CSCI 3260 Principles of Computer Graphics

Course Project: Rotating Energy Rings (20%)

Demonstration Date: December 9, 2018

Tutor: Tian Yang

Late submission is NOT allowed

No Skeleton Code is provided, so START FROM SCRATCH EARLY!

Fail the course if you copy!

I. Introduction

In the year 3018, you drive a spacecraft to explore the "Wonder star". After the exploration, you have to maneuver your spacecraft to pass through several rotating "energy rings" to gain enough energy to return to the Earth (See Fig. 1).

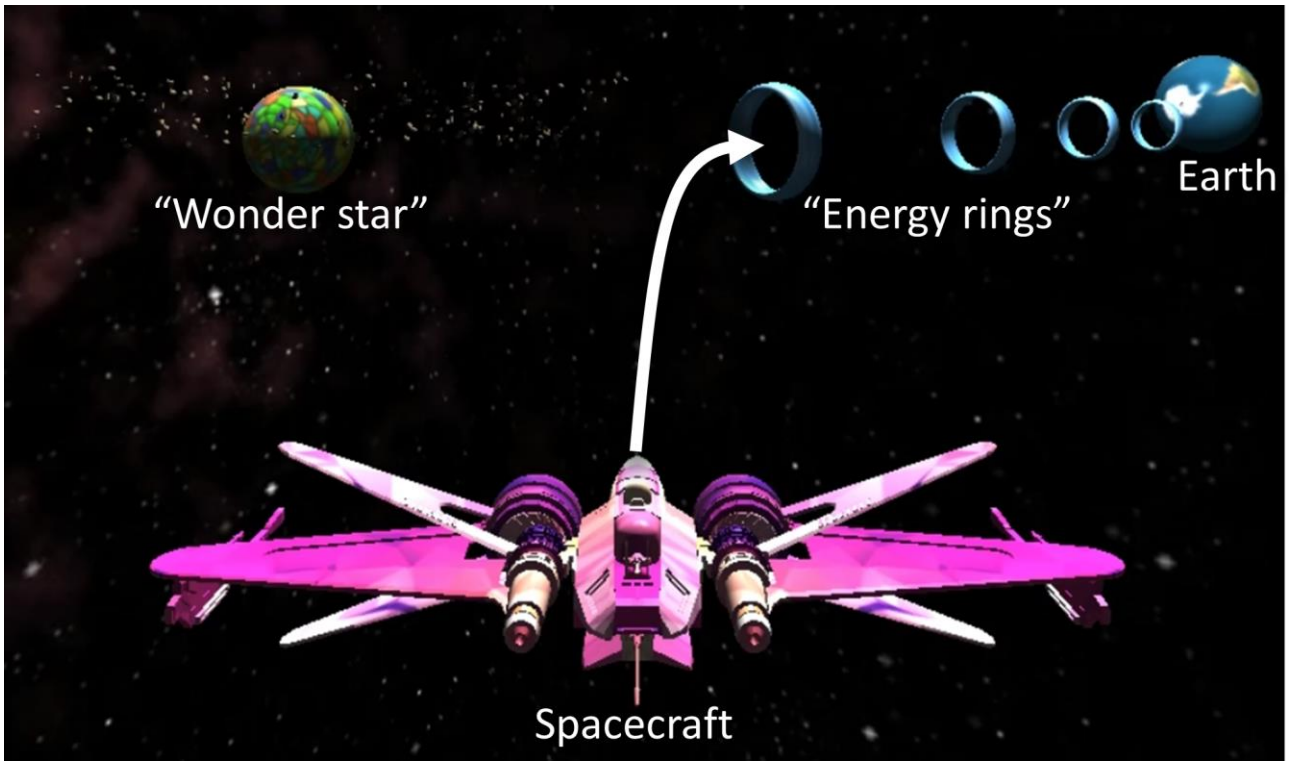

Fig. 1. The overall view of the project

You are required to write your own code from scratch to complete this project. All the basic techniques you need have been or will be introduced in our tutorials. Your best skeleton code is the solution programs of your assignments 1 and 2.

The ultimate objective of this project is to give you an opportunity to practice more with the basic but very important topics in Computer Graphics: you have to go through object loading, transformation matrix, lighting, texture mapping, skybox, shader and interaction before you get a satisfying mark.

II. Implementation Details

Basic Requirements:

1. Render the Wonder star, the Earth, the spacecraft and at least three energy rings with corresponding textures. In order to simplify this project, please keep their centroids on a plane that is perpendicular to the Y-axis of the world space.
2. The Wonder star, the Earth and the energy rings should do self-rotation all the time.
3. Create a skybox as the background of the virtual scene.
4. Generate an asteroid ring cloud that contains at least 200 random floating rocks around the Wonder star. These floating rocks should have random locations in a limited range.
5. All the floating rocks of the asteroid ring should move around the Wonder star simultaneously (See Fig.2).

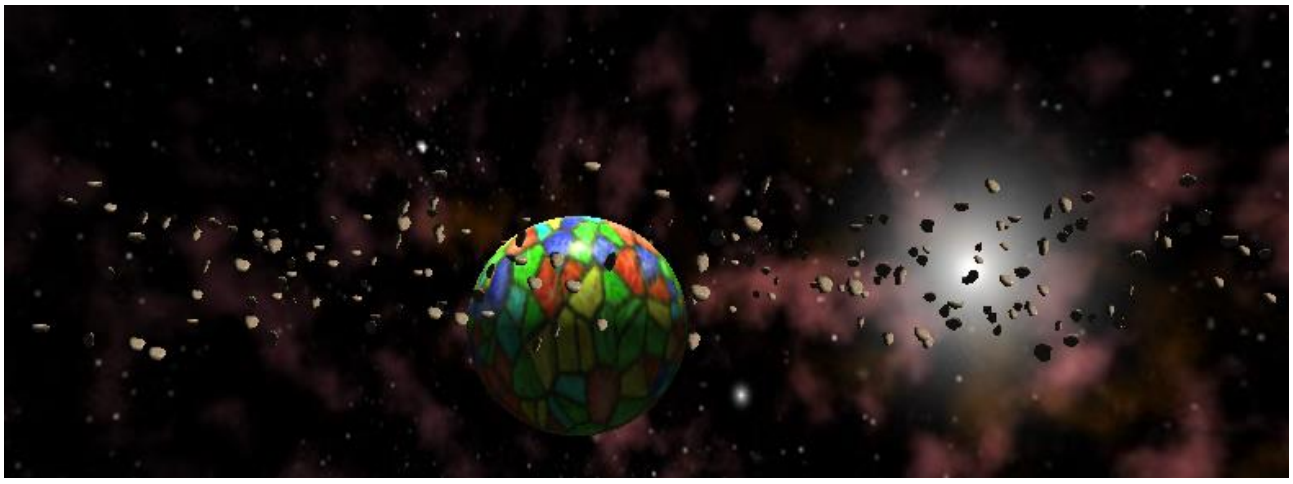

Fig. 2. The floating rocks around the Wonder star

6. Create a point light source. Basic light rendering (ambient, diffuse and specular) should be obviously observed on all objects. Please properly set your lighting parameters for clear demonstration. Keyboard interaction is allowed for you to tune parameters during the

demonstration.

7. The viewpoint should be behind the tail of the spacecraft and relatively static to it. The viewing direction should be consistent with the direction to which pointed by the head of the spacecraft. Watch our demo video for intuitive illustration.

8. For interaction:

   a) Mouse. Use a mouse to control the self-rotation of the spacecraft. For example, if you move the mouse to the left, the head of the spacecraft will turn left.

   b) Keyboard. Please use the following four keys to control the translations of the spacecraft:

      i. Up cursor key: Move the spacecraft forward by a certain distance.

      ii. Down cursor key: Move the spacecraft backward by a certain distance.

      iii. Left cursor key: Move the spacecraft to the left by a certain distance.

      iv. Right cursor key: Move the spacecraft to the right by a certain distance.

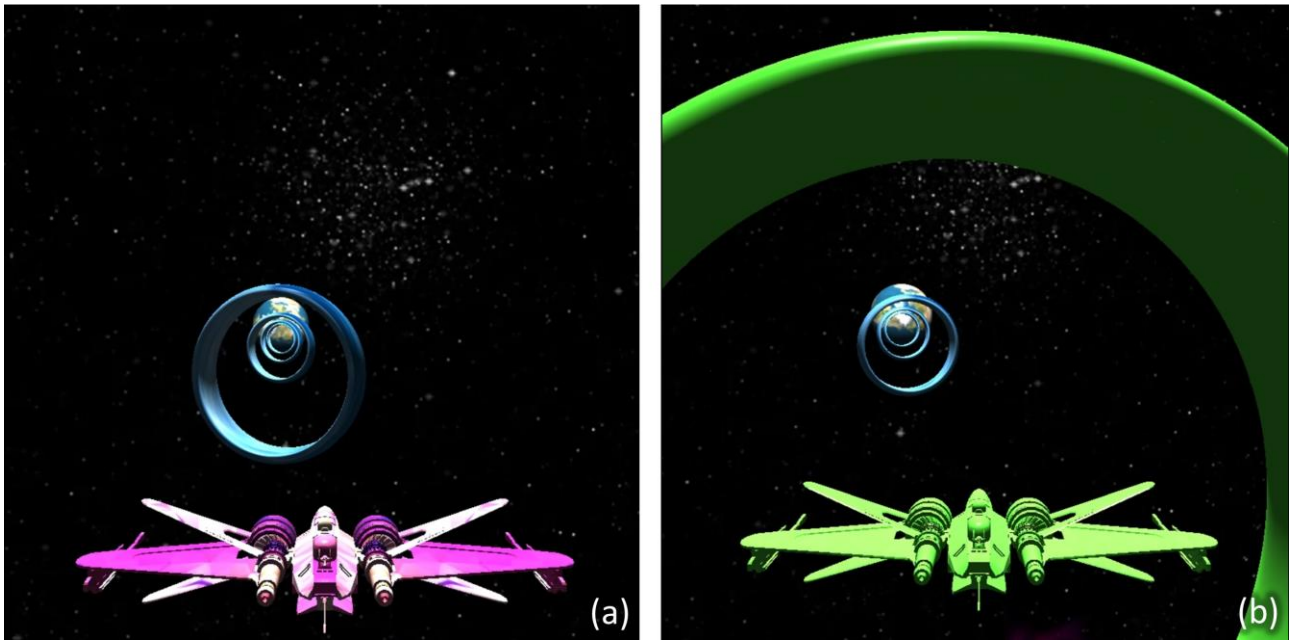

Fig. 3. (a) The spacecraft is approaching an energy ring. (b) The spacecraft is passing through an energy ring.

9. Visual feedback. While the spacecraft is passing through an energy ring, both the energy ring and the spacecraft turn green (See Fig. 3). You can use a green texture to achieve this effect.

Bonus requirement:

1. Add another light source. The basic light rendering result of two light sources should be determined according to the summation property of the Phong Illumination Model.
2. Do normal mapping for the Earth. We provide a normal map for the Earth. You should load both the Earth texture image and the normal map image to use them in the fragment shader.
3. Collision detection for the spacecraft and rocks/planets. Once a collision happens, disable the rendering of one object to make it disappear. We do not require the collision detection for the spacecraft and the energy rings.

III. Framework and Files

1. We provide the basic .obj files of the objects in this project (planet, spacecraft, energy ring and floating rock). Corresponding texture images are also provided.
2. We provide a demo video. Do watch it carefully to fully understand our requirements.
3. Your solution programs of assignment 1 and 2 should provide you a good starting point. Most tasks in this project can be decomposed into easy tasks that have been taught in our lectures and tutorials. We provide some suggestions for you:

   •Keep a good knowledge of the transformations among model space, world space and camera space.

   •Keep a good knowledge of rendering pipeline, VAO and VBO. You may be confused by handling so many objects at the same time. Try to use VAO and VBO to help you figure out, because various information of rendered objects can be attached with those items.

   •Try to keep clean coding style. Clean coding style is helpful for debugging. Keep your mind clear by proper annotations. Also, try to enclose the repeated codes into functions.

4. Recommended libraries:
   - FreeGLUT for creating and manage windows containing OpenGL contexts.
   - GLEW for querying and loading OpenGL extensions.
   - GLM which is a C++ mathematics library for graphics software.
   - Assimp for loading 3D object models from .obj files.
   - SOIL for loading textures from images.

IV. Report

Prepare key-frame snapshots of your scene in a report file (.pdf), basically including the following

parts:

1. A figure which shows the overall scene like fig. 1.
2. The frames which provide close look at the basic light rendering results on each kind of the objects.
3. The frame which shows that the spacecraft is passing through a ring with their colors changed.
4. The frames that can represent any bonus features that you have implemented.
5. Some brief and necessary descriptions of your implementation details.

V. Grading Scheme

Your final project will be graded by the following marking scheme:

	Basic	80%
1	Render two planets, a spacecraft and at least three energy rings	6%
2	Self-rotation for planets and the energy rings	6%
3	Render a skybox	6%
4	Render an asteroid ring cloud	12%
5	The rotation of the rocks	4%
6	Basic Light rendering	8%
7	Correct viewpoint	8%
8	Use mouse to control the self-rotation of the spacecraft	8%
9	Use keyboard to control the translations of the spacecraft	12%
10	Visual feedback for the spacecraft passing through an energy ring	10%
	Bonus	15%
1	Add another light source	5%
2	Normal mapping for the Earth	5%
3	Collision detection	5%
	Report	5%
	Total:	100%

Note: Considerable grade deduction will be given if the program is incomplete or fails to compile during the demonstration.

VI. Project submission and demonstration guidelines

1) Find your group member early. At most 2 members in one group.

2) Compress your project files (.h, .cpp, shaders) and project report in a .zip file. Name it with your group number (e.g. group01.zip) and submit it to the eLearning Blackboard before 9:30am, December 9, 2018. Only one student of each group has to submit. Late submission is not allowed.

3) The project demonstration will start at 10:00am, December 9, 2018. Time slot for each group will be collected later and announced via eLearning Blackboard. If your group is unavailable to demonstrate on that day, please inform us one week earlier with convincing proof for your absence, and we will arrange another day for you.

4) We will announce the demonstration venue via Blackboard later.

5) Please come to the demonstration venue earlier before your time slot to test your program. Make sure it can be executed successfully during the demonstration.

6) A few questions will be asked during the demonstration. You will be asked to explain some of the codes in your program and discuss about how the features are implemented.