INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Robust Higher-Order Filtering of Points

Florent Duguet, Frédo Durand and George Drettakis

N° 5165

April 2004

THÈME 3

ISSN 0249-6399

INRIA
SOPHIA ANTIPOLIS

Robust Higher-Order Filtering of Points

Florent Duguet*, Frédo Durand† and George Drettakis‡

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projets REVES

Rapport de recherche n° 5165 — April 2004 — 15 pages

Abstract: Processing noisy point data is a tedious task. It often requires smoothing and meshing, followed by mesh processing algorithms. In this work we present a robust filter which operates directly on points using higher order surface approximations, computed iteratively. We extend the bilateral filter by using and filtering surface curvature, and introducing a quality parameter based on our local surface approximation. Using this process input point data is denoised and smoothed, while preserving features. We also extend our filter to allow feature enhancement, by controlling shock formation resulting in sharp edges in regions of high curvature. We show results of our algorithms on several scanned point sets.

Key-words: Computer Graphics, Computational Geometry, Filtering, Denoising, Point-Based Geomety

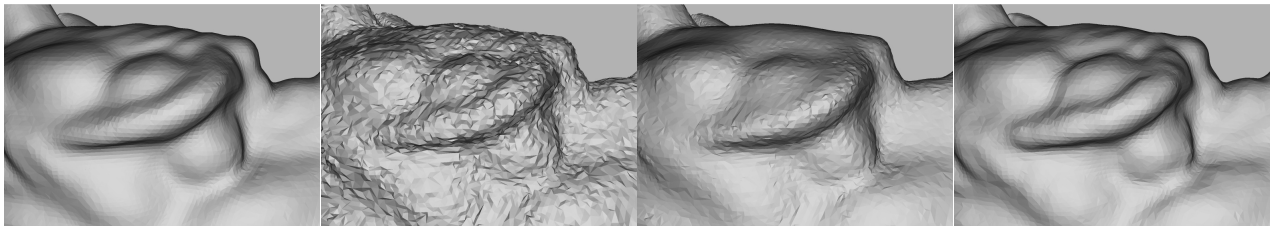
* REVES/INRIA Sophia-Antipolis
† Massachusetts Institute of Technology
‡ REVES/INRIA Sophia-Antipolis

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Filtrage Robuste de Points avec Prédicteurs Non Linéaires

Résumé : Le traitement de nuages de points bruités est une tâche difficile. Il est souvent nécessaire de filtrer et mailler les points avant de pouvoir les traiter. Dans ce travail, nous présentons un filtre robuste qui opère directement sur les nuages de points en utilisant des approximations de surface d'ordre deux, calculées itérativement. Nous étendons le filtrage bilatéral en utilisant et filtrant la courbure de la surface, ainsi qu'en introduisant un coefficient de qualité basé sur notre approximation locale de surface. En utilisant cette technique, les points sont débruités puis lissés, tout en préservant les éléments caractéristiques. Nous étendons également notre filtre afin de pouvoir accentuer les éléments caractéristiques tout en contrôlant la formation des arêtes vives dans les régions de forte courbure. Nous présentons des résultats de notre algorithme sur plusieurs nuages de points scannés.

Mots-clés : géométrie algorithmique, filtrage, débruitage, géométrie basée points



Original mesh | Noisy mesh | [Jones et al. 2003] | Our new approach

Figure 1: Comparison between our technique and a recent feature-preserving filtering method. Artificial Gaussian noise was added to a mesh ($\sigma = 1/5$ of the mean edge length), and the mesh was then smoothed using the two techniques. The meshes are flat shaded to better reveal the geometry.

1 Introduction

Advances in scanning technologies have resulted in a widespread availability of large 3D datasets, e.g. [LPC*00, BR02]. These advances put an important emphasis on the robust and efficient treatment of these models for modeling and rendering operations. In this work, we perform filtering and geometric processing directly on the acquired points using local surface approximations. We perform *local* processing on the *original* data points.

The core of our approach is a robust local approximation of the surface based on point-neighborhood information with no connectivity. We robustly estimate second-order polynomial height functions called *Jets* [CP03, GKS00]. They correspond to a Taylor expansion and encode differential properties such as normal and curvatures. We show that jets define a high-quality *local* surface approximation and alleviate the need for connectivity for a number of important geometric operations. Computation is *local*, *robust to outliers* and *preserves sharp features*. Our central contribution is the development of a feature-preserving filter on jets. It extends bilateral filtering [JDD03, FDCO03, CT03] in two important ways. First, our jet estimation alleviates the need for connectivity or normal information in the input data. Second, we exploit and robustly filter curvature information. This better captures shape information and results in higher-quality filtering.

Our approach also enables general point-data processing and display. Note that for all the applications we describe, we work on the *original* points (potentially moved during filtering). The result of the filtering process permits meshing of noisy input data using standard meshing algorithms [BBX95, ABK98, Cha03, JDB02], and the control of shock formation.

In summary, this paper makes the following contributions:

- An iterative evaluation of jets that provides normal and curvature from scattered noisy point clouds.
- The extension of bilateral filtering to point clouds using, and filtering, surface curvature.

We perform all operations directly on the scattered points with no need for connectivity. We do not require regular sampling of the geometry. The results of filtering, when combining *Jets* as second-order surface approximations and the bilateral filter, are much better than using planes or normals; See Figures 1, 13 and 12.

1.1 Related Work

Taubin [Tau95b] pioneered mesh filtering from a signal processing perspective. Desbrun et al. [DMSB99] improved stability for irregular meshes using a geometric-flow algorithm. However, these methods indistinctly smooth high-frequencies due to noise and features.

Borrowing inspiration from image processing [PM90], a number of researchers introduced *feature-preserving* mesh smoothing, e.g.,, [DMSB00, CDR00, ZF02, BX03]. The central idea of these techniques is to prevent diffusion across sharp features using a non-linear term depending on the curvature tensor. Other authors perform anisotropic diffusion on the normal field and later evolve the surface to match the normals [Tau01, BO01, OBS02, TWBO02].

Locally-adaptive Wiener filtering has been successfully adapted to meshes [PSZ01] and point-sample surfaces [PG01]. However, this method requires parameterization and/or a regular mesh topology. Alexa [Ale02] defined Wiener filtering of meshes using a technique closer to diffusion, but connectivity is still needed.

Bilateral filtering [TM98, DD02] was recently adapted to meshes by Jones et al. [JDD03], Fleishmann et al. [FDCO03] and Choudhury and Tumblin [CT03]. Filtering is not expressed as a diffusion but as a weighted sum of neighboring values. The weight depends not only on the spatial distance, but also on a similarity term that penalizes values across features. In

the adaptation to meshes, the local geometry is decomposed into a tangential and a normal term to allow the definition of spatial distance vs. similarity. The normal information allows vertices across sharp corners to be rejected as outliers. Our work builds upon bilateral filtering for the calculation of robust local surface approximation as we will discuss further in Section 3.

These techniques need connectivity or a good initial guess of the normal information. In general, the computation of reliable differential properties from discrete geometry is an important step for a number of geometry processing operations. Hoppe et al. cast normal calculation as an eigenvalue problem [HDD*92]. Taubin [Tau95a] also uses eigenanalysis and computes the curvature tensor. Desbrun et al. [DMSB99] propose a normal computation scheme for irregular meshes. They later extended it to higher-order quantities [MDPS02]. These methods however require connectivity. Mitra et al. [MN03] discuss the effect of noise on computation of normals for point clouds. Gopiy et al. [GKS00] define local polynomial approximations similar to jets, but they use a very local neighborhood and a simple estimation scheme. Cazals and Pouget [CP03] use a least-square formulation to compute jets. We extend these approaches to evaluate robust second-order surface approximations. In addition, we show how jets can be used for feature-preserving filtering.

Medioni et al. [Mea00] introduced a tensorial formulation of the shape of a point set. At each point, a symmetric quadratic form is estimated and its eigenanalysis reveals the shape of the sampled geometry: an isolated point, a curve or a surface. These surface tensors are similar in formulation to the 3D curvature tensor we use, but we present a new and robust method to construct and exploit them.

The moving least squares (MLS) [ABCO*03] is an iterative local surface approximation technique. For any input point close to the surface, and an input normal, the algorithm iterates on a polynomial height function as a local approximation of the surface. This approximation is iteratively refined by intersecting the ray defined by the point and the normal, with the height function, and doing a least squares fit on the neighboring points. Pauly et al. [PKKG03] use MLS to perform boolean operations on point clouds and obtain sharp features at the boundary between the two objects. The MLS reconstruction of point clouds itself is however smooth and does not preserve sharp features.

2 Differential properties for point clouds

Throughout this paper we consider the point set as a sampling of an underlying surface. In the case of data resulting from a 3D scan, this is the real surface of the scanned object. We will not construct an approximation of the complete surface at any time; We will rather construct local approximations to the surface, using a height function defined in a local frame at each input point. The estimation of local surface differential properties has been used for rendering in [KV01] for the case of parametric surfaces.

2.1 *Jet* definition

For our surface approximation, we will use height functions in a local frame for a given neighborhood. For conciseness, we will call these polynomial height functions, together with their local frames '*Jets*', as in [CP03]. The first-order *Jet* is the tangent plane. In our approach we use second-order height functions, or 2-*Jets*. We choose a local coordinate system for these height functions so that they can be expressed in the Monge form:

$$h(x,y) = \frac{1}{2}c_x x^2 + \frac{1}{2}c_y y^2 \tag{1}$$

In this coordinate system, the local surface normal is the $\mathbf{e}_z$ axis, c_x and c_y are the two principal curvatures, $\mathbf{e}_x$ and $\mathbf{e}_y$ are the associated directions.

We next describe the process of surface approximation with 2-*Jets* built on the input point cloud. Since we restrict our approach to 2-*Jets* , we will simply use the term jets from now on.

Jets are stored as a simple data structure associated with the input points. The structure contains the origin (which is the relative position with respect to the point), the local frame, in particular the vector e_x and the principal curvatures c_x and c_y.

2.2 *Jet* Estimation

We estimate the jets using an iterative algorithm, illustrated in 2D in Figure 2. The initial step of the algorithm provides a first approximation of the tangent plane. This coarse approximation is then refined using an iterative algorithm. The input of the algorithm is an origin (first estimation), and a small set of k nearest neighbors to the origin.

We use the method of [CP03], to approximate the $\mathbf{e}_z$ axis, and fit a second degree height function to the point set. In contrast to [CP03], Gaussian weights are used for the minimization step. We obtain an initial estimation of the height

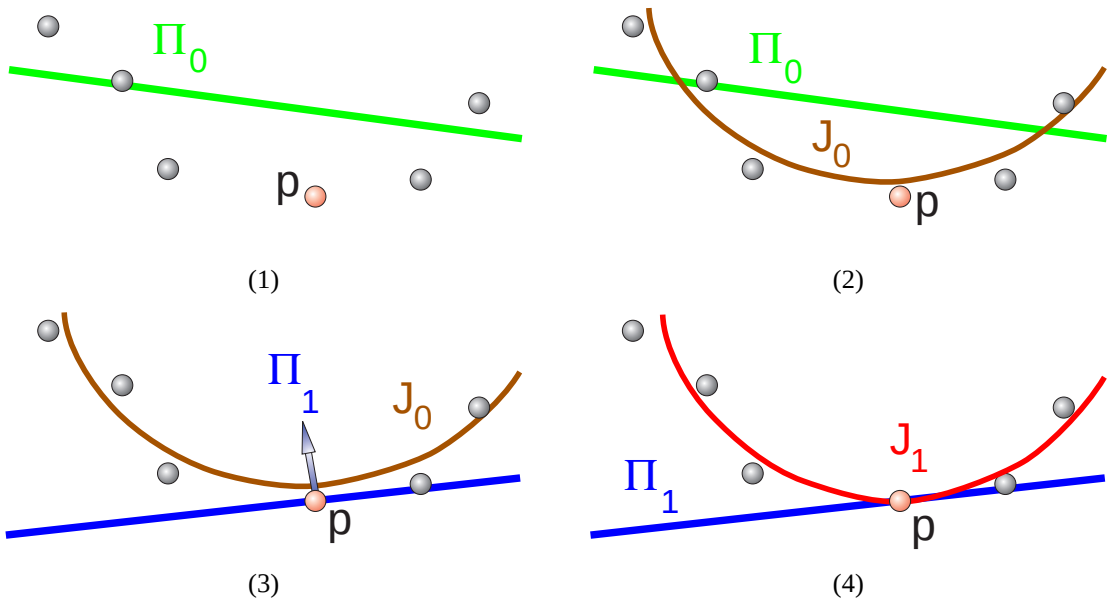


Figure 2: Illustration of jet estimation. (1) first estimation of the tangent plane, (2) first estimation of the jet, (3) refinement of the tangent plane, (4) refinement of the jet.

function which has the following form:

$$h(x,y) = c_0 + a_0 x + b_0 y + d_0 x^2 + e_0 xy + f_0 y^2 \tag{2}$$

We then extract the normal from this height function: $\vec{n} = [-a_0, -b_0, 1]$. We change the local frame according to the new normal and iterate (Figure 2(3)-(4)). Iterating four times provided sufficient precision in the examples we tested: The a_i and b_i factors decrease rapidly to a very small value. At this point, we fix the normal of the local frame, and the height function is given by:

$$h(x,y) = c_i + d_i x^2 + e_i xy + f_i y^2 \tag{3}$$

We then move the origin of the local frame along the $\mathbf{e}_z$ axis such that we can eliminate c_i, and rotate the $\mathbf{e}_x$ and $\mathbf{e}_y$ axes around the normal to eliminate the e_i coefficient. The height function is thus in its Monge form, and the principal curvatures, curvature directions and tangent plane are known. Results of jet estimations are illustrated in Figure 4, upper row.

Around boundaries, the quadric which best fits the existing points will be estimated. It is symmetric and will extend the surface further with the same second-order differential properties. This surface extension will be therefore second-order smooth.

Our approach for local surface approximation is related to the moving least squares (MLS) [ABCO*03]. For any point on the surface, given a normal, the MLS can provide a polynomial height function approximating the surface. The iterative algorithm for the polynomial height function estimation is quite similar to the one described above. However, the two techniques differ: in the MLS, the normal to the plane used for the height function definition is not modified during the process. Since the initial normal is obtained with a rough estimation (which depends on sampling uniformity), the normal needs to be refined during the process, so that the curvature estimation is more precise.

2.3 Sampling rate and jet quality

The jet estimation is the result of a minimization process. The function which is minimized is a weighted sum of the squared distance between a point and the jet. If the function is zero then all points are on the jet. We can thus assume that the jet is a good approximation of the surface. When the function has large values the jet is a poor approximation of the surface. We next define a quality function which is used during filtering, to control the influence of the jets. Intuitively, the influence of the jets in the various filters is high when their quality is high, and low otherwise.

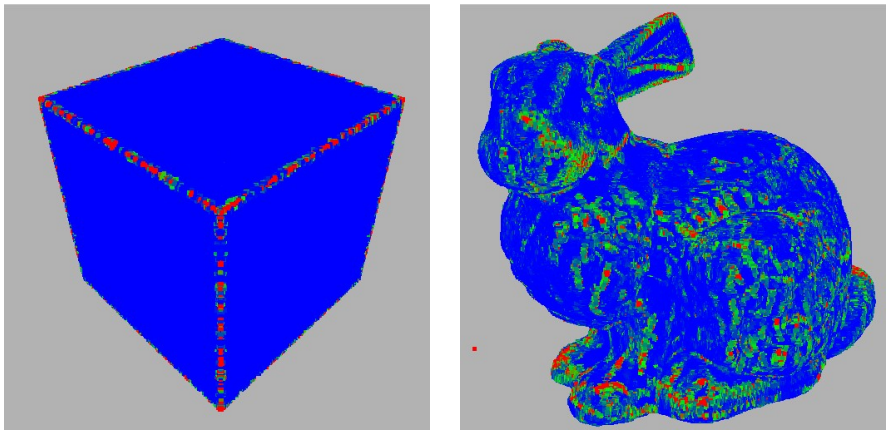


Figure 3: Illustration of the quality of a jet. (left) a cube randomly sampled. (right) the bunny point set. Blue is high quality, red is low quality.

We define the quality function as follows. Let w_i be the weight used for point i for the weighted least squares minimization. Let s_d be the average sampling distance, and $n = h(x, y) - z$. We define the cost of a jet as:

$$c(j) = \sum_{i=0}^{k} w_i e^{-\frac{s_d^2}{n^2}} \tag{4}$$

Using this cost function, we define a quality coefficient between 0 and 1 for a jet using an exponential: $q(j) = e^{-c(j)}$. This quality coefficient is illustrated in Figure 3. Note in the cube example that the quality coefficient is excellent in flat regions, and gets worse around sharp edges; overall, jets have lower quality in presence of noise and around sharp edges.

3 Bilateral Filtering of *Jets*

After local surface estimation using jets we proceed with bilateral filtering. A jet can be seen as a position, a normal, and two principal curvature directions and magnitudes. Each of these is filtered separately.

We introduce a new element to bilateral filtering which is the quality coefficient (see Section 2.3). This coefficient corresponds to the quality of the jet as an estimation of the surface.

3.1 Bilateral Filtering

For each of our filters, either curvature, normal or point filtering, we use the same filtering equation. Only some of the distance functions will change from one filter to another. Call X the quantity to be filtered. For bilateral filtering, we use a spatial weight, an influence weight, the quality coefficient and a predictor for X. Spatial and influence weights are given by Gaussians on distances. $\Pi_j(p)$ is the predictor of the quantity for point p on jet j. The quantity $q(j)$ is the quality of jet j. Let $d_s(j, p)$ be the spatial distance of point p with respect to jet j, and $d_i(j, p)$ the influence distance, then:

$$X'(p) = \frac{1}{k(p)} \sum_{j \in \Omega} q(j) e^{-\frac{d_s^2(j,p)}{k\sigma_f s_d^2}} e^{-\frac{d_i^2(j,p)}{\sigma_g s_d^2}} \Pi_j(p) \tag{5}$$

where $k(p)$ is the normalization factor.

$$k(p) = \sum_{j \in \Omega} q(j) e^{-\frac{d_s^2(j,p)}{k\sigma_f s_d^2}} e^{-\frac{d_i^2(j,p)}{\sigma_g s_d^2}} \tag{6}$$

We use the same distance function $d_s(j, p)$ for each filter: the Euclidean distance between point p and the origin of jet j, and the same value of σ_f. Since the distances are divided by the average sampling distance, the absolute size of the model is not a parameter, and we used the same value for $\sigma_f = 1$ for all models. The value of σ_g however depends on the distance function.

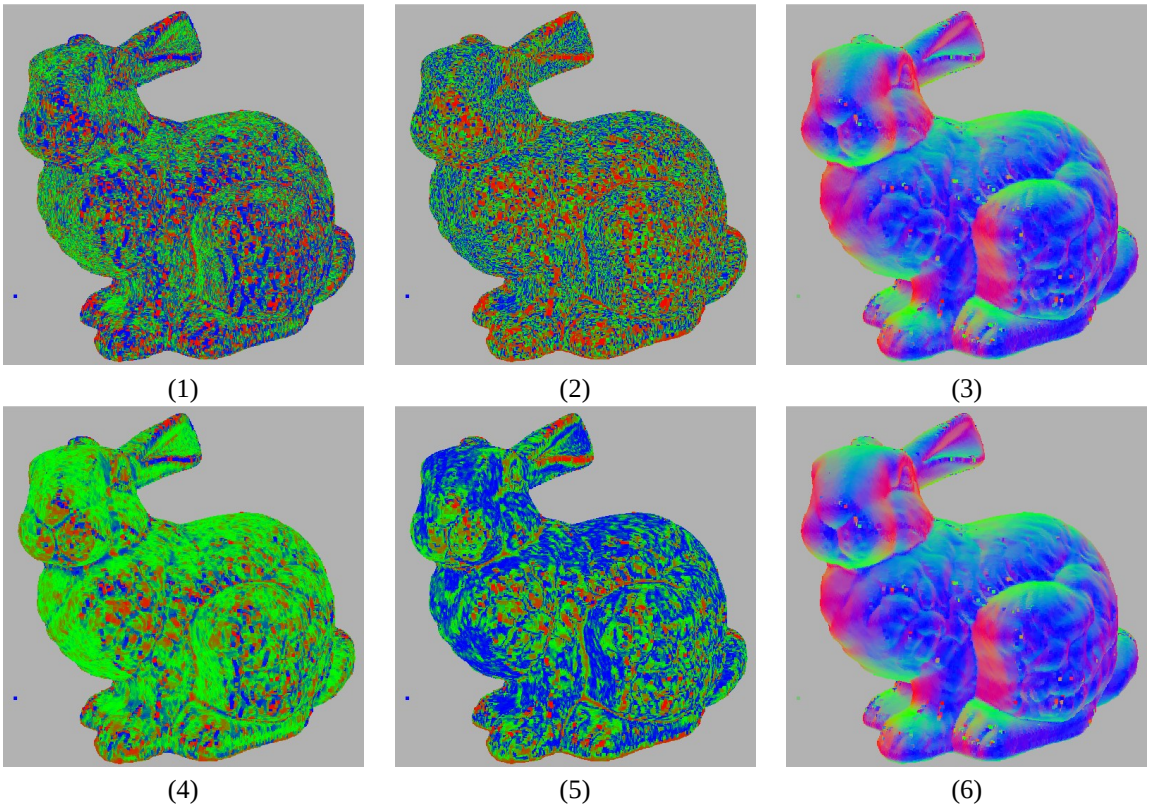


Figure 4: Illustration of the jet filtering. upper row: estimation, lower row: after filtering. (1,4) Gaussian curvature, (2,5) mean curvature, (3,6) normal. The false color scheme is the following: normals: xyz converted to rgb color; Mean curvature: blue for zero and red for high magnitude; Gaussian curvature: Green for zero, red for positive and blue for negative.

3.2 Using Second Order Predictors

Predictors Jones et al. [JDD03] presented a bilateral filter on meshes. In their technique, they used point predictors for filtering: the projection of the point on a plane defined by a neighboring face. We reused this idea of a predictor, but instead of a plane, we use our second order surface approximation (jet). We can use the predictor for both point prediction and normal prediction. Each of these is illustrated in Figure 5 for points and 6 for normals.

In what follows, we distinguish the processing of noisy and less noisy data. This depends on the input; noisy data will typically be filtered twice, passing the output of the first filter to the subsequent filter.

The position filter for noisy data is that of Jones et al. [JDD03] using new coefficients and new predictors. The influence distance is the distance between the point and its predictor on the jet. Our approach for less noisy data is inspired by Fleishmann et al. [FDCO03]: we filter point positions along their normals. We cast a ray originating at the point in the direction of the normal. The intersection of this ray with the jet is the predictor. We thus need good normals for this technique. The distance function is the distance between the point and the predictor projected on the axis of the normals. In what follows, the first approach, inspired by Jones et al. [JDD03], is called the drifting filter, and the second is called the non-drifting filter due to its behaviour during iterations.

Normal filtering For noisy data the influence distance is a distance between normals:

$$d_i(n, n') = (1 - \|n \cdot n'\|)^2 \tag{7}$$

This approach is close to that of Choudhury and Tumblin [CT03]. For less noisy data, the influence distance is the Euclidean distance from the point and the point predictor (similar to Jones [Jon03]).

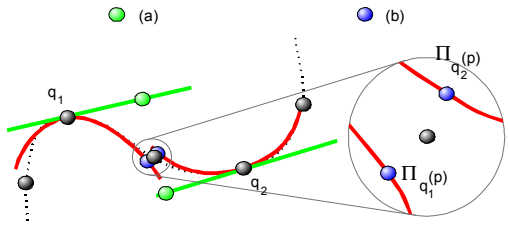


Figure 5: 2D illustration of point predictors using jets. Predictors using jets (b) are closer to the underlying surface (in dashed black) than predictors using planes (a) when the surface is curved.

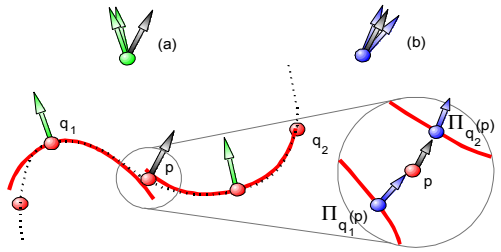


Figure 6: 2D illustration of normal predictors using jets. Normal predictors using jets (b) are better than the normals of neighboring jets (a) for filtering in curved regions. In this illustration, normals of the neighboring jets (q_1 and q_2) are poor estimators for the normals at point p.

3.3 Curvature Tensor

Curvatures on a surface are defined in the tangent plane as two vectors and magnitudes. Filtering two curvature magnitudes from two different curvature directions does not make sense. In order to filter curvatures, they must be reformulated in a frame-independent manner. Similar to the 2D curvature tensor, we rewrite our height function as a frame-independent equation using a 3D curvature tensor Γ. Let p be a point on the surface, taken from the origin of the local frame, and n the normal. The height function is thus given by:

$$n \cdot p = p \cdot \Gamma \cdot p \tag{8}$$

In the local frame of the Monge form, the normal is the $\mathbf{e}_z$ axis, and the curvature tensor a diagonal matrix with the two curvature magnitudes at x and y, and 0 at z. This equation can be used in any frame, the normal and curvature tensor being expressed in the frame of the jet. We thus have a frame-independent quantity which represents the curvature of the surface: the 3D curvature tensor Γ, with Q the frame transformation matrix.

$$\begin{aligned} \Gamma &= Q^t \begin{bmatrix} \frac{1}{2}c_x & 0 & 0 \\ 0 & \frac{1}{2}c_y & 0 \\ 0 & 0 & 0 \end{bmatrix} Q \\ \mathbf{n} &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} Q \end{aligned} \tag{9}$$

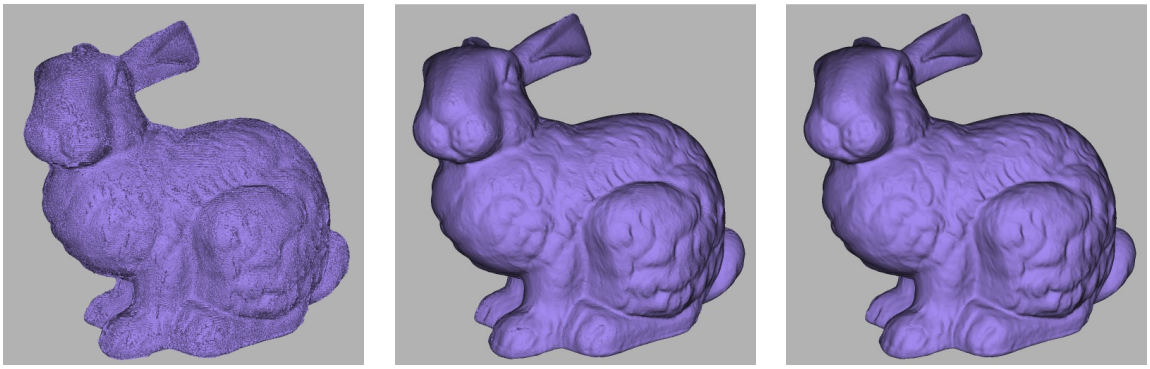
We filter the curvatures using the 3D curvature tensor in the bilateral filter (Eq. 5). The influence distance is the Euclidean distance between the tensors as 9D vectors. The tensor is considered as a regular 9D vector, the predictor is the tensor itself, and all linear computations can apply. However, the filtered tensor is modified to conform as a 3D curvature tensor. We cannot recover the normal from the curvature tensor using an eigensystem since the result is unpredictable for cylindrical or planar regions, and is highly prone to numerical precision errors. We thus use the previously estimated normal, and extract the closest tangent vectors which represent the principal curvature directions. Curvature magnitudes are then re-evaluated from the tensor. A final filtered curvature tensor is rebuilt using these values. Results of curvature filtering are shown Figure 4.

4 Applications and results

Our higher-order bilateral filtering method can be directly applied to achieve high-quality feature-preserving filtering on original 3D point data, and also effect feature enhancement. In addition, we show that the result of the filtering process allows direct meshing of noisy data, without the limitations of previous methods. We also enhance features of the models with an additional modification of the filter, by a penalization of high curvature regions.

4.1 Feature-preserving filtering

As described earlier, our process can have two passes, one for raw, noisy data, and a second for less noisy data, which may be the output of the first pass.



(a) (b) (c)

Figure 7: Results: (a) a partial triangulation of the original point set, (b) a triangulation of the point set after the first filtering pass, (c) after a second filtering pass.

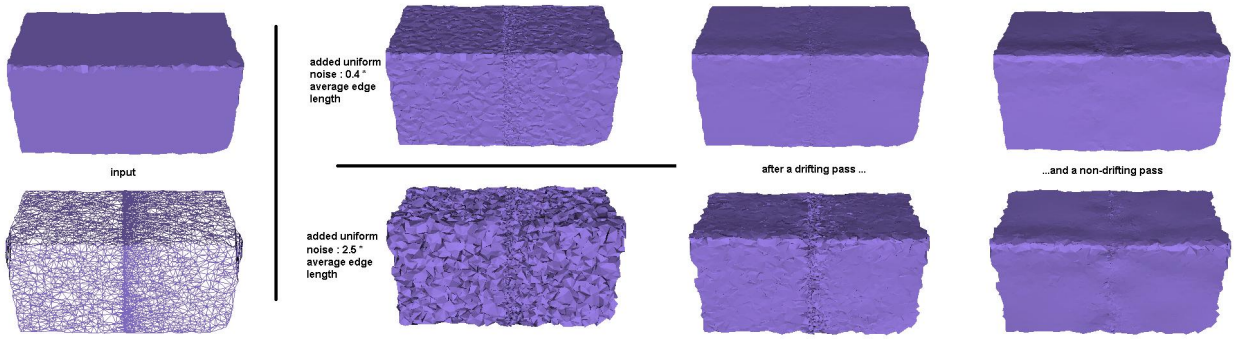


Figure 8: Example of filtering on an edge with irregular sampling. On the left, the edge (top is the flat shaded mesh, and bottom is the wireframe). Then artificial noise is added to this point set, which is then remeshed and rendered using flat shading for display. The top row illustrates the results for a uniform noise of amplitude $2/5$ of the average edge length. The bottom row illustrates the results for a uniform noise of amplitude $5/2$ of average edge length.

We applied our smoothing algorithm on several data sets. We applied it in particular to the bunny data set from *The Stanford 3D Scanning Repository*, made of approximately 360,000 points. These points are the aligned scanner data points before any reconstruction operations. After a first jet estimate, we applied a bilateral filter of curvature, normals, and points using drifting techniques, with a first result. We then reprocessed the data using non-drifting techniques, and obtained a second result. It it interesting to note that the average distance between the original points (aligned output of the scanner), and the final points is of the order of magnitude of the scanner's precision.

Existing methods ([BBX95, ABK98, ACK01] and [JDB02, Cha03]) can mesh noisy data, but often result in topological inconsistencies. Given that our approach filters the points directly, we choose to first smooth the data and then create the mesh, which resolved a number of these problems. In addition, meshing smoothed data is often faster in practice since we avoid the pathological cases of meshing approaches which often slow down execution. We used Geometry Factory software [Geo] to triangulate the point sets, see Figure 7 and 8 for results. The noisy point set was triangulated using the method of Chaine et al. [Cha03], since it provided a better result.

We implemented our algorithms on a Pentium 4, 2GHz, using the GNU Scientific Library for eigenanalysis and least squares minimization. Timings are given for $k = 25$ neighbors. Jet estimation (with four steps of local frame refinement) requires 2.1ms per jet; Curvature filtering 0.67ms; Normals filtering 0.63ms and Point filtering 0.64ms. See Table 1 for results.

In Figures 13 and 12, we compare our results to results of Jones et al. [JDD03]. We used the same models as input and compared to the output available on Jones's web page. We also illustrate the influence of the curvature tensor smoothing. We validated our results using Metro [CRS98] and measured the error with respect to the original geometry. The output of our technique has an RMS error 2.5 lower than that of Jones et al.

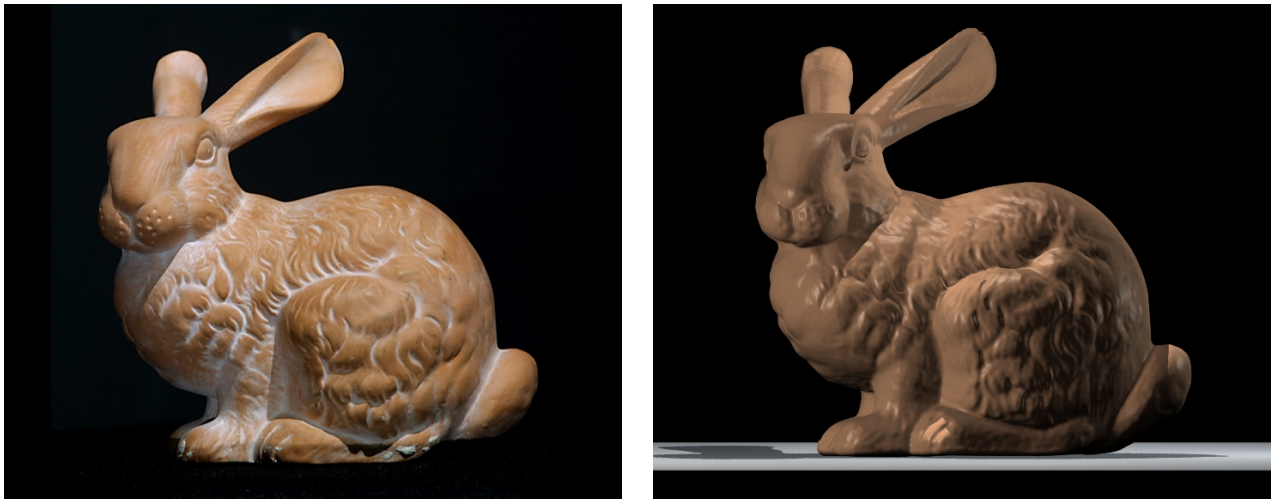


Figure 9: The Stanford Bunny. Left: a Photograph (from http://graphics.stanford.edu/data/3Dscanrep/); Right: a rendering of the reconstructed mesh.

Model	size	estim.	Γ	$\vec{n}$	p
Femme	40k	86s	26s	22s	22s
Dragon head	100k	210s	65s	57s	61s
Igea	135k	296s	82s	79s	76s
Bunny	360k	760s	243s	228s	232s

Table 1: Timings of our implementation. The machine used is a Pentium 4, 2.0 GHz with 2Gb RAM. The columns are the model name, model size in points, the time for jet estimation, for curvature tensor filtering, normal filtering and point filtering. Timings do not include precomputation of $k = 25$ closest neighbors finding. Those precomputation are about tens of seconds in our unoptimized implementation.

4.2 Shock Formation Control

Another application of our technique is a more aggressive filtering by curvature penalization. Our goal is to smooth parts of the model where high curvature regions are present. Adding a curvature constraint of a surface leads to shock formation, but in our approach, we ensure that the shock appears around features. We penalize high curvature regions of the data set in a manner similar to our previously defined quality parameter. The usage of the quality parameter in our filter penalized jets when they were bad approximations of the surface, typically around sharp features. The idea of curvature penalization is quite similar: we introduce a new coefficient in the filter so that jets of high curvature have a low weight for filtering. Then, points will rather align to flat regions and high curvatures will become sharp edges, or corners.

The parameter we propose to reflect this behaviour is the following:

$$s = tr(\Gamma^2) < \hat{d} >^2 \tag{10}$$

where tr is the trace (The trace of a matrix is the sum of its diagonal elements), Γ is the curvature tensor and $< \hat{d} >$ is the sampling distance.

This parameter is dimensionless, it does not change while the model is scaled, but if the sampling density is increased on the same model, this parameter decreases. The weight we use for the curvature penalization filter is

$$w_\Gamma = e^{-s * s_0} \tag{11}$$

where s_0 is a user-defined threshold. If s_0 is zero, then the weight is always one and no curvature penalization occurs. While increasing s_0, the high curvature jets have a lower weight in the filtering. We do this filtering iteratively increasing s_0 with small steps. We use the non-drifting version of the filter to perform these iterations. Figure 10 illustrates the shock formation in 2D. Results in 3D are shown on the Igea model on Figure 11.

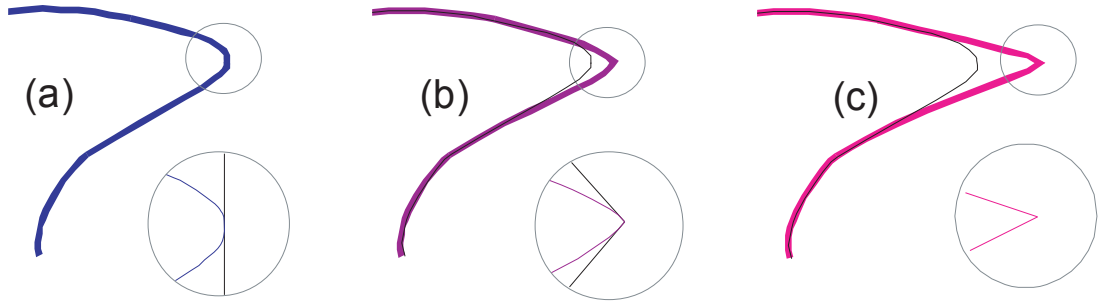


Figure 10: Shock Formation. In (a), the curve is smooth, in (b) it is created and in (c) the shock has been enhanced.

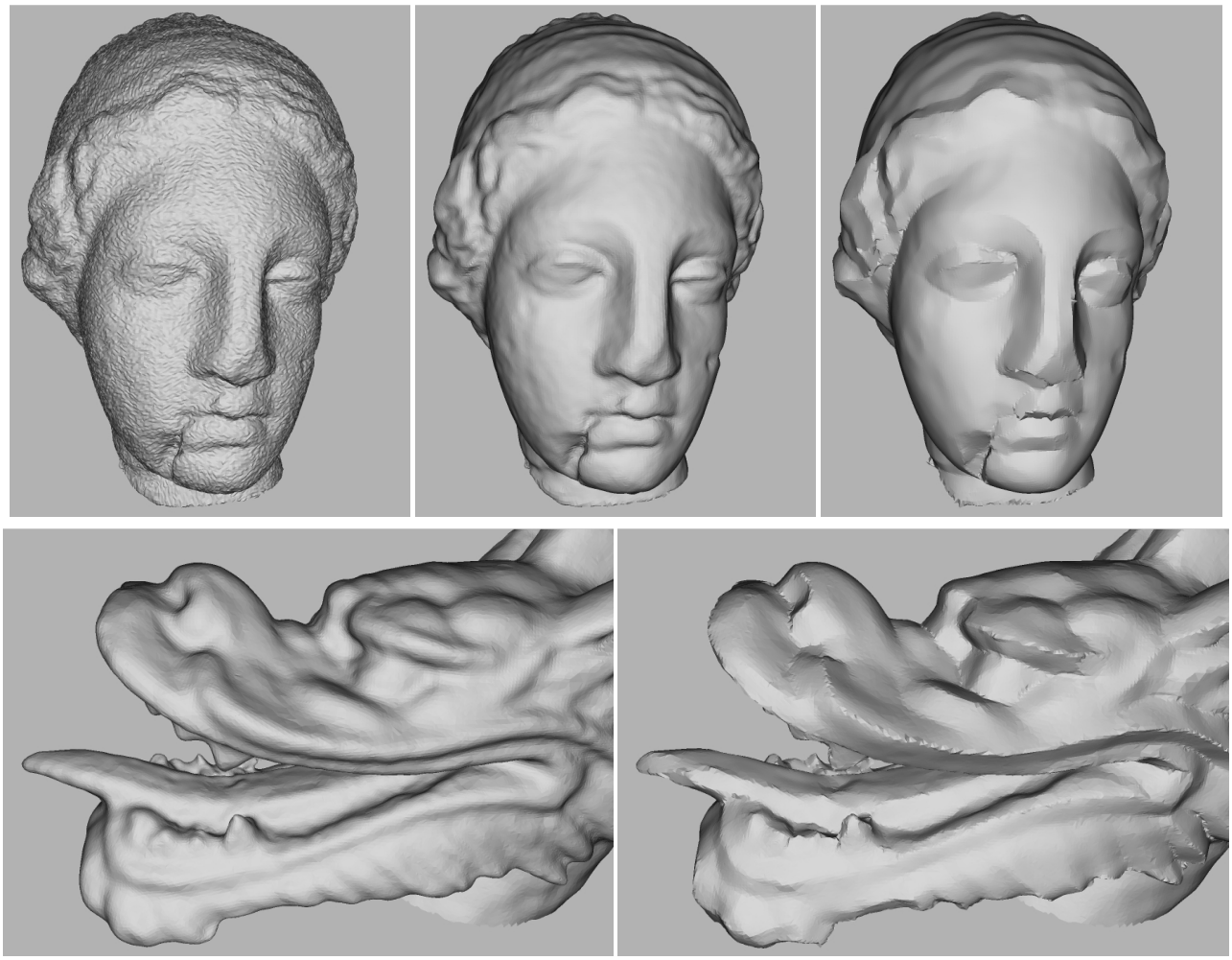


Figure 11: Top: Illustration of curvature penalization on the Igea model: on the left, the noisy Igea model, in the middle, the denoised Igea model, and on the right curved regions have been penalized and sharp edges have been created, smoothing further flat regions. We performed 20 iterations of smoothing with s_0 from 0 to 400. Bottom: Illustration on the dragon head: on the left the smoothed model, on the right, sharp edges appear. We performed 10 iterations of smoothing with s_0 from 0 to 200

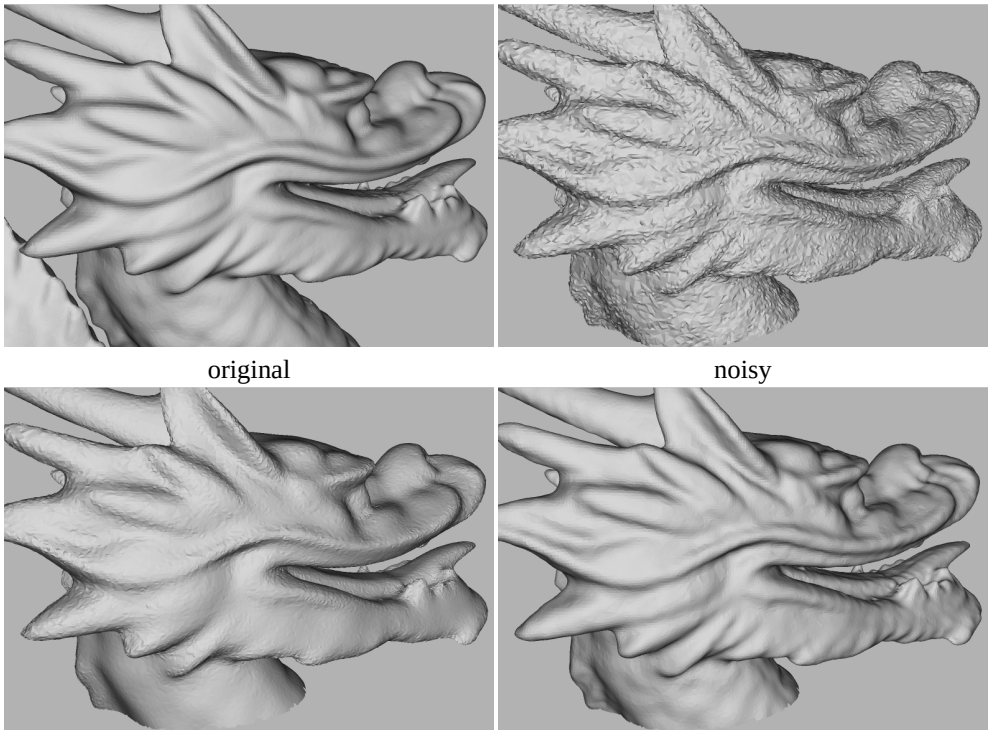


Figure 12: Comparison between our filter and the filter of Jones et al. 2003. Jones at al. used $\sigma_f = 4$, $\sigma_g = 1$, we used $\sigma_f = 5$, $\sigma_g = 1$.

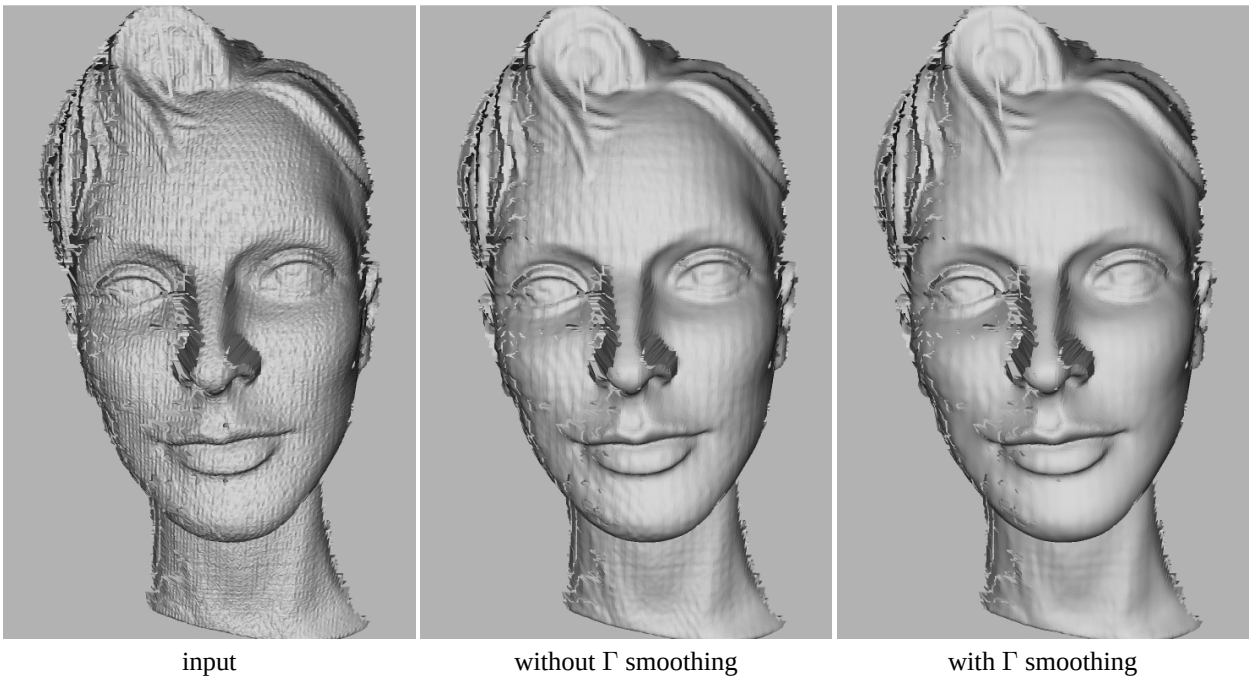


Figure 13: The femme model smoothed with our technique without and with curvature smoothing.

5 Conclusion

In this paper, we presented a new feature-preserving filter that leverages curvature to faithfully capture local shape. The second order predictors we use afford high-quality local surface estimators. Combined with our extension to the bilateral

filter, these jet predictors allow for robust estimation of geometry and differential properties. We introduced a new weight parameter based on the quality of our surface estimation. We also derived several filters with different behaviours and purposes, one closer to a denoising filter, another for fine tuning of surface smoothness. We introduced a new weight parameter in the bilateral filter, permitting curvature penalization and thus shock formation control around high curvature regions of the input model. We have shown the results of our algorithms on several models, illustrating the high-quality filtering achieved by our new approach. All these processing algorithms do not require connectivity, alleviating the fastidious process of meshing noisy point data. We believe that this new filter can facilitate modeling of geometries in a point or mesh representation, and lead to the development of a feature-aware editing tool.

6 Acknowledgements

This work has been possible thanks to MIT-France which supported the exchange of the first author at MIT. The authors wish to thank Matthias Zwicker and Ray Jones for early comments on the project and Matthieu Desbrun for the reading of an early draft. The rendering of the reconstructed bunny and the video have been made by Alexandre Olivier-Mangon, using Maya thanks to a donation of Alias—Wavefront.

References

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics 9*, 1 (Jan-Mar 2003), 3–15.

[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: A new voronoi-based surface reconstruction algorithm. In *Proceedings of SIGGRAPH 98* (July 1998), Computer Graphics Proceedings, Annual Conference Series, pp. 415–422.

[ACK01] AMENTA A. B., CHOI S., KOLLURI R. K.: The power crust, unions of balls, and the medial axis transform. *Computational Geometry Theory & Applications 19*, 2–3 (Jul 2001), 127–153.

[Ale02] ALEXA M.: Wiener Filtering of Meshes. In *Proceedings of Shape Modeling International* (2002), pp. 51–57.

[BBX95] BAJAJ C. L., BERNARDINI F., XU G.: Automatic reconstruction of surfaces and scalar fields from 3d scans. In *Proceedings of SIGGRAPH 95* (Aug. 1995), Computer Graphics Proceedings, Annual Conference Series, pp. 109–118.

[BO01] BELYAEV A., OHTAKE Y.: Nonlinear Diffusion of Normals for Crease Enhancement. In *Vision Geometry X, SPIE Annual Meeting* (2001), pp. 42–47.

[BR02] BERNARDINI F., RUSHMEIER H.: The 3d model acquisition pipeline. *Computer Graphics Forum 21*, 2 (2002), 149–172.

[BX03] BAJAJ C., XU G.: Anisotropic diffusion on surfaces and functions on surfaces. *ACM Transactions on Graphics 22*, 1 (January 2003), 4–32.

[CDR00] CLARENZ U., DIEWALD U., RUMPF M.: Anisotropic geometric diffusion in surface processing. In *IEEE Visualization 2000* (October 2000), pp. 397–405.

[Cha03] CHAINE R.: A geometric convection approach of 3-d reconstruction. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing* (2003), Eurographics Association, pp. 218–229.

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

[CRS98] CIGNONI P., ROCCHINI C., SCOPIGNO R.: Metro: Measuring error on simplified surfaces. *Computer Graphics Forum 17*, 2 (June 1998), ??–??

[CT03] CHOUDHURY P., TUMBLIN J.: The trilateral filter for high contrast images and meshes. In *Eurographics Symposium on Rendering: 14th Eurographics Workshop on Rendering* (June 2003), pp. 186–196.

[DD02] DURAND F., DORSEY J.: Fast bilateral filtering for the display of high-dynamic-range images. *ACM Transactions on Graphics 21*, 3 (2002), 257–266.

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of SIGGRAPH 99* (August 1999), pp. 317–324.

[DMSB00] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Anisotropic feature-preserving denoising of height fields and bivariate data. In *Graphics Interface* (2000), pp. 145–152.

[FDCO03] FLEISHMAN S., DRORI I., COHEN-OR D.: Bilateral mesh denoising. *ACM Transactions on Graphics 22*, 3 (July 2003), 950–953.

[Geo] GEOMETRYFACTORY:. http://www.geometryfactory.com/.

[GKS00] GOPIY M., KRISHNAN S., SILVA C. T.: Surface reconstruction based on lower dimensional localized delaunay triangulation. *Computer Graphics Forum 19*, 3 (Aug. 2000).

[HDD*92] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLE W.: Surface reconstruction from unorganized points. In *Computer Graphics (Proceedings of SIGGRAPH 92)* (July 1992), vol. 26,2, pp. 71–78.

[JDB02] J-D. BOISSONNAT F. C.: Smooth surface reconstruction via natural neighbour interpolation of distance functions. In *Computational Geometry - Theory and Application* (2002), vol. 22,1. Extended version from ACM Computational Geometry, 2000.

[JDD03] JONES T. R., DURAND F., DESBRUN M.: Non-iterative, feature-preserving mesh smoothing. *ACM Transactions on Graphics 22*, 3 (July 2003), 943–949.

[Jon03] JONES T. R.: *Feature Preserving Smoothing of 3D Surface Scans*. Master's thesis, MIT, 2003.

[KV01] KALAIAH A., VARSHNEY A.: Differential point rendering. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering* (June 2001), pp. 139–150.

[LPC*00] LEVOY M., PULLI K., CURLESS B., RUSINKIEWICZ S., KOLLER D., PEREIRA L., GINZTON M., ANDERSON S., DAVIS J., GINSBERG J., SHADE J., FULK D.: The digital michelangelo project: 3d scanning of large statues. In *Proceedings of ACM SIGGRAPH 2000* (July 2000), Computer Graphics Proceedings, Annual Conference Series, pp. 131–144.

[MDPS02] MEYER M., DESBRUN M., PETER SCHRÖDER A. H. B.: Discrete differential-geometry operators for triangulated 2-manifolds. In *VisMath '02, Berlin* (2002).

[Mea00] MEDIONI G., ET AL.: Tensor voting: Theory and applications, 2000.

[MN03] MITRA N. J., NGUYEN A.: Estimating surface normals in noisy point cloud data. In *Proceedings of the nineteenth conference on Computational geometry* (2003), ACM Press, pp. 322–328.

[OBS02] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: Mesh smoothing by adaptive and anisotropic gaussian filter applied to mesh normal. In *Vision, Modeling and Visualization (Erlangen, Germany)* (2002).

[PG01] PAULY M., GROSS M.: Spectral processing of point-sampled geometry. In *Proceedings of ACM SIGGRAPH 2001* (August 2001), pp. 379–386.

[PKKG03] PAULY M., KEISER R., KOBBELT L. P., GROSS M.: Shape modeling with point-sampled geometry. vol. 22, ACM Press, pp. 641–650.

[PM90] PERONA P., MALIK J.: Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence 12*, 7 (1990), 629–639.

[PSZ01] PENG J., STRELA V., ZORIN D.: *A Simple Algorithm for Surface Denoising*. Tech. rep., New York University, 2001.

[Tau95a] TAUBIN G.: Estimating the tensor of curvature of a surface from a polyhedral approximation. In *ICCV 1995* (1995).

[Tau95b] TAUBIN G.: A signal processing approach to fair surface design. In *Proceedings of SIGGRAPH 95* (August 1995), pp. 351–358.

[Tau01] TAUBIN G.: *Linear Anisotropic Mesh Filtering*. Tech. Rep. IBM Research Report RC2213, IBM, 2001.

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Proc. IEEE Int. Conf. on Computer Vision '98* (1998), pp. 836–846.

[TWBO02] TASDIZEN T., WHITAKER R., BURCHARD P., OSHER S.: Geometric surface smoothing via anisotropic diffusion of normals. In *Proceedings, IEEE Visualization 2002* (2002), pp. 125–132.

[ZF02] ZHANG H., FIUME E. L.: Mesh Smoothing with Shape or Feature Preservation. In *Advances in Modeling, Animation, and Rendering 2002, J. Vince and R. Earnshaw, editors* (2002), pp. 167–182.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399