DeepSet-Enhanced Edge Reinforcement Learning for UAV Autonomous Landing and Takeoff at Portable Vertiports

Zhirun Li*, Tiying Gao†, Chengyi Qu‡§
*University of New Mexico, USA, †Florida Atlantic University, USA, ‡Florida Gulf Coast University, USA.
*zhirunli@umn.edu, †tgao2024@fau.edu, ‡cqu@fgcu.edu

***Abstract*—Unmanned aerial vehicles (UAVs) are increasingly deployed for delivery, monitoring, and emergency response, yet their large-scale coordination in congested airspace remains constrained by latency, bandwidth limits, and collision risks. Portable vertiports offer a mobile infrastructure for energy replenishment and structured takeoff/landing, but simultaneous operations require ultra-reliable and low-latency decision-making across multiple agents. This paper presents DREAM (DeepSet-enhanced Reinforcement learning for Edge-based control in Ambiguous and dynamic environMents), a distributed edge intelligence framework that integrates deep reinforcement learning (DRL) with multi-access edge computing (MEC) principles. Each UAV operates as an edge client that performs real-time inference locally while synchronizing its policy with nearby vertiport servers through lightweight model updates, enabling scalable coordination without dependence on cloud backhaul. The framework adopts a centralized-training-and-decentralized-execution (CTDE) paradigm with a permutation-invariant DeepSet encoder and a safety-constrained Proximal Policy Optimization (PPO-GAE) network. Simulation results under variable network latency and dynamic obstacles demonstrate that DREAM achieves a 98 % mission success rate, reduces collisions by over 90 %, and sustains stable performance within 100 ms edge-to-UAV latency budgets. These results highlight the feasibility of edge-assisted multi-agent learning for autonomous vertiport operations and its potential integration into future 5G/6G-enabled aerial networks.**

***Index Terms*—Multi-Access Edge Computing, Unmanned Aerial Vehicles, Deep Reinforcement Learning, Vertiport Operations, Intelligent Collision Avoidance.**

I. Introduction

Unmanned aerial vehicles (UAVs) have become indispensable in modern intelligent transportation, surveillance, and disaster-response systems due to their flexibility, mobility, and rapid deployment capability. However, their large-scale coordination in shared airspace continues to pose significant challenges in terms of safety, latency, and energy efficiency. In particular, simultaneous takeoff and landing at portable vertiports introduces dense air-traffic interactions where multiple UAVs must negotiate constrained spaces under uncertain dynamics. Traditional centralized controllers rely heavily on cloud computation and high-bandwidth communication links, which can introduce delays and bottlenecks in real-time decision-making [1], [2]. Conversely, purely onboard autonomy often lacks global awareness and leads to suboptimal coordination, especially when multiple UAVs operate in close proximity [3]. These limitations underscore the need for a hybrid paradigm capable of combining distributed learning and edge-level intelligence for autonomous aerial systems.

Recent research in edge computing and networked control has offered promising solutions by enabling real-time analytics near the data source [4]. The concept of multi-access edge computing (MEC) allows UAVs to offload computational tasks to nearby edge servers, significantly reducing latency while maintaining operational safety. MEC-based UAV systems have been explored for applications such as aerial monitoring [5], path planning [6], vehicular networking [7] and other tasks [8]. However, most of these studies rely on deterministic models or static channel assumptions, which limit scalability in dense and dynamic environments. Meanwhile, deep reinforcement learning (DRL) has emerged as a data-driven alternative capable of handling uncertainty and adapting to complex mission objectives [9]. DRL frameworks have been applied to flight control [10], collision avoidance [11], and energy-aware trajectory optimization [12], yet few works fully integrate them with edge computing to handle the dual constraints of communication delay and distributed inference inherent in real UAV networks.

At the intersection of these research directions lies edge intelligence, DRL agents always operate under a centralized training and decentralized execution (CTDE) paradigm. Recent efforts have incorporated federated learning for UAV collaboration, allowing policy training across multiple edge nodes without sharing raw data [9]. Other studies have demonstrated that federated edge learning can enhance privacy and scalability for multi-agent coordination in dynamic networks [13]. Despite these advances, the majority of current approaches do not explicitly model network latency, resource contention, or the physical topology of vertiport-based UAV swarms.

In this paper, we introduce *DREAM (DeepSet-enhanced Reinforcement learning for Edge-based control in Ambiguous and dynamic environMents)*, a distributed learning framework for simultaneous UAV takeoff and landing at portable vertiports. As illustrated in **Fig. 1**, the vertiport serves as both a physical coordination hub and a micro-edge node that hosts lightweight computation for UAVs within its proximity. Each UAV functions as an edge client capable of performing real-time inference onboard, while periodically synchronizing policy parameters with a vertiport-level edge server through

§Corresponding Author.
This material is based upon work supported in part by the Dendritic: A Human-Centered AI and Data Science Institute, Florida Gulf Coast University.

lightweight federated updates. The proposed framework integrates a DeepSet encoder to process variable-sized local observations and a PPO–GAE-based actor–critic network optimized under partial observability. To guarantee safety during deployment, DREAM employs a constraint-aware safety layer that enforces dynamic feasibility and collision avoidance under realistic network delays.

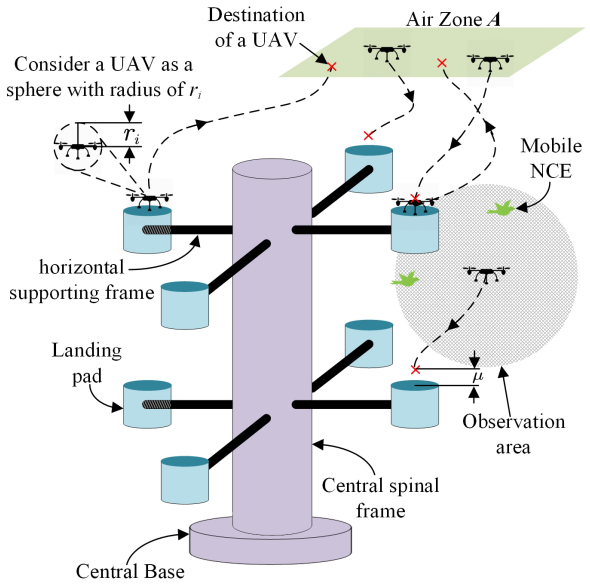

Fig. 1. System model of the portable vertiport environment illustrating UAVs, landing pads, and non-cooperative entities within the 3D operational space.

Simulation results demonstrate that the proposed approach achieves a 98% mission success rate and reduces collision occurrences by over 90% compared to conventional nearest-neighbor baselines. Furthermore, DREAM sustains stable control performance within 100 ms end-to-end latency budgets, confirming its practicality for 5G/6G-enabled aerial edge networks. By coupling reinforcement learning, federated edge intelligence, and communication-aware control, this work contributes to the broader goal of enabling resilient, scalable, and intelligent UAV coordination in next-generation networked environments.

II. System Models and Problem Formulation

In this section, we describe the operational environment and mathematical formulation that form the foundation of the proposed framework. The system model captures how multiple UAVs interact within a constrained vertiport space under edge-assisted coordination, while the problem formulation defines the distributed control objective enabled by multi-access edge computing (MEC) and federated learning principles.

A. System Models

Figure 1 illustrates the portable vertiport used in this study, where multiple UAVs conduct simultaneous takeoff and landing in a shared airspace. The vertiport consists of two stacked layers, each containing four cylindrical landing pads connected by a central support frame. This layout introduces both static obstacles and narrow flight corridors, requiring precise maneuvering and coordination. UAVs may either ascend from a pad toward a destination in the surrounding airspace or descend from above to an available pad.

To reflect real deployment scenarios, the vertiport operates as a micro-edge node equipped with a MEC server that supports local inference and lightweight federated policy aggregation. The MEC server collects model updates from UAVs, performs on-site aggregation, and broadcasts refined parameters to all connected agents. This design allows UAVs to coordinate adaptively without depending on a remote cloud, reducing latency and maintaining scalability across varying network conditions.

Each UAV is abstracted as a sphere of radius r_i, with state variables position $\boldsymbol{p}_i(t)$ and velocity $\boldsymbol{v}_i(t)$. Its motion follows a discrete kinematic model, where velocity is updated as $\boldsymbol{v}_i(t+1) = \boldsymbol{v}_i(t) + \Delta t\, \boldsymbol{a}_i(t)$ and position as $\boldsymbol{p}_i(t+1) = \boldsymbol{p}_i(t) + \frac{\boldsymbol{v}_i(t)+\boldsymbol{v}_i(t+1)}{2}\Delta t$, with $\boldsymbol{a}_i(t)$ denoting the applied acceleration and Δt the sampling step. All *non-cooperative entities* (NCEs) are modeled as cylindrical obstacles, where r_k and l_k denote the radius and height of obstacle k, respectively. This simplified formulation is sufficient for control design and collision-avoidance training.

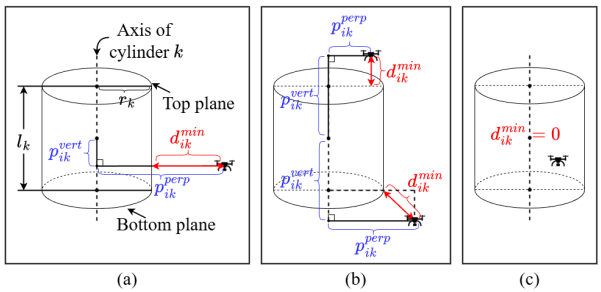

Fig. 2. Collision model of UAVs and obstacles illustrating distance constraints and safe-separation regions in the vertiport environment.

Safety constraints are defined by geometric separation, as shown in **Fig. 2**, which depicts typical collision cases between UAVs and cylindrical obstacles. A collision between UAVs i and j occurs when $|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)| \leq r_i + r_j + \delta$. For UAV–obstacle interactions, the minimum distance between UAV i and obstacle k is:

$$d_{ik}^{\min}(t) = \sqrt{(\max(0, \|p_{ik}^{\text{vert}}\| - l_k/2))^2 + (\max(0, \|p_{ik}^{\text{perp}}\| - r_k))^2}$$

where $\boldsymbol{p}_{ik}^{vert}(t)$ and $\boldsymbol{p}_{ik}^{perp}(t)$ are the axial and perpendicular components of the relative position vector between UAV i and obstacle k. A collision is detected when $d_{ik}^{\min}(t) \leq r_i + \delta$. This formulation compactly represents all possible contact scenarios, including side, top, and corner impacts.

B. Problem Formulation

The objective is to minimize the average mission completion time of all UAVs while ensuring safe and dynamically feasible flight. A mission is considered successful when $\|\boldsymbol{p}_i(T_i) - \boldsymbol{p}_i^{dest}\| \leq \epsilon^{dis}$ and $\|\boldsymbol{v}_i(T_i)\| \leq \epsilon^{spd}$, where T_i denotes the terminal time and $\boldsymbol{p}_i^{dest}$ is the designated destination.

In addition to cooperative UAVs, the airspace contains NCEs such as birds or hobbyist drones that move unpredictably and act as dynamic obstacles. Their presence increases the uncertainty of the environment and is explicitly incorporated into the safety constraints of the optimization.

A centralized optimization would require continuous global communication, which is impractical for real-time deployment. To formalize the control objective, we define Problem $\boldsymbol{P0}$ as:

$$\boldsymbol{P0}: \quad \min_{\{\boldsymbol{a}_i(t)\}} \sum_i T_i \tag{1a}$$

$$\text{s.t.} \quad \|\boldsymbol{p}_i(t) - \boldsymbol{p}_{i'}(t)\| \geq r_i + r_{i'} + \delta, \quad \forall i \neq i', t, \tag{1b}$$

$$d_{ik}^{\min}(t) \geq r_i + \delta, \quad \forall k \in \mathcal{K}^{\text{NCE}}, t, \tag{1c}$$

$$\|\boldsymbol{p}_i(T_i) - \boldsymbol{p}_i^{\text{dest}}\| \leq \epsilon^{\text{dis}}, \quad \|\boldsymbol{v}_i(T_i)\| \leq \epsilon^{\text{spd}}, \tag{1d}$$

$$\|\boldsymbol{a}_i(t)\| \leq a^{\max}, \quad \|\boldsymbol{v}_i(t)\| \leq v^{\max}, \quad \forall t. \tag{1e}$$

Here the sets $\mathcal{K}^{\text{NCE}}$ represent static obstacles (pads and structures) and mobile NCEs, respectively. The formulation minimizes the overall mission duration while enforcing safety and feasibility constraints across both cooperative and non-cooperative elements in the airspace.

Because solving $\boldsymbol{P0}$ requires global information exchange and high computational cost, it is intractable for real-time operation. Therefore, reformulate $\boldsymbol{P0}$ as a distributed sequential decision-making problem using reinforcement learning is provided in Section III. Each UAV determines its actions based on local observations and model parameters periodically updated by the MEC node, which performs federated policy aggregation to improve the shared model without transferring raw data. This hybrid edge–federated architecture ensures low-latency coordination, scalability, and privacy preservation, meeting the requirements of 5G/6G networked UAV systems that integrate communication, computation, and control.

III. DREAM Framework

In this section, we present the design of the proposed **DeepSet-Enhanced Reinforcement Learning for Edge-based Control in Ambiguous and Dynamic Environments (DREAM)** framework by reformulating it as a sequential decision-making task that can be learned efficiently under partial observability and limited communication.

A. Overview of DREAM Framework

Fig. 3 illustrates the overall architecture of the DREAM framework, which follows a centralized-training and decentralized-execution (CTDE) paradigm implemented within an edge–federated environment. The training phase takes place either offline or at the vertiport MEC server, where policy parameters collected from UAVs are aggregated using federated averaging to update a shared global model. Once training is complete, each UAV executes its local policy autonomously, requiring only lightweight synchronization with the MEC node during operation.

The DREAM architecture integrates the environment, learning, and execution processes into a unified system. The environment includes multiple UAVs and NCEs operating within the vertiport airspace, where interactions generate dynamic states and rewards. The training process occurs at the MEC node under the CTDE setting, allowing the UAVs to share policy gradients while maintaining data privacy. The execution process runs on each UAV, where the learned policy enables local decision-making in real time with minimal latency. This layered structure ensures both scalability and responsiveness for large-scale vertiport operations.

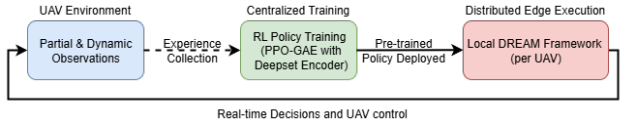

Fig. 3. Workflow of DeepSet-enhanced Reinforcement learning for Edge-based control in Ambiguous and dynamic environMents (DREAM) Framework: a centralized training with distributed edge execution.

B. MDP Formulation and DeepSet Representation

To solve $\boldsymbol{P0}$, the DREAM framework reformulates the control problem as a partially observable Markov decision process (POMDP) for each UAV i. The process is represented by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ denotes the transition model, R is the reward function, and γ is the discount factor.

Each UAV observes a local state $o_i(t) \subseteq \mathcal{S}$, consisting of its position, velocity, and relative information of neighboring UAVs and surrounding entities:

$$o_i(t) = \{p_i(t), v_i(t)\} \cup \mathcal{I}_i(t) \cup \mathcal{K}^{\text{NCE}}, \tag{2}$$

where $\mathcal{I}_i(t)$ represents neighboring UAVs within sensing range, and $\mathcal{K}^{\text{NCE}}$ represents all the observed information of the obstacles. Each UAV executes an acceleration command $a_i(t) \in \mathcal{A}$ based on its observation.

The immediate reward $R_i(t)$ encourages safe, efficient, and feasible operation and is defined as

$$R_i(t) = \begin{cases} +100, & \text{successful mission (landing or takeoff),} \\ -100, & \text{collision or boundary violation,} \\ -1, & \text{if } \|a_i(t)\| > a_{\max} \text{ or } \|v_i(t)\| > v_{\max}, \\ -2, & \text{time-step penalty,} \\ +1, & \text{progress toward destination} \end{cases} \tag{3}$$

Because the number of neighboring entities changes dynamically, the DREAM framework adopts a *DeepSet* [14] encoder to obtain a fixed-length, permutation-invariant latent representation. For each UAV i, the relational feature between itself and entity k is defined as

$$z_{ik}(t) = [p_i(t), v_i(t), o_k(t)], \tag{4}$$

where $o_k(t)$ represents the observed state of entity k. Each feature $z_{ik}(t)$ is embedded using a shared multilayer perceptron (MLP) $D(\cdot)$, and the encoded features are aggregated by a symmetric pooling operator $P(\cdot)$, such as summation or mean:

$$s_i(t) = P\big(\{D(z_{ik}(t))\}_{k \in \mathcal{K}_i(t)}\big). \tag{5}$$

The resulting latent vector $s_i(t)$ serves as the compact input to the policy and value networks, providing robustness to varying neighborhood sizes.

C. Policy Learning and Safety-Constrained Optimization

The latent state $s_i(t)$ is processed by an actor–critic network trained using *Proximal Policy Optimization (PPO)* [15] with *Generalized Advantage Estimation (GAE)* [16] . The actor, parameterized by θ, outputs the stochastic policy $\pi_\theta(a_i|s_i)$, while the critic, parameterized by ϕ, estimates the value function $V_\phi(s_i)$. The advantage estimate is given by

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \big[r_{t+l} + \gamma V_\phi(s_{t+l+1}) - V_\phi(s_{t+l}) \big], \quad (6)$$

where λ balances bias and variance in the estimation.

The PPO objective is defined as

$$L_{\text{PPO}}(\theta) = \mathbb{E}_t \Big[\min \Big(\rho_t(\theta)\hat{A}_t, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \Big) \Big], \quad (7)$$

where $\rho_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ is the policy ratio, and ϵ is the clipping threshold. The value loss is defined as

$$L_V(\phi) = \mathbb{E}_t[(V_\phi(s_t) - R_t)^2], \quad (8)$$

and the overall objective is

$$L(\theta, \phi) = L_{\text{PPO}}(\theta) - c_v L_V(\phi) + c_e H(\pi_\theta), \quad (9)$$

where $H(\pi_\theta)$ denotes the policy entropy term, and c_v and c_e are weighting coefficients.

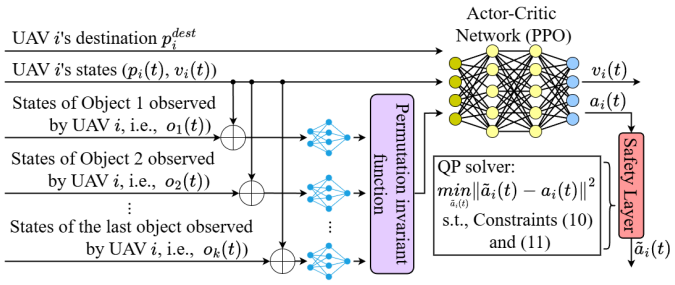

Fig. 4. Architecture of the DREAM framework showing the DeepSet encoder, PPO actor–critic network, and safety layer for UAV control.

To guarantee safe operation, DREAM integrates a *safety projection layer* that corrects unsafe actions before execution. The safety layer solves a quadratic program (QP) at each step:

$$\min_{\tilde{a}_i(t)} \|\tilde{a}_i(t) - a_i(t)\|^2 \quad \text{s.t. } \|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\| \geq r_i + r_j + \delta, \quad (10)$$

$$d_{ik}^{\min}(t) \geq r_i + \delta, \quad \|\tilde{a}_i(t)\| \leq a_{\max}. \quad (11)$$

The adjusted command $\tilde{a}_i(t)$ satisfies all physical and safety constraints derived from $\boldsymbol{P0}$. If the solver exceeds the 100 ms latency budget, the unmodified action $a_i(t)$ is applied to preserve responsiveness. During training, this layer is disabled so that the policy learns boundary-aware behavior through exploration.

Fig. 4 further illustrates the training workflow and convergence performance of the proposed approach. The PPO–GAE optimization achieves stable reward growth within approximately 2×10^5 iterations, while collision rates decline sharply during the early stages of learning. The figure also highlights the integration between the federated PPO training loop and the QP-based safety projection, confirming that the overall control pipeline operates within the 100 ms latency constraint. Overall, DREAM combines PPO-based policy optimization, DeepSet-based state representation, and QP-based safety correction into a unified control framework. This design ensures stable learning and real-time feasibility for distributed UAV coordination in a federated MEC environment. All source code developed for this study is publicly available in [17] for reproducibility and further research.

IV. Experiments and Evaluation Discussion

A. Evaluation Setup and Metrics Discussion

We evaluate the proposed DREAM framework through Python-based simulations of UAV landing and takeoff at a portable vertiport under dynamic airspace conditions. The simulation environment represents a three-dimensional space of $10 \times 10 \times 5\text{m}^3$, spanning $[-5, 5]$m in x and y and $[0, 5]$m in z, with the vertiport positioned at the origin. The vertiport contains two stacked layers of landing pads, one at 0.4 m and another at 1.3 m, each composed of 4 cylindrical pads (r=0.2m, h=0.2m) arranged in a cross pattern. The pads are supported by horizontal frames (l=2m, r=0.1m) connected to a central spine (h=1.6m, r=0.1m) and anchored to a circular base (r=0.4m). An entry/exit airspace region $\boldsymbol{A}$ is defined at (0, 0, 4.5)m, covering an area of 3×3m, which serves as the designated zone for UAV arrivals and departures.

Each UAV is modeled as a rigid sphere with a radius of 0.1 m. At the start of each episode, UAVs are randomly assigned to either landing or takeoff missions. Landing UAVs depart from $\boldsymbol{p}_i^{src} \in \boldsymbol{A}$ and target destinations $\boldsymbol{p}_i^{dest}$ located 0.5 m above a landing pad, whereas takeoff UAVs start 0.5 m above a pad and navigate toward $\boldsymbol{p}_i^{dest} \in \boldsymbol{A}$. To emulate realistic disturbances, mobile non-cooperative entities (NCEs), such as birds or small drones, are included as spherical obstacles with a radius of 0.25 m. These NCEs follow a random waypoint model with an average speed of 0.1 m/s, continuously selecting new destinations to introduce dynamic environmental uncertainty.

The DREAM framework uses two main neural components for policy learning and distributed control. The first is a DeepSet encoder that processes a variable number of observed entities. For each UAV i, every observation $\boldsymbol{o}_k(t)$ is concatenated with $(\boldsymbol{p}_i(t), \boldsymbol{v}_i(t))$ and passed through a shared MLP with 128 and 256 ReLU units and a 10-dimensional output. Mean pooling generates a fixed-size, permutation-invariant embedding. The second component is a PPO–GAE policy network that takes $\boldsymbol{p}_i^{dest}$, the UAV state, and the DeepSet embedding as inputs, using the same 128/256-layer architecture. Training alternates actor–critic updates with clipped PPO objectives and GAE, with full hyperparameters listed in **Table I**. A 100 ms step duration is used to match planned real-world deployment on the Crazyflie quadcopter, which reliably operates at 10 Hz [18]. The pretrained model generates each action within 9–12 ms, while the safety layer uses the `qpsolvers` 4.8.2 library with a 20 ms timeout; if no feasible solution is found, the original action is applied to maintain loop timing. Communication delay is negligible in current simulation, and measured Crazyflie radio latency in real-world

Table I. Simulation and Training Parameters Settings

Simulation Basic Setting	Value	*Training Environment Setting*	Value
Time slot Δt	100 ms	Optimizer	Adam
Max. episode length $T^{\max}$	500 steps	Discount factor γ	0.99
UAV max. velocity $v^{\max}$	1 m/s	Decay rate (β_1/β_2)	0.9, 0.999
UAV max. acceleration $a^{\max}$	1 m/s^2	Batch size	64
Environment time limit	50 s	PPO clipping factor ϵ	0.2
Number of UAVs	4	Entropy coefficient c_e	0.01

environment remains below 100 ms, supporting the practicality of the chosen timestep.

As a baseline, we compare against the *Nearest Neighbors Observation (NNO)* method [19], where each UAV selects the k closest entities and discards the rest, with zero padding when fewer are present. Unlike the permutation-invariant DeepSet encoder, NNO depends on fixed ordering and may omit important environmental information.

In terms of the simulation environment setup, we set each episode to include 6 UAVs and 4 mobile NCEs operating concurrently within the defined airspace. Simulations are executed on a workstation equipped with an NVIDIA GeForce RTX 4060 (12 GB) GPU and an Intel i7-12700 CPU. Evaluation focuses on three key performance aspects: mission success rate, collision rate, and average mission duration. The mission success rate quantifies the proportion of UAVs that successfully complete their assigned takeoff or landing missions, while the collision rate measures the frequency of UAV or obstacle collisions within an episode. The average mission duration captures the overall efficiency of the control policy by computing the mean time required for all successful missions. Together, these metrics comprehensively assess the effectiveness, safety, and responsiveness of the proposed DREAM framework under dynamic vertiport conditions.

B. Experiment Results and Observation

Fig. 5. Training convergence of the DREAM framework showing the mean episodic reward over time.

Training Phase Experiment Results: The training performance of the proposed DREAM framework is shown in **Fig. 5**, where the mean episodic reward is plotted against training timesteps. During the initial exploration phase (*red region*), the mean reward sharply drops to about -3×10^4 due to the UAVs frequently leaving the flight boundary during the initial exploration phase. Although these events do not end the episode, each incurs a strong negative penalty (-100), resulting in low cumulative rewards. As the PPO–GAE policy improves (*gray region*), UAVs gradually learn feasible trajectories that satisfy the dynamic and spatial constraints of the vertiport environment, leading to a steady increase in performance.

Table II. Performance Comparison of DREAM and Baseline Algorithms in Vertiport Takeoff and Landing Scenarios (100 Episodes, 600 UAVs in total)

Algorithm	NNO (baseline)	DREAM w/o safety layer	DREAM w/ safety layer
Success (%)	79.5	91.5	**98.0**
Arrival (s)	12.3 $\pm$ 1.2	**11.8 $\pm$ 1.1**	15.2 $\pm$ 1.5
Median (s)	12.3 $\pm$ 2	**11.8 $\pm$1.9**	15.2 $\pm$2.35
Δ Success (pp)	–	+12.0	**+18.5**
Δ Arrival (s)	–	-0.5	**+2.9**

After approximately 2.0×10^7 timesteps (*green region*), the learning curve stabilizes around a mean reward of 500, approaching the theoretical upper bound of 600 for six UAVs with minor time and acceleration penalties. Minor oscillations appear during later training due to exploratory updates allowed by the PPO clipping mechanism, which momentarily reduce performance but enhance long-term stability. Overall, the results in Fig. 5 confirm that DREAM achieves stable convergence, efficient coordination, and strong resilience against dynamic perturbations from non-cooperative entities.

Testing Phase Experiment Results: After achieving convergence in the training phase, the learned DREAM policy was evaluated in the testing environment to assess its generalization, safety, and coordination performance under vertiport operations. The quantitative results of this evaluation are summarized in **Table II**. Two primary metrics are considered: mission success rate and average arrival time, computed only from successful missions for consistency.

The baseline NNO algorithm achieves a success rate of 79.5% and an average arrival time of 12.3 s. In comparison, DREAM without the safety layer improves the success rate to 91.5%, reflecting the benefit of the DeepSet-based state encoding and PPO–GAE policy structure in enhancing spatial awareness and coordination among UAVs. The average arrival time also decreases slightly to 11.8 s, indicating faster trajectory optimization under decentralized control.

When the safety layer is activated, DREAM achieves the highest success rate of 98.0%, an 18.5 pp improvement over the baseline. Although the mean arrival time increases to 15.2 s, this trade-off arises from conservative adjustments imposed by the safety layer to ensure collision-free maneuvers. The overall results confirm that DREAM effectively balances mission efficiency and operational safety, achieving optimal reliability in complex multi-agent vertiport environments.

The reported standard deviations capture variability across missions, while the median and $p5$–$p95$ ranges offer a more robust view of the distribution, indicating that the majority of UAVs complete their missions within a relatively narrow time window even in dynamic environments.

Following the quantitative performance comparison in Table II, **Fig. 6** further analyzes the safety performance of each algorithm through detailed collision statistics across 100 test episodes. The baseline NNO method exhibits the highest total collision rate of 20.5%, primarily due to UAV–UAV interactions near the vertiport. DREAM without the safety layer reduces collisions to 8.5%, indicating that the DeepSet-based perception and PPO–GAE policy enhance cooperative awareness and motion coordination. When the safety layer is enabled, the total collision rate drops to only 2.0%, eliminating

impacts with landing platforms and greatly reducing UAV-to-UAV and UAV-to-Mobile NCE collisions. These results demonstrate that the safety-aware mechanism in DREAM substantially improves reliability and ensures safe multi-UAV operations in constrained vertiport environments.

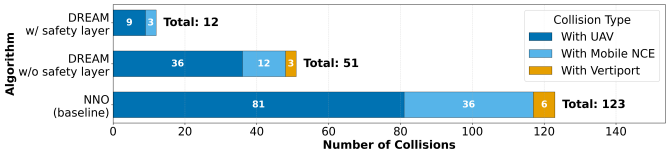

Fig. 6. Collision statistics comparison among NNO baseline, DREAM without safety, and DREAM with safety layer under vertiport operations.

Building on the safety evaluation, **Fig. 7** examines the scalability and robustness of DREAM under varying UAV and NCE densities. As the number of UAVs increases from 4 to 8, DREAM with the safety layer sustains a success rate above 96.5%, while the baseline NNO drops to 67.0% and DREAM without safety decreases to 84.6%. A similar trend appears as the number of NCEs rises, where DREAM consistently maintains success rates above 95.0%, demonstrating strong resilience to environmental uncertainty. These results confirm that DREAM effectively balances safety and scalability, maintaining coordination efficiency and mission success in dense and dynamic multi-agent airspace conditions.

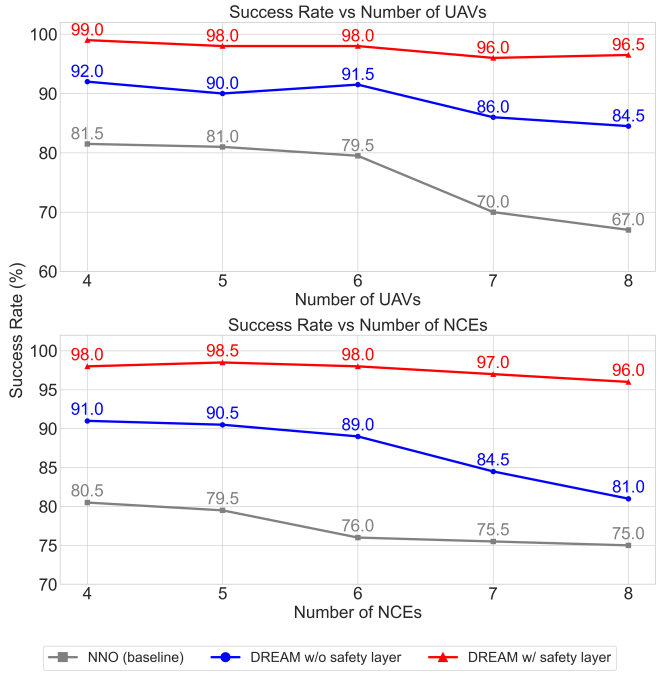

Fig. 7. Scalability and robustness analysis of DREAM compared with NNO baseline under varying UAV and NCE densities.

V. Conclusions and Future Work

This paper presented *DREAM*, a DeepSet-enhanced reinforcement learning framework for coordinating UAV takeoff and landing at portable vertiports. By combining permutation-invariant state encoding, PPO–GAE optimization, and a safety-aware control layer within an edge-enabled design, DREAM achieved reliable, collision-free performance under dynamic airspace conditions. Simulations showed notably higher success rates and improved safety over the baseline, confirming its effectiveness for real-time multi-UAV operations. Future work will enhance safety and scalability by incorporating control barrier functions, testing under more realistic network conditions, and leveraging federated learning for stronger generalization. We also plan to explore transformer-based encoders and benchmark against stronger baselines to further improve cooperative performance in dense UAV environments.

References

[1] Jaber Almutairi et al. Delay-optimal task offloading for uav-enabled edge-cloud computing systems. *IEEE Access*, 10:51575–51586, 2022.

[2] Zhuoyi Bai, Yifan Lin, Yang Cao, and Wei Wang. Delay-aware cooperative task offloading for multi-uav enabled edge-cloud computing. *IEEE Transactions on Mobile Computing*, 23(2):1034–1049, 2024.

[3] Shumaila Javaid et al. Communication and control in collaborative uavs: Recent advances and future trends. *IEEE Transactions on Intelligent Transportation Systems*, 24(6):5719–5739, 2023.

[4] Taiyuan Gong, Li Zhu, F. Richard Yu, and Tao Tang. Edge intelligence in intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 24(9):8919–8944, 2023.

[5] Lingxia Mu, Yichi Yang, Ban Wang, Youmin Zhang, Nan Feng, and Xuesong Xie. Edge computing-based real-time forest fire detection using uav thermal and color images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 18:6760–6771, 2025.

[6] Yuhao Zhang, Zhufang Kuang, Yanyan Feng, and Fen Hou. Task offloading and trajectory optimization for secure communications in dynamic user multi-uav mec systems. *IEEE Transactions on Mobile Computing*, 23(12):14427–14440, 2024.

[7] Lu Sun, Liangtian Wan, Jiashuai Wang, Lin Lin, and Mitsuo Gen. Joint resource scheduling for uav-enabled mobile edge computing system in internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 24(12):15624–15632, 2023.

[8] Tiying Gao, Dwight Goins, Christian Ballotti, et al. Learning-based uav swarm video analytics orchestration in disaster response management. *SN Computer Science*, 6:537, 2025.

[9] Kevin Kostage et al. Federated learning-enabled network incident anomaly detection optimization for drone swarms. ICDCN '25, New York, NY, USA, 2025. Association for Computing Machinery.

[10] Hangxing. Wu, Hui Ye, Wentao Xue, and Xiaofei Yang. Improved reinforcement learning using stability augmentation with application to quadrotor attitude control. *IEEE Access*, 10:67590–67604, 2022.

[11] Zijiang Yan, Wael Jaafar, Bassant Selim, and Hina Tabassum. Multi-uav speed control with collision avoidance and handover-aware cell association: Drl with action branching. In *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, pages 5067–5072, 2023.

[12] Yonatan Melese Worku et al. Deep rl for uav energy and coverage optimization in 6g-based iot remote sensing networks. In *2025 IEEE Aerospace Conference*, pages 1–14, 2025.

[13] Yahao Ding et al. Distributed machine learning for uav swarms: Computing, sensing, and semantics. *IEEE Internet of Things Journal*, 11(5):7447–7473, 2024.

[14] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets, 2018.

[15] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[16] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018.

[17] Zhirun Li, Tiying Gao, and Chengyi Qu. Deep-Set Edge RL Code. https://github.com/SECNetLabUNM/UAV_Landing_Takeoff_Control, Dec. 2025.

[18] Jacek Michalski, Marek Retinger, Piotr Kozierski, and Wojciech Giernacki. Position control of crazyflie 2.1 quadrotor uav based on active disturbance rejection control. In *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1106–1113, 2023.

[19] Liangkun Yu, Zhirun Li, Nirwan Ansari, and Xiang Sun. Hybrid transformer based multi-agent reinforcement learning for multiple unpiloted aerial vehicle coordination in air corridors. *IEEE Transactions on Mobile Computing*, 24(6):5482–5495, 2025.