Team Control Number

2429311

Problem Chosen

A

2017

ICM

1 Summary

This paper describes our plead to the mayor for not holding the drone light show

Drone light shows are generally well received. They can exhibit a variety of colors, pathways, and velocity. Recently there has been an increase in having drone light shows, in contrast to fireworks. Fireworks used to be the traditional way to hold a light show but drones are just better now. Here are some reasons why: Drones can be used at any time of the day, whereas fireworks are confined to the night time. Drones don't involve chemical combustion – which is bad for the environment – whereas fireworks do.

We'd like to simulate a drone light show that showcases a ferris wheel, dragon, and a bad apple movie. Some of the things that we have to account for are the number of drones, flight pattern of each individual drone, and to ensure that no drones collide with each other.

To predict the flight pathways of all drones, we began by breaking down each visual representation into a series of discrete points on a Cartesian coordinate system, where each point corresponds to an individual drone. After this initial separation, we categorized the points into groups based on their intended movements, allowing us to simplify the programming process for similar actions. The movement of the drones will be programmed using linear transformations, which will facilitate smooth and coordinated motion. This approach requires careful consideration of each drone's speed to ensure precise synchronization and prevent any potential collisions during the performance.

Contents

2 Introduction

2.1 Background

Drones are becoming more and more popular as technology becomes more accessible around the world, offering a unique alternative to fireworks. Drones are commonly used at national events, halftime shows, Fourth of July events, and much more.

Drones provide a better alternative to fireworks. Drones are made out of carbon fiber composites and do not involve chemical combustion. On the other hand, fireworks are a one-time use that pollute the environment. Fireworks release harmful chemicals and heavy metals into the air, contributing to air and water pollution, while drones, which are made from carbon fiber composites, do not involve combustion or release pollutants. Not to mention, drones have a much lower hazard risk than fireworks. The precision and customization towards choreography is in favor of drones, highlighting the re-usability and programmed movement.

While drone light shows are generally well received, there is still a controversy about drones. Conducting a drone show requires special aviation permission, requiring complex processes especially when the use of aerial space is used frequently. While drones are a much better alternative to fireworks in terms of environmental damage, drones use up large amounts of energy and resources. Having a light show made up of hundreds or thousands of drones poses a problem for environmental stability. Additionally, the addition of technology introduces another potential failure: technical failures. Technical failures could include a malfunction in a drone, compromising the safety of the spectators.

2.2 Problem Restatement

To determine the number and flight patterns of all our drones to simulate a ferris wheel, dragon, and an image of our choosing in a sky display. Additionally, to discuss the requirements regarding the number of drones, launch area, air space, safety considerations, and duration of the drone show.

3 Letter to Mayor

Dear Mayor,

After 2 weeks of concentrated research we have conducted a safe program agenda for the drone light show you requested, with safety guarantees and a set timeline with a set amount of drones. We have decided to entertain the population of our town with a festive Ferris wheel, a powerful dynamic dragon, and hosting a finale with a glorious piece of pop culture in Bad Apple, filled with delightful plot twists.

The Ferris wheel needs 498 drones. The drone formation will have a static base with a design of moving drones that rotate in a circular motion. The drones will spin around for 30 seconds, allowing ample time for the audience to enjoy the beauty of our ferris wheel

Transitioning to the beginning of the dragon show will take 15 seconds. The dragon required is 768 drones. The drones will go in specific directions to make it look like the dragon is flying, breathing air, and shaking its head.

Transitioning to the Bad Apple show will take 15 seconds. The Bad Apple show will involve drones in a rectangular format, some white and the others black. To display the show, the contrast between the black and white drones will show the Bad Apple movie. It will take around 30 seconds.

Here are the logistics for planning:

- Rent out a venue with 400 square meters of space
- Use aerial space from altitudes 500 - 1500 ft
- total launch space required: 200 square meters
- maximum audience: 1000 people
- amount of money needed: $768,000

Here are the safety considerations:

- Safety Warning: Throughout the light show, attendees must remain in the designated audience area. Drones may accidentally collide or run out of power, so people must stay vigilant and comply to event staff instructions
- Epilespy Warning: Please be aware that the drone light show will include flashing lights, rapid transitions between light and dark, and high-contrast visuals, which may trigger seizures in individuals with photosensitive epilepsy. If you are sensitive to such visuals, we recommend you refrain from attending this event or take necessary precautions
- Standby team: A standby team of safety personnel, including drone operators, medical staff, and security, will be present throughout the event to ensure quick responses to any issues. Please locate the nearest emergency contact points upon arrival, and do not hesitate to reach out to event staff in case of any concerns
- First aid team: A first aid team will be present to attend to any unplanned injuries or accidents that occur

- Weather Conditions: In any event of strong winds, heavy precipitation, or any other harsh weather that disturbs the drones, the show will be cancled for safety reasons. Atendees will be notified through official channels

In conclusion, we advocate not to display the drone light show.

The financial burden of hosting a large-scale drone light show is a significant deterrent. The cost of acquiring or renting 1,000 drones, along with the necessary equipment such as charging stations and controllers, represents a major initial investment. Additionally, operational expenses, including hiring skilled drone operators, maintenance staff, and safety personnel, would further escalate the budget. The need for liability insurance to cover potential accidents or technical malfunctions adds yet another layer of cost. Together, these financial demands make the project far too expensive for the local budget, especially when considering the other costs associated with venue rental and emergency support.

Moreover, the staff and regulatory requirements for safely conducting a drone show are complex and labor-intensive. Ensuring compliance with aviation authorities, obtaining permits, and adhering to local drone regulations involve considerable time and effort. The need for specialized personnel, such as licensed drone pilots, flight coordinators, and safety officers, creates logistical challenges and increases labor costs. The sheer number of people required to manage the event, from security to medical staff, makes the show difficult to coordinate and manage efficiently. In addition, the process of onboarding and training these individuals on safety protocols and regulations complicates the overall planning.

Finally, the energy consumption of such a show raises environmental and logistical concerns. The drones would require a substantial amount of power, both for flight and for charging between performances, which could strain local energy infrastructure. Without access to renewable energy sources, the carbon footprint of the event would be considerable, contradicting any sustainability efforts the town may wish to promote. Furthermore, the need for backup power, such as diesel generators, would contribute to noise and air pollution, making the environmental cost of the show another major reason not to proceed.

Thank you for reading this letter!

4 Assumptions and Justifications

- Assumption: All drones we utilize will have a radius of 9 inches.

  Justification: Fixing the drone's radius to a consistent value allows for a consistent volume for all drones. A consistent volume leads to easier calculations regarding collision avoidance and drone movement.

- Assumption: The speed of all drones are not affected by any external factors.

  Justification: Disregarding motion effects of external factors, such as wind, allows for the drones' movement to be decided only by the pre-programmed movement, eliminating inconsistent environmental motion calculations, as these are factors which cannot be easily accounted for.

- Assumption: Drone battery life is irrelevant in our calculations.

  Justification: Our model does not account for the drone's battery life. The possibility that a drone dies is omitted in our calculations as the average flight can take from 20 to 30 minutes. (1)

5 Drone Models

5.1 Ferris Wheel (498 Drones)

5.1.1 Image Construction

The following image was just made by playing around with equations in Desmos 3D. Each point represents a separate drone.

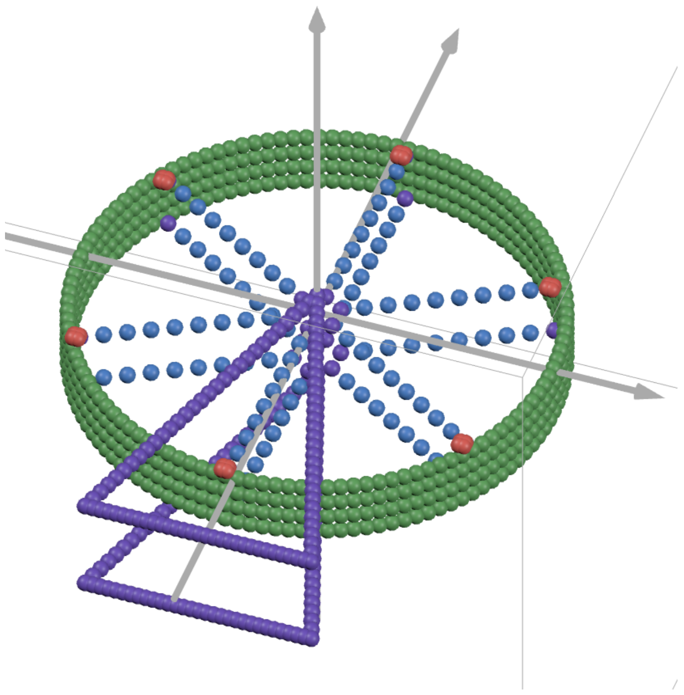


Figure 1: Ferris Wheel in Desmos 3D

5.1.2 Flight Path

The drone design of our Ferris wheel is split in two sections: static and motion

The static drones form an isosceles triangle, mimicking the base of a safe Ferris wheel. One corner of the triangle is positioned at the center of the motion Ferris wheel drone design. For a list $l = [0,\ 0.1, ..., 3]$, the base and the two legs of the triangle are a list of points represented as

$$\left(\frac{l}{3} - 0.5, -1.5, 0.1\right)$$

$$\left(-\frac{l}{6}, -\frac{l}{2}, 0.1\right)$$

$$\left(\frac{l}{6}, -\frac{l}{2}, 0.1\right)$$

$$\left(\frac{l}{3} - 0.5, -1.5, -0.35\right)$$

$$\left(-\frac{l}{6}, -\frac{l}{2}, -0.35\right)$$

$$\left(\frac{l}{6}, -\frac{l}{2}, -0.35\right)$$

The motion drones form the moving wheel of a Ferris wheel. All of the drones that are part of the wheel rotate in a circular motion, denoted by a time variable, t $\in [0, \pi]$. The five sections of the moving wheel are: outer circle, inner circle, outer circle and inner circle connectors, carriages, and carriage connectors. To model these sections over time, we utilized parametric equations with defined lists $a_2 = \left[\frac{\pi}{60}, \frac{2\pi}{60}, ..., 2\pi\right]$, $m = [0.2, 0.3,\ ...,\ 0.9]$, and $a = \left[\frac{\pi}{3}, \frac{2\pi}{3}, ..., 2\pi\right]$:

1. Outer Circle: $(1.04\cos(t + a_2),\ -1.04\sin(t + a_2))$
2. Inner Circle: $(0.1\cos(t + a),\ -0.1\sin(t + a))$
3. Outer and Inner Circle Connector: $(m\cos(t + a[6]),\ -m\sin(t + a[6]))$
4. Carriage: $(\cos(t + a) + 0.01,\ -\sin(t + a) - 0.06)$
5. Carriage Connector: $(\cos(t + a),\ -\sin(t + a))$

5.2 Dragon (768 Drones)

5.2.1 Image Construction

The starting point positions for the dragon was made by a script written to parse a .stl file (6) we downloaded from the website Thingiverse (5).

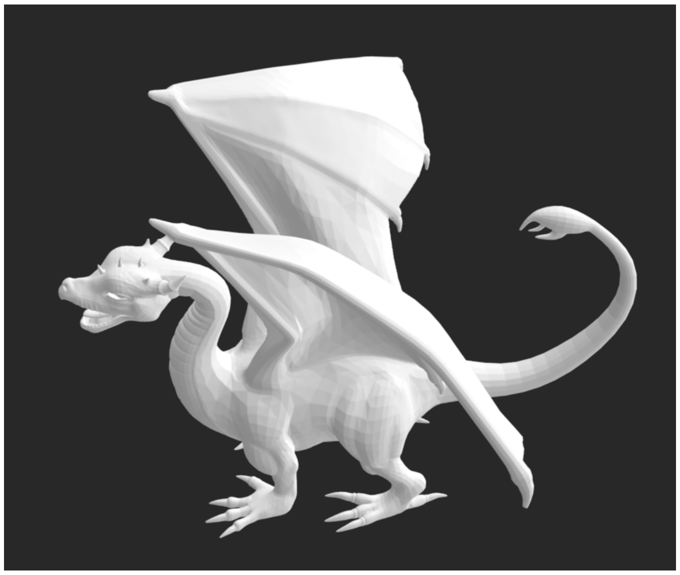


Figure 2: The .stl file when viewed in the preview app on MacOS

Parsing the .stl file resulted in 465326 points, of which we selected 768 by taking the mean of every 605 points.

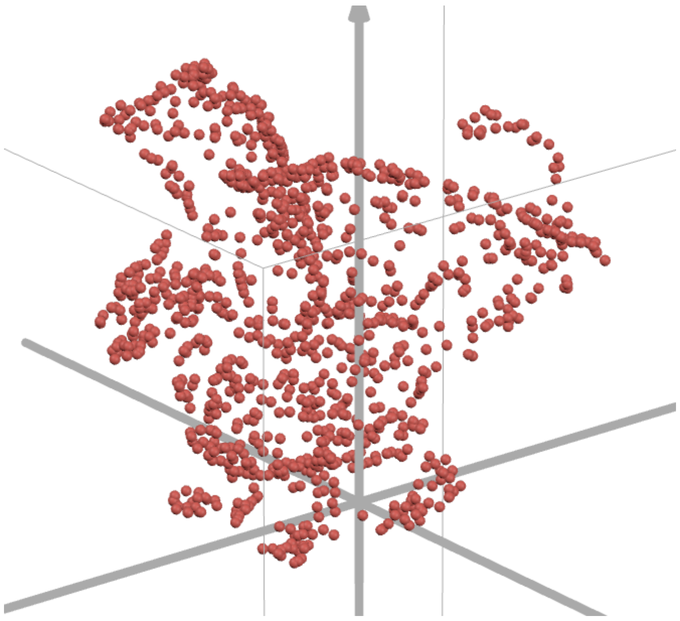


Figure 3: These points plotted in Desmos 3D

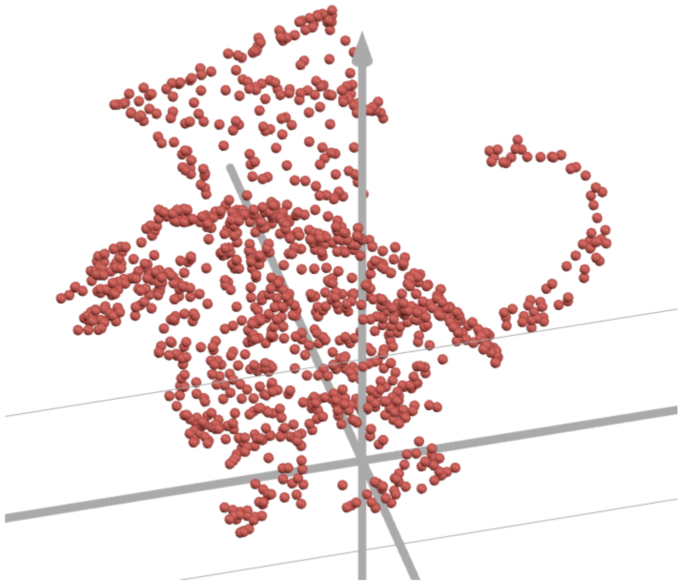


Figure 4: Another angle of the points plotted in Desmos 3D

5.2.2 Flight Path

Linear transformations allow our drones to operate through a combination of translations, rotations, and dilations.

Translations can move a list of drones to a new location by offseting them. Suppose a drone existed at some point (x, y), a vector, in the 2 dimensional case. A translation by (a, b) can be represented as

$$(x, y) + (a, b) = (x + a, y + b)$$

In the equation above, a is the number of units the drone shifts to the right, and b is the number of units the drone shifts upwards. In fact, we can define this translation more formally, for any dimension, with a function (we specifically do this for 3D, as that's the space that our drones will fly in). We define $\mathcal{T}_u(v)$ to be the translation function, where your input vector, v, is translated by an offset vector u:

$$\mathcal{T}_u(v) = u + v.$$

This is illustrated in the next page:

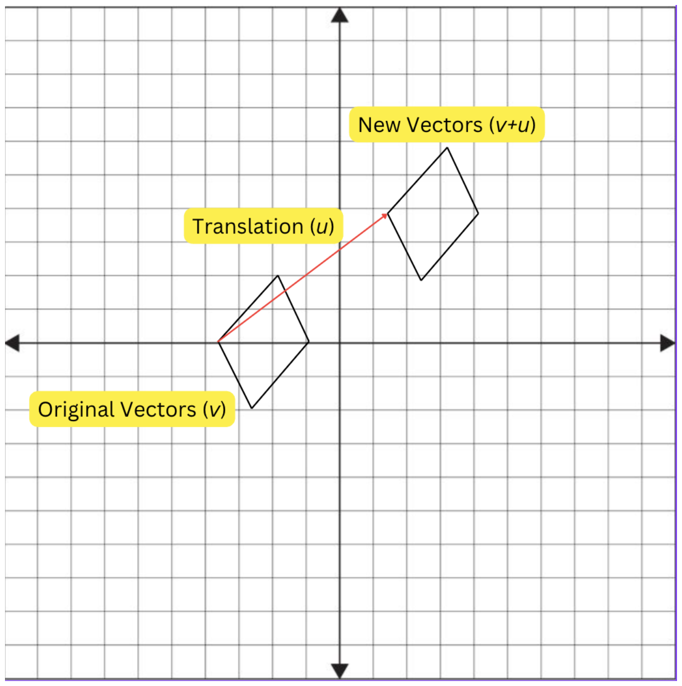


Figure 5: Translating a shape in 2D

Through this method, we can allow our dragon to move in a general direction, for example, fly.

Similarly, we can utilize Linear Algebra to rotate a drone about an arbitrary axis. The formula we utilize is called Rodrigues' Formula(2). In 3 dimensions, we can define a rotation about an axis k by angle θ on a point v as

$$\mathcal{R}_k^\theta(v) = v\cos(\theta) + (k \times v)\sin(\theta) + k(k \cdot v)(1 - \cos(\theta))$$

This rotation is shown in the image below:

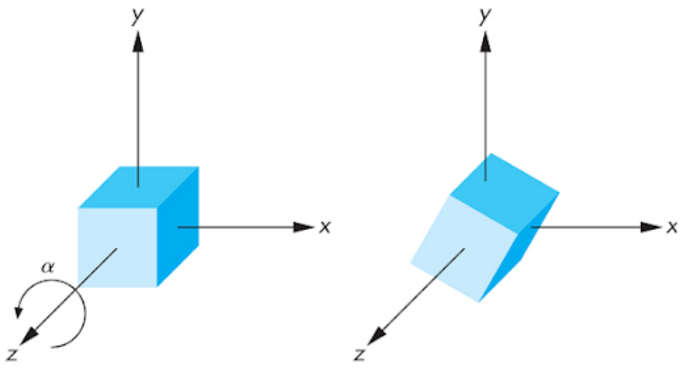


Figure 6: Rotating about an axis a shape in 3D (Credit: c-jump(3))

This method is the most useful of all; we can flap our wings, turn our body, wag our tail, etc.

The final method is dilation, which scales the position of the drones, moving them closer or further from the origin. Consider the position of an arbitrary done denoted as a vector, $v = (x, y, z)$. We define a dilation by a scaling factor of λ about a vector u as

$$D_u^\lambda(v) = \lambda(v - u) + v$$

This equation simply scales the difference between the points v and u, moving v closer or further from u. Dilation in 2D is shown below, scaling about the origin and an arbitrary point:

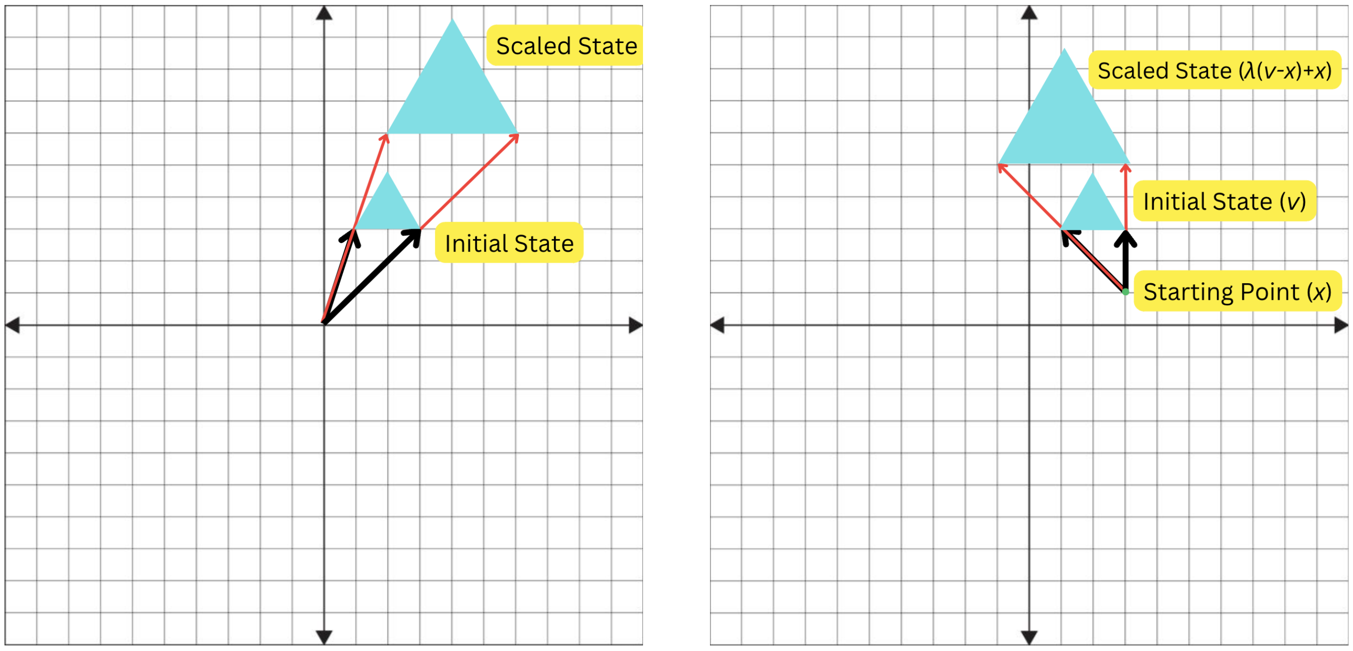


Figure 7: Dilating about origin and a point x in 2D

This method was utilized for making fire breathe out of the dragon. By scaling the fire, it would stay in the same direction, but get larger.

In order to make our entire animation, we utilize these transformations in combination with each other. For example, if we want our dragon to move around while simultaneously flapping its wings, we need to translate its entire body while also rotating the wing. However, in order to do this, we need to understand how we can rotate **only** the wing.

5.2.3 Drone Partitioning

Analyzing the image, it was clear that in order to implement specific transformations such as rotations to specific parts of the dragon, for instance flapping wings, we would need to isolate and classify specific parts of the dragon. In order to identify specific points, we decided to manually box them in by utilizing inequalities. Creating planes in 3D space, we could use them as boundaries to determine what points(drones) are apart of the wing, and which drones were apart of the body.

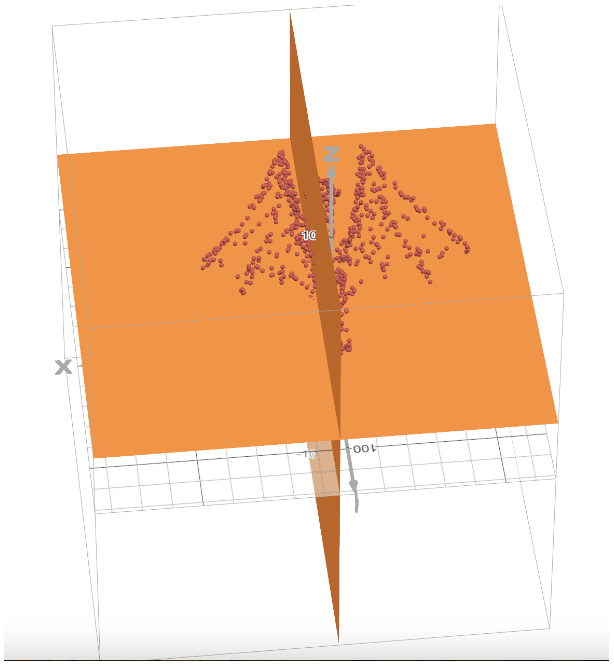


Figure 8: Dragon's left wing partition

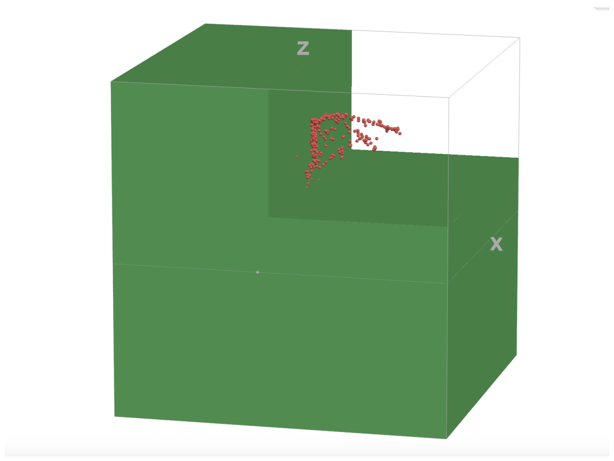


Figure 9: Dragon's left wing partition with inequalities

By checking certain points were greater than or less than certain boundaries, we create a box that can sort the points inside the one big list of vectors into smaller separate lists of vectors. On top of that, these planes can create boxes as big or small as we want them, so we can also use them to help us classify areas in which we may need to add elements that can help our movement appear more natural. For example, at the base of the wing, right near the body, we can add a ring of points such that we create a hinge joint, similar to how our elbow moves.

By stacking two to three of these joints, and applying the same rotation matrix with less magnitude, we can create a fluid stroke with our wings that seems to mesh well and appear natural with how an animal with solid muscles and bones would move.

5.3 Unique Image: Bad Apple!! Music Video (768 Drones)

For the unique image, we decided that our drones would show the classic “Bad Apple!!” music video, as it is a common to show it on unconventional displays. An example of this is YouTuber “Oby 1”’s video showing the ”Bad Apple!!” music video played on successive shots of carved apples (4).

5.3.1 Image Construction

Our screen display shows the video by changing the light level of the drones at the same speed as the frames per second (FPS) of the video. This display is created by first scaling down the video to a desired resolution (Figure 11), in this case 32 by 24 pixels, and then positioning the drones in a 2D grid in that resolution. Then, using the scaled down version of the video, we create a list of light values for each drone (which represents the pixels in the video).

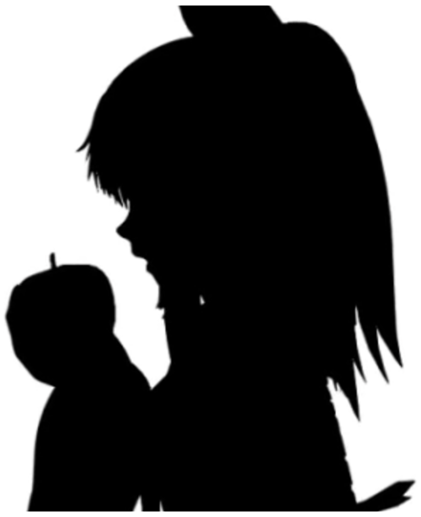


Figure 10: One frame from the original music video

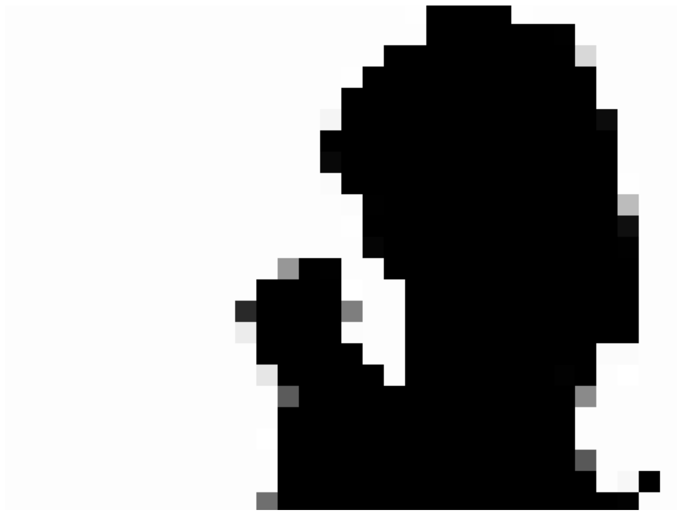


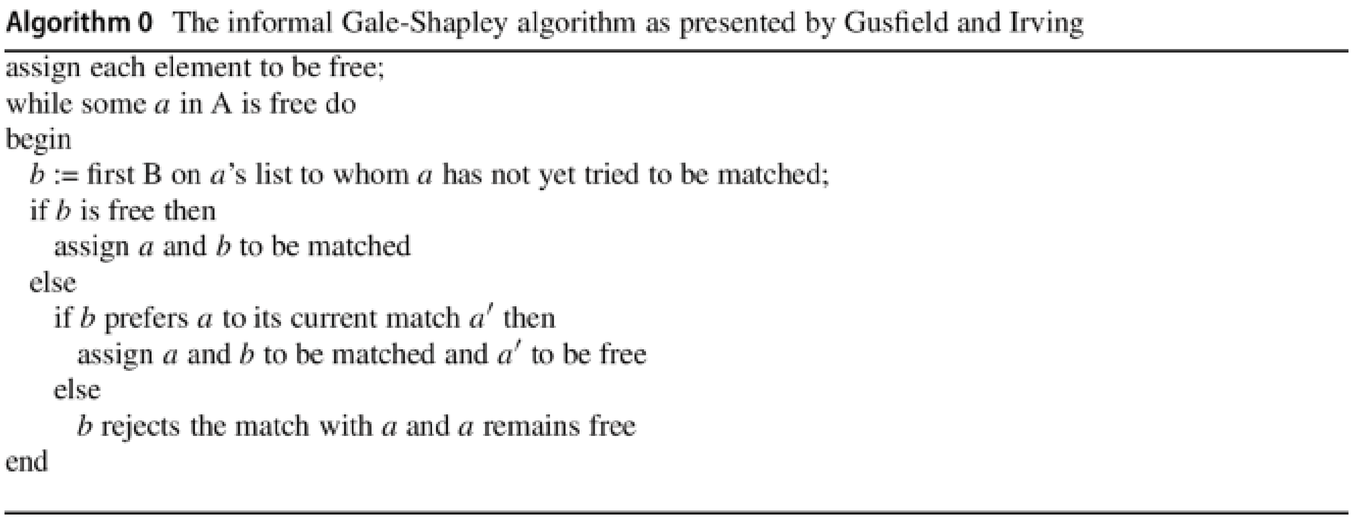
Figure 11: The same frame from the music video downscaled to 32 by 24

5.3.2 Flight Path

The flight path of this section of our light show is not really a path, since the drones wont be moving. Instead, we will describe the brightness of each drone in a list of length 6570(219 seconds in the video, 30 FPS).

5.4 Transitions Between Images

To transition from a Dragon to Bad Apple, we employed a system to match points from a start image to an end image that uses the stable-matching algorithm, shown below.



Algorithm 0 The informal Gale-Shapley algorithm as presented by Gusfield and Irving

```
assign each element to be free;
while some $a$ in A is free do
begin
  $b$ := first B on $a$'s list to whom $a$ has not yet tried to be matched;
  if $b$ is free then
    assign $a$ and $b$ to be matched
  else
    if $b$ prefers $a$ to its current match $a'$ then
      assign $a$ and $b$ to be matched and $a'$ to be free
    else
      $b$ rejects the match with $a$ and $a$ remains free
end
```

Figure 12: The Gale-Shapley stable matching algorithm

We start by assigning a ranking of points in the end list for each point in the start list, and vice versa. Then, we apply the algorithm to find a stable matching between the two lists. Then, we apply a linear transition between each start point and end point.

For example, if the start point is (x_s, y_s, z_s) and the end point is (x_e, y_e, z_e), the transition would look like

$$(x_s(1-t) + x_e t, y_s(1-t) + y_e t, z_s(1-t) + z_e t)$$

for a transition time $0 \leq t \leq 1$. We thus apply this to every single point in the start and end point with the same parameter t. Below are figures of the dragon in different transition times:

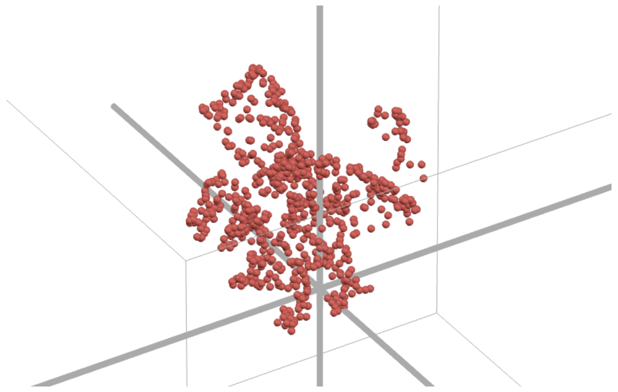


Figure 13: Dragon to screen transition at $t = 0.1$

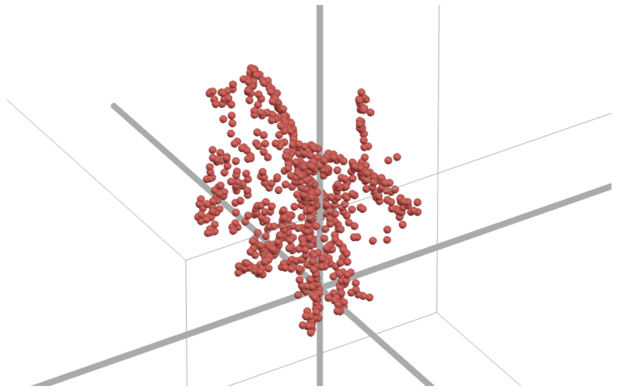


Figure 14: Dragon to screen transition at $t = 0.3$

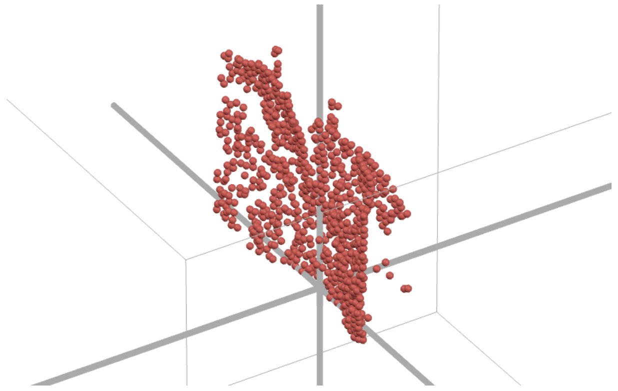


Figure 15: Dragon to screen transition at $t = 0.75$

5.5 Preventing Collisions

For preventing collisions, we have two solutions:

The first is to scale the drones' positions so that they are very far away from each other. Using this solution, through a program that we wrote to check collisions, we can see that a large scale size leads to no collisions. However, this solution is sometimes unreasonable since the larger scale means that the drones appear to move slower and use more energy (as more time is required). However, this shouldn't be a problem for our audience, as the drones are high in the sky, and the large scale changes to size will barely change the perception of our shapes.

The second solution to the problem is to find the collisions that would happen, and modify the flight paths such that one of the drones waits for a set interval of time before moving again. This is the better solution, as it does not come with the problems of the previous solution, and the slight stops will not impact viewer experience.

6 Model Testing

In order to actually test whether our model is implementable, we check to see if our predictions are correct in real life, and whether there are any observed collisions. To do this, implement our transformations through python code, create videos, and animate our points with respect to time in 3D-graphics engines such as Desmos-3D.

A multitude of images have been shown above, taken from graphs in Desmos-3D. For our Ferris Wheel, Desmos-3D has encoded features where it can automatically animate parametric equations. Referring back to the parametric equations inputted above5.1.2, we can see how as t changes, the final values of our parametric function are slight changes that add up over time.

Up above, we've shown you how we approached verifying transitions.

In order to plot our dragon, we took vectorized data from stl files, and inputted all those vectors into Desmos 3D. This allowed us to just experiment and play with still visuals and vector, but it wasn't too useful for animation. To allow animation, we utilized matplotlib, a python library that specializes in visualizations of data. By applying our transformation at extremely small scales, and incorporating a t time variable to each of the matrices or vector we added or multiplied, we could gather the small changes in our data that occured at different "frames". Putting each of those frames together while analyzing different time steps allowed us to not only create a confirmatory animation that our changes to our data work, but also allowed us to predict any points of collision.

7 Strengths and Weaknesses

7.1 Strengths of our Model

Ferris Wheel

1. Parametric equations in Ferris Wheel utilizes the strengths of Desmos 3D
   (a) Desmos 3D has clean animations, that are easy to use.
   (b) Desmos 3D is also easy to change, or add lines and equations to play around with quickly implementing ideas.
2. Parametric Equations provide shapes with points which are specialized to showcase overall change and trends over time, as points follow a function. This allows for an immersive and realistic model.
3. Parametric Equations can also be applied over any number of dimensions.

Dragon

1. The transformation model utilizes vectors, which are easy to work with in 3 dimensions, as they do not contain an overwhelming amount of information. Vector operations are not that computationally expensive in 3 dimensions, so it code should be easy to implement in the drones.

Bad Apple: This model has scalability, adaptability, and is safe to utilize. We can increase the FPS, resolution, or number of frames, and the Bad Apple model will still work. Furthermore, this model can adapt to any animation with any colors. Because the drones merely turn on and off, so no collisions can occur.

Transitions

1. Transitions occur quickly, and scale linearly in terms of time complexity, so the transition models aren't too computationally expensive.

7.2 Weaknesses of our Model

Ferris Wheel

1. A weakness of utilizing Desmos 3D is that it's not that powerful.

   (a) It gets harder to implement more complex shapes with parametric functions as we attempt to elaborate on our Ferris Wheel model.

2. Our model could be further iterated to contain further complexity, as there is no drone that is moving in all three dimensions in our current Ferris Wheel.

Dragon

1. Because of the fact that we used vectorized data to create a high quality 3D model of the dragon, and we add onto that the fact that we are manually manipulating the dragon using vector and matrix multiplication and addition, we get a limit on what we can realistically accomplish in the nuances concerning how the dragon moves, while still creating an animation that seems natural and fluid. If we copied synchronized movement of our points from a preexisting video, our dragon may have more fluid dynamics, however we may have less control and choice over what the dragon is actively doing.

Bad Apple and Transitions: By moving from 3D to 2D shapes, we lose information. Thus, our model isn't able to create a perfect representation of the transitions. For example, when transitioning from the Ferris Wheel to the Dragon, we need to turn on 270 more drones, so there are leftover possibilities that we could utilize. Furthermore, the animation will seem a bit strange from certain angles, as the drones have to line themselves up in order.. These same issues continue to be applied for our Bad Apple video, especially because we focus on the idea that everyone sees the same side of the drone made "screen". One possible solution to this problem for Bad Apple could be to have the screen parallel to the ground, such that spectators are below it. This may propose safety issues.

8 Citations/References

[1] https://skykam.co.uk/how-long-do-drone-batteries-last

[2] https://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula

[3] http://www.c-jump.com/bcc/common/Talk3/Math/Matrices/W01_0130_object_rotation_examp.htm

[4] https://www.youtube.com/watch?v=ywy-OwHejfs

[5] https://www.thingiverse.com/thing:218026

[6] https://github.com/Moonflower2022/drone-modeling

[7] https://www.desmos.com/3D/jgt5gfizgn

[8] https://www.desmos.com/3D/t4brevaxrc

[9] https://link.springer.com/article/10.1007/s10817-024-09700-x