Mahalanobis Centroidal Voronoi Tessellations

Ronald Richter, Marc Alexa

TU Berlin

Abstract

Anisotropic centroidal Voronoi tessellations (CVT) are a useful tool for segmenting surfaces in geometric modeling. We present a new approach to anisotropic CVT, where the local distance metric is learned from the embedding of the shape. Concretely, we define the distance metric implicitly as the minimizer of the CVT energy. Constraining the metric tensors to have unit determinant leads to the optimal distance metric being the inverse covariance matrix of the data (i.e. Mahalanobis distances). We explicitly cover the case of degenerate covariance and provide an algorithm to minimize the CVT energy. The resulting technique has applications in shape approximation, particularly in the case of noisy data, where normals are unreliable. We also put our approach in the context of other techniques. Among others, we show that Variational Shape Approximation can be interpreted in the same framework by constraining the metric tensor based on another norm.

1. Introduction

Centroidal Voronoi tessellations (CVT) have become an important tool for segmenting surfaces in geometric modeling. It has been shown that using *anisotropic* Voronoi cells has advantages for representing surfaces with anisotropic principal curvatures [1, 2, 3]. Yet, this requires the surface to carry information on the anisotropy (such as the curvature tensor).

In this work, we propose an alternative technique for defining the anisotropy in centroidal Voronoi tessellations. Our main observation is that the *embedding* of the surface provides this information. The local metric on the surface is well-approximated by an ellipsoid in ambient space, whose axes are aligned with the principal curvature axes [4]. Intuitively, these ellipsoids are the local anisotropic metric of the Voronoi cells. We constrain these ellipsoids to have the same volume. Then, minimizing the CVT energy (see Section 2) means minimizing the total volume of the ellipsoids, leading to ellipsoids that adapt to the surface data locally and, thus, represent its anisotropy. Hence, we *define the local metric as the minimizer of the CVT energy*.

We minimize the energy with a descent approach in the spirit of Lloyd's method [5] (see Section 3). This requires computing a locally optimal location and metric for a fixed cell. We show that this optimal metric is the inverse of the covariance of the data in each cell. In other words, the distance metric in each cell turns out to be the Mahalanobis distance [6] for each cell (up to scale).

We apply our technique to generating polygonal meshes from triangulated surfaces (see Section 4) and compare it to Variational Shape Approximation (VSA) [7]. Our approach is particularly well suited for tolerating noise. It could also be applied directly to surface represented as point sets, i.e. coming from depth images.

Our approach shows several connections between techniques we have not yet seen discussed in the literature: CVT, Gaussian mixture models fitted with expectation maximization (EM) [8] and VSA. We make these observations concrete in Section 5. These connections suggest that CVT will benefit from advances on EM in the machine learning literature.

2. Background and setup

The goal of CVT is to decompose a set $\mathcal{S} \subset \mathbb{R}^n$ into subsets $\mathcal{S}_i$. Each subset $\mathcal{S}_i$ is defined as the set of points $\mathbf{p} \in \mathcal{S}$ closest to a *site* or *generator* $\mathbf{x}_i$:

$$\mathcal{S}_i = \{\mathbf{p} : d(\mathbf{x}_i, \mathbf{p}) < d(\mathbf{x}_j, \mathbf{p}), \forall\, j \neq i\}. \tag{1}$$

The positions of the sites are governed by the functional

$$F(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}) = \sum_i \int_{\mathcal{S}_i} d^2(\mathbf{x}_i, \mathbf{p})\, d\mathbf{p}. \tag{2}$$

The local minima of F are called centroidal Voronoi tessellations with respect to d.

The classical choices for the distance function $d(\mathbf{x}_i, \mathbf{p})$ are the Euclidean distance in ambient space [5], the intrinsic distance along geodesics [9] (assuming $\mathcal{S}$ is equipped with a suitable metric) or the Euclidean distance in higher-dimensional space [10]. In the literature, however, several variants have been suggested:

- The integrand in F (rather than d) depends on $\mathbf{p}$. A simple case is an isotropically weighted Euclidean distance, i.e.

$$d_{\mathbf{p}}(\mathbf{x}_i, \mathbf{p}) = \rho(\mathbf{p})\|\mathbf{x}_i - \mathbf{p}\|. \tag{3}$$

  This is commonly used for stippling, e.g. [11]. More general are anisotropic weights [2, 3]

$$d(\mathbf{x}_i, \mathbf{p}) = \left((\mathbf{x}_i - \mathbf{p})^{\mathsf{T}}\mathbf{M}(\mathbf{p})(\mathbf{x}_i - \mathbf{p})\right)^{\frac{1}{2}}, \tag{4}$$

  where $\mathbf{M}$ is a symmetric matrix representing a metric tensor varying over $\mathcal{S}$.

- An additive constant in the distance function

$$d(\mathbf{x}_i, \mathbf{p}) = \|\mathbf{x}_i - \mathbf{p}\| + c_i \tag{5}$$

  leads to *power* diagrams [12], whose centroidal version also has been used for stippling-type applications [13, 14].

- Modification of the distance based on a metric tensor defined per cell $\mathcal{S}_i$ [1]

$$d(\mathbf{x}_i, \mathbf{p}) = \left((\mathbf{x}_i - \mathbf{p})^{\mathsf{T}}\mathbf{M}_i(\mathbf{x}_i - \mathbf{p})\right)^{\frac{1}{2}}. \tag{6}$$

In all approaches above, the modification of the metric is based on auxiliary data attached to the set $\mathbf{S}$ — in other words, *it is part of the input* to the minimization.

Our formulation of the minimization problem is identical to the last modification, i.e. each $\mathcal{S}_i$ is defined by a location $\mathbf{x}_i$ and a metric tensor $\mathbf{M}_i \in \mathbb{R}^{n\times n}$. However, in our approach the $\mathbf{M}_i$ are themselves minimizers of the functional

$$F(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{M}_0, \ldots, \mathbf{M}_{k-1}) = \sum_i \int_{\Omega_i} (\mathbf{p} - \mathbf{x}_i)^{\mathsf{T}}\mathbf{M}_i(\mathbf{p} - \mathbf{x}_j)\, d\mathbf{p}. \tag{7}$$

Minimizing F without extra constraints results in the trivial solution $\mathbf{M}_i = \mathbf{0}$. To guarantee that the unit distance ellipsoids cannot degenerate, we constrain the determinant of $\mathbf{M}_i$ to unity. Our objective is thus

$$\min_{\mathbf{M}_i = \mathbf{M}_i^{\mathsf{T}}, |\mathbf{M}_i| = 1} F(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}, \mathbf{M}_0, \ldots, \mathbf{M}_{k-1}). \tag{8}$$

Next, we discuss how to compute the minimization.

3. Optimization

Our general strategy is similar to Lloyd's approach [5]: start with $\mathbf{x}_i$ randomly sampled from $\mathcal{S}$ and $\mathbf{M}_i = \mathbf{I}$ and then repeat the following steps until convergence:

1. Compute cells $\mathcal{S}_i$ for fixed $\mathbf{x}_i, \mathbf{M}_i$ based on Eq. 1.
2. Optimize $\mathbf{x}_i, \mathbf{M}_i$ for fixed $\mathcal{S}_i$ based on Eq. 8.

We note that in each step the functional F cannot increase. Hence, the algorithm is guaranteed to find a local minimum.

For reasons of exposition, we start with the derivation of $\mathbf{x}_i, \mathbf{M}_i$. Where necessary for concrete algorithmic explanations, we consider $\mathcal{S}$ to be given as a triangulated surface.

3.1. Computing the local distance metric

We now consider the problem of finding the local centroids and local metric tensors for fixed cells $\mathcal{S}_i$. The optimization can be performed per cell and we drop the index i for convenience. Our objective is

$$\min_{\mathbf{x}, \mathbf{M} = \mathbf{M}^{\mathsf{T}}, |\mathbf{M}| = 1} \int (\mathbf{p} - \mathbf{x})^{\mathsf{T}}\mathbf{M}(\mathbf{p} - \mathbf{x})\, d\mathbf{p}. \tag{9}$$

To solve this problem, we consider the Lagrangian

$$L = \int (\mathbf{p} - \mathbf{x})^{\mathsf{T}}\mathbf{M}(\mathbf{p} - \mathbf{x})\, d\mathbf{p} + \lambda\,(|\mathbf{M}| - 1) \tag{10}$$

and set the gradient w.r.t. the unknowns to zero. Doing so for the location $\mathbf{x}$ yields

$$\mathbf{0} = \frac{\partial L}{\partial \mathbf{x}} = \int \mathbf{M}(\mathbf{p} - \mathbf{x})\, d\mathbf{p} = \mathbf{M}\left(\int \mathbf{p}\, d\mathbf{p} - \mathbf{x}\int 1\, d\mathbf{p}\right) \tag{11}$$

which gives the well known fact that the location $\mathbf{x}$ that minimizes squared distances is the (weighted) centroid, regardless of the metric tensor. That is we have

$$\mathbf{x} = \frac{\int \mathbf{p}\, d\mathbf{p}}{\int 1\, d\mathbf{p}}. \tag{12}$$

For taking the gradient w.r.t. $\mathbf{M}$ we note that

$$\frac{\partial(\mathbf{p} - \mathbf{x})^{\mathsf{T}}\mathbf{M}(\mathbf{p} - \mathbf{x})}{\partial \mathbf{M}} = (\mathbf{p} - \mathbf{x})(\mathbf{p} - \mathbf{x})^{\mathsf{T}} \quad \text{([15], Eq. 72)}$$

and

$$\frac{\partial |\mathbf{M}|}{\partial \mathbf{M}} = |\mathbf{M}|\left(\mathbf{M}^{-1}\right)^{\mathsf{T}}. \quad \text{([15], Eq. 49)}$$

Denoting the covariance matrix as

$$\mathbf{C} = \int (\mathbf{p} - \mathbf{x})(\mathbf{p} - \mathbf{x})^{\mathsf{T}} d\mathbf{p}. \tag{13}$$

Let us write the necessary condition for the derivative of the Lagrangian as

$$\mathbf{0} = \frac{\partial L}{\partial \mathbf{M}} = \mathbf{C} + \lambda |\mathbf{M}| \left(\mathbf{M}^{-1}\right)^{\mathsf{T}}. \tag{14}$$

Assuming $\mathbf{C}$ is non-singular, we can denote the solution as

$$\mathbf{M} = |\mathbf{C}|^{\frac{1}{n}} \mathbf{C}^{-1}, \tag{15}$$

where the scale factor $|\mathbf{C}|^{\frac{1}{n}}$ ensures $|\mathbf{M}| = 1$, as desired. Note that the desired symmetry of $\mathbf{M}$ follows from the symmetry of $\mathbf{C}$.

This solution is a continuous analogy (up to scale) to the Mahalanobis distance [6]. How $\mathbf{C}$ can be computed for a set of triangles is described in the appendix. Figure 1 illustrates the distances on the surface to the sites, as well as the corresponding covariances $\mathbf{C}$ and metric tensors $\mathbf{M}$ as ellipsoids.

3.2. Degenerate sites and anisotropy

To obtain the solution for the local metric $\mathbf{M}$ in Eq. 15 we had to assume that the local covariance matrix is non-singular. However, in practice it can occur that the variance in one or more directions is zero. Note that it is a feature of our approach that it generates cells that are close to this condition. Hence the situation will naturally occur and we cannot ignore it or treat it merely as a numerical problem.

To understand the problem better, consider the eigendecomposition

$$\mathbf{C} = \mathbf{D}\, \mathrm{diag}(\lambda_0, \ldots, \lambda_{n-1})\, \mathbf{D}^{\mathsf{T}}. \tag{16}$$

Note that the eigenvalues λ_j are real and non-negative because $\mathbf{C}_j$ is real symmetric and positive-semidefinite. We assume them to be in decreasing order. Degeneracy of a cell corresponds to one or more of the eigenvalues being zero. Without loss of generality, we assume exactly one eigenvalue to be zero, i.e. $\lambda_{n-1} = 0$.

As before, it is desirable that the distance weighting is inversely proportional to the variance *for those directions with non-zero variance*. For the direction with zero-variance, there is no obvious choice for distance weighting. This means $\mathbf{M}$ is of the form

$$\mathbf{M} = \mathbf{D}^{\mathsf{T}}\, \mathrm{diag}(\lambda_0^{-1}, \ldots, \lambda_{n-2}^{-1}, \gamma)\, \mathbf{D}. \tag{17}$$

Together with the condition $|\mathbf{M}| = 1$ this yields a one-parameter family of solutions dependent on γ. Increasing γ, decreases all other eigenvalues indirectly through the constraint

$$1 = |\mathbf{M}| = \lambda_0^{-1} \cdots \lambda_{n-2}^{-1} \gamma \tag{18}$$

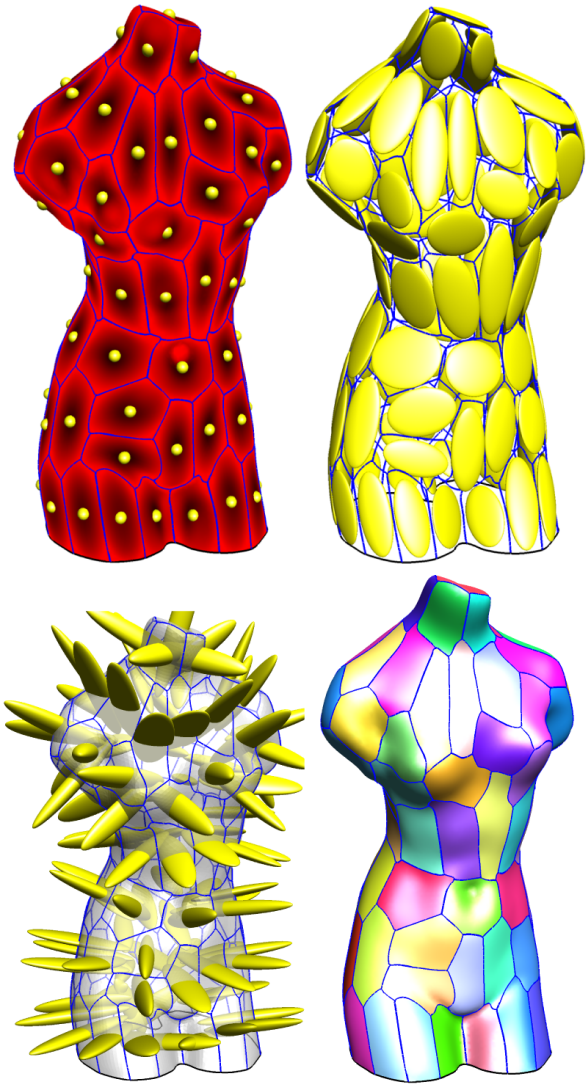

Figure 1: Top left: Illustration of the Mahalanobis distance between vertices and the associated cell centroid (red shade equals distance). Top right: Interpolated covariance matrix visualized by ellipsoids within each cell. Bottom left: Computed metric tensor visualized by ellipsoids within each cell. Bottom right: Computed regions, i.e. intersections of cells with triangle mesh.

and, thus, decreases the sum of weighted squared distances. Consequently, the minimum is attained for $\gamma \to \infty$.

In order to define a finite solution for the metric in all cases we modify the covariance of cells. Intuitively, we assign a small ϵ-ball to the empty cell. We make the size of this ball relative to the largest eigenvalue of $\mathbf{C}$. So, conceptually, we re-define the covariance of a site to be

$$\mathbf{C}' = (1 - \epsilon)\mathbf{C} + \epsilon \lambda_0 \mathbf{I}, \tag{19}$$

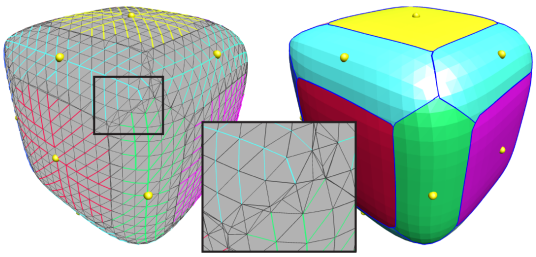

Figure 2: The colored edges illustrate the shortest paths originating from the cell centroids under the respective cell metric (left). The orignal triangle mesh is subdivided by cutting edges which intersect with cell boundaries (left, center). By subdividing the original mesh each face can be uniquely assigned to a cell (right).

which, conveniently, yields the standard isotropic model for $\epsilon = 1$.

This effectively means the inverse eigenvalues are defined as

$$\lambda_j^+ = \Big((1-\epsilon)\lambda_j + \epsilon\lambda_0\Big)^{-1}, \tag{20}$$

which is well-defined because $\lambda_j \geq 0$, as explained above. We hence have for the metric tensor of a site:

$$\mathbf{M} = \left(\prod_{j=0}^{n-1} \lambda_j^+\right)^{-\frac{1}{n}} \mathbf{D}^\mathsf{T} \operatorname{diag}(\lambda_0^+, \ldots, \lambda_{n-1}^+)\, \mathbf{D}. \tag{21}$$

3.3. Computing the cells

The anisotropic distance measure leads to complicated cells. We are only concerned with the restriction of the cells to $\mathcal{S}$. We assume that the surface is given as a triangulation and use the combinatorial structure to define neighborhoods along the surface. Based on this connectivity, we re-define the restricted cells to be the simply connected component of the surface containing the site $\mathbf{x}_i$ (similar to [16]).

The global structure of the algorithm is as follows: first, we assign each vertex in the triangulation to one of the sites, essentially by a Dijsktra-like traversal; second, edges with vertices assigned to different sites are identified and an intersection point on the edges is computed; third, on triangles with three intersected edges a Voronoi vertex is approximated.

Assigning vertices. As a first step, we identify a representative vertex for each site $\mathbf{x}_i$, which is the closest vertex to $\mathbf{x}_i$ (we assume that this assignment identifies distinct vertices for the $\mathbf{x}_i$; if not, sites are discarded). To reduce the computational complexity of this step, we generate a *kd*-tree over the vertices for nearest neighbor search. The key step in nearest neighbor search based on *kd*-trees [17] is culling a sub-tree if the squared distance

$$d_i^2(\mathbf{p}) = (\mathbf{x}_i - \mathbf{p})^\mathsf{T} \mathbf{M}_i (\mathbf{x}_i - \mathbf{p}) \tag{22}$$

of any point $\mathbf{p}$ in the half space represented by the subtree to $\mathbf{x}_i$ is larger than the squared distance of the current best candidate vertex [18]. Because the distance measure is strictly convex, this means computing the distance between $\mathbf{x}_i$ and the plane representing the half space satisfying $\mathbf{n}^\mathsf{T}\mathbf{p} > c$. For a point on the ellipsoid having critical distance to the plane $\mathbf{n}^\mathsf{T}\mathbf{p} = c$ its normal must be parallel to $\mathbf{n}$. The normals on the distance ellipsoid are parallel to the gradient of the distance function

$$\frac{\partial d_i(\mathbf{p})}{\partial \mathbf{p}} = \mathbf{M}_i(\mathbf{x}_i - \mathbf{p}), \tag{23}$$

so the point on a plane with normal $\mathbf{n}$ closest to the distance ellipsoid around $\mathbf{x}_i$ has to be on the line defined by

$$\mathbf{M}_i(\mathbf{x}_i - \mathbf{p}) = \mu\mathbf{n} \quad \Rightarrow \quad \mathbf{p}(\mu) = \mu\mathbf{M}_i^{-1}\mathbf{n} - \mathbf{x}_i. \tag{24}$$

Intersecting this line with the plane $\mathbf{n}^\mathsf{T}\mathbf{p} = c$ yields μ as

$$\mu = \frac{c + \mathbf{n}^\mathsf{T}\mathbf{x}_i}{\mathbf{n}^\mathsf{T}\mathbf{M}_i^{-1}\mathbf{n}}. \tag{25}$$

The closest point on the plane is obtained by evaluating the parametric line at μ. This point can then be evaluated for its distance $d_i(\mathbf{p}(\mu))$, which is used for culling subtrees.

The vertices are added to a priority queue based on their squared distance to the closest site. The front element in the queue is processed as follows: the vertex is assigned to the closest site i. For all neighbors of the vertex the weighted squared distance to $\mathbf{x}_i$ is computed. Conceptually, vertices are added with the computed distance to the queue if they a) are not queued already or b) the computed distance is smaller than the distance currently associated with the vertex. In practice, it is easier to always add a vertex to the queue and discard vertices that have already been assigned to a site. After processing all queue elements we have a complete assignment of vertices to cells (see Figure 2, left).

Computing intersections. For a better approximation of the cell regions on the mesh, we subdivide the original mesh by splitting edges lying on region boundaries. Those edges are identified by comparing the cell assignments of the incident vertices. If they are different, the edge intersects the boundary of both cells. In this case we approximate the intersection by a bisection method and add a new vertex at this position (see Figure 2, left and center).

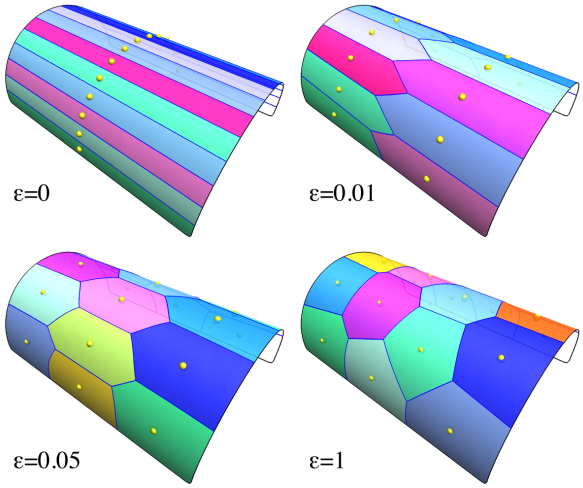

Figure 3: Changing parameter ϵ effects the cell shapes and thus the final segmentation. For ϵ near zero the cells adapt to geometry of the mesh. For larger ϵ the cells get more isotropic and finally ($\epsilon = 1$) approach Voronoi properties.

If all edges of a triangle are intersected, we assume the restriction of the Voronoi edge to the surface is inside the triangle (which may or may not be the case). We approximate the location of the Voronoi vertex on the surface as the centroid of the intersection points on the edges.

Intersecting edges and adding Voronoi vertices on triangles induces new triangles. In this augmented triangulation, each triangle can be uniquely assigned to a cell (see Figure 2, right).

We note that this piecewise linear approximation to the true restricted Voronoi cells (apart from having approximate geometry) might be topologically incorrect. In particular, we restrict the topology of the Voronoi complex on each edge to at most two cells meeting in one point; and on each triangle to at most three cells meeting in one point. If the true topology of the Voronoi complex is more complex than this, it is necessarily simplified by the approximation. However, as long as the Voronoi complex is coarse relative to the triangulation, we have not observed any adverse effects resulting from the piecewise linear approximation.

4. Evaluation & Applications

4.1. Controlling cell shapes

The parameter ϵ (see Section 3.2) influences the shape of the cells and thus the final segmentation of the mesh. By setting $\epsilon = 0$ we approach the limiting case where the metric tensor is computed by the unmodified covariances of each cell. As a result, the optimized cells maximally adapt to the local geometry of the mesh. However, to avoid degenerate cells in planar regions (singular covariance matrices) we have to set ϵ to be greater than zero. In order to generate more compact cells, we choose higher values for ϵ, i.e. increasing the isotropy of the metric tensor. In the extremal case, when we set $\epsilon = 1$, we retrieve a standard centroidal Voronoi tessellation of the mesh. Figure 3 shows the final segmentations of the same mesh with the same number of sites but with different values for parameter ϵ. For most of our experiments we typically select small values for ϵ (in the range $[10^{-2}, 10^{-6}]$) to exploit the anisotropy property of the metric tensor but also to avoid instabilities due to degenerate cells.

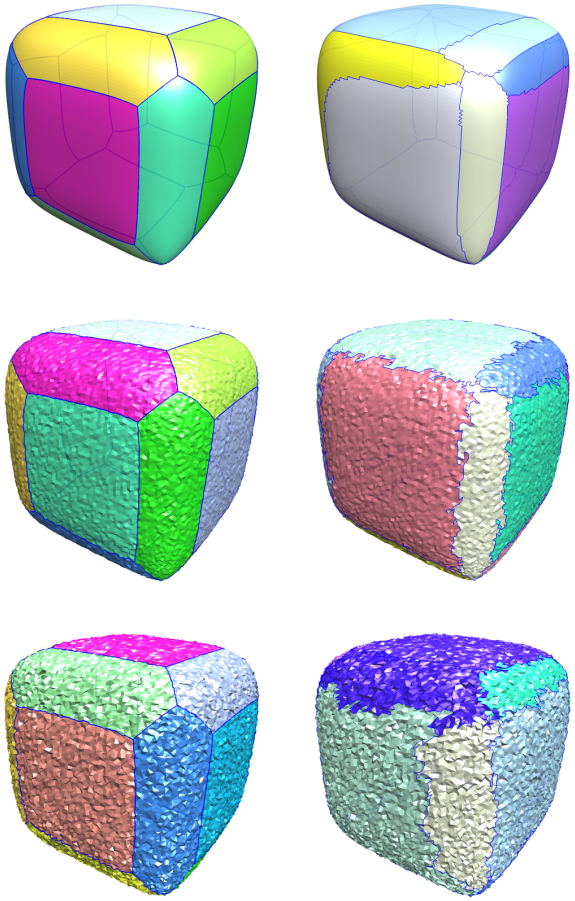

Figure 4: Segmentations generated by our algorithm (left) and Variational Shape Approximation with $L^{2,1}$ metric (right). The original mesh (top) was modified by applying a distortion function on vertex positions in normal direction with different intensities of Gaussian noise (middle: $s = 0.001$, bottom: $s = 0.002$, see Section 4.2). One can observe that our algorithm is more robust against noise than VSA.

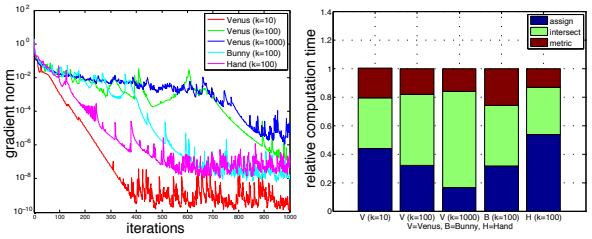

Figure 5: Left: Gradient norm $|\nabla F|$ over 1000 iterations for different models and/or different number of cells k. Right: Relative computation times of different parts of the algorithm: (a) assigning vertices to cells, (b) intersect mesh at region boundaries and (c) compute metric tensors and cell centroids

Model	k	time (sec) / iter.	
		max. iter.	$\|\nabla F\| < 10^{-5}$
Venus (47k)	10	26.5 / 1000	5.3 / 202
Venus (47k)	100	40.0 / 1000	32.2 / 805
Venus (47k)	1000	94.3 / 1000	80.1 / 849
Bunny (253k)	100	254.7 / 1000	119.3 / 462
Hand (1688k)	100	2362.6 / 1000	623.2 / 264

Table 1: Computation times for different models and number of cells (k). The last two columns show the computation time (in seconds) along with the number of iterations in two versions: a) the number of iterations is constant (1000) and b) the number of iteration depends on $|\nabla F|$, i.e. the algorithm stops if $|\nabla F|$ is smaller than a threshold (10^{-5}).

4.2. Robustness against noise

We evaluated the stability of our algorithm in the presence of noise in vertex positions. In our experiments we randomly displace all vertices along the normal direction. The magnitude of displacement is drawn from the normal distribution $\mathcal{N}(\mu, \sigma^2)$ with mean $\mu = 0$ and standard deviation $\sigma = sB_{diag}$, where s is the parameter to control the displacement relative to the bounding box diagonal B_{diag} of the mesh. Figure 4 shows the segmentation results for varying noise levels applied on the same input mesh. Although the vertex distortion may change the final segmentation we still obtain geometry-aligned segments and we do not observe instabilities due to noise.

We employed VSA with $L^{2,1}$ metric on the same input meshes with the same initial sites and equal number of iterations. For smooth surfaces we obtained comparable segmentation results. However, on noisy input meshes VSA tends to produce inaccurate segmentations or may not converge (see figure 4, right). We note that the recommended version of VSA make use of normal information, which is likely unreliable in the presence of noise.

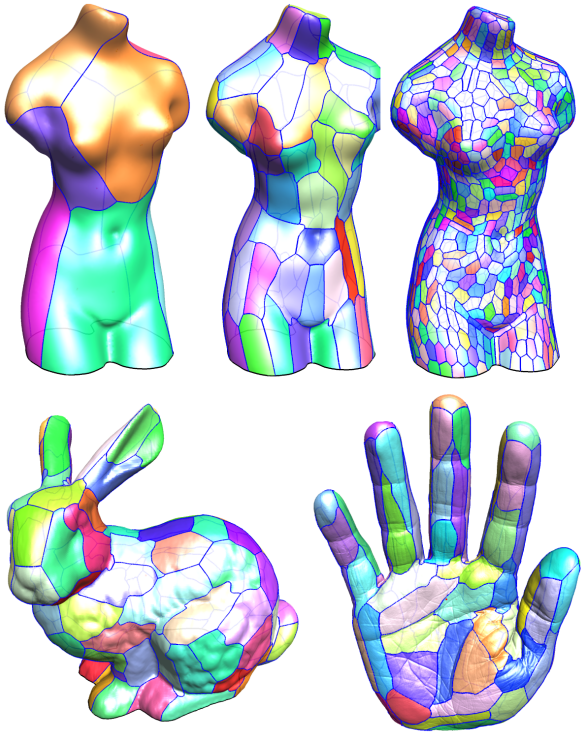

Figure 6: Models for time measurements: Venus (47k faces, 10,100,1000 segments), Bunny (253k faces, 100 segments), Hand (1688k faces, 100 segments). Results after 1000 iterations with $\epsilon = 10^{-3}$.

4.3. Time statistics

We gathered computation time statistics for a set of exemplary input models and parameters (see Figure 6). In a first experiment we fixed the number of iterations to 1000 and in a second experiment we added a stopping condition: the optimization halts when the gradient norm $|\nabla F|$ is smaller than 10^{-5} (similar to [19]) which turned out to be reasonable in practice. The plots of the gradient norm are shown on the left side of Figure 5 for all experiments (log-scale). Table 1 summarizes the overall computation times for different configurations. As expected, the time complexity is proportional to the number of faces and the number of cells. Figure 5 (right) shows the relative computation times of the three optimization steps: assigning vertices to cells, subdivide mesh at region boundaries and computing the metric tensor for all cells (as described in 3). For the same model the relative time for subdividing the mesh along region boundaries increases with the number of cells. For all experiments and measurements we use a standard machine (Intel Xeon 2.66 GHz) without exploiting parallelization techniques.

4.4. Farthest-point initialization

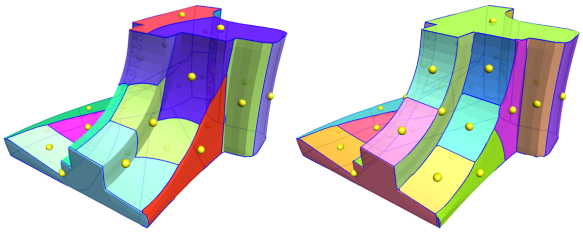

Figure 7: Fandisk model with random sampling initialization of sites (left) and farthest point initialization (right).

Instead of sampling points randomly on the surface to initialize sites we can employ the farthest point sampling heuristic to place sites on the surface. Starting with one randomly sampled site we sequentially add a new one when the iterative optimization for the current set of sites reaches one of the stopping criteria (maximum number of iterations reached or algorithm converged). We define the farthest point as the vertex which deviates most from the tangential plane induced by the covariance matrix of the associated cell. The motivation for this approach is to add more sites in regions with high curvature whereas planar regions are sufficiently represented by a single site. Especially for non-smooth surfaces this strategy leads more satisfactory results. Figure 7 shows the difference between random sampling and farthest point initialization of sites.

4.5. Polygonal mesh extraction

A typical application of mesh segmentations is to simplify highly tesselated meshes to coarser surface representations. With our segmentation algorithm we are able to generate maximal planar segments which minimize the error of a linear approximation of all segments. Converting a given segmentation into a polygonal mesh is straightforward: First, we identify the Voronoi vertices, i.e. the intersection points of three cells with the surface. Second, we connect the Voronoi vertices based on the inherent combinatoric information of the segmentation. We obtain the adjacencies from the trisection triangles, i.e. triangles with three unique cell assignments at their vertices (see Section 3.3). We also keep the ordering of the segments at those triangles with respect to the global mesh orientation. With the local informations of cell adjacencies and ordering we are able to extract the combinatorics of the whole segmentation. We finalize the mesh generation by assigning the intersection points of the Voronoi vertices to the polygonal mesh vertices. Figure 8 shows the results of this

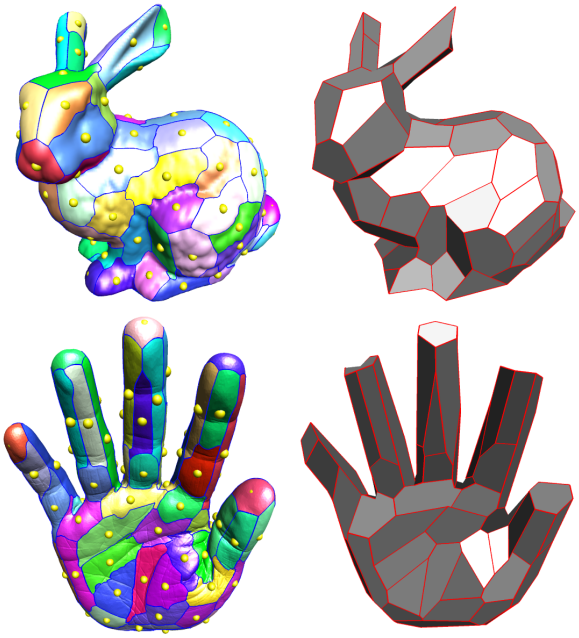

Figure 8: Left: segmentation of models "Bunny" (70k faces) and "Hand" (100k faces) with $\epsilon = 10^{-6}$ and 100 segments each. Right: extracted polygonal meshes

method for two examples. We note that this method is only applicable for simply-connected segmentations. A possible extension could be to refine the existing segmentation to make it simply-connected. We could also improve the resulting geometry by optimizing the vertex positions or introducing additional vertices similar to VSA ([7]).

5. Discussion

Our approach has interesting connections to other techniques, in the graphics community and in machine learning. We also briefly discuss generalizations and limitations.

5.1. Connection to Variational Shape Approximation

Not surprisingly, the positional variant of VSA [7] can be derived in our general setup: note that the covariance matrix $\mathbf{M}$ provides a best fitting plane (the *proxy* in the nomenclature of VSA) as the eigenvector corresponding to the smallest eigenvalue. In VSA, distances are measured only in this direction. In other words, if we set the eigenvalues of $\mathbf{M}$ to be $(1, 0, 0)$ in our approach, still using the eigenvectors $\mathbf{D}$ of the covariance matrix, distances would be measured as in VSA.

This choice of metric $\mathbf{M}$ would be the minimizer to the function F under the constraint $\|\mathbf{M}\| = 1$, where $\|\cdot\|$ is the *operator* norm, i.e. the largest singular value of $\mathbf{M}$. For symmetric $\mathbf{M}$ this constrains the first eigenvalue to be one, and in order to minimize squared weighted distances, the obvious choice for the remaining eigenvalues is zero. We note that this argument extends to any Ky-Fan norm (sum of first k singular values) on $\mathbf{M}$.

As is mentioned by Cohen-Steiner et al. [7], this choice of metric is generally unstable in the optimization. We have made similar observations when testing the idea of constraining the operator norm.

5.2. Relation to Gaussian mixture models and expectation maximization

The algorithm described in Section 3 follows naturally as a descent algorithm for minimizing Eq. 8 and could also be interpreted as a generalization of Lloyd's algorithm [5], incorporating the update of the local metric.

However, the resulting algorithm also exhibits a striking resemblance to fitting a Gaussian mixture model to given data using expectation maximization (for an introduction to these topics see the book by Bishop [8]). The main differences are in a continuous vs. discrete setting, the hard vs. soft assignment of the data to the sites, and our treatment of degenerate data. The first two distinctions seem to separate a large volume of literature on centroidal Voronoi tessellations on the one hand and the machine learning literature on EM on the other hand. It appears to be fruitful to take a more global viewpoint and unite the two approaches.

The problem of degeneracy also appears in the literature on Gaussian mixture models and expectation maximization. Here, degenerate data results in the function to be maximized being unbounded. However, the soft assignment of data to sites often reduces this problem in practice. Apart form this advantage, we also observed in preliminary tests that soft assignments lead to globally better minimization results, without the need for farthest point sampling.

On the level of justification for the approaches, we provide an argument for using inverse covariance weighting of the data: we *derive* this metric as the minimizer of a representation energy of the data (see Section 3.1); whereas considering the covariance of the data appears to be *postulated as being natural* in Gaussian mixture models (or Mahalanobis distance, for that matter).

5.3. Generalizations

As presented in Section 2, there are many variants for the computation of the local centroid and the distance metric. It would be easy to incorporate local tensor information in our approach. In a sense, we already do so by considering the centroid and covariance of each triangle.

Using anything more complex than squared weighted distances is more cumbersome: while the computation of local centroids is known [3], it is not immediately clear how to compute the metric $\mathbf{M}$ that minimizes distances under L_p norms.

The surface need not be given as a triangulation—as long as there is some notion of neighborhood the algorithm works as we described it. This could be a set of points endowed with a k-nearest neighbor graph, or depth images using the natural connectivity of the points in the image plane in smooth regions.

5.4. Limitations

The standard centroidal Voronoi tessellation can be effectively computed using Newton or quasi-Newton techniques [20, 19]. It is not clear if the derivatives necessary for such an approach could be derived for our formulation.

A potential problem is the complicated shape of the cells, which could lead to undesirable shapes of the segments on the surface. Concretely, the cell complex defined by the cells might not have the same topology as the underlying surface. This might also have implications if the dual of the cell complex should be extracted to generate a triangulation. However, these issues have already been discussed in detail [1].

We note that due to the re-definition of the convariances and the metric tensors (Section 3.1) the functional F may not decrease in every optimization step. Although we did not observed any convergence issues in our experiments we will address this fact in future work.

Acknowledgements

This work has been supported by the ERC through grant ERC-2010-StG 259550 (XSHAPE). We would like to thank the anonymous reviewers for their comments and feedback.

References

[1] F. Labelle, J. R. Shewchuk, Anisotropic voronoi diagrams and guaranteed-quality anisotropic mesh generation, in: Proceedings of the Nineteenth Annual Symposium on Computational

Geometry, SCG '03, ACM, New York, NY, USA, 2003, pp. 191–200. doi:10.1145/777792.777822.
URL http://doi.acm.org/10.1145/777792.777822

[2] Q. Du, D. Wang, Anisotropic centroidal voronoi tessellations and their applications, SIAM J. Sci. Comput. 26 (3) (2005) 737–761. doi:10.1137/S1064827503428527.
URL http://dx.doi.org/10.1137/S1064827503428527

[3] B. Lévy, Y. Liu, Lp centroidal voronoi tessellation and its applications, ACM Trans. Graph. 29 (4) (2010) 119:1–119:11. doi:10.1145/1778765.1778856.
URL http://doi.acm.org/10.1145/1778765.1778856

[4] P. S. Heckbert, M. Garland, Optimal triangulation and quadric-based surface simplification, Computational Geometry 14 (13) (1999) 49 – 65. doi:http://dx.doi.org/10.1016/S0925-7721(99)00030-9.
URL http://www.sciencedirect.com/science/article/pii/S0925772199000309

[5] S. Lloyd, Least squares quantization in pcm, IEEE Trans. Inf. Theor. 28 (2) (2006) 129–137. doi:10.1109/TIT.1982.1056489.
URL http://dx.doi.org/10.1109/TIT.1982.1056489

[6] P. C. Mahalanobis, On the generalized distance in statistics, Proceedings of the National Institute of Sciences (Calcutta) 2 (1936) 49–55.
URL http://ci.nii.ac.jp/naid/10004710165/en/

[7] D. Cohen-Steiner, P. Alliez, M. Desbrun, Variational shape approximation, ACM Trans. Graph. 23 (3) (2004) 905–914. doi:10.1145/1015706.1015817.
URL http://doi.acm.org/10.1145/1015706.1015817

[8] C. M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[9] F.-E. Wolter, Distance function and cut loci on a complete riemannian manifold, Archiv der Mathematik 32 (1) (1979) 92–96.

[10] B. Levy, N. Bonneel, Variational anisotropic surface meshing with voronoi parallel linear enumeration, in: Proceedings of the 21st International Meshing Roundtable (IMR'12), 2012.

[11] A. Secord, Weighted voronoi stippling, in: Proceedings of the 2Nd International Symposium on Non-photorealistic Animation and Rendering, NPAR '02, ACM, New York, NY, USA, 2002, pp. 37–43. doi:10.1145/508530.508537.
URL http://doi.acm.org/10.1145/508530.508537

[12] F. Aurenhammer, Power diagrams: properties, algorithms and applications, SIAM Journal on Computing 16 (1) (1987) 78–96.

[13] M. Balzer, T. Schlömer, O. Deussen, Capacity-constrained point distributions: A variant of lloyd's method, ACM Trans. Graph. 28 (3) (2009) 86:1–86:8. doi:10.1145/1531326.1531392.
URL http://doi.acm.org/10.1145/1531326.1531392

[14] M. Alexa, W. Matusik, Images from self-occlusion, in: Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging, CAe '11, ACM, New York, NY, USA, 2011, pp. 17–24. doi:10.1145/2030441.2030445.
URL http://doi.acm.org/10.1145/2030441.2030445

[15] K. B. Petersen, M. S. Pedersen, The matrix cookbook, version 20121115 (nov 2012).
URL http://www2.imm.dtu.dk/pubdb/p.php?3274

[16] S. Valette, J. M. Chassery, R. Prost, Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams, IEEE Transactions on Visualization and Computer Graphics 14 (2) (2008) 369–381. doi:10.1109/TVCG.2007.70430.
URL http://dx.doi.org/10.1109/TVCG.2007.70430

[17] J. L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (9) (1975) 509–517. doi:10.1145/361002.361007.
URL http://doi.acm.org/10.1145/361002.361007

[18] J. H. Friedman, J. L. Bentley, R. A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Trans. Math. Softw. 3 (3) (1977) 209–226. doi:10.1145/355744.355745.
URL http://doi.acm.org/10.1145/355744.355745

[19] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, C. Yang, On centroidal voronoi tessellation – energy smoothness and fast computation, ACM Trans. Graph. 28 (4) (2009) 101:1–101:17. doi:10.1145/1559755.1559758.
URL http://doi.acm.org/10.1145/1559755.1559758

[20] J. Nocedal, Updating quasi-newton matrices with limited storage, Mathematics of computation 35 (151) (1980) 773–782. doi:10.1090/S0025-5718-1980-0572855-7.

Appendix A. Covariance of a triangle mesh

To minimize the metric of a cell we need to compute the covariance matrix of a triangulated surface. This can be done by adding up the covariance matrices of individual triangles. A triangle T with vertices $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^3$ and area A has the covariance matrix

$$\mathbf{C}(T) = \frac{A}{12}(\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2)\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}(\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2)^{\mathsf{T}}. \quad \text{(A.1)}$$