Persistent Multi-UAV Road Network Monitoring via Decentralized Control

Meng Xu[1] and Nikolaos Geroliminis*[1]

[1] *Urban Transport Systems Laboratory (LUTS), École Polytechnique Fedéralé de Lausanne (EPFL), Lausanne, CH-1015, Switzerland*

Short summary

Persistent road-network monitoring with multiple UAVs requires continuously refreshing time-sensitive traffic knowledge under battery constraints. Unlike one-shot coverage, monitoring quality depends on both where and when UAVs revisit road segments, making long-term coordination a central challenge. In this paper, we formulate persistent multi-UAV road-network monitoring as an asynchronous graph decision problem with temporally decaying road knowledge. We develop a decentralized monitoring framework in which UAVs repeatedly select road segments to maintain network-wide awareness while operating under charging constraints. To improve decision quality without sacrificing scalability, we combine learned decentralized control based on multi-agent proximal policy optimization (MAPPO) with short-horizon planning based on predictor + upper confidence bounds for trees (PUCT) search. The framework also operates in the presence of background delivery UAV traffic, whose motion affects coordination in the shared road network. Experiments in an urban road-network environment show that the framework provides an effective and practical solution for persistent monitoring and improves decentralized monitoring performance in dynamic road networks.
Keywords: persistent monitoring, multi-UAV systems, asynchronous decision making

1 Introduction

Urban road network monitoring is a core function of intelligent transportation systems, supporting traffic safety, congestion mitigation, logistics operations, anomaly detection, and emergency response. However, maintaining up-to-date situational awareness over large urban networks remains difficult. Traditional approaches based on fixed cameras or static sensing infrastructure often suffer from limited spatial coverage, sensing errors, high deployment and maintenance costs, and poor adaptability to evolving hotspots. In contrast, unmanned aerial vehicles (UAVs) provide a flexible mobile sensing platform that can actively reposition over space and time (Huang et al., 2021; Savkin & Huang, 2021; Arranz et al., 2023), making them well suited for large-scale and dynamic road-state monitoring. This motivates the problem of persistent multi-UAV road-network monitoring, where a fleet of UAVs must coordinate online to maintain effective and adaptive coverage over a changing urban network.

This problem is substantially more difficult than conventional UAV routing or static coverage planning. Urban road networks are naturally graph-structured: UAVs traverse edges and make routing decisions at nodes, so each decision constrains future reachable segments and creates strong temporal dependencies. A centralized controller could in principle coordinate all UAVs jointly, but its computational burden grows rapidly with both fleet size and look-ahead horizon, making multi-step planning in realistic settings difficult to scale. Centralized control is also vulnerable to disruptions, since delays or failures of individual UAVs can undermine fleet-level plans. These limitations motivate decentralized control, in which each UAV acts independently. Yet decentralization creates its own challenges: without explicit coordination, UAVs may repeatedly cover the same areas, while the monitoring value of each road segment changes over time as previously collected information becomes stale. Each UAV must therefore reason not only about the evolving monitoring priorities of the network, but also about the likely future behavior of its teammates.

A key feature of decentralized control is asynchronous execution: UAVs make decisions at their own decision epochs rather than following a shared global clock. This improves robustness, removes synchronization bottlenecks, and allows continuous adaptation to local observations. However, it also makes coordination harder, since UAVs cannot directly observe one another's next actions at

the same decision step. The challenge becomes even greater in realistic urban environments where monitoring UAVs may coexist with background operational UAV traffic, such as delivery or inspection UAVs, which further increases environmental non-stationarity. Existing asynchronous coordination approaches, including heuristics, auction-based mechanisms, and game-theoretic methods, provide useful insights but often rely on hand-crafted logic, simplified interaction assumptions, or optimization procedures that scale poorly in large and dynamic graph-based environments.

To address these challenges, we propose a hybrid planning-learning framework for persistent multi-UAV road-network monitoring. The framework is built on three components. First, we adopt a decentralized MAPPO backbone (Yu et al., 2022) under centralized training and decentralized execution (CTDE), enabling UAVs to learn cooperatively with global information during training while acting independently from local observations at execution time. Second, we enhance each UAV with rolling-horizon PUCT search (Rosin, 2011; Silver et al., 2017) at its own decision epochs. Instead of directly following the policy output, each UAV performs a short-horizon local search over the road graph, guided by policy priors and critic-based leaf evaluation. To improve coordination under asynchronous execution, UAVs broadcast planned short-horizon trajectories, allowing others to use the most recently shared trajectories as coordination signals during search. Third, we introduce a search-distillation mechanism that feeds planning improvements back into the shared policy through an auxiliary loss, helping the policy gradually internalize the coordination and foresight benefits of online search. In this way, the proposed framework combines decentralized MARL, coordination-aware local planning, and policy distillation to improve monitoring effectiveness while preserving tractability in large-scale asynchronous multi-UAV systems.

2 Problem Formulation

We study persistent multi-UAV road network monitoring over a graph $G = (V, E)$, where each node $v \in V$ denotes a road intersection and each edge $e \in E$ denotes a traversable road segment. A fleet of monitoring UAVs repeatedly patrols this graph to refresh road knowledge over time. Unlike one-shot coverage problems, the objective here is persistent monitoring: the information associated with a road segment becomes stale as time evolves, so the UAV fleet must continuously revisit the road network to maintain high monitoring quality. Each road segment $e \in E$ is associated with a physical length ℓ_e, a time-varying road-knowledge value $K(e, t)$, and a set of discretized sub-segments S_e. The sub-segment decomposition enables localized modeling of visibility and partial monitoring coverage within a single road segment. In urban environments, visibility may be incomplete due to road curvature or occlusions from surrounding structures. By partitioning a road segment into sub-segments, the system can track which portions are actually visible and effectively monitored during UAV traversal.

Each UAV follows simple deterministic motion dynamics. At time t, UAV i is characterized by a state $\mathbf{x}_i(t) = [p_i(t), b_i(t)]$ where $p_i \in \mathbb{R}^2$ denotes its current position and b_i its battery level. In addition, each UAV is associated with deterministic parameters such as a nominal speed ν and sensing radius r. The sensing footprint is modeled as a circular region centered at the current UAV position. The effective monitoring region also depends on whether a UAV is traversing a road segment or located at an intersection. While flying along a segment, the UAV mainly observes a local monitoring window around its current position on that road. In contrast, when the UAV reaches an intersection, it can additionally observe the entering or exiting portions of all road segments connected to that intersection. Additionally, UAVs are subject to battery constraints. During flight, battery decreases at a constant discharge rate γ_b, while charging at a station restores energy at constant recharge rate γ_c. A UAV will be forced to return to charging station whenever its remaining battery is no longer sufficient to reach the nearest charging station while maintaining a predefined safety margin upon arrival. Importantly, the return trajectory still contributes to monitoring, since the UAV continues to observe road segments while traveling back to a charging station. Charging stations are assumed to have unlimited service capacity, so recharging can begin immediately upon arrival.

The monitoring process is governed by a road knowledge field defined on the graph. For each road segment e, the edge-level knowledge is aggregated from its sub-segments as

$$K(e, t) = \frac{1}{|S_e|} \sum_{s \in S_e} K(s, t). \tag{1}$$

Knowledge evolves through two separate processes: information decay and information acquisition. When a segment is not actively monitored, its knowledge decreases over time, reflecting

the fact that traffic conditions change and stale information becomes less reliable. When a UAV effectively monitors a segment or part of it, the corresponding knowledge is refreshed. In addition to controllable monitoring UAVs, the system may also receive information from background information sources that are accessible but not directly controllable by the service provider. In our experiments, these background sources are instantiated as delivery UAVs following shortest-path routes for transportation tasks. More generally, the same abstraction may represent other external sensing sources, such as loop detectors, camera-equipped vehicles, or third-party sensing platforms. Whenever such a source provides valid observation over a sub-segment, the corresponding knowledge is also updated. Each segment is assigned an edge-specific decay parameter α_e, which depends on local historical traffic conditions. In this work, the knowledge dynamics for sub-segments can be written as

$$K(s, t + \Delta t) = \begin{cases} 1 & \text{if monitored} \\ K(s,t) - \alpha_e \Delta t & \text{otherwise} \end{cases} \tag{2}$$

The aggregated knowledge of sub-segments then determines the knowledge of the full road segment. This formulation allows different road segments to exhibit different rates of information loss while also accommodating multiple heterogeneous sources of information acquisition.

UAVs make decisions asynchronously on the graph. When the UAV i reaches an intersection, it selects an outgoing edge from its current node and traverses the corresponding road segment. Since road segments have heterogeneous lengths, different actions incur different travel times. The decision process is therefore event-driven rather than synchronized across the fleet. A monitor action of one UAV is thus treated as an edge-traversal decision rather than an instantaneous control step: the corresponding reward is accumulated during traversal and settled only when the UAV reaches the destination node.

Under decentralized execution, each monitoring UAV selects actions based on local information only, while training may exploit richer centralized state information. The resulting problem is an asynchronous multi-agent graph decision problem with graph-structured reachability, temporally decaying monitoring value, battery-constrained motion, and decentralized coordination under partial observability.

3 Methodology

Overall Framework

The framework combines three tightly coupled components: a shared decentralized policy for action selection, a centralized critic for state evaluation, and a rolling-horizon PUCT search planner that provides short-horizon planning guidance to improve and shape the shared policy. Algorithm 1 introduces the pseudocode of the framework.

All monitoring UAVs share a single policy network. Once reaching an intersection, the UAV i constructs its local observation o_i^t, and the shared policy outputs logits over the feasible outgoing edges of the current road node. A softmax on these logits defines the decentralized action distribution $\pi_\theta(a_i^t \mid o_i^t)$ which represents the policy's preference over the candidate next-edge actions. Thus, the shared policy is the component that directly scores and gives a probability of feasible next moves based on local information. The centralized critic plays a different role. Instead of selecting actions, it evaluates how favorable the current state is for the long-term monitoring objective. To this end, the critic receives a richer global state representation than the policy, including fleet-level information. The critic therefore provides value estimates for policy learning during training. These value estimates support policy optimization during training and are also used in rolling-horizon PUCT search to evaluate leaf states reached by simulated short-horizon trajectories.

Together, once a UAV reaches an intersection, it first queries the shared policy using its local observation to obtain a probability distribution over feasible next-edge actions. Rather than directly executing the action from this distribution, the UAV uses these policy probabilities as priors to initialize a local PUCT tree search over the road graph. The search then explores promising branches guided by these priors, while leaf nodes are evaluated by the critic instead of by expensive rollouts. Algorithm 2 gives pseudocode of PUCT search. After multiple search iterations, the planner returns a short-horizon trajectory together with a search-improved preference over feasible next-edge actions. The UAV executes only the first action selected from this search result, and replans from a fresh observation at the next intersection.

In order to let the policy network learn from the PUCT results, the search results are distilled back into the shared policy, so that the policy gradually absorbs the corrections discovered by PUCT.

In this way, the framework forms a closed planning-learning loop: the shared policy proposes promising local action distributions, PUCT refines them through short-horizon search with critic-based evaluation, and the resulting search preference is fed back to improve the policy itself. The following three subsections detail these components in turn: the shared actor, the centralized critic, and the rolling-horizon PUCT search.

Algorithm 1 MAPPO-Search Framework

Require: Road network G, shared policy π_θ, centralized critic V_ϕ

Initialize policy parameters θ and critic parameters ϕ
Initialize rollout buffer $\mathcal{B}$
for each training episode **do**
    Reset environment
    Initialize recurrent hidden states for all monitoring UAVs
    Initialize pending transitions for all monitoring UAVs as empty
    while episode not finished **do**
        Advance environment by one time frame
        Update road knowledge decay
        Move all UAVs along their current edge
        for each monitoring UAV i **do**
            Accumulate monitoring reward along the currently traversed edge
            if UAV i reaches the destination node of its current edge **then**
                Construct next local observation o_i'
                Aggregate the accumulated reward r_i
                Finalize the pending transition of UAV i using $(o_i', r_i, \Delta t)$
                Store the completed transition in $\mathcal{B}$
            end if
        end for
        for each monitoring UAV i that reaches an intersection **do**
            Construct current local observation o_i
            $(a_i, p_i^{\text{search}}, \text{traj}_i) \leftarrow \text{PUCT}(o_i, s_i, \pi_\theta, V_\phi)$
            Broadcast the planned trajectory traj_i to other monitoring UAVs
            Execute only the first action a_i
            Start a pending transition for UAV i with (o_i, a_i)
        end for
        if update condition is met **then**
            Compute time-aware GAE advantages and return targets from $\mathcal{B}$
            Update θ and ϕ
            Clear $\mathcal{B}$
        end if
    end while
end for

Shared Policy

A single policy network is shared across all monitoring UAVs to generate action scores over feasible outgoing edges. These action scores are then normalized into a probability distribution over feasible next-edge actions. Once some UAV reaches intersection, it first aggregates information according to the graph structure:

$$z(v,t) = \text{GNN}\big(v,\ \{x_v^t = \big[\deg_{\text{out}}(v),\ \deg_{\text{in}}(v),\ \bar{k}(v,t)\big]\}_{v\in V},\ G\big). \tag{3}$$

Here $x_v^t = \big[\deg_{\text{out}}(v), \deg_{\text{in}}(v), \bar{k}(v,t)\big]$ denote the node features. $\deg_{\text{out}}(v)$ and $\deg_{\text{in}}(v)$ are the outgoing and incoming degrees of node v, and $\bar{k}(v,t)$ is the mean knowledge of road segments incident to node v. A pooled global graph embedding is then obtained as

$$g^t = \text{Pool}\big(\{z(v,t) : v \in V\}\big), \tag{4}$$

which summarizes the current graph-level monitoring context. For UAV i located at node v_i^t, we define the current-node feature vector

$$nf_i^t = \Big[b_i^t,\ \deg_{\text{out}}(v_i^t),\ \deg_{\text{in}}(v_i^t),\ t, z(v_i^t, t)\Big], \tag{5}$$

where b_i^t is the current battery level. The actor also constructs neighboring-node features and candidate-edge features. For each neighboring node u, neighbor-node features $nnf_i^t(u)$ contains node embeddings of u and its mean knowledge of road segments incident to node u:

$$nnf_i^t(u) = \big[z(u,t),\ \bar{k}(u,t)\big]. \tag{6}$$

For the corresponding outgoing edge $e = (v_i^t, u)$, edge features are represented as

$$ef_i^t(e) = \big[\ell_e,\ k_e(t),\ \alpha_e\big], \tag{7}$$

where ℓ_e is the edge length, $k(e,t)$ is the current knowledge of edge e, and α_e is the decay rate.
The policy network maintains a recurrent hidden state to preserve short-term temporal context across asynchronous decisions. Let $h_i^{t,\text{in}}$ denote the recurrent state carried into decision epoch t. The policy network first encodes the current ego-level context by combining the current-node feature nf_i^t with the pooled global graph embedding g^t, and feeds the result together with $h_i^{t,\text{in}}$ into the GRU:

$$h_i^{t,\text{out}} = \text{GRU}\Big([nf_i^t,\ g^t],\ h_i^{t,\text{in}}\Big). \tag{8}$$

The updated hidden state $h_i^{t,\text{out}}$ serves as a compact summary of the UAV's recent decision history and current graph-aware context. The policy then combines this updated hidden representation with the corresponding neighboring-node feature $nnf_i^t(u)$ and edge feature $ef_i^t((v_i^t, u))$ to compute an action logit:

$$q_i^t(u) = f_\theta\big(h_i^{t,\text{out}},\ nnf_i^t(u),\ ef_i^t((v_i^t, u))\big). \tag{9}$$

These logits score the feasible outgoing edges of the current node. Finally, a softmax over all feasible logits defines the decentralized action distribution

$$\pi_\theta(a_i^t \mid o_i^t) = \text{softmax}\big(\{q_i^t(u) : u \in \mathcal{N}(v_i^t)\}\big). \tag{10}$$

In this way, the shared policy first summarizes graph-level and ego-level information through the recurrent hidden state, and then evaluates each feasible next-edge action using candidate-specific neighboring-node and edge features.

Centralized Critic

The centralized critic evaluates how favorable the current state is for the long-term monitoring objective. To this end, it receives a richer global state representation than the decentralized policy under CTDE. These value estimates support policy optimization during training and are also reused in rolling-horizon PUCT search to evaluate leaf states reached by simulated short-horizon trajectories.
The critic state is represented as

$$s_t = \big[g^t,\ \text{graph_stats}_t,\ \text{fleet}_t,\ t\big]. \tag{11}$$

The first term g^t is the pooled global graph embedding produced by the shared graph encoder and provides a compact summary of the current graph-level monitoring context. The graph-level statistics are defined as

$$\text{graph_stats}_t = \big[c_t,\ \tilde{c}_t,\ \mu_k(t),\ \sigma_k^2(t),\ r_{\text{low}}(t)\big], \tag{12}$$

where c_t is the coverage ratio, $\tilde{c}_t$ is the weighted coverage ratio, $\mu_k(t)$ is the mean edge knowledge, $\sigma_k^2(t)$ is the variance of edge knowledge, and $r_{\text{low}}(t)$ is the proportion of low-knowledge edges in the network. The fleet summary is constructed as

$$\text{fleet}_t = \bigoplus_{i=1}^{M}\Big[z(v_i^t, t),\ b_i,\ \mathbf{1}\{\text{moving}_i^t\}\Big], \tag{13}$$

where M is the number of monitoring UAVs, and $\mathbf{1}\{\text{moving}_i^t\}$ is an indicator that equals 1 if UAV i is currently available for controlled movement, and 0 otherwise. In particular, $\mathbf{1}\{\text{moving}_i^t\} = 0$

when the UAV is charging or is committed to a return-to-charge trajectory, during which its motion is no longer controlled by the policy. The concatenated critic state s_t is then passed through a multilayer perceptron to produce the scalar value estimate

$$V_\phi(s_t) = \mathrm{MLP}_\phi(s_t), \tag{14}$$

which summarizes the long-term monitoring utility of the current global state. In this way, the centralized critic maps graph-level monitoring statistics, fleet-level states, and temporal context into a global value estimate while decentralized execution is preserved at test time.

Algorithm 2 Rolling-Horizon PUCT Search

Require: Priors p, critic network V_ϕ

$v \leftarrow$ root
for $n = 1$ to N **do**
  while v is expanded and not terminal **do** ▷ Selection
    Select child c maximizing $Q(c) + c_{\mathrm{puct}} P(c) \frac{\sqrt{N(\mathrm{parent}(c))}}{1+N(c)}$
    $v \leftarrow c$
  end while
  if v is non terminal **then** ▷ Expansion
    for each of its neighbor node **do**
      Create child node
    end for
    Select one expanded child for evaluation
  end if
  if v violates battery safety **then** ▷ Evaluation
    value $\leftarrow -\infty$
  else
    Construct critic state s_v at node v
    Evaluate leaf: value $\leftarrow U_{\mathrm{acc}}(v) + \lambda_{\mathrm{term}} V_\phi(s_v)$
  end if
  while $v \neq \varnothing$ **do** ▷ Backpropagation
    $N(v) \leftarrow N(v) + 1$
    if value is finite **then**
      $W(v) \leftarrow W(v) +$ value
    end if
    $v \leftarrow \mathrm{parent}(v)$
  end while
end for
Return $a^\star$ corresponding to the most visited root child

Training Objective

The proposed framework is trained with a MAPPO objective augmented by search distillation. Environment rewards remain the primary learning signal, while planner preferences provide an auxiliary supervision signal that transfers short-horizon search improvements back into the shared policy.

Because monitor-UAV actions are asynchronous and have heterogeneous durations, temporal discounting must respect elapsed simulated time rather than assume unit time steps. Let r_t be the reward after action a_t, $V_\phi(s_t)$ the critic estimate at the current state, $V_\phi(s_{t+1})$ the critic estimate at the next state, Δt_t the action duration, and m_t a transition mask. We define a time-aware discount factor

$$\gamma_t = \gamma^{\Delta t_t}, \tag{15}$$

which ensures that future rewards are discounted proportionally to elapsed simulated time rather than to the number of decision steps. The temporal-difference residual is

$$\delta_t = r_t + m_t \gamma_t V_\phi(s_{t+1}) - V_\phi(s_t). \tag{16}$$

Generalized advantage estimation is then computed recursively as

$$\hat{A}_t = \delta_t + m_t \gamma_t \lambda \hat{A}_{t+1}, \tag{17}$$

and the return target is

$$\hat{R}_t = \hat{A}_t + V_\phi(s_t). \tag{18}$$

In implementation, this recursion is performed independently for each agent trajectory in the batch, since asynchronous transitions from different UAVs are interleaved in the rollout buffer.
The actor is optimized with the clipped PPO objective,

$$L_{\text{actor}}(\theta) = -\mathbb{E}_t\Big[\min\Big(w_t(\theta)\hat{A}_t,\ \text{clip}(w_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\Big)\Big], \tag{19}$$

where

$$w_t(\theta) = \frac{\pi_\theta(a_t \mid o_t)}{\pi_{\theta_{\text{old}}}(a_t \mid o_t)}. \tag{20}$$

As in standard PPO, clipping the policy ratio stabilizes optimization by preventing overly aggressive updates. In our implementation, log-probability differences are additionally clamped before exponentiation for numerical stability.
The centralized critic is trained by regressing toward the GAE-based return target:

$$L_{\text{critic}}(\phi) = \mathbb{E}_t\left[\left(V_\phi(s_t) - \hat{R}_t\right)^2\right]. \tag{21}$$

To stabilize value learning under heterogeneous global-state features, we normalize critic inputs using running mean and variance statistics. We also include an entropy regularizer

$$L_{\text{entropy}}(\theta) = \mathbb{E}_t[\mathcal{H}(\pi_\theta(\cdot \mid o_t))] \tag{22}$$

to maintain action diversity during training.
To distill planner preferences into the policy, we introduce an auxiliary search distillation loss. For each search-augmented transition, the planner returns a top-k set of candidate root actions together with search scores derived from root visit counts. Let $\mathcal{A}_t^{\text{top-}k}$ denote these candidate actions and let p_t^{search} be the normalized planner preference over them. We then define

$$L_{\text{distill}}(\theta) = -\mathbb{E}_t\left[\sum_{a \in \mathcal{A}_t^{\text{top-}k}} p_t^{\text{search}}(a) \log \pi_\theta(a \mid o_t)\right]. \tag{23}$$

Here, p_t^{search} is normalized only over the planner's top-k candidates, whereas $\pi_\theta(a \mid o_t)$ remains the policy probability under the full feasible action space. This partial cross-entropy encourages the policy to assign higher probability to planner-preferred actions without explicitly suppressing actions outside the top-k set. In practice, invalid padded top-k slots are masked before normalization, and distillation is applied only when the planner produces at least one feasible candidate, preventing numerical instability in all-infeasible cases.
The overall training objective is

$$L = L_{\text{actor}} + c_v L_{\text{critic}} - c_e L_{\text{entropy}} + c_d L_{\text{distill}}, \tag{24}$$

where c_v, c_e, and c_d are weighting coefficients for value regression, entropy regularization, and search distillation, respectively. The objective L is minimized jointly over the actor and critic parameters, with reward-driven reinforcement learning remaining primary and search distillation acting as an auxiliary supervision signal.

4 Case Study

Experimental Setup

We evaluate the proposed framework in a high-resolution simulation environment built on a real-world urban road network from the Eixample district of Barcelona, Spain. The network contains 805 intersection nodes and 1397 road segments, covering approximately 7.6 km^2. Dangling nodes with only one connection are removed to ensure a navigable topology. Since UAVs operate above the

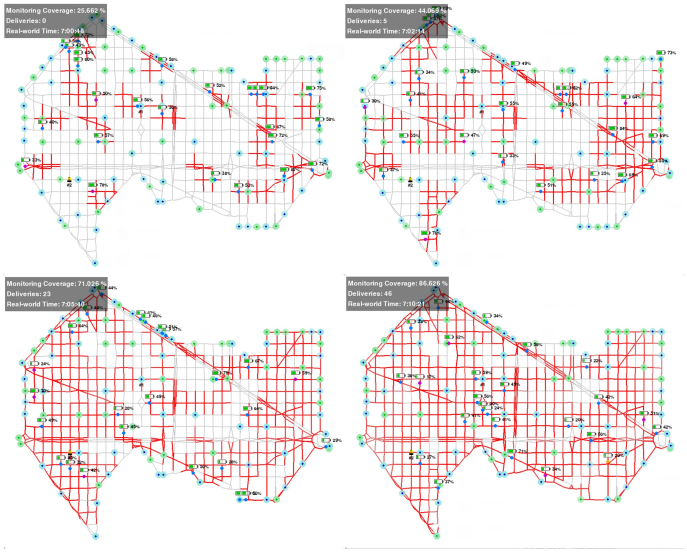


Figure 1: Simulation interface displaying the urban road network, monitoring UAVs (purple), delivery UAVs (blue), parcel tasks (pickup/delivery), charging stations, and dynamic knowledge states of road segments.

road network and are not constrained by one-way street regulations, all road segments are treated as bidirectional. The resulting graph has a dense connectivity structure with an average node degree of 6.9, providing a realistic and challenging testbed for large-scale multi-UAV coordination. Each simulation episode spans 6 hours of real-world operation.

We deploy a homogeneous fleet of UAVs. Each UAV flies at $\nu = 8m/s$, and monitors the road network with a circular sensing radius of $r = 30m$. Energy constraints are modeled explicitly. Initial battery levels are sampled uniformly from 30% to 90% to capture deployment variability. The battery consumption rate is set to $\gamma_b = 1.68\%/min$, corresponding to roughly one hour of operation under a full charge, while the charging rate is set to $\gamma_c = 4.8\%/min$, corresponding to a full recharge in approximately 20 minutes. Two charging stations are placed in the network. Initial UAV positions are randomly sampled over the network.

To model background information sources that are observable but not controllable by the monitoring system, we introduce an additional fleet of parcel-delivery UAVs. Delivery requests arrive dynamically according to a Poisson process with rate 0.8 orders per second. Because parcel-level origin-destination demand is unavailable, we generate OD pairs using calibrated human mobility flows for Barcelona from Aimsun (Casas et al., 2010). These parcel-delivery UAVs operate independently of the monitoring fleet and are not controlled by the proposed framework. Once assigned a delivery task, each delivery UAV follows the corresponding shortest path between its pickup and delivery locations. Although their trajectories are not optimized for persistent monitoring, they still contribute background observations while traversing the road network. In this way, they serve as an exogenous information source that partially refreshes road knowledge without participating in the coordinated monitoring policy.

Road knowledge evolves according to edge-specific decay rates α_e, which are derived from real traffic characteristics as described in Section 2. In the experiments, each edge is assigned a decay rate in the range $[0.001, 0.0016]$ s^{-1}. Intuitively, these values correspond to refresh intervals ranging from roughly 10 to 16 minutes, so roads with faster-changing traffic conditions require more frequent revisits than relatively stable roads.

Our method uses a rolling-horizon planner with horizon length as 5 minutes and 1,000 PUCT iterations at each decision epoch. Figure 1 illustrates the simulation environment, including the road network, UAV trajectories, parcel tasks, charging stations, and the evolving road-knowledge field.

Results and Analysis

Figure 2 compares two extreme fleet compositions under the same total fleet size of 15 UAVs: one with no dedicated monitoring UAVs and 15 delivery UAVs (M0-D15), and the other with 15 monitoring UAVs and no delivery UAVs (M15-D0). In Fig. 2(a), the weighted coverage curves rise and

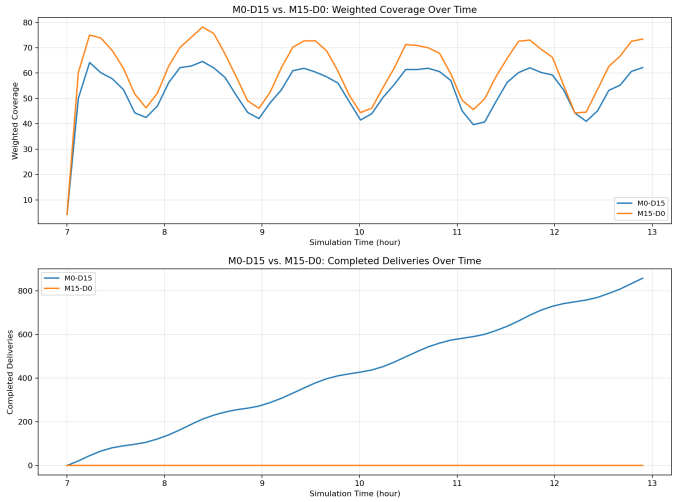


Figure 2: No-Monitoring v.s. No-Delivery (15 UAVs). Here, MX-DY denotes a fleet with X monitoring UAVs and Y delivery UAVs.

fall over time because persistent monitoring is governed by a refresh-decay cycle: road knowledge is refreshed when UAVs revisit road segments, but decays when segments remain unobserved. The top panel shows that the M15-D0 setting consistently achieves higher weighted coverage throughout the episode, indicating that dedicated monitoring UAVs are much more effective at sustaining persistent sensing quality than relying only on delivery-induced background observations. The oscillatory pattern also suggests that maintaining uniformly high coverage over the entire network requires sufficient revisit capacity; in practice, this means either more monitoring UAVs or more effective monitoring resources. This also provides indirect evidence that the learned monitoring policy is effective: when the fleet is fully allocated to controllable monitoring UAVs, the system is able to maintain substantially higher coverage over time, suggesting that the RL-trained policy successfully exploits dedicated monitoring resources rather than relying on incidental information collected by delivery traffic. The bottom panel further highlights the functional difference between the two fleet types: M0-D15 continuously completes delivery tasks, whereas M15-D0 produces no deliveries at all. Therefore, background delivery flows can contribute useful information, but they cannot substitute for dedicated monitoring resources when persistent monitoring is the primary objective.

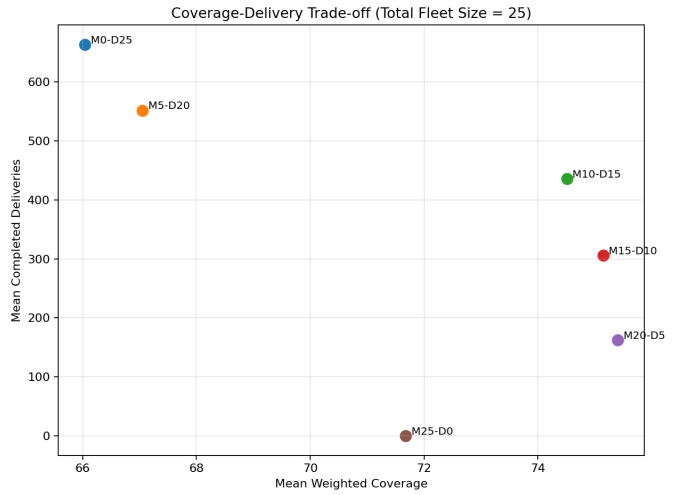


Figure 3: Performance for fixed fleet size. Here, MX-DY denotes a fleet with X monitoring UAVs and Y delivery UAVs.

Figure 3 summarizes the trade-off between weighted coverage and delivery performance for all fleet compositions with the same total number of 25 UAVs. Here MX-DY denotes a fleet with X monitoring UAVs and Y delivery UAVs. First, monitoring UAVs are not always better in larger numbers.

As the fleet composition shifts from delivery-dominant to monitoring-dominant, weighted coverage increases steadily, but the number of completed deliveries drops sharply. In particular, when too many UAVs are allocated to monitoring, delivery performance can collapse almost completely, as seen in the M25-D0 case. This shows that simply increasing the number of monitoring UAVs does not improve overall system performance when delivery service is also a core objective. Second, the monitoring benefit of adding more monitoring UAVs is itself subject to diminishing returns. The gain in weighted coverage becomes progressively smaller as the number of monitoring UAVs increases, especially once the system has already reached relatively high coverage levels. In other words, after a certain point, allocating additional UAVs to monitoring yields only limited sensing improvement while still incurring a substantial loss in delivery capacity. Third, having both types of UAVs is clearly better than having only one type. Mixed configurations such as M10-D15 or M15-D10 achieve strong coverage while still maintaining substantial delivery throughput, and therefore dominate the extreme cases in terms of overall balance. This indicates that the two UAV types play complementary roles: monitoring UAVs provide active persistent sensing, while delivery UAVs provide transportation service together with passive background information. The best operating point is therefore not located at either extreme, but in a coordinated mixed-fleet regime that balances monitoring quality and delivery efficiency.

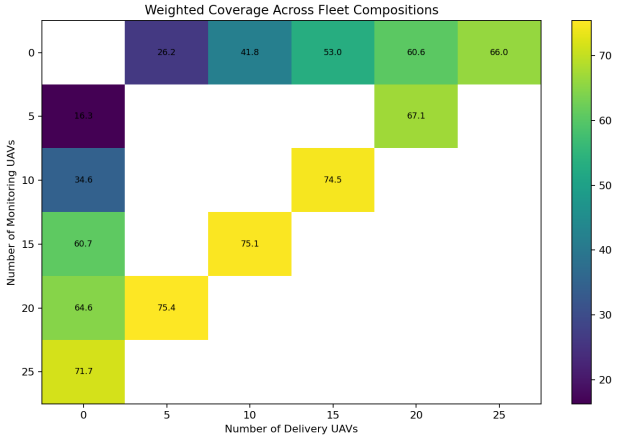


Figure 4: Heatmap of Weighted Coverage

Figure 4 shows the weighted coverage achieved by different combinations of monitoring UAVs and delivery UAVs. The dominant trend is that increasing the number of monitoring UAVs leads to a strong and consistent improvement in coverage, confirming that dedicated monitoring capacity is the main driver of sensing performance. At the same time, delivery UAVs also provide a positive contribution: for a fixed number of monitoring UAVs, adding delivery UAVs generally improves weighted coverage because these delivery UAVs traverse the network and refresh road knowledge even though they are not directly controlled by the monitoring policy. This means that delivery UAVs act as useful background information sources. An interesting pattern is that the marginal gain from adding more monitoring UAVs is not uniform across all settings, suggesting diminishing returns once monitoring density becomes sufficiently high. In several cases, a moderate number of monitoring UAVs combined with some delivery UAVs can already achieve coverage close to that of a monitoring-heavy configuration. Overall, the heatmap confirms that monitoring UAVs remain essential, but background delivery traffic can meaningfully complement the sensing process.

5 Conclusions

This paper presented a hybrid planning-learning framework for persistent multi-UAV road network monitoring. The proposed approach combines a shared decentralized MAPPO policy with PUCT search, allowing monitoring UAVs to retain the scalability of decentralized execution while benefiting from explicit short-horizon lookahead. By incorporating a coordination-aware local search score and traffic-informed road-knowledge dynamics, the framework is able to operate effectively in an asynchronous graph-based monitoring environment with battery and charging constraints. Preliminary results demonstrate the effectiveness of the proposed algorithm for persistent monitoring.

References

Arranz, R., Carramiñana, D., Miguel, G. d., Besada, J. A., & Bernardos, A. M. (2023). Application of deep reinforcement learning to uav swarming for ground surveillance. *Sensors*, *23*(21), 8766.

Casas, J., Ferrer, J. L., Garcia, D., Perarnau, J., & Torday, A. (2010). Traffic simulation with aimsun. In *Fundamentals of traffic simulation* (pp. 173–232). Springer.

Huang, H., Savkin, A. V., & Huang, C. (2021). Decentralized autonomous navigation of a uav network for road traffic monitoring. *IEEE Transactions on Aerospace and Electronic Systems*, *57*(4), 2558–2564.

Rosin, C. D. (2011). Multi-armed bandits with episode context. *Annals of Mathematics and Artificial Intelligence*, *61*(3), 203–230.

Savkin, A. V., & Huang, H. (2021). Navigation of a uav network for optimal surveillance of a group of ground targets moving along a road. *IEEE transactions on intelligent transportation systems*, *23*(7), 9281–9285.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... others (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., & Wu, Y. (2022). The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems*, *35*, 24611–24624.