Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Adaptive triangular mesh coarsening with centroidal Voronoi tessellations[*]

Zhen-yu SHU[†1,2], Guo-zhao WANG[†‡1], Chen-shi DONG[1]
([1]*Institute of Computer Graphics and Image Processing, Department of Mathematics, Zhejiang University, Hangzhou 310027, China*)
([2]*Laboratory of Information and Optimization Technologies, Ningbo Institute of Technology, Zhejiang University, Ningbo 315100, China*)
[†]E-mail: littlerain_szy@sohu.com; wgz@math.zju.edu.cn

Received Mar. 28, 2008; Revision accepted July 30, 2008; Crosschecked Feb. 9, 2009

Abstract: We present a novel algorithm for adaptive triangular mesh coarsening. The algorithm has two stages. First, the input triangular mesh is refined by iteratively applying the adaptive subdivision operator that performs a so-called red-green split. Second, the refined mesh is simplified by a clustering algorithm based on centroidal Voronoi tessellations (CVTs). The accuracy and good quality of the output triangular mesh are achieved by combining adaptive subdivision and the CVTs technique. Test results showed the mesh coarsening scheme to be robust and effective. Examples are shown that validate the method.

Key words: Triangular mesh, Mesh coarsening, Surface subdivision, Centroidal Voronoi tessellations (CVTs)
doi:10.1631/jzus.A0820229 **Document code:** A **CLC number:** TP31; O29

INTRODUCTION

Nowadays, large meshes are commonly used in numerous application areas including computer aided design, medical imaging and virtual reality. Large polygonal meshes are produced as piecewise linear approximations of 3D object surfaces. Using modern range scanning devices, it is possible to acquire models consisting of several million or even billion polygons. Irrespective of the domains of applications, large meshes require simplification to be handled at all. However, in finite element analysis, the quality of the triangular mesh is also important as it may affect the accuracy of the numerical results. Therefore, it is necessary to reduce the number of elements in the mesh to a reasonable level, while preserving the accuracy of the geometric approximation and the mesh quality.

In this paper, we present a new triangular mesh coarsening algorithm, which resamples the surface mesh to an adaptive mesh with much fewer elements than the original mesh. We combine the adaptive subdivision operator (Bank *et al.*, 1983; Vasilescu and Terzopoulos, 1992; Verfürth, 1994) with centroidal Voronoi tessellations (CVTs) (Du *et al.*, 1999) to obtain a more accurate approximation to the initial mesh and to output a remeshing surface with high quality. Adaptive subdivision splits those triangles that satisfy some prescribed criteria according to surface curvature, while other triangles remain coarse. This operator adapts the size of triangles in the mesh to the local curvature of the underlying surfaces. CVTs are Voronoi tessellations of a region such that the generating points of the tessellations are also the centroids of the corresponding Voronoi regions. Such tessellations are theoretically the optimal strategy for resampling (Du *et al.*, 1999). By combining the two techniques above, the algorithm can generate adaptive triangular meshes with high quality.

[‡] Corresponding author
[*] Project supported by the National Natural Science Foundation of China (No. 60773179) and the National Basic Research Program (973) of China (No. 2004CB318000)

PREVIOUS WORK

Over the last decade, an abundance of simplification algorithms has been proposed. One group of very useful algorithms (Schroeder *et al.*, 1992; Cohen *et al.*, 1996; Hoppe, 1996; El-Sana and Varshney, 1997; Garland and Heckbert, 1997; Kobbelt *et al.*, 1998; Lindstrom and Turk, 1998; Zelinka and Garland, 2002) is iterative in nature, applying small, local changes to the mesh, each of which removes a small set of triangles and replaces it with a different, smaller set of triangles. This approach is primarily concerned with approximation error and preserving topology and does not emphasize sampling or the quality of triangles. A detailed survey was reported by Garland (1999).

Another group of algorithms (Eck *et al.*, 1995; Lee *et al.*, 1998; Kobbelt *et al.*, 1999; Guskov *et al.*, 2000; Hormann *et al.*, 2001) is based on a special structure, the so-called 'subdivision connectivity'. This special type of mesh connectivity is generated by iteratively applying a uniform subdivision operator to a coarse base mesh. The main advantage of subdivision connection meshes is the direct availability of multiresolution algorithms, since different refinement levels provide levels of detail. However, the base mesh construction and the vertex sampling are not easy to control.

The third group of algorithms is related to remeshing approaches, which input a triangular mesh and resample it such that the new tessellations still approximate the same surfaces but also satisfy some quality requirements. Remeshing algorithms using these methods (Turk, 1992; Hoppe *et al.*, 1993; Frey and Borouchaki, 1998; Frey, 2000) are usually based on local modification operators enabling simplification (e.g., vertex or edge collapsing), enrichment (edge splitting), or improvement (e.g., node relocation or edge swapping). More recent works (Alliez *et al.*, 2002; 2003a; 2003b; 2005; Gu *et al.*, 2002) are based on global parameterization of the original mesh, and then on resampling in the global parametric space. These techniques yield good results, but have the disadvantages of heavy computational load and numerical instability. The simplified alternatives to global parameterization are local parameterization methods (Surazhsky and Gotsman, 2003; Surazhsky *et al.*, 2003). Sifri *et al.*(2003) and Peyré and Cohen (2006) presented methods for remeshing surfaces using geodesic path calculations. The results gave a set of vertices uniformly or adaptively distributed on the surface. The survey by Alliez and Gotsman (2003) provided more information on these methods.

Valette and Chassery (2004) proposed a fast and efficient algorithm for uniform coarsening, which can be used for large models. The first step of this method is clustering of the mesh triangles into an approximation of CVTs. The second step consists of replacing each cluster by a vertex, and then constructing the Delaunay triangulation according to the CVTs. This method outputs meshes with high subsampling ratios and high mesh quality. Valette *et al.*(2005) extended the method of (Valette and Chassery, 2004) to include adaptive behavior by regarding local estimated curvatures as weights in the energy term to be minimized when clustering. They also introduced some methods to speed up the clustering and guarantee the validity. In this paper, we improve the method of (Valette and Chassery, 2004) by incorporating a curvature-adapted behavior in a different way from (Valette *et al.*, 2005). The experimental results show that our method can produce better results. Valette *et al.*(2008) presented a generic framework for 3D surface remeshing based on metric-driven discrete Voronoi diagram construction. It can efficiently produce anisotropic or isotropic results at low computational cost.

Cohen-Steiner *et al.*(2004) introduced a new concept called 'shape proxy'. Using this concept, they drove the distortion error down through repeated clustering of faces into best-fitting regions. The final partition regions approximate the result of the CVTs based on a new error metric. Because the partition regions have no guarantee of being almost flat and convex, the output polygonal mesh needs some post-processing.

OUTLINE OF OUR ALGORITHM

Our algorithm inputs a triangular mesh M, and outputs an adaptive triangular mesh with much fewer triangles than the original mesh M. There are two main steps in the algorithm:

(1) Adaptive refinement: To calculate the curvature of every triangle in the input mesh and then to apply a red-green split on the triangles that satisfy a

prescribed criterion according to the curvature.

(2) Coarsening mesh: To partition the input mesh into n regions based on CVTs and then to construct the dual Delaunay triangulation. The method we shall present is based on (Valette and Chassery, 2004). However, we modify the energy term adopted by them and add a new constraint term to make the resulting meshes adapt to the curvature. Also, we use different methods to estimate the density function and to avoid the clusters falling into several departments.

ADAPTIVE REFINEMENT

Like (Surazhsky and Gotsman, 2003), we use the triangle area as a factor to produce a mesh reflecting the curvature of the original mesh. Intuitively, the large curved regions of M will contain small triangles and a dense vertex sampling, while almost flat regions will have large triangles with more sparse vertices. We need only to specify the ratios between triangle areas depending on the curvatures to define a curvature function.

For every vertex v of the triangle mesh M, we use the method of (Meyer *et al.*, 2002) to estimate the discrete Gaussian and mean curvatures, denoted as $K(v)$ and $H(v)$, respectively. Then we define a hybrid curvature function as [Surazhsky and Gotsman (2003) used the function to define the density function]

$$F(v)=0.5K(v)+0.5H^2(v).$$

For every triangle $t(v_1, v_2, v_3)$ of the mesh M, we define the refinement criterion function as

$$D(t)=(F(v_1)+F(v_2)+F(v_3))/3.$$

The function $D(t)$ is sensitive to noise in either the geometry or the connectivity of the mesh M (Fig.1). To alleviate this, we truncate all extreme values of $D(t)$. We will use some statistical tools to process the data calculated by the function $D(t)$. Let $\bar{D}$ be the mean of $D(t)$ and $\tilde{D}$ be the standard deviation of $D(t)$ on the mesh. We have

$$\bar{D}=\frac{1}{m}\sum_{i=1}^{m}D(t_i),\quad \tilde{D}=\sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(D(t_i)-\bar{D})^2},$$

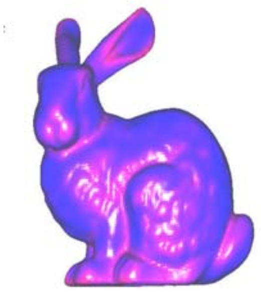

Fig.1 Curvature estimation on the Stanford Bunny
The regions in light color represent regions with large curvature and those in deep color represent the regions with small curvature

where t_i is a triangle in the mesh M, and m denotes the number of triangles in M.

For each triangle t_i in the mesh, we truncate its corresponding curvature $D(t_i)$ by the following rule:

$$D(t_i)=\begin{cases}\bar{D}+2\tilde{D}, & D(t_i)>\bar{D}+2\tilde{D},\\ \bar{D}-2\tilde{D}, & D(t_i)<\bar{D}-2\tilde{D}.\end{cases}$$

We also assign a threshold value δ according to the deviation value of $D(t)$:

$$\delta=\bar{D}+\alpha\tilde{D},$$

where α is a positive user-defined value. Usually we use α=0.8.

We split the triangle t into four small triangles, where $D(t)$ is greater than the given threshold δ. Other triangles remain coarse. To avoid cracks in the mesh where two triangles from different levels meet, we have to use a special technique called 'red-green triangulation' (Bank *et al.*, 1983; Vasilescu and Terzopoulos, 1992; Verfürth, 1994). A normal 1-to-4 split is called a red split. To fix cracks in the mesh, triangle bisection is used, which is called a green split. We assign red color to the triangle where D is greater than the value δ. The remaining triangles are assigned green color. First, we subdivide all red triangles using a 1-to-4 split. Second, we deal with the green triangles according to the color of their neighboring triangles. We divide the green triangles into four groups (Fig.2):

(1) There is no red triangle around the green triangle (Fig.2a). We do nothing to this green triangle.

(2) There is one red triangle around the green triangle (Fig.2b). We split the common edge between

the red triangle and blue triangle at the midpoint. The green triangle is replaced with two small triangles.

(3) There are two red triangles around the green triangle (Fig.2c). The green triangle is replaced with three small triangles.

(4) There are three red triangles around the green triangle (Fig.2d). We subdivide the green triangle using a 1-to-4 split.

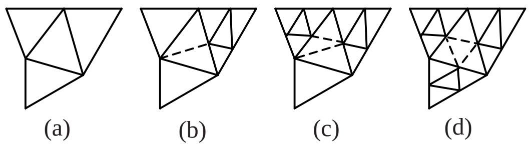

Fig.2 The red-green triangulation
(a)~(d) correspond to zero, one, two and three red triangle(s) around the green triangle, respectively

It is easy to see that we will obtain a mesh on which the regions with larger curvature contain more triangles than the regions with smaller curvature if we keep doing the adaptive refinement operation iteratively on M (Fig.3). In general, we repeat the adaptive refinement operation on M several times. To decide how many times are enough, we introduce a ratio as a measure. Let A be a set that contains some triangles. The size of A is denoted by $n(A)$ and the total sum of the area of triangles in A is denoted by $s(A)$. Now we define $\tau(A)=n(A)/s(A)$, and τ represents the number of triangles per unit area. Then we divide all triangles in M into two sets, denoted as A_1 and A_2. A_1 contains all the triangles whose curvature is greater than $\bar{D}$ and A_2 contains all the other triangles. Let $\lambda=\tau(A_1)/\tau(A_2)$, and λ measures the ratio of the number of triangles with higher curvature per unit area, which contain more detailed features of the model, to the number of

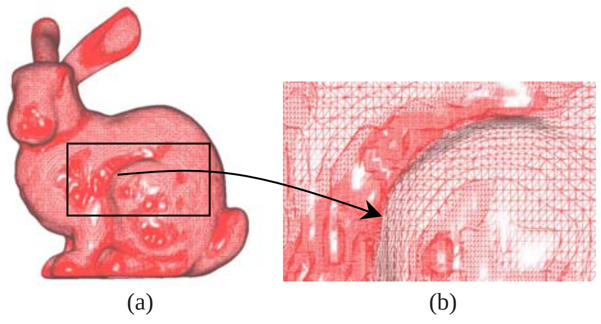

Fig.3 The Stanford Bunny mesh by adaptive refinement (applied twice to the input mesh)
(a) The whole mesh; (b) The enlarged picture of a selected region

triangles with lower curvature per unit area. Apparently, the goal of a red-green split is to make λ greater. In practice, we do a red-green split on M iteratively until λ is greater than a certain value β, which is decided by the user. Then we can obtain the ideal mesh. Generally, a greater β leads to more red-green splits, and we should balance the quality and the performance. In this paper, we use β=2.

COARSENING MESH

In this section, we will briefly introduce CVTs [details can be found in (Du *et al.*, 1999)] and then present our clustering algorithm following the work of (Valette and Chassery, 2004).

Centroidal Voronoi tessellations

Definition 1 Given an open set Ω of $\mathbb{R}^m$, and n different points z_i (i=0, 1, ..., n−1), the Voronoi tessellations can be defined as n different regions V_i such that

$$V_i=\left\{x\in\Omega \middle| d(x,z_i)<d(x,z_j),\ j=0,1,...,n-1, j\neq i\right\},$$

where d is a function of distance.

Given a region V and a density function $\rho(x)$ defined on V, the mass centroid z^* of V is defined by

$$z^*=\int_V x\rho(x)\mathrm{d}x \Big/ \int_V \rho(x)\mathrm{d}x.$$

Thus, given n points z_i (i=0, 1, ..., n−1), in the domain Ω, we can define their associated Voronoi regions V_i (i=0, 1, ..., n−1), which form a tessellation of Ω. Given the regions V_i (i=0, 1, ..., n−1), we can also define their mass centroids z_i^* (i=0, 1, ..., n−1).

Definition 2 Given the set of points z_i (i=0, 1, ..., n−1), in the domain Ω and a positive density function $\rho(x)$ defined on the domain Ω, a Voronoi tessellation is called a CVT if

$$z_i=z_i^*,\ \ i=0, 1, ..., n-1.$$

That is, the points z_i are both the generators and the centroids of the Voronoi regions V_i. The corresponding dual Delaunay triangulation is referred to as CVDT.

Moreover, CVTs minimize the energy given as

$$E=\sum_{i=0}^{n-1}\int_{V_i}\rho(x)\|x-z_i\|^2\,\mathrm{d}x. \tag{1}$$

One way to build a CVT is to use Lloyd's relaxation method (Lloyd, 1982). Valette and Chassery (2004) presented a fast approach to approximating the CVTs, based on the global minimization of the energy term E defined in Eq.(1). Here, our face clustering method is based on their method, but we simplify the energy term E in a different way.

Our face clustering method

Like (Valette and Chassery, 2004), we want to construct CVTs on a discrete set: a triangular mesh M. We approximate a Voronoi region to be the union of several triangles of the mesh M. Each region V_i is the union of several triangles t_j of the mesh M, and each triangle t_j belongs to only one region V_i. Constructing such tessellations becomes a clustering problem: we want to merge all triangles of M into n regions V_i, where each region has only one connected component. However, if we minimize the energy term described in Eq.(1) directly, it is difficult to produce a curvature adapted coarsened mesh. The main reason is that, although we execute adaptive refinement to the mesh first, the effect of adaptive refinement is greatly reduced by face clustering. Many triangles split from the same triangle in the refinement phase are clustered into one region again when minimizing the energy leading to a fairly uniform result. To make the resulting mesh adapt to the curvature distribution of the original mesh, we must retain the effect of adaptive refinement. Thus, we need to add some constraints to Eq.(1) to prevent the split triangles from being merged into one region. Let m_i be the number of the triangles that belong to region V_i and m be the mean of m_i. We modify the energy term in Eq.(1) and rewrite it as

$$E=\sum_{i=0}^{n-1}\left(\left(\frac{m_i}{m}\right)^r\sum_{t_j\in V_i}\int_{t_j}\rho(x)\|x-z_i\|^2\,\mathrm{d}x\right), \tag{2}$$

where $m=\sum_{i=0}^{n-1}m_i/n$, and r is a positive user-given parameter, which controls the curvature adapted behavior. If r=0, Eq.(2) is the same as Eq.(1). The higher value of r will prevent the triangles split in the adaptive refinement step from being merged more strictly.

The centroids z_i of regions V_i can be rewritten as

$$z_i=\sum_{t_j\in V_i}\int_{t_j}\rho(x)x\mathrm{d}x\Big/\sum_{t_j\in V_i}\int_{t_j}\rho(x)\mathrm{d}x. \tag{3}$$

To calculate the centroids y_j of triangles t_j, $y_j=(v_1+v_2+v_3)/3$, where v_1, v_2, v_3 are the vertices of the triangle t_j, we denote the area of the triangle t_j as s_j, which can be computed as $s_j=\int_{t_j}\mathrm{d}x$.

Now we can approximate the energy term E and the centroids z_i as follows:

$$E^{(1)}=\sum_{i=0}^{n-1}\left(\left(\frac{m_i}{m}\right)^r\sum_{t_j\in V_i}\rho(y_j)s_j\|y_j-z_j^{(1)}\|^2\right),$$
$$z_j^{(1)}=\sum_{t_j\in V_i}\rho(y_j)s_jy_j\Big/\sum_{t_j\in V_i}\rho(y_j)s_j.$$

We denote the mass of the triangle t_j as w_j, which can be calculated as $w_j=\rho(y_j)s_j$. We assume that the mass on each triangle of the mesh is the same; i.e., the mass is distributed uniformly on the mesh, although the density function is not uniform. Then, the energy term and the centroids are simplified further:

$$E^{(2)}=\sum_{i=0}^{n-1}\left(\left(\frac{m_i}{m}\right)^r\left(\sum_{t_j\in V_i}\|y_j-z_i^{(2)}\|^2\right)\right), \tag{4}$$

$$z_j^{(2)}=\sum_{t_j\in V_i}y_j\Big/\sum_{t_j\in V_i}1. \tag{5}$$

By combining Eqs.(4) and (5) we obtain

$$E^{(2)}=\sum_{i=0}^{n-1}\left(\left(\frac{m_i}{m}\right)^r\left(\sum_{t_j\in V_i}\|y_j\|^2-\left\|\sum_{t_j\in V_i}y_j\right\|^2\Big/\sum_{t_j\in V_i}1\right)\right). \tag{6}$$

Obviously, minimizing $E^{(2)}$ is equivalent to minimizing $E^{(3)}$ described below:

$$E^{(3)}=\sum_{i=0}^{n-1}\left((m_i)^r\left(\sum_{t_j\in V_i}\|y_j\|^2-\left\|\sum_{t_j\in V_i}y_j\right\|^2\Big/\sum_{t_j\in V_i}1\right)\right). \tag{7}$$

At this point, the energy term $E^{(3)}$ only depends on the chosen clustering of the input mesh. We no longer need to explicitly compute the positions of the Voronoi sites. We minimize the above energy term $E^{(3)}$ in Eq.(7) with an iterative algorithm (Valette and Chassery, 2004).

We also follow the triangulation method of (Valette and Chassery, 2004) to construct the dual mesh according to the results of the final clustering. Then we obtain the coarsening triangular mesh.

Mesh coarsening method

There are four main steps to obtain the coarsened mesh [details can be found in (Valette and Chassery, 2004)]:

Step 1: Initialization. A number n is chosen as the number of clusters. This number is decided by the user. Then we decompose the input mesh into n regions. Usually, we use a region-growing strategy to achieve this. We randomly pick n different seed triangles in the mesh as seed regions and aggregate adjacent triangles to seed regions until no regions can be grown.

Step 2: Energy minimization. For each boundary triangle (which is on the boundary of a cluster), we compute the value of $E^{(3)}$ in two cases: (1) The initial configuration—t_j belongs to the cluster V_i, and is the neighboring triangle of the cluster V_k; (2) V_k grows and V_i shrinks, and t_j belongs to V_k. The case resulting in the smallest energy $E^{(3)}$ is chosen and the cluster configurations are updated.

Step 3: Repeat Step 2 until a satisfactory convergence is achieved (Fig.4).

Step 4: The dual Delaunay triangulation is similar to Delaunay triangulations, where a triangle is created for each point where three Voronoi regions meet. The three vertices of the triangle are the three Voronoi seeds. Here, the Voronoi seeds are approximated by choosing the closest points to the cluster centroids. The ambiguous cases exist when more than three different Voronoi regions meet at one single point. As a consequence, if k clusters meet at one single vertex, then we create $k-2$ triangles.

Cluster separation detection

In practice, the algorithm may output some clusters that contain several disconnected components. This will be a problem in the dual Delaunay triangulation. Valette and Chassery (2004) presented a method to fix it. However, we adopt a different method, similar to the one introduced by Valette *et al.*(2005), to solve it. The method is based on boundary testing. The details are as follows: when a triangle T is removed from a Voronoi region V_1 and added to another Voronoi region V_2, if it will lead to the shape of V_1's boundary looking like '∞' (Fig.5b), i.e., there is a vertex P that has four connected edges on V_1's boundary, let the value of $E^{(3)}$ be infinite. It will prevent T from being removed from V_1 and also prevent V_1 from being divided into several disconnected parts. Although we use this method each time when energy testing, the expense of this method is low because the boundary information has already been cached when clustering. In our experience, this step is efficient and costs only about 10% of the total clustering time. Using this method, no Voronoi region will ever fall into several disconnected parts when clustering. The speed of coarsening could be improved compared to the method of (Valette and Chassery, 2004) for there is no need to repeat the energy minimization step. There is also no need to worry about the validity of the clustering anymore.

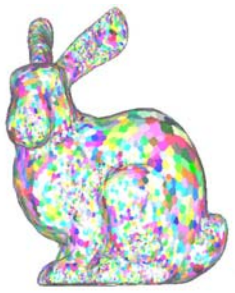

Fig.4 Result of the face clustering algorithm on the Stanford Bunny

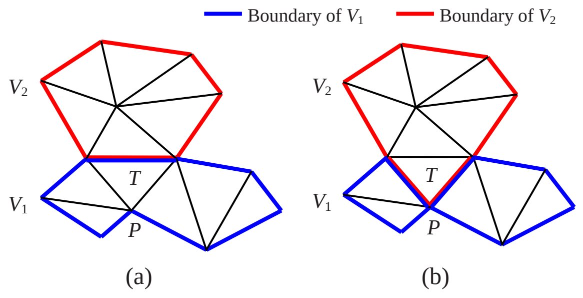

Fig.5 Boundary testing

(a) Before the testing; (b) After the testing

Post-process

At the end of the algorithm, we use a post-process to further improve the quality of the coarsening mesh. A uniform refinement is obtained by using a 1-to-4 splitting operator. We apply the uniform refinement iteratively three times to the coarsening mesh M_c, which we generate in the above context. Then we choose the vertex number of the mesh M_c as the number of clusters, and use the method in the subsection "Mesh coarsening method" to coarsen the mesh after refinement again. This post-process can be treated as a remeshing algorithm. A comparison of our experimental results with and without post-process is made in the next section. It shows that the post-process step further improves the resulting meshes' quality.

EXAMPLES

In this section, we show the results of testing our adaptive coarsening algorithm on five models: Rocker Arm, Stanford Bunny, Igea Artifact, Venus, and Laurent's Hand. The results are given in Figs.6~11 and Tables 1 and 2.

The traditional way of measuring the quality of a triangular mesh is by measuring the geometric properties of the resulting triangles. For the geometry, statistics are usually collected on the minimal angle of the triangles. Obviously this value is anywhere between 0° and 60°. For a high-quality mesh, the minimum of these values should be no less than 10°, and the average should be no less than 45° according to (Surazhsky and Gotsman, 2003). Table 1 shows the results obtained for the models presented in this paper. The column of N_{vo} shows the number of vertices in the original mesh. The column of N_{vc} shows the number of vertices in the coarsened mesh. We also measured the output mesh based on the triangles' shapes [minimal quality Q_{min} and average quality Q_{av}, which range between 0 and 1, as defined by Frey and Borouchaki (1997)]. Table 1 also shows the Hausdorff distance d (as a percentage of the mesh bounding box diagonal) between the original model and the coarsened one, measured with the Metro tool (Cignoni *et al.*, 1998).

In Table 2, we compare our results with those of quadric error metrics (QEM) (Garland and Heckbert, 1997).

As we have seen, our algorithm can produce meshes adapted to the local curvature of the underlying surface. It allows us to match surface features (Figs.7, 9a, 9c, 9e and 11) better than uniform coarsening methods. Another advantage of our algorithm is that it uses the CVT techniques, which enable us to output meshes with high quality. As shown in Table 1, all the minimum angles of the triangles in our output mesh were above 15° and the average angles were above 45°. Although the input mesh had a very low quality (Figs.6, 8 and 10), the output mesh from our algorithm was still good. The post-process plays a key role in our algorithm, enabling us to output meshes with better quality (Figs.9a~9f). In Figs.9a, 9c and 9e, we show the output meshes with the post-process. In Figs.9b, 9d and 9f, we show the output meshes without the post-process. It is clear that

Table 1 Our results obtained for reference models (*r*=2)

Figure	Model	With post-process	N_{vo}	N_{vc}	α_{min} (°)	α_{av} (°)	$P_{<30°}$ (%)	Q_{min}	Q_{av}	d (%)
6	Rock Arm		66 372		0.0001	33.63	11.00	0.0018	0.61	
7	Rock Arm	Yes	66 372	8000	16.7300	46.25	0.70	0.3300	0.83	0.8
8	Stanford Bunny		54 664		0.0011	32.38	14.00	0.0034	0.60	
9a	Stanford Bunny	Yes	54 664	3000	17.7600	46.40	0.70	0.3328	0.83	0.9
9b	Stanford Bunny	No	54 664	3000	8.1290	43.49	3.10	0.1506	0.79	0.9
9c	Stanford Bunny	Yes	54 664	5000	15.8700	46.21	0.75	0.2522	0.82	0.7
9d	Stanford Bunny	No	54 664	5000	6.8790	42.52	3.60	0.1074	0.78	0.7
9e	Stanford Bunny	Yes	54 664	8000	17.4300	45.87	0.82	0.2756	0.82	0.5
9f	Stanford Bunny	No	54 664	8000	0.6764	40.81	5.36	0.0136	0.75	0.5
10	Igea Artifact		184 494		0.0250	31.85	14.38	0.0007	0.59	
11	Igea Artifact	No	184 494	10 000	17.8100	46.68	0.50	0.3225	0.83	0.4

N_{vo}: number of vertices in the original mesh; N_{vc}: number of vertices in the coarsened mesh; α_{min}: the minimal angle of the triangles; α_{av}: the average minimal angle; $P_{<30°}$: the percentage of angles which are less than 30°; Q_{min}: minimal shape quality; Q_{av}: average shape quality; d: Hausdorff distance measured by the Metro tool

Table 2 Comparison of the results from using the quadric error metric (QEM), the method of (Valette *et al.*, 2005) and our proposed method (r=0.5)

Model	Method	N_{vo}	N_{vc}	α_{min} (°)	α_{av} (°)	$P_{<30°}$ (%)	Q_{min}	Q_{av}	d (%)	Time (s)
Venus	QEM	134345	2233	1.1800	35.19	32.77	0.0220	0.66	0.3	14.40
	Valette *et al.*, 2005	134345	2235	21.7100	47.15	0.47	0.3900	0.84	0.8	79.70
	Proposed	134345	2235	20.0400	49.86	0.36	0.4390	0.88	0.6	80.88
Stanford Bunny	QEM	34839	5000	1.0750	34.53	35.53	0.0220	0.65	0.2	3.20
	Valette *et al.*, 2005	34839	5000	3.7900	43.38	6.59	0.0650	0.79	0.9	29.70
	Proposed	34839	5000	2.4150	48.96	1.00	0.0380	0.86	0.6	30.90
Laurent's Hand	QEM	50085	5000	0.7106	33.96	37.23	0.0130	0.64	0.2	3.80
	Valette *et al.*, 2005	50085	5000	9.5390	43.58	6.26	0.2340	0.79	0.3	41.40
	Proposed	50085	5000	6.8350	47.19	1.00	0.1040	0.84	0.5	39.77

N_{vo}: number of vertices in the original mesh; N_{vc}: number of vertices in the coarsened mesh; α_{min}: the minimal angle of the triangles; α_{av}: the average minimal angle; $P_{<30°}$: the percentage of angles which are less than 30°; Q_{min}: minimal shape quality; Q_{av}: average shape quality; d: Hausdorff distance measured by the Metro tool

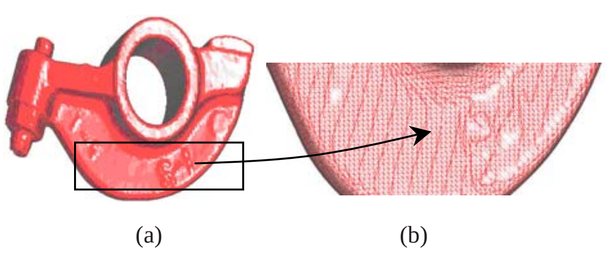

Fig.6 Rocker Arm

(a) The input mesh (shown by a flat shaded mode); (b) An enlarged picture of the selected region (shown by a Gouraud shaded mode). The original model is available at http://www.cyberware.com/samples/index.html. We used the tool 'PolyMender' to process the original mesh and obtained the input mesh. The PolyMender software is available at http://www.cs.wustl.edu/~taoju/index.htm#soft. We also used this tool to process the following Stanford Bunny and Igea Artifact, to obtain our input mesh

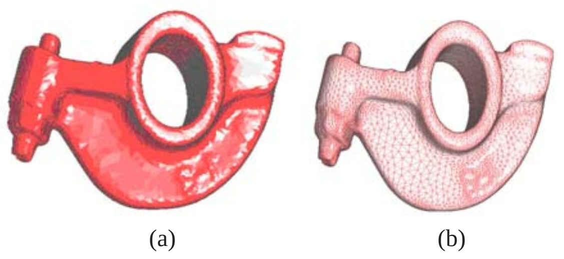

Fig.7 Rocker Arm with 8000 points and the output mesh by our algorithm (r=2)

(a) Flat shaded mode; (b) Gouraud shaded mode

the quality of output meshes was improved when the post-process was included. The processing time was measured on a low-end AT/AT compatible PC with AMD Sempron® CPU 1.66 GHz and 1 GB RAM. It took about 1~3 min for the above models to run.

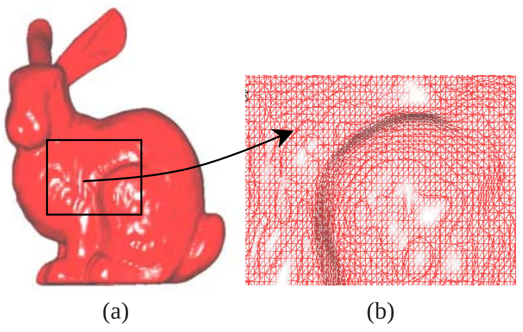

Fig.8 Stanford Bunny

(a) The input mesh (shown by a flat shaded mode); (b) An enlarged picture of the selected region (shown by a Gouraud shaded mode). The original model is available at http://graphics.stanford.edu/data/3Dscanrep/

Table 2 shows a comparison among the results using our method, the well-known QEM method and the method proposed by Valette *et al.*(2005). From the experimental results, we can see that while the quality of our output meshes and those of (Valette *et al.*, 2005) is better than the quality of those from QEM, the Hausdorff distances are greater. We think the main reason is that QEM, which is based on minimal distances, is primarily concerned with approximation error and preserving topology, but our method and that of (Valette *et al.*, 2005), which are based on CVTs, emphasize the quality of triangles. Experimentally, the approximation errors of our results are small enough and can be acceptable. From our experimental results, we can also see that most qualities of our method's output meshes are better than those of the method of (Valette *et al.*, 2005). Our Hausdorff distances are smaller except in the Laurent's Hand model.

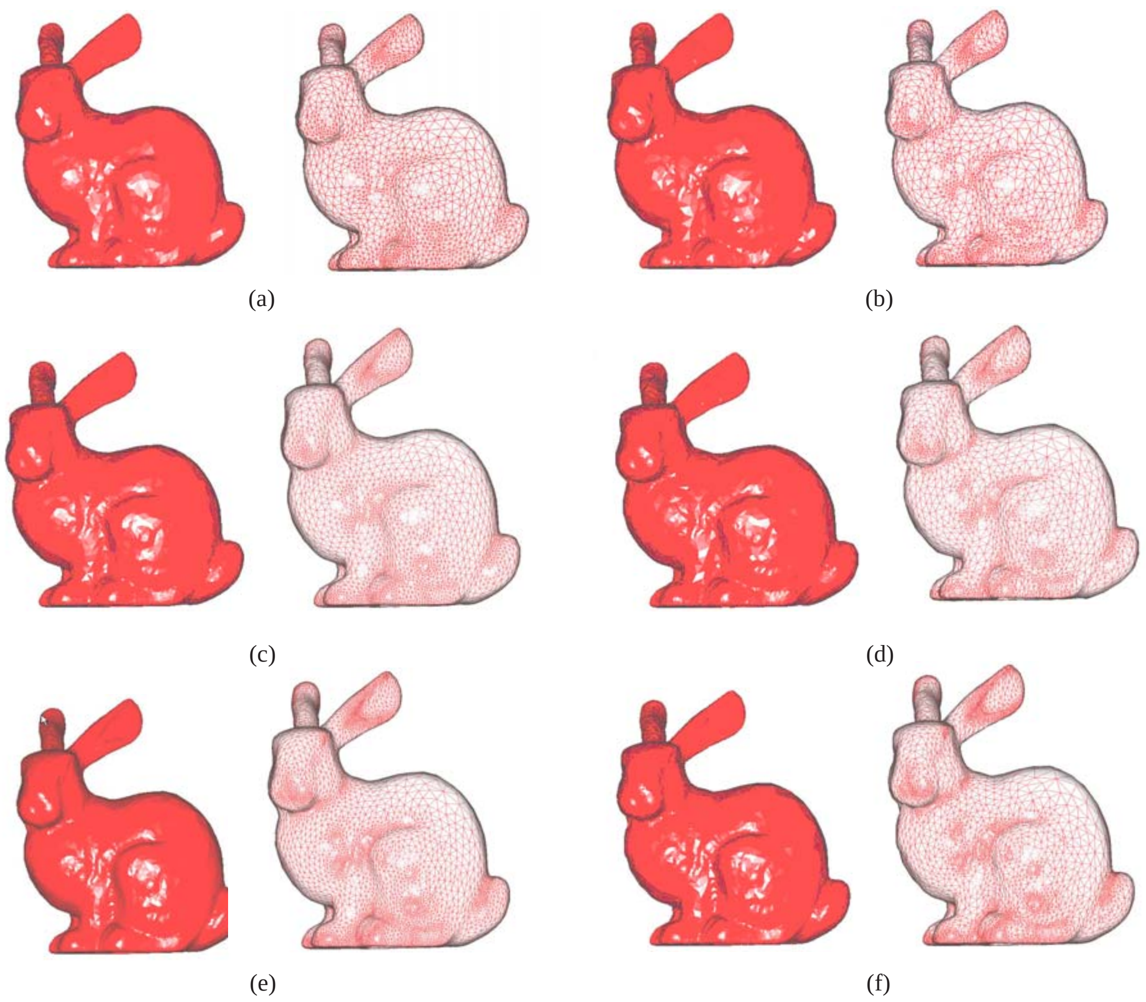

Fig.9 Stanford Bunny with different numbers of points and the output mesh using our algorithm or without post-process (*r*=2)
In each subfigure, the left is the flat shaded mode and the right is the Gouraud shaded mode. (a), (b): With 3000 points; (c), (d): With 5000 points; (e), (f): With 8000 points. (a), (c), (e): Using our algorithm; (b), (d), (f): Without post-process

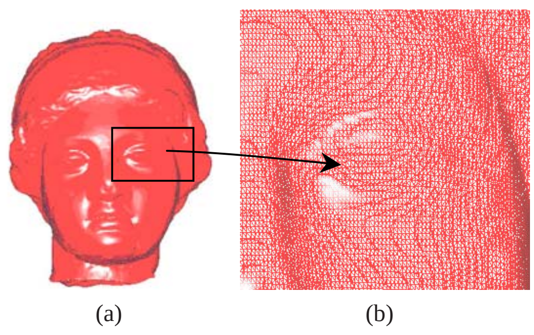

Fig.10 Igea Artifact
(a) The input mesh (shown in a flat shaded mode); (b) An enlarged picture of the selected region (shown by a Gouraud shaded mode). The original model is available at http://www.cyberware.com/samples/index.html

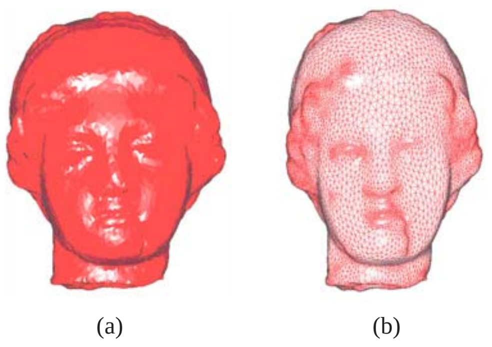

Fig.11 Igea Artifact with 10 000 points and the output mesh without post-process (*r*=2)
(a) Flat shaded mode; (b) Gouraud shaded mode

CONCLUSION

In this paper we present an efficient algorithm for adaptive mesh coarsening, which is suitable for smoothing meshes with arbitrary topology. Objective criteria show that the output meshes have good quality. Using the adaptive subdivision operator, the algorithm produces a mesh reflecting the curvature of the original mesh. Intuitively, large curved regions will contain small triangles and a dense vertex sampling, and vice versa. In the ideal case, our method produces a mesh in which the mass of each triangle is almost equal; i.e., the mass on the mesh is distributed uniformly. This helps us to simplify the energy term in the face clustering algorithm based on CVTs. In future work, we plan to extend the method to deal with meshes with boundaries or sharp features.

ACKNOWLEDGEMENT

The Bunny model was used by courtesy of Stanford University. Laurent's Hand model was provided courtesy of INRIA by the AIM@SHAPE Shape Repository.

References

Alliez, P., Gotsman, C., 2003. Recent Advances in Compression of 3D Meshes. Proc. Symp. on Multiresolution in Geometric Modeling, p.53-82.

Alliez, P., Meyer, M., Desbrun, M., 2002. Interactive geometry remeshing. *ACM Trans. Graph.*, **21**(3):347-354. [doi:10.1145/566654.566588]

Alliez, P., Cohen-Steiner, D., Devillers, O., Lévy, B., Desbrun, M., 2003a. Anisotropic polygonal remeshing. *ACM Trans. Graph.*, **22**(3):485-493. [doi:10.1145/882262.882296]

Alliez, P., Verdiere, E.C., Devillers, O., Isenburg, M., 2003b. Isotropic Surface Remeshing. Proc. Shape Modeling Int., p.49-58. [doi:10.1109/SMI.2003.1199601]

Alliez, P., Verdiere, E.C., Devillers, O., Isenburg, M., 2005. Centroidal Voronoi diagrams for isotropic surface remeshing. *Graph. Models*, **67**(3):204-231. [doi:10.1016/j.gmod.2004.06.007]

Bank, R., Sherman, A., Weiser, A., 1983. Some Refinement Algorithms and Data Structures for Regular Local Mesh Refinement. Scientific Computing IMACS, Amsterdam, North-Holland, p.3-17.

Cignoni, P., Rocchini, C., Scopigno, R., 1998. Metro: measuring error on simplified surfaces. *Comput. Graph. Forum*, **17**(2):167-174. [doi:10.1111/1467-8659.00236]

Cohen, J., Varshney, A., Manocha, D., Turk, G., Weber, H., Agarwal, P., Brooks, F., Wright, W., 1996. Simplification Envelopes. Proc. 23rd Annual Conf. on Computer Graphics and Interactive Techniques, p.119-128. [doi:10.1145/237170.237220]

Cohen-Steiner, D., Alliez, P., Desbrun, M., 2004. Variational Shape Approximation. ACM SIGGRAPH, Los Angeles, California, p.905-914.

Du, Q., Faber, V., Gunzburger, M., 1999. Centroidal Voronoi tessellations: applications and algorithms. *SIAM Rev.*, **41**(4):637-676. [doi:10.1137/S0036144599352836]

Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., Stuetzle, W., 1995. Multiresolution Analysis of Arbitrary Meshes. Proc. 22nd Annual Conf. on Computer Graphics and Interactive Techniques, p.173-182. [doi:10.1145/218380.218440]

El-Sana, J., Varshney, A., 1997. Controlled Simplification of Genus for Polygonal Models. Proc. Visualization, p.403-410, 565.

Frey, P.J., 2000. About Surface Remeshing. Proc. 9th Int. Meshing Roundtable, p.123-136.

Frey, P.J., Borouchaki, H., 1997. Surface Mesh Evaluation. Proc. 6th Int. Meshing Roundtable, p.363-374.

Frey, P.J., Borouchaki, H., 1998. Geometric surface mesh optimization. *Comput. Visual. Sci.*, **1**(3):113-121. [doi:10.1007/s007910050011]

Garland, M., 1999. Multiresolution Modeling: Survey & Future Opportunities. Eurographics—State of the Art Reports, p.111-131.

Garland, M., Heckbert, P.S., 1997. Surface Simplification Using Quadric Error Metrics. Computer Graphics Proc., Annual Conf. Series, New York, p.209-216.

Gu, X., Gortler, S.J., Hoppe, H., 2002. Geometry images. *ACM Trans. Graph.*, **21**(3):355-361. [doi:10.1145/566654.566589]

Guskov, I., Vidimce, K., Sweldens, W., Schröder, P., 2000. Normal Meshes. Proc. 27th Annual Conf. on Computer Graphics and Interactive Techniques, p.95-102. [doi:10.1145/344779.344831]

Hoppe, H., 1996. Progressive Meshes. Proc. 23rd Annual Conf. on Computer Graphics and Interactive Techniques, p.99-108. [doi:10.1145/237170.237216]

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., 1993. Mesh Optimization. Proc. 20th Annual Conf. on Computer Graphics and Interactive Techniques, p.19-26. [doi:10.1145/166117.166119]

Hormann, K., Labsik, U., Greiner, G., 2001. Remeshing triangulated surfaces with optimal parameterizations. *Computer-Aided Design*, **33**(11):779-788. [doi:10.1016/S0010-4485(01)00094-X]

Kobbelt, L., Campagna, S., Seidel, H., 1998. A General Framework for Mesh Decimation. Proc. Graphics Interface, p.43-50.

Kobbelt, L.P., Vorsatz, J., Labsik, U., Seidel, H., 1999. A shrink wrapping approach to remeshing polygonal surfaces. *Comput. Graph. Forum*, **18**(3):119-130. [doi:10.1111/1467-8659.00333]

Lee, A.W.F., Sweldens, W., Schröder, P., Cowsar, L., Dobkin, D., 1998. MAPS: Multiresolution Adaptive Parameterization of Surfaces. Proc. 25th Annual Conf. on Computer Graphics and Interactive Techniques, p.95-104. [doi:10.1145/280814.280828]

Lindstrom, P., Turk, G., 1998. Fast and Memory Efficient Polygonal Simplification. Proc. Conf. on Visualization, Research Triangle Park, North Carolina, USA, p.279-286.

Lloyd, S., 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, **28**(2):129-137. [doi:10.1109/TIT.1982.1056489]

Meyer, M., Desbrun, M., Schröder, P., Barr, A.H., 2002. Discrete Differential Geometry Operators for Triangulated 2-Manifolds. Visualization and Mathematics III, p.35-57.

Peyré, G., Cohen, L., 2006. Geodesic remeshing using front propagation. *Int. J. Comput. Vis.*, **69**(1):145-156. [doi:10.1007/s11263-006-6859-3]

Schroeder, W.J., Zarge, J.A., Lorensen, W.E., 1992. Decimation of Triangle Meshes. Proc. 19th Annual Conf. on Computer Graphics and Interactive Techniques, p.65-70. [doi:10.1145/133994.134010]

Sifri, O., Sheffer, A., Gotsman, C., 2003. Geodesic-based Surface Remeshing. Int. Meshing Roundtable, p.189-199.

Surazhsky, V., Gotsman, C., 2003. Explicit Surface Remeshing. Proc. Eurographics/ACM SIGGRAPH Symp. on Geometry Processing, Aachen, Germany, p.20-30.

Surazhsky, V., Alliez, P., Gotsman, C., 2003. Isotropic Remeshing of Surfaces: A Local Parameterization Approach. Proc. Int. Meshing Roundtable, p.215-224.

Turk, G., 1992. Re-tiling polygonal surfaces. *ACM SIGGRAPH Comput. Graph.*, **26**(2):55-64. [doi:10.1145/142920.134008]

Valette, S., Chassery, J., 2004. Approximated centroidal Voronoi diagrams for uniform polygonal mesh coarsening. *Comput. Graph. Forum*, **23**(3):381-389. [doi:10.1111/j.1467-8659.2004.00769.x]

Valette, S., Kompatsiaris, I., Chassery, J., 2005. Adaptive Polygonal Mesh Simplification with Discrete Centroidal Voronoi Diagrams. 2nd Int. Conf. on Machine Intelligence, Tozeur, Tunisia, p.655-662.

Valette, S., Chassery, J., Prost, R., 2008. Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams. *IEEE Trans. Visual. Comput. Graph.*, **14**(2):369-381. [doi:10.1109/TVCG.2007.70430]

Vasilescu, M., Terzopoulos, D., 1992. Adaptive Meshes and Shells: Irregular Triangulation, Discontinuities, and Hierarchical Subdivision. Proc. Computer Vision and Pattern Recognition, p.829-832. [doi:10.1109/CVPR.1992.223247]

Verfürth, R., 1994. A posteriori error estimation and adaptive mesh-refinement techniques. *J. Comput. Appl. Math.*, **50**(1-3):67-83. [doi:10.1016/0377-0427(94)90290-9]

Zelinka, S., Garland, M., 2002. Permission grids: practical, error-bounded simplification. *ACM Trans. Graph.*, **21**(2):207-229. [doi:10.1145/508357.508363]