

**Spring 2008**  
**22C:251 Advanced OpenGL Techniques**  
**Assignment 1**

**Due: Thursday, February 14th at 11:59pm**

**Goal:** Learn to use OpenGL display lists, vertex arrays, and vertex buffer objects. Learn to add simple blending.

**Problem 1 (25 points):** Create an OpenGL/GLUT program that loads OBJ 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.

- **Part A:** Add a framerate counter to your window, displaying the count in either the lower-left corner of the window or in the title bar. You may use the code snippets posted online, which should make this easy.
- **Part B:** When displaying an object, allow the user to switch between immediate mode rendering, rendering from a display list, rendering with vertex arrays, and rendering with vertex arrays stored in vertex buffer objects.
- **Part C:** Run your program with (at least) the 6 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 24 table showing how the framerates of the models (with each of the 4 rendering modes) vary with screen resolution.
- **Part D:** Make some observations about the speeds of the four rendering modes. Also make a guess 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.
  - Use the `glutPassiveMotionFunc()` callback to capture mouse movement without requiring the user to click the mouse button.
- **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 (this should modify the alpha component of the material properties).
- **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 various `GL_TEXTURE_ENV` modes specified by the `glTexEnv*()` function (use at least `GL_DECAL`, `GL_REPLACE`, `GL_MODULATE`, and `GL_ADD`).

**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 Problem 1, Part D. Please e-mail me a link to your web page.