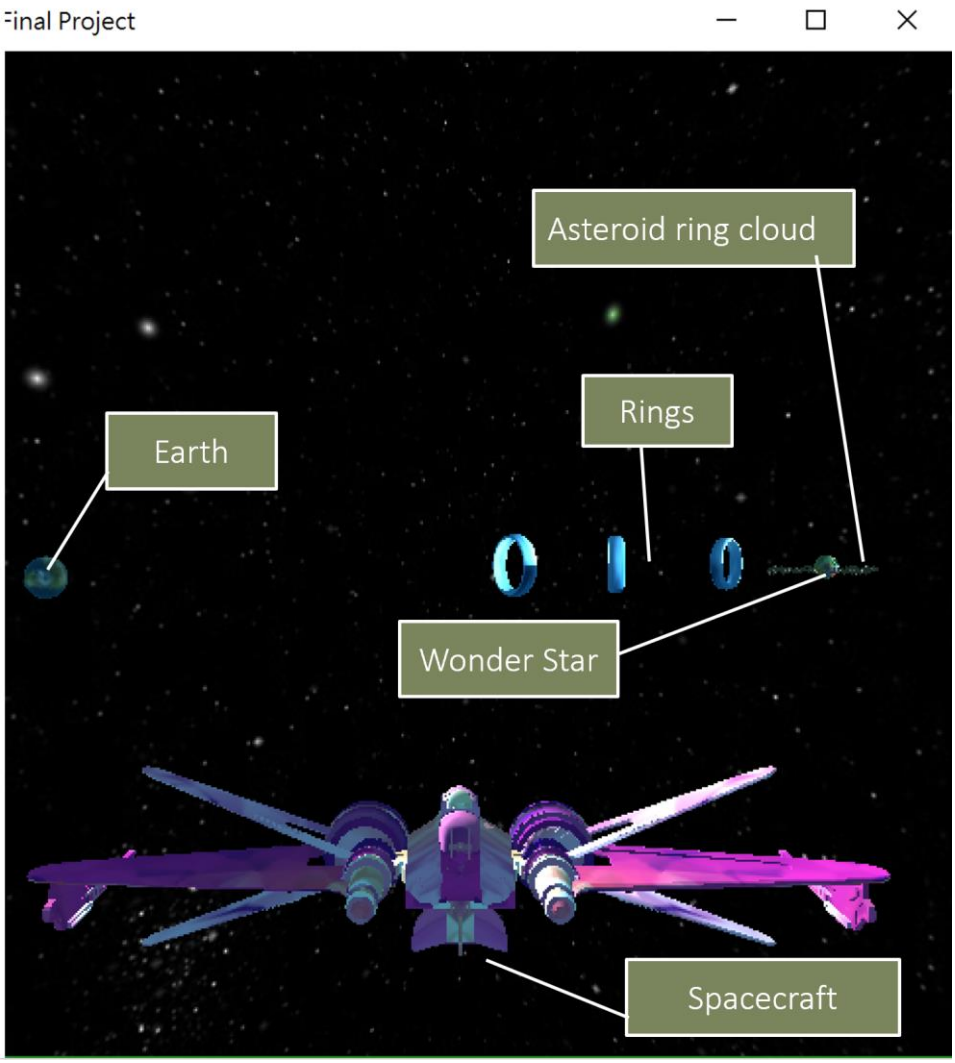
Project report

1155092634 Cheung Kam Ho
1155083016 Lam Ming Yuen

Overall:



This photo shows the scene of the project

For implement Skybox, we construct a function void installSkyBoxShaders() to use two more shader “VertexShaderCodeSkyBox.glsl” and “FragmentShaderCodeSkyBox.glsl”. It helps us handle two different work:

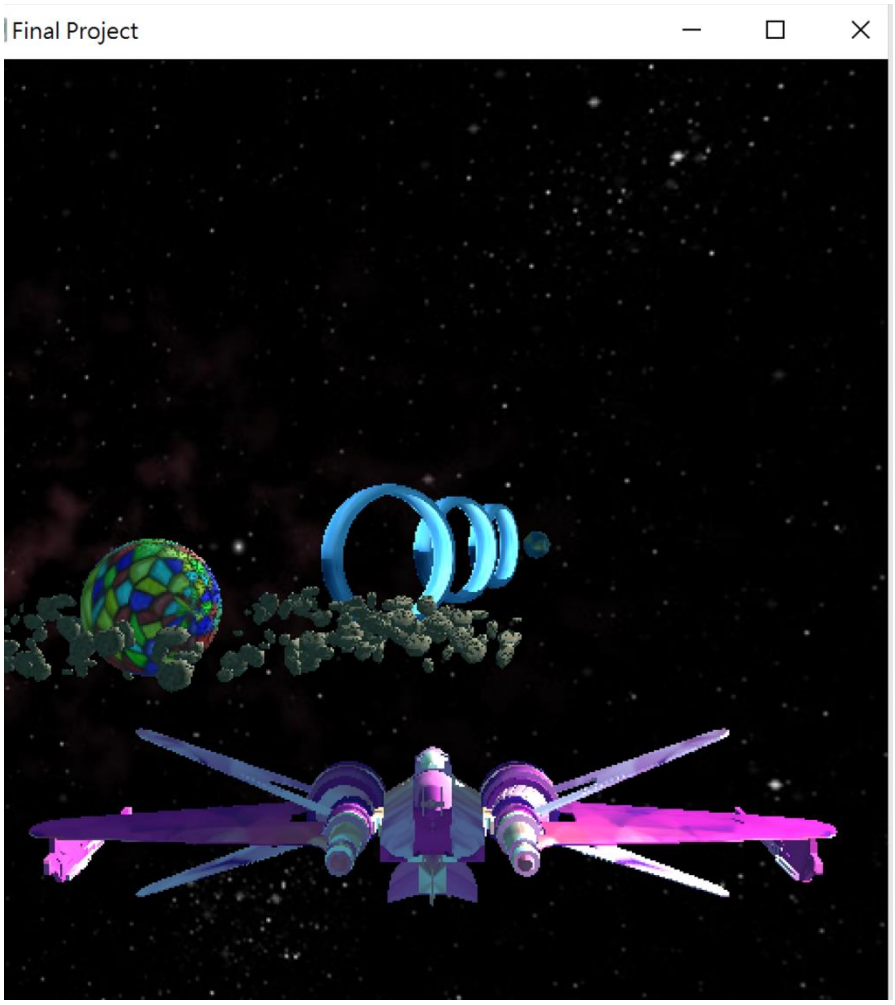
skybox and object.

```
void installSkyBoxShaders()
{
    GLuint vertexShaderID = glCreateShader(GL_VERTEX_SHADER);
    GLuint fragmentShaderID = glCreateShader(GL_FRAGMENT_SHADER);

    const GLchar* adapter[1];
    string temp = readShaderCode("VertexShaderCodeSkybox.glsl");
    adapter[0] = temp.c_str();
    glShaderSource(vertexShaderID, 1, adapter, 0);
    temp = readShaderCode("FragmentShaderCodeSkybox.glsl");
```

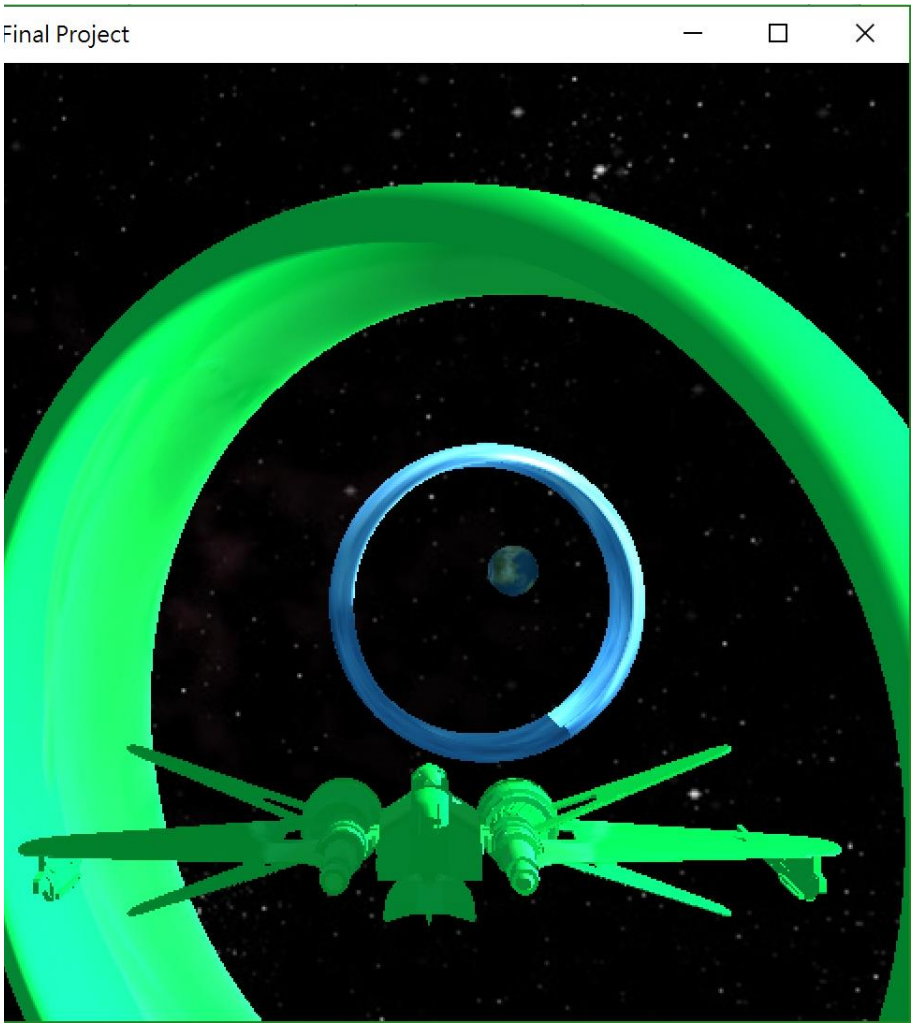
Basic light rendering results:

We build white ambient light and diffuse light basically.



This photo shows the light of the project which is from right hand-side.

Ring Feedback:



The ring and spacecraft will give a green color feedback by changing the a green texture.

To implement it, we use distance to detect whether the spacecraft is crossing though the ring. For Ring 1 to Ring 3, we build a Boolean “flag”:

A part of pseudocode:

If distance between spacecraft and ring < 5.0f:

Both object change texture to green;

End-if;

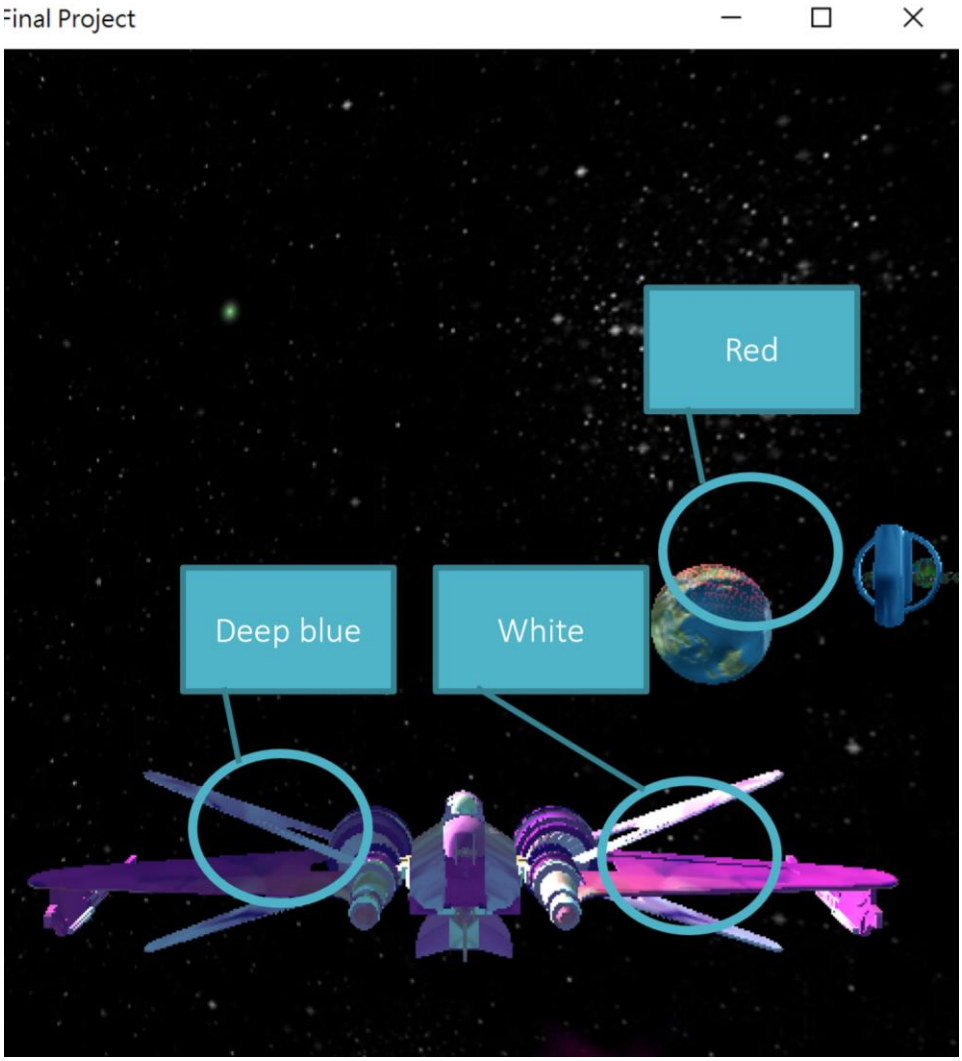
```
if (glm::distance(glm::vec3(35.0f, 1.0f, 0.0f), glm::vec3(SC_x + Xmove * glm::radians(360.0f)*cos(Ymove*0.003f), 0.0f, SC_z - Xmove * glm::radians(360.0f)*sin(Ymove*0.003f))) < 4.0f) {
    passing_ring_3 = true;
}
if (glm::distance(glm::vec3(50.0f, 1.0f, 0.0f), glm::vec3(SC_x + Xmove * glm::radians(360.0f)*cos(Ymove*0.003f), 0.0f, SC_z - Xmove * glm::radians(360.0f)*sin(Ymove*0.003f))) < 4.0f) {
    passing_ring_2 = true;
}
if (glm::distance(glm::vec3(65.0f, 1.0f, 0.0f), glm::vec3(SC_x + Xmove * glm::radians(360.0f)*cos(Ymove*0.003f), 0.0f, SC_z - Xmove * glm::radians(360.0f)*sin(Ymove*0.003f))) < 4.0f) {
    passing_ring_1 = true;
}
glEnable(GL_CULL_FACE);
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
//TODO:
//Set lighting information, such as position and color of lighting source
//Set transformation matrix
//Bind different textures

//glEnable(GL_TEXTURE_CUBE_MAP);
glEnable(GL_TEXTURE_2D);
glDepthMask(GL_FALSE);

GLuint modelSkyBoxUniformLocation;
GLuint modelProjectionMatrixUniformLocation;
GLuint modelStatusMatrixUniformLocation;
GLuint modelSKBXUniformLocation;
glm::mat4 View;
```

Bouns:

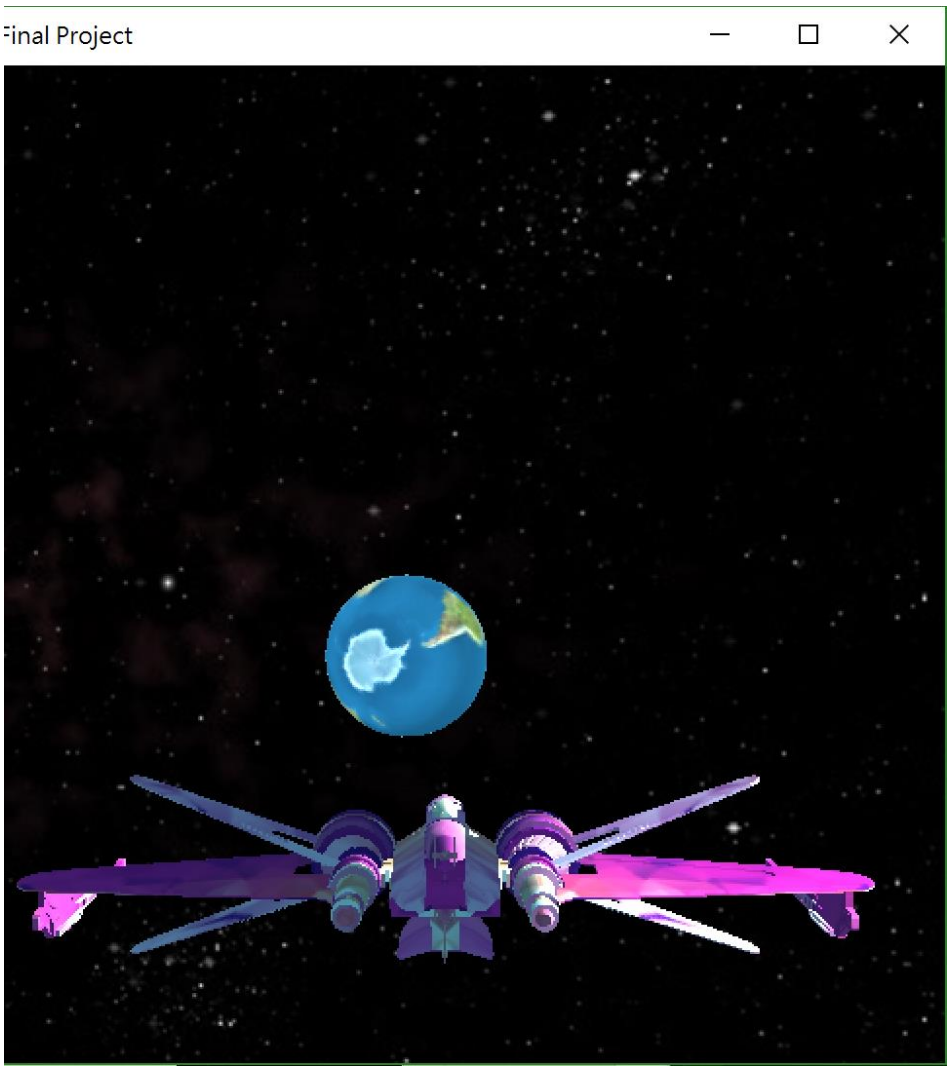
1. We add another light sources.



We add blue ambient light and red specular light although it is not clear to show.

2. Normal Mapping

We add normal mapping into the earth object. It looks like:



Compared with non-normal mapping,