Generic Remeshing of 3D Triangular Meshes with Metric-Dependent Discrete Voronoi Diagrams

Sébastien Valette[1], Jean-Marc Chassery[2] and Rémy Prost[1], *Member, IEEE*,
[1]CREATIS-LRMN, Lyon, France*
[2]GIPSA-LAB, Grenoble, France

***Abstract*— In this paper, we propose a generic framework for 3D surface remeshing. Based on a metric-driven Discrete Voronoi Diagram construction, our output is an optimized 3D triangular mesh with a user defined vertex budget. Our approach can deal with a wide range of applications, from high quality mesh generation to shape approximation. By using appropriate metric constraints the method generates isotropic or anisotropic elements. Based on point-sampling, our algorithm combines the robustness and theoretical strength of Delaunay criteria with the efficiency of entirely discrete geometry processing . Besides the general described framework, we show experimental results using isotropic, quadric-enhanced isotropic and anisotropic metrics which prove the efficiency of our method on large meshes, at a low computational cost.**

I. Introduction

With the ever increasing range of applications using sampled 3D geometric models, resampling has become a very important feature for inter-operability between those applications. As an example, the accuracy of current 3D scanners has been improved, and they are able to produce very faithful 3D meshes of the scanned model, for the price of a large number of vertices. As a consequence, a resampling step is usually carried out before displaying, storing, or using the mesh in another application. Also, the mesh triangle shape factor can be important when considering finite element simulations. In this paper, we propose an adaptive surface mesh coarsening algorithm, which samples the input surface to a mesh with fewer elements than the original mesh. Extension of this approach also leads to remeshing, when one wants the constructed model to have an arbitrary number of elements. Our approach extends the work of Valette and Chassery [1] to non-uniform and anisotropic discrete Centroidal Voronoi Diagrams. The complexity of our algorithm (in terms of calculations and memory requirements) is low, allowing the processing of large meshes up to several million triangles.

II. Previous Work

Coarsening a mesh consists in resampling the original surface with a lower number of vertices. This field of research has been explored in many ways in recent years. A good review of existing remeshing approaches is given in [2], and coarsening approaches are described more precisely in [3].

In [4] and [1], the triangles of the input mesh are clustered and a new coarsened mesh is built according to the clustering. These approaches are efficient when the number of triangles of the output mesh is much lower than the number of triangles of the input mesh. The approach of Cohen-Steiner et al. [4] aims to create approximation-efficient meshes, whereas the approach of Valette and Chassery [1] aims to create uniform output triangulations. Note that in [5] an extension to the work of Valette and Chassery to adaptive coarsening is proposed. In [6], similar clustering approaches are used to create base domains on polygonal meshes. These base domains are then combined with parametrization techniques to process quadrangular remeshing of the original model. Note that clustering can have several possible applications, aside from remeshing. As an example, in [7] a hierarchical clustering approach is proposed, with a multiresolution radiosity application example. The coarsening of very large meshes (made of millions of vertices) is also an issue when the mesh data structure cannot fit inside the computer memory. As a consequence, out-of-core approaches have been proposed [8]–[11].

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 [12]–[15]. They provide good results, but are limited in practice by the parametrization step, involving heavy calculations and numerical instability. To overcome these problems, methods in [16], [17] were proposed, involving local parametrization and optimization of the remeshed model. Other works [18], [19] distribute new vertices directly on the original surface mesh, to build a new tessellation which can be further optimized.

In [20] and [21] the authors propose to remesh the model using geodesic distances: the new vertices are created using geodesic front propagation, and their distribution can be driven by local curvature.

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

III. Contribution and paper outline

The proposed approach is an extension of the work by Valette and Chassery [1]. It is based on partitioning (clustering) the input mesh in a variational framework, in order to distribute efficiently the vertex budget on the mesh, according

*CREATIS-LRMN, Université de Lyon, INSA, UCB, CNRS UMR 5220, Inserm U630

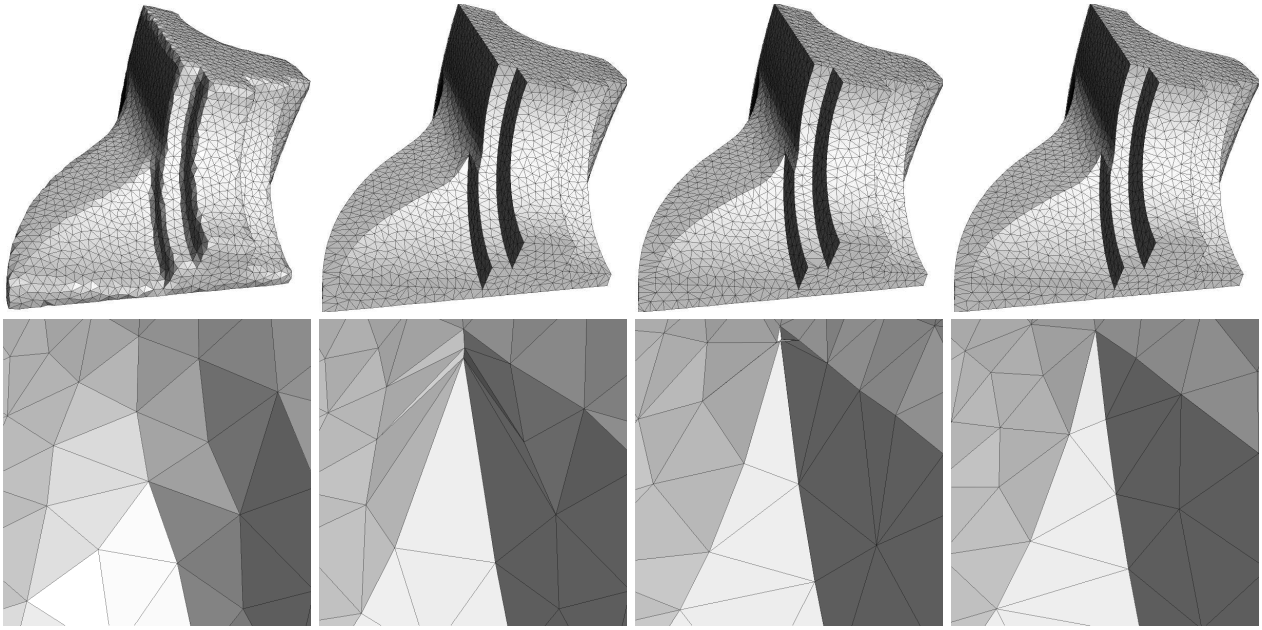


Fig. 1. Remeshing the fandisk to 3k vertices. Top : results. Bottom : closeup view. The previous approach builds a coarsened mesh according to linear criteria (each vertex is the center of mass of its corresponding cluster). The resulting mesh elements have good aspect ratio, but the sharp details of the original model are lost (left). Post-processing relocates the vertices according to approximation criteria. The resulting mesh (middle left) respects faithfully the original model features, but the good aspect ratio is lost for some triangles (see closeup view). A Lloyd-based clustering leads to similar results (middle right). With the proposed approach (right), both properties are preserved by embedding the approximation criteria inside the minimization algorithm.

to user-defined criteria. In this paper, several key-points are addressed :

- The clustering is driven by the minimization of a discrete energy term. The minimization approach is enhanced by generalizing the notion of Voronoi Diagrams, in spirit with Constrained Voronoi Diagram definitions [22]. This generalization allows us to define arbitrary vertex placement strategies which are embedded inside the minimization step, directly constructing accurate meshes without post-processing. As a consequence, the removal of post-processing steps keeps the overall mesh quality from decreasing. As an example, figure 1 shows the results obtained from the Fandisk model by the previous approaches with the proposed one. Clearly, the quality of the resulting mesh (in terms of element aspect ratio) is well preserved. Also, this scheme avoids the need for curve sampling along sharp features, as the created vertices naturally align with the underlying features.
- The clustering is driven by a user-defined metric, allowing the creation of isotropic or anisotropic elements, depending on the desired output. Thus, our approach has uniform sampling capabilities as well as approximation-efficient properties, depending on the chosen metric. Figure 2 shows some results with different metrics on the hand model : The left model was constructed using an isotropic metric and results in elements having good aspect ratio. The model displayed in the middle was created using an anisotropic metric. Unfortunately, the post-processing used to enhance the approximation quality of the mesh induces artifacts, and the resulting mesh is not satisfactory. The model on the right was created using the anisotropic metric with embedded vertex placement strategy, and is made of anisotropic elements with a good approximation quality.
- We also give some details about the minimization algorithm, and enhance it with a safe acceleration scheme which dramatically reduces computing times.

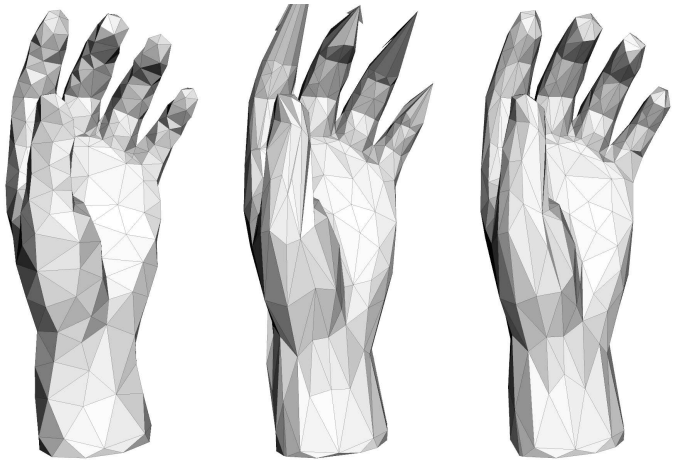


Fig. 2. Coarsening the hand model. Left : Isotropic metric. Center : Anisotropic metric + post-processing. Artifacts are clearly visible. Right : Anisotropic metric with approximation-effective embedded vertices placement scheme.

Section IV and V of this paper give technical overviews of Voronoi Diagrams, both for their continuous and new discrete definitions. In section VI, we explain some implementation details along with theoretical justifications. Section VII shows some experimental results, and a conclusion follows.

IV. Voronoi Diagrams in the continuous setting

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 (or Voronoi Tesselation) can be defined as n distinct cells (or regions) C_i such that:

$$C_i = \{w \in \Omega | d(w, z_i) < d(w, z_j) j = 1, 2, \ldots, n, j \neq i\} \quad (1)$$

where d is a distance measure. These diagrams are well known in the literature [23]. The dual of a Voronoi Diagram (VD) is a Delaunay Triangulation (DT), which has the property that the out-circle of every triangle does not contain any other site when considering the 2D plane.

A Centroidal Voronoi Diagram (CVD) is a Voronoi Diagram where each Voronoi site z_i is also the mass centroid of its Voronoi Region [24]:

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

where $\rho(x)$ is a density function.

Centroidal Voronoi Diagrams minimize the energy given as:

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

Constructing a CVD can be done, using algorithms such as k-means or Lloyd relaxations [25]. The practical efficiency of CVD construction has been demonstrated for a wide range of applications [24].

More generally, the definition of VD stands for non-euclidean settings. Indeed, only a notion of distance and density is needed for such a computation. Recent works have introduced new investigation techniques [26], [27], using Anisotropic Voronoi Diagrams (AVD), involving anisotropic distance measures. Those two approaches are very similar, since they measure distances on the plane with Riemannian metric tensors, which can be represented by 2×2 matrices. The distance between two points p_1 and p_2 on the plane with respect to the tensor K_m can be computed as:

$$d_m(p_1, p_2) = \sqrt{(p_2 - p_1)^T K_m (p_2 - p_1)} \quad (4)$$

This notion is referred to as a directional distance. Labelle and Shewchuk [27] define AVD cells as:

$$C_i = \{w \in \Omega | d_{z_i}(w, z_i) < d_{z_j}(w, z_j) j = 0, 1, \ldots, n-1, j \neq i\} \quad (5)$$

on the other hand, Du and Wang [26] propose :

$$C_i = \{w \in \Omega | d_w(w, z_i) < d_w(w, z_j) j = 0, 1, \ldots, n-1, j \neq i\} \quad (6)$$

Note that the difference between those two definitions is the choice of the tensors for the distance computation: With Labelle and Shewchuk's definition, distances are measured according to the Voronoi Sites z_i. As a consequence, there is no need to define a tensor field for this kind of diagram, only one tensor is needed for each site.

In the second case, distances are computed according to tensors defined on each point w of the space. This requires the definition of a tensor field on the entire domain.

In [26], Du and Wang proved that their definition is more consistent with the classical definition of Voronoi Diagrams and CVD. As an example, if the tensor field is isotropic (but non-uniform), their definition reduces to the classical VD definition. Moreover, defining Riemannian tensors for the Voronoi sites can be problematic for sharp features. As an example, a site placed on the corner of a cube would have an ill-defined metric Tensor whereas it is very possible to define accurate tensors for the points belonging to the flat regions of its cell. More details on the differences between these two definitions are given in [26].

V. Voronoi Diagrams in a discrete setting

In [1] a discrete definition of CVD is given. Ω is no longer a continuous space, but a polygonal mesh M. Subsequently we will only consider triangular meshes, but extension to the polygonal case is straightforward. Partitioning M can be done in two ways : building clusters C_i of triangles, as proposed in [1], or by building clusters of vertices. We found it more practical to cluster the mesh vertices instead of the mesh triangles, mainly for two reasons :

- For a triangular mesh, the number of vertices is about half the number of triangles, and clustering vertices reduces the required memory space for the clustering data.
- Clustering vertices is more rigorous when considering topological changes that may occur during the simplification, and is better suited for non-manifold meshes.

In the following equations, we will refer to items I_j which may be triangles or vertices of the mesh.

A. Isotropic case

The discrete definition of the CVD consists in reformulating the energy term E (equation (3)) and trying to find the clustering minimizing E_{iso}, which is defined by:

$$E_{iso} = \sum_i \left(\sum_{I_j \in C_i} \int_{I_j} \rho(x)\|x - z_i\|^2 dx \right) \quad (7)$$

In this equation, the domain I_j considered in the integral term is :

- the j^{th} triangle when clustering triangles
- the j^{th} vertex dual cell

Figure 3 shows the difference between those two cases.

Note that in contrast to previous definitions of CVD, we will make no assumption on the points z_i, which were identified previously as centers of mass of their respecting clusters. This generalization will be of great help when considering non-planar meshes, where the best location of the points z_i might not be the cluster centroids. As a consequence, we can simply assume that the coordinates of z_i depend on their respective cluster configuration.

It is easy to demonstrate that the individual contribution of each item I_j to the global energy term E_{iso} can be simplified to:

$$\int_{I_j} \rho(x)\|x - z_i\|^2 dx = M_j \|z_i - \gamma_j\|^2 + Aj \quad (8)$$

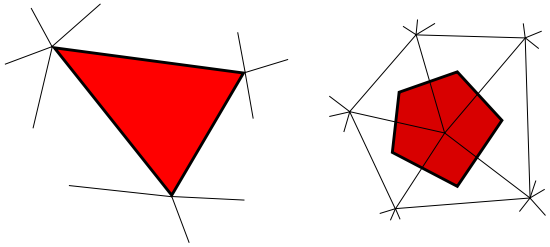


Fig. 3. The domains taken into account when computing integral values. Left: when clustering triangles, the elementary domains are simply the triangles themselves. Right : when clustering vertices, the elementary domains are their vertex respective dual cell.

where

$$A_j = \int_{I_j} \rho(x)\|x-\gamma_i\|^2 dx \quad (9)$$

$$M_j = \int_{I_j} \rho(x) dx \quad (10)$$

$$\gamma_j = \frac{1}{M_j}\int_{I_j} \rho(x) x dx \quad (11)$$

A_j depends only on the geometry of I_j and on the density function $\rho(x)$, M_j is the global weight of I_j according to $\rho(x)$ and γ_j is the center of mass of I_j. By considering each item's individual contribution to E_{iso}, following equation (8), we obtain:

$$E_{iso} = \sum_{i=1}^{n}\left(\sum_{I_j\in C_i} M_j\|z_i-\gamma_j\|^2\right) + \sum_j A_j \quad (12)$$

which simplifies to :

$$E_{iso} = \sum_j A_j + \sum_j M_j\|\gamma_j\|^2 + F_{iso} \quad (13)$$

with

$$F_{iso} = \sum_i L_{iso,i} \quad (14)$$

and

$$L_{iso,i} = \|z_i\|^2 \sum_{I_j\in C_i} M_j - 2z_i^T \sum_{I_j\in C_i} M_j\gamma_j \quad (15)$$

$L_{iso,i}$ is the individual contribution of the cluster C_i to the global energy F_{iso}. Equation 13 proves that whatever the cluster configuration is, the contribution of the terms A_j and $M_j\|\gamma_j\|^2$ will always be the same. We can then safely omit their computation to minimize the energy depicted by F_{iso}

Finally, this energy-term is flat-exact, meaning that its minimization is consistent and equivalent to a Discrete Centroidal Voronoi Diagram (DCVD) on the plane, with no assumption on the input mesh sampling properties (i.e. uniformity or aspect ratio). Note that if one makes the assumption that the Voronoi seeds z_i are the centroids of their respective clusters, equation (14) simplifies to the energy term given in [1], [5].

B. Anisotropic case

In order to extend the discrete CVD described before to anisotropic discrete CVD, we consider the work of Du and Wang [26]. Following their definition of directional distance (equation (4)), and using a similar evaluation of the previous section we define an anisotropy-based energy function as:

$$E_{aniso} = \sum_i \left[\sum_{I_j\in C_i} (\gamma_j - z_i)^T K_j (\gamma_j - z_i)\right] \quad (16)$$

whose minimization leads to an anisotropic partitioning of the initial mesh. Again, simplifications lead to another energy term:

$$F_{aniso} = \sum_i L_{aniso,i} \quad (17)$$

with

$$L_{aniso,i} = z_i^T\left(\sum_{I_j\in C_i} K_j\right) z_i - 2z_i^T\left(\sum_{I_j\in C_i} K_j\gamma_j\right) \quad (18)$$

Note that when the directional distance tensor field K_j is chosen to be isotropic, equation (17) reduces to the isotropic energy term F_{iso} defined in equation (14).

C. Voronoi Center Location

In previous works [1], the Voronoi site locations are defined to be the center of mass of their respective cluster. This placement strategy is not optimal for the case of 3D meshes, since for curved clusters, the barycenter will be *inside* or *outside* the object, with no proof that it is the best position for surface approximation. Indeed, this position can be further optimized to enhance the quality of the approximating mesh.

In [5], the authors propose to relocate the cluster site positions (the output mesh vertices) using Quadric Error Metrics [28]. This post-processing was previously proposed by Lindstrom [8]. The Quadric Error Metric (QEM) associates each triangle. T_i with a 4×4 matrix Q_i which reflects the distance from a given point to the plane tangent to T_i. For a given set of triangles, an 'optimal' vertex position can be computed from the sum of the QEM matrices associated to the triangles. This framework was proved to be very efficient, and has been linked to approximation theory in [29]. Figure 1 (center) shows the effect of such post-processing on the fandisk model.

Actually, this post-processing can be embedded inside the minimization scheme : for each cluster, we store and update its QEM matrix. This allows us to compute at each iteration the best location for a given site, and then inject this location in the computation of the energy term F. As a result, during the clustering, the cluster sites are well placed, and the post-processing is avoided. Figure 1 (right) shows the results obtained on the fandisk model. Note that we use QEM only to evaluate the positions of the clusters centers. This new placement scheme has actually an impact on the energy value E, as shown by equations (15) and (18), but the shape of the optimized clusters will still be driven by the chosen

metrics. As a consequence, when a cluster contains a local feature, the resulting vertex will be well placed on the feature while the energy minimization optimizes the clusters shape independently. Note that if a cluster evolves during the energy minimization, its equivalent vertex can also move, but when the cluster lays on a feature, the vertex will slide along that feature. In figure 18, the top image shows a clustering of the fandisk with 3000 clusters. One can notice the good alignment of the clusters with the features of the mesh. This clustering was obtained without any feature-aware initialization. The middle image shows a clustering with 1500 clusters. Given this low number of clusters, the algorithm cannot represent faithfully the original mesh with uniform sampling. On this example, one cluster spans two corners of the fandisk, and it results in one lost corner in the coarsened mesh (bottom image).

VI. Implementation

In this section, we propose to partition the input mesh according to Delaunay criteria, extending [1]. We will explain several key-points, namely the chosen clustering-meshing approach, the chosen metrics and implementation details.

A. Clustering algorithm

It is possible to efficiently minimize the energy terms F_{iso} or F_{aniso} with an iterative algorithm that updates the clustering according to tests on the boundaries between the different clusters. Assuming that a given edge e (further referred to as a *boundary edge*) is on the boundary between two clusters C_a and C_b (see figure 4), e has two adjacent items I_j and I_k belonging respectively to C_a and C_b, three values of F are computed:

- F_{init} (the initial configuration) : I_j belongs to C_a and I_k belongs to C_b.
- F_1 (C_a grows and C_b shrinks) : both I_j and I_k belong to C_a.
- F_2 (C_a shrinks and C_b grows): both I_j and I_k belong to C_b.

the cluster configuration is updated according to the lowest energy term between F_{init}, F_1 and F_2. By looping in the boundary edge set (the set of edges between two different clusters), we iteratively minimize F. By definition, we know that E is always positive. F differs from E by only an additive constant, and as each local modification reduces F, the convergence of the algorithm is guaranteed. See Algorithm 1 for a pseudo-code equivalent of our algorithm. Figure 4 depicts the existing analogy between vertices clustering and triangle clustering. Note that in this context, F refers to F_{iso} or F_{aniso} depending on the chosen setting. A fast and efficient computation of F is possible by storing the data in accumulator arrays. Moreover, during an elementary test, we actually do not need to really compare the global values of F between the three possibilities. We just need to compare the values $L_a + L_b$, as only clusters C_a and C_b are to be modified.

Figure 5 shows an example of clustering on a randomly triangulated plane. The original plane (left) consists in 4 areas with a different sampling density. The four regions contain

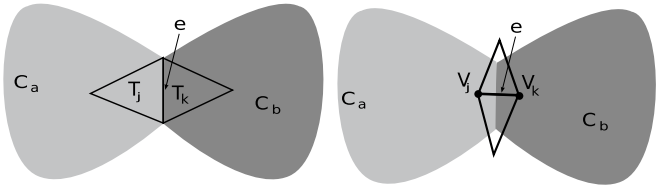


Fig. 4. Local neighborhood used for the clustering evolution. The items I_j can either be triangles T_j (figure on the left) or vertices V_j (figure on the right) depending on the chosen clustering framework. Left (resp. right): The triangles T_j and T_k (resp. vertices V_j and V_k) originally belong to the clusters C_a and C_b, and the test consists in checking whether changing the configuration (moving T_j to C_b or T_k to C_a (resp. V_j to C_b or V_k to C_a) will decrease the global energy term.

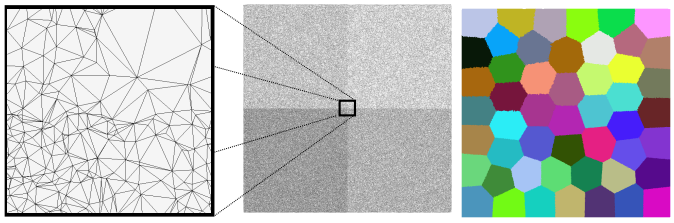


Fig. 5. Center: a triangulated plane (triangular items I_j) falls into 4 parts having different vertex density (close-up view on the left image). Despite the sharp density changes, the clustering (right) remains uniform over the plane

respectively (from top left to bottom right) 10000, 20000, 40000 and 80000 vertices. Notice that despite the sharp density changes in the original sampling, the resulting clustering (right) is uniform, which proves that our approach is sampling independent.

This minimization algorithm has several advantages over Lloyd relaxation :

- We keep track of the boundaries between the clusters using a simple FIFO queue containing all the candidate edges. Thus, the complexity of looping on the boundary elements is linear. On the other hands, the algorithms proposed in [4], [20] use priority queues which slow the clustering down when dealing with large meshes, exhibiting $O(nlog(n))$ complexity instead of a linear one.
- Our minimization algorithm has a guaranteed convergence. Moreover, when the algorithm is close to convergence, only a subset of the boundary edges is actually modified, because some regions already have reached local convergence. Thus, during a loop, we keep track of the clusters which were modified during the previous loop, and we are able to avoid testing a boundary edge for clustering update if its two neighboring clusters have not previously changed. This way, a lot of useless tests are avoided, and the clustering speed is increased. Typically, this scheme reduces the computing time by at least 50% when using a low complexity metric, and more than 80% when using a complex metric such as the QEM enhanced metric.
- The tests on boundary edges involve mostly local topological and geometric operations. Consequently, we have been able to implement this algorithm in a parallel way, which improves the speed of our approach on multicore architectures when using computationally expensive met-

Algorithm 1: pseudo-code for our clustering algorithm.

Data: An initial clustering (each cluster has at least one item I associated)
Result: An optimized clustering
begin
  Fill the queue $Queue1$ with the edges present on clusters boundaries;
  Empty the queue $Queue2$;
  repeat
    $Modifications = 0$;
    while $Queue1$ *not empty* **do**
      Pop a candidate edge e from $Queue1$;
      if *the edge* e *is on a boundary between different clusters AND the edge* e *was not already tested in this loop* **then**
        C_a and C_b are the clusters for which e is a boundary;
        for *the three different cases (see figure 4)* **do**
          Compute z_a, z_b and $L_a + L_b$
        end
        Compare $L_a + L_b$ between the three cases;
        if *the minimal energy does not come from the initial configuration* **then**
          Update the clusters according to the case giving the minimal energy;
          Push the modified item neighbor edges in $Queue2$;
          Increment $Modifications$;
        else
          Push the candidate edge e in $Queue2$;
        end
      end
    end
    Swap the queues $Queue1$ and $Queue2$;
  until $Modifications = 0$;
end

rics.

B. Guaranteed valid clusters

Once the clustering done, each cluster has to be a connected set of vertices. One way to respect this constraint, after the algorithm convergence, is to ”clean” the clusters falling into several connected components, and to restart the clustering step again, as proposed in [1]. These two steps can be repeated until the constraint is respected. Figure 6 shows the effect of the cleaning step on a clustering having a defect. Although this approach works well in practice, there is no theoretical proof that it will always succeed, and running alternatively the clustering step and the cleaning step can be computationally expensive. To overcome this difficulty, we run a three step algorithm. First, we run the clustering algorithm as described by algorithm 1. During this optimization, one does not need the convergence to be achieved, as a second optimization step will be used later. As a consequence, the optimization (energy minimization) stops when the clustering algorithm is near convergence. In our experiments, we defined near-convergence to be achieved when the number of modified items during a loop on the candidate edges is smaller than 0.1%. Afterwards, we run the cleaning step. If some cleaning was done (meaning that some clusters did not respect the connexity constraint), we then re-apply the clustering step, with an additional embedded checking step. Figure 4 displays a local boundary context used during clustering evolution. Each time a vertex V_j has to move from one cluster C_a to another cluster C_b, we check if this modification does not break the connectedness property of the cluster C_a, which can be easily done by checking the vertex 1-ring configuration.

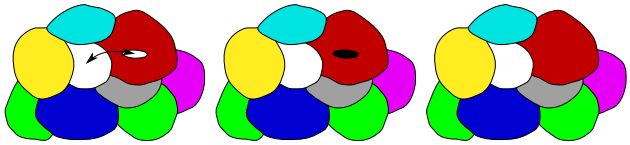


Fig. 6. Clustering cleaning : the clustering (left) has a defect : the white clusters falls into two connex components. A cleaning step resets the smallest component to *not associated*, in black color (center). After few iterations of the clustering algorithm, the disconnected component has disappeared(right)

With this constraint, after the second clustering step, all the clusters are guaranteed to have only one connecting component. Note that we do not take this constraint into account during the first minimization process, as it would significantly decrease the speed of the algorithm, and it could prevent the removal of the input mesh topological noise.

C. Meshing

As explained in section II, many works have already proposed a clustering-based simplification of the input mesh. Variational approaches such as those proposed in [4] and [1] are the most promising, since they are based on global optimization of the clustering. Both aggregate the mesh triangles into clusters, but the meshing strategies are dual.

Basically, Cohen-Steiner et al. [4] construct one polygon for each created cluster, and the polygon vertex positions are computed according to the cluster adjacency relationships. As a consequence, the produced clusters must have as much as possible a planar shape. Note that the polygons can also be further modified depending on the type of desired output mesh e.g. quad-dominant or pure triangular, but it would be very hard to restrict the properties of the resulting polygons to a specific type.

On the other hand, Valette and Chassery [1] create one vertex for each cluster. Meshing is done by creating triangles by dualizing the clustering i.e. two vertices are adjacent if their corresponding clusters are adjacent too. The resulting mesh contains only triangles. In this case, the clusters do not need to satisfy the planarity criterion. Moreover, this approach provides a direct control on the vertex positions, which can be vital when considering approximation quality.

D. Dealing with mesh boundaries

Note that meshes with boundaries need a supplementary meshing step to adjust the coarsened model, by adding extra

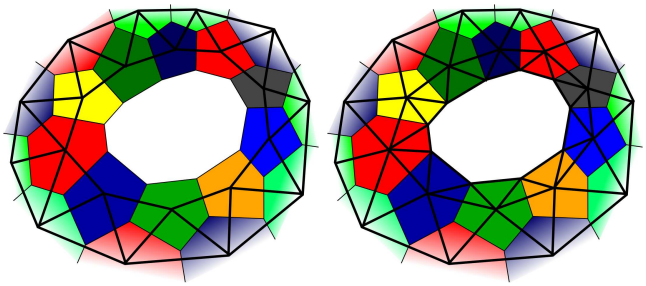


Fig. 7. A part of a mesh contains a boundary, resulting in a hole in the clustering. The resulting triangulation (left) has also one boundary, but it is wider than the original one. Adding a triangle strip (right) creates a boundary closer to the original one

vertices and triangles on the boundaries. Basically, each time two clusters meet at one boundary, one vertex and two triangles are added. This results in the construction of a triangle strip for each boundary. Figure 7 depicts how this procedure fixes the new mesh boundaries.

E. Extension to 3D surfaces and challenges

The previous definition of DCVD stands for planar configuration, but is still very reliable when considering 3D surfaces. Indeed, the equations only involve measures of distance and weights. A strict equivalent of DCVD for 3D surfaces would involve the computation of geodesic distances (which would be computationally prohibitive), but when considering highly sampled meshes, the error induced by using euclidean distance instead of geodesic distance remains low. Moreover, if we compare such a discrete approach with parametrization-based Delaunay algorithms [12], [13], parametrization also introduces distortion in the remeshing process. Those works compensate the parametrization distortion by introducing scaling factors based on the ratio between distances on the parametrized plane and euclidean (but not geodesic) distances on the mesh. As a consequence, those approaches seem to have at least the same shortcomings as regards geometric accuracy. Also, computing geodesic distances would probably increase the influence of the geometric noise present in the input mesh.

F. Remeshing by over-sampling

Cluster-based approaches have a restriction : the resulting mesh will have fewer vertices than the original one. However, we are able to construct meshes with as many vertices as the original ones, by simply subdividing the input mesh using linear, Loop or Butterfly schemes. Figure 8 shows a remeshed version of the Stanford Bunny with 36k vertices. The input mesh (36k Vertices) was subdivided twice to obtain a mesh with 1111k triangles, well suited for a clustering approach.

G. Efficient initial sampling

To begin the clustering process, an initial sampling step must be done, to associate at least one item I_j to each cluster C_i. In [1], the initial sampling is done by randomly selecting one vertex of the mesh for each cluster. As a consequence, the clusters will be equally distributed over the

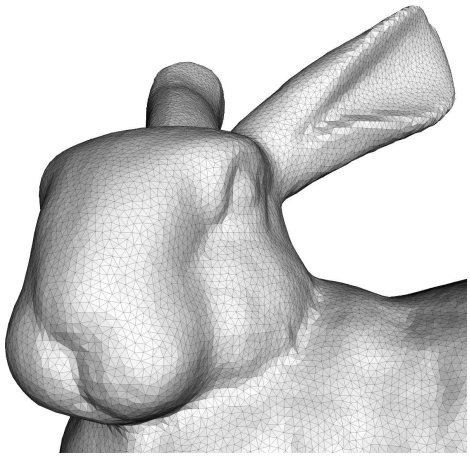


Fig. 8. Uniform remeshing of the Stanford Bunny to 36k vertices

original mesh. This is convenient for uniform coarsening, as the goal is to build clusters with the same area. But this is not appropriate for adaptive clustering, since the regions with higher density should contain more clusters than regions with low density. Indeed, if we randomly distribute the clusters during the energy minimization process, the clusters in low density regions will slowly move towards regions with high density, resulting in very low convergence speed. To alleviate this problem, we propose to distribute the clusters according to the density function. For this aim, we first compute a global average cluster density:

$$D = \frac{1}{n}\sum_{j=1}^{n}\rho_j \tag{19}$$

where n is the number of desired clusters. This density corresponds to the average accumulated density that each cluster should have at the end of the clustering process. We try to initialize the clustering with clusters having such an accumulated density. For each cluster, we randomly pick a free vertex V_f (a vertex which was not previously associated to any cluster) and grow a region around V_f until its accumulated density reaches D. If at one point some clusters remain to be initialized and no more vertices are free (which can happen, as we operate on a discrete set), we randomly pick one non-free vertex for each non initialized cluster. In practice, this initial sampling strategy accelerates the convergence of the approach, and allocates more clusters in regions needing a higher sampling rate.

H. Metrics

First, when considering isotropic settings, the clustering can be optimized by maximizing the compactness of the cells (equation (3)), which requires the definition of a density function ρ on the mesh. Choosing a uniform density leads to uniform clustering [1]. Adaptive clustering is also possible by defining a density map according to some curvature measures [5]. Adaptivity is a key feature for many applications, when some parts of the mesh must contain more vertices than other parts. As we aim at applying our scheme to very

complex meshes, the curvature measure has to be very robust against bad sampling conditions that may be encountered when processing such models. We propose to compute a curvature indicator with such properties. We calculate the matrix $A_{2\times 2}$ of the Weingarten map of the surface using a polynomial fitting of the local neighborhood of each vertex, as explained in [30]. The local principal curvatures $k_{j,1}$ and $k_{j,2}$ (resp. the principal directions $D_{j,1}$ and $D_{j,2}$) are the eigenvalues of A (resp. the eigenvectors). In all our experiments, we chose the neighborhood of a vertex to be its 3-ring. Finally, we set each vertex weight ρ_j to:

$$\rho_j = |P_j| \left(\sqrt{k_{j,1}^2 + k_{j,2}^2} \right)^{\gamma} \tag{20}$$

where $|P_j|$ is one third of the area of the triangles around P_j and γ is a gradation parameter which controls the curvature adapted behavior of our scheme. Considering [13], setting $\gamma = 0$ will produce uniform clustering whereas higher values of γ will give more and more importance to the regions with high curvatures. Subsequently, we will refer to this metric as the I Metric.

To offer our algorithm anisotropic behavior, following the energy term defined by equation (17), we need to define directional distance tensors for each vertex of the input mesh. Again, local curvature computation can lead to the definition of a directional 3×3 distance tensor for each vertex K_j defined as:

$$K_j = M_j^T M_j \tag{21}$$

with

$$M_j = \begin{pmatrix} \sqrt{\|k_{j,1}\|} D_{j,1}^T \\ \sqrt{\|k_{j,2}\|} D_{j,2}^T \\ 0 \end{pmatrix} \tag{22}$$

this metric tensor ensures that regions with constant principal directions and curvatures will produce clusters with an elongation ratio equal to $\sqrt{\left|\frac{k_{j,1}}{k_{j,1}}\right|}$, which is consistent with approximation theory [31].

VII. Results and discussion

Figure 9 compares the clustering efficiency of our approach with Lloyd relaxation for two cases : using a linear isotropic metric, and a quadric enhanced isotropic metric, both applied on the statuette model. The horizontal axis is the time, while the vertical axis gives the energy value F_{iso}, which differs from E_{iso} only by a constant value. For both cases (and all our experiments), our approach led to values of F_{iso} lower than what Lloyd relaxation gave. The relative difference between the two algorithms was around 10^{-6} for the statuette model. While this improvement is not significant in terms of energy value, our algorithm has other advantages, in terms of acceleration and convergence. Our approach always reaches convergence whereas Lloyd relaxations failed to produce a stable clustering for the statuette model. More generally, we sometimes observed convergence with Lloyd relaxation when using the simple isotropic metric, but at least an order of magnitude slower than with our approach, and never when combining quadric-based placement and Lloyd iterations. Finally,

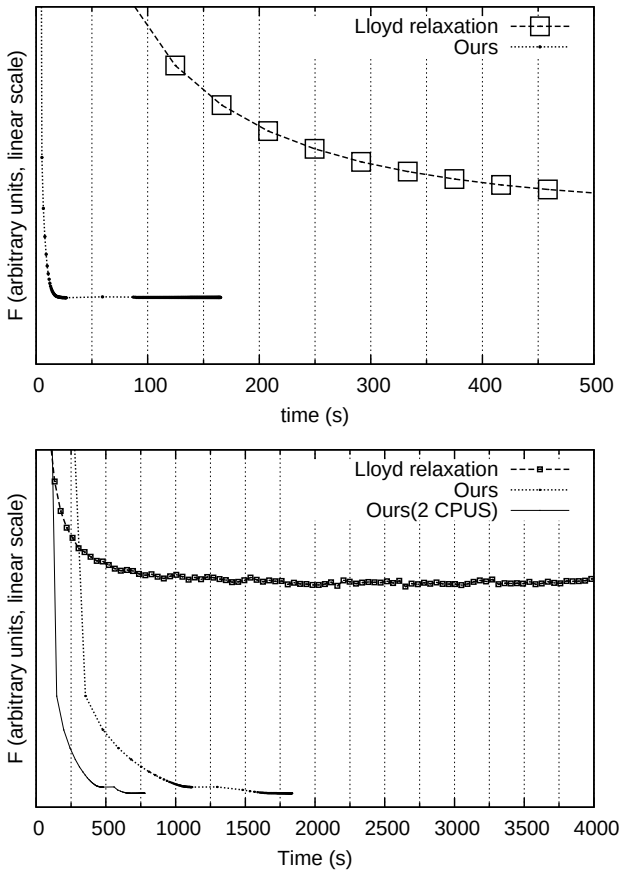


Fig. 9. Comparison of efficiency between Lloyd relaxation clustering and Our proposal on the statuette model. Top : isotropic metric (I). Bottom : isotropic metric with quadric-based vertices placement (IQ).

we do not need to manually stop our minimization algorithm (by defining a fixed number of iterations, or by measuring the energy decrease rate), which could be problematic when processing large meshes. Indeed, it is observed that in the last minimization steps, only a small subset of the clustering is evolving. An arbitrary decision to stop the minimization could penalize the clustering quality in these regions.

We compared the speed between Lloyd relaxation and our approach. One single Lloyd relaxation step lasts 45s in average. With the isotropic metric, our approach converges within 165s, which is less than the time needed to perform 5 Lloyd iterations. As figure 9 shows, processing 5 Lloyd relaxation steps is far from convergence. When using quadrics-based placement, the difference is smaller, but still our approach reaches convergence, in contrast with Lloyd relaxations. The curves for the quadric-based placement metric also reveal the effect of the connexity constraint embedded in our algorithm. One can clearly observe that convergence is reached twice. As explained in section VI-B, the first convergence (or near convergence) is reached without connexity constraint. Afterwards, the cleaning step and the constrained clustering proceed, and the energy term gets even lower. To explain this, one can notice that CVD clustering tends to create clusters which are as compact as possible, and that compactness and connexity are actually not contradictory properties. As a result,

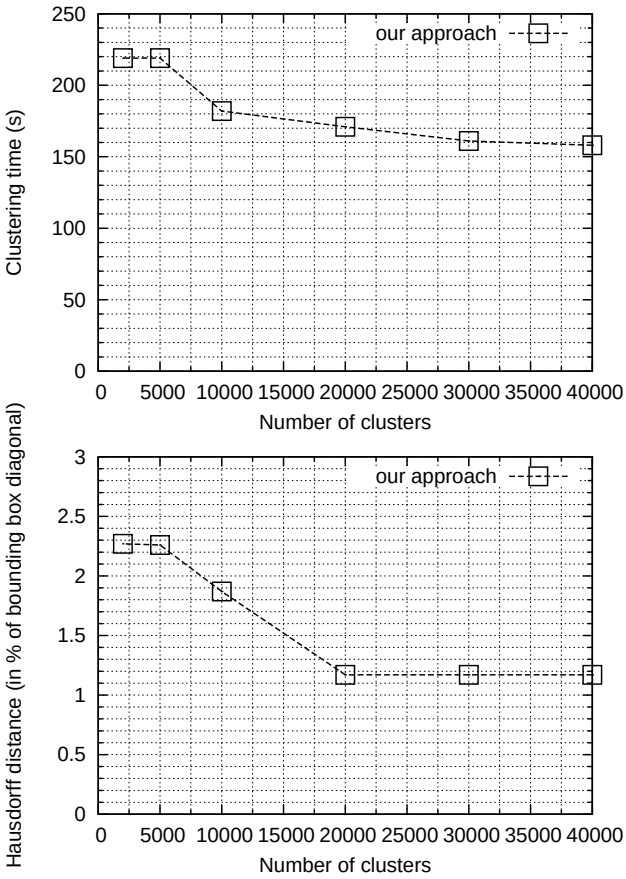


Fig. 10. Clustering with the IQ metric for the David model. Top : clustering time vs. number of wanted clusters. Bottom : Hausdorff distance between the original and coarsened model vs. number of wanted clusters.

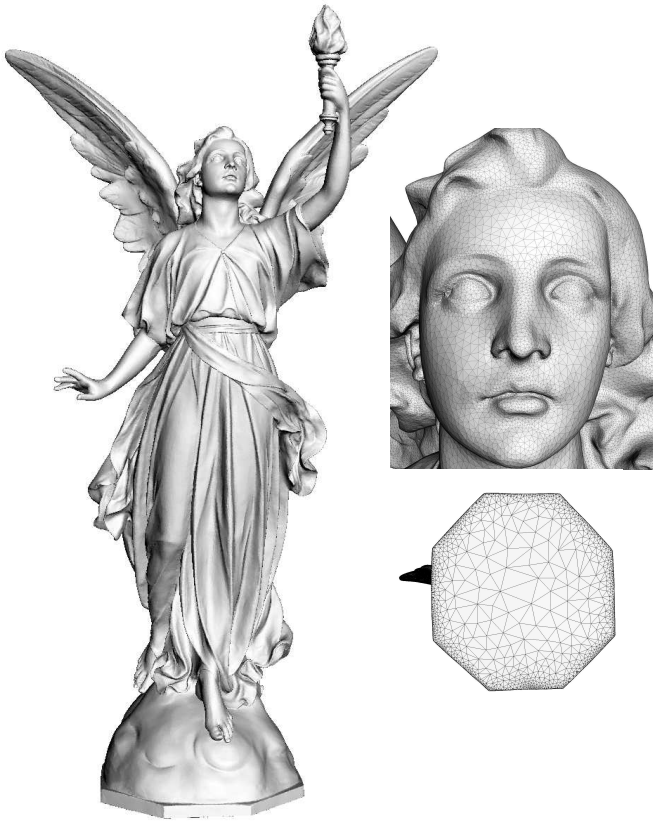


Fig. 11. Left: A 500k vertices coarsened version of the Lucy model. Right: closeup views of the face and pedestal with displayed edges : adaptivity is noticeable in relatively flat regions

the connexity constraint helps our algorithm to reach lower energy values. Our implementation can take advantage of multicore workstations. Tested on a dual Xeon processors workstation, clustering using quadric-based placement takes less than half of the time needed to do the same task with only one processor on the workstation (the speed ratio is superior to 2 because those processors have hyperthreading capabilities). Future improvements could introduce other clustering optimization schemes, to reach even lower minima value for the energy function F_{iso} or F_{aniso}. As an example, Cohen-Steiner et al [4] proposed the *tunneling* of clusters from over-sampled regions to under-sampled ones. Figure 10 displays the clustering time and approximation error vs the number of wanted clusters, for the David model, with the isotropic metric enhanced by quadric-based placement.

Table I shows the timings and quality measures for some results displayed in this paper. The results were obtained with a desktop computer running at 3GHz, with 2GB of RAM, except for the Lucy model and the Michelangelo David, which were processed on a SGI workstation due to memory requirements (The Lucy model itself fits in more than 2GB with our data structure, and the David model, originally made of 507k vertices was subdivided twice, in order to remesh it to 500k vertices). The first two columns are the number of vertices of the input and output meshes. The third column gives the metric used for the clustering (respectively I for isotropic with Voronoi Centers taken as cluster centroids, IQ for isotropic with Voronoi Centers optimized with QEM, AQ for anisotropic metric with Voronoi Centers optimized with QEM). The parameter between parenthesis is the gradation parameter γ defined in section VI-H. Note that for the anisotropic metrics, this parameter is only used for the sampling initialization. The next two columns show the time spent on the curvature measure computation and on the clustering. These last two steps dominate the processing time. Note that we experimentally measured the SGI workstation to be half as fast as the used desktop computer. The last two columns show for each model the percentage of minimal internal angles bellow 30^o and the average triangle aspect ratio, as defined in [33].

Figure 11 shows a coarsened version of Lucy to 500k vertices, using the isotropic metric. Note that here, the sampling is well adapted, as shown by the closeup views. Figure 12 shows the rockerarm coarsened to 1000 vertices (AQ metric) and the Buddha model coarsened to 40k vertices (AQ metric). The comparison of coarsened versions of the hand models (IQ, A and AQ metric, figures 2) gives a representative overview of what we observed in our experiments : the anisotropic metric does elongate the triangles along the directions of minimal curvatures, but choosing the clusters centers as centroids (A metric) produces artifacts at the meshes extremities. Introducing QEM based centers localization (AQ metric) solves this problem. Figure 14 shows a closeup view of coarsened Buddha

Model	#v (original)	#v2 (coarsened)	Metric	curvature time (s)	clustering time (s)	$\angle < 30^o$ (%)	Q_{av}
Lucy	14M	500k	IQ(1.5)	213 (12 CPUS)	8357	3.73	0.77
		500k	IQ(1.5)		2822 (4CPUS)	3.73	0.77
David	507k	500k	IQ (1.5)	76	6365	6.9	0.73
Statuette	5M	300k	I (1.5)	319	165	10	0.69
		300k	IQ (1.5)	319	1665	8.4	0.71
		300k	AQ (1.5)	328	1826	16	0.62
Buddha	500k	20k	IQ (1.5)	47	255	7.5	0.72
		20k	AQ (1.5)	49	295	17	0.61

TABLE I

PROCESSING TIMES AND QUALITY MEASURES FOR THE PROCESSED MESHES. THE COLUMNS ARE RESPECTIVELY THE NUMBER OF VERTICES OF THE INPUT AND OUTPUT MESHES, THE METRIC USED FOR THE CLUSTERING, THE TIME SPENT ON THE CURVATURE MEASURE COMPUTATION AND ON THE CLUSTERING, THE PERCENTAGE OF MINIMAL INTERNAL ANGLES BELLOW 30^o AND THE AVERAGE TRIANGLE ASPECT RATIO.



Fig. 12. Coarsened versions of the rockerarm model (1000 vertices) and the buddha model (20k vertices).

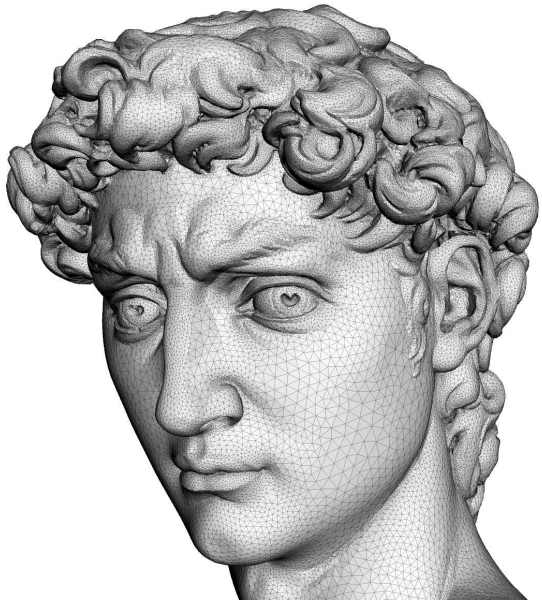


Fig. 13. Closeup view of the David model remeshed to 500k vertices (Isotropic metric)

models (left : AQ metric; right: IQ metric). The anisotropic behavior of the AQ metric is clearly visible in elongated regions (the cloth around the Buddha's neck), and sampling remains isotropic in spherical regions (e.g. on the head). Note that the sharp features located on the back of the model are better preserved with the AQ metric.

On figure 13, we can see a closeup view of the Michelangelo David remeshed to 500k vertices, illustrating that the limitation of our approach in a remeshing point of view is only its memory footprint.

Figure 15 shows a remeshed version of the Statuette model to 500k vertices, using the IQ metric. the right side compares the results between the IQ (top) and AQ (bottom) metrics. Again, the anisotropic metric gives more pleasant results. As the results table shows, the IQ metric is about 10 times slower than the I metric. This is due to the QEM based center localization, which requires for each iteration a 3×3 singular value decomposition in order to have a robust placement. Anisotropic clustering exhibits a reasonable overhead compared to isotropic clustering (below 20%).

Figure 17 and table III compare the mesh quality between our approach and [16]. One one hand, our approach provides a triangulation with less quality than [16]. On the other hand, table II shows that our approach provides a model which is far more faithful to the original model, with a Hausdorff distance about 4 times smaller than with [16]. Note that this table also shows the average and RMS errors between the original and coarsened models (in both directions, as these distances measures are not symmetric), obtained with Metro [32].

Figure 16 shows the hand model coarsened to 300 vertices, using qslim [28], our approach and VSA [4]. Clearly, our results are close to the ones of qslim, VSA efficiently capturing the anisotropy of the model, but failing to represent it with the same precision. We tried our algorithm with two different random initializations. In table II, we can see that in terms

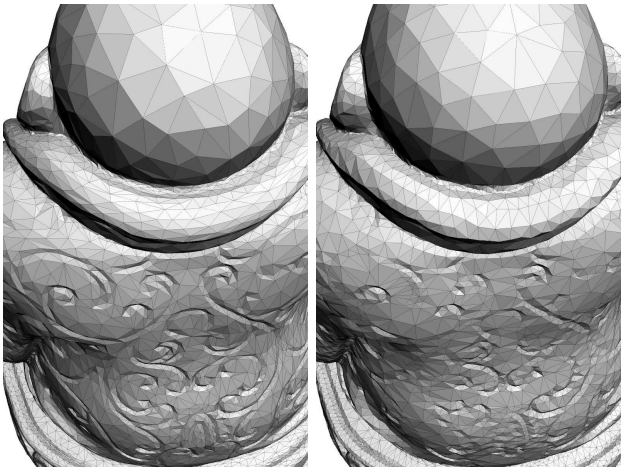


Fig. 14. Closeup view of the Buddha model. Left : anisotropic metric (AQ). Right : isotropic metric (IQ).

Model	Hausdorff d. $\times 10^{-3}$	Mean distance $\times 10^{-3}$		RMS distance $\times 10^{-3}$	
David [SAG03]	27.1	1.6	1.6	1.9	1.9
David [ours]	6.22	0.06	0.1	0.05	0.1
Hand [GH97]	16.9	2.3	2.3	3.0	3.0
Hand [CAD04]	34.6	7.2	7.3	9.3	9.4
Hand [ours]	37.6	3.9	3.9	5.2	5.5
Hand *[ours]	32.9	3.8	3.8	5.1	5.2

TABLE II

COMPARISON OF APPROXIMATION QUALITIES BETWEEN SEVERAL APPROACHES. THE FIRST COLUMN IS THE HAUSDORFF DISTANCE BETWEEN THE ORIGINAL AND COARSENED MODELS. NEXT ARE MEAN AND AVERAGE DISTANCES FOR BOTH DIRECTIONS. THE LAST LINE WAS OBTAINED WITH A DIFFERENT INITIAL CONDITION.



Fig. 15. Comparison between 2 approaches for isotropic coarsening. Left: the statuette model coarsened to 500k vertices (AQ metric). Right : comparison between IQ (top) and AQ (bottom) metrics.

Model	Q_{min}	Q_{av}	$\angle_{min}$	$\angle_{min,av}$	$\angle < 30^o$
David [SAG03]	0.027	0.91	0.92	52.9	0.41
David [ours]	0.013	0.80	0.85	45.2	1.2

TABLE III

COMPARISON OF TRIANGULATION QUALITIES

of Hausdorff distance, our approach gives results similar to VSA, while in terms of average or RMS distance, our approach provides a significant improvement over VSA. The last line of table II shows the results obtained with our approach, but with an different random initialization, showing the robustness of our approach.

VIII. CONCLUSION

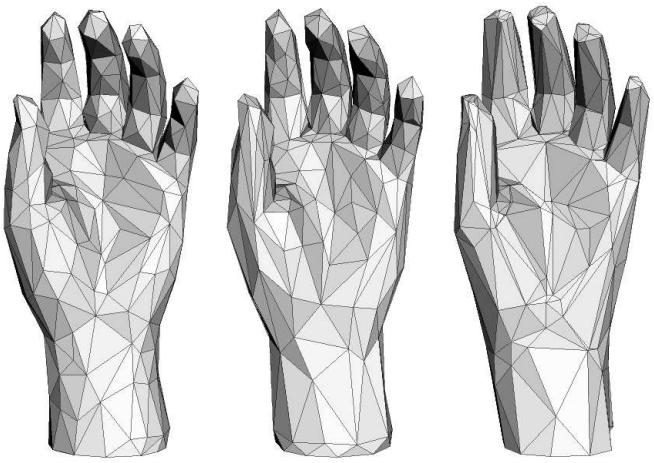
We proposed a generalization to anisotropic remeshing of the isotropic approach proposed in [1]. Based on discrete Delaunay criteria, this algorithm is able to process large meshes and to create meshes made of isotropic and/or anisotropic elements. The proposed framework is general, and the metric definition, which drives the elements aspect ratio, could be improved in further works. We plan to define new metrics based on Local Feature Size, to enhance the approximation quality, and to use filtering to enhance noise removal. Also, giving out-of-core features to this approach could be of great help when considering large models.

ACKNOWLEDGMENTS

The authors thank Christophe Perra and Fabrice Bellet for their support with the implementation. The models displayed in this paper are courtesy of the Stanford 3D scanning Repository, the Aim@Shape Shape Repository and the Digital Michelangelo Project. This work was supported in part by the Région Rhône Alpes Cluster 2 ISLE, PP3, subproject I3M: Imagerie Médicale et Modélisation Multiéchelles : du petit animal à l 'Homme. This work is within the scope of the scientific topics of the PRC-GDR ISIS research group of the French National Center for Scientific Research (CNRS). We also thank Pierre Alliez for his comments on the paper and for providing comparison models. Finally, we thank the reviewers for their constructive remarks, which helped us a lot improving the quality of the paper.

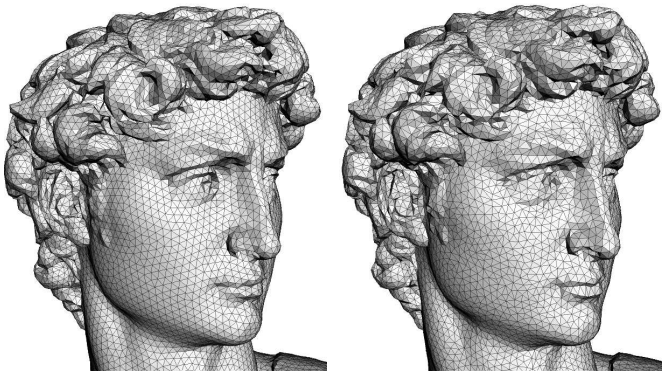
REFERENCES

[1] S. Valette and J.-M. Chassery, "Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening," *Computer Graphics Forum (Eurographics 2004 proceedings)*, vol. 23, no. 3, pp. 381–389, 2004.



[GH 97] [Ours] [CAD 03]

Fig. 16. Coarsened versions of the hand model with 300 vertices. Using (from left to right) qslim [28], our approach and VSA [4]



[SAG 03] [Ours]

Fig. 17. the Michelangelo David coarsened to 100k vertices. Left : with [16]. Right : proposed approach (IQ metric).

[2] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene, "Recent advances in remeshing of surfaces," *Part of the state-of-the-art report of the AIM@SHAPE EU network*, 2005.

[3] D. P. Luebke, "A developer's survey of polygonal simplification algorithms," *IEEE Computer Graphics & Applications*, vol. 21, no. 3, pp. 24–35, May/June 2001.

[4] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variationnal Shape Approximation," *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, 2004.

[5] 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*, 2005, pp. 655–662.

[6] I. Boier-Martin, H. Rushmeier, and J. Jin, "Parameterization of triangle meshes over quadrilateral domains," in *Proceedings of the Symposium on Geometry Processing*, 2004.

[7] M. Garland, A. Willmott, and P. S. Heckbert, "Hierarchical face clustering on polygonal surfaces," in *Proceedings of the Symposium on Interactive 3D Graphics*, 2001.

[8] P. Lindstrom, "Out-of-core simplification of large polygonal models," in *Siggraph 20000, Computer Graphics Proceedings*, K. Akeley, Ed. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000, pp. 259–262.

[9] S. Schaefer and J. Warren, "Adaptive vertex clustering using octrees," in *Proceedings of SIAM Geometric Design and Computing*, 2003.

[10] M. Isenburg, P. Lindstrom, S. Gumhold, and J. Snoeyink, "Large mesh simplification using processing sequences," in *IEEE Visualization conference proceedings*, 2003.

[11] J. Wu and L. Kobbelt, "A stream algorithm for the decimation of massive meshes," *Graphics Interface Proceedings*, pp. 185–192, 2003.

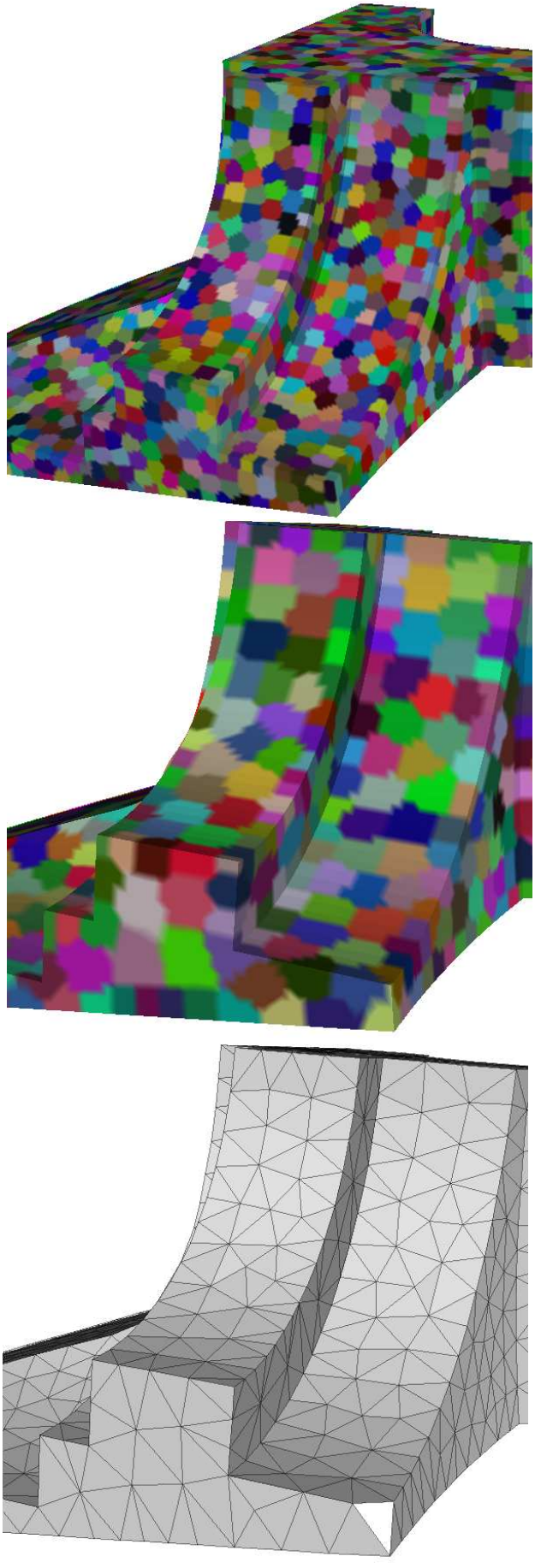
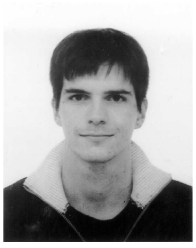


Fig. 18. Clustering results on the fandisk. Top : 3000 clusters. Middle : 1500 clusters. Here, one cluster spans two corners. Bottom : the resulting coarsened mesh misses one corner.

[12] P. Alliez, M. Meyer, and M. Desbrun, "Interactive Geometry Remeshing," *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, vol. 21(3), pp. 347–354, 2002.
[13] P. Alliez, É. C. de Verdière, O. Devillers, and M. Isenburg, "Isotropic surface remeshing," in *Proceedings of Shape Modeling International*, 2003, pp. 49–58.
[14] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Levy, and M. Desbrun, "Anisotropic polygonal remeshing," *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, pp. 485–493, 2003.
[15] X. Gu, S. Gortler, and H. Hoppe, "Geometry images," *ACM SIGGRAPH Conference Proceedings*, pp. 355–361, 2002.
[16] V. Surazhsky, P. Alliez, and C. Gotsman, "Isotropic remeshing of surfaces: a local parameterization approach," in *Proceedings of 12th International Meshing Roundtable*, 2003.
[17] V. Surazhsky and C. Gotsman, "Explicit surface remeshing," in *Proceedings of the ACM/Eurographics Symposium on Geometry Processing*, June 2003.
[18] J. Lötjönen, P.-J. Reissman, I. E. Magnin, J. Nenonen, and T. Katila, "A triangulation method of an arbitrary point set for biomagnetic problems," *IEEE Transactions on Magnetics*, vol. 34, no. 4, pp. 2228–2233, 1998.
[19] G. Turk, "Re-tiling polygonal surfaces," *Computer Graphics*, vol. 26, no. 2, pp. 55–64, 1992.
[20] G. Peyré and L. Cohen, "Geodesic remeshing using front propagation," in *IEEE workshop on Variational, Geometric and Level Set Methods in Computer Vision*, 2003.
[21] O. Sifri, A. Sheffer, and C. Gotsman, "Geodesic-based surface remeshing," in *International Meshing Roundtable*, 2003.
[22] Q. Du, M. D. Gunzburger, and L. Ju, "Constrained centroidal voronoi tessellations for surfaces," *SIAM Journal on Scientific Computing*, vol. 24, no. 5, pp. 1488–1506, 2003.
[23] F. Aurenhammer, "Voronoi diagrams - a survey of a fundamental geometric data structure," *ACM Computing Surveys*, vol. 23, no. 3, pp. 345–405, 1991.
[24] Q. Du, V. Faber, and M. Gunzburger, "Centroidal voronoi tesselations: applications and algorithms," *SIAM Review*, no. 41(4), 1999.
[25] S. P. Lloyd, "Least squares quantization in pcm," *IEEE Trans. Inform. Theory*, vol. 28, pp. 129–137, Mar. 1982.
[26] Q. Du and D. Wang, "Anisotropic centroidal voronoi tessellations and their applications," *SIAM J. Sci Comp.*, vol. 26, pp. 737–761, 2005.
[27] F. Labelle and J. R. Shewchuk, "Anisotropic voronoi diagrams and guaranteed-quality anisotropic mesh generation," in *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, 2003, pp. 191–200.
[28] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," *Computer Graphics*, vol. 31, no. Annual Conference Series, pp. 209–216, 1997.
[29] P. S. Heckbert and M. Garland, "Optimal triangulation and quadric-based surface simplification," *Journal of Computational Geometry: Theory and Applications*, vol. 14, no. 1–3, pp. 49–65, November 1999.
[30] F. Cazals and M. Pouget, "Estimating differential quantities using polynomial fitting of osculating jets," *Computer Aided Geometric Design*, vol. 22(2), pp. 121–146, 2005.
[31] R. B. Simpson, "Anisotropic mesh transformations and optimal error control," *Applied Numerical Mathematics*, vol. 14, no. 1–3, pp. 183–198, 1994.
[32] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: Measuring error on simplified surfaces," *Computer Graphics Forum*, vol. 17, no. 2, pp. 167–174, 1998.
[33] P. Frey and H. Borouchaki, "Surface mesh evaluation," in *6th International Meshing Roundtable*, 1997, pp. 363–374.



Sébastien VALETTE was born in France, in 1975. He recieved the M.S. Degree from the Electrical Engineering Department, at the National Institute for Applied Sciences (INSA) of Lyon, France, in 1998. He obtained the PhD Degree at INSA of Lyon in 2002. He is currently a CNRS researcher in the 'Volume Image Processing' project of the CREATIS-LRMN Laboratory. His research interests include 3D processing, wavelets, progressive compression and multiresolution analysis.



Dr Jean-Marc CHASSERY has position of Director of Research at CNRS. He is responsible of the GIPSA-lab unit (Grenoble Image sPeech Signal Automatic) englobing about 300 members, including about 100 PhD. Dr Jean-Marc CHASSERY develops activities around digital geometry for image analysis and also watermarking approaches oriented to augmented-content, security and steganalysis for images and videos.



Rémy PROST (M82) received his doctorate degree in Electronics Engineering and his Docteur és Sciences degree from Lyon University and the National Institute of Applied Sciences (INSA), Lyon, France, in 1977 and 1987 respectively. He is currently a professor in the Department of Electrical Engineering at INSA-Lyon. Both his teaching and research interests include digital signal processing, inverse problems, image data compression, multiresolution algorithms, wavelets, shape modelling and three-dimensional mesh processing. He leads the Volume (3D) Image Processing project in the CREATIS-LRMN Laboratory (CNRS 5220, INSERM U630, INSA-Lyon, Claude Bernard Lyon I University of Lyon, France).