|  |  |  |
| --- | --- | --- |
|  | **Rochester Institute of Technology**  **Golisano College of Computing and Information Sciences**  **School of Interactive Games and Media**  **2145 Golisano Hall – (585) 475-7680** |  |

**Data Structures & Algorithms for Games & Simulation II**

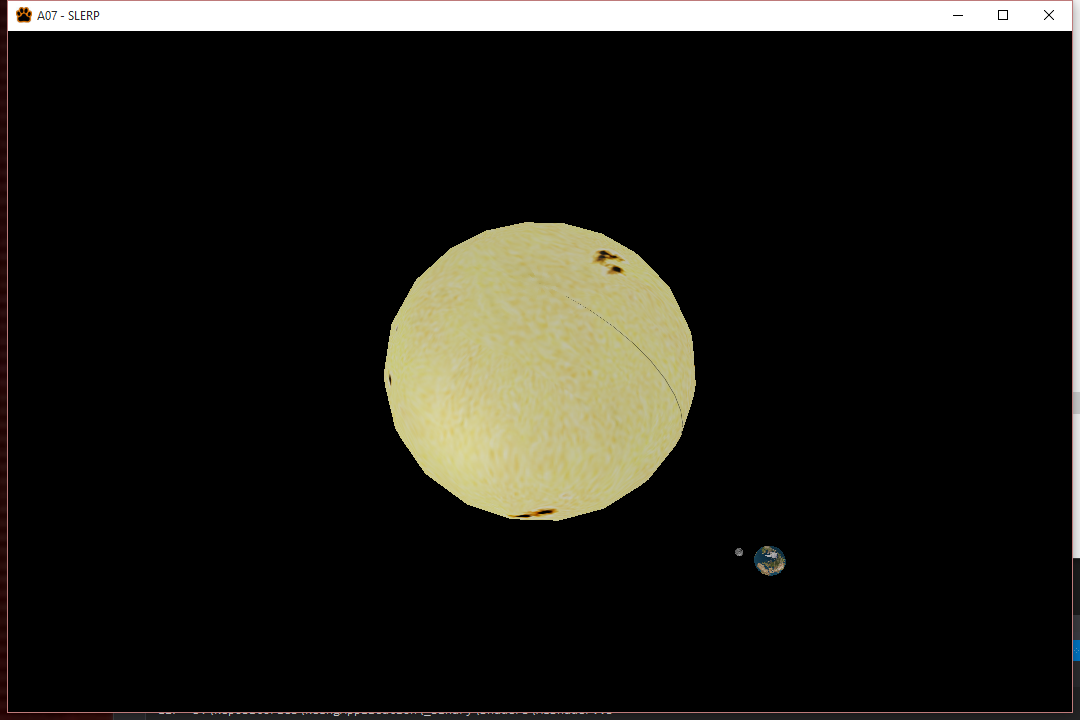
**IGME 309, 2016-17 Spring**

**A7: SLERP**

**Due: March 26th 2017 (first Sunday after break) at 23:59Hrs.**

Instructions:

You have starting code through ReEngine, you may use your own solution if you want to, but the translation of starting code is entirely your responsibility. Startup code is available at: <https://github.com/labigm/ReEngineApp_2016s> under the **A07\_SLERP** project. An example binary may be found under the \_Binary folder as usual.



This homework assignment is similar to A05\_Transformations but your implementation needs to be timed-based and all of your orientations performed though quaternion interpolation.

The Sun is going to be static in the origin of the World Coordinate System and the Earth is going to perform a 360 degrees rotation around it in 365 seconds (Unless a different time is specified). During that time it should have rotated around its own axis 365 times (A second is an earth’s day). The Moon is going to be revolving around its own axis but will also be revolving around the Earth (which again, in turn, should be revolving around the sun) in such a way that it only shows one face to the Earth at all time; its rotation period should be 28 earth-days.

The scales and distances to be used will be:

|  |  |  |
| --- | --- | --- |
| Planet/Star | Size | Distance from the sun |
| Sun | 5.936(relative to the world) | 0(relative to the world) |
| Earth | 0.524(relative to the world) | 11(relative to the world) |
| Moon | 0.27(relative to the Earth) | 2(relative to the Earth) |

More info about moon’s rotation in: <https://www.youtube.com/watch?v=OZIB_leg75Q>

You are provided with the following variables:

float m\_fDay; Which is how much time in seconds a day is. You can modify this time though the use of plus and minus in your keyboard making the day’s length shorter or larger.

float fEarthHalfOrbTime = 182.5f \* m\_fDay; Which is how much time the Earth would take to do half an orbit around the sun

float fEarthHalfRevTime = 0.5f \* m\_fDay; Which is how much time the Earth would take to do half an revolution around its own axis

float fMoonHalfOrbTime = 14.0f \* m\_fDay; Which is how much time the Moon would take to do half an orbit around the Earth. Abusing the fact that we can just avoid setting a value to the moon revolution to have an synchronous rotation we will not calculate it.

static int nEarthOrbits; static int nEarthRevolutions; static int nMoonOrbits; Will be counts on how many times the objects have performed the named operations.

All values are time dependent. You will need to come up with a way of counting how many cycles the objects have performed and increase the necessary variables. If you do not modify the starting values of m\_fDay a full revolution of the Earth will take one second and 28 seconds in the case of the Moon.

All your orientations need to be done using quaternion interpolation (SLERP) and even though glm has a glm::SLERP method it is NOT the right method of doing quaternion interpolation for orientations, the right way of doing it is using glm::mix which will take 2 quaternions and a scalar that should determine the reason of change between the two of them.

You will need to investigate the use of:

glm::quat myQuaternion = glm::quat(vector3(0.0f, 0.0f, 0.0f));

glm::mix(qFirstQuaternion, qSecondQuaternion, fPercentage);

glm::mat4\_cast(myQuaternion)

Loading the models is done for you at the beginning of the program, and they are added to the render list for you. You will only need to assign the transformation matrix yourself like so:

m\_pMeshMngr->SetModelMatrix(m4Sun, "Sun");

Extra (15%):

Make the sun move using the X, Y, Z keys (positive but Negative if pressing Shift) and let the Earth and the Moon move with it (revolving of course).

Extra (10%):

Find Venus proportions online and add the model to the program using its rotation and orbit times in scale to Earth’s (1 second).

Submit to the dropbox labeled A7\_SLERP