Harshavardhan Reddy | IMT 2012 045
Mr. Srikanth TK | Computer Graphics
29 April 2016

Scene Graphs and Animation

Game: *An Inter-Planetary War Simulator*

This assignment deals with the implementation of Scene-Graphs and Animation. It requires having multiple interactive objects that move relative to one another. This report deals with the experiences involving the making of the game and the usage of the elements in the making.

The Game

The game, as the name suggests involves planets trying to shoot down other planets down with an *Energy Blaze*. The Energy Blaze is a fictional technology that generates an energy capturing field which gains energy by passing through a star. The effects on the star are immediate, but temporary. The Blaze is deadly not because of the damage it creates upon impact, but because of the changes that it brings about in the victim planet. The Blaze on impact, engulfs the planet and converts most of its linear momentum into angular momentum about the planets axis. This will eventually lead to the planet's destruction by spiralling it into its parent star.

One major effect that will be noted when playing is the *Coriolis Effect*. The pseudo force (*Coriolis Force*) makes it look like the Blaze is following an arc of a path rather than a linear path as expected. So, when shooting a planet, the aiming does take some practice and trials before getting a hang of it.

The Development

Scene Graphs

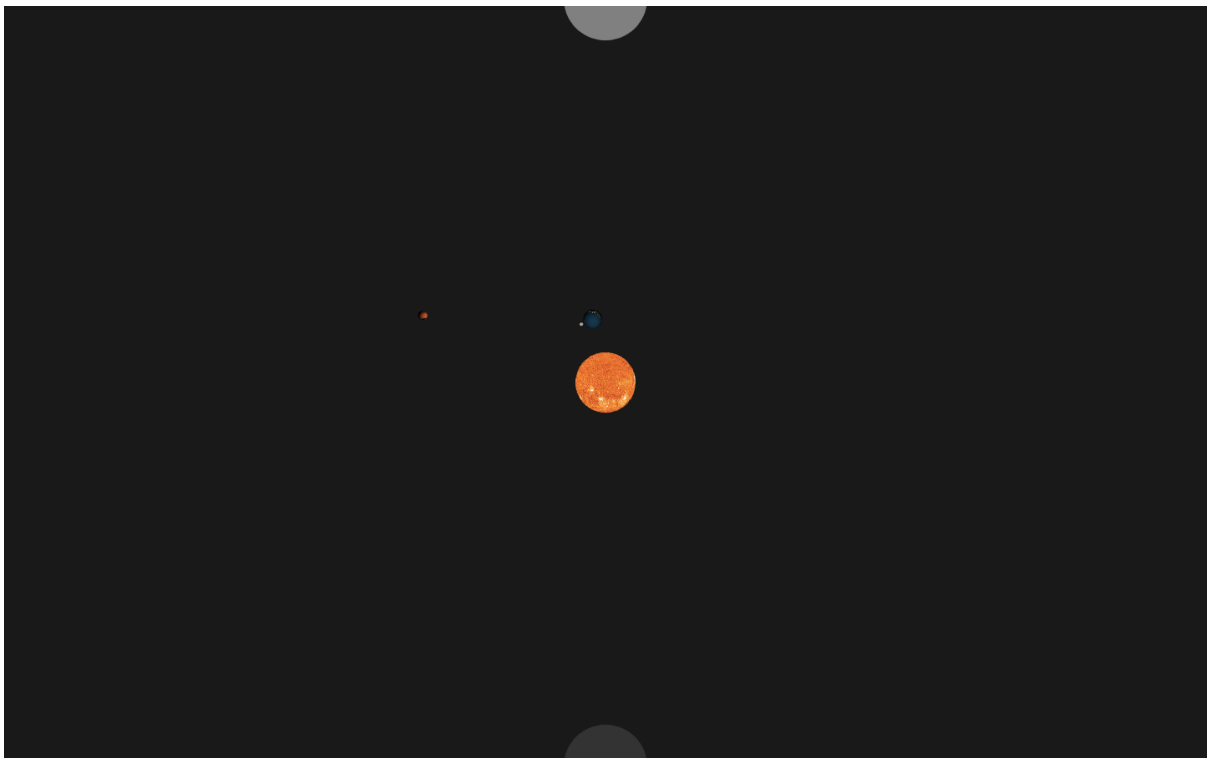
One of the first tasks was implementing a minimalistic the Scene Graph. Two classes, Scene and SceneNode were made. The implementations are pretty typical with methods like Scene::addNode, Scene::transfer, etc. Every node can store a call back function which gets called when drawing it. This function can be created outside and associated accordingly. Conveniently, the function can be changed when desired. The scene contains two default SceneNodes called “root” and “null”. Every independent SceneNode adds as a child to this node. All the nodes that are not yet to be drawn are added to “null”. When required to draw, the transfer method is used to transfer it from the current parent “null” to anything pleased. The Scene class has one draw function that calls recursively so that we’ll have a DF traversal over the SceneGraph.

Scene Making

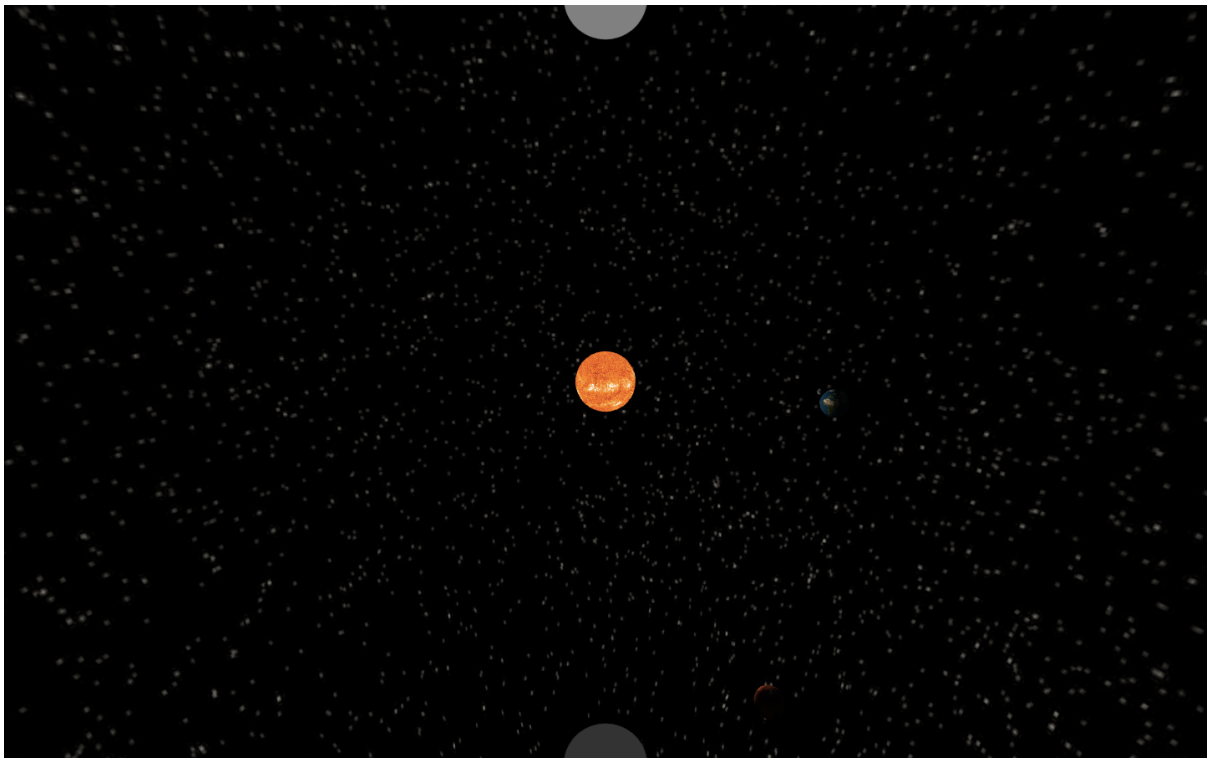
The following objects were acquired/created:

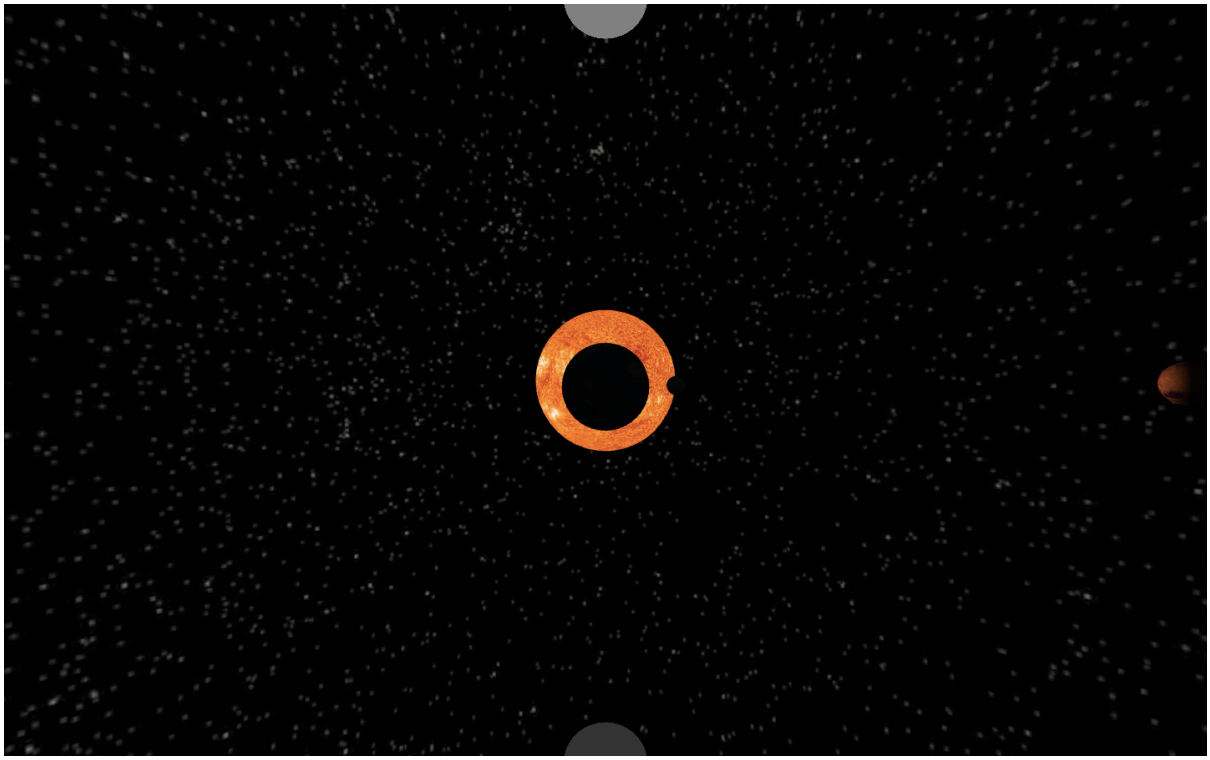
- Sun
- Earth
- Moon
- Mars
- Earth Blaze
- Mars Blaze
- Visible Universe

The sun object was put at the centre and earth as its child with a translate transform and a rotate transform to make it rotate about its axis. The moon was added to the earth as a child with similar transformations. The mars was added to sun as a child with similar transformations again. The scene would then look like this:



This looks too bland possibly because of the missing stars in the background. So a huge sphere engulfing the entire scene was added with the starry texture making the scene look a little more believable. The entire solar system revolves about the galaxy as well. The scene is fixed to the Sun's frame though. So, to simulate this the background stars are also given a rotation parameter. The scene then looks like this:

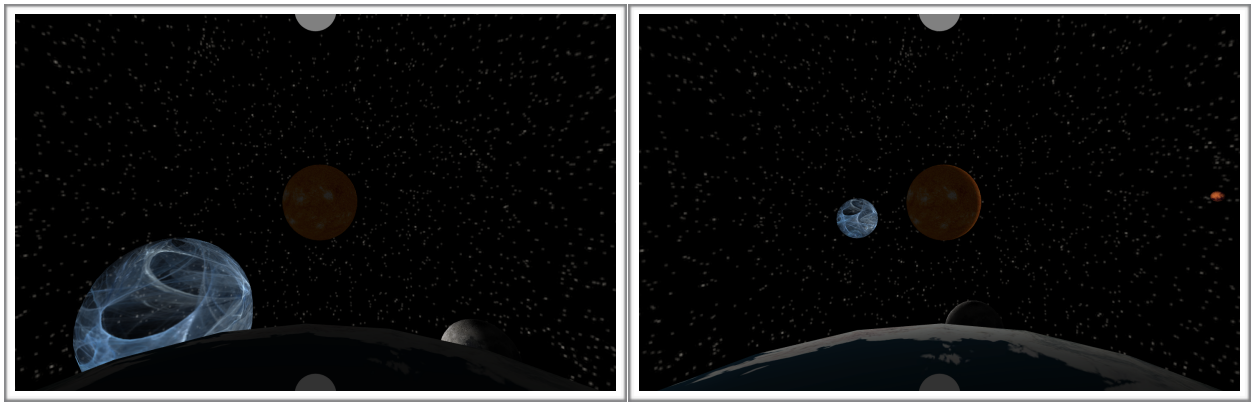




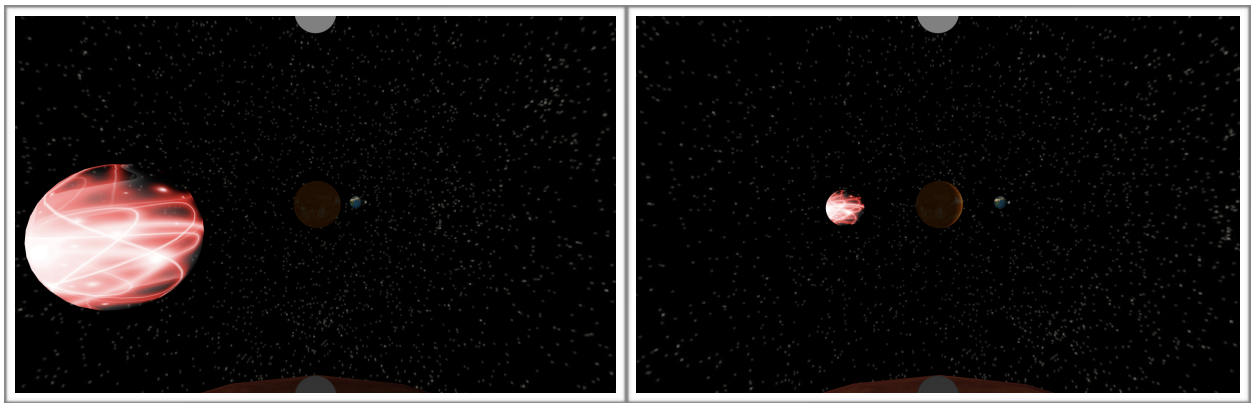
What we see here is a solar eclipse from our POV.

Blazes

The blazes are added by default to “null” of the scene. This implies that they are not drawn. When the Blaze is launched, it is transferred to “root” and made to travel from the point of origin (location of the planet at the time of launching the Blaze). It is made to travel through the sun and beyond. Below is earth launching a Blaze attack as viewed from Earth:



And this is mars launching the Blaze attack as viewed from Mars:



Collision Detection

The next step of implementation was the collision detection. The object coordinates were calculated and the distance was measured. If this is lower than the threshold, the collision was called.

Sounds

This was the final step, where the user would have a better feedback of what's happening in the game. There's a constant low frequency background hum which gives a tense impression. This audio track is overlaid with another low frequency low-key music which adds to the effect. The launching of the blazes have their own sounds, different sounds for different planets. On the blaze impacting planet, an electric zip is played. When the planet falls into the Sun, an explosion is played. All of this is accompanied by the computer advising at different stages, like on impact (for instance, speaking out "mars is going down") and on collision, etc.

Demo

A demo of the game at various stages is uploaded to Youtube.

Playlist Link