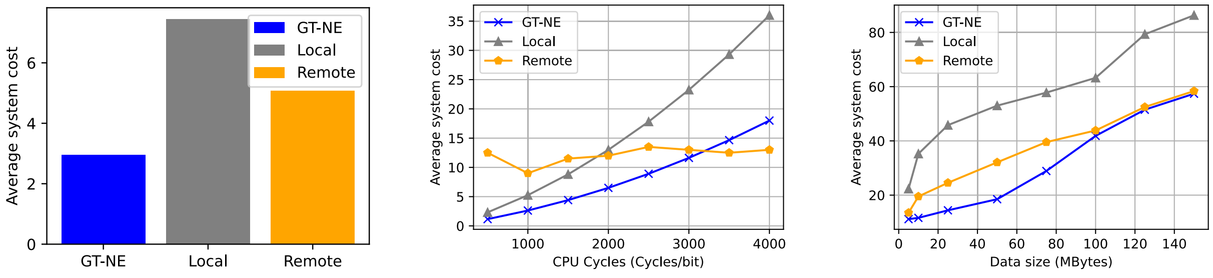
In this paper, we presented a comprehensive framework that leverages an airship and UAVs embedded with RISs to enhance disaster response capabilities in 6G networks. Our system effectively addresses the critical challenges of maintaining low latency (*i.e.,* high offloading rates and efficient data processing), minimizing energy consumption, and ensuring reliable connectivity in disrupted environments. By integrating optimization techniques such as game theory for task offloading and MARL for UAV movement and RIS control, we optimized edge computing processes and UAV trajectories, significantly improving the overall system performance. The simulation results demonstrated substantial enhancements in energy efficiency, latency reduction, and QoS-aware coverage compared to conventional methods. Specifically, the airship, functioning as a central node for energy distribution and edge computing, combined with UAV-mounted RISs, enabled efficient and reliable communication, even in challenging environments. The dynamic adjustment of UAV locations and RIS configurations further ensured optimal coverage and resource allocation, reducing the processing burden on GDs and enhancing operational efficiency in disaster response scenarios. This system's ability to dynamically adapt to changing conditions and optimize computational tasks and energy resources underscores its potential for practical deployment.

6.1. Discussion

While the proposed system demonstrates strong performance in simulated disaster response scenarios, several important aspects remain open for discussion. One key challenge lies in the scalability of the framework when deployed in large-scale disaster zones with high densities of GDs and UAVs. The coordination complexity, increased communication overhead, and real-time task management across many agents can strain system resources. Although the MARL framework allows decentralized control, scaling it up may require model compression techniques such as Tiny Reinforcement Learning (TinyRL) and efficient neural architectures to ensure energy efficiency and reduced computational burden. Furthermore, while our current simulations validate the feasibility of the proposed system, field trials and hardware-in-the-loop testing are necessary to assess real-world performance under dynamic environmental conditions, interference, and regulatory constraints. Addressing these challenges is crucial for transitioning from conceptual validation to actual deployment.

6.2. Future works

Future research will focus on improving the scalability of the proposed framework to support large-scale, dense disaster response scenarios. A key challenge for this will be integrating lightweight learning architectures and distributed training mechanisms to handle high volumes of UAVs, ground devices (GDs), and dynamic task offloading in real-time. Another critical direction for future work is real-world testing in diverse terrains and environmental conditions, which will help validate the robustness and adaptability of the UAV-HAP-RIS framework under practical disaster scenarios. While our simulations have demonstrated strong results, field trials will provide valuable insights into operational performance and help identify potential discrepancies between simulated and real-world outcomes. Additionally, we plan to tackle the challenges of real-time dynamic CSI acquisition, a crucial factor for efficient RIS beamforming and communication optimization. The dynamic nature of the disaster environment necessitates adaptive mechanisms that can adjust to rapidly changing wireless channels. To improve communication efficiency, future work will focus on developing adaptive techniques for real-time CSI acquisition and integration with RIS technologies. Furthermore, a promising direction for future research involves integrating our framework with broader disaster-response infrastructure, such as terrestrial base stations, emergency vehicles, and IoT-based environmental sensors. This integration will enhance situational awareness, improve coordination, and support a more comprehensive response. We will also investigate



(a) Average system cost.

(b) System cost vs. CPU cycles.

(c) System cost vs. Data sizes.

Fig. 10. Performance of our edge-enabled system in terms of average system cost.

adaptive and predictive models for UAV trajectory planning in multi-disaster zone environments, ensuring that UAVs can efficiently navigate complex and changing disaster areas. Lastly, we aim to explore the integration of edge intelligence, which can reduce reliance on centralized processing and improve decision-making speed, particularly in resource-constrained or communication-challenged environments. This will significantly improve the overall efficiency of the disaster recovery system.

CRediT authorship contribution statement

Jamal Alotaibi: Writing – review & editing, Writing – original draft, Funding acquisition, Formal analysis. **Omar Sami Oubbati:** Writing – review & editing, Writing – original draft, Validation, Conceptualization. **Mohammed Atiquzzaman:** Writing – review & editing, Writing – original draft, Validation, Conceptualization. **Fares Alromithy:** Writing – review & editing, Writing – original draft, Validation. **Mohammad Rashed Altimania:** Writing – review & editing, Writing – original draft, Validation.

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.

Acknowledgment

The Researchers would like to thank the Deanship of Graduate Studies and Scientific Research at Qassim University for financial support (QU-APC-2025).

Data availability

No data was used for the research described in the article.

References

Abkenar, F.S., Ramezani, P., Iranmanesh, S., Murali, S., Chulerttiyawong, D., Wan, X., Jamalipour, A., Raad, R., 2022. A survey on mobility of edge computing networks in IoT: State-of-the-art, architectures, and challenges. IEEE Commun. Surv. & Tutorials 24 (4), 2329–2365.

Alam, M.S., Kurt, G.K., Yanikomeroglu, H., Zhu, P., DJao, N.D., 2021. High altitude platform station based super macro base station constellations. IEEE Commun. Mag. 59 (1), 103–109.

Bai, L., Han, R., Liu, J., Yu, Q., Choi, J., Zhang, W., 2020. Air-to-ground wireless links for high-speed UAVs. IEEE J. Sel. Areas Commun. 38 (12), 2918–2930.

Cabuk, U.C., Tosun, M., Dagdeviren, O., Ozturk, Y., 2024. Modeling energy consumption of small drones for swarm missions. IEEE Trans. Intell. Transp. Syst.

Caswell, M.K., 2015. Need for vertical delineation of air space: Can google's project loon survive without it. Tulane J. Int. Comp. Law 24, 205.

Dai, M., Huang, N., Wu, Y., Gao, J., Su, Z., 2022. Unmanned-aerial-vehicle-assisted wireless networks: Advancements, challenges, and solutions. IEEE Internet Things J. 10 (5), 4117–4147.

Ding, C., Wang, J.-B., Zhang, H., Lin, M., Li, G.Y., 2021. Joint optimization of transmission and computation resources for satellite and high altitude platform assisted edge computing. IEEE Trans. Wirel. Commun. 21 (2), 1362–1377.

Duo, B., He, M., Wu, Q., Zhang, Z., 2023. Joint dual-UAV trajectory and RIS design for ARIS-assisted aerial computing in IoT. IEEE Internet Things J. 10 (22), 19584–19594.

Egorov, M., 2016. Multi-agent deep reinforcement learning. In: CS231n: Convolutional neural networks for visual recognition. Stanford University Stanford, CA, USA, pp. 1–8.

Eskandari, M., Huang, H., Savkin, A.V., Ni, W., 2022. Model predictive control-based 3D navigation of a RIS-equipped UAV for LoS wireless communication with a ground intelligent vehicle. IEEE Trans. Intell. Veh. 8 (3), 2371–2384.

Esposito, C., Zhao, Z., Alcarria, R., Rizzo, G., 2019. Game theoretic optimal user association in emergency networks. In: Proceedings of the International Conference on Ad-Hoc Networks and Wireless. Springer, pp. 18–31.

Gavan, J., Tapuchi, S., Grace, D., 2009. Concepts and main applications of high-altitude-platform radio relays. URSI Radio Sci. Bull. 2009 (330), 20–31.

He, Y., Nie, L., Guo, T., Kaur, K., Hassan, M.M., Yu, K., 2022. A NOMA-enabled framework for relay deployment and network optimization in double-layer airborne access VANETs. IEEE Trans. Intell. Transp. Syst. 23 (11), 22452–22466.

Kim, S., 2024. Hierarchical aerial offload computing algorithm based on the stackelberg-evolutionary game model. Comput. Netw. 245, 110348.

Lee, W., Kim, T., 2023. Multiagent reinforcement learning in controlling offloading ratio and trajectory for multi-UAV mobile-edge computing. IEEE Internet Things J. 11 (2), 3417–3429.

Li, S., Ale, L., Chen, H., Tan, F., Quek, T.Q., Zhang, N., Dong, M., Ota, K., 2024. Joint computation offloading and multi-dimensional resource allocation in air-ground integrated vehicular edge computing network. IEEE Internet Things J. 11 (20), 32687–32700.

Li, Z., Chen, W., Cao, H., Tang, H., Wang, K., Li, J., 2022. Joint communication and trajectory design for intelligent reflecting surface empowered UAV SWIPT networks. IEEE Trans. Veh. Technol. 71 (12), 12840–12855.

Li, L., Guan, W., Zhao, C., Su, Y., Huo, J., 2023. Trajectory planning, phase shift design, and IoT devices association in flying-RIS-assisted mobile edge computing. IEEE Internet Things J. 11 (1), 147–157.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Liu, F., Dong, X., Yu, J., Hua, Y., Li, Q., Ren, Z., 2022a. Distributed Nash equilibrium seeking of N-coalition noncooperative games with application to UAV swarms. IEEE Trans. Netw. Sci. Eng. 9 (4), 2392–2405.

Liu, W., Zhang, S., Ansari, N., 2022b. Secure mobile edge computing via a drone-enabled FSO-based heterogeneous network. IEEE Trans. Veh. Technol. 71 (11), 12264–12274.

Liwang, M., Wang, X., 2022. Overbooking-empowered computing resource provisioning in cloud-aided mobile edge networks. IEEE/ACM Trans. Netw. 30 (5), 2289–2303.

McEnroe, P., Wang, S., Liyanage, M., 2022. A survey on the convergence of edge computing and AI for UAVs: Opportunities and challenges. IEEE Internet Things J. 9 (17), 15435–15459.

Nguyen, M.T., Le, L.B., 2022. Multi-UAV trajectory control, resource allocation, and NOMA user pairing for uplink energy minimization. IEEE Internet Things J. 9 (23), 23728–23740.

Qi, W., Yang, C., Song, Q., Guan, Y., Guo, L., Jamalipour, A., 2024. Minimizing age of information for hybrid UAV-ris-assisted vehicular networks. IEEE Internet Things J. 11 (10), 17886–17895.

Ramakrishnan, K., Yuksel, M., Seferoglu, H., Chen, J., Blalock, R.A., 2022. Resilient communication for dynamic first responder teams in disaster management. IEEE Commun. Mag. 60 (9), 93–99.

Samir, M., Elhattab, M., Assi, C., Sharafeddine, S., Ghrayeb, A., 2021. Optimizing age of information through aerial reconfigurable intelligent surfaces: A deep reinforcement learning approach. IEEE Trans. Veh. Technol. 70 (4), 3978–3983.

Tyrovolas, D., Mekikis, P.-V., Tegos, S.A., Diamantoulakis, P.D., Liaskos, C.K., Karagiannidis, G.K., 2022. Energy-aware design of UAV-mounted RIS networks for IoT data collection. IEEE Trans. Commun. 71 (2), 1168–1178.

Van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double q-learning. In: Proc. of the AAAI Conf. on Artificial Intelligence, vol. 30, (no. 1).

Wang, Y., Chen, W., Luan, T.H., Su, Z., Xu, Q., Li, R., Chen, N., 2022. Task offloading for post-disaster rescue in unmanned aerial vehicles networks. IEEE/ACM Trans. Netw. 30 (4), 1525–1539.

Wang, Z., Liu, L., Cui, S., 2020. Channel estimation for intelligent reflecting surface assisted multiuser communications: Framework, algorithms, and analysis. IEEE Trans. Wirel. Commun. 19 (10), 6607–6620.

Wang, Y., Zhang, C., Ge, T., Pan, M., 2024. Computation offloading via multi-agent deep reinforcement learning in aerial hierarchical edge computing systems. IEEE Trans. Netw. Sci. Eng.

Wu, C., Ke, F., Yang, X., Wen, M., Li, D., Zhang, X., 2023. Joint energy and information precoding for NOMA-based WPCNs aided by reconfigurable intelligent surface. IEEE Trans. Veh. Technol. 72 (11), 14559–14572.

Yang, H., Lin, K., Xiao, L., Zhao, Y., Xiong, Z., Han, Z., 2024. Energy harvesting UAV-ris-assisted maritime communications based on deep reinforcement learning against jamming. IEEE Trans. Wirel. Commun.

Yang, H., Liu, S., Xiao, L., Zhang, Y., Xiong, Z., Zhuang, W., 2023. Learning-based reliable and secure transmission for UAV-ris-assisted communication systems. IEEE Trans. Wirel. Commun.

Zhang, H., Di, B., Song, L., Han, Z., 2020. Reconfigurable intelligent surfaces assisted communications with limited phase shifts: How many phase shifts are enough? IEEE Trans. Veh. Technol. 69 (4), 4498–4502.

Zhang, H., Huang, M., Zhou, H., Wang, X., Wang, N., Long, K., 2022a. Capacity maximization in RIS-uav networks: a DDQN-based trajectory and phase shift optimization approach. IEEE Trans. Wirel. Commun. 22 (4), 2583–2591.

Zhang, L., Wang, Y., Min, M., Guo, C., Sharma, V., Han, Z., 2022b. Privacy-aware laser wireless power transfer for aerial multi-access edge computing: a colonel blotto game approach. IEEE Internet Things J. 10 (7), 5923–5939.

Zhao, Y., Zhou, F., Feng, L., Li, W., Sun, Y., Imran, M.A., 2023. Backhaul-constrained coverage analysis of integrated high and low altitude platforms aerial communication system in post-disaster areas. IEEE Commun. Lett. 27 (6), 1629–1633.

Jamal Alotaibi received the B.S. degree in Computer engineering from Qassim University, in 2012. He joined Wayne state university 2016 and received M. S. degree in Electrical and Computer Engineering. He worked in STC the telecommunication company as networks engineering. Dr. Alotaibi worked as Software engineering at Ford Company MI, from 2018 till 2022. In 2022, he received his Ph.D. degree from Electrical and Computer Engineering Department, College of Engineering, Wayne State University. Dr. Alotaibi research interests include Internet of Things (IoT) applications and security, computer networks, wireless and mobile networks, machine learning, and cybersecurity. Dr. Alotaibi currently working as Assistant Professor in department of computer engineering at Qassim university.

Omar Sami Oubbati is an Associate Professor at the University Gustave Eiffel in the region of Paris, France. He is a member of the Gaspard Monge Computer Science laboratory (LIGM CNRS UMR 8049). He received his degree of Engineer (2010), M.Sc. in Computer Engineering (2011), M.Sc. degree (2014), and a Ph.D. in Computer Science (2018), all from University of Laghouat, Algeria. From Oct. 2016 to Oct. 2017, he was a Visiting Ph.D. Student with the Laboratory of Computer Science, University of Avignon, France. He spent 6 years as an Assistant Professor at the Electronics department, University of Laghouat, Algeria and a Research Assistant in the Computer Science and Mathematics Lab (LIM) at the same university. His main research interests are in Flying and Vehicular ad hoc networks, Energy harvesting and Mobile Edge Computing, Energy efficiency and Internet of Things (IoT). He is the recipient of the 2019 Best Survey Paper for Vehicular Communications (Elsevier). He has actively served as a reviewer for flagship IEEE Transactions journals and conferences, and participated as a Technical Program Committee Member for a variety of international conferences, such as IEEE ICC, IEEE CCNC, IEEE ICCCN, IEEE WCNC, IEEE ICAEE, and IEEE ICAIT. He serves on the editorial board of Vehicular Communications Journal of Elsevier and Communications Networks Journal of Frontiersin. He has also served as guest editor for a number of international journals. He is a member of the IEEE and IEEE Communications Society.

Mohammed Atiquzzaman received the M.S. and Ph.D. degrees in electrical engineering and electronics from the University of Manchester, U.K., in 1984 and 1987, respectively. He currently holds the Edith J. Kinney Gaylord Presidential Professorship with the School of Computer Science, University of Oklahoma, USA. His research has been funded by the National Science Foundation, National Aeronautics and Space Administration, U.S. Air Force, Cisco, and Honeywell. He coauthored Performance of TCP/IP Over ATM Networks and has authored more than 300 refereed publications. His current research interests include areas of transport protocols, wireless and mobile networks, ad hoc networks, satellite networks, power-aware networking, and optical communications. He Co-Chaired the IEEE High Performance Switching and Routing Symposium (2003, 2011), IEEE GLOBECOM and ICC (2014, 2012, 2010, 2009, 2007, and 2006), IEEE VTC (2013), and SPIE Quality of Service Over Next Generation Data Networks conferences (2001, 2002, and 2003). He was the Panels Co-Chair of INFOCOM'05, and has been on the program committee of many conferences, such as INFOCOM, GLOBECOM, ICCCN, ICCIT, Local Computer Networks, and serves on the review panels at the National Science Foundation. He was the Chair of the IEEE Communication Society Technical Committee on Communications Switching and Routing. He received the IEEE Communication Society's Fred W. Ellersick Prize and the NASA Group Achievement Award for outstanding work to further NASA Glenn Research Center's efforts in the area of the Advanced Communications/Air Traffic Management's Fiber Optic Signal Distribution for Aeronautical Communications project. He received from IEEE the 2018 Satellite and Space Communications Technical Recognition Award for valuable contributions to the Satellite and Space Communications scientific community. He also received the 2017 Distinguished Technical Achievement Award from IEEE Communications Society in recognition of outstanding technical contributions and services in the area of communications switching and routing. He is the Editor in Chief of Journal of Networks and Computer Applications, the founding Editor in Chief of Vehicular Communications, and serves served on the editorial boards of many journals, including IEEE Communications Magazine, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, Real Time Imaging Journal, Journal of Sensor Networks, and International Journal of Communication Systems.

Fares Alromithy is an Assistant Professor of Electrical and Computer Engineering at the University of Tabuk, Saudi Arabia. He earned his B.S. in Electrical Engineering from Indiana University — Purdue University Indianapolis in 2009, followed by an M.S. from Wayne State University, and a Ph.D. from Oakland University. Dr. Alromithy's research explores advanced smart sensors, microelectromechanical systems (MEMS), nanoelectronics, and integrated microsystems. His work focuses on developing miniature, innovative sensors with applications in sectors like healthcare, telecommunications, and environmental monitoring.

Mohammad Rashed Altimania is an Assistant Professor at University of Tabuk, Saudi Arabia. In March 2010, He completed his Bachelor of Science Degree in Electrical Engineering from Qassim University. He obtained a Master of Science degree in Electrical Engineering from the University of Tennessee at Chattanooga in May 2014. He received his Ph.D. Degree in Electrical Engineering from Missouri University of Science and Technology (Rolla, MO, USA) in 2020. His current research interests include Modeling and Design of Power Electronic Converters, Switched-Capacitor Converters, Power and Energy systems, Renewable Energy Harvesting (Wind and Solar Energy), Smart Grids, Virtual Power Plants, Distributed Generations, and Solar Desalination.