Feature Sensitive Mesh Processing

Leif Kobbelt Mario Botsch

Computer Graphics Group, RWTH Aachen

Abstract

Many mesh processing algorithms assume the actual geometry of a triangle mesh to be characterized by the vertex positions only. From the manifold point of view however, triangle meshes have to be considered as continuous piecewise linear surfaces. In sufficiently smooth and flat regions of the surface this observation does not really matter since any triangulation will yield a decent approximation to the underlying geometry. In the presence of sharply curved features however, this is not true. Here, severe alias-artifacts can affect the perceived surface quality and can lead to quite bad approximation behavior.

In this paper we will discuss several consequences of this observation and present recently developed algorithms for feature sensitive mesh generation and re-meshing. We will report recent results in feature sensitive surface extraction from volume data, surface anti-aliasing by remeshing of blend regions in technical data sets, and diffusion based remeshing of triangle meshes.

Keywords: geometric modeling, polygonal meshes, feature sensitivity

1 Introduction

Because of their simplicity and efficiency, polygonal meshes are a well accepted universal surface representation in the field of Computer Graphics. For several years now a large number of mesh processing algorithms has been developed, many of them providing discrete analogons to methods originally derived for continous parametric surface representations.

In parallel to this developement polygonal meshes are also increasingly used in more sophisticated engineering applications and numerical simulations (e.g. CFD). In this context one has to guarantee that the meshes to be used for simulation are sufficiently good approximations to their continous physical counterparts, otherwise the simulation's results will not be reliable. In the presence of sharply curved features a high-quality approximation becomes even more important since these geometric features are usually very relevant for the simulation and will therefore have a strong influence on the results.

Since the key to successfully using polygonal meshes in this kind of applications is a sufficient high-quality representation of the geometric features we will review the approximation properties of polygonal meshes and discuss the consequences in Section 2.

After that we will present three examples of feature sensitive mesh processing algorithms. We will enhance the standard Marching Cubes algorithm to detect and reconstruct sharp features in the surface extraction process in Section 3. In Section 4 we derive an optimal sampling pattern for feature and blend regions in technical data sets and use this pattern to resample given meshes thereby effectively reducing alias-artifacts. In order to improve the geometric as well as topological regularity of given triangle meshes a diffusion based remeshing method is finally presented in Section 5.

2 Approximation Properties and Normal Noise

From approximation theory we know that approximating a smooth (sufficiently differentiable) surface S using a piecewise linear and continous interpolant M (polygonal mesh) converges with quadratic approximation order. The approximation error locally is a function of surface curvature. Doubling the sampling density of M by refining the mesh will therefore decrease the approximation error by a factor of $\frac{1}{4}$. However, in the vicinity of sharp features the surface S is no longer differentiable since normal vectors are not continous across that feature. This loss of differentiability causes the approximation order to drop down to linear, leading to much slower convergence in these areas.

Since sharply curved features correspond to high frequencies of the surface signal S [Taubin 1995b; Taubin 2000] and since the polygonal mesh M is a finite discrete sampling of this surface, signal processing theory tells us that the sampling density has to sufficiently adapt to the signal's frequency spectrum, otherwise we will not be able to capture all geometric features and end up with alias-artifacts. Following these priniciples one has to adjust the sampling density of M to the curvature of the underlying surface S, i.e. we can use a coarser sampling in smooth and flat areas and have to increase the vertex density only in highly curved regions of the surface.

Unfortunately, the degenerate case of sharp features of S corresponds to an infinitely high curvature or frequency spectrum, respectively, requiring us (in theory) to use infinitely many samples in their vicinity. As Fig. 1 clearly depicts, increasing the sampling

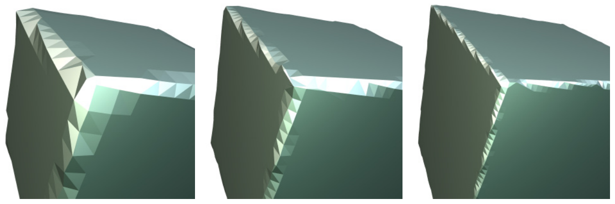

Figure 1: Alias error at high-frequency geometric details. By refining the mesh, the effect becomes less and less visible due to the convergence of the mesh M to the continous surface S but the problem is not really solved since the normal vectors of M do not converge to the normals of S. These alias-errors can only be removed by placing samples exactly on the sharp features.

density will result in a sequence of meshes which converge pointwise to S, but whose normal vectors will never converge to the normals of S. Hence, refining the mesh will in fact *not* remove the alias-artifacts, it will just shift them up to a higher frequency band. Algorithms requiring first order surface information like normal vectors will then give anything but reliable results in these cases. Such methods may be as simple as surface shading showing specular artifacts or more sophisticated simulations like CFD, where these randomly tilted normals may cause erroneous turbulences.

Since already small perturbations of nearby vertex positions can cause large deviations of normal vectors, the approximation error alone will not be a sufficient measure for reconstruction quality. We have to consider the piecewise linear nature of triangle meshes and — in addition to individual vertex positions only — we also have to take the global sampling pattern into account. If we refer to vertex positions as information of order zero then the sampling pattern has influence on the derivative information like triangle normals and can be regarded as a first order measure for reconstruction or surface quality.

We define that a mesh M is a high-quality approximation of a surface S if the mesh normals of M are a subset of the real surface normals of S. If instead the normal vectors are randomly tilted away from the correct direction, we refer to this effect as *normal noise*, similar to *surface noise* being a high-frequency perturbation of vertex positions (cf. Fig. 1,2). The process of reducing or even removing these alias-artifacts is then called *surface anti-aliasing*.

The amount of normal noise is also a measure for mesh quality: high quality surfaces in geometric modeling and CAD are usually characterized by a low variation of curvature, also called *fairness* [Moreton and Séquin 1992; Taubin 1995b; Desbrun et al. 1999]. In the discrete setting of polygonal meshes we can derive a discrete analogon to the concept of surface curvature by considering the *normal jump* across an edge, i.e. the angle between the normals of two incident faces. A triangle mesh is then said to be of high quality if the variation of these normal jumps is low. For low quality meshes with a strong variation of normal jumps we are back to the notion of normal noise again (cf. Fig. 2).

We have seen that geometric features are on the one hand the most significant mesh regions for many applications and that they are on the other hand the most difficult regions from an approximation point of view. If we are given a fixed vertex budget and since all samples have to be placed *on* the surface, the only remaining degrees of freedom are to move the vertices *within* the surface, i.e. to choose the sampling pattern.

The only way to generate meshes of superior quality and free of normal noise is to have this sampling pattern aligned to the surface features:

- We will show in Section 4 that for strongly curved features the mesh has to be aligned to the principal curvature directions of the underlying geometry.

- In the extreme case of infinitely sharp features, surface samples have to be placed exactly on the respective feature edges or corners to get rid of the alias-artifacts.

3 Isosurface Extraction

Besides *explicit* surface representations like polygonal meshes the other important representation are *implicit* surfaces [Bloomenthal et al. 1997]. While the first one is defined to be the *range* of function $f : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$, implicit surfaces are the *kernel* of a volumetric scalar function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$. Although F can be any function mapping points outside/inside the object to positive/negative values, a natural choice is the *signed distance field* function that assigns to every point in $\mathbb{R}^3$ its signed distance to the surface of the object.

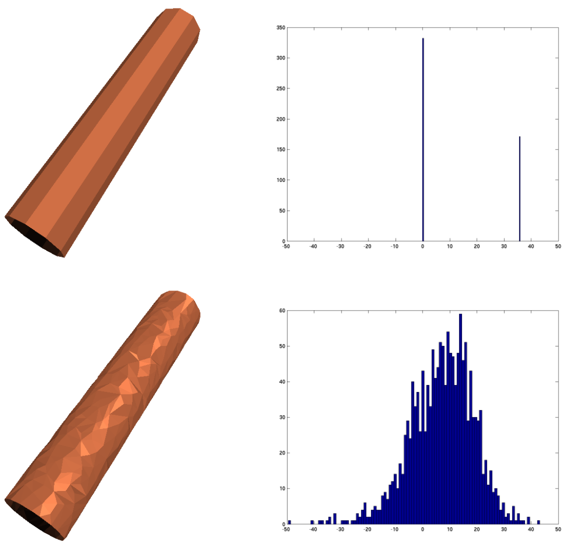
Figure 2: Different sampling of a cylinder and their corresponding variations of *normal jumps*, shown in form of histograms: the normal jumps of the upper model have two different values, corresponding to its two different principal curvatures. The randomly sampled mesh at the bottom has a higher sampling density but is affected by *normal noise*.

Since both explicit and implicit representations have their own advantages and drawbacks one usually chooses the best suitable representation based on the application's needs. In order to freely switch between these surface representations we have to provide conversion algorithms between them. The respective conversion from implicit to explicit refers to extracting the zero-level isosurface from the volume. To get a decent approximation quality we have to ensure that no important geometric detail is lost after the conversion.

The standard algorithm for this contouring task is the *Marching Cubes* [Lorensen and Cline 1987] or any of its variants [Nielson and Hamann 1991; Montani et al. 1994b; Montani et al. 1994a]. The function F is represented by sampling it on a uniform 3D grid $g_{i,j,k}$ and interpolating the resulting values $F_{i,j,k}$ using a tri-linear function in the interior of the grid cells. For each grid cell intersected by the isosurface a small patch is generated. The union of all these patches finally gives the desired isosurface. In order to generate a patch for a specific cell, sample points are placed at the intersection of the cell edges with the isosurface and connected based on a triangulation pattern from a pre-computed look-up table. Since the tri-linear approximation of F is actually linear along the cell edges, the geometric position of these samples is determined based on linear interpolation of the distance values at the edge's endpoints.

While this approach works well in flat regions, the results can strongly deviate from the exact intersection point near sharp features (cf. Fig. 4). Since distance values are decoupled from the directions they are measured in, we interpolate between distances corresponding to different directions near sharp features, lacking any geometric meaning.

Using a different discretization of the distance field F resolves this problem: instead of using just *scalar* distances $d_{i,j,k} = F_{i,j,k}$ we store *directed* distances at the grid nodes. Since Marching Cubes computes sample points on the grid edges only we also have to consider distance computations along these edges only. For the directed distance field we therefore store at each grid point

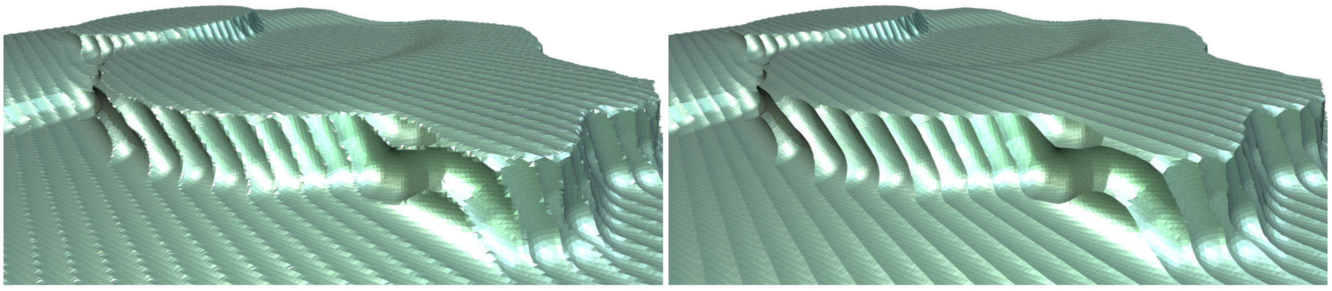
Figure 3: The result of a milling simulation: the milling tool's envelope (constructed from unions of spheres and cylinders) is subtracted from the work piece. The upper image shows the surface extracted by the standard MC algorithm, the lower image shows the extended MC surface. The sharp ridges are better visible due to the clearly reduced alias.

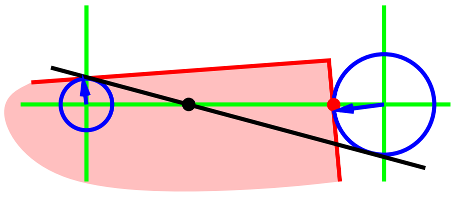
Figure 4: Consider two neighboring grid points in the vicinity of a sharp feature (corner) of the contour S. Sampling the scalar valued distance function F at both grid points and estimating the sample point by linear interpolation leads to a bad estimation of the true intersection point between the contour and the cell edge.

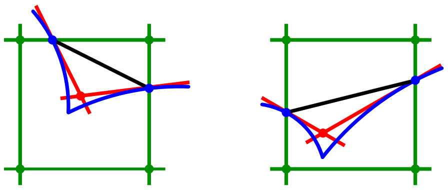
Figure 5: By using point and normal information on both sides of the sharp feature one can find a good estimate for the feature point at the intersection of the tangent elements.

$g_{i,j,k}$ three directed distances in x, y and z direction, i.e. we store $\mathbf{d}_{i,j,k} = (d_x, d_y, d_z)_{i,j,k}$. Basing the sample point computation on these distance values will eventually result in the correct intersection points.

Although memory consumption increases by a factor of three, directed distances have the advantage that they are usually much cheaper to compute than the true minimal Euclidean distance. Generating directed distances is basically performed by ray casting along the coordinate axes and many highly efficient algorithms are available for this task [Arvo and Kirk 1989; Kalra and Barr 1989].

Even if we can compute exact intersection points the major problem with any discretization of F remains. The standard Marching Cubes algorithm computes all samples on a globally uniform grid that cannot be aligned to features of the underlying geometry, causing alias-artifacts at features as we pointed out in Section 2. Unless we are able to place sample points *on* the feature we will not get rid of these errors, even if we increase the grid resolution (cf. Fig. 1). The problems to be solved are therefore the detection, sampling and reconstruction of features on a per cell basis.

The key idea is to use additional local information from the distance field F and to extrapolate the behaviour of the surface near the feature. Instead of using the sample point's position only we also make use of its tangent plane or normal vector, respectively, i.e. the gradient of F at this point. Instead of directly connecting sample points on the edges we approximate the surface using local *tangent elements* and estimate the feature's position by intersecting them (cf. Fig. 5). Even if the surface is not differentiable near the feature it can be assumed to be at least piecewise differentiable on both sides of it. Hence, additionally using gradient information brings us back to quadratic approximation order on each side of the feature.

In 3D the situation behaves similarly but is slightly more complicated, since we have to distinguish between *feature edges* and *feature corners*. Feature detection and classification is performed based on the normal vectors of the sample points. In absence of a feature we generate the standard Marching Cubes patch. Otherwise we place a sample point on the feature and adjust the triangulation table accordingly.

In order to place a sample point on the feature we have to compute the intersection of all tangent elements defined by sample points and their respective normal vectors. This leads to a linear system that can be (numerically) both underdetermined as well as overdetermined. Using the pseudo-inverse based on singular value decomposition enables us to *robustly* handle both cases and will result in a Least Norm or Least Squares solution [Golub and van Loan 1996].

Combining the proposed *directed distance* with this *Extended Marching Cubes* algorithm allows for feature-sensitive isosurface extraction [Kobbelt et al. 2001]. Since the simple algorithmic structure of the standard Marching Cubes is preserved, this improved version can serve as a replacement in all cases where the additional tangent plane information can be provided, like, e.g, the CSG application in Fig. 3.

More recently Ju et al. [Ju et al. 2002] combined this approach with the surface-nets from [Frisken et al. 2000] to extract adaptive isosurfaces from volume data that also preserve sharp features. While the provided adaptivity solves the problem of uniform overtesselation of Marching Cubes like algorithms, it has the severe drawback, that the resulting meshes may not be manifold.

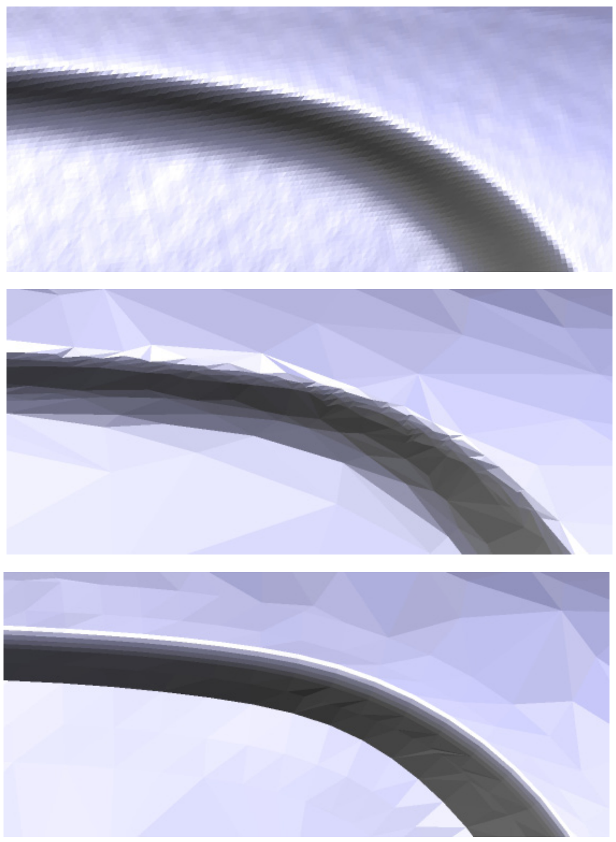

Figure 6: Geometric alias effects such as normal noise become clearly visible under specular shading. The top image shows an original 3D-scan of a feature region. Although the point positions have been sampled at high precision and high resolution, the normals of the resulting mesh deviate strongly from the normals of the original surface. Applying mesh decimation (center) even increases normal noise. In the bottom image we applied our alias-reducing feature resampling. Although the mesh resolution has not changed, the quality has improved due to effective normal noise elimination. The respective normal vectors now are a subset of the correct surface normals.

4 Resampling Blend Regions

The input meshes to be used in numerical simulations are often the result of a reverse engineering process, like e.g. 3D scanning a physical prototype. In order to be sure not to lose relevant geometric detail at early stages of the reconstruction pipeline, these meshes may initially contain several millions of sample points [Bernardini et al. 1999; Levoy et al. 2000]. As a consequence these datasets have to be decimated down to a complexity the target application is able to handle [Garland and Heckbert 1997; Kobbelt et al. 1998].

Since the applied mesh decimation schemes usually are *greedy* algorithms that only consider the *local* shape to decide about which vertex to remove in the next step, one has no direct influence on the global distribution of the mesh vertices on the surface. Although the sampling density may locally adapt to the surface curvature there is no possibility to achieve global effects like an alignment of the triangulation to sharply curved features in the geometry. As we pointed out in Section 2 this will unavoidably result in normal noise since the local surface shape at a surface point is not only characterized by curvature radii but also by the corresponding principal directions (cf. Fig. 6).

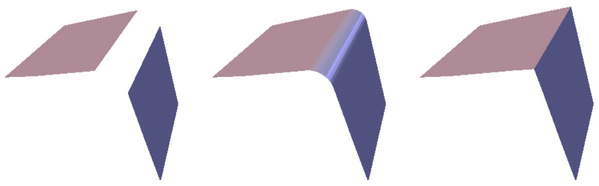

Figure 7: Feature regions on a complex surface usually emerge from blending two separate patches (left) along their corresponding boundary, e.g. by rolling a ball of prescribed radius along the boundary (center). A degenerate ball of radius zero results in a sharp feature edge (right).

These alias-artifacts become particularly evident in the vicinity of feature lines of the original shape. In the case of technical models the features are usually blend regions between separate parts of the surface, e.g. by rolling a ball of prescribed radius along the common boundary (cf. Fig. 7). Since the radius of this ball is much smaller than the curvature of the trajectory it is moving on, these feature regions are characterized by strongly differing principal curvatures $\kappa_1 \ll \kappa_2$. In order to reduce or even remove normal noise, we have to derive a suitable sampling pattern for these blend regions [Botsch and Kobbelt 2001a].

If we first consider an orthogonal cylinder, it is easy to see that random sampling causes normal noise (cf. Fig. 2). If we want the normals of the approximating mesh to be correct — in the sense that they are a subset of the real cylinder's normals — each triangle must have one of its edges parallel to the cylinder's axis. Consequently all vertices should lie on a set of lines which are parallel to the cylinder axis and distributed equally around the cylinder. Each strip between two of these lines can be tesselated by a planar triangulation. The normal jump between two triangles is either zero (within the same strip) or a constant angle (between two strips) that only depends on the number of strips. Hence the normal noise is minimal, the two different values correspond to the two different principal curvatures of the cylinder.

Next we have to generalize this sampling pattern to rolling ball blends, i.e. to surfaces that are the envelope of a ball or a circular profile swept along a space curve. In this case we have to discretize the surface in two directions, one being around the feature and one along the center curve. The moving profile itself is again discretized by a regular n-gon. When sweeping this closed polygon instead of the circle, we obtain n ruled surfaces and the normal jump between neighboring ruled patches is $2\pi/n$.

Discretizing the moving profile along the center curve is less critical since the corresponding trajectories are lines of minimal curvature and therefore will lead to small normal jumps only. However, to preserve the constant normal jump property as good as possible, we have to discretize all trajectories in a synchronized manner, i.e. we discretize different time instances of the sweeping n-gon. Since the resulting quadrilaterals are almost planar, splitting them into two triangles by inserting a diagonal will not introduce a significant normal jump. This intuitive and natural sampling pattern can be shown to minimize normal noise and therefore to reduce surface alias-artifacts to a minimum [Botsch and Kobbelt 2001a].

Resampling a given triangle mesh in order to better represent feature regions is a more difficult task, since we are neither given the moving profile nor the center curve. Therefore we have to construct the sampling pattern based on principal curvature directions. Although this could be done by a robust feature detection algorithm, we aimed at a semi-automatic approach to give the designer maximum flexibility. In an interactive remeshing session the user constructs a tensor-product-like *fishbone* structure (spine and orthogo-

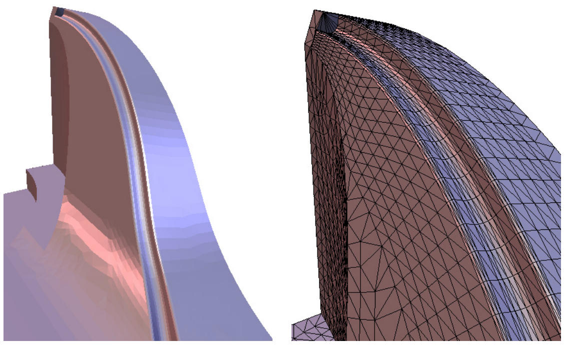

Figure 8: In addition to anti-aliased resampling the tensor-product structure of the *fishbone* metaphor also provides quite sophisticated higher-level modeling operations like exchanging the swept blend profile.

nal ribs) which will finally be tesselated to an anti-aliased surface patch and stitched into the surface mesh. The fishbone structure not only allows for anti-aliased resampling, it also provides higher-level modifications like changing curvature radii or completely replacing the swept profile (cf. Fig. 8). In addition the tensor-product structure of the fishbones reduces several complicated bivariate problems to much simpler and numerically more robust univariate problems.

We applied the surface anti-aliasing technique in the context of CFD simulation for conceptual car design and shape quality control. The normal noise contained in the models after the triangulation and decimation phase could effectively be removed, see Fig. 10 and also Fig. 6.

5 Remeshing

If the input meshes for the target application do not result from a reverse engineering process, but instead have to be constructed from scratch, the typical approach is to use sophisticated CAD systems. Since numerical computations are typically applied to triangle meshes, most CAD systems provide surface tesselation algorithms that convert their internal NURBS based surface representations to polygonal meshes. Although these algorithms can take a given approximation error into account and also can exploit knowledge about all (continous) curvature properties of the surface, they usually generate meshes of rather poor quality regarding the aspect ratio of triangles.

The resulting degenerate faces are prohibitive for numerically stable computations since, e.g., no robust face areas or normals vectors can be derived from them [Botsch and Kobbelt 2001b]. As a consequence, these surfaces have to be *remeshed* to give an approximation satisfying a specified error tolerance but in addition providing superior mesh quality. Feature sensitivity in this case aims at the preservation of features in this resampling process. The goal of the remeshing is therefore to create an as regular as possible triangulation that also exactly samples the relevant geometric features of the underlying geometry.

Obtaining a regular triangulation can be split into a *geometric* and a *topological* optimization process. Geometric regularity refers to an even distribution of vertices on the surface, leading to edges of all about the same lengths. Similar to [Turk 1992] a particle-system on the surface based on global tangential relaxation and repulsion leads to evenly distributed samples. Edge collapsing and edge splitting operators ensure that all edge lengths stay between two user specified tresholds ε_{min} and ε_{max} [Kobbelt et al. 2000]. Following Euler's formula topological regularity requires the vertex valences to be close to six. An edge flip based optimization helps to decrese the valence excess to accomplish this constraint [Hoppe et al. 1993].

While this remeshing strategy results in very regular meshes it is still useless without feature preservation (cf. Fig. 9, top). In order to snap vertices to feature edges and feature corners we enhance the particle system approach by *feature attraction* forces [Vorsatz et al. 2001]. Computing a hierarchical curvature field [Taubin 1995a; Meyer et al. 2002] on the surface and attracting mesh vertices to features following the gradient of this field finally results in a fully automatic procedure to exactly sample features of the surface geometry (cf. Fig. 9, bottom).

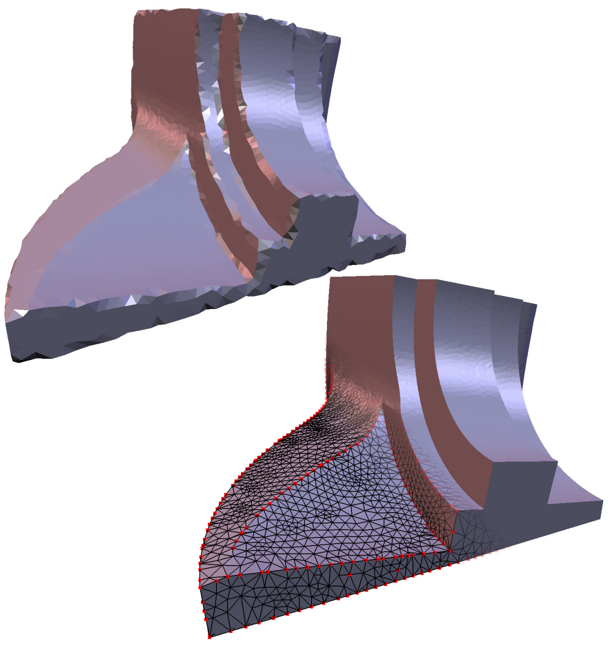

Figure 9: Regular remeshing of given meshes leads to severe alias-error without any feature-preservation mechanism (left). Using a hierarchical curvature gradient field to attract and snap vertices onto features effectively removes sampling artifacts (right).

6 Conclusion

We have shown that using triangle meshes in sophisticated downstream applications such as numerical simulations requires a high-quality approximation and alias-free representation of surfaces and feature regions. Highly curved features typically are the most significant surface parts for many numerical simulations. Unfortunately they are also the surface regions most difficult to approximate without normal noise.

As a consequence there is a growing demand for feature-sensitive algorithms in the field of mesh processing. We have presented recently developed feature-aware algorithms for isosurface extraction, blend-region resampling and diffusion based remeshing.

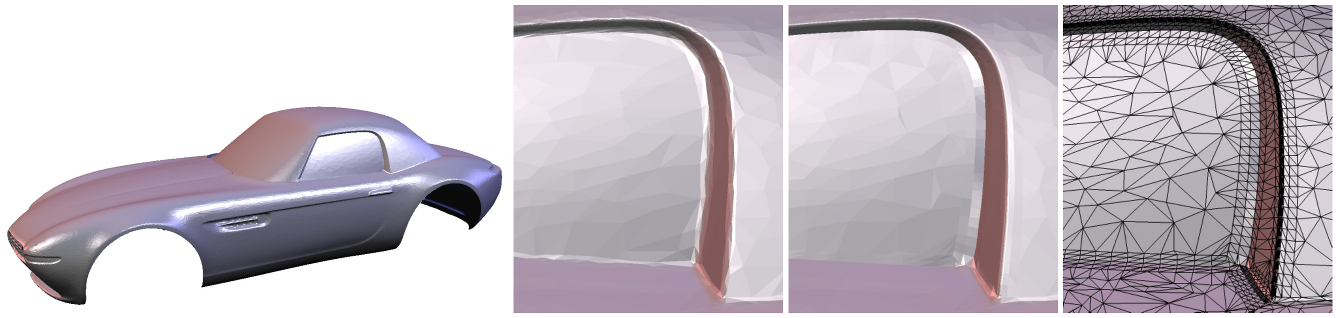

Figure 10: We applied the surface anti-aliasing technique to a detailed BMW Z8 model which is supposed to be used for CFD simulation. The normal noise in the vicinity of the feature regions of the decimated model can cause severe numerical instabilities. In the remeshed feature regions around the driver's window the normal noise has effectively been removed.

References

ARVO, J., AND KIRK, D. 1989. *An introduction to Ray Tracing*. Academic Press, ch. A Survey of Ray Tracing Acceleration Techniques, 201–262.

BERNARDINI, F., MITTLEMAN, J., RUSHMEIER, H., SILVA, C., AND TAUBIN, G. 1999. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics 5*, 4, 349–359.

BLOOMENTHAL, J., BAJAJ, C., BLINN, J., CANI-GASCUEL, M.-P., ROCKWOOD, A., WYVILL, B., AND WYVILL, G. 1997. *Introduction to implicit surfaces*. Morgan Kaufmann Publishers.

BOTSCH, M., AND KOBBELT, L. 2001. Resampling feature and blend regions in polygonal meshes for surface anti-aliasing. In *Eurographics 2001 Conference Proceedings*, 402–410.

BOTSCH, M., AND KOBBELT, L. 2001. A robust procedure to eliminate degenerate faces from triangle meshes. In *Vision, Modeling, Visualization 2001 Proceedings*, 283–289.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Siggraph 1999 Conference Proceedings*, 317–324.

FRISKEN, S., PERRY, R., ROCKWOOD, A., AND JONES, T. 2000. Adaptively Sampled Distance Fields: A general representation of shape for computer graphics. In *Siggraph 00 Conference Proceedings*, 249–254.

GARLAND, M., AND HECKBERT, P. 1997. Surface simplification using quadric error metrics. In *SIGGRAPH 1997 Conference Proceedings*, 209–216.

GOLUB, G., AND VAN LOAN, C. 1996. *Matrix Computations*, 3rd ed. Johns Hopkins Univ. Press.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1993. Mesh optimization. In *Siggraph 1993 Conference Proceedings*, 19–26.

JU, T., LASASSO, F., SCHAEFER, S., AND WARREN, J. 2002. Dual Contouring of Hermite Data. In *Siggraph 02 Conference Proceedings*, 339–346.

KALRA, D., AND BARR, A. 1989. Guaranteed ray intersections with implicit surfaces. In *Siggraph 1989 Conference Proceedings*, 297–306.

KOBBELT, L., CAMPAGNA, S., AND SEIDEL, H.-P. 1998. A general framework for mesh decimation. In *Graphics Interface*, 43–50.

KOBBELT, L., BAREUTHER, T., AND SEIDEL, H.-P. 2000. Multiresolution shape deformations for meshes with dynamic connectivity. In *Eurographics 2000 Conference Proceedings*, 249–260.

KOBBELT, L., BOTSCH, M., SCHWANECKE, U., AND SEIDEL, H.-P. 2001. Feature sensitive surface extraction from volume data. In *Siggraph 2001 Conference Proceedings*, 57–66.

LEVOY, M., PULLI, K., CURLESS, B., RUSINKIEWICZ, S., KOLLER, D., PEREIRA, L., GINZTON, M., ANDERSON, S., DAVIS, J., GINSBERG, J., SHADE, J., AND FULK, D. 2000. The digital michelangelo project: 3d scanning of large statues. In *Siggraph 2000 Conference Proceedings*, 131–144.

LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: a high resolution 3D surface construction algorithm. In *Siggraph 1987 Conference Proceedings*, 163–170.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2002. Discrete differential-geometry operators for triangulated 2-manifolds. In *VisMath 2002 Conference Proceedings*.

MONTANI, C., SCATENI, R., AND SCOPIGNO, R. 1994. Discretized marching cubes. In *IEEE Visualization 1994 Conference Proceedings*, 281–287.

MONTANI, C., SCATENI, R., AND SCOPIGNO, R. 1994. A modified look-up table for implicit disambiguation of Marching Cubes. In *The Visual Computer*, vol. 10, 353–355.

MORETON, H. P., AND SÉQUIN, C. H. 1992. Functional optimization for fair surface design. In *Siggraph 1992 Conference Proceedings*, 167–176.

NIELSON, G., AND HAMANN, B. 1991. The asymptotic decider: resolving the ambiguity in Marching Cubes. In *Visualization 91*, 83–91.

TAUBIN, G. 1995. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *Proceedings of the 5th Int. Conf. on Computer Vision*, 902–907.

TAUBIN, G. 1995. A signal processing approach to fair surface design. In *Siggraph 1995 Conference Proceedings*, 351–358.

TAUBIN, G. 2000. Geometric signal processing on polygonal meshes. In *Eurographics 00 State of the Art Reports*.

TURK, G. 1992. Re-tiling polygonal surfaces. In *Siggraph 1992 Conference Proceedings*, 55–64.

VORSATZ, J., RÖSSL, C., KOBBELT, L., AND SEIDEL, H.-P. 2001. Feature Sensitive Remeshing. In *Eurographics 2001 Conference Proceedings*, 393–401.