Level of Detail for Point Model Rendering

Felipe Carvalho Antonio Oliveira Ricardo Marroquim
Universidade Federal do Rio de Janeiro - UFRJ/COPPE
{fmc,oliveira,ricardo}@lcg.ufrj.br

Abstract

Point-Based representation became a popular alternative to polygonal meshes for representing 3D geometric models. 3D photography and scanning systems acquire the geometry and appearance of real-world objects as point samples. In this work we present a method for efficiently creating a hierarchical multiresolution structure for point models. A variant of the octree is used to partition the space, while the merging of samples is driven by two error metrics.

1. Introduction

Point Based representation has been proposed as an alternative to polygonal meshes (usually triangular) for 3D surfaces, offering a number of benefits [2]. One of the main advantage of discrete point primitives over polygonal meshes is the lack of topological information; only a set of points, with additional attributes such as normal and color, have to be stored and processed. This allows for a simple and compact representation, ideal for efficient rendering and editing.

Today's technology for 3D surface acquisition has reached sampling densities that makes interactive visualization of the acquired data sets a difficult task[5]. With the increase of scene complexity, projected triangles become smaller than a single pixel. In this case, triangle based scanline rendering wastes time in superfluous sub-pixel computation. Level of detail (LOD) methods [3] can remove such tiny geometric details dynamically, but at the expense of a large CPU load for on-the-fly tree traversal and retriangulation. However, the enormous processing power of modern graphics hardware is underused.

2. Space-Partitioning Hierarchy

The multiresolution point representation used is a *point-octree*, which partitions the space adaptively according to the sample distribution (data driven), rather than regularly in space (space-driven), such as *region-octrees* [4].

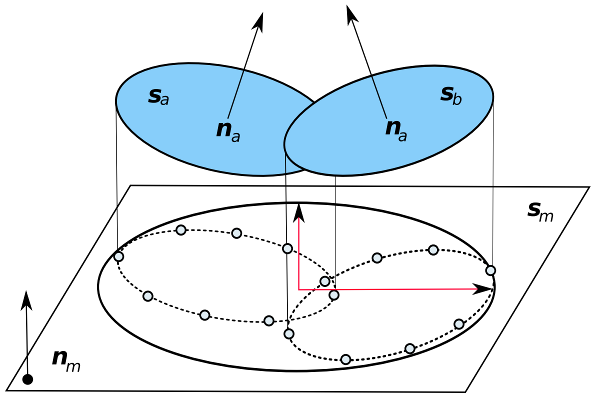

Figure 1. Merging splats

Each leaf node L_n of the hierarchy H, contains a set of k splats $s_c = \{s_1...s_k\}$, while each internal node I_n has a representative elliptical splat covering the extent of its children. Elliptical splats were chosen over circular splats, since the same of amount of surface can be covered with less samples [2]. Furthermore, for efficient back face culling, each $I_n \in H$ also includes a normal-cone with semi-angle θ bounding all samples in s_c.

3. Merging splats

Given a set of splats where each splat has center c_i and normal n_i, the new merged splat s_m has center and normal defined as [6]:

$$c_m = \frac{\sum_i |s_i| \cdot c_i}{\sum_i |s_i|} \tag{1}$$

$$n_m = \frac{\sum_i |s_i| \cdot n_i}{\sum_i |s_i|}, \tag{2}$$

where $|s_i|$ stands for the area of splat s_i. To compute the extent of s_m, we sample n (8 is enough) points p_j^i on the boundary of each splat s_i and project them to the splat plane of s_m. Then we apply principal component analysis (PCA) to the set of points p_j^i to find the main axis directions and proper scaling.

4. Error Measures

A good error measure should be able to distinguish smooth regions, that can be well handled by a single elliptical disk, from detailed or boundary regions. To this end, we use two error measures: a *perpendicular error* [1] and a *tangential error* . The *perpendicular error* e_p is the minimum distance between two planes orthogonal to the node's normal, which define a region enclosing all children in the node's descendent leaves. The computation of e_p is illustrated in Figure 2, and can be expressed as:

$$\begin{aligned} e_p &= \max\{a_i + d_i\} - \min\{a_i - d_i\} \qquad (3) \\ \text{with} \quad & d_i = r_i\sqrt{1-(n_i \cdot n)^2} \\ & a_i = (c_i - p)\cdot n, \end{aligned}$$

where r_i is half the length of the major axis of s_i.

For the *tangential error*, s_m and its respective children are rendered offscreen in a small viewport. s_m is first rendered in blue, while its children are all rendered in red. The *tangential error* e_t (Figure 3) is proportional to the ratio between the area in red and the total painted area. Specifically, it is given by:

$$e_t = r_m\sqrt{\frac{n_{blue}}{n_{blue}+n_{red}}}, \qquad (4)$$

where n_{blue} and n_{red}, represent the number of pixels with the respective colors. If none of these two errors exceeds a predefined threshold ε, the s_m is rendered following the approach of [1]. Otherwise the tree of its node decendants must be explored .

5 Conclusion and Future Work

We present a multiresolution hierarchy for point samples based on a space partitioning data structure, i.e., the point-octree. Elliptical splats are created in each internal node

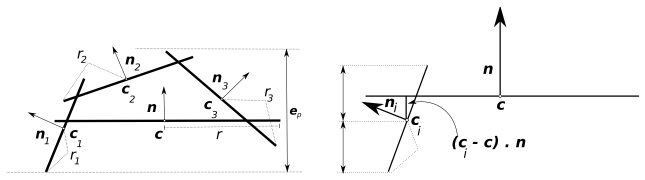

Figure 2. Perpendicular error

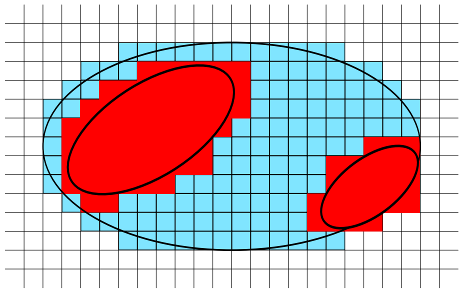

Figure 3. Tangential error

by analyzing the local surface properties. The two error metrics are used to define which splats should be projected for a given viewpoint, taking into account distance to the eye as well as surface smoothness, i.e, finer resolution splats along the silhouettes and detailed regions.

As future work, a more sophisticated partitioning refinement shall be investigated, such as curvature driven methods. This will allow for a more coherent clustering based on surface properties, instead of spatial distribution. Furthermore, the hierarchical data structure should only be used to create the coarser resolution splats, while an array-like structure that maps well to the GPU, should be used for efficient rendering. In addition, the perpendicular error can be improved for elliptical splats, since it was derived for circular splats [1, 4, 5]. In this case, the error is overestimated using the larger ellipse's axis.

References

[1] C. Dachsbacher, C. Vogelgsang, and M. Stamminger. Sequential point trees. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 657–662, New York, NY, USA, 2003. ACM.

[2] L. Kobbelt and M. Botsch. A survey of point-based techniques in computer graphics. *Computers Graphics*, 28(6):801–814, Dec. 2004.

[3] D. Luebke, B. Watson, J. D. Cohen, M. Reddy, and A. Varshney. *Level of Detail for 3D Graphics*. Elsevier Science Inc., New York, NY, USA, 2002.

[4] R. Pajarola. Efficient level-of-details for point based rendering. In *Proceedings IASTED Invernational Conference on Computer Graphics and Imaging*, Calgary, AB, Canada,, 2003. ACTA Press.

[5] S. Rusinkiewicz and M. Levoy. Qsplat: a multiresolution point rendering system for large meshes. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 343–352, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[6] J. Wu, Z. Zhang, and L. Kobbelt. Progressive splatting. *Symposium on Point-Based Graphics*, 0:25–142, 2005.