CMSC 427 Computer Graphics
Programming Assignment 6: Final Project Guidelines
Release Date: November 4, 2015
Due Date: December 15, 2015, 11:59pm
Project Submission:

1) Delete all intermediate files (run the command make distclean) and the Makefile created during the compilation process. Also delete the executable file.
2) Place all files for the project in a folder and ZIP up the folder. You will submit your project via the submit server. To submit a zip file, login to the submit server webpage and look for the link to make a *web submission*.

Description

For the final project, you have much more freedom to work on something of your choosing. You may work by yourself or in groups of up to 3 students total. We will take this into consideration when grading. We will expect more from groups of 3 students. Your final project must be accompanied by a write-up explaining in detail what you implemented and must include screenshots of the approach, and evidence that it works correctly. The write-up should include a section that justifies each of the grading criteria (see below) and why your project meets these objectives. You are required to extend the framework that we have provided for viewing .OBJ models using OpenGL or ray cast rendering in the previous assignments. Your final project should extend this framework in some non-trivial way. If you have any questions, please ask well in advance of setting out to complete your projects.

There will be no extensions or additional time allowed on the final project because we have to submit final grades. If we do not have your project in hand when entering grades, you will receive 0 points.

Timeline

Below is a timeline of the project and milestones that we expect everyone to adhere to.

Activity	Deadline
Project specs released	November 4
Find groups and select a project – notify the professor and TA the group you're in and the project you have chosen.	November 12
Progress Update – each group should email the professor and the TA a short explanation on the current status of the project. A rudimentary implementation of the project should be working at this time (screenshots highly recommended).	December 1
Turn in project and write-up	December 15

Sample Project Ideas

Implement a demonstration of the marching cubes algorithm.
http://threejs.org/examples/#webgl_marchingcubes

Implement dual quaternion skinning.
http://www.cis.upenn.edu/~ladislav/dq/index.html

Implement a loader for FBX files using Adobe FBX API. Your loader must support textures and animations. Provide an easy to use API for FBX files.
http://www.autodesk.com/products/fbx/overview

http://usa.autodesk.com/adsk/servlet/pc/item?id=24746731&siteID=123112

Implement path tracing and illustrate global illumination features such as light bleeding and caustics.
http://madebyevan.com/webgl-path-tracing/

Implement spherical and/or cube environment mapping in the accelerated ray tracing renderer you developed in assignment #5.

Implement image-based lighting by computing irradiance at each visible surface point/pixel.

Implement subsurface scattering.
https://en.wikipedia.org/wiki/Subsurface_scattering

Implement ambient occlusion for rendering 3D models.
https://en.wikipedia.org/wiki/Ambient_occlusion
Implement instancing for ray-cast rendering. Demonstrate how you can use instancing to render a scene with many copies of a model.

Implement a scene complexity reduction algorithm such as view frustum culling and occlusion culling. Illustrate that the algorithms can be used to render a very large and complex scene.

Implement a 3D particle simulation.
https://www.youtube.com/watch?v=opuS5Z2AlNw

Implement a mass-spring system with collisions e.g.
https://www.youtube.com/watch?v=oOUqiylGd04

Implement a simple ridged body physics simulation e.g.
https://www.youtube.com/watch?v=TRpuHj2vWJ0 or https://www.youtube.com/watch?v=tT81VPk_ukU
Or http://chandlerprall.github.io/Physijs/

Extend the OpenGL assignment to implement environment mapping, normal mapping, and displacement mapping.

Demonstrate, using possible geometry and tessellation shaders, how an object made up of NURBs surfaces can be drawn (e.g. http://threejs.org/examples/webgl_geometry_nurbs.html)

Grading Criteria

In your write-ups, you should justify why your project meets these criteria. We understand that some projects are harder to implement than others and that unexpected challenges can come up along the way. Your project should document a few weeks of progress towards achieving your goal, not hurried work that occurred right before the deadline, unless that hurried work resulted in something incredibly impressive.

- ***Technical difficulty:*** How challenging was the algorithm to implement? Did it require deriving mathematical details? Was their some interesting and important implementation detail or challenge to overcome?

- ***Analysis and Correctness*****:** Did the group implement an algorithm that substantially improves performance? How did the group assess whether the algorithm improved performance? Were there any other approaches used to determine if the approach implemented was correct?

- ***Creativity*****:** Is the final project artistically interesting? How could the outcome be used in a more advanced project? Does the project reflect technical creativity where the group had to derive new and interesting math?

- ***Demonstration of Approach*****:** How well does the code turned in demonstrate the algorithm? Are good example models and data used? Is the product easy to use?

- ***Relationship to Course:*** How is this assignment related to the lecture material in the course? How does it go beyond the lecture material?