Justin Lu
Graphics I
5/2/2021
Dr. Krishnan Seetharaman

Final Project Report

For the project, I was successfully able to create 9 basic shapes in the canvas, applying translation, rotation, and scale. Allowing general tools such as copy, paste, and unlimited undo. Also, I was able to successfully save the canvas as JPEG and JSON file to load the JSON file back to the canvas. For the graphic implementation, I did not rely on any third-party JavaScript library to implement the features. As for general purposes, I used jQuery, a lightweight library good for traversing the HTML tags, and bootstrap-icon library for borrowing the icons to use it on the toolboxes. Also, I use a WebGL-Lint, a very useful WebGL debugger that helps me a ton when solving WebGL-related bugs.

The general structure of the program is that I created manager classes that keep track of various information within the program. For example, MeshManager keeps track of all the meshes that are present in the canvas and responsible for adding and deleting the shapes as requested. RendererManager is responsible for drawing the shapes on the canvas by referencing the list of shapes from MeshManager. Other than these two, the program has a UndoManager, class keeping track of the snapshots of all the previous modifications. UIManager, which responsible for updating the input from the user updates the UI accordingly. And last but not least, SelectionManager, a class that keeping track of all the mesh selected in the canvas.

The most difficult part of this project was creating a copy of the instances, not only for copy and paste but also for saving as a JSON file. The way JavaScript use reference to assign the object and arrays unlike number and string, and gave me a lot of frustration. For example, I have a Curve class that contains all the necessary information to draw it on the screen such as position, transformation matrix, index buffer, and most importantly, buffer id for passing the data into the GPU. To create a copy of the curve class, I cannot simply create a new Curve class and pass in all the array and object as a parameter because JavaScript will pass these objects as a reference and now two instances are sharing the same one resources, leading to an issue when one is deleted, other will lose the data as well. To solve the problem, I take the approach Three.js, an open-source 3D graphic library. In their source code, they have a function called copy and clone in every class. Copy takes in an instance of the same class and traversals through the properties to assigning them by values, and the clone will instantiate a new class of itself while calling copy function and passing itself as a parameter (sample code below). Since every class has the clone function. There is less worry about the accidental sharing of the resources.

```
clone() { return new BufferGeometry().copy( this ); }
```

As for the bugs that I couldn't fix, the scaling and rotation of the line, polyline, and curve are not accurate. I was attempting to solve this by allowing the user to modify the lines using the verticies (that's why the line selection is in dots and not a solid line), but unfortunately, I don't have enough time to implement it.

As for the extra credit. SelectionManager class allows to select of multiple shapes and modify at once, and select and de-select all the shapes using A-key. Also, I added the ability to change the color of shapes and background canvas.