EUROGRAPHICS 2004 / M.-P. Cani and M. Slater
(Guest Editors)

Volume 23 (2004), Number 3

Approximated Centroidal Voronoi Diagrams for Uniform Polygonal Mesh Coarsening

Sébastien Valette and Jean-Marc Chassery

LIS, Grenoble, France

Abstract

We present a novel clustering algorithm for polygonal meshes which approximates a Centroidal Voronoi Diagram construction. The clustering provides an efficient way to construct uniform tessellations, and therefore leads to uniform coarsening of polygonal meshes, when the output triangulation has many fewer elements than the input mesh. The mesh topology is also simplified by the clustering algorithm. Based on a mathematical framework, our algorithm is easy to implement, and has low memory requirements. We demonstrate the efficiency of the proposed scheme by processing several reference meshes having up to 1 million triangles and very high genus within a few minutes on a low-end computer.

1. Introduction

3D meshes are used in a vast majority of 3D applications such as Computer Aided Design, Medical Imaging, Virtual Reality and Video Games. 3D models are constructed by designers, or can be generated automatically from real objects using 3D scanners. Nowadays, the models can have up to several million or even billion elements (vertices) and sometimes need a preprocessing step to match a given application requirements. The processing step sometimes consists in reducing the complexity of the mesh (in terms of number of elements, topology or smoothness) to accelerate rendering or transmission, increasing its elements aspect ratio (for accurate finite elements analysis), or remeshing (to meet a given connectivity constraint). As a consequence, automatic or semi-automatic geometry processing becomes increasingly important for interactions between various applications. We propose in this paper a novel surface mesh coarsening algorithm, which resamples the surface to a uniform mesh with many fewer elements than the original mesh. Our approach is based on a clustering of the original mesh cells, mimicking a Centroidal Voronoi Diagram (CVD) construction, which is theoretically the optimal strategy for resampling [DFG99]. The complexity of our algorithm (in terms of calculations and memory requirements) is low, allowing the processing of large meshes, as shown in the results section, where processing meshes with up to 1 million triangles takes only few minutes on a low-end desktop computer.

2. Previous Work

Coarsening a mesh consists in resampling the original surface with a lower number of vertices. The number of existing approaches for mesh resampling is very high. For simplicity, we can split the existing approaches in three categories: refinement, decimation, or direct approaches, which will be described more precisely, due to promising recent advances.

Refinement approaches [EDD*95, DHI92, LSS*98], approximate the original surface with a coarse mesh which is iteratively refined until a given precision is reached.

Decimation approaches, such as [GH97, Hop96, MTT97, RB93, VP04] also process the mesh iteratively, constructing several resolution levels. For a given mesh, several resolution levels are constructed by means of elementary simplifications (edge collapse or face merge, as an example), until the approximation error reaches a user-defined maximum. A survey of coarsening approaches is made in [HG97].

In opposition to the first two categories, direct approaches (or remeshing approaches) compute a mesh with a given number of elements or approximation error budget in a single resolution way. Some approaches remesh the original surface in a global parametric space [AMD02, AdVDI03, ACSD*03, GGH02] They provide good results, but are limited in practice by the parametrization step, involving heavy calculations and numerical instability. To overcome these problems, some ap-

proaches [SAG03, SG03] were proposed, involving local parametrization and optimization of the remeshed model. Other works [LRM*98, Tur92] distribute new vertices directly on the original surface mesh, to build a new tessellation which can be further optimized.

In [PC03] and [SSG03] the authors propose to remesh the model using geodesic distances: the new vertices are created using geodesic front propagation. Note that the vertices distribution can also be adapted to local curvature.

Note that remeshing approaches allow the construction of meshes with as many vertices as wanted. Indeed, mesh coarsening is not the main goal of remeshing approaches, as they permit other improvement (in terms of triangles aspect ratio) and shape adapted remeshing (e.g. adaption of the sampling according to the local curvature).

In [NT03], Nooruddin and Turk propose to simplify the mesh topology by a volumetric approach: the mesh is converted to a volumetric representation (voxels) which topology is simplified by means of morphological operations. Afterwards, the volume is re-converted to a polygonal model which is further simplified.

We can also mention out-of-core approaches for coarsening [ILGS03, WK03], used for large models which do not fit entirely inside the computer RAM.

3. Our Approach

In this paper, we propose an algorithm for mesh coarsening, which produces uniform triangulations. We only deal with triangular input meshes, but the extension to the polygonal case is straightforward. Our approach can be applied to manifold meshes with any genus and any number of holes. The first step is a clustering of the mesh cells (triangles) into an approximation of a Centroidal Voronoi Diagram (CVD), which is the main contribution of this paper. The clustering is based on an energy minimization step and a validity checking step.

The second step consists in replacing each cluster by a single vertex, and constructing the triangulation according to the clusters adjacency relations. We assume that the subsampling factor of the coarsening is high i.e. the ratio between the number of original vertices and the number of vertices of the resulting mesh is high. In this paper, we display results with meshes which number of vertices is at least divided by 20. Those high subsampling ratios enable us to formalize a new clustering approach, noticing that even if the input surface is a discrete set (the union of several polygons), it can be seen as a continuous space, as the input polygons will be small compared to the output ones. Note that our approach simultaneously simplifies the mesh geometry and its topology, and thus can be seen as a topological and geometric filter.

4. Technical Background

In this section, we make an overview of Centroidal Voronoi Diagrams (CVD) in terms of energy minimization. Supplementary details can be found in [DFG99].

4.1. Voronoi Diagrams

Given an open set Ω of $\mathbb{R}^a$, and n different sites (or seeds) $z_{i;i=0,1,\ldots,n-1}$, the Voronoi Diagram can be defined as n distinct regions V_i such that:

$$V_i = \{w \in \Omega | d(w,z_i) < d(w,z_j) j = 0,1,\ldots,n-1, j \neq i\} \quad (1)$$

where d is a function of distance. These diagrams are well known in the literature. The dual of a Voronoi Diagram is a Delaunay triangulation, which has the property that the outcircle of every triangle does not contain any other site.

4.2. Centroidal Voronoi Diagrams

A Centroidal Voronoi Diagram is a Voronoi Diagram where each Voronoi site z_i is also the mass centroid of its Voronoi Region:

$$z_i = \frac{\int_{V_i} x.\rho(x)dx}{\int_{V_i} \rho(x)dx} \quad (2)$$

where $\rho(x)$ is a density function of V_i

Moreover, Centroidal Voronoi Diagrams minimize the Energy given as:

$$E = \sum_{i=0}^{n-1} \int_{V_i} \rho(x)\|x - z_i\|^2 dx \quad (3)$$

Constructing a Centroidal Voronoi Diagram (CVD) can be done using K-means clustering and Lloyd's relaxation method [Llo82], as an example. CVDs have intrinsic properties which make them optimal for a wide range of applications[DFG99] because they optimize the compactness of the created Voronoi Regions (see equation 3.

We present here a novel approach to approximate a Centroidal Voronoi Diagram, based on the global minimization of the energy term E defined in equation 3, but which computation involves only local queries, and is therefore very fast.

5. Our clustering algorithm

5.1. Simplifying the energy term

Our augorithm is based on the construction of a clustering which minimizes equation 3. We want to construct a CVD on a discrete set: a polygonal mesh M. We first consider that

this mesh M is planar (all vertices are coplanar). Now we restrict the boundaries of each Voronoi Region V_i to be a subset of the edges of M. As a consequence, a Voronoi region is the union of several mesh cells (triangles or polygons) C_j. Note that with such restriction, the regions V_i are no more Voronoi regions in the strict sense, which is the first approximation we make: each region V_i is the union of several cells C_j, and a cell C_j is a part of one and only one region V_i. Constructing such diagram comes now as a clustering problem: we want to merge the cells C_j of the mesh M into n clusters (which look like Voronoi regions) V_i, each cluster having only 1 connected component.

We can now rewrite E by introducing the mesh cells C_i:

$$E = \sum_{i=0}^{n-1} \left(\sum_{C_j \in V_i} \int_{C_j} \rho(x) \|x - z_i\|^2 dx \right) \quad (4)$$

we now choose $\rho(x)$ to be uniform. As a consequence, $\int_{C_j} \rho(x)dx = area(C_j)$. Here comes our second approximation on E: we approximate each cell C_j by a single point : its centroid $\gamma_j = \frac{\int_{C_j} xdx}{\int_{C_j} dx}$ with a weight $\rho_j = area(C_j)$. Equation 4 now becomes :

$$E = \sum_{i=0}^{n-1} \left(\sum_{C_j \in V_i} \rho_j \|\gamma_j - z_i\|^2 \right) \quad (5)$$

As our goal is to build the CVD, we have to find where to place the Voronoi Sites z_i of each region V_i. We know that for a given CVD, each site z_i equals to the centroid of V_i. With our assumptions, the centroid $\overline{\gamma_i}$ of each cluster V_i can be easily computed as:

$$\overline{\gamma_i} = \frac{\sum_{C_j \in V_i} \rho_j \gamma_j}{\sum_{C_j \in V_i} \rho_j} \quad (6)$$

We now substitute z_i in equation 5 by $\overline{\gamma_i}$, to obtain a new Energy F:

$$F = \sum_{i=0}^{n-1} \left(\sum_{C_j \in V_i} \rho_j \|\gamma_j - \overline{\gamma_i}\|^2 \right) \quad (7)$$

At this point, F only depends on the chosen clustering of the original mesh, which is very interesting, as we no more need to explicitly compute the position of the Voronoi sites. We only face a global variance minimization problem for each region V_i, which we solve in the following section.

5.2. Energy minimization

By combining equations 7 and 6 we obtain:

$$F = \sum_{i=0}^{n-1} \left(\sum_{C_j \in V_i} \rho_j \|\gamma_j\|^2 - \frac{\left\| \sum_{C_j \in V_i} \rho_j \gamma_j \right\|^2}{\sum_{C_j \in V_i} \rho_j} \right) \quad (8)$$

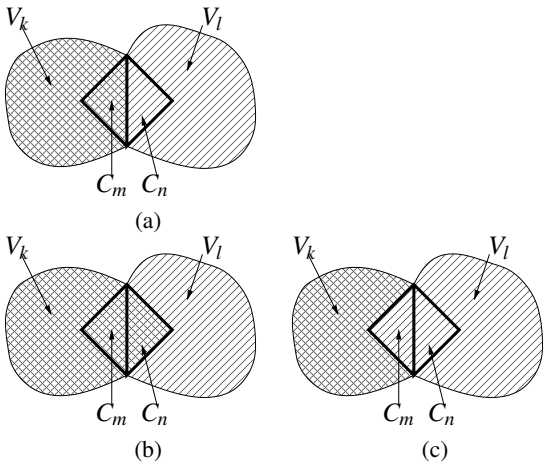

Figure 1: *The three cases of energy computation for a given edge:(a) initial configuration ($C_m \in V_k$, $C_n \in V_l$); (b) V_k grows ($C_m \in V_k$ and $C_n \in V_k$); (c) V_l grows ($C_m \in V_l$ and $C_n \in V_l$.*

We propose to minimize this energy term with an iterative algorithm that updates the clusters according to boundary tests. As the boundaries of the regions V_i is a subset of the mesh edges, a local test for each edge e between two different clusters is processed. Let us assume that a given edge e is on the boundary between two clusters V_k and V_l. The edge e has two adjacent cells C_m and C_n belonging respectively to V_k and V_l. We have to compute the value of F for three cases:

- F_{init} (the initial configuration) : C_m belongs to V_k and C_n belongs to V_l.
- F_1 (V_k grows and V_l shrinks) : both C_m and C_n belong to V_k.
- F_2 (V_k shrinks and V_l grows): both C_m and C_n belong to V_l.

Afterwards, the case resulting in the smallest energy F is chosen, and the clusters configuration is updated. By looping in the boundary edge set (the set of edges between two different clusters), we iteratively minimize F. As F is always positive and each local modification reduces F, the convergence of the algorithm is guaranteed.

Figure 1 shows the three computed energies for an edge between Cells C_m and C_n, belonging respectively to the regions V_k and V_l.

5.3. Efficient implementation

For a fast and efficient implementation, a few number of variables must be kept in memory. Indeed, each time we process the energy test, we use for each Cluster V_i the values $\sum_{C_j \in V_i} \rho_j \|\gamma_j\|^2$, $\sum_{C_j \in V_i} \rho_j$ and $\sum_{C_j \in V_i} \rho_j \gamma_j$ which are stored in three different arrays, respectively *SGamma2*, *SRho* and *SGamma*. The values of these arrays are updated iteratively,

as the clusters configuration changes. But for each test, as the envisaged clustering modification is local, only two of these three arrays are actually needed : let us assume that we want to perform the energy test on a given edge e, with two adjacent cells C_m and C_n, belonging respectively to the clusters V_k and V_l. The evolution of the energy F will only depend on the contributions of V_k and V_l to F which sum to:

$$L_1 = \sum_{C_j \in (V_k \bigcup V_l)} \rho_j \|\gamma_j\|^2 - \frac{\left\|\sum_{C_j \in V_k} \rho_j \gamma_j\right\|^2}{\sum_{C_j \in V_k} \rho_j} - \frac{\left\|\sum_{C_j \in V_l} \rho_j \gamma_j\right\|^2}{\sum_{C_j \in V_l} \rho_j} \tag{9}$$

Moreover, in the three envisaged cases, $V_k \bigcup V_l$ will remain constant. Then, we can omit to compute the first term of L_1 as it will be the same in the three cases, and the test will only consist in calculating :

$$L_2 = -\frac{\left\|\sum_{C_j \in V_k} \rho_j \gamma_j\right\|^2}{\sum_{C_j \in V_k} \rho_j} - \frac{\left\|\sum_{C_j \in Vl} \rho_j \gamma_j\right\|^2}{\sum_{C_j \in V_l} \rho_j} \tag{10}$$

for each of the three cases. We are left with an equation which does not contains, $\sum_{C_j \in V_l} \rho_j \|\gamma_j\|^2$ anymore. Finally, for fast computation, we only need two arrays: *SGamma* and *SRho* which are updated as the clustering evolves through time.

5.4. Initialization

Section 5.2 gives a way to update a given cluster configuration, but we have to build a configuration to start with. Instead of using another graph partitioning for initialization, me slightly modify our algorithm. We now assume that the number of clusters n is chosen. We randomly pick n different cells C_j in M. Each picked cell is then given one distinct cluster. All other cells do not belong to any cluster, which is equivalent to associating them to the *null* cluster. The mesh is now partitioned into $n+1$ clusters : n clusters, each containing one cell, and the *null* cluster containing all the other cells. As an example, for a triangular mesh, the boundary edge set contains $3n$ edges, assuming that all the picked cells are isolated. The loop over the boundary edge set can now start. For each boundary edge e with two adjacent cells C_m and C_n, we perform a test before computing the three energy cases defined in section 5.2: if one of the two cells belongs to the *null* cluster, it is automatically given to the other cell's cluster, without any energy computation. The given algorithm is able to compute a clustering of M from end to end.

5.5. Validity of the clustering

We add an additional constraint to our clustering algorithm: we want each cluster to be a 1-connected set of cells, to make the further triangulation construction easier. As a consequence, when the energy minimization step is finished, we have to check whether this constraint is respected. We experimentally observed that the clusters hardly ever fall into several disconnected parts. In the few encountered cases, we solved this issue as follows : for each cluster with several distinct components, we find the component with the highest area and leave it unchanged. Smaller components are associated to the *null* cluster defined in section 5.4. Afterwards, we restart the energy minimization algorithm. The number of disconnected clusters is then drastically reduced, and we can repeat this step until there is no disconnected cluster. There is no guarantee that the clustering will be valid after a given number of iterations, but experimentaly, after a maximum of five loops, the clustering was valid. Further investigations will address the theoretical conditions for such validity.

5.6. 2D examples : sampling a square

Figure 2 shows two examples of clustering with our scheme on planar triangulations. We built two different meshes, both with 20k vertices. For each mesh, the vertices coordinates are randomly chosen, but with different distributions. The vertices of the upper triangulation are uniformly distributed on the square, while the vertices of the lower triangulation are concentrated on the lower side of the square. On the left side is displayed the initial triangles pick (random cells), the right side shows the final clustering, which clusters are separated by black-painted edges. We can clearly see that the clustering is uniform on the first example, but the second configuration contains more clusters on the lower part of the square. As a consequence, our clustering is affected by the original mesh sampling, but this effect remains quite low experimentally.

5.7. Extension to the 3D case

In all above equations, all the vertices of the original mesh are coplanar. As we aim at coarsening 3D meshes, we have to extend this framework to the 3D case. The exact extension should involve the computation of geodesic distances on the mesh, as made in [PC03] and [SSG03]. But this would prevent us from using the simplifications that lead us to equation 10. As a consequence, we approximate the geodesic distance with the Euclidian distance and we can adjust our approch by processing three coordinates instead of two. This approximation introduces some error which is small in flat regions but increases with the local curvature. One interesting feature of such approximation is that it processes a filtering of the geometry in regions which contains relatively high frequency details that cannot be represented efficiently with a coarse mesh.

6. Building up the triangulation

Similarly to Voronoi Diagrams which are dual to Delaunay triangulations, we can build a triangulation by dualization of the constructed diagram.

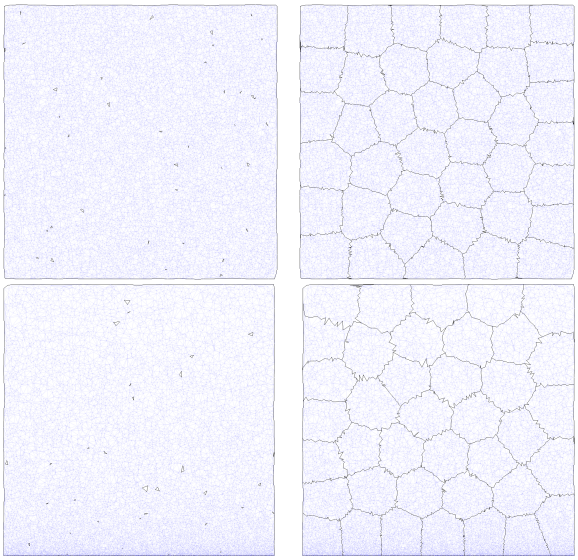

Figure 2: *Results on a planar triangulation. Top : uniform triangulation, Bottom : non uniform triangulation. Left : initial state (randomly picked triangles); right : final clustering.*

6.1. Mesh vertices

Dualizing the diagram implies the creation of a vertex for each cluster of the diagram. An easy way to compute the coordinates of a given vertex Ve_i corresponding to a given cluster V_i is to take the centroid $\overline{\gamma_i}$ of the cluster, defined in equation 6, which is already computed during the clustering step. But for convex parts, the centroid of a given cluster is inside the 3D object and outside for concave parts. To eliminate this effect, for each cluster C_i, we set its vertex coordinates to the coordinates of the original mesh vertex which is the closest to the centroid $\overline{\gamma_i}$.

6.2. Triangulation

Once the mesh geometry (vertices coordinates) is created, the construction of its connectivity is straightforward, similarly to Delaunay triangulations, where a triangle is created for each point where three Voronoi regions meet. The three triangle vertices are indeed the three Voronoi seeds. We can easily imitate this scheme by looping over the original mesh vertices, and create one triangle for each vertex lying on the boundary of three different clusters. Again, in analogy with a Delaunay Triangulation Construction (DTC), degenerate situations sometimes arise. For DTC, ambiguous cases exist when four different Voronoi seeds are isocyclic, implying that four Voronoi regions meet at one single point. As we operate on the discrete set of the original mesh cells, these cases happen more frequently than in the continuous case of DTC. But resolving these degenerate cases is easy, as for a vertex lying on the boundary of four different clusters, we

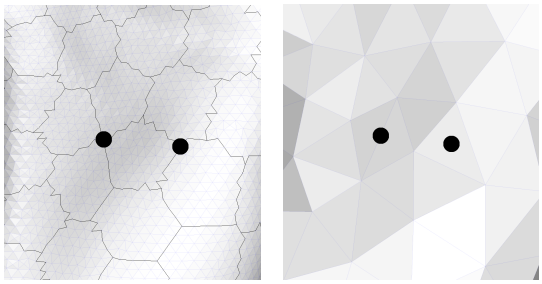

Figure 3: *Construction of the triangulation and degenerate cases. Left: Diagram; right : associated triangulation. one of the two marked original mesh vertices is adjacent to four different clusters, resulting in the creation of two triangles instead of one.*

create two triangles instead of one. More generally, if n clusters meet at one single vertex, then we create $n-2$ triangles (in practice, the highest encountered value for n was 5). Figure 6.2 shows an example of triangulation (right) constructed from the result of our clustering algorithm (left). One of the two tagged vertices is adjacent to three different clusters, resulting in the creation of a single triangle, while the other vertex is adjacent to four clusters. In this case, two triangles are created. We can clearly see that this degenerate encountered case does not penalize the constructed mesh quality.

6.3. Topology simplification

As we use a clustering approach for mesh coarsening, the resulting mesh can be topologically simpler than the original mesh. This happens when a given handle or hole is small compared to the resampling step. Moreover, flattening happens in relatively thin connected regions of the mesh. This results in a triangulation which may not be manifold. To solve this problem, we do not create the triangles appearing twice, and we flip some edges of the triangulation.

7. Results

Figure 4 shows a result obtained on a sphere with 160k faces, resampled to a mesh with 500 vertices. The left image shows the processed clustering of the original mesh. This original mesh is very irregular : a vertex on the center of the picture has its valence equal to 150. Despite this degenerated situation, the clustering remains regular and the resulting triangulation on the right is very uniform.

We have tried our approach on a large set of reference meshes having up to several hundred thousands of vertices. Figure 5 shows the results obtained when coarsening the Stanford Bunny to a mesh with 300 vertices. Figure 6 shows several coarsened versions of the original Happy Buddha model. The number of vertices for these models is respectively (from left to right) 2k, 5k, 10k and 15k. Clearly, the

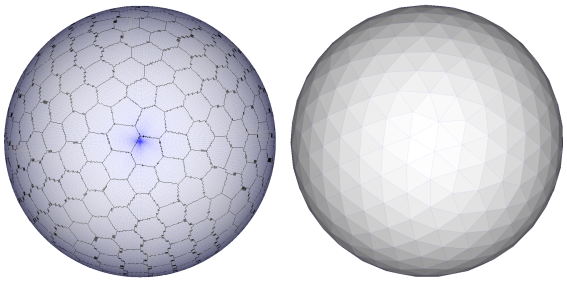

Figure 4: *Sampling a triangulated sphere : the proposed approach constructs uniform clusters (left), resulting in a very uniform coarsened sphere (right).*

constructed meshes have uniform sampling. The topology of the model was also filtered by our scheme as expected, as the original model has genus 104 and the coarsened models have their genus reduced bellow 10.

Figure 7 shows the coarsened models of the Dragon, Golf Club, Hip, Phone and David. These output meshes have respectively 20k, 5k, 20k, 2k and 20k vertices.

Figure 8 shows the results of our approach on the Max Planck model (top). This model is interesting, as there is a 'V' shape connectivity watermark on the nose. Clearly, the effect of this watermark on the results is not visually perceptible, and the constructed mesh remains regular in this region (bottom).

Table 1 shows results obtained for all the models presented in this paper. The first column is the number of vertices of the original mesh. The second one is the number of vertices of the coarsened mesh. We computed two different objective criteria to measure the quality of the output meshes. One is based on the angles of the resulting triangles (the minimal angle $\angle_{min}$, the average minimal angle $\angle_{av}$, and the percentage of angles which are less than 30 degrees $\angle < 30^o$) and the second one is based on the triangles shape (minimal quality Q_{min}, average quality Q_{av}, which ranges between 0 and 1, as defined in [FB97]). Both criteria show that our algorithm outputs meshes with high quality, which are suitable for finite elements analysis, as an example. The table also shows the Hausdorff distance (in percentage of the mesh bounding box diagonal) between the original model and the coarsened one, measured with the Metro tool[CRS98]. For the large input models, the Hausdorff distance remains below 1% and is therefore negligible, except for coarsened models with very few vertices (the Happy Buddha with 2000 and 5000 vertices). The processing times were measured on a 450Mhz PC with 512 MB RAM and show that our algorithm can run at interactive rates on high-end computers. Note that the proposed approach aims at creating uniform triangulations, so state of the art approximation algorithms may produce better meshes in terms of approximation quality.

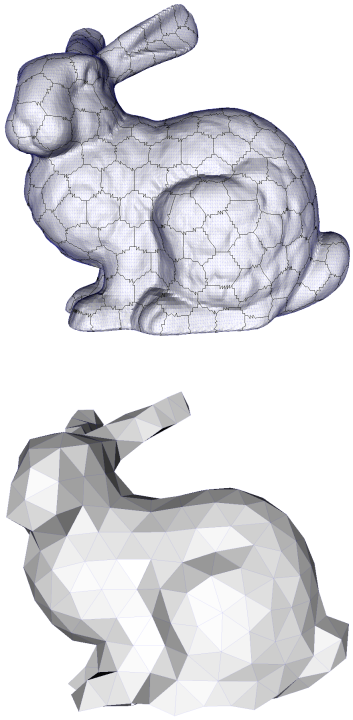

Figure 5: *Processing the Stanford Bunny. Top : approximated Centroidal Voronoi Diagram. Bottom: triangulation.*

8. Conclusion and perspectives

We proposed in this paper a fast an efficient algorithm for uniform mesh coarsening, which can be used for large models. Objective criteria show that the output meshes have good properties. The strength of our approach is that it removes handles that are small compared to the resampling step (and can then be considered as topological noise), and is therefore insensitive to meshes with degenerated topologies. Many applications can derive from our mesh construction. As an example, it can be used as a base domain for multiresolution remeshing, in spirit with [PC03]. Further work will address the processing of meshes with sharp features, and preliminary studies showed that our clustering approach can be improved in order to have an anisotropic and/or curvature-adapted behavior. This would lead us to new approaches of polygonal remeshing for large meshes, which elements would be locally adapted to the original surface.

Acknowledgements

The models used in this paper are courtesy of Leif Kobbelt, Cyberware, Stanford University and the Digital Michelangelo Project. The authors also thank Pierre Alliez and Cédric Gérot for their helpful comments on the paper.

Model	#v (original)	#v2 (coarsened)	$\angle_{min}$ (deg)	$\angle_{av}$ (deg)	$\angle < 30^o$ (%)	Q_{min}	Q_{av}	time (s)	Hausdorff dist. (% of BBox diag.)
Buddha	543k	2k	23.0	50.4	0.05	0.39	0.88	275	1.83
Buddha	543k	5k	20.3	50.0	0.03	0.37	0.88	226	1.11
Buddha	543k	10k	8.8	49.3	0.09	0.15	0.87	349	0.86
Buddha	543k	15k	15.6	48.4	0.22	0.29	0.85	439	0.84
Hip	530k	20k	26.1	49.3	0.01	0.44	0.87	595	0.26
David	507k	20k	14.6	48.5	0.10	0.22	0.86	337	0.18
Dragon	437k	20k	12.2	47.6	0.25	0.25	0.84	230	0.89
Golf Club	209k	5k	24.2	50.7	0.02	0.40	0.88	86	0.63
Phone	83k	2k	32.6	50.9	0	0.57	0.89	25	1.37
Sphere	79k	500	37.2	52.9	0	0.59	0.91	49	0.2
Bunny	70k	300	35.5	50.8	0	0.62	0.89	28	4.40
MaxPlanck	23k	1500	16.9	47.3	0.24	0.29	0.84	7	1.68

Table 1: *Results obtained on a 450Mhz PC for several reference meshes.*

Figure 6: *Several coarsened versions of the Happy Buddha model, with respectively 2k, 5k, 10k and 15k vertices.*

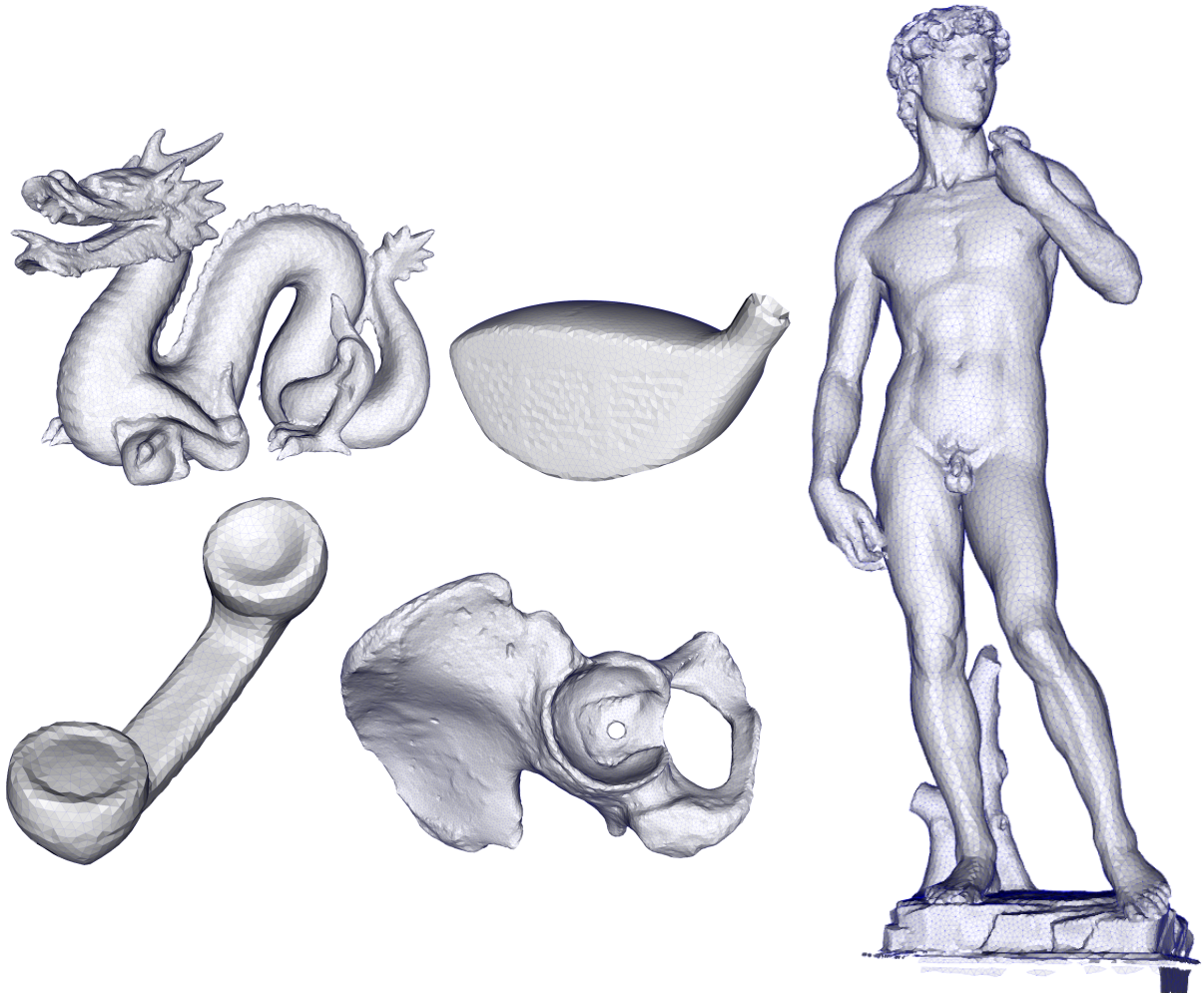

Figure 7: *Results on a set of reference meshes.*

References

[ACSD*03] ALLIEZ P., COHEN-STEINER D., DEVILLERS O., LEVY B., DESBRUN M.: Anisotropic polygonal remeshing. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference* (2003), 485–493.

[AdVDI03] ALLIEZ P., DE VERDIÈRE É. C., DEVILLERS O., ISENBURG M.: Isotropic surface remeshing. In *Proceedings of Shape Modeling International* (2003), pp. 49–58.

[AMD02] ALLIEZ P., MEYER M., DESBRUN M.: Interactive Geometry Remeshing. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference 21(3)* (2002), 347–354.

[CRS98] CIGNONI P., ROCCHINI C., SCOPIGNO R.: Metro: Measuring error on simplified surfaces. *Computer Graphics Forum 17*, 2 (1998), 167–174.

[DFG99] DU Q., FABER V., GUNZBURGER M.: Centroidal voronoi tesselations: applications and algorithms. *SIAM Review*, 41(4) (1999).

[DHI92] DELINGETTE H., HEBERT M., IKEUCHI K.: Shape representation and image segmentation using deformable surfaces. *Image and Vision Computing 10(3)* (April 1992), 132–144.

[EDD*95] ECK M., DEROSE T., DUCHAMP T., HOPPE H., LOUNSBERY M., STUETZLE W.: Multiresolution analysis of arbitrary meshes. *Computer Graphics 29*, Annual Conference Series (1995), 173–182.

[FB97] FREY P., BOROUCHAKI H.: Surface mesh

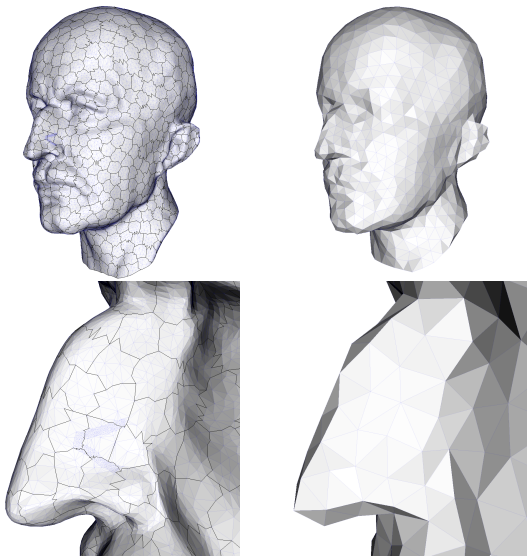

Figure 8: *Example on the Max Planck model. Top: entire mesh, bottom: closeup view of the nose, where a 'V' shaped connectivity watermark has been placed. Left : clustering, right : the resulting triangulation was not affected by the mark.*

evaluation. In *6th International Meshing Roundtable* (1997), pp. 363–374.

[GGH02] GU X., GORTLER S., HOPPE H.: Geometry images. *ACM SIGGRAPH Conference Proceedings* (2002), 355–361.

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. *Computer Graphics 31*, Annual Conference Series (1997), 209–216.

[HG97] HECKBERT P. S., GARLAND M.: Survey of polygonal surface simplification algorithms. *SIGGRAPH 97 courses notes* (1997).

[Hop96] HOPPE H.: Progressive meshes. *Computer Graphics 30*, Annual Conference Series (1996), 99–108.

[ILGS03] ISENBURG M., LINDSTROM P., GUMHOLD S., SNOEYINK J.: Large mesh simplification using processing sequences. In *IEEE Visualization conference proceedings* (2003).

[Llo82] LLOYD S. P.: Least squares quantization in pcm. *IEEE Trans. Inform. Theory 28* (Mar. 1982), 129–137.

[LRM*98] LÖTJÖNEN J., REISSMAN P.-J., MAGNIN I. E., NENONEN J., KATILA T.: A triangulation method of an arbitrary point set for biomagnetic problems. *IEEE Transactions on Magnetics 34*, 4 (1998), 2228–2233.

[LSS*98] LEE A. W. F., SWELDENS W., SCHRÖDER P., COWSAR L., DOBKIN D.: Maps: Multiresolution adaptive parameterization of surfaces. *ACM SIGGRAPH Conference Proceedings* (1998), 95–104.

[MTT97] MILLER G. L., TALMOR D., TENG S. H.: Optimal good-aspect-ratio coarsening for unstructured meshes. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms* (1997).

[NT03] NOORUDDIN F., TURK G.: Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics 9*, 2 (April-June 2003), 191–205.

[PC03] PEYRÉ G., COHEN L.: Geodesic remeshing using front propagation. In *IEEE workshop on Variational, Geometric and Level Set Methods in Computer Vision* (2003).

[RB93] ROSSIGNAC J., BORREL P.: Multi-resolution 3d approximations for rendering complex scenes. In *Geometric Modeling in Computer Graphics* (1993), Springer Verlag B. F., Kunii T., (Eds.), pp. 455–465.

[SAG03] SURAZHSKY V., ALLIEZ P., GOTSMAN C.: Isotropic remeshing of surfaces: a local parameterization approach. In *Proceedings of 12th International Meshing Roundtable* (2003).

[SG03] SURAZHSKY V., GOTSMAN C.: Explicit surface remeshing. In *Proceedings of the ACM/Eurographics Symposium on Geometry Processing* (June 2003).

[SSG03] SIFRI O., SHEFFER A., GOTSMAN C.: Geodesic-based surface remeshing. In *International Meshing Roundtable* (2003).

[Tur92] TURK G.: Re-tiling polygonal surfaces. *Computer Graphics 26*, 2 (1992), 55–64.

[VP04] VALETTE S., PROST R.: Wavelet-based multiresolution analysis of irregular surface meshes. *IEEE Transactions on Visualization ans Computer Graphics 10*, 2 (2004), 113–122.

[WK03] WU J., KOBBELT L.: A stream algorithm for the decimation of massive meshes. *Graphics Interface Proceedings* (2003), 185–192.