Feature Preserving Volumetric Data Simplification for Application in Medical Imaging

C. Jin T. Fevens S. Li S. P. Mudur
Computer Science and Software Engineering Department
Concordia University
1455 de Maisonneuve Blvd. W.
Canada, H3G 1M8, Montreal, QC
{chao_jin, fevens, shuo_li, mudur}@cs.concordia.ca

ABSTRACT

In this paper, we propose a new simplification algorithm to reduce the large amount of redundancy in 3D medical image datasets and generate a new representation in tetrahedral meshes with considerably lower storage requirements. In the proposed algorithm, we first apply level set segmentation to partition the volume data into several homogenous sub-regions. We consider the interior boundaries between sub-regions as contributing more to the significant visible features. Next we convert the regular grid data into a tetrahedral representation and simplify the irregular volume representation by iteratively removing tetrahedra without significantly altering the exterior boundary or interior field distribution features. Within each sub-region, field gradients, tetrahedral aspect ratio changes and variances of interior region values are further used so as to maintain features of the original dataset in regional interiors. We tested our algorithm on several 3D medical datasets. The promising results show that we reduce redundancy and yet preserve important features and structures present in the original data set for decimation rates up to 50%.

Keywords

Medical Imagery, Mesh Simplification, Irregular Tetrahedral Meshes, Level of Detail

1. INTRODUCTION

In the past few decades, 3D medical image analysis has become one of the most active research areas supporting computer aided diagnosis. Medical scanning devices, such as those generating Computed Tomograph (CT) or Magnetic Resonance Imaging (MRI) scans, are continuously increasing in their resolution capabilities. The resulting volumetric data sets are thus getting larger with increasing demands on time and storage resources for tasks such as archiving, loading, rendering, transmission, etc.

A more efficient volumetric representation, which maintains the same features but uses less physical storage space, is necessary for further visualization or analysis [Kau91a, Kau93a, Nie00a]. However, due to the specific characteristics of regular grids, it is very hard to achieve accurate simplification. For example, to simply replace clusters of voxels (a volume element in a 3D regular grid) with 'supervoxels' would introduce too much noise and blur significant features. Therefore, we use another representation of volume data, a tetrahedral mesh, to perform the simplification. The tetrahedral mesh has attracted much attention over the last decade or so since it provides greater flexibility and other representations can be converted into tetrahedral meshes relatively easily. The basic representational primitives, tetrahedra, are easy to deform and to merge or subdivide. It is convenient to assign properties and functions to the vertices and to tetrahedral cells. Computational steps such as interpolation, integration, and differentiation are fast and often can be done in similar forms. For example, finite element analysis is conveniently performed on tetrahedral meshes. Also, the triangles that are generated by the faces of tetrahedra may be rendered using hardware acceleration [Yao00a].

Conference proceedings ISBN 80-903100-7-9
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

Simplification of tetrahedral meshes has been studied in the past decade [Chi03a, Cho02a, Cig00a, Gel99a, Hong03a, Sta98a, Tro98a, Tro99a]. In 1998 and 1999, Trotts et al. [Tro98a, Tro99a] presented a method for simplification by extending a polygonal geometry deduction technique with a trivariate spline function associated with each tetrahedron to detect the features. In 1998, Staadt and Gross [Sta98a] extended the work of Hoppe [Hop96a] on progressive triangular meshes to tetrahedral meshes to generate an incrementally refined progressive tetrahedralization based on edge collapse. In 2000, Cignoni et al. [Cig00a] gave a framework for incremental 3D mesh simplification also based on edge collapse. In 2002, Chopra and Meyer [Cho02a] introduced a fast tetrahedral decimation algorithm called TetFusion. It preserves all cells, with large gradients. Recently, two groups of researchers, Hong and Kaufman [Hong03a], and Chiang and Lu [Chi03a] used Morse theory to detect topological critical points of original data, with the aim to preserve the topological structure of isosurfaces during simplification.

However, existing algorithms are not well suited for 3D medical data sets because the data are naturally divided into different regions, which need to be treated individually, and the data obtained from scanning devices are usually noisy. Traditional volumetric simplification algorithms deal with features based on local measurement, such as gradient or aspect ratio [Cho02a, Cig00a, Sta98a, Tro98a, Tro99a]. But local measurements sometimes lead to the misidentification of features. For example, in Fig. 1, we can see that the data is composed of two regions. However, by the local feature measurement method, based on the gradients, it cannot partition the two regions. Instead these methods misidentify the boundary feature as a set of horizontal lines.

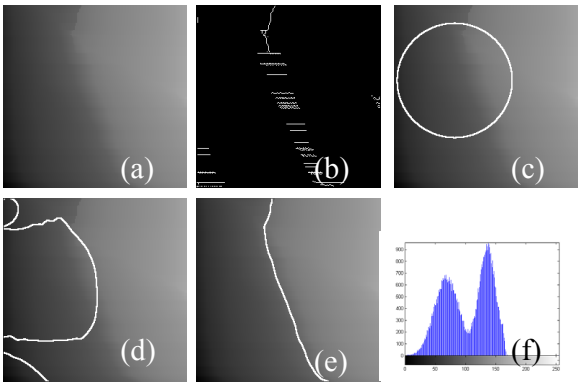

Figure 1: a) a 2D dataset which includes the two Gaussian distributions shown in f). b) the incorrect features detected by gradient method. c) and d) two intermediate stages in level set segmentation process. e) feature detection result by using level set method.

The drawback in the method of maintaining the topological structure of isosurfaces [Chi03a, Hong03a], is that it is too sensitive to noise, which is very likely to be present in 3D medical data [Web03a]. Thus, they cannot achieve high decimation rates.

To meet the requirement of simplifying tetrahedral medical images while preserving visible features, our new method first applies level set segmentation, which is robust to noise, to partition the volumetric medical data into several homogenous regions. Then, it simplifies each region individually. The final result of our method is a tetrahedral mesh, which maintains the fidelity of features as present in the original data while using much less physical storage space.

This paper is organized as follows. In section 2, we introduce our methodology. Section 3 shows our experimental results, and the final section 4 contains our conclusions.

2. METHODOLOGY

The first step of our algorithm is to apply level set segmentation in a preprocessing phase to partition the data into several homogenous regions. Then, the regular grid data is converted into an irregular mesh by simple tetrahedralization. The boundaries between sub-regions represent different features and should be preserved during the simplification phase. Finally, the tetrahedral cell decimation is applied to each sub-region individually. Within each sub-region, the algorithm will always preserve features detected by local feature measurement. Thus, by this approach, we preserve more of the significant features than other existing algorithms. Also by simplifying within regions without worrying about topology, we have greater flexibility during the simplification phase while at the same time not losing out on any visible features.

Level set segmentation

2.1.1 Level Set Function

In 1988, Osher and Sethian [Osh88a] proposed a new segmentation approach based on the class of deformable models, referred as "level set" or "geodesic active contours or surfaces". The approach, based on an evolving curve naturally dividing the image, represented as the domain $\Omega \in \mathbb{R}^2$, into two parts. The method has become popular because of its ability to capture the topology of shapes in 3D datasets. The curve C is represented implicitly via a *Lipschitz Function*, which is also referred to as the level set function ϕ, where $C=\{(x,y) \mid \phi(x, y) = 0\}$. The curve divides the image into a region where $\phi(x, y)$ is positive valued and a complementary region where $\phi(x, y)$ is negative

valued. The evolution of the curve is given by the zero-level curve at time *t*, and the curve *C* evolves according to direction and speed of dictated by force *F*, as described in Equation (1).

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| F, \quad \phi(0, x, y) = \phi_0(x, y) \tag{1}$$

Equation (1) is level set function, where the set $C = \{(x, y) | \phi_0(x, y) = 0\}$ defines the initial contour.

A particular case is the motion by mean curvature. This is given by

$$F = div\left(\frac{\nabla \phi}{|\nabla \phi|}\right)$$

as the curvature of ϕ passing through (x, y), which when substituted into Equation (1) gives us:

$$\begin{cases} \dfrac{\partial \phi}{\partial t} = |\nabla \phi| \, div\left(\dfrac{\nabla \phi}{|\nabla \phi|}\right), & t \in (0, \infty), x \in \mathbb{R}^2 \\ \phi(0, x, y) = \phi_0(x, y), & x \in \mathbb{R}^2 \end{cases} \tag{2}$$

where ϕ_0 is initial level set function.

2.1.2 Level Set Segmentation

In 2000, Samson *et al*. [Sam00a] proposed a supervised classification model to find a partition composed of homogeneous regions, assuming that the number of classes and their attribute value properties are known. Therefore, for segmentation into *K* regions, the proposed method uses *K* level set functions ϕ_i : (i ∈[1, *K*]) to represent each of region as shown in Equation (3), which demonstrates the energy function. It consists of three terms: minimal variance energy E_{minv}, minimal length energy E_{minl}, and non-overlap energy $E_{nonover}$.

$$E(\phi_i, \cdots \phi_k) = E_{minv}(\phi_i, \cdots \phi_k) + E_{minl}(\phi_i, \cdots \phi_k) + E_{nonover}(\phi_i, \cdots \phi_k) \tag{3}$$

where

$$E_{minv}(\phi_i, \cdots \phi_k) = \sum_{i=1}^{k} e_i \int_{\Omega} H_\alpha(\phi_i)(1 - c_i)^2 \, dxdy$$

$$E_{minl}(\phi_i, \cdots \phi_k) = \sum_{i=1}^{k} \gamma_i \int_{\Omega} \delta_\alpha(\phi_i) |\Delta \phi_i| \, dxdy$$

$$E_{nonover}(\phi_i, \cdots \phi_k) = \frac{\lambda}{2} \int_{\Omega} (\sum_{i=1}^{k} H_\alpha(\phi_i) - 1)^2 \, dxdy$$

$$\forall i, e_i, \gamma_i, \lambda \in \mathbb{R},$$

The evolution of the level function ϕ_i is shown in Equation (4).

$$\phi_i^{t+1} - \phi_i^t = \Delta t \cdot \delta_\alpha(\phi_i^t) \cdot \left[v_i div\left(\frac{\Delta \phi_i}{|\Delta \phi_i|}\right) - e_i(I - c_i)^2 - \beta\left(\sum_{i=1} kH_\alpha(\phi_i) - 1\right)^2 \right] \tag{4}$$

In Equation (4), the $H_\alpha(\cdot)$ is the *Heaviside Function*, c_i are the means of positive areas in level set function ϕ_k, and $\delta_\alpha(\cdot)$ is the Dirac delta function and v_i and β are constants.

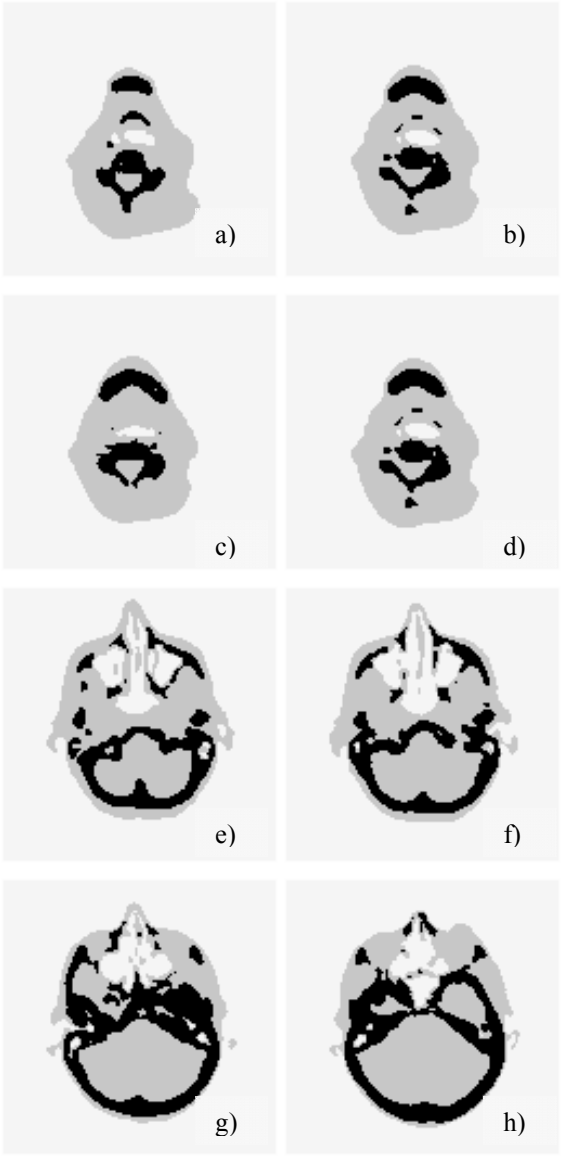

Figure 2: Region partition results. The segmentation phase divides the data into three homogenous regions, which are shown in white, gray and black. a) to d) are segmentation result for the CT-neck data; and e) to h) are for the CT-head data, shown in slices for easier identification.

After performing the level set segmentation, we divide the data into different homogeneous regions (see Fig. 2), and label all vertices located on the interior boundaries of homogeneous regions as feature vertices; and all vertices located inside homogeneous regions as normal vertices (see Fig. 3).

Region Based Feature Preserving Simplification Algorithms

2.2.1 Definitions

Before we begin the discussion of the feature preserving simplification algorithm, we introduce some definitions which we will need later.

- Neighbor Sets

  For a tetrahedral cell τ, $A(\tau)$ defines a set of cells which share one and only one vertex with τ; $D(\tau)$ defines a set of cells which share at least two vertices with τ. We denote the union of $A(\tau)$ and $D(\tau)$ as *Neighbor*(τ), the neighbor set of cell τ.

- Normal Cell

  If all vertices of a cell τ are normal vertices and they are all located in same sub-region, we denote the cell τ as a normal cell.

- Feature Cell

  If a cell contains one or more feature vertex, we label it as a feature cell.

- Cross-Region Cell

  If a cell has vertices in more than one sub-region, but it is not a feature cell, we label it as a cross region cell.

- Cross-Region Neighbor Cell

  If a cell is a normal cell, and it has a neighbor cell which is a cross region cell, we label it as a cross region neighbor cell.

- Boundary Cell

  If a cell contains at least one vertex on the external boundary, we label it as a boundary cell.

2.2.2 Feature Preserving Rules

After feature detection, a number of rules are defined to preserve both interior and exterior features.

1. Normal Cell: The normal cells may be collapsed freely. Here we define the collapse operation for normal cells as τ to $\boldsymbol{v}_c$, where $\boldsymbol{v}_c$ is the centroid of the cell. By choosing the centroid, the volume of the deleted cell is distributed evenly amongst the remaining local neighboring cells.
2. Feature Cells: If any feature cell τ has more than one feature vertex, we do not perform the collapse operation on it. If it has exactly one feature vertex $\boldsymbol{v}_f$, we define the collapse operation as collapsing τ to $\boldsymbol{v}_f$.
3. Cross-region Cell: For any cross-region cell τ, to achieve the decimation of different sub-regions individually, we do not perform the collapse operation on it.
4. Cross-region Neighbor Cells: For any cell τ, if its $A(\tau)$ or $D(\tau)$ contains only one cross-region cell, doing a collapse operation to the centroid will cause the change of the structure of the sub-region. To avoid this problem, we define the collapse operation for such a cell as the collapse of τ to the vertex $\boldsymbol{v}_n$ which is the common vertex with the cross-region cell. If its $A(\tau)$ or $D(\tau)$ contains more than one feature cell and more than one feature vertex, we do not collapse the cell. Cells in $D(\tau)$ may contain any number of feature points.

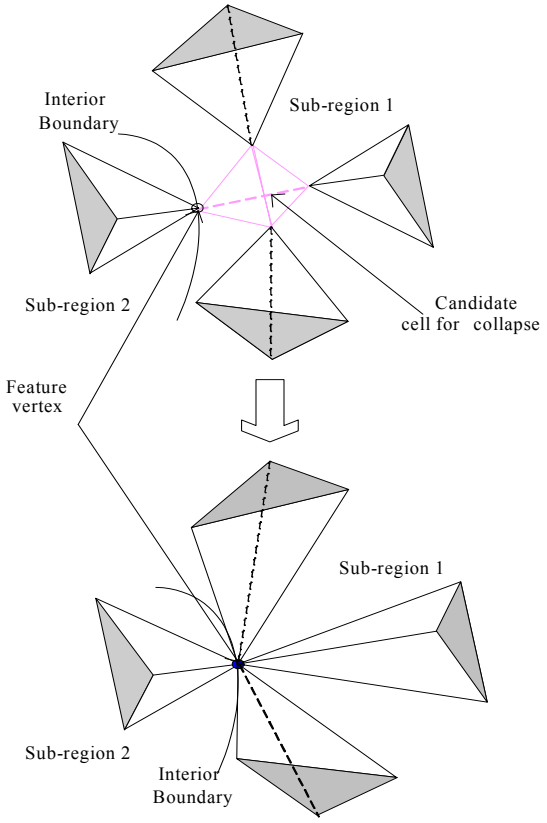

Figure 3: The operation τ to $\boldsymbol{v}_f$. The candidate cell, which is in center, is a feature cell. Its feature vertex is labeled by $\boldsymbol{v}_f$. As rule 1 describes, we collapse the cell τ into its feature vertex $\boldsymbol{v}_f$.

The above tests are defined so as to ensure that there is no large change in interior boundaries of sub-regions. In some cases, the geometry of the exterior boundary should also be preserved. Therefore, we define a boundary cell check as follows:

5. Boundary Cells: If the boundary cell τ has exactly one vertex on the boundary, we define the decimation operation as a collapse of τ to $\boldsymbol{v_b}$ (the vertex on the exterior boundary). We denote the operation as τ to $\boldsymbol{v_b}$. If it has more than one vertex on the boundary, we do not perform the collapse operation on it.

We also define a flipping check to ensure no flipping occurs during the simplification process.

6. Flipping Check: For any cell inside the $A(\tau)$, where $\boldsymbol{v_o}$ moves to $\boldsymbol{v'_o}$, $\boldsymbol{v'_o}$ should stay on the same side of its facing triangle, see Fig. 4. We test all the cells in $A(\tau)$ and check whether its signed volume has the same sign (positive or negative) before and after the collapse operation. If any cell changes the sign of its volume, we do not allow the collapse operation and return the dataset to the previous state before the collapse.

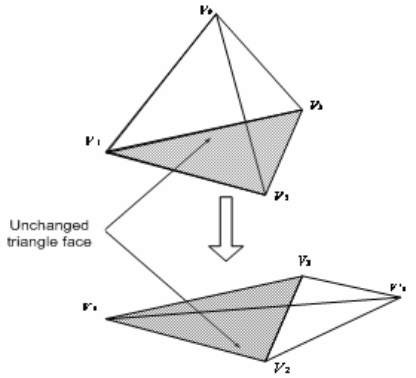

Figure 4: The flipping problem. After the collapse operation, it is possible for some cell that the moving vertex v_o goes to the other side of the unchanged triangle face $v_1v_2v_3$.

2.2.3 Error Prediction

We use a combination of regional and local error prediction functions to choose the cell to be collapsed next.

2.2.3.1 Regional Error Prediction Function

Because each sub-region has homogenous field value distribution within it, we expect our error prediction function to reflect the modification of distribution affected by decimation operation. Therefore, we forbid any large change of the homogenous distribution. We suppose the mean value of the attributes in each homogenous sub-region will not change. The cell, whose distribution is close to the mean of the region, has more priority to be chosen as the collapse candidate.

We define the regional error prediction function as shown in Equation (5):

$$\varepsilon_r = \frac{\delta_c}{\delta_N}, \qquad \delta_N = \frac{1}{N}\sum_{i=0}^{N}(s_i - \bar{s})^2, \qquad \delta_c = \frac{1}{4}\sum_{i=0}^{3}(s_i - \bar{s})^2 \tag{5}$$

Where s_i is the attributes value of original vertices of collapse cell, $\bar{s}$ is the mean of distribution of region N, δ_N is the variance of distribution of region N, and δ_c is the variance of cell τ.

2.2.3.2 Local Error Prediction Function

A large gradient change means there is a possible feature [Cho02a, Chi03a, Kin98a, Kau93a, Sta98a]. We define the gradient error prediction function as in Equation (6):

$$\varepsilon_g = \frac{1}{4}\sum_{i=0}^{3} | s_i - s_n^{'} | \tag{6}$$

Here s_i is the attribute values of the original vertices of the collapse cell, and $s_n^{'}$ is the attribute value of new vertex which is created after the collapse operation.

Also, we do not want the decimation operation to change the attributes distribution of candidate's neighbor sets very much. Therefore, we define three measurements for the aspect ratio of cell as the form in Equation (7), where v_i is the original vertex of cell τ; v_n is the new vertex generated after collapse operation; a, b and c represent the three edges of cell τ which share one common vertex:

$$\varepsilon_n = \frac{volume(\tau)}{\sum_{\tau_i \in A(\tau)\cup D(\tau)} volume(\tau_i)} \tag{7}$$

$$\varepsilon_s = \frac{1}{4}\sum_{i=0}^{3} | v_i - v_n |$$

$$\varepsilon_v = \frac{1}{3!} | a \cdot (b \times c) |$$

The final error prediction function is the combination of regional and local functions, shown in Equation (8):

$$\Delta\varepsilon = \omega_r\varepsilon_r + \omega_g\varepsilon_g + \omega_s\varepsilon_s + \omega_v\varepsilon_v + \omega_n\varepsilon_n \tag{8}$$

2.2.4 Greedy Decimation Based Simplification

To achieve fast processing speed, we choose the greedy method to implement our algorithm.

1. Compute the predicted error based on the error prediction function shown in Equation (8) for all tetrahedral cells, and arrange them into a priority queue, ordered by predicted error.
2. While there is still at least one cell remaining in the priority queue, with the predicted error less

than the user specified tolerance, pick the first cell τ, and do following:

a. Delete τ from the priority queue.

b. Determine $A(\tau)$ and $D(\tau)$ of the cell τ.

c. Check the type of the current cell τ: feature cell, cross-region cell, cross-region neighbor cell, boundary cell or normal cell, as described in Section (2.1.2).

d. If the cell is a cross-region cell, or has two or more feature vertices, skip step 2e, 2f and 2g. Otherwise, perform the collapse operation based on feature preservation rules 1, 2, 3 described in Section (2.1.2).

e. Carry out the flipping check (feature preservation rule 6) among the cells in $A(\tau)$. If the collapse operation causes any flipping problem, then recover the original vertices of τ and skip steps 2f and 2g.

f. Delete all cells in $D(\tau)$ from the priority queue, and update cell-vertex index.

g. Remove all cells in $A(\tau)$ from the priority queue, and label it to forbid any further selection of collapse candidates.

3. Save the result.

3. EXPERIMENTAL RESULTS

We have implemented our algorithm on a Windows Platform with a 2.39GHz Intel Pentium 4 CPU and NVIDIA Quadro4900 XGL adapter with 128 Meg RAM. We use ZSweep [Far00a] as our irregular mesh rendering technique to obtain the final images. We have tested our algorithm on several medical datasets. Below we provide the results for dataset CT-head with the resolution 128x128x53, and CT-neck with the resolution 128x128x13. Results for other datasets show similar performance statistics.

Data	Number of cells	Number of vertices	Number of feature vertices
CT-head	4,193,540	868,352	140,233
CT-neck	967,740	212,992	27,528

Table 1 Detailed information of data sets

Data	ω_r	ω_g	ω_s	ω_v	ω_n
CT-head	10	2	1	1	0.5
CT-neck	10	2	1	1	0.8

Table 2 Parameters used in simplification algorithm.

Table (1) shows the detailed information of our testing data. Table (2) shows the parameters we have used in error prediction function to perform our simplification algorithm.

Fig 5 is the comparison of our method with the simplification algorithm, TetFusion [Cho02a]. It clearly shows that for the same decimation rate, our method is better than the algorithm that uses only local error measurement. The horizontal line shows the number of cells that has been decimated. The vertical line shows the average difference of color defined, in form of Equation (9) defined on average of difference in image color per pixel, of the rendering result with the original one, where n is the number of pixel.

$$diff_1 = \frac{\sum_{i=0}^{n}\sqrt{(r_{i0}-r_{i1})^2+(g_{i0}-g_{i1})^2+(b_{i0}-b_{i1})^2}}{n} \qquad (9)$$

With increasing decimation numbers, we draw two plots. The upper one is obtained by TetFusion method, which is based on local measurement, and the lower one is obtained by our method, which has a regional based measurement. As it is shown clearly, for the same number of decimation cells, the color difference of our method is less than the one of TetFusion. That shows our method maintains more volume interior information than TetFusion does.

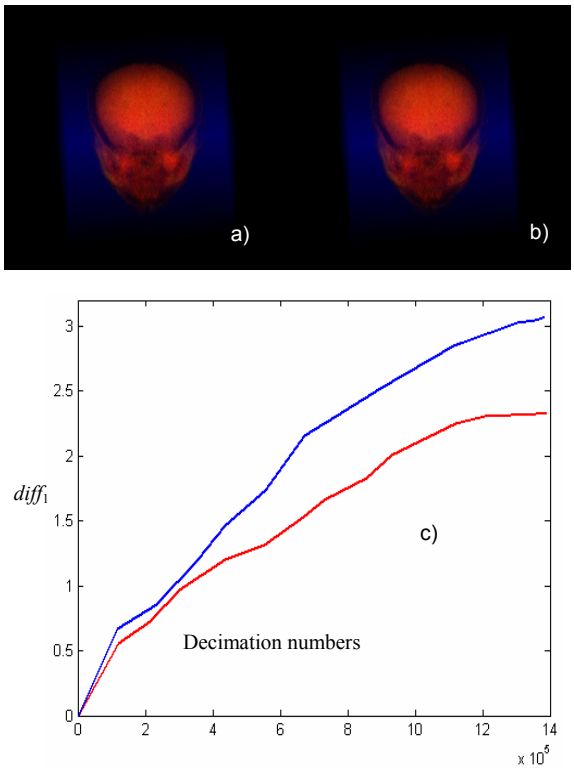

Figure 5: Demonstration that our feature preserving simplification algorithm is an improvement of previous approaches. a) is generated by our method, and b) is generated by TetFusion [Cho02a]. c) demonstrates the average color comparison as the form in Equation (9). The upper plot shows the $diff_1$ of TetFusion, and the lower one shows our method.

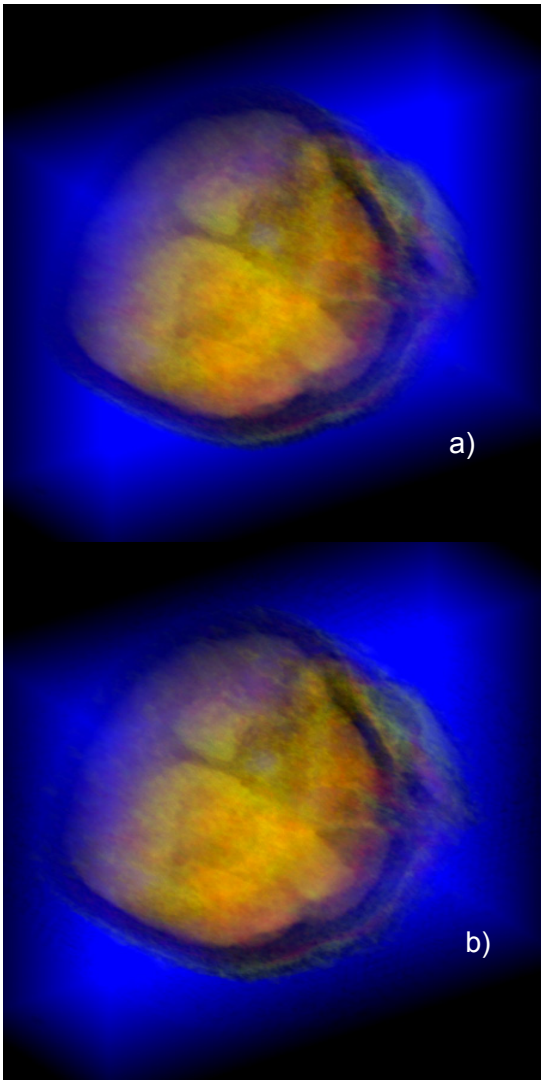

Figure 6: The rendering result of CT head data. a) The original image; b) The 50% simplified result.

Fig. 6 shows that after applying our simplification algorithm, the resultant dataset contains half size of original one. However, the direct rendering result is very promising. The half size dataset maintains nearly the same information of the original one. We can clearly identify the structure of original data.

Fig. 7 shows another experimental result with a numerical comparison between the original image with simplified one. The value, shown in Fig. 7-(c), demonstrates the total error ($diff_2$) per pixel between Fig. 7-(a) and (b) as defined by Equation (10), which is defined on difference in image color per pixel. The color bar shows the range of the difference.

$$diff_2 = \sqrt{(r_{i0} - r_{i1})^2 + (g_{i0} - g_{i1})^2 + (b_{i0} - b_{i1})^2} \quad (10)$$

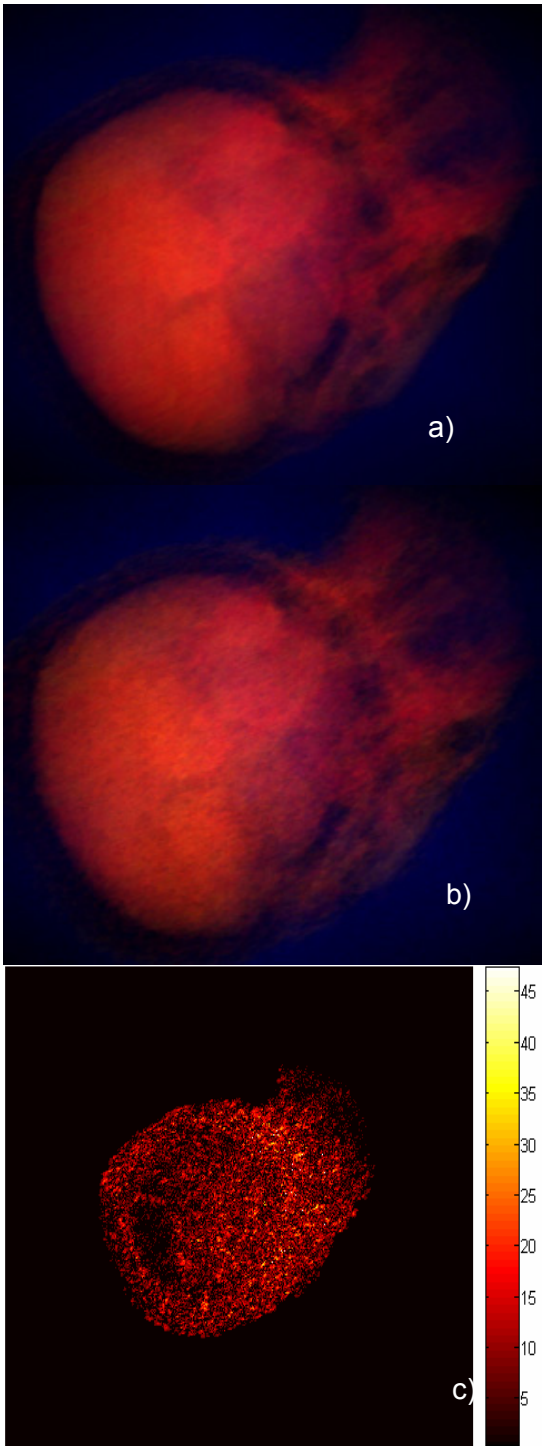

Figure 7: Demonstration of the simplification results on CT-Head data. a) is the image rendered by original data. b) is the image rendered by 48% simplified data. c) is a comparison image by analyzing the color difference between a) and b) pixelwise. The value, shown in c), is obtained by the Equation 9. The color bar represents the range of the difference.

4. CONCLUSION

In this paper, we have proposed a new simplification algorithm to reduce the large amount of redundancy of 3D medical datasets. A cutting edge technique of image processing, namely level set segmentation, is applied in a preprocess phase to simplify the data to achieve a noise robust region partition of volume data. We convert the regular grid data into a tetrahedral representation and then simplify the data while preserving both regional and local features. Our final result is a simplified tetrahedral mesh, which can be further analyzed, visualized, and animated. The experimental results amply show the advantages of this approach.

5. ACKNOWLEDGMENTS

We thank Dr. R. Farias, Computer Science Dept., Mississippi State University, Dr. Joseph S.B. Mitchell, Department of Applied Mathematics & Statistics, and Dr. C. T. Silva, School of Computing, University of Utah, for providing us the ZSweep code [Far00a]. We thank the maintainers of the web page www.volvis.org for the CT head, CT-neck, and other medical datasets used in our testing of the algorithm.

6. REFERENCES

[Chi03a] Chiang, Y. J. and Lu, X., "Progressive Simplification of Tetrahedral Meshes Preserving all Isosurface Topologies", EG'03, conf. proc. Granada, Spain, Spring Press, pp.493-504, 2003

[Cho02a] Chopra, P. and Meyer, J., "Tetfusion: An Algorithm for Rapid Tetrahedral Mesh Simplification", IEEE VIS'02, conf. proc. Boston, MA, USA, IEEE Computer Society, pp.133-140, 2002

[Cig00a] Cignoni, P., Costanza, C., Montani, C., Rocchin, C. and Scopigno, R. "Simplification of Tetrahedral Meshes with Accurate Error Evaluation", in IEEE VIS'00 conf. proc. Salt Lake City, UT, USA, IEEE Computer Society, pp. 85-92, 2000

[Far00a] Farias, R., Mitchell, J. S.B., and Silva, C. T., "Zsweep: An Efficient and Exact Projection Algorithm for Unstructured Volume Rendering", IEEE VolVis'00, symposium, Salt Lake City, UT, USA, IEEE Computer Society, pp. 91-99, 2000.

[Gel99a] Gelder, A. V., Verma, V. and Wilhelms, J., "Volume Decimation of Irregular Tetrahedral Grids", Journal of Computer Graphics International, pp 222-230, 1999.

[Ger00a] Gerstner, T., and Pajarola, R., "Topology Preserving and Controlled Topology Simplifying Multiresolution Isosurface Extraction", IEEE VIS'00, conf. proc. Salt Lake City, UT, USA, pp.259-266. IEEE Computer Society Press, 2000.

[Hong03a] Hong, W. and Kaufman, A. E., "Feature Preserved Volume Simplification". ACM SMA'03, sym. proc. Karlsruhe, Germany, ACM Press, pp. 334-339. 2003.

[Hop96a] Hoppe, H., "Progressive Meshes". ACM SIGGRAPH'96, conf. proc. New Orleans, LO, USA, pp 99-108. ACM Press, 1996.

[Kau91a] Kaufman, A. E., "3D Volume Visualization", EG'00, conf. tutorial, Interlaken, Switzerland, Springer Press, pp175-203, 1991.

[Kau93a] Kaufman, A. E., Cohen, D. and Yagel, R., "Volume Graphics". Journal Computer, 26(7):51-64, 1993.

[Nie00a] Nielson, G. M., "Volume Modeling", Volume Modeling. In: M. Chen et al. (eds.), Volume Graphics, Springer Press, pp. 29-48, 2000.

[Osh88a] Osher, S. and Sethian, J. A., "Fronts Propagating with Curvature-dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations", Journal Comput. Phys. 79(1):12-49, 1988.

[Sam00a] Samson, C., Feraud, L. B., Aubert, G. and Zerubia, J., "A Level Set Model for Image Classification", Journal Computer Vision, Volume 40, Issue 3, pp 187-197, 2000.

[Sta98a] Staadt, O. G. and Gross, M. H., "Progressive Tetrahedralizations", IEEE VIS'98, conf. proc., Research Triangle Park, NC, USA, IEEE Computer Society Press, pp 397-402, 1998.

[Tro98a] Trotts, I. J., Hamann, B., Joy, K. I. and Wiley, D. F., "Simplification of Tetrahedral Meshes", IEEE VIS'98, conf. proc., Research Triangle Park, NC, USA, IEEE Computer Society Press, pp. 287-295, 1998.

[Tro99a] Trotts, I. J., Hamann, B., Joy, K. I. and Wiley, D. F., "Simplification of Tetrahedral Meshes with Error Bounds", Journal IEEE TVCG, vol 5(3):224-237, 1999.

[Web03a] Weber, G. H., Scheuermann, G. and Hamann, B., "Detecting Critical Regions in Scalar Fields", VisSym'03, Sym. Proc., Grenoble, France, Eurographics Association, pp. 85-94, 2003.

[Yao00a] Yao, J. H., and Taylor, R. H., "Tetrahedral Mesh Modeling of Density Data for Anatomical Altases and Intensity-Based Registration", MICCAI'00, conf. proc. Pittsburgh, PA, USA, Springer Press, pp. 531-540, 2000