Surface Mesh Discrete Curvature Estimators

Release 0.01

Arnaud Gelas, Alexandre Gouaillard
and Sean Megason

September 7, 2008

Department of System Biology, Harvard Medical School,
Boston, MA 02115, USA

Abstract

Computing local curvatures of a given surface is important for applications, shape analysis, surface segmentation, meshing, and surface evolution. For a given smooth surface (with a given analytical expression which is sufficiently differentiable) curvatures can be analytically and directly computed. However in real applications, one often deals with a surface mesh which is an insufficiently differentiable approximation, and thus curvatures must be estimated. Based on a surface mesh data structure (`itk::QuadEdgeMesh` [3]), we introduce and implement curvature estimators following the approach of Meyer*et al.* [4]. We show on a sphere that this method results in more stable curvature approximations than the commonly used discrete estimators (as used in VTK: `vtkCurvatures`).

Contents

1 Introduction

Local curvatures provide information about the local behavior of the surface in vicinity of a given point, and so play an important role when working with surfaces. For example, the Gaussian curvature gives information about the location of the surface with the tangent plane at the considered point; a positive value means that the surface is locally either a peak or a valley; a negative value means that the surface is locally a saddle; a null value means the surface is locally flat in at least one direction (planar or cylindrical). The computation of local curvatures is a necessary step in many applications such as shape analysis, surface segmentation, adaptive surface meshing, remeshing, and surface evolution with active contours.

Unfortunately local curvatures are only defined for smooth surfaces, and when dealing with surface meshes only estimators are available. Over the last decade, many estimators have been proposed in the literature [1]. Here we first introduce differential properties of 2-manifold smooth surfaces (in section 2.1), then we present discrete estimators from [4] that we implemented (in section 2.2) and provide information about their use in an application (see section 3-4). Finally in section 5, we test our method on a sphere mesh and compare our results with the theoretical ones and with the usual discrete estimators (as it is used in VTK: `vtkCurvatures`).

2 Background

Taken from [1], we first introduce differential properties of 2-manifold surfaces and then give details from the discrete curvature estimators from [4].

2.1 Differential Geometry

Consider a continuous surface $\mathcal{S} \subset \mathbb{R}^3$ given in a parametric form as

$$\mathbf{x}(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix}, \ (u,v) \in \mathbb{R}^2 \tag{1}$$

where x, y, z are differentiable functions in u and v. The partial derivatives $\mathbf{x}_u$ and $\mathbf{x}_v$ span the tangent plane to $\mathcal{S}$ at a given location $\mathbf{p}$. Assuming a regular parametrization ($\mathbf{x}_u \times \mathbf{x}_v$), the normal vector is then given as $\mathbf{n} = \mathbf{x}_u \times \mathbf{x}_v / \|\mathbf{x}_u \times \mathbf{x}_v\|$.

First Fundamental form

The *first fundamental form* of $\mathbf{x}$ is given by the matrix

$$\mathbf{I} = \begin{bmatrix} E & F \\ F & G \end{bmatrix} = \begin{bmatrix} \mathbf{x}_u^T\mathbf{x}_u & \mathbf{x}_u^T\mathbf{x}_v \\ \mathbf{x}_u^T\mathbf{x}_v & \mathbf{x}_v^T\mathbf{x}_v \end{bmatrix} \tag{2}$$

which defines an inner product on the tangent space of $\mathcal{S}$.

Second Fundamental form

The *second fundamental form* is defined as

$$\mathbf{II} = \begin{bmatrix} e & f \\ f & g \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{uu}^T\mathbf{n} & \mathbf{x}_{uv}^T\mathbf{n} \\ \mathbf{x}_{uv}^T\mathbf{n} & \mathbf{x}_{vv}^T\mathbf{n} \end{bmatrix} \tag{3}$$

Alternatively, $\mathbf{II}$ can be expressed using the identities $\mathbf{x}_{uu}^T\mathbf{n} = -\mathbf{x}_u^T\mathbf{n}_u$, $\mathbf{x}_{uv}^T\mathbf{n} = \mathbf{x}_{vu}^T\mathbf{n} = -\frac{1}{2}\left(\mathbf{x}_u^T\mathbf{n}_v + \mathbf{x}_v^T\mathbf{n}_u\right)$.

The symmetric bilinear first and second fundamental forms allows the length, angles, area, and curvatures on the surface to be measured.

Normal Curvature

Let $\mathbf{t} = a\mathbf{x}_u + b\mathbf{x}_v$ be a unit vector in the tangent plane at $\mathbf{p}$, represented as $\bar{\mathbf{t}} = (a,b)^T$ in the local coordinate system. The *normal curvature* $\kappa_n(\bar{\mathbf{t}})$ is the curvature of the planar curve that results from intersecting $\mathcal{S}$ with the plane through $\mathbf{p}$ spanned by $\mathbf{n}$ and $\mathbf{t}$. The normal curvature in direction $\bar{\mathbf{t}}$ can be expressed in terms of the fundamental forms as

$$\kappa_n(\bar{\mathbf{t}}) = \frac{\bar{\mathbf{t}}^T \cdot \mathbf{II} \cdot \bar{\mathbf{t}}}{\bar{\mathbf{t}}^T \cdot \mathbf{I} \cdot \bar{\mathbf{t}}} = \frac{e\,a^2 + 2f\,ab + g\,b^2}{E\,a^2 + 2F\,ab + G\,b^2} \tag{4}$$

Curvature Tensor

The minimal normal curvature κ_1 and the maximal normal curvature κ_2 are called the *principal curvatures*. The associated tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are called *principal directions* and are always perpendicular to each other.

The principal curvatures are also obtained as eigenvalues of the *Weingarten curvature matrix* (or second fundamental tensor)

$$\mathbf{W} = \frac{1}{EG-F^2}\begin{bmatrix} eG-fF & fG-gF \\ fE-eF & gE-fF \end{bmatrix} \tag{5}$$

$\mathbf{W}$ represents the Weingarten map or shape operator, which measures the directional derivative of the normal, *i.e.* $\mathbf{W}\bar{\mathbf{t}} = \frac{\partial \mathbf{n}}{\bar{\mathbf{t}}}$. This allows the normal curvature to be expressed as

$$\kappa_n(\bar{\mathbf{t}}) = \bar{\mathbf{t}}^T \cdot \mathbf{W} \cdot \bar{\mathbf{t}}$$

With a local coordinate system defined by the principal directions $\mathbf{t}_1$ and $\mathbf{t}_2$, $\mathbf{W}$ is a diagonal matrix:

$$\mathbf{W} = \begin{bmatrix} \bar{\mathbf{t}}_1 & \bar{\mathbf{t}}_2 \end{bmatrix} \begin{bmatrix} \kappa_1 & 0 \\ 0 & \kappa_2 \end{bmatrix} \begin{bmatrix} \bar{\mathbf{t}}_1 \\ \bar{\mathbf{t}}_2 \end{bmatrix} \tag{6}$$

Then the normal curvature can also be written as

$$\kappa_n(\bar{\mathbf{t}}) = \kappa_n(\phi) = \kappa_1 \cos^2\phi + \kappa_2 \sin^2\phi \tag{7}$$

where ϕ is the angle between $\bar{\mathbf{t}}$ and $\bar{\mathbf{t}}_1$.

The *curvature tensor* $\mathbf{T}$ is expressed as a symmetric 3×3 matrix with the eigenvalues $\kappa_1, \kappa_2, 0$ and the corresponding eigenvectors $\mathbf{t}_1, \mathbf{t}_2, \mathbf{n}$. The tensor $\mathbf{T}$ measures the change of the unit normal with respect to a tangent vector $\mathbf{t}$ independently of the parametrization. It can be constructed as

$$\mathbf{T} = \mathbf{P} \cdot\ \mathbf{D} \cdot\ \mathbf{P}^{-1} \tag{8}$$

with $\mathbf{P} = [\mathbf{t}_1, \mathbf{t}_2, \mathbf{n}]$ and $\mathbf{D} = \mathrm{diag}(\kappa_1, \kappa_2, 0)$.

Gaussian Curvature

The *Gaussian curvature* K is defined as the product of the principal curvatures

$$K = \kappa_1 \cdot \kappa_2 = \det(\mathbf{W}) \tag{9}$$

Gaussian curvature can also be expressed as

$$K = \lim_{diam(\mathcal{A}) \to 0} \frac{\mathcal{A}^{\mathcal{G}}}{\mathcal{A}} \tag{10}$$

where $\mathcal{A}^{\mathcal{G}}$ is the area of the image of the Gauss map (also called the spherical image) associated with infinitesimal surface $\mathcal{A}$.

Mean Curvature

The *mean curvature* H is defined as the average of the principal curvatures

$$H = \frac{\kappa_1 + \kappa_2}{2} = \frac{1}{2}\mathrm{trace}(\mathbf{W}) \tag{11}$$

or can alternatively be defined as the continuous average of the normal curvatures

$$H = \frac{1}{2\pi} \int_0^{2\pi} \kappa_n(\phi)\, d\phi \tag{12}$$

Lagrange noticed that $H = 0$ is the Euler-Lagrange equation for surface area minimization. This provides a direct link between surface area minimization and mean area curvature flow:

$$2H \cdot \mathbf{n} = \lim_{diam(\mathcal{A}) \to 0} \frac{\nabla \mathcal{A}}{\mathcal{A}} \tag{13}$$

where $\mathcal{A}$ is an infinitesimal area around a point $\mathbf{p}$ on the surface, $diam(\mathcal{A})$ its diameter.

2.2 Discrete Differential Operators

The differential properties defined in the previous sections require a surface to be sufficiently differentiable, at least it requires the existence of the second derivatives. Since polygonal meshes are piecewise linear surface, the concepts introduced above cannot be applied directly. Thus these differential properties of the underlying surface must be approximated from the mesh data. Several approaches have been proposed in recent years (see [1]), and here we only describe the ones proposed by Meyer *et al.* [4].

Discrete Gaussian Curvature

Meyer *et al.* [4] express the discrete Gaussian curvature as:

$$K(\mathbf{p}_i) = \frac{2\pi - \sum_{j=1}^{\#f} \theta_j}{\mathcal{A}_{\text{Mixed}}} \tag{14}$$

where θ_j is the angle of the j-th face at the vertex $\mathbf{p}$ (see Figure 1(a)), $\#f$ denotes the number of faces around this vertex, and $\mathcal{A}_{\text{Mixed}}$ is the area of the region of influence of a vertex (see the corresponding paragraph below).

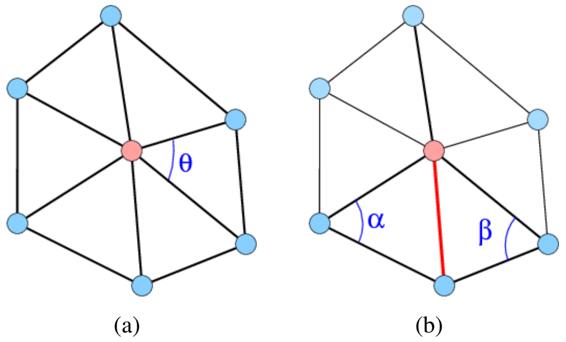


Figure 1: θ is the angle of one face at a vertex $\mathbf{p}$ used for computation of the discrete Gaussian curvature (a). α and β are the opposite angles to a given edge (here in red) used for computing the discrete mean curvature (b).

Discrete Mean Curvature

Following [4] the mean curvature can be approximated by

$$H(\mathbf{p}_i) = \frac{1}{4\mathcal{A}_{\text{Mixed}}} \left\| \sum_{j \in N(i)} (\cot\alpha_{ij} + \cot\beta_{ij})(\mathbf{p}_i - \mathbf{p}_j) \right\| \tag{15}$$

where α_{ij} and β_{ij} are the angle opposite to the edge $\mathbf{p}_i\mathbf{p}_j$ (see Figure 1(b)), $N(i)$ is the set of vertices in the 0-ring of $\mathbf{p}_i$, $\mathcal{A}_{\text{Mixed}}$ is the area of the region of influence of a vertex (see the corresponding paragraph below).

Region of influence

The *region of influence* of a given vertex does not overlap with any other region of influence, and represents a local part of the surface on the vicinity of $\mathbf{p}$. Following the definition of Meyer *et al.* [4], there are 3 different configurations for its computation (see Figure 2), and thus the computation of its area follows the algorithm:

- $\mathcal{A}_{\text{Mixed}} = 0$
- For each triangle T from the 1-ring neighborhood of $\mathbf{p}$

```
– If $T$ is non-obtuse (Voronoi safe)
  $\mathcal{A}_{\text{Mixed}}$ += Voronoi region of $\mathbf{p}$ in $T$
– Else
    ∗ If the angle of $T$ at $\mathbf{p}$ is obtuse
      $\mathcal{A}_{\text{Mixed}}$ += area(T) / 2
    ∗ Else
      $\mathcal{A}_{\text{Mixed}}$ += area(T) / 4
```

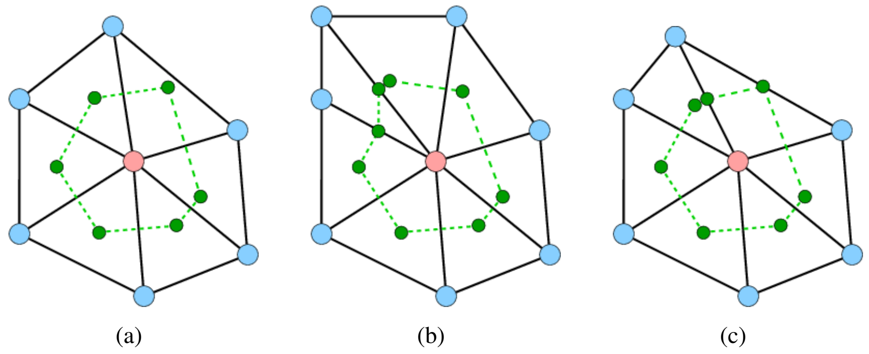


Figure 2: Here we show the 3 different configurations for the region of influence of a given vertex $\mathbf{p}$. The region of influence is delimited by the green lines, and its area is $\mathcal{A}_{\text{Mixed}}$. All triangles are non-obtuse and the region of influence is identical to the Voronoi region (a). If there is one obtuse triangle, the influence region is then composed of Voronoi region for each non-obtuse triangle, then for the obtuse ones the corresponding region differs if the angle at the vertex $\mathbf{p}$ is obtuse (b) or not (c).

Principal Curvatures

From Eq. 9- 11, the principal curvatures can then be computed as

$$\kappa_1(\mathbf{p}) = H(\mathbf{p}) + \sqrt{\Delta(\mathbf{p})} \quad (16)$$

$$\kappa_2(\mathbf{p}) = H(\mathbf{p}) - \sqrt{\Delta(\mathbf{p})} \quad (17)$$

where $\Delta(\mathbf{p}) = H^2(\mathbf{p}) - G(\mathbf{p})$, $G(\mathbf{p})$ is given by Eq. 14 and $H(\mathbf{p})$ by Eq. 15.

While the continuous case Δ is always positive, in the discrete case we must ensure that it is, by thresholding it to zero (even if it occurs rarely).

3 Implementation

We provide implementations of the previous discrete curvature estimators, *i.e.*

- Gaussian curvature (`itk::QEMeshDiscreteGaussianCurvatureEstimator`),
- mean curvature (`itk::QEMeshDiscreteMeanCurvatureEstimator`),
- minimal curvature (`itk::QEMeshDiscreteMinCurvatureEstimator`),
- maximal curvature (`itk::QEMeshDiscreteMaxCurvatureEstimator`).

4 Usage

These 4 filters are used in the same way, and in the following example we assume the user wants to compute the Gaussian curvature of a given mesh. However to compute the mean, minimal or maximal curvature, the user only needs to define the corresponding estimator as the `CurvatureFilterType`, see given source code `DiscreteCurvatureEstimator.cxx`.

Note. *If the connectivity of the starting mesh is not important, we recommend prefiltering the mesh through a Delaunay conforming filter (*`itk::DelaunayConform`*[2]) to improve the accuracy of curvature approximations.*

5 Results

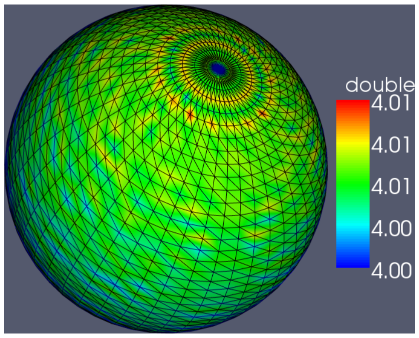
To validate our implementations, we estimate curvatures on a sphere of radius 0.5. All the curvatures are known for the smooth surface and thus we can easily compare the theoretical values to the experimental ones (see Figure 3).

- Gaussian curvature K is supposed to be 4 at any point of the sphere. In practice with our filter, we get 4 with a maximal error below 1% (see Figure 3(a)).
- mean curvature H is supposed to be 2 at any point of the sphere. In practice with our filter, we get 2 with a maximal error below 1% (see Figure 3(b)).
- $\Delta = H^2 - K$ is supposed to be 0 at any point of the sphere. In practice some negative value occurs, and there the maximum error is about 1% (see Figure 3(c)).
- min curvature κ_1 is supposed to be 2 at any point of the sphere. In pratice with our filter, we get 2 with a maximal error error 7% (see Figure 3(d)).
- max curvature $\kappa_2 = 2$ is supposed to be 2 at any point of the sphere. In pratice with our filter, we get 2 with a maximal error error 7% (see Figure 3(e)).

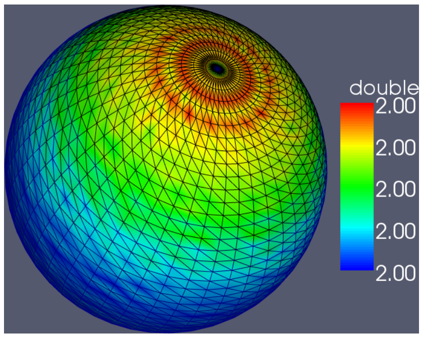
Here we compare the results on the sphere of our filter with the one provided in VTK (see Figure 4-5). Both estimators provide excellent approximations of curvature where the vertex degree is 6, however vertices with extraordinary degree and really elongated triangles, *i.e.* poles, exhibit an important error when using usual estimators. As claimed by Meyer*et al.* [4] their estimators is less dependent to the triangle shape and vertex degree.

Acknowledgment

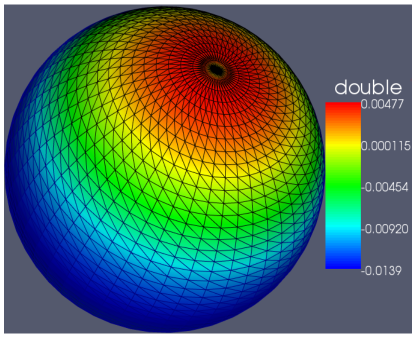
This work was funded by a grant from the NHGRI (P50HG004071-02) to found the Center for in toto genomic analysis of vertebrate development.



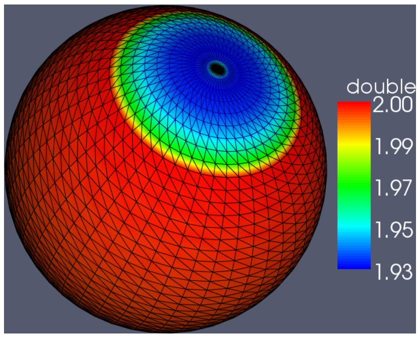
(a) Gaussian K



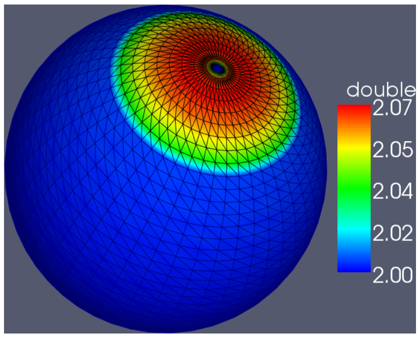
(b) mean H



(c) $H^2 - K$



(d) κ_1



(e) κ_2

Figure 3: Results using our implementations for discrete curvature estimators on a sphere.

References

[1] Mario Botsch, Mark Pauly, Leif Kobbelt, Pierre Alliez, Bruno Lévy, Stephan Bischoff, and Christian Rössl. Geometric modeling based on polygonal meshes. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, page 1, New York, NY, USA, 2007. ACM. 2, 4

[2] Arnaud Gelas, Alexandre Gouaillard, and Sean Megason. Triangular meshes delaunay conforming filter. *Insight Journal*, July-December 2008. http://hdl.handle.net/1926/1489. 7

[3] Alexandre Gouaillard, Leonardo Florez-Valencia, and Eric Boix. itkQuadEdgeMesh: A discrete orientable 2-manifold data structure for image processing. *Insight Journal*, July-December 2006. http://hdl.handle.net/1926/306. 1

[4] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *VisMath*, Berlin, Germany, 2002. 1, 2, 4, 5, 7

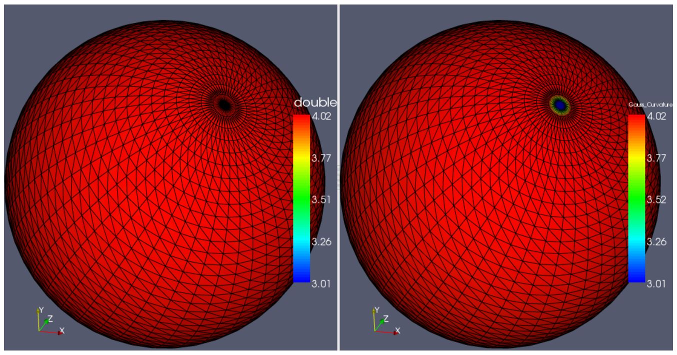


Figure 4: Comparison of our Gaussian curvature estimator on the left and the one provided by VTK.

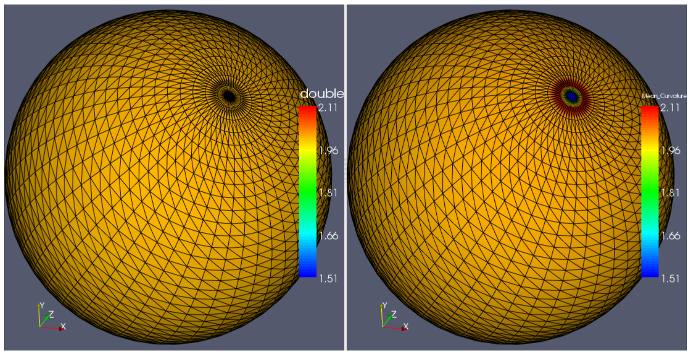


Figure 5: Comparison of our mean curvature estimator on the left and the one provided by VTK.