

Article

Research on Route Optimization for Truck–Drone Delivery Considering En Route Synchronization

Shukang Zheng ¹, Genhua Ma ^{2,*}, Hanpei Yang ¹, Zichen Du ³, Ye Lu ³ and Yinjia Chen ³

¹ College of Environment, Hohai University, Nanjing 210024, China; shukangzheng_hhu@outlook.com (S.Z.); yanghanpei@hhu.edu.cn (H.Y.)

² College of Harbour, Coastal and Offshore Engineering, Hohai University, Nanjing 210024, China

³ College of Civil and Transportation Engineering, Hohai University, Nanjing 210024, China; 231304050005@hhu.edu.cn (Z.D.); ly18136309066@outlook.com (Y.L.); cyj199839@outlook.com (Y.C.)

* Correspondence: magenghua@hhu.edu.cn

Abstract

Truck–UAV collaborative delivery can improve last-mile logistics efficiency, but fixed-node rendezvous often causes waiting loss and service delay. To address this problem, this paper proposes a route optimization method integrating en route synchronization, pseudo-node insertion, and GAT-PPO. Pseudo-nodes are generated along truck travel arcs to provide flexible UAV recovery points, and a time-recursive simulation model is developed to evaluate makespan and total tardiness under soft time windows. In the proposed framework, GAT is used to capture spatial–temporal relationships among nodes, while PPO supports sequential routing decisions and UAV dispatch coordination. Experiments on Solomon VRPTW instances with clustered, random, and mixed customer distributions show that GAT-PPO achieves the shortest total travel distance, the lowest total tardiness, and the shortest completion time among Random, NN, NN+2-opt, MLP-PPO, ALNS, GA, and VNS. Ablation results further confirm the contributions of GAT, PPO, pseudo-node insertion, en route synchronization, and UAV collaboration. The results indicate that the proposed framework can effectively reduce synchronization waiting loss and improve the temporal efficiency of truck–UAV collaborative delivery.

Keywords: truck–drone; graph attention network; proximal policy optimization; vehicle Routing Problem with Time Windows (VRPTW); pseudo-node; en route synchronization

1. Introduction

Against the background of the digital economy and the high-quality transformation of urban logistics in China, last-mile delivery has become one of the most pressure-intensive links in the logistics system. In 2025, China’s express delivery volume exceeded 180 billion parcels for the first time, reflecting the rapid expansion of parcel demand and the increasing operational pressure on terminal distribution networks. Meanwhile, the 14th Five-Year Plan for Modern Logistics Development emphasizes the intelligent transformation of logistics systems and encourages the application of smart and unmanned logistics technologies. Traditional single-mode delivery systems are increasingly constrained by road congestion, service-time requirements, and limited access to dispersed customers. Truck–UAV collaborative delivery provides a promising solution by combining the truck’s endurance and payload capacity with the UAV’s mobility and flexible access capability. However, its operational efficiency depends strongly on truck–UAV synchronization. In particular, when UAV recovery is restricted to fixed physical nodes, the truck or UAV may have to



Academic Editor: Arkadiusz Gola

Received: 15 June 2026

Revised: 12 July 2026

Accepted: 22 July 2026

Published: 4 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

wait for the other vehicle, and such waiting loss can propagate to subsequent customers through temporal recursion, thereby increasing total tardiness and system makespan.

UAV-based logistics delivery has primarily emerged in response to the demand for lightweight parcel transportation in last-mile logistics. Practical UAV delivery developed alongside pilot operations by DHL, UPS, and Google Wing, demonstrating the feasibility of using small aerial vehicles for selected last-mile tasks [1–3]. Academic work subsequently broadened the problem from individual UAV navigation to multi-UAV cooperation and healthcare delivery [4–6]. Specialized and emerging neural routing settings further illustrate the expanding scope of UAV-related routing research [7]. For integrated ground–air delivery, the flying sidekick problem, drone-aided routing reviews, and traveling-salesman-with-drone formulations established core launch, service, recovery, and route-partitioning concepts [8–10]. Further analytical and coordinated-routing studies examined performance bounds, tandem truck–drone networks, and the effect of relative vehicle movement [11–13]. Later work incorporated time windows and two-echelon cooperation [14,15]. Other extensions addressed multiple drones, truck-and-drone fleets, and broader vehicle-routing structures [16–18]. Multiple vehicles and dynamic requests were subsequently considered [19,20]. Collectively, this literature provides the modeling basis for truck–UAV collaborative delivery, while also showing that route quality depends on how ground movement, UAV sorties, service timing, and rendezvous decisions are coordinated. Synchronization determines whether the spatial advantage of a UAV sortie is converted into system-level time savings. Vehicle-routing research with drones formalized the coupling between truck movement and UAV launch-recovery decisions [21]. Pseudo-node insertion then expanded synchronization choices beyond customer locations by creating virtual rendezvous candidates [22], while hybrid vehicle–drone routing considered coordinated pickup, delivery, and recovery at noncustomer positions [23]. Under fixed-node synchronization, a truck may be forced to stop at a customer node even when it could otherwise progress toward the next service area, and an early UAV may similarly remain idle until the truck arrives. En route synchronization relaxes this restriction by coordinating recovery while the truck advances along its route. In this setting, pseudo-nodes do not represent additional delivery customers. They discretize candidate positions along truck travel arcs in which the truck and UAV can be temporally matched. The resulting recovery flexibility can shorten idle waiting, but it also requires explicit feasibility checks and a time-recursive representation of how any remaining mismatch affects later services.

Solution approaches have developed from mathematical comparison and exact search to metaheuristics and learning-based routing. Comparative work clarifies the structural differences among synchronized truck-and-drone delivery models [24], while branch-and-bound provides an exact approach for small traveling-salesman-with-drone instances [25]. For sequential learning, PPO controls policy updates through a clipped objective [26], whereas attention-based routing and pointer networks provide neural mechanisms for representing and constructing combinatorial routes [27,28]. Hybrid VNS/Tabu search has been applied to vehicle routing with drones and en route operations [29], and exact decomposition provides another benchmark approach for the TSP with a drone [30]. The original graph attention network supplies the principal graph-encoding foundation [31], while en route truck–drone parcel-delivery research supports moving-rendezvous route design [32]. Related methodological foundations include an exact formulation for energy-aware drone routing [33] and GAT-augmented PPO in a network-mapping domain [34]. Attentive graph networks have also been studied for multi-agent trajectory planning [35], and generalized advantage estimation supports policy-gradient learning [36]. Recent UAV–UGV research highlights the importance of task coordination, reliable communication, and environmental adaptability in heterogeneous aerial–ground systems [37]. Studies in

demanding terrain further demonstrate that platform mobility and resilient communication are critical to coordinated unmanned operations [38].

Existing studies on truck–UAV collaborative delivery still face limitations in handling waiting losses caused by fixed-node rendezvous, integrating pseudo-node recovery into the routing process, and representing spatial–temporal dependencies in sequential decision-making. To isolate and analyze the effect of en route synchronization, this study deliberately focuses on a benchmark-level single-truck and single-UAV setting, in which each UAV sortie serves one customer and then returns to the truck for recovery. Under this controlled setting, the methodological contribution lies in integrating adapted pseudo-node recovery and en route synchronization mechanisms with GAT-based state encoding, PPO-based truck-route construction, time-recursive waiting propagation, and feasibility-constrained UAV dispatch within a unified hierarchical framework. It should be noted that the current model is intended for methodological validation on standard benchmark instances rather than direct deployment in full-scale logistics systems. Real-world factors such as multiple trucks and UAVs, heterogeneous fleets, payload-dependent UAV energy consumption, weather effects, traffic congestion, communication uncertainty, and dynamic customer requests are left for future extensions.

Figure 1 presents the overall methodology and research workflow of this study. The research proceeds from problem analysis and model formulation to hierarchical GAT-PPO algorithm design and experimental validation, thereby providing a structured overview of the proposed truck–UAV routing framework.

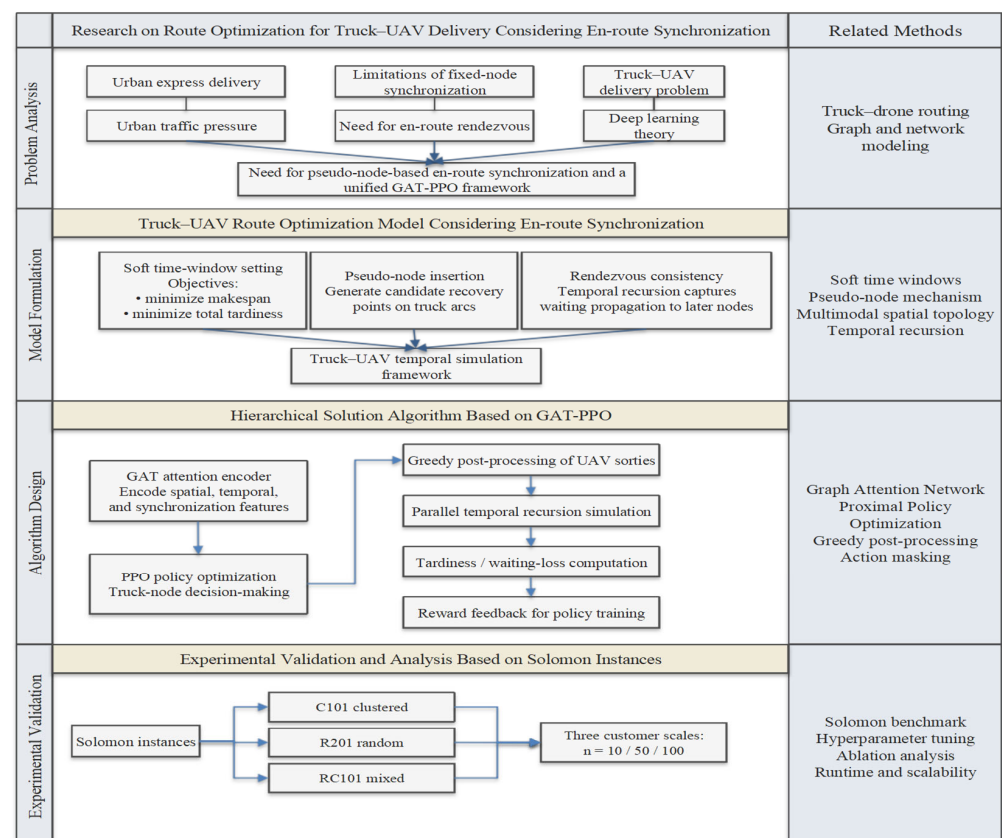


Figure 1. Technical route of the proposed truck–UAV route optimization framework.

The remainder of this paper is organized as follows. Section 2 formulates the truck–UAV collaborative delivery problem and introduces the en route synchronization and pseudo-node insertion mechanism. Section 3 presents the hierarchical GAT-PPO solution framework, including graph construction, graph attention encoding, PPO-based route

decision-making, and UAV dispatch post-processing. Section 4 reports the experimental settings, comparative results, pseudo-node analysis, ablation analysis, and computational performance. Section 5 summarizes the main findings, implications, limitations, and future research directions.

2. A Path Optimization Model for Truck–UAV Collaborative Delivery Considering En Route Synchronization

2.1. Problem Description

To address the urban last-mile delivery scenario, and to highlight the critical role of the synchronization mechanism while ensuring operability and model controllability, this study adopts a single-customer mission rule: each UAV mission is composed of three segments, namely the take-off point, customer point, and recovery point. A single take-off mission only visits one customer to complete the service, after which the UAV must rendezvous with the truck at a designated recovery point for retrieval. Customer delivery tasks can be flexibly allocated between the truck and the UAV; each customer can either be directly served by the truck or independently delivered by the UAV in a single-customer mission. If a customer is served by the UAV, the truck does not need to visit that node. To balance engineering feasibility and scheduling flexibility, this study sets strategic constraints: the UAV take-off points are strictly limited to fixed nodes, while pseudo-nodes only serve as candidate recovery points for the UAV and do not undertake the take-off function. The timeliness requirement for terminal distribution is reflected in the form of time windows. This study adopts a soft time-window approach: for each customer, a specific time window is assigned. If the service start time is earlier than the lower bound of the time window, the service will be delayed until the lower bound is reached before starting; if the service start time exceeds the upper bound of the time window, the resulting delay will be included in the total delay. Taking the makespan and total delay as the core evaluation indicators, the waiting effect in the collaborative system can be fully characterized without explicitly modeling the waiting cost.

The scope of this study is defined by several modeling assumptions. First, the delivery system consists of one truck and one UAV, and each UAV sortie serves only one customer. This setting is adopted to focus on the synchronization relationship between the truck route, UAV dispatch, and en route recovery-point selection. Second, truck and UAV travel times are assumed to be deterministic. Traffic congestion, time-dependent road speeds, weather-induced flight delays, and communication interruptions are not explicitly modeled. Third, UAV endurance is represented by a maximum mission-time threshold, while detailed battery consumption, payload-dependent energy use, wind effects, and vertical take-off and landing energy are not considered. Fourth, all customer requests and time windows are assumed to be known before route optimization, and no new customer requests, cancelations, or real-time order changes are introduced during execution. These assumptions make the problem suitable for benchmark-based algorithmic evaluation, but they also limit the direct applicability of the model to more complex real-world logistics systems. Therefore, this study should be understood as a benchmark-based theoretical and computational validation rather than a real-world deployment case based on a specific logistics company.

To facilitate the formalized description of the subsequent model, this study uniformly defines the sets, parameters, decision variables, and optimization indicators involved, and the meaning of each symbol is presented in Table 1.

Table 1. Key symbols and their meanings.

Symbol	Meaning
N	Set of customer nodes
P	Set of pseudo-nodes
V	Set of nodes
E^T	Set of road arcs for truck travel
E^D	Set of flight arcs for UAV travel
τ_{ij}^T	Travel time of the truck from node i to node j
τ_{ij}^D	Flight time of the UAV from node i to node j
$[a_k, b_k]$	Soft time-window of node k
s_k	Service/operation time at node k (rendezvous operation time for pseudo-nodes)
$T^{\max}$	Maximum mission time for single-customer flight (endurance threshold)
x_{ij}	Binary variable indicating whether the truck traverses arc (i, j)
y_i	Binary variable indicating whether customer i is served by the truck
z_{uiv}	Flight mission: Take-off point $u \rightarrow$ Service point $i \rightarrow$ Recovery point v
t_k^{arr}	Arrival time
t_k	Service start time
t_k^{dep}	Node departure time
$C_{\max}$	System makespan
$tard_i$	Delay of customer i
$TARD$	Total system delay
r	Conversion coefficient between truck travel distance and travel time
T_s	Total service time of the truck for all customers
T_w	Total waiting time of the truck for UAV rendezvous

2.2. Model Formulation

To evaluate the overall operational efficiency and service quality of truck–UAV collaborative delivery, a weighted objective function is constructed by considering both system makespan and total tardiness, as shown in Equation (1).

$$Z = \omega_1 C_{\max} + \omega_2 TARD \quad (1)$$

where the weight coefficient $\omega_1, \omega_2 \geq 0$ reflects the trade-off between operational efficiency and service punctuality $\omega_1 + \omega_2 = 1$. The objective function is used as the comprehensive evaluation criterion for comparing different routing and scheduling schemes.

The core of the objective function is to minimize the weighted sum of two temporal losses. Let the system start at T_{start} , and let the completion time of all delivery tasks be denoted by T_{end} . The makespan represents the total duration from the start of delivery to the completion of all customer services and the return of the truck to the depot. It can be decomposed into truck travel time, customer service time, and synchronization waiting time, as shown in Equation (2).

$$C_{\max} = T_{end} - T_{start} = rD + T_s + T_w \quad (2)$$

where r is the conversion coefficient that transforms truck travel distance into travel time, D represents the truck travel component, T denotes the accumulated service duration of all customers, and T_w denotes the time spent by the truck waiting for UAV rendezvous. Since truck speed and customer service time are fixed in this study, T_s does not affect route selection. In contrast, T_w and $TARD$ directly influence the service start time of subsequent customers and therefore affect total tardiness. Thus, the waiting effect is captured through makespan and tardiness propagation, and an additional independent waiting-cost term is not required.

Let the customer set be N . For customer $i \in N$, the service start time is denoted by t_i , and the soft time-window is $[a_i, b_i]$. If $tard_i$ exceeds $t_i > b_i$, tardiness is generated;

otherwise, the tardiness is zero. The total tardiness is obtained by summing the tardiness of all customers, as shown in Equations (3)–(5).

$$t_i = \max(a_i, t_i^{arr}) \quad (3)$$

$$TARD = \sum_{i \in N} tard_i \quad (4)$$

$$tard_i = \max(0, t_i - b_i) \quad (5)$$

Since $C_{\max}$ and $TARD$ are global indicators that can only be calculated after a complete delivery sequence is generated, they are not directly suitable as stepwise reward signals for reinforcement learning. Therefore, the global objective is decomposed into locally computable step-level components. Specifically, the total loss can be represented by the accumulated travel-distance loss and tardiness loss over the decision process, as shown in Equation (6).

$$Z = \omega_1 C_{\max} + \omega_2 TARD \approx \omega_1 rD + \omega_2 TARD + \text{constant} \quad (6)$$

$D = \sum_{i=0}^T d_i$, let d_i be the stepwise travel distance at decision step i , and let $tard_i$ denote the corresponding tardiness penalty calculated by the temporal simulator. By substituting these stepwise components into the global objective, Equation (7) can be obtained.

$$\omega_1 rD + \omega_2 TARD = \sum (\omega_1 r d_i + \omega_2 tard_i) = \omega_1 r \sum (d_i + \alpha tard_i) \quad (7)$$

Since $\omega_1 r$ is positive, minimizing Z is consistent with minimizing $\sum (d_i + \alpha tard_i)$. Therefore, the reward function used during training is constructed from the stepwise travel distance and tardiness penalty, as shown in Equation (8).

$$r = -(d_i + \alpha tard_i) \quad (8)$$

where d_i denotes the truck travel-distance penalty generated by the current decision, $tard_i$ represents the immediate tardiness penalty of the currently served customer, and $\alpha = \omega_2 / (\omega_1 r)$ controls the relative importance of tardiness in the reward function.

The cumulative reward over the whole trajectory is expressed in Equation (9).

$$\sum_{t=0}^T r_t = -(D + \alpha \cdot TARD) \quad (9)$$

The cumulative reward is consistent with the minimization direction of the global objective. Therefore, improving stepwise decisions can contribute to reducing the overall objective value. The penalty coefficient α is treated consistently with the scale of ω_1 and ω_2 . The stepwise penalty d_i provides immediate feedback for policy learning, while $tard_i$ is recursively calculated by the temporal simulator. This recursion naturally incorporates the propagation effect of truck–UAV synchronization waiting on subsequent service start times. The design alleviates the sparse-reward problem caused by using only global terminal indicators.

For truck movement on the road network, if the truck travels from node i to node j , the arrival time, service start time, and departure time should satisfy the temporal recursion relationship determined by truck travel time τ_{ij}^T and node service time s_i , as shown in Equations (10)–(12).

$$t_j^{arr} = t_i^{dep} + \tau_{ij}^T \quad (10)$$

$$t_j = \max(a_j, t_j^{arr}) \quad (11)$$

$$t_j^{dep} = t_j + s_j \quad (12)$$

Each customer must be served exactly once by either the truck or the UAV. Let $y_i = 1$ indicate whether customer i is served by the truck. If customer i is not served by the truck, then one feasible UAV sortie must be arranged to serve this customer. This task coverage relationship is formulated in Equation (13).

$$y_i + \sum_{u \in V} \sum_{v \in V} z_{uiv} = 1, \forall i \in N \quad (13)$$

When $y_i = 0$, the truck does not need to visit customer i , and a corresponding UAV sortie must be generated to complete the service.

The truck starts from and returns to the depot, as shown in Equation (14).

$$\sum_{j:(0,j) \in E^T} x_{0j} = 1, \sum_{j:(j,0) \in E^T} x_{j0} = 1 \quad (14)$$

For each customer node $i \in N$, if it is served by the truck, the truck must visit it exactly once; if it is served by the UAV, the truck should not visit it. The consistency between in-degree, out-degree, and the service assignment variable y_i is formulated in Equation (15).

$$\sum_{j:(j,i) \in E^T} x_{ji} = y_i, \sum_{j:(i,j) \in E^T} x_{ij} = y_i, \forall i \in N \quad (15)$$

For pseudo-node $p \in P$, whether the truck stops at this node is determined by the optimization model. A binary visit variable $g_p \in \{0, 1\}$ is introduced to describe whether the truck visits the pseudo-node, and the corresponding flow consistency constraint is given in Equation (16).

$$\sum_{j:(j,p) \in E^T} x_{jp} = g_p, \sum_{j:(p,j) \in E^T} x_{pj} = g_p, \forall p \in P \quad (16)$$

For any road arc $(i, j) \in E^T$, if $x_{ij} = 1$, the arrival and departure times must satisfy the temporal recursion relationship. A sufficiently large constant M is introduced to linearize this logical condition, as shown in Equation (17).

$$t_j^{arr} \geq t_i^{dep} + \tau_{ij}^T - M(1 - x_{ij}), \forall (i, j) \in E^T \quad (17)$$

The service start time must satisfy both the arrival-time constraint and the lower bound of the soft time-window, as shown in Equation (18).

$$t_j \geq a_j, t_j \geq t_j^{arr}, \forall j \in V \quad (18)$$

The departure time is then determined by the service start time and the service or operation duration at the node, as shown in Equation (19).

$$t_j^{dep} = t_j + s_j, \forall j \in V \quad (19)$$

In the truck–UAV collaborative delivery system, the UAV must rendezvous with the truck after completing customer service. The rendezvous consistency constraint defines the spatio-temporal relationship between the truck and the UAV at the recovery point. If the truck arrives earlier than the UAV, it must wait until the UAV lands and completes the recovery operation before continuing to the next node.

Assume that the UAV takes off from launch node u , serves customer i , and is recovered at recovery node v . The UAV arrival time $t_{arr}^D(v; u, i)$ at the recovery node v is determined

by the truck arrival time at the launch node, the UAV flight time, and the UAV service time, as shown in Equation (20).

$$t_{arr}^D(v; u, i) = t_u^{dep} + \tau_{ui}^D + s_i + \tau_{iv}^D \quad (20)$$

where t_u^{dep} denotes the time at which the truck reaches the UAV launch node u , τ_{ui}^D represents the UAV flight time from u to customer i , s_i is the customer service time, and τ_{iv}^D denotes the flight time from customer i to recovery node v .

The truck departure time t_v^{dep} from the recovery node must be no earlier than the UAV arrival time $t_{arr}^D(v; u, i)$ at that node. This rendezvous consistency condition is formulated in Equation (21).

$$t_v^{dep} \geq t_{arr}^D(v; u, i) - M(1 - z_{uiv}) \quad (21)$$

The binary sortie variable $z_{uiv} \in \{0, 1\}$ indicates whether the UAV sortie from launch node u to customer i and then to recovery node v is selected. If $z_{uiv} = 1$, the sortie is selected, the rendezvous consistency constraint becomes active, and the truck must wait until the UAV arrives at the recovery node. Otherwise, when $z_{uiv} = 0$, the constraint remains inactive.

2.3. En Route Synchronization and Pseudo-Node Insertion Mechanism

A pseudo-node is defined as a geographical location that is neither a customer site nor a depot, and is located on the travel arc where the UAV can depart or rendezvous with the truck. In the truck–UAV joint delivery system, the UAV and the truck cannot always arrive at the predetermined rendezvous point simultaneously, and such asynchronous phenomena will generate waiting time loss and directly increase the overall completion time. Some existing studies have proposed the concept of en route recovery, yet they rely on simple heuristic methods. In collaborative scenarios allowing en route synchronization, the rendezvous location can be any continuous point on the truck's travel arc, rather than being limited to fixed customer nodes or the depot. To reduce the modeling complexity caused by continuous rendezvous locations, a widely adopted approach is to introduce the concept of pseudo-node. The pseudo-node is set on the road arc (i, j) , and its relative position can be described by the positional parameter $\mu \in (0, 1)$ on the arc, as illustrated in Figure 2.

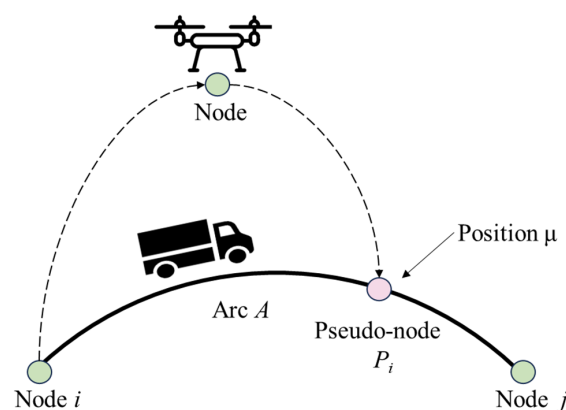


Figure 2. Pseudo-node insertion on a truck travel arc, where λ denotes the relative position of the candidate UAV recovery point.

En route synchronization is a collaborative mode proposed based on the pseudo-node mechanism. It refers to a collaborative mode where, after the UAV completes the last-mile delivery task, it achieves spatio-temporal matching and recovery docking with the moving truck within the continuous space of the truck's travel arc. This mode can ignore the

hard constraints of traditional fixed-node synchronization on the rendezvous location, expanding the rendezvous point from discrete real nodes such as customer nodes and depots to the continuous arc of the truck's travel path. Through the pseudo-node insertion mechanism, dynamic spatio-temporal matching between the truck and the UAV is realized, converting the invalid stationary waiting of the truck at fixed nodes into dynamic docking during the truck's travel, thereby fundamentally reducing the idle waiting consumption in collaboration and improving the overall efficiency of the system.

Considering that continuous optimization would lead to a remarkable increase in modeling complexity, a discretization approximation strategy is adopted. Two candidate pseudo-nodes are generated on each recovery arc with two sets of position parameters. In this study, pseudo-nodes are generated on each candidate truck travel arc by linear interpolation. The position parameter μ is used to describe the relative location of a pseudo-node on the arc. Specifically, $\mu = 1/3$ and $\mu = 2/3$ are selected, so that two candidate pseudo-nodes are generated on each candidate recovery arc.

The two-position design is adopted for two reasons. First, $\mu = 1/3$ and $\mu = 2/3$ divide the truck arc into three segments and represent two typical temporal patterns of en route recovery. The position $\mu = 1/3$ corresponds to an early recovery candidate, where the truck passes the pseudo-node earlier, while $\mu = 2/3$ corresponds to a late recovery candidate, where the truck passes the pseudo-node later. Therefore, the recovery time can be adjusted within the arc instead of being restricted to fixed physical nodes. Second, increasing the number of position levels would increase the number of candidate pseudo-nodes and enlarge the graph structure. This would further increase UAV sortie feasibility checking and reduce training efficiency. Therefore, the two-position setting is used as a tractable discretization strategy to balance synchronization flexibility and computational cost. It should be noted that this setting is not intended to prove the globally optimal pseudo-node location, but to provide representative early and late recovery candidates for benchmark-based validation.

After serving customer i , the UAV flies directly to the pseudo-node p on arc (u, r) to complete en route recovery, and the rendezvous constraints are formulated by temporal consistency conditions, as shown in Figure 3. First, the truck is allowed to skip customer i served by the UAV and travel directly to the farther physical node r , while the UAV flies synchronously and gradually approaches during the truck's movement. Second, the dual-position option of the pseudo-node enables further adjustment of the recovery time within the arc segment. When $\mu = 1/3$, the pseudo-node is close to the launch side, and the truck passes by earlier; when $\mu = 2/3$, it is close to the terminal side of the arc, and the truck passes by later. The two-position gears directly affect the truck's arrival time at the pseudo-node, thereby adjusting the UAV's airborne waiting time.

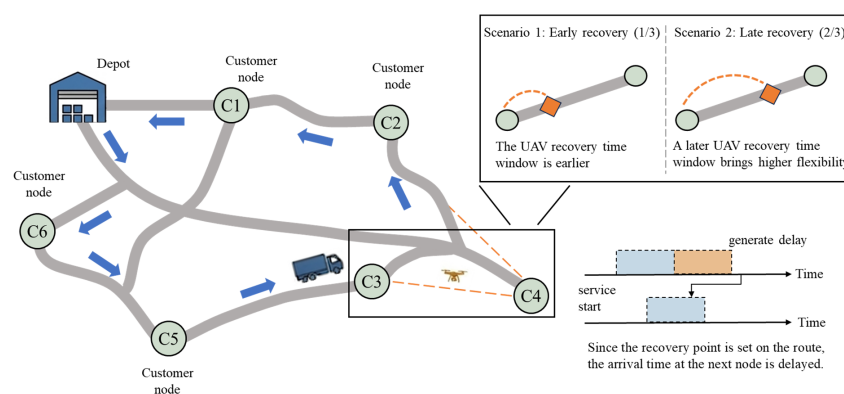


Figure 3. Pseudo-node-based UAV recovery process, including UAV launch, customer service, and en route recovery along the truck arc.

3. Hierarchical Solution Algorithm Based on GAT-PPO

3.1. Construction and Training of GAT Model

Graph Attention Network (GAT) is a graph neural network that learns node representations by assigning different attention weights to neighboring nodes. In this study, GAT is used to encode the spatial relationships, time-window urgency, and synchronization-related features in the truck–UAV delivery graph. The collaborative delivery system is modeled as a multimodal delivery network graph, which not only preserves the feasible adjacency movement relationships of the truck in the road network, but also characterizes the flight reachability of the UAV from the launch node to the customer node and then to the recovery node. This provides structured input for subsequent graph attention encoding and reinforcement-learning decision-making. Given the setting of a single truck, a single UAV, and one customer per flight sortie, this section presents the graph construction method and feature engineering from three perspectives: node set, edge set, as well as node and edge features. The three-dimensional spatial topology of the multimodal collaborative delivery network is shown in Figure 4.

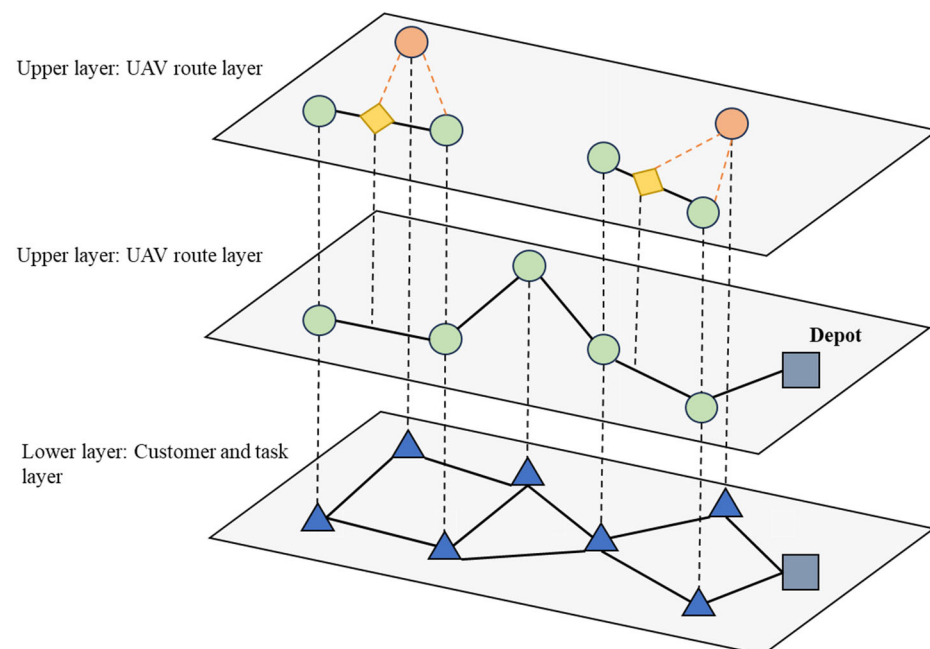


Figure 4. Multimodal truck–UAV delivery graph. Gray squares denote the depot; blue triangles, green circles, orange circles, and yellow diamonds denote customer tasks, physical nodes, UAV-served customers, and pseudo-nodes, respectively; solid black, orange dashed, and vertical dashed lines denote truck-road edges, UAV-flight edges, and inter-layer mappings, respectively.

The node set is denoted as $V = \{0\} \cup N \cup P$, in which 0 represents the depot node, N denotes the set of customer nodes, and P stands for the set of pseudo-nodes. Construct road adjacent edges E^T and flight-reachable edges E^D , which, respectively, represent the drivable adjacency relationships of the truck in the road network and the reachable flight relationships of the UAV from node i to node j in the airspace.

Node features are divided into static features and dynamic features. Static features include node coordinates, node type identifier, customer service time, and time-window deadline. Dynamic features contain the flag of whether a node has been served or selected as a recovery point, the estimated service start time, and the urgency degree defined according to the deadline. Among them, the service start time and urgency degree are updated dynamically in the iteration. After each decision step, the updated temporal

information is output through time sequence simulation and written back into the graph state, serving as the input for graph attention encoding and policy output in the next step.

For road edges, road distance or travel time is adopted as edge features. For flight edges, the flight time is calculated based on the Euclidean distance, with fixed take-off and landing time consumption incorporated. Under the setting of one customer per UAV sortie, flight edges are mainly used to establish the relational connections corresponding to UAV launching, customer service and recovery operations, and participate in endurance constraint judgment through action masking in subsequent solution procedures.

As shown in Figure 5, the original feature vector of each node contains coordinates, time windows, service time, and dynamic temporal features such as the current arrival time and remaining time-window width. After unifying the feature dimensions of different types of nodes, these features are concatenated into the initial node feature vector $h_i^{(0)}$. The input layer maps $h_i^{(0)}$ to a unified hidden dimension via linear transformation to obtain the node embedding of the 0-th layer, which is taken as the input for subsequent attention aggregation.

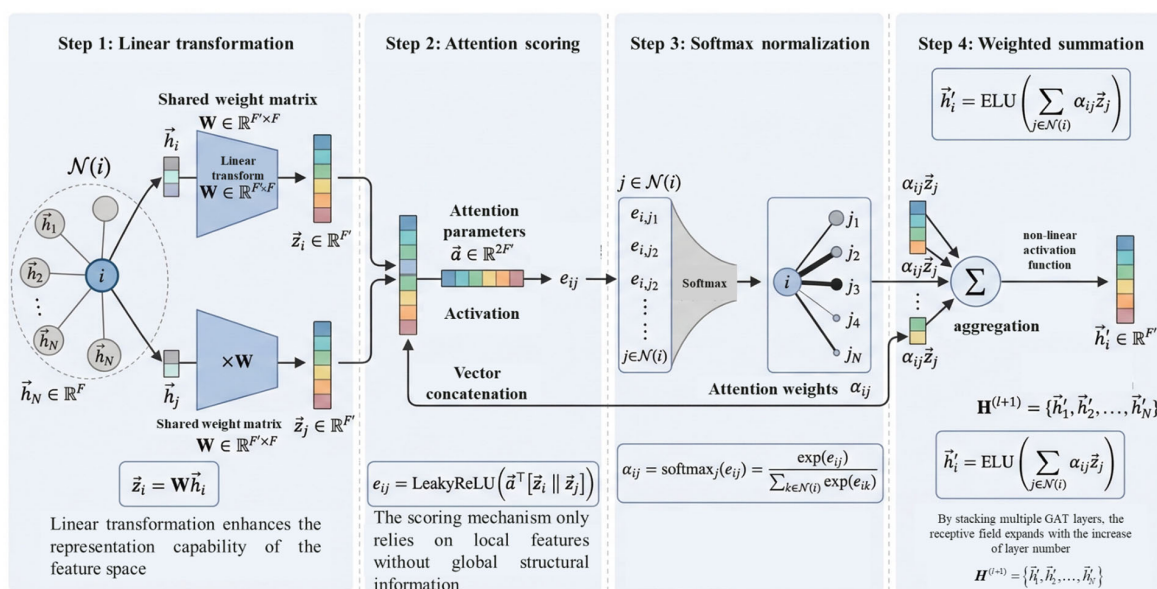


Figure 5. Single-head attention aggregation in the GAT encoder, where neighboring node features are weighted and aggregated to update node embeddings.

At layer l , for node i and its neighbor node j , their weighted feature representations $h_i^{(l)}$ and $h_j^{(l)}$ are first linearly transformed to obtain $\tilde{h}_i^{(l)} = W^{(l)} h_i^{(l)}$ and $\tilde{h}_j^{(l)} = W^{(l)} h_j^{(l)}$. Subsequently, the two vectors are concatenated, and the attention score is calculated using a learnable parameter vector together with the LeakyReLU activation function:

$$e_{ij} = \text{LeakyReLU}(a^T [\tilde{h}_i^{(l)} || \tilde{h}_j^{(l)}]) \quad (22)$$

Softmax normalization is performed on all neighbors of node i to yield the attention weight α_{ij} , which denotes the information contribution proportion of neighbor j to node i . The neighborhood representations are weighted and summed according to the attention weights, and the output is passed through an activation function to obtain the node representation of layer $l + 1$:

$$h_i^{(l+1)} = \sigma \left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} \tilde{h}_j^{(l)} \right) \quad (23)$$

The above process is repeated for L layers. Each layer further aggregates neighborhood information within a wider range based on the output of the previous layer, and finally obtains the high-level embedding $h_i^{(L)}$ of each node.

After the node-level embeddings $h_i^{(L)}$ are acquired, average pooling is implemented on the final-layer representations of all nodes in the graph to generate the graph-level embedding h_G .

$$h_G = \frac{1}{|V|} \sum_{i \in V} h_i^{(L)} \quad (24)$$

The node-level representation $h_i^{(L)}$ is used for scoring and ranking the candidate set, while the graph-level representation h_G is adopted to characterize the overall urgency level, both jointly supporting policy output. As a shared encoder for the policy network and value network, the graph attention network is updated together with policy parameters during the training process of proximal policy optimization.

3.2. Proximal Policy Optimization Training and Dynamic Decision-Making Method

The dynamic improvement process of collaborative delivery is modeled as a Markov Decision Process (MDP). Based on the PPO training strategy, the implementation details are clarified from three aspects: the design of state, action and reward function, feasibility constraints and action masking, as well as training process and stability settings.

The state space is based on the multimodal delivery graph and its node and edge features, including structural information (truck visit sequence, pseudo-node set and edge relationships) and temporal information (node arrival, waiting, service times and urgency features). The simulator writes back dynamic features to update the state. The action space is defined as selecting the next node to be visited by the truck. The action mask sets the probability of visited nodes to zero to ensure the validity of actions, and the UAV sortie assignment is completed by a greedy post-processing algorithm. The reward function is the weighted sum of distance and delay penalties, providing a clear optimization direction for the model through single-step local penalties and ensuring training stability.

Policy output must satisfy hard constraints. This paper introduces an action masking mechanism as shown in Table 2 to filter infeasible actions. The control parameter u is expressed as a step function, taking one when the corresponding feasibility condition is satisfied and zero otherwise.

Table 2. Summary of action masking rules.

Stage	Candidate Set	Feasibility Conditions	Output
PPO masking: select node to visit	$\pi \cap N$	Node not visited Set probabilities of visited nodes to 0 via action masking	$mask^{(1)}(i) \in \{0, 1\}$
Post-processing: select recovery node r	$\{0\} \cup \pi$	$\tau_{ui} + \tau_{ir} + \delta \leq T_{\max}$ Waiting Time ≤ 120 s UAV available and idle	Feasible sortie (u, i, r)
Global: occupancy consistency	—	No new sortie allowed before UAV recovery	Skip current dispatch

The PPO training adopts a cyclic workflow of sampling–evaluation–updating. In each episode, the policy network samples actions to generate the truck visiting sequence, and a post-processing algorithm determines UAV sorties. A parallel temporal simulator recursively feeds back evaluation indicators and writes back the updated system state. End-to-end joint training is adopted, in which the attention encoder and PPO policy parameters are updated collaboratively. Entropy regularization and gradient clipping are introduced to alleviate policy collapse and gradient explosion. The overall training loop is illus-

trated in Figure 6, which is well-adapted to strongly constrained non-stationary temporal feedback scenarios.

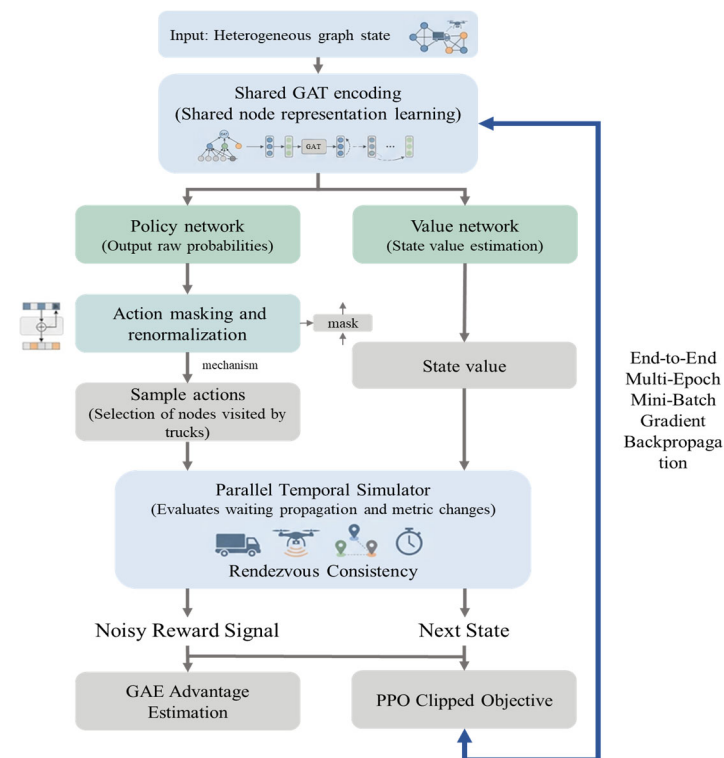


Figure 6. Training loop of the GAT-PPO framework, including graph encoding, PPO decision-making, UAV dispatch post-processing, temporal simulation, and reward feedback.

3.3. Interaction and Collaborative Optimization Mechanism Between GAT and PPO

The GAT-PPO framework adopts a hierarchical structure for truck routing and UAV dispatch post-processing (Figure 7). The GAT encoder extracts features from the multimodal delivery graph, and the PPO policy network selects truck visiting nodes stepwise via graph embedding to generate the full visit sequence. Once determined, this sequence is traversed by a deterministic post-processing algorithm to greedily allocate UAV sorties.

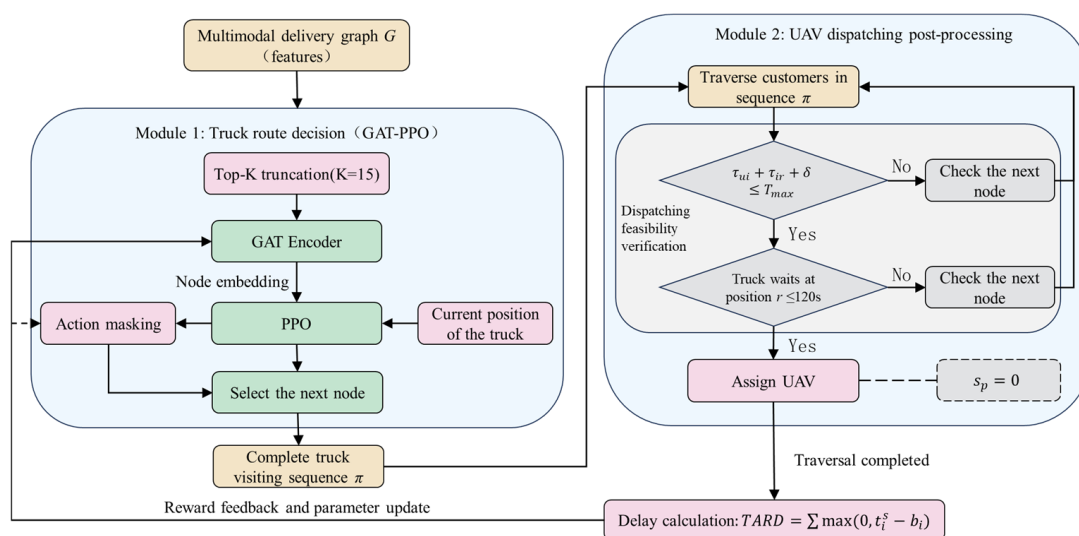


Figure 7. GAT-PPO decision and post-processing flow for truck routing, UAV sortie selection, pseudo-node recovery, and feasibility checking.

Information interaction is embodied in the node embedding encoded by GAT, which integrates static features such as coordinates, time windows and service time as well as dynamic temporal features, providing decision-making basis for PPO. The visit sequence output by PPO determines the calculation of take-off points, recovery candidates and waiting time in greedy post-processing. This forms a closed training loop involving graph representation, route decision, dispatch post-processing, reward feedback and parameter update during the optimization process.

The service time of pseudo-nodes is set to $s_p = 0$. A delay is recorded in TRAD only if the service start time t_i^s exceeds the deadline b_i . Top-K truncation is applied to flight-reachable edges solely to reduce the computational cost of GAT encoding, and does not change the judgment logic of endurance constraints and waiting time in subsequent UAV sortie allocation.

4. Empirical Analysis and Result Comparison

4.1. Experimental Data and Parameter Calibration

The experimental hardware is equipped with an RTX 4090 GPU (24 GB VRAM), 16 vCPU Intel Xeon Platinum 8352V (2.10 GHz) and 120 GB RAM. The software environment uses PyTorch 1.11.0 with CUDA 11.3. For realistic urban delivery settings, the truck speed is set to 12.5 m/s and UAV speed to 15 m/s. The maximum UAV endurance is 600 s, and individual customer service time is 90 s.

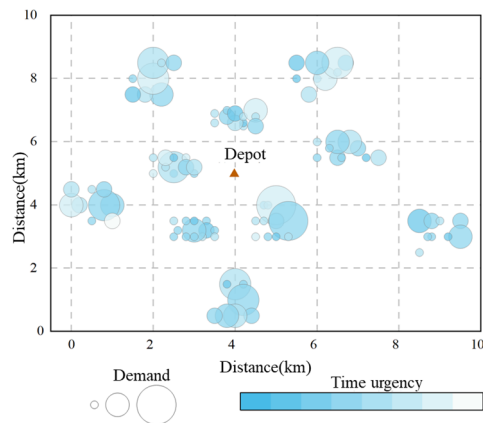
The Solomon VRPTW benchmark is adopted as the urban last-mile delivery dataset, covering clustered (C), random (R) and mixed (RC) demand patterns. These benchmark instances are used to provide standardized and repeatable test scenarios for evaluating the proposed framework under different customer distributions and time-window conditions, rather than to represent a specific real company or an actual logistics deployment case. The dimensionless coordinates are scaled to a 10 km $\times$ 10 km urban area, with time units converted into seconds at 1 unit = 8 s. Three typical instances C101, R201 and RC201 are selected, each containing 100 customers and one depot with differentiated spatial and time-window features. Their spatial distributions and time-window Gantt diagrams are presented in Figure 8.

4.2. Hyperparameter Optimization

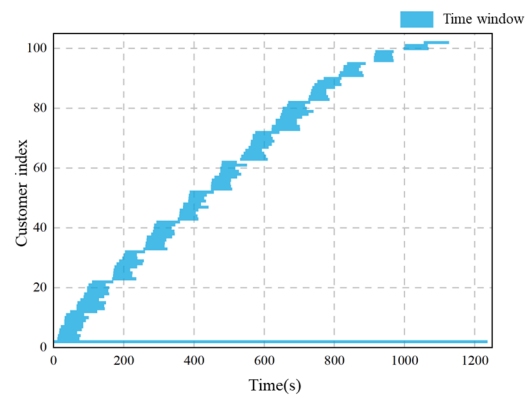
Hyperparameter optimization is implemented using the Optuna version 4.7.0 framework, which is a Bayesian optimization tool that employs the Tree-structured Parzen Estimator (TPE) algorithm to search for optimal hyperparameter combinations. In each iteration, the TPE algorithm adjusts the prior distribution of hyperparameters based on historical observations to determine the next search direction, and updates its estimation model using the objective function value. This enables the efficient exploration of the hyperparameter space and identifies near-global optimal combinations within limited trials.

The hyperparameters optimized in this study include reinforcement-learning parameters (learning rate, discount factor, clip coefficient, etc.) and neural network architecture parameters (number of GAT layers, hidden dimension, etc.). All hyperparameters are tuned on a small-scale instance with 10 customers, and their search ranges are listed in Table 3.

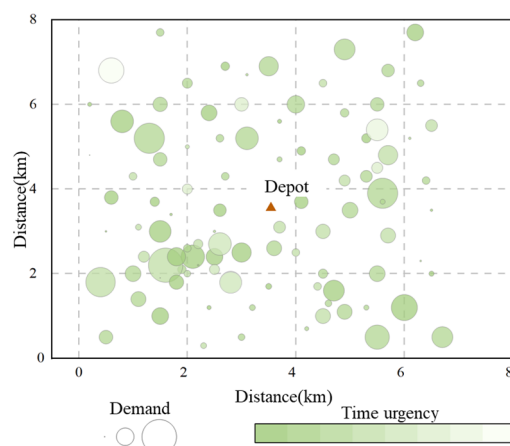
A total of 100 trials are conducted in the hyperparameter optimization process, with 200 training steps performed at the scale of $n = 10$ for each trial. Representative records of partial optimization trials are listed in Table 4, with the optimization objective set to minimize delivery delay.



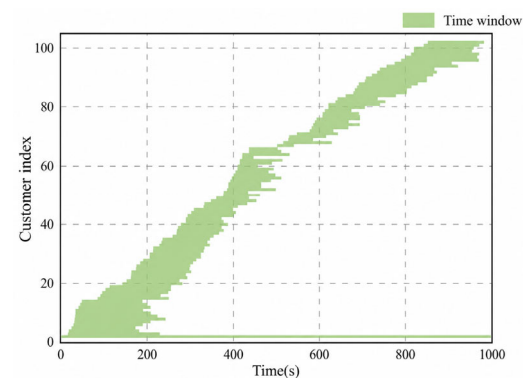
(a) Spatial distribution of C101 instance.



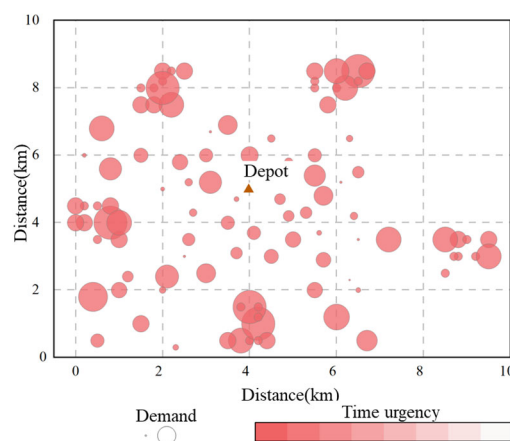
(b) Gantt chart of customer time windows for C101 instance.



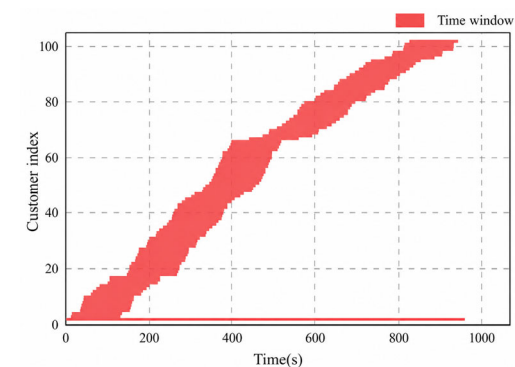
(c) Spatial distribution of R201 instance.



(d) Gantt chart of customer time windows for R201 instance.



(e) Spatial distribution of RC201 instance.



(f) Gantt chart of customer time windows for RC201 instance.

Figure 8. Spatial distribution and customer time-window characteristics of typical instances.

The optimal hyperparameter combination obtained from optimization is presented in Table 5. The graph attention network adopted in this paper captures the relative spatial and temporal constraints between nodes via the attention mechanism, rather than relying on absolute coordinate information. This learning pattern possesses strong inductive bias, and its decision logic exhibits satisfactory transferability across instances of different scales.

Table 3. Hyperparameter settings of the model.

Hyperparameter Name	Meaning	Value Range	Type
lr	Learning Rate	$(1 \times 10^{-5}, 1 \times 10^{-3})$	float
γ	Discount Factor	(0.90, 0.99)	float
λ	GAE Lambda	(0.90, 0.99)	float
vf	Value Function Coefficient	(0.1, 1.0)	float
ent	Entropy Coefficient	(0.01, 0.1)	float
clip	PPO Clip Epsilon	(0.1, 0.3)	float
α	Delay Penalty Weight	(1.0, 20.0)	float

Table 4. Results of hyperparameter optimization trials.

Hyperparameter Name	Trial 0	Trial 5	Trial 15	Trial 25	Trial 30	Trial 62
lr	3.5×10^{-4}	1.2×10^{-4}	2.5×10^{-4}	1.8×10^{-4}	1.4×10^{-4}	4.1×10^{-4}
γ	0.982	0.915	0.945	0.959	0.941	0.966
λ	0.954	0.923	0.968	0.979	0.965	0.971
vf	0.42	0.85	0.52	0.37	0.55	0.21
ent	0.051	0.092	0.048	0.023	0.021	0.034
clip	0.18	0.22	0.21	0.24	0.25	0.15
α	16.41	8.35	5.60	2.77	2.97	1.02
Objective value	28,619.1 s	22,115.4 s	29,201.7 s	20,402.9 s	28,053.0 s	30,815.6 s

Table 5. Optimal hyperparameter combination.

Parameter Type	Hyperparameter Name	Optimal Value
Network Structure	Number of stacked GAT layers	2
	Hidden layer dimension	128
	KNN neighbor number	15
Reinforcement Learning	lr	1.85×10^{-4}
	γ	0.959
	λ	0.979
	vf	0.37
	ent	0.023
	clip	0.24
	α	2.77

Furthermore, the reinforcement-learning hyperparameters mainly control the stability of policy update and the variance of gradient estimation. Such training dynamic characteristics remain essentially consistent for instances of different scales under the same task logic. Therefore, acquiring optimal hyperparameters rapidly on small-scale instances and increasing training steps on large-scale ones to expand the search space is a strategy that balances computational efficiency and model performance. The optimized hyperparameter set can be directly applied to independent training for instances of $n = 50$ and $n = 100$.

With the optimal hyperparameters determined, independent training is conducted on Solomon instances of different scales. To enhance model generalization, one instance is randomly sampled from the corresponding instance pool for episode rollout at each training step. The Adam optimizer is adopted for gradient updating. Considering the search space complexity of different problem scales, the training steps are set as follows: 2000 steps for $n = 10$, 3000 steps for $n = 50$, and 4000 steps for $n = 100$ instances.

4.3. Analysis of Experimental Results

To further verify the global optimization capability of the proposed algorithm for objective Z , and comprehensively evaluate the system performance in terms of operational

efficiency and service punctuality, the overall objective function values of different algorithms under various distribution types and node scales are calculated and plotted based on the established comprehensive evaluation system, as shown in Figure 9.

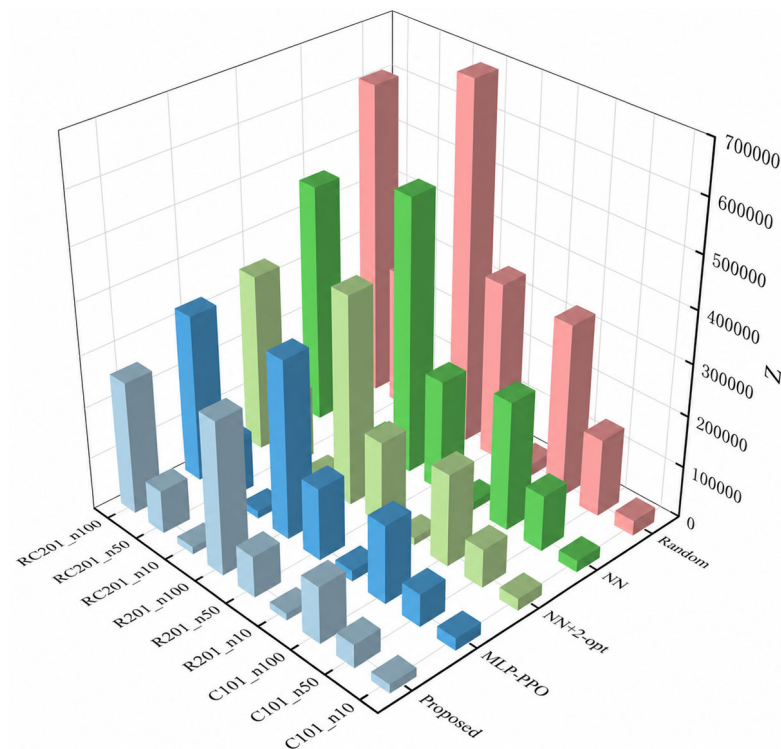


Figure 9. Comparison of objective function values of different route optimization algorithms under different node quantities.

It can be intuitively observed from the spatial distribution of comprehensive objective function values in Figure 9 that the comprehensive cost Z of the proposed GAT-PPO algorithm remains stably at the lowest level across all test instances and customer scales. With the increase in customer nodes, the temporal–spatial coupling constraints of collaborative delivery become exponentially complex. Traditional heuristic algorithms are prone to falling into local optima, leading to a noticeable upward trend of Z . In contrast, the proposed algorithm achieves the slowest growth rate, demonstrating outstanding adaptability to large-scale scenarios.

Moreover, compared horizontally with the MLP-PPO algorithm without the graph attention mechanism, the GAT-PPO algorithm achieves a significant reduction in the value of Z on all types of instances. This strongly verifies that the constructed multimodal graph attention encoding mechanism can effectively capture complex temporal dependencies and spatial neighborhood relationships.

The comparative data of each algorithm in terms of total travel distance, total delay time and completion time are presented in Tables 6–8, respectively.

As shown in Table 6, the proposed GAT-PPO algorithm achieves the shortest total travel distance in all tested instances, indicating that the graph-attention-based policy can generate more compact routing structures. This advantage is particularly evident in medium- and large-scale instances. For example, in the R201 instance with $n = 50$, GAT-PPO obtains a total distance of 94.54 km, which is lower than NN+2-opt, MLP-PPO, ALNS, GA, and VNS. In the C101 instance with $n = 100$, GAT-PPO achieves a distance of 230.71 km, reducing the travel distance by 17.73% compared with NN+2-opt and by 44.8% compared to ALNS. Similar results can also be observed in the R201 and RC201 instances with $n = 100$, where GAT-PPO consistently produces the shortest route among all

compared algorithms. The superiority of GAT-PPO in total travel distance mainly benefits from its graph attention encoder, which can capture spatial relationships among customer nodes and support more compact route construction. However, total travel distance is only an auxiliary performance indicator in this study. Since the overall objective function is mainly determined by makespan and total tardiness, a shorter travel distance does not necessarily imply the best comprehensive objective value. Therefore, the results in Table 6 demonstrate the advantage of GAT-PPO in route compactness, while the final performance of each algorithm still needs to be evaluated together with makespan, tardiness, waiting time, and the aggregated objective value.

Table 6. Comparison of total travel distance of each algorithm on three types of instances (km).

Instance	n	Random	NN	NN+2-opt	MLP-PPO	GAT-PPO	ALNS	GA	VNS
C101	10	51.74	39.19	41.5	40.34	37.24	68.91	82.47	76.38
	50	204.82	198.82	148.97	162.30	100.02	168.72	243.15	188.04
	100	406.96	370.25	280.45	315.42	230.71	412.56	537.92	452.89
R201	10	39.08	34.69	31.85	30.65	26.76	40.28	70.61	55.93
	50	171.40	148.16	139.93	133.68	94.54	172.34	216.58	177.49
	100	344.29	272.4	236.85	255.4	216.24	372.16	519.44	424.73
RC201	10	49.61	43.01	40.68	39.61	38.56	73.82	85.76	80.91
	50	217.30	185.42	175.40	188.75	102.40	186.39	234.88	203.47
	100	473.33	354.82	334.20	345.20	297.86	514.76	711.35	569.24

Table 7. Comparison of total delay of each algorithm on three types of instances (s).

Instance	n	Random	NN	NN+2-opt	MLP-PPO	GAT-PPO	ALNS	GA	VNS
C101	10	35,810.2	25,021.4	25,574.0	28,540.5	18,642.7	20,293.5	22,103.8	23,354.6
	50	184,500.5	125,000.4	84,000.6	72,000.5	50,391.2	55,082.3	60,218.4	63,157.9
	100	423,000.5	310,000.4	215,000.2	186,000.5	128,667.5	140,388.6	153,224.6	161,193.8
R201	10	31,091.5	20,003.3	15,511.7	26,850.2	13,524.9	14,831.9	16,272.3	16,958.1
	50	46,3126	284,769.7	215,000.5	185,000.2	97,188.6	106,028.5	115,704.2	121,752.4
	100	927,836.5	713,313.7	542,369.2	460,000.5	361,294.8	394,508.2	430,856.9	452,711.3
RC201	10	32,500.0	26,858.2	19,631.0	17,100.5	14,718.5	15,990.4	17,396.4	18,425.7
	50	353,865.2	254,538.4	185,000.5	148,000.2	101,012.9	110,343.1	120,561.8	126,583.2
	100	817,151.2	619,243.0	458,904.7	435,000.5	321,862.6	351,292.4	383,519.5	403,247.1

Table 8. Comparison of completion time of each algorithm on three types of instances (s).

Instance	n	Random	NN	NN+2-opt	MLP-PPO	GAT-PPO	ALNS	GA	VNS
C101	10	15,519	14,160.8	11,701.3	11,240.5	6236.4	6602.4	6518.7	6559.1
	50	57,249.1	47,135.9	44,148.3	38,560.2	15,994.8	17,081.8	17,011.6	16,841.3
	100	95,106.8	83,271	77,938.7	56,420.8	25,352.7	26,951.8	26,729.4	26,671.8
R201	10	9207.5	8262.3	7605.7	7150.2	6561.5	7027.2	7024.9	6905.6
	50	26,551.9	18,972.1	16,161.9	14,840.6	12,897.2	13,686.2	13,548.6	13,567.9
	100	48,990	31,103.1	28,670.7	25,850.4	21,705.3	23,158.7	23,043.8	22,832.7
RC201	10	10,490.5	9413	8706.3	8520.5	6479.6	6855.7	6762.8	6815.4
	50	27,367.5	17,624.9	15,607.4	14,530.8	12,562.9	13,424.0	13,376.2	13,219.8
	100	48,292.1	29,208.2	27,364.6	25,840.3	23,199.7	24,657.0	24,443.5	24,402.6

As shown in Table 7, the proposed GAT-PPO algorithm achieves the lowest total delay in all tested instances, indicating that the graph-attention-based reinforcement-learning framework is effective in controlling service tardiness under different customer distributions and problem scales. Compared with ALNS, GA, VNS, MLP-PPO, NN, NN+2-opt, and the random strategy, GAT-PPO consistently obtains lower total delay, especially in medium- and large-scale instances. This result suggests that the proposed framework can better capture time-window urgency and synchronization-related temporal dependencies during truck–drone collaborative decision-making. For example, in the R201 instance

with $n = 50$, the total delay of GAT-PPO is 97,188.6 s, which is lower than 106,028.5 s of ALNS, 115,704.2 s of GA, 121,752.4 s of VNS, 185,000.2 s of MLP-PPO, 215,000.5 s of NN+2-opt, and 284,769.7 s of NN. In the large-scale R201 instance with $n = 100$, GAT-PPO also reduces total delay from 394,508.2 s of ALNS to 361,294.8 s, showing its advantage in time-window-sensitive scheduling when customer locations are dispersed. Compared to MLP-PPO, GAT-PPO consistently reduces total delay across all instance types and scales. This confirms that the graph attention encoder helps the PPO policy capture spatial-temporal correlations, customer urgency, and synchronization constraints more effectively than a plain MLP-based policy. In addition, the total delay of R201 and RC201 instances is generally higher than that of C101, especially when $n = 50$ and $n = 100$. This is because random and mixed customer distributions tend to generate longer travel deviations and more complex synchronization relationships, while clustered customer distributions can form more compact routes and reduce cumulative time-window violations. Therefore, Table 7 demonstrates that GAT-PPO has a clear advantage in total delay control. This result supports the effectiveness of combining graph attention representation with PPO-based sequential decision-making for truck-drone synchronization problems.

As shown in Table 8, the proposed GAT-PPO algorithm achieves the shortest completion time in all tested instances, indicating that the graph-attention-based reinforcement-learning framework is effective in controlling the overall makespan of truck-drone collaborative delivery. Compared to Random, NN, NN+2-opt, MLP-PPO, ALNS, GA, and VNS, GAT-PPO consistently obtains lower completion time across different customer distributions and problem scales. This result suggests that the proposed framework can better coordinate truck movement, UAV dispatch, and recovery timing, thereby improving the temporal efficiency of the collaborative delivery system. For example, in the C101 instance with $n = 100$, the completion time of GAT-PPO is 25,352.7 s, which is 67.47% lower than 77,938.7 s of NN+2-opt, 55.06% lower than 56,420.8 s of MLP-PPO, and 5.93% lower than 26,951.8 s of ALNS. In the large-scale R201 instance, GAT-PPO obtains a completion time of 21,705.3 s, which is lower than 28,670.7 s of NN+2-opt, 25,850.4 s of MLP-PPO, 23,158.7 s of ALNS, 23,043.8 s of GA, and 22,832.7 s of VNS. Similar advantages can also be observed in the RC201 instances, showing that GAT-PPO maintains stable makespan-control performance under clustered, random, and mixed customer distributions. The improvement in completion time mainly benefits from the graph attention encoder and the sequential decision mechanism of PPO. The graph attention module helps the model capture spatial clustering, time-window urgency, and synchronization-related dependencies among nodes, while PPO supports adaptive sequential route construction and UAV dispatch decisions. Compared to MLP-PPO, the consistent reduction in completion time demonstrates that graph-structured representation is more suitable than a plain MLP encoder for truck-drone synchronization problems. Overall, Table 8 shows that GAT-PPO has a clear advantage in completion-time control. Together with the results in Tables 6 and 7, the proposed framework demonstrates strong performance in route compactness, tardiness reduction, and makespan control, supporting the effectiveness of combining graph attention representation with PPO-based sequential decision-making.

To intuitively illustrate the influence of the proposed method on route planning, route analysis is conducted based on three customer distribution types from the three datasets, as shown in Figure 10.

As shown in Figure 10a, the traditional fixed-node synchronization limits drone recovery to customer nodes, which causes waiting propagation, timeout penalties and efficiency loss. With pseudo-nodes, recovery can be arranged along truck driving arcs instead of customer stops, avoiding idle waiting. Figure 10b shows the truck route contains unnecessary detours without pseudo-nodes, and becomes much smoother after adopting pseudo-nodes.

Figure 10c demonstrates that pseudo-nodes enable the deep decoupling of truck-and-drone scheduling in large-scale scenarios. On the RC201_n100 instance, optimization rates reach 27.6% and 20.6%, validating its engineering applicability in complex environments.

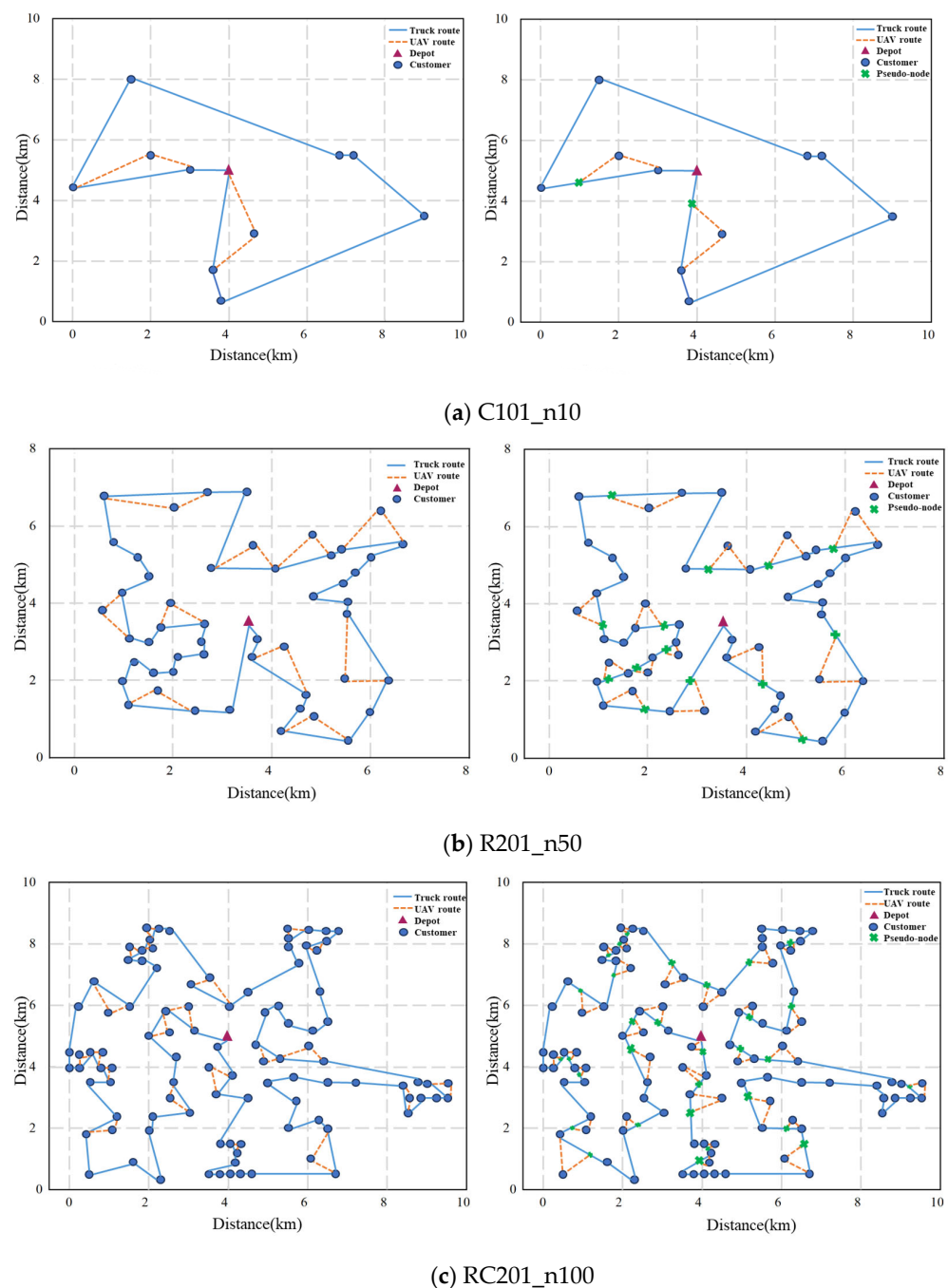


Figure 10. Comparison of routing results before and after optimization for typical instances.

Furthermore, to verify the generality of this mechanism, quantitative comparisons of collaborative performance with and without pseudo-nodes are conducted on three instance types with different node scales and random seeds, as listed in Table 9. In this comparison, the “without pseudo-node” setting represents the fixed-node synchronization scenario, where UAV recovery is restricted to physical nodes and the truck may need to stop and wait for the UAV. The “with pseudo-node” setting represents the en route synchronization scenario, where the truck can continue moving along its travel arc while the UAV serves the customer, and the UAV is recovered at a pseudo-node on the arc. Therefore, Table 9

compares the two synchronization scenarios of truck stopping during UAV delivery and truck moving during UAV delivery.

Table 9. Instance-level comparison with and without pseudo-node insertion.

Instance	n	Seed	Collaborative Sortie	Without Pseudo-Node		With Pseudo-Node		Δ_{TARD} (%)	$\Delta_{C_{max}}$ (%)
				<i>TARD</i> (s)	<i>C_{max}</i> (s)	<i>TARD</i> (s)	<i>C_{max}</i>		
C101	10	0	2	24,297.8	7900.4	19,032.2	6660.0	21.7	15.7
		1	2	27,492.0	7963.2	20,706.8	7041.6	24.7	11.6
		2	1	29,662.9	6445.4	22,608.2	6102.0	23.8	5.3
	50	0	17	82,140.5	21,540.8	53,820.6	17,081.8	34.5	20.7
		1	19	86,880.2	22,105.4	55,140.4	16,840.5	36.5	23.8
		2	18	90,541.1	20,976.2	56,285.9	17,323.1	37.8	17.4
	100	—	30	205,480.0	35,840.5	140,388.6	26,951.8	31.7	24.8
R201	10	0	2	19,317.4	7409.6	14,790.0	7074.8	23.4	4.5
		1	2	16,910.1	6569.6	13,854.2	6162.2	18.1	6.2
		2	2	20,425.6	7294.5	16,930.8	6944.3	17.1	4.8
	50	0	14	161,341.5	16,254.4	105,420.3	13,686.2	34.7	15.8
		1	15	160,620.8	16,840.5	107,840.6	14,510.4	32.9	13.8
		2	15	164,320.2	15,820.2	104,824.5	12,862.0	36.2	18.7
	100	—	35	469,640.2	28,415.6	394,508.2	23,158.7	16.0	18.5
RC201	10	0	2	19,410.5	7231.2	15,552.0	6664.6	19.9	7.8
		1	3	20,150.2	8076.5	17,640.8	7084.8	12.5	12.3
		2	2	17,670.8	6986.9	15,463.2	6774.4	12.5	3.0
	50	0	17	144,520.4	18,542.5	110,343.1	13,427.3	23.6	27.6
		1	19	152,040.6	19,120.6	108,450.2	12,850.5	28.7	32.8
		2	15	146,990.2	17,964.4	112,236.0	13,994.1	23.6	22.1
	100	—	31	485,120.6	31,054.2	351,292.4	24,657.0	27.6	20.6

Based on the data in Table 9, the dynamic pseudo-node mechanism effectively reduces both total delay and completion time across different customer distribution scenarios and customer scales. The two optimization rates reported in the last columns represent the reduction rates of total delay and completion time, respectively. Compared to the setting without pseudo-nodes, the introduction of pseudo-nodes allows the UAV to be recovered at flexible positions along the truck route rather than only at fixed physical nodes, thereby improving the temporal matching between the truck and the UAV. For the clustered C101 instances, the improvement becomes more evident as the customer scale increases. When $n = 100$, the total delay is reduced from 205,480.0 s to 140,388.6 s, and the completion time is reduced from 35,840.5 s to 26,951.8 s, corresponding to reduction rates of 31.7% and 24.8%, respectively. For the random R201 instances, the pseudo-node mechanism also shows clear benefits. In particular, when $n = 50$, the reduction rate of total delay reaches 34.7–36.2% under different random seeds, indicating that flexible recovery points are especially useful when customer locations are dispersed and truck–drone synchronization becomes more difficult. For the mixed RC201 instances, the mechanism also maintains stable improvement. At $n = 100$, total delay and completion time are reduced by 27.6% and 20.6%, respectively. Overall, Table 9 provides instance-level evidence that pseudo-node insertion can reduce synchronization-related delay and improve the efficiency of truck–drone collaborative delivery. However, Table 9 only evaluates the effect of the pseudo-node mechanism and cannot fully explain the individual contributions of other key components, such as GAT, PPO, en route synchronization, and UAV collaboration. Therefore, to further verify the role of each component, an overall ablation study is conducted, and the results are reported in Table 10.

As shown in Table 10, removing any key component increases the objective value, indicating that each module contributes to the overall performance of the proposed framework. The full model achieves the lowest objective value of 50,594.788. When PPO is removed, the objective value increases by 34.875%, showing that reinforcement-learning-based sequential decision-making plays an important role in route construction and UAV dispatch coordination. When the GAT encoder is removed, the objective value increases by 42.764%, indicating that graph attention encoding improves the model’s ability to capture

spatial relationships, time-window urgency, and synchronization-related dependencies among nodes.

Table 10. Overall ablation results of the main components of the proposed framework.

Variant	Makespan Mean (s)	Tardiness Mean (s)	Objective Mean	Objective Increase (%)
Full model	12,754.862	88,434.714	50,594.788	0.000
w/o PPO	15,264.770	121,214.671	68,239.720	34.875
w/o GAT	15,551.635	128,910.652	72,231.143	42.764
w/o pseudo-node insertion	15,731.076	121,189.219	68,460.148	35.311
w/o en route synchronization	18,004.085	157,245.130	87,624.608	73.189
w/o drone	18,252.674	160,842.757	89,547.715	76.990

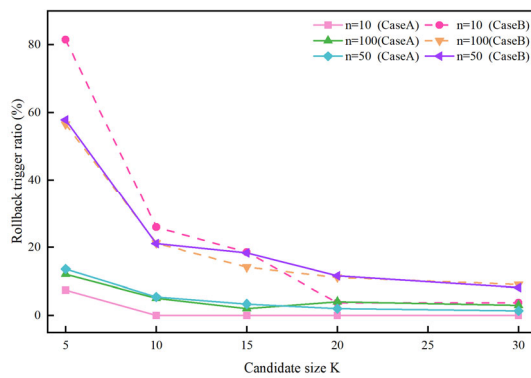
The synchronization-related components also have significant effects. Removing pseudo-node insertion increases the objective value by 35.311%, which is consistent with the instance-level results in Table 9, and confirms that flexible UAV recovery points help reduce synchronization loss. When en route synchronization is disabled, the objective value increases by 73.189%, showing that dynamic rendezvous along truck arcs is one of the core mechanisms of the proposed framework. The w/o drone variant produces the worst result, with a 76.990% increase in the objective value, indicating that truck–drone collaboration itself is essential for improving delivery efficiency. The ablation results further explain the source of the reported improvements. The pseudo-node mechanism and en route synchronization mainly reduce waiting propagation and tardiness accumulation, while GAT and PPO mainly improve spatial–temporal representation and sequential routing decisions. Therefore, the overall improvement in the full model does not come from a single component. Instead, it results from the combined effects of flexible recovery-point selection, graph-based feature encoding, reinforcement-learning-based route decision-making, and feasibility-constrained UAV dispatch.

Overall, the ablation results demonstrate that the proposed framework benefits from both learning-based decision modules and collaborative synchronization mechanisms. PPO supports sequential decision-making, GAT enhances graph-structured spatial–temporal representation, pseudo-node insertion improves the flexibility of UAV recovery, and en route synchronization substantially reduces coordination loss. These results verify the individual contribution of each main component and provide direct evidence for the necessity of the proposed GAT-PPO framework with pseudo-node-based en route synchronization. To further compare the solution-layer performance of Schemes A and B, their fallback trigger ratios and numbers of feasible candidates for the C101, R201, and RC201 instances are presented in Figure 11.

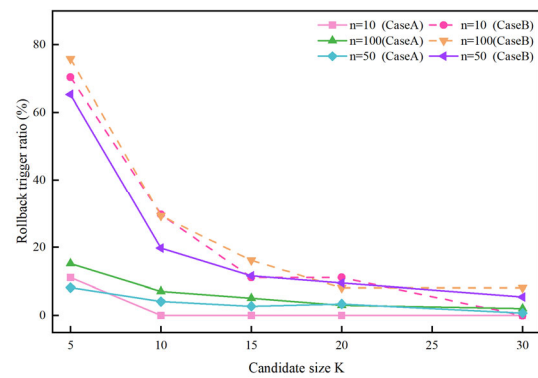
Therefore, Scheme A with $K = 15$ is adopted in the main experiments. Under this setting, the fallback ratio of instances at all scales is controlled within 5%, which balances algorithm completeness and computational efficiency.

Table 11 presents the normalized distance, runtime and fallback ratio of the two schemes for three types of instances at different scales when $K = 15$.

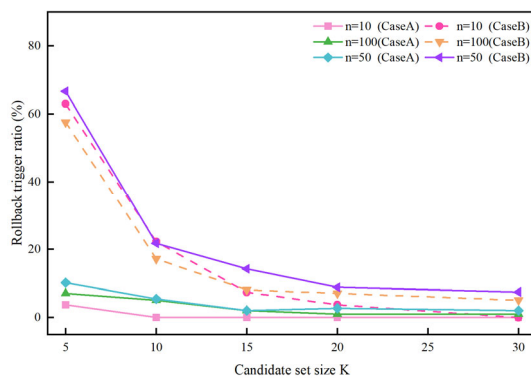
It can be seen from Table 10 that at $K = 15$, Scheme A and Scheme B have identical normalized objective values for all instances with $n = 10$, showing no difference in search quality for small-scale cases. As the problem scale increases, the objective value of Scheme B fluctuates, and its running time and fallback ratio are consistently higher than those of Scheme A. For medium- and large-scale instances, the fallback ratio of Scheme A remains within 5%. Considering the objective value, running time and fallback ratio comprehensively, Scheme A is selected for the main experiments.



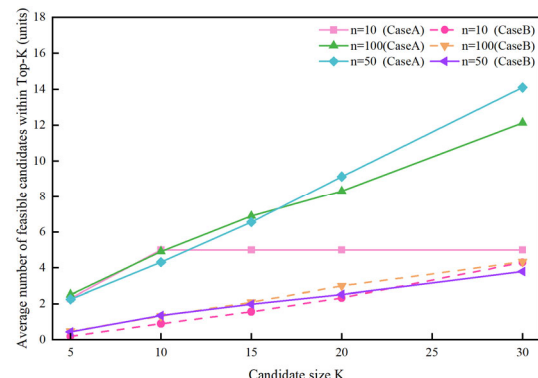
(a) Fallback trigger ratio for C101



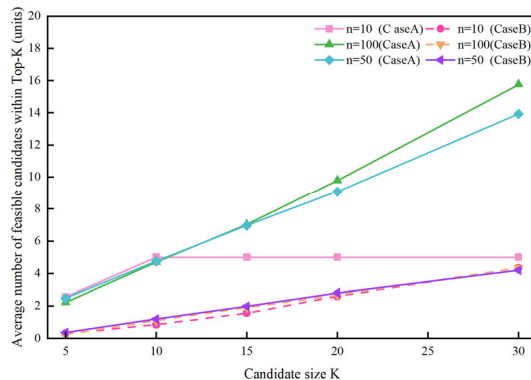
(b) Fallback trigger ratio for R201



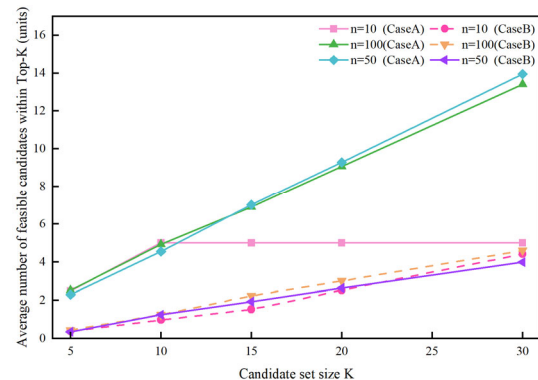
(c) Fallback trigger ratio for RC201



(d) Feasible candidate quantity for C101



(e) Feasible candidate quantity for R201



(f) Feasible candidate quantity for RC201

Figure 11. Comparison of fallback trigger ratio and feasible candidate quantity between Scheme A and Scheme B for typical instances.

Based on the comparative experiments and the analysis with and without pseudo-nodes, this section systematically evaluates the overall performance of the collaborative scheme. Comprehensive analysis is conducted from three perspectives, namely objective value distribution, running time and inference stability, as shown in Table 12.

In terms of objective value, as the customer scale expands from 10 to 100, the average route distance of all three instance types rises obviously, which conforms to the inherent rule that route length and synchronization waiting cost accumulate with the increase in problem scale. For multi-derivation sub-instances at $n = 10$ and $n = 50$, the gap between the mean and optimal values reflects the fluctuation in solution difficulty caused by spatial distribution differences among sub-instances, rather than policy randomness. RC201_n50

has the largest standard deviation, indicating that the collaborative decision quality in mixed distribution scenarios is more sensitive to the spatial pattern of sub-instances.

Table 11. Comparison of Scheme A and Scheme B solution layers at $K = 15$.

Instance	n	Normalized Distance		Running Time		Solution Fallback	
		Scheme A	Scheme B	Scheme A	Scheme B	Scheme A	Scheme B
C101	10	32.186	32.186	0.123	0.184	0.000	0.167
	50	49.050	49.126	0.534	0.677	0.033	0.180
	100	58.923	64.072	1.219	1.894	0.020	0.140
R201	10	24.695	24.695	0.098	0.155	0.000	0.100
	50	49.437	50.745	0.473	0.495	0.027	0.113
	100	85.234	84.205	0.854	1.295	0.050	0.160
RC201	10	30.012	30.012	0.095	0.129	0.000	0.067
	50	58.866	58.721	0.454	0.481	0.020	0.140
	100	73.215	73.215	0.909	2.344	0.020	0.080

Table 12. Evaluation results of GAT-PPO on each instance.

Instance	n	Derivation Number	Objective Value (km)	Optimal Objective Value (km)	Running Time (ms)
			Mean $\pm$ Standard Deviation		Mean $\pm$ Standard Deviation
C101	10	3	40.206 $\pm$ 3.875	37.243	83.4 $\pm$ 4.7
	50	3	109.517 $\pm$ 7.695	100.016	1892.3 $\pm$ 50.2
	100	—	230.711	230.711	8561.2
R201	10	3	29.252 $\pm$ 3.211	26.761	81.3 $\pm$ 4.8
	50	3	98.598 $\pm$ 3.875	94.535	2033.3 $\pm$ 180.6
	100	—	216.240	216.240	8699.8
RC201	10	3	41.385 $\pm$ 2.020	39.560	82.3 $\pm$ 0.9
	50	3	127.233 $\pm$ 17.567	102.398	1895.1 $\pm$ 22.4
	100	—	297.857	297.857	8358.6

In terms of running time, the inference time increases nonlinearly with scale, which is consistent with the computational complexity of GAT neighborhood aggregation over the full graph. Meanwhile, the time difference among the three instance types at the same scale is small, proving the algorithm has good computational stability for different spatial distributions.

To improve result traceability, Table 13 presents the detailed evaluation of each derived sub-instance. In most instances, the route distances of three sub-instances are close. This shows that the stratified sampling strategy maintains the spatial characteristics of the original instance and keeps the solution difficulty of sub-instances at a similar level. In terms of iteration steps, the steps exactly match the customer number n : 10 steps for $n = 10$, 50 steps for $n = 50$ and 100 steps for $n = 100$. It verifies that under deterministic inference, the policy network visits exactly one customer node per step without repetition or omission.

The above results also indicate the scalability limitation of the proposed framework. As the number of customers increases from 10 to 100, the number of decision steps increases accordingly, and the inference time also rises. The computational burden mainly comes from graph attention encoding over the delivery graph, PPO-based sequential policy inference, pseudo-node candidate expansion, and UAV sortie feasibility checking. In terms of memory consumption, the required storage increases with the number of nodes, edges, pseudo-nodes, node embeddings, and attention coefficients. Therefore, although the proposed framework is effective for the tested benchmark instances, direct application to much larger logistics systems would require further scalability improvements, such as sparse graph encoding, Top-K candidate screening, decomposition strategies, and rolling-horizon rescheduling.

Table 13. Detailed evaluation of GAT-PPO.

Instance	n	Seed	Distance (km)	Running Time (ms)	Steps
C101	10	0	37.695	78.2	10
		1	37.243	89.6	10
		2	45.679	82.4	10
	50	0	109.673	1835.3	50
		1	100.016	1884.4	50
		2	118.862	1957.4	50
	100	—	230.711	8561.2	100
R201	10	0	26.761	76.4	10
		1	33.084	87.9	10
		2	32.910	79.7	10
	50	0	97.445	1903.2	50
		1	94.535	2288.7	50
		2	103.814	1908.1	50
	100	—	216.240	8699.8	100
RC201	10	0	40.394	81.1	10
		1	39.560	83.4	10
		2	44.202	82.4	10
	50	0	139.082	1904.8	50
		1	102.398	1916.3	50
		2	140.219	1864.1	50
	100	—	297.857	8358.6	100

5. Conclusions

This paper investigates the route optimization problem of single-truck and single-UAV collaborative delivery considering en route synchronization. To reduce the waiting loss caused by fixed-node rendezvous, a pseudo-node insertion mechanism is introduced to allow the UAV to be recovered at flexible positions along truck arcs. Based on this mechanism, a collaborative scheduling framework combining Graph Attention Network and Proximal Policy Optimization is developed. The optimization objective considers both completion time and total tardiness, and the proposed framework jointly coordinates truck routing, UAV dispatch, and recovery-point selection. It should be clarified that pseudo-node recovery, GAT, and PPO are not proposed as entirely new standalone methods in this paper; rather, the contribution lies in integrating and adapting these components into a unified hierarchical framework for the studied truck–UAV en route synchronization problem.

Experiments on Solomon benchmark instances show that the proposed GAT-PPO framework achieves strong performance under different customer distributions and problem scales. Compared with Random, NN, NN+2-opt, MLP-PPO, ALNS, GA, and VNS, GAT-PPO obtains the shortest total travel distance, the lowest total tardiness, and the shortest completion time in the tested instances. These results indicate that graph attention encoding helps capture spatial relationships, time-window urgency, and synchronization-related dependencies among nodes. The comparison to MLP-PPO further confirms the contribution of graph-structured representation to improving the PPO policy. In addition, the pseudo-node mechanism consistently reduces total tardiness and completion time, demonstrating its effectiveness in improving truck–UAV temporal matching.

The ablation experiments further verify the contribution of the main components of the proposed framework. Removing PPO, GAT, pseudo-node insertion, en route synchronization, or UAV collaboration all increases the objective value, indicating that each module contributes to the final performance. PPO supports sequential decision-making, GAT improves graph-structured spatial–temporal representation, pseudo-node insertion increases the flexibility of UAV recovery, and en route synchronization reduces coordination

loss between the truck and the UAV. These results show that the proposed framework benefits from the integration of learning-based decision-making and synchronization-oriented routing design. Theoretical and managerial implications can be drawn from these findings. From a theoretical perspective, this study provides an integrated framework that combines pseudo-node-based en route synchronization, the temporal simulation of waiting propagation, graph attention encoding, and PPO-based sequential decision-making for truck–UAV collaborative routing. This helps extend truck–UAV synchronization research from fixed-node recovery toward arc-based flexible recovery and graph-based learning decisions. From a managerial perspective, the proposed framework provides useful insights for logistics planning. When fixed-node UAV recovery causes truck waiting and downstream delay propagation, allowing UAV recovery at flexible positions along truck travel arcs can improve temporal coordination. Therefore, en route synchronization is more suitable for benchmark delivery scenarios with tight time windows, dispersed customers, and high sensitivity to service delay.

Several limitations should also be acknowledged. First, the present study is based on a simplified single-truck and single-UAV scenario, and each UAV sortie serves only one customer. Therefore, the results mainly demonstrate the methodological value of en route synchronization and pseudo-node recovery under a controlled benchmark setting, rather than proving direct applicability to large-scale real logistics systems. Second, the current UAV and traffic models remain simplified. UAV endurance is represented by a fixed mission-time threshold, while detailed battery consumption, payload-dependent energy use, wind and weather effects, take-off and landing energy, and time-dependent traffic congestion are not explicitly considered. Third, customer demand and time windows are assumed to be known in advance, so dynamic requests, cancellations, communication delays, and online disturbances are not covered. In addition, a full robustness or sensitivity analysis under stochastic demand, weather, traffic, and energy-parameter variations remains necessary.

The proposed GAT-PPO framework also has limitations in computational efficiency and scalability. In the comparative experiments, the mean runtime of GAT-PPO is 31.937 s, which is higher than those of ALNS, GA, VNS, and MLP-PPO. The additional computational cost mainly comes from graph attention encoding, PPO-based sequential policy inference, and UAV sortie feasibility checking with pseudo-node candidates. Therefore, GAT-PPO is not stated to be faster than mature heuristic, metaheuristic, or standard reinforcement-learning baselines in the current static benchmark setting. When the problem is extended to larger logistics networks involving multiple trucks and multiple UAVs, the computational burden will further increase due to multi-vehicle task allocation, expanded synchronization decisions, and more complex feasibility constraints. Future research should extend the framework to multi-truck and multi-UAV collaborative systems, incorporate energy-aware UAV modeling, time-dependent road travel times, weather and communication constraints, and improve scalability through sparse graph encoding, Top-K candidate screening, decomposition strategies, hybrid reinforcement-learning and metaheuristic search, multi-agent decision-making, robust optimization, and rolling-horizon rescheduling mechanisms.

Author Contributions: Conceptualization, S.Z., G.M. and H.Y.; data curation, S.Z. and Z.D.; formal analysis, S.Z. and Y.L.; funding acquisition, G.M. and H.Y.; investigation, Y.L. and Z.D.; methodology, S.Z., G.M. and H.Y.; project administration, H.Y.; supervision, H.Y.; validation, Y.C.; visualization, Z.D. and Y.C.; writing—original draft, S.Z.; writing—review and editing, G.M. and H.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 52572341), project title: “Research on Coupling Optimization of Urban Underground and Ground Logistics Distribution Network Based on Subway”, direct cost: RMB 500,000, project period: January 2026 to December 2029.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

Abbreviation	Full Form
UAV	Unmanned aerial vehicle
VRPTW	Vehicle routing problem with time windows
GAT	Graph Attention Network
PPO	Proximal Policy Optimization
GAT-PPO	Graph Attention Network-Proximal Policy Optimization
MLP-PPO	Multi-Layer Perceptron-Proximal Policy Optimization
NN	Nearest neighbor
TPE	Tree-structured Parzen Estimator
GPU	Graphics processing unit
VRAM	Video random-access memory
RAM	Random-access memory
CUDA	Compute Unified Device Architecture

References

1. Bryan, V. Drone Delivery: DHL ‘Parcelcopter’ Flies to German Isle. Available online: <https://www.reuters.com/article/world/drone-delivery-dhl-parcelcopter-flies-to-german-isle-idUSKCN0HJ1EC/> (accessed on 1 January 2024).
2. Rubin, B.F. UPS Tucks Delivery Drone into Delivery Truck to Make a Delivery. Available online: <https://www.cnet.com/tech/tech-industry/ups-workhorse-delivery-drone-test-lithia-florida/> (accessed on 1 January 2024).
3. Stewart, J. Google Tests Drone Deliveries in Project Wing Trials. Available online: <https://www.bbc.com/news/technology-28964260> (accessed on 1 January 2024).
4. Chikhaoui, K.; Ghazzai, H.; Massoud, Y. PPO-based reinforcement learning for UAV navigation in urban environments. In Proceedings of the 2022 IEEE 65th International Midwest Symposium on Circuits and Systems (MWSCAS), Fukuoka, Japan, 7–10 August 2022; pp. 1–4.
5. Yang, Q.Y.; Lu, J.L.; Zhang, Y.W. Multi-UAV cooperative path planning via graph neural network and proximal policy optimization. *IEEE Access* **2025**, *13*, 193575–193588. [CrossRef]
6. Scott, J.E.; Scott, C.H. Models for drone delivery of medications and other healthcare items. *Int. J. Healthc. Inf. Syst. Inform.* **2018**, *13*, 20–34. [CrossRef]
7. Giang, L.T.; Viet, V.H.; Tung, N.X. Vehicle routing problems via quantum graph attention network deep reinforcement learning. *arXiv* **2025**, arXiv:2511.15175.
8. Murray, C.C.; Chu, A.G. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transp. Res. Part C Emerg. Technol.* **2015**, *54*, 86–109. [CrossRef]
9. Macrina, G.; Di Puglia Pugliese, L.; Guerriero, F.; Laporte, G. Drone-aided routing: A literature review. *Transp. Res. Part C Emerg. Technol.* **2020**, *120*, 102762. [CrossRef]
10. Agatz, N.; Bouman, P.; Schmidt, M. Optimization approaches for the traveling salesman problem with drone. *Transp. Sci.* **2018**, *52*, 965–981. [CrossRef]
11. Wang, X.; Poikonen, S.; Golden, B. The vehicle routing problem with drones: Several worst-case results. *Optim. Lett.* **2017**, *11*, 679–697.

12. Ferrandez, S.M.; Harbison, T.; Weber, T. Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. *J. Ind. Eng. Manag.* **2016**, *9*, 374–388. [\[CrossRef\]](#)
13. Carlsson, J.G.; Song, S.Y. Coordinated logistics with a truck and a drone. *Manag. Sci.* **2018**, *64*, 4052–4069. [\[CrossRef\]](#)
14. Sacramento, D.; Pisinger, D.; Ropke, S. An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transp. Res. Part C Emerg. Technol.* **2019**, *102*, 289–315. [\[CrossRef\]](#)
15. Luo, Z.H.; Liu, Z.; Shi, J.M. A two-echelon cooperated routing problem for a ground vehicle and its carried unmanned aerial vehicle. *Sensors* **2017**, *17*, 1144. [\[CrossRef\]](#) [\[PubMed\]](#)
16. Murray, C.C.; Raj, R. The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. *Transp. Res. Part C Emerg. Technol.* **2020**, *110*, 368–398. [\[CrossRef\]](#)
17. Moshref-Javadi, M.; Hemmati, A.; Winkenbach, M. A truck and drones model for last-mile delivery: A mathematical model and heuristic approach. *Appl. Math. Model.* **2020**, *80*, 290–318. [\[CrossRef\]](#)
18. Daknama, R.; Kraus, E. Vehicle routing with drones. *arXiv* **2017**, arXiv:1705.06431.
19. Kitjacharoenchai, P.; Ventresca, M.; Moshref-Javadi, M. Multiple traveling salesman problem with drones: Mathematical model and heuristic approach. *Comput. Ind. Eng.* **2019**, *129*, 14–30. [\[CrossRef\]](#)
20. Cui, H.P.; Li, K.Y.; Jia, S. Dynamic collaborative truck-drone delivery with en-route synchronization and random requests. *Transp. Res. Part E Logist. Transp. Rev.* **2024**, *192*, 103802. [\[CrossRef\]](#)
21. Wang, Z.; Sheu, J.B. Vehicle routing problem with drones. *Transp. Res. Part B Methodol.* **2019**, *122*, 350–364. [\[CrossRef\]](#)
22. Chung, S.H.; Sah, B.; Lee, J. Pseudo node insertion method for synchronization in drone–truck combined operations. *Comput. Ind. Eng.* **2024**, *189*, 109963. [\[CrossRef\]](#)
23. Karak, A.; Abdelghany, K. The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transp. Res. Part C Emerg. Technol.* **2019**, *102*, 427–449. [\[CrossRef\]](#)
24. Moshref-Javadi, M.; Hemmati, A.; Winkenbach, M. A comparative analysis of synchronized truck-and-drone delivery models. *Comput. Ind. Eng.* **2021**, *162*, 107648. [\[CrossRef\]](#)
25. Poikonen, S.; Golden, B.L.; Wasil, E.A. A branch-and-bound approach to the traveling salesman problem with a drone. *INFORMS J. Comput.* **2019**, *31*, 335–346. [\[CrossRef\]](#)
26. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
27. Kool, W.; van Hoof, H.; Welling, M. Attention, learn to solve routing problems. *arXiv* **2019**, arXiv:1803.08475.
28. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer networks. *arXiv* **2015**, arXiv:1506.03134.
29. Schermer, D.; Moeini, M.; Wendt, O. A hybrid VNS/Tabu search algorithm for solving the vehicle routing problem with drones and en route operations. *Comput. Oper. Res.* **2019**, *109*, 134–158. [\[CrossRef\]](#)
30. Vásquez, S.A.; Angulo, G.; Klapp, M.A. An exact solution method for the TSP with drone based on decomposition. *Comput. Oper. Res.* **2021**, *127*, 105127. [\[CrossRef\]](#)
31. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph attention networks. *arXiv* **2018**, arXiv:1710.10903.
32. Marinelli, M.; Caggiani, L.; Ottomanelli, M.; Dell’Orco, M. En route truck-drone parcel delivery for optimal vehicle routing strategies. *IET Intell. Transp. Syst.* **2018**, *12*, 253–261. [\[CrossRef\]](#)
33. Cheng, C.; Adulyasak, Y.; Rousseau, L.M. Drone routing with energy function: Formulation and exact algorithm. *Transp. Res. Part B Methodol.* **2020**, *139*, 364–387. [\[CrossRef\]](#)
34. Ali, Q.; Ullah, I. Joint-node-link mapping for virtual network embedding in 5G/6G environments using GAT-augmented PPO. *Mob. Netw. Appl.* **2025**, *30*, 899–914. [\[CrossRef\]](#)
35. Sejersen, J.L.F.; Suzumura, T.; Kayacan, E. GIANT—Global path integration and attentive graph networks for multi-agent trajectory planning. In Proceedings of the 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hangzhou, China, 19–25 October 2025; pp. 10556–10563.
36. Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv* **2015**, arXiv:1506.02438.
37. Munasinghe, I.; Perera, A.; Deo, R.C. A comprehensive review of UAV-UGV collaboration: Advancements and challenges. *J. Sens. Actuator Netw.* **2024**, *13*, 81. [\[CrossRef\]](#)
38. Nowakowski, M.; Łukaszewicz, A. Unmanned ground vehicle for swampy area reconnaissance with unmanned aerial vehicle support: A conceptual framework. *Adv. Sci. Technol. Res. J.* **2026**, *20*, 373–386. [\[CrossRef\]](#) [\[PubMed\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.