COMPUTER-AIDED DESIGN & APPLICATIONS, 2017
VOL. 14, NO. 1, 58–69
http://dx.doi.org/10.1080/16864360.2016.1199756

Multivariate analysis of curvature estimators

Libor Váša[a], Tom Kühnert[b] and Guido Brunnett[b]

[a]NTIS – New Technologies for the Information Society, University of West Bohemia, Czech Republic; [b]Faculty of Informatics, Chemnitz University of Technology, Germany

ABSTRACT

Principal curvature is one of the defining features of surfaces studied in differential geometry. While well-defined and easy to evaluate for smooth surfaces, it cannot be evaluated exactly if the surface is represented by a polygon mesh, unless some special conditions apply. Nevertheless, estimating the curvature of a surface mesh is a crucial step in common mesh processing algorithms, such as mesh segmentation, mesh smoothing, remeshing and others. While a wealth of approaches for estimating the curvature has been proposed in the literature, aiming at the best possible precision of the estimation, an objective study identifying the strengths and weaknesses of the different methods was usually lacking. We present results of a comprehensive study focused on different aspects of curvature estimation. We extend some of the estimators in order to match properties of others and thus provide comparable results. The results of the study indicate that currently there is no one universally optimal method of curvature estimation. Choosing an appropriate curvature estimation algorithm is highly application specific, and we provide the guidance required for making such choice.

KEYWORDS

Curvature; estimation; mesh; multivariate; differential geometry; accuracy

1. Introduction

3D models in the form of polyhedral meshes are used in a multitude of applications and shape understanding for this approximation to a smooth model surface has received great attention in research. Estimating continuous properties, such as surface normals and curvatures, from the discrete representation is an important step for processing such models. Curvature in particular is omnipresent in algorithms including visualization (e.g. in non-photorealistic rendering), mesh smoothing, remeshing, mesh segmentation, perceptual mesh comparison, reverse engineering and many others.

Various methods have been presented to estimate curvature on polyhedral meshes and the mathematical background is well understood. Most publications on this problem develop a new estimation algorithm and show how it outperforms the existing ones. We suspect that this is sometimes assisted by selecting data where the estimator excels and not considering enough parameters which might vary with different meshes. As a result, the comprehensive understanding of the problem and the selection of the right curvature estimator is often hindered. This also stems from the fact that many factors influence the result, and thus a multivariate comparison exceeds the scope of most papers.

One example is the assumption on the local shape of the smooth surface when fitting a specific analytical surface for which the curvature properties are known. In this setting, the estimator with the correct assumption will most probably prevail in any evaluation. Furthermore, a comparison using synthetic meshes might not be representative for the selection of the right estimator for noisy data. A third example is the effective area in which the estimator uses the discrete mesh. While most researchers are aware of the assets and drawbacks of the smoothing effect which is produced when enlarging this area, we often find that curvature estimators with different ranges are compared to each other. For an example see Fig. 1, where a curvature-based mesh segmentation is heavily influenced by the assumption on the smoothness of the underlying shape.

Our goal is to assess the performance of popular curvature estimators based on a wide range of factors. We want to aid the reader in selecting the proper curvature estimator for a particular problem at hand. We modify existing approaches in order to make them comparable to others and applicable to other datasets. Despite those extensions, our aim is not to present another novel approach which performs best in all situations. The focus of the evaluation is on the accuracy of the estimated

CONTACT Libor Váša lvasa@kiv.zcu.cz

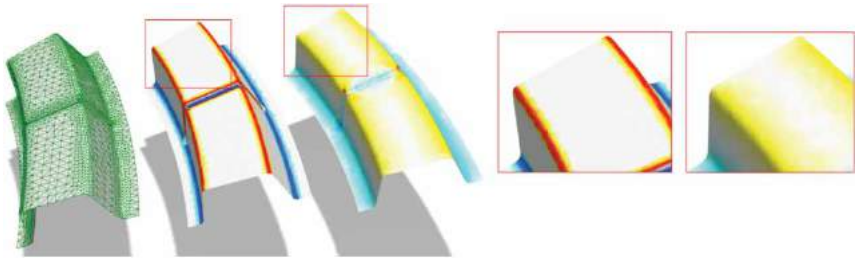

Figure 1. A polyhedral mesh for a metal forming process exported from the *CATIA V5* software. The color-coding shows the results of two different curvature estimators. Choosing the better suited algorithm depends heavily on the assumptions of the approaches, for example which region of the mesh is assumed to be smooth. This also shows that none of the estimators assuming a smooth surface can produce correct values for the vertices where the underlying surface is not continuous in curvature. A mesh segmentation for example must be aware of this limitation.

curvatures which are compared against the exact values of various analytic surfaces.

Section 2 introduces the mathematical context, especially including the fundamental forms. Building on this, we aim to present popular approaches to curvature estimation in a coherent language. Section 3 describes major parameters of the accuracy evaluation and Section 4 presents the modifications required for comparing the different estimators. The analysis in Section 5 constitutes the main part of the paper where the popular approaches and their extensions are examined for an impartial opinion. The results are summarized in the last section.

2. Background

2.1. Mathematical background and notations

Let a smooth surface S be locally described at point $\mathbf{p}$ as a function $X(u,v)\colon R^2 \to R^3$ using a local parametrization of S. This provides a tangential coordinate system with basis X_u (short for partial derivative of $X(u,v)$ at $\mathbf{p}$ in direction u), X_v, and a unit normal N (orthogonal to X_u and X_v). The *first fundamental form* describes the inner product of two parameter space vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ as the inner product of their corresponding tangential vectors in R^3, $\mathbf{s}_1 = (X_u, X_v)\mathbf{t}_1$ and $\mathbf{s}_2 = (X_u, X_v)\mathbf{t}_2$ (see Fig. 2). This function $I\colon R^2 \to R$ is often written as symmetric 2×2 tensor I with elements E, F and G. The *second fundamental form* describes how the surface normal N changes in the vicinity of $\mathbf{p}$. It is either expressed using the *shape operator*, which for a given tangential direction produces a directional derivative of N, or through the combination of the second order derivatives of X, which are then projected onto the normal. The tensor II, with elements e, f and g, is connected to I through the *Weingarten matrix*, a 2×2 tensor $W = I^{-1} \cdot II$. The *principal curvatures* and *principal directions* are now available as eigenvalues and eigenvectors of W. Note that these equations get much simpler when the vectors X_u and X_v are orthonormal. In this case, the terms *second fundamental tensor*, *Weingarten matrix* and *shape operator* are often used interchangeably.

$$I(\mathbf{t}_1,\mathbf{t}_2) = \mathbf{s}_1^T\mathbf{s}_2 = \mathbf{t}_1^T \begin{pmatrix} E & F \\ F & G \end{pmatrix} \mathbf{t}_2 = \mathbf{t}_1^T \begin{pmatrix} X_u^T X_u & X_u^T X_v \\ X_v^T X_u & X_v^T X_v \end{pmatrix} \mathbf{t}_2 \tag{2.1}$$

$$II(\mathbf{t}_1,\mathbf{t}_2) = Shape(\mathbf{s}_1)^T\mathbf{s}_2 = \mathbf{t}_1^T \begin{pmatrix} e & f \\ g & g \end{pmatrix} \mathbf{t}_2 = \mathbf{t}_1^T \begin{pmatrix} X_{uu}^T N & X_{uv}^T N \\ X_{vu}^T N & X_{vv}^T N \end{pmatrix} \mathbf{t}_2 \tag{2.2}$$

$$W = \frac{1}{EG - FF} \begin{pmatrix} eG - fF & fG - gF \\ fE - eF & gE - fF \end{pmatrix} \tag{2.3}$$

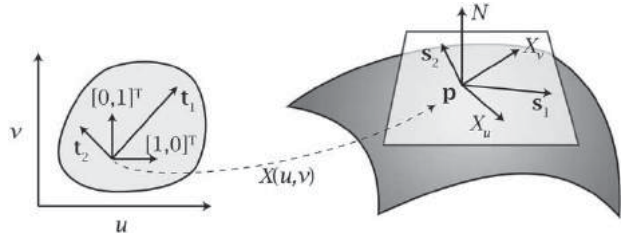

Figure 2. Parametric surface and tangent plane.

Those properties of S are now to be estimated from its discretization as polygonal mesh. In the following, we will use $\mathbf{p}$ to denote not only a point on the smooth surface, but also a vertex of the mesh. Neighbors to $\mathbf{p}$ are vertices adjacent in the mesh, written as $\mathbf{q}$ or $\mathbf{q}_i$ for any or a specific neighbor. N is in a similar fashion reused as vertex-normal at $\mathbf{p}$. Please note that the computation of N is also an estimation and Section 3 discusses how N can be obtained.

2.2. Estimation approaches

There are several general groups of approaches to curvature estimation, based on different concepts. A first group of estimators fits an analytic surface or a curve into the vertices of a local region. Goldfeather and Interrante [5] describe three approaches which are widely used. For all of them, a local orthonormal coordinate system $L = \{U, V, N\}$ is constructed at a vertex $\mathbf{p}$ using the tangent plane orthogonal to the vertex normal N. This local frame allows retrieving the coefficients of the Weingarten Matrix W, from which the principal curvatures and directions are available.

The *Normal Curvature Approximation* estimates W by its property to yield the normal curvature for a given direction: $k_t = \mathbf{t}^T W \mathbf{t}$. For any vertex $\mathbf{q}$ adjacent to $\mathbf{p}$, this curvature can at the same time be approximated as the inverse radius of the osculating circle passing through $\mathbf{p}$ (with normal N) and $\mathbf{q}$. The corresponding tangential direction $\mathbf{t}$ is then the edge between $\mathbf{p}$ and $\mathbf{q}$, projected to the tangent plane and normalized. For each of the n (non-degenerate) neighbors we obtain Eqn. (2.4) and we can solve this linear system for the unknown coefficients of W via least squares. Due to the orthonormal basis, only e, f and g have to be found. Similar approaches using the normal curvature have been used by [12] and others.

$$k_n = \mathbf{t}^T W \mathbf{t} = 2\frac{(\mathbf{p}-\mathbf{q})^T N}{(\mathbf{p}-\mathbf{q})^T(\mathbf{p}-\mathbf{q})}$$
$$\mathbf{t} = \begin{pmatrix} U^T \\ V^T \end{pmatrix} (\mathbf{p}-\mathbf{q}) \tag{2.4}$$

The *Quadratic Surface Approximation* fits the surface $f(u,v) = (A/2)u^2 + Buv + (C/2)v^2$ into $\mathbf{p}$ and its adjacent vertices. From Eqns. (2.1) to (2.3), one can see that A, B and C are the three unknowns of the now symmetric matrix W for this surface. By transforming all neighbors $\mathbf{q}$ into L, a linear system of n equations can be solved for W. Goldfeather and Interrante show that scaling each equation by factor $2/l^2$ (where l is the respective length of $(\mathbf{p}-\mathbf{q})$ in the tangent plane) makes this approximation approach similar to the Normal Curvature Approximation (see also Fig. 3). The only difference is that parabolas are used instead of circles.

This approach is further expanded to the *Adjacent-Normal Cubic Approximation* by also using the vertex normals of the neighbors $\mathbf{q}$. The surface function is expanded to $f(u,v) = (A/2)u^2 + Buv + (C/2)v^2 + Du^3 + Eu^2v + Fuv^2 + Gv^3$, for which W is still the same. The normal of this function at any point (u,v) can be computed from the cross product of the partial derivatives of f and thus be integrated into the linear system for fitting the surface. Although there are now seven unknown variables (A through G), the curvature depends only on the first three. A detailed insight in fitting truncated Taylor expansions to a smooth surface is provided by Cazals and Pouget[2]. They already show that the estimation quality heavily depends on the type of surface which is being used for the fit.

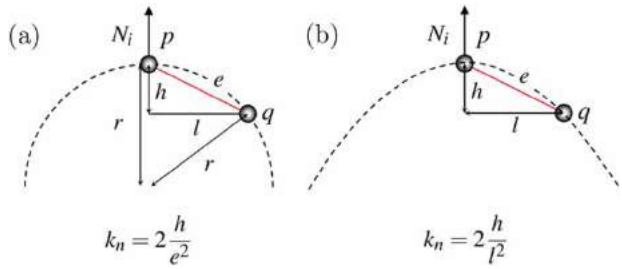

Figure 3. Geometric interpretation from fitting circles and parabolas (dotted) to an edge (red). a) The normal curvature can be approximated using the radius of the osculating circle. For this Eqn. (2.4) is established from the lengths of the edges of the two indicated triangles. Note how the direction of the edge defines the the sign of the curvature.) Quadratic Surface Approximation uses the same idea by fitting a parabola to each edge [5].

Meyer et al. suggest in [10] using a least squares fit only for estimating the principal directions and propose to compute the principal curvatures from mean and Gaussian curvature (see Eqn. (2.5)). The discrete mean curvature H and the discrete Gaussian curvature K are approximated over local integrals using a specific area A on the triangulated mesh. The angles α_i and β_i in Eqn. (2.6) are the angles opposing the edge $(\mathbf{p}, \mathbf{q}_i)$ and there are n adjacent edges for $\mathbf{p}$. θ_j is the angle in each of the k triangles adjacent to $\mathbf{p}$ in the corner corresponding to $\mathbf{p}$.

$$k_{1,2} = H \pm \sqrt{H^2 - K} \tag{2.5}$$
$$H = \frac{1}{2A_{mixed}} \sum_{i=1}^{n} (\cot\alpha_i + cot\beta_i)(\mathbf{p}-\mathbf{q}_i)$$
$$K = \frac{1}{2A_{mixed}} \left(2\pi - \sum_{j=1}^{k} \theta_j\right) \tag{2.6}$$

Cohen-Steiner and Morvan[3] propose to use a tensor which essentially represents the shape operator defined by the dihedral of two triangles, a region around the edge (given by edge length) and the direction of the edge. This tensor is averaged in a certain area (such as the geodesic disc around a vertex) and scaled with area size. The principal curvatures and directions are retrieved as the two maximum eigenvalues of the three-dimensional tensor.

Rusinkiewicz[11] starts by estimating the second fundamental tensors per triangle. An orthonormal parametrization in the plane of the triangle is constructed

from the triangle normal and one of its edges. When multiplying II with a parameter space vector $\mathbf{t} = (t(u), t(v))$, we directly obtain the directional derivative of normal N with respect to $\mathbf{t}$ (see Eqn. (2.7)). II is computed using the three equations provided by the three edges in each triangle and the normal differences along those edges. Subsequently, the per-vertex tensor is computed as a weighted average of the per-face tensors, which must be rotated from the coordinate system of each triangle into the coordinate systems of the vertex. Due to the chosen parametrization, the curvature is now available through the eigenvalues and eigenvectors of the (per-vertex) tensor II. A similar method is discussed by Theisel[13].

$$II \begin{pmatrix} t(u) \\ t(v) \end{pmatrix} = \begin{pmatrix} X_u^T N_u t(u) + X_u^T N_v t(v) \\ X_v^T N_u t(u) + X_v^T N_v t(v) \end{pmatrix} = \begin{pmatrix} X_u^T N_t \\ X_v^T N_t \end{pmatrix} \tag{2.7}$$

Other methods and variations of the approaches above exist[7][8][6], but to the best of our knowledge the six above described approaches are used most widely. All of those works are aware of the discrete representation of the smooth surface and many already discuss important factors, such as the size of the patch used for the estimation or the local regularity of the mesh. At the same time, the comparison to other methods often disregards additional factors. For example, both Rusinkiewicz and Goldfeather compare their estimators, which effectively use the 2-ring neighborhood, against methods which rely only on the 1-ring.

2.3. Contributions

In this work, we illustrate, explain and evaluate the factors which should influence the choice of a proper curvature estimation method. Our contribution is threefold:

- Factors influencing the choice of the estimator are illustrated and practical use cases highlighted (Section 3).
- Existing estimators are modified so that they can be compared to others, specifically in terms of estimation range (Section 4).
- An evaluation protocol is designed and results are presented to assess the accuracy of different estimators when varying important factors (Section 5).

3. Factors

Curvature estimation on a polygonal mesh is obviously limited by the quality of the data. Information which has been lost due to smoothing, simplification or discretization cannot be recovered without additional knowledge, such as the parameters for the subdivision surface. Furthermore, the performance of an estimator is influenced by how well the assumptions of the estimation approach apply to the data at hand. In the following, we discuss data properties and algorithmic characteristics which should be considered when comparing and choosing curvature estimators. We distinguish between properties which are inherent to the data and properties which are inherent to the estimation approach.

3.1. Data connectivity

The first three factors which influence the choice of the curvature estimator originate from the surface representation. While we discuss only polygonal meshes, those can still have various forms and specifications. **Non-triangle meshes** are one representation on which not all of the presented estimators can operate. This includes especially quadrilateral meshes and mixed quad/triangle meshes which both are widely used in modeling. A simple triangulation of such polygons should not be used because different triangulations generate different curvature values, especially if the polygons are not planar.

Borders or features in a polygonal mesh may in some cases hinder the curvature estimation. Such topological edges exist in meshes with holes or regions which are semantically separated (for example through a segmentation process). The neighborhood of a vertex near to or on such an edge might not support the assumptions of some estimators.

Regularity is another factor that is considered in most research on curvature estimation (see for example [5]). For triangle meshes, a regular triangulation is often defined by a vertex neighborhood which has a valence of six (i.e. each vertex has six adjacent vertices) and an equal spacing between most of the neighbors. Irregularity is introduced by changing the vertex positions on the surface and either keeping the mesh connectivity[5] or changing it[11].

3.2. Data noise

Depending on the origin of the polygonal mesh, different kinds of noise appear. While a mesh exported from a CAD modeling tool will typically be noise free, 3D scanning of a real-world surface will probably produce a Gaussian noise, and mesh compression might introduce a uniform quantization noise. This affects the sampling of the surface and consequently also the curvature estimation.

3.3. Data discretization

One of the most important factors influencing the quality of the mesh is the sampling rate (sampling density). Practical considerations, such as storage space, processing times, scanning resolution or numerical issues, limit the density of most meshes and simultaneously the accuracy of the representation of the original surface. This accuracy can be reduced to a point where the curvature cannot possibly be estimated correctly.

If the curvature estimator takes into account a large neighborhood around a vertex (for example to fit an analytical surface), the discretization must provide sufficient smoothness in this region. This problem is thus closely related to the assumptions of the estimator. Triangulation of an analytical surface is a typical example where this is important. Most triangulation approaches allow to specify a maximum distance between the original and the triangulated surface, below which the sampling rate is globally increased. It is thus necessary to analyze how the precision of the curvature estimation varies with changing sampling rates when data noise and sampling irregularity are present.

3.4. Estimation range

All estimators obtain the curvature properties of the mesh by examining a local neighborhood. We define the range of an estimator as the distance in which a change in the mesh has an affect on the estimation result. This support area for an estimation can be the 1-ring neighborhood ($R1$) of directly adjacent neighbors to a vertex. If the per-vertex normals of the 1-ring neighbors are required, the estimator effectively utilizes the 2-ring neighborhood ($R2$). We will discuss which approaches allow to specify this range as a parameter and how a larger range influences accuracy.

While a larger estimation range will increase robustness against noise, it smooths shape information at the same time (see Fig. 4). As a result, subsequent processing steps will be heavily influenced by the range of the estimator. For example, a segmentation process on the mesh in Fig. 1 profits from a smaller range of the estimator $R1$ when the mesh does not contain noise. With a high range $R3$, highly curved areas cannot be captured. This example also shows that none of the estimators presented can produce a correct curvature value if the surface is not smooth, i.e. the curvature is not continuous. The yellow colored vertices in the middle picture have a value in between the high curvature of the bend (red) and the zero curvature or the planar region (white). Smoothing a mesh or its curvature values also has an effect similar to increasing estimation range. See Fig. 5 for an example with a noisy data set.

Estimation range is one major motivation for this research, since we feel that few publications consider this factor adequately. In most comparisons, we find estimators of different range compared to one another. It has not yet been analyzed how the popular approaches to curvature estimation perform on a similar, more comparable range.

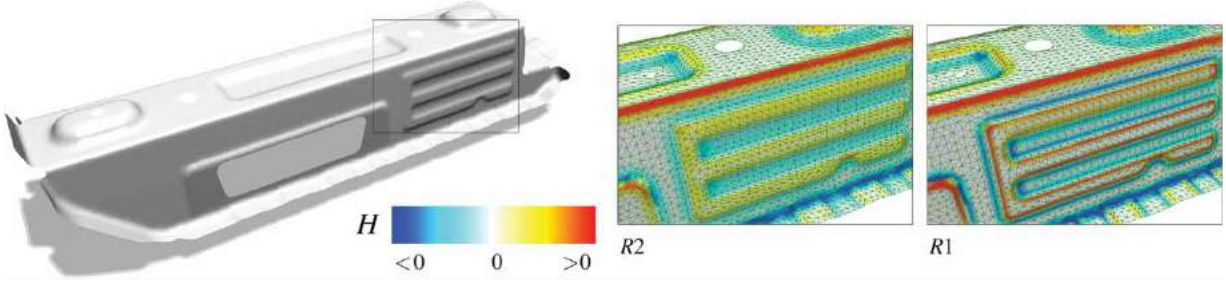

Figure 4. The selection of the curvature estimator and its properties influences the interpretation of the polygonal mesh. In such a coarse mesh, the 2-ring estimator **RU** ($R2$) cannot separate high curvature and planar regions as well as the 1-ring **QS** ($R1$). However, this alone is not significant if the sampling density is increased or noise is present. Please note also that the mesh has a border and that not all estimators will produce a good value for border vertices in curved regions. The color coding shows the mean curvature as indicated.

Figure 5. Example of a scanned model with about 200k triangles. Curvature is estimated using (left to right) **QS** ($R1$), **MRU** ($R2$) and **CSM** ($R3$). Note the impact of the estimation range on the interpretation of noise and features.

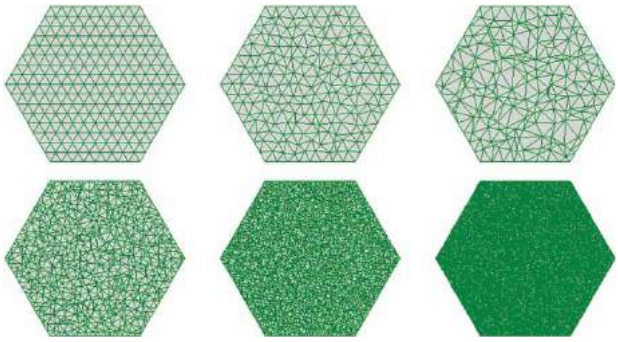

Figure 6. Planar versions of meshes used for the evaluation. Top row: increasing irregularity (0°, 14° and 25° RMS error) Bottom row: increasing sampling rate (817, 3169 and 12481 vertices).

3.5. Estimator assumptions

Besides the size of the region in which the original surface is expected to be smooth, some curvature estimation approaches assume a certain type of surface, for example when fitting an analytical surface to the discrete values. This does not imply that they fail with different surfaces, but the accuracy has been shown to be optimal for the correct surface type[2]. It has to be shown how estimators for a specific surface perform against other estimators and we want to point out that it is not predicative to use a single class of surface for the comparison (as done in [8] on spheres).

The per-vertex normals which are used in some estimators can be seen as another assumption of the approach. One must consider that the computation of those is also an estimation which is limited by similar factors. Goldfeather and Interrante[5] elaborate on how different normal estimations influence the accuracy of their method.

3.6. Other estimator properties

Three more factors should be discussed for the different estimators from a practical point of view. *Implementation complexity* describes how easily a method can be implemented, *Run time complexity* allows comparing the approaches in terms of computational requirements and *Locality* assesses how well an estimator can be used to compute curvature only locally, allowing parallelization.

4. Estimator modification

In order to compare the accuracy of the methods, we need to ensure that they operate on the same data and that the results can be interpreted in a comparable manner. As discussed earlier, different ranges imply different interpretations of the data. This section shows how most estimation approaches can be modified so that they become comparable.

4.1. Range extension

Among the estimators presented in Section 2, the method from Cohen-Steiner and Morvan already has an explicit notion of the range in which the polygonal mesh is examined. We relate the approximated radius of the geodesic disc to the maximum distance found in the 1-ring.

The methods by Goldfeather and Interrante can be extended to a different region in a straightforward manner. We use a simple breadth-first search to collect all vertices in a given range and add them to the system of equations. The range R to which neighbors $\mathbf{q}_j$ of a vertex $\mathbf{p}$ are collected is given as a multiple of the maximum distance to $\mathbf{p}$ in its 1-ring neighborhood. We chose the maximum distance because any smaller distance would downgrade the performance of the original estimator (for a factor of 1). This can easily be seen if one imagines a vertex in 2D and two outgoing edges to two other 2D neighbors. If curvature is approximated by an osculating circle, we (incorrectly) increase the curvature when regarding only a smaller region around the vertex (and hence shorter edges). Note also that we do not use discrete rings, but all neighboring vertices in the given range. This allows comparing those estimators to the method of Cohen-Steiner and Morvan. The price for this approximation is that the accuracy increases when the triangulation is irregular and longer edges are present in the 1-ring.

The estimators of Meyer[10] and Rusinkiewicz[11] cannot be extended in such a manner. While the former will hence be compared using the original 1-ring formulation, we present an adaption to the idea of Rusinkiewicz which allows a larger range than the original approach.

4.2. Modified Rusinkiewicz method

Rusinkiewicz[11] shows how the second fundamental tensor can be computed for each triangle and carefully averaged to a per-vertex tensor (see Section 2). For computing this tensor, the property of the shape operator is used that it gives the directional derivative of the normal. We now utilize the same observation to compute the per-vertex curvature.

For each vertex $\mathbf{p}$, we first construct an orthonormal coordinate system with the vertex normal N as normal of a tangent plane. As Rusinkiewicz, we use the normals estimated by the method of Max[9]. The approach also works with normals computed differently and improves when better (or exact) normals are available. Vectors X_u and X_v in this plane are obtained using

the (non-degenerate) edge to any adjacent vertex **q**. For each edge of the 1-ring, we assume that, according to Eqn. (2.7), the vector $\mathbf{t} = ((X_u, X_v)^T(\mathbf{p} - \mathbf{q}))$ multiplied with the fundamental tensor II equals the directional derivative of N in direction **t** in parameter space. As in the original Rusinkiewicz approach, we approximate this derivative using finite differences between the per-vertex normals N and N_q of **p** and **q**. This gives for each edge the equations (4.1) and (4.2) from which the second fundamental tensor and the principal curvatures and directions can be computed using least-squares.

$$\begin{pmatrix} e & f \\ f & g \end{pmatrix} \begin{pmatrix} X_u^T(p-q) \\ X_v^T(p-q) \end{pmatrix} = \begin{pmatrix} X_u^T(N_q - N \\ X_v^T(N_q - N) \end{pmatrix} \tag{4.1}$$

$$\begin{pmatrix} X_u^T(p-q) & X_v^T(p-q) & 0 \end{pmatrix} \begin{pmatrix} e \\ f \\ g \end{pmatrix} = X_u^T(N_q - N)$$

$$\begin{pmatrix} 0 & X_u^T(p-q) & X_v^T(p-q) \end{pmatrix} \begin{pmatrix} e \\ f \\ g \end{pmatrix} = X_v^T(N_q - N) \tag{4.2}$$

Our modification has several advantages over the original approach for our study. First, we *can now use the estimation range as a parameter* as proposed for the approaches of Goldfeather and Interrante by simply adding more distant vertices to the system of equations. This allows to extend the range of the estimator to more than the 2-ring on which the original Rusinkiewicz method effectively operates. While the original method requires a triangle mesh, our modification can also be applied to different *connectivities*, such as quad-meshes or even point clouds. The estimator has a *better locality* since the per-triangle step is no longer required, making parallelization trivial. Since the per-vertex averaging with the careful rotation of the tensor is omitted, the algorithm is *easy to implement* and *faster than the original* method.

5. Comparison

We present a study which compares the accuracy of the estimators from Section 2 (including extensions from Section 4), while changing the parameters presented in Section 3. The estimation ranges *Rk* are the variants of the methods using *k*-times the maximum 1-ring distance as local support. In the rest of the paper, we refer to the methods using following abbreviations:

- **My**: Meyer [10] (*R*1)
- **CSM**: Cohen-Steiner Morvan [3] (*R*1 to *R*4)
- **NC**: Normal Curvature Approximation [5] (*R*1 to *R*4)
- **QS**: Quadratic Surface Approximation [5] (*R*1 to *R*4)
- **AN**: Adjacent-Normal Cubic Appr. [5] (*R*2 to *R*4)
- **RU**: Original Rusinkiewicz [11] (*R*2)
- **MRU**: Modified Rusinkiewicz: Section 4(*R*2 to *R*4)

5.1. Approach

We evaluate the estimation accuracy using analytic functions for which the curvatures are known. Such function can be represented by a triangle mesh on which the curvature is estimated and compared against the ground truth. Since some estimators are expected to perform better only on surfaces of certain type, we use three different functions (see Eqn. (5.1)). The functions were chosen so that they include similar types of surfaces as in the evaluations in [11] and [5].

$$\begin{aligned} f_{quadratic}(x,y) &= Ax^2 + Bx + Cxy + Dy + Ey^2 \\ f_{ellipsoid}(x,y) &= C\sqrt{1 - A^2x^2 - B^2y^2} \\ f_{trigonometric}(x,y) &= \sin x^A + \cos By \end{aligned} \tag{5.1}$$

The meshes that represent those functions are based on a hexagonal shape formed by six equilateral triangles of edge length 1. We randomly selected the parameters A to E in such a range that comparable curvature values are obtained (randomly in range for the quadratic and trigonometric case $[-2, +2]$, except the exponential A, which is selected from $[+1, +2]$. The ellipsoid parameters are in interval $[-10, +10]$, where A and B are scaled so that $A^2 + B^2 \leq 1$).

The polygonal mesh must be created with the remaining data influencing factors, i.e. the discretization, topology and noise, in mind. We select four subdivisions of the hexagon shaped mesh with 217, 817, 3169 and 12481 vertices. In the plane, those meshes are regular, each non-border vertex has a valence of six and all angles and edge lengths are constant. We can now increase the irregularity by randomly shifting non-border vertices (in the plane) and changing connectivity to restore a Delaunay triangulation when necessary, since most meshing algorithms produce such a triangulation. We measure the amount of irregularity as root mean square difference of all angles from the expected 60° (see Fig. 6). Subsequently the z coordinates are computed using Eqn. (5.1).

Since not all estimators can handle mesh borders correctly, we exclude all vertices which have a border vertex in their *R*4-range from further computations. Finally, data noise is introduced by perturbing all vertices (in 3D) randomly. The noise magnitude has a Gaussian distribution with standard deviation derived from the mean edge length of the whole mesh, so that the noise is comparable for different mesh resolutions. Fig. 7 shows some example meshes.

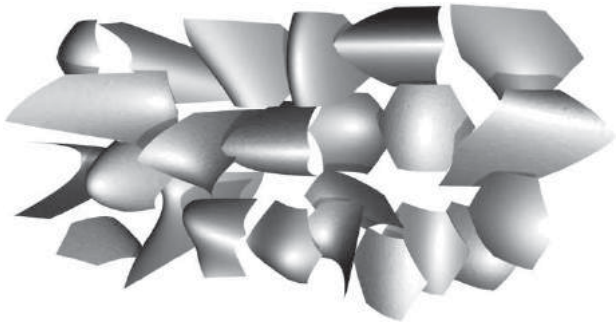

Figure 7. A visualization of 24 meshes (all with 3169 vertices) used for the evaluation, including noisy and exact surfaces.

Our evaluation thus considers the factors *sampling density, noise, mesh irregularity, estimator assumption on the type of function* and the *estimation range*. We evaluate the error e_{abs} between the exact principal curvature values k_1 and k_2 and the approximated values $\hat{k}_1$ and $\hat{k}_2$ as $e_{abs} = |k_1 - \hat{k}_1| + |k_2 - \hat{k}_2|$. Our tests have shown that this error depends almost linearly on $|k_1| + |k_2|$. This allows us to present the relative error measure e_{rel} from Eqn. (5.2) which is, in contrast to e_{abs}, comparable between different vertices. From this, we can compute the quadratic mean (or RMS) over all vertices for each permutation of the parameters (density, noise, (ir)regularity, function, estimator range). For each setting we use an average over 1000 random surfaces. With four different densities, four standard deviations of noise, three levels of irregularity, three functions and 20 estimators, this requires 2,880,000 curvature estimations.

$$e_{rel} = \frac{|k_1 - \hat{k}_1| + |k_2 - \hat{k}_2|}{|k_1| + |k_2|} \tag{5.2}$$

There are three further considerations that are of interest. First, we do not use the exact normals, but compute them by the method of Max[9]. While we would like to separate the problems of normal and curvature estimation, we observed that the tangent space induced by the exact normals on noisy meshes causes a higher average error than the approximated normals for the methods which fit surfaces. While Goldfeather and Interrante do not observe a significant change in similar tests, we explain our observation by the disadvantageous tangent plane for quadratic and cubic surface fitting. Second, we were also interested in how the accuracy behaves if we smooth the mesh or the curvature values instead of using a larger estimation range. Finally, relative error e_{rel} is sometimes unstable, especially when both principal curvatures are close to zero. To the quadratic mean we only include points with less than than 100% error, since outliers would significantly change the result without much practical meaning.

5.2. Independent factors

The presented version of the approaches **My** and **CSM** are restricted to the topology of a closed triangular mesh without features. All other estimators can easily be modified in such a way, that also border and feature edges exist which disrupt the (topological) circle on the mesh. The **My**, **CSM** and **RU** approaches are limited to triangle meshes while all other approaches can also be used with general polygon meshes and sometimes even other representations, such as point clouds. Note that this statement does not indicate how accurately they perform on such meshes.

The locality of the estimators and the possibility to parallelize them for huge datasets or to use them locally is also implicit in the approaches. Assuming that we can estimate a vertex normal from its local neighborhood, the **QS**, **NC**, and **My** approach depend only on the 1-ring. If normals are available for all vertices, then the **AN** and our **MRU** method are local, too. The original **RU** method, the **CSM** approach, as well as all extended range versions presented in this paper depend on information beyond the 1-ring or necessitate separate per-triangle or per-vertex steps.

Runtime complexity is another factor which we can assess based on the experiences from the test runs. However, one must be careful with those results since not all our implementations are optimized. We still share our observations since the measurements match our estimations of the methods algorithmic complexities.

The **QS** ($R1$), the **NC** ($R1$) and our **MRU** ($R2$) are finished fastest (this does include the time for estimating the normals). If those need 100% time, then the **My** approach ($R1$) and **RU** ($R2$) need 200% (they pass twice over the mesh) and the **AN** ($R2$) 300% (7×7-matrix inversion for each vertex). Our implementation of the **CSM** approach for range $R2$ needs about ten times longer than the fastest estimators (due to the edge-collection and intersection with the geodesic disk). All estimators with an extended range also require a nearly constant time (1000%) for each additional range step (i.e. linear increase from $R1$ to $R4$) due to the collection of neighboring vertices.

5.3. Interdependent factors

To comprehend and discuss the data collected in our study, we have to reduce the dimensions of our results. We first show which approximations are feasible in the data, decoupling some of the factors from others. Later we analyze and interpret the remaining information.

The irregularity of the mesh was tested in the three steps of 0°, 14° and 25° angle RMS difference from the regular 60° in each angle of the triangulation. Tab. 1

Table 1. Increase in error with increasing irregularity. The rows show the $R1$ to $R3$-estimators. The percentages are relative to the respective 0° error: this 0° error differs for the various approaches. All values are for meshes without noise and averaged over the remaining parameters.

Irreg.	**CSM**	**QS**	**NC**	**My**		
14°	86%	479%	466%	343%		
25°	111%	828%	810%	550%		
Irreg.	**CSM**	**QS**	**NC**	**AN**	**MRU**	**RU**
14°	110%	207%	212%	374%	352%	313%
25°	129%	486%	516%	820%	526%	476%
Irreg.	**CSM**	**QS**	**NC**	**AN**	**MRU**	
14°	107%	134%	135%	279%	127%	
25°	124%	214%	221%	691%	176%	

shows the average increase in estimation error, where the 0° difference is used as 100%. The average is built over all sampling densities and functions, but without noise. It can be seen that the error of most estimators increases proportionally to our irregularity measure. The tensor average of **CSM** is significantly more robust against irregular triangulation than other estimators. When the estimator range increases, this degradation is reduced, which is easily explained by the better representation of the original surface. This also leads to the consideration that the increase in error is not only caused by the estimator, but also by the worse representation of the surface when irregularity increases. We explain the peculiar decrease in error for the **CSM** approach (in $R1$, first row) for an increased irregularity by the higher maximum distance in the 1-Ring (which in turn controls the size of the geodesic disc used in the estimation). Therefore, slightly more geometric information is collected in the case of a slightly skewed mesh. One must also be careful not to compare the estimators based on those values alone since they are relative to the estimator's accuracy at 0° irregularity. In the following analysis, we reduce the dimensionality by averaging the results over all three levels of irregularity.

Of the three functions used, we found that the elliptic function, followed by the trigonometric function, often give the best results if averaged over all estimators. After analyzing a subset of our data, we conclude that this is not caused by differences in the estimators, but by the function choice. While we ensured a comparable maximum in curvature values, the distribution of the values may vary. We use the average of the results over all functions since it reduces the benefit some estimators have on particular surfaces.

Next, we separate the noise factor from our combined analysis. It is clearly visible in the data that the estimation error increases (as was to be expected) with a more noisy representation. The behavior of each estimator in the presence of noise is similar for different levels of noise introduced. It can be described as linear in our samples in the range between 33% and 100% noise (of 10% median edge length). This supports Rusinkiewicz's observations in [11]. Because of this, we limit the analysis here to the two settings: no noise and 100% noise. Through preliminary tests with uniform noise (with equivalent variance), we assume that the observations remain valid under such conditions.

The remaining data is shown in Fig. 8 and contains the three dimensions: mesh density, estimator and the resulting relative error. The density is specified by the number of vertices used in each test mesh and the height of each bar denotes the RMS of the relative error e_{rel}. While we investigate the significance of the total height of all bars (densities) for an estimator, only the height of the

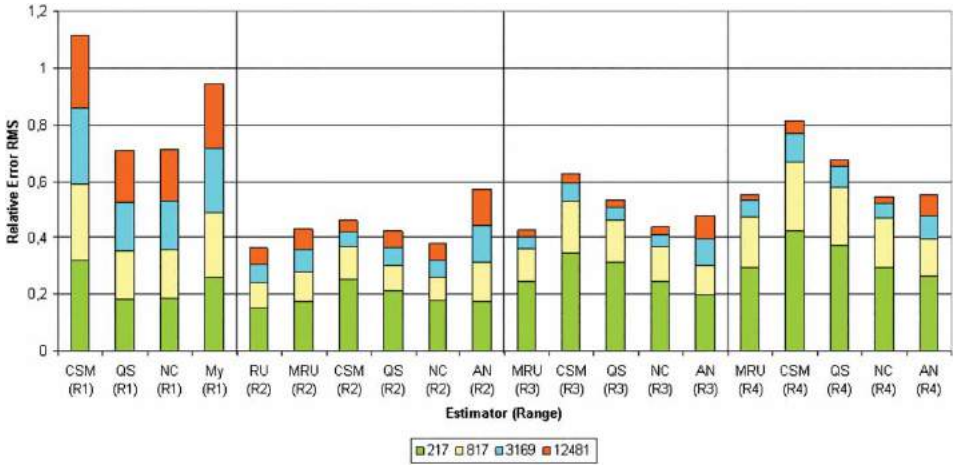

Figure 8. Error of the curvature estimators for four sampling densities (from 217 to 12481 vertices), with different ranges and without noise. The bars indicating the error are stacked for visualization purpose only and the error for each density is only the height of the respective (single-colored) bar.

single bars gives the correct error. When investigating the behavior of the lowest density mesh (217 vertices), it can be seen that the accuracy is highest for the ranges $R1$ and $R2$. Higher estimation ranges take into account a too large part of the mesh to register large curvature values. As the density increases, the higher estimation ranges start to perform better. This can be seen clearly with the most dense mesh (12481 vertices), for which the error is greatly reduced in the $R4$ estimators.

These results are still connected to the irregularity in the meshes, which has been averaged in this example. If we reduce the irregularity, most of the $R1$-estimators perform much better (see Tab. 1) and the the increase in error from $R1$ to $R4$ seems linear.

As expected, the interdependence of the range and the density is recognizable in the data. When it is known in which range the original surface was smooth, a proper curvature estimation range can be chosen for the best result. As an example in our data, the mesh with 817 vertices is best processed with the $R2$-estimators and the more dense meshes get better results with larger estimation ranges. If such information is not available, the error will be similar to what the average of the different densities represents (represented in the graph as the total height of columns).

Several observations presented by other researchers are confirmed in this data. Consider the **AN** ($R2$), the **QS** ($R1$) and **NC** ($R1$): the data show that the former outperforms the other two for all densities. However, if the version of the three estimators with equal ranges (and thus equal interpretation of the data) are compared, especially the **NC** often performs better. Note that these estimators have to collect a different number of vertices even with the same range. Furthermore, the improved performance of **RU** is reproduced for dense irregular meshes[11]. However, if the irregularity is reduced (as discussed above), the lower density meshes profit more from an $R1$-estimator.

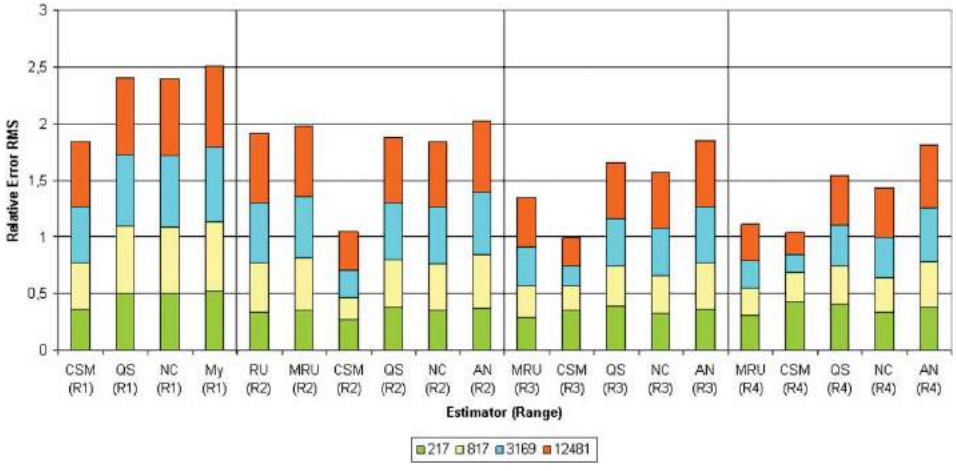

Figure 9. Error of the curvature estimators for four sampling densities (from 217 to 12481 vertices), with different ranges and with noise. The bars indicating the error are stacked only as visualization.

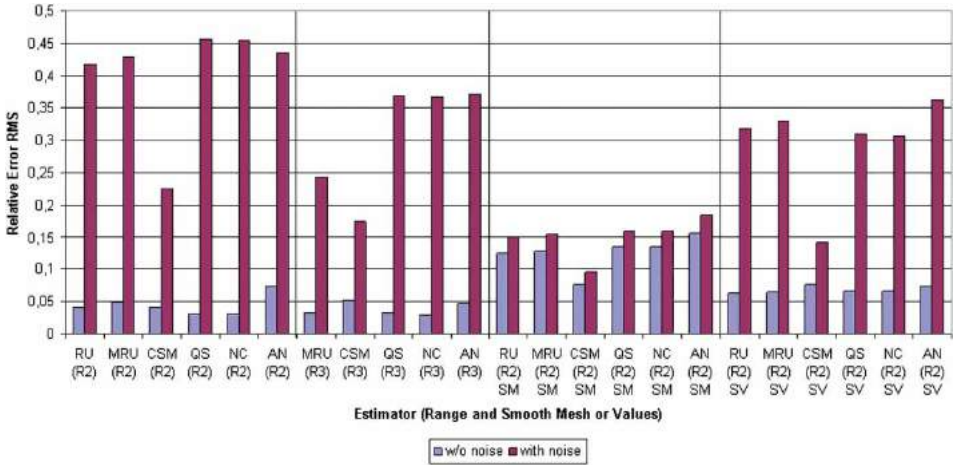

Figure 10. Error of the curvature estimators, highest sampling density.

The graph in Fig. 9 shows the same analysis on noisy surfaces (10% of the median edge length, Gaussian distribution). Note that the scale of this graph is substantially different from the graph without noise. As expected, this is caused by drop in performance of many estimators when noise is present. Notably, the **CSM** method handles noise comparably well (see **CSM** (*R*2) to **CSM** (*R*4)). Second comes the proposed **MRU** version, when a larger range is allowed (see **MRU** (*R*3) and **MRU** (*R*4)).

In Fig. 10, the increase of estimation range is compared to mesh smoothing using the Cotan-Laplacian[4]. A density was selected for which a large estimation range can still capture significant curvature values (3169 vertices). Popular *R*2-estimators and their *R*3-extensions are compared to smoothing either the mesh before (*R*2*SM*) or the curvature values after (*R*2*SV*) the *R*2 curvature estimation.

The *R*3-estimators are useful mainly without noise. Please note that while smoothing looks like the better option for noisy meshes, the average error cannot capture whether more features are lost than with the *R*3-estimators.

6. Results and conclusions

We identified three major mechanisms to be considered when choosing a curvature estimator for a particular data set.

Estimation range has a major influence on accuracy, data interpretation and runtime properties. While this range must be small enough to capture all relevant details in an object, a larger range improves accuracy (if the surface is smooth) and robustness against noise (see also Fig. 8). Comparing estimators with different ranges is not sufficient and we presented an evaluation which compares multiple estimators on equal (and different) ranges. We presented how popular estimators can be extended for this purpose. Increasing the estimation range produces *different results* than a smoothing operation. Any operation using curvature values should be aware of the profound influence of the estimation range.

Noise and irregularity of the data are highly relevant factors for choosing an estimator. Without noise, the method of Rusinkiewicz[11] and its modification proposed here perform best also for irregular meshes, while on more regular meshes the approaches from Goldfeather and Interrante[5] perform very well too. With noisy data, the method by Cohen-Steiner and Morvan [3] is less prone to estimation errors. We have demonstrated that smoothing the noisy meshes gives better results than smoothing curvature values afterwards.

Practical properties of the approaches were also compared and can be used to select the estimator. The Quadratic Surface and Normal Curvature Approximation as well as our modified Rusinkiewicz method are easiest to implement and have the fastest runtime.

We conclude that among the reviewed curvature estimators, there is none outperforming the others in all aspects. Our modification to the estimator by Rusinkiewicz[11] demonstrated how complex such a comparison is. The modified method is faster than all other estimators with comparable estimation range. It is easier to implement and has a higher locality than the original method, and it has a higher accuracy on larger estimation ranges. At the same time, the increased range might prevent it from finding some details. It also gives better results than the method of Cohen-Steiner and Morvan[3], but only for meshes without noise.

We suggest that future curvature estimation research evaluates any new estimation approach on comparable settings, especially for the estimation range. It is further strongly advised to evaluate the particular estimator with meshes of varying irregularity, density and levels of noise.

Acknowledgements

The authors would like to thank Markus Werner and Marco Proehl from the Fraunhofer Institute for Machine Tools and Forming Technology (Germany) for providing the models in Fig. 1 and 4 and Dr. David Brunner from Image Instruments GmbH (Germany) for the model in Fig. 8. Libor Váša was supported by the project LO1506 of the Czech Ministry of Education, Youth and Sports.

ORCiD

Libor Váša http://orcid.org/0000-0002-0213-3769
Tom Kühnert http://orcid.org/0000-0002-5824-0088
Guido Brunnett http://orcid.org/0000-0002-8224-015X

References

[1] Alliez, P.; Cohen-Steiner, D.; Devillers, O.; Levy, B.; Desbrun, M.: Anisotropic polygonal remeshing, ACM Trans. Graph., 22(3), 2003, 485–493. http://dx.doi.org/10.1145/1201775.882296.

[2] Cazals, F.; Pouget, M.: Estimating differential quantities using polynomial fitting of osculating jets, in Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 177–187, SGP '03, Eurographics Association, 2003, ISBN 1-58113-687-0.

[3] Cohen-Steiner, D.; Morvan, J.-M.: Restricted delaunay triangulations and normal cycle, in Proceedings of the Symposium on Computational Geometry, 312–321, SCG '03, ACM, 2003, ISBN1-58113-663-3. http://dx.doi.org/10.1145/777792.777839.

[4] Desbrun, M.; Meyer, M.; Schroder, P.; Barr, A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow, in Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques,

317–324, SIGGRAPH '99, 1999, ISBN 0-201-48560-5, http://dx.doi.org/10.1145/311535.311576.

[5] Goldfeather, J.; Interrante, V.: A novel cubic-order algorithm for approximating principal direction vectors, ACM Trans. Graph., 23(1), 2004, 45–63. http://dx.doi.org/10.1145/966131.966134.

[6] Hildebrandt, K.; Polthier, K.: Generalized shape operators on polyhedral surfaces, Comput. Aided Geom. Des., 28(5):321–343, 2011. http://dx.doi.org/10.1016/j.cagd.2011.05.001.

[7] Kalogerakis, E.; Simari, P.; Nowrouzezahrai, D.; Singh, K.: Robust statistical estimation of curvature on discretized surfaces, in Proceedings of the Fifth Eurographics Symposium on Geometry Processing, 13–22, SGP '07, Eurographics Association, 2007, ISBN 978-3-905673-46-3.

[8] Kim, H.-S.; Kim, H.-S.: New computation of normal vector and curvature, W. Trans. on Comp., 8(10), 2009, 1661–1670.

[9] Max, N.: Weights for computing vertex normals from facet normals, J. Graph. Tools, 4(2), 1999, 1–6. http://dx.doi.org/10.1080/10867651.1999.10487501.

[10] Meyer, M.; Desbrun, M.; Schröder, P.; Barr, A.: Discrete Differential Geometry Operators for Triangulated 2-Manifolds, in International Workshop on Visualization and Mathematics, 2002. http://dx.doi.org/10.1007/978-3-662-05105-42.

[11] Rusinkiewicz, S.: Estimating curvatures and their derivatives on triangle meshes, in Symposium on 3D Data Processing, Visualization, and Transmission, 2004. http://dx.doi.org/10.1109/3DPVT.2004.54.

[12] Taubin, G.: Estimating the tensor of curvature of a surface from a polyhedral approximation, in Proceedings of the Fifth International Conference on Computer Vision, 902, ICCV '95, IEEE Computer Society, 1995. http://dx.doi.org/10.1109/ICCV.1995.466840.

[13] Theisel, H.; Rossl, C.; Zayer, R.; Seidel, H.-P.: Normal based estimation of the curvature tensor for triangular meshes, in Proceedings of the Computer Graphics and Applications, 12th Pacific Conference, 288–297, PG '04, IEEE Computer Society, 2004, ISBN 0-7695-2234-3, http://dx.doi.org/10.1109/PCCGA.2004.1348359.