Real-time Mesh Decimation

Toren Monson

There are various applications that involve mesh simplification and refinement. In meshes resulting from 3D scans, there will be many areas of the mesh that do not require a complex series of polygons to represent the model. Here simplification can be done. Also, there will be sections that require smoothing as the scans will not be perfect.

In contrast to 3D scans, 3D scenes generated for such things as computer games and simulations will have complex scenes that need to be simplified in sections determined by the point of view of the camera. This requires multiple levels of simplification in the mesh as well as very fast simplification.

For the purposes of this project, I propose to create a mesh decimator which will be made to run quickly over the scene targeting certain areas of the scene determined by some point of reference. There are three different algorithms used in decimation, namely edge decimation, vertex decimation, and triangle decimation. I will choose one or more of them to do the decimation where I would use more than one in the case of comparison.

As an extension to the project, I will look into the difficulty and effectiveness of extending the algorithm to run on a GPU using the CUDA architecture. With this, I can compare effectiveness between GPU and CPU as well as find the bottlenecks that exist in GPU implemented algorithms.

**Short Term Goals:**

2/23: I plan to have obtained various meshes running under a 3D graphics application such as Blender and using blender to perform various levels of decimation.
2/23: A good understanding of the various algorithms used to perform decimation and have chosen one (or more if I choose to do comparisons)
2/23: Three papers under my belt
3/8: Have a basic skeleton program up and running doing decimation albeit very bad
3/8: Good understanding of whether this can be implemented well on a GPU
3/8: Four more papers under my belt