International Journal of Geo-Information

MDPI

Article

Unmanned Aerial Vehicle Route Planning in the Presence of a Threat Environment Based on a Virtual Globe Platform

Ming Zhang [1], Chen Su [1], Yuan Liu [2], Mingyuan Hu [2] and Yuesheng Zhu [1,*]

[1] Shenzhen Graduate School, Peking University, Shenzhen 518055, China; zhangming@sz.pku.edu.cn (M.Z.); 1501213962@sz.pku.edu.cn (C.S.)

[2] Institute of Space and Earth Information Science, The Chinese University of Hong Kong, Hong Kong; wandoumou2014@outlook.com (Y.L.); humingyuan@gmail.com (M.H.)

* Correspondence: zhuys@pku.edu.cn; Tel.: +86-755-2603-5352

Academic Editor: Wolfgang Kainz

Received: 5 July 2016; Accepted: 28 September 2016; Published: 10 October 2016

Abstract: Route planning is a key technology for an unmanned aerial vehicle (UAV) to fly reliably and safely in the presence of a threat environment. Existing route planning methods are mainly based on the simulation scene, whereas approaches based on the virtual globe platform have rarely been reported. In this paper, a new planning space for the virtual globe and the planner is proposed and a common threat model is constructed for threats including a no-fly zone, hazardous weather, radar coverage area, missile killing zone and dynamic threats. Additionally, an improved ant colony optimization (ACO) algorithm is developed to enhance route planning efficiency and terrain masking ability. Our route planning methods are optimized on the virtual globe platform for practicability. A route planning system and six types of planners were developed and implemented on the virtual globe platform. Finally, our evaluation results demonstrate that our optimum planner has better performance in terms of fuel consumption, terrain masking, and risk avoidance. Experiments also demonstrate that the method and system described in this paper can be used to perform global route planning and mission operations.

Keywords: unmanned aerial vehicle; route planning; virtual globe; ant colony optimization; 3D environments

1. Introduction

A UAV is an aircraft without a pilot on board that can be remotely controlled or flown automatically based on a pre-planned route or automation system [1]. With the development of the aviation electronics industry, UAVs play increasingly important roles in military and civil fields [2,3].

Generally, route planning for a UAV is an optimization problem that aims to generate a feasible route based on the tasks. The problem is an NP-hard problem [3].

For different types of tasks, scholars have selected different route planning methods. The vector field method is used in static or dynamic target tracking [4,5]. The rapidly-exploring random-tree (RRT) method has been applied to the path planning problem of indoor robots and mini UAVs [6–8]. Genetic algorithms (GA) are used to solve the travelling salesman problems related to UAVs, such as maximum information collection [9]. The evolutionary algorithm (EA) is used for multi-constraint route planning in a simulation scenario [10–12]. The particle swarm optimizer (PSO) is used to solve the path planning problem of UAVs on the sea [13]. Improved ACO [14,15], A* and Theta* [16] algorithms are used for route planning in three-dimensional environments. To improve the investigation efficiency of unmanned bombs, scholars have proposed the Quantum Wind Driven [17] algorithm.

Scholars have proposed a variety of optimization and improvement methods for the route planning problem under different conditions and have solved these problems relatively well. For different scenarios, each algorithm also has its own limitations: the ability of RRT to avoid obstacles is unsatisfactory; the processing time of the A* algorithm will increase explosively as the planning scene enlarges; and the computational complexity of GA and EA algorithms is high [18]. Scholars tend to optimize the selected algorithm according to their own simulation scenarios but the results of the experiments seem to not be very objective. For example, the results obtained by PSO were much better than GA in [13], whereas the results of PSO were inferior to GA in [18].

The virtual globe platform has the great advantages of low cost and ease-of-use in data collection, browsing, visualization and other aspects [19]. For the path planning problem for a high-endurance UAV, the virtual globe platform is a good choice to realize the modelling and visualization of a very large environment. There are many common virtual globe platforms such as Skyline, Google Earth, Virtual Earth, World Wind, and ArcGlobe.

Moreover, UAVs work in dangerous enemy territory in military penetration tasks. Avoiding threats from an enemy is a key factor in the success of tasks. Route planning for penetration finds a feasible route between the start point and end point in the presence of a threat environment. The defence of medium and high altitude areas in air defence systems is improving because of the development of the Radar Netting Technique [20]. There is no opportunity to penetrate without stealth aircraft. However, there are many radar blind zones in low-altitude areas because of topography and the curvature of the earth, and low-level flight becomes an important way to penetrate. On the virtual globe platform, threat modelling and route planning are more real and effective and mission operations can be performed [21]. Using the interactive capabilities of the virtual globe platform, operations such as parameter adjustment, route editing and storage, and flight simulation can be realized and successfully applied to industrial fields.

On the virtual globe platform, the method of model construction, planning space partitions and the realization of the algorithm will be different from the previous studies in the following ways:

- When modelling the radar threat areas, the maximum coverage range of early-warning radar can be up to hundreds, even thousands of kilometres, with larger signal coverage in high-altitude areas than at low-altitudes. By using the virtual globe platform, we can construct a more reasonable radar threat model according to the radar equation and fully consider the influences of earth curvature and terrain masking.
- The scale of the terrain data is large on the virtual globe platform. In view of this problem, this paper proposes a multi-granularity planning space to achieve a balance between accuracy and efficiency.
- In low-altitude penetration, a good valley-following ability can effectively avoid a radar threat, including unknown radar threats. We propose a strengthened local valley-following algorithm for route planning.
- The planning space needs to be transformed between Cartesian systems and Geodetic systems. Because of the overhead problem generated by large-scale data and space transformation, we optimize some of the algorithm's implementation details.
- Because there are certain errors in a UAV's navigation system and control system, this paper refers to industry standards, such as the performance based navigation (PBN) standard [22], to optimize and improve the robustness of the route to avoid collisions because of flight errors.

Section 2 describes the route planning problem and evaluation indexes in the risk environment. In Section 3 we propose a multi-granularity planning space on the virtual globe platform for route planning and consider multiple types of threats. In Section 4, we propose an improved ACO algorithm with valley-following and threat avoidance for route planning and some route optimization algorithms to make the route more effective and robust. Section 5 shows the practicability of the proposed methods through experiments. Section 6 presents conclusions and proposes future research work.

2. Route Planning Problem Descriptions

The location of a point P in geodetic space Ψ^3 is described by longitude x, latitude y, and altitude H. A UAV flies along the designated route point sequence $R_{route} = \{r_1, r_2, \cdots, r_n\}$, $r_i \in \Psi^3$, where r_1 is the start point and r_n is the end point. The goal of penetration routing is to obtain a feasible route with minimal fuel consumption, maximum terrain masking, and minimum risk.

Both the navigation systems and the control systems of UAVs contain certain deviations. The international civil aviation organization proposed the concept of the required navigation performance (RNP). RNP [22] describes the precision that can be attained during at least 95% of the flight time; the precision unit is the nautical mile (nmi). In Figure 1, the segment width of the route is defined as 4 × RNP. The planned route may not satisfy the performance constraints of UAVs.

Figure 1. Plan view for RNP segment width.

We evaluate the planning route on four dimensions: minimal fuel consumption, maximum terrain masking, minimum risk and performance safety. Additionally, we evaluate the efficiency of the planner:

- L_{route}: Route length. Connect the points in R_{route} successively and obtain a series of line segments, then calculate the total length of the segments to obtain the route length. To evaluate the result, we introduce the theoretically shortest length of the route, L_{min}. Then, we connect the r_1 and r_n of R_{route} to obtain the straight line segment r_1r_n. L_{min} is the length of r_1r_n.
- h_{avT}: Average terrain altitude passed by the route. h_{avT} is calculated as follows: Obtain the interpolation points $\{r_1(x_1, y_1, H_1), r_2(x_2, y_2, H_2), \cdots, r_k(x_k, y_k, H_k)\}$ with interval $\forall$ of R_{route}. We can then obtain points on the terrain $\{r_1(x_1, y_1, H_{1T}), r_2(x_2, y_2, H_{2T}), \cdots, r_k(x_k, y_k, H_{kT})\}$, and $h_{avT} = \sum_{i=1}^{k} H_{iT}/k$. To evaluate the result, we introduce a reference H_{avT}, the average terrain altitude passed by r_1r_n. We can understand the terrain masking ability of the planner by contrasting the result with H_{avT}.
- L_{FA}: Route length that passes through the threat zones.
- L_{US}: Route length that does not meet the flight performance safety requirements.
- Processing time.

3. Build the Planning Space

3.1. Virtual Globe Space

For the penetration tasks on the virtual globe, the planning space needs to be converted between Geodetic space Ψ^3 and Cartesian space υ^3. Ψ^3 treats the earth as a reference ellipsoid. $P(x, y, H) \in \Psi^3$ can be converted into $Q(X, Y, Z) \in \upsilon^3$ as follows:

$$\begin{cases} X = (M + H)\cos(y)\cos(x) \\ Y = (M + H)\cos(y)\sin(x) \\ Z = \left[M\left(1 - \frac{a^2-b^2}{a^2}\right) + H\right]\sin(y) \end{cases} \quad (1)$$

where M represents the radius of curvature of the reference ellipsoid, computed as follows:

$$M = \frac{a}{\sqrt{1 - \frac{a^2 - b^2}{a^2}\sin^2(y)}} \quad (2)$$

where a and b represent the length of the major axis and the minor axis of the reference ellipsoid, respectively.

$Q(X, Y, Z) \in \upsilon^3$ can be converted into $P(x, y, H) \in \Psi^3$ as follows:

$$\begin{cases} x = \arctan(\frac{Y}{X}) \\ y = \arctan(\frac{Z(M+H)}{\sqrt{(X^2+Y^2)\left[M\left(1-\frac{a^2-b^2}{a^2}\right)+H\right]}}) \\ H = \frac{\sqrt{X^2+Y^2}}{\cos(y)} - M \end{cases} \tag{3}$$

The reference ellipsoid used by the Global Positioning System and virtual globe platforms is the World Geodetic System of 1984 ellipsoid (WGS-84 ellipsoid) [22].

The lengths of the same longitude interval in different latitudes and the same latitude interval in different longitudes are not fixed in Ψ^3. For example, the distance between (90°, 29°, 4000 m) and (91°, 29°, 4000 m) is 97,422.04 m, whereas the distance between (90°, 26°, 4000 m) and (91°, 26°, 4000 m) is 100,114.77 m. In the later sections, the planning space is in Ψ^3, but the distance measure, interpolation algorithm, equation calculation and other operations are carried out in υ^3.

The terrain is usually described by digital elevation model (DEM) data, whereas terrain texture is described by digital orthophoto map (DOM) data. The data of DEM and DOM can be represented as $\{(x, y, z)_i\}$, where x is longitude and y is latitude. In DEM data, z is the altitude value of the corresponding position, whereas z is the pixel value for DOM data, as Figure 2a,b illustrate.

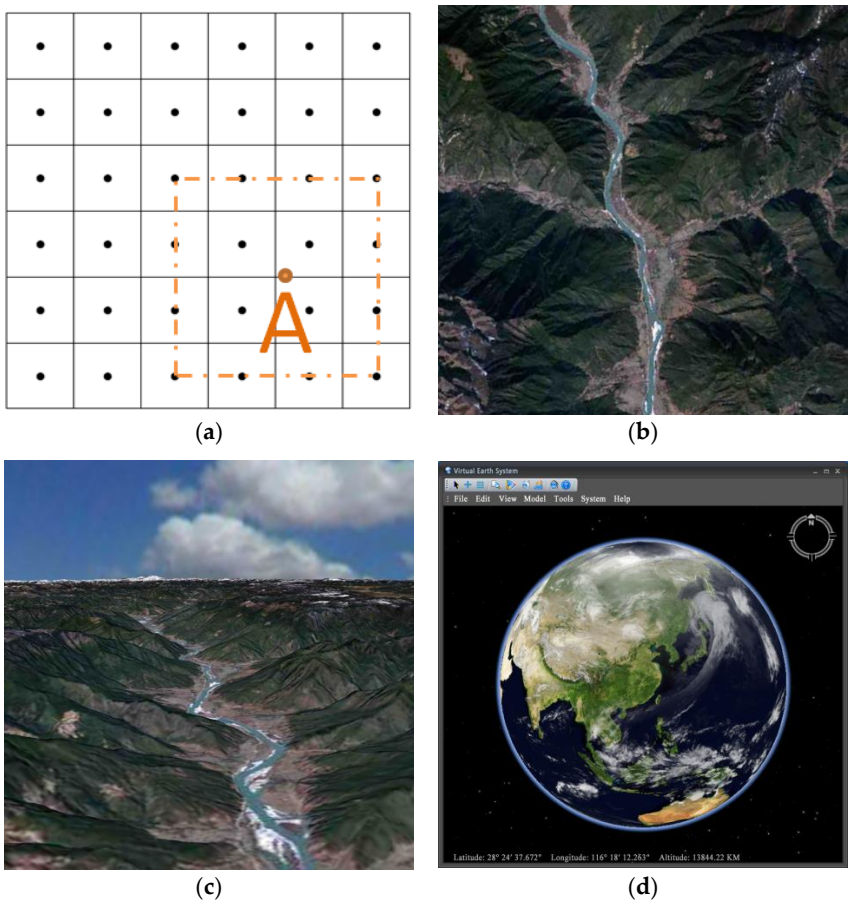


Figure 2. (**a**) Description of regular grid DEM; (**b**) DOM data; (**c**) DEM and DOM data visualization; (**d**) Virtual globe platform.

DEM and DOM describe discrete space, whereas the real world belongs to contiguous space. The information of any position can be obtained by using a spatial interpolation method through the adjacent points, and the bicubic interpolation algorithm is described as follows:

$$Z_p = f(x_p, y_p) = \sum_{i=0}^{3} \sum_{j=0}^{3-i} a_{ij} x_p^i y_p^j \quad (4)$$

We can use the adjacent 4 × 4 points of point A in Figure 2a to calculate the coefficients of the interpolation function to determine the altitude value of point A.

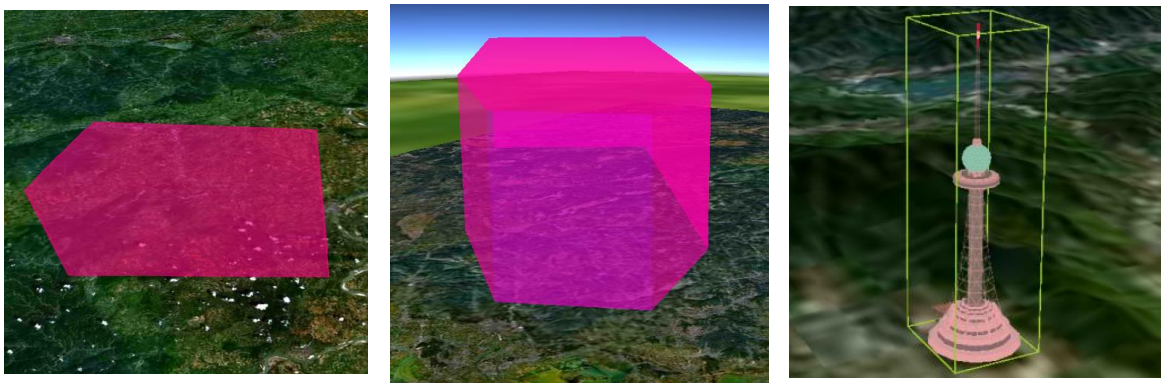
In Figure 2c, we can realize the visualization of 3D terrain by superimposing the DOM data on the DEM data. In this study, we used the improved NASA-WorldWind platform; the data source was the SRTM DEM data at 90-m resolution and the Landsat DOM data at 60-m resolution provided publicly by NASA, as shown in Figure 2d.

3.2. Threat Modelling

3.2.1. General Types of Threats

No-fly zones, hazardous weather, high-rise buildings and low-altitude control zones need to be considered when planning routes.

No-fly zones are regions that a UAV cannot fly into. A no-fly zone can be described as the enclosed area defined by the point set $A = \{A_1, A_2, \cdots, A_m\}$, $m > 2$, as shown in Figure 3a.



(**a**) (**b**) (**c**)

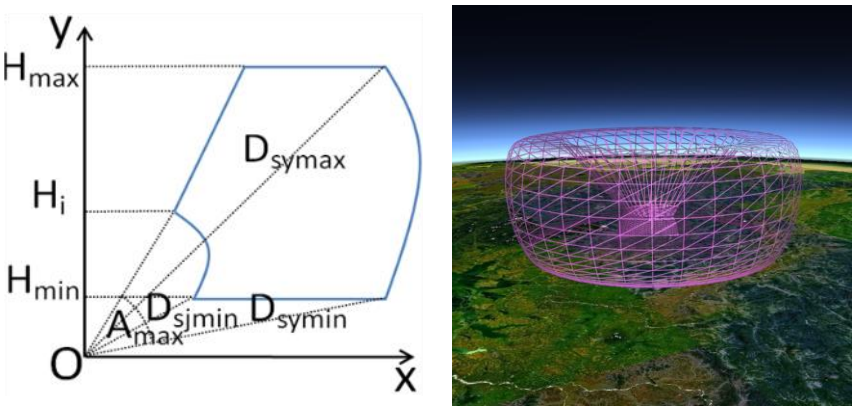
Figure 3. (**a**) 2D No-fly zone; (**b**) 3D slowly hazardous weather model; (**c**) 3D threat model of a 500-m high TV tower described as {(114.655225, 23.269952), (114.658225, 23.269952), (114.658225, 23.272952), (114.655225, 23.272952), 500}.

High buildings, slowly hazardous weather and low-altitude control zones are low-altitude regions that UAVs cannot fly into. Such threats can be described as $\{\{A_1, A_2, A_3, \cdots, A_m\}, H_{max}\}$, $m > 2$, where H_{max} is the upper bound. This type of threat is modelled as observed in Figure 3b,c. These threats can be dealt with as terrain data; we transform this type of threat area into terrain before route planning.

3.2.2. Killing Zone of an Air Defense Missile

The killing zone [23,24] is an important guideline for judging the campaign performance of Air Defense Missile Weapon Systems. A killing zone describes an area of space in which a missile can destroy a target at no less than a certain probability once the target enters the area. The mathematical model of a vertical killing zone is shown in Figure 4a. Horizontal lines define the high and low boundaries with heights of H_{max} and H_{min}, respectively. D_{symin} is the minimum slant range of the far

boundary of the killing zone and D_{symax} is the maximum slant range. The minimum slant range and the maximum height angle of the near boundary of the killing zone are D_{sjmin} and A_{max}, respectively. The near boundary and far boundary are the arcs with O as the centre. Figure 4b shows the 3D visualization model of the missile killing zone.



(**a**) (**b**)

Figure 4. (**a**) Model of a missile killing zone; (**b**) 3D visualization of a missile killing zone.

3.2.3. Radar Threat Space

Radar plays a very important role in modern air defence systems; it is a key threat to the penetration of UAVs that needs to be avoided by route planning. Affected by the terrain masking and the earth curvature, it is difficult for radar to find a target in low-altitude flight.

The radar threat model should refer to the radar detection probability threshold that a UAV can maximally tolerate. Given the detection probability threshold and radar performance parameters, the maximum range of radar R_{max} can be estimated by the radar equation [25]. Assuming that the radar is deployed at $R_a(x_a, y_a, H_a)$ in Ψ^3, we transform it into υ^3 as $R_a(X_a, Y_a, Z_a)$, with the direction of pitch angle θ and azimuth angle β, and we can describe $p_k(X_k, Y_k, Z_k)$ on the boundary of the radar coverage as:

$$\begin{cases} X_k = X_a = R_{max} f'(\theta) \cos(\beta) \\ Y_k = Y_a = R_{max} f'(\theta) \sin(\beta) \\ Z_k = Z_a = R_{max} f'(\theta) \end{cases} \quad (5)$$

where $f'(\theta)$ describes the beam pattern of the radar antenna in the direction of the pitch angle.

We use the Gaussian function to approximate the beam pattern as:

$$f'(\theta) = e^{-4\ln\sqrt{2}\theta^2/\theta_b} \quad (6)$$

where θ_b is the signal beam width.

The radar threat curved surface can thus be obtained. The probability of radar detection on this surface is constant. Outside the curved surface, the radar detection probability is less than the probability threshold and the UAV is regarded as safe. Within the curved surface, the radar detection probability is higher than the probability threshold and the UAV is not safe.

In low-altitude areas, radar signals may be masked by terrain. In Figure 5a, radar beams are partly masked by terrain and the shaded area represents the area that signals can reach. The visualization of the radar beam can be constructed as follows:

1. Calculate the detection range of the radar with different sampled pitch angles by using Equations (5) and (6); this produces a series of sampling points such as A–G, as shown in Figure 5a.

2. Connect the radar centre O to the sampling point and obtain a series of sampling points on this straight line by sampling with short intervals. Then, we transform these sampling points to Ψ^3 and compare the altitudes of the sampling points with the terrain. Once the sampling point is below the terrain, we can determine that the subsequent region cannot be reached by the radar signal. In Figure 5a, according to the terrain visibility analysis of OD, the boundary point D is moved to M.

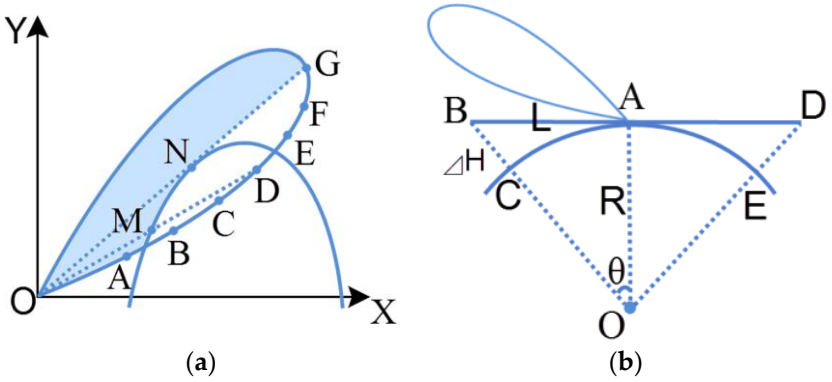


Figure 5. (**a**) Radar signal propagation; (**b**) Earth curvature's influence.

If we calculate the radar coverage directly, the earth curvature will introduce considerable error. As shown in Figure 5b, O is the earth centre, BD is the horizontal plane, the circular arc CE is the geoid, and point B is in the same horizontal plane as point A. When B is very near to A, the altitudes of A and B are approximately the same and may be ignored. However, when B is far from A, the elevation difference between A and B due to the influence of the earth curvature, ΔH in Figure 5b, cannot be ignored. L represents the length of AB, R is the average curvature radius of the earth, and ΔH can be calculated by:

$$\Delta H = \sqrt{R^2 + L^2} - R \quad (7)$$

When the distance between two points is 100 km, the elevation difference is approximately 785 m because of the influence of the curvature. First, we need to convert from Ψ^3 to υ^3 when calculating the boundary of the radar coverage. Then, we can calculate the radar coverage and perform visibility analysis. Finally, the results are converted back to Ψ^3 and the errors caused by the curvature of the earth have been eliminated.

Figure 6 shows the 3D visualization of the radar network coverage under the influence of the earth's curvature.

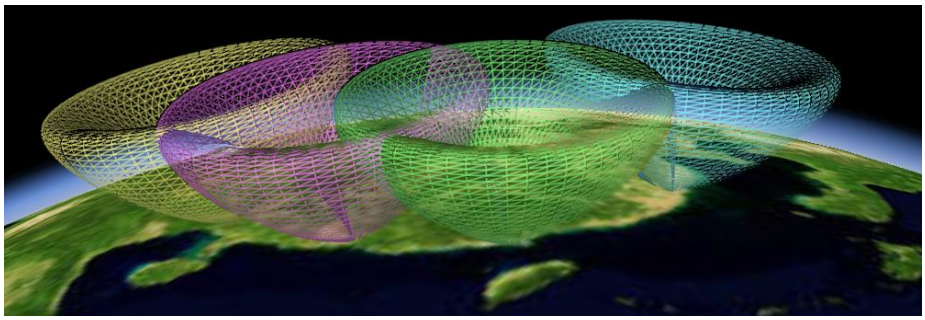


Figure 6. Visualization of the radar networking coverage on the virtual globe.

3.3. Dynamic Threat

The models presented in Section 3.2 can simulate most of the threats in normal circumstances. Sometimes the threat movement needs to be considered, for example, the forecast of hazardous weather such as a typhoon (hurricane) and other predictable dynamic threat areas. At this time, we need to introduce the dynamic threat model. Based on the general threat model, a time stamp t is introduced, the property of a dynamic threat at t is described by $\text{threat}_t = \{\{A_1, A_2, A_3, \cdots, A_m\}, H_{max}, t\}$, $m > 2$. A full path dynamic threat can be described by $\{\text{threat}_{t_1}, \text{threat}_{t_2}, \cdots, \text{threat}_{t_n}\}$, where the threat begins at t_1 and disappears at t_n. As shown in Figure 7a,b, if we want to know the state of a threat at any time t_k, we can calculate it as follows:

1. Search the interval $[t_i, t_{i+1}]$ that satisfies $t_i \leq t_k < t_{i+1}$ to find the threat_{t_i} and $\text{threat}_{t_{i+1}}$. If not found, then there is no threat area at this time.
2. Given the threat $t_k = 2.4$ shown in Figure 7b, use linear interpolation for threat_{t_i} and $\text{threat}_{t_{i+1}}$ to obtain threat_{t_k}. As shown in Figure 7b, we can use linear interpolation for the four vertices of threat_{t_i} and $\text{threat}_{t_{i+1}}$ to obtain threat_{t_k}.

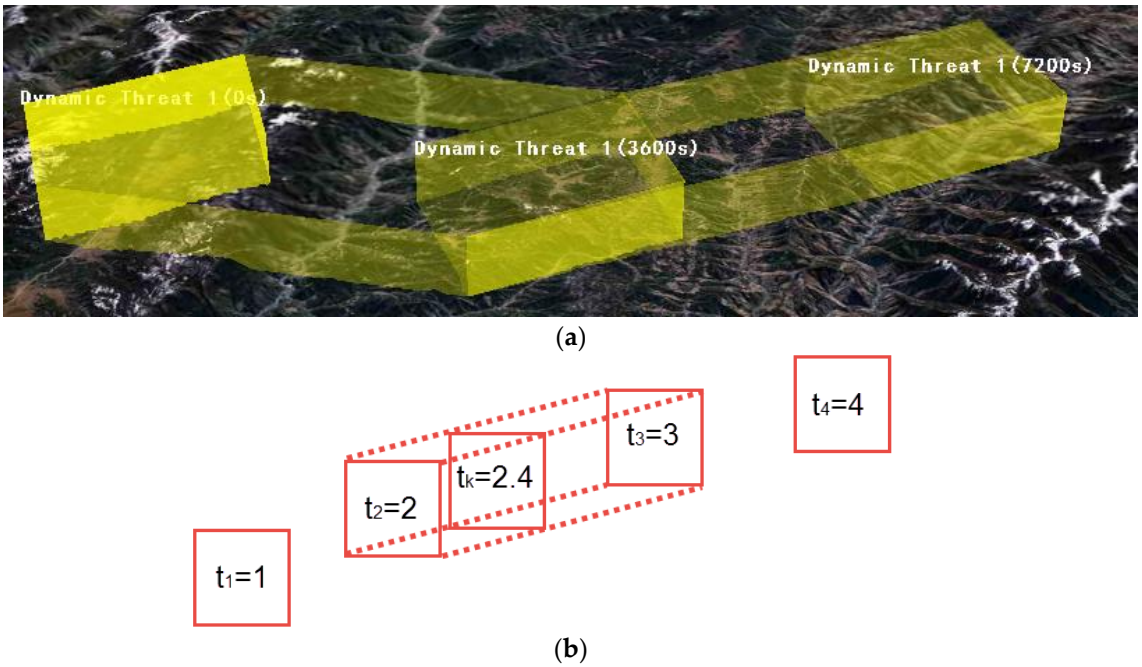


Figure 7. Dynamic threat model. (**a**) There is a dynamic threat described as {{(99.94497, 27.60066), (99.859, 27.38197), (99.68604, 27.47941), (99.77718, 27.68262), 3000, 0 s}, {(99.54497, 27.40066), (99.459, 27.18197), (99.28604, 27.27941), (99.37718, 27.48262), (99.54497, 27.40066), 3000, 3600 s}, {(99.34497, 27.00066), (99.259, 26.78197), (99.08604, 26.87941), (99.17718, 27.08262), (99.34497, 27.00066), 3000, 7200 s}}; (**b**) Perform linear interpolation for the dynamic threat to obtain the threat status at any time.

3.4. Build the Grid Planning Space

Route planning on the virtual globe platform in this study is based on a graph search algorithm. We define a planning graph composed of nodes and edges. The graph needs to effectively express threats and terrain features.

In low-altitude penetration missions, threat models need to be marked. However, real-time sampling of a threat body at any point in the planning space will dramatically increase the delay time of the planner. This paper introduces a type of grid space with location information, elevation information, and other attribute information.

In Figure 8, the space is divided into a two-dimensional grid by equal intervals of latitude and longitude; each node in the grid has eight extension nodes. However, the actual distance represented by the equal intervals of latitude and longitude is different. In the virtual globe platform, the planning space is actually an irregular eight-connected grid space.

A UAV flies at the safe height h by default in this space. Each node stores the information including [lon, lat, alt, H_{min}], where lon and lat represent the longitude and latitude of the node, respectively, alt represents the terrain altitude superposed by the high buildings and low-altitude control zones, and H_{min} is the lowest bound of the radar coverage area or the missile killing zone in this area. If the node h that needs to be extended is higher than H_{min}, the node will be eliminated. As the figure shows, nodes A, B, C and D are in a radar coverage area. However, because of the terrain masking effect, points B and C can be used as planning candidate nodes with higher H_{min}, whereas nodes A and D with low H_{min} will be directly eliminated. As the figure shows, node E, F, G and H are in a missile threat area and will be eliminated because of their low H_{min}. If the current point is located in the no-fly zone, H_{min} is set to 0 and the point will be eliminated by the planner. Because of the planning space generated beforehand, the planner can quickly eliminate some infeasible nodes.

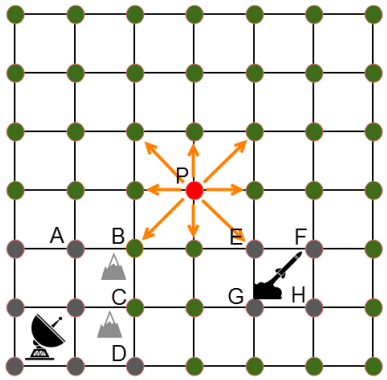


Figure 8. Space division.

3.4.1. Threat Area Expansion

Considering the navigation error and the route optimization error, as shown in Figure 9a, the planned route in grid space may intersect the threatened area. We expand the boundary of the threat models when mapped to the planning space. The low boundary of the radar area and the missile killing zone are also reduced. Figure 9b shows that the route is safe after expanding the threat area.

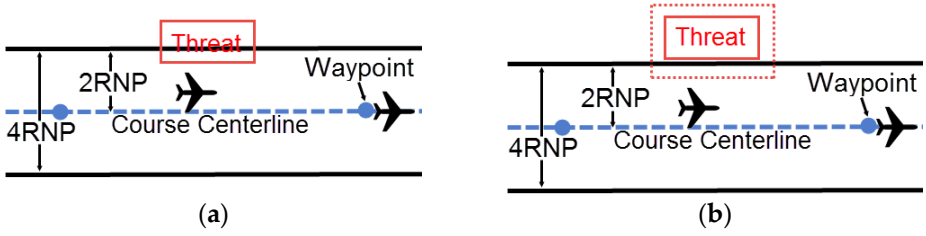


Figure 9. (**a**) Route planning with threat; (**b**) Route planning with threat area expansion.

3.4.2. Multi-Granularity Planning Space

The finer granularity of the grid space is because the more nodes in planning space, the more precise the planning results will be, although the planning will take more time. To obtain a balance between performance and precision for different application scenarios and navigation performance parameters of a UAV, different grid sizes need to be used.

We need resampling to construct different sizes of the grid using DEM data. Common resampling methods are Box Splines [26], interpolation [27] and the discrete wavelet transform [28]. This paper uses the bicubic interpolation method of Equation (4) to resample. We generate grid space with multi-granularity beforehand and choose different grid sizes according to the requirements of different tasks.

4. Route Planning Method

We propose an improved ACO-based planner that considers the local terrain environment to obtain the initial route. An ACO algorithm simulates the foraging process of ants in principle, and its developer used it to efficiently solve the Travelling Salesman Problem [29,30]. The pheromone model and the probability model are the core of this algorithm [31]. ACO algorithms have been widely used for pattern classification [32], cloud computing [33], network coding [34], robot path planning [35] and in other fields. We propose some optimizations to improve the ACO algorithm for the study of the virtual globe platform and penetration route planning.

4.1. Basic ACO Algorithm [29,30]

The ant colony begins searching for food without being told where the food is. In the process of moving, ants release pheromones into the environment. When there is no pheromone in the environment, the ants perform a random walk, and when the pheromone concentration is high, the ants will move along the pheromone path with a greater probability. If the ant finds food, it will move along its pheromone path to return to its nest. Along the shorter path, the ants make the roundtrip more quickly, which is more likely to attract more ants to walk along this path. The pheromone concentration will thus be further enhanced and eventually the ant colony will walk along the shortest path. The basic ACO algorithm simulates the pathfinding and feedback processes of the ant colony.

4.1.1. Roulette Wheel Selection Model

The ACO algorithm uses a roulette wheel selection model to simulate the walk choice of ants in nature. This model has a larger probability to choose adjacent nodes with higher pheromone concentration; it allows ants to make mistakes that have a small probability and retain the chance to find a better path. In the model, the ant colony has m ants. The transfer probability $p_{ij}^{k}(t)$ of an ant k $(0 \leq k < m)$ moving from node i to an adjacent node j in the t round of iteration is expressed as:

$$p_{ij}^{k}(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{s \in J_k(i)} \tau_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)}, & \text{if } j \epsilon J_k(i) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $\tau_{ij}(t)$ is the residual pheromone of the edge $< i, j >$. $\eta_{ij}(t)$ is the heuristic function from node i to node j; when solving the TSP, $\eta_{ij}(t) = 1/c_{ij}$. c_{ij} reflects the cost of the walk from node i to node j. α and β are the weights of the pheromone and the heuristic function, respectively. Adjacent nodes $J_k(i)$ can be chosen by the ant k in node i. According to the planning space defined above, each node has eight expanding nodes.

4.1.2. Pheromone Updating Model

The pheromone updating model provides a feedback mechanism in the ACO algorithm. After all the ants have completed an iteration round, pheromones will be left on the path traversed by the ants and the pheromone concentration is updated as follows:

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \tag{9}$$

where ρ is the pheromone retention coefficient $(0 \leq \rho < 1)$. $\Delta\tau_{ij}^{k}(t)$ represents the pheromones left at the edge of $< i, j >$ by the ant k at the iteration of round t and is calculated as follows:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{1}{L^{k}}, \text{ if } < i, j > \in Q_{k} \\ 0, \text{ otherwise} \end{cases} \tag{10}$$

where Q_k is the set of edges traversed by ant k and L^k is the total cost of the edges.

4.2. Parameter Optimization

In the basic ACO algorithm, the ants do not know the target location. Our application scenario knows the location of the target and we can thus introduce a heuristic function to guide the ants to accelerate the iterative process of the algorithm. In the pheromone feedback mechanism of the basic ACO algorithm, the pheromone left by the less successful ants may interfere with a better result, which easily causes the algorithm to fall into a local optimum. To solve the UAV problem described in this paper, we improved the heuristic cost function, pheromone update mechanisms, algorithm efficiency and other aspects of the algorithm. Additionally, to avoid the ants returning, we set up tabu lists to mark the nodes that the ants have traversed; the expanding nodes located in the threat area or in the tabu lists are excluded.

4.2.1. Cost Function

In view of the planning space built in Chapter 3, we introduce the cost function:

$$\begin{cases} W = \sum_{i=1}^{n} w_{(i-1)i} \\ w_{ij} = \frac{l_{ijtraj}}{l_{max}} + \varepsilon \frac{h_{ijtraj} - h_{min}}{h_{max} - h_{min}} \end{cases} \tag{11}$$

where w_{ij} is the cost of the edge $< i, j >$, l_{ijtraj} is the Euclidean length of the edge $< i, j >$, which approximately reflects the fuel consumption cost of the UAV, and h_{ijtraj} is the average terrain elevation of node i and j, which is $(alt_i + alt_j)/2$ and reflects the altitude of the route. Because the units of the fuel consumption cost and height cost are not of the same order of magnitude, they are normalized. l_{max} is the largest Euclidean distance of the two adjacent nodes in the grid. Because planning space is irregular, we set l_{max} = grid width × 160 KM when the grid width is determined. h_{max} and h_{min} are the highest and lowest elevation of the planning space, respectively. ε is the weight, which determines the valley-following performance of the planner and enables the UAV to fly as low as possible. Flying at low-altitude can improve the survival rate of the UAV if the deployment information of the enemy radar is unknown.

4.2.2. Heuristic Function and Valley-Following

In the basic ACO algorithm, the ants have difficulty reaching the target successfully because the c_{ij} can only reflect the cost of node i to j, and thus the direction to the end node is unknown. In this study, a heuristic cost function is introduced that can promote the ant to move towards the target:

$$\eta_{ij}(t) = 2 + \frac{T_i - T_j}{l_{max}} \tag{12}$$

where T_j is the distance between node j and the target point and T_i is the distance between node i and the end point. If $T_j > T_i$, the ants are moving farther and farther away from the target and will be assigned a smaller probability to select this node.

Equation (12) can promote the ants to walk to the destination node; however, the ants cannot use the local valley terrain information to achieve valley-following. When we analysed the terrain data, we found that the valley region was continuous, and the enhanced local valley-following algorithm is

proposed accordingly. The eight adjacent nodes of the current point i are A–H, and the implementation method is as follows:

1. Calculate the average terrain altitude of A–H points.
2. Compare the average altitude with the altitude of point i; if greater than the threshold δ, the UAV at point i is located in the mountainous area.
3. If the UAV is located in the mountainous region, the adjacent A–H points are sorted according to their altitudes, and the 2ε points with highest elevation are set to $\eta_{ij}(t) = 0$.
4. If the UAV is located in the non-mountainous region, all the neighbouring nodes are reachable.

After this treatment, η_{ij} can promote movement of the ants to the target point and make full use of local terrain information.

4.2.3. Pheromone Update Mechanisms

The updating mechanism of the basic ACO algorithm makes it easy for premature results to occur. The pheromone update strategy we use is as follows:

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{best*}(t) \tag{13}$$

where $\Delta\tau_{ij}^{best*}(t)$ is the pheromone increment of global optimal ant:

$$\Delta\tau_{ij}^{best*}(t) = \begin{cases} \frac{1}{W^{best*}}, \text{ if } < i, j > \in Q_{best} \\ 0, \text{ otherwise} \end{cases} \tag{14}$$

where Q_{best} is the set of edges traversed by the global optimal ant. W^{best*} was calculated according to Equation (11) and is the cost of the global optimal route. To ensure that the nodes that the global optimal ant did not walk can be reached with a certain probability and to prevent premature results because of too much pheromone on the global optimal route, we limit the pheromone concentration to $[\tau_{min}, \tau_{max}]$.

$$\begin{cases} \tau_{max} = \frac{1}{1-\rho}\frac{1}{W^{best*}} \\ \tau_{min} = \frac{\tau_{max}}{n} \end{cases} \tag{15}$$

where n is the estimated node number of the global optimal route.

4.2.4. Algorithm Efficiency Improvement

Because of the introduction of the virtual globe platform, every distance calculation needs to transform between two coordinate systems. To reduce the time cost of the repeated distance computation of the ACO, we calculate the distance between the adjacent nodes in advance. This paper introduces a three-dimensional array of off-line storage; the third dimension of the array is used to store the distance from the current node i to the eight adjacent nodes. Additionally, a two-dimensional array is used to store the distance between i and the end node.

In the ACO algorithm, the worst performing ant may run a very biased path with a relatively small probability, thus affecting the overall efficiency of the algorithm. In this paper, the maximum number of search steps is 10 n; when the search steps of ants exceed the maximum number, the result is discarded.

4.3. Dynamic Threat Avoidance

When considering dynamic threats, we need to introduce a dynamic threat avoidance algorithm. To avoid the dynamic threat, the time t of the ant at any node should be known. The ants carry out the following calculation after walking each step to save the current time.

Given a UAV flight speed of v, the route point set that the ant walked is $\{r_1, r_2, \cdots, r_k\}$, $k > 1$; when $k = 1$, the UAV start time is t_0. If we know the time t_{k-1} of the ant at node $k-1$, when the ant walks to route point k we look up the distance of $r_{k-1}r_k$, named $l_{r_{k-1}r_k}$, and the current time $t_k = t_{k-1} + l_{r_{k-1}r_k}/v$. The time of the ant at any node can thus be determined.

The ants excluded the expanding nodes by the following steps:

1. When the ant k at node i selects the eight expanding nodes, the nodes in the static threat and the ant's tabu list are first excluded and we then can obtain the remaining expanding nodes $J_k(i)$.
2. By using the method mentioned above, we calculate the time the ant arrives at every expanding node j, named t_j.
3. We use the method mentioned in Section 3.3 to implement linear interpolation for every dynamic threat in the dynamic threat set and obtain the threat areas at t_j.
4. If the node j is located in any threat area, we exclude this node.

The dynamic threat avoidance algorithm will consume many computing resources; therefore, we compile two route planning versions during use. When there is no dynamic threat in the planning scenario, the planning algorithm is used, named MACO, and when we need to consider the dynamic threat, we introduce the dynamic threat avoidance algorithm and name the planning algorithm MACOD.

4.4. Route Optimization

The route planned by the ACO algorithms above is composed of a series of broken line segments. To determine the feasible route, we need to optimize the route by a series of operations.

4.4.1. Route Compression

To avoid frequent turning of the UAV, it is necessary to compress the waypoints, as shown in Figure 10. When the broken line is compressed, it will be shorter than the original path planned by the grid search algorithm. By compressing the route, errors are introduced; however, the route is optimized to a certain extent.

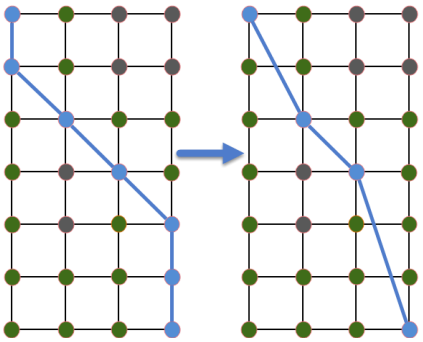


Figure 10. Length changed before and after compression.

When compressing the waypoints, the adjusted route should not cross the threatened area, and the average altitude of the route has not been significantly improved. After deleting waypoint r_i, the route should meet the criteria: (a) the adjusted route segments $\{r_{i-1}, r_{i+1}\}$ should not be located in the threatened area and (b) the difference value of average altitude between $\{r_{i-1}, r_{i+1}\}$ and $\{r_{i-1}, r_i, r_{i+1}\}$ should not exceed the specified threshold £. The average altitude is calculated as follows: interpolate for the route with interval $\forall$ in υ^3, then transform the interval points to Ψ^3 and obtain the terrain altitude for them.

This paper improves the Douglas-Peucker algorithm [36]; the improved algorithm steps are as follows:

1. First, the algorithm compares the distance between each waypoint and the link line of two endpoints. If the distance is less than the tolerance ϵ, judge the route segment to determine if it meets the conditions (a) and (b) after removing this waypoint. If satisfied, then delete this waypoint and if not, then keep this one.
2. Select the point that has the farthest distance with the link line of two endpoints as the separation waypoint such that the route is divided into two segments, and then perform Step 1 recursively for these two segments until there is no waypoint to be deleted.

4.4.2. Smooth Route Turning

After compression, the route still consists of broken lines. The UAV cannot smoothly complete the turning process. There are some common segment fitting methods, such as the Dubins curve [15] and the Bézier curve [2], and some of the authors do not smooth the broken line [12]. In fact, a Fly-by Fix or Fly-over Fix [22] is usually used in PBN flight procedures for turning. In this study, we use the Fly-by Fix for UAV smooth turning. In Figure 11a, DTA is the length of the minimum straight route segment required by the Fly-by Fix; it is determined by the flight speed, turning angle and other parameters and is calculated as follows:

$$\begin{cases} R = \frac{(\min[500, V_{KTAS} + V_{KTW}])^2}{\tan(\theta) \times 68625.4} \\ DTA = R \times \tan\frac{\alpha}{2} \end{cases} \quad (16)$$

where R is the turning radius, V_{KTAS} is the true airspeed, V_{KTW} is the tailwind, and θ is the slope angle.

UAV flight tends to avoid frequent rising or descending. If a UAV needs to fly from the waypoint A to waypoint B with different elevations, the UAV should rise/descend with maximum lifting angle to the horizontal line of B and then maintain level flight to B.

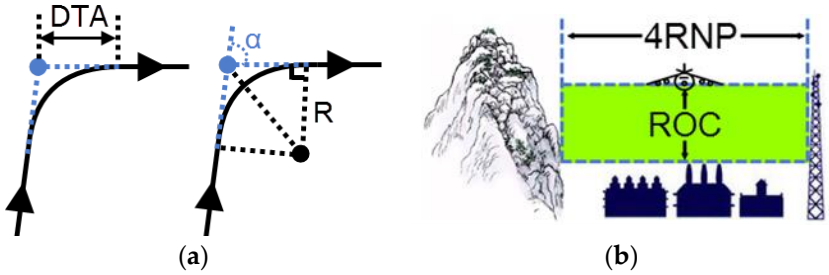


Figure 11. (**a**) Distance of turn anticipation model; (**b**) Profile view for RNP segment width.

4.4.3. Route Buffer Area

The route buffer area provides a vertical and horizontal buffer for the route for UAV safety. For the shadow regions shown in Figure 11b, the route buffer area regards the route as the centre line, with a width of 4 × RNP, and the height is a rectangular buffer needed to override the obstacle (ROC). We do interpolation for the entire buffer. If the altitude difference between the interpolation point and the terrain is less than ROC, the height of the waypoint needs to be increased so that the route buffer area does not intersect the terrain and the threat zone.

5. Results and Discussion

In this chapter, we first introduce the platform and parameters in Section 5.1. Pilot experiments with the route planner are reported in Sections 5.2–5.4. In Section 5.5, we compare six types of planners. To compare the different algorithms, we do not introduce dynamic threats and dynamic threat avoidance algorithms from Section 5.1 to Section 5.5. The planner in this paper is currently named MACO-pl. In Section 5.6, we introduce the dynamic threat and evaluate the performance of MACOD-pl.

5.1. Platform and Parameters

On the basis of the above methods, we developed a route planning system based on a virtual globe platform. This system provides interactive functions such as parameter input, route generation, route editing, route evaluation, threat modelling, data management, flight simulation and so on. Some interactive screenshots can be found in Appendix A. All of the route planning experiments that follow were completed on our route planning platform. The hardware environment was as follows: CPU: Core i7-4790; memory: 8GB DDR3 1600 MHz. The flight parameters of the UAV are shown in Table 1.

Table 1. The UAV flight parameters.

V_{KTAS} (km/h)	V_{KTW} (km/h)	Maximum Lifting Angle	h, ROC (m)	RNP (nmi)	Turning Fix
350	0	20°	400	0.25	Fly-by Fix

The route buffer area width was 2 × RNP = 0.5 nmi, and h_{max} and h_{min} were 5000 and 0, respectively. The algorithm processing time was the average time of the three experiments. The implementation of the algorithm used different programming skills, programming languages and compilers, all of which have large impacts on performance; as a result, the processing time can only be used as a reference.

Much literature research and discussion was reviewed for the selection of parameters in the ACO algorithm [37]. The selection of parameters was based on a large number of experiments. For a specific application, the ACO algorithm requires many experiments to determine better parameters. Through experiments, we found that the processing time of the ACO algorithm was inversely proportional to the evaporation coefficient. With the increase of the evaporation coefficient, the positive feedback effect of pheromones was increased. However, the increase of the evaporation coefficient may cause the algorithm to sink into a local optimum. If the number of ants is too small, the algorithm may terminate prematurely; however, too many ants can increase the time of algorithm iterations. In this study, the parameters of the pheromone were: retention coefficient: $\rho = 0.7$; weight coefficient: $\alpha = 1$, $\beta = 4$; ant number: 30; iteration number: 200; node number n between the start point and end point; threshold $\delta = 10$ m; local uplift threshold in route compression: $£ = 10$ m; and interpolation interval: $\forall = 100$ m.

This experiment used five coordinate groups, as shown in Table 2.

Table 2. Experimental coordinate groups.

Serial Number	Start Point	End Point	L_{min}	H_{avT}
	(x, y, H)	(x, y, H)	(km)	(m)
1	99.96463°	99.43347°	110.613	2339.93
	26.88202°	27.75616°		
	1819.58 m	2530.68 m		
2	90.86051°	94.46516°	350.051	4492.95
	29.30996°	29.42218°		
	3573.99 m	2951.15 m		
3	121.84255°	119.05692°	342.512	4.49
	30.98066°	32.95193°		
	21.51 m	14.17 m		
4	99.44990°	118.45611°	1909.435	1186.37
	27.81014°	25.09058°		
	1723.22 m	397.35 m		
5	109.78029°	120.41494°	2259.830	203.71
	18.85893°	37.01404°		
	445.07 m	132.31 m		

The planning space needs to be generated in advance, and the altitude property of the node can be stored off-line and can be used based on need. In this study, we generated a multi-granularity planning space; the pre-generated longitude range was [90, 125] and the pre-generated latitude range was [15, 45], corresponding to more than 10,000,000 square kilometres of planning area. At a grid width of 0.01°, the number of nodes was 10,500,000, and the grid space required approximately 5 min to be generated in the experimental machine. Before the implementation of the planning algorithm, we could load the corresponding offline block based on the size of the planning space.

Figure 12 shows the planning area of the first experimental coordinate group; the area is surrounded by the points (100.284, 27.622), (100.047, 26.723), (99.093, 26.920), (99.295, 27.812). Points A and B were the start and end points, respectively.

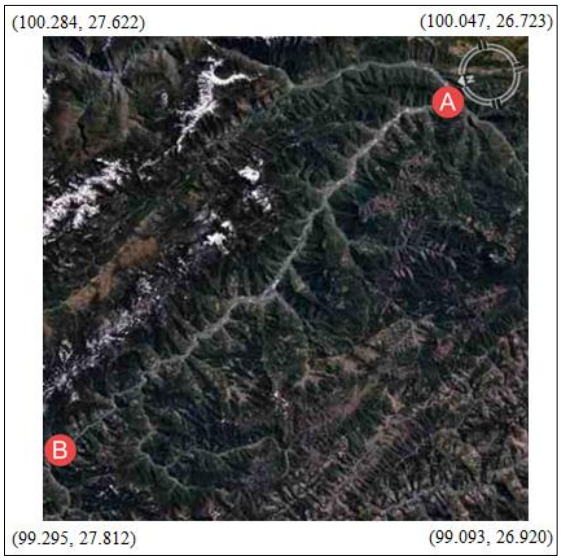


Figure 12. Cartographic Map-1.

5.2. Get the Initial Route

The parameter ε balances the valley-following ability and the length of the route. Different experimental results can be obtained by setting different coefficients. The initial altitude of the route is the sum of the node elevation and the ROC value. The experimental results are shown in Figure 13 and Table 3. It can be concluded that when ε was relatively high, the valley-following performance of the algorithm was better. However, a higher ε will increase the difficulty for an ant to reach the end point, which also increases the running time of the algorithm. When ε was 0, the algorithm calculated the shortest path; however, the path was longer than the shortest path in the continuous world because of the impact of irregular topography and the grid space. When $\varepsilon \geq 3$, a feasible solution may not be obtained. Suitable values of ε can be selected according to the application requirements. For the following experiments we chose $\varepsilon = 2$.

Table 3. Comparison results for different coefficients.

Coefficients (ε)	Planning Time (s)	L_{route} (km)	h_{avT} (m)	Waypoint Number	Compared with L_{min}	Compared with H_{avT}
0	1.01	118.428	2488.07	86	+7.07%	+6.33%
0.5	1.09	119.074	2351.98	87	+7.65%	+0.51%
1	1.20	119.924	2152.15	88	+8.42%	−8.03%
1.5	1.29	119.995	2138.84	89	+8.48%	−8.59%
2	1.37	120.617	2058.38	90	+9.04%	−12.03%
2.5	1.55	123.085	2028.94	92	+11.28%	−13.29%
3	N/A					

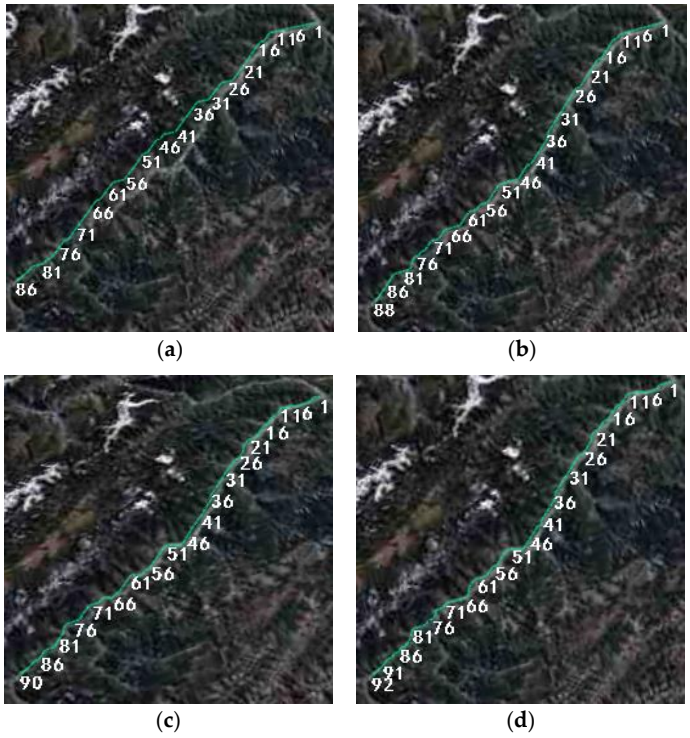


Figure 13. Experiment results with different height cost coefficients ε. (**a**) $\varepsilon = 0$; (**b**) $\varepsilon = 1$; (**c**) $\varepsilon = 2$; (**d**) $\varepsilon = 2.5$.

Figure 14 shows the convergence results of the MACO algorithm when $\varepsilon = 2$.

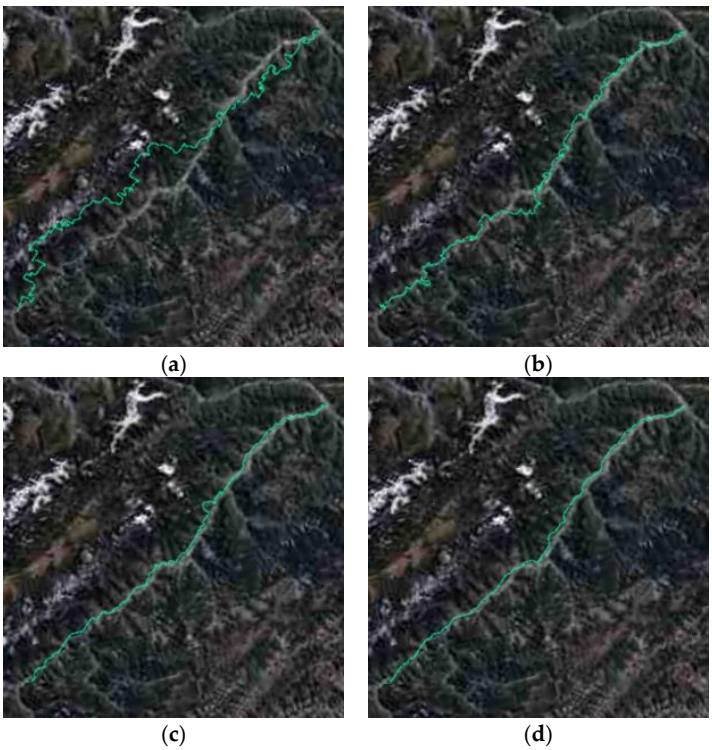


Figure 14. Global optimal solutions for different iterations t. (**a**) t = 1; (**b**) t = 25; (**c**) t = 50; (**d**) t = 100.

5.3. Route Optimization

5.3.1. Route Compression

The results of Figure 13c were first optimized by compression. As shown in Figure 15 and Table 4, different results were obtained by setting a different tolerance ϵ.

Table 4 shows that the length of the compressed route may have been shorter than the shortest path obtained by the ACO algorithm in Table 3.

There are both advantages and disadvantages in the influence of tolerance, and the tolerance should be selected based on the actual needs.

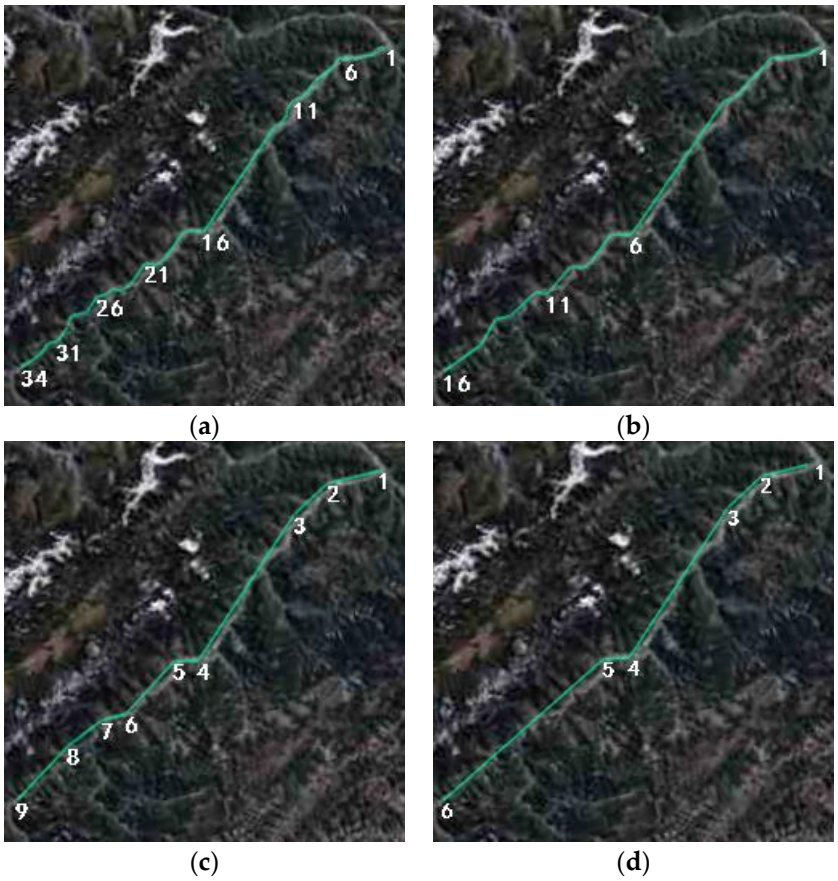


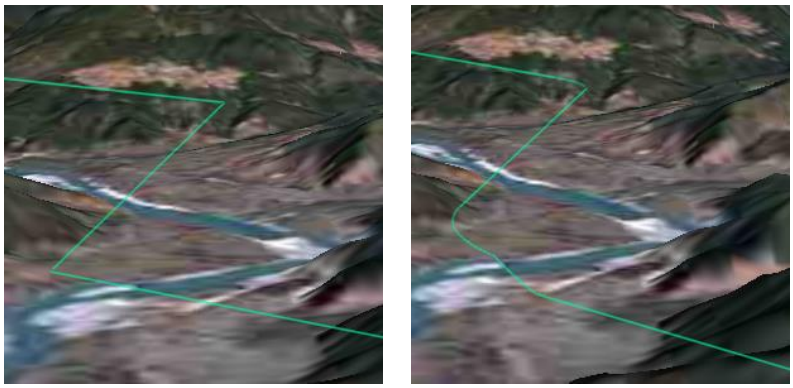
Figure 15. Comparison results for different tolerances ϵ. (**a**) ϵ = 0.25 nmi; (**b**) ϵ = 0.50 nmi; (**c**) ϵ = 0.75 nmi; (**d**) ϵ = 1.00 nmi.

Table 4. Comparison results for different tolerances.

ϵ (nmi)	L_{route} (km)	h_{avT} (m)	Waypoint Number	Compared with L_{min}	Compared with H_{avT}
0.25	120.129	2063.33	34	+8.60%	−11.82%
0.50	117.172	2060.58	16	+5.93%	−11.94%
0.75	116.003	2141.67	9	+4.87%	−8.47%
1.00	115.061	2120.13	6	+4.02%	−9.39%

5.3.2. Smooth Route Turning

As Figure 16 shows, we used Equation (16) and Fly-by Fix to address all UAV turning corners. With respect to the results of Figure 16, the route length after adjustment was 115.495 km with an average terrain height of 2142.80 m. The turning curve is a set of route control points with an interval of 50 m.



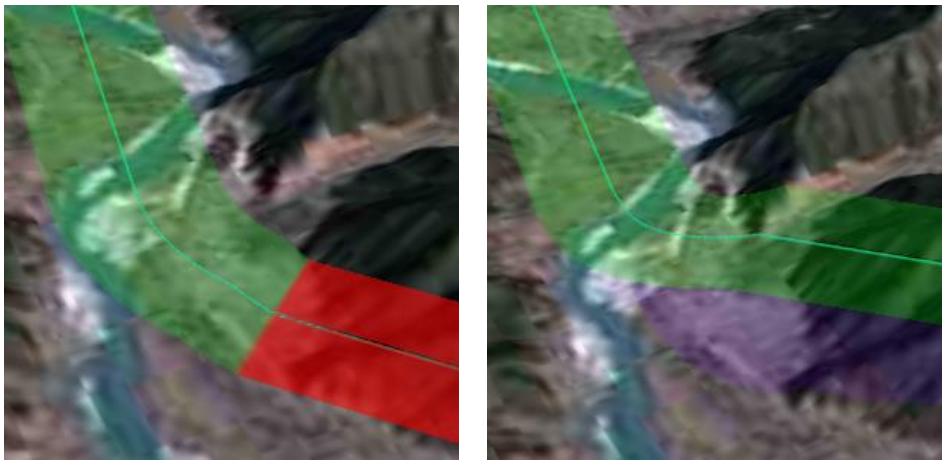
(**a**) (**b**)

Figure 16. Turning, falling and rising corner adjustment. (**a**) Before turning corner adjustment; (**b**) After turning corner adjustment.

5.3.3. Route Buffer Areas

Figure 17a is the result of the introduction of route buffer zones. A route segment that does not meet the flight safety requirements is marked in red. Figure 17b is the result after height adjustment. After the introduction of the route buffer zone, the route length was adjusted to 115.609 km with an average height of 3228.90 m, and the terrain average height was 2142.50 m. Figure 18 is the vertical figure of route and terrain. The Terrain Altitude-curve is the terrain altitude that the planning route passes through, and the 2RNP Terrain Altitude-curve is the highest terrain altitude within the 2RNP range. The initial route was unsafe; we obtained a safe route that could meet the safety requirements after adjusting the elevation.

After route optimization, $L_{US} = 0$; however, the average height of the route changed and the average terrain height passed through by the UAV also changed. Errors introduced by the optimization process may cause a route to pass through a threat area, as was mentioned in Section 3.4.1, and the threat area should thus be expanded when it is mapped into the planning space. The expansion width of the threat zone should be greater than the navigation performance error (0.5 nmi); the width of the corresponding expansion in the 0.01° grid width planning space was 1 node.



(**a**) (**b**)

Figure 17. Route optimization with route buffer areas. (**a**) Before height adjustment; (**b**) After height adjustment.

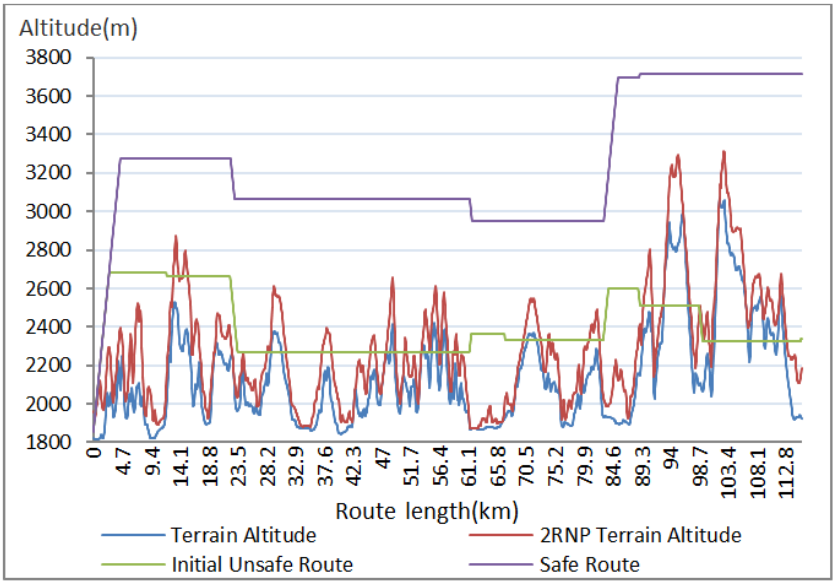


Figure 18. Vertical flight path.

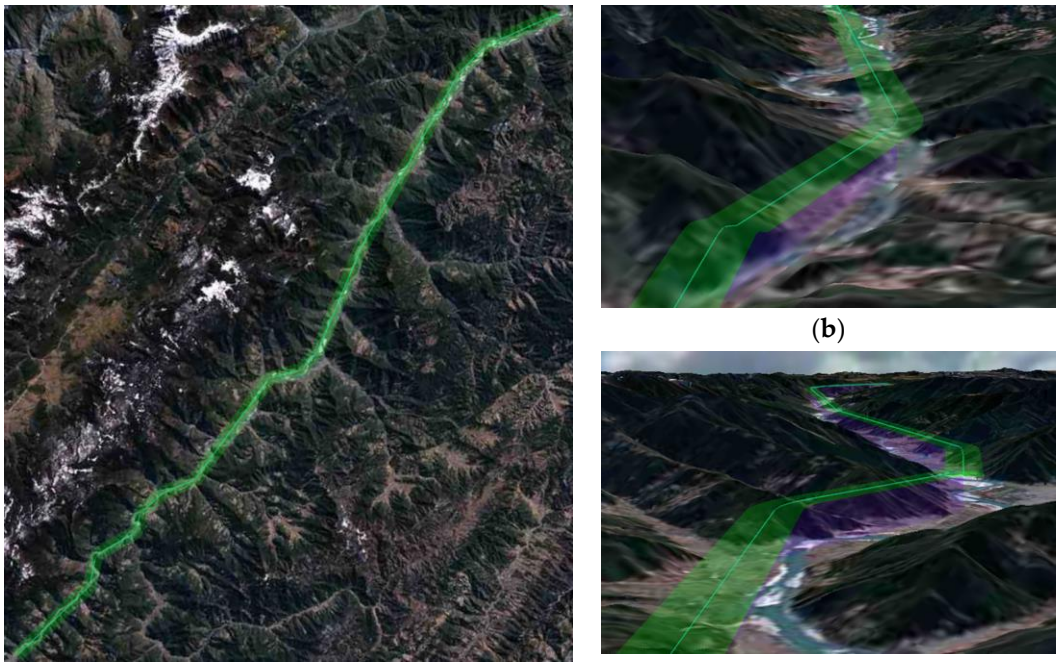
5.4. Multi-Granularity Results

In Section 3.4.2, we introduced a multi-granularity planning space suitable for different applications. Based on different grids, the experimental results are shown in Table 5.

Table 5. Comparison of results with different granularity.

Grid Width	ϵ (nmi)	Processing Time (s)	L_{route} (km)	h_{avT} (m)	Compared with L_{min}	Compared with H_{avT}
0.007°	0.75	2.15	115.117	1975.70	+4.07%	−15.57%
0.007°	0.50	2.15	116.899	1947.84	+5.68%	−16.76%
0.006°	0.75	2.73	114.404	1979.44	+3.43%	−15.41%
0.006°	0.50	2.73	117.060	1953.05	+5.83%	−16.53%
0.005°	0.75	3.21	114.086	1942.91	+3.14%	−16.97%
0.005°	0.50	3.21	115.603	1905.73	+4.51%	−18.56%
0.004°	0.75	4.24	114.403	1955.89	+3.43%	−16.41%
0.004°	0.50	4.24	115.523	1923.16	+4.44%	−17.81%
0.003°	0.75	6.18	114.498	1961.64	+3.51%	−16.17%
0.003°	0.50	6.18	115.300	1925.10	+4.24%	−17.73%
0.002°	0.75	7.37	115.033	1973.27	+4.00%	−15.67%
0.002°	0.50	7.37	115.711	1961.85	+4.61%	−16.16%

As the grid width was reduced and the planning space became closer to the real world, the route we obtained was more consistent with reality. The algorithm ran slower; however, when the grid width was reduced to a certain value limited by factors such as the input parameters and the experimental terrain data source, the results no longer improved. In the experiment scene shown in Figure 19, a better effect was achieved when grid width was 0.005° and ϵ was 0.50 nmi. The following experiments are based on these parameters.



(**a**) (**b**) (**c**)

Figure 19. (**a**) The optimal result for grid width = 0.005°, $\epsilon = 0.50$ nmi, $\varepsilon = 2$, terrain average altitude = 1905.73 m, and average altitude of the route after introducing the route buffer = 2698.37 m; (**b**,**c**) are local route details.

5.5. Comparison Experiments

On our virtual globe platform we also implemented the route planner of the basic ACO pheromone model named ACO-pl and the Rank-based pheromone model named RAS-pl. The number of retaining ants of RAS-pl was 5. We used the local valley-following method described in Section 4.2.2 and the shortest distance from the target as the heuristic function to realize an A* based planner, named A*-pl.

Additionally, we implemented the planners based on PSO and EA, named PSO-pl and EA-pl, respectively. Because these two methods use the dimension-reduction encode method to achieve evolutionary computation, the local valley-following method in Section 4.2.2 was not applicable. This study used the minimal flight altitude method in [12] to update the average altitude of the route. Because of the interval difference between adjacent waypoints, interval interpolation for the route was needed when calculating the average altitude. In the following experiments, the inertial weight coefficient of PSO-pl was linear from 0.9 to 0.4. The iteration count of PSO-pl and EA-pl was 200 and the population size was 30.

When implementing the six types of planners, this study used data structures to store pre-calculated distances between the adjacent nodes and pre-calculated distances between each node and the end node. The adjacent waypoints in MACO-pl, ACO-pl, RAS-pl and A*-pl are adjacent nodes in the grid planning space, as shown in Figure 20a. The algorithm proposed and used in this study achieved computational acceleration. While the adjacent waypoints in PSO-pl and EA-pl are not necessarily adjacent nodes, as shown in Figure 20b, the waypoints P and Q can only move in two directions. At this point, P and Q are not adjacent nodes and thus distances must be calculated. Although the waypoints Q and H are adjacent nodes, we can obtain the distance between Q and H by a table lookup.

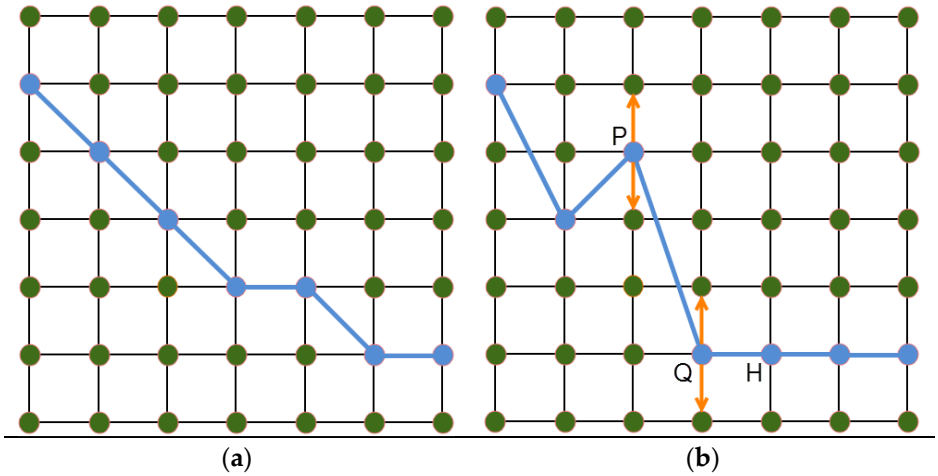


Figure 20. The relationship between waypoint and node for different algorithms. (**a**) MACO, ACO, RAS, A*; (**b**) PSO, EA.

The comparison experiments used the same grid size and UAV parameters and were conducted for different coordinate groups. The results are shown in Table 6 and Figure 21.

Table 6. Planning results of 1–5 coordinate groups.

Serial Number	Method	Processing Time (s)	L_{route} (km)	h_{avT} (m)	Compared with L_{min}	Compared with H_{avT}
1	MACO-pl	3.21	115.603	1905.73	+4.51%	−18.56%
1	ACO-pl	10.07	118.831	2262.32	+7.43%	−3.32%
1	RAS-pl	5.66	117.267	1914.28	+6.02%	−18.19%
1	A*-pl	2.16	116.875	1916.29	+5.66%	−18.10%
1	PSO-pl	9.78	118.679	2177.42	+7.29%	−6.94%
1	EA-pl	10.84	119.441	2190.32	+7.98%	−6.39%
2	MACO-pl	12.39	398.782	3471.04	+13.92%	−22.74%
2	ACO-pl	38.48	389.907	4074.93	+11.39%	−9.30%
2	RAS-pl	20.40	399.823	3623.61	+14.22%	−19.35%
2	A*-pl	148.14	396.934	3681.73	+13.39%	−18.05%
2	PSO-pl	40.02	386.679	4081.56	+10.46%	−9.15%
2	EA-pl	41.28	385.041	4094.39	+10.00%	−8.87%
3	MACO-pl	10.96	362.433	2.85	+5.82%	−36.53%
3	ACO-pl	36.42	368.345	2.98	+7.54%	−33.63%
3	RAS-pl	18.67	365.184	2.93	+6.62%	−34.74%
3	A*-pl	116.23	363.245	2.94	+6.05%	−34.52%
3	PSO-pl	39.57	366.176	4.01	+6.91%	−10.69%
3	EA-pl	38.95	364.504	3.95	+6.42%	−12.03%
4	MACO-pl	80.12	2115.333	1043.90	+10.78%	−12.01%
4	ACO-pl	201.15	2135.830	1097.04	+11.85%	−7.53%
4	RAS-pl	149.98	2136.106	1063.77	+11.87%	−10.33%
4	A*-pl	N/A				
4	PSO-pl	278.26	2104.857	1137.25	+10.23%	−4.14%
4	EA-pl	299.38	2109.891	1131.32	+10.50%	−4.64%
5	MACO-pl	78.84	2371.012	84.59	+4.92%	−58.48%
5	ACO-pl	197.43	2401.023	116.56	+6.25%	−42.78%
5	RAS-pl	136.01	2390.174	85.99	+5.76%	−57.79%
5	A*-pl	N/A				
5	PSO-pl	301.44	2398.322	127.76	+6.13%	−37.28%
5	EA-pl	279.64	2388.459	134.31	+5.69%	−34.07%

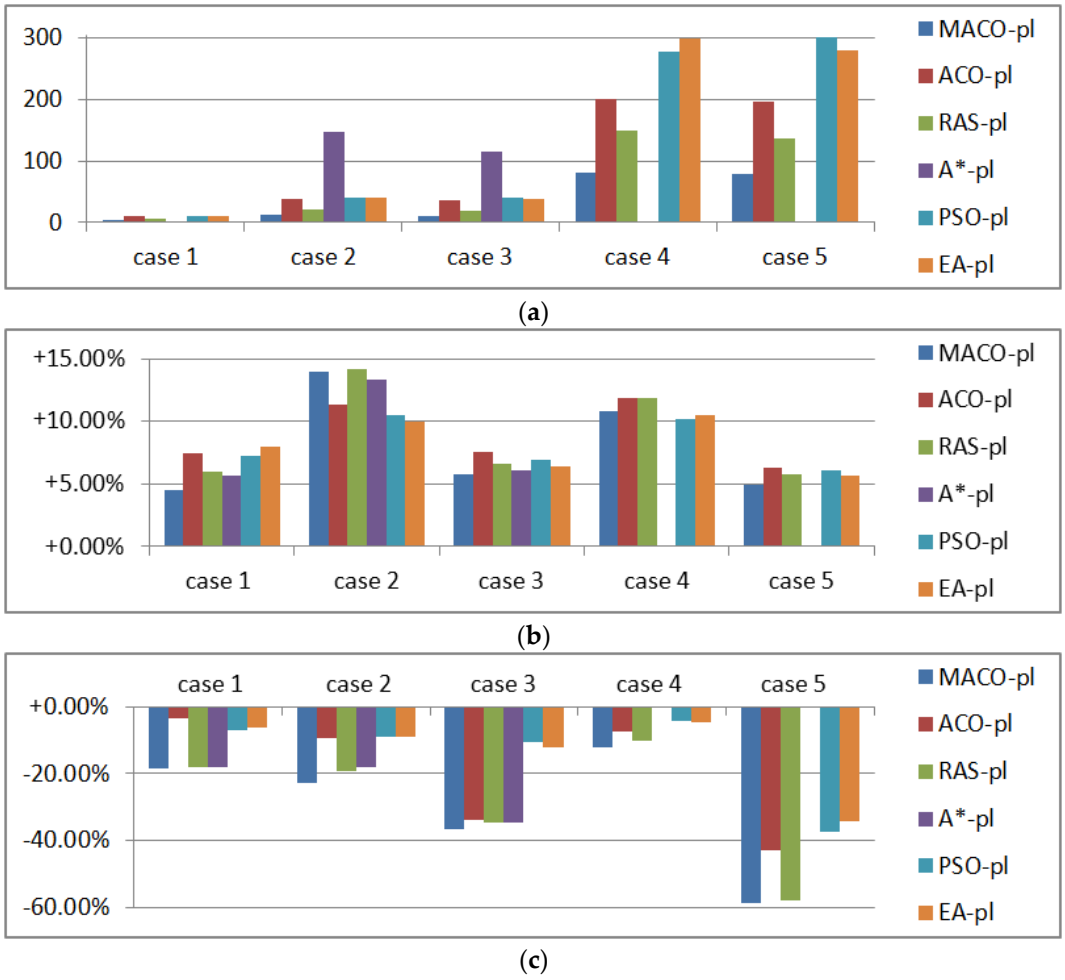


Figure 21. Planning results of 1–5 coordinate groups. (**a**) Processing time (s); (**b**) Comparison with L_{min}; (**c**) Comparison with H_{avT}.

The pheromone evaporation model of ACO-pl easily resulted in the local route planning optimum. Compared with ACO-pl, RAS-pl used the results of higher ranking ants and had better exploration ability. The A* algorithm is highly efficient in solving a simple shortest path problem; however, the performance of this algorithm was reduced when the local terrain was introduced. A*-pl can have good solutions in small scenes; however, its open table became increasingly large when the scene became larger and more complex. Although we introduced a binary heap to speed up the query, it remained very difficult to find a solution. The valley-following abilities of EA-pl and PSO-pl were not as good; however, the performance for the height and length of these methods was relatively balanced. Moreover, EA-pl and PSO-pl were less efficient than MACO-pl because of frequent interpolation and distance calculation. Compared with the A* algorithm, intelligent planners have advantages in solving large-scale complex problems. We demonstrated by the comparisons that MACO-pl produced a better solution and that the algorithm was more efficient.

Figure 22 shows the results of MACO-pl. In the area of Figure 22a, the planner could track the trend of the terrain in the valley, which reflects better terrain tracking ability. Figure 22b is mostly a plains area, and the trend of the route was relatively straight. Figure 22c,d show the planning results in large-scale planning scenarios. Figure 23 shows some local details of these routes.

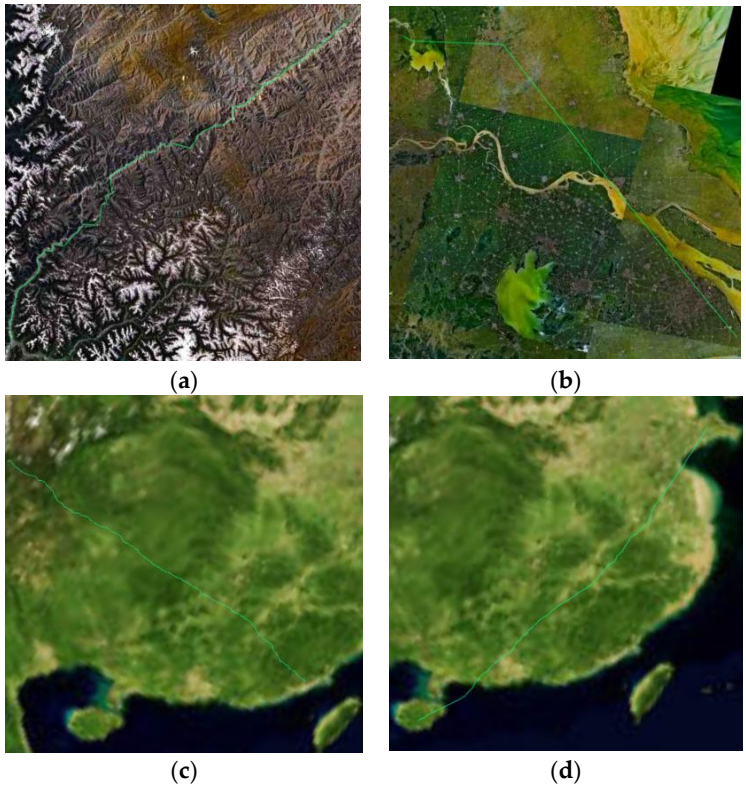


Figure 22. Planning results of 2–5 coordinate groups using MACO-pl. (**a**) Coordinate group 2; (**b**) Coordinate group 3; (**c**) Coordinate group 4; (**d**) Coordinate group 5.

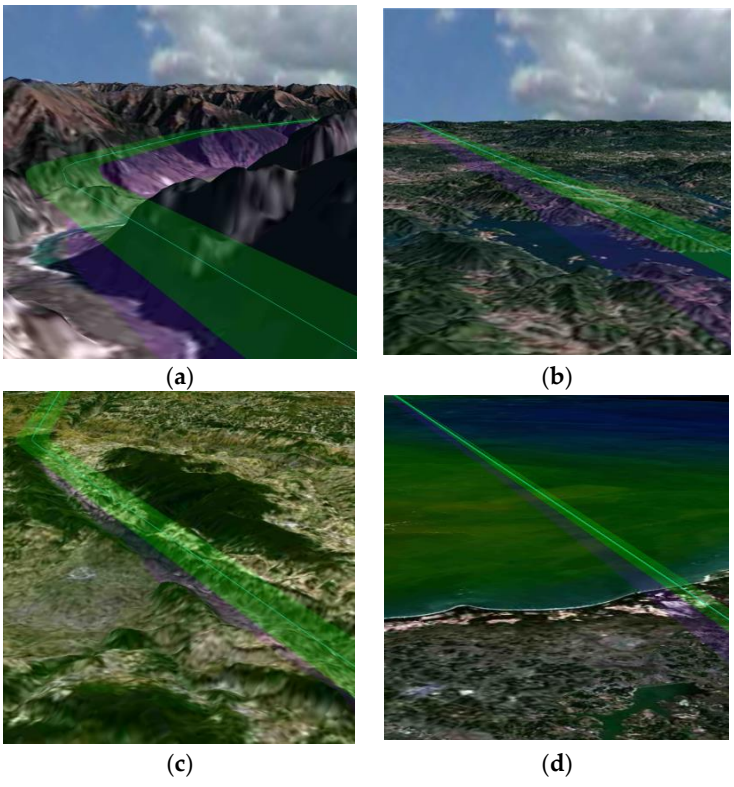
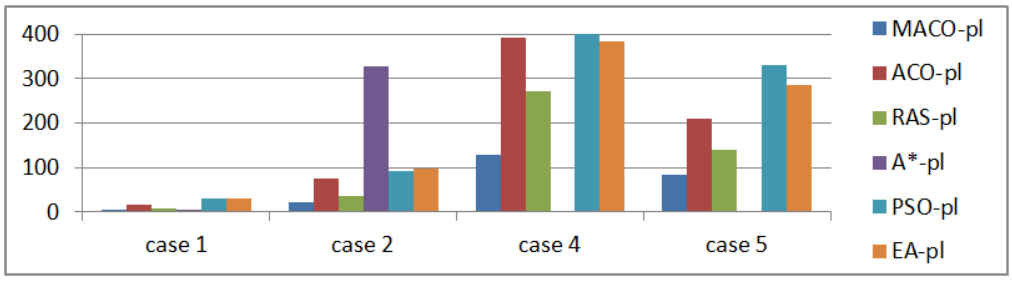


Figure 23. Local details of the routes of 2–5 coordinate groups. (**a**) Coordinate group 2; (**b**) Coordinate group 3; (**c**) Coordinate group 4; (**d**) Coordinate group 5.

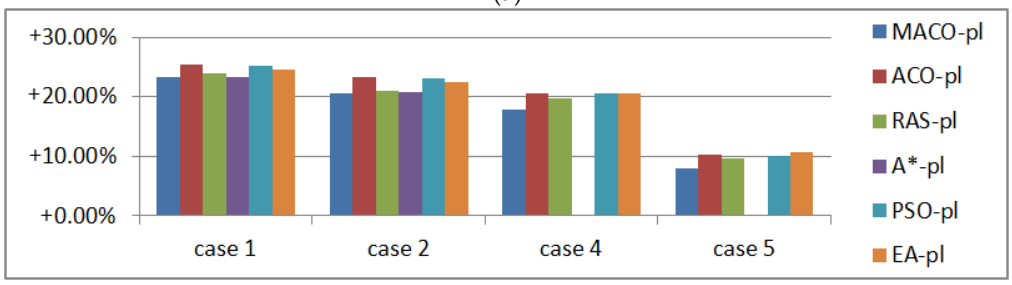
After marking the radar zone, no-fly zone, and missile threat zone, we did experiments on coordinate groups 1, 2, 4, and 5. Table 7 and Figure 24 show the results of different planners. We demonstrated that the MACO-pl had a better solution. Figure 25 shows local results by MACO-pl of four coordinate groups after plotting threatened areas. The planned routes could make use of the terrain condition to avoid the threatened area.

Table 7. Planning results in threat environments.

Serial Number	Method	Processing Time (s)	L_{route} (km)	h_{avT} (m)	L_{FA} (km)	Compared with L_{min}	Compared with H_{avT}
1	MACO-pl	5.63	136.480	2725.64	0	+23.39%	+16.48%
1	ACO-pl	17.02	138.737	2870.48	0	+25.43%	+22.67%
1	RAS-pl	9.39	137.158	2757.80	0	+24.00%	+17.86%
1	A*-pl	5.27	136.497	2789.72	0	+23.40%	+19.22%
1	PSO-pl	29.84	138.528	2955.42	0	+25.24%	+26.30%
1	EA-pl	31.15	137.917	2918.81	0	+24.68%	+24.74%
2	MACO-pl	20.85	421.922	4379.01	0	+20.53%	−2.54%
2	ACO-pl	75.35	431.943	4687.66	0	+23.39%	+4.33%
2	RAS-pl	37.28	423.871	4470.49	0	+21.09%	−0.50%
2	A*-pl	326.92	422.775	4450.11	0	+20.78%	−0.95%
2	PSO-pl	91.49	430.563	4716.47	0	+23.00%	+4.97%
2	EA-pl	99.02	428.719	4671.99	0	+22.47%	+3.98%
4	MACO-pl	127.39	2250.836	1086.37	0	+17.88%	−8.43%
4	ACO-pl	391.50	2301.760	1143.74	0	+20.55%	−3.59%
4	RAS-pl	271.98	2284.541	1109.44	0	+19.64%	−6.49%
4	A*-pl	N/A					
4	PSO-pl	401.32	2304.283	1217.07	0	+20.68%	+2.59%
4	EA-pl	384.56	2300.575	1203.64	0	+20.48%	+1.46%
5	MACO-pl	84.26	2438.912	86.22	0	+7.92%	−57.68%
5	ACO-pl	209.92	2494.026	119.14	0	+10.36%	−41.51%
5	RAS-pl	140.15	2477.678	86.85	0	+9.64%	−57.37%
5	A*-pl	N/A					
5	PSO-pl	329.48	2489.205	129.49	0	+10.15%	−36.43%
5	EA-pl	286.77	2503.418	136.61	0	+10.78%	−32.94%

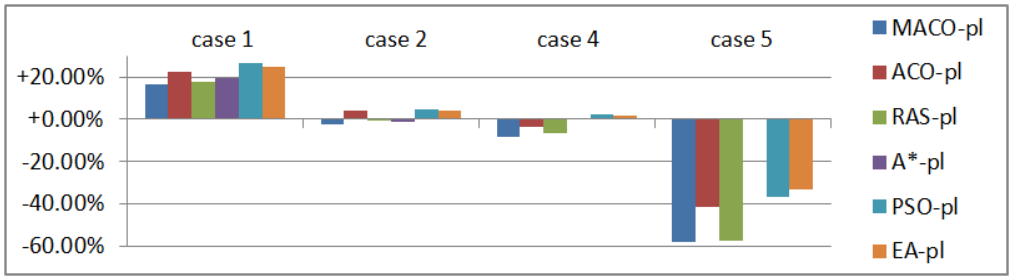


(**a**)



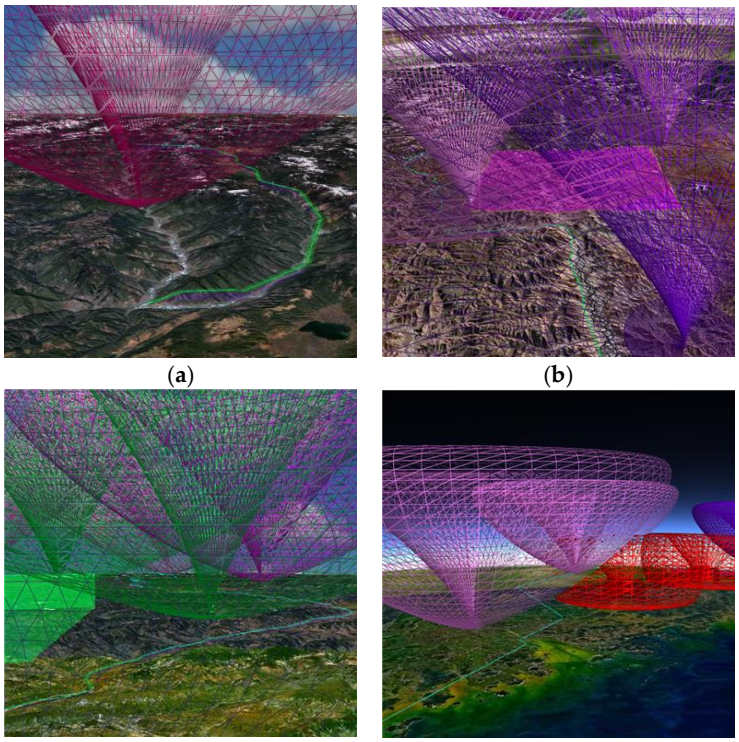
(**b**)

Figure 24. *Cont.*



(**c**)

Figure 24. Planning results under threat environments. (**a**) Processing time (s); (**b**) Comparison with L_{min}; (**c**) Comparison with H_{avT}.



(**a**) (**b**)

(**c**) (**d**)

Figure 25. Low altitude penetration routes using MACO-pl of different cases. (**a**) Case 1; (**b**) Case 2; (**c**) Case 4; (**d**) Case 5.

5.6. Experiments on Dynamic MACO

This section introduces the dynamic threat area discussed in Section 3.3. Based on MACOD-pl, we used coordinate group 1 to test the algorithm. Considering that the route optimization algorithms would bring approximately 1%–3% error to the initial route length, which is approximately 1–4 km under the current scenario, the maximum time error was approximately 40 s, and the moving speed of the dynamic threat area in the figure was approximately 12.6 m/s. The maximum error range was thus approximately 510 m. To ensure route safety, the boundary of the dynamic threat area was expanded by 0.01°. As shown in Figure 26 and Table 8, when the start time of the UAV was not the same, the UAV could meet the threat at different times and places, and the planned routes were thus different. When the start time of the UAV was between 0 and 2000 s, the dynamic threat area would block the UAV from flying in the valley area. When the start time of the UAV was 3000 s, the dynamic threat area would pass through the valley area and the result of route planning was similar to the result without the threat. Because of the introduction of dynamic threats, the dynamic threat avoidance algorithm needed

to consume many computing resources, and the operating efficiency of the MACOD-pl algorithm was much lower than that of the MACO-pl algorithm.

Table 8. The results of different start times under dynamic threats.

Start Time (s)	Planning Time (s)	L_{route} (km)	h_{avT} (m)	Compared with L_{min}	Compared with H_{avT}
0	10.01	133.850	2112.65	+21.01%	−9.71%
1000	12.14	135.633	2724.18	+22.62%	+16.42%
2000	12.89	136.204	2725.49	+23.14%	+16.48%
3000	9.49	115.377	1906.71	+4.31%	−18.51%

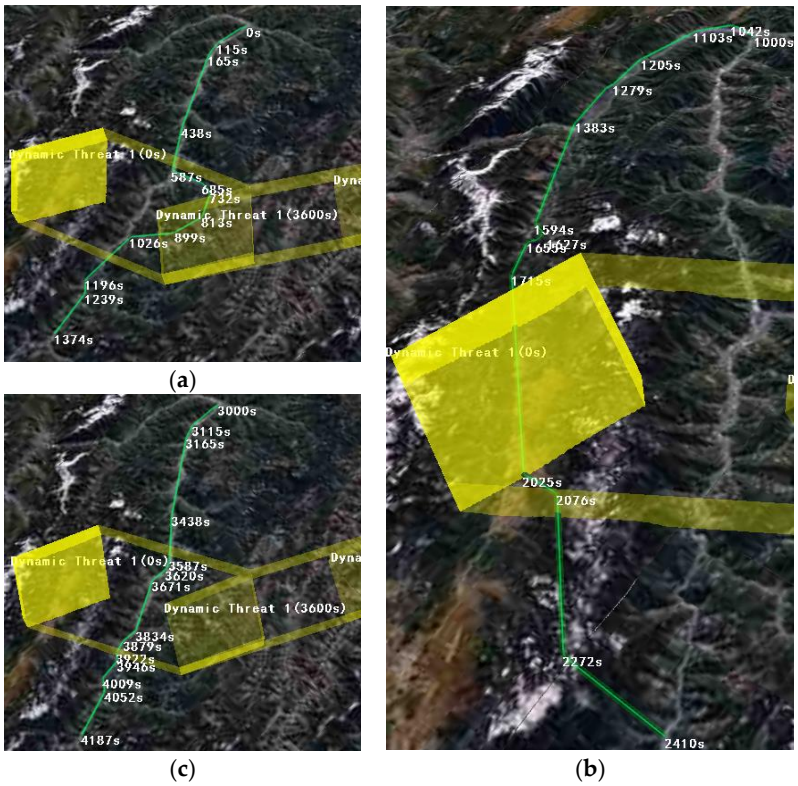


Figure 26. The results of different start times under dynamic threats. (**a**) Start time = 0 s; (**b**) Start time = 1000 s; (**c**) Start time = 3000 s.

6. Conclusions and Future Work

This paper presented an improved ACO-based planner based on the virtual globe platform to solve the route planning problem in risk environments. We developed a route planning system and realized six different planners in the static threat. By experimental analysis, we demonstrated that our optimum planner had better performance in terms of fuel consumption, terrain masking, and risk avoidance. Finally, we demonstrated the effectiveness of MACOD-pl in the dynamic threat environment. Benefiting from the virtual globe platform, the method presented in this paper can achieve route planning globally and meet the current maximum combat radius of UAVs in battlefields. The planned route can effectively avoid various threats and, by changing the parameters, the planner is suitable for military penetration, search-and-rescue, and many other scenarios. Our route planning system can perform not only automatic planning but also route editing and storage, flight simulation experiments, and other tasks.

Many other types of planning algorithms exist; for example, some algorithms use a rotated coordinate frame to reduce the dimension of the problem [12,18]. Further research on how to apply

other algorithms to the virtual globe platform and efficiently obtain better results would be worthwhile, as well as further research on the model of different dynamic threats and the efficient use of dynamic threat avoidance algorithms.

Acknowledgments: The authors would like to thank the support of Shenzhen Government, Project No. GRCK2015092515503432. We would also like to thank NASA for providing data.

Author Contributions: Ming Zhang and Yuesheng Zhu designed the project; Ming Zhang, Chen Su, Yuan Liu and Mingyuan Hu prepared the manuscript. All the authors have participated in the project.

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

Appendix A

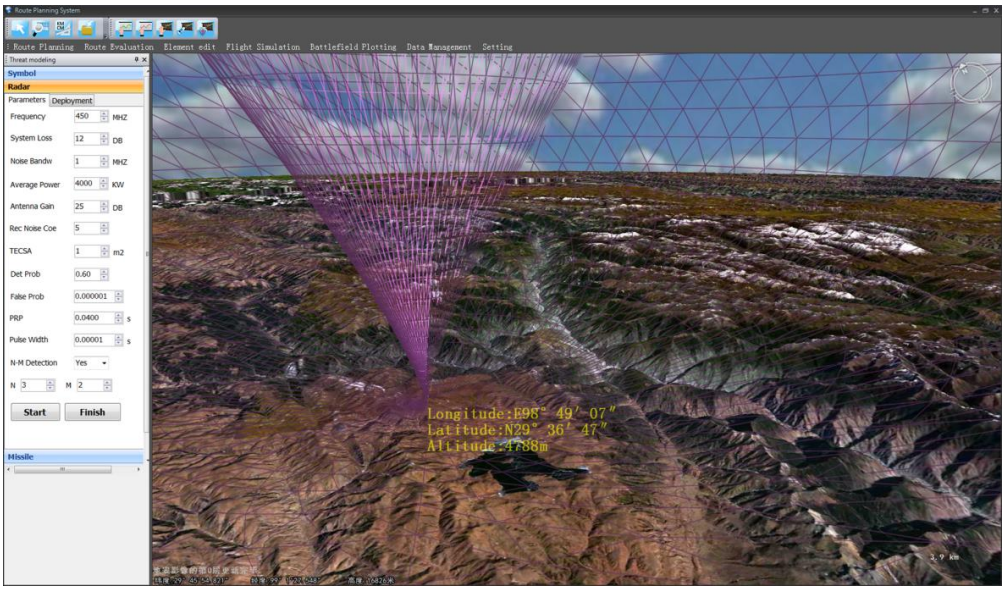


Figure A1. Threat modelling in the route planning system in which a radar threat zone is marked interactively.

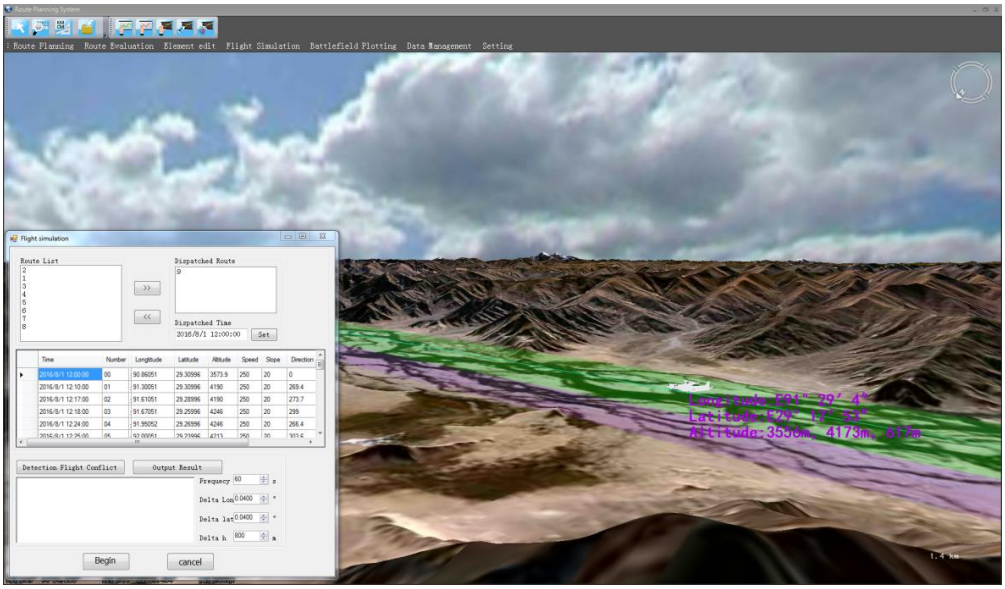


Figure A2. Screenshot of a flight simulation in the route planning system in which a simulated flight experiment is performed for the saved route.

References

1. Kimon, P.V.; George, J.V. *Handbook of Unmanned Aerial Vehicles*; Springer: Heidelberg, Germany, 2015.
2. Neto, A.A.; Macharet, D.G.; Campos, M.F.M. Feasible RRT-based path planning using seventh order Bézier curves. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 18–22 October 2010.
3. Goerzen, C.; Kong, Z.; Mettler, B. A survey of motion planning algorithms from the perspective of autonomous UAV guidance. *J. Intell. Robot. Syst.* **2010**, *57*, 65–100. [CrossRef]
4. Nelson, D.R.; Barber, D.B.; McLain, T.W.; Beard, R.W. Vector field path following for miniature air vehicles. *IEEE Trans. Robot.* **2007**, *23*, 519–529. [CrossRef]
5. Chen, H.; Chang, K.; Agate, C.S. UAV path planning with Tangent-plus-Lyapunov vector field guidance and obstacle avoidance. *IEEE Trans. Aerosp. Electron. Syst.* **2013**, *49*, 840–856. [CrossRef]
6. Lee, J.; Pippin, C.; Balch, T. Cost based planning with RRT in outdoor environments. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008), Nice, France, 22–26 September 2008.
7. Wen, N.; Zhao, L.; Su, X.; Ma, P. UAV online path planning algorithm in a low altitude dangerous environment. *IEEE/CAA J. Autom. Sin.* **2015**, *2*, 173–185.
8. Wen, N.; Zhao, L.; Su, X.; Ma, P. Online UAV path planning in uncertain and hostile environments. *Int. J. Mach. Learn. Cyber.* **2015**, 1–19. [CrossRef]
9. Ergezer, H.; Leblebicioglu, K. Path planning for UAVs for maximum information collection. *IEEE Trans. Aerosp. Electron. Syst.* **2013**, *49*, 502–520. [CrossRef]
10. Zheng, C.; Li, L.; Xu, F.; Sun, F.; Ding, M. Evolutionary route planner for unmanned air vehicles. *IEEE Trans. Robot.* **2005**, *21*, 609–620. [CrossRef]
11. Mittal, S.; Deb, K. Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms. In Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2007), Singapore, 25–28 September 2007.
12. Yang, P.; Tang, K.; Lozano, J.A.; Cao, X. Path planning for single unmanned aerial vehicle by separately evolving waypoints. *IEEE Trans. Robot.* **2015**, *31*, 1130–1146. [CrossRef]
13. Fu, Y.; Ding, M.; Zhou, C.; Hu, H. Route planning for Unmanned Aerial Vehicle (UAV) on the sea using hybrid differential evolution and quantum-behaved particle swarm optimization. *IEEE Trans. Syst. Man Cybern. Syst.* **2013**, *43*, 1451–1465. [CrossRef]
14. Ling, X.; Hao, Y. Effective 3-D path planning for UAV in presence of threat netting. In Proceedings of the 2015 Fifth International Conference on Communication Systems and Network Technologies (CSNT), Gwalior, India, 4–6 April 2015.
15. Duan, H.B.; Yu, Y.X.; Zhang, X.Y.; Shao, S. Three-dimension path planning for UCAV using hybrid meta-heuristic ACO-DE algorithm. *Simul. Model. Pract. Theory* **2010**, *18*, 1104–1115. [CrossRef]
16. Garcia, M.; Viguria, A.; Ollero, A. Dynamic graph-search algorithm for global path planning in presence of hazardous weather. *J. Intell. Robot. Syst.* **2013**, *69*, 285–295. [CrossRef]
17. Zhou, Y.; Bao, Z.; Wang, R.; Qiao, S.; Zhou, Y. Quantum wind driven optimization for unmanned combat air vehicle path planning. *Appl. Sci.* **2015**, *5*, 1457–1483. [CrossRef]
18. Yao, P.; Wang, H. Dynamic Adaptive Ant Lion Optimizer applied to route planning for unmanned aerial vehicle. *Soft Comput.* **2016**. [CrossRef]
19. Le, Y.; Peng, G. Google Earth as a virtual globe tool for Earth science applications at the global scale: progress and perspectives. *Int. J. Remote Sens.* **2012**, *33*, 3966–3986.
20. Zheng, G.; Zheng, Y. Radar netting technology & its development. In Proceedings of the 2011 IEEE CIE International Conference on Radar, Chengdu, China, 24–27 October 2011.
21. Bell, D.G.; Kuehnel, F.; Maxwell, C.; Kim, R.; Kasraie, K.; Gaskins, T.; Hogan, P.; Coughlan, J. NASA world wind: Opensource GIS for mission operations. In Proceedings of the 2007 IEEE Aerospace Conference, Big Sky, MT, USA, 3–10 March 2007.
22. United States Standard for Performance Based Navigation (PBN) Instrument Procedure Design Document Information. Available online: http://www.faa.gov/regulations_policies/ (accessed on 10 October 2015).
23. Mingan, Z. Killing zone boundary and defense efficiency of surface-air missile. *Tactical Missile Technol.* **1992**, *1*, 1–8. (In Chinese)

24. Haifeng, L. A Study on Spatial Modeling Method of Threats in Mission Rehearsal of Tactical Aviation. Master's Thesis, National University of Defense Technology, Changsha, China, 2006. (In Chinese)
25. Skolnik, M.I. *Introduction to Radar Systems*, 3rd ed.; McGraw-Hill Book Company: New York, NY, USA, 2002.
26. Hebert, D.J. A box spline subdivision pyramid algorithm. *Appl. Math. Lett.* **1999**, *12*, 57–62. [CrossRef]
27. Donoho, D.L.; Yu, P.Y. Nonlinear pyramid transforms based on median-interpolation. *SIAM J. Math. Anal.* **2000**, *31*, 1030–1061. [CrossRef]
28. Vishwanath, M. The recursive pyramid algorithm for the discrete wavelet transform. *IEEE Trans. Signal Process.* **1994**, *42*, 673–676. [CrossRef]
29. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Cybern.* **1996**, *26*, 29–41. [CrossRef] [PubMed]
30. Dorigo, M.; Gambardella, L.M. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Trans. Evol. Comput.* **1997**, *1*, 53–66. [CrossRef]
31. Adubi, S.A.; Misra, S. A comparative study on the ant colony optimization algorithms. In Proceedings of the 11th International Conference on Electronics, Computer and Computation (ICECCO), Abuja, Nigeria, 29 September–1 October 2014.
32. Albinati, J.; Oliveira, S.E.; Otero, F.E.; Pappa, G.L. An ant colony-based semi-supervised approach for learning classification rules. *Swarm Intell.* **2015**, *9*, 315–341. [CrossRef]
33. Farahnakian, F.; Ashraf, A.; Pahikkala, T.; Liljeberg, P.; Plosila, J.; Porres, I.; Tenhunen, H. Using ant colony system to consolidate VMs for green cloud computing. *IEEE Trans. Serv. Comput.* **2015**, *8*, 187–198.
34. Wang, Z.; Xing, H.; Li, T.; Yang, Y.; Qu, R.; Pan, Y. A modified ant colony optimization algorithm for network coding resource minimization. *IEEE Trans. Evol. Comput.* **2015**, *1*, 1–18. [CrossRef]
35. Adel, A.; Akbar, H. Autonomously implemented versatile path planning for mobile robots based on cellular automata and ant colony. *Int. J. Comput. Intell. Syst.* **2012**, *5*, 39–52.
36. Saalfeld, A. Topologically consistent line simplification with the douglas-peucker algorithm. *Cartogr. Geogr. Inf. Sci.* **1999**, *26*, 7–18. [CrossRef]
37. Stützle, T.; López-Ibánez, M.; Pellegrini, P.; Maur, M.; De Oca, M.M.; Birattari, M.; Dorigo, M. *Parameter Adaptation in Ant Colony Optimization. Autonomous Search*; Springer: Heidelberg, Germany, 2011.