CSCI 3260 Principles of Computer Graphics

Course Project: Universe Travel (20%)

Demonstration Date: Dec 2 (Friday), 2016

No Late Submission is allowed

Fail the course if you copy

I. Introduction

In this project, you are basically required to build a universe-like world and try to travel around and between planets. Once you implement our basic requirements, then you are the director for the rest of the story. The basic scene is like the following picture:

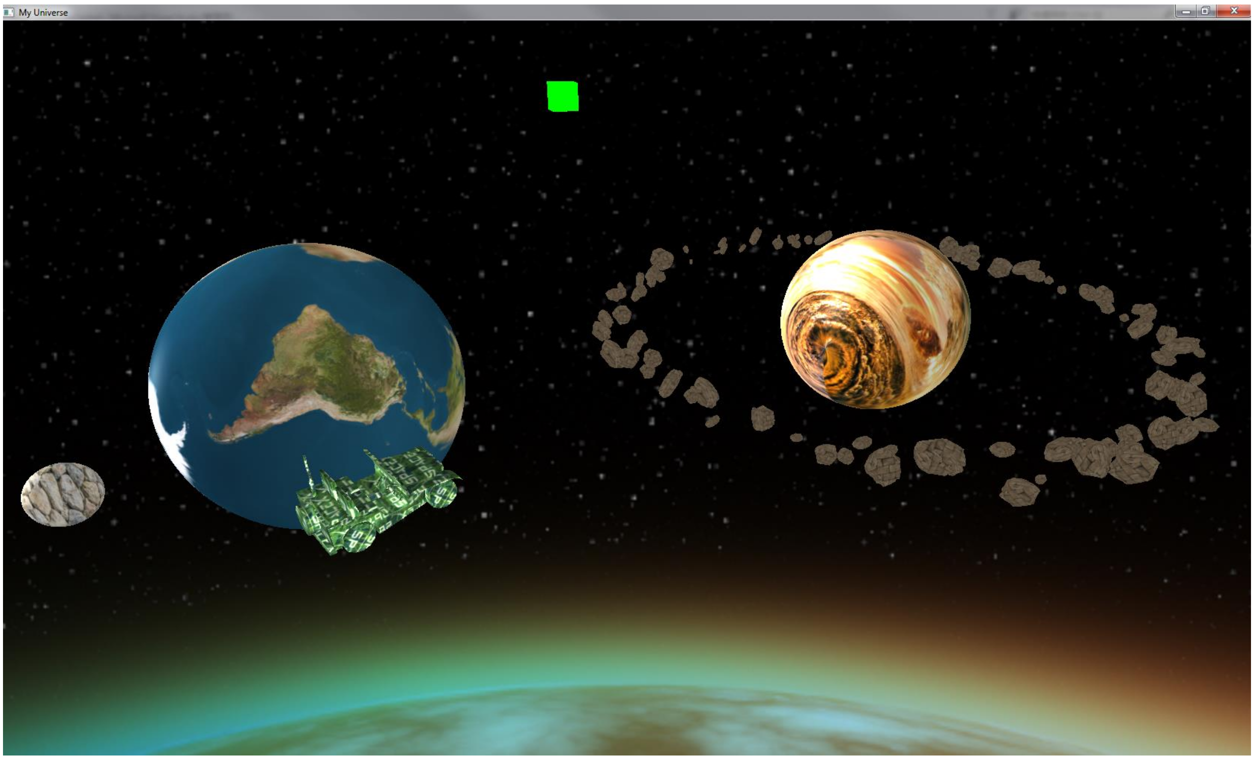
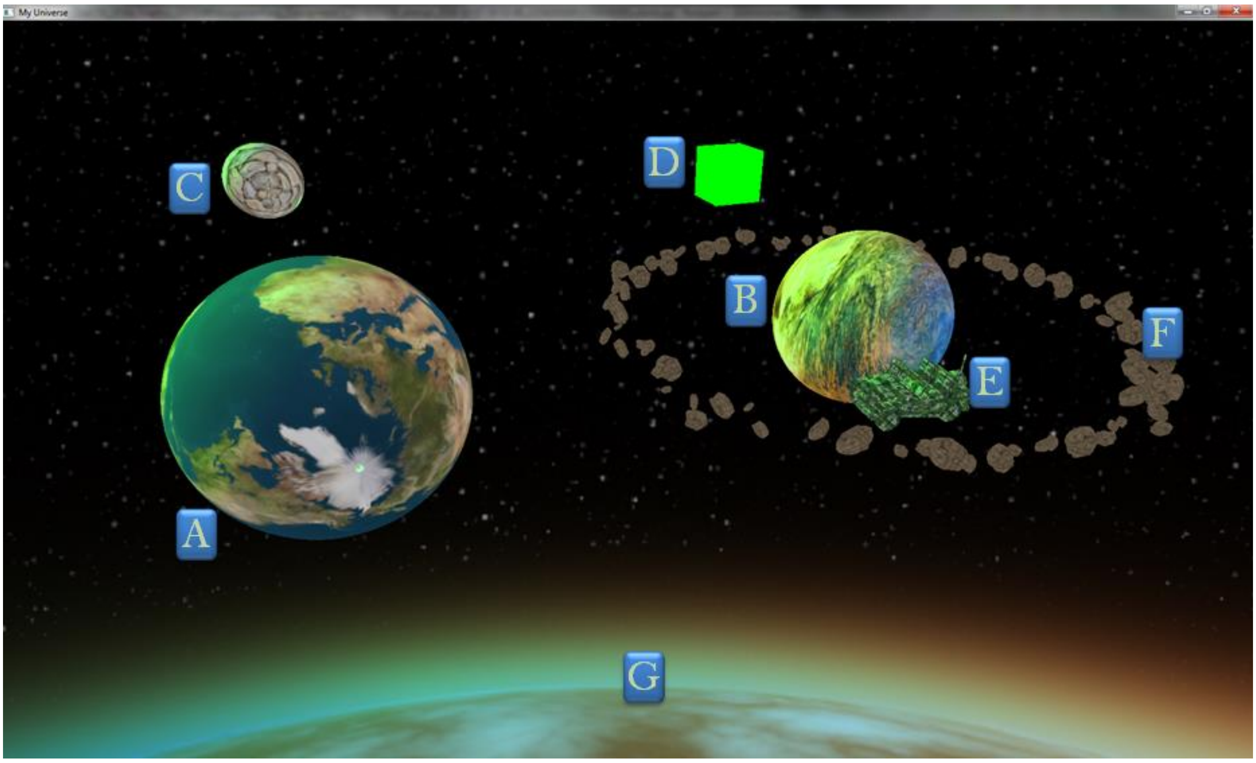


Fig. 1. Scene of the universe transportation.

You are required to write your own code from scratch to build all elements in your Universe, since all basic techniques needed have been or will be introduced in our tutorials. Your best skeleton code is your assignments 1 and 2. We will only provide some very short code for necessary support. The program should allow the user to explore the Universe from different aspects. Another snapshot is shown below.



The ultimate objective of this project is to give you an opportunity to practice more with the basic but very important topics in graphics: you will go through *object loading, transformation matrix, viewpoint switch, skybox, lighting, (multiple) texture mapping, multiple shader, normal mapping, instancing, interaction and GUI* before you get a satisfying mark. You have played with these basic techniques in assignments 1 and 2, but running them together is really a challenge. Once you get to the destination, you will have a good insight of the rendering pipeline in OpenGL and hope to learn more about the OpenGL for more complex rendering.

II. Implementation Details

Basic Requirements:

You should accomplish the following goals to get the basic points.

1. Create 3 planets (A, B, C), 1 light source box (D) and 1 space vehicle (E) and place them properly. You can load any possible objects or images as the planets, light source, vehicle and Skybox. Keep the aspect ratio of the scene correct when we change the window size with “glutReshapeFunc” function.
2. Create a Skybox (G) as background and it should rotate accordingly when we you change the view direction around the Universe. No translation should happen for Skybox.
3. Generate an asteroid ring cloud F which contains at least 200 random floating rocks around planet B. Those floating rocks should have random sizes and locations in a limited range.
4. Conduct single texture mapping for planet C, space vehicle E and every rock in F; Conduct multiple texture mapping for planet B; Conduct normal mapping for planet A. Basically, only pure color is required for the light source box D. Textures used for A, B, C, E and F should be different from each other. You can use any texture you like, even a chocolate or strawberry.

5. Basic light rendering (ambient, diffuse and specular) should be observed on planet A, B, C, and the space vehicle E.
6. *For planet A and B*, they should do self-rotation all the time. To be a little bit physically right, it should be better if speeds of A and B can be associated with their volume size.
7. *For planet C*, it should move around A at a designed orbit (circle or other you like). Also, C should have a self-rotation at the same time, just like what moon does against earth.
8. *For light source D*, it should move from negative Z to positive Z along a straight line which lies somewhere between planet A and B. It should be better if you can make this movement periodically, just like sunrise and sunset. Generally, the light source is not visible, but here we should put an object which just locates at the light position to visualize the light source.
9. *For space vehicle E*, it should firstly move around A at a designed orbit before we start our space travel.
10. *For asteroid ring cloud F*, all the floating rocks in it should move around planet B simultaneously.
11. *For viewpoint switch*, at least 3 distinctive viewpoints (such as –X, +X, +Y axis) should be provided as choices so that we can explore your Universe from different aspects.
12. *For space travel*, the basic setting is, before travel starts, the space vehicle E should move as described in No.7 requirement. Once we send the order to start travel, the space vehicle E should be transformed to a fix site on its orbit immediately. Then, the vehicle E should move to a fix site close to planet B gradually.
13. *For interaction*:
    a) *Keyboard*. Please set lower case ‘a’, ‘s’, ‘d’ as switch between 3 different viewpoints; Please set upper case ‘T’ as the switch to start space travel. Please set the up and down arrow keys as speed control with which we can control the speed of the space vehicle. ‘Esc’ is preserved for exit.

    b) *Mouse*. Please make the setting so that we can zoom in/out when we use the middle wheel of mouse. Control the position of camera by mouse (same as assignment 2) , which means:

       i. When mouse moves up, the whole scene we see moves down.

       ii. When mouse moves left, the whole scene we see moves right.

Bonus Features

YOU CAN ACHIEVE ALL BASIC REQUIREMENTS by using the information from the course and tutorials. Once you have done that, you are the director for the rest of the story. Presenting your creativity should be more exciting than finishing the project. Here are several suggestions from us.

The bonus points are given depending on the excellence and difficulty of your work. Implementing challenging and diverse visual effects will get you higher score. Here are some examples:

- Add another visible light source. All basic light rendering results should then be changed according to the summation property of the Phong Illumination Model. *(5 points)*
- Provide an extra keyboard option so that we can attach our viewpoint to the space vehicle. By doing this,

we can really enjoy our space travel and have a close touch with the planet and floating rocks. *(4 points)*

- Continue the travel: once the space vehicle E arrives at the site close to B, let E start the next movement at a new orbit around planet B. Also, you need to pay attention to the speed of E, since now it's on a planet with a different "gravity coefficient". You can even try to change the appearance of planet B once the vehicle arrives at the site close to B, just like space emigration will do to other planets. *(5 points)*
- Generate the asteroid ring cloud with more than 5000 floating rocks. Some basic generation method will becomes un-renderable as so many repeated objects need to be rendered at the same time. So you are encouraged to explore the amazing instanced rendering. *(6 points)*
- Use more "advanced" texture mappings to make objects more realistic, such as shadow mapping environment mapping, displacement mapping etc. *(7 points)*
- For the space vehicle: load complex objects from .obj file with the library Open Asset Import Library (Assimp) and render different parts of the object with different textures to form a more fantastic vehicle. *(7 points)*
- Create a GUI with the GLUI User Interface Library so that more vivid control interactions can be illustrated. If you decide to use GUI, then please basically provide the control items (buttons, sliders) with the same functions as that described in *Keyboard interaction* section. *(8 points)*
- Generate some fog or create a tunnel between planet A and B, so that the journey will seems to be more curious when you travel with your space vehicle to planet B. *(5 points)*

Other interesting and fantastic bonus idea will be graded accordingly. The full mark for bonus part is 20 points.

III. Framework and Files

1. We will provide the basic .obj files for the planet, vehicle and floating rock, also texture images and skybox. You are encouraged to illustrate your creativities with the items you like.
2. Your assignment 1 and 2 should provide you a good starting point, and you can immediately start this project from them. Most of the tasks in this project can be decomposed into easy tasks that have been taught in our lectures and tutorials, but before you start, several things need to be mentioned:
   - Keep a good knowledge of the transformation matrix. Although no complex rendering algorithms are involved in this project, but a lot of rendering are realized by hierarchical transformation matrices (such as the self-rotation, rotation around other planet, matrix array for floating rock, and viewpoint switch). So, being good at playing with matrix is important for this project.
   - Keep a good knowledge of rendering pipeline, VAO, VBO. You may be confused by handling so many objects at the same time. Try to use VAO, VBO object to help you figure out, because various information of rendered objects are attached with those items.
   - Get a good practice with multiple shaders and multiple shader program objects. Equip your shader programs for different objects with the right components, and then enable your shader program for specific object to render at the right time.
   - Try to keep clean coding style. Since more than 7 objects need to be rendered, a lot of stuff (lighting, transformation matrix and texture etc.) need to be done before drawing, and those stuff are placed in one paint function, so try to keep your mind clear by clear annotation. Also, try to enclose the repeated codes into functions. Clean coding style is also helpful for debugging.
3. The libraries that you can use are limited as the following list: FreeGLUT, GLEW, GLM, GLUI, Assimp

and SOIL. If you really need more libraries to realize more fantastic rendering effects, please contact tutors for permission.

IV. Report

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

1. The frames from different timepoints which can illustrate the movement of all moving objects before space travel begins.
2. The frames which can provide close look at the basic light rendering results (zoom in function can facilitate you in this part).
3. The frames which can provide close look at the normal mapping rendering results on planet A.
4. The frames which illustrate the space travel procedure: stay at fix site close to planet A -> travel between planet A and B -> arrives at the site close to planet B.
5. The frames which are captured from 3 distinctive viewpoints.
6. The frames that can represent any bonus features that you have implemented. Some brief and necessary descriptions of your implementation details are appreciated.

V. Grading Scheme

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

	Basic	**80%**
1	Object creation (3+4+2 points)	9%
2	Create the skybox (4 points)	4%
3	Generate the asteroid ring cloud (7 points)	7%
4	Texture mapping (2+2+3+3+3 points)	13%
5	Light rendering (3+3+3 points)	9%
6	Self-rotation of planet A and B (2+2 points)	4%
7	Self-rotation of planet C, rotation around planet A (3+4 points)	7%
8	Movement of Light source box (3 points)	3%
9	Space vehicle E move around planet A (3 points)	3%
10	Floating rocks move around planet B (5 points)	5%
11	Viewpoint switch (3 points)	3%
12	Basic space travel (7 points)	7%
13	Interaction (3+3 points)	6%
	Bonus	**20%**
	Total:	**100%**

Note: Considerable grade deduction will be given if the program is incomplete or fails compilation in demonstration.

VI. Project demonstration and submission guidelines

1) The project demonstration will start at **10:00 am, 2-Dec-2016**. Randomized time slot for each group will be announced via eLearning Blackboard. Please upload your project files to eLearning **before 9:30am** of the due date. **No further modification or late submission is allowed**. You cannot update the uploaded codes (.h, .cpp and shaders) in you demonstration. If you group are unavailable to demonstrate on that day, please inform us one week earlier with convincing proof for your absence, and then we will arrange another day for you.

2) We will announce the demonstration venue via eLearning after we reserve it.

3) Please come and test your program in the demonstration venue earlier to make sure it can be executed successfully during the project demonstration.

4) 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 features are implemented.

5) After your demonstration, zip the whole project and your report in a .zip. Name it with your group number (e.g. group01.zip) and submit the .zip via eLearning Blackboard. (https://elearn.cuhk.edu.hk/) (Only one student of each group has to submit)

VII. Acknowledgements

Thanks Prof. Heng, Prof. Philip, Xianzhi Li and Yueming Jin for discussion and contribution.