Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira,
Edoardo Patelli, Hélder S. Sousa, Rui A. Silva and Monica Santamaria
 Published by Research Publishing, Singapore.
doi: 10.3850/ESREL2026061419_esrel26-p26319-cd

Context-Aware Multi-UAV Fleet Coordination for Flood Response using Deep Reinforcement Learning

Kilian Thoma
Munich University of Applied Sciences HM, 80335 Munich. E-mail: kilian.thoma@hm.edu

Weian Guo
Sino-German College of Applied Sciences, Tongji University, 201804, Shanghai, China. E-mail: guoweian@tongji.edu.cn

Marcin Hinz
Munich University of Applied Sciences HM, 80335 Munich. E-mail: marcin.hinz@hm.edu

With the increasing frequency of severe flood events, the need for rapid and effective disaster response is critical. While autonomous fleets are important for situational awareness, conventional guidance systems utilize static trajectory planning, which proves inefficient in environments with heterogeneous infrastructure density. To address this limitation, we introduce a centralized coordination framework. We extend the standard Deep Reinforcement Learning (DRL) agents by fusing egocentric perception with static geospatial layers (OpenStreetMap, Digital Elevation Model) via a dual-stream Late-Fusion Architecture. This transforms the task from blind coverage to strategic navigation, allowing the fleet to proactively prioritize high-value residential zones. We evaluate the approach in a high-fidelity simulation of the 2021 Ahr Valley flood. Compared to a systematic Lawnmower baseline, the context-aware agents demonstrate superior efficiency. While maintaining comparable spatial coverage (64.3%), the system achieves a discovery rate of 50.5%, representing a net semantic gain of 4.9%. Crucially, the fleet exhibits emergent learned coordination and reactive deviation, diverting from linear paths to trace infrastructure corridors while maintaining safe inter-agent separation. These results validate that geospatial context awareness serves as a robust surrogate for global planning in time-critical environments.

Keywords: Multi-UAV Fleet, Deep Reinforcement Learning, Flood Response, Context-Aware Coordination

1. Introduction

The increasing frequency and intensity of extreme weather events, driven by climate change, have established a critical need for rapid and scalable disaster response mechanisms (Tradowsky et al., 2023). In the immediate aftermath of floods, such as the catastrophic 2021 event in the Ahr Valley, Germany, infrastructure is often compromised, rendering ground-based rescue operations hazardous or impossible (Surmann et al., 2022). The speed of survivor identification and structural assessment determines the outcome of the rescue operation. While often broadly categorized under UAV swarms, we specifically structure our system as a coordinated multi-UAV fleet. Unlike decentralized swarms that rely on emergent behavior, a fleet architecture allows us to leverage global context for strategic coordination.

The logic for this centralized approach is grounded in computational capability and global data accessibility. While modern UAVs possess increasing onboard processing power, the real-time fusion of high-resolution geospatial layers remains computationally prohibitive. Specifically, utilizing Digital Elevation Models (DEM) and OpenStreetMap (OSM) data alongside deep neural networks imposes severe SWaP (Size, Weight, and Power) constraints. By offloading the computational demand to a high-performance Ground Control Station (GCS), we decouple the decision-making intelligence from the robotic platform. This architecture enables the fleet to use static map databases and computationally expensive Late-Fusion architectures that would exceed the computational and thermal envelopes of standard embedded flight controllers. Consequently, the

agents' onboard Global Navigation Satellite System (GNSS) positioning can be consistently correlated with global static map layers to maximize semantic yield without compromising the real-time control loop.

Current deployment strategies typically rely on one of two paradigms: rigid pre-planning or purely reactive control. Pre-planned Lawnmower patterns ensure systematic scanning but lack the flexibility to adapt to real-time observations or terrain topology. Conversely, reactive agents navigate autonomously to avoid collisions but often suffer from a semantic gap, treating all geographic features equally. While Meyer et al. (2025) successfully demonstrated the viability of Deep Reinforcement Learning (DRL) for flood coverage, their approach relied on abstract state information and a coarse grid resolution ($10m$). This abstraction simplifies coordination but fails to capture the intricate topology of urban disaster zones.

To address these limitations, we introduce a context-aware DRL framework that increases both the fidelity and efficiency of the fleet. Unlike previous methods, our agents operate with realistic, egocentric perception ($2m$ resolution), mimicking the constraints of onboard cameras. To overcome the challenge of partial observability, we enhance the observation space with static geospatial layers. This allows the fleet to naturally learn prioritization behaviors, proactively navigating toward residential zones while ignoring empty terrain, without hard-coded rules.

2. Problem Statement

To align abstract simulations with the real-world deployment, we construct a data-driven environment grounded in the 2021 Ahr Valley disaster(Copernicus EMS, 2021). Unlike abstract grid worlds, this setup incorporates realistic topography, uneven infrastructure distribution, and non-holonomic flight dynamics. The following subsections detail how raw geospatial data is transformed into a coherent simulation explicitly designed to train robust, context-aware policies.

2.1. *Flood environment*

To ensure metric accuracy for distance and velocity calculations, all geospatial layers are projected into a unified coordinate system (UTM Zone 32N, EPSG:25832). To generate a robust training set, we employ a grid-based partitioning strategy covering the entire geographical bounding box of the disaster zone. We divide the global map into aligned $1\,km \times 1\,km$ tiles. Notably, this grid encompasses the surrounding topology, including mountain ridges and uninhabited valleys. This inclusion of negative samples (tiles with zero damage) is essential for training the agent to actively differentiate between relevant search sectors and empty terrain, preventing false-positive biases. A representative example of such a training tile is depicted in Figure 1.

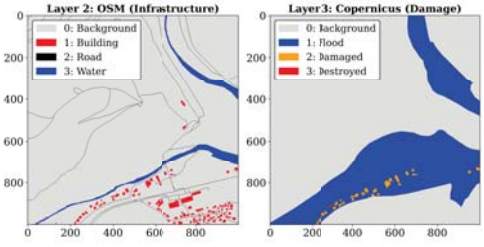

Fig. 1. Representative 1×1 km Training Tile. **Left:** OSM layer showing roads and building footprints. **Right:** Corresponding Flood Ground Truth.

2.2. *Data processing*

To ensure neural network stability, all geospatial inputs are normalized to the floating-point range $[0, 1]$. A critical challenge in the Copernicus EMS dataset is the spatial sparsity of the vector layer, where destroyed buildings are encoded as single coordinate centroids. We address this via spatial semantic fusion: by spatially correlating sparse damage points from Copernicus with building polygons from OSM, we rasterize the entire building footprint as a volumetric target.

Furthermore, we employ a feature-hierarchic intensity encoding for the infrastructure layer. To aid the convolutional neural network (CNN) in feature segregation, semantic classes are mapped

to discrete intensity bands: Background (0.0), Buildings (0.33), Roads (0.66), and Water (1.0). This ordinal scaling is designed to reflect semantic prioritization: by assigning the highest intensity to Water bodies (1.0), we create a strong activation signal for the primary hazard source. Roads (0.66), which often run parallel to rivers in valley topologies, are emphasized as high-probability search corridors, while Buildings (0.33) are contextualized relative to these dominant geographic features.

2.3. *Flight dynamics*

To ensure comparability with established baselines, we strictly adopt the physical constraints defined by Meyer et al. (2025). The agents are modeled as **fixed-wing UAVs** operating at a constant altitude $h = 200\,m$ with a restricted mission horizon of $T_{max} = 45\,s$. Given a standard optical Field of View ($FoV = 50°$), this results in a ground sensor footprint of approx. $192 \times 192\,m$. By fixing the altitude, the navigation problem is effectively reduced to a 2D kinematic manifold.

The agents are modeled as non-holonomic units operating within a discrete-time framework ($dt = 0.2s$). The state evolution follows a standard unicycle model:

$$x_{t+1} = x_t + v \cdot \cos(\theta_t) \cdot dt \tag{1}$$

$$y_{t+1} = y_t + v \cdot \sin(\theta_t) \cdot dt \tag{2}$$

To ensure strict metric consistency we also fix the velocity at $v = 25.0\,m/s$ ($90\,km/h$). This combination of high velocity and short duration simulates a *Rapid Sector Assessment* mission profile, which requires forward planning rather than hover-and-scan behaviors. The action space restricts yaw rates to $\omega \in \{-15°/s, 0°/s, +15°/s\}$. This results in a minimum turning radius of $R \approx 95\,m$. While substantial, this radius is physically mandated by the fixed-wing aerodynamics at $v = 25\,m/s$. Tighter maneuvers would require aggressive banking ($> 35°$), which would tilt the sensors away from the target area, leading to gaps in the scanned surface.

2.4. *Agent's visual perspective*

The proposed architecture operates exclusively on high-resolution local sensor data, discarding the assumption of global observability found in simplified models. To ensure *localized consistency* in bandwidth-constrained environments, the observation generation assumes a GNSS-driven alignment: the agent's onboard position is used to crop and rotate the static semantic layers (OSM/DEM) relative to its current heading. This allows for precise geo-joining of internal map data with external sensor inputs.

We address the constraints of restricted local sensing by employing a multi-resolution observation stack. It combines **global priors** with **simulated live perception**:

(1) **Global Priors (OSM, DEM):** Static map data retrieved from the central database, aligned with the agent's GNSS position.
(2) **Simulated Perception (Flood):** The ground-truth damage layer is fed as a visual channel, simulating the output of a robust onboard semantic segmentation module.
(3) **Coverage History:** Spatially aggregated context ($500 \times 500\,m$) to facilitate boundary anticipation.

To decouple the decision-making logic from the global cardinal orientation, all visual inputs are rotated to align with the agent's current velocity vector. Additionally, we implement a Virtual Confinement Mechanism via the coverage channel, rendering the domain exterior as fully saturated (1.0). This design utilizes the coverage maximization objective to induce a natural boundary. Since the exterior offers zero utility, the policy is implicitly biased back toward the area of interest without explicit collision penalties.

3. System Architecture

We formulate the multi-UAV flood response mission as a centralized control problem, interacting with the environment via the Gymnasium interface. To enable training on complex geospatial data ($> 10^6$ steps), the underlying physics engine employs Just-In-Time (JIT) compilation via the Numba library (Lam et al., 2015). This

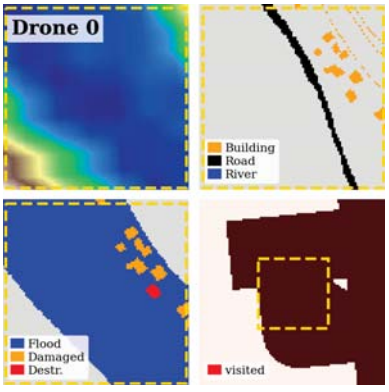

Fig. 2. Visualization of the Egocentric Observation Space. From left to right: DEM (Terrain), OSM (Infrastructure), Damage (Flood/targets), and Coverage History. The yellow dotted line indicates the active input region fed into the CNN, highlighting the scale difference between the local semantic layers and the zoomed-out coverage context.

translates Python byte-code into optimized machine code, thereby eliminating typical interpreter bottlenecks. This system transforms the physical unicycle simulation into a numerical decision-making process, where a single neural network policy jointly controls the fleet to maximize the discovery of critical infrastructure.

3.1. *Actions*

The agent operates in a discrete action space designed for high-frequency decision-making. We utilize a multi-discrete action space to control the squad of $N = 3$ drones simultaneously. Each drone i independently selects an action $a_t^i \in \{0, 1, 2\}$ at every timestep, mapping directly to yaw rate adjustments: Left ($-15°/s$), Straight ($0°/s$), and Right ($+15°/s$). To stabilize flight trajectories and emulate realistic communication latency, we implement an action repeat mechanism ($k = 4$). This reduces the effective control frequency to $0.8s$, smoothing the flight path while maintaining sufficient agility.

3.2. *States*

In addition to visual input, the agent receives a compact vector representation of the fleet's kinematics. The state vector $S_{vec} \in \mathbb{R}^{12}$ consists of position and heading information for all 3 drones. Coordinates (x, y) are normalized to the unit range $[0, 1]$ relative to the map size ($1000m$), while heading is encoded as trigonometric components $(\sin\theta, \cos\theta)$ to avoid discontinuity at $360°$. This precise localization data is critical for computing termination conditions. The episode is immediately terminated if any agent violates the map boundary ($S_{pos} \notin [0, 1]$) or if the Euclidean distance between any two agents falls below the safety threshold ($d_{ij} < 50m$).

3.3. *Observations*

The global observation space is a dictionary containing the kinematic state vector and a stacked visual tensor of shape $(12, 96, 96)$. This tensor aggregates the egocentric viewpoints of all N UAVs (4 channels per unit), allowing the centralized policy to spatially correlate their fields of view. The channels Terrain, Infrastructure, Flood, and Coverage are stacked depth-wise. Regarding the boundary handling described in Section 2, the virtual border is implemented here as data saturation: areas outside the valid map boundaries are encoded with a constant value of 1.0 in the coverage channel. This presents the border to the network not as a physical wall, but as a zone of zero information gain.

3.4. *Rewards*

The reward function acts as the primary driver for the emergent behavior. It is composed of a dense exploration term, sparse semantic discovery terms, and strict safety penalties:

$$R_t = R_{explore} + R_{semantic} + R_{safety} \quad (3)$$

We employ a **magnitude-based prioritization** strategy rather than standard normalization. By maintaining a difference of several orders of magnitude between the dense exploration signal ($w_{pixel} \approx 10^{-4}$) and the sparse safety penalties ($w_{crash} \approx 500.0$), we enforce a behavioral hard constraint: the agent learns that collision avoidance is non-negotiable. The specific weights for semantic targets follow a hierarchy ($w_{flood} < w_{damaged} < w_{destroyed}$), treated as searchable hyperparameters during optimization.

3.5. *Network Architecture*

The policy utilizes a hierarchical late-fusion architecture. The visual encoder is a 4-layer CNN utilizing rectified linear unit (ReLU) activations. It comprises a strided initial layer (5×5, stride 2) followed by three convolutional blocks (3×3) with 2×2 Max-Pooling, mapping the 96×96 input to a 512-dimensional embedding (256 channels $\times 6 \times 6$).

In parallel, the kinematic state is encoded via a 2-layer MLP ($32 \rightarrow 64$ units). Feature streams are concatenated per agent ($512 + 64 = 576$) and subsequently aggregated across the fleet ($576 \times 3 = 1728$) into a final fusing network ($1024 \rightarrow 512$) to predict joint actions.

4. Training of the Agents

The training process is implemented using the Stable-Baselines3 library (Raffin et al., 2021), utilizing Proximal Policy Optimization (PPO) (Schulman et al., 2017) as the core algorithm.. PPO was selected for its sample efficiency and proven stability in high-dimensional state spaces, which is critical when mapping raw egocentric vision to discrete fleet commands.

4.1. *Hyperparameter Optimization (HPO)*

The efficacy of the proposed reward function is highly sensitive to the relative scaling of its components. Manual tuning is infeasible due to the non-linear interaction between safety constraints and exploration incentives. We conduct a distributed Bayesian optimization search using the Optuna framework (Akiba et al., 2019), based on the Tree-structured Parzen Estimator (TPE) (Bergstra et al., 2011). Crucially, this process optimizes both algorithmic parameters and reward weights simultaneously. The specific search ranges are listed in Table 1.

4.2. *Multi-Objective Optimization (MOO)*

Standard Reinforcement Learning collapses all goals into a single scalar return. However, maximizing this scalar often leads to reward hacking, where an agent exploits dense signals without achieving the actual semantic mission. To prevent this and remove human bias from the evaluation, we decouple the training reward from the selection criteria. Consistent with the optimization framework defined previously, the TPE sampler targets three operational goals: **discovery rate** (the percentage of total damaged infrastructure area captured at least once within any UAV's sensor footprint during the $45,\mathrm{s}$ mission), **coverage rate** (the total unique geographic surface area scanned by the fleet's sensors at least once), and **survival steps** (average flight duration). This process generates a **pareto front** of solutions, from which the final agent is selected.

Table 1. Hyperparameter Search Space and Reward Configuration. The optimization includes both PPO algorithm parameters and the specific weights for the reward function.

Parameter	Range / Value	Sampling
Algorithmic Parameters		
Learning Rate	$10^{-5} - 5 \cdot 10^{-4}$	Log
Entropy Coef	$10^{-6} - 0.02$	Log
Reward Weights (Objective Function)		
Pixel (Dense)	$10^{-5} - 5 \cdot 10^{-4}$	Log
Flood	$0.005 - 0.10$	Step 0.005
Damaged	$0.01 - 0.20$	Step 0.005
Survival	$0.00 - 0.10$	Step 0.005
Crash Penalty	$50.0 - 500.0$	Step 25.0
Time Penalty	$0.001 - 0.02$	Log
Completion	50.0	Fixed

4.3. *Train & Test Strategy*

To ensure robust evaluation, we implement a fixed-count stratified sampling strategy for map selection, sequestering a balanced set of important (high damage) and empty tiles for validation. Beyond map selection, we strictly differentiate the agent initialization logic between training and testing to simulate realistic deployment conditions.

During training, we utilize a uniform sector spawn strategy. Agents are initialized along all four cardinal directions (North, East, South, West) with positions sampled from a uniform distribution ($U \sim [0, L]$). This acts as a data augmentation technique, forcing the policy to learn robust navigation and formation assembly from any arbitrary entry vector.

For evaluation, we switch to a probabilistic initialization strategy where agent starting positions are sampled from a multivariate Gaussian distribution. This simulates the fleet transitioning from a previous area of interest to the current target. Agents are initialized near ideal formation points but with added Gaussian noise applied to both lateral and longitudinal coordinates. This noise mimics real-world operational factors such as GNSS deviations or temporal variations. This ensures that the reported test metrics reflect the agent's ability to recover from realistic initial imperfections, validating the robustness of the localized consistency assumption.

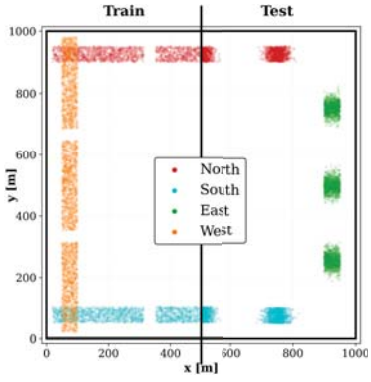

Fig. 3. Visualization of Spawning Strategies. Training (Left) maximizes geometric variance for robustness. Testing (Right) mimics realistic GNSS drift and formation imperfections during AOI entry.

5. Results

We evaluate the performance of the proposed framework within the data-driven simulation described in Section 2. To ensure a fair comparison, the learned policy is benchmarked against a standard Lawnmower heuristic. We define the Lawnmower baseline as a rigid, pre-planned boustrophedon pattern (alternating parallel sweeps) designed to scan the map systematically without sensory feedback. Both the DRL agents and the heuristic operate within the exact same environment instance and are subject to identical physical constraints. Consequently, any observed variance in performance is attributable solely to the efficiency of the decision-making logic rather than physical advantages.

5.1. *Hyperparameter Optimization and Model Selection*

The final configuration, derived from the automated search, addresses the conflicting multi-objective function comprising discovery rate, coverage, and survival time.

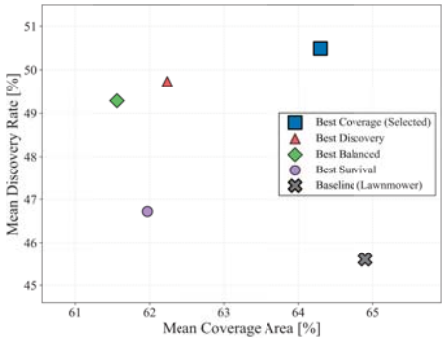

Fig. 4. Performance comparison of the four retrained RL candidates versus the Lawnmower Baseline. The scatter plot displays the mean discovery rate against the mean coverage area evaluated on the Test Data.

Figure 4 presents the aggregated performance of four distinct candidates selected from the pareto front alongside the Lawnmower baseline. All metrics represent the mean performance derived from the unseen test dataset. The axes represent the trade-off between semantic discovery rate (y-axis) and mean coverage (x-axis). Based on this distribution, the best coverage agent (blue square) was selected for the detailed comparative analysis, as it offers the optimal balance between spatial scanning and target identification.

5.2. *Comparative Performance on Test Scenarios*

Final evaluation was performed on a dedicated test set of unseen geography, comprising both high-damage scenarios and featureless terrain. The results presented are averaged over 20 random seeds per scenario to account for initialization noise.

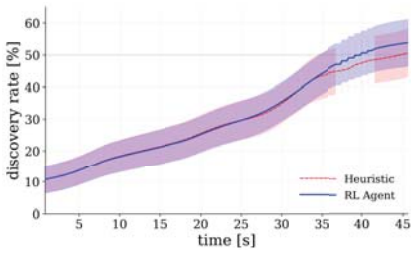

Fig. 5. Time-to-Discovery Analysis. The RL Agent (Blue) consistently outperforms the Heuristic (Red) throughout the episode. The gap widens significantly after $t = 30s$, indicating superior prioritization of relevant sectors.

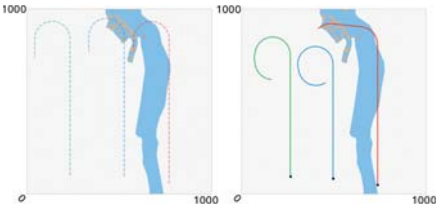

Fig. 6. Trajectory Comparison. **Left:** The Lawnmower baseline rigidly follows geometry. **Right:** The RL agent adapts dynamically to the damage clusters. **Legend:** The **red trajectory** tracks the flight path of Drone 3. Background colors denote **flood water (blue)** and **damaged buildings (orange)**.

The Lawnmower baseline achieves a marginally higher spatial coverage (64.90%). It is crucial to note that within the convex boundaries of a $1 \times 1\,km$ sector, the Boustrophedon strategy represents a coverage-maximizing geometric primitive for exhaustive area scanning. Consequently, high coverage is an inherent property of this trajectory design. However, due to its spatially agnostic nature, it yields a lower discovery rate (45.60%), as it distributes search effort uniformly across empty terrain and critical infrastructure alike.

In contrast, the RL Agent achieves comparable coverage (64.29%) but demonstrates superior utility in meaningful data acquisition, reaching a discovery rate of 50.49%. This constitutes an absolute net semantic gain of $+4.9\%$. Crucially, the aggregate final metric fails to capture the temporal dynamics. As illustrated in Figure 5, the RL agent exhibits decisive **semantic acceleration**. By inferring high-probability sectors from geospatial priors, the policy identifies flood zones and damaged buildings significantly earlier in the episode. Conversely, the baseline incurs high latency, as discovery is contingent upon the pre-planned trajectory physically intersecting with targets.

5.3. *Behavioral Analysis and Fleet Coordination*

The quantitative advantage is supported by a qualitative analysis of the flight behavior. Figure 6 contrasts the trajectories of both approaches.

The baseline executes a rigid sweep pattern regardless of the terrain content. While systematic, this forces the fleet to traverse zero-utility terrain with the same fidelity as urban areas, strictly ignoring environmental features.

In contrast, visual inspection of the RL agent reveals an emergent Hybrid Search Strategy. In featureless terrain, the agent adopts a quasi-systematic scanning pattern (straight flight vectors), which effectively mimics the Lawnmower to maximize the dense coverage reward. However, a crucial behavioral divergence occurs upon detection. Telemetry indicates a **reactive deviation**: when the agent detects important features in its vision, it breaks the systematic vector to trace the topology. This confirms that the policy has learned to balance global exploration (quasi-systematic scanning) with local exploitation (following targets), whereas the baseline is restricted to blind exploration.

Finally, we observe effective fleet coordination. Leveraging the centralized architecture, the agent learns to actively minimize sensor overlap to maximize the global coverage. Quantitative analysis reveals a mean inter-agent separation distance of $\mu = 249\,m$. This value exceeds the sensor footprint ($192\,m$) by a factor of $1.3\times$, confirming that the system successfully learned a robust deconfliction strategy without requiring hard-coded formation rules.

6. Discussion and Conclusion

6.1. *Discussion of Results*

Interpreting these results requires considering the simulation's constraints. Specifically, the performance metrics must be weighed against the strict temporal limits and the spatial saturation of the operational domain.

The relevance of marginal gains: While a net discovery gain of $+4.9\%$ might appear incremental, its significance is amplified by the extremely short mission duration of only $45\,s$. Achieving this yield within less than a minute highlights the model's efficiency. By prioritizing high-probability sectors immediately, the agent maximizes the information density per flight second. This early accumulation of situational awareness is far more valuable to first responders than a delayed, albeit exhaustive, map completion.

Sensor-to-area saturation: The experimental setup ($1\,km^2$ domain, $192\,m$ sensor FoV, $25\,m/s$ velocity) creates a spatially saturated environment. This convexity inherently favors the Lawnmower baseline, as systematic sweeping is near-optimal in small, bounded spaces where empty traversals are rare. The fact that the DRL policy outperformed the heuristic in this specific domain, while operating under the significantly higher complexity of partial observability, strongly validates the robustness of the learned behavior.

Conservative lower bound: We hypothesize that the reported performance gap represents a conservative lower bound. On the current small maps, semantic features (e.g., rivers, roads) are frequently truncated by boundaries, preventing the emergence of long-horizon tracking behaviors. In realistic, large-scale theaters where targets are sparsely distributed, the Lawnmower's blind sweeping would become exponentially inefficient, whereas the context-aware agent could fully exploit its ability to ignore empty terrain and trace infrastructure corridors.

6.2. *Conclusion*

In this work, we presented a context-aware deep reinforcement learning framework for autonomous multi-UAV flood response. By combining egocentric vision with geospatial priors (OSM/DEM), the proposed method enables agents to prioritize critical infrastructure using only local observations. The evaluation confirms that the policy not only achieves higher semantic discovery rates than a systematic baseline but also exhibits intelligent, reactive flight patterns. While the purely systematic approach remains efficient in small, convex boundaries, our results demonstrate that context-aware agents provide a decisive time advantage in identifying survivors. Future work will focus on validating scalability in large environments ($> 10, \mathrm{km}^2$) and isolate DRL efficiency gains from the inherent advantage of OSM/DEM priors.

References

Akiba, T., S. Sano, T. Yanase, T. Ohta, and M. Koyama (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proc. 25th ACM SIGKDD*, pp. 2623–2631.

Bergstra, J., R. Bardenet, Y. Bengio, and B. Kégl (2011). Algorithms for hyper-parameter optimization. In *Proc. NeurIPS*.

Copernicus EMS (2021). Flood in western germany, grading monitor [emsr517].

Lam, S. K., A. Pitrou, and S. Seibert (2015). Numba: A llvm-based python jit compiler. In *Proc. LLVM-HPC*, pp. 1–6.

Meyer, M., S. Schorer, A. Żurek, and M. Hinz (2025). Uav swarm coordination for flood area coverage using reinforcement learning. *Proc. ESREL*.

Raffin, A., A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann (2021). Stable-baselines3: Reliable reinforcement learning implementations. *J. Mach. Learn. Res. 22*(268), 1–8.

Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov (2017). Proximal policy optimization algorithms. *arXiv:1707.06347*.

Surmann, H. et al. (2022). Deployment of aerial robots during the flood disaster in erftstadt/blessem 2021. In *Proc. IEEE SSRR*.

Tradowsky, J. et al. (2023). Attribution of heavy rainfall events leading to severe flooding in western europe. *Clim. Change*.