Final Report of Implementing a Brief Solar System with OpenGL

Professor： Mike Bailey

Students ： Hao Wang

Dec 09, 2015

Proposal

I am planning to display a solar system for my final project. This project will consist of:

- A stationary Sun is in the center of the world coordinate. A Mercury, a Venus, an Earth, a Mars, a Jupiter, a Saturn, a Uranus, and a Neptune are moving around the Sun as a circle track.
- Each planet has rotation and revolution.
- The Sun is lighting. The planets are textured.
- The Saturn has its ring. The Earth has the Moon.
- The size of the system and each aster will not be perfectly scaled down, but the rotation and revolution time of each planet will. This work will be done on research.
- The rotation and revolution of the planets and the lighting of the Sun are controlled by keyboard or right-mouse menu.
- Both orthographic and perspective views will be implemented.
- Drag the left-mouse button to rotate the eye sight view. Drag the middle-mouse button to zoom in or out.

Strategy

I did not use display list.

In the beginning, I decide to fix my proposal about the Sun, because the Sun also need to be textured. I declare every texture pointer I need to use. There are 1 star and 8 planet, which are quite a lot. Secondly, I declare some control variable “TrackOn” “RevolutionOn” “RotationOn” and “Light0On” so that I can turn on and off the track, revolution, rotation and lighting by the keyboard button “1” “2” “3” and “4”. After that I declare 2 animation variable “Revolutionunit” and “Rotationunit” in order to display the rotation and revolution of the asters.

In the Animation() function, I write the following code:

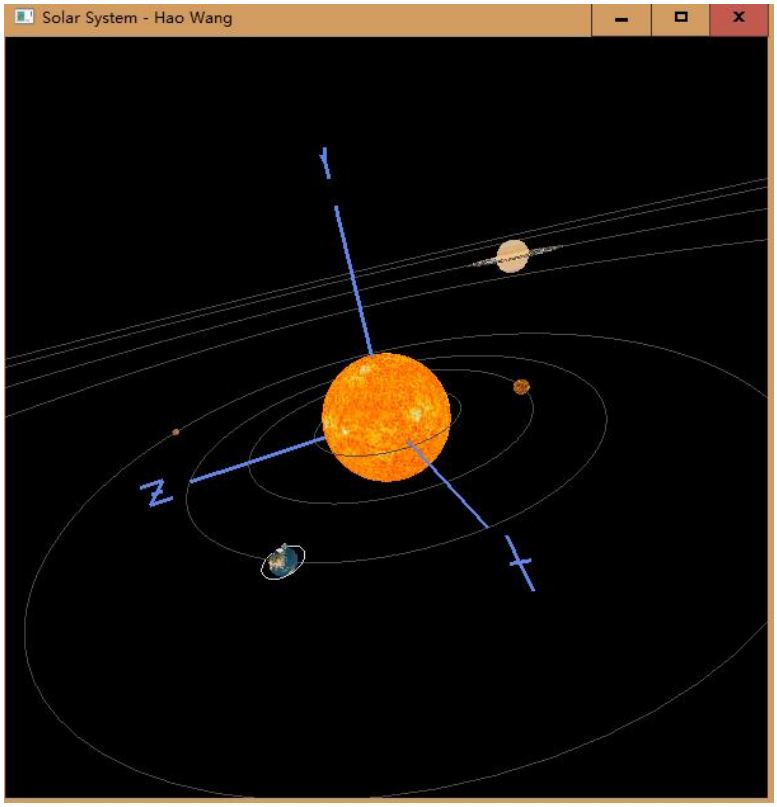
```
if (RevolutionOn) {
    Revolutionunit += 5.;
}

if (RotationOn) {
    Rotationunit += .5;
    }
```

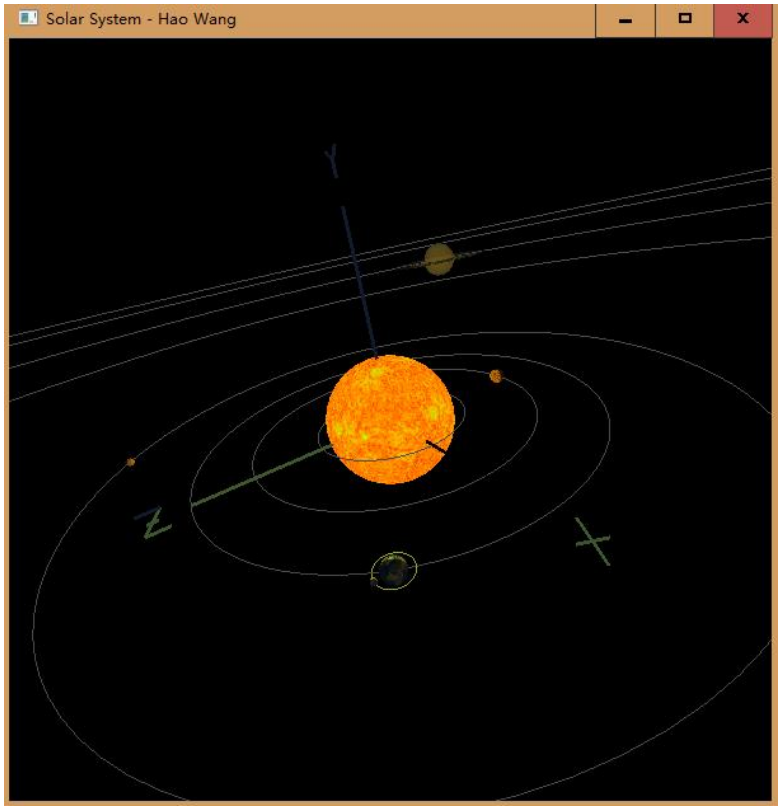
This means if the control variable is on, the corresponding display variable changes so that an animation could be displayed.

I draw the sun as a big static point light but still textured its surface and implement its rotation. Then draw the circle track of each planet if the track control variable is on. And then draw the 8 planet. I textured every planet and set their surface to "GL_MODULATE" if the light control variable is on. Most importantly, I implement their revolution and rotation and set a perfect scale to them. Every planet's revolution is in a scale, even though the Uranus and the Neptune have a slow speed. Every aster's rotation is in a scale, even though the Venus rotates too slowly. The distances from planets to the Sun is in a scale, even though the Neptune gets far away. The radius of every planet is also in a scale, even though their size are huge different. The reason why I can scale everything is that for each kind of aster motion I only have one animation variable. These variable can be treated as a unit of the scaling. Thus all I need to do is doing research to find out the real time of each aster motion, and multiply one of these units. All of my research data are from NASA website. [1]

Result



Normal view



With lighting

The result meets all requirement of my proposal. In addition, it also satisfies

- The Sun is both lighting and textured.
- The track of each planet is shown.
- Keyboard buttons "1" "2" "3" and "4" control the display of tracks, revolutions, rotations and the Sun's lighting.
- The sun also has rotation. The moon also has its track, revolution and rotation.
- When turn off the lighting of the sun, the normal "GL_REPLACE" surface of each planet will be displayed.

Conclusion

After the project, my ability of doing works with OpenGL is enhanced. In this project, I used almost everything I learn from class, including hierarchical transformations, texturing animation, lighting and keyboard controlling. This time I do not use these knowledge separately like our normal project, but put all of them together. There are some new problems I have never met before. For example, how can I implement different speed of texture movement? My solution is modifying the MjbSphere() function the professor gave us for project 3. I add the animation control variable as a parameter of MjbSphere() function. Thus I can use different value when I call this function in the main program. In other words, this project also enhances my integration ability. This is the most I satisfy with.

Reference

[1]. NASA website. Retrieved from: https://www.nasa.gov/