Lab 5 – CSC 572  
Due 11/25/08 to be demoed in lab  
This project is to be worked on in pairs or individually (any pairs you’d like)

Objective: Understand the basics of texture mapping, displacement maps, and normal maps.

You will be modifying your mesh program and/or a provided texture mapping program in order to explore various concepts of displacement maps and normal maps. You will need to generate your own very simple rectangular polygon (orthogonal to the camera view). You will then texture map it with the displacement map image, then create a displacement map rendering using the displament map image and finally you will create a normal mapped rendering by creating your own normal map from the displacement mapped mesh you created and then render it as a normal map. You will want to implement a toggle between different rendering techniques (i.e. a keyboard event).

You will need to download the displacement map (shown above) at:  
http://commons.wikimedia.org/wiki/Image:DisplacementMappingExampleHeightMap.png

A base program, which reads in bmp images and uses them as texture maps can be found at:  
http://users.csc.calpoly.edu/~zwood/teaching/csc471/material/tex_release.tar

Note that you can either look for a png reader or convert the png image to a bmp.

Here are some recommended steps:

1) Create a rectangular polygon and texture map it with the full resolution displacement map image.
2) Create a method to subdivide or grid your base polygon into nxn polygons, where n= (the number of pixels in the displacement map -1). Note that you will want your mesh data structure to support some sense of neighbors for later normal calculations.
3) Write a method to displace each vertex using the displacement map (displace each vertex in the direction of the polygons normal by the scalar amount in the displacement map image (black = no displacement, white=maximum displacement)
4) From the displacement mapped mesh, compute vertex normals based on new neighboring polygon’s positions. Create a normal map, by creating an image and storing the new normals per pixel (again (n-1)x(n-1)). If time does not allow creation of a normal map, you may use a different normal map (e.g.:  
http://www.bencloward.com/tutorials_normal_maps2.shtml)
5) Using your created normal map, write a per pixel shader to render your original base rectangle using the created normal map. See:  
6) Toggle between modes and explore the advantages and disadvantages of these various representations.