frontiers in Communications and Networks

ORIGINAL RESEARCH
published: 15 September 2021
doi: 10.3389/frcmn.2021.709265

Delivering Resources for Augmented Reality by UAVs: a Reinforcement Learning Approach

Damiano Brunori[1], Stefania Colonnese[2], Francesca Cuomo[2], Giovanna Flore[1] and Luca Iocchi[1]*

[1]Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, Rome, Italy, [2]Dipartimento di Ingegneria dell'Informazione, Elettronica e Telecomunicazioni, Sapienza Università di Roma, Rome, Italy

Unmanned aerial vehicles (UAVs) are supposed to be used to provide different services from video surveillance to communication facilities during critical and high-demanding scenarios. Augmented reality streaming services are especially demanding in terms of required throughput, computing resources at the user device, as well as user data collection for advanced applications, for example, location-based or interactive ones. This work is focused on the experimental utilization of a framework adopting reinforcement learning (RL) approaches to define the paths crossed by UAVs in delivering resources for augmented reality services. We develop an OpenAI Gym-based simulator that is tuned and tested to study the behavior of UAVs trained with RL to fly around a given area and serve augmented reality users. We provide abstractions for the environment, the UAVs, the users, and their requests. A reward function is then defined to encompass several quality-of-experience parameters. We train our agents and observe how they behave as a function of the number of UAVs and users at different hours of the day.

Keywords: UAVs networks, multi-service, reinforncement learning, augmented reality, simulation

OPEN ACCESS

Edited by:
Jaap Van De Beek, Luleå University of Technology, Sweden

Reviewed by:
Hajar El Hammouti, King Abdullah University of Science and Technology, Saudi Arabia
Atefeh Hajijamali Arani, University of Waterloo, Canada

****Correspondence:***
Damiano Brunori
brunori@diag.uniroma1.it

Specialty section:
This article was submitted to Aerial and Space Networks, a section of the journal Frontiers in Communications and Networks

Received: *13 May 2021*
Accepted: *26 July 2021*
Published: *15 September 2021*

Citation:
Brunori D, Colonnese S, Cuomo F, Flore G and Iocchi L (2021) Delivering Resources for Augmented Reality by UAVs: a Reinforcement Learning Approach. Front. Comms. Net 2:709265. doi: 10.3389/frcmn.2021.709265

1 INTRODUCTION

Latest advancements in science and communication unlock the opportunity to employ in everyday life new and exciting technologies that will improve the experience of the users in surprising and innovative ways. Augmented reality (AR) is the perfect example: it is becoming a service frequently desired for a wide range of purposes, but it requires high data rates, heavy rendering computation resources at the user device, and support for a rich application layer feedback channel for advanced services such as geo-referenced ones.

Among others, next-generation networks are expected to support advanced multimedia services, encompassing single, multi-view, or 360° video sequences, as well as more complex visual data involving 2D/3D natural and synthetic video objects (Mangiante et al. 2017) referred to as mixed and augmented reality, ISO/IEC JTC-1 (2016). In Milani et al. (2020), Taha et al. (2020), and Sheikhipour et al. (2019), static and dynamic representations are studied for textured point clouds, mesh, and volumetric data. Fujihashi et al. (2019) propose an end-to-end transmission scheme for 3D point clouds with visual information, presenting results related to the open 3D point cloud database Pan et al. (2020). Furthermore, the growing interest to represent this kind of object in several applications has solicited a parallel standardization activity as in Jang et al. (2019). New standards have been released Lafruit et al. (2019) with the purpose of 3D static and dynamic visual content compression for immersive reality multimedia services (360° video with head-mounted displays and free navigation in 3D space with

head-mounted and 3D light field displays). Augmented reality streaming toward a mobile device represents typical challenges discussed in Lai et al. (2019) and Chakareski (2020); the quality of experience is related to the visual content Han et al. (2020), Zhang et al. (2020), rate adaptation Hosseini and Timmerer (2018), Liu et al. (2020), and device power consumption Cheng (2020).

The increasing popularity of these high-tech facilities will indubitably increase the network traffic and the users' needs for huge amounts of both exchanged data and computational capacity. The support of such services is still very challenging, especially in rural environments or where high bandwidth interconnections are unavailable. Since the existing network infrastructures may not be capable of sustaining the heavy loads associated to AR, unmanned aerial vehicles (UAVs) come into play. Resources can be delivered to users without the need for modifying the preexisting infrastructure, making these new services feasible, sustainable, and accessible.

1.1 Our Goal

In this article, we address the specific use case of UAVs serving AR users. We model the needs of the AR users in a simple and effective way, that is, as a composition of requests of throughput provisioning, edge mobile computing, and data gathering. We represent UAVs, space and time in a realistic but computationally affordable way. In order to make UAVs autonomous, we train them through reinforcement learning (RL) and observe their behavior changing throughout the various epochs. We encompass different environments and situations, showing how (in percentage) the users' demands are satisfied and taking into account various metrics for the quality of experience (QoE) of AR services. The ultimate goal is to provide a framework to test and evaluate these challenging services in areas where the infrastructure is supported by UAVs as well as algorithms and methods to compute optimal behaviors maximizing some performance metrics. RL method allows dealing with the big amount of parameters required by such complex and challenging problems without the need of any modeling, resulting to be a smart and fast technique. The use of a proper simulation framework also allows us to execute many experiments to learn optimal policies (i.e., behaviors) and to analyze the impact of some parameters on the solution. Such policies can then be used to produce flight plans for real UAVs. Finally, producing relevant trajectories for specific missions is an important task for collision risk analysis. Thus, this work contributes also to the definition and validation of a UAV collision risk model, developed within the BUBBLES project[1].

1.2 Reinforcement Learning Advantages for UAV Applications

The main motivation for using RL as a basic technique to solve the considered problem is given by the advantage of using a simulation tool instead of a formal model. Indeed, in complex cases and scenarios like the one considered in this article, the number of parameters to be modeled may be very high and it increases with the complexity of the scenarios. Model-based approaches tend to provide optimal solutions for the features that are explicitly modeled, but sometimes formal methods do not simply allow for modeling some complex aspects of the system we want to analyze. On the other hand, developing simulation tools for such scenarios is a more common choice that can also exploit previous work in this area. Reinforcement learning is a preferred solution when, like in the cases considered in this article, providing a simulation tool for a scenario is more convenient and effective than having a formal model.

The rest of the article is organized as follows. **Section 2** discusses some key related works while **Section 3** introduces the considered scenario. The adopted models are described in **Section 4**. Experimental settings are discussed in **Section 5** and the achieved results in **Section 6**. The conclusions are given in **Section 7**.

2 RELATED WORK

UAVs may be employed for a large range of services due to the fact that a big interest is arising in having them as a part of the network infrastructure. The use of UAVs to provide services from the sky and its main challenges are described in Zeng et al. (2019), while Ferranti et al. (2020) describes SkyCell: a prototyping platform for 5G autonomous aerial base stations demonstrating the feasibility of an aerial base station where wireless backhaul, autonomous mobility, and 5G functionalities are integrated within a unified framework.

UAVs' features can be exploited also in an AR service scenario. In Chakareski (2019), a system of UAV is employed to capture different views, and reinforcement learning is employed to drive the UAV in such a way as to maximize video reconstruction fidelity at the remote user, subject to given per-UAV and overall bit-rate budget. In Tan et al. (2020), UAVs provide cache-enabled edge computing elements to support social augmented reality services, while in Santos et al. (2021) interactive augmented reality is related to the potential of UAVs in the evolution of digital photogrammetry toward georeferencing is explored. Thus, UAV deployment paves the way for the development of advanced mobile augmented reality services.

In general, when we have to face complex services like AR, first we need to deal with the complexity in providing this service through UAVs integrated with (or replacing part of) the communication infrastructure. Then, we have to provide their fundamental components like the bandwidth, the computing capability, the data gathering, and also the energy for transmission and movement in the case of aerial nodes like the UAVs. Jiang et al. (2019) show how cache-enabled UAVs are used to assist mobile-edge computing: here the best position among Internet of Things devices is searched to maximize data throughput. A study of the optimal path using Q-Learning is performed in Zouaoui et al. (2019), taking into account the QoE and the energy consumption, while Colonnese et al. (2019) focuses on the study on the support of video services and their quality and delay optimization. With respect to these two latter works, the framework used in our work allows UAVs to

[1]http://bubbles-project.eu/.

move according to a more accurate and precise path: indeed drones can fly between cells (in which the considered map has been initially split), without letting them move only between Point of Interests.

Wang et al. (2019) instead discuss the ways to adapt UAV deployment to the best provisioning of instantaneous wireless traffic in a given territory. The authors propose an adaptive deployment scheme for a UAV-aided communication network, where the UAV adapts its displacement direction and distance to serve the instantaneous traffic of randomly moving users in the target cell.

Optimal joint resource allocation and path planning are also definitively relevant for multi-UAV development. One possible application for which path planning is needed is that of energy-efficient mobile computing Zhou et al. (2018). Energy efficiency is very desirable as it has been observed in Chakareski et al. (2019), where a framework is designed for a multi-tier multi-band mmWave cellular network integrating UAV-based aerial small cells. Energy efficiency is taken into account also in works focused on data gathering such as Liu et al. (2021), where deep-learning techniques are used to train multiple UAVs ensuring a fair service for all the considered users. When the energy is also provided by renewable sources, like in Chiaraviglio et al. (2019), the plan of the UAV missions has to take into account the grid-connected microgenerations.

Multiple UAVs can also provide services of wireless power transmission for multiple energy receivers (ERs), as described in Wang et al. (2017), Zhang et al. (2019) and Xu et al. (2017). In Wang et al. (2017) and Zhang et al. (2019), alongside the power transfer service, UAVs also provide clients (IoTs) edge-computing, with tasks offloading which can be binary, for example, Wang et al. (2017), or partial, for example, Zhang et al. (2019). In Xu et al. (2017), a trajectory is designed to maximize the amount of energy transferred to all ERs during a finite charging period. Zhou et al. (2019) uses deep learning to pre-train a UAV employed to collect data from various IoTs devices trying to minimize the Age of Information (AoI) to enhance data freshness. Bertizzolo et al. (2020) propose a "Swarm Control" to easily manage multiple UAVs: a software-defined framework centralized control for UAV wireless networks based on distributed optimization principles. This allows to set and modify the behavior of a whole aerial network in real-time. In Cheng (2020), a model for mobile augmented reality users and an approach merging caching and cloud computing to AR is provided. The composition of AR and edge computing is also studied in Tan et al. (2020), where multiple UAVs allow caching for social AR applications.

Our work starts from a multi-agent environment defined in Brunori et al. (2021) and extends it with additional features, providing more complex and realistic implementation details. The major novelty of our work with respect to the other ones is the original model of the AR user demands, defined as a composition of requests for different resources and modeled as continuous functions over the timeline (24 h). Our framework allows replicating a realistic scenario in which the requests vary according to the moment of the day. Moreover, it allows to render two different environments (urban and rural) and observe the different impacts that UAVs can have in serving AR users in these scenarios.

3 OPERATIONAL SCENARIO

We consider an operational scenario in which there exist multiple AR-user requests (see **Figure 1**). The demands for AR applications translate into users requesting at the same time and jointly three services identified as 1) throughput to support high data rate AR video download, 2) edge computing for efficient AR rendering, and 3) data collection for customized user services based on user-generated data (e.g., location, speed, and acceleration). The users are scattered in a grid-like area where some charging stations are also placed. Multiple UAVs operate in the area. Each UAV starts from one of the charging stations placed in the considered area and reaches the users to serve them. This service is constrained by the limited UAV battery that is supposed to last 30 min in our experiments; however, the proposed tool can be used in different environments built on top of the proposed setting.

3.1 Space and Time

The proposed setting is based on an operational area that is a 3D grid divided into $10 \times 10 \times 10$ cells. Each cell has a size of 25 m and thus the agent, that is, the UAV, is able to traverse the area one cell at a time at a speed v (m/s): in our experiments, an agent can go from the center of a cell to the center of another one in 5 s. The building-like obstacles are scattered on the area and the users can be inside or outside these buildings, which obviously have to be avoided by UAVs. We pose also the *charging stations* modeled as low buildings (platforms) acting as a home base for the UAVs which take off, land, charge, and pause in the idle mode on top of them. We designed two different kinds of areas: an urban one with frequent and tall buildings (**Figure 2A**) and a rural one with smaller and rarer obstacles (**Figure 2B**).

We model our AR high requests of demanding users as a composition of sub-requests for three different services:

- *Throughput-TH*: due to the high visual complexity of AR content, the AR user needs bandwidth to maintain a connection with the servers and exchange the needed data;
- *Edge Computing-EC*: offload of heavy computing tasks will be crucial to make AR feasible for a generic user, which may not possess the needed resources to sustain the high computational tasks that AR requires, including the rendering of virtual user-dependent AR views; the idea is then to provide some offloading capabilities toward the UAVs that are here assumed to have the needed computational resources;
- *Data Gathering-DG*: in order to merge reality with the virtual world providing rich, interactive, geo-referenced AR services, it will be needed to gather data about the environment of the users and surroundings. Moreover, data are also commonly fetched either to perform statistic analysis or to keep relevant information about the users.

Thus, each user generates a general request (expressed for sake of convenience in kbps) related to service of throughput, edge computing, or data gathering. The time is assumed to be split into timeslots of f seconds and each movement of a UAV lasts exactly one time slot. UAVs perceive time as integers (i.e., timeslots),

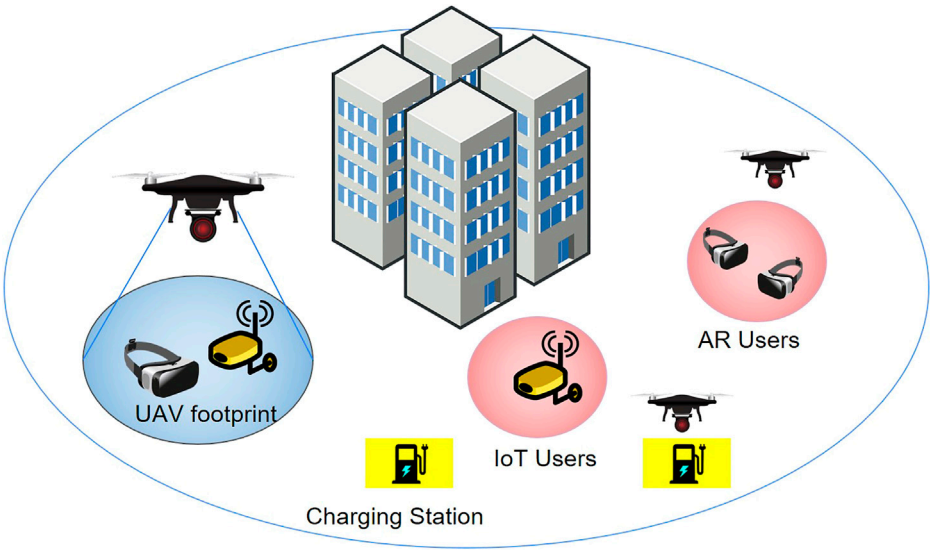

FIGURE 1 | An urban area where multiple UAVs operate to serve some AR and IoT users asking for service; some charging stations are placed in the considered area.

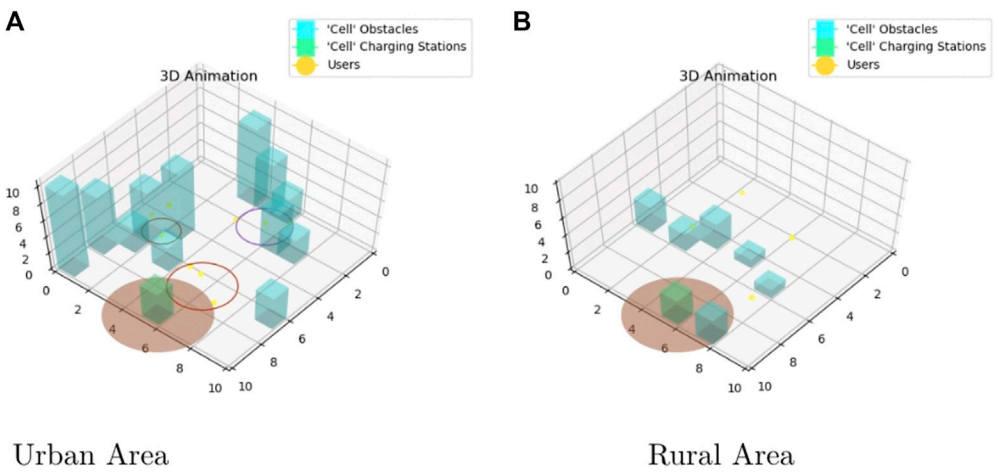

FIGURE 2 | Two different areas: Urban area **(A)** and rural area **(B)**.

while the requests of the users are defined by using a function that takes as input the time as a real number h (described in the next section).

Given an initial hour (h_0), the timing of the user request at the $n-th$ timeslot is computed as follows:

$$h = h_0 + (n * t_s/3600)\ [Hours], \tag{1}$$

where $h_0 \in [0, 24]$ and t_s is the timeslot duration expressed in seconds. Notice that the resulting h is in $\in [0, 24) \subset \mathbb{R}$.

The simulations considered in this article last for 30 min, which is also the battery autonomy B of each UAV. The channels for wireless communication to the UAVs are assumed to be ideal since the scope of the proposed approach is to evaluate the behavior of the UAV as a function of the users' requests.

3.2 Model of User Requests

To model the user requests, we derived from the literature some significant curves as for the time evolution of the three services above. The evolution of throughput curve (3) is obtained by artificially recreating the *Columbia University Commodity Internet Traffic (Aggregate)* curves representing the data collected in the Columbia University campus Columbia University Information Technology (n.a.). Very similar curves can be found also in Graham-Cumming (2021), which deals with

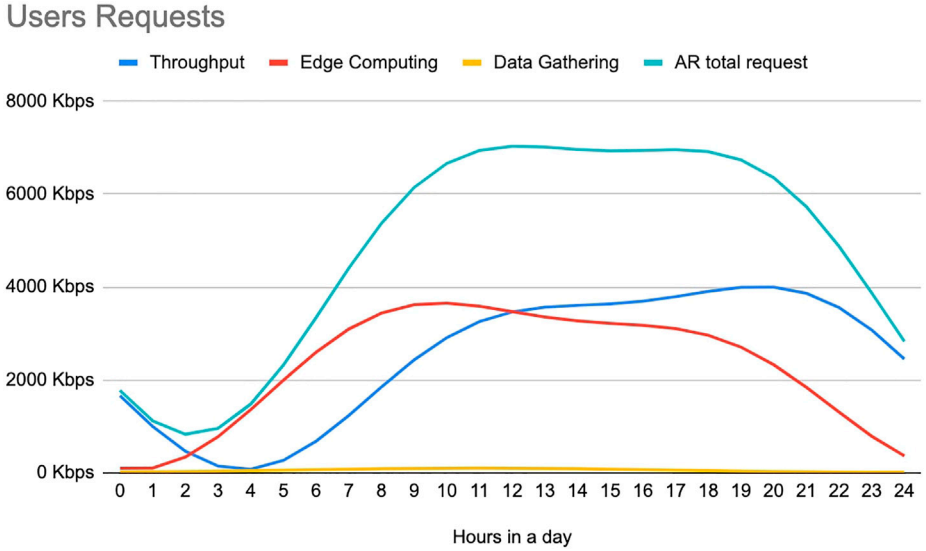

FIGURE 3 | Evolution of the three service requests and their composition during the day.

the internet traffic during the first COVID-19 outbreaks and the following lockdown: we focused on data traffic related to Northern Italy. For what concerns the edge computing curve equation(**4**), it is shaped in such a way as to replicate the trend of the requests received from the Akamai servers [Peill-Moelter (2012), Palasamudram et al. (2012)]. The last curve (5) representing the data gathering requests has been modeled following a Gaussian evolution, with a peak assumed around 12 PM: this is mainly due to the lack of data available in the literature.

The curves were designed according to the following assumptions:

- The request of throughput slightly increases in the evening, when high-quality entertainment services will be required most.
- The request of edge computing slightly increases during the morning, to account for typically higher user mobility.
- The request of data gathering is modeled after a flat Gaussian curve to represent a permanent, although low, throughput request.

Given a time hour value h, then the overall request for the AR service is defined as a weighted sum of $TH(h)$, $EC(h)$, $DG(h)$, respectively denoting the throughput, the edge computing and data gathering request functions. The resulting AR request is computed as follows:

$$AR(h) = \alpha_{AR} * TH(h) + \beta_{AR} * EC(h) + \gamma_{AR} * DG(h) \ [Kpbs], \tag{2}$$

where α_{AR}, β_{AR}, and γ_{AR} are the coefficients (whose values in the simulations belong to the interval [0.05, 1]) encompassing different weights that can be assigned to the three type of resources. The services requests $TH(h)$, $EC(h)$, $DG(h)$ are modeled during the 24 h. For the sake of concreteness, we specify the amount of kbps requested for each service depending on the hour of the day in which the requests occur:

$$TH(h) = P1 * \sin\left(\frac{h}{4} + \frac{\pi}{8}\right) + P2 * \sin\left(\frac{h}{2} + 2.2 + \frac{\pi}{4}\right) + P3 \ [kpbs], \tag{3}$$

$$EC(h) = -0.9 * \left(P1 * \sin\left(\frac{h}{4} + \frac{\pi}{2}\right) + P2 * \sin\left(\frac{h}{2} - 2 + \pi\right) - P3\right) \ [kpbs], \tag{4}$$

$$DG(h) = \mathcal{N}(\sigma, \mu, h) = \max\left(\frac{1401}{\sigma\sqrt{2\pi}} * e^{-\frac{(h-\mu)^2}{2\sigma^2}}, 1\right) \ [kpbs], \tag{5}$$

where $\sigma = 31.2$ and $\mu = 11$ represent, respectively, variance and expected value.

The curves in **Figure 3** can be flexibly tuned with the parameters P1, P2, and P3. In **Table 1** we report the values assigned to these variables in our case study, chosen to reach a peak of 4000 kpbs to simulate a typical high request in the case of AR. **Figure 3** shows the AR request with these three coefficients all set to 1, corresponding to a simulation with the maximum possible request for each user at each moment of the day. The edge computing and throughput requests are higher with respect to data gathering, indeed they are in the order of Mbps, changing drastically from day to night. On the other hand, the data gathering requests are much lower (100 kbps at their peak) and it is mainly due to the fact that the information related to this service uses less bandwidth, and it is not necessarily tied to a physical user.

Table 1 summarizes the main parameters used in the equation previously defined, providing a direct association between the meaning and the value of each parameter.

3.3 UAVs

Each UAV moving in the considered area takes off from a charging station acting as a home base, and it can move up,

TABLE 1 | Scenario parameters used in the experiments.

Parameter	Name	Values		
		First scenario	Second scenario	Third scenario
V	UAV speed	5 m/s	5 m/s	5 m/s
t_s	Timeslot duration	5 s	5 s	5 s
B	Battery autonomy	30 min	30 min	30 min
P1	First tuning parameter	1836	1836	1836
P2	Second tuning parameter	612	612	612
P3	Third tuning parameter	2,501	2,501	2,501
h_0	Initial hour	21	10	10
U	Number of UAVs	1	From 1 to 3	1,2
N	AR-users asking for a service	From 1 to 15	5, 10, 15	4
M	Number of charging stations	1	1	1,2
E	Number of episodes	20,000	20,000	30,000

down, left, right, forward, and backward. It can also fly in place (Hover), rest (Idle), or charge in the charging station. The goal of each UAV is to find and serve most of the users in a span of half an hour providing the highest QoE. It has limited bandwidth and limited battery. UAVs are also battery aware and are supposed to learn to choose the right moment to go to the closest charging station and charge. They also need to learn where the obstacles are placed in order to avoid them and not failing.

4 REINFORCEMENT LEARNING ENVIRONMENT

Our environment involves an operating volume that is made up of K small cells whose size A_k, $k = 0, \cdots K-1$ can be selected based on the desired resolution per cell. Obstacles can be randomly generated and placed on the operational volume and also different heights for each building are randomly generated.

We can set different N users spread out on the map of the considered operational scenario. U UAVs can be deployed and M charging stations (CSs) are placed in locations that are equidistant from the middle of the operational volume and each of them is placed at an equal distance from another. UAVs serve users while flying and their service performance is evaluated based on specific metrics.

UAVs are represented as *agents* defined by a state representation (including key state variables like its battery level) and an action space (which can differ based on the considered scenario). The following *assumptions* are considered:

- UAVs have a given footprint (due to their communication coverage) and each of them can detect and serve only users inside it.
- UAVs are not aware of other agents and do not explicitly communicate with each other.

Since the state representation of each UAV is not affected by the others, these assumptions guarantee the scalability of the model: scalability is guaranteed at the cost of possibly not being able to find a global optimal solution for the whole system.

Although UAVs operate in the same environment at the same time by learning a behavior independently of each other, they receive reward signals that contain also the effects of the action of the other agents. Thanks to these global reward signals, UAVs will implicitly learn to avoid obstacles and also to spread around the environment in order to maximize the performance metrics related to the services provided to the users.

4.1 Agents Model

In order to train our UAVs as learning agents, we modeled each of our agents as an independent Markov decision processes (MDP), denoted by the following tuple:

$$\langle \mathbf{S}, \mathbf{A}, \delta, r \rangle,$$

where

- $\mathbf{S}$ is the set of all the possible states;
- $\mathbf{A}$ is the set of the possible actions;
- δ represents the transition function;
- r is the reward function.

4.1.1 State Space

The state $\mathbf{s} \in \mathbf{S}$ of each agent is defined by the following tuple:

$$\mathbf{s} = < (x, y, z), b, t >,$$

where

- x, y, and z are the coordinates related to the current position of the considered agent (i.e., UAV);
- b represents the current battery level;
- t represents the current time (made up by intervals of timeslots $t_s = 5s$) associated with the current agent position and battery level.

4.1.2 Action Space

Each agent can choose its current action from a set of actions defined as follows:

$$\mathbf{A} = < left, right, forward, backward, up, down, hover, idle, charge, go\_to\_charge >.$$

More in detail, UAVs can:

- move horizontally by one cell: *left*, *right*, *forward*, and *backward*
- move vertically by one cell: *up* and *down*
- fly in place: *hover*
- be operative without flying: *idle*
- charge and go to the charging station: *charge* and *go_to_charge*. The latter is a complex action that is made up of all the actions that an agent should perform to reach the closest charging station through the A* planning algorithm.

The execution of each action leads to

- increase t by one timeslot, except for *go_to_charge* in which t is increased by as many timeslots as needed to reach the closest charging station;
- decrease of the value of the battery level b, except for the *charge* in which it is increased;
- change of the agent position (x, y, z), only for the movement actions.

4.1.3 Transition Function

The transition function δ of the MDP of each agent is not known by the agent and only implemented in the simulator. The implementation of the transition when executing the actions described above is straightforward and it is not reported in details in this article, since it is not relevant for the learning process. Q-learning algorithm is model-free and therefore the transition function is not learned.

4.1.4 Reward Function

The reward function r represents the objective function that we want to optimize. Since the problem we are dealing with is complex and challenging, this function includes several parameters that are useful to describe the considered problem. We provide a high-level explanation of the reward function to improve understanding of the optimization target. Notice also that the reward function is not known by the agent and only computed within the simulator and thus it can be modified without affecting the implementation of the learning agent.

Generally speaking, the reward function aims at optimizing the following terms:

- services requests time coverage;
- users service waiting time;
- number of served users;
- battery consumption during UAV task execution.

More specifically, the reward r is defined as a combination of four components, that is, **R1**, **WP**, **F**, and **B**. Each of these components refers to the different aspects listed above and is explained in detail in the next paragraphs.

1) R1: Composition of the service-related metrics

The metrics taken into account are as follows:

- TH: Average Connectivity Throughput

Referring to the users requesting connectivity, we want to maximize the amount of throughput averaged among the overall number of users requesting it. Thus, holds:

$$TH = \sum_{i=1}^{n_{th}} \frac{th_i}{MPR_{th}}, \tag{6}$$

where n_{th} is the number of users requesting the service, th_i is the throughput achieved for the $i - th$ user, and MPR_{th} is the maximum value of a user throughput request.

- EC: Average Edge Computing Throughput

For users requesting edge computing, we maximize the amount of throughput requested for offloading tasks averaged onto the overall number of users as follows:

$$EC = \sum_{i=1}^{n_{ec}} \frac{ec_i}{MPR_{ec}}, \tag{7}$$

where n_{ec} is the number of users requesting the Edge Computing service, ec_i is the throughput achieved for offloading the tasks of the $i - th$ user and MPR has the same meaning of **Eq. 6**, but referring to the edge computing request.

- DG: Age of Information of Data Gathering

For data gathering users, we want to minimize the age of information (AoI), thus we can maximize the freshness of information. AoI of the data produced by a user is incremented of one point for each timeslot in which the user is requesting the service without being served:

$$DG = -\sum_{i=1}^{n_{dg}} \frac{AoI_i}{MPA}, \tag{8}$$

where n_{dg} is the number of users requesting the Data Gathering service, AOI_i is the Age of Information of the user i and MPA is the maximum possible AoI of a single user.

MPR and MPA parameters are used to normalize the weight of each service in the reward in such a way that each of the listed metrics is included in the interval of values [0, 1]. When operating in a realistic setup, it is assumed to approximately know the traffic service of the considered area. Thus, it is a likely choice to perform a normalization based on the presumed traffic related to a specific service for the considered operative area.

TH, EC, and DG are finally linearly combined to obtain the first part of the overall reward:

$$R1 = u_{th} * TH + u_{ec} * EC + u_{dg} * DG, \tag{9}$$

where u_{th}, u_{ec}, and u_{dg} are coefficients that can be tuned in order to give higher priority to a specific parameter rather than another.

2) WP: User waiting time for delivery

The waiting time is computed as the sum of all the seconds in which a user was active and requesting a service that was not provided. Timely provision of drone service is a relevant UAV path planning factor since it affects both the experience of the users and service feasibility, therefore we introduce the term to penalize the waiting time for service. Our reward includes a *waiting penalty* if the average waiting time T is larger than a specific threshold called *critical time*. We apply then a *discount* on our reward if the UAVs are too late, and eventually, the following quantity will be subtracted from the reward:

$$WP = \begin{cases} 0, & \text{if } T < CT \\ \dfrac{T}{N}, & \text{otherwise} \end{cases}, \tag{10}$$

where CT is the *critical time* threshold mentioned above and T is the waiting time related to the users.

3) F: Balancing factor

This component of the reward aims to provide a service to the highest number of users, despite either the low request or the distance. This part of the reward is expressed as the difference between the sum of served users $served_i$ (over the number of users requesting in that specific moment n_r) and the previously computed *waiting penalty*:

$$F = max\left(\sum_{i=1}^{n_r} \frac{served_i}{n_r} - WP, \quad 0\right). \tag{11}$$

(4) B: Battery consumption

We want the UAV to go and charge before reaching a fail state due to exhausted battery, by using it in an efficient way without wasting energy through unnecessary actions. Thus, the component related to the battery consumption is defined as follows:

$$B_r = \begin{cases} \dfrac{battery\_level}{needed\_battery}, & \text{if } needed\_battery >= 0 \\ battery\_level, & \text{otherwise} \end{cases}, \tag{12}$$

where *needed_battery* indicates the battery level percentage needed to get to the closest charging station. All of the above components are eventually combined in the final reward function, and each of them is multiplied by a coefficient to better tune the weight of every parameter:

$$r = \alpha_c B_r + \alpha_s * (R1 + \alpha_f * F - \alpha_{wp} WP). \tag{13}$$

The coefficients α_f and α_{wp} as well as the ones used in R1 (i.e., u_{th}, u_{ec}, and u_{dg}) can be freely chosen to provide greater relevance respectively to F or to WP. Coefficients α_s and α_c depend instead on the battery level: while the battery decreases, α_c will increase and α_s will decrease, giving more importance to the battery level with respect to the service provision.

4.2 Simulation Framework

The final product of this modeling is a simulator that takes as input different parameters such as the hour related to the service requesting time, the number of users, and UAVs and allows the training of agents with a Q-learning algorithm. We provide in output an animation of the behavior of the UAVs and of the users, the quality of experience metrics registered per episode, and the Q-table. As for the input, the main parameters are as follows:

- Hour of the day: it affects the requests of the users;
- Numbers of UAVs;
- Number of users: we can set a minimum and a maximum number of users;
- Urban or Rural setting: a flag indicating if we are working in a rural or in an urban scenario and defining accordingly the height and the number of buildings in the selected area.

For finer tuning, other parameters can be modified:

- Space and Time: it is possible to choose the size of the grid in all the three dimensions and time representation;
- Battery: the basic battery consumption can be set. For hovering, horizontal and vertical actions, a different consumption rate was set. All these parameters are related to a specific variable that allows you to speed or slow down the battery consumption at will;
- User Request: the basics for users requests are here defined, that is, requests types coefficients and requests duration;
- Actions: different subsets of actions are available based on the current conditions, for example, when a UAV is charging it cannot *go_to_charge* and if it is not on a charging station, then it cannot *charge*;
- Others: more useful variables and macros are implemented to define path colors and save folders used for an easy understanding and orientation among the obtained results.

4.3 Training

Once all the parameters are set, we train UAVs as learning agents with the Q-learning algorithm. We chose to initialize the Q-table randomly with values $\in [0, 0.5]$: the *learning rate* is set to 0.9 in such a way that the initial value (randomly selected) did not affect the learning too much. The *discount factor* instead is set to 0.95 providing a relatively high discount for the future rewards and thus returning solutions preferring to get high rewards as soon as possible.

The Q-table is indexed through the agent state **s** and an action **a**. The combination of space, battery, time, and action provides a very large matrix to handle: the Q-table will have a size of $|\mathbf{S}| \times |\mathbf{A}| = 3.6 \cdot 10^8$. This results in a very sparse matrix and therefore it will not be efficient to initialize and use a matrix so large. For this reason, we initialize a row of the matrix only at the moment in which it is considered, allowing working only with the data we strictly need.

In order to ensure a greater exploration in the first episodes and maximize the reward afterward, we decided to use the *exploit/explore* parameter ϵ which is decreased at each epoch starting from a maximum of 1 up to a minimum of 0.01. ϵ indicates the probability to choose an action randomly, while $1 - \epsilon$ is the probability to choose an action according to the Q-table. The decreasing trend of ϵ is derived from the following function:

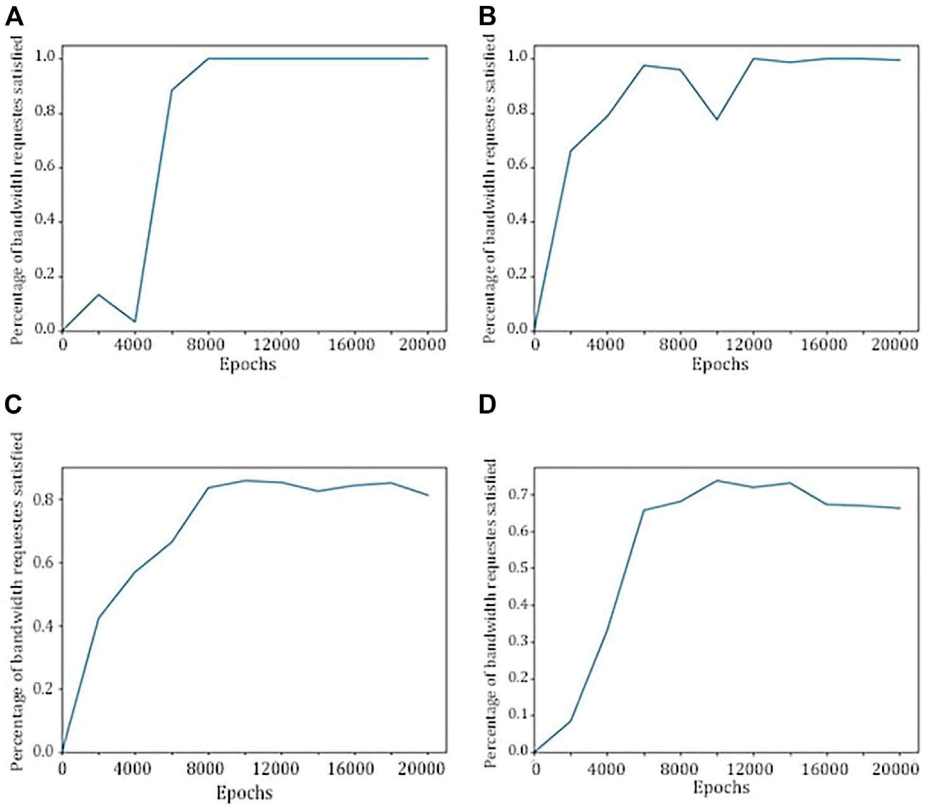

FIGURE 4 | Percentage of satisfied bandwidth requests by a single UAV with 1 user **(A)**, 5 users **(B)**, 10 users **(C)**, and 15 users **(D)**.

$$\epsilon_{t+1} = (10^2 * \epsilon_t)^{-\frac{1}{(E-P)}}, \tag{14}$$

where E represents the total number of episodes, P is the number of episodes we want ϵ to be 0.01, and t denotes the current epoch.

5 EXPERIMENTAL SETTINGS

To validate the proposed approach, we explored different scenarios varying several parameters initial hour settings, amounts and positions of users, number of charging stations, and frequency of obstacles. The settings used in the reported experimental analysis are summarized in **Table 1**.

All the tests performed can be grouped into the following three macro cases:

1) Urban environment and single UAV: the initial hour is set to 21, and a single UAV is assigned from 1 to 15 users.
2) Urban environment and multiple UAVs: at first we set the initial hour to 10, observing how 1, 2, or 3 UAVs provide the requested services coming from 3, 5, 10, or 15 users; after that, we show how a change in the initial hour setting affects the performance.
3) Rural environment and multiple UAVs: we place few users (i.e., 4) far from each other and observe how one or two UAVs handle rarer users and a greater travel time.

5.1 Metrics Employed

In order to measure the quality of experience of our AR-users, we considered a different specific metric for each type of service provided by UAVs. We took into account also the more general *delivery time* metric, which can be applied to all the sub-users of the users, regardless of the requested service. The metrics are based on the following assessment parameters:

1) *Throughput*: percentage of throughput granted with respect to the one demanded. High throughput is desirable so that users can download in a relatively short time their AR object with high quality and high resolution, shrinking the gap between the virtual stimuli and the real ones;
2) *Edge Computing*: percentage of throughput provided for edge computing over the total requested. Throughput for edge computing is a good metric because the higher the throughput the more tasks are offloaded and with greater speed;
3) *Age of Information*: number of iterations elapsed before the information is collected. If the collected information about a user becomes obsolete, then it cannot be used anymore. Therefore, the environment and the QoE of the user need to be monitored constantly: the freshness of gathered data is really relevant in AR;
4) *Delivery Time*: indicates the elapsed time between a user request and its provision. Each time a user demanding for

TABLE 2 | Results for the three considered scenarios.

N	U	M	TH metric	EC metric	AoI	TH DT	EC DT	DG DT
First scenario								
1	1	1	100%	100%	32	150 s	135 s	160 s
2	1	1	100%	93.75%	0	75 s	130 s	0 s
3	1	1	100.00%	100.00%	25	45 s	65 s	125 s
4	1	1	100.00%	100.00%	0	40 s	35 s	0 s
5	1	1	99.42%	100.00%	0	85 s	15 s	0 s
6	1	1	98.20%	85.27%	0	90 s	95 s	0 s
7	1	1	84.75%	100.00%	0	60 s	80 s	0 s
8	1	1	99.77%	92.07%	29	35 s	45 s	145 s
9	1	1	99.38%	93.07%	18	50 s	45 s	90 s
10	1	1	81.25%	81.25%	20	305 s	130 s	100 s
11	1	1	88.31%	97.52%	9	160 s	85 s	45 s
12	1	1	95.59%	92.68%	0	55 s	0 s	180 s
13	1	1	78.81%	98.95%	0	210 s	130 s	0 s
14	1	1	87.27%	85.32%	10	330 s	210 s	50 s
15	1	1	66.25%	65.69%	53	465 s	455 s	265 s
Second scenario								
5	1	1	90.15%	89.88%	0	110 s	180 s	0 s
5	2	1	98.48%	100.00%	0	25 s	70 s	0 s
5	3	1	93.03%	95.33%	0	85 s	115 s	0 s
10	1	1	72.33%	72.30%	1	405 s	440 s	5 s
10	2	1	95.22%	96.43%	0	165 s	210 s	0 s
10	3	1	99.91%	100.00%	0	80 s	110 s	0 s
15	1	1	78.96%	65.65%	49	490 s	540 s	245 s
15	2	1	100.00%	93.49%	43	70 s	150 s	215 s
15	3	1	100.00%	100.00%	18	5 s	35 s	90 s
Third scenario								
4	1	1	74.89%	62.25%	279	835 s	860 s	1,395 s
4	2	2	100.00%	100.00%	9	45 s	0 s	45 s

a service is not served, 5 s are added to our *delivery time*. This measure is really significant, mainly for *augmented reality* applications which need to be very responsive in order to avoid annoying frame delays.

6 EXPERIMENTAL RESULTS

We first discuss some general results which are common to each case. From the overall experimentation activity, we have observed that UAVs learned:

- where the users are placed and to reach them as fast as they can without any path deviation;
- to manage their battery autonomy and go and charge it when needed;
- to perceive in a proper way the environment surrounding them so as to avoid failure during the actual mission.

Thus, we can deduce that the UAVs acquired an adequate autonomy level in moving both in urban and rural environments. The learned behaviors are thus effective in the considered operational scenario and thus they can be considered representative also for other kinds of analysis (e.g., collision risk).

6.1 First Scenario: Urban Environment and Single UAV

We conducted 15 different experiments with the same settings for the hour, number of UAVs, and number of charging stations and varying only the number of AR users (see **Table 1**).

The results are illustrated through graphs showing the metrics averaged over the number of users for each epoch. We show the cases with 1, 5, 10, and 15 users. It is clear how the behavior of our UAV adjusts itself and our metrics easily converge especially for the service-related metrics. In **Figure 4**, we can look at the trend of the percentage of satisfied bandwidth requests and we can notice that as the number of users increases, performance worsens and the convergence slows down. Because of the limited bandwidth, a single drone can satisfy by itself up to 5 users, as shown by the TH and EC metrics in *First Scenario* section of **Table 2**. In the same table, it is possible to see the metrics regarding the cases with more than 5 users: it is clear that a single UAV cannot serve all the users, but it can provide great support to other sources of connection and services such as a base station or (as we will see in the next experiments) other UAVs.

6.2 Second Scenario: Urban Environment and Multiple UAVs

In this second scenario, we analyze the impact of increasing the number of UAVs to improve the quality of experience: we manage to cover at least 99% of the requests for just 5 users (as seen in the previous scenario) and proceed to try to cover up to 15. Dealing with this scenario, we can appreciate the importance

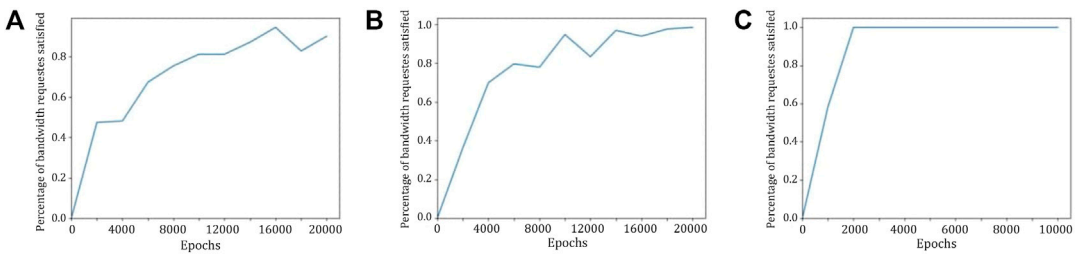

FIGURE 5 | Percentage of satisfied bandwidth requests with 1 UAV **(A)**, 2 UAVs **(B)**, and (3) UAVs in case of 10 users.

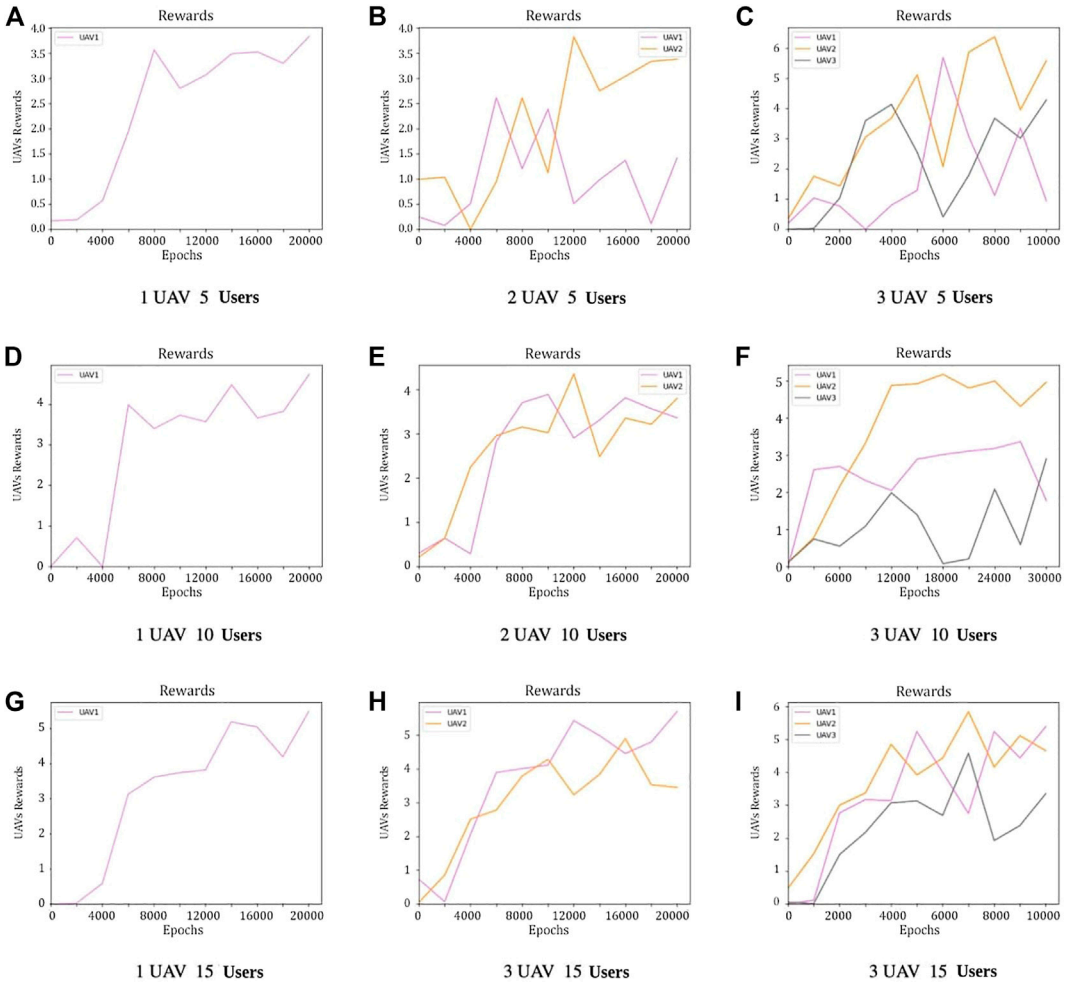

FIGURE 6 | Reward behavior for different numbers of UAVs over the epochs; 5-10-15 users are included in the scenario.

TABLE 3 | Comparative analysis results obtained for an operative scenario made up of 2 UAVs which are expected to serve 5–10-15 users. Q-learning random initialization refers to the usage of Q-learning algorithm by randomly initializing the values of the Q-table, while the greedy algorithm is the comparative baseline we used to evaluate UAVs performance.

N	TH metric	EC metric	AoI	TH DT	EC DT	DG DT
Q-learning with random initialization						
5	83.60%	52.09%	122	655 s	760 s	608 s
10	99.43%	92.76%	0	150 s	235 s	0 s
15	98.06%	95.78%	0	55 s	105 s	0 s
Greedy algorithm						
5	12.14%	40.76%	184	1,070 s	1,160 s	921 s
10	74.84%	60.58%	105	760 s	705 s	525 s
15	37.30%	46.64%	140	885 s	880 s	698 s

of scalability and modularity of UAVs, giving us the opportunity to add or remove bandwidth over the considered area. We can observe how 1, 2, or 3 UAVs handle different amounts of users (**Figure 5**).

Again in **Figure 5**, we can observe the throughput benefits deriving from the addition of more UAVs. We will also visualize in **Table 2** the resulting values of the last episode related to our second scenario simulations. Analyzing the values in this table, we can deduce that having more UAVs generally results in more user coverage. Nevertheless, this general behavior cannot always be ensured due to the initial random initialization of the Q-table (e.g., for the case with 5 users and 3 UAVs, we can observe worse results with respect to the case with the same number of users but with 2 UAVs).

In **Figure 6**, we can see how the reward of UAVs changes over the epochs and with the addition of more and more users.

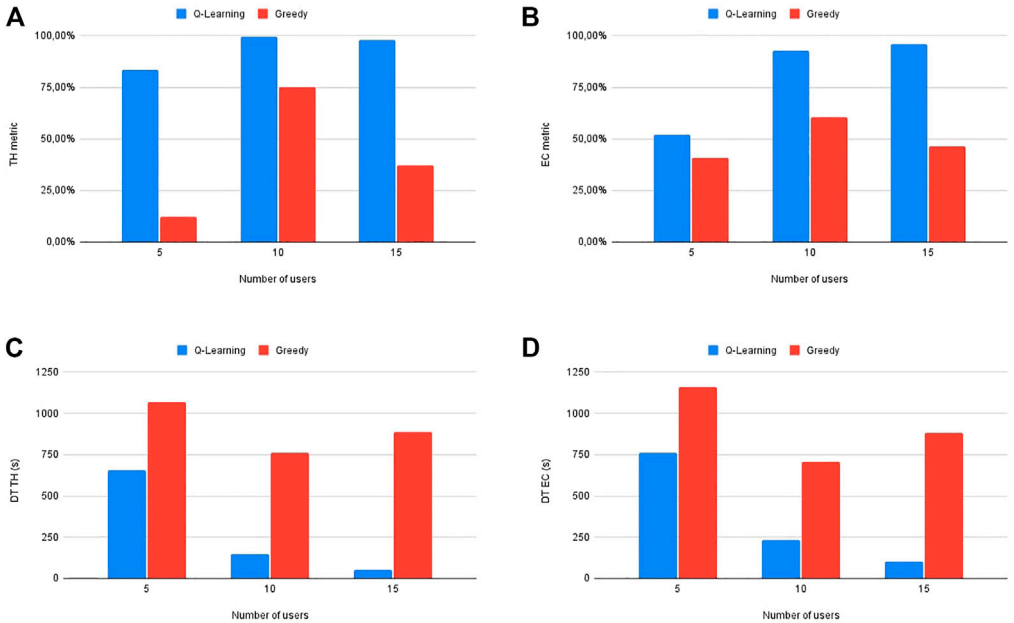

FIGURE 7 | Comparative representation (see also **Table 3**) between Q-learning and greedy algorithm performance considering **(A)** TH metric, **(B)** EC metric, **(C)** TH Delivery Time, and **(D)** EC Delivery Time.

6.3 Third Scenario: Rural Environment and Multiple UAVs

The rural environments is modeled in such a way as to have few users scattered in different areas in order to look at the UAVs behavior when large distances among users are set. The ϵ parameter is initially set to 2, ensuring a bigger exploration of the area to find all the users. Users scattered over a bigger area make the UAVs consume more energy to move from one part of the area to another: indeed here the number of charging stations has been increased with respect to previous scenarios. In **Table 2**, we can see how a single UAV struggles to satisfy all the requests on its own and how much performance can be improved by adding another UAV to help the first one in covering scattered users.

6.4 Comparative Baseline

Table 3 reports a comparison for a specific operational scenario between the main performance parameters related to the used Q-learning algorithm and a comparative greedy baseline. The considered scenarios are made up of 2 UAVs serving 5, 10, and 15 users. The used *Greedy algorithm* is based on the assumption that UAVs have a preliminary knowledge of the distribution of users in the environment; according to this knowledge, UAVs move toward the closest users with respect to their position without taking into account if users are already served or not. Looking at **Table 3** we can notice that TH, EC metrics are maximized (by minimizing their delay time) in all cases by applying *Q-learning with random initialization*. The worst results are instead the ones related to the usage of the *Greedy algorithm*. Thus, *Q-learning* always outperforms the *Greedy algorithm* with respect to any parameter shown in **Table 3**.

These results can be observed also in **Figure 7** which shows the good performance in terms of high TH and EC achieved by the *Q-learning* (subfigure (A) and (B)) as a function of the number of users as well the good results in terms of reduction of the delivery time in the case of subfigures (C) and (D).

We can also notice that by keeping the number UAVs constant, it seems that they perform better as the number of users increases in a reasonable way (with respect to the number of used drones): this can be obviously explained considering that UAVs can easily interfere with each other when serving a small number of users distributed in the same area. Finally, the reason why the AoI is equal to zero in most of the *Q-learning* cases is due to the fact that UAVs learn to place themselves in the users' area before users start to ask for a DG service.

7 CONCLUSION AND FUTURE WORK

In this article, we presented a new framework to simulate multi-UAV service providers, including a simulator and a reinforcement learning environment. In particular, we studied the task of deploying multiple UAVs to serve augmented reality users during a time window of 30 min. We presented a reward function that aims at optimizing different aspects: battery, throughput, waiting time, and age of information. We built a plausible model of requests over 24 h and also a novel model of an AR user, addressing different types of necessities. We trained UAVs within two different kinds of areas: rural and urban environments.

As shown by the experimental results, UAVs learned how to reach users faster and without failing. We studied the condition allowing UAVs to be the only internet providers with an adequate quality of experience: several scenarios including different numbers of UAVs and users have been tested. We also showed how easy is to adjust bandwidth by adding or subtracting a service drone from the

considered UAVs system, allowing to provide internet and other resources in a scalable and dynamic way.

As a further step of this work, it could be very interesting to investigate UAV's speed impact on the system performance; also fairness metrics like *Jain's fairness* could be considered to improve the performance evaluation. Future works may also include improving the simulator with additional features to increase the realism of the operational scenarios. Finally, the proposed framework can be extended to study other multi-UAV service providers and can be used for the comparative analysis of different approaches.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

FC, LI, and SC contributed to the conception and design of the proposed framework. DB and GF developed the sotware tools and perfomed the simulation analysis. All authors contributed to manuscript writing, revision, read, and approved the submitted version.

FUNDING

This paper has been partially supported by the BUBBLES project. The BUBBLES project received funding from the SESAR Joint Undertaking under the European Union's Horizon 2020 research and innovation program with grant agreement No. 893206. This research has been partially supported by the ERC Advanced Grant WhiteMech (No. 834228) and by the EU ICT-48 2020 project TAILOR (No. 952215).

REFERENCES

Bertizzolo, L., D'oro, S., Ferranti, L., Bonati, L., Demirors, E., Guan, Z., et al. (2020). "Swarmcontrol: An Automated Distributed Control Framework for Self-Optimizing Drone Networks," in IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, Toronto, Canada. [Dataset]. doi:10.1109/infocom41043.2020.9155231

Brunori, D., Colonnese, S., Cuomo, F., and Iocchi, L. (2021). "A Reinforcement Learning Environment for Multi-Service Uav-Enabled Wireless Systems," in PerVehicle 2021, International Workshop on Pervasive Computing for Vehicular Systems in Conjunction with IEEE PerCom 2021, Kassel, Germany, March 22–26, 2021. doi:10.1109/percomworkshops51409.2021.9431048

Chakareski, J., Naqvi, S., Mastronarde, N., Xu, J., Afghah, F., and Razi, A. (2019). An Energy Efficient Framework for Uav-Assisted Millimeter Wave 5g Heterogeneous Cellular Networks. *IEEE Trans. Green. Commun. Netw.* 3, 37–44. doi:10.1109/TGCN.2019.2892141

Chakareski, J. (2019). Uav-iot for Next Generation Virtual Reality. *IEEE Trans. Image Process.* 28, 5977–5990. doi:10.1109/tip.2019.2921869

Chakareski, J. (2020). Viewport-Adaptive Scalable Multi-User Virtual Reality Mobile-Edge Streaming. *IEEE Trans. Image Process.* 29, 6330–6342. doi:10.1109/tip.2020.2986547

Cheng, Y. (2020). Edge Caching and Computing in 5g for Mobile Augmented Reality and Haptic Internet. *Comput. Commun.* 158, 24–31. doi:10.1016/j.comcom.2020.04.054

Chiaraviglio, L., D'andreagiovanni, F., Choo, R., Cuomo, F., and Colonnese, S. (2019). Joint Optimization of Area Throughput and Grid-Connected Microgeneration in Uav-Based Mobile Networks. *IEEE Access* 7, 69545–69558. doi:10.1109/access.2019.2920065

Colonnese, S., Cuomo, F., Pagliari, G., and Chiaraviglio, L. (2019). Q-Square: A Q-Learning Approach to Provide a Qoe Aware Uav Flight Path in Cellular Networks. *Ad Hoc Networks* 91, 101872. doi:10.1016/j.adhoc.2019.101872

Columbia University Information Technology (n.a.). CUIT Internet Usage Graphs. [Dataset] Available at: http://www.columbia.edu/acis/networks/bandwidth.html.

Ferranti, L., Bonati, L., D'Oro, S., and Melodia, T. (2020). "Skycell: A Prototyping Platform for 5g Aerial Base Stations," in 2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks", August 31 - September 03, 2020 (WoWMoM), 329–334. doi:10.1109/WoWMoM49955.2020.00062

Fujihashi, T., Koike-Akino, T., Watanabe, T., and Orlik, P. V. (2019). "Holocast: Graph Signal Processing for Graceful Point Cloud Delivery," in ICC 2019-2019 IEEE International Conference on Communications (ICC), Shanghai, China, May 20–24, 2019 (IEEE), 1–7. doi:10.1109/icc.2019.8761819

Graham-Cumming, J. (2021). COVID-19 Impacts on Internet Traffic: Seattle. Northern Italy and South Korea. [Dataset] Available at: https://blog.cloudflare.com/covid-19-impacts-on-internet-traffic-seattle-italy-and-south-korea/.

Han, B., Liu, Y., and Qian, F. (2020). "Vivo: Visibility-Aware Mobile Volumetric Video Streaming," in Proceedings of the 26th Annual International Conference on Mobile Computing and Networking, London, United Kingdom, September 21–25, 2020, 1–13.

Hosseini, M., and Timmerer, C. (2018). "Dynamic Adaptive Point Cloud Streaming," in Proceedings of the 23rd Packet Video Workshop, Amsterdam, Netherlands, June 12, 2018, 25–30. doi:10.1145/3210424.3210429

ISO/IEC JTC-1 (2016). Text of 2nd CD Mixed and Augmented Reality (MAR) Reference Model. Tech. rep.

Jang, E. S., Preda, M., Mammou, K., Tourapis, A. M., Kim, J., Graziosi, D. B., et al. (2019). Video-Based Point-Cloud-Compression Standard in Mpeg: from Evidence Collection to Committee Draft [Standards in a Nutshell]. *IEEE Signal. Process. Mag.* 36, 118–123. doi:10.1109/msp.2019.2900721

Jiang, B., Yang, J., Xu, H., Song, H., and Zheng, G. (2019). Multimedia Data Throughput Maximization in Internet-Of-Things System Based on Optimization of Cache-Enabled Uav. *IEEE Internet Things J.* 6, 3525–3532. doi:10.1109/jiot.2018.2886964

Lafruit, G., Bonatto, D., Tulvan, C., Preda, M., and Yu, L. (2019). Understanding Mpeg-I Coding Standardization in Immersive Vr/ar Applications. *SMPTE Mot. Imag. J.* 128, 33–39. doi:10.5594/jmi.2019.2941362

Lai, Z., Hu, Y. C., Cui, Y., Sun, L., Dai, N., and Lee, H.-S. (2019). Furion: Engineering High-Quality Immersive Virtual Reality on Today's Mobile Devices. *IEEE Trans. Mobile Comput.* 19, 1586–1602. doi:10.1109/TMC.2019.2913364

Liu, C. H., Dai, Z., Zhao, Y., Crowcroft, J., Wu, D., and Leung, K. K. (2021). Distributed and Energy-Efficient mobile Crowdsensing with Charging Stations by Deep Reinforcement Learning. *IEEE Trans. Mobile Comput.* 20, 130–146. doi:10.1109/TMC.2019.2938509

Liu, Z., Li, J., Chen, X., Wu, C., Ishihara, S., Ji, Y., et al. (2020). Fuzzy Logic-Based Adaptive point Cloud Video Streaming. *IEEE Open J. Comput. Soc.* 1, 121–130. doi:10.1109/ojcs.2020.3006205

Mangiante, S., Klas, G., Navon, A., GuanHua, Z., Ran, J., and Silva, M. D. (2017). "Vr Is on the Edge: How to Deliver 360 Videos in Mobile Networks," in Proceedings of the Workshop on Virtual Reality and Augmented Reality Network, Los Angeles, CA, August 2017, 30–35.

Milani, S., Polo, E., and Limuti, S. (2020). A Transform Coding Strategy for Dynamic Point Clouds. *IEEE Trans. Image Process.* 29, 8213–8225. doi:10.1109/tip.2020.3011811

Palasamudram, D. S., Sitaraman, R. K., Urgaonkar, B., and Urgaonkar, R. (2012). "Using Batteries to Reduce the Power Costs of Internet-Scale Distributed Networks," in SoCC '12 Proceedings of the Third ACM Symposium on Cloud Computing (New York, NY, USA: Association for Computing Machinery). doi:10.1145/2391229.2391240

Pan, Y., Gao, B., Mei, J., Geng, S., Li, C., and Zhao, H. (2020). "Semanticposs: A Point Cloud Dataset with Large Quantity of Dynamic Instances," in 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, November 13, 2020 (IEEE), 687–693. doi:10.1109/iv47402.2020.9304596

Peill-Moelter, N. (2012). Batteries Included: A Learner and Greener Internet Using Smart Batteries. [Dataset] Available at: https://blogs.akamai.com/2012/10/batteries-included-a-leaner-and-greener-internet-using-smart-batteries.html (Accessed 2021).

POSS (2020). Semanticposs Dataset. Available online at http://www.poss.pku.edu.cn/semanticposs.html.

Santos, I., Henriques, R., Mariano, G., and Santos, E. (2021). "UAV's Multimedia Technology and Augmented Reality (Geointegration): New Concept and New Paradigm of Geodiversity Presentation," in *Global Geographical Heritage, Geoparks and Geotourism* (Singapore: Springer), 59–74. doi:10.1007/978-981-15-4956-4_4

Sheikhipour, N., Pesonen, M., Schwarz, S., and Vadakital, V. K. M. (2019). "Improved Single-Layer Coding of Volumetric Data," in 2019 8th European Workshop on Visual Information Processing (EUVIP), Rome, Italy, October 28–31, 2019 (IEEE), 152–157. doi:10.1109/euvip47703.2019.8946131

Taha, B., Hayat, M., Berretti, S., and Werghi, N. (2020). "Fused Geometry Augmented Images for Analyzing Textured Mesh," in 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, October 25–28, 2020 (IEEE), 2651–2655. doi:10.1109/icip40778.2020.9191099

Tan, Z., Qu, H., Zhao, J., Zhou, S., and Wang, W. (2020). Uav-Aided Edge/Fog Computing in Smart Iot Community for Social Augmented Reality. *IEEE Internet Things J.* 7, 4872–4884. doi:10.1109/jiot.2020.2971325

Wang, F., Xu, J., Wang, X., and Cui, S. (2017). *Joint Offloading and Computing Optimization in Wireless Powered mobile-edge Computing Systems*. [Dataset].

Wang, Z., Duan, L., and Zhang, R. (2019). Adaptive Deployment for Uav-Aided Communication Networks. *IEEE Trans. Wireless Commun.* 18, 4531–4543. doi:10.1109/TWC.2019.2926279

Xu, J., Zeng, Y., and Zhang, R. (2017). *Uav-Enabled Wireless Power Transfer: Trajectory Design and Energy Optimization*. [Dataset].

Zeng, Y., Wu, Q., and Zhang, R. (2019). *Accessing from the Sky: A Tutorial on Uav Communications for 5g and Beyond*. [Dataset].

Zhang, A., Wang, C., Liu, X., Han, B., and Qian, F. (2020). "Mobile Volumetric Video Streaming Enhanced by Super Resolution," in Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services, Toronto, Canada, June 15–19, 2020, 462–463. doi:10.1145/3386901.3396598

Zhang, X., Zhong, Y., Liu, P., Zhou, F., and Wang, Y. (2019). Resource Allocation for a Uav-Enabled mobile-Edge Computing System: Computation Efficiency Maximization. *IEEE Access* 7, 113345–113354. doi:10.1109/ACCESS.2019.2935217

Zhou, C., He, H., Yang, P., Lyu, F., Wu, W., Cheng, N., et al. (2019). "Deep Rl-Based Trajectory Planning for Aoi Minimization in Uav-Assisted Iot," in 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xi'an, China, October 23–25, 2019, 1–6. doi:10.1109/wcsp.2019.8928091

Zhou, F., Wu, Y., Sun, H., and Chu, Z. (2018). "Uav-Enabled Mobile Edge Computing: Offloading Optimization and Trajectory Design," in 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, May 20-24, 2018, 1–6. doi:10.1109/ICC.2018.8422277

Zouaoui, H., Faricelli, S., Cuomo, F., Colonnese, S., and Chiaraviglio, L. (2019). "Energy and Quality Aware Multi-Uav Flight Path Design Through Q-Learning Algorithms," in *Wired/Wireless Internet Communications*. Editors M. Di Felice, E. Natalizio, R. Bruno, and A. Kassler (Cham: Springer International Publishing), 246–257. doi:10.1007/978-3-030-30523-9_20

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.