Assignment 2

Amith Sai Konda
IMT2018037
IIITB
Bangalore
amith.sai@iiitb.org

I. Problem Statement

Design a simple 3-D model. Render and manipulate the created 3-D model.

II. Selected Models

I have selected pyramid, urn and mug models from the website given in the question.

III. Files

Files created and their description are shown below.

- index.html is the html file for creating the document in webpage.
- app.js is the main javascript file where all the modes and eventlisteneres are implemented.
- axes.js contains the information about the axes(color,etc) and a draw function.
- pyramid model is imported to pyramid.js and a draw function is implemented in it.
- urn model is imported to urn.js and a draw function is implemented in it.
- mug model is imported to mug.js and a draw function is implemented in it.
- vertex.js contains the vertex shader.
- fragment.js contains the fragment shader.
- tranform.js has all the matrices for the transformation of the models.
- renderer.js is used to create the canvas and render it in webpage.
- shader.js is used to create the program and link the program with vertexshader and fragmentshader.

IV. Description

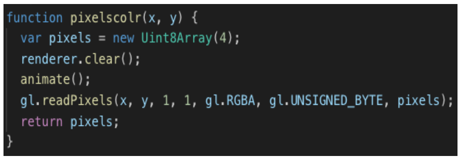
- when we compile the program, the webpage renders an axis and the 3 selected models at the origin.
- We have 10 modes in this assignment, d-h,m,x,y and z and the description for this modes is shown below.

1) Mode ”d”
   - In this mode we have to place the selected models at the vertices of a triangle.
   - Triangle can be anything with centroid (0,0,0).
2) Mode ”e”
   - In this mode we have to place the selected models at the middle of the edges of the triangle.
   - Triangle we use here is same as the triangle used in mode ’d’.
3) Mode ”f”
   - In this mode we have to rotate the selected model by 90 degrees each along an axis.
4) Mode ”g”
   - In this mode we have to scale the selected models.
5) Mode ”h”
   - In this mode we can select a model we want.
6) Mode ”m”
   - In this mode we can minimize the scale of the selected models.
7) Mode ”x”
   - In this mode we can drag the model along x axis.
8) Mode ”y”
   - In this mode we can drag the model along y axis.
9) Mode ”z”
   - In this mode we can drag the model along z axis.

V. Approach

1) Approach for mode ’d’
   - I have selected 3 points in the canvas which will form a triangle with centroid near the origin.
   - Points I have chosen are (5,5,0),(0,-5,0),(-5,0,0)
   - I translated the models to that points using translate matrix, created at transform.js.
2) Approach for mode ’e’
   - First I found the mid points of the edges of the triangle.
   - I translated the models to that points using translate matrix, created at transform.js.
3) Approach for mode ’f’
   - I have aligned an axis to each of the selected models.
   - Later, I rotated each selected model by 90 degrees using rotate matrix, created at rotate.js
4) Approach for mode ’g’
   - At first I have scaled the pyramid and urn to 1. But I scales mug to 30 because, it is so small to see.
   - Later, I scaled 0.5,2,3 to mug,urn and pyramid respectively.
   - I used scale matrix written in transform.js.

5) Approach for mode ’m’
   - I scaled 2,0.5,1/3 to mug,urn and pyramid respectively.
   - I used scale matrix written in transform.js.
6) Approach for mode ’h’
   - A function was implemented to read the color of the pixel on the canvas.



```
function pixelscolr(x, y) {
  var pixels = new Uint8Array(4);
  renderer.clear();
  animate();
  gl.readPixels(x, y, 1, 1, gl.RGBA, gl.UNSIGNED_BYTE, pixels);
  return pixels;
}
```

   - Later it is check with the color of the model.
   - If the color is matched with the pixel color, it is changed to black color.
   - Changing of black color is done at vertex shader by using a flag pick.
7) Apprach for mode ’x’
   - I am rotating the camera in a circle around the model along xaxis.
   - I have stored the staring point of clicking and the present point of dragging in two variables.
   - Later, if the difference between them is negative then I am decreasing the angle of rotation by 1. Otherwise I am increasing it by 1.
   - In short, I am changing the position of camera.
   - Since I am moving in a circle, I am changing the position of the camera using

$$ycos(thetha) + zsin(thetha)$$
$$zcos(thetha) - ysin(thetha)$$

     since I have the angle.
   - Later, I am setting the new camera matrix to all the models.
8) Approach for mode ’y’
   - I am rotating the camera in a circle around the model along yaxis.
   - Appraoch is same as the mode x.
   - Equations used are

$$xcos(thetha) + zsin(thetha)$$
$$zcos(thetha) - zsin(thetha)$$

9) Approach for mode ’z’
   - I am rotating the camera in a circle around the model along zaxis.
   - Appraoch is same as the mode x.
   - Equations used are

$$xcos(thetha) + ysin(thetha)$$
$$ycos(thetha) - xsin(thetha)$$

VI. Questions

1) *To what extent were you able to reuse code from Assignment 1?*

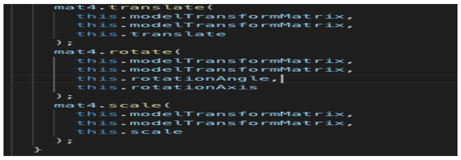
   I used my last code as a template for this. I reused most parts of shader.js, renderer.js, style.css from the last assignment. I haven’t used matrices in the last assignment, so most of the code in this assignment was written from scratch.

2) *What were the primary changes in the use of WebGL in moving from 2D to 3D?*

   In 2D we look the image from the z-axis. But in 3d we should add perspective because there will be an image in the 3D. Basically perspective is if we take a 3D image, the farthest object should appear smaller and the nearest object should be larger. Here we are defining nearer and farther with respect to a camera. So when we come to 3D, perspective and camera are two major things to look at.

3) *How were the translate, scale and rotate matrices arranged such that rotations and scaling are independent of the position or orientation of the object?*

   I have implemented separate matrices for translating, rotataing and scaling. So when we are doing something we are not considering other things.



```
    mat4.translate(
        this.modelTransformMatrix,
        this.modelTransformMatrix,
        this.translate
    );
    mat4.rotate(
        this.modelTransformMatrix,
        this.modelTransformMatrix,
        this.rotationAngle,
        this.rotationAxis
    );
    mat4.scale(
        this.modelTransformMatrix,
        this.modelTransformMatrix,
        this.scale
    );
}
```

VII. Conclusion

This assignment helped me to learn

- Creating 3D models (using a modeling tool).
- Importing 3D mesh models.
- Transformations of 3D objects.
- View transformations.
- Picking model objects and their constituent parts in 3D.

VIII. References

- https://webglfundamentals.org/
- https://www.youtube.com/watch?v=kB0ZVUrI4Aw&ab_channel=IndigoCode
- https://1stwebdesigner.com/15-css-background-effects/
- https://webglfundamentals.org/webgl/lessons/webgl-3d-perspective.html
- https://webglfundamentals.org/webgl/lessons/webgl-3d-camera.html

- https://www.youtube.com/user/AndrewPPrice
- https://www.khronos.org/registry/OpenGL-Refpages/es2.0/xhtml/glReadPixels.xml
- https://github.com/Amit-Tomar