# Continuous Maneuver Control and Data Capture Scheduling of Autonomous Drone in Wireless Sensor Networks

Early Access

**Kai Li***
**Wei Ni**
**Falko Dressler**

*CISTER Research Centre

CISTER-TR-210101

2021/01/05

# Continuous Maneuver Control and Data Capture Scheduling of Autonomous Drone in Wireless Sensor Networks

Kai Li*, Wei Ni, Falko Dressler

*CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: kai@isep.ipp.pt, Wei.Ni@data61.csiro.au, dressler@ccs-labs.org

https://www.cister-labs.pt

## Abstract

Thanks to flexible deployment and excellent maneuverability, autonomous drones are regarded as an effective means to enable aerial data capture in large-scale wireless sensor networks with limited to no cellular infrastructure, e.g., smart farming in a remote area. A key challenge in drone-assisted sensor networks is that the autonomous drone's maneuvering can give rise to buffer overflows at the ground sensors and unsuccessful data collection due to lossy airborne channels. In this paper, we propose a new Deep Deterministic Policy Gradient based Maneuver Control (DDPG-MC) scheme which minimizes the overall data packet loss through online training instantaneous headings and patrol velocities of the drone, and the selection of the ground sensors for data collection in a continuous action space. Moreover, the maneuver control of the drone and communication schedule is formulated as an absorbing Markov chain, where network states consist of battery energy levels, data queue backlogs, timestamps of the data collection, and channel conditions between the ground sensors and the drone. An experience replay memory is utilized onboard at the drone to store the training experiences of the maneuver control and communication schedule at each time step.

# Continuous Maneuver Control and Data Capture Scheduling of Autonomous Drone in Wireless Sensor Networks

Kai Li, *Senior Member, IEEE*, Wei Ni, *Senior Member, IEEE*, and Falko Dressler, *Fellow, IEEE*

**Abstract**—Thanks to flexible deployment and excellent maneuverability, autonomous drones are regarded as an effective means to enable aerial data capture in large-scale wireless sensor networks with limited to no cellular infrastructure, e.g., smart farming in a remote area. A key challenge in drone-assisted sensor networks is that the autonomous drone's maneuvering can give rise to buffer overflows at the ground sensors and unsuccessful data collection due to lossy airborne channels. In this paper, we propose a new deep deterministic policy gradient based maneuver control (DDPG-MC) scheme which minimizes the overall data packet loss through online training instantaneous headings and patrol velocities of the drone, and the selection of the ground sensors for data collection in a continuous action space. Moreover, the maneuver control of the drone and communication schedule is formulated as an absorbing Markov chain, where network states consist of battery energy levels, data queue backlogs, timestamps of the data collection, and channel conditions between the ground sensors and the drone. An experience replay memory is utilized onboard at the drone to store the training experiences of the maneuver control and communication schedule at each time step. Numerical results demonstrate that the proposed DDPG-MC achieves 15.2 and 47.6 percent lower packet loss rate than deep Q-learning-based flight control and non-learning scheduling policies, respectively.

**Index Terms—**Autonomous drone, maneuver control, data collection, deep reinforcement learning, absorbing markov chain

## 1 Introduction

Recent advances of wireless sensing techniques allow for deploying a large number of sensing devices for sustainable environmental monitoring [1], [2]. Sensory data are generated and stored in a data queue at the sensor, awaiting to be uploaded to a remote base station. Data collection in large-scale wireless sensor networks is difficult since sensors can be airlifted to remote, human-unfriendly environments, e.g., disaster stricken areas, rural vineyards, or battlefields [3]. In such harsh environments, conventional terrestrial communication networks requiring persistent power supplies are unavailable or unreliable. Thanks to their flexible deployment and excellent maneuverability, autonomous drones provide an effective means to collect data from the ground sensors, offload command or software patch, or restore communications [4], [5]. The drone can move sufficiently close to a ground sensor, leveraging a dominant line-of-sight (LoS) between the drone and the sensor [6]. The drone maneuver can enhance the network coverage while the LoS link enables a high data rate for the drone-sensor communications. Several international initiatives have been launched to study the feasibility of using drones for providing wireless access for ground sensor networks. For example, SoftBank company partnered with NASA and U.S. aerospace company AeroVironment developed a high-altitude autonomous drone to provide communication connectivity from the sky [7]. Optus and Ericsson delivered Australia's first 5G teleoperated drone controlled over a live 5G network to track and identify objects [8]. Verizon tested different types of drones to improve network connectivity [9]. The 3rd Generation Partnership Project (3GPP) studied capability of the Long Term Evolution (LTE) to support drones [10].

Fig. 1 illustrates an application of drone-assisted sensor networks for precision agriculture. Specifically, a large number of energy harvesting powered sensors are deployed in a vineyard for sensing and monitoring temperature, soil moisture, and illumination time. The ground sensor can be equipped with solar panels, wind power generators, or wireless power receiver to harvest renewable energy from ambient resources for opportunistically recharging its battery [11], [12], [13]. A drone equipped with a wireless radio and onboard data processors hovers over the vineyard. The drone is typically powered by batteries, which leads to a finite cruising time. Moreover, the heading and patrol velocity of the drone can adaptively change and select the ground sensors for data collection along the flight trajectory.

A low battery level of the ground sensor can potentially prevent data inside the finite buffers from being transmitted in time, hence resulting in the overflow of the buffers upon

- *Kai Li is with the Real-Time and Embedded Computing Systems Research Centre (CISTER), 4249015 Porto, Portugal. E-mail: kai@isep.ipp.pt.*
- *Wei Ni is with the Digital Productivity and Services Flagship, Commonwealth Scientific and Industrial Research Organization (CSIRO), Sydney, NSW 2122, Australia. E-mail: wei.ni@data61.csiro.au.*
- *Falko Dressler is with the School of Electrical Engineering and Computer Science, Technical University of Berlin, 10623 Berlin, Germany. E-mail: dressler@ccs-labs.org.*

*Manuscript received 4 June 2020; revised 19 Nov. 2020; accepted 21 Dec. 2020. Date of publication 5 Jan. 2021; date of current version 1 July 2022. (Corresponding author: Kai Li.) Recommended for acceptance by J. Wu. Digital Object Identifier no. 10.1109/TMC.2021.3049178*

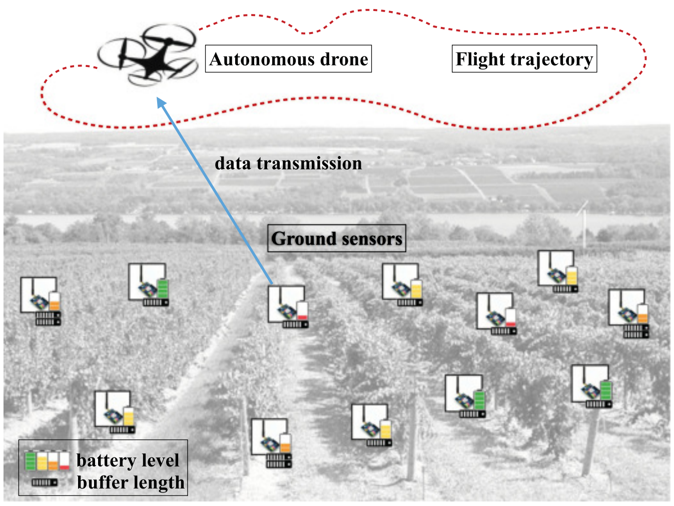


Fig. 1. A large number of sensors are deployed in a vineyard for precision agriculture. The autonomous drone adjusts the heading and the patrol velocity to maneuver over the target field, while selecting the ground sensors to transmit data.

the arrivals of new data. Specifically, the newly arrived data packets are generally queued in the buffer of a ground sensor, until the battery level of the ground sensor is sufficient to complete the transmissions of the earlier packets and power the transmissions of the newly arrived packets. The battery of the ground sensor is recharged by harvesting renewable energy from solar, wind, or other renewable energy sources. The energy harvesting depends heavily on the environmental conditions. For example, a cloudy or rainy weather, or a windless day would result in a small amount of harvested energy [14].

Despite their consistent sampling intervals, the sensory data arrivals at the transmit buffer of the ground sensors can have large variations. This is because in many cases, only changes get reported to reduce the communication overhead of a system and the energy requirement of the transmitter [15]. In many other cases, the packet generation of the ground sensors can be event-triggered, e.g., wildlife camera traps which shoot photos only when their infrared sensors are triggered by passing animals. In the above cases, it is reasonable to assume that the ground sensors undergo random data arrivals (at their transmit buffers). Some sensors may periodically generate packets. The periodic packet arrivals in the data queue are predictable; while the battery energy levels and channel conditions still experience time-varying randomness, depending on the environments.

Selecting a ground sensor for data collection may result in a buffer overflow at other unselected sensors, since new data arrivals at those sensors may have to be dropped if their buffers are already full and overflow. Despite memory chips and storage capacity of sensors have been continuously improving, the data buffer can still overflow for two reasons. First, according to queueing theory, a queue grows infinitely, as long as the incoming data rate into the queue is higher than the outgoing data rate from the queue. In the situation where the network has too many sensors to get their data collected in a timely manner or the drone's trajectory is poorly planned, the queues would build up and the sensors would suffer from buffer overflows. Second, there is a trend of increasingly large data sizes being exported in emerging sensing platforms, e.g., weed identification or wildlife monitoring with cameras [16], [17], and insect detection based on optoacoustic sensors [18]. The typical sizes of high-resolution images and acoustic data are several megabytes. However, many off-the-shelf sensors have a limited data memory with the consideration of the cost and market competitiveness.

Moreover, selecting a ground sensor with a poor channel condition gives rise to packet errors of the transmissions or buffer overflows at other sensors. In practice, the instantaneous knowledge of the battery energy levels, data queue backlogs, and channel conditions of the ground sensors is not available at the drone. Therefore, the joint optimization of the maneuver control of the drone and the selection of the ground sensors is crucial to minimize packet losses resulting from buffer overflows and fading channels in the drone-assisted sensor network.

In this paper, we investigate the continuous maneuver control and data capture scheduling of autonomous drone in wireless sensor networks. The main contributions can be summarized as follows:

1) To the best of our knowledge, this is the first attempt to investigate the joint optimization of the continuous maneuver control of an autonomous drone and the communication schedule to minimize the data loss. The drone-assisted data collection in wireless sensor networks is formulated as an absorbing Markov chain, where the network states consist of the battery energy levels, data queue backlogs, Time-To-be-Alive (TTA) values, and the channel conditions between the ground sensors and the drone.
2) An onboard Deep Deterministic Policy Gradient based Maneuver Control (DDPG-MC) is proposed to optimize the continuous maneuver control of the drone, which is typically with large state and action spaces. The onboard DDPG-MC jointly optimizes the online maneuver control and communication schedule through online training actions of the drone, i.e., the instantaneous headings, patrol velocities, and the real-time selection of the transmitting ground sensors. An experience replay memory is utilized to store the training experiences of the maneuver control and communication schedule at each time step, which stabilizes the training of DDPG-MC and improves sample efficiency by repeatedly reusing experience tuples.
3) To verify our design, we implement DDPG-MC in Python 3.5 running on top of Google TensorFlow. Numerical results demonstrate that the proposed DDPG-MC achieves at least 47.6 percent reduction in the overall packet loss, as compared to existing non-learning heuristics.

The rest of this paper is organized as follows. Section 2 reviews the related work on the trajectory planning of drones and communication scheduling schemes. Section 3 presents the flight model of autonomous drones and the channel model. The joint optimization of the maneuver control and communication schedule is formulated as the absorbing Markov chain in Section 4. In Section 5, a new onboard DDPG-MC scheme is designed to optimize the decision process of the absorbing Markov chain, thereby optimizing the headings and patrol velocities, as well as the transmission schedule of the ground sensors. Numerical results are presented in Section 6. Section 7 concludes the paper.

## 2 Related Work

### 2.1 Trajectory Planning

A trajectory planning algorithm is presented in [19] to reduce the communication delay of data collection. Given the predetermined waypoints, the radius of the trajectory is adjusted to alleviate data traffic congestion according to data buffer occupancy at the drone. The communication delay between the drone and the ground nodes can be reduced via the trajectory planning and the communication scheduling [20]. The propulsion energy of the drone can be the dominant factor determining the throughput-energy tradeoff in air-ground communications. In [21], an energy tradeoff is studied, where the transmission energy reduction at the ground sensor is at an increasing cost of the propulsion energy at the drone. The energy tradeoff is characterized by a circular or straight-line trajectory in accordance with the transmit power allocation of the ground sensors. The authors of [22] present a trajectory planning algorithm based on the data uploading time and the elapsed time since the drone leaves the radio coverage of the sensor. It is shown that the trajectory corresponds to the shortest Hamiltonian path in the ground sensor network, where the distance between the two sensors indicates their inter-visit time. In [23], the trajectory planning is formulated as a mixed integer non-linear programming to reduce the average path loss between the drone and the ground sensor. The trajectory planning is decoupled between multiple sub-problems which separately schedule the ground sensors' transmissions, trajectories, and altitudes of the drone.

In [24], drones provide emergent wireless coverage to a remote area. The deployment time of the drone depends on the velocity, altitude and radio coverage radius. The drone deployment algorithm is developed to reduce the deployment time given the same initial or different dispatching locations. A trajectory planning algorithm based on random tree generation is studied in [25] to avoid collisions with moving obstacles. The random tree generated by sampling the waypoints adds the trajectory with no collision to a graph as a candidate feasible path. The trajectory of the drone can also be designed with a continuous-time formulation for collision avoidance [26]. A replanning system is constructed from mapping to trajectory generation, where the trajectory responds to some previously unknown or unseen obstacles. In [27], a trajectory planning algorithm is developed for data sender localization. The waypoints are generated at the drone to reduce the localization uncertainty, while a maximum likelihood estimator estimates the data sender's location based on the ground sensor measurements.

In [28], a number of charging stations on the ground are uniformly deployed to satisfy the energy needs of the drone. The drone is assigned to serve the entire area in a sustainable way. The charging stations are allocated to charge the drone along its flight trajectory. In [29], the trajectory of a drone is designed to improve the energy efficiency of a point-to-point communication between the drone and a ground device, by taking into account the propulsion energy consumption of the drone. An algorithm is developed to maximize the energy efficiency, subject to the constraints on the drone's trajectory, including its initial/final locations and velocities, and maximum speed. The drone can adapt its displacement direction and distance to serve the ground users' wireless traffic [30]. The optimal displacement distance is designed to improve the average throughput for variable-rate applications and the success probability for fixed-rate applications.

### 2.2 Communication Scheduling

Energy harvesting drones are employed to extend network coverage and wireless access for ground sensors in [31]. The energy consumption of a drone is reduced by adapting the ground sensor assignment, the trajectory, and transmit power of the drone. In [32], the trajectory of the drone and the communication schedules are designed to improve network throughput of OFDMA users on the ground. Since the network throughput decreases with the increasing transmit rate of the OFDMA user, the throughput gain arising from the drone's mobility becomes less significant. The work in [33] focuses on network congestion prediction for drone-assisted data communications. A drone deployment algorithm is developed to reduce the transmit power of the drone, while reducing the propulsion energy based on the predicted network traffic. In the drone-assisted sensor network, the wake-up schedule of the ground sensors and the trajectory of the drone are jointly optimized to lower the energy consumption of the ground sensors [34]. The optimization also ensures the required amount of data collected from each ground sensor.

## 3 Autonomous Flight and Channel Model

In this section, we present the flight model of the autonomous drone and the channel model. A communication protocol for the drone-assisted data collection in wireless sensor networks is also studied. Specific notations used in this article are summarized in Table 1.

### 3.1 Flight Model of the Autonomous Drone

Let $(x(t), y(t), z)$ denote the position of the drone at time $t$. The drone is assumed to manoeuvre in an altitude hold mode [35], i.e., the altitude of the drone can be maintained steady. The instantaneous patrol velocity of the drone is $v(t)$, where $V_{min} < v(t) \leq V_{max}$. $V_{min}$ and $V_{max}$ are the minimum and the maximum velocities allowed, respectively. Let $\Delta t$ denote the flight duration from $(x(t), y(t), z)$ to $(x(t+1), y(t+1), z)$ and $\Delta v(t)/\Delta t = (v(t+1) - v(t))/\Delta t$ is the acceleration of the drone. Consider $V_{min} \leq v(t) \leq V_{max}$, where $V_{min}$ and $V_{max}$ are the minimum and the maximum velocities of the drone, respectively. The acceleration of the drone fulfills $0 \leq \Delta v(t)/\Delta t \leq (V_{max} - V_{min})/\Delta t$. For example, we set $V_{max}$ = 15 m/s, $V_{min} > 0$, and $\Delta t$ = 1 s, the acceleration of the drone $\Delta v(t)/\Delta t$ is within $[0, 15)$ m/s$^2$.

By applying the proposed DDPG-MC framework, $(x(t+1), y(t+1), z)$ and $v(t+1)$, are learned and optimized given

TABLE 1
The List of Fundamental Variables Defined in System Model

| Notation | Definition |
|---|---|
| $N$ | total number of ground sensors |
| $e_i(t)$ | battery energy level of ground sensor $i$ at time $t$ |
| $d_i(t)$ | data buffer length of ground sensor $i$ at time $t$ |
| $P_i(t)$ | transmit power of the ground sensor at time $t$ |
| $e_{\text{drone}}(t)$ | battery energy of the drone at time $t$ |
| $\theta(t)$ | turning angle of the drone at time $t$ |
| $v(t)$ | patrol velocity of the drone at time $t$ |
| $h_i(t)$ | channel condition between the drone and sensor $i$ at time $t$ |
| $\tau_i$ | TTA value of the ground sensor |
| $A$ | total number of absorbing states in the formulated Markov chain |
| $\alpha, \beta$ | network states of the formulated Markov chain |
| $\zeta_{\text{episode}}$ | random process for action exploration |
| $\delta$ | discount factor |
| $U_\alpha$ | the action taken by the drone at state $\alpha$ |
| $K$ | minibacth size of the experience replay |
| $M$ | number of episodes in the proposed DDPG-MC framework |

the location $(x(t), y(t), z)$ and the velocity $v(t)$ at the location. Given the current and the next locations and their associated speeds, and the current heading, the tangential acceleration $\Delta v(t)/\Delta t$ can be evaluated by satisfying $\Delta v(t)/\Delta t \leq (V_{max} - V_{min})/\Delta t$, the rotation center and radius can be specified, and the heading at the next location, i.e., $\theta(t+1)$, can be specified accordingly.

Fig. 2 describes the flight model of the drone, where $\theta(t)$ is the heading at $t$ [36], [37], and the coordinates of the circle centre are $(x_c(t), y_c(t), z)$. Therefore, the drone's location at time $t+1$ is given by

$$\begin{cases} x(t+1) = x_c(t) + \Big[(x(t) - x_c(t))\cos\theta(t) \\ \qquad - (y(t) - y_c(t))\sin\theta(t)\Big] \\ y(t+1) = y_c(t) + \Big[(x(t) - x_c(t))\sin\theta(t) \\ \qquad + (y(t) - y_c(t))\cos\theta(t)\Big], \end{cases} \tag{1}$$

where $\theta(t) \in (0, \pi]$. In particular, $\theta(t) = \pi$ indicates that the drone moves forward without changing the heading. It is assumed that the drone does not move backward, i.e., $\theta(t) \neq 0$.

The drone flies along a trajectory which consists of a large number of waypoints $(x(t), y(t), z)$. The instantaneous headings and patrol velocities of the drone can be adjusted online according to the proposed DDPG-MC framework. The drone also collects sensory data from the ground sensors. Beamforming is enabled at the drone to enhance the received signal strength (RSS) in both directions and reduce the bit error rate (BER) in the uplink.

The battery level of the autonomous drone is denoted by $e_{\text{drone}}(t)$, which can be measured by the onboard sensors. The drone has to suspend the cruise when the propulsion energy of the drone drops below the minimum energy level $e^{min}_{\text{drone}}$.

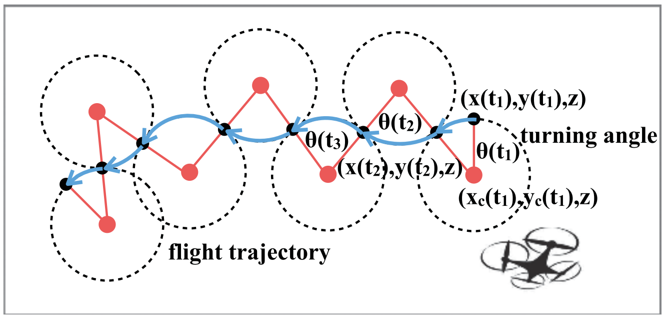


Fig. 2. The flight model of the autonomous drone.

## 3.2 Channel Model

We consider that $N$ ground sensors are deployed in a remote area. Sensor $i \in [1, N]$ can harvest renewable energy from the ambient environment to recharge its battery and power its operations, e.g., sensing, computing and communication. The battery level of sensor $i$ is denoted by $e_i(t) \leq E$, where $E$ is the battery capacity of the ground sensor. The data queue length of the ground sensor is $d_i(t) \in [1, D]$, where $D$ is the buffer size. In addition, the ground sensors undergo random data arrivals, and buffer the data to be collected by the drone. The buffers are finite, and the new data arrivals have to be dropped if the buffers are full and overflow.

The drone moves at a low altitude for data collection, where the probability of LoS between the drone and the ground sensors is given by [38]

$$\Pr_{\text{LoS}}(t) = \frac{1}{1 + a\exp(-b[\varphi_i(t) - a])}, \tag{2}$$

where $a$ and $b$ are two Sigmoid function parameters. $\varphi_i(t)$ is the elevation angle between the drone and sensor $i$ at time $t$. Furthermore, the path loss between the drone and sensor $i$ is given by

$$\begin{aligned} h_i(t) = {} & \Pr_{\text{LoS}}(\varphi_i(t))(\eta_{\text{LoS}} - \eta_{\text{NLoS}}) + 20\log(\mathcal{R}\sec\varphi_i(t)) \\ & + 20\log(f_c) + 20\log(4\pi/v_c) + \eta_{\text{NLoS}}, \end{aligned} \tag{3}$$

where $\mathcal{R}$, $f_c$, and $v_c$ are the radius of the radio coverage of the drone, the carrier frequency, and the speed of light, respectively. $\eta_{\text{LoS}}$ and $\eta_{\text{NLoS}}$ stand for the excessive path loss of LoS and non-LoS, respectively. The value of $(\eta_{\text{LoS}}, \eta_{\text{NLoS}})$ pair can be (0.1, 21), (1.0, 20), (1.6, 23), or (2.3, 34), corresponding to suburban, urban, dense urban, or highrise urban scenarios [39].

The complex coefficient of the reciprocal wireless channel between the drone and the ground sensor can be known by channel reciprocity. Given the data rate of the ground sensor $r_i(t)$, the transmit power of the ground sensor, denoted by $P_i(t)$, can be given by

$$P_i(t) \approx \frac{\kappa_2^{-1}\ln\frac{\kappa_1}{\varepsilon}}{\|h_i(t)\|^2}(2^{r_i(t)} - 1), \tag{4}$$

where $\kappa_1$ and $\kappa_2$ are two channel constants [40]. $\varepsilon$ is the required BER between the ground sensors and the drone.

## 4 Formulation Of Absorbing Markov Chain

Let $e_i(t)$, $d_i(t)$, and $h_i(t)$ denote the battery level and the queue length of the ground sensors, and the channel quality, respectively. At each time slot $t$, a ground sensor, e.g., the $i$th sensor, is selected by the drone for data transmission. To estimate the battery level and the queue length of the ground sensors, the TTA value, denoted by $\tau_i$, is recorded and updated at the drone for sensor $i$. $\tau_i$ increases by 1 at time $t$ if sensor $i$ is not selected, and $\tau_i$ returns to 0 when a new packet is collected from $i$.

Due to the limited battery energy of the drone, the maneuver control of the drone and communication schedule can be modeled as an absorbing Markov chain. The network state $\alpha$ is given by

$$\alpha = \langle e_{\text{drone}}(t), e_i(t), d_i(t), h_i(t), \tau_i(t)\rangle \tag{5}$$

where $i \in [1, N]$, and the absorbing states are referred to as network states with $e_{\text{drone}}(t) = e_{\text{drone}}^{min}$.

Let $A$ denote the number of the absorbing states in which $e_{\text{drone}}(t) = e_{\text{drone}}^{min}$. Assume that the number of transitions from state $\alpha$ to the absorbing state is $B$. In other words, the drone can take $B$ actions for the maneuver control and communication scheduling until the drone depletes the propulsion energy. Moreover, the absorbing Markov chain can be characterized by using the following transition matrix $\mathbf{Z}$ in the canonical form:

$$\mathbf{Z} = \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}, \tag{6}$$

where $\mathbf{0}$ is an $A \times B$ zero matrix, $\mathbf{1}$ is an $A \times A$ identity matrix, $\mathbf{X}$ is a $B \times B$ transition probability matrix with the elements of $\{\Pr_i\{\beta|\alpha\}\}$ ($\alpha, \beta = [1, B]$) that specifies the transition probability of $i$ from state $\alpha$ to state $\beta$, and $\mathbf{Y}$ is the absorbing probability matrix containing the probabilities $\{\Pr_i\{\beta'|\alpha\}\}$ of the transition from state $\alpha$ to the absorbing state $\beta'$.

At state $\alpha$, a sensor, i.e., sensor $i$, is selected by the drone and the sensor transits to the next state, i.e., state $\beta$. The transition probability depends on the following possible transitions.

- $(e_i(\alpha), d_i(\alpha), e_{\text{drone}}(\alpha))$ transits to $(e_i(\beta) = e_i(\alpha) + \Delta e_i, d_i(\beta) = d_i(\alpha) + 1, e_{\text{drone}}(\beta) = e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}})$: Herein, $d_i(\beta) = d_i(\alpha) + 1$ indicates a new data packet is buffered; or in other words, the data transmission of sensor $i$ is unsuccessful. The drone's battery at state $\beta$ is $e_{\text{drone}}(\beta) = e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}}$, where $\Delta e_{\text{drone}}$ is the propulsion energy consumption of the drone. Let $\Pr_{\Delta e}$ denote the probability that the ground sensor harvests the energy. If the sensor manages to harvest the energy (i.e., $\Delta e_i > 0$), then

$$\Pr_i\{\beta|\alpha\} = \lambda(1 - (1 - \epsilon)^D)\Pr_{\Delta e}. \tag{7}$$

Otherwise, energy harvesting is unsuccessful, i.e., $1 - \Pr_{\Delta e}$, and $\Delta e_i = 0$, we have

$$\Pr_i\{\beta|\alpha\} = \lambda(1 - (1 - \epsilon)^D)(1 - \Pr_{\Delta e}). \tag{8}$$

Moreover, if the drone does not have sufficient propulsion energy, then state $\alpha$ is the absorbing state, i.e., $e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}} \leq 0$. The transition probability is $\Pr_i\{\beta|\alpha\} = 0$.

- $(e_i(\alpha), d_i(\alpha), e_{\text{drone}}(\alpha))$ transits to $(e_i(\beta) = e_i(\alpha) + \Delta e_i, d_i(\beta) = d_i(\alpha) - 1, e_{\text{drone}}(\beta) = e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}})$: Here, $d_i(\beta) = d_i(\alpha) - 1$ indicates that the buffer of the selected sensor $i$ decreases by 1; or in other words, the data transmission is successful. If the battery of the drone is non-empty at state $\beta$, i.e., $e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}} > 0$ and the energy $\Delta e_i (> 0)$ is harvested by sensor $i$, the transition probability is given by

$$\Pr_i\{\beta|\alpha\} = (1 - \lambda)(1 - \epsilon)^D \Pr_{\Delta e}, \tag{9}$$

where $1 - \lambda$ indicates that there is no new packet arrival at state $\alpha$. If the energy harvesting of sensor $i$ is unsuccessful, i.e., $\Delta e_i = 0$, then

$$\Pr_i\{\beta|\alpha\} = (1 - \lambda)(1 - \epsilon)^D (1 - \Pr_{\Delta e}). \tag{10}$$

In addition, $\Pr_i\{\beta|\alpha\} = 0$ if state $\alpha$ is the absorbing state, i.e., $e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}} \leq 0$.

- $(e_i(\alpha), d_i(\alpha), e_{\text{drone}}(\alpha))$ transits to $(e_i(\beta) = e_i(\alpha) + \Delta e_i, d_i(\beta) = d_i(\alpha), e_{\text{drone}}(\beta) = e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}})$: Here, $d_i(\beta) = d_i(\alpha)$ indicates that the buffer of the selected ground sensor remains unchanged, due to either a successful transmission with a new packet arrival (which gives $\lambda(1 - \epsilon)^D$), or a failed transmission with no new packet arrival (which gives $(1 - \lambda)(1 - (1 - \epsilon)^D)$). If the harvested energy $\Delta e_i > 0$ and $e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}} > 0$,

$$\Pr_i\{\beta|\alpha\} = [(1 - \lambda)(1 - (1 - \epsilon)^D) + \lambda(1 - \epsilon)^D]\Pr_{\Delta e}. \tag{11}$$

If $\Delta e_i = 0$, then

$$\Pr_i\{\beta|\alpha\} = [(1 - \lambda)(1 - (1 - \epsilon)^D) + \lambda(1 - \epsilon)^D](1 - \Pr_{\Delta e}). \tag{12}$$

Otherwise, state $\alpha$ is the absorbing state and $\Pr_i\{\beta|\alpha\} = 0$.

For the unselected ground sensors $j \in [1, N]$ and $j \neq i$, at the next state, their buffers can either remain unchanged ($d_j(\beta) = d_j(\alpha)$) or increase by 1 due to a new packet arrival ($d_j(\beta) = d_j(\alpha) + 1$). Consequently, we have the state transition probability, as follows.

- $(e_j(\alpha), d_j(\alpha), e_{\text{drone}}(\alpha))$ transits to $(e_j(\beta) = e_j(\alpha) + \Delta e_j, d_j(\beta) = d_j(\alpha), e_{\text{drone}}(\beta) = e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}})$: If $\Delta e_j$ is harvested and $e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}} > 0$, then

$$\Pr_j\{\beta|\alpha\} = (1 - \lambda)\Pr_{\Delta e}. \tag{13}$$

If the energy is not harvested, i.e., $(1 - \Pr_{\Delta e})$, then

$$\Pr_j\{\beta|\alpha\} = (1 - \lambda)(1 - \Pr_{\Delta e}). \tag{14}$$

Otherwise, state $\alpha$ is the absorbing state and $\Pr_j\{\beta|\alpha\} = 0$.

- $(e_j(\alpha), d_j(\alpha), e_{\text{drone}}(\alpha))$ transits to $(e_j(\beta) = e_j(\alpha) + \Delta e_j, d_j(\beta) = d_j(\alpha) + 1, e_{\text{drone}}(\beta) = e_{\text{drone}}(\alpha) - \Delta e_{\text{drone}})$: In this case, a new packet is buffered with probability $\lambda$, thus

$$\Pr_j\{\beta|\alpha\} = \begin{cases} \lambda \Pr_{\Delta e}, & \text{if } \Delta e_j > 0; \\ \lambda(1 - \Pr_{\Delta e}), & \text{otherwise.} \end{cases} \tag{15}$$

At the absorbing state $\alpha$, $\Pr_j\{\beta|\alpha\} = 0$.

The optimal policy in the absorbing Markov chain can be obtained by classical approaches, e.g., value iteration or policy iteration. The value iteration method repeatedly updates the estimate of the optimal action-value function until the Bellman optimality equation converges. The policy iteration

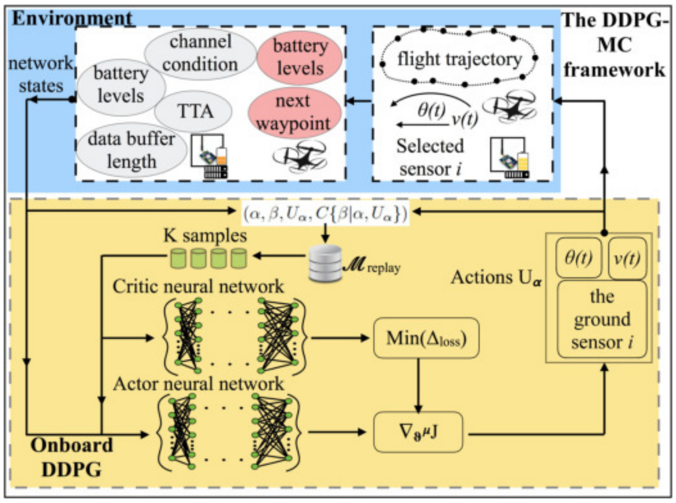


Fig. 3. An illustration of the DDPG-MC architecture, where deep reinforcement learning with experience relay is carried out at the drone to optimize its actions.

method evaluates the optimized policy at each of iterations, which is a protracted iterative computation involving multiple sweeps through the state set. However, both the value iteration and policy iteration methods require the transition probabilities of all states to be known at the drone in prior. In contrast, this paper is interested in a practical scenario where the drone has no a-priori knowledge on $\Pr_i\{\beta|\alpha\}$. Reinforcement learning can solve Markov decision processes in the absence of the knowledge of the state transition probabilities, i.e., $\Pr_i\{\beta|\alpha\}$. One of the popular reinforcement learning techniques is Q-learning, where an agent interacts with the environment to minimize the long-term cost. Q-learning typically supports discrete state and action spaces, and therefore is not suitable for the continuous state and action spaces in the drone maneuver problem considered here. Even after being discretized, the state and action spaces in the drone-assisted sensor network are typically large. Q-learning would suffer from the well-known curse-of-dimensionality [41], and therefore is not adequate to solve the online maneuver control and communication schedule.

## 5 DDPG For Maneuver Control and Communication Scheduling

In this section, we propose to use DDPG-MC to solve the Markov decision process with the large and continuous state and action spaces. As an effective deep reinforcement learning technique, DDPG-MC can optimize the continuous maneuver control and communication schedule in the absence of the a-priori knowledge of the state transition probabilities, i.e., $\Pr_i\{\beta|\alpha\}$, and minimize the long-term accumulated costs of the system (i.e., the packet loss of all ground sensors).

### 5.1 DDPG-MC Framework

DDPG is a learning approach that concurrently learns an action-value function and a policy. DDPG utilizes an Actor-Critic architecture to combine the value iteration and the policy iteration to implement the proposition of the continuous state space and the continuous action space by using deep reinforcement learning. This is different from deep Q-networks which focus on a discrete action space. Moreover, DDPG can enlarge the state space of the absorbing Markov chain compared with reinforcement learning which suffers from the well-known curse of dimensionality [42]. Therefore, in this paper, the joint optimization of the online continuous maneuver control and communication schedule is developed based on DDPG.

As a form of stochastic policy gradient, deterministic policy gradients enable a deterministic mapping from the network state to the optimal actions of the drone in the absorbing Markov chain. The structure of the proposed DDPG-MC framework is depicted in Fig. 3, where DDPG is trained onboard at the drone for the maneuver control and the ground sensor selection. The actions of the drone in DDPG-MC define

$$U_\alpha = (\theta(\alpha), v(\alpha), \{i_\alpha \in [1, N]\}), \tag{16}$$

where $U_\alpha \in \mathcal{A}$, and $\mathcal{A}$ contains all the actions that the drone can carry out for optimization of the maneuver control and communication schedule.

In Fig. 3, the network states, including $e_i(t)$ and $d_i(t)$ ($i \in [1, N]$) from the ground sensors, and $e_{\text{drone}}(t)$, $(x(t), y(t), z)$, $\tau_i$ and $h_i(t)$ from the drone, are observed in the environment for training DDPG-MC. $C\{\beta|\alpha, U_\alpha\}$ is the network cost when action $U_\alpha$ is taken and the system transits from state $\alpha$ to state $\beta$. $C\{\beta|\alpha, U_\alpha\}$ is measured by the packet loss of the system. In other words, $C\{\beta|\alpha, U_\alpha\}$ counts the number of packets dropped or lost during the state transition. The packet loss can be caused by both buffer overflows and channel fading during the state transition. Moreover, the experience tuple $(\alpha, \beta, U_\alpha, C\{\beta|\alpha, U_\alpha\})$ is stored in the replay memory $\mathcal{M}_{\text{replay}}$ of the drone at each training step. $K$ samples (or minibatches) of the experience in $\mathcal{M}_{\text{replay}}$ are used along with the input states from the environment to train the DDPG-MC onboard.

DDPG-MC is built based on the actor-critic neural network structure [43]. Due to the continuity of the maneuver control of the drone, the action-value function $Q\{\alpha, U_\alpha\}$ is presumed to be differentiable with respect to the action argument. This allows us to set up a gradient-based learning rule for the maneuver control and communication scheduling policy $\mu(\alpha)$. Instead of exhaustively evaluating the entire action space to minimize $Q\{\alpha, U_\alpha\}$, DDPG-MC approximates the optimal actions of the maneuver control and communication schedule with $Q\{\alpha, \mu(\alpha)\}$.

The actor neural network in DDPG-MC generates the actions of setting $\theta(t)$ and $v(t)$, and selects the ground sensor $\{i_t \in [1, N]\}$. The critic neural network approximates the optimal action-value function $Q\{\alpha, U_\alpha\}$ that calculates the expected accumulated network cost, i.e., the overall data loss, after observing the state $\alpha$ and taking the action $U_\alpha$. Let $\mu\{\alpha|\vartheta^\mu\}$ and $\mu'\{\alpha|\vartheta^{\mu'}\}$ denote the actor's policy of maneuver control and sensor selection, and the target actor function, respectively. $\vartheta^\mu$ and $\vartheta^{\mu'}$ are the two weights for policy update.

As shown in Fig. 3, the critic network learns the optimal $Q\{\alpha, U_\alpha\}$ using the Bellman equation to minimize the approximation loss $\Delta_{\text{loss}}$ that defines

$$\Delta_{\text{loss}} = \frac{1}{K}\sum_k (C\{\beta|\alpha, U_\alpha\}_k + \delta Q'\{\alpha_{k+1}, \mu'\{\alpha_{k+1}|\vartheta^{\mu'}\}|\vartheta^{Q'}\} - Q\{\alpha_k, U_{\alpha_k}|\vartheta^Q\})^2, \tag{17}$$

where $\delta$ is the discount factor, and $Q\{\alpha_k, U_{\alpha_k}|\vartheta^Q\}$ is parameterized by the weight $\vartheta^Q$ in the critic network. $Q'\{\cdot\}$ is the target action-value function in the critic network.

The objective of DDPG-MC is to minimize the expected packet loss of the ground sensors, i.e., $\mathbb{E}[Q\{\alpha, U_\alpha\}]$. The actor network in DDPG-MC is updated by applying the chain rule on the expected packet loss from the initial distribution $J$ with respect to the actor weights $\vartheta^\mu$. The gradient of the DDPG-MC policy is given by

$$\nabla_{\vartheta^\mu} J \approx \mathbb{E}_{\alpha_t}[\nabla_{\vartheta^\mu} Q\{\alpha, U_\alpha|\vartheta^Q\}|_{\alpha=\alpha_t, U_\alpha=\mu(\alpha_t|\vartheta^\mu)}]. \tag{18}$$

Furthermore, the policy in DDPG-MC is also trained with $K$ minibatches of experience in $\mathcal{M}_{\text{replay}}$, as depicted in Fig. 3. Hence, $\nabla_{\vartheta^\mu} J$ can be calculated by the mean of the sum of gradients from the experience replay, which is

$$\nabla_{\vartheta^\mu} J \approx \frac{1}{K}\sum_k \nabla_{U_\alpha} Q\{\alpha, U_\alpha|\vartheta^Q\}|_{\alpha=\alpha_k, U_\alpha=\mu(\alpha_k)} \times \nabla_{\vartheta^\mu}\mu\{\alpha|\vartheta^\mu\}|_{\alpha_k}. \tag{19}$$

According to the DDPG-MC architecture in Fig. 3, Algorithm 1 is formulated to demonstrate the DDPG-MC implementation with deep reinforcement learning. Given a total of $M$ episodes and a training time of $t_{\text{learning}}$, action $U_\alpha$ is carried out by the drone at every time step with a random process $\zeta_t$ for action exploration, as given by

$$U_\alpha = \mu\{\alpha|\vartheta^\mu\}_t + \zeta_t, \tag{20}$$

The experience of maneuver control and sensor selection, i.e., $(\alpha, \beta, U_\alpha, C\{\beta|\alpha, U_\alpha\})$, is stored in $\mathcal{M}_{\text{replay}}$, and $K$ samples are used to minimize $\Delta_{\text{loss}}$. Moreover, the actor policy is updated at the drone with the sampled policy gradients according to (19). With the optimized actor policy, the two target neural networks can be updated onboard at the drone, where

$$\begin{cases} \vartheta^{Q'} & \leftarrow \epsilon\vartheta^Q + (1-\epsilon)\vartheta^{Q'} \\ \vartheta^{\mu'} & \leftarrow \epsilon\vartheta^\mu + (1-\epsilon)\vartheta^{\mu'} \end{cases}. \tag{21}$$

The drone can only observe the network state of itself and the selected ground sensor at any moment, including the sensor's battery level, queue length, channel quality, and TTA. Suppose that sensor $i$ is selected at time $t$. The observed network state is $\alpha_i = \langle e_{\text{drone}}(t), e_i(t), d_i(t), h_i(t), \tau_i(t)\rangle$. The drone can evaluate the packet loss pertaining to the selection, based on this observation and the records of the rest of the ground nodes in the experience replay memory. In the experience replay memory, each record is associated with a timestamp, i.e., TTA, indicating how many slots have elapsed since the latest observation of a node. By replaying the memory of the unselected sensors based on their TTAs, the drone can approximate the network state (in addition to the observation of the selected sensors), evaluate the packet loss, and produce a piece of training experience. As part of the network state, the TTA can have a strong impact on the actions of the drone. In particular, a ground sensor with a large TTA value potentially has a long data queue and is likely to suffer from a buffer overflow. Moreover, with the increasing TTA of a sensor, the experience replay can become less accurate at the drone. To this end, the proposed approach is effective, as reduces the TTAs of the sensors and improves the learning accuracy.

The observation and evaluation are also used to update the experience replay memory of the drone. Specifically, the training experience of selecting the particular sensor, including the packet loss and the timestamps of all the rest of the sensors, is associated with the TTA of the sensor and added to the experience replay memory. By carrying out the experience replay, DDPG-MC can learn online the underlying patterns of the data and energy arrivals, and the channel dynamics of the ground sensors.

Some sensors may periodically generate packets. The periodic packet arrivals in the data queue are predictable, while the battery energy levels and channel conditions still experience time-varying randomness, depending on the environments. The proposed DDPG-MC can optimize the actions of the drone by learning the dynamics of these elements.

**Algorithm 1.** DDPG-MC Framework

```
1: 1. Initialize:
2:   $\alpha, \beta \in \mathcal{S}$, $U_\alpha \in \mathcal{A}$, learning time $\rightarrow t_{\text{learning}}$, and experience replay capacity $\rightarrow \mathcal{M}_{\text{replay}}$.
3:   The critic network $Q\{\alpha, U_\alpha|\vartheta^Q\}$ and the actor network $\mu\{\alpha|\vartheta^\mu\}$ are randomly initialized, where the two weights are $\vartheta^Q$ and $\vartheta^\mu$.
4:   Initializing target networks $Q'$ and $\mu'$ with the weights $\vartheta^{Q'} \leftarrow \vartheta^Q$ and $\vartheta^{\mu'} \leftarrow \vartheta^\mu$.
5: 2. Learning:
6: for episode $1 \rightarrow M$ do
7:   The drone observes network state $\alpha$. Random process for exploration $\rightarrow \zeta_t$.
8:   while $t \leq t_{\text{learning}}$ do
9:     Action $U_\alpha$ is carried out by the drone, where $U_\alpha = \mu\{\alpha|\vartheta^\mu\}_t + \zeta_t$, which sets $\theta(\alpha)$ and $v(\alpha)$ of the drone, and selects a sensor for data collection.
10:    The drone calculates $C\{\beta|\alpha, U_\alpha\}$, and obtains a new state observation $\beta$.
11:    Onboard at the drone: $(\alpha, \beta, U_\alpha, C\{\beta|\alpha, U_\alpha\}) \rightarrow \mathcal{M}_{\text{replay}}$.
12:    The drone randomly takes a minibatch of $K$ samples from the onboard memory $\mathcal{M}_{\text{replay}}$.
13:    For each sample $k$, $y_k = C\{\beta|\alpha, U_\alpha\}_k + \delta Q'\{\alpha_{k+1}, \mu'\{\alpha_{k+1}|\vartheta^{\mu'}\}|\vartheta^{Q'}\}$.
14:    Minimizing the loss function onboard at the drone, where $\Delta_{\text{loss}} \leftarrow \frac{1}{K}\sum_k (y_k - Q\{\alpha_k, U_{\alpha_k}|\vartheta^Q\})^2$.
15:    According to (19), the drone updates the actor policy with the sampled policy gradients.
16:    With the optimized actor policy, $\vartheta^{Q'} \leftarrow \epsilon\vartheta^Q + (1-\epsilon)\vartheta^{Q'}$ and $\vartheta^{\mu'} \leftarrow \epsilon\vartheta^\mu + (1-\epsilon)\vartheta^{\mu'}$.
17:  end while
18: end for
```

## 5.2 Complexity Analysis

To minimize $\Delta_{\text{loss}}$, the complexity of the proposed DDPG-MC lies in updating the actor policy with $\nabla_{\vartheta^\mu} J$ and conducting the experience replay for training the four neural networks. Moreover, the drone observes a network state from the environment at each training episode. This leads to $O(\mathcal{S})$, where $\mathcal{S}$ is the total number of network states. Consider $W$ and $G$ fully connected layers in the actor and critic networks, respectively. The computations of activation layers in DDPG-MC lead to the complexity of $O(\sum_{w=0}^{W-1} n_w^{tensor} n_{w+1}^{tensor} + \sum_{g=0}^{G-1} n_g^{tensor} n_{g+1}^{tensor})$, where $n_w^{tensor}$ is the number of tensors at the $w$th layer in the actor network, and $n_g^{tensor}$ is that in the critic network. Therefore, the overall time complexity of DDPG-MC is $O(\mathcal{S})+ O(\sum_{w=0}^{W-1} n_w^{tensor} n_{w+1}^{tensor} + \sum_{g=0}^{G-1} n_g^{tensor} n_{g+1}^{tensor})$.

## 6 Performance Evaluation

In this section, we first demonstrate the implementation of the proposed DDPG-MC framework on Google TensorFlow (the symbolic math library for numerical computation) [35]. Numerical results are presented to evaluate the packet loss rate against the maneuver control of the drone, the number of ground sensors, the data buffer size and the data arrivals of the ground sensor.

### 6.1 Implementation of DDPG-MC on TensorFlow

DDPG-MC is implemented in Python 3.5 on TensorFlow. A desktop with 4-core Intel i7-6700K 4 GHz CPUs and 16 G memory based on 64-bit Ubuntu 16.04 is used for the TensorFlow setup. DDPG-MC is trained for 300 episodes, where $M$ = 300, while $t_{\text{learning}}$ = 200 epochs. The onboard memory $\mathcal{M}_{\text{replay}}$ keeps 10,000 training records, while each training episode can use the mini-bacth of 100 samples.

The area of interest is set to be a square area with a size of 1,000 m $\times$ 1,000 m. $N$ ground sensors are distributed in the region, where $N$ is from 100 to 600. The data packets are generated at each sensor according to the packet arrival probability $\lambda$ = 0.5. The maximum transmit power is 100 milliwatts. The battery energy of the ground sensor has 800 Joules, while the battery capacity of the drone has $2.5 \times 10^5$ Joules. The drone has the highest patrol velocity $V$ = 15 m/s. In addition, we assume that the BER needs to be no greater than 0.05 percent, i.e., $\varepsilon \leq$ 0.05 percent, to achieve correct detection and decoding at the drone. Thus, the required transmit power of the ground sensor can be given in (4).

### 6.2 Performance of DDPG-MC

For performance comparison, DDPG-MC is compared with three other onboard online trajectory planning and communication scheduling policies as

- Sequential visiting and Random Scheduling policy (SeqRS). The area of interest is evenly divided into 25 subareas, where each subarea contains one waypoint. The drone sequentially visits all the 25 waypoints, while the drone randomly selects one ground sensor to collect data at each time slot. The maneuver control and communication schedule of SeqRS are independent of the battery and buffer length of the ground sensor, or channel variation.
- Sequential visiting and Channel-Aware scheduling policy (SeqCA). The drone sequentially visits the 25 predetermined waypoints, where a-prior knowledge on the channels in the target field is assumed to be known to the drone. At each time slot, the ground sensor with the highest SNR is given the highest priority to transmit data.
- Deep Q-Networks based transmission scheduling policy (DQN) [44]. Given the predetermined trajectory of the drone, DQN is trained to schedule the data transmission of the ground sensors by learning the change of their battery levels, buffer lengths, and channels.

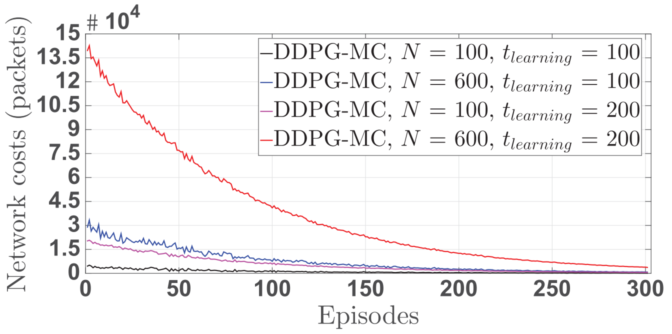


Fig. 4. Network cost, i.e., packet loss, in terms of the episodes of DDPG-MC.

#### 6.2.1 Network Cost of DDPG-MC

The total number of episodes, i.e., learning iterations, is set to 300, each of which contains a series of consecutive training epochs. The ground sensors are uniformly distributed in the target area. Fig. 4 shows the network cost, i.e., packet loss, at each training episode of the proposed DDPG-MC, given $N$ = 100 or 600, and $t_{\text{learning}}$ = 100 or 200, respectively. Generally, DDPG-MC has a high network cost at the first 10 episodes of the training process. With an increasing number of episodes, the network cost drops significantly until it reaches a relatively stable value. It confirms the fact that DDPG can converge after a number of episodes when the actor and the critic neural networks are sufficiently trained. Particularly, the network cost of DDPG-MC with $t_{\text{learning}}$ = 100 is about 2968 packets lower than the one with $t_{\text{learning}}$ = 200, when $N$ = 600. The reason is that more data packets are generated at the ground sensors in an extended $t_{\text{learning}}$, which leads to more overflowed buffers.

#### 6.2.2 Maneuver Control

Figs. 5a, 5b, and 5c study the trajectories of the drone with regard to three deployments of the ground sensors, i.e., uniform distribution, normal distribution, and ring-shaped distribution, where $N$ = 100. Moreover, Fig. 5d presents the network cost of DDPG-MC according to the above three deployments. As observed, the maneuver control of the drone is persistently adapted by DDPG-MC given different deployments of the sensors. This is because DDPG-MC optimizes $\theta(\alpha)$ and $v(\alpha)$ in the continuous action space while learning the network state dynamics, to determine the optimal trajectory as well as the ground sensor for minimizing the network cost.

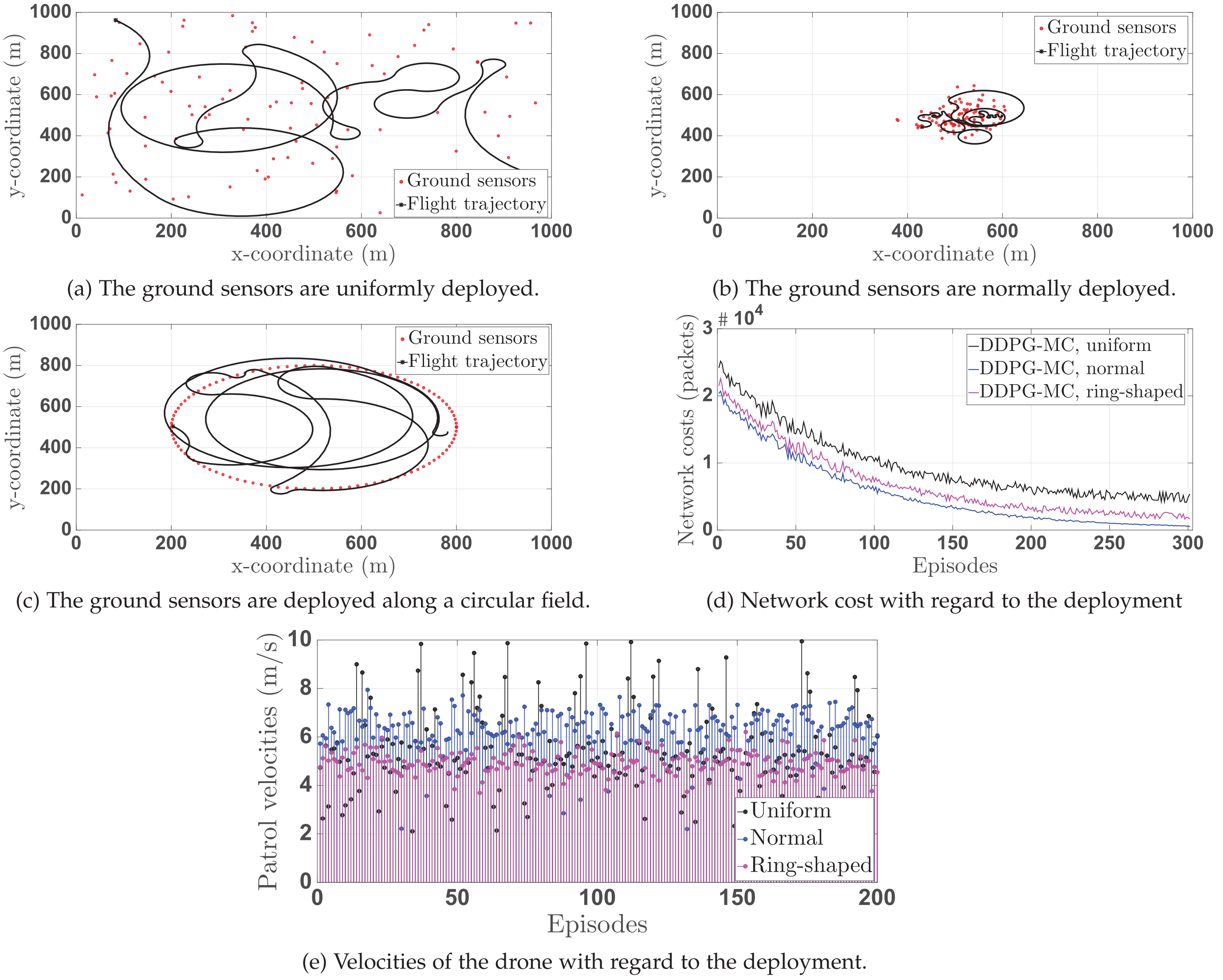


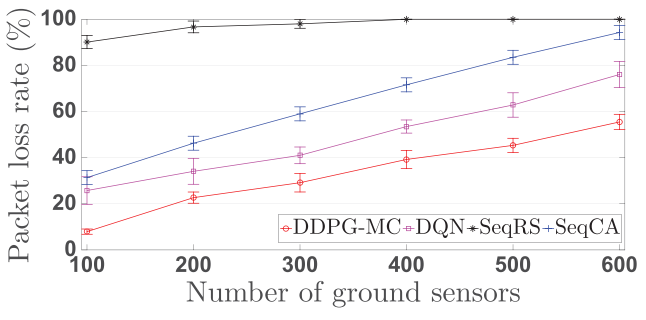
Fig. 5. The trajectories and the velocities of the drone, and the network cost of DDPG-MC with regard to three deployments of the ground sensors.

Furthermore, we also observe in Fig. 5d that the network cost of DDPG-MC with the uniform deployment of the sensors is slightly higher than the ones with the ring-shaped and the normal deployment. This is reasonable because the ground sensors within the radio coverage of the drone can be selected for the data collection, while the others may experience buffer overflows. Therefore, a dense deployment of the ground sensors is likely to reduce the packet loss stemming from overflowing buffers.

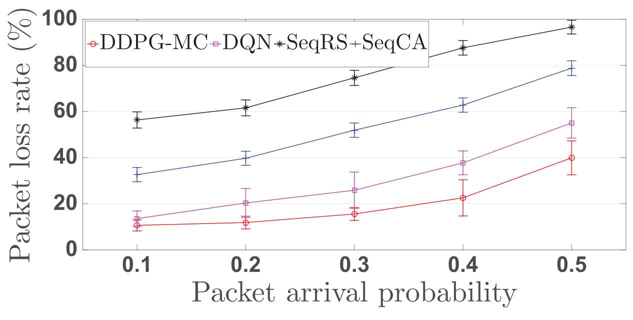
Fig. 5e shows that the velocity of the drone is dynamically adjusted by DDPG-MC according to the sensor deployment. The velocity has the largest fluctuation, ranging between 2 m/s and 10 m/s, when the ground sensors are uniformly deployed. The velocity fluctuates between 3.5 m/s and 6 m/s given the ring-shaped deployment of the sensors. In addition, the velocity in the normal deployment is between 4 m/s and 8 m/s, which is smaller than the one in the uniform deployment, while the value is generally higher than the one in the ring-shaped deployment. Fig. 5e indicates that the regular shape of the sensor deployment leads to the stable velocity control carried out by DDPG-MC, while the sparse deployment can fluctuate the velocity in a wide range.

### 6.2.3 Packet Loss Rate

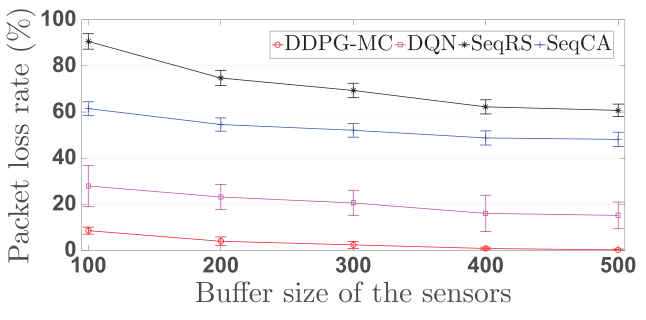
In this case, the deployment of the ground sensors follows the uniform distribution. Fig. 6 presents the packet loss rate of DQN, SeqRS, SeqCA, and the proposed DDPG-MC with regards to the number of ground sensors, the buffer sizes, the packet arrival probabilities, and the altitudes of the drone. The altitude of the drone is maintained at 100 meters during the flight, unless otherwise specified. In Fig. 6a, DDPG-MC achieves the smallest packet loss rate. When $N$ = 100, the packet loss rate of DDPG-MC is smaller than SeqRS and SeqCA by 82.2 percent and 23.5 percent, respectively. The performance gains keep growing with $N$. The reason is that DDPG-MC learns the ground sensors' buffer lengths, battery levels, and channel states, so that the maneuver control and the node selection can minimize the data packet loss of the entire network. Furthermore, DDPG-MC generally achieves 18.6 percent lower packet loss rate than DQN.



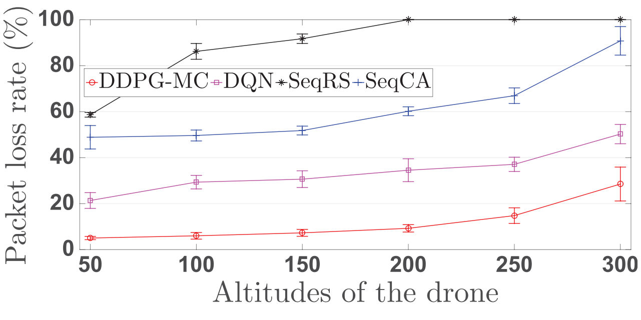
(a) $N$ increases from 100 to 600.



(b) $\lambda$ increases from 0.1 to 0.5.



(c) $D$ increases 100 from 500.



(d) $z$ grows from 50 to 300 meters.

Fig. 6. Packet loss rate with regards to the number of ground sensors, the packet arrival probabilities, the buffer sizes, and the altitudes of the drone. Each error bar presents the standard deviation over ten experiments.

This is because the action space in DQN is discrete, where the drone adapts the heading and the velocity intermittently. As a result, some maneuver control and communication scheduling policies that can achieve a smaller network cost are not explored by DQN. In contrast, $\theta(\alpha)$ and $v(\alpha)$ in DDPG-MC are optimized in the continuous action space, which adjusts the trajectory in real time while scheduling more potential ground sensors for minimizing the packet loss rate.

Fig. 6b shows that the packet loss rate of DQN, SeqCA, and SeqRS grows to 55.3 percent, 79.6 percent, and 96.3 percent, respectively, when the packet arrival rate $\lambda$ increases from 0.1 to 0.5. On the contrary, the packet loss rate of DDPG-MC increases to 40.1 percent which is lower than the other three benchmarks. The reason is that DDPG-MC optimizes the future maneuver control and communication schedules at every location of the drone by taking advantage of the learning experience in the replay memory, which controls $\theta(\alpha)$ and $v(\alpha)$ adapting to the data traffic. It can also be observed from Fig. 6b that the performance gap decreases with $\lambda$. This is reasonable because a larger $\lambda$ leads to more buffer overflows at the ground sensors, while one ground sensor can be selected by the drone for the data transmission.

Fig. 6c depicts the packet loss rate when the buffer size of the ground sensor, $D$, is extended from 100 to 500. In general, the packet loss rate drops with an increased $D$. Particularly, DDPG-MC outperforms DQN, SeqCA, and SeqRS on the packet loss rate by 15.2 percent, 47.6 percent, and 60.3 percent, respectively, when $D = 500$. Although DQN can also learn the actions of the drone based on the experience replay, DDPG-MC trains the actions in the continuous action space that is much larger than the one with DQN. Therefore, DDPG-MC obtains the actions that can further minimize the network cost than the DQN policy.

As shown in Fig. 6d, the packet loss rates of DDPG-MC, DQN, SeqCA, and SeqRS generally grow when the altitude of the drone increases from 50 meters to 300 meters, due to the increasing large-scale fading. DDPG-MC achieves the lowest packet loss rate when the drone is under the different altitudes. In addition, when the altitude increases from 50 m to 300 m, the packet loss rate of DDPG-MC increases by 20 percent, which is much lower than the 30 percent of DQN, 41 percent of SeqCA, or 40 percent of SeqRS.

### 6.2.4 *Goodput of the Ground Sensors*

In this case, we study the goodput of the ground sensors, which is illustrated by the number of generated (stored in the data queue) and transmitted packets. The deployment of the ground sensors follows a uniform distribution with the average density of 0.5 per square meter. Given $N = 100$, it can be observed in Fig. 7 that all the ground sensors are scheduled to transmit the data to the drone. This

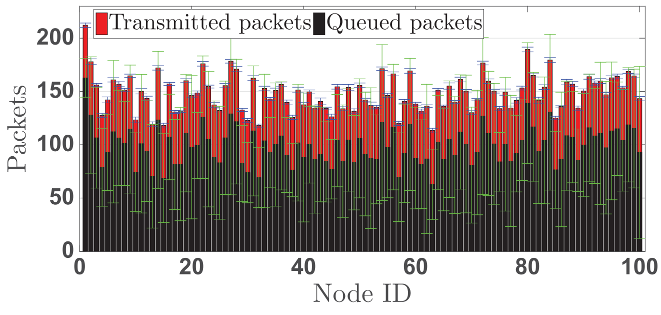


Fig. 7. The goodput of the ground sensors given $N = 100$, where the standard deviation is calculated based on 10 experiments.

TABLE 2
Jain's Fairness Index With Regard to the Three Deployments of the Ground Sensors

| | $\mathcal{J}$ |
|---|---|
| **Uniform** | 0.98 |
| **Normal** | 0.99 |
| **Ring-shaped** | 0.94 |

is benefited from the joint optimization of maneuver control and node selection in DDPG-MC. Interestingly, we also see that most of the ground sensors have similar number of transmitted packets given a specific data buffer size. This is because the TTA value $\tau_i$ recorded at the drone increases by 1 if sensor $i$ is not selected to transmit data. Sensor $i$ can have a large number of buffer overflows with the growth of the TTA value. Therefore, DDPG-MC adapts the drone maneuver to collect data of sensor $i$ to minimize the packet loss.

We also adopt Jain's index, denoted as $\mathcal{J}$, as the fairness measurement of the transmitted packets in the following:

$$\mathcal{J} = \frac{\left(\sum_{i=1}^{N} d_i^{tx}\right)^2}{N\sum_{i=1}^{N}(d_i^{tx})^2}, \tag{22}$$

where $d_i^{tx}$ is the number of transmitted packets of sensor $i$. Table 2 shows the Jain's fairness index under the three deployment schemes of the ground sensors. The indexes achieved by DDPG-MC are over 0.93 under all the three deployments. This is because DDPG-MC can adjust the drone maneuver to visit and schedule the ground sensors, to minimize the packet loss of the sensors. The index is 0.94 under the ring-shaped deployment, lower than 0.98 under the uniform deployment or 0.99 under the normal deployment. This is due to the fact that the drone tries to align its trajectory with the circular deployment of the sensors, as shown in Fig. 5c. The trajectory can be too long to have all sensors visited. The delay can be too high between two visits to every sensor. If the delay is longer than the average interval between packet arrivals at a sensor, the buffers of the sensors would overflow. In this case, the drone has to bypass some sensors, take a shortcut, and reduce the number of sensors undergoing buffer overflows, costing the fairness of the bypassed sensors.

### 6.2.5 Runtime Measurements

Fig. 8 shows the runtime measurements of DDPG-MC, where $N$ and $t_{\text{learning}}$ are set to 100 or 600, and 100 or 200. The runtime of DDPG-MC with $N$ = 100 and $t_{\text{learning}}$ = 100 is around 0.52 ms, and it increases to 1.12 ms when $t_{\text{learning}}$ grows to 200.

This is because the increased $t_{\text{learning}}$ leads to more training iterations in DDPG-MC, which consumes extra time on updating the onboard neural networks. Moreover, the runtime increases from 1.12 ms to 3.49 ms when $N$ = 600. This is because the increased $N$ enlarges the state space. Nevertheless, the relative increase in the runtime is much slower than that in the number of ground sensors. This is because the action space does not grow dramatically with

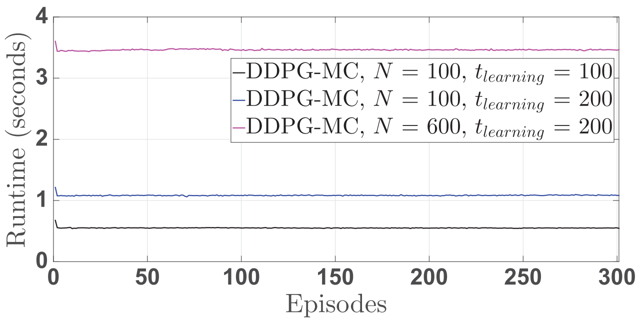


Fig. 8. Runtime measurements of DDPG-MC with regards to the number of ground sensors and the learning iterations.

$N$. In particular, given a network state, the actions that the drone can take are limited by its current position and the number of sensors within the radio coverage.

## 7 Conclusions and Discussions

This paper investigates the joint maneuver control of the drone and communication schedule. The drone-assisted data collection is formulated as an absorbing Markov chain to minimize the data lost due to buffer overflows at the ground sensors and fading airborne channels. Given the continuous action space of the maneuver control, onboard DDPG-MC is proposed to optimally determine the instantaneous headings and patrol velocities as well as the selection of the ground sensor for the data collection. The proposed DDPG-MC utilizes the experience replay to train the policy gradients for minimizing the approximation loss between the actor-critic neural networks and the target neural networks. DDPG-MC is implemented on Google TensorFlow. Numerical results demonstrate that DDPG-MC dynamically adapts the maneuver control for minimizing the packet loss under diverse deployments of the ground sensors. Moreover, DDPG-MC significantly reduces the packet loss rate with regards to different number of ground sensors, buffer sizes, and packet arrival probabilities, compared to the state-of-the-art strategies.

The proposed DDPG-MC scheme is elemental to drone-assisted sensor networks, and can be potentially extended in the scenarios where multiple drones are employed to collect the data of ground sensors. The multiple drones individually or collaboratively make their decisions of maneuver control and communication schedule based on their observed network states. DDPG-MC can be conducted at each of the drones to train its action, according to its observed network states which implicitly reflect the actions of the rest of the drones.

## Acknowledgments

This work was supported in part by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UIDB/04234/2020), also by the Operational Competitiveness Programme and Internationalization (COMPETE 2020) under the PT2020 Partnership Agreement, through the European Regional Development Fund (ERDF), and by national funds through the FCT, within project(s) POCI-01-0145-FEDER-029074 (ARNET).

## REFERENCES

[1] Z. Li, Y. Jiang, Y. Gao, L. Sang, and D. Yang, "On buffer-constrained throughput of a wireless-powered communication system," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 2, pp. 283–297, Feb. 2019.

[2] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 3, pp. 443–461, Third Quarter 2011.

[3] D. S. Ghataoura, J. E. Mitchell, and G. E. Matich, "Networking and application interface technology for wireless sensor network surveillance and monitoring," *IEEE Commun. Magazine*, vol. 49, no. 10, pp. 90–97, Oct. 2011.

[4] H. Wang, H. Zhao, J. Zhang, D. Ma, J. Li, and J. Wei, "Survey on unmanned aerial vehicle networks: A cyber physical system perspective," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1027–1070, Second Quarter 2019.

[5] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.

[6] A. Fotouhi *et al.*, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3417–3442, Fourth Quarter 2019.

[7] S. Corp., "Softbank corp. develops aircraft that delivers telecommunications connectivity from the stratosphere," 2019. [Online]. Available: https://www.softbank.jp/en/corp/news/press/sbkk/2019/20190425_02/

[8] A. Garcia, "Optus and ericsson complete australia's first 5G drone flight," 2019. [Online]. Available: https://www.optus.com.au/about/media-centre/media-releases/2019/11/optus-and-ericsson-complete-australias-first-5g-drone-flight

[9] C. Ashraf, "How verizon 5G ultra wideband is lifting drone technology to the next level," 2019. [Online]. Available: https://www.verizon.com/about/our-company/5g/how-verizon-5g-ultra-wideband-lifting-drone-technology-next-level

[10] J. Meredith, "3GPP: Study on enhanced support for aerial vehicles," 2018. [Online]. Available: http://www.3gpp.org/dynareport/36777.htm

[11] K. Lee, J.-R. Lee, and H.-H. Choi, "Learning-based joint optimization of transmit power and harvesting time in wireless-powered networks with co-channel interference," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3500–3504, Mar. 2020.

[12] A. Jushi, A. Pegatoquet, and T. N. Le, "Wind energy harvesting for autonomous wireless sensor networks," in *Proc. Euromicro Conf. Digit. Syst. Des.*, 2016, pp. 301–308.

[13] K. Li, W. Ni, L. Duan, M. Abolhasan, and J. Niu, "SWPT: A joint-scheduling model for wireless powered sensor networks," in *Proc. IEEE Global Commun. Conf.*, 2017, pp. 1–6.

[14] K. Li *et al.*, "Fair scheduling for data collection in mobile sensor networks with energy harvesting," *IEEE Trans. Mobile Comput.*, vol. 18, no. 6, pp. 1274–1287, Jun. 2019.

[15] F. Wang, S. Wu, K. Wang, and X. Hu, "Energy-efficient clustering using correlation and random update based on data change rate for wireless sensor networks," *IEEE Sensors J.*, vol. 16, no. 13, pp. 5471–5480, Jul. 2016.

[16] A.-J. Garcia-Sanchez *et al.*, "Wireless sensor network deployment for monitoring wildlife passages," *Sensors*, vol. 10, no. 8, pp. 7236–7262, 2010.

[17] J. Romeo *et al.*, "Camera sensor arrangement for crop/weed detection accuracy in agronomic images," *Sensors*, vol. 13, no. 4, pp. 4348–4366, 2013.

[18] I. Potamitis and I. Rigakis, "Novel noise-robust optoacoustic sensors to identify insects through wingbeats," *IEEE Sensors J.*, vol. 15, no. 8, pp. 4621–4631, Aug. 2015.

[19] Z. M. Fadlullah, D. Takaishi, H. Nishiyama, N. Kato, and R. Miura, "A dynamic trajectory control algorithm for improving the communication throughput and delay in UAV-aided networks," *IEEE Netw.*, vol. 30, no. 1, pp. 100–105, Jan./Feb. 2016.

[20] Q. Wu, L. Liu, and R. Zhang, "Fundamental trade-offs in communication and trajectory design for UAV-enabled wireless network," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 36–44, Feb. 2019.

[21] D. Yang, Q. Wu, Y. Zeng, and R. Zhang, "Energy tradeoff in ground-to-UAV communication via trajectory design," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6721–6726, Jul. 2018.

[22] J. Liu, X. Wang, B. Bai, and H. Dai, "Age-optimal trajectory planning for UAV-assisted data collection," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2018, pp. 553–558.

[23] W. Shi *et al.*, "3D multi-drone-cell trajectory design for efficient IoT data collection," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–6.

[24] X. Zhang and L. Duan, "Fast deployment of UAV networks for optimal wireless coverage," *IEEE Trans. Mobile Comput.*, vol. 18, no. 3, pp. 588–601, Mar. 2019.

[25] Y. Lin and S. Saripalli, "Sampling-based path planning for UAV collision avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3179–3192, Nov. 2017.

[26] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 5332–5339.

[27] K. Dogancay, "UAV path planning for passive emitter localization," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 2, pp. 1150–1166, Apr. 2012.

[28] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Trans, Wireless Commun.*, vol. 16, no. 6, pp. 3747–3760, Jun. 2017.

[29] P.-V. Mekikis and A. Antonopoulos, "Breaking the boundaries of aerial networks with charging stations," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–6.

[30] Z. Wang, L. Duan, and R. Zhang, "Adaptive deployment for UAV-aided communication networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4531–4543, Sep. 2019.

[31] X. Li, H. Yao, J. Wang, S. Wu, C. Jiang, and Y. Qian, "Rechargeable multi-UAV aided seamless coverage for QoS-guaranteed IoT networks," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 902–10 914, Dec. 2019.

[32] Q. Wu and R. Zhang, "Common throughput maximization in UAV-enabled OFDMA systems with delay consideration," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6614–6627, Dec. 2018.

[33] Q. Zhang, M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Machine learning for predictive on-demand deployment of UAVs for wireless communications," in *Proc. IEEE Global Commun. Conf.*, 2018, pp. 1–6.

[34] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in UAV enabled wireless sensor network," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 328–331, Jun. 2018.

[35] C. R. Ashokkumar and G. W. York, "Observer based controllers for UAV maneuver options," in *Proc. AIAA Guid. Navigation Control Conf.*, 2016, Art. no. 0643.

[36] A. Fotouhi, M. Ding, L. G. Giordano, M. Hassan, J. Li, and Z. Lin, "Joint optimization of access and backhaul links for UAVs based on reinforcement learning," in *Proc. IEEE Globecom Workshops*, 2019, pp. 1–6.

[37] A. Fotouhi, M. Ding, and M. Hassan, "Understanding autonomous drone maneuverability for internet of things applications," in *Proc. IEEE 18th Int. Symp. A World Wireless Mobile Multimedia Netw.*, 2017, pp. 1–6.

[38] 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.

[39] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *Proc. IEEE Global Commun. Conf.*, 2014, pp. 2898–2904.

[40] K. Li, W. Ni, X. Wang, R. P. Liu, S. S. Kanhere, and S. Jha, "Energy-efficient cooperative relaying for unmanned aerial vehicles," *IEEE Trans. Mobile Comput.*, vol. 15, no. 6, pp. 1377–1386, Jun. 2016.

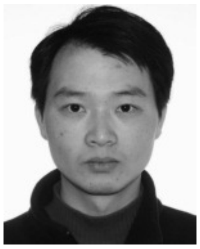
[41] I. Osband and B. Van Roy, "Near-optimal reinforcement learning in factored MDPs," in *Proc. Advances Neural Inf. Process. Syst.*, 2014, pp. 604–612.

[42] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.

[43] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[44] K. Li, W. Ni, E. Tovar, and A. Jamalipour, "On-board deep Q-network for UAV-assisted online power transfer and data collection," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 12 215–12 226, Dec. 2019.

**Kai Li** (Senior Member, IEEE) received the BE degree from Shandong University, China, in 2009, the MS degree from the Hong Kong University of Science and Technology, Hong Kong, in 2010, and the PhD degree in computer science from the University of New South Wales, Sydney, Australia, in 2014. Currently, he is a senior research scientist and project leader with Real-Time and Embedded Computing Systems Research Centre (CISTER), Portugal. He is also a research fellow with Carnegie Mellon Portugal Research Program. Prior to this, he was a postdoctoral research fellow with the SUTD-MIT International Design Centre, The Singapore University of Technology and Design, Singapore (2014-2016). He was a visiting research assistant with ICT Centre, CSIRO, Australia (2012-2013). From 2010 to 2011, he was a research assistant with Mobile Technologies Centre, Chinese University of Hong Kong. His research interests include vehicular communications and security, resource allocation optimization, cyber-physical systems, Internet of Things (IoT), human sensing systems, sensor networks, and UAV networks. He has been serving as an associate editor for the *IEEE Access Journal*, Demo Co-chair for ACM/IEEE IPSN 2018, the TPC member of IEEE Globecom'18, MASS'18, VTC-Spring'18, Globecom'17, VTC'17, and VTC'16.



**Wei Ni** (Senior Member, IEEE) received the BE and PhD degrees in electronic engineering from Fudan University, Shanghai, China, in 2000 and 2005, respectively. Currently, he is a group leader and principal research scientist with CSIRO, Sydney, Australia, and an adjunct professor with the University of Technology Sydney and an honorary professor with Macquarie University, Sydney. Prior to this, he was a postdoctoral research fellow with Shanghai Jiaotong University from 2005-2008, a deputy project manager with the Bell Labs, Alcatel/Alcatel-Lucent from 2005-2008, and a senior researcher with Devices R&D, Nokia from 2008-2009. His research interests include signal processing, stochastic optimization, as well as their applications to network efficiency and integrity. He is the chair of IEEE Vehicular Technology Society (VTS) New South Wales (NSW) Chapter since 2020 and an editor of the *IEEE Transactions on Wireless Communications* since 2018. He served first the secretary and then vice-chair of the IEEE NSW VTS Chapter from 2015-2019, track chair for VTC-Spring 2017, track co-chair for IEEE VTCSpring 2016, publication chair for BodyNet 2015, and student travel grant chair for WPMC 2014.

**Falko Dressler** (Fellow, IEEE) received the MSc and PhD degrees from the Department of Computer Science, University of Erlangen, in 1998 and 2003, respectively. He is currently a full professor and chair for Data Communications and Networking, School of Electrical Engineering and Computer Science, TU Berlin. He has been associate editor-in-chief for the *IEEE Transactions on Mobile Computing* and the *Elsevier Computer Communications,* as well as an editor for journals such as the *IEEE/ACM Transactions on Networking*, *IEEE Transactions on Network Science and Engineering*, *Elsevier Ad Hoc Networks*, and *Elsevier Nano Communication Networks*. He has been chairing conferences, such as IEEE INFOCOM, ACM MobiSys, ACM MobiHoc, IEEE VNC, IEEE GLOBECOM. He authored the textbooks Self-Organization in sensor and Actor Networks published by Wiley & Sons and Vehicular Networking published by Cambridge University Press. He has been an IEEE Distinguished Lecturer as well as an ACM Distinguished Speaker. He is also a ACM distinguished member. He is a member of the German National Academy of Science and Engineering (acatech). He has been serving on the IEEE COMSOC Conference Council and the ACM SIGMOBILE Executive Committee. His research interests include adaptive wireless networking (radio, visible light, molecular communications) and embedded system design (from microcontroller to Linux kernel) with applications in ad hoc and sensor networks, the Internet of Things, and cooperative autonomous driving systems.