GPU-based LOD of genus-zero models (sketches_0425)

Krzysztof Niski and Jonathan Cohen

Johns Hopkins University

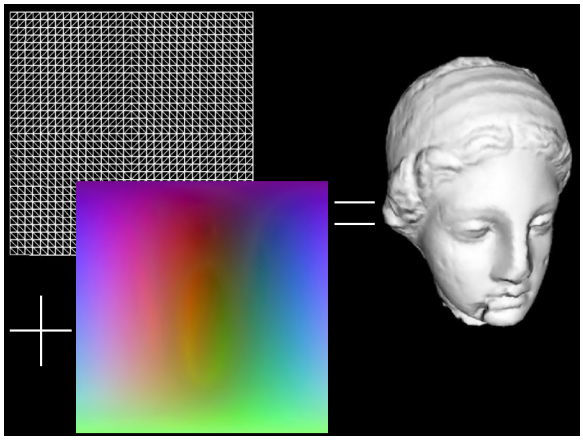

Figure 1: By combining the geometry image of igea with a triangle grid, we can reconstruct the original mesh at the triangle resolution of the grid. The reconstructed mesh on the right is composed of 8k normal mapped triangles

1 Introduction

This sketch presents a novel method of performing Level of Detail (LOD) creation and management that is capable of performing discrete, continuous and view dependant simplification. This method is based on the ability of graphics hardware to perform texture lookups inside vertex programs, which in turn allows arbitrary models to be rendered from a geometry image.

2 Method

The first step in this LOD process is to parametrize the model and recover a geometry image and a normal map from it. The model must be genus-zero in order to be parameterized in this manner. Once the model has been parameterized it is flattened by performing a cylindrical cut. Although the cylindrical cut introduces large amounts of stretch at the poles of the sphere, it allows us to move the image on a plane while still assuring a water tight model, and can be compensated for by optimizing the stretch metric of the parametrization step.

To re-mesh the model we use a vertex program to replace vertex coordinates from a base mesh with coordinates from the geometry map, thus reconstructing the model contained in the geometry image. The base mesh is a tesselated plane, which can be created either at run time or as a pre-process, and will determine the final triangle count of the reconstructed model. By creating planes with different triangle counts as a pre-process we use a discrete LOD model to swap planes depending on the desired amount of detail. If the planes are created at run time, a continuous LOD model is used, as the planes can have an arbitrary triangle count.

To perform view-dependant simplification without recomputing the base mesh every frame we have to be able to shift the texture in two dimensions. Because of the parametrization method, the texture can only be shifted along the u axis without creating discontinuities. To shift the texture along the v axis we create an array of planes in which the over-tessellated area is on a different position along the v axis. Using this approach we can shift the texture along the u axis for free by passing a parameter to the vertex program, and shift it along the v axis by using the appropriate plane from the array.

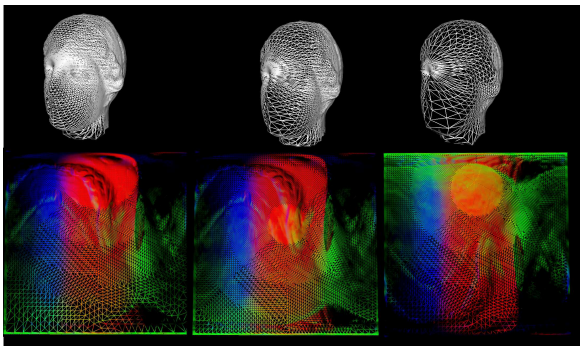

Figure 2: The view-dependent models along with the geometry that created them. The LOD of the front of the face decreases as the camera moves towards the back of the head. In the third image the geometry image is flipped to take advantage of symmetry of the base meshes

The reulsting model is rendered by mapping the geometry image onto the tesselated plane, and rendering it through the vertex and fragment programs. Normal mapping along with manual bilinear filtering inside the vertex program is also used to increase the quality of the reconstructed model.

3 Conclusion

Using this technique it is possible to render a model from a geometry image with very little overhead from the LOD section of the pipeline. The traditionally expensive view dependant node selection operations can be performed very quickly in the graphics hardware by shifting the texture. This method also reduces the amount of data being sent to the graphics card because the base meshes are not model-dependant, and can be permanently stored in video memory. This system can also be used for patch based rendering using seamless texture atlases, giving a coarse grained view dependent LOD system that does not have patch neighbor LOD level restrictions.

References

PRAUN, E., AND HOPPE, H. Spherical parametrization and remeshing. In *Proceedings of SIGGRAPH 2003*.