Surface segmentation for improved remeshing

John Edwards[1], Wenping Wang[2], and Chandrajit Bajaj[1]

[1] Department of Computer Science
The University of Texas, Austin, TX, USA
john.edwards@utexas.edu, bajaj@cs.utexas.edu
[2] Department of Computer Science
The University of Hong Kong, Hong Kong, China
wenping@cs.hku.hk

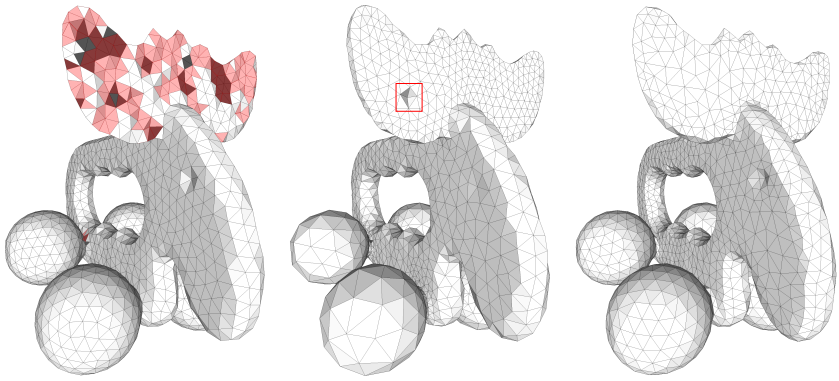
Summary. Many remeshing techniques sample the input surface in a meaningful way and then triangulate the samples to produce an output triangulated mesh. One class of methods samples in a parametrization of the surface. Another class samples directly on the surface. These latter methods must have sufficient density of samples to achieve outputs that are homeomorphic to the input. In many datasets samples must be very dense even in some nearly planar regions due to small local feature size. We present an isotropic remeshing algorithm called κCVT that achieves topological correctness while sampling sparsely in all flat regions, regardless of local feature size. This is accomplished by segmenting the surface, remeshing the segmented subsurfaces individually and then stitching them back together. We show that κCVT produces quality meshes using fewer triangles than other methods. The output quality meshes are both homeomorphic and geometrically close to the input surface.

1 Introduction

Surface remeshing is the process of transforming an input surface mesh S into an output surface mesh W. Often a given model needs to be remeshed to meet the needs of a given application. For example, the number of triangles may be too large for a graphics application to render efficiently, or low triangle quality may cause numerical instability in a FEM simulation. Triangle quality improvement and reducing the number of triangles (decimation) are two important, but often competing, goals in remeshing.

Many remeshing methods sample and optimize points directly on S and then use a triangulation of these points to produce W. These algorithms run into trouble when the number of sample points is very low. Not only does the geometric approximation suffer, but topological errors can be introduced, in the sense that W is not homeomorphic to S, and also that W may not be 2-manifold.

In areas where the surface has high curvature this phenomenon makes intuitive sense. But what can be at first surprising is that in many cases areas of very low



(a) (b) (c)

Fig. 1. Remeshing the Toy Elk model using 2000 sample points. Non-manifold edges and vertices are highlighted in red. (a) Uniform CVT. A shortage of triangles in the horns results in topological errors. (b) *lfs* CVT. Despite a large number of triangles in the horn area there is still one non-manifold vertex. (c) Our method, κCVT. Our method produces improved meshes by distributing samples according to curvature rather than local feature size while avoiding topological errors.

curvature require an inordinately large number of triangles to approximate faithfully. This occurs when other sections of the surface, even if distant geodesically, come in close proximity to the flat region. The fact that topological and geometric errors increase with fewer triangles in such areas is an unfortunate side-effect of using the euclidean metric to approximate geodesic distances. One way to look at this is in terms of the local feature size (*lfs*), which is defined as the distance from a point $p \in S$ to the closest point on the medial axis of S. The r-sampling theorem, discussed further in section 2, states that the number of samples needed to ensure that W is homeomorphic to S is dependent on *lfs*. Thus, even if an area is nearly planar, large numbers of triangles will be needed if the *lfs* is low.

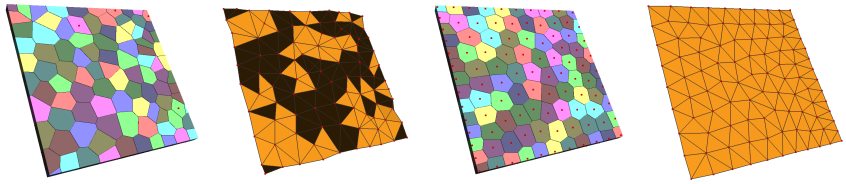
The effects are felt beyond these areas of low curvature. Since a large number of samples may be allocated to flat areas, an insufficient number of samples may be left for areas where they are inherently needed – areas that are highly detailed. The simple solution to this problem is to add more sample points, but if keeping the number of triangles low is important then an alternative solution is desired.

We propose an algorithm that largely removes the requirement for dense sampling in featurless areas. This is done through a surface segmentation algorithm that decomposes S into a set of subsurfaces $M = \{M_i\}$ such that for any $p \in M_i$, $\mathit{lfs}(p) \approx 1/\kappa(p)$ where κ is the maximum of the absolute values of the principal curvatures κ_1, κ_2 at p. We call a surface with this property "curvature dominant." Such surfaces with this property can be remeshed with enough samples to preserve features with high curvature, while flattish areas can be approximated with fewer triangles without risk of topological errors. Our remeshing of each individual subregion is done using CVT with density function $\rho = \sqrt{\kappa}$. Hence the moniker κCVT.

With remeshed subregions in place, our stitching algorithm composes them back into a single triangulation. The stitching algorithm uses information on connectivity between regions as a heuristic for accurately finding correspondences between triangles of different subregions.

In datasets with flattish regions as described, our method requires fewer triangles to produce topologically correct meshes that are geometrically very accurate.

Our processing pipeline is to first segment S into subsurfaces $\{M_i\}$. We then remesh each M_i individually using CVT, followed by stitching. Since understanding CVT remeshing is important to understanding the segmentation algorithm, our presentation in this paper follows a slightly different order. After a discussion of related work (section 1.1), we briefly describe the CVT algorithm and theorems related to topological correctness of remeshed surfaces (section 2). We then discuss the curvature dominance property and surface segmentation (section 3) followed by stitching (section 4). We end with results and conclusions (section 5).



(a) (b) (c) (d)

Fig. 2. Examples of remeshing a thin box. (a) Voronoi diagram of seeds after running CVT to convergence. The Voronoi cells are badly shaped because seeds are influencing cells on the opposite side of the box. Seeds are shown in red. Most of the seeds have drifted inside the box and are not visible. (b) Dual of the Voronoi diagram. Triangles in black are facing away from the viewer – not only are many of the triangles poorly shaped, but there are topological errors as well. (c) Half of the box has been removed. With a single sheet the Voronoi cells are as expected: fairly regular hexagons. (d) The Dual has well-shaped triangles and no topological errors.

1.1 Related work

Much work has been done in the area of surface remeshing. Some remeshing approaches are geared toward decimation, or reduction of the number of triangles, and work directly on the mesh (e.g. [13, 15]) using a series of geometric operations such as edge collapse. These approaches typically have some error metric that is maintained and decimation halts once the metric reaches a threshold. Other similar approaches use optimization [16] that maintains topology but is computationally expensive. Another approach proposed by Cheng et al. [7] uses Delaunay refinement to successfully remesh piecewise smooth meshes, including non-manifolds.

CVT is a popular remeshing technique that minimizes an energy function designed to simultaneously reproduce the input mesh faithfully while producing well-

shaped triangles. A primer on CVT in its general formulation (not necessarily applied to surface remeshing) is given by Du et al. [8]. CVT has been applied to surface remeshing. Sample points are placed on the surface and their locations are optimized by minimizing the CVT energy function in order to produce quality triangles that approximate the surface well. CVT methods fall into two camps: those that optimize the points in parameter space and those that work directly on the mesh.

Of those that parametrize the surface, Alliez et al. [2] use a parameter representation of the surface as a whole. To avoid the difficulty of entire surface parametrization, various approaches parametrize locally [3, 21]. This simplifies parametrization, but returning to 3D involves stitching, and optimization is not done globally, yielding triangles that are not consistently uniform.

Another approach is to optimize sample points directly on the surface. This has the advantage of being a global optimization and triangles have been shown to be of higher quality [22]. These methods have been hampered by two issues. The first is performance. Lloyd's algorithm [19] has been the implementation of choice to minimize the CVT energy function despite its linear convergence rate [8]. Only recently was the CVT energy function shown by Liu et al. [18] to have C^2 smoothness, making it a candidate for more efficient optimization techniques. In the same work the limited-memory BFGS method (L-BFGS) [17] was applied to the minimization of the CVT energy function with favorable results. Another work that further improved the performance of CVT is that of Yan et al. [22] which proposed an algorithm to efficiently and exactly compute the *Restricted Voronoi Diagram* (RVD), a necessary ingredient in direct methods.

The second issue with these methods is the need for high sampling rates to achieve topological correctness. While the method in [22] detects topological problems and corrects them by inserting additional samples, the requirement for large numbers of samples, possibly even in flat regions, is troublesome. Peyre et al. [20] use geodesic approximations directly, which largely obviates the sample density requirements. Their application is surface segmentation, and CVT is used to optimize two competing conditions (compactness and boundaries lying on sharp features) on surface regions. While effective for segmentation, in the context of remeshing, where the number of regions is very large, the performance is problematic. Our approach allows flat regions to be remeshed with few samples, regardless of local feature size, and does so while still using the euclidean metric.

Alliez et al. [4] provide a more thorough survey of remeshing techniques.

2 CVT remeshing

Our presentation of the CVT follows that given in [22].

The Voronoi Diagram, or Voronoi Tessellation, is defined as follows. Given n distinct sample, or seed, points $X = \{x_i\}_{i=1}^n$ in $\mathbb{R}^N$, each point x_i lies within a set of points

$$\Omega_i = \{x \in \mathbb{R}^N | \|x - x_i\| \leq \|x - x_j\|, \forall j \neq i\}. \tag{1}$$

The set of points Ω_i is called the Voronoi cell of x_i and the set of all Voronoi cells $V = \{\Omega_i\}_{i=1}^n$ is a decomposition of $\mathbb{R}^N$ and is called the Voronoi Diagram determined by X. The Centroidal Voronoi Tessellation (CVT) is a special case of the Voronoi Tessellation such that each seed $x_i \in X$ coincides with the center of mass of its corresponding Voronoi region Ω_i. Given a density function $\rho(x) > 0$, the center of mass x^* is defined as

$$x_i^* = \frac{\int_{\Omega_i} \rho(x) x \, d\sigma}{\int_{\Omega_i} \rho(x) \, d\sigma} \tag{2}$$

For computation purposes, the CVT can be formulated as a critical point of

$$F(X) = \sum_{i=1}^{n} \int_{\Omega_i} \rho(x) \|x - x_i\|^2 \, d\sigma \tag{3}$$

Equation (3) is known as the CVT energy function and is typically minimized using Lloyd's algorithm [19] or, more recently, the limited-memory BFGS method (L-BFGS) [18, 17].

In the case of surface remeshing, two modifications to CVT have been proposed. The first is Restricted CVT (RCVT). Given a surface $S \subset \mathbb{R}^3$, a set of seeds X, and the induced Voronoi Diagram V, the Restricted Voronoi Diagram (RVD) is the set of all restricted Voronoi cells (RVC) $\mathscr{R} = \{R_i\}_{i=1}^n$ where $R_i = \Omega_i \cap S$. In other words, each RVC is the Voronoi cell restricted to the surface S. RCVT uses a slightly modified energy function that utilizes the RVD:

$$F(X) = \sum_{i=1}^{n} \int_{R_i} \rho(x) \|x - x_i\|^2 \, d\sigma \tag{4}$$

Constrained CVT (CCVT) was introduced in [9] and is the same as RCVT except that the seed points are restricted to S, as given in

$$x_i^* = \arg\min_{y \in S} \int_{x \in R_i} \rho(x) \|y - x\|^2 \, d\sigma \tag{5}$$

The Restricted Voronoi Diagram is given by either CCVT or RCVT. The Restricted Delaunay Triangulation (RVT) is dual of the RVD, in that each vertex in the RVT corresponds to a cell in the RVD, two vertices share an edge if their corresponding RVD cells share an edge, and a triangle exists where three cells share a Voronoi vertex.

We use the efficient RVD computation given in [22] and the energy minimization given in [18]. For simplicity in writing, we use the term "CVT" to refer to the remeshing process comprising CCVT/RCVT and RVT.

2.1 Topological correctness

CVT does not guarantee that the output mesh W is homeomorphic to S. A theorem states a sufficient condition to avoid topological errors.

Theorem 1. *Topological ball property [10]. If each Restricted Voronoi Cell is a topological disk then the Restricted Delaunay Triangulation is homeomorphic to S.*

Figure 3 illustrates the topological ball property in 2D. If the RVD fails to meet this property, well-spaced sample points can be added, as implied in the figure. This solution is formalized in a theorem from the literature on surface reconstruction from point clouds.

Theorem 2. *r-sampling theorem [5]. If no point p on surface S is farther than $r \cdot \text{lfs}(p)$ from a seed point $x \in X$ where r is a constant then the Restricted Delaunay Triangulation induced by X is homeomorphic to S.*

If W has topological errors, which can be detected using the topological ball property, then we simply add seed points and re-run the algorithm. This is the approach taken in [22]. Eventually there are enough seeds to meet the r-sample criterion given in theorem 2 and W will be homeomorphic to S (although termination has not been proven). Since *lfs* is based on euclidean distance, points on S that are geodesically very far from a given point $p \in S$ can cause *lfs*(p) to be small, requiring the sample density to be unduly high at p, even if the curvature at p is very low. This is the very artifact that we seek to avoid.

We briefly digress to note that our approach is intended to enable meeting the topological ball property with fewer triangles in areas of low curvature. It is not a general solution for all areas of S. To illustrate this point we mention two conditions that together are sufficient to meet the topological ball property, of which only the first condition is targeted by κCVT. Let $\mathscr{B}(p, r)$ be a ball of radius r centered at p.

Condition 1 *$\mathscr{B}(p, r) \cap S$ is a single connected component.*

Condition 2 *$\mathscr{B}(p, r) \cap S$ has a single boundary.*

Condition 1 is typically violated in areas where both *lfs* and curvature are small. Condition 2 is violated in areas where S is similar to a cylinder with small radius. In this case, $\mathscr{B}(p, r) \cap S$ may be a cylinder with open ends, which is a single connected component but has two boundaries, thus is not homeomorphic to a disk. While this can cause topological problems, there is nothing to be gained in segmentation since local feature size is already dominated by curvature.

2.2 Geometric accuracy and triangle quality

CVT has been shown to give excellent results in terms of mean Hausdorff error [22]. Further, the density function ρ is an intuitive way to control the trade-off between geometric accuracy and triangle quality. Meshes produced from CVT using a constant density function $\rho = c$ tend to have very well-shaped triangles of roughly uniform size. Using such uniform triangles can lead to reduced geometric accuracy in areas of high curvature, not to mention topological problems in areas of low feature size. The logical solution is to increase the number of triangles in these areas by setting $\rho = 1/\textit{lfs}^2$. This produces gradation in the triangle sizes, affecting triangle quality.

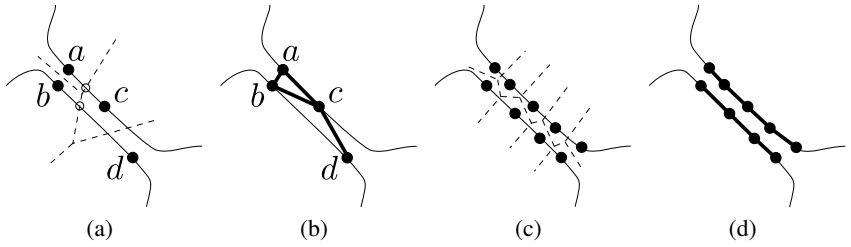


Fig. 3. 2D graphic of the RVT. Resulting meshes are not necessarily homeomorphic to the input mesh when certain conditions are not met. (a) There are only 4 sample points (black dots) in the region of interest. The (unrestricted) Voronoi diagram is shown in dashed lines. Note that the Restricted Voronoi Cell $R_c = \Omega_c \cap S$ corresponding to point c has two connected components and thus is not a topological disk. Shared "edges" between samples a and c and samples b and c are circled. These induce edges between cells in the triangulation. (b) RDT induced by RVD. Edges between sample points (bold lines) are used to reconstruct the surface. Because samples are not dense enough, the resulting surface is not homeomorphic to the input. (c) Voronoi diagram of densely-sampled points. All RVCs are topological disks. (d) Resulting surface is homeomorphic.

3 Surface segmentation

If S has flat regions with low *lfs* there are two options when remeshing with CVT. The first is to use a constant density function $\rho = c$ and risk topological errors. The second option is to use $\rho = 1/\mathit{lfs}^2$ which may produce large numbers of triangles in those regions rather than allocating them to regions of high curvature. A surface that is curvature dominant, however, can use curvature as the density function. This ensures that few triangles will be used in flat regions while still maintaining good likelihood that the topological ball property will be met. We approximate curvature using CGAL's implementation of the approach in [6]. Segmentation requires local feature size for the vertices of S which we approximate using the approach given in [1] by computing the distance to the nearest pole [5].

We first fix some notation. If we let A and C denote closed, adjacent triangles in a 2-manifold triangulated surface S, then $A \cap C$ is the shared edge between them, which we denote AC. Let V_A be the set of three vertices defining triangle A. Further, let $d(p, \Delta) = \arg\min_{q \in \Delta} \|p - q\|$ be the minimum distance between p and 1- or 2-simplex Δ. We define $r_p = 2 \cdot \alpha \cdot \mathit{lfs}(p)$ and $r_A = \arg\min_{v_i \in V_A} r_{v_i}$. All of our experiments use $\alpha = 1.1$. Let $P_{A,\Delta}$ be the set of all points $p \in A$ that are within r_A of the simplex Δ. That is, $P_{A,\Delta} = \{p \in A | d(p, \Delta) < r_A\}$. We note that for any $p \in P_{A,\Delta}, \mathscr{B}(p, r_A) \cap \Delta \neq \emptyset$. See figure 7. Let $\mathit{lfs}_S(p)$ be the local feature size at p with respect to surface S.

Compatibility table

Our algorithm partitions a surface triangulation S into subsurfaces $M = \{M_i\}$ such that the ball $\mathscr{B}(p, r_A)$ centered at any point $p \in A \in M_i$ will yield a single connected

component when intersected with M_i (figure 4). The first step in our algorithm is to build a table of all "incompatible" pairs of triangles in S (figure 5). Triangles $A \in S$ and $B \in S$ are incompatible if there exists any pair of points $p \in A$ and $q \in \mathscr{B}(p, r_A) \cap B$ such that there is no path from q to p residing entirely in $\mathscr{B}(p, r_A) \cap S$. (Note that a surface P is a single connected component *iff* for any two points $p, q \in P$ there exists a path $\Gamma_{q,p}$ from q to p such that $\Gamma_{q,p} \subset P$.) In this case, both A and B cannot exist in the same subsurface M_i without violating condition 1.

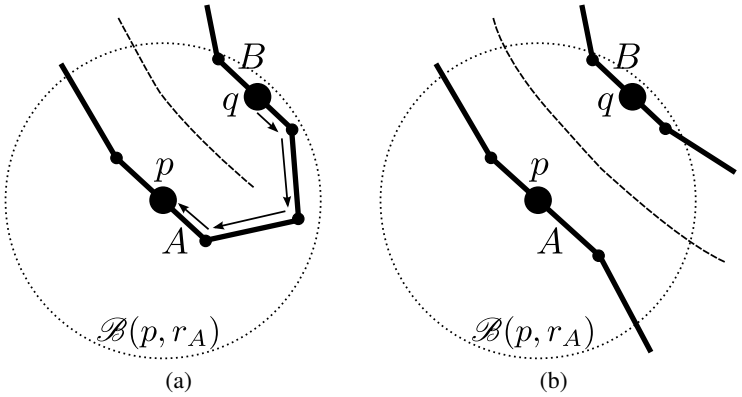


Fig. 4. 2D graphic illustrating use of lfs_S to find incompatible triangles. (a) A and B are compatible. For any $p \in A$, $q \in \mathscr{B}(p, r_A) \cap B$ there is a path from q to p, similar to the path shown along the arrows. The path lies entirely in $\mathscr{B}(p, r_A) \cap S$. (b) A and B are incompatible. There is no path from q to p.

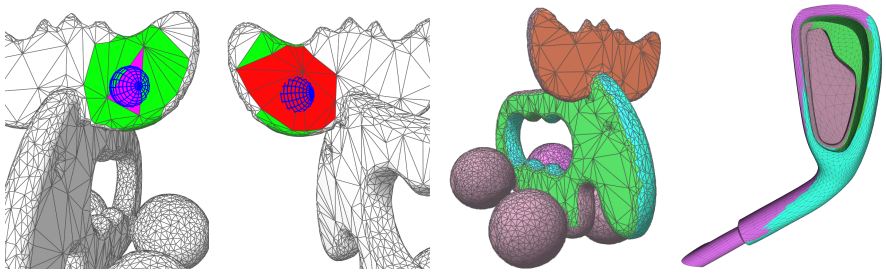


Fig. 5. Identification of incompatible triangles. (a) The triangle of interest A is in magenta. The blue ball has radius r_A. Triangles compatible with A are shown in green. Any ball $\mathscr{B}(p \in A, r_A)$ intersected with the compatible triangles will yield a single connected component. (b) Rotated to see the opposite sheet near A. Triangles within a distance r_A of A that are incompatible with A are shown in red. (c), (d) Two examples of final segmentation. All triangles of the same color are compatible with each other.

To build the compatibility table, we first construct an R-tree [14] with all triangles in S. We then iterate through each triangle $A \in S$. Given A, we find, using the R-tree, the set of all triangles $B = \{B_i\}$ such that each B_i is within r_A of A (i.e. $\exists p \in A, q \in B_i$ s.t. $\|q - p\| < r_A$). We then compute P_{A,B_i}. We do this in three steps. First we construct a prism $Z = \{q | q' \in B_i, \|q - q'\| \leq r_A\}$ around B_i (figure 6) where q' is the projection of q onto the plane defined by B_i. In other words, we sweep B_i in its normal direction by $-r_A$ to r_A. We then construct a set of three cylinders $X = \{X_{\{123\}}\}$ of radius r_A around each edge of B_i. Finally we construct a set of three spheres $Y = \{Y_{\{123\}}\}$ of radius r_A around each vertex of B_i. We can now compute $P_{A,B_i} = A \cap (Z \cup \bigcup_{X_i \in X} X_i \cup \bigcup_{Y_j \in Y} Y_j)$ and $P_{A,B_iC} = A \cap (X_j \cup Y_k \cup Y_l)$ where C is an edge-neighbor of B_i and X_j (Y_k, Y_l resp.) their shared edge (vertices). Since B_i is within r_A of A, $P_{A,B_i} \neq \emptyset$. For every point $p \in P_{A,B_i}$ and every point $q \in \mathscr{B}(p, r_A) \cap B_i$ there must exist a path $\Gamma_{q,p} \subset \mathscr{B}(p, r_A) \cap S$. Let $\mathscr{N}_{B_i}$ be the edge-neighbors of B_i.

Our algorithm uses a breadth-first search from A over all $B_i \in B$, flagging each B_i as compatible or incompatible. A and B_i are incompatible if $\bigcup_{N_j \in \mathscr{N}_{B_i}} P_{A,N_jB_i} \setminus P_{A,B_i} \neq \emptyset$. See figure 7. Each set of points $P_{A,\Delta}$ is the union of ellipses and polygons (figure 6). Using these primitives directly makes for very expensive boolean set computations, so our implementation converts $P_{A,\Delta}$ to a raster representation so that less expensive bitwise boolean operations can be used.

Region merging

Let Λ_i be a set of triangles. With the compatibility table built, we define a binary predicate operator $\Lambda_i \oplus \Lambda_j$ that yields true iff every triangle in Λ_i is compatible with every triangle in Λ_j. Segmentation proceeds as a region merge algorithm.

Data: triangles $S = \{T_i\}$
Result: regions $M = \{M_i\}$
1 Initialize one region Λ_i per triangle T_i
2 **foreach** *region* Λ_i **do**
3 **foreach** *neighbor* $\overline{\Lambda}_j$ *of* Λ_i **do**
4 **if** $\overline{\Lambda}_j \oplus \Lambda_i$ **then**
5 merge $\overline{\Lambda}_j$ and Λ_i
6 **end**
7 **end**
8 **end**
9 Repeat lines 2-8 until no regions are merged

Algorithm 1: Segment via region merging

Region merging outputs a set of subsurfaces $M = \{M_i\}$. CVT with density function $\rho = \sqrt{\kappa_S}$ is applied to each M_i. $\kappa_S(p)$ is the curvature at point p with respect to surface S. It is important to use the curvatures from the original surface, since edges with high curvature in S may be boundary edges in M_i with very low curvature. Even though the new boundary is flattish, we still desire high density there so that the stitching algorithm has border edges that have roughly the same spacing between two subsurfaces.

Remeshing

With the surface S segmented into subsurfaces we now remesh subsurfaces individually using CVT with $\rho = \sqrt{\kappa}$. The N samples must be distributed among the different subsurfaces for initial placement and optimization. We allocate N_i seeds to each subsurface M_i where $N_i = N \int_{M_i} \rho(x)\, d\sigma / \int_S \rho(x)\, d\sigma$. One subtlety in the remeshing stage is that we must ensure that adjacent boundaries of subsurfaces have compatible sample density so that subsequent stitching is uniform. For this reason, our density function uses the curvature computed with respect to the original surface S rather than with respect to a subsurface M_i.

In some cases, CVT remeshing of a segmented region yields a vertex that is shared by only two triangles such that the edges are not orderable. This is automatically detected and repaired by either breaking the triangles apart or adding two new triangles that connect them.

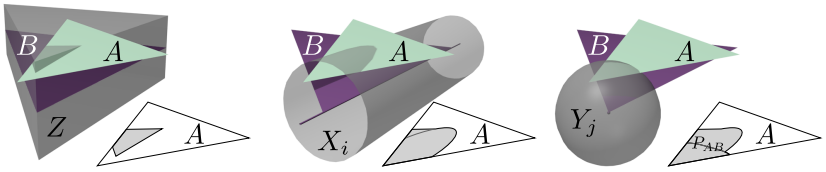


Fig. 6. Construction of P_{AB}. Triangle A is in green and triangle B is in purple. (a) Intersection of A with prism Z. The shaded portion of the 2D graphic is the intersection restricted to A. (b) Intersection of A with cylinder X_i. The other cylinders are not shown. (c) Intersection of A with sphere Y_j. The other spheres are not shown. The 2D graphic shows P_{AB}.

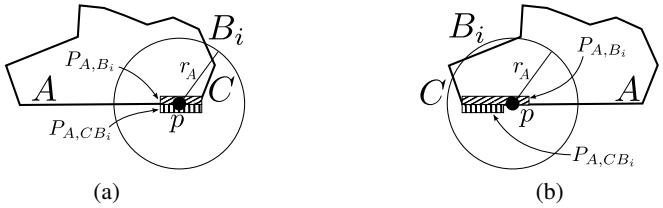


Fig. 7. 2D illustration of $P_{A,\Delta}$. (a) In this case, given that C is compatible with A, $P_{A,B_i} \setminus P_{A,CB_i} = \emptyset$, so A and B_i are compatible. That is, $\mathscr{B}(p \in A, r_A) \cap S$ is a single connected component. (b) Given C is compatible with A, $P_{A,B_i} \setminus P_{A,CB_i} \neq \emptyset$, so A and B_i are incompatible. $\mathscr{B}(p \in A, r_A) \cap S$ yields two connected components.

4 Stitching

The segmentation algorithm described in section 3 yields a set of subsurfaces $\{M_i\}$. After segmentation, but before remeshing, we build a table JUNCTIONS of n-way surface boundary intersections. If a vertex is shared by multiple subsurfaces, an entry is made in the table listing the index of each subsurface. After remeshing, we consult JUNCTIONS to find subsurface correspondences. Suppose an entry of the table consists of 3 subsurfaces M_i, M_j, and M_k. We first search to find a "connector triangle" t_c such that t_c shares a vertex with each of the three subsurfaces and is optimal in some sense. See figure 8. Let E_i be the boundary edges of M_i and let V_i be the vertices in E_i. We iterate through each vertex of one set of edges. Let V_j^p (resp. V_k^p) be the closest n_α vertices in E_j (E_k) to p in terms of euclidean distance. The set of all candidate connector triangles is $\{(p \in V_i, q \in V_j^p, r \in V_k^p)\}$. For $n_\alpha = 4$ (the value used in our experiments) there are then $16 \cdot |V_i|$ candidate connectors.

For each candidate connector triangle t_c we stitch a distance of n_β triangles in each direction (in experiments we used $n_\beta = 5$, but we use $n_\beta = 2$ in the figure for clarity). Let T_c be the set composed of t_c and the $3 \cdot n_\beta$ neighborhood stitch triangles. We assign a score to t_c using the cost function

$$cost(t_c) = \sum_{t \in T_c} area(t) \cdot Q(t)^{-\gamma}. \tag{6}$$

γ is a user-defined parameter (we used $\gamma = 0.5$) and $Q(t)$ is the triangle quality measure [11]

$$Q_t = \frac{6}{\sqrt{3}} \frac{r_t}{h_t} \tag{7}$$

where r_t and h_t are the inradius and longest edge length of t, respectively. Once all candidate triangles are scored we choose the one with the lowest score. This gives an approximately optimal triangle according to our cost measure that connects three surfaces. Once we've found connecting triangles for each entry in JUNCTIONS we simply search out from one such triangle, adding triangles along the way until we reach another connecting triangle.

5 Experimental results and conclusions

We compare our results with direct CVT methods and the method proposed by Fuhrmann et al. [12] which samples points directly on the mesh and then optimizes their placement in local parameter domains. The latter method isn't well-suited for aggressive decimation (the triangulation of initial samples fails), so we only report its results for higher sampling rates. We used 100 Lloyd iterations and used a density contrast exponent of 0.5 with no laplacian smoothing. In this discussion we refer to unsegmented CVT using the density function $\rho = 1$ as *uniform* CVT and unsegmented CVT with $\rho = 1/lfs^2$ as *lfs* CVT. We refer to our method of segmenting, CVT with $\rho = \sqrt{\kappa}$, and stitching as κCVT.

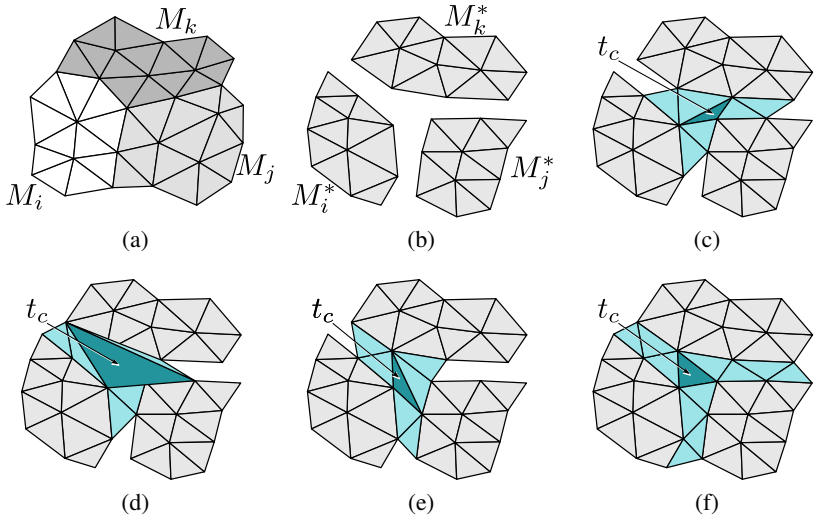


Fig. 8. Finding connector triangles and stitching. (a) Segmented triangulation. $M_{\{ijk\}}$ are three subsurfaces in M. The table JUNCTIONS has an entry (i, j, k) since the three subsurfaces share a vertex. From the JUNCTIONS table we see that M_i^*, M_j^*, and M_k^* need to be stitched. (b) Segmented regions $M_{\{ijk\}}^*$ after remeshing using CVT. (c), (d), (e) Three candidate connector triangles. The connector is dark cyan. Neighbor stitches up to $N_\beta = 2$ distance in each direction are light cyan. (f) The connector triangle which, along with associated neighbor stitches, gives the best score. Final stitching is shown.

Figure 1 uses 2000 sample points to remesh the Toy Elk model. This is far too few sample points to meet the criteria for topological correctness when using uniform CVT, and many errors result. Even the *lfs* method produces one non-manifold vertex in the horn region. Our method yields a mesh that is manifold and homeomorphic to S. In addition, our method also improves geometric error across almost all experiments, measuring geometric error with mean Hausdorff error (figure 9 and table 1). In fact, our results show geometric error improvement of as much as 20% compared to the next-best method. In the single case that another method outperformed κCVT, our method had less than half the number of topological errors (see also figure 10). In all other cases κCVT had identical or improved geometric error while reducing the number of topological errors to 0. In general there is some loss in average triangle quality when compared to uniform CVT. Similar to that of *lfs* CVT, the quality suffers due to the triangle size gradation. This is due to the fact that the optimization is not global and so even an optimal stitching algorithm cannot yield good triangles in every case.

Figure 10 is an example of our method avoiding topological errors that can result from extreme decimation. Uniform CVT has topological errors in the flat regions while *lfs* CVT has errors in the cylindrical regions (since it allocates most sample points to the flat regions). Our method distributes samples better, although it also has some errors in the cylindrical regions, as explained in section 2.1.

Figure 11 shows results from remeshing the Fish model at two levels of decimation.

Figure 12 helps explain the gains in geometric error. The upper box highlights the horn region. The method from [12] suffers in high-curvature regions due to local optimization. The uniform CVT method fails to allocate enough triangles around the edges of the horns. The *lfs* CVT method places far more triangles than needed in the flat horn region, causing a lack of triangles in the higher curvature but lower *lfs* ball region shown in the lower box. Our method uses far fewer triangles in the flat horn region, which doesn't affect geometric error, and allocates more triangles to the edges of the horns (a failing of uniform CVT) and to the ball region (a failing of *lfs* CVT).

We are interested in furthering this work by improving Q_{min} and θ_{min} by performing a final CVT energy optimization over the stitches and surrounding region. As we did not use feature preservation in this current work, we anticipate adding it in the future, similar to that done in [22]. We note that the method is parallelizable – after decomposition we can remesh the seperate segments in parallel, which we expect to implement in a distributed fashion.

Our method is a tool that can be used to more effectively control the trade-offs between geometric, topological, and triangle qualities. It is useful in cases where flat sheets pass close to each other, freeing the CVT algorithm from allocating inordinate numbers of samples into those regions while avoiding topological errors.

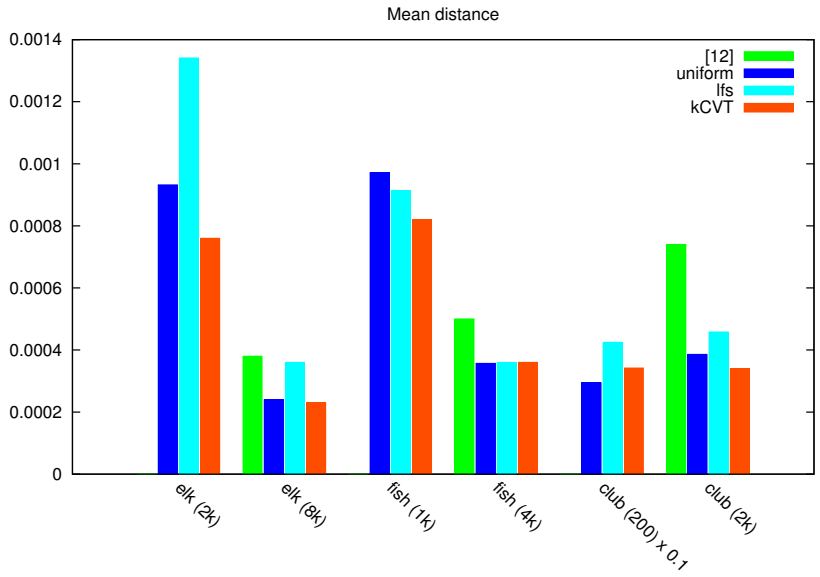
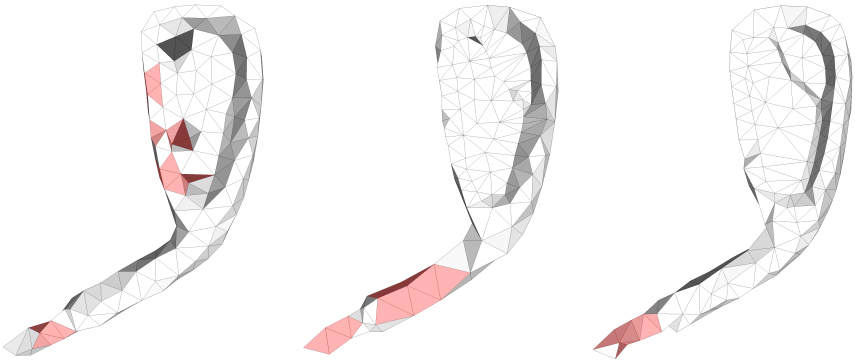


Fig. 9. Graph of mean distance from input mesh to output mesh. κCVT performed equally or better in every test but one.



(a) (b) (c)

Fig. 10. Remeshing the Club model with 200 sample points. Triangles with non-manifold edges are highlighted in red. (a) Uniform CVT. (b) *lfs* CVT. (c) κCVT.

model	# samples	method	errors	$H_{mean} \times 10^3$	$H_{RMS} \times 10^3$	Q_{min}	Q_{ave}	θ_{min}	$\theta_{min,ave}$
Elk	2000	uniform	400	0.94	1.48	0.448	0.884	22.4	50.8
		lfs	15	1.31	1.76	0.347	0.858	19.3	48.7
		κCVT	0	0.76	1.00	0.220	0.849	11.7	48.1
Elk	8000	[12]	0	0.38	0.63	0.058	0.902	2.6	52.2
		uniform	0	0.24	0.37	0.509	0.916	24.4	53.2
		lfs	0	0.36	0.49	0.451	0.893	22.6	51.4
		κCVT	0	0.23	0.34	0.259	0.885	15.2	50.9
Fish	1000	uniform	95	0.97	0.16	0.525	0.872	28.6	49.7
		lfs	14	0.91	0.12	0.420	0.830	18.3	46.4
		κCVT	0	0.82	0.12	0.236	0.809	13.1	45.0
Fish	4000	[12]	0	0.50	0.85	0.070	0.898	2.7	51.8
		uniform	11	0.36	0.53	0.580	0.898	26.3	51.7
		lfs	0	0.36	0.51	0.407	0.864	19.4	49.1
		κCVT	0	0.36	0.58	0.160	0.863	6.5	49.0
Club	200	uniform	51	2.94	4.08	0.570	0.842	30.1	47.4
		lfs	31	4.25	6.36	0.362	0.770	13.8	41.8
		κCVT	19	3.42	4.92	0.173	0.728	9.2	39.5
Club	2000	[12]	-	0.74	1.54	~ 0	0.832	~ 0	47.5
		uniform	0	0.39	0.70	0.555	0.893	32.9	51.5
		lfs	0	0.46	0.85	0.314	0.834	12.5	46.8
		κCVT	0	0.34	0.62	0.082	0.855	4.5	48.5

Table 1. Table of quality statistics of our method compared to CVT without pre-segmentation. *errors* is the number of faces with non-manifold edges or vertices. H_{mean} and H_{RMS} are the mean and RMS of one-way distance, or error, from S to W divided by the bounding box diagonal, respectively. Error in the opposite direction is similar. Q_{min} (resp. Q_{ave}) is the minimum (average) Q-measure given by equation (7). θ_{min} (resp. $\theta_{min,ave}$) is the minimum (average minimum) triangle angle.

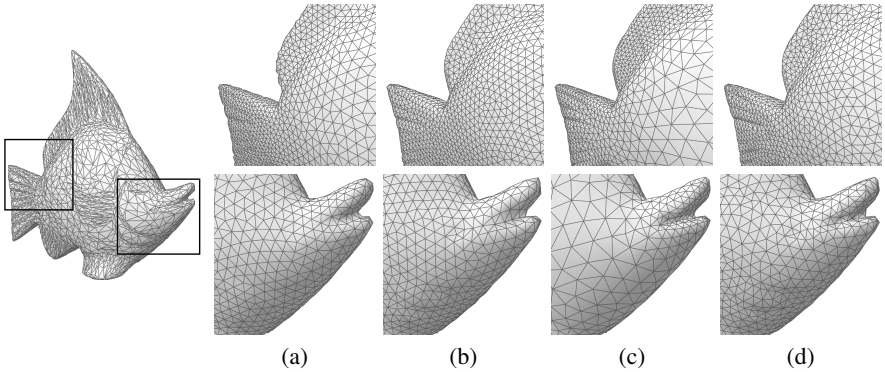


Fig. 11. Remeshing the Fish model using 4000 sample points. Original model is shown on the left. (a) Algorithm from [12]. (b) Uniform CVT. (c) *lfs* CVT. (d) Our method, κCVT. Our method preserves features of the gills below the mouth (features that are lost using [12] and *lfs* CVT) while avoiding topological errors (uniform CVT has 11 non-manifold edges).

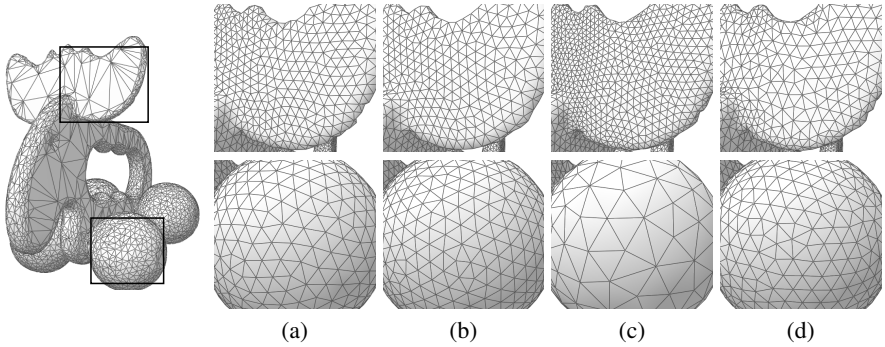


Fig. 12. Remeshing the Toy Elk model using 8000 sample points. Original model is shown on the left. (a) Algorithm from [12]. (b) Uniform CVT. (c) *lfs* CVT. (d) Our method, κCVT. Our method produces improved meshes by distributing samples according to curvature rather than local feature size while avoiding topological errors.

Acknowledgements

The Elk and Fish models are from aim@shape. The Club model is courtesy Hughes Hoppe. The implementation of the algorithm in [12] is courtesy Simon Fuhrmann. We appreciate the helpful comments of the reviewers. The work of Wenping Wang was partially supported by the Research Grant Council of Hong Kong (718209 and 718010), the National Basic Research Program of China (2011CB302400), and the State Key Program of NSFC project (60933008). The work of Chandrajit Bajaj was supported in part by NIH (R01-EB00487 and R01-GM074258), and a grant from the UT-Portugal colab project. This work was done while John Edwards was visiting The University of Hong Kong.

References

1. P. Alliez, D. Cohen-Steiner, M. Yvinec, and M. Desbrun. Variational tetrahedral meshing. *ACM Transactions on Graphics*, 24(3):617–625, 2005.
2. P. Alliez, EC de Verdire, O. Devillers, and M. Isenburg. Isotropic surface remeshing. In *Shape Modeling International, 2003*, pages 49–58. IEEE, 2003.
3. P. Alliez, M. Meyer, and M. Desbrun. Interactive geometry remeshing. *ACM Transactions on Graphics*, 21(3):347–354, 2002.
4. P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. Recent advances in remeshing of surfaces. *Shape analysis and structuring*, pages 53–82, 2008.
5. N. Amenta and M. Bern. Surface reconstruction by voronoi filtering. *Discrete & Computational Geometry*, 22(4):481–504, 1999.
6. F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121–146, 2005.
7. S.W. Cheng, T.K. Dey, and E.A. Ramos. Delaunay refinement for piecewise smooth complexes. *Discrete & Computational Geometry*, 43(1):121–166, 2010.
8. Q. Du, V. Faber, and M. Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM review*, pages 637–676, 1999.
9. Q. Du, M.D. Gunzburger, and L. Ju. Constrained centroidal voronoi tessellations for surfaces. *SIAM Journal on Scientific Computing*, 24(5):1488–1506, 2003.
10. H. Edelsbrunner and N.R. Shah. Triangulating topological spaces. *International Journal of Computational Geometry and Applications*, 7(4):365–378, 1997.
11. P.J. Frey and H. Borouchaki. Surface mesh evaluation. 1997.
12. S. Fuhrmann, J. Ackermann, T. Kalbe, and M. Goesele. Direct resampling for isotropic surface remeshing. In *Vision, Modeling, and Visualization*, pages 9–16, 2010.
13. M. Garland and P.S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 209–216. ACM Press/Addison-Wesley Publishing Co., 1997.
14. A. Guttman. *R-trees: a dynamic index structure for spatial searching*, volume 14. ACM, 1984.
15. H. Hoppe. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108. ACM, 1996.
16. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 19–26. ACM, 1993.
17. D.C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.
18. Y. Liu, W. Wang, B. Lévy, F. Sun, D.M. Yan, L. Lu, and C. Yang. On centroidal voronoi tessellation – energy smoothness and fast computation. *ACM Transactions on Graphics*, 28(4):101, 2009.
19. S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.
20. G. Peyre and L. Cohen. Surface segmentation using geodesic centroidal tesselation. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 995–1002. IEEE, 2004.
21. V. Surazhsky, P. Alliez, C. Gotsman, et al. Isotropic remeshing of surfaces: a local parameterization approach. 2003.
22. D.M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang. Isotropic remeshing with fast and exact computation of restricted voronoi diagram. In *Computer graphics forum*, volume 28, pages 1445–1454. Wiley Online Library, 2009.