Trajectory Optimization for Autonomous Flying Base Station via Reinforcement Learning

Harald Bayerlein, Paul de Kerret, and David Gesbert
Communication Systems Department, EURECOM
Sophia Antipolis, France
Email: {harald.bayerlein, paul.dekerret, gesbert}@eurecom.fr

***Abstract*—In this work, we study the optimal trajectory of an unmanned aerial vehicle (UAV) acting as a base station (BS) to serve multiple users. Considering multiple flying epochs, we leverage the tools of reinforcement learning (RL) with the UAV acting as an autonomous agent in the environment to learn the trajectory that maximizes the sum rate of the transmission during flying time. By applying Q-learning, a model-free RL technique, an agent is trained to make movement decisions for the UAV. We compare table-based and neural network (NN) approximations of the Q-function and analyze the results. In contrast to previous works, movement decisions are directly made by the neural network and the algorithm requires no explicit information about the environment and is able to learn the topology of the network to improve the system-wide performance.**

I. Introduction

Compared to traditional mobile network infrastructure, mounting base stations (BSs) or access points (APs) on unmanned aerial vehicles (UAVs) promises faster and dynamic network deployment, the possibility to extend coverage beyond existing stationary APs and provide additional capacity to users in localized areas of high demand, such as concerts and sports events. Fast deployment is especially useful in scenarios when sudden network failure occurs and delayed re-establishment not acceptable, e.g. in disaster and search-and-rescue situations [1]. In remote areas where it is not feasible or economically efficient to extend permanent network infrastructure, high-flying balloons or unmanned solar planes (as in Google's project Loon and Facebook's Internet.org initiative) could provide Internet access to half the world's population currently without it.

In all mentioned scenarios where flying APs hold promise, a decisive factor for the system's ability to serve the highest possible number of users with the best achievable Quality of Service (QoS) is the UAV's location. Previous work has either addressed the placement problem of finding optimal positions for flying APs (e.g. [2], [3]) or optimizing the UAV's trajectory from start to end [4]–[7]. Whereas fixed locations fulfilling a certain communication network's goal are determined in the placement problem, the alternative is to embed the optimization of the communication system with the path planning of the UAV base station. This allows for optimizing the users' QoS during the whole flying time as well as combining it with other mission critical objectives such as energy conservation by reducing flying time (e.g. [2] and [6]) or integrating landing spots for the UAV in the trajectory [4].

The authors acknowledge the support of the SeCIF project within the French-German Academy for the Industry of the Future as well as the support from the PERFUME project funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 670896).

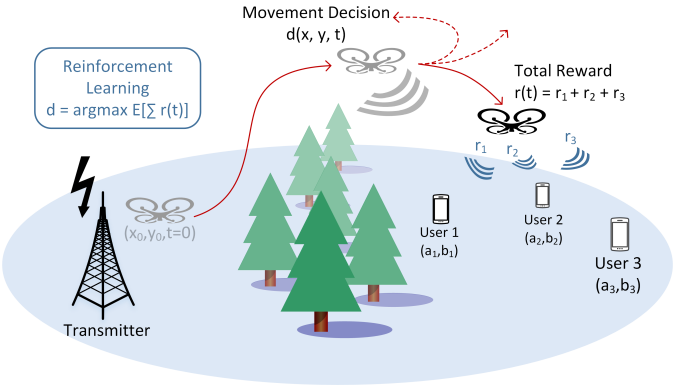

Fig. 1. UAV BS optimizing its trajectory to maximize the sum rate of the transmission to a group of users, e.g. in case of stationary transmitter failure.

In this work and as depicted in figure 1, we consider the UAV acting as a BS serving multiple users maximizing the sum of the information rate over the flying time, but a multitude of other applications exist. [8] and [9] provide summaries of the general challenges and opportunities. In [4], [10] and [11], the authors investigate an IoT-driven scenario where an autonomous drone gathers data from distant network nodes. The authors of [7] and [12] work on an application where an existing ground-based communications network could be used for beyond line-of-sight (LOS) control of UAVs if resulting interference within the ground network is managed. In [2] and [5], a scenario similar to this work is considered where a UAV-mounted BS serves a group of users. Whereas the authors in [5] also maximize the sum rate of the users, the goal in [2] is to cover the highest possible number of users while minimizing transmit power.

Recent successes in the application of deep reinforcement learning to problems of control, perception and planning achieving superhuman performance, e.g. playing Atari video games [13], have created interest in many areas, though RL-based path planning for mobile robots and UAVs in particular has not been investigated widely. Deep learning applications

in UAV guidance focus often on perception and have mostly auxiliary functions for the actual path planning, see [14] for a review. In [3], a radio map is learned which is then used to find optimal UAV relay positions. In [7] a deep RL system based on echo state network (ESN) cells is used to guide cellular-connected UAVs towards a destination while minimizing interference.

Our work focuses on a different scenario where the UAV carries a base station and becomes part of the mobile communication infrastructure serving a group of users. Movement decisions to maximize the sum rate over flying time are made directly by a reinforcement Q-learning system. Previous works not employing machine learning often rely on strict models of the environment or assume the channel state information (CSI) to be predictable. In contrast, the Q-learning algorithm requires no explicit information about the environment and is able to learn the topology of the network to improve the system-wide performance. We compare a standard table-based approach and a neural network as Q-function approximators.

II. System Model

A. UAV Model

The UAV has a maximum flying time T, by the end of which it is supposed to return to a final position. During the flying time $t \in [0, T]$, the UAV's position is given by $(x(t), y(t))$ and a constant altitude H. It is moving with a constant velocity V. The initial position of the UAV is (x_0, y_0), whereas (x_f, y_f) is the final position. $x(t)$ and $y(t)$ are smooth functions of class C^∞ and defined as

$$x : \begin{pmatrix} [0,T] \to \mathbb{R} \\ t \to x(t) \end{pmatrix} \qquad y : \begin{pmatrix} [0,T] \to \mathbb{R} \\ t \to y(t) \end{pmatrix} \tag{1}$$

subjected to

$$\begin{aligned} x(0) &= x_0, & y(0) &= y_0 \\ x(T) &= x_f, & y(T) &= y_f \end{aligned}$$

The UAV's constant velocity is enforced over the time derivatives $\dot{x}(t)$ and $\dot{y}(t)$ with

$$\sqrt{\dot{x}^2(t) + \dot{y}^2(t)} = V, \ t \in [0, T] \tag{2}$$

B. Communication Channel Model

The communication channel between the UAV AP and a number of K users is described by the log-distance path loss model including small-scale fading and a constant attenuation factor in the shadow of the obstacle.

The communication link is modeled as an orthogonal point-to-point channel. The information rate of the k-th user, $k \in \{1, ..., K\}$ located at a constant position $(a_k, b_k) \in \mathbb{R}^2$ at ground level is given by

$$R_k(t) = \log_2 \left(1 + \frac{P}{N} \cdot L_k\right) \tag{3}$$

with transmit power P, noise power N and pathloss L_k of the k-th user. The UAV-user distance $d_k(t)$ with the UAV at constant altitude H and all users at ground level is given as

$$d_k(t) = \sqrt{H^2 + (x(t) - a_k)^2 + (y(t) - b_k)^2} \tag{4}$$

With the pathloss exponent set to $\alpha = 2$ for vacuum, the pathloss for user k is given as

$$L_k = d_k(t)^{-\alpha} \cdot 10^{X_{Rayleigh}/10} \cdot \beta_{shadow} \tag{5}$$

where small-scale fading was modeled as a Rayleigh distributed random variable $X_{Rayleigh}$ with scaling factor $\sigma = 1$. The attenuation through obstacle obstruction was modeled with a discrete factor $\beta_{shadow} \in \{1, 0.01\}$ which is set to $\beta_{shadow} = 0.01$ in the obstacle's shadow and to $\beta_{shadow} = 1$ everywhere else. Using the described model, the maximization problem can be formulated as

$$\max_{x(t), y(t)} \int_{t=0}^{T} \sum_{k=1}^{K} R_k(t) dt \tag{6}$$

To guarantee that a feasible solution exists, T and V must be chosen such that the UAV is at least able to travel from initial to final position along the minimum distance path, i.e. $VT \geq \sqrt{(x_f - x_0)^2 + (y_f - y_0)^2}$.

III. Fundamentals of Q-Learning

Q-learning is a model-free reinforcement learning method firstly proposed by Watkins and developed further in 1992 [15]. It is classified as model-free because it has no internal representation of the environment.

Reinforcement learning in general proceeds in a cycle of interactions between an agent and its environment. At time t, the agent observes a state $s_t \in S$, performs an action $a_t \in A$ and subsequently receives a reward $r_t \in \mathbb{R}$. The time index is then incremented and the environment propagates the agent to a new state s_{t+1}, from where the cycle restarts.

Q-learning specifically allows an agent to learn to act optimally in an environment that can be represented by a Markov decision process (MDP). Consider a finite MDP $\langle S, A, P, R, \gamma \rangle$ with state space S, action space A, state transition probability $P_a(s, s') = Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$, reward function $R_a(s, s')$ and discount factor $\gamma \in [0, 1)$ which controls the importance of future rewards in relation to present reward.

The goal for the agent is to learn a behavior rule that maximizes the reward it receives. A behavior rule that tells it how to select actions given a certain state is referred to as a policy and can be stochastic in general. It is given as

$$\pi(a|s) = Pr\left[a_t = a \mid t_t = s\right] \tag{7}$$

Q-learning is based on iteratively improving the state-action value function (or Q-function) which represents an expectation of the future reward when taking action a in state s and following policy π from thereon after. The Q-function is

$$Q^\pi(s, a) = E_\pi\{R_t \mid s_t = s, a_t = a\} \tag{8}$$

where the discounted sum of all future rewards at current time t is called the return $R_t \in \mathbb{R}$ given by

$$R_t = \sum_{k=0}^{T-1} \gamma^k r_{t+1+k} \tag{9}$$

with discount factor $\gamma \in [0, 1)$ as set in the MDP definition and reaching the terminal state at time $t + T$. Given the Q-function with perfect information $Q^{\pi^*}(s, a)$, an optimal policy can be derived by selecting actions greedily:

$$\pi^*(a|s) = \arg\max_a Q^{\pi^*}(s, a) \tag{10}$$

From combining (8) and (9) it follows that R_t can be approximated in expectation based on the agent's next step in the environment. The central Q-learning update rule to make iterative improvements on the Q-function is therefore given by

$$Q^\pi(s_t, a_t) \leftarrow Q^\pi(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q^\pi(s_{t+1}, a) - Q^\pi(s_t, a_t) \right) \tag{11}$$

with learning rate $\alpha \in [0, 1]$ determining to what extend old information is overridden and discount factor $\gamma \in [0, 1)$ balancing the importance of short-term and long-term reward. γ approaching 1 will make the agent focus on gaining long-term reward, whereas choosing $\gamma = 0$ will make it consider only the immediate reward of an action. A value of $\gamma = 1$ could lead to action values diverging. Q-learning will converge to the optimal policy regardless of the exploration strategy being followed, under the assumption that each state-action pair is visited an infinite number of times, and the learning parameter α is decreased appropriately [16].

IV. Q-learning for Trajectory Optimization

A. Table-based Q-learning

In this section, we describe how the Q-learning algorithm was adapted for trajectory optimization of a UAV BS inside a simulated environment with the Q-function being approximated by a four-dimensional table of Q-values. Each Q-value represents thereby a unique state-action pair and a higher value relative to other values promises a higher return according to definition (8).

In order to promote initial exploration of the state space, the Q-table is initialized with high Q-values to entice the agent to visit each state-action pair at least once, a concept know as optimism in the face of uncertainty. After the UAV's position (x_0, y_0) and the time index $t = 0$ have been initialized, the agent makes its first movement decision according to the ϵ-greedy policy, where with probability $\epsilon \in [0, 1]$ a random action to explore the state space is taken and in all other cases the action that maximizes the Q-function. Therefore a balance must be found between the share of random and non-random actions which is referred to as the *exploration-exploitation trade-off* [16]. The probability for random actions ϵ is exponentially decreasing over the learning time.

The agent's initial movement decision starts the first learning episode which terminates upon reaching the maximum flying time T. Each new movement decision is evaluated by the environment according to the achieved sum rate computed with the channel model described in II-B and a numerical reward based on the rate result issued to the agent. The reward is then used to update the Q-value of the state and chosen action according to the rule defined in (11). As the drone is propagated to its new position and the time index t is incremented, the cycle repeats until the maximum flying time is reached and the learning episode ends. Random action probability ϵ is decreased and drone position and time index are reset for the start of a new episode. The number of episodes must be chosen so that sufficient knowledge of environment and network topology have accumulated through iterative updates of the Q-table.

B. NN-based Q-learning (Q-net)

Representation of the Q-function by a table is clearly not practical in large state and actions spaces as table size increases exponentially. Instead, the Q-function can be represented by an alternative nonlinear function approximator, such as a neural network (NN) composed of connected artificial neurons organized in layers.

A model with two hidden layers, each with a number of $n_{nodes} = 100$ neurons, proofed adequate to make a direct comparison with the standard table-based approach. Choosing the right architecture and learning parameters is in general a difficult task and has to be done through heuristics and extensive simulations. The NN input was chosen to contain only the minimal information of one state space sample, feeding current position (x_t, y_t) of the drone and time index t into the network denoted Q^π_θ with NN parameters θ. Four output nodes directly represent the Q-values of the action space. The neural network was implemented using Google's TensorFlow library.

The basic procedure of the NN Q-learning algorithm is the same as in the table-based approach described in the previous section IV-A. However during training, the weights of the network are iteratively updated based on the reward signal such that the output Q-values better represent the achieved reward using the update rule (11). To avoid divergence and oscillations typically associated with NN-based Q-learning, the training process makes use of the replay memory and target network improvements described in [13].

V. Simulation

A straight-forward, simulated environment was set up to evaluate the Q-learning algorithm. The state space $S = \{x, y, t\}$ was chosen to contain position of the drone and time. For simplicity, available actions for the drone were limited to movement in four directions $A = \{'up', 'right', 'down', 'left'\}$ within the plane of constant altitude $H = 20$. It follows that there are four Q-values representing the action space for each position in the grid and each time index, as is shown in figure 2 exemplarily for one position and time index.

A. Environment

The simulated environment, as depicted in figure 2, is based on a 15 by 15 grid world which is populated at initialization with two static users at ground level and a cuboid obstacle with height equal to the constant altitude of the drone standing on a 2 by 4 ground plane. The initial and final position of the UAV are set to the lower left corner $(x_0, y_0) = (x_f, y_f) = (0, 0)$.

The obstacle obstructs the LOS connection between users and UAV BS in part of the area. The signal strength in the shadowed part, shown as gray in figure 2, is reduced by a fixed factor of $\beta_{shadow} = 0.01$. After initialization of the environment and placement of users and obstacle, a shadowing map for the whole area is computed utilizing ray tracing, which is then used as a lookup table during the learning process. In addition, random samples at each new time index from a Rayleigh distributed random variable $X_{Rayleigh}$ modeling small-scale fading are used to compute the current information rate for each user according to the channel model equations (3) and (5). The resulting sum rate forms the basis of the reward signal.

B. Learning Parameters

The main component of the reward signal is the achieved sum rate between users and BS. An additional negative reward is added if the action chosen by the agent leads the UAV to step outside the 15 by 15 grid. The third component can be added by a safety check that activates when the UAV fails to make the decision to return to the landing position before the maximum flying time $T = 50$ is reached. The safety system then activates and forces the UAV to return while awarding a negative reward for each necessary activation of the system.

Except when the safety system is activated, movement decisions are made based on the ϵ-greedy policy described in IV-A. The probability for random actions ϵ is exponentially decreasing over the learning time with decay constant $\lambda_{action} = 14$ for NN and table-based approach alike.

No explicit rules exist to choose the learning parameters and the parameters for the update rule (11) in general, which is why they have to be found through a combination of heuristics and search on the parameter space. For the table-based approximation, a combination of constant learning rate $\alpha_{table} = 0.3$ and number of learning episodes $n_{table} = 800,000$ was selected. As the goal in our scenario independent of approach is to maximize the sum rate over the whole flying time, the discount factor was set to $\gamma = 0.99$ to make the agent focus on long-term reward for both approaches.

The learning rate in the update rule (11) for the NN-based approach is set to $\alpha_{nn} = 1$. Instead, the learning speed during NN training is controlled with the gradient descent step size, which is exponentially decayed with decay constant $\lambda_{gradient} = 5$ over the whole training time from a value of 0.005 to 0.00005. A number of $n_{nn} = 27,000$ learning episodes proved sufficient for the training of the NN.

VI. Results

The final learned trajectory by the table-based approach is depicted in figure 2 while the development of the resulting sum rate per episode during learning is shown in figure 3 for both approaches. It is important to note that the number of episodes in figure 3 is clipped due to the fact that the table-based solution only converged after $n = 800,000$ episodes in comparison to $n = 27,000$ for Q-net. The NN approximator therefore shows a much higher training data efficiency, mainly due to the fact that training data can be reused in the NN training. The sum rate shows a high increase in the respective first third of the learning phase when the rough layout of the trajectory is learned. Exploration slows down in the later phases of the learning process, which means that only details in the trajectory change and the absolute impact on the sum rate is consequently small.

The final trajectory shows that the agent is able to infer information about network topology and environment from the reward signal. Both approaches, the table-based and NN approximators, converge to a trajectory with the same characteristics. Specifically, the agent's behavior shows that it learned the following:

- The UAV reaches the maximum cumulative rate point between the two users on a short and efficient path.
- It avoids flying through the shadowed area keeping the sum rate high during the whole flying time.
- While the action space does not allow for hovering on one position, the drone learns to circle around the maximum cumulative rate point.
- The agent decides to return to its landing position in time to avoid crashing and on an efficient trajectory.

In this simple environment, both approaches are able to find efficient trajectories in reasonable computation time. This changes for larger state spaces. Evaluating both approaches in a 30 by 30 grid environment with four randomly placed users and obstacles each showed that table-based learning is not able to find a trajectory outperforming random movement decisions within a realistic computation time. In the same environment, Q-net converges to a high sum rate trajectory within $n_{NN} = 30,000$ training episodes and a computation time of about one hour on a basic office computer.

VII. Discussion

A. Summary

We have introduced a novel Q-learning system to directly make movement decisions for a UAV BS serving multiple users. The UAV acts as an autonomous agent in the environment to learn the trajectory that maximizes the sum rate of the transmission over the whole flying time without the need for explicit information about the environment. We have formulated a maximization problem for the sum rate, which we solved iteratively by approximating the Q-function. Our simulation has shown that the agent is able to learn the network topology and infer information about the environment to find a trajectory that maximizes the sum rate and which lets

the UAV autonomously return to its landing spot within the flying time limit. Comparing table-based and neural network approximators for the Q-function showed that using a table is not feasible for large state spaces, but training a NN provides the necessary scalability and proved to be more efficient using less training data then table-based Q-learning.

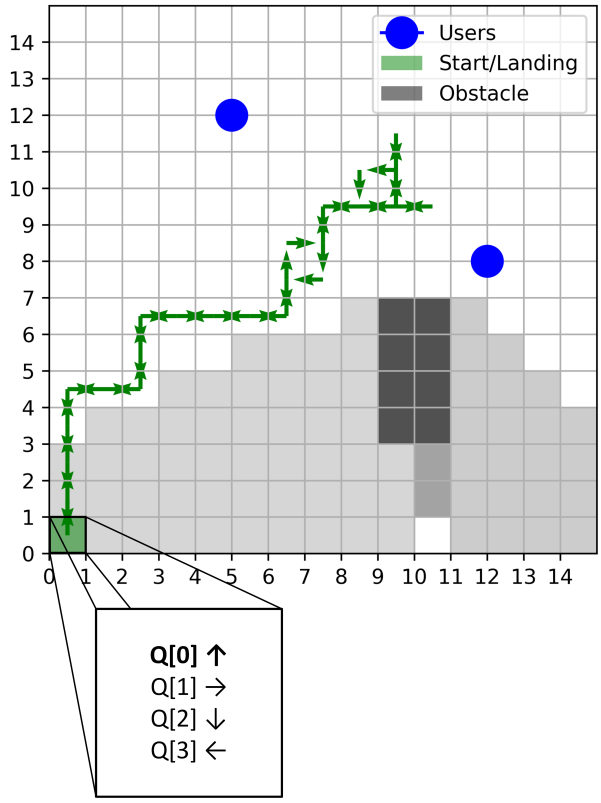

Fig. 2. The final trajectory for the table-based approach after completed episode $n_{table} = 800,000$ depicted in the simulation environment with two users and one obstacle. The gray area is in the shadow of the obstacle. As a visualization for the table, the four Q-values, one for each action, are shown for the start position $(0, 0)$. At time index $t = 0$, the Q-value for action 'up' was learned to promise the highest future return.

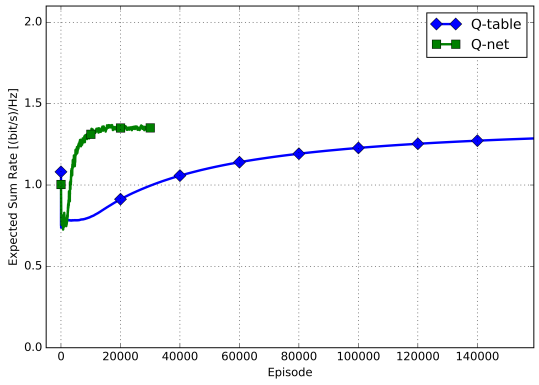

Fig. 3. Sum information rate between UAV BS and users per episode over learning time comparing table-based and NN approximators of the Q-function. The plot only shows a clipped range of learning episodes as the table-based solution converges after $n_{table} = 800,000$ episodes, whereas Q-net only needs $n_{NN} = 27,000$ episodes to come to a similar solution.

B. Limitations and Future Work

The relatively high number of learning episodes to obtain the described results shows a limitation in choosing a Q-learning approach in comparison to methods in previous works. This is a consequence of the generality of Q-learning and the avoidance to make any assumptions about the environment contained within the approach. Integrating even a coarse model of the environment with an alternative model-based RL method would result in faster convergence, but also entail a loss in learning universality. The long learning time is put into perspective by the fact that the main training can be completed offline based on the prior distribution before a shorter adaptation phase to the true setting. Future work will include considerations about dynamically changing environments, as well as a more detailed look at real-world constraints such as the energy efficiency of the learned trajectory.

References

[1] K. Namuduri, "Flying cell towers to the rescue," *IEEE Spectrum*, vol. 54, no. 9, pp. 38–43, Sep. 2017.

[2] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station (UAV-bs) for energy-efficient maximal coverage," *IEEE Wireless Communications Letters*, vol. 6, no. 4, pp. 434–437, Aug. 2017.

[3] J. Chen and D. Gesbert, "Optimal positioning of flying relays for wireless networks: A LOS map approach," in *IEEE International Conference on Communications (ICC)*, 2017.

[4] R. Gangula, D. Gesbert, D.-F. Külzer, and J. M. Franceschi Quintero, "A landing spot approach to enhancing the performance of UAV-aided wireless networks," in *IEEE International Conference on Communications (ICC) (accepted), 2018, Kansas City, MO, USA*.

[5] R. Gangula, P. de Kerret, O. Esrafilian, and D. Gesbert, "Trajectory optimization for mobile access point," in *Asilomar Conference on Signals, Systems, and Computers, 2017, Pacific Grove, CA, USA*.

[6] Y. Zeng and R. Zhang, "Energy-efficient uav communication with trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3747–3760, 2017.

[7] U. Challita, W. Saad, and C. Bettstetter, "Cellular-connected uavs over 5g: Deep reinforcement learning for interference management," *arXiv preprint arXiv:1801.05500*, 2018.

[8] 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, 2016.

[9] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in uav communication networks," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1123–1152, 2016.

[10] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in uav enabled wireless sensor network," *submitted to IEEE Wireless Commun. Letters, available online at https://arxiv. org/abs/1708.00221*, 2017.

[11] J. Gong, T.-H. Chang, C. Shen, and X. Chen, "Aviation time minimization of uav for data collection over wireless sensor networks," *arXiv preprint arXiv:1801.02799*, 2018.

[12] B. Van der Bergh, A. Chiumento, and S. Pollin, "Lte in the sky: trading off propagation benefits with interference costs for aerial nodes," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 44–50, 2016.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[14] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy, "A review of deep learning methods and applications for unmanned aerial vehicles," *Journal of Sensors*, vol. 2017, no. 3296874, 2017.

[15] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[16] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*, 2nd ed. Cambridge, Massachusetts: MIT Press, 2017.