# A Deep Q-Learning based, Base-Station Connectivity-Aware, Decentralized Pheromone Mobility Model for Autonomous UAV Networks

Shreyas Devaraju, *Student Member, IEEE*, Alexander Ihler, *Member, IEEE*, Sunil Kumar, *Senior Member, IEEE*



***Abstract*—UAV networks consisting of low SWaP (size, weight, and power), fixed-wing UAVs are used in many applications, including area monitoring, search and rescue, surveillance, and tracking. Performing these operations efficiently requires a scalable, decentralized, autonomous UAV network architecture with high network connectivity. Whereas fast area coverage is needed for quickly sensing the area, strong node degree and base station (BS) connectivity are needed for UAV control and coordination and for transmitting sensed information to the BS in real time. However, the area coverage and connectivity exhibit a fundamental trade-off: maintaining connectivity restricts the UAVs' ability to explore.**

**In this paper, we first present a node degree and BS connectivity-aware distributed pheromone (BS-CAP) mobility model to autonomously coordinate the UAV movements in a decentralized UAV network. This model maintains a desired connectivity among 1-hop neighbors and to the BS while achieving fast area coverage. Next, we propose a deep Q-learning policy based BS-CAP model (BSCAP-DQN) to further tune and improve the coverage and connectivity trade-off. Since it is not practical to know the complete topology of such a network in real time, the proposed mobility models work online, are fully distributed, and rely on neighborhood information. Our simulations demonstrate that both proposed models achieve efficient area coverage and desired node degree and BS connectivity, improving significantly over existing schemes.**

***Index Terms*—Unmanned aerial vehicles, autonomous UAV swarm, autonomous UAV networks, pheromone mobility model, node degree, base station connectivity, network connectivity, deep Q-learning.**

## I. Introduction

Unmanned aerial vehicles (UAVs) equipped with self-localization and sensing capabilities have become popular for applications such as search-and-rescue, surveillance and area monitoring, and target tracking [1]–[3]. We focus on low SWaP (size, weight, and power), fixed-wing UAVs, which offer a balance of portability and area coverage (with higher speeds and longer lifetimes than rotor-based UAVs). However, low SWaP UAVs face several practical difficulties: they have a limited range of communication and are more prone to failure than larger UAVs. These issues motivate a **decentralized** and **autonomous** network of UAVs, in which the nodes explore an area, perform local sensing, and communicate with their neighbors without any global knowledge of network topology. While decentralized networks are easily scalable and have no single point of failure, the autonomous node movement generates uncertain trajectories, leading to disconnected networks and unstable communication routes. To gather the sensed information and allow real-time coordination without dedicated communications infrastructure (which is typically unavailable in the target environments), the UAVs must act so as to maintain connectivity. In particular, they must trade off between area coverage and network connectivity, since dispersing the UAVs to improve coverage can negatively impact their connectivity, and vice-versa [3].

More concretely, we consider a scenario consisting of a swarm of 30-50 low SWaP fixed-wing UAVs (e.g., [4]–[7]) performing area monitoring and surveillance in a 6 km $\times$ 6 km area, in which no fixed communication infrastructure, such as a cellular network, is available. The inexpensive, low SWaP UAVs have a communication range of around 1000 m (e.g., [8]–[10]) and fly at low altitudes at speeds varying from 20 m/s to 40 m/s (e.g., [5]). Fig. 1 illustrates an area monitoring application in an inaccessible, disaster-hit area. With no central authority in charge of path planning, each UAV autonomously determines its trajectory based on its local neighborhood information. The individual UAVs collaboratively explore the area to collect information, which must then be reliably sent to an aerial base station (BS). The aerial BS can then forward the information to a control center to make informed decisions in real time.

These settings require a moderate to high density of UAVs in the target area, since too few low SWaP UAVs flying at low altitudes cannot cover the area in a limited operational time. The network's decentralized nature enables it to be robust to loss of UAVs (due to limited battery power, hostile environments, or other failures); by acting autonomously, the network can react to such changes, including enduring periods of disconnection, and UAVs do not need to maintain communications with a pilot. The UAVs continuously monitor the area, find objects of interest, and then report back. When they find a target of interest, they initiate a connection to the BS (located at the boundary of the monitored area) and communicate the sensed information to the BS in real time.

One common method to coordinate and control the UAV movement is to use stigmergic digital pheromones [1], [11], [12]. Pheromone-based mobility models are simple, scalable, and robust, and can be implemented in both centralized or decentralized systems. However, these models focus only on

S. Devaraju is with Computational Science, San Diego State University & University of California, Irvine, CA, USA, sdevaraju@sdsu.edu

A. Ihler is with School of Information & Computer Science, University of California, Irvine, CA, USA, ihler@ics.uci.edu

S. Kumar is with Electrical & Computer Engineering Department, San Diego State University, San Diego, CA USA, skumar@sdsu.edu

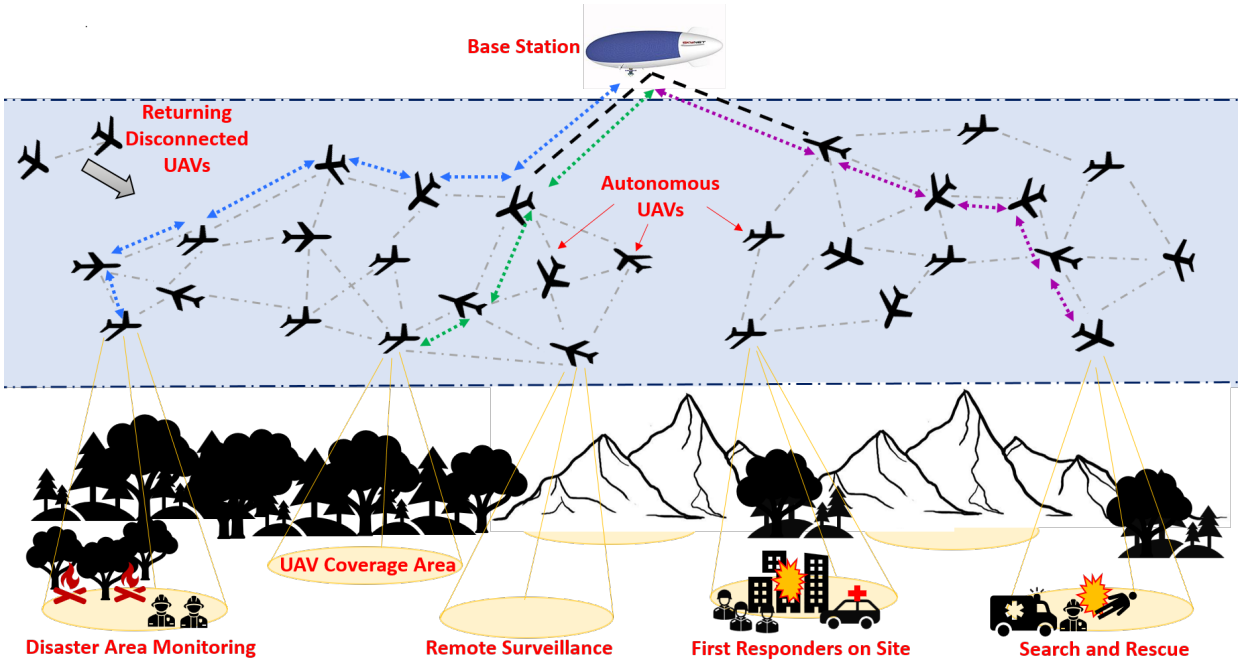



Fig. 1: Decentralized, autonomous UAV network performing monitoring, search and surveillance in inaccessible disaster areas.

the area coverage while ignoring the connectivity among the UAVs.

Many existing schemes for maintaining connectivity in UAV networks are either centralized or use hierarchical architecture or relay nodes [11], [13], [14]. These approaches are plagued by scalability issues and delays, and are vulnerable to attacks (such as in battle-field deployments) or node failures [15]. Other methods (e.g., [1], [16], [17]) modify the UAV mobility model to maintain local connectivity among the nodes. However, few UAV mobility schemes attempt to address both local connectivity (node degree) and BS connectivity in a decentralized, non-hierarchical UAV network [8], [18].

In previous work, we proposed a distributed, connectivity-aware pheromone mobility model (CAP) [19] and a deep Q-learning based CAP model (CAP-DQN) [20] that balance efficient area coverage with a desired node degree in autonomous UAV networks. Importantly however, it did not consider BS connectivity. In this work, we extend our approach to address the problem of achieving fast area coverage while maintaining both local (node degree) and BS connectivity in an autonomous, decentralized, non-hierarchical UAV network.

We make several specific contributions beyond [19], [20]:

- We design a heuristic-based node degree- and BS-connectivity aware pheromone mobility model (BS-CAP), by modifying the CAP scheme [19] trajectory selection and its hello information exchanges to support BS connectivity information.
- We then adapt and improve on our heuristic policy using reinforcement learning. We use deep Q-learning to train a neural network model mapping the UAV's locally available information to each action's expected utility, and use this to select the optimal policy. Compared to the CAP-DQN scheme [20], our model includes additional state information and an updated reward function that accounts for both connectivity criteria.
- We include two new performance measures (*Percentage of Time Connected to BS* and *Giant Component*), compare our method to an additional reference scheme, and evaluate the impact of node failures.
- Since it is not practical to know the complete topology of a decentralized network in real-time, both mobility models work online and are fully distributed, enabling the UAVs to make autonomous decisions in an unknown environment.
- Both proposed models achieve a fast area coverage, strong node degree, high BS connectivity, and robustness to node failures. This provides fast information sensing from the monitored area and delivers the information to the BS in real time.

Paper Organization: We first review several existing UAV mobility models in Section II, followed by an overview of the pheromone-based mobility model on which our approach is based in Section III. We then describe our proposed node degree and BS connectivity aware pheromone (BS-CAP) mobility model in Section IV, followed by our proposed deep Q-learning based BSCAP-DQN model in Section V. We evaluate the performance of our approaches in simulation and discuss the results in Section VI, followed by concluding remarks in Section VII.

## II. Related Work

Several algorithms, such as particle swarm optimization, artificial bee colony, and ant colony optimization, have been proposed to coordinate swarms for various search, rescue, and tracking applications [21]. One widely used approach is the use of stigmergic digital pheromones [1], [11], [12], which act

as spatio-temporal potential fields to coordinate and control the UAV movements. In digital pheromone schemes, information about the pheromone map is communicated between agents in the network through direct or indirect communication. In decentralized UAV networks, each UAV maintains a pheromone map of its neighborhood through direct communication, wherein the pheromone deposits are communicated only locally. Some proposed schemes use a fusion of digital pheromone algorithm and flocking behaviors to coordinate a group of UAVs for performing distributed target search and tracking [1], [16]. These schemes do not maintain BS connectivity.

Decentralized and non-hierarchical UAV networks that modify the mobility model to prioritize BS connectivity include [8], [18]. The ConCov model [8] is a distributed, hybrid mobility scheme that focuses on fast area coverage and BS connectivity. It uses artificial potential fields to minimize overlap in the areas sensed by neighboring UAVs. It also uses local neighbor locations and routing information to tune its performance between area coverage and BS connectivity.

Messous et al. [18] studied BS connectivity in UAV swarms by using a fuzzy inference system to compute a weight for the UAV's connectivity to its neighbors, hop count to the BS, and energy level. However, designing the fuzzy inference system can be difficult for complex systems where the relationships among different inputs are not well understood. The schemes proposed in [17], [22] consider coverage of ground users and reliable network connectivity. However, they do not explicitly consider BS connectivity; instead, they assume that one of the UAVs is connected to the BS (a sink or gateway node). Another related body of literature is connectivity preservation schemes for multi-robot systems using algebraic connectivity [23], [24]. These schemes assume the network is initially connected and do not typically consider BS connectivity.

Another set of approaches uses a centralized or hierarchical network architecture. A centralized scheme was proposed in [14] to position the UAVs to maximize coverage and sensor data acquisition while maintaining BS connectivity, where a topographical map is known beforehand. A clustering approach proposed in [11] uses dual pheromones for target tracking and area coverage, while cluster head (CH) nodes maintain stable network connectivity. Relay UAVs may be deployed as a separate layer to provide continuous BS connectivity in a hierarchical network [13], [25]. However, the selection and placement of relay nodes add additional complexity and require network topology information. Moreover, CH or relay UAVs in hierarchical networks can become bottleneck nodes, leading to delays and congestion, and can be vulnerable to attacks [15]. Our proposed decentralized scheme avoids these issues.

Next, we review reinforcement learning (RL) based schemes for UAV mobility. The highly dynamic and complex nature of UAV networks means that a UAV's actions can have long-term consequences on the system's evolution, making reinforcement learning an appealing framework [26]. However, using multi-agent deep reinforcement learning (DRL) algorithms in UAV networks presents several challenges, such as high dimensional states and observations with increasing node density, which make it difficult to find optimal policies [27]. Moreover, communication range constraints make the environment only partially observed to each UAV. Yue et al. [28] use RL for multi-UAV search of targets but do not consider the connectivity. In [29], a group of rotary-wing UAVs act as aerial BS to provide communication coverage to ground users. A distributed UAV control is designed to maximize communication coverage and fairness and minimize UAV energy consumption via DRL. However, the connectivity and coverage trade-off is not evaluated. In [30], DRL is used for UAV path planning with connectivity constraints. Here, the UAV learns to fly to its destination in the shortest time while maintaining reliable communication to the ground BS in a cellular network.

We use DRL to optimize the trade-off between area coverage and connectivity (node degree as well as BS connectivity) using only local pheromone and connectivity information. In our scheme, we consider the multi-agent RL problem as a single-agent RL scenario, using centralized training with a distributed evaluation approach.

A few UAV mobility schemes consider the impact of node failures due to the energy consumption [18], [29]. These are more important for the rotary-wing UAVs (which have a higher power consumption and lower endurance [31]). Simulations show that our proposed schemes are robust and experience only a limited impact when nodes fail. Our mobility models form a network with built-in redundancy (strong average node degree and BS connectivity). Moreover, UAVs in our network design are not used as CH, gateway or relay nodes and hence the failure of a few nodes does not significantly impact network performance.

### A. ConCov Mobility Model

Given the disadvantages of schemes that use centralized or hierarchical architectures or dedicated relay nodes, we focus on decentralized, non-hierarchical, and autonomous UAV network architectures. There are relatively few such distributed and connectivity-aware UAV mobility models in the literature; we compare our proposed schemes against the ConCov model [8], which uses a modified flocking behavior that can trade off between coverage and BS connectivity.

Separate models for area coverage (for target detection) and BS connectivity (for notification of targets to BS) were studied in [3]. Later, in [8], the author presented the ConCov model which is a distributed, hybrid mobility scheme that focuses on fast area coverage and BS connectivity. The ConCov model utilizes artificial potential fields to minimize the overlap in the area sensed by neighboring UAVs. It also uses local neighbor locations and routing information to tune its performance between area coverage and BS connectivity. The heading of a UAV is given by,

$$\vec{R_v} = \omega \frac{\vec{R_v^{cov}}}{|\vec{R_v^{cov}}|} + (1-\omega)\frac{\vec{R_v^{con}}}{|\vec{R_v^{con}}|} \quad (1)$$

where $\vec{R_v^{cov}}$ is a "repellent" force to reduce the coverage overlap, $\vec{R_v^{con}}$ is an "attraction" force to prevent disconnections from the BS, and $\omega$ is a weight to balance the two forces.

Each UAV uses its current and next positions, the headings of neighboring UAVs, the resulting network graph, and the route to BS to calculate the two forces.

The repellent force, $\vec{R_v^{cov}}$, is defined by a combination of the force vector $\vec{F_{ii}}$ along the current heading direction $\theta_i^c$, and the sum of the repel force vectors $\vec{F_{ij}}$ from its neighbors $j \in N_i^s$ along $\theta_{ij}$. The $\hat{\theta}$ notation is used to represent a unit vector with direction $\theta$.

$$\vec{R_v^{cov}} = \vec{F_{ii}} + \sum_{j \in N_i^s} \vec{F_{ij}} \tag{2}$$

$$\text{where,} \quad \vec{F_{ii}} = \frac{1}{r}\hat{\theta_i^c} \quad \text{and} \quad \vec{F_{ij}} = \frac{1}{d_{ij}}\hat{\theta_{ij}} \tag{3}$$

while $r$ is the UAV's sensing range and $d_{ij}$ is the distance between UAVs $i$ and $j$.

The attraction force, $\vec{R_v^{con}}$, is applied to a UAV $i$ to maintain BS connectivity [3]. If UAV $i$ remains connected to BS after the sensing period $ts$, $\vec{R_v^{con}} = \hat{\theta_i^c}$. If UAV $i$ becomes disconnected from the BS after $ts$, then

$$\vec{R_v^{con}} = \vec{J_{ii}} + \vec{J_{ik}}, \tag{4}$$

$$\text{where} \quad \vec{J_{ii}} = \hat{\theta_i^c} \quad \text{and} \quad \vec{J_{ik}} = \hat{\theta_{ik}}. \tag{5}$$

Here, $\vec{J_{ii}}$ is the vector along the current direction $\theta_i^c$ and $\vec{J_{ik}}$ is the vector from UAV $i$ to its 1-hop neighbor $k$ that has a route to the BS. The ConCov algorithm is described in Pseudocode 1.

**Pseudocode 1:** ConCov

```
At each UAV i with current direction θ_i^c;
if time interval of ts has elapsed since last direction change then
    // Update UAV's direction
    Compute repulsive force, R_v^cov from (2);
    Compute attractive force, R_v^con;
    if there exists a route to BS from UAV i after a time interval of ts then
        R_v^con = θ̂_i^c;
    else
        //No route to BS exists
        R_v^con = J_ii + J_ik from (4);
    end
    Compute resultant force R_v from (1);
    // New UAV direction
    θ_i^c = ∠R_v
else
    Maintain current direction θ_i^c;
end
```

## III. Overview of Pheromone Mobility Model

The pheromone mobility model uses repel digital pheromones to promote exploration and fast coverage of an area with no prior information [1], [12], [19]. A digital pheromone has characteristics modeled after natural pheromones, such as deposition, evaporation and diffusion, with values stored in a digital pheromone map. We assume the UAVs move in a two-dimensional space (i.e., at constant altitude) to search a given area; although the UAVs' positions and trajectories are continuous-valued, to represent the pheromone map we divide the area into a grid of $C^2$ cells, each identified by its $(x, y)$ coordinates.[1] Each UAV moves towards the cells with minimum repel pheromone value and deposits a repel pheromone of magnitude '1' in the cells scanned along its trajectory. After a UAV deposits pheromone in a cell $(x, y)$, it is progressively diffused to the surrounding cells at a constant diffusion rate $\psi \in [0, 1]$. This encourages UAVs to disperse towards unvisited cells. The pheromone value of each cell also evaporates, decreasing its intensity over time by a constant rate $\lambda \in [0, 1]$. This evaporation allows the UAVs to revisit already scanned cells after some time gap, for example if environment or target locations change over time [19].

Mathematically the pheromone value $p_{(x,y)}$ in a cell $(x, y)$ at time $t$ is described as [1], [19],

$$p_{(x,y)}(t) = (1-\lambda) \cdot [(1-\psi) \cdot p_{(x,y)}(t-1) + \partial p_{(x,y)}(t-1,t) + \partial d_{(x,y)}(t-1,t)] \tag{6}$$

where $(1-\psi) \cdot p_{(x,y)}(t-1)$ is the pheromone value remaining in cell $(x, y)$ after diffusion to the surrounding cells, $\partial p_{(x,y)}(t-1,t)$ is the new pheromone value deposited in the update interval $(t-1,t)$, and $\partial d_{(x,y)}(t-1,t)$ is the additional pheromone diffused to the current cell from its eight surrounding cells in the update interval $(t-1,t)$, which is described as [1], [19],

$$\partial d_{(x,y)}(t-1,t) = \frac{\psi}{8} \cdot \sum_{a=-1}^{1} \sum_{b=-1}^{1} p_{(x+a,y+b)}(t-1) \tag{7}$$

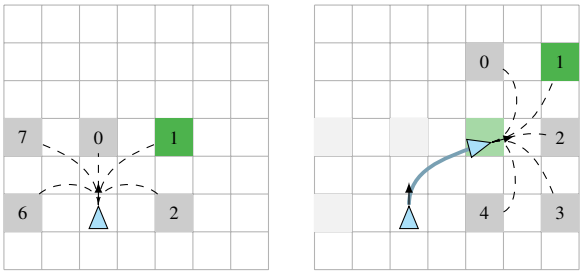
In a decentralized UAV network, the UAVs exchange their digital pheromone maps with their 1-hop neighbors by using the periodic 'hello messages' [19].

## IV. Node Degree and BS Connectivity Aware Pheromone Mobility Model (BS-CAP)

Pheromone mobility models achieve fast area coverage by pushing the UAVs away from each other and recently visited cells. However, this can lead to a weak node degree and BS connectivity due to the limited transmission range of the UAVs, especially when the UAV density is low. Maintaining strong node degree is required in a decentralized network to ensure robust network communication and coordination among UAVs. Similarly, strong BS connectivity enables robust transmission of information from UAVs to the BS.

In this section, we describe our proposed BS-CAP mobility model for decentralized, autonomous UAV networks to balance strong average node degree and BS connectivity with fast area coverage. Our model uses a weighted combination of the repel pheromone value (see Section IV-A and IV-B) and node degree (see Section IV-C) at the cells, along with 'route availability to BS' information, to determine a UAV's trajectory (see Section IV-E). Since it is not practical to know the complete topology of such a network in real-time, the proposed scheme is fully distributed and relies only on neighborhood information (see Section IV-D).

[1]For convenience, we use the same discrete grid to track our area coverage (in our experimental evaluations), and to define the waypoints used in our trajectory selection policies.




(a) (b)

Fig. 2: UAV headings and next-waypoints. (a) Given its current heading, a UAV selects one (green) of the five forward-facing cells out of its eight possible next-waypoint cells (gray). (b) Arriving at this waypoint, the UAV selects its new next-waypoint; discretizing its current heading, waypoint 2 is the closest to straight, so that its forward-facing options are $\{0, 1, 2, 3, 4\}$.

### A. Next-Waypoint

As the UAVs fly over the monitored area, we use the pheromone map grid to determine where each UAV should fly next. The location on the grid toward which a UAV decides to fly is called the 'next waypoint'; the distance between the current and next-waypoints is a function of the UAV speed. We restrict our choice of waypoints based on the flight trajectory constraints of a fixed-wing UAV, giving smooth turn trajectories [19]. Concretely, the current (continuous) heading of each UAV (0 to 360 degrees) is discretized into 8 possible directions, each corresponding to a possible next-waypoint a distance away. However, for our trajectory, we only consider up to five "forward-facing" next-waypoints: the one along the current (discretized) heading, and two each to the left and right (corresponding to sharper or more gradual turns). Fig. 2 illustrates an example of a UAV's next-waypoint cells: initially, the UAV has a current (discretized) heading of '0', and moves to cell '1' out of the five possible next-waypoint cells (6, 7, 0, 1, 2). Upon reaching cell '1', the UAV has current (discretized) heading direction of '2'; it then moves next-waypoint cell '1' out of the possible options (0, 1, 2, 3, or 4).

### B. "Look-Ahead" Pheromone Value

In a basic pheromone mobility model, each UAV moves towards an unvisited region in the map by selecting the cell with the minimum repel pheromone value. In contrast, our scheme computes a 'look-ahead pheromone' value, $P' \in [0, 1]$, for each potential next-waypoint cell, equal to a weighted average of the repel pheromone at that cell and its eight 1-hop neighbors. Concretely, for next-waypoint cell $(x, y)$,

$$P'_{(x,y)} = \frac{1}{12} \cdot \left(3 \cdot P_{(x,y)} + \sum_{a,b \in \{-1,0,1\}} P_{(x+a,y+b)}\right) \quad (8)$$

where $P_{(x',y')} \in [0, 1]$ is the pheromone value in cell $(x', y')$ [19].

Moving to cells with minimum $P'$ value, the UAV is more likely to steer toward unvisited regions of the map, increasing the area coverage.

### C. Distance-Weighted Node Degree

The node degree of a UAV represents the number of its 1-hop neighbors. Since UAV pairs that are close to their maximum transmission range are more likely to lose their connection in the near future, we calculate a **distance-weighted connectivity** $(\gamma_{uv})$ between two UAVs $u$, $v$ as a function of their Euclidean distance $d_{uv}$ and transmission range $T_x$. It is defined as [19],

$$\gamma_{uv} = \begin{cases} 1 & d_{uv} \leq (0.6 \cdot T_x) \\ 2.5(1 - \frac{d_{uv}}{T_x}) & (0.6 \cdot T_x) < d_{uv} \leq T_x \\ 0 & d_{uv} > T_x \end{cases} \quad (9)$$

The **distance-weighted node degree** $K_u$ of UAV $u$ is defined as the sum of its $\gamma_{uv}$ over its $\mathcal{N}$ 1-hop neighbors [19]:

$$K_u = \sum_{v \in \mathcal{N}} \gamma_{uv}. \quad (10)$$

Intuitively, we set $\gamma_{uv} = 1$ when the distance between UAVs is within 60% of the transmission range, because the probability they will remain connected is high. Beyond 60%, the value $\gamma_{uv}$ decreases linearly with distance $d_{uv}$, discounting more distant neighbors that are less likely to remain connected in the near future.

### D. Distributed Information Exchange using 'Hello Messages'

To share information in the distributed network, a Hello message containing each UAV's updated local information is propagated to its 1-hop neighbors. The pheromone value and connectivity information of a UAV's neighbors is obtained from the Hello messages and used to select the next-waypoint cell. In our scheme, each UAV exchanges Hello messages with its 1-hop neighbors every 2 seconds. Each Hello message is 24 bytes, and consists of the UAV Id (7 bits), its current location (18 bits with 10 m resolution), next-waypoint cell (12 bits for the cell id, assuming 60 x 60 cells of 100 m x 100 m resolution in a 6 km x 6 km area), and a local pheromone map (6-bit pheromone values in the 5 x 5 cells centered at the UAV's current cell, for 150 bits total). The Hello packet of a UAV also includes the hop count of the shortest route to BS (4 bits). Thus, every UAV can compute the node degree $K_i$ and availability of a route to BS at the next-waypoint cell $i$.

### E. Next-Waypoint Selection

Of the five possible next-waypoint cells, we consider only cells where a UAV can maintain a route to the BS, and select a cell $i$ with maximum score $W_i$, where:

$$W_i = \begin{cases} \alpha_i(1 - P'_i) & \text{if } \exists \text{ route to BS} \\ 0 & \text{if } \nexists \text{ route to BS} \end{cases} \quad (11)$$

where $P'_i$ is the 'look-ahead pheromone' value from (8) at next-waypoint cell $i$, calculated using the UAV's own pheromone map and pheromone information received from its neighbors' Hello messages. The value $\alpha_i \in [0, 1]$ is the UAV's normalized $K_i$ at cell $i$:

$$\alpha_i = \begin{cases} \frac{K_i}{\beta} & K_i \leq \beta \\ 1 & \beta < K_i \leq \beta' \\ 1/3 & K_i > \beta' \end{cases} \quad (12)$$

We compute $K_i$ from (10) using the heading information of the UAV's 1-hop neighbors in their most recent Hello messages. Selecting the values $\beta$, $\beta'$ in (12) allows the designer to tune the connectivity and coverage balance in our model. For example, selecting a small $\beta$ (e.g., $\beta$ = 0.5) results in a model with faster area coverage at the cost of connectivity (node degree). On the other hand, a larger $\beta$ (e.g., $\beta$ = 2.5) results in better node degree at the cost of a higher coverage time. Intuitively, UAVs with a $K_i = \beta'$ are considered to have strong node degree; even higher values of $K_i$ are discouraged by decreasing $\alpha$ when $K_i > \beta'$ in (12). This is desirable behavior since too-high node degrees correspond to congregations of UAVs, which can degrade the area coverage performance at low to moderate node densities.

Each UAV maintains connectivity by selecting the next-waypoint with maximum $W_i$ as per (11), provided a route exists to the BS. If no route from the UAV to the BS exists at any of the 5 possible next-waypoint cells, we select the next-waypoint closest to that UAV's 1-hop neighbor which does have a route to the BS. The next-waypoint selection process of a UAV is described in Pseudocode 2.

**Pseudocode 2:** UAV Next-Waypoint Cell Selection

```
if UAV reaches next-waypoint cell then
    // Deposit repel pheromone
    Add repel pheromone value = 1 for the current cell in its
      digital pheromone map;
    // Select a new next-waypoint cell (i)
    Identify the next-waypoint cells (out of 5 possible cells),
      where the UAV can maintain a route to BS;
    if there exists a route to BS from at least one of the
      next-waypoint cells then
        Calculate the 'look-ahead pheromone'(P_i') value for
          each of the next-waypoint cells using (8);
        Calculate estimated node degree K_i of the UAV at
          each of the next-waypoint cells via (10) ;
        Calculate the W_i value for each of these
          next-waypoint cells using (11);
        Select cell i = arg max_i W_i as the UAV's
          next-waypoint;
    else
        //No route to BS exists from the next-waypoint cells
        Select a next-waypoint cell i closest to the UAV's
          1-hop neighbor which has a route to BS.
    end
else
    // Follow smooth-trajectory towards the selected
      next-waypoint cell
end
```

All values required for the calculation of $P_i'$ and $\alpha_i$ are received from the neighbors through Hello messages. Computation of $P_i'$ at a next-waypoint involves 10 additions and 2 multiplications. Similarly, the computation of $\alpha_i$ involves a maximum of 11 multiplications and 5 additions, assuming 5 1-hop neighbors. Therefore, the computation of $W_i$ for each next-waypoint cell in (11) requires 16 additions and 14 multiplications. Since our scheme uses up to 5 possible next-waypoint cells, at most 80 additions and 70 multiplications may be required to select the best next-waypoint.

## V. Deep Q-Learning based BS-CAP (BSCAP-DQN) Mobility Model

In the BS-CAP model, we use local connectivity information (node degree) to augment the pheromone information and allow trade-offs between coverage and connectivity in our network, while information on the availability of routes to the BS from the next-waypoint cells is used to maintain BS connectivity. In this section, we use the framework of reinforcement learning (RL), in particular, deep Q-Learning (DQN) [32], [33] to train a policy (BSCAP-DQN) for the UAVs that *explicitly* optimizes a trade-off between coverage, node degree and BS connectivity. RL [34] attempts to optimize an agent's policy, represented as a distribution over actions given the current state, in order to maximize a (discounted) cumulative reward to that agent over time. So-called "online" RL training methods use agent's current policy to decide what actions to take, influencing the evolution of the system and subsequent experiences used for training. [2] Typically, the experience is gathered by following an "exploration policy" which is related to, but may be more random than, the model's current estimate of the best policy; for example, $\epsilon$-*greedy* policies mostly follow the currently estimated best action, but take a random action with probability $\epsilon$ instead. "Offline" training [35], in contrast, gathers and stores experience by following some (possibly unknown) policy, which can then be used for training the agent's policy [20].

In practice, since the evolution of our system depends on the behavior of multiple UAVs (all following the same policy and using limited state information), it is important to use online training so that changes to the policy can propagate to affect the experience and outcomes observed by other UAVs. However, starting with a purely random policy leads to slow learning since the UAVs must spend time to learn the rewards associated with exploration. Thus, to make training more efficient, we first perform pre-training using offline DQN on a database of experience obtained by following an $\epsilon$-greedy exploration variant of our fixed BS-CAP policy from Section IV. After this pre-training, we perform online training using DQN with experience replay [32] to further refine the agents' policy. Our implementation uses standard modeling and training libraries from PyTorch [36]. Fig. 3 illustrates the use of offline pre-training followed by online reinforcement learning [20].

Since our system consists of multiple UAVs, it can be considered an example of multi-agent reinforcement learning [27]. However, the structure of our problem constrains the nature of our agents. In "centralized" multi-agent RL, the agents coordinate through a central decision-maker that incorporates all agents' observations and determines a set of actions jointly. However, large numbers of agents lead to an exponential explosion in the size of the observation and action spaces. Moreover, we would like our agents to continue operating

[2] Although the training method is called "online RL", our policy is fully trained in simulation beforehand, and held fixed during the actual UAV deployment. The short lifetime of the UAVs does not allow for significant exploration of policies during any single deployment. In this sense our DQN policy defines a comparable but explicitly optimized alternative to the BS-CAP policy.

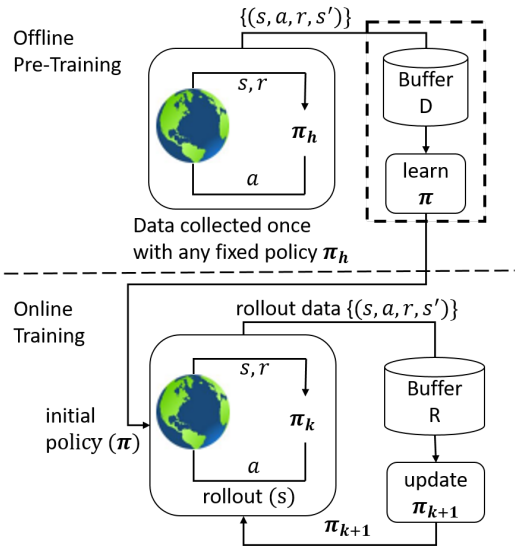



Fig. 3: We use offline pre-training, followed by online training, to improve training efficiency while accounting for the effect of policy changes in our multi-UAV setting [20].

even when out of contact with one another, making centralized reasoning unsuitable.

"Concurrent" multi-agent RL allows each agent to learn its own individual policy. This allows, for example, agents to learn complementary policies that perform different roles in order to achieve a shared objective. However, in our setting, we prefer to select a homogeneous policy for the agents (i.e., each UAV follows the same behavior given its local observations), since we would like to apply our policy in settings where there may be more or fewer UAVs than during training.

Our approach is thus an example of "parameter-sharing" multi-agent RL, in which the agents share the parameters of a single policy, which is trained using the experiences of all agents simultaneously (based on each agent's individual observations). Compared to other multi-agent approaches, this method is highly scalable, and can use a variety of training techniques, including DQN and deep deterministic policy gradient methods [33].

### A. DQN Problem Formulation

In DQN, we model the state-action value function $q(s, a)$ using a neural network. The function $q$ captures the long-term value associated with being in a state $s$ and taking action $a$. Given a current estimate of $q$, our agent can select its next action $a$ to maximize $q(s, a)$ given its current state $s$.

In our setting, each UAV moves from its current waypoint to its selected next waypoint. Upon reaching the next-waypoint, the UAV must choose a new next-waypoint from a set of five possibilities based on the "look-ahead" pheromone ($P'$), distance-weighted node degree ($K$), route availability to BS at those next-waypoints, normalized distances from the next-waypoint cells to the UAV's 1-hop neighbor that has the shortest route to BS, the normalized distance from the UAV to BS, and the node degree of BS. Thus our *input state* $s$ consists of 22 real values. Note that the $P'$ and $K$ help to learn a policy with faster map coverage and better node degree, while route availability to BS at next-waypoint cells, normalized distances from the next-waypoint cells to the UAV's 1-hop neighbor with a route to BS, the normalized distance from the UAV to BS, and the node degree of the BS help in learning to maintain strong BS connectivity.

Our *action space* consists of five possible actions corresponding to the five forward-facing directions (next-waypoint cells) with respect to the current UAV heading.

Our RL setting also requires us to define a set of rewards $r$, which are obtained when the agent takes action $a$ from state $s$. We provide three sources of reward: one for area coverage ($r_c$), one for node degree ($r_k$) and one for BS connectivity ($r_b$). The total reward ($r$) is defined as,

$$r = m \cdot r_c + r_k + n \cdot r_b \tag{13}$$

Here, the coverage reward, $r_c$, is defined as the difference between the number of new cells scanned and the number of already-scanned cells along the path taken by the UAV between its two waypoint cells. This favors the paths that improve rapid coverage of the environment assuming that the UAVs scan any cells as they pass overhead. Our node degree reward, $r_k$, is calculated based on $K_i$ of the UAV when it reaches its next waypoint $i$; specifically,

$$r_k = \begin{cases} -1 & 1 < K_i \leq 2 \\ 0 & 2 < K_i < 3 \\ -4 & \text{otherwise.} \end{cases} \tag{14}$$

$K_i \leq 2$ incurs a penalty to avoid very low node degree. A value of $2 < K_i < 3$ gives adequate node degree. $K_i \geq 3$ incurs a penalty to avoid very high node degree that causes multiple UAVs to congregate, which can degrade the area coverage performance.

The BS connectivity reward, $r_b$ is given for maintaining connectivity to the BS:

$$r_b = \begin{cases} 0 & \text{if } \exists \text{ route to BS} \\ -3 & \text{if } \nexists \text{ route to BS} \end{cases} \tag{15}$$

The model's preferred balance between area coverage and BS connectivity can be changed by varying $m$ and $n$ in the total reward function in (13). Q-learning then optimizes the agent's policy to maximize the expected sum of rewards, discounted by a geometric weighting $\gamma^k$ for rewards $k$ steps in the future. For our environment, we use discount factor $\gamma = 0.9$.

Ideally, we would train our reinforcement agent to directly optimize the performance objectives used to evaluate in Section VI-A. However, these evaluation metrics are holistic, global run-based quantities that are difficult to use as reward functions due to their long feedback delays, and generating a sufficiently large number of full system trajectories is computationally prohibitive. For these reasons, we train our agent using local rewards at each step that provide immediate feedback toward behaviors that benefit our overall objectives, learning a good policy faster with fewer runs. One potential direction for future research is to develop local reward structures that could more closely align with our desired holistic objectives to improve BSCAP-DQN's performance.

TABLE I: Neural Network Description

| Layers | Layer type | Size | Activation function |
|---|---|---|---|
| Input layer | - | (22) | - |
| Hidden layer 1 | Fully connected | (24) | Leaky ReLU |
| Hidden layer 2 | Fully connected | (16) | Leaky ReLU |
| Output layer | - | (5) | - |

Our DQN neural network consists of two dense, fully connected hidden layers with Leaky ReLU activation functions of size 24 and 16 hidden nodes, respectively. The input layer takes in the length-22 state vector $s$, and the output layer corresponds to 5 Q-function values, i.e., $Q(s,a)$ for each possible action $a$, as described in **Table I**. During development, we experimented with a number of other similar network architectures but concluded that this two-layer network performed sufficiently well and used it in all reported experiments.

### *B. Offline Pre-training Using BS-CAP*

Like in [20], an offline static dataset $D$ for pre-training our DQN is generated using a randomized exploration version of our BS-CAP model policy, $\pi_h$. Since our BS-CAP policy is deterministic, we alter it to take a uniformly-at-random action with some probability $\epsilon = 0.1$ to ensure that the learner can see the outcome of actions that would not normally be followed by BS-CAP. We build a dataset consisting of state ($s$), action ($a$), reward ($r$), and next-state ($s'$) transition sequences from 10,000 episodes under simulation settings as described in Section VI. Each episode is run for 2000 seconds.

For pre-training, we initialize the DQN with weights using the default ("kaiming_uniform") PyTorch weight initialization. We then train on our offline dataset for $N = 50$ epochs (passes through the data), using stochastic gradient descent with mini-batch size $M = 1024$, which are randomly sampled to avoid temporal correlation. We implement our model in PyTorch and use the SGD optimizer with a decaying learning rate initialized to 0.0001 to optimize the squared error, $\big(Q(s_t,a_t) - y_t\big)^2$, where $y_t$ is the Bellman target, $y_t = r_t + \gamma \max_{a'} Q(s'_t, a'))$. The DQN pre-training process is given in Pseudocode 3.

**Pseudocode 3:** Offline DQN Pre-Training

```
Initialize Q-network with random weights θ;
Initialize target network Q̂ with weights θ̂ = θ;
Load Offline Experience D = {(s_t, a_t, r_t, s'_t)};
Initialize minibatch size M, total epochs N;
t=1,U=3000;
for epoch = 1 ... N do
    Partition D into minibatches {B_i} of size M = 1024 at random
    for each B_i do
        t=t+1;
        for each (s_t, a_t, r_t, s'_t) ∈ B_i do
            Calculate targets, y_t = r_t + γ max_a' Q(s'_t, a'; θ̂);
        end
        Calculate MSE loss on minibatch B_i: L_i = (1/M) Σ_t (y_t − Q(s_t, a_t; θ))^2;
        Take a gradient step on θ at rate α;
        if t mod U == 0 then
            Copy Q-network to target network: θ̂ = θ;
        end
    end
    Update learning rate scheduler α;
end
```

### *C. Online Training*

Like in [20], after pre-training, we simulate additional environmental interactions while UAVs follow an $\epsilon$-greedy exploration version of the current DQN policy. We update a single, universal policy from transition sequences obtained from all agents and stored in a replay memory. Samples from the replay memory are then used to update the DQN's value function $Q$. Online training ensures that changes to the UAV policy are reflected in the behavior of the other UAVs as well, so that the policy will be optimized to perform well within a network of identical agents.

We train the DQN online for a total of $N'$= 4000 episodes and stop each episode at 2000 seconds. During each episode, the transition sequences $(s,a,r,s')$ of individual UAVs are stored into a replay memory of 10,000 transitions. We update our Q-network after every $B' = 30$ UAV steps (i.e., transitions saved to replay memory) by sampling a minibatch of size $M' = 512$ at random from the replay buffer. During online training, we use the ADAM optimizer to adaptively select the learning rate (initialized to 0.0001), and estimate our target values $y_t$ using a *target network* $\hat{Q}$ [32], i.e., $y_t = r_t + \gamma \max_{a'} \hat{Q}(s'_t, a'))$, where $\hat{Q}$ is a periodically updated copy of $Q$, updated after every $C = 100$ gradient steps. The online DQN training process is shown in Pseudocode 4.

**Pseudocode 4:** Online DQN Training

```
Initialize θ using offline trained network weights;
Initialize target network Q̂ with weights θ̂ = θ;
Initialize replay memory R; t = 1;
for episode = 1 ... N' do
    while time ≤ 2000s do
        UAV selects action a_t via ε-greedy[Q(s_t, a_t; θ)];
        Execute action a_t, and observe reward r_t and next state s'_t;
        Store (s_t, a_t, r_t, s'_t) in R;
        t = t + 1;
        if t mod B' == 0 then
            Sample random minibatch of M' = 512 transitions (s_j, a_j, r_j, s'_j) from R;
            Calculate target Q values, y_j = r_j + γ max_a' Q̂(s'_j, a'_j; θ̂);
            Calculate MSE Loss, L = (1/M') Σ_{j=0}^{M'-1} (y_j − Q(s_j, a_j; θ))^2;
            Take a gradient step on θ using ADAM;
            if t mod B' · C == 0 then
                Copy Q-network to target network: θ̂ = θ;
            end
        end
    end
end
```

Hyperparameters such as episode value, batch size and gradient steps were selected based on empirical trials on small systems and then used in the larger training process. Although

training can be slow, it is accomplished beforehand and does not affect the efficiency of the resulting policy. For the neural network in **Table I**, evaluating the trained DQN policy requires about 992 multiplications and 907 additions to compute the best next-waypoint among the five possibilities. Since the next-waypoint is computed only periodically, the DQN adds little energy consumption or device complexity.

## VI. Simulation Results and Discussion

The performance of our proposed BS-CAP and BSCAP-DQN models are compared against the repel pheromone and ConCov [8] models, discussed in Sections III and II-A, respectively. The UAV network is simulated in Python3, and Table II shows the simulation parameters.

As in our introduction's scenario, we consider a swarm of 30 to 50 low SWaP fixed-wing UAVs (e.g., [4]–[7]) for monitoring a 6 km $\times$ 6 km area where fixed communication infrastructure, such as a cellular network, is not available. The aerial BS is located at the bottom center of the map, and the UAVs are launched from the vicinity of the BS. The UAVs are equipped with GPS and have a transmission range of 1 km (e.g., [4]). We assume a multihop topology with free space propagation (e.g., [10]). For simplicity, the UAVs are assumed to be point masses, and their mobility is limited to the X-Y plane flying at a constant altitude (typically from 200m to 1km above the ground [5]–[7]). UAVs perform collision avoidance through trajectory modifications. To facilitate representing the pheromone map, waypoints, and coverage statistics, the area is divided into grid cells of 100 m $\times$ 100 m each, consistent with other literature [1], [8], [20], [37], [38]. A UAV scans the cell in which it currently resides and deposits a repel pheromone of magnitude 1. We use pheromone evaporation $\lambda$ and diffusion $\psi$ rates of 0.006 each. Each simulation is run for 3000 s, and performance parameters are averaged over 30 simulation runs to study the coverage and connectivity behavior of the network at different stages.

TABLE II: Simulation Parameters

| Parameters | Values |
|---|---|
| Simulation Time | 3000 s |
| Map Area | 6 km $\times$ 6 km |
| Cell Size | 100 m $\times$ 100 m |
| Transmission Range | 1 km |
| Number of UAVs | 30, 50 |
| UAV Speed | 20 m/s, 40 m/s |
| Evaporation Rate | 0.006 |
| Diffusion Rate | 0.006 |
| BS Location | Bottom center of map |
| Number of Runs | 30 |

Both our models and ConCov [8] contain parameters that control the balance of coverage versus connectivity. In our experiments, we vary these parameters to produce behaviors with differing emphases. For ConCov, we select values of $\omega \in (0, 1)$ in Eq. (1), which emphasizes connectivity as $\omega$ decreases; we use a sensing period of 5 s to update the UAV headings. In our BS-CAP model, we vary $\beta \in [0.5, 2.5]$ in Eq. (12), where increasing $\beta$ results in a higher node degree; there is no explicit tuning parameter for BS connectivity. In our BSCAP-DQN model, the balance between coverage and connectivity, including the BS connectivity, can be varied by setting $n \in [1, 4]$ in the reward function (13).

### A. Performance Metrics

We measure the behavior and performance of our UAV swarm using a number of summarizing statistics. The *coverage* properties of the swarm are measured by:

- *Coverage ($C_v$)*: The average percentage of cells in the map visited by UAVs at least once within a given duration of time; we prefer a higher $C_v$ in a given duration of time.
- *Coverage Time ($Tc$)*: The average time taken to scan 90% of cells in the map. A lower value of $Tc$ indicates faster area coverage.
- *Coverage Fairness ($F$)*: Represents how equally all the cells of the map are visited during a given time period, as measured by Jain's fairness index [39],

$$F = \frac{(\sum_i x_i)^2}{n \sum_i x_i^2} \tag{16}$$

where $x_i$ is the number of scans of cell $i$, and $n$ is the total number of cells in the map. A higher value of $F$ is desired.

The *connectivity* properties are measured by:

- *Number of Connected Components ($NCC$)*: Average number of disjoint components in the UAV network (with no path between them), sampled every 10 s. NCC measures how disconnected the network is; its optimal (minimal) value is 1.
- *Average Node Degree ($AND$)*: The node degree of a node $u$ ($N_D(u)$) is its number of links (or 1-hop neighbors). $AND$ is the average node degree of all $V$ nodes in the network, computed by sampling the network every 10 s.

$$AND = \frac{\sum_u N_D(u)}{V} \tag{17}$$

- *Percentage of Time Connected to BS ($T_{bs}$)*: Average percentage of time a UAV is connected, directly or indirectly (through a multihop path), to the BS throughout the simulation.
- *Giant Component ($G$)*: Average size of the connected subgraph (component) with the largest number of nodes in the network; larger $G$ indicates that fewer UAVs are isolated.

### B. Results and Discussion

We evaluate the coverage ($T_c$, $F$) and connectivity ($NCC$, $AND$, and $T_{bs}$) statistics of three connectivity-aware models: BS-CAP, BSCAP-DQN, and ConCov, as well as a basic pheromone model, under four different conditions: densities of 30 or 50 UAVs, at speeds of either 20 m/s or 40 m/s. In general, better connectivity comes at the cost of slower coverage; by varying parameters in each scheme, we can trade off between these quantities. We select three parameter settings for each connectivity-aware method: in BS-CAP, we take $\beta \in \{0.5,\ 1.5,\ 2.5\}$ and $\beta' = 3$; in BSCAP-DQN, we

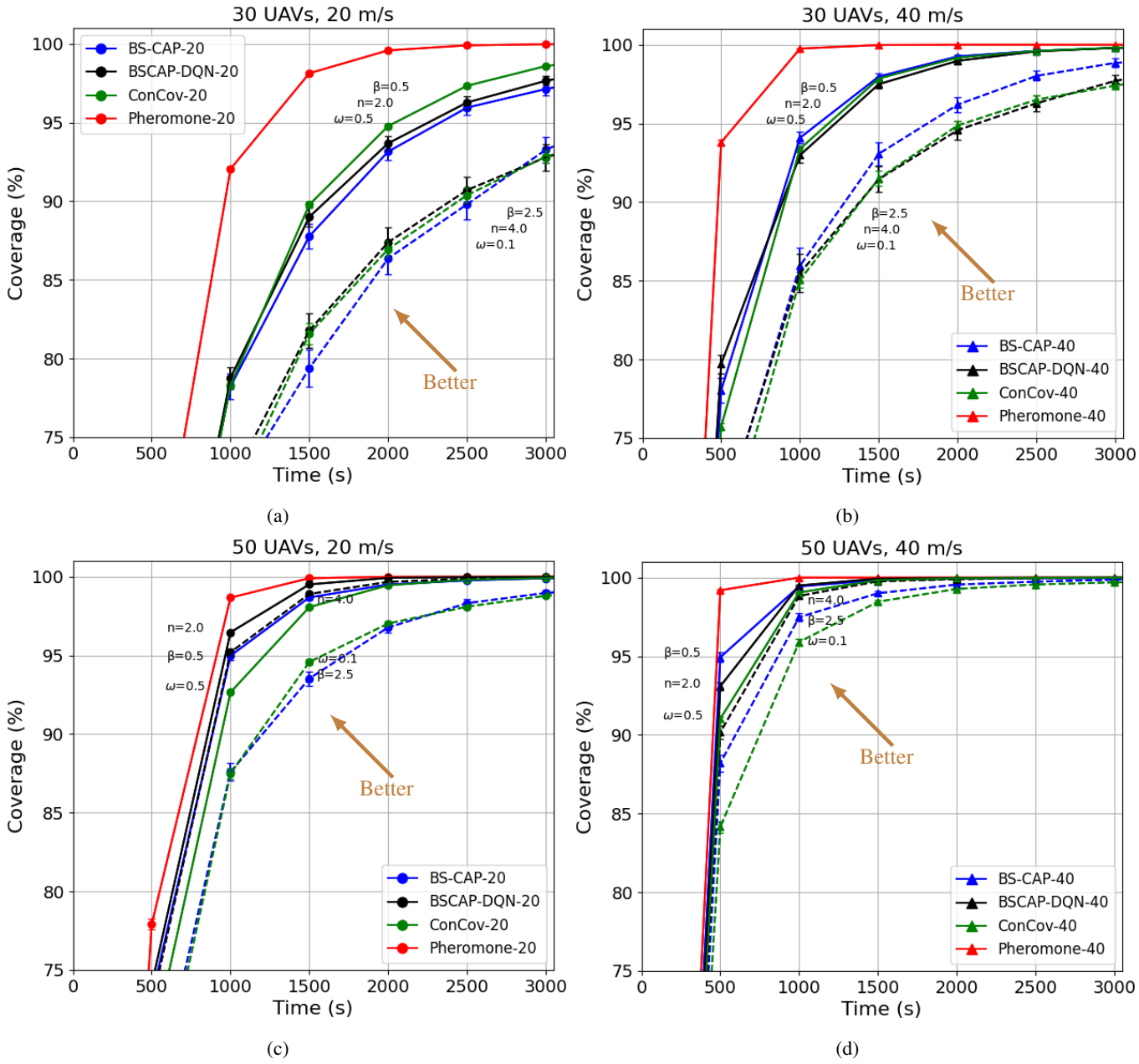
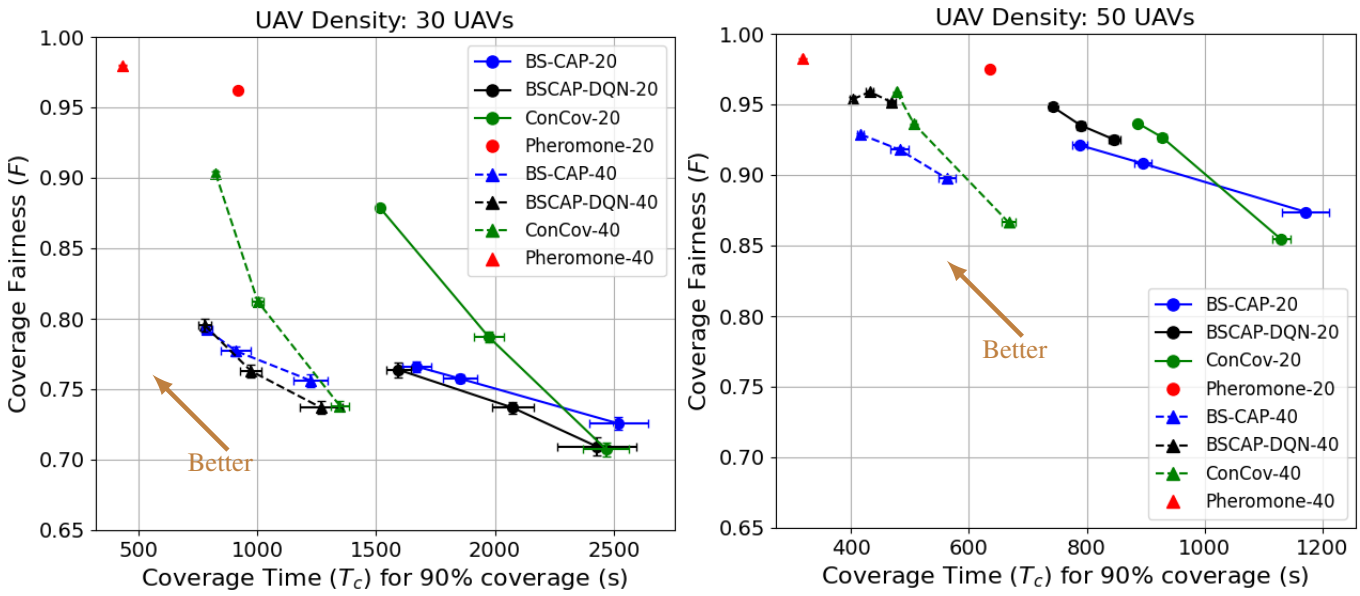



Fig. 4: Coverage vs. Time plots for 30 and 50 UAVs at 20 m/s and 40 m/s. While more coverage in less time is preferred, we experiment with several parameter settings to balance coverage with connectivity (illustrated in subsequent plots); the parameter values are chosen to produce similar coverage curves among the three tested algorithms (BS-CAP, BSCAP-DQN, and ConCov). The intermediate parameter settings ($\beta = 1.5$, $n = 3$, and $\omega = 0.3$) are omitted from these plots for clarity.

take $m = 3$ and $n \in \{2,\ 3,\ 4\}$; and in ConCov, we take $\omega \in \{0.5,\ 0.3,\ 0.1\}$. Figures 4-8 show performance plots for each method, on average over the 30 runs, with error bars representing the standard error of the averages. Suffix "-20" (e.g., BS-CAP-20, ConCov-20) indicate results at 20 m/s, while "-40" indicates UAVs at 40 m/s speeds.

*1) Coverage Performance:* The **Coverage vs. Time plots** in Figure 4 represent the total map area covered in a given time. We illustrate each algorithm with its largest and smallest parameter settings, omitting the middle values to reduce clutter. As expected, increasing $\beta$ or $n$, or decreasing $\omega$ in their respective models increases the emphasis on connectivity over coverage performance. Increasing node density and/or speed leads to a faster coverage of a given area for all models. By design, at our parameter settings we have relatively similar coverage profiles among the three methods; the basic pheromone model gives the fastest coverage (with no consideration for connectivity). Their comparable coverage profiles allow us to see clear distinctions between the methods when comparing their performance under the other metrics, illustrated in Figures 5–7.




(a) Coverage Fairness

(b) Coverage Fairness

Fig. 5: Coverage Fairness performance plots for 30 and 50 UAVs, at 20 m/s and 40 m/s. (a) For 30 UAVs, ConCov achieves higher $F$ values than BS-CAP and BSCAP-DQN at lower $T_c$ values. The $F$ values of all the models converge when $T_c$ increases. (b) For 50 UAVs, BSCAP-DQN achieves higher $F$ values than BS-CAP and ConCov at lower $T_c$ values.

The ***F* vs. $T_c$ performance plots** for 30 UAVs and 50 UAVs at both speeds are shown in Figures 5a and 5b, respectively. For 30 UAVs, ConCov achieves a higher $F$ value than our proposed models at lower $T_c$ values, but its connectivity performance is worse (see Section VI-B2 below). The $F$ values of all the models converge when $T_c$ increases. With 50 UAVs, BSCAP-DQN achieves higher $F$ values than BS-CAP and ConCov for lower $T_c$ values, even though we do not have an explicit reward for fairness in our BSCAP-DQN formulation.

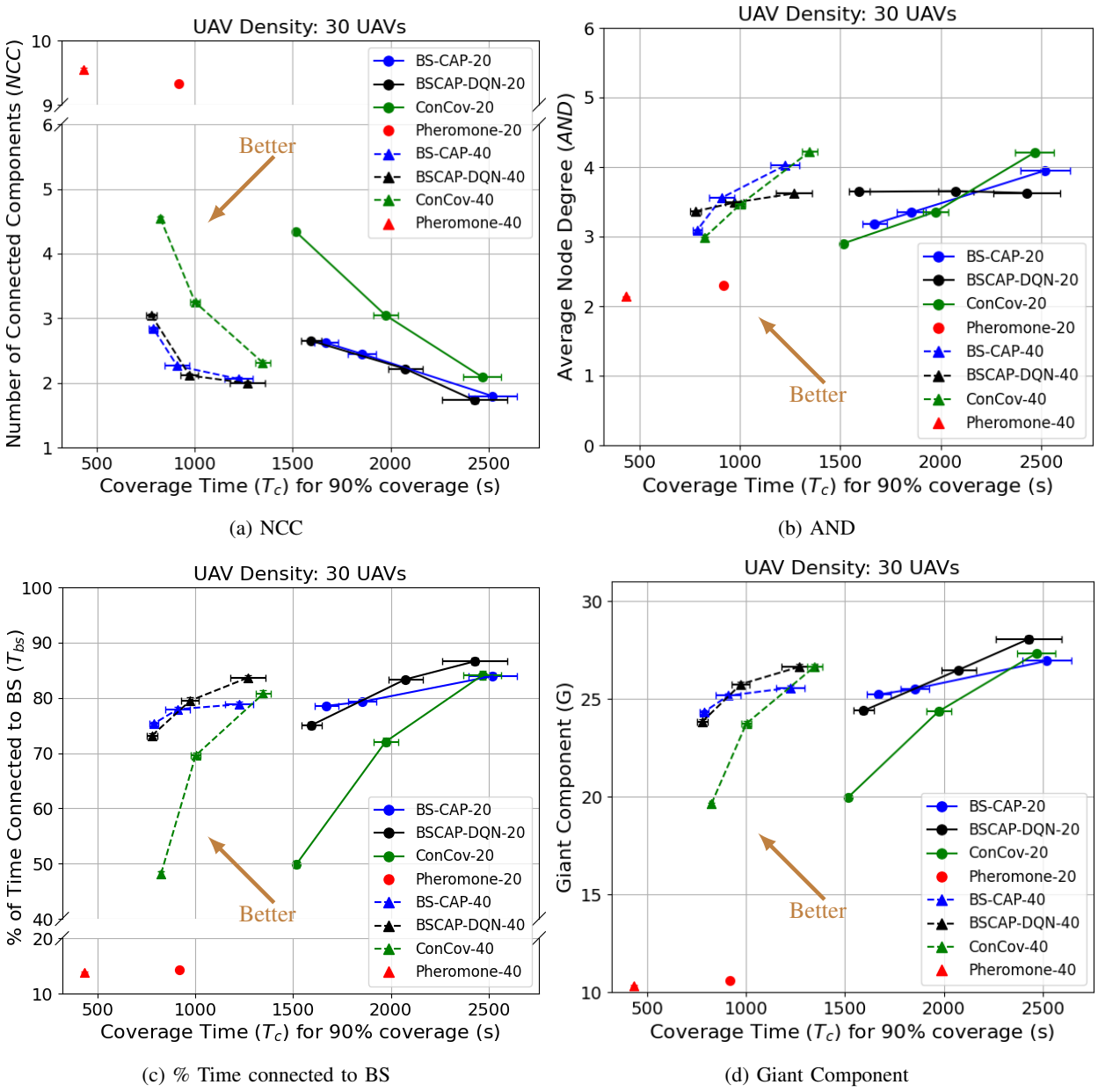
*2) Connectivity Performance:* The number of connected components (NCC) measures the "disconnectedness" of the network. In Figures 6a and 7a, we see ***NCC* vs. $T_c$ performance plots** for 30 and 50 UAVs, respectively. For fast area coverage and strong connectivity, we prefer low $NCC$ along with a low coverage time $T_c$. For both UAV densities and speeds, BS-CAP and BSCAP-DQN achieve better $NCC$ performance than the ConCov model for a given $T_c$ value. Our optimized BSCAP-DQN gives a similar or slightly better (lower) $NCC$ than our fixed BS-CAP heuristic for a given $T_c$ value. For example, in Figure 6a we see that at $T_c \approx$750 s for 30 UAVs, BSCAP-DQN-40 and BS-CAP-40 achieve $NCC$ of 2.9, while ConCov-40 has $NCC \approx 4.6$. Similarly, 50 UAVs at 40 m/s (Figure 7a) with policies giving $T_c \approx 500$ s, BSCAP-DQN-40 and BS-CAP-40 achieve $NCC$ of 1.4 and 1.5, respectively, while ConCov-40 has $NCC \approx 2.9$.

Average node degree (AND) captures a more local connectivity assessment. Figures 6b and 7b show ***AND* vs. $T_c$ performance plots** for 30 and 50 UAVs at both speeds. For both node densities and speeds, the BS-CAP and BSCAP-DQN achieve reasonably high $AND \geq 3$, maintaining a sufficient number of neighbor UAVs for network connectivity.[3] The ability to tune the BSCAP-DQN model by setting reward $r_k$ in (14) allows us to maintain an almost constant $AND$ of around 3.5 and 4 for 30 and 50 UAVs, respectively. In comparison, for 30 UAVs (Figure 6b), $AND$ values for BS-CAP models vary from 3 to 4, while ConCov models vary from 2.9 to 4.2. Similarly, for 50 UAVs (Figure 7b), BS-CAP's $AND$ values vary from 3.9 to 4.9, while ConCov's vary from 3.5 to 5.6.

Perhaps most importantly, we would like to maintain connection to the BS as much as possible. The ***$T_{bs}$* vs. $T_c$ performance plots** are shown in Figures 6c and 7c. We see that BS-CAP and BSCAP-DQN achieve higher BS connection time $T_{bs}$ than ConCov, at smaller $T_c$ values (faster coverage) for both settings of density and speed. For example, at $T_c \approx$ 2000 s for 30 UAVs at 20 m/s, BSCAP-DQN-20 and BS-CAP-20 achieve $T_{bs}$ of above 80%, compared to 72% by ConCov-20. When we tune the models to achieve higher $T_{bs}$ (stronger BS connectivity), the coverage time $T_c$ increases due to the stricter connectivity constraints. We find that the BSCAP-DQN model outperforms the BS-CAP model at higher $T_{bs}$, perhaps because the value of $n$ in (13) allows more direct control over the coverage vs. BS connectivity trade-off in BSCAP-DQN, compared to BS-CAP.

The **G vs. $T_c$ performance plots** for 30 and 50 UAVs are shown in Figures 6d and 7d, respectively. At both densities, BS-CAP and BSCAP-DQN achieve a larger giant network

[3]We consider that a model with $AND \geq 3$ represents reasonably strong node degree. In fact, a higher value of $AND$ may increase the co-channel interference and the probability of packet collisions during communication. Moreover, maintaining higher connectivity typically degrades coverage, especially at low UAV densities.




(a) NCC

(b) AND

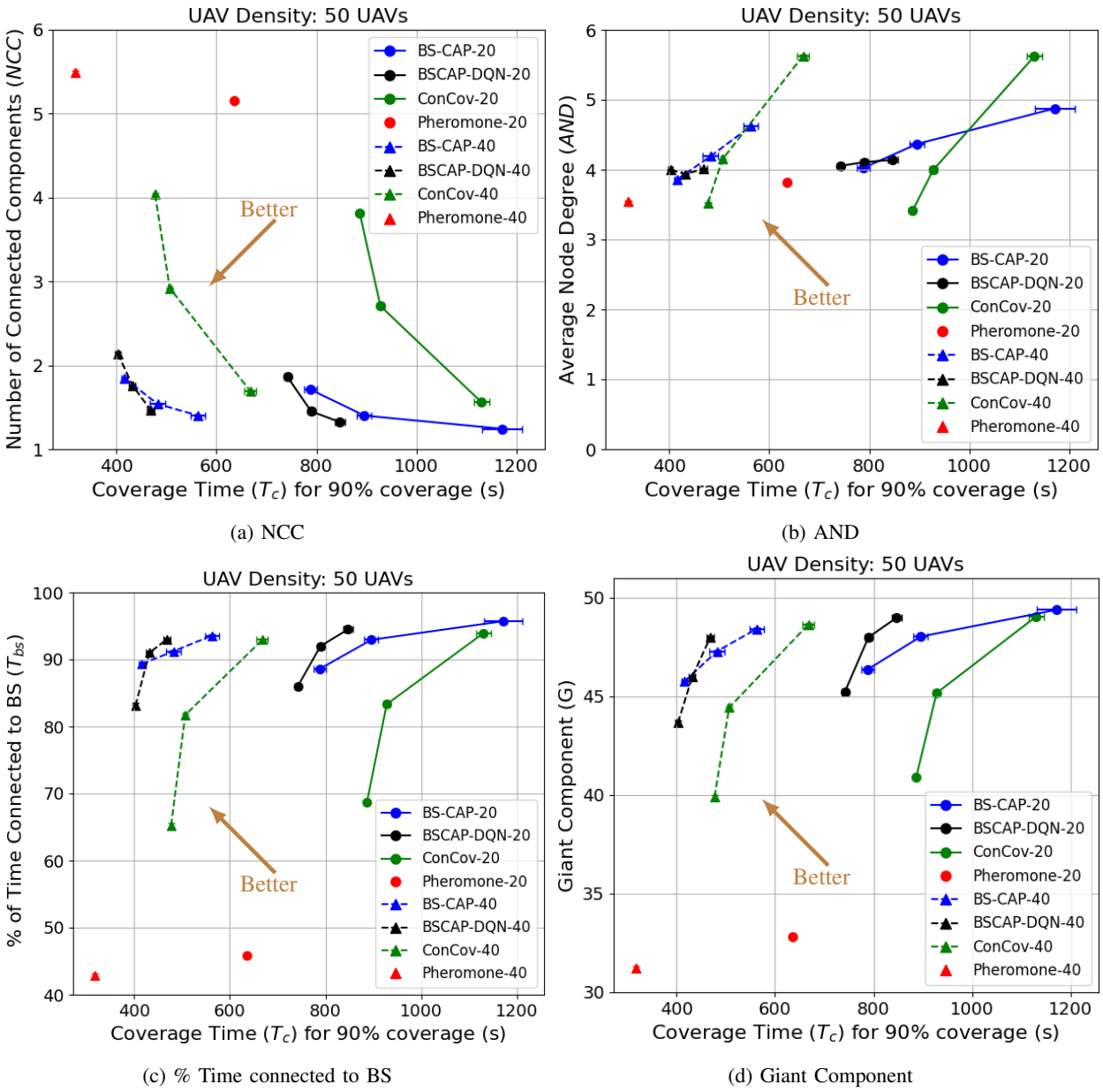
(c) % Time connected to BS

(d) Giant Component

Fig. 6: Connectivity performance plots for 30 UAVs at 20 m/s and 40 m/s. (a) At comparable coverage time, BS-CAP and BSCAP-DQN provide a smaller NCC (indicating better connectivity). The three tested parameter settings for each method are joined by lines, suggesting how intermediate values might fare. (b) All three methods maintain an average node degree between 3 - 4, providing a sufficient neighborhood. (c) BS-CAP and BSCAP-DQN provide more time connected to the base station than ConCov at similar coverage times. (d) Our methods' better connectivity also results in a larger giant component $G$, with fewer isolated UAVs.

component (G) than ConCov for faster coverage times (low $T_c$). A larger value of $G$ indicates that fewer isolated UAVs are present in the network; its increase is related to our methods' improved NCC and $T_{bs}$ connectivity statistics. For example, at $T_c \approx 800$ s for 30 UAVs at 40 m/s (Figure 6d), BS-CAP and BSCAP-DQN achieve a $G$ value of 24, compared to ConCov's $G$ value of 19.5. The component size $G$ improves in all three models as we increase the emphasis on connectivity.

**Comparing $T_{bs}$ vs. $T_c$ Performance of CAP-DQN and BSCAP-DQN**: To study the impact of considering the BS connectivity on coverage performance, we compare the coverage ($T_c$) and BS connectivity ($T_{bs}$) performance of BSCAP-DQN model with an earlier version which did not consider BS connectivity, denoted CAP-DQN [20]. We find that BSCAP-DQN achieves around 300% and 125% increase in $T_{bs}$ compared to the CAP-DQN policy for 30 and 50 UAVs, respectively; see Figure 8. Since CAP-DQN has no BS connectivity constraints, it provides faster coverage times ($T_c$) with good node degree, but fails to maintain a strong BS connectivity.




(a) NCC

(b) AND

(c) % Time connected to BS

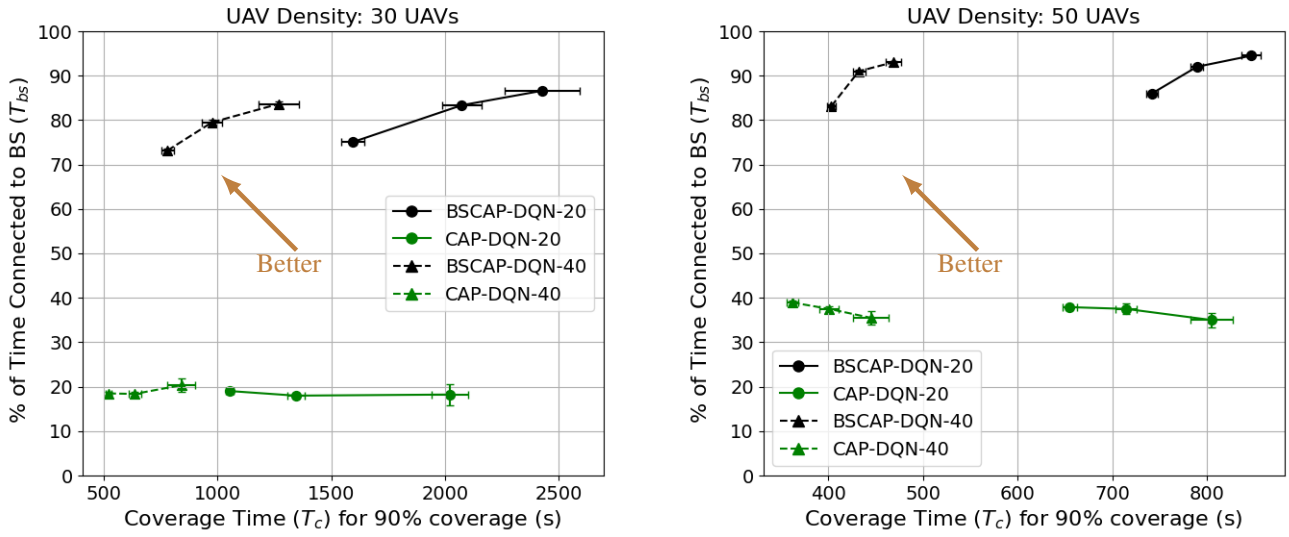
(d) Giant Component

Fig. 7: Connectivity performance plots for 50 UAVs at 20 m/s and 40 m/s. (a) At comparable coverage time, BS-CAP and BSCAP-DQN provide a smaller NCC (indicating better connectivity). (b) BS-CAP and BSCAP-DQN maintain an average node degree between 3.9 - 4.9, providing a sufficient neighborhood. (c) BS-CAP and BSCAP-DQN provide more time connected to the base station than ConCov at similar coverage times. (d) Our methods' better connectivity also results in a larger giant component $G$, with fewer isolated UAVs.

*3) Impact of Node Failures:* Low SWaP UAVs are prone to failure due to mechanical malfunctions or energy consumption. In this section, we study the area coverage and connectivity performance of our mobility models when a fraction of nodes randomly fail during the simulation time of 2000 s. We consider networks at both densities (30 and 50 nodes), at a speed of 20 m/s. Nodes fail progressively with 10% and 30% nodes failing by the end of simulation. Other simulation parameters are kept as in Table II. We evaluate using parameter values: $\beta = 1.5$ and $\beta' = 3$ in BS-CAP, $m = 3$ and $n = 3$ in BSCAP-DQN, and $\omega = 0.3$ in ConCov.

The **Coverage vs. Time plots** in Figure 9 show the total map area covered in a given time by the three mobility models. We see coverage performance decreases gracefully, in proportion to the number of node failures. For example, the area coverage by BS-CAP decreases from ≈80% to ≈78.5% and ≈76%, when 10% and 30% nodes fail, respectively.

In Table III, we show the connectivity ($NCC$, $AND$, $T_{bs}$, $G$) and coverage fairness ($F$) performance for 10% and 30% node failures. Both $NCC$ and $AND$ are hardly impacted, while $T_{bs}$, $G$ and $F$ decrease gracefully in proportion to the




(a) (b)

Fig. 8: $T_{bs}$ BSCAP-DQN vs. CAP-DQN performance plots for 30 and 50 UAVs, at 20 m/s and 40 m/s. BSCAP-DQN achieves higher $T_{bs}$ than the CAP-DQN since CAP-DQN does not consider the BS connectivity constraints.

number of node failures. This indicates all three mobility models are relatively robust to node failure.

In conclusion, at both densities of UAVs and both speeds, the pheromone model provides the best coverage performance ($T_c$, $F$), but has the worst connectivity performance ($NCC$, $AND$, $T_{bs}$, $G$) since it does not consider any connectivity information. Our proposed BSCAP-DQN and BS-CAP models outperform the pheromone and ConCov models by providing better connectivity performance (lower NCC, higher AND, and higher $T_{bs}$). BSCAP-DQN and BS-CAP provide better connectivity at a similar coverage performance compared to ConCov. Finally, we find that our RL-based BSCAP-DQN model performs only slightly better than our heuristic BS-CAP model, suggesting that our heuristic weighting performs reasonably well. BSCAP-DQN and BS-CAP are relatively robust to the node failures.

## VII. Conclusion

We considered a decentralized, multi-hop UAV network consisting of low SWaP fixed-wing UAVs. When monitoring an area autonomously, the area coverage and connectivity requirements of the UAV network exhibit a fundamental trade-off. Although fast area coverage is needed to quickly scan the area, strong node degree and BS connectivity are needed to coordinate the UAVs and transmit sensed information to the BS. To facilitate reliable communication among UAVs and to the BS in an autonomous UAV network, we designed a connectivity-aware pheromone (BS-CAP) mobility model. We then developed a deep Q-learning policy based BS-CAP model (BSCAP-DQN). Both BS-CAP and BSCAP-DQN facilitate efficient area coverage while maintaining strong node degree and BS connectivity, and significantly improve over existing schemes. Our proposed schemes work online, are fully distributed, rely only on neighborhood information, and are robust to node failures, making them practical for real-time coordination in a decentralized UAV network.

Our RL-based model provides slightly better performance than our heuristic BS-CAP model. This both suggests that the BS-CAP heuristic performs reasonably well, and that to improve significantly further we may need to incorporate more information into the state representation of our RL agent. For example, we could expand the state to include a history of pheromone and connectivity observations or additional information (obtained in a distributed manner) from the UAV's neighbors, such as their recent trajectories or their own pheromone information. We leave these as avenues for future research.

## VIII. Acknowledgment of Support and Disclaimer

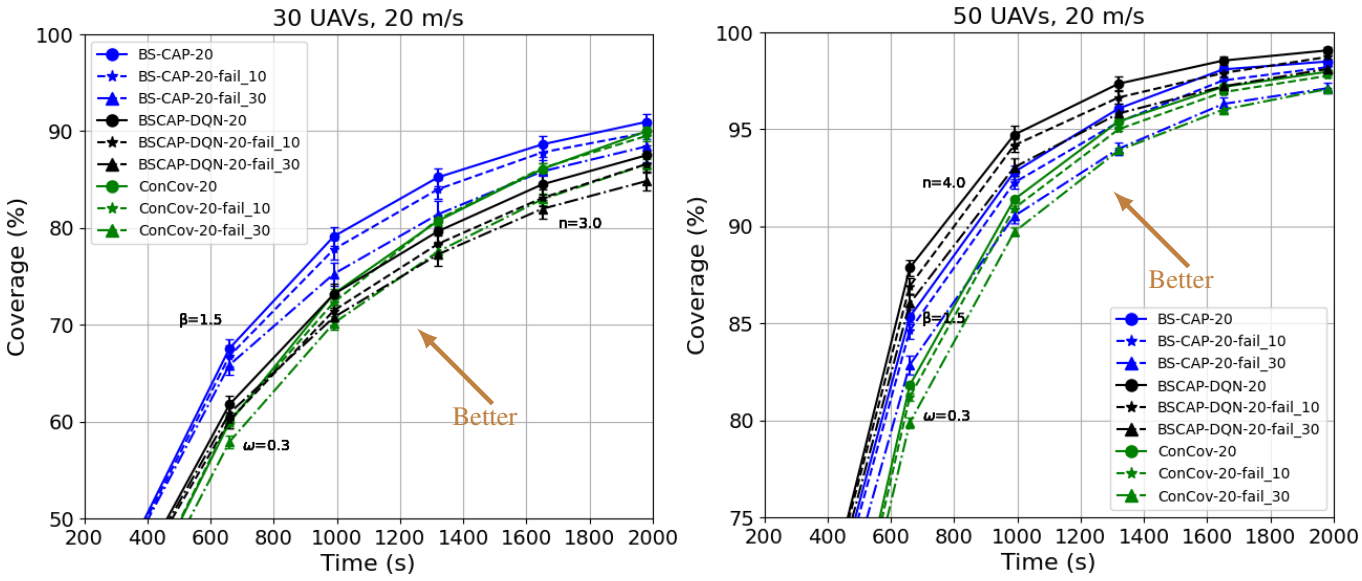
This material is based on research sponsored by the Air Force Research Laboratory under agreement No. FA8750-18-1-0023. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

## References

[1] A. L. Alfeo, M. G.C.A. Cimino, N. De Francesco, M. Lega, G. Vaglini, "Design and simulation of the emergent behavior of small drones swarming for distributed target localization," J. Computational Sci., vol. 29, pp. 19-33, 2018.

[2] I. Martinez-Alpiste, G. Golcarenarenji, Q. Wang and J. M. Alcaraz-Calero, "Search and rescue operation using UAVs: A case study," Expert Systems with Applications, vol 178, p.114937, 2021.

[3] E. Y. Adam, "Leveraging connectivity for coverage in drone networks for target detection." Balkan J. Electrical and Computer Engineering, vol. 7, pp. 218-225, 2019.




(a) (b)

Fig. 9: Coverage vs. Time plots for 30 and 50 UAVs at 20 m/s with 0%, 10% and 30% node failure. UAV failures result in decreased coverage performance. BS-CAP, BSCAP-DQN and ConCov use parameter settings of $\beta = 1.5$, $n = 3$ and $\omega = 0.3$, respectively; for 50 UAVs, BSCAP-DQN-20 uses $n = 4$ in order to show comparable coverage performance. Suffix "-fail_10" and "-fail_30" indicate 10% and 30% node failures, respectively.

TABLE III: Performance metrics of BS-CAP, BSCAP-DQN and ConCov for different percentages of UAV failure using parameter settings of $\beta = 1.5$, $n = 3$ and $\omega = 0.3$, respectively; for 50 UAVs, BSCAP-DQN-20 uses $n = 4$.

| | % Node Failures | 30 UAVs, 20 m/s | | | 50 UAVs, 20 m/s | | |
|---|---|---|---|---|---|---|---|
| | | BS-CAP | BSCAP-DQN | ConCov | BS-CAP | BSCAP-DQN | ConCov |
| NCC | 0 | 2.3 | 2.1 | 3 | 1.4 | 1.3 | 2.7 |
| | 10 | 2.3 | 2.2 | 3 | 1.5 | 1.4 | 2.8 |
| | 30 | 2.4 | 2.3 | 3.1 | 1.6 | 1.5 | 3 |
| AND | 0 | 3.5 | 3.8 | 3.4 | 4.4 | 4.2 | 4.1 |
| | 10 | 3.4 | 3.8 | 3.4 | 4.3 | 4.2 | 4.1 |
| | 30 | 3.3 | 3.8 | 3.3 | 4.2 | 4.2 | 4 |
| Tbs % | 0 | 80 | 84 | 72 | 94 | 95 | 84 |
| | 10 | 76 | 80 | 69 | 91 | 91 | 80 |
| | 30 | 70 | 73 | 61 | 84 | 84 | 70 |
| G | 0 | 26 | 27 | 24 | 48 | 49 | 45 |
| | 10 | 25 | 25 | 23 | 46 | 46 | 43 |
| | 30 | 22 | 23 | 22 | 42 | 42 | 39 |
| F | 0 | 0.76 | 0.74 | 0.78 | 0.91 | 0.92 | 0.92 |
| | 10 | 0.74 | 0.73 | 0.77 | 0.89 | 0.91 | 0.91 |
| | 30 | 0.72 | 0.71 | 0.74 | 0.86 | 0.88 | 0.88 |


[4] D. Hambling, "The US Navy wants swarms of thousands of small drones." https://www.technologyreview.com/2022/10/24/1062039/us-navy-swarms-of-thousands-of-small-drones/ (accessed Oct. 15, 2023).

[5] "AeroVironment Switchblade," Wikipedia, https://en.wikipedia.org/wiki/AeroVironment_Switchblade (accessed Oct. 15, 2023).

[6] "Raytheon Coyote," Wikipedia, https://en.wikipedia.org/wiki/Raytheon_Coyote (accessed Oct. 15, 2023).

[7] "Zala Lancet," Wikipedia, https://en.wikipedia.org/wiki/ZALA_Lancet (accessed Oct. 15, 2023).

[8] E. Y. Adam, "Connectivity considerations for mission planning of a search and rescue drone team," Turkish J. Electrical Engineering and Computer Sciences, 28(4), pp. 2228-2243, 2020.

[9] S. Garg, A. Ihler, E. S. Bentley and S. Kumar, "A Cross-Layer, Mobility, and Congestion-Aware Routing Protocol for UAV Networks," IEEE Trans. Aerospace and Electronic Systems, vol. 59, pp. 3778-3796, Aug. 2023.

[10] M. Alam, N. Ahmed, R. Matam and F. A. Barbhuiya, "IEEE 802.11ah-Enabled Internet of Drone Architecture," in IEEE Internet of Things Magazine, vol. 5, pp. 174-178, March 2022.

[11] M. R. Brust et al., "Target tracking optimization of UAV swarms based on dual-pheromone clustering," 3rd IEEE Int. Conf. Cybernetics, pp. 1-8, 2017.

[12] R. Rajan, M. Otte and D. Sofge, "Novel physicomimetic bio-inspired algorithm for search and rescue applications," IEEE Symp. Series Comput. Intell., pp. 1-8, 2017.

[13] E. Yanmaz, "Joint or decoupled optimization: Multi-UAV path planning for search and rescue" Ad Hoc Networks, vol. 138, p. 103018, 2023.

[14] H. J. Na and S. J. Yoo, "PSO-Based Dynamic UAV Positioning

Algorithm for Sensing Information Acquisition in Wireless Sensor Networks," in IEEE Access, vol. 7, pp. 77499-77513, 2019.

[15] A. I. Hentati, and L. C. Fourati, "Comprehensive survey of UAVs communication networks," Computer Standards & Interfaces, 72, p. 103451, 2020.

[16] Y. Shao, Z. Zhao, R. Li, and Y. Zhou, "Target detection for multi-UAVs via digital pheromones and navigation algorithm in unknown environments," Frontiers of Inform. Tech. & Electronic Eng. 21, pp. 796-808, 2020.

[17] H. Zhao, H. Wang, W. Wu, and J. Wei, "Deployment Algorithms for UAV Airborne Networks Toward On-Demand Coverage," IEEE J. Selected Areas in Commun., vol. 36, pp. 2015-2031, Sept. 2018.

[18] M. Messous, H. Sedjelmaci, and S. Senouci, "Implementing an emerging mobility model for a fleet of UAVs based on a fuzzy logic inference system," Pervasive and Mobile Computing, 42, pp. 393-410, 2017.

[19] S. Devaraju, A. Ihler, and S. Kumar, "A Connectivity-Aware Pheromone Mobility Model for Autonomous UAV Networks," IEEE Consumer Communications & Networking Conf. (RoboCom'23 Workshop), Las Vegas, NV, Jan. 2023.

[20] S. Devaraju, A. Ihler, and S. Kumar, "A Deep Q-Learning Connectivity-Aware Pheromone Mobility Model for Autonomous UAV Networks," IEEE Int. Conf. Computing, Networking & Comm., Honolulu, Hawaii, Feb. 2023.

[21] H. Duan, J. Zhao, Y. Deng, Y. Shi and X. Ding, "Dynamic Discrete Pigeon-Inspired Optimization for Multi-UAV Cooperative Search-Attack Mission Planning," IEEE Trans. Aerospace and Electronic Systems, vol. 57, pp. 706-720, Feb. 2021.

[22] H. El Hammouti, M. Benjillali, B. Shihada and M. -S. Alouini, "Learn-As-You-Fly: A Distributed Algorithm for Joint 3D Placement and User Association in Multi-UAVs Networks," IEEE Trans. Wireless Commun., vol. 18, pp. 5831-5844, Dec. 2019.

[23] H. A. Poonawala, and M. W. Spong, "Decentralized estimation of the algebraic connectivity for strongly connected networks," IEEE American Control Conf., pp. 4068–4073, 2015.

[24] K. Khateri, M. Pourgholi, M. Montazeri and L. Sabattini, "A Comparison Between Decentralized Local and Global Methods for Connectivity Maintenance of Multi-Robot Networks," in IEEE Robotics and Automation Letters, vol. 4, pp. 633-640, April 2019.

[25] J. Scherer, and B. Rinner, "Short and full horizon motion planning for persistent multi-UAV surveillance with energy and communication constraints," Proc. IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS), pp. 230–235, 2017.

[26] S. Rezwan and W. Choi, "A survey on applications of reinforcement learning in flying ad-hoc networks," Electronics, vol. 10, pp. 449, 2021.

[27] J. K. Gupta, M. Egorov, and M. J. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," Proc. Int. Conf. Auton. Agents Multiagent Syst., pp. 66-83, May 2017.

[28] W. Yue, X. Guan, and L. Wang. "A Novel Searching Method Using Reinforcement Learning Scheme for Multi-UAVs in Unknown Environments," Applied Sciences, vol. 9, pp. 4964, 2019.

[29] C. H. Liu, X. Ma, X. Gao and J. Tang, "Distributed Energy-Efficient Multi-UAV Navigation for Long-Term Communication Coverage by Deep Reinforcement Learning," IEEE Trans. Mobile Computing, vol. 19, pp. 1274-1285, 1 June 2020.

[30] L. Yu, F. Wu, Z. Xu, Z. Xie and D. Yang, "UAV path design with connectivity constraint based on deep reinforcement learning," Physical Communication, 52, p. 101582, 2022.

[31] K. L. Pham et al., "The study of electrical energy power supply system for UAVs based on the energy storage technology," Aerospace, vol. 9, p. 500, 2022.

[32] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, pp. 529, 2015.

[33] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," arXiv preprint arXiv:1509.02971, Sep 2015.

[34] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction,* MIT press, 2018.

[35] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," arXiv preprint arXiv:2005.01643, May 2020.

[36] A. Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," Advances in Neural Information Processing Systems, vol. 32, pp. 8026-8037, 2019.

[37] "Multi-sensor payloads: Intelligence, Surveillance, and Reconnaissance (ISR) micro-gimbals," AeroVironment, Inc., https://www.avinc.com/uas/payloads (accessed Oct. 16, 2023).

[38] M. Rosalie, M. R. Brust, G. Danoy, S. Chaumette, and P. Bouvry, "Coverage Optimization with Connectivity Preservation for UAV Swarms Applying Chaotic Dynamics," IEEE Int. Conf. Autonomic Computing (ICAC), Columbus, OH, USA, 2017, pp. 113-118.

[39] R. K. Jain, D. M. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," Eastern Research Laboratory, Digital Equipment Corporation, 1984.