For office use only	Team Control Number	For office use only
T1 ________	8040	F1 ________
T2 ________		F2 ________
T3 ________	Problem Chosen	F3 ________
T4 ________	A	F4 ________

2017

HiMCM

Summary Sheet

The problem of sky display with Drone can be interpreted as designing visible and safety locus of drone lights to form image of three different objects, which are Ferries wheel, dragon, and a self-designed pattern (a smiling face) respectively. In this paper, we successfully developed a few comprehensive mathematical models to determine the number of drones required and model their initial location and the flight paths mathematically in both static and animate state.

We addressed this problem mainly through space coordinate frame which allowed us to formulate the motions of all the drones in all animations. In the first part, we designed the Ferris wheel as a composition of several geometrical shapes, a circle with spokes and two identical triangles. Considering the space coordinate of every points and using the idea of matrix, we constructed the model based that on the thought of substituting every points own coordinate with its consecutive former one in the same circular motion path.

In the second part, we made a two dimensional dragon that has detailed features such as head, tail, claw, and scale. We made the body of the dragon as the shape of sin(x), and the movement model was created, enabling the dragon to move smoothly and periodically during the animation.

For the third part, our team decided to create a calm face that would eventually turn into a smile face. We made a few functions to express straight lines that represent expressionless eyes and mouth were bent to a smile face by changing the coordinates of the points in the straight line to coordinates of points that fit in parabolas..

After finishing all three models, our team gave out the evaluation for each of them with MATLAB programming. Based on the calculation, we determined to use up to 427 drones to create three sky displays. And the required launch area would take up 9m×9m, and the air space of 100m in length, 50m in width and 80m in height was required. The duration for the light show would be 30 minutes.

In summary, our mathematical model could successfully determine the requirements for this aerial light show.

2017
20th Annual High School Mathematical Contest in Modeling (HiMCM)
Summary Sheet

Team number: 8040 Problem chosen: A

Abstract

The problem of sky display with Drone can be interpreted as designing visible and safety locus of drone lights to form image of three different objects, which are Ferries wheel, dragon, and a self-designed pattern (a smiling face) respectively. In this paper, we successfully developed a few comprehensive mathematical models to determine the number of drones required and model their initial location and the flight paths mathematically in both static and animate state.

We addressed this problem mainly through space coordinate frame which allowed us to formulate the motions of all the drones in all animations. In the first part, we designed the Ferris wheel as a composition of several geometrical shapes, a circle with spokes and two identical triangles. Considering the space coordinate of every points and using the idea of matrix, we constructed the model based that on the thought of substituting every points own coordinate with its consecutive former one in the same circular motion path.

In the second part, we made a two dimensional dragon that has detailed features such as head, tail, claw, and scale. We made the body of the dragon as the shape of sin(x), and the movement model was created, enabling the dragon to move smoothly and periodically during the animation.

For the third part, our team decided to create a calm face that would eventually turn into a smile face. We made a few functions to express straight lines that represent expressionless eyes and mouth were bent to a smile face by changing the coordinates of the points in the straight line to coordinates of points that fit in parabolas..

After finishing all three models, our team gave out the evaluation for each of them with MATLAB programming. Based on the calculation, we determined to use up to 427 drones to create three sky displays. And the required launch area would take up 9m×9m, and the air space of 100m in length, 50m in width and 80m in height was required. The duration for the light show would be 30 minutes.

In summary, our mathematical model could successfully determine the requirements for this aerial light show.

Keywords: drone, aerial light show, Ferris wheel, dragon, MATLAB programming

A letter to the Mayor

Dear Mr. Mayor：

Using a cluster of drones to perform a choreographed aerial light show will be a stirring as well as an environmentally friendly “firework show” for the enormous citizens. From your requirement, our team successfully determined the number of drones needed and the flight route of the three figures: Ferries wheel, dragon, and a self-designed pattern (a smiling face). Moreover, we evaluated the safety and the requirement of the performance site and space.

We treated the first pattern of ferries wheel as three parts：two triangles along with a circle and spokes (wheel) in the middle of them. In order to form a desirable viewing angle for the audiences to enjoy the performances, the plane of the pattern presented a 45° angle with respect to the horizon. We obtained the spin model of the wheel by using several rotation mathematical formulas. The total number of drones for demonstrating the ferries wheel was 427.

In the second part, we made a two dimensional dragon that has detailed features such as head, tail, claw, and scale. We made the body of the dragon as the shape of sin(x), and the movement model is created, enabling the dragon to move smoothly and periodically. To illustrate the figure of dragon, only 363 drones in total were needed.

The third pattern that we designed was similar to a dynamic emoji. At first, the pattern was just a poker face, the eyes and the mouth were just three straight line segments. By applying movement models we designed, the poker face would turn to a smiling face. The total number of drones needed was 240.

Based on the calculation, our group decided to use up to 427 drones to create three possible sky displays. And the required launch area would took up 9m×9m, and the air space of 80m as height, 50m as width, and 100m as length was required. The duration for the light show would be 30 minutes.

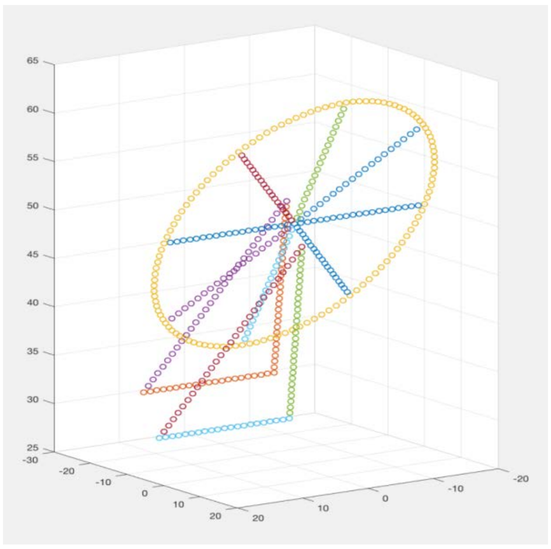
To sum up, we have already determined the prerequisites for this aerial light show, and we recommend that this show to be performed for its beautiful and magnificent visual effects. Attached on the next page is the schematic diagram of the drone clusters as sky displays. It is a pleasure for us to design such a wonderful event and hope the aerial light show will hold soon!

Yours sincerely,

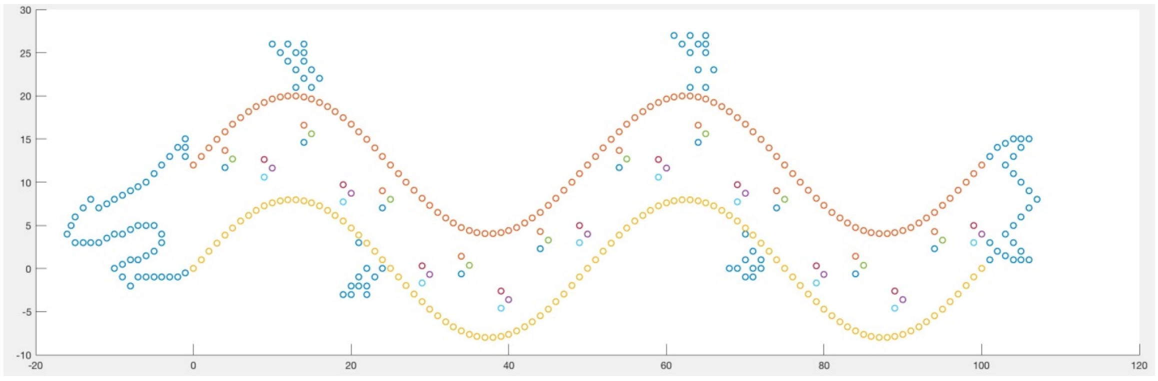
Team#8040

Schematic diagram

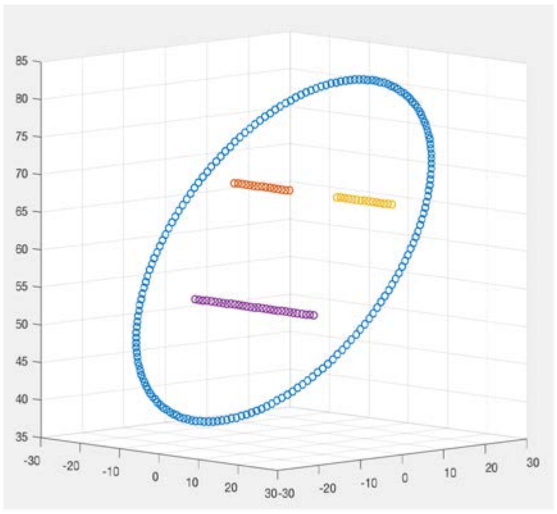
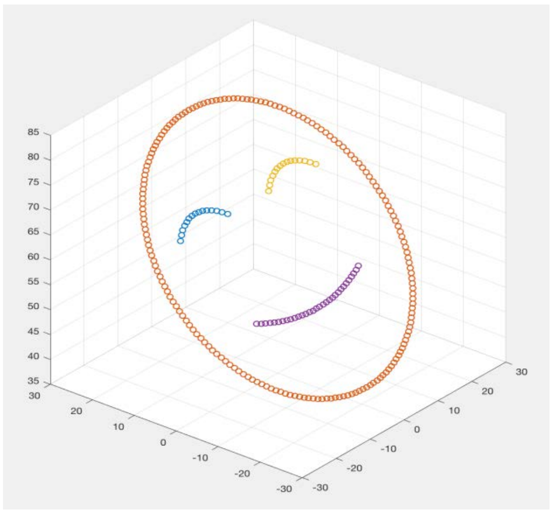
Display 1：Ferris Wheel



Display 2：Dragon



Display 3：Smile face

Catalogue

Abstract……………………………………………………………..1

Aletter to the Mayor………..…………………………………..2

1. Restatement…………...………………………………………...5

2. Assumption& Justification……………………...…....….…5

3. Definition of Variables ……………………...…......…...…7

4. Math Modeling…………..…………………………………8

4.1 Math Modeling of the Ferris wheel………………..…....... **8**

4.2 Math Modeling of the Dragon………………..…………....**16**

4.3Math Modeling of the Smile face………………………..…...**25**

5. The requirements for the 3-display light show …………… 28

6. Evaluation of the Model……..…………………………….30

7. Conclusion………..………………………………………..…...32

8. Reference……………………..……………………………..….33

9. Appendix……………..…………………………………..…..34

1. Restatement

Because Intel has succeeded in making a beautiful choreographed light show by using a cluster of 500 drones, our city have decided to add an aerial light show due to the annual festival. The mayor asked our team to create three possible sky displays by using drones.

The problem of Sky Display with Drone can be interpreted as designing visible and safety locus of drone lights to form image of three different objects, which are respectively Ferris wheel, Dragon and an image created by our team. The core of the problem solving is to determine the number of drones required and model their initial location and the flight paths mathematically in both static and animate state. Besides the core, there are several realistic problems for our team to solve, such as required launch area, required air space, safety considerations, and duration of the aerial light show.

2. Assumption& Justification

Assumption 1: We treat the figures as two-dimensional patterns, but several layers might exist.

Justification 1: Though making the whole aerial light show in a three-dimensional way might be cool and beautiful; it is really hard to achieve effect. Even the only example the question gave failed to make the patterns in the video to stand up and become three-dimensional. There are, however, layers of the pattern in the video, so we decided to make the light show similar to them.

Assumption 2: The angle of elevation for viewing the sky light displays is 45°.

Justification 2: From the video, we find a fact that their images are facing down toward the audience. In order to show the best part of our show and reduce our difficulty of calculation, we decide to estimate the angle of elevation for the light show is 45°.

Assumption 3: For each drone in the figures, the air flow interference caused by neighboring drones can be neglected.

Justification 3: According to our research about the drones used in Intel's aerial light show, the maximum tolerable wind speed in GPS mode is 10m/s, which means that the normal days wind will not affect our light show, so we think that we can neglect the air flow interference. If the wind speed at the day of our light show is greater than 12m/s, we would cancel our light show and find another day to do the sky displays.

Assumption 4: The average flying altitude of the aerial light show patterns is 50meters.

Justification 4: After looking up some information on the internet, we find that normal fireworks can fly to a height about 50m above the ground. Since our display should be similar to a fireworks display, we think an average of 50m above the ground will bring us the best visual effect by the show.

Assumption 5: The safety distance between the drones is 1 meter.

Justification 5: By asking professional staffs in DJI-Innovations Company and estimating the distances between drones of the Intel's light show, we get the conclusion that drones would be safe enough when they are one meter apart from each other, though they normally flies 3 meter apart from each other.

3. Definition of Variables

Symbol	Definitions
n	The number of drones on the circle of the Ferris wheel
r	The radius of the Ferris wheel
θ	The angle between the radius of the circle which that drone is on and the +y axis (Spinning clockwise)
c	The distance from one point on the radius to the center of the Ferris wheel at the position θ = 0, 1/4π, 1/2π, 3/4π
m	The range of the drones' x-axis which act as the base line of the triangle
d	The range of the drones' y-axis which act as the side boundary of the triangle
(x_c, y_c, z_c)	The center coordinate of the circle
t	The amount of time for the drones to be on position
$x_1(t)$	The location of the drone on x-axis
$y_1(t)$	The location of the drone on y-axis
$z_1(t)$	The location of the drone onz-axis
$v_1(t)$	The speed of the drone on x-axis
$v_2(t)$	The speed of the drone on y-axis
$v_3(t)$	The speed of the drone on z-axis
v(t)	The vector that shows the actual speed of the drone
T	The duration time of the displays
N_{total}	The total number of drones required for the sky display

4. Math Modeling

4.1 Model of the Ferris wheel

4.1.1 Definition the coordinate of the drone location

In the first part of the question, we need to find the way initial location for each drone device that will result in the sky display of a static image. In order to achieve this, we should determine the initial location and flight path of one drone, and then we can apply them to the rest of the drone.

If the initial coordinate of the drone is (a, b), and the final coordinate of the drone is (x, y), the direction of the flight can be represented by θ which is the angle between +y axis and the direction that the drone is flying(goes in clockwise) in the coordinate system (Figure 1).

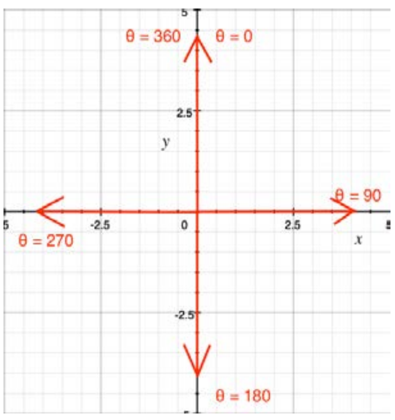


Figure 1. The Definition of the drone location coordinate

With the information provided above, the direction of the flight can be represented easily.

If x-a > 0 and y-b > 0, $\theta_{initial} = 0$, $\theta_{final} = \theta_{initial}$ + arctan[(x-a)/(y-b)];

If x-a > 0 and y-b < 0, $\theta_{initial} = 90$, $\theta_{final} = \theta_{initial}$ + arctan[(b-y)/(x-a)];

If x-a < 0 and y-b < 0, $\theta_{initial} = 180$, $\theta_{final} = \theta_{initial}$ + arctan[(a-x)/(b-y)];

If x-a < 0 and y-b > 0, $\theta_{initial} = 270$, $\theta_{final} = \theta_{initial}$ + arctan[(y-b)/(a-x)].

To make an image of a Ferris wheel, we need to separate the Ferris wheel to three different parts: ***the wheel, the triangle support, and the spokes***, which looks just like that plot shown below (Figure 2). Black represents for the wheel, red for the spokes, and pink for the triangle, while the center coordinate of the circle is (x_c, y_c, z_c) .

In order to make our Ferris wheel more stereoscopic, we decided to put a triangle on each side of the Ferris wheel. The rest of the modeling will be divided into three parts, each for a category.

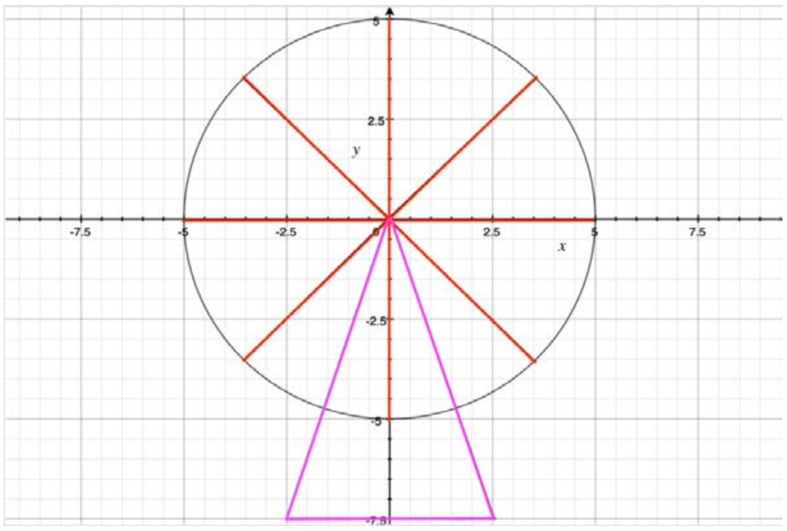


Figure 2. The plot of the Ferris wheel to three different parts: the wheel, the triangle support, and the spokes

(1) *Wheel*

In order to make a model that fits into any situation, we use the angle θ to represent varies conditions. Assuming the radius of the circle is r, so for any point on the circle, its coordinate is

$$(x + \sin\theta * r, y + \cos\theta * r, z)$$

And because we are using drones to form a wheel, the $\Delta\theta$ between them is constant, so the variables θ can be represented in the way:

$$\theta = 0 + \Delta\theta * n$$

$$(0 \leq n < 2\pi/\Delta\theta, \text{ n is an integer})$$

Here, ***n*** can represent the number of the drones.

Finally, in order to let people have a better view of the Ferris wheel, it has to tilt at an angle of 45 degrees(just like the drone light show held by Intel) based on the line y = x, so the final coordinate on the wheel is

$$\begin{cases} x = x_c + r\sin\theta \\ y = (\, y_c + r\cos\theta) \times \frac{\sqrt{2}}{2} \\ z = \, z_c + (y_c + r\cos\theta) \times \frac{\sqrt{2}}{2} \end{cases}$$

(2) ***Spokes***

In the Ferris wheel, there are eight radius, or four diameters acting as the spokes of the wheel. Assuming the distance from one point on the radius to the center of the circle is ***c*** , the coordinates of the point on the spokes can also be represented by *c* and *θ*, which is:

*(x+sinθ*c, y+cosθ*c, z)*

(-19 < *c* < 19, *c* ≠ 0 ; *θ* = 0, 1/4π, 1/2π, 3/4π in four different situation)

Then, in order to enhance people's viewing experience, we title the spokes at an angle of 45 degree base on the line y = x, and the resulting coordinate is

$$\begin{cases} x = x_c + c\sin\theta \\ y = (y_c + c\cos\theta) \times \frac{\sqrt{2}}{2} \\ z = z_c + (y + c\cos\theta) \times \frac{\sqrt{2}}{2} \end{cases}$$

(3) ***Triangle***

At the bottom of the Ferris wheel, there should be a triangle stand that supports the Ferris wheel. Assuming the range of the drones' x-axis which act as the base line of the triangle is *m*, and the range of the drones' y-axis which act as the side boundary of the triangle is *d*, we will make two triangle supports on each side of the Ferris wheel, and the coordinates of it can be represented as:

$$\begin{cases} x = x_c + m \\ y = \left(y_c - \frac{3}{2}r\right) \times \frac{\sqrt{2}}{2} - \frac{3\sqrt{2}}{2} \\ z = z_c + \frac{3\sqrt{2}}{2} - \frac{\sqrt{2}}{2}(\frac{3}{2}r - y_c) \end{cases} \qquad \begin{cases} x = x_c + m \\ y = \left(y_c - \frac{3}{2}r\right) \times \frac{\sqrt{2}}{2} + \frac{3\sqrt{2}}{2} \\ z = z_c - \frac{3\sqrt{2}}{2} - \frac{\sqrt{2}}{2}(\frac{3}{2}r - y_c) \end{cases}$$

(-1/2*r* ≤ *m* ≤ 1/2*r*)

$$\begin{cases} x = x_c + \frac{1}{3}d \\ y = (y_c + d)\times\frac{\sqrt{2}}{2} - \frac{3\sqrt{2}}{2} \\ z = z_c + \frac{3\sqrt{2}}{2} - \frac{\sqrt{2}}{2}(-y_c - d) \end{cases} \qquad \begin{cases} x = x_c + \frac{1}{3}d \\ y = (y_c + d)\times\frac{\sqrt{2}}{2} + \frac{3\sqrt{2}}{2} \\ z = z_c - \frac{3\sqrt{2}}{2} - \frac{\sqrt{2}}{2}(-y_c - d) \end{cases}$$

($-3/2r<d \le 0$)

$$\begin{cases} x = x_c - \frac{1}{3}d \\ y = (y_c + d)\times\frac{\sqrt{2}}{2} - \frac{3\sqrt{2}}{2} \\ z = z_c + \frac{3\sqrt{2}}{2} - \frac{\sqrt{2}}{2}(-y_c - d) \end{cases} \qquad \begin{cases} x = x_c - \frac{1}{3}d \\ y = (y_c + d)\times\frac{\sqrt{2}}{2} + \frac{3\sqrt{2}}{2} \\ z = z_c - \frac{3\sqrt{2}}{2} - \frac{\sqrt{2}}{2}(-y_c - d) \end{cases}$$

($-3/2r <d < 0$)

4.1.2 Initial location of the drones for the Ferris wheel

Now, we suppose the i^{th} drone has the location(x_{i0}, y_{i0}, z_{io}) when t=0, and the Ferris wheel is supposed as a perpendicular to the plane "*xoy*". The radius of the circle is *r*, and the center of a circle is (x_c, y_c, z_c). Then we get the circle function:

$$(x-x_c)^2+(y-y_c)^2+(z-z_c)^2=r^2$$

In order to be easy to calculate, we set the space rectangular coordinate system to be parallel to the plane "*yoz*", while the center of the circle is on the *z*-axis. In this case, the circle would tangent to the origin of coordinate, so that the abscissa is now 0, thus we can get:

$$\begin{cases} (x-x_c)^2+(y-y_c)^2+(z-z_c)^2=r^2 \\ x=0 \end{cases}$$

Then

$$x_c^{\ 2}+(y-y_c)^2+(z-z_c)^2=r^2$$

Because there are total *n* drones on the circle, we will uniformly set these drones, while there will be a drone arranged at $\frac{2\pi}{n}$ interval, just like the following graph (Figure 3) :

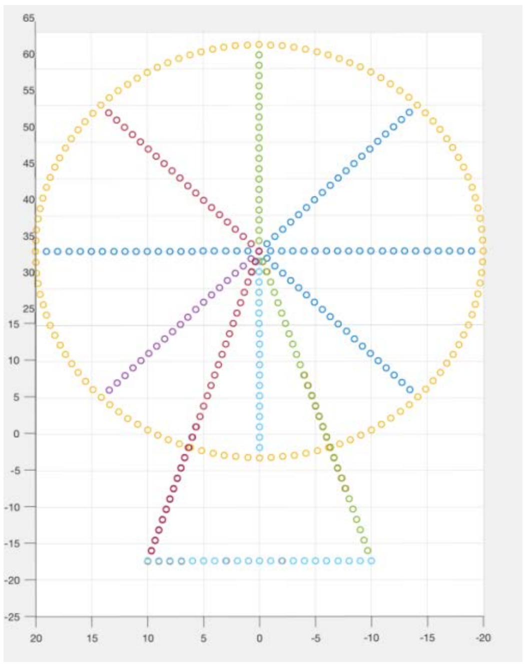


Figure 3. The graph of initial location of the drones for the Ferris wheel

The corresponding coordinates are:

$$\begin{cases} (0, y_1, z_1) \\ (0, y_2, z_2) \\ \vdots \\ (0, y_n, z_n) \end{cases}$$

Since the following coordinate can be obtained by spinning the previous coordinate. From the rotation formula, we get:

$$(y_k, z_k)^T = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} (y_{k-1}, z_{k-1})^T, k = 2,3,\ldots,\mathrm{n}, \theta = \frac{2\pi}{n}$$

Which is the same as:

$$(y_k, z_k)^T = \begin{bmatrix} \cos(\frac{2\pi}{n}) & \sin(\frac{2\pi}{n}) \\ -\sin(\frac{2\pi}{n}) & \cos(\frac{2\pi}{n}) \end{bmatrix} (y_{k-1}, z_{k-1})^T, k = 2,3,\ldots,\mathrm{n}$$

Since:

$$\begin{cases} {x_c}^2 + (y_{k-1} - y_c)^2 + (z_{k-1} - z_c)^2 = r^2 \\ {x_c}^2 + (y_k - y_c)^2 + (z_k - z_c)^2 = r^2 \end{cases}$$

Then we can get:

$$\begin{cases} x_c^2 + y_{k-1}^2 - 2y_{k-1}y_c + y_c^2 + (z_{k-1} - z_c)^2 = r^2 \\ x_c^2 + y_k^2 - 2y_k y_c + y_c^2 + (z_k - z_c)^2 = r^2 \end{cases}$$

$$\begin{cases} y_{k-1}^2 - 2y_{k-1}y_c + y_c^2 + (z_{k-1} - z_c)^2 + x_c^2 - r^2 = 0 \\ x_c^2 + y_k^2 - 2y_k y_c + y_c^2 + (z_k - z_c)^2 - r^2 = 0 \end{cases}$$

$$\begin{cases} y_{k-1}^2 - 2y_{k-1}y_c + y_c^2 + (z_{k-1} - z_c)^2 + x_c^2 - r^2 = 0 \\ y_k^2 - 2y_k y_c + y_c^2 + (z_k - z_c)^2 + x_c^2 - r^2 = 0 \end{cases}$$

$$\begin{cases} y_{k-1}^2 - 2y_{k-1}y_c + y_c^2 + (z_{k-1} - z_c)^2 + x_c^2 - r^2 = 0 \\ y_k^2 - 2y_k y_c + y_c^2 + (z_k - z_c)^2 + x_c^2 - r^2 = 0 \end{cases}$$

$$y_{k-1} = \frac{2y_c \pm \sqrt{(2y_c)^2 - 4(y_c^2 + (z_{k-1} - z_c)^2 + x_c^2 - r^2)}}{2}$$

$$y_k = \frac{2y_c \pm \sqrt{(2y_c)^2 - 4(y_c^2 + (z_k - z_c)^2 + x_c^2 - r^2)}}{2}$$

Assuming the initial value of the iteration is $(0, y_1, z_1)$, which can arbitrarily select a point on the circle, and then the sequential of points on the circle is numbered in clockwise. Finally, the location of all drones can be solved by constantly iterating.

4.1.3 The flight path of each drone for the Ferris wheel

Assume the number of drones we required on the circle of the Ferris wheel is *n*, and the "i^{th}" drone is expressed as a_i. We take a point in the real space as the original point, and we can establish a space rectangular coordinate system based on that point. In this situation, the location for drone $a_i$in time *t* is$(x_i(t), y_i(t), z_i(t))$, and the duration time for the display is T.

Then the curvilinear equation that represents the path for the drone from the launch area to the final exhibit position is:

$$\begin{cases} x = x_1(t) \\ \mathrm{y} = \mathrm{y}_1(t), 0 \le t \le T_1 \\ \mathrm{z} = \mathrm{z}_1(t) \end{cases}$$

Because the process of flying is continuous, $x_i(t), y_i(t)$ and $z_i(t)$is considered to be continuous function.

Moreover, we assume the location of one drone at the initial time *t* is $(\mathrm{x},\mathrm{y},\mathrm{z})$, and then the velocity vector of the drone is $\vec{v}(t)=(v_1(\mathrm{t}),v_2(\mathrm{t}),v_3(\mathrm{t}))$, $v_1(\mathrm{t}),v_2(\mathrm{t}),v_3(\mathrm{t})$ is the speed for the drone in *x, y,* and *z* axis respectively. We can get the magnitude of the velocity:

$$\left|\vec{v}(t)\right|=\sqrt{v_1(t)^2+v_2(t)^2+v_3(t)^2}$$

According to the relationship between distance and velocity, we get:

$$\begin{cases}\dfrac{dx}{dt}=v_1(t)\\ \dfrac{dy}{dt}=v_2(t), \quad 0\le t\le T_1\\ \dfrac{dz}{dt}=v_3(t)\end{cases}$$

4.1.4 Animate rotation of the Ferris wheel

According to the rotation formula, when a unit circle is rotating clockwise in a plane, the coordinates before rotating is $(x_i(t_1),y_i(t_1))$, and the point after rotating is $(x_i'(t_1),y_i'(t_1))$, the rotating angle is *θ*. Then we have:

$$(x_i'(t_1),y_i'(t_1))^T=\begin{bmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{bmatrix}(x_i(t_1),y_i(t_1))^T$$

So:

$$\begin{cases}x_i'(t_1)=x_i(t_1)\cos\theta+y_i(t_1)\sin\theta\\ y_i'(t_1)=-x_i(t_1)\sin\theta+y_i(t_1)\cos\theta\end{cases}$$

$$\begin{cases}\Delta x=x_i(t_1)(\cos\theta-1)+y_i(t_1)\sin\theta\\ \Delta y=-x_i(t_1)\sin\theta+y_i(t_1)(\cos\theta-1)\end{cases}$$

Because its in a three-dimensional space, we can apply the formula and get:

$$\begin{cases}y_i'(t_1)=y_i(t_1)\cos\theta+z_i(t_1)\sin\theta\\ z_i'(t_1)-z_0=-y_i(t_1)\sin\theta+(z_i(t_1)-z_0)\cos\theta\end{cases}$$

$$\Delta_2(x_i(t_1),y_i(t_1),z_i(t_1))=(0,y_i(t_1)(\cos\theta-1)+z_i(t_1)\sin\theta),-y_i(t_1)\sin\theta+(z_i(t_1)-z_0)(\cos\theta-1)-z_0$$

So the total increment is:

$$\Delta(x_i(t_1),y_i(t_1),z_i(t_1))=\Delta_1(x_i(t_1),y_i(t_1),z_i(t_1))+\Delta_2(x_i(t_1),y_i(t_1),z_i(t_1))$$

As a result:

$$(x_i(t_1+\Delta t), y_i(t_1+\Delta t), z_i(t_1+\Delta t)) = (x_i(t_1), y_i(t_1), z_i(t_1))$$

Accordingly, the Ferris wheel can be able to spin with incrementally iterate the function.

4.1.5 Examination of the model of the Ferris wheel

In the first display, we would like to build a Ferris wheel with the height of the wheel 50m, radius of 20m. We assume the radius of the circle *r*, the hypotenuse of the triangle *d*, and the base of the triangle *m* is 20m, 30m, and 20m respectively. Because the safety distance is 1meter, we can calculate the drones needed to form the Ferris wheel based on the calculations:

(1) The amount of drones for the circle: $2\pi \times 20 \approx 120$;

(2) The amount of drones for the spoke: $19 \times 8 + 1 = 153$;

(3) The amount of drones for one triangle: $20 + 2 \times 30 - 3 = 77$;

Combining all these drones together, we can get:

The amount of drones for the Ferris wheel: $77 \times 2 + 120 + 153 = 427$

We determined the center of the wheel to be (0, 0, 50), and using the formula provided previously we can calculate all the coordinates for the drones, and make a 3-D version of the Ferris wheel with MATLAB programming (shown in Figure 4).

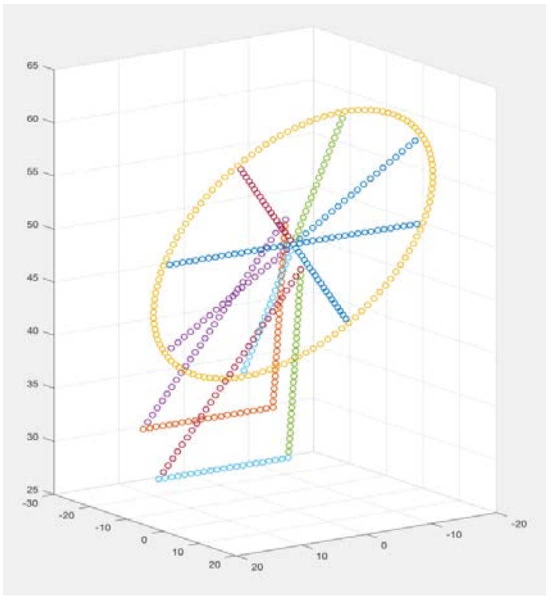


Figure 4. The 3-D graph of the Ferris wheel with MATLAB programming

4.2 Math Modeling of the Dragon

4.2.1 Introduction of Chinese Dragon

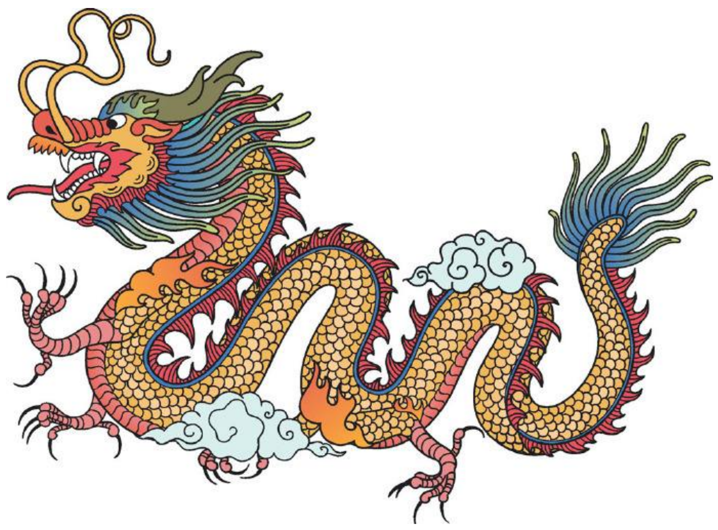


Figure 5. The picture of Chinese dragon

The above figure 5 shows Chinese dragon, which symbolizes dignity and imperial authority. In the second part of the aerial light show, the drones will roughly form an image of Chinese dragon.

Display 2 requires us to design a pattern that demonstrates the figure of dragon to the audiences. In order that the aerial light show performs dynamically and vividly, we separate the dragon’s body into five parts: ***the body lines, scales, claws, head and tail***.

4.2.2 The math model of the Dragon

To illustrate the Dragon display accurately, we establish a rectangular plane coordinate system, assuming the junction point of boundary of the lower side of the

body (neck) and the head as origin (0,0) (this point can be seen as throat if we view it from the anatomical point of view) and the unit on each axis is meter (m). After splitting the dragon into five parts, we begin to analyze the figure respectively, utilizing functions to approximate the position of each drone in the cluster.

(1) ***Body Lines***

The traditional body figure of Chinese dragons is similar to sine function, as we have seen from the above picture. Therefore, we approximate the body lines of it by using two sine functions that are the same in amplitude and period, but different in y-coordinate for an identical x-input:

$$\left\{\begin{array}{l} y = 8\sin(\frac{\pi x}{25}) + 12 \\ y = 8\sin(\frac{\pi x}{25}) \end{array}\right\}.$$

From the equations above, the period for each function is 50. We cut out 2 periods of the functions between x=0 and x= 100 to simulate the body lines of the functions. To form a pair of "parallel" sine graphs, 100 drones at each side of the body boundary are needed. For the sake of visual effect, the gap between two successive drones should be as small as possible to make the image full and continuous.

However, we must obey the rule: the safe distance between two drones is 1m. Therefore, by letting two successive drones keep a safe distance of 1m in the horizontal distance, the drone cluster can demonstrate the best visual effect; at the same time, the flight of the drones can be deemed as a safe flight.

(2) ***Scales***

To present the figure, there must be at least three drones to represent one piece of scale. In our design, the position of three drones can be seen as three vertexes of an isosceles right triangle: the drone lies at the top and the bottom left vertex are parallel to each other, while another vertex is located to the right of these two vertexes. Since the outline of scales is roughly a piece of arc, which is a closed curve, the spacing between two neighboring drones should also be as small as possible to make the pattern seem continuous.

In consideration of the safe distance and the complexity of calculation process, the distance between the right drone and the top left or the bottom left drone should both be $\sqrt{2} \approx 1.414m$, and thus the horizontal distance and vertical distance between them are equal to 1m.

Except for determining basic single pattern of a piece of scale, how to place the single piece of scale inside dragon's body need to be determined. In order to express the position of each single scale, we assign the drone located at the right corner as the datum point. All datum points can be represented with two functions:

$$\begin{cases} y = 8\sin(\frac{\pi x}{25}) + 4 \\ y = 8\sin(\frac{\pi x}{25}) + 8 \end{cases}$$

First, we divide the vertical distance between upper boundary and the lower boundary of the body into three equal parts, the length of each part is 4m, which is equivalent to shift the lower body line $\left(y = 8\sin(\frac{\pi x}{25})\right)$ upward by 4 units and 8 units. The distance between each datum point is 5 meters; however, in consideration of the safety, successive scales cannot locate on the same graph. Therefore, when x=5, the scale locates on the function $y = 8\sin(\frac{\pi x}{25}) + 8$; when n=10, the scale locates on the function $y = 8\sin(\frac{\pi x}{25}) + 4$, etc.

Using mathematical language to describe the location of the datum points can be represented by the following:

$$\begin{cases} y = 8\sin(\frac{\pi x}{25}) + 4 \left\{x < 100 \mid x = 10n + 10, n \in N\right\} \\ y = 8\sin(\frac{\pi x}{25}) + 8 \left\{x < 100 \mid x = 10n + 5, n \in N\right\} \end{cases}$$

Based on the datum point (right corner of the isosceles right triangle), the coordinate of top left corners and the bottom left corners can also be determined easily by shifting the function where the datum points located on. For the top left vertex, the new function representing it can be obtained by shifting the original function 1 unit upward and one unit leftward:

$$\begin{cases} y = 8\sin(\frac{\pi(x+1)}{25}) + 5\left\{x < 100 \mid x = 10n + 10, n \in N\right\} \\ y = 8\sin(\frac{\pi(x+1)}{25}) + 9\left\{x < 100 \mid x = 10n + 5, n \in N\right\} \end{cases}$$

For the bottom left vertex, the new function representing it can be obtained by shifting the original function 1 unit downward and 1 unit leftward:

$$\begin{cases} y = 8\sin(\frac{\pi(x+1)}{25}) + 3\left\{x < 100 \mid x = 10n + 10, n \in N\right\} \\ y = 8\sin(\frac{\pi(x+1)}{25}) + 7\left\{x < 100 \mid x = 10n + 5, n \in N\right\} \end{cases}$$

(3) *Claws*

For the dragon claws, the expected visual effect is that the claws are tightly attached to the body of the dragon and they do not need to be flexible. There are 4 claws which affiliate to the dragons are arranged in a relatively symmetrical way -- two on one side and two on the other side.

In order to achieve the animation of the claws that they are tightly attached to the body, our team constructs the model of the claws to be relative fixed points to the nearest point of the body, which means it is nevertheless a combination of several moving points relative to the sky.

For instance, one of the position where our team decided to place a claw is at (12, g(12)), which is for the claw at the upper front. Since the expected width of the arm is approximately 2m, the other point that helps establish the arm is at (14, g(14)). As for the rest of the points that compose that arm and that claw, their coordinates are recorded in the table (labeled UF, for upper front).

For calculation convenience, the x-y coordinate of the points composing the arm are taken to the nearest whole numbers. Similarly, every point is assigned a serial number and a particular coordinate(Lower Front -- LF, Upper End -- UE, Lower End -- LE). The total number of drone need for the dragon paws is 48.

Table1. The serial number and the x-y coordinate of the dragon claws

Serial Number	x-coordinate	y-coordinate	Serial Number	x-coordinate	y-coordinate
UF-01	13	21	UE-01	63	21
UF-02	15	21	UE-02	65	21
UF-03	14	22	UE-03	64	23
UF-04	16	22	UE-04	66	23
UF-05	13	23	UE-05	63	25
UF-06	15	23	UE-06	65	25
UF-07	12	24	UE-07	62	26
UF-08	14	24	UE-08	64	26
UF-09	11	25	UE-09	65	26
UF-10	10	26	UE-10	61	27
UF-11	13	25	UE-11	63	27
UF-12	12	26	UE-12	65	27
UF-13	14	25	LE-01	70	4
UF-14	14	26	LE-02	71	2
LF-01	21	3	LE-03	73	2
LF-02	24	0	LE-04	70	1
LF-03	22	0	LE-05	72	1
LF-04	21	-1	LE-06	68	0
LF-05	23	-1	LE-07	69	0
LF-06	22	-2	LE-08	71	0
LF-07	21	-2	LE-09	72	0
LF-08	20	-2	LE-10	70	-1
LF-09	22	-3	LE-11	71	-1
LF-10	20	-3			
LF-11	19	-3			

(4) *Head*

The dragon head is a sophisticated image which roughly delineates the 2-dimensional outline of the dragon. However, since it is the head of the dragon, it should be stable as a real dragon. Therefore, the image of the head can be presented as planned configuration of drones. Therefore, the points with assigned coordinates are listed below to form an image of the dragon's head. <u>The total number of drone required for head is 49.</u>

Table2. The serial number and the x-y coordinate of the dragon head

Serial Number	x-coordinate	y-coordinate	Serial Number	x-coordinate	y-coordinate
H-01	0	12	H-26	-10	4
H-02	-1	13	H-27	-9	4
H-03	-1	14	H-28	-8	4.5
H-04	-1	15	H-29	-7	5
H-05	-2	14	H-30	-6	5
H-06	-3	13	H-31	-5	5
H-07	-4	12	H-32	-4	4
H-08	-5	11	H-33	-4	3
H-09	-6	10	H-34	-5	2
H-10	-7	9.5	H-35	-6	1.5
H-11	-8	9	H-36	-7	1
H-12	-9	8.5	H-37	-8	1
H-13	-10	8	H-38	-9	0.5
H-14	-11	7.5	H-39	-10	0
H-15	-12	7	H-40	-9	-1
H-16	-13	8	H-41	-8	-2
H-17	-14	7	H-42	-7	-1
H-18	-15	6	H-43	-6	-1
H-19	-15.5	5	H-44	-5	-1
H-20	-16	4	H-45	-4	-1
H-21	-15	3	H-46	-3	-1
H-22	-14	3	H-47	-2	-1
H-23	-13	3	H-48	-1	-0.5
H-24	-12	3	H-49	0	0
H-25	-11	3.5			

(5) ***Tail***

The dragon tail is source of power to bring the dragon into motion, and it is the part which moves the most sharply. To keep it simple, the tail is a symmetrical image. To achieve the most optimum animation that can generate a both understandable and require the least complex calculation, our team make the tail of the dragon oscillate up and down periodically.

To be specific, all the single particles are designed to wave up and down periodically. Therefore, similar to the claws, every single drone should be assigned a coordinate. The total number of drone required for the tail is 26.

Table3. The serial number and the x-y coordinate of the dragon tail

Serial Number	x-coordinate	y-coordinate	Serial Number	x-coordinate	y-coordinate
T-01	101	13	T-14	106	7
T-02	102	14	T-15	105	6
T-03	103	14.5	T-16	104	5
T-04	104	15	T-17	103	4
T-05	105	15	T-18	104	3
T-06	106	15	T-19	105	2
T-07	105	14	T-20	106	1
T-08	104	13	T-21	105	1
T-09	103	12	T-22	104	1
T-10	104	11	T-23	103	1.5
T-11	105	10	T-24	102	2
T-12	106	9	T-25	101	3
T-13	107	8	T-26	100	1

4.2.3 Examination of the model of the Dragon

Based on the calculations and information above, in order to construct the dragon figure the total number of drones needed is 383. The calculation process is shown here:

$$N_{total}=N_{body\ lines}+N_{scales}+N_{claws}+N_{head}+N_{tail}$$
$$=100*2+3*20+48+49+26$$
$$=383$$

Then we used MATLAB programming to create the initiated dragon image that we estimate to happen shown in figure 6.

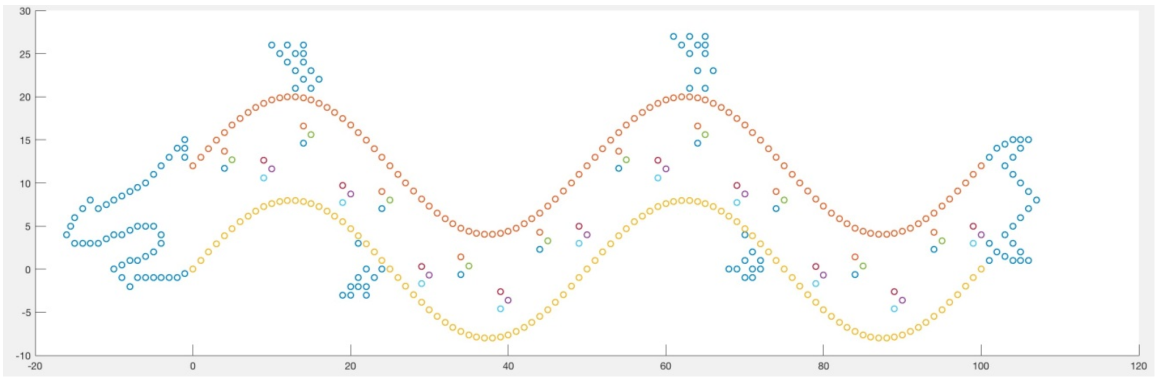


Figure 6. The 2-D graph of the Dragon with MATLAB programming

4.2.4 The movement of drones of the Dragon

Since the dragon is made up of hundreds of drones, every single drone plays an important role in determining animation of the dragon image. Therefore, a way to describe the motion of drones, which are regarded as particles, is needed. A wavelike periodic function is appropriate to describe the motion.

Consider the shape of the dragon and its nature of movement which is like stirring the air beside it in a shape of "S". So this effect can be accomplished by wavelike motions of all the particles on the body. The initiated dragon image is shown on the Figure 6, which means the animation starts from this static state. Wavelike up-and-down motion is a kind of harmonic motion. To begin with, period is the first required to know. A particle that is in a perpendicular harmonic motion has the same average perpendicular velocity of the same particle that is running around a perpendicular circle in a radius same as the range of the harmonic motion with the perpendicular velocity of the former.

Therefore, as shown by the Figure 6, the period of the motion of one particle can be determined by the circumference of the circle drawn by the particle divided by the tangential velocity, the same as the largest perpendicular velocity described above. Given the maximum speed of the drones 3m/s and the radius of circle 12m, period of the motion is $\mathrm{T_{min}} = \frac{2\pi r}{\mathrm{v}} = 8\pi(s)$. Since the minimum period is known, T=32s is confirmed to be the period of motion for all of our particles.

For the Figure 7, the function of v_y can be presented as $v_{\mathrm{by}}(\mathrm{t}) = -v \sin \omega \cdot t$,

where v_x is the perpendicular velocity, v is the tangential velocity, ω is the angular velocity and t is the time elapsed.

From $\mathrm{T} = 32\mathrm{s}$ we are able to calculate $\omega = \frac{2\pi}{\mathrm{T}} = \frac{\pi}{16}$. The v can be determined by the circumference of the circle divided by the period decided by us, so $v = \frac{2\pi r}{T} = 0.75\pi(m/s)$.

The function of the perpendicular velocity of each particle is rewritten as

$$v_y(t) = -0.75\pi \sin\frac{\pi}{16}t$$

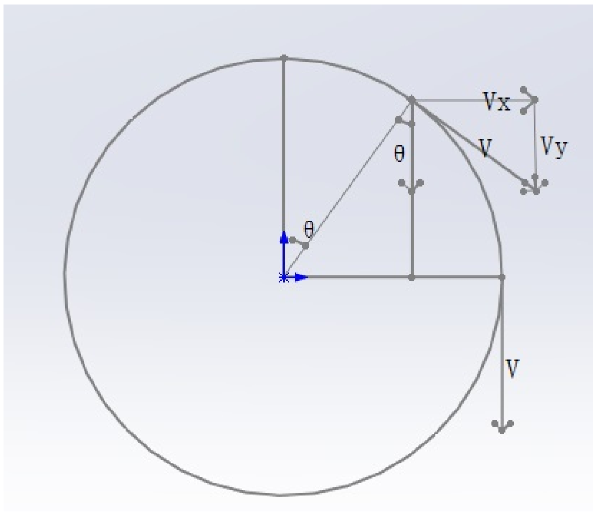


Figure 7. The graph of the perpendicular velocity of each particle

Our team decide to divide the dragon body into 2 separate parts(the main part of the body and the front part of the body) in order to make the dragon body to wave without breaking in the middle. The function $v_y(t) = -0.75\pi \sin\frac{\pi}{16}t$ is for the particle at x=12.5, however, there is no drone at x=12.5 because all the drones are assigned integer x-coordinate. But all the points on sine function of dragon's body are closely related. One whole period of the function can be divided into 50 as the number of drones contained in one period. As waves proceed, the y-coordinate of consecutive particles are related in uniform time difference. So the velocity of particles behind can be determined by the function

$$v_{ny}(t) = -0.75\pi \sin[\frac{\pi}{16}(t - \frac{32(x-12.5)}{50})], \quad x \in [13,107], \quad x \in I .$$

When all the particles in the domain [13,107] oscillates in the velocity described by the function, the main part of the body is animated.

For the particle in the interval [1, 12], the wave cannot be naturally represented by sine function because the head of the dragon does not move at all. Therefore, it is designed as a waving string that is fixed at one end. The particles on the string are also going to move up and down periodically. The velocity of particle in this interval can be determined by $v = \frac{2\pi f(x)}{\mathrm{T}} = \frac{\pi f(x)}{16}$.

So the function of the perpendicular velocity is

$$v_{ay}(t) = -\frac{\pi\left|8\sin(\frac{\pi}{25}\mathrm{x})\right|}{16}\sin(\frac{\pi}{16}t),\quad \mathrm{x}\in[1,12],\quad x \in I .$$

When all the particles waves as the function describes, the front part of the body is animated.

In conclusion, the velocities of particles can be described by the following function:

$$\begin{cases} v_{ay}(t) = -\frac{\pi\left|8\sin(\frac{\pi}{25}\mathrm{x})\right|}{16}\sin(\frac{\pi}{16}t), x\in[1,12], x\in I \\ v_{\mathrm{by}}(t) = -0.75\pi\sin[\frac{\pi}{16}(t - \frac{32(\mathrm{x}-12.5)}{50})], x\in[13,107], x\in I \end{cases},$$

Where x is the x-coordinate of each drone and t=0s at the initial of the animation, which also means all the drones have $v_x = 0\mathrm{m/s}$.

4.3 Math Modeling of the Smile Face

4.3.1 Initial location of the drones for the Smile face

In this image, the outline of the face can use the same equation as the wheel in the Ferris wheel. Accordingly, the coordinate of the point on the contour of the face can be represented as

$(x+\sin\theta*r, y+\cos\theta*r, z)$, which $\theta = 0 + \Delta\theta*n$,

($0 < n \leq 360$, n is an integer, $\Delta\theta = 360/n_{total}$).

Also, the coordinate of the eyes can be represented as

$(x + n, y+1/3*r, z)$

(n is an integer, $-20 \leq n \leq -6$ for the left eye, and $6 \leq n \leq 20$ for the right eye).

Finally, the coordinate of the mouth can be represented as

$(x + m, y - 1/3*r, z)$ (m is an integer, $-15 \leq m \leq 15$)

For better viewing experience, we title the smiling face by 45 degree, and the equation result in:

$$\begin{cases} x = x + r\sin\theta \\ y = (y + r\cos\theta) \times \frac{\sqrt{2}}{2} \\ z = z + (y + r\cos\theta) \times \frac{\sqrt{2}}{2} \end{cases} \qquad \begin{cases} x = x + n \\ y = (y + 1/3 * r) \times \frac{\sqrt{2}}{2} \\ z = z + (y + 1/3 * r) \times \frac{\sqrt{2}}{2} \end{cases} \qquad \begin{cases} x = x + m \\ y = (y - 1/3 * r) \times \frac{\sqrt{2}}{2} \\ z = z + (y - 1/3 * r) \times \frac{\sqrt{2}}{2} \end{cases}$$

$\theta = 0 + \Delta\theta * n$, ($0 < n \leq 360$, n is an integer, $-20 \leq n \leq -6$, m is an integer, n is an integer, $\Delta\theta = 360/n_{total}$for the left eye, and $6 \leq n \leq 20$, $-15 \leq m \leq 15$ for the right eye.)

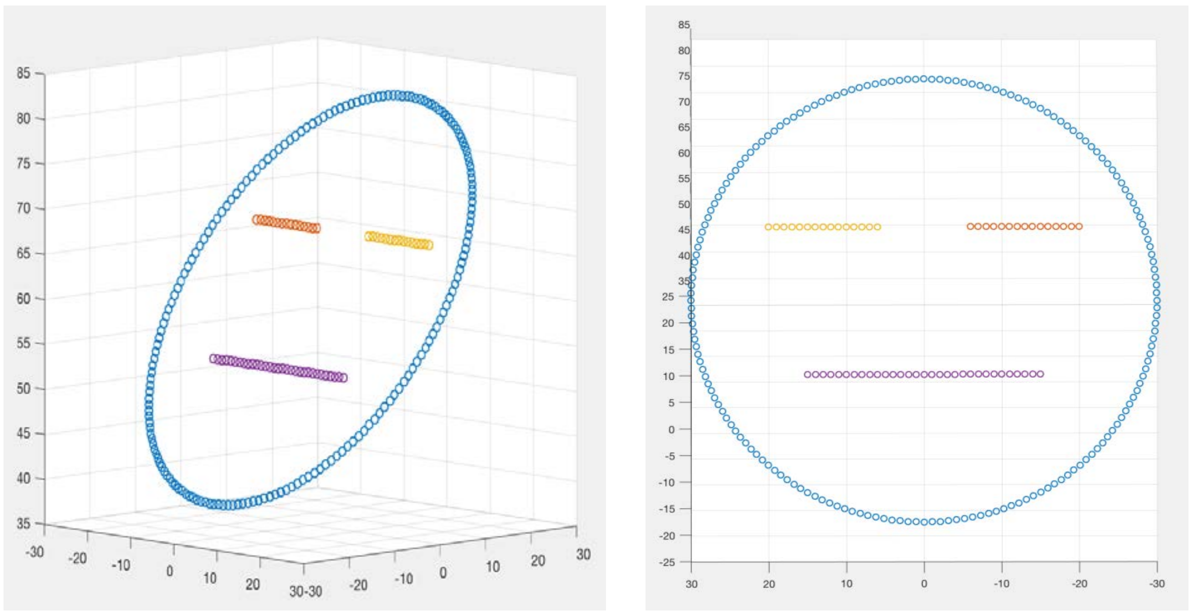


Figure 8. Initial location of the drones for the Smile face with with MATLAB

4.3.2 Animate rotation of the Smile face

For the movement from the normal face to the smile face, we need to have the final coordinate that result in the smiling face. In order to achieve this, we can turn the eyes of the face into two quadratic equations using vertex equation, that the vertex of them are (-13,13),(13,13). And we can also apply it to the mouth, whose vertex is (0,-13). As a result, the coordinate of the left eye after tilted for 45 degree is

$$\begin{cases} x = x + n \\ y = (y - 3/49 \times (n + 13)^2 + 13) \times \frac{\sqrt{2}}{2} \\ z = z + (y - 3/49 \times (n + 13)^2 + 13) \times \frac{\sqrt{2}}{2} \end{cases}$$

($-20 \leq n \leq -6$, n is an integer)

Same for the right eye:

$$\begin{cases} x = x + n \\ y = (y - 3/49 \times (n - 13)^2 + 13) \times \dfrac{\sqrt{2}}{2} \\ z = z + (y - 3/49 \times (n - 13)^2 + 13) \times \dfrac{\sqrt{2}}{2} \end{cases}$$

($6 \leq n \leq 20$, n is an integer)

Which can also be applied for the mouth:

$$\begin{cases} x = x + n \\ y = (y + \dfrac{3}{225} n^2 - 13) \times \dfrac{\sqrt{2}}{2} \\ z = z + (y + \dfrac{3}{225} n^2 - 13) \times \dfrac{\sqrt{2}}{2} \end{cases}$$

($-15 \leq n \leq 15$, n is an integer)

And then we get:

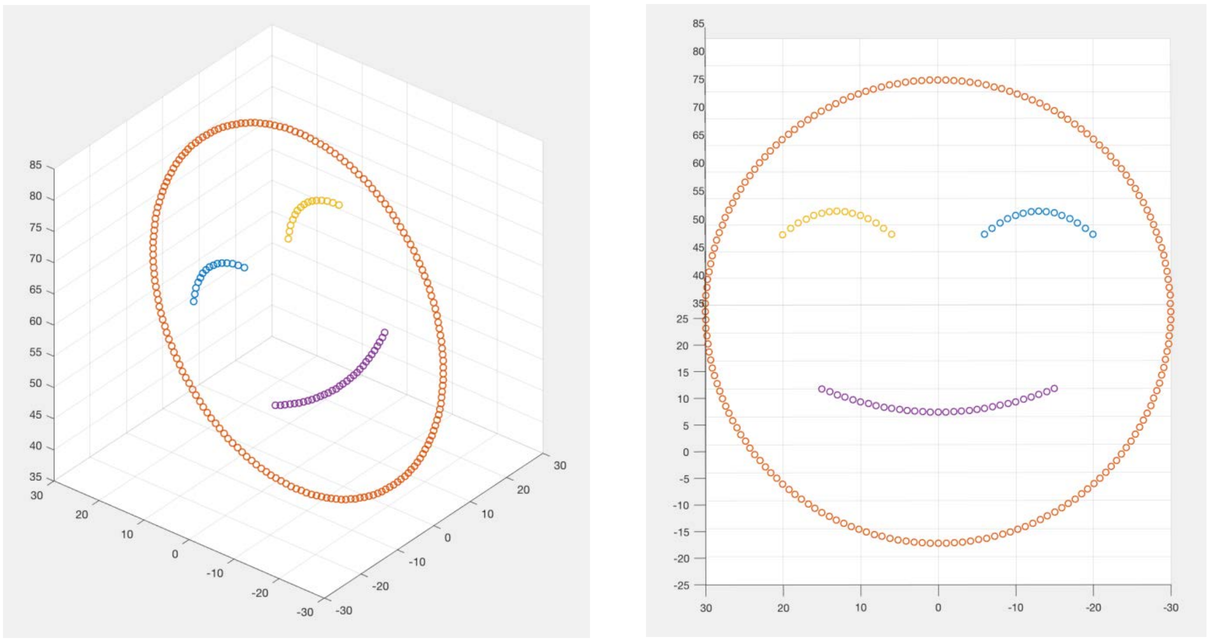


Figure 9. Animate rotation of the Smile face with MATLAB

Also, the movement is really easy, drones acting as eyes and mouth just need to move to the new position with the same x coordinate, so it is easy to categorize and control.

4.3.3 Examination of the model of the Smile face

Because the smile face consist of a face, a mouth, and two eyes, the total amount of drones needed for making the smile face is simply adding them up. We decided to make a smile face with radius of 30m, so the drones needed for making the face is 180.

Also, each eye will need 15 drones, and the mouth need 30 drones. Adding them up, we need a total of 240 drones for making the smile face.

Then we used MATLAB programming to create the smile face image shown in Figure 8 and 9.

5. The requirements for the Drone Clusters Lightshow

5.1 The number of drones, required launch area and air space

Because too many of drones would easily cause problems for controlling, it would become too big for audiences to see. After discussion and calculation, our group decides to use no more than 427 drones to create three possible sky displays.

For the required launch area, we decide to take up an area of 9m×9m, which is 81 m^2 as the launching platform. If the coordinate of the central square is (0,0), which is just under the center of the wheel (0,0,50). There will be 6 drones stack on each square, so the total amount of drones is 486, enough for the Ferris wheel which only need 427 drones. In this case, we would be able to launch 486 drones, which can satisfy our need for each model. The required air space is 50m×50m×50m for the Ferris wheel, 20m×20m×100m for the dragon, and 50m×50m×50m for the smiling face. Combining these information, we would need an air space of 80m as height, 50m as width, and 100m as length.

5.2 The launching process

For each square area, we will pile 6 drones together, and let them fly one by one. Also, for the launching process for the Ferries wheel, to avoid the drones crashing into each other, the drones in the upper layer will be marked with a smaller number. And the first 77 drones will first fly and form the upper triangle, while the drones marked 78 - 197 will fly secondly and form the wheel. Then those marked 198 - 350 will go up thirdly and form the spokes. Finally, those marked 350 - 427 will form the triangle in front, and the beautiful Ferris wheel was made. For the returning trip of the drones, it’s just the opposite sequence of the departure.

5.3 The duration time for the light show and safety considerations

We expect the duration for the light show is 30 minutes, and each of the displays will last for 1 minute until they switch to another pattern. Once all three models has displayed, which takes three minutes. This sequence will repeat for 10 times. By giving out a plan like this, audiences would be able to see all three models in a short time, so that they would not be bored. And they would be able to see the model they like in a short time even if they miss it for one time.

For safety considerations, we consider 1 m as the safety distance between drones based on the research on the parameter of a type of Intel's drone. Besides, we expect audiences to be 80 m away from the center of the light show, so that in this case, the light show will be directly facing to them. Moreover, if one drone get out of control and falls down, the audiences would be safe in that distance.

In Summary, the requirements for the Drone Clusters Lightshow are listed below:

Table 4. The requirements for the Drone Clusters Lightshow

Displayshow	number of drones	required launch area	air space	duration of the show
1. Ferries wheel	427	9m×9m	100m×50m×80m (L*W*H)	30 minutes
2. Dragon	383			
3. Smile Face	240			

6. Evaluation of the Model

6.1 Evaluation of the Ferris wheel Model

We believe our model of the Ferris wheel is one of the best, that consist of varies components which will vastly increase our models ornamental value and operational simplicity. Firstly, we tilted our pattern by 45 degree, so people can easily view the beautiful Ferris wheel without holding their head straight upward which is tiring. Secondly, we use two triangles as the support of the Ferris wheel, which makes the Ferris wheel seems to be more as a three-dimensional object because it has three layers. Thirdly, we give every point its unique coordinates, so it is much easier to operate. Having its initial coordinate and final coordinate is enough for it to finish its mission. Finally, we animated the ferries wheel by letting it to spin, which brings the audiences a better experience and as well a surprise.

There are still disadvantages of our model, though, that at first the color of the light of our drones is the same, so it’s a bit hard to differentiate the different parts of the Ferris wheel. Second, we didn’t use the drones to present the seats of the Ferris wheel, although it will be a much bigger project if we decided to do that.

6.2 Evaluation of the Dragon Model

6.2.1 Strengths

Our model gives a clear description of the positions of each drone, constituting the basic profile and features of a traditional Chinese dragon, which ensures inerrably and fluently: The whole model is decomposed into five parts and we analyzed the five parts as separate figures respectively, which suddenly makes a complicated mathematics problem seem relatively easy and logical to solve.

We use equations to derive a set of movement equations in order to let the dragon behaves like a real lively dragon. Instead of using complex velocity equations, we apply simple periodic functions to describe the motion of the drones that consists the dragon.

6.2.2 Drawbacks

Our model is relatively ideal, with no air flow disturbance between neighboring drones. Determining the influence of air flow would be difficult without the accurate data of the drones used for light show.

The positions of drones for head, claws, and tails cannot be simply represented by some brief equations. Instead, we have to use coordinates of several points to denote the positions of these parts.

6.3 Evaluation of the Smile face Model

This pattern is actually designed on purpose, that we hope our drones' show can bring everyone a happy day. I think our smiling face has a few advantages. First, our smiling face is well designed, and everybody can recognize it immediately. Second, it turns in a process of no expression into smile, which increases its ornamental value and makes it more appealing to the audience. Third, the smiling face also tilted at an angle of 45 degree, which again makes the audience more convenient to watch the show.

On the other hand, the smiling face model also has some disadvantages. First, the animation is sort of one-way, that it can only turn from no expression to smile and can't come back, otherwise it is too wired. Accordingly, the time of animation is a bit short, and in another way of saying it, a bit simple. Second, the smiling face is sort of a plane picture, which may not be so ornamental from the side. But after all it is still a successful and meaningful project.

7. Conclusion

Our team is asked to design an aerial light shows of drones. We address this problem mainly through space coordinate frame which allows us to formulate the motions of all the drones in all animations.

For the first show, the Ferris wheel, we firstly designed it as a composition of several geometrical shapes, a circle with spokes and two identical triangles. The two triangles don’t move basically, while we only consider the motion of the spokes and the circle since they are the only dynamic part in this animation. Considering the space coordinate of every points and using the idea of matrix, we construct the model based on the thought of substituting every points own coordinate with its consecutive former one in the same circular motion path. Moreover, we tilted this image by 45 degree angle toward the ground to optimize its visual effect.

In the second part, we utilized a different way to describe the motion. In order to achieve the wavelike motion of dragon body, we define the speed of every moving particle with periodical sine function, including the dragon body and details within, such as scales, claws and the tail.

In the last show which is an unsmiling face changing into a smiling face, the straight lines that represent expressionless eyes and mouth are bent to a smile face by changing the coordinates of the points in the straight line to coordinates of points that fit in parabolas..

After finishing all three models, our team gives out the evaluation for each of them with MATLAB programming. Based on the calculation, our group decides to use no more than 427 drones to create three possible sky displays. And the required launch area takes up 9m×9m, and the air space of 80m as height, 50m as width, and 100m as length is need. The duration for the light show is 30 minutes. This suggests that our mathematical model could successfully solve this problem.

8. Reference

[1] 2017 HiMCM Contest Instructions

http://www.comap.com/highschool/contests/himcm/instructions.html

[2] Intel® Falcon™ 8+ System for Professional Inspection and Surveying Brochure

https://www.intel.com/content/www/us/en/drones/falcon-8-plus-brochure.html

[3] The picture of Chinese Dragon

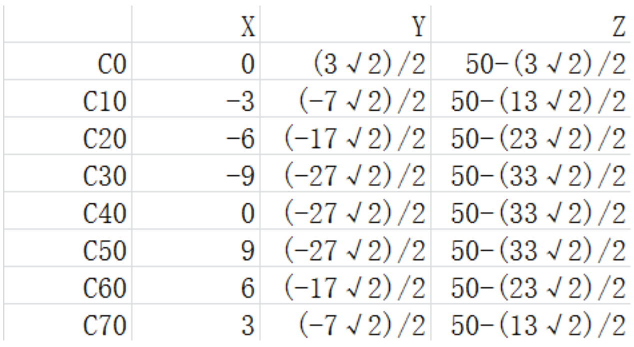
http://www.lanimg.com/shiliang/201504/94403.html

[4] Liu Weiguo. MATLAB programming and application - Second Edition. Higher education press, 2006.

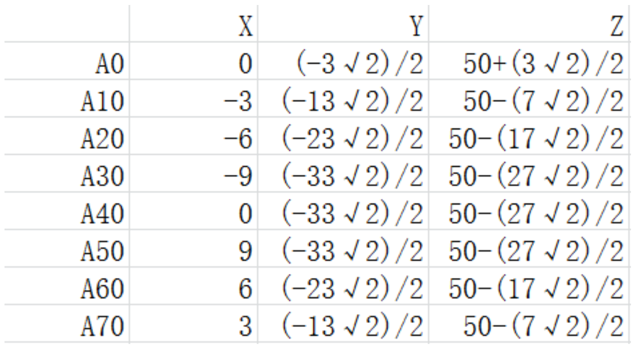
[5] Chuanqi Qin, Ting Wang, Yuanfeng Jin. Study of MATLAB and Its Application in Mathematical Modeling [J]. Modeling and Simulation, 2015, 4(3), 61-71.

9. Appendix

9.1 The Analysis of certain drones in the Ferris wheel

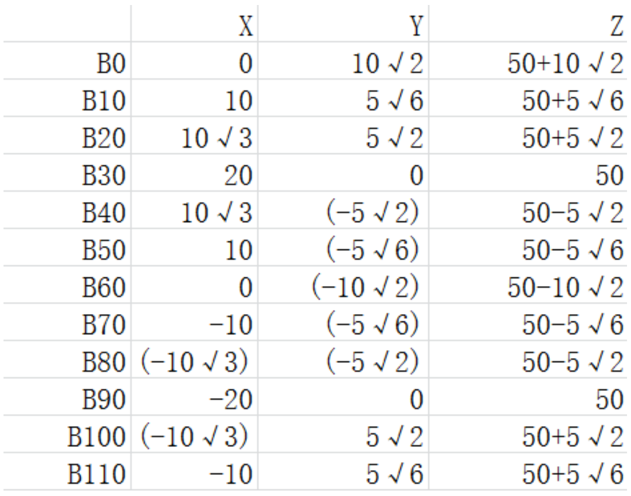


	X	Y	Z
C0	0	$(3\sqrt{2})/2$	$50-(3\sqrt{2})/2$
C10	-3	$(-7\sqrt{2})/2$	$50-(13\sqrt{2})/2$
C20	-6	$(-17\sqrt{2})/2$	$50-(23\sqrt{2})/2$
C30	-9	$(-27\sqrt{2})/2$	$50-(33\sqrt{2})/2$
C40	0	$(-27\sqrt{2})/2$	$50-(33\sqrt{2})/2$
C50	9	$(-27\sqrt{2})/2$	$50-(33\sqrt{2})/2$
C60	6	$(-17\sqrt{2})/2$	$50-(23\sqrt{2})/2$
C70	3	$(-7\sqrt{2})/2$	$50-(13\sqrt{2})/2$



	X	Y	Z
A0	0	$(-3\sqrt{2})/2$	$50+(3\sqrt{2})/2$
A10	-3	$(-13\sqrt{2})/2$	$50-(7\sqrt{2})/2$
A20	-6	$(-23\sqrt{2})/2$	$50-(17\sqrt{2})/2$
A30	-9	$(-33\sqrt{2})/2$	$50-(27\sqrt{2})/2$
A40	0	$(-33\sqrt{2})/2$	$50-(27\sqrt{2})/2$
A50	9	$(-33\sqrt{2})/2$	$50-(27\sqrt{2})/2$
A60	6	$(-23\sqrt{2})/2$	$50-(17\sqrt{2})/2$
A70	3	$(-13\sqrt{2})/2$	$50-(7\sqrt{2})/2$

These two charts show the location of the innermost and outermost triangle, the number after “A” and “C” is the serial number of the drones, which is counted progressive increasing counter-clockwise. I choose to describe the location of the first drone of each ten drone to show the framework of the triangles, which would easily let the mayor know the basic shape of the holder our Ferris wheel.



	X	Y	Z
B0	0	$10\sqrt{2}$	$50+10\sqrt{2}$
B10	10	$5\sqrt{6}$	$50+5\sqrt{6}$
B20	$10\sqrt{3}$	$5\sqrt{2}$	$50+5\sqrt{2}$
B30	20	0	50
B40	$10\sqrt{3}$	$(-5\sqrt{2})$	$50-5\sqrt{2}$
B50	10	$(-5\sqrt{6})$	$50-5\sqrt{6}$
B60	0	$(-10\sqrt{2})$	$50-10\sqrt{2}$
B70	-10	$(-5\sqrt{6})$	$50-5\sqrt{6}$
B80	$(-10\sqrt{3})$	$(-5\sqrt{2})$	$50-5\sqrt{2}$
B90	-20	0	50
B100	$(-10\sqrt{3})$	$5\sqrt{2}$	$50+5\sqrt{2}$
B110	-10	$5\sqrt{6}$	$50+5\sqrt{6}$

This chart shows the framework of outer circle of the Ferris wheel. The numbers after “B” is the series number, counted clock-wise. We decided to use 120 drones to show this circle. That is because in this case, there can be one drone each 3 degree that is turned, which would be helpful for the calculation when we are animating the Ferris wheel.

	X	Y	Z
B0000	0	0	50
B0010	0	5 √2	50+5 √2
B4510	5 √2	5	55
B9010	10	0	50
B13510	5 √2	−5	45
B18010	0	(−5 √2)	50−5 √2
B22510	(−5 √2)	−5	45
B27010	(−10)	0	50
B31510	(−5 √2)	5	55

The spoke of the Ferris is also a hard part in our calculation due to the 45° downward facing to the audience. By giving out these numbers, people who are in charge of preparing the light show can use the corresponding points to locate the lines that should have one drone each meter.

9.2 MATLAB Code for the Ferris wheel

```
a = 0:1/60*pi:2*pib = sin(a)*20
c = cos(a)*20*2^(1/2)/2d = 50+c
e = -19:1:19f = cos(1/2*pi)*e*2^(1/2)/2
g = 50 + fh = sin(1/4*pi)*e
i = cos(1/4*pi)*e*2^(1/2)/2j = 50+ i
k = sin(0)*el = cos(0)*e*2^(1/2)/2
m = 50 + ln = sin(3/4*pi)*e
o = cos(3/4*pi)*e*2^(1/2)/2p = 50 + o
q = -10:1:10r = (-33*2^(1/2))/2
s = 50 + (3*2^(1/2))/2 - 2^(1/2)/2*(3/2*20)
```

```
t = (-27*2^(1/2))/2

u = 50 - (3*2^(1/2))/2 - 2^(1/2)/2*(3/2*20)v = -29:1:0

w = 1/3*vx = v*2^(1/2)-3*2^(1/2)/2

y = 50+3*2^(1/2)/2-2^(1/2)*(-v)z = v*2^(1/2)+3*2^(1/2)/2

z1 = 50-3*2^(1/2)/2-2^(1/2)*(-v)z2 = -1/3*v

z3 = v*2^(1/2)-3*2^(1/2)/2z4 = 50+3*2^(1/2)/2-2^(1/2)*(-v)

z5 = v*2^(1/2)+3*2^(1/2)/2z6 = 50-3*2^(1/2)/2-2^(1/2)*(-v)

scatter3(b,c,d)hold on

scatter3(e,f,g)scatter3(h,i,j)

scatter3(k,l,m)scatter3(n,o,p)

scatter3(q,r,s)scatter3(q,t,u)

scatter3(w,x,y)scatter3(w,z,z1)

scatter3(z2,z3,z4)scatter3(z2,z5,z6)
```

9.3 MATLAB Code for the Dragon

```
c=1:1:100

c= Columns 1 through 16

  1    2    3   4    5   6   7   8    9   10   11   12   13   14   15   16

  Columns 17 through 32

  17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32

  Columns 33 through 48

  33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48

  Columns 49 through 64

  49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64

  Columns 65 through 80

  65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80

  Columns 81 through 96

  81   82   83   84   85   86   87   88   89   90   91   92   93   94   95   96

  Columns 97 through 100
```

```
   97    98    99    100
d=8*sin((pi*c)/25)
e=1:10:100
e =1        11        21        31        41        51        61        71        81        91
f=8*sin((pi*e)/25)+12;
g=10:10:100
g =10        20        30        40        50        60        70        80        90      100
h=8*sin((pi*g)/25)+4;
i=5:10:95
i =5        15        25        35        45        55        65        75        85        95
j=8*sin((pi*i)/25)+8;
k=10:10:100
k =10        20        30        40        50        60        70        80        90      100
k=10:10:100
l=8*sin((pi*k)/25)+5;
m=5:10:95
m= 5        15        25        35        45        55        65        75        85        95
n=8*sin((pi*m)/25)+9;
o=10:10:100
o =10        20        30        40        50        60        70        80        90      100
p=8*sin((pi*o)/25)+3;
q=5:10:95
q =5        15        25        35        45        55        65        75        85        95
r=8*sin((pi*r)/25)+7;
scatter(a,b)
hold on
scatter(c,d)
scatter(e,f)
scatter(g,h)
scatter(i,j)
scatter(k,l)
scatter(m.n)
scatter(o,p)
scatter(q,r)
```

Comments: (a,b) is set of all the points that represent head, paws, and tail.

9.3 MATLAB Code for the Dragon

(1) No Smile Face:

```
face = 0:1/90*pi:2*pi;
facex = sin(face)*30
facey = cos(face)*30*2^(1/2)/2
facez = 60+facey;
scatter3(facex,facey,facez)
hold on
lefteye = -20:1:-6
lefteyex = lefteye;
lefteyey = (lefteyex*0+1/3*30)*2^(1/2)/2;
lefteyez = 60+lefteyey;
scatter3(lefteyex,lefteyey,lefteyez)
righteye = 6:1:20
righteyex = righteye
righteyey = (righteyex*0+1/3*30)*2^(1/2)/2
righteyez = 60+righteyey
scatter3(righteyex,righteyey,righteyez)
mouth = -15:1:15
mouthx = mouth
mouthy = (mouthx*0-1/3*30)*2^(1/2)/2;
```

```
mouthz = 60+mouthy
scatter3(mouthx,mouthy,mouthz)
```

(2) Smiling Face:

```
face = 0:1/90*pi:2*pi;
facex = sin(face)*30
facey = cos(face)*30*2^(1/2)/2
facez = 60+facey;
scatter3(facex,facey,facez)
hold on
lefteye = -20:1:-6
lefteyex = lefteye;
lefteyey = (-3/49*(lefteye+13).^2+13)*2^(1/2)/2;
lefteyez = 60+lefteyey
scatter3(lefteyex,lefteyey,lefteyez)
righteye = 6:1:20
righteyex = righteye
righteyey = (-3/49*(righteye-13).^2+13)*2^(1/2)/2;
righteyez = 60+righteyey
scatter3(righteyex,righteyey,righteyez)
mouth = -15:1:15
mouthx = mouth
mouthy = (3/225*mouth.^2-13)*2^(1/2)/2;
mouthz = 60+mouthy
scatter3(mouthx,mouthy,mouthz)
```