Learning Collision-Free Dispatch Policies for Route-Wise Decision-Dependent Anomaly Detection

Xin Xia
University of Wisconsin-Madison
xxia65@wisc.edu

Chen Zhang
Tsinghua University
zhangchen01@tsinghua.edu.cn

Kaibo Liu*
University of Wisconsin-Madison
kliu8@wisc.edu

Abstract

Mobile sensing for real-time anomaly detection couples routing, sampling, and stopping: dispatch determines which locations are observed, while observations update the evidence used for future dispatch. We study this problem for a fleet of unmanned aerial vehicles (UAVs), where each UAV observes only its current location and anomalies may occur at unknown locations and times. The goal is to detect anomalies quickly while controlling false alarms under decision-dependent partial observations, local mobility, and collision-avoidance constraints. Existing quickest-detection methods typically ignore mobility and collision constraints, whereas learning-based routing methods scale to large fleets but are not designed for statistically calibrated detection. We formulate route-wise monitoring as a partially observable Markov decision process (POMDP) and propose DISPATCH-PPO, a deep reinforcement learning method for long-horizon detection-aware dispatch. Its policy architecture, DISPATCHNET, uses autoregressive decoding with dynamic feasibility masking to generate collision-free joint UAV moves without enumerating the exponential joint action space. We establish theoretical properties for persistent coverage, evidence-based focusing, and detection delay through a resource-allocation information-rate bound. Experiments show that DISPATCH-PPO reduces detection delay relative to heuristic and optimization-based baselines, while transferring effectively to real-robot and wildfire monitoring case studies.

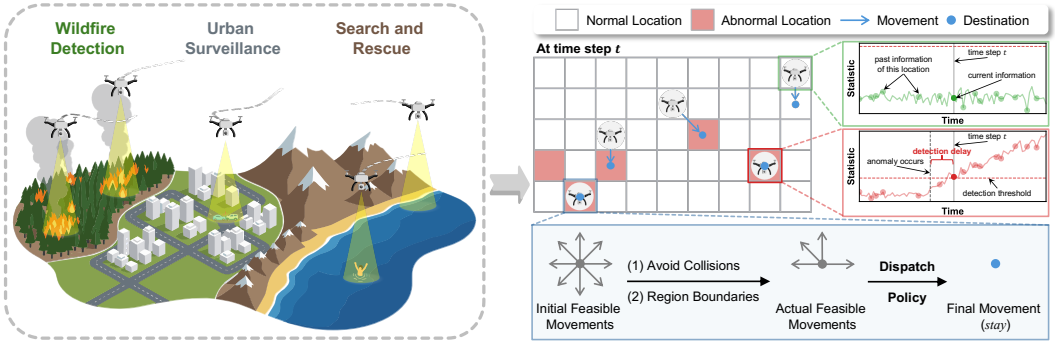
1 Introduction

Mobile sensing collects data sequentially at locations visited by sensors whose routes are controlled over time [1, 2]. For real-time anomaly detection, the dispatch policy determines which locations are observed, while the resulting observations update anomaly evidence that should guide future dispatch. Thus, routing, sampling, and stopping are tightly coupled: the objective is to detect anomalies that occur at unknown locations and times as quickly as possible while maintaining false-alarm control.

We study this problem in the context of route-wise monitoring with a fleet of unmanned aerial vehicles (UAVs), where each UAV observes only the location it is currently visiting. This problem is difficult because detection and dispatch cannot be designed separately. A small fleet can observe only a few locations per time step, making observations sparse and decision-dependent. Dispatch decisions affect both the information collected immediately and the information that may become available in the future. Mobility constraints further affect information acquisition, since one UAV can only move

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Real-world applications of UAV-based anomaly detection

(b) Constrained dispatch of a coordinated team of UAVs

Figure 1: Route-wise quickest detection with a coordinated UAV fleet.

a limited distance to neighboring locations per time step, and reaching a distant location requires a sequence of feasible moves. Thus, poor routing decisions can leave parts of the region unobserved for extended periods. With multiple UAVs, these long-horizon routing effects must be coordinated under hard feasibility constraints to prevent collisions and redundant sensing, which makes the joint action space combinatorially large. Consequently, quick detection requires a policy that plans routes for information acquisition while enforcing mobility and safety constraints.

Existing methods address important parts of this problem. Classical multivariate statistical process control (MSPC) provides sequential detection procedures with calibrated false-alarm control, commonly through the in-control average run length (ARL_0) [3, 4]. However, these methods typically assume instantaneous or unconstrained sampling and do not directly handle mobile, collision-constrained sensing. Conversely, informative routing, neural dispatch, and multi-robot coordination methods provide scalable tools for path planning and fleet control [5–18], but are not designed around calibrated stopping rules or quickest-detection objectives. This leaves open a framework that jointly supports false-alarm controlled detection, long-horizon decision-dependent sampling, scalable multi-UAV dispatch, and hard collision-free coordination.

We address this gap through a detection-aware dispatch framework for route-wise monitoring. The framework maintains a compact state that combines per-location anomaly evidence, visit information, and UAV positions, yielding a partially observable Markov decision process (POMDP) with decision-dependent observations. To optimize long-horizon dispatch, we develop DISPATCHPPO with DISPATCHNET, which factorizes multi-UAV action generation autoregressively, enabling the policy to directly construct feasible joint moves without enumerating the exponential joint action space. Dynamic feasibility masks enforce collision-free dispatch by construction, while the learned policy allocates sensing effort toward locations that are informative for detection. During execution, an exploration-mixture strategy provably guarantees persistent coverage and anomaly-focusing behavior, both of which are needed to discover new anomalies and confirm suspected ones. Detection thresholds are then calibrated to satisfy the desired ARL_0 false-alarm constraint.

Our contributions are threefold: (i) We formulate multi-UAV route-wise anomaly monitoring as a POMDP that combines mobility-constrained, decision-dependent partial observations with MSPC evidence statistics and ARL_0-calibrated false-alarm control. (ii) Building on this formulation, we develop DISPATCHPPO with DISPATCHNET, an autoregressive masked policy that samples collision-free joint UAV moves without enumerating the $|\mathcal{A}_{\mathrm{loc}}|^N$ action space and decodes in $O(K+N)$ time per decision step. (iii) We then analyze the executed policy, establishing persistent coverage, anomaly focusing, and detection-delay bounds under the proposed dispatch mechanism. Empirically, experiments across grid scales and fleet sizes show lower detection delay than benchmark policies while remaining orders of magnitude faster per step than look-ahead planning. The method is further evaluated in Robotarium experiments and a real wildfire monitoring case study.

2 Related Work

Table 1 positions our work relative to several related lines of research. MSPC and quickest anomaly detection provide principled sequential stopping rules with calibrated false-alarm control and delay-

Table 1: Related-work comparison. ✓: central focus, △ : partial support, ×: not addressed.

Literature Track	False-alarm control	Quickest detection	Scalable policy	Multi-UAV coordination	Hard collision-free joint moves
Classic MSPC [19–24]	✓	✓	△	×	×
Route-wise sampling [1, 2, 25, 26]	△	✓	×	△	×
Informative routing [5–7]	×	×	✓	×	×
Neural routing and dispatch [8–14]	×	×	✓	✓	×
Safe coordination [15–18]	×	×	△	✓	✓
Our Work	✓	✓	✓	✓	✓

oriented objectives [3, 4]. Under partial observability, adaptive sampling and bandit-style monitoring methods decide which streams to observe over time [19–24]. These methods capture false-alarm calibration and detection objectives, but typically model sampling as unconstrained switching, ignoring travel-time and collision constraints. Mobility-aware quickest-detection and route-wise sampling methods couple surveillance trajectories with detection criteria [1, 2, 25, 26]; however, they are often model-based, rely on parametric routing rules or short-horizon replanning, and do not focus on scalable policies that generate collision-feasible joint moves in dense multi-UAV environments.

A complementary literature studies scalable routing and coordination with different objectives. Informative path planning and active sensing optimize trajectories for mapping or estimation [5], with recent learning-based methods improving online replanning under uncertainty [6, 7]. Neural combinatorial optimization and deep reinforcement learning further provide scalable policies for vehicle routing and dynamic fleet dispatch [8–14], but generally optimize information-gain or logistics objectives rather than ARL-calibrated quickest detection. Collision-free multi-robot coordination methods explicitly enforce safety constraints [15–18], but are not designed to allocate sensing effort using false-alarm-controlled detection statistics. In summary, existing work does not jointly support false-alarm controlled detection, long-horizon decision-dependent sampling, scalable dispatch, and hard collision-free coordination. Our work integrates MSPC evidence accumulation, ARL_0 calibration, and route-wise dispatch into a policy that produces collision-free joint actions by construction.

3 Problem Formulation

Environment dynamics. We consider a region discretized into K locations (e.g., an $I \times J$ grid with $K = I \cdot J$). A fleet of N UAVs ($N \ll K$) patrols the region in discrete time. Let $p_t^n \in \{1, \dots, K\}$ be the location of UAV $n \in \{1, \dots, N\}$ at time t, and let $\boldsymbol{p}_t = (p_t^1, \dots, p_t^N)$ denote the fleet locations at time t. Each UAV selects a local action a_t^n from a fixed set $\mathcal{A}_{\mathrm{loc}} = \{\mathsf{stay}, \mathsf{north}, \mathsf{south}, \mathsf{east}, \mathsf{west}, \mathsf{NE}, \mathsf{NW}, \mathsf{SE}, \mathsf{SW}\}$. Let $\phi(p, a)$ denote the next location obtained by applying action a at location p. The joint action $\boldsymbol{a}_t = (a_t^1, \dots, a_t^N)$ deterministically induces the next configuration $\boldsymbol{p}_{t+1}$ with $p_{t+1}^n = \phi(p_t^n, a_t^n)$. For safety, we require *hard collision avoidance*: $p_t^n \neq p_t^{n'}$ for all $n \neq n'$ and all t. This couples the UAVs through a time-varying feasible joint-action set $\mathcal{A}(\boldsymbol{p}_t) = \Big\{ \boldsymbol{a} \in \mathcal{A}_{\mathrm{loc}}^N : \ \phi(p_t^n, a^n) \neq \phi(p_t^{n'}, a^{n'}) \ \forall n \neq n', \text{ and all moves stay in-region} \Big\}$.

At time t, an underlying measurement field $\boldsymbol{x}_t = (x_t^1, \dots, x_t^K)$ exists over all locations, but the fleet observes only the visited set $\mathcal{V}_t = \{p_t^1, \dots, p_t^N\}$ and receives measurements $\{(k, x_t^k) : k \in \mathcal{V}_t\}$. Initially, the system is *in-control*, i.e., no anomaly. At an unknown change time $\tau < \infty$, an unknown subset $\mathcal{B} \subseteq [K]$ becomes *out-of-control*. Following standard quickest-detection models, without loss of generality, we consider a persistent mean shift: for $t < \tau$, $x_t^k \sim \mathcal{N}(0, 1)$ (after standardization), while for $t \geq \tau$ and $k \in \mathcal{B}$, $x_t^k \sim \mathcal{N}(\mu_k, 1)$ with unknown μ_k satisfying $|\mu_k| \geq \mu_{\min} > 0$, where $\mu_{\min}$ refers to the minimum interested magnitude of detection, and locations $k \notin \mathcal{B}$ remain in-control.

Objective: quickest detection with false-alarm control. Let $\mathcal{H}_t$ denote the observation history up to time t. A dispatch policy π selects $\boldsymbol{a}_t \in \mathcal{A}(\boldsymbol{p}_t)$ based on $\mathcal{H}_t$. A sequential detection statistic $\Lambda_t = \Lambda(\mathcal{H}_t)$ summarizes accumulated evidence, and an alarm is raised at

$$T(H) = \inf\{t \geq 1 : \Lambda_t \geq H\}, \tag{1}$$

where the threshold $H > 0$ is chosen to meet a target average run length $\mathrm{ARL}_0(\pi, H) := \mathbb{E}_{\tau=\infty}^{\pi}[T(H)]$. The post-change performance is measured by the detection delay $\mathrm{ARL}_1(\pi, H) :=$

$\mathbb{E}^\pi[T(H) - \tau \mid T(H) > \tau,\ \tau < \infty]$. With a prescribed false-alarm level α, our objective is:

$$\min_{\pi,\, H} \ \mathrm{ARL}_1(\pi, H) \quad \text{s.t.} \quad \mathrm{ARL}_0(\pi, H) \geq \alpha.$$

Directly optimizing this objective is challenging in the route-wise setting. Specifically, Λ_t depends on decision-dependent partial observations and H must be calibrated under the long-run behavior induced by π. These factors motivate the POMDP formulation and learning approach in later sections.

Assumptions. **(A1)** The mobility graph over feasible locations and local moves is finite and connected, and the initial configuration $\boldsymbol{p}_0$ is collision-free. **(A2)** Motion is synchronous and local: $p^n_{t+1} = \phi(p^n_t, a^n_t)$, with boundary-violating moves excluded from the feasible action set. **(A3)** Collisions are defined at the cell level, i.e., $p^n_t \neq p^{n'}_t$ for all $n \neq n'$. In-transit conflicts, such as intersecting diagonal moves, are not modeled here and can be handled separately by safety controllers [15, 16]. **(A4)** Measurements $\{\boldsymbol{x}_t\}_{t\geq 1}$ are independent over time. Pre-change measurements are stationary and standardized, taken here as $\mathcal{N}(0,1)$; in practice, $\boldsymbol{x}_t$ can be defined as standardized residuals from a baseline model. After τ, affected locations $k \in \mathcal{B}$ undergo persistent mean shifts with $|\mu_k| \geq \mu_{\min} > 0$, while unaffected locations remain in control. Spatial correlation within $\boldsymbol{x}_t$ at a fixed time is allowed. **(A5)** The anomaly process $(\tau, \mathcal{B}, \{\mu_k\})$ is exogenous and independent of UAV actions. **(A6)** Sufficient in-control data (or the simulator) is available for offline training.

4 Decision-Dependent POMDP for Multi-UAV Anomaly Monitoring

We model route-wise dispatch as a POMDP $(\mathcal{S}, \mathcal{O}, \mathcal{A}, P, r, \gamma)$. The latent state $s_t \in \mathcal{S}$ includes the UAV configuration $\boldsymbol{p}_t$ and the (unobserved) anomaly state that governs the measurement field $\boldsymbol{x}_t$. At time t, the central controller (termed dispatcher) observes a feature vector $\boldsymbol{o}_t \in \mathcal{O}$ reflecting the latent state s_t, and selects a feasible joint action $\boldsymbol{a}_t \in \mathcal{A}(\boldsymbol{p}_t)$, and receives a reward r_t. The transition kernel P is induced by (i) deterministic motion $\boldsymbol{p}_{t+1}$ given $\boldsymbol{p}_t$ and $\boldsymbol{a}_t$, and (ii) the environment dynamics. We use a discount factor $\gamma \in (0,1)$ close to 1 to emphasize long-horizon routing effects.

Observation construction. To couple route-wise sampling with sequential detection, we maintain location-wise statistics and incorporate them in observation features $\boldsymbol{o}_t$ to reflect the location's status. For each location $k \in [K]$, we maintain a two-sided CUSUM statistic to detect a minimum mean shift $\mu_{\min} > 0$ [3, 4]. Let $z^k_t = \mathbb{1}\{k \in \mathcal{V}_t\}$ indicate whether k is observed at time t, and $w^{k,+}_t$ and $w^{k,-}_t$ denote the one-sided upper and lower CUSUMs, initialized at 0. If $z^k_t = 1$, the statistics are updated using the real measurement x^k_t:

$$w^{k,+}_t = \max\Big\{0,\ w^{k,+}_{t-1} + \mu_{\min} x^k_t - \tfrac{1}{2}\mu^2_{\min}\Big\}, \quad w^{k,-}_t = \max\Big\{0,\ w^{k,-}_{t-1} - \mu_{\min} x^k_t - \tfrac{1}{2}\mu^2_{\min}\Big\}.$$

If $z^k_t = 0$, we apply a small uncertainty inflation with $C^U > 0$ by setting $w^{k,+}_t = w^{k,+}_{t-1} + C^U$ and $w^{k,-}_t = w^{k,-}_{t-1} + C^U$. The abnormal evidence is summarized by $w^k_t = \max\{w^{k,+}_t, w^{k,-}_t\}$, which increases when location k shows sustained abnormal evidence or remains unobserved for a long time.

The observation vector combines the current sampling configuration $\boldsymbol{z}_t$, visit-history vector $\boldsymbol{u}_t$, and evidence scores $\boldsymbol{w}_t$, where u^k_t records the number of steps since location k was last visited and $u^k_t = 0$ if $z^k_t = 1$. Specifically, $\boldsymbol{o}_t = [z^1_t, \ldots, z^K_t;\ u^1_t, \ldots, u^K_t;\ w^1_t, \ldots, w^K_t] \in \{0,1\}^K \times \mathbb{Z}^K_{\geq 0} \times \mathbb{R}^K_{\geq 0}$.

Reward shaping. Directly optimizing ARL_1 is difficult because it depends on the calibrated stopping time. We therefore use a training-only proxy reward, $r_t = R^U_t + R^A_t$, where $R^U_t = -\frac{C^U}{K}\sum_{k=1}^K u^k_t$ encourages exploration by penalizing long-unvisited locations, and $R^A_t = \frac{1}{N}\sum_{n=1}^N r^n_t$ encourages post-change concentration on affected locations. Specifically,

$$r^n_t = \begin{cases} +1, & t \geq \tau,\ \mathcal{B} \neq \emptyset,\ p^n_t \in \mathcal{B}, \\ -1, & t \geq \tau,\ \mathcal{B} \neq \emptyset,\ p^n_t \notin \mathcal{B}, \\ 0, & t < \tau \text{ or } \mathcal{B} = \emptyset. \end{cases}$$

Here, τ and $\mathcal{B}$ are used only to define rewards in offline simulated training. At deployment, the policy observes only $\boldsymbol{o}_t$, and alarms are triggered only by the ARL_0-calibrated stopping rule.

System-level detection statistic. Following aggregation schemes in adaptive-sampling MSPC [2, 19], we define $\Lambda_t = \sum_{i=1}^N \hat{w}^{(i)}_t$, where $\hat{w}^{(1)}_t \geq \cdots \geq \hat{w}^{(K)}_t$ are the sorted values of $\{w^k_t\}_{k=1}^K$. According to Equation (1), the procedure stops at $T(H) = \inf\{t \geq 1 : \sum_{i=1}^N \hat{w}^{(i)}_t > H\}$.

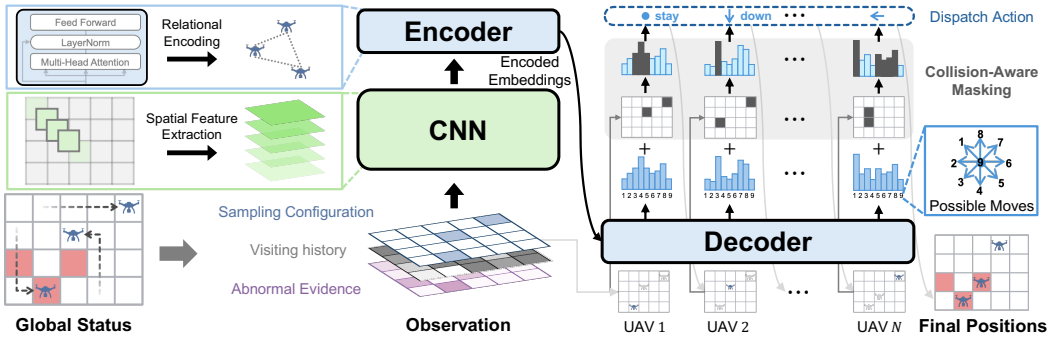


Figure 2: DISPATCHNET architecture. The encoder extracts spatial features and inter-UAV interactions from $(\boldsymbol{o}_t)$. An autoregressive decoder then samples per-UAV moves with a dynamic feasibility mask, producing a collision-free joint action without enumerating $\mathcal{A}(\boldsymbol{p}_t)$.

5 Learning-Based Dispatch Policy Optimization

In the POMDP of Section 4, the Dispatcher seeks a policy π that maximizes the discounted return up to $T(H)$, $\mathcal{J}(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^{T(H)} \gamma^t r_t\right]$. Direct optimization is intractable because the detection threshold H depends on the deployed policy and is unavailable during training. This motivates us to propose a two-stage design: Stage I learns a collision-feasible policy π_θ offline using the shaped proxy reward, and Stage II deploys an executed policy π^{exec} that augments π_θ with persistent exploration and effective anomaly focusing. We then calibrate H under π^{exec} to satisfy the target ARL_0.

5.1 Stage I: Offline Learning via DISPATCHPPO

In Stage I, we learn a long-horizon dispatch policy using fixed-horizon simulation episodes of length T_{ep}. Each episode samples an anomaly scenario (e.g., τ and $\mathcal{B}$) and generates measurements and proxy rewards accordingly. Tabular methods are infeasible due to high-dimensional observations and the combinatorial joint action space under collision constraints. We therefore parameterize the dispatcher as a neural policy $\pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{o}_t, \boldsymbol{p}_t)$ and maximize the discounted return $\mathcal{J}(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T_{\text{ep}}-1} \gamma^t r_t\right]$.

DISPATCHNET: Collision-free autoregressive policy. The dispatcher selects a joint move $\boldsymbol{a}_t = (a_t^1, \ldots, a_t^N)$ subject to local mobility and collision-avoidance constraints. Parallel per-UAV sampling may produce infeasible moves, while enumerating all joint moves scales exponentially as $|\mathcal{A}_{\text{loc}}|^N$ with $|\mathcal{A}_{\text{loc}}| = 9$. DISPATCHNET addresses this by using the encoder-decoder architecture in Figure 2, with autoregressive decoding and dynamic feasibility masking.

Encoder: Given the grid observation $\boldsymbol{o}_t$, the encoder first produces a spatial feature map via a CNN [27] block $\boldsymbol{H}_t = f_{\text{cnn}}(\boldsymbol{o}_t) \in \mathbb{R}^{I \times J \times d}$. For UAV n, we form a token by indexing $\boldsymbol{H}_t$ at its current position and adding identity and position embeddings:

$$\boldsymbol{e}_t^n = \boldsymbol{H}_t[p_t^n] + \boldsymbol{e}_{\text{id}}(n) + W_{\text{pos}}[i_t^n, j_t^n]^\top \in \mathbb{R}^d, \qquad n \in [N],$$

where (i_t^n, j_t^n) are the grid coordinates of p_t^n. A Transformer [28] encoder block then models inter-UAV interactions (e.g., congestion and mutual exclusion),

$$(\boldsymbol{h}_t^1, \ldots, \boldsymbol{h}_t^N) = f_{\text{tr}}(\boldsymbol{e}_t^1, \ldots, \boldsymbol{e}_t^N), \qquad \boldsymbol{h}_t^n \in \mathbb{R}^d.$$

Autoregressive decoder: The decoder assigns actions sequentially for $n = 1, \ldots, N$. Autoregressive decoding is essential here because hard collision constraints are combinatorial and cannot be enforced by parallel per-UAV visiting. For UAV n, the decoder outputs logits $\boldsymbol{\ell}_t^n \in \mathbb{R}^{|\mathcal{A}_{\text{loc}}|}$ using $\boldsymbol{h}_t^n$ and an autoregressive context. In our implementation, the context is an embedding of the previous action:

$$\boldsymbol{v}_t^n = \left[\boldsymbol{h}_t^n;\ \text{Emb}_{\text{act}}(a_t^{n-1})\right], \qquad \boldsymbol{\ell}_t^n = g_{\text{dec}}(\boldsymbol{v}_t^n),$$

with a fixed start token for $n = 1$. Let $\mathcal{A}_{\text{base}}^n(\boldsymbol{o}_t) \subseteq \mathcal{A}_{\text{loc}}$ denote per-UAV base feasibility (e.g., boundary constraints). To prevent collisions during decoding, we reserve (i) destinations already

assigned to earlier UAVs ($1 \le i < n$), and (ii) the current positions of UAVs not yet decoded ($n < j \le N$), and construct a reservation set $\mathcal{R}_t^n$ for UAV n:

$$\mathcal{R}_t^n = \{\hat{p}_{t+1}^i = \phi(p_t^i, a_t^i) : 1 \le i < n\} \cup \{p_t^j : n < j \le N\}.$$

This construction prevents both direct collisions and position swaps, while guaranteeing that every UAV always retains at least the stay action as feasible. The decoder's feasible action set for UAV n is

$$\mathcal{A}_t^n(\boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1}) = \big\{a \in \mathcal{A}_{\text{base}}^n(\boldsymbol{o}_t) : \phi(p_t^n, a) \notin \mathcal{R}_t^n\big\}.$$

Define the mask $m_t^n(a) = \mathbb{1}\{a \in \mathcal{A}_t^n(\boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1})\}$. We sample a_t^n from the masked categorical distribution

$$\tilde{\pi}_\theta(a_t^n = a \mid \boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1}) = \frac{m_t^n(a)\exp(\ell_t^n(a))}{\sum_{a' \in \mathcal{A}_{\text{loc}}} m_t^n(a')\exp(\ell_t^n(a'))}, \quad a \in \mathcal{A}_{\text{loc}}.$$

Algorithm 1 gives the full DISPATCHNET procedure, and Proposition 1 shows that its autoregressive decoder generates collision-free actions in $O(K+N)$ time. Details are in Appendices A.1 and B. We use DISPATCHNET as the policy π_θ and optimize it with a Proximal Policy Optimization (PPO)-style algorithm [29], denoted DISPATCHPPO. Using PPO for stable and scalable training, DISPATCHPPO's main contribution is its feasibility-aware autoregressive policy architecture, which is independent of the specific policy-gradient optimizer. Full implementation details are provided in Appendix A.2.

5.2 Stage II: Online Execution with Exploration-Mixture

Stage II defines the executed policy π^{exec} for deployment (online anomaly detection under false-alarm control) by reweighting π_θ learned from Stage I within the decoder's masked feasible sets $\{\mathcal{A}_t^n\}_{n=1}^N$. The goal is to enforce (i) persistent coverage of the entire region and (ii) an evidence-based focusing that increases repeated visiting on anomalous locations while remaining collision-free.

At time t, when decoding UAV n given the partial assignment $a_t^{1:n-1}$, let $k = p_t^n$ and $\mathcal{N}_t^n(k)$ be the feasible destination locations. Define the local evidence gap as $\Delta_t^{\text{loc}}(n;k) = w_t^k - \max_{k' \in \mathcal{N}_t^n(k)\setminus\{k\}} w_t^{k'}$. A stay bias at k is then computed via a sigmoid mapping: $\xi_t^n(k) = \frac{1}{1+\exp\left(-\Delta_t^{\text{loc}}(n;k)\right)} \in [0,1]$. Accordingly, we construct a bias-adjusted policy π_θ^{lock} by assigning probability $\xi_t^n(k)$ to the stay action, and renormalizing the remaining feasible-action probabilities to sum to $1 - \xi_t^n(k)$, which thus preserves the relative preferences among non-stay moves from Stage I. When w_t^k is locally dominant, $\xi_t^n(k)$ increases, biasing the UAV to stay and resample k.

To prevent any region from being permanently ignored, we introduce an exploration scheme with a fixed ϵ_{IC}-mixture of π_θ^{lock} and the uniform distribution over the masked feasible set $\mathcal{A}_t^n$:

$$\pi^{\text{exec}}\big(a_t^n \mid \boldsymbol{o}_t, \boldsymbol{p}_t, a_t^{1:n-1}\big) = (1-\epsilon_{\text{IC}})\,\pi_\theta^{\text{lock}}\big(a_t^n \mid \boldsymbol{o}_t, \boldsymbol{p}_t, a_t^{1:n-1}\big) + \epsilon_{\text{IC}}\,\text{Unif}(\mathcal{A}_t^n), \tag{2}$$

where exploration rate $\epsilon_{\text{IC}} \in (0,1)$. Since both terms are supported on $\mathcal{A}_t^n$, π^{exec} always selects feasible actions and remains collision-free.

5.3 Theoretical Properties of the Executed Policy

For any reference time $t_0 \ge 0$, define the normalized long-run visitation frequency $q_k(t_0) := \liminf_{T\to\infty} \frac{1}{N(T-t_0)} \sum_{t=t_0}^{T-1} z_t^k$, and hit rate $\rho_k(t_0) := N q_k(t_0) = \liminf_{T\to\infty} \frac{1}{T-t_0} \sum_{t=t_0}^{T-1} z_t^k$. Here $\rho_k(t_0)$ is the asymptotic fraction of time steps at which k is observed, and $q_k(t_0)$ is the corresponding fraction of the fleet's N resources allocated to k after t_0.

Theorem 1 (Persistent Coverage)**.** *Suppose $C^U > \mu_{\min}|\mu_k| - \frac{1}{2}\mu_{\min}^2$ for every $k \in [K]$. As $H \to \infty$, the following hold with probability 1: (i) every location $k \in [K]$ is visited almost surely ; and (ii) every location has a positive long-run visitation frequency .*

For proof, see Appendix C.1. Theorem 1 shows that the executed dispatch policy enforces persistent exploration, i.e, no region is permanently ignored, which is essential for controlling false alarms under unknown anomaly locations. On the contrary, after a change occurs, rapid detection requires *concentrating* observations on truly affected locations. We formalize this behavior in Theorem 2.

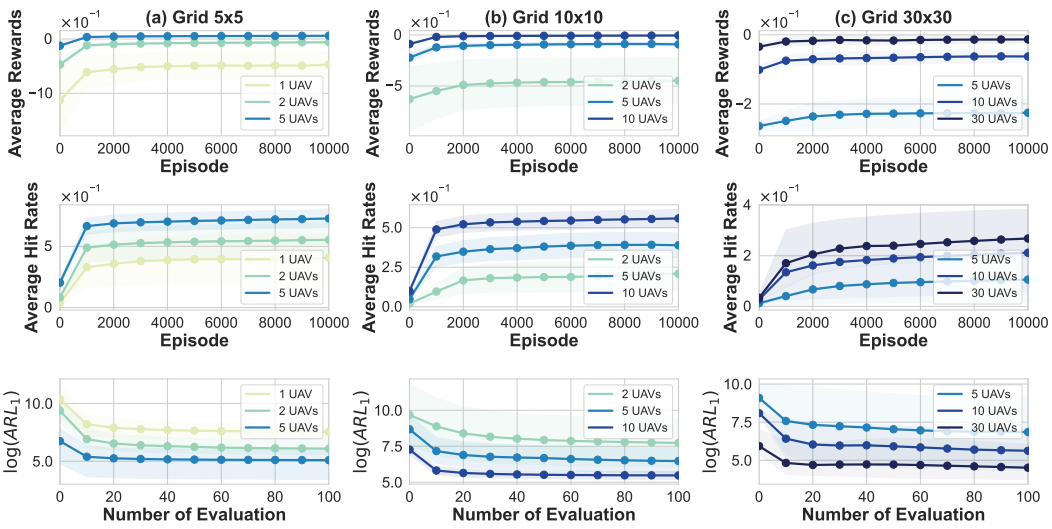


Figure 3: Stage I training and validation curves for DISPATCHPPO. *Top & Middle:* mean per-step proxy reward and anomaly hit rate over 10,000 training episodes (T_{ep}=200). *Bottom:* proxy detection delay $\log(\text{ARL}_1)$ from periodic validation (every 100 episodes; 1,000 rollouts; horizon 2,000) using a fixed pseudo-threshold $\hat{H}$. In each setting, the number of anomalous locations equals N.

Theorem 2 (Evidence-Based Focusing)**.** *Suppose* $\mathcal{B} = \{b \in [K] : \mu_{\min}|\mu_b| - \frac{1}{2}\mu_{\min}^2 > C^U\}$*, and let* $k \in \mathcal{B}$ *be an affected location that is no neighbor of* k *is in* $\mathcal{B}$*. If location* k *is visited after change with* $\rho_k(\tau) > 0$*, then as* $t \to \infty$ *through* $\{t : z_t^k = 1\}$*, one has:* $\xi_t^{n_t(k)}(k) \to 1$*. Here* $n_t(k)$ *denotes the unique UAV index located at* k *at time* t*. Consequently, the executed policy's one-step* **stay** *probability at* k *approaches:* $\liminf_{t\to\infty:\, z_t^k=1} \pi^{\text{exec}}\Big(a_t^{n_t(k)} = \textbf{stay} \,\Big|\, \boldsymbol{o}_t,\ \boldsymbol{a}_t^{1:n_t(k)-1}\Big) \geq 1 - \frac{8\epsilon_{\text{IC}}}{9}$.

For proof, see Appendix C.2. Theorem 2 indicates that, for large isolated anomalies, the UAVs tend to keep focusing on them, i.e., ensuring the anomaly can be quickly detected. Finally, we relate post-change visitation frequency $q_k(\tau)$ to detection delay via an information-rate bound.

Theorem 3 (Detection-Delay Bound)**.** *Let* f_k *and* g_k *denote the pre-change and post-change densities of the measurements at location* k*, and define the Kullback-Leibler information* $I(g_k, f_k) := \mathbb{E}_{g_k}\left[\log \frac{g_k(X)}{f_k(X)}\right]$*. Let* $D(\tau) := \{k \in \mathcal{B} : q_k(\tau) > 0\}$*. Then, as* $H \to \infty$*, the expected detection delay satisfies* $\mathbb{E}[T(H) - \tau] \leq \min_{k\in D(\tau)} \frac{H}{N\, q_k(\tau)\, I(g_k, f_k)} + O(1)$*, where* $O(1)$ *does not grow with* H*.*

For proof, see Appendix C.3. Recall that for $t \geq \tau$ and $\mathcal{B} \neq \emptyset$, $R_t^A = \frac{1}{N}\sum_{n=1}^N r_t^n = \frac{1}{N}\left(2\sum_{k\in\mathcal{B}} z_t^k - N\right) = 2 \cdot \frac{1}{N}\sum_{k\in\mathcal{B}} z_t^k - 1$. Thus, maximizing $\mathcal{J}(\pi)$ encourages allocating more resources to the abnormal set, as quantified by $\sum_{k\in\mathcal{B}} q_k(\tau)$. This, in turn, reduces the upper bound on detection delay and leads to faster anomaly detection. We empirically validate this result in Section 6.

6 Experiments

We evaluate DISPATCHPPO on simulated environments and two real-world cases: a real-robot testbed and a wildfire monitoring case study (Appendix D.4). We compare against heuristic and optimization-based dispatch baselines, with implementation details and additional results in Appendix D.

Setup. We consider 5×5, 10×10, and 30×30 grids with varying fleet sizes N. An anomaly occurs at $\tau \sim \text{Unif}[5, 50]$, modeled as a persistent mean shift of magnitude Δ on a randomly affected set $\mathcal{B}$. We set $\mu_{\min} = 1$, the smallest shift magnitude of interest [2, 19]. DISPATCHPPO is trained offline and selected via periodic validation. At evaluation, we calibrate the threshold H by Monte Carlo under in-control scenarios to achieve ARL_0=370 [23], and then estimate the detection delay ARL_1 under each Δ. Code is available at `https://github.com/xxia99/DispatchPPO`.

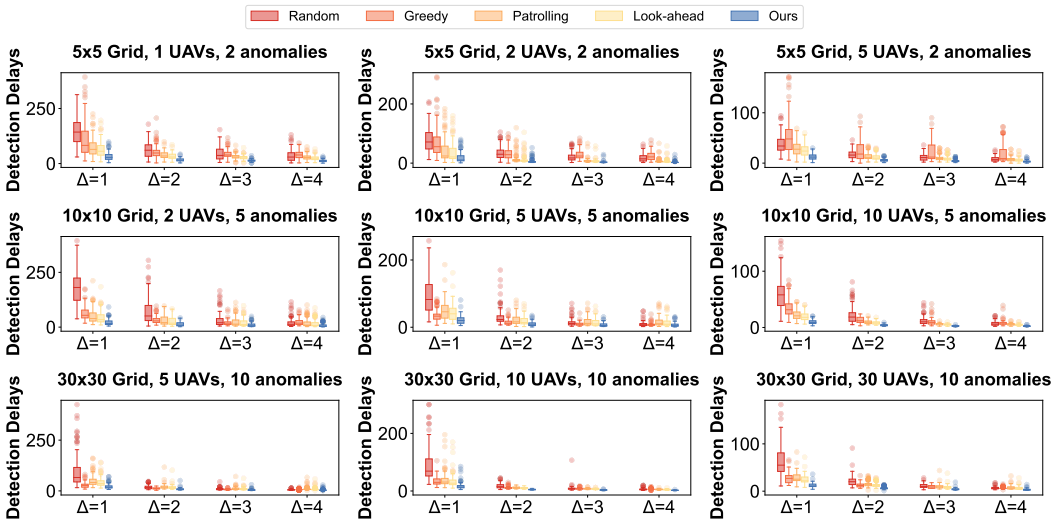


Figure 4: Detection delays ARL_1 under calibrated false-alarm control across grid sizes and fleet sizes.

Baselines. We compare against: (i) *Random*, where each UAV samples a uniformly random feasible local move; (ii) *Greedy*, where each UAV moves to the neighboring cell with the largest current statistic w_t^k; (iii) *Patrolling*, where each UAV follows a fixed boustrophedon route and stays when its current local statistic exceeds a preset threshold; and (iv) *Look-ahead* [2], which solves a 5-step collision-free route-planning problem at each step to maximize total collected anomaly evidence.

Analysis of Stage I training. Figure 3 summarizes offline learning dynamics. Across all settings, policy updates consistently improve the return, as reflected by higher average per-step reward and hit rate. Larger fleets achieve higher hit rates, while larger grids generally require more episodes to converge. At fixed intervals, we pause training and evaluate the current policy in a validation environment. Because the detection threshold H is not yet calibrated at this stage, we evaluate checkpoints using a conservative pseudo-threshold $\hat{H}$ and report the resulting proxy delay as $\log(\mathrm{ARL}_1)$. The validation detection delay generally decreases over training, which suggests that optimization under the proxy reward is effective and drives the policy toward quick anomaly detection. We select the checkpoint with the lowest validation delay for Stage II execution and subsequent ARL_0 calibration.

Performance across problem scales. Using the selected checkpoint, we calibrate the alarm threshold H for each policy to satisfy ARL_0=370 (with 1000 in-control replications; horizon 2000). We then evaluate ARL_1 for mean shifts $\Delta \in \{1, 2, 3, 4\}$ using 1000 out-of-control replications (horizon 2000). Figure 4 shows that DISPATCHPPO consistently achieves smaller detection delays than all baselines across grid sizes and fleet sizes. Two trends are consistent across methods. First, detection delay decreases as the change magnitude Δ increases, since stronger shifts accumulate evidence faster. Second, within each grid, detection delay decreases as the fleet size N increases, since more locations are sampled at each step. The advantage of DISPATCHPPO is most pronounced in the most challenging regimes, namely small N and weak anomalies (especially $\Delta = 1$), where sparse decision-dependent observations and collision constraints make effective coordination critical. As N or Δ increases, the task becomes easier and all methods attain lower delay; however, DISPATCHPPO remains consistently the best. These observations align with Theorem 3, which indicates lower detection delay under stronger post-change evidence and higher effective sensing allocation.

Sensitivity Analysis. Figure 5 examines two important parameters in Stage II: the uncertainty inflation C^U and the exploration-mixture rate ϵ_{IC}. Across all anomaly magnitudes Δ, the best performance is consistently achieved with a small exploration rate ($\epsilon_{\mathrm{IC}} \in [0.01, 0.05]$), while the optimal value of C^U increases with abnormal magnitude. This observation is consistent with the intuition that stronger anomalies are easier to recognize, thus allowing for greater exploratory behavior, which can, in turn, help detect anomalies more quickly. In contrast, large exploration rate ($\epsilon_{\mathrm{IC}} \geq 0.5$) consistently increases detection delay, and ϵ_{IC}=1 changes execution to random visiting. Performance degradation is most pronounced when a large ϵ_{IC} is paired with a large C^U (top-right region of

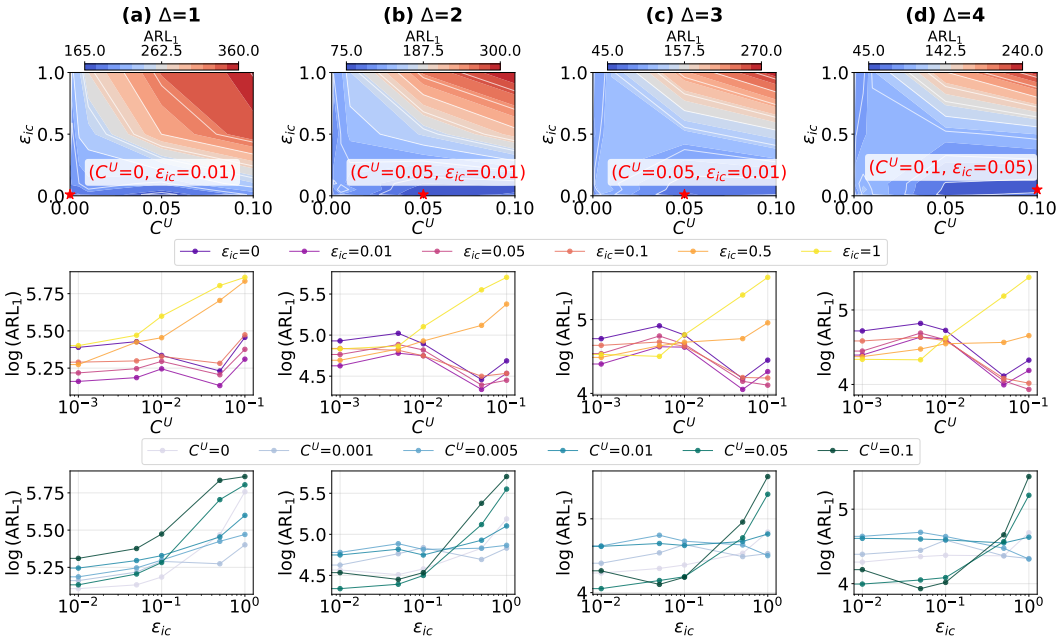


Figure 5: Sensitivity to C^U and ϵ_{IC} (10×10 grid, N=2). Columns vary the abnormal magnitude Δ. *Top:* ARL_1 heatmaps over $(C^U, \epsilon_{\text{IC}})$ (lower is better); the red $\star$ marks the best combination. *Middle:* $\log(\text{ARL}_1)$ vs. C^U for each fixed ϵ_{IC}. *Bottom:* $\log(\text{ARL}_1)$ vs. ϵ_{IC} for each fixed C^U.

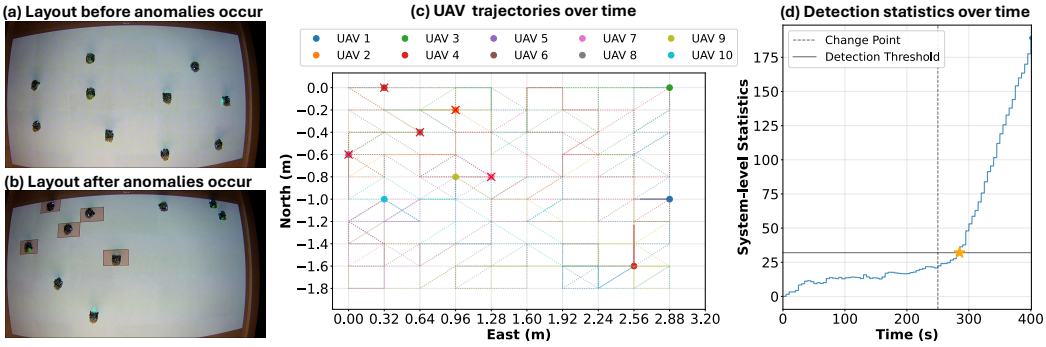


Figure 6: Robotarium [30] deployment of DISPATCHPPO. $N = 10$ robots monitor a region with $|\mathcal{B}| = 5$ anomalies that occur from $t = 250$s (full horizon: 400s). (a) Pre-change exploration. (b) Post-change focusing (anomalies in red). (c) Trajectories (solid: last 10s tail; earlier segments dotted; anomalies marked by red $\times$). (d) Detection statistic over time (orange $\star$ indicates alarm triggered).

the heatmaps). In such a case, excessive C^U causes unvisited locations to accumulate large w_t^k, while large ϵ_{IC} prevents sustained concentration on locally dominant evidence, thereby slowing the focusing behavior (Theorem 2). By contrast, small exploration rates ($\epsilon_{\text{IC}} \in [0.01, 0.1]$) preserve the coverage guarantee (Theorem 1) while enabling rapid focusing behavior, yielding consistently low detection delays across a wide range of C^U.

Case study: real-robot deployment. To assess sim-to-real transfer, we deploy the policy learned in simulation directly to Robotarium [30], without fine-tuning. We evaluate DISPATCHPPO with $N = 10$ robots monitoring a region with $|\mathcal{B}| = 5$ anomalies. As shown in Figure 6, the robots first disperse across the region and then concentrate around anomalous locations after the change. These trajectories are consistent with the persistent-coverage and anomaly-focusing behaviors in Theorems 1 and 2. The detection statistic remains stable before the change, then rises sharply and

crosses the threshold about 34s after anomalies occur. These results suggest that the learned policy transfers to real-robot control while retaining coverage, adaptive concentration, and timely detection.

7 Conclusion

We studied route-wise quickest detection with a coordinated UAV fleet under local mobility, decision-dependent partial observations, and hard collision avoidance. We formulated the problem as a POMDP whose observation is a compact belief surrogate built from location-wise MSPC statistics and UAV positions. To address long-horizon planning and combinatorial feasibility, we proposed a two-stage learning-and-execution pipeline. In Stage I, DISPATCHPPO learns long-horizon dispatch policies using DISPATCHNET, an autoregressive encoder-decoder with dynamic feasibility masking that outputs collision-free joint moves without enumerating the exponential joint-action space. In Stage II, we execute an exploration-mixture policy with a CUSUM-driven stay gate that maintains persistent coverage while adaptively concentrating sensing efforts as local anomaly evidence becomes dominant. Under stated regularity conditions, we established guarantees for persistent coverage, evidence-based focusing, and an information-rate bound that links post-change sensing allocation to detection delay. Extensive experiments demonstrate that the proposed approach consistently reduces detection delay compared to other baselines while remaining computationally efficient and transferable to real-robot deployment and a reconstructed wildfire monitoring case study.

Limitations and future work. Our formulation and guarantees rely on several assumptions that may hold approximately in practice. First, we discretize the environment, assume synchronous motion, and enforce hard collision avoidance. These assumptions are reasonable for robots operating on a gridded patrol region, but they abstract away continuous-time dynamics, kinematic constraints, and sensing-resource limits such as battery capacity. Extending the framework to continuous dynamics and richer operational constraints is a natural direction for future work. Second, the anomaly model assumes an exogenous persistent distribution shift, conditionally independent measurements, and a Gaussian in-control model for MSPC updates. These assumptions can be partially satisfied when normal-operation data are available to estimate a baseline, but they abstract away from correlated noise and data heterogeneity. An important extension is to handle weaker distributional assumptions and more general anomaly processes, for example, by replacing the parametric MSPC component with nonparametric or distributionally robust alternatives. Despite these idealizations, the empirical results remain consistent with the qualitative conclusions of the theory. The sensitivity analysis supports the predicted coverage-focusing tradeoff, and the real-robot and wildfire studies suggest that the method remains effective over a broad operating range beyond the idealized setting.

Broader impacts. Earlier anomaly detection with small UAV fleets can benefit wildfire response, search and rescue, and infrastructure monitoring. The same dispatch capability could also be misused for surveillance of people. The method detects statistical changes in sensor streams and does not identify individuals, and deployments should follow the regulations that govern UAV operation and data collection in the monitored area.

Acknowledgments and Disclosure of Funding

This work was supported in part by the U.S. Department of Energy under Award DE-NE0009404 and in part by the Office of Naval Research under Grant N00014-23-1-2495. The authors declare no competing interests.

References

[1] Yue Jiang and Ana María Estrada Gómez. An adaptive sampling strategy for real-time anomaly detection with unmanned sensing vehicles. *Technometrics*, 66(3):438–454, 2024. doi: 10.1080/00401706.2024.2322645.

[2] Dongmin Li, Miao Bai, and Xiaochen Xian. Data-driven pathwise sampling approaches for online anomaly detection. *Technometrics*, 66(4):600–613, 2024. doi: 10.1080/00401706.2024.2342314.

[3] E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1–2):100–115, 1954. doi: 10.1093/biomet/41.1-2.100.

[4] H. Vincent Poor and Olympia Hadjiliadis. *Quickest Detection*. Cambridge University Press, 2008. doi: 10.1017/CBO9780511754678.

[5] Geoffrey A. Hollinger and Gaurav S. Sukhatme. Sampling-based robotic information gathering algorithms. *The International Journal of Robotics Research*, 33(9):1271–1287, 2014. doi: 10.1177/0278364914533443.

[6] Julius Rückin, Liren Jin, and Marija Popović. Adaptive informative path planning using deep reinforcement learning for UAV-based active sensing. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4473–4479, 2022. doi: 10.1109/ICRA46639.2022.9812025.

[7] Marija Popović, Joshua Ott, Julius Rückin, and Mykel J. Kochenderfer. Learning-based methods for adaptive informative path planning. *Robotics and Autonomous Systems*, 179:104727, 2024. doi: 10.1016/j.robot.2024.104727.

[8] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28, pages 2692–2700, 2015.

[9] MohammadReza Nazari, Afshin Oroojlooy, Lawrence V. Snyder, and Martin Takáč. Reinforcement learning for solving the vehicle routing problem. In *Advances in Neural Information Processing Systems*, volume 31, pages 9861–9871, 2018.

[10] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019.

[11] Jieyi Bi, Yining Ma, Jianan Zhou, Wen Song, Zhiguang Cao, Yaoxin Wu, and Jie Zhang. Learning to handle complex constraints for vehicle routing problems. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL `https://arxiv.org/abs/2410.21066`.

[12] Yi Ma, Xiaotian Hao, Jianye Hao, Jiawen Lu, Xing Liu, Xialiang Tong, Mingxuan Yuan, Zhigang Li, Jie Tang, and Zhaopeng Meng. A hierarchical reinforcement learning based optimization framework for large-scale dynamic pickup and delivery problems. In *Advances in Neural Information Processing Systems*, volume 34, pages 23609–23620, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/c6a01432c8138d46ba39957a8250e027-Abstract.html`.

[13] Kai Jungel, Axel Parmentier, Maximilian Schiffer, and Thibaut Vidal. Learning-based online optimization for autonomous mobility-on-demand fleet control. *INFORMS Journal on Computing*, 38(3):745–765, 2025. doi: 10.1287/ijoc.2024.0637.

[14] Matthew S. Maxwell, Mateo Restrepo, Shane G. Henderson, and Huseyin Topaloglu. Approximate dynamic programming for ambulance redeployment. *INFORMS Journal on Computing*, 22(2):266–281, 2010. doi: 10.1287/ijoc.1090.0345.

[15] Roni Stern, Nathan R. Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne T. Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, T. K. Satish Kumar, Roman Barták, and Eli Boyarski. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search*, volume 10, pages 151–158, 2019.

[16] Wenhao Luo, Wen Sun, and Ashish Kapoor. Multi-robot collision avoidance under uncertainty with probabilistic safety barrier certificates. In *Advances in Neural Information Processing Systems*, 2020.

[17] Akifumi Wachi, Xun Shen, and Yanan Sui. A survey of constraint formulations in safe reinforcement learning. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 8262–8271. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/913. URL `https://doi.org/10.24963/ijcai.2024/913`. Survey Track.

[18] Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022. doi: 10.48550/arXiv.2205.10330.

[19] Kaibo Liu, Yajun Mei, and Jianjun Shi. An adaptive sampling strategy for online high-dimensional process monitoring. *Technometrics*, 57(3):305–319, 2015. doi: 10.1080/00401706.2014.947005.

[20] Xiaochen Xian, Andi Wang, and Kaibo Liu. A nonparametric adaptive sampling strategy for online monitoring of big data streams. *Technometrics*, 60(1):14–25, 2018. doi: 10.1080/00401706.2017.1317291.

[21] Xiaochen Xian, Chen Zhang, Scott Bonk, and Kaibo Liu. Online monitoring of big data streams: A rank-based sampling algorithm by data augmentation. *Journal of Quality Technology*, 53(2):135–153, 2021. doi: 10.1080/00224065.2019.1681924.

[22] Jie Guo, Hao Yan, and Chen Zhang. A bayesian partially observable online change detection approach with thompson sampling. *Technometrics*, 65(2):179–191, 2023. doi: 10.1080/00401706.2022.2127914.

[23] Honghan Ye, Xiaochen Xian, Jing-Ru C. Cheng, Brock Hable, Robert W. Shannon, Mojtaba Kadkhodaie Elyaderani, and Kaibo Liu. Online nonparametric monitoring of heterogeneous data streams with partial observations based on thompson sampling. *IISE Transactions*, 55(4):392–404, 2023. doi: 10.1080/24725854.2022.2039423.

[24] Haoqian Li, Honghan Ye, Jing-Ru C. Cheng, and Kaibo Liu. Online monitoring of heterogeneous partially observable data streams based on q-learning. *IEEE Transactions on Automation Science and Engineering*, 22:4802–4817, 2025. doi: 10.1109/TASE.2024.3411770.

[25] Vaibhav Srivastava, Fabio Pasqualetti, and Francesco Bullo. Stochastic surveillance strategies for spatial quickest detection. *The International Journal of Robotics Research*, 32(12):1438–1458, 2013. doi: 10.1177/0278364913490322.

[26] Pushkarini Agharkar and Francesco Bullo. Quickest detection over robotic roadmaps. *IEEE Transactions on Robotics*, 32(1):252–259, 2016. doi: 10.1109/TRO.2015.2506165.

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[29] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[30] Sean Wilson, Paul Glotfelter, Li Wang, Siddharth Mayya, Gennaro Notomista, Mark Mote, and Magnus Egerstedt. The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems. *IEEE Control Systems Magazine*, 40(1):26–44, 2020. doi: 10.1109/MCS.2019.2949973.

[31] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[32] Kazuoki Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19(3):357–367, 1967. doi: 10.2748/tmj/1178243286.

[33] Rick Durrett. *Probability: Theory and Examples*. Cambridge University Press, 5 edition, 2019. doi: 10.1017/9781108591034.

[34] David Siegmund. *Sequential Analysis: Tests and Confidence Intervals*. Springer Series in Statistics. Springer, 1 edition, 1985. doi: 10.1007/978-1-4757-1862-1.

A Supplemental Details of DISPATCHPPO

A.1 Algorithm of DISPATCHNET

In this subsection, we provide the detailed algorithms for implementing DISPATCHNET. Algorithm 1 describes the forward pass of DISPATCHNET, which generates a collision-free joint action through autoregressive decoding with dynamic feasibility masking. The key idea is to assign actions to UAVs sequentially while maintaining a reservation set that prevents collisions. Algorithm 2 presents an equivalent implementation of the collision mask using two boolean occupancy tables for destinations and future positions, enabling $O(1)$ feasibility checks for each candidate action.

Algorithm 1 DISPATCHNET (Stage I) forward pass: collision-free autoregressive decoding

Require: Observation $\boldsymbol{o}_t$; UAV positions $\boldsymbol{p}_t = (p_t^1, \dots, p_t^N)$; base feasibility sets $\{\mathcal{A}_{\text{base}}^n(\boldsymbol{o}_t)\}_{n=1}^N$; CNN encoder f_{cnn}; interaction encoder f_{tr}; UAV-id embeddings $\boldsymbol{e}_{\text{id}}(\cdot)$; position embedding matrix W_{pos}; action embedding $\text{Emb}_{\text{act}}(\cdot)$; decoder head g_{dec}; destination map $\phi(\cdot,\cdot)$.

Ensure: Joint action $\boldsymbol{a}_t = (a_t^1, \dots, a_t^N)$.

$\boldsymbol{H}_t \leftarrow f_{\text{cnn}}(\boldsymbol{o}_t) \in \mathbb{R}^{I \times J \times d}$.
for $n = 1$ **to** N **do**
  $\boldsymbol{e}_t^n \leftarrow \boldsymbol{H}_t[p_t^n] + \boldsymbol{e}_{\text{id}}(n) + W_{\text{pos}}[i_t^n, j_t^n]^\top \in \mathbb{R}^d$.
end for
$(\boldsymbol{h}_t^1, \dots, \boldsymbol{h}_t^N) \leftarrow f_{\text{tr}}(\boldsymbol{e}_t^1, \dots, \boldsymbol{e}_t^N)$.
$\mathcal{D} \leftarrow \emptyset$ ▷ assigned destinations $\{\hat{p}_{t+1}^i : i \le n-1\}$
$\mathcal{F} \leftarrow \{p_t^2, \dots, p_t^N\}$ ▷ positions of not-yet-decoded UAVs
$a_t^0 \leftarrow$ start.
for $n = 1$ **to** N **do**
  $\mathcal{F} \leftarrow \mathcal{F} \setminus \{p_t^n\}$ ▷ now $\mathcal{F} = \{p_t^{n+1}, \dots, p_t^N\}$
  $\boldsymbol{v}_t^n \leftarrow [\boldsymbol{h}_t^n;\ \text{Emb}_{\text{act}}(a_t^{n-1})]$.
  $\boldsymbol{\ell}_t^n \leftarrow g_{\text{dec}}(\boldsymbol{v}_t^n) \in \mathbb{R}^{|\mathcal{A}_{\text{loc}}|}$.
  $\mathcal{R}_t^n \leftarrow \mathcal{D} \cup \mathcal{F}$.
  $\mathcal{A}_t^n(\boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1}) \leftarrow \{a \in \mathcal{A}_{\text{base}}^n(\boldsymbol{o}_t) : \phi(p_t^n, a) \notin \mathcal{R}_t^n\}$.
  Sample $a_t^n \sim \tilde{\pi}_\theta(\cdot \mid \boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1})$ where $\tilde{\pi}_\theta(a) \propto \exp(\ell_t^n(a))\ \mathbb{1}\{a \in \mathcal{A}_t^n(\boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1})\}$.
  $\hat{p}_{t+1}^n \leftarrow \phi(p_t^n, a_t^n)$; $\mathcal{D} \leftarrow \mathcal{D} \cup \{\hat{p}_{t+1}^n\}$.
end for
return $\boldsymbol{a}_t = (a_t^1, \dots, a_t^N)$.

A.2 DISPATCHPPO: scalable trust-region learning.

We optimize the policy parameters using a PPO-style trust-region method. At iteration $g \in [G]$, the policy π_{θ_g} and value function $\mathcal{V}_{\psi_g}$ are parameterized by θ_g and ψ_g, respectively. Following [29], the clipped surrogate objective for updating θ_g is

$$\mathcal{L}_{\theta_g} = \mathbb{E}_t\Big[\min\Big\{\Omega_{g,t}\hat{A}_t,\ \text{clip}(\Omega_{g,t}, 1-\epsilon, 1+\epsilon)\hat{A}_t\Big\}\Big],$$

where $\Omega_{g,t} = \frac{\pi_{\theta_g}(\alpha_t|\boldsymbol{o}_t)}{\pi_{\theta_{g-1}}(\alpha_t|\boldsymbol{o}_t)}$ and $\epsilon > 0$ is the clipping parameter. The advantage estimate $\hat{A}_t$ is computed via generalized advantage estimation (GAE) [31] using rollout returns under $\pi_{\theta_{g-1}}$: $\hat{A}_t = (\gamma)^{T-t+1}\mathcal{V}_{\psi_g}(\boldsymbol{o}_{T+1}) - \mathcal{V}_{\psi_g}(\boldsymbol{o}_t) + \sum_{d=0}^{T-t}(\gamma)^d r_{t+d}$. The value network is trained by the squared-error loss [29]:

$$\mathcal{L}_{\psi_g} = \mathbb{E}_t\left[\left(\mathcal{V}_{\psi_g}(\boldsymbol{o}_t) - \sum_{d=0}^{T-t-1}\gamma^d r_{t+d} - \gamma^{T-t}\mathcal{V}_{\psi_g}(\boldsymbol{o}_T)\right)^2\right].$$

Together, these updates yield a stable and scalable learning procedure for coordinated UAV dispatch under hard mobility and collision constraints.

Algorithm 2 Implementing the collision mask in DISPATCHNET via two occupancy tables

Require: Collision-free positions $\boldsymbol{p}_t = (p_t^1, \ldots, p_t^N)$; local action set $\mathcal{A}_{\text{loc}}$; base feasibility sets $\{\mathcal{A}^n_{\text{base}}(\boldsymbol{o}_t)\}_{n=1}^N$; destination map ϕ.
Ensure: Masks $\{m_t^n(a)\}$ for $\mathcal{A}_t^n$ (and optionally sampled actions).
$\text{occ}_{\text{dest}}[1{:}K] \leftarrow$ false; $\text{occ}_{\text{future}}[1{:}K] \leftarrow$ false.
for $j = 2$ **to** N **do**
  $\text{occ}_{\text{future}}[p_t^j] \leftarrow$ true
end for
for $n = 1$ **to** N **do**
  $\text{occ}_{\text{future}}[p_t^n] \leftarrow$ false ▷ remove current p_t^n from future set
  for $a \in \mathcal{A}_{\text{loc}}$ **do**
    $c \leftarrow \phi(p_t^n, a)$.
    $m_t^n(a) \leftarrow \mathbb{1}\{a \in \mathcal{A}^n_{\text{base}}(\boldsymbol{o}_t)\} \cdot \mathbb{1}\{\neg\text{occ}_{\text{dest}}[c]\} \cdot \mathbb{1}\{\neg\text{occ}_{\text{future}}[c]\}$.
  end for
  (If sampling) draw a_t^n from the masked distribution and set $c^\star \leftarrow \phi(p_t^n, a_t^n)$.
  (If sampling) $\text{occ}_{\text{dest}}[c^\star] \leftarrow$ true.
end for

B Theoretical Investigation of DispatchNet Properties

In this appendix, we prove Proposition 1, which states that the autoregressive decoder in DISPATCHNET always produces a feasible, collision-free joint action with scalable per-step complexity.

Proposition 1. *Suppose $\boldsymbol{p}_t$ is collision-free. Then $\mathcal{A}_t^n(\boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1})$ is nonempty for every n, and any $\boldsymbol{a}_t$ produced by Algorithm 1 induces a collision-free $\boldsymbol{p}_{t+1}$. Moreover, for a constant $|\mathcal{A}_{\text{loc}}|$, the autoregressive decoder with dynamic feasibility masking runs in $O(K+N)$ time per step.*

Proof of Proposition 1. **Nonemptiness of $\mathcal{A}_t^n$.** We show that at each decoder step n, the feasible action set of UAV n, $\mathcal{A}^n(\boldsymbol{o}_t,\, a_t^{1:n-1})$, is non-empty. For $n = 1$, UAV 1 can always choose the stay action at its current location, so $\mathcal{A}^1$ contains at least that action. For $1 < n \le N$, recall that before assigning UAV n's action, the reserved set is:

$$\mathcal{R}_t^n = \{\hat{p}_{t+1}^i :\ 1 \le i < n\} \ \cup\ \{p_t^j :\ n < j \le N\}.$$

We claim that UAV n's current cell p_t^n is never reserved, i.e., $p_t^n \notin \mathcal{R}_t^n$. Specifically, (i) since the configuration $\boldsymbol{p}_t$ is collision-free, $p_t^n \neq p_t^j$ for all $j > n$, so $p_t^n \notin \{p_t^j :\ n < j \le N\}$. (ii) For the already-assigned destinations, fix any $i < n$. When UAV i was decoded, the mask reserved the current positions of all not-yet-decoded UAVs, which included p_t^n. Hence, UAV i was forbidden to choose a move with destination p_t^n, implying $\hat{p}_{t+1}^i \neq p_t^n$. Therefore $p_t^n \notin \{\hat{p}_{t+1}^i :\ 1 \le i < n\}$, and thus $p_t^n \notin \mathcal{R}_t^n$. Moreover, since $\textsf{stay} \in \mathcal{A}^n_{\text{base}}(\boldsymbol{o}_t)$ and $\phi(p_t^n, \textsf{stay}) = p_t^n \notin \mathcal{R}_t^n$, the action stay is always feasible for UAV n. Consequently, $\mathcal{A}_t^n(\boldsymbol{o}_t, \boldsymbol{a}_t^{1:n-1})$ is nonempty for every n.

Collision-free next configuration. We now show that the joint action $\boldsymbol{a}_t = (a_t^1, \ldots, a_t^N)$ is always collision-free, i.e., no two UAVs attempt to move to the same cell. Let $\hat{p}_{t+1}^n := \phi(p_t^n, a_t^n)$ be the destination chosen for UAV n for time $t+1$. Suppose, for contradiction, that UAV i and UAV j ($i < j$) both end up with $\phi(p_t^i, a_t^i) = \phi(p_t^j, a_t^j) = q$, meaning they plan to occupy the same cell q at time $t+1$. There are two cases:

Case 1. If $q = \phi(p_t^i, a_t^i)$ has been chosen by the earlier UAV i, then q would belong to the reserved set $\mathcal{R}_t^j$ when UAV j chooses its action (since $j > i$). By construction, the decoder forbids UAV j selecting any action leading to locations in $\mathcal{R}_t^j$. Thus, j could not have chosen an action ending at q.

Case 2. If instead $q = \phi(p_t^j, a_t^j)$ is currently occupied by the later UAV j, then we consider the moment when UAV i chose its action. Because $j > i$, at that time p_t^j (the current position of UAV j) was in $\mathcal{R}_t^i$, so UAV i was prevented from moving into p_t^j.

In both cases, we reach a contradiction. Therefore, no two UAVs collide; the decoder's masking mechanism guarantees that all chosen destinations $\hat{p}_{t+1}^1, \ldots, \hat{p}_{t+1}^N$ are distinct. This proves that DISPATCHNET always produces a feasible, collision-free joint action.

Complexity. We implement collision checking in the masking step with two boolean occupancy tables over grid cells, as in Algorithm 2: $\mathrm{occ}_{\mathrm{dest}}[1{:}K]$ marks destinations already assigned to earlier UAVs, and $\mathrm{occ}_{\mathrm{future}}[1{:}K]$ marks the current positions of UAVs not yet decoded. These tables require $O(K)$ memory. At decoder step n, we test at most $|\mathcal{A}_{\mathrm{loc}}|$ candidate actions. Each feasibility test performs $O(1)$ table lookups, so masking and sampling cost $O(|\mathcal{A}_{\mathrm{loc}}|)$ per UAV and $O(N|\mathcal{A}_{\mathrm{loc}}|)$ per time step. For constant $|\mathcal{A}_{\mathrm{loc}}|$, this is $O(N)$ time per step. In our implementation, the occupancy tables are cleared from scratch at every time step. This adds an $O(K)$ reset cost. Therefore, the auto-regressive decoding yields an overall complexity of $O(K+N)$.

□

C Theoretical Investigation of SPC-Related Results

In this appendix, we prove the theoretical properties of the executed dispatch policy claimed by Theorems 1, 2, and 3, along with supporting lemmas.

Recall $\mathcal{H}_t$ denotes the full dispatcher history up to time t. At decoder step n and time t, $\mathcal{A}_t^n \subseteq \mathcal{A}_{\mathrm{loc}}$ denotes the masked feasible action set. We will use the following two standard tools.

Azuma-Hoeffding inequality [32]. If $\{X_m\}_{m\geq 1}$ is a martingale difference sequence bounded by $|X_m| \leq 1$ and $\mathbb{E}[X_m \mid \mathcal{G}_{m-1}] = 0$ for each m, then for any $\epsilon > 0$ and any number of terms M,

$$\Pr\Big\{\Big|\sum_{m=1}^{M} X_m\Big| \geq \epsilon M\Big\} \;\leq\; 2\exp\Big(-\frac{\epsilon^2 M}{2}\Big).$$

In particular, with probability 1, the average $\frac{1}{M}\sum_{m=1}^{M} X_m$ converges to 0 as $M \to \infty$.

Borel-Cantelli lemma [33]. If $\sum_{m=1}^{\infty} \Pr(E_m) < \infty$ for a sequence of events $E_1, E_2, \ldots$, then with probability 1 only finitely many of the E_m occur. Equivalently, $\Pr(E_m \text{ infinitely often}) = 0$. In our context, we use this lemma to show that if each time block has a fixed positive probability of containing a visit to a location, then as time goes on, such visits occur infinitely often almost surely.

C.1 Proof and Lemmas of Theorem 1

Lemma 1 (Uniform Exploration Guarantees Coverage)**.** *At any time t and decoder step n, let $\mathcal{A}_t^n$ be the feasible action set. Under the executed policy π^{exec} in* (2)*, for any feasible action $a \in \mathcal{A}_t^n$,*

$$\pi^{\mathrm{exec}}(a_t^n = a \mid \boldsymbol{o}_t, \boldsymbol{p}_t, a_t^{1:n-1}) \;\geq\; \frac{\epsilon_{\mathrm{IC}}}{|\mathcal{A}_t^n|} \;\geq\; \frac{\epsilon_{\mathrm{IC}}}{9}.$$

In particular, if stay $\in \mathcal{A}_t^n$ *and* $\pi_\theta^{\mathrm{lock}}(a_t^n = \textsf{stay} \mid \cdot) = \xi_t^n(p_t^n)$*, then*

$$\pi^{\mathrm{exec}}(a_t^n = \textsf{stay} \mid \cdot) \;\geq\; (1-\epsilon_{\mathrm{IC}})\,\xi_t^n(p_t^n) + \frac{\epsilon_{\mathrm{IC}}}{|\mathcal{A}_t^n|}.$$

Proof. By definition,

$$\pi^{\mathrm{exec}}(a) = (1-\epsilon_{\mathrm{IC}})\,\pi_\theta^{\mathrm{lock}}(a) + \epsilon_{\mathrm{IC}}\frac{1}{|\mathcal{A}_t^n|} \;\geq\; \epsilon_{\mathrm{IC}}\frac{1}{|\mathcal{A}_t^n|}.$$

Since $|\mathcal{A}_t^n| \leq |\mathcal{A}_{\mathrm{loc}}| = 9$, the first claim follows. The second is the special case $a = \textsf{stay}$. □

Lemma 2 (Reachability in a Connected Graph)**.** *Let $G = ([K], \mathcal{E})$ be the mobility graph induced by the grid and one-step moves, including* stay *as a self-loop. Assume G is connected, and let D be its diameter. Then for any collision-free UAV configuration $\boldsymbol{p}_t = (p_t^1, \ldots, p_t^N)$ and any target location $k \in [K]$, there exists a collision-free sequence of at most D joint moves that brings some UAV to k.*

Proof. If one of the UAVs is already at k, the claim is immediate. Otherwise, because G is connected, we can take any UAV and move it step by step along a shortest path in G from its current cell to k. More formally, let $u \in \{p_t^1, \ldots, p_t^N\}$ be an occupied cell closest to k in graph distance, and let $(v_0, v_1, \ldots, v_d)$ be a shortest path from $k = v_0$ to $u = v_d$ in G (with $d \leq D$). By choice of

u, none of $v_0, \ldots, v_{d-1}$ is currently occupied (if a closer UAV occupied one of those, that would contradict u being the closest). Now, instruct the UAV at u to move along this path in reverse: $v_d \to v_{d-1} \to \cdots \to v_0$ while all other UAVs execute stay moves. Because the intermediate target $v_{d-1}, v_{d-2}, \ldots, v_0$ are initially unoccupied and no other UAV moves into them, the chosen UAV can safely traverse to k in $d \leq D$ steps without colliding with others. This proves that a collision-free route to any k exists with length at most D. □

Using these lemmas, we now prove Theorem 1.

Proof of Theorem 1. Consider the executed policy π^{exec} under the stated assumptions, and fix any location $k \in [K]$. We show that (a) k is visited infinitely often almost surely, and (b) its long-run visitation frequency is strictly positive. These two conclusions imply the theorem's claims (i) and (ii). For clarity, we break the argument into three steps.

Step 1: k is visited every D time steps with a fixed probability. Let D be the graph diameter from Lemma 2, and partition time into blocks of length D, with block m starting at $t_m = mD$. By Lemma 2, from any configuration at time t_m, there exists a feasible sequence of at most D joint moves that brings some UAV to k within block m. By Lemma 1, each individual action in this sequence is executed with probability at least $\epsilon_{\text{IC}}/9$. Since the block contains ND individual moves,

$$\Pr\{k \text{ is visited during block } m \mid \mathcal{H}_{t_m}\} \;\geq\; \Big(\frac{\epsilon_{\text{IC}}}{9}\Big)^{ND} = \delta > 0,$$

for every block m. Here δ depends only on ϵ_{IC}, N, and D, and not on m or the past.

Step 2: k is visited infinitely often. Define event $A_m := \{\text{block } m \text{ contains at least one visit to } k\}$, and its complement $A_m^c := \{\text{location } k \text{ is not visited at all during block } m\}$.

In step 1, we have just shown $\Pr(A_m \mid \mathcal{H}_{t_m}) \geq \delta$ for all m. In particular, $\Pr(A_m) \geq \delta$ unconditionally. Intuitively, if each block has a nonzero chance of success, eventually we should get infinitely many successes.

To formalize this, consider the probability that k is no longer visited after some block M. For any $L \geq 1$, using the law of total probability (conditioning sequentially on histories), we get

$$\begin{aligned}&\Pr(A_M^c \cap A_{M+1}^c \cap \cdots \cap A_{M+L-1}^c)\\ =\,&\Pr(A_M^c)\Pr(A_{M+1}^c \mid A_M^c)\Pr(A_{M+2}^c \mid A_M^c, A_{M+1}^c)\cdots \Pr(A_{M+L-1}^c \mid A_M^c, \ldots, A_{M+L-2}^c).\end{aligned}$$

But $\Pr(A_r^c \mid A_M^c, \ldots, A_{r-1}^c) \leq 1 - \delta$ for each $r \geq M$, since given no visit occurred in the previous blocks, the probability of no visit in the next block is at most $(1-\delta)$ (actually it is exactly $(1-\delta)$ if the conditional probability of a visit stays δ). Thus

$$\Pr\{\text{no visit to } k \text{ in blocks } M \text{ through } M+L-1\} \;\leq\; (1-\delta)^L.$$

As $L \to \infty$, $(1-\delta)^L \to 0$. Therefore

$$\Pr\{A_m \text{ fails for all } m \geq M\} \;=\; 0 \quad \text{for each fixed } M.$$

Therefore, with probability 1, infinitely many blocks contain a visit to k, so k is visited infinitely often almost surely.

Step 3: The long-run hit rate at k is positive. We now know that $Z_m := \mathbb{I}\{A_m\}$ (the indicator of a visit in block m) is 1 infinitely often. Moreover, we have $\mathbb{E}[Z_m \mid \mathcal{H}_{t_m}] \geq \delta$ for all m by Step 1.

Define $X_m := Z_m - \mathbb{E}[Z_m \mid \mathcal{H}_{t_m}]$. Then $\{X_m\}$ is a martingale difference sequence with respect to the filtration $\mathcal{H}_{t_m}$, and we have $|X_m| \leq 1$. Also note $\mathbb{E}[X_m \mid \mathcal{H}_{t_m}] = 0$ by construction.

By the Azuma-Hoeffding inequality, for any $\varepsilon > 0$,

$$\Pr\Big\{\sum_{m=0}^{M-1} X_m \leq -\varepsilon M\Big\} \;\leq\; \exp\Big(-\frac{\varepsilon^2 M}{2}\Big).$$

Summing the right-hand side over M, we can have: $\sum_{M=1}^{\infty} \exp\left(-\frac{\varepsilon^2 M}{2}\right) = \frac{\exp(-\varepsilon^2/2)}{1-\exp(-\varepsilon^2/2)} < \infty$. Thus, we have $\sum_{M=1}^{\infty} \Pr\Big\{\sum_{m=0}^{M-1} X_m \;\leq\; -\varepsilon M\Big\} \;<\; \infty$. By the Borel-Cantelli lemma, event

$\sum_{m=0}^{M-1} X_m \le -\varepsilon M$ occurs only finitely times. That is, eventually $\sum_{m=0}^{M-1} X_m > -\varepsilon M$ for all large M. Since $\varepsilon > 0$ was arbitrary, we get $\liminf_{M\to\infty} \frac{1}{M} \sum_{m=0}^{M-1} X_m \ge 0$ almost surely.

Given that $Z_m = \mathbb{E}[Z_m \mid \mathcal{H}_{t_m}] + X_m$, we can have:

$$\frac{1}{M} \sum_{m=0}^{M-1} Z_m \;=\; \frac{1}{M} \sum_{m=0}^{M-1} \mathbb{E}[Z_m \mid \mathcal{H}_{t_m}] \;+\; \frac{1}{M} \sum_{m=0}^{M-1} X_m \;\ge\; \delta + \frac{1}{M} \sum_{m=0}^{M-1} X_m \,,$$

since $\mathbb{E}[Z_m \mid \mathcal{H}_{t_m}] \ge \delta$ (by step 1) and $\liminf_{M\to\infty} \frac{1}{M} \sum_{m=0}^{M-1} X_m \ge 0$, taking $\liminf$ as $M \to \infty$, we obtain

$$\liminf_{M\to\infty} \frac{1}{M} \sum_{m=0}^{M-1} Z_m \;\ge\; \delta \qquad \text{almost surely.}$$

Remember Z_m indicates a visit in block m. Each block has length D, so a total of M blocks is MD time steps. If we spread the Z's back out over actual time steps, we find

$$\liminf_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} z_t^k \;\ge\; \frac{1}{D} \liminf_{M\to\infty} \frac{1}{M} \sum_{m=0}^{M-1} Z_m \;\ge\; \frac{\delta}{D} \;>\; 0 \qquad \text{with probability 1.}$$

Thus, the fraction of time steps that UAVs spend at k has a positive lower bound almost surely. Equivalently, the long-run visitation rate $q_k := \liminf_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} z_t^k$ is a positive random variable (in fact, bounded below by δ/D almost surely). This completes the proof of Theorem 1. □

C.2 Proof and Lemmas of Theorem 2

Recall that for each location k we maintain two one-sided CUSUM statistics (for detecting upward or downward mean shift). For a minimal detectable shift $\mu_{\min}$, the one-step log-likelihood increments are

$$\ell^+(x) := \mu_{\min} x - \tfrac{1}{2}\mu_{\min}^2, \qquad \ell^-(x) := -\mu_{\min} x - \tfrac{1}{2}\mu_{\min}^2.$$

These are chosen so that $\mathbb{E}[\ell^+(X)] = \mathbb{E}[\ell^-(X)] = 0$ under the in-control distribution $X \sim \mathcal{N}(0,1)$. For an affected location $k \in \mathcal{B}$ with post-change distribution $x_t^k \sim \mathcal{N}(\mu_k, 1)$, the CUSUM updates have a positive drift. In fact, one of the two sides (upward or downward) will accumulate evidence faster. We define

$$\kappa_k := \max\{\mathbb{E}[\ell^+(X)],\ \mathbb{E}[\ell^-(X)]\} = \mu_{\min}|\mu_k| - \tfrac{1}{2}\mu_{\min}^2 \,.$$

Now we analyze the post-change behavior and prove the anomaly focusing theorem.

Lemma 3 (Repeated visiting affected location gains evidence advantage)**.** *Suppose $\tau < \infty$ and $k \in \mathcal{B}$ satisfies $\kappa_k > C^U$. Let $N_k(t;\tau) := \sum_{s=\tau+1}^{t} z_s^k$ be the number of post-change visits to k by time t. Then for any $\varepsilon \in (0, \kappa_k - C^U)$, with probability 1 there exists a finite random constant b_k such that*

$$w_t^k \ge w_\tau^k + C^U(t-\tau) + (\kappa_k - C^U - \varepsilon)\, N_k(t;\tau) - b_k, \qquad \forall t \ge \tau.$$

Proof. Let $b \in \{+,-\}$ attain κ_k, i.e., $\mathbb{E}[\ell^b(X)] = \kappa_k$. Let $t_1^k < t_2^k < \cdots$ be the post-τ times when k is visited, and define

$$\eta_m := \ell^b(x_{t_m^k}^k) - C^U.$$

Because sensing decisions are non-anticipative and $\{x_t^k\}$ are i.i.d. over time under the model, $\{\eta_m\}$ are i.i.d. with mean $\mathbb{E}[\eta_m] = \kappa_k - C^U > 0$.

For the chosen side b, when $z_t^k = 0$, the recursion adds C^U, and when $z_t^k = 1$, $w_t^{k,b} = \max\{0, w_{t-1}^{k,b} + \ell^b(x_t^k)\} \ge w_{t-1}^{k,b} + \ell^b(x_t^k)$. Summing over $t = \tau+1, \ldots, t$ yields

$$w_t^{k,b} \ge w_\tau^{k,b} + C^U(t-\tau) + \sum_{m=1}^{N_k(t;\tau)} \eta_m.$$

Since $w_t^k = \max\{w_t^{k,+}, w_t^{k,-}\} \ge w_t^{k,b}$, it suffices to lower-bound the visited sum.

By the strong law of large numbers (Theorem 2.4.1 of [33]),

$$\frac{1}{M}\sum_{m=1}^{M}\eta_m \;\to\; \kappa_k - C^U \qquad \text{almost surely as } M\to\infty.$$

For any $\varepsilon \in (0, \kappa_k - C^U)$, on an event of probability 1, there exists a (random) index $M_\varepsilon < \infty$ such that for all $M \geq M_\varepsilon$, $\sum_{m=1}^{M}\eta_m \;\geq\; (\kappa_k - C^U - \varepsilon)\, M$. Define a finite constant

$$b_{k,0} := \max_{0\leq r\leq M_\varepsilon}\Big((\kappa_k - C^U - \varepsilon)r - \sum_{m=1}^{r}\eta_m\Big) < \infty.$$

Then for all $M \geq 0$,

$$\sum_{m=1}^{M}\eta_m \;\geq\; (\kappa_k - C^U - \varepsilon)\, M - b_{k,0},$$

The inequality holds with slack $b_{k,0}$ for $M \leq M_\varepsilon$ by definition of the maximum, and for $M \geq M_\varepsilon$ because $b_{k,0} \geq 0$.

Substituting $M = N_k(t;\tau)$ and using $w_t^k \geq w_t^{k,b}$ yields

$$w_t^k \;\geq\; w_\tau^{k,b} + C^U(t-\tau) + (\kappa_k - C^U - \varepsilon)\, N_k(t;\tau) - b_{k,0}.$$

Finally, define $b_k := b_{k,0} + (w_\tau^k - w_\tau^{k,b})$, which is finite almost surely because $w_\tau^k \geq w_\tau^{k,b}$ and $w_\tau^k < \infty$. This gives the claimed bound with w_τ^k. □

Lemma 4 (Unaffected neighbors grow no faster than $C^U t$)**.** *Fix any unaffected location $j \notin \mathcal{B}$ and any $t_0 \geq 1$. Then under the two-sided CUSUM recursion with inflation C^U,*

$$w_t^j \leq w_{t_0}^j + C^U(t-t_0) + o(t) \qquad \textit{with probability } 1 \textit{ as } t\to\infty.$$

Proof. Fix $j \notin \mathcal{B}$ and one side $b \in \{+,-\}$. Let $\sigma_1^j < \sigma_2^j < \cdots$ be the visiting times of j after t_0, and set $\eta_i := \ell^b(x^j_{\sigma_i^j})$. Since j is unaffected, $x^j_{\sigma_i^j} \sim \mathcal{N}(0,1)$ and $\mathbb{E}[\eta_i] = -\frac{1}{2}\mu_{\min}^2 < 0$.

Consider the sample-indexed reflected random walk

$$S_m := \sum_{i=1}^{m}\eta_i, \qquad W_m := S_m - \min_{0\leq r\leq m} S_r \;=\; \max_{1\leq r\leq m}(S_m - S_{r-1}),$$

which matches the CUSUM update at sample times. By the strong law, $S_m/m \to \mathbb{E}[\eta_1] < 0$ almost surely, so for any $\epsilon > 0$, for all large m we have $S_m \leq (\mathbb{E}[\eta_1] + \epsilon)m$ and also $S_r \geq (\mathbb{E}[\eta_1] - \epsilon)r$ for all large r. This implies $W_m \leq 2\epsilon m + O(1)$ and hence $W_m = o(m)$ almost surely.

Between samples, the recursion adds C^U each time step. Therefore, for $t \geq t_0$,

$$w_t^{j,b} \leq w_{t_0}^{j,b} + C^U(t-t_0) + W_{N_j(t)}, \qquad N_j(t) := \sum_{s=t_0+1}^{t} z_s^j,$$

and since $N_j(t) \leq t - t_0$ and $W_m = o(m)$, we obtain $w_t^{j,b} \leq w_{t_0}^{j,b} + C^U(t-t_0) + o(t)$ almost surely. Taking the maximum over $b \in \{+,-\}$ yields the claim for w_t^j. □

Proof of Theorem 2. Fix k as in the theorem and work on the probability-1 event where $\rho_k(\tau) > 0$. Apply Lemma 3 with $t_0 = \tau$ to get, for any small $\varepsilon \in (0, \kappa_k - C^U)$,

$$w_t^k \geq w_\tau^k + C^U(t-\tau) + (\kappa_k - C^U - \varepsilon)N_k(t;\tau) - O(1),$$

where $O(1)$ denotes a finite random constant. For any neighbor j of k, isolation implies $j \notin \mathcal{B}$, so Lemma 4 gives

$$w_t^j \leq w_\tau^j + C^U(t-\tau) + o(t) \qquad \text{almost surely.}$$

Taking the maximum over the finite neighbor set preserves the same form:

$$\max_{j\in\mathcal{N}(k)\setminus\{k\}} w_t^j \le C^U(t-\tau)+o(t)+O(1) \qquad \text{almost surely.}$$

Therefore,

$$w_t^k - \max_{j\in\mathcal{N}(k)\setminus\{k\}} w_t^j \ge (\kappa_k - C^U - \varepsilon)N_k(t;\tau) - o(t) - O(1).$$

Since $\rho_k(\tau) > 0$ implies $N_k(t;\tau)$ grows linearly in $t-\tau$, the right-hand side diverges to $+\infty$ almost surely.

Whenever $z_t^k = 1$, let $n_t(k)$ be the unique UAV index with $p_t^{n_t(k)} = k$. Because $\mathcal{N}_t^{n_t(k)}(k) \subseteq \mathcal{N}(k)$,

$$\Delta_t^{\mathrm{loc}}(n_t(k);k) = w_t^k - \max_{k'\in\mathcal{N}_t^{n_t(k)}(k)\setminus\{k\}} w_t^{k'} \ge w_t^k - \max_{j\in\mathcal{N}(k)\setminus\{k\}} w_t^j,$$

so $\Delta_t^{\mathrm{loc}}(n_t(k);k) \to +\infty$ along $\{t : z_t^k = 1\}$.

Since $\xi_t^{n_t(k)}(k) = \sigma(\Delta_t^{\mathrm{loc}}(n_t(k);k))$ and $\sigma(x) \to 1$ as $x \to +\infty$, we have $\xi_t^{n_t(k)}(k) \to 1$ along $\{t : z_t^k = 1\}$. Finally, Lemma 1 yields, whenever $z_t^k = 1$,

$$\pi^{\mathrm{exec}}(\mathsf{stay}) \ge (1-\epsilon_{\mathrm{IC}})\,\xi_t^{n_t(k)}(k) + \frac{\epsilon_{\mathrm{IC}}}{|\mathcal{A}_t^{n_t(k)}|} \ge (1-\epsilon_{\mathrm{IC}})\,\xi_t^{n_t(k)}(k) + \frac{\epsilon_{\mathrm{IC}}}{9}.$$

Taking $\liminf$ along $\{t : z_t^k = 1\}$ and using $\xi_t^{n_t(k)}(k) \to 1$ gives $1 - \epsilon_{\mathrm{IC}} + \epsilon_{\mathrm{IC}}/9$, which concludes the proof. □

C.3 Proof of Theorem 3

In this subsection, we analyze the expected detection delay. Recall that $T(H)$ is the stopping time when the global statistic Λ_t hits the threshold H.

Proof of Theorem 3. Fix any $k \in D(\tau)$. Consider an auxiliary scheme that monitors only the local CUSUM statistic at location k and stops when it crosses H. Let the (one-sided) likelihood-ratio increment be

$$\ell_k(x) := \log\frac{g_k(x)}{f_k(x)}, \qquad \mathbb{E}_{g_k}[\ell_k(X)] = I(g_k, f_k) > 0,$$

and define the sample-updated CUSUM at k along its visit times as W^k. Let $T_k'(H)$ denote the stopping time of this auxiliary scheme:

$$T_k'(H) := \inf\{t \ge \tau :\ W_t^k \ge H\},$$

where W_t^k is only updated when k is observed and otherwise remains unchanged.

For each location j, define the total number of times it is observed by any UAV before $T_k'(H)$:

$$S_j(H) := \sum_{t=\tau}^{T_k'(H)-1} z_t^j.$$

Because each time step produces exactly N observations across the N UAVs, $\sum_{j=1}^K z_t^j = N$ for all t, and hence

$$\sum_{j=1}^K S_j(H) = \sum_{t=\tau}^{T_k'(H)-1}\sum_{j=1}^K z_t^j = \sum_{t=\tau}^{T_k'(H)-1} N = N\,(T_k'(H)-\tau).$$

Therefore,

$$T_k'(H) - \tau = \frac{1}{N}\sum_{j=1}^K S_j(H), \qquad \text{and} \qquad \mathbb{E}[T_k'(H)-\tau] = \frac{1}{N}\,\mathbb{E}\left[\sum_{j=1}^K S_j(H)\right]. \tag{3}$$

Under the auxiliary scheme, the CUSUM at k is updated only when k is observed, and $T'_k(H)$ depends only on this local statistic. Hence, the number of CUSUM updates applied at k before stopping is exactly $S_k(H)$. Standard renewal arguments for the likelihood-ratio CUSUM with positive drift yield

$$\mathbb{E}[S_k(H)] \;\le\; \frac{H}{I(g_k, f_k)} + O(1), \qquad H \to \infty, \tag{4}$$

where the $O(1)$ term does not depend on H (see, e.g., 3, 4, 34).

By definition, $q_k(\tau)$ is the long-run fraction of sensing effort allocated to location k after τ, and thus we have the asymptotic relation

$$\lim_{H\to\infty} \frac{\mathbb{E}\big[\sum_{j=1}^{K} S_j(H)\big]}{\mathbb{E}[S_k(H)]} = \frac{1}{q_k(\tau)}. \tag{5}$$

Combining (3), (4), and (5) gives

$$\begin{aligned}\mathbb{E}[T'_k(H) - \tau] &= \frac{1}{N} \frac{\mathbb{E}\big[\sum_{j=1}^{K} S_j(H)\big]}{\mathbb{E}[S_k(H)]}\, \mathbb{E}[S_k(H)] \\ &\le\; \frac{1}{N q_k(\tau)} \left(\frac{H}{I(g_k, f_k)} + O(1) \right) = \frac{H}{N q_k(\tau) I(g_k, f_k)} + O(1). \end{aligned} \tag{6}$$

At any time t, the global statistic satisfies $\Lambda_t \ge \max_{j\in[K]} w_t^j \ge w_t^k$. Moreover, the deployed updating rule adds nonnegative inflation between consecutive visits to k (i.e., when $z_t^k = 0$), while the auxiliary CUSUM updates only at the visit times of k. Therefore, along the post-change visit times, the actually deployed statistic dominates the auxiliary sample-updated CUSUM. Consequently, $T(H) \le T'_k(H)$, and hence

$$\mathbb{E}[T(H) - \tau] \le \mathbb{E}[T'_k(H) - \tau] \le \frac{H}{N q_k(\tau) I(g_k, f_k)} + O(1) = \frac{H}{\rho_k(\tau) I(g_k, f_k)} + O(1).$$

□

D Supplemental Experiments

D.1 Implementation Details of Baseline Methods

We summarize the baseline dispatch policies used in Section 6. All baselines take the current per-cell statistic map and the current UAV positions as input, and choose the moves of N UAVs from the same local action space: the 8-neighborhood plus stay.

Random. Each UAV selects a feasible action uniformly at random from its local action space. This baseline has no tunable parameters beyond the environment-imposed feasibility constraints, namely boundary and collision constraints.

Greedy. The greedy baseline scores each UAV's one-step feasible next cells using the current statistic map and prefers moves toward locally larger statistic values. To enforce collision avoidance, it solves a one-step assignment over the feasible candidate cells via bipartite matching, so that UAVs are assigned to distinct next cells before executing the matched moves.

Patrolling. The patrolling baseline partitions the grid into N disjoint vertical stripes and assigns one UAV to each stripe. Within its stripe, each UAV follows a predefined boustrophedon route and reverses direction at the endpoints. At each step, it applies a stay-or-move rule: it stays if the local statistic at its current cell exceeds a preset local threshold; otherwise, it advances to the next cell on the route. The local threshold is chosen according to the principle used in a prior work [2].

Look-ahead. At each time step, the look-ahead baseline solves a mixed-integer linear program (MILP) over the next 5 steps to plan collision-free UAV routes that maximize the total expected anomaly evidence collected [2]. The model uses binary occupancy variables, motion constraints, and vertex-collision constraints, and optimizes a discounted sum of per-visit scores combining the current statistic map with a weighted uncertainty term. The uncertainty increases additively on unvisited locations and is incorporated via standard big-M linearization. After solving, the policy executes only the first planned move and replans at the next time step.

Table 2: Selected pseudo thresholds for different grid and fleet sizes.

Grid Size	5×5			10×10			30×30		
Fleet Size	1	2	5	2	5	10	5	10	30
$\hat{H}$	12	20	50	12	30	100	50	50	100

Table 3: Hyperparameters of DISPATCHPPO.

Hyperparameter	Value	Hyperparameter	Value	Hyperparameter	Value
actor lr	$1e{-4}$	critic lr	$1e{-4}$	optimizer	Adam
gamma	0.99	GAE λ	0.95	PPO epochs	15
clip param	0.2	mini-batches	1	entropy coeff	0.01
value loss coeff	0.5	hidden-sizes	[128, 128]	activation	ReLU
orthogonal init	True	number of layers	2	chunk length	30
rollout threads	100	training threads	16	eval interval	100
use GAE	True	proper time	True	huber δ	10.0

D.2 Implementation Details of DISPATCHPPO

For the CUSUM implementation, we set the minimum detectable shift to $\mu_{\min} = 1$. For Stage II execution, we use $C^U = 0.01$ and $\epsilon_{\text{IC}} = 0.01$ on 5×5 and 10×10 grids, and $C^U = 0.001$ and $\epsilon_{\text{IC}} = 0.01$ on 30×30 grids. During offline training, the detection threshold H is not yet calibrated. We therefore use a pseudo-threshold $\hat{H}$ to evaluate the resulting proxy delay and select the best checkpoint. The value of $\hat{H}$ should be sufficiently large to reveal the decreasing trend of detection delay during training. The selected $\hat{H}$ values for different grid and fleet sizes are summarized in Table 2. The PPO training hyperparameters of DISPATCHPPO are listed in Table 3. All experiments are implemented in Python and run on a Linux server with Intel Xeon Platinum 8358 CPUs and NVIDIA A100 (80GB) GPUs. Code is available at `https://github.com/xxia99/DispatchPPO`.

D.3 Additional Simulation Study

Ablation study. We ablate key design choices in DISPATCHPPO to understand how they affect Stage I learning, focusing on (i) the DISPATCHNET parameterization, (ii) the autoregressive collision-aware decoder, and (iii) the reward shaping. In Figure 7, *Ours* uses the full DISPATCHNET with autoregressive decoding and trains with $R_t = R_t^U + R_t^A$. We compare against: (i) *w/o* R_t^U, which keeps the architecture but trains with $R_t = R_t^A$ only; (ii) *w/o Decoder*, which removes the autoregressive decoder; and (iii) *w/o DispatchNet*, which removes the full encoder-decoder and reduces to a standard PPO policy. Because these variants do not all share the same training reward, we report two common metrics: (i) the average hit rate achieved during training and (ii) the detection delay measured during periodic validation (reported as $\log(\text{ARL}_1)$).

Figure 7(a) shows that the full model learns substantially higher hit rates, quickly reduces $\log(\text{ARL}_1)$, and converges to the lowest delay. Removing R_t^U causes a small but consistent degradation, suggesting that this term in the reward function improves post-change performance by promoting broader exploration during learning. In contrast, removing the autoregressive decoder substantially worsens both the training hit rate and the validation delay, as it eliminates the mechanism for coordinating multi-UAV actions sequentially under collision constraints. Removing DISPATCHNET further degrades performance relative to *w/o Decoder*, indicating that the encoder's spatial/relational features are beneficial for decision-making under partial observations. Overall, these results suggest that (i) the autoregressive decoder and DISPATCHNET architecture are important to improve coordinated joint control, and (ii) the coverage term R_t^U helps improve detection performance via more effective exploration.

Time Efficiency Comparison. Figure 8 compares wall-clock decision time per step (log scale) for the heuristic-based *Greedy*, the optimization-based *Look-ahead*, and our learning-based DISPATCHPPO. As the fleet size increases (left), DISPATCHPPO remains in the millisecond regime and grows mildly with N. This aligns with Proposition 1, which guarantees linear scaling with the number of

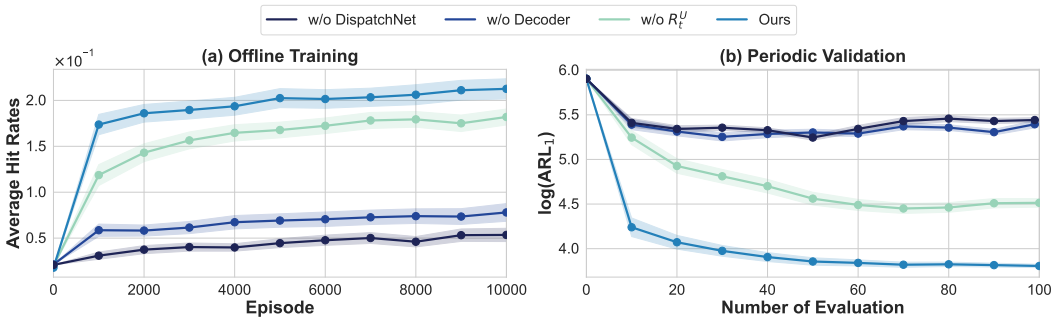


Figure 7: Ablation study of DISPATCHPPO. (a) Offline training: average anomaly hit rate (fraction of anomalous locations visited per step). (b) Periodic validation: evaluated detection delays $\log(\mathrm{ARL}_1)$. Experiments are conducted on a medium-scale simulation, a 10×10 grid with $N = 2$ UAVs.

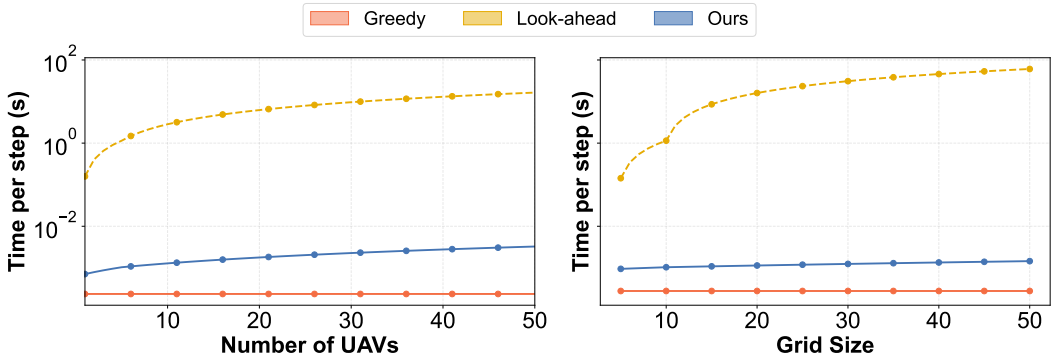


Figure 8: Per-step decision time (log scale). Wall-clock time per decision (seconds) for *Greedy*, *Look-ahead*, and *Ours*, as a function of (left) fleet size N (on a fixed 10×10 grid) and (right) grid size (with fixed $N = 5$ UAVs).

UAVs for constant local action sets. In contrast, *Look-ahead* becomes orders of magnitude slower, increasing from $\approx 10^{-1}$ s/step to $\mathcal{O}(10)$ s/step as N grows. Scaling the grid size (right) shows a similar pattern: DISPATCHPPO stays near 10^{-3} s/step with only a modest increase due to spatial feature computation, while *Greedy* remains nearly constant and *Look-ahead* scales poorly.

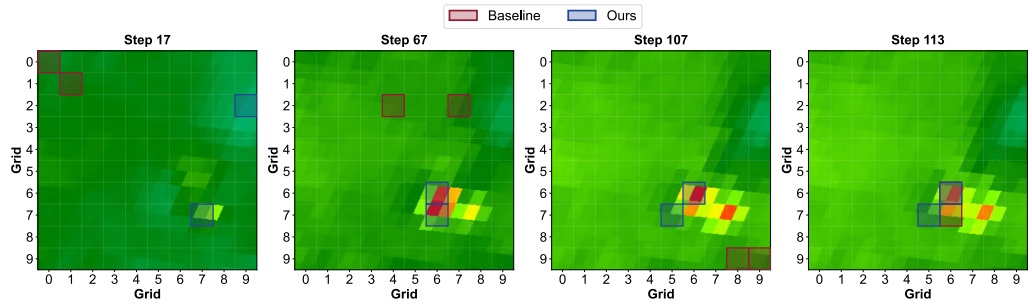


Figure 9: Comparison of dispatch developed by DISPATCHPPO and the baseline Lookahead policy [2]) on a reconstructed real wildfire case study. The red intensity indicates the wildfire.

Table 4: Wildfire detection performance of the five routing strategies.

Performance	Lookahead	Patrolling	Greedy	Random	Ours
Detection time (min)	110	135	115	250	57
Decision Time (Seconds/Step)	1.99	0.0011	0.0013	0.0016	0.0096

D.4 Additional Case Study: Real-World Wildfire Monitoring.

We further evaluate DISPATCHPPO on early wildfire detection using real satellite shortwave infrared imagery that captures evolving thermal intensity over time. We reconstruct the development of the Cherrywood Fire in southern Nevada on May 20, 2021 (`https://cimss.ssec.wisc.edu/satellite-blog/archives/40926`) by mapping pixel-level red intensity to a 10×10 grid of locations, where each location aggregates local intensity and is standardized using unaffected areas. We focus on a 500-minute time window, and adjacent grid moves correspond to feasible UAV travel within one step. Table 4 shows that DISPATCHPPO triggers the quickest alarm while remaining computationally lightweight per decision step. Figure 9 visualizes 2-UAV routes under our method and a baseline (*Look-ahead*) at representative time steps. Early in the event (e.g., Step 17), DISPATCHPPO disperses UAVs for exploration, avoiding redundant coverage. As the wildfire becomes significant (e.g., Step 67), UAVs reallocate toward the emerging hotspot and quickly transition to evidence-based focusing (Steps 107-113). In contrast, the baseline becomes trapped in less-informative areas and reaches the affected region substantially later, leading to delayed detection. This result further demonstrates the practical effectiveness of the proposed approach.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes; the abstract/introduction state the route-wise quickest-detection problem and summarize the formulation, method, theory, and experimental evidence consistent with the results in Sections 3–6 and the appendix.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes; key modeling/theory assumptions are stated in Section 3, and the paper includes a limitations discussion in Section 7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.

- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Yes; assumptions are stated in Section 3 and in the theorem statements, and full proofs (with referenced lemmas) are provided in Appendix B and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes; the environments, anomaly generation, ARL_0 calibration, baselines, and evaluation protocol are specified in Section 6, with additional implementation details in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.

- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code and data are publicly available at `https://github.com/xxia99/DispatchPPO`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Yes; we report training/evaluation protocols, ARL_0 calibration targets, and key hyperparameters in Section 6 and Appendix D.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Yes; the main simulation results report variability over Monte Carlo replications and state the number of replications and monitoring horizons in Section 6.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We report the CPU/GPU types used in Appendix D, and per-step decision-time comparisons for execution in Figure 8 of Appendix D.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes; the work uses simulation and a standard robot testbed without human subjects, and we have reviewed and complied with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes. Section 7 discusses positive uses (wildfire response, search and rescue, infrastructure monitoring), the dual-use concern of surveillance, and safeguards: the method detects statistical changes in sensor streams without identifying individuals, and deployments should follow the regulations on UAV operation and data collection.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)? item[] Answer: [N/A]

    Justification: N/A; we do not release high-risk generative models or sensitive datasets, and the contributions are algorithmic components evaluated in simulation and controlled testbeds.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes; we cite all external assets used in this paper (e.g., algorithms, Robotarium, and the wildfire imagery source).

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide well-commented documentation together with our codebase at `https://github.com/xxia99/DispatchPPO`.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: N/A; the paper does not involve crowdsourcing or human-subjects experiments.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: N/A; the work does not involve human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: N/A; LLMs are not used as part of the proposed method or experiments.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.