

**Spring 2006**  
**22C:196 Advanced OpenGL Rendering**  
**Assignment 1**

**Due: Thursday, February 9th at 11:59pm**

**Goal:** Learn to use OpenGL's display lists and vertex arrays to help speed rendering of large models. Additionally, play around with OpenGL's alpha blending.

**Problem 1 (25 points):** Create an OpenGL/GLUT program that loads OBJ (or SMF) files, scales them, and displays them in the center of the screen. By incrementing a counter in your idle function, rotate the object around the vertical axis (i.e., the y-axis). Make sure the object is illuminated with simple OpenGL lighting. Note that the `glm` code provided for 22C:151 will read both OBJ and SMF formats.

- **Part A:** Devise some method for timing how long each frame takes to render. Usually, averaging this over several frames give a more constant framerate. Display this value either in the corner of your OpenGL window or in the title bar. (You will probably want to display frames per second rather than seconds per frame.)
- **Part B:** When displaying an object, allow the user to switch between immediate mode rendering, rendering from a display list, and rendering with vertex arrays.
- **Part C:** Run your program with (at least) the 7 objects provided on the web page while running your program with OpenGL window resolutions of  $128^2$ ,  $256^2$ ,  $512^2$ , and either  $768^2$  or (if possible on your monitor)  $1024^2$ . Make a 4 x 21 table showing how the framerates of the models (with each of the 3 rendering modes) varies with screen resolution.
- **Part D:** Make some observations about the speed of immediate mode versus display lists versus vertex arrays. Also make some guesses as to how many triangles your graphics card can render before it stops being fill limited and becomes geometry limited (or CPU bound). If you do not think your card has yet become geometry bound using any of the models, feel free to render them two or three times each frame to increase the complexity.

**Problem 2 (25 points):** Using the same program as above, select a model and resolution (of at least  $512^2$ ) combination that runs quickly on your machine. Add a checkered floor under the model, as in Homework 7 from 22C:151. Make sure you specify `GLUT_ALPHA` in `glutInitDisplayMode()`.

- **Part A:** Add a green sphere to your scene that moves around the scene (in X and Y) as you move your mouse. Make sure the sphere passes in front of your checkered floor and the OBJ model. Light this sphere using OpenGL lighting.
- **Part B:** Enable alpha blending on the sphere, so that objects behind the sphere can be seen. Bind the '+' and '-' keys to increase or decrease the sphere's alpha value between 0 and 1.
- **Part C:** Load the 'earth.rgb' texture and apply it to the sphere. Code to read RGB files is linked from the web page. Feel free to open the image in GIMP to have a look, but don't convert the image to PPM or BMP files, because they do not support an alpha channel!
- **Part D:** Apply the Earth texture to the sphere (with alpha blending enabled), allowing the user to cycle between the various `GL_TEXTURE_ENV` modes specified by your `glTexEnv*()` function (i.e., `GL_DECAL`, `GL_REPLACE`, `GL_MODULATE`, `GL_BLEND`, `GL_ADD`, and `GL_COMBINE`).

**NOTE:** Please include a README file describing the platform your program runs on as well how to compile your program. If you do not include all the models with your submission (and please don't), tell me where the models must be for your program to find them.

**NOTE:** Also post a web page with images from your assignment as well as the tables you have generated and your answers to Part D. Please e-mail me a link to your web page.