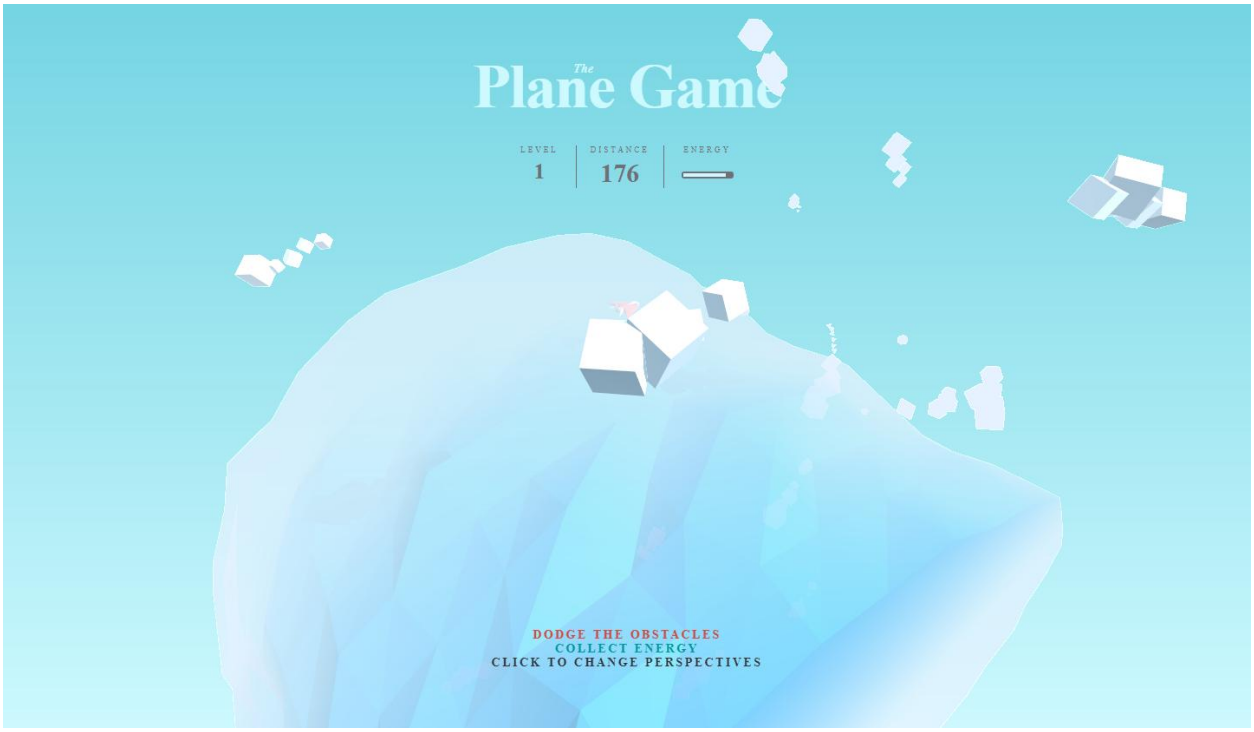
Name: Dan Ericson Atanacio
Student#: 7790804
Course: COMP3490

The Plane Game

The project I decided to do is create a 3D game using Three.js/WebGL. Prior to the development of this game, I followed a tutorial I found online about animating a basic 3D scene with Three.js by Karim Maaloul, “The Making of ‘The Aviator’: Animating a Basic 3D Scene with Three.js” [1]. After completing the tutorial, I then used what I had as a base to create my game. In the past, I have created several 2D “Shoot’Em Up” type genres of games where the player controls a ship and shoots enemies, and dodge objects so I was very inspired to re-create one of my games in 3D.

The Scene

One of the things I enjoyed the most about ‘The Aviator’ was the simplistic design of low polycount objects using a variety of simple shapes. I found this design to be the most appropriate theme for my game as it strongly resembles the theme of my past creations and I wanted to keep it simple for performance considerations.



Name: Dan Ericson Atanacio
Student#: 7790804
Course: COMP3490

The scene itself is a very simple setup. It consists of the plane, the sea, the clouds and enemies. One of the important things that the tutorial of 'The Aviator' taught me was to decide a colour palette for your game before starting to code the scene. Doing this helps keep your theme consistent. The theme I decided to go with is a mid-day ocean flying type of visual, so I went with a lot of varieties of blue and chose some contrasting colours to make the plane and other objects stand out.

The sea is basically a large cylinder geometry that just rotates about its z-axis to simulate the movement of the plane. An interesting point about the sea is the manipulation of the cylinder vertices to create waves. Each vertex along the length of the cylinder has its data stored and then modified to oscillate its position vertically and horizontally based on trigonometric functions. Some important parts are to remember to merge the vertices of the cylinder when creating the geometry to ensure the continuity of the waves, and to tell the renderer that the geometry has changed.

While the sky itself is a little more complex, it is also set up differently than other objects in the scene. To make things simple, the gradient background of the sky was created using a linear gradient background done in HTML through CSS. The sky itself is an object container that contains clouds, and the sky container rotates thus rotating all the clouds it contains. As for the clouds, each cloud is an object container as well. However, each cloud is a collection of box geometries of random sizes and of random amounts, where each box is placed close enough to each other, and rotated randomly to form a cloud.

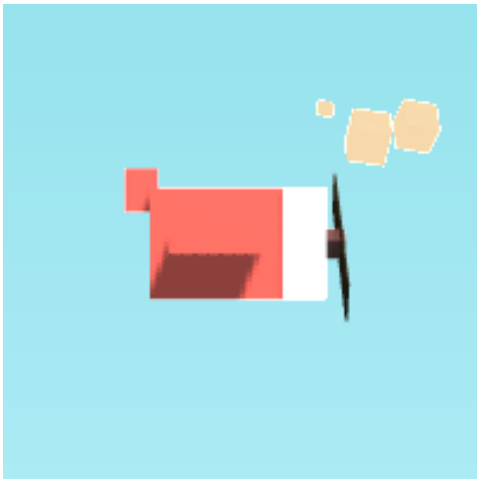
The lighting as well was setup up with only three types of lights, a hemisphere light, an ambient light, and a directional light. The hemisphere light combined with the directional light helped create a scene that resembles how a sun would have affected the scene. While the ambient light helped bring out the colours from the dull looking scene.

Finally, a blue fog effect was added to the scene to help with the visual theme of flying over the ocean and through the clouds.

Name: Dan Ericson Atanacio
Student#: 7790804
Course: COMP3490

The Plane

After completing the tutorial I was left with a very simple plane.



It was composed of several box geometries to make up the entire plane. One of most interesting things I learned from this was learning how to write functions expressions in JavaScript as opposed to function declarations, such that I can create the entire plane separately with its own variables and assign it to a variable as a form of Object Orientation similar to classes. By doing this I was able to individually control the airplane's propeller, and rotate it.

However, this plane was not good enough for me. Since I wanted to re-create one of my old games, I took the time to follow one of my old designs I drew for a plane in my past projects and try to create it in 3D.

Name: Dan Ericson Atanacio
Student#: 7790804
Course: COMP3490

This is one of my old drawings that I used in my old 2D games.

And on the right is a screen shot of my old design re-created in 3D. I changed the design slightly because my 2D games had a space theme to it, as my 3D game was no longer in space I wanted to make it look a bit more realistic.

The entire plane was created almost entirely of box geometry, excluding the jet which is a cylinder that contains rotating propellers scaled and fitted from the previous model, and the nose which is a pyramid.

One of the challenges of creating the plane was manipulating each vertex to create the weird geometry that I needed. Furthermore, creating the pyramid for the nose was a challenge as well but it was completed with the help of some source code online that simply creates a pyramid with sizes of one in each vertex and I simply used a scaling matrix to make it the size I wanted and positioned it accordingly [2].

Another interesting part about the plane is the flag attached to the back. The flag is composed of 12 cubes equally spaced apart in three rows and four columns, and a flag pole. The interesting part is that depending on the speed the plane is going, the flag oscillates based on a cosine function. The setup for creating the flag was based off of the tutorial for 'The Aviator' and it was modified to oscillate in such a way that I desired.

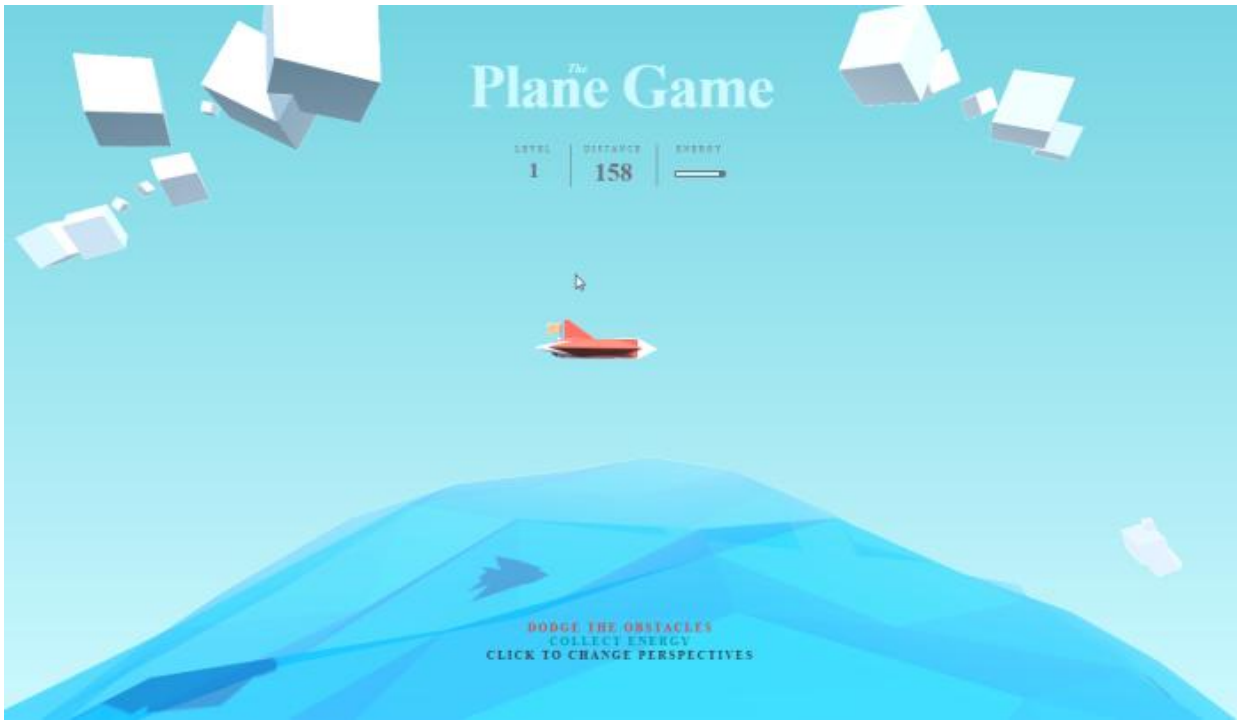
Name: Dan Ericson Atanacio
Student#: 7790804
Course: COMP3490

The Game

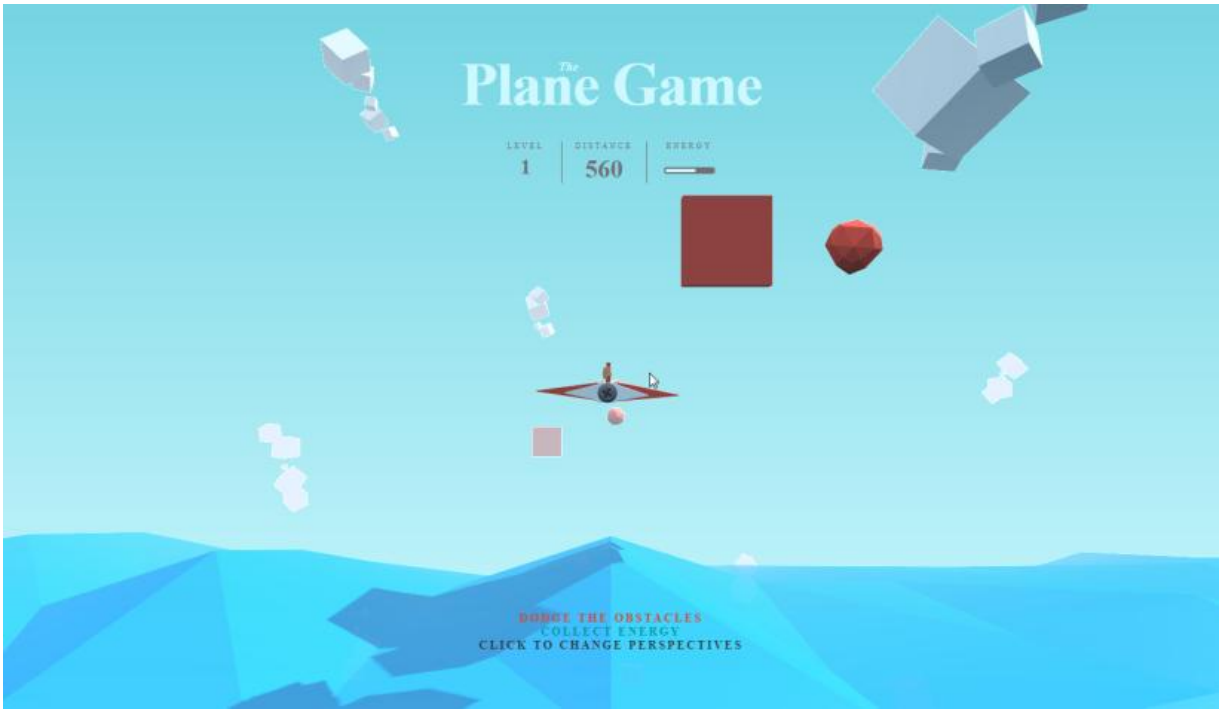
As for the functionality of the game, I later decided that the game should take on a slightly different concept than the traditional “Shoot’Em Up” type genre to really show off the abilities of being able to move in 3D.

The objective of the game is that you are flying a plane controlled by the mouse and you must dodge obstacles and fly as far as you can before running out of energy. The player is required to collect more energy to keep flying. Also, the further you fly, the more difficult the game becomes.

In the tutorial of ‘The Aviator’, it was designed to be a side scrolling game with a view perpendicular to where the plane is flying.



But that wasn't enough to really show off the type of 3D game I wanted to create. So, I decided that I wanted to improve the dynamic of the game by adding an alternate perspective that can be switched between the two freely, while also changing everything to have a depth aspect to them.



My intention of being able to change the view is to change how the game is supposed to be played. By providing core reasons to be in each different perspective, it forces the player to actively switch between the two so that the player will be able to avoid the obstacles, as well as showing off some of the design features of the plane, like the flag and jet that you won't be able to see from the side view alone.

An interesting effect done to enhance the visuals of the game is the use of Tween.js. Tween.js allows for smooth animations in Three.js, and it was used to help create some nice-looking particle effects when colliding with objects. When particles are triggered to spawn, it creates an array of particles which are simply coloured tetrahedron geometry that use Tween.js to rotate, scale to fade out, and to send each tetrahedron in random directions.

Finally, as the final result of the game is far off from the traditional "Shoot'Em Up" type genre because it does not exactly involve shooting any enemies, it is however a great base to work with to develop any future projects I decide to do. Especially once I decide to fully re-create one of my past 2D "Shoot'Em Up" games I will be making use of this, as I learned a lot from the creation of this game.

Name: Dan Ericson Atanacio
Student#: 7790804
Course: COMP3490

References

1. https://tympanus.net/codrops/2016/04/26/the-aviator-animating-basic-3d-scene-threejs/
2. https://stackoverflow.com/questions/26418591/how-to-make-a-rectangular-pyramid-in-three-js-r68