Journal of Field Robotics

WILEY

RESEARCH ARTICLE OPEN ACCESS

Deep Reinforcement Learning Based Autonomous Decision-Making for Cooperative Uncrewed Aerial Vehicles: A Search and Rescue Real World Application

Thomas Hickling | Maxwell Hogan | Abdulla Tammam | Nabil Aouf

School of Science and Technology, City St George's University of London, London, UK

Correspondence: Thomas Hickling (tom.hickling@city.ac.uk)

Received: 27 February 2025 | **Revised:** 1 April 2026 | **Accepted:** 13 May 2026

Keywords: cooperative UAVs | DRL | DRL task allocation | GNSS-denied | LiDAR-SLAM | search and rescue | sim-to-real

ABSTRACT

This paper presents the first end-to-end framework that combines guidance, navigation, and centralized task allocation for multiple UAVs performing autonomous search-and-rescue (SAR) in GNSS-denied indoor environments. A twin delayed deep deterministic policy gradient controller is trained with an artificial potential field (APF) reward that blends attractive and repulsive potentials with continuous control, accelerating convergence and yielding smoother, safer trajectories than distance-only baselines. Collaborative mission assignment is solved by a deep Graph Attention Network that, at each decision step, reasons over the drone-task graph to produce near-optimal allocations with negligible onboard compute. To arrest the notorious Z-drift of indoor LiDAR-SLAM, we fuse depth-camera altimetry with IMU vertical velocity in a lightweight complementary filter, giving centimeter-level altitude stability without external beacons. The resulting system was deployed on two 1 m-class quad-rotors and flight-tested in a cluttered, multi-level disaster mock-up designed for the NATO-Sapience Autonomous Cooperative Drone Competition. Compared with prior DRL guidance that remains largely in simulation, our framework demonstrates an ability to navigate complex indoor environments, securing first place in the 2024 event. These results demonstrate that APF-shaped DRL and GAT-driven cooperation can translate to reliable real-world SAR operations.

1 | Introduction

Uncrewed aerial vehicles (UAVs) are increasingly being considered for search and rescue (SAR) missions because they can access hazardous or confined areas while providing rapid situational awareness (Scoles 2024). Their use is particularly attractive in Global Navigation Satellite System (GNSS)-denied environments, such as indoor disaster sites, where human access may be dangerous and conventional autonomous navigation methods are less reliable. However, operation in such environments remains challenging because UAVs must localize, avoid obstacles, navigate in cluttered spaces, and coordinate with other agents despite limited prior knowledge of the scene.

Deep reinforcement learning (DRL) has shown considerable promise for autonomous decision-making in complex robotic tasks, including UAV navigation, cooperative search, and multiagent coordination (Azar et al. 2021; Shen et al. 2023). Nevertheless, transferring DRL-based UAV systems from simulation to real-world GNSS-denied environments remains difficult. In cooperative SAR settings, this challenge extends beyond low-level guidance to include robust localization, multi-UAV task allocation, and reliable deployment on physical platforms.

This paper presents the development and real-world deployment of a cooperative multi-UAV system for GNSS-denied

FIGURE 1 | The arena built for the Sapience competition as seen by the eight cameras used for monitoring the UAVs. [Color figure can be viewed at wileyonlinelibrary.com]

indoor SAR scenarios. The work was developed within the NATO-funded SAPIENCE program and evaluated in the Autonomous Cooperative Drone Competition (NATO 2024), which focuses on autonomous operation in realistic disaster-response environments. This focus on cooperative autonomy in realistic mission settings is aligned with recent developments in multi-UAV search and task-allocation research (Ghauri et al. 2024; Song et al. 2025). The competition tasks included exploration, mapping, object localization, and aid-delivery-relevant behaviors in a large indoor arena designed to emulate a hazardous SAR setting, as shown in Figures 1 and 2.

The proposed system integrates three core capabilities. First, each UAV employs a DRL-based guidance policy trained in simulation using twin delayed deep deterministic policy gradient (TD3) in the AirSim environment (Fujimoto et al. 2018; Shah et al. 2017). Second, cooperative task allocation is performed using a graph-based DRL method that enables the two UAVs to coordinate area coverage and adapt task decisions during operation. Third, localization is achieved using LiDAR-SLAM, with additional compensation for altitude drift encountered in narrow indoor corridors. This is important because purely vision-based localization approaches may perform poorly in environments with limited visual texture, making LiDAR-based localization a more suitable option in this setting (Tourani et al. 2022; Grisetti et al. 2005).

The aim of this work is to demonstrate that cooperative autonomous UAV operation in challenging indoor SAR environments can be achieved through the integration of robust

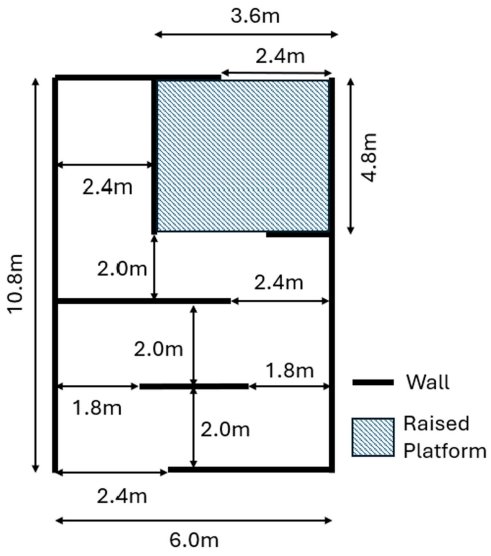

FIGURE 2 | The floor plan for the constructed building. [Color figure can be viewed at wileyonlinelibrary.com]

localization, DRL-based guidance, and learned task allocation. Rather than treating these elements independently, the paper focuses on the practical deployment of an integrated multi-UAV system and the lessons arising from its transition from simulation to real-world operation. Accordingly, this work is positioned within recent efforts to combine learned guidance, cooperative search, and task-level coordination in multi-UAV systems for challenging operational environments (Shen et al. 2023; Song et al. 2025).

Contributions. The main contributions of this paper are as follows:

1. *Real-world deployment and evaluation of a cooperative DRL-guided SAR UAV system*. We present the end-to-end deployment of a two-UAV autonomous system in a large and structurally complex indoor disaster mock-up, with emphasis on the practical challenges of simulation-to-real transfer for cooperative aerial robotics.
2. *An APF-referenced action-alignment reward for continuous UAV guidance*. We introduce a reward design in which an artificial potential field (APF)-inspired scheme generates a reference action $a_t^\star$ in the same control space as the commanded UAV velocity, and the agent is rewarded for aligning its action a_t with this reference. The reward incorporates obstacle-proximity mode switching, clipping for training stability, and terminal overrides for collision and goal completion. This formulation enabled stable training and successful real-world deployment in a cluttered indoor environment in which a standard distance-based reward was insufficient under the same training conditions.
3. *A graph-attention-based DRL task allocation method for cooperative search*. We formulate multi-UAV coordination as a graph-based decision problem and train a graph attention network (GAT) actor-critic model to allocate tasks centrally while the individual UAVs execute decentralized low-level guidance policies.
4. *Altitude-drift mitigation for indoor LiDAR-SLAM*. To reduce vertical drift in indoor LiDAR-SLAM, we combine depth-camera-derived altitude estimates with IMU vertical velocity in a lightweight complementary filtering framework, producing a more stable global altitude estimate during flight.

2 | Related Work

2.1 | DRL for Autonomous Guidance and Obstacle Avoidance

DRL has been widely explored for autonomous UAV guidance and obstacle avoidance, particularly in environments where model-based navigation is difficult to formulate robustly. Existing methods commonly operate in either discrete action spaces, in which the UAV selects from a finite set of manoeuvres, or continuous action spaces, in which control commands are produced directly. Discrete-action approaches are often associated with DQN-like methods, whereas continuous-control problems are more commonly addressed using algorithms such as DDPG, PPO, and TD3 (Azar et al. 2021; Zhu et al. 2024; Hu et al. 2022; Mnih et al. 2015; Lillicrap et al. 2015; Schulman et al. 2017; Fujimoto et al. 2018; Hickling et al. 2023; He et al. 2021).

A central difficulty in DRL-based UAV guidance is sample inefficiency. Because learning is driven by interactions and reward feedback rather than labeled supervision, training can require large numbers of episodes, particularly in cluttered environments. For this reason, several studies have incorporated prior structure or demonstrations to improve early learning. Representative examples include learning from demonstrations in both discrete and continuous settings (Hester et al. 2018; Vecerik et al. 2017).

APFs are one of the most common sources of such prior structure in UAV navigation. In the recent literature, APF and DRL are typically combined in three ways: by using APF to guide exploration during training, by embedding potential-inspired terms in the reward, or by applying APF-derived bias or safety logic during execution. In the first category, APF-derived reference actions can act as a curriculum-like scaffold during early training, improving stability and reducing catastrophic exploration (Zhang et al. 2024; Hickling et al. 2023). In the second, APF-inspired attractive and repulsive terms are incorporated directly into the reward in order to densify feedback beyond sparse terminal signals (Li and Wu 2020; Dong et al. 2023). In the third, APF is used to modify or constrain execution-time behavior, for example through safety shields or hybrid planner-policy schemes (Li et al. 2023; Ren et al. 2025). These approaches have shown promise, but they can remain sensitive to reward design, discretised control choices, or discrepancies between the learned policy and the behavior actually executed on the robot.

Recent work has also continued to examine DRL for cooperative search and multi-UAV decision-making. For example, Shen et al. (Shen et al. 2023) used reinforcement learning with a digital-twin-driven training framework for cooperative multi-UAV search, while Liu et al. (Liu, Li, et al. 2024) considered reinforcement-learning-based cooperative search for moving targets in three-dimensional environments. These studies demonstrate continued progress in learned multi-UAV autonomy, but remain focused primarily on the search or planning problem rather than the real-world deployment of an integrated indoor guidance system.

The present work is most closely related to APF-guided and APF-shaped DRL methods, but differs in both formulation and deployment focus. Rather than using APF only as an external action source or safety override, the proposed method uses APF to generate a reference action in the same continuous control space as the deployed commands, and rewards the policy for alignment with this reference. This provides dense feedback expressed directly in the commanded variables used for flight. In addition, unlike much of the prior literature, the approach is evaluated through real-world deployment in a cluttered, GNSS-denied indoor environment using full-scale cooperative UAVs.

2.2 | DRL for Graph-Based Task Allocation

Task allocation is a central problem in cooperative multi-UAV systems, particularly in search and rescue scenarios where mission priorities, robot states, and environmental conditions may change during operation. Classical approaches have included rule-based, auction-based, and heuristic optimization methods in both centralized and decentralized forms (Sargolzaei et al. 2020; Zhao et al. 2015; Cui et al. 2019).

Although such methods can be effective in structured settings, they may struggle to scale or adapt when the environment is uncertain and highly dynamic.

More recent work has therefore examined machine learning and DRL for multi-UAV task allocation and coordination. A recent review by Ghauri et al. (Ghauri et al. 2024) highlights the breadth of contemporary task-allocation strategies for SAR and emphasizes ongoing challenges in balancing efficiency, scalability, and robustness. Comparative work has also revisited the trade-off between centralized and distributed formulations; for example, Song et al. (Song et al. 2025) provide a unified evaluation framework for these alternatives in multi-UAV task allocation. In parallel, path-planning and search studies continue to emphasize the importance of coordination among multiple UAVs in collaborative area-coverage missions (Yu and Lee 2023). Together, these works show that multi-UAV cooperation remains an active and evolving research direction.

Graph-based representations have become particularly attractive for this problem because they provide a natural way to encode agents, tasks, and their relationships. Nodes can represent UAVs, tasks, or waypoints, while edges can encode communication links, traversal costs, or task dependencies. Within this setting, graph neural networks enable structured reasoning over relational information, and graph attention networks (GATs) extend this by learning the relative importance of neighboring nodes and edges (Velickovic et al. 2017; Du et al. 2024; Liu, Huang, et al. 2024; Shao et al. 2021). This is especially relevant to task allocation, where different tasks or agent-task relationships may vary in urgency or cost over time.

Prior graph-based multiagent reinforcement learning has shown the value of attention mechanisms for coordination. For example, Ryu et al. (Ryu et al. 2020) proposed a hierarchical graph-attention actor-critic architecture for scalable multi-agent learning in simulated mixed cooperative-competitive environments. However, such work is typically evaluated in abstract benchmarks rather than in fielded aerial systems. Similarly, distributed multi-robot allocation schemes such as those studied by Alshaboti and Baroudi (Alshaboti and Baroudi 2021) provide useful comparisons with auction- and rule-based strategies, but do not address the combination of graph-based learning, real UAV deployment, and GNSS-denied operation considered here.

Against this background, the present paper adopts a graph-attention-based DRL allocator in a centralized task-assignment setting for cooperative indoor UAV search. The emphasis is not only on allocator design, but also on integration with learned low-level guidance and real-world deployment constraints.

2.3 | LiDAR-SLAM Adaptation for Indoor Navigation

Reliable localization is essential for autonomous UAV operation in GNSS-denied environments. In indoor settings, purely vision-based methods may be unreliable when scenes contain limited texture, poor lighting, or repetitive structure (Tourani et al. 2022). LiDAR-SLAM therefore provides an attractive alternative because it can supply geometry-driven localization and mapping independently of visual appearance (Grisetti et al. 2005).

However, LiDAR-SLAM in indoor corridors is known to suffer from degeneracy caused by long, narrow, and weakly distinctive geometry. In such environments, the available geometric constraints may be insufficient to maintain stable estimation in all axes, leading in practice to drift, particularly in altitude. Prior work has addressed corridor-like sensing limitations by augmenting LiDAR with additional sensing modalities. For example, Zhang et al. (Zhang and Maher Atia 2020) combined radar and LiDAR in order to provide richer environmental constraints in feature-poor settings.

In contrast, this paper does not seek to redesign the full SLAM back-end. Instead, it addresses the practical altitude-drift issue observed during indoor operation by fusing depth-camera-derived altitude information with inertial vertical velocity in a lightweight complementary filtering framework. This provides a pragmatic adaptation for deployment in the competition environment while preserving the mapping and localization benefits of the LiDAR-SLAM pipeline.

3 | DRL-Based UAV Guidance

Autonomous UAV guidance has traditionally been addressed using classical planning and reactive methods such as Dijkstra search, A*, and APFs. While these methods can provide effective obstacle avoidance and path generation in structured settings, they may become difficult to tune for cluttered and uncertain indoor environments. DRL offers an alternative by learning control policies directly from interaction, enabling the agent to map sensory observations to control commands without requiring an explicit environment model.

In this work, UAV guidance is learned using a continuous-control DRL formulation. At each time step, the policy receives the current observation of the environment and outputs velocity-level control commands. Compared with discrete manoeuvre selection, continuous control is better suited to indoor flight because it allows smoother and more precise motion in confined spaces.

3.1 | Twin Delayed Deep Deterministic Policy Gradient

The guidance policy is trained using Twin Delayed Deep Deterministic Policy Gradient (TD3) (Fujimoto et al. 2018), an off-policy actor-critic algorithm designed for continuous control. TD3 improves upon Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al. 2015) through three main modifications. First, clipped double Q-learning uses the minimum of the two critic estimates when forming the target value, which reduces overestimation bias. Second, the actor is updated less frequently than the critics, allowing the value estimates to stabilize before policy updates are applied. Third, target-policy smoothing perturbs the target action with small noise when

computing target Q-values, which improves robustness to sharp value peaks.

These properties make TD3 suitable for UAV guidance in cluttered environments, where stable learning and smooth control outputs are required. During training, transitions are stored in a replay buffer and sampled in mini-batches for critic and actor updates.

3.2 | APF-Referenced Reward Formulation

A key aspect of the proposed guidance method is the reward design. Rather than using only sparse terminal rewards or simple distance-based shaping, the proposed method incorporates an APF as a source of reference behavior. APFs are widely used in robotics to combine attraction toward a goal with repulsion from nearby obstacles (Khatib 1986). The total potential field is written as

$$U(q) = U_{\text{att}}(q) + U_{\text{rep}}(q), \tag{1}$$

where q denotes the UAV position, U_{att} is the attractive potential associated with the goal, and U_{rep} is the repulsive potential associated with obstacles. The corresponding guidance direction is obtained from the negative gradient of the potential,

$$F(q) = -\nabla U(q). \tag{2}$$

In the proposed formulation, the APF is not used as an external controller that directly overrides the learned policy. Instead, it is used to generate a reference action $a_t^\star$ in the same continuous control space as the UAV command. The reward then encourages agreement between the executed action a_t and this APF-derived reference, while additional terminal terms are used to reward goal completion and penalize collision. This provides dense and physically meaningful training feedback while preserving the flexibility of the learned policy.

3.3 | Benefits of the APF-Referenced Design

This reward formulation has three practical advantages. First, it improves learning efficiency by providing informative step-wise guidance during training. Second, it encourages smoother obstacle-avoidance behavior than sparse or purely distance-based rewards. Third, it combines the inductive bias of a classical navigation method with the adaptability of DRL, which is beneficial for deployment in cluttered indoor environments where reliable sim-to-real transfer is required.

4 | Graph-Based Task Allocation

Task allocation determines how mission objectives are distributed among multiple UAVs so that the team can operate efficiently and avoid redundant effort. In indoor search and rescue settings, this requires the allocator to account for changing task states, UAV locations, and mission progress during operation. Classical methods such as rule-based assignment, auctions, and heuristic optimization can be effective in structured settings, but they may be difficult to adapt to dynamic environments with evolving task dependencies.

In this work, task allocation is formulated as a decision-making problem and solved using a graph-based DRL approach. The graph representation captures the relationships between UAVs and candidate tasks, allowing the allocator to reason over both node attributes and pairwise costs in a structured manner.

4.1 | Graph Representation

The allocation problem is represented as a graph in which nodes encode UAVs and tasks, and edges encode the relationships between them, such as traversal cost, assignment suitability, or other pairwise information relevant to coordination. This structure is appropriate for cooperative multi-UAV search because the importance of a task depends not only on the task itself, but also on its relationship to the current states of the available UAVs and to the wider mission configuration.

4.2 | Graph Attention Network Allocator

The allocator uses a Graph Attention Network (GAT) (Velickovic et al. 2017) within an actor-critic DRL framework. In contrast to graph convolutional networks (GCNs), which aggregate neighboring features with fixed normalized weights, GATs learn attention coefficients that determine the relative importance of neighboring nodes during message passing. For node i, the attention coefficient associated with neighbor j is computed as

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}(\mathbf{a}^T\left[W\boldsymbol{h}_i \parallel W\boldsymbol{h}_j\right])\right)}{\sum_{k\in\mathcal{N}(i)}\exp\left(\text{LeakyReLU}(\mathbf{a}^T\left[W\boldsymbol{h}_i \parallel W\boldsymbol{h}_k\right])\right)}, \tag{3}$$

where $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ are node features, W is a shared linear transformation, $\boldsymbol{a}$ is a learnable attention vector, and $\parallel$ denotes concatenation. The LeakyReLU$(\cdot)$ activation introduces a small non-zero slope for negative inputs, which helps avoid inactive units during training.

The resulting coefficients weight the contribution of neighboring nodes when forming the updated node representation. This mechanism is useful for task allocation because it allows the network to emphasize the most relevant agent-task relationships rather than treating all graph connections equally.

4.3 | Role in the Proposed System

The graph-attention allocator is used to assign tasks centrally across the UAV team, while each UAV executes its assigned objective using its own low-level guidance policy. This separation allows the task allocator to focus on

mission-level coordination and the guidance policy to focus on local navigation and obstacle avoidance. The use of graph attention is intended to improve the allocator's ability to capture structured relationships between tasks and agents in a dynamic indoor search scenario.

5 | Indoor Localization and Altitude Drift Mitigation

Reliable localization is essential for autonomous UAV operation in GNSS-denied environments. In indoor missions, the platform must estimate its position, velocity, and orientation without access to satellite navigation, while maintaining sufficient accuracy for obstacle avoidance, task execution, and map generation. Common approaches include visual odometry, inertial odometry, and LiDAR-based simultaneous localization and mapping (LiDAR-SLAM). Visual methods can degrade in feature-poor or poorly lit environments, while inertial estimates accumulate drift over time. For this reason, LiDAR-SLAM provides an attractive basis for localization in indoor search and rescue scenarios.

5.1 | LiDAR-SLAM for Indoor Navigation

LiDAR-SLAM constructs a map of the environment while simultaneously estimating the pose of the UAV by matching successive three-dimensional scans (Grisetti et al. 2005; Magnusson 2009; Segal et al. 2009). In principle, this allows the platform to localize robustly in environments where appearance-based methods are unreliable. Loop closure and graph-based optimization can further reduce accumulated error by recognizing previously visited areas and correcting the estimated trajectory.

5.2 | Altitude Drift in Corridor Environments

Although LiDAR-SLAM is effective in many indoor settings, narrow corridors and enclosed spaces can produce weak geometric constraints. In such environments, the scans may contain insufficient variation to constrain all axes of motion equally well, which can lead to drift in the estimated trajectory. In the present system, this effect was most evident in the vertical axis, where long corridor-like sections caused instability in the altitude estimate.

5.3 | Proposed Adaptation

To mitigate this issue, the system augments the LiDAR-SLAM estimate with an additional altitude correction stage. Depth-camera-derived altitude information is fused with vertical velocity from the inertial measurement unit using a lightweight complementary filter. This approach does not alter the underlying LiDAR-SLAM back-end; instead, it provides a practical correction for altitude drift during flight while retaining the benefits of LiDAR-based mapping and localization. The corrected altitude estimate is then used within the UAV state estimate for navigation and task execution in the indoor competition environment.

6 | Problem Formulation

We consider a cooperative UAV system comprising two learned decision-making modules: (i) a continuous-control guidance policy executed onboard each UAV for collision-aware navigation; and (ii) a centralized task-allocation policy executed at the ground station for assigning tasks across the UAV team. The guidance policy operates at a higher control frequency using local observations under partial observability, whereas the allocator operates at a lower frequency to coordinate mission-level tasking.

6.1 | Guidance as a Partially Observable Markov Decision Process

Each UAV navigation episode is modeled as a partially observable Markov decision process (POMDP). At time step t, the UAV receives a multimodal observation

$$o_t = \left(I_t^{\text{depth}}, \ell_t^{\text{LiDAR}}, p_t\right), \tag{4}$$

where I_t^{depth} is the forward-facing depth image, ℓ_t^{LiDAR} is a one-dimensional LiDAR sweep, and p_t is a compact positional state containing goal-related quantities, such as range, bearing, and altitude error, together with recent action history. The policy outputs a continuous bounded action

$$a_t = \left[v_{x,t}, \omega_t, v_{z,t}\right] \in [-1, 1]^3, \tag{5}$$

corresponding to forward velocity, yaw-rate, and vertical velocity commands. The environment then transitions to a new latent state, from which the next observation is generated, and the UAV receives an immediate reward r_t.

(a) APF-derived reference action. We compute an APF-inspired reference action $a_t^\star$ from the goal geometry and obstacle proximity. When the UAV is more than 1 m from nearby obstacles, the reference is governed by attractive terms:

$$v_{x,t}^\star = \text{clip}(\cos(\theta_t), 0.0, 1.0), \tag{6}$$

$$\omega_t^\star = \text{clip}\left(\theta_t \cdot \frac{3}{\pi}, -1.0, 1.0\right), \tag{7}$$

$$v_{z,t}^\star = \text{clip}\left(2\left(z_{\text{goal}} - z_t\right), -1.0, 1.0\right), \tag{8}$$

where θ_t is the heading error relative to the goal direction, and $\text{clip}(x, l, u) = \min(\max(x, l), u)$ saturates x to the interval $[l, u]$.

When an obstacle is detected within 1 m, the reference switches to a repulsive mode. The forward reference velocity is then set to move away from the obstacle according to the obstacle bearing θ_{obs}:

$$v_{x,t}^\star = -0.5\cos(\theta_{\text{obs}}), \tag{9}$$

and the yaw-rate reference follows a piecewise turning function $\omega_t^\star(\theta_{\text{obs}})$ that commands a turn away from the obstacle as a function of relative obstacle angle.

(b) Reward. The guidance reward is designed to encourage alignment between the executed action and the APF-derived reference action. The base reward is

$$r_t^{\text{base}} = 1 - \sum_{i \in \{x,\omega,z\}} \left(a_{t,i} - a_{t,i}^{\star}\right)^2, \tag{10}$$

which is clipped for training stability:

$$r_t = \text{clip}\left(r_t^{\text{base}}, -1.5, 1.0\right). \tag{11}$$

Terminal outcomes override this shaping reward: reaching the goal yields $r_t = R_{\text{goal}} = 2$, whereas collision with an obstacle yields $r_t = R_{\text{collision}} = -2$.

(c) Objective and termination. The guidance policy $\pi_\phi(a_t|o_t)$ is trained to maximize the expected discounted return

$$J(\pi_\phi) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right], \tag{12}$$

where γ is the discount factor. Episodes terminate upon goal completion, collision, or reaching a fixed horizon T.

6.2 | Task Allocation as a Centralized Graph Decision Process

Task allocation is formulated as a centralized reinforcement-learning problem defined over a fully connected graph $G_t = (V_t, E_t)$ at each decision epoch t. The graph comprises UAV nodes and task nodes. Node features encode the current status of UAVs and tasks, including occupancy and completion indicators, while edge attributes encode pairwise traversal costs computed from collision-free shortest-path distances between entities.

The allocator outputs an assignment matrix

$$A_t = \left[a_{\text{nk},t}\right] \in \{0,1\}^{N \times (M+1)}, \tag{13}$$

where $a_{\text{nk},t} = 1$ assigns UAV n to task k, with $k = 0$ denoting an idle action. The assignment is constrained such that each UAV receives exactly one decision:

$$\sum_{k=0}^{M} a_{\text{nk},t} = 1, \forall\, n \in \{1, \ldots, N\}. \tag{14}$$

The allocation reward encourages task completion while penalizing excess travel and collision events:

$$r_t = \Delta C_t - \lambda_d \Delta D_t - \lambda_c C_t, \tag{15}$$

where ΔC_t is the number of newly completed tasks, ΔD_t is the incremental distance traveled relative to a baseline, and C_t indicates collision events. Allocation episodes terminate when all tasks are completed or when a fixed horizon is reached.

7 | Methodology

This section describes the physical platform, sensing configuration, learning architectures, training environments, and deployment pipeline used to realize the cooperative UAV system.

7.1 | System Overview

The developed system combines three principal components: LiDAR-SLAM-based localization for GNSS-denied flight, a DRL-based onboard guidance policy for local obstacle-aware navigation, and a graph-attention-based central task allocator for coordinating the UAV team. The overall design was developed for indoor search and rescue operation within the SAPIENCE competition setting, where the UAVs were required to navigate autonomously, explore the environment, and coordinate task execution in a cluttered indoor arena.

7.2 | UAV Platform and Sensor Suite

Figure 3 shows the UAV platform used in this work. The platform was equipped with a Velodyne VLP-16 LiDAR (Velodyne Lidar 2016), an Intel RealSense D455f RGB-D camera (Intel Corporation 2024), and an NVIDIA Jetson Orin Nano companion computer (NVIDIA Corporation 2024b). The Jetson platform provided sufficient onboard computational capability for real-time inference of the guidance network and execution of the supporting perception pipeline.

The software stack was implemented in ROS 2 using Isaac ROS (NVIDIA Corporation 2024a), which provided the middleware for sensor integration, message passing, and interface development between perception, guidance, and flight-control components. Real-time localization and mapping were provided through the `lidarslam_ros2` package (Sasaki 2024). In practice, this LiDAR-SLAM pipeline provided reliable planar localization and heading estimation, but altitude drift was observed in feature-poor corridor sections, motivating the additional correction method described below.

7.3 | Altitude Drift Mitigation for LiDAR-SLAM

A practical challenge in the competition environment was the vertical drift of the LiDAR-SLAM estimate in corridor-like regions with limited geometric variation. A dedicated laser

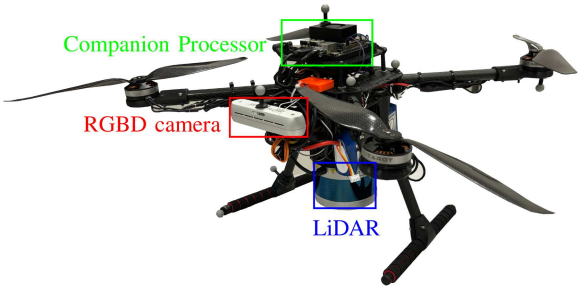

FIGURE 3 | City university's autonomous drone. [Color figure can be viewed at wileyonlinelibrary.com]

altimeter or radar altimeter could not be adopted because these sensors were not included in the permitted competition hardware. Instead, an additional Intel RealSense depth camera was used to provide altitude-related measurements.

To obtain a robust depth estimate, five equally spaced image regions were sampled from the depth frame: one at the image center and four located centrally within each quadrant. The median depth value was computed for each region, and the final altitude measurement was taken as the median of these five values. This reduced the influence of spurious depth readings and local outliers while taking advantage of the wider field of view of the depth sensor.

Because the operating environment was largely planar but included multiple levels, altitude correction was applied using a control loop that compared the vertical velocity measured by the flight controller IMU with the vertical rate inferred from consecutive depth measurements. Significant deviations between these quantities were used to detect changes in floor height, allowing the relative depth estimate to be updated and thereby reducing drift in the global altitude estimate.

7.4 | Guidance Network Architecture

The actor network used for TD3-based guidance fuses three sensing streams: depth imagery, LiDAR scans, and compact positional state variables, as illustrated in Figure 4. The network outputs bounded velocity-level control commands for the UAV.

7.4.1 | *Depth Branch*

Depth frames are processed by eight two-dimensional convolutional layers grouped into four blocks, each followed by max-pooling. LeakyReLU activations and batch normalization are used throughout (LeCun et al. 1998; Maas et al. 2013; Ioffe 2015). The final feature map is flattened and passed through two fully connected layers to produce a 128-dimensional embedding with a `tanh` output activation.

7.4.2 | *LiDAR Branch*

The one-dimensional LiDAR scan is processed by five one-dimensional convolutional layers arranged in three blocks, again using LeakyReLU activations and batch normalization. A final fully connected layer compresses the representation to a 128-dimensional embedding.

7.4.3 | *Positional-State Branch*

A compact state vector containing goal range, goal bearing, altitude error, relative height, and the previous x-velocity, z-velocity, and yaw-rate commands is mapped to a 128-dimensional embedding using a fully connected layer with `tanh` activation.

7.4.4 | *Fusion and Control Output*

The three 128-dimensional embeddings are concatenated to form a 384-dimensional fused representation. This is processed by five fully connected layers with LeakyReLU activation, followed by a final linear layer with `tanh` activation that outputs bounded forward velocity, vertical velocity, and yaw-rate commands in the range $[-1, 1]$.

Within the TD3 framework, this network serves as the actor and target actor. The critic networks use the same observation encoder, but concatenate the action at the fusion stage and regress a scalar Q-value, as shown in Figure 5.

7.5 | Guidance Training Setup

The guidance policy was trained in a custom simulation environment developed in Unreal Engine 4 (Epic Games 2014) and integrated with Microsoft AirSim. This environment

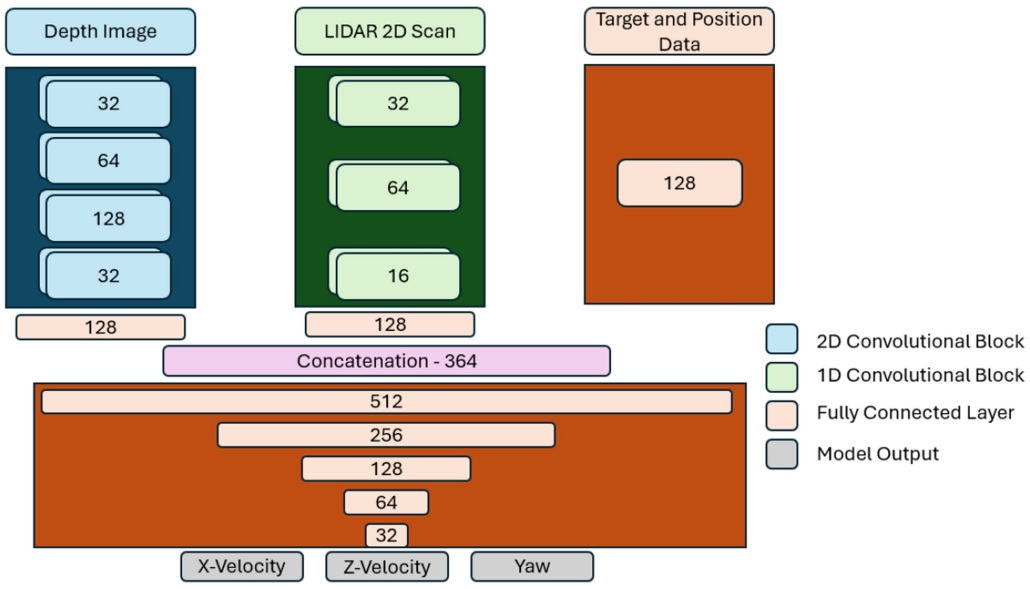

FIGURE 4 | The architecture for the guidance AIs actor network. [Color figure can be viewed at wileyonlinelibrary.com]

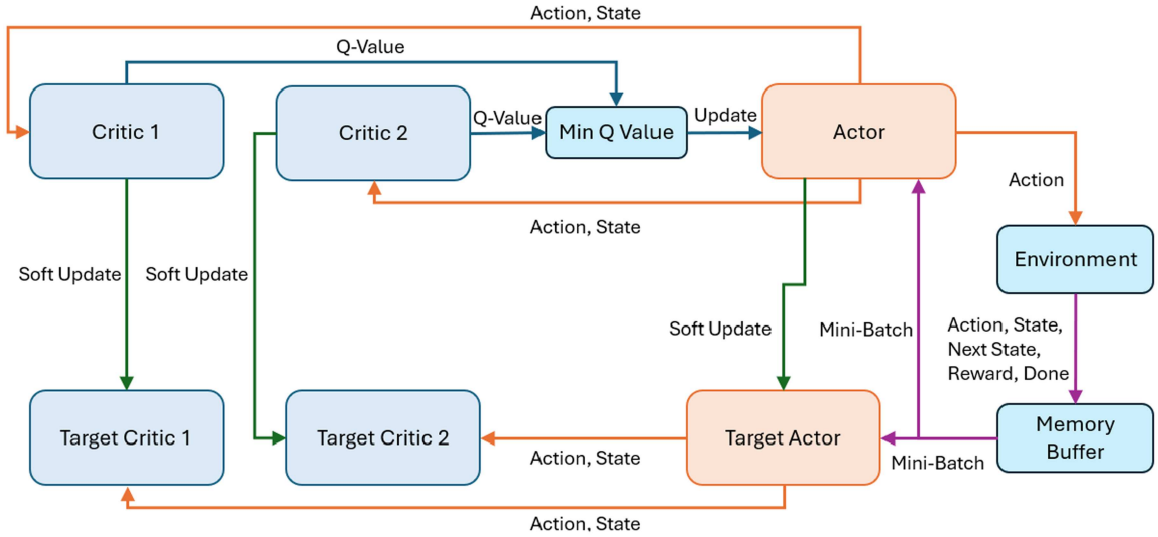

FIGURE 5 | The TD3 algorithm architecture used for training the guidance AI. [Color figure can be viewed at wileyonlinelibrary.com]

provided physics simulation, sensor emulation, and a controlled setting for repeated policy training.

The training environment included multiple obstacle configurations designed to expose the UAV to a range of navigation challenges, including narrow corridors, walls requiring circumnavigation, raised sections, and barriers that required vertical manoeuvring. The route structure alternated between two path configurations during training in order to encourage adaptability rather than memorization of a single layout.

Training was performed on a workstation equipped with an NVIDIA RTX A4500 GPU (NVIDIA Corporation 2021). The agent was trained for 1000 episodes. After training, candidate models were evaluated in a separate validation environment designed to assess generalization to related but unseen obstacle arrangements. The model selected for real-world deployment was the one that demonstrated the most reliable goal completion and obstacle avoidance performance in this validation stage.

7.6 | Task Allocation Network Architecture

The cooperative task allocator was implemented as a centralized graph-attention-based actor-critic model trained using TD3. A centralized allocator was adopted for two practical reasons. First, the competition rules explicitly permitted and encouraged a centralized decision-making framework. Second, each UAV already maintained a communication link to the ground station for map-related data exchange, making centralized task assignment simpler and more reliable than introducing an additional peer-to-peer coordination layer.

The actor network uses stacked GATConv layers (Fey and Lenssen 2019) to process the task-allocation graph, as shown in Figure 6. A preprocessing linear layer expands the node features to 64 dimensions, followed by batch normalization and LeakyReLU activation. Two graph-attention layers with five

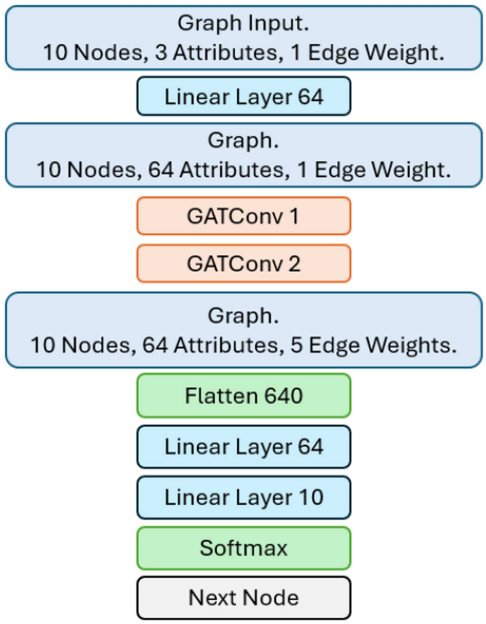

FIGURE 6 | The network architecture of the GAT-Based model. [Color figure can be viewed at wileyonlinelibrary.com]

attention heads each are then applied to produce context-aware node embeddings. A final multilayer perceptron produces an $(M + 1)$-way softmax output for each UAV, yielding the task assignment in a single forward pass.

The two critic networks share the same graph encoder as the actor and concatenate the chosen action before the final regression layers in order to estimate the corresponding Q-values.

7.7 | Task Allocation Training Setup

The task allocator was trained for 20,000 episodes on an RTX 3070 Mobile GPU. Mini-batches were sampled from a replay

buffer, and the TD3 training procedure used clipped double-Q learning, delayed actor updates, and target-policy smoothing noise applied to next-state actions. The reward design encouraged productive and efficient task assignment while penalizing collisions and unnecessary travel, as defined in Section 6.

7.8 | System Deployment

7.8.1 | *Guidance Deployment*

The trained guidance model was deployed on the NVIDIA Jetson Orin Nano (NVIDIA Corporation 2024b). Owing to the relatively lightweight network and the available onboard compute, the inference pipeline was implemented directly in Python. ROS 2 Isaac (NVIDIA Corporation 2024a) was used to manage communication between sensors, estimation modules, and the guidance controller. Guidance outputs were transmitted to the PX4 flight controller through the μXRCE-DDS bridge and the `px4_msgs` ROS 2 interface (PX4 Development Team 2023), enabling real-time command execution onboard the UAV.

7.8.2 | *Task Allocation Deployment*

Task allocation was executed on a laptop-based ground station communicating with the UAVs over Wi-Fi. Each UAV periodically requested updated tasking information from the central server and received a target node together with an associated path through the environment graph. Because direct traversal between nodes was not always feasible, the server generated intermediate waypoints when necessary to route the UAV around obstacles or walls.

Once a UAV reached its assigned target node, it issued a new request to the task allocation server and received its next assignment. Separate graph instances were maintained for each UAV and updated as tasks were completed, ensuring that both agents operated with consistent and current mission information. After all assigned nodes had been visited, the server generated and transmitted a return path to the take-off location.

8 | Test Results and Analysis

This section evaluates the proposed system in simulation, controlled real-world testing, and the SAPIENCE competition environment. The results are organized to assess four aspects of performance: guidance-policy validation, task-allocation performance, localization accuracy with altitude correction, and end-to-end cooperative execution during mapping and delivery missions.

8.1 | Simulation Validation

8.1.1 | *Guidance-Policy Validation*

Training of the guidance policy produced 260 candidate models that satisfied the initial selection criterion based on successful target acquisition. In the first validation stage, these models were re-evaluated in the training environment, and approximately the top 10% were retained for further testing. Using a threshold of 20 goal completions across four episodes, 34 models were selected.

These 34 models were then evaluated in a separate validation environment designed to better reflect the geometry and manoeuvring requirements of the intended real-world deployment, as shown in Figure 7. In this setting, the best-performing model achieved 304 successful goal reaches out of a possible 320 across 20 validation runs, with only two collisions. Both failures occurred during a demanding manoeuvre involving descent from an elevated platform followed by a 180° turn around a corner.

The models that progressed to this second validation stage were drawn from a training interval spanning approximately 77,000 to 170,000 observed states, out of a total of approximately 250,000 states encountered during training. This suggests that performance saturated before the end of training, and that continued exposure to the same environment did not necessarily yield improved policies. Based on the validation results, model 155260 was selected for real-world deployment.

For comparison, an alternative guidance policy was trained using a conventional distance-to-goal reward. Despite several trials with different hyperparameter settings and reward weightings, this formulation did not produce a model that met the same deployment threshold as the APF-referenced reward within the available training budget. Although this does not constitute a full convergence study, it provides empirical support for the practicality of the proposed reward design under the project constraints.

8.1.2 | *Task-Allocation Validation*

The task-allocation model was validated using a two-stage procedure. During training, models that completed the graph in fewer than six decision steps were retained for further evaluation. In the first validation stage, these candidate models were tested on the graph corresponding to the intended real-world layout. In the second stage, the selected models were evaluated on randomly generated graphs in order to assess the generality of the learned allocation strategy.

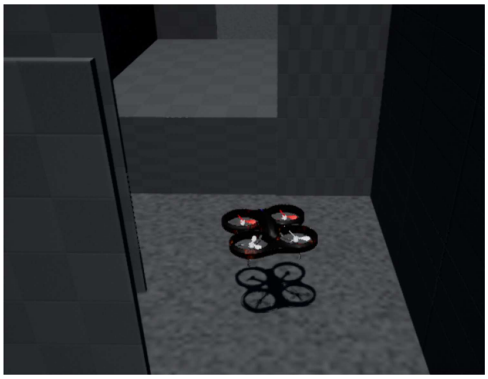

FIGURE 7 | The trained model running through the validation environment. [Color figure can be viewed at wileyonlinelibrary.com]

Each model was tested five times on each random graph, and performance was assessed using the total travel distance accumulated across all trials. The selected model consistently completed the real-world graph in four decision steps, indicating that no unnecessary reassignment steps were introduced. It also achieved the lowest average travel distance among the shortlisted models, with a mean distance 6.0 m below the median of the candidate set.

To contextualize the proposed graph-attention allocator, its assignment outcome was compared with representative classical baselines using the same waypoint-graph cost metric as the learned method, that is, shortest-path distances on the corridor-constrained graph. The baselines comprised greedy nearest-task assignment, an auction-style bidding allocator with bids proportional to path cost, and an exact one-step minimum-cost matching formulation recomputed after each completed task. On the field layout used in this work, all methods produced the same task sequence for both UAVs and therefore the same total travel distance and makespan, as shown in Table 1. This indicates that the deployed competition layout exhibits a ceiling effect, in which the optimal assignment is largely unambiguous once accurate graph distances are used. Accordingly, the main value of the proposed allocator on this layout lies not in outperforming the classical methods on static cost alone, but in demonstrating competitive performance within an integrated learned autonomy stack suitable for deployment.

8.2 | Testing in a Real-World Environment

The real-world testing of the UAV systems was conducted in the Autonomous Systems Arena at City, St George's University of London. For initial tests, the arena was configured as an open space measuring 12 m by 8 m, with large cardboard boxes used as obstacles. These obstacles served to limit potential damage to both the UAV and the testing environment while providing basic challenges for guidance and navigation.

The arena is equipped with Optitrack cameras (NaturalPoint Inc. 2023), which provide precise positional data by tracking reflective markers affixed to the UAV. This setup ensures accurate localization of the UAV without relying on GNSS, facilitating the evaluation of the adapted LiDAR-SLAM odometry solution adopted in this controlled environment.

Following the initial phase of simple obstacles, a more complex structure was assembled within the arena, featuring a floor plan of 10.8 m by 6 m and walls measuring 2.4 m in height. This structure included a network of 2 m wide corridors and a room measuring 3.6 m by 4.8 m, featuring a 1 m raised floor. This setup was designed to more closely resemble the real-world environment of indoor search and rescue challenges. A 2D schematic of the layout is presented in Figure 2. Unlike simulations, where every action is precisely controlled and predictable, real-world environments introduce various sources of uncertainty and unpredictability.

8.2.1 | *Real-World Sensor Characteristics*

Compared with their simulated counterparts, the real sensors were affected by measurement noise, occlusion, and shadowing. This was particularly evident for the Intel RealSense D455f depth camera, whose stereoscopic depth estimates degraded at shorter ranges. To mitigate this, an alternative depth representation was generated by interpolating the LiDAR point cloud over the region of interest. This produced a sharper depth image with fewer shadow artefacts than the camera-derived depth image, as illustrated in Figures 8 and 9. The point-cloud-derived image also avoided the blind regions associated with stereoscopic sensing. Its main limitation, however, was the restricted vertical field of view of the 16-line LiDAR, which captured a smaller vertical portion of walls than the RGB-D camera. Figure 10 shows the planar LiDAR scan used as an additional guidance input for all-around obstacle sensing.

8.2.2 | *Obstacle Avoidance and Path Following*

Initial flight tests were conducted in position-control mode in order to verify stable take-off, landing, and waypoint tracking. Preliminary guidance tests in obstacle-free conditions showed that the onboard control loop operated with a cycle time of 0.05–0.07 s, corresponding to an update frequency of approximately 14–20 Hz. Under these conditions, the UAV exhibited overshoot near the goal because momentum was not sufficiently reduced as the target was approached. This was mitigated by reducing the commanded speed near the goal and enlarging the accepted yaw-alignment range.

Subsequent tests introduced cardboard obstacles, initially singly and then in more complex arrangements, as shown in Figure 11. The UAV successfully executed basic avoidance manoeuvres around isolated obstacles. However, in denser arrangements, the vehicle occasionally passed too close to obstacles because the simulation-trained policy did not fully account for the momentum of the real platform. To improve robustness, an emergency avoidance override was introduced. When the planar LiDAR detected an obstacle within 0.6 m, the commanded velocity was temporarily biased away from the obstacle direction according to

TABLE 1 | Task allocation comparison of the GNN with classical methods in the Sapience layout.

Method	Total distance (m)	Makespan (m)	Steps
Greedy nearest-task	24.60	12.60	5
Auction/bidding	24.60	12.60	5
Optimal matching	24.60	12.60	5
Ours (GAT-DRL)	24.60	12.60	5

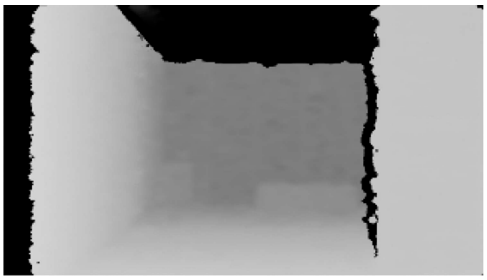

FIGURE 8 | The depth image from the Intel Realsense D455f camera. [Color figure can be viewed at wileyonlinelibrary.com]

FIGURE 9 | The depth image extrapolated from the Velodyne 16-Line Puck sensor. [Color figure can be viewed at wileyonlinelibrary.com]

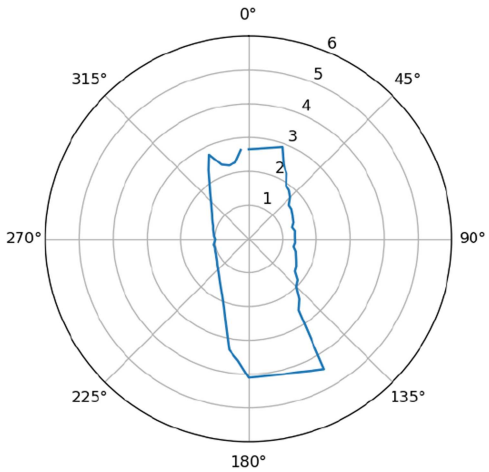

FIGURE 10 | The 2D LiDAR scan used as an input to the guidance AI. [Color figure can be viewed at wileyonlinelibrary.com]

$$v_x = -0.2\cos\left(\frac{\pi s}{36}\right), \tag{16}$$

$$v_y = -0.2\sin\left(\frac{\pi s}{36}\right), \tag{17}$$

where $s \in \{0, \ldots, 71\}$ denotes the index of the nearest LiDAR return sector. This modification substantially improved clearance in near-collision situations. Unlike the learned policy, the emergency override used a temporary planar repulsive command in the horizontal plane, including a lateral component v_y, for last-resort clearance.

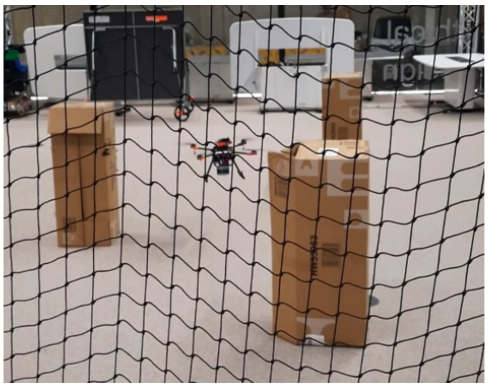

FIGURE 11 | The UAV being tested for obstacle avoidance with the cardboard obstacles. [Color figure can be viewed at wileyonlinelibrary.com]

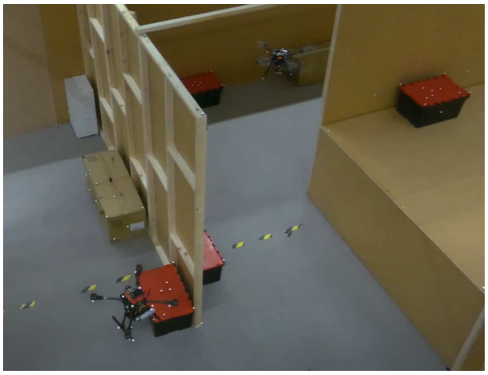

FIGURE 12 | The UAVs moving through the course cooperatively. [Color figure can be viewed at wileyonlinelibrary.com]

The arena was then expanded to include corridor sections and a raised platform. Although the guidance policy could command altitude changes, direct z-velocity control initially produced oscillatory behavior. Limiting the commanded vertical speed to ± 0.3 m/s stabilized these transitions. An additional condition was also imposed to ensure that the UAV fully cleared the raised ledge before descending. With these changes in place, the UAV successfully traversed the complete environment in multiple patterns, demonstrating readiness for cooperative testing (Figure 12).

8.2.3 | *Real-World Task-Allocation Performance*

To evaluate real-world task allocation, the two UAVs were positioned at opposite ends of the test structure, as shown in Figure 13. A prior two-dimensional floor plan of the environment was used to define the graph nodes and admissible internode connections.

The task-allocation server computed the shortest feasible route between assigned nodes and inserted intermediate waypoints when direct traversal was not possible. A conflict-detection mechanism was also included to prevent path intersections that might arise during execution. In all evaluated trials, the UAVs successfully navigated to their assigned nodes and executed the planned waypoint sequences, as illustrated in Figure 14. These results confirm that the central allocator and the onboard

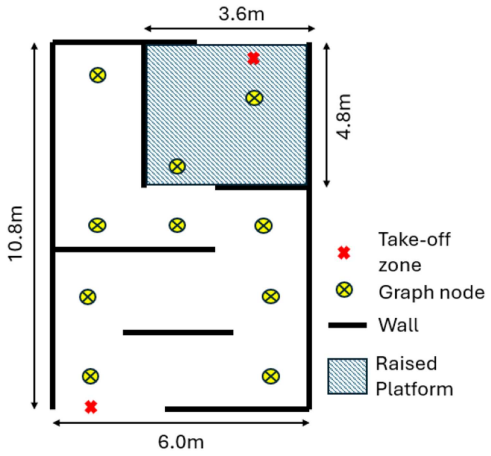

FIGURE 13 | The floor plan for the constructed building with the node locations for the graph. [Color figure can be viewed at wileyonlinelibrary.com]

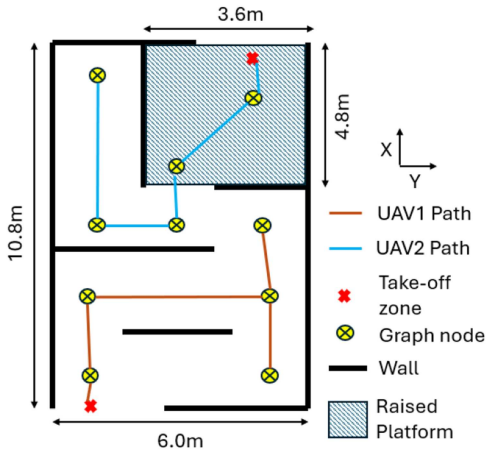

FIGURE 14 | Executed waponit paths for the two UAVs on the constructed indoor layout. [Color figure can be viewed at wileyonlinelibrary.com]

guidance policies could be combined reliably in real-world operation.

8.2.4 | LiDAR-SLAM Altitude-Correction Performance

The adapted LiDAR-SLAM solution was evaluated in the indoor arena using OptiTrack as an external reference. Although OptiTrack provides high-accuracy ground-truth positioning, its measurements were occasionally degraded by occlusion from the arena walls, which introduced brief tracking errors and occasional dropouts.

The baseline LiDAR-SLAM solution provided accurate localization in the horizontal plane but exhibited substantial drift in the vertical direction. To address this, the altitude-correction method described in Section VI.B was applied. Figure 15 compares the raw LiDAR-SLAM estimate, the corrected estimate, and the OptiTrack reference over time for the three position axes. The raw estimate closely follows the reference in x and y, but diverges significantly in z shortly after take-off. In contrast, the corrected estimate maintains substantially better agreement with the reference throughout the flight.

The remaining visible deviations are associated mainly with intervals in which the OptiTrack reference itself is affected by occlusion. This is supported by the repeated recovery of the UAV to consistent landing heights, indicating that the corrected altitude estimate remained stable even when the external reference was intermittently degraded. Figure 16 further shows that the UAV maintained consistent altitude transitions between the raised and lower floor levels during operation.

8.3 | The Sapience Autonomous Cooperative Drone Competition

The Sapience competition took place over a week in August 2024 at City, St George's University of London, with four teams from NATO and NATO partner countries participating. The competition consisted of three tasks designed to simulate a search and rescue operation within a specially constructed indoor arena. Emphasis was placed on the speed of task completion to encourage cooperative use of the two UAVs made available for use in this competition. Teams were additionally evaluated on the innovation in their solutions and the quality of their system outputs.

The first task required each team to produce a legible 3D arena map quickly. In the second task, teams were tasked with detecting and localizing three mannequins and five boxes within the arena, with accuracy in detection and localization serving as key evaluation criteria. The final task involved completing eight deliveries to the mannequins, with the last two deliveries requiring synchronization between the UAVs. Teams were assessed on the number of successful deliveries and the speed of completion within the allotted time. These tasks simulate essential search and rescue activities, such as those required in the aftermath of an earthquake.

The present paper focuses on the first and third competition tasks, as these provide the clearest evaluation of cooperative autonomous flight, mapping, and coordinated mission execution. The second task primarily involved short preplanned data-collection flights for object detection and localization, and is therefore outside the scope of the present study.

8.3.1 | System Performance in the 3D Mapping Task

The first competition task required rapid generation of a legible three-dimensional map of the environment. This task exercised all major components of the system, including cooperative navigation, task allocation, GNSS-denied localization, and altitude adaptation across multiple floor levels.

Figure 16 shows the trajectories of the two UAVs during the mapping task, expressed in meters in the LiDAR-SLAM world frame. The origin (0, 0, 0) corresponds to the lower-level take-off location shown in Figure 14, while the second UAV started on the raised platform at approximately (10.5, 3.2, 1.0). The

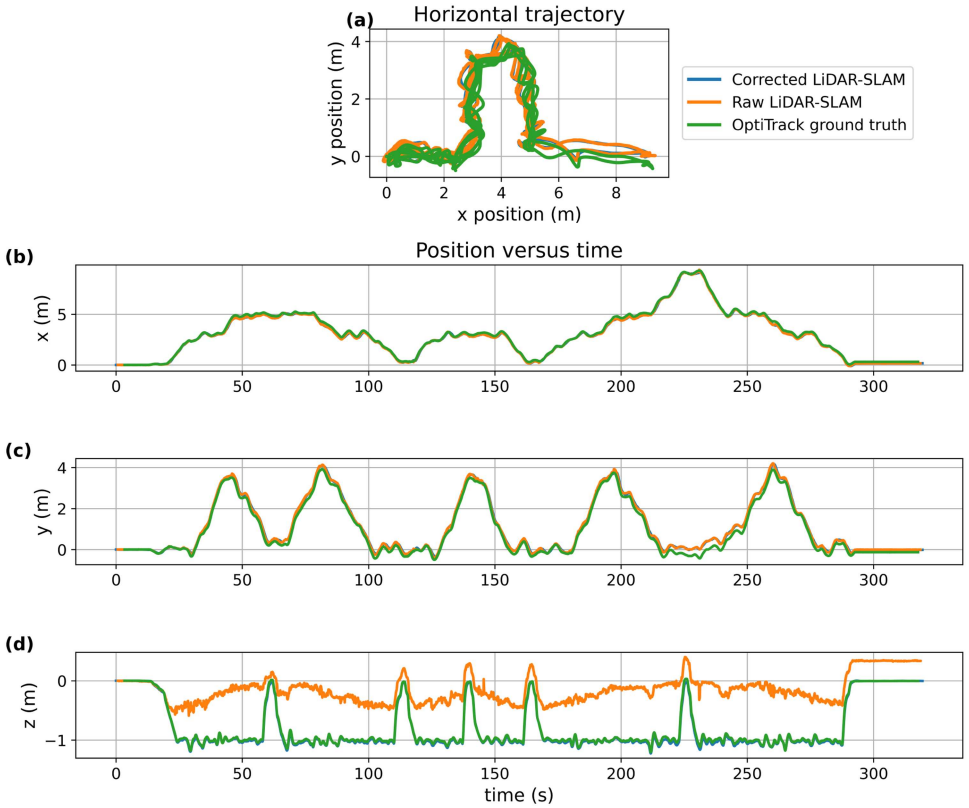

FIGURE 15 | Comparison of raw LiDAR-SLAM, corrected LiDAR-SLAM, and OptiTrack ground truth. (a) Horizontal trajectory in the x-y plane. (b–d) Position components as functions of time. All trajectories are normalized to start at the origin to highlight relative drift behavior. [Color figure can be viewed at wileyonlinelibrary.com]

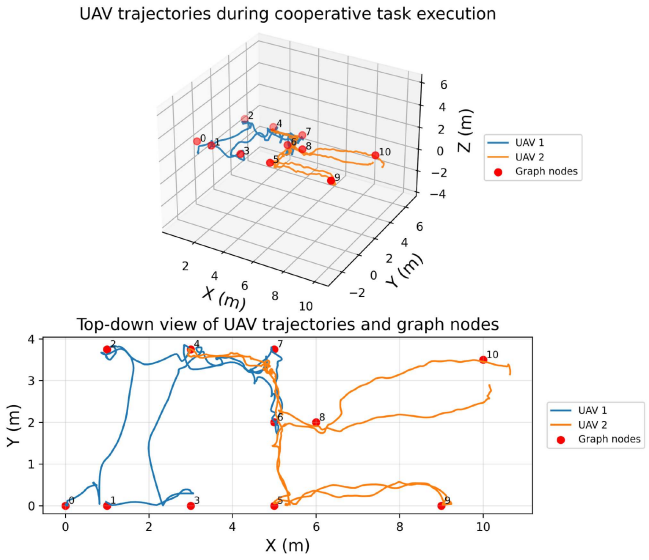

FIGURE 16 | Trajectories of the two UAVs during the cooperative mapping task. The plot shows the LiDAR-SLAM world-frame positions in meters, including both the three-dimensional view and the top-down projection. [Color figure can be viewed at wileyonlinelibrary.com]

three-dimensional view shows that UAV 1 covered the lower section of the arena, including the circuit around the free-standing wall, whereas UAV 2 descended from the raised platform to survey the long corridor before returning to its starting area.

The top-down projection highlights a local difficulty for UAV 1 near the central junction, where multiple nearby obstacles reduced the available clearance and occasionally required repeated attempts to pass the gap between the lower inner wall and the eastern outer wall. Although the resulting path is less smooth than a pre-planned trajectory, it demonstrates the ability of the learned guidance policy to adapt online in a realistic indoor SAR setting.

Figure 17 compares two flights of UAV 1 and illustrates the effect of a hardware modification to the electronic speed controller (ESC) placement. The pre-competition flight exhibited degraded altitude stability associated with ESC overheating, whereas the competition flight showed substantially improved height regulation after the ESCs were relocated to the UAV arms.

The resulting merged point cloud is shown in Figure 18. The second UAV was initialized with an approximate relative pose to enable subsequent point-cloud registration between the two platforms. Using cooperative operation, the mapping task was completed in 134 s, compared with 287 s for the equivalent single-UAV run, corresponding to a reduction of 53.3% in completion time.

8.3.2 | *System Performance in the Delivery Task*

The delivery task required the UAVs to complete eight deliveries to three designated locations in the arena. A delivery was

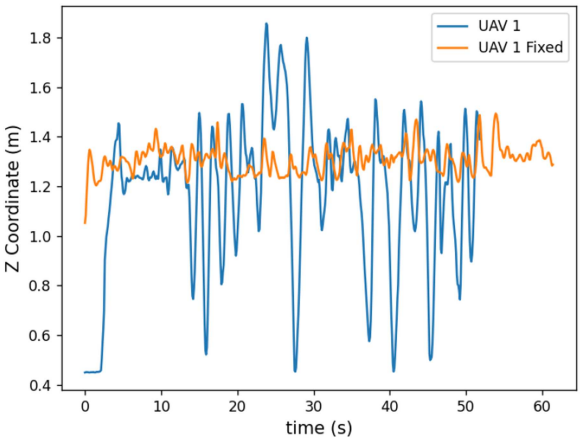

FIGURE 17 | Altitude traces of UAV 1 before and after relocation of the ESCs. The modified configuration reduced altitude instability associated with ESC overheating during flight. [Color figure can be viewed at wileyonlinelibrary.com]

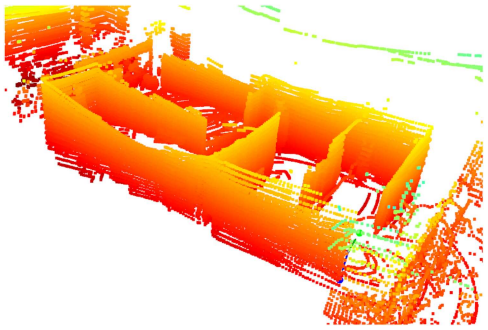

FIGURE 18 | Merged three-dimensional point cloud generated from the cooperative mapping flight using the registered LiDAR-SLAM outputs of both UAVs. [Color figure can be viewed at wileyonlinelibrary.com]

counted as successful when a UAV reached the target, landed for a specified dwell time, and then returned to its start location. The final two deliveries were required to be completed simultaneously, providing a direct test of cooperative timing.

Figure 19 shows the executed trajectories during the task. To avoid conflict at the first delivery point, UAV 1 performed the initial delivery before UAV 2 was released. After UAV 2 completed its first task, UAV 1 resumed its subsequent deliveries. To improve temporal alignment for the final synchronized deliveries, UAV 1 was delayed by 10 s at each take-off because UAV 2 had longer interdelivery travel distances.

For the final synchronised delivery, the first UAV to arrive at its target hovered above the landing point and transmitted a ready signal to the ground station. Once both UAVs had reported readiness, the server issued a simultaneous landing command. The timing traces in Figure 20 show the sequence of take-offs, landings, and the final synchronised landing event. During this task, the maximum UAV speed was increased to 1 m/s in order to prioritize completion time. Figure 21 shows representative onboard visual output during one of the delivery runs.

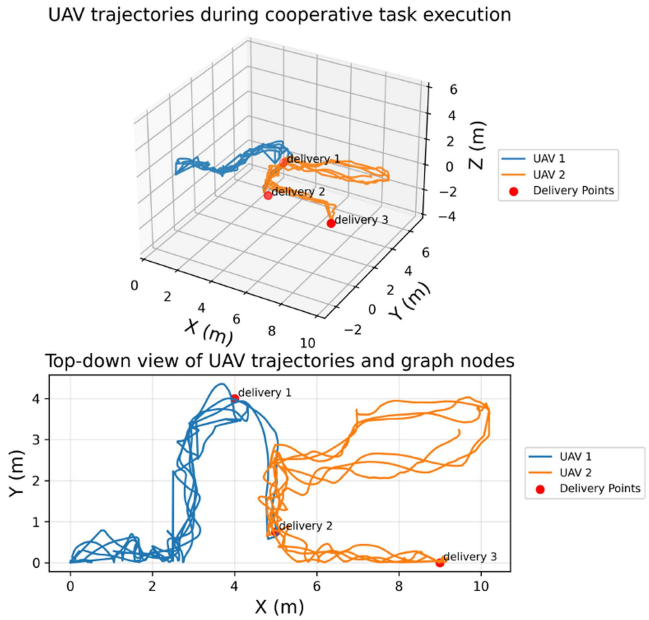

FIGURE 19 | The path of the two UAVs through the building whilst doing the delivery task. [Color figure can be viewed at wileyonlinelibrary.com]

8.4 | Demonstration on an Alternative Layout

To provide additional evidence that the proposed cooperative framework is not specific to a single arena geometry, we include a brief demonstration on a secondary indoor layout used for system checks and demonstrations. This layout is intentionally simpler than the Sapience competition arena due to the complexities of erecting the Sapience arena and is therefore not presented as a stronger benchmark. Rather, it serves as a compact sanity check that the same end-to-end pipeline (waypoint graph construction, graph-based task allocation, and learned guidance execution) can be applied without altering the controller structure or task allocation logic.

Figure 22 shows the alternative layout and the set of task waypoints. The waypoint graph is constructed in the same manner as in the competition setting, and allocator decisions are computed using a GNN to determine the most efficient way to visit all nodes and complete the cooperative point cloud. The arena consists of two drones, one at each end, separated by a T-shaped obstacle in the middle with three mannequins arranged around it. The drones are also commanded to land at the opposite take-off site to reduce the time required to complete the task.

Table 2 compares the proposed GAT-DRL allocator against representative classical methods on the demonstration layout, using shortest-path distances on the same waypoint graph as the cost metric. The auction/bidding and optimal matching baselines yield identical performance in this instance, with a total travel distance of 26.30 m and a makespan of 14.88 m (where makespan denotes the maximum distance assigned to any single UAV). In contrast, the proposed allocator reduces the total travel distance to 21.19 m and the makespan to 11.94 m, corresponding to reductions of 19.4% and 19.8%, respectively. The proposed method also completes the allocation in fewer

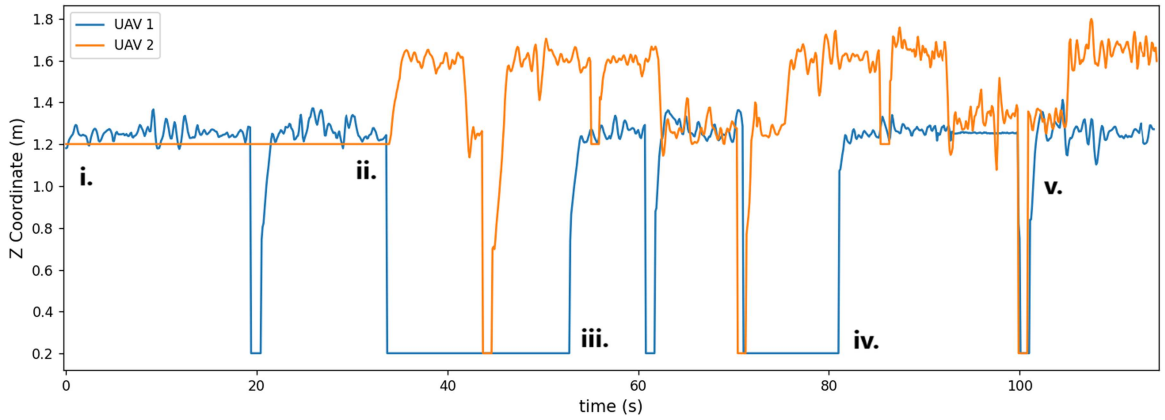

FIGURE 20 | The height (z) against time to show the deliveries and the final synced delivery. (i) UAV 2 does not take off until UAV 1 has completed its first delivery. (ii) Once UAV 1 is landed, UAV 2 takes off to complete its task. (iii) As UAV 2 makes the last conflicted delivery, UAV 1 starts again but is delayed by 10 s to better sync the deliveries at the end. (iv) The UAV 1 delayed take-off when at base. (v) The sync delivery takes place once the UAVs detect they have both reached the delivery point. [Color figure can be viewed at wileyonlinelibrary.com]

FIGURE 21 | Representative onboard camera view during the delivery task, showing a UAV descending from the raised section to the lower floor. [Color figure can be viewed at wileyonlinelibrary.com]

steps between nodes (4 vs. 7), indicating a more stable and efficient division of work between the two UAVs on this layout.

Figure 23 summarizes the resulting routes from the odometry as measured by the UAVs. Although this layout is simpler, the result demonstrates that the allocator and guidance policy remain coherent under a change in arena geometry, producing consistent task separations between agents and feasible collision-free routes.

To complement the top-down layout depiction, Figure 24 shows the aggregated 3D point cloud collected during the Demo-layout flight.

9 | Additional Lessons Learned From UAV Deployment

Deployment of the UAV system in the SAPIENCE competition environment highlighted several practical issues that were not fully exposed during simulation-based development. These included controller-output tuning for stable real-world flight, the need for an additional short-range collision-mitigation mechanism, and limitations in the fidelity of the simulation environment used for training.

9.1 | Tuning of the Guidance Policy Outputs

Initial deployment of the learned guidance policy showed that some control outputs, particularly the vertical velocity command, were too aggressive for stable execution by the flight controller. In particular, direct use of the learned z-velocity output led to oscillatory behavior around the target altitude. Constraining the commanded vertical velocity to ± 0.3 m/s substantially improved stability and enabled smoother altitude regulation near target positions. Additional tuning of forward velocity and yaw-rate behavior near goals also improved waypoint convergence and reduced overshoot during approach.

9.2 | Need for an Additional Collision-Mitigation Layer

Although the guidance policy was trained with an obstacle-avoidance objective, real-world deployment showed that learned avoidance alone was not always sufficient under the true platform dynamics. Vehicle momentum, actuation delay, and modeling mismatch occasionally caused the UAV to pass too close to obstacles, particularly in confined sections of the arena. For this reason, an additional APF-inspired emergency avoidance layer was introduced as a safety mechanism during deployment.

This highlighted an important lesson from the field trials: reward shaping alone may not fully capture the safety requirements of real flight, and deployment-ready systems may still require explicit short-range protection layers or additional robustness mechanisms.

9.3 | Limitations of the Training Environment

A further lesson concerned the quality and representativeness of the simulation environment. The AirSim-based training and validation environments were sufficient for initial policy development, but they did not fully capture the geometric, material, and sensing complexities of the real arena. In practice, larger open spaces, reflective surfaces, netting, and visually irregular materials produced sensing artefacts that were absent from the training domain. The depth camera was particularly affected by these discrepancies. Creating highly realistic depth-consistent environments in AirSim also proved challenging, because scene assets and textures required adaptation for correct depth rendering. This limited the realism of the simulated sensing pipeline and contributed to the remaining simulation-to-real gap observed during deployment.

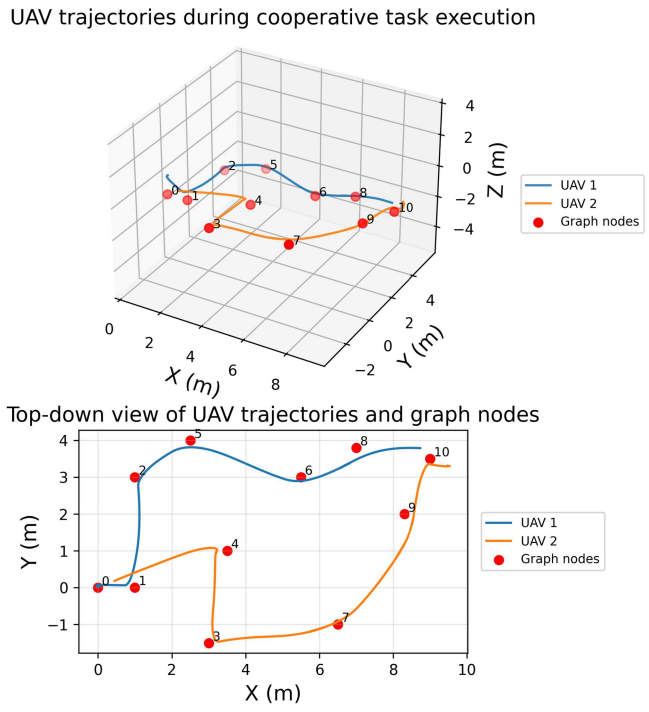

FIGURE 22 | The routes of the two UAVs when completing the Demonstration task. [Color figure can be viewed at wileyonlinelibrary.com]

10 | Future Work

The three-stage NATO SAPIENCE program provides a natural path for extending the present framework beyond the indoor competition setting. Future work will focus on broadening the operating environment, increasing decentralization, and strengthening multi-agent perception, guidance, and localization.

10.1 | Outdoor and Mixed Indoor-Outdoor SAR

The subsequent SAPIENCE stages include fully outdoor missions followed by mixed indoor-outdoor scenarios. These

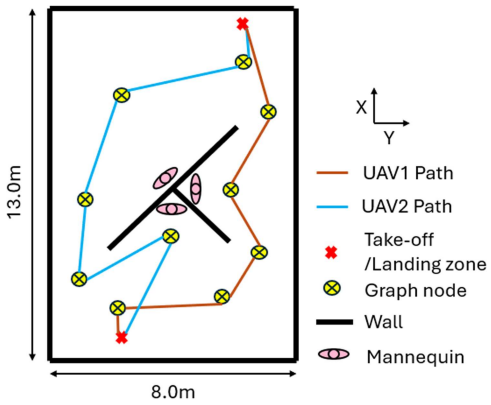

FIGURE 23 | Waypoint-graph visualization of the Demo layout, showing the routes executed by each UAV. [Color figure can be viewed at wileyonlinelibrary.com]

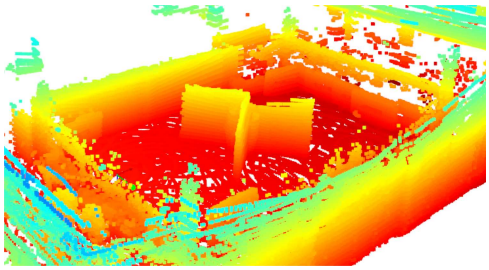

FIGURE 24 | The combined 3D point cloud collected from both UAVs as they have completed their allotted tasks. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 2 | Task-allocation comparison on the alternative demonstration layout. Makespan is reported as the maximum path length assigned to any single UAV.

Method	Total distance (m)	Makespan (m)	Steps
Greedy nearest-task	29.26	14.64	7
Auction/bidding	26.30	14.88	7
Optimal matching	26.30	14.88	7
Ours (GAT-DRL)	21.19	11.94	4

environments introduce additional challenges, including feature-sparse outdoor areas, larger operating ranges, and intermittent or unavailable communication with a central ground station. Addressing these conditions will require fully onboard autonomy, more robust multi-sensor odometry, and airframes capable of carrying increased sensing and power payloads for extended mission duration.

10.2 | Multiagent DRL for Guidance

The present work uses independently trained TD3 guidance policies with shared state information at execution time. A natural next step is to investigate multiagent DRL formulations in which cooperation is represented directly during training. Centralized-critic or joint-value approaches could allow the guidance policies to model pairwise collision avoidance, cooperative coverage, and shared mission objectives more explicitly than the current independent-controller design.

10.3 | Decentralized Task Allocation

The current task-allocation framework relies on a central ground-station server. For future competition stages and practical field deployment, this functionality will need to be integrated into the UAV. Two decentralized deployment strategies are of particular interest: a master-slave architecture, in which one UAV performs inference and broadcasts assignments, and a fully peer-to-peer approach, in which task allocation is distributed across the fleet. The latter would reduce the single-point-of-failure risk and better support operations when infrastructure is unavailable.

10.4 | Collaborative Perception and Mapping

Future work will also extend the present localization and mapping framework towards collaborative multi-UAV perception. Sharing compact pose-graph information, loop-closure cues, or lightweight perceptual summaries across the team could improve global consistency and reduce accumulated localization error. In particular, the depth-IMU altitude-correction approach developed here could be extended to a fleet-level multiagent estimation framework, enabling more reliable altitude consistency across agents during large-scale indoor or mixed-environment operation.

10.5 | Towards End-to-End Cooperative Autonomy

Beyond the competition setting, a longer-term objective is to replace the current waypoint-based execution framework with more tightly integrated cooperative autonomy. This includes goal-conditioned multiagent policies capable of reasoning jointly over neighboring agents, local obstacle structure, and mission objectives without relying exclusively on predefined intermediate waypoints. Coupled with shared perception and decentralized coordination, such a framework would move the present system toward a more general-purpose platform for collaborative UAV autonomy in cluttered GNSS-denied environments.

11 | Conclusion

This paper has presented the development and real-world deployment of a cooperative UAV system for operation in GNSS-denied indoor environments. The system combined LiDAR-SLAM-based localization and mapping, a velocity-level DRL guidance policy trained using an APF-referenced reward, and a centralized graph-attention-based task allocator for multi-UAV coordination. Unlike many prior studies that remain limited to simulation or simplified platforms, the present work evaluated the complete system through field deployment in a realistic indoor competition environment.

The SAPIENCE Autonomous Cooperative Drone Competition provided a demanding testbed for the proposed framework. Across cooperative mapping and synchronised delivery tasks, the system demonstrated the ability to perform autonomous navigation, coordinate multiple UAVs, and maintain operation in cluttered indoor spaces with constrained visibility, narrow passages, and changing floor height. The deployment also revealed several practical issues, including momentum-related near-collision behavior, altitude drift in corridor-like environments, and hardware-related flight instability, all of which informed subsequent improvements to the deployed system.

The results show that DRL-based guidance, when integrated with appropriate localization and coordination mechanisms, can support cooperative UAV operation beyond small-scale or purely simulated studies. More broadly, the work emphasizes the importance of end-to-end real-world evaluation in closing the simulation-to-real gap. The system design, reward formulation, and deployment lessons reported here provide a practical foundation for future work on collaborative aerial robotics in search and rescue and other GNSS-denied applications.

Acknowledgments

We would like to thank NATO for helping to organize the SAPIENCE competition, the various staff at City St George's University of London for their help in setting it up, and the other competition teams from Delft University of Technology, the University of Alabama in Huntsville, and the University of Klagenfurt for their excellent work and help in making it a success. I would finally like to thank the other members of our RAMI research group for their patience and help while organizing the competition.

Data Availability Statement

Since this competition is not open-source, the code is safeguarded by an intellectual property agreement under the NATO SPS Sapience program, and unfortunately, cannot be shared.

References

Alshaboti, M., and U. Baroudi. 2021. "Multi-Robot Task Allocation System: Fuzzy Auction-Based and Adaptive Multi-Threshold Approaches." *SN Computer Science* 2, no. 2: 87.

Azar, A. T., A. Koubaa, N. Ali Mohamed, et al. 2021. "Drone Deep Reinforcement Learning: A Review." *Electronics* 10, no. 9: 999.

Cui, J., Y. Liu, and A. Nallanathan. 2019. "Multi-Agent Reinforcement Learning-Based Resource Allocation for Uav Networks." *IEEE Transactions on Wireless Communications* 19, no. 2: 729–743.

Dong, R., X. Pan, T. Wang, and G. Chen. 2023. "Uav Path Planning Based on Deep Reinforcement Learning." In Artificial Intelligence for Robotics and Autonomous Systems Applications, edited by A. T. Azar, and A. Koubâa, 27–65. Springer.

Du, Y., N. Qi, X. Li, et al. 2024. "Distributed Multi-UAV Trajectory Planning for Downlink Transmission: A GNN-Enhanced DRL Approach." *IEEE Wireless Communications Letters* 13, no. 12: 3378–3582.

Epic Games. 2014. "Unreal Engine 4." Version 4.0 and Later. https://www.unrealengine.com/.

Fey, M., and J. E. Lenssen 2019. "Fast Graph Representation Learning With Pytorch Geometric." arXiv preprint arXiv:1903.02428.

Fujimoto, S., H. Hoof, and D. Meger 2018. "Addressing Function Approximation Error in Actor-Critic Methods." In International Conference on Machine Learning, edited by J. Dy, and A. Krause, 1587–1596. PMLR.

Ghauri, S. A., M. Sarfraz, R. A. Qamar, M. F. Sohail, and S. A. Khan. 2024. "A Review of Multi-UAV Task Allocation Algorithms for a Search and Rescue Scenario." *Journal of Sensor and Actuator Networks* 13, no. 5: 47.

Grisetti, G., C. Stachniss, and W. Burgard 2005. "Improving Grid-Based Slam With Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling." In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2432–2437. IEEE.

He, L., N. Aouf, and B. Song. 2021. "Explainable Deep Reinforcement Learning for UAV Autonomous Path Planning." *Aerospace science and technology* 118: 107052.

Hester, T., M. Vecerik, O. Pietquin, et al. 2018. "Deep q-Learning From Demonstrations." In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32). edited by S. A. McIlraith, and K. Q. Weinberger. AAAI Press.

Hickling, T., N. Aouf, and P. Spencer. 2023. "Robust Adversarial Attacks Detection Based on Explainable Deep Reinforcement Learning for UAV Guidance and Planning." *IEEE Transactions on Intelligent Vehicles* 8, no. 10: 4381–4394.

Hu, J., X. Yang, W. Wang, P. Wei, L. Ying, and Y. Liu. 2022. "Obstacle Avoidance for UAS in Continuous Action Space Using Deep Reinforcement Learning." *IEEE Access* 10: 90623–90634.

Intel Corporation. 2024. "Intel RealSense Depth Camera D455f."

Ioffe, S. 2015. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." arXiv preprint arXiv:1502.03167.

Khatib, O. 1986. "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." *International Journal of Robotics Research* 5, no. 1: 90–98.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE* 86, no. 11: 2278–2324.

Li, B., and Y. Wu. 2020. "Path Planning for UAV Ground Target Tracking via Deep Reinforcement Learning." *IEEE Access* 8: 29064–29074.

Li, Y., X. Zhang, Y. Zhu, and Z. Gao 2023. "A UAV Path Planning Method in Three-Dimensional Urban Airspace Based on Safe Reinforcement Learning." In 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC), 1–7. IEEE.

Lillicrap, T. P., J. J. Hunt, A. Pritzel, et al. 2015. "Continuous Control With Deep Reinforcement Learning." arXiv Preprint arXiv: 1509.02971.

Liu, Y., X. Li, J. Wang, F. Wei, and J. Yang. 2024. "Reinforcement-Learning-Based Multi-UAV Cooperative Search for Moving Targets in 3d Scenarios." *Drones* 8, no. 8: 378.

Liu, Z., L. Huang, Z. Gao, M. Luo, S. Hosseinalipour, and H. Dai. 2024. "GA-DRL: Graph Neural Network-Augmented Deep Reinforcement Learning for DAG Task Scheduling Over Dynamic Vehicular Clouds." *IEEE Transactions on Network and Service Management* 21, no. 4: 4226–4242.

Maas, A. L., A. Y. Hannun, and A. Y. Ng, et al. 2013. "Rectifier Nonlinearities Improve Neural Network Acoustic Models." In *Proceedings of the ICML (Vol. 30)*, 3.

Magnusson, M. 2009. "The Three-Dimensional Normal-Distributions Transform: An Efficient Representation for Registration, Surface Analysis, and Loop Detection." PhD thesis, Örebro universitet.

Mnih, V., K. Kavukcuoglu, D. Silver, et al. 2015. "Human-Level Control Through Deep Reinforcement Learning." *Nature* 518, no. 7540: 529–533.

NATO. 2024. "Sapience Autonomous Cooperative Drone Competition." https://sapienceproject.com/.

NaturalPoint, Inc. 2023. "OptiTrack Motion Capture Systems." https://optitrack.com.

NVIDIA Corporation. 2021. "NVIDIA RTX A4500 Graphics Card." https://www.nvidia.com/en-us/design-visualization/rtx-a4500/.

NVIDIA Corporation. 2024a. "NVIDIA Isaac ROS: Accelerated Perception and AI for ROS Developers."

NVIDIA Corporation. 2024b. "NVIDIA Jetson Orin Nano Series Modules Datasheet." DS-11105-001_v1.3, Rev F, August 2024.

PX4 Development Team. 2023. "Micro XRCE-DDS Middleware Integration." https://docs.px4.io/main/en/middleware/uxrce_dds.html.

Ren, X., N. Geng, Y. Zhang, L. Xiao, and D. Gong. 2025. "PG-ITD3: A Potential Field-Guided Deep Reinforcement Learning Approach for UAV Path Planning After Disaster." *IEEE Transactions on Automation Science and Engineering* 22: 20221–20233.

Ryu, H., H. Shin, and J. Park. 2020. "Multi-Agent Actor-Critic With Hierarchical Graph Attention Network." In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34), edited by V. Conitzer, and F. Sha, 7236–7243. AAAI Press.

Sargolzaei, A., A. Abbaspour, and C. D. Crane. 2020. "Control of Cooperative Unmanned Aerial Vehicles: Review of Applications, Challenges, and Algorithms." *Optimization, Learning, and Control for Interdependent Complex Networks* 1123: 229–255.

Sasaki, R. 2024. "lidarslam_ros2: Lidar Slam for ros 2." https://github.com/rsasaki0109/lidarslam_ros2.

Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. 2017. "Proximal Policy Optimization Algorithms." arXiv preprint arXiv:1707.06347.

Scoles, S. 2024. "How Drones Are Revolutionizing Search and Rescue." Scientific American.

Segal, A., D. Haehnel, and S. Thrun. 2009. "Generalized-ICP." In Robotics: Science and Systems (Vol. 2), edited by J. Trinkle, Y. Matsuoka, and J. A. Castellanos. 435. MIT Press.

Shah, S., D. Dey, C. Lovett, and A. Kapoor. 2017. "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles." In Field and Service Robotics. edited by M. Hutter and R. Siegwart, Springer International Publishing.

Shao, Y., R. Li, B. Hu, Y. Wu, Z. Zhao, and H. Zhang. 2021. "Graph Attention Network-Based Multi-Agent Reinforcement Learning

for Slicing Resource Management in Dense Cellular Network." *IEEE Transactions on Vehicular Technology* 70, no. 10: 10792–10803.

Shen, G., L. Lei, X. Zhang, Z. Li, S. Cai, and L. Zhang. 2023. "Multi-UAV Cooperative Search Based on Reinforcement Learning With a Digital Twin Driven Training Framework." *IEEE Transactions on Vehicular Technology* 72, no. 7: 8354–8368.

Song, Y., Z. Ma, N. Chen, S. Zhou, and S. Srigrarom. 2025. "Comparative Analysis of Centralized and Distributed Multi-UAV Task Allocation Algorithms: A Unified Evaluation Framework." *Drones* 9, no. 8: 530.

Tourani, A., H. Bavle, J. L. Sanchez-Lopez, and H. Voos. 2022. "Visual SLAM: What Are the Current Trends and What to Expect?." *Sensors* 22, no. 23: 9297.

Vecerik, M., T. Hester, J. Scholz, et al. 2017. "Leveraging Demonstrations for Deep Reinforcement Learning on Robotics Problems With Sparse Rewards." arXiv preprint arXiv:1707.08817.

Velickovic, P., G. Cucurull, A. Casanova, et al. 2017. "Graph Attention Networks." *stat* 1050, no. 20: 10–48550.

Velodyne Lidar, I. 2016. "Velodyne VLP-16 User Manual." https://www.manualslib.com/manual/1407706/Velodyne-Vlp-16.html.

Yu, Y., and S. Lee. 2023. "Efficient Multi-UAV Path Planning for Collaborative Area Search Operations." *Applied Sciences* 13, no. 15: 8728.

Zhang, A., and M. Maher Atia. 2020. "Comparison of 2d Localization Using Radar and Lidar in Long Corridors." In 2020 IEEE SENSORS, edited by J. Wang, 1–4. IEEE.

Zhang, Y., C. Yan, J. Xiao, and M. Feroskhan. 2024. "NPE-DRL: Enhancing Perception Constrained Obstacle Avoidance With Non-Expert Policy Guided Reinforcement Learning." *IEEE Transactions on Artificial Intelligence* 6, no. 1: 184–198.

Zhao, W., Q. Meng, and P. W. Chung. 2015. "A Heuristic Distributed Task Allocation Method for Multivehicle Multitask Problems and Its Application to Search and Rescue Scenario." *IEEE Transactions on Cybernetics* 46, no. 4: 902–915.

Zhu, Y., Y. Tan, Y. Chen, L. Chen, and K. Y. Lee. 2024. "UAV Path Planning Based on Random Obstacle Training and Linear Soft Update of DRL in Dense Urban Environment." *Energies* 17, no. 11: 2762.