2014 International Conference on Unmanned Aircraft Systems (ICUAS)
May 27-30, 2014. Orlando, FL, USA

Optimal Flight Path Planning for UAVs in 3-D Threat Environment*

Yaohong Qu, Yintao Zhang , and Youmin Zhang, *Senior Member, IEEE*

Abstract—Nowadays, the environments surrounding modern battlefield are becoming increasingly complicated, since the threats are not only from the ground but also from the sky. UAV with reconnaissance mission will take more risk when flying along an improper planned path, so path planning of UAV in complex 3-D environments is very significant and challenging. Aimed at the problem, this paper proposes a novel optimal path planning method for UAV based on the flight space partitioning, Dijkstra algorithm and potential field theory. Specifically, under the cases that the locations of threats are assumed to be known and the whole flight space is partitioned into a number of cells and each cell has a safest node. Then, a 3-D network is formed by connecting the nodes of adjacent cells and a shortest suboptimal path is marked on the network with Dijkstra algorithm. Finally, the optimal path is obtained with artificial potential field method. To verify the proposed algorithm, simulation results in two cases are shown.

I. Introduction

In recent years, unmanned aerial vehicles (UAV) have been efficiently used both in civil aviations and military operations [1]. As adjustment of the planned path during the mission period is very little, the path planning is crucial to improve reconnaissance efficiency [2]. Over decades several path planning methods have been investigated, such as A-star algorithm [3], ant colony algorithm [4], genetic algorithm [5], artificial potential field [6], particle swarm optimization [7], and etc. Among all these methods, most solve the problems in 2-D environment [3-7]. However, faced with the complex flight environment, trajectory planning in 2-D cannot meet the mission requirements of the UAVs. So the path planning algorithms in 3-D space have been paid more attention by many researchers [8-11]. The conventional path planning methods for UAVs in 3-D only take the height information as an additional consideration in the cost function [8], so the optimal trajectory cannot be obtained.

Aimed at the problems above, a path planning algorithm in the 3-D flight environments with multiple threats (such as, missiles radars and terrain) is proposed in this paper, which is combined with 3-D flight space division, Dijkstra algorithm and artificial potential field theory. At first, the locations of threats in the flight environment are regarded as the network nodes. According to 3-D Delaunay rules [13], the entire space of flying environment is divided into several closely connected tetrahedrons with the nodes. Next, we can obtain a 3-D network by crossing lines between two inscribed sphere centers of all the adjacent tetrahedrons. Then the initial shortest route is designed with Dijkstra algorithm on the network. Finally, the optimal path is obtained by using artificial potential field method, which can effectively avoid the threats from radars and meet the requirements of UAV dynamics.

The remaining part of this paper is organized as follows: Section 2 proposes the flight path planning problem for UAVs. And the, mathematic modeling of the flight environment and threats is built in sections 3.1-3.2. Then the path planning algorithms are explained in sections 3.3-3.4. Next, section 4 presents experimental results which validate the effectiveness of the proposed algorithm. Finally, conclusions and further works are outlined in Section 5.

II. Problem Formulation

Nowadays, the battlefield environments are becoming increasingly complicated. With development and application of the high technology, the range of modern battlefield is increasing rapidly and the number of threats in the battlefield is also increasing. Moreover, the threats become diversified. Thus, it makes great sense to study on planning a safe and effective path in complex battlefield.

All the threats which are taken into account in path planning problem can be summarized into natural factors and human factors. The former includes terrain information (such as mountains) and weather information (such as the lightning and rainstorm) and they are no-fly zones (NFZ). The latter includes radars, missiles, aircrafts, and etc. Therefore, UAV will take a greater risk when flying over these areas.

As it is shown in Fig. 1, this is a simulation of a complex battlefield. There are mountains (terrain information) and torrential rain (bad weather information) in the north. Also a number of radars, missiles, and all kinds of threats scatters in various parts of the battlefield. Of course, it is not easy to plan a perfect trajectory through such complex battlefield. As it is known that the path of UAV plays decisive role on the quality of mission and chances of survival. Therefore, by considering terrain, weather, threats, UAV kinetics, flight time and fuel consumption comprehensively, this paper focuses on obtaining an optimal path from starting point *S* to the target

*Research supported by the National Natural Sciences Foundation of China (60974146) and also Natural Sciences and Engineering Research Council of Canada (NSERC).

Yaohong Qu is with the School of Automation, Northwestern Polytechnical University, Xi'an 710072 P. R. China. (e-mail: qyh0809@126.com; phone: 086-88493142; fax: 086-88431302).

Yintao Zhang is with the School of Automation, Northwestern Polytechnical University, Xi'an 710072 P. R. China. (e-mail: zhangyintao23@163.com).

Youmin Zhang is with the Department of Mechanical and Industrial Engineering, Concordia University, Montreal, Quebec H3G 1M8, Canada (e-mail: Youmin.Zhang@concordia.ca).

point *T*. In order to illustrate more clearly the problem, three different planned paths are sketched in the figure.

Path I: The UAV successfully avoids threats, but it cannot fly across by the influence of natural factors.

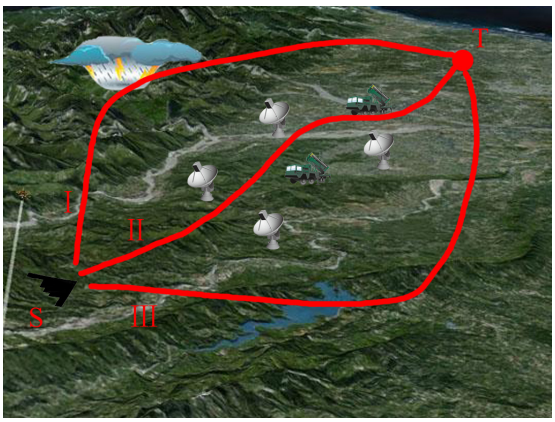

Figure 1. An illustration for battlefield and different planned flight paths

Path II: The UAV reaches the terminal position quickly, but it flies through a very dangerous area.

Path III: The UAV keeps a distance from threats and the route is available, but it takes a long way and fails to complete the reconnaissance missions.

Obviously, it is difficult to find a safe and effective path from *S* to *T* in the battlefield.

III. Algorithm Description

This paper proposes a path planning algorithm containing graph theory, Dijkstra algorithm and potential field thought. In the first place, based on the locations of threats, the infinite space of battlefield is mapped to the finite 3-D Delaunay triangulation which reduces the difficulty of the research. And the 3-D Delaunay triangulation is a collection of tetrahedrons. We can obtain a 3-D network by crossing lines between two inscribed sphere centers of all the adjacent tetrahedrons. Then, the initial path, which is the shortest path on the network, can be obtained with Dijkstra algorithm. But the initial path did not consider UAV dynamics, and it is still unknown whether it can meet the requirements of the UAV flight. In order to improve the path, artificial potential field method is used to make a smooth and feasible path for UAV flight. Finally, the optimal path for UAV is obtained which can effectively avoid the threats from various parts of the battlefield.

3.1. Threat modeling

In modern warfare, almost all anti-air weapons need radar to track and lock the air target. To illuminate the method, without loss of generality, the hostile threats are simply taken as radars. The radar equation is given by

$$P = \frac{FGC\sigma}{(4\pi)^2 R^4} \tag{1}$$

where P is the power received by radar; F is power of the tra nsmitter; G is gain of the antenna; C is effective acreage of the antenna; σ is section acreage of radar; R is the distance from radar to object.

We can define constant Q:

$$Q = \frac{GC\sigma}{(4\pi)^2} \tag{2}$$

Therefore, the threat from radar P is only related with transmitting power of radar F and the distance R from UAV to radar. We can define:

$$P = \frac{F}{R^4} Q \tag{3}$$

Assume the scope of radar covers the entire battlefield and the UAV flies at a constant speed.

3.2. Flight space modelling

Usually, the battlefield surrounding of UAV is very complicated and the exact mathematical model is difficult to obtain. As we all know, among the solutions of 2-D problems about path planning, Voronoi diagram has been widely used [12].

With the locations of threats, a 2-D Delaunay triangulation can be obtained easily [14]. The entire surface is divided into a number of triangles and it is not difficulty to find all circumscribed centers of the triangles. The Voronoi diagram is designed by connecting the circumscribed centers of adjacent triangles and the circumscribed center is far away from threats nearby than any other point in the triangle. Creating the Voronoi diagram partitions the surface into a number of convex polygons cells and each cell has one threat. The Voronoi polygon edges are equivalent to a set of perpendicular bisectors which are equidistant from the closest threats, which means maximizing their distance from the closest threats. The starting and target points are not contained within cells but connected to the nodes forming the cell. After that, the shortest path is searched on the network in Voronoi diagram. Fig. 2 shows a Voronoi diagram created by a number of known threats and the shortest path from starting point *A* to the target *B*. Where the red dotted lines represent the Voronoi network and the blue stars stand for the threats.

Inspired by the Voronoi diagram, this paper attempts to find a 3-D network which is similar to the Voronoi diagram. Considering the formulation of the 3-D network, we use 3-D Delaunay method to partition the space of battlefield because of the following advantages [14]: Uniqueness (there will be only one consequence no matter where we start building from), Regional (adding, deleting or moving a node will only affect the adjacent area), neatest (with the shortest diagonal), and a Shell of convex polyhedron.

Up to now, there are many methods to form the 3-D Delaunay triangulation and this paper selects a fast one, see

[13] for an excellent overview. The 3-D Delaunay method divides the entire space into different tetrahedrons and each tetrahedron is a cell. The vertices of the tetrahedrons in Delaunay triangulation correspond with the threats in the battlefield. We need to find the "safest node" in each cell to form the 3-D network.

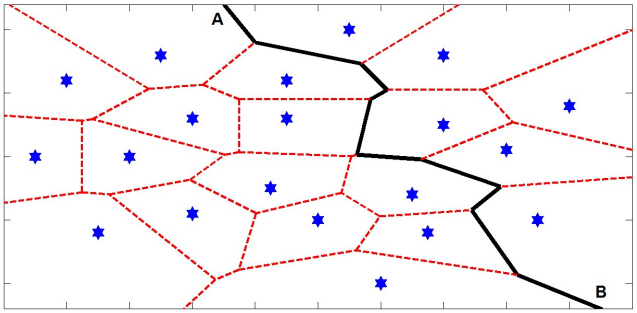

Figure 2. The shortest path in the Voronoi diagram

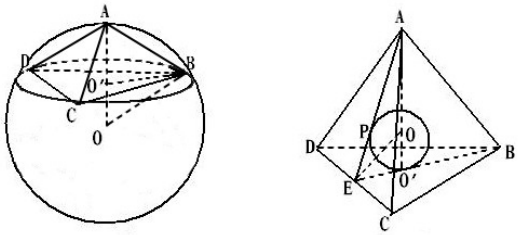

Figure 3. The circumscribed sphere center and the inscribed sphere center

In order to avoid the threats, the node in a tetrahedron should keep a distance from all 4 vertexes as far as possible. In this section, the intensity of threats is not considered; it will be discussed later in the artificial potential section. So the circumscribed sphere center of a tetrahedron seems very appropriate to be the node, because the center is not too close to each vertex. But there comes a problem: the circumscribed sphere center is not always in the tetrahedron, see models in Fig. 3. Thus, the circumscribed sphere center may be very close to other vertex, and then we must consider the threats from another cell. The problem becomes diversified and complex. It makes no senses which have been done to partitioning the space.

Actually, the 4 threats in a tetrahedron can be decomposed in different vectors. We define 4 vectors as perpendiculars of 4 tetrahedron faces, the positive direction is inward. In this way, the threats in a cell are transferred on 4 vectors. To avoid a long vector in the 4 threat vectors, the length of each vector must be very close. Obviously, the inscribed sphere center of a tetrahedron matches all the requirements.

Since the nodes have been found, the following work is to construct the network. If two tetrahedrons are adjacent, the line between their nodes exists (which means UAVs can fly along this line). Find all the adjacent tetrahedrons and connect their nodes. Figs. 4-7 shows how the network is formed. In the simulation, we use balls to represent threats and the radius of balls to represent the intensity of threats. The blue lines are 3-D Delaunay triangulation. The red points are nodes (the inscribed sphere centers of tetrahedrons). The red dotted lines form the 3-D network.

Then the network is obtained, shown in Fig 8. We can search the shortest trajectory from one node to another along the lines in the network.

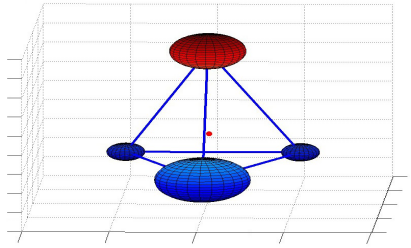

Figure 4. The environment with four threats has one node and zero line

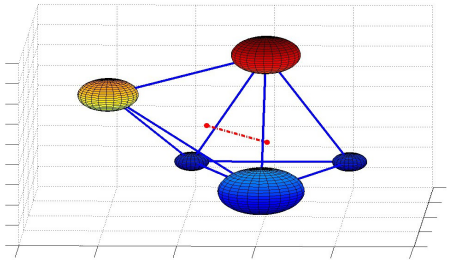

Figure 5. The environment with five threats has two nodes and one line

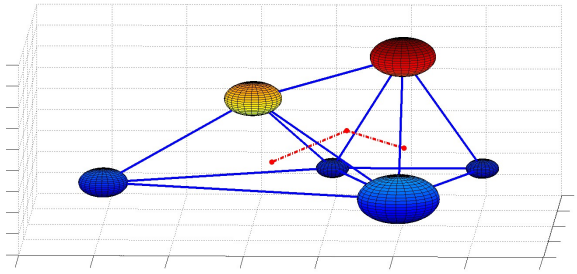

Figure 6. The environment with six threats has three nodes and two lines

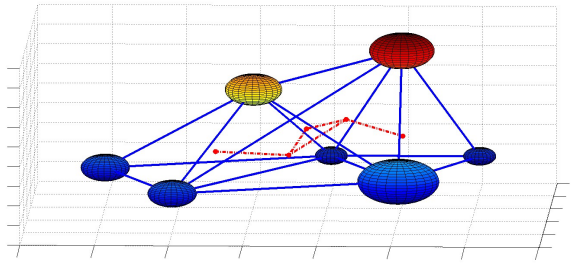

Figure 7. The environment with seven threats has five nodes and five lines

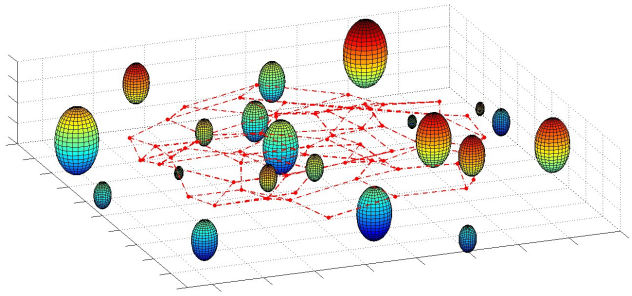

Figure 8. The integral network

Fig. 8 shows the simulation environment with 20 threats and it has 63 nodes and 103 lines. If the locations of threats vary, the environment is changed, and the numbers of nodes and lines are different.

3.3. The Dijkstra algorithm

After the network is formed, we need to find the shortest path in it. This is a typical single-source shortest path problem. The commonly used algorithms are: Dijkstra algorithm, A-star algorithm, Bellman-Ford algorithm, Floyd-Warshall algorithm and Johnson algorithm. In single-source shortest path problem for a graph with nonnegative edge path costs, the Dijkstra algorithm is the most convenient way and reduces the amount of computation among all the algorithms. So we select Dijkstra algorithm to obtain the shortest path from initial node to terminal node on the network. Here, the path cost is equivalent to the path length.

Before introducing the algorithm, the initial node and the terminal node in the network are needed to be determined. Since the starting point and the target point of the path are not in the network, the nearest node from starting point is determined as initial node and the nearest node from target point is determined as terminal node. The specific description is as follows:

All the nodes are numbered from 1 to n. The number of initial node is 1 and the number of terminal node is n. In different environment, the number of nodes is different. For example, the space (shown in Fig. 8) is partitioned into 63 tetrahedrons, which means 63 nodes on the network. Dijkstra algorithm divided the nodes on the network into 3 parts: the unlabeled nodes, the current nodes and the labeled nodes. This paper defines 3 variables to keep them; S is the collection of labeled nodes which have been calculated; T stands for the unlabeled nodes which have not been calculated; *CurNode* presents the nodes which are calculating currently. The collection of all nodes is V, and V is the sum of S, T and *CurNode*. Moreover, we define $L(v,w)$ as the shortest path length from v to w. Obviously, L is a $n\times n$ symmetric matrix. Initially, the main diagonal element is 0 and the rest elements are infinity. Then we find out all the directly connected nodes and keep the length information at the right place in L. The $n\times 1$ matrix P is designed to record the path code where $P(n)$ stands for the shortest path code from initial node 1 to the node n. For example, $P(2)=(1,2)$ means the shortest path from initial node 1 to node 2 is directly the line between them; $P(9)=(1,2,5,9)$ means the shortest path from initial node to node 9 passes node 2 and node 5. After that, the concrete implementation steps are as follows:

Step 1. Initialization: S is empty; *CurNode* only has the initial node and the rest nodes are all in T (including the terminal node).

Step 2. Put the nodes in *CurNode* into S and clear *CurNode*. Find out all the adjacent nodes of the previous nodes in *CurNode* and keep the adjacent ones in *CurNode*. The adjacent nodes refer to 2 nodes which are directly connected to each other on the network. After that, the adjacent nodes are removed from T.

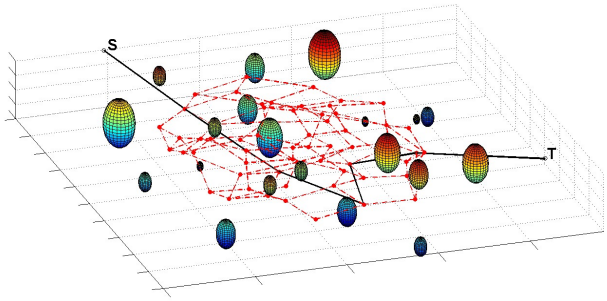

Figure 9. The shortest path on the network

Step 3. For each node in *CurNode*, find out whether the node is in S.

If the node is in S, calculate the length of the new path from initial node to it, then compare the new length and previous length, store the shorter one to L and the corresponding path code to P.

If the node is not in S, calculate the path length and keep it at the right place in L, then store the corresponding path code to P.

Step 4. Repeat the step 2 and step 3 until T is empty.

After all these, the shortest initial path $P(n)$ is obtained (shown in Fig 9, the collection of red dotted lines is the network and the balls are threats).

3.4. The artificial potential field method

The initial path is not a practical path for UAV because the UAV kinetics is not considered, such as pitch angle and steering angle. In order to obtain the optimal result, artificial potential field method is used in this paper to improve the path. According to the potential field theory, the basic idea for the problem is that regard the threats of the battlefield as sources of repulsive force and the trajectory as a mass—spring link from the starting point to the target (Shown in Fig. 10). The trajectory is discretized to n average parts and $n+1$ mass points (including starting and terminal points). The "spring" connecting each mass point can be viewed as a rubber band which only feels tension (pull the mass points) but no thrust (push the mass points).

For each initial state of mass-spring link, under the repulsive force from threats, the system will finally reach the steady state (when the entire link is at its minimum potential energy) after fluctuations.

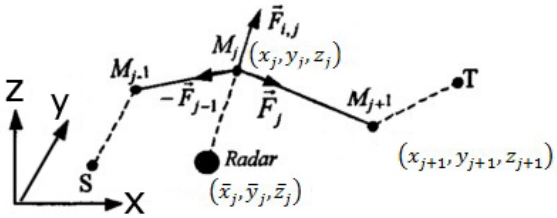

Figure 10. The mass-spring link based on artificial potential field

And all the masses are force balanced. Eventually, the steady state is the optimal path.

The detail of the method is described as follows:

The distance between 2 adjacent mass points is calculated as:

$$\Delta d_j = \sqrt{\left(x_j - x_{j+1}\right)^2 + \left(y_j - y_{j+1}\right)^2 + \left(z_j - z_{j+1}\right)^2} \tag{4}$$

The unit vector from j to j+1 is:

$$n_j = \begin{bmatrix} \left(x_{j+1} - x_j\right)/\Delta d_j \\ \left(y_{j+1} - y_j\right)/\Delta d_j \\ \left(z_{j+1} - z_j\right)/\Delta d_j \end{bmatrix} \tag{5}$$

The unit vector from j-1 to j is:

$$n_{j-1} = \begin{bmatrix} \left(x_{j-1} - x_j\right)/\Delta d_{j-1} \\ \left(y_{j-1} - y_j\right)/\Delta d_{j-1} \\ \left(z_{j-1} - z_j\right)/\Delta d_{j-1} \end{bmatrix} \tag{6}$$

According to *Hooke's Law*, the spring force acting on mass point j is expressed as (k is the spring coefficient):

$$\overrightarrow{F_j} = k\Delta d_j n_j \tag{7}$$

$$-\overrightarrow{F_{j-1}} = -k\Delta d_{j-1} n_{j-1} \tag{8}$$

Two linear damping forces acting on point j are:

$$\vec{F}_{j+} = -b\bullet\left(\begin{bmatrix}\dot{x}_j \\ \dot{y}_j\end{bmatrix}\bullet n_j - \begin{bmatrix}\dot{x}_{j+1} \\ \dot{y}_{j+1}\end{bmatrix}\bullet n_j\right)\bullet n_j = -b\bullet\begin{bmatrix}\dot{x}_j - \dot{x}_{j+1} \\ \dot{y}_j - \dot{y}_{j+1}\end{bmatrix} \tag{9}$$

$$\vec{F}_{j-} = -b\bullet\left(\begin{bmatrix}\dot{x}_j \\ \dot{y}_j\end{bmatrix}\bullet n_{j-1} - \begin{bmatrix}\dot{x}_{j-1} \\ \dot{y}_{j-1}\end{bmatrix}\bullet n_{j-1}\right)\bullet n_{j-1} = -b\bullet\begin{bmatrix}\dot{x}_j - \dot{x}_{j-1} \\ \dot{y}_j - \dot{y}_{j-1}\end{bmatrix} \tag{10}$$

The distance between threat i and mass point j is:

$$d_{i,j} = \sqrt{\left(x_j - x_i\right)^2 + \left(y_j - y_i\right)^2 + \left(z_j - z_i\right)^2} \tag{11}$$

The unit vector from threat i to mass point j is:

$$n_{i,j} = \begin{bmatrix} \left(x_j - x_i\right)/d_{i,j} \\ \left(y_j - y_i\right)/d_{i,j} \\ \left(z_j - z_i\right)/d_{i,j} \end{bmatrix} \tag{12}$$

Finally, the repulsive force from threat i to mass point j is:

$$F_{i,j} = \frac{Q}{d_{i,j}^4} n_{i,j} \tag{13}$$

where, Q is a constant value.

By *Newton's Law*, the motion equation of mass j is:

$$M_j \begin{bmatrix} \ddot{x}_j \\ \ddot{y}_j \\ \ddot{z}_j \end{bmatrix} = \overrightarrow{F_j} - \overrightarrow{F_{j-1}} + \sum_{i=1}^{N} \overrightarrow{F_{i,j}} \tag{14}$$

The motion equations of all mass points compose a set of ordinary differential equations. Despite the equations above have nonlinear term, but they satisfy *Lipschitz* condition. Thus, the differential equations have a unique solution with the given initial conditions.

As a next step, the pitch angle and steering angle as one discrete point by another are verified. Adjust the variable k until every point is appropriate.

Finally, the final optimal path is obtained.

IV. Simulation Results and Analysis

In the simulation, a battlefield with 20 threats scattered in various locations is considered. The location and intensity of threats are known. UAVs fly from starting point S to the target point T throughout the battlefield.

The trajectory cost J is judged by an evaluation function:

$$J = wJ_R + (1-w)J_L \tag{15}$$

where w ($0<w<1$) is the weight coefficient, denoting the trade-off between length cost and threat cost.

Length cost is the length of the trajectory. Since the entire path is formed by connecting a plurality of line segments, the length cost is

$$J_L = \sum_{p=1}^{N} L \tag{16}$$

where N is the number of line segments, L is the length of the p-th line segment.

Threat cost means the degree of UAV exposure to the radars and the threat cost of the p-th line segment can be expressed by

$$J_R = \sum_{p=1}^{N}[L\sum_{i=1}^{n}\int_a^b \frac{1}{\left(\sqrt{\left(x-\overline{x}_i\right)^2 + \left(f(x)-\overline{y}_i\right)^2 + \left(g(x)-\overline{z}_i\right)^2}\right)^4} dx] \tag{17}$$

where n is the number of radars; a and b are the start and the end of the p-th line segment; $f(x)=y$ and $g(x)=z$.

Case 1. *Different entry point and exit point of the network*

The network designed in section 3.2 does not include the starting point *S* and the target point *T*, so the entry point and the exit point on the network must be determined.

Obviously, we can make a shortlist by eyes in Fig 11. Other nodes are neither too far away nor too close to the threat.

Entry point: **7, 15, 35**

Exit point: **6, 39, 48**

Table I shows the cost *J* of the trajectory with different entry point and exit point. Here the weight coefficient w=0.5 and the spring coefficient k=1000.

TABLE I. THE COSTS *J* OF VARIOUS PATHS

Entry point	Exit via 6	Exit via 39	Exit via 48
7	1183	1235	1258
15	935	998	1021
35	1169	1021	1086

In Table I, path with **15** as entry point and **6** as exit point is the optimal path but the other points could not be excluded. When the value of w varies, the entry and exit points may change. The following table shows the optimal entry and exit points with different w. Here the spring coefficient k=1000.

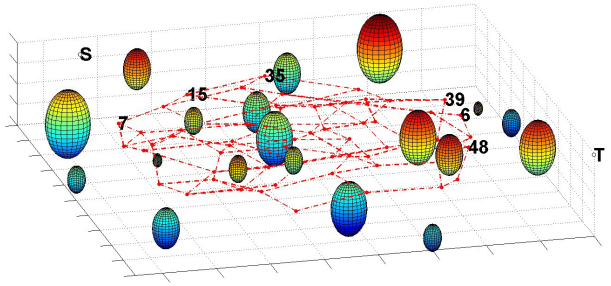

Figure 11. Different entry points and exit points on the network

TABLE II. THE OPTIMAL ENTRY AND EXIT POINTS

The value of *w*	The entry point	The exit point
0.1	**15**	**39**
0.2	**15**	**39**
0.3	**15**	**6**
0.4	**15**	**6**
0.5	**15**	**6**
0.6	**15**	**6**
0.7	**15**	**6**
0.8	**15**	**48**
0.9	**15**	**48**

In Table II, it is obvious that **15** as entry point is optimal and the choice of exit point depends on the value of w.

Case 2. *Different "shortest path"*

In section 3.3, a Dijkstra algorithm is designed to search the shortest path on the network. In the algorithm, the cost function is simply the length cost, with ignoring the threat cost. Although the network itself is created to avoid the threats, but the location of each node is only affected by the threats nearby, without the further threats. So the shortest path may be suboptimal. If one takes (15) as the cost function in Dijkstra algorithm, (15) would be calculated repeatedly both in Dijkstra algorithm and artificial potential field method. Also the integral operation in (15) is complicated. So the improvement may be superfluous. Response to the puzzle above, the following simulations will tell whether it is necessary to take (15) as the cost function.

Comparison for the "shortest path" $P(n)$ with two algorithms obtained by the former algorithm and the improved one is shown in Table III, here the weight coefficient w increases by 0.1 and the spring coefficient k=1000.

TABLE III. THE DIFFERENT "SHORTEST PATH"

w	The former algorithm	The improved algorithm
0.1	15-26-17-25-11-32-19-9-6	15-27-47-43-42-40-39
0.2	15-26-17-25-11-32-19-9-6	15-27-47-47-22-40-39
0.3	15-26-17-25-11-32-19-9-6	15-26-17-34-56-44-42-9-6
0.4	15-26-17-25-11-32-19-9-6	15-26-17-25-11-32-19-9-6
0.5	15-26-17-25-11-32-19-9-6	15-26-17-25-11-32-19-9-6
0.6	15-26-17-25-11-32-19-9-6	15-26-17-25-11-32-19-9-6
0.7	15-26-17-25-11-32-19-9-6	15-26-17-21-5-11-32-19-9-6
0.8	15-26-17-25-11-32-19-9-6	15-26-17-21-5-11-62-41-48
0.9	15-26-17-25-11-32-19-9-6	15-26-17-21-5-11-62-41-48

In Table III, the shortest path with improved algorithm differs from the path with former algorithm when w is 0.1, 0.2, 0.3, 0.7, 0.8, 0.9. And the two paths are same when w is 0.4, 0.5, 0.6. We can make a conclusion that when $w<0.4$ & $w>0.6$, the improved algorithm is necessary; when $0.4 \leqslant w \leqslant 0.6$, the former algorithm is more convenient.

Case 3. *Different* ***w*** *and* ***k***

In the section 3.4, we adjust the spring coefficient k to meet the needs of UAV dynamics. Through a lot of experiments, $500<k<3000$ is an appropriate range in this simulation. Different k makes different path, so not only w but also k affect the trajectory cost J. With a given w, adjust the value of k inside the appropriate range to obtain the minimum trajectory cost J.

Figs. 12-14 show the results of simulation, the black line from *S* to *T* is the optimal path under the given conditions. The collection of red dotted lines is the network and the balls represent threats.

In the simulation, k is a multiple of the hundred and J approximates to integer. 3 typical values (w=0.2, 0.5 & 0.8)

are selected which obviously show the differences between planned paths with different w and k.

When w=0.2, more attention is paid on the length cost. The trajectory is significantly shorter than the other two but closer to the threats.

When w=0.5, a balance between the length and threat is sought. The trajectory is neither too long nor too close to the threats.

When w=0.8, threats are emphasized. The trajectory tries to avoid the threats as much as possible within the allowable range.

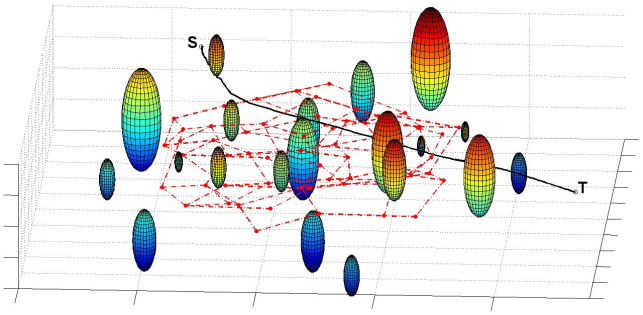

Figure 12. Planned flight path with w=0.2, k=2500, J=882

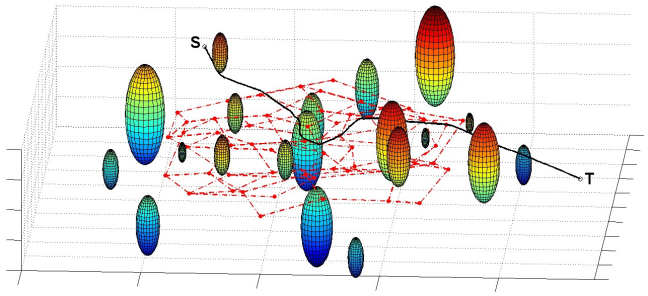

Figure 13. Planned flight path with w=0.5, k=1000, J=935

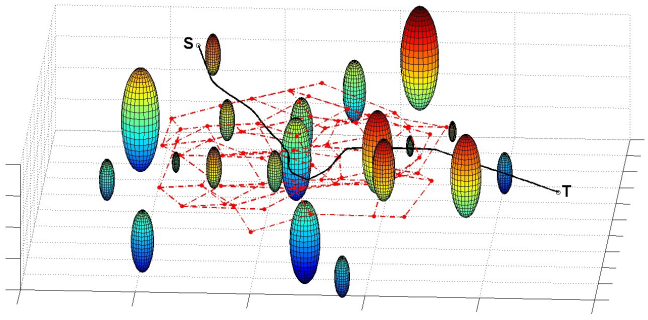

Figure 14. Planned flight path with w=0.8, k=600, J=1036

V. Conclusion

In this paper, an effective strategy for obtaining an optimal path for UAV in 3-D environments is investigated and proposed. This proposed method, which is based on combination of graph theory, Dijkstra algorithm and potential field, considers comprehensively the effects of threats, UAV kinetics and flight length. The proposed method is tested in a simulation of battlefield and the simulation results demonstrated the effectiveness and feasibility of the proposed method.

Acknowledgment

The authors would like to thank all the colleagues of the Control and Information Institution in Northwestern Polytechnical University and the researchers of the Diagnosis, Flight Control and Simulation Lab in Concordia University for their help.

References

[1] R. K. Barnhart, S. B. Hottman, D. M. Marshall, et al, *Introduction to Unmanned Aircraft Systems*. CRC Press, Taylor & Francis Group, 2011.

[2] C. A. Liu, H. P. Wang, W. J. Li, "On Path Planning for More Efficient Reconnaissance of UAV," *Journal of Northwestern Polytechnical University*, 2003 (8), pp. 490-494.

[3] L. Merino, J. Wiklund, F. Caballero, et al, "Vision-based multi-UAV position estimation," *IEEE Robotics & Automation Magazine*, Vol. 13, No. 3, 2006, pp. 53-62.

[4] B. Sathyaraj, L. Jain, A. Finn, et al, "Multiple UAVs path planning algorithms: A comparative study," *Fuzzy Optimization and Decision Making*, 2008, 7(3): pp. 257-267.

[5] A. Fridman, S. Weber, V. Kumar, et al, "Distributed path planning for connectivity under uncertainty by ant colony optimization," *2008 American Control Conference,* June 2008, pp. 1952-1958.

[6] K. J. Obermeyer, "Path planning for a UAV performing reconnaissance of static ground targets in terrain," *AIAA Modeling and Simulation Technologies Conference,* August 2009, pp. 1-11, AIAA-2009-5888.

[7] F. L Leng, *Decentralized Motion Planning Within an Artificial Potential Framework for Cooperative Payload Transport by Multi-robot Collectives*, M. Sc. Thesis, State University of New York, New York, US, December. 2004.

[8] T.-Y. Sun, C.-L. Huo, S.-J. Tsai, et al, "Optimal UAV flight path planning using skeletonization and partical swarm optimizer," *2008 IEEE Congress on Evolutionary Computation*, June 2008, pp. 279-288.

[9] Z. Hao, J. Zhang, "3-D path planning for UAV in complex condition," *Institute of Engineering*, Air Force Engineering University, Xi'an, China (in Chinese).

[10] I. Oz, H. R. Topcuoglu and M. Ermis, "A meta-heuristic based three-dimensional path planning environment for unmanned aerial vehicles," *Sage Journals*, 2012.

[11] C. Rasche, C. Stern, L. Kleinjohann and B. Kleinjohann, "A distributed multi-UAV path planning approach for 3D environments," *International Conference on Automation, Robotics and Applications*, 2011, pp.7-12.

[12] N. Özalp and O. K. Sahingoz, "Optimal UAV path planning in a 3D threat environment by using parallel evolutionary algorithms," *International Conference on Unmanned Aircraft Systems*, 2013.

[13] Y. Qu, Q. Pan and J. Yan, "Flight path planning of UAV based on heuristically search and genetic algorithms," in *the 31st Annual Conference of IEEE Industrial Electronics Society*, 2005.

[14] H. Borouchaki, S. H. Lo, "Fast delaunay triangulation in three dimensions," *Comp. Meth. Appl. Mech. Engineering*, 1995, pp.153-167.

[15] J. F. Thompson, B. K. Soni, N. P. Weatherwill, *Handbook of Grid Generation*, CRC Press, Boca Raton, 1999.