Int. J. Pure Appl. Sci. Technol., 13(2) (2012), pp. 26-31

International Journal of Pure and Applied Sciences and Technology

ISSN 2229 - 6107

Available online at ***www.ijopaasat.in***

Research Paper

Straight Polygon Simplification of 3D Graphical Models

Md. Safiuddin Sheikh[1] and Md. Haider Ali[2, *]

[1]Assistant Professor, Mathematics, DSHE, Dhaka, Ph. D. Fellow, National University, Gazipur-1704, Bangladesh

[2]Professor, Dept. of Computer Science & Engineering, University of Dhaka, Dhaka-1000, Bangladesh

* Corresponding author, e-mail: (safiuddin.du@yahoo.com)

(Received: 23-9-12; Accepted: 18-10-12)

Abstract: *In this paper we have proposed a specific method of simplification of three dimensional (3D) graphical models (polygonal meshes). Many applications in computer graphics require highly detailed complex models. However, the level of detail actually necessary may vary considerably. To control processing time, it is often desirable to use approximations in place of excessively detailed models. We have developed a simplification algorithm which can produce quality approximations of polygonal meshes. The algorithm uses iterative contractions of vertex pairs to simplify models. Our algorithm can produce good approximations of polygonal surface models which is one of the fastest algorithms available for producing quality approximations and capable of simplifying graphical models of reasonably high complexity. Besides these we have found a new scope of preserving 3D shape at boundary area while simplifying the polygonal meshes.*

Keywords: Polygonal mesh, Levels of details (LOD), Polygon mesh simplification, Pair contraction.

Introduction

Many computer graphics applications require complex, highly detailed models to maintain a convincing level of realism. Consequently, models are often created or acquired at a very high resolution to accommodate this need for detail. However, the full complexity of such models is not always required, and since the computational cost of using a model is directly related to its complexity. So it is more useful to have simpler versions of complex models. We would like to produce automatically these simplified models. Simplification is the basis for the construction of level of detail (LOD) representations. Adaptive Mesh Simplification (AMS) simplifies the polygonal

surface with minimum change in topology. The most common use of polygonal simplification is to generate LODs of the objects in a scene, by representing distant objects with a lower LOD and nearby objects with a higher LOD. AMS of the other hand only eliminate those polygons which will not hamper the originality of the object shape. Recent work on surface simplification algorithms has focused on this goal. As with most other works in this area, we will focus on the simplification of polygonal models. We will assume that the model consists of triangles only. This implies no loss of generality, since every polygon in the original model can be triangulated as part of a preprocessing phase. We have developed an algorithm which produces simplified versions of such polygonal models. Our algorithm is based on the iterative contraction of vertex pairs. The primary advantages of our algorithm are:

Straight: Like most other polygon simplification algorithms, ours need not to evaluate error metric or energy function, so it requires less memory and processing time.
Efficiency: The algorithm is capable of simplifying complex models rapidly. For example, our implementation can create a 3560 face approximation (38% less of the original) of a 5804 face model in 02 seconds.
Quality: The approximations produced by our algorithm maintain high quality to the original model. After simplification the model remains almost same as the original.

Background and Related Work

We are primarily concerned with the problem of polygonal surface simplification. Given an initial triangulated surface, we want to generate a simplified model that, as much as possible, faithfully reproduces the features of the original. We assume that the input model (M_i) has been triangulated. The target approximation (M_t) will satisfy some given target criteria which are typically either a desired face count or a vision convincing level. We are interested in surface simplification algorithm that needs to maintain the topology of the model.

In recent years, the problem of surface simplification, and the more general problem of multiresolution modeling, has received increasing attention. Several different algorithms have been formulated for simplifying surfaces. Those algorithms which are most relevant to our work can be broadly categorized into three classes:

1. Vertex Decimation: Schroeder *et al.* [3] describe an algorithm which we would term vertex decimation. Their method iteratively selects a vertex for removal, removes all adjacent faces, and re-triangulates the resulting hole. The method uses vertex classification and re-triangulation schemes which are inherently limited to manifold surfaces, and they carefully maintain the topology of the model. While these are important features in some domains, they are restrictions for multiresolution rendering systems.

2. Vertex Clustering: The algorithm described by Rossignac and Borrel [2] is one of the few capable of processing arbitrary polygonal input. A bounding box is placed around the original model and divided into a grid. Within each cell, the cell's vertices are clustered together into a single vertex, and the model faces are updated accordingly. This process can be very fast, and can make drastic topological alterations to the model. However, while the size of the grid cells does provide a geometric error bound, the quality of the output is often quite low. In addition, it is difficult to construct an approximation with a specific face count, since the number of faces is only indirectly determined by the specified grid dimensions.

3. Iterative Edge Contraction: Several algorithms have been published that simplify models by iteratively contracting edges. The essential difference between these algorithms lies in how they choose an edge to contract. Some notable examples of such algorithms are those of H. Hoppe *et al.* [1].

It is critical that the approximate model lie within some distance of the original model and its topology remain unchanged. None of these previously developed algorithms provide the combination of straight, efficiency and quality that we assume.

Discussion

Edge Collapse: Our simplification algorithm is based on the iterative contraction of vertex pairs; a generalization of the iterative Edge Collapse technique. A pair contraction, which we will write (v_0, v_1) → v_0, moves the vertices v_0 and v_1 to the new position v_0, connects all their incident edges to v_0, and deletes the vertex v_1. Subsequently, any edges or faces which have become degenerate are removed. The effect of a contraction is small and highly localized. If (v_0, v_1) is an edge, then one or more faces will be removed. Starting with the initial model M_i, a sequence of pair contractions is applied until the simplification targets are satisfied and a final approximation M_t is produced. Because each contraction corresponds to a local incremental modification of the current model, the algorithm actually generates a sequence of models M_i, M_{i-1}, . . . , M_t. Thus, a single run can produce a large number of approximate models.

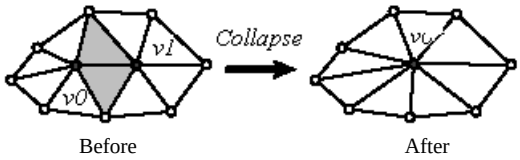


Figure1: Collapse of the edge (v_0, v_1) into a single point v_0.
(The shaded triangles become degenerate and are removed during the collapse.)

Pair Selection: We have chosen to select the set of valid pairs at initialization time, and to consider only these pairs during the course of the algorithm. Our decision is based on the assumption that, in a good approximation, points do not move far from their original positions. We will say that a pair (v_0, v_1) is a valid pair for contraction if:

1. (v_0, v_1) is a shared edge and
2. $Nv_0 .Nv_1 < t$, where t is a threshold parameter and N is unit normal vector.

Using a threshold of t = 0 gives a simple edge contraction algorithm. Higher thresholds preserve good topology. Naturally, this threshold must be chosen with some care; if it is too high, less number of polygons will be reduced, which is presumably undesirable. We must track the set of valid pairs during the course of iterative contraction. With each vertex, we associate the set of pairs of which is a member. When we perform the contraction (v_0, v_1) → v_0, not only does v_0 acquire all the edges that were linked to v_1, it also merges the set of pairs from v_1 into its own set. Every occurrence of v_1 in a valid pair is replaced by v_0, and duplicate pairs are removed.

Summary of Algorithm and Future works

We have first converted the conventional polygon data file into “pointed to a vertex list”. For this, we have made 2 (two) different files "Triangle (*.tri)" and "Vertex (*.ver)" from scanned CTdata where "Vertex (*.ver)" is pointed array of vertex and "Triangle (*.tri)" is the index for polygon and using these files we have generated polygonal data in (“*.ply”) file format which has the information of vertex list in float (x, y, z) triples and index number of triangles in int(n_1, n_2, n_3) forms. Then using above (“*.ply”) file we have generated another polygonal data in (“*.ply”) file format that has more information than above which includes vertex list in (x, y, z) triples with surface normals at each vertex (nx, ny, nz) triples, index number of triangles and edge list of triangles. We have used a geometry removal algorithm “Edge Collapse” method to simplify the polygonal model. At first we have made the member vertex list with shared triangles. To achieve more reliable results, when

corners of two faces intersect at a point, the faces should be defined as sharing a single vertex rather than using two separate vertices which happen to be coincident in space. Then we have collapsed an edge by averaging the end points of the edge and averaging the surface normals of concerning vertices under some conditions. By changing the conditions we can simplify the model up to a user defined level. Next we have upgraded the triangle list, vertex list and edge list respectively. Our algorithm can quickly produce good quality approximations of polygonal surface models. It is one of the fastest algorithms available for producing quality approximations and capable of simplifying graphical models of reasonably high complexity.

The fundamental operation of our algorithm is Edge Collapse, which we will write (v_0, v_1) v_0. To perform this simple collapse, we need to perform the following three steps:

(1) To move vertices v_0 and v_1 to the position v_0,
(2) To replace all occurrences of v_1 with v_0, and
(3) To delete v_1 and any degenerate faces.

There are certainly further improvements that could be made to the algorithm. We have used a simple process for selecting valid pairs. It is quite possible that a more sophisticated adaptive process could produce better results. We have not addressed the issue of disjoint vertices and surface properties such as color, texture etc. A more complete system would also include support for multiple attribute values per vertex. It would capable to associate surface properties such as color, texture etc. This would help to solve the problems posed by texture seams and by surface normal discontinuities. We also believe that the memory usage of our implementation could be reduced.

Results

Our algorithm can produce quality approximations in short amounts of time. Table: 1 summarizes the running time of our current implementation using the models shown below (Figure:2). It is noticeable that features such as horns, eyes, mouth and hooves remain recognizable through many simplifications. Only at extremely low levels of detail do they begin to disappear. In our experiments, we have also found that using optimal vertex placement tends to produce better shaped meshes. Figures: 3 demonstrate the performance of our algorithm on larger models and compare the efficiency of our algorithm with Jeff Somers algorithm. This represents a significant simplification (38.7% of the original), but all the major details of the original still remain. In particular, we have noticed that the contours on the interior of the ear, eye, nose and the large contours around the mouth have been preserved significantly.

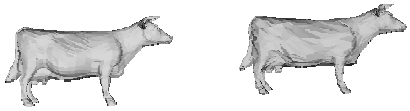
Conclusion

We have proposed a surface simplification algorithm which is capable of rapidly producing high quality approximations of graphical models. Our algorithm uses iterative pair contractions to simplify models. Our algorithm provides simplicity, efficiency and quality. While certain other algorithms are faster or generate higher quality approximations than ours, they typically do not meet the capability of our algorithm in all three areas. The only algorithm capable of simplifying arbitrary polygonal models, vertex clustering [2], does not reliably produce high quality approximations. None of higher quality methods [1] supports simplicity. However, our system uses a more efficient means to track plane sets. Besides these we have found a new scope of preserving 3D shape at boundary area after significant simplification of the original model. There remain various aspects of our algorithm that could be improved upon. We have used a simple process for selecting valid pairs. It is quite possible that a more sophisticated adaptive process could produce better results. We have not addressed the issue of disjoint vertices and surface properties such as color, texture etc.

Tables and Figures:

No.	Model	Faces	Faces After Reduction	Faces Reduced	Simplifying Time(S)	Reduction (%)
(a)	Cow	5804	3560	2240	02	38.7 %
(b)	Cat	671	437	234	01	34.9 %
(c)	Hind	6448	4812	1636	0.8	25.4 %
(d)	Car	104254	6307	3927	02.5	37.5 %

Table: 1



5804 angle 3560 Triangle

2. (a) Cow model

671 Triangle 437 Triangle

2. (b) Cat model

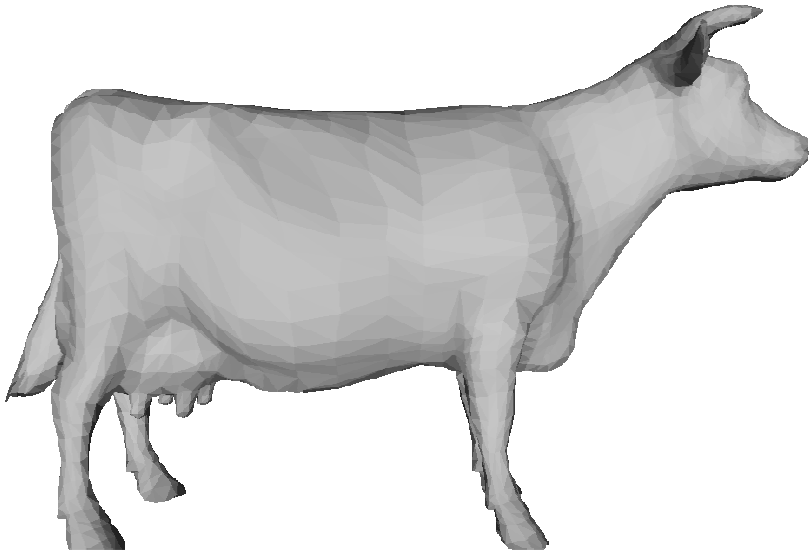
6448 Triangle 4812 Triangle

2. (c) Hind model

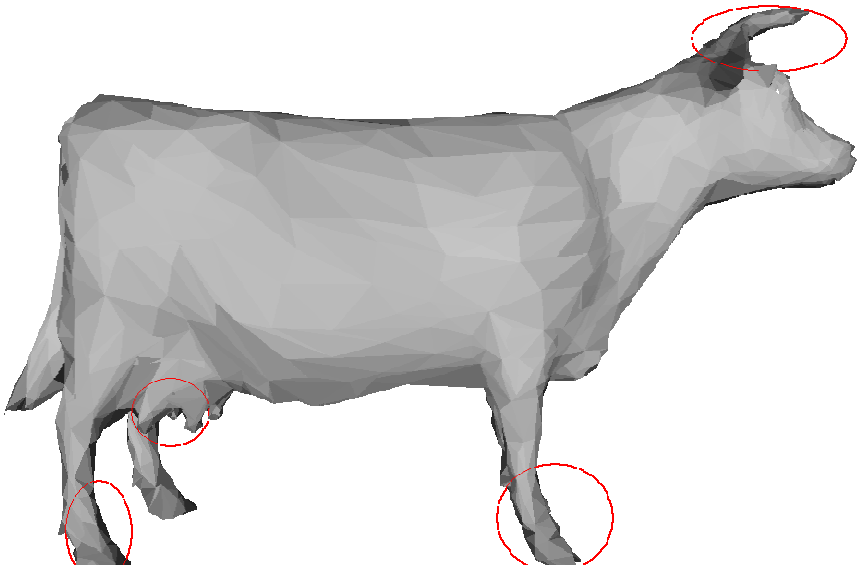
10254 Triangle 6307 Triangle

2. (d) Car model

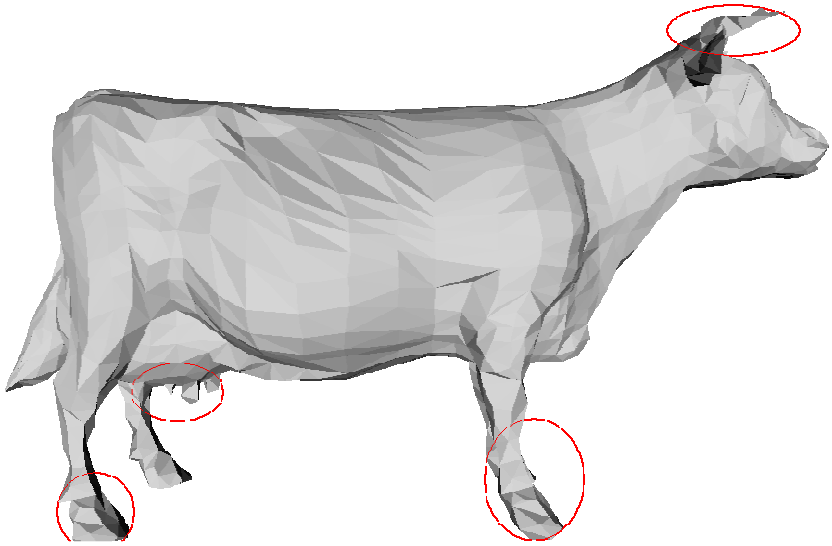
Figure 2: Original and Simplified Models.



3. (a) Original Model (Flat Shading): 5804 triangles



3. (b) Simplified Model (Our Algorithm): 3560 triangles



3. (c) Simplified Model (Jef Somer's Algorithm): 3560 triangles

Figure 3: Vision convincing comparison between Our Algorithm and Jef Somer's Algorithm. (The red encircled area shows that our algorithm is more efficient to preserve the boundary area).

References

[1] H. Hoppe, T. DeRose, T. Duchamp, J. McDonaldz and W. Stuetzlez, Mesh Optimization, University of Washington, *(SIGGRAPH '93 Proc) of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, (1993), 19-26, ACM New York, USA.

[2] J. Rossignac and P. Borrel, Multi-resolution 3D approximations for rendering complex scenes, Modeling in Computer Graphics: Methods and Applications, Microsoft Academic Publications, 1993.

[3] W.J. Schroeder, J.A. Zarge and W.E. Lorensen, Decimation of triangle meshes, *Computer Graphics (SIGGRAPH '92 Proc.)*, 26(2) (1992), 65-70.