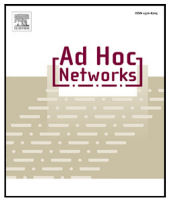
Ad Hoc Networks 157 (2024) 103463

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc



Energy efficient deployment of aerial base stations for mobile users in multi-hop UAV networks

Kyungho Ryu, Wooseong Kim *

Department of Computer Engineering, Gachon University, 1342, SeongNam, 13122, Gyeonggi, South Korea

ARTICLE INFO

Keywords:
UAV base station
Flying ad hoc network
Energy efficiency
Trajectory prediction
POMDP
Deep reinforcement learning

ABSTRACT

Unmanned aerial vehicles (UAVs) are popularly considered as aerial base stations in a Low-Altitude Platform (LAP) to provide wireless connections to ground users in disaster and digital divide situation. Most of previous studies have investigated energy efficient UAV deployment only for the stationary ground users. Contrarily, we propose an on-line deployment algorithm for ground mobile users (GMUs) who are partially observable in the multi-UAV system. In the framework of Partially Observable Markov Decision Process (POMDP) with large state and action space, we use the Monte Carlo Tree Search (MCTS) algorithm enhanced by the Double Progressive Widening (DPW) and Deep Neural Network (DNN)-based guidance. For online learning of the DNN, simulated samples are used with Proximal Policy Optimization (PPO) that prevents excessive training for a particular belief. From experiment in urban environment, the proposed scheme shows better learning performance than the MCTS with rollout policy and faster convergence than training with environment samples. In addition, it optimizes dual objectives proportionally in a trade-off between energy saving and throughput including prediction error on GMU location.

1. Introduction

Unmanned aerial vehicles (UAVs) are attractive technology for disaster management [1,2], surveillance for civil security and tactical ad-hoc networking, which provide wireless connectivity to ground mobile users (GMUs) who are isolated from cellular network services [3] and to ground sensors to collect information of surrounding environment such as fire detection [4,5]. To cover wide disaster or tactical areas, multiple UAVs constitute an ad-hoc network system such as an airborne network (called as Flying ad-hoc network (FANET)), which provides multi-hop connections to a ground control center (GCC) or remotely piloted aircraft systems (RPAS) as a gateway for Internet service. The GCC has separate narrow-band control and broad-band communication links that are used for UAV navigation and GMU traffic, respectively [6]; the control signal typically carried by a low-frequency channel, e.g., 960 MHz is assumed to reach directly UAVs across most of the tactical area while the data signal through a higher frequency channel, e.g., 5 GHz is relayed by multiple UAVs with higher order modulation.

The airborne network needs an efficient control algorithm to deploy UAVs having limited battery power and radio range according to GMU locations within the serving region. The GMU mobility that induces topology change in the UAV network is unfortunately unknown to the GCC, a command center for UAV navigation. Accordingly, the GCC must predict the GMU mobility based on past observations and relocate UAVs to sustain end-to-end connections of the GMUs in the multi-hop airborne network. Meanwhile, UAVs need to minimize energy consumption for the relocation.

There have been tremendous studies about multi-UAV systems until now. First, the survey on the airborne [6] introduced traditional issues such as multi-hop routing [7], handover, and energy efficiency like in conventional multi-hop ad hoc networks. Another survey [5] introduced previous works on UAV-mounted aerial base stations that can extend coverage and capacity of existing terrestrial wireless networks and pointed out several key challenges such as channel modeling [8], energy and flight time limitation, path planning, interference management, performance analysis, backhaul connectivity, and security. In [9], hierarchical airborne networks are introduced where different types of vehicles cooperate for networking at different altitudes such as Low Altitude Platform (LAP), High Altitude Platform (HAP), and Satellite layer. Also, a realization of the UAV-based airborne networking based on the 3GPP standard was discussed in [10].

In this study, we develop an online algorithm for UAV deployment in a partially observable environment, which aims at achieving robust backhaul connectivity of the FANET and energy saving. This deployment algorithm for UAV base stations has been extensively studied in the literature. Initially, multi-UAV deployment was studied to

* Corresponding author.
E-mail address: wooseong@gachon.ac.kr (W. Kim).

https://doi.org/10.1016/j.adhoc.2024.103463
Received 9 October 2023; Received in revised form 19 January 2024; Accepted 27 February 2024
Available online 2 March 2024

extend a wireless sensor network (WSN) for remote and static ground sensors [11,12]. The following studies focus on energy efficiency in the UAV control for sensory data collection. For instance, the works in [13,14] searched for optimal altitude to cover maximum ground area for energy saving. The works in [15,16] improved the previous works finding optimal placement in 3D space with various types of antennas. Recently, the concept of base stations on low altitude platforms (LAPs) attracted researchers' attention for emergency communication and the digital divide in under-developed areas. The works in [17,18] found the minimum number of UAVs for serving static ground users. The works in [19–21] investigated the multi-hop A2 A backhaul capacity with UAV mobility in the airborne network, where authors solve a joint optimization problem to obtain optimal trajectory, bandwidth, and transmission power for static ground users. In [22], single and dual multi-hop A2 A links were compared with respect to end-to-end throughput. The work in [23] solved another joint problem of energy minimization and throughput maximization in the same environment. As recent studies considering ground user mobility, the work in [24] suggested similar constraints with us such as A2G link capacity, multi-hop backhaul routing and ground user mobility that demands a new approach for the end-to-end wireless connectivity and network throughput. Also, the work in [25] proposed a multi-agent Deep Reinforcement Learning (DRL) algorithm for fast and reliable packet delivery in the multi-UAV relay network. However, these studies have no consideration for UAV energy saving and GMU observability in real environments.

To the best of our knowledge, this is the first study to solve a joint optimization problem of UAV energy saving and throughput maximization in multi-hop UAV networks with the prediction of ground user mobility. Furthermore, we have an additional constraint of partial observation on GMU mobility because tracing all GMUs using enough UAVs for the entire area is impractical. Therefore, we solve a problem modeled by the Partially Observable Markov Decision Process (POMDP) in this study.

Due to the large state and action space compared to limited sampling for online execution, we consider the simulation-based Monte Carlo Tree Search (MCTS) algorithm to find sub-optimal UAV deployment in a feasible time. In detail, our proposed solution mainly consists of the following three folds:

- First, we develop a simulator based on the Cell-based Probabilistic Trajectory Prediction (CPTP) model. In disaster environments without sufficient trajectory data for GMUs, the CPTP model can continuously improve performance by dynamically reconstructing the model whenever a new trajectory of GMUs is observed.
- Second, we extend the MCTS-based Partially Observable Monte Carlo Planning (POMCP) [26] for the GCC to find feasible UAV control in the partially observable environment. Additionally, we apply the Double Progressive Widening (DPW) to the MCTS-based POMCP to handle to deal with large state and action space.
- Third, we propose a Deep Neural Network (DNN)-based reinforcement learning (DRL) to guide the MTCS search, which narrows down the scope of action space and accelerates the search procedure. We train the actor–critic network using simulation samples for an online algorithm. To avoid overfitting caused by the simulation samples, we use the Proximal Policy Optimization (PPO) for the DNN training.

Our solution repeatedly performs the above three procedures updating models in the closed-loop. Based on the CPTP model, first the MCTS finds a best UAV deployment according to sampling guided by the DRL. Subsequently, the DRL is conducted with samples generated during the MCTS. From environments, the GCC obtains GMU observation information from UAVs after relocating UAVs based on the best action. This collected information is used for updating the CPTP model and the next procedures follow again.

We evaluate the proposed UAV deployment algorithm in two different network complexity with {20 GMUs, 20 UAVs} and {30 GMUs, 12 UAVs} in 2 square-kilometer serving area and 400 m UAV radio coverage. In the first case of low complexity, it achieves a 70% data rate against the demand rate of GMUs and 40% energy saving against the maximum energy consumption of UAVs, having 88 m average distance error for GMU location. In the second case of high complexity, the data rate is 47% and the energy saving is 40% with the 260 m distance error for GMUs. Also, we compare our learning performance with other MCTS variants in the experiment section.

The remainder of this paper is organized as follows. In Section 2, we explore related works that utilize the DRL for the control of UAVs within UAV networks. The background of this paper is elaborated in Section 3, covering topics such the POMCP, belief state update, and the PPO. We introduce the Multi-UAV airborne network with A2 A/A2G channel model, network flow model in FANET, and energy consumption model for propulsion and communication in Section 4. In Section 5, we propose the CPTP algorithm for GMU mobility prediction. We describe our UAV mobility control system and Monte Carlo Tree Search (MCTS) algorithm for the Partially Observable Markov Decision Process (POMDP) in Section 6 and present the experimental results in Section 7. Finally, we conclude this study in Section 8.

2. Related works

In recent literature, the DRL is popularly used to control the trajectories of UAVs in multi-UAV systems. Table 1 shows related works on controlling UAVs through DRL in UAV networks. The work in [27] optimized the trajectories of UAVs and access policies of GMUs to maximize the fair throughput of GMUs. As each GMU competes to maximize its throughput, the optimization problem is formulated as a cooperative–competitive game and solved with the Multi-agent DRL algorithm. The work in [28] proposed a Deep Deterministic Policy Gradient (DDPG) based algorithm to provide optimal radio coverage to points of interest (POI) for fairness and energy saving. Following works [29,30] solved the same problem in a distributed manner with Multi-Agent DDPG (MADDPG) and Soft Deep Recurrent Graph Network (SDRGN) respectively. These studies assumed single-hop UAV networks with stationary users. Meanwhile, there are studies that consider realistic situations by assuming the mobility of users. The work in [31] used the Genetic Algorithm-based K-means (GAK-means) and Q-learning algorithm to obtain cell partition information of mobile users and control the trajectory and transmission power of individual UAVs. The work in [32] applied an echo state network and a multi-agent Q-learning algorithm to predict the mobility of users and control the trajectory and transmission power of UAVs, where satellite helps in inter-UAV communication. Both studies aim only to maximize throughput. The works in [33,34] controlled the UAV's trajectory maximizing throughput and minimizing energy consumption. On the other hand, among studies considering multi-hop backhauling in UAV networks, the work in [35] used DRL for reliable backhauling in UAV networks with various UAV node failure events. The work in [36] optimized packet delivery ratio and energy consumption while considering environmental obstacles, and wind, where the optimization problem is solved with a multi-agent Asynchronous Advantage Actor Critic (A3C) network.

3. Background

3.1. Partially observable Monte-Carlo planning

The Monte-Carlo planning is an efficient approach for online planning in the large POMDP problem, in which a simulator provides sample sequences of state, observation, and reward on behalf of a generative MDP model. The Monte Carlo Tree Search (MCTS) algorithm conducts the best-first search at each state s_t as a tree node by selecting greedily an action a_t with the highest value; The MCTS evaluates

the action value function $Q(s,a)$ to perform approximately optimal action in each state s. In the Partially Observable Monte-Carlo Planning (POMCP) [26], the node includes the belief state $B(s,h)$ instead of the hidden state, s. Accordingly, the algorithm chooses an action based on the belief state. The belief state, $B(s,h)$ at time t is established by the history of past actions and observations $h_t = \{a_0, o_1, \dots, a_{t-1}, o_t\}$ or $h_t a_{t+1} = \{a_0, o_1, \dots, a_{t-1}, o_t, a_{t+1}\}$, which can be defined as the probability distribution for all possible states, $B_t(s,h) = \Pr(s_t = s | h_t = h)$. The belief state B_t is updated gradually from the previous belief state, B_{t-1} using previous action, a_{t-1} and current observation, o_t according to the Bayes' theorem as below:

$$B_t(s', h_t) = \frac{O(o_t \mid s', a_{t-1}) \sum_{s \in S} P(s' \mid s, a_{t-1}) B_{t-1}(s, h_{t-1})}{\Pr(o_t | a_{t-1}, B_{t-1}(s, h_{t-1}))} \quad (1)$$

where $\Pr(o_t|a_{t-1}, B_{t-1}(s,h_{t-1})) = \sum_{s' \in S} O(o_t \mid s', a_{t-1}) \sum_{s \in S} P(s' \mid s, a_{t-1}) B_{t-1}(s, h_{t-1})$.

Each belief node in the search tree contains a tuple of $\langle N(h), V(h), B(h) \rangle$; $N(h)$ is the number of visits to the history h, $V(h)$ is a value function of the history h and $B(h)$ is a set of particles for the history h. For the search tree expansion, a new child belief node is created by sampling randomly a state s from the current belief state, $s \sim B(s,h)$, and selecting the action that maximizes following Upper Confidence Bound 1 (UCB1).

$$UCB(ha) = V(ha) + c \cdot \sqrt{\frac{\log N(h)}{N(ha)}}, \quad (2)$$

where $N(ha)$ is the number of visits to the history ha and $N(h) = \sum_{a \in A} N(ha)$. Their ratio encourages the selection of unexplored actions in the search tree with the control variable c [26,37].

The action value function $V(ha)$ can be derived by the Bellman equation $V(ha) = \mathbb{E}\left[\sum_{i=0}^{k} \gamma^i r_{t+i} | h_t = h, a_t = a\right]$. For this, the simulator G generates sequences of a next state, next observation, and reward, $\{s_{t+1}, o_{t+1}, r_t\} \sim G(s,a)$ using random rollout policy, a_t.

3.2. Belief state update using particle filter

The popular weighted particle filtering is impracticable to derive particle weights in POMDP models with large state space. On the other hand, the unweighted particle filter approximates the belief state using the sample states obtained through simulation without knowledge of an exact POMDP model; those sample states correspond to particles, $B_t(h_t) = \cup_{i=1}^{M} \{s_t^i\}, s_t^i \in S$. Consequently, the belief state $B_t(s', h_t)$ is approximated by M particle as follows:

$$\hat{B}_t(s', h_t) \approx \frac{1}{M} \left[\sum_{i=1}^{M} \delta(s', s_t^i) | s_t^i \in B(h_t) \right], \quad (3)$$

where $\delta(\cdot,\cdot)$ is the Kronecker delta function. The probability of state s' is determined by the number of particles equal to s' in $B_t(h_t)$. More particles for history h_t can approximate the belief state $\hat{B}(s', h_t)$ closely as $\lim_{M \to \infty} \hat{B}_t(s', h_t) = B_t(s', h_t), \forall s' \in S$. Therefore, the unweighted particle filter can efficiently approximate the belief state in POMDP models with a large state space.

3.3. Proximal policy optimization

The PPO is a reliable and stable policy optimization method that prevents excessive updates and guarantees monotonic improvement. The PPO optimizes the "surrogate" objective function using stochastic gradient ascent as the Trust Region Policy Optimization (TRPO) [38].

$$J(\theta)^{TRPO} = \max_{\theta} \mathbb{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right], \quad (4)$$

where the advantage value for the mini-batch samples is estimated through a truncated version of Generalized Advantage Estimation (GAE) as follows,

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1}, \quad \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t), \quad (5)$$

Table 1
Related work on machine learning approaches for UAV control in UAV networks.

Paper	Users	UAV networks	Optimization
[25]	Static	Multi-hop	Throughput
[27]	Static	Single	Throughput, Fairness
[28–30]	Static	Single	Coverage, Fairness, Energy
[31,32]	Mobile	Single	Throughput
[33,34]	Mobile	Single	Throughput, Energy
[35]	Static	Multi-hop	Coverage
[36]	Static	Multi-hop	Throughput, Energy

where γ and λ are discount vectors for stable updates with reduced sample variance.

The TRPO maximizes Eq. (4) in a trust region constrained by the Kullback–Leibler (KL) divergence of $\pi_\theta(a_t|s_t)$ and $\pi_{\theta_{old}}(a_t|s_t)$, which eventually induces overfit avoidance and monotonic increment. Instead of the TRPO demanding heavy calculation for the constraint, the PPO uses a simple clipping method with a new objective as below,

$$J(\theta)^{PPO} = \mathbb{E}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right], \quad (6)$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$. The PPO clips the ratio through ϵ and selects the minimum value between the clipped and unclipped objective. Since the surrogate objective is maximized along the update process, the PPO can achieve comparable performance to the TRPO.

4. Multi-UAV ad-hoc system

For disaster management or urban surveillance, the multiple UAVs constitute an ad-hoc networked system to collect data from deployed ground sensors and support mobile users on the ground. When the mobile users cannot reach the Internet through near base stations that are powered off due to the disaster, the multi-UAVs can provide them the Internet connections to a gateway base station using multi-hop routing as shown in Fig. 1.

According to the deployment of ground mobile users (GMUs), the necessary number of UAVs can be varying to serve those GMUs and build a multi-hop air backbone. Some UAVs are used for linking even though there is no user in the surveillance area. The multi-UAV topology that dynamically changes according to user mobility is controlled by a ground control center (GCC). The GCC controls the location and trajectory of each UAV using ad-hoc networking under partially observable conditions dependent on GMU mobility.

4.1. UAV feature

UAVs are categorized into two types such as multi-rotor UAVs and fixed-wing UAVs. The fixed-wing UAVs normally have longer flight endurance capabilities with more payload, e.g., 40 vs. 12 min while multi-rotors (Raven X8) can provide stable hovering [39]. Therefore, the multi-rotor UAVs are appropriate for Low Altitude Platforms (LAP) as quasi-stationary aerial platforms in contrary to High Altitude Platforms (above 10 km). The average flight speed of the fixed wing is faster than the multi-rotors, e.g., 15 vs. 8 m/s. Both UAVs can be equipped with a communication module for the ground connection using Ku-band, UHF-band, or cellular frequency band, and an 802.11 module for inter-UAV communication. In this study, UAVs are supposed to have mostly no connection to the ground station during flight due to the limitation of radio coverage.

4.2. UAV mobility and service region

According to [40], UAVs have several mobility patterns such as random way-point, stay-at, and oval movement for hovering. In this study, we consider 3 movement phases: (i) move straight toward a target area for user support by way-point mobility, (ii) stay at the destination

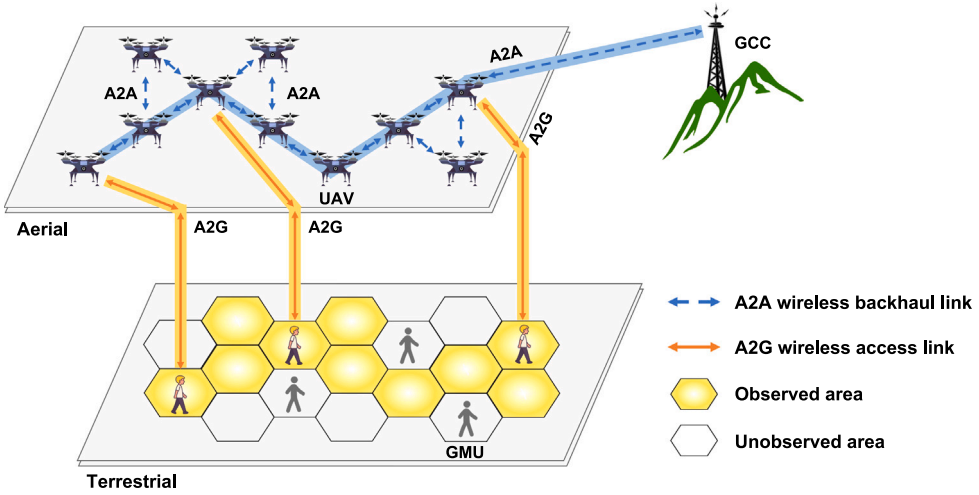


Fig. 1. Multi-hop UAV ad-hoc network for LAP.

by moving circularly or hovering to provide wireless connectivity to ground terminals, and (iii) predict ground user mobility and move waypoint to a next area or switch to an idle state to save energy. The idle state is possible in places such as charging stations or building rooftops that are assumed available over the search area in this study.

All UAVs fly horizontally at minimum altitude for energy efficiency except time for landing and take-off; UAVs consume much energy for propulsion when increasing the altitude. Low altitude reduces a projected area as a serving region with narrow Angle of View (AoV), but is advantageous to provide a higher data rate. Additionally, the wind speed varies propelling energy according to wind direction. In this study, we consider the constant wind speed and flight speed for simplicity. However, we consider energy variation by varying air density according to altitude.

The UAV velocity is varying with its application from several ten meters per second (mps) to several hundred mps. Especially, some military or critical mission-purposed UAVs can fly by more than several hundred mps for several hours or days based on gas, fuel cells, and self-charging. But in this study, we consider inexpensive commercial UAVs for civil platforms that can fly less than 20 mps.

In this study, the serving area is sectorized into sub-regions like the Manhattan grid, where the size of the sub-region is determined according to the search region of the UAV. The individual flight trajectory of each UAV is given by the GCC which predicts GMU locations based on previous partial observations by the limited number of UAVs.

4.3. UAV wireless communications

4.3.1. UAV to GMU communications

There have been many simulation and measurement studies on the air-to-ground (A2G) channel model [41]. According to the geo-locations, the reflection and diffraction along ray proration can vary. In [42], a statistical propagation model was developed using a ray tracing simulator to predict the A2G path loss from a LAP UAV to a terrestrial device in urban environments, where virtual-city environments like Manhattan are simulated according to ITU-R statistical parameters at 700 MHz, 2000 MHz and 5800 MHz, with elevation angles above 15 degrees.

In this study, ultra-low altitude A2G communications are assumed for urban environments where various structures and buildings are deployed to provide probabilistically LOS or Non-LOS (NLOS) from UAVs. The statistic model of the LOS between the UAVs and GMUs is modeled as a logistic function with elevation angle in [13] as below,

$$p_{i,k}^{LOS}(n) = \frac{1}{1 + c_1 \exp(-c_2 \theta_k(n) + c_1 c_2)}, \tag{7}$$

where c_1 and c_2 are propagation parameters and $\theta_k(n) = \frac{180}{\pi}\sin^{-1}(H/d_{i,k}(n))$ is elevation angle.

The expected channel gain with the randomness of LOS link between an UAV i and GMU k at a discrete n time slot can be [13,42],

$$\mathbb{E}[|h_{i,k}(n)|^2] = p_{i,k}^{LOS}(n)\beta_k(n) + (1 - p_{i,k}^{LOS}(n))\kappa\beta_k(n), \tag{8}$$

where the channel coefficient is $h_{i,k}(n) = \sqrt{\beta_k(n)}\,\tilde{h}_k(n)$ with small-scale fading, $\mathbb{E}[|\tilde{h_{i,k}}(n)|^2] = 1$. The $\beta_k(n)$ is pathloss $\beta_0 d_{i,k}^{-\alpha}(n)$ and addition attenuation, $\kappa < 1$ exists for NLOS.

Consequently, the achievable data rate can be

$$r_{ik} = B_i \log_2\left(1 + \frac{|h_{i,k}(n)|^2 p_i^{tx}}{\sigma^2}\right) \tag{9}$$

with transmission power p_i^{tx} and noise σ^2.

4.3.2. Inter-UAV communications

The UAVs are supposed to communicate with each other using WLAN technologies such as IEEE 802.11a/g/n within several hundred meters. The air-to-air (A2 A) communication link is determined by a particularly dominant line-of-sight (LOS) link, while ground reflection is varying with altitude. The A2 A communication links can be characterized by the Rice model that reflects strength of dominant LOS (ρ) and non-dominant NLOS paths (σ) and its probability density function is as below.

$$f(x|\rho,\sigma) = \frac{x}{\sigma^2} exp(\frac{-x^2-\rho^2}{2\sigma^2}) I_0(\frac{x\rho}{\sigma^2}) \tag{10}$$

The Rician fading is shaped by the factor $\mathrm{K}(\rho_0,\sigma) = \frac{\rho_0^2}{2\sigma^2}$ which is varying with the UAV altitude, h and can be approximated by a function of the height, $\sigma = ah^b + c$, $a = 212.3, b = -2.221, c = 1.289$ and $\rho_0 = 6.469$ [43]. Low altitude that has no dominant path ($\rho = 0$) follows Rayleigh fading.

The Rician fading channel, $h_{i,j}(n)$ between a node i and j is

$$h_{i,j}(n) = \sqrt{\beta_{i,j}(n)}\left[\frac{h_{i,j}^{NLOS}(n) + \sqrt{K}h_{i,j}^{LOS}(n)}{\sqrt{1+K}}\right], \tag{11}$$

where $h_{i,j}^{NLOS}(n)$ is the zero-mean complex Gaussian process for scattering/diffuse, $\sim \mathcal{CN}(0,1)$ and $h_{i,j}^{LOS}(n)$ is the specular LOS component as deterministic complex exponential, $e^{j2\pi f_d \cos\theta t + \phi}$ with angle of arrival (AoA) θ and phase ϕ, and $|h_{i,j}^{LOS}(n)| = 1$.

4.3.3. Multi-hop routing in UAV airborne network

The backhaul ad-hoc network with multi-UAVs needs a multi-hop routing protocol to provide end-to-end wireless connectivity to the mobile users on the ground. Many routing protocols for Flying ad hoc

networks (FANET) have been introduced, which are based on network topology, geolocation information or both [7].

The multi-hop airborne network has routing constraints such as the flow conservation rule in Eq. (12) and the single path rule in Eq. (13) as below.

$$\sum_{j\in\mathcal{N}} f_{ij}^{u} - \sum_{j\in\mathcal{N}} f_{ji}^{u} = \begin{cases} R^{u}, & \text{if } i = \text{ source} \\ -R^{u}, & \text{if } i = \text{ sink} \\ 0 & \text{otherwise} \end{cases} \quad \forall u \in \mathcal{U}, \forall i \in \mathcal{N}. \tag{12}$$

where f_{ij}^{u} is a GMU data flow on the A2 A link i, j, and R^{u} represents corresponding demand rate of the GMU u.

$$\sum_{j\in\mathcal{N}} x_{ij}^{u} - \sum_{j\in\mathcal{N}} x_{ji}^{u} = \begin{cases} 1, & \text{if } i = \text{ source} \\ -1, & \text{if } i = \text{ sink} \\ 0 & \text{otherwise} \end{cases} \quad \forall u \in \mathcal{U}, \forall i \in \mathcal{N} \tag{13}$$

A matrix with $x_{ij}^{u} = \{0, 1\}$ indicates routing information of a GMU data flow, f^{u}. In this study, the routing information is given by the GCC which calculates the shortest paths from each serving UAV to the gateway, i.e., GCC using the Dijkstra algorithm in a weighted graph. Practically, the GCC broadcasts the routing information over the UAV airborne networks. For this, individual UAV positions and trajectories are controlled to provide robust A2 A wireless connectivity.

As link capacity constraint of A2 A communication, aggregated user flows on a A2 A link must be less than the maximum link capacity r_{ij}^{max}

$$\sum_{u\in\mathcal{U}} f_{ij}^{u} \le r_{ij}^{max}, \quad \forall i, j \in \mathcal{N} \tag{14}$$

For the A2G capacity constraint, the total number of necessary sub-channels for GMU data traffic should not exceed the maximum sub-channels of a given A2G system bandwidth.

$$\sum_{u\in\mathcal{U}} \lceil \frac{f_{iu}^{u}}{r_{iu}} \rceil \le |\mathcal{B}_{i}^{ch}|, \quad \forall i \in \mathcal{N}, \tag{15}$$

where r_{iu} is the maximum capacity of a particular A2G link that is varying with individual A2G channel and $\mathcal{B}_{i}^{ch}$ is a set of sub-channels of an UAV i.

In this study, we assume that all GMUs demand the same data rate and sub-channels are evenly assigned to the GMUs regardless of individual channel gain. Efficient resource scheduling for different demands and channels will be considered in future study.

4.4. Energy consumption

Energy consumption occurs by UAV movement and wireless communication; energy consumption from flight is mostly about 300 W while Wi-Fi and GPS only consume several ten mW [44]. As UAVs receive instruction from the GCC for topology control, they consume propulsion energy to move to the next location. Subsequently, they spend energy on multi-hop wireless communications to serve user traffic.

4.4.1. Propulsion energy consumption

The propulsion energy consumption of fixed-wing UAVs is simply modeled in [45,46] for consistent level flight as a serving base station.

Meanwhile, rotary-wing UAVs have different energy consumption models for hovering and forward flight, which are described in [47] respectively. For the power consumption of hovering,

$$P_h = \underbrace{\frac{\delta}{8}\rho s A \Omega^3 R^3}_{P_0} + \underbrace{(1+k)\frac{W^{1/2}}{\sqrt{2\rho A}}}_{P_i} \tag{16}$$

where parameters and simulation values are shown in Table 2 (refer to [47] for details).

Table 2
UAV propulsion energy model parameters.

Parameters	Meaning	Value	Unit
δ	Profile drag coefficient	0.012	–
ρ	Air density	1.225	kg/m^3
s	Rotor solidity	0.05	–
A	Rotor disc area	0.503	m^2
Ω	Blade angular velocity	300	radians/second
R	Rotor radius	0.4	m
W	UAV weight	20	N
d_0	Fuselage drag ratio	0.6	–
v_0	Mean rotor induced velocity in hover	4.03	m/s
k	Correction factor to induced power	0.1	–

The power consumption model of the forward flight is

$$P_f(v) = P_0\left(1 + \frac{3v^2}{\Omega^2 R^2}\right) + P_i\left(\sqrt{1 + \frac{v^4}{4v_0^4}} - \frac{v^2}{2v_0^2}\right)^{1/2} + \frac{1}{2} d_0 \rho s A v^3 \tag{17}$$

where v is forward speed, v_0 is hovering velocity and d_0 is fuselage drag ratio.

Consequently, propulsion energy consumption can be

$$E_i^{p}(t) = (P_h + P_f(v_i(t))) \cdot t, \tag{18}$$

where individual UAV velocity, $v_i(t)$ is determined by trajectory control.

4.4.2. Communication energy consumption

For power consumption for communication, UAVs use transmission power according to data rate. For instance, Soekris device consumes measured energy like 1.58 and 1.14 W as P_{tx} and P_{rx}, respectively for 48 Mbps data rate with MCS 64-QAM 2/3 [48]. Therefore, energy consumption on A2 A and A2G links is varying with user traffic on them.

Power consumption of an inter-UAV link ij for the A2 A communications consists of static power consumption of an active transceiver, $P_{0_i} = 3.5$ W and dynamic power consumption loaded by the amount of user data rate,

$$P_{ij}^{a2a} = P_{0_i} + P_{max}^{a2a} \cdot \frac{\sum_{u\in\mathcal{U}} f_{ij}^{u}}{r_{ij}^{max}} \tag{19}$$

, where dynamic power consumption of a A2 A link, ij can be derived by proportional transmission power to maximum value P_{max}^{a2a}, which is traffic load on the A2 A link, i.e., aggregated user data rate, $\sum_u f_{ij}^{u}$ over achievable data rate r_{ij}^{max}.

The total power consumption of each UAV i for A2 A links is

$$P_i^{a2a} = \sum_{j\in\mathcal{N}} P_{ij}^{a2a} \tag{20}$$

For A2G communication, an UAV as an air base station provides sub-channels of OFDMA to mobile users. Similar to the A2 A communication, the power consumption of the A2G link can be approximated by the user data rate. Power consumption of the A2G communication is formulated as below,

$$P_i^{a2g} = P_{0_i} + \Delta_p \cdot \frac{P_{max}^{a2g}}{|U_i|} \sum_{u\in\mathcal{U}_i} \min\left(\frac{f_{iu}^{u}}{r_{iu}}, 1\right), \quad \forall i \in \mathcal{N} \tag{21}$$

, where static power consumption P_{0_i} is 6.8 W and Δ_p is 4 as a multiplier for load-dependent power consumption, which is comparable with pico-cell power consumption [49]. Similarly, the transmission power on each A2G link iu can be scaled by user data rate f_{iu} against link capacity, r_{iu} with bandwidth B_{iu} assigned for the GMU u; the access link bandwidth is evenly assigned to the individual user in this study.

The communication energy consumption of each UAV is

$$E_i^{c}(t) = (P_i^{a2g} + P_i^{a2a}) \cdot t \tag{22}$$

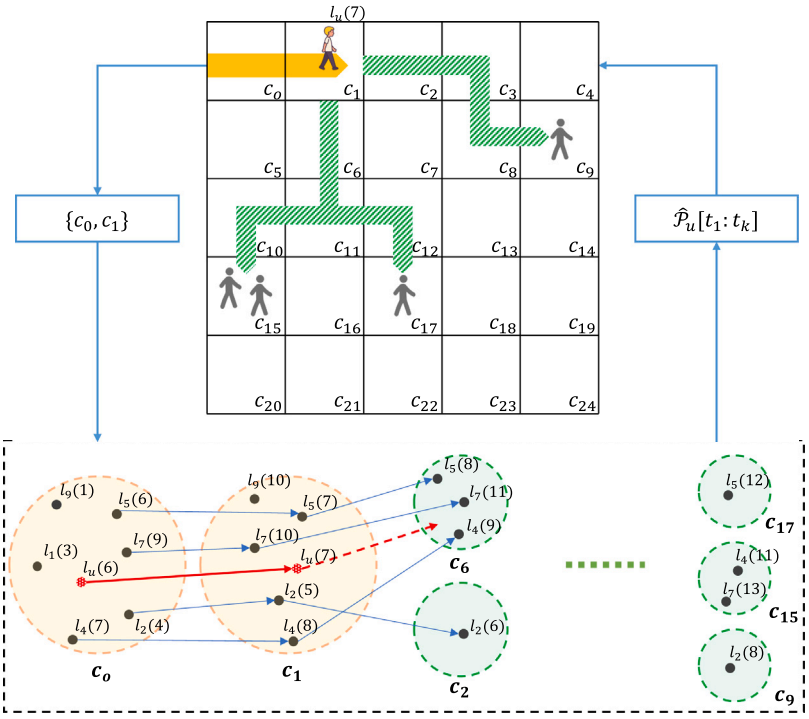


Fig. 2. Trajectory prediction model.

Therefore, the total energy consumption of all operating UAVs in the airborne network from t_n until t_{n+1} is

$$\sum_{i\in\mathcal{N}} \int_{t_n}^{t_{n+1}} E_i^p(t, v_i(t)) + E_i^c(t)dt, \tag{23}$$

5. GMU mobility prediction

For GMU mobility prediction, we propose a Cell-based Probabilistic Trajectory Prediction (CPTP) in a grid topology.

5.1. Cell-based trajectory management

Our path prediction mechanism probabilistically finds a future path, i.e., a sequence of cells in a Manhattan grid based on past trajectory statistics in dynamic environments. Therefore, the entire service region is divided into grid cells, $c_i, i \in TG$ of a trajectory grid (TG), considering radio coverage of A2G communication.

The GMU trajectory prediction is achieved by the following two-phase procedure: (i) trajectory information update, (ii) trajectory prediction. In the update procedure, trajectory information of each GMU such as a GMU identifier, previous τ-cells along a trajectory, a next cell, and update time is recorded in the current cell. Also, each cell counts the number of visiting GMUs so far and traces their transitions to the next cells, which are used to derive a prior probability in Eq. (28) and (30).

In the prediction procedure, a GMU trajectory during the last τ discrete time is used to predict future mobility. The τ-backward trajectory of GMU u is denoted by $\mathcal{P}_u[t_{n-\tau} : t_n] = \{l_u(t_{n-\tau+1}), l_u(t_{n-\tau+2}), \dots, l_u(t_n)\}$ of $\mathcal{P}_u[-\infty : n]$ at the current discrete time t_n where the GMU location $l_u(t)$ indicates a cell, c_i at time t. Our algorithm finds a set of reference GMUs (RGMU or RU) that have the same sequence in their trajectories, i.e., a τ-length sequence of locations l as in Eq. (24).

$$\begin{aligned} &\mathcal{P}_u[t_{n-\tau} : t_n] = \mathcal{P}_i[t_{m-\tau} : t_m], \ \forall i \in U \setminus u, \\ &\qquad |\mathcal{P}_i[t_{m+1} : \infty]| > 0, \quad t_n, t_m \in T \end{aligned} \tag{24}$$

where n, m are a particular time in the discrete time set T and the trajectory length $|\mathcal{P}_i|$ of an RGMU from the m needs to be long enough to provide information for the prediction of u trajectory.

In the example of Fig. 2, suppose that the GMU u is now at c_1 at current time $t_n = 7$ after passing through the cell, c_0 and c_1 during last two steps, the $\tau = 2$ backward trajectory of the GMU u is $\mathcal{P}_u[6 : 7] = \{c_0, c_1\}$ and a set of RGMUs can be $\{2, 4, 5, 7\}$ who passed through same cells c_0 and c_1 previously according to Eq. (24). Longer τ can increase the similarity of the trajectory of RGMU with the GMU u and improve prediction accuracy. On the other hand, such a long τ reduces the set size of the available RGMUs for prediction when the trajectory data are scarce due to the small number of GMUs. Accordingly, it is challenging to decide optimal τ according to environments with varying numbers of GMUs.

5.2. Probabilistic path prediction

Future trajectory of a GMU u from time t_{n+1} to t_{n+k} is denoted as $\hat{\mathcal{P}}_u[t_{n+1} : t_{n+k}]$ at the current time t_n, e.g., $\hat{\mathcal{P}}_u[t_{n+1} : t_{n+k}] = \{c_2, c_8, \dots, c_{13}\}$.

For the GMU u trajectory prediction, we first align the different current times t_n, t_m of the GMU u, and RGMUs by t_0. Future k locations of the GMU u from t_1 are determined by the following equation,

$$\Pr(\hat{\mathcal{P}}_u[t_1 : t_k] \mid \mathcal{P}_{RU}[t_1 : t_k]) \tag{25}$$

where the group trajectory of RGMUs is $\mathcal{P}_{RU}[t_1 : t_k] = \bigcup_{i=t_1}^{t_k} l_{RU}(i)$, $l_{RU}(i) = \{l_u(i) | u \in RGMU, i \in T\}$, from initial locations of the RGMUs, $l_{RU}(t_1)$ to kth locations $l_{RU}(t_k)$.

According to the individual trajectory, the RGMU group trajectory can be split like in Fig. 2, where the RGMUs split into the cell c_2 and c_6. We decide a next cell of the GMU u using grid-based prediction filtering that outputs the probabilistic distribution for the cells $\Pr(c^k | \mathcal{P}_{RU}[t_1 : t_k])$ with the input set of RGMU trajectory, $\mathcal{P}_{RU}[t_1 : t_k]$ at each time step.

The probability distribution of GMU u's path, η^u for the given RGMU trajectory at time t_{k-1} is,

$$\eta^u(t_{k-1}) = \Pr(\hat{\mathcal{P}}_u[t_1 : t_{k-1}] \mid \mathcal{P}_{RU}[t_1 : t_{k-1}]), \tag{26}$$

When the probability that GMU u visits a cell j is $\eta_j^u(t_{k-1})$, the probability to move toward the next cell i from the current cell j is updated by Bayesian inference in Eq. (27).

$$\eta_i^u(t_k) = \eta_j^u(t_{k-1}) \Pr(l_u(t_k) = c_i | l_u(t_{k-1}) = c_j) \cdot \Pr(l_{RU}(t_k) | l_u(t_k) = c_i), \tag{27}$$

where the $\Pr(l_u(t_k) = c_i | l_u(t_{k-1}) = c_j)$ is the state transition probability from the cell c_j at time t_{k-1} to cell c_i at time t_k as described in Eq. (28). The $\Pr(l_{RU}(t_k) | l_u(t_k) = c_i)$ is a likelihood function, where $l_{RU}(t_k)$ is cell distribution of RGMUs and $l_u(t_k) = c_i$ is the event the GMU u moves to the cell i.

The state transition probability is determined by the number of cell transitions from c_j to c_i.

$$\Pr(l_u(t_k) = c_i | l_u(t_{k-1}) = c_j) = \frac{v_{c_i|c_j}}{\sum_{n \in l_{RU}(t_k)} v_{c_n|c_j}}, \tag{28}$$

where $v_{c_i|c_j}$ denotes the number of moves from c_j to c_i.

Algorithm 1: Cell-based probabilistic trajectory prediction

```
1  Function PredictTraj(u, τ, k):
2      Find reference GMUs from Eq. (24) by τ
3      Find future trajectory of RGMU in TG, P_RU[t_1 : t_k]
4      η^u_{1:k} = { }
5      for t ∈ t_1 : t_k do
6          η^u_t = { }
7          for i ∈ l_RU(t) do
8              η^u_i(t) calculated from Eq. (25)–(27)
9              η^u_t = η^u_t ∪ {η^u_i(t)}
10         end for
11         η^u_{1:k} = η^u_{1:k} ∪ {η^u_t}
12     end for
13     return η^u_{1:k}
14 End Function
```

The likelihood function that the probability that $l_{RU}(t_k)$ will be observed when the GMU u is in cell c_i at time t_k can be derived by Bayes theorem as in the following equation.

$$\Pr(l_{RU}(t_k) | l_u(t_k) = c_i) = \frac{\Pr(l_u(t_k) = c_i | l_{RU}(t_k)) \Pr(l_{RU}(t_k))}{\Pr(l_u(t_k) = c_i)} \tag{29}$$

where $\Pr(l_u(t_k) = c_i)$ is the prior probability that the u will be located at $l_u(t_k) = c_i$ and $\Pr(l_u(t_k) = c_i | l_{RU}(t_k))$ is the probability having a GMU u in $l_u(t_k) = c_i$ when $l_{RU}(t_k)$ is observed, which are derived by following Eq. (30) and (31), respectively.

$$\Pr(l_u(t_k) = c_i) = \frac{v_{c_i}}{\sum_{j \in l_{RU}(t_k)} v_{c_j}}, \tag{30}$$

where v_{c_i} is the number of visits to a cell c_i by GMUs.

$$\Pr(l_u(t_k) = c_i | l_{RU}(t_k)) = \frac{|RU_{c_i}(t_k)|}{|RU_{c_j}(t_{k-1})|}, \tag{31}$$

where $RU_{c_i}(t_k) = \{u | l_u(t_k) = c_i, u \in RGMU\}$. Accordingly, $\Pr(l_u(t_k) = c_i | l_{RU}(t_k))$ is proportional to number of overlapping RGMUs between two cells. In Fig. 2, for instance, the probability $\Pr(c_2(1) | c_1(0))$ and $\Pr(c_6(1) | c_1(0))$ are 1/4 and 3/4, respectively.

Algorithm 1 describes the trajectory prediction algorithm. GMU visiting probabilities for cells of $l_{RU}(t)$ are calculated by Dynamic program according to Eq. (27)–(31) to predict the future path of the GMU u (line 7–10), which is repeated k times. Thereby the future trajectory of the GMU u until t_k, $\hat{\mathcal{P}}_u[t_1 : t_k]$ is sampled by $\eta^u(t_1) \sim \eta^u(t_k)$, respectively.

6. UAV mobility control in air borne network

In this section, we propose a solution for multi-UAVs trajectory control under uncertain location information of the GMUs. We model the problem as the Partially Observable Markov Decision Process (POMDP) capable of sequential decision making under the state of uncertainty.

6.1. POMDP model

In the disaster and mission-critical scenario, the GMU behavior and mobility are often invisible to the hovering UAVs due to complicated ground structures and broad service areas. Accordingly, the POMDP model can be considered to allow the GCC to decide UAV actions under uncertainty of the true environment state in contrast to the MDP model that requires the full state information. The POMDP model is defined by a tuple $(S, A, O, P, \Omega, R, \gamma)$ where each parameter is described below.

- S : The state is a combination of location information of UAVs and GMUs; the UAV state is UAV deployment over the grid and the GMU state is number of GMUs staying at each cell of the grid network that has consistent state space regardless of the number of GMUs. Therefore, the vector size of the S is the sum of the number of UAVs and the number of the trajectory grid cells, $|\mathcal{N}| + |TG|$. The state is defined as:

$$s_t = [l_{\mathcal{N}}(t), \zeta_{TG}(t)], \tag{32}$$

  where the states of UAVs is a vector, $l_{\mathcal{N}}(t) = [l_1(t), l_2(t), \ldots, l_{|\mathcal{N}|}(t)]$ and states of GMUs is presented by $\zeta_{TG}(t) = [\zeta_1(t), \zeta_2(t), \ldots, \zeta_{|TG|}(t)]$, where $\zeta_i(t) = \sum_{u \in \mathcal{U}} \zeta_{c_i}^u(t)$ and $\zeta_{c_i}^u(t)$ is

$$\zeta_{c_i}^u(t) = \begin{cases} 1, & \text{if } l_u(t) = c_i \\ 0, & \text{otherwise,} \end{cases} \tag{33}$$

- A : The action is an instruction for a new deployment of UAVs over the grid. We configure the re-allocation time period considering the maximum speed of the UAV over the longest flight distance, i.e., diagonal distance in the grid. Therefore, all UAVs can be relocated to any cell regardless of their current location.

$$a_t = [l_1(t), l_2(t), \ldots, l_{|\mathcal{N}|}(t)] \tag{34}$$

  Considering the limited energy of the UAV, in this study we minimize energy consumption by activating only some number of UAVs as serving cells while deactivating other remaining UAVs. According to GMU deployment and traffic, however, A2 A and A2G communication link capacity are probably not enough for the user demands as in Eq. (14)–(15). In future work, we will study locating multiple UAVs at a same cell to enlarge the access and backhaul link capacity.
- Ω : A set of observations is composed of $o_t = [\hat{\zeta}_1(t), \hat{\zeta}_2(t), \ldots, \hat{\zeta}_{|TG|}(t)]$ as distribution information of GMUs.

$$\hat{\zeta}_i(t) = \begin{cases} \sum_{u \in \hat{\mathcal{U}}} \zeta_{c_i}^u(t), & \text{if } i \in l_{\mathcal{N}} \\ 0, & \text{otherwise,} \end{cases} \tag{35}$$

  The GMUs are partially observed, $\hat{\mathcal{U}}$ according to UAV deployment, $l_{\mathcal{N}}$ and A2G communication links; in our scenario, there are a limited number of UAVs compared to the broad service area. The GMU distribution information is collected by serving UAVs through A2G communication. In detail, an UAV periodically broadcasts cell information, after which GMUs within the same cell report their presence to the serving UAV.
- $P(\cdot | s, a)$: The state transition probability toward the next state at the given state s and the action a is uncertain due to the difficulty to obtain sufficient statistics in the partial observable environment. Furthermore, the curse of dimensionality, $\boldsymbol{S} \times \boldsymbol{A} \times \boldsymbol{S}$ for the probability distribution demands sampling state transitions using the simulator rather than considering all possible transitions [silver-pomcp].

- $O(\cdot|s,a)$: The observation probability, $\Pr(o_{t+1}=o|s_{t+1}=s,a_t=a)$ is determined by the observation of GMUs, $\hat{\zeta}_{TG}$ at the given state s and the action a. Because the observation is limited in the large state space, we also sample the observations using the simulator like the state transition probability.
- $R(s,a)$: The reward r_t consists of the data rate of GMUs and energy consumption of UAVs, where the data rate of each GMU is calculated according to A2X link capacity Eq. (9) and network flow constraints of Eq. (12)–(15), and energy consumption of UAVs is defined in Eq. (23). The objective of this study is to find the best UAV control policy to maximize the sum of the data rate of GMUs while minimizing the energy consumption of UAVs.

$$r_t = \frac{\omega_1}{|\mathcal{U}|}\cdot\sum_{u\in\mathcal{U}}\frac{r^u}{R^u} - \frac{\omega_2}{|\mathcal{N}|}\cdot\sum_{i\in\mathcal{N}}\frac{\int_{t_n}^{t_{n+1}} E_i^p(t,v_i(t))+E_i^c(t)dt}{E_i^{\max}}, \tag{36}$$

  where ω_1 and ω_2 are weights for the data rate and energy consumption, respectively. For normalization, GMU rate r_u is divided by the individual demand rate, R_u, and also energy consumption of each UAV is divided by the maximum energy consumption, $E_i^{\max}$, which is the sum of the maximum propulsion and communication energy consumption. The maximum propulsion energy is consumed energy for flight to the possible farthest cell from the current location and the maximum communication energy is transmission energy with maximum data rate.
- γ is the discount factor for reward, [0,1)

We have the belief state $B(s,h)$ instead of hidden state s, where the belief state is a probability distribution for all possible states. Updating the belief state by Eq. (1) is unsuitable for our proposed system that has too large state space to derive all state and observation transition probabilities which require excessive computational cost [26]. As an approximate solution, we consider the unweighted particle filtering [26] to update the belief state by Eq. (3).

6.2. Efficient MCTS algorithm in large POMDP

We adopt the Monte Carlo Tree Search (MCTS) algorithm that constructs a forward search tree for the best-first search using the Monte Carlo Simulation and evaluates the action value function $Q(s,a)$ to perform approximately optimal action in each state s. Each belief node in the search tree contains a tuple of $\langle N(h), V(h), B(h)\rangle$; $N(h)$ is the number of visits to the history h, $V(h)$ is a value function of the history h and $B(h)$ is a set of particles for the history h.

6.2.1. Double progressive widening based Monte Carlo tree search

Our system has a large action and observation space for UAV deployment in the Manhattan grid, which leads to form a wide and flat search tree. Under the large action space, the MCTS algorithm selects rarely the same action for a given history h because there are too many unexplored actions with infinite UCB1 value as $N(ha)\to 0$ in Eq. (2). Also, the large observation space proportional to the state space limits the probability that the simulation $G(s,a)$ revisits the same next state s' and observation o'. Accordingly, it is difficult for the search tree to grow more than a single layer since every simulation step mostly generates a new child belief node from the root.

With the limited and uniformly distributed particle samples for each history h, $B(hao')$, it is critical to approximate the belief state through the unweighted particle filtering.

To tackle this problem without excessive sampling, we adopt the Double Progressive Widening (DPW) as an improved UCB1 version [50] that gradually increases the number of child nodes considering the number of node visits $N(h)$ along the search progress. In detail, a child node for action or observation is created only when the number of child nodes is less than $\kappa_a N(h)^{\alpha_a}$ and $\kappa_o N(ha)^{\alpha_o}$ respectively as shown in Algorithm 2 where the variable κ and α are used to control node generation. Accordingly, the DPW allows the MCTS algorithm to shape the search tree with limited childs and to search deeply for a particular history h.

Algorithm 2: Monte Carlo Tree Search Guided by DPW

```
1  Function Simulate(s, h, η^U(t), k, θ):
2      if k < t in η^U(t) then
3          return 0
4      end if
5      a = SelectAction(s, h, θ)
6      if |C(ha)| ≤ κ_o N(ha)^α_o then
7          s', o, r ← G(s, a, η^U(t))
8          C(ha) = C(ha) ∪ {o}
9          B(hao) = B(hao) ∪ {s'}
10         if |B(hao)| == 1 then
11             q = r + γ · V_θ(s')
12         else
13             q = r + γSimulate(s', hao, η^U(t + 1), k, θ)
14         end if
15     else
16         o ← random sample C(ha)
17         s' ← random sample B(hao)
18         r = R(s, a, s')
19         q = r + γSimulate(s', hao, η^U(t + 1), k, θ)
20     end if
21     N(h) = N(h) + 1
22     N(ha) = N(ha) + 1
23     V(ha) = V(ha) + (q − V(ha)) / N(ha)
24     return q
25 End Function
26 Function SelectAction(s, h, θ):
27     if |C(h)| ≤ κ_a N(h)^α_a then
28         a ← π_θ(a|s)
29         C(h) = C(h) ∪ {a}
30     end if
31     return arg max_{a∈C(h)} V(ha) + c sqrt(log N(h) / N(ha))
32 End Function
```

6.2.2. MCTS simulation based on DPW

In this section, we explain the detailed procedure of our MCTS algorithm shown in Algorithm 2.

In `Simulate` function, the MCTS procedure ends when the trajectory prediction step t of a particular GMU u exceeds the maximum prediction length k (line 2–4), where $\eta^{\mathcal{U}}(t)=\cup_{u=1}^{\mathcal{U}}\{\eta^u(t)\}$ that is used to sample the cell distribution of GMUs, ζ_{TG}.

In other words, the simulation stops if a particular GMU is unobserved for more than k steps. Initially, the trajectory of each GMU is predicted by the CPTP model for future k steps. In the next step, some of GMUs can be observed and simulated again for the next k length trajectory prediction using the updated model by the recent observation as in $t=0$. Meanwhile, others continue using the remaining k-1 length trajectory as in $t=1$.

For selecting an action a, a new action node is created by the DPW algorithm in the `SelectAction` (line 27) and is appended to $C(h)$ (line 28–29) where $C(h)$ is a set of actions for history h and an action a is given by the policy function $\pi_\theta(a|s)$. The `SelectAction` function finally selects the action that maximizes UCB1 from $C(h)$ for history h (line 31).

With the action a in the `Simulate` function (line 5), a new child belief node is created similarly by the DPW algorithm (line 6–14) only if the number of child belief nodes for history ha is less than $\kappa_o N(h,a)^{\alpha_o}$ (line 6). Otherwise, the next state and the next observation are sampled randomly from the sets $B(hao)$ and $C(ha)$, respectively (line 16–17) until enough particles are collected to approximate the belief state $B(s,hao)$ for a new history hao.

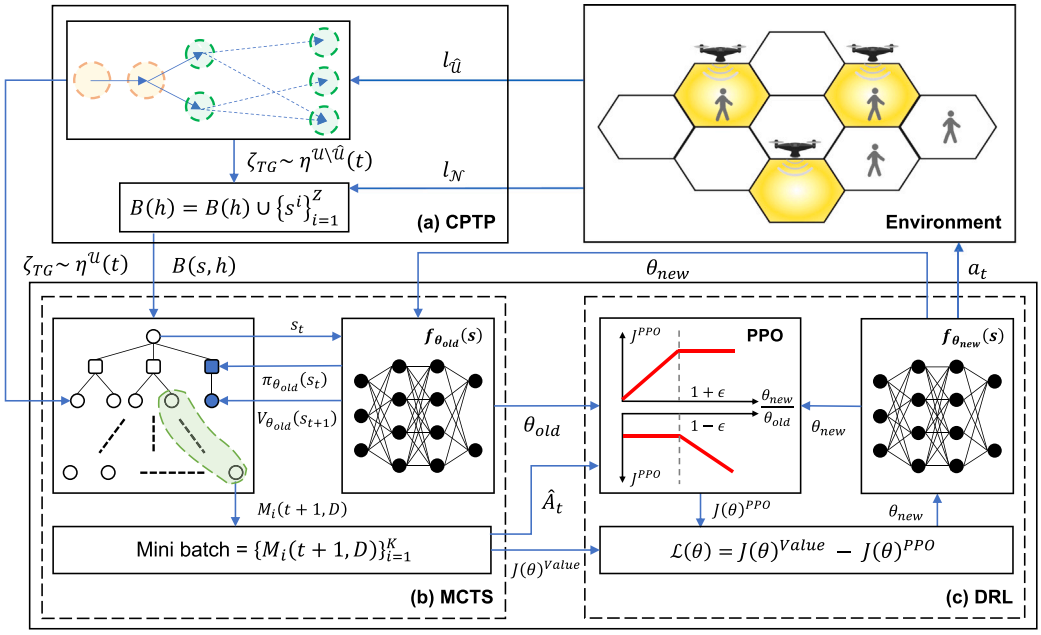


Fig. 3. System architecture.

Subsequently, the simulator G produces a new next state, observation, and reward for the given action based on the CPTP (line 7). If there is already a child node corresponding to the new history hao, the simulation continues recursively at the child node after adding 1 to $\eta^{\mathcal{U}}(t)$ (line 13). Otherwise, a new leaf node is created with a value estimated by the state value function $V_\theta(s)$ (line 11). Finally, the MCTS algorithm updates the action value function $V(ha)$ from the leaf node to the root node by backpropagation (line 23).

6.2.3. MCTS guided by deep reinforcement learning

During the tree expansion by the above simulation, a child node for a new history hao is typically evaluated by discounted returns from the rollout policy. However, the performance of the planning by the random sampling is degraded generally in the environment with a large state and action space [37,51].

To reduce the search range, we adopt the Deep Reinforcement Learning (DRL) algorithm for the MCTS. In `SelectAction` of Algorithm 2, accordingly the action set corresponding to child nodes, $C(h)$ explored by the policy function $\pi_\theta(a|s)$ can be effectively limited for the search area reduction. Also, the leaf node can be evaluated directly by the $V_\theta(s')$ in the `Simulate` function (line 11) instead of simulating k steps with the rollout policy.

For the online algorithm, we use the samples obtained from the simulation for learning which results in fast network backbone construction and GMU discovery. In the Go game with high sample complexity and its varying distribution against the opponents [37,51], the samples obtained during the simulation are unused with concern of overfitting. The work in [52] learns the DNN-based policy and value function through the belief state and the MCTS in the POMDP environment, where simulation samples are also not used due to the overfitting problem. However, such variation is limited in our system and more gain from backbone construction is significant in terms of user throughput. To resolve the concern about the possible overfitting problem, we apply Proximal Policy Optimization (PPO) [53] for the policy network.

Using PPO-based POMCP, we establish a multi-UAV airborne networking system as shown in Fig. 3, which consists of 3 modules indicated by (a) CPTP, (b) MCTS and (c) DRL in the figure.

In the (a) CPTP, when the GMU is observed by UAVs, we update the GMU trajectory from the time that the GMU was not observed to the present. GMU locations are predicted repeatedly using observed mobility data from a previous execution in environment. In the (b) MCTS, the root and following belief nodes are presented as circle shapes and the actions from the belief state are presented as squares. Based on the belief state $B(s,h)$, the MCTS is conducted using simulation with the CPTP and DRL algorithm in Algorithm 3. Once the MCTS search tree is built, we can extract samples for the DRL. Each sample, $\{s,a,r\}$ from the next node to the root to a leaf node in the green-colored region in the figure constitutes mini-batch $M_i(t+1,D) = \{s_{t+1+d}, a_{t+1+d}, r_{t+1+d}\}_{d=1}^{D}$. A state of the root node is likely to be biased to only a few states among all possible states of belief state, $B(s,h)$, while states of following belief nodes are distributed uniformly by random sampling of `Simulate` function (line 16). Thereby, the root node should be excluded from learning in order to avoid situation that the root node is included repeatedly for each sample and the trained model is biased for the root.

In addition, some groups of samples as the mini-batches will be unused to avoid overfitting if a mini-batch starting from a certain first belief node as in the green-colored region is used already ℓ_{max} times for training. This learning limit for a particular mini-batch is reset at the next time $t+1$.

In the (c) DRL, we establish a combined actor–critic network for two parameterized functions, the policy function $\pi_\theta(a|s)$ and value function $V_\theta(s)$ using the Deep Neural Network (DNN) [51]. To deal with exploding discrete actions in our environment, we use the Gaussian policy for continuous action space instead of the softmax policy that provides mean and variance for the normal distribution even though such discretization on continuous action values can cause information loss and eventually leads to a suboptimal policy.

The model θ for the combined actor–critic network is trained K times on mini-batches of length D from the (b) MCTS as shown in (c) DRL. The advantage value for the mini-batch $\hat{A}_i$ is calculated by Eq. (5), where most of the MCTS samples have almost the same depth (to say, collected mini-batches have similar size, D) because we uniformly sample among existing belief nodes after building the search tree. As the model trained with a small batch is unstable for the learning process, reward scaling is useful to improve the stability as shown in the study of the Deep Deterministic Policy Gradient (DDPG) with the small batch size [54]. Therefore we scale the reward to achieve stable learning regardless of the number of simulations.

The loss corresponding to the advantage value $\hat{A}_t$ is calculated as follows for the combined actor–critic network:

$$\mathcal{L}(\theta) = J(\theta)^{Value} - J(\theta)^{PPO}, \tag{37}$$

where $J(\theta)^{Value}$ is the loss for the value function and $J(\theta)^{PPO}$ is objective for the policy function as in Eq. (6), respectively. The loss for the value function is derived from the smooth L1 function:

$$J(\theta)^{Value} = \mathbb{E}_t \left[\text{Smooth}_{L1}(|\hat{V}^{GAE}(s_t) - V_\theta(s_t)|)\right], \tag{38}$$

where $\hat{V}^{GAE}(s_t)$ is the target value depending on GAE, i.e., $\hat{V}^{GAE}(s_t) = V_\theta(s_t) + \hat{A}_t$. Subsequently, we update a model by the following procedure.

$$\theta_{new} = \theta - \alpha_\ell \frac{\partial}{\partial \theta} \mathcal{L}(\theta), \tag{39}$$

where α_ℓ is the learning rate.

Algorithm 3: PPO-based Partially Observable Monte-Carlo Planning

1 **Function** `GenerateStep`$(h, \eta^{\mathcal{U}}_{1:k}, \theta)$:
2 $l_{\mathcal{N}}, l_{\hat{\mathcal{U}}} \sim$ Env
3 **for** $i \in 1 : Z$ **do**
4 $l_{\mathcal{U}\setminus\hat{\mathcal{U}}} \leftarrow$ sample from $\eta^{\mathcal{U}\setminus\hat{\mathcal{U}}}(t)$
5 ζ_{TG} computed from Eq. (33) by $l_{\mathcal{U}}$
6 $s = l_{\mathcal{N}} \cup \zeta_{TG}$
7 $B(h) = B(h) \cup \{s\}$
8 **end for**
9 $B(s, h)$ approximated from Eq. (3) by $B(h)$
10 **for** $u \in 1 : \mathcal{U}$ **do**
11 **if** $u \in \hat{\mathcal{U}}$ **then**
12 $\eta^u_{1:k} =$ `PredictTraj`(u, τ, k)
13 update $\eta^{\mathcal{U}}_{1:k}$ by $\eta^u_{1:k}$
14 set $t = 0$ for u
15 **else**
16 set $t = t + 1$ for u
17 **end if**
18 **end for**
19 **for** $i \in 1 : N$ **do**
20 $s \sim B(s, h)$
21 `Simulate`$(s, h, \eta^{\mathcal{U}}(t), k, \theta)$
22 **end for**
23 $\{M_i(t+1, d)\}_{i=1}^K \leftarrow$ samples from MCTS
24 $\theta_{old} = \theta$
25 $L(h) = 0$
26 **for** $i \in 1 : K$ **do**
27 **if** $\ell_{max} < L(h_{t+1}), h_{t+1} \sim M_i(t+1, d)$ **then**
28 continue
29 **end if**
30 $r^i_{t+1:D} = r^i_{t+1:D}/w_r, r^i_{t+1:D} \sim M_i(t+1, d)$
31 $\hat{A}_t$ computed from Eq. (5) by $M_i(t+1, d)$
32 $J(\theta)^{PPO}$ computed from Eq. (6) by $\hat{A}_t, \theta_{old}$
33 $J(\theta)^{Value}$ computed from Eq. (38) by $\hat{A}_t$
34 $\mathcal{L}(\theta) = J(\theta)^{Value} - J(\theta)^{PPO}$
35 θ updated from Eq. (39) by $\mathcal{L}(\theta)$
36 $L(h_{t+1}) = L(h_{t+1}) + 1$
37 **end for**
38 **return** $\theta, \arg\max_{a \in C(h)} V(ha)$
39 **End Function**

Algorithm 3 describes the detailed procedure of the PPO-based POMCP. In the `GenerateStep`, lines 2–8 describe the update process of the belief state, which is achieved through particles that are collected in the previous MCTS process, $B_t(h_t) = \{s^i | s^i \sim G(s, a, \eta^{\mathcal{U}}(t))\}_{i=1}^M$. Those particles for the current history can be limited by number of the actions in the previous history and number of observations induced by those actions. In the worst case, there is no particle for a particular history h in the belief states generated by the previous MCTS procedure when the history h is created actually by observation and action toward the environment. Therefore it is a challenge to collect enough particles to approximate the belief state. For this, we add artificially Z particles using the CPTP model with newly observed GMU positions before updating the belief state, $B(h) = B(h) \cup \{s^i\}_{i=1}^Z$.

Depending on whether GMUs are observed, the mobility models for GMUs are updated differently in lines 10–18. If a GMU u is observed, the model $\eta^u_{1:k}$ is updated by the new τ-backward trajectory and subsequently the prediction length t is reset to 0 (line 11–14). Otherwise, the previous η^u is kept without changing but the prediction step t just increases by 1, which reduces accordingly available prediction as $\eta^u_{t:k}$ (line 15–16). Here the number of GMUs is supposed to be given for our algorithm even though it is challenging to figure out the exact number of GMUs. For this, the proposed system approximates the number of GMUs by managing active GMUs. In detail, the GCC updates a list of active GMUs when it receives a report from a serving UAV about a newly associated GMU for mobility prediction. Afterwards, the GCC changes a GMU inactive when it is detached intentionally from the network or it has been unobserved for a designated duration.

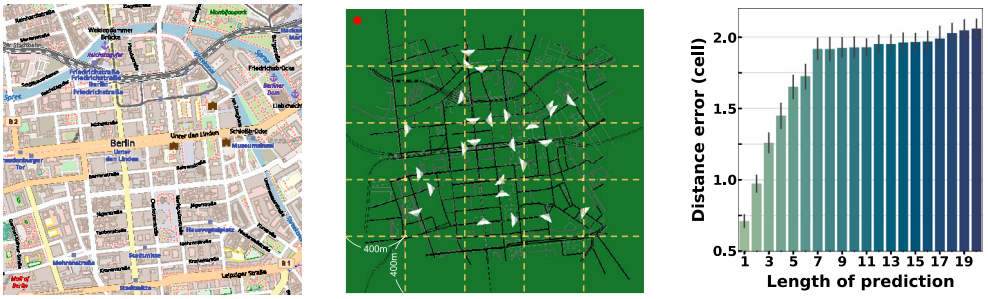
From performing the MCTS algorithm N times in lines 19–22, mini-batches $\{M_i(\cdot)\}_{i=1}^K$ are collected in line 23. To limit the number of training for a particular belief node less than ℓ_{max}, we configure a $L(h)$, the number of training at history h, for all h to 0 and check the number of training for $M_i(\cdot)$ (lines 27–29) while learning with the collected mini-batches K times (line 26–37). Here we only check the belief node for history h_{t+1} because $L(h_{t+1})$ for the first belief node is always bigger than $L(h > h_{t+1})$ of the following belief nodes. Line 30 rescales all rewards of the samples in the mini-batch $M_i(\cdot)$ with w_r. Lines 31–35 update θ for the DRL according to Eq. (5)–(39). Finally, line 38 returns the DNN model weight θ and the action a that maximizes the history value function $V(ha)$. The complexity of the Algorithm 3 is $\mathcal{O}(Z) + \mathcal{O}(\mathcal{U} \cdot k \cdot |l_{RU}(t)|) + \mathcal{O}(N \log(N)) + \mathcal{O}(K)$. The complexity of the MCTS is $\mathcal{O}(N \log(N))$ as it is determined by searching the belief tree, where N is equal to the number of belief nodes. In order to perform approximate optimal action through the MCTS planning, N must be large enough, and in various experimental environments, we observed that the complexity of the MCTS is always larger than other complexities. Therefore, the complexity of the algorithm is $\mathcal{O}(N \log(N))$.

7. Experiment

7.1. Environment realization using Urban mobility simulator

To realize the multi-UAV environment of this study, we use the SUMO (simulation of urban mobility) simulator which is an open source simulator for urban vehicular and pedestrian traffic. We import a part of Berlin map, a 2 × 2 km^2 Manhattan grid shown in Fig. 4(a) and divide this environment region into 25 serving cells of 400 × 400 m^2 like in Fig. 2 considering UAV A2G coverage as an aerial base station. Each cell is indexed sequentially from c_0 to c_{24}. UAVs are deployed to the 25 cells and serve GMUs at the center of cells with 20 m altitude. Hovering enabled by rotary-wing UAVs will allow the UAVs to have more stable wireless connections to the GMUs rather than circular or 8-shape flying by fixed-wing UAVs. Each UAV has A2 A links toward other UAVs at the adjacent cells including diagonal ones. Especially, some UAVs located at the cell c_0, c_1, c_5, and c_6 can have an A2G link to the GCC, the red point on the corner of the cell c_0 as shown in Fig. 4(b).

We configure the UAV speed as 20 m/s for relocation that is highly fast speed for off-the-shelf rotor drones to reduce the service discontinuity of the GMU during the UAVs relocation period; since the propulsion power consumption is critical to the speed of a rotary-wing UAV [47], relocation speed can be controlled for energy saving if



(a) (b) (c)

Fig. 4. Urban environment for experiments. (a) OpenStreetMap network of Berlin. (b) SUMO network imported Berlin. (c) Trajectory prediction distance error of the CPTP model.

Table 3
PPO-based POMCP parameters.

Parameters	Meaning	Value
τ	Backward trajectory length of GMU	2
k	Maximum trajectory prediction length of GMU	20
κ_o, α_o	DPW variables for state	1, 0.3
κ_a, α_a	DPW variables for action	1, 0.3
c	UCB1 coefficient	5.0
Z	Additional number of particles for belief state	100
N	Number of the simulations	2000
ℓ_{max}	Learning threshold for a specific belief node	30
w_r	Weight for rescaling reward	0.01
γ, λ	Discount factors for advantage estimation	0.9, 0.9
ϵ	clipping variable for the PPO	0.2
α_ℓ	Learning rate for θ	0.00005

handover between UAVs is realized to avoid the service discontinuity. Each GMU assumes to need a minimum constant data rate, 300 Kbps for disaster applications, e.g., voice over IP based emergency calls. Although some GMUs instantly use a lower rate than the assigned data rate, the proposed algorithm can calculate varying sum rates of GMUs per cell as a particular number of GMUs with the given constant rate. In other words, the proposed algorithm that aims to maximize the sum rate of all GMUs can select a cell for dispatching an aerial base station based on the sum of the actual demand rate rather than just number of associated GMUs. To increase the minimum data rate, broadband channels, multiple antennas, and duplicated serving UAVs can be considered.

In the test scenario, the GMUs move toward a random destination along roads with random speed, [0, 1.4 m/s] as shown in Fig. 4(b) where GMU locations are denoted as the triangle symbols with moving direction. In random mobility simulated by the SUMO simulator, GMUs move along roads that contain sidewalks and so they have similar paths. The CPTP model is learned based on these patterns. We obtain trajectories of all GMUs; during the SUMO simulation, the GMU identifier, GPS location information (longitude, latitude), and time-stamps are logged periodically every 300 s, allowing GMUs sufficient time to change their cells. Also, UAVs are relocated every 300 s. We do not consider the risk of collisions that may occur during reallocation for simplicity.

7.2. POMCP simulator setup

For the CPTP of POMCP simulation, each GMU has initially a 50-length trajectory to guarantee some reference GMUs in the CPTP model. In addition, only the recent 500 trajectories of GMUs are kept for each cell. Otherwise, the number of candidate RGMUs increases significantly which makes the CPTP model impractical with high search complexity and computational cost.

We test out our CPTP model to demonstrate its feasibility for the POMCP simulation, where we set τ-backward trajectory length to 2. We observe a slight performance improvement after setting τ to 2. Fig. 4(c) shows the performance in terms of distance error according to the prediction length from $k = 1$ to 20, where the CPTP model is built through 30 GMUs. Experiment result with the urban mobility simulation shows very limited prediction error, [0.71,2.06] inter-cell distance even for a long future trajectory. Fig. 4(c) is the experimental result in an environment where all GMUs are fully observable. Afterwards, in Section 7.4, this experimental result is used for performance comparison with the CPTP model built through the proposed algorithm in a partially observable environment.

For a value and policy function approximation in the simulation, we build a DNN with 2 hidden layers (64 × 64 perceptron) for a combined actor–critic network. As the initial scale of the mean returned by the DNN with Gaussian policy is misaligned to the action value range $[0, 24]$, accordingly we re-scale the mean and round up the real value sampled through the Gaussian policy for the Integer action value. The other parameters required for the DNN are addressed in Table 3.

In this experiment, a Linux 20.04 server equipped with Intel CPU i7-9700KF, GPU GeForce RTX 2080, and 32 GB RAM is used, which is inexpensive computing power for the GCC.

7.3. PPO-based POMCP performance

In this section, we investigate the PPO-based POMCP performance with varying network complexity, number of UAVs, and GMUs. For this, experiment is performed for following two cases: (1) $\{\mathcal{N}, \mathcal{U}\} = \{12, 30\}$, and (2) $\{\mathcal{N}, \mathcal{U}\} = \{20, 20\}$, where number of UAVs and GMUs are changed together. Because there are problems when handling network complexity with only one variable. First, if only the number of GMUs is considered, the excessive number of GMUs causes the problem of deploying GMUs in most cells and makes the UAVs immobile because the GMUs are always present in their cells. On the other hand, deploying many UAVs will cover all service areas and eliminate the need for learning; A small number of UAVs places UAVs only on cells around the GCC and make them immobile. Therefore, we vary both variables together to show meaningful performance differences depending on network complexity.

7.3.1. Number of simulations and mini-batches

First, Figs. 5(a) and 6(a) show search performance according to the number of MCTS simulations N for those two scenarios, where the learning threshold for each belief node ℓ_{max} is configured by 30. In $\{20, 20\}$, the reward of all cases converges similarly into 0.65. However, the convergence speed for $N = 1000$ is slower than others because of a small number of mini-batches that increase variance in the learning procedure. The number of mini-batches K for $N = 1000$, 2000, and

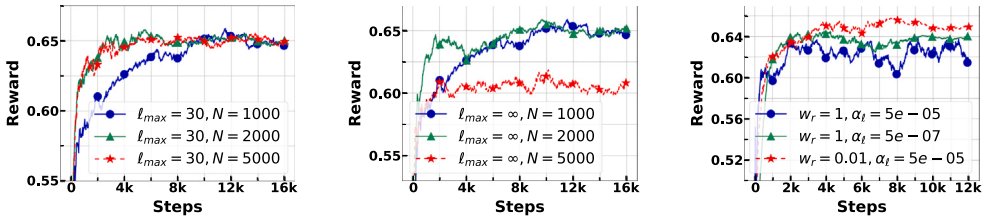


Fig. 5. In $\{\mathcal{N}, \mathcal{U}\} = \{20, 20\}$, performance evaluation according to the PPO-based POMCP hyperparameters. (a) the number of simulations. (b) learning threshold for a specific belief node. (c) weight for rescaling reward and learning rate for θ.

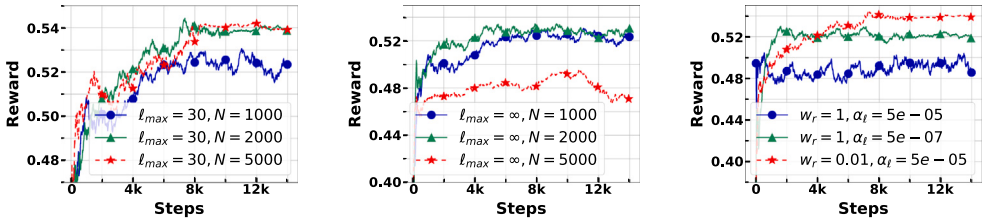


Fig. 6. In $\{\mathcal{N}, \mathcal{U}\} = \{12, 30\}$, performance evaluation according to the PPO-based POMCP hyperparameters. (a) the number of simulations. (b) learning threshold for a specific belief node. (c) weight for rescaling reward and learning rate for θ.

5000 is 144, 300, and 390, respectively. On the other hand, in $\{12, 30\}$, the $N = 1000$ case shows a lower reward, about 2% than others, and unstable convergence because it has insufficient simulation for $B(s, h)$. The number of unobserved GMUs enlarges the variance of the belief state $B(s, h)$ as the network complexity increases with a small number of UAVs. Accordingly, the $N = 1000$ fluctuates between 0.52 and 0.53 in $\{12, 30\}$ with large variance.

7.3.2. Learning threshold for belief nodes

Figs. 5(b) and 6(b) show experimental results on the learning threshold, $\ell_{max} = \infty$. In contrast to the case $\ell_{max} = 30$, the reward does not increase gradually according to increasing simulations because the number of samples increases significantly for the same belief node, especially in the case of $N = 5000$; the K for $N = 1000$, 2000, and 5000 is 144, 750, and 2184, respectively. In other words, the high K value induces repeated learning especially on the top-level samples at the search tree. The $N = 1000$ case with the same K shows no change in performance as proof, while the $N = 5000$ case decreases from 0.65 to 0.61. In $\{12, 30\}$, the reward for both $N = 2000$ and 5000 decreases from 0.54 to 0.53 and 0.47, respectively. In summary, more UAVs against number of GMUs are gainful in terms of reward, and a learning cap for a particular belief node is helpful for learning stability.

For the following experiments, we use the $N = 2000$ which gives the best performance in both scenarios. With $N = 2000$, the proposed POMCP algorithm with search and learning takes only 5.2 s in our test bed, which is much smaller than the relocation cycle of 300 s and can be reduced by the high computing power of the GCC. Accordingly, our approach is appropriate for the purpose of online UAV control in multi-hop airborne networks.

7.3.3. Reward scaling and learning rate

Figs. 5(c) and 6(c) investigate the effect of reward scaling w_r on performance compared to the learning rate α_ℓ in the above two scenarios. In $\{20, 20\}$, the $w_r = 1, \alpha_\ell = 5e-05$ case that does not scale the reward has $\hat{A}_t$ about 100 times more than the $w_r = 0.01, \alpha_\ell = 5e{-}05$ case, which initially increases the learning speed, but fluctuates between 0.6 and 0.64. Alternatively, the $w_r = 1, \alpha_\ell = 5e{-}07$ case reduces α_ℓ by 100 times to make its scale equal to the $w_r = 0.01, \alpha_\ell = 5e-05$. In comparison of two approaches by α_ℓ and w_r, the $w_r = 0.01, \alpha_\ell = 5e-05$ case outperforms the other case. With the advantage function calculated in Eq. (5), $\hat{A}_t = \sum_t^{T-1} (\gamma)^{t-1} r_t + \gamma^{T-t+2} V(s_T) - V(s_t)$ with $\lambda = 1$, the α_ℓ adjusts not only the value function, but also reward in Eq. (39), while the w_r scales only rewards of samples from simulations. Also, we observe that the $w_r = 0.01, \alpha_\ell = 5e-05$ case improves reward performance by 0.54 compared to both $w_r = 1, \alpha_\ell = 5e-05$ and $w_r = 1, \alpha_\ell = 5e-07$ cases with lower reward, 0.49 and 0.52 in the $\{12, 30\}$ scenario. Hereafter we set the reward rescaling factor w_r with 0.01 to provide stable learning regardless of the K.

7.3.4. UAV deployment error

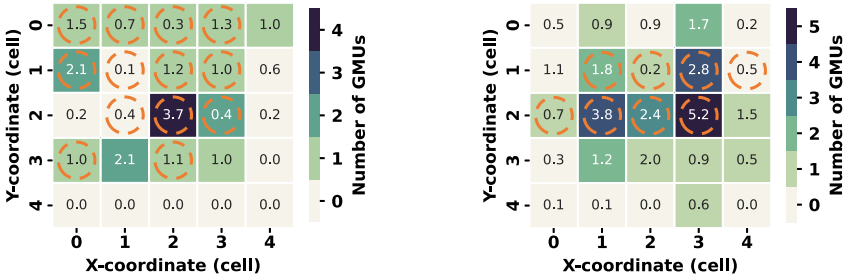
Fig. 7 illustrates the average location of GMUs sampled from the belief state $B(s, h)$, the actual location of GMUs, and an action obtained from the MCTS in the corresponding $B(s, h)$. In the figure, the color index of each cell indicates the number of actual GMUs, the number of each cell is the average of GMUs sampled from $B(s, h)$ $\sum_{s \in B(h)} B(s, h) \cdot \zeta_{TG}$, and the dashed circles mean UAV location by a selected action.

In Fig. 7(a) for the $\{20, 20\}$, the selected action successfully dispatches an UAV to the cell (2, 2) with the most GMUs, but places UAVs to two cells (1, 1) and (1, 2) without actual GMUs. Conversely, three cells (1, 3), (3, 3) and (4, 0) having actual GMUs have no serving UAV. In Fig. 7(b) for the $\{12, 30\}$, more GMUs have outage for less number of UAVs; 8 cells with GMUs have no UAV compared to a single mislocation (4, 1). In total, 12 and 8 UAVs are located in those two cases while the remaining UAVs are kept inactive for energy conservation.

This deployment error occurs due to limited actions evaluated during the MCTS. In the process of adding actions to the action set $C(h)$ (lines 27–30 of the `SelectAction` function), the size of the set $C(h)$ is controlled by the DPW parameters κ_a and α_a; in $N = 2000$, the size of $C(h)$ corresponding to the root belief node is 10. Each action in the $C(h)$ is derived by the DNN for a specific state s. Therefore, only 10 actions are evaluated during the MCTS, not for all states sampled from belief state $B(s, h)$. Although the difference between actual and predictive GMU distribution is insignificant as shown in the figure, some UAVs are misplaced. As part of future work, we will study further about a model that evaluates the whole of $B(s, h)$.

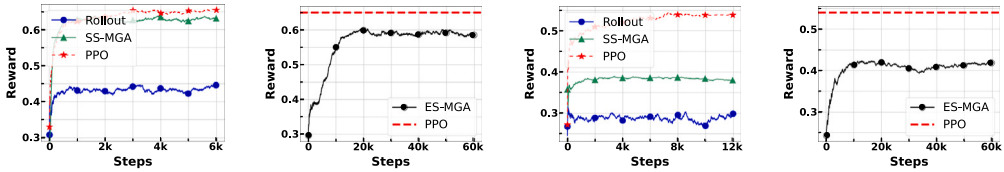
7.3.5. Comparison of several MCTS variants

Fig. 8 shows the performance of the proposed algorithm compared to three algorithms, (i) Rollout, (ii) Simulation Sample based Mini-batch Gradient Ascent (SS-MGA), and (iii) Environment Sample based Mini-batch Gradient Ascent (ES-MGA) model.



(a) $\{\mathcal{N},\mathcal{U}\} = \{20, 20\}$ (b) $\{\mathcal{N},\mathcal{U}\} = \{12, 30\}$

Fig. 7. Performance evaluation of whether the PPO-based POMCP tracks trajectories of GMUs and appropriately deploys multi-UAV.



(a) $\{\mathcal{N},\mathcal{U}\} = \{20, 20\}$ (b) $\{\mathcal{N},\mathcal{U}\} = \{20, 20\}$ (c) $\{\mathcal{N},\mathcal{U}\} = \{12, 30\}$ (d) $\{\mathcal{N},\mathcal{U}\} = \{12, 30\}$

Fig. 8. Comparison of performance obtained through several MCTS algorithms.

- Rollout : this rollout model adopts a random rollout policy instead of the DRL when the tree is expanded in the DPW-based MCTS. In order to evaluate the value of a new belief node, the rollout model expects the value with random actions from the current time t to $t+10$, $V(s) = \sum_{i=t}^{t+10} \gamma^{i-t} r_t$. In addition, the rollout model selects an action randomly when adding the new action to the set of actions $C(h)$ instead of the DRL.
- SS-MGA : the SS-MGA model uses mini-batch gradient ascent instead of the PPO for the objective of the policy function, $J(\theta)^{MGA} = \mathbb{E}_t \left[\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t\right]$. This model is almost identical to the proposed model, except for the constraint on the objective of the policy function in line 32 of Algorithm 3. Since the PPO has the constraint through the clipping function in contrast to the SS-MGA model, the proposed algorithm can mitigate overfitting resulting from the use of simulation samples.
- ES-MGA : In contrast to the proposed model using simulation samples, the ES-MGA model uses samples collected directly from the environment. The ES-MGA model uses a mini-batch gradient ascent for the objective of the policy function. The size of a mini-batch is 32.

Figs. 8(a) and 8(c) compare the rollout, SS-MGA, and PPO model in terms of reward: 0.49, 0.63, and 0.65, respectively in $\{20, 20\}$, and 0.29, 0.37 and 0.54 in $\{12, 30\}$. In the rollout model, limited random actions cause the worst performance; the number of evaluated actions is only 10 in $N = 2000$. Accordingly, more simulations are needed to derive an approximate solution with the random actions in the current $B(s, h)$. Furthermore, the higher network complexity exacerbates search performance as shown in the case $\{12, 30\}$. The effect of the network complexity looks clear when comparing the reward gap between the PPO and SS-MGA model in each $\{20, 20\}$ and $\{12, 30\}$. Since the SS-MGA model uses the objective of the policy function without clipping function of the PPO, the SS-MGA model is overfit to a specific batch at each iteration and falls into a local minimum as variance of the collected samples increases.

On the other hand, Figs. 8(b) and 8(d) compare the PPO and ES-SGA using samples from the environment. The ES-SGA shows slow learning speed compared to the PPO since it obtains only a single sample from each step: $\{20, 20\} = 20$ K, and $\{12, 30\} = 15$ K. In addition, the ES-MGA model has lower performance than the PPO: $\{20, 20\} = 0.59$, $\{12, 30\} = 0.41$. The ES-MGA model performs worse than the PPO because the environmental samples can be non-Independent Identically Distributed (non-IID) and cause overfitting in the model. To solve the non-IID problem, the ES-MGA model should train a DRL model after collecting enough samples for long duration as an offline learning approach [37,52]. However, the offline learning is inappropriate for our dynamic disaster situation.

7.4. Airborne network performance

This section shows overall performance of multi-UAV system having dual objectives that are maximized by the PPO-based POMCP proportionally with objective weights ω_1 and $\omega_2 = 1 - \omega_1$ in Eq. (36). First, the average data rate of GMUs is $\{12, 30\} = 0$, 140, 142, 153, and 160 Kbps, and $\{20, 20\} = 92$, 180, 210, 215, and 218 Kbps according to the ω_1 weight for data rate sum as shown in Fig. 9(a). Increasing ω_1 encourages selecting an action that maximizes throughput instead of energy saving. If each data rate is normalized by the demand rate (300 Kbps), the ratio value can be $\{12, 30\} = 0$, 0.47, 0.47, 0.51, and 0.53, and $\{20, 20\} = 0.31$, 0.6, 0.7, 0.72, and 0.73, which is proportional to the number of observed GMUs in Fig. 9(b): $\{12, 30\} = 4$, 14, 14.3, 16, and 18, and $\{20, 20\} = 5.6$, 11, 14.9, 15, and 15.1. In other words, the GMU observability related to UAV deployment affects mainly entire network throughput. The number of available UAVs is another key for better throughput according to comparison of the two scenarios. In $\{12, 30\}$ with $\omega_1 = 0.1$, the data rate is 0 regardless of several observed GMUs because the UAV network backbone is not established due to lack of active UAVs; most of UAVs are inactive for reward from energy saving.

To verify the capability of the proposed algorithm for future trajectory prediction of GMUs, the average prediction distance error of GMUs is measured according to the ω_1 in $\{12, 30\} = 1.55$, 0.67, 0.65, 0.5, 0.42, and $\{20, 20\} = 1.05$, 0.55, 0.22, 0.22, 0.21 as shown in Fig. 9(c). Each prediction distance error is calculated as the difference between the predicted GMU position and the actual GMU position. More UAVs enhance GMU observability and lessen prediction error.

In our experiment, there is little difference in performance between the CPTP model built through the proposed algorithm in a partially observable environment and the CPTP model with fully observable GMUs because the proposed algorithm tracks the trajectory of GMUs successfully from the initial environment without sufficient trajectory information of GMUs, achieving overall low prediction length errors.

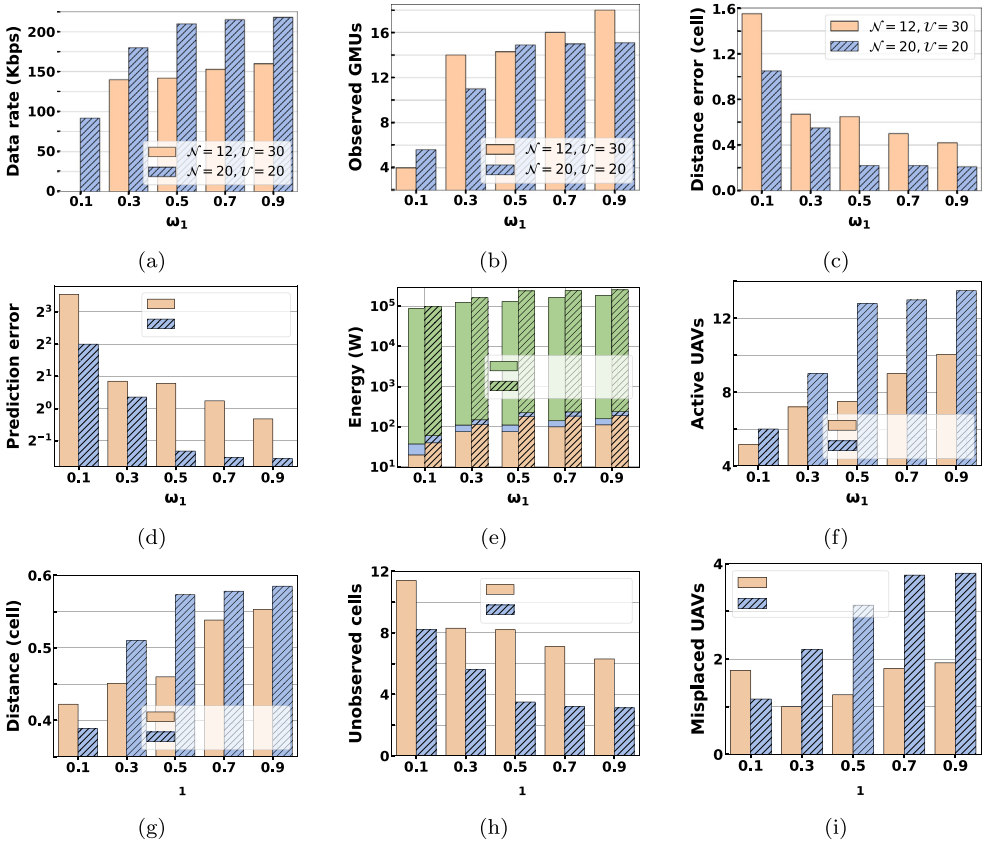


Fig. 9. Airborne network performance evaluation of the PPO-based POMCP according to objective weight. (a) Average date rate of GMUs. (b) Number of observed GMUs. (c) Average trajectory prediction distance error of GMUs. (d) Average trajectory prediction length error of GMUs. (e) Total Energy consumption of UAVs. (f) Number of active UAVs. (g) Average moving distance of active UAVs upon reallocation. (h) Number of unobserved cells with GMUs. (i) Number of UAVs misplaced in cells without GUMs.

However, performance can be varying with the scenarios such as a size of the experimental map that affects number of unobserved GMUs and non-observation duration.

From aforementioned result, the average prediction length error, a period that a GMU has not been detected from UAVs can be derived by both the average prediction distance error and distance error/prediction length relation of Fig. 4(c), which is $\{12, 30\} = 4.5$, 0.48, 0.46, 0.36, and 0.3, and $\{20, 20\} = 2.3$, 0.39, 0.16, 0.16, and 0.15. Meanwhile, the measured average prediction length error is $\{12, 30\} = 11.7$, 1.8, 1.72, 1.18, and 0.8, and $\{20, 20\} = 4$, 1.28, 0.4, 0.35, and 0.34 as shown in Fig. 9(d).

Even with the same distance error, those derived and measured average prediction length errors are different; the measured error is significantly higher than the other because individual prediction length error is highly varying according to GMUs. For example of an environment with 4 GMUs, suppose that the prediction length error for all 4 GMUs is 3, the average prediction length error is 3 and the expected distance error becomes 1 according to Fig. 4(c). On the other hand, if each prediction length error for those 4 GMUs is 0, 3, 3, and 18, the average prediction length error increases up to 6, but the distance error is same as 1 (each distance error is 0, 1, 1, and 2). Despite of several GMUs having high prediction length errors, the average distance error is consistent because the increment of the distance error becomes saturated with the prediction length as shown in Fig. 4(c).

Such a deep isolation of some GMUs addressed by high prediction length error is originated by the objective of the sum rate maximization. Therefore, fair rate assignment among GMUs needs to be considered in future works.

Next, Fig. 9(e) shows the total energy consumption of UAVs, which increases in proportion to ω_1 for providing suitable coverage: $\{12, 30\} = 87.3$, 124.5, 130.4, 164.9, and 185.9 kW, and $\{20, 20\} = 99.3$, 161.9, 239.9, 244.4, and 254.9 kW. Energy consumption is composed of A2 A communication, A2G communication, and propulsion energy consumption in order, where (A2 A, A2G) communication energy consumption are $\{12, 30\} = (20, 18)$, (77, 34), (77, 34), (100, 42), and (112, 47) W, and $\{20, 20\} = (40, 21)$, (114, 36), (180, 50), (185, 50), and (191, 52) W. Therefore, most of the energy consumption occurs in the propulsion energy. Propulsion energy is determined by the number of active UAVs and their moving distance.

The energy consumption increases in proportion to the number of active UAVs in Fig. 9(f): $\{12, 30\} = 5.2$, 7.2, 7.5, 9, and 11, and $\{20, 20\} = 6$, 9, 12.8, 13, and 13.5. When the active UAVs are relocated, Fig. 9(g) shows the average movement distance of the active UAVs: $\{12, 30\} = 0.42$, 0.45, 0.46, 0.54, and 0.55, and $\{20, 20\} = 0.39$, 0.51, 0.57, 0.58, and 0.59. This result confirms that only half of the active UAVs move one cell and the rest of them just switch on their operation mode. In other words, the proposed algorithm concentrates on minimization of propulsion energy consumption, which accounts for most of the energy consumption.

Figs. 9(h) and 9(i) show the detailed evaluation for the limitation mentioned in Fig. 7. Fig. 9(h) shows the number of outage cells due to UAV absence: $\{12, 30\} = 11.4$, 8.3, 8.2, 7.1, and 6.3, and $\{20, 20\} = 8.2$, 5.6, 3.5, 3.2, and 3.12. Conversely, Fig. 9(i) shows the number of cells with unnecessary UAVs: $\{12, 30\} = 1.76$, 1, 1.25, 1.8, and 1.92, and $\{20, 20\} = 1.16$, 2.2, 3.13, 3.76, and 3.8. When calculating the results of Fig. 9(i), cells deployed for multi-hop backhaul routing purpose are excluded. The $\{20, 20\}$ shows better performance trivially than $\{12, 30\}$ in Fig. 9(h) by deploying more UAVs. Meanwhile, many UAV deployment increases the probability of misplaced UAVs, which consequently makes the $\{20, 20\}$ achieve lower performance in Fig. 9(i). We will further study about finding optimal number of UAVs for a given network and reducing occasion of the misplacement.

8. Conclusion

We propose an online multi-UAV deployment algorithm for multi-hop airborne networks that provide wireless connections to GMUs in disaster situations and digital divide. To maximize the dual objectives of energy saving and throughput in a partially observable environment with user mobility, we enhance the MCTS-based POMCP using the DPW and DNN-assisted search. For the MCTS simulation, we develop the CPTP model for future trajectory prediction of GMUs and generate training samples for online learning.

Experimental results demonstrate the feasibility of the proposed algorithm with accurate prediction on GMUs' future trajectory and reliable deployment for a multi-UAV ad-hoc network. However, we still have the following challenges for future works. First, misplacement of UAVs against empty cells is critical as possible actions are insufficiently explored for states sampled from belief state. Second, fairness should be considered in the objective to prevent long outage of several GMUs. Third, a DRL framework for GMU trajectory prediction in the environment with a large service area and many GMUs needs to be developed instead of the CPTP model that suffers from increasing time complexity by the number of unobserved GMUs. Fourth, in order to solve POMDP, there are other algorithms such as the RNN-based DRL as well as the MCTS-based algorithm. Further analysis is needed through comparison with various algorithms.

CRediT authorship contribution statement

Kyungho Ryu: Writing – original draft, Visualization, Software, Methodology. **Wooseong Kim:** Writing – review & editing, Validation, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Wooseong Kim reports financial support was provided by National Research Foundation of Korea.

Data availability

No data was used for the research described in the article.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF), South Korea funded by the Ministry of Science and Information & Communication Technologies (2022R1F1A1074767) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (RS-2023-00228316).

References

[1] M. Erdelj, E. Natalizio, K.R. Chowdhury, I.F. Akyildiz, Help from the sky: Leveraging UAVs for disaster management, IEEE Pervasive Comput. 16 (1) (2017) 24–32.

[2] K.G. Panda, S. Das, D. Sen, W. Arif, Design and deployment of UAV-aided post-disaster emergency network, IEEE Access 7 (2019) 102985–102999.

[3] N. Kumar, M. Ghosh, C. Singhal, UaV network for surveillance of inaccessible regions with zero blind spots, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, IEEE, 2020, pp. 1213–1218.

[4] K. Gomez, A. Hourani, L. Goratti, R. Riggio, S. Kandeepan, I. Bucaille, Capacity evaluation of aerial LTE base-stations for public safety communications, in: 2015 European Conference on Networks and Communications, EuCNC, IEEE, 2015, pp. 133–138.

[5] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, M. Debbah, A tutorial on UAVs for wireless networks: Applications, challenges, and open problems, IEEE Commun. Surv. Tutor. 21 (3) (2019) 2334–2360.

[6] L. Gupta, R. Jain, G. Vaszkun, Survey of important issues in UAV communication networks, IEEE Commun. Surv. Tutor. 18 (2) (2015) 1123–1152.

[7] D.S. Lakew, U. Sa'ad, N.-N. Dao, W. Na, S. Cho, Routing in flying ad hoc networks: A comprehensive survey, IEEE Commun. Surv. Tutor. 22 (2) (2020) 1071–1120.

[8] A.A. Khuwaja, Y. Chen, N. Zhao, M.-S. Alouini, P. Dobbins, A survey of channel modeling for UAV communications, IEEE Commun. Surv. Tutor. 20 (4) (2018) 2804–2821.

[9] X. Cao, P. Yang, M. Alzenad, X. Xi, D. Wu, H. Yanikomeroglu, Airborne communication networks: A survey, IEEE J. Sel. Areas Commun. 36 (9) (2018) 1907–1926.

[10] B. Li, Z. Fei, Y. Zhang, UAV communications for 5G and beyond: Recent advances and future trends, IEEE Internet Things J. 6 (2) (2018) 2241–2263.

[11] E.P. De Freitas, T. Heimfarth, I.F. Netto, C.E. Lino, C.E. Pereira, A.M. Ferreira, F.R. Wagner, T. Larsson, UAV relay network to support WSN connectivity, in: International Congress on Ultra Modern Telecommunications and Control Systems, IEEE, 2010, pp. 309–314.

[12] D. Orfanus, E.P. De Freitas, F. Eliassen, Self-organization as a supporting paradigm for military UAV relay networks, IEEE Commun. Lett. 20 (4) (2016) 804–807.

[13] A. Al-Hourani, S. Kandeepan, S. Lardner, Optimal LAP altitude for maximum coverage, IEEE Wirel. Commun. Lett. 3 (6) (2014) 569–572.

[14] M. Mozaffari, W. Saad, M. Bennis, M. Debbah, Drone small cells in the clouds: Design, deployment and performance analysis, in: 2015 IEEE Global Communications Conference, GLOBECOM, IEEE, 2015, pp. 1–6.

[15] M. Mozaffari, W. Saad, M. Bennis, M. Debbah, Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage, IEEE Commun. Lett. 20 (8) (2016) 1647–1650.

[16] M. Mozaffari, W. Saad, M. Bennis, M. Debbah, Mobile Unmanned Aerial Vehicles (UAVs) for energy-efficient Internet of Things communications, IEEE Trans. Wireless Commun. 16 (11) (2017) 7574–7589.

[17] E. Kalantari, H. Yanikomeroglu, A. Yongacoglu, On the number and 3D placement of drone base stations in wireless cellular networks, in: 2016 IEEE 84th Vehicular Technology Conference, VTC-Fall, IEEE, 2016, pp. 1–6.

[18] J. Košmerl, A. Vilhar, Base stations placement optimization in wireless networks for emergency communications, in: 2014 IEEE International Conference on Communications Workshops, ICC, IEEE, 2014, pp. 200–205.

[19] P. Li, J. Xu, Placement optimization for UAV-enabled wireless networks with multi-HoP backhauls, J. Commun. Inf. Netw. 3 (4) (2018) 64–73.

[20] J. Fan, M. Cui, G. Zhang, Y. Chen, Throughput improvement for multi-HoP UAV relaying, IEEE Access 7 (2019) 147732–147742.

[21] G. Zhang, H. Yan, Y. Zeng, M. Cui, Y. Liu, Trajectory optimization and power allocation for multi-HoP UAV relaying communications, IEEE Access 6 (2018) 48566–48576.

[22] Y. Chen, N. Zhao, Z. Ding, M.-S. Alouini, Multiple UAVs as relays: Multi-HoP single link versus multiple dual-HoP links, IEEE Trans. Wireless Commun. 17 (9) (2018) 6348–6359.

[23] U. Choi, C. Moon, J. Ahn, Energy minimization of dynamic multi-UAV communication network for cooperative multi-HoP data gathering, IEEE Trans. Aerosp. Electron. Syst. (2022).

[24] A. Gholami, N. Torkzaban, J.S. Baras, C. Papagianni, Joint mobility-aware UAV placement and routing in multi-HoP UAV relaying systems, in: International Conference on Ad Hoc Networks, Springer, 2021, pp. 55–69.

[25] R. Ding, J. Chen, W. Wu, J. Liu, F. Gao, X. Shen, Packet routing in dynamic multi-HoP UAV relay network: A multi-agent learning approach, IEEE Trans. Veh. Technol. 71 (9) (2022) 10059–10072.

[26] D. Silver, J. Veness, Monte-Carlo planning in large POMDPs, in: Advances in Neural Information Processing Systems, vol. 23, 2010.

[27] R. Ding, Y. Xu, F. Gao, X. Shen, Trajectory design and access control for air–ground coordinated communications system with multiagent deep reinforcement learning, IEEE Internet Things J. 9 (8) (2021) 5785–5798.

[28] C.H. Liu, Z. Chen, J. Tang, J. Xu, C. Piao, Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach, IEEE J. Sel. Areas Commun. 36 (9) (2018) 2059–2070.

[29] C.H. Liu, X. Ma, X. Gao, J. Tang, Distributed energy-efficient multi-UAV navigation for long-term communication coverage by deep reinforcement learning, IEEE Trans. Mob. Comput. 19 (6) (2019) 1274–1285.

[30] Z. Ye, K. Wang, Y. Chen, X. Jiang, G. Song, Multi-UAV navigation for partially observable communication coverage by graph reinforcement learning, IEEE Trans. Mob. Comput. (2022).

[31] X. Liu, Y. Liu, Y. Chen, Reinforcement learning in multiple-UAV networks: Deployment and movement design, IEEE Trans. Veh. Technol. 68 (8) (2019) 8036–8049.

[32] X. Liu, Y. Liu, Y. Chen, L. Hanzo, Trajectory design and power control for multi-UAV assisted wireless networks: A machine learning approach, IEEE Trans. Veh. Technol. 68 (8) (2019) 7957–7969.

[33] B. Omoniwa, B. Galkin, I. Dusparic, Communication-enabled deep reinforcement learning to optimise energy-efficiency in UAV-assisted networks, Veh. Commun. 43 (2023) 100640.
[34] R. Sun, D. Zhao, L. Ding, J. Zhang, H. Ma, UAV-Net+: Effective and energy-efficient UAV network deployment for extending cell tower coverage with dynamic demands, IEEE Trans. Veh. Technol. 72 (1) (2022) 973–985.
[35] P. Karmakar, V.K. Shah, S. Roy, K. Hazra, S. Saha, S. Nandi, Reliable backhauling in aerial communication networks against uav failures: A deep reinforcement learning approach, IEEE Trans. Netw. Serv. Manag. 19 (3) (2022) 2798–2811.
[36] C. Qu, F.B. Sorbelli, R. Singh, P. Calyam, S.K. Das, Environmentally-aware and energy-efficient multi-drone coordination and networking for disaster response, IEEE Trans. Netw. Serv. Manag. (2023).
[37] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, nature 529 (7587) (2016) 484–489.
[38] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in: International Conference on Machine Learning, PMLR, 2015, pp. 1889–1897.
[39] M. Boon, A. Drijfhout, S. Tesfamichael, Comparison of a fixed-wing and multi-rotor UaV for environmental mapping applications: A case study, Int. Arch. Photogramm. Remote Sens. Spatial Inform. Sci. 42 (2017) 47.
[40] O. Bouachir, A. Abrassart, F. Garcia, N. Larrieu, A mobility model for UAV ad hoc network, in: 2014 International Conference on Unmanned Aircraft Systems, ICUAS, IEEE, 2014, pp. 383–388.
[41] C. Yan, L. Fu, J. Zhang, J. Wang, A comprehensive survey on UAV communication channel modeling, IEEE Access 7 (2019) 107769–107792.
[42] A. Al-Hourani, S. Kandeepan, A. Jamalipour, Modeling air-to-ground path loss for low altitude platforms in urban environments, in: 2014 IEEE Global Communications Conference, IEEE, 2014, pp. 2898–2904.
[43] N. Goddemeier, C. Wietfeld, Investigation of air-to-air channel characteristics and a UAV specific extension to the rice model, in: 2015 IEEE Globecom Workshops, GC Wkshps, IEEE, 2015, pp. 1–5.
[44] H.V. Abeywickrama, B.A. Jayawickrama, Y. He, E. Dutkiewicz, Comprehensive energy consumption model for unmanned aerial vehicles, based on empirical studies of battery performance, IEEE Access 6 (2018) 58383–58394, http://dx.doi.org/10.1109/ACCESS.2018.2875040.
[45] Y. Zeng, R. Zhang, Energy-efficient UAV communication with trajectory optimization, IEEE Trans. Wireless Commun. 16 (6) (2017) 3747–3760.
[46] F. Dong, L. Li, Z. Lu, Q. Pan, W. Zheng, Energy-efficiency for fixed-wing UAV-enabled data collection and forwarding, in: 2019 IEEE International Conference on Communications Workshops, ICC Workshops, 2019, pp. 1–6, http://dx.doi.org/10.1109/ICCW.2019.8757098.
[47] Y. Zeng, J. Xu, R. Zhang, Energy minimization for wireless communication with rotary-wing UAV, IEEE Trans. Wireless Commun. 18 (4) (2019) 2329–2345.
[48] A. Garcia-Saavedra, P. Serrano, A. Banchs, G. Bianchi, Energy consumption anatomy of 802.11 devices and its implication on modeling and design, in: Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies, CoNEXT '12, Association for Computing Machinery, New York, NY, USA, 2012, pp. 169–180, http://dx.doi.org/10.1145/2413176.2413197.
[49] K. Ryu, W. Kim, Multi-objective optimization of energy saving and throughput in heterogeneous networks using deep reinforcement learning, Sensors 21 (23) (2021) 7925.
[50] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, N. Bonnard, Continuous upper confidence trees, in: International Conference on Learning and Intelligent Optimization, Springer, 2011, pp. 433–445.
[51] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, nature 550 (7676) (2017) 354–359.
[52] C.-J. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, M.J. Kochenderfer, Combining planning and deep reinforcement learning in tactical decision making for autonomous driving, IEEE Trans. Intell. Veh. 5 (2) (2019) 294–305.
[53] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347.
[54] Y. Duan, X. Chen, R. Houthooft, J. Schulman, P. Abbeel, Benchmarking deep reinforcement learning for continuous control, in: International Conference on Machine Learning, PMLR, 2016, pp. 1329–1338.

Kyungho Ryu received the B.S. degree in computer engineering from Gachon University, South Korea, where he is currently pursuing the M.S. degree in computer engineering. He developed many distributed web-applications-based on blockchain. His research interest includes distributed system-based on blockchain and off-chain technologies.

Wooseong Kim received the Ph.D. degree in computer science from UCLA. He used to work as a Researcher with Samsung Electronics, Hyundai Motor, LG Electronics, SK Hynix Semiconductor, and so on. He is currently an Associate Professor with the Computer Engineering Department, Gachon University, South Korea. He had standardization activity in several SDOs like 3GPP, TTA, and ETSI. His research interests include blockchain, P2P networks, multihop ad hoc networks, 5G telecommunication systems, SDN/NFV, the IoT, and so on.