Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 597092, 12 pages
http://dx.doi.org/10.1155/2014/597092

Research Article

Efficient UAV Path Planning with Multiconstraints in a 3D Large Battlefield Environment

Weiwei Zhan, Wei Wang, Nengcheng Chen, and Chao Wang

State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, 129 Luoyu Road, Wuhan 430079, China

Correspondence should be addressed to Chao Wang; c.wang@whu.edu.cn

Received 12 October 2013; Revised 6 January 2014; Accepted 6 January 2014; Published 26 February 2014

Academic Editor: Xinjie Zhang

This study introduces an improved A^* algorithm for the real-time path planning of Unmanned Air Vehicles (UAVs) in a 3D large-scale battlefield environment to solve the problem that UAVs require high survival rates and low fuel consumption. The algorithm is able to find the optimal path between two waypoints in the target space and comprehensively takes factors such as altitude, detection probability, and path length into account. It considers the maneuverability constraints of the UAV, including the safety altitude, climb rate, and turning radius, to obtain the final flyable path. Finally, the authors test the algorithm in an approximately 2,500,000 square meter area containing radars, no-fly zones, and extreme weather conditions to measure its feasibility, stability, and efficiency.

1. Introduction

With the development of various modern high-maneuverability air defense weapons and increasingly perfected air defense system, the capability of vehicles to break through enemy defenses at medium and high altitudes has decreased. Modern military low-altitude penetration is primarily performed by UAVs. The key lies in the path planning for UAVs [1]. Research in this area has been ongoing for many years, and several algorithms were developed for this problem.

The artificial potential field method was developed by Khatbi to plan the trajectory for the robots. This algorithm is intelligible in its mathematical description and has been widely used for real-time obstacle avoidance and smooth trajectory control [2]. However, the method has its inherent limits when applied to UAV path planning [3]; for example, in complex mountainous terrains, there are often multiple obstacles near the target point; it would result in a greater repulsion than attraction for the UAV. Thus, the UAV cannot reach the target [4]. Some researchers hoped to establish a unified potential function to solve this problem [5]; unfortunately, it still requires regular obstacle to avoid huge calculation requirements, which is nearly intolerable for real-time path planning.

Some researchers have attempted to find the solution in intelligent optimization algorithms, such as genetic algorithm (GA) [6, 7] and ant colony algorithm (ACA) [8, 9]. The former has good global searching maneuverability and can quickly find all of the solutions without falling into local optimal [7]. The latter is highly robust and good at searching for better solution [8], and their experimental results present the feasibility to solve this problem. However, these algorithms have common limits that are difficult to avoid in solving large-scale UAV path planning problems. For example, genetic algorithm is weak for local search with low efficiency in the later periods and easily reaches premature convergence [6]. Ant colony algorithm is sensitive to the initial parameters. An inappropriate setting decreases the search rate and yields poor results [9]. Furthermore, the real-time path planning of UAVs requires high efficiency and accuracy; these algorithms may not be proper solutions.

Graph search algorithms have been developed to find the optimal trajectory between two nodes on connected graphs. The greatest advantage of these algorithms, including Dijkstra, Bellman-Ford, and A^* algorithms, is their straightforward implementation and low computational cost to get the optimal path, which makes it the most suitable method in theory [1]. The A^* solver is one of the most widely used

algorithms among them. It was developed to analyze more effectively the domain in order to avoid distributed obstacles, which is largely applicable to robotics, space exploration, and video games [10]. Though maturely used in 2D graph searching, the A^* algorithm still faces some challenges in the UAV path planning in a 3D complex battlefield environment [11, 12], and previous research results may have problems as follows: (1) most experimental space was simple artificial topographies. The obstacles were few and regular, and complex threats such as radars and extreme weather conditions were not considered [13–15]; therefore, they did not verify their feasibility, efficiency, and convergence in large-scale complex space. (2) Some path planning is essentially performed in a 2D plane [16–18]. Some researchers use 2D spatial partitioning methods, like Voronoi map, to divide the target space into several sections to construct connected network graph. (3) The algorithms did not obtain a flyable path; most searching results are break lines or smoothing curves [14–17]. The maneuverability of the UAV, including the turning radius, safety altitude, climb rate, minimum step size, and fuel consumption, should be considered to get a flyable path, which is a fundamental difference from general robot path planning.

In light of the limitations above, the paper proposed an improved A^* algorithm for the UAV path planning in a 3D large-scale battlefield. This space is composed of real terrain data, radars, no-fly zones, and extreme weather conditions. The algorithm considers terrain following, threat avoidance, and fuel consumption to obtain a flyable path that follows the maneuverability constraints of a UAV. Meanwhile, an assessment algorithm was designed to evaluate the final path. Finally, the algorithm was compared with ACA and GA to present its advantage.

2. Materials and Methods

2.1. Modeling of the Battlefield Environment. The target space in this paper consists of a DEM map and battlefield plot information. Because the modeling methods for the former are very mature, this paper focuses on modeling the plot information, which includes radars, no-fly zones, and extreme weather.

Radars have been widely used to detect the UAVs in modern air defense system. This paper does not treat radar like a normal obstacle but build a detection probability model, which distinguishes this research from most previous studies [19]. The process for computing the detection probability at any point in the target space is essentially the inverse of the radar equation and pattern function [20]. This value is calculated as follows.

(1) From the target point and radar antenna's location, calculate the pitch angle, θ, and use the antenna pattern formula (1) to calculate the pattern function value $F(\theta)$ as follows:

$$F(\theta) = e^{-2.78\theta^2/\theta_B^2}. \tag{1}$$

(2) Calculate the range from the target point to the radar center $R(\theta)$, and determine the maximum radar detection range $R_{\max}$. Consider

$$R(\theta) = R_{\max} * \sqrt{F(\theta)}. \tag{2}$$

(3) Invert the radar equation and the minimum SNR formula to determine the radar detection probability. The radar equation is shown in

$$R_{\max} = \sqrt[4]{\frac{P_{\text{av}} G^2 T_m \lambda^2 \sigma}{(4\pi)^3 k T_0 F_n t B [S/N]_{\min} L}}, \tag{3}$$

where P_{av} is the average transmitted power, G is the antenna gain, λ is the wavelength, σ is the radar cross section, k is the Boltzmann constant, T_m is the pulse recurrence period, t is the pulse length, T_0 is the absolute temperature in K, F_n is the noise figure of the receiver, B is the noise bandwidth of the receiver, L is the system loss factor, and $[S/N]_{\min}$ is the minimum SNR based on single pulse detection, which is defined as follows:

$$\begin{aligned} \left[\frac{S}{N}\right]_{\min} &= A + 0.12AB + 1.7B, \\ A &= \ln\left[\frac{0.62}{P_f}\right], \\ B &= \ln\left[\frac{P_d}{1 - P_d}\right], \end{aligned} \tag{4}$$

where P_d is the detection probability, and P_f is the false alarm probability. We can obtain the detection probability P_d from formulas (3) and (4).

(4) Because modern radar normally adopts the standard of "M detections in N scans" [20], the detection probability, P, of any point must satisfy the following formula:

$$p_d = \sum_{M}^{N} \frac{N!}{K!\,(N-K)!} p^K (1-p)^{N-K}. \tag{5}$$

This formula is nonlinear, so we can use the more efficient Newton iteration method [21] to determine the radar detection probability at any point in the target space.

Furthermore, we should also consider the influence of the terrain on the signal propagation; see Figure 1. Point O is the center of the radar antenna. If AB is the first mountain section, the area below BC is a blind zone because the beam is blocked by AB, and the final detection region in this section is the blue area between OABCDO. Figure 2 shows the radar coverage area ignoring the effect of terrain, while Figure 3 shows the real detection area. The real detection probability at any point is given by the following:

$$p = \begin{cases} \text{Newton}\left(p_d(x, y, z)\right), & (x, y, z) \in Z_R, \\ 0, & (x, y, z) \notin Z_R, \end{cases} \tag{6}$$

where Z_R is the radar coverage and Newton is the Newton iteration method.

Extreme weather, including thunderstorms, blizzards, hails, hazes, and strong airstreams, poses a significant threat to the flight of UAVs. The climate threat model is divided into 2D and 3D zones based on its sphere of action. The former are described using a closed curve $\{P_i(x, y)\}$, where x and y represent the longitude and latitude, respectively,

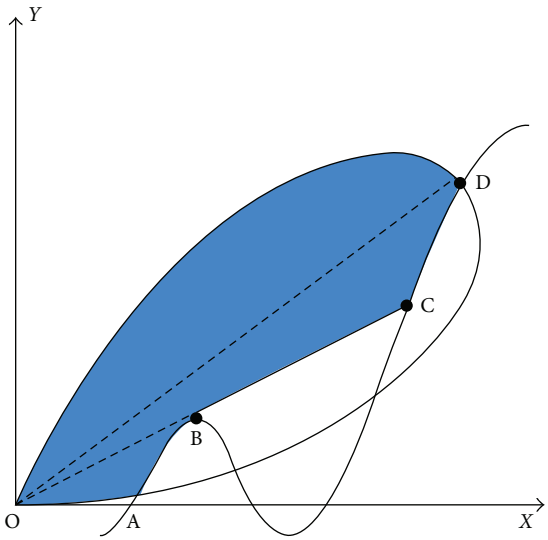


Figure 1: Radar signal propagation graph.

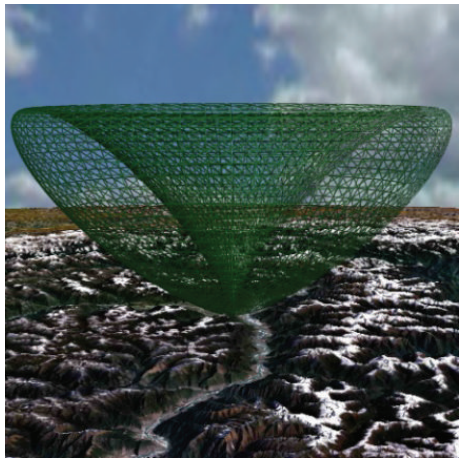


Figure 2: Sector radar coverage.

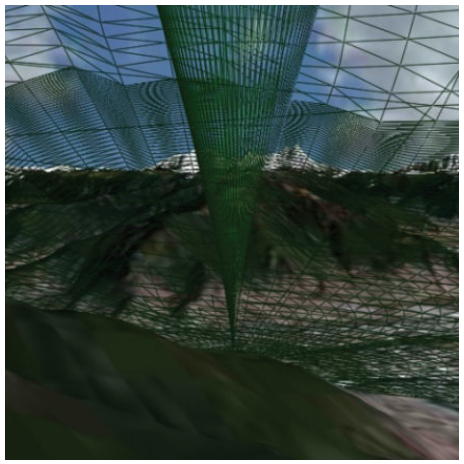


Figure 3: Terrain influence on radar coverage.

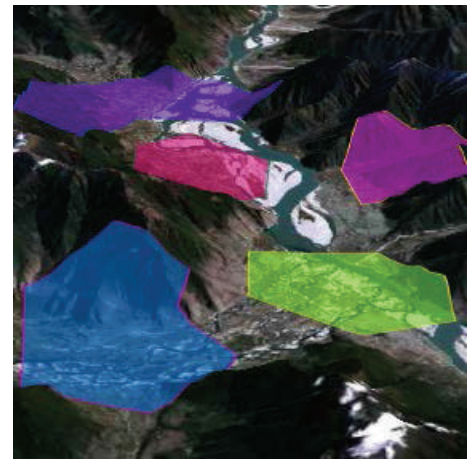


Figure 4: 2D climate threat model.

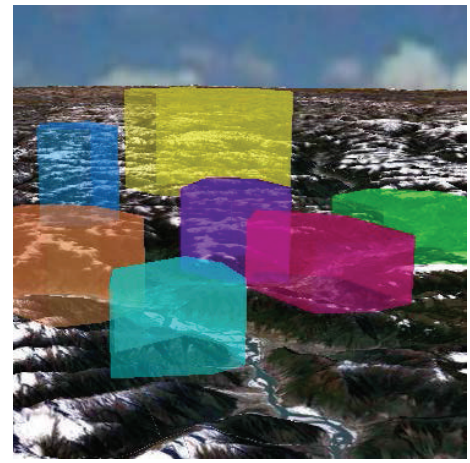


Figure 5: 3D climate threat model.

and are shown in Figure 4. The latter are described using polygonal prisms $(\{P_i(x, y)\}, H_{\min}, H_{\max})$. $H_{\min}$ and $H_{\max}$ are the altitude limits, respectively, and are shown in Figure 5. No-fly zones are described similarly to the 2D climate model.

2.2. Path Planning Strategies

2.2.1. Target Space Partition. The 3D area described by the DEM and DOM is a continuous raster space; therefore, there is no node-edge graph network present in traditional path searching, and spatial partitioning can solve this problem.

Some researchers used a Voronoi diagram to partition the space at a certain altitude [22]. Because UAVs change altitude during flight, this method is only applicable to certain flight missions in flat areas with small section changes at different altitudes. However, for mountainous terrain, this method cannot obtain a valuable optimal path. In addition, some researchers tried to use regular 3D boxes for the space partitioning [14]. It means that there are at least 26 nodes to expand during the searching process, which requires a large amount of time and memory to obtain an optimal solution, especially, in large space. This paper recommends a 2.5D

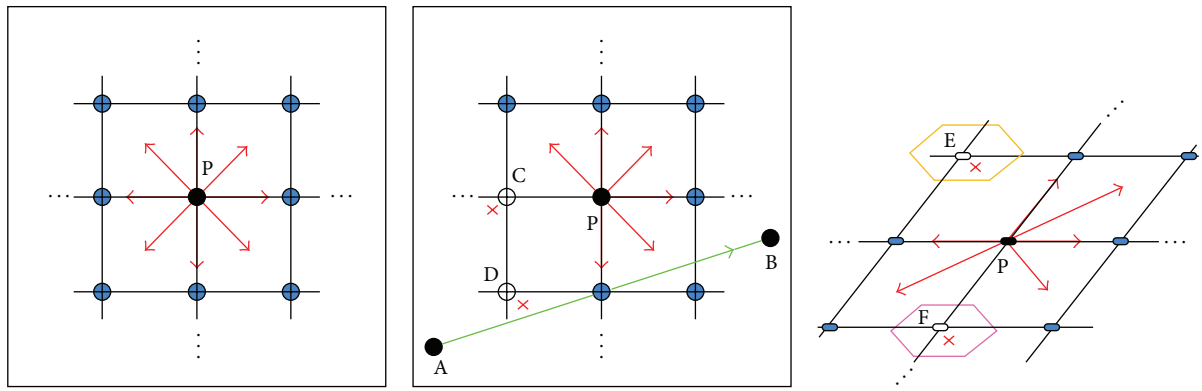


Figure 6: Space partition model.

partition model to solve this problem. The model is actually a surface partition model, which logically divides the DEM data into a series of independent rectangular grids in some accuracy. The terrain information in a grid is represented by its center. Each center point records the longitude, latitude, and altitude, as shown in Figure 6(a). The centers form a routing network needed in the routing process. This model has only 8 directions to expand for each node, which makes it more efficient than the 3D grid partition method.

To further improve the convergence rate and efficiency of the algorithm, some researchers recommend considering the maneuverability constraints of UAVs [14, 17], including the minimum step size and turning radius, in nodes expansion. Satisfying the minimum step size requirement means increase of the grid size, which does improve the search speed to some degree, but it causes an excessive loss of terrain details and may greatly decrease the accuracy of the searching result. Considering the limits to the UAV's speed, slope, and turning angle may decrease expansion directions, but some global optimal nodes may be missed due to the fact that the grid points are not the final waypoints. However, this method can be applied in local searching to filter out directions that the angle to the global direction is too large (e.g., >150°) as shown in Figure 6(b). In addition, this method can also filter out nodes in the no-fly zone and extreme climate threat as shown in Figure 6(c).

2.2.2. Initial Path Based on the Improved A^* Algorithm. The A^* algorithm cost function should be defined to evaluate the cost of the expansion nodes and obtain the node-edge list of the optimal path. The function is as follows:

$$f(n) = g(n) + h(n), \tag{7}$$

where n is the node being expanded, $g(n)$ is the actual cost to from the initial node to node n, and $h(n)$ is the estimated cost from node n to the target node. The heuristic function $g(n)$ is the main factor affecting the search result, and a reference formula was made [23] to plan a terrain following path. Consider

$$J = \int_0^{t_f} \left(\omega_1 c_t^2 + \omega_2 h^2 + \omega_3 f_{\mathrm{TA}}\right) dt, \tag{8}$$

where c_t is the distance from a specified route, h is the altitude, f_{TA} is the threat value, and ω_1, ω_2, and ω_3 are their weights, respectively.

This heuristic function may have some problems when applied to the subject of this paper. First, c_t is a relative distance, but h is an absolute altitude, and they have differing magnitudes. The distance to the center of the threat, f_{TA}, is positively correlated with the cost function, and the lower this value is the better it is. However, c_t and h run counter to this value, which makes it not easy to assign a reasonable weight to each factor. Second, the threat of radar should be measured as a probability rather than a relative distance. Thus, it is several orders of magnitude lower than the other factors and the function would be too insensitive to ω_1, ω_2, and ω_3, which makes finding three values to develop a meaningful cost function for determining the optimal path difficult.

To solve the above problems, the cost function is improved as follows:

$$J(n) = \sum_1^n \left(\omega_1 c_i + \omega_2 p_i + \omega_3 h_i\right). \tag{9}$$

In this formula, c_i is the surface distance from node i to node $i-1$, which is the penalty for the route length. p_i is the detection probability between nodes i and $i-1$ and is aimed at increasing the survival rate of the UAV. h_i is the weighted average altitude between nodes i and $i-1$.

These three cost function factors all push in the same direction; that is, the solution requires a reduced route length, altitude, and detection probability. However, these factors are not on the same magnitude, and the penalty factors should be normalized. The key to adopting a 0-1 scale is to determine the maximum and minimum values for each factor. We cannot directly determine the maximum and minimum distance because it varies for each route segment

due to the salient relief; however, regardless of the real or artificial terrain data, the maximum and minimum altitude is either already known or can be calculated from the data. Once $h_{\max}$ and $h_{\min}$ are determined, we can find the max and min of c_i; see as shown in Figure 7. Consider

$$c_{\max} = \sqrt{\Delta d^2 + (h_{\max} - h_{\min})^2} * (n_i - 1), \quad c_{\min} = \sqrt{d_i^2 - \Delta h_i^2}, \tag{10}$$

where $n_i = \lceil \sqrt{d_i^2 - \Delta h_i^2}/\Delta d \rceil$ is the number of interpolation points between nodes i and $i-1$ and d_i, Δh_i, and Δd are the Euclidean distance, altitude difference, and sampling interval, respectively.

p_i is the radar detection probability between nodes i and $i-1$ and should be considered as the sum of all radars in the space. The corresponding formula is shown as follows:

$$p_i = \sum_{i}^{m} p(R_i) - \sum_{1 \le i \le j \le m}^{m} p(R_i R_j) + \sum_{1 \le i \le j \le k \le m}^{m} p(R_i R_j R_k) - \cdots + (-1)^m p(R_1 R_2 \cdots R_m), \tag{11}$$

where $p(R_j)$ is the weighted average probability of detection by the jth radar between nodes i and $i-1$ for n_i interpolation points; the detection probability of each interpolation point can be calculated from formula (6). The normalization of these three factors is shown as follows:

$$C_i = \frac{(c_i - c_{\min})}{(c_{\max} - c_{\min})}, \quad H_i = \frac{(h_i - h_{\min})}{(h_{\max} - h_{\min})}, \quad P_i = p_i. \tag{12}$$

Formula (13) is the new cost function using these normalizations. In this formula, $\omega_1 + \omega_2 + \omega_3 = 1$. Because the route length, altitude, and detection probability were normalized, the user can more directly assign the respective weights according to the planning objective. Consider

$$g(n) = \sum_{1}^{n} (\omega_1 C_i + \omega_2 H_i + \omega_3 P_i). \tag{13}$$

In formula (7), $h(n)$ was defined as the Euclidean distance from node n to the target point, which helps node n to approach the target point. Consider

$$h(n) = \frac{(d_n - d_{\min})}{(d_{\max} - d_{\min})}. \tag{14}$$

In this formula, d_n, $d_{\max}$, and $d_{\min}$ are the Euclidean, maximum, and minimum distances from node n to the target point as calculated using a similar formula to (10). However, it is worth mentioning that the sampling interval, ΔD, is generally greater than Δd to increase the efficiency of $h(n)$.

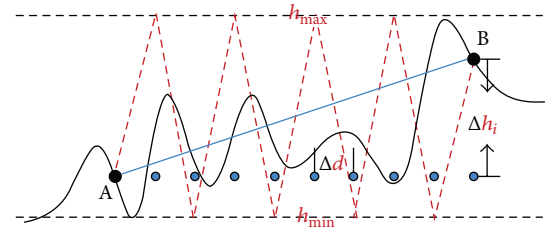


Figure 7: Computation of $c_{\max}$ and $c_{\min}$.

Once the target space and cost function are set, the standard A^* path searching process can begin. Open and closed lists were defined to store the expanded nodes, which were implemented with the minimum binary heap and linear lists to increase the efficiency.

Furthermore, to compare the searching result in different parameters, formula (15) is designed to evaluate the final path as follows:

$$V = \varepsilon_1 \frac{(C - C_{\min})}{(C_{\max} - C_{\min})} + \varepsilon_2 \frac{(H - H_{\min})}{(H_{\max} - H_{\min})} + \varepsilon_3 P, \tag{15}$$

where C is the route length, H is the mean altitude, and P is the detection probability. $C_{\max}$ and $H_{\max}$ are the maximum of C and H, respectively, and $C_{\min}$ and $H_{\min}$ are the minimum. ε_1, ε_2, and ε_3 are the weights of C, H, and P. Similar to the formula of $g(n)$, a smaller value of V indicates a better path.

2.2.3. The Path Optimization Algorithms. The search results from Section 2.2.2 may not satisfy the UAV maneuverability constraints, such as minimum step size, turning radius, turning angle, climb rate, and safety altitude. Therefore, a series of algorithms are developed to obtain the final flyable path.

The minimum step size and turning radius are constraints on the top view. The former requires the length of each route segment to be above a certain value $S_{\min}$, while the latter requires every segment to be long enough for two continuous turns; therefore, we have to compress the initial route. To satisfy the two constraints, in this paper, the FFP algorithm is used for the data compression [24], and the longest rectilinear trend can be furthest preserved to avoid frequent turning of the UAV. In addition, the turning angle constraint requires every segment angle to be below a critical value, $C_{\min}$. A potential field algorithm can be developed to solve this problem [5]. As UAVs cannot fly along a broken line, some researchers recommend various curve smoothing methods such as the B-spline curve [25]. However, these methods obviously conflict with the minimum step size constraint. In fact, UAVs generally turn by escribing or flyby [26].

The climb rate and safety altitude are constraints on the vertical view. UAVs tend to maintain posture without frequent turns or climbs. If the altitude of two waypoints, A and B, is different, the UAV does not fly directly from A to B but first climbs to O from A at a certain climb rate before flying horizontally to B as shown in Figure 8. The safety altitude is the minimum distance from the ground to ensure safe flight. Because of various factors, like strong turbulence and automatic-control error, a UAV usually deviates from the

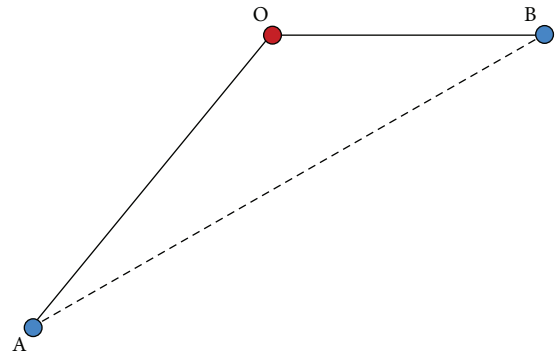


Figure 8: UAV climbing model.

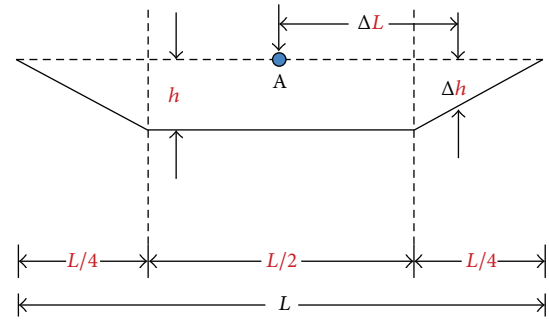


Figure 9: Route safety altitude model.

predefined path. Therefore, the algorithm should consider the safety altitude across the entire deviation area. This area is the safety corridor of the path, and its mathematical model is shown in Figure 9 [26]. This corridor is divided into the primary and secondary areas with the former providing the entire safety altitude h and the latter providing safety between 0 and h. The safety altitude, Δh, at any point is calculated as follows:

$$\Delta h = h * \left(1 - 2 * \frac{\Delta L}{L}\right), \tag{16}$$

where h is the safety altitude, ΔL is the distance between the target point and the predefined path, and L is the width of the safety corridor.

Based on the mathematic models above, we propose the following algorithm to meet the safety altitude constraint between waypoint A and waypoint B.

(1) Divide AB into AO and OB, and O is the level-flight point calculated from the climb rate at A.

(2) Calculate the minimum signed altitude difference, ΔH_1, for the OB segment. Begin the spatial interpolation for OB to obtain a series of interpolated points, $\{P_i\}$. Divide the cross section of P_i into a primary area and two secondary areas, and calculate the corresponding minimum signed heights. Consider

$$(\Delta H_1)_i = \min\left(\min\left(h_B - \Delta h - h_j\right), \min\left(h_B - h - h_k\right)\right), \tag{17}$$

where $(\Delta H_1)_i$ is the minimum signed height difference for cross section P_i, h_B is the altitude of the OB segment, h is the entire safety altitude, Δh is the safety altitude of the secondary areas as calculated from formula (16), h_j is the altitude of the interpolation point j in the primary area, and h_k is the altitude of the interpolation point k in the secondary area. The minimum $(\Delta H_1)_i$ value can be obtained by traversing $\{P_i\}$.

(3) Calculate the minimum signed height difference, ΔH_2, of the AO segment. First, get the interpolation points, $\{Q_i\}$, between AO by spatial interpolation, and the minimum signed height difference for cross section i of Q_i is calculated by

$$(\Delta H_2)_i = \min\left(\min\left(h_i - \Delta h - h_j\right), \min\left(h_i - h - h_k\right)\right), \tag{18}$$

where $h_i = h_A + AO * (h_O - h_A)/AQ_i$, and ΔH_2 can be calculated by traversing $\{Q_i\}$.

(4) Adjust the altitude of O and B to $h_B + \Delta H_1$ and the altitude of A to $h_A + \Delta H_2$. Create a radial through A using the climb rate and calculate the intersection with OB, which is the new level-flight point O.

3. Results and Discussion

The test space was about 2,500,000 Km^2 between Linzhi in Tibet and Chengdu in Sichuan province and was constructed with SRTM terrain and LANDSAT image data in 30 m accuracy. The area is a mountainous district that can verify the feasibility and adaptability of the algorithms in this paper. The algorithms were developed with C# and Direct 3D, and they have been successfully used for a 3DGIS platform, Gaea Explorer [27]. The hardware environment was as follows: CPU: Intel Core2 Duo E8200, memory: Kinston 1 G, video card: ATI Radeon HD 2600 XT.

To prove the feasibility of the algorithms in this paper, we developed the parameters in Table 1. Before searching, we defined the UAV parameters in Table 2.

In Figure 10, (a) shows the initial search result, (b) shows the compression results, (c) shows the smoothing results, (d) shows the results that consider the climb rate, (e) shows the safety corridor result, (f) shows the results that consider the safety altitude, and both (g) and (h) show the local features of the safety corridor from the top and front views, respectively. To prove the safety of the final path, we calculated the profile at $\pm L/4$ and $\pm L/2$, as shown in Figure 11. From these results, we determined that the algorithms were effective for low-altitude UAV path planning.

In Table 3, four experimental groups were designed to show the adaptability, convergence, and efficiency of the algorithm at various distances and grid sizes. Table 4 shows the parameters to evaluate the final path, where H_{aver} is the average altitude of the starting point and target point.

Table 1: The algorithm parameters.

Starting point (°, °, M)	Target point (°, °, M)	Surface distance (KM)	Grid size (°)	$\omega_1 : \omega_2 : \omega_3$
93.832779	94.372746			
29.127954	29.31984	59.783	0.01	1 : 2 : 1
2949.2	2923.6			

Table 2: UAV maneuverability parameters.

Speed (KM/H)	Slope (°)	Climb rate (M/S)	Turning method	Safety altitude (M)	Safety corridor width (NM)
300	20	120	Escribing	300	2

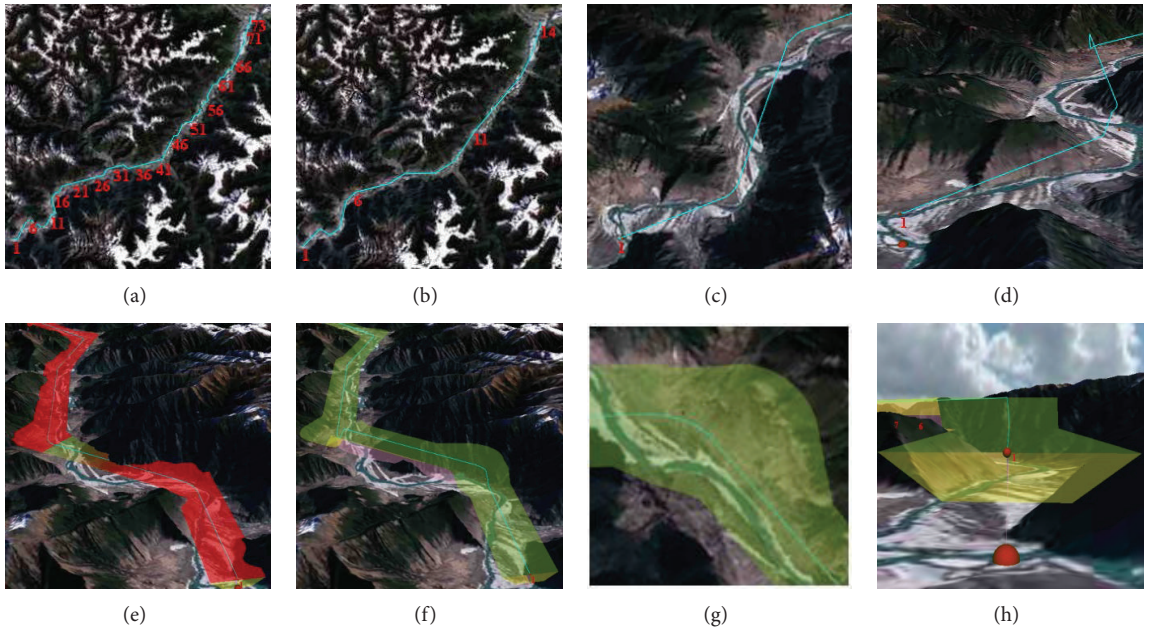


Figure 10: Algorithms results.

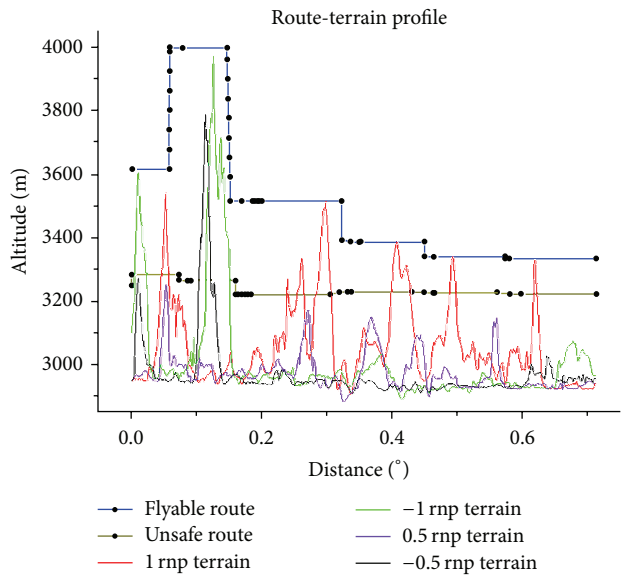


Figure 11: Route-terrain profile.

Table 3: Four control groups.

Starting point (°, °, M)	Target point (°, °, M)	Surface distance (KM)	Grid size (°)	$\omega_1 : \omega_2 : \omega_3$
93.832779	94.372746	59.783	0.005	1 : 2 : 1
29.127954	29.31984		0.01	
2949.2	2923.6		0.02	
90.739257	94.835207	419.073	0.005	
29.292307	29.506482		0.01	
3606.3	2915.5		0.02	
87.631027	94.816805	729.349	0.005	
29.130563	29.463092		0.01	
3985.1	2914.9		0.02	
93.562434	105.587785	1236.916	0.005	
29.168065	30.119375		0.01	
2976.5	399.4		0.02	

Table 4: Assessment parameters.

Minimum length (KM)	Maximum length (KM)	Minimum altitude (M)	Maximum altitude (M)	$\varepsilon_1 : \varepsilon_2 : \varepsilon_3$
Surface distance	2 ∗ surface distance	H_{aver}	4500	1 : 1 : 1

Table 5 shows that the algorithm can quickly determine the optimal path for various distances and grid sizes, which means that the algorithm is stable, convergent, and efficient. The time cost is proportional to the distance and inversely proportional to the grid size. Furthermore, the assessment value is smaller when the space partition grid is 0.01° rather than 0.005° or 0.02°.

To present the influence on the searching result of the changes of $\omega_1 : \omega_2 : \omega_3$, the four groups in Table 4 were compared at the ratio of the weight factors ω_1, ω_2, and ω_3 being equal to 2 : 1 : 1, 1 : 2 : 1, or 1 : 1 : 1. The result in Table 6 shows that the time cost is proportional to the weight of the route length and inversely proportional to the weight of the route altitude. A higher weight of the length results in faster approach to the target point, but the UAVs would be more likely to be detected by enemies because higher nodes may be chosen in nodes extension. Considering the survival rate of the UAVs, a higher weight of the altitude may be more reasonable, and it was seemingly backed up by Table 6, which shows that the assessment value is the smallest when the weight ratio is 1 : 2 : 1 rather than 1 : 1 : 1 or 2 : 1 : 1.

Figures 12(a)–12(d) show the results of the four comparison groups at a grid size of 0.01° and a weight ratio of 1 : 2 : 1.

We created a battlefield environment of approximately 2,500,000 Km^2, as shown in Figure 13, to demonstrate the algorithm's adaptability and convergence for large-scale complex situations including no-fly zones, extreme weather, and radars.

Table 7 shows the search results for the four groups at a grid size of 0.01°. The results prove that the algorithm is stable and convergent even for a complex battlefield environment. By comparing with the results shown in Table 4, we can see that the search time and route altitude increased slightly. Figure 14(a) shows the searching results for the first group; Figure 14(b) shows the local features. The other images are the results for the other three groups. These results indicate that the algorithm can effectively avoid no-fly zones and extreme weather while automatically searching for blind zones in radars network to accomplish a low-altitude penetration.

To present the advantages of the algorithm in solving the problem, we compared it with the ACA and the GA using the same parameters as in Table 3. The grid size is 0.001, and the iteration of ACS and GA is 2000.

The comparison results in Table 8 show that our algorithm has greater advantage in time efficiency than the other two algorithms, especially, in large scales, which is quite important in real-time path planning. On the surface, the advantage in the searching result is not so obvious, but the stability and convergence to get the optimal path still have comparative preponderance.

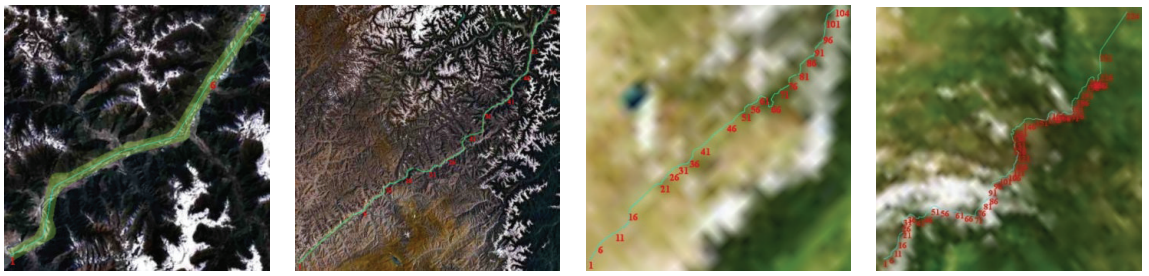
4. Conclusions

An improved A^* algorithm was developed for the path planning in large-scale 3D battlefields. The paper recommended a 2.5D spatial partition method for the 3D raster space, proposed a probability calculation model for radars network, and improved the A^* cost function to get an optimum route by considering the route length, the altitude, and the detection probability. A series of path optimization algorithms were developed to follow the maneuverability constraints of UAVs to obtain a final flyable path, and an assessment algorithm was designed to evaluate the path. The experimental results show that the improved A^* algorithm is stable, convergent, and efficient, and the comparison with ACA and GA shows its great advantage.

However, some issues should be further discussed. First, it is not so clear to quantify the planning goal to the optimal parameters, which is the key focus of decision makers. It was indicated that a comparatively good result was obtained at a grid size of 0.01° and a weight ratio of 1 : 2 : 1, but it is not easy to determine the exact optimal value of them. Table 6 presents

Table 5: The comparison results.

Assessment value	Route length (KM)	Detection probability	Mean altitude (M)	Total calculation time (S)		
				Initial path	Compress and smoothen	Altitude adjusting
0.133043	60.4	0	3542.2	0.012	0.169 0.121	0.036
0.129928	60.611	0	3528.9	0.004	0.129 0.086	0.039
0.207789	59.932	0	3907.2	0.002	0.104 0.065	0.037
0.346597	487.291	0	4347.6	3.19	4.462 0.941	0.331
0.207219	441.867	0	3963.8	0.271	1.51 0.952	0.287
0.395662	427.289	0	4707.4	0.086	1.138 0.774	0.278
0.450168	853.568	0	4689.2	9.916	13.063 2.693	0.454
0.273318	805.585	0	4201.2	0.543	3.268 2.241	0.484
0.328756	778.665	0	4414.6	0.068	2.202 1.684	0.45
0.323873	1591.054	0	3615.2	44.584	52.065 6.538	0.788
0.27795	1521.588	0	3385.6	3.63	11.488 6.915	0.943
0.294937	1399.922	0	3808.5	0.79	7.251 5.763	0.698



(a) (b) (c) (d)

Figure 12: Algorithms results at grid size of 0.01°.

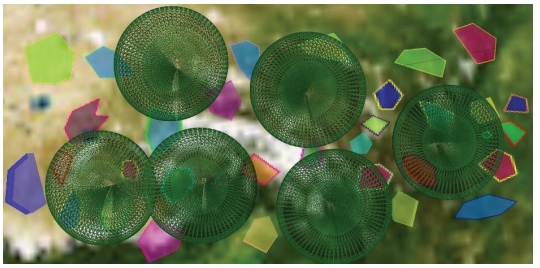
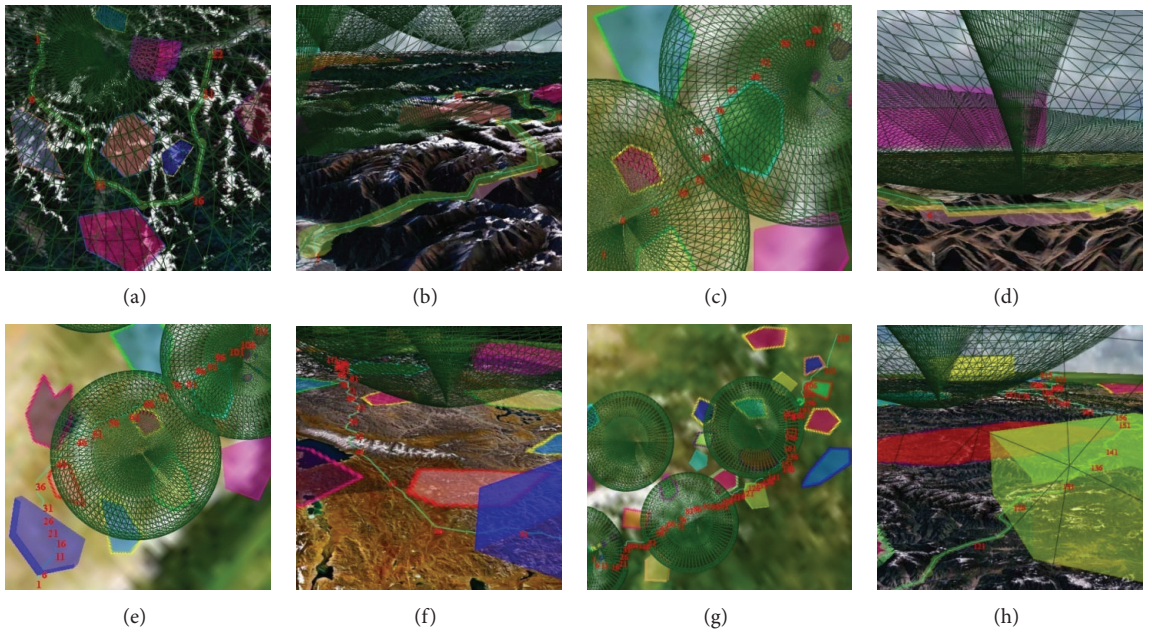


Figure 13: 3D battlefield environment.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 14: Algorithms results for the 3D battlefield environment.

Table 6: The comparison of different weights.

Assessment value	Route length (KM)	Detection probability	Mean altitude (M)	Total calculation time (S)		
				Initial path	Compress and smoothen	Altitude adjusting
0.189288	58.95	0	3846.1	0.004	0.132 0.091	0.037
0.130927	60.611	0	3528.9	0.004	0.129 0.086	0.039
0.27982	59.201	0	4264.2	0.003	0.135 0.084	0.042
0.379233	413.297	0	4687.7	0.103	1.121 0.764	0.254
0.207394	441.867	0	3963.8	0.271	1.51 0.952	0.287
0.436574	423.076	0	4872.4	0.088	1.364 0.983	0.293
0.413541	751.683	0	4720.5	0.261	2.316 1.643	0.412
0.273318	805.585	0	4201.2	0.543	3.268 2.241	0.484
0.482125	769.564	0	4910.8	0.179	2.58 1.875	0.526
0.285577	1342.626	0	3856.8	1.327	6.137 4.112	0.698
0.27795	1521.588	0	3385.6	3.63	11.488 6.915	0.943
0.310799	1398.655	0	3942.2	1.158	7.743 5.371	1.214

Table 7: The results in the battlefield environment.

Waypoints number	Route length (KM)	Detection probability	Mean altitude (M)	Total calculation time (S)		
				Initial path	Compress and smoothen	Altitude adjusting
22	143.436	0.0019	3953.1	0.093	0.483 0.308	0.082
71	513.648	0.0023	4655.4	0.77	2.361 1.264	0.327
111	865.65	0.0017	5133.7	2.732	6.048 2.871	0.445
215	1592.591	0.0011	3750.4	11.998	19.146 6.127	1.021

Table 8: Comparison results of different algorithms.

Algorithm type	Assessment value	Route length (KM)	Detection probability	Mean altitude (M)	Total calculation time (S)		
					Initial path	Compress and smoothen	Altitude adjusting
Improved A^*	0.130927	60.611	0	3528.9	0.004	0.129 0.086	0.039
ACS	0.133615	61.678	0	3513.6	110.265	90.715 110.265	110.265
GA	0.134115	59.898	0	3562.5	110.142	110.265 0.092	0.031
Improved A^*	0.207394	441.867	0	3963.8	0.271	1.51 0.952	0.287
ACS	0.231413	446.073	0	4041.3	1394.478	1396.646 1.957	0.211
GA	0.223132	450.576	0	3997.2	1766.738	1769.705 2.732	0.235
Improved A^*	0.273318	805.585	0	4201.2	0.543	3.268 2.241	0.484
ACS	0.27901	793.658	0	4236.3	3643.819	3652.089 7.717	0.553
GA	0.285504	799.532	0	4248.3	2931.554	2944.994 12.793	0.647
Improved A^*	0.27795	1521.588	0	3385.6	3.63	11.488 6.915	0.943
ACS	0.280544	1496.949	0	3463.5	10848.625	11541.301 31.539	1.137
GA	0.297615	1588.795	0	3398.7	7570.303	7645.141 73.15	1.688

that the path is better when the value of $\omega_1 : \omega_2 : \omega_3$ is 1 : 2 : 1 rather than 1 : 1 : 1 or 2 : 1 : 1, but the assessment still depends on the parameters in Table 4.

Second, since the final path cannot be directly obtained by the A^* process as mentioned in Section 2.2.1., the searching result after the optimization process may no longer be the optimal solution. Although we adjusted the algorithms to locally optimize the route segments, globally optimizing the entire path is still to be solved.

Finally, the comparison of the improved A^* algorithm with ACA and GA shows its great superiority in time efficiency, but the advantage is not so obvious in the planning result, and more iterations of ACA or GA may even obtain better result than A^* algorithm. However, considering the

internal limitations of the latter, like their stability (sensitive to initial parameters), convergence, and efficiency, they need to be further studied to be used in the UAV path planning.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by Grants from the National Basic Research Program of China (973 Program) (no. 2011CB707101), the National High Technology Research and Development Program of China (863 Program) (nos. 2011AA010500 and 2013AA01A608), the National Key Projects (2013ZX07503-001), and the National Nature Science Foundation of China (NSFC) Program (no. 2010CDB08403). The authors appreciate the support and help from both the Air Defense Commanding Troops and their research centers.

References

[1] B. M. Sathyaraj, L. C. Jain, A. Finn, and S. Drake, "Multiple UAVs path planning algorithms: a comparative study," *Fuzzy Optimization and Decision Making*, vol. 7, no. 3, pp. 257–267, 2008.

[2] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Autonomous Robots*, vol. 13, no. 3, pp. 207–222, 2002.

[3] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1398–1404, Sacramento, Calif, USA, April 1991.

[4] D. P. Horner and A. J. Healey, "Use of artificial potential fields for UAV guidance and optimization of WLAN communications," in *Proceedings of the IEEE/OES Autonomous Underwater Vehicles*, pp. 88–95, June 2004.

[5] J. Ruchti, R. Senkbeil, J. Carroll, J. Dickinson, J. Holt, and S. Biaz, "UAV collision avoidance using artificial potential fields," Tech. Rep. #CSSE11-03, 2011.

[6] F. C. J. Allaire, M. Tarbouchi, G. Labonté, and G. Fusina, "FPGA implementation of genetic algorithm for UAV real-time path planning," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1–3, pp. 495–510, 2009.

[7] D. M. Pierre, N. Zakaria, and A. J. Pal, "Master-slave parallel vector-evaluated genetic algorithm for unmanned aerial vehicle's path planning," in *Proceedings of the 11th International Conference on Hybrid Intelligent Systems (HIS '11)*, pp. 517–521, Malacca, Malaysia, December 2011.

[8] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda, and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Applied Soft Computing*, vol. 9, no. 3, pp. 1102–1110, 2009.

[9] A. Jevtić, D. Andina, A. Jaimes, J. Gomez, and M. Jamshidi, "Unmanned aerial vehicle route optimization using ant system algorithm," in *Proceedings of the 5th International Conference on System of Systems Engineering (SoSE '10)*, pp. 1–6, Loughborough, UK, June 2010.

[10] http://theory.stanford.edu/~amitp/GameProgramming/index.html.

[11] R. Samar and A. Rehman, "Autonomous terrain-following for unmanned air vehicles," *Mechatronics*, vol. 21, no. 5, pp. 844–860, 2011.

[12] J. Benton, S. S. Iyengar, W. Deng, N. Brener, and V. S. Subrahmanian, "Tactical route planning: new algorithms for decomposing the map," in *Proceedings of the 7th International Conference on Tools with Artificial Intelligence*, pp. 268–277, Herndon, Va, USA, November 1995.

[13] R. J. Szczerba, P. Galkowski, I. S. Glickstein, and N. Ternullo, "Robust algorithm for real-time route planning," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 3, pp. 869–878, 2000.

[14] H. Cao, N. E. Brener, and S. S. Iyengar, "3D large grid route planner for the autonomous underwater vehicles," *International Journal of Intelligent Computing and Cybernetics*, vol. 2, no. 3, pp. 455–476, 2009.

[15] M. G. H. Bell, "Hyperstar: a multi-path Astar algorithm for risk averse vehicle navigation," *Transportation Research B*, vol. 43, no. 1, pp. 97–107, 2009.

[16] L. de Filippis, G. Guglieri, and F. Quagliotti, "Path planning strategies for UAVS in 3D environments," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1–4, pp. 247–264, 2012.

[17] M. Jun and R. D'Andrea, "Path planning for unmanned aerial vehicles in uncertain and adversarial environments," in *Cooperative Control: Models, Applications and Algorithms*, vol. 1, pp. 95–110, Springer, New York, NY, USA, 2003.

[18] L. de Filippis, G. Guglieri, and F. Quagliotti, "A minimum risk approach for path planning of UAVs," *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1–4, pp. 203–219, 2011.

[19] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.

[20] M. I. Skolnik, *Radar Handbook*, McGraw-Hill Professional, New York, NY, USA, 1991.

[21] S. Abbasbandy, "Improving Newton-Raphson method for nonlinear equations by modified Adomian decomposition method," *Applied Mathematics and Computation*, vol. 145, no. 2-3, pp. 887–893, 2003.

[22] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV," *Aerospace Science and Technology*, vol. 16, no. 1, pp. 47–55, 2012.

[23] S. J. Asseo, "Terrain following/terrain avoidance path optimization using the method of steepest descent," in *Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON '88)*, vol. 3, pp. 1128–1136, Dayton, Ohio, USA, May 1988.

[24] G. Chen and L. Li, "An optimized algorithm for lossy compression of real-time data," in *Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS '10)*, pp. 187–191, Xiamen, China, October 2010.

[25] K. Yang and S. Sukkarieh, "3D smooth path planning for a UAV in cluttered natural environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08)*, pp. 794–800, Nice, France, September 2008.

[26] U.S. Department of Transportation Federal Aviation Administration, "Standard for required navigation performance (RNP) approach procedures with special aircraft and aircrew authorization required," June 2005.

[27] http://www.gaeaway.com/html.asp?type=20&page=2.