Differentiated QoS DDPG-based Slicing and Drone Positioning for Next Generation Networks

Ghoshana Bista, Abbas Bradai, Emmanuel Moulay

Differentiated QoS DDPG-based Slicing and Drone Positioning for Next Generation Networks

Ghoshana Bista
XLIM UMR CNRS 7252
Université de Poitiers
Poitiers, France
ghoshana.bista@univ-poitiers.fr

Abbas Bradai
LEAT UMR CNRS 7248
Université Côte d'Azur
Nice, France
abbas.bradai@univ-cotedazur.fr

Emmanuel Moulay
XLIM UMR CNRS 7252
Université de Poitiers
Poitiers, France
emmanuel.moulay@univ-poitiers.fr

***Abstract*—In the imminent era of 6G, Unmanned Aerial Vehicles (UAVs) emerge as indispensable connectors, seamlessly integrating ground and space networks with unparalleled flexibility and dynamic mobility. This article focuses on enhancing Quality of Service (QoS) by categorizing users into Premium, Silver, and Bronze tiers, akin to 5G slicing. Leveraging Deep Deterministic Policy Gradient (DDPG), the study unfolds in two phases. In the first phase, UAVs are strategically deployed using DDPG to optimize network coverage in areas of higher user density. This intelligent deployment adapts to user distribution patterns. In the second phase, dynamic repositioning using DDPG meets QoS requirements by prioritizing users based on categories. This two-phase approach showcases UAV adaptability in optimizing wireless communication systems in the evolving landscape of 6G networks. The proposed solution, driven by DDPG, ensures optimal coverage and responsiveness to diverse user needs.**

***Index Terms*—Deep reinforcement learning, Drone, Slicing, Quality Of Service**

I. Introduction

In the dynamic landscape of technological evolution over the past decade, unmanned aerial vehicles (UAVs), colloquially referred to as drones or unmanned aircraft systems (UAS), have emerged as technological frontiers, propelled by substantial advancements in machine learning (ML) and artificial intelligence (AI). This transformative journey has unfolded a spectrum of applications, spanning the realms of delivery services, disaster response, agriculture, and military operations. In times of natural disasters, where stationary infrastructure may be damaged, drones play a vital role as flying base stations (BSs). For instance, in the aftermath of events like earthquakes, when traditional base stations are compromised, UAVs can quickly re-establish communication [1]. Furthermore, integrating drones with wireless networks has garnered attention for its flexibility and mobility, making it appealing for commercial and academic purposes.

UAVs offer a versatile platform for wireless communication, addressing challenges and providing unique advantages. Their mobility allows for rapid deployment in emergency scenarios, and they can establish direct line-of-sight communication links during challenging circumstances. In the communication sphere, drones act as relay nodes, extending communication range, forming mesh networks for ad-hoc connections, and even serving as aerial base stations in cellular networks, often referred to as flying radio access networks (FRAN) [2]. Research efforts focus on optimizing UAV flight paths, known as trajectory optimization, to meet user demands. Energy resource optimization, communication protocol algorithms, and ensuring secure and private communication for UAVs are areas of active investigation. Amid the advantages, challenges exist in implementing UAVs to meet diverse user and situational demands. This article specifically delves into the challenges of autonomous trajectory optimization, addressing them using DRL (Deep Reinforcement Learning). DRL, a subset of machine learning, combines reinforcement learning with deep learning, proving effective in decision-making and sequential interactions [3]. Autonomous trajectory optimization uses the DDPG (Deep Deterministic Policy Gradient) algorithm, which excels in handling continuous action spaces [4]. DDPG employs a critic network to estimate expected cumulative rewards and utilizes experience replay to enhance training stability and efficiency. With demonstrated success, DDPG finds practical applications, including drone implementation for wireless network provision, such as in current 5G networks. Also, in the context of future technologies like 6G, UAVs are expected to play a crucial role, densely populating the aerial space and serving as an intermediary network layer connecting ground and space-based networks [5]. This article presents a simulation that optimizes drone trajectories in two phases. Initially, drones are deployed in a grid, using DRL to prioritize areas with higher user density based on user categories (Premium (A) or Gold, Silver (B), Bronze (C)). Subsequently, the drone's height and movement are adjusted with DRL to meet QoS demands for each user category.

The paper is organized as follows: After a literature review in Section II, the main DDPG algorithms for drones are presented in Section III, and simulation results are provided in Section IV. Finally, a conclusion is addressed in Section V.

II. Literature Review

Numerous studies have investigated the implementation of drones, particularly their applications and challenges in wireless communication [6], [7]. One systematic literature

This work was supported by CHIST-ERA under Grant SAMBAS CHIST-ERA-20-SICT-003, in part by FWO, in part by ANR, in part by NKFIH, and in part by UKRI.

review emphasized the significance of the delay factor in meeting network requirements for UAV-based Internet of Vehicles (IoV) [8]. Another survey delved into UAV-assisted wireless communications, exploring recent advancements and future trends in the field [9]. Additionally, a comprehensive survey on communication and networking technologies for UAVs provided insights into lessons learned, challenges, open issues, and future directions in UAV communications. Various survey papers have also addressed the application of drones in wireless networks, focusing on resource allocation and management in 5G and beyond, as highlighted in works such as [10] and [11].

The Internet of Drones (IoD) has recently surfaced as a promising technology to enhance traditional ground-based wireless networks' coverage, connectivity, and reliability, as highlighted in [10]. IoD is increasingly being integrated into various networks, including cellular networks, wireless sensor networks (WSNs), IoT networks, and reconfigured intelligent surface (RIS)-aided networks [12].

Conventional techniques, such as optimization- and game theory-based approaches, face challenges in addressing communication issues in dynamic drone environments due to their high complexity, as noted in [11] and [13]. As a result, emerging machine learning approaches like DRL are gaining prominence for addressing wireless network challenges.

Drones, with their reduced maintenance costs, seamless integration with various systems, and high mobility, find applications in civil and military domains [10] and [14]. They prove particularly effective in wireless coverage and search and rescue missions and are expected to play a significant role in future cellular networks, enhancing capacity and coverage. Additionally, drones minimize communication delays and improve user throughput by providing aerial caching at small base stations [15].

The diverse applications of IoD networks have led to developing specialized drone systems, such as IoT-enabled drones [16] and LiDAR-based drones [17]. IoD proves invaluable in real-time urban traffic monitoring and management to support public transportation, as discussed in [18]. Drones find versatile applications in reconnaissance, including traffic surveillance, indoor and outdoor monitoring, and environmental surveillance [18]. In scenarios like environmental disasters, a swarm of drones can be deployed for efficient search and rescue operations. The mobility of drones within a network enables them to establish a reliable line of sight (LoS) connection with ground entities, significantly enhancing coverage and network performance compared to conventional ground-based wireless networks. Leveraging drones as aerial base stations offer various strategies for distributing and offloading user traffic, making them effective integrators with cellular and vehicular networks. This integration not only improves overall network performance but also helps alleviate congestion [5]. We introduce innovative enhancements to the Deep Deterministic Policy Gradient (DDPG) framework by integrating differentiated Quality of Service (QoS) management. Our approach promises tailored, efficient service delivery by aligning system responses with user priorities, marking a significant leap in optimizing network performance and user satisfaction.

III. Proposed Methodology

A. Reinforcement learning

Reinforcement learning (RL) facilitates the acquisition of an optimal decision-making policy through sequential learning and trial-and-error processes. The formalization of RL is based on the framework of Markov Decision Processes (MDP), comprising a set of states (S), a set of actions (A), and transition dynamics denoted as T($s_{t+1}|\ s_t$,a_t). At each time step (t), the agent (drone) selects an action (a_t), determined by observations (in this context QoS) represented by the current state (s_t). Following the drone's movement, the environment responds with a reward (r_{t+1}) and transitions to a new state (s_{t+1}). The drone aims to develop an optimal strategy or policy through repeated experiments and adjustments of its position, maximizing the cumulative rewards over time. This learning process involves utilizing action-value functions, commonly known as Q-values. These Q-values express the expected cumulative rewards r associated with taking a specific action A in a given state S following the rule as in Equation 1.

$$Q(s,a) = E^{\pi}\left[r_{t+1}|S_t = s, A_t = a\right] \tag{1}$$

forming the basis for the drone's decision-making and adaptation to achieve optimal performance.

Reinforcement learning (RL) faces challenges in efficiently finding optimal strategies, especially in large-scale wireless communication scenarios. Experts have introduced Deep Reinforcement Learning (DRL) to address this limitation, combining RL with deep learning techniques. This integration aims to enhance performance in complex environments. A policy-based algorithm called Deep Deterministic Policy Gradient (DDPG) is commonly employed for scenarios with dynamics and continuous action spaces, such as drone movements. DDPG combines features of both policy gradient and Q-learning techniques. In this algorithm, an actor, represented by a deep neural network, selects actions based on the current state of the environment. Additionally, a critic, implemented as a Q-value deep neural network, assesses the quality of actions taken by the actor-network. This collaborative approach in DDPG contributes to more effective decision-making and adaptation in dynamic and continuous action spaces like those encountered in drone movements within wireless communication networks.

B. DDPG based algorithms

We address real-world scenarios, particularly focusing on the critical role of network services during natural disasters for effective rescue operations. In such emergencies, network services are vital for various entities, including firefighters and medical facilities, where the QoS is of utmost importance and cannot be compromised. To cater to these requirements, we incorporate deep reinforcement learning into the configuration of network slicing. The simulation unfolds in two stages using two DDPG, illustrated in Figure 1. Initially, we examine

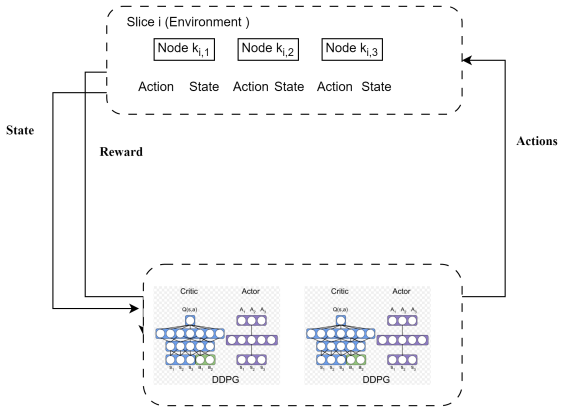

Fig. 1: Multiple DDPG for different categories of users (Nodes)

the optimal drone deployment locations in a grid scenario, assigning one drone to each cell. Considering user equipment (UE) density, we categorize users into three groups: Category A (Premium or Gold), Category B (Silver), and Category C (Bronze). The UE density in each cell is dynamically generated, and the deployment prioritizes cells with a higher density of Category A users. Using DDPG, we deploy three drones in the experiment, selecting the top three cells with the highest user density. Let C' be the set of cells where drones can be deployed, and $c \in C'$ represents a specific cell. The dynamic UE density in each cell c at time t is denoted as $D(c,t)$, encompassing users from the three categories (A, B, C). The system's state space S at time t is defined by Equation 2.

$$S(t) = \{D_A(c,t) \mid c \in C'\} \tag{2}$$

where D_A is the UE density in each cell specifically for category A users.

The action space consists of the decision to deploy a drone in each cell. Let $A(c,t)$ be the action representing whether a drone is deployed in cell c at time t. The action space is defined as in Equation 3.

$$A(t) = \{A(c,t) \mid c \in C'\}. \tag{3}$$

The priority of each cell is given by Equation 4.

$$P(c,t) = D_A(c,t) \tag{4}$$

is determined based on the UE density of category A users. The DDPG algorithm involves training a policy network $\pi(S)$ that outputs actions based on the current state S. In this case, the policy network is given by Equation 5.

$$\pi(S(t)) = \{A(c,t) \mid c \in C'\} \tag{5}$$

At each time step, the agent deploys drones based on the policy, selecting the top three cells with the highest category A density as in Equation 6.

$$TopThree(A(t) = TopThree(\pi(S(t))) \tag{6}$$

where $TopThree(\cdot)$ selects the top three elements with the highest value. The reward for each drone is calculated based on the UE density at its current position as in Equation 7.

$$Reward_i = UE_{DN}[Dr_{i,x}gd_x, Dr_{i,y}gd_y] \tag{7}$$

where UE_{DN} is the array representing UE density across the grid, $Dr_{x,y}$ is the x-coordinate of drone i, $Dr_{i,y}$ is the y-coordinate for the drone i, gd_y is the size of the grid along y-axis and gd_x is the size of the grid along the x-axis. The total reward is calculated as the sum of all individual rewards obtained throughout the episode.

Algorithm 1 defines a simulation framework for drone movements in a 3D environment. The DroneEnvironment class initializes the environment, including drones, obstacles, and parameters, and provides methods for calculating density, moving drones, and updating environmental conditions. The DDPGAgent class manages the agent's behavior, including defining neural network models, optimizers, and training procedures.

Algorithm 1 includes a visualization function that plots drone trajectories and highlights positions with the highest User Equipment (UE) density for each episode. In the main section, the environment and agent instances are created, and the visualization function is called for a specified number of episodes. This algorithm is a foundation for studying drone behaviors in 3D space using a DDPG approach.

Algorithm 1 Drone Environment & DDPG Agent

1: **1. Initialize:** Setup environment, drones, and DDPG agent
2: **for** each episode **do**
3: **2. Reset:** Initialize environment and variables
4: **for** each time step **do**
5: **3. Interact:** Select actions, move drones, calculate rewards, and train agent
6: **4. Record:** Store drone positions and find density peaks
7: **end for**
8: **end for**
9: **5. Visualize:** Display drone trajectories, density peaks, and total reward

After deploying the drones in a cell, we now focus on the second part of our simulation, which is how we provide QoS to the different user categories depending upon the priority of users. Algorithm 2 is the pseudo algorithm for how we carry out our operation. Algorithm 2 outlines a comprehensive framework for training a drone system through DRL, employing the DDPG approach. The algorithm begins by initializing the environment, defining action and observation spaces, and setting up initial QoS criteria for diverse user categories. It dynamically updates QoS criteria based on environmental conditions. The agent is then initialized, featuring actor and critic neural networks, and a replay buffer is established for storing training experiences. The main training loop iterates through episodes and time steps, involving the selection of actions, drone movement, reward calculation, and training of

the DDPG agent. Visualization functions are incorporated to depict drone trajectories and total rewards. The algorithm also addresses reward and observation calculations, environment updates, and the resetting of the environment at the beginning of each episode. Training data is efficiently stored in the replay buffer, providing a robust foundation for the reinforcement learning-based training of a drone system in dynamic and varying conditions.

Algorithm 2 DDPG Algorithm for Drone System

Initialization:
Initialize environment, agent, and replay buffer.
Training Loop:
for each episode **do**
    Reset environment and variables.
    for each time step **do**
        Interact with the environment, train agents, and record positions.
    end for
end for
Visualization & Rewards:
Implement visualization for trajectories and rewards.
Observation & Environment Updates:
Implement observation generation and update environment.
Reset & Training Data:
Implement environment reset and store data in replay buffer.

Algorithm 3 is a continuation of Algorithm 2, which outlines the training process for a drone system using the DDPG approach. It begins by initializing the DDPG agent with hyperparameters such as state and action dimensions, learning rates for the actor and critic, gamma, epsilon decay, and action space specifications. The training loop executes over multiple episodes, involving resetting the environment and dynamic updates. At each time step, the actor-network selects actions, incorporating exploration noise, and the subsequent execution of these actions leads to the observation of the next state and associated rewards. Individual latency, throughput, and reliability rewards are calculated and stored, contributing to the overall episode reward for all users. The algorithm employs a replay buffer to sample minibatches for training, preprocesses relevant data, and computes current QoS-based rewards. Critic and actor networks are updated using computed gradients, and a soft update is applied to target networks. The algorithm concludes with storing total rewards per episode and plotting various metrics, including latency, throughput, reliability, total rewards, and QoS criteria across episodes.

1) Reward function: This method computes the reward for a given state and action, incorporating QoS rewards based on the user category. The distance to the target is calculated using the Euclidean norm, and a negative reward is assigned proportional to this distance. The method then determines the user category index in a 1D array based on the drone's position coordinates. The received power is computed using a channel

Algorithm 3 DDPG Algorithm for Drone System (Continued)

Initialize DDPG Agent:
Set hyperparameters, build actor and critic networks.
Training Loop:
for each episode **do**
    Reset environment, initialize episode and QoS rewards.
    for each time step **do**
        Interact with environment, calculate individual rewards, and update networks.
        Store rewards, update episode reward, and sample minibatch for training.
        Update critic and actor networks, perform soft target updates.
    end for
    Store total reward and update QoS history.
end for

model, and the user category is extracted from the received power values. The code calculates QoS rewards for three user categories (A, B, and C) based on the received power and specified QoS criteria. The weights for each category's reward are adjusted based on their importance. The total reward combines QoS rewards for Categories A, B, and C. The resulting rewards are returned as a list.

The rewards for Category A are based on specified QoS criteria, including latency, throughput, reliability, and distance. This reward function takes as input the current drone positions (current positions), user positions (user positions), QoS criteria for Category A (QoS criteria), and the received power as in given Equation 8.

$$Reward_A = \sigma \left(\sum_{i=1}^{N_1} \gamma_{i_A} + \sum_{i=1}^{N_1} \alpha_{i_A} + \sum_{i=1}^{N_1} \theta_{i_A} \right) \quad (8)$$

where σ is a weight assign to the category A, γ_i the latency rewards, α_i the throughput rewards, θ_i the Signal-to-Interference-plus-Noise Ratio (SINR) rewards of the category A and N_1 the number of users in Category A, Gold.

The rewards for Category B are based on various QoS criteria, including latency, throughput, reliability, and distance. This reward function takes as input the current drone positions (current positions), user positions (user positions), QoS criteria for Category B (QoS criteria), and the received power. It is given as in Equation 9.

$$Reward_B = \beta \left(\sum_{i=1}^{N_2} \gamma_{i_B} + \sum_{i=1}^{N_2} \alpha_{i_B} + \sum_{i=1}^{N_2} \theta_{i_B} \right) \quad (9)$$

where β is the weight assign to the category B, γ_i the latency rewards, α_i the throughput rewards, θ_i the SINR rewards of the category B and N_2 the number of users in Category B, Sliver. It then calculates QoS-based rewards for latency, throughput, and reliability. The rewards are adjusted based on the received power, aiming to balance the impact of the channel conditions on different QoS metrics.

The rewards for Category C are based on specified QoS criteria, including latency, throughput, reliability, and distance. The function takes as input the current drone positions (current positions), user positions (user positions), QoS criteria for Category C (QoS criteria), and the received power. It is given by Equation 10.

$$Reward_C = \omega \left(\sum_{i=1}^{N_3} \gamma_{i_C} + \sum_{i=1}^{N_3} \alpha_{i_C} + \sum_{i=1}^{N_3} \theta_{i_C} \right) \quad (10)$$

where ω is weight assign to the category C, γ_i the latency rewards, α_i the throughput rewards, θ_i the SINR rewards of the category C and N_3 the number of users in Category C, Bronze.

2) Latency: The latency reward is based on the user position, current position, and target latency. Here is the mathematical representation in Equation 11.

$$Latency = \frac{\mid C_p - U_p \mid}{S_l} \quad (11)$$

where C_p is current drone position, U_p is the user position, and S_l is speed of light. The formula calculates latency as the ratio of the spatial separation between the drone and the user to the speed of light. This ratio indicates the time it takes for a signal to travel from the drone to the user or vice versa. Lower latency values indicate quicker communication between the drone and the user, while higher values imply longer communication delays.

3) SINR: The function below calculates the SINR, which is the ratio of the signal power to the sum of interference and noise powers, shown in Equation 12.

$$SINR_{dB} = \frac{S_p}{I_p + N_P} \quad (12)$$

where S_p is the single power, I_p the interference power and N_p the noise power. The SINR is scaled based on the target SINR, and the resulting value represents the SINR reward. The function is designed to emphasize the importance of meeting SINR targets.

4) Throughput: The function below calculates throughput in Mbps based on a given data rate and bandwidth and assigns a reward accordingly, as in Equation 13.

$$Th = B \times (1 + SINR) \quad (13)$$

where Th is throughput, SINR is the Signal-to-Interference-plus-Noise Ratio and B is bandwidth.

5) Drone position: Drone postion are ajusted with the feedback provided by system, after the deep lering is implemented so that it can meet the desire objective. Height of the dron is adjusted as in equation.

$$H_{new} = clip(H_{current} + \delta H, H_{min}, H_{max}) \quad (14)$$

$H_{new}$is the drone's new height after applying action. $H_{current}$ is the drone's current height before the action is applied δH is the change in height as dictated by the action. H_{min} and H_{max}represent the minimum and maximum allowable heights for the drone, ensuring it remains within a predefined vertical space. These values prevent the drone from going below ground level or exceeding a certain altitude limit.

The clip function ensures that the new height new H_{new} does not exceed the minmum and maximum height boundaries. It resticts the value within the range $[H_{min}, H_{max}]$

IV. Simulation results

In our study, we conducted simulations using Python 3.1 within the Google Colab environment, leveraging the power of TensorFlow for DRL. The experiment was bifurcated into two distinct phases to comprehensively evaluate the performance of the deployed drones.

Phase 1: Deployment using DDPG. The initial simulation phase focused on deploying drones in suitable areas, guided by the DDPG algorithm. Suitability was determined by identifying locations with a higher density of users. The primary goal was strategically positioning drones in areas with elevated user density.

Phase 2: Dynamic Drone Adjustments for QoS. The subsequent part of our experiment delved into dynamic adjustments to the drones' movements. This encompassed variations in height and back-and-forth positional changes, allowing for meticulous control. The objective was to optimize Quality of Service (QoS) specifically for prioritized users.

Our evaluation criteria spanned both phases, assessing the overall efficacy of the deployed drones. We gauged the success of the deployment strategy in identifying high-density user areas during Phase 1. In Phase 2, we evaluated the precision of dynamic adjustments to enhance QoS for priority users. This concise approach captures the essence of our experimental design and evaluation criteria. This structured approach ensured a thorough examination of our experiment, shedding light on the effectiveness of the DRL-based methodology in drone deployment and dynamic adjustments for optimized service delivery. We simulated the $10\times10\times10$ (X, Y, and Z) grid. Initially, three drones were deployed, each positioned in the top three cells with the highest user density on the grid. Using the following parameters as shown in Table I and Table II. Table I includes parameters related to the simulation setup or the environment, while Table II contains hyperparameters specifically used in the implementation of the Deep Deterministic Policy Gradients (DDPG) algorithm. This meticulous setup ensured a detailed exploration of drone deployment strategies within a three-dimensional space, guided by specific hyperparameters for the DDPG algorithm.

In Figure 2, the rewards for the initial phase of up to 1000 episodes are illustrated, showcasing consistently positive and promising outcomes. This measure demonstrates the overall success and effectiveness of the algorithm over the specified period. The visual representation affirms the effectiveness of our DDPG algorithm, demonstrating its proficiency in selecting higher-density cells within the grid. Figures 3 and 4 present density maps depicting the distribution of different user categories across the grid. A specific episode (episode 81) was randomly selected from the thousand episodes to offer a

TABLE I: Initial parameters for first part

Parameter	Description	Default Value
Grid_size	Size of the 3D grid representing the environment	(10, 10, 10)
Num_drones	Number of drones in the environment	3
Sensor_range	Sensor range of each drone in the environment	3 unit, means a drone can sense objects up to 3 grid cells away
Num_users	Number of user for A,B and C	rand(1,10)

TABLE II: DDPG Hyperparameters for first part

Hyperparameter	Value
State Dimension	$10 \times 10 \times 10$
Action Dimension	2
Action High	1.0
Actor Learning Rate	0.001
Critic Learning Rate	0.002
Actor Hidden Layer 1	64 neurons, ReLU activation
Actor Hidden Layer 2	64 neurons, ReLU activation
Actor Output Layer	Tanh activation, Random Uniform initialization $(-0.003, 0.003)$
Critic Hidden Layer 1	64 neurons, ReLU activation
Critic Hidden Layer 2	64 neurons, ReLU activation
Critic Output Layer	Linear activation
Discount Factor (γ)	0.99
Single power(S_p)	2500 dbm
Noise power(N_p)	10 db
Bandwidth(N_p)	1e7Hz

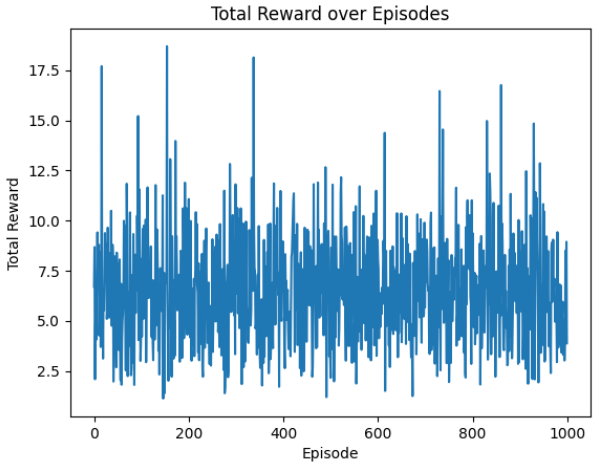

Fig. 2: Total reward per Episodes

more detailed insight, revealing the density distribution before and after the episode's conclusion.

In Figure 5, we visually represent drone placements within individual cells. Each dot is color-coded, with blue, yellow, and green signifying different drones. Notably, the red cross

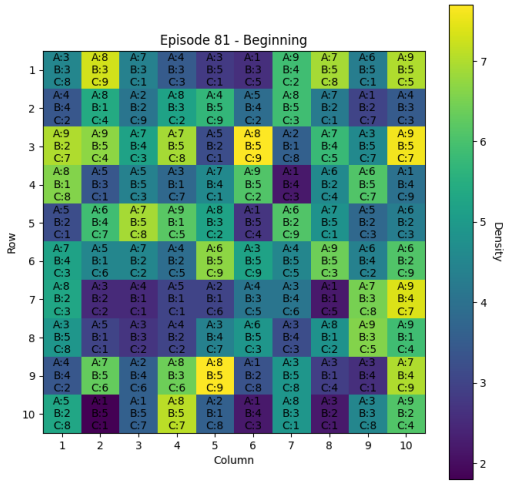

Fig. 3: Density Heat map

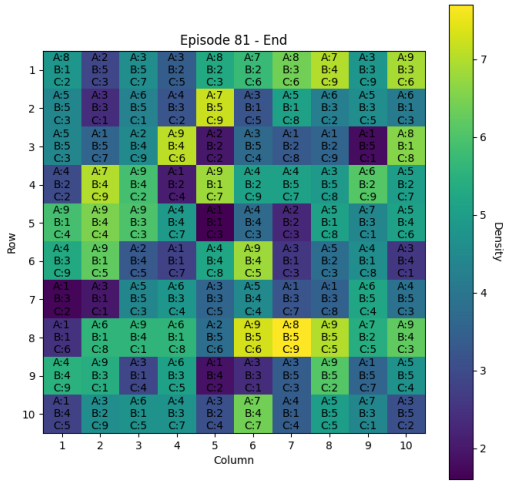

Fig. 4: Density Heat map

marks the cell with the highest density on the grid for each episode. This visualization offers insights into the strategic positioning of drones and the dynamic changes in density across the grid.

The figure provided offers a focused view of data extracted from a subset of episodes, precisely 10 episodes instead of 1000 episodes. This deliberate choice aims to optimize clarity in the visual representation, preventing potential visual clutter that might arise with a larger dataset. This strategic decision ensures a more lucid depiction of the DDPG algorithm's performance in determining optimal cell locations for deploying drones. By narrowing the focus to a subset of episodes, the figure highlights critical patterns and trends without overwhelming the viewer with excessive detail. In summary, the collective presentation of figures in the subset affirms the effectiveness of the DDPG algorithm. The visualizations of rewards, density maps, and drone placements consistently illustrate the algorithm's ability to select optimal cells for deploying drones. This clarity in representation underscores

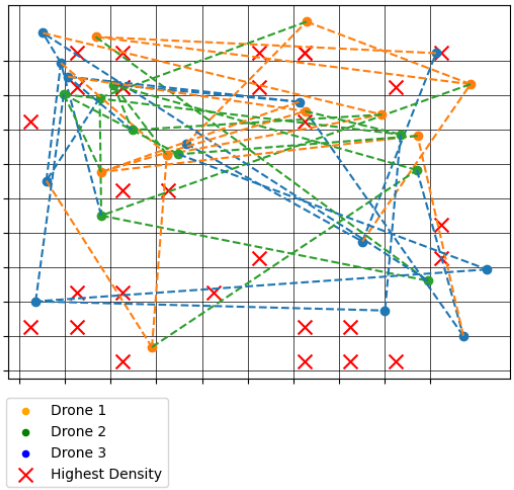

Fig. 5: Drones and high-density cells over episodes

the algorithm's robust performance and ability to make informed decisions across various scenarios, contributing to a comprehensive understanding of its efficacy.

TABLE III: DDPG Hyperparameters for the Second Part

Hyperparameter for second part	Value
State Dimension	9
Action Dimension	3
Learning Rate (Actor)	0.6
Learning Rate (Critic)	0.5
Discount Factor (Gamma)	1
Epsilon Decay	0.995
Buffer Size	10,000
Batch Size	128
Exploration Noise Scale	0.5
Target Update Tau	0.001

TABLE IV: Parameters in Drone Environment for the Second Part

Parameter	Value
num_users	100
num_categories	3
num_tx_antennas	2
num_rx_antennas	2
hline tx antenna gain	10 dB [19]
rx antenna gain	10 dB [19]
beamforming angle deg	65 degrees [20]
shadowing db	2 dB

As outlined previously, in the next phase of our study, we optimize the drone's position to meet diverse user priorities, aiming to fine-tune its location for specific Quality of Service (QoS) requirements. Table III and Table IV show the initial parameters and DDPG hyperparameters used for the second part of the algorithm. Figure 6 visually summarizes the outcomes, illustrating latency for distinct priority levels. The graph showcases our successful spatial optimization strategy, adapting the drone's position to meet unique QoS needs. A closer look at Figure 6 reveals varying latency levels for different user categories, emphasizing the effectiveness of our approach. We prioritize Gold users, ensuring their optimal latency, before addressing other categories. This intentional prioritization guarantees a consistently satisfactory experience for Gold users, demonstrating our commitment to meeting diverse user needs.

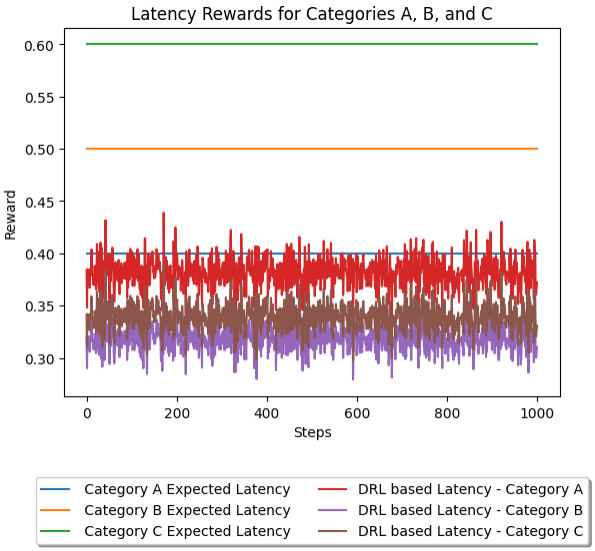

Fig. 6: QoS Latency obtained prioritizing Gold (A) users and expected for different users

Referring to Figure 7, it vividly illustrates throughput rewards for various user categories. Our algorithm consistently excels in meeting the throughput demands of Gold users, as evidenced by the upward trajectory of the Gold user throughput curve. This reaffirms our commitment to delivering a high-quality and dependable service tailored to the specific requirements of this critical user category.

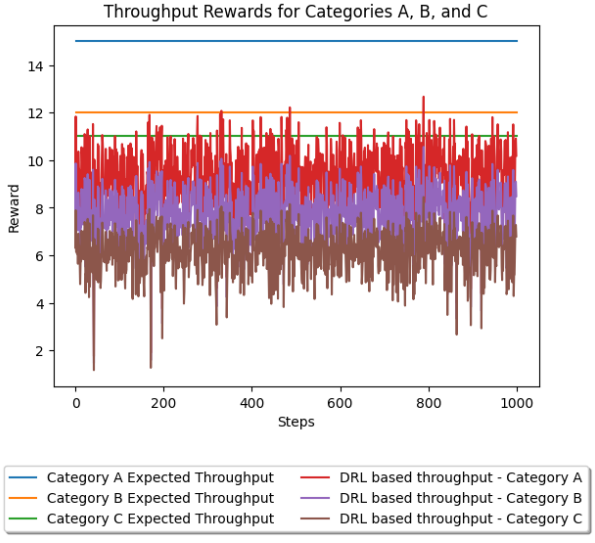

Fig. 7: QoS Throughput obtained prioritizing Gold (A) users and expected for different users

Fig. 8: QoS SINR obtained and expected for different users

The observed trend in SINR rewards, illustrated in Figure 8, underscores the algorithm's consistent adherence to Quality of Service (QoS) standards, with a distinct emphasis on Gold users, particularly User A. SINR, a critical metric gauging signal quality, is reliably fulfilled for premium users across almost every episode. This graphical representation accentuates the algorithm's effectiveness in elevating QoS, notably for prioritized user groups. The special attention given to User A within the Gold category suggests a targeted approach to meet the diverse needs of users, exemplifying the algorithm's nuanced handling of different user segments. Overall, the visual depiction in Figure 8 provides a clear insight into the algorithm's reliability and ability to enhance QoS for designated user categories consistently.

V. Conclusion

This paper employs the DDPG algorithm for drone deployment. It dynamically adjusts drone positions for Quality of Service (QoS) differentiation by assigning different priorities to users based on their QoS requirements. Our methodology emphasizes prioritizing QoS by assigning varying priorities to users. The problem formulation involves ensuring and optimizing QoS tailored to distinct user classes. To tackle the nuanced realm of QoS requirements, we harness the capabilities of DDPG, seamlessly integrating crucial metrics such as latency, Signal-to-Interference-plus-Noise Ratio (SINR), and throughput. Our algorithm consistently meets user QoS expectations through extensive simulations, demonstrating effectiveness in real-world scenarios. Future work aims to extend simulations to intricate environments by introducing multiple Deep Reinforcement Learning (DRL) agents. This extension allows us to explore the algorithm's adaptability and performance in complex scenarios, providing a comprehensive understanding of its capabilities in diverse and dynamic network environments.

References

[1] A. Shahbazi, "Machine Learning Techniques for UAV-assisted Networks," Ph.D. dissertation, Université Paris-Saclay, 2022. [Online]. Available: https://theses.hal.science/tel-03889218

[2] H. Bayerlein, "Machine learning methods for uav-aided wireless networks," Ph.D. dissertation, Sorbonne Université, 2021.

[3] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[4] A. Mekrache, A. Bradai, E. Moulay, and S. Dawaliby, "Deep reinforcement learning techniques for vehicular networks: Recent advances and future trends towards 6g," *Vehicular Communications*, vol. 33, p. 100398, 2022.

[5] N. Gupta, S. Agarwal, and D. Mishra, "Uav deployment for throughput maximization in a uav-assisted cellular communications," in *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2021, pp. 1055–1060.

[6] A. T. Azar, A. Koubaa, N. Ali Mohamed, H. A. Ibrahim, Z. F. Ibrahim, M. Kazim, A. Ammar, B. Benjdira, A. M. Khamis, I. A. Hameed *et al.*, "Drone deep reinforcement learning: A review," *Electronics*, vol. 10, no. 9, p. 999, 2021.

[7] H. Kurunathan, H. Huang, K. Li, W. Ni, and E. Hossain, "Machine learning-aided operations and communications of unmanned aerial vehicles: A contemporary survey," *IEEE Communications Surveys & Tutorials*, 2023.

[8] J. Hu, C. Chen, L. Cai, M. R. Khosravi, Q. Pei, and S. Wan, "Uav-assisted vehicular edge computing for the 6g internet of vehicles: Architecture, intelligence, and challenges," *IEEE Communications Standards Magazine*, vol. 5, no. 2, pp. 12–18, 2021.

[9] X. Gu and G. Zhang, "A survey on uav-assisted wireless communications: Recent advances and future trends," *Computer Communications*, 2023.

[10] P. Boccadoro, D. Striccoli, and L. A. Grieco, "An extensive survey on the internet of drones," *Ad Hoc Networks*, vol. 122, p. 102600, 2021.

[11] A. Alwarafy, M. Abdallah, B. S. Çiftler, A. Al-Fuqaha, and M. Hamdi, "The frontiers of deep reinforcement learning for resource management in future wireless hetnets: Techniques, challenges, and research directions," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 322–365, 2022.

[12] M. Gharibi, R. Boutaba, and S. L. Waslander, "Internet of drones," *IEEE Access*, vol. 4, pp. 1148–1162, 2016.

[13] A. Alwarafy, A. Albaseer, B. S. Ciftler, M. Abdallah, and A. Al-Fuqaha, "Ai-based radio resource allocation in support of the massive heterogeneity of 6g networks," in *2021 IEEE 4th 5G World Forum (5GWF)*. IEEE, 2021, pp. 464–469.

[14] L. Abualigah, A. Diabat, P. Sumari, and A. H. Gandomi, "Applications, deployments, and integration of internet of drones (iod): a review," *IEEE Sensors Journal*, vol. 21, no. 22, pp. 25 532–25 546, 2021.

[15] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on uavs for wireless networks: Applications, challenges, and open problems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[16] Y. Cao, L. Zhang, and Y.-C. Liang, "Deep reinforcement learning for channel and power allocation in uav-enabled iot systems," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[17] Y.-C. Lin and A. Habib, "Quality control and crop characterization framework for multi-temporal uav lidar data over mechanized agricultural fields," *Remote Sensing of Environment*, vol. 256, p. 112299, 2021.

[18] N. Dilshad, J. Hwang, J. Song, and N. Sung, "Applications and challenges in video surveillance via drone: A brief survey," in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2020, pp. 728–732.

[19] C. U. Lee, G. Noh, B. Ahn, J.-W. Yu, and H. L. Lee, "Tilted-beam switched array antenna for uav mounted radar applications with 360° coverage," *Electronics*, vol. 8, no. 11, 2019. [Online]. Available: https://www.mdpi.com/2079-9292/8/11/1240

[20] A. Hughes, S. Y. Jun, C. Gentile, D. Caudill, J. Chuang, J. Senic, and D. G. Michelson, "Measuring the impact of beamwidth on the correlation distance of 60 ghz indoor and outdoor channels," *IEEE Open Journal of Vehicular Technology*, vol. 2, pp. 180–193, 2021.