Globe Rendering

Bharadwaj Nidumolu and Abhiram Venigalla
{b.nidumolu001@umb.edu},{abhiram.venigalla001@umb.edu}
University of Massachusetts Boston

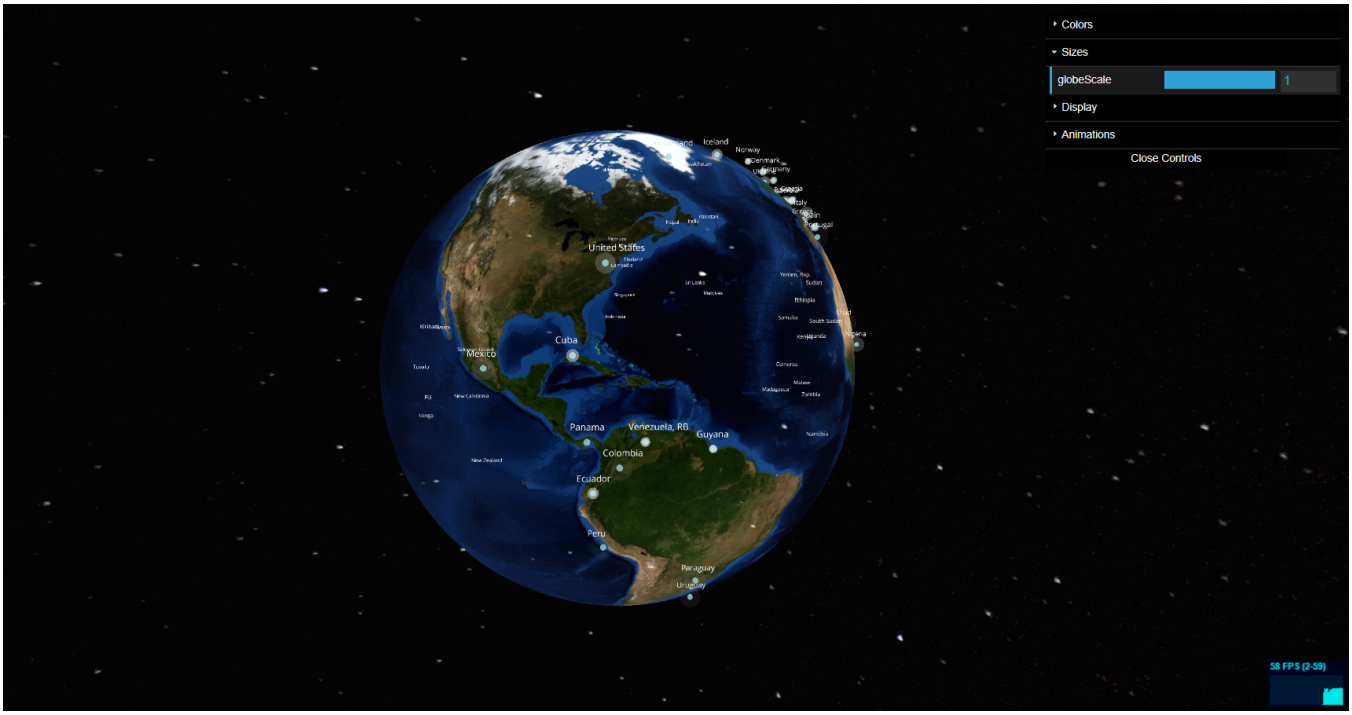

Figure 1: 3D Globe.

ABSTRACT

This project demonstrates how to texturize a geometrical object called a sphere, plot data on it like major captial cities, add animations to it, and then render it to the scene using sky-box image.

KEYWORDS

WebGL, Visualization, Three.js

ACM Reference Format:
Bharadwaj Nidumolu and Abhiram Venigalla. 2022. Globe Rendering. In *CS460: Computer Graphics at UMass Boston, Fall 2022.* Boston, MA, USA, 2 pages. https://CS460.org

1 INTRODUCTION

In our secondary school days, we used to receive 10 points on our final exam just for locating countries on a world map. As a result, we proposed this idea of Globe rendering project to make the world map digital (3D) in an interactive way.

Here we can zoom in as we all zoo out the globe, we an also remove city names by un-checking markers. We even ca move the globe right, left.

CS460, Fall 2022, Boston, MA

ACM ISBN 1337.
https://CS460.org

2 RELATED WORK

Three.js [**?**] https://threejs.org/docs/api/en/geometries/SphereGeometry.

3 METHOD

1) A skybox was created to get 3D effect of the sky.

2) Used Three.SphereGeometry and gave the texture with Loader.load() along with position, light and the scale adjustment.

3)We have loaded cities data with longitude and latitude with the help of json.

4) Finally, adding this globe to scene to render it.

3.1 Implementation

1) SkyBox

initScene = () => this.scene = new THREE.Scene(); this.scene.background = new THREE.CubeTextureLoader().load([
'../Final$project/px.png$',
'$../Finalproject/nx.png$',
'$../Finalproject/py.png$',
'$../Finalproject/ny.png$',
'$../Finalproject/pz.png$',
'$../Finalproject/nz.png$',
])

2)GUI

$functionsetup(app)$
$constcontrollers$ = [];
app.addControlGui(gui =>
const colorFolder = gui.addFolder('Colors');
controllers.push(colorFolder.addColor(config.colors, 'globeMarkerColor'))
controllers.push(colorFolder.addColor(config.colors, 'globeMarkerGlow'))

const sizeFolder = gui.addFolder('Sizes')
controllers.push(sizeFolder.add(config.scale, 'globeScale', 0.1, 1))

const displayFolder = gui.addFolder('Display');
controllers.push(displayFolder.add(config.display, 'map'))
controllers.push(displayFolder.add(config.display, 'markerLabel'))
controllers.push(displayFolder.add(config.display, 'markerPoint'))
const animationsFolder = gui.addFolder('Animations');
controllers.push(animationsFolder.add(animations, 'rotateGlobe'));
sizeFolder.open(););

3)Animations

const animations = rotateGlobe: false

we ckeck the rotate box in gui the globe will rotate

3.2 Milestones

How did you structure the development?

3.2.1 Milestone 1. Texturing the globe.

3.2.2 Milestone 2. Loading point on the globe.

3.2.3 Milestone 3. Rendering it to the scene.

3.3 Challenges

- Challenge 1: Collecting the cities data and loading them onto the globe makes us difficult.
- Challenge 2: Resolving issues of some part code which was blocked by chrome and other browser.

Table 1: Some example table

Device	Performance
iPhone	45 FPS
Macbook	60 FPS

4 RESULTS

Finally we rendered a 3D animated globe.

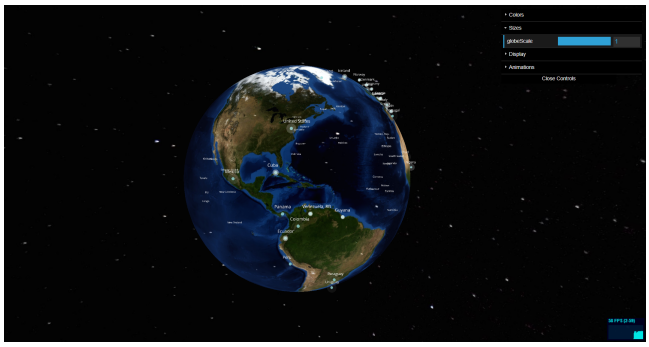

Figure 2: Final Output

5 CONCLUSIONS

From this project, we learnt various Three.js elements texturing and skybox imagining in a 3D world.

6 REFERENCES

https://globe.gl/ by Vasco Asturiano