A Tetrahedron Based Volume Model Simplification Algorithm

Jinjin Hong, Lixia Yan, Jiaoying Shi
State Key Lab. of CAD&CG, Zhejiang University
Hangzhou, 310027, P.R.China

Abstract

In VISC the volume models may be consisted of a large number of tetrahedrons. It is often required to reduce the number of tetrahedrons for approximating such objects. In this paper we present a new algorithm to reduce the number of tetrahedrons in a tetrahedron-based volume model. The key advantages of the new algorithm are: (1) it is simple to implement; (2) high reduction rates and excellent results can be achieved; (3) It allows a user defined simplification rate to generate a multi-resolution representation. This algorithm has been used in different application areas such as volume rendering, finite elements computation.

Keywords: *Tetrahedron, Simplification, Mesh, Finite Element.*

1. INTRODUCTION

Tetrahedron meshes are one of the most popular representations of volume model for Visualization in Scientific Computing (VISC). There is an increasing set of data acquisition techniques which generates tetrahedron meshes as output, such as Delauny Tetrahedron Reconstruction, Spatial Mesh Reconstruction, Finite Element Generation. However, most of these techniques generate much more tetrahedrons than necessary to present the given object with a small approximation error. Such huge amount of data lead to problems on data storage, volume rendering and finite element computation. Animation and real time processing of such data set is almost impossible even on high performance hardware.

Few researches on simplifying volume data have been developed until now while many techniques aiming at reducing surface complexity were published. Following, some general and valuable solutions are mentioned:

- Hansen's coplanar facet merging[6]: Coplanar or nearly coplanar data are searched, merged in larger polygon and then retrianglated into fewer simple facets.
- Schroder's mesh decimation[7]: all vertices that satisfy some criterion are removed and the remaining holes are retrianglated.
- Turk's mesh re-tiling[8]: a new vertices is inserted at random on the original surface mesh, and then moved on the surface to be displaced on maximum curvature locations; the original vertices are then iteratively removed. A retiled mesh, built on the new vertices only, is returned.
- Rossignac's point coalescence[9]: the ambient space is subdivides into smaller subspaces. Multiple vertices in a subregion are merged into a single vertex using a weighted approximation. The merged vertices are then reconnected with their neighbors to form a collection of facets.
- Hoppe's mesh optimization[10]: the mesh is evaluated by a global energy function and minimized either by removing/moving vertices or collapsing /swapping edges.

All these algorithms are only available for surface simplification either by merging elements or by resampling vertices of the original object. Our work, on the other hand, provides a method to simplify the tetrahedron mesh of a volume model. The basis of our tetrahedron mesh simplification algorithm is to reduce the number of border tetrahedrons using surface vertex removal and then construct regular hexahedron mesh to replace the internal tetrahedrons of the original model. The resulting hole between the simplified surface and hexahedron mesh will be filled with tetrahedrons at last.

The remainder of the paper is organized as follows. We outline the conception of volume data in the next section and implementation of our simplification algorithm in section 3. An Example is given to test our algorithm in section 4. Section 5 concludes the paper with some remarks on future research directions.

2. THE VOLUME DATA

Volume visualization is a methodology used for realizing the inner structure and complex behavior of 3D volume objects. The Element in a volume data set can be a tetrahedron (called 4-nodes), a pentahedron (called 6-nodes), or a hexahedron (called 8-nodes). Tetrahedron data are most popularly used to approximate a volume model because of their ability to form any polyhedrons discretionarily. According to our paper, three types of tetrahedrons are defined, see Figure 1. Assume E_0 and E_1 are two parallel planes where each vertex of a tetrahedron is located:

- T0-tetrahedron: three vertices of the tetrahedron are in the upper plane E_0 and the rest one is in the lower plane E_1.
- T1-tetrahedron: three vertices of the tetrahedron are in the lower plane E_1 and the rest one is in the upper plane E_0.
- T2-tetrahedron: two vertices of the tetrahedron are in the upper plane E_0 and the other two are in the lower plane E_1.

A polyhedron can be divided into several T0, T1 and/or T2 tetrahedrons. For example, showed in Figure 2, a hexahedron is divided into two pentahedrons and each of the pentahedrons is divided into three tetrahedrons respectively. This decomposition, however, is not unique because the diagonal edges change across a polyhedron. Since the faces of a polyhedron are usually non-planar, it is important to ensure that adjoining polyhedrons have matching diagonals to prevent gaps.

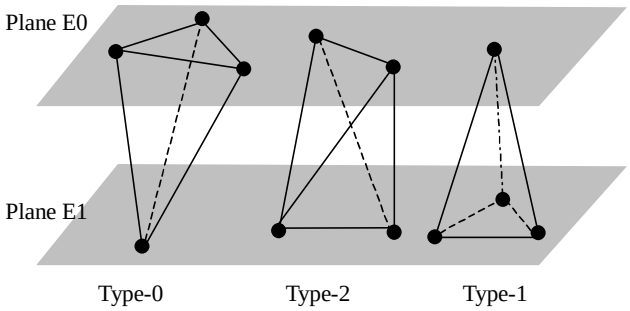


Figure 1: The T0, T1 and T2 tetrahedrons.

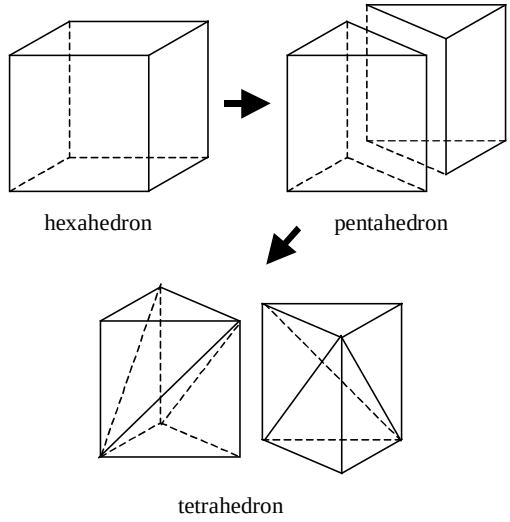


Figure 2: Decomposition of a hexahedron.

3. DSCRIPTION OF THE ALGORITHM

Differing from the traditional surface simplification, our volume simplification algorithm not only simplify complexity of the surface but the internal volume data as well. Sharp edges of the volume model must be preserved in course of simplification. In addition, the simplified tetrahedron mesh should be more regular than the original mesh in order that the post-processing such as force analysis, conflict detection, deformation computation and volume rendering can take advantage of its regularity.

The input objects that our algorithm can accept and process are layered tetrahedrons gained from 3D reconstruction of layered scanning images (MRI or CT images). A layered tetrahedron is defined as a tetrahedron with vertices only on two adjoining planes parallel with each other, see Figure 3.

3.1 The Main Loop

Our simplification algorithm will establish a multi-resolution volume model depending on the user requirements of simplification rate. In this section, we adopt a layered simplification approach benefited from the independency of tetrahedrons between two arbitrary layers. After the original data and user defined simplification rate r have been read, the number of simplified layers L_S as well as the average number of simplified tetrahedrons per layer T_S is calculated as:

$$L_S * T_S = (\sum_{i=0}^{L_T} T_i) * r \quad (1)$$

$$L_T / L_S = (\frac{1}{L_T} \sum_{i=0}^{L_T} T_i) / T_S \quad (2)$$

where r is a user defined simplification rate which is percentage of the reduced number of simplified tetrahedrons and the number of original tetrahedrons. L_T is the number of original layers and L_S the number of simplified layers. T_i denotes the number of original tetrahedrons in layer i and T_S the average number of simplified tetrahedrons per layer.

Our algorithm starts by fetching M layers of tetrahedrons from the input (M = L_T / L_S), once the manipulation to these tetrahedrons is completed, one layer of newly-generated tetrahedrons is output. We finish when all of the original tetrahedral data are processed. Therefore, the manipulation to each M layers of tetrahedrons is what we are interested in. First, all of the tetrahedrons are classified into two categories: border tetrahedrons and non-border (internal) tetrahedrons. We define the former as a tetrahedron that includes the surface facets of a volume model and the latter as a tetrahedron not including any surface facets. Next, a vertex removal approach is introduced to simplify those border tetrahedrons and the preservation of sharp edges should be considered. Then a number of hexahedrons will be substituted for those non-border tetrahedrons. Finally, the resulting hole between the simplified surface and substituent hexahedrons is filled with tetrahedrons. More detailed discussion can be seen in the following. Let Tetra1, Tetra2, … , TetraM denote tetrahedrons in layer 1, 2, … , M, with vertices located in M+1 layers denoting Layer0, Layer1, … , LayerM.

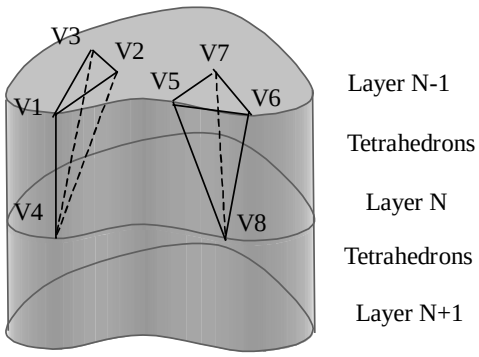


Figure 3: A layered tetrahedron model. Tetrahedron $V_1V_2V_3V_4$ and $V_5V_6V_7V_8$ are called non-border (internal) tetrahedron and border tetrahedron respectively if the former does not include the model's surface facet while the latter does.

3.2 Surface Simplification

Our border tetrahedral simplification is a typical surface simplification algorithm, that is, it starts with the original surface and successively simplifies it. It removes vertices from Layer1 to Layer(M-1) and retriangulates the resulting holes until no further vertices can be removed. The triangle mesh left, with all vertices in Layer0 or LayerM, therefore, is the simplified surface that we need. Figure 4 illustrate if we remove a vertex Vr from the surface its adjacent triangles are removed and the remaining hole is retriangulated.

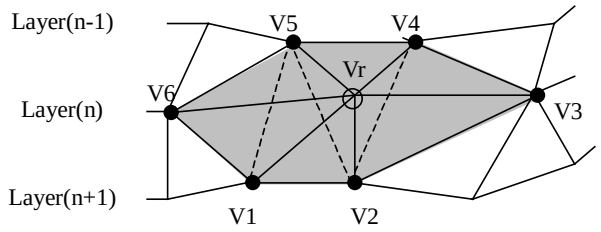


Figure 4: Removing vertex Vr and retriangulating the remaining hole.

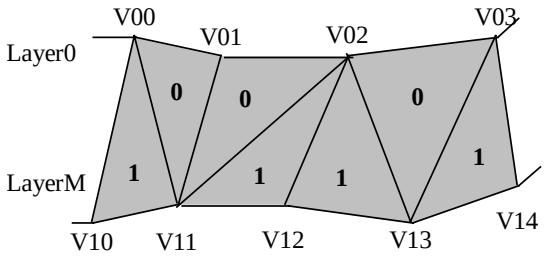


Figure 5: The simplified surface where all of vertices are in Layer0 or LayerM. We call a triangle T0-triangle if its base-side in Layer0, otherwise call it T1-triangle.

3.2.1 Vertex Removing

One of the crucial parts in the algorithm is vertex removal because in this step not only have the vertices to be removed but also the new triangle mesh is built to approximate the surface. As Figure 4, let C_r be the triangle set where each triangle adjoins the vertex V_r in Layer(n), $n \in \{1,2,\cdots,M-1\}$ and S_r the new triangle set produced by removing the vertex V_r and triangulating the hole $V_1V_2V_3V_4V_5V_6$. Since there are only two vertices V_3 and V_6 belonging to Layer(n) in C_r, we connect them with their adjacent vertices to form two triangles $\Delta V_2V_3V_4$ and $\Delta V_1V_5V_6$. The rest hole $V_1V_2V_4V_5$ is then filled with triangles $\Delta V_5V_1V_2$, $\Delta V_5V_2V_4$ with base-sides in Layer(n+1) or Layer(n-1). All of these triangles combine to form S_r that we finally need.

Yet there exist some other algorithms to construct triangle mesh, e.g. the surface reconstruction by extracting the contour lines of Layer0 and LayerM, the main disadvantage of those algorithms is that they are more complicated and time-consuming. Also, it's hard to handle topology ambiguity of the complicated volume model.

3.3 Hexahedron Mesh Construction

In this section we substitute regular hexahedrons for the internal tetrahedrons. The algorithm proceeds firstly by constructing a closing box for M layers of original tetrahedrons. The closing box is divide into

$$N = \sqrt{T_S} \tag{3}$$

sub-hexahedrons where T_S is the number of simplified tetrahedrons per layer. According to the tetrahedron each hexahedron includes, it falls into 3 classifications: A-hexahedron which does not includes any tetrahedrons of original model, B-hexahedron which at least includes one border tetrahedron and C-hexahedrons which only includes non-border tetrahedrons. We adopt all C-hexahedrons and subdivide each of them into 6 tetrahedrons as the simplified non-border tetrahedrons.

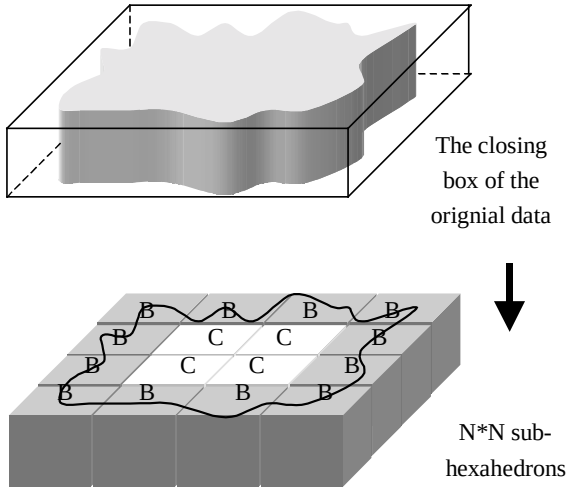
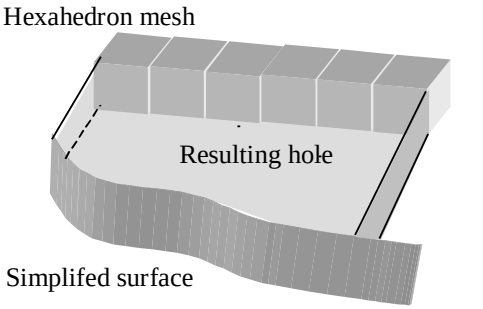


Figure 6: Dividing the closing box of the original data into N*N hexahedrons.

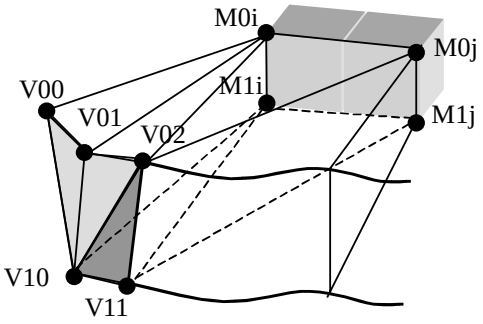
3.4 Filling The Resulting Hole

The main problem in this section is how to fill the resulting hole between the simplified surface and hexahedrons we built with tetrahedrons (Figure 7a). While, in general, this is a very complicated task, it can be easily solved by keeping track of the correspondence between the simplified surface and hexahedrons. The correspondence is the clue to this problem. It allows all vertices in surface to be counterclockwise sorted and assigned as V_{00}, V_{01}, V_{02}, V_{03} … if they belong to Layer1 or as V_{10}, V_{11}, V_{12}, V_{13} … if they belong to LayerM. Correspondingly, each vertex in hexahedron mesh's surface should also be sorted counterclockwise and assigned as M_{00}, M_{01}, M_{02}, M_{03}…

or M_{10}, M_{11}, M_{12}, M_{13} …. Note that in Figure 5, we define two kinds of triangles in simplified surface: T0-triangle with base-side in Layer0 and T1-trinangle with base-side in LayerM. The essential steps of hole filling algorithm are as follows:



(a)



(b)

Figure 7: (a) is the resulting hole between the simplified surface and hexahedron mesh. (b) is a method to fill the resulting hole.

(1) We start with an arbitrary T0-triangle in the simplified surface and search for its adjacent triangle counterclockwise until we reach a T1-triangle. The combination of those continuous T0 and T1 triangles is defined as a triangle-set unit B_0. For example, two T0-triangles $\Delta V_{00}V_{01}V_{10}$, $\Delta V_{01}V_{02}V_{10}$ and one T1-triangle $\Delta V_{10}V_{11}V_{02}$ form a triangle-set unit B_0, see Figure 7(b). Then we have to search every arris counterclockwise in the hexahedron mesh's surface to find an arris $M_{0i}M_{1i}$ that is nearest to B_0. The distance between an arris $M_{0i}M_{1i}$ and a triangle-set unit B_0 is defined by

$$d(M_{0i}M_{1i}, B_0) = Max[d(M_{0i}, B_0), d(M_{1i}, B_0)] \tag{4}$$

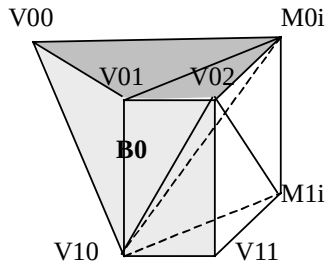
where $d(M, B_0)$ is the distance between a point M and a triangle-set unit B_0 defined by

$$d(M_{0i}, B_0) = \underset{p \in B_0}{Max}\left(p^T v_{0i}\right)^2$$

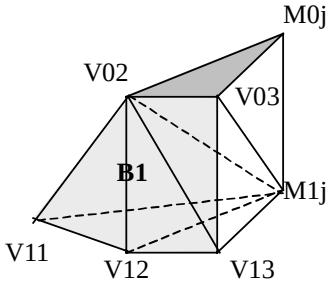
$$d(M_{1i}, B_0) = \underset{p \in B_0}{Max}\left(p^T v_{1i}\right)^2 \tag{5}$$

where $v_{0i} = [x_{0i} \quad y_{0i} \quad z_{0i} \quad 1]^T$, $v_{1i} = [x_{1i} \quad y_{1i} \quad z_{1i} \quad 1]^T$ are the vectors of vertices M_{0i} and M_{1i} respectively; $p = [a \quad b \quad c \quad d]^T$ indicates the plane equation $ax + by + cz + d = 0$ of each triangle in B_0. Note that $a^2 + b^2 + c^2 = 1$.

Now that we have got a polyhedron $V_{00}V_{01}V_{02}V_{11}V_{10}M_{0i}M_{1i}$, we can divide it into several T0-tetrahedrons $V_{00}V_{01}M_{oi}V_{10}$, $V_{01}V_{02}M_{oi}V_{10}$, one T2-tetrahedron $M_{0i}V_{02}M_{1i}V_{10}$, and one T1-tetrahedron $V_{10}V_{11}M_{1i}V_{02}$ to fill the resulting hole, see Figure 8(a). Indicated B_0 to be used.



(a)



(b)

Figure 8: Decomposition of a hexahedron.

(2) If the next triangle adjacent to B_0 is a T1-triangle, keep on searching for T1-triangles counterclockwise in the simplified surface until we reach a T0-triangle, thus the next triangle-set unit B_1, contrary to B_0, is the combination of several T1-triangles and one T0-triangle. Otherwise if the triangle adjacent to B_0 is a T0-triangle, B_1 is combined with several T0-triangles and one T1-triangel. Once in the hexahedron mesh's surface an arris

$M_{0j}M_{1j}$ that's nearest to B_1 is found, decomposition is done showed in Figure 8(b). A polyhedron $V_{02}V_{03}M_{0j}V_{11}V_{12}V_{13}M_{1j}$ is divide into several T1-tetrahedrons $V_{11}V_{12}M_{1j}V_{02}$, $V_{12}V_{13}M_{1j}V_{02}$, one T2-tetrahedron $V_{03}V_{02}V_{13}M_{1j}$ and one T0-tetrahedron $V_{02}V_{03}M_{0j}M_{1j}$. Indicate B_1 to be used. In order to void tetrahedron intersecting, each counterclockwise search must resume the previous search from the previous ending position, and moreover, any arrises or triangle-set units having been indicated used must not be searched once again.

(3) $M_{0i}M_{1i}M_{1j}M_{0j}V_{02}V_{11}$ is a pentahedron which is divided into 3 tetrahedrons to fill the hole, see Figure 7. Indicate arrises $M_{0i}M_{0i}$, $M_{0i+1}M_{1i+1}\ldots M_{0j}M_{1j}$ to be used.

(4) Repeat step (1), (2), (3) until we reach the triangle-set unit B_0 again. We complete when the resulting hole is filled without any gaps.

3.5 Exception Control

The rest triangles between the final triangle-set unit B_f and first triangle-set unit B_0 can not always form a triangle-set unit. We distinguish two cases assuming that the last triangle of B_f is a T1-triangle.

Case 1: If the rest triangles are several T1-triangles (Figure 9a), we have to insert same number of T1-tetrahedrons between B_f and B_0 (Figure 10a).

Case 2: If the rest triangles are several T0-triangles instead (Figure 9b), we will insert same number of T0-tetrahedrons and one T2-tetrahedron between B_f and B_0 (Figure 10b).

It should also be considered that C-hexahedrons do not exist if the user defined simplification rate are high enough. That case can also be solved using our simplification algorithm by degrading all C-hexahedrons to one arris.

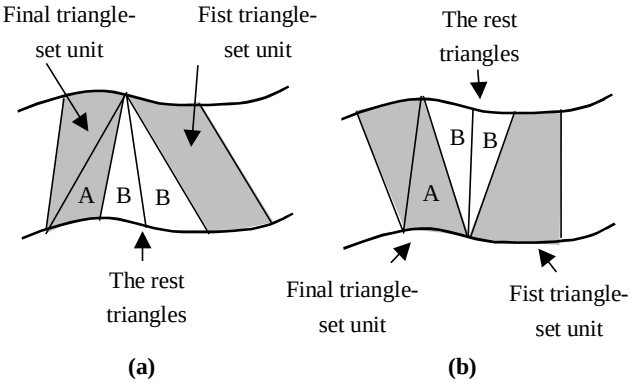


Figure 9: The rest triangles between B_f and B_0.

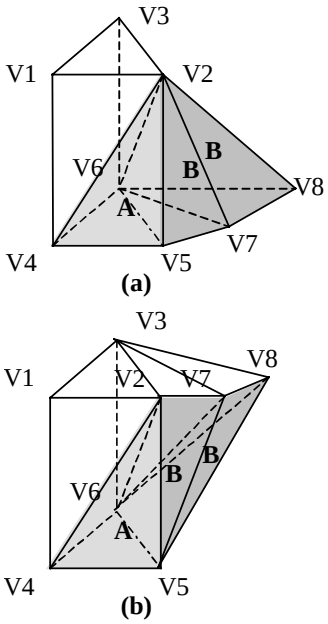


Figure 10: Tetrahedron insertion. In (a), we add two T1-tetrahedrons $V_5V_6V_7V_2$, $V_7V_6V_8V_2$ to polyhedron $V_1V_2V_3V_4V_5V_6$ while in (b) we add two T0-tetrahedrons $V_2V_7V_3V_6$, $V_7V_8V_3V_6$ and one T2-tetrahedron $V_5V_6V_8V_7$ to it.

4. RESULTS

An example is given to illustrate the results of our tetrahedron simplification algorithm, which have been very encouraging and are summarized below. Figure 11 illustrates tetrahedron mesh hierarchy of a pelvis with 16104 tetrahedrons. The original model, showed in Figure 11(a), is simplified with different simplified rate of 50%, 75%, 90%, 95%, 99%, so the resulting number of tetrahedrons in the simplified model is reduced. The simplified model is quit similar to the original one when the simplification rate is below 75%, and it's acceptable when simplification rate is 90%, and some main features still can be preserved when simplification rate rise to 99%. The simplification results are presented in Table 1, with the index specifying the corresponding model in Figure 11.

Table 1: Simplification of a pelvis mode

Index	Simplification rate	Layers	Vertices	Tetrahedrons
(a)	Original model	23	2764	16104
(b)	50%	23	1674	7302
(c)	75%	12	661	2671
(d)	90%	8	344	1227
(e)	95%	6	232	714
(f)	99%	3	84	179

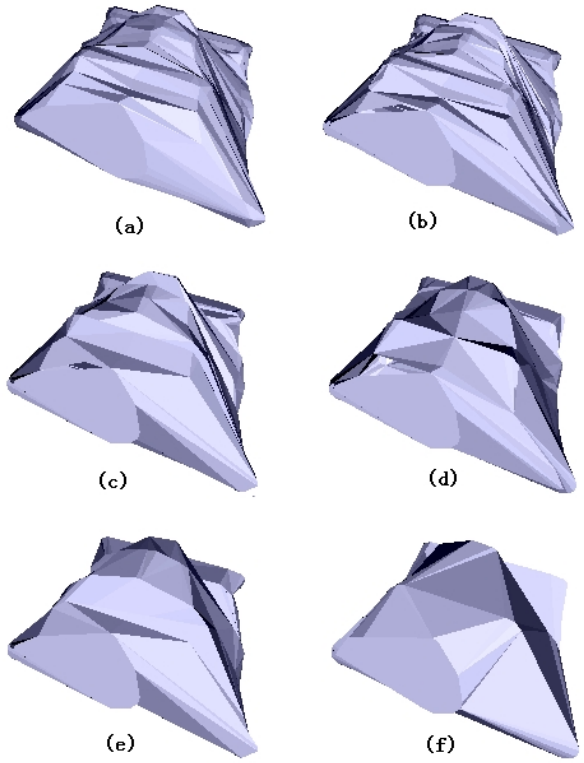


Figure 11: Simplifed models with different simplification rate. (a) is the original model; (b), (c), (d), (e), (f) is the simplified ones with simplified rates of 50%, 75%, 90%, 95%, 99% respectively.

5. CONCLUSION

We have described an algorithm for solving tetrahedron simplification problem. The strengths of our method are that it (a) works for volume data; (b) can preserve sharp edges; (c) establish a multi-resolution volume data; (d) is easy to implement.

This tetrahedron simplification algorithm has been applied to our virtual surgery simulation system. One we are currently exploring, is the use of multi-resolution object hierarchies in collision detection, cutting and suturing. The idea here is to recursively do operations among the multi-resolution description of object, starting from the lowest resolution representations and moving up to the higher resolutions. Furthermore, this hierarchical approach can be interrupted allowing us to trade accuracy for speed.

6. ACKNOWLEDGEMENTS

Our special thanks go to Chiyi Cheng and Jianning Wang for their numerous useful suggestions. We are also grateful to the anonymous reviewers for their useful comments. This project is granted by the National Natural Science Foundation of China.

7. REFERENCES

[1] John Waterworth. *Virtual Reality in Medicine: A Survey of the State of the Art. Report RR-98.02. Department of informatics, Umea University, Sweden.*

[2] Reinhard Klein, Gunther Liebich, W. Straber. *Mesh Reduction with Error Control. In Visualization'96 Conference Proceeding, pages 311-318.*

[3] Taosong He, Lichan Hong, Arie Kaufman, Amitabh Varshney, Sidney Wang. *Voxel Based Object Simplification. In Visualization'95 Conference Proceedings, pages 296-303.*

[4] David N. Kenwright, David A. Lane. *Optimaization of Time-Dependent Particle Tracing Using Tetrahedral Decomposition. In Visualization'95 Conference Proceeding, pages 321-328.*

[5] Jiaoying Shi, Wenli Cai. *Algorithms and Systems on Visualization in Scientific Computing, Science Publishing Company, 1996.*

[6] Charles Hansen, Paul Hinker. *Isosurface extraction SIMD architectures. In Visualization'92, pages1-21, Oct 1992.*

[7] William J. Schroeder, Jonathan A. Zarge, William E. Lorensen. *Decimation of Triangle Meshes. In Computer Graphics (SIGRAPH'92 Proceedings), volume 26, pages 65-70, July 1992.*

[8] Greg Turk. *Re-tilling Polygonal Surfaces. In Computer Graphics (SIGRAPH'92 Proceedings), volume 26, pages 55-64, July 1992.*

[9] J. Rossignac, P. Borrel. *Multi-resolution 3D Approximation for Rendering Complex Scences. In Modeling in Computer Graphics: Mothods and Applications, pages 455-465. Springer Verlag, 1993.*

[10] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, Werner Stuetzle. *Mesh Optimization. In Computer Graphics(SIGRAPH'93 Proceedings), volume 27, pages 19-26, August 1993.*

About the author

Jinjin Hong is a Master student in State Key Lab. of CAD&CG. Her main research interests include CG, VISC.

E-mail: **hongjj@cad.zju.edu.cn**

Lixia Yan is a PHD student in State Key Lab. of CAD&CG. Her main research interests include CG, VISC.

E-mail: **yanlx@cad.zju.edu.cn**

Jiaoying Shi is a Professor in State Key Lab. of CAD&CG. His main research interests include CG, VISC, Multimedia, VR and Computer Architecture.

E-mail: **jyshi@cad.zju.edu.cn**