For office use only	Team Control Number	For office use only
T1 ________	8206	F1 ________
T2 ________		F2 ________
T3 ________	Problem Chosen	F3 ________
T4 ________	A	F4 ________

2017
HiMCM
Summary Sheet

Step aside, fireworks - traditional pyrotechnics now have a new competitor for lighting up the night sky! Intel recently performed a 3D light show using 500 LED-carrying drones and captivated the world's attention. The Mayor of our city asked our team to analyze the viability of a potential light show for our annual city festival, which would contain three separate displays: a Ferris wheel, a dragon, and a third display of our choosing, which we elected to be an expanding balloon.

Intel's Shooting Star drones are not commercially available, so we chose the DJI Phantom 3, a mid-priced drone with a long battery life and a pre-programmable GPS control system. The DJI Phantom 3s will be placed in a rectangular grid on the ground before the show for easy set up. We mathematically modeled the paths that each of these drones would take in performing a display demonstration using both MATLAB and Python algorithms. Our drone's paths were split up into three major phases: Transition from Grid to Display, Animation, and Transition from Display to Grid. The model we developed computed the position and velocity of the drones in each of these phases and generated an animation of drone movement for each display.

During the first phase (Transition from Grid to Display), a Python script uses the Hungarian Algorithm to compute the shortest set of paths between the grid location and the initial points of the display. Our model also implements a collision avoidance protocol, which uses the predicted flight paths of the drones to fine-tune their velocities such that no drones collide.

Once the initial display image was formed during the first phase, we made the image dynamic by moving it around the aerial space in the Animation phase. The Ferris wheel's spokes and wheel spin around its center while the entire Ferris wheel rotates around its vertical axis so that the entire crowd can view it. The dragon follows a helical toroid curve, mimicking the traditionally portrayed movement of dragons. The balloon gradually expands as if it is being inflated by using a center of homothety at the bottom of the balloon to scale it up over time. At the end of each animation phase, the drones return to the grid pattern before transforming into the next animation (Transition from Display to Grid).

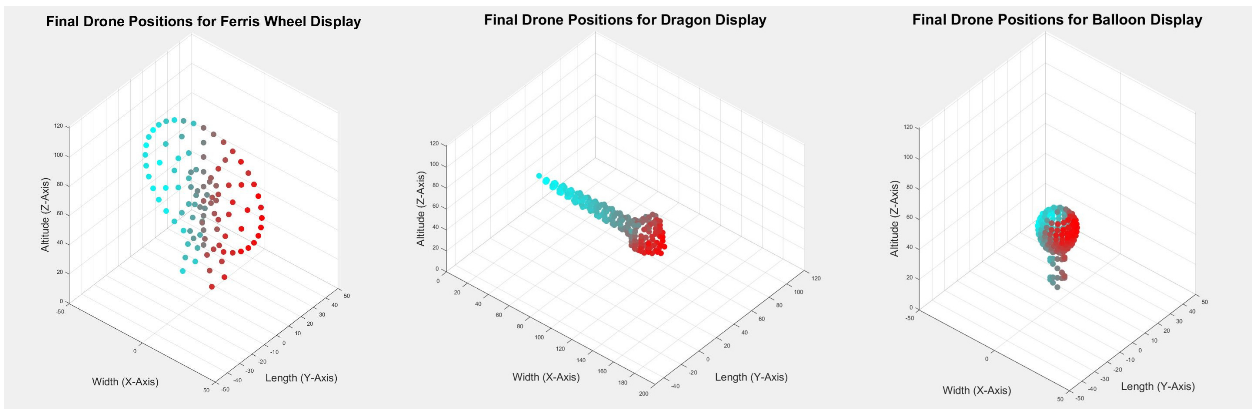
Our model indicates that this 3-display light show requires 196 drones (worth $97,804) and an airspace the size of a right cylinder whose base has a radius of 250 meters and whose height ranges from 10 meters to 115 meters above the ground. The show lasts 9.7 minutes. We recommend that all viewers stand at least 15 meters away in the all directions from the drone-occupied airspace and that drones are only flown when wind speed is less than 11.72 m/s for safety reasons.

Overall, our planned light show uses a variety of mathematical models to create a fluid and dynamic light display without any collisions, and is protected by enough regulations to ensure viewer safety. We've demonstrated that a drone light show is a viable and exciting prospect for our city's festival.

Dear Mayor,

Our team has looked into your proposal of hosting a drone light show for the upcoming annual festival for our city, and believe it would be a fantastic addition to the festival! Of course, drone light shows are a radiant display of cutting edge technology, so our city's residents would be thrilled to have the chance to see one. Beyond spectacular visuals, drone light shows are also extremely beneficial to the city. Drone light shows seem expensive at first: the light show we have planned uses 196 DJI Phantom 3 Standards, which cost a total of \$97,804. Nevertheless, the light show drones are an investment which will probably be profitable in the long-term; large cities like ours normally spend upwards of \$100,000 on annual firework shows. By cutting back on the annual firework show and showing a drone display instead, the city can save some of its firework costs. Unlike fireworks, light show drones are reusable, so the city will continue to save money at every festival where it shows the drone display. Replacing fireworks with drones also has health benefits for the city. Studies have shown that fireworks can increase water and air pollution levels by orders of magnitude. We wholeheartedly recommend the drone show because it will save the city's money, reduce pollution, and most importantly, bring a new exciting attraction to the festival!

We have mathematically modeled a full drone light show containing three animated formations: the Ferris wheel and Dragon you requested, and an expanding balloon of our own choice. Not only does our wheel in the Ferris wheel rotate normally, but the entire Ferris wheel rotates sideways as well to show the complete three dimensions of the drone display. The Dragon is created through carefully placed circular and square drone sections. We programmed the Dragon to move along a helical torus pathway around the display area, resembling a Dragon twisting and turning through the sky. For our final display, we used scalar vectors of homothety to mimic a slowly inflating balloon. The show lasts just under ten minutes, but the drones have a battery life of 22 minutes in flight, so the show could conceivably be repeated twice within the drones' battery life. We've included a diagram below do give you an idea of what the show may look like.



To ensure that the drone fleet moves fluidly through the show without any collisions, we designed an algorithm to plot the optimal paths for each drone such that the entire fleet can move and transform at once without any crashes or hiccups. The algorithm uses a complex optimization algorithm called the Hungarian algorithm to assign each drone in the fleet to a position on the display while minimizing total flight distance and preventing path intersections. Then, the algorithm predicts the flight paths of every drone and adjusts the speed of each drone to prevent any possible collisions. Drones guided by this algorithm are incredibly coordinated and will not crash into each other, unless by a widespread major malfunction or disturbance outside of our control.

Though our model does not take into account wind speed or turbulence, we have taken some precautions to have an idea of the effects they will cause. We have noted that a wind speed of more than 11.72 m/s can cause our drones to fail to follow their paths and potentially crash. As much as we would like to see this show succeed, we advise you to avoid using these drones in harsh wind conditions and consider cancelling the drone show should the weather at the festival not permit it. In addition, we modeled the effects if we were to unfortunately lose contact with on of our drones and have the drone fall uncontrollably. In order to ensure the safety of the viewers under catastrophic circumstances, viewers must stay at least 15 meters away from the display area.

Our launching area for the light show is quite small, only requiring a 45.4 m by 39.4 m area. This launch area is efficient to set up and allows all 196 drones to be launched at once: a team of two could accomplish it in a matter of hours. The show will require an airspace in the shape of a cylinder whose base radius is 250 meters and whose height ranges from 10 m to 115 m from the ground, with a 15 m perimeter around it to ensure viewer safety. Although this is a large area to cover, it is necessary to rope off this area to ensure viewer safety. Because the show is airborne, it could be held over a body of water if a 250 m by 250 m area is unfeasible to clear off on land.

Detailed descriptions and figures of the light show images, animations, computer algorithms, launch requirements, safety analysis, and model testing are available in the attached paper. We hope you will support our proposal and if so, we look forward to seeing the show at the festival!

Sincerely,
Team 8206

1 Introduction

1.1 Problem Restatement

Drone Light Shows have recently captivated the world's attention, and millions of people have been awed by the colorful, dynamic sights present in these demonstrations. Intel® has shown in the past year that automated drones that house LED lights have immense potential to perform spectacular light shows and customized performances [1]. In October 2016, Intel® set the Guinness World Record of largest drone fleet, with 500 drones flying in a custom drone LED light show. In January 2017, 300 of these drones made a reappearance in the Super Bowl LI Half-Time Show with Lady Gaga [2]. While only a few large-scale drone light shows have been conducted, many people are already claiming that this new form of entertainment will "Make Fireworks Obsolete" [3]. The Mayor of our city wishes to provide their residents with this incredible experience, and has requested us to investigate the potential of conducting a drone light show the city's upcoming annual festival.

In order to model, coordinate, and execute a drone light show, there are a multitude of factors to consider. First, and perhaps the most economically important question, is how many drones are needed for the light show; having too few would form an image that is not sharp enough, while using too many is not financially viable.

Next, it is important to determine the initial locations and paths of each of the drones for each of the 3 displays so we know location from which the drones should be launched. We must also take into consideration the launch area, safety regulations, air space, and duration of the light show.

To successfully execute the light show, each drone needs to be autonomously programmed to follow a specific path. More specifically, the drone must be able to fly the given route in time and avoid collisions with other drones on the way to its destination. In this paper, we describe our mathematical modeling of a drone light show which contains a Ferris wheel, a Dragon, and a balloon.

1.2 Analysis of the Problem

Although Intel® established the precedent for drone light shows, the Shooting Star™ LED drones Intel® uses are not available for retail. We reviewed commercially available alternatives to the Intel® Shooting Star™, and propose using DJI's Phantom 3 Standard drone in our light show. It is a mid-priced drone, retailing at $499.00 [4], and its speed, battery life, and control range are fairly standard of drones of similar price ranges. However, what makes the Phantom 3 stand out is its autopilot flight control mode. Before flight, the Phantom 3's path can be pre-programmed into GPS "waypoints," allowing the user to define its position, altitude, speed, and heading at each waypoint [5]. This feature can be used to control the flight paths of every drone in the light show. For our light show, we have mathematically modeled the flight paths of each drone formation. These models can be used to parametrically program the Phantom 3s to autopilot through the formations of the light show. The Phantom 3 also comes equipped with a video camera, which can be easily removed and replaced with a bright LED. Because of its autopilot features and relative inexpensiveness, we believe that the Phantom 3 is the most suitable drone for our purposes.

The light show we designed use a total of 196 drones. As each Phantom 3 costs $499.00, the drone fleet for our show will cost approximately $97,804. This seems like a hefty price for a light show, but in the long term it is an investment that will actually save the city money. Large cities spend enormous amounts of money on their firework shows: in 2016, Houston spent $100,000 on its 4th of July fireworks show [6]. By investing in a drone light show, our city can save money over the next several years by cutting back on fireworks and reusing the light show drones instead. Fireworks are also a hazard to health and safety: a study in the journal *Environmental Science & Technology* found that perchlorate levels in the water of Oklahoma Lake rises up to 1,028 times above background levels within 14 hours of July 4th, and a Chinese study found the air pollution levels in Beijing five times higher than normal during the 2006 Lantern Festival, a celebration in which fireworks are heavily used [7]. By switching to drone light shows, we can save money and reduce pollution in the city by cutting back on fireworks used during the festival.

2 Assumptions and Justifications

- Assumption: The light show occurs during favorable weather conditions for drone flight.
  - Justification: Weather is a serious factor in the safety and viability of drone flights. Intel® engineers have stated they "buffer in some days for environmental factors" for their public light shows drones [8]. We expect the mayor to have scheduled the show during favorable weather and to call off the show if the weather is not suitable for flight.
- Assumption: Drones cannot move faster than 16 m/s horizontally or 3 m/s vertically.
  - Justification: DJI lists these as the maximum Phantom 3 flight speeds [4].
- Assumption: Drones have a maximum acceleration of 3.725 m/s^2.
  - Justification: The Phantom 3 can accelerate from 0 to 40 mph in 4.8 seconds. This is equivalent to 17.88 m/s in 4.8 seconds, or 3.725 m/s^2 [9].
- Assumption: Drone LED lights are visible within the entire arena
  - Justification: The LED lights can be seen clearly in the given video of Intel®'s drone light show from several hundred feet away. We will be using comparable LEDs, so they should be similarly viewable.
- Assumption: Drones can not fly above human viewers and must remain below 121.92 m in altitude.
  - Justification: A fundamental Federal Aviation Administration (FAA) requirement for unmanned aerial vehicles is that they cannot exceed 121.92 m (400 ft) in altitude [10]. In addition, the FAA requires that drones are not flown directly above human viewers for safety purposes [11].

- Assumption: Drones cannot fly within 3 meters of each other horizontally and 1 meters of each other vertically.
  - Justification: The accuracy of the drones' GPS positioning system is $\pm$ 1.5 meters horizontally and $\pm$ 0.5 meters vertically [4], so they must be spaced appropriately to avoid collisions.
- Assumption: The drone light show must last 20 minutes or less.
  - Justification: The Phantom 3 has a battery life of 22 minutes of flight time [4]. Landing the drones after 20 minutes gives a few minutes of buffer time to ensure no drones lose power in flight.
- Assumption: Drones will not greatly interfere with each other drones' flight paths when spaced properly.
  - Justification: Calculating the turbulence and air pressures caused by each individual drone on the air would greatly complicate our model. NASA modeled the air turbulence of drones earlier this year, and used several supercomputers to model it computationally [12]. This is not feasible in our model, so we assume that the drone's auto-pilot will be able to account for any minor air fluctuations during the light show.

3 Problem Analysis

3.1 Initial Drone Positions

The initial position of the drones are crucial in ensuring that the operators place the drones in the proper configuration. While the operators could place the drones in a display-dependent prearranged pattern on the ground to make the model's computation easier, we chose to arrange the drones in a rectangular grid. This results in a slightly more complex flight path computation, but significantly decreases the difficulty associated with setting up the drone and results in less work for the operators. Intel® uses the same technique, so the grid arrangement is a tested and proven method.

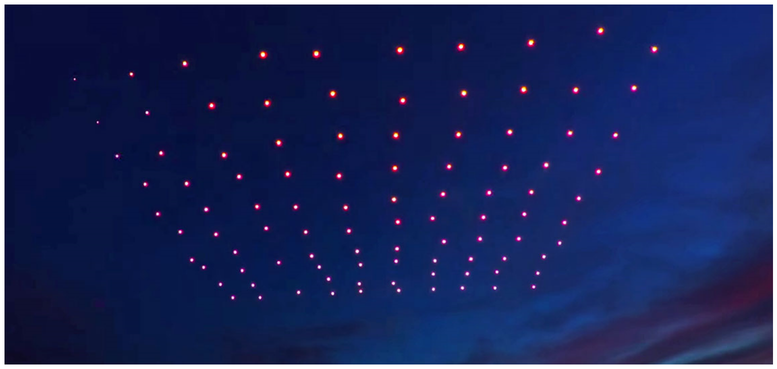


Figure 1: The initial grid arrangement Intel® Shooting Star™ Drones during a 2016 test flight in Palm Springs, California [13]

The DJI Phantom 3 drone has a horizontal GPS hover accuracy is ±1.5 m [4]. Therefore, in order to ensure our drones do not collide into each other because of variations in GPS signal, we spaced the drones out by 3 m in each direction. The drone operators would place the drones on the ground in this grid pattern, and once the drones are turned on, they would hover in this grid pattern before starting the display. Additionally, the drones will return to this initial grid pattern upon completing a display before moving to a subsequent display.

An example grid configuration for the Ferris wheel display is shown below.

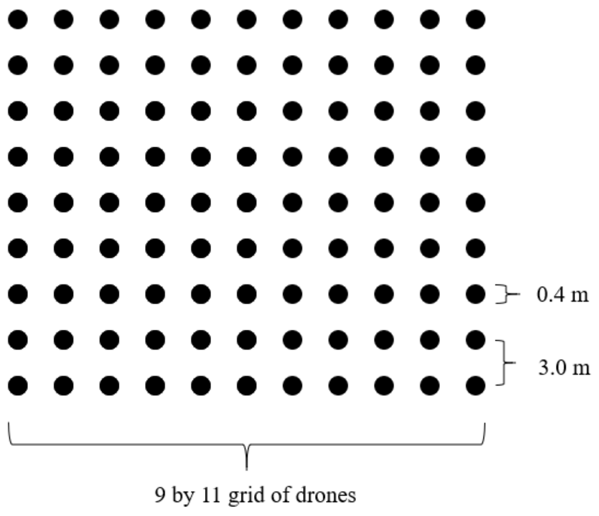


Figure 2: Diagram illustrating the initial positions for drones in Ferris wheel display. 99 drones are arranged in a grid pattern separated by 3.0 m.

3.2 Drone Spatial Range

Due to FFA regulations, we must constrain our drone altitudes to below 121.92 m. While the FAA does not provide a specific requirement for the minimum altitude of drones during public demonstrations, we established a lower altitude limit to ensure complete visibility for all viewers on the ground. We computed this lower altitude limit to be 10 m, as shown by the below illustration.

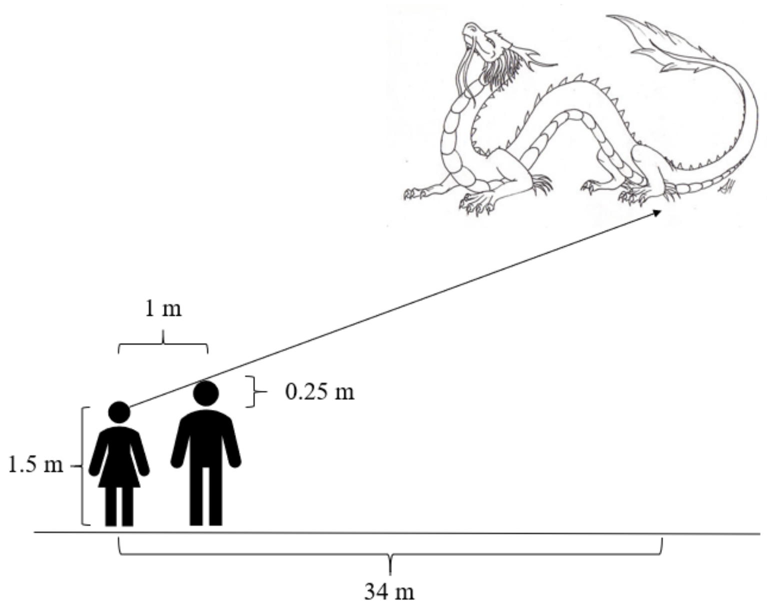


Figure 3: Diagram demonstrating why the drone display must begin at least 10 m. The average male height is approximately 1.75 m and the average female height is 1.5 m. Given that they stand a meter apart and the display is 100 ft away from them, the drone display must be at least 10 m above the ground to allow all viewers to see the entire display.

Thus, our altitude range for operating drones was determined to be 10 m to 121.92 m, approximately 110 meters of vertical space. The lateral range (length and width) of the demonstration arena was different for each display, as these parameters not regulated by the FAA. The DJI Phantom 3 has an FCC-Compliant control range of 5 km, so the range of the drone does not restrict our lateral range at all either [4].

3.3 Number of Drones

We took the approach of attempting to maximize the size of each display in order to appeal to the large number of viewers. The number of drones was determined by using Python software to generate point clouds which resembled each illustration. The number of drones was changed by adjusting the resolution of the pointcloud. Once we determined that the object could most likely be properly identified as a Ferris wheel or as a Dragon, we stopped adding points. The specifics of the drones used in each display are detailed within Sections 5-7.

4 Drone Pathing and Collision Avoidance

Our model splits the drone's path into three major parts for each display. Part 1 is referred to as the *Transition from Grid to Display*, in which the drones move from their static hover state above the ground into their display configuration. Part 2 is referred to as *Animation*, in which the drones move about the area and animate their particular display object to convey movement. Part 3 is referred to as *Transition from Display to Grid*, in which the drones move back to hovering in a grid formation after completing the animation. The algorithms used for parts 1 and 3 are described in general in this section and more specifically for each display in Sections 5-7. Part 2, *Animation*, is display-specific, and is detailed in Sections 5-7.

4.1 Drone Path Mapping

Our model creates a path for each drone from its initial position in the grid to one point on the display formation. Since the shortest path between two points is a linear path, we decided to assign all drones to a linear path to their destination in the display. Our decision to use linear paths is corroborated by the fact that Intel® appears to be using linear paths for their Shooting Star™ drones in public demonstrations [1]. We developed an algorithm in which the sum of the flight path distances was minimized. This shortest linear path method improves efficiency by reducing battery drain.

Minimizing path length not only minimizes transition time, but also contains no intersecting paths, as proven by the Extremal Principle. Consider for the sake of contradiction a mapping between two graphs with two intersecting paths that also minimizes the sum of path lengths, S_m, as shown in Figure 4.

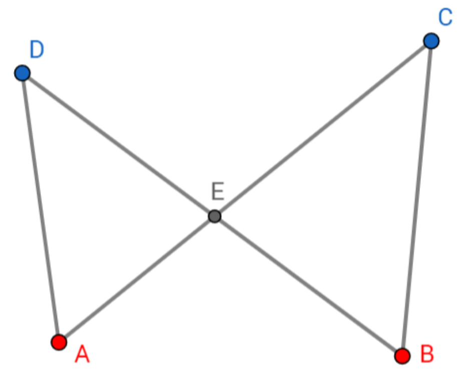


Figure 4: This is a intersection of the paths between red and blue with A traveling to C and B traveling to D.

By the Triangle Inequality,

$$\begin{aligned} AE + DE \geq AD, BE + CE \geq BC \\ So, AE + DE + BE + CE \geq AD + BC \\ AC + BD \geq AD + BC \end{aligned}$$

So if there was an intersection in S_m, then that is not the mapping with the shortest sum of flight paths since connecting A to D and B to C produces a shorter mapping. This contradicts the previous assumption of S_m having a intersection. A successful mapping and computation of shortest sum of paths means that there are no intersection between the paths of the drones. This proof validates our theory that the shortest sum of flight paths method is effective and will contain no intersections, so we proceeded with this method.

Finding the matching of the shortest sum of flight paths is an example of the well known assignment problem. The assignment problem finds the minimum sum to match the points of a weighted bipartite graph together. In the scenario of a drone light show, it is the mapping with the smallest sum of distances between the initial and final points in a formation. We designed a drone pathing algorithm to find the optimal transition mapping and prevent as many drone collisions as possible.

4.2 Computational Model

The Hungarian Algorithm is a common solution to the assignment problem that solves the problem in $\mathcal{O}(n^3)$ time, which is computationally fast enough considering n' is on the magnitude of 100's. A flowchart of the Hungarian Algorithm is show in Figure 5 below.

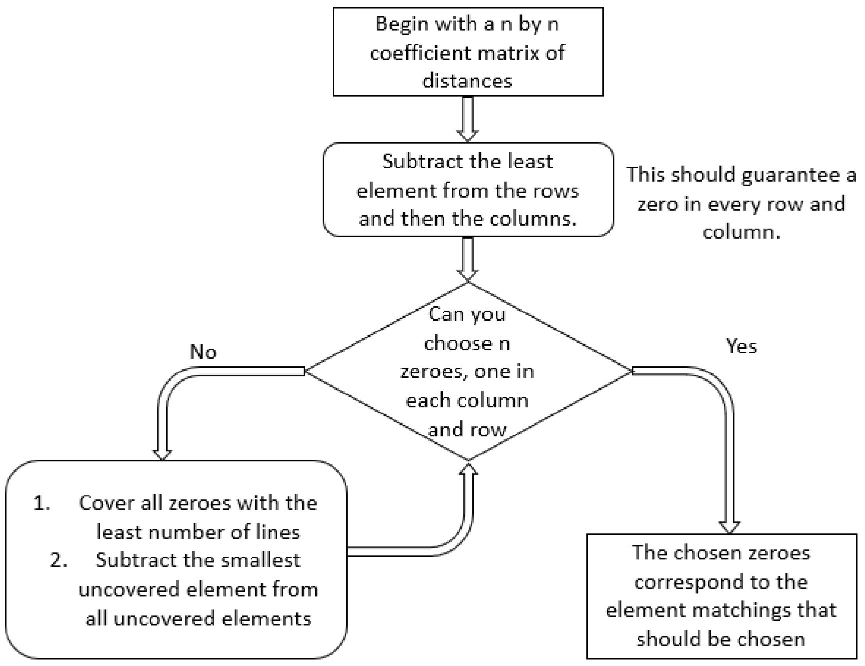

Begin with a n by n coefficient matrix of distances
Subtract the least element from the rows and then the columns.
This should guarantee a zero in every row and column.
Can you choose n zeroes, one in each column and row
No
Yes
1. Cover all zeroes with the least number of lines
2. Subtract the smallest uncovered element from all uncovered elements
The chosen zeroes correspond to the element matchings that should be chosen

Figure 5: The Hungarian Algorithm works by observing the distance matrix and progressively reducing rows and columns, until an optimal solution occurs where each chosen worker and job, there is no better worker to complete this job and no other job the worker could better do.

Although employing the Hungarian algorithm in our solution prevent the drone paths from crossing, the drones occupy a physical space around their respective paths, so the Hungarian algorithm alone is insufficient to prevent collisions. For two drones with closely-intersecting paths to not collide and crash, they must travel down their respective paths at different speeds so they cross the point of incidence at different times. Thus, each drone must move at a specific speed down its path such that the space of no two drones will overlap during the whole transition. To find these speeds, we simulated drone movement during transitions using a Python program. A flowchart of the program is shown in Figure 6 below:

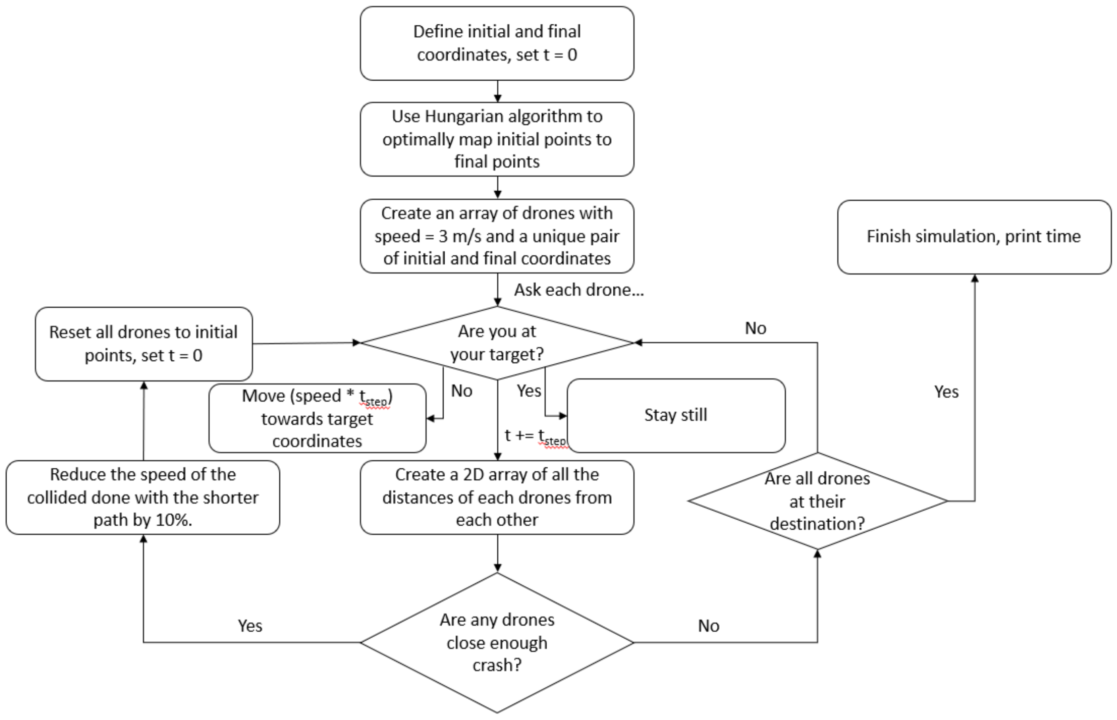

Define initial and final coordinates, set t = 0
Use Hungarian algorithm to optimally map initial points to final points
Create an array of drones with speed = 3 m/s and a unique pair of initial and final coordinates
Ask each drone...
Are you at your target?
No
Yes
Move (speed * t_{step}) towards target coordinates
Stay still
t += t_{step}
Create a 2D array of all the distances of each drones from each other
Are any drones close enough crash?
Yes
No
Reduce the speed of the collided done with the shorter path by 10%.
Reset all drones to initial points, set t = 0
Are all drones at their destination?
No
Yes
Finish simulation, print time

Figure 6: Diagram of drone transition simulation. The program takes in a set of initial and final coordinates, max drone speed, a time iteration step size, and x, y, and z collision distances as inputs, and outputs the time needed for the formation to assemble. The program starts by pairing initial and final points using the Hungarian algorithm, then creating drones at each of the starting locations. 3 m/s maximum speed, 0.1 second step size, ±0.5 meter x and y collisions distances, and ±0.2 meter z collision distances were used in our simulations. In each iteration of the simulation, the program checks for collisions. If any collisions are detected, the paths of the two colliding drones are offset by reducing the speed of the drone with the shorter path length. Then, the simulation is reset with the updated speed(s). This back-propagating speed determination method is repeated until all drones complete their paths without colliding.

We used the Python program to simulate the formation of all three designs described later in the paper. The program calculated that it would take 38.0 seconds to form the Ferris wheel and 19.6 seconds to form the balloon. Unfortunately, due to the high computational cost and runtime, we were unable to predict the formation time of the Dragon display within permitted time limit, so we estimated it to take about 30 seconds based on other results. We suspect that the Dragon display's complex geometry produced this high runtime.

4.3 Transition from Display to Grid

Planning a path for the drones to move back from the display to the grid after they completed the animation is relatively simple. We reversed the paths that the drones took in Part 1 of their path (Transition from Grid to Display) to compute their paths for their transition back to the grid. The paths still do not intersect and are the most efficient way to return to the grid. So, only Part 1 (Transition from Grid to Display) is discussed in Sections 5-7 to avoid redundancy.

5 Ferris Wheel Model

5.1 Definition

From now on, we introduce some notation to properly describe the paths of the drones. Define the position vector as $\vec{s}(t)$ and the velocity vector as $\vec{v}(t)$. Any vector with a hat written in the form $\hat{u}$ represents a unit vector of a vector $\vec{u}$.

5.2 Number of Drones

Our model of the Ferris wheel consists of one wheel of a inner and outer circle, with 12 spokes connecting the inner and outer circle. On either side of the wheel are 2 legs of an isosceles triangle serving as a base to "support" the wheel. 36 drones were used to create the outer circle and another 36 were used to create the spokes. 1 drone was used for the center, and 13 were used for each of the two triangular bases, resulting in a total of 99 drones.

5.3 Transition from Grid to Ferris Wheel

The Ferris wheel required 99 drones, so the drones were arranged on the ground in a 9 by 11 grid of drones whose center would be directly beneath the center of the Ferris wheel. Each drone was assigned a target position on the Ferris wheel to transition from the grid to the Ferris wheel display. In order to compute the ideal target Ferris wheel position and the path of each drone, the Hungarian algorithm described in Section 3.4 was used.

The initial grid locations are illustrated in left the section of the below figure. The drones then moved along their linear paths to arrive at their Ferris wheel display destinations, as illustrated in right section of the below figure.

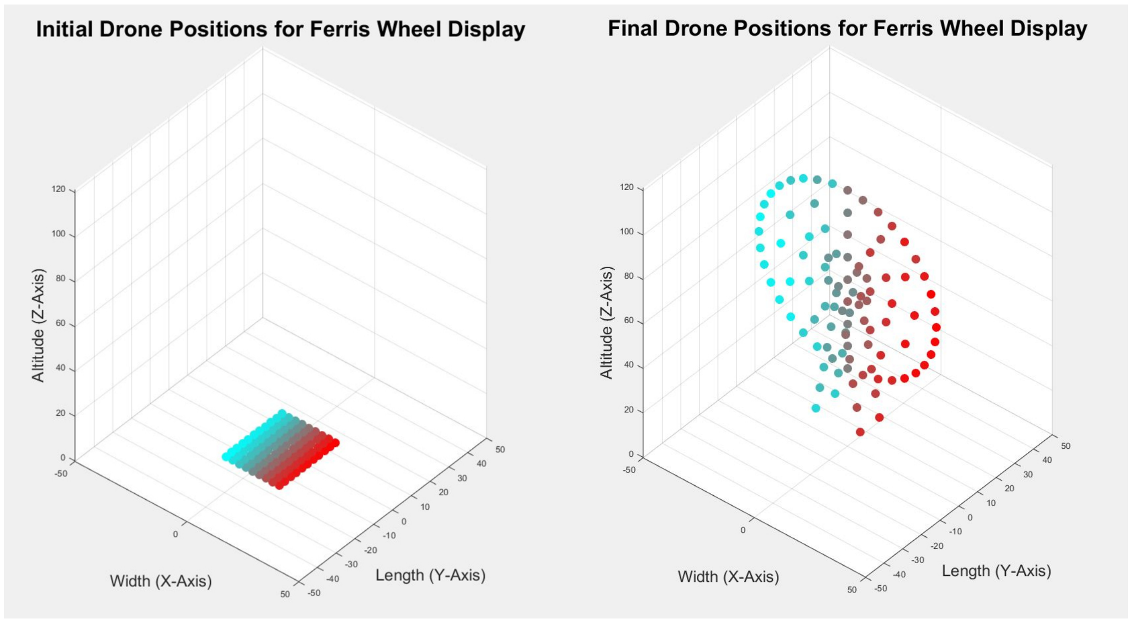


Figure 7: Left: The initial points of the drones before transitioning to the Ferris wheel. Right: Drones after movement to Ferris wheel display. The drones are color-coded based on their initial location, showing that the Hungarian algorithm sorts the drones laterally to avoid collisions. Different perspectives of the Ferris wheel are in Appendix C.

A digital animation which illustrates the movement of the points from their initial grid to display positions can be found at: https://www.youtube.com/watch?v=NDoO-spFgyk

We computed vectors which represent the path of each drone from their optimized initial positions to their final position in the display, illustrated below in Figure 8.

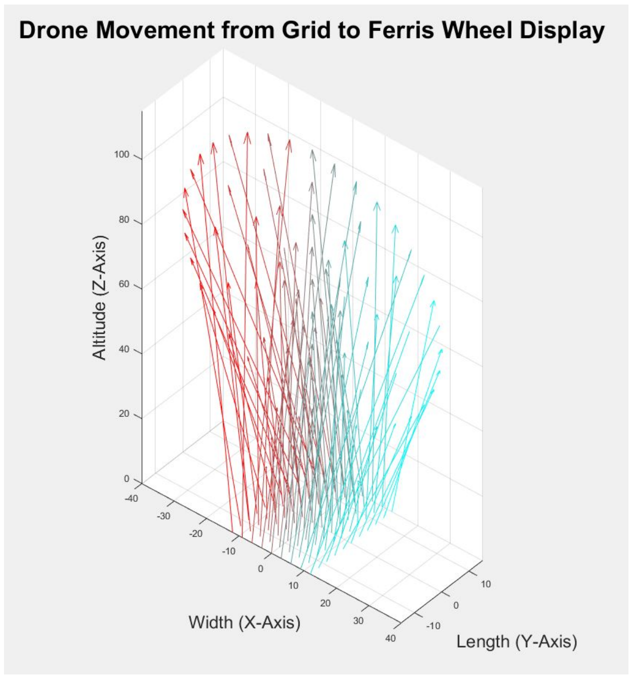


Figure 8: Mapping of all of the points from the grid to the Ferris wheel by using the Hungarian Algorithm. Due to the Extremal Principal, none of the mapped lines cross over each other.

Additionally, we graphed the distanced traveled by each drone along their respective paths, displayed in Figure 9 below.

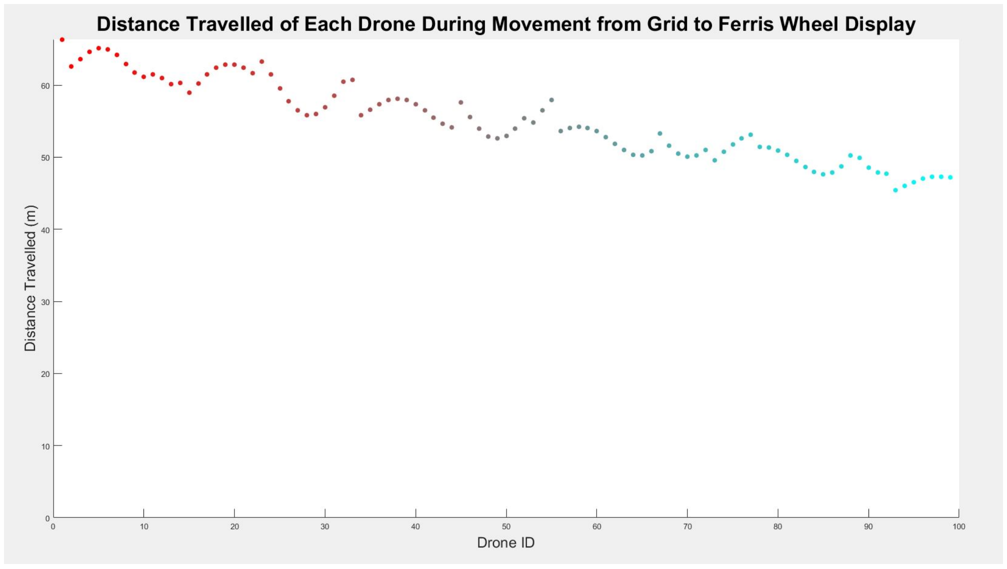


Figure 9: The distance traveled by the drones while transitioning to the Ferris wheel display. The Drone ID is simply a unique number assigned to each of the drones. The Hungarian algorithm ensures that all distances travelled are relatively low and eliminates any outliers in mapping.

5.4 Animation of Ferris Wheel

For our Ferris wheel display, we wanted the viewer to be able to not only see the rotation of the Ferris wheel's wheel, but also to view the sides and back of the Ferris wheel. To accomplish this, we made our Ferris wheel rotate on the vertical axis from the top of the wheel to the base in addition to having the wheel rotate. This would allow viewers to see the all parts of the Ferris wheel, not just one side.

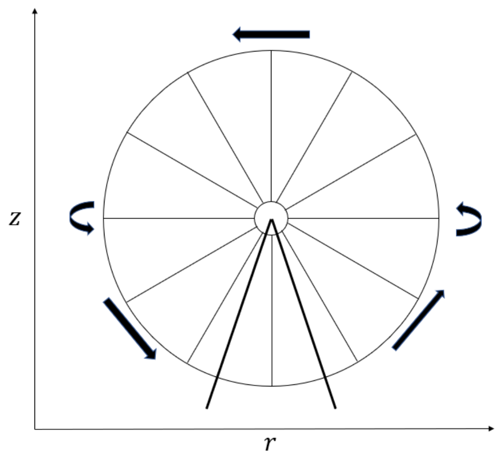


Figure 10: Diagram of the motion of the Ferris wheel. The isolated wheel rotates about the axis orthogonal to z and r. The whole wheel, including the base, rotates about the z axis. The r axis moves dynamically with the rotation of the Ferris wheel.

If we consider the rotation of the Ferris wheel (excluding the base)about the dynamic axis orthagonal to z and r, we can model the path of the drones on that wheel as a parametric equation of a circle. This can be expressed in two axes z and r.

$$\vec{s}(t) =< r, z >=< r_0 \cos(bt), r_0 \sin(bt) >$$

b is $2\pi/P$, where P is the period of the wheel, and r_0 is the radius of the Ferris wheel which is 40 m on the outer wheel and decreases by 10 m for each subsequent ring.

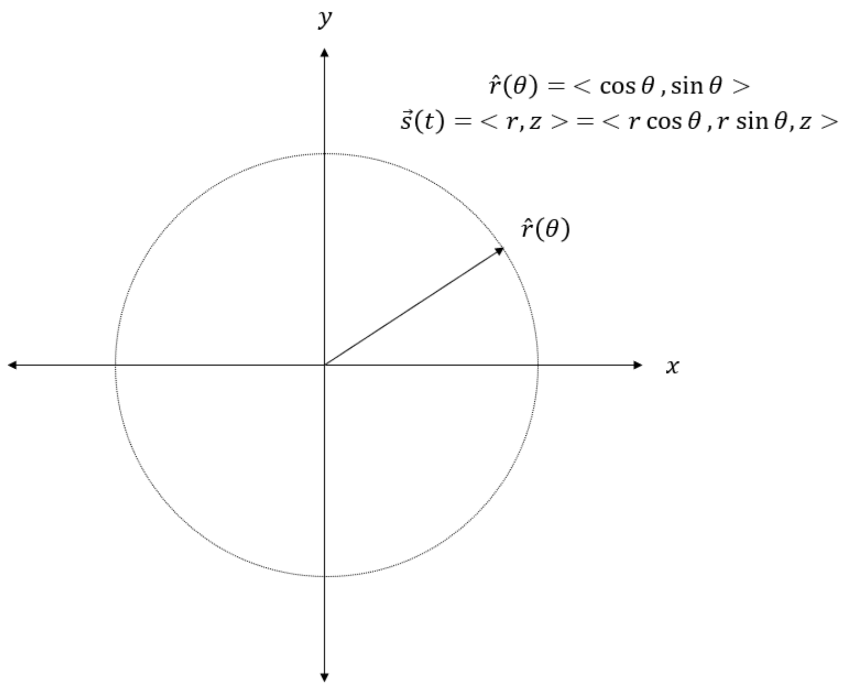

y
$\hat{r}(\theta) = < \cos\theta\,, \sin\theta >$
$\vec{s}(t) = < r, z > = < r\cos\theta\,, r\sin\theta, z >$
$\hat{r}(\theta)$
x

Figure 11: A graph of the relation between $\vec{r}$ and x, y.

However r is not a constant axis: it rotates with the Ferris wheel. If we say the Ferris wheel rotates on a period of $2\pi/a$, we can express x and y in terms of a, b, t.

Vector function for the wheel:

$$\vec{s}(t) = r_0 < \cos(at)\cos(bt), \cos(at)\sin(bt), \cos(bt) > + < 0, 0, 80 >$$

We now wish to find the velocity and make sure the max velocity is under 3 m/s.

$$\vec{v}(t) = r_0 < -a{*}\sin(at)\cos(bt) - b{*}\cos(at)\sin(bt), a{*}\cos(at)\cos(bt) - b{*}\sin(at)\sin(bt), b{*}\cos(bt) >$$

$$(\|\vec{v}(t)\|)^2 = r_0^2 * (a^2\cos^2(bt) + b^2)$$

This shows that the highest speed a drone would have to fly in this configuration is equal to $r_0\sqrt{a^2+b^2}$. To confine the drone speed below 3 m/s, we defined $a = \frac{1}{20}$ and $b = \frac{1}{20}$. Thus, the wheel of the Ferris Wheel has a period of 40π seconds which is the same as the entire Ferris Wheel's rotational period.

The rotation of the bases of the Ferris wheel can be modeled based on the rotation of the entire Ferris Wheel using $\vec{s}(t) =< r\cos(at), r\sin(at), z >$, where r is the drone's distance from the z-axis.

6 Dragon Model

6.1 Number of Drones

To model the Dragon, we juxtaposed rings and squares of similar size along a line parallel to the x axis as shown below.

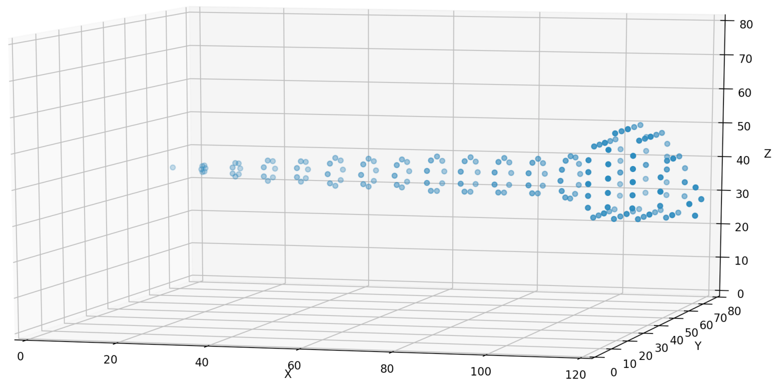


Figure 12: A 3D point cloud of the rings and squares placed to create the image of the Dragon.

Although the model does not perfectly aesthetically represent a Dragon, the cross sections of the Dragon (the rings and squares) can be modified to account for this by adding ears, eyes, teeth, and legs in future iterations. The particular cross sections of this model and the number of drones found in each cross section is described in Appendix A. The total number of drones used for our Dragon was 196.

6.2 Transition from Grid to Dragon

As with the transition to the Ferris wheel, the paths of the drones used in the transition to the Dragon display were found by using the Hungarian algorithm.

The initial grid locations are illustrated in left the section of the below figure. The drones then moved along their linear paths to arrive at their Ferris wheel display destinations, as illustrated in right section of the below figure.

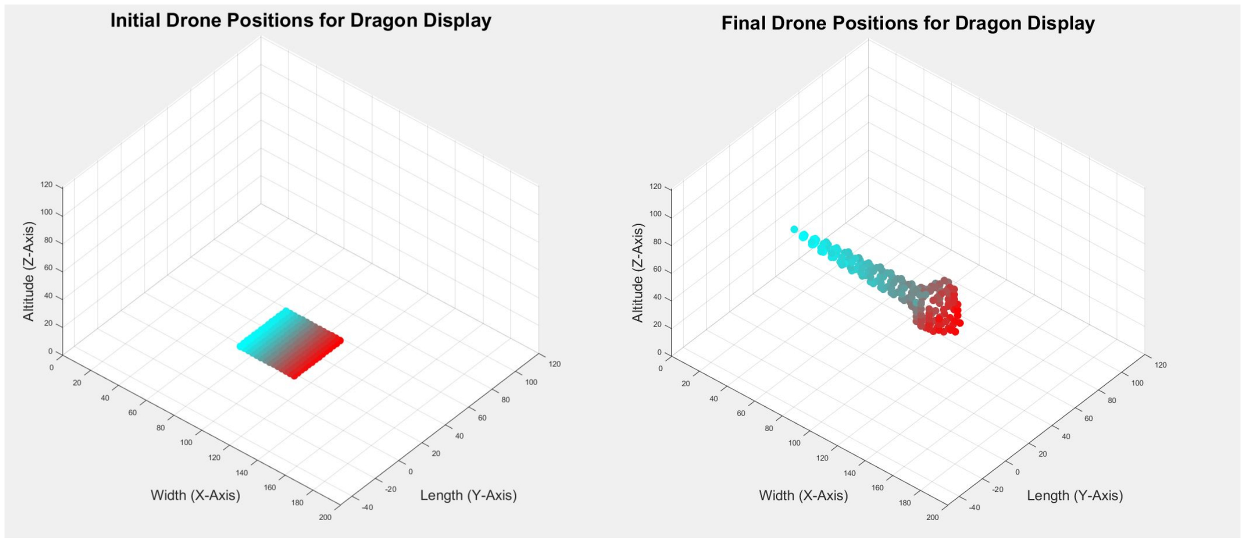


Figure 13: Left: The initial points of the drones before transitioning to the Dragon display. Right: Drones after movement to Dragon display. Different perspectives of the Dragon are found in Appendix D.

A digital animation which illustrates the movement of the points from their initial grid to display positions can be found here: https://www.youtube.com/watch?v=4D8Gh0yg_00

We computed vectors which represent the path of each drone from their optimized initial positions to their final position in the display, and they are illustrated below.

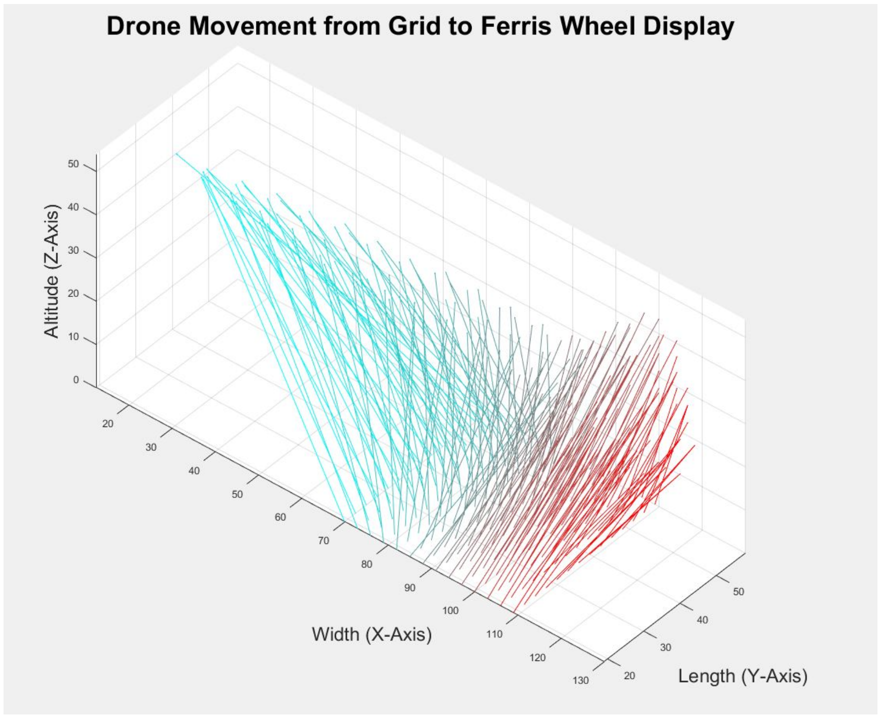


Figure 14: The paths of each drone moving from its respective initial location to final location.

Additionally, we computed the distance traveled by each drone along its respective path, which is displayed below.

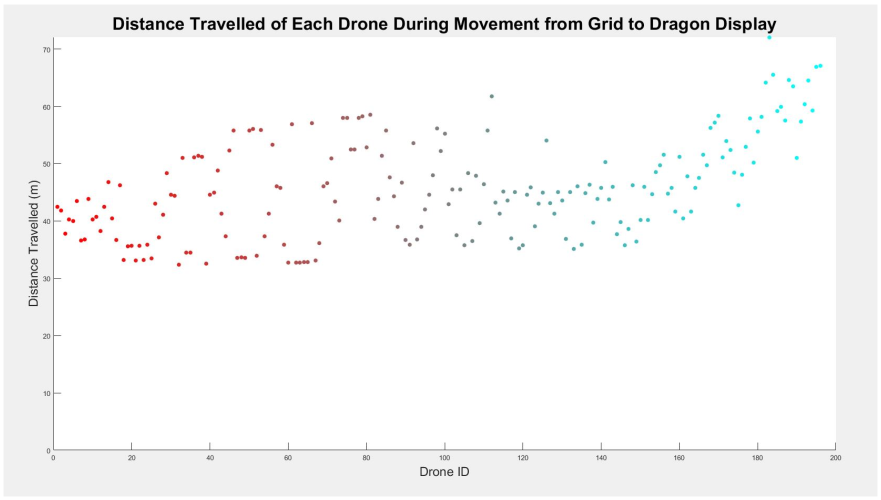


Figure 15: The distance traveled by of each of the drones while transitioning to the Dragon display.

6.3 Animation of Dragon

In order to accurately represent the movement of a mythical Dragon, we wanted the Dragon to move around the field in a twisting, helical path. So, we made our Dragon follow the vector path of a helical toroid around the city, to mimic Dragons' movement patterns.

A general diagram of a toroid is shown below.

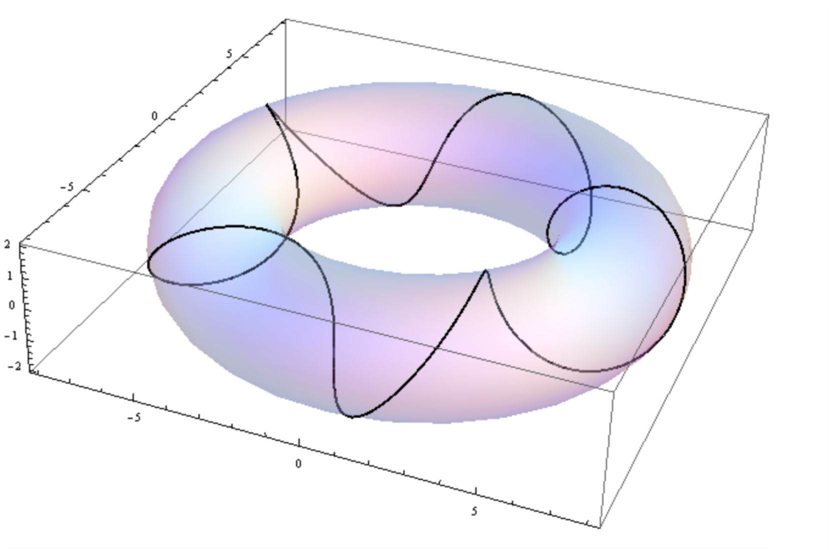


Figure 16: Representation of a general toroid.[14]

The general equation of a toroid is

$$x(t) = R + r * \cos(w * t)) * \cos(t)$$
$$y(t) = (R + r * \cos(w * t)) * \sin(t)$$
$$z(t) = r * \sin(w * t)$$

where r represents the minor radius of the torus, R represents the major radius of the torus, and w represents the number of winds that the function makes throughout one loop of the torus.

We chose to use the below parameters to construct a reasonably-sized path for the Dragon

$$R = 100m, \quad r = 25m, \quad w = 4$$

Our specific toroid with the Dragon overlaid is displayed below.

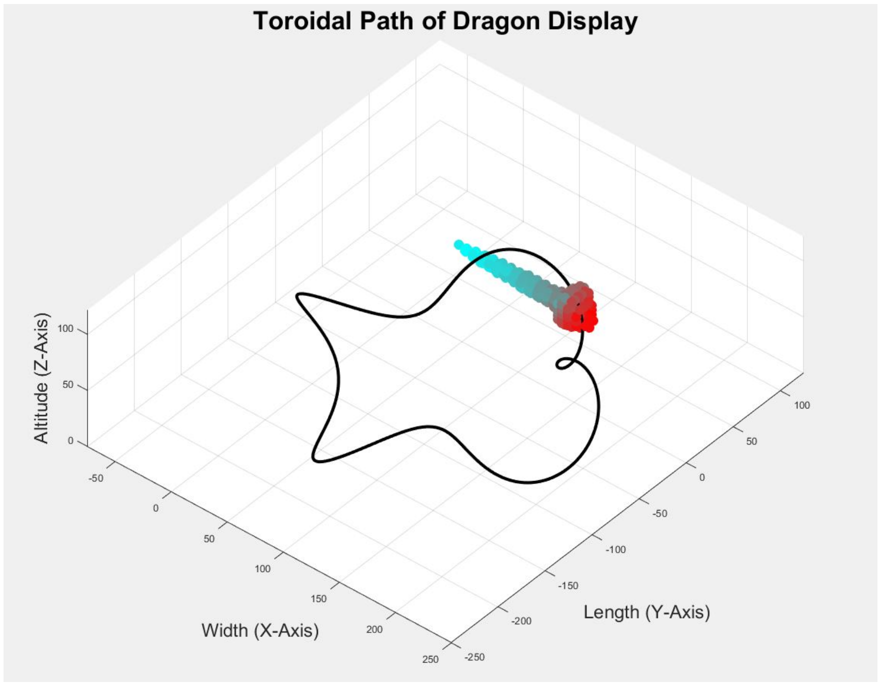


Figure 17: The Dragon on the helical toroidal path. Different perspectives are found in Appendix D.

We wanted the center of the head and the center of each circular section to follow the path of the toroid. This was done by orienting the plane of each circular section and the medial plane of the head of the Dragon to be perpendicular to the path on which the circular section or head is traveling. To find the path that each ring and the head would follow, we generated the following vector curve from the shape of the torus using our previously defined parameters.

$$\vec{s}(t) =< (100 + 25\cos(4at))\cos(at), (100 + 25\cos(4at))\sin(at), 25\sin(4at) + 75 >$$

We included the +75 in the z-component, to be the average height of the helical torus to ensure the Dragon was in the air at 75 m, and $a = 2\pi/P$, where P is the period of the Dragon to revolve around the helical toroidal path.

To find the direction the Dragon is moving along, we calculated the velocity vector.

$$\vec{v}(t) =< V_x, V_y, V_z >=$$
$$< -a\sin(at)(100 + 25\cos(4at)) - 100a\sin(4at)\cos(at),$$
$$a\cos(at)(100 + 25\cos(4at)) - 100a\sin(4at)\sin(at),$$
$$100a\cos(4at) >$$

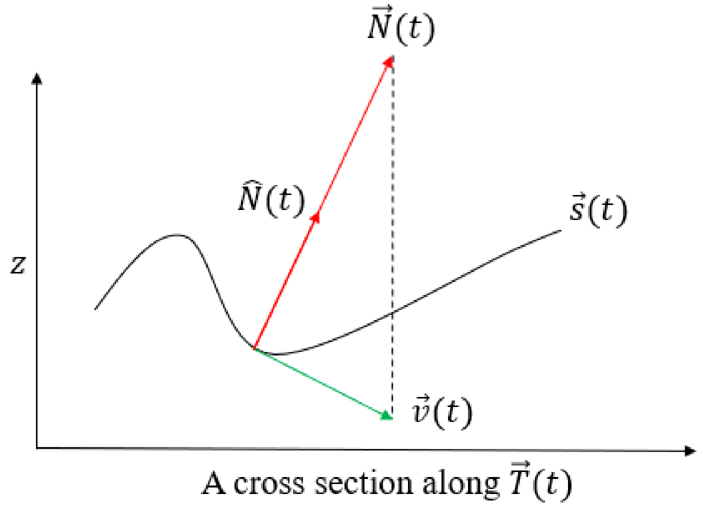


Figure 18: This shows the relation between the vectors.

As we want the plane of each ring to move along the helical torus in the direction that its center is moving, we oriented the plane to be perpendicular to the velocity vector. We do this by finding a vector $\vec{N}(t)$ perpendicular to the $\vec{v}(t)$ such that the normal vector can be rotated about $\vec{v}(t)$ to create the Dragon's circular section. Thus

$$\vec{N}(t) \cdot \vec{v}(t) = 0$$

The set of points that the terminal ends of each of the rotations of this vector $\vec{N}$ would be each of the drones, in the oriented plane.

For a given ring, we wish to set the top drone such that is it's projection on the xy-plane is the same direction as the velocity vector's projection. So we wish for

$$\vec{N}(t) =< v_x, v_y, N_z >$$

for some N_z such that $\vec{N}(t) \cdot \vec{v}(t) = 0$; solving for N_z gives us $N_z = -\frac{(V_x^2+V_y^2)}{V_z}$.

Next, we equally space the number of drones throughout that ring. We do this by finding the normalized normal vector: $\hat{N}(t) = \frac{\vec{N}(t)}{\|\vec{N}(t)\|}$.

To rotate $\hat{N}(t)$ around $\vec{v}(t)$ we use a rotation matrix, $R_{(\theta,\hat{u})}$ that rotates vectors by a angle θ around a given axis u.

$R_{(\theta,\hat{})} =$

$$\begin{bmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_y u_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_z u_x(1-\cos\theta) - u_y\sin\theta & u_z u_y(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{bmatrix}$$

Then for each circular section C_i, with D_i drones in that section, the drones would follow a path of $\vec{s}(t) + R_{(n*\theta,\vec{v}(t))} * \hat{N}(t) * T$, where $\theta = \frac{2\pi}{D_i}$ for $1 \leq n \leq D_x$ and T is the radius of the dragon's body.

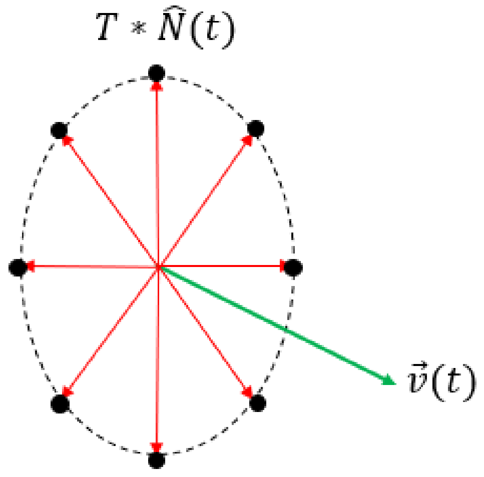


Figure 19: This shows how $T\hat{N}(t)$ is rotated around $\vec{v}(t)$ to create the drones to form a ring of a Dragon. This example showcases when $D_i = 8$.

To calculate our initial parameter a, which represents the speed at which the Dragon moves through the toroid curve, we attempt to approximate the max velocity of the Dragon. The curve moves around the torus circle of radius R every $2\pi/a$ seconds, so it travels at R/a m/s on average. Doing so again for the circle of radius r in the torus, gives us a velocity of r/a m/s. Adding these together tell us the velocity of the drone is at most $(R+r)/a = 125/a$ m/s. To ensure the velocity is lower than 3 m/s, we set $a = \frac{1}{50}$, so the period of the entire toroid curve is 50π seconds.

7 Balloon Model

7.1 Number of Drones

For our custom model, we decided to to create a balloon slowly being blown up and then eventually bursting when it became too big. To model this we assume the balloon is initially formed at some starting volume of 1 breath. Its geometry is shown in the figure below.

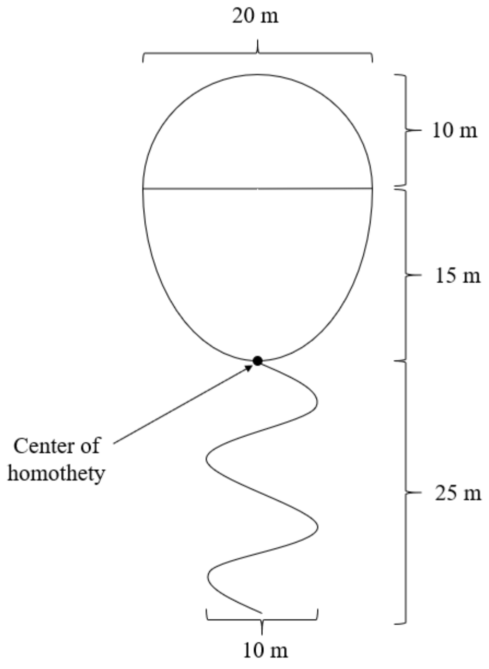


Figure 20: A graphical description of the balloon's initial starting position. Only the balloon in the upper half will be affected by homothety, the string will remain constant.

The balloon is comprised of two hemispheres of radius 5 m, where the bottom hemisphere is stretched by a factor of 1.5 in the z-axis. The string is modeled by a sine wave of two periods with amplitude 5 m. The bottommost part of the string starts at $z = 10$m.

For drone placements, we plan to use 21 drones for the string, each drone placed $\pi/5$ radians away along the sin curve.. This includes the drone at the center of homothety.

We modeled the balloon by horizontal rings placed in altitudes 2.5 m apart, starting at the bottom of the balloon, $z = 35$m and ending at the top, $z = 60$m. We then created a grid of points such that distance between drones is roughly constant in both vertical and horizontal distances. To accomplish this we try to make drones on the same ring approximately 3 m apart from each other. We calculated the radius of either hemisphere at given heights.

$r(z) =$

$$\begin{cases} 10\sqrt{1-(1-\frac{z-35}{15})^2} & 35 \leq z \leq 50 \\ 10\sqrt{1-(\frac{z-50}{10})^2} & 50 \leq z \leq 60 \end{cases}$$

We then used this to calculate the circumference, $2\pi r(z)$ and put $d(z) = \lfloor \frac{2\pi r(z)}{3} \rfloor$ drones on each ring. We then spaced all the drones an equal distance from each other along the circle of the balloon. This results in a total of 176 drones for the balloon. The locations of the drones on a given ring at height z can be described below.

$\vec{s}(0) =< r(z) * \cos(\frac{2\pi}{i*d(z)}, r(z) * \sin(\frac{2\pi}{i*d(z)}), z >$, for $1 \leq i \leq d(z)$

A chart of the drone placement on the rings is attached in Appendix B.

7.2 Transition from Grid to Balloon

As with the transition to the Ferris wheel and the Dragon, the paths of the drones used in the transition to the balloon display were found by using the Hungarian algorithm. The initial grid locations are illustrated in left the section of the below figure. The drones then moved along their linear paths to arrive at their Balloon display destinations, as illustrated in right section of the below figure.

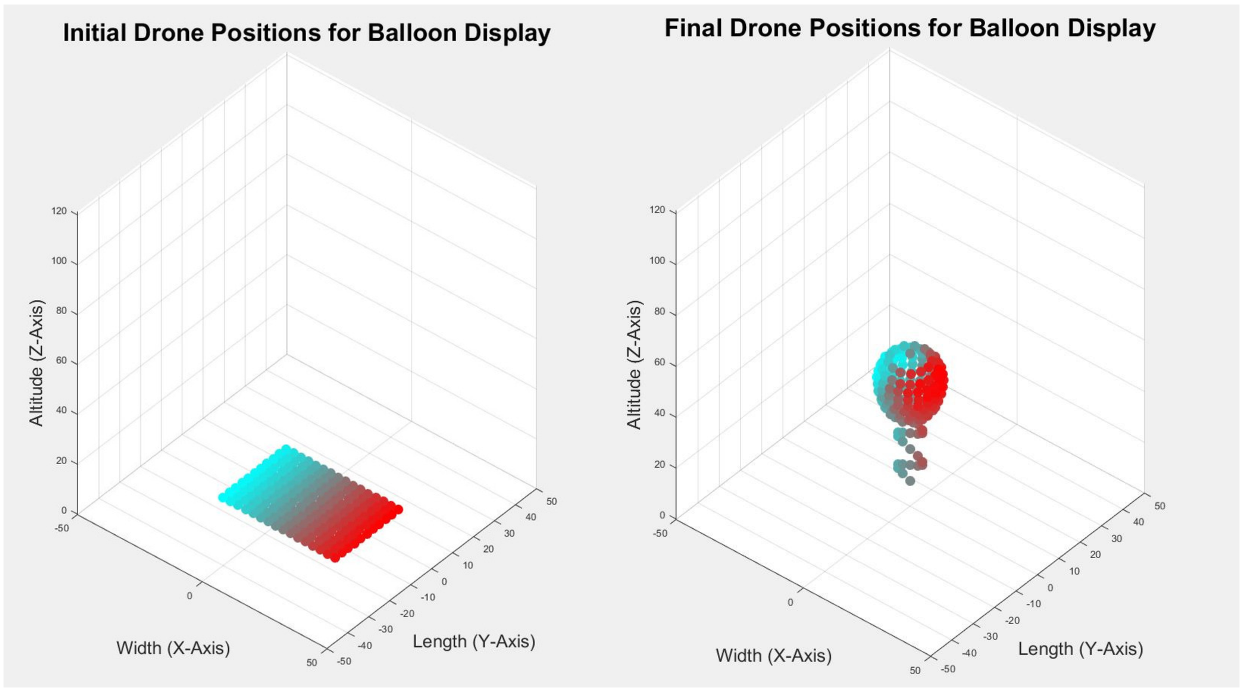


Figure 21: Left: The initial points of the drones before transitioning to the Balloon. Right: Drones after movement to Balloon display.

A digital animation which illustrates the movement of the points from their initial grid to display positions can be found here: https://www.youtube.com/watch?v=N6eDCU_Aco8

We computed vectors which represent the path of each drone from their optimized initial positions to their final position in the display, and they are illustrated below.

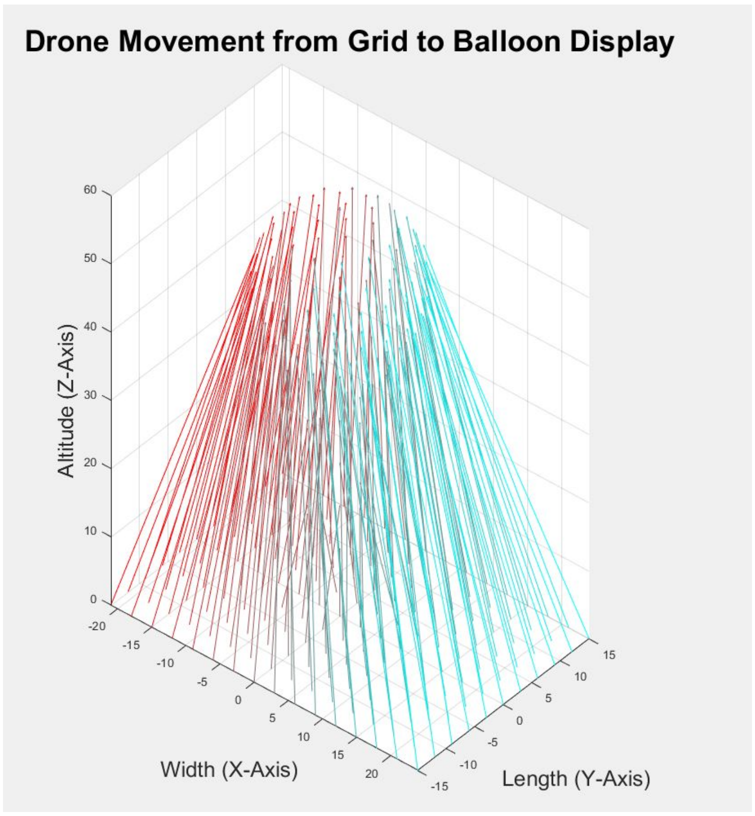


Figure 22: The paths of each drone moving from its respective initial location to final location.

Additionally, we computed the distance traveled by each drone along its respective path, which is displayed below.

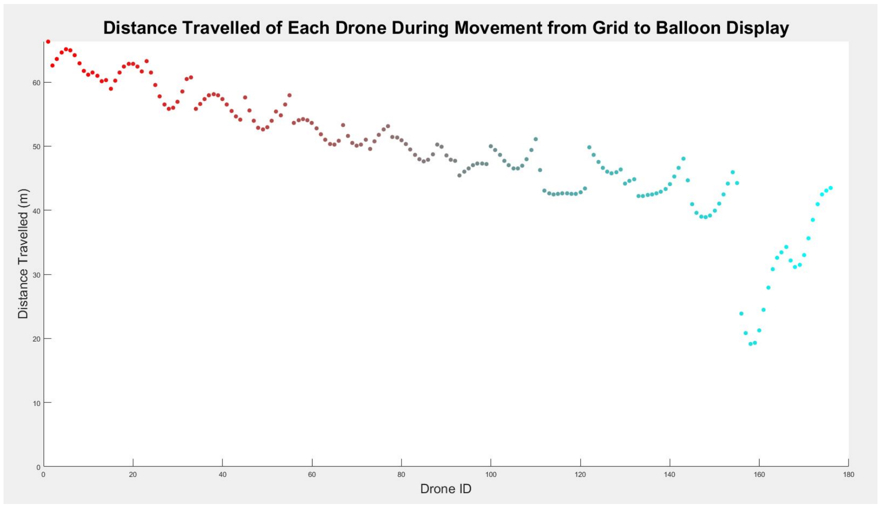


Figure 23: The distance traveled by of each of the drones while transitioning to the balloon display.

7.3 Animation of Balloon

We use the bottom point of the balloon as a center of homothety to base our scaling of the balloon as it is. To determine the scaling factor $k(t)$ we assume a cubic relation between length and volume. So

$$V(t) \propto k(t)^3$$

We wish to display the process of the balloon being blown up and then eventually popping. We assume a person blows an equal amount of air each breath and takes 1 second to inhale and 3 seconds to exhale from personal testing. We assume a constant rate of breath while exhaling we can express $V(t)$ as such. $V(t) =$

$$\begin{cases} \frac{x}{3} - \frac{n-4}{3} & 4n-4 \le x \le 4n-1, n \in \mathbb{N} \\ n+1 & 4n-1 \le x \le 4n, n \in \mathbb{N} \end{cases}$$

So $k(t) = (\frac{V(t)}{V(0)})^{\frac{1}{3}}$. Since $V(0) = 1$, we can say $k(t) = (V(t))^{\frac{1}{3}}$

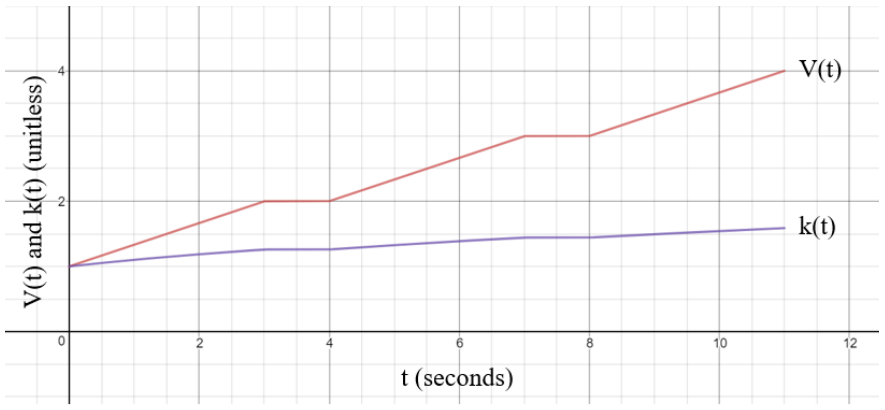


Figure 24: The graphs of $V(t)$ and $k(t)$ for $0 \le t \le 10$.

Scaling works by first finding the displacement vector between, $\vec{x}$, between the center of homothety and the original drone position. When scaled by a factor k, the new displacement vector should satisfy $\vec{x'} = k\vec{x}$.

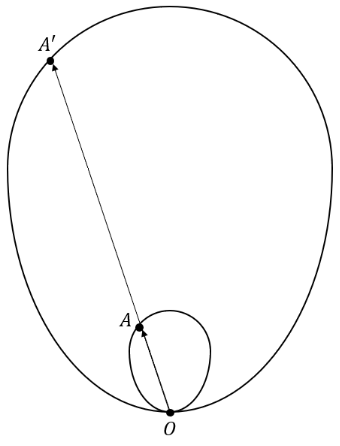


Figure 25: An example of using a homothety center to scale an image I. Notice A and its scaled version A'. Here the scaling factor $k = 4$. So $\vec{OA'} = 4\vec{OA}$

So the paths of a given drone starting at $\vec{s}$ is $\vec{s}(t) =< 0, 0, 35 > +k(t) * (\vec{s}- < 0, 0, 35 >)$. We wish to show that our drones can actually fly this path so we attempt to find the max velocity a drone must fly.

$$\vec{v}(t) = k'(t) * \vec{s}$$

$$\|\vec{v}(t)\| = k'(t) * \|\vec{s}\|$$

$$k(t) = V(t)^{\frac{1}{3}}$$

By implicit differentiation,

$$k'(t) = \frac{1}{3}V(t)^{-\frac{2}{3}}V'(t)$$

The max value of $V(t)^{-\frac{2}{3}}$ occurs at $t = 0$ where $V(t)$ is minimized. At that time, $V(t) = 1$, therefore $V(t)^{-\frac{2}{3}} = 1$. At this time $V'(t)$ also reaches its maximal value of $\frac{1}{3}$. So the max value of $k'(t)$ is $\frac{1}{9}$. The max value of $\|\vec{s}\|$ is 25, which occurs for the drone at the top of the balloon. So even in worst conditions the max velocity of a drone in this display is $\frac{25}{9}$ m/s, which is under the 3 m/s vertical speed limit.

We plan for the balloon to enlarge until $k(t) = 3$, where the max height of a drone is 110 m, still within the allowed airspace. This occurs first at $t = 103$ s. A visualization of the enlarged balloon is present in Appendix E.

8 Sensitivity Analysis

8.1 Drone Failure

A major risk in the light show is drone failure. All drones, even the DJI Phantom 3 that we are using, are susceptible to signal loss and control loss. If this occurs, there is no way for the operator's computer to control it. Luckily, the DJI autopilot should automatically try to land safely if connection is lost with the pilot, but many reports claim that this safety mechanism has failed in the past[4]. Additionally, other failure conditions, such as the battery dying or disconnecting could cause the drone to become uncontrollable. So, we wondered, what distance away would the crowd have to be to be safe? How long would it take for the drone crash into a nearby lower drone, and how much warning time do we have to change the course of nearby drones directly underneath the failed drone?

To account for these scenarios we calculated a kinematics scenario.

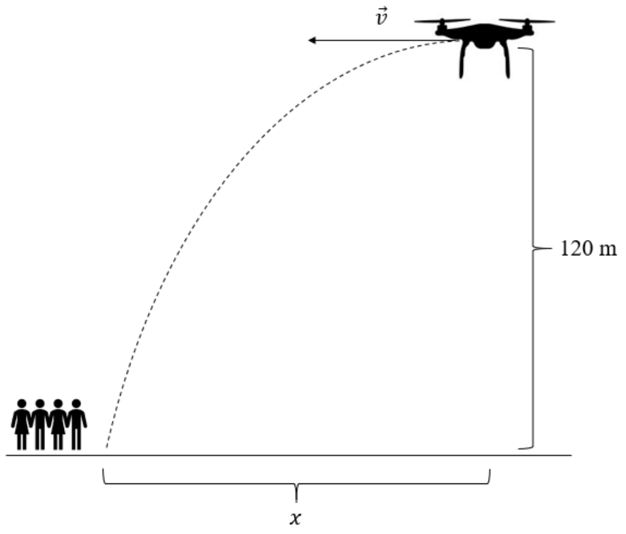


Figure 26: A worst case scenario of a disconnected battery, where the drone is headed straight towards the crowd.

We assume the drone follows projectile motion while falling.

Considering the y-axis, we see

$$\frac{1}{2}a_y t^2 = \Delta y$$

where $a_y = g$ =9.8 m/s

$$\frac{1}{2}(9.8m/s)t^2 = 120m$$

$$t = 4.95s$$

Since $\Delta x = v_x t$, we only need to know v_x. The maximum horizontal speed that our drones fly is 3 m/s, so as long as the crowd is $3 * 4.95 = 14.85$m away, drones should not crash into the crowd.

8.2 Wind Conditions

Our model does not consider any air resistance during the light show. However, strong winds could be fatal to the light display. Although we cannot predict the effects of strong winds on the drone formations, we can calculate an upper bound permissible winds using drag mechanics. The 1.280 kg drone can accelerate at 3.725 m/s^2 in still air conditions [4].

Newton's second law, $F = ma$, can be used to calculate the maximum thrust of the drone's propellers, 4.768 N. The force of the wind on a drone is determined using the drag force equation:

$$F_D = \frac{1}{2}\rho v^2 C_D A$$

where F_D is the drag force, ρ is the density of air, v is the relative velocity of the air to the drone, C_D is drag coefficient, and A is the cross-sectional area of the drone. To retain flight control, the drag force on the drone cannot exceed the thrust force of the propeller. Rearranging this equation for maximum relative wind velocity yields:

$$v_{max} = \sqrt{\frac{2F_T}{\rho C_D A}}$$

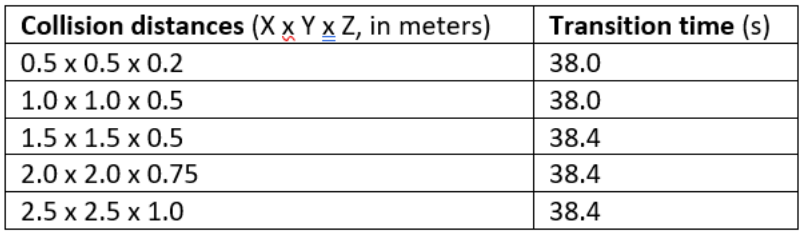
where F_T is the thrust of the propellers. The cross-sectional shape and drag coefficient, which depends on the shape, of the drone is unknown, so we will assume the drone is a hemisphere for this calculation. The drone is 0.5 m x 0.5 m x 0.2 m in size [15]. This gives its semi-circle cross section approximation an average radius of 0.23 m and an area of $A = \frac{\pi}{2}(0.23m)^2 = 0.0855m^2$. A hemisphere has a drag coefficient of 0.42 [16] and air has a density of 1.225 kg/m^3 at regular temperature and pressure [17]. Substituting all of these variables into the velocity equation yields a maximum relative velocity of

$$v_{max} = \sqrt{\frac{2 * 4.768N}{1.225kg/m^3 * 0.42 * 0.0855m^2}} = 14.72m/s$$

Drones will experience the highest relative wind speed when flying upwind. The drones fly at most 3 m/s during all displays, so if the wind velocity is greater than 11.72 m/s, our drones are at risk of losing control.

8.3 Collision Avoidance Algorithm

To test the sensitivity of the collision avoidance algorithms, we re-ran the Python program with varying collision distances. Due to the large computation time, we only had time to analyze the sensitivity of the Ferris wheel transition. The results of our analysis are shown in the table below:



Collision distances (X x Y x Z, in meters)	**Transition time** (s)
0.5 x 0.5 x 0.2	38.0
1.0 x 1.0 x 0.5	38.0
1.5 x 1.5 x 0.5	38.4
2.0 x 2.0 x 0.75	38.4
2.5 x 2.5 x 1.0	38.4

Figure 27: Calculated transition times of Ferris wheel transformation for varying collision distances, where collision distance represents the minimum distance at which the program considers the drones to be collided. The consistency in the data shows that the collision avoidance algorithm used is very efficient at pathing drones around each other regardless of collision distances.

Although we were only able to test the sensitivity of one configuration, the consistency in our results suggests that the collision avoidance algorithm is not very sensitive to changes in drone factors. Given more time, we would want to repeat this sensitivity test with the other two drone configurations.

9 Overall Display Requirements

9.1 Required Number of Drones and Launch Area

As the largest number of drones in a display is 196 for the Dragon display, this number is the required number of drones.

The launch area of the drones is calculated as follows:
For a grid in the shape $a{*}b$, the width and length would be $(a-1)*3+0.4$ m and $(b-1)*3+0.4$ m.

- For the 9x11 grid (ferris wheel), the launch area required is 24.4 m by 30.4 m.
- For the 14x14 grid (Dragon), the launch area required is 39.4 m by 39.4 m.
- For the 16x11 grid (balloon), the launch area required is 45.4 m by 30.4 m.

As the longest width required is 45.4 m and the longest length required is 39.4 m, a rectangle of these side lengths would be required for the launch area.

9.2 Air Space

As mentioned in Section 3, the altitude range for the drones is 10 m to 121.92 m, restricted by FAA regulations and human safety considerations. However, the highest drone in our light show only goes up to 115 m, in the Ferris wheel. For the length and width dimensions, the Dragon travels the furthest, traveling at a maximum of 250 m away from the origin. Thus, the airspace is a right cylinder whose bases are centered at the (x, y) origin. Its bases have a radius of 250 m while its height runs from 10 m to 115 m.

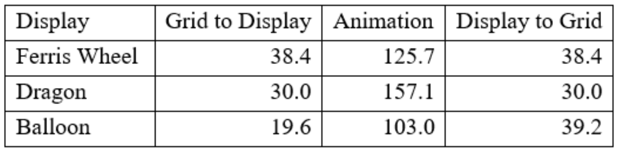
9.3 Safety Considerations

There are two safety considerations to address during the operation of the drones in the light show, the first of which is how far away the viewers should stand from the airspace. As calculated in our sensitivity analysis, the viewers should stand at least 15 m away from the airspace with regards to the position based on x and y.

The second safety consideration is one that has already been addressed: how far up the drone show should be placed so that they do not hit a person. We decided to place the lowest drone in the show at a minimum of 10 m above the ground, so that there is a minimal probability of a drone hitting a person.

9.4 Time

Our team calculated the time it would take each display to arrange itself from the starting launch grid, perform the animated display, and then return to the launch grid.



Display	Grid to Display	Animation	Display to Grid
Ferris Wheel	38.4	125.7	38.4
Dragon	30.0	157.1	30.0
Balloon	19.6	103.0	39.2

Figure 28: A time table of the 3 transitions for each of the three displays.

Launch times were calculated using our collision simulating algorithm. Though this simulation was not able to run the Dragon display, we interpolated based on the height the drones had to gain to reach their initial animation points. We predicted the Dragon display would take 30 s to form since the average height for the Dragon was 75 m, while the balloon had 65 m and the Ferris wheel has 115. This is a high estimate since they're a larger amount of collisions and congestion for the drones to reach the initial Dragon positions.

The animation time was based on the modeled curves. The Ferris wheel rotates one time on a period of 40π second. The Dragon did one full period of 50π seconds. The balloon was planned to continue growing until it hit 27 times larger than its initial size. This would cause the highest balloon length to be 110 m and take 103 seconds.

The display to grid times are identical to the grid to display times except for the fact that the balloon takes twice the time, since the balloon's final positions are twice as high as its initial conditions, causing a longer flight path.

Overall, we predict that 581.4 seconds (approximately 9.7 minutes) for the all three displays to be performed. This falls well within the DJI Phantom Drone's maximum flight time of 22 minutes, and two drown shows could most likely be executed without changing batteries [4].

10 Strengths & Weaknesses

10.1 Strengths

- Our model mathematically describes the linear path (including initial and final positions) for each drone for both transitions and animations. We are able to calculate a definitive start and end point by optimizing each drone's destination in the display using the Hungarian algorithm. This concept can be witnessed within Figure 7. The blue drones, which start towards the left of the x-axis move to the left of the Ferris wheel. Similarly, the red drones, which start towards the right of the x-axis move to the right of the Ferris wheel. The brown drones, which are closest to the center of the Ferris wheel, form the wheel's center. This notion is also seen in the Dragon and Balloon models, indicating our optimization of shortest-distance linear paths is highly functional.

- Our model outputs the velocity for each drone during all transitions and animations. Using our path functions and collisions avoidance program, a drone (such as a DJI Phantom 3) could be easily loaded with a matrix of velocity commands and can follow the path accurately.
- The pathing algorithm prevents drone collisions without sacrificing overall fluidity by optimizing drone transition formations and finely tuning drone velocities such that all drones will move to their target locations as efficiently as possible while not colliding.
- Safety analysis predicts the highest wind speed in which our drones may fly so that event organizers can plan a safe light show date based on weather forecasts.
- The collision avoidance algorithm to move the drones such that the interaction between them is minimized. This algorithm is based on the idea that the drones move to their target points such that the sum of the distances collectively traveled is minimized, otherwise known as the Extremal Principle.

10.2 Weaknesses

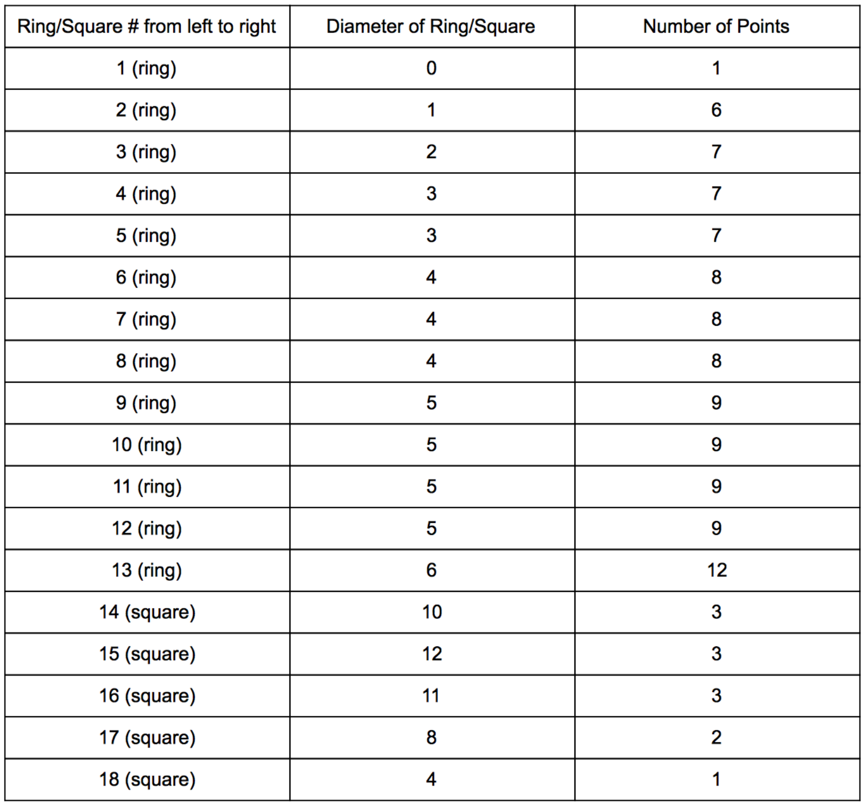
- Our model does not account for turbulence between drones. Turbulence is a significant real word phenomenon that does occur between aircraft. Though the drones are spaced significantly away, there is a chance that turbulence could interfere with the drone flights when they are more densely packed (such as during takeoff or landing).
- Our model does not account for luminous flux; essentially, it does not account for the fact that objects get less bright the further away they are seen. We are assuming that the drones' lights will always be seen, but should take into account factors like light pollution and diminishing intensity.
- The pathing algorithm is very computationally expensive: the runtime increases rapidly with increasing data set complexity, increasing collision radius, decreasing time step size, and decreasing change in velocity with each trial. However, if the time step or velocity change factor is too large, the drones will choke up on each other and the simulation will jam. If the collision radius is too small, the simulation may not be sufficient to account for real-world drone collisions.
- The algorithm cannot calculate the collision avoidance routes for the drones' transition from the 14x14 grid to the Dragon, due to the program's excessive runtime for complex designs. This could be remedied by using a more powerful computing platform or having a longer period of time for computations.

11 Future Work

- To account for luminous flux, we could compare the light pollution in the city to the luminous flux emanating from the individual light sources. If the difference in light pollution and luminous flux is small enough such that the drones are not visible, then the show is not viable at that specific distance from the viewer; we must decrease the distance.

- To model (without collisions) the paths of the drones as they form the Dragon, using a more powerful computational platform which could handle the number of intense calculations that arise from the collision avoidance protocol.
- Conduct more in-depth sensitivity analysis on the two untested drone configurations.
- Accounting for differences in velocity and position caused by wind turbulence would be challenging but worthwhile goal. Using concepts such as curl and divergence to model common wind patterns in the air would correct any error in the drones' positions and velocities.

A Diameters and Side Lengths for Dragon's Cross-Sections



Ring/Square # from left to right	Diameter of Ring/Square	Number of Points
1 (ring)	0	1
2 (ring)	1	6
3 (ring)	2	7
4 (ring)	3	7
5 (ring)	3	7
6 (ring)	4	8
7 (ring)	4	8
8 (ring)	4	8
9 (ring)	5	9
10 (ring)	5	9
11 (ring)	5	9
12 (ring)	5	9
13 (ring)	6	12
14 (square)	10	3
15 (square)	12	3
16 (square)	11	3
17 (square)	8	2
18 (square)	4	1

Figure 29: The diameters of the rings and the side lengths of the squares that are the cross sections of the Dragon.

B Dimensions and Number of Drones for Balloon Rings

Ring	Height (m)	Circumference (m)	Drones
1	0.0	0.0	1
2	2.5	34.7	11
3	5.0	46.8	15
4	7.5	54.4	18
5	10.0	59.2	19
6	12.5	62.0	20
7	15.0	62.8	20
8	17.5	60.8	20
9	20.0	54.4	18
10	22.5	41.6	13
11	25.0	0.0	1

Figure 30: The number of drones in and circumference of each ring that make up the balloon, along with their respective heights.

C Figures for Ferris Wheel Tranformations

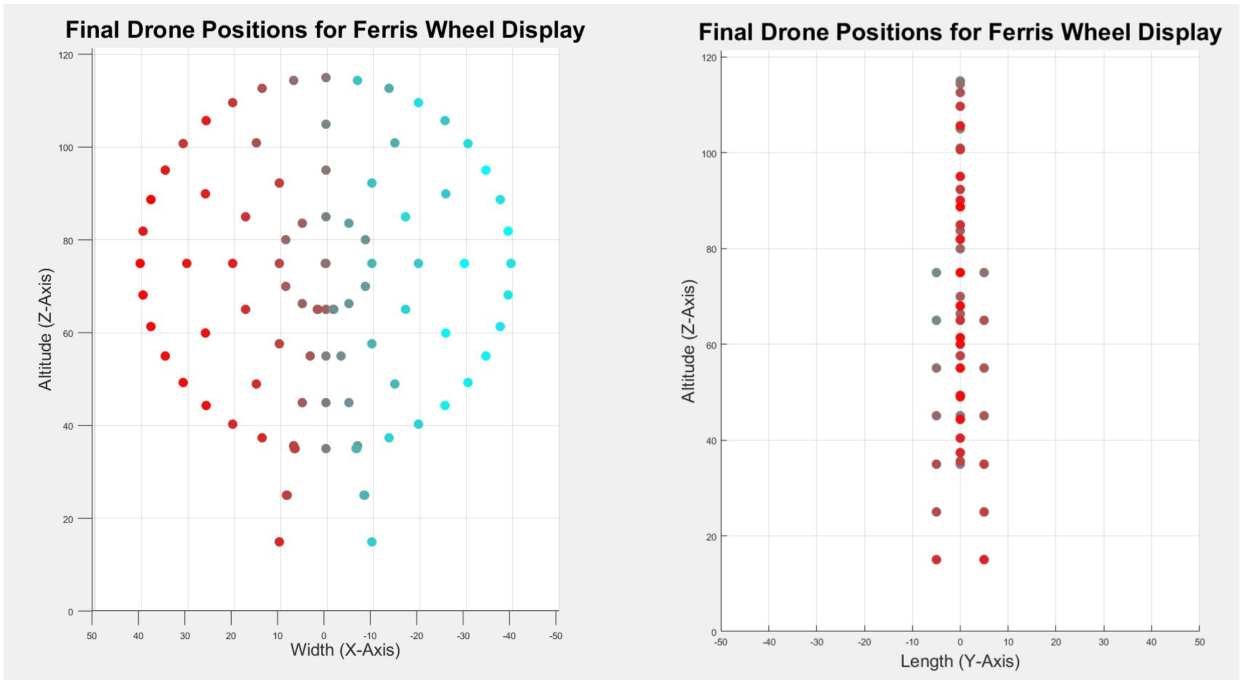


Figure 31: Different perspectives of the Ferris wheel.

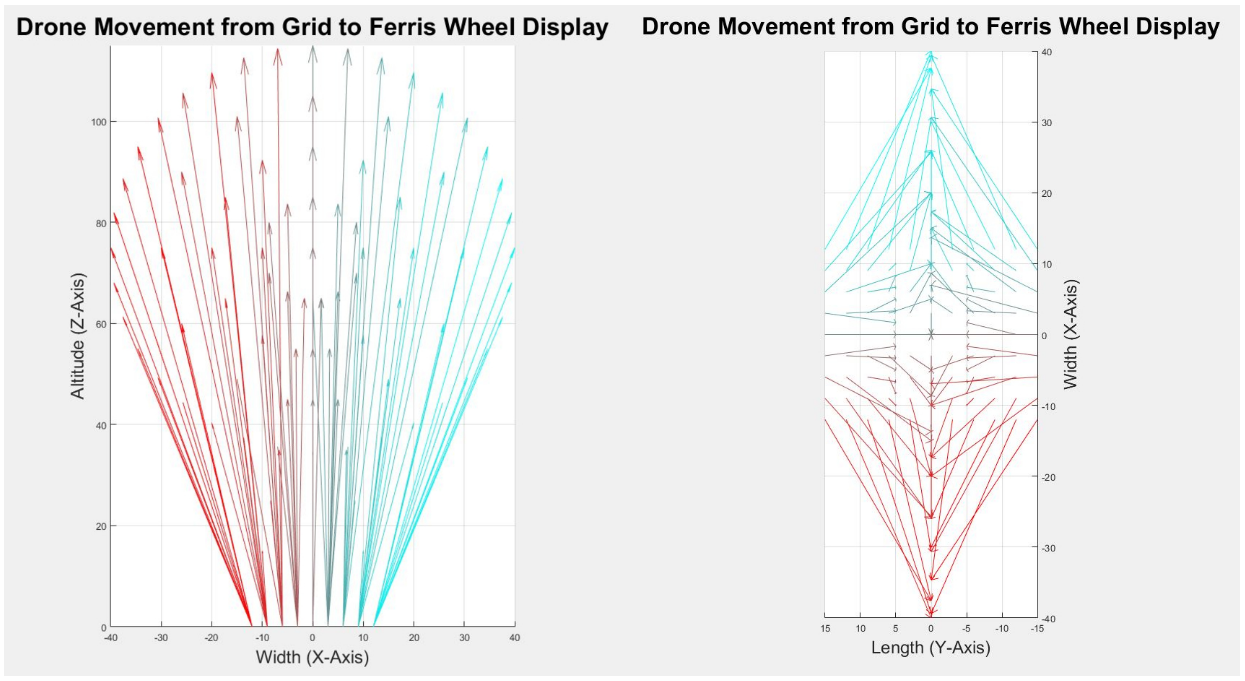


Figure 32: Two different perspectives of the transition vectors from the 9x11 grid to the Ferris wheel.

D Figures for Dragon Transformations

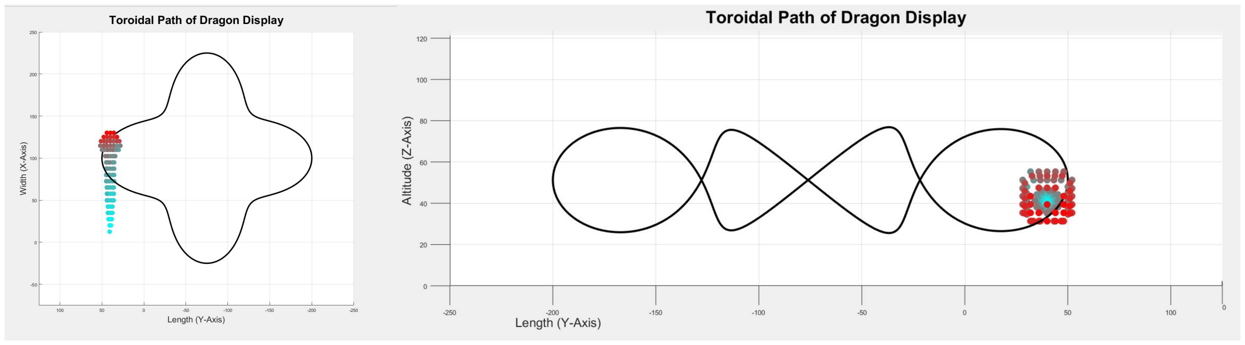


Figure 33: The path taken by a Dragon along a toroidal curve.

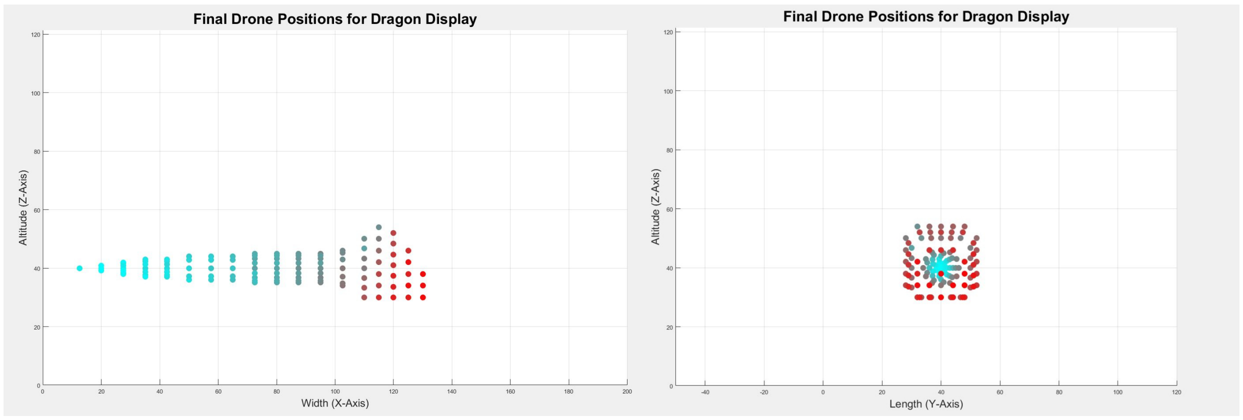


Figure 34: Different perspectives of the Dragon.

E Figures for Balloon Tranformations

The initial balloon display drone locations are illustrated in left the section of the below figure. The drones then moved along their paths to arrive at their blown-up or expanded display destinations, as illustrated in right section of the below figure.

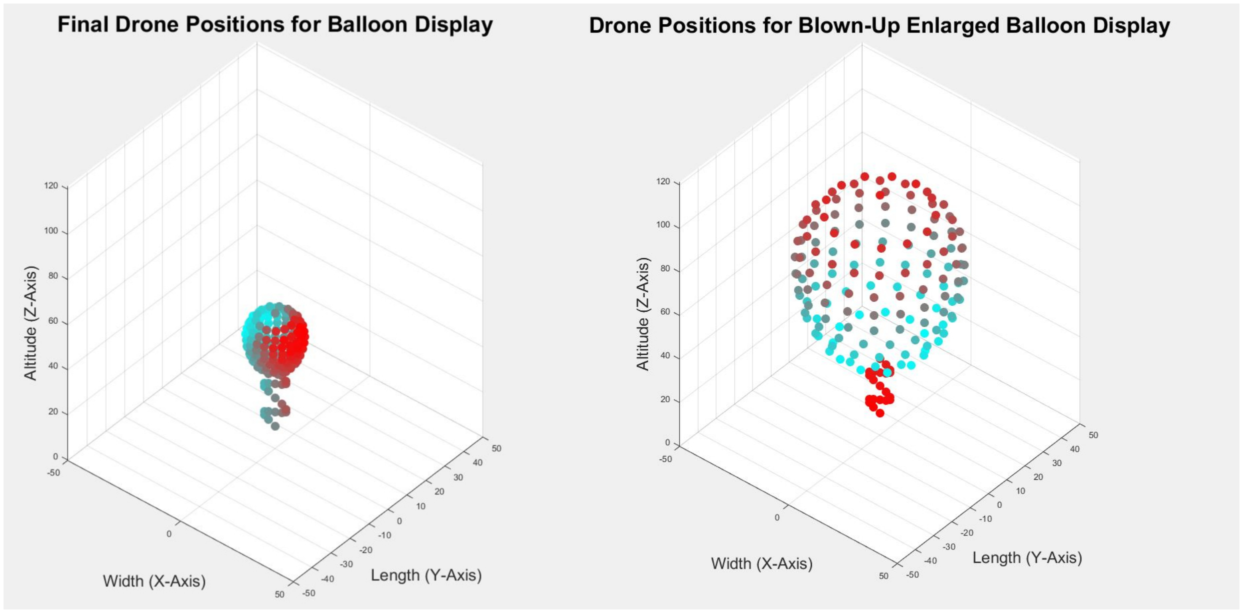


Figure 35: Left: The positions of the drones representing a non-expanded balloon. Right: Drones after "blowing up" of the balloon to enlarged balloon display.

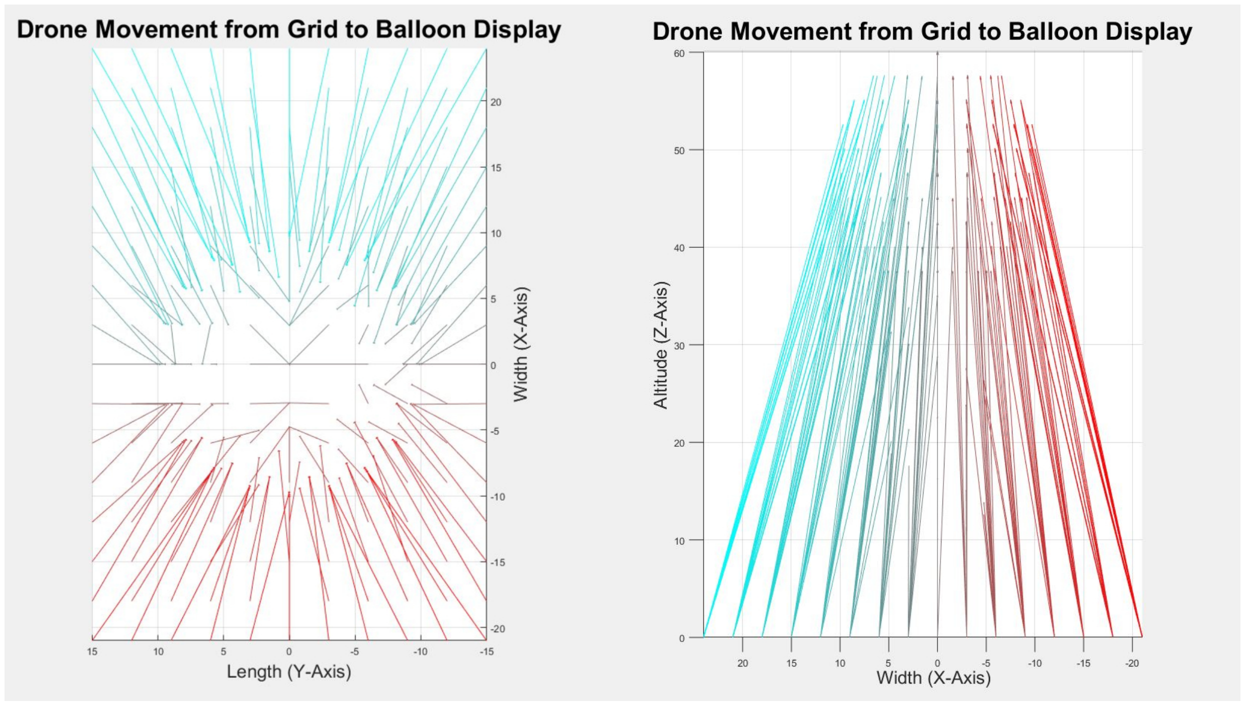


Figure 36: Two different perspectives of the transition vectors from the 16x11 grid to the balloon.

F References

[1] Ken Kaplan. *500 Drones Light Night Sky to Set Record*. URL: https://iq.intel.com/500-drones-light-show-sets-record/.

[2] Brian Barret. *Lady Gaga's Halftime Shown Drones Have a Bright Future*. URL: https://www.wired.com/2017/02/lady-gaga-halftime-show-drones/.

[3] Ian Morris. *Intel Has Drones That Will Make Fireworks Obsolete*. URL: https://www.forbes.com/sites/ianmorris/2017/09/19/intel-has-drones-that-will-make-fireworks-obsolete/.

[4] *Phantom 3 Standard - Specs*. URL: https://www.dji.com/phantom-3-standard.

[5] *DJI Phantom 3 4K WiFi Quadcopter 4K Video Drone (DJI Refurbished Unit)*. URL: http://www.dronenerds.com/products/refurbished/phantom-3-refurbished/dji-phantom-3-4k-wifi-quadcopter-dji-refurbished-unit-cp-pt-000308-r-dji.html?gclid=EAIaIQobChMIwLXshOi31wIVaTPTCh052A5uEAkYAyABEgLwp_D_BwE.

[6] Sally French. *Walt Disney World announces its first drone light show*. URL: https://www.marketwatch.com/story/can-drones-replace-fireworks-2016-11-04.

[7] Leo Hickman. *Enough of those toxic firework displays!* URL: https://www.theguardian.com/environment/2007/nov/01/ethicalliving.pollution.

[8] Natalie Cheung. *Technology Behind the Intel Drone Light Shows*. URL: https://www.roboticstomorrow.com/article/2017/05/technology-behind-the-intel-drone-light-shows/10022.

[9] drvonhoss. *DJI Phantom 3 Acceleration Test*. URL: https://www.youtube.com/watch?v=sNPLnX4wk4Y.

[10] Federal Aerospace Administration. *Getting Started: Unmmanned Aircraft Systems*. URL: https://www.faa.gov/uas/getting_started/.

[11] CBS News. *Behind the making of a synchronized drone light show*. URL: https://www.cbsnews.com/news/synchronized-drones-Intel-light-show/.

[12] Andrew Lizeswki. *NASA's Supercomputers Reveal the Incredible Turbulence Produced By a Drone*. URL: https://sploid.gizmodo.com/nasas-supercomputers-reveal-the-incredible-turbulence-p-1791179507.

[13] Billy Steele. *Intel's latest light show was the first FAA-approved drone swarm*. URL: https://www.engadget.com/2016/05/05/intel-faa-approved-drone-swarm-light-show/.

[14] *Do These Equations Create A Helix Wrapped Into A Torus*. URL: https://math.stackexchange.com/questions/324527/do-these-equations-create-a-helix-wrapped-into-a-torus.

[15] *Phantom 3 size: Complete Measurements*. URL: http://www.engabao.com/phantom-3-size-complete-measurements/.

[16] *Drag Coefficient*. URL: https://en.wikipedia.org/wiki/Drag_coefficient.

[17] *Density of Air*. URL: https://en.wikipedia.org/wiki/Density_of_air.