3D OBJECT DATABASE SIMPLIFICATION USING A VERTEX CLUSTERING ALGORITHM*

Tolga Bayik, Mehmet B. Akhan

University of Hertfordshire
Faculty of Engineering and Information Sciences
Department of Electrical and Electronic Engineering
Hatfield, Herts AL10 9AB
United Kingdom
t.bayik@herts.ac.uk, m.b.akhan@herts.ac.uk

ABSTRACT

3D graphics rendering is computationally very intensive particularly for interactive applications. The complexity of the 3D object databases becomes a major problem in a rendering pipeline as the number of polygons is increased with model complexity. Simplifying large object databases and eliminating some of the polygons in the database with minimal impact on visual quality is desirable for faster rendering. This paper discusses the necessity for simplifying 3D object databases in accelerating 3D graphics and introduces a new method based on vertex clustering technique.

Keywords: 3D Graphics, Rendering, Levels of Details, Simplification

INTRODUCTION

During the 3D rendering process, not all the details need to be rendered on the screen. If the object is very small or distant to the observer, the object will be projected on a small portion of the screen. In these cases, the object could be modelled with fewer polygons, as the details will be less important to the viewer. If some of the polygons can be eliminated just before the rendering process according to a criterion with minimal impact on visual quality, this would save time for the rendering process. Especially interactive applications and real-time rendering requires fast rendering and reducing the number of input polygons has a drastic effect on the rendering time.

One of the most popular methods proposed for optimising the rendering of small and distant objects is multiresolution modelling. Depending on the screen size of a given object or cluster of objects, the appropriate level of detail within the model would be chosen [Clar76]. Therefore, distant objects

* This work is supported by Texas Instruments Europe

would be rendered using a model that contains fewer polygons and nearby objects would be rendered using a model containing more polygons. The principal challenge is to automatically convert a detailed model into a simpler model by minimising the number of polygons without substantially reducing the image quality.

Erikson classifies the polygonal simplification algorithms into three as adaptive subdivision, geometry removal and sampling [Erik96].

Adaptive subdivision algorithms start with a very simple model. More details are added, recursively subdividing the model until the model approximates the original one to a certain degree. The drawback of the subdivision algorithms is that they may result in gaps after subdividing the neighbouring polygons a different number of times. The algorithm should take care of these cracks and fill them if necessary. DeHaemer and Zyda present an adaptive subdivision algorithm that fits polygons to a set of samples [Deha91]

Geometry removal algorithms remove faces and/or vertices from the model repeatedly until the model cannot be simplified any more or a certain user-specified degree of approximation is met. Hinker and Hansen used a geometry removal algorithm by replacing polygons with almost parallel normal vectors with a single polygon [Hink93].

Sampling algorithms aim to reduce the vertex count using a sampling mechanism. Turk used a combination of sampling and geometry removal ideas that is effective for models that represent curved surfaces rather than models containing sharp discontinuities [Turk92]. He uses point repulsion to achieve simplification and good coverage of the surface. The algorithm starts with randomly distributing a user specified number of vertices over the model and simulating repulsion forces between the vertices. Two vertices repel each other if they are too close to each other. Using the repulsion forces, he evenly distributes the vertex points. Then these points are triangulated and the original vertices are removed to produce the simplified version.

Our main initiative was decimation methods and vertex clustering techniques. A review of these techniques will be presented in more detail as we have used parts from each in our algorithm.

DECIMATION METHODS

Another classification according to Heckbert involves decimation methods [Heck97]. He further classifies the decimation methods into following categories:

- Vertex Decimation
- Edge Decimation
- Triangle Decimation
- Patch Decimation

Vertex decimation methods basically delete a vertex and retriangulate the deleted vertex' neighbourhood.

Edge decimation methods merge two vertices after deleting one edge and two triangles.

Triangle decimation methods delete one triangle and three edges. Then the three vertices are merged and the neighbourhood is retriangulated.

Patch decimation methods delete more than one adjacent triangle. The boundary is then retriangulated.

Schroeder, Zarge, and Lorensen [Schr92] propose a vertex decimation algorithm where the unimportant points are iteratively removed from the mesh. Local retriangulations are performed to preserve the surface. Their algorithm assumes a triangulated surface and characterises the local vertex geometry and topology using the information from adjacent faces. If removing a vertex preserves local topology, then the removing operation is considered. The algorithm makes multiple passes over the data until the desired error is achieved or there is no more simplification available. The error at a vertex is defined as the distance from the point to the average plane of the adjacent vertices. On each pass, if the distance between the vertex and its average plane is less than a certain threshold, then the vertex is deleted. Then the resulting hole is retriangulated. As the errors are measured with respect to the previous approximation, not relative to the input points, errors can accumulate.

Soucy and Laurendeau also used a vertex decimation algorithm [Souc92, Souc96]. The algorithm makes multiple passes over the data until the error exceeds a specified tolerance or the desired size of model is achieved. On each pass, the vertex with least error is deleted, and its adjacent triangles are retriangulated. The error of a vertex is a measure of the error that would result from its removal.

Guéziec developed a method for simplification using edge decimation [Guez95]. His algorithm basically deals with the problem of finding a suboptimal approximation with a given error bound. To delete an edge and merge its two endpoints into a single vertex, Guéziec defines the edge collapse, or edge contraction operators. If the following conditions are met, then a collapse is allowed:

- the topology of the surface is preserved,
- the normals of the modified faces change little,
- the new triangles are well shaped,
- the error radius for the new vertex is below an error threshold

Gourdon developed a method for simplification that differs from almost all other simplification algorithms in that it does not assume the surface mesh to be a triangular one [Gour95]. He removes edges iteratively based on a curvature criterion, preserving the Euler characteristic of the model, hence preserving the topology. The Euler characteristic of a model is defined as: $\xi=F-E+V$ where F is the number of faces, E is the number of edges and V is the number of vertices.

Hamann developed a triangle decimation method that deletes the triangles in increasing order of weight, which is based on local curvature in the model [Hama94]. As a result, low curvature triangles are deleted first. The user can specify a percentage of triangles to be deleted that allows the user to create several levels of detail of the model.

Kalvin and Taylor developed a patch decimation method called "superfaces" [Kalv94,Kalv96]. Their adaptive subdivision method is quite similar to Hinker-Hansen [Hink93] as it finds patches of triangles with nearly parallel normal vectors, and then retriangulates each patch. Instead of defining patches based on angles between normal vectors as Hinker-Hansen did, Kalvin-Taylor define the patches based on distance-to-plane, which is less sensitive to noise. Figure 1 shows triangles with nearly parallel normal vectors.

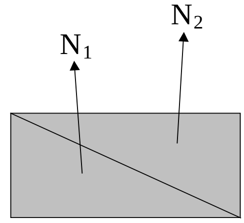

Figure 1 TwoTriangles with nearly parallel normal vectors

VERTEX CLUSTERING

Vertex clustering algorithms are generally a subset of decimation methods. After clustering the vertices, a representative vertex is chosen to reduce the number of vertices, hence simplifying the object.

Rossignac and Borrel have used the idea of vertex clustering for the simplification of general models with arbitrary topology [Ross97]. Given a polygonal model that has been triangulated, they subdivide its bounding volume into a grid of boxes of user-specified size. They first assign weights to all vertices according to their visual importance. Two factors are used for this purpose. The first factor depends on the assumption that the vertices that have a higher probability of lying on the object's silhouettes are important. This factor is estimated by the inverse of the maximum angle between any pair of edges adjacent to the vertex. The second factor favours the vertices belonging to large faces. The vertices that belong to large polygon faces should not be affected by the removal of small details. This factor is estimated using the face area.

After assigning weights to all vertices, they then triangulate each face and group the vertices together into clusters based on geometric proximity. A box containing the object is uniformly divided into cells and all vertices within each box are merged together into a new representative vertex, being the centroid of the vertices within each box. A simplified model is then synthesised from these representative vertices.

The effects of uniform vertex clustering are shown in Figure 2. In grid 4b, the rectangle has collapsed to a single point. Separate components have been joined together by bridging a gap in grids 3a and 4a. Also note that a thin area is collapsed into a single line segment in grids 2c-1d.

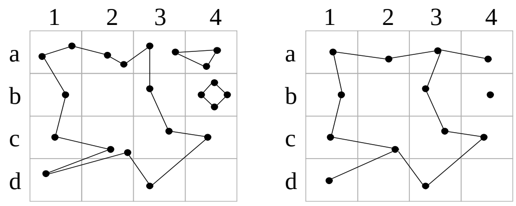

Figure 2 Vertex Clustering

This algorithm is quite general as it accepts polygonal models, but averaging the vertices removes high frequency details that might have significant importance [Heck97]. Instead of using a single average per box, using more representative vertices could be one of the solutions to this problem and investigated in this paper.

Low and Tan used floating-cell clustering instead of uniform-subdivision clustering to preserve the important features [Lowk97]. They use the idea of point repulsion to achieve simplification as Turk has suggested [Turk92]. The highest weight vertex in a cell attracts vertices in its vicinity from immediately adjacent cells. Also, instead of using the inverse of the maximum angle between all pairs of incident edges on the candidate vertex as Rossignac and Borrel did [Ross97], they used the cosine of the half of the angle. Their clustering algorithm suggested replacing the uniform grid with a set of cluster cells that could be any simple shape, such as cubes or spheres.

In general, vertex clustering algorithms are fast and general. Vertex clustering has the advantage that it can deal with any arbitrary collection of polygons. However, clustering tends to make overly aggressive radical changes to the surface, as it does not take into account the geometry, only statistics on the distribution of vertices. One solution suggested in this paper is to start with a smaller grid and increase the grid size incrementally until the desired level of simplification is achieved. This way, radical changes will not be permitted

ITERATIVE VERTEX CLUSTERING WITH MULTIPLE CENTROIDS

The main contribution of this paper concentrates on two problems encountered during uniform clustering:

- Only one representative vertex is chosen no matter how many vertices there are in a certain box.
- Clustering is done in a single step which would make overly aggressive changes.

The iterative vertex clustering with multiple centroids algorithm begins with subdividing the boundary volume of the object into a grid of boxes as shown in Figure 3. The size of the boxes could be changed and should be flexible to control the level of detail required.

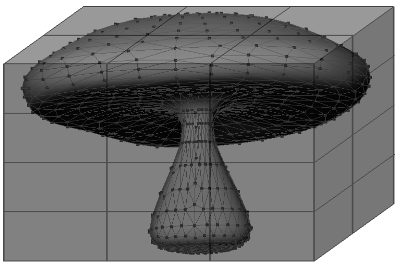

Figure 3 Subdividing the boundary volume of the object into grid of boxes.

Occupancy is defined as the number of vertices in a single box. If the occupancy is high, it is decided that more representative vertices for that box are needed. If the occupancy is low, then fewer representative vertices for that box are needed. So, rather than assigning a single value to each box, several representative vertices are chosen for a box depending on occupancy. The K-means algorithm is used for this clustering and quantisation procedure.

The K-means algorithm works as follows [Lind80, Bayi95]:

Suppose that we have M training vectors denoted by f_i for $1 \leq i \leq M$. The reconstruction levels r_i are determined by minimising the average distortion D

$$D = \frac{1}{\mathrm{M}} \sum_{i=1}^{i=M} d(f_i, f_i^*)$$

where

$$d(f_i, f_i^*) = (f^* - f)^T (f^* - f).$$

In the K-means algorithm, we begin with an initial estimate of r_I for $1 \leq i \leq L$. Then, we classify the M training vectors into L different groups or clusters corresponding to each reconstruction level. This can be done by comparing a training vector with each of the reconstruction levels and choosing the level that results in the smallest distortion. Note that we quantise only the given training vectors, not all possible vectors of f. A new reconstruction level is determined from the vectors in each cluster. Suppose that f_i for $1 \leq i \leq M_i$ are M_1 training vectors quantised to the first reconstruction level r_1. The new estimate of r_1 is obtained by minimising

$$\frac{1}{\mathrm{M}_1} \sum_{i=1}^{i=M_1} d(f_i, r_1)$$

If the distortion function $d(f,f^*)$ used is the squared error $(f^*-f)^T(f^*-f)$, then the new estimate of r_1 is just the average of M_1 vectors f_i, $1 \leq i \leq M_1$. A new estimate of all other reconstruction levels r_i for $2 \leq i \leq L$ is similarly obtained. This is the end of first iteration of the procedure and can be stopped when the average distortion D does not change significantly between two consecutive iterations.

This algorithm is computationally complex but converges to a local minimum of D. It is successfully used in image processing applications.

As initial estimates, equally spaced values on the object diagonal are taken. The number of representative vertices is chosen as a user defined percentage of the number of the vertices in the box. The output of the K-means algorithm is the cluster centres. Then, the degenerate polygons, redundant edges and vertices are eliminated and the model is retriangulated using the vertices left out at the end of simplification procedure to synthesise the simplified model.

The whole procedure can be repeated several times to produce better results. Instead of a single pass, one can begin with the smaller sized boxes. The output could then be simplified again with larger sized boxes. The procedure terminates when the output does not change significantly any more or a certain level of detail goal is achieved. The output can be compared with the original model and an error term can be calculated. The advantage of using adaptively larger sized boxes relies on the fact that clustering makes rather radical changes to the object's geometry. Smaller sized boxes will reduce the effect of sudden changes.

RESULTS AND DISCUSSION

The suggested algorithm divides up the object space into 3D grids and performs simplification for each box using the K-means algorithm. If the aimed simplification is not achieved, the number of boxes is decreased until the goal is achieved iteratively. Starting with more boxes and using the output of the simplification to further reduce the number of vertices does not allow the object to change radically. Figure 4 shows the original image rendered, showing the input vertex points as well. Figure 5 and Figure 6 show different level of details of the original object by using the iterative vertex clustering algorithm. Similarly, Figure 7 shows another object rendered that contained more polygons. Figure 8 and Figure 9 show different level of details of the original object using the same technique.

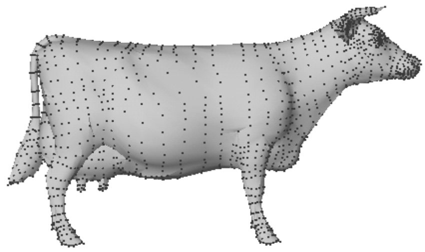

Figure 4 Original Cow image 2903 vertices, 5800 Polygons

Figure 5 Simplified version - 1221 vertices, 2504 Polygons

Figure 6 Simplified version - 542 vertices, 1098 Polygons

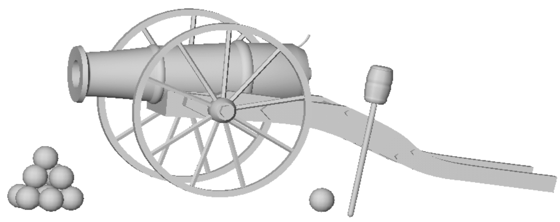

Figure 7 Original Canon image 12261 vertices, 24364 Polygons

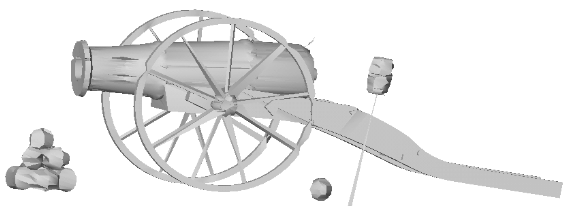

Figure 8 Simplified version - 2797 vertices, 6091 Polygons

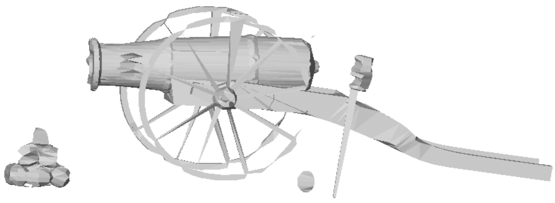

Figure 9 Simplified version - 1707 vertices, 3760 Polygons

Vertex clustering algorithms do not usually produce optimal simplifications as they group the vertices that are geometrically close, not necessarily topologically close. The algorithm could be modified to produce better results by taking into account the geometry of the object as well as its statistical distribution.

ACKNOWLEDGEMENTS

The authors would like to thank University of Hertfordshire Electronics and Electrical Engineering Department and Texas Instruments Europe for their support of this work. Special thanks go to Robert Owen, Texas Instruments, Northampton for his continuous support and encouragement. The authors would also like to express their appreciation to David Lauder for his valuable suggestions, helpful comments and proofreading.

REFERENCES

[Bayi95] Bayik, T., Color Morphological Filtering and its Applications to the Automatization of Jacquard Looms, M.S. thesis, Bogazici University, Istanbul, Turkey 1995

[Clar76] Clark, James H., Hierarchical geometric models for visible surface algorithms. *CACM*, 19(10):547-554, Oct. 1976

[Deha91] DeHaemer, M. , Zyda M. J. Simplification of objects rendered by polygonal approximations. *Computers and Graphics*, Vol. 15, no:2 pp.175-184, 1991.

[Erik96] Erikson, C. Polygonal Simplification: An Overview. *Technical Report TR96-016, 1996, Department of Computer Science, UNC – Chapel Hill, NC 27599-3175.*

[Gour95] Gourdon, A. Simplification of irregular surface meshes in 3D medical images. In *Computer Vision, Virtual Reality, and Robotics in Medicine (CVRMed '95)*, pp. 413–419, Apr. 1995

[Guez95] Guéziec, A.. Surface simplification with variable tolerance. *Second Annual Intl. Symp. on Medica Robotics and Computer Assisted Surgery (MRCAS '95)*, pp. 132–139, November 1995.

[Hama94] Hamann, B. A data reduction scheme for triangulated surfaces. *Computer-Aided Geometric Design*, 11:197–214, 1994.

[Heck97] Heckbert, P. and Garland, M. Algorithms for Surface Simplification. *Multiresolution Surface Modeling Course SIGGRAPH '97, Los Angeles, California, August 199.*

[Hink93] Hinker, P. and Hansen, C. Geometric optimization. *Proc. Visualization '93*, pages 189–195, San Jose, CA, October 1993.

[Kalv91] Kalvin, A.D., Cutting, C.B., Haddad, B. and Noz, M. E. Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures. *Medical Imaging V: Image Processing*, volume 1445, pp. 247–258. SPIE, Feb. 1991

[Kalv94] Kalvin, A.D. and Taylor, R. H. Superfaces: Polyhedral approximation with bounded error. *Medical Imaging: Image Capture, Formatting, and Display*, vol. 2164, pp. 2–13. SPIE, Feb. 1994

[Kalv96] Kalvin, A.D. and Taylor, R. H. Superfaces: polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Applications*, 16(3), May 1996. http://www.computer.org/pubs/cg&a/articles/g30064.pdf

[Lind80] Linde, Y., Buzo, A. and Gray, A. M. An Algorithm For Vector Quantizer Design. *IEEE Transactions on Communications, vol. COM-28, pp. 84-95 Jan. 1980*

[Lowk97] Low, K. and Tan, T. Model Simplification Using Vertex Clustering. *1997 Symposium on Interactive 3D Graphics*, Providence, RI, USA, pp. 75-81. http://www.iscs.nus.sg/~tants/.

[Ross97] Rossignac, J. Geometric Simplification and Compression. *Multiresolution Surface Modeling Course SIGGRAPH '97, Los Angeles, California, August 1997*

[Schr92] Schroeder, W. J., Zarge, J. A. and Lorensen, W. E. Decimation of triangle meshes. *Computer Graphics (SIGGRAPH '92 Proc.)*, 26(2):65–70, July 1992. Also appeared in *Multiresolution Surface Modeling Course SIGGRAPH '97, Los Angeles, California, August 1997*

[Souc92] Soucy, M. and Laurendeau, D. Multiresolution surface modeling from multiple range views. *Conf. On Computer Vision and Pattern Recognition (CVPR '92)*, pp. 348–353, June 1992.

[Souc96] Soucy, M. and Laurendeau, D. Multiresolution surface modeling based on hierarchical triangulation. *Computer Vision and Image Understanding*, 63(1):1–14, 1996.

[Turk92] Turk, G. Re-tiling polygonal surfaces. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):55-64, July 1992.