Vision, Modeling, and Visualization (2011)
Peter Eisert, Konrad Polthier, and Joachim Hornegger (Eds.)

SBL Mesh Filter: A Fast Separable Approximation of Bilateral Mesh Filtering

Guillaume Vialaneix[1,2] and Tamy Boubekeur[2,3]

[1]EDF R&D, MFEE Department
[2]Telecom ParisTech
[3]CNRS-LTCI

Abstract

Bilateral mesh filtering is a simple and powerful feature-preserving filtering operator which allows to smooth or remove noise from surface meshes while preserving important features in a non-iterative way. However, to be effective, such a filter may require quite a large support size, inducing slow processing when applied on high resolution meshes such as the ones produced by automatic 3D acquisition devices. In this paper, we propose a separable approximation of bilateral mesh filtering based on a local decomposition of the bi-dimensional filter into a product of two one-dimensional ones. In particular, we show that this approximation leads to piecewise smooth surfaces which are very close to the ones produced by the exact filter, using only a fraction of the usual required time. Compared to bilateral image filtering, the main problem here is to find meaningful directions at every point to orient the two one-dimensional filters. Our solution exploits the minimum and maximum curvature directions at each point and demonstrates a significant speed-up on meshes ranging from thousands to millions of elements, enabling feature-preserving filtering with large support size in a variety of practical scenarii. Our approach is simple, easy to implement and orthogonal to other kinds of optimizations such as higher dimensional clustering using a bilateral grid or a Gaussian kd-tree and can therefore be combined to them to reach even higher performance.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

1. Introduction

Surface meshes are ubiquitous in computer graphics applications and can be automatically acquired using various 3D scanning devices. Resulting shapes often suffer from a small scale high frequency noise which needs to be filtered before using the mesh. Mesh filtering methods (Section 2) – mostly inspired by classical signal processing techniques – help removing this shortcoming. However, special care has to be taken to preserve important feature lines. While earlier methods provided pure low-pass filters only, recent feature preserving filters have emerged as standard components in most mesh processing pipelines. Feature preserving filters all relate to some sort of similarity analysis. Given a surface vertex they establish which of the neighbors in the vertex' vicinity should have a stronger influence on its filtered position. Mimicking signal and image filtering, bilateral mesh filtering (BL) [JD03] appears as one of the most efficient and versatile solutions to the problem of feature-preserving mesh denoising.

Unfortunately, BL filtering often requires a large local support to effectively remove noise while preserving important features. In this paper, we propose an optimization scheme converting the exact BL filter into a separable approximation (SBL). Our technique is inspired by the separable bilateral image filter [PV05]. Unfortunately, this approach cannot be applied directly, as there is usually no explicit 2D global parameterization provided with raw 3D surface mesh. Therefore, we propose to separate BL filtering using smoothed minimum and maximum curvature directions to restrict the number of combined neighbors at each pass

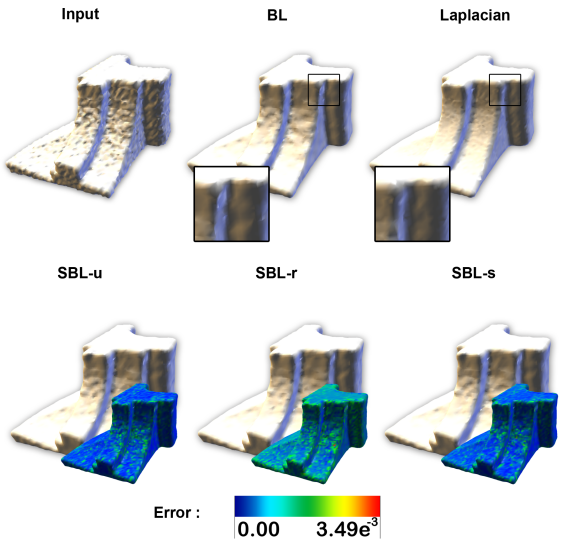

Figure 1: *Our approximation scheme provides similar results when compared to exact bilateral filtering while being several times faster. From top left to bottom right : a noisy mesh, bilateral filtering, Laplacian filtering, separable bilateral (SBL) filtering decomposed along unsmoothed curvature direction (SBL-u), SBL using random directions (SBL-r) and SBL using smoothed curvature directions (SBL-s). Error maps compare to exact BL filter and are given on a per-vertex basis using the L^2 norm.*

(Section 3). We show experimentally that this solution is superior to other choices in its approximation power, as shown in fig. 1, and that the resulting SBL filter is several times faster than the exact BL one while introducing a negligible error (Section 4).

2. Previous Work

Mesh Denoising. The seminal work of Taubin [Tau95] initiated signal processing for meshes by introducing low-pass filtering for mesh denoising. In particular, Taubin showed the link between low-pass signal filtering and mesh fairing. He introduced the Laplacian of a discrete surface signal as a generalization of the Fourier descriptors, leading to a linear-time algorithm for mesh smoothing.

Later, Desbrun et al. [DMSB99] introduced an implicit integration scheme for computing the diffusion equation on a mesh. They proposed a new definition of the Laplacian operator, as such a smoothing operator can be seen as a time integration of the heat equation. This allowed them to define the curvature flow which relies only on intrinsic geometric properties and offers a good alternative to the diffusion flow.

Donoho [Don95] presented a wavelet-based method computing a wavelet transform of the mesh and thresholding its coefficients before applying a reverse transform. Peng et al. [PSZ01] adopt a similar strategy, but use Wiener filter and Gaussian Scale Mixtures (GSM) as transform operators.

Also inspired by the vast repository of image denoising techniques, Yoshizawa et al. [YBS06] recently adapted the NL-means filter [BCM05] to meshes, by searching among all vertices those having the most similar neighborhoods with a given point to filter, giving them higher weights independently of their location.

Yagou and Ohtake [OBS02] [YOB02] [YOB03] proposed several techniques based upon a two-step scheme, consisting in smoothing the normals of faces, and then trying to make the vertices' positions to fit those normals.

For a recent overview of polygonal mesh smoothing methods, we refer the reader to the book by Botsch et al. [BKP*10]. In the following, we focus on the formulation of Jones and Durand [JD03].

Bilateral Mesh Filtering. Considering a noisy object (e.g. an image or a mesh), the simplest way to remove noise is to apply a low-pass (e.g. Gaussian) filter which replaces each sample (e.g. pixel or vertex) by a weighted average of its neighbors. For each sample, such a filter ignores the actual "values" (range) of the neighboring samples and considers only their distances in space to weight their contributions in a (local) combination. Consequently, noise is filtered out, but feature lines are proportionally blurred. The idea of bilateral filtering [TM98] is to introduce a second weighting term based on the difference in range between object samples (i.e. pixel color) to weight their relative contribution. In fact, when using Gaussian kernels, the bilateral filter can be seen as a single higher-dimensional Gaussian filter [PD09], applied in a space made of both image space and range dimensions. Note however that a range space based on pixel value differences is the simplest similarity space that can be imagined for pictures. More robust similarity comparisons have been later introduced, such as the ones based on local neighborhood around samples in the context of Non-Local Means filter [BCM05]. For a complete introduction to bilateral biltering, we refer the reader to the course by Paris et al. [PKTD08].

From a signal processing point of view, the main difference between an image and a surface mesh is that the former has a trivial global space parameterization (i.e. pixel coordinates) which is decorralated from the range of the signal function (i.e. pixel color) while for the latter, the vertex 3D position embeds in general both signal and parameterization, i.e. space and range values.

Several strategies have been proposed to apply bilateral filtering on meshes [JD03] [FDCO03] and point-sampled surfaces [JDZ04]. Although not restricted to this particular form, we focus on Jones and Durand formulation [JD03].

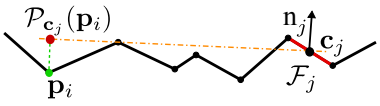

Figure 2: *The projection $\mathcal{P}_{\mathbf{c}_j}(\mathbf{p}_i)$ of a vertex $\mathbf{p}_i$ according to its neighboring face $\mathcal{F}_j$. The weight of $\mathcal{F}_j$ with respect to $\mathbf{p}_i$ is given by the L^2 norm $\|\mathbf{p}_i - \mathcal{P}_{\mathbf{c}_j}(\mathbf{p}_i)\|$.*

They define a range space for meshes by the mean of predictions computed as the projection of the vertex onto the tangent plane of its neighbors. This process is depicted in Figure 2.

Considering a mesh $\mathcal{M}$, and one of its vertices $\mathbf{p}_i$, then for each of the faces $\mathcal{F}_j$ in its neighborhood $\mathcal{N}_i$, we can compute the projection of $\mathbf{p}_i$ on the plane defined by the center $\mathbf{c}_j$ and the *smoothed* normal $\mathbf{n}_j$ of $\mathcal{F}_j$

$$\mathcal{P}_{\mathbf{c}_j}(\mathbf{p}_i) = \mathbf{p}_i + \mathbf{n}_j((\mathbf{c}_j - \mathbf{p}_i)\cdot\mathbf{n}_j) \quad (1)$$

Using a spatial kernel G_{σ_s} and a range kernel G_{σ_r}, a vertex $\mathbf{p}_i$ is filtered according to the neighboring faces set $\mathcal{N}_i$ by

$$BL_{\mathcal{N}_i}(\mathbf{p}_i) = \frac{1}{W(p)} \sum_{\mathcal{F}_j \in \mathcal{N}_i} a_j G_{\sigma_s}(d_{ij}) G_{\sigma_r}(h_{ij})\, \mathcal{P}_{\mathbf{c}_j}(\mathbf{p}_i) \quad (2)$$

with

$$W(p) = \sum_{\mathcal{F}_j \in \mathcal{N}_i} a_j G_{\sigma_s}(d_{ij}) G_{\sigma_r}(h_{ij}) \quad (3)$$

a_j the area of the face j, $d_{ij} = \|\mathbf{p}_i - \mathbf{c}_j\|$ and $h_{ij} = \|\mathbf{p}_i - \mathcal{P}_{\mathbf{c}_j}(\mathbf{p}_i)\|$. Note that robustness is improved by first smoothing face normals using a spatial kernel $G_{\sigma_s/2}$ [JD03]. The set $\mathcal{N}_i$ can be collected via a mesh flood filling process starting at $\mathbf{p}_i$ and selecting all faces located within a distance σ_s.

In our implementation, we use Gaussian kernels

$$G_\sigma(x) = e^{\frac{-x^2}{2\sigma^2}} \quad (4)$$

Although local, bilateral mesh filtering remains a slow process and different acceleration techniques have been proposed. In particular, the bilateral grid [CPD07] has been introduced for for images. The basic idea is to embed the image into a three-dimensional space with the third dimension corresponding to intensity before using the so-defined 3D distance to evaluate combination weights. More recently, Adams et al. [AGDL09] improved this method by replacing the grid with a kd-tree, accelerating the sampling and interpolation phases of the bilateral grid technique. These methods can be applied to bilateral mesh filtering using a higher number of dimensions.

We explore an orthogonal approach to BL mesh filtering acceleration, which can therefore be combined with these higher dimensional ones for even faster processing.

3. Separable Bilateral Mesh Filtering

The key idea of our approximation model is to speed-up the BL filter by reducing the size of $\mathcal{N}_i$ while still covering the same support size. More precisely, our approximation works in two passes: in the first pass we first collect a set of neighboring faces restricted to one tangent direction on the surface and then filter the vertex using this reduced set only. This first pass is applied to all vertices. In the second pass, we filter the output of the first pass using an orthogonal tangent direction. This approach is inspired by the classical separable Gaussian filter for images. Note however that while the exact solution is reproduced in the case of Gaussian filtering, such a decomposition leads only to an approximate BL filter [PV05].

Therefore, considering the case of meshes, the main question is to determine two directions at each vertex to collect the restricted sets. Our key observation is that using the minimum and maximum curvature directions yields a consistent way to decompose the BL filter according to the local surface anisotropy, capturing the feature orientation along which approximation artifacts are better hidden.

Thus, we compute minimum and maximum curvature directions $\{\mathbf{u}_i, \mathbf{v}_i\}$ at every vertex $\mathbf{p}_i$ of $\mathcal{M}$ to define locally the filtering direction for our two passes. We use the Rusinkiewicz estimator [Rus04] for curvature directions. This estimator uses the second fundamental tensor on the tangent plane to obtain constraints on the normal derivative, fitted in the least squares sense and averaged over faces. As noise perturbates most mesh curvatures estimators, we smooth the curvature direction field using the kernel $G_{\sigma_s/2}$. In practice, this operation is performed simultaneously with the normal smoothing preprocess (Section 2).

Using these curvature direction we define two orthogonal planes $\Pi^i_{\mathbf{u}} = \{\mathbf{n}_i, \mathbf{u}_i\}$ and $\Pi^i_{\mathbf{v}} = \{\mathbf{n}_i, \mathbf{v}_i\}$, one for each curvature direction and both intersecting along the vertex normal. These planes intersect the neighboring faces and offer a straightforward predicate to collect the restricted set of faces

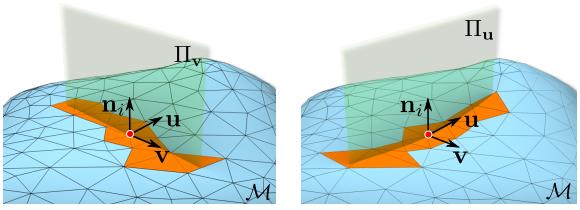

Figure 3: *Two local planes $\Pi^i_{\mathbf{u}}$ and $\Pi^i_{\mathbf{v}}$ (green), allow to query for two restricted sets of neighborhing faces (orange), one for each pass, which are aligned with curvature directions.*

for each pass. We illustrate these planes and these sets on Figure 3.

The restricted sets of neighborhing faces related to each pass are defined as the intersection of the full ball-neighborhood $\mathcal{N}_i$ with the planes

$$\mathcal{N}_i^{\mathbf{u}} = \left\{ \mathcal{F}_j \in \mathcal{N}_i \quad | \quad \mathcal{F}_j \cap \Pi_{\mathbf{u}}^i \neq \varnothing \right\} \tag{5}$$

We compute $\mathcal{N}_i^{\mathbf{v}}$ similarly and in both cases we use a restricted flood filling process implemented using a queue where only faces intersecting the current plane are pushed up to the maximum distance σ_s.

Finally, our SBL approximation consists of the combination of two restricted BL filterings:

$$SBL(\mathbf{p}_i) = BL_{\mathcal{N}_i^{\mathbf{u}}}\left(BL_{\mathcal{N}_i^{\mathbf{v}}}(\mathbf{p}_i)\right) \approx BL_{\mathcal{N}_i}(\mathbf{p}_i) \tag{6}$$

Variations Overall, selecting smooth curvature directions yields better approximation results consistently when compared to using directly curvature directions or random pairs of orthogonal directions at each vertex. In the following, we refer to Table 1 for the different variants of our algorithm.

algorithm	$\{\mathbf{u},\mathbf{v}\}$
SBL-r	random pairs of orthogonal tangents
SBL-u	curvatures directions
SBL-s	smoothed curvatures directions

Table 1: *The different variations of the SBL filter.*

4. Results and Discussion

In this section, we analyze the performance improvement offered by the SBL filter and the error introduced when compared to the ground thruth BL filter. We also compare the three different variations of the SBL filter with respect to their approximation accuracy. Performance is measured on an Intel Core 2 Duo, 2.53GHz with 4.0 GB of RAM, single-threaded. All filters supporting sizes are expressed relatively to the object bounding box diagonal.

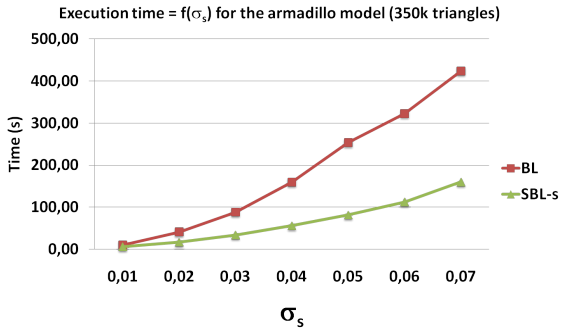

Figure 4: *Comparison between BL and SBL-s filtering w.r.t. σ_s on the Armadillo model.*

Mesh	Faces	BL (s)	SBL-s (s)	RMS error
sphere	20k	1.0	0.7	$8.3 \cdot 10^{-4}$
armadillo	350k	95	38	$3.4 \cdot 10^{-4}$
filigree	1M	647	152	$4.7 \cdot 10^{-4}$
Red_box	1.4M	600	193	$3.6 \cdot 10^{-4}$

Table 2: *Error and timing for different meshes ($\sigma_s = 0.03$ and $\sigma_r = 0.03$). The RMS error measures the difference per-vertex between exact BL and our SBL-s approximation. Meshes are normalized to the unit cube.*

In Figure 4, we measure the execution time of several BL and SBL-s filtering processes with growing support sizes σ_s. We can observe that the SBL-s filter remains between 2 and 3 times faster than the BL filter on this example, with a larger support size inducing a higher acceleration factor. Note that these timings include the normal filtering process which cannot be avoided for robustness reasons. The curvature filtering process used for the SBL-s algorithm is time-consuming, with total execution times for SBL-r and SBL-u variants being about twice as fast. Reducing its support size allows to trade speed for accuracy.

In Figure. 5, we illustrate the per-vertex error introduced by both BL and SBL-s filtering when applied to an artificially noisy model (i.e., for which we can measure error to ground truth). Qualitatively, one can see that our filter is almost as good as the BL filter we try to approximate. Figure. 6 confirms that differences between the two denoised results (BL and SBL) are very small.

In Figure 7 we plot the distribution of errors induced by the three variants of our approximation scheme when compared to exact BL mesh filtering. First of all, the global error is overall small in the three cases and its distributions re-

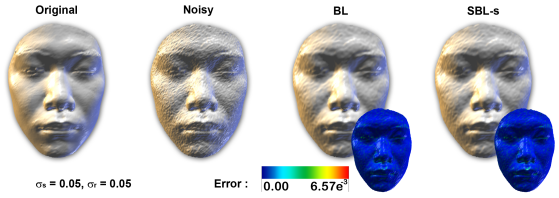

Figure 5: *Comparison between BL and SBL-s filtering power. From left to right: original surface, same surface with added noise (40% of the mean edge size), BL denoising and our SBL-s denoising.*

stricted near small amplitudes. Second, we can observe how our decomposition strategy based on smoothed main curvature directions yields a significant better result than the two other variants. Surprisingly enough, the use of random directions to collect the restricted face set is not particularly less accurate than using non-smoothed directions.

Table 2 gives timing and RMS error between exact BL and SBL-s filtering on four different models, some of which are illustrated in Figure 9. We can observe that the error remains particularly low even for a quite large filter support size (5% of the object size) while the process is sped up by a factor ranging from 2 to 3. Based on all our experiments, we observe that the RMS error is each time inferior to 10^{-3}, considering a unit mesh. When decomposing this error into normal and tangential components, it appears that the later dominates significantly (Figure 8).

Finally, Figure 9 shows the final meshes obtained with the SBL filters. All meshes are scaled to a unit box. We plot the approximation error between the exact BL filter and the three variants of the SBL approximation on a per-vertex basis. Note that even the maximum error measured on each mesh remains low w.r.t. to the object size. Analyzing these different measures, we can conclude that although not formally separable, our separable approximation of bilateral mesh filtering is very close to the exact solution.

The particular choice of smooth curvature directions for the filter decomposition demonstrates to be superior to other variants in terms of accuracy. The reason is that approximation errors are distributed along the separation axis and therefore better hidden when these are aligned with anisotropic surface features, i.e. main curvature directions. However it is important to point out that SBL-r and SBL-u filters are about twice faster than the SBL-s filter, as the curvature direction smoothing operation — although performed on a much smaller support size than the SBL filter itself — remains expensive. Still, this also provides a simple quality-versus-speed parameter and setting it to zero boils down to the SBL-u filter.

Figure 6: *Histograms of the per-vertex error between the filtered meshes (BL and SBL) and the original one. Differences between the two curves are not easy to detect for a large range, which demonstrates the good approximation of our method.*

5. Conclusion

We have introduced a separable approximation of bilateral mesh filtering which is simple and easy to implement. By decomposing the filter into two successive filters with smaller support sizes which are applied along smoothed curvature directions only, we obtained smooth meshes with preserved features which are visually equivalent to the result provided by the exact filter, while being computed several time faster. Our approximation model can be intuitively controlled by setting the support size of the curvature directions filtering process, trading quality for speed.

Our acceleration technique is orthogonal to high dimensional embedding methods and can therefore be combined with them. Such techniques enable fast neighbors contribution quieries, while our SBL fitler reduces the number of neighbors to consider. In praticular, we plan to experiment combinations of SBL filtering with the bilateral grid [CPD07] and the gaussian kd-tree [AGDL09] methods.

Finally, a similar strategy may be applied for other classes of feature preserving filters, such as NL-Means. We will also explore this direction in future work.

References

[AGDL09] ADAMS A., GELFAND N., DOLSON J., LEVOY M.: Gaussian kd-trees for fast high-dimensional filtering. In *ACM SIGGRAPH 2009 papers* (New York, NY, USA, 2009), SIGGRAPH '09, ACM, pp. 21:1–21:12.

Figure 7: *Error histograms for the different versions of our SBL filter (w.r.t the exact BL filter). These distributions were computed for the* `Red_circular_box` *model, with* $\sigma_s = 0.05$ *and* $\sigma_r = 0.02$.

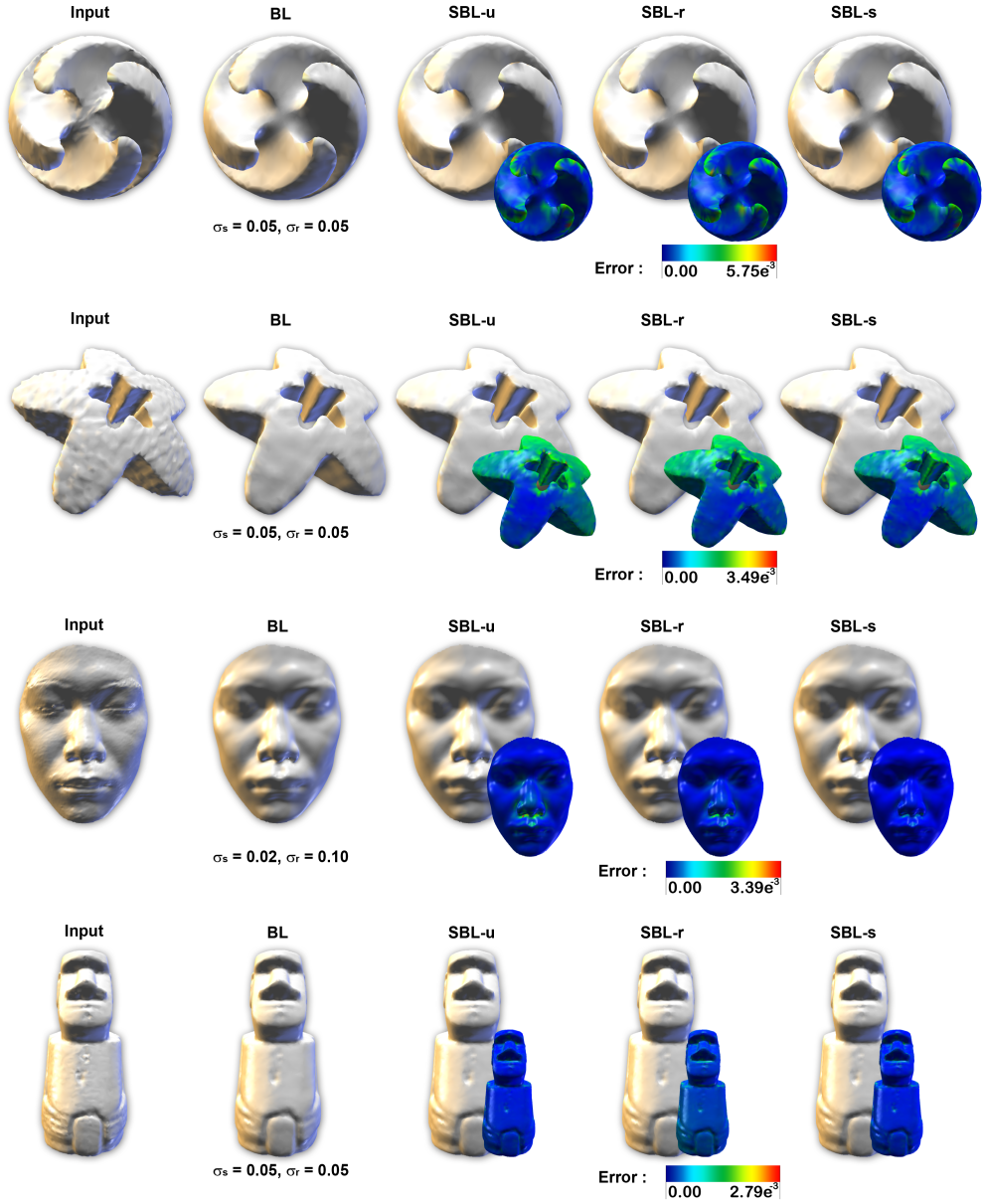

Figure 9: *The results of our different algorithms on different models. From left to right : the input model, the result of the BL filtering, the SBL-u variant (the two directions come from curvature directions computation, without any smoothing), the SBL-r technique (the two orthogonal directions are choosen randomly for each point), the SBL-s method (the two directions come from curvature directions computation, with previous smoothing on those curvatures). The indicated error is the per-vertex L^2-norm of the distance between the BL result and our algorithms, with respect to the model diagonal.*

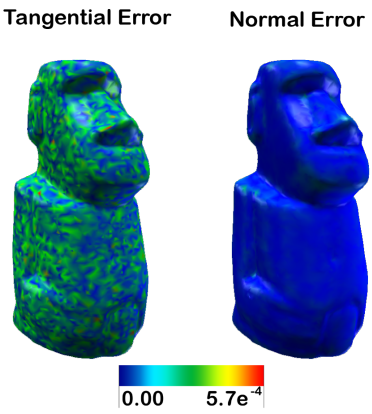

Figure 8: *Per-vertex L^2 error decomposed into normal and tangential components. The small normal component explains the visual fidelity of our approximation.*

[BCM05] BUADES A., COLL B., MOREL J.-M.: A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 2, pp. 60 – 65.

[BKP*10] BOTSCH M., KOBBELT L., PAULY M., ALLIEZ P., LEVY B.: *Polygon Mesh Processing*. AK Peters, 2010.

[CPD07] CHEN J., PARIS S., DURAND F.: Real-time edge-aware image processing with the bilateral grid. In *ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM.

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), SIGGRAPH '99, pp. 317–324.

[Don95] DONOHO D.: De-noising by soft-thresholding. *Information Theory, IEEE Transactions on 41*, 3 (May 1995), 613 –627.

[FDCO03] FLEISHMAN S., DRORI I., COHEN-OR D.: Bilateral mesh denoising. In *ACM SIGGRAPH 2003 Papers* (2003), SIGGRAPH '03, pp. 950–953.

[JD03] JONES T. R., DURAND F.: Non-iterative, feature-preserving mesh smoothing. *ACM Transactions on Graphics 22* (2003), 943–949.

[JDZ04] JONES T. R., DURAND F., ZWICKER M.: Normal improvement for point rendering. *IEEE Comput. Graph. Appl. 24* (July 2004), 53–56.

[OBS02] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: Mesh smoothing by adaptive and anisotropic gaussian filter applied to mesh normals. In *VISION MODELING AND VISUALIZATION* (2002), pp. 203–2010.

[PD09] PARIS S., DURAND F.: A fast approximation of the bilateral filter using a signal processing approach. *Int. J. Comput. Vision 81* (January 2009), 24–52.

[PKTD08] PARIS S., KORNPROBST P., TUMBLIN J., DURAND F.: A gentle introduction to bilateral filtering and its applications. In *Class at ACM SIGGRAPH'08* (2008).

[PSZ01] PENG J., STRELA V., ZORIN D.: A simple algorithm for surface denoising. In *Proceedings of the conference on Visualization '01* (2001), VIS '01, pp. 107–112.

[PV05] PHAM T. Q., VLIET L. J.: Separable bilateral filtering for fast video preprocessing. In *In IEEE Internat. Conf. on Multimedia & Expo, CD1Ű4* (2005), IEEE, pp. 1–4.

[Rus04] RUSINKIEWICZ S.: Estimating curvatures and their derivatives on triangle meshes. In *Symposium on 3D Data Processing, Visualization, and Transmission* (Sept. 2004), pp. 486 – 493.

[Tau95] TAUBIN G.: A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), SIGGRAPH '95, pp. 351–358.

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision* (1998), ICCV '98, pp. 839–.

[YBS06] YOSHIZAWA S., BELYAEV A., SEIDEL H.-P.: Smoothing by example: Mesh denoising by averaging with similarity-based weights. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006* (2006), pp. 9–.

[YOB02] YAGOU H., OHTAKE Y., BELYAEV A.: Mesh smoothing via mean and median filtering applied to face normals. In *Proceedings of the Geometric Modeling and Processing — Theory and Applications (GMP'02)* (2002), GMP '02, pp. 124–.

[YOB03] YAGOU H., OHTAKE Y., BELYAEV A.: Mesh denoising via iterative alpha-trimming and nonlinear diffusion of normals with automatic thresholding. In *Computer Graphics International, 2003. Proceedings* (2003), pp. 28 – 33.