Second US-Europe Air Transportation Research and Development Symposium (ATRDS2026)

# Learning-Based Pre-tactical Conflict Management Strategy for Urban Air Mobility

Yuheng Wang[a], Tinghe Zhang[a], Ang Li[a,b,*]
[a] Department of Aeronautical and Aviation Engineering
[b] Research Centre for Low-Altitude Economy (RCLAE)
Hong Kong Polytechnic University
Hong Kong, China

Peng Wei
Department of Mechanical and Aerospace Engineering
George Washington University
Washington DC, United States

***Abstract*—Conflict management in air traffic management (ATM) traditionally consists of strategic planning and tactical avoidance. However, such a two-level structure is insufficient for the UAV System Traffic Management (UTM) context because UTM operations are characterized by mixed scheduled and instant demands. The instant demand cannot be captured by strategic Demand Capacity Balancing (DCB), and will give too much computational burden on the tactical level for self-separation. To address this limitation, a pre-tactical conflict management is proposed between the strategic and tactical levels. Operating within a short planning horizon prior to departure, this layer performs centralized conflict management to proactively mitigate large-scale conflicts and reduce real-time tactical workload. To enable efficient and scalable decision-making, three key components are developed. An adaptive airspace abstraction transforms submitted free-flight trajectories into a structured conflict graph, balancing operational flexibility and computational tractability. The pre-tactical conflict management problem is formulated as a multi-agent contextual bandit. A Centralized Training Decentralized Execution (CTDE) learning framework outputs coordinated actions for conflict resolution, including ground delay, speed adjustment, and altitude rerouting. Numerical experiments demonstrate effective conflict resolution performance and strong scalability across varying traffic densities. The proposed architecture provides a structural extension to existing UTM systems for managing dynamic operations.**

***Keywords*—UAV System Traffic Management; Conflict Management; Reinforcement Learning**

## I. Introduction

UTM refers to the provision of safe and efficient air transportation services within urban environments using unmanned or electric vertical takeoff and landing aircraft. As a promising solution to alleviate ground traffic congestion, UTM has attracted extensive exploration in applications such as parcel delivery, air taxis, infrastructure inspection, monitoring, and emergency medical services. However, with the rapid growth of demand, low-altitude urban airspace is expected to become increasingly congested. High traffic density in constrained airspace significantly increases the risk of trajectory conflicts and potential collisions. Therefore, integrating efficient conflict management mechanisms into the Urban Traffic Management (UTM) system has become a fundamental requirement for safe and scalable UTM operations.

Conflict management is a core component of both traditional Air Traffic Management (ATM) and modern UTM systems. In current UTM architectures, service providers supply airspace information, while UAV flight operators independently generate flight plans, including scheduling, routing, and separation strategies, subject to airspace constraints. In conventional ATM, conflict management is generally structured into two phases [1]:

**Strategic phase.** Long-horizon planning and demand-capacity balancing (DCB), often leveraging optimization models for system-level scheduling and routing decisions [2]–[4].

**Tactical phase.** Real-time in-flight collision avoidance, increasingly supported by learning-based or control-based methods [5], [6].

Several studies have attempted to transfer this two-level conflict management framework from commercial aviation to UTM environments [7], [8], adapting strategic and tactical methodologies to low-altitude airspace.

Despite the structural similarity, UTM operations exhibit characteristics that fundamentally differentiate them from traditional airline operations. Unlike commercial aviation, where flight schedules can be finalized months in advance, UTM demand is highly dynamic and includes both scheduled and instant flight requests. Instant demand generation, such as food delivery and emergency medical services, is submitted shortly before departure. The strategic level is unable to capture such short-notice or instant demand in a timely manner. Large-scale instant demand may also exceed the upper limit of the handling capacity of the tactical self-separation, thereby increasing computational burden and operational risk. As a result, relying solely on the traditional strategic–tactical conflict management architecture may lead to either insufficient anticipation of conflicts or excessive reliance on reactive collision avoidance. Therefore, a more timely and computationally efficient conflict management mechanism is needed to handle dynamic demand before flights enter the airspace.

To address this gap, we introduce a pre-tactical conflict management layer, positioned between strategic and tactical levels. The pre-tactical phase operates within a rolling time window prior to departure and jointly evaluates both scheduled and newly submitted flight requests. Its objective

is to mitigate potential conflicts before operations commence, thereby reducing the burden on real-time tactical collision avoidance. In contrast to strategic long-term optimization and tactical reactive avoidance, the pre-tactical phase focuses on short-horizon, centralized, and computationally efficient conflict management under dynamic demand conditions. In essence, this pre-tactical conflict management framework constitutes a DCB problem with instant demand, focusing on short planning horizons, fast response times and centralized conflict management.

Given the requirement for fast response to on-demand conflict management, learning-based methods provide a promising solution for pre-tactical conflict management. Existing learning-based conflict management studies primarily concentrate on the tactical phase. Multi-Agent Reinforcement Learning (MARL) approaches have been widely used to train UAVs to learn decentralized collision avoidance behaviors under conflict graphs [9], [10]. Other methods, including recurrent neural networks [11], physics-informed neural networks [12], and model predictive control [13], have also been adopted to handle dynamic and physically constrained environments. However, most current learning-based approaches exhibit two limitations:

(a) Local decision structures: Many methods rely on local observers, which reduce computational overhead but cannot guarantee globally coordinated conflict resolution under high traffic density [14]–[16].
(b) Continuous policy updating: A large portion of RL-based conflict management methods are designed for fully online control, where agents continuously update or adapt their policies during operation based on streaming observations, and thereby introducing additional communication and computational costs [17].

Although some studies have explored learning-based methods for strategic scenarios [18], [19], they typically rely on fixed airspace structures and secondary policy updates, limiting adaptability to dynamic demand patterns. These limitations motivate the development of a learning-based approach that enables globally coordinated, single-stage, and computationally efficient decision-making for pre-tactical conflict management.

To overcome the above limitations, we propose a global learning-based conflict management framework for UTM pre-tactical conflict management. The proposed method extracts structured conflict information from submitted flight trajectories and outputs coordinated planning decisions to manage large-scale traffic efficiently. The main contributions of this paper are summarized as follows:

- We propose a novel pre-tactical conflict management architecture that centrally handles large-scale potential conflicts under mixed scheduled and on-demand demand in UTM.
- We design an adaptive dynamic structural airspace representation that is constructed as a conflict-based graph from a submitted free-flight plan, balancing trajectory flexibility and computational tractability.
- We implement a learning-based framework for conflict resolution, realizing a fast response as well as maintaining near-optimal performance.

The remainder of the paper is organized as follows. Section II presents the problem statement and the overall framework of pre-tactical conflict management in the UTM operational context. Section III introduces the proposed methodology, including adaptive airspace abstraction, conflict modeling, and the learning-based decision framework. Section IV describes the experimental setup and discusses the performance evaluation results. Section V concludes the paper and outlines future research directions.

## II. Problem Statement

This paper focuses on conflict management for UTM in city-scale airspace. We consider a UTM setting where multiple service providers submit flight requests for low-altitude operations over the urban area. A strategic layer already exists in the overall system, providing long-term planning, such as DCB and route allocation over a large time scale (e.g., days in advance).

However, as mentioned in Sec. I, UTM operations are characterized by a mixture of scheduled services (e.g., parcel delivery) and instant, on-demand requests (e.g., food delivery, emergency response). The submission time of such instant demands is relatively close to the planned departure time, making it difficult for the strategic layer to comprehensively re-plan them. Conflicts that cannot be resolved by the strategic layer in time are pushed to the tactical collision avoidance phase, leading to an increase in the workload of the tactical layer. This may further exceed the handling capacity of the tactical layer, which in turn raises computational burden and operational risk.

To address this issue, we focus on the pre-tactical phase, which operates within a short period before actual operations (e.g., from tens of minutes up to several hours before departure), when most flight requests, including instant demand, have already been collected. We assume that service providers submit their requests to a centralized airspace manager for system-level coordination. Each flight request contains a free-flight trajectory that satisfies basic airspace constraints, together with the corresponding scheduled departure time and flight speed. The objective of the pre-tactical layer is to perform demand–capacity balancing on the submitted free-flight plans, thereby reducing large-scale potential conflicts and alleviating the burden on the subsequent real-time tactical layer, while minimizing the disruptions to the originally requested plans and computational cost.

## III. Methodology

This section presents the proposed pre-tactical conflict management framework for on-demand UTM operations. Given a batch of submitted flight plans, a conflict graph is constructed to capture spatial–temporal interactions among trajectories. Each trajectory is subsequently abstracted into a sequence of nodes with associated arrival times. On this basis, the pre-tactical conflict management problem is formulated as a multi-agent contextual bandit, and a learning-based resolution policy is trained under CTDE scheme. The

overall workflow of the proposed Pre-tactical Learning-Based Conflict Management Framework is illustrated in Fig. 1.

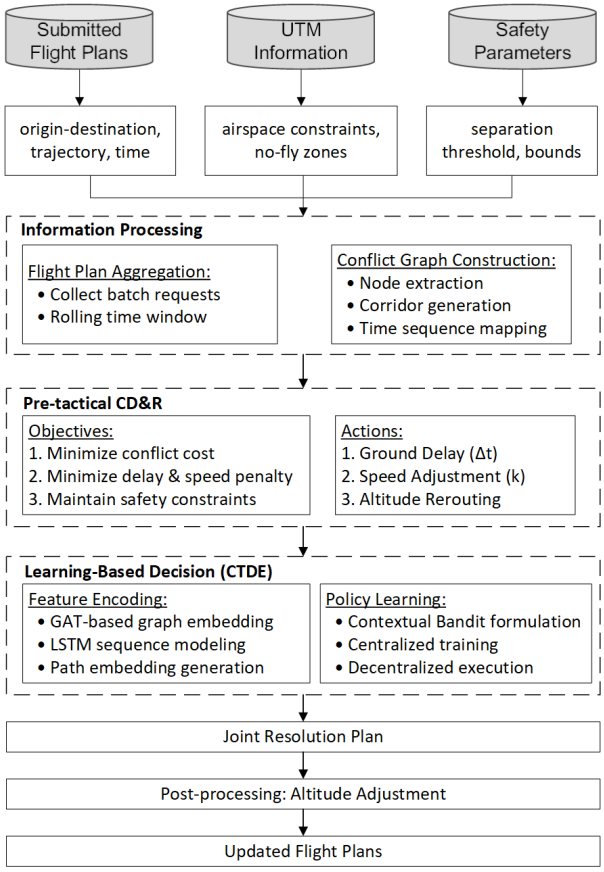



Figure 1. Pre-tactical Learning-Based Conflict Management Framework

### A. Conflict Graph Construction

The first stage of conflict management is airspace representation and conflict abstraction. Instead of assuming a predefined, ground-transportation-based airspace structure [20], [21], our framework operates directly on the free-flight trajectories submitted by operators. This preserves flight flexibility and allows the system to accommodate heterogeneous, on-demand requests. Given a batch of flight plans, we construct the conflict graph in two steps:

*1) Node and Link Generation:* As illustrated in Fig. 2(a), we define the graph node as:

- origin and destination points of each flight,
- intersection points between trajectories,
- the entry and exit points of shared segment.

Then each pair of adjacent nodes on one flight trajectory is modeled as a corridor.

To avoid redundant representations and preserve safety margins, spatially proximate nodes (node A in Fig. 2(a)) and nearly overlapping trajectory segments (segment B in Fig. 2(a)) are merged if their pairwise distance falls below a predefined threshold.

*2) Graph Abstraction and Flight Representation:* Based on the resulting node and link network, we represent the submitted flight trajectories as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as shown in Fig. 2(b), where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges (including shared corridors and regular corridors between nodes).

Each submitted flight plan is then abstracted into

- a node sequence $\{\{v_{i,j}\}_{j=0}^{N_i-1}\}_{i=0}^{N-1}$ describing the ordered nodes traversed by flight $i$,
- the corresponding arrival time sequences $\{\{t_{i,j}\}_{j=0}^{N_i-1}\}_{i=0}^{N-1}$.

Note that potential conflicts can only arise at intersection nodes (e.g., blue nodes in Fig. 2(b)) or within shared corridors (e.g., blue shadowed region in Fig. 2(b)). Therefore, pre-tactical conflict management only detects and adjusts the arrival times of each flight at each relevant node or corridor, which substantially simplifies both information processing and decision-making.

Time adjustment at the first node corresponds to ground delay, whereas time adjustments at subsequent nodes result from a combination of ground delay and speed adjustment. In addition, to prevent head-on conflicts between UAVs flying in opposite directions within the same corridor, a nearby altitude layer is reserved and can be used to implement an altitude-change maneuver when necessary.

### B. Conflict Definition

Based on the conflict graph, conflicts between flights can be categorized into two types: conflicts in intersection nodes, and conflicts within shared corridors. We define the corresponding time-based conflict conditions below.

*1) Conflicts in Intersection Nodes:* For intersection nodes, conflicts are determined by the temporal separation. Let $t_{i,j}$ and $t_{m,n}$ denote the arrival times of flights $i$ and $m$ at a shared node, where $j$ and $n$ are the indices of this node in their respective node sequences (note that $j \neq n$ in general). Then a conflict is defined as:

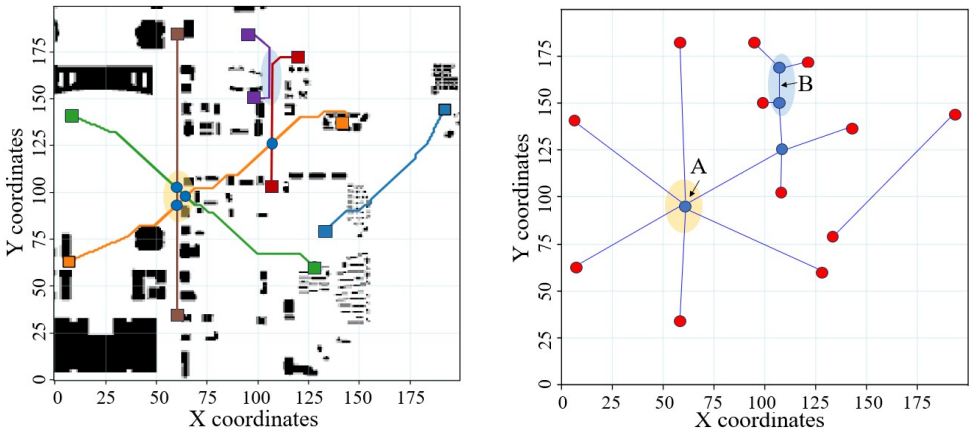
$$|t_{i,j} - t_{m,n}| \leq \delta, \tag{1}$$

where $\delta$ is the required minimum time separation, determined by the maximum absolute speed of flight $i$ and $m$.

Because conflicts can only arise at shared nodes or shared edges, using time-based separation as the primitive safety metric is sufficient for pre-tactical conflict management. Meanwhile, in dense free-flight airspace, continuous monitoring distance for all pairs of flights is computationally expensive, representing separation constraints via time thresholds provides an efficient and operationally meaningful approximation.

*2) Conflicts in Shared Corridors:* For a shared corridor, two flights may conflict at any position within the corridor. To ensure consistency, we extend the intersection conflict definition to shared corridor scenarios. For simplicity of analysis, we assume that each flight maintains a constant speed within the corridor. The framework can be extended to bounded speed profiles without altering the conflict structure.

Let $t_{i,1}$ and $t_{i,2}$ denote the entry and exit times of flight $i$ into and out of a corridor, respectively. Similarly, $t_{m,1}$ and




(a) Original flight trajectories.

(b) Abstracted undirected graph. *Red node means start or end point, and blue node means shared point.*

Figure 2. Illustration of airspace abstraction generation process.

$t_{m,2}$ represent the entry and exit times for flight $m$. Let $dir_i, dir_m \in \{0, 1\}$ denote the flight directions along the corridor. We can further categorize the conflicts into three types:

When the flights travel in the opposite direction (i.e., $dir_i \neq dir_m$), a conflict occurs whenever their time intervals of presence in the corridor overlap beyond the safety threshold, which is:

$$\begin{cases} t_{m,1} - t_{i,2} \leq \delta \\ t_{i,1} - t_{m,2} \leq \delta. \end{cases} \tag{2}$$

When $dir_i = dir_m$, a conflict may occur inside the corridor if the entry order and exit order are inverted due to different speeds:

$$((t_{i,1} < t_{m,1}) \wedge (t_{i,2} > t_{m,2})) \text{ or } ((t_{i,1} > t_{m,1}) \wedge (t_{i,2} < t_{m,2})). \tag{3}$$

In the meantime, same-direction conflicts may also arise at the entry or exit of the corridor:

$$|t_{i,1} - t_{m,1}| \leq \delta \text{ or } |t_{i,2} - t_{m,2}| \leq \delta. \tag{4}$$

These conditions generalize the node-based intersection conflict definition to extended corridors while preserving a time-based safety formulation suitable for pre-tactical planning.

### C. Contextual Bandit Modeling

With the conflict structure defined, the decision problem is next formulated as a contextual bandit. Pre-tactical conflict management is executed periodically (e.g., every tens of minutes to several hours) before operations, based on batches of on-demand flight requests. Within each planning horizon, a conflict resolution policy is computed once and does not update until the next batch of demands arrives. At the beginning of each new planning horizon, ongoing unfinished flights are represented as demand from their current positions to their destinations, and then re-integrated into the planning process, but the ground delay of these flights are fixed to be zero (i.e., continue ongoing flight). This setting differs from tactical conflict management, which is typically modeled as a Markov Decision Process (MDP) and requires continuous policy update. In the pre-tactical phase, the decision for a given batch does not influence the state of subsequent planning horizons; thus, there is no inter-horizon state transition induced by the actions. Therefore, we model the pre-tactical conflict management problem as a multi-agent contextual bandit.

Let $\mathcal{N} = \{1, 2, \ldots, N\}$ denote the set of flights (agents). At the beginning of a planning phase, a global state $S$ is observed, which includes all submitted flight plans and the derived conflict graph. For each agent $i \in \mathcal{N}$, it obtains an observation $o_i$ derived from global state $S$ and then samples an action $a_i$ from its action space $A_i$. After that, the joint action $A = \{a_i\}_{i \in \mathcal{N}}$ is evaluated by a global reward function $R: S \times A \to \mathbb{R}$. In each planning horizon, *i.e.*, a batch of flights to be scheduled, the learning-based model only output the conflict resolution policy once. In next planning horizon, the model will deal with new batch of flights and the initial state would be different from the resolved state in this horizon. Therefore, the state transition distribution $P$ is independent of the action $A$, and the reward $R$ does not depend on any subsequent states or actions. The detailed components of the multi-agent contextual bandit model are specified as follows:

*1) Agent:* Each flight is modeled as an agent. Each agent is associated with a unique flight plan, represented by its traversed node sequence and corresponding arrival times. All agents act simultaneously in each conflict management phase.

*2) Observation:* In the pre-tactical phase, the model performs system-wide planning. Consequently, each agent can be provided with global information derived from the full state $S$. Specifically, the agent's observation information consists of three indicators: the traversed node index lists of all flights $\{\{node_{i,j}\}_{j=0}^{N_i-1}\}_{i=0}^{N-1}$, the corresponding time lists $\{\{t_{i,j}\}_{j=0}^{N_i-1}\}_{i=0}^{N-1}$, and the undirected graph $\mathcal{G}$ composed of all flight trajectories. Here, $N$ is the total number of flights, $N_i$ is the number of nodes traversed by the $i$-th flight.

*3) Action:* In each conflict resolution phase, agent $i$ samples an action $a_i \in A_i$. The action space $A_i$ can be further defined as a nonnegative ground delay $\Delta t_i \geq 0$, a speed adjustment factor $k_i$ for each agent. The agent, *i.e.*, the

learning-based method, can execute a combination of all the above actions. As illustrated in Fig. 3, the ground delay shifts the entire speed–time profile along the time axis, whereas the speed factor $k_i$ scales the profile in time. To ensure safety, if head-on conflicts persist in a corridor after time and speed adjustments, the altitude-routing decision $dir_{edge,t}$ is used to reroute some flights to an adjacent altitude layer, effectively decoupling remaining opposite-direction flows (the detailed policy is shown in the following subsection).

*4) Reward:* The performance of the conflict resolution policy is evaluated by the return value. Since the pre-tactical conflict management aims to minimize the overall cost, all agents share the same return value. Defining $S$ as the environment states, $A = \{\Delta t_i, k_i\}$ as the set of action taken by learning-based method and $\Delta H_U = \{\Delta h_u(t)\}$ represents the set of altitude adjustments, we can define the system's reward as:

$$\begin{aligned} R(S, A, \Delta H_U) =& r_{\text{conflict}}(S, A, \Delta H_U) + r_{\text{delay}}(A) \\ &+ r_{\text{speed}}(A) + r_{\text{altitude}}(\Delta H_U), \end{aligned} \tag{5}$$

where,

- $r_{\text{conflict}}(S, A, \Delta H_U) = C_{\text{original}}(S) - C_{\text{resolved}}(S')$ represents reduction in conflict cost of the whole system, where $C(S) = \sum_{i=0}^{N_c-1} f(\delta t_i)$ denotes the conflict cost, $N_c$ is the number of conflicts, $\delta t_i$ is the minimum time gap between corresponding pair of conflicting flights when arriving the same node or inside the same corridor, and $S'$ denotes the global state after resolution. The function $f(\delta t_i)$ is monotonically decreasing in $\delta t_i$, assigning higher penalties to smaller separation distances.
- $r_{\text{delay}}(A) = -k_{\text{delay}} \sum_{i=0}^{N-1} \Delta t_i$ is the total time cost of the ground delay for all flights, where $k_{\text{delay}}$ is the delay cost coefficient.
- $r_{\text{speed}}(A) = -k_{\text{speed}} \sum_{i=0}^{N-1} |k_i - 1.0|$ quantifies the speed changes of all flights to avoid sharp acceleration or deceleration, where $k_{\text{speed}}$ is the speed adjustment cost coefficient.
- $r_{\text{altitude}}(\Delta H_U) = \sum r_{\text{altitude}}(\Delta h_u(t))$ denotes the total cost of rerouting, which is linearly correlated with the number of flights that needs to be rerouted to adjacent altitudes. Although the rerouting reward is not directly determined by $A$, the altitude adjustment actions $\Delta h_u(t) \in \{0, 1\}$ are determined by $A$ and then affect the rerouting reward.

This reward design explicitly balances safety (conflict reduction) with operational efficiency (delay, speed changes, and route stability), which is central to transportation system optimization. Note that the altitude adjustment is a postprocessing action which are not the output of learning-based method. However, as the altitude adjustment would affect the overall action cost, reward of the overall conflict resolution policy also considers the altitude adjustment action, *i.e.*, the learning-based method views the altitude adjustment as a fixed resolution policy.

With the above modeling, we can abstract the pre-tactical conflict resolution problem as the following optimization problem:

$$\max_{A} \quad R(S, A, \Delta H_U) \tag{6a}$$

$$\text{s.t.} \quad 0 \le \Delta t_i \le \Delta t_{\max}, \tag{6b}$$

$$k_{\min} \le k_i \le k_{\max}, \tag{6c}$$

$$\Delta H_U = \underset{\Delta h_u \in \{0,1\}}{\operatorname{argmax}} \; [r_{\text{conflict}}(S, A, \Delta H_U) + r_{\text{altitude}}(\Delta H_U)]. \tag{6d}$$

In the learning process, the model learns to maximize the overall reward by output ground delay and speed adjustment for each agent. Since the altitude adjustment is a postprocessing action and determined by model's action, the model can learn to operate with altitude adjustment policy.

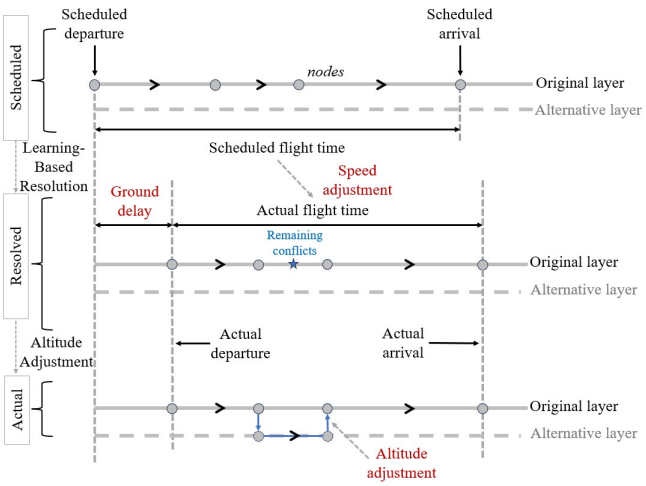



Figure 3. Mixed action consists of ground delay, speed adjustment and altitude adjustment.

### D. CTDE Learning-Based Framework

Given the contextual bandit formulation, we design a learning-based conflict resolution policy under a CTDE framework. To improve training stability and model scalability, we implement the Advantage Actor-Critic (A2C) algorithm to learn the conflict resolution policy. In A2C algorithm, an actor is applied for each agent to output the conflict resolution policy and In A2C algorithm, the actor outputs the conflict resolution policy and the critic estimates the overall performance of the policy. During training, the critic receives the global state $S$ and the joint action $A$ and outputs a value estimate $V^{\pi}(S)$. Thus, the advantage function can be defined as:

$$Adv^{\pi}(S, A) = R(S, A) - V^{\pi}(S), \tag{7}$$

which measures how much better (or worse) the realized reward $R(S, A)$ is compared to the expected value $V^{\pi}(S)$ under policy $\pi$. As the altitude adjustment $\Delta H_U$ is determined by model action $A$, here we simplify $R(S, A, \Delta H_U)$ as $R(S, A)$. Under the contextual bandit setting, the advantage function depends solely on the current state–action pair.

The parameters of actors and the critic are updated via the following loss functions.

$$J(\pi) = \mathbb{E}\big[\log \pi(A|S) \cdot Adv^{\pi}(S, A)\big], \tag{8}$$

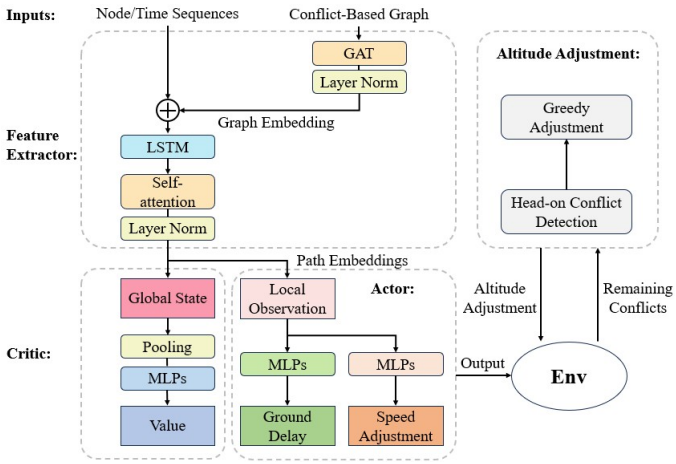



Figure 4. The structure of the learning-based algorithm.

$$L(\phi) = \mathbb{E}\big[\frac{1}{2}(R(S,A) - V^{\pi}(S))^2\big], \tag{9}$$

where $\log \pi(A|S)$ is the log-likelihood of the joint action under the current policy $\pi$, and $R(S,A) - V^{\pi}(S)$ represents the estimation error of the critic. After the model converges, only the actor is used to output conflict resolution policy.

A key challenge in pre-tactical conflict management is that, in different planning phases, both the number of flights and the length of each flight's path vary, leading to a dynamically changing model input. To address this issue, we employ the Graph Attention Network (GAT) and Long Short-Term Memory (LSTM) to process the undirected graph $\mathcal{G}$ and flight sequences (i.e., nodes and time sequences), respectively. Both GAT and LSTM can naturally handle variable-size and variable-length inputs, making the framework scalable across different traffic densities and network topologies. The detailed structure of the learning-based model is illustrated in the upper part of Fig. 4, which has three main modules:

*1) Feature Extractor:* This module processes the global state $S$ and produces path-level embeddings for each flight. Graph information is first encoded by GAT, and then combined with the corresponding time information. The combined information is mapped to path-level embeddings via LSTM, so that each path embedding incorporates both local trajectory information and global structural information of $\mathcal{G}$.

*2) Actor:* For flight $i$, the actor takes its own path-level embedding and a pooled embedding of all other flights as input, together with conflict-related features (e.g., local conflict density or degree). Then, it outputs the resolution action.

*3) Critic:* The critic aggregates all path-level embeddings into a global representation and outputs the estimated value $V^{\pi}(S)$ of the current policy for the given planning horizon.

After the learning-based actions are applied (i.e., ground delays and speed adjustments), a final conflict detection step is performed, with particular attention to head-on conflicts in shared corridors (shown in the lower part of Fig. 4). As mentioned in Sec. III-B, when more than two flights pass through the same corridor, purely time-based adjustments may require large delays or speed changes to ensure non-overlapping presence intervals. In such cases, these time adjustments become economically inefficient. To maintain safety while limiting additional cost, we introduce a greedy altitude-adjustment post-processing step after executing learning-based actions: For each corridor and each overlapping time interval with residual head-on conflicts, the algorithm greedily reassigned flights in the direction with fewer aircraft to an adjacent altitude layer. The corresponding reroute cost is then added to $r_{\text{reroute}}(\Delta H_U)$ in the reward function. Given the relatively low frequency of persistent head-on conflicts in free-flight environments, this greedy post-processing step has limited

**Algorithm 1** Training of Pre-tactical Conflict Management Policy in Contextual Bandit Setting

**Require:** Training environment providing batches of flight plans, safety threshold, learning rates $\alpha_\pi$, $\alpha_V$

Initialize actor parameters $\theta$ and critic parameters $\phi$
**for** each training episode **do**
  Generate a batch of flight plans
  Construct conflict graph $\mathcal{G}$ and node/time sequences
  **Feature Extraction**
  Obtain node-level graph embeddings with GAT
  **for** each flight $i$ **do**
    Gather node embeddings and time sequences
    Obtain a path-level embedding $h_i$ with LSTM
  **end for**
  **Action Sampling**
  **for** each flight $i$ **do**
    Aggregate local embedding and embeddings of other flights $z_i = [h_i,\ f(h_{-i}),\ \text{conflict\_features}_i]$
    Sample an action from the current conflict resolution policy
    $$(\Delta t_i, k_i) \sim \pi_\theta(\cdot \mid z_i)$$
  **end for**
  Denote the joint action as $A = \{a_i\}_{i=0}^{N-1}$
  Estimate value $V^{\pi_\theta}(S)$ using critic $V_\phi$
  Apply joint action to the environment
  **Head-on Conflicts Resolution**
  Detect remaining head-on conflicts
  **for** each corridor with remaining head-on conflicts **do**
    Obtain altitude adjustment actions greedily according to the number of flights in each direction
  **end for**
  Apply altitude adjustment to the environment
  **Model Update**
  Compute the global reward $R(S,A)$ using (5)
  Compute advantage using (7)
  Update actor parameters via gradient ascent:
  $$\theta \leftarrow \theta + \alpha_\pi \nabla_\theta \big(\log \pi_\theta(A \mid S) \cdot Adv^{\pi}(S,A)\big)$$
  Update critic parameters via regression to the one-step return:
  $$\phi \leftarrow \phi - \alpha_V \nabla_\phi \big(\tfrac{1}{2}(R(S,A) - V_\phi(S))^2\big)$$
**end for**
**return** Trained actor $\pi_\theta$

impact on global optimality while ensuring safety. At the same time, it keeps computational complexity manageable.

The detailed process is described in Algorithm. 1.

## IV. Experiments And Analysis

This section evaluates the proposed learning-based pre-tactical conflict management framework in a simulated urban delivery scenario. We first describe the simulation environment and conflict characteristics. Then the training procedure of the learning-based policy in the simulation environment is presented. Finally, we benchmark the learning-based algorithm against other conflict resolution methods and analyze its generalizability under different traffic demand densities and initial conditions.

### A. Simulation Environment

*1) Airspace and Trajectory Generation:* The simulation environment is constructed over a $2$ km $\times$ $2$ km urban region in the central district of Shenzhen. The horizontal airspace is discretized into $10$ m $\times$ $10$ m basic blocks. The tall buildings and no-fly zone are modeled as the non-traversable areas, and UAVs are allowed to freely fly in the remaining area (free-flight airspace).

For a given number of flights, origin–destination (OD) pairs are randomly generated, subject to a minimum Euclidean distance of $600$ m between origin and destination. This lower bound ensures non-trivial path lengths and increases the likelihood of potential conflicts. Given each OD pair, a flight trajectory is generated using the $A^*$ algorithm, which simulates route choice in a free-flight environment. Departure times are independently sampled from a uniform distribution over $[0, 10]$ min, and the average flight speed is set to $1$ m/s.

As described in Sec. III-A, all trajectories are then abstracted into an undirected conflict graph, and each flight plan is mapped to its corresponding node sequence and arrival time sequence (Fig. 2). In practice, the proposed algorithm directly operates on any submitted plans. The graph-based abstraction is independent of the particular urban layout or trajectory generation method. Therefore, the proposed framework can be applied to different cities and airspace configurations as long as flight plans can be mapped to node–time sequences.

*2) Conflict Characteristics:* According to the conflict definitions in Sec. III-B, we set the safety time threshold under flight speed $1$ m/s to $60$ s. That is, two flights passing through the same conflict node or shared corridor must be separated by more than $60$ s in time to avoid a conflict.

Fig. 5 visualizes the initial conflict distributions for different demand densities. It is evident that the conflict density grows sharply as the number of flights increases. Conflicts concentrate in corridors formed by building constraints and the central area of the airspace, in which the traffic density is high. This indicates that the simulation setup can generate congestion-prone urban air traffic that is challenging for pre-tactical conflict management.

### B. Model Training and Evaluation

With the above environment, we train the proposed CTDE learning-based framework to obtain the conflict resolution policy.

To ensure operational realism, we constrain the ground delay as $\Delta t_i \in [0, 300]s$ and the speed adjustment factor as $k_i \in [0.5, 1.5]$. The actor output Beta distribution within the corresponding range and sample the conflict resolution action. Altitude adjustments follow the head-on conflicts resolution module introduced in Sec. III-D and are identical across all compared methods in the experiments. A relatively high weight is assigned to the conflict reduction term $r_{\text{conflict}}$ in the reward function (5), biasing the learned policy towards strong safety performance and reflecting the safety-first principle in transportation operations.

In the training process, the model outputs a joint action (ground delays and speed adjustments for all flights), and the environment returns the immediate global reward. The simulation environment is then reset to the initial state. Policy parameters are updated every 50 steps using the mean reward over these steps; we therefore refer to 50 steps as one episode. Under the contextual bandit setting, the objective is to maximize the expected one-step return, which coincides with the global reward.

Fig. 7 shows the training return for the scenario with 50 flights taking off in 10 min. The learning curve exhibits stable improvement and convergence, and the policy converges to an optimal solution after approximately $5000$ episodes.

Fig. 8(a) depicts the remaining conflict percentage over episodes. When the model converges to its optimal, all conflicts in the airspace are successfully resolved by the learned policy. Fig. 8(b) shows the action cost $|r_{\text{delay}}(A) + r_{\text{speed}}(A) + r_{\text{altitude}}(A)|$, which reflects the operational penalty paid to achieve conflict resolution.

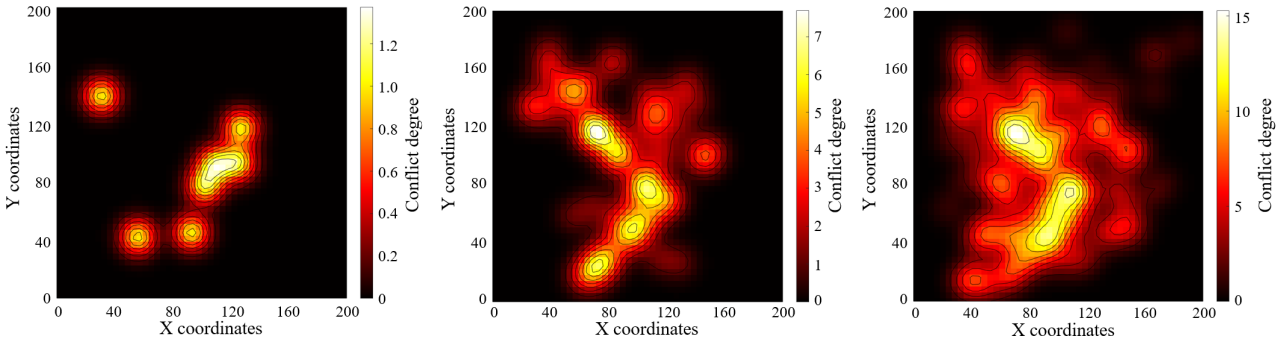
At the beginning of training, the model's actions are close to random sampling in the overall action space. Hence, it frequently takes large adjustments without effectively resolving conflicts, resulting in high action cost and low reward. As training progresses, the model learns the valid resolution policy, and both the number of remaining conflicts and the action cost decrease. After all conflicts are resolved (around 5000 episodes), the policy continues to explore alternative feasible actions that further reduce the action cost.

### C. Algorithm Test

*1) Test with Other Methods:* To evaluate the effectiveness of the learned policy, we compare it with the First-Come-First-Serve (FCFS) method. FCFS resolves conflicts according to departure time order and generates mixed actions consisting of ground delays and speed adjustments, subject to the same bounds as in Sec. IV-B.

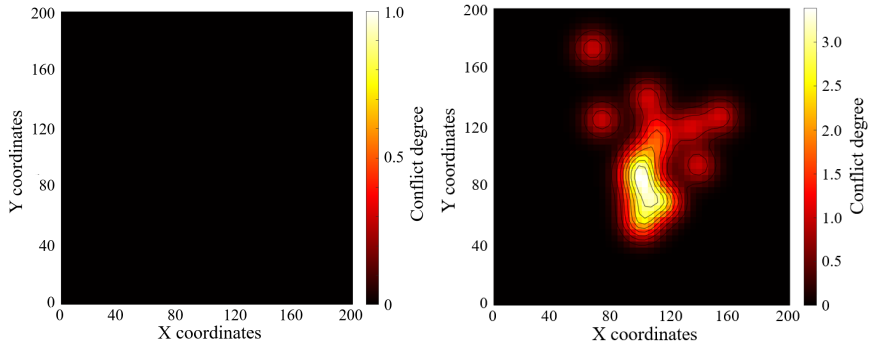
To further show the conflict resolution performance of the algorithm, we consider three demand levels with 20, 50 and 80 flights within a 10-minute horizon, corresponding to small-, medium- and large-scale conflict situations. The initial numbers of conflicts are 7, 86, and 241, respectively.

In the meantime, to quantify the solution quality compared with the global optimal solution, we additionally formulate an optimization model that maximizes the same reward function as the learning-based approach under the same action bounds, as shown in Eq. 6. The problem is then solved using Gurobi on AMD R7 9700X. The time limitation is set to 60 min.




(a) 20 flights in 10 min (b) 50 flights in 10 min (c) 80 flights in 10 min

Figure 5. Initial conflicts heat maps under different demand densities.




(a) 20 & 50 flights in 10 min. *The conflicts have been completely resolved.* (b) 80 flights in 10 min

Figure 6. Conflicts heat maps after resolution under different demand densities.

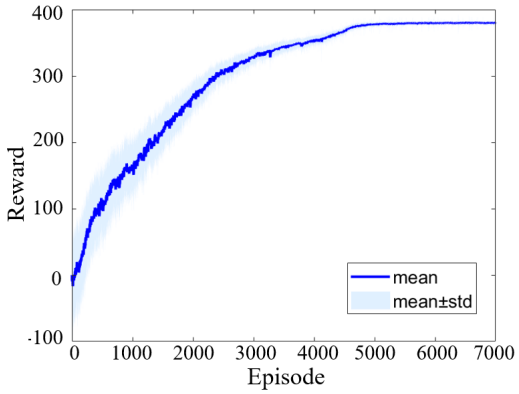



Figure 7. Return curve of training process (50 flights in 10 min).

Table I summarizes the performance of the proposed learning-based method, the FCFS baseline, and the optimization-based method.

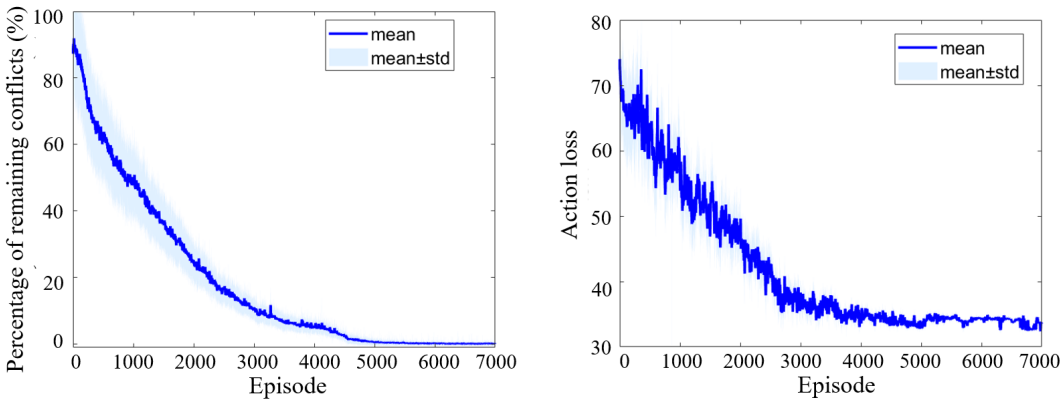
In the 20-flight scenario, all methods can resolve almost all conflicts. The learning-based algorithm achieves a final reward close to the global optimum given by the optimization-based method, while the FCFS strategies incur much higher action cost, leading to a lower overall reward. This demonstrates that the learned policy effectively coordinates multiple control variables to exploit the feasible solution space, whereas rule-based methods struggle to balance safety and operational efficiency. However, in small-scale conflict situation, the learning-based model does not show great potential in time consumption,

In the 50-flight scenario, the learning-based method remains close to the global optimum, indicating near-optimal performance in medium-scale conflict situations. In contrast, the FCFS strategies fail to eliminate all conflicts, leading to substantially lower rewards.

In the 80-flight scenario, the conflict situation becomes highly congested. The optimization-based solver fails to find a globally optimal solution that resolves all conflicts within 60 min. The learning-based approach successfully resolves more than 90% of conflicts, significantly outperforming the FCFS baselines, which leave a large number of conflicts unresolved.

The time consumption demonstrates that the learning-based approach can transfer the majority of computational effort to the training phase, thereby maximizing response speed in the execution phase. In medium- and large-scale conflict situations, comparing to optimization method, the increase in training time remains within a controllablle range, while the execution speed shows a good advantage.

Fig. 6 further shows the distributions of remaining conflicts in different demand densities after learning-based resolution. Compared with the initial conflict distributions in Fig. 5, the conflicts in 20- and 50-flight scenario have been fully resolved. While in the 80-flight scenario, the maximum conflict degree drops from 15 to around 3, and large areas




(a) Remaining conflict percentage curve.

(b) Action cost curve.

Figure 8. Conflict resolution performance of training process (50 flights in 10 min).

TABLE I. Conflict resolution performance with different resolution methods.

| Method | Flights | Initial Conflicts | Total Reward | Conflict Resolution Rate (%) | Time Consumed |
|---|---|---|---|---|---|
| Optimization | 20 | 7 | 30.2 | 100.0 | 0.2s |
| FCFS | | | 25.1 | 100.0 | 0.01s |
| Learning-Based Model | | | 30.0 | 100.0 | 5min (training); 0.01s (execution) |
| Optimization | 50 | 86 | 385.8 | 100.0 | 27min |
| FCFS | | | 325.0 | 90.7 | 0.42s |
| Learning-Based Model | | | 380.0 | 100.0 | 1h (training); 0.013s (execution) |
| Optimization | 80 | 241 | — | — | — |
| FCFS | | | 396.0 | 51.0 | 3.23s |
| Learning-Based Model | | | 856.0 | 91.3 | 2.5h (training); 0.02s (execution) |

*Note:* "—" indicates that the optimal solution was not found within the 60-minute time limit.

that were previously congested become conflict-free. Residual conflicts are confined to a few localized regions around a single dominant corridor, and their intensity is much lower than in the initial state.

*2) Test over Different Initial States:* A practical pre-tactical conflict management system deals with varying conflicts depending on the submitted batch of flight plans, which requires the system to be capable of handling varying OD distributions and conflict patterns. To verify this, we generate five independent test scenarios with completely new OD pairs, trajectories, and departure times for each demand level (20, 50, and 80 flights in 10 min). The conflict graphs and initial conflict numbers differ across these scenarios, even when the total number of flights is the same.

The test results in Table. II show that, in the 20- and 50-flight scenarios, the learning-based method can almost resolve 100% of conflicts across all cases. The only case that fails to achieve 100% resolution rate in 50-flight scenario has a significantly higher number of conflicts than other cases, approaching the large-scale conflict scenario. In the 80-flight scenario, the conflict resolution rates are around 90%. Compared with the resolution performances of using FCFS method, the learning-based algorithm shows a more stable and better conflict resolution performance.

To further place the heterogeneous cases on a comparable basis, we evaluate the conflict resolution performances with the optimization-based benchmark. For each case where the Gurobi solver returns an optimal solution within the time limit

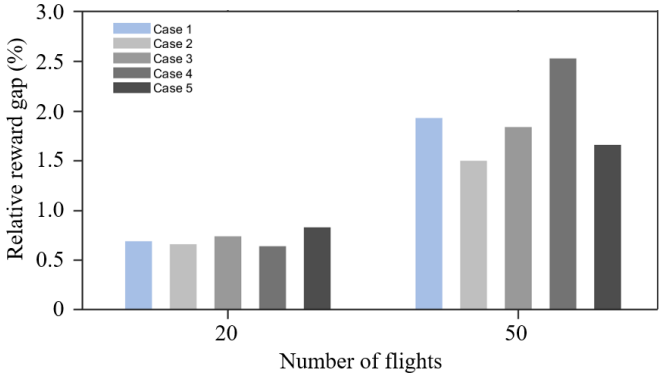



Figure 9. Relative reward gaps to optimization-based solution under different initial cases.

(i.e., all 20- and 50-flight scenarios), we compute the relative reward gap as

$$\text{Relative Reward Gap} = \frac{R_{\text{opt}} - R_{\text{RL}}}{R_{\text{opt}}} \times 100\%,$$

where $R_{\text{opt}}$ and $R_{\text{RL}}$ denote the total rewards of the optimization-based and learning-based methods, respectively. The relative reward gap for each case is shown in Fig. 9. It shows that, in the scenarios with same number of flights (i.e., similar conflict scale), the learning-based algorithm maintains a stable near-optimality level. Only Case 4 in 50-flight scenario, which has the highest initial conflict count in

TABLE II. Conflict resolution performance under different cases

| Test Cases | | Initial Conflicts | Total Reward | Conflict Resolution Rate (%) | Conflict Resolution Rate under FCFS (%) |
|---|---|---|---|---|---|
| 20 Flights | Case 1 | 5 | 21.4 | 100 | 100 |
| | Case 2 | 5 | 20.2 | 100 | 100 |
| | Case 3 | 6 | 22.6 | 100 | 100 |
| | Case 4 | 7 | 30.0 | 100 | 100 |
| | Case 5 | 13 | 58.7 | 100 | 100 |
| 50 Flights | Case 1 | 53 | 208 | 100 | 84.9 |
| | Case 2 | 57 | 223 | 100 | 68.4 |
| | Case 3 | 75 | 304 | 100 | 84.0 |
| | Case 4 | 86 | 380 | 100 | 90.7 |
| | Case 5 | 137 | 519 | 98.5 | 71.5 |
| 80 Flights | Case 1 | 175 | 580 | 88 | 59.4 |
| | Case 2 | 202 | 727 | 92.1 | 64.9 |
| | Case 3 | 227 | 835 | 92.5 | 69.2 |
| | Case 4 | 241 | 865 | 91.3 | 51.0 |
| | Case 5 | 293 | 1035 | 87 | 59.0 |

its group, exhibits a slightly larger gap than the other cases. This indicates that the learning-based algorithm maintains stable resolution performance across different realizations of urban traffic with the same demand level.

## V. Conclusions

This paper proposes a learning-based pre-tactical conflict management framework for UTM with mixed scheduled and instant flight demands. The pre-tactical layer is located between the strategic and tactical layers. This layer uses full knowledge of submitted plans to resolve conflicts before flights enter the airspace, thereby reducing real-time tactical complexity and risk. In detailed implementation, we proposed a conflict-graph based abstraction that is constructed directly from submitted free-flight trajectories, without relying on pre-defined route networks. Then the optimization problem is modeled as a multi-agent contextual bandit. A learning-based model is implemented to learn the globally coordinated and scalable conflict resolution policy. Finally, the greedy altitude-adjustment post-processor specifically handles residual head-on conflicts. Numerical experiments show that the proposed learning-based method resolves all conflicts with final rewards close to the optimization-based method in small- and medium-scale conflicts. Even in large-scale conflicts, the learned policy still resolves around 90% of conflicts, showing high effectiveness in conflict resolution.

In general, this paper attempts to connect the pre-tactical conflict management architecture with a global, learning-based decision mechanism, balancing rapid decision-making and flight safety. But this study has several limitations that point to future research directions. The simulations rely on a fixed 2D urban geometry. Uncertainties in wind, navigation errors, and communication delays are not modeled. Extending the framework to 3D trajectory planning with stochastic disturbances is essential for more realistic UTM deployment. Meanwhile, the reward function uses fixed, hand-tuned weights to all flights. Other approaches, such as calibrating weights from operational preferences, and introducing fairness or priority between operators, would be further adopted to improve the acceptability in transportation practice.


## Acknowledgment

This study was supported by the Research Center for Low-Altitude Economy (RCLAE), The Hong Kong Polytechnic University (P0058170).


## References


[1] International Civil Aviation Organization (ICAO). Annex 11 to the convention on international civil aviation - Air traffic services, 2011.

[2] Li, A., Hansen, M., & Zou, B. Traffic management and resource allocation for UAV-based parcel delivery in low-altitude urban space. Transportation Research Part C: Emerging Technologies. vol. 143, 103808, August 2022.

[3] Tang, Y., Xu, Y., Lv, R., & Inalhan, G. Conflict probability based strategic conflict resolution for UAS traffic management considering Reasonable-Time-To-Act principle. Transportation Research Part C: Emerging Technologies, vol. 179, 105276, July 2025.

[4] Shi, Z., & Ng, W. K. A collision-free path planning algorithm for unmanned aerial vehicle delivery. In 2018 international conference on unmanned aircraft systems (ICUAS). pp. 358-362, June 2018.

[5] Isufaj, R., Omeri, M., & Piera, M. A. Multi-UAV conflict resolution with graph convolutional reinforcement learning. Applied Sciences, 12, no. 2: 610 , January 2022.

[6] Li, Y., Zhang, Y., Guo, T., Liu, Y., Lv, Y., & Du, W. Graph reinforcement learning for multi-aircraft conflict resolution. IEEE Transactions on Intelligent Vehicles, vol. 9, no. 3, pp. 4529-4540, March 2024 .

[7] Zheng, Y., Li, Y., Cheng, J., Li, C., & Hu, S. Two-Stage Hierarchical 4D Low-Risk Trajectory Planning for Urban Air Logistics. Drones 9, no. 4: 26, March 2025.

[8] Du, S., Zhong, G., Wang, F., Wu, L., Zhang, H., & Xue, D. A framework for collaborative UAM traffic flow optimization with mission preferences: Incorporating customized strategy synergy into strategic conflict management. Transportation Research Part E: Logistics and Transportation Review, vol. 202, 104326, July 2025.

[9] Li, Y., Li, J., Wang, J., Zhang, X., Ding, H., & Du, W. Multi-scale graph enhanced reinforcement learning for conflict resolution in dense UAV networks. IEEE Internet of Things Journal. vol. 12, no. 21, pp. 44290-44303, November 2025.

[10] Sui, D., Zhou, Z., & Cui, X. Priority-based intelligent resolution method of multi-aircraft flight conflicts. The Aeronautical Journal, vol. 129, no. 1332, pp. 326–350, October 2025.

[11] Deniz, S., Wu, Y., Shi, Y., & Wang, Z. A reinforcement learning approach to vehicle coordination for structured advanced air mobility. Green Energy and Intelligent Transportation, vol. 3, no. 2, 100157, April 2024.

[12] Zhao, P., & Liu, Y. Physics informed deep reinforcement learning for aircraft conflict resolution. IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 8288-8301, July 2022.

[13] Richards, A., & How, J. P. Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility. In Proceedings of the 2003 American Control Conference, vol. 5, pp. 4034-4040, June 2003.

[14] El-Tantawy, S., Abdulhai, B., & Abdelgawad, H. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto. IEEE transactions on Intelligent transportation systems, vol. 14, no. 3, pp. 1140-1150, September. 2013.

[15] Omidshafiei, S., Pazis, J., Amato, C., How, J. P., & Vian, J. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In International conference on machine learning. PMLR, pp. 2681-2690, July 2017.

[16] Gupta, J. K., Egorov, M., & Kochenderfer, M. Cooperative multi-agent control using deep reinforcement learning. In International conference on autonomous agents and multiagent systems. Cham: Springer International Publishing, vol 10642, pp. 66-83, November 2017.

[17] Brittain, M. W., Alvarez, L. E., & Breeden, K. Improving autonomous separation assurance through distributed reinforcement learning with attention networks. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, No. 21, pp. 22857-22863, March 2024.

[18] Huang, C., Petrunin, I., & Tsourdos, A. Strategic conflict management for performance-based urban air mobility operations with multi-agent reinforcement learning. In 2022 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 442-451, June 2022.

[19] Mas-Pujol, S., Salami, E., & Pastor, E. Image-based multi-agent reinforcement learning for demand–capacity balancing. Aerospace 9, no. 10: 599, October 2022 .

[20] Bertram, J., Zambreno, J., & Wei, P. Efficient unmanned aerial systems navigation with collision avoidance in dense urban environments. IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 8, pp. 8163-8173, August 2023 .

[21] He, X., Li, L., Mo, Y., Sun, Z., & Qin, S. J. Air corridor planning for urban drone delivery: Complexity analysis and comparison via multi-commodity network flow and graph search. Transportation Research Part E: Logistics and Transportation Review, vol. 193, 103859, January 2025.