CS/EECE 433 Fall 2006
E. Angel

Term Project

Approximately 50\% of your grade will be based on your term project. There will be no final examination. The projects can combine reading, research and programming in varying proportions. A typical project will consist of both a program and a (short) report. Projects are due at the last class. Looking through the book may suggest projects to some of you; others may already have a project in mind. In order that each of you have a project and we agree on what is required for a particular project, I would like each of you to hand in a one page proposal by Oct 24. This will give you some time to look at possible projects and/or consult with me. I will review each proposal and approve, disapprove or suggest modifications. My two concerns are the amount of effort a given project will require (either too much or too little is unacceptable) and that a project will involve computer graphics.

Although there are many possible software packages you can use, OpenGL should be the easiest to use as you all have experience with its basics. It should be the choice for most interactive projects.

I have no preference whether you carry out your project using linux, Windows, or Mac OSX but you must be able to demonstrate your project to me. If you have special equipment you would like to use, such as at your job, explain this in your proposal.

If you want to do a visualization project, you might want to check out the Visualization Toolkit (VTK) which available over the net and can be run either under linux or under Windows. It provides a variety of functions that can be put together, either through tck or a C++ program, to perform a variety of scientific visualization algorithms.

Some possible projects and areas include:

- CAD/painting: projects under this heading can range from a painting package to a three dimensional design system for a particular application areas such as robotics, architecture, VLSI design or circuit layout.

- 3D Modeler: most of our software systems lack modelers. A project in this area might build an interactive modeler whose output could go to high quality renderer such as Renderman.

- Geometric Package: One of the disadvantages of OpenGL is that it is not object oriented. A project in this areas
- could implement a library that supports geometric types such as points and vectors directly. The library could be built on top of OpenGL.

- An Object-Oriented System: Build a simple scene graph system on top of OpenGL that would allow you to describe and render simple scenes with basic objects (spheres, cubes), simple attributes, and lights.

- Ray tracers: ray tracing provides an alternate model for building three dimensional graphics. A simple ray tracer that supports only a few objects types and simple shading is not too difficult.

- Algorithms: You can use the raster facilities of OpenGL (or another system) to either build up a graphics package of your own or investigate a particular class of algorithms such as hidden line algorithms.

- Solid Modeling: Most of the course will deal with objects defined by their edges (lines or curves) or surfaces (polygons or polynomial surfaces). An alternative is define objects as combinations (union, intersection, differences) of solids such as quadrics.

- Contour Plotting and Mesh Plotting: two examples of simple visualizations for displaying three dimensional data.

- Scientific Visualization: many projects can be done using one of the available visualizations systems. A typical project might involve adding a module for a particular type of visualization or creating the software to visualize the data from a particular experiment or simulation.

- Particle systems: One method of procedural modeling in graphics is through a system of particles. Each particle obeys physical laws that depend on what you are trying to simulate. The positions of the particles at each step in a simulation determine where objects are rendered. For example, a mesh of particles might be used to simulate cloth while a group of independent particles may be used as for the locations of characters in an animation.

- Advanced rendering: Many possibilities here. most of which can be done with OpenGL. Some possibilities are texture mapping. compositing, and use of the accumulation buffer.

- Programmable Shaders: Recent graphics cards allow the application programmer to program the rendering pipeline. One advantage is that you can create shaders more complex than the ones based on the modified Phong model that execute in real time. You can use the OpenGL Shading Language (GLSL), a high level language that makes it easy to program these cards.

The above list is not exclusive. You can suggest projects in other areas or come in and discuss possible projects with me.

Proposals are due October 24.