Spider: A robust curvature estimator for noisy, irregular meshes

Technical report CSRG-531, Dynamic Graphics Project, Department of Computer Science, University of Toronto, ©September 2005

Patricio Simari*
Dynamic Graphics Project
University of Toronto

Karan Singh†
Dynamic Graphics Project
University of Toronto

Hans Pedersen‡
Metris, Inc.

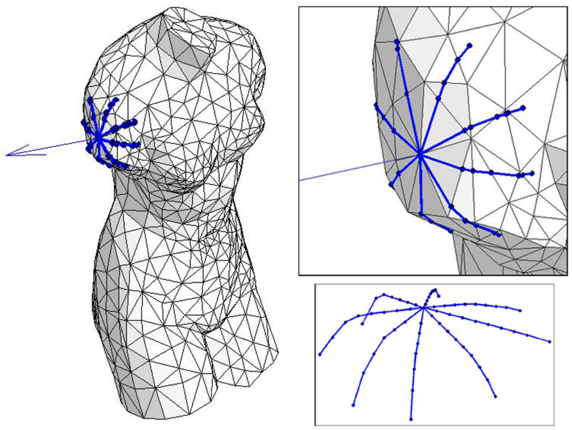

Figure 1: Example of polylines of intersection found at a given point and normal for an angular step size of $\frac{\pi}{4}$. Bottom right: the regularly resampled polylines.

Abstract

Surface curvature properties are often as important as surface position in understanding shape. Curvature properties are typically computed at mesh vertices by operating on an associated ring neighbourhood of faces. Such approaches are not well suited to noisy, non-uniformly sampled meshes with irregular tessellations. In this paper, we present a principled approach to curvature estimation that can be computed at any point on the unprocessed mesh surface and is robust to noisy and irregular surface sampling and tessellation. The approach achieves user controlled smoothing of the curvature field thus also making it robust to noisy sampling. The nature of the smoothing, in contrast to current approaches, is determined by intuitive user-specified parameters and becomes naturally anisotropic in the vicinity of feature edges. We show this approach to provide better visual results than ring neighbourhood approaches on noisy, non-uniformly sampled meshes with irregular tessellations.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling— [G.2.3]: Discrete Mathematics—Applications

1 Introduction

Polygon based meshes are among the oldest geometric representation in Computer Graphics and currently the most prevalent. Dense meshes are more than often faceted approximations of a smooth surface [Levoy et al. 2000]. Higher order surface derivatives such as normal and curvature on such meshes are as important as surface position in the way we perceive and express shape [DeCarlo et al. 2003]. Aggregates of these and other such geometric attributes are often called surface-features, which provide a higher level description of shape. The definition, extraction, filtering and editing of surface-features on these meshes is an active area of research. The robust estimation of geometric attributes such as normals and curvature for a given mesh representation of the surface is thus critical for most surface-feature related applications such as feature extraction [Ohtake et al. 2004], surface fitting [Moreton and Sequin 1992], mesh simplification [Heckbert and Garland 1999], surface fairing [Desbrun et al. 1999], multiresolution mesh editing [Zorin et al. 1997] and suggestive contours for object illustration [DeCarlo et al. 2003].

Despite its importance and pervasiveness, there is no consensus as pointed out by Meyer et. al [Meyer et al. 2003] on the best estimation for even the simplest of attributes, the surface normal, on a dense mesh approximation of a smooth surface. One reason for this is that the smooth surface represented by a discrete mesh is not unique. In fact a number of the practical surfaces represented by dense meshes are only piecewise smooth themselves. Another reason is that practical meshes often contain signal noise that gets magnified on normal and curvature calculations but many surface denoising algorithms [Desbrun et al. 1999] depend upon robust normal and curvature estimation. Rather than attempt to achieve a perfectly accurate attribute computation for a given class of meshes approximating some continuous surface, our philosophy is to develop an algorithm for general meshes that satisfies the attribute properties required by various surface-feature related applications. These desirable properties for any algorithm computing the geometric attributes are as follows:

- Consistent accuracy: The attribute computed at any given point on the mesh should be a consistently accurate approximation of the corresponding attribute on the continuous surface.

- Continuity: The computed attribute should faithfully reproduce discontinuities and extrema of the attribute on the original surface and be continuous everywhere else. This is particularly important since continuity errors in attributes such as a surface normal are magnified in higher order derivatives like curvature.

- Frequency: The computation should be capable of producing attribute values within given frequency bands. This is essential for the continuity property as described above on noisy meshes.

- Robustness: The computation should be relatively insensitive to changes in mesh connectivity. The computation should also be independent of mesh scale for scale invariant attributes.

- Efficiency and scalability: Given the ever-increasing size of data samples used to represent geometry, the attribute computation should be localized and efficient.

*e-mail: psimari@cs.toronto.edu
†e-mail: karan@dgp.toronto.edu
‡e-mail: hans@metris.com

1.1 Previous Work

Rendering a faceted mesh as a smooth surface requires the definition of a continuous normal vector across the mesh. This is usually accomplished using interpolated vertex normals [Phong 1975] within a face. While interpolating vertex normals obtained by averaging the face normals incident to a vertex has worked well historically, current modeling techniques, shape acquisition and manufacturing technology typically generate dense mesh approximations of surfaces, where the vertex connectivity of the mesh is an arbitrary artifact of the creation process [Levoy et al. 2000]. Surface derivative estimation algorithms for such meshes that are sensitive to vertex connectivity are at a conceptual disadvantage. We classify existing surface derivative estimation algorithms here into vertex connectivity based and vertex connectivity independent.

Almost all work on surface derivative estimation on meshes has been vertex connectivity based. In these approaches an n-ring neighborhood of vertices, faces and edges adjacent to a vertex are used to compute the surface derivatives at a vertex. The attribute at an arbitrary point on the mesh is then computed by interpolating the vertex attributes of the mesh face to which the point belongs. The surface attribute at a vertex is often computed as weighted average of the attributes of neighbouring faces, edges or vertices. For example, vertex normals are often defined as a weighted average of the normals of adjacent faces of a mesh, by the incident angle [Thurmer and Wuthrich 1998] or surface area [Taubin 1995] of each face at the vertex, or derived so the mesh locally approximates a sphere [Max 1999].

As described by Rusinkiewicz [Rusinkiewicz 2004], approaches to computing the curvature tensor can be broadly classified as based on patch fitting, normal curvature or tensor averaging. Patch fitting methods such as [Hamann 1993], [Cazals and Pouget 2003], [Goldfeather and Interrante 2004] fit an analytic surface to a local region of mesh points and then compute the curvature analytically from the surface. The approach shows good results for uniformly sampled meshes but can fall prey to degenerate vertex configurations that are well approximated by many simple analytic surfaces. Normal curvature approaches such as [Taubin 1995] and more recently [Meyer et al. 2003] estimate the normal curvature along edges incident to a vertex and then extract the curvature tensor using an eigen decomposition. Meyer *et. al* [Meyer et al. 2003], provide insight and examples of where prior normal curvature approaches produce unsatisfactory results. Their approach computes the normal at a vertex by solving the Laplace-Beltrami operator for the mean curvature normal vector for a one-ring neighbourhood. Tensor averaging approaches [Cohen-Steiner and Morvan 2003] compute the average of the curvature tensor over a small area, like the Voronoi region around a vertex. Such approaches are less sensitive to degenerate and sliver faces but still work best for regular uniformly sampled meshes. The approach of Rusinkiewicz [Rusinkiewicz 2004] is based on computing a curvature tensor for every face and then averaging the tensor appropriately to compute the curvature at mesh vertices.

In general, ring neighbourhood approaches as a discrete differential element around a vertex are problematic for highly non-uniform meshes. None of the approaches described deal directly with noisy meshes leaving any noise to be handled separately by surface fairing techniques as a preprocessing step or as a postprocessing step where the field of normal and curvature values can be filtered as shown by Taubin [Taubin 2001]. By definition, morphological operations such as diagonal flipping will change the vertex normal computed by these approaches, to a varying degree.

1.2 Motivation

Comparatively there has been little work on the computation of surface derivatives that is not based on a ring neighborhood around a vertex. This is in no small measure due to efficiency considerations since most approaches that are global [Zhang and Fiume 2003] or based on continuous surface fitting are likely to be computationally expensive. We note, however, that the connectivity of a mesh, in addition to defining a surface manifold over a set of vertices, provides an efficient data structure to track the growth of geometric elements on that surface. In particular, we are able to locally grow a fixed length polyline of intersection between the mesh and an analytic primitive such as a sphere or plane very efficiently. We also note that the surface normal at a point on a continuous surface can be approximated the normal to the asymptotic plane of the intersection curve between the surface and a ball of shrinking radius, centered at the point. Finally we observe that given a surface normal, the normal curvature in any direction can be calculated by fitting an analytic curve to the polyline of intersection between the normal plane and the mesh.

We thus propose a hybrid of a normal curvature and patch fit approach, where we estimate surface normal and curvature at any arbitrary point on the mesh by computing a number of intersection curves of the mesh with analytic primitives which are resampled regularly, so as to be largely insensitive to mesh resolution or specific vertex connectivity that ring neighbourhood approaches suffer from. These intersection curves are then fit to analytic curves and used to search for the two principle curvature directions. The curve fitting has fewer constraints and ambiguities that can plague analytic surface fitting approaches. The length of the intersection curves can also be completely independent of the tessellation resolution scale of the mesh and provides users with an intuitive handle over the feature sizes on the mesh they wish to capture. While our approach is clearly more computationally expensive than most ring neighbourhood approaches, it is usable at interactive rates for dense meshes.

2 Approach

2.1 Estimating normal at a surface point

Given a point on the mesh surface and a geodesic radius, we can estimate the surface normal at said point by finding the intersection of the mesh with a sphere of said radius centered at the point. The normal estimate can then be obtained by finding the best fit plane to the resulting intersection. Figure 2 illustrates this idea.

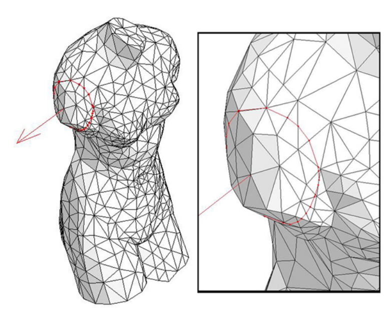

Figure 2: Estimating surface normal at a mesh point by finding best fit plane to intersection of mesh with sphere centered at the point.

2.2 Estimating normal curvature in a specified direction

Given a mesh point and a direction we wish to estimate the associated curvature value. Let us consider the plane determined by said point, direction, and the point's current surface normal estimate. We may find the intersection of this plane with the given mesh by finding its edge intersections. We construct a polyline away from the point in either direction that is contained in the plane, until its total length surpasses a user-specified geodesic radius. The last segment of the polyline is then shortened as needed so the total length of the polyline matches the geodesic radius exactly. This can be done easily and efficiently (in time linear to the radius) by using edge-face information. This polyline can now be seen as an approximation of the one dimensional curve that is the local surface/plane intersection. Note that the polyline's segments can be of varying length as a result of the varying sizes of the intersected faces. To address this, we resample it at regular domain intervals.

We now wish to estimate the mesh's curvature along this planar intersection. Working in a local coordinate system where the point in question is the origin, the point's estimated normal is the up vector, and the polyline's points all lie in the $x = 0$ plane, we fit an n-degree polynomial to the resulting (y,z) pairs. We remove the constant term so as to ensure that the polynomial passes through the origen (and thus the point in question) and add a linear constraint so that the tangent at the origin is as close to zero as possible, to reflect the fact that the local z axis should be normal to the curve at the origin. This results in a linear system which can be efficiently solved in closed form.

Thus, the surface's curvature in the direction is estimated as the polynomial's curvature at the origin and the normal is derived from the polynomial's tangent, both through simple calculation.

Figure 1 shows the polylines of intersection found for eight regularly spaced directions. The curves' appearance motivates the operator's name: 'spider'.

2.3 Estimating principal curvature values and curvature normal

In order to estimate the principal curvature directions we use the above method to estimate the surface's curvature at regular angular intervals and return the pair of orthogonal curvatures whose difference is largest in magnitude. The estimated normal is the average of the principal direction normals.

2.4 Iterating the process

The obtained surface normal estimate can be used to repeat the above process until the difference with the previous estimate is sufficiently small or a maximum number of iterations are computed.

Figure 3 shows the result of applying this iterative scheme to a tessellated unit cube. Here we use a purposefully relatively large geodesic radius of .5 to show how the normal field is smoothed. We also illustrate the robustness of the approach to varying face sizes by comparing the results for three increasingly finer tessellations.

2.5 Anisotropy

The operator will naturally become anisotropic in the vicinity of mesh holes. If no more faces are found along which to continue searching for intersection, the radius in that direction will necessarily be shorter.

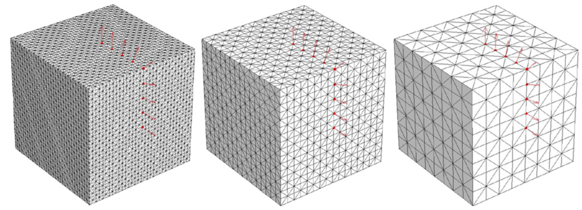

Figure 3: Illustration of normal smoothing: result of applying iterative scheme to a tessellated unit cube using geodesic radius of .5. The final average difference between corresponding surface normals across tessellations is approximately .0076 radians.

Another reason to retard the growth of a spider limb in a given direction is in the presence of feature edges, since they imply discontinuities in curvature. If feature edges can be detected robustly or indicated by the user, our algorithm can easily use this information to provide a better estimate of curvature values. In such a case, we simply stop extending the polyline upon intersection with a feature edge that is sufficiently removed from being parallel (easily measured by angle of intersection) to the plane. Figure 4 illustrates this concept. We can imagine generalizing this idea to other attributes defined continuously on the mesh's surface that penalize the growth of spider limbs.

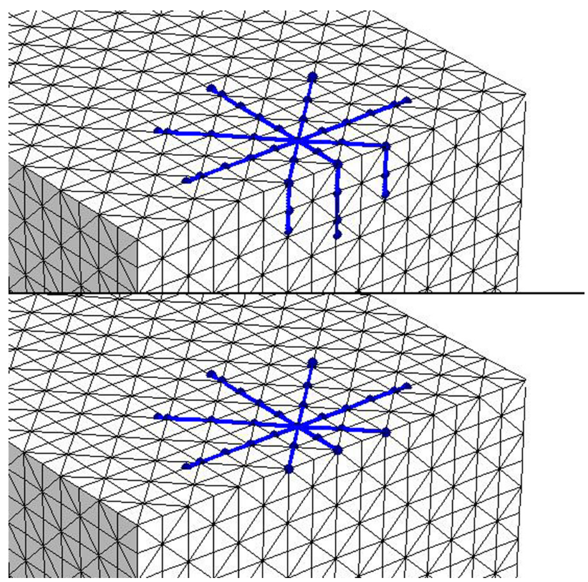

Figure 4: Top: isotropic spider ignores feature edges. Bottom: anisotropic spider aborts growth upon intersection with a feature edge.

2.6 Parameters

The proposed procedure requires a number of user specified parameters, including the geodesic radius, the number of directions sampled, the degree of the polynomial, the resampling increment size used on the obtained polylines and the maximum number of iterations if normals have not converged.

The size of the geodesic radius will have an impact on how smooth the results are. On noisier meshes a greater radius is recommended.

With respect to the number of directions sampled, there is a trade off between speed and accuracy. The more directions sampled the more accurate the estimate at the cost of speed. In our case we use 32 pairs of regularly spaced orthogonal directions for a total of 64.

In our implementation we use quadratic polynomials, given that they provide smooth results on noisy meshes as well as the fact that quadrics are the lowest degree polynomials that can locally approximate all combinations of positive/negative principal curvatures (hills, bowls and saddles.)

We resample the polylines at increments of $\frac{1}{10}r$, where r is the specified geodesic radius, but also making sure to include the origen as well as the polyline end points.

Regarding iterations, in cases where the surface normal is reliable (non noisy meshes) we perform only one iteration. In the noisy mesh examples we use ten iterations as maximum.

3 Results

In figure 5 we show the robustness of our approach to mesh noise. We generate a hyperbolic paraboloid and evaluate mean curvature using our operator. The dotted line shows the relative error obtained using the approach of Meyer *et al.* [Meyer et al. 2003] on the noisy mesh. The solid curve shows how the error of our approach decreases asymptotically as the geodesic radius increases (from 1 to 3 times the average edge length of the mesh.)

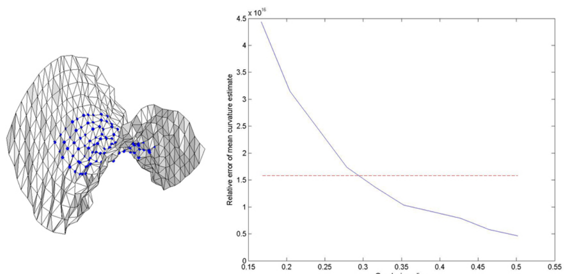

Figure 5: Left: saddle mesh with artificially added uniform noise in vertex normal direction with magnitude in $[-a/2, a/2]$ where a is the average edge length. Right: mean relative error of the curvature estimate on the indicated vertices decreases asymptotically as the geodesic radius of the operator increases. Dotted line represents the mean relative error of the mean curvature estimated using the one ring neighbourhood approach of Meyer *et al.*

In figure 6, we demonstrate our approach's robustness to less than ideal tessellations. To the left, we show the given surface witch contains irregular tessellations and face sizes as well as sliver faces. Next, is the mean curvature resulting from applying the approach of Meyer *et al.* [Meyer et al. 2003], followed by the results obtained from a Laplacian smoothing of the obtained curvature field. Finally, on the far right, we see the direct result of our approach. Notice that there are some irregularities in the coloration due to the fact that for rendering purposes we specify a color per vertex and face colors are interpolated.

Next, in figure 7 we compare the results of estimating mean curvature and surface normals using isotropic and anisotropic (feature sensitive) approaches. We tested both approaches on a noisy unit cube mesh. As can be seen in the result, the anisotropic version performs a better job at estimating curvature and normals near feature edges, as well as along them and their intersections.

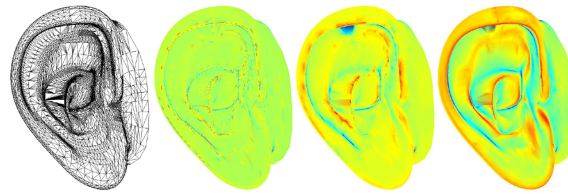

Figure 6: From left to right: 1) Mesh tessellation is non uniform, containing faces with broad range of sizes, often adjacent to each other, as well as sliver faces. 2) Mean curvature results using ring neighbourhood of Meyer *et. al* 3) Smoothed results using Laplacian smoothing of mean curvature values. (10 iterations, blending of .2) 4) Direct mean curvature results of our approach using a geodesic radius equal to the average edge length of the mesh.

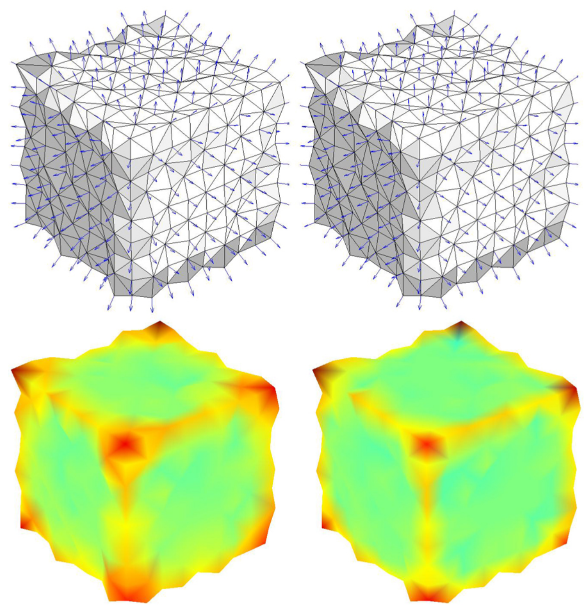

Figure 7: Unit cube mesh with artificially added uniform noise in vertex normal direction with magnitude in $[-a/4, a/4]$ where a is the average edge length. Top: estimated surface normals. Bottom: estimated mean curvature. Left: isotropic estimation using geodesic radius of .5. Right: anisotropic estimation using same geodesic radius.

Figures 8, 9, and 10 further illustrate our approach on dense meshes. In particular, figure 8 shows a dense mesh resulting from a noisy range scan and the resulting smooth mean curvature values estimated by the spider operator. Figure 9 compares the Gaussian curvature optained from an implementation of [Taubin 1995] and compares it to the results produced by applying our operator with increasing radius. Finally, figure 10 shows how the results of applying the spider operator can be used to extract features from mesh data.

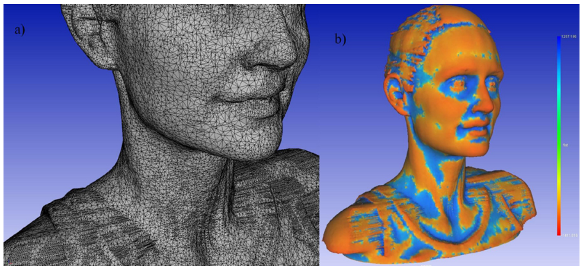

Figure 8: Spider on a noisy 3D scan. a) wireframe mesh b) Mean curvature plot using a 4x average edge length spider.

3.1 Mesh smoothing

Finally, we apply our surface sampling approach to mesh smoothing as an alternative to naive Laplacian smoothing in the presence of irregular tessellations. The idea is much the same, except instead of using a ring of adjacent vertices at each step we sample the neighboring surface within a specified geodesic radius, analogously to how we previously grew spider.

Here we implement a simple approach using a geodesic ring. To obtain the ring, we find the polylines of intersection just as before and simply retain their end points. We then move each vertex by a small amount in the direction of the ring's centroid and iterate the process.

This approach is illustrated in figure 11. We start with a mesh containing non regular tessellation and artificially add noise. We then compare the results of applying naive Laplacian smoothing with our approach. As can be seen, our approach preserves features present in the mesh such as in the ears and hooves, as well as the anisotropic nature of the tessellation. In the case of Laplacian smoothing the features are quickly lost.

4 Conclusions and future work

We have presented a new local shape parameterization operator, called a spider, aimed at the geometric processing of problematic, irregular and non-uniformly sampled meshes. In particular we show the spider to be useful in estimating the surface normal and curvature at any point on the mesh. We also show the spider to be useful in other mesh operations such as smoothing and feature curve extraction. A particular problem in geometric processing of meshes is one of scale and providing a robust and smooth normal and curvature map without blurring out features. For many applications such as contour rendering [DeCarlo et al. 2003] or feature extraction [Ohtake et al. 2004] the precise values of the surface derivatives are less important than the salience of the values, in that they are free of algorithmic degeneracies, are continuous while preserving discontinuous features such as sharp edges and in general allow for the accurate inference of high level descriptions of shape. The iterative anisotropic spider shows promise in this direction. Specifying the size and allowed variation in the length of spider limbs allows a user intuitive control over local sampling that is independent of the resolution or scale of the mesh.

In this paper we often tie the spider length to a function of average edge length over a neighborhood or the whole mesh. A spider that is able to adaptively increase its length in directions of low curvature and shrink its limbs in high curvature shows promise at

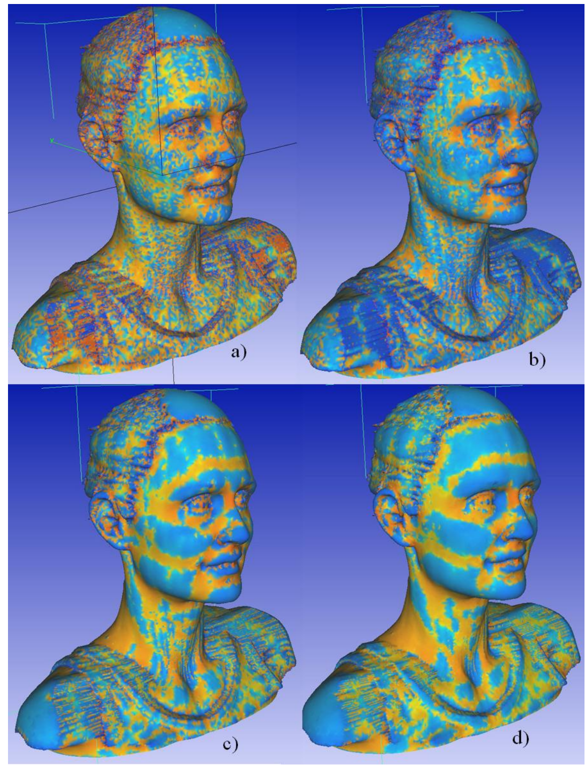

Figure 9: Gaussian curvature (blue positive, orange negative) on a noisy 3D scan. a) Taubin curvature tensor, b),c), and d) 1x, 4x, 8x average edge length spider.

automatically adjusting its length appropriately to different regions of a mesh and is subject of future work.

We show our approach to visually perform better in the case of poorly sampled mesh than ring neighbourhood approaches, while being efficient enough for interactive applications. While the ideas in this paper are presented in the context of dense meshes, mesh connectivity is only used as a data structure to facilitate the efficient computation of intersection curves. With the appropriate spatial hashing, the spider can be applied directly to dense point clouds or other discrete surface representations.

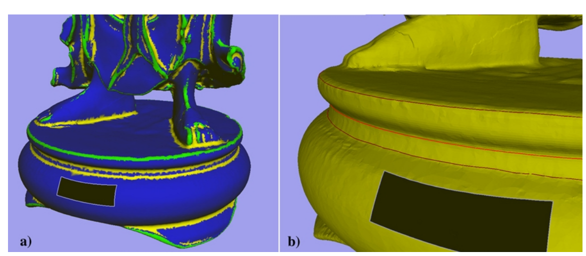

Figure 10: Feature curve extraction on base of Buddha statue using the spider. a) feature map computed by thresholding principle curvature values (green is highly convex, yellow is highly concave, blue is overall low curvature). b) Iso-curvature curves extracted from feature map.

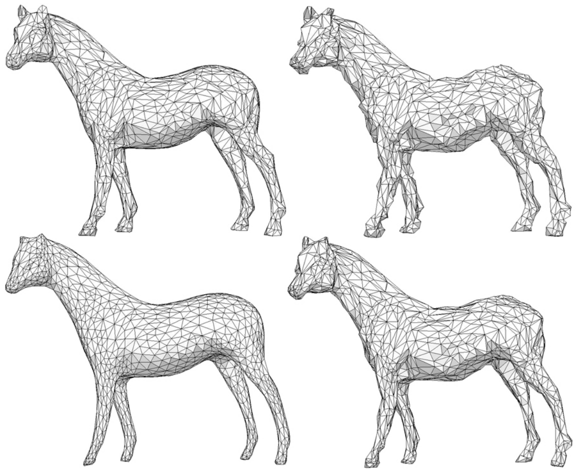

Figure 11: Top left: original mesh containing non regular tessellation. Top right: mesh with artificially added uniform noise in vertex normal direction with magnitude in $[-a/2, a/2]$ where a is the average edge length. Bottom left: Mesh smoothed using Laplacian smoothing. 10 iterations, blending coefficient of .2 Bottom right: Mesh smoothed using average of sampled geodesic ring of radius $a/2$ where a is the noisy mesh's average edge length. Iterations and blending coefficient are the same.

References

CAZALS, F., AND POUGET, M. 2003. Estimating differential quantities using polynomial fitting of osculating jets. *Proc. Symposium on Geometry Processing.*.

COHEN-STEINER, D., AND MORVAN, J.-M. 2003. Restricted delaunay triangulations and normal cycle. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, ACM Press, New York, NY, USA, 312–321.

DECARLO, D., FINKELSTEIN, A., RUSINKIEWICZ, S., AND SANTELLA, A. 2003. Suggestive contours for conveying shape. *ACM Trans. Graph. 22*, 3, 848–855.

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

GOLDFEATHER, J., AND INTERRANTE, V. 2004. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Transactions on Graphics 23*, 1.

HAMANN, B. 1993. Curvature approximation for triangulated surfaces. 139–153.

HECKBERT, P. S., AND GARLAND, M. 1999. Optimal triangulation and quadric-based surface simplification. *Comput. Geom. Theory Appl. 14*, 1-3, 49–65.

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 *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 131–144.

MAX, N. 1999. for computing vertex normals from facet normals. In *Journal of Graphics Tools*, vol. 4, 1–6.

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

MORETON, H. P., AND SEQUIN, C. H. 1992. Functional optimization for fair surface design. Tech. rep., Berkeley, CA, USA.

OHTAKE, Y., BELYAEV, A., AND SEIDEL, H.-P. 2004. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph. 23*, 3, 609–612.

PHONG, B. T. 1975. Illumination for computer generated pictures. *Commun. ACM 18*, 6, 311–317.

RUSINKIEWICZ, S. 2004. Estimating curvatures and their derivatives on triangle meshes. *3D Data Processing, Visualization, and Transmission, 2nd International Symposium on (3DPVT'04)*, 486–493.

TAUBIN, G. 1995. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, IEEE Computer Society, Washington, DC, USA, 902.

TAUBIN, G. 2001. Linear anisotropic mesh filters. *IBM Tech. Report RC-22213*.

THURMER, G., AND WUTHRICH, C. A. 1998. Computing vertex normals from polygonal facets. *J. Graph. Tools 3*, 1, 43–46.

ZHANG, R., AND FIUME, E. 2003. Butterworth filtering and implicit fairing of irregular meshes. *Proceedings of Pacific Graphics*, 502–506.

ZORIN, D., SCHRODER, P., AND SWELDENS, W. 1997. Interactive multiresolution mesh editing. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 259–268.