A Reinforcement Learning-based Path Planning for Collaborative UAVs

Shahnila Rahim
School of Computer Science and Engineering,
Kyungpook National University,
South Korea.
shahnila.rahim@knu.ac.kr

Mian Muaz Razaq
School of Computer Science and Engineering,
Kyungpook National University,
South Korea.
mianmuaz97@gmail.com

Shih Yu Chang
Department of Applied Data Science,
San Jose University,
USA.
shihyu.chang@sjsu.edu

Limei Peng
School of Computer Science and Engineering,
Kyungpook National University.
South Korea.
auroraplm@knu.ac.kr (Corresponding author)

***Abstract*—Unmanned Aerial Vehicles (UAVs) are widely used in search and rescue missions for unknown environments, where maximized coverage for unknown devices is required. This paper considers using collaborative UAVs (Col-UAV) to execute such tasks. It proposes to plan efficient trajectories for multiple UAVs to collaboratively maximize the number of devices to cover within minimized flying time. The proposed reinforcement learning (RL)-based Col-UAV scheme lets all UAVs share their traveling information by maintaining a common Q-table, which reduces the overall time and the memory complexities. We simulate the proposed RL Col-UAV scheme under various simulation environments with different grid sizes and compare the performance with other baselines. The simulation results show that the RL Col-UAVs scheme can find the optimal number of UAVs required to deploy for the diverse simulation environment and outperforms its counterparts in finding a maximum number of devices in a minimum time.**

***Index Terms*—Reinforcement learning, Unmanned Aerial Vehicle (UAV), Path Planning, Collaborative UAVs**

I. Introduction

The future of the Internet of Things (IoT) is intended to autonomously determine the connection of devices, people, and processes. Drones or unmanned aerial vehicles (UAVs) have attracted plenty of industries, academics, and regulatory organizations, thanks to their capabilities of increasing the communication coverage and providing on-demand connectivity [1]. As a result, the role of UAVs is rising in many areas like military, civil, and public applications. Notably, when UAVs execute sensing tasks for IoT devices on request, they can hover at high altitudes to provide the line-of-sight (LoS) communications. Furthermore, UAVs can be deployed to provide network coverage for people in remote areas such as disaster zones, and UAVs can also be used for collecting data in wireless sensor network (WSN) [2]. However, the fast deployment of UAVs, the optimal number of UAVs required, and the efficient trajectory of deployed UAVs are critical in scenarios with unexpected network failures and stringent delay requirements.

Since reinforcement learning (RL) is capable of finding an optimal policy by learning in a trial-and-error manner, it is widely used in wireless communications such as UAV applications, collaborative robots etc. [3]. In particular, RL can train UAVs to improve their policy to achieve the main objective without having complete environmental information. Thus, RL is suitable for working with a dynamic and unknown environment where UAVs must learn autonomously to optimize the policy in a fully or partially observed environment. Various research focus on applying RL techniques in UAV control to attain the desired optimized trajectory for data collection, tracking, navigation and localization [1], [2], [4] and [5]. Whereas, most of the existing research on UAV applications are based on a single UAV [2] and [4]; nonetheless, due to limited on-board battery, it is pretty challenging for one UAV to increase the flight time to complete the whole desired task in most applications.

To solve the aforementioned problem, this paper proposes the scheme of Q-learning-based Collaborative UAVs (Col-UAV), where multiple UAVs work together to execute the IoT device sensing/detecting tasks for a given area faster and more efficiently. Nevertheless, it is challenging to successfully deploy multiple UAVs to share their information and complete the task collaboratively without having prior knowledge. Few efforts have been made on collaborative UAVs, though there is numerous existing research on collaborative robots.

In [6], several robots collaborate to perform computations in distributed DNN, and robots harvest the computational energy

SAC '22, April 25–29, 2022, Virtual Event,

ACM ISBN 978-1-4503-8713-2/22/04…$15.00
https://doi.org/10.1145/3477314.3507052

of low-energy robots, aiming to maximize the throughput in an adaptive environment. Each robot in a collaborative group shares the sensed environmental data immediately with its neighbors in [7]. Some research regarding collaborative UAVs has been done in the wireless sensor network. In [8], multi-linear programming is used to find the trajectories passing through the near estimated positions of the cluster heads. In [9], a mathematical model was used to show the constraints and uncertainties of flying UAVs, focusing on developing and analyzing the coordinate control system.

To the best of our knowledge, collaborative UAVs are not sufficiently explored in UAV applications where multiple UAVs work together to minimize the overall power consumption and flying time to achieve a common goal. This study proposes to use multiple UAVs to find the shortest collaborative trajectory based on Q-learning while maximizing the number of detected IoT devices in an unknown environment within a minimum flying time. The collaborative UAVs share their information by a common Q-table, aiming to minimize the overhead of repeated entries in the table and thus minimize the overall flying time required to maximize the devices that can be covered. Moreover, the proposed scheme can also find the optimal number of collaborative UAVs needed in different environments.

The remainder of this paper is organized as follows. Section II presents the problem formulation, which includes the details of the system model, RL environment, and description of the proposed Col-UAV scheme. Results are discussed in section III. Section IV concludes the paper.

II. Problem Formulation

A. System Model

This paper considers the UAV-assisted IoT networks, where k number of devices are randomly distributed in a certain geographical area. The set of k devices is denoted by $D = \{D_1, D_2, D_3, \ldots, D_k\}$. The area of interest (AoI) is presented as a square divided into $m \times n$ grids, where $m, n \in N$. The coordinate set of all grid cells is denoted by $c_{x,y}$, where $c_{x,y} = \{c_{1,1}, c_{1,2}..., c_{i,j}, ...\}$. Here, we assume the size of each grid is the same and equal to the UAV communication coverage. UAVs are flying over the AoI at a fixed altitude h, trying to maximize the number of devices can be covered within the minimum flying time T_{max}. UAVs start their mission from the initial starting position, i.e., SP_i, where i is the UAV index, i.e., $i = 1, 2, ...$ The position of the i^{th} UAV at time step t is defined as $U_i^t = (x_i^t, y_i^t, h)$. All UAVs stop at the common final position P_f as shown in Fig. 1.

Fig. 1 shows that the collaborative UAVs start their mission and try to improve their trajectory to maximize the number of devices that can be covered without colliding with each other. As the environment is unknown, UAVs are unaware of the number of devices and their location coordinates. Moreover, when a device is covered by one of the UAVs, it will be marked as collected so that the other UAVs try to avoid detecting it again. With the increasing area size, the optimal number of collaborative UAVs required to achieve the objective within minimum flight time changes.

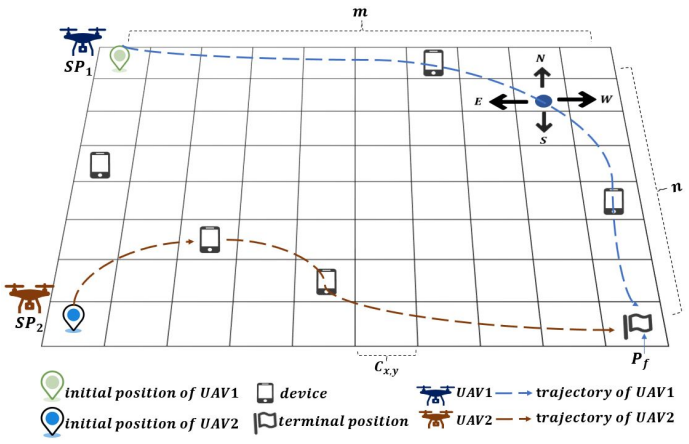

Fig. 1: An illustration of Q learning based collaborative UAVs.

Since UAVs do not have prior knowledge about IoT devices, it is pretty challenging to find the optimal trajectory with the traditional techniques. This paper proposes to solve this problem by reinforcement learning. Specifically, collaborative UAVs learn autonomously from their shared experiences by maintaining the detecting history in a common Q-table and improving their policies in a trial-and-error manner to achieve the overall objective.

B. Reinforcement Learning Environment

Reinforcement Learning (RL) is a process of learning from scratch [7]. In RL, an agent observes the environment and then takes some action, examines the consequential results of the action, and automatically adjusts the policies to achieve the objective based on the examination results. In summary, RL is a cycle of interaction between an environment and an agent. At time t, agents observe state $s_t \in S$, perform an action $a_t \in A$, and consequently receive some reward $r_t \in R$. The increment in time propagates the agent to the new state s_{t+1} in an environment where this cycle restarts.

Q-learning allows agent to act optimally in an environment represented by Markov Decision Model (MDP) [10]. We also formulate our problem with the MDP which consists of a four-element tuple (S, A, R, P_a) , where S, A and R represent the state, the action, and the reward function, respectively, and P_a is the state transition probability function defined as follows, i.e., $P_a(s, s') = P_r(s_{t+1} = s' | s_t = s, a_t = a)$. Particularly, at time slot t, agent observes state s_t in an environment and takes action a_t; after performing this action in the environment the state transits to s_{t+1} and the agent receives a reward r_t, which can be negative or positive. In our case, we have multiple agents, with each corresponding to a UAV, trying to cover as many devices as possible together. Correspondingly, we define the action, state and reward as follows:

1) **State**: The state at time t is composed of four parts $s_t = (\mu_i^t, \varnothing_i^t, c_{x,y}^t, \lambda_i^t)$:

- μ_i^t is the position of UAV_i at time slot t.
- $\varnothing_i^t$ is the remaining flying time of UAV_i at time *t*.
- $c_{x,y}^t$ is agent cell location at time *t*.
- λ_i^t are the coordinates of other collaborative UAVs in the environment at time *t*.

2) **Action**: As shown in Fig. 1, UAV_i has four moving actions at each state and one additional hovering action when it is finally assigned to detect an IoT device; therefore, the action set is {north, south, east, west, hover}, which are represented by the set $A = \{+x, +y, -x, -y, 0\}$ to represent the above action, where $+y$, $-y$, $+x$, or $-x$ express that UAV_i is changing its states by moving upwards (north), downwards (south), left (east), or right (west), respectively. In contrast, zero represents that UAV_i is hovering and will not change its direction.

3) **Reward**: The reward in the perspective of Col-UAV assisted path planning to cover IoT devices must encourage the agents to minimize the flying time while maximizing the number of devices detected. Regarding this, when the collaborative UAVs reach the terminal positions at some time slot t within the maximum flying time t_{max}, they respectively will get a positive reward. Whereas small penalty will be given to each UAV_i for every action taken without accomplishing the mission. Each UAV in a collaborative group will be rewarded based on their own actions in each state. The reward of UAV_i, at each time step t, i.e., r_i^t, is defined as,

$$r_i^t = \begin{cases} +\omega, & \text{if } \mu_i^t = d_j, \\ -\omega_1, & \text{if } \varnothing_i^t < 0, \\ +\omega_2, & \text{if } \mu_i^t = P_f, t \leq T_{max}, \\ -1, & \text{otherwise.} \end{cases} \tag{1}$$

where ω, ω_1 and ω_2 are positive constant numbers. In eq. (1), a positive reward is given when UAV_i and device d_j position is same, marking the j^{th} device as detected. Furthermore, UAV_i will get a negative reward when the remaining flying time is less than 0. Moreover, a positive reward is given when UAV_i reaches the final position. The expected discounted cumulative reward for UAV_i at the current time, i.e., R_i^t, can be defined as,

$$R_i^t = \sum_{t=1}^{n} \gamma^{t-1} r_i^t(s, a_i) \tag{2}$$

4) **State Transition**: The location enumerates of UAV_i can be expressed as,

$$\mu_i^{t+1} = \begin{cases} \mu_i^t + (0,1), & \text{if } a_i^t = North\ . \\ \mu_i^t - (0,1), & \text{if } a_i^t = South. \\ \mu_i^t + (1,0), & \text{if } a_i^t = East. \\ \mu_i^t - (1,0), & \text{if } a_i^t = West. \\ \mu_i^t, & \text{if } a_t = Hovering, \end{cases} \tag{3}$$

where ±(0,1) and ±(1,0) indicate the change in the coordinate of UAV_i corresponding to the action.

C. Q-Learning-Based Collaborative UAVs

This section describes the proposed Col-UAVs scheme in details. We assume all the UAVs in a collaborative group are identical and have the same sets of actions and states mentioned above in II-B. Assume the collaborative UAVs can communicate with each other and share information. We employ the Q-learning in these collaborative UAVs so that agent i can improve the state-action pair by learning from the behavior rule that maximizes the reward it receives. The policy π briefly tells how to select the actions for UAV_i, i.e., a_i, at some particular state. For UAV_i, the policy is selected by ϵ-greedy to keep the balance between exploration and exploitation actions, where ϵ is the probability of choosing either to explore in next action. This is defined by eq. (4),

$$\pi_i(s) = \begin{cases} a \in argMaxQ_i(s, a_i), & \text{with the probability 1-}\epsilon; \\ \text{a random action } a_i, & \text{otherwise.} \end{cases} \tag{4}$$

In our approach, UAVs maintain only one Q-table which is shared by all collaborative UAVs, which also reduces the overall size of Q-table as redundant state-action pairs are not stored in the table. In this way, each UAV will not only learn from their own experiences but also from other UAVs' experience. Q-function for UAVs working in a collaborative environment is defined by bellman equation [11] as in eq. (5),

$$Q(s_i, a_i) = Q(s_i, a_i) + \alpha * (R_i^t + \gamma * Q(s_i', a_i') - Q(s_i, a_i)) \tag{5}$$

where $0 \leq \alpha \leq 1$ is the learning rate determining to what extent old information is overridden and the discount factor $0 \leq \gamma \leq 1$ balances the importance of long-term and short-term reward. Choosing $\gamma = 0$ will consider only the immediate reward of an action, whereas, $\gamma = 1$ will make UAV_i focus on gaining long-term reward.

The details are shown in Algorithm 1. At the beginning of the process, some parameters are initialized, and the algorithm runs for the given number of episodes. Action is selected by ϵ-greedy policy defined in eq. (4). When the action is taken, the agent moves to the next state and observes the environment. If the UAV_i detects a device in the new state, it will get a high reward and mark the cell that contains the device as collected. Other UAVs will get a -1 penalty in that cell for the constant movement without completing the mission as defined in eq. (1). After receiving the reward, Q-value is updated. The episode runs until the UAV_i reaches the terminating position.

III. Performance Evaluation

A. Simulation Setup

The extensive simulation runs in the straightforward environment on jupyter notebook in python. We consider four simulation environments differing in the grid size for a given area. Specifically, we consider the scenarios of 10×10, 20×20, 30×30, and 50×50 equally-sized grids of length

Algorithm 1: Q-Learning-Based Collaborative UAVs for Path Planning

Input : Parameters for learning $\gamma \in$ [0,1], $\alpha \in$ [0,1], $\epsilon \in$ [0,1]; agents information: T_{max}, P_f, U_i, episodes; $Q_i(s, a_i) := 0$; number of UAVs
Output: Rewards: $R_{i,t}$; optimal policies: π_i; number of devices covered; flying time

while *epi* $\leq$ *Episodes* **do**
  while UAV_i *does not reach the terminating position* **do**
    Select action by $\epsilon - greedy$ policy in eq.(4);
    UAV_i takes action a_i;
    Moves to the next state s' and observes if any device is found;
    if a device is found;
    then receives a reward R_i and marks it as collected;
    receives a reward R_i;
    Update the Q-value based on eq. (5), i,e., $Q(s_i, a_i) = Q(s_i, a_i) + \alpha * (R_i^t + \gamma * Q(s_i', a_i') - Q(s_i, a_i))$;
    Update state, action and time step;
  end while
  epi = epi + 1;
end while

$25m \times 25m$. The starting coordinates are assigned to each UAV. For instance, the environment having grid 10×10 will start from origin (0,0) in the case of using one UAV, while for the case of using two UAVs, one will start from origin (0, 0) and other will start from other corner of the grid, such as (9,0). All the UAVs will try to reach the final position, which in this case can be (9, 9). There are 50 devices for 10×10 grids and 200 devices for all other environment settings. All the devices are randomly distributed in the given area. The other simulation parameters are given in Table I.

TABLE I: Simulation parameters

Parameter	Value
Episodes E	10000
Epsilon in ϵ-greedy	0.5
Learning Rate α	0.3
Discount Factor γ	0.6
UAVi Speed v	25 m/s
UAVi altitude h	25 m

B. Results and Analyses

In this subsection, we evaluate the performance of the proposed Q-learning-based Col-UAV scheme in terms of the optimal number of collaborative UAVs required for finding a path that can cover the maximum number of IoT devices. Specifically, we consider the following simulation environments, sayings an area with the grid size of 10×10 and a total of 50 IoT devices randomly distributed in environment 1, and a grid size of 20×20, 30×30, and 50×50 in environment 2, 3, and 4, respectively, with the same number of 200 IoT devices randomly distributed in the areas.

Algorithm 2: Random action Based UAVs for Path Planning

Input : Number of UAVs, Iterations
Output: Number of devices covered, time

while *epi* $\leq$ *Episodes* **do**
  while UAV_i *not reach to terminal* **do**
    Select action randomly;
    UAV_i takes action a_i;
    Check the episode, $if epi\%10 = 0$;
    then take the action a_i 2 times ;
    Moves to the next state s' and observe if the device is found.;
    if device if found ;
    then receives a reward R_i and mark it as collected;
    Update state, action and time step;
  end while
  epi = epi + 1
end while

To evaluate the performance of the proposed Col-UAV scheme, we consider the following three baseline schemes, saying one-UAV-based on Random Action (1UAV-R), two-UAV-based on Random Action (2UAV-R), and one-UAV-based on Q-learning (1UAV-QL). 1UAV-R uses a single UAV to complete the mission by taking random actions. 2UAV-R uses two collaborative UAVs that try to complete the mission by taking random actions. We use the double-action strategy, which is also used in 1UAV-R, to avoid the agent getting stuck between two states. Furthermore, once a UAV detects a device, that device will be marked as covered as presented in Algorithm 2. 1UAV-QL uses a single UAV to complete the mission based on Q-learning.

Fig. 2 shows the performance in environment 1. Fig. 2a shows the flying time of UAV(s) in 1000^{th} iteration of the four approaches, i.e., the proposed Col-UAV using two UAVs (2Col-UAV-QL), 1UAV-R, 2UAV-R, and 1UAV-QL. It depicts that the proposed 2Col-UAV-QL achieves the minimum flying time while covering 44 devices in 17 minutes, as shown in Fig. 2b. Fig. 2c shows the average number of steps taken by agent(s) to converge to optimal path while covering the devices. From these figures, we can clearly see that the proposed 2Col-UAV-QL scheme outperforms the other counterparts concerning the flying time and the number of steps. Moreover, it covers the maximum number of devices compared to other approaches. In contrast, 1UAV-R takes maximum time to complete the path while covering 39 out of 50 IoT devices.

Fig. 3 shows the performance in environment 2 where the

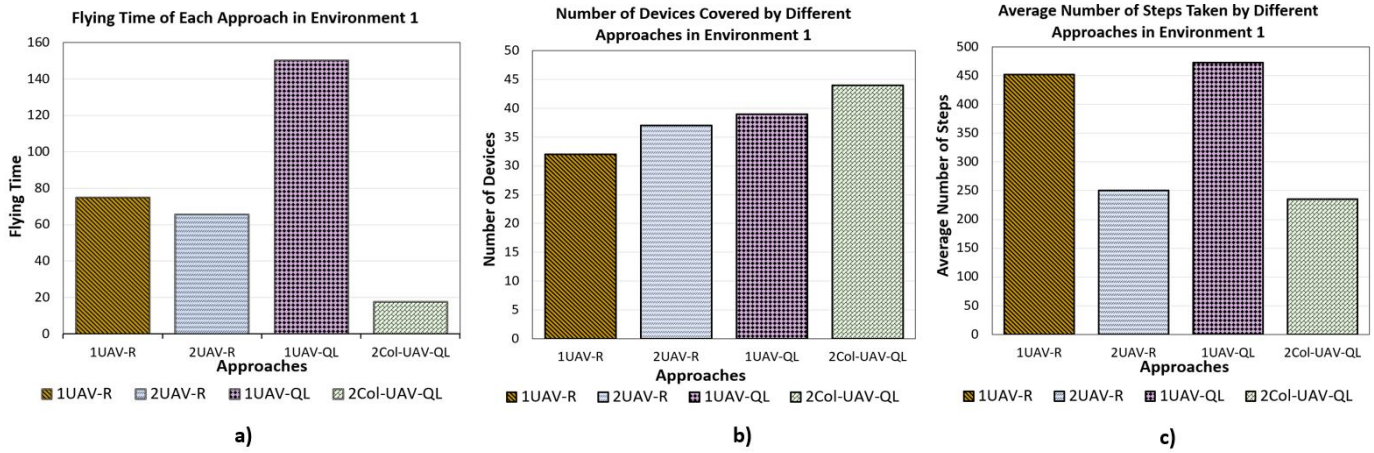

Fig. 2: Grid size: 10 x 10; IoT devices: 50. (a) Flying time for the proposed Col-UAV scheme and other naive baselines; (b) The number of devices covered in each approach; (c) The number of average steps taken in each episode.

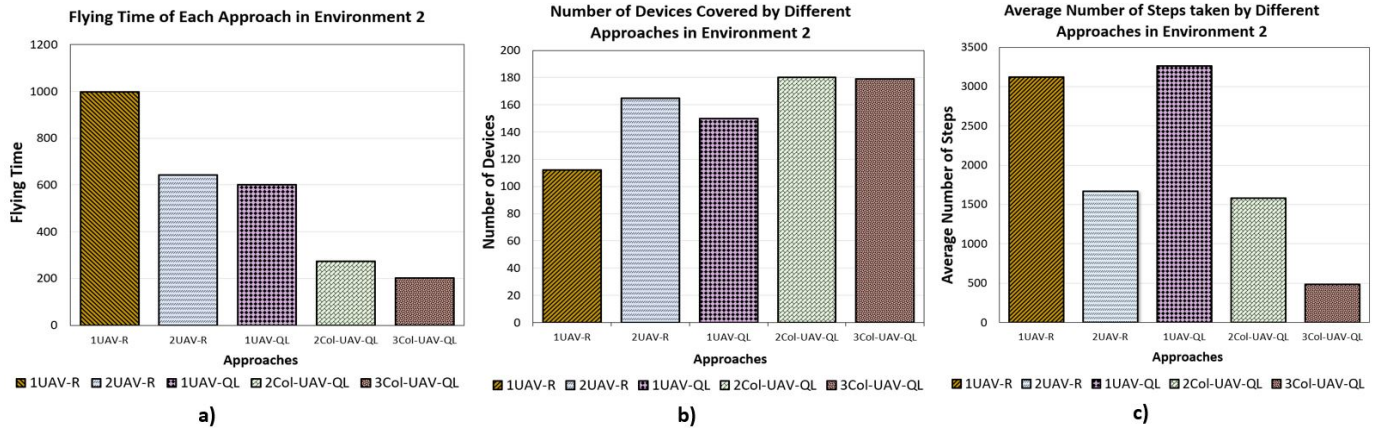

Fig. 3: Grid size: 20 x 20; IoT devices: 200. (a) Flying time for the proposed Col-UAV scheme and other naive baselines; (b) The number of devices covered in each approach; (c) The number of average steps taken in each episode.

grid size is extended to 20 x 20 with 200 randomly distributed IoT devices. In this environment, we consider the proposed Col-UAV scheme using two and three collaborative UAVs (2Col-UAV-QL and 3Col-UAV-QL), respectively, trying to find an optimal number UAVs. Fig. 3b shows the number of devices out of 200 covered by different schemes. Moreover, Fig. 3c depicts the overall average steps taken by an agent to converge. We can see that the optimal number of collaborative UAVs is two since there is no significant difference in the performance between 2UAV-QL and 3UAV-QL.

Fig. 4 shows the performance in environment 3 where the grid size is extended to 30 x 30 with 200 randomly distributed IoT devices. By observing Figs. 4a, 4b and 4c, 1UAV-R covers 86 devices in 605 minutes, 2UAV-R covers 105 devices in 385 minutes, and 1UAV-QL covers 95 devices in 923 minutes. On the other hand, both 2Col-UAVs-QL and 3Col-UAVs-QL achieve the same performance in covering almost the same number of devices, i.e., 157 and 160, respectively, in a minimum time when compared to others. It is clear that in an environment with a grid size of 30 x 30, the optimal number of UAVs required to deploy would be two.

Fig. 5 shows the performance for environment 4, which extends the grid size to 50 x 50. From Figs. 5a, 5b and 5c, we can observe that the proposed 3Col-UAV-QL performs significantly better than other approaches. The 1UAV-R scheme performs the worst in taking the maximum time and most steps to complete the task. Thus, for the environment with a grid size of 50 x 50, the optimal number of collaborative UAVs is three, which covers 138 devices in 1079 minutes.

IV. Conclusions

This paper considered using multiple UAVs to detect IoT devices in an unknown environment and proposed a reinforcement learning (RL)-based collaborative UAVs (Col-UAV) scheme. In the proposed Col-UAV scheme, UAVs can collaborate by sharing a common Q-table. Thus, they can autonomously and collaboratively find a trajectory path that covers the maximum number of IoT devices on their way. Given the related unreliability and challenges, we transformed

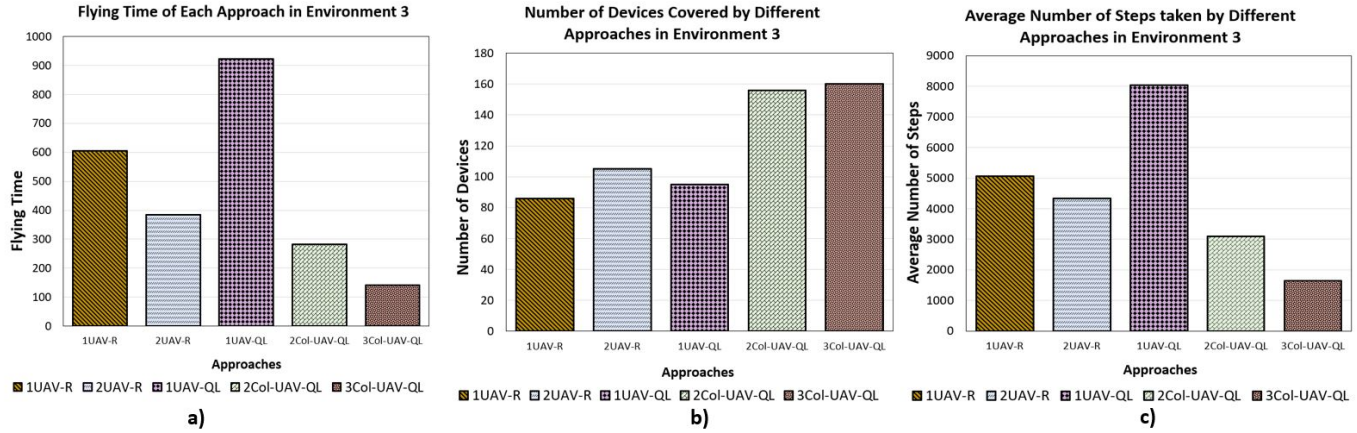

Fig. 4: Grid size: 30 x 30; IoT devices: 200. (a) Flying time for the proposed Col-UAV scheme and other naive baselines; (b) The number of devices covered in each approach; (c) The number of average steps taken in each episode.

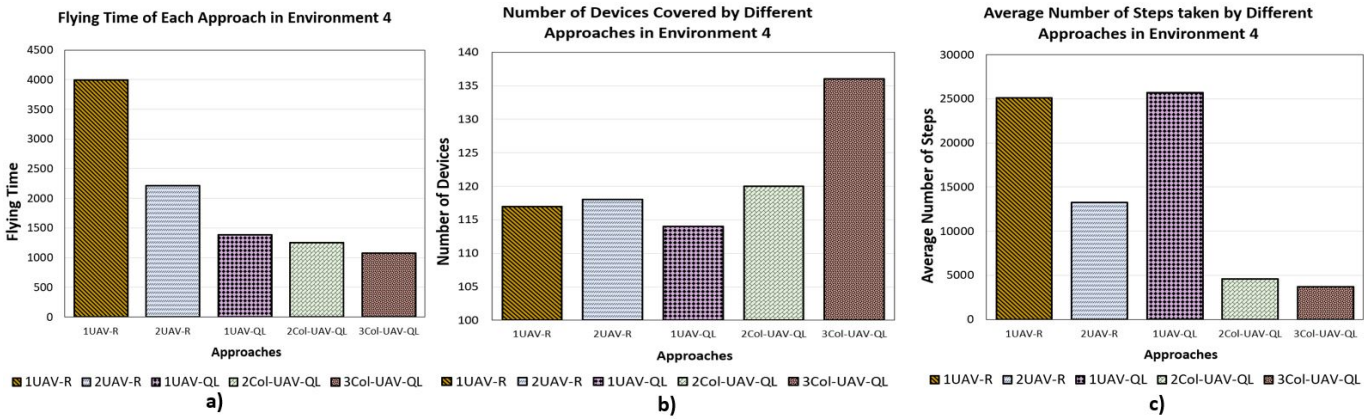

Fig. 5: Grid size: 50 x 50; IoT devices: 200. (a) Flying time for the proposed Col-UAV scheme and other naive baselines; (b) The number of devices covered in each approach; (c) The number of average steps taken in each episode.

our problem into MDP by dividing the area into equal small grid cells and applying the RL Q-learning approach therein, allowing the collaborative UAVs to optimize the trajectory without prior knowledge of the environment. The simulation results showed the superiority of our approach in different environment settings. Furthermore, we also discovered the optimal number of collaborative UAVs required to deploy in different environments with varying grid sizes.

V. Acknowledge

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (Grant No.: 2020R1I1A3072688).

References

[1] J. Cui and et al., "Adaptive uav-trajectory optimization under quality of service constraints: A model-free solution," *IEEE Access*, 2020.
[2] K. Nguyen and et al., "3d uav trajectory and data collection optimisation via deep reinforcement learning," *arXiv preprint*, 2021.
[3] H. La and et al., "Multirobot cooperative learning for predator avoidance," *IEEE Transactions on Control Systems Technology*, 2014.
[4] H. Pham and et al., "Reinforcement learning for autonomous uav navigation using function approximation," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2018.
[5] D. Ebrahimi and et al., "Autonomous uav trajectory for localizing ground objects: A reinforcement learning approach," *IEEE Transactions on Mobile Computing*, 2020.
[6] R. Hadidi and et al., "Distributed perception by collaborative robots," *IEEE Robotics and Automation Letters*, 2018.
[7] L. Kong and et al., "Adasharing: Adaptive data sharing in collaborative robots," *IEEE Transactions on Industrial Electronics*, 2017.
[8] D. Popescu and et al., "A collaborative uav-wsn network for monitoring large areas," *Sensors*, 2018.
[9] X. Zhu and et al., "Model of collaborative uav swarm toward coordination and control mechanisms study," *Procedia Computer Science*, 2015.
[10] Watkins and et al., "Q-learning," *Machine learning*, 1992.
[11] R. Sutton and et al., "Reinforcement learning: an introduction mit press," *Cambridge, MA*, 1998.
[12] M. Yi and et al., "Deep reinforcement learning for fresh data collection in uav-assisted iot networks," in *INFOCOM WKSHPS*, IEEE, 2020.
[13] Battocletti, *Reinforcement Learning approach for cooperative UAVs exploration of critical environments*. PhD thesis, Politecnico di Torino, 2021.