Mesh Clustering by Approximating Centroidal Voronoi Tessellation *

Fengtao Fan
Depart. of Computer Science
University of Kentucky
ffan2@uky.edu

Fuhua (Frank) Cheng
Depart. of Computer Science
University of Kentucky
cheng@cs.uky.edu

Conglin Huang
Depart. of Computer Science
University of Kentucky
conglin.huang@uky.edu

Yong Li
Depart. of Computer Science
University of Kentucky
yong.li@uky.edu

Jianzhong Wang
Depart. of Computer Science
University of Kentucky
jwangf@uky.edu

Shuhua Lai
Depart. of Mathematics and
Computer Science
Virginia State University
slai@vsu.edu

ABSTRACT

An elegant and efficient mesh clustering algorithm is presented. The faces of a polygonal mesh are divided into different clusters for mesh coarsening purpose by approximating the Centroidal Voronoi Tessellation of the mesh. The mesh coarsening process after clustering can be done in an isotropic or anisotropic fashion. The presented algorithm improves previous techniques in local geometric operations and parallel updates. The new algorithm is very simple but is guaranteed to converge, and comes out better approximating meshes with the same computation cost. Moreover, the new algorithm is suitable for the variational shape approximation problem with $L^{2,1}$ distortion error metric and the convergence is guaranteed. Examples demonstrating efficiency of the new algorithm are also included in the paper.

Categories and Subject Descriptors

I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling —*Physically based modeling*

Keywords

Mesh Clustering, Centroidal Voronoi Tessellation, Shape Approximation

1. INTRODUCTION

3D mesh models are used in many important areas such as geometric modeling, computer animation, and CAD. With the availability of powerful laser scanners, large and dense meshes are easily acquired from physical world. However, since the full complexity of such models is not always required, coarsening a dense mesh, i.e., replacing the original mesh with a simpler but close enough mesh, is a necessary pre-processing step in many applications. Many mesh coarsening techniques have been presented, including the global optimization method [10, 18] and remeshing for mesh coarsening [22, 15, 13, 8].

Mesh clustering is to partition the faces or vertices of the mesh into different regions. Generally, these regions are required to be nonoverlapping and connected. One major application of the clustering technique is for mesh coarsening. Such a method builds the approximating mesh based on the clustering of the dense mesh. In mesh coarsening, clustering may not be explicitly required in a greedy clustering technique, like mesh decimation. A decimation method creates implicit partitions of the mesh through greedy and repeatedly collapsing mesh faces or vertices [6, 9, 17]. The resulting mesh is always sub-optimal [2]. The other clustering method for mesh coarsening is to construct the mesh clusters explicitly. The new mesh clustering technique, also designed for mesh coarsening, falls into this category.

There are quite a few papers discussing mesh approximation based on explicitly constructing clusters. Clustering by approximating the Centroidal Voronoi Tessellation (CVT) [3] on triangular meshes is first discussed in [23]. After constructing the clusters, the mesh is uniformly coarsened based on the clusters. Adaptive coarsening of a mesh based on clustering from Centroidal Voronoi Tessellation is presented in [25]. An extension from uniform mesh coarsening [23] to anisotropic mesh coarsening is discussed in [24]. A theoretical framework of variational shape approximation based on optimal mesh clustering with respect to some distortion error metric is presented in [2]. Especially, optimal clustering using $L^{2,1}$ metric faithfully captures the anisotropic nature of the mesh. A hierarchical face clustering technique is developed in [19]. Many applications such as collision detection, surface simplification and multiresolution radiosity benefit from this hierarchical clustering technique. Clustering faces in a set of characteristic regions to build a higher-level description of mesh geometry is explored in [12, 21, 16, 7]. Accelerating general iterative clustering algorithms for

*(Does NOT produce the permission block, copyright information nor page numbering). For use with ACM_PROC_ARTICLE-SP.CLS. Supported by ACM.

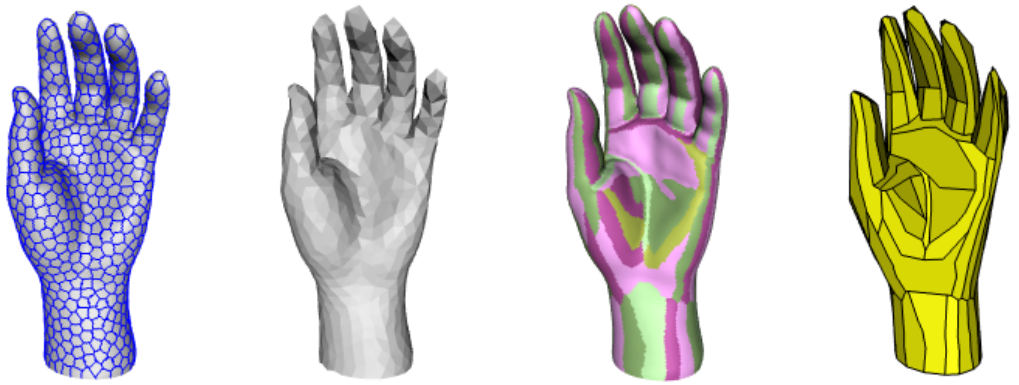

Figure 1: Clustering and approximation results on a hand model: the left-most figure shows the 500 clusters generated by approximating CVT on the mesh; the second from left figure is the uniformly coarsened mesh; the third from left figure has 98 clusters in different colors while using $L^{2,1}$ metric for clustering; the right-most figure is the approximating polygonal mesh.

meshes on GPU is discussed in [11].

This paper is inspired by the work presented in [23, 24, 2]. The goal here is to do clustering by approximating constrained Centroidal Voronoi Tessellation [4] or Centroidal Voronoi Tessellation [3] on a polygonal mesh. Starting with an initial partitioning of the mesh, the new algorithm iteratively tests the boundary edges between different clusters to update the cluster configuration until the boundary edges do not change any more. This boundary testing algorithm is also discussed in [23, 24]. But we derive a simpler algorithm by presenting a more rigorous mathematical analysis. The new algorithm is intuitive in that it only needs to compare the distances from one face centroid to centroids of adjacent clusters. The new algorithm is also extended for optimal geometric partitioning with respect to $L^{2,1}$ in [2]. The exciting result is that the new algorithm is guaranteed to converge while the algorithm based on Lloyd method in [2] is not. In summary, the contributions of this paper include:

1. a simpler algorithm which only needs to compare distances is derived for constructing clusters on a polygonal mesh by approximating Constrained Centroidal Voronoi Diagram or Centroidal Voronoi Diagram;
2. the new algorithm updates cluster configurations after comparing all boundary edges, not after comparing each boundary edge. This updating scheme improves the quality of the output coarse mesh.
3. the new algorithm for clustering with $L^{2,1}$ metric is guaranteed to converge. Sharing the same advantages of clustering with Centroidal Voronoi Tessellation methods, the new algorithm is fast.

The remaining part of the paper is organized as follows: Section 2 gives some basics on Centroidal Voronoi Tessellation and its extension; Section 3 presents an analysis and the new clustering algorithm; Section 4 discusses the boundary testing algorithm for clustering with $L^{2,1}$ metric; Section 5 proposes some strategies to make implementation more efficient; Section 6 gives applications of the new algorithm; test results are shown in Section 7; the conclusion is given in Section 8.

2. CENTROIDAL VORONOI TESSELLATION

Voronoi diagrams or *Voronoi tessellation* are essential structures in computational geometry and have been used in many important applications [20]. Given a domain Ω in $\Re^n$ and a set of points $\{\mathbf{z}_i\}_{i=1}^k$, the corresponding *Voronoi diagram* $\{V_i\}_{i=1}^k$ is a partition of Ω such that:

(1) $V_i \cap V_j = \emptyset$ and $\cup_{i=1}^k \overline{V}_i = \overline{\Omega}$, and
(2) $V_i = \{\mathbf{x} \in \Omega \mid |\mathbf{x} - \mathbf{z}_i| < |\mathbf{x} - \mathbf{z}_j| \text{ for } j = 1, 2, .., k, j \neq i\}$

$\{\mathbf{z}_i\}_{i=1}^k$ are called the *generators* and $\{V_i\}_{i=1}^k$ the *Voronoi regions*.

Centroidal Voronoi Tessellation (CVT) is an extension of Voronoi Tessellation by requiring that the generators are also the mass centroids of the Voronoi regions. Given a density function $\rho(\mathbf{x})$ on V, the mass centroid $\mathbf{z}^*$ of V is defined as

$$\mathbf{z}^* = \frac{\int_V \mathbf{x}\rho(\mathbf{x})d\mathbf{x}}{\int_V \rho(\mathbf{x})d\mathbf{x}}$$

Specifically, CVT of Ω is a minimizer of the energy functional [3] :

$$F(\mathbf{z}) = \sum_{i=1}^{n} \int_{V_i} \rho(\mathbf{x})|\mathbf{x} - \mathbf{z}_i|^2 d\mathbf{x} \tag{1}$$

where $\mathbf{z}_i \in \Omega$.

Constrained Centroidal Voronoi Tessellation [4] is the restriction of CVT to a surface. If a density function $\rho(\mathbf{x})$ is defined on a surface $\mathbf{S}$, we can define the *constrained mass centroid* $\mathbf{z}^c$ of a region $V \subseteq \mathbf{S}$ as the solution to the following minimization problem:

$$\min_{\mathbf{z} \in \mathbf{S}} \int_V \rho(\mathbf{x})|\mathbf{x} - \mathbf{z}|^2 d\mathbf{x} \tag{2}$$

A Voronoi Tessellation on a surface $\mathbf{S}$ is a Constrained Centroidal Voronoi Tessellation (CCVT) if and only if the generators $\mathbf{z}_i$ associated with each Voronoi region V_i are also the constrained mass centroid of V_i. Several applications of CCVT can be found in [4]. Furthermore, CCVT of surface $\mathbf{S}$ is also the minimizer of an energy functional similar to

the one defined in eq. (1) except now $\mathbf{z}_i \in \mathbf{S}$ [4]. Note that although $\mathbf{x}$ and $\mathbf{z}_i$ are points of the surface $\mathbf{S}$, CCVT uses the Euclidean distance instead of the geodesic distance. This minimization property is very important. We will take a deeper look of it in a later section. Several algorithms for constructing CVT and CCVT, such as the Lloyd method and k-means method, are presented in [3, 4].

In this paper, we will show how to construct discrete CCVT and CVT on a polygonal mesh. Discrete CVT was thoroughly investigated in [23, 24]. Here, we will give a rigorous analysis of constructing CCVT on a polygonal mesh. We choose analyzing discrete CCVT because discrete CVT can be viewed as a special case of discrete CCVT. In fact, most of the examples presented in this paper are implemented to construct discrete CVT on triangular meshes. We first present our derivations below, then point out the differences from those in [23, 24].

3. DISCRETE CONSTRAINED CENTROIDAL VORONOI TESSELLATION ON A POLYGONAL MESH

Given a polygonal mesh M and a cluster number n, we will try to divide the faces of M into n connected sets of faces V_i $(i = 1, 2, \ldots, n)$ by constructing a CCVT on M. These clusters $\{V_i\}$ form a discrete CCVT on the mesh M. Although discrete CCVT can be defined for any polygonal mesh, we will concentrate on triangular meshes in this paper.

In the continuous setting, CCVT is the minimizer of an energy functional similar to the one defined in eq. (1). For the discrete version of CCVT on a triangular mesh M, the region V_i is a connected collection of triangles. We can rewrite the energy functional as

$$F(\mathbf{z}) = \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i} \int_{T_k} \rho(\mathbf{x}) |\mathbf{x} - \mathbf{z}_i|^2 d\mathbf{x} \Big)$$

where T_k's are triangles in V_i. In this paper, we only consider the uniform case, i.e., $\rho(\mathbf{x}) = 1$. Then the energy functional is

$$F(\mathbf{z}) = \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i} \int_{T_k} |\mathbf{x} - \mathbf{z}_i|^2 d\mathbf{x} \Big) \tag{3}$$

In fact, the following equation holds

$$\int_{T_k} |\mathbf{x} - \mathbf{z}_i|^2 d\mathbf{x} = |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| + \frac{|T_k|}{12} \sum_{j=1}^{3} |\mathbf{x}_k^j - \mathbf{x}_k|^2 \tag{4}$$

where $|T_k|$ is the area of triangle T_k with vertices $\mathbf{x}_k^j (j = 1, 2, 3)$ and $\mathbf{x}_k$ is the centroid of T_k. The derivation of this formula is shown in the appendix. Note that the second term on the right hand side is a constant for each triangle. We will use σ_k to denote this term for triangle T_k. Substituting the integral in eq. (3) with eq. (4), we have

$$F(\mathbf{z}) = \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i} |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| \Big) + \sum_{T_k \in M} \sigma_k \tag{5}$$

The last constant item is not essential in subsequent work, hence, will be omitted for $F(\mathbf{z})$. The constrained mass centroid $\mathbf{z}_i$ of V_i on a continuous surface $\mathbf{S}$ is defined as a solution to the minimization problem defined in eq. (2) with V replaced with V_i. For discrete CCVT on M, we can use the same argument as in reformulating $F(\mathbf{z})$ to rewrite the minimization problem as:

$$\min_{\mathbf{z} \in \mathbf{M}} \Big(\sum_{T_k \in V_i} |\mathbf{x}_k - \mathbf{z}|^2 |T_k| + \sum_{T_k \in V_i} \sigma_k \Big)$$

The last constant item is not essential in the minimization process and, hence, will be omitted too. Furthermore, the above equation without the constant can be simplified as

$$\min_{\mathbf{z} \in \mathbf{M}} \Big(\sum_{T_k \in V_i} |\mathbf{x}_k - \bar{\mathbf{z}}_i|^2 |T_k| + \sum_{T_k \in V_i} |\bar{\mathbf{z}}_i - \mathbf{z}|^2 |T_k| \Big) \tag{6}$$

where $\bar{\mathbf{z}}_i = \frac{\sum_{T_k \in V_i} |T_k| \mathbf{x}_k}{\sum_{T_k \in V_i} |T_k|}$ is the mass centroid of V_i.

A proof of eq. (6) can be found in the appendix. Thus the constrained mass centroid of V_i is the point on M that is closest to its mass centroid $\bar{\mathbf{z}}_i$. Eqs. (5) and (6) are the counterparts of eqs. (1) and (2) in the discrete case. Before we describe the algorithm, two important properties have to be highlighted first.

PROPERTY 3.1. *Let $\{(V_i, \mathbf{z}_i)\}$ be the current cluster configuration where $\mathbf{z}_i$ is the constrained mass centroid of V_i, and for each triangle $T_r \in V_i$'s, let $\mathbf{x_r}$ be its centroid. If $|\mathbf{x_k} - \mathbf{z}_q|^2 < |\mathbf{x_k} - \mathbf{z}_p|^2$ for some triangle $T_k \in V_p$ and V_q adjacent to V_p, then*

$$F'(\mathbf{z}) < F(\mathbf{z})$$

where

$$F'(\mathbf{z}) = \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i'} |\mathbf{x}_k - \mathbf{z'}_i|^2 |T_k| \Big), \tag{7}$$

$\mathbf{z'}_i$ is the constrained mass center of V_i' and

$$V_i' = \begin{cases} V_i & i \neq p, q \\ V_p - \{T_k\} & i = p \\ V_q \cup \{T_k\} & i = q \ . \end{cases}$$

Note that since $|\mathbf{x_k} - \mathbf{z}_q|^2 < |\mathbf{x_k} - \mathbf{z}_p|^2$, it is clear that

$$\sum_{T_j \in V_p - \{T_k\}} |\mathbf{x}_j - \mathbf{z}_p|^2 |T_j| + \sum_{T_j \in V_q \cup \{T_k\}} |\mathbf{x}_j - \mathbf{z}_q|^2 |T_j| < \sum_{T_j \in V_p} |\mathbf{x}_j - \mathbf{z}_p|^2 |T_j| + \sum_{T_j \in V_q} |\mathbf{x}_j - \mathbf{z}_q|^2 |T_j|.$$

From the minimization property of the constrained mass centroid $\mathbf{z'}_i$, the following inequality holds:

$$\sum_{T_j \in V_t'} |\mathbf{x}_j - \mathbf{z'}_t|^2 |T_j| \leq \sum_{T_j \in V_t'} |\mathbf{x}_j - \mathbf{z}_t|^2 |T_j| \ , \quad t = p, q$$

Combining these two steps, $F'(\mathbf{z}) < F(\mathbf{z})$ follows readily.

PROPERTY 3.2. *Let $\{(V_i, \mathbf{z}_i)\}$ be the current cluster configuration, and triangles $T_k \in V_p$ and $T_s \in V_q$ with centroids $\mathbf{x}_k$ and $\mathbf{x}_s$, respectively, share a common edge. If*

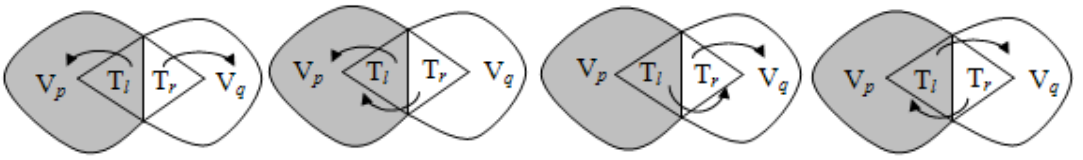

Figure 2: Illustration of 4 cases in distance comparison. The presence of an arrow indicates direction of the movement after the comparison. These are cases 1, 2, 3 and 4 from left to right in that order.

$|\mathbf{x}_k - \mathbf{z}_p|^2 > |\mathbf{x}_k - \mathbf{z}_q|^2$, $|\mathbf{x}_s - \mathbf{z}_q|^2 > |\mathbf{x}_s - \mathbf{z}_p|^2$ *and* $|\mathbf{x}_k - \mathbf{z}_p|^2|T_k| + |\mathbf{x}_s - \mathbf{z}_p|^2|T_s| < |\mathbf{x}_k - \mathbf{z}_q|^2|T_k| + |\mathbf{x}_s - \mathbf{z}_q|^2|T_s|$ *then*

$$F'(\mathbf{z}) < F(\mathbf{z})$$

where $F'(\mathbf{z})$ *is defined in eq. (7).*

This property can easily be proved following an argument similar to that of property 3.1. In fact, reassigning either T_k or T_s will lower the value of the energy functional $F(\mathbf{z})$. In a greedy spirit, we simply choose the smaller one, which is reflected by the third given inequality. One can not simply assign T_k to V_q and T_s to V_p because the result could violate the connectivity requirement for clusters.

3.1 Energy minimization

Recall that a discrete CCVT of a triangular mesh M is a minimizer of the discrete energy functional (5). In the following we propose an algorithm to iteratively reduce the value of $F(\mathbf{z})$ until a limit point is reached. The main idea of the algorithm is to update the clusters by comparing distances from triangle centroids of a cluster to mass centroids of adjacent clusters. The triangles that have to be considered are just boundary triangles, i.e., triangles sharing a *cluster edge*. A mesh edge is called a *cluster edge* if it is shared by two triangle faces of different clusters. The distance comparing procedure is stated below.

Let edge e_{lr} be a *cluster edge* in the current cluster configuration $\{(V_i, \mathbf{z}_i)\}$. e_{lr} is shared by triangles T_l and T_r, where $T_l \in V_p$ and $T_r \in V_q$ are in different clusters. Let $\mathbf{x}_l$ and $\mathbf{x}_r$ be the centroids of T_l and T_r, respectively. Denote $|\mathbf{x}_l - \mathbf{z}_p|^2$, $|\mathbf{x}_l - \mathbf{z}_q|^2$, $|\mathbf{x}_r - \mathbf{z}_p|^2$ and $|\mathbf{x}_r - \mathbf{z}_q|^2$ with d_{lp}, d_{lq}, d_{rp} and d_{rq}, respectively. We need to compare d_{lp} with d_{lq}, and d_{rp} with d_{rq}, totally four cases. Figure 2 illustrates these 4 cases.

1. $d_{lp} \le d_{lq}$ and $d_{rp} \ge d_{rq}$.
   Do nothing. This is exactly what the convergent state should be.
2. $d_{lp} \le d_{lq}$ and $d_{rp} < d_{rq}$.
   Move T_r to V_p. According to property 3.1, this movement lowers the value of the energy functional $F(\mathbf{z})$.
3. $d_{lp} > d_{lq}$ and $d_{rp} \ge d_{rq}$.
   Move T_l to V_q. The new value of the energy functional $F(\mathbf{z})$ will be lower, according to property 3.1.
4. $d_{lp} > d_{lq}$ and $d_{rp} < d_{rq}$.
   One more test is needed to decide which triangle should be moved.
   - If $d_{lp}|T_l| + d_{rp}|T_r| < d_{lq}|T_l| + d_{rq}|T_r|$, move T_r to V_p.
   - Otherwise, move T_l to V_q.

   The value of the energy functional $F(\mathbf{z})$ will be lower after the movement, according to property 3.2.

Based on this distance comparison process for a single it cluster edge, one can derive an algorithm which updates the mass centroids of the clusters immediately after finishing the above comparison process for each *cluster edge*. This algorithm should work because the energy functional decreases after the distance comparison process for each *cluster edge*. The problem with this algorithm is, it involves too many mass centroid updating steps for clusters. Instead, we propose an algorithm which would update the mass centroids of the clusters only after we finish distance comparison for all the *cluster edges* in the current cluster configuration. We call such a scheme *configuration-wise updating*. Correctness of such an approach is verified below.

Let $\{(V_i, \mathbf{z}_i)\}$ be the current cluster configuration. Before the distance comparison process starts, two triangle sets V_i^+ and V_i^- are attached to each cluster V_i to record information during the distance comparison process. V_i^+ records triangles not belonging to V_i initially but are moved to V_i somewhere during the comparison process. V_i^- records triangles belonging to V_i initially but are moved to other clusters somewhere during the comparison process. Note that if $T_k \in V_i^+$ then there exists a j such that $|\mathbf{x}_k - \mathbf{z}_i|^2 < |\mathbf{x}_k - \mathbf{z}_j|^2$. And if $T_k \in V_i^-$ then there exists a j such that $|\mathbf{x}_k - \mathbf{z}_j|^2 < |\mathbf{x}_k - \mathbf{z}_i|^2$. It is clear that $V_i^+ \cap V_i^- = \emptyset$. After the distance comparison process is done for all *cluster edges*, the new cluster V_i' can be written as

$$V_i' = (V_i \cup V_i^+) - V_i^-$$

We claim that the new energy functional is smaller, i.e., $F'(\mathbf{z}) < F(\mathbf{z})$. This is shown below:

$$\begin{aligned}
F'(\mathbf{z}) &= \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i'} |\mathbf{x}_k - \mathbf{z'}_i|^2 |T_k| \Big) \\
&< \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i'} |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| \Big) \\
&= \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i - V_i^-} |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| + \sum_{T_k \in V_i^+} |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| \Big) \\
&< \sum_{i=1}^{n} \Big(\sum_{T_k \in V_i - V_i^-} |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| + \sum_{T_k \in V_i^-} |\mathbf{x}_k - \mathbf{z}_i|^2 |T_k| \Big) \\
&= F(\mathbf{z})
\end{aligned}$$

where $\mathbf{z'}_i$ is the constrained mass centroid of V_i' and $\mathbf{z}_i$ is the constrained mass centroid of V_i. The first inequality follows

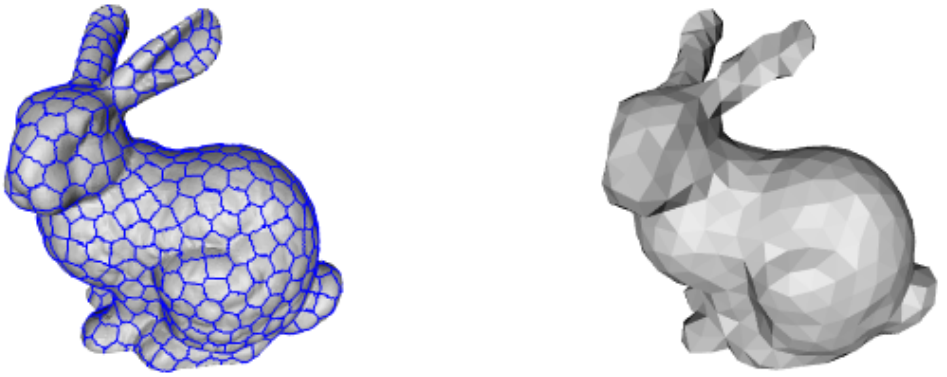

Figure 3: The left figure has 500 clusters. The right figure is a coarsened mesh by CVT.

from optimality of the constrained mass centroid, and the second inequality follows from properties of V_i^+ and V_i^-. Thus the new energy functional decreases after the one-time updating.

The constrained mass centroid $\mathbf{z}_i$ of the cluster V_i is the closest point from M to the mass centroid $\bar{\mathbf{z}}_i$. $\bar{\mathbf{z}}_i$ plays an important role in getting $\mathbf{z}_i$. In the following we derive a recursive formula to update the mass centroid $\bar{\mathbf{z}}_i$:

$$\begin{aligned}\bar{\mathbf{z}}_i' &= \frac{\sum_{T_j\in V_i'}|T_j|\mathbf{x}_j}{\sum_{T_j\in V_i'}|T_j|} \\ &= \frac{\sum_{T_j\in V_i}|T_j|\mathbf{x}_j+\sum_{T_j\in V_i^+}|T_j|\mathbf{x}_j-\sum_{T_j\in V_i^-}|T_j|\mathbf{x}_j}{\sum_{T_j\in V_i'}|T_j|} \\ &= \frac{\sum_{T_j\in V_i}|T_j|}{\sum_{T_j\in V_i'}|T_j|}\bar{\mathbf{z}}_i + \frac{\sum_{T_j\in V_i^+}|T_j|-\sum_{T_j\in V_i^-}|T_j|}{\sum_{T_j\in V_i'}|T_j|}\mathbf{var}(\bar{\mathbf{z}}_\mathbf{i})\end{aligned} \tag{8}$$

where

$$\mathbf{var}(\bar{\mathbf{z}}_\mathbf{i}) = \frac{\sum_{T_j\in V_i^+}|T_j|\mathbf{x}_j - \sum_{T_j\in V_i^-}|T_j|\mathbf{x}_j}{\sum_{T_j\in V_i^+}|T_j| - \sum_{T_j\in V_i^-}|T_j|}$$

is the variation of the mass centroid $\mathbf{z}_i$. If the denominator equals zero, the new centroid will be computed directly. These are what we will record during the distance comparison process for *cluster edges*.

The above comprehensive analysis induces an efficient clustering algorithm. With a valid initial cluster configuration, we perform distance comparison for each *cluster edge* and record the centroid variations of adjacent clusters at the same time. After completing the distance comparison process for all *cluster edges*, we update the mass centroids of clusters using eq. (8) and update the *cluster edge* set. This process is iterated until the *cluster edge* set no longer changes.

It is obvious that the energy functional $F(\mathbf{z})$ has a global minimum on the triangular mesh M. As $F(\mathbf{z})$ decreases strictly after each configuration-wise updating, it is guaranteed to converge to a limit point. But the "minimum" it achieves might not be the global minimum of $F(\mathbf{z})$. For our clustering goal, it doesn't matter much. The limit cluster configuration always gives a very good clustering of M.

Remark: Although our results are for discrete CCVT on M, there are parallel results for discrete CVD on $M \subset \Re^3$. Because the mass centroid $\bar{\mathbf{z}}_i = \frac{\sum_{T_k\in V_i}|T_k|\mathbf{x}_k}{\sum_{T_k\in V_i}|T_k|}$ of a cluster V_i on a triangular mesh M can also be viewed as a solution to the minimization problem

$$\min_{\mathbf{z}\in\Re^3}\sum_{T_k\in V_i}|\mathbf{x}_k-\mathbf{z}|^2|T_k|$$

This is true because we have

$$\sum_{T_k\in V_i}|\mathbf{x}_k-\mathbf{z}|^2|T_k| = \sum_{T_k\in V_i}|\mathbf{x}_k-\bar{\mathbf{z}}_i|^2|T_k| + \sum_{T_k\in V_i}|\bar{\mathbf{z}}_i-\mathbf{z}|^2|T_k|$$

Thus it is obvious that $\bar{\mathbf{z}}_i$ is the solution to this minimization problem. We present the approximated results by CVT on the bunny model in Figure 3. The discrete CVT method runs much faster than the discrete CCVT method because the CCVT method needs to find the closest points in each iteration. Our examples in this paper are mainly the results from the discrete CVT method.

Approximation of CVT on triangular meshes is also thoroughly discussed in [23, 24]. Especially, anisotropic approximation of CVT is also presented in [24]. The new algorithm is different from those in [23, 24] in two folds. First, the new algorithm has a simpler local geometric operation for cluster boundary test, namely, distance comparison. The methods in [23, 24] involve computation of certain terms of the energy functional, i.e. $L_{iso,i} = |\mathbf{z}_i|^2\sum_{T_k\in V_i}|T_k| - 2\mathbf{z}_i^T\sum_{T_k\in V_i}|T_k|\mathbf{x}_k$ for each cluster V_i in [24]. Their algorithm for $L_{iso,i}$ works as follows. For each *cluster edge* whose adjacent triangles are $T_i \in V_p$ and $T_j \in V_q$, their algorithm computes $L_{iso,p} + L_{iso,q}$ for 3 cases: 1) $T_i \in V_p$ and $T_j \in V_q$; 2) $T_i \in V_p$ and $T_j \in V_p$; 3) $T_i \in V_q$ and $T_j \in V_q$. The algorithm does the reassignment according to the minimum $L_{iso,p} + L_{iso,q}$ of the 3 cases. The involved computation is less intuitive and lack of geometric meanings. Second, the new algorithm updates the cluster configuration after comparing all the *cluster edges*, while those algorithms in [23, 24] update the clusters after comparing each *cluster edge*. The experimental differences will be discussed in Section 6 (Applications). In addition to the above two differences, the new algorithm is also applicable to the variational shape approximation problem in [2], which will be discussed in the next section.

4. BOUNDARY TESTING ALGORITHM FOR CLUSTERING WITH $L^{2,1}$ METRIC

A novel metric $L^{2,1}$ is introduced for geometric partitioning of a triangular mesh M in [2]. A geometric partition of M consists of a set of connected collections of triangles $\{R_i\}_{i=1}^n$ such that $R_i \cap R_j = \emptyset$ $(i \neq j)$, $\cup_{i=1}^n R_i = M$ and R_i is connected. For each region R_i, we can define a proxy plane $P_i = (\overline{\mathbf{X}}_i, \overline{\mathbf{N}}_i)$, where $\overline{\mathbf{X}}_i$ is the average point of the centroids and $\overline{\mathbf{N}}_i$ is the average normal of the triangles in R_i. Given a region R_i and its associated proxy plane P_i, the $L^{2,1}$ for R_i is defined as :

$$L^{2,1}(R_i, P_i) = \int\int_{x \in R_i} |\mathbf{n}(x) - \overline{\mathbf{N}}_i|^2 dx$$

where $\mathbf{n}(x)$ is the normal of $x \in R_i$. For triangular mesh, it can be precisely written as

$$L^{2,1}(R_i, P_i) = \sum_{T_k \in R_i} |\mathbf{n}_k - \mathbf{N}_i|^2 |T_k|$$

where $\mathbf{n}_k$ is the unit normal of triangle T_k and $\mathbf{N}_i$ is the normalized vector of $\sum_{T_k \in R_i} \mathbf{n}_k |T_k|$.

Based on the $L^{2,1}$ metric, an optimal geometric partition of M and a given partition number n can be defined as the minimizer of the distortion error:

$$E(M, P) = \sum_{i=1}^{n} L^{2,1}(R_i, P_i)$$

The advantage of using $L^{2,1}$ metric for shape approximation is thoroughly discussed in [2], i.e. anisotropy capturing. An algorithm for constructing an optimal geometric partition is also proposed in [2]. This algorithm always produces a good partition of M, but it is pointed out in [2] that it is not guaranteed to converge. Here, we develop a different algorithm for constructing an optimal geometric partition which minimizes the distortion error. This algorithm is based on boundary testing and distance comparison. It is very fast and convergent.

Before proving convergence of the algorithm, we explore the minimization property of $\mathbf{N}_i$ of the proxy plane first. Precisely, $\mathbf{N}_i$ is the solution to the minimization problem:

$$\min_{|\mathbf{N}|=1} \sum_{T_k \in R_i} |\mathbf{n}_k - \mathbf{N}|^2 |T_k|$$

Note that, similar to eq. (6), we have

$$\sum_{T_k \in R_i} |\mathbf{n}_k - \mathbf{N}|^2 |T_k| = \sum_{T_k \in R_i} |\mathbf{n}_k - \overline{\mathbf{N}}_i|^2 |T_k| + \sum_{T_k \in R_i} |\overline{\mathbf{N}}_i - \mathbf{N}|^2 |T_k|$$

where $\overline{\mathbf{N}}_i = \frac{\sum_{T_k \in R_i} \mathbf{n}_k |T_k|}{\sum_{T_k \in R_i} |T_k|}$. And it is obvious that

$$|\mathbf{N}_i - \overline{\mathbf{N}}_i|^2 = \min_{|N|=1} |\mathbf{N} - \overline{\mathbf{N}}_i|^2$$

Thus the minimization property of $\mathbf{N}_i$ follows. As far as a geometric partition is concerned, a region of a geometric partition is nothing but a cluster as we have discussed in the previous section. Thus boundary edges between different regions are just like *cluster edges* between different clusters. We will use the term *cluster edge* here too. Then for each *cluster edge*, there are also two properties similar to properties 3.1 and 3.2 for discrete CCVT.

PROPERTY 4.1. *Let $\{(R_i, \mathbf{N}_i)\}$ be the current geometric partition, and triangle $T_k \in R_p$ with the unit normal $\mathbf{n}_k$. If $|\mathbf{n}_k - \mathbf{N}_p|^2 > |\mathbf{n}_k - \mathbf{N}_q|^2$ for some R_q adjacent to R_p, then*

$$E(M, R') < E(M, R)$$

where $E(M, R') = \sum_{i=1}^{n} \left(\sum_{T_k \in R'_i} |\mathbf{n}_k - \mathbf{N'}_i|^2 |T_k| \right)$, $\mathbf{N'}_i$ is the normalized vector of $\overline{\mathbf{N}}_i$ of R'_i and

$$R'_i = \begin{cases} R_i & i \neq p, q \\ R_p - \{T_k\} & i = p \\ R_q \cup \{T_k\} & i = q \end{cases}$$

PROPERTY 4.2. *Let $\{(R_i, \mathbf{N}_i)\}$ be the current cluster configuration, and triangles $T_k \in R_p$ and $T_s \in R_q$, with unit normals $\mathbf{n}_k$ and $\mathbf{n}_s$, respectively, share an edge e_{ks}. If $|\mathbf{n}_k - \mathbf{N}_p|^2 > |\mathbf{n}_k - \mathbf{N}_q|^2$, $|\mathbf{n}_s - \mathbf{N}_q|^2 > |\mathbf{n}_s - \mathbf{N}_p|^2$ and*

$$\begin{aligned} &|\mathbf{n}_k - \mathbf{N}_p|^2 |T_k| + |\mathbf{n}_s - \mathbf{N}_p|^2 |T_s| \\ &< |\mathbf{n}_k - \mathbf{N}_q|^2 |T_k| + |\mathbf{n}_s - \mathbf{N}_q|^2 |T_s| \end{aligned}$$

then

$$E(M, R') < E(M, R)$$

where $E(M, R')$ is the same as that in property 4.1.

The correctness of properties 4.1 and 4.2 can be easily verified like properties 3.1 and 3.2 because of the optimality of $\mathbf{N}_i$. Besides, we can also update the unit normal $\mathbf{N}_i$ of the proxy plane P_i only after distance comparison for all *cluster edges*. The validity of such a configuration-wise updating is again based on the optimality of $\mathbf{N}_i$, same as the discrete CCVT. Another parallel result from discrete CCVT is that these $\mathbf{N}_i$ can be computed recursively. $\mathbf{N}_i$ of each proxy plane is of unit length. We can get it by normalizing the weighted normal $\overline{\mathbf{N}}_i$ for each region. We keep recording these $\overline{\mathbf{N}}_i$ for regions. Then we can update $\overline{\mathbf{N}}_i$ using a similar formula as eq. (8) by introducing the variation normal $\mathbf{var}(\overline{\mathbf{N}}_i)$ for each region during the distance comparison process.

Now we can deduce a convergent iterative algorithm for computing the optimal geometric partition with respect to the $L^{2,1}$ metric. This algorithm is similar to the boundary testing algorithm for CCVT. Starting with a valid initial geometric partition of M, we perform distance comparison for each *cluster edge* and record the variation normal $\mathbf{var}(\overline{\mathbf{N}}_i)$ for each region at the same time. We then update the proxy unit normal $\overline{\mathbf{N}}_i$ by $\mathbf{var}(\overline{\mathbf{N}}_i)$ and update the cluster edge sets. This process iterates until the cluster edge set doesn't change any more.

Note that this algorithm converges to a "minimum" of the distortion error $E(M, R)$. But it might not be the global minimizer of $E(M, R)$. Nevertheless, test results show that this algorithm always converges to a near-optimal result quickly. For geometric partitioning, it would not be a problem to get a near-optimal partition. The theoretical advantage of the new algorithm is that it is guaranteed to convergent. We will discuss practical performance of this algorithm in the application section.

Figure 4: Illustration of edge contraction in dual graph (defined by solid edges) [19]. Left: dual graph before edge contraction; Right: dual graph after edge contraction, where two nodes in shaded area are merged into a single node.

5. ACCELERATION STRATEGIES FOR IMPLEMENTATION

Several strategies can be used to accelerate the clustering process. We will explore several possibilities here, such as the ones considered in [24].

5.1 Initialization

Our iterative algorithm always begins with a valid initial cluster configuration, i.e., the clusters are connected and non-overlapping. A good initialization can reduce the clustering time significantly. We apply the hierarchical face clustering idea in [19] to design our cluster initialization. Hierarchical face clustering respects the connected requirement of clusters strictly. In the following we introduce a different edge contraction criterion for each distortion metric.

The hierarchical face clustering is to partition the faces of a triangular mesh into different connected sets of faces. It builds such a hierarchical structure on the dual graph of the mesh. The dual graph is constructed by mapping each triangle in the mesh to a vertex in the dual graph, and generating an edge to connect two vertices if the corresponding triangles in the original mesh are adjacent. Then edge contraction is applied on the dual graph iteratively. An edge contraction merges two dual vertices into a single vertex. Figure 4 illustrates this concept. This means grouping two sets of faces into a single cluster. After each contraction, the dual graph is updated by replacing the two vertices with one vertex and associating edges adjacent to these two vertices to the new vertex. The edge chosen for contraction is based on a cost function. In [19], the cost function is the planarity criterion. For our algorithm, we define the cost function for an edge e_{ij} that connects $(V_i, \mathbf{z}_i, |V_i|)$ and $(V_j, \mathbf{z}_j, |V_j|)$ as:

$$F(e_{ij}) = \frac{|\mathbf{z}_{ij} - \mathbf{z}_i|^2 |V_i| + |\mathbf{z}_{ij} - \mathbf{z}_j|^2 |V_j|}{|V_i| + |V_j|}$$

where $|V_p| = \sum_{T_k \in V_p} |T_k|$ $(p = i, j)$, $\mathbf{z}_i$ and $\mathbf{z}_j$ are the "mass centroids" of V_i and V_j, respectively, and $\mathbf{z}_{ij} = \frac{\mathbf{z}_i|V_i| + \mathbf{z}_j|V_j|}{|V_i| + |V_j|}$. $\mathbf{z}_i$ depends on our distortion error metric. It is the mass centroid for the CVT constructing case and is the unit normal of the proxy plane for the optimal geometric partitioning case. The edge cost function $F(e_{ij})$ is just the energy functional $F(\mathbf{z})$ when there are only two faces V_i and V_j.

Given the cluster number n, we first decide an number k such that the face number N_f of the mesh satisfies $n2^k < N_f \leq n2^{k+1}$. Then we carry out $k+1$ levels of edge contraction on the dual graph. For the initial level, we will contract $N_f - n2^k$ edges to reduce the nodes in dual graph to $n2^k$. For the next k levels of edge contraction, we will contract half of of edges in current dual graph to reduce half of the nodes. In the end, we will get exactly n clusters. It is always possible to sort the dual edges according to the value of its cost function. In order to speed the initialization process, we only sort the dual edges in the last few levels which have mush less edges. Our hierarchical initialization generates exactly n connected clusters and accelerates the clustering convergence.

5.2 Accelerators

Our algorithm converges. Thus only a few *cluster edges* need to be modified when the limit point is close. As pointed out in [24], one highly efficient strategy is to keep tracking whether a cluster is about to settle down. If it becomes static, then we don't perform the distance comparison for *cluster edges* adjacent to this cluster any more. Another accelerating strategy is to trace potential *cluster edges* while doing distance comparison for current *cluster edges*. Precisely speaking, if a current *cluster edge* is not modified, it means the two triangles sharing this edge are not reassigned. Then we simply take itself as a potential edge. If it is modified, then we consider (the five) edges of the two adjacent triangles as potential edges. This potential edge tracking strategy also accelerates the clustering process.

5.3 Validity of clusters

As stated before, a valid cluster must be connected, but our algorithm does not guarantee that the resulting clusters are connected. In practice, a cluster might end up consisting of several distinct connected clusters. Once such a situation happens, we keep the largest non-isolated cluster and reassign triangles to other clusters that are closest to them. Another exception is that a cluster may be isolated, which means it is adjacent to a single cluster only. In such a situation, we just reassign the triangles to their closest clusters. In our experiments, these exceptions rarely happened when clustering by discrete CCVT or CVT. But they happened quite often when clustering by $L^{2,1}$ metric.

6. APPLICATIONS

Depending on the clustering criteria, a mesh M can either be uniformly coarsened after the construction of a discrete CVT or CCVT, or be approximated in an anisotropic fashion following the construction of an optimal geometric partition with respect to an $L^{2,1}$ metric. Uniform mesh coarsening and anisotropic approximation for $L^{2,1}$ metric are discussed in details in [23, 24] and [2], respectively. We will explore both techniques in our applications as well.

6.1 Uniform Mesh Coarsening

Once clustering of a mesh M by approximating CVT or CCVT is done, the following work of getting a coarsened

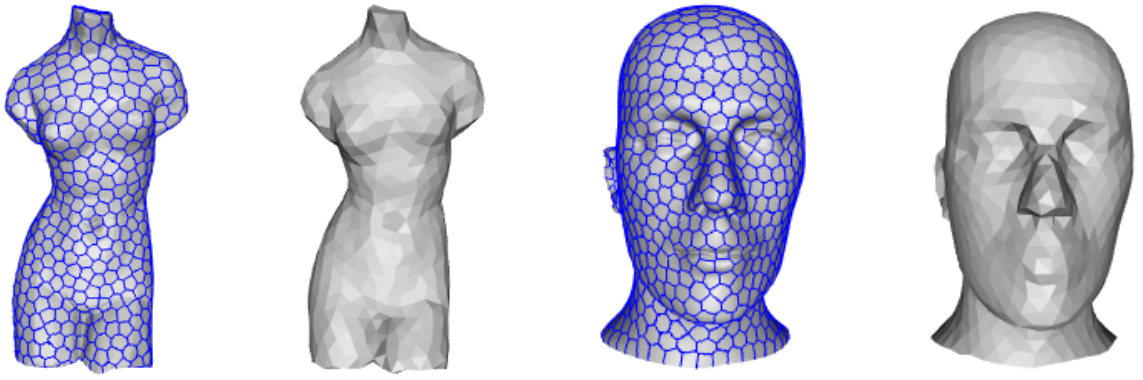

Figure 5: Results of uniform coarsening on a statue and a head model. The left most and the second from right figures are meshes with 800 clusters and 1000 clusters, respectively. The second from left and the right most figures are coarsened results.

Models	#F (org)	#V (org)	#V (approx)	time(s)	min $\angle$	Ave. min $\angle$	$\angle < 30°$	$Q_{\min}$	Q_{ave}
hand	72.9k	36.6k	1000	0.102	31.177	49.5895	0	0.580444	0.868164
head	214k	107k	1000	2.11	34.3362	52.1764	0	0.597016	0.903267
bunny	69.4k	34.8k	500	0.095	31.3123	50.321	0	0.581436	0.879907
sphere	131k	65.5k	200	0.388	38.549	53.329	0	0.640395	0.915695
horse	354k	177k	1.5k	6.24637	36.6344	51.6152	0	0.615523	0.897908
statue	272k	136k	800	1.457	32.0611	51.9163	0	0.541623	0.900297

Table 1: Results for uniform mesh coarsening

triangular mesh is relatively simple. The process is like a Delaunay triangulation, with each cluster treated as a logical vertex. The process is illustrated below.

First, a vertex is created for each cluster. Several techniques can be used here. One approach is, for each cluster, take the vertex that is closest to its mass centroid [23]. A second choice is to use the Quadric Error Metric to compute a vertex [24]. The first approach is used here since it conforms more with the spirit of CCVT on a mesh.

The second step is to triangulate the vertices created in the above step. Delaunay triangulation is used here. Two vertices are connected with an edge in the coarsened mesh if the corresponding clusters of these vertices are adjacent to each other. Thus a vertex point that is shared by three clusters corresponds to a triangle in the coarsened mesh. Degenerate cases, however, would arise if a vertex point is shared by more than 3 clusters. The solution to such degenerate cases is quite simple. If a vertex point is shared by n (≥ 4) clusters, then there would be an n-side polygon in the coarsened mesh corresponding to this vertex point. We simply triangulate this polygon to get $n-2$ triangles. Hence the output coarsened mesh is always a triangular mesh. Examples are shown in the next section.

6.2 Anisotropic Shape Approximation

After getting an optimal geometric partition with respect to the $L^{2,1}$ error metric, one can use the strategy presented in [2] to construct an anisotropic polygonal mesh to approximate the original mesh.

The first step is *anchor vertex* creation. An *anchor vertex* corresponds to a vertex in the original mesh that is shared by more than 3 clusters. The position of the anchor vertex is the average of the projections of the corresponding vertex to its shared clusters. The next step is *edge extraction*. A vertex corresponding to an anchor vertex is connected to other such vertices by *cluster edges* in the original mesh. We connect two anchor vertices if their corresponding vertices are linked with *cluster edges* and there are no other such vertices between them. Because of anisotropic property, an edge between two anchor vertices may not faithfully capture the geometry on the original mesh. Therefore, sometime it is necessary to insert new anchor vertices between two anchor vertices to get more edges according to some criteria such as: $d \cdot \sin(\mathbf{N}_i, \mathbf{N}_j) / \|(\mathbf{a}, \mathbf{b})\|$ is less than a threshold, where $\|(\mathbf{a}, \mathbf{b})\|$ is the distance between $\mathbf{a}$ and $\mathbf{b}$, d is the largest distance from the vertices on the boundary arc in the original mesh to $(\mathbf{a}, \mathbf{b})$ and $(\mathbf{N}_i, \mathbf{N}_j)$ is the angle between the two adjacent clusters R_i and R_j. After edge contraction, *triangulation* is performed. A multi-source Dijkstra's shortest path algorithm is applied to do the pseudo-Constrained Delaunay Triangulation, which will preserve the edges on the boundary in the final triangulation. In this algorithm, each anchor vertex represents a color. This algorithm first colors vertices in the original mesh which are on the boundary with the color of the closest anchor vertex. Then do color assignment for interior points in each cluster. Then for each triangle in the original mesh whose three vertices have distinct colors, we create a triangle in the approximating mesh whose edges are connected according to the colors in the referring triangle. More on postprocessing is discussed in detail in [2]. To clearly show the anisotropic effect, we only show polygonal mesh representations of the examples used in this paper.

7. RESULTS

Models	#F (org)	#V (org)	#F (approx)	time(s)
hand	72.9k	36.6k	98	1.542
face	98.4k	49.3k	119	1.084
fandisk	12.9k	6.4k	32	0.032
monster	130k	65k	120	1.8327

Table 2: Results for anisotropic mesh approximation

The algorithms presented in this paper are implemented on a laptop computer with 1G memory and Intel Core 2 CPU T7200 under Windows. Performance data for uniform mesh coarsening applications are collected in Table 1. Experimental data for anisotropic shape approximation are gathered in Table 2. Notice that the new algorithms run very fast. It takes only a few seconds to get the job done for a mesh with more than 200k faces.

For the uniform mesh coarsening application, the quality of the output mesh M is measured in several aspects, as listed in Table 1. '$\min\angle$' stands for the minimum angle degree of the triangle faces in M. Similarly, 'Ave. $\min\angle$' computes the average minimum angle degree. '$\angle < 30^\circ$' counts the number of angles smaller than 30 degrees. '$Q_{\min}$' (minimal quality) and 'Q_{ave}' (average quality) measure the triangle shapes. Both terms are defined in [5]. The examples have also been tested using the program provided by the authors of [23, 24]. The execution times for models bunny and hand are 0.328s and 0.266s, respectively, which are slower than the new algorithm. However, the execution times on the statue model and the horse model are 0.954s and 1.547s, respectively, which are better than the new algorithm. But the output meshes of the new algorithms always have better mesh quality. For instance, for the sphere model, the data generated by the program of [23, 24] are $\min\angle = 37.7732$, Ave. $\min\angle = 52.8697$, $\angle < 30^\circ = 0$, $Q_{\min} = 0.710607$ and $Q_{ave} = 0.9096$. Data generated by the new algorithm are better.

From Table 2, one can see that the new algorithm runs very fast and gives good approximation results. This shows that the new algorithm is practical. The new algorithm contracts one face for each cluster. Then the number of clusters in the clustering result for each mesh is just the '#F (approx)' in Table 2, such as 32 clusters for the fandisk model. The anisotropic nature of the $L^{2,1}$ error metric is demonstrated in these examples.

8. CONCLUSIONS

In this paper we propose a novel clustering algorithm for a polygonal mesh M by approximating CVT or CCVT on M. The new clustering algorithm is also suitable for clustering construction with respect to the $L^{2,1}$ error metric. We present a rigorous mathematical analysis for the new algorithm. Our algorithm possesses the intrinsic distance comparison as the local geometric operation, which is simpler and more intuitive than those used in [23, 24]. Moreover, our algorithm updates the cluster configuration only after comparing all *cluster edges*. The proposed algorithm based on Lloyd method for constructing the optimal geometric partition in [2] is not guaranteed to converge. But the new algorithm is proved to converge for constructing discrete CCVT and CVT on M or clustering with $L^{2,1}$ metric. Although the new algorithm runs more or less as those in [24], the coarse mesh produced by the new algorithm has a better mesh quality. Depending on the clustering criteria, we show examples for both isotropic and anisotropic mesh approximations. The anisotropic mesh approximation by using CVT is also investigated in [24]. It seems an interesting problem to generalize the new algorithm for the anisotropic case. This will be investigated in the future.

APPENDIX

A. INTEGRATION OVER A TRIANGLE

Suppose the triangle T_k has three vertices $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$. Let $\mathbf{x} \in T_k$. Then we have $\mathbf{x} = \lambda_1(\mathbf{x})\mathbf{x}_1 + \lambda_2(\mathbf{x})\mathbf{x}_2 + \lambda_3(\mathbf{x})\mathbf{x}_3$, where $\lambda_i(\mathbf{x})(i = 1, 2, 3)$ are the barycentric coordinates of $\mathbf{x}$. We have $\sum_{i=1}^{3} \lambda_i(\mathbf{x}) = 1$. Then

$$\begin{aligned}
\int_{T_k} |\mathbf{x} - \mathbf{z}_i|^2 d\mathbf{x} &= \int_{T_k} |\lambda_1(\mathbf{x})\mathbf{x}_1 + \lambda_2(\mathbf{x})\mathbf{x}_2 + \lambda_3(\mathbf{x})\mathbf{x}_3 - \mathbf{z}_i|^2 d\mathbf{x} \\
&= \int_{T_k} |\sum_{j=1}^{3} \lambda_j(\mathbf{x})(\mathbf{x}_j - \mathbf{z}_i)|^2 d\mathbf{x} \\
&= \sum_{j=1}^{3} \int_{T_k} \lambda_j(\mathbf{x})^2 |\mathbf{x}_j - \mathbf{z}_i|^2 d\mathbf{x} \\
&\quad +2 \int_{T_k} \lambda_1(\mathbf{x})\lambda_2(\mathbf{x}) < \mathbf{x}_1 - \mathbf{z}_i, \mathbf{x}_2 - \mathbf{z}_i > d\mathbf{x} \\
&\quad +2 \int_{T_k} \lambda_1(\mathbf{x})\lambda_3(\mathbf{x}) < \mathbf{x}_1 - \mathbf{z}_i, \mathbf{x}_3 - \mathbf{z}_i > d\mathbf{x} \\
&\quad +2 \int_{T_k} \lambda_2(\mathbf{x})\lambda_3(\mathbf{x}) < \mathbf{x}_2 - \mathbf{z}_i, \mathbf{x}_3 - \mathbf{z}_i > d\mathbf{x}
\end{aligned}$$

Using the fact that $\int_{T_k} \lambda_i(\mathbf{x})\lambda_j(\mathbf{x})d\mathbf{x} = |T_k|/12$ $(i \neq j)$ and $\int_{T_k} \lambda_i(\mathbf{x})^2 d\mathbf{x} = |T_k|/6$ $(i = 1, 2, 3)$ (see [1]), we have

$$\int_{T_k} |\mathbf{x}-\mathbf{z}_i|^2 d\mathbf{x} = \frac{|T_k|}{6}\Big(\sum_{j=1}^{3} |\mathbf{x}_j-\mathbf{z}_i|^2 + \sum_{1\le r<s\le 3} < \mathbf{x}_r-\mathbf{z}_i, \mathbf{x}_s-\mathbf{z}_i > \Big)$$

This equality is also a special case of [14]. Consequently, we have

$$\begin{aligned}\int_{T_k} |\mathbf{x}-\mathbf{z}_i|^2 d\mathbf{x} &= \frac{|T_k|}{12}\Big(\sum_{j=1}^{3}|\mathbf{x}_j-\mathbf{x}_k+\mathbf{x}_k-\mathbf{z}_i|^2\Big)\\ &\quad+\frac{|T_k|}{12}\Big(\sum_{j=1}^{3}|\mathbf{x}_j-\mathbf{z}_i|^2\\ &\quad+\sum_{1\le r<s\le 3} 2<\mathbf{x}_r-\mathbf{z}_i,\mathbf{x}_s-\mathbf{z}_i>\Big)\\ &= \frac{|T_k|}{12}\Big(\sum_{j=1}^{3}|\mathbf{x}_j-\mathbf{x}_k|^2+3|\mathbf{x}_k-\mathbf{z}_i|^2\Big)\\ &\quad+\frac{9|T_k|}{12}|\mathbf{x}_k-\mathbf{z}_i|^2\\ &= \frac{|T_k|}{12}\Big(\sum_{j=1}^{3}|\mathbf{x}_j-\mathbf{x}_k|^2\Big)+|\mathbf{x}_k-\mathbf{z}_i|^2|T_k|\end{aligned}$$

where $\mathbf{x}_k=\frac{\sum_{i=1}^{3}\mathbf{x}_i}{3}$.

B. EXPANSION WITH THE MASS CENTROID

Let $\bar{\mathbf{z}}_i=\frac{\sum_{T_k\in V_i}|T_k|\mathbf{x}_k}{\sum_{T_k\in V_i}|T_k|}$ be the mass centroid of V_i. $\bar{\mathbf{z}}_i$ satisfies the following equation

$$\sum_{T_k\in V_i}|T_k|(\mathbf{x}_k-\bar{\mathbf{z}}_i)=0$$

Consequently,

$$\begin{aligned}\sum_{T_k\in V_i}|\mathbf{x}_k-\mathbf{z}|^2|T_k| &= \sum_{T_k\in V_i}|\mathbf{x}_k-\bar{\mathbf{z}}_i+\bar{\mathbf{z}}_i-\mathbf{z}|^2|T_k|\\ &= \sum_{T_k\in V_i}\big(|\mathbf{x}_k-\bar{\mathbf{z}}_i|^2+2<\mathbf{x}_k-\bar{\mathbf{z}}_i,\bar{\mathbf{z}}_i-\mathbf{z}>\\ &\quad+|\bar{\mathbf{z}}_i-\mathbf{z}|^2\big)|T_k|\\ &= 2<\sum_{T_k\in V_i}|T_k|(\mathbf{x}_k-\bar{\mathbf{z}}_i),\bar{\mathbf{z}}_i-\mathbf{z}>\\ &\quad+\sum_{T_k\in V_i}|\mathbf{x}_k-\bar{\mathbf{z}}_i|^2|T_k|+\sum_{T_k\in V_i}|\bar{\mathbf{z}}_i-\mathbf{z}|^2|T_k|\\ &= \sum_{T_k\in V_i}|\mathbf{x}_k-\bar{\mathbf{z}}_i|^2|T_k|+\sum_{T_k\in V_i}|\bar{\mathbf{z}}_i-\mathbf{z}|^2|T_k|\end{aligned}$$

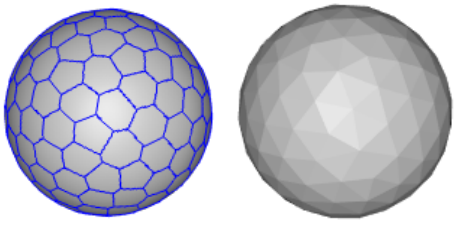

Figure 6: Clustering and coarsening results for a sphere model. Left: a sphere with 200 clusters. Right: coarsened mesh.

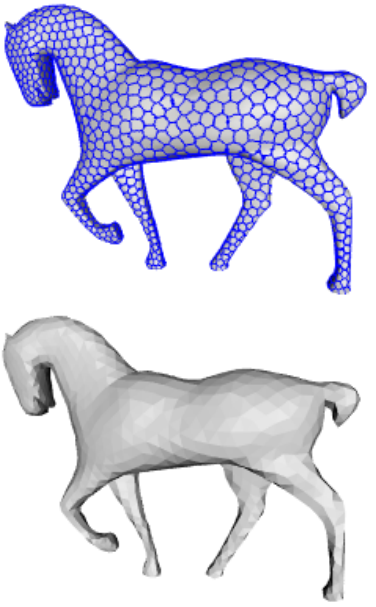

Figure 7: Clustering and coarsening results for a horse model. Top: a mesh with 1500 clusters. Bottom: coarsened mesh.

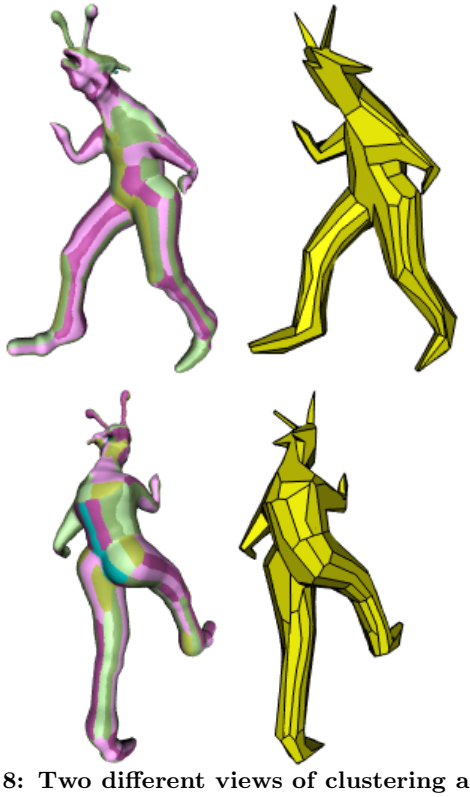

Figure 8: Two different views of clustering and approximation of a monster model with 120 clusters.

C. REFERENCES

[1] L. Chen. New analysis of the sphere covering problems and optimal polytope approximation of convex bodies. *J. Approx. Theory*, 133(1):134–145, 2005.

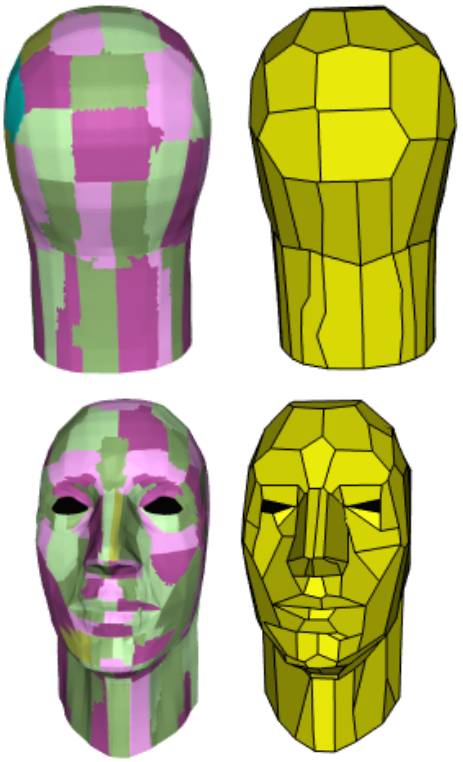

Figure 9: Two different views of clustering and approximation of a face model with 119 clusters

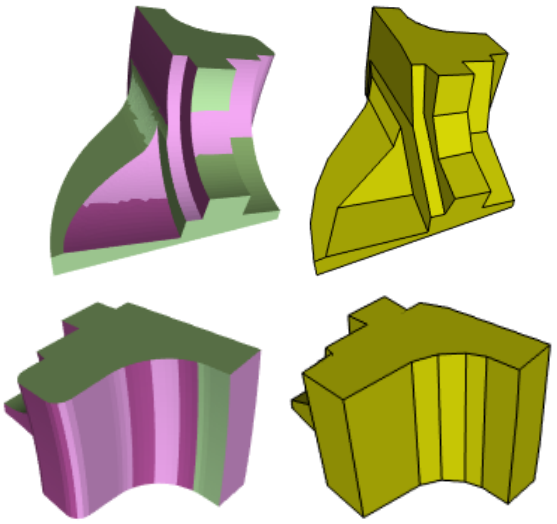

Figure 10: Two different views of clustering and approximation of a fandisk model with 32 clusters

[2] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, pages 905–914, 2004.

[3] Q. Du, V. Faber, and M. Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM Rev.*, 41(4):637–676, 1999.

[4] Q. Du, M. D. Gunzburger, and L. Ju. Constrained centroidal voronoi tessellations for surfaces. *SIAM J. Sci. Comput.*, 24(5):1488–1506, 2002.

[5] P. Frey and H. Borouchaki. Surface mesh evaluation. In *16th International Meshing Roundtable*, pages 363–374, 1997.

[6] M. Garland and P. S. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *VIS '98: Proceedings of the conference on Visualization '98*, pages 263–269, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[7] E. Grinspun and P. Schröder. Normal bounds for subdivision-surface interference detection. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 333–340, Washington, DC, USA, 2001. IEEE Computer Society.

[8] I. Guskov, K. Vidimče, W. Sweldens, and P. Schröder. Normal meshes. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 95–102, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[9] H. Hoppe. Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA, 1996. ACM.

[10] H. HOPPE, T. DEROSE, T. DUCHAMP, J. MCDONALD, and W. STUETZLE. Mesh optimization. In *ACM SIGGRAPH Proceedings*, pages 19–26, 1993.

[11] J. C. H. Jesse D. Hall. Gpu acceleration of iterative clustering. *The ACM Workshop on General Purpose Computing on Graphics Processors, and SIGGRAPH 2004 poster*, Aug. 2004.

[12] A. Kalvin and R. Taylor. Superfaces: Polygonal mesh simplification with bounded error. *IEEE Comput. Graph. Appl.*, 16(3):64–77, 1996.

[13] L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel. A shrink wrapping approach to remeshing polygonal surfaces. 18(3):119–130, 1999. (Proceedings of EUROGRAPHICS'99).

[14] J. B. Lasserre and K. E. Avrachenkov. The multi-dimensional version of $\int_a^b x^p dx$. *The American Mathematical Monthly*, 108(2):151–154, 2001.

[15] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. Maps: multiresolution adaptive parameterization of surfaces. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 95–104, New York, NY, USA, 1998. ACM.

[16] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371, 2002.

[17] P. Lindstrom and G. Turk. Fast and memory efficient

polygonal simplification. In *VIS '98: Proceedings of the conference on Visualization '98*, pages 279–286, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[18] P. Lindstrom and G. Turk. Image-driven simplification. *ACM Trans. Graph.*, 19(3):204–241, 2000.

[19] A. W. M. Garland and P. S. Heckberty. Hierarchical face clustering on polygonal surfaces. In *Proceedings of the Symposium on Interactive 3D Graphics*, 2001.

[20] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations Concepts and Applications of Voronoi Diagrams*. John Wiley & Son, 1992.

[21] A. Sheffer. Model simplification for meshing using face clustering. *Comptuer-Aided Design*, 33:925–934, 2001.

[22] G. Turk. Re-tiling polygonal surfaces. *SIGGRAPH Comput. Graph.*, 26(2):55–64, 1992.

[23] S. Valette and J.-M. Chassery. Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening. 23(3):381–389, 2004. (Proc. Eurographics'04).

[24] S. Valette, J.-M. Chassery, and R. Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):369–381, 2008.

[25] S. Valette, I. Kompatsiaris, and J.-M. Chassery. Adaptive polygonal mesh simplification with discrete centroidal voronoi diagrams. In *Proceedings of 2nd International Conference on Machine Intelligence ICMI 2005*, pages 655–662, 2005.