The Bubble Machine

Madeline Gent & Carolyn Bergdolt

Graphics Final Report

Project Summary:

Our program is an interactive bubble-blowing experience. When the user begins the program, they are placed in a beach environment which they can explore by clicking and dragging around the browser window. To blow a bubble, the user selects the size ("big" or "small") and clicks the "Blow Bubble" button, afterwhich a bubble appears and floats away in a random direction. To change the bubble blowing environment, the user can click "Change Background".

In implementing parts of this project, we relied heavily on examples provided on threejs.org. A detailed description of our reliance on these examples is documented at the end of this report.

Main Functions:

1. Click & drag environment exploration
2. Blow bubble
    a. User has choice of bubble size
3. Change environment

How to Use:

To open the program:

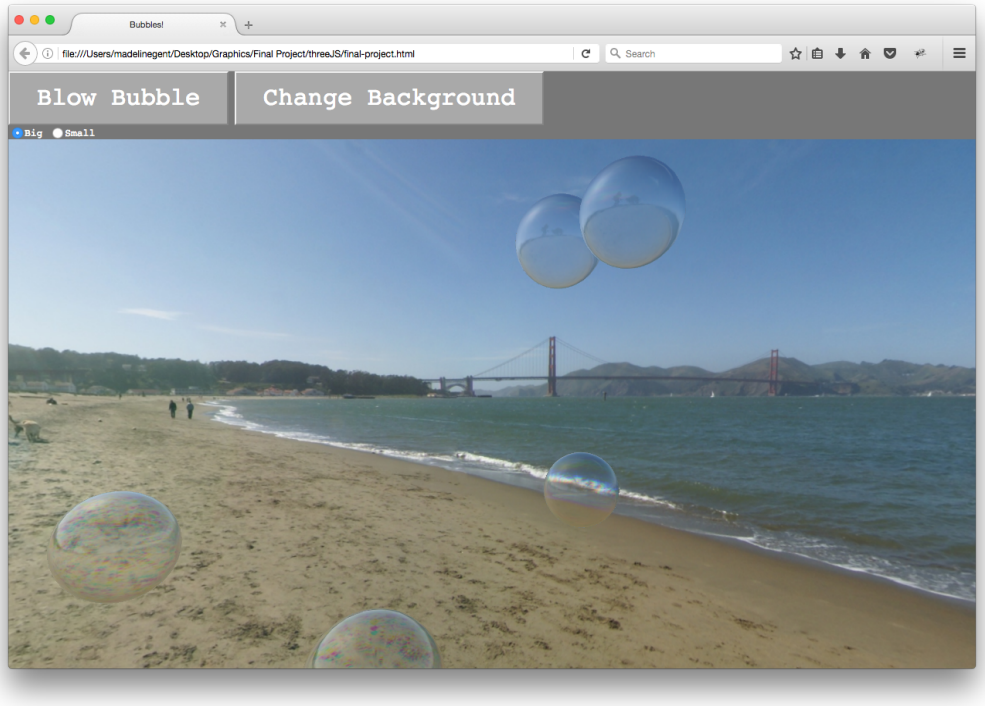
- Open the folder named "Final Project"
- Open "final-project.html" in Firefox

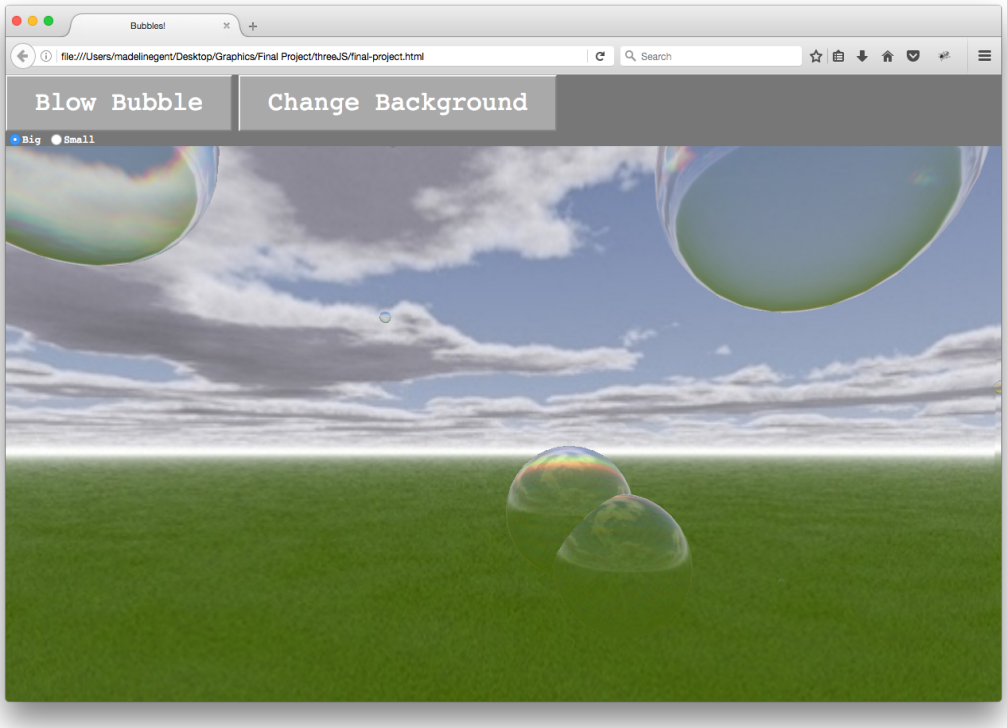
To interact with the program:

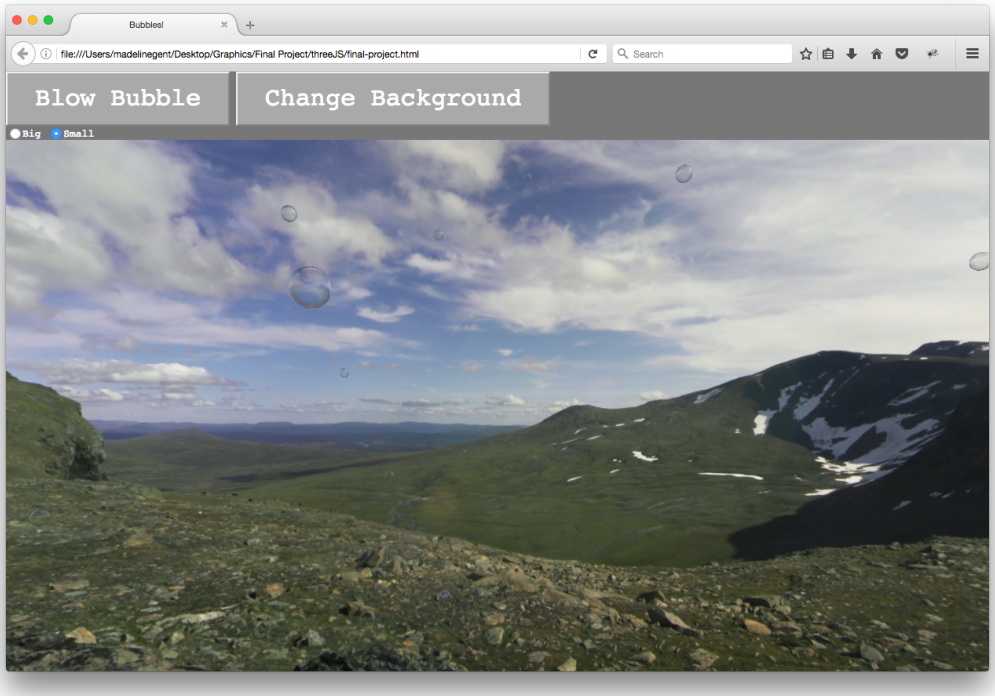
- To explore the environment, click on the browser window and drag the mouse in the direction you want to turn. For example, to look to the left, click and hold the mouse while moving the cursor to the left.

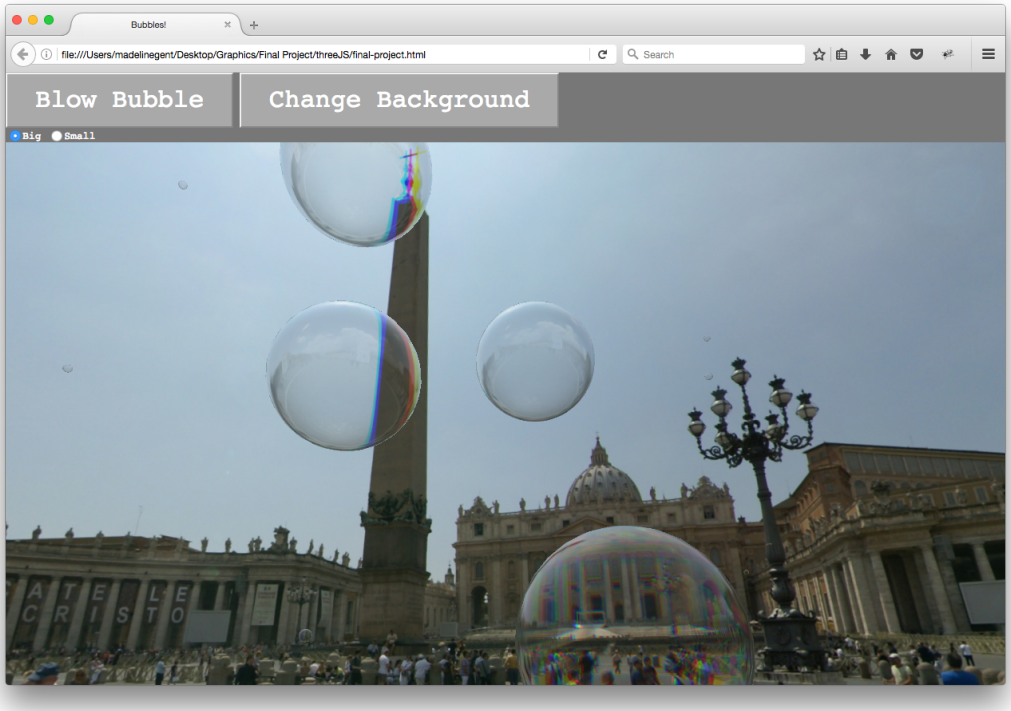
- To blow a bubble, select bubble size from the radio buttons in the top left corner ("big" is the default size). Next, click the "Blow Bubble" button directly above the radio buttons.
- To change the environment, click the "Change Background" button next to the "Blow Bubble" button.

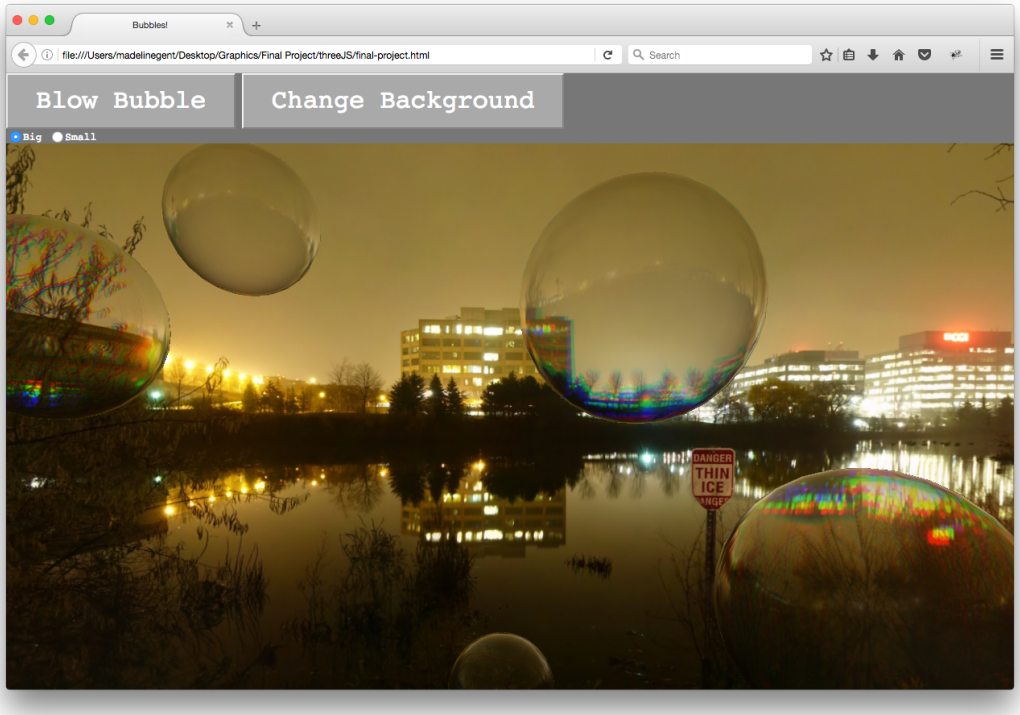
The screenshots below show the different sized bubbles in each environment.




Bubbles!
file:///Users/madelinegent/Desktop/Graphics/Final Project/threeJS/final-project.html
Search
Blow Bubble
Change Background
Big
Small


Bubbles!
file:///Users/madelinegent/Desktop/Graphics/Final Project/threeJS/final-project.html
Search
Blow Bubble
Change Background
Big
Small


Bubbles!
file:///Users/madelinegent/Desktop/Graphics/Final Project/threeJS/final-project.html
Search
Blow Bubble
Change Background
Big
Small


Bubbles!
file:///Users/madelinegent/Desktop/Graphics/Final Project/threeJS/final-project.html
Search
Blow Bubble
Change Background
Big
Small
DANGER
THIN
ICE

Technical Challenges:

Bubble Texture

The first big challenge we encountered was how to make the bubbles look realistic. Because we were unfamiliar with ThreeJS, we didn't have a comprehensive knowledge of all the available textures to us and how to use them. We started by using the default ThreeJS mesh material ("THREE.MeshBasicMaterial") and no specific shader. This did not give us the effect we desired. The bubbles looked too solid and heavy, and they were not as transparent as we would've liked. After a lot of research into the issue and some ineffective attempts at applying a bubble-like texture, Carolyn found the fresnel shader. She applied this shader to our uniforms and material, and it finally made the bubbles look like what we'd imagined.

Bubble Origin

When we first brainstormed for this project, we wanted the user to be able to click anywhere on the screen to make a bubble appear. However, after working with the code, we found that this was not going to be feasible in the way we'd hoped. The main problem was that the coordinate system that the mouse resides in (which returns the position of the mouse) did not correlate with the coordinate system of the environment. So, if one were to click in the middle of the computer screen, those 2D coordinates could not be effectively mapped to the 3D environment. Also, issues arose when the user would click and drag the window to see more of the environment, since the computer interpreted this action as blowing a bubble as well. After many hours of experimentation and discussion, Maddie came up with the idea to change the bubble blowing action from a screen click to a button click. This eliminated the problem of the conflicting coordinate systems as well as the environment drag/bubble blow confusion. If we had chosen to use a static 2D image as the background instead of a cube map, it would've been more feasible to implement screen-click bubbles, but we thought that the cube map feature was more important.

ThreeJS

Working with ThreeJS was a technical challenge in itself. Since we did not spend time working with ThreeJS in class, there was a bit of a learning curve at the beginning of the project. In order

to combat this, both of us spent time looking at ThreeJS tutorials and documentation online (https://threejs.org/). After gaining a basic understanding of the library, we found that ThreeJS was actually simpler to use in some ways, specifically when creating the bubble spheres and applying textures. The abundance of online documentation and example code also helped provide us with guidance if we got stuck.

Team Members:

To work on this project, Maddie and I got together at a couple different times, and we each worked on different functionalities of the program, integrating our changes whenever one of us had a breakthrough with the implementation. I feel like Maddie and I collaborated very well on this project, and each of us pulled our own weight as far as implementation and coding were concerned. The specific task division is outlined below:

Carolyn:

- Applied fresnel shader to bubbles
- Found and applied background cube maps
- Implemented click-and-drag environment exploration

Maddie:

- Created bubbles
- Implemented bubble motion
- Created top menu bar for user interaction
-

What We Learned:

- How to effectively use the ThreeJS library and API
- The importance of shaders in creating realistic images
- How to make object and environment textures work together to create lifelike scenes

Use of Example Code:

We used several example code files from threejs.org in our creation of this project. To start, we used *webgl_materials_cubemap_ball_reflection.html* as our base code. However, as we made progress, we modified a good chunk of it to meet our purposes. The parts that remained the same are the basic cube map implementation, sphere creation, and sphere rendering, and most of the basic essential framework of the program. We modified the shading of the spheres to implement fresnel shading, and the file we used for reference on that part was *webgl_materials_shaders_fresnel.html*. We didn't like the way the environment shifted with simple mouse movement, and wanted the user to be able to pan around while clicking and dragging. The *webgl_materials_envmaps.html* file provided us with the example code we needed to implement this feature.