Yuji Kobashi · Junya Suzuki · Han Kyul Joo · Takashi Maekawa

Reuse of B-spline-Based Shape Interrogation Tools for Triangular Mesh Models

Abstract In many engineering applications, a smooth surface is often approximated by a mesh of polygons. In a number of downstream applications, it is frequently necessary to estimate the differential invariant properties of the underlying smooth surfaces of the mesh. Such applications include first-order surface interrogation methods that entail the use of isophotes, reflection lines, and highlight lines, and second-order surface interrogation methods such as the computation of geodesics, geodesic offsets, lines of curvature, and detection of umbilics. However, we are not able to directly apply these tools that were developed for B-spline surfaces to tessellated surfaces. This article describes a unifying technique that enables us to use the shape interrogation tools developed for B-spline surface on objects represented by triangular meshes. First, the region of interest of a given triangular mesh is transformed into a graph function (z=h(x,y)) so that we can treat the triangular domain within the rectangular domain. Each triangular mesh is then converted into a cubic graph triangular Bézier patch so that the positions as well as the derivatives of the surface can be evaluated for any given point (x,y) in the domain. A number of illustrative examples are given that show the effectiveness of our algorithm.

Keywords Shape interrogation · B-spline surface · Triangular mesh · PN triangles

1 Introduction

In reverse engineering, densely scanned point clouds are usually tessellated into a triangular mesh. Constant density surfaces generated from medical images, which are produced via computed tomography (CT) or magnetic resonance (MR), are represented by triangular meshes [14]. Models in animation and games are often represented by a mesh of polygons that portray a high degree of visual quality. The differential invariant properties of the underlying smooth surfaces of mesh models are often required in downstream applications. Such applications include shape interrogation, shape matching, rendering, feature detection, and additive manufacturing. Although in the last three decades, many tools have been developed for B-spline surfaces to deal with these applications, we are not able to directly apply these tools to tessellated surfaces due to the following two reasons. The first reason is that triangular meshes are flat and possess only C^0 continuity; thus, classical differential geometry cannot be applied. Furthermore, the coordinates of a point of a B-spline surface patch are expressed as functions of the parameters u and v in a closed rectangle, while those of a triangular mesh are expressed in a triangular domain.

On the other hand, since these tools developed for B-spline surfaces cannot be applied to mesh models, many techniques for the estimation of the geometric properties of a triangular mesh based on discrete differential geometry have been developed. There is no consensus, however, on how to achieve accurate estimations of simple surface attributes such as normal vectors and curvatures [20]. Thus, the making of robust estimates of normal vectors and curvature tensor still remains an area of active research [9]. Most existing techniques estimate the surface attributes at the vertices of the input polygon, an approach that may not be appropriate for our purposes, since the computation of geodesics and lines of curvature requires the estimation of such attributes inside the triangular domain. One can, of course, linearly interpolate the normal vectors and curvatures of a flat triangle, but because the real geometry is not linear, the results may not be accurate. Furthermore, the derivatives of the first fundamental form coefficients, which are used in the geodesic computation, cannot be obtained from curvatures and normals. It is also known that the estimation of surface normals and curvatures from the polygonal approximation of the surface by least squares fitting does not guarantee accurate approximation [9].

In addition to these developments in discrete differential geometry [20], many surface interrogation tools that are specially designed for a triangular mesh have been developed in the last decade. Yong et al. [32] used highlight lines to remove local irregularities in a triangular mesh. This was done by first calculating the highlight distance value for each mesh vertex, and then using linear interpolation to obtain the highlight distance value for the interior points. The computation of a discrete geodesic path between two points on triangular meshes and the computation of a discrete point-source geodesic distance field have been studied by many researchers [21,5,31]. These algorithms are quite different from those being studied for B-splines, since a path goes through alternating sequences of vertices and edges, and only straight line distances are considered, without using differential geometry. Alliez et al. [1] used lines of minimum and maximum curvatures in remeshing in order to determine the appropriate edges for a quad-dominant mesh that reflects the symmetries of the input shape. They used curvature tensor fields to detect umbilics. However, due to the linear interpolation within each triangle, there can be only one umbilic point per triangle. Kalogerakis et al. [10] extracted lines of curvature from point clouds, a method that is applicable to surfaces of arbitrary genus and is robust to noise and outliers.

In this paper we present a unifying technique that enables us to use the surface interrogation tools developed for B-spline surfaces [3,18,9] for triangular mesh models, including first-order surface interrogation methods such as reflection lines and highlight lines [24,17,22], and second-order surface interrogation methods such as the computation of geodesics [15], geodesic offsets [23], lines of curvature [19], and the detection of umbilics. In general, second-order surface interrogation tools for triangular meshes are more difficult than those of first-order interrogation tools due to the higher order of differentiation involved. In order to evaluate high order differentiation accurately, the input mesh must be in high-quality. The last decade has witnessed the introduction of several remeshing algorithms that convert low-quality meshes into highly regular meshes in terms of their size, connectivity, vertex sampling, and maximized angles [27]. Furthermore, remeshing software is readily available on the Internet, e.g., [2]. Accordingly, we assume that the input mesh has been modified by remeshing software so that it becomes a high-quality mesh.

The newly proposed method allows us to compute various surface interrogation tools on a triangular mesh model as though the surface were a B-spline surface without any change in the original algorithm. Our main contributions can be summarized as follows:

– It presents a unifying technique for reusing the shape interrogation tools developed for B-spline, such as circular reflection lines, computation of lines of curvature, geodesics, and detection of umbilics, for a triangular mesh without any change in the original algorithm.
– It introduces a novel approach to parameterize a piece-wise linear triangular mesh as a set of graph triangular Bézier patches. This parameterization facilitates a more accurate approximation of the surface differential properties than other alternatives.
– Most importantly, it combines these techniques into an effective procedure for automatically and accurately computing shape interrogation tools for triangular meshes.

In our unifying technique, we first transform the region of interest of a given triangular mesh into a graph function (z=h(x,y)) so that we can treat the triangular domain within the rectangular domain. At the same time, we convert the triangular mesh into a smooth surface using cubic graph Bézier triangles so that we are able to compute the differentiations. It is not our intent to compare the performance of our methods with that of the methods that were developed especially for triangular meshes.

The remainder of the paper is structured as follows. In Section 2, we briefly review the differential geometry of surfaces. In Section 3, we introduce a novel method to extract the differential invariants properties of the underlying smooth surfaces of the mesh. In Section 4, the method introduced in Section 3 is applied to the computation of circular reflection lines, which is a first-order surface interrogation tool, and in Sections 5 and 6, the second-order surface interrogation tools, including the computation of lines of curvature, the detection of umbilic points, geodesics, and geodesic offsets, are applied, in the given order. Finally, Section 7 concludes the paper.

2 Brief review of differential geometry and notations

In this section, we first offer a brief review of the differential geometry of surfaces [26,7,24]. A point (x, y, z) of a parametric surface is expressed as functions of the parameters u and v in a closed rectangle:

$$\mathbf{r}(u,v) = (x(u,v), y(u,v), z(u,v)), \quad 0 \le u \le 1, \quad 0 \le v \le 1 \ . \tag{1}$$

A unit surface normal vector of the parametric surface is defined by

$$\mathbf{N} = \frac{\mathbf{r}_u \times \mathbf{r}_v}{|\mathbf{r}_u \times \mathbf{r}_v|} \ , \tag{2}$$

where subscripts u and v denote partial differentiation with respect to u and v, respectively. The first fundamental form coefficients E, F, and G and the second fundamental form coefficients L, M, and N of a surface are given by

$$E = \mathbf{r}_u \cdot \mathbf{r}_u, \quad F = \mathbf{r}_u \cdot \mathbf{r}_v, \quad G = \mathbf{r}_v \cdot \mathbf{r}_v \ , \tag{3}$$

$$L = \mathbf{r}_{uu} \cdot \mathbf{N}, \quad M = \mathbf{r}_{uv} \cdot \mathbf{N}, \quad N = \mathbf{r}_{vv} \cdot \mathbf{N} \ . \tag{4}$$

A normal curvature at point P is given by

$$\kappa_n = \frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2} \ , \tag{5}$$

where $\lambda = \frac{dv}{du}$ is the direction of the line tangent to the curve passing through P. The extreme values of κ_n can be obtained by evaluating $\frac{d\kappa_n}{d\lambda} = 0$ in (5), which gives:

$$(E + 2F\lambda + G\lambda^2)(N\lambda + M) - (L + 2M\lambda + N\lambda^2)(G\lambda + F) = 0 \ , \tag{6}$$

and hence,

$$\kappa_n = \frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2} = \frac{M + N\lambda}{F + G\lambda} \ . \tag{7}$$

Furthermore, since

$$\begin{aligned} E + 2F\lambda + G\lambda^2 &= (E + F\lambda) + \lambda(F + G\lambda) \ , \\ L + 2M\lambda + N\lambda^2 &= (L + M\lambda) + \lambda(M + N\lambda) \ , \end{aligned} \tag{8}$$

(6) can be reduced to

$$(E + F\lambda)(M + N\lambda) = (L + M\lambda)(F + G\lambda) \ , \tag{9}$$

and hence,

$$\kappa_n = \frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2} = \frac{M + N\lambda}{F + G\lambda} = \frac{L + M\lambda}{E + F\lambda} \ . \tag{10}$$

Therefore, the extreme values of κ_n satisfy the two simultaneous equations

$$\begin{aligned} (L - \kappa_n E)du + (M - \kappa_n F)dv &= 0 \ , \\ (M - \kappa_n F)du + (N - \kappa_n G)dv &= 0 \ . \end{aligned} \tag{11}$$

These equations form a homogeneous linear system of equations for du and dv, which will have a nontrivial solution if and only if

$$det\begin{vmatrix} L-\kappa_n E & M-\kappa_n F \\ M-\kappa_n F & N-\kappa_n G \end{vmatrix} = 0 \ , \tag{12}$$

where $det|\ \ |$ denotes the determinant of a matrix, or by expansion

$$(EG-F^2)\kappa_n^2 - (EN+GL-2FM)\kappa_n + (LN-M^2) = 0 \ . \tag{13}$$

If we express Gaussian and mean curvatures using the the first and second fundamental form coefficients,

$$K = \frac{LN-M^2}{EG-F^2}, \tag{14}$$

$$H = \frac{EN+GL-2FM}{2(EG-F^2)} \ , \tag{15}$$

then the quadratic equation for κ_n (13) is simplified to:

$$\kappa_n^2 - 2H\kappa_n + K = 0 \ . \tag{16}$$

Solving the quadratic equation yields

$$\kappa_1 = H + \sqrt{H^2-K}, \quad \kappa_2 = H - \sqrt{H^2-K} \ , \tag{17}$$

where κ_1 is the maximum principal curvature and κ_2 is the minimum principal curvature. The directions in the tangent plane for which κ_n takes maximum and minimum values are called the principal directions.

3 Triangular domain to rectangular domain

In order to use the shape interrogation tools developed for tensor product B-spline surfaces for triangular meshes, two obstacles must be overcome.

1. A B-spline surface (patch) is a tensor product surface defined on a rectangular domain, while a triangular mesh is defined on a triangular domain.
2. A B-spline surface of order k is (k-p-1) times differentiable at a knot with multiplicity p, and thus, it has C^{k-p-1} continuity, while a triangular mesh is piecewise flat and has only C^0 continuity.

The first obstacle can be overcome by transforming the coordinate system of the region of interest of a given triangular mesh into a graph function (z=h(x,y)) or into a parametric form (x,y,h(x,y)) so that the triangular domain can be treated within the rectangular domain. This coordinate transformation technique is introduced in Section 3.1. The second obstacle can be overcome by replacing the flat triangles with graph triangular Bézier patches, an approach that is presented in Sections 3.2 and 3.3. We employ the half-edge data structure for the handling of the triangular mesh.

3.1 Coordinate transformation

It is a well known fact that locally, any surface is the graph of a differential function [7]. Accordingly, we represent the surface in a graph form in the region of interest. Consider a global frame O-XYZ and a vertex of a triangular mesh $\mathbf{R} = [X, Y, Z]^T$, as illustrated in Figure 1. To represent the triangular mesh in a graph form, an orthogonal Cartesian reference frame o-xyz is attached to a vertex point, which can be any vertex in the region of interest. We represent the vertex point $\mathbf{R}$ in the frame o-xyz as $\mathbf{r} = (x, y, z)^T$. We choose unit vectors $\mathbf{a}$, $\bar{\mathbf{N}} \times \mathbf{a}$, and $\bar{\mathbf{N}}$ as the directions of the x, y, and z axes, respectively, as shown in Figure 1, where $\bar{\mathbf{N}} = (\bar{N}_X, \bar{N}_Y, \bar{N}_Z)^T$ is an average of the unit normal vectors in the region of interest, and $\mathbf{a} = (a_X, a_Y, a_Z)^T$ is an arbitrary unit vector that is orthogonal to $\bar{\mathbf{N}}$. The unit normal vector at a vertex of a triangular mesh can be evaluated using Equation (29).

If we concatenate these three unit vectors $\mathbf{a}$, $\bar{\mathbf{N}}\times\mathbf{a}$, and $\bar{\mathbf{N}}$ in a single matrix, we obtain a description of the orientation of the graph form with respect to the frame *o-xyz*, which is called a *rotation matrix* $\boldsymbol{\Omega}$:

$$\boldsymbol{\Omega} = \begin{pmatrix} a_X & \bar{N}_Y a_Z - \bar{N}_Z a_Y & \bar{N}_X \\ a_Y & \bar{N}_Z a_X - \bar{N}_X a_Z & \bar{N}_Y \\ a_Z & \bar{N}_X a_Y - \bar{N}_Y a_X & \bar{N}_Z \end{pmatrix}_{(o)} . \tag{18}$$

Then, the relationship between $\mathbf{R}$ and $\mathbf{r}$ is:

$$\mathbf{R} = \mathbf{R}_o + \boldsymbol{\Omega}\mathbf{r} . \tag{19}$$

Using (19), we can solve for $\mathbf{r}$ as a function of $\mathbf{R}$, which is the coordinate of point P expressed in frame *o-xyz* as a function of the coordinate of point P expressed in the O-XYZ frame, as

$$\mathbf{r} = \boldsymbol{\Omega}^{-1}(\mathbf{R} - \mathbf{R}_o) , \tag{20}$$

where $\boldsymbol{\Omega}^{-1}$ is the inverse matrix of $\boldsymbol{\Omega}$. Since $\boldsymbol{\Omega}$ is an orthonormal matrix, $\boldsymbol{\Omega}^{-1}$ can be replaced by the transpose matrix $\boldsymbol{\Omega}^T$; therefore

$$\mathbf{r} = \begin{pmatrix} x \\ y \\ h(x,y) \end{pmatrix} = \boldsymbol{\Omega}^T[\mathbf{R} - \mathbf{R}_o] , \tag{21}$$

or equivalently:

$$x = \mathbf{a} \cdot [\mathbf{R} - \mathbf{R}_o] , \tag{22}$$
$$y = (\bar{\mathbf{N}} \times \mathbf{a}) \cdot [\mathbf{R} - \mathbf{R}_o] , \tag{23}$$
$$z = h(x,y) = \bar{\mathbf{N}} \cdot [\mathbf{R} - \mathbf{R}_o] . \tag{24}$$

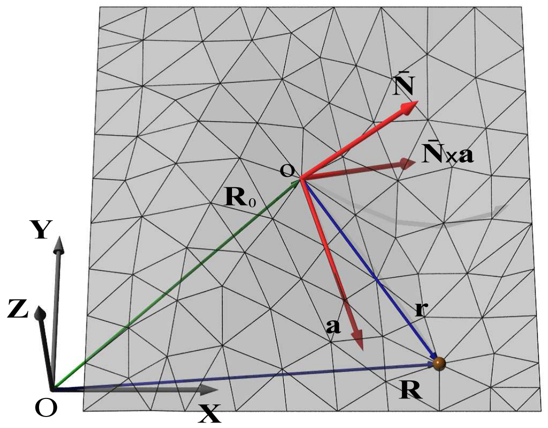


Fig. 1 Coordinate transformation.

3.2 Curved PN triangles

A point $\mathbf{p}$ can be represented by barycentric coordinates (u, v, w) with respect to a triangle with vertices $\mathbf{P}_1$, $\mathbf{P}_2$, and $\mathbf{P}_3$ as:

$$\mathbf{p}(u,v,w) = u\mathbf{P}_1 + v\mathbf{P}_2 + w\mathbf{P}_3 , \tag{25}$$

where $u+v+w=1$ [8]. If we denote the orthogonal projection of $\triangle\mathbf{P}_1\mathbf{P}_2\mathbf{P}_3$ onto the xy-plane as $\triangle\hat{\mathbf{P}}_1\hat{\mathbf{P}}_2\hat{\mathbf{P}}_3$, then the area of $\triangle\hat{\mathbf{P}}_1\hat{\mathbf{P}}_2\hat{\mathbf{P}}_3$ is obtained by

$$area\left(\hat{\mathbf{P}}_1,\hat{\mathbf{P}}_2,\hat{\mathbf{P}}_3\right)=\frac{1}{2}det\begin{vmatrix}\hat{\mathbf{P}}_1^x & \hat{\mathbf{P}}_2^x & \hat{\mathbf{P}}_3^x\\ \hat{\mathbf{P}}_1^y & \hat{\mathbf{P}}_2^y & \hat{\mathbf{P}}_3^y\\ 1 & 1 & 1\end{vmatrix}, \tag{26}$$

where superscripts x and y denote the x and y components of the vectors $\mathbf{P}_1$, $\mathbf{P}_2$, and $\mathbf{P}_3$, respectively. Given a point $\mathbf{p}$ in $\triangle\mathbf{P}_1\mathbf{P}_2\mathbf{P}_3$, if we denote its orthogonal projection onto the xy-plane by $\hat{\mathbf{p}}$, the barycentric coordinates (u,v,w) with respect to the triangle $\triangle\hat{\mathbf{P}}_1\hat{\mathbf{P}}_2\hat{\mathbf{P}}_3$ are evaluated as:

$$u=\frac{area\left(\hat{\mathbf{p}},\hat{\mathbf{P}}_2,\hat{\mathbf{P}}_3\right)}{area\left(\hat{\mathbf{P}}_1,\hat{\mathbf{P}}_2,\hat{\mathbf{P}}_3\right)},\quad v=\frac{area\left(\hat{\mathbf{P}}_1,\hat{\mathbf{p}},\hat{\mathbf{P}}_3\right)}{area\left(\hat{\mathbf{P}}_1,\hat{\mathbf{P}}_2,\hat{\mathbf{P}}_3\right)},\quad w=\frac{area\left(\hat{\mathbf{P}}_1,\hat{\mathbf{P}}_2,\hat{\mathbf{p}}\right)}{area\left(\hat{\mathbf{P}}_1,\hat{\mathbf{P}}_2,\hat{\mathbf{P}}_3\right)}. \tag{27}$$

Note that barycentric coordinates (u,v,w) of point $\mathbf{p}$ with respect to the triangle $\triangle\mathbf{P}_1\mathbf{P}_2\mathbf{P}_3$ have the same values as those evaluated by Equation (27).

Vlachos et al. [29] introduced so-called curved point-normal triangles (PN triangles), which replace conventional flat triangles with a small number of flat sub-triangles that are computed based on the geometry of triangular cubic Bézier patches, in order to improve the visual quality for vertex shading operations. The triangular cubic Bézier patch $\mathbf{b}(\mathbf{u})$ is defined as follows [8]:

$$\begin{aligned}\mathbf{b}(\mathbf{u})&=\sum_{|\mathbf{i}|=3}\mathbf{b}_\mathbf{i}B^3_\mathbf{i}(\mathbf{u})=\sum_{|\mathbf{i}|=3}\mathbf{b}_{ijk}\frac{3!}{i!j!k!}u^iv^jw^k\\&=\mathbf{b}_{300}u^3+\mathbf{b}_{030}v^3+\mathbf{b}_{003}w^3+\mathbf{b}_{210}3u^2v+\mathbf{b}_{120}3uv^2+\mathbf{b}_{201}3u^2w\\&+\mathbf{b}_{021}3v^2w+\mathbf{b}_{102}3uw^2+\mathbf{b}_{012}3vw^2+\mathbf{b}_{111}6uvw,\end{aligned} \tag{28}$$

where $\mathbf{u}=(u,v,w)$, $u,v,w\geq0$, $u+v+w=1$, $|\mathbf{i}|=i+j+k$, and $\mathbf{b}_\mathbf{i}$ are the control points.

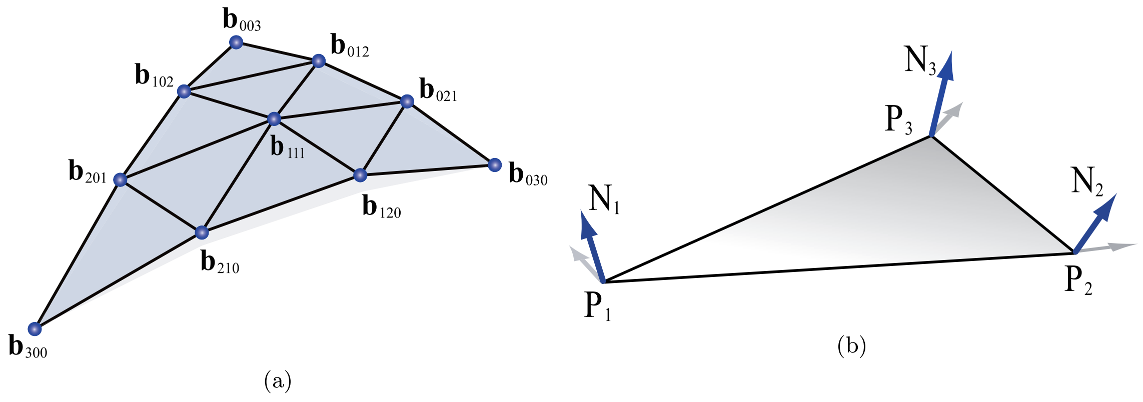


Fig. 2 Curved PN triangle Bézier patch: (a) Control net of the PN triangle. (b) Input data for the PN triangle, points $\mathbf{P}_i$, and normals $\mathbf{N}_i$ $(i=1,2,3)$.

The ten control points of a cubic Bézier patch, which are shown in Figure 2(a), are determined based on the point and normal information at the vertices of the flat triangle (see Figure 2(b)). The unit normal vector $\mathbf{N}_i$ at the vertex $\mathbf{P}_i$ is estimated by taking the average of the unit normal vector $\mathbf{N}_j$ of the surrounding polygons that meet at the vertex $\mathbf{P}_i$ weighted by the facet angles α_j at the vertex [28]:

$$\mathbf{N}_i=\frac{\sum_{j=1}^n\alpha_j\mathbf{N}_j}{|\sum_{j=1}^n\alpha_j\mathbf{N}_j|}. \tag{29}$$

Vlachos et al. [29] group the control points $\mathbf{b}_{ijk}$ into three groups as:

1. vertex coefficients: $\mathbf{b}_{300}$, $\mathbf{b}_{030}$, $\mathbf{b}_{003}$.
2. tangent coefficients: $\mathbf{b}_{210}$, $\mathbf{b}_{120}$, $\mathbf{b}_{021}$, $\mathbf{b}_{012}$, $\mathbf{b}_{102}$, $\mathbf{b}_{201}$.
3. center coefficient: $\mathbf{b}_{111}$.

The vertex coefficients are placed so that they match the corner positions:

$$\mathbf{b}_{300} = \mathbf{P}_1, \;\; \mathbf{b}_{030} = \mathbf{P}_2, \;\; \mathbf{b}_{003} = \mathbf{P}_3 \; . \tag{30}$$

The tangent coefficients of the curved PN triangles are obtained by first distributing the $\mathbf{b}_{ijk}$ uniformly over the flat triangle, and then projecting the two tangent coefficients on the flat triangle that are closest to each corner onto the tangent plane defined by the normal at the corner, as follows:

$$\begin{aligned} \mathbf{b}_{210} &= (2\mathbf{P}_1 + \mathbf{P}_2 - w_{12}\mathbf{N}_1)/3, & \mathbf{b}_{120} &= (2\mathbf{P}_2 + \mathbf{P}_1 - w_{21}\mathbf{N}_2)/3, \\ \mathbf{b}_{021} &= (2\mathbf{P}_2 + \mathbf{P}_3 - w_{23}\mathbf{N}_2)/3, & \mathbf{b}_{012} &= (2\mathbf{P}_3 + \mathbf{P}_2 - w_{32}\mathbf{N}_3)/3, \\ \mathbf{b}_{102} &= (2\mathbf{P}_3 + \mathbf{P}_1 - w_{31}\mathbf{N}_3)/3, & \mathbf{b}_{201} &= (2\mathbf{P}_1 + \mathbf{P}_3 - w_{13}\mathbf{N}_1)/3, \end{aligned} \tag{31}$$

where $w_{ij} = (\mathbf{P}_j - \mathbf{P}_i) \cdot \mathbf{N}_i$. Finally, the center coefficient is defined as follows:

$$\begin{aligned} \mathbf{E} &= (\mathbf{b}_{210} + \mathbf{b}_{120} + \mathbf{b}_{021} + \mathbf{b}_{012} + \mathbf{b}_{102} + \mathbf{b}_{201})/6, \\ \mathbf{V} &= (\mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3)/3, \\ \mathbf{b}_{111} &= \mathbf{E} + (\mathbf{E} - \mathbf{V})/2. \end{aligned} \tag{32}$$

3.3 Graph PN triangular patch

After the coordinate transformation, the region of interest of the input triangular mesh is represented in the nonparametric surface form of z=h(x,y), which can be converted into a parametric form (x, y, h(x,y)). Accordingly, the curved PN triangles introduced in Section 3.2 cannot be directly used for our purposes. In this section, we introduce a graph of a curved PN triangular patch whose control points are determined so that the resulting Bézier patch is a graph function (see Figure 3(b)).

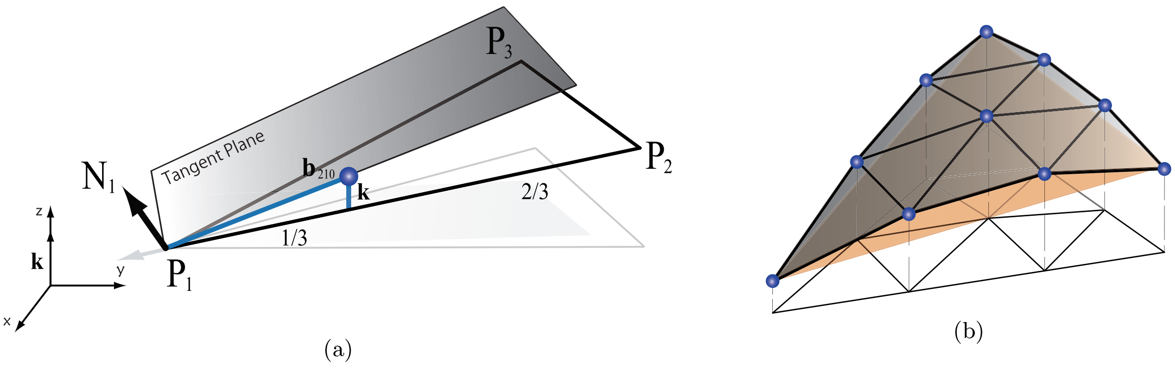


Fig. 3 Graph triangular Bézier patch: (a) Construction of a tangent coefficient $\mathbf{b}_{210}$. (b) Graph cubic Bézier patch. The original triangular mesh is depicted in orange.

Using the linear precision property of the Bernstein polynomials [8], the variable u can be written as:

$$u = \sum_{|\mathbf{i}|=n} \frac{i}{n} B^n_{\mathbf{i}}(\mathbf{u}), \tag{33}$$

and analogous formulas can be written for the variables v and w. The height function z can be expressed in the Bernstein form:

$$z = \sum_{|\mathbf{i}|=n} h_{\mathbf{i}} B^n_{\mathbf{i}}(\mathbf{u}), \tag{34}$$

where $h_{\mathbf{i}}$ are the height coefficients. Thus, a graph of function z is given by a triangular Bézier surface:

$$\mathbf{b}(\mathbf{u}) = \begin{bmatrix} u \\ v \\ w \\ z \end{bmatrix} = \sum_{|\mathbf{i}|=n} \begin{bmatrix} i/n \\ j/n \\ k/n \\ h_{\mathbf{i}} \end{bmatrix} B^n_{\mathbf{i}}(\mathbf{u}). \tag{35}$$

Equation (35) can be expressed in terms of the vertices of the domain triangle $\triangle \mathbf{P}_1\mathbf{P}_2\mathbf{P}_3$ as follows:

$$\mathbf{b}(\mathbf{u}) = \sum_{|\mathbf{i}|=n} \left[\mathbf{p}\left(\frac{i}{n}, \frac{j}{n}, \frac{k}{n}\right) + h_{\mathbf{i}}\mathbf{k} \right] B^n_{\mathbf{i}}(\mathbf{u}) \,, \tag{36}$$

where $\mathbf{p}(\frac{i}{n}, \frac{j}{n}, \frac{k}{n})$ are the base points on $\triangle \mathbf{P}_1\mathbf{P}_2\mathbf{P}_3$ as defined by Equation (25), and $\mathbf{k} = (0,0,1)^T$ is a unit vector along the local z-axis. The vertex height coefficients h_{300}, h_{030}, h_{003} are equal to zero, while the tangent height coefficients h_{210}, h_{120}, h_{021}, h_{012}, h_{102}, h_{201} are obtained by computing the line-plane intersections. For each corner, we have lines that emanate from the two closest base points along $\mathbf{k}$, and intersect them with the tangent plane defined by the normal at the corner, as shown in Figure 3(a). This process is different from that used for the original curved PN triangles. The tangent height coefficient h_{210}, for example, is computed by intersecting the line

$$\mathbf{L}_{210} = \frac{2\mathbf{P}_1 + \mathbf{P}_2}{3} + h_{210}\mathbf{k} \,, \tag{37}$$

with the tangent plane

$$(\mathbf{L}_{210} - \mathbf{P}_1) \cdot \mathbf{N}_1 = 0 \,, \tag{38}$$

which yields

$$h_{210} = \frac{(\mathbf{P}_1 - \mathbf{P}_2) \cdot \mathbf{N}_1}{3\mathbf{k} \cdot \mathbf{N}_1} \,. \tag{39}$$

Accordingly, we obtain

$$\mathbf{b}_{210} = \frac{2\mathbf{P}_1 + \mathbf{P}_2}{3} + \frac{(\mathbf{P}_1 - \mathbf{P}_2) \cdot \mathbf{N}_1}{3\mathbf{k} \cdot \mathbf{N}_1}\mathbf{k} \,. \tag{40}$$

Similarly, we obtain

$$\begin{aligned} \mathbf{b}_{201} &= \frac{2\mathbf{P}_1 + \mathbf{P}_3}{3} + \frac{(\mathbf{P}_1 - \mathbf{P}_3) \cdot \mathbf{N}_1}{3\mathbf{k} \cdot \mathbf{N}_1}\mathbf{k}, & \mathbf{b}_{102} &= \frac{2\mathbf{P}_3 + \mathbf{P}_1}{3} + \frac{(\mathbf{P}_3 - \mathbf{P}_1) \cdot \mathbf{N}_3}{3\mathbf{k} \cdot \mathbf{N}_3}\mathbf{k}, \\ \mathbf{b}_{012} &= \frac{2\mathbf{P}_3 + \mathbf{P}_2}{3} + \frac{(\mathbf{P}_3 - \mathbf{P}_2) \cdot \mathbf{N}_3}{3\mathbf{k} \cdot \mathbf{N}_3}\mathbf{k}, & \mathbf{b}_{021} &= \frac{2\mathbf{P}_2 + \mathbf{P}_3}{3} + \frac{(\mathbf{P}_2 - \mathbf{P}_3) \cdot \mathbf{N}_2}{3\mathbf{k} \cdot \mathbf{N}_2}\mathbf{k}, \\ \mathbf{b}_{120} &= \frac{2\mathbf{P}_2 + \mathbf{P}_1}{3} + \frac{(\mathbf{P}_2 - \mathbf{P}_1) \cdot \mathbf{N}_2}{3\mathbf{k} \cdot \mathbf{N}_2}\mathbf{k}. \end{aligned} \tag{41}$$

In order to apply the shape interrogation tools to the triangular Bézier patches, we must first identify in which triangle the point lies. If we suppose that an orthogonally projected point $\hat{\mathbf{p}}$ lies inside $\triangle \hat{\mathbf{P}}_1\hat{\mathbf{P}}_2\hat{\mathbf{P}}_3$, then $\hat{\mathbf{p}}$ can be expressed as

$$\hat{\mathbf{p}} = \hat{\mathbf{P}}_1 + \alpha\mathbf{V}_1 + \beta\mathbf{V}_2, \tag{42}$$

where $\mathbf{V}_1$ and $\mathbf{V}_2$ denote the vectors from $\hat{\mathbf{P}}_1$ to $\hat{\mathbf{P}}_2$ and $\hat{\mathbf{P}}_1$ to $\hat{\mathbf{P}}_3$, and α and β are constants. Solving for α and β yields

$$\alpha = \frac{det|\hat{\mathbf{p}}\mathbf{V}_2| - det|\mathbf{V}_0\mathbf{V}_2|}{det|\mathbf{V}_1\mathbf{V}_2|}, \quad \beta = -\frac{det|\hat{\mathbf{p}}\mathbf{V}_1| - det|\mathbf{V}_0\mathbf{V}_1|}{det|\mathbf{V}_1\mathbf{V}_2|}. \tag{43}$$

The point $\hat{\mathbf{p}}$ then lies inside $\triangle \hat{\mathbf{P}}_1\hat{\mathbf{P}}_2\hat{\mathbf{P}}_3$ if $\alpha, \beta > 0$ and $\alpha + \beta < 1$ [30]. Second, differentiations with respect to x and y of the local coordinates are necessary. The derivatives for tensor product patches are evaluated by partial derivatives, while the directional derivatives are used for triangular patches as follows:

$$D_{\mathbf{d}}\mathbf{b}(\mathbf{u}) = d\mathbf{b}_u(\mathbf{u}) + e\mathbf{b}_v(\mathbf{u}) + f\mathbf{b}_w(\mathbf{u}), \tag{44}$$

where subscript $\mathbf{d}{=}(d, e, f)$ denotes the derivative direction. It is defined by $\mathbf{d}{=}\mathbf{u}_2 - \mathbf{u}_1$, where $\mathbf{u}_1$ and $\mathbf{u}_2$ are two points in the triangular domain, and is characterized by $d + e + f{=}0$ [8]. Note that the tangent vector depends on the length of $\mathbf{d}$. The second order derivative is also computed:

$$D^2_{\mathbf{d}}\mathbf{b}(\mathbf{u}) = \begin{bmatrix} d \; e \; f \end{bmatrix} \begin{bmatrix} \frac{\partial^2\mathbf{b}}{\partial u^2} & \frac{\partial^2\mathbf{b}}{\partial u\partial v} & \frac{\partial^2\mathbf{b}}{\partial u\partial w} \\ \frac{\partial^2\mathbf{b}}{\partial u\partial v} & \frac{\partial^2\mathbf{b}}{\partial v^2} & \frac{\partial^2\mathbf{b}}{\partial v\partial w} \\ \frac{\partial^2\mathbf{b}}{\partial u\partial w} & \frac{\partial^2\mathbf{b}}{\partial v\partial w} & \frac{\partial^2\mathbf{b}}{\partial w^2} \end{bmatrix} \begin{bmatrix} d \\ e \\ f \end{bmatrix} . \tag{45}$$

The derivative direction in x is obtained by taking the partial derivative of u, v, w in Equation (27) with respect to x:

$$d = \frac{\partial u}{\partial x} = \frac{1}{S}(\hat{\mathbf{P}}^y_2 - \hat{\mathbf{P}}^y_3), \quad e = \frac{\partial v}{\partial x} = \frac{1}{S}(\hat{\mathbf{P}}^y_3 - \hat{\mathbf{P}}^y_1), \quad f = \frac{\partial w}{\partial x} = \frac{1}{S}(\hat{\mathbf{P}}^y_1 - \hat{\mathbf{P}}^y_2), \tag{46}$$

where $S = area(\hat{\mathbf{P}}_1, \hat{\mathbf{P}}_2, \hat{\mathbf{P}}_3)$. Similarly, the derivative direction in y is obtained as follows:

$$d = \frac{\partial u}{\partial y} = \frac{1}{S}(\hat{\mathbf{P}}^x_3 - \hat{\mathbf{P}}^x_2), \quad e = \frac{\partial v}{\partial y} = \frac{1}{S}(\hat{\mathbf{P}}^x_1 - \hat{\mathbf{P}}^x_3), \quad f = \frac{\partial w}{\partial y} = \frac{1}{S}(\hat{\mathbf{P}}^x_2 - \hat{\mathbf{P}}^x_1). \tag{47}$$

Note that the normal vectors of the resulting graph Bézier patch at the vertices coincide with the given normal vectors $\mathbf{N}_1$, $\mathbf{N}_2$, and $\mathbf{N}_3$:

$$\frac{D_{(-1,1,0)}\mathbf{b}(1,0,0) \times D_{(-1,0,1)}\mathbf{b}(1,0,0)}{|D_{(-1,1,0)}\mathbf{b}(1,0,0) \times D_{(-1,0,1)}\mathbf{b}(1,0,0)|} = \frac{(\mathbf{b}_{210} - \mathbf{b}_{300}) \times (\mathbf{b}_{201} - \mathbf{b}_{300})}{|(\mathbf{b}_{210} - \mathbf{b}_{300}) \times (\mathbf{b}_{201} - \mathbf{b}_{300})|} = \mathbf{N}_1, \tag{48}$$

$$\frac{D_{(0,-1,1)}\mathbf{b}(0,1,0) \times D_{(1,-1,0)}\mathbf{b}(0,1,0)}{|D_{(0,-1,1)}\mathbf{b}(0,1,0) \times D_{(1,-1,0)}\mathbf{b}(0,1,0)|} = \frac{(\mathbf{b}_{021} - \mathbf{b}_{030}) \times (\mathbf{b}_{120} - \mathbf{b}_{030})}{|(\mathbf{b}_{021} - \mathbf{b}_{030}) \times (\mathbf{b}_{120} - \mathbf{b}_{030})|} = \mathbf{N}_2, \tag{49}$$

$$\frac{D_{(1,0,-1)}\mathbf{b}(0,0,1) \times D_{(0,1,-1)}\mathbf{b}(0,0,1)}{|D_{(1,0,-1)}\mathbf{b}(0,0,1) \times D_{(0,1,-1)}\mathbf{b}(0,0,1)|} = \frac{(\mathbf{b}_{102} - \mathbf{b}_{003}) \times (\mathbf{b}_{012} - \mathbf{b}_{003})}{|(\mathbf{b}_{102} - \mathbf{b}_{003}) \times (\mathbf{b}_{012} - \mathbf{b}_{003})|} = \mathbf{N}_3 . \tag{50}$$

Furthermore, two adjacent patches share the same cubic Bézier curve, and hence, it is water-tight.

4 Circular reflection lines

Reflection lines [12] simulate the mirror images of a family of radiating parallel straight lines on a smooth surface as viewed from a fixed point. In this method, deviations of the surface from a smooth shape can be detected by irregularities of the reflection lines. These surface deviations are corrected by modifying the original surface so that the new surface has reflection lines without any irregularities. Maekawa et al. [17,22] extended the concept of reflection lines to circular reflection lines by replacing a family of parallel light lines with concentric light circles so that the surface fairness can be captured in all directions.

A parametric representation of the circular light source is given by

$$\mathbf{L}(\theta) = \mathbf{A} + R(\cos\theta\mathbf{n} + \sin\theta\mathbf{b}) , \tag{51}$$

where $\mathbf{A}$ and R are the center point and the radius of the circular light source, respectively. The unit vectors $\mathbf{n}$ and $\mathbf{b}$ that lie in the plane that contains the circular light are orthogonal. They form a frame (or trihedron) together with a unit vector $\mathbf{t}$ such that $\mathbf{t} = \mathbf{n} \times \mathbf{b}$, and hence, $\mathbf{t}$ is perpendicular to the plane that contains the circular light, as shown in Figure 4(a).

Let us denote the eye position by $\mathbf{E}$, the unit vectors $\frac{\mathbf{E}-\mathbf{Q}}{|\mathbf{E}-\mathbf{Q}|}$ and $\frac{\mathbf{L}(\theta)-\mathbf{Q}}{|\mathbf{L}(\theta)-\mathbf{Q}|}$ by $\mathbf{e}$ and $\mathbf{c}$ as shown in Figure 4(a), where $\mathbf{Q}$ is a point on the reflection line. We then have the following three equations for $\mathbf{c}$:

$$\mathbf{c} \cdot \mathbf{N}(u, v) = \cos\alpha, \qquad \mathbf{c} \cdot \mathbf{e}(u, v) = \cos 2\alpha, \qquad |\mathbf{c}| = 1 , \tag{52}$$

where $\mathbf{N}$ is a unit surface normal at $\mathbf{Q}$, and $\cos\alpha = \mathbf{e}\cdot\mathbf{N}(u,v)$. Let us now define an extension of vector $\mathbf{c}$ at a surface point $\mathbf{Q}$ as

$$\mathbf{F}(\tau) = \mathbf{Q} + \tau\mathbf{c}\ , \tag{53}$$

where τ is a parameter. The distance vector $\mathbf{d}$ directed from the line $\mathbf{F}(\tau)$ to the circle $\mathbf{L}(\theta)$ is given by

$$\mathbf{d} = \mathbf{A} + R(\cos\theta\mathbf{n} + \sin\theta\mathbf{b}) - (\mathbf{Q} + \tau\mathbf{c})\ . \tag{54}$$

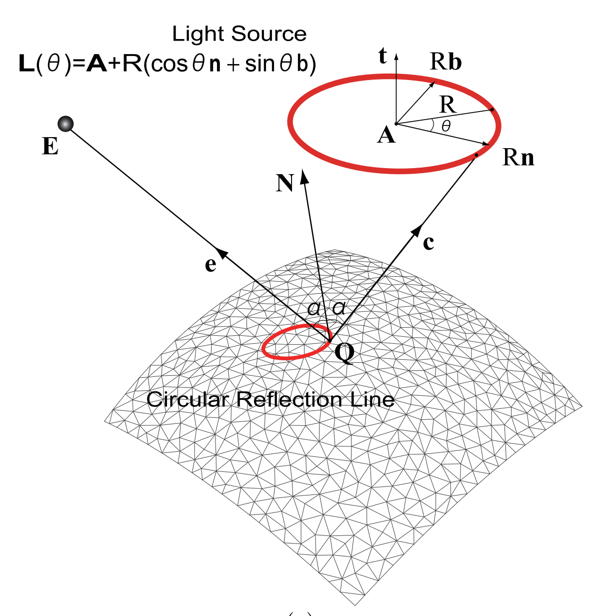
Furthermore, the squared distance function D is defined as

$$D(\tau,\theta) = \mathbf{d}\cdot\mathbf{d} = |(\mathbf{A} + R(\cos\theta\mathbf{n} + \sin\theta\mathbf{b})) - (\mathbf{Q} + \tau\mathbf{c})|^2\ . \tag{55}$$

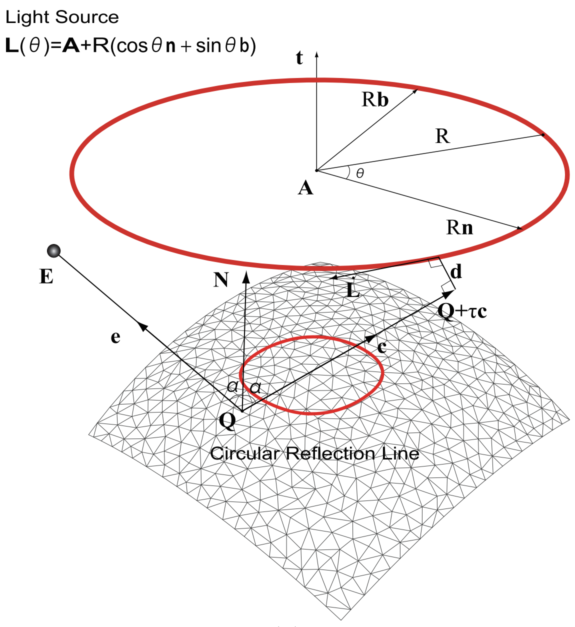
To compute the minimum distance, we need to evaluate the stationary points of the squared distance function, which satisfy the following two equations [24]

$$D_\tau(\tau,\theta) = D_\theta(\tau,\theta) = 0\ , \tag{56}$$

from which τ and θ are computed.



(a)



(b)

Fig. 4 Circular reflection lines: (a) Definition. (b) Signed distance function.

The tangent vector of the circle is given by differentiating Equation (51) with respect to θ, which yields

$$\dot{\mathbf{L}}(\theta) = R(-\sin\theta\mathbf{n} + \cos\theta\mathbf{b})\ . \tag{57}$$

By definition, we have $\mathbf{d}\cdot\mathbf{c} = 0$ and $\mathbf{d}\cdot\dot{\mathbf{L}} = 0$, and hence, $(\mathbf{c}\times\dot{\mathbf{L}})\cdot\mathbf{c} = 0$ and $(\mathbf{c}\times\dot{\mathbf{L}})\cdot\dot{\mathbf{L}} = 0$ according to the definition of the triple scalar product (see Figure 4(b)). We can thus conclude that the distance vector $\mathbf{d}$ is parallel to $\mathbf{c}\times\dot{\mathbf{L}}$. A signed distance function d_s can be defined by obtaining the dot product with the unit vector $\frac{\mathbf{c}\times\dot{\mathbf{L}}}{|\mathbf{c}\times\dot{\mathbf{L}}|}$ as follows:

$$d_s(x,y) = (\mathbf{A} + R(\cos\theta\mathbf{n} + \sin\theta\mathbf{b}) - (\mathbf{Q}(x,y) + \tau\mathbf{c}(x,y)))\cdot\frac{\mathbf{c}(x,y)\times\dot{\mathbf{L}}(\theta)}{|\mathbf{c}(x,y)\times\dot{\mathbf{L}}(\theta)|}\ . \tag{58}$$

A circular reflection line is defined as a set of points on a surface on which the distance between a circular light source and an extension of the unit vector **c** at the circular reflection lines is zero. In other words, circular reflection lines are points on a surface where the signed distance function vanishes. If we construct a signed distance surface $(x, y, d_s(x, y))$ by evaluating the discrete values of d_s at the grid points in the xy-plane, we can easily compute the pre-image of the circular reflection lines using the contour algorithm [4], in which the curves of zero height are computed as surface-plane intersection problems.

Figure 5 shows the circular reflection lines on a wave-like tessellated surface captured from three different view points.

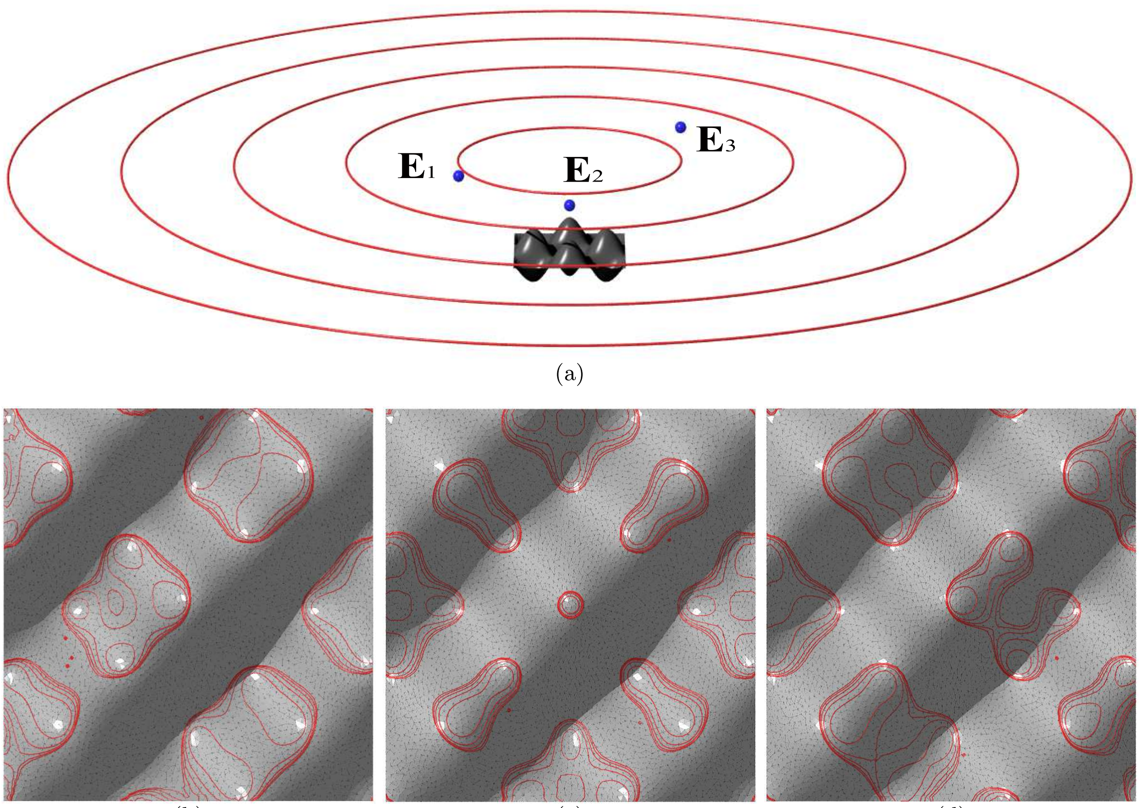


Fig. 5 Circular reflection lines on a wave-like surface (9K triangles): (a) Circular light sources with three different view points $\mathbf{E_1}$, $\mathbf{E_2}$, and $\mathbf{E_3}$. (b) Tessellated surface lit by circular light sources seen from view point $\mathbf{E_1}$. (c) Seen from view point $\mathbf{E_2}$. (d) Seen from view point $\mathbf{E_3}$.

5 Lines of Curvature

A curve on a surface whose tangent at each point is in a principal direction at that point is called a line of curvature. A line of curvature indicates a directional flow of the maximum or the minimum curvature across the surface [19,24]. Since at each point there are two principal directions that are orthogonal, the lines of curvature form an orthogonal net of lines, except at the umbilics. An umbilic is a point on a surface where all normal curvatures are equal, and thus the principal directions are indeterminate. Therefore, the orthogonal net of lines of curvature becomes singular at an umbilic [19].

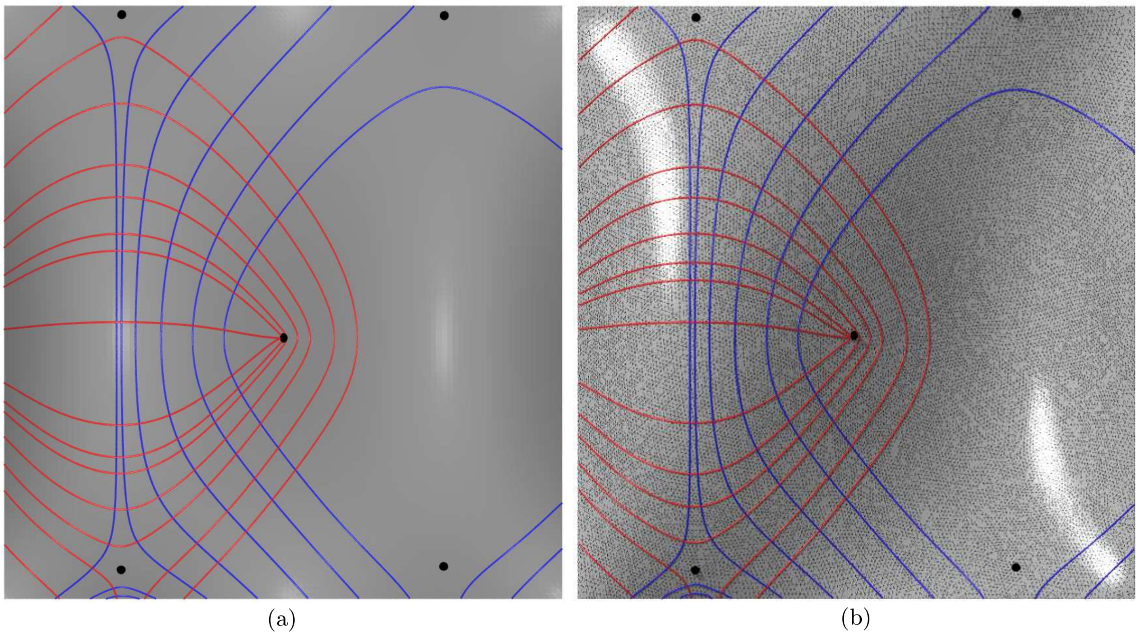


Fig. 6 Computation of lines of curvature and detection of umbilics: (a) B-spline surface. (b) Mesh counterpart (53K triangles).

Table 1 Numerical results for the computation of lines of curvature and the locations of umbilics in Figure 6.

Location of umbilic points				Lines of curvature	
B-spline		Mesh		B-spline	Mesh
u	v	x	y	Time (s)	Time (s)
0.211	0.052	0.212	0.051	3.09	28.85
0.211	0.984	0.212	0.980		
0.789	0.052	0.788	0.050		
0.789	0.984	0.788	0.989		
0.500	0.440	0.495	0.444		

The principal curvature functions κ are defined in (17) as $\kappa(x, y) = H(x, y) \pm \sqrt{H^2(x, y) - K(x, y)}$. If we assume that

$$W(x, y) = H^2(x, y) - K(x, y) \ , \tag{59}$$

then the umbilic occurs precisely at a point where the function $W(x, y)$ is zero. Since κ is a real-valued function, it follows that $W(x, y) \geq 0$, and hence an umbilic occurs where the function $W(x, y)$ has a global minimum [19]. Therefore, the locations of umbilics can be found by first evaluating the discrete values of $W(x, y)$ at the grid points in the xy plane, and then constructing a graph $(x, y, W(x, y))$, and then finding the closed contour [4] with height ϵ using surface-plane intersection problems, where ϵ is a small positive real number. First, we begin with a larger value of ϵ, which generates closed contours surrounding the umbilics. If the circumference of the contour is larger than the prescribed tolerance, ϵ is lowered to compute the smaller loop; otherwise, the average of the vertices of the closed contour polygon is computed as the location of an umbilic point.

Every principal curvature direction vector must fulfill (11). Hence, from the first equation in (11), we obtain

$$\frac{du}{ds} = \eta(M + \kappa F) \ , \quad \frac{dv}{ds} = -\eta(L + \kappa E) \ , \tag{60}$$

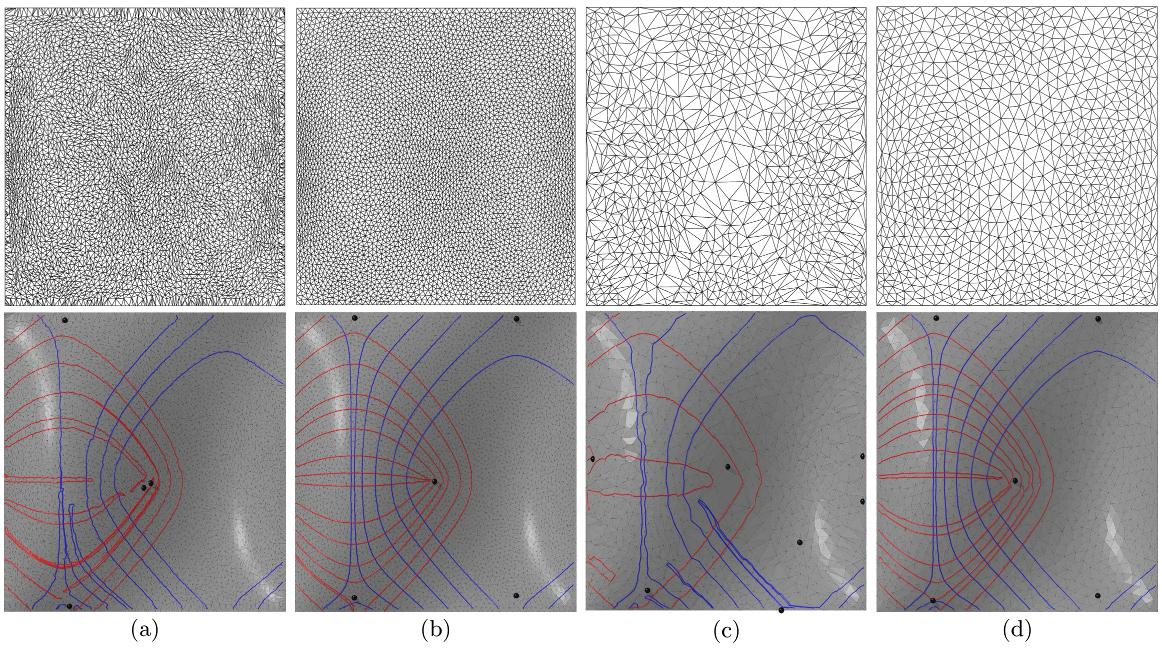


Fig. 7 Sensitivity analysis based on density and quality of tessellated surface. Top row: Mesh structures. Bottom row: Corresponding lines of curvature and detection of umbilics. (a) Low-quality 9K mesh. (b) High-quality 9K mesh. (c) Low-quality 1.7K mesh. (d) High-quality 1.7K mesh.

where the nonzero factor η is determined by applying the normalization condition of the first fundamental form

$$E\left(\frac{du}{ds}\right)^2 + 2F\frac{du}{ds}\frac{dv}{ds} + G\left(\frac{dv}{ds}\right)^2 = 1\ , \tag{61}$$

which imposes the curvature line as an arc length parametrized curve, which in turn, results in

$$\eta = \frac{\pm 1}{\sqrt{E(M+\kappa F)^2 - 2F(M+\kappa F)(L+\kappa E) + G(L+\kappa E)^2}}\ . \tag{62}$$

Since a principal curvature direction vector must also fulfill the second equation of (11) we also obtain

$$\frac{du}{ds} = \mu(N+\kappa G)\ ,\quad \frac{dv}{ds} = -\mu(M+\kappa F)\ . \tag{63}$$

Likewise, μ is determined to be

$$\mu = \frac{\pm 1}{\sqrt{E(N+\kappa G)^2 - 2F(N+\kappa G)(M+\kappa F) + G(M+\kappa F)^2}}\ . \tag{64}$$

The solutions u' and v' of the first and the second equations of (11) are linearly dependent, because the system of linear equations given by (11) has a rank smaller than 2 [19,24]. The details regarding the manner in which a system of differential equations should be used, and how the signs of η and μ are determined, are fully discussed in [19,24]. We can trace the lines of curvature by integrating the initial value problem for a system of coupled nonlinear ordinary differential equations using standard numerical techniques [6] such as the Runge-Kutta method.

In order to check the accuracy of our technique, we compared the lines of curvature as well as the location of umbilics on a B-spline surface and its tessellated counterpart. As depicted in Figure 6, the computational results for the B-spline surface (a) and its mesh counterpart (b), are very similar. In the figure, the red lines correspond to the lines of curvature of the maximum principal curvature, while the blue lines correspond to those of the minimum principal curvature. The locations of umbilics are depicted by the small black circles. The precise locations of the umbilic points on this specific B-spline surface are given in [19], while those on the mesh counterpart are obtained by the contouring method. Table 1 tabulates the location of umbilics and the computational time for lines of curvature in Figure 6. Note that the surface used in Figure 6 is anti-symmetric with respect to $u = 0.5$, and hence, it is not evenly symmetric with regard to the mid-plane.

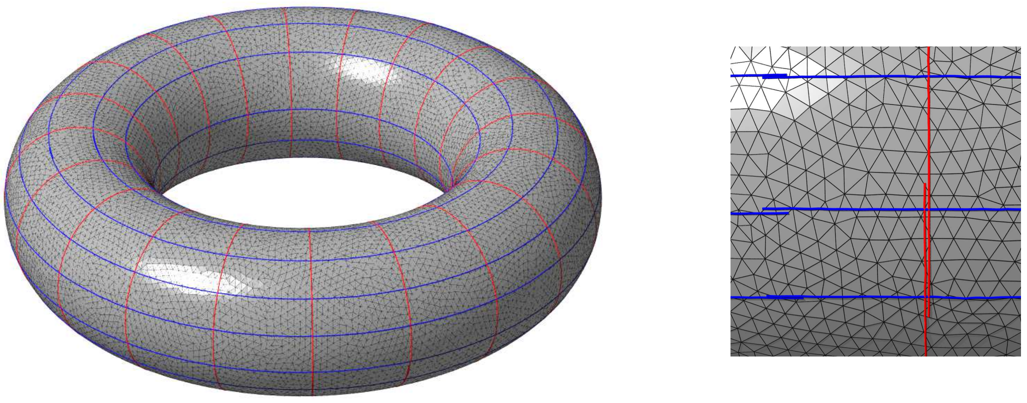


Fig. 8 Computation of the lines of curvature: (a) Torus (36K triangles). (b) Close-up view. Lines of curvature do not match at the start and end points due to the accumulated errors of the Runge-Kutta integration and the approximation error of the B-spline surface owing to the graph triangular Bézier patches.

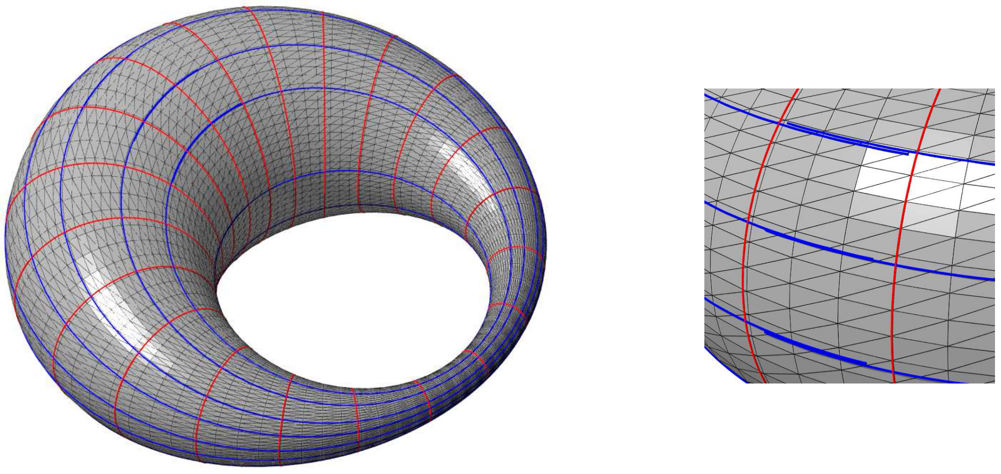
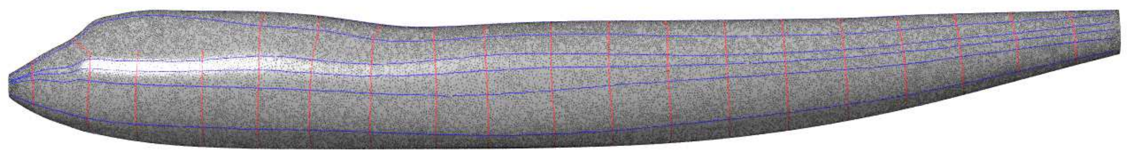


Fig. 9 Computation of the lines of curvature: (a) Ring cyclide (10K triangles). (b) Close-up view. Lines of curvature do not match at the start and end points due to the accumulated errors of the Runge-Kutta integration and the approximation error of the B-spline surface owing to the graph triangular Bézier patches.

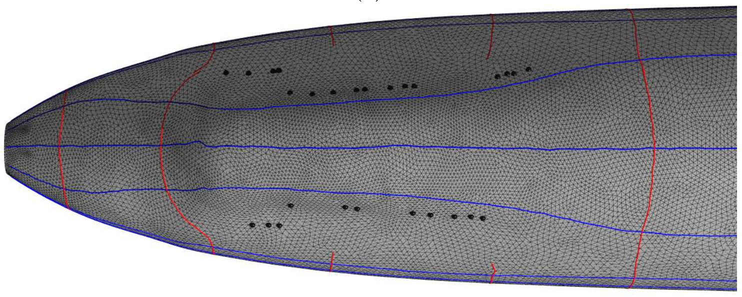
We conducted a sensitivity analysis of the method against the mesh quality, as shown in Figure 7. The top row of Figure 7 shows a variety of input meshes. The first two meshes have 9K triangles of low and high quality, whereas the next two meshes have less triangles (1.7K), also of low and high quality. Here, a high-quality mesh is one that it has been modified by remeshing software. The images in the bottom row are the computational results of the lines of curvature and detection of umbilics, corresponding to the meshes in the top row. It is observed that the density of the mesh does not affect the result as long as high-quality meshes are used for the computation. On the other hand, low-quality meshes cannot generate good results, regardless of the mesh density.

In some engineering applications, the computation of the lines of curvature of the entire object is required. In such cases, one needs to switch from one coordinate system to the other during the computation so that the entire object is covered by all the local graph coordinate systems. When switching from one coordinate system to the other, the two coordinate systems must share an overlapping region. Accordingly, the integration of the initial value problem based on one coordinate system can be terminated in the common region, and can be restarted from the terminated point using the new coordinate system. Such cases are illustrated in Figure 8(a) and Figure 9(a), which depict a torus and a ring cyclide, respectively, both of which are analytical surfaces. For each of these two models, the coordinate transformation is conducted 120 times. Due to the accumulated errors of the Runge-Kutta integration and the approximation error of the B-spline surface owing to the graph triangular Bézier patches, the lines of curvature do not match at the start and end points, as can be seen in Figure 8(b) and Figure 9(b).

Finally, the lines of curvature on a fuselage are illustrated in Figure 10. The fuselage has two lines of non-generic spherical umbilics near the cockpit, as shown in Figure 10(b). Hence, the computation of the lines of curvature is terminated near the umbilics.



(a)



(b)

Fig. 10 Computation of lines of curvature: (a) Fuselage of an airplane (167K triangles). (b) Close-up view. Some of the lines of curvature near the cockpit are terminated because of the presence of the lines of non-generic umbilics.

6 Geodesic

The computation of shortest paths on free-form surfaces is an important problem in ship design, robot-motion planning, the computation of medial axis transforms of trimmed surface patches, terrain navigation, and NC machining [26,15,24]. This section provides the governing equations and solution methods for computing the shortest path between two points on a free-form parametric, as well as the computation of geodesic offsets.

6.1 Geodesic equation

Let C be an arc length parametrized regular curve on the parametric surface $\mathbf{r}(u,v)$ which passes through point P; let it be denoted by

$$\mathbf{r}(s) = \mathbf{r}(u(s), v(s)) \ . \tag{65}$$

Let $\boldsymbol{t}$ be a unit tangent vector of C at P, $\boldsymbol{n}$ be a unit normal vector of C at P, $\mathbf{N}$ be a unit surface normal vector of S at P, and $\mathbf{u}$ be a unit vector perpendicular to $\boldsymbol{t}$ in the tangent plane of the surface, defined by $\mathbf{u} = \mathbf{N}\times\boldsymbol{t}$. The $\mathbf{u}$ component of the curvature vector $\mathbf{k}$ of $\mathbf{r}(s)$ is the geodesic curvature vector $\mathbf{k}_g$, and is given by

$$\mathbf{k}_g = (\mathbf{k}\cdot\mathbf{u})\mathbf{u} \ . \tag{66}$$

The scalar function

$$\kappa_g = \mathbf{k}\cdot\mathbf{u} \ , \tag{67}$$

is called the geodesic curvature of C at P. According to the definition [26,15,24], any geodesic on a surface must satisfy $\kappa_g = 0$, which leads to the following differential equations:

$$\frac{d^2u}{ds^2} + \Gamma^1_{11}\left(\frac{du}{ds}\right)^2 + 2\Gamma^1_{12}\frac{du}{ds}\frac{dv}{ds} + \Gamma^1_{22}\left(\frac{dv}{ds}\right)^2 = 0 \ , \tag{68}$$

$$\frac{d^2v}{ds^2} + \Gamma^2_{11}\left(\frac{du}{ds}\right)^2 + 2\Gamma^2_{12}\frac{du}{ds}\frac{dv}{ds} + \Gamma^2_{22}\left(\frac{dv}{ds}\right)^2 = 0 \ , \tag{69}$$

where Γ^i_{jk} $(i,j,k = 1,2)$ are the Christoffel symbols, which are defined as follows [26]:

$$\begin{aligned}
\Gamma^1_{11} &= \frac{GE_u - 2FF_u + FE_v}{2(EG-F^2)}, & \Gamma^2_{11} &= \frac{2EF_u - EE_v + FE_u}{2(EG-F^2)} \ , \\
\Gamma^1_{12} &= \frac{GE_v - FG_u}{2(EG-F^2)}, & \Gamma^2_{12} &= \frac{EG_u - FE_v}{2(EG-F^2)} \ , \\
\Gamma^1_{22} &= \frac{2GF_v - GG_u - FG_v}{2(EG-F^2)}, & \Gamma^2_{22} &= \frac{EG_v - 2FF_v + FG_u}{2(EG-F^2)} \ .
\end{aligned} \tag{70}$$

These two second order differential equations can be rewritten as a system of four first order differential equations [13]:

$$\frac{du}{ds} = p \ , \tag{71}$$

$$\frac{dv}{ds} = q \ , \tag{72}$$

$$\frac{dp}{ds} = -\Gamma^1_{11}p^2 - 2\Gamma^1_{12}pq - \Gamma^1_{22}q^2 \ , \tag{73}$$

$$\frac{dq}{ds} = -\Gamma^2_{11}p^2 - 2\Gamma^2_{12}pq - \Gamma^2_{22}q^2 \ . \tag{74}$$

6.2 Two point boundary value problem

The system of four first order ordinary differential equations (71) to (74) can be solved as an initial value problem (IVP), in which all four boundary conditions are given at one point, or as a boundary value problem (BVP), in which four boundary conditions are specified at two distinct points. Most of the problems that arise in the application of geodesics are not IVPs but rather, are BVPs, which are much more difficult to solve. It is well known that the solution of an IVP is unique, while for a BVP, it is possible for the differential equations to have many solutions, or even no solution [11,19,24].

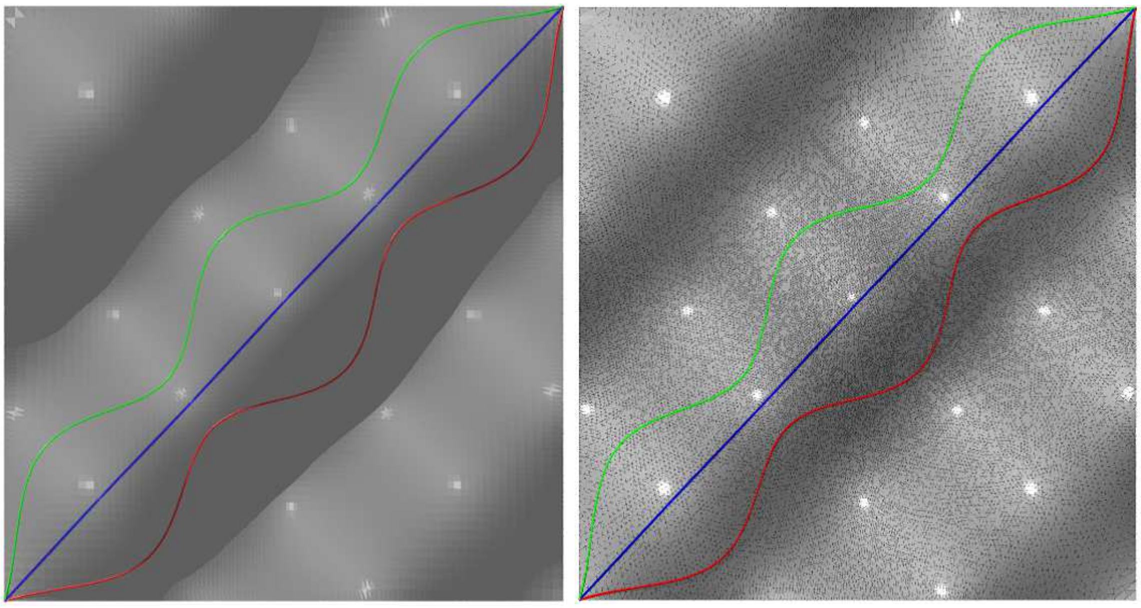


Fig. 11 Geodesic paths on a wave-like surface between the points of two corners (on the same surface used in Figure 5). (a) Bicubic B-spline surface. (b) Triangulated mesh of the B-spline surface. (53K triangles)

Table 2 Numerical results for the computation of the geodesic path between corner points of a wave-like surface. The number of mesh points used in the relaxation method is 500.

	B-spline		Mesh	
	Distance	Time (s)	Distance	Time (s)
Left	1.661	0.55	1.661	7.82
Middle	1.865	0.03	1.867	0.52
Right	1.661	0.58	1.662	4.25

There are two commonly used approaches to the numerical solution of BVPs, namely the relaxation method and the shooting method. The relaxation method begins with an initial guess and improves the solution iteratively. The relaxation method [25,15] begins by first discretizing the governing equations by means of the finite differences on a mesh with m points. The computation begins with an initial guess and improves the solution iteratively, or in other words, “relaxes” to the true solution. The computation of geodesic paths on a wave-like surface between the points of two corners is illustrated in Figure 11. It is easy to see that the computational results are almost identical for a B-spline surface (a) and for its tessellated surface (b). The initial guess is obtained based on the circular arc approximation [15,24]. There are three geodesic paths that satisfy the geodesic equations (71) to (74). The middle geodesic path is not a minimal path, while the other two paths are the shortest path due to symmetry. The computational results, such as the geodesic distances and the computational time, are listed in Table 2.

The governing idea of the shooting method is that if all values of y(s) are known at s = A, then the problem can be reduced to an IVP. However, y(A) can be found only by solving the problem. Therefore, we assume values at s = A, which are not given as boundary conditions at s = A, and then compute the solution of the resulting IVP to s = B. We assume a value for p_A and solve the differential equation as an IVP. Using (61), q_A can be obtained as follows:

$$q_A = \frac{-Fp_A \pm\sqrt{F^2p_A^2 - G(Ep_A^2 - 1)}}{G}, \tag{75}$$

We also assume the entire arc length of the geodesic paths in order to terminate the integration. The computed values of y(B) will not, in general, agree with the corresponding boundary condition at s = B. Consequently, we need to adjust the initial values for p_A and s and try again. This process is repeated until the computed values at the final point agree with the boundary conditions, using Newton's method. However, it is well known that unlike the relaxation method, the shooting method is often quite sensitive to the unknown initial values at point A [24].

When s = A and s = B are located in different coordinate systems, it may be difficult to apply the relaxation method, since it is based on the finite difference method. The shooting method, on the other hand, which is based on IVP, can be easily applied to these situations, which are similar to the computation of the lines of curvature. Such a case is depicted in Figure 12.

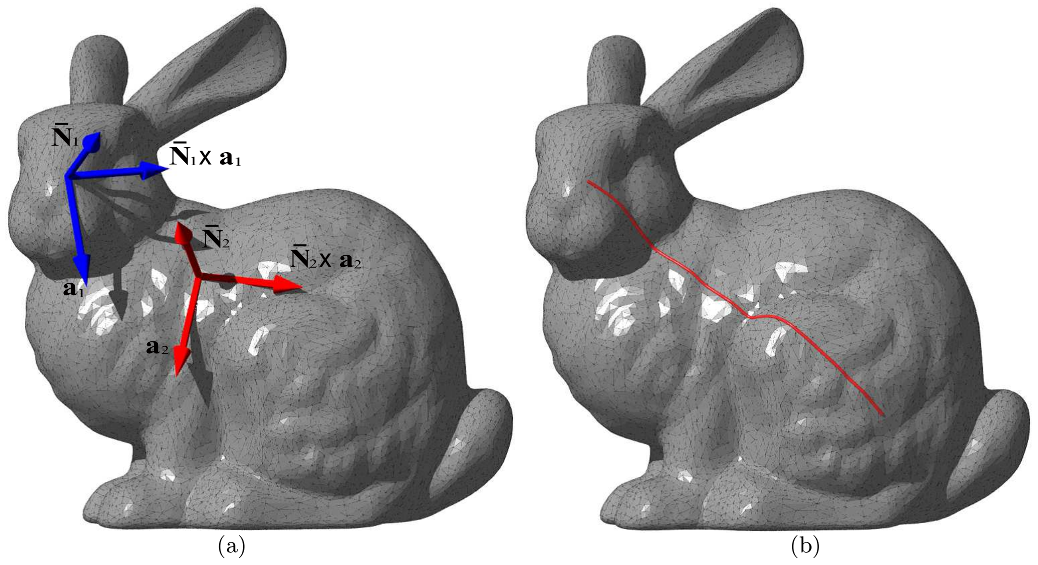

$\bar{\mathbf{N}}_1$
$\bar{\mathbf{N}}_1 \times \mathbf{a}_1$
$\bar{\mathbf{N}}_2$
$\bar{\mathbf{N}}_2 \times \mathbf{a}_2$
$\mathbf{a}_1$
$\mathbf{a}_2$
(a)
(b)

Fig. 12 Computation of geodesics on the Stanford bunny model (18K triangles): (a) Two different coordinate systems ($\mathbf{a}_1$, $\bar{\mathbf{N}}_1 \times \mathbf{a}_1$, $\bar{\mathbf{N}}_1$) and ($\mathbf{a}_2$, $\bar{\mathbf{N}}_2 \times \mathbf{a}_2$, $\bar{\mathbf{N}}_2$).(b) Computation of geodesic path using the shooting method, in which two different coordinate systems are used.

We have compared the computation of the geodesic path using our algorithm with that of the ICH2, which is the improved Chen and Han's algorithm, developed by Xin and Wang [31], for the wave-like surface. Figure 13 and Table 3 indicate that ICH2 is very fast; however, it can find only one shortest path, even if there is another shortest path that arises from the symmetry of the surface. On the other hand, our method finds three geodesic paths that satisfy the geometric equations (for 9K and 5K mesh), two of which provide the shortest path. However, for very coarse mesh (0.5K), it can only find the right side geodesic path. Furthermore, Table 3 shows that as the mesh becomes coarser, the length of the path computed by ICH2 becomes shorter than the actual length; in contrast, our method computes the length of the path very accurately.

Table 3 Comparison between ICH2 and our method for wave-like surface. The computational time of our method includes only the right side geodesic path.

	9K Triangles		5K Triangles		0.5K Triangles	
	Distance	Time (s)	Distance	Time (s)	Distance	Time (s)
ICH2	1.650	0.156	1.627	0.047	1.592	0.002
Our method	1.661	1.016	1.662	0.468	1.671	0.579

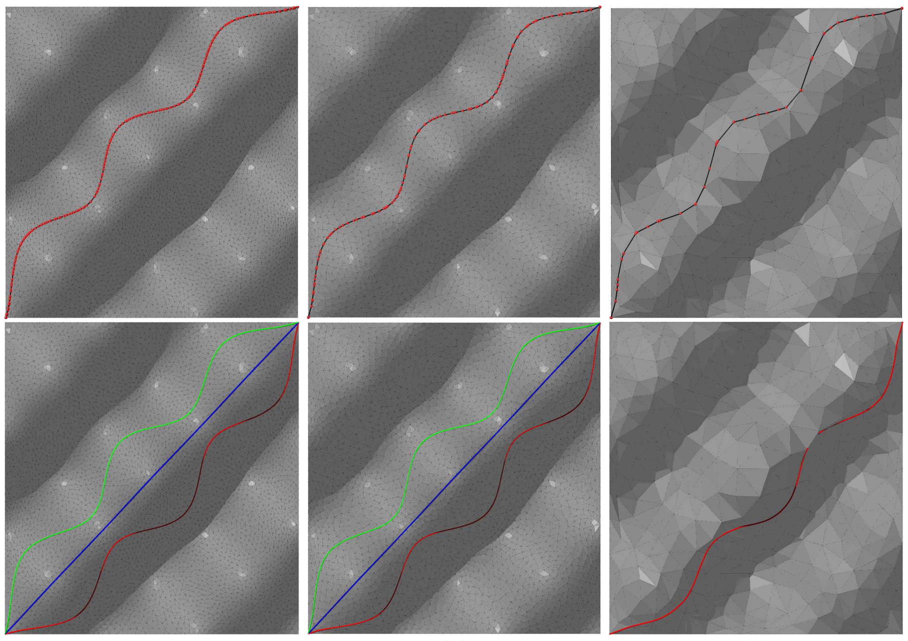


Fig. 13 Numerical results for computation of geodesic path between corner points of wave-like surface. Top row: Results of ICH2. Bottom row: Results of our method. (a) High-quality 9K mesh. (b) High-quality 5K mesh. (c) High-quality 0.5K mesh.

6.3 Geodesic offsets

In this section we focus on geodesic offsets, which are different from the classical definition of offset. Geodesic offsets, also known as geodesic parallels, are well known in classical differential geometry. Traditionally, the spacing between adjacent tool paths, which is referred to as the side-step or pick-feed, is kept constant in either the Euclidean space or in the parameter space. Geodesic offset curves have been used to generate tool paths on a part for zig-zag finishing that uses 3-axis NC machining with a ball-end cutter, so that the scallop-height, which is the cusp height of the material removed by the cutter becomes constant [16,24]. This leads to a significant reduction in the size of the cutter location data, and hence, in the machining time.

Let us consider an arbitrary curve C on a surface. The locus of points at a constant distance measured from curve C along the geodesic curve drawn orthogonal to C is called the geodesic offset (see Figure 14). Patrikalakis and Bardis [23] provided an algorithm to construct such geodesic offsets on NURBS surfaces. The equations of the geodesics consist of four first order nonlinear ordinary differential equations (71) to (74) that are solved as an initial value problem.

Let us consider a progenitor curve lying on a parametric surface $\mathbf{r} = \mathbf{r}(u, v)$ given by $\mathbf{r}^c(t) = \mathbf{r}(u^c(t), v^c(t))$ and an arc length parametrized geodesic curve $\mathbf{r}^g(s) = \mathbf{r}(u^g(s), v^g(s))$ orthogonal to $\mathbf{r}^c$. We select n points on the progenitor curve t_i, $0 \le i \le n-1$, and compute a geodesic path for each point by a distance equal to d_g as an IVP. The initial direction $\mathbf{t}^g = (\frac{du^g}{ds}, \frac{dv^g}{ds}) = (p, q)$ can be determined by the condition that the tangent vector along the progenitor curve $\dot{\mathbf{r}}^c$ and the unit tangent vector of the geodesic curve $\mathbf{t}^g$ are orthogonal

$$(\dot{u}^c E + \dot{v}^c F)p + (\dot{u}^c F + \dot{v}^c G)q = 0 \ , \tag{76}$$

and by the normalization condition,

$$E(p)^2 + 2Fpq + G(q)^2 = 1 \ , \tag{77}$$

which leads to

$$p = \pm \frac{\omega_2}{\sqrt{E\omega_2^2 - 2F\omega_1\omega_2 + G\omega_1^2}} \ , \tag{78}$$

$$q = \mp \frac{\omega_1}{\sqrt{E\omega_2^2 - 2F\omega_1\omega_2 + G\omega_1^2}} , \tag{79}$$

where $\omega_1 = \dot{u}^c E + \dot{v}^c F$ and $\omega_2 = \dot{u}^c F + \dot{v}^c G$ [23,24]. The positive and negative signs in (78) and (79) correspond to the two possible directions of the geodesic path relative to the progenitor curve. The terminal points of the geodesic paths, which depart orthogonally from n selected points of the progenitor curve on the surface, are interpolated in the surface patch parameter space by a B-spline curve, thus ensuring that the offset curve lies entirely on the surface.

The geodesic offsets on the tessellated surface used in Figure 6(b) are shown in Figure 14. The blue line represents the progenitor curve, whose pre-image is a parabola on which each geodesic offset distance is 0.667.

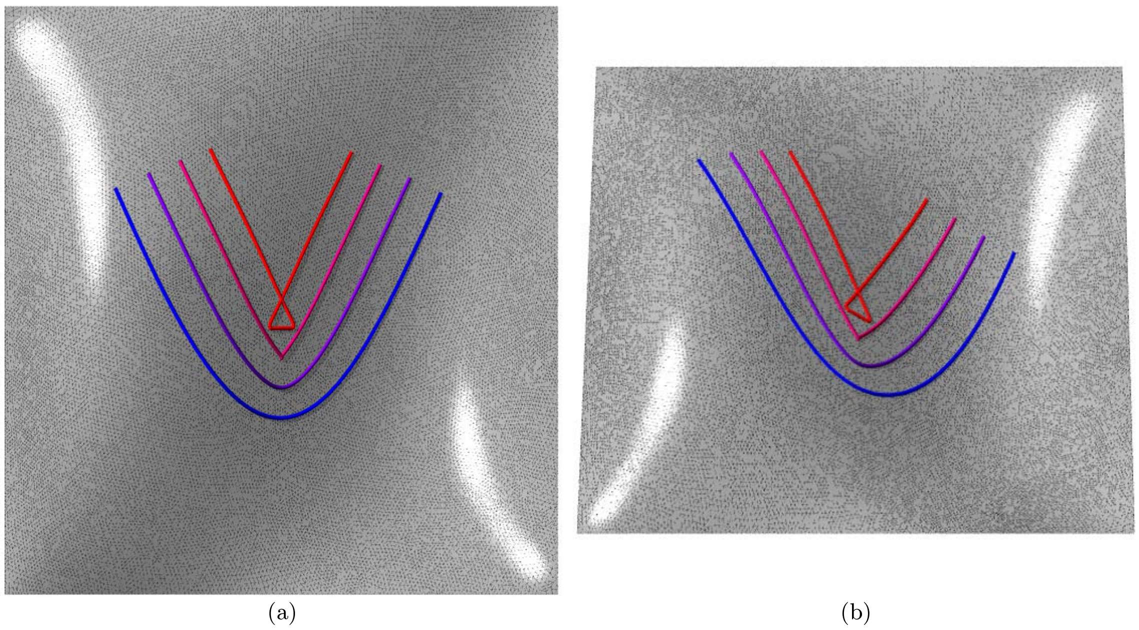


Fig. 14 Geodesic offsets on the tessellated surface used in Figure 6(b). The blue line represents the progenitor curve whose pre-image is a parabola. The cool colors indicate a small offset distance.

7 Conclusions

We have introduced a novel unifying technique that enables us to use the shape interrogation tools developed for B-spline surfaces for objects represented by triangular meshes. Without any change in the original algorithm, the newly proposed method allows us to compute various surface interrogation tools on a triangular mesh model as though the surface were a B-spline surface. Such tools include circular reflection lines, the computation of lines of curvature, geodesics, geodesic offsets, and the detection of the umbilics. We assume that the input mesh has been modified by remeshing software, which is readily available on the Internet, so that it becomes a high-quality mesh. It is observed that the density of the mesh does not affect the result as long as high-quality meshes are used for the computation. On the other hand, low-quality meshes cannot generate good results, regardless of the mesh density. Many illustrative examples have been given that show the effectiveness of the unifying techniques. We acknowledge that the computational cost of the current implementation is relatively high.

Acknowledgements A portion of this work is supported by Komatsu Ltd. The bunny model is courtesy of the Stanford University Computer Graphics Laboratory.

References

1. P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM Transactions on Graphics*, 22(3):485–493, 2003.
2. M. Attene and B. Falcidieno. ReMESH: An interactive environment to edit and repair triangle meshes *http://remesh.sourceforge.net/. In SMI 06: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006*, pages 271–276, 2006.
3. J. Beck, R. Farouki, and J. Hinds. Surface analysis methods. *IEEE Computer Graphics and Applications*, 6(12):18–36, 1986.
4. P. Bourke. Conrec - a contouring subroutine. *Byte Magazine*, 12(8), 1987.
5. J. Chen and Y. Han. Shortest paths on a polyhedron. *Proceedings of the sixth annual symposium on computational geometry*, pages 360–369, 1990.
6. G. Dahlquist and Å. Björck. *Numerical Methods.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1974.
7. M. P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Inc., Upper Saddle River, NJ, 1976.
8. G. Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide, 5th Edition.* Morgan Kaufmann, 2002.
9. S. Hahmann, A. Belyaev, L. Busé, G. Elber, B. Mourrain, and C. Rössl. Shape interrogation - A state of the art. In *Shape Analysis and Structuring*, Mathematics and Visualization, pages 1–57. Springer Verlag, 2008.
10. E. Kalogerakis, D. Nowrouzezahrai, P. Simari, and K. Singh. Extracting lines of curvature from noisy point clouds. *Computer-Aided Design*, 41(4):282–292, 2009.
11. H. B. Keller. *Numerical Methods for Two-Point Boundary Value Problems.* Blaisdell, Waltham, MA, 1968.
12. R. Klass. Correction of local surface irregularities using reflection lines. *Computer-Aided Design*, 12(2):73–77, 1980.
13. M. M. Lipschutz. *Theory and Problems of Differential Geometry.* Schaum's Outline Series: McGraw-Hill, 1969.
14. W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, 1987.
15. T. Maekawa. Computation of shortest paths on free-form parametric surfaces. *Journal of Mechanical Design, ASME Transactions*, 118(4):499–508, 1996.
16. T. Maekawa. An overview of offset curves and surfaces. *Computer-Aided Design*, 31(3):165–173, 1999.
17. T. Maekawa, Y. Nishimura, and T. Sasaki. Circular highlight/reflection lines. *Computer-Aided Design and Applications*, 2(1-4):291–300, 2005.
18. T. Maekawa and N. M. Patrikalakis. Interrogation of differential geometry properties for design and manufacture. *The Visual Computer*, 10(4):216–237, 1994.
19. T. Maekawa, F. E. Wolter, and N. M. Patrikalakis. Umbilics and lines of curvature for shape interrogation. *Computer Aided Geometric Design*, 13(2):133–161, 1996.
20. M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, pages 35–57, 2003.
21. J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM Journal on Computation*, 16(4):647–668, 1987.
22. Y. Nishiyama, Y. Nishimura, T. Sasaki, and T. Maekawa. Surface faring using circular highlight lines. *Computer-Aided Design and Applications*, 4 (1-4):405–414, 2007.
23. N. M. Patrikalakis and L. Bardis. Offsets of curves on rational B-spline surfaces. *Engineering with Computers*, 5(1):39–46, 1989.
24. N. M. Patrikalakis and T. Maekawa. *Shape Interrogation for Computer Aided Design and Manufacturing.* Springer-Verlag, Heidelberg, 2002.
25. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.
26. D. J. Struik. *Lectures on Classical Differential Geometry.* Addison-Wesley, Cambridge, MA, 1950.
27. V. Surazhsky and C. Gotsman. Explicit surface remeshing. *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 20–30, 2003.
28. G. Thürmer and C. A. Wüthrich. Computing vertex normals form polygonal facets. *Journal of Graphics, Gpu, and Game Tools*, 3(1):43–46, 1998.
29. A. Vlachos, J. Peters, C. Boyd, and J. L. Mitchell. Curved PN triangles. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 159–166, 2001.
30. E. W. Weisstein. "Triangle Interior." from MathWorld – A Wolfram Web Resource. *http://mathworld.wolfram.com/TriangleInterior.html.*
31. S. Q. Xin and G. J. Wang. Improving Chen & Han's algorithm on the discrete geodesic problem. *ACM Transactions on Graphics*, 28(4):1–8, 2009.
32. J. Yong, B. Deng, F. Cheng, B. Wang, K. Wu, and H. Gu. Removing local irregularities of triangular meshes with highlight line models. *Science in China Series F: Information Sciences*, 52(3):418–430, 2009.