Hafed A. Alghamdi

Abstract

Space is a Space, and the development of any Solar System requires tremendous Computer Graphics programming and solar scientific calculations. Moreover, solar systems most of the time concentrate on the actual distances and objects sizes. However, in this project we have implemented a Solar system that combine both proportional distances and sizes, estimated distances and sizes with multiple camera positions associated with each planet in our solar system.

Introduction:

In this project initially the idea was implementing a solar system that have real textures, distances and sizes in addition to the camera position at the Top of each planet. It was a great idea to have another version that provide a good looking with manageable sizes and appearance and shows at least which planet is bigger than the other and which planet is nearer to the sun from the one orbiting far from the sun. Therefore, both versions have been implemented in this project with a very simple User Interface to toggle between the versions and choosing a planet to view the camera position. Moreover, another idea shows that multiple views could provide more information about the planet and how they look like from the near and far distances.

The development of a Solar System can be easy if we can just estimate the sizes and just implementing it. However, real distances and sizes are required most of the time for advanced educational needs. Therefore, both version were implemented to give the normal user an intuition up to estimated scale about the sizes of our solar planets and the distances between them. This project could help kids when using the estimated version of this program which is not with real distances neither sizes. However, the other proportional version contains up to scale distances and sizes and it could help in astronomical education purposes. The main source for all actual distances and sizes is NASA website as they have done a great job in astronomic calculations and comparisons. Moreover, most of the textures are real and have been collected from NASA's website. However, as in this project the aim is to seek for realisms, we tried as much as possible to use real images for the entire solar system.

Finally, most of the ideas in this project was based on the knowledge gained in Computer Graphics CS550 and Scientific Visualization CS553 Classes.

Implementation

In this section, we are not going to discuss the implementation of easy things. However, the ideas that worth mentioning only are presented in this report. So, here what we have implemented in this project so far:

1- An estimated Solar System that gives great appearance and fit into the screen
2- Proportional Solar System that simulate the exact distances and sizes
3- Multiple camera positions for each planet
4- Turn On/OFF Animation
5- Circles to show the orbits of each planet and its most popular moons

6- Turn On/Off Lines
7- Saturn Ring as an image
8- Apply multiple Transparency options
9- Simple User Interface that provides a great control over the scene where the end user can Scale the view up, down, left and right in addition to the zooming in/out.

- **Estimated Solar System**

This version of our implementation can show great view style as it considers the appearance of the solar system not solar real distances and sizes. However all the timing, sizes and distances was in sync which means any changes applies to one portion the other portions will be affected as well. This kind of work shows an attractive view as in Figure 1: Estimated Solar System.

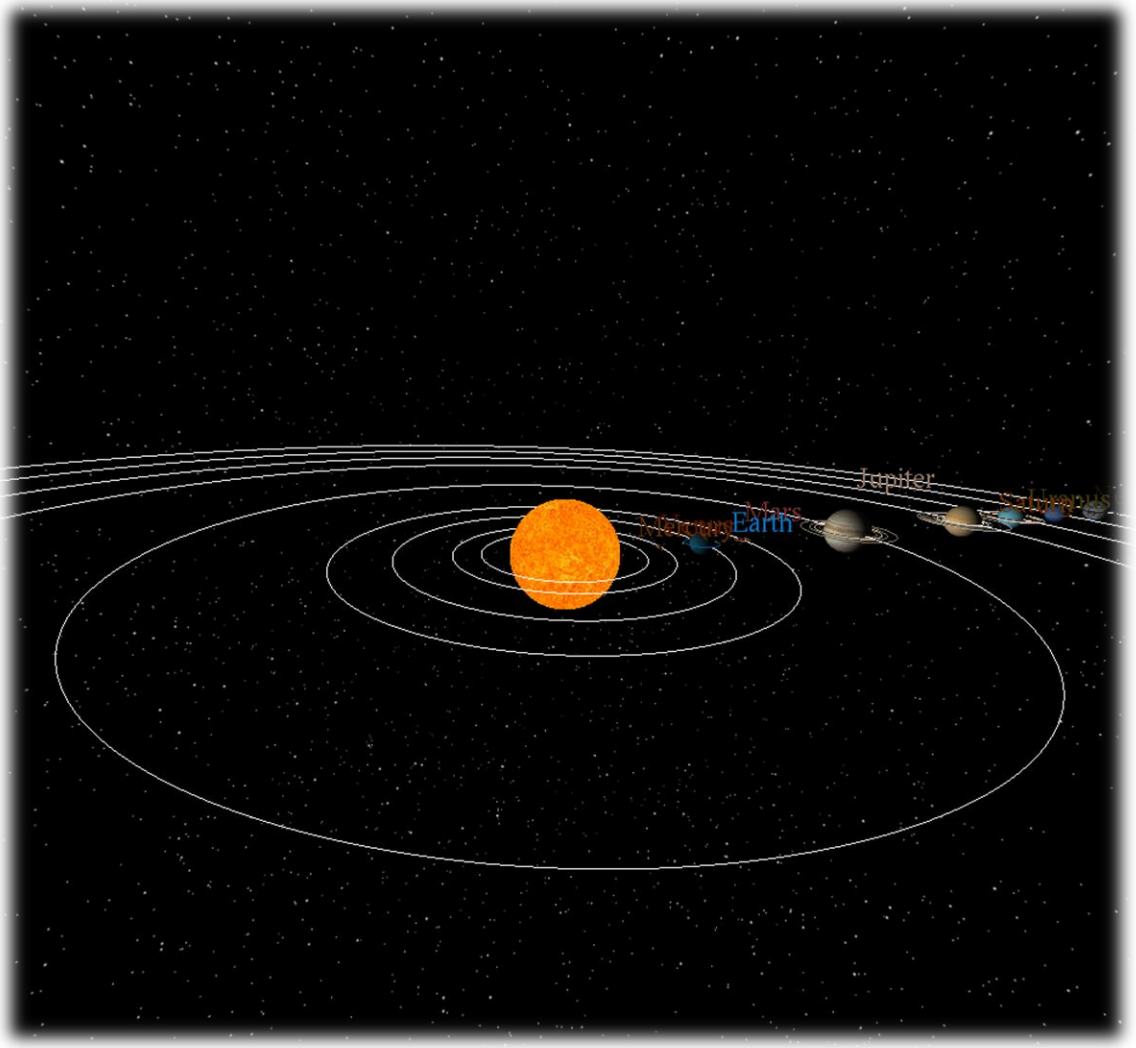


Figure 1: Estimated Solar System

- **Proportional Solar System**

This version of the project is using actual proportional data from NASA with in addition to extensive calculations to simulate our solar system as shown in Table 1: Data collected and calculated to simulate the Solar System .

Planet / moon	Distance from origin (AU)	Equatorial Radios	Orbit period days	Rotation days
Sun	0	109	None	
Mercury	0.4	0.383	88	58.6
Venus	0.7	0.949	225	-243
Earth	1	1	365	1
The Moon	0.00257	0.374	27.29	
Mars	1.5	0.53	687	1.02
Phobos	0.000062	0.017	27.3	
Deimos	0.00015	0.00097	1.26	
Jupiter	5.2	11.209	4331	0.413
Io	0.00282	0.286	1.77	
Europa	0.00449	0.245	3.55	
Ganymede	0.00716	0.413	7.15	
Callisto	0.0126	0.377	16.69	
Saturn	9.54	9.449	10759	0.444
Dione	0.0025	0.0881	2.737	
Enceladus	0.0016	0.0395	1.37	
Iapetus	0.0238	0.1154	79.33	
Mimas	0.0012	0.03	0.942	
Rhea	0.112	0.0035	4.518	
Titan	0.0081	0.40	15.95	
Tethys	0.02	0.0836	1.888	
Uranus	19.2	4.007	30799	-0.718
Titania	0.003	0.123	8.7	
Oberon	0.0039	0.1195	13.46	
Umbriel	0.001	0.1088	4.144	
Neptune	30	3.883	60190	
Triton	0.0024	0.212	5.877	
Pluto	39.4	0.186	87932	-6.387
Charon	0.00017	0.094	6.387	

Table 1: Data collected and calculated to simulate the Solar System proportionally

The key idea here is how to keep things up to scale and simulate the real solar system. As we mentioned earlier that space is a space, which means it has no limits and you can implement the real distances and sizes but may not fit in the screen or they might be so small that no one can see them on the screen. Therefore, a good idea is to make animation timing syncs aside from the distances and sizes. Also, make the distances syncs with other planets altogether away from the timing and sizes and so on. By having a scaler on each part of these three factors, we were able to sync and scale with easily and get great results. Figure 2: Proportional solar System Version

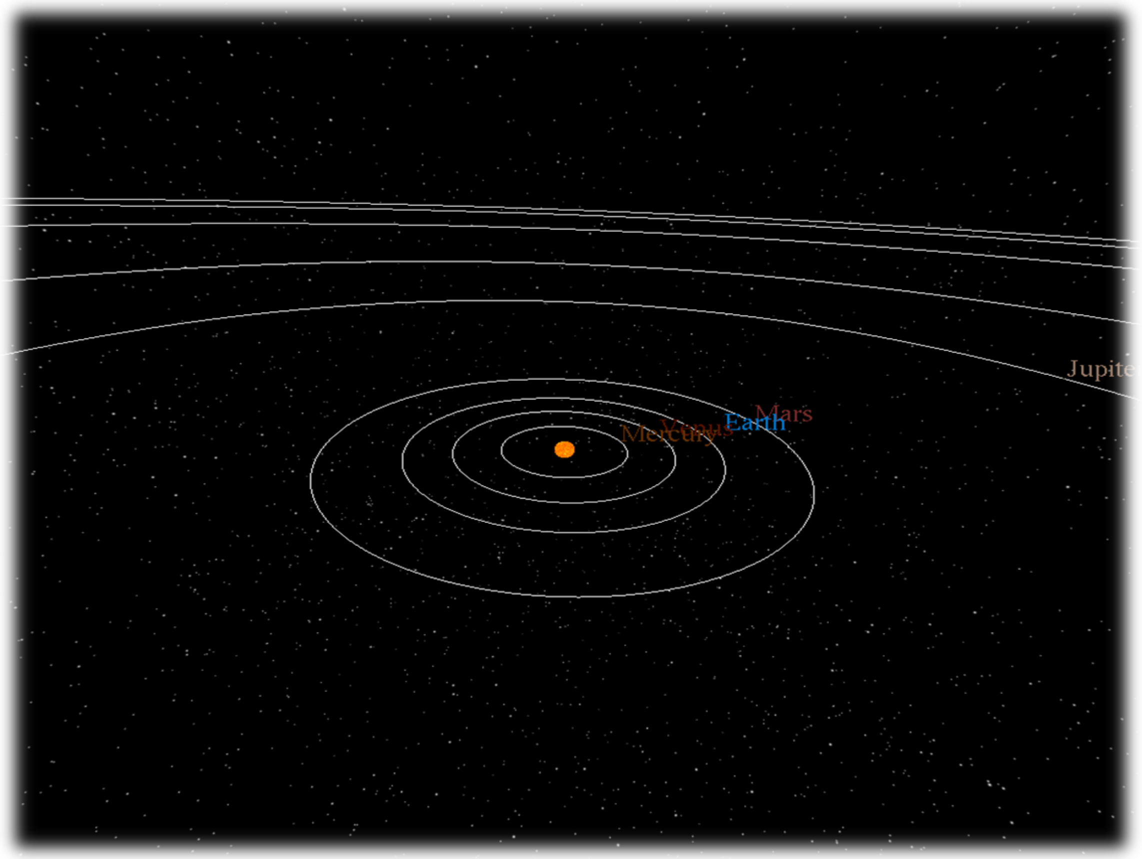
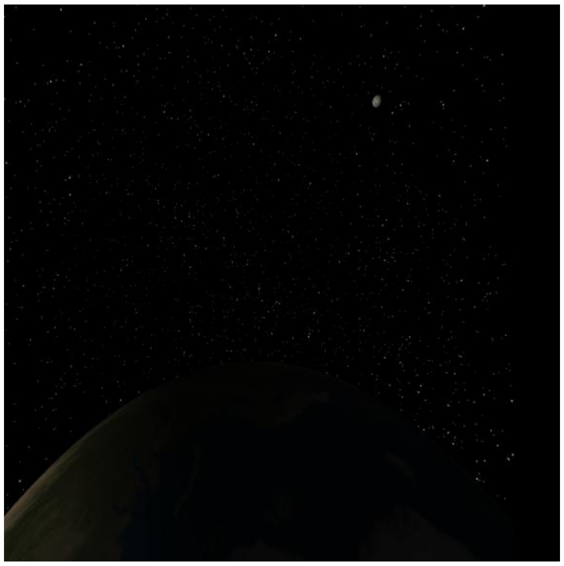


Figure 2: Proportional solar System Version

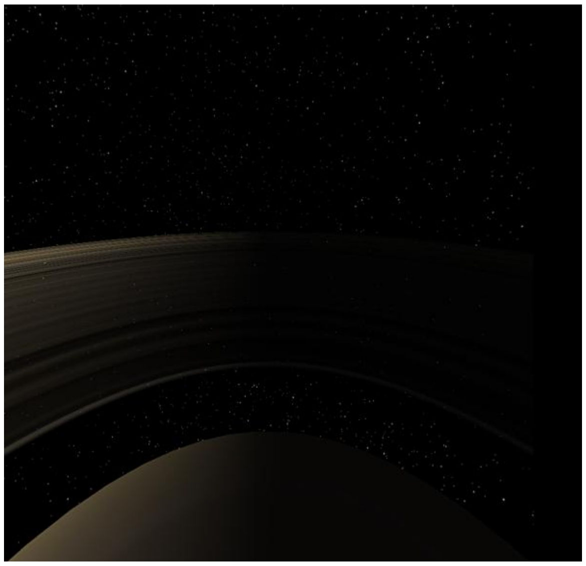
- **Multiple camera positions**

Applying multiple camera positions was not easy in the solar system as the camera position needs to follow a planet or be at the top of a planet and so on. Therefore, an idea was suggested by Professor Bailey and we were able to implement it. It is basically implemented by calculating the planet position using the cosine and sine functions based on the planet current location and putting it into a vector[3], then calculate the camera position using the sine and cosine functions based on the planet current location and put it into another vector[3]. Having these two vectors allows us to use the gluLookAt function to put the camera position at the top of each planet, near the planet or far away from the planet. Figure 3: Solution to make the camera position follows the planet while orbiting:

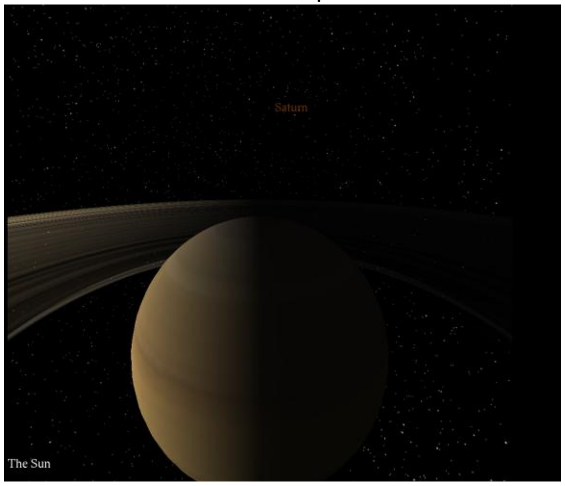
```
//Caculate the planet  postion
GLfloat UranusPos[3] = {Uranus_distance * DistanceScaler * cos(-uranus * M_PI / 180), 0, Uranus_distance * DistanceScaler * sin(-uranus
                        * M_PI / 180)};
//Caculate the Camera Position
GLfloat cameraPos[3] = {Uranus_distance * DistanceScaler * cos(-uranus * M_PI / 180), (8 * SizeScaler), Uranus_distance * DistanceScaler * sin(-uranus
                        * M_PI / 180)};
//Setup the camear on the top of the planet  pointing
gluLookAt(cameraPos[0], cameraPos[1], cameraPos[2], UranusPos[0], UranusPos[1], UranusPos[2] - (11 * SizeScaler), 0, 0, -1);
```
Figure 3: Solution to make the camera position follows the planet while orbiting



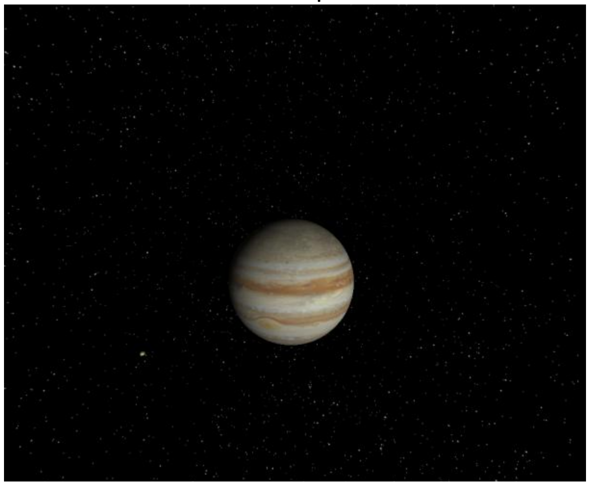
Camera Position At the Top of the Earth



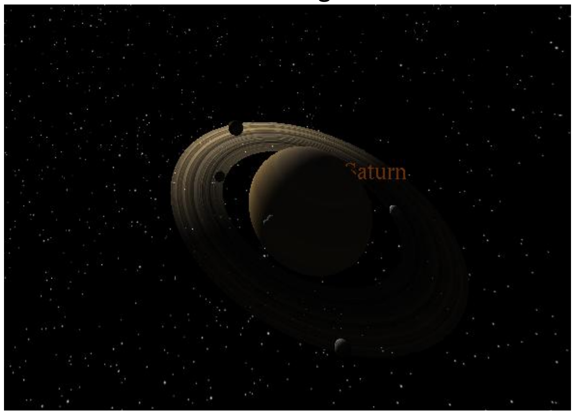
Camera Position At the Top of Saturn



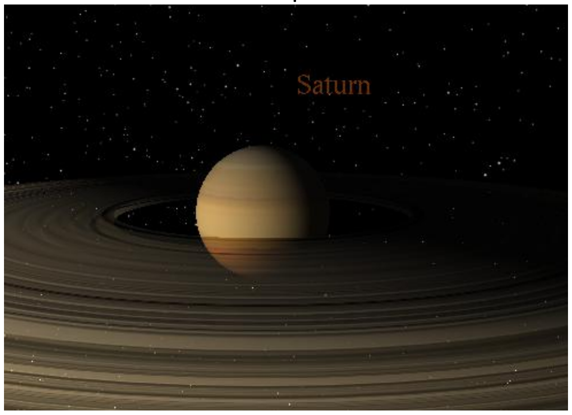
Camera Position followoing Saturn



Camera Position near Jupiture



See through Saturn Rings the Stars



See through Saturn a portion of Saturn Planet

Figure 4

- **Apply multiple Transparency options**

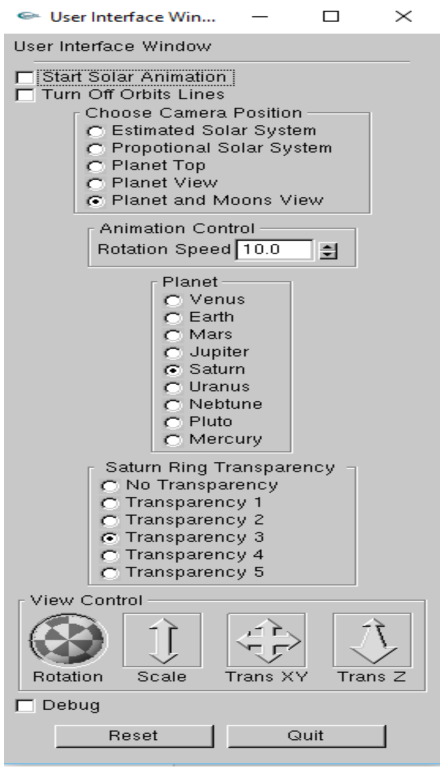
We have implemented multiple transparency options to see through Saturn Rings, the stars and the hidden part of Saturn by the Rings. This has been done using blending function:

$$glBindTexture(GL\_TEXTURE\_2D, Saturn\_Ring\_Tex)$$

This function allows us to see the stars through Saturn Rings Texture and the covered part of the planet by the Rings. Figure 4: shows some of the Transparency options used in this project.

- **Simple User Interface**

In this project we used OpenGl User Interface library to develop a simple and effective user interface as shown in Figure 5. The UI will allow the user to have control over the scene by selecting the Camera position from the radio buttons and choose the Planet accordingly or vice versa. This will allow the user to move from one position to another or changing from one planet to another easily and quickly. Moreover, the user is able to control the speed of the solar system animation in addition the Turn On/Of checkbox to stop or start the animation of the solar system. Finally, the user is able to enable and disable the planets and moons orbits lines in addition to the view control section.



User Interface Win...

User Interface Window

☐ Start Solar Animation
☐ Turn Off Orbits Lines

Choose Camera Position
- Estimated Solar System
- Propotional Solar System
- Planet Top
- Planet View
- [✓ Planet and Moons View]

Animation Control
Rotation Speed 10.0

Planet
- Venus
- Earth
- Mars
- Jupiter
- [✓ Saturn]
- Uranus
- Nebtune
- Pluto
- Mercury

Saturn Ring Transparency
- No Transparency
- Transparency 1
- Transparency 2
- [✓ Transparency 3]
- Transparency 4
- Transparency 5

View Control

☐ Debug

Figure 5

Conclusion

In this project we were able to implement the solar system with multiple camera positions, estimated Solar System that can be fit in the screen and see through Saturn Rings using OpenGl Transparency function.

References

1- https://solarsystem.nasa.gov/planets/compare?Object1=earth&Object2=uranus&System=Metric&refresh=Refresh
2- https://en.wikipedia.org/
3- Textures From www.NASA.org
4-
5- www.StackFlow.com
6- Computer Graphics CS 550
7- Computer Graphics CS 553
8- www.opengl.org