

**Three.js – Project 3**

Vincent Testagrossa

CMSC 405 6380 Computer Graphics (2232)

Professor: Lauren King

Due February 21st, 2023 11:59 PM

Contents

[Project Description 3](#_Toc127783132)

[Program Usage 3](#_Toc127783133)

[Testing 5](#_Toc127783134)

[Figures 7](#_Toc127783135)

[Lessons Learned 13](#_Toc127783136)

[References 14](#_Toc127783137)

# Project Description

This project creates a unique 3D animated scene using the Three.js library. It includes two light sources, and many different shapes (a few cubes and many spheres). There are GUI widgets that control light intensity and visibility, mesh visibility, and turns on/off animations. The scene is a spaceship flying through an infinite field of donuts in space (with a skybox displaying the space background texture).

# Program Usage

Running this program requires that a webserver hosts the directory where index.html resides. This project was tested using the http-server package for node.js, but others may work as well. A [list of options](https://threejs.org/docs/#manual/en/introduction/How-to-run-things-locally) have been made available on the Three.js website.

**Installation**

Installation of http-server requires the [installation of node.js and npm](https://docs.npmjs.com/downloading-and-installing-node-js-and-npm). Once completed, the following command will install http-server globally so it can be run from anywhere in the filesystem:

npm install --global http-server

A script is included to run the npm package [http-server](https://www.npmjs.com/package/http-server) on port 80 using the following command in powershell:

http-server -p 80

[Figure 1](#Figure1) shows the command being run from within the same path as index.html (the “Project 3” directory), but this may be a different path depending on where the files were extracted. [Figure 2](#Figure2) shows the output of the server when it is first run and no requests have been made from the browser. This will vary depending on the user’s network settings. The user may use [*http://localhost*](http://localhost)*,* [*http://127.0.0.1*](http://127.0.0.1), or <http://[local_ip_address>] in order to load the page once the server is running and hosting the “Project 3” directory. In [Figure 3](#Figure3), *localhost* was used.

**Using the Program**

The scene uses OrbitControls to control the position of the camera with the following controls:

Rotate:

Hold left click within the scene and move the mouse to rotate the view.

Zoom:

Scroll the mouse wheel up or down to zoom in or out.

Pan:

Hold right click, or hold shift (or CTRL) and hold left click to pan the scene in the direction the mouse is moved.

**GUI**

Click the category to expand the folder you wish to view. Sliders may be adjusted by holding left click and dragging left or right, or may be manually changed by entering a number in the text field to the right. Checkboxes work like other checkboxes, in that they need to be clicked and released to toggle the checked state. To collapse the folders, the name of the folder may be clicked again. To hide the controls, click on *close controls.* See [Figures 4-6](#Figure4) for examples.

# Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Case | Input | Expected Output | Actual Output | Pass / Fail |
| 1. Load Scene | Run the server script and type “localhost” into the address bar, then press enter. | The scene loads with the GUI controls expanded in the top right. The skybox loads with the textures and the field of donuts generates and starts the animation, then the ship loads with the thrusters animating. | See [Figure 7](#Figure7). | Pass |
| 1. Adjust rotation with OrbitControls | Left click and hold in the scene, then move the mouse left, right, up, and down. | The scene rotates right, then left, then forward, then backwards, centered around the ship. | See [Figures 8-11.](#Figure8) | Pass |
| 1. Adujst zoom and pan with OrbitControls | Scroll up, scroll down, right click and drag left/right | The scene zooms in, then out, then pans left/right. | See [Figure 12](#Figure12). | Pass |
| 1. Adjust Lighting with GUI controls | Make adjustments to the lighting sliders, enable and disable each one. | Sliders adjust intensity of lights, while checkboxes disable themcompletely. | See [Figure 13](#Figure13). | Pass |
| 1. Enable/Disable models with GUI controls | Select each model in the list, and press the button for the donuts. | Each model is no longer visible when the checkbox is unchecked.  The donut effect toggles when the button is pushed. | See [Figure 14](#Figure14). | Pass |
| 1. Stop and start animation | Press the stop animation, then press start animation | Scene animation stops and controls are disabled. GUI controls remain functional. | As expected. | Pass |

# Figures

Text

Description automatically generated

Figure 1-Running http-server from the directory that contains the index.html file

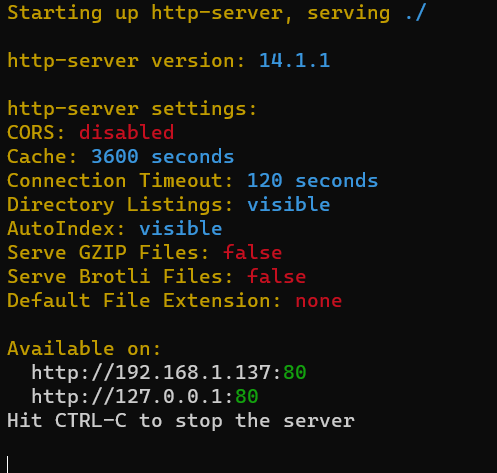


Figure 2-The console will look similar to this when the server is running



Figure 3-Once the server is running, localhost is entered into the address bar of a browser and the page loads.

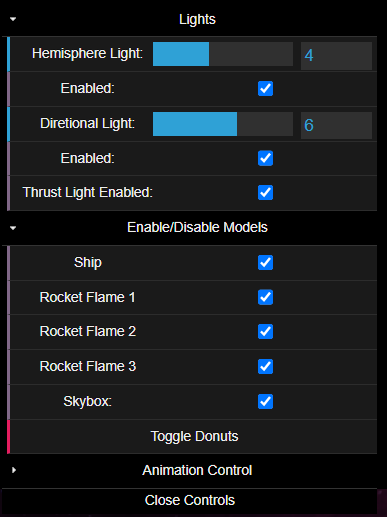


Figure 4-GUI with two folders opened: Lights and Enable/Disable Models

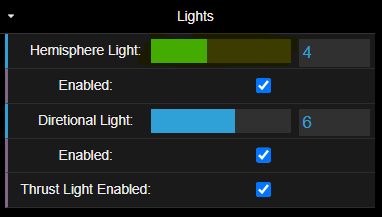


Figure 5-A slider for Hemisphere Light is highlighted.

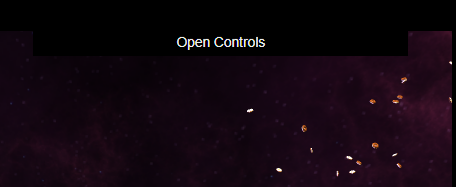


Figure 6-GUI view when Close controls was selected.



Figure 7-The scene is loaded.

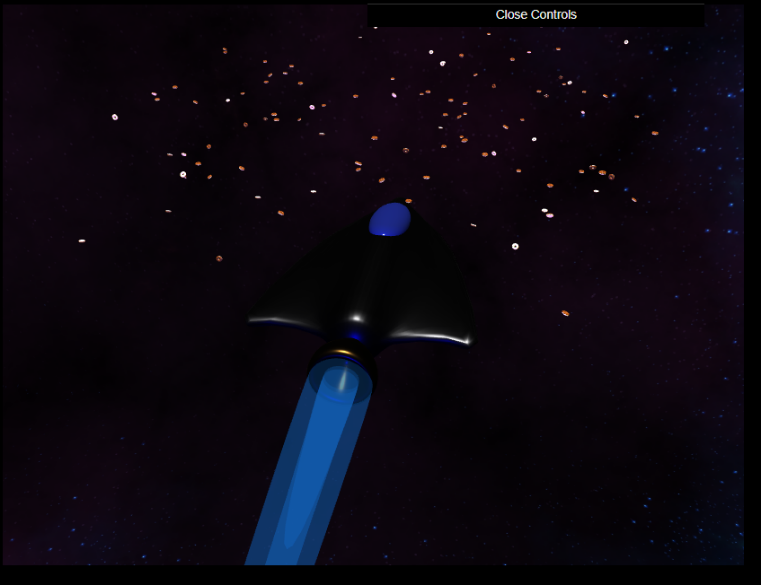


Figure 8-Scene is rotated to the right.

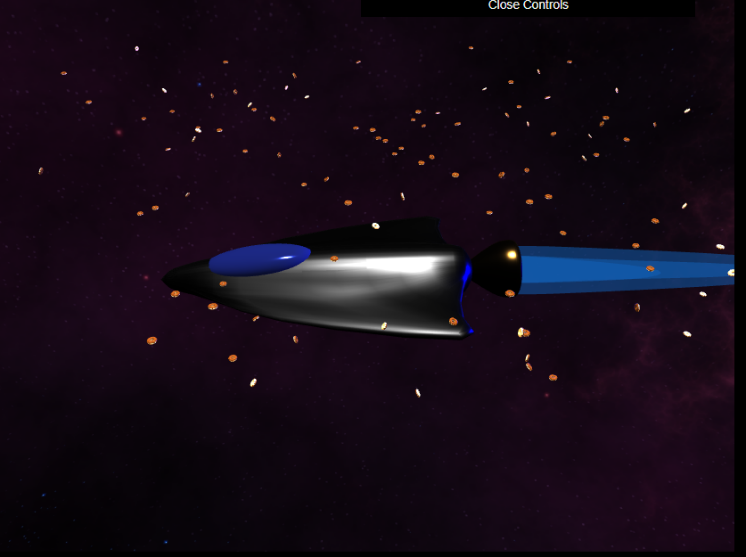


Figure 9-Scene is rotated to the left.

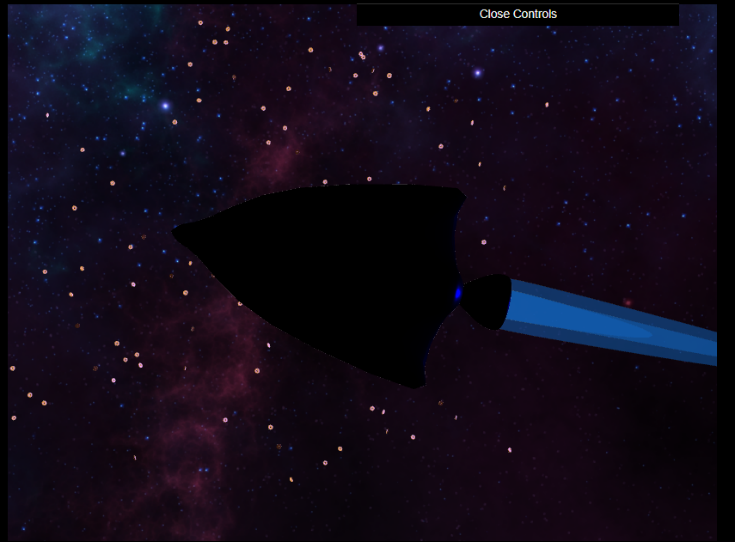


Figure 10-Scene is rotated to the bottom.

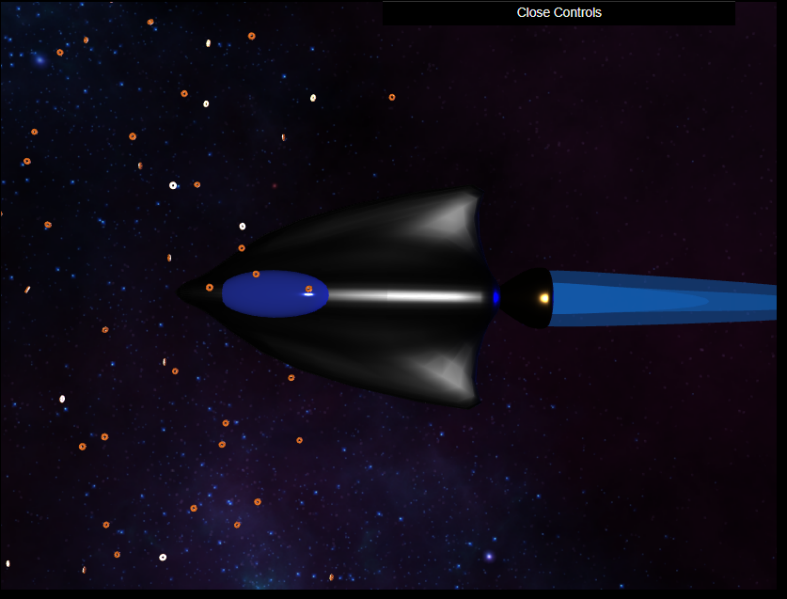


Figure 11-Scene is rotated to the top.

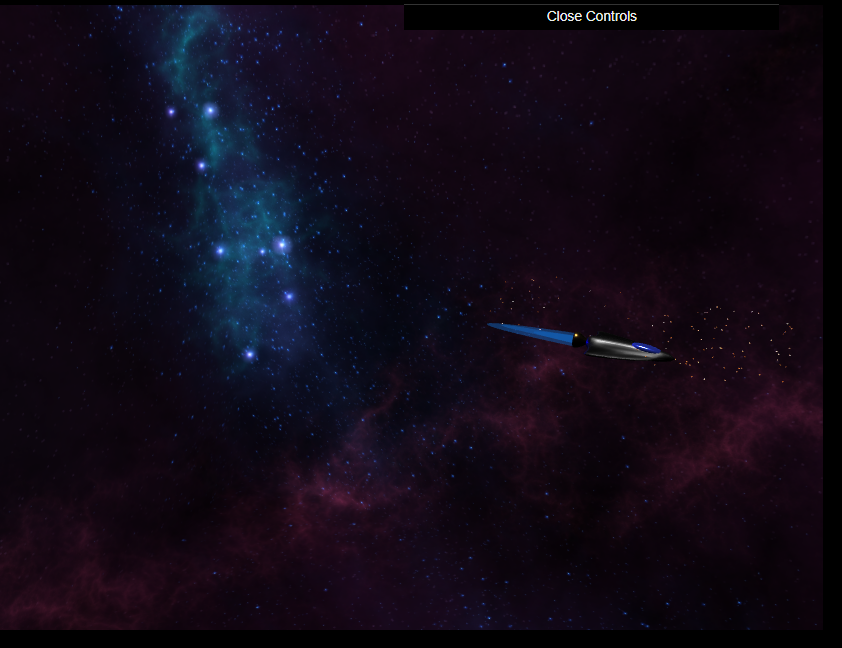


Figure 12-Scene is zoomed out and panned to the left.



Figure 13-Hemisphere light is at maximum, but disabled and the thrust lighting is not active, so the blue glow is missing.

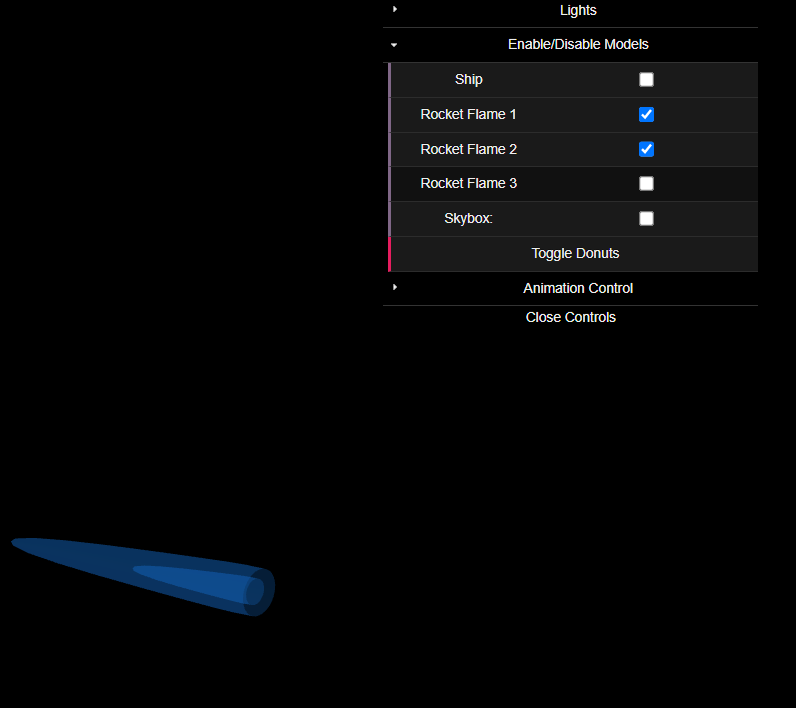


Figure 14-Only two of the 3 Flame meshes remain. The skybox, the ship, and donuts have been removed.

# Lessons Learned

1. I wanted to load my own models from blender like the examples, but the JSON format has changed since Blender has changed versions. While searching for solutions, I learned about the GLTF format through a tutorial that I followed to create the structure of this project (Blue, n.d.). Structuring the files this way turned out to be a very good choice, because once I learned how it was done, adding and modifying different files became trivial.
2. The Blender tutorials by the Blender Guru on youtube were also invaluable. I had no idea the software has come so far. The last time I used the program was about 15 years ago, when the software was very new, barely worked and not user-friendly at all. I’m glad I took this route, because I was able to quickly model the ship once I had gone through his donut tutorials.
3. Destructuring assignments are something I had never encountered before. It took a little while to understand them at least a little bit, but I still had problems trying to get them to work with module level variables (so I could work with properties in other functions within the module, mostly for the GUI functions).
4. Three.js has a lot of built-in modules, but GUI is separate. I was going to build my own with HTML controls, but I found dat.gui and looked at how easy it was to connect via the API (Mccurdy, 2019). This was another instance of the structure from the book working in my favor, since I could just import the module and work with it with minimal effort.
5. In order to get the files to run locally (without installation of the modules), I had to change some of the import statements in the three.js files to match the folder structure I had created. It is able to run on a platform without npm/node.js, but an http server will be needed to load the project. This made the project larger, but I wasn’t sure if there was going to be an issue since I didn’t use the provided three.js build.
6. The skybox was loaded in a very similar way to the provided examples, however there were a couple of helper functions in a tutorial (Codinhood.com, 2020) that I found that made everything a lot cleaner.

# References

Blue, L. (n.d.). *Discover three.js*. Retrieved from Discover three.js!: https://discoverthreejs.com/

Codinhood.com. (2020, March 27). *How to Create a Skybox with Three.js*. Retrieved from Codinhood: https://codinhood.com/post/create-skybox-with-threejs

Mccurdy, D. (2019, March 30). *dat.GUI API*. Retrieved from github: https://github.com/dataarts/dat.gui/blob/master/API.md