Foveated 3D Model Simplification

Irene Cheng
Department of Computing Science, Univ. of Alberta, Edmonton, CANADA
lin@cs.ualberta.ca

Abstract

Visualization of 3D images is becoming more commonplace for a variety of applications including online games and e-commerce. For efficient online visualization of 3D objects it is necessary to quickly adapt 3D models (including the wireframe and texture) to the available computational or network resources. In this paper we propose enhancements to 3D model simplification based on interactive Level-Of-Detail (LOD) update with foveation. Our technique differs from the others in that simplification is based on both surface curvature surface and the region of interest. Data structures to implement model simplification are also described. Experimental results comparing the simplified models and the computational time demonstrate the feasibility of our approach.

1. Introduction

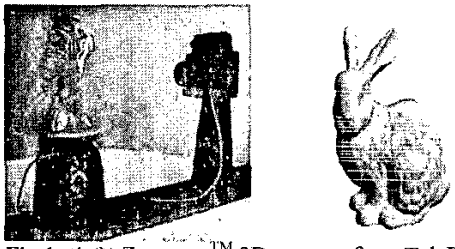

Fig.1: (left) Zoomage™ 3D scanner from TelePhotogenics used to capture 3D data. (right) Stanford bunny.

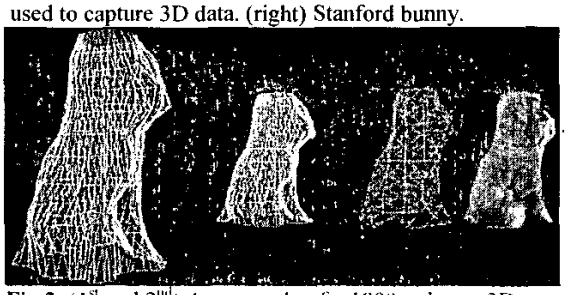

Fig.2: (1st and 2nd) An example of a 1800 polygon 3D object at different distances. (3rd) Wireframe of the same object using 180 polygons. (4th) Texture mapped on 180 polygons.

3D visualization is an expanding area of multimedia research covering graphics, imaging and network transmission. With advances in laser scanning and digital imaging it is now possible to scan objects with super high resolution texture (surface image) and depth at various surface locations (connected into a wireframe). For example, the Zoomage 3D scanner (Fig. 1, left) can produce texture of 200 mega pixels and wireframes with over 2 million triangles; or the Stanford bunny (Fig. 1, right) commonly used as a test object has 69,000 triangles. The trend in multimedia applications is to use more and more polygons in order to produce photo-realistic 3D scenes. However, a large number of polygons impose another challenge to researchers in terms of storage, processing, rendering and transmission. For example in Fig.2, when the mesh is closer to the viewpoint, more polygons can show better detail, but when the object is farther away, keeping the same number of polygons is not necessary. In Fig.2, only 180 polygons are rendered without loosing the perceptual quality of the object. Rendering time is also saved, by reducing the number of polygons from 1800 to 180. Simplification is also useful for online visualization of 3D objects; for example the bandwidth between a server and a client (viewing workstation) can be accurately monitored [13] and the quality of a model and associated texture can be adjusted to allow the best possible visualization within a given time interval. Level-of-detail (LOD) [3,4,6] is a 3D-visualization topic dealing with efficient polygon meshing and texture mapping based on viewpoint. Since human perception is less sensitive to details on an object when it moves further away and gets smaller, it is inefficient to render the same number of polygons as when the object is, say, a few feet away. Therefore the objective of LOD is to represent areas of low perceptual importance with a few large triangles, and represent areas of high perceptual importance with many small triangles to preserve smoother curve surfaces.

Given a sample of range data generated from laser scanning, a simple way to produce different levels of detail is to take subsets from the sample. Decrease of sample size means decrease in detail. This method is simple but does not represent high-density areas with more scan points when merging polygons. To overcome this problem, we need to apply *different simplification strategies* to regions of different densities. A curvature equalization method is described by Scarlatos *et al.* [10]. They tried to balance the curvature of the input data within each triangle by adjusting the triangulation of the original surface. This work was developed for shape fitting, whereas we consider adaptive 3D multimedia transmission. Kalvin *et al.* used a simple patch decimation method in their efforts to create surface models from medical data [7], after an initial polygonal surface is created to approximate the input data; adjacent coplanar polygons are merged to simplify the model. Since only precisely coplanar faces are merged, the degree of

0-7803-7946-2/03/$17.00 ©2003 IEEE

simplification is largely dependent on the curvature of the object, and thus only limited simplification is obtained. Hinker *et al.* extended the patch decimation method to merge the nearly coplanar polygons [5]. If the angle between the normal vectors of two adjacent triangles is below a given bound error, the two triangles are merged. Finally, the merged polygons are re-triangulated with a simple and robust method. However, this method is ineffective for surfaces with high curvature.

Extensive psychovisual studies have shown that animate vision has much higher resolution around the center of the visual field (fovea) compared to the periphery [9,11]. The advantage of foveation in interactive online 3D visualization include: (i) The ability to provide reasonable quality over a network with low and dynamically varying bandwidth, such as the Internet; (ii) Fast update of a 3D model based on viewer interaction; (iii) Compact representation of the 3D data transmitted reducing bandwidth utilization. Fovea was not considered in [3,4,6]. In [2], fovea was a parameter of the contrast sensitivity function. However, no mathematical model was given to analyze its effect on perceptually driven simplification. The focus of this report is on simplifying a model using a user specified point of interest (fovea). We generalize an LOD approach using curvature to simplify based on foveation. In past research [1,8] viewing image and video with foveation, and foveated texture mapping was discussed. However, creating foveated LOD representations was not considered.

The remainder of this report is organized as follows: Extending LOD based simplification with foveation is discussed in detail in Section 2. Section 3 describes some of the data structure used in the implementation. Section 4 shows results from a preliminary version of the implementation of the proposed approach. Future work and conclusion are outlined in Section 5.

2. Foveated LOD simplification using curvature

A simple approach to model simplification is based on curvature variations between nearby surface patches. For example, simplification can be decided based on the angle between the normals of adjacent triangles. The deviation is defined by a value cos θ, $0 \leq \cos\theta \leq 1$, which is computed by taking the dot product of the two normalized vectors v_1 and v_2. When face1 and face2 in Figure 3 are nearly parallel, cos θ is close to 1. In other words, a vertex with a high cos θ value can be removed while one with a low cos θ value should be maintained in order to preserve details. Whether or not a vertex should be removed is controlled by a threshold T. A vertex is removed if its associated cos θ value is > T. An approach for simplification based on the distance between an object and a viewer can be described as follows.

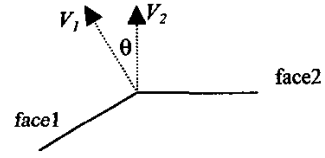

Fig.3: The normals (dotted lines) of two adjacent triangles.

Suppose $Dist_{vc}$ is the distance between the viewpoint and the object center, the value T is defined by:

$T = (1 - Dist_{vc} / Dist_{max})$

where $Dist_{max}$ is the maximum distance possible between the viewpoint and the object center. As the object moves away from the viewpoint, T becomes smaller. More vertices will be removed from the mesh which will contain lesser triangles. This strategy is based on the observation that human perception is less sensitive to detail at a farther distance.

The above approach, and others in LOD simplification, is based on the assumptions that: (i) all parts of an object at a given distance are equally important to a viewer, and (ii) there are no bandwidth restrictions limiting the amount of data that can be transmitted in a reasonable amount of time. We next develop a modified method that addresses these limitations.

Foveated Simplification

In this method we allow a viewer (client site over a network) to select a point of interest (fovea) on an object. The fovea is represented by (F_x, F_y). The threshold T used for simplification of two adjacent patches varies as a function of the distance between the fovea and the mid-point (X, Y) of the patches being simplified. The distance is computed as the sum of two parts: (a) The vertical distance between Fy and Y, normalized with respect to the height (H) of an object; and (b) the minimum horizontal angle between X and Fx, normalized by π the maximum possible angle. Figure 4 illustrates this concept. To keep the distance function more general we can apply different weights (w_v and w_h) to the two distance components. We compute the distance in this manner because the usual concept of Euclidean distance is not meaningful for the surface of a 3D object; for example, two points on opposite sides of a thin object may have a very small Euclidean distance but may be quite far going along the surface of the object. Finally, the threshold T for simplification can be computed as an exponential function of the 3D surface distance. Formally, the simplification process can be summarized in the following steps:

Step 1: Record fovea location (F_x, F_y) based on viewer input. Represent fovea as (F_α, F_y), i.e., represent the x-axis as an angle with respect to a central vertical axis.
[Note that this representation can be automatically generated if an object is scanned as in Figure 1, left.]

Step 2: For all pairs of adjacent planar patches (faces) on the object compute an intermediate point (X,Y). Represent (X, Y) as (α,Y) as in Step 1. Compute distance to fovea as:

$$D = w_h\left(\frac{\beta}{\pi}\right) + w_v\left(\frac{|Y - F_y|}{H}\right)$$

where β is minimum horizontal angle between the directions α and F_α. The threshold can then be computed as:

$$T = e^{-KD}$$

where K is a constant determining the rate at which the threshold decreases as we move away from the fovea.

Merge two faces if $\cos\theta > T$, where θ is as described in Figure 3.

[Note that the distance function defined here is intuitive for the data shown in Figure 5, generated by a scanning system like the one in Figure 2, since depth is computed for fixed angular rotations (α) and at a fixed number of vertical positions (Y).]

Step 3: Repeat Step 2 until total number of polygons meet the desired limit.

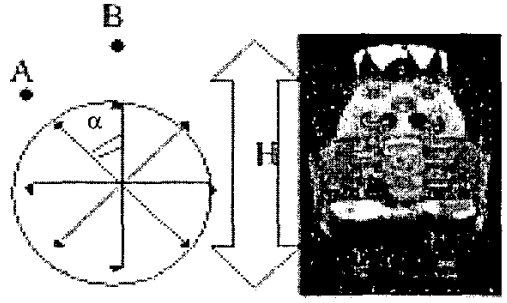

Fig. 4: Angular angle α between two points A and B (left), and height H of an object (right).

3. Data Structure

A 3D mesh is constructed by triangulation using a set of sample points in three-dimensional space (Fig.5, left). Sample points are generated from the Zoomage3D laser scanner. The data structure used in this project is designed for storing the original high resolution scan points, from which lower resolution versions can be sub-sampled.
The data structure is composed of three main components: the grid, the mesh and the strategy (or function) list.

The Grid *G*

The scan points from the original high-resolution sample are stored in an x by y grid representation, where x is the number of scan points horizontally and y is the number of scan lines vertically. Re-meshing from the original sample set is therefore possible.

Each grid element G_{ij} contains the following data members:

- *gridIndex* identifies an element G_{ij} in the grid.
- *pixel* defines a 3d-vertex associated with a gridIndex.
- *texel* defines the 2d coordinates in the texture map corresponding to the vertex.
- *simplify* has default value of false until $\cos\theta > T$ as explained in step 2. A true value will trigger a vertex removal process.
- *removed* has a default value of false unless the vertex is removed during simplification.
- *LOD* denotes the level of simplification. A flat surface is assigned LOD 0, which means more triangles merged. It is used as an index to reference a particular simplification function in the strategy list.

The Mesh *M*

While the content of the G structure is static, the range data stored in the M structure is dynamic. The mesh structure M contains a list of triangles. Instead of storing the actual coordinates, each triangle is defined by three gridIndices. These triangles are arranged in strips. Stripification [12] is a technique used to avoid duplications by minimizing the number of vertices transmitted. The M structure is dynamically changed in each simplification. It stores the modified version of the mesh computed based on $\cos\theta$ and T. The structure M references the G structure through gridIndices.

Fig. 5: A sample of 3D points (left), texture (right).

4. Experimental Results

Java3D is used in the current implementation. The experiments were performed on a Pentium III laptop running Windows Millennium. The dog object shown has a height of 6 inches.
Figure 6 shows the user interface used to test the foveated simplification method. The interface allows various parameters in the model to be modified. Figure 7 shows simplification with the fovea located on the back of the

head of the dog. Observe that the triangles get larger as we look towards the tail of the dog. Figure 8 shows the same simplification example as in Figure 7, but the front view of the dog. Note that the front part of the head is significantly simplified. However, the region around the legs is not simplified as much. This happens because of the large curvature variations around the legs preventing simplification from occurring.

Fig. 6: User interface used to test simplification.

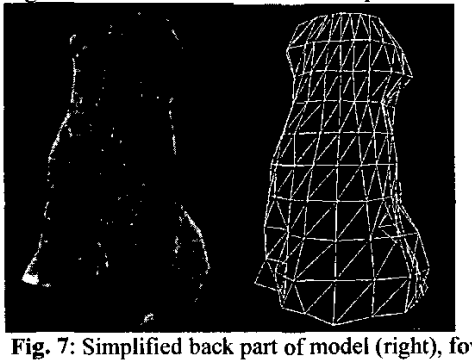

Fig. 7: Simplified back part of model (right), fovea on back of head.

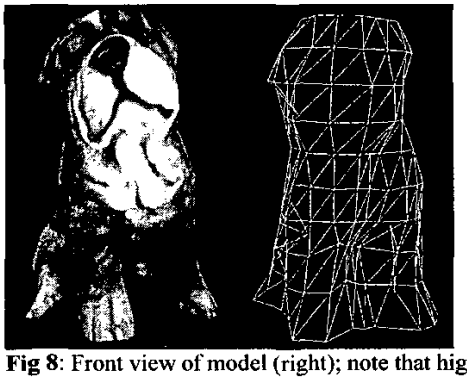

Fig 8: Front view of model (right); note that high curvature variations around legs of dog result in less simplification even though location is far from fovea.

5. Conclusion and Future Work

In this paper we proposed an approach for enhancing 3D model simplification algorithms for bandwidth limited online applications with interactive foveated simplification. The work discussed a single strategy for curvature based simplification with a single fovea implementation. In future, multi foveae integration and multi strategies will be considered.

The implementation described does not incorporate foveated texture transmission. We intend to add texture transmission with foveation, following the approaches taken in our past research [1], in the near future.

We also plan to integrate the simplification software in our past research on optimal bandwidth monitoring and develop strategies for efficient implementation over wireless networks.

6. Acknowledgements

The author would like to thank A. Basu for his helpful suggestions.

7. References

[1] I. Cheng and A. Basu, "Efficient visualization of super high resolution 3D images," IEEE 3D PVT Conference Proceedings, Italy, 2002.

[2] D. Luebke and B. Hallen, "Perceptually Driven Simplification for Interactive Rendering," Rendering Techniques, Ed. S. Gortler and K. Myszkowski, Springer-Verlag, London 2001.

[3] J. Cohen et al., "Simplification Envelopes" Proceedings ACM SIGGRAPH, 1996, L.A. pp. 119-128.

[4] J.C.Xia et al., "Adaptive Real-Time Level-of-detail-based Rendering for Polygonal Models" IEEE Trans. on Visualization and Computer Graphics, June 1997.

[5] P. Hinker and C. Hansen. Geometric optimization. In *Proc. Visualization '93*, pages 189–195, 1993.

[6] H.Hoppe, "Progressive meshes" Proceedings of SIGGRAPH 1996, L.A. pp. 99-108.

[7] [Kalv91] A.D. Kalvin, C.B. Cutting, B. Haddad, and M.E. Noz. Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures. *SPIE Image Processing*, 247–259, 1991.

[8] T.H. Reeves and J.A. Robinson. Adaptive Foveation of MPEG video. *ACM Multimedia Conference*,231-241, 1996.

[9] G. Sandini and M. Tistarelli. Vision and space-variant sensing. In *ECCV-94 Workshop*, 398-425, 1994.

[10] L. L. Scarlatos and T. Pavlidis, "Optimizing triangulations by curvature equalization," In *Proc. Visualization'92*, pp. 333–339. 1992.

[11] E.L. Schwartz. Computational anatomy and functional architecture of striate cortex. Vision Research, 20:645-669, 1980.

[12] A.J.Stewart, "Tunneling for Triangle Strips in Continuous Level-of-Detail Meshes," Proceedings of Graphics Interface, pp. 91-100, 2001.

[13] Y. Yu, I. Cheng and A. Basu, "Optimal adaptive bandwidth monitoring," IEEE Trans. on Multimedia, to appear, (short version in Proceedings of IEEE ISCAS, Scottsdale, USA, 2002).