Deep Q-Network Based Dynamic Movement Strategy in a UAV-Assisted Network

Xukai Zhong, Yiming Huo, Member, IEEE, Xiaodai Dong, Senior Member, IEEE, and Zhonghua Liang, Senior Member, IEEE

Abstract—Unmanned aerial vehicle (UAV)-assisted communications is a promising solution to improve the performance of future wireless networks, where UAVs are deployed as base stations for enhancing the quality of service (QoS) provided to ground users when traditional terrestrial base stations are unavailable or not sufficient. An effective framework is proposed in this paper to manage the dynamic movement of multiple unmanned aerial vehicles (UAVs) in response to ground user mobility, with the objective to maximize the sum data rate of the ground users. First, we discuss the relationship between the air-to-ground (A2G) path loss (PL) and the location of UAVs. Then a deep Q-network (DQN) based method is proposed to adjust the locations of UAVs to maximize the sum data rate of the user equipment (UE). Finally, simulation results show that the proposed method is capable of adjusting UAV locations in a real-time condition to improve the QoS of the entire network.

Index Terms—Unmanned aerial vehicle (UAV), UAV-assisted network, reinforcement learning, user equipment (UE), quality of service (QoS).

I. INTRODUCTION

The unprecedented demand for high-quality wireless communications has fueled the evolution of wireless technologies and communications networks. The unmanned aerial vehicle (UAV)-assisted network where UAVs are deployed and function as aerial base stations to assist the terrestrial base stations is an effective complementary solution to emergency wireless service recovery after natural disasters or infrastructure damage [1]. Also, in Internet of Things (IoT) networks, UAVs can be used as aerial base stations to collect data from ground devices, in which building a complete cellular infrastructure is not affordable [2]. The authors in [3] proposed a multi-layer UAV network model for UAV-enabled 5G and beyond applications. Despite advantages such as flexibility, mobility, cost and time efficiency in UAV-assisted networks, one key design challenge is to determine the move strategy for UAVs. Since in realistic situations, the environment where UAVs are deployed is highly dynamic, it is critical for UAVs to adjust its locations regularly to cope with varying conditions. Furthermore, utilizing machine learning techniques for the UAV communication recently has seen unprecedented growing popularity [4].

A. Related Work

Regarding the existing research related to the UAVs deployment, the authors in [5] modeled the static UAV deployment problem based on circle packing theory and studied the relationship between the number of deployed UAVs and the coverage time. Moreover, [6] proposed an efficient air-to-ground (A2G) channel model with probabilistic path loss (PL) and discussed a method to derive the optimal altitudes of UAVs based on the A2G channel model. The work in [7] discussed an optimization problem in the UAV-assisted network which aims to maximize the number of covered UEs while minimizing the interference between UAVs. Moreover, the movement control for the UAVs serving wireless communications has been studied by reference [8].

Nowadays, machine learning techniques have gained popularity in solving UAV deployment problem and the reinforcement learning algorithm has proved to be an efficient solution of solving dynamic problem such as UAVs movement management in the UAV-assisted network [9]–[11]. In particular, a machine learning framework based on Gaussian mixture model (GMM) and a weighted expectation maximization (WEM) algorithm to predict the locations of UAVs with the total power consumption minimized was proposed in [9]. Furthermore, authors in [10] studied a Q-learning based algorithm to find the optimal trajectory to maximize the sum rates of fixed location ground users for a single UAV base station (UAV-BS), assuming a random initial location. Reference [11] proposed a Q-learning based movement design for multiple UAV-BSs. In addition, authors in [12] also demonstrated a reinforcement learning leveraged handover mechanism for cellular-connected drone system.

B. Our Contribution

Despite aforementioned works, there has been few study on real-time movement strategy for UAVs to cope with the ground UEs’ mobility in a UAV-assisted network, which is very critical in practical application scenarios. In this research, we investigate a real-time dynamic UAV movement strategy design on a deep learning framework called deep Q-network (DQN) [13] to maximize the sum data rate. Unlike the existing literature about UAV trajectory planning in which the ground users are assumed geographically fixed, our contribution formulates the design problem of the UAVs’ movement strategy to find the optimal locations of UAVs in every single time instant, in response to the ground users’ random movement.
II. SYSTEM MODEL

A. System Description

Fig. 1 shows the framework of UAV-assisted wireless communications system model where UAVs serve as aerial base stations and provide hot spot wireless communications to the ground UEs. Also, the traditional terrestrial infrastructures are capable of serving the UEs which are not covered by UAV-BSs. Let \( \mathcal{P} \) be the set of all the UEs which are labelled as \( i = 1, 2, ..., |\mathcal{P}| \). \( \mathcal{Q} \) denotes the set of available UAV-BSs labelled as \( j = 1, 2, ..., |\mathcal{Q}| \) and \( \mathcal{O} \) denotes the set of ground base stations (GBSs) labelled as \( k = 1, 2, ..., |\mathcal{O}| \). In our system, we assume that the UEs are assigned to the closest base station to receive wireless communication service and all the UAV-BSs cells are deployed at the same altitude \( H \). Ground users are assumed to move from time to time and the location of the \( i \)-th UE at time \( t \) can be expressed as \( m_i(t) = [x_i(t), y_i(t)] \), \( t \in T \) where \( T \) is the time window considered. Similarly, the locations of UAV-BS \( j \) can be written as \( n_j(t) = [\tilde{x}_j(t), \tilde{y}_j(t)] \). Also, \( u_k = [\tilde{x}_k, \tilde{y}_k] \) denotes the location of the \( k \)-th GBS, which is a known parameter in the study.

B. Signal Model

The A2G channel model proposed in [6] considers the line-of-sight (LoS) communication occurring with a certain probability. At each time instant, the probability of having LoS and non line-of-sight (NLoS) communication between the UAV \( j \) and the user \( i \) are formulated in [6]

\[
\begin{align*}
P_{\text{LoS}} &= 1 + a \exp(-b(\frac{130}{\pi} \tan^{-1}(\frac{H}{r_{ij}}) - a)), \\
P_{\text{NLoS}} &= 1 - P_{\text{LoS}},
\end{align*}
\]

(1)

where \( a \) and \( b \) are environment dependent variables and \( r_{ij} = \sqrt{(x_j - \tilde{x}_i)^2 + (y_j - \tilde{y}_i)^2} \) is the horizontal Euclidean distance between the \( i \)-th user and \( j \)-th UAV. Then the path loss (PL) for LoS and NLoS can be written as

\[
\begin{align*}
L_{\text{LoS}} &= 20 \log_{10}(\frac{4\pi f_c d_{ij}}{c}) + \eta_{\text{LoS}}, \\
L_{\text{NLoS}} &= 20 \log_{10}(\frac{4\pi f_c d_{ij}}{c}) + \eta_{\text{NLoS}},
\end{align*}
\]

(2)

where \( f_c \) is the carrier frequency, \( c \) is the speed of light and \( d_{ij} \) denotes the distance between the UE and UAV-BS given by \( d_{ij} = \sqrt{H_j^2 + r_{ij}^2} \). Moreover, \( \eta_{\text{LoS}} \) and \( \eta_{\text{NLoS}} \) are the environment dependent average additional PL for LoS and NLoS conditions, respectively. According to (1) and (2), the PL can be written as

\[
L_{ij} = L_{\text{LoS}} \times P_{\text{LoS}} + L_{\text{NLoS}} \times P_{\text{NLoS}}
\]

\[
= \frac{A}{1 + a \exp(-b(\arctan(\frac{H}{r_{ij}}) - a))} + 20 \log(H^2 + r_{ij}^2) + B,
\]

(3)

where \( A = \eta_{\text{LoS}} - \eta_{\text{NLoS}} \) and \( B = 20 \log(\frac{4\pi L}{\eta_{\text{LoS}}}) + \eta_{\text{NLoS}} \).

The path loss for UEs which are associated with the GBSs at distance \( r_{ik} \) can be modeled by \( L_{ik} = \eta_{\text{LoS}}^{\bar{i}} \) where \( \eta \) is the additional PL over the free space PL and \( \alpha \) is the PL exponent.

Moreover, the signal-to-interference-plus-noise ratio (SINR) experienced at a UE at a distance \( r_{ij} \) from its associated UAV-BS \( j \) can be expressed as

\[
\text{SINR}_{ij} = \frac{P_j h_{ij} L_{ij}^{-1}}{\sigma^2 + \sum_{k \in \mathcal{O}} P_k h_{ik} L_{ik}^{-1}}.
\]

(4)

where

\[
I_{ij} = P_j h_{ij} P_{\text{LoS}}^{-1} \text{ and } I_{ik} = P_k h_{ik} P_{\text{LoS}}^{-1},
\]

(5)

represents interference from other UAV-BSs/GBSs, \( P_j \) represents the transmit power of the \( j \)-th base station, \( h_{ij} \) is the small fading power assumed to be an independent number following the exponential distribution and \( \sigma^2 \) is the variance of the additive white Gaussian noise component. For the UEs served by a GBS, their SINRs can be expressed in a similar manner. According to the Shannon Capacity Theorem, the data rate \( C_i \) of the \( i \)-th UE can be expressed as \( C_i = B \log_2(1 + \text{SINR}_{ij}) \) where \( B \) is the bandwidth of the channel.

III. FUNDAMENTAL OF REINFORCEMENT LEARNING

Reinforcement learning generally proceeds in a cycle of interactions between an agent and its environment. At time \( t \), the agent observes a state \( s_t \in S \), and performs an action \( a_t \in A \) and subsequently receives a reward \( r_t \in R \). The time index is then incremented and the environment propagates the agent to a new state \( S_{t+1} \), from where the cycle restarts. Therefore, the whole process is a Markov Decision Process (MDP) [14].

The task of the reinforcement learning is to train an agent interacting with the environment to provide the feedback to each of its actions. The agent arrives at different states by performing actions that lead to a reward so that we could reinforce the agents to learn to choose the best actions based on the reward. Therefore, the only objective of the agent is to maximize its total reward across an episode. The way the agent chooses its actions is known as policy.

A. Q-Learning

Q-learning specifically allows an agent to learn to act optimally in a given environment. The goal for the agent is to learn a behavior rule that maximizes the reward it receives. Q-learning is an off-policy reinforcement learning algorithm which finds the best action for a given state. It is considered off-policy because the Q-learning function learns from actions...
that are outside the current policy. More specifically, Q-learning learns a policy that maximizes the total reward.

- Q-Value: The Q-Value \( Q(s, a) \) represents the total rewards of agents being at state \( s \) and performing action \( a \), the Q-Value for each state and action can be found in the Q-Table. It can be computed by:

\[
Q(s, a) = r(s, a) + \gamma \max_a Q(s', a)
\]

(6)

where the above equation states that the Q-Value which is derived from the agent being at state \( s \) and taking action \( a \) equals to the immediate reward \( r(s, a) \) plus the highest possible Q-Value of the next state \( s' \) times a discount factor \( \gamma \) which represents the contribution of future rewards. To be more specific, the Q-Value is the sum of the instantaneous reward at the current time step and an observation of the next time step.

- Q-Table: Q-Table is a look up table which states the Q-Value that represents the future values of actions for each states, and is updated regularly.

To begin with, the Q-Table is initialized with all zeros. Then the agent chooses an action based on epsilon greedy strategy \( \alpha \) that 90% the agent chooses the action with highest Q-Value while 10% the agent chooses a random action. Based on the action the agent chooses, the reward of performing the action is observed. Then the updated Q-Value is calculated by the old Q-Value plus the learning rate times the sum of the instantaneous reward plus the expected future value. The expected future value equals the difference between the old Q-Value and maximum possible Q-Value for the next time step. The formula is shown as:

\[
Q_{new}(s, a) = Q_{old}(s, a) + \alpha (r(s, a) + \gamma \max_a Q(s', a) - Q_{old}(s, a))
\]

(7)

B. Deep Q-Network

The Q-Learning is a powerful algorithm to create a look up table for the agent so that the agent is capable for making rational action in each state. However, the drawback of Q-Learning is when there are too many states in the environment, it requires a large amount of memory since we need a long Q-Table. Therefore, the neural network is a powerful tool that can be utilized to compute Q-value.

In deep Q-Network, the next action is determined by the maximum output of the neural network. Referring to equation (7), if we make the loss function \( \text{Loss} = (r + \gamma \max_a Q(s', a; \Theta) - Q(s, a; \Theta))^2 \) where \( \Theta \) represents the parameters of the Q-Network, it becomes a simple regression problem.

However, in this loss function, \( Q(s, a; \Theta) \) plays the role of a desired target in a regression problem which needs to be stationary in order to converge the network. Therefore a separate network is used to calculate the target. This target network has the same architecture as the network to predict Q-Value but with frozen parameters. The parameters of the predicted network are copied to target network in every \( C \) iterations and \( C \) is a predetermined value.

Also, another important factor in Deep Q-Network is experience replay. It stores a fixed size of samples from training data into a memory tuple. In each training step, a mini-batch of samples are randomly selected from the memory to train the Q-Network. Experience replay breaks up the correlation in the training data by sampling batch of experiences randomly from a large memory pool which also helps the network to converge.

IV. UAV-BS MOVEMENT STRATEGY

The dynamic UAV-BS movement strategy problem can be treated as a design of determining the positions of the UAV-BSs at each time instant. The objective is to find the optimal positions for all UAV-BSs in each time-slot, to maximize the sum data rates of users. \( \gamma_{ij/k}(t) \) is a binary variable indicating whether the user \( i \) is associated with UAV-BS \( j \) or GBS \( k \) at time instant \( t \), with 1 for service and 0 for no association. Thus, the optimization problem at each time instant \( t \) can be formulated as:

\[
\begin{align*}
&\text{maximize} & \sum_{i=1}^{[P]} \sum_{j \in Q} C_i(t), \\
&\text{s.t.} & C_1 : & ||n_{j}(t) - \gamma_{ij}(t)m_{ij}(t)|| \leq ||n_{j}(t) - m_{ij}(t)|| \\
& & & + M ||1 - \gamma_{ij}(t)||, \forall j \in Q, \forall j \in \{Q, Q \setminus j\} \\
& C_2 : & ||u_{ik}(t) - \gamma_{ik}(t)m_{ik}(t)|| \leq ||u_{ik}(t) - m_{ik}(t)|| \\
& & & + M ||1 - \gamma_{ik}(t)||, \forall k \in O, \forall k \in \{Q, Q \setminus k\} \\
& C_3 : & \sum_{j} \gamma_{ij}(t) + \sum_{k} \gamma_{ik}(t) = 1, \forall i, j, k.
\end{align*}
\]

Constraints \( C_1 \) and \( C_2 \) in (6) guarantee all the UEs are associated with the nearest UAV-BSs/GBSs where, \( M \) is a large number to ensure the constraints hold in any UE association conditions. Then \( C_3 \) guarantees all the UEs are associated with only a single base station. Therefore, the objective of the optimization problem is to find the optimal positions of UAV-BSs in each instant over time duration \( T \) so that the sum data rates of the users can be maximized. Although the UEs’ movement is random, the UEs’ distribution follows a certain principle. As a consequence, the environment is considered as partly random and partly stationary, which makes the whole process follow the MDP.

V. DEEP Q-NETWORK BASED UAV-BSS MOVEMENT DESIGN

In this section, given the real-time locations of a set of UEs, we present a reinforcement learning based UAV-BS movement strategy to obtain the optimal real-time locations of UAV-BSs. Before discussing the movement of UAV-BSs, the mobility model of UEs needs to be discussed first. The random walk model [15] is chosen as the UE mobility model in this paper, but other models can be easily included. The moving direction of UEs are uniformly distributed among left, right, forward, backward and staying still. Moreover, the initial positions of the ground users are assumed to be fixed. At each instant \( t \in T \) when ground users move, all UAV-BSs take action in response to the movement of the ground users.
Algorithm 1 Deep Q-Network Based UAV-BS Movement Strategy

Required: Initial Position of UAV-BSs, $m_i(0)$ and UEs $n_j(0)$

1: Initialize replay memory $D$ with capacity $N$, mini-batch size $B$, initialize action-value network $Q$ with weight $\Theta_j \in Q$, target network $Q$ with weight $\tilde{\Theta}_j \in Q$ with random weights.
2: for each episode do
3:   Reset UAV-BSs to the initial positions
4:   for each time step $t$ do
5:     for each UAV-BS agent $j$ do
6:       Observe $s_t^{j}$
7:       Choose the action $a_t^{j}$ which maximizes the $Q(s_t^{j}, a_t^{j}; \tilde{\Theta}_j)$
8:     end for
9:   All agents take actions, observe rewards $r_t^{j}$, update state $s_t^{j} \rightarrow s_{t+1}^{j}$
10: for each UAV-BS agent $j$ do
11:   Observe $s_{t+1}^{j}$
12:   Store $(s_t^{j}, a_t^{j}, r_t^{j}, s_{t+1}^{j})$ into replay memory $D_j$
13: Uniformly sample mini batch from replay memory $D_j$
14: Perform a gradient descent on $\text{Loss} = (r_t^{j} + \gamma \max_{a'}Q(s_{t+1}^{j}, a'; \tilde{\Theta}_j) - Q(s_t^{j}, a_t^{j}; \tilde{\Theta}_j))^2$ with respect to network parameters $\tilde{\Theta}_j$.
15: Update $\tilde{\Theta}_j = \Theta_j$ every $C$ time steps
16: end for
17: end for
18: end for

B. Environment

The Deep Q-Network addresses constraints by responding to the actions taken by the agents from the environment. In our problem formulation, the UEs are assigned to the nearest base stations and one UE can only be assigned to one base station. Therefore, the feedback from the environment has to follow these constraints to decide the instantaneous rewards.

C. Action Space

At each time step, all the UAV-BSs take an action $a_t \in A$ which includes choosing a direction for UAV-BSs to move according to the current state $s_t$, based on the decision from Q-Network $Q$. In our model, we assume that all UAV-BSs move in the same speed in any time step, therefore the moving distance for any UAV-BS from any time instant $t$ to $t+1$ is assumed to be the same. More specifically, since we assume that all the UAV-BSs are at the same altitude $H$, there are 5 different actions in $A$: $(1,0)$ means the UAV-BS will turn right, $(-1,0)$ means the UAV-BS will turn left, $(0,1)$ means the UAV-BS will move forward, $(0,-1)$ means the UAV-BS will move backward and $(0,0)$ means the UAV-BS will stay still. All UAV-BSs take actions once after another in a sequential manner.

D. Reward Design

After performing an action, the UAV-BS has a different location so the UEs need to change the association based on problem (8). Therefore, the new association comes with a new instantaneous sum data rates of the ground UEs. The principle of designing the reward function is to improve the UEs’ instantaneous data rates, which enables the agent to receive a positive reward. When the action results in a reduction of the sum data rates of the UEs, the UAV-BS receives a negative reward. Thus, the reward function can be expressed as

$$r_t = \begin{cases} 
1, & \text{if sum rates increase} \\
-0.2, & \text{if sum rates remain the same}, \\
-1, & \text{if sum rates decrease}
\end{cases}$$

where the ratio of the positive reward and negative reward is 1 in order to avoid any bias. Also, the reason to design a reward for an unchanged sum data rate is that in a practical situation, moving an UAV consumes the energy and resource, if the movement does not contribute to the objective, a negative reward is granted but its absolute value is much less than the case of a decreased sum data rate.

E. Training Procedure

The training procedure requires a learning rate $\alpha$ and a discount factor $\gamma$. The learning procedure is divided into several episodes, and the positions of UAV-BSs will be reset to the initial values at the beginning of each episode. We leverage a DQN with experience replay to train the agents $[13]$. In each episode, each agent takes actions based on the Q-Values which are outputted by the neural network and a reward is generated in each step. Therefore, the parameters of the neural network can be updated so after going through all the training episodes
the neural network is capable of rational decisions for the UAV-BSs for each step. To be more specific, each agent $j$ has a DQN $\hat{Q}$ that takes an input of the observation of the current state $s_t^j$ and generate the output of the value functions corresponding to all the actions. At each training step $t$, each agent chooses the action $a_t^j$ which leads to the maximum estimated Q value. Based on the action taken by the agent, the transition tuple $(s_t^j, a_t^j, r_t^j, s_{t+1}^j)$ is collected and stored into the replay memory $D$ with a size of $N$. Then, in each episode, a predetermined size of the mini-batch experiences $E$ are uniformly sampled to update $\Theta$ using gradient descent method to minimize the loss function

$$Loss = \sum_{E} (r_t^j + \gamma \max_{a'} \hat{Q}(s_{t+1}^j, a'; \tilde{\Theta}_j) - \hat{Q}(s_t^j, a_t^j; \tilde{\Theta}_j))^2$$

where $\tilde{\Theta}_j$ is the parameter set of a target network $\hat{Q}$ which is replaced by the parameter set $\Theta_j$ of training Q-Network $Q$ every $C$ time steps. The experience replay can improve the training efficiency by breaking the correlation between samples so as to stabilize the training.

VI. NUMERICAL RESULTS

In our simulation, we consider UAV-assisted model in a 5000 m $\times$ 5000 m area and uniformly divide the entire area into 4 sections, i.e., Section 1 : $0 < x \leq 2500, 0 < y \leq 2500$, Section 2 : $2500 < x \leq 5000, 0 < y \leq 2500$, Section 3 : $0 < x \leq 2500, 2500 < y \leq 5000$, Section 4 : $2500 < x \leq 5000, 2500 < y \leq 5000$. We assume that initially all of the UEs are distributed in the whole area, and then in the middle of the time duration, the majority (90%) of the UEs converge to Section 1. At the end of the time duration, all the UEs go back to the uniformly distributed in the whole area. The UEs follow random walk mobility model inside the section area. There is one GBS available located at $u_0 = [2500, 2500]$. Further, we consider a period $T$ with 500 time instant and 50000 training episodes. Moreover, referring to [7], the environment parameters are set up as follows: $f_c = 2$ GHz, $P_{L_{\text{max}}} = 103$ dB, $(\alpha, b, \eta_{\text{LoS}}, \eta_{\text{NLoS}})$ is configured to be $(9.61, 0.43, 0.1, 20)$ corresponding to the urban environment. The transmit powers of UAV-BSs and GBS are set to be 37 dBm and 40 dBm, respectively. Also, the Deep-Q Network parameter set $(\alpha, \beta, N, B, C)$ is configured to be $(0.01, 0.9, 2000, 50, 200)$ and the structure of the network is configured to be 2 input neurons in the input layer, 10 neurons in the hidden layer and 5 neurons in the output layer. Also, the movement step size for UAV-BS is configured to be 1 meter. Fig. 3 shows the UEs distribution and their association in one time instant. The UEs and base stations with same color represent the association and all the UEs are associated with the closest base stations.

![Fig. 3. Snapshot of 500 UEs and association with 4 UAV-BSs in a 5000 m $\times$ 5000 m area.](image)

**TABLE I**

| Algorithm                  | Processing Time (ms) |
|---------------------------|----------------------|
| Deep Q-Network            | 210                  |
| Exhaustive Search         | 4117                 |
| K-Means                   | 387                  |
| Fixed                     | 0                    |

![Fig. 4. Sum data rate versus the number of training episodes.](image)

![Fig. 5. Sum data rate comparison of different methods.](image)

Fig. 4 further plots the sum data rates against the number of training episodes. It can be observed that the UAV-BSs are capable of carrying out their actions via iterative learning from their past experience to improve the performance.

Fig. 5 shows the comparison of the sum data rates in all the time instants with different algorithms. It can be observed
that the overall performance in 500 time instant of Deep Q-Network outperforms the fixed locations or K-Means deployment strategy and closely follows the performance of the exhaustive search. However, considering the computation cost results in Table I which is obtained using Intel Core i5- 4430 Processor to run the algorithm 10 times and take the average processing time. Exhaustive search as expected achieves the highest performance but the computation complexity can be too high for real-time processing. The Deep Q-Network performs close to the exhaustive search but with significantly less processing resource and time, which is particularly critical for low-latency communications and mission execution involving UAVs.

VII. CONCLUSION

This paper has proposed and evaluated a dynamic UAV-BS deployment strategy for optimizing the real-time performance of wireless communication services when all the UEs are moving. A Deep Q-Network based algorithm has been proposed to maximize the sum data rates of ground UEs in a dynamic UAV-assisted network. Results have shown that the proposed algorithm outperforms other existing dynamic deployment algorithms.

There is a potential direction for the future works. For example, in our research, we have considered a relatively large area where the time step is set to be relatively small, therefore, the action taken by one agent has little impact on the other UAV-BSs. Using a multi-agent reinforcement learning to enable multiple UAVs to take actions while considering their interactive impacts will be performed in the future study.

REFERENCES

[1] Y. Zeng, R. Zhang, and T. J. Lim, “Wireless communications with unmanned aerial vehicles: opportunities and challenges,” IEEE Communications Magazine, vol. 54, no. 5, pp. 36–42, May 2016.

[2] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, “Mobile unmanned aerial vehicles (uavs) for energy-efficient internet of things communications,” IEEE Trans. Wireless Commun., vol. 16, no. 11, pp. 7574–7589, Nov. 2017.

[3] Y. Huo, X. Dong, T. Lu, W. Xu, and M. Yuen, “Distributed and multi-layer uav networks for next-generation wireless communication and power transfer: A feasibility study,” IEEE Internet Things J., 2019.

[4] P. S. Bithas, E. T. Michailidis, N. Nomikos, D. Vouyioukas, and A. G. Kanatas, “A survey on machine-learning techniques for uav-based communications,” vol. 19, no. 23, p. 5170, 2019.

[5] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, “Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage,” IEEE Commun. Lett., vol. 20, no. 8, pp. 1647–1650, Aug. 2016.

[6] 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.

[7] W. Shi, J. Li, W. Xu, H. Zhou, N. Zhang, and X. Shen, “3D drone-cell deployment optimization for drone assisted radio access networks,” in Proc. IEEE/CIC Int. Conf. Communications in China (ICCC), Oct. 2017, pp. 1–6.

[8] H. Kim, J. Park, M. Bennis, and S. Kim, “Massive uav-to-ground communication and its stable movement control: A mean-field approach,” in 2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), 2018, pp. 1–5.

[9] 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 Communications Conf. (GLOBECOM), Dec. 2018, pp. 1–6.

[10] H. Bayerlein, P. De Kerret, and D. Gesbert, “Trajectory optimization for autonomous flying base station via reinforcement learning,” in Proc. IEEE 19th Int. Workshop Signal Processing Advances in Wireless Communications (SPAWC), Jun. 2018, pp. 1–5.

[11] X. Liu, Y. Liu, and Y. Chen, “Reinforcement learning in multiple-UAV networks: Deployment and movement design,” IEEE Trans. Veh. Technol., vol. 68, no. 8, pp. 8036–8049, Aug. 2019.

[12] Y. Chen, X. Lin, T. Khan, and M. Mozaffari, “Efficient drone mobility support using reinforcement learning,” in 2020 IEEE Wireless Communications and Networking Conference (WCNC), May 2020, pp. 1–6.

[13] M. Volodymyr, K. Koray, S. David, A. R. Andrei, and V. Joel, “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015.

[14] E. Altman, Constrained Markov Decision Processes. Boca Raton, Fl, USA: CRC Press, 1999.

[15] J. Ren, G. Zhang, and D. Li, “Multicast capacity for vanets with directional antenna and delay constraint under random walk mobility model,” IEEE Access, vol. 5, pp. 3958–3970, 2017.