Inter-frame Caching for High Quality Terrain Rendering

Mårten Larsson*
Norrköping Visualization and Interaction Studio,
Linköping University, Sweden

Magnus Wrenninge†
Digital Domain

Douglas Roble‡
Digital Domain

1 Introduction

At Digital Domain a terrain renderer called Terragen is currently used for creating high resolution images of procedurally generated terrain for visual effects in feature films. This is an object based renderer that generates the mesh representing the landscape by subdividing triangles and shading the triangle vertices in the process. This sketch will describe how we used and adapted a real-time technique called Real-time Optimally Adapting Meshes (ROAM) [Dushaineau et al. 1997] to generate a frame-to-frame caching scheme in order to decrease render times in Terragen. It will also describe how we were able to use frame-to-frame caching for network rendering.

2 Using ROAM techniques

Like ROAM, Terragen uses a view dependent LOD scheme for adapting its mesh to the current view. In each render, a coarse representation of the geometry (for planets this is a box) is subdivided to a level of detail suitable for the current view. This subdivision starts from this very coarse mesh in each frame. To avoid recalculating the surface completely in each frame a caching scheme based on ROAM is now used.
The mesh representation in Terragen is based on triangle bintrees, but instead of using a single bintree for the whole mesh several trees are used. Every triangle in the coarsest level of the mesh (i.e. the box) will be the root triangle of a triangle bintree, all part of the total mesh. The mesh is created in the first frame, and adjusted in the following frames using four distinct loops. First the triangle bintrees are traversed and all non cached values are reset. The bintrees are then traversed and its triangles split to the right detail level. The next loop merges the triangles that are unnecessarily small, and removes triangles that are outside the camera frustum in the current frame. Finally the bintree is traversed and triangles a few levels above the leaf level are added to a list. This list is then sorted so triangles closer to the camera are first in the list. Later this list is used for the drawing process. Having a sorted list saves us from having to compute illumination and shadowing for triangles that are occluded.

We called this new way of using a ROAM-like mesh *Multiple Optimally Adapting Meshes (MOAM)*

*e-mail: marten@martenlarsson.com
†e-mail: magnusw@d2.com
‡e-mail:doug@d2.com

Figure 1: Sample frame.

3 Adapted bucket rendering

The mesh can contain several millions of triangles. This is too many to keep in memory, therefore one mesh per bucket is used. Since very few triangles are shared between buckets in the same frame, and since it's only feasible to have the mesh for one bucket in memory at a time, normal bucket rendering is not suitable, since it would not take advantage of the caching scheme at all. For the caching scheme to work we introduce a concept called *bucket frame*. A bucket frame is a bucket in the image with a given number of frames associated with it. Instead of rendering all buckets in a single frame, a number of frames for a specific bucket are rendered. When one bucket frame is rendered, its tree is removed from memory and the next bucket frame is started. This takes full advantage of the caching scheme. It also works very well with network rendering, where a bucket frame is sent to each CPU instead of a normal frame.

4 Conclusion

Using ideas from real-time rendering we were able to make render times much shorter with no loss in quality for image sequences rendered in Terragen. We have seen render times shortened by up to a factor of five when using MOAM compared to the standard Terragen rendering method.

We found that great care must be taken to ensure that the triangle bintrees do not use too much memory. In particular, buckets containing the horizon typically use a lot of memory. To avoid running out of memory the bucket sizes can be made smaller, or we can stop caching at a coarser level when we start to run out of memory in a bucket. The remaining triangle splits can then be done one by one. After the last subdivision levels are generated and drawn they can be discarded. This makes the caching scheme less efficient, but ensures that frames finish rendering in all cases.

References

DUSHAINEAU, M., WOLINSKY, M., SIGETI, D. E., MILLER, M. C., ALDRICH, C., AND MINEEV-WEINSTEIN, M. B. 1997. Roaming terrain: Real-time optimally adapting meshes. In *Proceedings of IEEE Visualization '97*, 81–88.