IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 June 2025, accepted 26 June 2025, date of publication 30 June 2025, date of current version 10 July 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3584645

RESEARCH ARTICLE

Toward Energy Efficiency and Fairness in UAV-Based Task Offloading

MOHAMED EL-EMARY[1], ALI RANJHA[1], DIALA NABOULSI[1], (Member, IEEE), AND RAZVAN STANICA[2]

[1]Department of Software Engineering and IT, École de Technologie Supérieure, Montreal, QC H3C 1K3, Canada
[2]Inria, CITI, INSA Lyon, 69621 Villeurbanne, France

Corresponding author: Mohamed El-Emary (mohamed-ibrahim-mahmoud.el-emary.1@ens.etsmtl.ca)

This work was supported by Mitacs/Ultra Intelligence and Communications under Project IT25839.

ABSTRACT The rising demand for compute-intensive mobile applications challenges the limited energy and processing power of user equipment (UE). While Mobile Edge Computing (MEC) enables task offloading to nearby servers, deploying fixed MEC infrastructure is often impractical in settings like disaster zones or temporary high-density events. Furthermore, challenges such as high task delays, limited UE battery life, and unfair load distribution persist. To address these issues, we propose a system where Unmanned Aerial Vehicles (UAVs) serve as mobile relays between UEs and MEC servers. This results in a joint optimization framework combining 1) a Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm for UAV trajectory control to enhance service coverage and energy efficiency, with 2) a low-complexity task offloading algorithm for UEs. The framework is explicitly designed to minimize UE energy consumption while promoting fairness in task allocation and data rates. Simulations demonstrate that our approach significantly outperforms state-of-the-art benchmarks, reducing UE energy consumption by 25–30% and improving fairness indices by up to 90%. The proposed system proves scalable and robust, making it suitable for real-time deployment in resource-constrained environments with dynamic workloads.

INDEX TERMS Mobile edge computing, task offloading, trajectory design, unmanned aerial vehicle.

I. INTRODUCTION

The number of connected devices is steadily rising, with forecasts predicting over 9 billion mobile subscriptions and more than 40 billion Internet of Things (IoT) devices by 2030 [1]. These devices are expected to support a wide range of services demanding intensive computations—such as augmented reality [2] and machine learning [3]—that challenge their portability and energy efficiency. To address this gap, task offloading has emerged [4], allowing mobile and IoT devices to transfer computational tasks to dedicated infrastructure. Meanwhile, networking architectures are evolving to bring computing closer to users through paradigms like mobile edge computing (MEC) [5] and fog computing [6]. Integrating task offloading with MEC is thus a natural fit [4], offering a viable solution for services requiring high computational power, low latency, and energy efficiency.

However, in many scenarios, deploying fixed computing and communication infrastructure near the UE is impractical or prohibitively expensive. Examples include large-scale IoT deployments [7], disaster response [8], and tactical networks [9], where offloading capacity is often limited by the absence of MEC infrastructure or poor UE connectivity. A promising solution is to use unmanned aerial vehicles (UAVs) to bridge this gap. By forwarding computational burdens from resource-constrained mobile devices to infrastructure-mounted servers, the mobility and proximity of UAVs enhance processing efficiency and reduce latency. Moreover, UAVs support dynamic task allocation by adapting to workload variations and geographic changes. This synergy with MEC enhances resource utilization and enables applications in fields like disaster response, surveillance, agriculture, and smart cities [10].

The associate editor coordinating the review of this manuscript and approving it for publication was Cong Pu.

For instance, in post-disaster scenarios, first responders' UEs must handle intensive tasks like real-time imaging and sensor data analysis while conserving energy. Offloading these tasks to nearby servers saves energy and extends device lifetime. Similarly, in smart agriculture, energy-constrained IoT devices can offload tasks like soil analysis and crop monitoring, enabling longer operation in remote areas.

In these situations, the fixed infrastructure is often far from UEs due to sparse deployment. This highlights a second dimension of energy saving: using UAVs as relays[1] between UEs and fixed computing infrastructure.

In the context of UAV-assisted task offloading, some studies [11] have proposed mounting computation servers directly on UAVs. However, UAVs face significant energy constraints that are heavily influenced by payload weight. As a result, carrying powerful servers onboard is currently impractical. In this work, we therefore consider UAVs solely as communication devices that enhance UE connectivity and support task offloading process.

When integrating UAVs as relays into MEC systems, challenges such as path planning, radio resource allocation [12], interference management [13], and optimal positioning [14] have been explored. These studies, mostly from the UAV research community, often prioritize UAV-centric metrics like energy usage, coverage, and offloading delays. In contrast, we argue that UE energy consumption is the most critical metric. While UAVs do consume energy, we treat them as controllable, replaceable relays—unlike UEs in large-scale IoT or first-responder deployments, where device autonomy must be preserved. Accordingly, our focus is on minimizing UE energy consumption in UAV-assisted MEC systems while ensuring fair UE-UAV associations.

UAVs are particularly advantageous for MEC integration. Their unmatched flexibility and mobility enable rapid deployment in areas lacking fixed infrastructure, such as disaster zones or remote regions. This ensures continuous connectivity and computational support, which is vital for emergency response. Furthermore, unlike stationary infrastructure, UAVs can dynamically adjust their positions to optimize network coverage and data transfer rates. This mobility enables on-demand connectivity and access to computing resources, improving the efficiency of task offloading. For example, in high-density settings like festivals, UAVs can reposition to balance the load and reduce congestion.

By serving as relays, UAVs also help lower data transmission latency by reducing the distance data must travel, which accelerates the processing of critical information. This is particularly useful for real-time applications such as augmented reality and time-sensitive IoT tasks. Furthermore, their ability to carry various sensors makes them versatile platforms for surveillance, environmental monitoring, and agricultural assessment.

[1]The term *relay* is used in this paper simply to designate an intermediate node in the network, not in the sense used in the cooperative communications community.

Compared to technologies like satellite communication or fixed terrestrial infrastructure, UAVs are more cost-effective and deployable without major groundwork. Satellites offer wide coverage but suffer from high latency and deployment costs, while fixed infrastructure lacks the responsiveness of UAVs. Therefore, UAV-MEC integration offers a robust and efficient solution for delivering computation and connectivity in diverse environments, improving the performance of modern networked applications.

While several studies have explored UAV-assisted MEC systems to enhance coverage and enable offloading, most focus on optimizing isolated aspects such as UAV path planning or UE-specific offloading. These approaches, however, often overlook the combined impact of UE fairness, energy efficiency, and scalable UAV control under dynamic task distributions. In contrast, our approach integrates a fairness-aware offloading mechanism with energy-efficient UAV coordination using a Multi-Agent Deep Deterministic Policy Gradient (MADDPG) framework. This combination enables UAVs to adapt their trajectories to better serve diverse UE demands and ensure equitable task distribution. For this work, we assume the UEs are static; in our target scenarios, user mobility is often limited, making this a reasonable simplification that allows for a more tractable analysis of the core, learning-based coordination. To the best of our knowledge, this is the first work to jointly optimize UAV trajectories and UE offloading with explicit fairness and energy objectives in a fully decentralized, learning-based setting.

The main contributions of this work are as follows:

- We propose a **joint optimization framework** for MEC task offloading using UAV relays, targeting both UE energy minimization and fairness in task distribution.
- We design a **UAV trajectory control algorithm** based on MADDPG that allows each UAV to manage its path autonomously, enabling flexible and efficient resource allocation.
- We introduce a **low-complexity UE offloading method** that leverages UAV trajectory data to allocate tasks efficiently with minimal overhead.
- Our framework ensures **geographical fairness** by evenly distributing the UE load across UAVs, which prevents bottlenecks and improves system performance.
- We focus on **UE energy minimization** by optimizing UAV movement and balancing offloaded workloads.
- We demonstrate that our approach **outperforms existing methods** in UE energy consumption, offloading efficiency, and fairness, and is **scalable** to varying UAV and UE densities.
- We present a comparative analysis showing our system's superior performance in **resource allocation fairness** and **computational efficiency**, supporting its use in dynamic, real-time environments.

The rest of this article is organized as follows. Section II discusses the motivation and problem scope. Section III reviews related work. Section IV presents the system model.

Section V discusses the problem formulation. Section VI describes the proposed algorithm. Section VII shows evaluation results. Conclusions are given in Section VIII.

II. MOTIVATION AND PROBLEM SCOPE

Task offloading becomes necessary in scenarios where computationally intensive tasks must be executed with limited on-device resources. A compelling example is disaster management, where UAVs can act as relays to offload real-time imaging and sensor data to MEC servers for damage assessment and victim location. In such contexts, UAVs are vital for establishing immediate connectivity, bypassing damaged infrastructure, and enabling coordinated responses. For instance, they can facilitate secure communication in conflict zones, support flood relief with real-time monitoring, and assist in rescue operations by guiding resource allocation to affected areas.

This capability is critical across various emergencies. During earthquakes, UAVs can survey structural damage, locate trapped victims, and deliver medical supplies to inaccessible regions. In wildfires, they are essential for monitoring fire spread and providing real-time thermal imaging to firefighting teams. A recent example is their deployment during the January 2025 wildfires in Southern California, where UAVs supported evacuation efforts and firefighting strategies with real-time data. This adaptability underscores their importance in ensuring a timely and efficient disaster response.

We focus on UAVs as relays because this role maximizes their operational endurance and mobility. Unlike hosting onboard computational servers, which would quickly deplete a UAV's battery, relaying data to ground-based MEC servers enables longer flight times and greater scalability. This distinction is crucial for real-world deployments where energy constraints significantly limit UAV operation time.

This study explicitly addresses task offloading in UAV-assisted MEC systems where UAVs function as communication relays, not as data collectors or processors. Our approach aims to minimize UE energy consumption, improve offloading efficiency, and ensure fairness in UAV-UE associations. To simplify the system model and reduce control complexity, we assume that UEs are static. This assumption reflects realistic deployments, such as smart agriculture, disaster recovery, or industrial monitoring, where user devices are typically stationary. By targeting these goals under this well-defined scope, we enhance the real-world feasibility of our proposed UAV-assisted MEC framework.

III. RELATED WORK

The integration of UAVs into MEC systems—particularly for task offloading—has been widely studied. One line of research considers UAVs as carriers of computing infrastructure. For instance, the authors in [15] optimized the 3D placement of server-equipped UAV base stations (UAV-BSs) to maximize user coverage while minimizing transmit power for given Quality of Service (QoS) requirements. Their method decouples placement into vertical and horizontal components, but its reliance on a simplified path loss model without accounting for user mobility or variable channels limits its real-world applicability.

Similarly, the authors in [16] used a multi-agent reinforcement learning approach for task offloading in UAV- and satellite-assisted IoT systems. They employed a MADDPG framework to jointly optimize trajectories and offloading decisions, aiming to reduce latency and energy consumption. However, the computational demands of the multi-agent environment may challenge its real-time deployment in dynamic networks. In [17], a cooperative multi-agent deep reinforcement learning (MADRL) framework was also proposed to jointly optimize UAV trajectories, task allocation, and resource management to reduce latency and energy use. While effective, this framework may face scalability challenges in large-scale scenarios.

From a similar perspective, [18] proposed a cooperative DRL algorithm for UAV-assisted crowd-sensing systems to

TABLE 1. Notations and definitions (part 1).

Notation	Definition
D	Set of user equipment (UE) devices
N	Set of UAVs
S	Set of base stations (BSs) with MEC servers
H	Fixed height of UAVs
P_{static}	Direct paths between UEs and BSs
$P_{dynamic}$	Routes involving UAVs as relays
$X_{n,t}, Y_{n,t}$	Coordinates of UAV n at time t
x_d, y_d	Coordinates of UE d
X_s, Y_s	Coordinates of BS s
$W_{d,n,t}$	Euclidean Distance between UE d and UAV n at time t
$W_{n,s,t}$	Euclidean Distance between UAV n and BS s at time t
$W_{n,n',t}$	Euclidean Distance between UAV n and UAV n' at time t
$W_{d,s}$	Euclidean Distance between UE d and BS s
$P_d^{transmit}$	Transmit power of UE d
$P_n^{transmit}$	Transmit power of UAV n
$P_s^{transmit}$	Transmit power of BS s
$P_d^{receive}$	Receive power of UE d
E_d	Total energy consumption of UE d
$E_d^{offload}$	Offloading energy consumption of UE d
E_d^{local}	Local processing energy consumption of UE d
$E_j^{transmit}$	Transmit energy per offloaded task
$E_j^{receive}$	Reception energy consumption for result of offloaded task
E_d^{tx}	Energy consumption during transmission for UE d
E_d^{rx}	Energy consumption during reception for UE d
E_j^d	Energy consumption of UE d for locally executing task j
E_{min}	Minimum threshold for UE d energy consumption
t_j^{route}	Routing time for task j
t_j^{return}	Return time for task j
$t_j^{route(dn)}$	Transmission time for task j from UE d to UAV n
$t_j^{route(ds)}$	Transmission time for task j from UE d to BS s
$c_{n,t}$	Relative UE load of UAV n
$STATE$	State space
$State_{t+1}$	New state

TABLE 2. Notations and definitions (part 2).

Notation	Definition
$t_j^{route(ns)}$	Transmission time from UAV n to BS s
$t_j^{return(sd)}$	Return time for task j from BS s to UE d
$t_j^{return(nd)}$	Return time for task j from UAV n to UE d
$t_j^{return(sn)}$	Return time for task j from BS s to UAV n
t_j^{E2E}	End-to-end delay for task j
B	Channel Bandwidth
β_0^d	Channel power gain at a reference distance of 1 meter from UE d
β_0^n	Channel power gain at a reference distance of 1 meter from UAV n
β_0^s	Channel power gain at a reference distance of 1 meter from BS s
N_0	Power spectrum density
r_{dn}	Data rate between UE d and UAV n
r_{ds}	Data rate between UE d and BS s
r_{sd}	Data rate between BS s and UE d
r_{ns}	Data rate between UAV n and BS s
r_{nd}	Data rate between UAV n and UE d
R_j	Task result size
I_j	Size of task j
C_d	Computational load of UE d
$\alpha_{n,t}$	Direction of UAV n at time t
$d_{n,t}$	Distance moved by UAV n at time t
d^{max}	Boundaries of UAV desired location
$z_{d,t}$	Binary decision variable for task offloading by UE d at time t
W^{max}	Maximum coverage range of UAVs
W^u	Minimum safe distance between UAVs
$fair_t^e$	Fairness index for geographical distribution
$fair_t^d$	Fairness index for UE load distribution
$fair_t^{energy}$	Fairness index for energy consumption
$rew_{n,t}$	Reward function for UAV n at time t
$fair_t^{rate}$	Fairness index for data rate
T_j^{Exp}	Task expiry time
T_j^{Arr}	Task arrival time
A	Action space
a_t	Action policy
$Racum_t$	Cumulative reward
γ	Discount factor
T	Number of time slots
$obs_{n,t}$	Observation
$Q^n(state_t, act_t)$	Critic network
Pen_n	UAV crash penalty
δ_n	TD error
$P_{n,k}$	Probability of K-th sampled transition
K	Mini batch
ε	positive constant
$L(\theta^{Q^n})$	UAV loss function policy gradient

maximize system utility by jointly optimizing data sensing and task offloading. The framework's robustness, however, may be offset by scalability issues and high computational overhead. The work in [19] also introduced a reinforcement learning strategy using Proximal Policy Optimization (PPO) to optimize offloading decisions by considering UAV and UE mobility. Despite its fast convergence, the method's computational complexity could limit its scalability in large-scale deployments.

However, due to their inherent energy limitations, we argue that equipping UAVs with powerful computation servers is impractical. Our approach is thus distinguished from these previous works by considering UAVs solely as communication relays, which offers a more practical and energy-efficient solution.

In this context, [20] integrated game theory with multi-agent reinforcement learning (MARL) to improve UAV-assisted offloading. Their method combines potential games for service assignment with a MADDPG-based trajectory optimization to balance energy use and reduce delays. This hybrid approach, however, increases computational complexity, potentially limiting its real-time applicability.

Trajectory planning is a key challenge, and various studies have addressed it. The authors in [21] proposed a matching-based DRL approach that integrates matching theory with MADDPG to enhance secure data transmission. In [22], a strategy combining game theory and Deep Deterministic Policy Gradient (DDPG) was presented to reduce offloading delays and improve energy efficiency. A framework for fixed-wing UAVs was proposed in [23] that minimizes completion time using successive convex approximation. While these methods are effective, their added complexity or reliance on precise environmental modeling can hinder real-time deployment and adaptability.

Several works have focused on communication challenges. The studies in [24], [25], and [26] explored enhancing communication efficiency by optimizing for data rates through NOMA, reducing interference via altitude and power control, and jointly optimizing scheduling, power, and trajectory, respectively. Furthermore, the works in [27] and [28] used DRL to dynamically control trajectories for task offloading, offering practical strategies to improve energy efficiency and fairness.

Focusing specifically on fairness, [29] introduced energy-aware trajectory optimization for UAV-mounted reconfigurable intelligent surfaces (RIS), while [30] examined the trade-off between energy efficiency and service fairness in UAV swarm orchestration. These studies form a foundation for the expanded fairness evaluation in our framework.

Other recent works have explored different optimization techniques. The authors in [31] used Ant Colony Optimization (ACO) to balance task completion time and operational cost, while [32] used Lyapunov optimization for joint trajectory, caching, and migration management. To handle complex, dependent tasks, the work in [33] used a twin-delayed deep deterministic (TD3)-based algorithm. To address dense networks, [34] used Adaptive Particle Swarm Optimization (APSO) in a multi-layer MEC architecture, and [35] combined a matching algorithm with K-means for urban IoT environments.

Several studies have also explicitly addressed user mobility. In [36], a multi-agent soft actor-critic (MSAC) algorithm was proposed for dynamic trajectory adaptation, though it assumes static UEs to reduce control overhead. The

work in [37] proposed an algorithm to account for user mobility using a decomposed Mixed-integer non-linear programming (MINLP) strategy. In contrast, our approach does not consider trajectory prediction as it focuses on static UE applications where coordination is prioritized. Similarly, [38] introduced a spatial-temporal Double Deep Q-Network (DDQN) framework for dynamic environments like intelligent transportation, while our method favors a more lightweight, decentralized inference tailored to stable settings.

Considering the broader MEC field, works such as [39], [40], [41], and [42] have employed DRL techniques like DDQN to enhance performance in vehicular and IoT environments. However, the solutions in these specific works rely on static infrastructure, limiting their adaptability. Furthermore, they overlook the explicit consideration of fairness in resource distribution. To address these particular limitations, our framework introduces UAVs as dynamic mobile aerial relays. Coupled with a sophisticated MADDPG model, our solution not only optimizes traditional metrics but also explicitly prioritizes fairness across multiple dimensions.

In a related effort to handle dynamic environments, the authors in [43] presented a framework to maximize the computation rate in UAV-assisted, wireless-powered MEC systems with mobile users. While their swarm intelligence-based framework is designed for user mobility, our work differs in its primary objectives. We focus on minimizing UE energy consumption and maximizing fairness rather than on computation rate alone. Moreover, our approach utilizes a MADDPG model for decentralized control, which contrasts with their swarm-based method. From a broader optimization perspective, the work in [44] addresses premature convergence in Particle Swarm Optimization (PSO) by introducing a double hierarchical swarm structure. While this offers a powerful general-purpose optimizer, our work is distinct in its application-specific approach, formulating the UAV-MEC challenge as a multi-agent problem to be solved with DRL rather than a swarm-based metaheuristic.

Lastly, [45] proposed a Prioritized Experience Replay (PER)-DDPG algorithm for secure offloading, jointly optimizing UAV mobility and jamming power. While resilient in adversarial scenarios, their security-centric, mobility-aware model differs from our static UE assumption, which enables a leaner control structure for our mission-oriented applications.

Many existing works emphasize energy challenges at the UAV or system level. However, UE energy consumption is a critical and often overlooked issue. In many deployments, recharging a UAV is feasible, while replacing a UE battery is not [46], [47], [48], [49]. Recognizing this gap, our study focuses exclusively on minimizing UE energy consumption and ensuring fairness in a multi-UAV-assisted MEC system.

These prior works collectively highlight diverse approaches to task offloading, UAV coordination, and energy-aware control in MEC systems. A consolidated overview of related studies and their corresponding techniques is presented in Table 3, allowing for clear comparison across problem domains and methodological trends.

In our work, we propose a joint optimization framework that aims to minimize UE energy consumption, achieve geographical fairness among UEs, and ensure balanced UE load distribution across UAVs. Given the problem's complexity, we adopt a multi-agent deep reinforcement learning approach for UAV trajectory control, specifically using the well-established MADDPG method to autonomously manage each UAV's flight path. Additionally, we introduce a simplified strategy to enhance UE offloading decisions based on UAV trajectories.

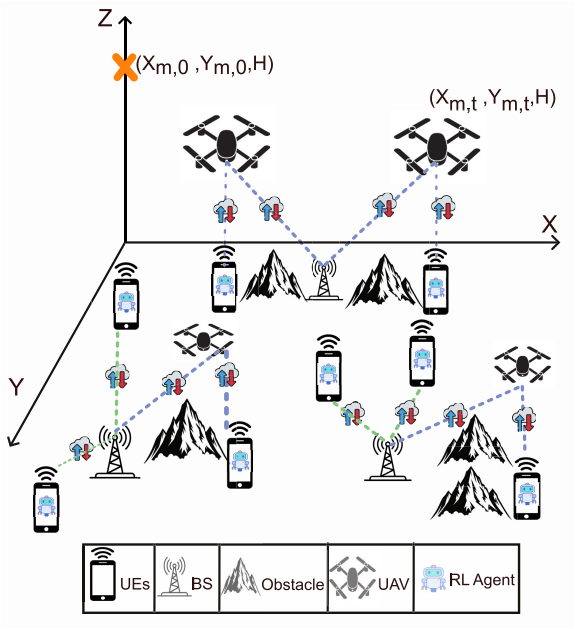

FIGURE 1. System model of the UAV-assisted MEC system.

IV. SYSTEM MODEL

This section outlines the system architecture and defines the primary variables and constraints in the UAV-assisted MEC environment. The key notations are summarized in Table 1 and Table 2.

We consider a set of UE devices located in a specific geographical area. Let d represent an individual UE and D the complete set of UEs, as shown in Fig. 1. Similarly, a set of Base Stations (BSs), each equipped with a MEC server, is defined. Let s denote a server and S be the set of all servers.

We consider a set of UAVs (denoted by N, with each UAV represented by $n \in N$) that supports offloading from UEs to servers [50]. These UAVs operate at a fixed height H and assist ground UEs by relaying tasks. The fixed-height assumption simplifies the optimization by focusing on horizontal trajectories and offloading decisions, avoiding the added complexity of altitude control. In practice, this assumption is justified by regulatory constraints and UAV limitations like battery capacity and flight stability. Although

TABLE 3. Consolidated overview of related works in UAV-assisted MEC systems.

Ref.	Problem										Techniques				
	UAVs as Computing Carriers	UAVs as Communication Relays	UAV Trajectory & Mobility	Fairness & Load Balance	Communication Link Design	Dynamic UE Offloading	Architectures & Management	Security & Static Infrastructure	Energy-Efficient UAV Use	Security & Jamming	Reinforcement Learning	Game Theory & Matching	Convex/ACO/Graph Optimization	Lyapunov & DAG Aware	PSO & Heuristics
[15]	✓					✓							✓		
[16]	✓		✓			✓					✓				
[17]	✓		✓			✓					✓				
[18]	✓		✓			✓					✓				
[19]	✓		✓			✓					✓				
[20]		✓	✓			✓					✓	✓			
[21]		✓	✓							✓	✓	✓			
[22]			✓	✓							✓	✓			
[27]			✓			✓					✓				
[28]			✓			✓					✓				
[31]			✓			✓							✓		
[32]			✓			✓	✓						✓	✓	
[23]			✓										✓		
[29]				✓											
[30]				✓											
[24]					✓										
[25]					✓										
[26]					✓										
[36]			✓			✓					✓				
[37]						✓									
[38]						✓					✓				
[33]						✓					✓			✓	
[34]						✓	✓								✓
[35]						✓	✓								✓
[45]								✓		✓	✓				
[39]								✓			✓				
[40]								✓			✓				
[41]								✓			✓				
[42]								✓			✓				
[46]									✓				✓		
[47]									✓						
[48]									✓						
[49]									✓						

a variable altitude could offer benefits, it requires complex 3D optimization beyond this study's scope. The fixed height H thus provides a practical trade-off between performance and computational feasibility.

Without loss of generality, we assume each UE has a set of computational tasks to offload. These tasks can be sent directly to nearby MEC servers or, if unavailable, to more distant servers via UAV relays [51]. Let j denote an individual task and J the set of all tasks, where the subset J_d represents tasks assigned to UE d. Tasks arrive independently at UEs, each with a unique arrival time and deadline.

We define a set of communication paths between D and S as P. Each path $p \in P$ enables a device d to offload task j to a server s. The path set P includes two types: P_{static} for direct UE–BS links and $P_{dynamic}$ for routes that include UAVs as relays [52]. While P_{static} paths are always available, $P_{dynamic}$ routes are generated dynamically based on UAV–UE communication.

We adopt a slotted time system, where T is the set of time slots. In each slot t, every UAV n moves in a direction given by angle $\alpha_{n,t} \in [0, 2\pi)$ and covers a distance $d_n^t \in [0, d^{max}]$, remaining within the operational area. The initial coordinates of UAV n are $[X_{n,0}, Y_{n,0}, H]$. Its position at time t is $[X_{n,t}, Y_{n,t}, H]$, where $X_{n,t} = X_{n,0} + \sum_{t'=0}^{t} d_n^{t'} \cdot \cos(\alpha_{n,t'})$ and $Y_{n,t} = Y_{n,0} + \sum_{t'=0}^{t} d_n^{t'} \cdot \sin(\alpha_{n,t'})$.

Our model assumes UAVs function solely as relays, forwarding tasks to MEC servers without performing onboard execution. This design respects UAVs' energy and mobility constraints, allowing for prolonged and effective operations. Prior studies [53], [54] have shown that using UAVs as communication relays enhances both energy efficiency and system scalability. By focusing on this role, our framework emphasizes minimizing UE energy consumption and promoting fairness. This approach is particularly effective in scenarios like disaster recovery [46], where UAVs provide essential connectivity, and smart agriculture [48], where they support energy-constrained IoT devices.

A. COMMUNICATION MODEL

The communication distances between system entities are defined as follows:

- The distance between a static UE d and UAV n at time slot t is given by Eq. (1):

$$W_{d,n,t} = \sqrt{(X_{n,t} - x_d)^2 + (Y_{n,t} - y_d)^2 + H^2}, \quad \forall d \in D,\ n \in N,\ t \in T. \tag{1}$$

  This expression computes the Euclidean distance between a UE and a UAV, which is critical for determining communication feasibility and power requirements.
- The distance between UAV n and BS s at time slot t is defined in Eq. (2):

$$W_{n,s,t} = \sqrt{(X_{n,t} - X_s)^2 + (Y_{n,t} - Y_s)^2 + H^2}, \quad \forall n \in N,\ s \in S,\ t \in T. \tag{2}$$

  This metric is used to assess the energy and delay for relaying tasks from UAVs to MEC servers, where $[X_s, Y_s]$ are the BS coordinates.
- The horizontal distance between UAV n and UAV n' at time slot t is evaluated in Eq. (3):

$$W_{n,n',t} = \sqrt{(X_{n,t} - X_{n',t})^2 + (Y_{n,t} - Y_{n',t})^2}, \quad \forall n, n' \in N, t \in T. \tag{3}$$

  This equation measures the planar separation between two UAVs, essential for maintaining a minimum safe distance W^u for collision avoidance.
- The UE-to-BS distance is quantified by Eq. (4):

$$W_{d,s} = \sqrt{(\|X_d - X_s\|)^2 + (\|Y_d - Y_s\|)^2}, \quad \forall d \in D,\ s \in S. \tag{4}$$

  This formulation determines the direct distance between a UE and a BS, which is crucial for evaluating offloading paths that bypass UAV relays.

B. DELAY CONSIDERATIONS

To evaluate the communication delay for task offloading, we compute the total communication time for a task j under different routing scenarios.

- When a task is routed through a UAV to a BS, the communication time is given by Eq. (5):

$$t_j^{route} = t_j^{route(dn)} + t_j^{route(ns)}, \tag{5}$$

  where $t_j^{route(dn)} = I_j / r_{dn}$ is the UE-to-UAV transmission time, and $t_j^{route(ns)} = I_j / r_{ns}$ is the UAV-to-BS transmission time. The data rates r_{dn} and r_{ns} are defined as:

$$r_{dn} = B \cdot \log_2\left(1 + \frac{P_d^{transmit} \cdot \beta_0^d}{B \cdot N_0 \cdot W_{d,n,t}^2}\right),$$
$$r_{ns} = B \cdot \log_2\left(1 + \frac{P_n^{transmit} \cdot \beta_0^n}{B \cdot N_0 \cdot W_{n,s,t}^2}\right).$$

  Here, I_j is the input task size, B is the channel bandwidth, N_0 is the noise power spectral density, $P_d^{transmit}$ and $P_n^{transmit}$ are the transmit powers of the UE and UAV, and β_0^d and β_0^n are the respective channel power gains at a reference distance of 1 meter.
- When a task is offloaded directly to a BS, the time is calculated using Eq. (6):

$$t_j^{route} = t_j^{route(ds)}, \tag{6}$$

  where $t_j^{route(ds)} = I_j / r_{ds}$, and the data rate r_{ds} is:

$$r_{ds} = B \cdot \log_2\left(1 + \frac{P_d^{transmit} \cdot \beta_0^d}{B \cdot N_0 \cdot W_{d,s}^2}\right).$$

  This models the delay over a single wireless hop, capturing the impact of bandwidth, power, noise, and distance.
- If a task result is returned via a UAV, the return time is given by Eq. (7):

$$t_j^{return} = t_j^{return(sn)} + t_j^{return(nd)}, \tag{7}$$

  where the BS-to-UAV time is $t_j^{return(sn)} = R_j / r_{sn}$ and the UAV-to-UE time is $t_j^{return(nd)} = R_j / r_{nd}$. The result size is R_j, and the data rates are:

$$r_{sn} = B \cdot \log_2\left(1 + \frac{P_s^{transmit} \cdot \beta_0^s}{B \cdot N_0 \cdot W_{n,s,t}^2}\right),$$
$$r_{nd} = B \cdot \log_2\left(1 + \frac{P_n^{transmit} \cdot \beta_0^n}{B \cdot N_0 \cdot W_{d,n,t}^2}\right).$$

- If a result is returned directly from a BS, the time is calculated using Eq. (8):

$$t_j^{return} = t_j^{return(sd)}, \tag{8}$$

  where $t_j^{return(sd)} = R_j / r_{sd}$, and the data rate r_{sd} is:

$$r_{sd} = B \cdot \log_2\left(1 + \frac{P_s^{transmit} \cdot \beta_0^s}{B \cdot N_0 \cdot W_{d,s}^2}\right).$$

- The total end-to-end (E2E) delay for task j is the sum of its routing and return times:

$$T_j^{E2E} = t_j^{route} + t_j^{return}. \tag{9}$$

This final metric is crucial for evaluating task execution performance under realistic wireless conditions.

C. ENERGY CONSIDERATIONS

We note that a task is either offloaded or computed locally and will thus be accounted for in either the offloading energy or the local computation energy. Hence, The total energy consumption of a UE d, E_d, consists of offloading energy and local computation energy:

$$E_d = E_d^{offload} + E_d^{local}, \tag{10}$$

where the components are defined as follows: label=·

- $E_d^{offload} = E_d^{tx} + E_d^{rx}$: The total energy for offloaded tasks, comprising transmission (E_d^{tx}) and reception (E_d^{rx}) energy. label=○
  - $E_d^{tx} = \sum_{s\in S}\sum_{j\in J_d}\sum_{t\in T} E_j^{transmit}$
  - $E_d^{rx} = \sum_{s\in S}\sum_{j\in J_d}\sum_{t\in T} E_j^{receive}$
  - $E_j^{transmit} = P_d^{transmit} \cdot t_j^{route}$
  - $E_j^{receive} = P_d^{receive} \cdot t_j^{return}$
- $E_d^{local} = \sum_{j\in J_d} E_j^d$: The cumulative energy for executing tasks locally. For each task j, the local energy E_j^d depends on the number of CPU cycles k_d^j, device frequency f_d, and effective switched capacitance C_d.

D. DECISION VARIABLES

- The binary decision variable for offloading at time t for UE d is:

$$z_{d,t} = \begin{cases} 1, & \text{if task is offloaded} \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

- The relative UE load on UAV n at time t is the proportion of UEs that choose to offload:

$$c_{n,t} = \frac{\sum_{d\in D} z_{d,t}}{|D|}, \quad \forall t \in T, \quad n \in N. \tag{12}$$

This metric measures the instantaneous demand on each UAV and is key for load balancing.

E. FAIRNESS CALCULATION MODEL

- **Fairness among UAVs:** To prevent task congestion and ensure balanced workloads, we measure fairness among UAVs using an index inspired by Jain's fairness metric [55]:

$$fair_t^d = \frac{\left(\sum_{n\in N}\sum_{t'=1}^{t} c_{n,t'}\right)^2}{|N| \cdot \sum_{n\in N}\left(\sum_{t'=1}^{t} c_{n,t'}\right)^2}. \tag{13}$$

A value of $fair_t^d$ close to 1 indicates an equitable distribution of the UE load across UAVs.

- **Fairness among UEs:** To ensure UEs are served equitably over time [56], we use a geographical fairness index:

$$fair_t^e = \frac{\left(\sum_{d\in D_{active}}\sum_{t'=1}^{t} z_d^{t'}\right)^2}{|D_{active}| \cdot \sum_{d\in D_{active}}\left(\sum_{t'=1}^{t} z_d^{t'}\right)^2}. \tag{14}$$

A value of $fair_t^e$ near 1 signifies that all active UEs (D_{active}) have been served for a similar number of time slots.

- **UEs Energy Consumption Fairness:** To evaluate fairness in energy usage across UEs, we define the Energy Consumption Fairness Index:

$$fair_t^{energy} = \frac{\left(\sum_{d\in D}\sum_{t'=1}^{t} E_d^{t'}\right)^2}{|D| \cdot \sum_{d\in D}\left(\sum_{t'=1}^{t} E_d^{t'}\right)^2}. \tag{15}$$

A value close to 1 indicates that energy consumption is evenly distributed among all UEs.

- **Data Rate Fairness:** Similarly, to evaluate fairness in data rate allocation among UEs, we define the Data Rate Fairness Index:

$$fair_t^{rate} = \frac{\left(\sum_{d\in D}\sum_{t'=1}^{t} r_{nd}^{t'}\right)^2}{|D| \cdot \sum_{d\in D}\left(\sum_{t'=1}^{t} r_{nd}^{t'}\right)^2}. \tag{16}$$

A value close to 1 indicates that data rates are being allocated equitably among UEs.

V. PROBLEM FORMULATION

With the above fairness metrics in place, we define the optimization problem as:

$$\begin{aligned} P1: \quad & \max_{z_d^t} \quad \sum_{t=1}^{T} \frac{fair_t^e \cdot fair_t^d \cdot fair_t^{energy} \cdot fair_t^{rate}}{\sum_{d\in D} z_d^t \cdot E_d} \\ s.t. \quad & \mathscr{C}1: \quad W_{n,n',t} \geq W^u, \quad \forall n, n' \in N,\ n \neq n',\ \forall t \in T \\ & \mathscr{C}2: \quad T_j^{E2E} \leq T_j^{Exp} - T_j^{Arr}, \quad \forall j \in J \\ & \mathscr{C}3: \quad E_d^t \geq E_{min}, \quad \forall d \in D, \quad \forall t \in T \\ & \mathscr{C}4: \quad W_{d,n,t} \leq W^{max}, \quad \forall d \in D, \quad n \in N, \quad t \in T \end{aligned} \tag{17}$$

This optimization problem in Eq. (17) aims to maximize the fairness-aware objective function for UAV-based task offloading. The numerator aggregates four fairness components—geographical distribution $fair_t^e$, UE load $fair_t^d$, energy usage $fair_t^{energy}$, and data rate $fair_t^{rate}$—while the denominator penalizes high energy consumption across UEs.

The constraints in the optimization problem are designed to ensure system feasibility and fairness. Constraint $\mathscr{C}1$ enforces collision avoidance by maintaining a minimum distance W^u between any two UAVs at all times. Constraint $\mathscr{C}2$ ensures that the total end-to-end delay T_j^{E2E}, which includes both the task routing time and result return time,

must not exceed the available time budget $T_j^{Exp} - T_j^{Arr}$. Where T_j^{Exp} is the task expiry time and T_j^{Arr} is the task arrival time. Constraint $\mathscr{C}3$ ensures UE battery preservation by maintaining the energy level above the threshold E_{min}, thus enabling basic operational functionality. Lastly, constraint $\mathscr{C}4$ enforces UAV coverage limits, requiring that a UE must be within the communication range $W^{\max}$ of a UAV to allow successful task offloading.

Our primary goal is to minimize the energy consumption of the UEs while also achieving optimal fairness in multiple dimensions, including the distribution of UE load across UAVs, fairness in energy consumption, and fairness in data rate allocation, ensuring efficient and equitable resource utilization in dynamic UAV-assisted MEC systems.

VI. PROPOSED ALGORITHMS

Our problem is divided into two sub-problems: UAV positioning and UE-UAV task offloading. This section details the methodology for each sub-problem and presents the complete algorithm that integrates both solutions.

A. UAV POSITIONING SUB-PROBLEM

1) UAV TRAJECTORIES AND OPTIMIZATION LOGIC

To achieve efficient and fair task offloading, we optimize UAV trajectories using a MADDPG framework. In this approach, each UAV operates as an autonomous agent, learning from its environment to determine an optimal flight path. The key optimization objectives are to:

- Minimize the task offloading delay for UEs.
- Ensure fair distribution of UAV services across all UEs.
- Reduce UE energy consumption by minimizing unnecessary UAV movements.

At each time step, a UAV evaluates its current position, the location of nearby UEs, and the task demand using a reward function (Eq. 17) designed to prioritize fairness and energy efficiency. The trajectories are updated dynamically, allowing the UAVs to adapt to real-time changes in UE distribution and workload demands.

2) PROPOSED SOLUTION

Our objective is to solve the UAV positioning problem to fairly serve all UEs and provide immediate assistance for their offloaded tasks. A widely accepted approach for such problems is the Markov decision process (MDP) [57], which is based on a state space $STATE$ (with states $state_t = state_1, \ldots, state_T$) and an action space A (with actions $a_t = a_1, \ldots, a_T$). In the MDP framework, an agent engages with the environment in discrete time steps (TSs) by assessing the current state and selecting an action.

Taking an action yields a reward, rew_t, and leads to a new state, $state_{t+1}$. A policy, represented by the function $a_t = \pi(state_t)$ [58], maps the current state to a valid action. The agent's primary goal is to find an optimal policy that maximizes the cumulative reward, $Racum_t = \sum_{i=t}^{T} \gamma^{i-1} \cdot rew_i$, where $\gamma \in (0, 1)$ is a discount factor.

Building on this, we propose a multi-UAV reinforcement learning-based trajectory control algorithm, which uses an observable Markov game framework solved with the MADDPG algorithm. MADDPG was chosen because it is model-free and enables UAVs to make adaptive, real-time decisions in a decentralized manner while optimizing global objectives through centralized training. This feature allows our method to handle the dynamic, multi-agent nature of the problem, where UAVs must cooperate to minimize energy consumption, ensure fairness, and reduce latency while adhering to system constraints like collision avoidance. By leveraging MADDPG, our framework achieves the scalability and robustness necessary for large-scale, real-world UAV-assisted MEC systems, avoiding the computational burden of classical iterative solvers.

Each UAV is controlled by a dedicated agent. The system involves $|M|$ agents interacting with an environment characterized by a set of states, $STATE \triangleq \{state_t, t \in T\}$, and actions, $A \triangleq \{act_t, t \in T\}$. Each state $state_t$ includes the private observation $obs_{n,t}$ for each agent. Within each TS, every agent n obtains its observation $obs_{n,t}$, executes an action $act_{n,t}$, and receives a reward $rew_{n,t}$, after which the environment transitions to a new state. Each agent possesses an actor network ($act_{n,t} = \pi^n(obs_{n,t})$), a critic network ($Q^n(state_t, act_t)$), and corresponding target networks ($act_{n,t+1} = \pi^{n'}(obs_{n,t+1})$ and $Q^{n'}(state_{t+1}, act_{t+1})$).

Our framework combines decentralized execution with centralized training. During training, every agent shares its observation $obs_{n,t}$ and action $act_{n,t}$ with the environment. The environment, in turn, provides each agent with the global state $state_t$, which is composed of all agents' observations and actions. This enables the critic network of each agent to be trained using comprehensive system information. Subsequently, during the testing phase, each agent acts based only on its own private observation, allowing for decentralized execution.

We define the observation, action, and reward for each agent at TS t as follows:

1) **Observation** $obs_{n,t}$: The observation for agent m controlling UAV n incorporates the UAV's coordinates $[X_{n,t}, Y_{n,t}]$, the set of relative distances to other UAVs ($W_{n,n',t}, \forall n' \in N, n' \neq n$) to prevent collisions, and the accumulated service times for UEs and the current UE-load on UAV n to enhance exploration.
2) **Action** $act_{n,t}$: The action is defined by the UAV's flying direction and distance, represented as $act_{n,t} = \{\alpha_{n,t}, dist_{n,t}\}$ for the n-th UAV in the t-th TS.
3) **Reward Function** $rew_{n,t}$: The reward function is defined in Eq. (18):

$$rew_{n,t} = \frac{fair_t^e \cdot fair_t^d \cdot fair_t^{energy} \cdot fair_t^{rate}}{\frac{1}{D} \cdot \sum_{d=1}^{|D|} z_d^t \cdot E_d} - pen_n \tag{18}$$

This function guides UAV trajectory decisions by combining fairness indices with normalized UE energy

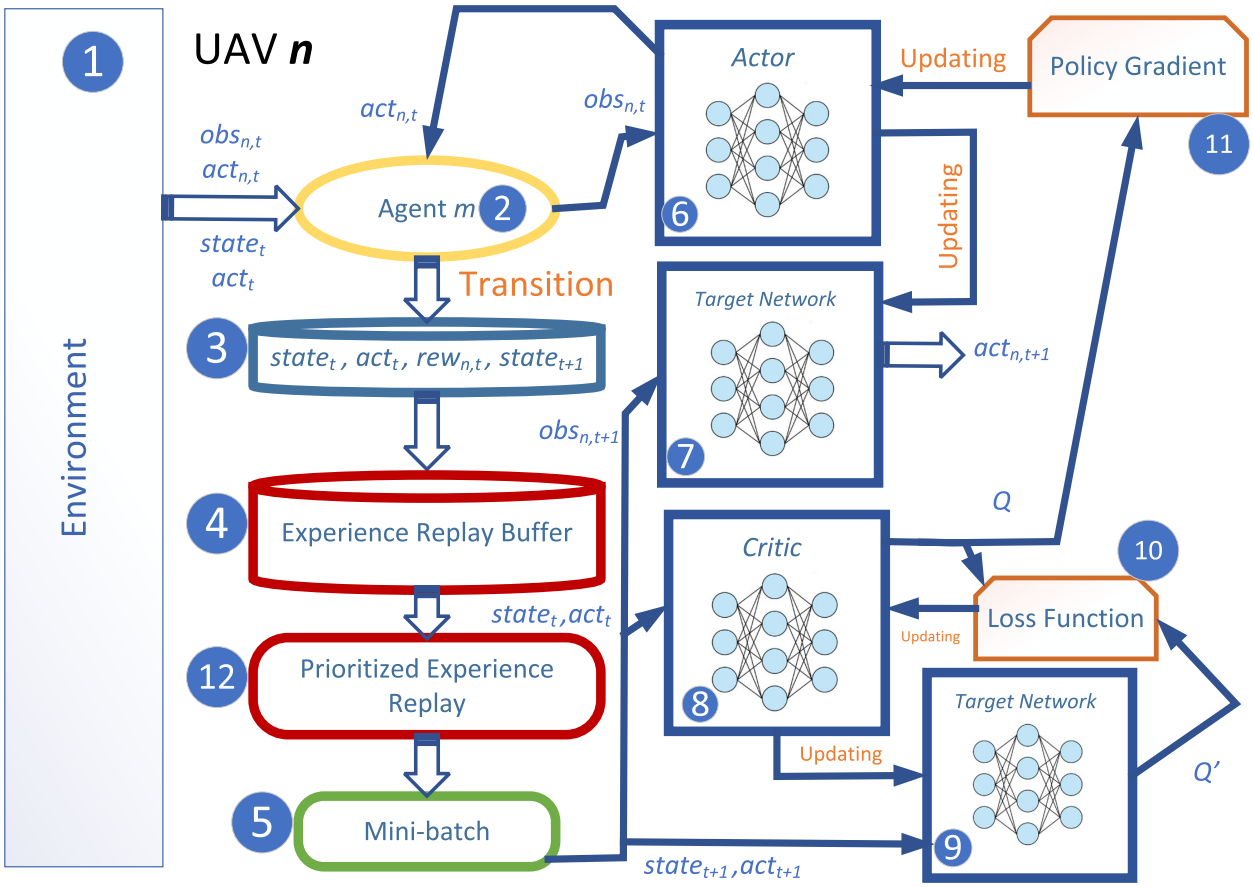

FIGURE 2. UAV *n* and its agent *m* structure.

consumption. The penalty term pen_n discourages collisions and boundary violations.

The structure of an agent is depicted in Fig. 2. Each agent ② selects an action from its actor network ⑥. The transition experience $e_{n,t} \triangleq \{s_t, a_t, r_{n,t}, s_{t+1}\}$ ③ is stored in an experience replay buffer ④. To improve learning, we do not sample transitions randomly. Instead, following the proposal in [59], we prioritize transitions with a high temporal difference (TD) error, which indicates a valuable learning signal. The TD error, δ_n, is computed using the current and target critic networks (θ^{Q^n} and $\theta^{Q^{n'}}$) as shown in Eq. (19):

$$\delta_n = rew_{n,t} + \gamma \cdot Q^{n'}\left(state_{t+1}, act_{t+1} \mid \theta^{Q^{n'}}\right) - Q^n\left(state_t, act_t \mid \theta^{Q^n}\right), \quad \forall n \in N, \forall t \in T. \tag{19}$$

This error quantifies the gap between the predicted and target Q-values and is used to accelerate convergence via prioritized experience replay ⑫. The probability of sampling the k-th transition is given by Eq. (20):

$$P_{n,k} = \frac{\left(|\delta_{n,k}| + \epsilon\right)^{\beta}}{\sum_{k'=1}^{K}\left(|\delta_{n,k'}| + \epsilon\right)^{\beta}}, \quad \forall n \in N \tag{20}$$

where K is the mini-batch size, and ϵ and β are constants that control the prioritization.

The loss function ⑩ for UAV n minimizes the squared TD-error, weighted by inverse sampling probabilities, to update the critic network, as defined in Eq. (21):

$$L(\theta^{Q^n}) = \mathbb{E}\left[\frac{1}{(K \cdot P_{n,k})^{\mu}}(\delta_n)^2\right] \tag{21}$$

where μ is a constant. The critic network ⑧ is updated using the policy gradient ⑪ from [60] to train the actor network ⑥, as shown in Eq. (22):

$$\nabla_{\theta^{\pi^n}} J = \mathbb{E}\left[\nabla_{\theta^{\pi^n}} \pi^n\left(obs_{n,t} | \theta^{\pi^n}\right) \nabla_{act_{n,t}} Q^n(state_t, a_t | \theta^{Q^n})\right], \quad \forall n \in N, \quad \forall t \in T. \tag{22}$$

This gradient update guides UAVs toward actions that maximize the expected critic value, thereby optimizing for fairness and energy efficiency.

B. UE-UAV TASK OFFLOADING SUB-PROBLEM

We present a low-complexity method for UEs to optimize their offloading decisions using a matching algorithm that considers UAV positions. After the UAVs have moved, each UE chooses the UAV that results in the least energy consumption for offloading. If no suitable UAV is available, the UE can use the heuristic from [52] to process the task locally or offload it to the closest ground-based BS.

C. PROPOSED ALGORITHM

Our proposed algorithm is named Multi-UAV Reinforcement Learning-based Trajectory Control (MUT). Its procedure is detailed in Algorithm 1. During the initialization phase (Lines 1-5), each UAV sets up its actor, critic, and target networks. The main training loop (Lines 6-28) proceeds as follows: each UAV gathers observations, selects an action from its actor network (with added noise for exploration), and executes it. If a collision or boundary violation occurs, the UAV remains in its current position and receives a penalty. Based on the new UAV positions, each UE makes its optimal offloading decision. The resulting reward and new observation are obtained, and the transition experience is stored in the replay buffer.

In the learning procedure (Lines 28-34), a mini-batch of K transitions is sampled from the buffer using the prioritized replay scheme. The actor and critic networks are then updated using the policy gradient (Eq. 22) and the loss function (Eq. 21), respectively. Finally, the target networks are updated, and the priorities of the sampled transitions are revised.

Algorithm 1 Proposed Algorithm MUT

1 **for** *each UAV n candidate in N* **do**
2 **Initialize:** actor network $\pi^n(.)$, critic network $Q^n(.)$ with parameters θ^{π^n} and θ^{Q^n};
3 **Initialize:** target networks $\pi^{n'}(.)$ and $Q^{n'}(.)$ with parameters $\theta^{\pi^{n'}} = \theta^{\pi^n}$ and $\theta^{Q^{n'}} = \theta^{Q^n}$;
4 **Initialize:** experience replay buffer B_n;
5 **end**
6 **for** *Episode = 1, 2, …, $e^{\max}$* **do**
7 **for** *each UAV n in N* **do**
8 **Initialize:** Observation $obs_{n,t}$;
9 **end**
10 **for** *each TS t in T* **do**
11 **Obtain:** State $state_t$;
12 **for** *each UAV n in N* **do**
13 **Obtain:** Action $act_{n,t} = \pi^n\left(obs_{n,t}|\theta^{\pi^n}\right) + \epsilon$; **Execute:** $act_{n,t}$.
14 **end**
15 **Obtain:** Action act_t;
16 **for** *each UE d in D* **do**
17 **Obtain:** Available offloading decision that consumes the least energy; **Calculate:** $E_{d,n,t}$;
18 **end**
19 **for** *each UAV n in N* **do**
20 **Obtain:** $rew_{n,t}$ according to (17);
21 **Obtain:** $obs_{n,t+1}$;
22 **end**
23 **Obtain:** $state_{t+1}$;
24 **for** *each UAV n in N* **do**
25 **Store:** Transition $\{s_t, act_t, r_{n,t}, s_{t+1}\}$ into experience replay buffer B_n with priority $|\delta_n| + \epsilon$;
26 **if** *learning process starts* **then**
27 Sample a mini-batch of K transitions from B_n with probability $P_{n,k}$;
28 Update actor network according to (21);
29 Update critic network according to (22);
30 Update target networks with updating rate τ;
31 Update priorities of K transitions;
32 **end**
33 **end**
34 **end**
35 **end**

VII. NUMERICAL RESULTS AND DISCUSSIONS

A. SIMULATION SETUP

The reinforcement learning model was trained and evaluated using Python 3.7 and MATLAB 2022b on a laptop with an Intel Core i5-11400H processor (2.7 GHz), 8 GB RAM, and integrated Intel graphics. No discrete GPU was used. Training the agents over 1000 episodes took approximately 2 hours, and once trained, the MUT algorithm runs in real-time, with each decision cycle completing in under 100 milliseconds. This performance on mid-range hardware confirms the computational efficiency and practical deployability of our solution.

The training environment was generated using a customized simulator that emulates dynamic UAV positions and task arrival rates. This approach allows for repeatable, controlled evaluations that reflect realistic MEC scenarios without requiring proprietary or real-world datasets.

The actor and critic networks were trained with learning rates of 3×10^{-5} and 10^{-4}, respectively, using the Adam optimizer [61]. Our simulation environment consists of 50 UEs and 10 BSs randomly distributed in a target square-shaped region. To analyze the impact of resource availability, we evaluate two primary scenarios with 3 and 4 UAVs.

Each training episode consists of 20 time slots. In the four-UAV scenario, the initial coordinates were set to [10, 10], [90, 90], [10, 90], and [90, 10] to ensure wide initial coverage. Tasks are generated at each time slot, with every UE producing one new task. In addition to the primary scenarios, we deployed an evaluation with obstacles to test the algorithm's collision avoidance capabilities. All other simulation parameters are presented in Table 4.

B. BENCHMARK APPROACHES

We validate our proposed algorithm against the following solutions:

- **RANDOM**: A UAV flies toward a randomly selected UE. Upon reaching it, a new random UE is selected as the next destination.

TABLE 4. Simulation parameters.

Notation	Value
$\|D\|$	50 UEs
$\|S\|$	10 BSs
$\|N\|$	3 or 4 UAVs
T	20
W^{max}	20 m
W^{u}	1 m
H	50 m
B	10 MHz
P_d	0.1 Watt
σ^2	-90 dBm
k_d	10^{-28}
g_o	1.42×10^{-4}
γ	0.95
K	256
τ	0.01
ε	0.001
B_n	10^5
e^{max}	1000
p_n	10
μ	0.4
ϕ	0.9995

- **CIRCLE**: All UAVs fly in a circular path with a radius of W^{max} around the geometric center of the UE cluster for the entire simulation.
- **Reinforcement Learning-based Trajectory Control (RAT) Algorithm**: This algorithm, from [28], aims to minimize UE energy consumption by optimizing user association, UAV trajectory, and resource allocation. RAT is a DRL-based trajectory control algorithm that uses prioritized experience replay to improve training convergence.
- **Optimization-embedding Multi-agent Deep Reinforcement Learning (OMADRL) algorithm**: Proposed in [27], the OMADRL algorithm seeks to achieve energy efficiency and service fairness. In this framework, each UAV autonomously learns its trajectory via MADRL, while the optimal offloading decision is found by solving a separate mixed-integer nonlinear programming problem.
  The OMADRL algorithm was implemented based on the methodology described in its original paper. Key components, including the reward function and policy gradient updates, were reproduced and validated by replicating results on benchmark scenarios. The implementation was then integrated into our simulation framework to ensure a fair comparison.

C. MUT COMPUTATIONAL COMPLEXITY ANALYSIS

This section analyzes the computational complexity of the proposed MUT framework, quantifying both the asymptotic complexity (Big-O notation) and the floating-point operations (FLOPs) for its two main components. The first component is the MADDPG-based architecture for decentralized UAV trajectory optimization. The second is the lightweight UE-UAV matching algorithm for task offloading. Both components are analyzed below to provide insight into the scalability and efficiency of the MUT framework.

1) BIG-O COMPLEXITY ANALYSIS OF MUT ALGORITHM

Our MUT algorithm applies the MADDPG approach for joint optimization. Similar to other works using MADDPG [62], the time complexity is primarily influenced by the dimensionality of the neural networks. During each training iteration, the framework collects interaction data from all N UAVs. A subset of K data samples is then drawn from this repository to update the policy (actor) and value (critic) networks of each UAV. The actor network processes state information to generate actions, while the critic network evaluates the state-action pairs.

The per-sample training complexity for an actor network is $\mathcal{O}_{\text{actor}} = \mathcal{O}(\rho\psi + \psi^2 + \psi\lambda)$, where ρ is the state space dimension, ψ is the number of neurons per hidden layer, and λ is the action space dimension. The critic network's per-sample complexity is $\mathcal{O}_{\text{critic}} = \mathcal{O}((\rho + \lambda)\psi + \psi^2 + \psi)$. Since the target networks mirror their primary counterparts, the complexity for training a single UAV's networks is given by (23):

$$\mathcal{O}_{\text{single}} = \mathcal{O}\left(2(\rho\psi + \psi^2 + \psi\lambda) + 2((\rho + \lambda)\psi + \psi^2 + \psi)\right) \tag{23}$$

Thus, for N UAVs and a minibatch of K samples, the total training complexity is given by (24):

$$\mathcal{O}\left(2NK\left(\rho\psi + \psi^2 + \psi\lambda + (\rho + \lambda)\psi + \psi^2 + \psi\right)\right) \tag{24}$$

2) FLOPs ESTIMATION FOR MUT ALGORITHM

The learning component of MUT is built around a feed-forward actor network for each UAV agent, designed to efficiently process observations and generate actions for trajectory optimization. The analysis presented here accounts only for the forward pass (inference) operations, not the backward pass and weight updates involved in training.

The input layer's size depends on the number of UAVs. For our scenarios with 50 UEs and 3 or 4 UAVs, the input vector includes the UAV's position (2 elements), relative distances to other UAVs (2 or 3 elements), accumulated service times for UEs (50 elements), and the current UE load (1 element), resulting in 55 or 56 input neurons. This structure effectively captures the spatial and service-related context for decision-making.

The network contains two hidden layers of 64 neurons each. This size, a power of 2, is a common practice in deep reinforcement learning that balances model capacity with computational efficiency. It is sufficient to learn complex relationships without the significant overhead of larger

TABLE 5. Trainable parameters per layer for 3 and 4 UAVs.

Layer	3 UAVs		4 UAVs	
	Input	Output	Input	Output
Input to Hidden 1	55	64	56	64
Hidden 1 to Hidden 2	64	64	64	64
Hidden 2 to Output	64	2	64	2

layers, and has been optimized through empirical tuning for deployment on resource-constrained UAVs. The output layer has 2 neurons representing the continuous action variables: flying direction ($\alpha_{n,t} \in [0, 2\pi)$) and distance ($dist_{n,t} \in [0, d^{max}]$).

The number of trainable parameters in a dense layer is given by the standard formula in Eq. (25) [63]:

$$\text{Parameters} = (\text{Input size} \times \text{Output size}) + \text{Output size} \tag{25}$$

Based on this, the total number of trainable parameters ζ in our actor network is:

For 3 UAVs:

$$\begin{aligned}
&\text{Layer 1:} \quad 55 \cdot 64 + 64 = 3,584 \\
&\text{Layer 2:} \quad 64 \cdot 64 + 64 = 4,160 \\
&\text{Layer 3:} \quad 64 \cdot 2 + 2 = 130 \\
&\text{Total:} \quad \zeta = 3,584 + 4,160 + 130 = \mathbf{7,874}
\end{aligned}$$

For 4 UAVs:

$$\begin{aligned}
&\text{Layer 1:} \quad 56 \cdot 64 + 64 = 3,648 \\
&\text{Layer 2:} \quad 64 \cdot 64 + 64 = 4,160 \\
&\text{Layer 3:} \quad 64 \cdot 2 + 2 = 130 \\
&\text{Total:} \quad \zeta = 3,648 + 4,160 + 130 = \mathbf{7,938}
\end{aligned}$$

A detailed breakdown of these parameters is presented in Table 5.

Using TensorFlow's profiler [64], the total floating-point operations (FLOPs) per forward pass are estimated as $\zeta_{flops} = 2 \times \zeta$, resulting in:

$$\begin{aligned}
\zeta_{flops} &= 2 \times 7,874 = 15,748 \quad \text{(3 UAVs)} \\
\zeta_{flops} &= 2 \times 7,938 = 15,876 \quad \text{(4 UAVs)}
\end{aligned}$$

It is important to distinguish between trainable parameters (ζ), which indicate memory requirements, and FLOPs (ζ_{flops}), which quantify the arithmetic workload. The FLOPs per forward pass are typically 1.5-2 times the number of parameters in dense networks [65].

The total forward pass complexity of the MADDPG component over a simulation of $\epsilon^{max} = 1000$ episodes is computed with Eq. (26):

$$\text{FLOPs}_{\text{MADDPG}} = \epsilon^{max} \cdot N \cdot \zeta_{flops} \tag{26}$$

This results in total complexities of 47,244,000 FLOPs for 3 UAVs and 63,504,000 FLOPs for 4 UAVs.

The second component of MUT is the UE-side offloading algorithm. At each time step, each of the D UEs selects the closest UAV in a greedy matching process. The complexity of this component over an entire simulation is $\text{FLOPs}_{\text{offload}} = \epsilon^{max} \cdot D \cdot N$. With $D = 50$, this totals approximately 150,000 FLOPs for 3 UAVs and 200,000 for 4 UAVs. Although this is less than 0.5% of the total execution complexity, it is included for a complete analysis of the MUT system's runtime cost.

D. RESULTS

This subsection presents a comprehensive evaluation of the proposed MUT algorithm against several benchmarks. We analyze UAV trajectories, energy consumption, multi-dimensional fairness, and performance sensitivity.

1) UAV TRAJECTORY AND OBSTACLE AVOIDANCE

Figs. 3 and 4 illustrate the optimized UAV trajectories for three and four UAVs, respectively. The proposed framework enables UAVs to dynamically adjust their paths, minimizing travel distance and overlap while maximizing service coverage for the UE population. The heatmaps confirm that service is distributed efficiently; UEs near the center receive consistent coverage, while edge UEs are visited sufficiently to maintain fairness. The addition of a fourth UAV enhances service at the periphery and demonstrates that the optimization process scales seamlessly, ensuring an even workload distribution across all UAVs.

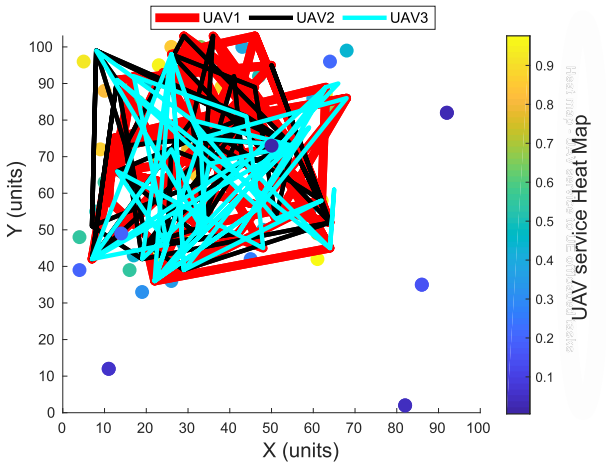

FIGURE 3. UAV trajectories (with 3 UAVs, UE positions are presented via dots).

Furthermore, we evaluated the algorithm's performance in environments with obstacles. As shown in Figs. 5 and 6, the MUT algorithm successfully enables UAVs to navigate safely between start and end points without collisions. This capability is attributed to the fast convergence of the model, which learns to prioritize obstacle avoidance. The focused view in Fig. 6 highlights the algorithm's precision

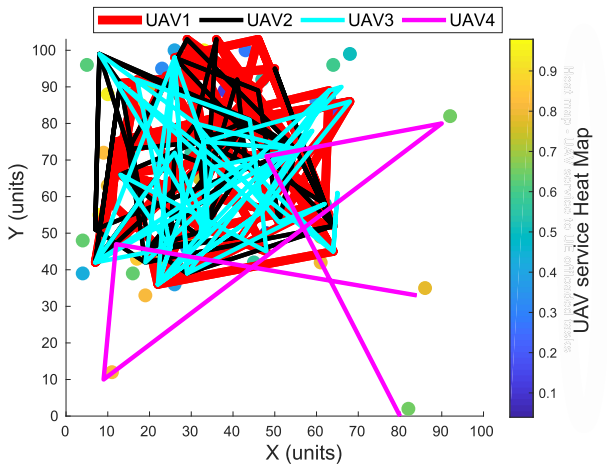

FIGURE 4. UAV trajectories (with 4 UAVs, UE positions are presented via dots).

in navigating tight spaces, reinforcing its robustness for real-world deployments.

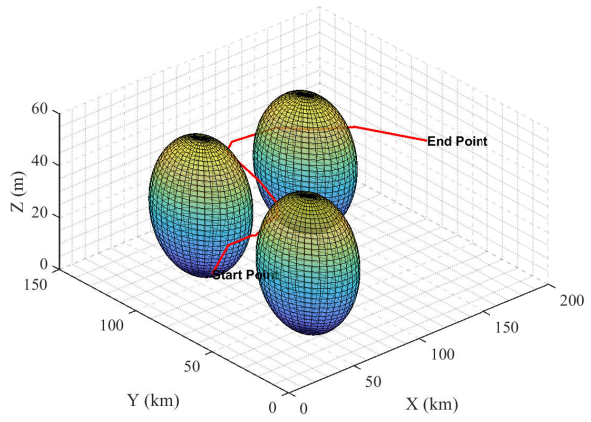

FIGURE 5. 3D visualization of the UAV collision avoidance trajectory via the proposed algorithm learned actions.

2) UE ENERGY CONSUMPTION

Figs. 7 and 8 show the total UE energy consumption, expressed as a percentage of the energy required for full local execution. In both the three-UAV and four-UAV scenarios, the MUT approach significantly outperforms all benchmarks in energy efficiency. This improvement is due to intelligent trajectory planning and resource allocation, which extends the operational lifetime of UEs. The results also indicate that increasing the number of UAVs from three to four leads to a further reduction in UE energy consumption, confirming that our system's efficiency scales with the number of available aerial relays.

3) FAIRNESS ANALYSIS AND REWARDS

We evaluated fairness from multiple perspectives. In terms of geographical fairness among UEs ($fair_t^e$), MUT achieves a

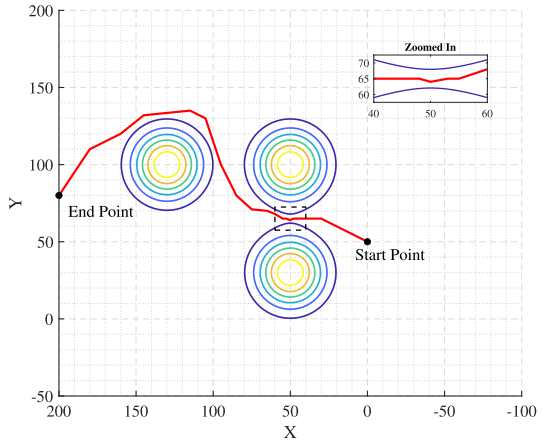

FIGURE 6. Planar (2D) top-down view of the UAV collision avoidance trajectory via the proposed algorithm learned actions.

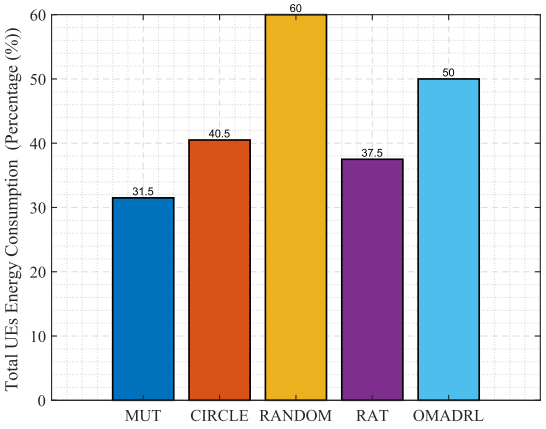

FIGURE 7. Total UEs energy consumption of MUT, CIRCLE, RANDOM, RAT and OMADRL (using 3 UAVs).

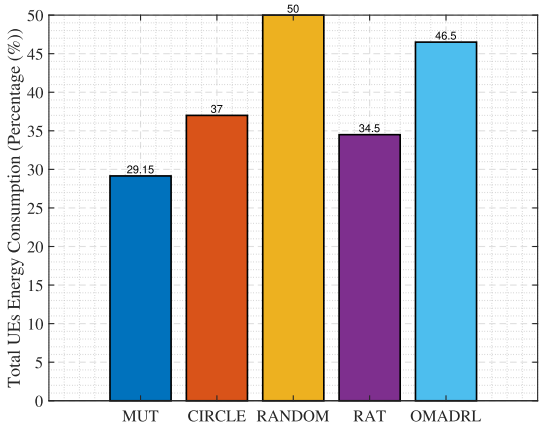

FIGURE 8. Total UEs energy consumption of MUT, CIRCLE, RANDOM, RAT and OMADRL (using 4 UAVs).

high fairness index of nearly 85% with three UAVs (Fig. 9) and improves further to 90% with four UAVs (Fig. 10), while benchmarks stagnate at lower values. This demonstrates that MUT distributes service more equitably across the UE population. For fairness among UAVs ($fair_{t'}^d$), which reflects a balanced UE load distribution, MUT also performs

exceptionally well. As shown in Figs. 11 and 12, it achieves near-perfect fairness (approaching 100%) in the four-UAV scenario, indicating uniform utilization.

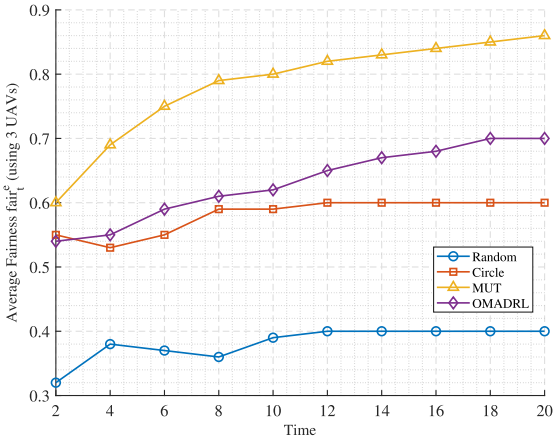

FIGURE 9. Average UE fairness $fair_t^e$ performance of MUT, CIRCLE, RANDOM and OMADRL (using 3 UAVs).

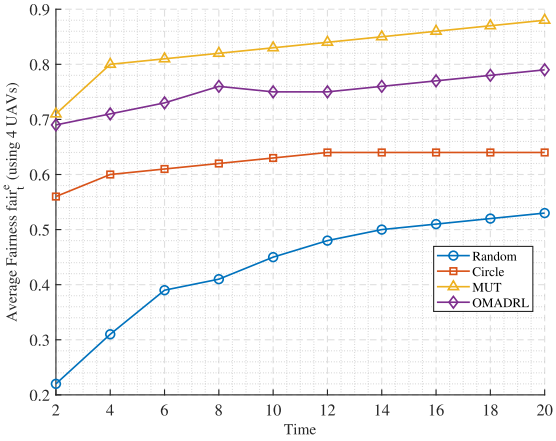

FIGURE 10. Average UE fairness $fair_t^e$ performance of MUT, CIRCLE, RANDOM and OMADRL in time (using 4 UAVs).

The superior performance is also reflected in the learning process itself. Fig. 13 shows that MUT consistently achieves higher average accumulated rewards over its training episodes compared to the RAT and OMADRL algorithms, indicating more effective and efficient learning.

Furthermore, Figs. 14 and 15 show the fairness of energy consumption and data rate distribution, respectively. In both cases, MUT maintains a fairness index consistently above 0.9, ensuring equitable resource allocation, which is critical for maintaining Quality of Service (QoS) across all UEs.

4) SENSITIVITY TO ALTITUDE AND UAV DENSITY

Finally, we analyzed the framework's sensitivity to the number of UAVs and their flight altitude. As shown in Figs. 16 and 17, increasing the number of UAVs consistently improves both final UE energy levels and the task completion rate. Lower altitudes (e.g., 50 m) yield the best performance due to shorter communication distances, which enable faster, more reliable, and more energy-efficient task offloading.

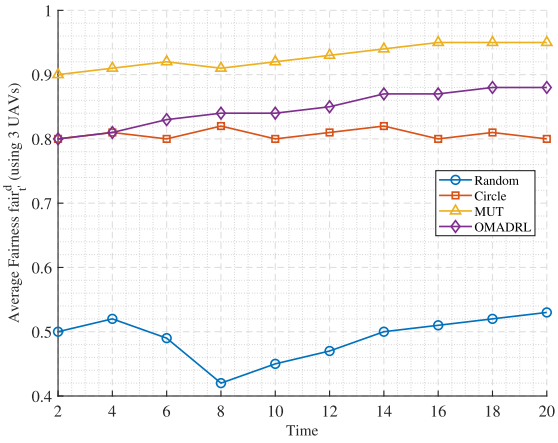

FIGURE 11. Average UAV fairness $fair_{t'}^d$ of MUT, CIRCLE, RANDOM and OMADRL in time (using 3 UAVs).

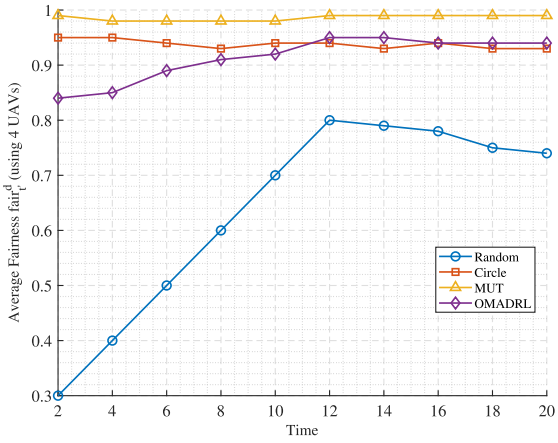

FIGURE 12. Average UAV fairness $fair_{t'}^d$ of MUT, CIRCLE, RANDOM and OMADRL in time (using 4 UAVs).

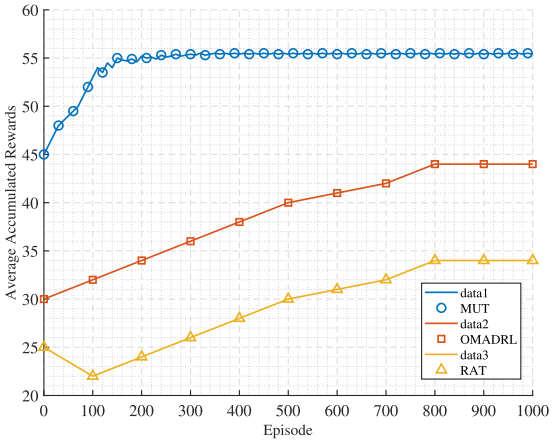

FIGURE 13. Accumulated reward versus training episodes for MUT, RAT and OMADRL algorithms.

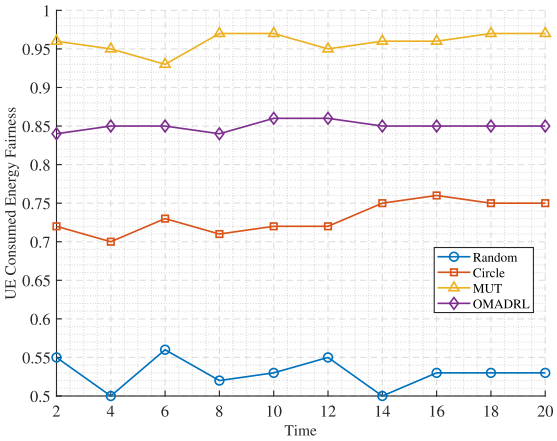

FIGURE 14. Average Fairness with respect to UE energy consumption for MUT, CIRCLE, RANDOM and OMADRL in time (using 4 UAVs).

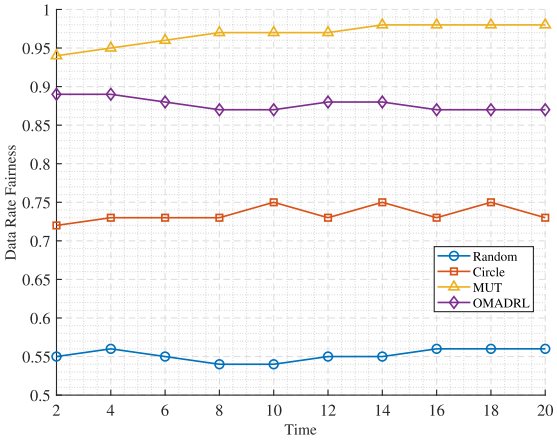

FIGURE 15. Fairness with respect to data rate distribution for MUT, CIRCLE, RANDOM and OMADRL in time (using 4 UAVs).

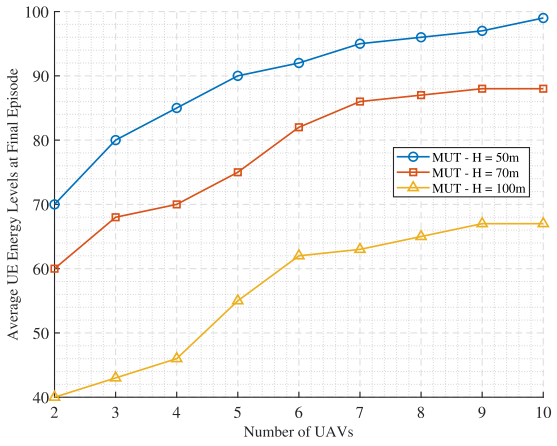

FIGURE 16. Average UE Energy Level vs Number of UAVs.

5) SUMMARY OF RESULTS

These results demonstrate the effectiveness of the proposed MUT algorithm in optimizing UAV trajectories, reducing UE energy consumption, and ensuring fair resource distribution. MUT consistently outperforms benchmark approaches across all evaluated metrics, confirming its suitability for real-world deployment. Specifically, MUT surpasses OMADRL by leveraging a dynamic reward function that integrates both fairness and energy efficiency, enabling a more balanced optimization. The centralized training with decentralized execution framework ensures scalability, while the faster convergence reduces computational overhead, enhancing real-time applicability.

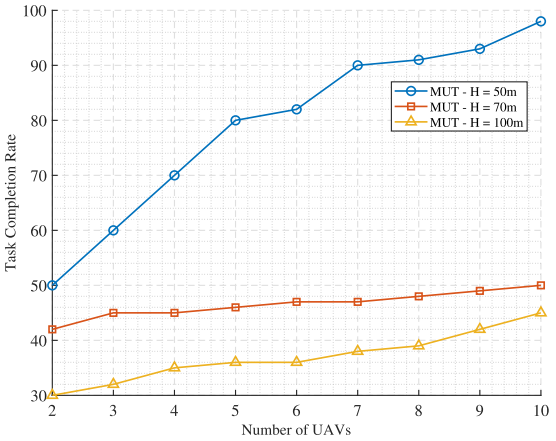

FIGURE 17. Task Completion Rate vs Number of UAVs.

VIII. CONCLUSION

This work introduced a UAV-assisted task offloading framework that combines decentralized MADDPG-based trajectory control with a low-complexity greedy offloading strategy. Our approach successfully reduces UE energy consumption and enhances fairness, with simulations demonstrating energy savings of up to 30% and fairness indices exceeding 0.9 across various scenarios.

Future work will explore several promising directions, including support for heterogeneous UAV fleets, multi-tier aerial networks, and the integration of security and privacy mechanisms. Adaptability to heterogeneous platforms can be achieved by extending the reinforcement learning (RL) model, where the reward function is adjusted to reflect each UAV's specific capabilities, such as distinct communication capacities or levels of autonomy. Furthermore, for real-time applications, task prioritization can be integrated into the offloading and scheduling mechanisms, enabling the RL agent to dynamically respond to task urgency and support latency-sensitive operations.

REFERENCES

[1] S. Sinha. (2024). *State of Iot 2024: Number of Connected Iot Devices Growing 13% To 18.8 Billion Globally*. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/

[2] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, "A survey on mobile augmented reality with 5G mobile edge computing: Architectures, applications, and technical aspects," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 1160–1192, 2nd Quart., 2021.

[3] T. K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, and N. Kato, "Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 38–67, 1st Quart., 2020.

[4] N. Zhang, S. Guo, Y. Dong, and D. Liu, "Joint task offloading and data caching in mobile edge computing networks," *Comput. Netw.*, vol. 182, Dec. 2020, Art. no. 107446.
[5] B. Ali, M. A. Gregory, and S. Li, "Multi-access edge computing architecture, data security and privacy: A review," *IEEE Access*, vol. 9, pp. 18706–18721, 2021.
[6] L. Hou, M. A. Gregory, and S. Li, "A survey of multi-access edge computing and vehicular networking," *IEEE Access*, vol. 10, pp. 123436–123451, 2022.
[7] O. Aouedi, T.-H. Vu, A. Sacco, D. C. Nguyen, K. Piamrat, G. Marchetto, and Q.-V. Pham, "A survey on intelligent Internet of Things: Applications, security, privacy, and future directions," *IEEE Commun. Surveys Tuts.*, vol. 27, pp. 1238–1292, 2nd Quart., Apr. 2025.
[8] Z. Ning, H. Hu, X. Wang, L. Guo, S. Guo, G. Wang, and X. Gao, "Mobile edge computing and machine learning in the Internet of Unmanned aerial vehicles: A survey," *ACM Comput. Surveys*, vol. 56, no. 1, pp. 1–31, Jan. 2024.
[9] J. Perazzone, M. B. Dwyer, K. Chan, C. Anderson, and S. Brown, "Enabling machine learning on resource-constrained tactical networks," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Nov. 2022, pp. 932–937.
[10] A. Ranjha, D. Naboulsi, and M. Elemary, "Towards facilitating URLLC in UAV-enabled MEC systems for 6G networks," in *Proc. Int. Symp. Ubiquitous Netw.*, Jan. 2023, pp. 55–67.
[11] T. Baidya, A. Nabi, and S. Moh, "Trajectory-aware offloading decision in UAV-aided edge computing: A comprehensive survey," *Sensors*, vol. 24, no. 6, p. 1837, Mar. 2024.
[12] A. A. Baktayan, A. Thabit Zahary, and I. Ahmed Al-Baltah, "A systematic mapping study of UAV-enabled mobile edge computing for task offloading," *IEEE Access*, vol. 12, pp. 101936–101970, 2024.
[13] C. Diaz-Vilor, M. A. Almasi, A. M. Abdelhady, A. Celik, A. M. Eltawil, and H. Jafarkhani, "Sensing and communication in UAV cellular networks: Design and optimization," *IEEE Trans. Wireless Commun.*, vol. 23, no. 6, pp. 5456–5472, Jun. 2024.
[14] I. A. Elnabty, Y. Fahmy, and M. Kafafy, "A survey on UAV placement optimization for UAV-assisted communication in 5G and beyond networks," *Phys. Commun.*, vol. 51, Apr. 2022, Art. no. 101564.
[15] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station (UAV-BS) for energy-efficient maximal coverage," *IEEE Wireless Commun. Lett.*, vol. 6, no. 4, pp. 434–437, Aug. 2017.
[16] K. Yu, Q. Cui, Z. Zhang, X. Huang, X. Zhang, and X. Tao, "Efficient UAV/Satellite-assisted IoT task offloading: A multi-agent reinforcement learning solution," in *Proc. 27th Asia–Pacific Conf. Commun. (APCC)*, Oct. 2022, pp. 83–88.
[17] N. Zhao, Z. Ye, Y. Pei, Y.-C. Liang, and D. Niyato, "Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 6949–6960, Sep. 2022.
[18] T. Cai, Z. Yang, Y. Chen, W. Chen, Z. Zheng, Y. Yu, and H.-N. Dai, "Cooperative data sensing and computation offloading in UAV-assisted crowdsensing with multi-agent deep reinforcement learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3197–3211, Sep. 2022.
[19] X. Zhang, J. Wang, B. Wang, and F. Jiang, "Offloading strategy for UAV-assisted mobile edge computing based on reinforcement learning," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Aug. 2022, pp. 702–707.
[20] A. Gao, Q. Wang, W. Liang, and Z. Ding, "Game combined multi-agent reinforcement learning approach for UAV assisted offloading," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 12888–12901, Dec. 2021.
[21] K. Chen, A. Gao, W. Duan, and W. Liang, "Matching combined multi-agent reinforcement learning for UAV secure data dissemination," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2022, pp. 3367–3370.
[22] A. Gao, Q. Wang, K. Chen, and W. Liang, "Multi-UAV assisted offloading optimization: A game combined reinforcement learning approach," *IEEE Commun. Lett.*, vol. 25, no. 8, pp. 2629–2633, Aug. 2021.
[23] H. Wang, J. Wang, G. Ding, J. Chen, F. Gao, and Z. Han, "Completion time minimization with path planning for fixed-wing UAV communications," *IEEE Trans. Wireless Commun.*, vol. 18, no. 7, pp. 3485–3499, Jul. 2019.
[24] F. Cui, Y. Cai, Z. Qin, M. Zhao, and G. Y. Li, "Multiple access for mobile-UAV enabled networks: Joint trajectory design and resource allocation," *IEEE Trans. Commun.*, vol. 67, no. 7, pp. 4980–4994, Jul. 2019.
[25] J. Wang, C. Jiang, Z. Wei, C. Pan, H. Zhang, and Y. Ren, "Joint UAV hovering altitude and power control for Space-Air-Ground IoT networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1741–1753, Apr. 2019.
[26] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 2109–2121, Mar. 2018.
[27] X. Li, X. Du, N. Zhao, and X. Wang, "Computing over the sky: Joint UAV trajectory and task offloading scheme based on optimization-embedding multi-agent deep reinforcement learning," *IEEE Trans. Commun.*, vol. 72, no. 3, pp. 1355–1369, Mar. 2024.
[28] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and A. Nallanathan, "Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 21, no. 10, pp. 3536–3550, Oct. 2022.
[29] D. Tyrovolas, N. A. Mitsiou, T. G. Boufikos, P.-V. Mekikis, S. A. Tegos, P. D. Diamantoulakis, S. Ioannidis, C. K. Liaskos, and G. K. Karagiannidis, "Energy-aware trajectory optimization for UAV-mounted RIS and full-duplex relay," *IEEE Internet Things J.*, vol. 11, no. 13, pp. 24259–24272, Jul. 2024.
[30] P.-V. Mekikis, P. S. Bouzinis, N. A. Mitsiou, S. A. Tegos, D. Tyrovolas, V. K. Papanikolaou, and G. K. Karagiannidis, "Enabling wireless-powered IoT through incentive-based UAV swarm orchestration," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 2548–2560, 2023.
[31] Y. Wang, J. Zhu, H. Huang, and F. Xiao, "Bi-objective ant colony optimization for trajectory planning and task offloading in UAV-assisted MEC systems," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 12360–12377, Dec. 2024.
[32] M. Zhao, R. Zhang, Z. He, and K. Li, "Joint optimization of trajectory, offloading, caching, and migration for UAV-assisted MEC," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 1981–1998, Mar. 2025.
[33] C. Zheng, K. Pan, J. Dong, L. Chen, Q. Guo, S. Wu, H. Luo, and X. Zhang, "Multi-agent collaborative optimization of UAV trajectory and latency-aware DAG task offloading in UAV-assisted MEC," *IEEE Access*, vol. 12, pp. 42521–42534, 2024.
[34] J. Ma, S. Xu, and X. Song, "Resource allocation and task offloading for mobile edge computing system based on UAV-EC collaboration," in *Proc. IEEE 24th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2024, pp. 207–211.
[35] S. Barick and C. Singhal, "UAV-assisted MEC architecture for collaborative task offloading in urban IoT environment," *IEEE Trans. Netw. Service Manage.*, vol. 22, no. 1, pp. 732–743, Feb. 2025.
[36] Y. Lu and Z. Luo, "An effective scheme for delay minimization in a Multi-UAV-Enabled NOMA-MEC system," *IEEE Commun. Lett.*, vol. 29, no. 1, pp. 40–44, Jan. 2025.
[37] W. Zhan, C. Luo, G. Min, C. Wang, Q. Zhu, and H. Duan, "Mobility-aware multi-user offloading optimization for mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3341–3356, Mar. 2020.
[38] T. Shi, T. Zhang, R. Zhong, Y. Liu, and R. Huang, "Cross-user dependent task offloading and resource allocation in spatial–temporal dynamic MEC networks," *IEEE Trans. Veh. Technol.*, vol. 73, no. 10, pp. 15584–15597, Oct. 2024.
[39] H. Min, A. M. Rahmani, P. Ghaderkourehpaz, K. Moghaddasi, and M. Hosseinzadeh, "A joint optimization of resource allocation management and multi-task offloading in high-mobility vehicular multi-access edge computing networks," *Ad Hoc Netw.*, vol. 166, Jan. 2025, Art. no. 103656. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870524002671
[40] K. Moghaddasi, S. Rajabi, F. S. Gharehchopogh, and A. Ghaffari, "An advanced deep reinforcement learning algorithm for three-layer D2D-edge-cloud computing architecture for efficient task offloading in the Internet of Things," *Sustain. Computing: Informat. Syst.*, vol. 43, Sep. 2024, Art. no. 100992. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210537924000374
[41] K. Moghaddasi, S. Rajabi, and F. S. Gharehchopogh, "Multi-objective secure task offloading strategy for blockchain-enabled IoV-MEC systems: A double deep Q-Network approach," *IEEE Access*, vol. 12, pp. 3437–3463, 2024.
[42] K. Moghaddasi, S. Rajabi, F. Soleimanian Gharehchopogh, and M. Hosseinzadeh, "An energy-efficient data offloading strategy for 5G-enabled vehicular edge computing networks using double deep Q-Network," *Wireless Pers. Commun.*, vol. 133, no. 3, pp. 2019–2064, Dec. 2023.
[43] Y. Chen, D. Pi, S. Yang, Y. Xu, B. Wang, S. Qin, and Y. Wang, "A dynamic optimization framework for computation rate maximization in UAV-assisted mobile edge computing," *IEEE Trans. Veh. Technol.*, early access, Mar. 10, 2025, doi: 10.1109/TVT.2025.3546026.

[44] L. Zhang, S.-K. Oh, W. Pedrycz, B. Yang, and L. Wang, "A promotive particle swarm optimizer with double hierarchical structures," *IEEE Trans. Cybern.*, vol. 52, no. 12, pp. 13308–13322, Dec. 2022.
[45] Z. Hu, D. Zhou, C. Shen, T. Wang, and L. Liu, "Task offloading strategy for UAV-assisted mobile edge computing with covert transmission," *Electronics*, vol. 14, no. 3, p. 446, Jan. 2025.
[46] N. Gupta, S. Agarwal, D. Mishra, and B. Kumbhani, "Trajectory and resource allocation for UAV replacement to provide uninterrupted service," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 7288–7302, Dec. 2023.
[47] Y. Wang and Z. Su, "An envy-free online UAV charging scheme with vehicle-mounted mobile wireless chargers," *China Commun.*, vol. 20, no. 8, pp. 89–102, Aug. 2023.
[48] K. Zhu, J. Yang, Y. Zhang, J. Nie, W. Y. B. Lim, H. Zhang, and Z. Xiong, "Aerial refueling: Scheduling wireless energy charging for UAV enabled data collection," *IEEE Trans. Green Commun. Netw.*, vol. 6, no. 3, pp. 1494–1510, Sep. 2022.
[49] Z. Bouček and M. Flídr, "Mission planner for uav battery replacement," in *IEEE Int. Conf. Multisensor Fusion Integr. Intell. Syst. (MFI)* Sep. 2024, pp. 1–6.
[50] X.-Y. Yu, W.-J. Niu, Y. Zhu, and H.-B. Zhu, "UAV-assisted cooperative offloading energy efficiency system for mobile edge computing," *Digit. Commun. Netw.*, vol. 10, no. 1, pp. 16–24, Feb. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S235286482200027X
[51] S. H. Alsamhi, A. V. Shvetsov, S. Kumar, J. Hassan, M. A. Alhartomi, S. V. Shvetsova, R. Sahal, and A. Hawbani, "Computing in the sky: A survey on intelligent ubiquitous computing for UAV-assisted 6G networks and industry 4.0/5.0," *Drones*, vol. 6, no. 7, p. 177, Jul. 2022.
[52] M. El-Emary, A. Ranjha, D. Naboulsi, and R. Stanica, "Energy-efficient task offloading and trajectory design for UAV-based MEC systems," in *Proc. 19th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Jun. 2023, pp. 274–279.
[53] J. Ji, K. Zhu, C. Yi, and D. Niyato, "Energy consumption minimization in UAV-assisted mobile-edge computing systems: Joint resource allocation and trajectory design," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8570–8584, May 2021.
[54] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen, "Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3424–3438, Mar. 2020.
[55] A. Khele, C. Jiang, and H. Wang, "Fairness-aware optimization of vehicle-to-vehicle interaction for smart EV charging coordination," in *Proc. IEEE/IAS 59th Ind. Commercial Power Syst. Tech. Conf. (I&CPS)*, May 2023, pp. 1–9.
[56] S. Lai, X. Fan, Q. Ye, Z. Tan, Y. Zhang, X. He, and P. Nanda, "FairEdge: A fairness-oriented task offloading scheme for IoT applications in mobile cloudlet networks," *IEEE Access*, vol. 8, pp. 13516–13526, 2020.
[57] J. Li, Q. Liu, P. Wu, F. Shu, and S. Jin, "Task offloading for UAV-based mobile edge computing via deep reinforcement learning," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Aug. 2018, pp. 798–802.
[58] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[59] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2015, *arXiv:1511.05952*.
[60] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2014, pp. 387–395.
[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[62] J. Du, Z. Kong, A. Sun, J. Kang, D. Niyato, X. Chu, and F. R. Yu, "MADDPG-based joint service placement and task offloading in MEC empowered Air–Ground integrated networks," *IEEE Internet Things J.*, vol. 11, no. 6, pp. 10600–10615, Mar. 2024.
[63] G. Huang, Z. Liu, G. Pleiss, L. V. D. Maaten, and K. Q. Weinberger, "Convolutional networks with dense connectivity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 8704–8716, Dec. 2022.
[64] S. W. D. Chien, A. Podobas, I. B. Peng, and S. Markidis, "Tf-darshan: Understanding fine-grained I/O performance in machine learning workloads," in *Proc. IEEE Int. Conf. Cluster Comput. (CLUSTER)*, Sep. 2020, pp. 359–370.
[65] J. Sevilla, L. Heim, M. Hobbhahn, T. Besiroglu, A. Ho, and P. Villalobos. (2022). *Estimating Training Compute of Deep Learning Models*. [Online]. Available: https://epoch.ai/blog/estimating-training-compute

MOHAMED EL-EMARY received the M.Sc. degree in electronics and communication engineering from the Arab Academy for Science, Technology, and Maritime Transport, Egypt, in 2018. He is currently pursuing the Ph.D. degree in software and IT engineering with École de Technologie Supérieure (ÉTS), Montreal, QC, Canada. From 2012 to 2020, he was a Network Consultant with Orange Business Services, Egypt. His current research interests include next-generation mobile communication systems, machine learning, and resource management.

ALI RANJHA received the M.S. degree in innovation in telecommunications from Lancaster University, Lancaster, U.K., in 2018, under a prestigious Higher Education Funding Council of England Bursary, and the Ph.D. degree from École de Technologie Supérieure (ÉTS), Université du Québec, Montreal, QC, Canada, in January 2022. He is currently a Postdoctoral Researcher with ÉTS, Université du Québec. His research interests include diverse areas, such as fundamental communication theory, unmanned aerial vehicle communications, the Internet of Things, ultra-reliable and low-latency communications, and optimization in resource-constrained networks.

DIALA NABOULSI (Member, IEEE) received the Ph.D. degree in computer science from INSA Lyon, Villeurbanne, France, in 2015. She is currently an Associate Professor with École de Technologie Supérieure (ÉTS), Canada. She has been involved in many Canadian and European research projects, with a strong record of industrial collaborations. Her research interests include mobile networks, virtualized networks, and wireless networks.

RAZVAN STANICA received the M.Eng. and Ph.D. degrees in computer science from INP Toulouse, France, in 2008 and 2011, respectively. He is currently an Associate Professor with INSA Lyon, France, and a Research Scientist with the Inria Agora Team, CITI Laboratory. His research interests include wireless mobile networks, with a special focus on communication networks in urban environments.