Distributed Deep Reinforcement Learning for Intelligent Traffic Monitoring with a Team of Aerial Robots

Behzad Khamidehi and Elvino S. Sousa

Abstract—This paper studies the traffic monitoring problem in a road network using a team of aerial robots. The problem is challenging due to two main reasons. First, the traffic events are stochastic, both temporally and spatially. Second, the problem has a non-homogeneous structure as the traffic events arrive at different locations of the road network at different rates. Accordingly, some locations require more visits by the robots compared to other locations. To address these issues, we define an uncertainty metric for each location of the road network and formulate a path planning problem for the aerial robots to minimize the network’s average uncertainty. We express this problem as a partially observable Markov decision process (POMDP) and propose a distributed and scalable algorithm based on deep reinforcement learning to solve it. We consider two different scenarios depending on the communication mode between the agents (aerial robots) and the traffic management center (TMC). The first scenario assumes that the agents continuously communicate with the TMC to send/receive real-time information about the traffic events. Hence, the agents have global and real-time knowledge of the environment. However, in the second scenario, we consider a challenging setting where the observation of the aerial robots is partial and limited to their sensing ranges. Moreover, in contrast to the first scenario, the information exchange between the aerial robots and the TMC is restricted to specific time instances. We evaluate the performance of our proposed algorithm in both scenarios for a real road network topology and demonstrate its functionality in a traffic monitoring system.

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) have recently attracted considerable interest in a wide range of applications. Aerial reach, high mobility, and cost-effective deployment are the key features that make the UAVs an ideal candidate for applications such as drone delivery [1]–[3], search and rescue [4], [5], wireless communications [6]–[8], and mapping, tracking, and monitoring [9]–[11]. UAV-assisted traffic monitoring in urban areas is another emerging application that can play a key role in intelligent transportation systems (ITSs) [12]. Currently, the monitoring is performed by a set of networked cameras installed in different locations of the road network. However, the implementation cost of these systems is usually high. Hence, they are not economical solutions for monitoring short-term traffic events. Moreover, these systems do not offer flexible solutions for the dead zones or locations without appropriate infrastructures [13]. To overcome these limitations, we can integrate UAVs into traffic monitoring systems.

The authors are with the Department of Electrical and Computer Engineering, University of Toronto, ON M5S 1A1, Canada {b.khamidehi, es.sousa}@utoronto.ca

Fig. 1: Traffic monitoring using a network of aerial robots.

The UAV-assisted traffic monitoring has been investigated in several recent studies [13]–[23]. In [13], a single UAV traffic monitoring system has been developed to capture traffic videos from the road network and send them to the traffic management center (TMC). In [14], a cooperative traffic monitoring system has been considered to help terrestrial vehicles to have full information about their surroundings based on the UAV’s images. In [15]–[17], the authors adopted deep learning to estimate the traffic flow parameters from the UAV’s captured videos. In [19], the authors developed a parking occupancy detection algorithm based on the UAV’s images. In [20], multi-UAV tour planning problem has been studied to monitor the traffic on a given road network. However, the proposed algorithm is an offline planning one that only uses the road network topology and does not consider any dynamics in the system. In [21], an extended multiple traveling salesman problem has been studied to schedule a team of UAVs for traffic monitoring purposes. However, similar to [20], the considered problem is offline, where the visit points and the corresponding visit time windows are known. In [23], a dynamic traffic monitoring problem has been investigated where the authors assumed the UAVs can accurately detect the vehicles and estimate their true positions. Given this information, a simple path planning algorithm has been proposed to follow the gravity center of the vehicle clusters in the road network.

Our Contributions. In contrast to the mentioned studies that focus on either an offline problem setting [20], [21], or a scenario with perfect knowledge of the road vehicles [23], we consider a dynamic and online problem setting with partial observations, and solve the navigation problem for a team of UAVs under this limitation. Due to the random and time-
varying nature of the traffic events in the road networks, the UAVs must regularly visit different locations to catch the traffic events. To address this issue, we define an uncertainty metric for each location of the road network and formulate a path planning problem for the UAVs to minimize the network’s average uncertainty. We express this multi-UAV traffic monitoring problem as a partially observable Markov decision process (POMDP) and propose a decentralized and scalable solution based on deep reinforcement learning (RL) to solve the problem. Depending on the communication mode between the agents and the TMC, we consider two scenarios for the traffic monitoring problem. In the first scenario, we assume that the agents (UAVs) continuously communicate with the TMC and hence, they have perfect and real-time knowledge of the environment. However, in the second scenario, we consider a challenging setting where the information exchange between the agents and the TMC is restricted to specific time instances. In other words, we do not consider a continuous communication between the agents and the TMC. Moreover, we assume that the visibility of each UAV is limited to its sensing range, and hence, it has partial observation from its surrounding environment. We evaluate the performance of our proposed method for a real road network topology in downtown Toronto. Evaluation results show the effectiveness of our proposed algorithm for traffic monitoring purposes.

II. SYSTEM MODEL

We consider a team of \( N \) aerial vehicles that monitor traffic conditions in a given road network, as shown in Fig. 1. We use a grid-world representation of size \( M \times M \) for the environment. The total number of grid-cells is represented by \( K \triangleq M \times M \) and index \( k \) is used to refer to the \( k \)-th cell. The task of the UAVs is to visit different locations of the road network to capture images from the traffic conditions. These images will be sent to the TMC for traffic regulation purposes.

A. Agent Model

We use index \( i \) to refer to the \( i \)-th agent (UAV). The position of the \( i \)-th agent at time \( t \) is represented by \( p_i(t) \). Each UAV has a downward-facing camera that captures images from the streets and the traffic conditions. We assume that the camera’s field of view (FoV) can cover one grid cell (currently positioned cell). The UAV also has radio sensors (transmitter/receiver) to send its collected data to the TMC. The static global map of the environment is also given to all agents. Using a GPS sensor, each agent can localize itself on the map. This map also gives the locations of the static obstacles and no-fly zones. The agents must avoid collision with both these static obstacles and the moving objects, which are other agents in our model. Moreover, since the task of the aerial vehicles is to gather information from the road network, it will be a waste of resources if two agents cover the same cell simultaneously. Hence, we have

\[
p_{i_1}(t) \neq p_{i_2}(t), \quad \forall i_1 \neq i_2, \quad \forall t. \tag{1}
\]

B. Uncertainty Model

The goal of aerial vehicles is to monitor the traffic condition and collect information about the traffic events such as traffic jam(s), accident(s), traffic law violation(s), etc. These events can appear in different locations of the environment in a random and time-varying basis. To address the randomness of the events, we define an uncertainty metric for each location of the road network. Given this uncertainty model, we can form an uncertainty map for the environment. This map gives the probability of having an event in each location, or equivalently, it shows the locations that require a visit by the aerial vehicles because there is no confidence about their traffic conditions. Depending on the communication mode between the agents and the TMC, we consider two models for the uncertainty in our system.

Scenario I (continuous communication). In this scenario, we assume that the agents have a continuous communication with the TMC. Hence, the locations of the traffic events are given to the agents by the TMC. Let \( e_k(t) \) denote an indicator function taking value of 1 if there is an active event in the \( k \)-th grid-cell at time \( t \) and 0, otherwise. In this scenario, the UAVs know the values of \( e_k(t), \forall k \) at each time \( t \). The goal of the UAVs is to visit the locations with active events, i.e., the locations with \( e_k(t) = 1 \). We define the uncertainty of the \( k \)-th cell as

\[
u_k(t) = \begin{cases} 1 & \text{if } e_k(t) = 1, \\ 0 & \text{otherwise}. \end{cases} \tag{2}
\]

To reduce the uncertainty of the environment, the agents should visit locations with \( e_k(t) = 1 \). After visiting a location with an active event, its corresponding \( e_k(t) \) will be 0, meaning that an agent has visited the location and there is no further uncertainty about it. The value of uncertainty remains 0 until another event emerges at this location.

Scenario II (limited communication). In this case, the communication between the agents and the TMC is limited to specific time instances. As a result, the agents do not have complete information about the events and their locations. Let \( \nu_k(t) \) denote an indicator function that takes value of 1 if the \( k \)-th cell is visited by one of the agents at time \( t \). Otherwise, we have \( \nu_k(t) = 0 \). Moreover, let \( \tau_k \) denote the last time that the \( k \)-th grid-cell has been visited by an agent. Under a Poisson distribution, the probability that we have at least one event in the \( k \)-th cell in interval \( [\tau_k, t] \) is \( 1 - e^{-\alpha_k(t-\tau_k)} \), where \( \alpha_k \) is the rate of event arrival in the \( k \)-th grid-cell. We can use this probability as the uncertainty metric. In other words, at time \( t \), we can define the uncertainty of the \( k \)-th cell as

\[
u_k(t) = 1 - e^{-\alpha_k(t-\tau_k(t))}, \tag{3}
\]

where

\[
\tau_k(t) = \max_{0 \leq \tau' \leq t} \{ \tau' | \nu_k(\tau') = 1 \}. \tag{4}
\]

According to this definition, when \( t = \tau_k(t) \), the \( k \)-th cell is visited by an agent. Hence, there is no uncertainty about this cell, and the value of uncertainty is 0. However, as \( t \)
increases, the value uncertainty increases based on [3]. When the value of \( t \) becomes sufficiently large, the uncertainty tends to 1. This implies that there is no further confidence about the corresponding cell as it has been a long time since the last agent visited this cell.

C. Sensing Range and Information Exchange

As discussed earlier, in scenario I, each agent has complete knowledge of the environment as the locations of active events (events with \( e_k(t) = 1 \)) and other agents’ real-time locations are given to each agent by the TMC. As a result, there is no limitation for the sensing range of each UAV in scenario I. We can assume that each agent has access to the global and real-time uncertainty map of the environment in this scenario. However, in scenario II, we assume that the visibility of each agent is limited to its sensing range. Let \( N_i(t) \) denote the set of agents that are located in the sensing range of the \( i \)-th agent at time \( t \).

\[
N_i(t) = \{ i' \mid \| p_i(t) - p_{i'}(t) \| \leq r_s, i' \neq i \}, \tag{5}
\]

where \( r_s \) is the sensing range of each robot. In scenario II, at time \( t \), the \( i \)-th agent only knows locations of the agents in \( N_i(t) \). Moreover, in this scenario, the information exchange between the agents and the TMC is performed every \( T_u \) time units. For this purpose, each agent has a memory that keeps a record of the last \( L \) locations (cells) the agent has visited. This information is sent to the TMC every time the agent and the TMC communicate (every \( T_u \) time units). Using this information, the TMC updates the uncertainty map of the environment and sends it back to the agents. It is worth mentioning that there is no need for synchronous communication between all agents. In other words, the agents can communicate with the TMC at different time instances.

D. Uncertainty Map Update

In scenario I, all agents have access to the global uncertainty map. Let \( \mathcal{V}^u(t : t+1) \) and \( \mathcal{V}^n(t : t+1) \) denote the set of indices corresponding to the cells that have new events in interval \( [t, t+1) \) and the cells that are visited by one of the agents in interval \( [t, t+1) \), respectively. To update the uncertainty map in this scenario, we set

\[
e_k(t+1) = 0, \quad \forall k \in \mathcal{V}^n(t : t+1),
\]

\[
e_k(t+1) = 1, \quad \forall k \in \mathcal{V}^u(t : t+1).
\]

For all other grid-cells, we have \( e_k(t+1) = e_k(t) \). Using these values, the new uncertainty can be derived based on the uncertainty equation in [3].

In scenario II, as we discussed earlier, the agents do not have access to the global uncertainty map at all time instances. Hence, each agent maintains a local uncertainty map for itself and updates this map using its local information. Once the agent communicates with the TMC, it can update its local map with the global uncertainty map (every \( T_u \) time units). In what follows, we discuss how the uncertainty map is updated locally and globally by each agent and the TMC, respectively.

- **Local update**: In the time interval between two consecutive updates by the TMC, each UAV updates its own uncertainty map using its local collected data. In particular, at time \( t \), each agent sets \( \tau_k(t+1) = t+1 \) for its current cell and the cells that are in its sensing range and have been visited by one of the agents in time interval \( [t, t+1) \). For other cells, the agent sets \( \tau_k(t+1) = \tau_k(t) \). Using the value of \( \tau_k(t+1) \) and [3], the agent updates its local uncertainty map.

- **Global update**: Every \( T_u \) time units, the agents send their visited locations and the corresponding visit times to the TMC. Let \( \mathcal{V}^u(t : t + T_u) \) denote the set of all cells that have been visited by at least one of the agents in interval \( [t, t + T_u) \). We have

\[
\mathcal{V}^u(t : t + T_u) = \bigcup_{i=1}^{T_u} \mathcal{V}^u(t + i - 1 : t + i).
\]

If \( k \in \mathcal{V}^u(t : t + T_u) \), the TMC will set \( \tau_k(t + T_u) \) to the time that the cell has been visited by an agent. In case that the \( k \)-th cell has been visited more than once during \( [t, t + T_u) \), the TMC sets \( \tau_k(t + T_u) \) to the last time that the cell has been visited. For other cells that have not been visited by any of the agents in interval \( [t, t + T_u) \), the TMC sets \( \tau_k(t + T_u) = \tau_k(t) \). Using the value of \( \tau_k(t + T_u) \) and [3], the TMC evaluates the new uncertainty map and broadcasts it to the agents.

E. Problem Definition

To formulate the problem, first, we define the average uncertainty of the environment as

\[
\bar{u} = \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{K} u_k(t), \tag{6}
\]

where \( T \) is the total monitoring time. The goal of the agents is to minimize the average uncertainty (\( \bar{u} \)) in the environment. To achieve this goal, the UAVs need appropriate paths to follow. These paths must satisfy the condition in [11] throughout the UAVs’ flights. In the next section, we describe how to formulate the path-planning problem as a POMDP and solve it using reinforcement learning techniques.

III. METHODOLOGY

A. Reinforcement Learning Overview

Reinforcement learning is a framework for solving sequential decision-making problems. In RL, the agent interacts with the environment in a sequence of discrete time instances as follows: At each time \( t \), the agent receives observation \( o_t \) from the environment. This observation is a representation of the true state of the environment, denoted by \( s_t \), which is not directly observable by the agent. Using this observation, the agent takes action \( a_t \), receives reward \( r_{t+1} \) from the environment and goes to a new state \( s_{t+1} \) which is available to the agent through its observation \( o_{t+1} \), and this procedure continues. To formulate this interaction, we can use POMDPs. A POMDP can be expressed as a tuple \( \langle S, A, T, R, \Omega, O, \gamma \rangle \), where \( S \) is the state space, \( A \) is the
finite action space, \( T(s', s, a) = P(s'|s, a) \) is the transition function that maps actions and states to a distribution over the next states, \( R : S \times A \rightarrow \mathbb{R} \) is the reward function, \( \Omega \) is the observation space, \( O(s, a, o) = P(o|s, a) \) is the observation function, and \( \gamma \in (0, 1] \) is the discount factor. The action selection mechanism of the agent is called policy and is denoted by \( \pi(a|o) = P(a_i|a_o) \). Let \( Q_\pi(o, a) \) denote the expected return the agent receives over the long run if it starts from a state with observation \( o \), take action \( a \), and follow policy \( \pi \) afterwards. This function is referred to as Q-function and is defined as

\[
Q_\pi(o, a) = \mathbb{E}_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|o = o_i, a = a_t \right\}.
\]  

(7)

The goal of the RL agent is to find a policy \( \pi^* \) that maximizes \( Q_\pi(o, a) \). In problems with a large state/action space, we can use a multi-layer neural network to represent Q-function. In other words, we have \( Q_\pi(o, a) \approx Q(o, a; \theta) \), where \( \theta \) is the parameter of the neural network. The corresponding neural network is called Q-network.

**Deep Q-networks (DQN).** To obtain the Q-function, we can use DQN [24], [25]. The key components of this algorithm are the target network and the experience replay memory. We denote the parameter of the target network with \( \theta^- \) which is a periodic copy of \( \theta \). At each time \( t \), the agent implements an \( \epsilon \)-greedy action selection to explore its environment. Upon taking the action, the agent’s experience tuple, i.e., \( (o_t, a_t, o_{t+1}, r_{t+1}) \), is stored in the replay memory \( D \). To update \( \theta \), we sample a mini-batch of size \( b \) from \( D \) and define the target values for each sample as

\[
y_t = r_{t+1} + \gamma \max_a Q(o_{t+1}, a; \theta^-).
\]

(8)

By minimizing the loss function defined as

\[
\mathcal{L}(\theta) = E_\pi \{ (Q(o_t, a_t; \theta) - y_t)^2 \},
\]

(9)

we can update \( \theta \).

**B. Multi-robot Traffic Monitoring as a POMDP**

Now, we can formulate the problem as a POMDP. In what follows, we introduce the components of our considered POMDP.

**State.** The state of the \( i \)-th agent is defined as

\[
s_i(t) = [p_i(t), M, p_{-i}(t), U(t)],
\]

(10)

where \( p_i(t) \in \mathbb{R}^2 \) is the position of the \( i \)-th robot in the map, \( M \in \mathbb{R}^{M \times M} \) is the map of the environment which consists of the static obstacles, streets, intersections, etc., \( p_{-i}(t) \in \mathbb{R}^{2 \times (N-1)} \) includes the locations of all other robots (except \( i \)), and \( U(t) \in \mathbb{R}^{M \times M} \) is the global uncertainty map of the environment.

**Observation.** In scenario I, the observation of each agent is the same as its state. As a result, the considered POMDP reduces to a fully observable MDP. However, in scenario II, the observation of each agent is limited to a certain range and the states are not directly observable by the agents. Therefore, at time \( t \), the \( i \)-th agent (\( \forall i \)) only knows the locations of the agents that are in \( N_i(t) \). Accordingly, it can updates its uncertainty map only using this limited information and it does not know the global and true uncertainty of the whole road network. Let \( U_i(t) \in \mathbb{R}^{M \times M} \) denote the \( i \)-th agent’s local uncertainty map at time \( t \). This map is updated locally by the \( i \)-th agent. According to our discussion, the observation of the \( i \)-th agent has the following components

\[
[p_i(t), M, \{p'_{i'}(t), \forall i' \in N_i(t)\}, U_i(t)].
\]

(11)

Instead of \( [\Pi] \), we can use a multi-channel representation for the observation and define \( o_i(t) \) as

\[
o_i(t) = [p_i(t), M, U_i(t)].
\]

(12)

In this representation, \( P_i \in \mathbb{R}^{M \times M} \) is the position channel that encodes the position of the \( i \)-th agent and its neighbouring agents. For this channel, we use different values to differentiate between the ego vehicle (\( i \)-th agent) and those in \( N_i(t) \). We also use different values to differentiate between different objects of the second channel such as obstacles, roads, etc.

**Action.** We denote action of the \( i \)-th agent by \( a_i(t) \). At each time \( t \), the agent can change its current cell and go to one of its neighboring cells such as north, south, west, east, northwest, northeast, southwest, and southeast. The agent can also remain in its current cell. Hence, the action space of each agent has size 9.

**Reward.** We consider a reward function that has the following components:
between the agents and the TMC takes place every \(T\) time. As described earlier, the information exchange stores its experience tuple in its local memory. Moreover, the agents in environment. At each time, the agents Algorithm 1. At each episode, we randomly initialize posi-
tions of the agents in environment. A strategy for the agents is to visit the locations with higher uncertainties, as the contribution of such locations in (9) is more than locations with small uncertainties. To achieve this goal, we use the uncertainty function \(u_k(t)\) to define \(r_t^u\) as
\[
r_t = r_t^c + r_t^n + \lambda r_t^u,
\]
where \(\lambda\) is the parameter of the reward function.

C. Algorithm Description

To solve the given POMDP, we use a distributed algorithm based on DQN. The description of the algorithm is given in Algorithm 1. At each episode, we randomly initialize positions of the agents in environment. At each time, the agents adopt \(\epsilon\)-greedy policies. Upon taking an action, each agent stores its experience tuple in its local memory. Moreover, the agents update its local uncertainty map to use in the next round. As described earlier, the information exchange between the agents and the TMC takes place every \(T_u\) time units. In this stage, the agents send their visited locations and the corresponding experience tuples (stored in the local memories) to the TMC. The TMC adds these experiences to the global memory \(D\). The TMC uses the received data to update the global uncertainty map and sends this map to all agents. To train and update the parameter of the Q-network at each step of the episodes, a mini-batch of size \(b\) is sampled from \(D\) and the corresponding loss in (9) is minimized. This can be carried out in either centralized or decentralized fashion. The procedure continues until the Q-network is trained.

IV. Evaluation and Results

In this section, we evaluate the performance of our proposed algorithm on a real road network topology and present the results.

A. Experimental Setup and Environment

We implement the multi-robot traffic monitoring environment in Python. For our environment, we consider an area in downtown Toronto, as shown in Fig. 2a. The task of the aerial vehicles is to monitor the traffic condition on the main roads. We represent the given area with a grid of size 30 \(\times\) 30. We limit the maximum speed of the aerial vehicles to \(2m/s\) to allow them to capture precise images of the traffic condition. The UAV’s actions are made once per minute. Hence, each time slot in our evaluation is 1 min. We assume that \(\alpha_k = \alpha, \forall k\). Unless otherwise stated, the value of \(\alpha\) is set to 0.01.

B. Neural Network and Implementation Parameters

The structure of the neural network is given in Fig. 2b. The network has both convolutional (Conv) and fully connected (FC) layers. The size of the input image is 30 \(\times\) 30. We use rectified linear unit (ReLU) function as the activation function for both convolutional and fully connected layers. The last layer of the network has no activation as it estimates the Q-function values. For training, each agent implements
an ϵ-greedy policy to select its action. The value of ϵ is gradually annealed from 0.5 to 0.05. The parameters used for the training are given in Table I. The sensing range of the agents \( r_s \) is set to 1.5d, where d the width of each grid cell. In our experiment, \( d = 60 \text{m} \). For the reward function in (13), we consider the following components: \( r^f_1 = -20 \), \( r^n_1 = +1 \), and \( \lambda = 5 \).

C. Results and Discussion

Fig. 3 shows the training curves of our algorithm for both scenarios. The number of agents are considered as \( N = 3 \) and \( N = 10 \). At the beginning of the training, the agents do not know the optimal policy. Hence, they take inefficient actions. As training continues, the agents learn to adopt their paths such that the average uncertainty in the environment decreases. In scenario I, the agents learn to maximize their visits to locations with active events. However, in scenario II, they learn to visit locations with high uncertainty values more frequently. We also observe that the average reward decreases with the number of agents. In fact, according to (13), the received reward of each agent depends on the uncertainty of the visited location. As the number of agents increases, the uncertainty of the network decreases since we have more monitoring resources. Accordingly, the uncertainty term in (13) will have a smaller value which in turn reduces the received reward.

Fig. 4 shows sample paths for a team of 3 aerial vehicles in both scenarios. In Fig. 4a, we assume that there is no active event at the beginning, and all events emerge during the monitoring period. In contrast, in Fig. 4b, we consider a random uncertainty map at the beginning of the monitoring cycle. The task of the aerial vehicles in both scenarios is to dynamically adjust their paths to minimize the uncertainty in the network. In Fig. 4a, the agents learn to successfully visit the traffic events that emerged during the mission period to capture real-time images. However, in Fig. 4b, the agents choose their paths to visit locations with a high uncertainty value. Moreover, we observe that aerial vehicles learn to fly over the roads almost all time instances. Even for changing the roads, instead of choosing the shortest paths, they pick longer paths that cover the roads.

The number of agents (\( N \)) is another important factor that affects performance of the traffic monitoring system. In Fig. 5, we present the uncertainty of the road network for both scenarios as a function of \( N \). As we expect, the average uncertainty decreases with the number of agents. This reduction is more significant in scenario I compared to scenario II. The reason for this comes from the difference between the uncertainty models in (2) and (3). In scenario I, after visiting a location with an active event, the uncertainty of that location will be 0. This value remains unchanged until another event emerges in the mentioned location. In contrast, in scenario II, after visiting a location, its uncertainty does not remain constant. In other words, the uncertainty value is set to 0 upon the visit. However, the uncertainty value increases as time passes (see equation (3)). Hence, the value of uncertainty will be higher in this scenario, and accordingly, the percentage of the uncertainty reduction will be smaller.

Fig. 5b shows the effect of \( T_u \) on the performance of the proposed algorithm. When \( T_u \) is small, the agents will communicate with the TMC more frequently (\( T_u = 1 \) corresponds to the continuous communication between the agents and the TMC). As a result, their uncertainty models will be more accurate than when the agents communicate less often. This accuracy improves the probability that the agents visit locations with higher uncertainty values. We explain this issue with one example. Consider cell \( k \) with a high uncertainty value and assume that this cell has been visited by agent \( i_1 \) at time \( t \). Moreover, assume that agent \( i_2 \) is not in \( N_i(t) \). The true uncertainty of this location will be 0 after the visit. However, since this cell is not in the sensing range of agent \( i_2 \), agent \( i_2 \) does not become

\[ \text{Average reward} \]

\[ \text{Average sum uncertainty} \]

\[ \text{Episode} \]

\[ 0 \rightarrow 100 \rightarrow 200 \rightarrow 300 \rightarrow 400 \rightarrow 500 \]

\[ N = 3 \quad N = 10 \]

\[ \text{(a)} \quad \text{(b)} \quad \text{(c)} \quad \text{(d)} \]

\[ \text{Fig. 3: The training curves of our algorithm. (a) and (b) correspond to scenario I and (c) and (d) are for scenario II.} \]

\[ \text{(a)} \quad \text{(b)} \]

\[ \text{Fig. 4: Sample paths of the aerial vehicles for (a) scenario I and (b) scenario II.} \]
are aware of the visit. Accordingly, agent $i_2$ does not update the uncertainty of cell $k$ in its local uncertainty map. In contrast, it considers cell $k$ as a location with a high uncertainty value. Accordingly, the agent considers cell $k$ as a candidate for its next visit. The number of these inefficient visits increases by the value of $T_{m}$. It is worth mentioning that the resulting gap in uncertainty values will be negligible for a small number of agents. However, as the number of agents increases, it becomes more important to have precise knowledge of the environment to make efficient decisions.

V. CONCLUSIONS

We studied the traffic monitoring problem in a road network using a fleet of UAVs. To address the stochastic nature of the traffic events, we used an uncertainty metric to model the traffic monitoring problem. We considered two different scenarios, depending on the communication mode between the agents and the TMC. In the first scenario, we assumed that the agents continuously exchange information with the TMC, and hence, they have complete and real-time knowledge of the environment. However, in the second scenario, we assumed that the communication between the agents and the TMC is limited to specific time instances. Moreover, the observation of each agent is restricted to its sensing range. Therefore, the agents have partial observation of the environment. To develop a framework that works in both cases, we expressed the traffic monitoring problem as a POMDP and proposed a distributed algorithm based on deep Q-learning to control the agents’ movements. Experimental results showed the effectiveness of our proposed algorithm in reducing uncertainty of the environment.

REFERENCES

[1] S. Choudhury, K. Solovey, M. J. Kochenderfer, and M. Pavone, “Efficient Large-scale Multi-drone Delivery Using Transit Networks,” Journal of Artificial Intelligence Research, vol. 70, pp. 757–788, 2021.

[2] D. N. Das, R. Sewani, J. Wang, and M. K. Tiwari, “Synchronized Truck and Drone Routing in Package Delivery Logistics,” IEEE Transactions on Intelligent Transportation Systems, pp. 1–11, 2020.

[3] B. Khamidehi, M. Raic, and E. S. Sousa, “Dynamic Resource Management for Providing QoS in Drone Delivery Systems,” arXiv preprint arXiv:2103.04015, 2021.

[4] G. Bevacqua, J. Cacace, A. Finzi, and V. Lippiello, “Mixed-initiative Planning and Execution for Multiple Drones in Search and Rescue Missions,” in Proceedings of the International Conference on Automated Planning and Scheduling, vol. 25, no. 1, 2015.

[5] E. T. Alotaibi, S. S. Alqefari, and A. Koubba, “LSAR: Multi-uav collaboration for search and rescue missions,” IEEE Access, vol. 7, pp. 55817–55832, 2019.

[6] Y. Zeng, R. Zhang, and T. J. Lim, “Wireless Communications with Unmanned Aerial Vehicles: Opportunities and Challenges,” IEEE Communications Magazine, vol. 54, no. 5, pp. 36–42, 2016.

[7] A. Al-Hilo, M. Samir, C. Assi, S. Sharafaldine, and D. Ebrahimi, “UAV-Assisted Content Delivery in Intelligent Transportation Systems-Joint Trajectory Planning and Cache Management,” IEEE Transactions on Intelligent Transportation Systems, 2020.

[8] Y. Zeng, Q. Wu, and R. Zhang, “Accessing From the Sky: A Tutorial on UAV Communications for 5G and Beyond,” Proceedings of the IEEE, vol. 107, no. 12, pp. 2327–2375, 2019.

[9] R. Bailon-Ruiz, S. Lacroix, and A. Bit-Monnot, “Planning to Monitor Wildfires with a Fleet of UAVs,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 4729–4734.

[10] R. N. Hakasr and M. Schwager, “Distributed Deep Reinforcement Learning for Fighting Forest Fires with a Network of Aerial Robots,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 1067–1074.

[11] R. Bonatti, C. Ho, W. Wang, S. Choudhury, and S. Scherer, “Towards a Robust Aerial Cinematography Platform: Localizing and Tracking Moving Targets in Unstructured Environments,” in 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 229–236.

[12] H. Mencouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, “UAV-enabled Intelligent Transportation Systems for the Smart City: Applications and Challenges,” IEEE Communications Magazine, vol. 55, no. 3, pp. 22–28, 2017.

[13] H. Niu, N. Gonzalez-Prelic, and R. W. Heath, “A UAV-based Traffic Monitoring System-invited paper,” in 2018 IEEE 87th Vehicular Technology Conference (VTC Spring). IEEE, 2018, pp. 1–5.

[14] C. J. de Frias, A. Al-Kaff, F. M. Moreno, A. Madridano, and J. M. Armingol, “Intelligent Cooperative System for Traffic Monitoring in Smart Cities,” in 2020 IEEE Intelligent Vehicles Symposium (IV). IEEE, pp. 33–38.

[15] R. Ke, Z. Li, J. Tang, Z. Pan, and Y. Wang, “Real-time Traffic Flow Parameter Estimation from UAV Video Based on Ensemble Classifier and Optical Flow,” IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 1. pp. 54–64, 2018.

[16] J. Zhu, K. Sun, S. Jia, Q. Li, X. Hou, W. Lin, B. Liu, and G. Qiu, “Urban Traffic Density Estimation based on Ultrahigh-resolution UAV Video and Deep Neural Network,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 11, no. 12, pp. 4968–4981, 2018.

[17] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui, and Y. Wang, “Real-time Bidirectional Traffic Flow Parameter Estimation from Aerial Vodios,” IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 4, pp. 890–901, 2016.

[18] N. Xiao, Z. Li, Y. Yang, L. Qi, and R. Ke, “High-resolution Vehicle Trajectory Extraction and Denoising from Aerial Videos,” IEEE Transactions on Intelligent Transportation Systems, 2020.

[19] Y. Wang and B. Ren, “Quadrotor-enabled Autonomous Parking Occupancy Detection,” in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 8287–8292.

[20] C. Christodoulou and P. Kolios, “Optimized Tour Planning for Drone-based Urban Traffic Monitoring,” in 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring). IEEE, 2020, pp. 1–5.

[21] E. S. Rigas, P. Kolios, and G. Ellinas, “Extending the Multiple Traveling Salesman Problem for Scheduling a Fleet of Drones Performing Monitoring Missions,” in 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 2020, pp. 1–6.

[22] H. Huang, A. V. Savkin, and C. Huang, “Decentralised Autonomous Navigation of a UAV Network for Road Traffic Monitoring,” IEEE Transactions on Aerospace and Electronic Systems, 2021.

[23] M. Eloumi, R. Dhaou, B. Escrig, H. Idouli, and L. A. Saidane, “Monitoring Road Traffic with a UAV-based System,” in 2018 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2018, pp. 1–6.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with Deep Reinforcement Learning,” arXiv preprint arXiv:1312.5602, 2013.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., “Human-level Control Through Deep Reinforcement Learning,” nature, vol. 518, no. 7540, pp. 529–533, 2015.