A New 3-D Mesh Simplification Algorithm

Wu Xianfeng[1] Ye Jianhui[2]

Abstract

To simplify the 3D color head mesh ,it is more important to keep the boundary and quality of the head's sense organs including eyes, eyebrows, nose and mouth. In this paper, we present a novel mesh simplification algorithm based on region segmentation. The algorithm can be divided into two stages: segmentation and simplification. After the automatic segmentation of 3D color head mesh into different head parts, vertices are classed into region-boundary vertices and region-inner vertices. Using iterative edge collapse and region-weighted error metric, the algorithm generates continuous levels of detail (LOD). Results of several experiments are shown, demonstrating the validity and efficiency of our method.

Keywords: *mesh simplification, level of detail, image segmentation, multi-resolution model*

1 Introduction

With the development of computer technology, the complexity of the objects involved in modern virtual reality simulations and computer animations increases every day. Texture and lighting information is also necessary to produce more realistic rendering. On the other hand, the computation and storage requirements for such applications far exceed the capacity of modern graphics hardware system.

There are many approaches to acquire 3D models in a virtual world. Laser scanning, which uses active optical triangulation technique, is one of the most common methods for acquiring range data. Its speed and accuracy has increased dramatically in recent years with the development of geometrically stable imaging sensors such as CCD's and high-precision mechanical parts. We have developed a new kind of 3D laser color scanning system that produces color dataset of 3D models as well as range data. The triangle's number of a typical human head's 3d data is half million.

Because such large meshes are difficult to store, transmit, and render, many techniques have been developed for geometrically simplifying them [1]. However the majority of them are intended for accurate free-noise mesh, and the models are always geometric objects, not high-resolution human face model.

[1] Image Processing & Intelligent Control Key Laboratory of China, Institute of Pattern Recognition & Artificial Intelligence, Huazhong University of Science & Technology, Wuhan , 430074, China
Email: Summit.wu@263.net

[2] Image Processing & Intelligent Control Key Laboratory of China, IPR&AI, Huazhong University of Science & Technology, Wuhan , 430074, China
Email: echoyf@263.net

In this paper, we develop a new region-based mesh simplification algorithm for simplifying human head meshes with attributes, which offers the following advantages:

1. The output mesh generated of the algorithms we have presented has the property that its set of vertices is a subset of the set of vertices of the original mesh. As no new vertex is introduced, we can also use the original model's texture image.
2. It can preserve well the details of human head; yet simplify others more aggressively such as hair, face and neck. That is, it can preserve the region of interest (ROI) with more details.
3. It is not sensitive to noise introduced by the laser scanner system.
4. This algorithm can be also applied in other objects for preserving the details.

The remainder of this document is organized as follows. In section 2, we describe the most common algorithms for the simplification proceeds. Section 3 describes our new algorithm. The experiments and conclusions are presented in section 4.

2 Related Work

In recent years, so many algorithms have been formulated for simplifying mesh surfaces. Those algorithms can be categorized into the classes as follows:

- **Incremental methods based on local updates.** These methods perform the simplification as a sequence of local updates. Each update reduces mesh size and decreases the approximation precision. The updates cannot complete until the desired result is obtained. Some notable examples of such algorithms include Schroeder[3], Hoppe[4], Ganland[5]. Schroeder iteratively removes the vertex, as well as all adjacent faces, then retriangulations the result hole. Hoppe uses the Edge collapsing method to simplify the geometry. The edge collapse operation is attractive because it is a more atomic operation than vertex or face removal, and does not require the invocation of a triangulation algorithm. Although this algorithm performs successful in many objects , and is available in Microsoft DirectX Interface, it can not join unconnected regions. Garland develops a simplification method by quadric error metrics. The method is. Based on incremental edge-collapsing,. meanwhile it can also collapse vertex couples which are not connected. This algorithm performs efficiently in the objects with close boundary, but to the open boundary , it produces undesirable results.
- **Coplanar facets merging.** Coplanar or nearly coplanar facets are searched for in the mesh, merged into larger polygons, and then retriangulated into fewer facets than those originally required [6]; face merging is driven by a co-planarity test. The superfaces method [7] extends this approach by providing bounded approximations and more robust retriangulations of the merged faces;
- **Vertex clustering**. Rossignac[8] places a bounding box around the original model and divides it into grids. The vertices are clustered into one vertex if they concentrate in the same cell. Then the mesh faces are updated accordingly. This method executes very quickly, and can make drastic topological alterations to the model. Unfortunately, the quality of the output is often quite low. In addition, it is difficult to build an approximation with a specific face count , since the number of faces is only indirectly determined by the specified grid dimensions.
- **Wavelet-based approaches.** The wavelet decomposition approach seems very promising for surface simplification. Conversely, a regular, hierarchical decomposition is required to support wavelet decomposition, and computational efficiency is not at the best. Wavelet approaches have been proposed to manage regularly gridded meshes or more generic meshes [10] [11].

In particular, the multi-resolution analysis approach is based on a three-phase process (re–meshing, re-sampling and wavelet parametrization) to build a multi-resolution representation of the surface, from which any approximated representation can be extracted . An extension to this approach has recently been proposed to manage the approximation of both geometry and surface color [11].

– **Detail preserving methods**. All the methods above are mainly to deal with the simplification of the surface position. To a wide range of applications, the details are necessarily to be preserved, such as surface color, surface curvature, surface texture.
Hoppe [12] expands his progressive meshes algorithm to maintain the color and texture attributes by using a multi-variate error evaluation function including geometry and color information. Bajaj and chikore[12] presents their simplification algorithm with associated scalar fields to within a given tolerance. Cignoni[13] preserves the details with a post-processing phase after the general mesh-simplification. Hoppe [14] expands his progressive meshes algorithm to maintain the color and texture attributes by using a multi-variant error evaluation function including geometry and color information.

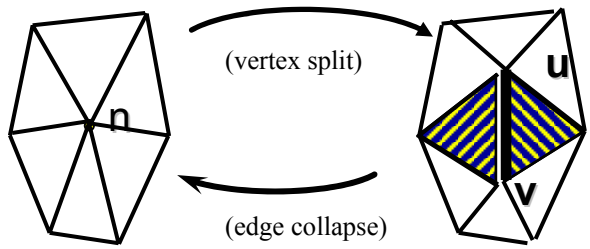


Fig.1: Edge collapse and edge split mesh simplification

Our method is one of the detail-preserving methods, it bases on the edge collapse methods(Fig1.). An edge collapse is a local mesh operation that takes two adjacent vertices u and v of the mesh and unifies them into one new vertex n, removing two faces in the process. Note that the position of the new unified vertex n can be arbitrary. We use the cost function to seek such collapsed-edges. Since the cost function presented in Progressive mesh is complicated, we use a new cost function initially described by Melax[15]. It can efficiently simplify the surfaces data , and preserve the details perfectly. Moreover, it is much simpler than other methods presented in [13][14].

To our practical applications, mainly the simplified object is 3D color human head mesh, so it has some special requirements for simplified model:

1. **The output mesh's vertices are the subset of the original mesh's vertices**. As no new vertex is introduced, we can also use the original model's texture map and other information for simplified model. The output mesh generated of Progressive Mesh contains new vertices, because new positions are selected for each edge collapsing action.

Pedro V. and co-workers presented a scheme "Texture Mapping Progressive Meshes" for defining a texture atlas parametrization over the PM representation of an arbitrary mesh [4]. This scheme permits the same texture image(s)to be used for all LOD mesh approximations in the PM sequence. But the scheme is too elaborate to implement.

2. **It can preserve head organs boundary**. In practice, the human eye is too familiar with human face. A slight distortion or dissymmetry of the sense organs will cause a great

noticeable visual error. In practice, the head organs boundary is adequately describe a human head. Many published algorithms do not use such knowledge. An algorithm using semantic and knowledge simplification is more helpful and important. For example, if the algorithm had the knowledge that a sharp edge was in fact a topological circle, it might be able to change into something simpler, like a pentagon. It is the same for the head parts.

3. **Controllable simplifications can be implemented**. It is sometimes necessary to let the amount of simplification vary across the mesh, in order to preserve some parts, yet simplify others more aggressively. This kind of adaptivity involves mainly visual or semantic considerations that may be done a priori and are controlled by the user. Our method can resolve this situation well by arrange different weights.

4. **The algorithm is not sensitive to error position of some vertices**. To simplify the 3D laser scanned model, the algorithm should take into account the possible sources of error inherent in scanner. Two sources of error are particularly relevant to range accuracy and results in a false position [5]. The error cannot be totally corrected in realistic scanner system. So we must have some methods to constrain the error, the region boundary is a good choice. The error is constrained by the region boundary and will not spread.

3 A Description of the Algorithm

3.1 Overview

Our surface simplification algorithm consists of two stages. In the first stage we class the vertices into two type: one is region-boundary vertex, which can only collapse to region-boundary vertex; the other is region-inner vertex, which can collapse to region-boundary vertex and region-inner vertex. In the second stage we use edge collapse operator to simplify the model. For the sake of simplicity and computationally efficiency, after each edge collapse (u; v)->n, the new vertex n is just the vertex v, not assigned a new position.

The basic steps of the algorithm are these:

Step 1. 3D head mesh is preprocessed: smoothing, correcting error of vertex position.

Step 2. Based on color, texture and curvature of vertex, we class the mesh into regions and class the vertices into two types: region-boundary and region-inner vertices.

Step 3. Sort the vertices using the least cost of simplification. This cost is measured using the error metric function, see Section 3.3.

Step 4. Apply the edge collapse operator for the vertex at the head of the list and record the corresponding vertex split in the progressive mesh structure including color, texture and normal information.

Step 5. Re-compute the cost for the vertices that have been affected by the operator and reorder the list.

Step 6. If the list is empty or the number of the vertices is reduced to the desired count, the algorithm terminates. Otherwise, return to step 3.

3.2 Partition Mesh into Regions

3D color head mesh scanned in the $m \times n$ size (m denote the columns, n denote row number) is represented in terms of the matrix:

$$M = \begin{bmatrix} (x_{11}, y_{11}, z_{11}, R_{11}, G_{11}, B_{11}) & & \cdots \\ & (x_{ij}, y_{ij}, z_{ij}, R_{ij}, G_{ij}, B_{ij}) & \\ \vdots & & (x_{mn}, y_{mn}, z_{mn}, R_{mn}, G_{mn}, B_{mn}) \end{bmatrix}, \quad (1)$$

where (x_{ij}, y_{ij}, z_{ij}) is the position of the vertex, (R_{ij}, G_{ij}, B_{ij}) is the color of the vertex. From the above mesh data structure, it is easy to get the texture image of the 3D mesh. Just omitting the 3D poison coordinate dimension, we get the texture image in the form of array. Fig2 is the texture image of a head.

Fig.2 The texture image of scanned data

The texture image is partitioned with edge-based segmentation algorithm. That is, we detect local discontinuities (the color features and the local curvature) and group them to form object boundaries between regions.

After experimenting with different color spaces, e.g. YUV, RGB, HSV, the YUV color space gave the best results for our test images. The luminance is taken into consideration to aid in the segmentation of those areas where the chrominance is not well defined, like for example the hair. The thresholds for each component are kept fixed.

Color information is not sufficient to separate nose and face, for their color is too close. The vertex curvature $\overline{C_u}$ is based on comparing vertex normal with nearby triangles. First we compute the normal vector $\overline{n_f}$. The vertex normal vector is defined as

$$\overline{n_u} = \frac{\sum_{f \in T_u} S * \overline{n_f}}{\sum_{f \in T_u} S} \tag{2}$$

where Tu is the set of triangles that contain vertex u, S is the triangle's area, $\overline{n_f}$ is the triangle's normal vector.

The $\overline{C_u}$ is the maximum normal difference.

$$\overline{C_u} = \max_{f \in T_u} \left\{ (1 - \overline{n_f} * \overline{n_u}) / 2 \right\} \tag{3}$$

Zero curvature means the surface is flat.

Median filtering is the post processing operations that are performed after the initial point extraction stage. The median operation is introduced in order to smooth the segmented object silhouettes and also eliminate any isolated misclassified pixels that may appear as impulsive-type noise. Square filter windows of size 5x5 and 7x7 provide a good balance between adequate noise suppression and sufficient detail preservation.

Results of the initial region partition, and the subsequent boundary are shown in Fig.3(a) left and Fig.4(a) left. Note that region boundaries align with important features on the face mesh.

3. 3 Error Metric

Once the mesh is partitioned into regions, we class the vertices into two types: one is region-boundary vertex, which compose of the region boundary and the mesh boundary; the other is region-inner vertex.

It makes sense to get rid of small details first. Based on these heuristics, we define the cost of merging an edge as the length of the edge multiplied by a curvature term [6]. The error cost to merge u into v at each step is defined as follows:

$$\cos t(u,v) = \| u - v \| \times \max_{f \in Tu} \{ \min_{n \in Tuv} \{ (1 - \overline{n_f} * \overline{n_u}) \div 2 \} \} \times W(u) \tag{4}$$

where Tu is the set of triangles that contain vertex u, Tuv is the set of triangles that contain both vertex u and v, $\overline{n_f}$ is the normal vector of the triangle, $W(u)$ is the region-weighted function. $(1 - \overline{n_f} * \overline{n_u}) \div 2$ is for normalization.

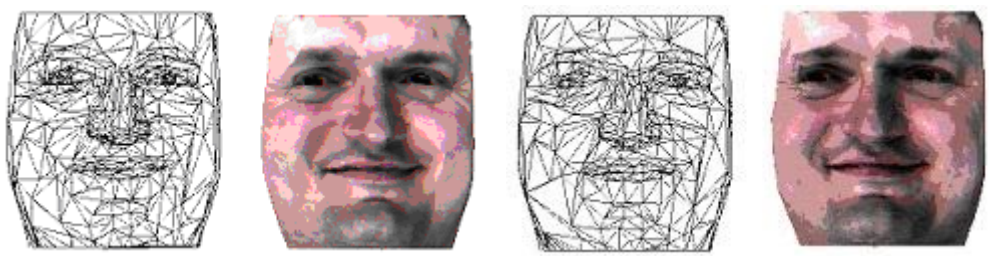
From the Equation (4), we can see that fewer polygons are needed to represent nearly coplanar surfaces while areas of high curvature need more polygons. The curvature term for

merging an edge uv is determined by comparing dot products of face normal in order to find the triangle adjacent to u that faces furthest away from the other triangles that are along uv. Note that the cost of merging vertex u into v may be different than the cost of merging v into u.

4 Results and Conclusions

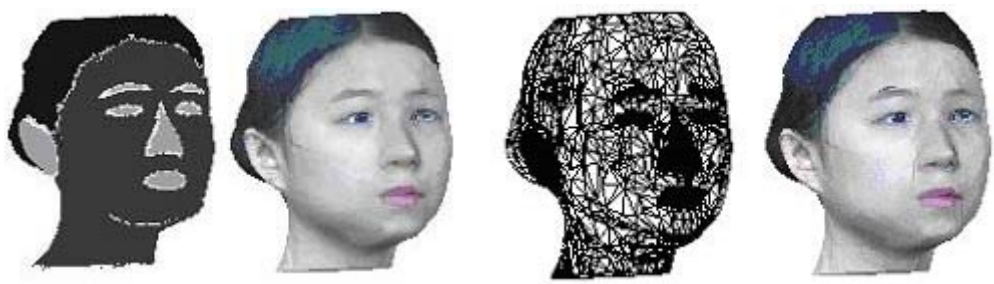
4.1 Results

3(a): original model (26,460 vertices) 3(b): simplified model (2,646 vertices)

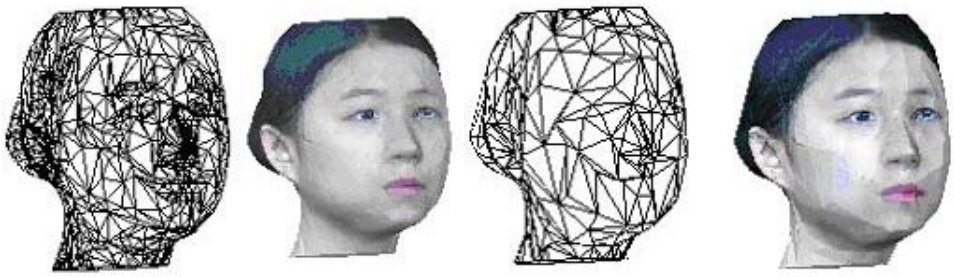


3(c): simplified model (1,000 vertices) 3(d): simplified model (500 vertices)

Fig.3: The LoDs of the face



4(a): original model (67,918 vertices) 4(b): simplified model (6.791 vertices)



4(c): original model (1,000 vertices) 4(d): simplified model (500 vertices)

4(e)simplified model by decimation method(1000 vertices)

Fig.4: The LoDs of the head

The proposed algorithm has been implemented in our 3D color laser scanner and the resulted simplified models can be post processed by Maya, AutoCad software. The models look reasonably good given the number of polygons used in each image. Fig.3 shows the LoDs of a face mesh containing 26,460 vertices. Fig.4 shows the LoDs of a complete 3D head mesh containing 67,918 vertices. In Fig.3(a) and Fig.4(a), left is the resulted region image, right is textured rendering image of original model. In Fig.3(b)(c)(d) and Fig.4(b)(c)(d), left is the wireframe, right is textured rendering image. Reductions to 1000 (c) and 500 (d) vertices are shown. Notice that the Fig.4 mesh has much noisy, but the simplified mesh are relatively good in keeping the head organs boundary and detail.

To compare with other methods, we use the decimation method [3] to simplify the girl head, Fig4(e) is the result with this algorithm ,the number of the vertex is 1000, same as Fig.4(c). Obviously, the result is unacceptable.

4.2 Conclusions

We have described a novel simplification algorithm which partitions a given surface into regions and use region-weighted simplification. The algorithm permits the same texture image to be used for all LOD mesh approximations and resistant to noise, while implement Controllable Simplification. In addition, the algorithm is easily implemented and applied.

Acknowledgement

This work is supported by National Science Foundation of China (69775022)

References

[1] P.Heckbert and M.Garland. Survey of polygonal surface simplification algorithms,. In Multiresolution Surface Modeling Course Notes. *ACM SIGGRAPH*, 1997.

[2] Clark . Hierarchical geometric models for visible surface algorithms, *Communications. of the ACM*, 19:547-554, 1976.

[3] W.Schroeder, J.Zarge, and W. Lorensen. Decimation of triangle meshes,. *ACM SIGGRAPH92*, 26: 65–70, 1992.

[4] H.Hoppe Progressive meshes, *ACM SIGGRAPH96*, 30:99-108.1996

[5] M.Garland and P. Heckbert. Surface simplification using quadric error metrics, *ACM Siggraph'97*, 209-216,1997

[6] A.D. Kalvin, C.B. Cutting, B. Haddad, and M.E. Noz. Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures, *SPIE Vol. 1445 Image Processing*, 247–259, 1991.

[7] D. Kalvin and R.H. Taylor. Superfaces: Poligonal mesh simplification with bounded error. *IEEE G.&A.*,16(3):64–77, 1996.

[8] J. Rossignac and P. Borrel Multi-resolution 3D approximation for rendering complex scenes. *Geometric Modeling in Computer Graphics*, 455–465. 1993.

[9] M. Reddy Perceptually-driven polygon reduction. *Computer Graphics Forum*, 15(4):191–203, 1997

[10] D.Hebert and H. Kim Image encoding with triangulation wavelets. *Proceedings SPIE*, (2569(1)):381–392,1995.

[11] A. Certain, J. Popovic, T. DeRose, T. Duchamp, D. Salesin, andW. Stuetzle. Interactive multiresolution surface *ACM (Siggraph '96)*, 91–98, 1996.

[12] C.Bajaj and D.Schikore error-bounded reduction of triangle meshes with multivariate data,. *SPIE, vol* 2656(34-45), 1996

[13] P. Cignoni, C. Montani, and R. Scopigno. A comparison of mesh simplification algorithms. Technical Report 97-08, Istituto CNUCE – C.N.R., Pisa, Italy, June 1997.

[14] Hoppe , Pedro V. Sander. Texture-Mapping Progressive Meshes, *SIGGRAPH 2001*, 409-416. 2001

[15] Stan Melax. A Simple, Fast, and Effective Polygon Reduction Algorithm, *Game Developer,* 1998, 11