CSCI 480: Computer Graphics
Spring 2005
Assignment #4

Due Fri 04/29/05, by midnight

Putting it all together
In this assignment, you will add shading to your custom wireframe renderer.
The result will be a very basic, but still a complete 3D rendering system!
Once again, the your renderer will be based on the one you made as part of assignment #2, **without** relying on OpenGL.

Shading
Your application is required to support the three basic shading modes: flat, Gourard and Phong, in accordance with Phong reflection model. For scan conversion, you will need to implement Crow's algorithm with Z-buffering.

Input
As for previous assignments, your program will get an OBJ file and a transformations as an input. However, the format of the transformation file is extended with the following (optional) fields:

material <ambient coeffiicients> <diffuse coeffiicients> <specular coeffiicients>
light <light type> <light intensity> <light position>

- *<ambient coeffiicients>, <diffuse coeffiicients>, <specular coeffiicients>* are all (r,g,b) triplets, each value in the [0..1] range (support for specular reflections is optional for extra credit).
- *<light type>* can assume two values: *ambient* and *directional*
- *<light intensity>* is, again, an (r,g,b) triplet, with values between 0 and 1.
- *<light position>* is what the name implies (naturally, irrelevant for ambient light)
- you can assume there is no more then MAX_LIGHT_SOURCES in the file (say, 20, for example).
- sample transformation files will be posted online.

Interactivity
Your renderer will have a graphical user interface (extending your GLUT skeleton should be the easiest way to go). The basic requirement is that the rendered object is displayed on the screen. Interactive transformations (as defined in assignment #3) are not required, but encouraged (may affect the grade favorably, although technically, they are not part of

this assignment). In addition, the user should be able to switch between the three shading modes mentioned above.

Guidelines:

- Submission is in pairs. In addition to electronic submission, you will be scheduled for a short live demo. Stay tuned for details!

- Start early. You may wish work with your partner on separate modules (say, scan conversion and Z-buffer). Allow enough time for integration. This is where things get painful sometimes, especially if the deadline is next morning.

- Beware of potential numerical errors. Prefer *double* over *float,* comparisons to *EPSILON* as opposed to 0.

- **Extra Credit:** Feel free to implement any additional features to your renderer. Points will be awarded for *completed* features only, based on the difficulty and implementation.

- Another option for extra credit (15%): a ***separate*** OpenGL – based renderer with shading.

- Bear in mind that efficiency of your implementation may affect your grade.

- Comment your code generously. Incomprehensible and non-documented code may lose you points.

- In your submission include the following:
  - Your source files, including the *Makefile.*
  - A README file with your name, 10 digit school ID (you and your partner), and anything we should know about your application.

- Submit your assignment using the command below (just one submission per pair):

- *submit –user csci480 –tag hw3 <your files>*

- Have fun with this one! The result will probably be the most rewarding part of this course.