Joonyong Rhee
JJR975
Solar System with a Growing Planet and Spacecraft

I have decided to create a Solar System consisting of two planets and a third planet that starts as a speck and grows in time. My fascination with outer space and exploration has inspired me to create my design. There is a orange and yellow diamond figure and a white tip with a decagon as its widest plate in the center. I have labelled this as the star to my Solar System. Circulating around the star in two orbits of different lengths are two planets which are all light blue spheres with noticeable white caps. The planet closer to the star is much smaller in size and has one moon. The planet farther from the star has a large and a small moon and is much larger in size. The moons are cubes with pyramids on two pairs of opposite faces. The moon is colored grey and white. There is also another planet that starts out as a speck and grows in time to become the medium-sized planet in a medium-sized orbit. My second object is a spacecraft that spins, turning its arms, with rotating solar panels.

There are several cool user inputs that are implemented in this design. Using the mouse and clicking on a spot will allow the user to be able to drag the entire Solar System to a location he or she would like. Simply click and hold on to the left mouse button and drag through the canvas to watch the Solar System move in the direction of the cursor. Let go to stop dragging the Solar System. If you drag the mouse beyond the canvas boundaries, the Solar System will continue to follow your cursor even if you are not holding down the left mouse button. If this is the case, click on the screen to stop moving the Solar System. Another very simple user input is the keyboard. Using the left and right arrow key, the user can speed up the entire model, including the spin of the spacecraft in a certain direction or reverse the spin. Simply hold down the arrow key of the direction you want the model to spin and watch it speed up in that direction or slow down and eventually reverse direction. All the objects in the Solar System will change speed, so the moons, planets, and spacecraft parts will all rotate faster when speeding up and all rotate slower when slowing down. By pressing the up arrow key, you can increase the spacecraft's acceleration left and by pressing the down arrow key, you can increase the spacecraft's acceleration right. There is also a pause and play button that will pause and resume the rotating animation.You can still use all keyboard and mouse functions to resume the spinning or move the objects while they are still. You can also enter any integer value into the input box and press submit to change the rotational velocity. The higher numbers spin counterclockwise faster, and the lower numbers spin clockwise faster.

The project started by originally designing the star. I wanted my star to be a full shape with a sense of power, but not a sphere. I decided that a 3-dimensional diamond would be a great shape for the star, but I did not want the widest plate to only have four sides. I wanted the star to have many edges on the widest plate sort of like a cartoonish implementation of the sun. I decided to add 10 sides on the widest plate and then bring all the tips to a central higher point

like a cone. By modifying bits of the sample code for a cylinder, I made one peak and duplicated it to make the other peak in the opposite direction. I colored each tip white and alternated with orange and yellow for all the vertices on the wide plate in order to color a nice warm star (See Figure 1).

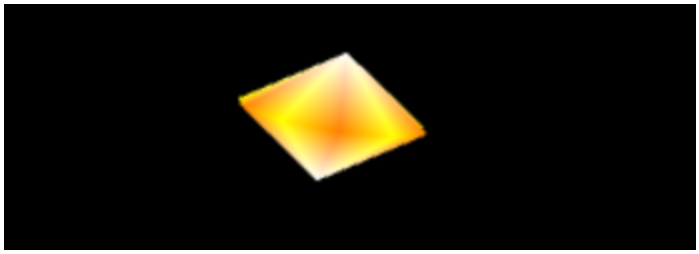

Figure 1: The star after it was fully designed. It is set on a slight rotation and centered on the canvas.

I then played around with setTranslate and setRotate to position and tilt the star in the position as shown in Figure 1. I then set a rotation speed and animation function to allow my star to rotate on its axis smoothly.

The next step for my Solar System was to add the planets. Here I studied and used the sample code for the sphere but modified it by making the ice caps larger and recoloring the sphere light blue with white ice caps. I built the first planet easily using the previous modelMatrix, however I had to push and pop the star modelMatrix from a stack for use in creating the other planets. Using the scale and translate functions, I could make these planets different sizes and make their orbits different lengths from the star (See Figure 2).

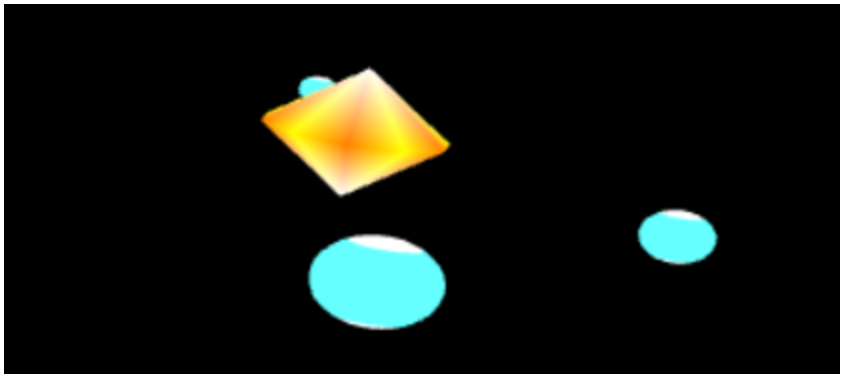

Figure 2: The three planets fully designed. They are all different sizes and have different length orbits.

I added different rotational velocities to each planet as well so they rotated around their axes at different speeds.

I then decided to design the moons. I decided to do this manually, by putting in each vertex by hand rather than using a for loop. I decided to do this because I wanted to go for a rather complicated shape. I wanted to start off with a cube and then have two pairs of planes, become outward pyramids. It would be similar to making two diamonds, however I thought that

because two of the planes were squares, it would be easier to hard code the vertices. I decided to color the pyramids all grey and the two squares white, in order to simulate watching the moon in many phases, as different amounts of grey and white were visible as the moons rotated (See Figure 3).

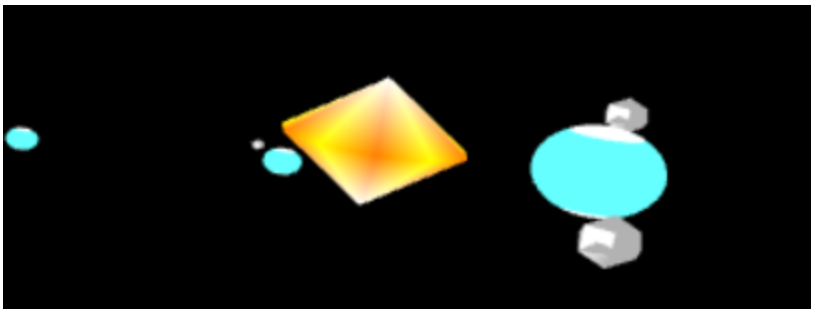

Figure 3: The moons are orbiting the planets. The big planet has two moons and the little planet has one moon.

Like the planets, I had to push and pop the modelMatrix of the large planet in order to set up the second moon on the large planet.

I decided that I wanted to add some user inputs that would allow users to interact with my model. I started by adding a simple rotation speed control and reverse by changing the ANGLE-STEP, or how large of an angle to make in a second so that it would go more and more positive as the right arrow key is pressed and would go more negative as the left arrow key is pressed. Eventually the Solar System can flip forward and backwards in its orbit if the opposite arrow key is pressed long enough. The next thing I implemented is the ability for the user to move the Solar System with the mouse in any direction the cursor is. I did this by measuring the distance and direction from a cursor's starting point and stopping point and then implementing those values into a translation of the modelMatrix for the star, which moves the entire Solar System. I also added an auto-scaling animation that allows a planet to start as a tiny dot and grow until it reaches a certain size and then maintain that size (See Figure 4).

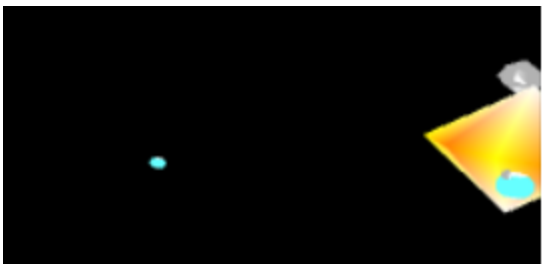

Figure 4: The model of the Solar System after being moved by the cursor to the left edge of the canvas. The little blue dot is the early stage of a growing medium-sized planet.

My second object is a spacecraft that stays in a spot unless the user uses keyboard inputs to move it. The spacecraft consists of a cylinder body, a longer cylinder arm that rotates around

the body, and two solar panel plates that rotate around the spacecraft arm as its axis. I started working on this spacecraft by modifying a sample code for a cylinder to have equal sized top and bottom circles (See Figure 5).

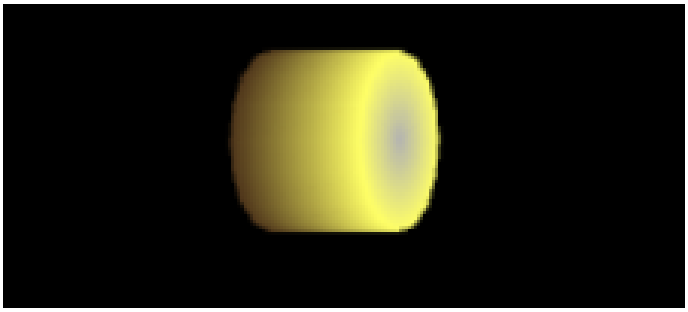

Figure 5: The starting cylinder for my spacecraft.

I then proceeded to add another cylinder but rotated and scaled differently so that two equal sides were rotating, centered at the center of the cylinder (See Figure 6).

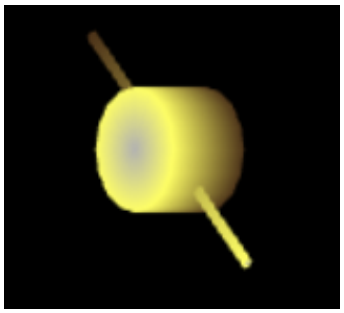

Figure 6: Another much thinner and longer cylinder rotating around the axis of the main cylinder.

The main cylinder is actually not rotating around its center height, but actually spinning around its center diameter. The cylinder rod is rotating around the center height of the main cylinder.

I added two square solar panels onto the two arms and made them rotate around the arm as its axis. I colored them blue and yellow so they look like solar panels. I also made their rotations opposite of each other and faster so they were more visual (See Figure 7).

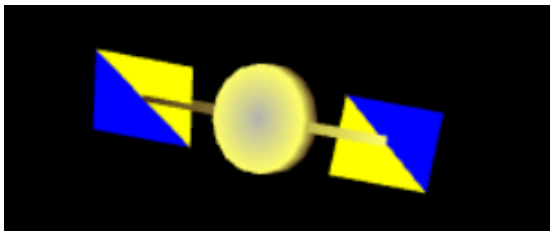

Figure 7: The solar panels attached and rotating around the cylinder rod as its axis