Shape smoothing with feature preserving weighted filters

Leon Kos[*]
University of Ljubljana, Slovenia

Jože Duhovnik
University of Ljubljana, Slovenia

Abstract

Several techniques for arbitrary shape recovery from scanned data attempt to recognize and further regularize shape. For arbitrary shape, one can recognize several shape features which should be guided with global shape parameters. Smoothed curves and surfaces created with subdivision can visually improve recovery when overall smoothness is expected. Local features such as sharp edges are not preserved during smoothing. In this paper we show procedural approach to preserve such features while globally smoothing shape. Weighted filters are applied according to local shape variations. For flat and smooth areas, weighted mean face is dominant. Sharp features are detected with normal difference variation and dominated by nearest face normal. After suggested face normals are calculaced, vertices are moved by simplified version of nonlinear diffusion. Performance of proposed method is compared with other methods for mesh smoothing and edge creasing on "CAD" and "media" models.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Hierarchy and geometric transformations

Keywords: triangular meshes, mesh smoothing, shape recovery, feature detection

1 Introduction

Triangle meshes are extensively used to represent 2-manifold meshes as the only reliable approximation of continuous surface. Scanners which produce large amount of point sets also introduce noise and sampling errors which make the reconstructed surface ragged. Large point sets are typically irregularly sampled and non uniformly distributed. Improvement of such scans is desired for many applications. Noise present in every acquisition method should be removed while preserving important features such as sharp edges and corners.

With large number of acquired points from 3D scanner, post-processing reduce dense areas of points to prescribed density and thus introduces artifacts on edges. On decimated and uniformly distributed meshes this is seen as high frequency noise in the position of the vertices. Denoising can be applied by just adjusting vertex positions. Such approach preserves connectivity and topology of the mesh if vertices are uniformly distributed over the mesh.

1.1 Related work

Many surface fairing methods have been proposed in recent years. One of the first signal processing application on meshes was introduced by [Taubin 1995]. Laplacian smoothing is applied to move vertices while compensating mesh shrinkage[Taubin 2000]. Each vertex move is calculated by factor λ/μ to the barycenter of its neighboring vertices. Common way to attenuate noise in mesh is through an isotropic diffusion process with implicit surface. Non equal edge lengths can be compensated by weights as proposed by [Desbrun et al. 1999]. Constraints on vertex move can be soft (weighted) or hard. Vertex move for important features can be constrained with marking by user, but this is tedious job for most applications.

To delineate high frequency noise from low frequency features, one can use weighted smoothing for different directions. [Meyer et al. 2003] uses anisotropic smoothing technique based on mean curvature flow (MCF) to preserve edges over principal curvatures κ_1 and κ_2.

Smoothing weight at vertex $\mathbf{x}_i$ is defined as:

$$w_i = \begin{cases} 1 & \text{if } |\kappa_1| \le T \text{ and } |\kappa_2| \le T \\ 0 & \text{if } |\kappa_1| > T \text{ and } |\kappa_2| > T \\ & \quad \text{and } \kappa_1\kappa_2 > 0 \\ \kappa_1/\kappa_H & \text{if } |\kappa_1| = \min(|\kappa_1|, |\kappa_2|, |\kappa_H|) \\ \kappa_2/\kappa_H & \text{if } |\kappa_2| = \min(|\kappa_1|, |\kappa_2|, |\kappa_H|) \\ -1 & \text{if } |\kappa_H| = \min(|\kappa_1|, |\kappa_2|, |\kappa_H|) \end{cases}$$

with user defined edge thresholding parameter T.

Anisotropic smoothing methods [Taubin 2001; Liu et al. 2002; Desbrun et al. 1999; Hildebrandt and Polthier 2006] were developed to preserve and enhance mesh features while denoising the surface.

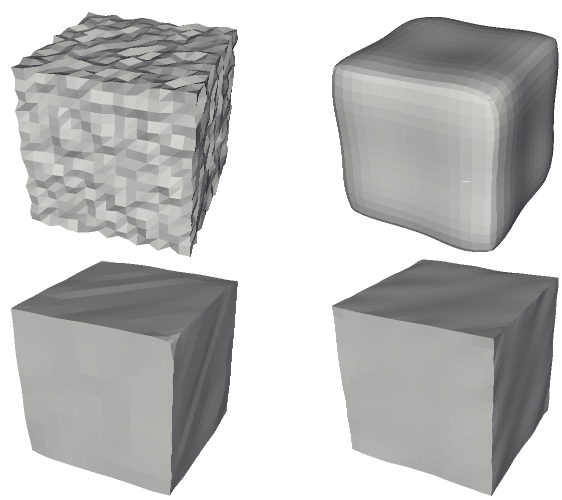


Figure 1: Top left: $16 \times 16 \times 16$ cube as a triangle mesh with 0.3 median edge length noise added. Top right: After smoothing with iterative mean filter. Bottom left: After smoothing by weighted median filter. Some corners are not recovered. Bottom right: After smoothing with feature sensivity.

Rather than computing per-vertex displacement which takes into account only small region covered by vertex (called star or umbrella), one can assume that face normals should be adjusted. This gives larger region and solves many convergence problems. [Taubin 2001] analyzed existence and uniqueness of a solution to the problem of updating vertex positions given a field of face normals. In general no solution exists. Taubin proposed least-squares optimization method to minimize over-constrained system. [Shen and

*email: leon.kos@uni-lj.si

Barner 2004] used this method for fuzzy median filtered surface normals.

[Yagou et al. 2002] showed interesting approach with recalculation of surface normals and then applying vertex move by simplified version of nonlinear diffusion proposed by [Ohtake et al. 2001]. This nonlinear diffusion tends to crease sharp edges. It has been shown by our test to be more effective and stable than iterative approach with mean square error as shown in [Shen and Barner 2004].

Recently [Chen and Cheng 2005] suggested to apply different filters based on local sharpness measure to further improve corners. Similarly [Sun et al. 2002] computed edge strength and applied weighted vertex step size to umbrella.

Figure 1 shows some features of each algorithm on synthetic cube mesh. Median filter is applied to angles and besides sharp edges, corners are known to be problematic. Median filter applied on observation window $\Omega = \{x_1, x_2, ..., x_N \in \Re\}$ where N is the window size is defined as

$$x_{MED} = \arg\min_{x} \sum_{i=1}^{N} |x - x_i| .$$

Median filtering requires sorting of window samples $x_1, x_2, ..., x_N$ such that $x_{(1)} < x_{(2)} < ... < x_{(N)}$. In such ordered samples, median value is selected as $x_{MED} = x_{(\frac{N+1}{2})}$. Extending this to face normals one can use angle between vectors [Shen and Barner 2004; Yagou et al. 2002], to select nearest normal:

$$\mathbf{n}_{AMED} = \arg\min_{n \in \Omega} \sum_{i=1}^{N} A(\mathbf{n}, \mathbf{n}_i) . \quad (1)$$

Proposed algorithm: In this paper, we extend the concepts of different smoothing algorithms when features are detected by measure of face neighborhood as proposed by [Chen and Cheng 2005]. Algorithm works in two steps. First we calculate weighted mean normals on triangle faces. Then we show how newly assigned face normals are used to move triangle vertices. Finally, we describe how different filter are applied to different surface areas in order to preserve important features. Applying different smoothing techniques based on mesh variance measure is more procedurally described in section 3.

2 Background

We begin with brief overview of triangle mesh smoothing theory used in our algorithm. Although face averaging on whole mesh do not produce sharp features, it is useful for areas with little variations. We extend mean face normals averaging with a weighted mean filter.

2.1 Face normals averaging

For mesh consisting of triangles T and normal $\mathbf{n}(T)$ on each triangle have neighboring triangles $\mathcal{N}(T)$ with at least one common vertex. Unit normal $\mathbf{n}(T)$ on each triangle is calculated with normalized cross product from oriented triangle vertices. Averaged normal $\mathbf{m}(T)$ can be computed as weighted normal sum of all neighboring triangles $\mathcal{N}(T)$:

$$\mathbf{m}(T) = \sum_{S \in \mathcal{N}(T)} w(S) A(S) \mathbf{n}(S) \quad , \quad (2)$$

where $A(S)$ is neighboring triangle area and $w(S)$ corresponding weight.

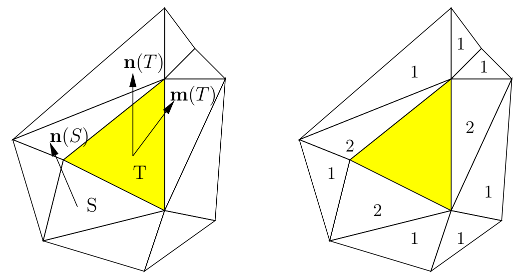


Figure 2: Left:Averaging face normal $\mathbf{m}(T)$ of triangle T from neighboring normals $\mathbf{n}(S)$. Right: Allocating weights to triangles.

Figure 2 shows how can weights be allocated to adjacent triangles with common edge.

Resulting averaged vectors must be normalized as

$$\mathbf{m}(T) \longleftarrow \frac{\mathbf{m}(T)}{||\mathbf{m}(T)||} \quad (3)$$

to be useful for further processing.

2.2 Vertex update

Difference between triangle normal $\mathbf{n}(T)$ and averaged $\mathbf{m}(T)$ are adjusted with simplified nonlinear diffusion proposed in [Ohtake et al. 2001].

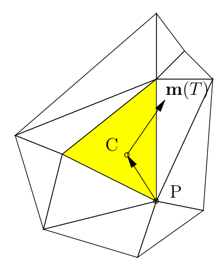


Figure 3: Updating vertex P in opposite direction of $\mathbf{m}(T)$ weighted by area of triangle and projected distance of centroid C from P.

Vertex P is adjusted by set of updated neighboring face normals $m(T)$ with the following procedure

$$P' = P + \frac{1}{\sum A(T)} \sum_{T \in \mathcal{N}(P)} w(T) \mathbf{m}(T) , \quad (4)$$

$$w(T) = A(T) \left[\vec{PC} \cdot \mathbf{m}(T) \right] . \quad (5)$$

Weight $w(T)$ is a weighted projection of vector $\vec{PC}$ on $\mathbf{m}(T)$. This (negative) weight tries to move point P in direction opposite to $\mathbf{m}(T)$ and thus aligns triangle normal $\mathbf{n}(T)$ with adjusted $\mathbf{m}(T)$. Triangle centroid $\mathbf{C}$ is computed from triangle vertices $\mathbf{P}_i$, $\mathbf{P}_j$ and $\mathbf{P}_k$ as

$$\mathbf{C} = \frac{\mathbf{P}_i + \mathbf{P}_j + \mathbf{P}_k}{3} . \quad (6)$$

If $\mathbf{n}(T)$ and $\mathbf{m}(T)$ are close, then P is not moved. Such diffusion can crease edges if face normals $\mathbf{m}(T)$ are properly assigned. As edges do not collapse and go round, mesh volume is also preserved.

2.3 Feature detection

In order to delineate piecewise smooth surfaces one has to prior delineate features such as edges and corners. Detecting surface variation can be done with principal curvatures using vertex as reference [Meyer et al. 2003] or calculated curvature over triangle face [Rusinkiewicz 2004]. Median filter is known to preserve edges, but can fail on corners. [Gonzalez and Woods 2002] showed for image processing, that min filter (nearest filter) preserves corners on image. Except for sharp features min filter does not have smoothing properties. Thus it is reasonable to apply different filter according to different local mesh properties. For face neighboring triangles $\mathscr{N}(T)$ surface variation is calculated as:

$$s_i = \frac{1}{N} \sum_{j \in \mathscr{N}(T)} (||\mathbf{n}_i - \mathbf{n}_j|| - n_{i,MED})^2, \quad (7)$$

where n_{MED} is median normal distance between triangle T and N neighboring triangles $\mathscr{N}(T)$ calculated with the following equation:

$$n_{i,MED} = \arg \min_{\mathbf{n}_i \in \mathscr{N}(T)} \sum_{j=1}^{N} ||\mathbf{n}_i - \mathbf{n}_j||, \quad (8)$$

where $||\cdot||$ denotes Euclidean distance norm. Note that we do not use mean value, but rather median distance, for measuring shape variance. Median filtering has been shown that it is more resistant to outliers in mesh.

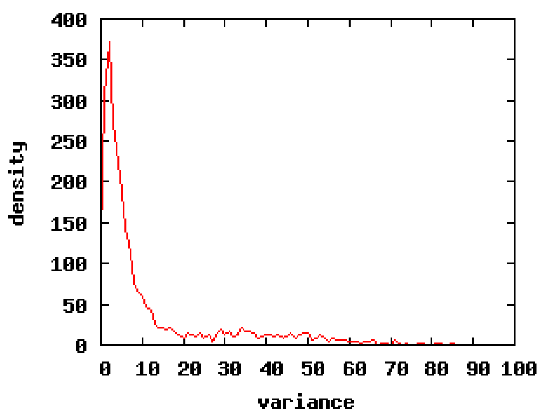


Figure 4: Histogram of normal distance variance for noisy cube shown in Figure 1.

Face normal distance variance is used to distinguish between sharp areas and smooth one. Figure 4 shows histogram of normal distance variance (7) normalized to $[0..100]$ range. Histogram shows that major areas are smooth and that sharp features can be detected by setting threshold value $s_{th} > 15$. Influence of different smoothing algorithms depending on detected feature should be also smooth. This means that there should be mesh area where two filter have influence. We choose linear transient function for filter influence, described in detail in section 3.

2.4 Preprocessing

Before we continue we consult general requirements for described algorithms. As one might suspect, mesh smoothing works well for uniformly distributed mesh model. For non-uniform distributed model and meshes with connectivity problems, preprocessing is required to prepare mesh for further processing. [Lee et al. 1998] showed how to construct smooth parametrization of irregular connectivity triangulation of arbitrary genus 2-manifolds.

We assume that 1-ring neighborhood is enough for detection of features. This assumption can be satisfied in advance with proper scanning or later with reparametrization. Preprocessing step also requires checks for mesh problems such as mesh crossover, holes and outliers. See [Guskov and Wood 2001] for example how to deal with mesh topological problems.

3 Algorithm

As mentioned earlier, our method for feature preserving mesh smoothing is based on feature dependent filter. To extend this to the case of ranking filter we first describe how face normals are selected and how vertices are moved, then show how feature measure is applied to weighted filter selection. Finally, we describe how algorithm applies into iterative loop.

Geometry database requires to have associativity of vertex and face connectivity. Additionally each vertex requires list of adjacent faces and each triangle face requires list of 1-ring faces as shown in figure 2.

Based on algorithms introduced in previous section we give procedural description of the mesh correction algorithm:

1. Compute averaged normal $m(T)$ for each triangle T using equation (2). Averaged normals should be normalized as suggested by Eq. (3). This normals will be used for smooth areas.

2. For 1-ring neighborhood $\mathscr{N}(T)$ of each triangle T determine closest face normal $\mathbf{e}_i$ from all neighboring normals n_j, $j \in \mathscr{N}(T)$.
$$\mathbf{e}_i = \min(||\mathbf{n}_j - \mathbf{n}_i||) \quad (9)$$
Resulting normal $\mathbf{e}_i$ will be used for areas with sharp features.

3. Calculate feature measure s_i, for each face from variance of the distance i-th face normal and neighboring face normals using equation (7). This shape variation is used to delineate which normal is dominantly selected. Transient areas are determined with the following weighted function.

4. Compute new face normal for each triangle by applying weighted feature dependent function
$$\mathbf{n}_i = W(s_i)\mathbf{m}_i + (1 - W(s_i))\mathbf{e}_i \quad (10)$$
where $W(s)$ is Gaussian weighting function
$$W(s) = \exp(-\frac{s^2}{2\sigma^2}), \quad (11)$$
with user-defined standard deviation σ which can be estimated from histogram like one shown in figure 4. Normalize resulting vector
$$\mathbf{n}_i \longleftarrow \frac{\mathbf{n}_i}{||\mathbf{n}_i||}.$$
This step results with new suggested normals $\mathbf{n}_i$ which should be satisfied as close as possible with vertex move.

5. Update each triangle vertex using equation (4). This completes one iteration of the shape recovery process.

6. Iterate previous steps until convergence. This step can use various convergence norms. In simplest case, predefined number of iterations can be prescribed. Usually 30 iterations is enough for most cases.

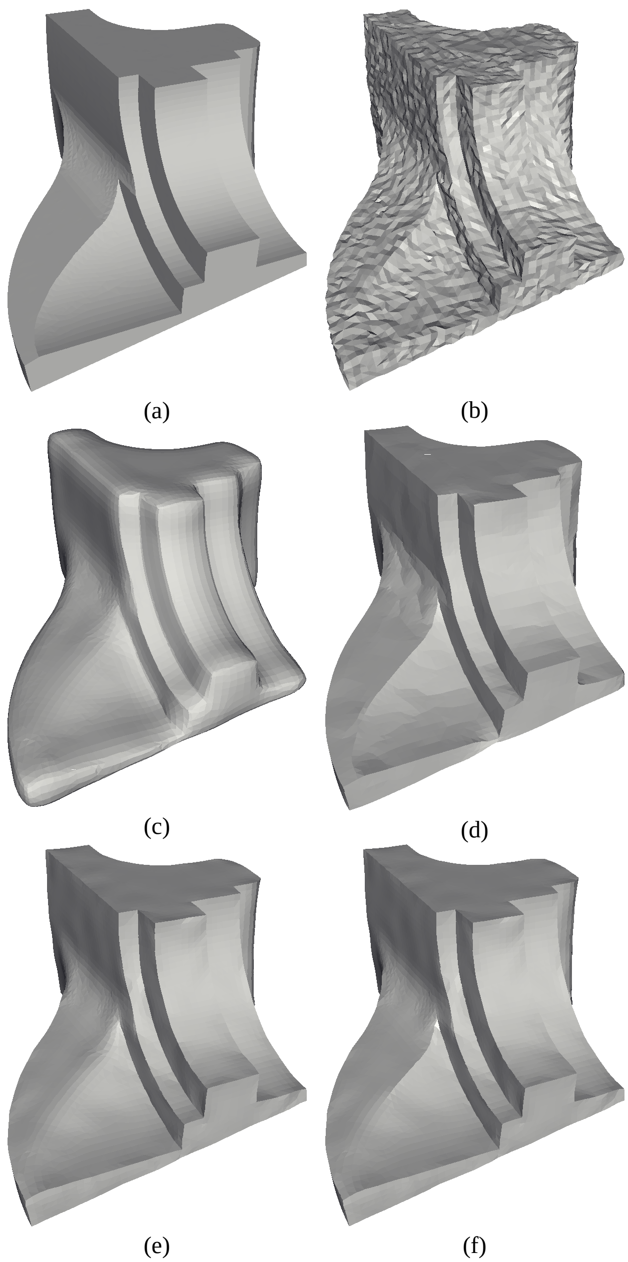


Figure 5: Comparison of smoothing algorithms on the fandisk model that has a vanishing ridge. All subfigures presented here use the same vertex update algorithm from section 2.2: (a) The original model (Courtesy of H.Hoppe). (b) Mesh corrupted with Gaussian noise (20% of median edge length). (c) Mean face normals smoothing [Yagou et al. 2002] applied. Noise is removed on the expense of round edges. (d) Angle median algorithm[Yagou et al. 2002] using weighted median filter with weights as shown in figure 2. Note that some edges collapsed together and that curved areas are piecewise flat and not smooth. (e) Feature dependent filter described in section 3 with user-defined $\sigma^2 = 0.003$ preserves corners and edges while smoothing large curved and flat areas. (f) With change of $\sigma^2 = 0.0005$ one can also fine tune reconstruction of vanishing edge which smoothly changes from sharp to smooth.

4 Results and Discussion

Several synthetic meshes ranging from single cube to more complex shapes with sharp features were generated for testing at different noise levels. Other models with known accurate surface (such as Fandisk model) as well as scanned models were evaluated with error metrics and visual performance. With the user defined feature detection parameters we were able to successfully delineate sharp edges and smooth areas. Measuring error is not well established in literature when feature preserving is concerned. For "media" models there are well established error metrics for triangular meshes [Cignoni et al. 1998]. Noise-added synthetic models have advantage that we can use also other error metrics such as L^2, angular or volume change. Most authors [Clarenz et al. 2004; Fleishman et al. 2005] rely on visual fairness of recovered shapes. We follow this practice giving examples of our method potentials on standard test meshes.

Figure 5 shows fandisk mesh selected for algorithm strength demonstration. Original fandisk mesh with a vanishing ridge was corrupted with zero mean Gaussian noise by 20% median edge length vertex move. With such corrupted mesh, vanishing ridge looses its sharpness in the middle. Then this mesh was processed with several algorithms for comparison. Mean smoothing algorithms [Taubin 2000; Yagou et al. 2002] are well suited when there are no features to preserve. Most non-artificial models and scanned statues, such as Stanford Scanning Repository, fall in this category. Angle median filter [Yagou et al. 2002] applied on fandisk mesh in figure 5d shows that it tends to converge into flat patches even there is evident that most patches should be smooth. Changing the same median algorithm from angle to curvature helps within smooth areas but fails on preserving edges and corners. Vanishing edge with angle median algorithm seems to be preserved, but details show that this edge is not contiguous.

Our feature preserving algorithm measures edge variance and aligns face normals according to neighborhood variation and balances influence of two competing basic algorithms. With varying parameter σ in equation (11) vanishing edge in figure 5e and 5f shows that even smoothness can be controlled. It should be noted that useful σ range is hard to determine although [Chen and Cheng 2005] for similar approach suggest Bayesian classifier. From our experience, most helpful is variance histogram as one shown in figure 4. Mesh tessellation and triangle (non) uniformity have impact on most surface fairing algorithms.

Additional tests in appendix show problems with some algorithms on synthetic CAD models with irregular tessellation and mesh configuration. This is due to non uniform and long triangles. Thin surfaces can also be problematic for most algorithms. It was verified that our algorithm is free of such configurations.

Another tests performed were on media meshes such as finger skin detail in figure 8 with 375000 triangles. This mesh shows that most of the averaging algorithms performs well with high density scans without evident noise. Media models normally do not have/require sharp edges, but when additional edge enhancement is required, our method can also perform well.

Computational complexity of the algorithm shows that there are no trigonometric functions and that algorithm can be used for quick in-place computation. Possibly in graphics hardware computation (GPU pipeline). Weighting function (10) can be further extended to piecewise smooth transfer function and thus additionally lowering complexity.

5 Conclusion

Our feature preserving algorithm showed that filtering noise with weighted edge marking can result in sharp resultant mesh. The proposed procedure is mathematically tractable, stable, and fast as no trigonometric functions are used. Robustness of the algorithm was tested on many different meshes to prove its general applicability even when such mesh processing is not clearly evident. Algorithm is free of degenerate configurations, and yields stable convergence where some algorithms fail. Standard deviation σ is the only user-defined parameter required can also be automatically selected. Additionally some heuristics for flat surfaces should be included when such scans are commonly acquired. Perhaps most interesting question for future is how to incorporate additional procedural feature detection into specific models and how such local feature sensivity performs on general meshes.

References

CHEN, C.-Y., AND CHENG, K. Y. 2005. A sharpness dependent filter for mesh smoothing. *Computer Aided Geometric Design*. to appear.

CIGNONI, P., ROCCHINI, C., AND SCOPIGNO, R. 1998. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum 17*, 2, 167–174.

CLARENZ, U., RUMPF, M., AND TELEA, A. 2004. Robust feature detection and local classification for surfaces based on moment analysis. *IEEE Transactions on Visualization and Computer Graphics 10*, 5, 516–524.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 317–324.

FLEISHMAN, S., COHEN-OR, D., AND SILVA, C. T. 2005. Robust moving least-squares fitting with sharp features. *ACM Trans. Graph. 24*, 3, 544–552.

GONZALEZ, R. C., AND WOODS, R. E. 2002. *Digital Image Processing*. Addison-Wesley Pub, ch. 3.

GUSKOV, I., AND WOOD, Z. 2001. Topological noise removal. In *Proceedings of Graphics Interface 2001*, B. Watson and J. W. Buchanan, Eds., 19–26.

HILDEBRANDT, K., AND POLTHIER, K. 2006. Anisotropic filtering of non-linear surface features. Tech. Rep. ZIB-Report 04-25, Konrad-Zuse-Zentrum für Informationstechnik Berlin, D-15195 Berlin-Dahlem, Germany, 6.

LEE, A. W. F., SWELDENS, W., SCHRÖDER, P., COWSAR, L., AND DOBKIN, D. 1998. MAPS: Multiresolution adaptive parameterization of surfaces. *Computer Graphics 32*, Annual Conference Series, 95–104.

LIU, X., BAO, H., SHUM, H., AND PENG, Q. 2002. A novel volume constrained smoothing method for meshes. *Graphical Models 64*, 169–182.

MEYER, M., DESBRUN, M., SCHRDER, P., AND BARR, A. 2003. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, Springer-Verlag, Heidelberg, H. C. Hege and K. Polthier, Eds., 35–57.

OHTAKE, Y., BELYAEV, A., AND BOGAEVSKI, I. 2001. Mesh regularization and adaptive smoothing. *Computer-Aided Design 33*, 11, 789–800.

RUSINKIEWICZ, S. 2004. Estimating curvatures and their derivatives on triangle meshes. In *Symposium on 3D Data Processing, Visualization, and Transmission*, IEEE Computer Society, Los Alamitos, CA, USA, 486–493.

SHEN, Y., AND BARNER, K. 2004. Fuzzy vector median-based surface smoothing. *IEEE Transactions on Visualization and Computer Graphics 10*, 3 (5-6), 252–265.

SUN, Y., PAGE, D. L., PAIK, J. K., KOSCHAN, A., AND ABIDI, M. A. 2002. Triangle meshes-based edge detection and its application to surface segmentation and adaptive surface smoothing. In *International Conference on Image Processing ICIP02*, vol. III, 825–828.

TAUBIN, G. 1995. A signal processing approach to fair surface design. In *SIGGRAPH Computer Graphics Proceedings*, 351–358.

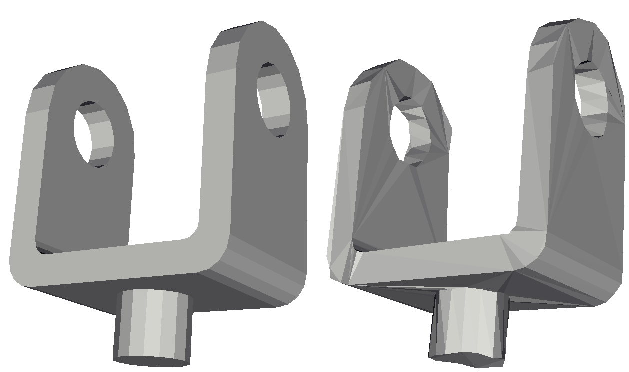
TAUBIN, G. 2000. Geometric signal processing on polygoonal meshes. In *Eurographics '2000*.

TAUBIN, G. 2001. IBM research report: Linear anisotropic mesh filtering. Tech. rep., IBM Research division, T.J. Watson Research Center, Yorktown heights, N.Y., 10.

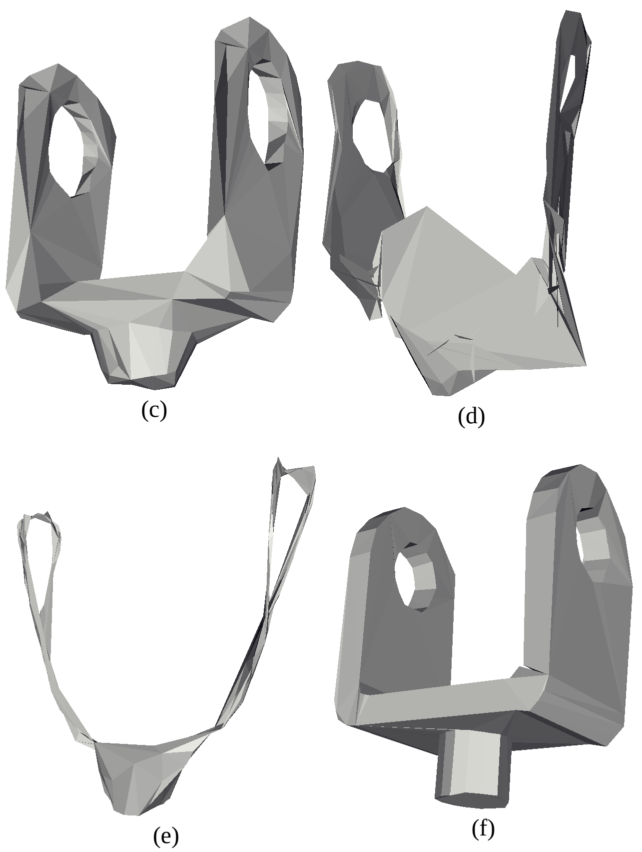
YAGOU, H., OHTAKE, Y., AND BELYAEV, A. 2002. Mesh smoothing via mean and median filtering applied to face normals. In *Geometric Modeling and Processing*, 124–131.

A Comparison of algorithms on synthetic and scanned objects

The following figures show some additional tests for various triangular mesh configurations with emphasized thin triangles, low/high density tessellation and detail of finger without added noise.



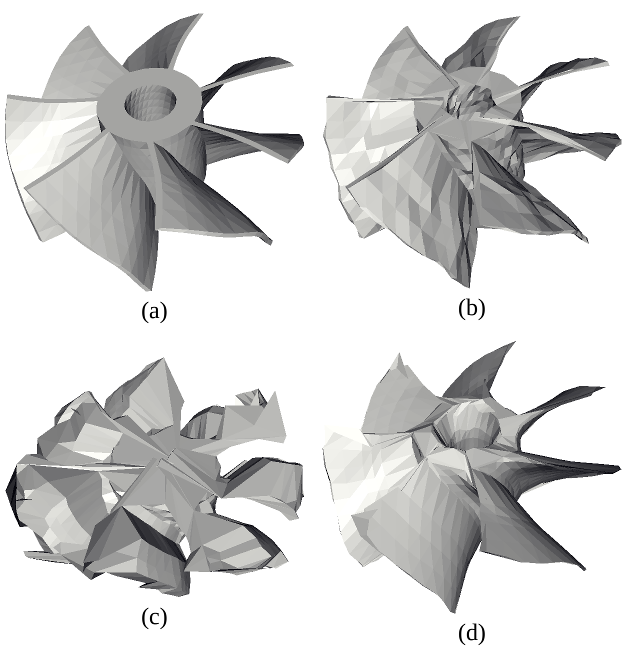
(a) (b)



(c) (d)

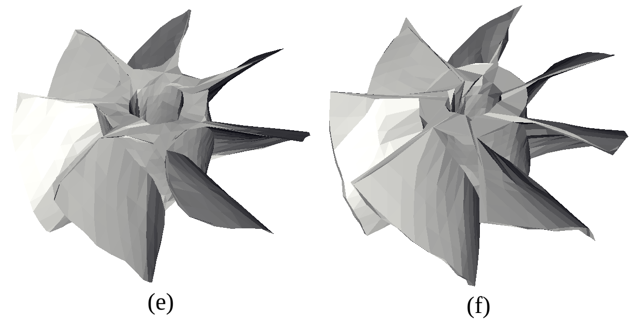
(e) (f)

Figure 6: U-joint sample shows that only feature dependent method regularize mesh. Mean and angle median methods do not converge. (a) Original model. (b) Model with applied noise shows long triangles. (c) Seven iterations of $\lambda - \mu$ smoothing [Taubin 1995]. (d) Mean face normals smoothing [Yagou et al. 2002] applied. Failed to converge due to non-baloon geometry. (e) Angle median algorithm[Yagou et al. 2002] using weighted median filter fails. (f) Feature dependent filter described in section 3 with user-defined $\sigma^2 = 0.003$ preserves corners and edges while smoothing large curved and flat areas.



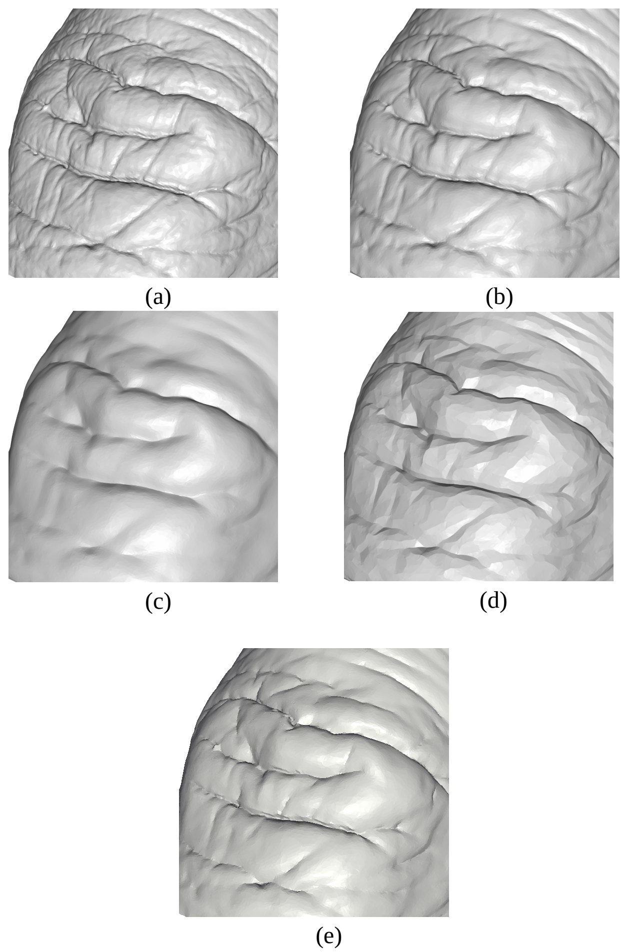
(a) (b)

(c) (d)



(e) (f)

Figure 7: Turbine fan sample shows that that only feature dependent method is able to preserve thin structures. (a) Original model. (b) Model with applied noise that change blade thickness. (c) Five iterations of $\lambda - \mu$ smoothing [Taubin 1995]. (d) Mean face normals smoothing [Yagou et al. 2002] applied. (e) Angle median algorithm[Yagou et al. 2002] using weighted median filter attempts to average central part. (f) Feature dependent filter described in section 3 preserves corners and edges while smoothing large curved and flat areas.



(a) (b)

(c) (d)

(e)

Figure 8: Finger skin detail: (a) Original model with dense mesh with no evident or added noise. (b) Taubin $\lambda - \mu$ smoothing [Taubin 1995] with 30 iterations performs fairly well with smoothing out noise. (c) Mean face normals smoothing tends to smooth out most of the skin details.. (d) Angle median algorithm[Yagou et al. 2002] converges to flat surfaces with sharp edges. (e) Feature dependent filter enhances some edges while removes others.