Computer-Aided Design and Applications
© 2008 CAD Solutions, LLC
http://www.cadanda.com

Curvature Computation for Triangular Meshes based on Local Parametrization

Fuhua (Frank) Cheng[a], Conglin Huang[a], Shuhua Lai[b], Fengtao Fan[a], Jiaxi Wang[a], Caimin Zhang[a]

[a]University of Kentucky, cheng@cs.uky.edu
[b]Virginia State University, slai@vsu.edu
[c]University of Shandong, China, czhang@sdu.edu.cn

ABSTRACT

A local parametrization based curvature computation technique for triangular meshes is presented. The computation process starts with interpolating the interested region of the given mesh with a Loop subdivision surface. The interpolation technique guarantees that the resulting surface reflects the local shape of the mesh, including features such as edges and corners; no data simplification is necessary. A blending technique is then applied to vicinities of extra-ordinary vertices to ensure continuity and boundedness of curvature at each extra-ordinary vertex. This blending process does not change the value of the surface at the extra-ordinary points. Curvatures for each given point are subsequently computed based on standard parametrization for Loop surfaces. Advantages of the new technique include that curvature can be computed in any direction for any point of the given mesh and higher accuracy of the computed results due to precise representation obtained by the interpolation process. Test results showing the effectiveness of the new technique are included.

Keywords: curvature, local parametrization, Loop subdivision surfaces, progressive interpolation
DOI: 10.3722/cadaps.2008.xxx-yyy

1. INTRODUCTION

Curvature is a fundamental property in computer graphics because it provides information on local shape of a surface. Curvature also facilitates many surface processing tasks, such as surface segmentation, surface reconstruction and registration. Actually, since it is invariant under viewpoint and parametrization transformation, surface curvature is of importance to almost all applications that deal with surface geometry. But due to limitation of current sampling techniques, such applications usually do not get smooth forms for the surfaces defining the model geometry. Instead they only have polyhedral approximations of the surfaces, such as triangular or quadrangular meshes. Hence, techniques are needed to estimate local surface curvatures using discrete data.

A large number of methods for computing surface curvatures on polygonal surfaces exist. These methods can be classified into several groups. Approaches such as [8] and [9] focus on finding good local approximation involving quadratic or cubic surfaces. Because of their approximation nature, these methods are sensitive to quantization noise. Approaches such as [10] and [11] employ discrete approximation formulas based on information of a point and its neighbors. These methods are often short in computation time, but not as accurate. Approaches such as [5] and [12] estimate the curvature by a discrete tensor. For each point on the surface, the tensor associates each surface tangent direction with the corresponding directional curvature.

Other approaches such as [13] employ a voting mechanism, an improvement version of the tensor based approaches. These methods gain in computation time at the cost of the attainable accuracy: the curvatures are smoothed out but some details are lost.

In this paper, a new method for estimating surface curvature based on local parametrization of a triangular mesh is presented. The idea is to interpolate the interested region of the given mesh with a C^2 -continuous, locally blended Loop subdivision surface [14] and use curvature of the locally blended Loop surface as an approximation to the curvature of the given mesh. The constructed surface faithfully reflects the shape of the interested region of the given mesh. Therefore, the computed curvature has a better approximation of the surface's curvature. No simplification of the data set is necessary. Most importantly, since the locally blended Loop surface has continuous curvature everywhere, one can compute curvature for any point of the given mesh in any direction.

The construction of the locally blended Loop surface requires two steps. First, a Loop subdivision surface that interpolate vertices of the interested region of the given mesh is constructed using a technique similar to the progressive interpolation technique for B-splines [4],[1],[3]. A blending technique similar to the one used for Catmull-Clark subdivision surfaces [6] is then applied to vicinities of the extra-ordinary points to ensure boundedness and continuity of curvature at the extraordinary points. Then each triangular face of the given mesh is parametrized using parametrization of the corresponding patch of the locally blended Loop surface.

The remaining part of the paper is arranged as follows. In Section 2 we present the over all idea of local parametrization based curvature computation. The process of using a progressive approach to construct a Loop subdivision surface to interpolate the interested region of a given triangular mesh is presented in Section 3; a proof showing that this process is well-defined in given in the Appendix. The process of blending vicinities of extra-ordinary points to ensure boundedness and continuity of surface curvature at those points is presented in Section 4. The curvature computation process is presented in Section 5. Implementation issues and test results are shown in Section 6. Concluding remarks are given in Section 7.

2. BASIC IDEA

First, we use a progressive interpolation technique [4],[1],[3] to find a Loop subdivision surface [14] that interpolates the interested region of the given triangular mesh. The idea is to view the given triangular mesh as the control mesh of a Loop subdivision surface, and iteratively upgrade locations of the vertices of this mesh until a new control mesh who limit surface interpolates the given mesh is obtained. The limit of the iterative interpolation process has the form of a global method. But the control points of the limit surface can be computed using a local approach. Therefore, the interpolation technique enjoys the advantages of both a local method and a global method. It does not require a data simplification process such as the one presented in [18] no matter how large the data set is.

With the technique for evaluating Loop subdivision surfaces at arbitrary parameter values being available [16], we can find the parametrization of each triangular surface of the control mesh. The computations of the first and second fundamental forms are easy since the parametrization of the surfaces are known.

However, a standard Loop subdivision surface is only C^1 -, not C^2 -continuous at extra-ordinary points. To avoid infinite curvature at the extra-ordinary points, the surface is then modified by applying a local blending technique similar to the one used for Catmull-Clark subdivision surfaces [6] to vicinities of extra-ordinary points of the given mesh. The blending process is done by blending the Loop surface with a low degree polynomial defined locally over a characteristic map of the extra-ordinary point [17]. The modified Loop surface is C^2 -continuous everywhere and, hence, has bounded and continuous curvature at the extra-ordinary points. This blending process does not change the value of the Loop surface at the extra-ordinary points. Therefore the modified Loop surface still interpolates all the vertices of the interested region of the given mesh.

Note that, with the parametrization known, we can compute not only the Gaussian and Mean curvatures at every point of the original mesh, but also curvatures in any direction at any point on the limit surface.

3. LOCAL INTERPOLATION

The process of constructing an interpolating Loop subdivision surface [14] based on progressive interpolation technique [4],[1],[3] is described below.

Given a 3D triangular mesh $M = M^0$. To interpolate the vertices of a region of M^0 with a Loop subdivision surface, one needs to find a control mesh $\overline{M}$ whose Loop surface passes through all the vertices of that region of M^0. Without loss of generality, we shall assume that region is M^0 itself. The job of finding the relationship between the vertices of $\overline{M}$ and the vertices of M^0 will be achieved through an iterative process, instead of a direct process.

First, we consider the Loop surface S^0 of M^0. For each vertex V^0 of M^0, we compute the distance between this vertex and its limit point V_∞^0 on S^0,

$$D^0 = V^0 - V_\infty^0,$$

and add this distance V^0 to get a new vertex called V^1 as follows:

$$V^1 = V^0 + D^0 .$$

The set of all the new vertices is called M^1. We then consider the Loop surface S^1 of M^1 and repeat the same process.

In general, if V^k is the new location of V^0 after k iterations of the above process and M^k is the set of all the new V^k's, then we consider the Loop surface S^k of M^k. We first compute the distance between V^0 and the limit point V_∞^k of V^k on S^k

$$D^k = V^0 - V_\infty^k \quad . \tag{1}$$

We then add this distance to V^k to get V^{k+1} as follows:

$$V^{k+1} = V^k + D^k . \tag{2}$$

The set of new vertices is called M^{k+1}.

This process generates a sequence of control meshes M^k and a sequence of corresponding Loop surfaces S^k. S^k converges to an interpolating surface of M^0 if the distance between S^k and M^0 converges to zero. Therefore the key task here is to prove that D^k converges to zero when k tends to infinity. This will be done in the Appendix.

Note that for each iteration in the above process, the main cost is the computation of the limit point V_∞^k of V^k on S^k. For a Loop surface, the limit point of a control vertex V with valence n can be calculated as follows:

$$V_\infty = \beta_n V + (1-\beta_n) Q \tag{3}$$

where

$$\beta_n = \frac{3}{11 - 8 \times (\frac{3}{8} + (\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n})^2)}$$

and

$$Q = \frac{1}{n}\sum_{i=1}^{n} Q_i$$

Q_i are adjacent vertices of V. This computation involves nearby vertices only. Hence the progressive interpolation process is a local method and, consequently, can handle meshes of any size.

Another point that should be pointed out is, even though this is an iterative process, one does not have to repeat each step strictly. By finding out when the distance between M^0 and S^k would be smaller than the given tolerance, one can go directly from M^0 to M^k, skipping the testing steps in between.

4. LOCAL BLENDING

With the construction of the Loop subdivision surface in the previous section, we can define a local (u,v) parameterization for each triangular face of the given mesh [16]. In case one of the vertices is an extra-ordinary point, then $(0,0)$ is set at the extraordinary point. The limit surface $S(u,v)$ is C^2-continuous for (u,v) different from $(0,0)$ and C^1-continuous at $(0,0)$. We assume each face has at most one extra-ordinary vertex here. If this is not the case, simply perform one subdivision on the constructed Loop surface first.

Then a low degree polynomial $P(u,v)$ is calculated to approximate the original Loop limit surface $S(u,v)$ near the extraordinary point. For simplicity, and also to avoid unexpected oscillations, quadratic polynomials are used here for $P(u,v)$. We calculate limit points on the subdivision surface after 3 levels of recursive subdivision. Knowing the values of the limit points and the (u,v) parameter values we can calculate the coefficients of $P(u,v)$ by a least-squares fit.

Using an approach similar to the one used for Catmull-Clark subdivision surfaces [6], the modified surface $\bar{S}$ is defined as:

$$\bar{S}(u,v) = S(u,v)W(u,v) + P(u,v)(1 - W(u,v))$$

where $0 < W(u,v) < 1$,is a C^2 continuous weight function that decays to zero at $(0,0)$, thereby canceling the irregularity of $S(u,v)$. Since the modified surface $\bar{S}$ coincides with P at the origin, $P(0,0) = S(0,0)$.

The following formula is chosen to define $W(u,v)$:

$$W(u,v) = \rho^2(3\rho^2 - 8\rho + 6), \quad \rho = \sqrt{u^2 + v^2}/\lambda, \quad \rho \le 1$$

λ is the sub-dominant eigenvalue of the subdivision matrix related to an extraordinary vertex of valence n.

At the extraordinary point itself, the modified surface $\bar{S}$ coincides with P up to its second-order derivatives, due to the interpolation requirement $S(0,0) = P(0,0)$. Thus, for evaluating $\bar{S}$ and its derivatives at the extraordinary point, we only need to evaluate the polynomial P at $(0,0)$.

5. CURVATURE COMPUTATION

After applying local blending on each extraordinary point, we have the parametrization for each face that gives C^2-continuity. Hence, we are able to compute the Gaussian and Mean curvature at all points on the original mesh: The first fundamental form:

$$E = \langle X_u, X_u \rangle,$$
$$F = \langle X_u, X_v \rangle,$$
$$G = \langle X_v, X_v \rangle.$$

The second fundamental form:

$$e = \langle N, X_{uu} \rangle,$$
$$f = \langle N, X_{uv} \rangle,$$
$$g = \langle N, X_{vv} \rangle.$$

Normal of a vertex can be computed either using the standard approximation technique of averaging the normals of adjacent faces, or by acquiring the cross product of X_u and X_v at that vertex. It turns out that, as expected, getting the cross product of two directional derivatives based on the parametrization is much more precise.

Based on the equations of Weingarten [2], we obtain Gaussian curvature K and Mean curvature H as follows:

$$K = (eg - f^2)/(EG - F^2),$$
$$H = (eg - 2fF + gE)/(2(EG - F^2)),$$

The principle curvatures k_1 and k_2 are the roots of the quadratic equation:

$$k^2 - 2Hk + K = 0\ ,$$

that is,

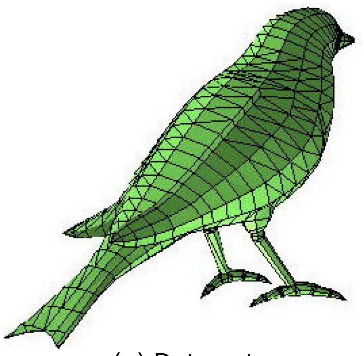
$$k = H \pm \sqrt{H^2 - K}\ .$$

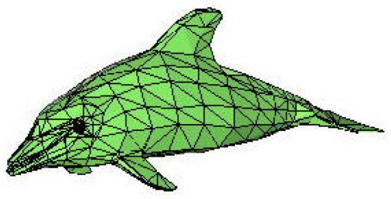
6. IMPLEMENTATION AND TEST RESULTS

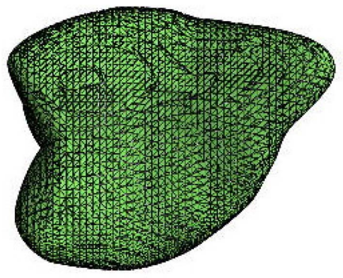
The local parametrization and curvature estimation process is implemented on Windows XP platform using OpenGL as the supporting graphics system. The algorithm is tested on meshes of various sizes. The number of faces of the meshes ranges from 400 to 18000. No simplification is needed for data set with more than 10,000 vertices because the new interpolation technique can handle meshes of any size. Some of the results are presented in Tab. 1 and Fig. 1. These results are compared with the discrete approach proposed by Cohen-Steiner and Morvan [5] which is more accurate than most discrete methods [7].

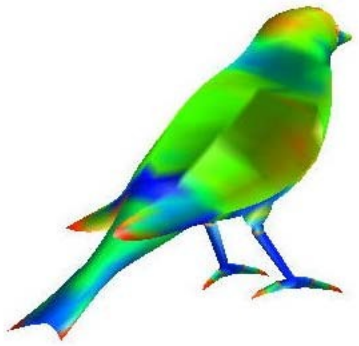
Model	*# of vertices*	*# of iterations*	*Max Error*
Bird	1129	4	0.0071073778
Dolphin	420	6	0.0065149639
Tooth1	9460	10	0.0063532368
Tooth2	16473	10	0.0068552462
Tooth3	5366	10	0.0065173326

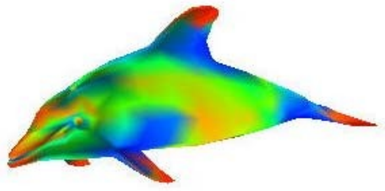
Tab. 1: Local parametrization based curvature estimation: test results.

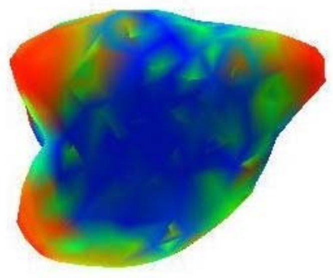

(a) Data set

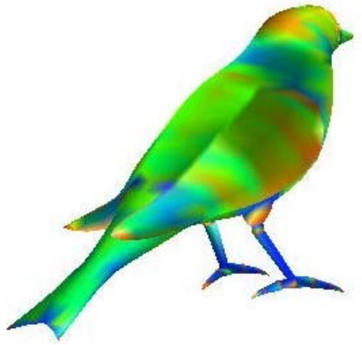

(b) Data set

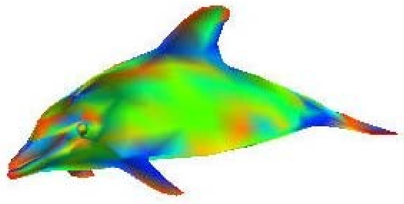

(c) Data set


(d) Curvature:discrete


(e) Curvature:discrete


(f) Curvature:discrete


(g) Curvature:parametrized


(h) Curvature:parametrized

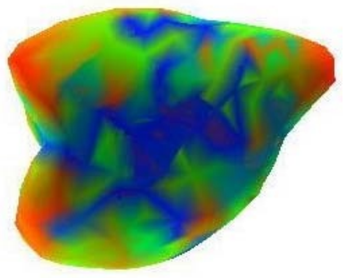

(i) Curvature:parametrized

Fig. 1: Examples of curvature estimation based on local parametrization.

7. CONCLUDING REMARKS

A local parametrization based curvature computation technique for triangular meshes is presented. This technique constructs a Loop subdivision surface that interpolates vertices of the interested region of the given mesh first. The constructed Loop surface is then modified in vicinities of extra-ordinary points to ensure boundedness and continuity of the curvature at the extra-ordinary points. No simplification of the data set is needed. Curvatures for each given point are subsequently computed based on standard parametrization for Loop surfaces. Advantages of the new technique include that curvature can be computed in any direction for any point of the given mesh and higher accuracy of the computed results due to precise representation obtained by the interpolation process. Test results show that indeed the new technique gives more details than classic, discrete structure based curvature estimation techniques.

ACKNOWLEDGEMENT

Research work presented here is supported by NSF (DMI-0422126) and KSTC (COMM-Fund-712). Triangular meshes used in this paper are downloaded from the Princeton Shape Benchmark [19].

APPENDIX A CONVERGENCE OF THE ITERATIVE INTERPOLATION PROCESS

The proof needs a fact about the eigenvalues of the product of positive definite matrices. This fact is presented in the following lemma.

Lemma 1 *Eigenvalues of the product of positive definite matrices are positive.*

The proof of Lemma 1 follows immediately from the fact that if P and Q are square matrices of the same dimension, then PQ and QP have the same eigenvalues (see, e.g., [15], p.14).

To prove the convergence of the iterative interpolation process for Loop subdivision surfaces, note that at the $(k+1)$ st step, the difference D^{k+1} can be written as:

$$\begin{aligned} D^{k+1} &= V^0 - V_\infty^{k+1} \\ &= V^0 - (\beta_n V^{k+1} + (1-\beta_n) Q^{k+1}) \end{aligned}$$

where Q^{k+1} is the average of the n adjacent vertices of V^{k+1}

$$Q^{k+1} = \frac{1}{n} \sum_{i=1}^{n} Q_i^{k+1}$$

By applying (2) to V^{k+1} and each Q_i^{k+1} ,

$$Q_i^{k+1} = Q_i^k + D_{Q_i^k}^k$$

we get

$$\begin{aligned} D^{k+1} &= V^0 - (\beta_n V^k + (1-\beta_n) Q^k) \\ &\quad - (\beta_n D^k + \frac{1-\beta_n}{n} \sum\nolimits_{i=1}^{n} D_{Q_i}^k) \\ &= D^k - (\beta_n D^k + \frac{1-\beta_n}{n} \sum\nolimits_{i=1}^{n} D_{Q_i}^k) \end{aligned} \tag{4}$$

where Q^k is the average of the \$n\$ adjacent vertices of V^k . In matrix form, we have

$$[D_1^{k+1}, D_2^{k+1}, \ldots, D_m^{k+1}]^T = (I-B) \begin{bmatrix} D_1^k \\ D_2^k \\ \vdots \\ D_m^k \end{bmatrix} = (I-B)^{k+1} \begin{bmatrix} D_1^0 \\ D_2^0 \\ \vdots \\ D_m^0 \end{bmatrix}$$

where m is the number of vertices in the given matrix, I is an identity matrix and B is a matrix of the following form:

$$\begin{pmatrix} \beta_{n_1} & \cdots & \frac{1-\beta_{n_1}}{n_1} & \cdots \\ \vdots & \ddots & & \\ \frac{1-\beta_{n_i}}{n_i} & \cdots & \beta_{n_i} & \cdots \\ \vdots & & & \beta_{n_m} \end{pmatrix}$$

The matrix B has the following properties:

1) $b_{ij} \geq 0$, and $\sum_{j=1}^{n} b_{ij} = 1$ (hence, $\|B\|_{\infty} = 1$);
2) There are $n_i + 1$ positive elements in the i -th row, and the positive elements in each row are equal except the element on the diagonal line;
3) If $b_{ij} = 0$, then $b_{ji} = 0$.

Properties 1) and 2) follow immediately from the formula of D^{k+1} in Eqn. (4). Property 3) is true because if a vertex V_i is an adjacent vertex to V_j then V_j is obviously an adjacent vertex to V_i . Due to these properties, we can write the matrix B as

$$B = DS$$

where D is a diagonal matrix

$$D = \begin{pmatrix} \frac{1-\beta_{n_1}}{n_1} & 0 & \cdots & 0 \\ 0 & \frac{1-\beta_{n_2}}{n_2} & \cdots & 0 \\ \vdots & & \ddots & \\ 0 & & & \frac{1-\beta_{n_m}}{n_m} \end{pmatrix}$$

and S is a symmetric matrix of the following form:

$$S = \begin{pmatrix} \frac{n_1\beta_{n_1}}{1-\beta_{n_1}} & \cdots & 1 & \cdots \\ \vdots & \ddots & & \\ 1 & \cdots & \frac{n_i\beta_{n_i}}{1-\beta_{n_i}} & \cdots \\ \vdots & & & \frac{n_m\beta_{n_m}}{1-\beta_{n_m}} \end{pmatrix}$$

D is obviously positive definite. We will show that the matrix S is also positive definite, a key point in the convergence proof.

Theorem 1 *The matrix S is positive definite.*

Proof: To prove S is positive definite, we have to show the quadric form

$$f(x_1, x_2, \ldots, x_m) = X^T SX$$

is positive for any non-zero $X = (x_1, x_2, \ldots, x_m)^T$.

Note that if vertices V_i and V_j are the endpoints of an edge e_{ij} in the mesh, then $s_{ij} = s_{ji} = 1$ in the matrix S . Hence, it is easy to see that

$$f(x_1, x_2, \ldots, x_m) = \sum_{e_{ij}} 2x_i x_j + \sum_{i=1}^{m} \frac{n_i \beta_{n_i}}{1-\beta_{n_i}} x_i^2$$

where e_{ij} in the first term ranges through all edges of the given mesh. On the other hand, if we use f_{ijr} to represent a face with vertices V_i , V_j and V_r in the mesh, then since an edge in a closed triangular mesh is shared by exactly two faces, the following relationship holds:

$$\sum_{f_{ijr}} (x_i + x_j + x_r)^2 = \sum_{e_{ij}} 4x_i x_j + \sum_{i=1}^{m} n_i x_i^2$$

where f_{ijr} on the left hand side ranges through all faces of the given mesh. The last term in the above equation follows from the fact that a vertex with valence n is shared by n faces of the mesh. Hence, $f(x_1, x_2, \ldots, x_m)$ can be expressed as

$$f(x_1, x_2, \ldots, x_m) = \sum_{f_{ijr}} \frac{1}{2}(x_i + x_j + x_r)^2 + \sum_{i=1}^{m} \left(\frac{n_i \beta_{n_i}}{1-\beta_{n_i}} - \frac{n_i}{2}\right) x_i^2$$

From Eqn. (3), it is easy to see that $\frac{n\beta_n}{1-\beta_n} \geq \frac{3}{5} n$ for $n \geq 3$. Hence, $f(x_1, x_2, \ldots, x_m)$ is positive for any none zero X and, consequently, S is positive definite.

Based on the above lemma and theorem, it is easy to conclude that the iterative interpolation process for Loop subdivision is convergent.

Theorem 2 *The iterative interpolation process for Loop subdivision surface is convergent.*

Proof: The iterative process is convergent if and only if absolute value of the eigenvalues of the matrix $P = I - B$ are all less than 1, or all eigenvalues λ_i , $1 \leq i \leq m$, of B are $0 < \lambda_i \leq 1$.

Since $\|B\|_\infty = 1$, we have $\lambda_i \leq 1$. On the other hand, since B is the product of two positive definite matrices D and S , following Lemma 1, all its eigenvalues must be positive. Hence, the iterative process is convergent.

REFERENCES

[1] de Boor, C.: How does Agee's method work? 1979 Army Numerical Analysis and Computers Conference,
ARO Report 79-3, Army Research Office, 299-302.

[2] Carmo, D. M.: Differential Geometry of Curves and Surfaces, Prentice Hall (February 1, 1976) ISBN-13: 978-0132125895.

[3] Lin, H.; Wang, G.; Dong, C.: Constructing Iterative Non-Uniform B-spline Curve and Surface to Fit Data Points, SCIENCE IN CHINA, Series F, 47, 2004, 315-331.

[4] Qi, D.; Tian, Z.; Zhang, Y.; Zheng, J. B.: The method of numeric polish in curve fitting, Acta Mathematica Sinica, 18, 1975, 173-184 .

[5] Cohen-Steiner, D.; Morvan, J.-M.: Proceedings of Restricted Delaunay Triangulations and Normal Cycle, 19th Annual ACM Symposium on Computational Geometry, 2003, 237-246.

[6] Levin, A.: Modified subdivision surfaces with continuous curvature SIGGRAPH, ACM Transactions on Graphics, 2006, 1035-1040.

[7] Magid, E.; Soldea, O.; Rivlin, E.: A comparison of Gaussian and mean curvature estimation methods on triangular meshes of range image data, Computer Vision and Image Understanding, 107(3), 2007, 139-159.

[8] Hamann, B.: Curvature approximation for triangulated surfaces, Computing Supplements, 8, 1993, 139-153.

[9] Krsek, P.; Lukacs, C.; Martin, R. R.: Algorithms for computing curvatures from range data The Mathematics of Surfaces VIII, Information Geometers, in: A. Ball et al. (Eds.), 1998, 1-16.

[10] Kim, S. J.; Kim, C.-H.; Levin, D.: Surface simplification using discrete curvature norm Third Israel-Korea Binational Conference on Geometric Modeling and Computer Graphics, Seoul, Korea, October, 2001.

[11] Meek, D. S.; Walton, D. J.: On surface normal and gaussian curvature approximations given data sampled from a smooth surface, Computer Aided Geometric Design, 17(6), 2000, 521-543.

[12] Taubin, G.: Estimating the tensor of curvature of a surface from a polyhedral approximation, Fifth International Conference on Computer Vision, 1995, 902-907.

[13] Page, D. L.; Sun, Y.; Paik, J.; Abidi, M. A.: Normal vector voting: Crease detection and curvature estimation on large, noisy meshes, Graphical Models, 64, 2002, 199-229.

[14] Loop, C. T.: Smooth subdivision surface based on triangle, Master's thesis, Department of Mathematics, University of Utah, 1987.

[15] Magnus, I. R.; Neudecker, H.: Matrix Differential Calculus with Applications in Statistics and Econometrics, New York, John Wiley and Sons, 1988.

[16] Stam, J.: Evaluation of Loop Subdivision Surfaces, SIGGRAPH Course Notes, 1999.

[17] Reif, U.: A unified approach to subdivision algorithms near extraordinary points, Computer Aided Geometric Design, 12(2), 1995, 153-174.

[18] Garland, M.; Heckber, P.: Surface simplification using quadric error metrics, Proceedings of SIGGRAPH, 1997:209-216.

[19] Shilane, P.; Min, P.; Kazhdan, M.; Funkhouser, T.: The Princeton Shape Benchmark, Shape Modeling Int'l, 2004.