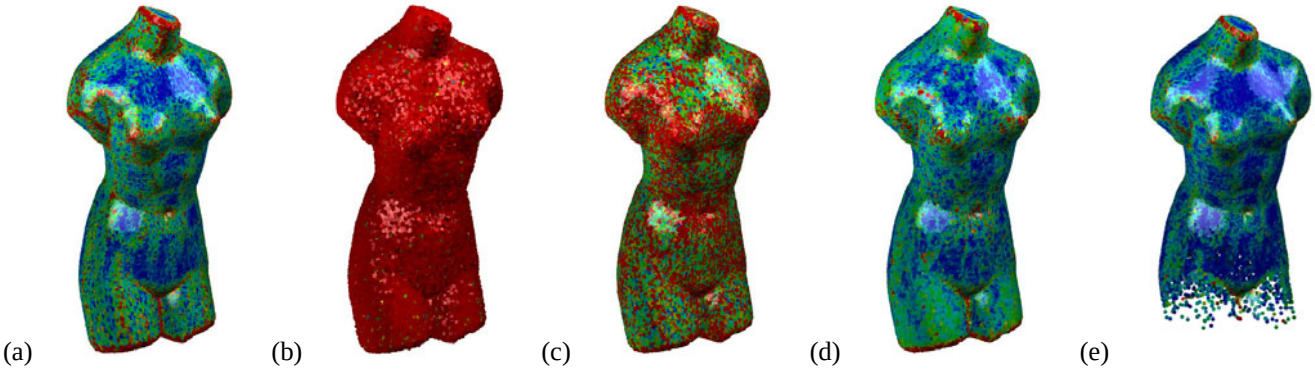
Anisotropic Smoothing of Point Sets*,**

Carsten Lange† Konrad Polthier‡

TU Berlin Zuse Institute Berlin



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

Figure 1: The initial point set of the Venus torso (a) was disturbed with a 3% normal and tangential noise to produce the initial noisy point set (b). Figures (c) and (d) show the point set after 20 respectively 50 iterations of the anisotropic fairing algorithm which detects and emphasizes high principal curvature values. Figure (e) shows a denoised torso with varying point density in z-direction. The original low resolution model of a Venus torso was subdivided to a resolution of 68K vertices and then disturbed with tangential noise to remove the visible subdivision patterns. All point sets are colored by their maximal principal curvature which is calculated from the point set.

ABSTRACT

The use of point sets instead of meshes became more popular during the last years. We present a new method for anisotropic fairing of a point sampled surface using an anisotropic geometric mean curvature flow. The main advantage of our approach is that the evolution removes noise from a point set while it detects and enhances geometric features of the surface such as edges and corners. We derive a shape operator, principal curvature properties of a point set, and an anisotropic Laplacian of the surface. This anisotropic Laplacian reflects curvature properties which can be understood as the point set analogue of Taubin's curvature-tensor for polyhedral surfaces. We combine these discrete tools with techniques from geometric diffusion and image processing. Several applications demonstrate the efficiency and accuracy of our method.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid and object representationsCurve, surface, solid and object representations.

Additional Keywords: point sets, anisotropic mean curvature, anisotropic sampling, shape operator, principal curvatures, anisotropic Laplace, mean curvature flow, feature detection.

* Supported by the DFG research center MATHEON "Mathematics for key technologies" in Berlin.

** Supported by the DFG Sonderfoschungsbereich 288 "Differentialgeometrie und Quantenphysik" in Berlin.

† TU Berlin
Institut für Mathematik, MA 6–2
Straße des 17. Juni 136
10623 Berlin
lange@math.tu-berlin.de

‡ Zuse Institute Berlin
Takustr. 7
14195 Berlin, Germany.
polthier@zib.de

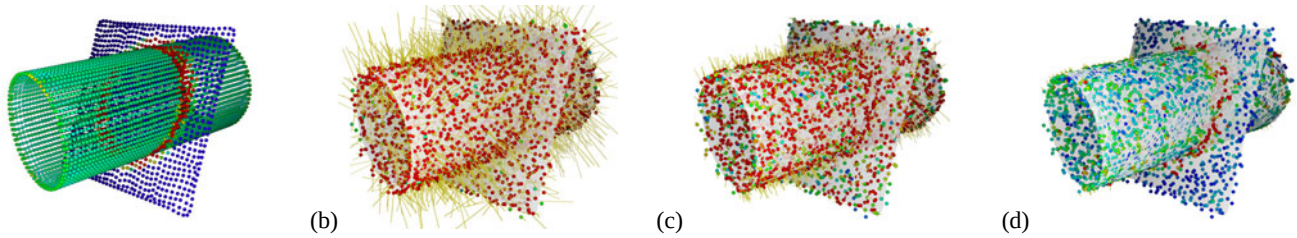
1 INTRODUCTION

A classical approach to describe surfaces is to describe them by local approximations. Many powerful and sophisticated methods have been developed to describe and design complex surfaces using splines, finite elements, or other polynomial approximations of higher order, and patch these local pieces of information together. Using relatively few sample points, it is possible to describe quite complex objects. Nevertheless, linear approximations, e.g. triangulations, became very popular. The advantage of easy handling is paid at the cost of a larger number of sample points needed for good approximations. Modern computer equipment is not only capable of handling large point sets, but also uses them: 3D-scanning devices naturally hand back large points sets that represent a given surface. An obvious direction of research aims therefore to associate structure (e.g. a triangulation) to these samples. By algorithms as the *crust algorithm* presented by Amenta, Bern, and Kamvysselis [2] such a mesh can be constructed. Moreover, it provably recovers the type of the surface correctly if some conditions on the sampling rate are satisfied. Once such a mesh is obtained, classical methods can be used to manipulate the data.

A different direction that became more and more popular during the last years is to deal with unstructured point samples only. At the cost of larger sample sets, the advantage is that no mesh structure has to be managed. Pure point based geometries are surprisingly flexible: up- and downsampling of the point set can be done as a postprocessing step as well as Gaussian fairing and surface editing; a good starting point for these subjects are for example [1] and [10].

Very often the samples one has to deal with are not precisely taken, that means, the sample points are obtained from measurements which are more or less accurate. Higher accuracy might be obtained at higher cost and/or more elaborate technical solutions, but software engineering can also help to obtain improved results.

Note: `http://www.zib.de/polthier/articles/pointSet` provides video animations of the experiments.



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

Figure 2: A cylinder with constant positive mean curvature is intersected with a plane. The point set consists of the union of both surfaces (a). 3% tangential and normal noise are added to the point set (b). After 50 iterations (d) the planar area clearly shows zero mean curvature (blue) while the cylindrical area (greenish) still has non-vanishing H. All figures are colored by the length of the anisotropic mean curvature vector. Figure (c) is after 15 iterations. Positive mean curvature leads to shrinking, and intersecting geometries have no well-defined principal curvature directions along the intersection line. Therefore this example demonstrates the behavior of our algorithm in two non-trivial test cases. Despite of the low resolution of the point set the positive mean curvature of the cylinder is recovered. The intersection line is recognized as an area of higher principal curvature and clearly separates the two areas, the curved cylinder and the flat plane.

A classical approach in denoising is the use of Laplacian curvature evolution. This has been implemented for meshes as well as for point based geometries. A side-effect of this *isotropic Gaussian fairing* that uses mean curvature is a smoothing effect at pointed features or edges. This is not wanted in general. We solve this problem by taking additional curvature information into account to obtain an *anisotropic fairing*. The additional curvature information are the principal curvatures and their directions. To obtain this data, we use ideas from differential geometry: *directional* and *principal curvatures* as well as the *Weingarten map* that is also known as *shape operator*. It is well-known that the eigenvalues of the Weingarten map are the principal curvatures and the Weingarten map can be expressed as an integral over the directional and mean curvatures. The main goal of this paper is to translate the directional curvatures, principal curvatures, and the Weingarten map into the setting of meshless point samples as follows: directional curvatures gives rise to a discretized Wingarten map via a discretized integral and an eigenanalysis of this Weingarten map yields the principal curvatures and directions. We finish with an application of these objects by presenting methods for anisotropic fairing of point sets which use directional curvature information and principal curvatures to detect features such as edges, see Figure 4 for example. A similar approach in the completely different setting of polyhedral meshes is described by Taubin [15, 16].

The paper is organized as follows:

1. We define directional curvatures for the points of the sample in subsection 3.1. This is done in an obvious way: We consider an approximation of directional curvatures for smooth surfaces.
2. We define a Weingarten map and principal curvatures for points of a point cloud in subsection 3.2 and 3.3. This is achieved by discretizing an integral formula for the smooth Weingarten map. A major difficulty is that sample densities may vary in different directions and one has to take this fact into account for the discretization.
3. We apply the derived notions of curvature to modify the well-known (isotropic) Laplacian to obtain an anisotropic Laplacian and use this anisotropic Laplacian for mean curvature flow techniques in Section 4. This enables us to fair noisy point samples without smoothing edges, see Figures 5, 6, and 7 for a first impression.

Independently, an anisotropic smoothing has been proposed that uses an eigenanalysis of the covariance matrix [4, 5] instead of the Weingarten map. We emphasize that this approach lacks curvature information since the covariance matrix at a point gives information of the tangential and normal directions at this point, not of their change. But it is this curvature that tells us about features such as edges.

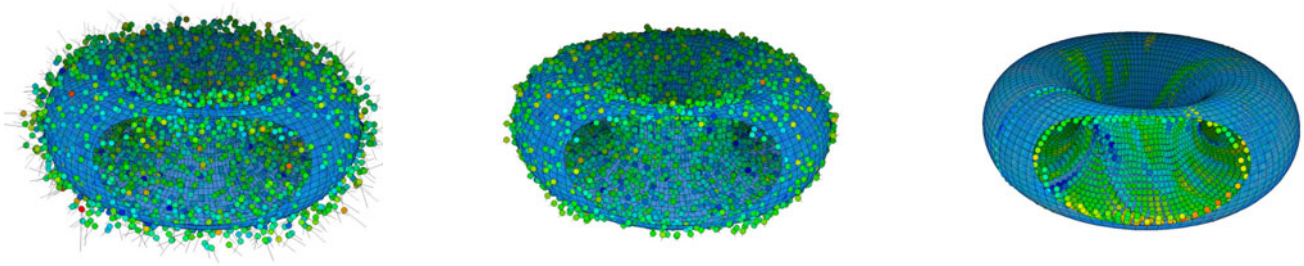
1.1 Related Work

A *moving least squares* method to associate a manifold structure to the point sample is considered for example in [9] and [1]. Levin describes a general method in $\mathbb{R}^d$ that consists of two steps. Firstly, an approximating hyperplane has to be determined for points p near the $(d-1)$-dimensional hypersurface S which is sampled by $\{p_i\}$. This is done by solving a non-linear minimization problem. Then the hypersurface is interpolated locally by polynomials that have the hyperplane defined in the first step as domain. In [1], this method is applied to surfaces in $\mathbb{R}^3$. The non-linear minimization problem is reformulated as an eigenvalue problem of an associated weighted covariance matrix. The second step is a system of linear equations whose size depends on the degree of the approximating polynomials. The main idea of this method is to implicitly define an approximating surface. [1] also describes a method to up- and downsample the set of sample points by using the implicitly defined surface.

Another *point based model* is described by Pauly, Kobbelt, and Gross, [10], where no implicit surface is constructed. They use a local approximation of the tangent space for each point of the sample that is also found by an appropriate eigenvector of a covariance matrix. To proceed in this direction one needs the notion of neighborhood of a point. Pauly, Kobbelt, and Gross consider k-neighborhoods, i.e. the k nearest neighbors w.r.t. Euclidean distance. Laplacian and multilevel smoothing is covered as well as up- and downsampling and surface editing.

From differential geometry we know how to compute the curvature tensor of a surface in a point. This knowledge can be used for an *approximation of the curvature tensor for polyhedral surfaces* at any vertex of the polyhedral surface in $\mathbb{R}^3$, as already described by Taubin [15]. Firstly, a normal vector is computed by averaging over (weighted) normals of all polygons that contain p. Here it is essential that one deals with meshes instead of point sets. The tangent plane at p is then defined, one has a projection of all edges e_i that contain p to this tangent plane, and a directional curvature in direction of e_i can be computed. This gives rise to a (3×3)-matrix that can be transformed into a discretized Weingarten map. Its eigenvalues are 0 with eigenvector in normal direction and the principal curvatures at p with principal directions as eigenvectors.

Isotropic Laplacian smoothing is discussed in the literature for meshes as well as for point based models, [6] and [10] and the references therein. A smooth Laplacian is linearly approximated in a vertex p by the umbrella operator for which a neighborhood of p must be specified. A discretized diffusion process (evolution) that is solved using an Euler scheme gives the denoising evolution. A well-known problem of this iterative smoothing is shrinkage. Different methods are introduced to cope with this: Taubin [16] varies



(a) (b) (c)

Figure 3: In contrast to all other examples, we use here the adjacency of the points of the undisturbed original torus in order to demonstrate the strong influence of correct neighbourhood information. Figures (b) and (c) after 15 respectively 50 iterations clearly demonstrate that the original surface is perfectly recovered if the correct spatial neighbourhood is used. The undisturbed torus is included for convenience and shown with reduced transparency.

a scale parameter, Desbrun, Meyer, Schröder, and Barr [6] rescale after each evolution step by the factor of shrinkage while Pauly, Kobbelt, and Gross [10] measure the shrinkage locally and displace neighbors to compensate the downsized volume. The *anisotropic diffusion problem* generalizes the model for isotropic smoothing. It is discussed in [3], [13], and [14] and dates back to work of Perona and Malik [11] who introduced this method as a smooth method for edge-enhancing and denoising of pictures.

The paper started as an effort to transfer the recent results for mesh optimization and noise removal of simplicial meshes [8] to point sets. This includes the derivation of a discrete shape operator to determine principal curvature directions.

2 REVIEW OF THE POINT BASED MODEL

Point samples taken from a smooth surface sufficiently dense do somehow reflect the structure of the surface. Points sufficiently close together should be distributed nearby the tangent plane, so the idea that the covariance matrix of neighboring vertices is somehow related to the tangent space and its normal is not too surprising.

The first task is to translate the objects from differential geometry to a point based model, that is in particular, we need a sensible definition of a tangent space. This is done by in a common approach using a notion of neighborhood. We interprete this in such a way that the tangent plane minimizes a least squares energy functional. If one has a notion of tangent space, it is straight-forward to transfer the technique of mean curvature flows from triangulated spaces to point clouds. This can also be found in the literature, [10].

The presented point based model is almost the same as the one used in [10] where a linear approximation is done more or less in the same way. The model differs from the one in [9] and [1] since they add a second step which defines their implicit surface. For our purposes a linear approximation is sufficient. The set of samples will be denoted by $P = \{\ p_i \mid 1 \leq i \leq N\ \}$ and we assume that P describes some underlying surface S that is embedded in $\mathbb{R}^3$. In particular, all points p_i are given by their 3 real-valued coordinates.

2.1 Neighborhoods

All computations will be based purely on neighborhoods induced by the Euclidean notion of vicinity instead of combinatorial proximity in the mesh setting. For fine samples and small Euclidean neighborhoods, both notions will be similar; a number of approaches for point sets are studied in [7]. The notion of k-nearest neighborhood N_k is used in [10]: $N_k(p_i)$ consists of the k nearest neighbors of p_i relative to the Euclidean distance. For the moving least-squares method normally all sample points are used and weighted according to distance. We consider an ϵ-k-neighborhood $N_k^\epsilon(p)$ of a sample point p, i.e. the intersection of the k sample points closest to p with the sample points contained in an ϵ-ball around p. The parameters ϵ and k will be globally set, therefore we use the shorthand N_p or N_i for neighborhoods of p or p_i.

2.2 Tangent Spaces

The neighborhood is now used to approximate the tangent space T_i at p_i which in turn is determined by a minimization problem: Let $b \in \mathbb{R}^3$ be any point and minimize the least squares energy given by

$$E_i(n, r) = \sum_{x \in N_i} (\langle x - b, n\rangle - r)^2,$$

where a 2-plane is given by its normal vector n and its distant r to b. It turns out that the barycenter $\bar{b}$ of N_i is distinguished in the sense that the minimizing hyperplane must contain $\bar{b}$. Moreover, a critical normal direction n is necessarily an eigenvector of the covariance matrix $M_i := \sum_{x \in N_i} (x - \bar{b})(x - \bar{b})^t$ of p_i:

$$M_i \cdot n = E_i(n, r) \cdot n. \tag{1}$$

We denote a unit-length eigenvector of the smallest eigenvalue of M_i by n_i and consider it as normal vector defining the approximation of the tangent space at p_i.

Approximating the tangent plane is sufficient for our purposes. We therefore do not use the higher order approximation proposed in the second step of the projection procedure of [9] and [1].

2.3 Isotropic Gaussian Fairing

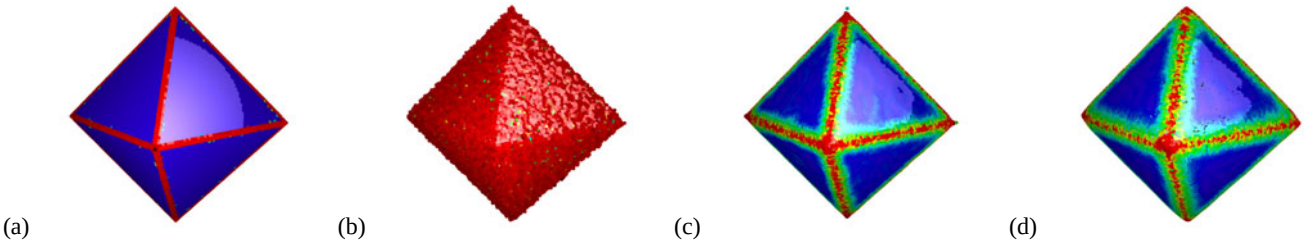
We shortly discuss some well-known facts and methods concerning the isotropic Laplacian to fix notation. Section 3.1 will then be concerned with the anisotropic case. As in differential geometry, we think of the Laplacian as the composition of div and ∇. Once a neighborhood is fixed, these operators have a combinatorial analogue.

The discrete version of $(\nabla)_{|p_i}$ is determined by the neighboring vertices $p_j \in N_i$ of p_i for vector-valued functions f:

$$(\nabla_{p_j})_{|p_i}(f) = (f(p_i) - f(p_j))e_{ij}$$

and

$$(\nabla\ )_{|p_i}(f) = \sum_{p_j \in N_i} (f(p_i) - f(p_j))e_{ij},$$



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

Figure 4: The octahedron is a standard test case for anisotropic smoothing. Our algorithm recognizes and recovers sharp edges. Note the single vertex on top of the Figure (c) which got stuck because all of its principal curvatures are above the curvature threshold. The original octahedron (a) with 32K vertices is obtained from a regular 4-1 subdivision. Figure (b) shows the initial point set with 1.5% tangential and normal noise. (c) shows the recovery of features and after 100 iterations with anisotropic MC flow. For comparison, Figure (d) shows an isotropic Laplacian smoothing which leads to lost sharpness of edges.

where e_{ij} denotes the vector $p_i - p_j$. The interpretation of div at p_i is as follows: For a vector $v = \sum_{p_j \in N_i} v_j e_{ij}$, the divergence $\mathrm{div}_{|p_i}$ at p_i is given as $\sum_{p_j \in N_i} \langle v, e_{ij} \rangle = \sum_{p_j \in N_i} v_j$. The inner product $\langle\ ,\ \rangle$ is not the inner product of $\mathbb{R}^3$ but the combinatorial inner product of $\mathbb{R}^{N_i}$ where the e_{ij} form an orthonormal basis. We define the isotropic Laplacian $\Delta_{|p_i}$ at p_i as $(\mathrm{div} \circ \nabla)_{|p_i}$, i.e.

$$\Delta_{|p_i} f = \sum_{p_j \in N_i} (f(p_i) - f(p_j)).$$

This definition coincides with the definition of Δ as umbrella operator. The Laplacian Δ can be interpreted as a matrix, a sample p_i is mapped to a linear combination of its neighbors.

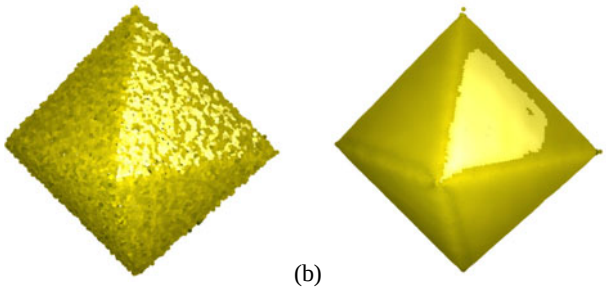
Isotropic Gaussian fairing is achieved via a PDE-method, i.e. by the diffusion equation

$$\frac{\partial S}{\partial t} = \lambda \Delta S,$$

which can be solved using a simple Euler scheme:

$$S^{n+1} = (\mathrm{Id} + \lambda \partial t \Delta) S^n,$$

see [6] for details in case of meshes and [10] in case of point sets. In each step the Laplacian has to be recomputed. For this computation, it has proven useful to keep the neighborhood of each point determined in the beginning instead of reassigning a new neighborhood at each step. Besides efficiency improvements, stability is also meliorated since clustering effects are prevented, [10].



(a) (b)

Figure 5: The two figures show the same point sets as in Figure 4 without vertex coloring. The coloring by principal curvature is replaced with a constant yellow point color. Figure (a) shows the initial noise and (b) the result of 100 iterations.

3 CURVATURES OF POINT SETS

The main goal of this section is the derivation of equation (4) for the Weingarten map W_i at each vertex p_i of a point set.

3.1 Directional Curvature

Once a tangent plane is specified, directional curvatures for points of the sample can be introduced by approximating a well-known formula from differential geometry.

Let us start with with the notion of directional curvature $\kappa_p(v)$, where v denotes a unit length vector in the tangent space in p of a smooth surface S. Such a directional curvature in differential geometry can be obtained by the following limit:

$$\kappa_p(v) = \lim_{s \to 0} \frac{2\langle n, \gamma(s) - p \rangle}{\|\gamma(s) - p\|^2},$$

where n denotes the surface normal and γ is a certain curve in S with $\gamma(0) = p$ and $\gamma'(0) = v$. Since we have defined a tangent space and a normal for each point p_i of the sample, we now define the directional curvature κ_{ij} in p_i in direction of $p_j \in N_i$:

$$\kappa_{ij} := \frac{2\langle n_i, p_j - p_i \rangle}{\|p_j - p_i\|^2}, \tag{2}$$

where n_i denotes the normal vector in p_i defined in the last paragraph. It is worth to mention that the directional curvature κ_p is a quadratic form and satisfies the identity

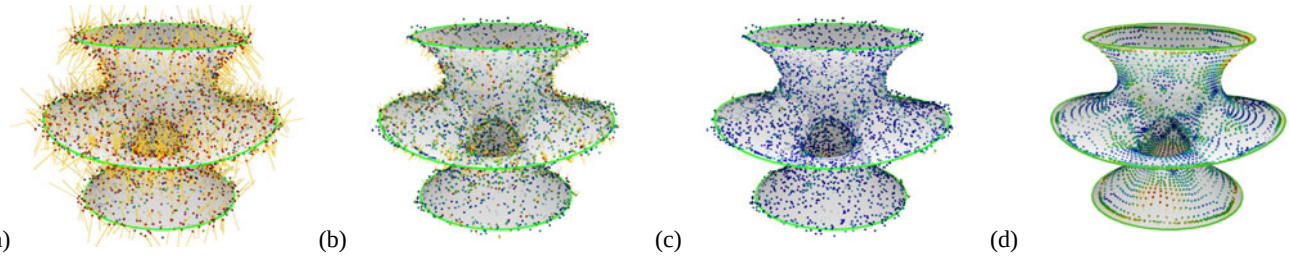
$$\kappa_p(e_\varphi) = e_\varphi^t \begin{pmatrix} \kappa_p^{11} & \kappa_p^{12} \\ \kappa_p^{21} & \kappa_p^{22} \end{pmatrix} e_\varphi, \tag{3}$$

where $e_\varphi = \binom{\cos\varphi}{\sin\varphi}$ relative to a basis $\{v_1, v_2\}$ of T_pS with $\kappa_p^{11} = \kappa_p(v_1)$, $\kappa_p^{22} = \kappa_p(v_2)$, and $\kappa_p^{12} = \kappa_p^{21}$.

3.2 Weingarten Map

We now rewrite the Weingarten map in integral form where directional curvatures can be found in the integrand. This integral formula is then approximated by a sum, but one has to be careful since the sample density in different directions may vary. We take this into account by estimating directional densities.

We can choose a basis that diagonalizes the matrix in (3). This is done by the principal curvature directions. Let us assume that v_1 and v_2 are already the principal curvature directions with principal curvatures κ_p^1 and κ_p^2. With respect to this basis the Weingarten



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

Figure 6: The smooth Costa surface is a minimal surface where both principal curvatures have equal absolute value and different sign. The points of a discrete Costa surface were randomly moved in normal and tangential direction with 3% noise (a) and then smoothed where Figures (b) and (c) show the status after 20 respectively 50 iterations. The vectors show the anisotropic mean curvature vector H whose length is also used for color coding of the vertices. The connectivity of the point set is taken from the noisy initial point set (a) and kept fixed during the iteration. Figure (d) shows the result of 50 iterations when using the connectivity of the unnoised point set (which is still different than the original connectivity of the triangle mesh).

map W_p reads as $\begin{pmatrix} \kappa_p^1 & 0 \\ 0 & \kappa_p^2 \end{pmatrix}$. The task is now to express the Weingarten map in integral form, i.e. we have to solve

$$W_p = \frac{1}{2\pi} \int_o^{2\pi} \mu_\varphi e_\varphi e_\varphi^t,$$

where $\mu_\varphi = \mu_1 \cos^2 \varphi + \mu_2 \sin^2 \varphi$. This yields

$$\mu_1 = \frac{3\kappa_p^1 - \kappa_p^2}{2} \qquad \mu_2 = \frac{-\kappa_p^1 + 3\kappa_p^2}{2}.$$

We denote the mean curvature of S at p by H_p and get W_p in terms of the directional curvatures κ_φ:

$$\begin{pmatrix} \kappa_p^1 & 0 \\ 0 & \kappa_p^2 \end{pmatrix} = \frac{1}{\pi} \int_0^{2\pi} (2\kappa_\varphi - H_p) e_\varphi e_\varphi^t,$$

These computations can also be performed in the ambient 3-space. We now translate this integral formula of the smooth category into a discrete formula in the point set setting by estimating the Weingarten map by

$$\sum_{p_j \in N_i} w_{ij} \kappa_{ij} e_{ij}^{\tan} (e_{ij}^{\tan})^t,$$

where N_i denotes the neighborhood of p_i, w_{ij} are weights that have to be determined in order to approximate the integral correctly, κ_{ij} is the directional curvature in p_i in direction of p_j, and $e_{ij}^{\tan}$ is the normalized tangential part of the vector $e_{ij} = (p_i - p_j)$.

The problem we face now is to estimate the weights w_{ij}. We will cope with this problem by estimating the density of samples in different directions. The approach is to consider the tangential part of the covariance matrix M_i (the covariance matrix encodes the distribution around the barycenter), express it similarly in integral form $\int_0^{2\pi} \delta_\varphi e_\varphi e_\varphi^t$ where we approximate the density δ_φ by the quadratic form $\delta_\varphi = \delta_1 \cos^2 \varphi + \delta_2 \sin^2 \varphi$. We obtain

$$\delta_1 = \frac{3c_1 - c_2}{2} \qquad \delta_2 = \frac{-c_1 + 3c_2}{2},$$

where c_i are the eigenvalues of the M_i. All computations are similar to those for μ_i earlier. If we now denote the normalized tangential part of an edge e_{ij} by $e_{ij}^{\tan}$ and the tangential eigenvectors of M_i by v_1 and v_2, we obtain for the density δ_{ij} in e_{ij} direction

$$\begin{aligned} \delta_{ij} &= \frac{3c_1 - c_2}{2} \langle e_{ij}^{\tan}, v_1 \rangle + \frac{-c_1 + 3c_2}{2} \langle e_{ij}^{\tan}, v_2 \rangle \\ &= 2 e_{ij}^{\tan} M_i (e_{ij}^{\tan})^t - \frac{1}{2} \operatorname{trace}(M_i). \end{aligned}$$

Since we know the result for regular $|N_i|$-gons of radius 1, we have to normalize, i.e. rescale by a factor $\frac{2}{N_i}$. Moreover, since we want integrative invariance, we have to substitute Δx by $\frac{2\pi \|p_i - p_j\|}{\delta_{ij}}$ in a Riemannian sum approximation and are now able to define the Weingarten map W_i at a vertex p_i of the point set:

$$W_i = \frac{1}{\pi} \sum_{p_i \in N_i} \frac{4\pi \|p_i - p_j\|}{|N_i| \delta_{ij}} (2\kappa_{ij} - H_i) e_{ij}^{\tan} (e_{ij}^{\tan})^t. \qquad (4)$$

Note that this shape operator is a translation of the operator Taubin [15] derived for polygonal meshes. The major difference in the point set category is our incorporation of a discrete directional density measure. This is not necessary in the mesh setting since volumes and interior angles of triangles are naturally available to incorporate directional denseties of the sample.

3.3 Principal Curvatures

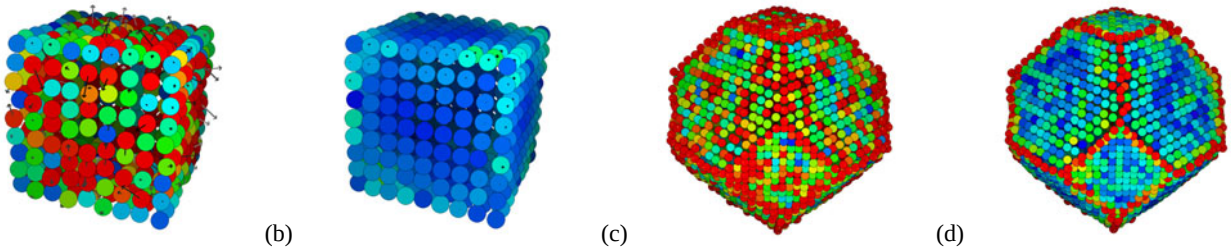
The eigenvalues and eigenvectors of our discrete shape operator W_i are the principal curvatures and principal curvature directions in p_i. They are fundamental in the next section to define the anisotropic Laplacian and for the anisotropic fairing algorithm we present.

4 ANISOTROPIC MEAN CURVATURE FLOW

We use an anisotropic Laplacian Δ^A to fair the point sample. The general idea of this approach as described in [13] and [3] can be summarized to solve a parabolic PDE with boundary constraints which reduces to the isotropic case for $A \equiv 1$. The modified PDE we consider is a bit less general compared to the one in [13] and is obtained by substituting the isotropic Laplacian by

$$\Delta^A_{|p_i} := \operatorname{div}_{|p_i} \circ (A_i \cdot \nabla)_{|p_i},$$

with $(A_i \cdot \nabla_{p_j})_{|p_i}(f) := g_{ij} \cdot (f(p_i) - f(p_j)) e_{ij}$. The real-valued function g is called cut-off function and will normally have the unit interval as range. The cut-off function can be used to further modify the Laplacian and to consider further geometric data, for example, we use it to distinguish between neighbors with small and high directional curvatures. Or we can detect features like an edge, for example, by comparing the two principal curvatures: If one is almost vanishing while the other is larger than a certain threshold, then we may consider the sample point as being sampled from a straight edge.



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

Figure 7: This example shows how the anisotropic mean curvature vector changes direction and length during an iteration on a low resolution model (<1K). The anisotropic shape operator recognizes the sharp edges and corner of the cube. The low resolution cube (a) has 2.5% noise, (b) is after 50 iterations. The truncated octahedron has two type of edges with different dihedral angle. One set of edges has the same dihedral angle as an octahedron (109.47°) and the others (125.26°) are about 15° larger. This difference makes it harder to distinguish edges on a noisy version of the truncated octahedron. Figure (c) shows the truncated octahedron with 2% noise. After 60 iterations the edges are clearly recovered as feature lines (d). All four surfaces show the anisotropic mean curvature vector and are colored by its magnitude.

5 EXPERIMENTAL RESULTS

The described ideas for computing directional and principal curvatures have been implemented using the JavaView environment [12]. We now discuss features of the implementation and observations made while fairing examples.

As neighborhoods are concerned, we compute a N_k^ϵ-neighborhood for each sample from the noisy data. In accordance to the well-established convention for isotropic denoising, [10], the N_k^ϵ-neighborhoods are computed once in the beginning and kept fixed during the evolution unless an update is forced by the user. Whenever N_k^ϵ-neighborhoods are recomputed, the user can reassign new global values for ϵ (the diameter of the ball containing the neighbors of a vertex) and an upper bound for the number k of samples in a neighborhood. As a side remark, we made the following interesting observation: the fairing works well if we determine the N_k^ϵ-neighborhood of a noisy point cloud. But assume one knows for some reason the N_k^ϵ-neighborhood of a point in case of a noiseless configuration and applies these (noiseless) N_k^ϵ-neighborhoods to the samples after adding noise. The result of fairing under this rather artificial condition is amazing, see Figures 3 and 6(d). This leads to the observation that the proper notion of neighborhood is extremely important for the whole process of denoising.

Tangent spaces, directional, principal curvatures, and the anisotropic Laplacian are automatically computed after each step of the Euler scheme.

The anisotropic Laplacian depends crucially on the choice of A_i as described in the preceeding section. At the time of writing, we offer two choices that are tuned by a threshold parameter λ: either a sharp cut-off function g_{ij}^{sharp} or a continuous cut-off function g_{ij}^{cont}. In the first case, the neighborhood might be lessened: a sample $p_j \in N_i$ is not considered for $\Delta^A_{|p_i}$ if $|\kappa_{ij}| \geq \lambda$, while in the continuous case the sample is shaded out. More precisely we consider

$$g_{ij}^{\text{sharp}} = \begin{cases} 1 & |\kappa_{ij}| < \lambda, \\ 0 & |\kappa_{ij}| \geq \lambda; \end{cases}$$

$$g_{ij}^{\text{cont}} = \begin{cases} 1 & |\kappa_{ij}| < \lambda, \\ \frac{\lambda^2}{\lambda^2 + 10(\kappa_{ij} - \lambda)^2} & |\kappa_{ij}| \geq \lambda. \end{cases}$$

The effect of this anisotropic fairing at each step of the mean curvature flow can be summarized as follows. The fairing process prefers to consider neighbors of a directional curvature less than λ. The idea is that rather "flat directions" are flattened by the mean curvature flow, while directions of "large" curvature remain such directions. The continuous cut-off function g_{ij}^{cont} is a slight variation of the cut-off function G described in [3].

In order to focus on the evolution in normal direction and to neglect tangential drift, the user can choose the *constrain interior* option where the mean curvature vector is projected onto the normal direction, and this projected vector is used in the evolution.

We color the samples according to their value of either the maximal principal curvature or the length of the anisotropic mean curvature vector. We parametrize the color circle from 0 to 2π and consider the linear function that assigns to the minumum the color that corresponds to 0 and to the maximum the color that corresponds to 1.5π.

We can also use the principal curvatures instead of directional curvatures to detect features of the point sample. Three different approaches have been studies so far. The first focusses on a parameter we call *edge quotient*. The user chooses an edge quotient threshold Q and for every point p of the sample the quotient q_p of the principal curvatures is computed. If $q_p < Q$ then p is contained in features that should be enhanced. This approach works fine in order to detect the edges of the cube, the octahedron and the edge of the Costa surface. Unfortunately, manually tuning is required to choose an appropriate value of Q. Secondly, we successfully detect a point of a feature if the larger principal curvature is larger than a threshold. Thirdly, we tried also to detect features by saying that a point p_i belongs to a feature if there is a directional curvature κ_{ij} such that $|\kappa_{ij}| > T$, where T is a parameter that has to be chosen sufficiently. This approach does not work well.

6 CONCLUSION AND FUTURE RESEARCH

It is possible to fair noisy point samples by the anisotropic smoothing presented in this paper. In contrast to the isotropic mean curvature flow method for denoising, that converges to a sphere, the presented method is able to recover edges of the original surface as the polytopes and the Venus surface show where no underlying mesh is used. Unfortunately, the features are not yet automatically detected; manual choice a proper value for the edge quotient is necessary.

Interesting is also the rôle which the choice of neighborhoods plays: The result of anisotropic (and isotropic) fairing changes significantly if different neighborhoods are chosen in the beginning. The examples of the torus and the Costa surface show an even better fairing if one determines the N_k^ϵ-neighborhood of the surface without noise, adds noise to the surface and starts anisotropic fairing with the noiseless N_k^ϵ-neighborhoods compared to the fairing started with proposed N_k^ϵ-neighborhoods determined from a noisy sample. We admit that these *true* neighborhoods are artificial. But

interesting questions are related. So far, the effect of different initial neighborhoods has not been studied. Is it possible to determine better neighborhoods than for example the N_k^ϵ-neighborhoods from the noisy sample?

REFERENCES

[1] M. Alexa, J. Behr, D. Cohn-Ohr, S. Fleishman, D. Levin, & C. T. Silva, *Computing And Rendering Point Set Surfaces*, IEEE TVCG, 9, pp. 3–15, 2003.

[2] N. Amenta, Marshall Bern, & M. Kamvysselis, *A New Voronoi-Based Surface Reconstruction Algorithm*, Proceedings of Siggraph 1998.

[3] U. Clarenz, U. Diewald, & M. Rumpf, *Anisotropic Diffusion in Surface Processing*, Proceedings Visualization 2000.

[4] U. Clarenz, M. Rumpf, & A. Telea, *Finite Elements on Point Based Surfaces*, to appear in Computers and Graphics, 2004.

[5] U. Clarenz, M. Rumpf, & A. Telea, *Fairing of Point Based Surfaces*, Computer Graphics International (CGI 2004), pp. 600–603, 2004.

[6] M. Desbrun, M.Meyer, P. Schröder, & A.H.Barr, *Implicit Fairing Of Irregular Meshes Using Diffusion And Curvature Flow*, Proceedings of SIGGRAPH 1999.

[7] M. Floater & M. Reimers, *Meshless Parametrization And Surface Reconstruction*, Comp. Aided Geom. Design, 18, pp. 77–92, 2001.

[8] K. Hildebrandt & K. Polthier, *Anisotropic Filtering of Non-Linear Surface Features*, Eurographics 2004, Computer Graphics Forum, 23(3):391–400, 2004.

[9] D. Levin, *Mesh-Independent Surface Interpolation*, to appear in *Geometric Modeling for Scientific Visualization*, Brunnett, Hamann, Müller (eds.), Springer-Verlag.

[10] M. Pauly, L. Kobbelt, & M. Gross, *Multiresolution Modeling Of Point-Sampled Geometry*, CS Technical Report #378, September 16, 2002, `http://graphics.ethz.cg/main.php?Menu=5&Submenu=2`.

[11] P. Perona & J. Malik, *Scale Space And Edge Detection Using Anisotropic Diffusion*, IEEE Computer Society Workshop on Computer Vision 1987.

[12] K. Polthier, K. Hildebrandt, E. Preuß, & U. Reitebuch, *JavaView – A 3D geometry viewer and mathematical visualization software*, `www.javaview.de`.

[13] T. Preußer & M. Rumpf, *Anisotropic Nonlinear Diffusion In Flow Visualization*, Proceedings IEEE Visualization 1999.

[14] T. Tasdizen, R. Whitaker, P. Burchard, & S. Osher, *Geometric Surface Smoothing via Anisotropic Diffusion of Normals*, Proceedings of IEEE Visualization 2002.

[15] G. Taubin, *Estimating The Tensor Of Curvature Of A Surface From A Polyhedral Approximation*, Fifth International Conference on Computer Vision (ICCV'95).

[16] G. Taubin, *A Signal Processing Approach To Fair Surface Design*, Proceedings of SIGGRAPH 1995.