The Dynamic Path Planning of Drone Clusters Based on the Improved Artificial Potential Field

H127

陶超凡	英才实验学院	2016090101011
徐子璇	英才实验学院	2016070905029
李松岷	英才实验学院	2016010905024

2017.11.20

-

The Dynamic Path Planning of Drone Clusters Based on the Improved Artificial Potential Field

H127

Summary

With hundreds of intel drones light up the sky, people are increasingly attracted by the drone clusters. Given an image, drone clusters can display it vividly in the sky. Our goal is to determine the positions of the drones and devise an optimized flight path for them.

To simplify modeling process, we assume the air space is big enough and there is not any obstacle in the sky. Besides, we firstly ignore the factor of weather.

We begin our work with the image processing. Using five operators respectively (Sobel, Roberts, Prewitt, Laplasian and Canny), we extract the edge-characteristics of the image and choose the most suitable one. Then we select 900 dots from the edge and determine the initial positions of the drones.

Second, we are expected to allocate the drones to their positions in the sky. It is a typical Allocation Problem and we use the Hungarian Algorithm to get the optimized solution.

Third, we establish the Dynamic Route Planning Model and regard Artificial Potential Field (APF)as a suitable and effective algorithm. By conducting two virtual potential fields—attraction field and repulsion field, we could quickly determine the drones' optimized route. Furthermore, based on the traditional APF, we investigate its limitations and put forward our modification. Via MATLAB, we successfully devise the optimized flight path.

Finally, we make sensitivity analysis for our model and analyze its strengths and weaknesses. We believe our model is robust and provide our recommendation to do the aerial light show in our university.

Content

1 Introduction

1.1Background

At festivals, we used to be familiar with parties, music and fireworks. They were traditional elements of a joyful night. But recently, thanks to the drone technology, intel led us witness an entirely new form of entertainment: a perfectly choreographed light show with 500 drones—drawing stunning pictures to light up the sky. For example, From **Fig. 1**, we can see the drone clusters displaying the spelling of ‘intel’.

Actually, drone clusters have already been an indispensable part in our daily life. Taking the advantages of automatic system and small size, drones have wide applications in reconnaissance and superaltitude attack. They also make increasing contribution for civil use, such as surveying, mapping and rescuing disaster areas.

Now a sky light show expands our understanding of drones. Aimed to present the required picture in the sky, the route planning is one of the most difficult tasks. Route planning is defined as finding an optimal trail from the starting point to the destination, in order to meet the expected performance indexes under specific constraints.

There are mainly two categories of common algorithms to deal with the route planning. One is the optimization algorithm based on Cybernetics, such as nonlinear programming method and conjugate gradient method; the other is search algorithm based on Geometry such as A* search algorithm, colony algorithm and artificial potential field method. It is essential to choose an suitable algorithm to determine the optimal flight route.

Fig.1 A light show with 500 drones

1.2 Problem restatement

Our university is considering an outdoor aerial light show. We were asked to investigate the idea of using drones to create three possible sky displays. The sky displays are as follows:

Display 1: Dragon
Display 2: The logo of uestc

Display 3: Create our own image--Firework

The problems that we need to solve in this paper are

- How to determine the number of drones required.
- How to determine the initial location for each drone.
- Which is the most suitable flight path of each drone or set of drones that would animate our images
- Describe our animation.
- There are many other requirements for our 3-display light show such as the number of drones, required launch area, required air space, safety considerations, duration of the show and so on. How will these factors affect the flight show.
- Write a two-page memo to report the results of our investigation and make a recommendation as to whether or not to do the aerial light show.

1.2 Fundamental Assumptions

To simplify the problem and make it convenient for us to simulate real-life conditions, we make the following basic assumptions, each of which is properly justified.

- **The air space is big enough and there is not any obstacle in the sky.** Compared with other flight events, the air space of a sky light display is relatively small. So it is natural to assume the air space unlimited.
- **The conditions of each drone are assumed to be the same and stable. The electric power of each drone is enough in the duration.** Even there are slight differences between the drones in the manufacturing process, it is unnecessary for us to consider them. Meanwhile we do not consider the electric power to simplify the problem.
- **We firstly ignore the factor of weather.** In reality, weather plays a crucial role in determining whether a sky light display will be a success. But to simplify the problem, we will not consider it initially and we will discuss it in the following sections.

1.4Notations

Table 1 Notations

Symbols	Definition
n	the total number of drones
k_a	attraction gain
k_r	repulsion gain
U_{att}	Attraction potential energy
U_{rep}	Repulsion potential energy
$\mathbf{F}_{att}$	Attraction
$\mathbf{F}_{rep}$	Repulsion
$\mathbf{F}_i$	the resultant force of drone i
R	the radius of the repulsion field
d_{ij}	the distance between the drone i and j
U_i	the total potential energy of drone i

2 Image Processing

2.1 Analyses

Our first task is to determine the minimum number of drones. Using the image processing techniques, we successfully transform the picture into an array of dots via MATLAB. These dots preserve the main features of the image and the number of them are as small as possible. Then the number of dots is the minimum number of drones.

The main technical route of our method could be described as follow:

- First, we decide to extract the edge-characteristics of the image. To present an image in front of the audience, we must preserve its main characters. Edge reflects the fundamental features of an image, containing most of the information and differing the object from the background. So extracting the edge-characteristics of the image is the basis of image segmentation and feature extraction.

- Then we try to find a suitable method to extract the edge-characteristics. We consider five typical operators of edge detection, which are Sobel, Roberts, Prewitt, Laplasian and Canny. Using these operators respectively, we process the given picture and get the edge of it. By comparing these operators, we select the most suitable one. It must preserve the edge-characteristics as much as possible, containing the minimum number of dots as well.

- We further select a number of pixels evenly from thousands of pixels on the edge. To figure out the suitable number of pixels we should choose, we try for some times and compare the differences between different images. We finally decide 900 as the final answer. The features of the image could be preserved well and the number of drones is not so big. //Thus the position of each drone could be determined.

- **Determine the horizontal and vertical coordinates.**
  Since we have transformed the picture into a set of dots, we can get the characteristic matrix of the image. We name the matrix as $A_{h\times w}$. Then x is the column ordinal number of A.

$$\text{If } \mathrm{mod}(h,x)\neq 0\text{, then } i=\mathrm{mod}(h,x)\text{,else i=h.}$$

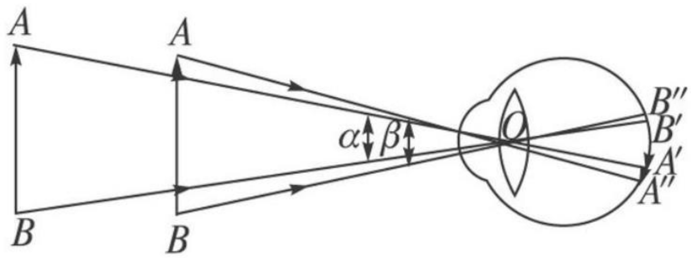
$$j=\frac{x-i}{h}+1$$

  Where :

  i is the horizontal coordinate of the drones' target positions.
  j is the vertical coordinate of the drones' target positions.

  In this way, through the characteristic matrix of the image, we could get the matric of the drones' target positions.

- **Determine the height**.
  The human eye diopter is defined as the largest degree that human could see. It is usually 120°.Specially, when people concentrate on a certain thing, it becomes 25°.Based on the principle of eyes imaging, the angle of view determines the size of the image on the retina. As figure shows with the angle of view becomes larger, the figure human could see will be bigger.



According to the scale model, our team regard the $\angle O$ of △ABO is the human eye diopter. It is easy to find that:

$$h=\frac{\overline{AB}}{2tan\frac{\angle O}{2}}$$

$\overline{AB}$ is the maximum size of the image showed.

As the equation shows, we can change the height of $\overline{AB}$, h, to satisfied the human eye diopter.

- **Duration of the light show**

A way to estate the duration of the light show, which is based on the times of iteration, is to use the maximum of the times of iteration to multiply the time it use.

$$T = \sum_{n=1}^{\max(N_i)} \frac{l_n}{v}$$

In the equation, T is the total time of each change

l_n is the length of step while the times of iteration is n

v is the max speed of the drone

N_i is the convergence times of iteration of ith drone

2.2Results

2.2.1Dragon

After extracting the edge-characteristics, we could see the different effects of these operators in Fig.2. We finally choose Laplasian based on the analyses above.

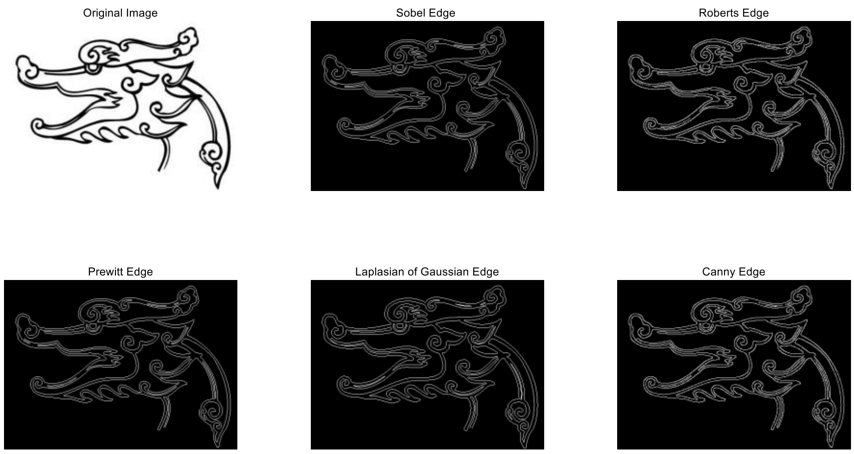


Fig 2 Extracting the edge-characteristics via different operators(dragon)

Table 2 The number of drones under different operators(dragon)

Operator	Sobel	Roberts	Prewitt	Laplasian	Canny
Number of drones	8705	11158	8697	8860	10495

Then we compute the characteristic matrix of the image, which is $A_{359\times494}$ .We set the distance between each element as 1m. As the method above, we get the matrix of the drones' target positions, which is shown in **Fig.4**.

Fig.3 The original picture (dragon)

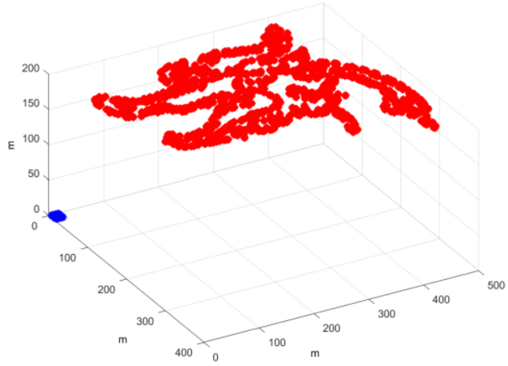


Fig.4 UAVs: initial positions and target positions(dragon)

2.2.2 The logo of uestc

Similarily, we could see the different effects of these operators in Fig.6. We finally choose Canny based on the analyses above.

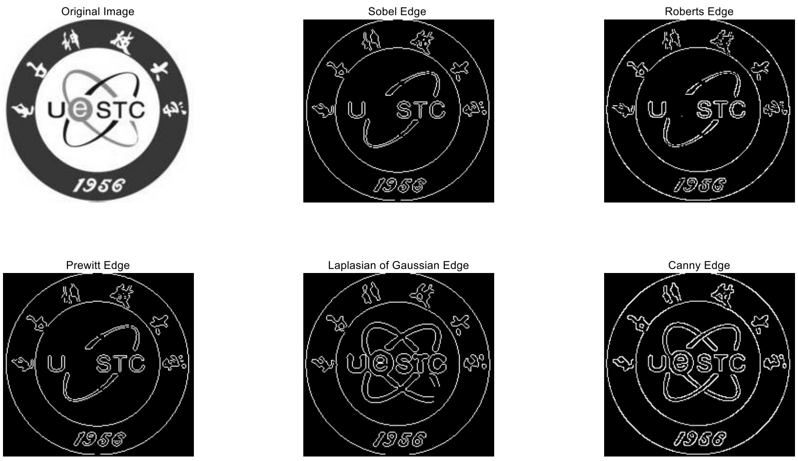


Fig.5 Extracting the edge-characteristics via different operators(uestc)

Table 3 The number of drones under different operators(uestc)

operator	Sobel	Roberts	Prewitt	Laplasian	Canny
Number of drones	2177	2442	2167	2644	3294

The characteristic matrix of the image is still $A_{359\times494}$ and the distance between each element is 1m. The matrix of the drones' target positions is shown in **Fig.7**.

Fig.6 The original picture (uestc)

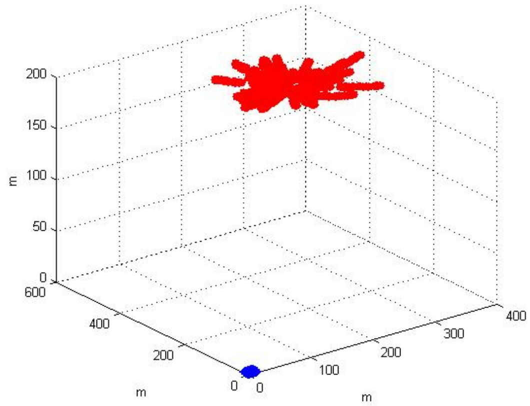


Fig.7 UAVs: initial positions and target positions(uestc)

2.2.3 Firework

Similarily, we could see the different effects of these operators in Fig.8. We finally choose Prewitt based on the analyses above.

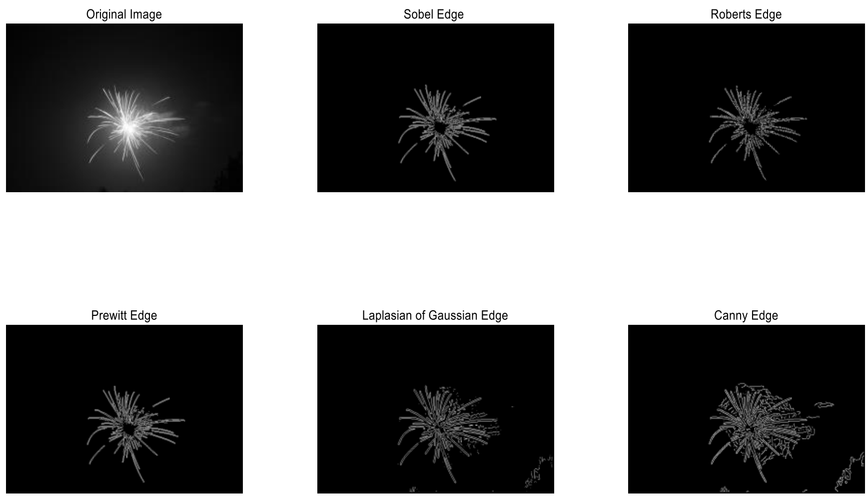


Fig.8 Extracting the edge-characteristics via different operators(firework)

Table 4 The number of drones under different operators

operator	Sobel	Roberts	Prewitt	Laplasian	Canny
Number of drones	4708	3889	4717	5332	7726

The characteristic matrix of the image is still $A_{359\times494}$ and the distance between each element is 1m. The matrix of the drones' target positions is shown in **Fig.10**.

Fig.9 The original picture (firework)

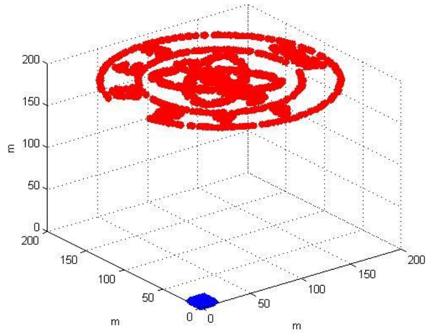


Fig.10 UAVs: initial positions and target positions(uestc)

3.The Allocation Model

Based on the analyses above, we have already transformed the picture into an array of dots. Each dot represents a position of the drone in the sky. The initial position of the drone is mapped to its position in the sky.

Our goal is to devise a strategy to allocate the drones to their positions in the sky. We regard this problem as an Assignment Problem and we use Hungarian Algorithm to get the optimization solution

Suppose the number of the drones is n. So there are a set of n points on the launch area and a set of n points in the sky. We name the dots on the launch area as $P_i(x_{i,}y_i)$ and the dots in the sky as $Q_j(x_{j,}y_j)$ for $i, j = 1, 2, ..., n$.

Then the distance between P_i and Q_j is

$$c_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

We construct a Coefficient Matrix:

$$D = (c_{ij})_{n\times n} = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{nn} & \cdots & c_{nn} \end{pmatrix}$$

Furthermore, we use another matrix $\mathbf{X}$ to represent the allocation of each dots.

$$\mathbf{X} = (x_{ij})_{n\times n} = \begin{cases} 0 & , P_i \text{ is not maped to } Q_j \\ 1 & , P_i \text{ is maped to } Q_j \end{cases} \quad i, j = 1, 2, ..., n$$

The objective function is

$$z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij} \quad .$$

Our goal is to get the minimum of z .

The constraints are

$$\begin{cases} \sum_{i=1}^{n} x_{ij} = 1 & i = 1, 2, ..., n \\ \sum_{j=1}^{n} x_{ij} = 1 & j = 1, 2, ..., n \\ x_{ij} = 0 \; or \; 1 & i, j = 1, 2, ..., n \end{cases} \quad .$$

We construct a new matric $(c'_{ij})_{n\times n}$.

$$c'_{ij} = M - c_{ij} \; .$$

Where M is a constant which is sufficient enough and we usually take M as the maximum of c_{ij} .

$$z' = \sum_{i=1}^{n}\sum_{j=1}^{n} (M - c_{ij}) x_{ij}$$

$$= \text{Mn-}\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij}$$

To get the minimum of z' ,we could derive the matrix $(x_{ij})_{n\times n}$, thus getting the mapping from the dots on land to the dots in the sky. Then the allocation of drones can be determined.

4. Model II: The Dynamic Route Planning Model

4.1 The Traditional Artificial Potential Field

To deal with dynamic routes planning of drone clusters, Artificial Potential Field (APF)is a common and useful algorithm which is proposed by Khatib in 1980s. APF is based on Operational Space Formulation. It is suitable for dynamic real-time system.

We think Artificial Potential Field is a suitable method especially for this situation. In our case, each drone needs to move to their destination and should not hit other drones in their way. The common problems for drone clusters are static threat such as terrain. But in our situation, the threats from other drones are dynamic. So we prefer to choose Deterministic Search Algorithm instead of Random Search Algorithm. Deterministic Search Algorithm does not search those unnecessary paths, thus saving a lot of time.

We could conduct two virtual potential fields—attraction field and repulsion field. Both of them are vector fields. Attraction is generated by the target (the drone's destination). It points from the drone to the destination. Repulsion is generated by the obstacles (the other drones). It points from the drone to the other ones around it. Besides, we have to mention that the repulsion field has a radius. Only when the drone is in the radius of the other ones, it will be repulsed in this threat area.

So drones move under the effects of the attraction and repulsion. Under these two forces, the drone moves in the direction to achieve the minimum potential energy. When the drone reaches its destination, the potential energy is zero. We define two differentiable continuous functions--**attraction potential function** U_{att} and **repulsion potential function** U_{rep}.

Suppose the number of drones is n, we could number the drones from 1,2, 3,…,*n*.
For *i*, *j*=1,2,3,…,*n*

$$U_{att,i} = \frac{1}{2} k_a d_{ij}^2$$

$$U_{rep,i,j} = \begin{cases} \frac{1}{2} k_r \left(\frac{1}{d_{ij}} - \frac{1}{R}\right)^2 & , d_{ij} \le R \\ 0 & , d_{ij} > R \end{cases}$$

Where:

k_a is attraction gain and k_r is repulsion gain.

d_i is the distance between the drone and its destination

R is the radius of the repulsion field.

d_{ij} is the distance between two drones.

So the total potential energy of each drone is

$$U_i = U_{att,i} + \sum_j U_{rep,i,j}$$

The negative gradients of $U_{att,i}$ and $U_{rep,i}$ are the forces acting on each drone.

$$\mathbf{F}_{att,i} = -\operatorname{grad}(U_{att,i}) = -k_a d_i$$

$$\mathbf{F}_{rep,i,j} = -grad(U_{rep,i,j}) = \begin{cases} k_r\left(\frac{1}{d_{ij}} - \frac{1}{R}\right)\frac{1}{d_{ij}^2} & ,d_{ij} \le R \\ 0 & ,d_{ij} > R \end{cases}$$

Where:

$\mathbf{F}_{att,i}$ is the attraction acting on the drone.

$\mathbf{F}_{rep,i,j}$ is the repulsion acting on the drone.

The resultant force of a drone is the vector sum of $\mathbf{F}_{att,i}$ and $\mathbf{F}_{rep,i,j}$:

$$\mathbf{F}_i = \mathbf{F}_{att,i} + \sum_j \mathbf{F}_{rep,i,j}$$

4.2 The Limitations of Artificial Potential Field

However, the Artificial Potential Field has some limitations.

- **Problem 1. Hit other drones**
  When the drone is far away from the destination, the attraction is extremely large while the repulsion is relatively small. So we can even ignore the repulsion. Under this circumstance, the drone may hit the other ones in the way.

- **Problem 2. Unreachable destination**
  When there is an obstacle near the drone's destination, the repulsion is extremely large while the attraction is relatively small, so it is difficult for the drone to reach its destination.

- **Problem 3. Local optimal**
  Local optimal problem is one of the biggest weakness of APF. When the attraction and the repulsion are equal and opposite, the resultant force is zero. The drone will stay stable at some positions where the potential energy is the minimum in a local place but not the whole space. That means the drone will not reach its destination. So we add a random perturbation to let the object jump out of the local optimum.

4.3 The Improved Artificial Potential Field

Based on the limitations above, we improve the traditional APF as follow:

- **The modification of Problem 1**

  We modify the attraction potential function U_{att}. This can avoid the large attraction when the drone is far away from its destination. Specifically, we put forward a certain distance d_{goal}^* .When the distance between the drone and its destination change, the attraction potential energy have different expressions:

$$U_{att,i} = \begin{cases} \frac{1}{2}k_a d_i^2 & d_i \le d_{goal}^* \\ d_{goal}^* k_a d_i - \frac{1}{2}k_a \left(d_{goal}^*\right)^2 & d_i \le d_{goal}^* \end{cases}.$$

Where: d_{goal}^* is a threshold of the distance between the drone and its destination. Meanwhile ,the negative gradient of $U_{att,i}$ is the attraction acting on each drone.

$$\mathbf{F}_{att,i} = -\operatorname{grad}(U_{att,i}) = \begin{cases} k_a\left(\mathbf{q}-\mathbf{q_{goal}}\right) & d_i \le d_{goal}^* \\ \dfrac{d_{goal}^* k_a\left(\mathbf{q}-\mathbf{q_{goal}}\right)}{d_i} & d_i \le d_{goal}^* \end{cases}$$

- **The modification of Problem 2**

We modify the repulsion potential function U_{rep} .Considering the unreachable destination with an obstacle nearby, we add the distance between the drone and its destination d_i into consideration.

$$U_{rep,i,j} = \begin{cases} \frac{1}{2}k_r\left(\frac{1}{d_{ij}} - \frac{1}{R}\right)^2 d_i^n & ,d_{ij} \le R \\ 0 & ,d_{ij} > R \end{cases}$$

Meanwhile, the negative gradient of $U_{rep,i,j}$ is the repulsion acting on each drone.

$$\mathbf{F}_{rep,i,j} = -grad(U_{rep,i,j}) = \begin{cases} F_{rep,i,j,\text{o-r}} \dfrac{\left(\mathbf{q}_i - \mathbf{q}_j\right)}{\|\mathbf{q}_i - \mathbf{q}_j\|} + F_{rep,i,j,\text{r-g}} \dfrac{\left(\mathbf{q}_{goal} - \mathbf{q}_i\right)}{\|\mathbf{q}_{goal} - \mathbf{q}_i\|} & d_{ij} \le R \\ 0 & d_{ij} > R \end{cases}$$

Where:

$$F_{rep,i,j,\text{o-r}} = k_r\left(\frac{1}{d_{ij}} - \frac{1}{R}\right)\frac{d_i^n}{d_{ij}^2}$$

$$F_{rep,i,j,\text{r-g}} = \frac{n}{2}k_r\left(\frac{1}{d_{ij}} - \frac{1}{R}\right)^2 d_i^{n-1}$$

That means the repulsion has two components of forces. One points from the drone i to the other drone j, the other points from the drone i to its destination.

Coordinate points iteration by : $\mathbf{q}(i,t+1)=\mathbf{q}(i,t)+l\times\frac{\mathbf{F}(i,t)}{\|\mathbf{F}(i,t)\|}$.

The flow chart of the improved APF algorithm is displayed as follow:

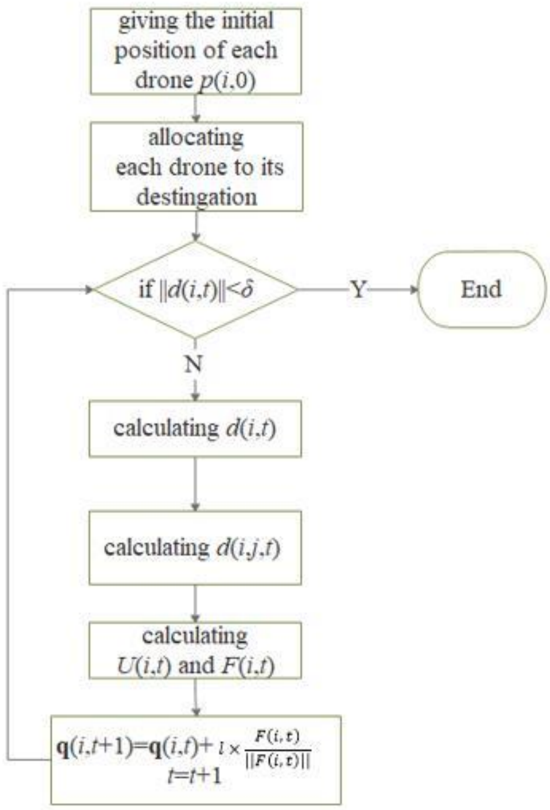


Fig11.The flow chart of the improved APF algorithm

- **The modification of Problem 3**
  We could add a random perturbation to let the object jump out of the local optimum.

4.4 Results

Based on The Improved Artificial Potential Field, we design the algorithm as Fig.11 shows.

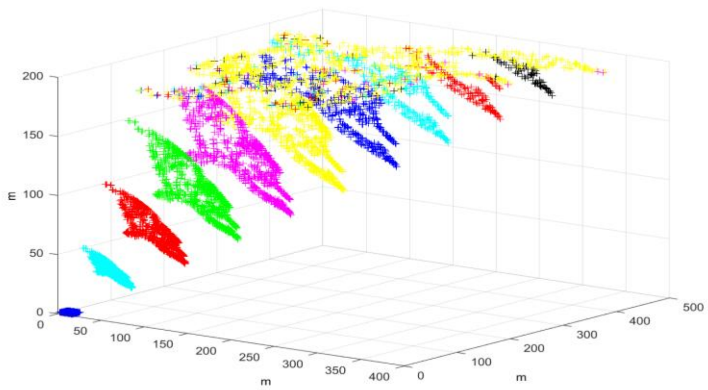


Fig.12 The trail of all the drones during 50 iterations (dragon)

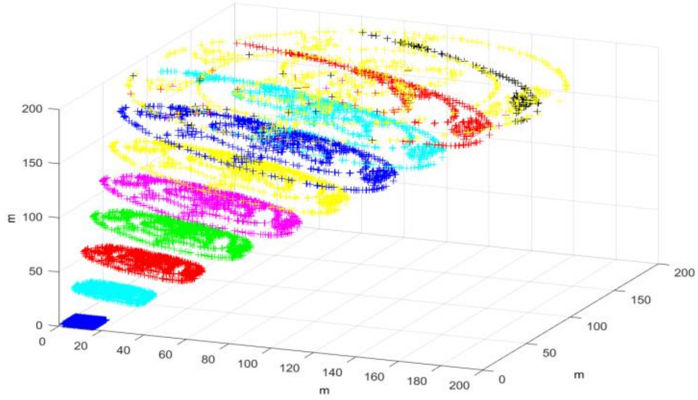


Fig.13 The trail of all the drones during 50 iteration (uestc)

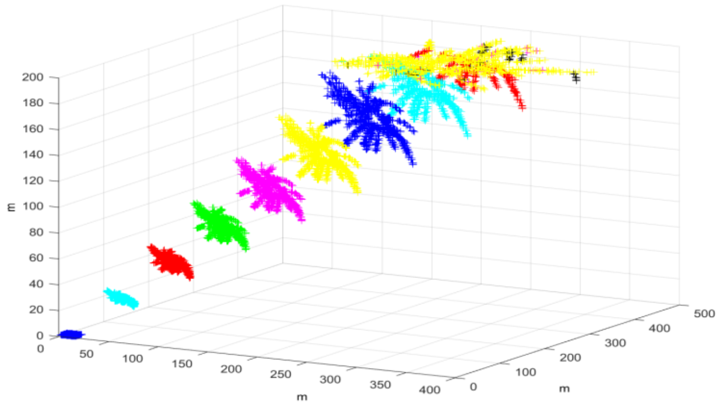
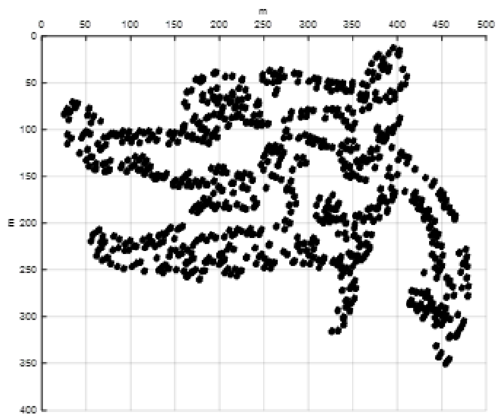


Fig. 14 The trail of all the drones during 50 iterations (firework)



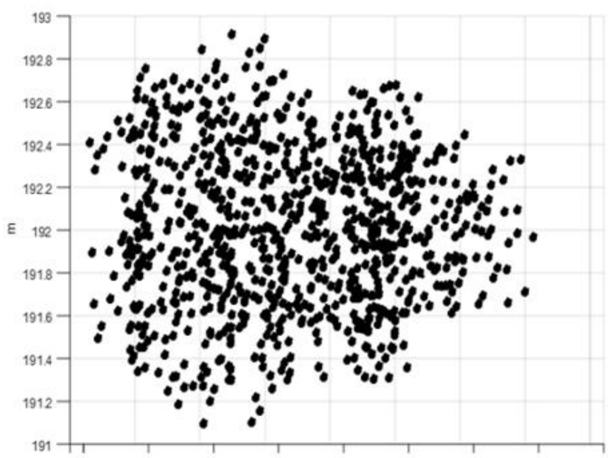


Fig.15 The vertical view of the drones after

300 iterations (dragon)

Fig.16 The lateral view of the drones after 600 interactions(dragon)

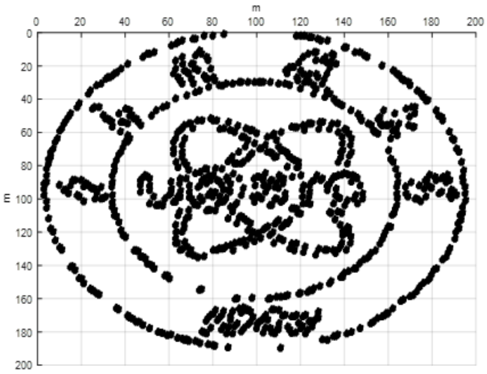


Fig.17 The vertical view of the drones after 600 iterations (uestc)

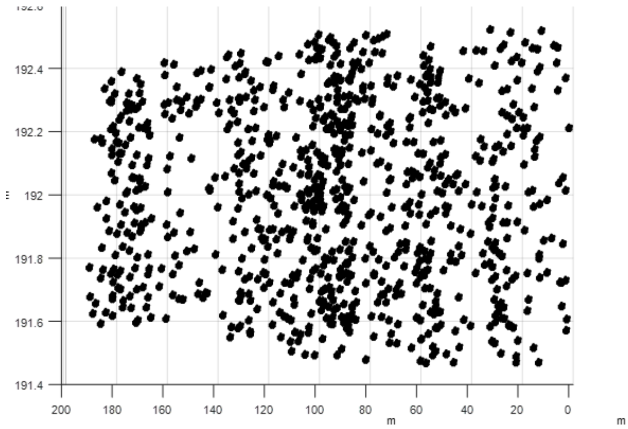


Fig.18 The lateral view of the drones after 600 interactions(uestc)

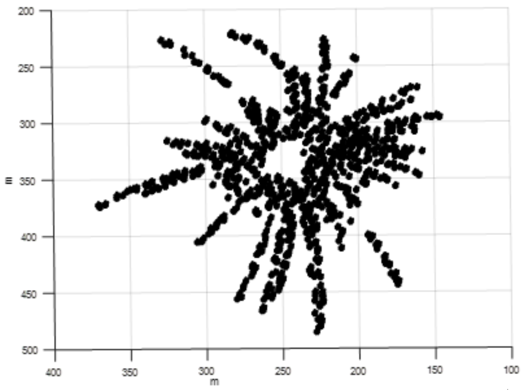


Fig.19 The vertical view of the drones after 600 iterations (firework)

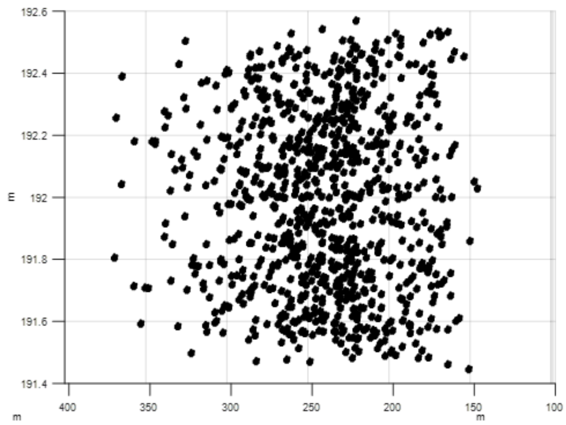


Fig.20 The lateral view of the drones after 600 interactions(firework)

5. Sensitivity Analysis

In reality, weather plays a crucial role in determining whether a sky light display will be a success. Many factors such as wind and rain will influence their flight paths or even causing dangerous accidents.

To test the stability of our model, we use the random perturbation strategy to simulate the influence of wind and rain. As mentioned above, according to APF, we conduct two virtual forces—attraction and repulsion. They determine the route of the drones.

Now we take the weather into consideration. Under the effect of wind or rain, the drone will deviate from its original position. So we conduct another force to simulate the wind force. The direction and magnitude of this vector is randomly produced. Since the wind power a drone can stand is limited, the magnitude must have lower and upper bonds, which is determined by the realistic situation.

6.Strengths and Weaknesses

6.1Strengths

- **Suitable algorithm.** Artificial Potential Field is a suitable and effective method especially for dynamic routes planning. As a Deterministic Search Algorithm, APF saves a lot of time calculating and it can find the optimized route quickly.
- **Flexible and Extendable**. Our model could be extended to apply to many situations. We give a general way to produce a sky light show by drones.

6.2Weaknesses

- We assume all the drones fly at the same height, so the possible positions of each drone in three-dimensional space is numerous and the flight path is relatively complicated. We may learn from those successful examples of aerial light shows: the drone clusters can be divided into different groups and each group moves in a fixed height. They make up a picture together.

7 Letter to the president of our university

Dear president,

In response to your questions regarding whether to hold the aerial light show, we try to use drone clusters to create three possible sky displays. Now we are honored to inform you of our work and give our recommendation.

To simplify the problem, we assume the air space is big enough for our show. And we suppose we have chosen a fine day to do the aerial light show. There is not any rain or wind.

Image processing is the first step of our work. To present a picture in front of the audience, we must derive the main characters of the image. So we extract the edge-characteristics of the image. After due considerations, we decide that we need 900 drones. They should be arranged in a 30×30 matrix. The distance between each one in the same line is 0.5 metres.

Then we use the Hungarian Algorithm to allocate the drones to their positions. It can ensure the total distance to be the minimum.

Further, Dynamic Route Planning is the most challenging task for our team. We regard Artificial Potential Field as an effective and suitable algorithm to devise an optimized flight path for each drone. Based on the traditional APF, we investigate its limitations and put forward our modification. Via MATLAB, we successfully devise the optimized flight path. So we think it is technically feasible to use drone clusters to create sky displays.

Finally, we make sensitivity analysis for our model and we believe our model is robust . Based on the study of our team, we recommend our university to hold the aerial light show. The reasons are as follows:

1.Based on our model, it is technically feasible to use drone clusters to create sky displays.

2. The launch area is only $225\,m^2$,which does not cover too much area. It is relatively easy to arrange the field. So we suggest that the playground is a suitable launch place.

3.It does not take too much human force to accomplish the whole display, most of the work can be accomplished by automation

4. In reality, weather plays a crucial role in determining whether a sky light display will be a success. Many factors such as wind and rain will influence the flight paths of drones or even causing dangerous accidents.

But the duration of the display is relatively small. According to the parameter of the intel drone, the longest duration of each drone is 20 minutes. So if we choose a fine day and the possibility of wind and rain will be small during the display.

5. We believe sky light displays is going to be a fashion gradually. By arranging the sky light display, teachers and students can learn more about drone clusters, enhancing their interest towards the advance technology.

However, we still have to mention the disadvantages of the sky light show. Safety problems are our primary concern. We must consider some possible accidents, such as the collision of drones or the fall of drones. The emergency measures must be prepared in advance.

To sum up, we successfully achieve your goal to create three possible sky displays. Since the reasons above, we recommend holding the sky light show. If you agree with our recommendation, we must do further study to investigate the possible threats and work out the emergency measures in advance.
Yours, sincerely

References

[1] Khatib O. (1986) Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. In: Cox I.J., Wilfong G.T. (eds) Autonomous Robot Vehicles. Springer, New York, NY
[2] Vadakkepat P, Tan K C, Wang M L. Evolutionary artificial potential fields and their application in real time robot path planning[C]// Evolutionary Computation, 2000. Proceedings of the 2000 Congress on. IEEE, 2000:256-263 vol.1.
[3] Zhongtong.G, Hongjie.H. Flight path planning for multiple UAVs based on artificial potential field method [J]. Automation application, 2017(9).
[4] Meng.L, Daobo.W, Tingting.B etc. UAV path planning based on ant colony optimization algorithm and artificial potential field [J]. Journal of Applied Sciences, 2012, 30(2):215-220.
[5] Yancai.C. Research on distributed cooperative control method for swarm UAVs [D]. Nanjing University of Aeronautics & Astronautics, 2011.
[6] Changan.L. Research on UAV route planning method [D]. Northwestern Polytechnical University, 2003.
[7] Shaolei.Z, Yahui.Q, Wenguang.Z etc. Formation control of unmanned aerial vehicle considering internal collision avoidance [J]. Tactical missile technology, 2016(6):94-98.

Appendix：

1. **Multiple ways to extract the images' features (including: sobel, roberts, prewitt, laplasian, canny arithmetic operators)**
2. **Initialize the detailed positions of initial planes and target planes**
3. **Optimal distribution of initial planes and target planes**
4. **Defination of the attractive forces and repulsive force**
5. **Main program**

1. **Multiple ways to extract the pictures' features (including: sobel, roberts, prewitt, laplasian, canny arithmetic operators)**

```
planes_num   = [];
%   dragon---Laplasian
%   uestc--- canny
%   firework---prewitt
I1=imread('C:\dragon3.jpg');
I2=imread('C:\uestc.jpg');
I3=imread('C:\firework.jpg');

I1=rgb2gray(I1);
I2=rgb2gray(I2);
```

```
I3=rgb2gray(I3);

%subplot(2,3,1);
%imshow(I1,[]);
%title('Original Image');

%sobelBW=edge(I1,'sobel');
%planes_num(1) = length(find(sobelBW==1));
%subplot(2,3,2);
%imshow(sobelBW);
%title('Sobel Edge ');

%robertsBW=edge(I1,'roberts');
%planes_num(2) = length(find(robertsBW==1));
%subplot(2,3,3);
%imshow(robertsBW);
%title('Roberts Edge');

prewittBW3=edge(I3,'prewitt');
planes_num(3) = length(find(prewittBW==1));
%subplot(2,3,4);
%imshow(prewittBW);
%title('Prewitt Edge');

logBW1=edge(I1,'log');
planes_num(1) = length(find(logBW==1));
%subplot(2,3,5);
%imshow(logBW);
%title('Laplasian of Gaussian Edge');

cannyBW2=edge(I2,'canny');
planes_num(2) = length(find(cannyBW==1));
%subplot(2,3,6);
%imshow(cannyBW);
%title('Canny Edge');
```

2. Initialize the detailed positions of initial planes and target planes

```
target_plane = cannyBW2;
init_num = planes_num(2);

ones_pos = find(target_plane==1);
target_real_plane = zeros(size(target_plane));
all_planes_num = 900; %规定每个图片都要用 900 个飞机组成
```

```
row = 30;
target_real_plane(ones_pos(round(linspace(1,init_num,all_planes_num)))) = 1;
imshow(target_real_plane)

ground_pos = cell(row,row); % 16*16 的飞机

dl1 = 0.5; %初始相邻的飞机相距 0.5m

for i = 1:row
    for j = 1:row
        ground_pos{i,j}=(1+[(i-1)*dl1,1+(j-1)*dl1,0]);
    end
end

target_pos = cell(row,row);
[hh_cen,ww_cen] = size(target_real_plane);

dl2 = 1; %空中稳定后的飞机的理想相邻距离 1m

height0 = 192; %理想飞行高度，加随机扰动

k = 1;
index = find(target_real_plane ==1);

for i =1:row
    for j = 1:row
        if   mod(index(k),hh_cen)~= 0
            ii = mod(index(k),hh_cen);    % xi=ground plane code
        else
            ii = hh_cen;
        end
        %height = height0 +0.01*height0*rand();
        jj = (index(k)-ii)/hh_cen+1;
        target_pos{i,j} = ([(ii-1)*dl2 ,(jj-1)*dl2, height0 ]);
        k = k+1;
    end
end

distance_mat =    zeros(row*row,row*row);
for i =1:row*row
    for j = 1:row*row
        distance_mat(i,j) = sqrt(sum((ground_pos{i} - target_pos{j}).^2));
    end
end
%*************************************************************%
%distribution problem
```

```
[order,fval]=fenpei(distance_mat);
for i =1:row*row
    plot3(ground_pos{i}(1),ground_pos{i}(2),ground_pos{i}(3),'b*');
    hold on
end
hold on
for i =1:row*row

plot3(target_pos{i}(1),target_pos{i}(2),target_pos{i}(3),'r*','linewidth',5,'markersize',
5);
    hold on
end
grid on
xlabel('m');
ylabel('m');
zlabel('m');
%****************************************************************%
distribute = find(order==1);
```

3. Optimal distribution of initial planes and target planes

```
function [y,fval]=fenpei(C)
C=C';
f=C(:);
[m,n]=size(C);
Aeq=zeros(2*n,n*n);
for i=1:n
    Aeq(1:n,1+(i-1)*n:i*n)=eye(n,n);
end
for i=1:n
    Aeq(i+n,1+(i-1)*n:i*n)=ones(1,n);
end
beq=ones(2*n,1);
lb=zeros(n*n,1);
ub=ones(n*n,1);
x=linprog(f',[],[],Aeq,beq,lb,ub);
y=reshape(x,n,n); y=y';
y=round(y); sol=zeros(n,n);
for i=1:n
    for j=1:n
        if y(i,j)==1
            sol(i,j)=C(j,i);
        end
    end
end
```

fval=sum(sol(:));

4. Defination of the attractive forces and repulsive force

```
function f_attr = get_attr(Qg,Q,lr)
k = 3; % gain coefficient
d_thre = 6*lr; % attractive force threshold
if distanceCost(Qg,Q) <= d_thre
    f_attr = k*(Qg-Q);
else
    f_attr = d_thre*k*(Qg-Q)/distanceCost(Q,Qg);
end

function f_rep = get_rep(Qg,Qo,Q)
m = 10; % reg force coefficient
n =2;   % capital adjustment
fr1 = [0 0 0];
fr2 = [0 0 0];
for i = 1:size(Qo(:),1)
    Q_thre = distanceCost(Qo(i),Q)/2;
    if distanceCost(Q,Qo(i)) <= Q_thre
            fr1=fr1+m*(1/distanceCost(Q,Qo(i))-
            1/Q_thre)*(distanceCost(Q,Qg)^n)/(distanceCost(Q,Qg)^2)...
                    *(Q-Qo(i))/distanceCost(Q,Qo(i));

        fr2=fr2+m*n/2*((1/distanceCost(Q,Qo(i))-                                   -
        1/Q_thre)^2)*(distanceCost(Q,Qg)^(n-1))...
                  *(Qg-Q)/distanceCost(Qg,Q);
    end
 end
 f_rep = fr1+fr2;
```

5. Main program

```
lr = 0.5;
iter_max = 600;
Q = ground_pos;
Qg = cell(row,row);
for i = 1:row*row
     if mod(distribute(i),row^2) ~= 0
          xi = mod(distribute(i),row^2);   % xi=ground plane code
     else
          xi = row*row;
     end
     yi = (distribute(i)-xi)/(row^2)+1; % yi=target plane code
    Qg{xi} = target_pos{yi};
```

```
end

State = cell(1,iter_max);
for j = 1:iter_max
        State{j} = Q;
       for i =1:row*row
              f_attr = get_attr(Qg{i},Q{i});
              Qo = ground_pos{[1:i-1,i+1:end]};
              f_rep = get_rep(Qg{i},Qo,Q{i});
              Q{i} = Q{i} + lr*(f_attr+f_rep)/norm([f_attr,f_rep],2);
        end
       lr = lr*0.99;
end

err = zeros(1,iter_max);
for j =1:iter_max
        for i = 1:row*row
                       err(j) = err(j)+distanceCost(State{j}{i},Qg{i});
        end
end
%**************************************************************%
% visualization
% iteration process
for j = 1:5:row*row
        for i = 1:5:iter_max
                plot3( State{i}{j}(1),State{i}{j}(2),State{i}{j}(3),'g-' )
                hold on
        end
end
% fianl position
for j = 1:row*row
        plot3(State{end}{j}(1),State{end}{j}(2),State{end}{j}(3),'k+','LineWidth',5)
        hold on
end
```