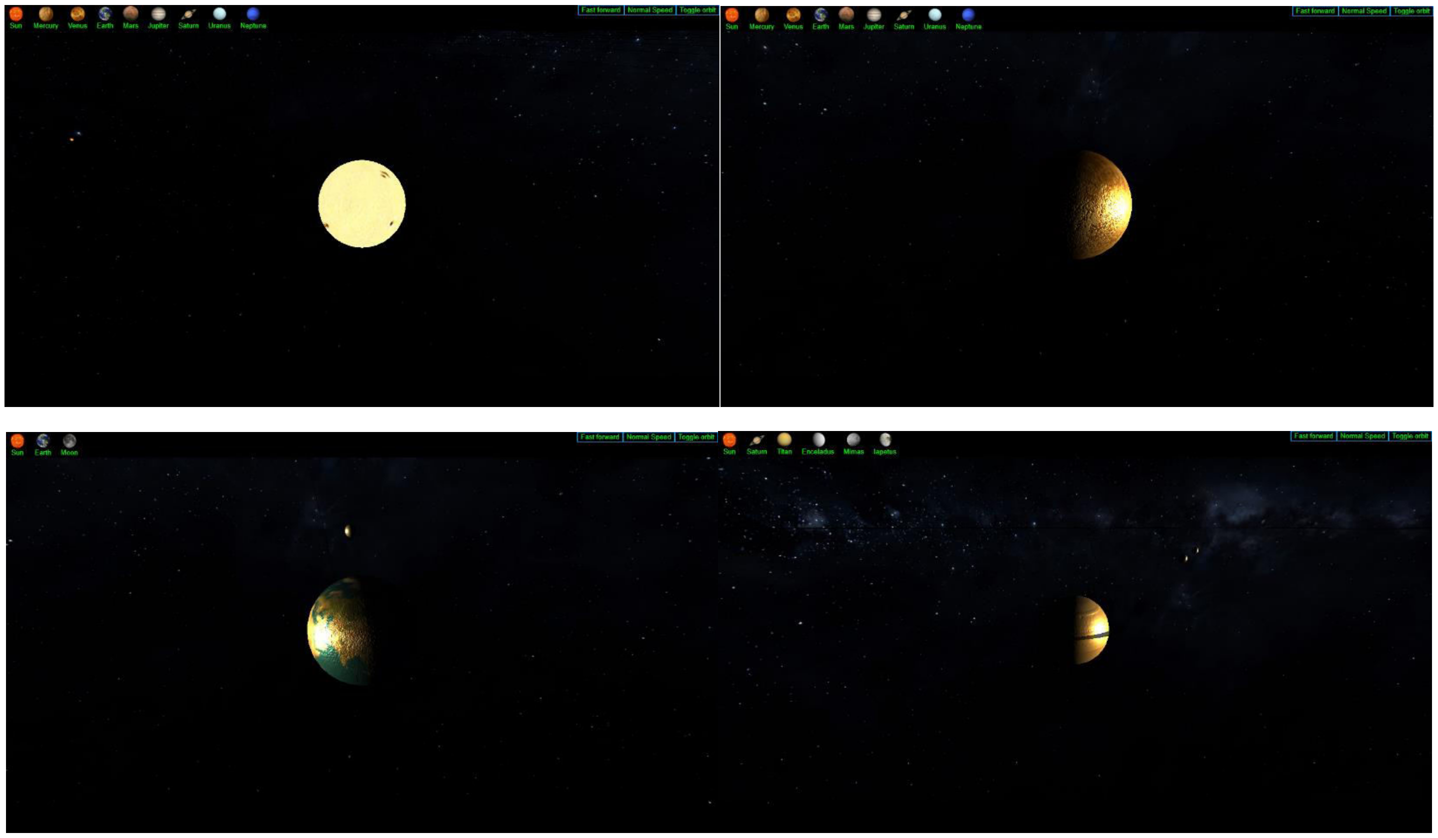
GRAPHIX MP DOCUMENTATION

SEDILLO, CLAUDE KRISTOFFER APRIL 20, 2018



I. PROJECT DESCRIPTION

My project is a simple simulator of the solar system. It involves 8 planetary models rotating on their own axis, 13 moons orbiting around their parent planet, all while revolving around a sphere modelled after the sun.

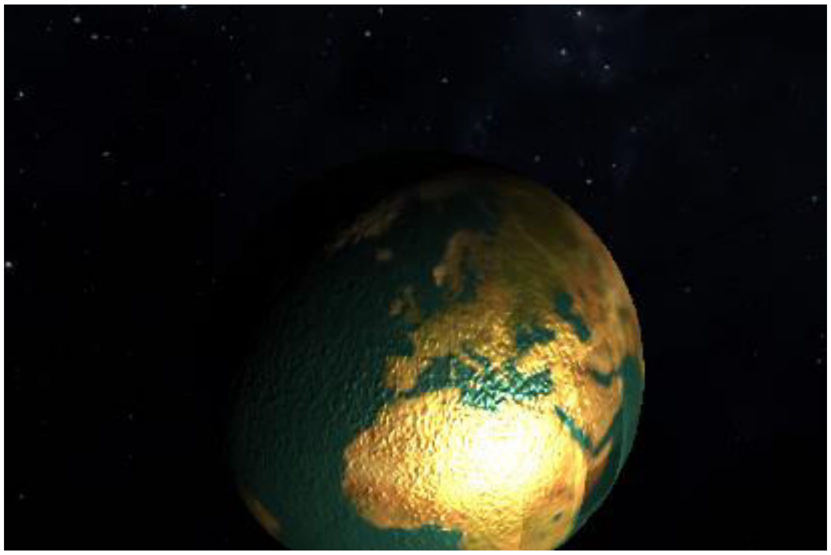
This project is inspired by my interest in astronomy. I got the idea of doing a space simulator from my interest in playing space simulation games such as Kerbal Space Program and Space engine.

II. CONCEPTS YOU APPLIED

The main concepts applied in the project mainly revolve around modeling transformation. Planets rotate on their axis and revolve around their parent star. Therefore, portraying their natural rotations with high accuracy is a must for a simulator of this nature. An example of details that must be portrayed accurately is the counter-clockwise rotation and revolution of Venus compared to the clockwise rotation of the other planets, and Uranus spinning on its side.

Another major concept in this simulator is the proper implementation of lighting. For my project, I applied a faint ambient light so that the user can still see a part of the unilluminated side of a planet, or else one side would be illuminated, and the other side would be pitch black.

The next concept I applied to my project is the implementation of shadows. For this project, I did not implement the proper axial tilts and orbit tilts of the planets and moons, and therefore when simulating the project there will come a time when a planet sits in the exact middle of the moon and the sun (or vice versa). When this happens, the object intersecting the sunlight must cast a shadow upon the object being blocked. Below is a screenshot of this phenomenon.



The last concept applied is proper camera positioning. When playing a space simulator game, the user expects to not only see the planets revolve around the sun, but to also see the planets in detail. Which is why having the ability to focus on a planet is a must.

The models used on this simulator are all made of SphereGeometry objects and made with MeshPhysicalMaterial objects. The difference is they all differ in size based on category. Below is the list of variations.

- terrestrialGeometry – Used by Mercury, Venus, Earth, and Mars. Has a radius of 1.
- moonGeometry – Used by all moons except for Phobos and Deimos. Has a radius of 0.2.
- marsMoonGeometry – Used by the moons of mars Phobos and Deimos. Has a radius of 0.05.
- sunGeometry – Used by the Sun. Has a radius of 18.
- gasGiantGeometry – Used by the gas giants Jupiter, Saturn, Uranus, and Neptune. Has a radius of 4.

III. RESOURCES THAT YOU BORROWED FROM THE PUBLIC DOMAIN

For my planet textures, there is a Three.js API available to the public called threex.planets. I simply used the planet maps and bump map image files located on the downloadable folder to map them to my spheres. The image file for my Milky Way background was downloaded from a website called solarsystemscope.com. The moons maps were downloaded from two websites: planetpixelemporium.com and stevealbers.net. All these sources are formally listed on part IV.

Here's a detailed list of my image sources:

- Sun, the 8 planets, and the moon – threex.planets
- Milky way background – solarsystemscope.com
- Phobos, Deimos (moons of Mars) – planetpixelemporium.com
- Other moons – stevealbers.net

The snippet used to create the rings of Saturn and Uranus was borrowed from the source code of Threex.planets, it is located at line 163 of the Threex.planets source code.

The snippet below is borrowed from a public source which I used to create the visualized orbit paths of my planets. This is the original code, but I modified it on my project according to my needs

```
var shape = new THREE.Shape();
shape.moveTo(orbit, 0);
shape.absarc(0, 0, orbit, 0, 2 * Math.PI, false);
var spacedPoints = shape.createSpacedPointsGeometry(128);
spacedPoints.rotateX(THREE.Math.degToRad(-90));
var orbit = new THREE.Line(spacedPoints, new THREE.LineBasicMaterial({
  color: "yellow"
 }));
 scene.add(orbit);
```

IV. REFERENCE

Albers, Steve. Planetary Maps. Retrieved from http://stevealbers.net/albers/sos/sos.html

Hastings-Trew, James (2006). JHT's Planetary Pixel Emporium. Moon textures. Retrieved from http://planetpixelemporium.com/

Kerbal Space Program. Learn more on https://www.kerbalspaceprogram.com/en/?page_id=7

INOVE. Solar System Scope. Planetary Textures. Retrieved from https://www.solarsystemscope.com/textures/

Jeromeetienne (Github username). Threex.planets. Planet textures. Retrieved from https://github.com/jeromeetienne/threex.planets

Orbit visualization snippet: code retrieved from https://jsfiddle.net/prisoner849/a2oqz9vx/ (line 54-62)

Space Engine – the universe simulator. Download on http://spaceengine.org/