Eurographics Symposium on Geometry Processing (2007)
Alexander Belyaev, Michael Garland (Editors)

Voronoi-based Variational Reconstruction of Unoriented Point Sets

P. Alliez[1] D. Cohen-Steiner[1] Y. Tong[2] M. Desbrun[2]

[1] INRIA Sophia-Antipolis [2] Caltech

Abstract

We introduce an algorithm for reconstructing watertight surfaces from unoriented point sets. Using the Voronoi diagram of the input point set, we deduce a tensor field whose principal axes and eccentricities locally represent respectively the most likely direction of the normal to the surface, and the confidence in this direction estimation. An implicit function is then computed by solving a generalized eigenvalue problem such that its gradient is most aligned with the principal axes of the tensor field, providing a best-fitting isosurface reconstruction. Our approach possesses a number of distinguishing features. In particular, the implicit function optimization provides resilience to noise, adjustable fitting to the data, and controllable smoothness of the reconstructed surface. Finally, the use of simplicial meshes (possibly restricted to a thin crust around the input data) and (an)isotropic Laplace operators renders the numerical treatment simple and robust.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [CG]: Computational Geometry and Object Modeling.

1 Introduction

Surface reconstruction from point clouds is motivated by a number of CAGD, point-based graphics, and reverse engineering applications where scattered point samples of a surface need to be turned into a proper, watertight surface mesh. Particularly challenging are point sets generated by laser scanners and hand-held digitizers, as they are often noisy (due to the inherent uncertainty of measurements), unorganized (due to the merging of several scans), and possibly containing large holes (due to occlusions during the acquisition process). In such a context, surface reconstruction can only be approximating—instead of interpolating—as data points are more of an indication of proximity to the surface than actual sample points.

While a number of algorithms can now efficiently reconstruct oriented points (*i.e.*, point sets where a normal is provided at each sample), fewer methods are able to *approximate raw (unoriented) point sets, with controllable smoothness*. In this paper, we introduce a Voronoi-based variational approach to surface reconstruction that can equally process unoriented point sets *and* oriented point sets, and can even exploit confidence measures on the normals if available. As our technique is based on isocontouring of a variationally-defined implicit function defined in the embedding space, we guarantee watertightness and smoothness while offering control over data fitting.

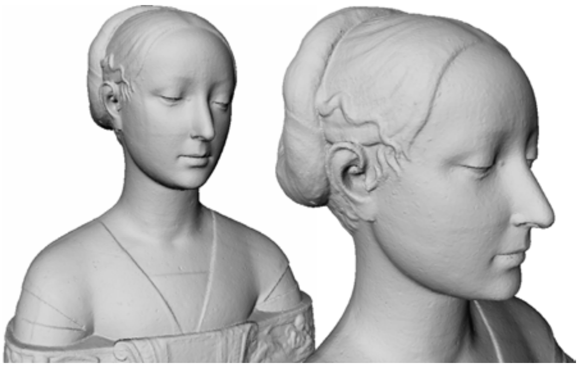
Figure 1: ***Sforza.*** *Our variational method allows high-fidelity reconstruction of unprocessed point sets (3D scanned* Sforza*, 200K points). The final mesh is extracted via isocontouring of a scalar function computed through optimization.*

1.1 Related Work

Delaunay-based surface reconstruction techniques were initially designed to establish a plausible connectivity between points [Boi84]. One of the first reconstruction techniques that came with theoretical guarantees was proposed by Amenta and Bern [AB99]. The rationale behind their technique was that when a sampling is noise-free and dense enough, all

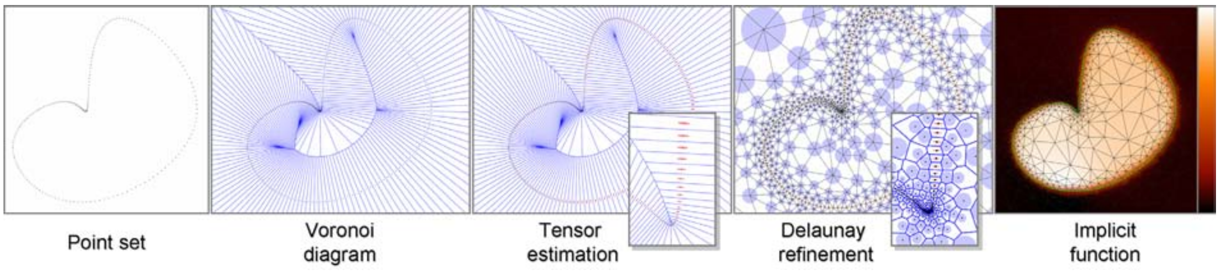

Figure 2: ***Our reconstruction procedure at a glance.*** *From left to right: input point set; its Voronoi diagram; covariance matrices of the cells shown as (rescaled) ellipses; Steiner points added through Delaunay refinement (isotropic tensors are assigned to Steiner points); piecewise linear function f (solution of a generalized eigenvalue problem) that best fits the input data, with the reconstructed curve (isocontouring of f).*

Voronoi cells are elongated in the direction of the normal to the inferred surface. An analysis of the point set's Voronoi diagram can then be used to derive an interpolating reconstructed surface. This technique has stimulated many improvements and variants: we refer the reader to [CG06] for a survey, and to [Dey06] for a recent comprehensive monograph. In practice however, most of these Voronoi-based techniques are interpolatory, thus inadequate in the presence of noise.

Noise and sparseness in typical point sets have led to an *approximating* class of approaches, where an implicit function is computed such that one of its isosurfaces best fits the data (see *e.g.*, [OBA*03]). These implicit approaches are particularly convenient as they all guarantee a watertight 2-manifold surface approximation by construction; they mostly differ in the norm used to match the point sets and the isosurface. Approximations of a signed distance function to the point set [CBC*01] or to estimated tangent planes [HDD*92, BC02] were proposed as possible implicit functions. More recently, a reconstruction method for oriented point sets was introduced in which an implicit function f is derived from a Poisson equation $\Delta f = div(\mathbf{n})$, providing the best L^2-match between the gradient of f and the input normals $\mathbf{n}$ [KBH06]. In practice, the normal field is first interpolated onto an octree via an isotropic smoothing kernel, offering control of the smoothness of the reconstructed surface. This algorithm scales remarkably well since it only involves solving a linear system, and its global L^2-minimization nature makes it resilient to noise. These techniques require a *consistent* orientation of the normals to perform correctly. Unfortunately, unless reliable normals are provided, finding such an orientation has been recognized to be an ill-posed problem when the sampling is sparse and noisy. In particular, one of the most robust approaches to normal orientation is through labeling a set of Voronoi poles; but it requires little or no noise and a dense-enough (ε-)sampling to guarantee consistent results—two conditions rarely met in practice.

Given this intrinsic difficulty, recent work has polarized on handling raw, unoriented point sets without attempting to locally estimate or orient the normals (*e.g.*, [KSO04, WCS05, HK06,PSQ06]). Initial work includes methods such as Tensor Voting [MM06], where local shape is inferred from neighboring sample points even in the presence of noise. More recently, a spectral reconstruction method [KSO04] has been shown remarkably robust to outliers due to its reliance on graph partitioning algorithms [SM00]. However, this method is interpolatory, requiring a systematic post-smoothing for noisy point clouds. Similarly, Hornung and Kobbelt [HK06] propose a min-cut algorithm on a regular grid, also requiring post-smoothing to remove grid aliasing. Finally, an approach based on eigen analysis but derived purely from an optimization standpoint, offers an approximating reconstruction where *smoothness* can be controlled [WCS05]—although various coefficients require adjustment to provide good results.

1.2 Overview

Our approach combines the generality and resilience of spectral methods as it can take a raw, unoriented and noisy point set as input. It also offers the quality of Poisson-based reconstruction through global fitting of normal direction evaluations and control over smoothness. Our algorithm performs equally well when normal information is included in the point set, even exploiting confidence measurements on the normals to improve the quality of the reconstructed surface. To provide a unified, variational framework for point set surface reconstruction, our algorithm proceeds in two main steps. First, if no normal information is provided, we perform a novel, Voronoi-PCA estimation of unoriented normals induced by the point set (Section 2). This first step results in a tensor field which encodes both the (unoriented) normal direction (through its eigenvector of maximum eigenvalue) *and* the confidence in the approximation (through its anisotropy). Second, an implicit function is computed via a generalized eigenvalue problem (Section 3) so as to make its gradient best fit the normal directions. Details of the algorithm and its implementation are given in Section 4, and a series of tests and comparisons are presented in Section 5.

2 Estimating Unoriented Normals

Our first goal is to estimate unoriented normals (along with their reliability) to the inferred surface from the input point set if no such information is available. Note that we will not try to infer an orientation, as this global task will be incumbent upon the second step of our approach. As a way of motivating our estimation, we briefly review a few closely related techniques.

2.1 Background

Normal estimation from point sets has received a lot of attention in the past few years (see *e.g.*, [PKKG03, MNG04, DLS05, OF05, LP05, HXMP05]). Various strategies have been proposed, guaranteeing high-order accuracies through, *e.g.*, local fitting [CP03]. However, when a point set is noisy and unstructured, a majority of methods have recourse to one of two popular techniques, both straightforward to implement.

Principal Component Analysis (PCA) A conventional technique for estimating normal directions is through *local* principal component analysis. The main idea is to define a small neighborhood around each input point (*e.g.*, the k-nearest neighbors [HDD*92]), compute the covariance matrix of the points in this neighborhood, and deduce the normal direction from the eigenvector associated to the smallest eigenvalue of the resulting covariance matrix. Many variants have been proposed to improve resilience to noise (see, for instance, [MNG04]).

Voronoi Poles Another common technique for estimating normal directions is *global* by nature as it requires the construction of the Voronoi diagram of the input point set. A subset of Voronoi vertices called *poles* [AB99] is then extracted, and used to estimate a normal direction at each sample point. In absence of noise and for dense-enough samples this method can be shown to provide a faithful normal estimate even for irregular sampling, with convergence rates depending on the elongation of the Voronoi cells. A variant by Dey and Sun [DS05] provides more resilience to noise.

2.2 A Voronoi-PCA Approach to Normal Estimation

Given the respective advantages of PCA and Voronoi poles, we propose a novel normal approximation technique that combines both of their qualities. We begin by computing the 3D Voronoi diagram of the input point set after adding dummy sample points on a very large bounding sphere (the resulting Voronoi cells of the input points are therefore not extending to infinity).

As the shape of these cells reflects the global distribution of the points in space, a key observation is that *the covariance matrix of the Voronoi cell of a sample point provides not only an estimate of the normal direction, but also a measure of how reliable this estimate is*. Indeed the eigenvector associated to the largest eigenvalue indicates the axis along which the cell is elongated (see inset)—a good approximate of the normal direction if the samples all lie on a common manifold. Moreover, as described in [AB99], the *confidence* in the estimate is related to how long and thin the Voronoi cell is, *i.e.*, to the anisotropy of the covariance tensor.

Covariance Matrix of a Voronoi Cell Let V be a finite Voronoi cell associated to a sample point p. The covariance matrix of V is defined by its order-2 moment with respect to its center of mass m:

$$\mathrm{cov}(V) = \int_V (X-m)(X-m)^t \, dX. \tag{1}$$

A simple way to compute this integral is to perform a decomposition of the Voronoi cell V into tetrahedra, so as to assemble the final covariance matrix in closed form as explained in Appendix A. If the sampling is of good quality, this procedure will be very accurate as each Voronoi cell is long and skinny. However, as Fig. 3(middle) illustrates, Voronoi cells can become small and isotropic if noise is present.

Covariance Matrix of a Union of Voronoi Cells To render our estimate robust to noisy point sets, we compute the covariance matrix of a *union* of Voronoi cells (closed-form formulas are also provided in Appendix A): as the Voronoi diagram of the point set partitions the whole domain, elongated cells *are* present beyond the noisy area, and accumulating enough neighbors will eventually render the union elongated enough (see Fig 3(right)). Notice that the idea of combining the influence from neighbors to promote noise resilience is commonplace, but our technique is adaptive in the sense that we use as many neighbors as needed to find a reliable approximation as described next.

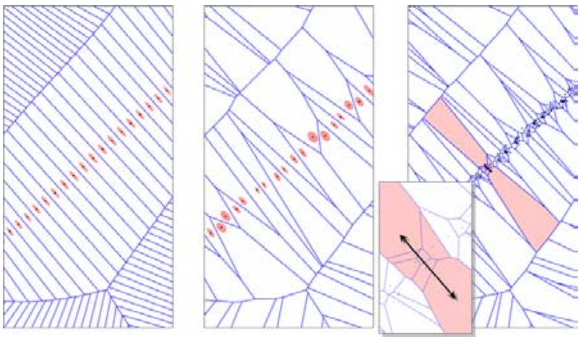

Figure 3: ***Voronoi cells of point sets.*** *Left: Voronoi diagram of a point set without noise. Middle: with noise the cells become irregular. Right: a denser, noisy point set shows even more diverse cell shapes.*

Estimation Procedure We implement our estimation procedure as follows: given a sample point p, we first compute the covariance matrix of its Voronoi cell $V(p)$, and measure its anisotropy $\sigma \in [0,1]$ as $\sigma = 1 - \mathrm{isotropy}(V)$, where $\mathrm{isotropy}(V)$ is the ratio between the smallest and the largest eigenvalues. We then iterate over its k nearest neighbor points $\{q_i\}$, and perform the same procedure for $V(p) \cup V(q_1)$, then for $V(p) \cup V(q_1) \cup V(q_2)$, etc., until either the anisotropy has reached a threshold of 0.9, *or* we reach the maximum number of nearest neighbors (typically $k = 50$). From these covariance matrices, the one with *maximum anisotropy* is then returned. Finally, for normalization purposes, we rescale this resulting tensor such that its maximum eigenvalue is unit. Notice that this evaluation procedure will stop at a single Voronoi cell when the sampling is dense and noise-free: there is indeed no incentive to add an extra neighborhood Voronoi cell as it would mainly thicken the domain of integration in Equ. (1), thus decreasing anisotropy.

This procedure benefits from both the qualities of PCA (local analysis of a number of neighboring samples) and those of the Voronoi-based approach (global analysis of the sample repartition via the Voronoi diagram). Our Voronoi-PCA normal estimation technique can be seen as an integral PCA approach, similar to [PWY*06] for curvature estimation: integration (over Voronoi cells in our case) leads to more stable estimations compared to the pole approach which bases its estimation on the position of a single pole. As the quality of this normal estimation is important to our reconstruction procedure, we will present in Section 5.1 numerical experiments showing that this procedure outperforms the two normal estimation techniques that it combines.

3 Generalized Eigenvalue Problem

We consider at this stage that a (covariant) tensor field encoding the normal directions has been either estimated through our Voronoi-PCA approach, or derived from normal information in the input data (as $\mathbf{nn}^t$). We now wish to compute an implicit function such that its gradient is best aligned with the principal axes of the tensor field: isosurfacing this scalar function will provide a best-fitting reconstruction.

3.1 Problem formulation

While a Poisson reconstruction [KBH06] can directly give us an implicit function such that its gradient best fits a normal field, we cannot have recourse to such a direct linear solve as we only have a direction field (*i.e.*, unoriented normals) through our Voronoi-PCA tensors. We need instead to find an implicit function f such that its gradient ∇f is everywhere best aligned to the the maximum eigenvalue direction of the tensor field C, where the notion of "best alignment" is weighted by the local confidence in the normal direction. We propose the following constrained maximization procedure to efficiently find such a function f:

Given a tensor field C, find the *maximizer* f of:

$$E_C^D(f) = \int_\Omega \nabla f^t \; C \; \nabla f \text{ subject to: } \int_\Omega \left[|\Delta f|^2 + \varepsilon |f|^2\right] = 1,$$

where Ω is the domain, and Δ is the Laplacian operator.

The interpretation of this optimization problem is as follows. The energy function E_C^D, called *anisotropic Dirichlet energy*, directly measures the alignment of ∇f with the normal direction indicated by C. Indeed, an isotropic tensor (*i.e.*, unknown normal direction) has little influence on this energy, whereas an anisotropic tensor (*i.e.*, high confidence in the normal direction) will play a major role—penalizing misalignment at reliable data point. We then add as a constraint that E_C^D must be maximized *over the unit ball defined by the biharmonic energy*. Just like the Dirichlet (harmonic) energy is a measure of the smoothness f, the biharmonic energy measures the smoothness of ∇f: therefore, this added constraint imposes a regularization of the maximizer f. A small amount of the L^2 norm of f is added to avoid having to constraint values of f (either on the boundary or inside the domain), as well as to improve conditioning: the resulting constraint is a Sobolev-like norm E^B on f. In practice, however, we replace this regularization term by a small amount of data fitting as we will discuss in Section 3.3.

Solving for this constrained maximization amounts to carefully balance smoothness of ∇f vs. alignment of the gradient: it will align ∇f to C if it is particularly rewarding for the anisotropic Dirichlet energy (*i.e.*, when the normal direction is particular reliable); on areas where the tensor is isotropic, the solver will favor smoothness of the function gradient instead. This global balancing act implicitly induces a consistent orientation to the tensor field since flipping the sign of ∇f between two incident vertices of the simplicial mesh significantly increases the biharmonic energy.

3.2 Discrete Formulation

We now assume that we have a tetrahedral mesh of the 3D domain with V vertices $\{v_i\}$ and E edges $\{e_i\}$. Each edge e_i is arbitrarily oriented. Given this mesh with a tensor C_i at each vertex i, we wish to solve for a node-based, piecewise linear function f, *i.e.*, to find a vector $F = (f_1, f_2, \ldots, f_V)^t$ that satisfies the aforementioned constrained maximization.

Discrete Anisotropic Dirichlet Energy The energies involved in the optimization are rather simple to express. Although various expressions of the anisotropic Dirichlet energy have been proposed in 2D in the context of quasi-harmonic parameterizations [Gus02, ZRS05], we construct our 3D energies via matrix assembly using the language of discrete forms [DKT06]. Thus, $E_C^D(F)$ is expressed as:

$$E_C^D(F) \approx F^t \, A \, F \quad \text{with} \quad A = d_0^t \star_C^1 d_0$$

where d_0 is the transpose of the signed vertex/edge incidence matrix (size ExV) of the mesh, and $\star_C^1$ is the Hodge star operator for the metric induced by C. We approximate this latter operator by the Euclidean diagonal Hodge star $\star^1$ *modulated* by the tensor C, resulting in the following diagonal matrix:

$$\forall i = 1 \ldots E, \; (\star_C^1)_{ii} = \frac{e_i^t C \, e_i}{e_i^t \, e_i} \; (\star^1)_{ii} \qquad \text{with:} \quad (\star^1)_{ii} = \frac{|e_i^*|}{|e_i|},$$

where e_i is the i^{th} (oriented) edge, $|e_i|$ is its length, and $|e_i^*|$ is the area of its dual Voronoi face. The value of the tensor C on an edge is found by averaging the tensor values at each vertex of the edge.

Discrete Biharmonic Energy For the biharmonic energy, the following (simplified) discretization performs adequately in practice (regularization will be added later through a data fitting term):

$$E^B(f) \approx F^t \; B \; F \quad \text{with} \quad B = (d_0^t \star^1 d_0)^2$$

where we used the same notations as above.

Optimization Procedure Using a Lagrange multiplier λ, we can now rewrite the constrained optimization as a maximization of the following functional:

$$E = F^t A F + \lambda (1 - F^t B F).$$

A necessary condition of optimality is $\partial E / \partial F = 0$, yielding:

$$AF = \lambda BF.$$

This expression expresses what is known as a *generalized eigenvalue problem* (GEP), where F is an eigenvector of this GEP, and λ is its corresponding eigenvalue. In fact, the solution to our constrained maximization is *the eigenvector of the GEP corresponding to the largest eigenvalue*.

Proof: Since the eigenvectors of a GEP form a basis, we can write a function F as:

$$F = \sum a_\lambda F_\lambda,$$

where F_λ is the eigenvector corresponding to the eigenvalue λ (i.e., $AF_\lambda = \lambda BF_\lambda$) such that $F_\lambda^t BF_\lambda = 1$ and, for $\lambda_1 \neq \lambda_2$, $F_{\lambda_1}^t BF_{\lambda_2} = 0$. Therefore:

$$E_C^D = (\sum_\lambda a_\lambda F_\lambda)^t B(\sum_\lambda \lambda a_\lambda F_\lambda) = \sum_\lambda \lambda a_\lambda^2 \leq \sum_\lambda \lambda_{\max} a_\lambda^2 = \lambda_{\max}.$$

Since the energy is bounded by the max eigenvalue $\lambda_{\max}$ and this value is attained by $E_C^D(F_{\lambda_{\max}})$, we get: $F = F_{\lambda_{\max}}$.$\square$

3.3 Generalizations

So far, our Voronoi-based variational approach requires no parameter-tweaking to provide a reconstruction of a point set. We can, however, easily extend this basic approach while keeping the exact same framework. In particular, the matrix B that contains the discrete biLaplacian can be modified in various ways to allow for more control over smoothness and/or interpolation:

- *Data fitting:* we can change the optimization results by adding a term that controls data fitting. We favor a value of 0 on the input points by adding to the constraint a fitting factor times the sum of the squares of the function values *at input points*. Changing the fitting factor will provide a controllable data fitting effect to our procedure. Variants, *e.g.*, where data fitting is spatially varying, are easily designed.
- *Splines-under-tension energy:* instead of only using the biLaplacian, we can constrain the optimization over a unit ball defined by a *linear combination* of the Dirichlet and the biharmonic energies. It will allow for a better tradeoff between smoothness of the results vs. fitting of the normal directions, as it is tantamount to a splines-under-tension energy [SW90].

The matrix B needs to be slightly modified in order to implement these two generalizations as we will detail in Section 4.3. While we only experimented with these two examples (and we will detail their implementation next), other modifications of A and/or B are possible, yet indubitably application-dependent.

4 Implementation

Our technique was implemented in C++. We now describe the algorithm along with the implementation choices we made for each of its steps.

4.1 Voronoi-PCA Estimation

As our algorithm has been implemented using the CGAL library [CGAL03], this step is straightforward. After reading in the input point set and adding a few dummy points on a large bounding sphere, we compute its Voronoi diagram. We then use the iterative procedure described in Section 2 to find a tensor per input point, using the formulas given in Appendix A to optimize the covariance matrix computations.

4.2 Delaunay Refinement

Before performing the constraint optimization, we must refine and improve the Delaunay tetrahedral mesh (dual of the Voronoi diagram previously computed) of our domain: since the point set can be arbitrary, any numerical approximation evaluated on the initial mesh would inevitably lead to numerical degeneracies due to the poor aspect ratio of tets. We thus have recourse to a Delaunay refinement procedure [RY06]: this simple priority-queue-based treatment iteratively improves tetrahedra with radius edge ratio greater than a threshold (set to 2 in all our tests) by adding Steiner points. At each Steiner point inserted to the triangulation, we set the tensor C_i to the 3x3 identity matrix to remain agnostic as to the normal direction there. In order to limit the size of the final mesh, we restrict the refinement to an enlarged bounding box of the input point set. Optionally, we can also refine the mesh within a *thin shell* around the input data by enforcing a maximum edge length as well, providing more degrees of freedom to the implicit function where it matters. Figure 4 illustrates this process in 2D and 3D. Note that this refinement procedure is akin to the use of a restricted octree in, *e.g.*, [KBH06]. We point out however that we do, instead, keep the original points, preventing any local "oversmoothing" that appears in methods where the input point set is discarded in favor of a substitute (regular or adaptive) grid.

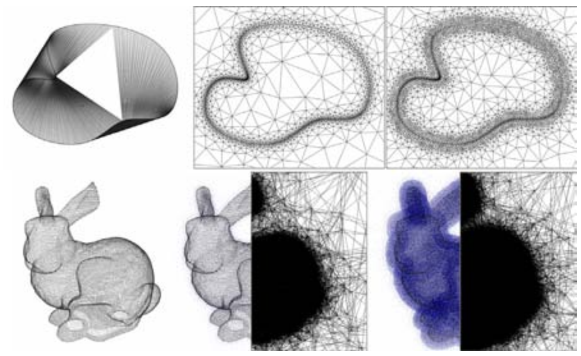
Figure 4: ***Delaunay refinement.*** *(top) 2D Delaunay triangulation (left) of a point set sampling a curve is iteratively refined by adding Steiner points with (right) or without (middle) sizing constraints within a shell around input points; (bottom) Example in 3D.*

4.3 Solver

We solve the generalized eigenvalue problem $AF = \lambda BF$ by turning it into a classical eigenvalue problem. We first precompute a Cholesky factorization of B using the TAUCS library [TCR05]; this results in a lower triangular matrix L such that $B = LL^t$. We now rewrite the GEP as:

$$AF = \lambda LL^t F \Leftrightarrow L^{-1}AL^{-t}L^t F = \lambda L^t F \Leftrightarrow \begin{cases} L^{-1}AL^{-t}G = \lambda G \\ G = L^t F \end{cases}$$

We then employ the implicitly restarted Arnoldi iteration method from the ARPACK++ library [GS], with $L^{-1}AL^{-t}$ as the Arnoldi operator and by requesting the maximum eigenvalue only. After convergence, we find the solution F by solving $L^t F = G$, as it corresponds to the eigenvector corresponding to the maximum eigenvalue of the original GEP. This eigenvector defines a piecewise linear function over the tetrahedral mesh: we are now ready for contouring.

Additional Terms Instead of only using the bilaplacian, we also add to B a data fitting matrix (denoted D) times a parameter μ_{fit}, as discussed in Section 3.3. The matrix D is a simple diagonal matrix with 1s only on the rows corresponding to input points (*i.e.*, for all points but the Steiner points added by Delaunay refinement). The parameter μ_{fit} not only allows us to tune the amount of data fitting, but also to tune the separation between connected components (see Fig. 8). Note also that we always include at least a small amount of data fitting term ($\mu_{\text{fit}} > 10^{-4}$) to efficiently substitute for the ε regularizing term in the smoothness constraint. Similarly, we add $\mu_\Delta d_0^t \star^1 d_0$ to offer more control over smoothness as described in Section 3.3. Once the user has chosen values for the coefficients μ_{fit} and μ_Δ, the solver proceeds as explained with the matrix B set to: $B = (d_0^t \star^1 d_0)^2 + \mu_{\text{fit}} D + \mu_\Delta d_0^t \star^1 d_0$.

4.4 Contouring

To find which isocontour to extract, we first evaluate the resulting function at all input points (not Steiner points), and pick the median value for contouring (outliers may affect the average; we found that the median provides a more robust isovalue). For the final isocontouring, either a simple marching-tetrahedra algorithm or a Delaunay-based surface meshing algorithm [BO05] can be used. We prefer the latter as it generates meshes with fewer elements, and with guaranteed *quality* of the mesh elements (all triangle angles are between 20 and 140 degrees). Figure 9 illustrates outputs of the Delaunay-based meshing algorithm with three uniform sizing criteria.

5 Results

Before showing results of our reconstruction technique, we present some numerical experiments on our Voronoi-PCA method.

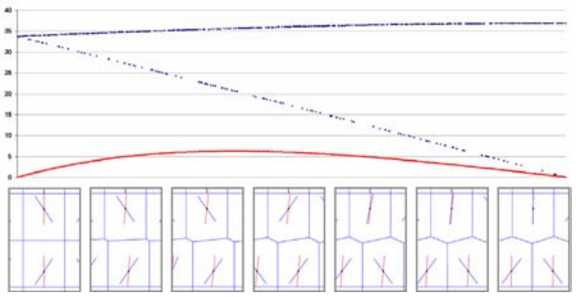

Figure 5: ***Poles vs. covariance matrices in 2D.*** *A point set samples two parallel lines; (bottom) from left to right, we translate the samples of the bottom line slowly; pole-based normal estimates are depicted in blue, while our covariance-based normal estimates (deriving from the Voronoi cells displayed as well) are in red; (top) the two curves compare the pole-based results (blue, very discontinuous) with the covariance-based estimate (red) using the angle error (in degrees) as the bottom line of points is shifted.*

5.1 Voronoi-PCA Estimation

We first illustrate how our integral-based normal estimation technique provides improved numerics compared to the usual pole-based approach by examining a very simple point set configuration, where 2D points are sampled uniformly along two parallel lines. When the bottom line is slowly shifted, the shape of the Voronoi cells evolves, triggering (often discontinuous) changes in the pole-based normal estimates. Because our approach relies on the *integrated* moment of the cells, it is significantly less sensitive to the shift as shown by the curve in Fig. 5. Notice also that while a k-nearest neighbor PCA approximation can dramatically lose accuracy when the (Euclidean-based) neighbors include points of both lines, our approach benefits from the global nature of the Voronoi diagram, making it more robust to sparse sampling.

The next experiment compares our normal estimation technique for 3D parametric surfaces with both pole and point-based PCA approaches. We sample a height field $z = sin(x)cos(y)$ (for which normals are known analytically) with different sampling criteria:

- **Noise free:** The height field is first sampled on a regular grid in parameter space in the interval $[-\pi, \pi]^2$, with rates ranging from 20×20 samples to 100×100 samples.
- **Noise in parameter space.** The height field is then sampled on a jittered grid in parameter space, with the same various densities as above. The noise is uniform and isotropic, with a maximum magnitude of half the grid spacing to make it scale with sampling density.
- **Noise in embedding space.** The samples of the first case (regular grid) are now jittered in the embedding space using an isotropic uniform noise of half the grid spacing.

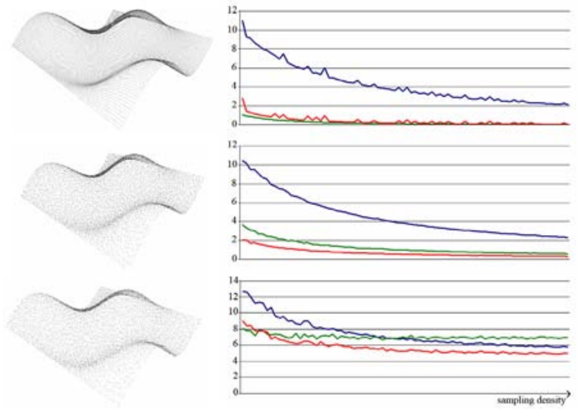

Figure 6: ***Normal estimation on a parametric surface.*** *(top) noise-free sampling, (middle) noise added in parameter space, (bottom) noise added in embedding space. The plots show the average angle deviation in degrees as a function of the sampling density. Pole-based estimation in blue, point-based PCA in green, and our covariance-based estimation with a single Voronoi cell in red.*

In these three contexts, we measure the average angle deviation between normal estimates and true normals for each sample density. For the point-based PCA technique, we always use the 8 nearest neighbors as it leads to the best estimates. As Fig. 6 indicates, our approach is, across all tests, either as good as or better than the two normal estimation techniques that it is built upon. Comparing various normal estimation techniques fairly is notoriously difficult as many parameters come into play, such as sampling density, sampling anisotropy, noise, and outliers—so our tests are not intended to be extensive and conclusive. Nevertheless, we conducted several other experiments (including a comparison with [MNG04]) that all show the numerical relevance of using a Voronoi-based covariance approach. A theoretical analysis of our normal estimation technique, which should draw upon Voronoi diagrams, principal component analysis, and possibly geometric measure theory, is left to future work.

5.2 Surface Reconstruction

As a proof-of-concept example, we begin with a 2D experiment to exemplify the similarities and differences between our approach and the Poisson-based method performed on the same simplicial mesh (*i.e.*, we implemented the Poisson equation of [KBH06] on a triangle mesh using the discretization described in [TLHD03]). As Fig. 7 illustrates, the first point set (dense, noise-free) is easily oriented using pole labeling (red/blue dots are poles), which allows us to use Poisson equation for reconstruction. Both methods lead to very similar reconstructed curves. When the sampling is sparser, the pole-based orienting process fails to provide the "right" orientation, and the Poisson reconstruction reflects this error, while the optimal implicit function remains quite unchanged.

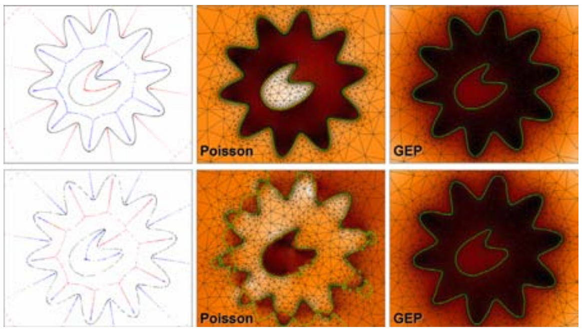

Figure 7: ***Comparison with Poisson reconstruction in 2D.*** *(top) our approach is very similar to a Poisson reconstruction for dense point sets as the normal orientations can reliably be deduced; (bottom) for sparser/noisier unoriented datasets, local orientation becomes prone to errors, while our variational approach remains valid.*

We also illustrate the effects of tuning the parameters (in 2D for clarity). μ_Δ allows controlling the smoothness (Fig. 8, top). μ_{fit} controls the fitting of the input points, and hence the separation of two components (Fig. 8, middle). It also provides a way to increase the contrast of the implicit function for nested components (Fig. 8, bottom left). A shape completion example from a sparse dataset, obtained with our splines-under-tension energy, is also shown (Fig. 8, bottom/middle). Our last 2D example illustrates (Fig. 10) the resilience to both sparsity of the point set as well as noise (including few outliers) in the data.

We also processed a number of 3D point sets issued from (laser range) scanners. A golf club head with 14K points (Fig. 11), a kitten model of genus 1 (Fig. 9), a bunny (Fig. 12), and a bust model with 250K points (Fig. 1) are presented. Additionally, we show in Fig.13 that our method applied to a raw, unoriented point set (206K points) recovers a similar surface to the Poisson reconstruction from [KBH06] for which additional normal information was provided. We note that even at octree depth 11, the Poisson-based mesh is comparatively oversmoothed, mostly due to the interpolation of the normals onto the octree leaves. If such an oversmoothing is desirable, we can similarly interpolate the tensor field around the input points, or simply increase the smoothness by tuning our two parameters (as demonstrated in Fig. 8).

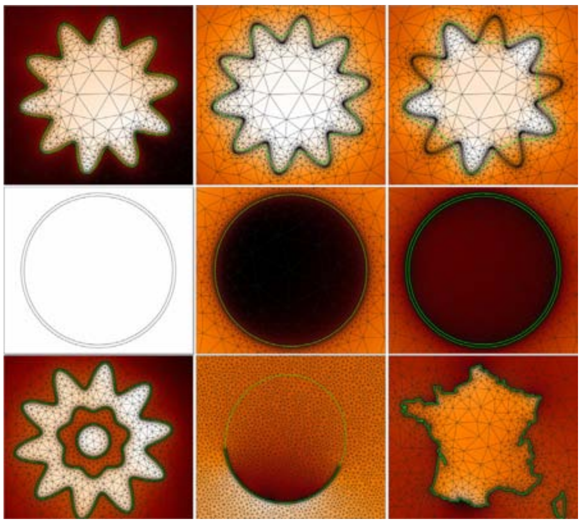

Figure 8: ***Reconstruction Parameters.*** *(top) adjusting μ_Δ (left to right) permits easy tuning of the resulting smoothness, (middle) while μ_{fit} controls the fit to the input point (middle & right), allowing to accurately separate and capture fine, nearby surface layers. (bottom) data fitting allows the reconstruction of nested components (left); smooth completion of sparse dataset is easily achieved using our splines-under-tension energy (middle); separate components of different geometric complexity can also be captured accurately.*

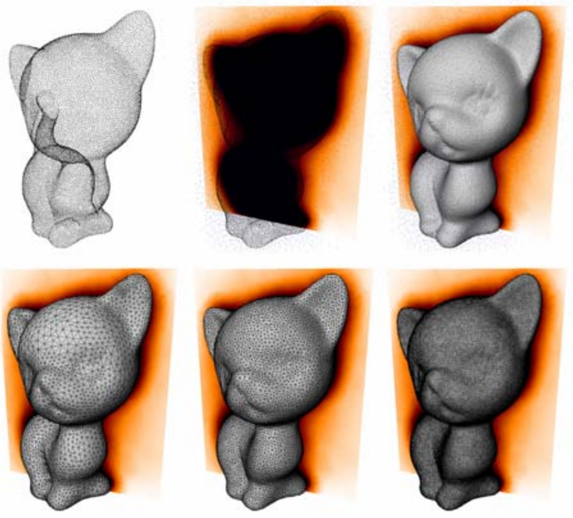

Figure 9: ***Kitten.*** *(top) 20K input point set, Steiner points and implicit function, and shaded reconstructed surface obtained by marching-tetrahedra. (bottom) three output meshes at increasing resolutions.*

Timings and Limitations On a laptop (1.8GHz, 2 GBytes) our current implementation constructs the initial Delaunay triangulation of the kitten (20K vertices, 132K tets, Fig. 9) in 1.2s, and Delaunay-refines it to 30K vertices in 6s. The matrix A contains around 8 non-zero elements per line, while B has around 35 non-zero elements per line. The Cholesky factorization takes 23s (including TAUCS super-nodal ordering). The 51 Arnoldi iterations to convergence take a total of 42s. The data structures used in our algorithm scale linearly in memory (in particular, a Delaunay tetrahedralization scales linearly in practice, and the number of non-zero elements per line in our matrices does not depend on the size of the model). The only bottleneck to a good scalability of the algorithm is the Cholesky factor: its memory requirement is high (2e+7 non-zero elements for the kitten). Out-of-core factorization is a viable option that we use for large models like *Bimba* and *Sforza*, but the overall timings consequently suffer (up to 25 minutes for 250K input points), and the super-nodal ordering still requires in-core execution. For larger datasets a 64-bit machine would be indispensable to address these memory issues.

Discussion Our approach offers a unique mix between Voronoi-based methods (to estimate normals via PCA), implicit-based approaches (as an implicit function is globally optimized to allow for smooth approximation through isocontouring), and spectral techniques (as our optimization procedure is expressible as a generalized eigenvalue problem). It empirically matches the results of oriented point set reconstruction techniques such as Poisson reconstruction, but without the need for normal orientation. We note that the closest existing approach is, from a numerical point of view, the work of [WCS05] as they too use a GEP with multiple energy terms. However, our technique differs quite drastically in its premises as, *e.g.*, they do not make use of anisotropic energies or tensors, they rely on regular grids only, and do not try to infer normal directions prior to the GEP, resulting in a delicate parameter tweaking to avoid spurious surface oscillations in the results.
Finally, a hybrid approach between our contribution and Poisson reconstruction might be most desirable for massive data: our approach could be used to orient a subsampled version of a raw, unoriented point set, thus inducing an orientation for the initial point set on which a Poisson solve can then be performed. Additionally, as confidence estimation of the normal direction might become part of the measurements provided by newer scanning devices, our technique could directly exploit this additional information. Similarly, confidence in the sample points could be incorporated through local variation of μ_{fit}.

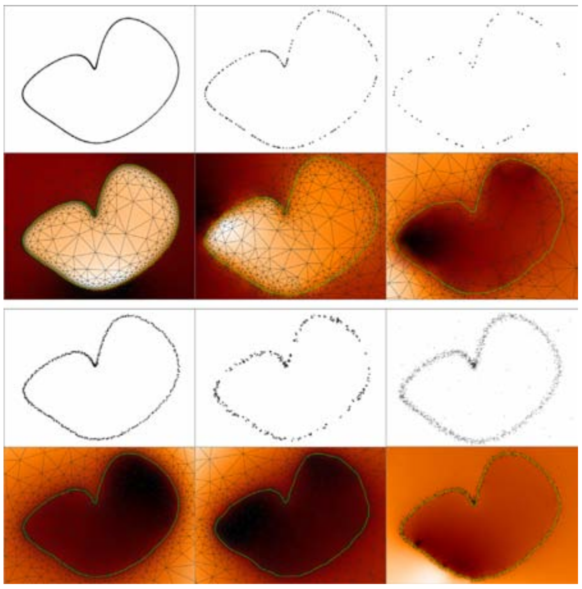
Figure 10: ***Noise and Sparsity Resilience.*** *(top) our approach can handle from dense (left) to sparse (right) raw datasets without any parameter tweaking; (bottom) similarly, results degrade gracefully with noise (left, middle), even in the case of outliers (right).*

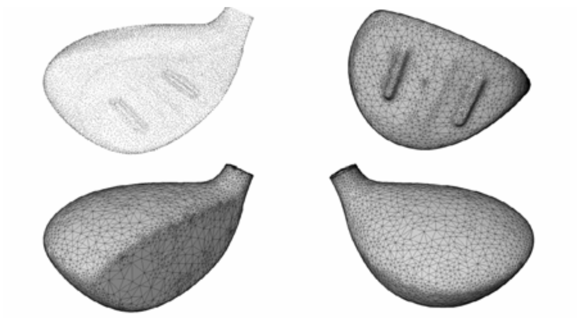
Figure 11: ***Golf club.*** *A scanned point set (14k points), and its reconstruction seen from various angles.*

6 Conclusion

We have presented an algorithm for reconstructing watertight surfaces, offering a unified approach for oriented and unoriented point sets. If no normal information is available, our technique starts by estimating both normal direction and confidence in the estimate via a novel, noise-resilient Voronoi-PCA procedure. The reconstruction then proceeds to find an implicit function that optimizes the alignment between its gradient and the estimated/provided normal directions. Distinctive features of our approach include the use of tensors in a generalized eigenvalue problem to balance fitting and smoothness based on data confidence, as well as the ability to add data fitting and smoothness control.

Future work We wish to study the theoretical properties of our normal estimation technique and its application to dimension detection (along the line of [DGGZ03]). Also, to achieve what [KBH06] described as a desirable *indicator* function, the L^2 norm used in our paper (and in Poisson reconstruction) is obviously inappropriate, as sharp transitions are overly penalized. An L^1 norm would, instead, be best in this case. However, it is not obvious that the computational overhead that such a norm requires is worth the added benefits. Finally, our approach to turn a tensor into an oriented vector field (*i.e.*, ∇f) might provide a new approach to applications such as quad remeshing, as a (curvature) tensor field could be processed along the line of our technique to provide curvature-aligned meshes without user interaction.

Acknowledgments The authors wish to thank David Bommes from RWTH Aachen for his help with ARPACK. Partial funding from the EU Network of Excellence AIM@SHAPE (IST NoE No 506766), the EU FET-Open project ACS (IST No 006413), NSF (CAREER CCR-0133983, and ITR DMS-0453145), DOE (DE-FG02-04ER25657), and Pixar.

References

[AB99] AMENTA N., BERN M. W.: Surface reconstruction by Voronoi filtering. *GEOMETRY: Discrete & Computational Geometry 22* (1999).

[BC02] BOISSONNAT, CAZALS: Smooth surface reconstruction via natural neighbour interpolation of distance functions. *CGTA: Computational Geometry: Theory and Applications 22* (2002).

[BO05] BOISSONNAT J.-D., OUDOT S.: Provably good sampling and meshing of surfaces. *Graph. Models 67*, 5 (2005), 405–451.

[Boi84] BOISSONNAT J.-D.: Geometric structures for three-dimensional shape representation. *ACM Trans. on Graphics 3*, 4 (1984), 266–286.

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *SIGGRAPH* (2001), pp. 67–76.

[CG06] CAZALS F., GIESEN J.: Delaunay triangulation based surface reconstruction. In *Effective Computational Geometry for Curves and Surfaces*, Boissonnat J., Teillaud M., (Eds.). Springer-Verlag, Math. and Visualization, 2006, pp. 231–276.

[CGAL03] Computational Geometry Algorithms Library CGAL-3.2. http://www.cgal.org/, 2003.

[CP03] CAZALS F., POUGET M.: Estimating differential quantities using polynomial fitting of osculating jets. In *Symposium on Geometry Processing* (2003), pp. 177–187.

[Dey06] DEY T. K.: *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge Monographs on Applied and Computational Mathematics, 2006.

[DGGZ03] DEY T. K., GIESEN J., GOSWAMI S., ZHAO W.: Shape dimension and approximation from samples. *Discrete & Computational Geometry 29*, 3 (2003).

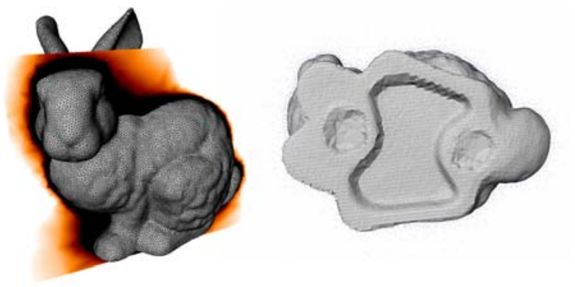
Figure 12: ***Bunny.*** *(left) the eigenvector (in false colors) of the solution of the generalized eigenvalue problem for the Bunny dataset—see also Fig. 4; (right) the hole at the bottom is filled.*

[DKT06] DESBRUN M., KANSO E., TONG Y.: Discrete differential forms for computational modeling. In *Discrete Differential Geometry*. ACM SIGGRAPH Course Notes, 2006.

[DLS05] DEY T. K., LI G., SUN J.: Normal estimation for point clouds: A comparison study for a Voronoi based method. In *Symposium on Point-Based Graphics* (2005), pp. 39–46.

[DS05] DEY T. K., SUN J.: *Normal and Feature Estimations from Noisy Point Clouds*. Tech. Rep. OSU-CISRC-7/50-TR50, Ohio State University, 2005.

[GS] GOMES F. M., SORENSEN D. C.: ARPACK++: A C++ implementation of ARPACK eigenvalue package.

[Gus02] GUSKOV I.: An anisotropic parameterization scheme. In *Proc. of Int. Meshing Roundtable* (2002), pp. 325–332.

[HDD*92] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLE W.: Surface reconstruction from unorganized points. In *Proc. of ACM SIGGRAPH* (1992), pp. 71–78.

[HK06] HORNUNG A., KOBBELT L.: Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. In *Symposium on Geometry Processing* (2006), pp. 41–50.

[HXMP05] HU G., XU J., MIAO L., PENG Q.: Bilateral estimation of vertex normal for point-sampled models. In *Int. Conf. on Comp. Science and Appl.* (2005), vol. 3480, pp. 758–768.

[KBH06] KAZHDAN M., BOLITHO M., HOPPE H.: Poisson Surface Reconstruction. In *Symposium on Geometry Processing* (2006), pp. 61–70.

[KSO04] KOLLURI R., SHEWCHUK J. R., O'BRIEN J. F.: Spectral surface reconstruction from noisy point clouds. In *Symposium on Geometry Processing* (2004), pp. 11–21.

[LP05] LANGE C., POLTHIER K.: Anisotropic smoothing of point sets. *Computer Aided Geometric Design 22*, 7 (2005), 680–692.

[MM06] MORDOHAI P., MEDIONI G.: *Tensor Voting: A Perceptual Organization Approach to Computer Vision and Machine Learning*. Morgan & Claypool, 2006.

[MNG04] MITRA N. J., NGUYEN A., GUIBAS L.: Estimating surface normals in noisy point cloud data. In *Int. J. of Comp. Geometry and Applications* (2004), vol. 14(4–5), pp. 261–276.

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H.-P.: Multi-level partition of unity implicits. In *Proc. of ACM SIGGRAPH* (2003), vol. 22(3), pp. 463–470.

[OF05] OUYANG D., FENG H.-Y.: On the normal vector estimation for point cloud data from smooth surfaces. *Computer-Aided Geometric Design 37*, 10 (2005), 1071–1079.

[PKKG03] PAULY M., KEISER R., KOBBELT L. P., GROSS M.: Shape modeling with point-sampled geometry. In *(SIGGRAPH)* (2003), vol. 22(3) of *ACM Trans. on Graphics*, pp. 641–650.

[PSQ06] PARIS S., SILLION F. X., QUAN L.: A surface reconstruction method using global graph cut optimization. *Int. J. Comput. Vision 66*, 2 (2006), 141–161.

[PWY*06] POTTMANN H., WALLNER J., YANG Y.-L., LAI Y.-K., HU S.-M.: Principal curvatures from the integral invariant viewpoint. *Comput. Aided Geom. Design* (2006). in print.

[RY06] RINEAU L., YVINEC M.: *A generic software design for Delaunay refinement meshing*. Tech. Rep. 5983, INRIA, 2006.

[SM00] SHI J., MALIK J.: Normalized cuts and image segmentation. In *Transactions on Pattern Analysis and Machine Intelligence* (2000), pp. 888–905.

[SW90] SMITH W. H. F., WESSEL P.: Gridding with continuous curvature splines in tension. *Geophysics 55*, 3 (1990), 293–305.

[TCR05] TOLEDO S., CHEN D., ROTKIN V.: TAUCS. Available at `http://www.tau.ac.il/~stoledo/taucs`, 2005.

[TLHD03] TONG Y., LOMBEYDA S., HIRANI A. N., DESBRUN M.: Discrete multiscale vector field decomposition. In *ACM SIGGRAPH* (2003), pp. 445–452.

[WCS05] WALDER C., CHAPELLE O., SCHÖLKOPF B.: Implicit surface modelling as an eigenvalue problem. In *Machine Learning ICML 2005* (2005), pp. 936–939.

[ZRS05] ZAYER R., R OSSL C., SEIDEL H.-P.: Setting the boundary free: A composite approach to surface parameterization. In *Symposium on Geometry Processing* (2005), pp. 91–100.

A Computing Covariance Matrices

Consider a given polyhedron Ω (either a bounded Voronoi cell V or the union of a set of such Voronoi cells). We can compute its covariance matrix around an arbitrary point p (center of mass in our application) by decomposing Ω into tetrahedra T_i formed by the triangles in a triangulation of the boundary surface and p, in a fashion similar to the computation of volume. Note that some of these tetrahedra may have negative volumes.

Without loss of generality, we can assume p to be the coordinates origin (by taking the relative coordinates w.r.t. p). For a single arbitrary tetrahedron T we start by considering the linear transform N that maps the canonical tetrahedron $\bar{T} = ((0,0,0)^t,(1,0,0)^t,(0,1,0)^t,(0,0,1)^t)$ to $T = (p,\mathbf{a},\mathbf{b},\mathbf{c})$. Consider the 3×3 matrix N defined as:

$$N = \begin{pmatrix}\mathbf{a} & \mathbf{b} & \mathbf{c}\end{pmatrix}$$

The order-2 moment matrix of T with respect to its base point p is defined as $det(N)NQN^t$, where Q is the order-2 moment matrix of the canonical tetrahedron $\bar{T}$ with respect to the origin:

$$Q = \iiint_{\bar{T}} \begin{pmatrix} x^2 & xy & xz \\ yx & y^2 & yz \\ zx & zy & z^2 \end{pmatrix} dxdydz = \frac{1}{120}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$$

The covariance matrix of Ω is then defined as

$$\mathrm{cov}(\Omega) = \sum_i det(N_i)\,N_i Q N_i^t.$$

Note that this expression is valid *independently* of whether Ω is convex or not.

Computation Speedup To speedup the computation of covariance matrices for unions of Voronoi cells U, we first compute the covariance matrix for each Voronoi cell V_i around its own center of mass. Now consider shifting the centroid of V_i from origin to p; we have:

$$\begin{aligned}\int_{V_i}(X+p)(X+p)^t dV &= \int_{V_i}(XX^t + Xp^t + pX^t + pp^t)dV \\ &= \mathrm{cov}(V_i) + m_i pp^t,\end{aligned}$$

where X means the relative coordinate w.r.t. the centroid of V, and m_i is the volume of V_i. The covariance matrix of the union becomes:

$$\mathrm{cov}(U) = \sum_i \left[\mathrm{cov}(V_i) + m_i p_i p_i^t\right],$$

where p_i is the displacement from the centroid of U to the centroid of V_i.

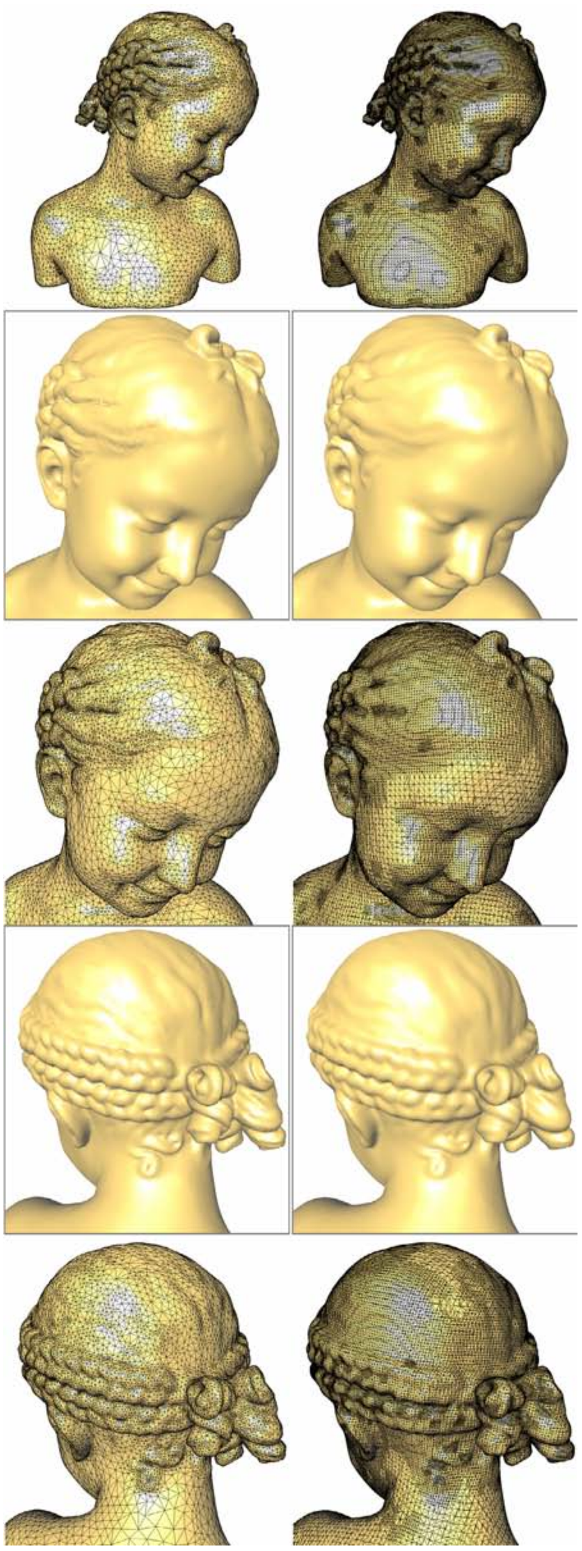
Figure 13: ***Comparison with Poisson reconstruction in 3D.*** *Without any normal information, our method (left) results in a similar shape (albeit less oversmoothed) to a Poisson reconstruction (octree depth 11) for which oriented normals were provided.*