Simultaneous Navigation and Radio Mapping for Cellular-Connected UAV with Deep Reinforcement Learning

Yong Zeng, Xiaoli Xu, Shi Jin, and Rui Zhang

Abstract

Cellular-connected unmanned aerial vehicle (UAV) is a promising technology to unlock the full potential of UAVs in the future by reusing the cellular base stations (BSs) to enable their air-ground communications. However, how to achieve ubiquitous three-dimensional (3D) communication coverage for the UAVs in the sky is a new challenge. In this paper, we tackle this challenge by a new *coverage-aware navigation* approach, which exploits the UAV's controllable mobility to design its navigation/trajectory to avoid the cellular BSs' coverage holes while accomplishing their missions. To this end, we formulate an UAV trajectory optimization problem to minimize the weighted sum of its mission completion time and expected communication outage duration, which, however, cannot be solved by the standard optimization techniques mainly due to the lack of an accurate and tractable end-to-end communication model in practice. To overcome this difficulty, we propose a new solution approach based on the technique of *deep reinforcement learning* (DRL). Specifically, by leveraging the state-of-the-art *dueling double deep Q network (dueling DDQN) with multi-step learning*, we first propose a UAV navigation algorithm based on direct RL, where the signal measurement at the UAV is used to directly train the *action-value function* of the navigation policy. To further improve the performance, we propose a new framework called *simultaneous navigation and radio mapping (SNARM)*, where the UAV's signal measurement is used not only for training the DQN directly, but also to create a radio map that is able to predict the outage probabilities at all locations in the area of interest. This thus enables the generation of simulated UAV trajectories and predicting their expected returns, which are then used to further train the DQN via *Dyna* technique, thus greatly improving the learning efficiency. Simulation results demonstrate the

Y. Zeng, X. Xu, and S. Jin are with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China. Y. Zeng is also with Purple Mountain Laboratories, Nanjing 211111, China (e-mail: {yong_zeng, xiaolixu, jinshi}@seu.edu.cn).

R. Zhang is with the Department of Electrical and Computer Engineering, National University of Singapore (e-mail: elezhang@nus.edu.sg).

Part of this work has been presented in IEEE Global Communications Conference, 9-13 December 2019, Waikoloa, HI, USA [1].

effectiveness of the proposed algorithms for coverage-aware UAV navigation, as well as the significantly improved performance of SNARM over direct RL.

Index Terms

Cellular-connected UAV, coverage-aware navigation, simultaneous navigation and radio mapping, deep reinforcement learning.

I. Introduction

Conventionally, cellular networks are designed to mainly serve terrestrial user equipments (UEs) with fixed infrastructure. With the continuous expansion of human activities towards the sky and the fast growing use of unmanned aerial vehicles (UAVs) for various applications, there have been increasing interests in integrating UAVs into cellular networks [2]. On one hand, dedicated UAVs could be dispatched as aerial base stations (BSs) or relays to assist wireless communications between devices without direct connectivity, or as flying access points (APs) for data collection and information dissemination, leading to a new three-dimensional (3D) network with *UAV-assisted communications* [3]. On the other hand, for those UAVs with their own missions, cellular network could be utilized to support their command and control (C&C) communication as well as payload data transmissions, corresponding to the other paradigm of *cellular-connected UAV* [4]. In particular, by reusing the existing densely deployed cellular BSs worldwide together with the advanced cellular technologies, cellular-connected UAV has the great potential to achieve truly remote UAV operation with unlimited range, not to mention other advantages such as the ease of legitimate UAV monitoring and management, high-capacity payload transmission, and cellular-enhanced positioning [2,4].

However, to practically realize the above vision of cellular-connected UAVs, there are still several critical challenges to be addressed. In particular, as existing cellular networks are mainly planned for ground coverage with BS antennas typically downtilted towards the ground [5], unlike that on the ground, ubiquitous cellular coverage in the sky cannot be guaranteed in general [6]. In fact, even for the forthcoming 5G or future 6G cellular networks that are planned to embrace the new type of aerial UEs, it is still unclear whether ubiquitous sky coverage is economically viable or not, even for some moderate range of altitude, considering various practical factors such as infrastructural and operational costs, as well as the anticipated aerial UE density in the

near future. Such coverage issue is exacerbated by the more severe aerial-ground interference as compared to the terrestrial counterpart [4,6,7], due to the high likelihood of having strong line of sight (LoS) channels for the UAVs with their non-associated co-channel BSs.

Fortunately, different from the conventional ground UEs whose mobility is typically random and uncontrollable, which renders ubiquitous ground coverage essential, the mobility of UAVs is more predictable and most of the time controllable, either autonomously by computer program or by remote pilots. This thus offers a new degree of freedom to circumvent the aforementioned coverage issue, via *coverage-aware navigation or trajectory design*: an approach that requires no or little modifications for cellular networks to serve aerial UEs. There are some preliminary efforts along this direction. In [8], by applying graph theory and convex optimization for cellular-connected UAV, the UAV trajectory is optimized to minimize the UAV travelling time while ensuring that it is always connected with at least one BS. A similar problem is studied in [9,10], by allowing certain tolerance for disconnection.

However, the conventional optimization-based UAV trajectory design like those mentioned above have practical limitations. First, formulating the corresponding optimization problems requires accurate and analytically tractable end-to-end communication models, including the BS antenna model, channel model, interference model and even local environmental model. Perhaps for such reasons, prior works such as [8]–[10] have mostly assumed some simplified models like isotropic radiation for antennas and/or free-space path loss channel model for aerial-ground links. Although there have been previous works considering more sophisticated channel models, like the probabilistic LoS channel model [11] and angle-dependent channel parameters [12,13], these are statistical models that can only predict the performance in the average sense without providing any performance guarantee for the local environment where the UAVs are actually deployed. Another limitation of off-line optimization-based trajectory design is the requirement of perfect and usually global knowledge of the channel model parameters, which is difficult to acquire in practice. Last but not least, even with the accurate channel model and the perfect information of all relevant parameters, most of these off-line optimization problems are highly non-convex and thus difficult to be efficiently solved.

To overcome the above limitations, we propose in this paper a new approach for coverage-aware UAV navigation based on reinforcement learning (RL) [14], which is one type of machine learning techniques for solving sequential decision problems. As shown in Fig. 1, we consider

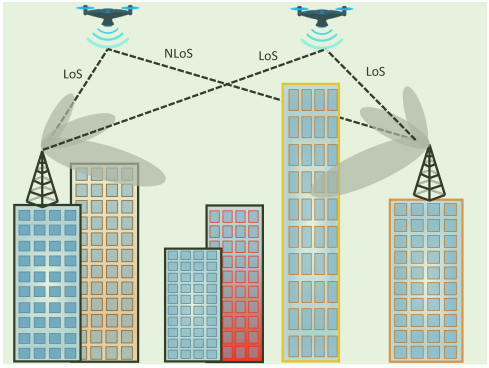


Fig. 1: Coverage-aware navigation for cellular-connected UAVs in urban environment.

a typical cellular-connected UAV system in urban environment, where multiple UAVs need to operate in a given airspace with their communications supported by cellular BSs. The UAV missions are to fly from their respective initial locations to a common final location with minimum time,[1] while maintaining a satisfactory communication connectivity with the cellular network with best effort. Our main contributions are summarized as follows:

- First, we formulate the UAV trajectory optimization problem to minimize the weighted sum of its mission completion time and the expected communication outage duration, which, however, is difficult to be solved via the standard optimization techniques mainly due to the lack of an accurate and tractable communication model. Fortunately, we show that the formulated problem can be transformed to an equivalent Markov decision process (MDP) for which a handful of RL techniques, such as the classic Q learning [14], can be applied to learn the UAV action policy, or the flying direction in our problem.
- As the resulting MDP involves continuous state space that essentially has infinite number of state-action pairs and thus renders the simple table-based Q learning inapplicable, we apply the celebrated deep neural network (DNN) to approximate the *action-value function* (or Q function), a technique commonly known as deep reinforcement learning (DRL) or more specifically, deep Q learning [15]. To train the DNN, we use the state-of-the-art dueling network architecture with double DQN (dueling DDQN) [16] and multi-step learning. Different from the conventional off-line optimization-based designs, the proposed DRL-based trajectory design does not require any prior knowledge about the channel model or propagation environment, but only utilizes the raw signal measurement at each UAV as

[1]The results of this paper can be extended to the general case of different final locations for the UAVs.

the input to improve its radio environmental awareness, as evident from the continuously improved estimation accuracy of the UAVs' action-value functions.

- As direct RL based on real experience or measured data only is known to be data inefficient, i.e., a large amount of data or agent-environment interaction is needed, we further propose a new design framework (termed *simultaneous navigation and radio mapping (SNARM)*) to improve the learning efficiency. Specifically, with SNARM, the network (say, a macrocell BS in the area of interest) maintains a database, also known as *radio map* [17,18], which provides the outage probability distribution over the space of interest with respect to the relatively stable (large-scale) channel parameters. As such, the obtained signal measurements as UAVs fly along their trajectories are used for dual purposes: to train the DQN to improve the estimation of the optimal action-value functions, as well as to improve the accuracy of the estimated radio map, which is used for the prediction of the outage probability for other locations in the considered area even if they have not been visited by any UAV yet. The exploitation of the radio map, whose accuracy is continuously improved as more signal measurements are accumulated, makes it possible to generate simulated UAV trajectories and predict their corresponding returns, without having to actually fly the UAVs along them. This thus leads to our proposed SNARM framework, by applying the classic *Dyna* [14] dueling DDQN with multi-step learning to train the action-value functions with a combined use of real trajectories and simulated trajectories, thus greatly reducing the actual UAV flights or measurement data required.
- Numerical results are provided, with the complete source codes available online[2], to verify the performance of the proposed algorithms. It is observed that the SNARM algorithm is able to learn the radio map very efficiently, and both direct RL and SNARM algorithms are able to effectively navigate UAVs to avoid the weak coverage regions of the cellular network. Furthermore, thanks to the simultaneous learning of the radio map, SNARM is able to significantly reduce the number of required UAV flights (or learning episodes) than direct RL while achieving comparable performance.

Note that applying RL techniques for UAV communications has received significant attention recently [19]–[23] and DRL has also been used in wireless networks [24] for solving various

[2]URL ....

problems like multiple access control [25], power allocation [26], modulation and coding [27], mobile edge computing [28], UAV BS placement [29], and UAV interference management [30], among others. On the other hand, local environmental-aware UAV path/trajectory design has been studied from different perspectives, such as that based on the known radio map [31] or 3D city map [32,33], or the joint online and off-line optimization for UAV-enabled data harvesting [34]. Compared to such existing path/trajectory designs, the proposed coverage-aware navigation algorithms based on direct RL or SNARM do not require any prior knowledge or assumption about the environment, but only utilize the UAV's measured data as the input for action and/or radio map learning, thus expected to be more robust to any imperfect knowledge or change of the local environment. Although large data measurements are generally required for the proposed algorithms, they are practically available according to the existing cellular communication protocols, such as the reference signal received power (RSRP) and reference signal received quality (RSRQ) measurements. Besides, as different UAVs in the same region share the same local radio propagation environment, regardless of operating concurrently or at different time, their measured data as they perform their respective missions can actually be collectively utilized for our proposed designs, to continuously improve the quality of the learning for the future missions, which thus greatly alleviates the burden for data acquisition in practice.

The rest of the paper is organized as follows. Section II introduces the system model and problem formulation. In Section III, we give a very brief background about RL and DRL, mainly to introduce the key concepts and main notations. Section IV presents the proposed algorithms, including the direct RL based on dueling DDQN with multi-step learning for UAV navigation, and SNARM to enable path learning with both real and simulated flight experience. Section V presents the numerical results, and finally we conclude the paper in Section VI.

II. System Model and Problem Formulation

As shown in Fig. 1, we consider a cellular-connected UAV system with arbitrary number of UAV users. Without loss of generality, we assume that the airspace of interest is a cubic volume, which is specified by $[x_L, x_U] \times [y_L, y_U] \times [z_L, z_U]$, with the subscripts L and U respectively denoting the lower and upper bounds of the 3D coordinate of the airspace. UAVs with various missions need to operate in this airspace with communications supported by cellular BSs. While different UAVs usually have different missions and may operate at different time, since they share

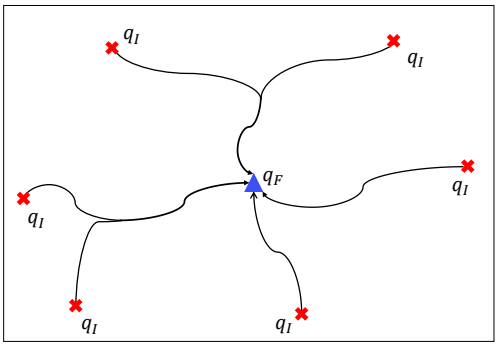


Fig. 2: UAVs with various initial locations fly to a common destination.

the common airspace, their flight experience including their signal measurements along their respective flying trajectories, can be exploited collectively to improve their radio environmental awareness. To make the problem concrete, we consider the scenario that the UAVs need to fly from their respective initial locations $\mathbf{q}_I \in \mathbb{R}^{3\times 1}$, which are generally different for different UAVs, to a common final location $\mathbf{q}_F \in \mathbb{R}^{3\times 1}$, as shown in Fig. 2. This setup corresponds to various UAV applications such as UAV-based parcel collection, with $\mathbf{q}_I$ corresponding to the customer's home location and $\mathbf{q}_F$ being the local processing station of courier company, or UAV-based aerial inspection, with $\mathbf{q}_I$ being the target point for inspection and $\mathbf{q}_F$ being the charging station in the local area, etc. We assume that collision/obstacle avoidance is guaranteed by e.g., ensuring that UAVs are sufficiently separated in operation time or space, and their flight altitudes are higher than the maximum building height. Without loss of generality, we can focus on one typical UAV, since other UAVs may apply similar strategies to update the common location-dependent database, such as the action-value function of the RL algorithms and the radio map of the environment (to be defined later). The UAV needs to be navigated from its initial location $\mathbf{q}_I$ to the final location $\mathbf{q}_F$ with the minimum mission completion time, while maintaining a satisfactory communication connectivity with the cellular network. Let T denote the mission completion time and $\mathbf{q}(t) \in \mathbb{R}^{3\times 1}$, $t \in [0, T]$, represent the UAV trajectory. Then we should have

$$\mathbf{q}(0) = \mathbf{q}_I, \ \mathbf{q}(T) = \mathbf{q}_F, \tag{1}$$

$$\mathbf{q}_L \preceq \mathbf{q}(t) \preceq \mathbf{q}_U, \forall t \in (0, T), \tag{2}$$

where $\mathbf{q}_L \triangleq [x_L; y_L; z_L]$ and $\mathbf{q}_U \triangleq [x_U; y_U; z_U]$, with $\preceq$ denoting the element-wise inequality.

Let M denote the total number of cells in the area of interest, and $h_m(t)$, $1 \le m \le M$, represent the baseband equivalent channel from cell m to the UAV at time t, which generally

depends on the BS antenna gains, the large-scale path loss and shadowing that are critically dependent on the local environment, as well as the small-scale fading. The received signal power at time t by the UAV from cell m can be written as

$$p_m(t) = \bar{P}_m |h_m(t)|^2 \tag{3}$$

$$= \bar{P}_m \beta_m(\mathbf{q}(t)) G_m(\mathbf{q}(t)) \tilde{h}_m(t), \ m = 1, \cdots, M, \tag{4}$$

where $\bar{P}_m$ denotes the transmit power of cell m that is assumed to be a constant, $\beta_m(\cdot)$ and $G_m(\cdot)$ denote the large-scale channel gain and the BS antenna gain, respectively, which generally depend on the UAV location $\mathbf{q}(t)$, and $\tilde{h}_m(t)$ is a random variable accounting for the small-scale fading. As the proposed designs in this paper work for arbitrary channels, a detailed discussion of one particular BS-UAV channel model adopted for our simulations is deferred to Section V.

Denote by $b(t) \in \{1, \cdots, M\}$ the cell associated to the UAV at time t. We say that the UAV is in outage at time t if its received signal-to-interference ratio (SIR) is below a certain threshold γ_{th}, i.e., when $\text{SIR}(t) < \gamma_{\text{th}}$, where

$$\text{SIR}(t) \triangleq \frac{p_{b(t)}(t)}{\sum_{m \neq b(t)} p_m(t)}. \tag{5}$$

Note that for simplicity, we have ignored the noise effect, since BS-UAV communication is known to suffer from more severe interference than the terrestrial counterparts, and thus its communication performance is usually interference-limited. Besides, we consider the worst-case scenario with universal frequency reuse so that all these non-associated BSs would contribute to the sum-interference term in (5). The extension of the proposed techniques for dynamic interference will be an interesting work for further investigation.

Due to the randomness of small-scale fading, for any given UAV location $\mathbf{q}(t)$ and cell association $b(t)$ at time t, $\text{SIR}(t)$ in (5) is a random variable, and the resulting outage probability is a function of $\mathbf{q}(t)$ and $b(t)$, i.e.,

$$P_{\text{out}}(\mathbf{q}(t), b(t)) \triangleq \Pr\left(\text{SIR}(t) < \gamma_{\text{th}}\right), \tag{6}$$

where $\Pr(\cdot)$ is the probability of the event taken with respect to the randomness of small-scale fading $\{\tilde{h}_m(t)\}_{m=1}^M$. Then the total expected outage duration across the UAV's mission duration T is

$$\bar{T}_{\text{out}}(\{\mathbf{q}(t), b(t)\}) = \int_0^T P_{\text{out}}(\mathbf{q}(t), b(t)) dt. \tag{7}$$

Intuitively, with longer mission completion time T, the UAV is more flexible to adapt its

trajectory to avoid the weak coverage regions of the cellular network in the sky and thus reduce the expected outage duration $\bar{T}_{\text{out}}$. Therefore, there in general exists a tradeoff between minimizing the mission completion time T and expected outage duration $\bar{T}_{\text{out}}$, which can be balanced by designing $\{\mathbf{q}(t)\}$ and $\{b(t)\}$ to minimize the weighted sum of these two metrics with certain weight $\mu \geq 0$, as formulated in the following optimization problem.

$$(\text{P0}): \min_{T,\{\mathbf{q}(t),b(t)\}} \; T + \mu \bar{T}_{\text{out}}(\{\mathbf{q}(t), b(t)\})$$

$$\text{s.t. } \|\dot{\mathbf{q}}(t)\| \leq V_{\max}, \; \forall t \in [0, T], \tag{8}$$

$$\mathbf{q}(0) = \mathbf{q}_I, \; \mathbf{q}(T) = \mathbf{q}_F, \tag{9}$$

$$\mathbf{q}_L \preceq \mathbf{q}(t) \preceq \mathbf{q}_U, \; \forall t \in [0, T], \tag{10}$$

$$b(t) \in \{1, \cdots, M\}, \tag{11}$$

where $V_{\max}$ denotes the maximum UAV speed. It can be shown that at the optimal solution to (P0), the UAV should fly with the maximum speed $V_{\max}$, i.e., $\dot{\mathbf{q}}(t) = V_{\max}\vec{\mathbf{v}}(t)$, where $\vec{\mathbf{v}}(t)$ denotes the UAV flying direction with $\|\vec{\mathbf{v}}(t)\| = 1$. Thus, (P0) can be equivalently written as

$$(\text{P1}): \min_{T,\{\mathbf{q}(t),\vec{\mathbf{v}}(t),b(t)\}} \; T + \mu \bar{T}_{\text{out}}(\{\mathbf{q}(t), b(t)\})$$

$$\text{s.t. } \dot{\mathbf{q}}(t) = V_{\max}\vec{\mathbf{v}}(t), \; \forall t \in [0, T], \tag{12}$$

$$\|\vec{\mathbf{v}}(t)\| = 1, \; \forall t \in [0, T], \tag{13}$$

$$(9) - (11).$$

In practice, finding the optimal UAV trajectory and cell association by directly solving the optimization problem (P1) faces several challenges. First, to complete the problem formulation of (P1) to make it ready for optimization, we still need to derive the closed-form expression for the expected outage duration $\bar{T}_{\text{out}}(\{\mathbf{q}(t), b(t)\})$ for any given UAV location $\mathbf{q}(t)$ and association $b(t)$, which is challenging, if not impossible. On one hand, this requires accurate and tractable modelling of the end-to-end channel $h_m(t)$ for cellular-UAV communication links, where practical 3D BS antenna pattern [35] and the frequent variations of LoS/NLoS connections, which depends on the actual building/terrain distributions in the area of interest, need to be considered. On the other hand, even with such modelling and the perfect knowledge of the terrain/building information for the area of interest, deriving the closed-form expression for $\bar{T}_{\text{out}}(\{\mathbf{q}(t), b(t)\})$ is mathematically difficult since the probability density function (pdf) of the SIR depends on the

UAV trajectory $\mathbf{q}(t)$ in sophisticated manners, as can be inferred from (4) and (5). Furthermore, even after completing the problem formulation of $(\mathrm{P1})$ with properly defined cost function, the resulting optimization problem will be highly non-convex and difficult to be efficiently solved.

To overcome the above issues, in the following, we propose a novel approach for coverage-aware UAV navigation by leveraging the powerful mathematical framework of RL and DNN, which only requires the UAV signal measurements along its flight as the input. By leveraging the state-of-the-art dueling DDQN with multi-step learning [36], we first propose the direct RL-based algorithm for coverage-aware UAV navigation, and then introduce the more advanced SNARM algorithm to improve the learning efficiency and UAV performance.

III. Overview of Deep Reinforcement Learning

This section aims to give a brief overview on RL and DRL, by introducing the key notations to be used in the sequel of this paper. The readers are referred to the classic textbook [14] for a more comprehensive description.

A. Basics of Reinforcement Learning

RL is a useful machine learning method to solve the MDP [14], which consists of an *agent* interacting with the *environment* iteratively. For the fully observable MDP, at each discrete time step n, the agent observes a state S_n, takes an action A_n, and then receives an immediate reward R_n and transits to the next state S_{n+1}. Mathematically, an MDP can be specified by 4-tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} >$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the state transition probability, with $P(s'|s,a)$ specifying the probability of transiting to the next state $s' \in \mathcal{S}$ given the current state $s \in \mathcal{S}$ after applying the action $a \in \mathcal{A}$, and $\mathcal{R}$ is the immediate reward received by the agent, usually denoted by R_n for reward received at time step n or $R(s,a)$ to show its general dependency on s and a.

The agent's actions are governed by its policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$, where $\pi(a|s)$ gives the probability of taking action $a \in \mathcal{A}$ when in state $s \in \mathcal{S}$. The goal of the agent is to improve its policy π based on its experience, so as to maximize its long-term expected *return* $\mathbb{E}[G_n]$, where the return $G_n \triangleq \sum_{k=0}^{\infty} \gamma^k R_{n+k}$ is the accumulated discounted reward from time step n onwards with a discount factor $0 \leq \gamma \leq 1$.

A key metric of RL is the *action-value function*, denoted as $Q_\pi(s,a)$, which is the expected return starting from state s, taking the action a, and following policy π thereafter,

i.e., $Q_\pi(s,a) = \mathbb{E}_\pi[G_n|S_n = s, A_n = a]$. The optimal action-value function is defined as $Q_*(s,a) = \max_\pi Q_\pi(s,a)$, $\forall s \in \mathcal{S}$ and $a \in \mathcal{A}$. If the optimal value function $Q_*(s,a)$ is known, the optimal policy $\pi_*(a|s)$ can be directly obtained as

$$\pi_*(a|s) = \begin{cases} 1, & \text{if } a = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q_*(s,a), \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Thus, an essential task of RL is to obtain the optimal value functions, which satisfy the celebrated Bellman optimality equation [14]

$$Q_*(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) \max_{a' \in \mathcal{A}} Q_*(s',a'), \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{15}$$

Bellman optimality equation is non-linear, and admits no closed-form solution in general. However, many iterative solutions have been proposed. Specifically, when the agent has the full knowledge of the MDP, including the reward $\mathcal{R}$ and the state-transition probabilities $\mathcal{P}$, the optimal value function can be obtained recursively based on dynamic programming, such as *value iteration* [14]. On the other hand, when the agent has no or incomplete prior knowledge about the MDP, it may apply the useful idea of *temporal difference* (TD) learning to improve its estimation of the value function by directly interacting with the environment. Specifically, TD learning is a class of *model-free* RL methods that learn the value functions based on the direct samples of the *state-action-reward-nextState* sequence, with the estimation of the value functions updated by the concept of *bootstrapping*. One classic TD learning method is *Q-learning*, which applies the following update to the action-value function with the observed sample reward-and-state transitions (S_n, A_n, R_n, S_{n+1}):

$$Q(S_n, A_n) \leftarrow Q(S_n, A_n) + \alpha \left[R_n + \gamma \max_{a \in \mathcal{A}} Q(S_{n+1}, a) - Q(S_n, A_n) \right],$$

where α is the learning rate. It has been shown that Q-learning is able to converge to the optimal action-value function Q_* if each state-action pair is visited by sufficient times and appropriate learning rate α is chosen [14].

B. Deep Reinforcement Learning

The RL learning method discussed in Section III-A is known as *table-based*, which requires storing one value for each state-action pair, and the value is updated only when that state-action pair is actually experienced. This becomes impractical for continuous state/action or when the number of discretized states/actions is very large. In order to practically apply RL algorithms to

large/continuous state-action space, one may resort to the useful technique of *function approximation* [14], where the action-value function is approximated by certain parametric function, e.g., $Q(s,a) \approx \hat{Q}(s,a;\boldsymbol{\theta})$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$, with a parameter vector $\boldsymbol{\theta}$. Function approximation brings two advantages over table-based RL. Firstly, instead of storing and updating the value functions for all state-action pairs, one only needs to learn the parameter $\boldsymbol{\theta}$, which typically has much lower dimension than the number of state-action pairs. Secondly, function approximation enables *generalization*, i.e., the ability to predict the values even for those state-action pairs that have never been experienced, since different state-action pairs are coupled with each other via the function $\hat{Q}(s,a;\boldsymbol{\theta})$ and parameter $\boldsymbol{\theta}$.

One powerful non-linear function approximation technique is via using artificial neural networks (ANNs), which are networks consisting of interconnected units (or neurons), with each unit computing a weighted sum of their input signals and then applying a nonlinear function (called the *activation function*) to produce its output. It has been theoretically shown that an ANN with even a single hidden layer containing a sufficiently large number of neurons is a universal function approximation [14], i.e., it can approximate any continuous function on a compact region to any degree of accuracy. Nonetheless, researchers have found that ANNs with deep architectures consisting of many hidden layers are of more practical usage for complex function approximations. A combination of RL with function approximation based on deep ANNs leads to the powerful framework of DRL [15].

With deep ANN for the action-value function approximation $Q(s,a) \approx \hat{Q}(s,a;\boldsymbol{\theta})$, the parameter vector $\boldsymbol{\theta}$ actually corresponds to the weight coefficients and bias of all links in the ANNs. In principle, $\boldsymbol{\theta}$ can be updated based on the classic backpropagation algorithm [37]. However, its application for DRL faces new challenges, since standard ANNs are usually trained in the *supervised learning* manner, i.e., the labels (in this case the true action values) of the input training data are known, which is obviously not the case for DRL. This issue can be addressed by the idea of *bootstrapping*, i.e., for each given *state-action-reward-nextState* transition (S_n, A_n, R_n, S_{n+1}), the parameter of the network $\boldsymbol{\theta}$ is updated to minimize the loss given by

$$\left(R_n + \gamma \max_{a \in \mathcal{A}} \hat{Q}(S_{n+1}, a; \boldsymbol{\theta}) - \hat{Q}(S_n, A_n; \boldsymbol{\theta})\right)^2. \tag{16}$$

However, as the target in (16) also depends on the parameter $\boldsymbol{\theta}$ to be actually updated, directly

applying the standard deep training algorithms with (16) can lead to oscillations or even divergence. This problem was addressed in the celebrated work [15], which proposed the simple but effective technique of "target network" to bring the Q-learning update closer to the standard supervised-learning setup. A target network, denoted as $\hat{Q}(s,a;\boldsymbol{\theta}^-)$ with parameter $\boldsymbol{\theta}^-$, is a copy of the Q network $\hat{Q}(s,a;\boldsymbol{\theta})$, but with its parameter $\boldsymbol{\theta}^-$ updated much less frequently. For example, after every B number of updates to the weights $\boldsymbol{\theta}$ of the Q network, we may set $\boldsymbol{\theta}^-$ equal to $\boldsymbol{\theta}$ and keep it unchanged for the next B updates of $\boldsymbol{\theta}$. Thus, the loss in (16) is modified to

$$\left(R_n + \gamma \max_{a\in\mathcal{A}} \hat{Q}(S_{n+1},a;\boldsymbol{\theta}^-) - \hat{Q}(S_n,A_n;\boldsymbol{\theta})\right)^2. \tag{17}$$

This helps to keep the target remaining relatively stable and hence improve the convergence property of the training.

Another essential technique proposed in [15] is *experience replay*, which stores the *state-action-reward-nextState* transition (S_n, A_n, R_n, S_{n+1}) into a replay memory and then randomly access them later to perform the weight updates. Together with mini-batch update, i.e., multiple experiences are sampled uniformly at random from the replay memory for each update, experience replay helps to reduce the variance of the updates by avoiding highly correlated data for successive updates.

Soon after the introduction of DQN in [15], many techniques for further performance improvement have been proposed, such as DDQN [38] to address the over-estimation bias of the maximum operation of the Q learning, DQN with prioritized experience replay (PER) [39], DQN with the dueling network architecture [16], and a combination of all major improvement techniques, termed *Rainbow*, has been introduced in [36].

IV. Proposed Algorithms

In this section, we present our proposed DRL-based algorithms for solving the coverage-aware UAV navigation problem (P1).

A. Coverage-Aware UAV Navigation as an MDP

The first step to apply RL algorithms for solving a real-world problem is to reformulate it as an MDP. As MDP is defined over discrete time steps, for the UAV navigation and cell association problem $(\mathrm{P1})$, we need to first discretize the time horizon $[0,T]$ into N time steps with certain

interval Δ_t, where $T = N\Delta_t$. Note that Δ_t should be small enough so that within each time step, the distance between the UAV and any BS in the area of interest are approximately unchanged, or their large-scale channel gain and the BS antenna gain towards the UAV, i.e., $\beta_m(\mathbf{q}(t))$ and $G_m(\mathbf{q}(t))$ in (4), remain approximately constant. As such, the UAV trajectory $\{\mathbf{q}(t)\}$ can be approximately represented by the sequence $\{\mathbf{q}_n\}_{n=1}^N$, thus (12) and (13) can be rewritten as

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta_s \vec{\mathbf{v}}_n, \ \forall n, \tag{18}$$

$$\|\vec{\mathbf{v}}_n\| = 1, \ \forall n, \tag{19}$$

where $\Delta_s = V_{\max}\Delta_t$ is the UAV (maximum) displacement per time step, and $\vec{\mathbf{v}}_n \triangleq \vec{\mathbf{v}}(n\Delta_t)$ denotes the UAV flying direction at time step n. Furthermore, as cell association is usually based on large-scale channel gains with BSs to avoid too frequent handover, the associated cell thus remains unchanged within each time step. Thus, the association policy $b(t)$ can be represented in discrete time as b_n. As a result, the expected outage duration (7) can be approximately rewritten as

$$\bar{T}_{\text{out}}(\{\mathbf{q}(t), b(t)\}) \approx \Delta_t \sum_{n=1}^{N} P_{\text{out}}(\mathbf{q}_n, b_n). \tag{20}$$

For each time step n with given UAV location $\mathbf{q}_n$ and cell association b_n, deriving a closed-form expression for $P_{\text{out}}(\mathbf{q}_n, b_n)$ is difficult as previously mentioned in Section II. Fortunately, this issue can be circumvented by noting the fact that the time step Δ_t, which corresponds to UAV trajectory discretization based on the large-scale channel variation only, is actually in a relatively large time scale (say fractions of a second) and typically contains a large number of channel coherence blocks (say on the order of milliseconds) due to the small-scale fading. This thus provides a practical method to evaluate (20) numerically based on the raw signal measurement at the UAV. To this end, for given $\mathbf{q}_n$ and b_n at time step n, we first denote the instantaneous SIR in (5) as $\text{SIR}(\mathbf{q}_n, b_n; \tilde{h})$, where $\tilde{h}$ includes all the random small-scale fading coefficients with the M cells in (4). Further define an indicator function $I(\cdot)$ as

$$I(\mathbf{q}, b; \tilde{h}) = \begin{cases} 1, & \text{if } \text{SIR}(\mathbf{q}, b; \tilde{h}) < \gamma_{\text{th}} \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

Then we have

$$P_{\text{out}}(\mathbf{q}_n, b_n) = \Pr\left(\text{SIR}(\mathbf{q}_n, b_n; \tilde{h}) < \gamma_{\text{th}}\right) \tag{22}$$

$$= \mathbb{E}_{\tilde{h}}\left[I(\mathbf{q}_n, b_n; \tilde{h})\right]. \tag{23}$$

Assume that within each time step n, the UAV performs J SIR measurements for each of the M cell associations, which could be achieved by leveraging the existing soft handover mechanisms with continuous RSRP and RSRQ reports. Denote the jth SIR measurement of time step n with cell association $b_n \in \{1, \cdots, M\}$ as $\mathrm{SIR}(\mathbf{q}_n, b_n; \tilde{h}[n,j])$, where $\tilde{h}[n,j]$ denotes the realization of the small-scale fading. The corresponding outage indicator value, denoted as $I(\mathbf{q}_n, b_n; \tilde{h}[n,j])$, can be obtained accordingly based on (21). Then the empirical outage probability can be obtained as

$$\hat{P}_{\text{out}}(\mathbf{q}_n, b_n) \triangleq \frac{1}{J}\sum_{j=1}^{J} I(\mathbf{q}_n, b_n; \tilde{h}[n,j]). \tag{24}$$

Then by applying the Law of Large Numbers to (23), we have $\lim_{J\to\infty} \hat{P}_{\text{out}}(\mathbf{q}_n, b_n) = P_{\text{out}}(\mathbf{q}_n, b_n)$. Therefore, as long as the UAV performs signal measurements sufficiently frequently so that $J \gg 1$, $P_{\text{out}}(\mathbf{q}_n, b_n)$ in (20) can be evaluated by its empirical value $\hat{P}_{\text{out}}(\mathbf{q}_n, b_n)$.

Based on the above discussion, (P1) can be approximated as

$$(\text{P2}): \max_{N, \{\mathbf{q}_n, \vec{\mathbf{v}}_n, b_n\}} \quad -N - \mu \sum_{n=1}^{N} \hat{P}_{\text{out}}(\mathbf{q}_n, b_n)$$

$$\text{s.t. } \mathbf{q}_{n+1} = \mathbf{q}_n + \Delta_s \vec{\mathbf{v}}_n, \ \forall n, \tag{25}$$

$$\|\vec{\mathbf{v}}_n\| = 1, \ \forall n, \tag{26}$$

$$\mathbf{q}_0 = \mathbf{q}_I, \mathbf{q}_N = \mathbf{q}_F, \tag{27}$$

$$\mathbf{q}_L \preceq \mathbf{q}_n \preceq \mathbf{q}_U, \forall n, \tag{28}$$

$$b_n \in \{1, \cdots, M\}. \tag{29}$$

Note that we have dropped the constant coefficient Δ_t in the objective of (P2). Obviously, the optimal cell association policy to (P2) can be easily obtained as $b_n^* = \arg\min_{b\in\{1,\cdots,M\}} \hat{P}_{\text{out}}(\mathbf{q}_n, b)$. With a slight abuse of notation, let

$$\hat{P}_{\text{out}}(\mathbf{q}_n) = \min_{b\in\{1,\cdots,M\}} \hat{P}_{\text{out}}(\mathbf{q}_n, b), \tag{30}$$

then (P2) reduces to

$$(\text{P3}): \max_{N, \{\mathbf{q}_n, \vec{\mathbf{v}}_n\}} \quad -N - \mu \sum_{n=1}^{N} \hat{P}_{\text{out}}(\mathbf{q}_n)$$

$$\text{s.t. (25)} - \text{(28)}.$$

As such, a straightforward mapping of (P3) to an MDP $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} >$ is obtained as follows

- $\mathcal{S}$: the state corresponds to the UAV location $\mathbf{q}_n$, and the state space constitutes all possible

UAV locations within the feasible region, i.e., $\mathcal{S} = \{\mathbf{q} : \mathbf{q}_L \preceq \mathbf{q} \preceq \mathbf{q}_U\}$.

- $\mathcal{A}$: the action space corresponds to the UAV flying direction, i.e., $\mathcal{A} = \{\vec{\mathbf{v}} : \|\vec{\mathbf{v}}\| = 1\}$.
- $\mathcal{P}$: the state transition probability is deterministic governed by (25).
- $\mathcal{R}$: the reward $R(\mathbf{q}) = -1 - \mu \hat{P}_{\text{out}}(\mathbf{q})$, so that every time step that the UAV takes would incur a penalty of 1, and an additional penalty with weight μ if the UAV enters into a location with certain outage probability. This will thus encourage the UAV to reach the destination $\mathbf{q}_F$ as soon as possible, while avoiding the weak coverage regions of the cellular network.

With the above MDP formulation, it is observed that the objective function of (P3) corresponds to the un-discounted (i.e., $\gamma = 1$) accumulated rewards over one *episode* up to time step N, i.e., $G_1 = \sum_{n=1}^{N} R_n$. This corresponds to one particular form of MDP, namely the *episodic tasks* [14], which are tasks containing a special state called the *terminal state* that separates the agent-environment interactions into *episodes*. For episodic tasks, each episode ends with the terminal state, followed by a random reset to the starting state to start the new episode. For the UAV navigation problem (P3), the terminal state corresponds to $\mathbf{q}_F$ and the start state is $\mathbf{q}_I$. After being formulated as an MDP, (P3) can be solved by applying various RL algorithms. In the following, we first introduce a solution based on the state-of-the-art dueling DDQN with multi-step learning, and then propose the SNARM framework to improve the learning efficiency.

B. Dueling DDQN Multi-Step Learning for UAV Navigation

Both the state and action spaces for the MDP defined in Section IV-A are continuous. The most straightforward approach for handling continuous state-action spaces is to discretize them. However, a naive discretization of both state and action spaces either leads to poor performance due to the discretization errors and/or results in a prohibitively large number of state-action pairs due to the curse of dimensionality [14]. In this paper, we only discretize the action space $\mathcal{A}$ while keeping the state space $\mathcal{S}$ as continuous, and use the powerful ANN to approximate the action-value function. By uniformly discretizing the action space $\mathcal{A}$ (i.e., the flying directions) into K values, we have $\hat{\mathcal{A}} = \{\vec{\mathbf{v}}^{(1)}, \cdots, \vec{\mathbf{v}}^{(K)}\}$.

With the above action space discretization, the action-value function $Q(\mathbf{q}, \vec{\mathbf{v}})$ contains the continuous state input $\mathbf{q} \in \mathcal{S}$ and the discrete action input $\vec{\mathbf{v}} \in \hat{\mathcal{A}}$. We use the state-of-the-art DQN with the dueling network architecture [16] to approximate $Q(\mathbf{q}, \vec{\mathbf{v}})$, as illustrated in Fig. 3. At each time step n, the input of the network corresponds to the state (or the UAV

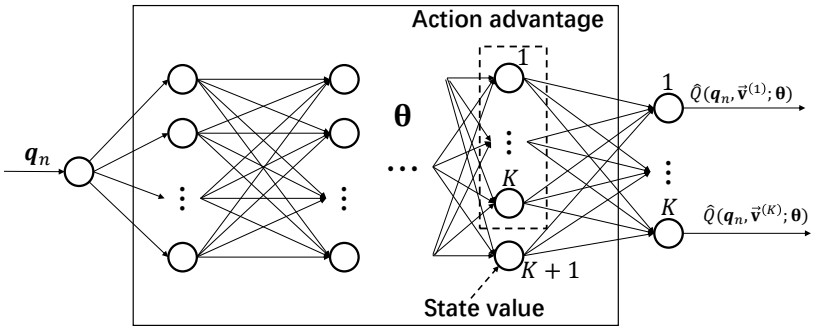


Fig. 3: Dueling DQN for coverage-aware UAV navigation.

location) $\mathbf{q}_n \in \mathcal{S}$, and it has K outputs, each corresponding to one action in $\hat{\mathcal{A}}$. Note that the distinguishing feature of dueling DQN is to first separately estimate the state values and the state-dependent *action advantages*, and then combine them in a smart way via an aggregating layer to give an estimate of the action value function Q [16]. Compared to the standard DQN, dueling DQN is able to learn the state-value function more efficiently, and is also more robust to the approximation errors when the action-value gaps between different actions of the same state are small [16]. The coefficients of the dueling DQN are denoted as $\boldsymbol{\theta}$, which is trained so that the output $\hat{Q}(\mathbf{q}, \vec{\mathbf{v}}; \boldsymbol{\theta})$ gives good approximations to the true action-value function $Q(\mathbf{q}, \vec{\mathbf{v}})$.

To train the dueling DQN, besides the standard techniques of *experience replay* and *target network* as discussed in Section III-B, we also apply the multi-step DDQN learning techniques [38]. Specifically, define the truncated N_1-step return from a given state $\mathbf{q}_n$ as

$$R_{n:n+N_1} = \sum_{i=0}^{N_1-1} \gamma^i R_{n+i+1}. \tag{31}$$

Then a multi-step DQN learning is to minimize the loss given by [14]

$$\left(R_{n:n+N_1} + \gamma^{N_1} \max_{\vec{\mathbf{v}}' \in \hat{\mathcal{A}}} \hat{Q}(\mathbf{q}_{n+N_1}, \vec{\mathbf{v}}'; \boldsymbol{\theta}^-) - \hat{Q}(\mathbf{q}_n, \vec{\mathbf{v}}_n; \boldsymbol{\theta}) \right)^2. \tag{32}$$

It is known that multi-step learning with appropriately chosen N_1 usually leads to faster learning [14].

Furthermore, double Q learning has been widely used to address the overestimation bias of the maximum operation in (32), via using two separate networks for the action selection and value evaluation of bootstrapping. Specifically, with DDQN, the loss of the multi-step learning in (32) is changed to

$$\left(R_{n:n+N_1} + \gamma^{N_1} \hat{Q}(\mathbf{q}_{n+N_1}, \vec{\mathbf{v}}^*; \boldsymbol{\theta}^-) - \hat{Q}(\mathbf{q}_n, \vec{\mathbf{v}}_n; \boldsymbol{\theta}) \right)^2, \tag{33}$$

where

$$\vec{\mathbf{v}}^* = \arg\max_{\vec{\mathbf{v}}' \in \hat{\mathcal{A}}} \hat{Q}(\mathbf{q}_{n+N_1}, \vec{\mathbf{v}}'; \boldsymbol{\theta}). \tag{34}$$

Note that we have used the target network with coefficients $\boldsymbol{\theta}^-$ to evaluate the bootstraping action in (33) while using the Q network with coefficients $\boldsymbol{\theta}$ for action selection in (34).

The proposed algorithm for coverage-aware UAV navigation with dueling DDQN multi-step learning is summarized in Algorithm 1. Note that the main steps of Algorithm 1 follow from the classic DQN algorithms in [15], except the following modifications. First, the more advanced dueling DQN architecture is used, instead of DQN. Second, to enable multi-step learning, a sliding window queue with capacity N_1 as in steps 6 and 11 is used to store the N_1 latest transitions. Therefore, at the current time step n, the UAV can calculate the accumulated return from the previous N_1 time steps, as in step 12 of Algorithm 1. Third, a double DQN learning is used in step 14.

While theoretically, the convergence of Algorithm 1 is guaranteed [14], in practice, a random initialization of the dueling DQN may require a prohibitively large number of time steps for the UAV to reach the destination $\mathbf{q}_F$. Intuitively, $\boldsymbol{\theta}$ should be initialized in a way such that in the early episodes when the UAV has no or limited knowledge about the radio environment, it should be encouraged to select actions for the shortest path flying. Thus, we propose the *distance-based* initialization for Algorithm 1, with $\boldsymbol{\theta}$ initialized so that $\hat{Q}(\mathbf{q}, \vec{\mathbf{v}}; \boldsymbol{\theta})$ approximates $-\|\mathbf{q}' - \mathbf{q}_F\|$, where $\mathbf{q}'$ is the next state with current state $\mathbf{q}$ and action $\vec{\mathbf{v}}$. Note that as the distance from any location $\mathcal{S}$ to the destination $\mathbf{q}_F$ is known *a priori*, the above initialization can be performed by sampling some locations in $\mathcal{S}$ without requiring any interaction with the environment. With such an initialization, it can be obtained from step 9 of Algorithm 1 that the UAV will choose the shortest-path action in the first episode with probability $1 - \epsilon$, and random exploration with probability ϵ. As the UAV accumulates more experience after some episodes so that the dueling DQN has been substantially updated, the ϵ-greedy policy in step 9 would generate UAV flying path that achieves a balance between minimizing the flying distance and avoiding the weak coverage region, as desired for problem (P3).

C. Simultaneous Navigation and Radio Mapping

The dueling DDQN multi-step learning proposed in Algorithm 1 is a *model-free* Q learning algorithm, where all data used to update the estimation of the action-value function is obtained

Algorithm 1 Dueling DDQN Multi-Step Learning for Coverage-Aware UAV Navigation.

1: **Initialize:** the maximum number of episodes $\bar{N}_{\text{epi}}$, maximum number of steps per episode $\bar{N}_{\text{step}}$, initial exploration ϵ_0, decaying rate α, set $\epsilon \leftarrow \epsilon_0$
2: **Initialize:** reaching-destination reward R_{des}, outbound penalty P_{ob}, outage penalty weight μ
3: **Initialize:** the replay memory queue D with capacity C
4: **Initialize:** the dueling DQN network with coefficients $\boldsymbol{\theta}$, the target network with coefficients $\boldsymbol{\theta}^- = \boldsymbol{\theta}$
5: **for** $n_{\text{epi}} = 1, \cdots, \bar{N}_{\text{epi}}$ **do**
6: Initialize a sliding window queue W with capacity N_1
7: Randomly initialize the state $\mathbf{q}_I \in \mathcal{S}$, set the time step $n \leftarrow 0$ and $\mathbf{q}_0 \leftarrow \mathbf{q}_I$.
8: **repeat**
9: Choose action $\vec{\mathbf{v}}_n$ from $\hat{\mathcal{A}}$ based on the ϵ-greedy policy, i.e., $\vec{\mathbf{v}}_n = \vec{\mathbf{v}}^{(k^*)}$, where

$$k^* = \begin{cases} \text{randi}(K), & \text{with prob. } \epsilon, \\ \underset{k=1,\cdots,K}{\text{argmax}}\, \hat{Q}(\mathbf{q}_n, \vec{\mathbf{v}}^{(k)}; \boldsymbol{\theta}), & \text{with prob. } 1-\epsilon. \end{cases}$$

10: Execute action $\vec{\mathbf{v}}_n$ and observe the next state $\mathbf{q}_{n+1}$, measure the signals and obtain the empirical outage probability $\hat{P}_{\text{out}}(\mathbf{q}_{n+1})$ as in (30), and set the reward as $R_n = -1 - \mu \hat{P}_{\text{out}}(\mathbf{q}_{n+1})$
11: Store transition $(\mathbf{q}_n, \vec{\mathbf{v}}_n, R_n, \mathbf{q}_{n+1})$ in the sliding window queue W
12: If $n \geq N_1$, calculate the N_1-step accumulated return $R_{(n-N_1):n}$ based on (31) using the stored transitions in W, and store the N_1-step transition $(\mathbf{q}_{n-N_1}, \vec{\mathbf{v}}_{n-N_1}, R_{(n-N_1):n}, \mathbf{q}_n)$ in the replay memory D
13: Sample random minibatch of N_1-step transition $(\mathbf{q}_j, \vec{\mathbf{v}}_j, R_{j:j+N_1}, \mathbf{q}_{j+N_1})$ from D
14: Set

$$y_j = \begin{cases} R_{j:j+N_1} + R_{\text{des}}, & \text{if } \mathbf{q}_{j+N_1} = \mathbf{q}_F \\ R_{j:j+N_1} - P_{\text{ob}}, & \text{if } \mathbf{q}_{j+N_1} \notin \mathcal{S} \\ R_{j:j+N_1} + \gamma^{N_1} \hat{Q}(\mathbf{q}_{j+N_1}, \vec{\mathbf{v}}^*; \boldsymbol{\theta}^-), & \text{otherwise}, \end{cases}$$

where $\vec{\mathbf{v}}^* = \underset{\vec{\mathbf{v}}' \in \hat{\mathcal{A}}}{\arg\max}\, \hat{Q}(\mathbf{q}_{j+N_1}, \vec{\mathbf{v}}'; \boldsymbol{\theta})$.
15: Perform a gradient descent step on $\left(y_j - \hat{Q}(\mathbf{q}_j, \vec{\mathbf{v}}_j; \boldsymbol{\theta})\right)^2$ with respect to network parameters $\boldsymbol{\theta}$
16: Update $n \leftarrow n+1$, $\epsilon \leftarrow \epsilon\alpha$
17: **until** $\mathbf{q}_n = \mathbf{q}_F$ or $\mathbf{q}_n \notin \mathcal{S}$, or $n = \bar{N}_{\text{step}}$.
18: After every B episodes, set the target network parameters $\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$
19: **end for**

via the direct interaction of the UAV with the environment, as in step 10 of Algorithm 1. While model-free RL requires no prior knowledge about the environment nor intending to estimate it, it usually leads to slow learning process and requires a large number of agent-environment interactions, which is typically costly or even risky to obtain. For instance, for the coverage-aware UAV navigation problem considered in this paper, each step 10 in Algorithm 1 actually requires to fly the UAV and perform signal measurements at designated locations to get the

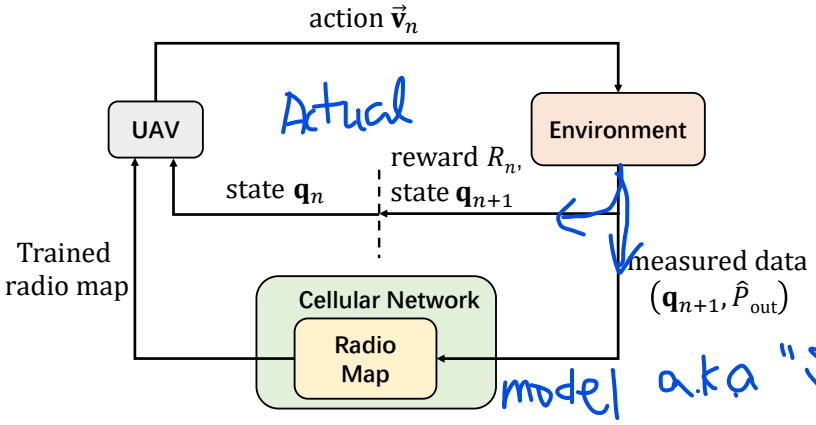


Fig. 4: SNARM for cellular-connected UAV.

empirical outage probability so as to obtain the reward value. Extensive UAV flying increases the safety risk due to e.g., mechanical fatigue. To overcome the above issues, in this subsection, we propose a novel SNARM framework for cellular-connected UAVs by utilizing the dueling DDQN multi-step learning algorithm with *model learning* [14].

For RL with model learning, each real experience obtained by the UAV-environment interaction can actually be used for at least two purposes: for *direct RL* as in Algorithm 1 and for *model learning* so as to gain useful information about the environment. A closer look at the MDP corresponding to problem (P3) would reveal that the essential information needed for coverage-aware navigation is the reward function $R(\mathbf{q})$, which depends on the outage probability $P_{\text{out}}(\mathbf{q})$ for any location $\mathbf{q}$ visited by the UAV. Therefore, model learning for (P3) corresponds to learning the *radio map* of the airspace where the UAVs are operating, which specifies the cellular outage probability $P_{\text{out}}(\mathbf{q})$, $\forall \mathbf{q} \in \mathcal{S}$. Thus, the actual measurement of the outage probability $\hat{P}_{\text{out}}(\mathbf{q})$ at location $\mathbf{q}$ can be simultaneously used for navigation, i.e., determining the UAV flying direction $\vec{\mathbf{v}}$ for the next time step, and also for radio mapping, i.e., helping to predict the outage probability for locations that the UAV has not visited yet. This leads to our proposed framework of SNARM.

A high-level overview of SNARM is illustrated in Fig. 4. Different from the standard RL setup, with SNARM for cellular-connected UAV, the cellular network (say, a selected macro-cell in the area of interest) maintains a radio map $\mathcal{M} = \{P_{\text{out}}(\mathbf{q}), \forall \mathbf{q} \in \mathcal{S}\}$ containing an estimation of the cellular outage probability for all locations in $\mathcal{S}$. Note that $\mathcal{M}$ might be highly inaccurate initially, but can be continuously improved as more real experience is accumulated. For any cellular-connected UAV flying in the considered airspace $\mathcal{S}$, it may either download the radio map $\mathcal{M}$ from the network off-line before the mission starts, or continuously receive the latest

update of $\mathcal{M}$ in real time. At each time step n as the UAV performs its mission, it observes its current location $\mathbf{q}_n$, together with the radio map $\mathcal{M}$, then determines its flying direction $\vec{\mathbf{v}}_n$ for the next step. After executing the action and reaching the new location $\mathbf{q}_{n+1}$, it measures the empirical outage probability $\hat{P}_{\text{out}}(\mathbf{q}_{n+1})$, and receives the corresponding reward R_n. The obtained empirical outage probability $\hat{P}_{\text{out}}(\mathbf{q}_{n+1})$ based on signal measurement can be used as the new data input to improve the radio map estimation. In the following, we explain the radio map update and UAV action selection of SNARM, respectively.

With any finite measurements $\{\mathbf{q}_n, \hat{P}_{\text{out}}(\mathbf{q}_n)\}$, predicting the outage probability $P_{\text{out}}(\mathbf{q}), \forall \mathbf{q} \in \mathcal{S}$ is essentially a *supervised learning* problem. In this paper, we use a feedforward fully-connected ANN with parameters $\boldsymbol{\xi}$ to represent the radio map, i.e., $\boldsymbol{\xi}$ is trained so that $P_{\text{out}}(\mathbf{q}) \approx \hat{P}_{\text{out}}(\mathbf{q}; \boldsymbol{\xi})$, $\forall \mathbf{q} \in \mathcal{S}$. Note that compared to the standard supervised learning problem, the radio mapping problem has two distinct characteristics. First, the data measurement $\{\mathbf{q}_n, \hat{P}(\mathbf{q}_n)\}$ only arrives incrementally as the UAV flies to new locations, instead of all made available at the beginning of the training. Second, as the UAV performs signal measurements as it flies, those data obtained at consecutive time steps are typically correlated, which is undesirable for supervised training. To tackle these two issues, similar to Algorithm 1, we use a database (replay memory) to store the measured data $\{\mathbf{q}_n, \hat{P}(\mathbf{q}_n)\}$, and at each training step, a minibatch is sampled at random from the database to update the network parameter $\boldsymbol{\xi}$.

On the other hand, the problem for UAV action selection with the assistance of a radio map $\mathcal{M}$ is known as *indirect RL*, for which various algorithms have been proposed [14]. Different from that for direct RL in Algorithm 1, with the radio map $\mathcal{M}$ in SNARM, the UAV is able to predict the estimated return for each trajectory it would take, without having to actually fly along it. As a result, we may generate as much simulated experience as we wish based on $\mathcal{M}$, which, together with the real experience, can be used to update the UAV policy based on any RL algorithm, such as Algorithm 1. However, as the radio map might be inaccurate at the initial stages, over-relying on the simulated experience may result in poor performance due to the model error. Therefore, we need to seamlessly integrate the learning with simulated experience based on the radio map and that with the real experience. A simple but effective architecture achieving this goal is known as *Dyna* [14], where for each RL update based on the real experience, we have several updates based on the simulated experience. The proposed SNARM based on Dyna dueling DDQN with multi-step learning is summarized in Algorithm 2.

Algorithm 2 Dyna dueling DDQN with multi-step learning for SNARM.

1: Download the radio map $\hat{P}_{\text{out}}(\mathbf{q};\boldsymbol{\xi})$ from the cellular network, and denote the existing database containing all measurements $\{\mathbf{q}, \hat{P}_{\text{out}}(\mathbf{q})\}$ as E
2: Steps 1–7 of Algorithm 1
3: Randomly initialize the simulated starting state $\tilde{\mathbf{q}}_I \in \mathcal{S}$, set the simulated time step $\tilde{n} \leftarrow 0$ and $\tilde{\mathbf{q}}_0 \leftarrow \tilde{\mathbf{q}}_I$
4: Steps 8–16 of Algorithm 1
5: Add the measured data $(\mathbf{q}_{n+1}, \hat{P}_{\text{out}}(\mathbf{q}_{n+1}))$ to E
6: Sample random minibatch from E and update the network parameter $\boldsymbol{\xi}$ for radio map
7: **for** $i = 1, \cdots, \tilde{N}$ **do**
8: Choose action for the simulated state $\tilde{\mathbf{q}}_{\tilde{n}}$ based on the ϵ-greedy policy similar as step 9 of Algorithm 1
9: Obtain the simulated next state $\tilde{\mathbf{q}}_{\tilde{n}+1}$, predict the outage probability $\hat{P}_{\text{out}}(\tilde{\mathbf{q}}_{\tilde{n}+1};\boldsymbol{\xi})$ based on the radio map $\mathcal{M}$ and the corresponding reward $\tilde{R}_{\tilde{n}} = -1 - \mu\hat{P}_{\text{out}}(\tilde{\mathbf{q}}_{\tilde{n}+1};\boldsymbol{\xi})$.
10: Perform similar operations as steps 11–15 of Algorithm 1 for the simulated experience
11: Update the simulated time step $\tilde{n} \leftarrow \tilde{n} + 1$
12: If $\tilde{\mathbf{q}}_{\tilde{n}} = \mathbf{q}_F$ or $\tilde{\mathbf{q}}_{\tilde{n}} \notin \mathcal{S}$ or $\tilde{n} = \bar{N}_{\text{step}}$, reset the simulated trajectory as in step 3
13: **end for**
14: Steps 17–19 of Algorithm 1.

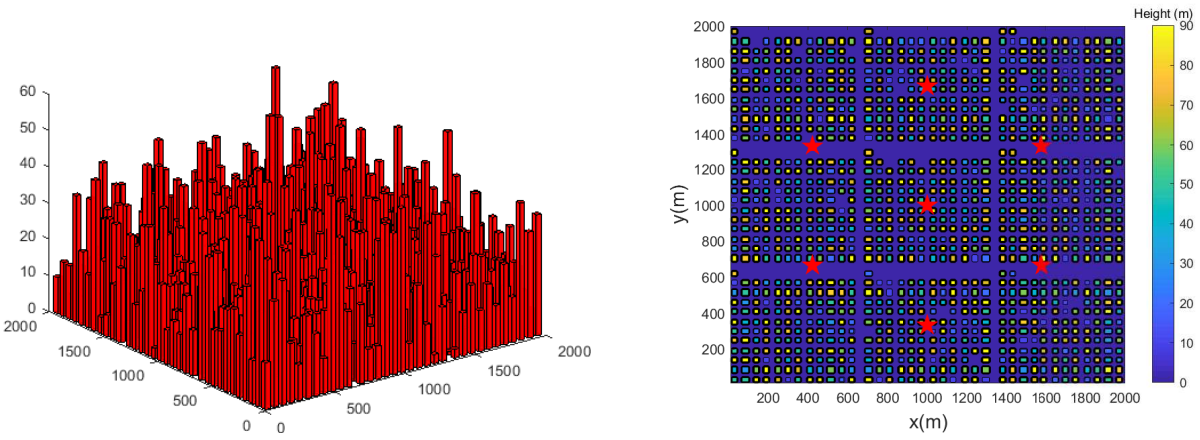
Obviously, Algorithm 2 expands Algorithm 1 by inserting the radio map learning operations and the double DQN update based on simulated UAV experience. Note that the simulated trajectory is initialized independently from the real trajectory (see steps 3 and 12 of Algorithm 2), and it is used $\tilde{N}$ times more frequently than the real trajectory, i.e., for each actual step the UAV takes in real experience, $\tilde{N}$ steps will be taken in the simulated trajectory, as in steps 7 to 13. Note that in general, $\tilde{N}$ should be increasing with the number of episodes, since as more real experience is accumulated, the constructed radio map becomes more accurate, which makes the predicted returns of the simulated trajectories more reliable. Regardless of simulated or real experience, the same action-value update based on dueling DDQN multi-step learning is performed. Besides, the radio map is updated in step 6 based on the actual measurement data, and used in step 9 to generate simulated UAV experience.

V. Numerical Results

In this section, numerical results are provided to evaluate the performance of the proposed algorithms. As shown in Fig. 5, we consider an urban area of size 2 km × 2 km with high-rise buildings, which corresponds to the most challenging environment for coverage-aware UAV navigation, since the LoS/NLoS links and the received signal strength may alter frequently as the

UAV flies (see Fig. 1). To accurately simulate the BS-UAV channels in the given environment, we first generate the building locations and heights based on one realization of the statistical model suggested by International Telecommunication Union (ITU) [40], which involves three parameters, namely, $\alpha_{\rm bd}$: the ratio of land area covered by buildings to the total land area; $\beta_{\rm bd}$: the mean number of buildings per unit area; and $\gamma_{\rm bd}$: a variable determining the building height distribution, which is modelled as Rayleigh distribution with mean value $\sigma_{\rm bd} > 0$. Note that such statistical building model has been widely used to obtain the LoS probability for aerial-ground links [11], which, however, only reflects the average characteristics over a large number of geographic areas of similar type. While for each local area with given building locations and heights, the presence/absence of LoS links with cellular BSs can be exactly determined by checking whether the line connecting the BS and UAV is blocked or not by any building. Fig. 5 shows the 3D and 2D views of one particular realization of the building locations and heights with $\alpha_{\rm bd} = 0.3$, $\beta_{\rm bd} = 300$ buildings/km^2, and $\sigma_{\rm bd} = 50$ m. For convenience, the building height is clipped to not exceed 90 m.

We assume that the considered area has 7 cellular BS sites with locations marked by red stars in Fig. 5, and the BS antenna height is 25 m [7]. With the standard sectorization technique, each BS site contains 3 sectors/cells. Thus, the total number of cells is $M = 21$. The BS antenna model follows the 3GPP specification [5], where an 8-element uniform linear array (ULA) is placed vertically. Each array element itself is directional, with half-power beamwidths along the vertical and horizontal dimensions both equal to 65°. Furthermore, pre-determined phase shifts are applied to the vertical antenna array so that its main lobe is electrically downtilted towards the ground by 10°. This leads to the directional antenna array with fixed 3D radiation pattern, as shown in Fig. 4 of [2]. To simulate the signal strength received by the UAV from each cell, we firstly determine whether there exists an LoS link between the UAV and each BS at the given UAV location based on the building realization, and then generate the BS-UAV path loss using the 3GPP model for urban Macro (UMa) given in Table B-2 of [7]. The small-scale fading coefficient is then added assuming Rayleigh fading for the NLoS case and Rician fading with 15 dB Rician factor for the LoS case. It is worth mentioning that for a given environment, the aforementioned LoS condition, antenna gain, path loss and small-scale fading with each BS are all dependent on the UAV location in a rather sophisticated manner. This makes them only suitable to generate the numerical values with given UAV location, instead of being directly used



(a) 3D view (b) Top view

Fig. 5: The building locations and heights in 3D and 2D views. The BS locations are marked by red stars, where each BS site contains 3 cells.

for UAV path planning with the conventional optimization technique as for solving (P1), even with accurate channel modelling and perfect global information. This thus justifies the use of RL techniques for UAV path design as pursued in this paper.

For both Algorithms 1 and 2, the dueling DQN adopted is a fully connected feedforward ANN consisting of 5 hidden layers. The numbers of neurons of the first 4 hidden layers are 512, 256, 128, and 128, respectively. The last hidden layer is the dueling layer with $K+1$ neurons, with one neuron corresponding to the estimation of the state-value and the other K neurons corresponding to the *action advantages* for the K actions, i.e., the difference between the action values and the state value of each state. The outputs of these $K+1$ neurons are then aggregated in the output layer to obtain the estimation of the K action values [16], as shown in Fig. 3. For SNARM in Algorithm 2, the ANN for radio mapping also has 5 hidden layers, with the corresponding numbers of neurons given by 512, 256, 128, 64, and 32, respectively. The activation for all hidden layers is by *ReLu* [37], and the ANN is trained with *Adam* optimizer to minimize the mean square error (MSE) loss, using Python with TensorFlow and Keras.[3] For ease of illustration, we assume that the UAV's flying altitude is fixed as $H = 100$ m, and the number of actions (or flying directions) at each time step is $K = 4$. All UAV missions have a common destination location $\mathbf{q}_F = [1400, 1600, 100]^T$ m, while their initial locations $\mathbf{q}_I$ are randomly generated. The transmit power of each cell is assumed to be $\bar{P}_m = 20$ dBm and the outage SIR threshold is $\gamma_{\text{th}} = 0$ dB. Other major simulation parameters are summarized in Table I.

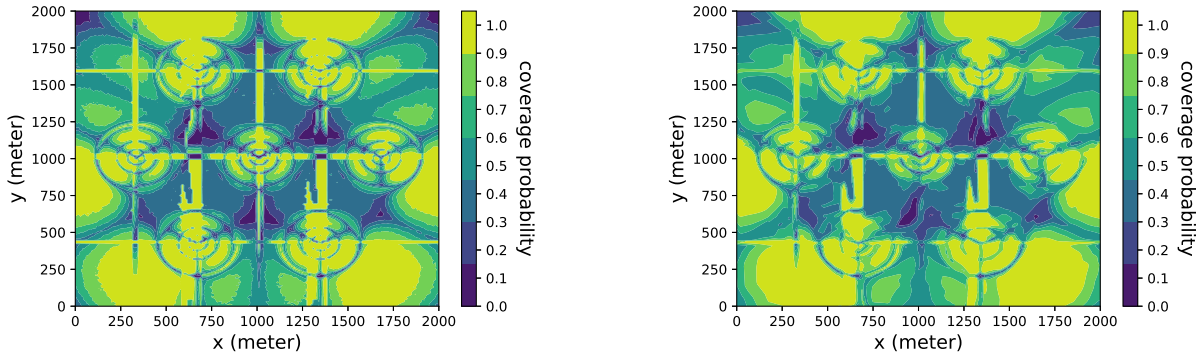
[3]https://keras.io/

TABLE I: Main simulation parameters.

Simulation parameter	**Value**
Maximum number of episodes $\bar{N}_{\text{epi}}$	5000
Initial exploration parameter ϵ_0	0.5
Exploration decaying rate α	0.998
Replay memory capacity C	100000
Steps for multi-step learning N_1	30
Update frequency for target network B	5
Maximum step per episode $\bar{N}_{\text{step}}$	200
Step displacement Δ_s	10 m
Reaching-destination reward R_{des}	200
Outbound penalty P_{ob}	10000
Outage penalty weight μ	40
Number of signal measurements per time step J	1000

Fig. 6(a) shows the true global coverage map for the considered area, i.e., the coverage (non-outage) probability for each location by the cellular network, where coverage probability is the complementary probability of the outage probability defined in (6). Note that such a global coverage map is numerically obtained based on the aforementioned 3D building and channel realizations via computer simulations, while it is not available for our proposed Algorithm 1 or Algorithm 2. It is observed from Fig. 6(a) that the coverage pattern in the sky is rather irregular due to the joint effects of the 3D BS antenna radiation pattern and building blockage. It is observed that around the center of the area, there are several weak coverage regions with coverage probability less than 30%. Obviously, effective coverage-aware UAV navigation should direct UAVs to avoid entering such weak coverage regions with the best effort.

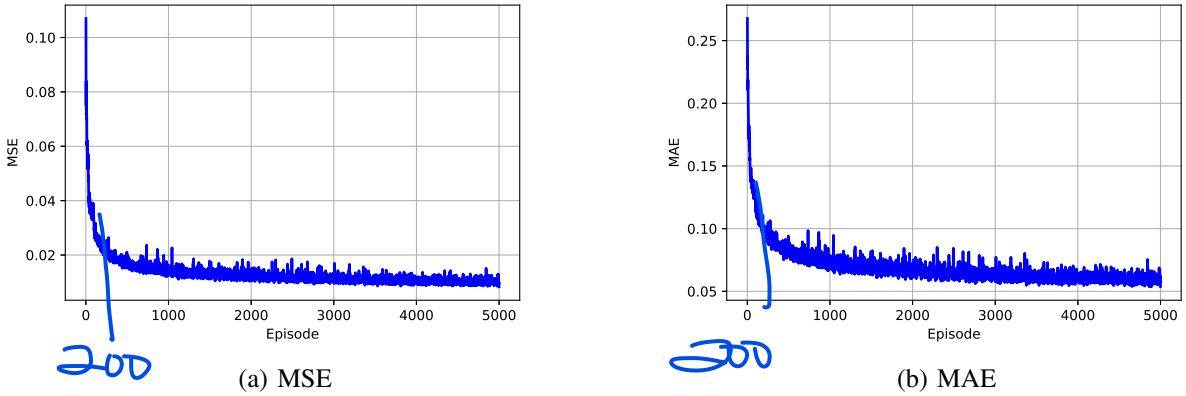
To validate the quality of radio map estimated by the SNARM framework proposed in Algorithm 2, Fig. 6(b) shows the finally learned coverage map by Algorithm 2. By comparing Fig. 6(a) and Fig. 6(b), it is observed that the learned coverage map is almost identical to the true map, with only slight differences. This thus validates the effectiveness of applying SNARM for the radio map estimation and coverage-aware path learning based on it. This is further corroborated by Fig.7, which shows the MSE and mean absolute error (MAE) of the learned radio map versus the episode number. Note that the MSE and MAE are calculated by comparing the predicted outage probabilities using the learned radio map versus their actual values in the true map for a set of randomly selected locations. It is observed that the quality of the learned radio map is rather poor initially, but it improves rather quickly with the episode



(a) True coverage map

(b) Learned coverage map

Fig. 6: The true versus learned coverage maps, where the coverage probability at each location is the complementary probability of the outage probability.



(a) MSE

(b) MAE

Fig. 7: The MSE and MAE of radio mapping versus episode number.

number as more signal measurements are accumulated. For example, with only 200 episodes, we can already achieve 87.1% of the MSE reduction that is achieved after 5000 episodes.

As for the performance of the proposed learning algorithms, Fig. 8 shows the moving average return per episode for the direct RL in Algorithm 1 and SNARM in Algorithm 2, where the moving window has the length of 200 episodes. For the SNARM algorithm, the number of steps $\tilde{N}$ based on simulated experience per actual step at episode n is set as $\tilde{N} = \min(\lfloor n/100 \rfloor, 10)$, so that $\tilde{N}$ increases with the episode number and converges to 10. Recall that the return of each episode for RL is the accumulated rewards for all time steps in the episode. It is observed that though experiencing certain fluctuation, as usually the case for RL algorithms, both proposed algorithms demonstrate an overall tendency of increasing average return. Furthermore, recall

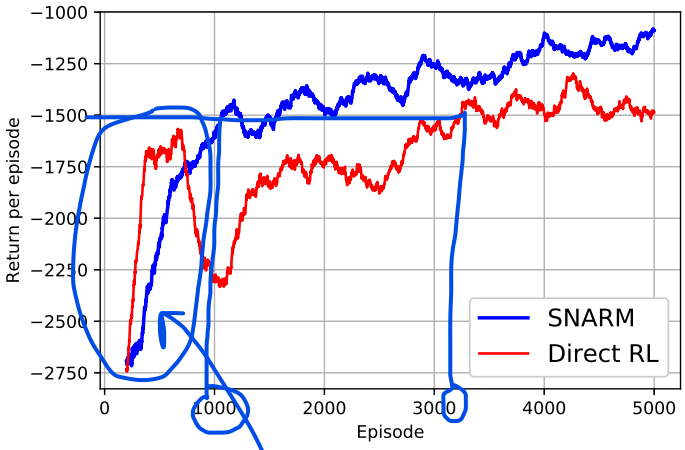
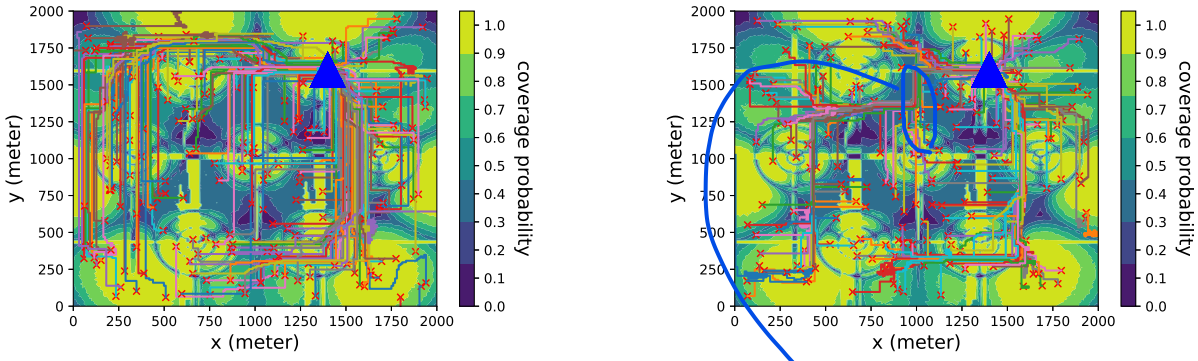


Fig. 8: Moving average return of direct RL versus SNARM over episode.

that the dueling DQNs are initialized to encourage shortest path flying for the early episodes when the UAV has completely no knowledge about the radio environment, which leads to an average return about -2700 initially. After training for 5000 episodes, the resulting return has increased to -1089 and -1486 for SNARM and direct RL, respectively. Another observation from the figure is that the direct RL has faster improvement over the SNARM algorithm at the earlier episodes. This is probably due to the insufficient accuracy of the learned radio map for SNARM, which may generate inefficient simulated experience. However, as the quality of the radio map improves, the SNARM algorithm achieves better performance than direct RL, thanks to its exploitation of the learned radio map for path planning. In practice, such a performance improvement translates to fewer agent-environment interactions, which is highly desirable due to the usually high cost of data acquisition from real experience. For example, to achieve an average return of -1500, Fig. 8 shows that about 3200 actual UAV flight episodes are required for direct RL in Algorithm 1, but this number is significantly reduced to about 1000 with SNARM in Algorithm 2.

Last, Fig. 9 shows the resulting UAV paths with the proposed algorithms for the last 200 episodes of learning, which correspond to 200 randomly generated initial locations, as denoted by red crosses in the figure. The common destination location $\mathbf{q}_F$ is labelled by a big blue triangle. Also shown in the figure is the true global coverage map. It is observed that both direct RL and SNARM algorithms are able to direct the UAVs to avoid the weak cellular coverage regions with the best effort, as evident from the generally detoured UAV paths as well as the



(a) Direct RL (b) SNARM

Fig. 9: The resulting UAV trajectories with direct RL in Algorithm 1 and SNARM in Algorithm 2. All episodes share a common destination location labelled by the big blue triangle, and their initial locations are randomly generated, labelled by red crosses.

much sparser paths in areas with low coverage probabilities. For example, as can be seen from Fig. 9(b), the SNARM algorithm is able to discover and follow the narrow "radio bridge" located at the x axis around 1000 m and the y axis from about 1000 m to 1700 m. This shows the peculiar effectiveness of SNARM for coverage-aware UAV path learning.

VI. Conclusions

This paper studies coverage-aware navigation for cellular-connected UAVs. To overcome the practical limitations of the conventional optimization-based path design approaches, we propose DRL-based algorithms, which only require UAV signal measurements as the input. By utilizing the state-of-the-art dueling DDQN with multi-step learning, a direct RL-based algorithm is first proposed, followed by the more advanced SNARM framework to enable radio mapping and reduce the real UAV flights for data acquisition. Numerical results are provided to show the effectiveness of the proposed algorithms for coverage-aware UAV navigation, and the superior performance of SNARM over direct RL based navigation.

References

[1] Y. Zeng and X. Xu, "Path design for cellular-connected UAV with reinforcement learning," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, 2019.

[2] Y. Zeng, Q. Wu, and R. Zhang, "Accessing from the sky: a tutorial on UAV communications for 5G and beyond," *Proc. of the IEEE*, vol. 107, no. 12, pp. 2327–2375, Dec. 2019.

[3] 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.

[4] Y. Zeng, J. Lyu, and R. Zhang, "Cellular-connected UAV: potentials, challenges and promising technologies," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 120–127, Feb. 2019.

[5] 3GPP TR 36.873: "Study on 3D channel model for LTE", V12.7.0, Dec., 2017.

[6] Qualcomm, "LTE unmanned aircraft systems," trial Report, available online at https://www.qualcomm.com/documents/lte-unmanned-aircraft-systems-trial-report, May, 2017.

[7] 3GPP TR 36.777: "Technical specification group radio access network: study on enhanced LTE support for aerial vehicles", V15.0.0, Dec., 2017.

[8] S. Zhang, Y. Zeng, and R. Zhang, "Cellular-enabled UAV communication: a connectivity-constrained trajectory optimization perspective," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2580–2604, Mar. 2019.

[9] S. Zhang and R. Zhang, "Trajectory design for cellular-connected UAV under outage duration constraint," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May, 2019.

[10] E. Bulut and I. Guvenc, "Trajectory optimization for cellular-connected UAVs with disconnectivity constraint," in *Proc. IEEE International Conference on Communications (ICC)*, May 2018.

[11] 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.

[12] M. M. Azari, F. Rosas, K.-C. Chen, and S. Pollin, "Ultra reliable UAV communication using altitude and cooperation diversity," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 330–344, Jan. 2018.

[13] 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.

[14] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. MIT Press, Cambridge, MA, 2018.

[15] V. Mnih, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[16] Z. Wang, *et al.*, "Dueling network architectures for deep reinforcement learning," in *Proc. of the 33rd Int. Conf. Machine Learning*, Jun. 2016, pp. 1995–2003.

[17] J. Chen, U. Yatnalli, and D. Gesbert, "Learning radio maps for UAV-aided wireless networks: a segmented regression approach," in *Proc. IEEE International Conference on Communications (ICC)*, May 2017.

[18] S. Bi, J. Lyu, Z. Ding, and R. Zhang, "Engineering radio map for wireless resource management," *IEEE Wireless Commun.*, vol. 26, no. 2, pp. 133–141, Apr. 2019.

[19] H. Bayerlein, P. Kerret, and D. Gesbert, "Trajectory optimization for autonomous flying base station via reinforcement learning," in *Proc. IEEE SPAWC*, Jun. 2018.

[20] L. Xiao, X. Lu, D. Xu, Y. Tang, L. Wang, and W. Zhuang, "UAV relay in VANETs against smart jamming with reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4087–4097, May 2018.

[21] J. Cui, Y. Liu, and A. Nallanathan, "Multi-agent reinforcement learning based resource allocation for UAV networks," *IEEE Trans. Wireless Commun.*, Aug. 2019, early Access.

[22] B. Khamidehi and E. S. Sousa, "Reinforcement learning-based trajectory design for the aerial base stations," available online at https://arxiv.org/pdf/1906.09550.pdf.

[23] J. Hu, H. Zhang, and L. Song, "Reinforcement learning for decentralized trajectory design in cellular UAV networks with sense-and-send protocol," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6177–6189, Aug. 2019.

[24] N. C. Luong, *et al.*, "Applications of deep reinforcement learning in communications and networking: a survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3133–3174, Fourthquarter 2019.

[25] Y. Yu, T. Wang, and S. C. Liew, "Deep-reinforcement learning multiple access for heterogeneous wireless networks," *IEEE J. Selected Areas Commun.*, vol. 37, no. 6, pp. 1277–1290, Jun. 2019.

[26] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Selected Areas Commun.*, vol. 37, pp. 2239–2250, 2019.

[27] L. Zhang, J. Tan, Y.-C. Liang, G. Feng, and D. Niyato, "Deep reinforcement learning based modulation and coding scheme selection in cognitive heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3281–3294, Jun. 2019.

[28] L. Huang, S. Bi, and Y. J. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Computing*, Jul. 2019, early access.

[29] J. Qiu, J. Lyu, and L. Fu, "Placement optimization of aerial base stations with deep reinforcement learning," available online at https://arxiv.org/abs/1911.08111.

[30] U. Challita, W. Saad, and C. Bettstetter, "Interference management for cellular-connected UAVs: a deep reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2125–2140, Apr. 2019.

[31] S. Zhang and R. Zhang, "Radio map based path planning for cellular-connected UAV," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Dec. 2019.

[32] J. Chen and D. Gesbert, "Local map-assisted positioning for flying wireless relays," arXiv:1801.03595, 2018.

[33] O. Esrafilian, R. Gangula, and D. Gesbert, "3D-map assisted uav trajectory design under cellular connectivity constraints," available online at https://arxiv.org/abs/1911.01124.

[34] C. You and R. Zhang, "Hybrid offline-online design for UAV-enabled data harvesting in probabilistic LoS channel," available online at https://arxiv.org/abs/1907.06181.

[35] X. Xu and Y. Zeng, "Cellular-connected UAV: performance analysis with 3D antenna modelling," in *Proc. IEEE International Conference on Communications (ICC) Workshops*, May 2019.

[36] M. Hessel, *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. of AAAI*, 2018.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[38] H. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. of AAAI*, 2016, pp. 2094–2100.

[39] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Proc. of ICLR*, 2015.

[40] ITU-R, Rec. P.1410-5 "Propagation data and prediction methods required for the design of terrestrial broadband radio access systems operating in a frequency range from 3 to 60 GHz", Radiowave propagation, Feb. 2012.

↳ 환경 (building) 생성