IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 30, 2020, accepted June 11, 2020, date of publication June 15, 2020, date of current version June 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3002538

A UAV-Assisted Data Collection for Wireless Sensor Networks: Autonomous Navigation and Scheduling

OMAR BOUHAMED[1], (Student Member, IEEE), HAKIM GHAZZAI[1], (Senior Member, IEEE), HICHEM BESBES[2], AND YEHIA MASSOUD[1], (Fellow, IEEE)

[1]School of Systems and Enterprises, Stevens Institute of Technology, Hoboken, NJ 07030, USA
[2]Higher School of Communications of Tunis, University of Carthage, Tunis 2083, Tunisia

Corresponding author: Hakim Ghazzai (hghazzai@stevens.edu)

ABSTRACT Nowadays, Wireless Sensor Networks (WSNs) are playing a vital and sustainable role in many verticals touching different aspects of our lives including civil, public, and military applications. WSNs majorly consist of a few to several sensor nodes, that are connected to each other via wireless communication links and require real-time or delayed data transfer. In this paper, we propose an autonomous Unmanned Aerial Vehicle (UAV)-enabled data gathering mechanism for delay-tolerant WSN applications. The objective is to employ a self-trained UAV as a flying mobile unit collecting data from ground sensor nodes spatially distributed in a given geographical area during a predefined period of time. In this approach, two Reinforcement Learning (RL) approaches, specifically Deep Deterministic Gradient Decent (DDPG) and Q-learning (QL) algorithms, are jointly employed to train the UAV to understand the environment and provide effective scheduling to accomplish its data collection mission. The DDPG is used to autonomously decide the best trajectory to adopt in an obstacle-constrained environment, while the QL is developed to determine the order of nodes to visit such that the data collection time is minimized. The schedule is obtained while considering the limited battery capacity of the flying unit, its need to return the charging station, the time windows of data acquisition, and the priority of certain sensor nodes. Customized reward functions are designed for each RL model and, through numerical simulations, we investigate their training performances. We also analyze the behavior of the autonomous UAV for different selected scenarios and corroborate the ability of the proposed approach in performing effective data collection. A comparison with the deterministic optimal solution is provided to validate the performance of the learning-based approach.

INDEX TERMS Internet-of-Things, data gathering, reinforcement learning, scheduling, unmanned aerial vehicles.

I. INTRODUCTION

Multi-rotor unmanned aerial vehicles (UAVs), *aka* drones, have become the central means for many novel applications. From real-time traffic monitoring to rapid good's delivery services, UAVs have been proven to be extremely beneficial in many fields where human intervention and ground machines are unable to perform in a timely and efficient manner due to diverse issues and physical obstacles [2]–[4].

In recent years, the use of UAVs in wireless Internet-of-things (IoT) systems have been proven to be very efficient by providing reliable connectivity. Thanks to their mobility and flexibility, UAVs can play an important role to support ground transceivers, especially in remote areas and/or for delay-tolerant applications [5]–[8]. This avoids long-range transmissions and the need of relaying data over multiple hops from each sensor node to the sink. In fact, UAVs can navigate as a flying mobile data collector, i.e., collecting data from dispersed sensors to forward it to a central node, e.g., to the cloud [9]. Moreover, the aerial data collection is distinguished by a better channel quality with higher line-of-sight (LoS) opportunities which provides higher communication range and lower latency. However, the use of UAVs for such communication applications is still facing several constraints such as the technical specifications of the UAVs and mainly their limited battery capacities [10]. Hence, the UAVs can be

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

exploited for short time periods before sending them back to the charging station to reload their batteries [11]. Moreover, the UAVs may need to fly at low altitudes to communicate with low transmission power sensors. Hence, navigating within an environment plenty of obstacles remain a challenging task to address. Therefore, there is a need to optimize the navigation and schedule of the UAV when collecting data while considering the different aspects, e.g., battery limitation [12], communication channel, geo-locations of the ground nodes [13], and obstacle avoidance [14].

A. LITERATURE REVIEW

Large-scale IoT systems have seen rapid development in recent years. Wireless Sensor Networks (WSN), composed of thousands of sensing, computing, and communication nodes, form the backbone of such systems. The integration of collaborative UAV-WSN systems led to increased monitoring areas and better overall performance in large regions of interest. Alongside this presented study, there are many relevant and recent contributions, research projects concerning UAV-assisted WSN implementations. In [15], the authors proposed an autonomous-underwater-vehicle (AUV) assisted data gathering scheme based on clustering and matrix completion (ACMC) to improve the data gathering efficiency in an underwater wireless sensor network. In [16], the paper presented a hierarchical structure based on the collaboration between a team of UAVs and a structure of federated wireless sensor networks for crop monitoring in precision agriculture. In [17], the authors proposed a data collection technique in WSNs using projection-based Compressive Data Gathering (CDG) and UAVs, where the CDG was utilized to reduce the number of transmissions and corresponding energy consumption through aggregating data en-route from sets of sensor nodes to a set of projection heads and UAVs were used to enhance the energy efficiency of the sensors by avoiding long-range and multiple hop transmissions to reach the destination sink node. The study in [18] focused on the experimental validation of the Geometry-based Localization technique to localize static sensor nodes on a studied WSN scenario. A scenario that consists of static sensor nodes with one mobile node, mounted on a UAV, broadcasting its position and collecting data. The work in [19] investigated the utilization of UAVs for data collection from dispersed mobile sensors distributed along a predefined linear path, where each having a different velocity. The authors considered different data collection algorithms and revealed that taking into consideration the contact duration and rate transmission between the UAV and the sensor nodes leads to the best performance. In [20], the authors focused on the clustering method in UAV-assisted WSNs, in which UAV can fly to a sensor node to collect data and then fetch the collected data to the nearby base station. In [21], The authors proposed a UAV-assisted cluster-head selection mechanism to balance the energy consumption and to increase the lifetime of a WSN cluster. Also, in [22], the paper presents a UAV-enabled WSN where a flying UAV is employed to collect data from multiple sensor nodes to maximize the minimum average data collection rate from all sensor nodes. Jointly, an optimization of the UAV communication scheduling and three-dimensional (3D) trajectory is taken into consideration. In [23], multiple optimization problems are sequentially solved to group ground sensors into clusters, determine UAV data collection stops, and provide the path that the UAV should follow to complete its tour in an energy-efficient manner. In [24], a Mixed-Integer Linear Program (MILP) was formulated to minimize the total traveled distance of the UAVs when collecting data from dispersed ground sensors. In [25], the authors proposed two modified meta-heuristic-based approximate solutions, namely genetic algorithm, and harmony search, to find the shortest path for multiple UAVs in order to gather data from several roadside units.

However, most of the previous studies are based on complex optimization solutions, evolutionary algorithms, or heuristic approaches [26]–[29]. Nowadays, the new research tendency of using artificial intelligence (AI), precisely, Reinforcement Learning (RL) came out as a new strategy that can grant the flying units sufficient intelligence to make local decisions to accomplish missions. RL is a Machine Learning (ML) technique that can be utilized to train autonomous agents in a semi-supervised manner to operate independently as a bottom-up alternative to the centralized systems discussed earlier. In [30], the authors proposed a ML pipeline for autonomous mobile terminals that first extracts spatial features through a Convolutional Neural Network (CNN), and utilizes Multi-Agent Deep Deterministic Policy (MA-DDPG) to learn to autonomously collect specified data in a region of interest. In [31], the study presented a mission-oriented path planning algorithm based on Q-learning for UAVs to autonomously navigate between tasks in a specified mission area whilst avoiding obstacles. Most of the developed models are limited to simplified 2D navigation space (i.e., fixed altitude), where the UAV is not able to change its altitude to cross over obstacles. In [32] and [33], the authors presented a Q-learning algorithm to solve the autonomous scheduling problem of UAVs. Q-learning was also employed to establish paths while avoiding obstacles in [34]. However, the authors used discrete actions (i.e. the environment is modeled as a grid world with limited UAV action space, degree of freedom). This may reduce the UAV efficiency when dealing with real-world environments, where the flying units operate according to a continuous action space. Moreover, these studies ignore the challenges related to limited battery constraints.

B. CONTRIBUTION

In our context, RL can empower UAVs with sufficient intelligence to make local decisions and autonomously accomplish necessary tasks without requiring the support of a central unit or human involvement. In [35], we have designed a single-algorithm RL solution for routing autonomous agents however, without considering the data collection challenges, and the environment hurdles such as obstacles. In this study, we develop a generic autonomous navigation and scheduling

approach using a combination of two RL-based frameworks for navigating and scheduling a UAV collecting data from multiple ground nodes with the objective of minimizing the data collection time. The developed RL framework are given as follows:

- The first framework aims to provide obstacle-aware navigation for the autonomous UAV to reach its target destination dispersed in a 3D area with continuous action space. To this end, a DDPG-based algorithm is developed with the objective to allow a UAV to determine the best course to accomplish its data gathering safely, i.e. obstacle avoidance. The output of this framework is the shortest and safest route connecting the UAV with its destination as well as the corresponding required flying time.
- The second framework aims to provide a trip plan for the autonomous UAV incorporating all the earlier mentioned aspects to gather the messages from the scattered sensor nodes within a predefined time horizon. The framework allows the UAV to return to the charging station to reload its battery when needed. A Q-learning-based approach is modeled with the objective to allow this UAV to determine the best schedule and rapidly accomplish its missions while taking into consideration the flying time needed, provided by the first framework, to safely reach its targets. A reward function is designed to force the UAV to reach the missions on time and minimize the completion time of the whole trip.

During the training phase, the UAV learns the environment, precisely the WSN, given its technical specifications and its surrounding obstacles and then, during the prediction phase, it determines its data collection plan by figuring out its trajectory and which node to visit first and at which time instant and when it needs to return to recharge its battery. The performances of the RL-based framework are validated through simulations and compared to the results obtained by an optimal MILP-based solution.

The rest of the paper is organized as follows. Section II introduces the system model of the UAV-assisted WSN. Section III-B presents the RL-based approach for data gathering. The section includes both the DDPG and QL frameworks. Section IV provides the simulation results analyzing the performance of the different components of the developed approach. Finally, the paper is concluded in Section V.

II. SYSTEM MODEL

In this study, we consider K wireless sensors deployed randomly in a given 3D geographical area, managed by an autonomous UAV, that is responsible to collect data generated by each node during a given makespan Γ as depicted in Fig. 1.

A. NETWORK MODEL

The set of sensors is defined as $\mathcal{W} = \{w_k, k = 1, \ldots, K\}$. We assume that the sensing units are equipped with a single unidirectional antenna. We presume that each sensor node w_k has a data reading, denoted by message m_k, intended to be sent to a sink node that could be either a cellular base station or a data collection central unit. Each sensing unit w_k is also defined by the parameters st_k, pt_k, and pr_k which are the acquisition time of data by the sensor (i.e., indicates the moment of availability of the message m_k to be transmitted), the time during which the message m_k should be collected by a UAV, and the message priority, respectively. In practice, different cases of st_k and pt_k values can be encountered depending on the application. For example, the scenario where st_k and pt_k are equal to zero and Γ, respectively, means that the message m_k is ready for collection anytime. Another case, we could face in practice, is that the message m_k is only available during a certain time window starting at the instant of data acquisition from the sensor and ending after a fixed time period, otherwise the data is lost. In both cases, we are dealing with delay-tolerant applications. Unlike delay-intolerant applications where data collection must occur in real-time and instantaneously, in delay-tolerant applications, strict real-time data collection is not required. Nevertheless, this does not necessarily mean that latency tolerance is infinite. Therefore, rapid data collection is still required in delay-tolerant wireless sensor networks in order to, for example, acquire the data as soon as possible or allow efficient exploitation of the UAVs for other tasks, e.g., data collection from another network nearby or other missions. To this end, we have developed this learning framework to automate the scheduling of UAVs in order to ensure the data collection of all assigned sensors within a minimum time frame.

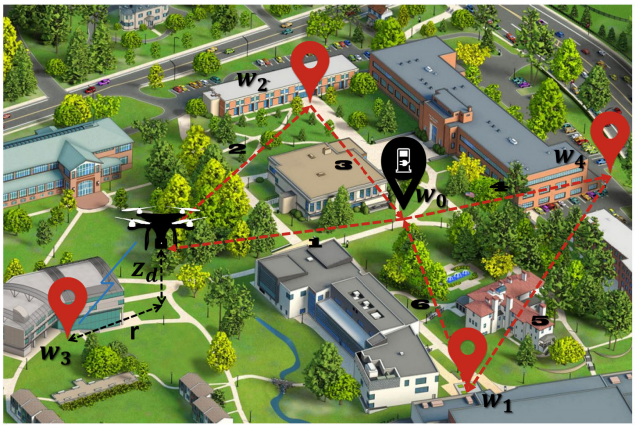

FIGURE 1. Illustration of the autonomous UAV data collection from four sensor nodes $\{w_k, k = 1, \ldots, 4\}$. The UAV starts its journey from the charging station (w_0).

Moreover, we assume that each sensing unit is characterized by its 3D geographical location $loc_{w_k} = (x_{w_k}, y_{w_k}, z_{w_k})$. The UAV, defined as d, is characterized by its current location $loc_d = (x_d, y_d, z_d)$, its battery level and capacity b_d and $\bar{b}_d$, and the charging power level c_d. At the beginning of the makespan, we assume that the UAV is placed at the charging station defined as w_0, located at $loc_{w_0} = (x_{w_0}, y_{w_0}, z_{w_0})$

assimilated as an extra special node always active and to which the UAV can go at any moment of its operation. In general, the UAV can be in three possible modes:

- waiting at the charging station w_0: in this situation, the UAV will need to remain at the charging station where it can reload its battery.
- collecting message from sensor node w_k: the UAV d should be located close to loc_{w_k} to be able to receive data from the sensor node w_k.
- flying from w_k to $w_{k'}$ where $k \neq k'$: the UAV is moving from a position to another to cover it or to return to the charging station w_0.

B. ENVIRONMENT AND OBSTACLE MODELS

Autonomous navigation for UAVs in a real environment is complex. Hence, Without loss of generality, we create a virtual 3D environment with high matching degree to the real-world urban areas. Unlike most of the existing virtual environments studied in the literature, which are usually modeled as a grid world, in this paper, we focus on a free space environment containing 3D obstacles that may have diverse shapes as illustrated in Fig. 2. Consequently, the UAV has the freedom to take any direction and speed to reach its target unlike the grid world, which restricts the mobility of UAV into a finite set of actions. The goal is to train the UAV to fly safely from any arbitrary starting position to reach any destination in the considered area with continuous action space. The objective is to communicate with a ground node and hence we consider that the UAV reaches a destination point at which it is able to communicate with the ground node with the specified data transfer rate.

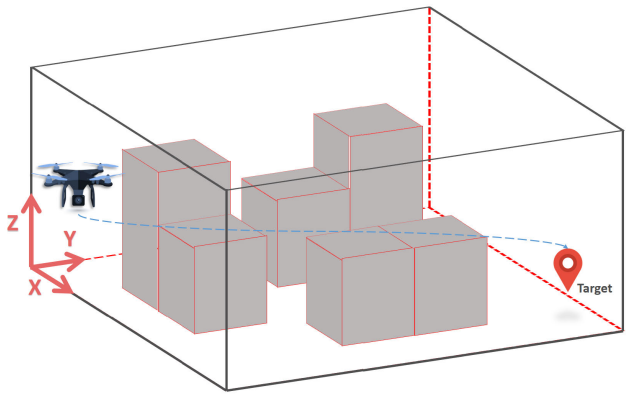

FIGURE 2. Illustration of the autonomous obstacle-aware UAV navigation in an urban environment.

The encoded environment contains a predefined set of obstacles that must be avoided by the UAV when flying. The goal is to train the UAV to fly safely from any arbitrary starting position to reach any destination in the considered area with continuous action space. In the framework, the investigated system assumes the environment's obstacles have different heights. Each one of them is represented by a 3D polygon characterized by its the starting reference point $[x_{obs}, y_{obs}]$, the set containing the edges of the base edg_{obs}, and its height h_{obs}. Hence, if it is flying at an altitude higher than the obstacle's height, the UAV can cross over the obstacles. Otherwise, the UAV can avoid it by flying around.

We also consider that the UAV is flying at an average speed V_d and hence, requires ft_{d,w_k} seconds to reach loc_{w_k} when leaving from its current location loc_d. The flying time between the two locations depends on the trajectory, denoted by Φ, followed by the UAV. The flying time can be then expressed as follows:

$$ft_{d,w_k} = \frac{\Delta^{\Phi}(d, w_k)}{V_d}, \tag{1}$$

where $\Delta^{\Phi}(d, w_k)$ is the distance traveled by the UAV to safely reach its destination w_k from its current starting position.

C. CHANNEL MODEL AND ACHIEVABLE RATE EXPRESSION

Due to the nature of the investigated network, we are using an air-to-ground channel type to model the communication link between the UAV and the ground sensors, i.e., $z_{w_k} = 0$. In this study, we only consider the large-scale path loss effect in the channel gain, denoted by h. Its expression is given as follows[1]:

$$h\left(\Delta^E(d, w_k)\right) = \frac{1}{\sqrt{PL\left(\Delta^E(d, w_k)\right)}}, \tag{2}$$

where PL is the path loss in its linear form between UAV d and sensor w_k. It depends on the euclidean distance separating the UAV and its target sensor, which is defined as $\Delta^E(d, w_k)$. The LoS links between the flying unit and the ground sensing nodes are assumed to be available with a certain probability denoted by p^{LoS}. The average air-to-ground free space path loss PL in dB is given as follows [36]:

$$PL_{dB}\left(\Delta^E(d, w_k)\right) = p^{LoS} PL_{dB}^{LoS} + (1 - p^{LoS}) PL_{dB}^{NLoS}, \tag{3}$$

with the non-LoS free path loss PL^{NLoS} and LoS free path loss PL^{LoS} are expressed as follows [36]:

$$PL_{dB}^{X}\left(\Delta^E(d, w_k)\right) = 10n \log_{10}\left(\frac{4\pi f \Delta^E(d, w_k)}{c}\right) + L_{dB}^{X}, \tag{4}$$

where $X = \{LoS, NLoS\}$, f is the carrier frequency, n is the path loss exponent, c is the speed of light, and L_{dB}^{X} is the average additional loss due to non-LoS/LoS link. The probability of LoS link between the UAV d and the sensor node w_k is given as follows:

$$p^{LoS}\left(\Delta^E(d, w_k)\right) = \frac{1}{1 + \beta_1 \exp\left(-\beta_2[\theta\left(\Delta^E(d, w_k)\right) - \beta_1]\right)}, \tag{5}$$

[1] Since we are dealing with delay-tolerant applications and determining the UAV schedule over a long period of time, we focus on the system performance based on its average statistics and hence, ignore the fast fading effect.

where β_1 and β_2 are constant parameters that depend on the environment and $\theta\left(\Delta^E(d, w_k)\right)$ is the elevation angle between the UAV d and the sensor node. Evaluating the expression of p^{LoS} shows a trade-off between the path loss and the distance separating the UAV and the sensor node that should be taken into consideration while finding the optimal altitude for the UAV. The higher the UAV's altitude, the lower the path loss. On the other hand, this increases the distance between the flying unit and the ground sensor node which may lead to a lower the transmission rate. In order to compute the required time to transmit a message m_k, denoted by T_k, from a sensor node to the UAV d, we compute the average transmission rate R_{d,w_k}. The transmission time T_k is expressed as:

$$T_k = \frac{m_k}{R_{d,w_k}}, \tag{6}$$

where m_k is the message size and R_{d,w_k} is computed using the truncated Shannon equation as follows:

$$R_{d,w_k} = \begin{cases} R_{max}, & \text{if } \gamma_{d,w_k} \geq \gamma_{max} \\ B\log_2\left(1+\gamma_{d,w_k}\right), & \text{if } \gamma_{min} < \gamma_{d,w_k} < \gamma_{max} \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where B is the total bandwidth of the channel, γ_{d,w_k} is the signal-to-interference-plus-noise (SINR) ratio expressed as follows:

$$\gamma_{d,w_k} = \frac{P^{tr}|h\left(\Delta^E(d, w_k)\right)|^2}{\Omega + N_0 B}, \tag{8}$$

where P^{tr} is the ground unit power in Watt with which the signal is transmitted, R_{max} is the maximum possible transmission rate corresponding to γ_{max}. Below γ_{min}, we assume that the signal is not effectively detected and hence, $R_{d,w_k} = 0$. Finally, Ω is the average interference power and N_0 is an additive white Gaussian noise. It is worth to note that, in this framework, we consider the average interference that might be caused by external devices simultaneously communicating while the UAV receives data from the ground nodes. In the context of multi-UAVs, mutual interference should be mitigated, e.g., with optimized radio resource allocation. The focus of this paper is mainly on the autonomous navigation and scheduling of the data gathering UAV. Incorporating interference mitigation solution is more elaborate and will be investigated in the future extension of this work.

In order to determine the UAV stops at which the UAV needs to hover to serve a ground node with a satisfactory data rate. We consider the characteristics of the air-to-ground channel in order to guarantee reliable communication by solving the following optimization problem, which aims to determine the maximum distance (i.e., radius) that can separate the stop location of the UAV from the ground sensor:

$$\begin{aligned} &\underset{z_d, r}{\text{maximize}} \quad r = \sqrt{(x_d - x_{w_k})^2 + (y_d - y_{w_k})^2}, \\ &\text{subject to: } R_{d,w_k} \geq \eta R_{max}, \end{aligned} \tag{9}$$

where r is the radius between the sensor and the parallel projection of the UAV stop location on the ground 2D plan, as shown in Fig. 1, and η is a tolerance coefficient regulating the minimum transmission rate threshold where $0 \leq \eta \leq 1$. Hence, r represents the maximum range of the stop where the UAV needs to hover to collect data from the ground node w_k. We assume that the exact UAV stop is chosen on the segment connecting the charging station and the node w_k with a radius r.

The optimization problem formulated in 9 is only used to determine the UAV stop guaranteeing a communication data rate higher than ηR_{max}. In this sense, the UAV does not need to hover on the top of the ground node to collect the data. Reducing η allows the UAV to stop farther but may increase the data transfer time. The optimization problem 9 is non-convex due to channel expression but can be sub-optimally solved using numerical or heuristic solutions.

In this paper, we are essentially focusing on determining the UAV trajectories and schedules given specific UAV stops at which it has to hover to collect the data. As future work, we will investigate a more elaborate problem where more optimized UAV stops are also determined.

D. MAKESPAN DISCRETIZATION

In this paper and in order to overcome the time dimension dilemma, we divide the makespan into N time periods $s_1, \ldots, s_N$ having the same length τ. Consequently, we associate to each sensor node w_k a certain number of time periods during which its message m_k should be collected. We define the set of time periods $\mathcal{T}_{w_k}$ associated to sensor node w_k as follows: $\mathcal{T}_{w_k} = \left\{s_{st}^{w_k}, \ldots, s_{et}^{w_k}\right\}$ where $s_{st}^{w_k}$ denotes the time period where the data is gathered by the sensor (e.g., the message m_k is available) while $s_{et}^{w_k}$ corresponds to the ending time of availability of the message.

E. ENERGY MODEL

The power consumption of the UAV is mainly composed of two components, one related to the propulsion and the other to the mounted communication interface. In this study, we adopt the following UAV power consumption presented in [37]:

$$P(V_d) = P^{bl}\left(1 + \frac{3{V_d}^2}{U_{tip}^2}\right) + P^{ind}\left(\sqrt{1 + \frac{{V_d}^4}{4v_0^4}} - \frac{{V_d}^2}{2v_0^2}\right)^{\frac{1}{2}} + \frac{1}{2}d_0\zeta sA{V_d}^3, \tag{10}$$

where P^{bl}, P^{ind}, U_{tip}, v_0, d_0 and s corresponds to the blade profile power, the induced power level when the UAV statically hover, the tip velocity of the rotor blade, the mean rotor induced speed in hover, the fuselage drag ratio and rotor solidity, respectively. The parameters ζ and A denote the air density and rotor disc area, respectively. The amount of power to be consumed by the UAV when hovering, i.e., when receiving data from other devices, can be obtained by setting V_d in (10) to 0 and hence, it can be expressed as follows:

$$P^{hov} = P(0) = P^{bl} + P^{ind}. \tag{11}$$

When receiving data, the UAV is assumed to hover statically and hence, consumes the hovering power as well as a reception power consumed by the communication interface, denoted by P^{md}. The total power consumed during data reception is denoted by P^{rx} and expressed as follows:

$$P^{rx} = P^{hov} + P^{md}. \tag{12}$$

The battery level of the UAV, denoted by b_d, can vary according to its three possible modes described earlier:

• 'Standby' mode: the UAV battery is charged as follows:

$$b_d(s_{n+1}) = \min(\bar{b}_d, b_d(s_n) + c_d \cdot \tau), \tag{13}$$

where $b_d(s_n)$ is the battery level at time period s_n.

• Collecting message from sensor node w_k: the battery level is reduced as follows:

$$b_d(s_{n+1}) = \max(0, b_d(s_n) - P^{rx} \cdot \tau). \tag{14}$$

• Moving from a position to another: in this case, the UAV battery is updated as follows:

$$b_d(s_{n+1}) = \max(0, b_d(s_n) - P(V_d) \cdot \tau). \tag{15}$$

III. RL FOR AUTONOMOUS DATA GATHERING

This section introduces the proposed autonomous UAV data collection approach and its different components. Afterwards, it defines the inputs, outputs, and reward functions of the navigation and scheduling RL frameworks.

A. PROPOSED RL APPROACH COMPONENTS

RL is a sub-category of ML that allows the agent to understand their surrounding environment and convert situations into actions such that it maximizes a certain metric usually defined as a reward [38]. The learner has zero knowledge about which actions to take, but instead, it must discover which ones yield the most reward by gaining experience during a training phase. Another RL characteristic is that actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two features, trial-and-error search, and delayed reward are the most important distinguishing characteristics of RL. In other words, the idea behind RL is to make the agent interactively learn from its environment using the received feedback from its own experiences. In this paper, the essential behavior that we intend to integrate into the UAV is to learn and take the best course of actions to maximize the total reward count and hence, achieve its application objective, i.e., gathering data from the dispersed sensors.

To this end, we propose to adopt a joint autonomous data collection approach where two RL frameworks are jointly utilized to enable the autonomous navigation and scheduling of the UAV so it can accomplish its data gathering tour. The components of the proposed approach are illustrated in Fig. 3. The first RL framework is based on the DDPG model and its objective is to find the safest and fastest route that the UAV needs to follow to go from any starting point to any destination in the 3D map while avoiding obstacles. In addition to

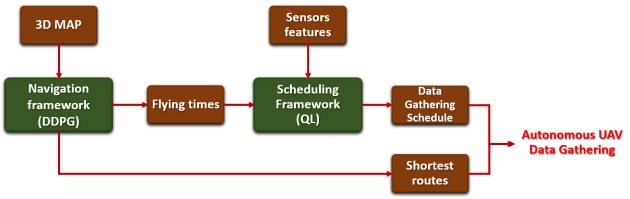

FIGURE 3. Illustration of the proposed autonomous data collection approach and the interconnection between the navigation and scheduling frameworks.

the route, the navigation framework delivers the total distance $\Delta^{\Phi}(d_1, d_2)$ to be traveled by the UAV to go from a position d_1 to another d_2 and the corresponding flying time, expressed in (1), so it can be integrated to the scheduling framework, i.e., the second RL framework, and used to determine the UAV scheduling. Indeed, the second framework considers the features of the UAV itself, the sensors, and the charging station, presented in Section II, to determine effective scheduling that takes into account the flying times between the different nodes, the corresponding energy consumption, and the data acquisition time of the data. The scheduling framework will automatically determine at which instant the UAV needs to return to the charging station to reload its battery. With QL, the scheduling framework is modeled such that the collection time is minimized. The proposed autonomous navigation and scheduling framework will deliver at the beginning of the makespan the action plan to be followed by the agent to complete its collection tour.

The choice of the RL approaches depends on the nature and the complexity of the investigated problem. For the data gathering framework, we proposed the use of Q-learning, a value-based reinforcement learning algorithm, which is considered as a tabular method. Tabular methods can solve problems in which the state and action spaces are small enough to be presented as arrays and tables. The investigated autonomous scheduling problem falls under the scope of such methods since the decisions that can be made by the UAV are limited: waiting at the charging station, collecting data from a sensor, or flying from a sensor node to another. Moreover, QL is considered computationally cheaper compared to other forms of RL algorithms which allows the achievement of the scheduling solutions with reduced complexity, i.e., less running time. As for the autonomous navigation problem, in contrast with the previous framework, we are facing a case of large state and action spaces, i.e., free 3D mobility, which makes tabular approaches impossible to use. That's why, we proposed the use of DDPG, in which the tabular method is substituted for neural network-based approach, which make it very suitable for high dimensional continuous action problems.

B. DDPG LEARNING FOR AUTONOMOUS NAVIGATION

To calculate the distance $\Delta^{\Phi}(d_1, d_2)$ traveled between two locations d_1 and d_2, the UAV needs to make a series of actions so it can move step by step till it reach its destination d_2.

At each action, we assume that the UAV chooses a distance to cross according to a certain direction in the 3D space during Δt units of time. The action is modeled using the spherical coordinates (ρ, ϕ, ψ). In other words, if the UAV is located at position (x_d, y_d, z_d) at time instant t, then it is location at time instant $t + \Delta t$, after taking a navigation action (ρ, ϕ, ψ) is expressed as follows:

$$\begin{aligned} x_d(t + \Delta t) &= x_d(t) + \rho \sin\phi \cos\psi, \\ y_d(t + \Delta t) &= y_d(t) + \rho \sin\phi \sin\psi, \\ z_d(t + \Delta t) &= z_d(t) + \rho \cos\phi, \end{aligned} \tag{16}$$

where ρ is the traveled radial distance by the UAV in each step ($\rho \in [\rho_{min}, \rho_{max}]$), where ρ_{max} is the maximum distance that the UAV can cross during the step length Δt. Its value depends on the speed of the UAV v_d. The parameter ψ denotes the inclination angle ($\psi \in [0, 2\pi]$), and ϕ represents the elevation angle ($\phi \in [0, \pi]$). For instance:

- if $\rho = \rho_{max}$, $\phi = \pi$, and any value of ψ, the UAV moves by ρ_{max} along the z-axis.
- if $\rho = \rho_{max}$, $\phi = \pi/2$, and $\psi = 0$, the UAV moves along the x-axis.

As previously stated, RL is a category of semi-supervised ML. It allows the UAV to automatically determine the ideal behavior taking into account the UAV characteristics and the environment's constraints, in order to maximize its performance. A simple reward feedback, *aka* reinforcement signal, is required for the UAV to learn how to behave. There are many different algorithms, such as the policy gradient methods, that tackle this issue. They rely on optimizing parametrized policies with respect to cumulative reward by gradient descent.

DDPG was developed as an extension of deep Q-network (DQN) algorithms introduced by Mnih *et al.* [39], which was the first approach combining deep and RL but only by handling low-dimensional action spaces. DDPG is also a deep RL algorithm, that can deal with large-dimensional/infinite action spaces. It tries to find an efficient behavior strategy for the agent to obtain maximal rewards in order to accomplish its assigned tasks [40]. This DPG algorithm has the capability to operate over continuous action spaces which is a major hurdle for classic RL methods like Q-learning.

1) ACTOR-CRITIC LEARNING mModel

DDPG is based on the actor-critic algorithm. It is essentially a hybrid method that combines the policy gradient and the value function. The policy function μ is known as the actor, while the value function Q^{nv} is referred to as the critic. Essentially, the actor output is a navigation action chosen from a continuous action space, given the current state of the environment $a^{nv} = \mu(s^{nv}|\theta^{\mu})$, which, in our case, has the form of a tuple $a^{nv} = [\rho, \phi, \psi]$. As for the critic, its output $Q^{nv}(s^{nv}, a^{nv}|\theta^{\mu})$ is a signal having the form of a Temporal Difference (TD) error to criticize the actions made by the actor knowing the current state of the environment. A diagram summarizing the actor-critic architecture is given in Fig. 4.

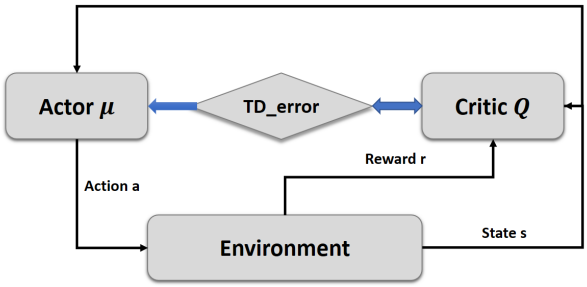

FIGURE 4. Architecture of the actor-critic learning model.

Note that the training phase of the DDPG model is executed for M^{nv} episodes where each one of them accounts for T^{nv} steps. We use the index i to denote an iteration within a single episode where $i = 1, \ldots, T^{nv}$. The actor and critic are designed with neural networks. The value network is updated based on Bellman equation [41] by minimizing the mean-squared loss between the updated Q value and the original value, which can be formulated as shown in Algorithm 1 (line 11). As for the policy network's update (line 13), it is based on the deterministic policy gradient theorem [40].

Some practical tricks are used to enhance the performance of the framework. A trade-off between exploration and exploitation is made by the use of ϵ-greedy algorithm, where a random navigation action a_i^{nv} is selecting with ϵ^{nv} probability, otherwise a precise navigation action $a_i^{nv} = \mu(s_i^{nv}|\theta^{\mu})$ is selected according to the current policy with a $1 - \epsilon^{nv}$ probability. Furthermore, an experience replay buffer b^{nv}, with size B^{nv}, is used during the training phase to break the temporal correlations. Each interaction with the environment is stored as tuples in the form of $[s_i^{nv}, a^{nv}, r^{nv}, s_{i+1}^{nv}]$, which are the current state, the navigation action to take, the reward of performing navigation action a^{nv} at state s_i^{nv}, and the next state, respectively (Algorithm 1 (line 9)) and, during the learning phase, a randomly extracted set of data from the buffer is used (Algorithm 1 (line 10)). Also, target networks are exploited to avoid the divergence of the learning algorithm caused by the direct updates of the networks' weights with the gradients obtained from the TD error signal.

2) REWARD FUNCTION FOR THE NAVIGATION FRAMEWORK

In an obstacle-constrained environment, the UAV must avoid obstacles and autonomously navigate to reach its destination in real-time. Therefore, the reward function, denoted by f_r, is modeled such that it encourages the UAV to reach its destination and, at the same time, penalizes it when crashing. Thus, the reward function is composed of two terms: target guidance reward and obstacle penalty. The target guidance reward, denoted by f_{gui}, is used to motivate the flying unit to reach its target as fast as possible, while the obstacle penalty, denoted by f_{obp} is responsible for alerting the UAV to keep a certain safety distance off the obstacles. The reward function

Algorithm 1 DDPG

Randomly initialize critic $Q^{nv}(s^{nv}, a^{nv}|\theta^{\mu})$ and actor $\mu(s^{nv}|\theta^{\mu})$ neural networks with weights θ^{Q} and θ^{μ}.
Initialize target networks Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^{Q}$, $\theta^{\mu'} \leftarrow \theta^{\mu}$.
Initialize replay buffer b^{nv}.
for episode = 1,...,M^{nv} **do**
  Receive first observation s_1^{nv}.
  for $i = 1, \ldots, T^{nv}$ **do**
    Select a_i^{nv} based on ϵ-greedy algorithm: select random action a_i^{nv} with ϵ^{nv} probability, otherwise $a_i^{nv} = \mu(s_i^{nv}|\theta^{\mu})$ according to the current policy.
    Execute action a_i^{nv} and observe reward r_i^{nv} and new state s_{i+1}^{nv}.
    Store transition $[s_i^{nv}, a_i^{nv}, r_i^{nv}, s_{i+1}^{nv}]$ in b^{nv}.
    Sample a random batch of N^{nv} transitions $[s_j^{nv}, a_j^{nv}, r_j^{nv}, s_{j+1}^{nv}]$.
    Set $y_j^{nv} = r_j^{nv} + \gamma Q'(s_{j+1}^{nv}, \mu'(s_{j+1}^{nv}|\theta^{\mu'})|\theta^{Q'})$.
    Update critic by minimizing the loss:
    $$L = \frac{1}{N^{nv}} \sum_j (y_j^{nv} - Q^{nv}(s_j^{nv}, a_j^{nv}|\theta^{Q}))^2$$
    Update the actor policy using policy gradient:
    $$\nabla_{\theta^{\mu}} \mu|_{s_j} \approx \frac{1}{N} \sum_j \nabla_a Q(s, a|\theta^{Q})|_{s=s_j, a=\mu(s_j)} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})|_{s_j}$$
    Update the target networks:
    $$\theta^{Q'} \leftarrow \nu\theta^{Q} + (1-\nu)\theta^{Q'}$$
    $$\theta^{\mu'} \leftarrow \nu\theta^{\mu} + (1-\nu)\theta^{\mu'}$$
  end for
end for

is formulated as follows:

$$f_r(\Delta(d, d_2), \sigma) = (1-\beta) f_{gui}(\Delta(d, d_2)) + \beta f_{obp}(\sigma), \quad (17a)$$

$$f_{gui}(\Delta(d, d_2)) = \exp(-5\Delta(d, d_2)^2), \quad (17b)$$

$$f_{obp}(\sigma) = \exp(-100\sigma) - 1, \quad (17c)$$

where $\Delta(d, d_2)$ measures separating the current location of the UAV to its destination d_2 and σ is the crash depth explained in Fig. 5 and β is a variable that regulates the balance between f_{obp} and f_{gui}. The obstacle penalty is modeled as a function of the crash depth σ to conserve the continuous nature of the reward function instead of using a discrete penalty, which proved to be more efficient to help the model to converge. When the crash depth is high, the UAV receives a higher penalty, whereas a small crash depth results in a lower penalty. The use of this approach helps the UAV learn efficiently over the training episodes on how to adjust its trajectory to avoid obstacles.

3) TRANSFER LEARNING

Transfer learning is a machine learning technique used to transfer the knowledge to speed up training and improve the performance of deep learning models. The proposed approach to train the UAV consists of two steps. Initially, we train the model in an obstacle-free environment. Training in such an environment grants the UAV the capability to reach any target in the covered 3D area with continuous space action. Then, the trained model on the obstacle-free environment will serve as a base for future models trained on other environments with obstacles. Afterwards, we transfer the acquired knowledge (i.e., source task) and use it to improve the UAV learning of new tasks where it updates its path based on the obstacle locations while flying toward its target. The adopted transfer learning technique applied to DDPG for autonomous UAV navigation is illustrated in Fig. 6. Once the training phase is completed offline, the UAV is capable to make instant decisions, while interacting with the environment, to manage real-time missions. Hence, it can autonomously provide a safe and fast trajectory to go from a location d_1 to a destination d_2 characterized by a distance $\Delta^{\Phi}(d_1, d_2)$ and a flying time ft_{d_1,d_2}.

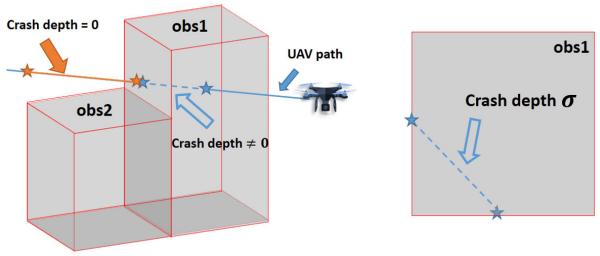

FIGURE 5. Example representing a collision scenario. The UAV's altitude is less than the obstacle's height (obs1). The action is chosen such that the UAV crosses the obstacle. This results in an obstacle penalty reflecting the underwent depth.

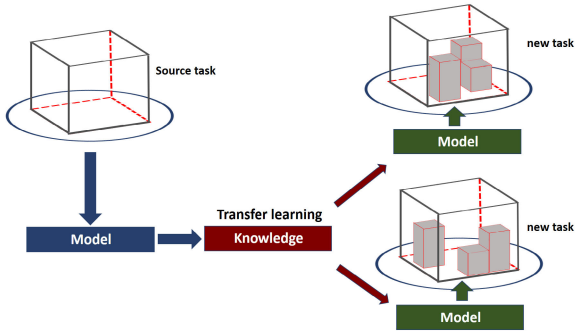

FIGURE 6. Illustration of the transfer-learning technique.

C. QL FOR AUTONOMOUS SCHEDULING

Once the UAV learns how to navigate safely within the environment, we develop an autonomous scheduling framework to accomplish the data collection tour while taking into account the energy consumption of the flying unit and the data acquisition time windows of the sensors. To this end, we propose to employ one of the fastest and simplest-structured RL algorithms: the QL, where ***Q*** in QL stands for quality [42]. Quality in this case represents how useful a given action is in maximizing coverage on highest priority events. QL is an off-policy temporal difference learning [43]. It is considered off-policy because the QL function learns from actions that are outside the current policy, like taking random actions, and therefore policy is not needed. More specifically, QL seeks to learn a policy that maximizes the total reward.

1) Q-TABLE STRUCTURE

In this framework, we focus on the positions that the UAV can visit during his tour, namely, the charging station w_0 and the locations at which it will collect the data from the sensors $w_k, \forall k \in \{1, \ldots, K\}$. We propose to divide the battery capacity into K energy levels. Thus, each UAV has E battery charging states that we refer as $\mathcal{L} = \{l_1, \ldots, l_E\}$, where the highest energy level $l_E = [\bar{b}(1 - \frac{1}{E}), \bar{b}]$ and the lowest level $l_1 = [0, \frac{\bar{b}}{E}]$. Since the UAV may make any decision (wait at the charging station, fly from a position to another, or collect data from a sensor) at any instant of time and at any battery level allowing it to execute that decision, we create the set of states $\mathcal{S}$ by concatenating the different $\mathcal{T}_{w_k}, \forall k = 0, \ldots, K$, while taking into account the different battery level states introduced previously as follows: $\mathcal{S} = \left\{\mathcal{T}^{l_1}_{w_0}, \ldots, \mathcal{T}^{l_1}_{w_K}, \ldots, \mathcal{T}^{l_E}_{w_0}, \ldots, \mathcal{T}^{l_E}_{w_K}\right\}$. On the other hand, the set of actions is defined as $\mathcal{A} = \mathcal{W}$, which encompasses the set of sensors plus the charging station. Hence, the $\boldsymbol{Q}$-table corresponds to a matrix having the size $[N(K+1)E \times K]$. At each action, the UAV will choose one of the sensors to serve given its current state.

QL revolves around the notion of updating the elements of the table $\boldsymbol{Q}$ that we present its structure in Fig. 7. The elements of the $\boldsymbol{Q}$-table denote the values of doing an action $a \in \mathcal{A}$ when the UAV is at a state $s \in \mathcal{S}$. Initially, the $\boldsymbol{Q}$-table is filled with zero-elements. Note that the training phase of the QL algorithm is executed for Ep episodes where each one of them accounts for I iterations. The following update rule inspired from the Bellman equation is used to fill the $\boldsymbol{Q}$-table [41]:

$$\begin{aligned} &Q[s_n^{w_k,l}, a](\iota) \\ &\quad = (1-\alpha)Q[s_n^{w_k,l}, a](\iota - 1) \\ &\qquad + \alpha[\mathcal{R}(s_n^{w_k,l}, a) + \zeta \max_{a \in \mathcal{A}} Q[s_{n'}^{w_{k'},l'}, a](\iota + 1)], \end{aligned} \tag{18}$$

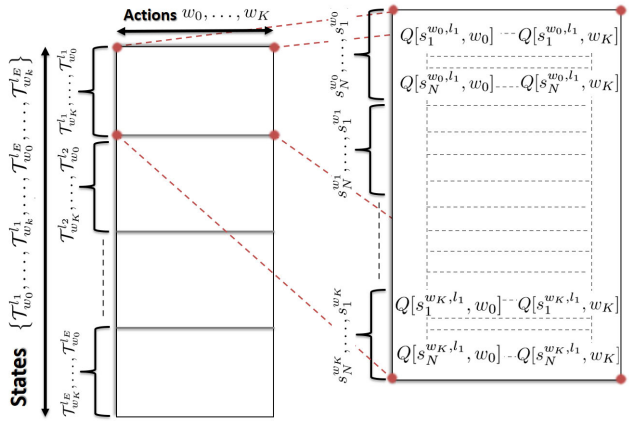

FIGURE 7. Q-table structure. Initially, the Q-table is divided into E sub-tables referring to each possible energy level. Likely, each sub-table is divided into divisions based on the the current state denoted by $w_0, \ldots, w_K$ (e.g., current mission). Each division is a matrix, presenting the worth of collecting data from sensor $w_{k'}$ (i.e., taking action $w_{k'}$) while the UAV is at sensor w_k at a time slot n.

where ι indicates the iteration of the learning algorithm where $\iota = 1, \ldots, I$, α denotes to the learning rate ($0 \leq \alpha \leq 1$), and ζ is the discount factor which regulates the impact of the next taken actions on the evaluation of the current state $0 \leq \zeta \leq 1$. The function $\mathcal{R}(s_n^{w_k,l}, a)$ quantifies the reward obtained when the action a is taken if the UAV is at the state $s_n^{w_k,l}$. Finally, the last term in (18), specifically $\max_{a \in \mathcal{A}} Q[s_{n'}^{w_{k'},l'}, a](i+1)$, is added to return the highest $\boldsymbol{Q}$ value among all possible actions in $\mathcal{A}$ associated to the new state $s_{n'}^{w_{k'},l'}$.

2) REWARD FUNCTION

Recall that, in our setting, we aim to collect the messages from dispersed sensors as fast as possible while taking into account the limited battery capacity and the delay needed to move from a location to another. If the UAV has no message to collect, it is encouraged to return to w_0 to either charge its battery in case of a low battery charging state or to go into standby mode. To deal with the flying time constraint, the reward function of the UAV is expressed as a function of the starting and ending times ($s_{st}^{w_k}$ and $s_{et}^{w_k}$) of the sensor node data acquisition period as well as the flying time ft spent between the sensors' locations provided by the navigation framework.

For an efficient assessment of the chosen action, we introduce the following time metrics:

- The arrival time of the UAV to collect data from sensor node a after being at state $s_n^{w_k}$, denoted by $AT(s_n^{w_k}, a)$, is expressed as:

$$AT(s_n^{w_k}, a) = s_n \cdot \tau + ft_{d,a}, \tag{19}$$

- The waiting time that the UAV must wait when it arrives to the data collection location of sensor a before starting the data reception after leaving the state $s_n^{w_k}$, denoted by $WT(s_n^{w_k}, a)$, is expressed as:

$$WT(s_n^{w_k}, a) = \begin{cases} s_{st}^{w_k}\, \tau - AT(s_n, a), & \text{if } s_{st}^{w_k} \leq AT(s_n, a), \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

- The remaining time left for collecting the data of sensor a if the UAV arrives its corresponding data collection stop after being at state $s_n^{w_k,l}$, denoted by $RT(s_n^{w_k}, a)$, is expressed as:

$$RT(s_n^{w_k}, a) = \begin{cases} s_{et}^{w_k}\, \tau - AT(s_n^{w_k}, a), & \text{if } s_{st}^{w_k} \leq AT(s_n^{w_k}, a), \\ (s_{et}^{w_k} - s_{st}^{w_k})\, \tau, & \text{otherwise.} \end{cases} \tag{21}$$

The metrics AT, WT, and RT are depicted in Fig. 8 for two different cases depending on the arrival time of the UAV to the data collection location of the server with respect to its data acquisition period. The metric AT is added to encourage the UAV collect the data from the sensor when the data acquisition time window is active. The UAV is penalized if it arrives to sensor node too early, i.e., WT is high. The remaining time left metric RT is considered to encourage the UAV go to sensors as soon as possible once the data to be collected

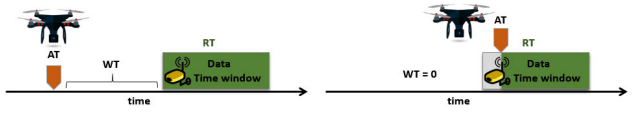

FIGURE 8. Explanatory diagrams illustrating the metrics *AT*, *WT*, and *RT* for two different cases. (left) when the UAV arrives to the data collection location before the availability of the message to transmit, (right) when the UAV arrives to the data collection location after the message acquisition time.

is available. This helps in accelerating the data collection process from all the nodes. The reward function, $\Psi(s_n^{w_k,l}, a)$, measures the worthiness of collecting data from sensor w_k when the UAV is located at state $s_n^{w_k,l}$ and is formulated as follows:

$$\Psi(s_n^{w_k,l}, a) = \begin{cases} pr_m \max\left(50\left(\frac{RT(s_n^{w_k,l}, a)}{(s_{et}^a - s_{st}^a)\cdot\tau}+1\right)\cdot\frac{1}{1+WT(s_n^{w_k,l}, a)}, 0\right), & \text{if } a \in \mathcal{A}\backslash\{0\}, \\ 100\left(-0.65\cdot\frac{b_d}{\bar{b}_d}+1\right), & \text{if } a \in \{0\}, \\ & \text{if } b_d \leq 0, \end{cases}$$

$$\Psi(s_n^{w_k,l}, a) = \begin{cases} 0, & \text{if } a \in \mathcal{A}\backslash\{0\}, \\ 100, & \text{if } a \in \{0\}, \end{cases} \quad \text{otherwise,} \tag{22}$$

where Ψ is formulated as a function having values $\in (0, 100\cdot \bar{pr})$ where $\bar{pr}$ is the highest priority that can be assigned to a sensor so it is data is collected first. The reward function for data collection from sensing nodes except w_0 is multiplied by pr_{w_k} to force the UAV prioritizes the mission with higher priority. Fig. 9 depicts an example of the reward function Ψ with respect to the time window of the data acquisition and discusses different cases. The reward function is modeled to encourage the UAV to cover tasks in case of filled battery (sensing nodes score higher than w_0 score) and the opposite, in case of empty battery.

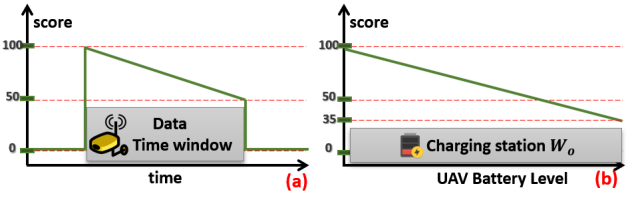

FIGURE 9. Illustration of the Reward function Ψ: (a) when collecting data from the sensing node for $w_k \neq w_0$. The reward is plotted as a function of time. It is equal as long as there is no data to collect from the sensor. Then, it reaches its maximum when *RT* is exactly equal to the acquisition time period. Finally, it linearly decreases to zero till the end of the data acquisition time. (b) When the UAV is at the charging station (w_0). The reward is plotted as a function of the UAV battery level. It is modeled as a linear function inversely proportional to the battery level. It is a at the maximum 100 when $b_d = 0$ and at the minimum when $b_d = \bar{b}_d$.

Also, the scheduling framework is formulated in a way that the UAV respects the transmission time of the messages (i.e., the UAV remains at the sensor location until it acquires the entire message from the sensing unit). Additionally, with the intention of better management and efficient exploitation of the UAV's energy, a reward score depending on the UAV battery charging state, is assigned to the w_0 when $a \in \{0\}$. Furthermore, during the offline training, an energy mask on the list of sensors list to collect data from is added as a safety measure for the UAV to avoid crashes due to battery depletion. The sensors requiring high energy are, then, masked.

The UAV learns to head, when it is possible, towards the sensing nodes and the charging station, by filling the $\boldsymbol{Q}$-table according to the reward function computed during the offline training phase. Hence, at its end, the $\boldsymbol{Q}$-table will include what is learned at each situation (i.e., the right choice for any battery level at any time step). Based on the final $\boldsymbol{Q}$-table value and to prevent the UAV from returning to an already served sensing node, the $\boldsymbol{Q}$-table's column that represents a served sensor as an action is removed from the $\boldsymbol{Q}$-table. The UAV will be able then to only consider the remaining non-served sensors and collect data during the time window of each sensor without violating the limited battery capacity constraint.

3) TRAINING PHASE

As mentioned earlier, the training phase of the QL-algorithm, illustrated in Fig. 10, is executed for *Ep* episodes where each one of them accounts for *I* iterations. At the beginning of each episode, the time is reset and the UAV is returned to w_0. At each iteration, the UAV is assumed to progress in time according to the exploration or exploitation decision. In exploration, an action is chosen randomly,

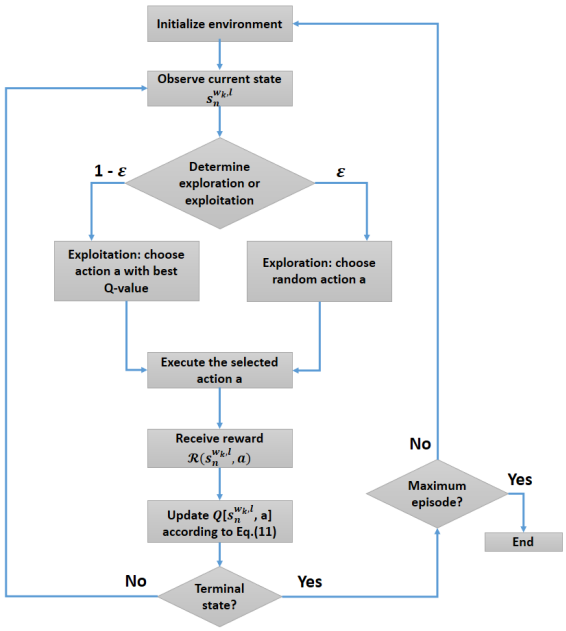

FIGURE 10. A block diagram describing the training phase of the proposed Q-learning algorithm for UAV data collection.

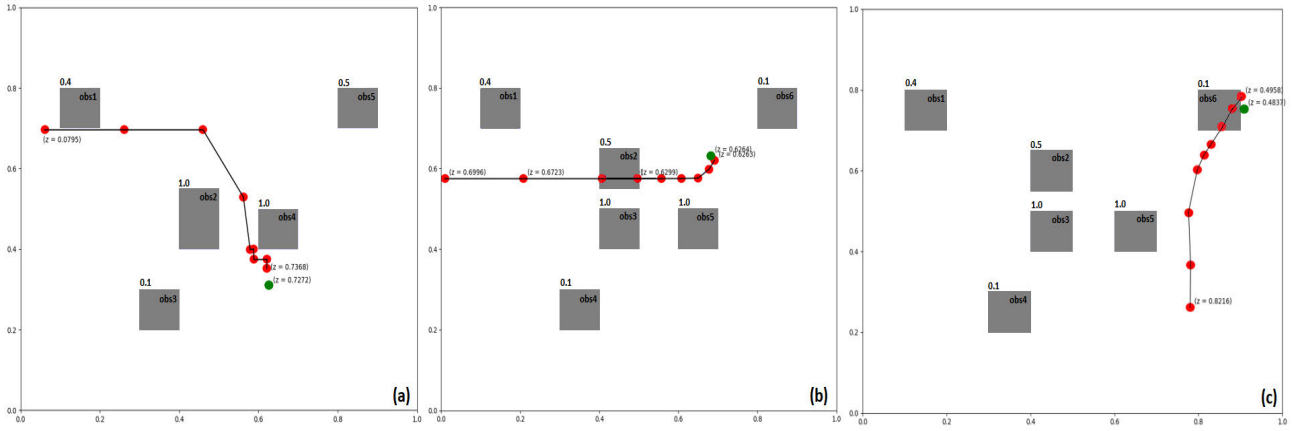

FIGURE 11. Path followed by UAV in three obstacle-constrained environments. Red dots refers to the UAV steps, green dot refers to the target, and gray boxes to the obstacles. (a) environment 1 (env1) and (b, c) two cases in environment 2 (env2).

while, in exploitation, the action is chosen such that the corresponding element in $\boldsymbol{Q}$ is maximized. To keep balance between exploration and exploitation actions, an ε-greedy action-selection policy is used with $(0 \leq \varepsilon \leq 1)$. At each progress step, one element of $\boldsymbol{Q}$ is updated as given in (18). A terminal state is reached when the maximum number of iterations I is achieved or the UAV fails to complete its mission due to energy depletion. In that case, a new episode starts.

IV. SIMULATION RESULTS

In this section, we investigate the performance of both RL frameworks including their training and testing phases. We aim to visualize the autonomous operation of the UAV while navigating in a 3D environment first and then in determining its scheduling.

A. AUTONOMOUS NAVIGATION

In this section, we study the behavior of the autonomous navigation framework for selected scenarios. We also visualize its efficiency in terms of crash rate and safe navigation accomplishment. To do so, we assume that the UAV starting location d_1, its target location d_2, and the obstacles' parameters are randomly generated within a cube-shaped area with 100 m edge length. We make sure that the locations of both the target and the UAV are outside the obstacles. The rest of the simulation parameters of the navigation framework are set in Table 1. The simulations are executed using Python in normalized 3D environments. For the sake of clarity, the figures concerning the UAV path planning are presented in only 2D dimension area (i.e., plan(x,y)) and we provide beside each dot, the altitude of either the target or the UAV.

In Fig. 11, we plot the trajectory adopted by the UAV to reach its destination using the autonomous navigation framework. In Fig. 11(a), the UAV successfully reached its destination location while avoiding the obstacles. In Fig. 11(b), on its way to the destination, the UAV crossed over $obs2$ $(z_d(4) \approx 0.63 > h_{obs2} = 0.5)$ in order to reach faster its target location unlike the case in Fig. 11(a), where the UAV could not cross over $obs2$ because of the obstacle height. The UAV is set to not fly at an altitude higher than that. In Fig. 11(c), having a higher altitude than $obs6$, the UAV crossed over $obs6$ to reach its target. In all cases, scenarios show some lacking in precision to reach the target location due to the fact of using infinite action space which makes it hard to get pinpoint accuracy. These scenarios show that the UAV successfully learned how to avoid obstacles to reach its destination for different scenarios.

In Fig. 12, we present the reward received by the UAV during its training phase. Fig. 12(a) shows that the UAV learns to obtain the maximum reward value in an obstacle-free environment. We successfully obtained a trained model capable of reaching targets in 3D environment with continuous action space. Then, using the knowledge gathered by the first training, we trained the model to be able to avoid obstacles. Fig. 12(b) shows that the UAV model has converged and

TABLE 1. Simulation parameters of the navigation framework.

Parameter	Value	Parameter	Value
β	4	ν	0.99
$\rho_{\max}$	0.2 m	T	100
M	40000	ϵ_{end},ϵ_{start}	0.1,0.9
B	10000	N	256

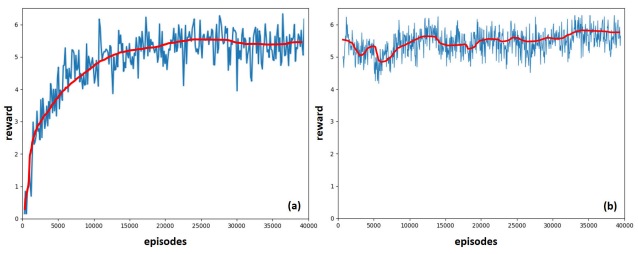

FIGURE 12. The reward received by the UAV during the training phase: (a) the source task (obstacle-free) and (b) the environment with obstacles.

reached the maximum possible reward value. Notice that during the first 10000 episodes, fluctuations in the reward can be noticed. Indeed, the RL model is adapting itself to the obstacle-constrained environment. These fluctuations change from an environment to another according to the density, locations, and heights of the obstacles.

During the testing phase and as shown in Table 2, for the obstacle-free environment, the UAV successfully reached its target for the tested cases, 100% success rate for 1000 test cases. As for the environment with obstacles, in the case of env1, the UAV successfully reached its target safely for 84% of the 1000 tested scenarios and in the case of env2, the reached its target safely for 82% of the 1000 tested scenarios.

TABLE 2. Task-completion rate.

Scenarios	obstacle free	obstacles env1	obstacles env2
Completion rate	100%	84%	82%

B. AUTONOMOUS SCHEDULING

In this section, we study the behavior of the autonomous scheduling framework for selected scenarios. Also, a comparison between the performance of the QL-based approach and an optimal MILP-based solution is provided [44]. For the sake of clarity and to be able to visualize the behavior of the UAV, we assume that $K = 5$ sensing nodes are randomly generated locations within 5×5 km^2 area. Nevertheless, the framework is able to provide results with a higher dimension of sensor nodes. In practice, to deal with the increased complexity of the training phase, a space-time division can be considered. For instance, we can divide a large geographical area into multiple subareas and treat each of them successively and separately. Also, the time horizon can be divided into multiple sub-periods, and each sub-period can be treated separately. In our simulations, we place the charging station w_0 is the center of the geographical area. The data collection takes place all along a makespan $\Gamma = 60$ minutes. The UAV's average speed V_d is fixed at 10 (m/s) and the average interference level $\Omega = -83$ dBm [45] as for the used power model parameters are given in Table 3. The total bandwidth B is set to 0.2 MHz while the noise density N_0 is approximately equal to -120 dBm. We investigate the scenario where the UAV has to collect the messages from all scattered sensor nodes as illustrated in Fig. 13.

TABLE 3. Power model parameters.

Notation	Physical meaning	Simulation Value
ρ	Air density in kg/m^3	1.225
A	Rotor disc area in m^2	0.503
U_{tip}	Tip speed of the rotor blade (m/s)	120
s	Rotor solidity $= \frac{\text{total blade area}}{\text{disc area}}$	0.05
d_0	Fuselage drag ratio	0.6
v_0	Mean rotor induced velocity in hover	4.03

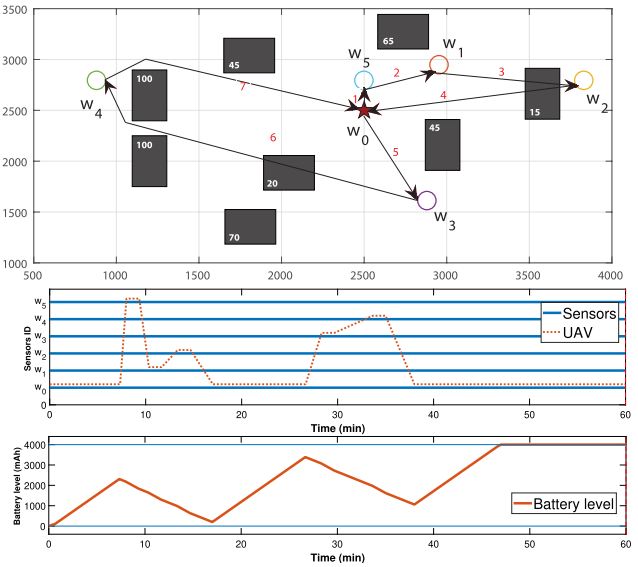

FIGURE 13. Scenario A: (a) UAV data collection path plan for five spatially distributed sensing nodes located in 5 × 5 km^2, (b) autonomous scheduling with limited battery capacity $\bar{b} = 4000$ mAh, and (c) the evolution of the UAV battery during the UAV trip.

In Scenario A, we consider $K = 5$ sensor nodes, having message size $m_k = 10$ Megabytes to be collected. The nodes are assumed to send a high amount of data, e.g., recorded videos or all data collected from neighborhood sensors.[2] Fig. 13 depicts the behavior of the UAV for Scenario A where it successfully collects the data from all sensors in 38 minutes. In the beginning, the UAV, having an empty battery, stays at the charging station w_0 until minute 7.3 to charge its battery till 2310 mAh in order to have enough energy to start roaming. Therefore, the flying unit heads toward sensor nodes w_5, w_1, and w_2, respectively. Then, it returns to the charging station, situated at loc_{w_0}, to charge its battery till it has enough energy to collect messages from the remaining sensor nodes w_3 and w_4 before returning to the charging station. The UAV successfully learned how to collect all the messages from the sensing nodes while respecting the battery limit.

In Scenario B (Fig. 14), we associate to the sensor w_3 a higher priority level than its peers. Hence, the UAV adjusts its trip plan compared to Scenario A. The UAV stays for 5.33 minutes at the charging station till it has enough energy, 1650 mAh, to head toward sensor w_3 to cover it first, as it has the highest priority, then collects data from sensor w_5. Afterward returns to w_0 to charge its depleted battery. Next, the flying unit collects data from sensors w_1, w_2, and w_3, respectively, with a return to the charging station in-between to avoid crashes. The UAV successfully learned how to collect all messages from the sensors while respecting the priority of the sensors and the battery limit.

[2]The reason for choosing large messages in our simulations is also for visualization purposes. Indeed, with low messages, the communication time will be very low and the UAV will not be seen statically hovering to collect the messages.

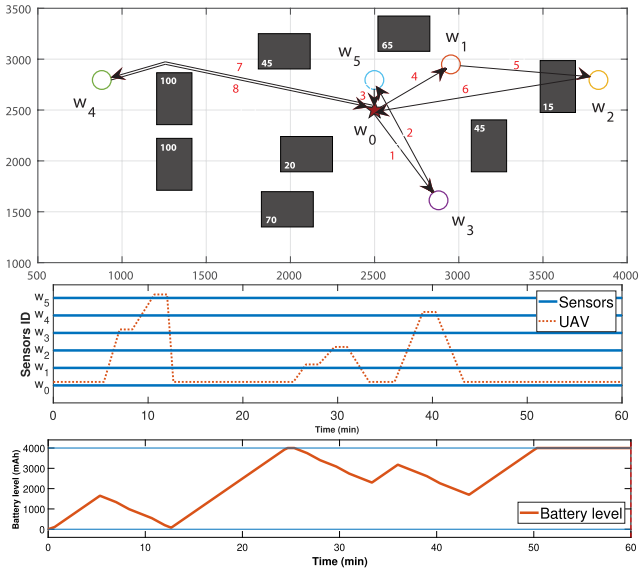

FIGURE 14. Scenario B: (a) UAV data collection path plan, (b) autonomous scheduling with limited battery capacity $\bar{b} = 4000$ mAh, and (c) the evolution of the UAV battery during the UAV trip. In this Scenario, sensor node w_3 has a priority level higher than its peers.

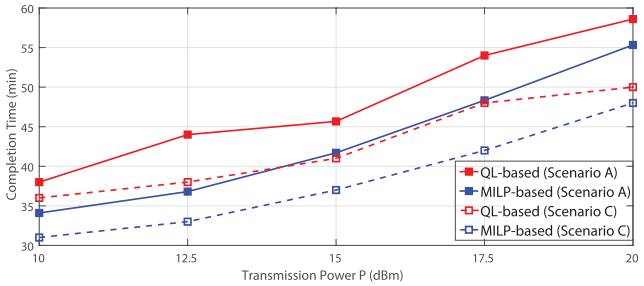

FIGURE 15. Tour completion time of the UAV as a function of the ground unit transmission power P^{tr} for two scenarios A and C. Blue line: MILP-based optimal solution and red line: proposed QL approach.

In Fig. 15, we present the required time to collect data from all the sensor nodes versus the ground unit transmission power P^{tr} for Scenario A and another Scenario C (same scenario as in A, but the messages size of sensors w_3 and w_4 are reduced to $m_k = 4$ Megabytes). We notice that for both scenarios, decreasing P^{tr} leads to higher data collection time. This is due to not only the decrease of the transmission rate but also to the more frequent battery reload since more hovering energy is needed. It is also shown that, in Scenario C, decreasing the message size makes the data collection procedure faster and that is due to lower transmission time. The figure also compares the performance of the QL approach with an optimal MILP based solution, i.e., a learning technique versus a deterministic one. The QL solution achieves a result close to the optimal with about 10% difference in the tour completion time on average. The behavior of the autonomous UAV for the investigated scenario corroborates the ability of QL to handle the data gathering problem efficiently.

V. CONCLUSION

In this paper, we have developed a UAV-based data collection approach for delay-tolerant applications in wireless sensor networks. The proposed approach is modeled based on a joint combination of two reinforcement learning techniques. The first one employs the DDPG model to enable autonomous navigation in an obstacle-constrained environment. It delivers the trajectory to adopt by the UAV to go from a position to another in the 3D map. The second RL model uses the output of the navigation framework to determine a scheduling of the UAV in order to accomplish its collection tour as fast as possible while considering the stored energy level, the need to return to the charging station to reload the battery, and the data acquisition time of the sensors. The UAV makes on-spot decisions to gather the messages from all the scattered sensor nodes within a predefined time horizon. In this study, the navigation decisions are made according to a continuous action space allowing an accurate avoidance of the obstacles. Moreover, a reward function for the scheduling framework is designed in order to maintain the UAV safety from crashes due to energy depletion. Through simulations, we have shown the effective operation of the UAV in autonomously determining its schedule and trip plan without involving external entities. It is also shown that the proposed scheduling approach is able to achieve close performance to the optimal deterministic solution.

As future work, we plan to study the multi-agent problem where a fleet of UAVs is simultaneously employed to gather data. We also aim to focus on designing an RL-based solution to allow the UAV to autonomously learn where to stop to communicate with the ground sensors. It is also worthy to design a solution dealing with multiple moving ground sensors.

ACKNOWLEDGMENT

This article was presented in part at the IEEE International Symposium on Circuits and Systems (ISCAS'20).

REFERENCES

[1] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "Autonomous UAV navigation: A DDPG-based deep reinforcement learning approach," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Seville, Spain, Oct. 2019. [Online]. Available: https://arxiv.org/abs/2003.10923

[2] K. Kuru, D. Ansell, W. Khan, and H. Yetgin, "Analysis and optimization of unmanned aerial vehicle swarms in logistics: An intelligent delivery platform," *IEEE Access*, vol. 7, pp. 15804–15831, Jan. 2019.

[3] O. Thakoor, J. Garg, and R. Nagi, "Multiagent UAV routing: A game theory analysis with tight price of anarchy bounds," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 100–116, Jan. 2020.

[4] X. He, J. R. Bourne, J. A. Steiner, C. Mortensen, K. C. Hoffman, C. J. Dudley, B. Rogers, D. M. Cropek, and K. K. Leang, "Autonomous chemical-sensing aerial robot for urban/suburban environmental monitoring," *IEEE Syst. J.*, vol. 13, no. 3, pp. 3524–3535, Sep. 2019.

[5] S. Say, H. Inata, J. Liu, and S. Shimamoto, "Priority-based data gathering framework in UAV-assisted wireless sensor networks," *IEEE Sensors J.*, vol. 16, no. 14, pp. 5785–5794, Jul. 2016.

[6] H. Baek and J. Lim, "Design of future UAV-relay tactical data link for reliable UAV control and situational awareness," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 144–150, Oct. 2018.

[7] J. Zong, C. Shen, J. Cheng, J. Gong, T.-H. Chang, L. Chen, and B. Ai, "Flight time minimization via UAV's trajectory design for ground sensor data collection," in *Proc. 16th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2019, pp. 255–259.

[8] T. Li, W. Liu, T. Wang, Z. Ming, X. Li, and M. Ma, "Trust data collections via vehicles joint with unmanned aerial vehicles in the smart Internet of Things," *Trans. Emerg. Telecommun. Technol.*, p. e3956, Apr. 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3956, doi: 10.1002/ett.3956.

[9] B. Jiang, G. Huang, T. Wang, J. Gui, and X. Zhu, "Trust based energy efficient data collection with unmanned aerial vehicle in edge network," *Trans. Emerg. Telecommun. Technol.*, p. e3942, Mar. 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3942

[10] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 36–42, May 2016.

[11] H. Ghazzai, H. Menouar, and A. Kadri, "On the placement of UAV docking stations for future intelligent transportation systems," in *Proc. IEEE 85th Veh. Technol. Conf. (VTC Spring)*, Sydney, NSW, Australia, Jun. 2017, pp. 1–6, doi: 10.1109/VTCSpring.2017.8108676.

[12] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48572–48634, 2019.

[13] A. Bahabry, X. Wan, H. Ghazzai, G. Vesonder, and Y. Massoud, "Collision-free navigation and efficient scheduling for fleet of multi-rotor drones in smart city," in *Proc. IEEE 62nd Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Dallas, TX, USA, Aug. 2019, pp. 552–555.

[14] D. Wang and Y. Yang, "Joint obstacle avoidance and 3D deployment for securing UAV-enabled cellular communications," *IEEE Access*, vol. 8, pp. 67813–67821, 2020.

[15] M. Huang, K. Zhang, Z. Zeng, T. Wang, and Y. Liu, "An AUV-assisted data gathering scheme based on clustering and matrix completion for smart ocean," *IEEE Internet Things J.*, early access, Apr. 2020, doi: 10.1109/JIOT.2020.2988035.

[16] D. Popescu, F. Stoican, L. Ichim, G. Stamatescu, and C. Dragana, "Collaborative UAV-WSN system for data acquisition and processing in agriculture," in *Proc. 10th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst. Technol. Appl. (IDAACS)*, Metz, France, vol. 1, Sep. 2019, pp. 519–524.

[17] D. Ebrahimi, S. Sharafeddine, P.-H. Ho, and C. Assi, "UAV-aided projection-based compressive data gathering in wireless sensor networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1893–1905, Apr. 2019.

[18] J. Grigulo and L. B. Becker, "Experimenting sensor nodes localization in WSN with UAV acting as mobile agent," in *Proc. IEEE 23rd Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Turin, Italy, Sep. 2018, pp. 808–815.

[19] X. Ma, R. Kacimi, and R. Dhaou, "Fairness-aware UAV-assisted data collection in mobile wireless sensor networks," in *Proc. Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Paphos, Cyprus, Sep. 2016, pp. 995–1001.

[20] J. Mi, X. Wen, C. Sun, Z. Lu, and W. Jing, "Energy-efficient and low package loss clustering in UAV-assisted WSN using Kmeans++ and fuzzy logic," in *Proc. IEEE/CIC Int. Conf. Commun. Workshops China (ICCC Workshops)*, Aug. 2019, pp. 210–215.

[21] S. K. Haider, M. A. Jamshed, A. Jiang, H. Pervaiz, and Q. Ni, "UAV-assisted cluster-head selection mechanism for wireless sensor network applications," in *Proc. UK/ China Emerg. Technol. (UCET)*, Glasgow, U.K., Aug. 2019, pp. 1–2.

[22] C. You and R. Zhang, "3D trajectory optimization in Rician fading for UAV-enabled data harvesting," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3192–3207, Jun. 2019.

[23] M. B. Ghorbel, D. Rodríguez-Duarte, H. Ghazzai, M. J. Hossain, and H. Menouar, "Joint position and travel path optimization for energy efficient wireless data gathering using unmanned aerial vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2165–2175, Mar. 2019.

[24] A. T. Albu-Salih and S. A. H. Seno, "Energy-efficient data gathering framework-based clustering via multiple UAVs in deadline-based WSN applications," *IEEE Access*, vol. 6, pp. 72275–72286, Nov. 2018.

[25] H. Binol, E. Bulut, K. Akkaya, and I. Guvenc, "Time optimal multi-UAV path planning for gathering its data from roadside units," in *Proc. IEEE 88th Veh. Technol. Conf. (VTC-Fall)*, Chicago, IL, USA, Aug. 2018, pp. 1–5.

[26] A. Bahabry, X. Wan, H. Ghazzai, H. Menouar, G. Vesonder, and Y. Massoud, "Low-altitude navigation for multi-rotor drones in urban areas," *IEEE Access*, vol. 7, pp. 87716–87731, 2019.

[27] J. Baek, S. I. Han, and Y. Han, "Optimal UAV route in wireless charging sensor networks," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1327–1335, Feb. 2020.

[28] J. Baek, S. I. Han, and Y. Han, "Energy-efficient UAV routing for wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1741–1750, Feb. 2020.

[29] P. Wu, F. Xiao, C. Sha, H. Huang, and L. Sun, "Trajectory optimization for UAVs' efficient charging in wireless rechargeable sensor networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4207–4220, Apr. 2020.

[30] C. H. Liu, Z. Chen, and Y. Zhan, "Energy-efficient distributed mobile crowd sensing: A deep learning approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1262–1276, Jun. 2019.

[31] S. Islam and A. Razi, "A path planning algorithm for collective monitoring using autonomous drones," in *Proc. 53rd Annu. Conf. Inf. Sci. Syst. (CISS)*, Baltimore, MD, USA, Mar. 2019, pp. 1–6.

[32] C. Yan and X. Xiang, "A path planning algorithm for UAV based on improved Q-Learning," in *Proc. 2nd Int. Conf. Robot. Autom. Sci. (ICRAS)*, Wuhan, China, Jun. 2018, pp. 1–5.

[33] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "Q-learning based routing scheduling for a multi-task autonomous agent," in *Proc. IEEE 62nd Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Dallas, TX, USA, Aug. 2019, pp. 634–637.

[34] V. N. Sichkar, "Reinforcement learning algorithms in global path planning for mobile robot," in *Proc. Int. Conf. Ind. Eng., Appl. Manuf. (ICIEAM)*, Sochi, Russia, Mar. 2019, pp. 1–5.

[35] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "A generic spatiotemporal scheduling for autonomous UAVs: A reinforcement learning-based approach," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 93–106, 2020.

[36] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Commun. Lett.*, vol. 3, no. 6, pp. 569–572, Dec. 2014.

[37] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.

[38] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction* (Adaptive Computation and Machine Learning). Cambridge, MA, USA: MIT Press, 2018.

[39] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[40] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. ICLR*, 2016. [Online]. Available: https://arxiv.org/abs/1509.02971

[41] R. Bellman, *Dynamic Programming* (Dover Books on Computer Science). New York, NY, USA: Dover, 2013.

[42] M. Ji, L. Zhang, and S. Wang, "A path planning approach based on Q-learning for robot arm," in *Proc. 3rd Int. Conf. Robot. Autom. Sci. (ICRAS)*, Wuhan, China, Jun. 2019, pp. 15–19.

[43] R. Munos, T. Stepleton, A. Harutyunyan, and M. Bellemare, "Safe and efficient off-policy reinforcement learning," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. New York, NY, USA: Curran Associates, Inc., 2016, pp. 1054–1062. [Online]. Available: http://papers.nips.cc/paper/6538-safe-and-efficient-off-policy-reinforcement-learning.pdf

[44] H. Ghazzai, A. Kadri, M. B. Ghorbel, H. Menouar, and Y. Massoud, "A generic spatiotemporal UAV scheduling framework for multi-event applications," *IEEE Access*, vol. 7, pp. 215–229, Jan. 2019.

[45] V. Yajnanarayana, Y.-P. E. Wang, S. Gao, S. Muruganathan, and X. L. Ericsson, "Interference mitigation methods for unmanned aerial vehicles served by cellular networks," in *Proc. IEEE 5G World Forum (5GWF)*, Silicon Valley, CA, USA, Jul. 2018, pp. 118–122.

OMAR BOUHAMED (Student Member, IEEE) received the National Engineering Diploma degree (Hons.) in telecommunications from the Higher School of Communication of Tunis (SUP'COM), University of Carthage, Tunisia, in 2019. He was a Research Assistant with the Stevens Institute of Technology, Hoboken, NJ, USA, in 2019. His general research interests include the intersection of machine learning, UAVs, optimization, and the Internet of Things.

HAKIM GHAZZAI (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from KAUST, Saudi Arabia, in 2015, and the Diplome d'Ingenieur degree (Hons.) in telecommunication engineering and the master's degree in high-rate transmission systems from the Ecole Superieure des Communications de Tunis (SUP'COM), Tunis, Tunisia, in 2010 and 2011, respectively. He was a Visiting Researcher with Karlstad University, Sweden, and a Research Scientist with the Qatar Mobility Innovations Center (QMIC), Doha, Qatar, from 2015 to 2018. He is currently a Research Scientist with the Stevens Institute of Technology, Hoboken, NJ, USA. His general research interests include the intersection of wireless networks, UAVs, the Internet of Things, intelligent transportation systems, and optimization. Since 2019, he has been on the Editorial Board of the IEEE Communications Letters and the IEEE Open Journal of the Communications Society.

HICHEM BESBES received the Ph.D. degree in electrical engineering from ENIT, University EL MANAR, Tunisia, in 1999, the Diplome d'Etudes Appronfondies (DEA) (master's diploma) degree in systems theory from ENIT, in 1991, the Engineering Diploma degree in electrical engineering from ENIT, in 1991, and the Habilitation à Diriger des Recherches (HDR) degree in telecommunications from the Ecole Superieure des Communications de Tunis (SUP'COM), in June 2005. He was the former Head of the Department of Applied Mathematics and Signal Processing, Higher School of Engineering in Communication in Tunisia (Sup'Com). He has more than 25 years of experience in higher education and ICT. He was a Postdoctoral Fellow of Concordia University, from 1999 to 2000, and a Visiting Scholar with Colorado State University, in 2011. He has served as a Senior Engineer and a Member of Technical Staff at Legerity Inc., and Celite Systems Inc., Austin, TX, USA, working on xDSL technologies. He is currently a Professor of wireless communication with the University of Carthage, the former President, the Chairman, and a member of the Board of the National Telecommunication Regulatory Authority of Tunisia (INTT), for more than five years. He has published more than 120 scientific articles in international journals and conferences dealing with wireless communications. He holds one U.S. patent, and he supervised nine Ph.D. thesis. Under his leadership, the INTT was elected as the Best African Regulator, in 2015, and he was nominated for the GSMA Government Leadership Award, in 2018.

YEHIA MASSOUD (Fellow, IEEE) received the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA. He is currently the Dean of the School of Systems and Enterprises, Stevens University of Science and Technology, Hoboken, NJ, USA. He has authored over 280 articles in peer-reviewed journals and conferences. Dr. Massoud has served as a Distinguished Lecturer by the IEEE Circuits and Systems Society and as an Elected Member of the IEEE Nanotechnology Council. He was selected as one of ten MIT Alumni Featured by MIT's Electrical Engineering and Computer Science department in 2012. He was a recipient of the Rising Star of Texas Medal, in 2007, the National Science Foundation CAREER Award, in 2005, the DAC Fellowship, in 2005, the Synopsys Special Recognition Engineering Award, in 2000, several best paper award nominations, and two best paper awards at the IEEE International Symposium on Quality Electronic Design, in 2007, and the IEEE International Conference on Nanotechnology, in 2011. He has held several academic and industrial positions, including a member of the Technical Staff at the Advanced Technology Group, Synopsys, Inc., Mountain View, CA, USA, a Tenured Faculty at the Departments of Electrical and Computer Engineering and Computer Science, Rice University, Houston, TX, USA, the W. R. Bunn Head of the Department of Electrical and Computer Engineering, The University of Alabama, Birmingham, AL, USA, and the Head of the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, USA. He has served as the Editor for Mixed-Signal Letters—The Americas and also as an Associate Editor for the IEEE Transactions on Very Large Scale Integration Systems and the IEEE Transactions on Circuits and Systems I: Regular Papers.