

Project

Animation of the Solar System

CS104: Fundamentals of Computer Graphics

Hon-Cheng WONG

Faculty of Information Technology




Macau University of Science and Technology
Macao, China

General information



- ⌘ This project can be completed by **one** or **two** students;
- ⌘ Due day: **12 December 2020 (Saturday)**
- ⌘ Please submit your **codes** and **report** to:
hcwong@must.edu.mo
- ⌘ Please don't submit the whole Microsoft Visual Studio project as its size is quite large (around 500M).

Project objective



- ⌘ This project aims to get you familiar with transformations, texture mapping, lighting and window event handling.
- ⌘ In this project, you need to use various drawing primitives, transformations, and window event handling functions given in **Modern OpenGL (vertex and fragment shaders), GLEW, GLFW, GLM, and SOIL2**, to write an animation for navigating the virtual solar system.

Project description



- ⌘ This virtual solar system consists of the Sun, eight planets (Venus, Earth, Mars, Jupiter, Saturn, Uranus, and Neptune), the Moon and a star-sphere (see the figure in the next slide, here we only show the first three planets).
- ⌘ The Sun is stationary during the whole animation. The eight planets circulate around the sun and self-rotate at constant angular velocities. The Moon circulates around the Earth with **NO** self-rotation. The whole solar system is surrounded by a star-sphere, which is a collection of stars located on a surface of a sphere.

1. *Journal of the American Medical Association*, 2000; 284: 2689-2695.




Implementation details



- ⌘ You can make use of **Program 4.4** and **Exercise 6.1** for getting started the project.
- ⌘ Get the demo from the course FTP to see how it should behave. Note that this demo was implemented with **Old OpenGL (fixed functions)**. You need to use **Modern OpenGL (vertex and fragment shaders)** to implement the project. **It is not necessary for your project to get exactly the same output as the demo.**

Implementation details


Modeling and transformation



- ⌘ In this part, you have to set up the virtual solar system. In order to enable texture mapping on the sphere, you are suggested to use Sphere class to create the planets.
- ⌘ The radius of the planets and that of the orbits are defined at the beginning of the C++/OpenGL application. You may use points to create the star-sphere.
- ⌘ You can define the reasonable orbit and radius for each planet.

Implementation details

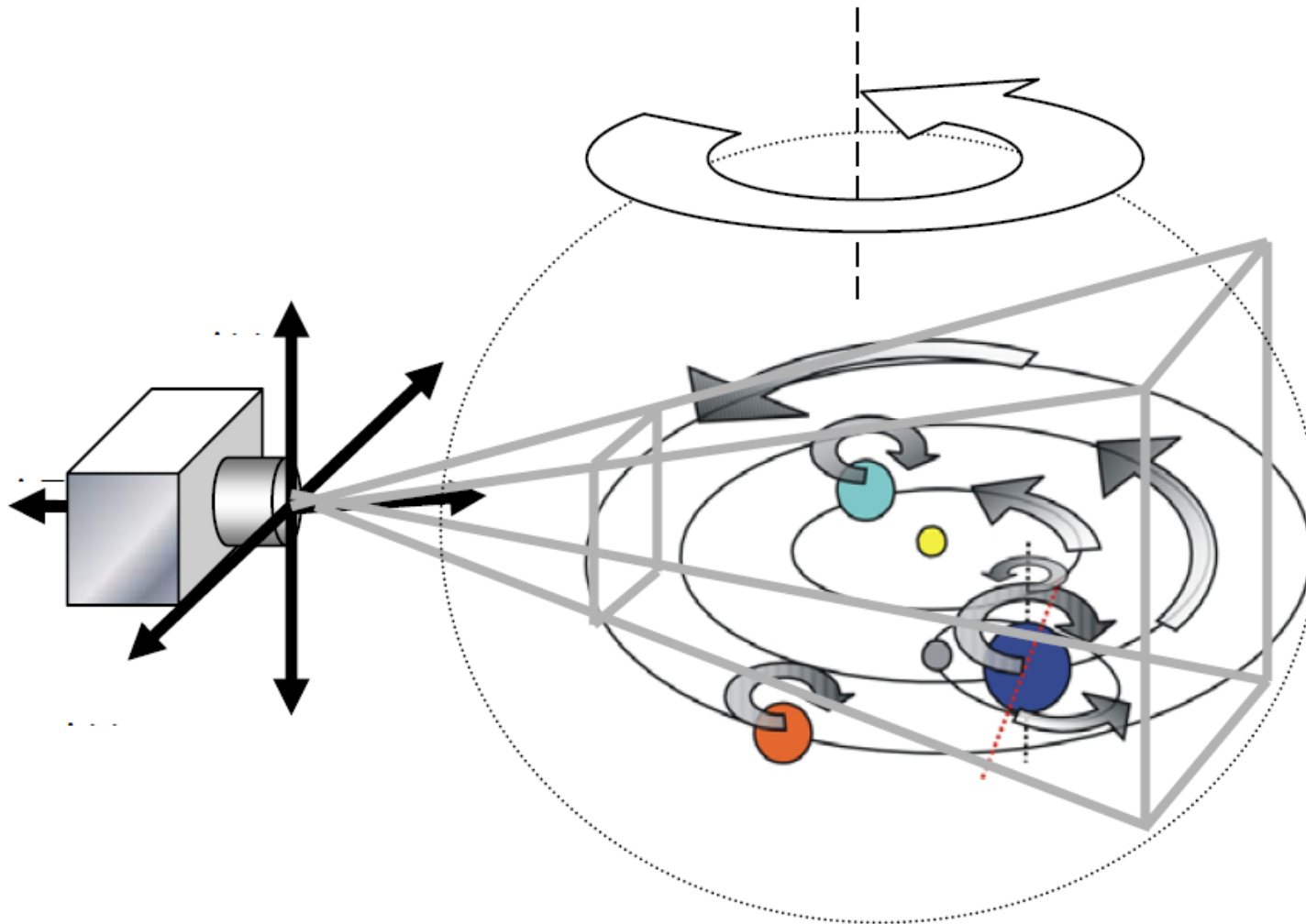
Modeling and transformation



- ⌘ After creating the planet models, you are required to perform various 3D transformations on the models. If we view from the top, the eight planets should circulate around the Sun in anti-clockwise direction and at the same time self-rotate in clockwise direction.
- ⌘ The Moon should also circulate around the Earth in anti-clockwise direction. In addition, you have to rotate the whole solar system, together with the star-sphere, in anti-clockwise direction. The overview of the transformation can be shown in the figure in the next slide.
- ⌘ You can define the rotation speed for each planet.

Implementation details

Modeling and transformation



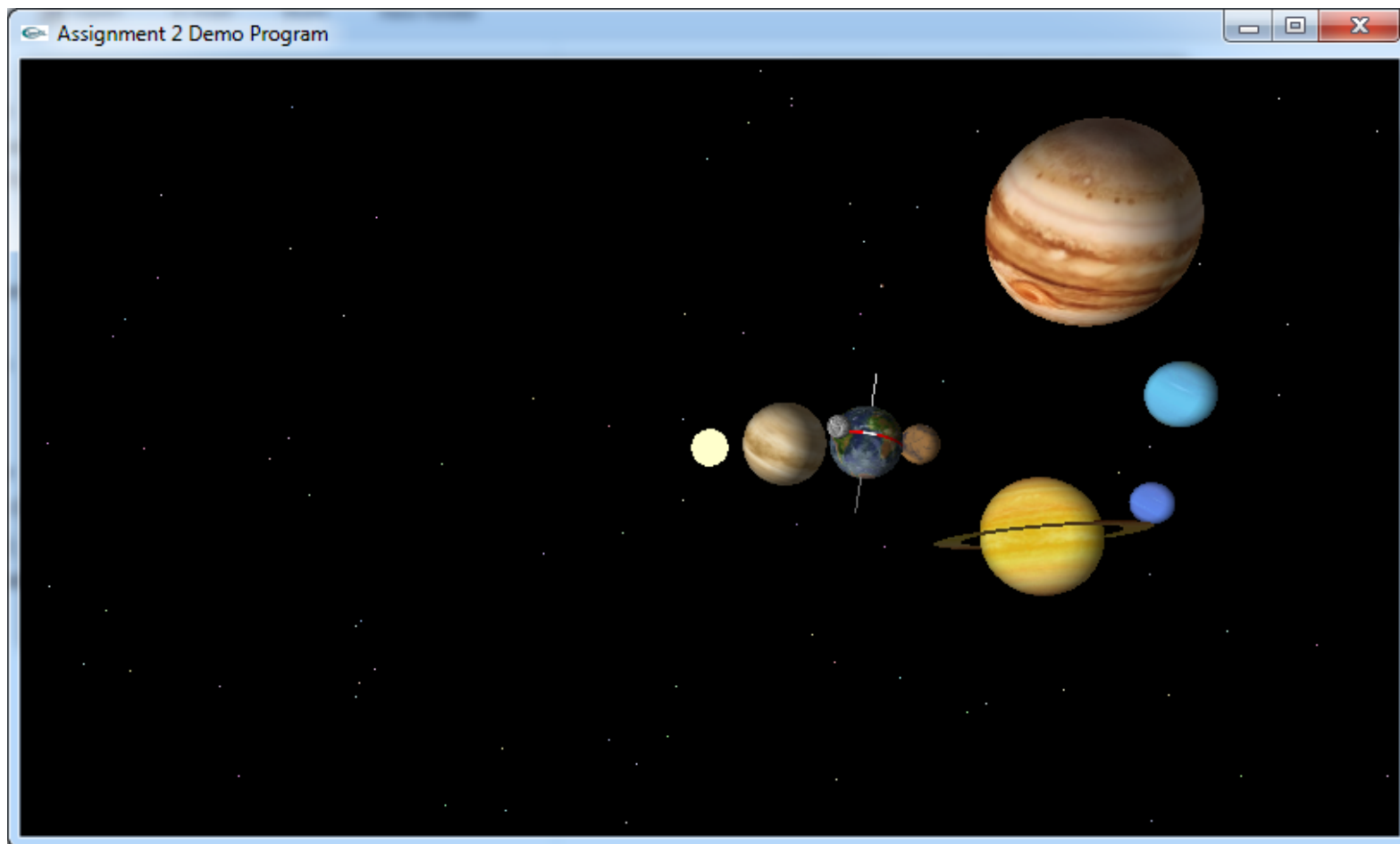
Implementation details

Texture mapping and lighting



- ⌘ You have to map the corresponding textures onto the surface of different planets. The texture maps (in bitmap format (*.bmp)) are provided.
- ⌘ The only light source for the whole solar system is the Sun. You should create a point light source at the same location with the Sun. Note that the point light source has no shape, thus you have to create a sphere to represent the Sun.

Example output of the project



Report



- ⌘ An English written report is required, and NO mark will be given to Chinese reports.
- ⌘ For your written report you need to include at least the following sections:
 - ☑ Design outline;
 - ☑ Key code fragments, and shaders of your project;
 - ☑ Output of your project;
 - ☑ Your feelings or opinions about this project.

Submission



- ⌘ You are required to submit a **softcopy** of your report. For your finished project, please zip your **source codes (*.cpp, *.gls)** and then sent it to E-mail box (**hcwong@must.edu.mo**) with the **subject title “Project”**. Your name(s) and the student ID(s) should be included in both your report and source codes.
- ⌘ **Missing any above item may lead to mark deduction.**

Grading scheme



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

- ☑ C++/OpenGL application: 10%
- ☑ Vertex shader: 10%
- ☑ Fragment shader: 10%
- ☑ English report: 10%.

⌘ The total grade of this project is **40%**.

⌘ If the project are done by two students, both students will receive the same mark.