Computer-Aided Design 45 (2013) 1639–1650

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Curvature tensor computation by piecewise surface interpolation☆

Xunnian Yang [a,*], Jianmin Zheng [b]

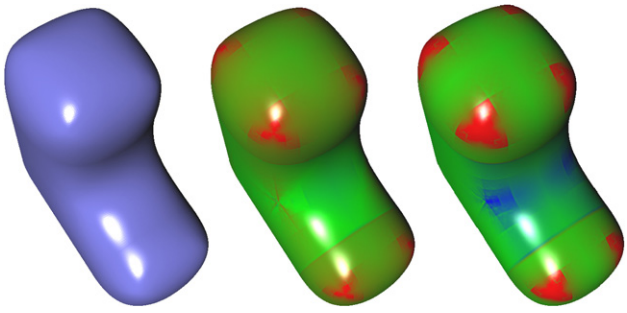
[a] Department of Mathematics, Zhejiang University, Hangzhou, China

[b] School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore

HIGHLIGHTS

- Closed form Taubin integral for smooth or piecewise smooth surfaces.
- An improved local surface interpolation scheme for triangular meshes.
- Novel method for accurate curvature tensor estimation of triangle meshes.

GRAPHICAL ABSTRACT



A Doo-Sabin subdivision mesh | mean curvature plot by the proposed method | Gaussian curvature plot by the proposed method

ARTICLE INFO

Article history:
Received 29 November 2012
Accepted 18 August 2013

Keywords:
Triangular meshes
Curvature tensor
Taubin integral
Piecewise surface interpolation
Closed form formula

ABSTRACT

Estimating principal curvatures and principal directions of a smooth surface represented by a triangular mesh is an important step in many CAD or graphics related tasks. This paper presents a new method for curvature tensor estimation on a triangular mesh by replacing flat triangles with triangular parametric patches. An improved local interpolation scheme of cubic triangular Bézier patches to vertices and vertex normals of triangle meshes is developed. Piecewise parametric surfaces that have C^0 continuity across boundary curves of adjacent patches and G^1 continuity at the joint vertices are obtained by the interpolation scheme. A closed form expression of Taubin integral – a 3×3 symmetric matrix in integral formulation – is derived based on the piecewise parametric surfaces. Principal curvatures and principal directions are then computed from the Taubin integral. The proposed method does not need to parameterize data points or solve a linear system which is usually required by other surface fitting methods. Compared to several state-of-the-art curvature estimation methods, the proposed method can generate more accurate results for general surface meshes. The experiments have demonstrated its accuracy, robustness and effectiveness.

1. Introduction

Principal curvatures (or equivalently mean and Gaussian curvatures) and principal directions are fundamental differential invariants in differential geometry. They are often referred to as the tensor of curvature and can be used to characterize the local shape of a surface. In CAD and computer graphics, curvatures are frequently used for shape processing [1], segmentation [2] and surface interrogation [3,4], and principal direction vectors are essential quantities for surface remeshing [5], non-photorealistic rendering [6,7] or texture mapping [8]. In computer vision and medical image analysis, these differential invariants are extensively used for recognition, registration, and free-form shape analysis [9,10]. Mathematically, the tensor of curvature is defined for twice differentiable surfaces. In practice geometric data are often

☆ This paper has been recommended for acceptance by Myung-Soo Kim.

* Corresponding author.
E-mail addresses: yxn@zju.edu.cn (X. Yang), ASJMZheng@ntu.edu.sg (J. Zheng).

0010-4485/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cad.2013.08.008

available as polygonal (typically triangular) meshes. Therefore when we discuss the curvatures on triangular meshes, we mean the curvatures of smooth surfaces that the meshes represent. Since in general we do not have the analytic representation of the underlying smooth surfaces, the true surface curvature information cannot be computed directly. In this paper, we present a novel method for accurately estimating the tensor of curvature of a subjacent, unknown, smooth surface from a triangular mesh.

1.1. Previous work

Many methods have been developed for estimating curvatures or curvature tensors on triangular meshes accurately or robustly. See for example Refs. [11,12,9,13,14] and references therein. When both principal curvatures and principal directions are estimated, the known techniques can roughly be classified into three categories.

(1) The first category estimates principal curvatures and principal directions by first estimating normal curvatures along a few sampled curves or directions [15–18]. By using Euler's theorem and Meusnier's theorem, a set of equations describing the relation between the principal curvatures and the pre-estimated normal curvatures is built. Then the principal curvatures and directions can be found by solving the equations using a least squares approach. The advantage of approaches of this category is efficiency and simplicity. However, the estimation accuracy is usually not very high and depends heavily on the sampling frequency, mesh resolution and mesh accuracy. The approaches may also suffer from robustness problems due to insufficient sampling rates.

(2) The second category is known as local surface fitting methods. The methods compute a simple analytical surface that locally approximates the mesh around a vertex and the principal curvatures and directions can then be computed by applying classic differential geometry methods to the obtained analytical surface [19–22]. Usually, 1, 2, or even 3-ring neighborhood of the mesh at a vertex is chosen as the local region and a lower order polynomial surface is typically used to fit to the points and/or normals of the chosen neighborhood. When the input data are accurate, local surface fitting often produces good estimation. However, surface fitting methods depend on which neighborhood to fit. This kind of method may suffer heavy computational cost or numerical stability problem because a linear system has to be solved for each vertex.

(3) The third category is discrete methods. Discrete methods attempt to calculate the curvature tensor directly from the local region of a mesh. Rusinkiewicz [23] proposed to compute curvature tensors using finite difference and least squares fitting. Cohen-Steiner and Morvan [24] proposed to compute average curvature tensor based on normal cycle theory. Another famous approach in this category is Taubin's integral method that computes the curvature tensor via eigen-analysis of a 3×3 symmetric matrix [25]. The 3×3 symmetric matrix, which we call *Taubin integral*, is defined by an integral of a normal-curvature-weighted second order symmetric tangent tensor. Given a triangular mesh, the Taubin integral for each vertex is approximated by a weighted tensor sum over the neighborhood around the vertex. Some variants or extensions have also been proposed to compute the tensor matrix from the input data [26–28]. In general, discrete methods are very efficient and somewhat robust. However, the accuracy of estimation is adversely affected by obtuse triangles in the mesh and valences other than four or six [12].

The performance of a particular curvature estimation method may be affected by many factors such as the presence of noise, valences of the mesh, mesh resolution, and normal vectors [12,9]. Nevertheless, the most important concerns about a curvature tensor estimation algorithm are its accuracy, robustness and efficiency. While many techniques have been developed to improve the robustness of curvature tensor estimation against data noise [23,29,30], they may generate inaccurate or blurred curvatures for graphics application. In other applications such as visualization of geometric details, edge detection, surface interrogation, etc., the accuracy of curvature tensor estimation plays an important role.

1.2. Our approach

Our work is inspired by Taubin's integral method. Taubin's method has a linear complexity, both in time and in space, as a function of the number of vertices of the mesh. All the computations are simple and direct with closed form expressions and the estimation results are reasonably accurate. However, it is noticed that in Taubin's approach, the calculation of integral is discretized based on the vertex valence and approximated based on the geometry of a triangular mesh. Since the input triangular mesh is only a linear approximation of a subjacent, unknown, smooth surface, we believe that for a triangular mesh with normal vectors at vertices, each triangle can be replaced by a surface patch that can better approximate the underlying surface and the computation based on the appropriately chosen patch can yield more accurate curvature tensor estimation. This motivates us to estimate curvature tensors on triangular meshes by computing continuous Taubin integral and interpolating piecewise smooth surfaces to the meshes.

The main contributions of the paper are as follows.

- A closed form expression is derived for the Taubin integral at the joint point of a set of parametric surface patches that may not be G^2 continuous but only have a common tangent plane at the point. The closed form Taubin integral at any point in a single surface patch is also obtained.
- An explicit, local scheme is presented for constructing piecewise cubic triangular Bézier patches to interpolate vertices and vertex normals of a triangle mesh. The free coefficients for every Bézier patch are carefully estimated for achieving high accuracy estimation of curvature tensors at vertices.
- A weighted sum of Taubin integrals is proposed, which leads to a robust scheme for curvature tensor estimation on meshes with data noise, and a unified algorithm is proposed to estimate curvature tensors for triangular meshes with accurate or noisy data.

1.3. Overview

Section 2 reviews pertinent background of surface curvatures. Section 3 derives a closed form formula for computing the Taubin integral at a point where the surface is formed by joining several parametric patches. Section 4 describes how to construct appropriate parametric surfaces interpolating triangles with given normal vectors at vertices. Section 5 outlines the algorithm steps for computing principal curvatures and principal directions for a given triangular mesh. The experimental examples and comparisons are given in Section 6. Section 7 concludes the paper.

2. Surface curvature review

Curvatures and the tensor of curvature are well studied in classical differential geometry (see for example Ref. [31]). The curvature tensor is closely related to the surface normal and normal curvature evaluation.

Assume that $\mathbf{p}$ is a point on a twice differentiable parameterized surface $X(u, v)$. The normal vector of the surface at point $\mathbf{p}$ can be obtained by $\mathbf{n} = \frac{X_u \times X_v}{\|X_u \times X_v\|}$, where X_u and X_v are the partial derivatives of $X(u, v)$ with respect to u and v, respectively. Let $\mathbf{r}(s) = X(u(s), v(s))$ be an arc length parameterized curve on the surface, which passes through point $\mathbf{p}$. The unit tangent vector of $\mathbf{r}(s)$ at point $\mathbf{p}$ can be computed by

$$T = \frac{d\mathbf{r}(s)}{ds} = X_u \frac{du}{ds} + X_v \frac{dv}{ds}. \tag{1}$$

From Eq. (1), we can obtain the squared length of an infinitesimal arc of curve $\mathbf{r}(s)$ near point $\mathbf{p}$:

$$ds^2 = Edu^2 + 2Fdudv + Gdv^2$$

where $E = X_u \cdot X_u$, $F = X_u \cdot X_v$ and $G = X_v \cdot X_v$.

With the surface normal and arc length defined, one may then compute the normal curvature $k_n(T)$ of the surface at point $\mathbf{p}$ along direction T as the curvature of a section curve which is the intersection of the original surface $X(u, v)$ and a plane through $\mathbf{p}$ spanned by vectors $\mathbf{n}$ and T. Let $L = \mathbf{n} \cdot X_{uu}$, $M = \mathbf{n} \cdot X_{uv}$ and $N = \mathbf{n} \cdot X_{vv}$, the normal curvature along direction T is computed by

$$k_n(T) = \frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2}. \tag{2}$$

Except for umbilics at which the normal curvatures in all directions are equal, there exist two orthogonal directions T_1 and T_2 along which the normal curvatures achieve the maximum value k_1 and the minimum value k_2, respectively. These two directions and curvatures are the principal directions and principal curvatures of the surface at the point. They can be computed by the Weingarten map in classical differential geometry.

Alternatively, Taubin [25] proposed to compute principal curvatures and principal directions using eigen-analysis of an integral matrix. Let T_ϕ be some unit length tangent vector at $\mathbf{p}$ of the surface where ϕ is the angle between T_ϕ and a fixed direction on the tangent plane. Taubin defined the 3×3 symmetric matrix M_p by the integral formula of

$$M_p = \frac{1}{2\pi} \int_{-\pi}^{\pi} k_n(T_\phi) T_\phi T_\phi^t d\phi, \tag{3}$$

where the upper t means the transpose of a vector. As T_ϕ is perpendicular to the normal vector $\mathbf{n}$, $\mathbf{n}$ is an eigenvector of M_p associated with the eigenvalue 0. Furthermore, it is proved in [25] that the other two eigenvectors of M_p are the principal direction vectors T_1 and T_2 of the surface at point $\mathbf{p}$ and their corresponding eigenvalues m_p^1 and m_p^2 are linear combinations of the principal curvatures: $m_p^1 = \frac{3}{8}k_1 + \frac{1}{8}k_2$ and $m_p^2 = \frac{1}{8}k_1 + \frac{3}{8}k_2$. Thus the principal curvatures k_1 and k_2 can be computed from the eigenvalues: $k_1 = 3m_p^1 - m_p^2$ and $k_2 = 3m_p^2 - m_p^1$.

3. Closed form expression for the Taubin integral

Note that the Taubin integral is given in integral formulation and the previous work computes it using discretization and approximation. In this section we derive a closed form formula for the evaluation of the Taubin integral M_p of (3) at a point where several patches with independent parameterization join with a common tangent plane to approximate the subjacent surface. If all the patches are curvature continuous with the subjacent surface at the point, the curvature tensor derived from M_p is exactly the one of the subjacent surface. Otherwise, M_p will produce an approximate curvature tensor for the subjacent surface.

Suppose n parametric surface patches are assembled with position continuity between every two adjacent patches and a common tangent plane at a joint point $\mathbf{p}$, as illustrated in Fig. 1. Denote the normal vector to the tangent plane by $\mathbf{n}$ and the tangent directions of the boundary lines of the surface patches at point $\mathbf{p}$ by $\bar{T}_i$ $(i = 0, 1, \ldots, n)$. Assume that the angle between vector $\bar{T}_i$ and a fixed direction on the tangent plane is ϕ_i, and the patch bounded by $\bar{T}_{i-1}$ and $\bar{T}_i$ is the ith surface patch $X_i(u, v)$. Then the Taubin integral at $\mathbf{p}$ can be computed by accumulating the integral for each surrounding surface patch:

$$M_p = \sum_{i=1}^{n} M_p^i = \sum_{i=1}^{n} \frac{1}{2\pi} \int_{\phi_{i-1}}^{\phi_i} k_n(T_\phi) T_\phi T_\phi^t d\phi. \tag{4}$$

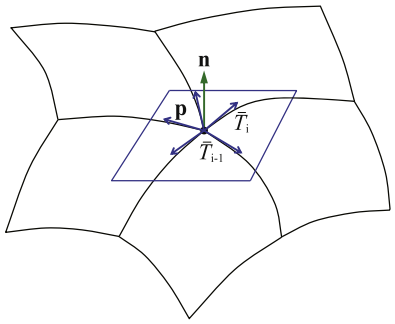


Fig. 1. Several patches surround a joint point with a common tangent plane.

Next we consider how to compute each M_p^i in Eq. (4). We convert the computation from the tangent space to the parameter domain of the surface. Without loss of generality, we assume that the parametric domain of the ith surface patch $X_i(u, v)$ is local rectangular at origin and $\mathbf{p} = X_i(0, 0)$. Noting that (du, dv) only represents a direction on the parametric domain, we can replace it by $(\cos\theta, \sin\theta)$ where θ is the angle between vector (du, dv) and the u-axis on the parameter domain. Thus the normal curvature $k_n(T)$ at point $\mathbf{p}$ can be computed by

$$k_n(T) = \frac{L_i \cos^2\theta + 2M_i \cos\theta \sin\theta + N_i \sin^2\theta}{E_i \cos^2\theta + 2F_i \cos\theta \sin\theta + G_i \sin^2\theta} \tag{5}$$

where $E_i = X_{iu} \cdot X_{iu}$, $F_i = X_{iu} \cdot X_{iv}$, $G_i = X_{iv} \cdot X_{iv}$, $L_i = \mathbf{n} \cdot X_{iuu}$, $M_i = \mathbf{n} \cdot X_{iuv}$ and $N_i = \mathbf{n} \cdot X_{ivv}$. For notational simplicity, in the following we may drop subscript "i" within L_i, M_i, N_i, E_i, F_i and G_i which are computed for the ith surface patch at the joint vertex.

From Eq. (1), the matrix of $T_\phi T_\phi^t$ can be formulated as

$$T_\phi T_\phi^t = R_1^i \left(\frac{du}{ds}\right)^2 + 2R_2^i \frac{du}{ds}\frac{dv}{ds} + R_3^i \left(\frac{dv}{ds}\right)^2,$$

where

$$\begin{cases} R_1^i = X_{iu} X_{iu}^t, \\ R_2^i = \dfrac{1}{2}(X_{iu} X_{iv}^t + X_{iv} X_{iu}^t), \\ R_3^i = X_{iv} X_{iv}^t. \end{cases} \tag{6}$$

In a similar way as we deal with the normal curvature, $T_\phi T_\phi^t$ can be rewritten as

$$T_\phi T_\phi^t = \frac{R_1^i \cos^2\theta + 2R_2^i \cos\theta \sin\theta + R_3^i \sin^2\theta}{E \cos^2\theta + 2F \cos\theta \sin\theta + G \sin^2\theta}. \tag{7}$$

As to the angle differential $d\phi$, it can be expressed by angle θ and its differential too. Assume that T_{ϕ_I} and $T_{\phi_{II}}$ are two unit vectors on the tangent plane that pass through point $\mathbf{p}$ and the vectors in the parameter domain corresponding to T_{ϕ_I} and $T_{\phi_{II}}$ are $(\cos\theta_I, \sin\theta_I)$ and $(\cos\theta_{II}, \sin\theta_{II})$, respectively. Then, from Eq. (1), we have

$$T_{\phi_I} = X_{iu}\frac{\cos\theta_I}{ds_I} + X_{iv}\frac{\sin\theta_I}{ds_I}$$

$$T_{\phi_{II}} = X_{iu}\frac{\cos\theta_{II}}{ds_{II}} + X_{iv}\frac{\sin\theta_{II}}{ds_{II}},$$

where $ds_I = (E\cos^2\theta_I + 2F\cos\theta_I \sin\theta_I + G\sin^2\theta_I)^{\frac{1}{2}}$ and $ds_{II} = (E\cos^2\theta_{II} + 2F\cos\theta_{II}\sin\theta_{II} + G\sin^2\theta_{II})^{\frac{1}{2}}$. From the expressions of T_{ϕ_I} and $T_{\phi_{II}}$, we have

$$T_{\phi_I} \times T_{\phi_{II}} = \frac{\sin(\theta_{II} - \theta_I)}{ds_I ds_{II}} (X_{iu} \times X_{iv}).$$

Without loss of generality, we assume that the direction of $T_{\phi_I} \times T_{\phi_{II}}$ is identical to the surface normal $\mathbf{n}$ at point $\mathbf{p}$. Then $T_{\phi_I} \times T_{\phi_{II}} = \sin(\phi_{II} - \phi_I)\mathbf{n}$. Since $X_{iu} \times X_{iv} = \|X_{iu} \times X_{iv}\|\mathbf{n}$, we obtain

$$\sin \triangle\phi = \|X_{iu} \times X_{iv}\| \frac{\sin \triangle\theta}{ds_I ds_{II}},$$

where $\triangle\phi = \phi_{II} - \phi_I$ and $\triangle\theta = \theta_{II} - \theta_I$. When $\triangle\phi$ and $\triangle\theta$ are small enough, we have

$$d\phi = \frac{\|X_{iu} \times X_{iv}\|}{E\cos^2\theta + 2F\cos\theta\sin\theta + G\sin^2\theta} d\theta. \tag{8}$$

Substituting (5), (7), and (8) into M_p^i of (4), and letting

$$D_i = \frac{1}{2\pi}\|X_{iu} \times X_{iv}\| = \frac{1}{2\pi}\sqrt{EG - F^2}$$

and

$$K(\theta) = \frac{L\cos^2\theta + 2M\cos\theta\sin\theta + N\sin^2\theta}{(E\cos^2\theta + 2F\cos\theta\sin\theta + G\sin^2\theta)^3},$$

give

$$\begin{aligned} M_p^i &= \frac{1}{2\pi}\int_{\phi_{i-1}}^{\phi_i} k_n(T_\phi) T_\phi T_\phi^t d\phi \\ &= D_i \int_0^{\frac{\pi}{2}} K(\theta)(\cos^2\theta R_1^i + 2\cos\theta\sin\theta R_2^i + \sin^2\theta R_3^i) d\theta \\ &= C_1^i R_1^i + C_2^i R_2^i + C_3^i R_3^i \end{aligned} \tag{9}$$

where

$$C_1^i = D_i \int_0^{\frac{\pi}{2}} K(\theta)\cos^2\theta d\theta,$$

$$C_2^i = D_i \int_0^{\frac{\pi}{2}} K(\theta) 2\cos\theta\sin\theta d\theta,$$

$$C_3^i = D_i \int_0^{\frac{\pi}{2}} K(\theta)\sin^2\theta d\theta.$$

The above integrals can be computed directly (see the Appendix for detailed derivation). Let $e = \frac{E}{\sqrt{EG-F^2}}, f = \frac{F}{\sqrt{EG-F^2}}$ and $g = \frac{G}{\sqrt{EG-F^2}}$, then the coefficients C_j^i $(j = 1, 2, 3)$ are

$$\begin{cases} C_1^i = \dfrac{aI_3 + bJ_3 + NI_2}{2\pi(EG - F^2)}, \\ C_2^i = \dfrac{-(af + b)I_3 + (a - bf)J_3 + (b - Nf)I_2 + NJ_2}{\pi G\sqrt{EG - F^2}}, \\ C_3^i = \dfrac{a'I_3 + b'J_3 + LI_2}{2\pi(EG - F^2)} \end{cases} \tag{10}$$

where

$$\begin{aligned} &a = Lg^2 - 2Mfg + N(f^2 - 1),\ b = 2(Mg - Nf), \\ &a' = L(f^2 - 1) - 2Mfe + Ne^2,\ b' = 2(Me - Lf), \\ &I_2 = \tfrac{\pi}{4} - \tfrac{1}{2}\arctan f - \tfrac{f}{2(1+f^2)},\ I_3 = \tfrac{3}{4}I_2 - \tfrac{f}{4(1+f^2)^2}, \\ &J_2 = \tfrac{1}{2(1+f^2)}, J_3 = \tfrac{1}{4(1+f^2)^2}. \end{aligned}$$

To sum up, the Taubin integral can be explicitly computed by

$$M_p = \sum_{i=1}^{n} \left(C_1^i R_1^i + C_2^i R_2^i + C_3^i R_3^i\right) \tag{11}$$

where C_j^i and R_j^i are given by (10) and (6), respectively.

3.1. Taubin integral for one parametric surface

In a similar way to compute the Taubin integral at the joint point of several surface patches, the above derivation can be adapted to derive a closed form expression for the Taubin integral of a parametric surface. Now there is only one parametric surface $X(u, v)$. Assume that $\mathbf{p}$ is a point on the surface. To compute the Taubin integral of the surface at $\mathbf{p}$, unlike the matrix M_p^i in Eq. (9) which is an integral on interval $[\phi_{i-1}, \phi_i]$, the integral now should be computed on interval $[0, 2\pi]$. Thus

$$\begin{aligned} M_p^X &= \frac{1}{2\pi}\int_0^{2\pi} k_n(T_\phi) T_\phi T_\phi^t d\phi \\ &= \frac{\sqrt{EG - F^2}}{2\pi}\int_0^{2\pi} K(\theta) \\ &\quad \times(\cos^2\theta R_1 + 2\cos\theta\sin\theta R_2 + \sin^2\theta R_3) d\theta \\ &= C_1 R_1 + C_2 R_2 + C_3 R_3 \end{aligned} \tag{12}$$

where $R_1 = X_u X_u^t$, $R_2 = \frac{1}{2}(X_u X_v^t + X_v X_u^t)$, $R_3 = X_v X_v^t$ and

$$C_1 = \frac{3LG^2 - 6MFG + N(EG + 2F^2)}{8(EG - F^2)^2},$$

$$C_2 = \frac{-3LFG + 2M(EG + 2F^2) - 3NEF}{4(EG - F^2)^2},$$

$$C_3 = \frac{L(EG + 2F^2) - 6MEF + 3NE^2}{8(EG - F^2)^2}.$$

By the integral matrix in Eq. (12) and Taubin's formula for eigen decomposition of the matrix, principal curvatures and principal directions of a parametric surface can be obtained in a new way other than the classical Weingarten transformation.

4. Piecewise surface interpolation

Interpolation of a triangular mesh by piecewise polynomial patches has been studied extensively. Walton and Meek [32] and Hahmann and Bonneau [33] proposed to interpolate meshes by G^1 smooth patches. Vlachos et al. [34] used G^0 surface patches and continuous normal patches for visually smooth rendering of surfaces. However, these methods may not be applicable for our purpose as they usually need a lot of computational costs or suffer low approximation accuracy for meshes sampled on a known surface. For an accurate and efficient curvature tensor estimation purpose, we interpolate surfaces to a triangular mesh satisfying the following requirements.

(a) The normal curvatures of any interpolating surface at mesh vertices should be as accurate as possible.
(b) The interpolating surfaces should not deviate from the triangular mesh too much when the triangle vertices lie on some sharp edges.
(c) Each surface patch should be constructed independently and explicitly just based on vertices and vertex normals of a triangle.

We follow the idea of the curved PN triangles method [34] and construct a cubic triangular Bézier patch for each triangle with given corner normals. Cubic triangular Bézier patches are the lowest degree surface patches that permit inflections along boundary curves and interpolate vertices and normals of a given triangle. While the original curved PN triangles method mainly focuses on contour smoothing of a triangular mesh, it in general does not satisfy requirement (a) stated above. We propose a new cubic Bézier surface interpolation method that satisfies all the three requirements very well. Fig. 2 is an example showing different results of piecewise surface interpolation by the curved PN triangles method and our proposed method.

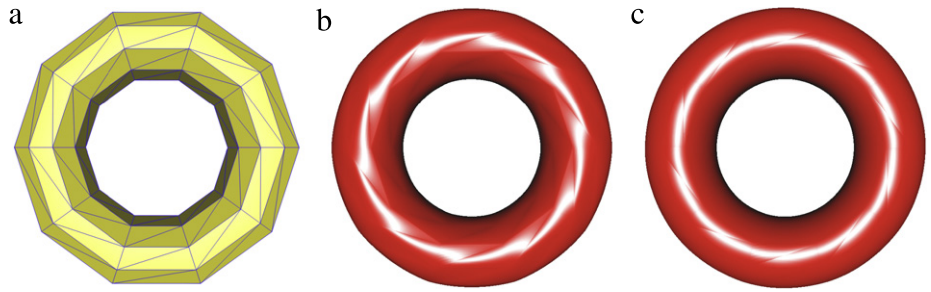


Fig. 2. Interpolation of a mesh by cubic triangular Bézier patches: (a) input triangular mesh; (b) the curved PN triangles method; (c) the proposed method.

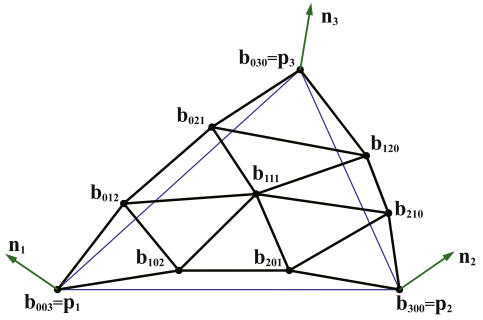


Fig. 3. The control net of a cubic triangular Bézier patch.

A cubic triangular Bézier patch is represented by

$$X(u,v)=\sum_{i+j+k=3}\mathbf{b}_{ijk}\frac{3!}{i!j!k!}u^iv^j(1-u-v)^k \tag{13}$$

where $u\geq 0$, $v\geq 0$, $1-u-v\geq 0$, and $\mathbf{b}_{ijk}$ are the control points (cf. [35]). The control points are connected to form a control net, as illustrated in Fig. 3. Similar to the curved PN triangles method, we classify the control points into three groups:

1. *vertex control points:* $\mathbf{b}_{003}$, $\mathbf{b}_{300}$, $\mathbf{b}_{030}$;
2. *boundary control points:* $\mathbf{b}_{120}$, $\mathbf{b}_{210}$, $\mathbf{b}_{012}$, $\mathbf{b}_{021}$, $\mathbf{b}_{102}$, $\mathbf{b}_{201}$;
3. *center control point:* $\mathbf{b}_{111}$.

Given an input triangle $\triangle\mathbf{p}_1\mathbf{p}_2\mathbf{p}_3$ together with the unit normals $\mathbf{n}_1$, $\mathbf{n}_2$ and $\mathbf{n}_3$ at three respective corner vertices, we determine the vertex control points first, followed by the boundary control points, and finally the center control point.

• *Vertex control points.* Based on the properties of triangular Bézier patches, the three vertex control points are obtained immediately:

$$\mathbf{b}_{003}=\mathbf{p}_1,\qquad \mathbf{b}_{300}=\mathbf{p}_2,\qquad \mathbf{b}_{030}=\mathbf{p}_3.$$

• *Boundary control points.* Let us consider boundary control points $\mathbf{b}_{102}$ and $\mathbf{b}_{201}$ corresponding to edge $\mathbf{p}_1\mathbf{p}_2$. The other boundary control points can be dealt with similarly. The four control points $\mathbf{b}_{003}(=\mathbf{p}_1)$, $\mathbf{b}_{102}$, $\mathbf{b}_{201}$, and $\mathbf{b}_{300}(=\mathbf{p}_2)$ define a cubic Bézier curve that is one boundary of the triangular Bézier patch $X(u,v)$. Since we require that a triangular patch be constructed independently from a triangle and two adjacent triangular patches share a common boundary curve, points $\mathbf{b}_{102}$ and $\mathbf{b}_{201}$ should be constructed only based on $\mathbf{p}_1$, $\mathbf{n}_1$, and $\mathbf{p}_2$, $\mathbf{n}_2$. Thus the problem now becomes a point–normal interpolation problem [36]. That is, given two points $\mathbf{p}_1$, $\mathbf{p}_2$ and two unit normal vectors $\mathbf{n}_1$, $\mathbf{n}_2$ at these two points, we want to find a spatial cubic Bézier curve that interpolates $\mathbf{p}_1$, $\mathbf{p}_2$ and is also tangential to their tangent planes at the respective points.

Obviously, point $\mathbf{b}_{102}$ should be on the tangent plane of the triangular patch at $\mathbf{p}_1$ with normal $\mathbf{n}_1$, and point $\mathbf{b}_{201}$ should be on the tangent plane at $\mathbf{p}_2$ with normal $\mathbf{n}_2$. It is thus suggested to compute

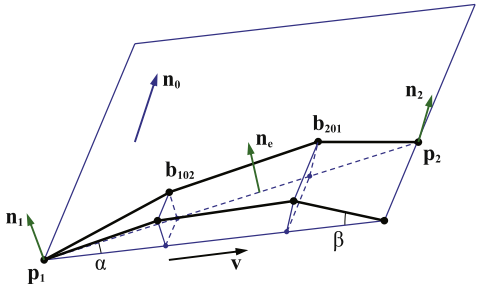


Fig. 4. Computing the control points for a boundary curve.

$\mathbf{b}_{102}$ and $\mathbf{b}_{201}$ by finding two "good" locations on edge $\mathbf{p}_1\mathbf{p}_2$ and projecting them onto the two tangent planes, respectively. Specifically, the two locations are represented by $(1-s_1)\mathbf{p}_1+s_1\mathbf{p}_2$ and $s_2\mathbf{p}_1+(1-s_2)\mathbf{p}_2$ with some parameters s_1 and s_2, and then we have

$$\begin{aligned}\mathbf{b}_{102}&=(1-s_1)\mathbf{p}_1+s_1\mathbf{p}_2-\omega_1\mathbf{n}_1\\ \mathbf{b}_{201}&=s_2\mathbf{p}_1+(1-s_2)\mathbf{p}_2-\omega_2\mathbf{n}_2\end{aligned} \tag{14}$$

where $\omega_1=s_1(\mathbf{p}_2-\mathbf{p}_1)\cdot\mathbf{n}_1$ and $\omega_2=s_2(\mathbf{p}_1-\mathbf{p}_2)\cdot\mathbf{n}_2$. In the curved PN triangles method [34], both s_1 and s_2 were set to $\frac{1}{3}$, but this setting cannot make the constructed curve reproduce the curvature of the circular arc even if the boundary data can define a circular arc. In the following we discuss how to appropriately choose s_1 and s_2 by taking consideration of the inflection case of the curve or the circular arc precision at the ends.

For two points $\mathbf{p}_1$ and $\mathbf{p}_2$ with respective unit normal vectors $\mathbf{n}_1$ and $\mathbf{n}_2$, it is always possible to find a cylinder (not necessarily circular cylinder) that meets the interpolation constraints. In fact, a cylinder can be formed by all the straight lines going through points on $X(u,0)$ along a generatrix direction $\mathbf{n}_0$. The generatrix direction $\mathbf{n}_0$ can be computed by $\mathbf{n}_0=\frac{\mathbf{n}_1\times\mathbf{n}_2}{\|\mathbf{n}_1\times\mathbf{n}_2\|}$ when $\mathbf{n}_1\times\mathbf{n}_2\neq 0$ or $\mathbf{n}_0=\frac{\mathbf{n}_1\times(\mathbf{p}_2-\mathbf{p}_1)}{\|\mathbf{n}_1\times(\mathbf{p}_2-\mathbf{p}_1)\|}$ when $\mathbf{n}_1\times\mathbf{n}_2=0$. It can be verified that the normal vector of the cylinder at point $\mathbf{p}_1$ is $\mathbf{n}_1$. Since both of the interpolating surface $X(u,v)$ and the cylinder pass through curve $X(u,0)$ and have the same normal vector at $\mathbf{p}_1$, the normal curvatures of these two surfaces along the tangent direction of $X(u,0)$ at $\mathbf{p}_1$ are the same based on Meusnier's theorem. We determine s_1 and s_2 based on the shape or precision of a directrix which lies in a plane with normal $\mathbf{n}_0$.

Let π_0 be the plane that passes through point $\mathbf{p}_1$ with normal $\mathbf{n}_0$. After projecting vector $\mathbf{p}_2-\mathbf{p}_1$ onto plane π_0, we obtain a unit vector $\mathbf{v}=\frac{\mathbf{p}_2-\mathbf{p}_1-v\mathbf{n}_0}{\|\mathbf{p}_2-\mathbf{p}_1-v\mathbf{n}_0\|}$, where $v=(\mathbf{p}_2-\mathbf{p}_1)\cdot\mathbf{n}_0$. With reference to Fig. 4, the angle between $\mathbf{v}$ and the tangent plane at $\mathbf{p}_1$ with normal $\mathbf{n}_1$ is denoted by α and the angle between $-\mathbf{v}$ and the tangent plane at $\mathbf{p}_2$ with normal $\mathbf{n}_2$ by β. The signs of α and β reflect whether the edge $\mathbf{p}_1\mathbf{p}_2$ is lying below or above the tangent plane at $\mathbf{p}_1$ and the tangent plane at $\mathbf{p}_2$. For example, if edge $\mathbf{p}_1\mathbf{p}_2$ lies below the tangent plane at $\mathbf{p}_1$ with normal $\mathbf{n}_1$, we have $\alpha>0$; otherwise, $\alpha<0$. The angles α and β are computed by

$$\sin\alpha=-\mathbf{v}\cdot\mathbf{n}_1,\qquad \sin\beta=\mathbf{v}\cdot\mathbf{n}_2.$$

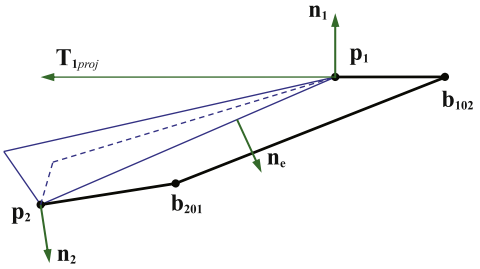


Fig. 5. Computing the boundary control points for an interpolating Bézier patch when a mesh vertex lies on a sharp corner.

If $\sin\alpha \sin\beta < 0$, the projection of a cubic Bézier curve meeting the constraints will have inflections. If the projection curve is inflectional, we choose parameters $s_1 = s_2 = \frac{1}{3}$; otherwise, the parameters are chosen to achieve local circular precision at ends. Let $\eta = \min\{|\alpha+\beta|/2, \pi/3\}$; we choose $s_1 = s_2 = s_0$, where

$$s_0 = \frac{1}{3}\left(\sqrt{1+\frac{3}{\cos^2\eta}} - 1\right),$$

when $\sin\alpha \sin\beta \geq 0$ holds. We assume $|\alpha+\beta|/2 \leq \pi/3$ for general edges and it is guaranteed that $s_0 < 1$ when $\eta \leq \pi/3$. If α and β are equal and be less than $\frac{\pi}{3}$, the curvatures at two ends of the projection curve are equal to the curvature of a circular arc that interpolates the two ends and end tangents of the projection curve. If α or β approaches $\frac{\pi}{2}$, vertex $\mathbf{p}_1$ or $\mathbf{p}_2$ is probably lying on a sharp edge or corner, and a bounded value for s_1 or s_2 can help to generate an interpolating surface close to the mesh. It also yields large values of curvature at the vertices lying on sharp edges or corners.

We also observe that it might be more appropriate to set s_1 and/or s_2 to be negative sometimes. From Eq. (14), we can find that the tangent vector of the cubic Bézier curve defined by $\mathbf{b}_{003}$, $\mathbf{b}_{102}$, $\mathbf{b}_{201}$ and $\mathbf{b}_{300}$ at $\mathbf{p}_1$ is $3s_1\mathbf{T}_{1\text{proj}}$ with $\mathbf{T}_{1\text{proj}} = \mathbf{p}_2 - \mathbf{p}_1 - [(\mathbf{p}_2 - \mathbf{p}_1)\cdot\mathbf{n}_1]\mathbf{n}_1$. Let $\mathbf{n}_{f_0}$ and $\mathbf{n}_{f_1}$ be the unit normal vectors of the two triangles sharing edge $\mathbf{p}_1\mathbf{p}_2$. We define the normal vector of the edge to be

$$\mathbf{n}_e = \frac{\mathbf{n}_{f_0} + \mathbf{n}_{f_1}}{\|\mathbf{n}_{f_0} + \mathbf{n}_{f_1}\|}.$$

If the angle between $\mathbf{n}_e$ and $\mathbf{n}_1$ is obtuse (i.e., $\mathbf{n}_1 \cdot \mathbf{n}_e < 0$), vertex $\mathbf{p}_1$ is probably lying on a sharp corner, and the projections of neighboring triangles onto the tangent plane passing through point $\mathbf{p}_1$ may not surround $\mathbf{p}_1$; see, for example Fig. 5. To guarantee that the projections of the interpolating Bézier patches surround the vertex $\mathbf{p}_1$, the tangent vector of the Bézier curve corresponding to edge $\mathbf{p}_1\mathbf{p}_2$ at $\mathbf{p}_1$ should have opposite direction as $\mathbf{T}_{1\text{proj}}$ when $\mathbf{n}_1 \cdot \mathbf{n}_e < 0$, and thus s_1 should be chosen a negative number (for example, $s_1 = -\frac{1}{3}$). The sign of s_2 can be determined similarly.

Combining all the above discussions, we arrive at our setting scheme for parameters s_i $(i = 1, 2)$:

$$s_i = \begin{cases} s_0, & \text{if } (\mathbf{n}_i \cdot \mathbf{n}_e \geq 0)\&(\sin\alpha \sin\beta \geq 0) \\ +\frac{1}{3}, & \text{if } (\mathbf{n}_i \cdot \mathbf{n}_e \geq 0)\&(\sin\alpha \sin\beta < 0) \\ -\frac{1}{3}, & \text{if } (\mathbf{n}_i \cdot \mathbf{n}_e < 0). \end{cases} \tag{15}$$

• *Center control point.* For each boundary curve of a triangular patch, we specify a vector as an approximate normal of the patch at the midpoint of the curve. Similar to the curved PN triangles method, to capture possible inflection, we compute the vector that is the average of the two end-normals reflected across the plane perpendicular to the edge. While the curved PN triangles method uses this vector as the middle control point for the quadratic normal map, we use the vector directly as an approximate normal, which is shown to work better in our experiments. If we let H_1, H_2 and H_3 denote such vectors for the three boundary curves of the triangular Bézier patch, they can be computed [34]:

$$\begin{aligned} H_1 &= \mathbf{h}_1/\|\mathbf{h}_1\|, \qquad \mathbf{h}_1 = \mathbf{n}_2 + \mathbf{n}_3 - \gamma_{23}(\mathbf{p}_3 - \mathbf{p}_2) \\ H_2 &= \mathbf{h}_2/\|\mathbf{h}_2\|, \qquad \mathbf{h}_2 = \mathbf{n}_3 + \mathbf{n}_1 - \gamma_{31}(\mathbf{p}_1 - \mathbf{p}_3) \\ H_3 &= \mathbf{h}_3/\|\mathbf{h}_3\|, \qquad \mathbf{h}_3 = \mathbf{n}_1 + \mathbf{n}_2 - \gamma_{12}(\mathbf{p}_2 - \mathbf{p}_1) \end{aligned}$$

where $\gamma_{ij} = \frac{2(\mathbf{p}_j - \mathbf{p}_i)\cdot(\mathbf{n}_i + \mathbf{n}_j)}{(\mathbf{p}_j - \mathbf{p}_i)\cdot(\mathbf{p}_j - \mathbf{p}_i)}$. We note that normal vectors at edges can also be computed by the method presented in [37], which can yield similar results.

We determine the center control point $\mathbf{b}_{111}$ heuristically by assuming

$$\mathbf{b}_{111} = B + h\mathbf{n}_t \tag{16}$$

where $B = \frac{1}{6}(\mathbf{b}_{012} + \mathbf{b}_{021} + \mathbf{b}_{120} + \mathbf{b}_{210} + \mathbf{b}_{201} + \mathbf{b}_{102})$ and $\mathbf{n}_t$ is the unit normal of triangle $\triangle\mathbf{p}_1\mathbf{p}_2\mathbf{p}_3$. Let D_1, D_2 and D_3 be the cross derivatives of the Bézier surface at the midpoints of the three triangle edges in directions perpendicular to the triangle edges. If $D_1 \cdot H_1 = D_2 \cdot H_2 = D_3 \cdot H_3 = 0$ hold for every interpolating Bézier patch, any two adjacent triangular Bézier patches have the same tangent plane at the midpoint of their common boundary. Since there is only one unknown scalar parameter h within the three equations, we compute h by solving equation $D_1 \cdot H_1 + D_2 \cdot H_2 + D_3 \cdot H_3 = 0$. With some calculations, we obtain

$$h = \frac{-2(E_1 \cdot H_1 + E_2 \cdot H_2 + E_3 \cdot H_3)}{\mathbf{n}_t \cdot (H_1 + H_2 + H_3)}$$

where

$$\begin{aligned} E_1 &= \frac{\mathbf{b}_{201} + 2B + \mathbf{b}_{021}}{4} - \frac{\mathbf{b}_{300} + 3\mathbf{b}_{210} + 3\mathbf{b}_{120} + \mathbf{b}_{030}}{8}, \\ E_2 &= \frac{\mathbf{b}_{102} + 2B + \mathbf{b}_{120}}{4} - \frac{\mathbf{b}_{030} + 3\mathbf{b}_{021} + 3\mathbf{b}_{012} + \mathbf{b}_{003}}{8}, \\ E_3 &= \frac{\mathbf{b}_{012} + 2B + \mathbf{b}_{210}}{4} - \frac{\mathbf{b}_{003} + 3\mathbf{b}_{102} + 3\mathbf{b}_{201} + \mathbf{b}_{300}}{8}. \end{aligned}$$

In practice, this solution still makes those interpolating Bézier patches approximately have continuous tangent planes across common boundaries.

5. The algorithm

This section describes our algorithm for curvature tensor estimation on triangular meshes by piecewise surface interpolation. The algorithm estimates the curvature tensors at mesh vertices by the closed form Taubin integral given in Section 3 and the carefully constructed surfaces described in Section 4. This usually gives quite accurate curvature tensors when the sampled points and normals are accurate. However, the estimation may suffer from noise when the mesh vertices or normals are not accurate.

To enhance the robustness of estimation for a mesh containing noise, we propose to compute the curvature tensor at a vertex by incorporating the Taubin integrals at the center of triangular patches incident to the vertex. Specifically, as in Section 3, we assume that $\mathbf{p}$ is the joint vertex of surface patches $X_i(u, v)$ with $\mathbf{p} = X_i(0, 0)$ for $i = 1, 2, \ldots, n$. Let M_p be the integral matrix (11) and $M_{p_i}^{X_i}$ be the integral matrix (12) of surface $X_i(u, v)$ at point $\mathbf{p}_i = X_i(u_i, v_i)$ with $u_i = v_i = 1/3$. We define a weighted sum of Taubin integrals at vertex $\mathbf{p}$ as

$$\widetilde{M}_p = (1-\rho)M_p + \rho\sum_{i=1}^{n}\omega_i M_{p_i}^{X_i}, \tag{17}$$

where $0 \leq \rho \leq 1$ is a parameter specified by users, and ω_i are the weights satisfying $\omega_i \geq 0$ and $\sum_{i=1}^{n}\omega_i = 1$. In general ω_i are chosen to be proportional to the areas of triangles incident to

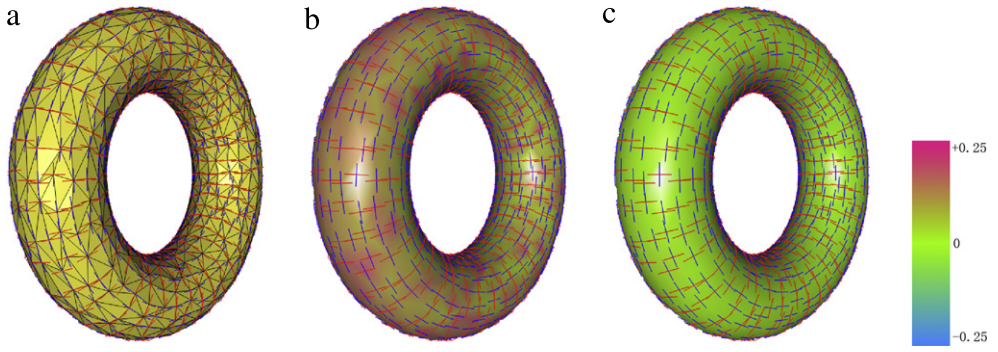


Fig. 6. Curvature tensor estimation using different surface interpolation schemes: (a) the triangular mesh and the exact principal directions; (b) mean curvature error plot and the principal directions by the PN triangles method; (c) mean curvature error plot and the principal directions using the proposed surface interpolation scheme.

the joint vertex. To further improve the accuracy of (17), we follow the technique in [23] to rotate the tangent plane to surface $X_i(u, v)$ through $\mathbf{p}_i$ and compute updated X_u and X_v for matrices R_1, R_2 and R_3 in $M_{p_i}^{X_i}$.

For a triangular mesh consisting of vertex set $\mathcal{V} = \{\mathbf{v}_i\}$, edge set $\mathcal{E} = \{(\mathbf{v}_i, \mathbf{v}_j)\}$ and face set $\mathcal{F} = \{(\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k)\}$, and a user specified parameter ρ, we estimate curvature tensors at mesh vertices as follows:

1. for each vertex $\mathbf{v}_i$ compute the unit normal vector $\mathbf{n}_i$ at the vertex as a weighted sum of normals to the surrounding facets with weights given in [38];
2. for each edge $(\mathbf{v}_i, \mathbf{v}_j)$ with unit normal vectors at two end vertices, construct a cubic curve by Eqs. (14) and (15);
3. for each triangle $(\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k)$ construct a cubic interpolating triangular Bézier patch by copying the control points of three cubic curves interpolating the edges and computing the center control point by Eq. (16);
4. for each vertex $\mathbf{v}_i$
   - if $\rho < 1$, compute the sum of Taubin integrals of surrounding Bézier patches at the vertex using Eqs. (9) and (11);
   - if $\rho > 0$, compute Taubin integrals at centers of surrounding Bézier patches using Eq. (12) and a weighted sum of Taubin integrals using Eq. (17);
   - compute the principal curvatures and principal directions from the Taubin integral using closed form expressions, just as Taubin's approach [25].

Note that the combination of the closed form Taubin integral and the curved PN triangles can also estimate curvature tensors for triangular meshes. Fig. 6(a) shows a triangulated torus model with 1800 randomly sampled points and exact curvature tensors at the points. When we interpolate the torus using PN triangles and compute curvature tensors for every vertex by computing the Taubin integral, the average and maximum errors of mean curvatures are 0.099527 and 0.464881, respectively; see Fig. 6(b) for the plot of mean curvature errors. The mean and maximum deviations of principal directions by the PN triangles method are 0.749599 and 8.909531 degrees, respectively. By our proposed surface interpolation scheme, the average and maximum errors of mean curvatures reduce to 0.008991 and 0.083860 while the mean and maximum deviations of principal directions become 0.221218 and 1.741704 degrees, respectively; see Fig. 6(c) for the plot of mean curvature errors using this new surface interpolation scheme.

6. Experimental results

This section evaluates our proposed algorithm with several models. We first test the accuracy of the proposed curvature tensor estimation method using two models generated from two analytical surfaces. Then we apply the proposed method to a subdivision surface and a denoised mesh to test its use in checking the smoothness of discrete models. Finally we apply the proposed method for visualizing the detail or estimating smooth curvatures of a real scan-reconstructed mesh.

We compare the proposed method with several state-of-the-art methods which use continuous surface patches or discrete techniques for curvature tensor estimation in CAD or graphics. The vertex normal (VN) triangle based method [14] is the latest one that computes curvatures using curved triangles. The cubic fitting algorithm [19] is a representative for analytic fitting methods, which can achieve very high accuracy. The discrete methods which are widely used in graphics include Taubin's method [25], the finite difference method proposed by Rusinkiewicz [23] and the tensor averaging method [24]. They are very competitive in speed for curvature tensor estimation. All these algorithms were implemented using C++ on a PC with Intel(R) Core(TM)2 CPU, T9900@3.06 GHz 3.07 GHz and 4G RAM.

The two analytical surfaces are defined as follows. The first surface is a ring-shaped surface:

$$\begin{cases} x(u, v) = (1.5 + 0.3\cos(v))\cos(u) \\ y(u, v) = (1.5 + 0.3\cos(v))\sin(u) \\ z(u, v) = 0.6\sin(v) \end{cases}$$

where $(u, v) \in [0, 2\pi] \times [0, 2\pi]$. The ring-shaped surface is generated by sweeping an ellipse along a circle. The second surface is a surface patch which was introduced by Goldfeather and Interrante in [19]. The equations of the G&I surface are

$$\begin{cases} x(u, v) = f(u)\cos(v) \\ y(u, v) = f(u)\sin(v) \\ z(u, v) = u + 0.2\sin(2x) + 0.15\cos(3xy) \end{cases}$$

where $f(u) = -2u^4 + 2u^2 + u/6 + 0.3$ and $(u, v) \in [-0.9, 1] \times [0, 2\pi]$. These two surfaces serve as the ground truth. Both are free of umbilics and thus the principal curvature and principal directions can be computed accurately for each point on the surfaces. We tessellate each surface into a low resolution mesh and a high resolution mesh with different numbers of sampled points on the surface. Meanwhile, the exact curvature tensors of the surface at all sampled points are evaluated for comparison. Fig. 7(a) illustrates two meshes tessellated from the ring-shaped surface, and the two meshes have 1800 or 7200 vertices, respectively. Fig. 7(b) illustrates the two meshes of the G&I surface which have 2550 and 10 100 vertices, respectively.

We test the accuracy of our proposed algorithm and other five methods using low resolution tessellated meshes and high resolution tessellated meshes shown in Fig. 7 with exact sampled points and normals. To test the robustness of these methods, we add noise to the four meshes by moving each vertex along its normal direction by a random distance within 2% and 5% of the mean edge length. The vertex normals for the noisy meshes are then computed as the weighted sum of the 1-ring facet normals using the method presented in [38]. The accuracy of the estimated curvature tensors

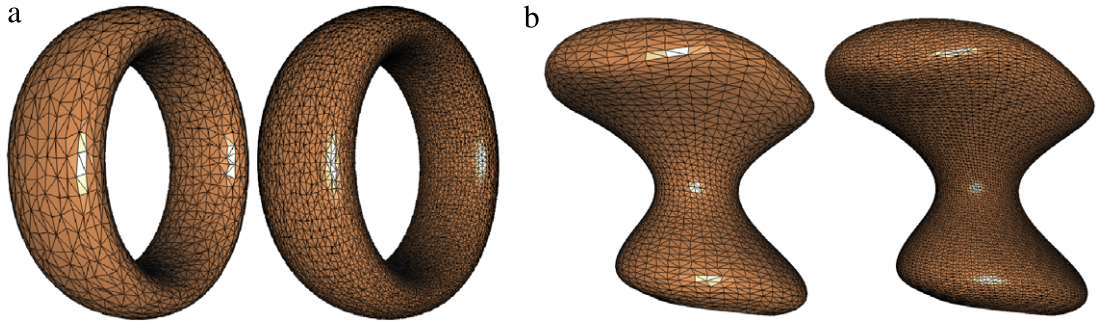


Fig. 7. Surface tessellation with low or high resolutions: (a) tessellation of a ring-shaped surface; (b) tessellation of the G&I surface.

Table 1
RMS errors of the estimated curvatures for the ring model.

Method	Ring (LR) (#vertices=1800)	Ring (HR) (#vertices=7200)	Ring (HR) (2% noise)	Ring (HR) (5% noise)
Proposed ($\rho = 0$)	0.044276	0.017946	1.241357	3.127922
Proposed ($\rho = 1$)	0.159244	0.069801	0.239621	0.536465
VN triangle	0.172309	0.062181	0.245828	0.575565
Cubic fitting	0.135025	0.031862	0.222330	0.371346
Finite difference	0.162506	0.070257	0.165780	0.308891
Tensor averaging	0.759555	0.748065	0.823753	1.163509
Taubin's method	0.997030	0.827438	0.928336	1.307278

Table 2
RMS errors of the estimated curvatures for the G&I surface.

Method	G&I (LR) (#vertices=2550)	G&I (HR) (#vertices=10100)	G&I (HR) (2% noise)	G&I (HR) (5% noise)
Proposed ($\rho = 0$)	0.064131	0.025054	2.327218	6.206422
Proposed ($\rho = 1$)	0.189813	0.080634	0.349591	0.875071
VN triangle	0.224454	0.082247	0.350479	1.067194
Cubic fitting	0.122904	0.029810	0.231335	0.535465
Finite difference	0.178445	0.074829	0.228927	0.513157
Tensor averaging	0.907732	0.901229	1.064728	1.742106
Taubin's method	0.864383	0.882754	1.193384	2.277044

by each method is measured by the root mean square (RMS) error, which is the square root of the average of square errors of all major and minor curvatures against the ground truth.

From Tables 1 and 2 we can see that our proposed method with $\rho = 0$ can achieve the highest accuracy for the two surfaces among all the six methods. Moreover, similar to the cubic fitting method, the finite difference method and the VN triangle method, our method with either $\rho = 0$ or $\rho = 1$ can raise the accuracy when the mesh resolution increases. However, our method with $\rho = 0$ is sensitive to noise. This is not surprising because a surface mesh with noisy vertices or disturbed normals may represent a different underlying surface. An accurate method should then be sensitive to the change of the vertices or the normals of the mesh. On the other hand, the cubic fitting method, the finite difference method and our method with $\rho = 1$ are robust against noise due to the use of the least squares fitting or averaging strategy though the accuracy of the results may not be the highest.

We next show the use of the proposed method for smoothness check of surfaces. We apply the curvature tensor estimation methods to a Doo–Sabin subdivision mesh (see Fig. 8(a)). An initial polygonal mesh is subdivided six times by Doo–Sabin subdivision, and then each face in the refined mesh is split into triangles such that the vertices have random valences. Note that this triangulated mesh is very close to the limit surface of Doo–Sabin subdivision though the vertices of the mesh do not necessarily lie on the limit surface. Thus accurately estimated curvature tensors of the triangulated mesh are assumed to have similar behavior as those of the limit surface. A Doo–Sabin subdivision surface is a generalized biquadratic B-spline surface, and it is composed of a set of parametric surface patches which have tangent plane continuity across the joint edges or the extraordinary points [39]. Fig. 9(a) and (b) display the mean curvatures and the Gaussian curvatures estimated by our

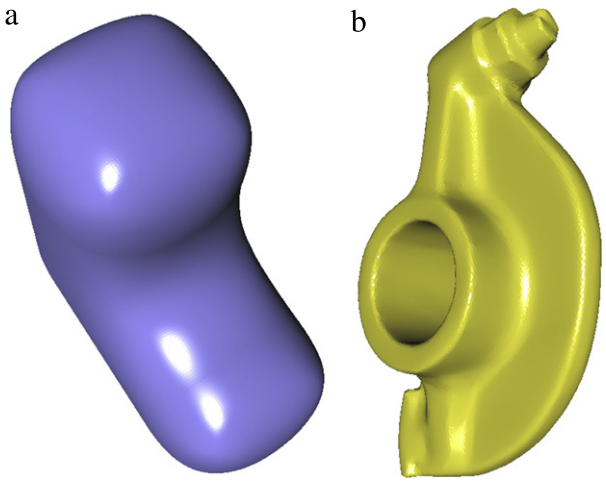


Fig. 8. A subdivision mesh and a denoised mesh: (a) a triangulated mesh by Doo–Sabin subdivision; (b) a smooth mesh by bilateral denoising.

proposed method with $\rho = 0$, respectively. In these two figures, the joint lines and individual surface patches are clearly visible. Fig. 9(h) shows that the Gaussian curvatures estimated by Taubin's method are also sensitive to the boundaries of surface patches. However, the accuracy of Taubin's method is influenced by vertex valences. Fig. 9(d), (e), (f), (g) and (c) show the plots of Gaussian curvatures estimated by the VN triangle method, the cubic fitting method, the finite difference method, the tensor averaging method and our method with $\rho = 1$, where the curvatures around the

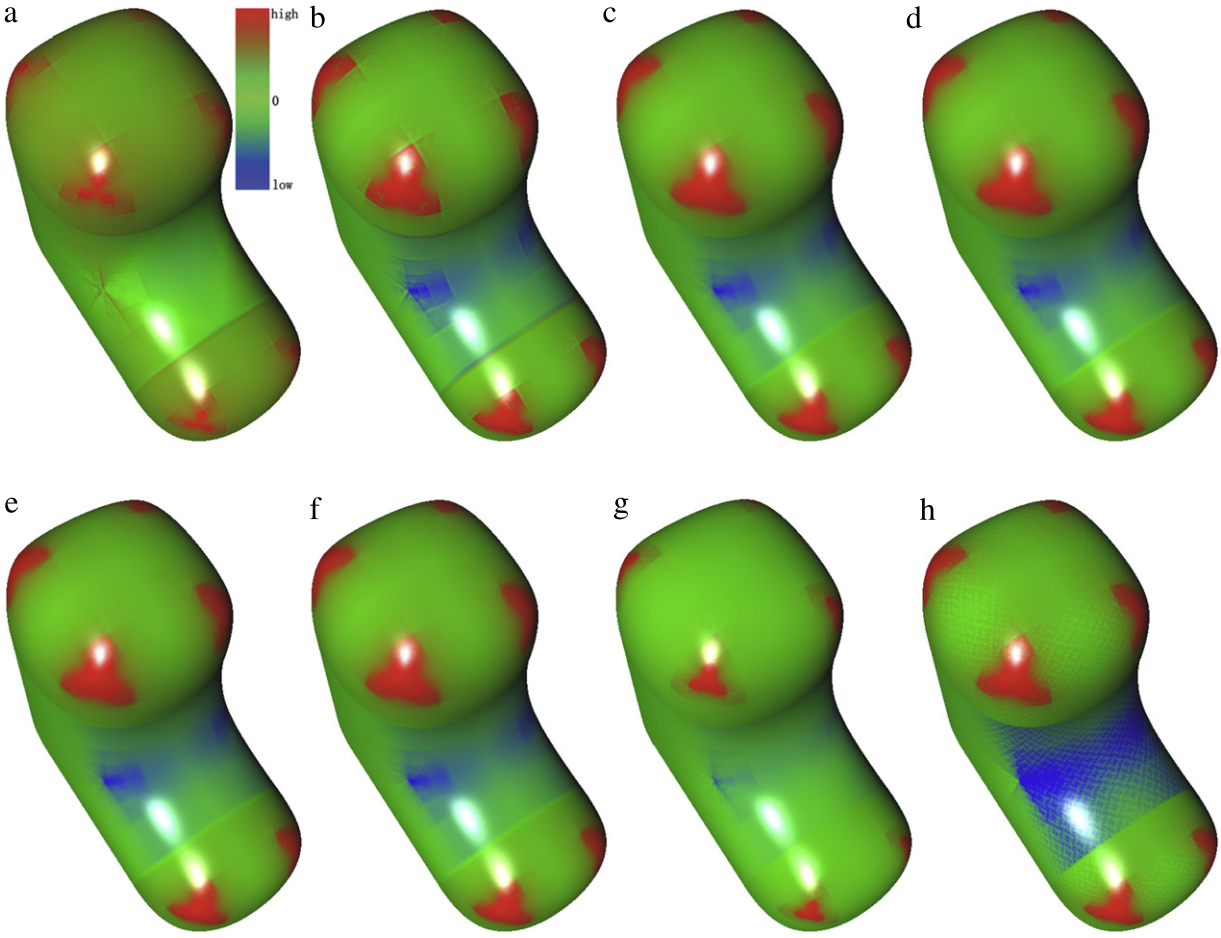


Fig. 9. Curvature estimation for a Doo–Sabin subdivision surface: (a) mean curvature by the proposed method ($\rho = 0$); (b) Gaussian curvature by the proposed method ($\rho = 0$); (c) Gaussian curvature by the proposed method ($\rho = 1$); (d) Gaussian curvature by the VN triangle method; (e) Gaussian curvature by the cubic fitting method; (f) Gaussian curvature by the finite difference approach; (g) Gaussian curvature by the tensor averaging method; (h) Gaussian curvature by Taubin's method. Readers can zoom in the figures to see even clearer details.

patch edges are seen to be blurred due to the process of the least squares fitting or averaging.

Fig. 8(b) is another visually smooth surface mesh obtained by bilateral mesh denoising [40]. To check the smoothness of the denoised mesh, we compute the mean curvature and Gaussian curvature of the mesh by our proposed method with $\rho = 0$. From the curvature plots in Fig. 10(a) and (b) we learn that the mesh is not as smooth as it looks. It still contains some stairs or artifacts even after bilateral denoising. The curvatures estimated by the proposed method with $\rho = 1$, the VN triangle method, the cubic fitting method, the finite difference method all look smooth; see Fig. 10(c)–(f), respectively. These methods have hidden the artifacts of the surface mesh. The plots of the mean curvatures estimated by the tensor averaging approach and Taubin's method are given in Fig. 10(g) and (h), which illustrate some curvature discontinuity of the denoised surface. However, the artifacts are not clear enough due to the limited accuracy of curvature estimation by these methods.

Our proposed curvature tensor estimation algorithm provides a tradeoff between accuracy and robustness, which is controlled by parameter ρ. A small value of ρ yields accurate but noise sensitive curvature tensors while a large value of ρ helps to estimate curvature tensors robustly against noise. Fig. 11(a) illustrates a triangular mesh reconstructed from real scanned data. From the figure we can see that the original surface contains low magnitude noise. The accurate mean curvature plot in Fig. 11(b) corresponding to $\rho = 0$ clearly depicts the noise distribution on the surface. When the parameter value is changed to $\rho = 0.5$ and $\rho = 1.0$, the curvature plots become smoother and smoother; see Fig. 11(c) and (d) for the results. It is also observed that the principal directions become smoother when the value of parameter ρ increases.

Finally, we compare the computational efficiency of our proposed algorithm and all other methods by reporting the running time for all the test models in Table 3. From the table, it can be seen that Taubin's method is the fastest and the cubic fitting algorithm is the slowest. Our proposed method has similar time cost as the finite difference algorithm. Due to its high accuracy and considerable efficiency, our proposed method can be used for curvature tensor computation for triangular meshes that have a large number of vertices.

7. Conclusions and discussion

This paper has derived the closed form Taubin integral for piecewise smooth surfaces or a single parametric surface, which leads to a new way for curvature tensor estimation for triangular meshes. The paper has also presented an improved local surface interpolation scheme for triangular meshes. Compared to other surface fitting methods, the proposed surface interpolation scheme does not need to solve any linear systems. Our proposed curvature tensor estimation method usually generates more accurate results for general surface meshes than the state-of-the-art methods. Also by using weighted sums of Taubin integrals, the proposed algorithm can robustly estimate curvature tensors for noisy meshes. The tradeoff between accuracy and robustness of curvature tensor estimation is controlled using a single parameter.

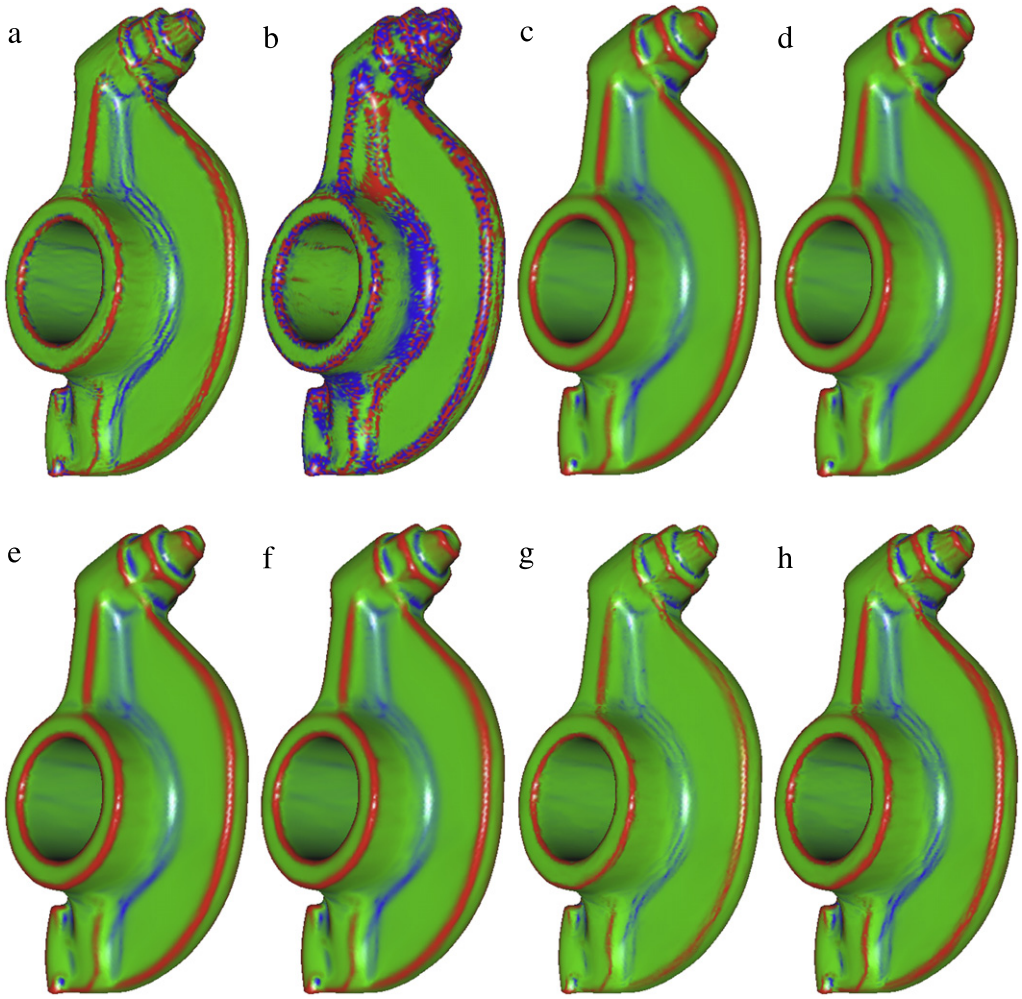


Fig. 10. Curvature estimation for a denoised surface: (a) mean curvature by the proposed method ($\rho = 0$); (b) Gaussian curvature by the proposed method ($\rho = 0$); (c) mean curvature by the proposed method ($\rho = 1$); (d) mean curvature by the VN triangle method; (e) mean curvature by the cubic fitting method; (f) mean curvature by the finite difference approach; (g) mean curvature by the tensor averaging method; (h) mean curvature by Taubin's method.

Table 3
Time costs for the test models (in seconds).

Model	#vertex	Taubin's method	Cubic fitting	VN triangle	Finite difference	Tensor averaging	Proposed ($\rho = 0$)	Proposed ($\rho = 1$)
Ring	7 200	0.047	0.765	0.406	0.140	0.109	0.171	0.296
goldfeather	10 100	0.063	1.030	0.562	0.203	0.156	0.234	0.390
L-shape	57 344	0.374	5.538	3.261	1.170	0.951	1.420	2.512
Rocker arm	40 177	0.250	3.916	2.278	0.811	0.671	0.983	1.763
Armadillo	165 954	1.139	15.491	8.721	3.791	3.011	4.056	6.739

Accurate curvature tensor estimation can be applied when the input vertices or normals are accurate or have low magnitude noise. Our proposed method can be used to visualize the joint lines between surface patches or the distribution of noise. Although the paper only discusses triangular meshes, the extension of the proposed method to general polygonal meshes is straightforward. One can replace piecewise triangular Bézier surface interpolation to other types of surface interpolation, and the curvature tensors at selected points can be computed by the closed form Taubin integral and the consequent eigen-analysis of the integral matrix.

Acknowledgments

We thank the editor and anonymous reviewers for their helpful comments. This work was supported by NNSF of China grants (60970077, 11290142, 61272300) and Multi-plAtform Game Innovation Centre (MAGIC) in Nanyang Technological University. MAGIC is funded by the Interactive Digital Media Programme Office (IDMPO) hosted by the Media Development Authority of Singapore.

Appendix. Calculations of C_1^i, C_2^i and C_3^i

To compute C_1^i, we use substitution $t = \tan\theta$ and obtain

$$\begin{aligned} C_1^i &= D_i \int_0^{\frac{\pi}{2}} \frac{L\cos^2\theta + M\sin 2\theta + N\sin^2\theta}{(E\cos^2\theta + F\sin 2\theta + G\sin^2\theta)^3} \cos^2\theta d\theta \\ &= D_i \int_0^{\frac{\pi}{2}} \frac{L + 2M\tan\theta + N\tan^2\theta}{(E + 2F\tan\theta + G\tan^2\theta)^3} d\tan\theta \\ &= D_i \int_0^{\infty} \frac{L + 2Mt + Nt^2}{(E + 2Ft + Gt^2)^3} dt. \end{aligned}$$

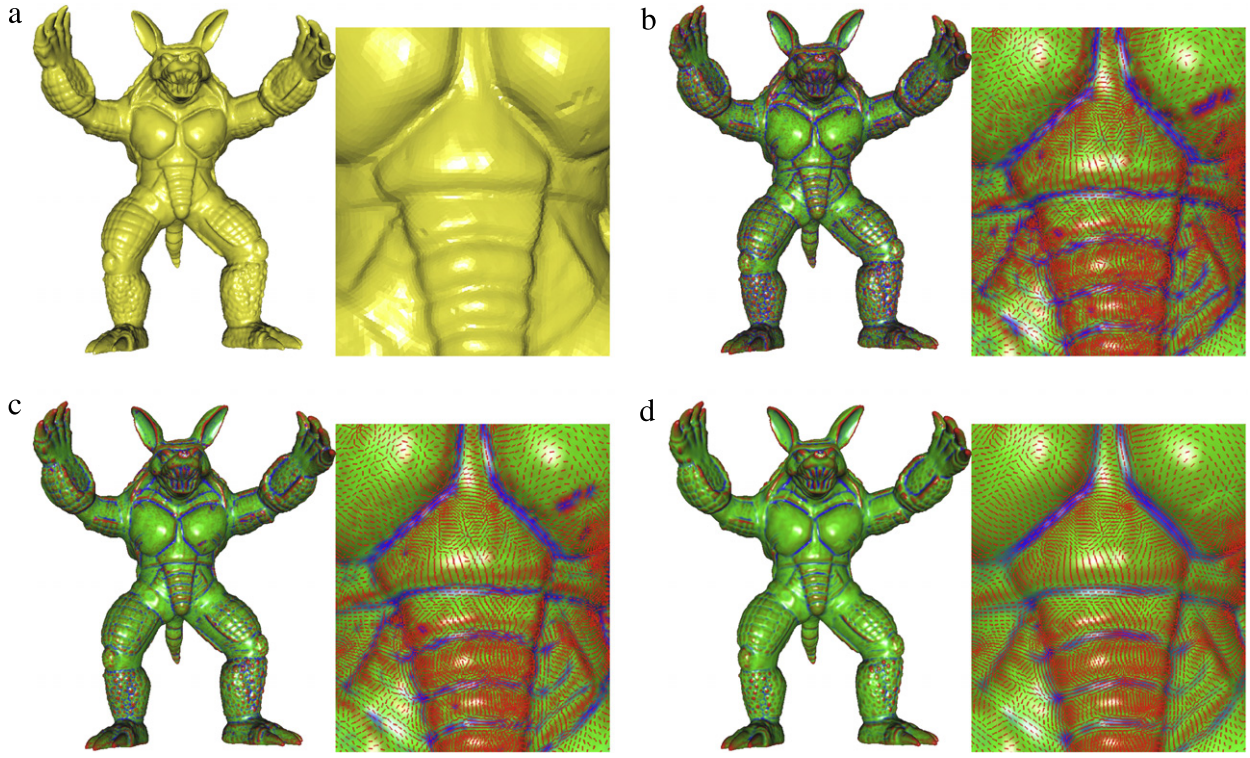


Fig. 11. Plots of mean curvatures and major principal directions of a scan-reconstructed surface by the proposed method: (a) the original surface; (b) $\rho = 0$; (c) $\rho = 0.5$; (d) $\rho = 1$.

We then rewrite $E + 2Ft + Gt^2 = \frac{EG-F^2}{G}[1 + (\frac{G}{\sqrt{EG-F^2}}t + \frac{F}{\sqrt{EG-F^2}})^2] = \frac{G}{g^2}[1+(gt+f)^2]$. By using substitution $\tau = gt+f$, C_1^i can be further expressed as

$$\begin{aligned} C_1^i &= \frac{D_i g^3}{G^3}\int_f^\infty \frac{Lg^2 + 2Mg(\tau - f) + N(\tau - f)^2}{(1+\tau^2)^3}d\tau \\ &= \frac{D_i g^3}{G^3}[Lg^2 - 2Mfg + N(f^2-1)]\int_f^\infty \frac{1}{(1+\tau^2)^3}d\tau \\ &\quad + \frac{D_i g^3}{G^3}(2Mg - 2Nf)\int_f^\infty \frac{\tau}{(1+\tau^2)^3}d\tau \\ &\quad + \frac{D_i g^3}{G^3}N\int_f^\infty \frac{1}{(1+\tau^2)^2}d\tau. \end{aligned}$$

Let $I_n = \int_f^\infty \frac{1}{(1+\tau^2)^n}d\tau$ and $J_n = \int_f^\infty \frac{\tau}{(1+\tau^2)^n}d\tau$. I_n and J_n can be computed recursively or explicitly: $I_1 = \frac{\pi}{2} - \arctan f$, $I_{n+1} = \frac{2n-1}{2n}I_n - \frac{f}{2n(1+f^2)^n}$ for $n > 0$, and $J_n = \frac{1}{2(n-1)}\frac{1}{(1+f^2)^{n-1}}$ for $n > 1$. Thus after simplification, the formula for C_1^i given in Section 3 can be obtained.

In a similar way, the formula for C_2^i can be derived with the substitution $t = \tan\theta$.

For C_3^i, we replace θ by $\frac{\pi}{2} - \varphi$. Then

$$\begin{aligned} C_3^i &= D_i\int_0^{\frac{\pi}{2}} \frac{L\cos^2\theta + M\sin 2\theta + N\sin^2\theta}{(E\cos^2\theta + F\sin 2\theta + G\sin^2\theta)^3}\sin^2\theta d\theta \\ &= D_i\int_0^{\frac{\pi}{2}} \frac{N\cos^2\varphi + M\sin 2\varphi + L\sin^2\varphi}{(G\cos^2\varphi + F\sin 2\varphi + E\sin^2\varphi)^3}\cos^2\varphi d\varphi. \end{aligned}$$

Therefore the formula for C_3^i can be obtained from the one for C_1^i by interchanging L and N, E and G.

References

[1] Eigensatz M, Sumner RW, Pauly M. Curvature-domain shape processing. Computer Graphics Forum 2008;27(2):241–50 (Eurographics Proceedings).
[2] Lavoue G, Dupont F, Baskurt A. A new cad mesh segmentation method based on curvature tensor analysis. Computer Aided Design 2005;37:975–87.
[3] Samson P, Mallet J-L. Curvature analysis of triangulated surfaces in structural geology. Journal of Mathematical Geology 1997;29(3):391–412.
[4] Moorhead R, Guan Y, Hagen H, Bötger S, Kotava N, Wagner C. Illustrative visualization: interrogating triangulated surfaces. Computing 2009;86:131–50.
[5] Alliez P, Cohen-Steiner D, Devillers O, Lévy B, Desbrun M. Anisotropic polygonal remeshing. ACM Transactions on Graphics 2003;22(3):485–93.
[6] Hertzmann A, Zorin D. Illustrating smooth surfaces. In: ACM SIGGRAPH'00. 2000. p. 517–26.
[7] Kim Y, Yu JY, Yu X, Lee S. Line-art illustration of dynamic and specular surfaces. ACM Transactions on Graphics 2008;27(5):156.
[8] Gorla G, Interrante V, Sapiro G. Texture synthesis for 3D shape representation. IEEE Transactions on Visualization and Computer Graphics 2003;9(4):512–24.
[9] 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 2007;107:139–59.
[10] Sundaram P, Zomorodian A, Beaulieu C, Napel S. Colon polyp detection using smoothed shape operators: preliminary results. Medical Image Analysis 2008;12:99–119.
[11] Petitjean S. A survey of methods for recovering quadrics in triangle meshes. ACM Computing Surveys 2002;34(2):211–62.
[12] Gatzke T, Grimm C. Estimating curvature on triangular meshes. International Journal of Shape Modeling 2006;12(1):1–28.
[13] Wang D, Clark B, Jiao X. An analysis and comparison of parameterization-based computation of differential quantities for discrete surfaces. Computer Aided Geometric Design 2009;26:510–27.
[14] Mao Z, Cao G, Ma Y, Lee K. Curvature estimation for meshes based on vertex normal triangles. Computer-Aided Design 2011;43(12):1561–6.
[15] Chen X, Schmitt F. Intrinsic surface properties from surface triangulation. In: Proceedings of the European conference on computer version. 1992. p. 739–43.
[16] Meyer M, Desbrun M, Schröder P, Barr AH. Discrete differential-geometry operators for triangulated 2-manifolds. In: Hege H-C, Polthier K, editors. Visualization and Mathematics III. Heidelberg: Springer-Verlag; 2003. p. 35–57.
[17] Adam G, Tang X. A sampling framework for accurate curvature estimation in discrete surfaces. IEEE Transactions on Visualization and Computer Graphics 2005;11(5):573–83.
[18] Batagelo HC, Wu ST. Estimating curvatures and their derivatives on meshes of arbitrary topology from sampling directions. The Visual Computer 2007;23:803–12.
[19] Goldfeather J, Interrante V. A novel cubic-order algorithm for approximating principal direction vectors. ACM Transactions on Graphics 2004;23(1):45–63.

[20] Theisel H, Rössl C, Zayer R, Seidel H-P. Normal based estimation of the curvature tensor for triangulated meshes. In: Cohen-Or D, Ko H-S, Terzopoulos D, Warren J, editors. 12th Pacific conference on computer graphics and applications. 2004. p. 288–97.
[21] Cazals F, Pouget M. Estimating differential quantities using polynomial fitting of osculating jets. Computer Aided Geometric Design 2005;22:121–46.
[22] Razdan A, Bae M. Curvature estimation scheme for triangle meshes using biquadratic Bézier patches. Computer Aided Design 2005;37:1481–91.
[23] Rusinkiewicz S. Estimating curvatures and their derivatives on triangle meshes. In: Symposium on 3D data processing, visualization, and transmission. Sep. 2004.
[24] Cohen-Steiner D, Morvan JM. Restricted delaunay triangulations and normal cycle. In: Proceedings of the nineteenth annual symposium on computational geometry. ACM Press; 2003. p. 312–21.
[25] Taubin G. Estimating the tensor of curvature of a surface from a polyhedral approximation. In: Proceedings of 5th international conference on computer vision. 1995. p. 902–07.
[26] Hameiri E, Shimshoni I. Estimating the principle curvatures and the darboux frame from real 3-D range data. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 2003;33(4):626–37.
[27] Tong WS, Tang C-K. Robust estimation of adaptive tensors of curvature by tensor voting. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005;27(3):434–49.
[28] Langer T, Belyaev A, Seidel HP. Exact and interpolatory quadratures for curvature tensor estimation. Computer Aided Geometric Design 2007;24:443–63.
[29] Pottmann H, Wallner J, Yang Y-L, Lai Y-K, Hu S-M. Principal curvatures from the integral invariant viewpoint. Computer Aided Geometric Design 2007;24:428–42.
[30] Chazal F, Cohen-Steiner D, Lieutier A, Thibert B. Stability of curvature measures. In: Proceedings of the Eurographics symposium on geometry processing. 2009.
[31] DoCarmo M. Differential geometry of curves and surfaces. Prentice Hall; 1976.
[32] Walton DJ, Meek DS. A triangular G^1 patch from boundary curves. Computer Aided Design 1996;28(2):113–23.
[33] Hahmann S, Bonneau G-P. Polynomial surfaces interpolating arbitrary triangulations. IEEE Transactions on Visualization and Computer Graphics 2003;9(1):99–109.
[34] Vlachos A, Peters J, Boyd C, Mitchell J. Curved PN triangles. In: Proceedings of the 2001 symposium on interactive 3D graphics. 2001. p. 159–66.
[35] Farin G. Triangular Bernstein–Bézier patches. Computer Aided Geometric Design 1986;3(2):83–127.
[36] Stiller J. Point-normal interpolation schemes reproducing spheres, cylinders and cones. Computer Aided Geometric Design 2007;24(5):286–301.
[37] Yang X, Zheng J. Shape aware normal interpolation for curved surface shading from polyhedral approximation. The Visual Computer 2013;29(3):189–201.
[38] Max N. Weights for computing vertex normals from facet normals. Jounrnal of Graphics Tools 1999;4(2):1–6.
[39] Doo D, Sabin M. Behaviour of recursive division surfaces near extraordinary points. Computer Aided Design 1978;10(6):356–60.
[40] Fleishman S, Drori I, Cohon-Or D. Bilateral mesh denoising. ACM Transactions on Graphics 2003;22(3):950–3.