Robust Feature Classification and Editing

Yu-Kun Lai[a], Qian-Yi Zhou[a], Shi-Min Hu[a], Johannes Wallner[b] and Helmut Pottmann[b]

Abstract— **Sharp edges, ridges, valleys and prongs are critical for the appearance and an accurate representation of a 3D model. In this paper, we propose a novel approach that deals with the global shape of features in a robust way. Based on a remeshing algorithm which delivers an isotropic mesh in a feature sensitive metric, features are recognized on multiple scales via integral invariants of local neighborhoods. Morphological and smoothing operations are then used for feature region extraction and classification into basic types such as ridges, valleys and prongs. The resulting representation of feature regions is further used for feature-specific editing operations.**

Index Terms— **feature sensitivity, remeshing, morphology, feature extraction, feature classification, feature editing.**

I. Introduction

FEATURES are important parts of geometric models. They come in different varieties: sharp edges, smoothed edges, ridges or valleys, prongs, bridges (see Figures 1 and 10) and others. The crucial role of features for a correct appearance and an accurate representation of a geometric model has led to an increasing activity in research dealing with features (see subsection I-A).

We assume the underlying surfaces to be processed are sufficiently smooth. For discrete representations like triangular meshes, we assume that they are piecewise linear approximations to smooth surfaces. In this setting, feature regions can typically be characterized by at least one high value of a principal curvature. Just as curvature depends on the scale, so do features. A sharp edge can be seen as a limit of a smoothly blended edge when the blending radius tends to zero. Roughly speaking, features are characterized by the way in which the unit surface normal varies along the surface Φ. It is therefore natural to consider the field of unit normal vectors $\mathbf{n}(\mathbf{x})$ attached to the surface point $\mathbf{x} \in \Phi$ as a vector-valued image defined on the surface. Borrowing the idea of an image manifold from Image Processing [1], one can now map each surface point $\mathbf{x}$ to a point $\mathbf{x}_f = (\mathbf{x}, w\mathbf{n})$ in $\mathbb{R}^6$. Here, w denotes a non-negative constant, whose magnitude regulates the amount of feature sensitivity and the scale on which one wants to respect features (see Section II-A). In this way, Φ is associated with a 2-dimensional surface $\Phi_f \subset \mathbb{R}^6$. By measuring distances of points and lengths of curves on Φ_f instead of Φ, we introduce a *feature sensitive (fs) metric* on the surface. It has been studied under the name *regularized isophotic metric* [2], and it has been applied to fs mathematical morphology on surfaces.

[a] Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, P.R.China
[b] Geometric Modeling and Industrial Geometry, Vienna University of Technology, Wiedner Hauptstr. 8-10/104, A-1040 Wien, Austria

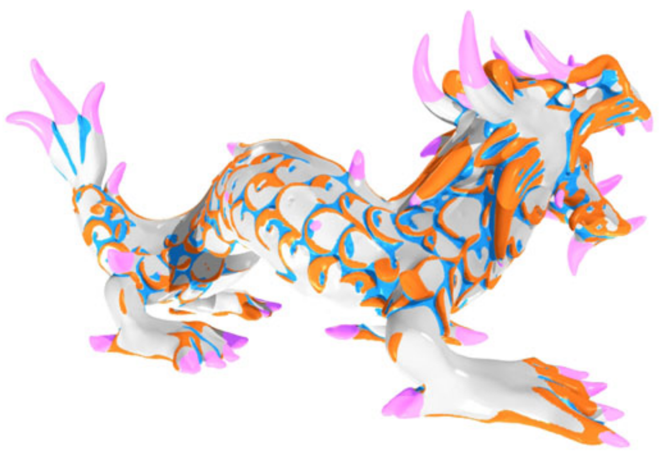

Fig. 1. Result of automatic feature classification: ridges (orange), valleys (blue), prongs (pink). See also Fig. 10.

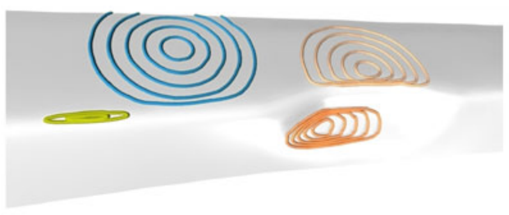

Fig. 2. Isolines of the distance from given points computed with respect to the feature sensitive metric.

As shown in Fig. 2, distances across features are much larger in the fs metric than in the ordinary Euclidean metric. This provides possibilities for fs geometry processing. An isotropic mesh of Φ_f delivers a *feature sensitive mesh* of Φ. In this paper, we show how to compute these meshes and study some of its geometric properties. Moreover, we propose fs remeshing as an effective tool for computation of integral invariants which provide good indicators for features. Most of the further processing pipeline such as feature region extraction and global feature classification can be largely based on this mesh.

A. Related Work

Feature Extraction. Features, especially feature lines (crest lines) have been discussed from a differential geometric perspective [3]. Feature extraction can then be performed by estimating differential quantities via local or global surface fitting (see [4]–[7] and the references therein). Also methods of computational differential geometry, which work directly with meshes and do not require an approximation step, have recently been extended for crest line extraction [8].

Our method for feature extraction uses integral invariants of approximate geodesic circles in the fs metric. In this sense, it is related to and motivated by work of Manay et al. [9], who noted that curvature of planar curves can be estimated in a robust way via integrals over local neighborhoods. A very similar idea is found in the work of Clarenz, Rumpf and Telea [10], [11], who recognize features by integral invariants of patches cut out from the surface by balls; the ball size determines the scale on which the features shall be detected. This method is robust and could also be used for feature detection prior to the use of our further processing. However, it does not prepare further processing as much as our approach is doing this.

Feature classification and editing. In contrast to the large body of work on local feature extraction, there is much less research on automatic classification and editing. To extract the global shape of features, ideas from mathematical morphology [12] have been extended to surfaces [13]. This work, though not based on feature sensitive metric and related techniques, has relations to our feature classification approach. However, our approach further classifies features and manipulates them according to their type. Clarenz et al. [11] propose a PDE-based algebraic multigrid algorithm to compute a set of multiscale basis functions, which are then applied for feature sensitive surface editing. Based on the characteristics of intersection curves with blowing bubbles, Mortara et al. [14] propose a method to locally classify vertices into a few types, which is different from our approach that takes global shape of features into account.

Fs parameterization. Almost any work dealing with features incorporates surface normals in some way, and there is even some work which does this in a way close to ours: Cohen-Steiner et al. [15] aim at approximating surfaces with help of Lloyd's clustering algorithm and a geometric error metric based on surface normals. However, the authors of [15] do not exploit the feature sensitive metric and the properties of the related image manifold. The purpose of combining normals with positions as well as the method used are different, and we also solve a different problem. The image manifold is also related to curvature estimation based on normal cycles [16]. Sander et al. [17] proposed a parameterization method that minimizes signal errors (approximated by first order Taylor expansion) and in this work they mention that normals can be considered as 3-dimensional signals; in a later contribution [18] they deal with fs parameterization.

Feature sensitive remeshing is a basic tool used in this paper for efficient feature extraction and processing on mesh models. We treat fs remeshing as isotropic remeshing in the fs metric. Remeshing is an active topic, and we will briefly describe some work which is related to ours. Isotropic remeshing was studied in [19]. The authors used error diffusion to initially distribute vertices, and then refined positions of vertices using global conformal parameterization and the weighted centroidal Voronoi diagram. Finally, a constrained Delaunay triangulation is applied to construct the mesh. Due to the use of a global parameterization, the approach does not work well when applied to models with complicated topology. The work in [20] and [21] improved this by using a set of local parameterizations. Botsch and Kobbelt [22] proposed an efficient and heuristic approach to generate isotropic meshes, which used iterative local modifications and reported good results on smooth models. Even with curvature-adaptive sampling [19]–[21], [23], [24] it is not easy to follow features with one large principal curvature well. In [25], a general approach based on principal curvature tensor estimation was proposed to generate isotropic triangles in umbilic regions and anisotropic quadrilaterals elsewhere.

Feature sensitive surface extraction from volume data. Kobbelt et al. [26] proposed a feature preserving approach to surface extraction from volume data. Their approach utilizes feature edge and feature point detection with some heuristics related to normal variation; then more samples are inserted in detected feature regions to improve the representation of sharp features. Ju et al. [27] improves this method, and the improved method is used for mesh repair with feature preservation [28]. A sampling pattern to represent feature and blend regions which minimizes normal variation and noise was proposed in [29]. This approach still depends on an accurate detection of feature regions. Vorsatz et al. [30] proposed another way for feature preserving remeshing, where an estimated scalar curvature field forces vertices to snap to features. Though explicit thresholding is avoided, curvature estimation is still necessary. Our approach not only deals with sharp features, but also smooth ones; it incorporates normal information, but no higher-order differential quantities.

B. Overview

Based on the feature sensitive metric, an efficient anisotropic remeshing method can be derived. By exploiting properties of the mapping between a surface Φ and its image manifold Φ_f, one can recognize and extract feature regions. The computation of integral invariants used for feature extraction can greatly benefit from the new remeshing approach. Morphology based on the feature sensitive metric and feature sensitive curve smoothing performed via Φ_f allow us to improve feature extraction results. We also show how a smoothed skeleton, constructed in the feature sensitive sense, is used for classification, for automatic modification of features, or for interactive editing operations.

The paper is organized as follows. Section II describes geometry and the algorithm for feature sensitive remeshing. The extraction of feature regions is discussed in Section III. Global descriptors such as the skeleton (with respect to the feature sensitive metric) of a feature region serve for classification and manipulation of feature regions in Section IV. Experimental results are presented and discussed in Section V and finally we conclude this paper and discuss future work in Section VI.

II. Feature Sensitive Remeshing

A. Definition and Geometric Properties

Any isotropic surface remeshing or sampling algorithm, which is not confined to the 3D case, can just as well be applied to $\Phi_f \subset \mathbb{R}^6$ instead of Φ. This is a simple approach to *feature sensitive remeshing* and *feature sensitive sampling*, respectively: it places more points in highly curved areas than in flat ones. We have verified this transfer for various remeshing and sampling algorithms [19], [21], [31], [32]. We would like to point out that the use of $\Phi_f \subset \mathbb{R}^6$ is mainly for a simple transfer from isotropic to non-isotropic remeshing algorithms and for a simple introduction of the fs metric. As will be seen from the developments given below, we can still explain everything in $\mathbb{R}^3$ via an appropriately combined processing of points and normals and thus the use of the image manifold in $\mathbb{R}^6$ does not result in any computational overhead over working in 3D.

Here we focus on *fs isotropic meshes*. They stem from nearly equilateral and equally sized triangles in $\mathbb{R}^6$. In the next sections, we will see that such meshes are very useful for feature extraction and for simplifying and accelerating the algorithms for fs morphology on surfaces which are proposed in [2]. They are also used for feature sensitive smoothing which achieves similar effects as geometric snakes [33], but in a simpler way.

The geometric properties of fs isotropic remeshing stem from the characteristics of the fs metric, especially the property of principal distortions. The first fundamental form of Φ_f has the symmetric positive definite matrix

$$M = I + w^2 III, \tag{1}$$

which is a combination of first and third fundamental form of Φ. More details about basic geometric properties can be found in [2].

A mapping between two surfaces, e.g., Φ and Φ_f, can be investigated with regard to metric distortions. In our case, the mapping α between the two surfaces is given by $\alpha : \mathbf{x}(u,v) \rightarrowtail \mathbf{x}_f(u,v)$. At some point (u,v), the *distortion* $\lambda(\dot{\mathbf{u}})$ *in direction* $\dot{\mathbf{u}} = (\dot{u},\dot{v})$ (seen as tangent direction of a curve $(u(t),v(t))$ in the parameter domain, whose image on Φ and Φ_f is

$\mathbf{c}$ and $\mathbf{c}_f$ respectively), is defined as $\lambda^2(\dot{\mathbf{u}}) := \dot{\mathbf{c}}_f^2/\dot{\mathbf{c}}^2$. It can be proved that

$$\lambda(\dot{\mathbf{u}}) = 1 - w^2 K + 2w^2 H \kappa_n(\dot{\mathbf{u}}), \tag{2}$$

where K and H are Gaussian and mean curvature, respectively, and κ_n is the normal curvature in direction $\dot{\mathbf{u}}$.

Of particular interest are the *extremal distortions*, whose geometric explanation is very simple. The affine first derivative mapping $D\alpha$ maps the unit circle k in a tangent plane of Φ (centered at the point $\mathbf{x}(u,v)$ of contact), to an ellipse k_f in the corresponding tangent plane of Φ_f at the point $\mathbf{x}_f(u,v)$. The distances of the vertices of that ellipse to its center are exactly the extremal distortions λ_1, λ_2. Equation 2 shows that the directions of extremal distortion between Φ and Φ_f are the principal curvature directions of Φ; with the principal curvatures κ_i $(i = 1, 2)$ of Φ, the extremal distortions λ_i are

$$\lambda_i^2 = 1 - w^2 K + 2w^2 H \kappa_i = 1 + w^2 \kappa_i^2. \tag{3}$$

Note that also the reverse mapping from Φ_f to Φ has extremal distortions. A unit circle l_f in a tangent plane of Φ_f is mapped under $D\alpha^{-1}$ to an ellipse l in the corresponding tangent plane of Φ. The axes of this ellipse l are in principal curvature direction, and its vertices are distance $1/\lambda_i$ to the center. These ellipses l nicely visualize the fs behavior of the metric and serve (if small) as approximate isolines of the fs distance from their center (see Fig. 2).

A near-isotropic triangulation with target length l of a smooth surface with maximum curvature κ has approximation error $\approx l^2\frac{\kappa}{2}$. We can then show that a fs isotropic mesh with target length l_f in feature space leads to a triangulation of the original surface with approximation error $\approx l_f^2\frac{\kappa}{2}\frac{1}{1+\kappa^2 w^2}$.

The computation of the image manifold Φ_f requires surface normals. For a smooth surface in any representation this is a simple task. However, we need to be careful with the following issues: the presence of noise, the scale and the presence of sharp features. The latter can be *edges* as intersection curves of smooth surfaces or *corners*, which are points, where at least three surface patches intersect or where the local shape is like the vertex of a cone. The computation of the image manifold Φ_f for triangle meshes with estimated normals can be done by simply mapping each vertex to the feature space in $\mathbb{R}^6$ while keeping the connectivity unchanged. For sharp features (edges or corners), the mapping is not one-to-one, or in the discrete case, the simple mapping will result in undersampling of the manifold embedded in $\mathbb{R}^6$, even if it is well sampled in $\mathbb{R}^3$. For most scanned models, sharp features are not well preserved, and will most likely be replaced by small blending regions, due to the limitation of acquisition or processing techniques. In this case, sharp features discussed here are not present at all. However, for CAD models or models constructed via feature preserving algorithms (e.g. [29], [30]) or feature recovery techniques (e.g. [34]), it is possible to have sharp features, though they are always rare compared to the whole model.

Noise and scale. We assume that we are given an error tolerance δ for points on the model and a parameter ε (usually small, but much larger than δ); only features of width $> \varepsilon$ shall be handled.

In the presence of noise or negligible features, we estimate normals from a neighborhood of size$\approx \varepsilon$, e.g., with local planar or quadratic fits (see e.g. [35]) and a fitting error $< \delta$. Even if this does not mean smoothing of the original data, this approach prevents a dramatic increase of the noise level in Φ_f. Moreover, marginal features - in contrast to relevant ones - do not manifest themselves in larger area of Φ_f.

If the model Φ gets scaled by a factor σ, Φ_f scales with this factor if the weight w is also multiplied by σ. Hence, w has to be judged in relation to the object size. Suitable values of w will therefore be given under the assumption that the model fits into the unit cube. Explicit estimation of the noise level is not practical for many cases and thus in practice it is chosen with the user's assistance. We have found that the same w is suitable for a wide range of models with similar scale, therefore choosing an appropriate w will not be a difficult task. Moreover, users may appreciate the freedom of choice of w, as a way to control the feature sensitivity. This is especially useful in remeshing. The values w used for the examples in this paper will be given in Section V.

Sharp features. The handling of sharp features depends on the application. In our paper, we assume the viewpoint that a sharp feature is a limit case of a smooth surface. The reader may consider sharp features smoothed with a very small blending radius. Then, a point $\mathbf{p}$ on a sharp edge $\mathbf{c} \subset \Phi$, with normals $\mathbf{n}^-$ and $\mathbf{n}^+$ of the adjacent smooth surfaces, corresponds to a circular arc $\mathbf{p}_f$ on the

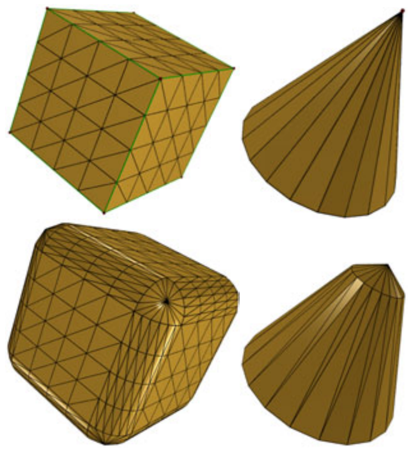

Fig. 3. The blow-up phenomenon at sharp edges and corners: Top: original mesh in $\mathbb{R}^3$. Bottom: Projection of the corresponding mesh in $\mathbb{R}^6$.

image manifold Φ_f; this arc has the endpoints $(\mathbf{p}, w\mathbf{n}^-)$ and $(\mathbf{p}, w\mathbf{n}^+)$. This is a case of surface expansion (see Fig. 3). A sharp edge is mapped to a surface region on Φ. Likewise, at a corner we have a two-dimensional set of surface normals and a corresponding spherical patch in the image manifold. Similarly, we need to insert more samples in expansion regions to ensure accurate sampling.

As discussed above, sharp edges or corners, rather than sharp blending regions, need special treatment. The detection of sharp edges and corners can be made by dihedral angle estimation; instabilities due to badly shaped triangles are avoided by the use of planar cuts orthogonal to the edge in consideration. More samples with the same position, but different normal vectors are then inserted to sample the feature regions with better accuracy (Fig. 3).

B. Remeshing Algorithm

Details about transferring the optimization algorithm in [32] to Φ_f to achieve fs isotropic remeshing will be discussed in this subsection. Advantages of this approach are its generality and its capability of handling models with arbitrary genus, with or without holes.

The remeshing process is carried out in two phases. The first phase spreads out a desired number of sampling points over the input mesh surface Φ_f and iteratively modifies their positions to achieve near-isotropic sampling. The second phase connects those sampling points with the information from the input mesh. This two-phase strategy is similar to the re-tiling approach in [23], however, we adapt it to fs isotropic remeshing, and utilize different algorithms in either phase.

The basic idea is the same as [32], but it is implemented on meshes embedded in $\mathbb{R}^6$. It is based on the minimization of an energy function, which is a sum of spring energies that push away vertices which are too close to each other. This is done by a projected gradient descent method. The manifold is the triangle mesh representation of Φ_f in $\mathbb{R}^6$; the projection onto it can be accelerated by the approximate nearest neighbor algorithm [36]. Because we are processing models containing various kinds of features, it is not ideal if we simply use Euclidean distances in $\mathbb{R}^6$ to judge their distances over the surface. We have found geodesic distances, computed by [37] for example, necessary to achieve fs isotropic remeshing of high quality.

The second phase takes the sample points and the original mesh as input, and reconstructs the output mesh with correct connectivity, keeping triangles as equilateral as possible. We suggest to recover connectivity by a set of overlapping local parameterizations. In a local region, a geodesic disk is constructed and the corresponding region is mapped to a unit circle in the parameter domain, using a feature sensitive adaptation of the parameterization method described in detail in the following paragraph. The samples falling into the disk are also mapped to the parameter domain, and a planar Delaunay algorithm can be applied to connect the local patch. Previously connected edges are treated as constraints, and thus a planar constrained Delaunay triangulation algorithm is applied. The constrained Delaunay triangulation has been implemented using the CGAL library [38]. The local parameterization scheme avoids topological problem encountered with the use of a global parameterization. The produced mesh is already nearly isotropic in almost all cases. However, in rare cases, especially near the boundary of two patches when the parameterization has relatively large distortions, we may still use a few iterations of Delaunay like edge swap to further improve the results.

The parameterization used in the reconstruction is a mapping from the 2-manifold embedded in $\mathbb{R}^6$ to a planar domain. Various parameterization methods can be adapted for this purpose. For a recent survey of parameterization methods, see [39]. The adaptation is usually straightforward, as the algorithms almost solely depend on edge lengths and related measurements. The parameterization is desirable if it can be computed efficiently, and if it is guaranteed to be a one-to-one mapping when the outer boundary is fixed to a convex polygon. Quasi-conformal parameterization is preferable, as it keeps Delaunay property after the mapping. We used the mean value parameterization proposed in [40] in our experiments, which satisfies the requirements and is an approximation of harmonic maps, which is in practice close to conformal parameterization.

Remeshing results of our method are shown in Fig. 4.

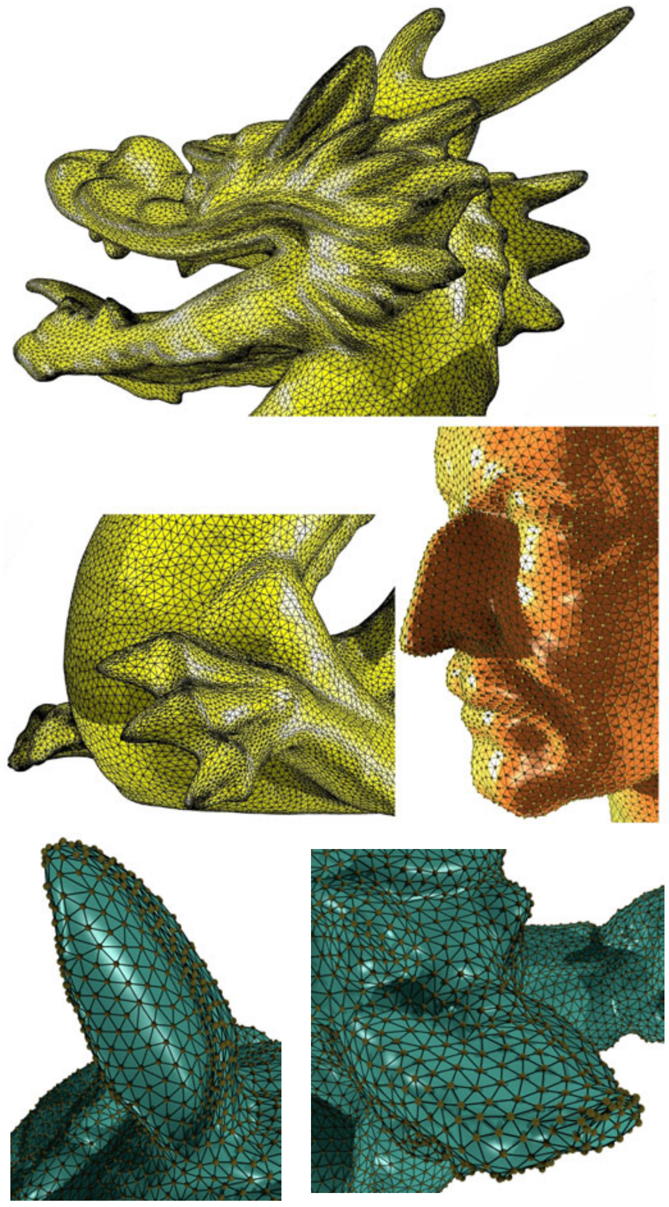
Fig. 4. Results of feature sensitive remeshing. Note that triangles are isotropic in flat regions while anisotropic in feature regions.

III. Feature Extraction

A. *Feature Filter*

Most feature detectors are based on differential invariants. Although this is reasonable for highly accurate and dense data, this is no longer the case for increasing noise or sparsity. From the viewpoint of stability, it is preferable to use *integral invariants*. This has been shown in the context of planar curve matching by Manay et al. [9] and for feature extraction by Clarenz et al. [10]. Although the latter approach would also serve very well for our purposes, we prefer the following approach which uses the result of fs remeshing.

The computation of such invariants requires the definition of *neighborhoods* of points and later also the measuring of *distances*. As to neighborhoods, Fig. 2 and the considerations of geometric properties of Φ_f lead to the idea to bound the local neighborhood of a point $\mathbf{v} \in \Phi$ by a *fs geodesic circle* $C(\mathbf{v}, r)$, i.e., the points whose fs distance from $\mathbf{v}$ equals r. The corresponding curve C_f in Φ_f is the boundary of a *geodesic fan* [41]. Thus, an efficient computation can be based on the latter reference. Moreover, fs isotropic meshes provide a good and efficient approximation of fs geodesic disk using topological disk with discrete radius r, the number of edges away from the center. More accurate geodesic disks can be approximated. Geodesic distance at each vertex to the center is first approximated, similar to previous geodesic distance computation method(e.g. [37]). Based on isotropic meshes, the computation can avoid backtracking, and as vertices on the same topological ring are similar in distance, a fixed number of rings are sufficient. Approximated geodesic disks can thus be formed by connecting points on edges that intersect with geodesic disk of desired radius. The asymptotic complexity of this approximation is identical (with a different global coefficient) to that of use of topological distance, and is still easy to implement.

As to the invariants themselves, we e.g. use the *compactness measure* $f_{com} = A/L^2$, where A and L are area and circumference of $C(\mathbf{v}, r)$ in Φ, respectively. This invariant assumes its highest values for nearly circular $C(\mathbf{v}, r)$; f_{com} is smaller if C is elongated. These facts follow from the isoperimetric inequality ($f_{com} \leq 1/(4\pi)$ for all planar curves and $f_{com} = 1/(4\pi)$ only for cicles). A curvature-like integral invariant "κ_{max}" is estimated via the shortest distance d_{min} of $C(\mathbf{v}, r)$ from $\mathbf{v}$: it satisfies $d_{min} = r/\sqrt{1 + w^2\kappa_{max}^2}$. Thus d_{min} and its dependence on r serves as an invariant.

For various values of r, we obtain invariants at different scales. A feature detector combines thresholds on f_{com} and d_{min}. We use the following *elementary feature responses*, which are defined via threshold values T_i:

—$R_1(T_1)$ is true $\iff d_{min} < T_1$. R_1 detects feature regions which exhibit small curvature radius in at least one direction.

—$R_2(T_2)$ is true $\iff f_{com} < T_2$. R_2 in combination with R_1 detects elongated features (ridges, valleys, sufficiently narrow bridges, tunnels, but also parts of prongs).

—$R_3(T_3)$ is true $\iff f_{com} > T_3$. Both R_3 and R_1 are true for the top of a prong (if it is not detected as a ridge).

Good thresholds are found via a statistical analysis of the values for f_{com} and d_{min} in a few user-selected feature and non-feature regions. Mean and deviation of these values in feature and non-feature regions can be estimated from the training data, and linear discriminant model with Gaussian distribution assumption can be used to derive the thresholds. Use $R_1(T_1)$ alone also works reasonably well in practice.

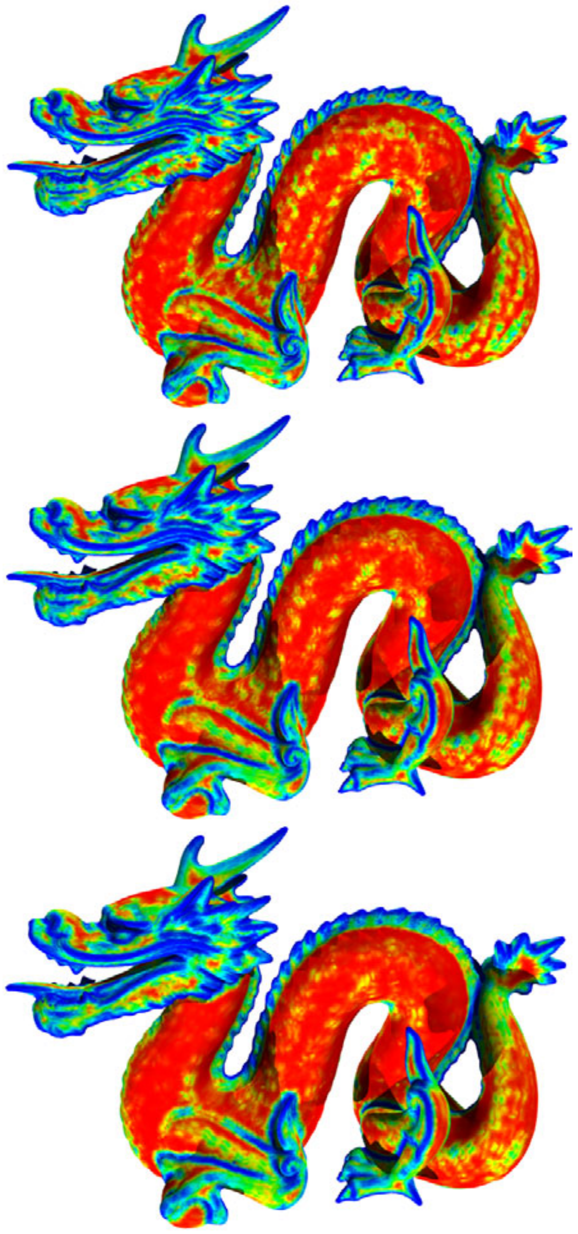
Fig. 5. Results of the feature filter using d_{min} for neighborhood size 1,2,3, increasing from top to bottom.

B. *Morphological improvement of the filter result*

The filtering described above elicits a positive response for some vertices of the mesh. Triangles with at least two positive vertices are then marked as candidates for a feature region ("black" triangles), others are not ("white" triangles). We may view this as a binary image on a triangulation and can apply methods of mathematical morphology [12] to it. Because we have a fs isotropic triangulation, it is appropriate to derive morphological operations with topological disks rather than fs geodesic disks as structuring elements. This is a special case of graph morphology [42]. We use a *closing* operation (morphological dilation followed by erosion) to fill small holes in feature regions and a *cleaning* operation to remove sufficiently small regions that are possibly caused by noise.

In some cases, however, the closing operation may happen to connect two disconnected, yet close regions. This is not desirable and may lead to misclassification in later steps. morphological *opening* (morphological erosion followed by dilation) may be performed before or after closing to work in an opposite way, i.e. avoid misleadingly connecting two regions. However, either way has its limitation. We suggest to further improve the results by using hysteresis thresholding initially proposed in image processing by Canny et al. [43]. The idea usually uses two thresholds rather than one used in simple thresholding. The first threshold is used to compute a preliminary classification of whether a face belongs to a feature region. Then fs morphological operations are used to improve the result. The opening operation does not remove faces that have higher

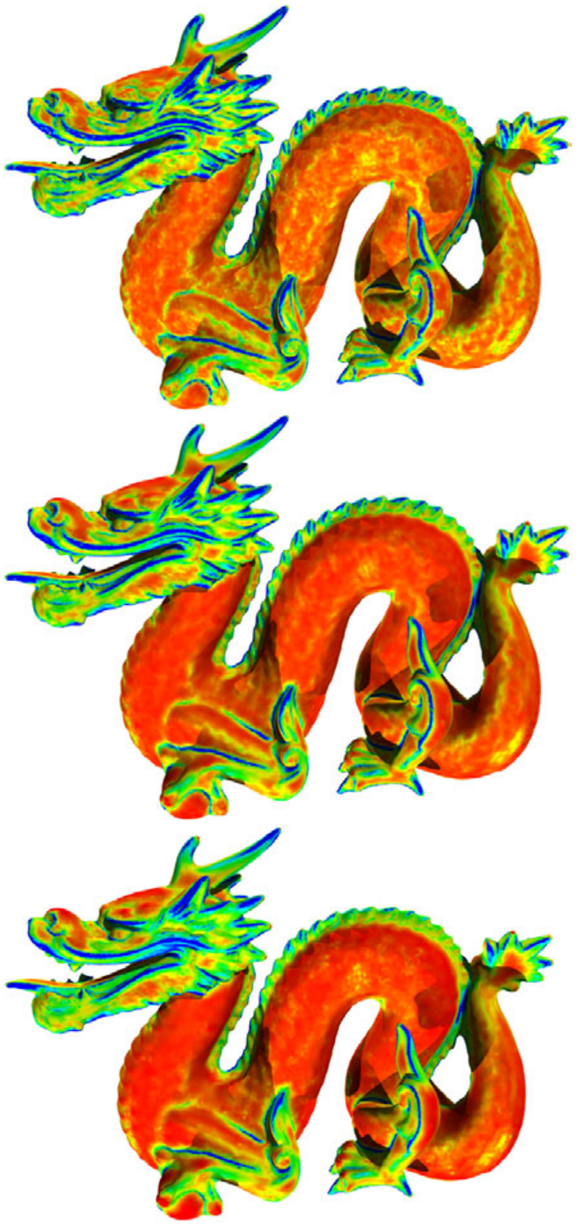

Fig. 6. Results of the feature filter based on f_{com} for neighborhood size 1,2,3, increasing from top to bottom.

feature response (using a tighter threshold) from feature regions. Similarly, the closing operation only recognizes faces that are sufficiently close to features (using a looser threshold) as feature regions. In this way, the results tend to reduce the possibility of misleading connection and disconnection of regions.

C. *Feature Sensitive Polygon Smoothing.*

The second stage of processing is to form connected components of feature triangles. Here two faces are considered adjacent if they share a common edge. Depending on the type of the feature,

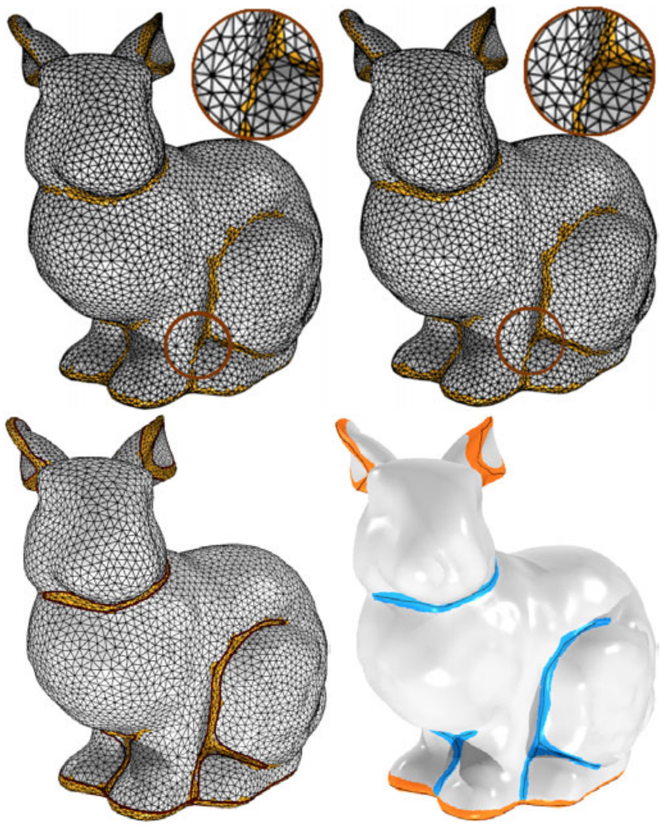

Fig. 7. Steps in the feature classification procedure: Filter response (top left); morphological closing (top right); boundary extraction (bottom left); skeleton and classification (bottom right).

we may have a certain number of closed boundary curves of a feature region. The boundary is actually a polygon on the mesh, formed by edges of the triangles which belong to the black feature region. To obtain a nicer result and better basis for further processing, we may apply smoothing to the boundary polygon. This should be done on $\Phi_f \subset \mathbb{R}^6$, in order to maintain a good alignment with the feature shape. We present fs polygon/curve smoothing; its applications go beyond the present one.

Let $\mathbf{x}_1, \dots, \mathbf{x}_n \in \Phi_f \subset \mathbb{R}^6$ be the current (closed) sequence of vertices (i.e. $\mathbf{x}_{n+1} = \mathbf{x}_1$) of the polygon which shall be smoothed. In the spirit of splines in manifolds [44], we may minimize a discretized tension spline energy,

$$F = \frac{1}{2} \sum_i \left[(\mathbf{x}_{i+1} + \mathbf{x}_{i-1} - 2\mathbf{x}_i)^2 + \lambda (\mathbf{x}_{i+1} - \mathbf{x}_i)^2 \right], \tag{4}$$

with a tension factor λ. The energy's partial derivative with respect to $\mathbf{x}_j$ is given by $\partial F / \partial \mathbf{x}_j = \Delta^4 \mathbf{x}_j - \lambda \Delta^2 \mathbf{x}_j$, where $\Delta^2 \mathbf{x}_j$ and $\Delta^4 \mathbf{x}_j$ are centralized second and fourth differences.

For a minimizer of F, which is restricted to Φ_f, at each vertex $\mathbf{x}_j$, the vector $\Delta^4\mathbf{x}_j - \lambda\Delta^2\mathbf{x}_j$ has to be orthogonal to Φ_f (by the Lagrangian multiplier rule). For the purpose of smoothing, we propose to simply apply a few iterations of a projected gradient descent as follows:

1) At the current vertex position $\mathbf{x}_j$, compute the negative gradient vector $\mathbf{v}_j := -\Delta^4\mathbf{x}_j + \lambda\Delta^2\mathbf{x}_j$.
2) Compute the orthogonal projection $\mathbf{t}_j$ of $\mathbf{v}_j$ into the tangent plane of Φ_f at $\mathbf{x}_j$.
3) Project the point $\mathbf{x}_j + s\mathbf{t}_j$ onto Φ_f where s is the step size.

For stability reasons, the step size s is controlled by the Armijo rule [45]. Fast projection onto the mesh in $\mathbb{R}^6$ can be done with the approximate nearest neighbor algorithm [36].

IV. Feature Classification and Manipulation

We describe how to classify feature regions and to capture them in a form which simplifies further processing. Inspired by modeling techniques such as ‘wires’ [46], we represent the feature region by its boundary and a skeleton.

A. *Skeleton computation.*

The skeleton of a feature region (with respect to the fs metric) could be computed by a fast marching algorithm which generates a distance field on the mesh in $\mathbb{R}^6$ [47]. However, we already have a fs isotropic mesh and thus we can compute an *approximate skeleton* in a more efficient way based on graph morphology [42]. For a similar approach, see [13].

In order to further improve the quality of this skeleton, post-pruning removes short side branches. Smoothing according to Section III-C produces a final skeleton. The difference to Section III-C is that the skeleton does not consist of a single polygon. We may view it as a collection of poly-lines to be smoothed such that connectivity is maintained. Some results are shown in Fig. 8. Note that the skeleton is in general not formed of crest lines.

B. *Feature classification.*

The developments above provide the basis for an automatic classification of feature regions. Feature

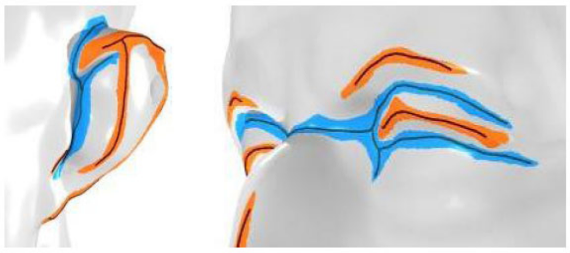

Fig. 8. Feature regions with smoothed boundary and skeleton.

classification is a wide and rather unexplored area. Here we confine ourselves to a few basic types: ridges and valleys, prongs, bridges and tunnels, as shown in Figures 1 and 10. Ridges and valleys are actually the same, and so are bridges and tunnels, if we switch from the interior of the surface to its exterior. Note that we may have aggregates of various feature region types, and that our classification is far away from being complete. We rather want to show how well the present framework helps in automatic feature classification.

Ridges and valleys have a skeleton $\mathbf{s}$ well aligned with the boundary of the feature region. The skeleton may be branching, and the feature region may have a quite complicated structure globally. However, in any case, the *skeleton ratio* $r_{skel} := l_s/l_b$ between the Euclidean boundary length l_b and the skeleton length l_s is approximately 0.5. Moreover, the Euclidean surface area A of the region is close to $l_s \cdot \theta$, with the average feature thickness θ. From this we may compute θ, which we expect to be of the same magnitude as the feature size ρ. *Feature size* means the smallest radius of curvature of a feature. To distinguish between a ridge and a valley, one may investigate points $\mathbf{p}'$ in local profile sections with planes through points $\mathbf{p} \in \mathbf{s}$ and orthogonal to $\mathbf{s}$. If the inner product $(\mathbf{p}' - \mathbf{p}) \cdot \mathbf{n}$ with the outward normal $\mathbf{n}$ at $\mathbf{p}$ is < 0 (> 0), we have local convexity (concavity) of the profile section and thus a ridge (valley).

A simple *prong* always has a single boundary loop, which is not well aligned with the skeleton $\mathbf{s}$; the latter is typically small. However, due to instabilities of the skeleton computation on a long prong with a comparably short boundary, we do not use it for classification. Rather, the surface area A is a further good indicator. Approximating the ‘shortest’ prong by a hemisphere, we see that A should be greater than $l_b^2/2\pi$. To find the top region of the prong, we search for a region with high curvature in both directions. It corresponds to a positive response of feature filters R_1 and R_3 with appropriate thresholds. A central point in this region serves as top point. This simple prong detection works well if the top region is not too far away from a spherical region, i.e., cross sections normal to the main direction of the prong are not far away from circles (high compactness measure).

There are more complicated prongs, seen e.g. at some toes of the dragon in Fig. 1, which get classified as ridges with this simple procedure. This is actually correct, if one judges such features at a smaller scale. At a larger scale, however, one may want to classify the same feature as a prong. This motivates our approach of using multiple thresholds for feature filtering. Improved feature classification uses the inclusion tree of feature regions detected at different scales.

A basic *bridge or tunnel* (without branches) has two closed boundary loops. The skeleton is not well aligned with the boundary, and there is no top region as for a prong. The triangles in the fs isotropic mesh should be elongated in a direction transversal to the boundary.

Feature Classification Algorithm. Based on these simple characteristics of elementary feature types, we implemented a feature classification algorithm (not considering bridges and tunnels) which produced the results shown in Figures 1 and 10. The basic classification decision is made by properties of elementary features discussed above. Here, we will present the algorithm structure used for classifying models. They are designed to handle some practical issues in classification like small holes in continuous regions *etc.* As the problem is itself ill posed, the classification process will make a few assumptions in decreasing order of possibilities and slightly modify feature regions (e.g. fill small holes) until a reliable result is obtained.

Pseudo-code of the Algorithm:

```
FeatureClassify (): Feature classification for fs isotropic meshes:
Input: fs isotropic mesh.
Output: detected feature regions with recognized labels.
1) Feature filter;
2) Morphological improvements;
3) Extract connected components by Breadth First Search;
4) For each component,
    a) smooth the patch boundary using fs smoothing;
    b) discard very small patches after smoothing;
    c) use FeatureClassifyPatch() to classify it;
5) Combine the results in each patch;

FeatureClassifyPatch (): Classification for a detected region:
Input: A feature region;
Output: The feature type it is supposed to be. However, this might also be a list, in cases the patch is further segmented.
1) Compute the skeleton for the patch;
2) Classify the patch into prong, ridge/valley etc. using properties of elementary features and a tight threshold;
3) If the patch is classified as ridge/valley:
    a) Try to segment the patch based on convexity;
    b) Extract segmented parts;
    c) Smooth the boundary for each part using fs smoothing;
    d) Discard patches that are too small;
    e) If more than one sub-patch exist, recursively call FeatureClassifyPatch() to classify them.
4) Fill small holes to simplify the boundary, and recursively call FeatureClassifyPatch() to classify it;
5) Classify the patch with properties of elemental features again, using a looser threshold.
```

Feature detection and classification for a specific scale is done as follows: By using the feature filter, a few faces are selected as features. Morphological operations are carried out to improve the filter result. Each connected component is considered as a feature region. The boundary is smoothed, and the smoothed boundary is snapped back to the mesh, which causes the boundary to be still made up of a continuous set of edges on the mesh. The region bounded by the updated boundary is treated as smoothed version of the feature region. We now check which type it most probably belongs to, by using characteristics discussed above. We assume that prongs have only one boundary; however, it is common in practice that ridges, valleys, or even a combination of ridges and valleys appear in a complicated configuration. There may even exist several holes in them. Simply counting the number of boundary loops is not reliable in practice. We then always check if it satisfies the two properties of ridges/valleys, namely following the feature, and satisfying the r_{skel} threshold. If these are satisfied well, they can be classified as a ridge or valley. For patches detected as ridges or valleys, which have a significant number of both convex and concave faces, a further *segmentation* is employed. This is due to the observation that ridges and valleys commonly appear side by side. Using local convexity checking described above, each feature face in the current feature region is marked as either convex or concave. Then morphological operations are employed (morphological opening followed by morphological closing for example, and now these two types will be considered as foreground and background respectively) to smooth out segmented regions. Sufficiently large connected regions of either of these two types will be formed, and then further classified (by the same process). If segmented feature regions are classified well, the segmentation is granted. If some feature region cannot be simply classified as any of these kinds, some small modifications can be performed. Very small holes with just a few non-feature faces will be filled, and such an operation will alter the statistical quantities of the feature region, thus change the classification result. Therefore, these operations must be done with care. Specifically, they can be carried out only for rejected region by the first classification. This recursive process is performed until each elementary part is either well classified, or rejected as unknown feature type.

Multiscale prong detection: For ridges and valleys, one scale of classification suffices to get reasonably good results. However, for a moderate scale, some prongs cannot be reliably detected, as it may be merged into a larger region, typically ridges. They will then be classified as part of ridges. Though this may also be correct in some sense, it would be more desirable to detect them as prongs, at least for some applications. Multi-scale prong detection can be performed, by detecting prongs at various scales (from stricter to looser thresholds). The feature regions will become larger, and prongs may be merged into nearby ridges in some scale. This phenomenon can be detected, and the largest prong before merging is fixed as the prong. In the desired scale, even the prong is directly adjacent to ridges, by this process, they can still be recognized as prongs, and the left part can well be considered as a ridge.

C. Feature manipulation.

After feature classification, a feature region F is outlined by its boundary and a central part: the latter is a point on the top for a prong and the smoothed skeleton for a ridge or valley. Therefore, we have the input to known modeling paradigms, e.g. based on a handle [48] or on wires [46]. Modifications may change the boundary of F by a dilation or erosion (performed with respect to the fs metric if appropriate). We may also change the central part, either automatically or in an interactive way and recompute the surface with known methods in the literature. In view of the wide area of possibilities, which are not really the focus of our work, we just give two examples illustrating the usefulness of our classification results and show how they work well in automatic or interactive modeling applications.

For prongs, typical operations include erosion and dilation. Erosion of a prong can make it thinner, shorter, or both. Dilation works in exactly the opposite way. It is performed by changing the boundary and/or central point, respectively. Both the central point and the boundary (as well as their near neighbors, which are required by the method to ensure smoothness) are considered as handles, and the positions of other vertices can be derived by the modeling approach in [48]. Fig. 11 shows the dilation and erosion effects for prongs of the dragon head model.

For ridges and valleys, sharpening and smoothing are useful operations. Ridge and valley sharpening can be performed by wire deformation [46]. The skeleton for a ridge or valley region may have complicated configuration, and we model each continuous segment of skeleton without branching as a wire. The skeleton part is smoothed with similar technique as boundary smoothing (but not closed), and then the smoothed result is considered as the reference curve. The wire curve is derived from moving vertices on the smoothed skeleton in the direction (for ridges) or opposite direction (for valleys) of local normals. To make the result visually better, the normals could be smoothed along the skeleton. The effective radius r is derived from the average distance of patch boundary to the nearest point on the smoothed skeleton. Only vertices that are no more than distance r away from the smoothed skeleton will be affected by the deformation. In regions affected by two or more wires, the combination formula in [46] is applied to get smooth transitions. We may choose as scaling factor s a value slightly smaller than 1 (e.g. 0.95) to make the region sharper. A similar idea can be used to achieve ridge/valley smoothing, but it works in an opposite way. To make the result smoother, a post-smoothing operator could be applied. One example of ridge sharpening is shown in Fig. 12.

V. Experimental Results

Fig. 4 shows a few results of feature sensitive remeshing. The dragon model (top and middle left, two closeups) is remeshed with $w = 0.05$, the Max Planck head model (middle right) is remeshed with $w = 0.04$, and the Armadillo model (bottom, two close-ups) is remeshed with $w = 0.03$. All the models are scaled to fit within a bounding cube of size 1. Due to this resizing, it will be better to use smaller w for complicated models. As meaningful features are relatively small, a smaller w will be sufficient to add feature sensitivity. Smoother models can usually work with relatively larger w as noise levels are lower and features are not as sharp. For models with similar scale, smoothness and noise level, the same w is sufficient to provide results of high quality. Note that in these examples, sample points are aggregated in feature regions and triangles are elongated along the feature edges. Fig. 9 shows the same rocker arm model remeshed with different weights. Note that for the purpose of feature classification, a wide range of weights are all possible, with appropriately selected thresholds.

Remeshing can be implemented efficiently. We suggest here a few improvements for performance.

The sampling before iterative optimization can be improved by error diffusion similar to the one used in [19]. We diffuse sampling errors from one face to its neighbors, keeping average number of samples almost identical in local regions. In this way, the initial guess would be a better one for isotropic remeshing. The surface area used for estimating the budget of samples should be computed in $\mathbb{R}^6$.

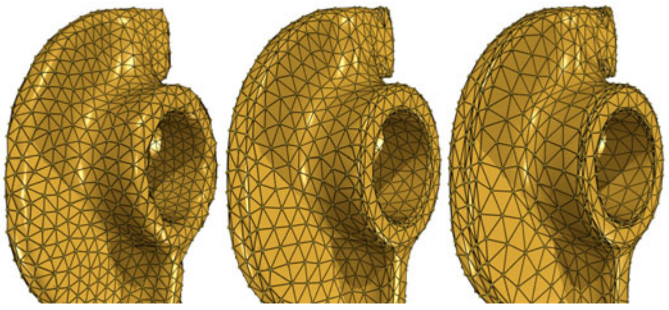
Fig. 9. Feature sensitive remeshing with different weights. Left: $w = 0$; center: $w = 0.1$, right: $w = 0.2$.

Geodesic distances are critical for achieving high quality results. However, even with an efficient estimation method, they are more expensive to compute, and we suggest to compute geodesic distances only when necessary, i.e., near sharp features. We can use a very rough dihedral based detector for this purpose, and then grow detected regions, to include almost all the regions when geodesic distances are crucial. For every vertex in such regions, geodesic information in its local neighborhood could be precomputed and stored in *window* structures [37] on a per edge base. These data need to be computed only once, and are applicable in each iteration.

The relatively expensive projection in $\mathbb{R}^6$ can also be reduced in later iterations, when most sampling points will not move much, by checking if the last projected face is still applicable in the new iteration. This check is inexpensive, and can greatly reduce the number of required ANN queries.

The remeshing time for all the models in this paper are no more than a few minutes. The most complicated example, remeshing Lucy model from about 800K triangles to about 200K triangles shown in Fig. 10(left) took 7 minutes on a Pentium IV 2.4GHz computer.

Fig. 10 gives two examples of automatic feature classification. The weight w used for remeshing is between 0.07 and 0.05, respectively. Features are extracted and classified into ridges (orange), valleys (blue) and prongs (pink). See also Fig. 1. The feature extraction and classification took about 5 seconds for the Max Planck head model shown in Fig. 8 and one minute for more complicated examples like the Lucy model.

Fig. 11 presents results of automatic dilation and erosion of prongs on the dragon head model. The features shown in this figure follows color-coding used in other figures. They are detected and classified automatically, and then used for corresponding modification. Fig. 12 shows an example of ridge sharpening. All the ridges in the Happy Buddha model are sharpened. Two close-ups of the original model and the sharpened results are on the left and right of the figure. The modification of prongs took 1 second and sharpening all the ridges took about 3 seconds.

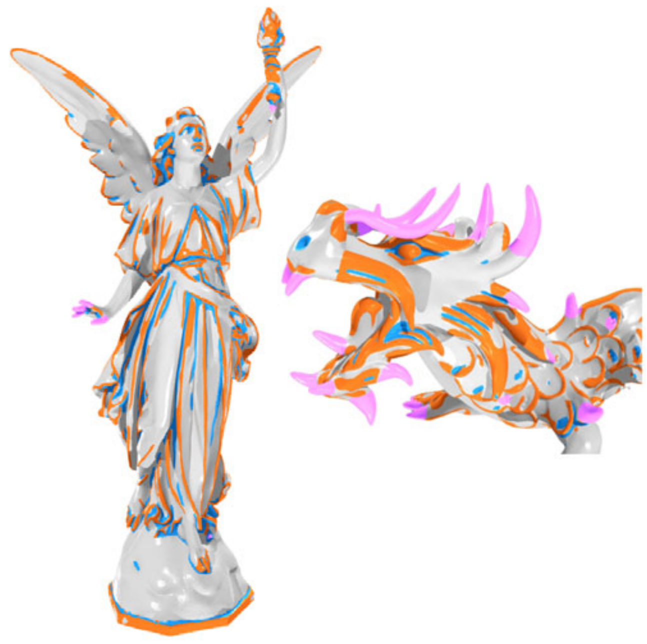

Fig. 10. Automatic detection and classification of features: ridges (orange), valleys (blue), prongs (pink). cf. Fig. 1.

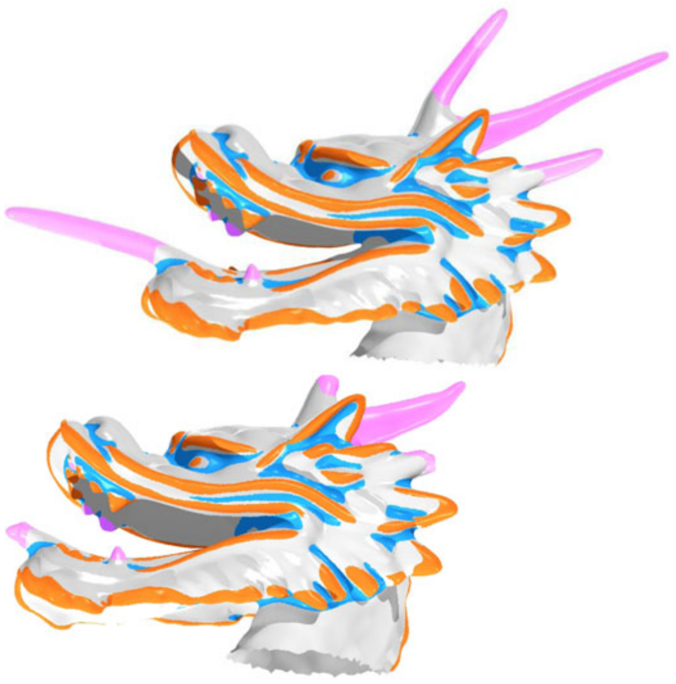

Fig. 11. Dilation (top) and erosion (bottom) of prong features.

VI. Conclusions and Future Work

Based on a feature sensitive metric and the idea of integral invariants, we have presented a robust feature extraction and classification algorithm. The by-product of feature sensitive remeshing is itself a useful tool for efficient computation of some integral quantities that reflect local characteristics of surface. We believe that the basic ideas in this paper can be applied to more applications: model segmentation or patch layout for fitting seem to be promising directions.

Our future research will mainly concentrate on feature extraction, classification and surface matching using further geometric invariants and statistical methods from pattern classification [49].

Acknowledgements

Models in this paper are courtesy of Cyberware, Stanford University, and the Max-Planck-Institut für Informatik. The authors would like to thank the reviewers for their helpful comments. This work was partially supported by the Natural Science Foundation of China (Project Numbers 60225016, 60333010, 60321002) and the National Basic Research Project of China (Project Number 2002CB312101). Part of this research has been carried out within the Competence Center *Advanced Computer Vision* and has been funded by the K*plus* program. This work was also supported by the Austrian Science Fund (Grant No. S9206).

References

[1] R. Kimmel, R. Malladi, and N. Sochen, “Images as embedded maps and minimal surfaces: movies, color, texture and volumetric medical images.” *Intl. J. Computer Vision*, vol. 39, pp. 111–129, 2000.

[2] H. Pottmann, T. Steiner, M. Hofer, C. Haider, and A. Hanbury, “The isophotic metric and its application to feature sensitive morphology on surfaces,” in *Proceedings of ECCV 2004, Part IV*, vol. 3021 of Lecture Notes in Computer Science. Springer, 2004, pp. 560–572.

[3] I. R. Porteous, *Geometric Differentiation for the Intelligence of Curves and Surfaces*. Cambridge University Press, 1994.

[4] F. Cazals and M. Pouget, “Estimating differential quantities using polynomial fitting of osculating jets,” pp. 177–178, 2003.

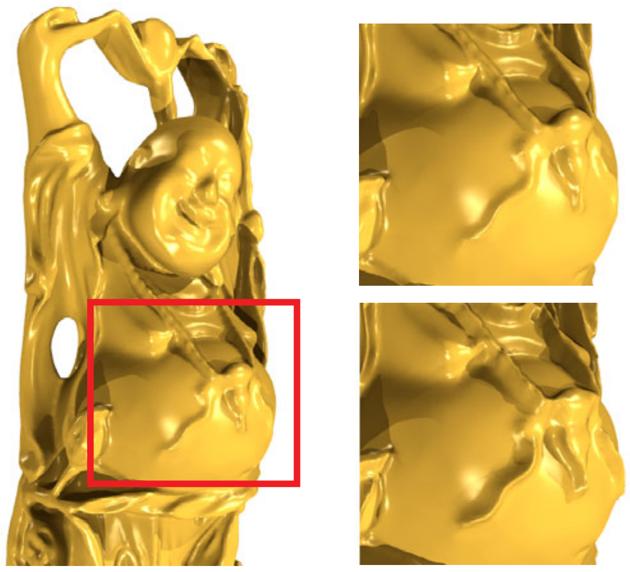

Fig. 12. Results of sharpening ridges on the Happy Buddha model. Left: original model; Top right: a close-up of original model; Bottom right: with ridges sharpened.

[5] Y. Ohtake, A. Belyaev, and H.-P. Seidel, "Ridge-valley lines on meshes via implicit surface fitting," in *Proceedings of SIGGRAPH*, 2004, pp. 609–612.

[6] G. Stylianou and G. Farin, "Crest lines for surface segmentation and flattening," *IEEE Trans. Visualization and Computer Graphics*, vol. 10, pp. 536–544, 2004.

[7] S. Yoshizawa, A. Belyaev, and H.-P. Seidel, "Fast and robust detection of crest lines on meshes," in *Proc. of ACM Symposium on Solid and Physical Modeling*, 2005.

[8] K. Hildebrandt, K. Polthier, and M. Wardetzky, "Smooth feature lines on surface meshes," in *Eurographics Symposium on Geometry Processing*, 2005, pp. 85–90.

[9] S. Manay, B.-W. Hong, A. J. Yezzi, and S. Soatto, "Integral invariant signatures," in *Proceedings of ECCV*. Springer, 2004, pp. 87–99.

[10] U. Clarenz, M. Rumpf, and A. Telea, "Robust feature detection and local classification for surfaces based on moment analysis," *IEEE Trans. Visualization and Computer Graphics*, 2004.

[11] U. Clarenz, M. Rumpf, M. A. Schweitzer, and A. Telea, "Feature sensitive multiscale editing on surfaces," *Visual Computer*, vol. 20, no. 5, pp. 329–343, 2004.

[12] H. Heijmans, *Morphological image operators*. Academic Press, Boston, 1994.

[13] C. Rössl, L. Kobbelt, and H.-P. Seidel, "Extraction of feature lines on triangulated surfaces using morphological operators," in *Proceedings of the 2000 AAAI Symposium on Smart Graphics*, 2000, pp. 71–75.

[14] M. Mortara, G. Patané, M. Spagnuolo, B. Falcidieno, and J. Rossignac, "Blowing bubbles for multi-scale analysis and decomposition of triangle meshes," *Algorithmica*, vol. 38, no. 1, pp. 227–248, 2003.

[15] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," in *Proceedings of SIGGRAPH*, 2004, pp. 905–914.

[16] D. Cohen-Steiner and J.-M. Morvan, "Restricted delauney triangulations and normal cycles," in *Proceedings of 19th ACM Symposium on Computational Geometry*, 2003, pp. 237–246.

[17] P. V. Sander, S. J. Gortler, J. Snyder, and H. Hoppe, "Signal-specialized parameterization," in *Proceedings of Eurographics Workshop on Rendering*, 2002, pp. 87–100.

[18] G. Tewari, J. Snyder, P. Sander, S. Gortler, and H. Hoppe, "Signal-specialized parameterization for piecewise linear reconstruction," in *Eurographics Symposium on Geometry Processing*, 2004, pp. 57–66.

[19] P. Alliez, Éric Colin de Verdière, O. Devillers, and M. Isenburg, "Isotropic surface remeshing," in *Proceedings of Shape Modeling International*, 2003, pp. 49–58.

[20] V. Surazhsky, , and C. Gotsman, "Explicit surface remeshing," in *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2003, pp. 20–30.

[21] V. Surazhsky, P. Alliez, and C. Gotsman, "Isotropic remeshing of surfaces: a local parameterization approach," in *Proceedings of 12th International Meshing Roundtable*, 2003.

[22] M. Botsch and L. Kobbelt, "A remeshing approach to multiresolution modeling," in *Proceedings of Eurographics Symposium on Geometry Processing*, 2004, pp. 189–196.

[23] G. Turk, "Re-tiling polygonal surfaces," in *Proceedings of SIGGRAPH*, E. E. Catmull, Ed., 1992, pp. 55–65.

[24] P. Alliez, M. Meyer, and M. Desbrun, "Interactive geometry remeshing," in *Proceedings of SIGGRAPH*, 2002, pp. 347–354.

[25] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun, "Anisotropic polygonal remeshing," in *Proceedings of SIGGRAPH*, 2003, pp. 485–493.

[26] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel, "Feature sensitive surface extraction from volume data," in *Proceedings of SIGGRAPH*, 2001, pp. 57–66.

[27] T. Ju, F. Losasso, S. Schaefer, and J. Warren, "Dual contouring of hermite data," in *Proceedings of SIGGRAPH*, 2002, pp. 339–346.

[28] T. Ju, "Robust repair of polygonal models," in *Proceedings of SIGGRAPH*, 2004, pp. 888–895.

[29] M. Botsch and L. Kobbelt, "Resampling feature and blend regions in polygonal meshes for surface anti-aliasing," *Computer Graphics Forum*, vol. 20, no. 3, pp. 402–410, 2001.

[30] J. Vorsatz, C. Rossl, L. P. Kobbelt, and H.-P. Seidel, "Feature sensitive remeshing," *Computer Graphics Forum*, vol. 20, no. 3, p. 393, 2001.

[31] M. Pauly, M. Gross, and L. Kobbelt, "Simplification of point-sampled surfaces," in *Proceedings of IEEE Visualization*, 2002, pp. 163–170.

[32] A. Witkin and P. Heckbert, "Using particles to sample and control implicit surfaces," in *Proceedings of SIGGRAPH*, 1994, pp. 269–277.

[33] Y. Lee and S. Lee, "Geometric snakes for triangular meshes," *Computer Graphics Forum*, vol. 21, no. 3, pp. 229–238, 2002.

[34] M. Attene, B. Falcidieno, J. Rossignac, and M. Spagnuolo, "Edge-sharpener: recovering sharp features in triangulations of non-adaptively re-meshed surfaces," in *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2003, pp. 62–69.

[35] T. Varady and R. Martin, *Handbook of CAGD*. North Holland, 2002, ch. Reverse engineering, pp. 651–681.

[36] D. Mount, "ANN library," 1998.

[37] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler, and H. Hoppe, "Fast exact and approximate geodesics on meshes," in *Proceedings of SIGGRAPH*, 2005, pp. 553–560.

[38] "CGAL: Computational geometry algorithm library, ver. 3.1," 2004.

[39] M. S. Floater and K. Hormann, *Advances in Multiresolution for Geometric Modelling*. Springer, Berlin, Heidelberg, 2005, ch. Suface parameterization: a tutorial and survey, pp. 157–186.

[40] M. S. Floater, "Mean value coordinates," *Computer Aided Geometric Design*, vol. 20, no. 1, pp. 19–27, 2003.

[41] S. Zelinka and M. Garland, "Similarity-based surface modelling using geodesic fans," in *Proceedings of Eurographics Symposium on Geometry Processing*, July 2004, pp. 209–218.
[42] L. Vincent, "Graphs and mathematical morphology," *Signal Processing*, vol. 16, pp. 365–388, 1989.
[43] J. A. Canny, "A computational approach to edge detection," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 679–698, 1986.
[44] M. Hofer and H. Pottmann, "Energy-minimizing splines in manifolds," in *Proceedings of SIGGRAPH*, 2004, pp. 284–293.
[45] C. T. Kelley, *Iterative Methods for Optimization*. SIAM, 1999.
[46] K. Singh and E. Fiume, "Wires: a geometric deformation technique," in *Proceedings of SIGGRAPH*, 1998, pp. 405–414.
[47] R. Kimmel and J. A. Sethian, "Computing geodesic paths on manifolds," *Applied Mathematics*, vol. 95, pp. 8431–8435, July 1998.
[48] M. Botsch and L. P. Kobbelt, "An intuitive framework for real-time freeform modeling," in *Proceedings of SIGGRAPH*, 2004, pp. 630–634.
[49] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley Interscience, 2001.

Yu-Kun Lai is a PhD student in Department of Computer Science and Technology at Tsinghua University. He got his bachelor's degree in Computer Science at Tsinghua University in 2003. His research interests include Computer Graphics, geometry processing and CAGD.

Qian-Yi Zhou is a master student in Department of Computer Science and Technology at Tsinghua University. He received his bachelor's degree in Computer Science from Tsinghua University in 2005. His research interests include Computer Graphics, Geometry Modeling and CAGD.

Shi-Min Hu is currently a professor of computer science at Tsinghua University. His research interests include digital geometry processing, video-based rendering, rendering, computer animation, and computer-aided geometric design. He obtained his Ph.D. in 1996 from Zhejiang University. He is on the editorial boards of Computer Aided Design.

Johannes Wallner is a member of the Institute of Discrete Mathematics and Geometry at TU Wien in Vienna. He received his Ph.D. from TU Wien in 1997 and researches in geometry. He coauthored the monograph *Computational Line Geometry* published by Springer in 2001. A recent focus of his work is smoothness analysis of nonlinear subdivision schemes, tolerance analysis, and geometry processing. See also the web pages at *http://www.geometrie.tuwien.ac.at/wallner*.

Helmut Pottmann is professor of Geometry at Vienna University of Technology. His research interests include classical geometry and its applications, especially to Computer Aided Geometric Design, Computer Vision, industrial geometry, kinematics, and the relations between geometry, numerical analysis and approximation theory. A list of recent publications can be found at *http://www.geometrie.tuwien.ac.at/pottmann*.