The Journal of Systems & Software 206 (2023) 111854

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

A MILP model on coordinated coverage path planning system for UAV-ship hybrid team scheduling software☆

Xiaopan Zhang [a], Furong Zhang [a], Zheng Tang [a], Xingjun Chen [b,*]

[a] Wuhan University of Technology, Wuhan, Hubei, China
[b] Dalian Naval Academy, Dalian, China

ARTICLE INFO

Article history:
Received 15 February 2023
Received in revised form 18 July 2023
Accepted 12 September 2023
Available online 17 September 2023

Keywords:
UAV-ship hybrid team
Coverage path planning system
Shipborne electric UAV
Scheduling software
Mixed integer linear programming

ABSTRACT

Shipborne unmanned aerial vehicles (UAVs) are safer and more flexible for maritime missions, but frequent recharging is needed during long-term patrols. A coordinated system for path planning between ships and electric UAVs is necessary for efficient large-area coverage. A two-stage approach is proposed to minimize the makespan overall UAVs' flight and the move distance overall ships combinationally. First, the target space is triangularized corresponding to the UAV camera field of view for generating air waypoints. Second, a MILP model is designed to connect suitable air waypoints for UAVs and marine waypoints for ship(s) to form the optimal path for them coordinating the requirements of the area coverage and the UAV recharging. The simulation experiments show the proposed model works for the scenario of either the static or the dynamic motherships in a unified way. In the static mode, the vessels are not migrated and the number of vessels and ship calling points required is the same. In the dynamic model, the ship can be repositioned to recover and recharge the drone, and the task can be accomplished simply by repositioning the ship between waypoints. Dynamic models have better interaction patterns than static models.

1. Introduction

UAV-ship Hybrid Team Scheduling Software is often used to manage and dispatch UAVs and ships in maritime rescue, monitoring, and survey tasks. This type of software completes corresponding tasks by coordinating the operation of drones and ships. UAV and ship scheduling strategies directly affect the efficiency of task completion. Continuously optimizing the scheduling and coordination strategies of UAVs and ships in different mission scenarios is also the key to improving the efficiency of UAV-Ship Hybrid Team Scheduling Software. In particular, The software has a powerful sea coverage path planning system, which has a vital impact on the reliability and success rate of mission completion. The conventional construction method of the sea coverage search is based on shipboard sensor detection, supplemented by manned aerial flight observation. The coordinated overseas coverage path planning system, as constructed, has several shortcomings, including insufficient coverage of sea observations, excessive dispatches of surface ships, high energy consumption during operations, and limited flexibility. To address these issues, the use of shipborne UAV formations to expand the detection range and enhance operational efficiency of the mothership has become a widely-explored direction in the field. In such large-scale missions (Cho et al., 2021; Jensen-Nau et al., 2021; Li et al., 2019), it is crucial to utilize the motion characteristics and onboard sensing capabilities of the UAVs to cover every point in the target area. The problem in question is commonly referred to as the UAV coverage path planning problem, and it is considered a subproblem of coverage path planning (CPP). The takeoff and landing conditions on ship-borne platforms are more challenging than those in land-based environments and also impose higher fuel safety requirements, hindering the widespread use of large fuel-powered drones. However, the endurance of electric drones is limited, and they must make multiple landings on the mothership to recharge during their search mission in large maritime areas. The use of conventional static single-center control mode restricts the flight range of the drone to the center mothership, preventing the formation from fully realizing its scale efficiency advantage. To address this, a multi-mothership bridging mechanism must be employed to allow the drone to transfer multiple times via distributed motherships and reach a larger area or to support the drone's expansion of its endurance boundary through the movement of the mothership. In both formation modes, the flight path planning of the drones must not only consider the movement of other peers but also be coordinated with the position or movement trajectory of the mothership. Currently, there are limited studies on the coordinated coverage path planning system

☆ Editor: Prof. W. Eric Wong.

* Corresponding author.
E-mail addresses: tom_xp@whut.edu.cn (X. Zhang), zhangfr@whut.edu.cn (F. Zhang), 276808@whut.edu.cn (Z. Tang), xjchenmail@163.com (X. Chen).

https://doi.org/10.1016/j.jss.2023.111854

for Maritime Search and Rescue missions that take into account the coordination of the mothership's position at sea.

The paper proposes a two-stage solution for UAV path planning for maritime search and rescue based on the UAV's own motion characteristics and onboard sensing capabilities. The solution takes into consideration the requirements for UAV formation multi-stage charging and landing during a sea search operation. The first stage involves dividing the area based on the UAV's sensing capabilities and forming discrete UAV waypoints. The second stage involves using a mixed integer linear programming model for systematic path planning and optimization. The paper is organized as follows: Chapter 2 provides an overview of the current research in the field; Chapter 3 introduces a mixed integer linear programming model for handling multiple ship scenarios; Chapter 4 establishes a test model to evaluate the optimization results; Chapter 5 concludes with a summary and outlook.

2. Related work

In recent years, many scholars have conducted research on the UAV coverage planning path, and the model types can be roughly divided into the following two categories: methods based on sweeping (Balampanis et al., 2017a; Xie et al., 2020; de Ocampo et al., 2018; Zhan et al., 2018), methods based on area segmentation (Cheng et al., 2022; Cho et al., 2021; Javier et al., 2021; Li et al., 2022; Miller et al., 1960; Zhang and Fu, 2021; Balampanis et al., 2017c). There are two cases of sweep-based methods, one is sweep coverage of a certain area, and the other is sweep coverage of several discrete areas (Xie et al., 2020; de Ocampo et al., 2018). In the first case, the sweep-based method has the advantage of being simple and efficient, and its algorithm design is also relatively simple, which can generate coverage paths in a very short time. There are sharp turns in the region boundary area based on the sweeping method, so there are a lot of researches to solve the U-turn problem (Zhan et al., 2018). In the second scenario, the areas are seen as discrete points and the order of visiting each target point is planned, and then each discrete area is scanned by sweeping. The problem is referred to as the TSP-CPP problem (Xie et al., 2020) by some authors. TSP is commonly used in unmanned aerial vehicle (UAV) path planning. When a limited capacity UAV fleet needs to fulfill a set of spatially distributed demand points, it can be considered a variation of the TSP problem, known as the VRP (Vehicle Routing Problem) (Klein Koerkamp et al., 2019). It plays a crucial role in physical distribution and logistics transportation, but the current VRP model is insufficient to solve drone delivery issues. The article (Dorling et al., 2017) addresses the challenge of last-mile delivery with UAVs, considering factors such as UAV reuse and energy consumption. To determine a set of flight paths for a group of m UAVs, the TSP problem can be extended to the MTSP problem. The integer programming model is a means of solving MTSP problems. The MTSP problem is commonly defined as follows: given a set of nodes, place m salespeople at the warehouse node and have them visit the remaining nodes (cities). The objective of the MTSP problem is to find a tour path for each salesperson, starting from the warehouse, visiting the intermediate nodes only once, and minimizing the total cost of visiting all nodes. Solutions and applications of the MTSP problem can be found in Bektas (2006). Sweep-based method, which considers the energy constraints of UAV, attempts to cover a large area, but this large area often consists of multiple discrete areas rather than a complete one (Xie et al., 2020; de Ocampo et al., 2018). This makes the sweep-based method not suitable for large-scale areas, but appropriate for areas with irregular curved edges. There exist subdivision-based methods to subdivide regions using triangles and squares, and algorithms are designed based on this method to solve the coverage path. These algorithms include graph search, integer programming, and heuristic algorithms. The subdivision-based algorithm is more efficient in covering complex areas, but at the same time, the author generally overlooks the cooperative work of UAVs, energy replenishment and recycling of UAVs. The article (Xie et al., 2020) examines the challenge of coverage planning for unmanned aerial vehicles (UAVs) in complex, non-concave coastal environments. The authors employ constrained Delaunay triangulation to discretize the region into optimized triangles. The circumcenter of each triangle is designated as the waypoint for the UAV, and a graph search algorithm is utilized to create a spiral flight path by visiting each circumcenter. The authors propose that a helical path may yield improved coverage compared to a boustrophedon motion (de Ocampo et al., 2018). Although the graph search algorithm proves effective in covering complex and irregular areas, it does not facilitate collaboration and coordination among multiple UAVs. Additionally, the article does not take into account the energy constraints and recovery of UAVs. In the article (Cho et al., 2021), a mixed linear integer programming (MILP) model is employed to address maritime search and rescue tasks. The search and rescue area is divided into square units and represented as a graph of nodes and edges. The MILP model is designed to optimize the coverage path and minimize the completion time for multiple UAVs. However, the model assumes that all drones depart from a single base and only take off once, making it applicable only to maritime search and rescue operations in small areas.

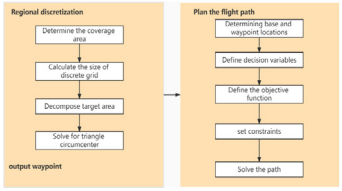

Fig. 1. Research ideas.

In scenarios where a large area needs to be covered for search and rescue or patrol missions, multiple drone bases may be present. If the number of drones is limited, one flight may not be sufficient to complete the task and multiple drone take-offs may be necessary. However, the energy consumption constraints of the UAVs were considered during the design process, but their landing locations were not specified. This can create difficulties for recovery and recharging of the UAVs, hindering their reuse. Another article (Javier et al., 2021) addresses the challenge of obstacle avoidance for UAVs in three-dimensional coverage paths. We propose a novel UAV path planning problem to make the UAV path coverage planning more feasible for large areas. The solution involves allowing the UAV formation to dynamically interact with the mothership during the mission, providing energy resupply to the UAVs as needed. Our research ideas are shown in Fig. 1

3. The proposed models and methods

3.1. Scene of model design

As depicted in Fig. 2, a large sea area A is considered and a set of waypoints for the UAVs is established after decomposition of the area (represented by black dots in Fig. 2). The area contains S positions for the UAV motherships to navigate in (represented by red dots in Fig. 2). Each mother ship is equipped with one or more UAVs fitted with orthographic sensors. The UAVs depart from their respective motherships to conduct surveillance missions for

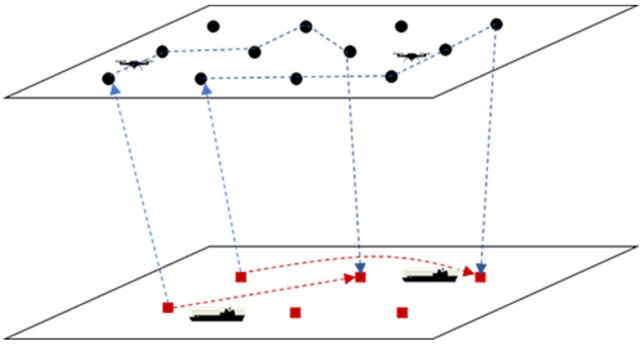

Fig. 2. Study scenario.

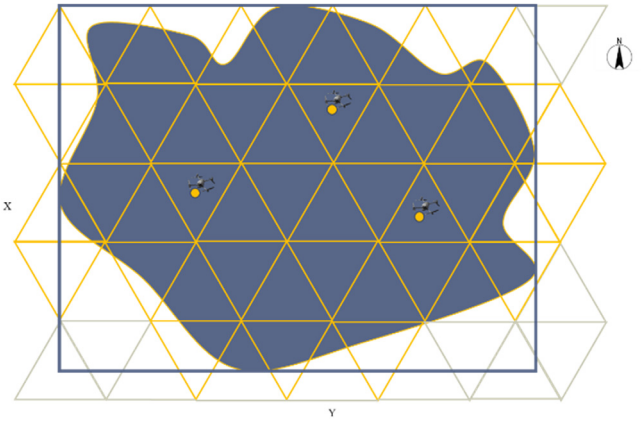

Fig. 3. Area discretization.

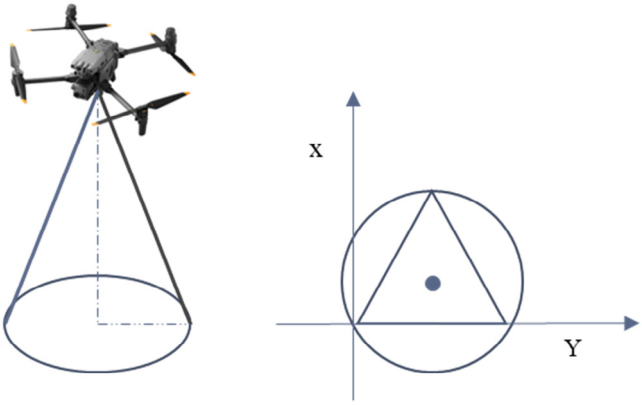

Fig. 4. UAV camera projection area.

the area. During flight, the UAVs maintain a constant altitude and their flight time cannot exceed the maximum limit set by the battery capacity. If the battery level is low, the UAVs can return to the mother ships for recharging. The mother ships can move in the water area when not performing launch and recovery operations. The aim of the path planning is to ensure that the UAVs cover all observation waypoints in the shortest duration possible.

3.2. Region decomposition based on regular triangle

The set of UAV waypoints in Fig. 2 must ensure complete coverage of the target sea area. To achieve this, the circumscribed rectangle of area A is first drawn, and a regular triangulation network is employed to divide the rectangle. The triangles intersecting with area A are retained to form the waypoints (as depicted in Fig. 3). The detailed principle will be described in Section 3.2.

In traditional grid-based path planning research, the sub-region is defined as polygonal grid cells proportional in size to the images captured by the camera sensor. In this study, regular triangles are selected as the regular mesh. The regular triangle is a triangle inscribed in a circle. To ensure that the drone fully covers the area during its movement, the size of the regular triangle must be less than or equal to the area covered by the camera (as shown in Fig. 4). If the drone is flying at a constant speed and altitude, the camera's footprint is circular and its diameter is determined by the drone's altitude and camera specifications (Balampanis et al., 2017b). The diameter of the sensor's sea surface projection is calculated as follows:

$$d = 2\tan(FOV/2) * h \tag{1}$$

where FOV is the field of view of the camera carried by the drone, and h is the flight altitude of the UAV. The circumcenter of the regular triangle is used to replace the regular triangle as the waypoint of UAV flight. The sides of the regular triangle are calculated as:

$$s = d\cos 30^\circ \tag{2}$$

Eqs. (3) and (4) are the formula for solving the outer center of the regular triangle.

$$X = ((C_1 * B_2) - (C_2 * B_1))/((A_1 * B_2) - (A_2 * B_1)) \tag{3}$$

$$Y = ((A_1 * C_2) - (A_2 * C_1))/((A_1 * B_2) - (A_2 * B_1)) \tag{4}$$

where (X, Y) are the coordinates of the circumcenter of the triangle, The calculation formula of A_1, A_2, B_1, B_2, C_1, and C_2 is Eqs. (5)–(10)

$$A_1 = 2 * (x_2 - x_1) \tag{5}$$

$$A_2 = 2 * (x_3 - x_2) \tag{6}$$

$$B_1 = 2 * (y_2 - y_1) \tag{7}$$

$$B_2 = 2 * (y_3 - y_2) \tag{8}$$

$$C_1 = x_2^2 + y_2^2 - y_1^2 - y_1^2 \tag{9}$$

$$C_2 = x_3^2 + y_3^2 - y_2^2 - y_2^2 \tag{10}$$

Where(x_1, y_1), (x_2, y_2), (x_3, y_3) are the coordinates of the three vertices of a triangle.

3.3. Path planning based on mixed integer linear programming

In this section, the paths for covering the nodes defined in section A will be determined. The Coverage Path Planning (CPP) problem is transformed into a problem similar to the Multi Traveling Salesman Problem (MTSP) when the region to be covered is represented as a graph. A Mixed Integer Linear Programming (MILP) model is formulated to minimize the completion time of the UAVs under constraints such as the location of nodes, the dynamics of the UAV, and other relevant factors. The parameters in the model are presented in Table 1 and the decision variables are listed in Table 2.

The proposed mathematical model incorporates the Big M Method for linearization in order to solve the collaborative coverage path planning problem between UAV formations and mother ships over large areas. The objective of the optimization is to minimize the total time taken to complete all observation waypoints while also keeping the mother-ship's path simple. This approach allows for a comprehensive solution to the problem, considering

Table 1
Model parameters.

Parameters	Description
P_0	Set of aerial waypoints (observation waypoints) for drones
P_1	Set of surface waypoints for ships
S	The number of ships
$p_s \in P_1$	The starting point for ship s, $1 \le s \le S$
K	The number of drones.
$s_k \in S$	Initial mother ship for drone k, $1 \le k \le K$
d_{ij}	Distance between waypoints i and j (drone cruise altitude)
h	Drone cruising height
v_k^c	Cruising speed of drone k (m/min)
v_k^u	Take-off speed of drone k (m/min)
v_k^d	Landing speed of drone k (m/min)
v_s	Vessel s sailing speed(m/s)
e_k	Range (time) of the drone k (min)
c_k	Charging time of the drone k (min)
D	Hyperparameter to control inter-waypoint distance threshold of the UAV flight path
N	Hyperparameter to control maximum available flights of each UAV, where a flight includes completely taking off – flying – landing procedure.
M	sufficiently large positive numbers

Table 2
Model decision variables.

Decision variables	Description
$c_{ijkn} \in \{0, 1\}$, $\forall i, j \in P_0, k \in K, n \in N$	1 if the UAV k flies from aerial waypoint i to aerial waypoint j during the nth segment of the cruise, 0 otherwise.
$a_{isjkn} \in \{0, 1\}$, $\forall i \in P_1, j \in P_0, s \in S, k \in K, n \in N$	1 if UAV k takes off from mothership s at water surface waypoint i to aerial waypoint j in the nth segment, 0 otherwise.
$b_{jsikn} \in \{0, 1\}$, $\forall i \in P_1, j \in P_0, s \in S, k \in K, n \in N$	1 if UAV k lands from aerial waypoint j to mothership s at water surface waypoint i in the nth segment, 0 otherwise.
$y_{ijs} \in \{0, 1\}$, $\forall i, j \in P_0, s \in S$	1 if mothership s is heading from water surface waypoint i to water surface waypoint j, 0 otherwise
$u_i \in R$, $\forall i \in P_0$	The waypoint potentials for subtour elimination (Bektas, 2006).
$f_{kn} \in R, \forall k \in K$, $n \in N$	The time at which the drone k is released in the nth segment of the flight
$r_{kn} \in R, \forall k \in K$, $n \in N$	Time for the drone k to land on the ship at the end of the nth segment of the flight
$g_{is} \in R, \forall i \in P_1$, $s \in S$	Time for mothership s to reach waypoint i at the surface waypoint
$q_{is} \in R, \forall i \in P_1$, $s \in S$	Time for mothership s to leave the surface waypoint i

multiple factors such as UAV flight time, mother ship path, and target coverage.

$$\min Z = \max_{\{k,n\}}\{\, r_{kn} - 2M \cdot \left(1 - \sum_{\{i,s,j\}} a_{isjkn}\right)\} + \sum_{\{i,s,j\}} y_{ijs} \tag{11}$$

The model constraints are as follows

(1) Each aerial waypoint does not contain a self-loop

$$c_{iikn} = 0, \forall i \in P_0, k \in K, n \in N \tag{12}$$

(2) Surface waypoint lines do not contain self-loops

$$y_{iis} = 0, \forall i \in P_0, s \in S$$

(3) Avoiding long distance sequential UAV waypoints in a route based on the experience that the UAV tends to visit neighbor points. An appropriate distance threshold, symboled as D, can shrink the search space without loss of optimization.

$$c_{ijkn} \cdot d_{ij} \le D, \forall i, j \in P_0, k \in K, n \in N \tag{13}$$

(4) No loops per flight (maximum of one flight into each waypoint) and the drone can only carry out its mission after it has been released

$$\sum_{\{i \in P_0\}} c_{ijkn} + \sum_{\{i \in P_1, s \in S\}} a_{isjkn} \le \sum_{\{i \in P_1, s \in S, j \in P_0\}} a_{isjkn}, \forall j \in P_0, k \in K, n \in N \tag{14}$$

(5) The drone flies in from waypoint ***i*** and out from waypoint ***i*** during flight.

$$\sum_{\{i \in P_0\}} c_{ijkn} + \sum_{\{i \in P_1, s \in S\}} a_{isjkn} = \sum_{\{i \in P_0\}} c_{jikn} + \sum_{\{i \in P_1, s \in S\}} b_{jsikn}, \forall j \in P_0, k \in K, n \in N \tag{15}$$

(6) In each flight segment, the drone takes off at most once

$$\sum_{\{i \in P_1, s \in S, j \in P_0\}} a_{isjkn} \le 1, \forall k \in K, n \in N \tag{16}$$

(7) In each flight segment, the number of drone landings is the same as the number of take-offs

$$\sum_{\{i \in P_1, s \in S, j \in P_0\}} a_{isjkn} = \sum_{\{i \in P_1, s \in S, j \in P_0\}} b_{jsikn}, \forall k \in K, n \in N \tag{17}$$

(8) The drone can only be released again from the recovery ship

$$\sum_{\{i \in P_1, j \in P_0\}} b_{jsikn} \ge \sum_{\{i \in P_1, j \in P_0\}} a_{isjk(n+1)}, \forall k \in K, s \in S, n \in N - 1 \tag{18}$$

(9) The launch ship for the first flight of the UAV is given

$$a_{isjk1} = 0, \forall s \ne s_k, i \in P_1, j \in P_0, k \in K \tag{19}$$

(10) The UAV covers each observed point at least once during the flight

$$\sum_{\{i \in P_0, k \in K, n \in N\}} c_{ijkn} + \sum_{\{i \in P_1, s \in S, k \in K, n \in N\}} a_{isjkn} \ge 1, \forall j \in P_0 \tag{20}$$

(11) Each ship waypoint does not contain a self-loop

$$y_{iis} = 0, \forall i \in P_1, s \in S \tag{21}$$

There is no loop on the ship's route (each waypoint passes at most once), and it needs to start from the starting point

$$\sum_{\{i \in P_1\}} y_{ijs} \le \sum_{\{j \in P_1 \setminus p_s\}} y_{p_s js}, \forall j \in P_1 \setminus p_s, s \in S \tag{22}$$

(12) The ship departs from the starting point no more than once (no loop, it will not return to the starting point during the mission execution)

$$\sum_{\{j \in P_1\}} y_{p_s js} \le 1, \forall s \in S \tag{23}$$

(13) After the ship enters a certain waypoint, it needs to depart from the waypoint or the voyage ends at the waypoint

$$\sum_{\{i \in P_1\}} y_{ijs} \ge \sum_{\{i \in P_1\}} y_{jis} \forall j \in P_1 \setminus p_s, s \in S \tag{24}$$

(14) The time when the ship arrives at the starting point is 0

$$g_{p_s s} = 0, \forall s \in S \tag{25}$$

(15) If the ship has no route to reach a waypoint, set the arrival time to M

$$g_{js} \geq (1 - \sum_{\{i \in P_1\}} y_{ijs}) \cdot M, \forall j \in P_1 \backslash p_s, s \in S \tag{26}$$

(16) The time of arrival of a ship at a waypoint is related to the departure time of the previous waypoint

$$g_{js} - q_{is} \geq y_{ijs} \cdot \frac{d_{ij}}{v_s} - M \cdot \left(1 - y_{ijs}\right), \forall i \in P_1, j \in P_1 \backslash p_s, s \in S \tag{27}$$

(17) A ship arrives at a waypoint earlier than it leaves

$$q_{is} \geq g_{is}, \forall i \in P_1 \backslash p_s, s \in S \tag{28}$$

(18) If the drone is not performing a segment of the mission, its release and recovery time is set to M

$$f_{kn} \geq \left(1 - \sum_{\{i \in P_1, s \in S, j \in P_0\}} a_{isjkn}\right) \cdot M, \forall k \in K, n \in N \tag{29}$$

$$r_{kn} \geq \left(1 - \sum_{\{i \in P_1, s \in S, j \in P_0\}} b_{jsikn}\right) \cdot M, \forall k \in K, n \in N \tag{30}$$

(19) If the UAV performs a certain mission, its release (or recovery) time should be later than the time when the ship arrives at the release waypoint, and earlier than the

(20) time when the ship leaves the release point

$$g_{is} - \left(1 - \sum_{\{j \in P_0\}} a_{isjkn}\right) \cdot M \leq f_{kn} \leq q_{is} + \left(1 - \sum_{\{j \in P_0\}} a_{isjkn}\right) \cdot M, \forall i \in P_1, s \in S, k \in K, n \in N \tag{31}$$

$$g_{is} - \left(1 - \sum_{\{j \in P_0\}} b_{jsikn}\right) \cdot M \leq r_{kn} \leq q_{is} + \left(1 - \sum_{\{j \in P_0\}} b_{jsikn}\right) \cdot M, \forall i \in P_1, s \in S, k \in K, n \in N \tag{32}$$

(21) The recovery time of the drone is later than the release time

$$r_{kn} \geq f_{kn}, \forall k \in K, n \in N \tag{33}$$

(22) Drone energy constraint: the drone cannot exceed its energy consumption in each segment of flight

$$r_{kn} - f_{kn} \leq e_k, \forall k \in K, n \in N \tag{34}$$

(23) The difference between the release time of the drone and the landing time of the previous phase should be greater than the charging time

$$f_{k(n+1)} - r_{kn} \geq c_k \cdot \left(\sum_{\{i \in P_1, s \in S, j \in P_0\}} a_{isjk(n+1)}\right), \forall k \in K, n \in N \tag{35}$$

Table 3
UAV properties.

Property	Value
FPV camera	(FOV) 145°
Maximum horizontal flight speed	S Mode:23 m/s
Maximum ascending speed	S Mode:6 m/s
Maximum descending speed (Vertical)	S Mode:5 m/s
Maximum flight time	S Mode:5 m/s
Drone charging time	45 min

(24) The drone flight time is equal to the sum of the flight time of each range.

$$r_{kn} - f_{kn} = \sum_{\{i \in P_0, j \in P_0\}} c_{ijkn} * \frac{h}{v_k^u} + \sum_{\{i \in P_1, j \in P_0, s \in S\}} \left[a_{isjkn} \cdot \left(\frac{h}{v_k^u} + \frac{d_{ij}}{v_k^c}\right) + b_{jsikn} \cdot \left(\frac{h}{v_k^u} + \frac{d_{ij}}{v_k^c}\right)\right] \tag{36}$$

(25) Subtour elimination constraints (SECs) are necessary to TSP-like problems (Bektas, 2006), which are typically formalized by the waypoint potentials. Here we propose the formulas (37–39) that can generate fewer constraints than the ones in Bektas (2006) for large number of K and N.

$$u_i \geq 0 \tag{37}$$

$$u_i \leq M \cdot \left(1 - \sum_{\{i \in P_1, s \in S, j \in P_0, k \in K\}} a_{isjkn}\right), \forall j \in P_1 \tag{38}$$

$$u_j \geq u_i + 1 - M \cdot \left(1 - \sum_{\{k \in K, n \in N\}} c_{ijkn}\right), \forall i, j \in P_0 \tag{39}$$

4. Experiments analysis

4.1. Motivation

The Mixed-Integer Linear Programming (MILP) model has the potential to provide a global optimal solution. However, due to the NP-hard complexity of multi-UAV path planning, solving the MILP model within a reasonable time frame becomes challenging, especially for large-scale scenarios. To address this issue, three hyperparameters can be employed to generate an approximate optimal solution. Specifically, these hyperparameters are D and N, as depicted in Table 1, which serve to reduce the solution space, and the early stop time denoted as T, which limits the solving duration. The experimental setup aims to evaluate the effectiveness of these hyperparameters and illustrate heuristic rules for assignment.

4.2. Environment and scenario settings

To ensure generality, we consider a rectangular sea area measuring 21 000 m x 17 000 m, as depicted in Fig. 5. Within this area, six ship waypoints are distributed regularly, represented by red blocks. For simulation purposes, our virtual UAV emulates the DJI R300RTK UAV, characterized by the properties outlined in Table 3. We assume that the UAV conducts horizontal patrols at an altitude of 1000 m during each flight. Consequently, there exist 36 UAV waypoints indicated by green dots in Fig. 5. For reference, all the coordinates pertaining to the environment can be found in Appendix A.

Accordingly, certain parameters in Table 1 can be established, as presented in Table 4, to serve as the fundamental settings

Table 4
Assignment of model parameters.

Parameters	Value
P_0	UAV waypoints set, $\{p_6 \sim p_{41}\}$ in Appendix 1
P_1	Ship waypoints set, $\{p_0 \sim p_5\}$ in Appendix 1
S	1.
p_{s_0}	p_0 for all experiments, which means the $ship_0$ always moving from the up-left waypoint in Fig. X
s_k	$ship1$ for all UAVs, which means all the UAVs king on $ship1$ initially.
d_{ij}	Euclidean distance between the projection points of p_i and p_j to horizontal plane
h	1.0 (km)
v_k^c	1.38 (km/min)
v_k^u	0.36 (km/min)
v_k^d	0.3 (km/min)
v_s	0.463 (km/min)
e_k	45 (min)
c_k	10 (min)

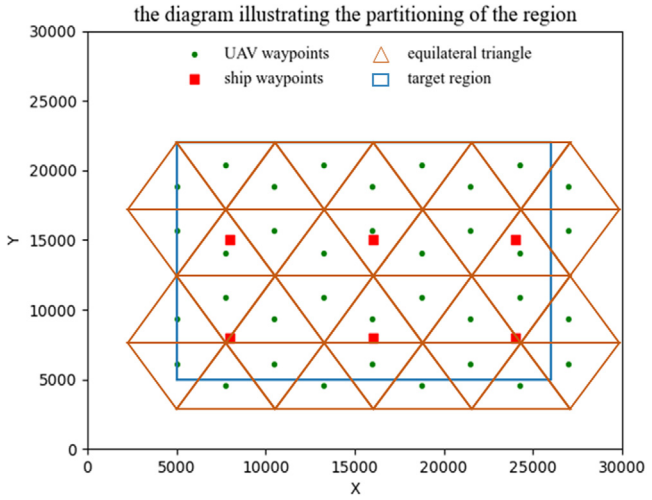

Fig. 5. Illustration of region partitioning and waypoint generation.

for all experiments. Consequently, the experiment can be defined by five hyperparameters: K (the number of UAVs), D (the inter-waypoint distance threshold), N (the maximum available flights), and T (the early stop solving time) from Table 1. These hyperparameters collectively shape the experiment denoted as D(K, D, N, T), representing a "dynamic system" with specific hyperparameters. In comparison, modified experiments referred to as the "static system" are conducted where a ship is fixed at each ship waypoint, thus serving as the baseline. These modified experiments are represented by the expression S(K, D, N, T), which corresponds to specific hyperparameters in the "static system" and can be easily generated using our proposed model.

All experiments are executed using the academic version of the CPLEX solver and implemented in Python 3.6. The computational system utilized for the experiments consists of an 11th Gen Intel(R) Core(TM) i7-11700 @2.5 GHz processor and 16 GB of memory.

4.3. Dynamic system performance assessment

The static system, where each ship waypoint is associated with a ship, establishes a lower bound for the dynamic system, where ships move among waypoints to intercept UAVs. As a result, four sets of comparative experiments are conducted to evaluate the performance of the dynamic system in the predefined environment. The settings and results of these experiments are presented

Table 5
Comparison between dynamic system and static system.

K	N	Objective		
		S (K, D, N, T) with 6 ships	D (K, D, N, T) with only 1 ship	D/ S
1	3	121.264	124.430	97.5%
2	2	65.590	73.602	89.1%
3	1	36.269	40.360	89.9%
4	1	30.328	35.560	85.3%

Table 6
Performance corresponding to N settings with respect to K.

K	N	Objective Value
1	2	Infeasible
1	**3**	**124.4482**
1	4	157.4731
1	5	146.119
2	1	Infeasible
2	**2**	**81.565**
2	3	91.335
2	4	93.500
3	**1**	**41.417**
3	2	85.316
3	3	118.212
3	4	1076.391
4	**1**	**35.560**
4	2	71.387
4	3	1001.000

in Table 5, where D and T are fixed at 5493.37(m) and 1500(s) respectively. Additionally, the hyperparameter N is determined based on the value of K, as indicated in Table 6.

The assignment of hyperparameters is derived from the statistical analysis of hyperparameter utility assessment experiments. The path solutions of UAVs (and ships) in the static and dynamic systems are illustrated in Fig. 7 and Fig. 8 respectively. The results demonstrate that the model generates accurate paths, and the dynamic system achieves more than 85% performance compared to the static baseline, while utilizing significantly fewer ships.

4.4. Early stop performance assessment

The CPLEX solver offers a feature that allows the definition of an early termination time to obtain the best solution before a specified deadline, particularly beneficial for large-scale models. When employing the branch and bound algorithm, the output solution typically exhibits a monotonically decreasing trend as the solution time increases. Thus, the hyperparameter T, representing the early stop time, can be utilized to control the solution quality.

To compare the outputs obtained with different early stop times, four sets of experiments were conducted using 1, 2, 3, and 4 drones, respectively. In each set, the hyperparameter D was fixed at 5493.37 (m), and the solver ran for 1500 (s). The hyperparameter N is set with respect to K as the rule in Table 5. During the solving process, every time spot when a better solution is found was recorded. Then the iterative curve by connecting solution points over time is shown as Fig. 6. All the curves illustrate rapid optimization phase followed by a transition to a low-gradient phase very soon. Thus, two typical values of hyperparameter T were assigned heuristically in the following experiments, where $T = 100$ (s) for the quick solving cases and $T = 1500$ (s) for the best solving cases.

4.5. Inter-waypoint distance threshold assessment

The hyperparameter D serves to impose a restriction on the path, ensuring that the waypoints are not further apart from each

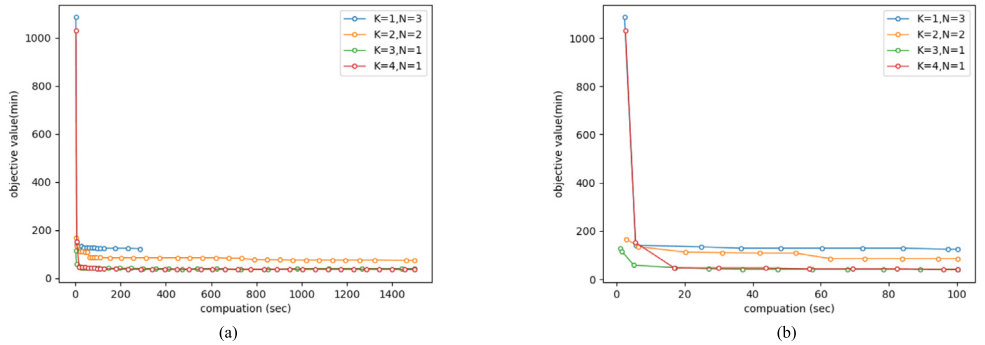

Fig. 6. Iterative Curves of CPLEX solving, where (a) shows iteration within 1500 (s) and (b) shows iteration within 100 (s).

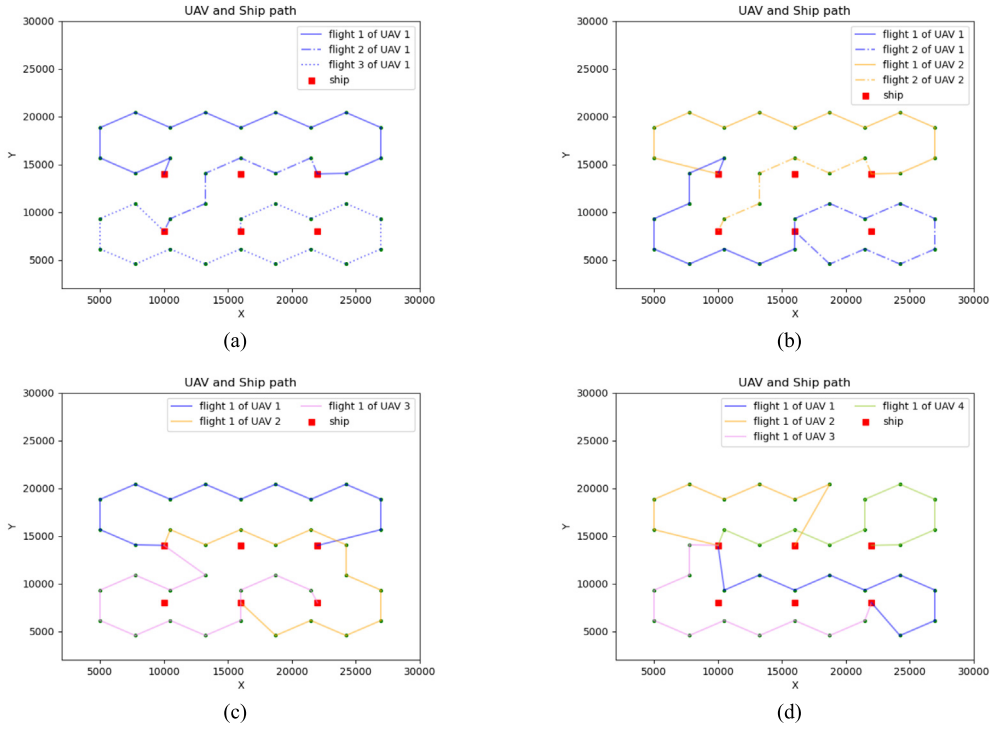

Fig. 7. Flights of UAVs in static system: (a) 1 UAV (b) 2 UAVs (c) 3 UAVs (d) 4 UAVs.

other than D. However, setting a smaller value for D reduces the diversity of path solutions and increases the likelihood of missing the optimal solution. Conversely, a larger value of D increases the risk of not finding a feasible solution within a practical time frame. Based on the statistics of the environment depicted in Fig. 5, it is observed that there are 23 distinct values between any two UAV waypoints.

Consequently, 23 batches of experiments are conducted, with each batch utilizing one of the available values of D. Additionally, each batch is executed under two different early stop assignments: $T = 200$(s) and $T = 1500$(s). The experiments cover teams consisting of 1 to 4 UAVs. As presented in Table 7, setting the minimum value of D, which only allows paths connecting the closest neighbors, renders the model infeasible under all conditions. On the other hand, employing a large value of D poses a high risk of failing to find a feasible solution within a practical duration. Generally, the second-grade value of D, which is 5493.37(m) in this case, strikes the best balance between efficiency and solution quality.

4.6. Comparison with heuristic algorithms

4.6.1. SA algorithm

The Simulated Annealing (SA) algorithm is a global optimization algorithm based on the process of simulated annealing, which is used to search for the optimal solution in a large search space (Kirkpatrick et al.). The SA algorithm can avoid getting stuck in local optima by accepting suboptimal solutions with a lower probability (Laarhoven and Aarts, 1987). The objective function for optimizing the model using the SA algorithm is given by:

$$\begin{aligned}\min Z = &\max_{\{k,n\}}\{r_{kn} - 2M \cdot (1 - \sum_{\{i,s,j\}} a_{isjkn})\} + 10 \\ &* \max\left((r_{kn} - f_{kn}) - e_k, 0\right) + 10 \\ &* \max(y_{ijs} * \frac{d_{ij}}{v_s} - (r_{kn} - f_{kn}), 0)\end{aligned}$$

The objective function consists of three components. The first component aims to minimize the total completion time of the

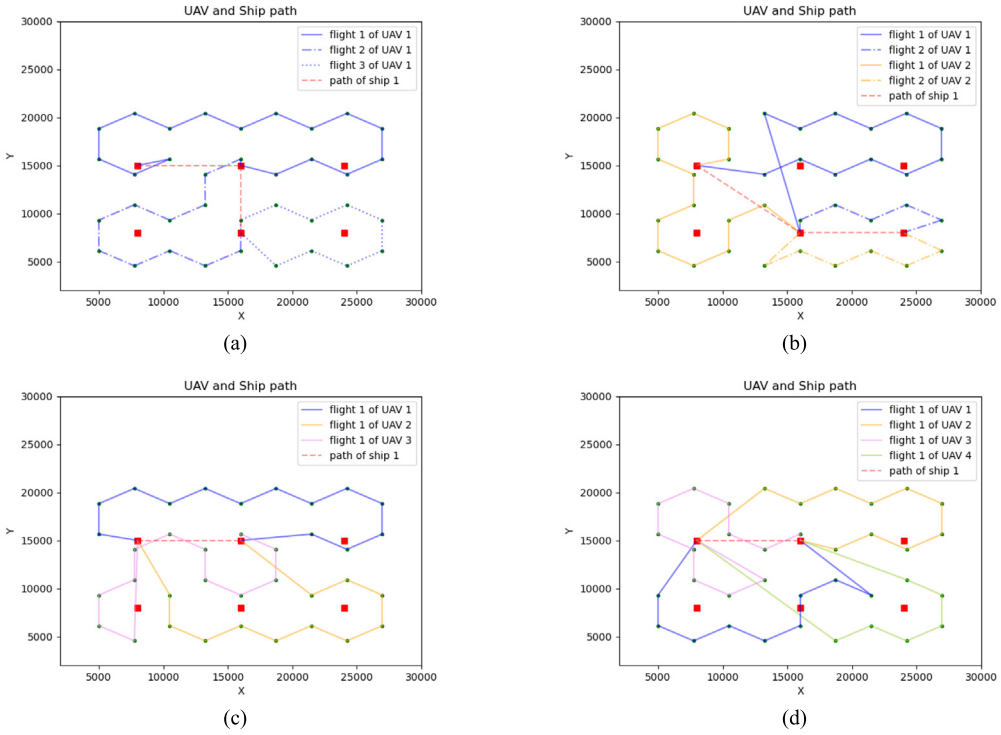

Fig. 8. Flights of UAVs and navigation path of ship in dynamic system: (a) 1 UAV (b) 2 UAVs (c) 3 UAVs (d) 4 UAVs.

Table 7
Performance of different D Values.

D (m)	Objective Value							
	T = 200 (s)				T = 1500 (s)			
	K = 1	K = 2	K = 3	K = 4	K = 1	K = 2	K = 3	K = 4
3171.60	Infeasible	Infeasible	Infeasible	Infeasible	Infeasible	Infeasible	Infeasible	Infeasible
5493.36	**124.43***	**84.72**	41.42	**37.54**	**124.43***	40.36	40.36	35.56
6343.19	125.60	-	68.42	-	125.60	39.22	39.22	-
8391.25	141.68	-	**40.08**	38.81	124.43	39.23	39.23	**35.15**
9514.78	128.46	-	-	36.70	127.96	-	-	35.24
10986.73	126.03	94.81	-	144.16	124.43	40.07	40.07	38.84
11435.35	140.12	-	-	94.30	124.93	-	-	36.10
12686.38	140.12	-	-	68.02	129.49	-	-	35.81
13824.66	139.10	-	-	-	124.43	**38.86**	**38.86**	-
14534.07	124.43	119.37	-	43.62	124.43	-	-	34.55
15857.97	124.43	119.37	-	44.34	124.43	-	-	34.55
16480.09	135.19	1073.33	-	-	124.44	-	-	-
16782.50	128.48	-	-	-	126.17	-	-	-
17658.69	134.16	-	-	78.65	125.98	-	-	35.51
19029.57	131.84	-	104.18	91.93	126.75	57.81	57.81	35.81
19292.06	171.40	126.41	-	-	125.60	-	-	-
19806.60	137.88	-	-	-	126.93	-	-	-
20797.53	132.48	-	-	1000.00	124.43	-	-	38.77
21973.45	141.41	152.33	-	1003.00	124.43	-	-	38.97
22201.16	141.41	117.88	-	45.73	124.43	-	-	39.21
22870.69	129.27	-	-	-	126.36	-	-	-
23945.02	135.61	164.01	-	-	127.83	-	-	-
25372.76	140.95	1031.76	-	-	132.77	-	-	-

* The numbers marked with an asterisk (*) indicate that the value has reached the optimal solution.
* The hyphen (-) in the table represents that the model did not find a feasible solution.

drones for the task. The second component ensures that the flight time for each segment of the drones does not exceed their energy capacity. The third component ensures that the travel time for each segment of the mother ships does not exceed the flight time of the drones.

SA algorithm's pseudo code is depicted in Appendix B.

4.6.2. NSGA-II

NSGA-II (Non-dominated Sorting Genetic Algorithm II) is a multi-objective optimization algorithm that was developed in 2002 by Deb et al. (2002). It is an improvement over genetic algorithms and is specifically designed to solve multi-objective optimization problems. The NSGA-II algorithm has also been applied

Table 8
Comparison of solution quality between Cplex Exact Solver, NSGA_II, and SA Algorithm.

Scenarios		Cplex		NSGA_II		SA	
Number of drones	Number of segments	Objective value (min)	Solving time (s)	Objective value (min)	Solving time (s)	Objective value (min)	Solving time (s)
1	3	**124.4482**	180	378.035	180	646.890	25
2	2	**78.279**	180	304.367	180	657.9911	25
3	1	**41.417**	180	172.638	180	368.4048	25
4	1	**36.920**	180	137.597	180	391.4221	25

to solve the multi-objective MTSP (Multiple Traveling Salesman Problem) (Bolaños et al., 2015). In this study, we will employ the NSGA-II algorithm to solve the model we have constructed. The NSGA-II algorithm finds a set of solutions ranked by the Pareto dominance concept. When solving the model proposed in this article, we need to encode the chromosomes (solutions) based on our model and define objective functions that align with the proposed model. The NSGA-II algorithm utilizes two objective functions for this purpose:

$$\min z = \max_{\{k,n\}}\{\, r_{kn} - 2M \cdot (1 - \sum_{\{i,s,j\}} a_{isjkn})\}$$

The objective function 1 aims to minimize the maximum landing time of the drones.

$$\min E = 10 * \max((r_{kn} - f_{kn}) - e_k, 0) + 10 \\ {}^{*}\max(y_{ijs} * \frac{d_{ij}}{v_s} - (r_{kn} - f_{kn}), 0)$$

The objective function 2 ensures that the flight time for each segment of the drones does not exceed their energy consumption and that the travel time for each segment of the mother ships does not exceed the flight time of the drones. SA algorithm's pseudo code is depicted in Appendix C.

4.6.3. Quality comparison of solutions

The SA algorithm quickly converges within a short time but generates the worst solution. The NSGA_II algorithm completes iterations within 180 s, so the experiment compared the results of the Cplex exact solver and the NSGA_II algorithm within 180 s. Compared to heuristic algorithms, the Cplex solver produces better solutions within the same running time. (See detailed results in Table 8). Both NSGA_II and SA algorithms are non-exact solvers. The reason they cannot achieve results comparable to the exact solutions of Cplex is that traditional NSGA_II and SA algorithms generate random initial solutions, which may not be feasible for the model. Therefore, the quality of optimized solutions based on these initial solutions is far inferior to the exact solutions obtained by the Cplex solver.

5. Conclusion and outlook

In summary, the two-stage approach proposed in this study addresses the problem of cooperative coverage path planning for drone formations and a mother ship carrying a large number of drones. A comparison was made between the static model and the dynamic model, with the dynamic model achieving 85% of the performance of the static model while utilizing fewer vessels. The static model offers the advantage of requiring the same number of mother ships as the number of docking points on the vessels, making it a potentially better choice for scenarios where the search area is smaller and the search team has an adequate number of mother ships. On the other hand, the dynamic model provides greater flexibility and improved interaction patterns, particularly in scenarios involving multiple drones. Furthermore, the impact of hyperparameters, namely K, D, N, and T, on the efficiency of the model's solution was investigated. Through experimental analysis, a balance between solution efficiency and quality was achieved by setting K to any value between 1 and 4, D to 5493.36, and observing the stabilization of solution quality after 200 s.

The results obtained using the precise solver CPLEX were compared to those obtained using the heuristic algorithms NSGA II and SA. It was found that the CPLEX solver yielded superior solutions for our designed model. Although our model successfully achieved the required drone coverage path planning for the given task scenario and exhibited relatively fast solution speed for a small number of waypoints, the solution time exponentially increased as the number of waypoints grew. Consequently, the solver's speed is not well-suited for solving problems involving a large number of waypoints. Therefore, further optimization of the heuristic algorithms is necessary to continually improve the model's solution speed.

Moreover, reinforcement learning and neural networks hold significant potential and offer ample room for development in addressing mid-term strategic planning problems. Prior research has demonstrated promising outcomes by utilizing reinforcement learning and neural network algorithms to tackle mid-term strategic planning problems (Hu et al., 2020). In the context of models such as drone coverage path planning, which resemble mid-term strategic planning problems, reinforcement learning and neural networks can provide more efficient, flexible, and intelligent solutions (Liu et al., 2023; Wang and Tang, 2021). This renders them well-suited for handling large-scale problems.

CRediT authorship contribution statement

Xiaopan Zhang: Conceptualizing and designing the study, Manuscript revision, Analysis of experimental data. **Furong Zhang:** Drafting the paper, Designing and conducting experiments, Analyzing data. **Zheng Tang:** Literature review, Writing of the paper. **Xingjun Chen:** Guidance, Supervision, Research design, Data interpretation, Manuscript preparation.

Declaration of competing interest

We confirm that there is no financial support that could compromise its objectivity, nor are there any known conflicts of interest related to this publication. We have ensured that the intellectual property rights associated with this work have been adequately protected and there are no obstacles to publishing it, including timing, with regard to intellectual property. We have complied with our institutions' regulations concerning intellectual property to confirm this. This research is supported by Natural Science Foundation of China. We further confirm that any aspect of the related work addressed in this manuscript was carried out with the approval of all relevant institutions and that such approval is acknowledged in the manuscript. We confirm that the manuscript has been read

and approved by all named authors. We verify that all listed authors have agreed to the order in which they are named in the manuscript.

Data availability

Data will be made available on request.

Acknowledgment

This research is supported by Natural Science Foundation of China (No. 71701208, No. 52079101 and No. 60904073).

Appendix A

See Table A.1.

Appendix B

Table A.1
The coordinates of the target region and waypoints.

Type	Key point	x (m)	y (m)
Target region corners	a_1	5000	5000
	a_2	26000	5000
	a_3	26000	22000
	a_4	5000	22000
Ship waypoints	p_0	8000	15000
	p_1	8000	8000
	p_2	16000	15000
	p_3	16000	8000
	p_4	24000	15000
	p_5	24000	8000
UAV waypoints	p_6	7746.68	20 414.20
	p_7	5000.00	18 828.41
	p_8	5000.00	15 656.81
	p_9	7746.68	14 071.01
	p_{10}	13 240.05	20 414.20
	p_{11}	10 493.36	18 828.41
	p_{12}	10 493.36	5656.81
	p_{13}	13 240.05	14 071.01
	p_{14}	18 733.41	20 414.20
	p_{15}	15 986.73	18 828.41
	p_{16}	15 986.73	15 656.81
	p_{17}	18 733.41	14 071.01
	p_{18}	24 226.78	20 414.20
	p_{19}	21 480.09	18 828.41
	p_{20}	21 480.09	15 656.81
	p_{21}	24 226.77	14 071.01
	p_{22}	26 973.45	18 828.41
	p_{23}	26 973.45	15 656.81
	p_{24}	7746.68	10 899.42
	p_{25}	5000.00	9313.62
	p_{26}	5000.00	6142.03
	p_{27}	7746.68	4556.23
	p_{28}	13 240.05	10 899.41
	p_{29}	10 493.36	9313.62
	p_{30}	10 493.36	6142.03
	p_{31}	13 240.05	4556.23
	p_{32}	18 733.41	10 899.42
	p_{33}	15 986.73	9313.62
	p_{34}	15 986.73	6142.03
	p_{35}	18 733.41	4556.23
	p_{36}	24 226.77	10 899.42
	p_{37}	21 480.09	9313.62
	p_{38}	21 480.09	6142.03
	p_{39}	24 226.77	4556.23
	p_{40}	26 973.45	9313.62
	p_{41}	26 973.45	6142.03

Algorithm 1 SA

Input:
Initial temperature:T
Cooling rate:alpha
Maximum iterations:num_iterations
Initial solution:S
number of UAV :num_UAV
number of flight segments:num_semage
waypoint set based on triangulation :Point
Output:
Best solution:S_best
Initialize the best solution S_best as the current solution.
while T > T_end:
for i in range(num_iterations):
Generate a neighboring solution S_new by perturbing the current solution S.
Calculate the target value of S_new as S_new_obj
Calculate the target value of S as S_obj
if S_new_obj < S_obj:
S = S_new
else :
generated number p between 0 and 1
if p < exp(-(S_new_obj - S_obj) / T):
S = S_new
Calculate the target value of S_best as S_best_obj
if S_ob<S_best_obj :
S_best = S
T = T * alpha
End

Appendix C

Algorithm 2 NSGA_II

Input:
Population size :N
Maximum number of generations:max_generations
Crossover probability:Pc
Initial solution:S
Mutation probability: Pm
number of UAV :num_UAV
number of flight segments:num_semage
waypoint set based on triangulation :Point
Output:
Pareto front of solutions
Initialize the population P with N randomly generated individuals.
Evaluate the fitness of each individual in the population.
Set the current generation count gen = 1.
While gen <= max_generations:
Create an empty offspring population Q
Repeat the following steps until Q is filled:
a. Select two parent individuals from P using binary tournament selection.
b. Perform crossover operation on the selected parents to produce two offspring individuals.
c. Apply mutation operation to the offspring individuals with probability Pm.
d. Evaluate the fitness of the offspring individuals.
e.Add the offspring individuals to Q.
Merge the population P and Q to create a combined population R.
Apply non-dominated sorting to R to assign a rank to each individual.
Perform crowding distance assignment within each rank in R.
Select the top N individuals from R based on their rank and crowding distance to form the next generation population P.
gen = gen+1
End

References

Balampanis, F., Aguiar, A.P., Maza, I., Ollero, A., 2017a. Path tracking for waypoint lists based on a pure pursuit method for fixed wing UAS. In: 2017 Workshop

on Research, Education and Development of Unmanned Aerial Systems. RED-UAS, pp. 55–59.
Balampanis, F., Maza, I., Ollero, A., 2017b. Coastal areas division and coverage with multiple UAVs for remote sensing. Sensors 17 (4), 808.
Balampanis, F., Maza, I., Ollero, A., 2017c. Spiral-like coverage path planning for multiple heterogeneous UAS operating in coastal regions. In: 2017 International Conference on Unmanned Aircraft Systems. ICUAS, pp. 617–624.
Bektas, T., 2006. The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega 34 (3), 209–219.
Bolaños, R., Echeverry, M., Escobar, J., 2015. A multiobjective non-dominated sorting genetic algorithm (NSGA-II) for the multiple traveling salesman problem. Decis. Sci. Lett. 4, 559–568.
Cheng, Y., Li, D., Wong, W.E., Zhao, M., Mo, D., 2022. Multi-UAV collaborative path planning using hierarchical reinforcement learning and simulated annealing. Int. J. Perform. Eng. 18 (7), 463–474.
Cho, S.W., Park, H.J., Lee, H., Shim, D.H., Sun, Y.K., 2021. Coverage path planning for multiple unmanned aerial vehicles in maritime search and rescue operations. Comput. Ind. Eng. (ISSN: 0360-8352) 161, 107612.
de Ocampo, A.L.P., Bandala, A.A., Dadios, E.P., 2018. Coverage path planning on multi-depot, fuel constraint UAV missions for smart farm monitoring. In: 2018 IEEE Region Ten Symposium. Tensymp, pp. 13–18.
Deb, K., Agrawal, S., Pratap, A., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6, 182–197.
Dorling, K., Heinrichs, J., Messier, G.G., Magierowski, S., 2017. Vehicle routing problems for drone delivery. IEEE Trans. Syst. Man Cybern. Syst. 47 (1), 70–85.
Hu, Y., Yao, Y., Lee, W.S., 2020. A reinforcement learning approach for optimizing multiple traveling salesman problems over graphs. Knowl.-Based Syst. 204, 106244.
Javier, M., López, B., Quevedo, F., Monje, C.A., Garrido, S., Moreno, L.E., 2021. Multi UAV coverage path planning in urban environments 21. p. 7365.
Jensen-Nau, K.R., Hermans, T., Leang, K.K., 2021. Near-optimal area-coverage path planning of energy-constrained aerial robots with application in autonomous environmental monitoring. Trans. Autom. Sci. Eng. 18 (3), 1453–1468.
Kirkpatrick, Scott, Gelatt, Jr., C.D., Vecchi, Jr., M.P., Optimization by simulated annealing. Science 220, 671–680, 198.
Klein Koerkamp, N.W., Borst, C., Mulder, M., van Paassen, M.M., 2019. Supporting humans in solving multi-UAV dynamic vehicle routing problems. IFAC-PapersOnLine 52, 359–364.
Laarhoven, P.J.M., Aarts, E.H.L., 1987. Simulated Annealing: Theory and Applications. In: Mathematics and Its Applications, p. 186.
Li, T., Wang, C., Q.-H., M. Max, d. Silva, C.W., 2019. Coverage sampling planner for UAV-enabled environmental exploration and field mapping. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, IEEE Press, pp. 2509–2516.
Li, D., Yin, W., Wong, W.E., Jian, M., Chau, M., 2022. Quality-oriented hybrid path planning based on A* and Q-learning for unmanned aerial vehicle. IEEE Access 10, 7664–7674.
Liu, X., Chai, Z.Y., Li, Y.L., Cheng, Y.Y., Zeng, Y., 2023. Multi-objective deep reinforcement learning for computation offloading in UAV-assisted multi-access edge computing. Inform. Sci. 642, 119154.
Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems. J. Assoc. Comput. Mach. 7, 326–329.
Wang, Q., Tang, C., 2021. Deep reinforcement learning for transportation network combinatorial optimization: A survey. Knowl.-Based Syst. 33, 107526.
Xie, J., Garcia Carrillo, L.R., Jin, L., 2020. Path planning for UAV to cover multiple separated convex polygonal regions. IEEE Access 8, 51770–51785.
Zhan, J., Xie, W., Guo, Q., Zhang, P., 2018. An improved UAV coverage search route planning method. In: 2018 IEEE CSAA Guidance, Navigation and Control Conference. CGNCC, pp. 1–5.
Zhang, C., Fu, W., 2021. Optimal model for patrols of UAVs in power grid under time constraints. Int. J. Perform. Eng. 17 (1), 103–113.

Zhang Xiaopan is an associate professor at the School of Resources and Environmental Engineering, Wuhan University of Technology. He holds a doctoral degree in Systems Engineering from Huazhong University of Science and Technology. His primary research areas include complex systems analysis and integration, operations research, and optimization algorithms, as well as geographic information systems.

Zhang Furong is a master's student at Wuhan University of Technology, under the supervision of Associate Professor Zhang Xiaopan. She obtained her bachelor's degree from Wuhan University of Technology in China in 2021 and is currently pursuing her master's degree at the same institution. Her research focuses on complex system modeling and optimization, geographic information systems.

Tang Zheng is a master's student at Wuhan University of Technology, also under the guidance of Associate Professor Zhang Xiaopan. He earned his bachelor's degree from Wuhan University of Technology in China in 2021 and is currently pursuing his master's degree at the same institution. His research interests lie in complex system modeling and optimization, geographic information systems.

Chen Xingjun is an associate researcher at the PLA Dalian Naval Academy. He has a doctoral degree in Systems Engineering from Huazhong University of Science and Technology. His primary research area is naval battlefield situation modeling.