ELSEVIER

Computer-Aided Design 35 (2003) 881–892

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Dynamic highlight line generation for locally deforming NURBS surfaces

Jun-Hai Yong[a,*], Fuhua (Frank) Cheng[a], Yifan Chen[b], Paul Stewart[b], Kenjiro T. Miura[c]

[a]*Laboratory of Graphics and Geometric Modeling, Department of Computer Science University of Kentucky, Lexington, KY 40506-0046, USA*
[b]*Ford Research Lab, Dearborn, MI 48121, USA*
[c]*Laboratory of Realistic Modeling, Department of Mechanical Engineering, Shizuoka University, Johoku, Hamamatsu 432-8561, Japan*

Received 12 December 2001; received in revised form 23 September 2002; accepted 24 September 2002

Abstract

The highlight line model is a powerful tool in assessing the quality of a surface. Efficient highlight line generation is especially important for an interactive design environment. In this paper, a method for dynamic generation of highlight lines on a locally deforming NURBS surface is presented. The method generates frames of the deforming surface and the corresponding highlight lines by directly modifying the current highlight lines using a Taylor expansion technique, instead of going through a tracing process. The highlight lines computation process adopted here enables a unified distance surface to generate all highlight lines in the highlight line family. The computation process is facilitated by looking up pre-calculated information of the tessellation mesh and an indexing technique for the distance surface. The indexing technique is presented to determine when the highlight line model should be re-generated and, to facilitate the highlight line re-generation process. The new technique is suitable for interactive design environments and animation applications as the updating process takes only one subtraction and one vector inner product to get the new parameters for each new node.

Keywords: NURBS surfaces; Highlight lines; Shape modification; Deformation

1. Introduction

The design of a free-form surface is a highly iterative process. In some cases, the shape of a surface has to be manipulated and analyzed many times before it can be used for downstream applications. Commonly used measures for analyzing the shape of a surface include *Gaussian* and *mean curvatures* as well as *normal curvatures* along given field of directions. *Isophotes* [17], *reflection lines* [6,7] and, more recently, *highlight lines* [1,2] have also been used in assessing the quality of a surface. These techniques proved to be more effective in that they are more intuitive to understand and easier to compute. The highlight line model, introduced by Beier and Chen [1,2], is a simplified version of the reflection line model. Like the reflection line model, the highlight line model is sensitive to the change of normal directions and, therefore, can be used to detect surface normal (curvature) irregularities. This sometimes is not possible with wireframe drawings or shaded images [23]. As far as highlight line generation for an moving object is concerned, no work has been reported in the literature yet. Such a work requires the consideration of dynamic features of the moving object in the rendering process. We address this problem in this paper by considering those features in the highlight line generation process for a locally deforming *non-uniform rational B-spline* (NURBS) surface, a function strongly needed in an interactive design environment.

There is a strong demand for efficient, dynamic highlight line generation on the graphics side as well. With the rapid, continuing advancement of processing power, programmers and artists now have more freedom than ever to include such features as highlight lines in graphics. Highlight lines can now walk hand in hand with shadows, adding unprecedented realism. Highlight lines on free form surfaces and less conventionally shaped objects cue the viewer to attributes such as surface composition and texture, underlying structure, and the grade of curves. This can prove to be invaluable in concept designs since highlight lines can help communicate form more precisely [20]. The use of highlight lines can

* Corresponding author. Current address: School of Software, Tsinghua University, Beijing 100084, People's Republic of China.
E-mail address: junhai@csr.uky.edu (J.-H. Yong).

0010-4485/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
doi:10.1016/S0010-4485(02)00165-3

therefore be expanded outside conventional industries such as automotive and aeronautical industries. Computer and video entertainment has already begun to harness the use of highlight lines in artistic rendering and 3D modeling. It is certainly necessary to streamline and accelerate the process of generating highlight lines so that they can be generated for complex objects and surfaces in a real-time environment.

In this paper, a dynamic highlight line generation technique for a locally deforming NURBS surface is presented. The real-time effect is achieved by directly modifying the highlight lines using a *Taylor expansion approach*, instead of going through a *tracing process*. The highlight line computation process adopted in this paper enables a unified *distance surface* to generate all the highlight lines in the highlight line family. The computation process is also facilitated by looking up pre-calculated information of the tessellation mesh and an *indexing technique* for the distance surface. The new technique updates the NURBS surface and the highlight lines both in real-time and, consequently, is suitable for an interactive design environment.

The remaining part of the paper is arranged as follows. The problem to be studied and related basic properties are presented in Section 2. A computation technique that allows the generation of all the highlight lines in the highlight line family using a unified distance surface function is also introduced there. The basic idea of the dynamic updating process, including updating of the NURBS surface and updating of the highlight lines, is presented in Section 3. A number of important issues related to the re-generation process of a highlight line model are discussed in Section 4. Examples showing the effectiveness of the new method are shown in Section 5. The concluding remarks are given in Section 6.

2. Problem formulation and basic properties

The problem to be studied in this paper can be described as follows: given a NURBS surface $\mathbf{S}(u,v)$ and a set of coplanar parallel linear light sources, construct a shaded image of the surface and the corresponding highlight lines, and then generates frames of the surface and the highlight lines in real-time when the surface is locally deformed (with the linear light sources fixed). The deformation is performed through a dynamic control point manipulation process. One control point is dynamically manipulated each time.

2.1. NURBS surface and rendering

A NURBS surface $\mathbf{S}(u,v)$ of degree $p \times q$ is defined as follows

$$\mathbf{S}(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} w_{i,j}\mathbf{P}_{i,j}N_{i,p}(u)N_{j,q}(v)}{\sum_{i=0}^{m}\sum_{j=0}^{n} w_{i,j}N_{i,p}(u)N_{j,q}(v)} \tag{1}$$

where $\mathbf{P}_{i,j}$ are control points, $w_{i,j}$ are weights, $N_{i,p}(u)$ are B-spline basis functions of degree p defined by a u parameter knot vector $\{u_i | 0 \le i \le m+p+1\}$, and $N_{j,q}(v)$ are B-spline basis functions of degree q defined by a v parameter knot vector $\{v_j | 0 \le j \le n+q+1\}$. The parameter space of $\mathbf{S}(u,v)$ is $[u_p, u_{m+1}] \times [v_q, v_{n+1}]$.

NURBS surfaces are widely used in CAD/CAM and computer animation. Their properties have been extensively studied (see, e.g. [16]). Rendering algorithms for NURBS surfaces have been developed using *point sampling approach* [11,18,21], *ray tracing approach* [15,22], and *tessellation approach* [12,13,19]. Tessellation approach is more popular in that it can take advantage of specialized hardware to perform illumination calculations in parallel with the surface tessellation [19]. We follow the approach presented in Ref. [13] (details can be found in Ref. [14] as well) to render a NURBS surface in this paper. The parameter space of a NURBS surface is divided into a *grid* of rectangles first. The size of each rectangle is determined by the approximation criterion presented in Ref. [4]. Surface coordinates $\mathbf{S}(u,v)$ and partial derivatives $\mathbf{S}_u(u,v)$ and $\mathbf{S}_v(u,v)$ at the *grid vertices* (i.e. vertices of rectangles) are calculated by a two-stage Cox-de Boor technique [13] (see also Ref. [14]). The normal vectors $\mathbf{N}(u,v) = \mathbf{S}_u(u,v)$ $\mathbf{S}_v(u,v)$ are then calculated. These values are stored in a lookup table. Thus, the NURBS surface is tessellated into a set of quadrilaterals, which can be readily divided into triangles. The resulting tessellants are scan converted using a standard method.

2.2. Highlight line model

A *highlight line* is the imprint of a linear light source on a surface [1,2]. If an array of coplanar parallel linear light sources are used, the imprint is a family of highlight lines, called a *highlight line model* (see Fig. 5 for two examples). The imprint of a linear light source is a set of surface points for which the perpendicular distance between the surface normal and the light source is zero. More precisely, let $\mathbf{L}(t)$ be the parametric representation of a linear light source

$$\mathbf{L}(t) = \mathbf{A} + t\mathbf{H}, \qquad t \in R, \tag{2}$$

where $\mathbf{A}$ is a point of $\mathbf{L}(t)$ and $\mathbf{H}$ is a vector defining the direction of $\mathbf{L}(t)$. For a given surface point $\mathbf{S}(u,v)$, the perpendicular distance between the normal vector $\mathbf{N}(u,v)$ and the linear light source is (shown in Fig. 1):

$$d(u,v) \equiv \frac{|[\mathbf{H} \times \mathbf{N}(u,v)] \cdot [\mathbf{S}(u,v) - \mathbf{A}]|}{\|\mathbf{H} \times \mathbf{N}(u,v)\|}. \tag{3}$$

A point $\mathbf{S}(u,v)$ belongs to the highlight line iff (if and only if) $d(u,v) = 0$. If a more realistic light source is considered, i.e. if the light source is a linear cylinder with radius $r > 0$, then the corresponding imprint on the surface is a highlight band [1,2]. A point $\mathbf{S}(u,v)$ of the surface belongs to a highlight band iff the perpendicular distance $d(u,v)$ between

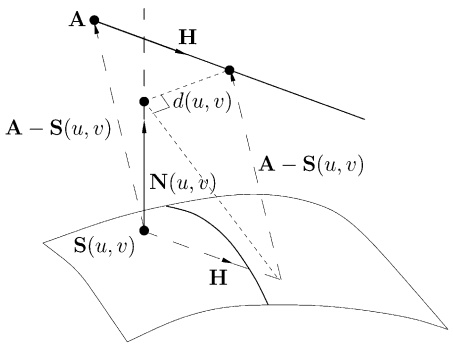


Fig. 1. An example of $d(u,v)$.

its normal $\mathbf{N}(u,v)$ and the axis of the cylinder is no larger than r.

By viewing the distance function (3) as a surface defined on the parameter space of $\mathbf{S}(u,v)$, the creation of a highlight line can be formulated as a surface-plane intersection problem, and solved using a surface-contouring technique [1,2]. This is done by intersecting the *distance surface* $d(u,v)$ with a zero plane (i.e. plane $d=0$) and mapping the resulting parametric curve on the surface $\mathbf{S}(u,v)$. For a highlight line model, a straightforward application of this technique requires the construction of a distance surface

$$d_i(u,v) \equiv \frac{|[\mathbf{H}\times\mathbf{N}(u,v)]\cdot[\mathbf{S}(u,v)-\mathbf{A}_i]|}{\|\mathbf{H}\times\mathbf{N}(u,v)\|} \tag{4}$$

for each light source $\mathbf{L}_i(t)=\mathbf{A}_i+t\mathbf{H}$. This is a very expensive approach. An elegant alternative proposed by Beier and Chen [1,2] is to approximate highlight lines with boundary curves of highlight bands. This approach replaces the expensive highlight line generation process with the process of intersecting a single (signed) distance surface $D_{bc}(u,v)$ with a sequence of planes $d=ic$, where $c>0$ is the distance between each pair of consecutive light sources. The distance surface is given by

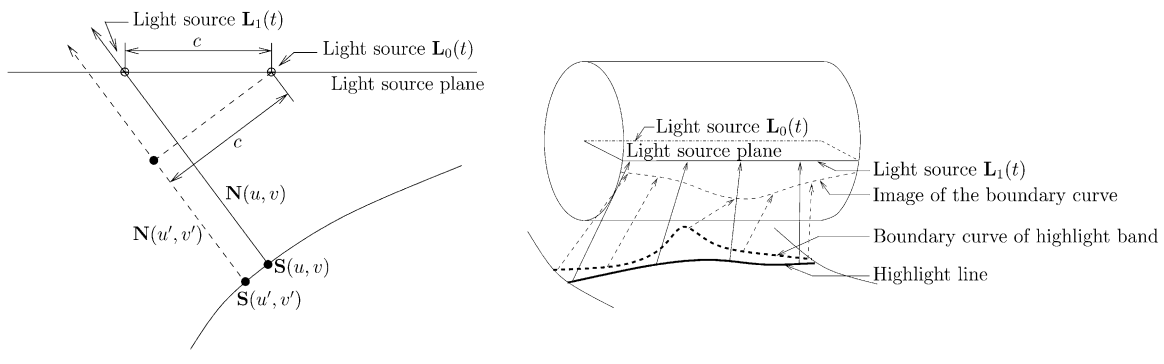
$$D_{bc}(u,v) = \frac{[\mathbf{H}\times\mathbf{N}(u,v)]\cdot[\mathbf{S}(u,v)-\mathbf{A}_0]}{\|\mathbf{H}\times\mathbf{N}(u,v)\|} \tag{5}$$

where $\mathbf{A}_0$ is a point on the *base light source* $\mathbf{L}_0(t)=\mathbf{A}_0+t\mathbf{H}$. This base light source is used as the axis of the light cylinders.

As shown in Fig. 2(a), a point $\mathbf{S}(u',v')$ on a boundary curve of a highlight band is usually a displacement of a point $\mathbf{S}(u,v)$ on a highlight line. For regions relatively flat, the displacement usually is quite uniform. But for regions with uneven curvature distribution, it is possible to obtain strange boundary curves while the exact highlight lines are actually quite normal. See the image of the boundary curve of a highlight band in Fig. 2(b) for the case $i=1$. (The cylinder in Fig. 2(b) is a light cylinder of radius c. The NURBS surface considered is somewhere below the light cylinder. The *image of the boundary curve* is the intersection curve of the normals of the boundary curve with the light cylinder.) The situation becomes worse when it is farther away from the base light source $\mathbf{L}_0(t)$.

Another problem with the above approach is that it is base light source dependent. If a different light source is used as the base light source, one might get different, sometimes dramatically different, boundary curves for the highlight bands. An example is shown in Fig. 3 where the surface is a bi-quadratic NURBS surface defined by the control points: (−20, −40,0), (0, −40,0), (20, −40,0), (−20,0,0), (0,0,100), (20,0,0), (−20,40,0), (0,40,0) and (20,40,0) with all weights being 1. The knot vectors of the surface are $\mathbf{U}=\mathbf{V}=\{0,0,0,1,1,1\}$. The light sources have direction (0,1,0), and are contained in the plane defined by the normal vector (0,0,1) and the point (0,0,26). The given NURBS surface is a quite smooth Bézier surface (Fig. 3(a)). However, when a different base light source is used, a loop is generated in the boundary curve set of the highlight bands (see the purple loop in Fig. 3(b)) and this irregular pattern certainly would lead the designer to doubt if there is an irregular region in the surface [3,23].

Therefore, a new approach that would inherit the advantage of Beier and Chen's approach, i.e. computing the highlight lines as the process of intersecting a single distance surface with a set of planes, but without the above problems is needed. In this paper, this is achieved by



(a) case I (b) case II

Fig. 2. Comparison of distance computation.

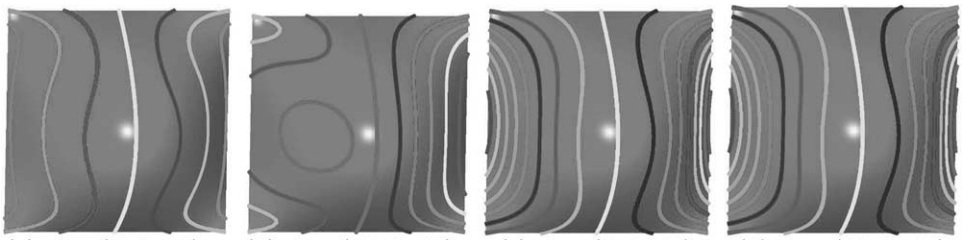


Fig. 3. Comparison of boundary curves of highlight bands (a) and (b) and boundary curves of highlight strips (c) and (d) while different base light sources are used.

replacing the light source cylinder with a *light source strip*, which is a slice of the light source cylinder on the light source plane. And, a new distance surface $D(u,v)$ is needed to substitute $D_{bc}(u,v)$. Mathematically, it is given by

$$D(u,v) = \frac{[\mathbf{H}\times\mathbf{N}(u,v)]\cdot[\mathbf{S}(u,v) - \mathbf{A}_0]}{\mathbf{X}\cdot[\mathbf{H}\times\mathbf{N}(u,v)]}, \tag{6}$$

where $\mathbf{X} = \mathbf{H}\times\mathbf{Z}$ and $\mathbf{Z}$ is the normal vector of the light source plane. Here, $\mathbf{H}$, $\mathbf{Z}$ and $\mathbf{N}(u,v)$ are normalized. Geometrically, it is the distance between the normal vector $\mathbf{N}(u,v)$ and the base light source $\mathbf{L}_0(t)$ along $\mathbf{X}$ direction. The relationship between $D(u,v)$ and $D_{bc}(u,v)$ is

$$D(u,v)\cos\theta = D_{bc}(u,v), \qquad \text{where } \cos\theta = \mathbf{X}\frac{[\mathbf{H}\times\mathbf{N}(u,v)]}{\|\mathbf{H}\times\mathbf{N}(u,v)\|},$$

which is shown in Fig. 4. Applying the principle $\mathbf{E}\cdot(\mathbf{F}\times\mathbf{G}) = (\mathbf{E}\times\mathbf{F})\cdot\mathbf{G}$ for any vectors $\mathbf{E}$, $\mathbf{F}$ and $\mathbf{G}$, Eq. (6) is simplified as

$$D(u,v) = \frac{[\mathbf{H}\times\mathbf{N}(u,v)]\cdot[\mathbf{S}(u,v) - \mathbf{A}_0]}{\mathbf{Z}\cdot\mathbf{N}(u,v)}. \tag{7}$$

For convenience of presentation, the imprint of a light source strip will be called a *highlight strip*. A point is in the ith highlight strip iff $-ic \le D(u,v) \le ic$ at that point. The left and right boundary curves of the ith highlight strip are given by

$$\{\mathbf{S}(u,v) | D(u,v) = -ic\} \text{ and } \{\mathbf{S}(u,v) | D(u,v) = ic\},$$

respectively. Assuming that there are no zero denominators in Eqs. (4) and (7), we have the following theorem.

Theorem 1. *$d_i(u,v) = 0$ if and only if $D(u,v) = ic$.*

The geometric meaning of this theorem is: the boundary curves of the highlight strips are exactly the highlight lines of the given light sources and they can also be obtained by intersecting a single distance surface (7) with a set of planes ($D = ic$).

It should be pointed out that this new approach is base light source independent, i.e. highlight line models generated using distance surface (7) are exactly the same no matter which light source is used as the base light source.

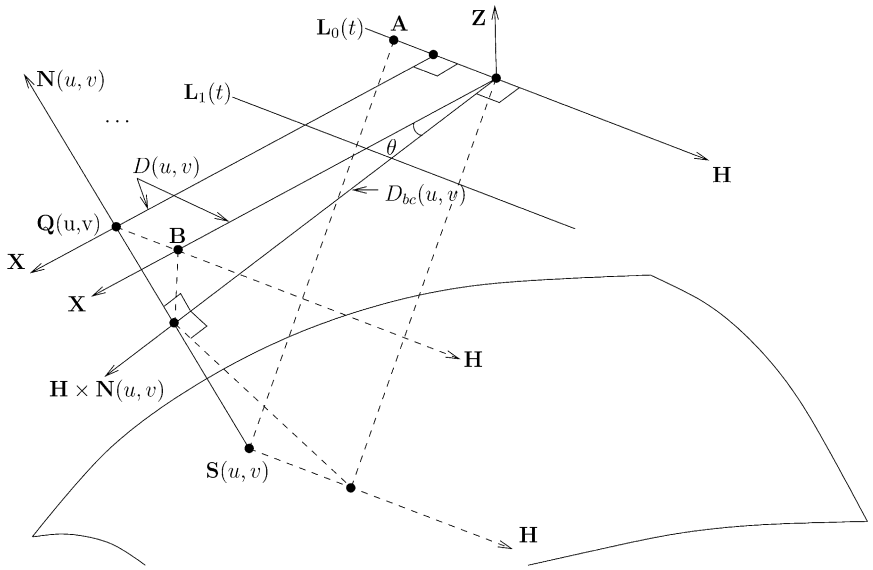


Fig. 4. An example of with $D(u,v)$, and normalized.

An example is shown in Fig. 3(c) and (d), where the surface considered is the same as the one used in Fig. 3(a) and (b).

As far as rendering is concerned, highlight lines are usually represented as polylines with a specified width and painted in different colors to different light sources. The values of $\mathbf{S}(u,v)$ and $\mathbf{N}(u,v)$ at the highlight line nodes (i.e. vertices of those polylines) are computed using linear interpolation from corresponding values at the vertices of the NURBS surface tessellation mesh. These polylines and the tessellation mesh, together with the $\mathbf{S}(u,v)$ and $\mathbf{N}(u,v)$ values of their vertices, are then sent to a standard scan conversion processor for rendering.

3. Dynamic frame generation

The generation of each frame consists of two parts: re-evaluate the surface and the highlight lines, and then re-draw them. The bottom line is to avoid unnecessary computation and redrawing process as much as possible. For a bicubic NURBS surface, the movement of a control point would affect the shape of 16 patches only. Therefore, only those 16 patches should be re-evaluated to get their new shape, and only the highlight lines of those 16 patches should be re-evaluated to get their new locations. However, this does not mean that only those 16 patches should be redrawn. A blocked portion of a remaining patch could become visible and a visible portion of a remaining patch could become invisible because of the movement of those 16 patches. Therefore, even though the remaining patches, including the corresponding highlight lines, do not require re-evaluation in the local deformation process, they should still be redrawn in each frame to keep the displayed information correct.

3.1. NURBS surface re-evaluation

Several works have been done in dynamic re-evaluation of NURBS surfaces. A brief survey can be found in Refs. [8–10] where an incremental method is also presented. The dynamic rendering process of this paper is different from that technique in that both the surface coordinates and the normal vectors at the grid vertices are computed here, an approach more suitable with standard rendering software such as OpenGL.

Let $\mathbf{P}_{k,l}$, $0 \le k \le m$ and $0 \le l \le n$, be the control point that is being modified. Let $\bar{\mathbf{P}}_{k,l}$ be the new location of $\mathbf{P}_{k,l}$ and $\bar{\mathbf{S}}(u,v)$ be the resulting new surface. $\bar{\mathbf{S}}(u,v)$ is different from $\mathbf{S}(u,v)$ in the following parametric region only

$$(u_{\max\{k,p\}},u_{\min\{m+1,k+p+1\}})\times(v_{\max\{l,q\}},v_{\min\{n+1,l+q+1\}}) \quad (8)$$

Therefore, we only need to compute new values of $\mathbf{S}(u,v)$, $\mathbf{S}_u(u,v)$ and, $\mathbf{S}_v(u,v)$ at grid vertices that lie in the above parametric region.

Let $\mathbf{V}_{k,l}$ denote the difference between $\bar{\mathbf{P}}_{k,l}$ and $\mathbf{P}_{k,l}$,

$$\mathbf{V}_{k,l} = \bar{\mathbf{P}}_{k,l} - \mathbf{P}_{k,l}. \quad (9)$$

It is easy to see that $\mathbf{S}(u,v)$, $\mathbf{S}_u(u,v)$ and, $\mathbf{S}_v(u,v)$ can be expressed as

$$\bar{\mathbf{S}}(u,v) = \mathbf{S}(u,v) + R_{k,l}(u,v)\mathbf{V}_{,l}, \quad (10)$$

$$\bar{\mathbf{S}}_u(u,v) = \mathbf{S}_u(u,v) + E_{k,l}(u,v)\mathbf{V}_{k,l} \quad (11)$$

and

$$\bar{\mathbf{S}}_v(u,v) = \mathbf{S}_v(u,v) + F_{k,l}(u,v)\mathbf{V}_{k,l}, \quad (12)$$

where

$$R_{k,l}(u,v) = \frac{N_{k,p}(u)N_{l,q}(v)w_{k,l}}{W(u,v)}, \quad (13)$$

$$E_{k,l}(u,v) = \frac{N'_{k,p}(u)N_{l,q}(v)w_{k,l} - W_u(u,v)R_{k,l}(u,v)}{W(u,v)}, \quad (14)$$

$$F_{k,l}(u,v) = \frac{N_{k,p}(u)N'_{l,q}(v)w_{k,l} - W_v(u,v)R_{k,l}(u,v)}{W(u,v)} \quad (15)$$

and

$$W(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}.$$

Eq. (10) was first used by Li et al. in Refs. [8–10] to render deforming NURBS surfaces. $R_{k,l}(u,v)$, $E_{k,l}(u,v)$ and $F_{k,l}(u,v)$ are independent of the control points. Their values can be computed and stored in a lookup table before the deformation process starts. Thus, each of $\bar{\mathbf{S}}(u,v)$, $\bar{\mathbf{S}}_u(u,v)$ and $\bar{\mathbf{S}}_v(u,v)$ can be obtained using only one vector addition and one vector multiplication for each grid vertex in the parametric region Eq. (8). The new values of $\bar{\mathbf{S}}(u,v)$, $\bar{\mathbf{S}}_u(u,v)$ and $\bar{\mathbf{S}}_v(u,v)$ are put into the corresponding entries of the lookup table before sending the contents of the lookup table to the scan conversion processor.

3.2. Highlight line model re-evaluation

The idea here is to avoid calculating the distance surface $d(u,v)$ but to update the highlight line nodes directly. This is possible by working on the Taylor expansion of a simplified form of the distance surface $d(u,v)$.

Let (s,t) be the parameters of a highlight line node that needs to be re-evaluated. And let Eq. (2) be the corresponding light source. We have $d_i(s,t) = 0$ (see Eq. (4) for its definition). Let

$$\Psi(u,v) = [\mathbf{H}\times\mathbf{N}(u,v)]\cdot[\mathbf{A}_i - \mathbf{S}(u,v)].$$

$d_i(u,v) = 0$ is equivalent to

$$\Psi(u,v) = 0,$$

if $\|\mathbf{H}\times\mathbf{N}(u,v)\| \neq 0$. Hence, one can use $\Psi(u,v)$, instead of $d_i(u,v)$, to determine the highlight line for the light source Eq. (2) if the direction of the light source is not parallel to any normal vectors in the region defined in Eq. (8).

Let $(\bar{s},\bar{t})$ be the parameters of the highlight line node in the new frame, and let $\bar{\Psi}(u,v)$ be the numerator of the distance surface for the new surface $\mathbf{S}(u,v)$.

$$\bar{\Psi}(u,v) = [\mathbf{H}\times\bar{\mathbf{N}}(u,v)]\cdot[\mathbf{A}_i - \bar{\mathbf{S}}(u,v)].$$

By definition, $\bar{\mathbf{S}}(\bar{s},\bar{t})$ is a point of the highlight line on the new surface $\bar{\mathbf{S}}(u,v)$ corresponding to the light source (2). Hence, we have

$$\bar{\Psi}(\bar{s},\bar{t}) = 0.$$

Note that $\bar{\Psi}(u,v)$ is also a function of $\bar{\mathbf{P}}_{k,l}$. $\bar{\Psi}(u,v)$ should actually be written as $\bar{\Psi}(u,v,\bar{\mathbf{P}}_{k,l})$.

Let the coordinates of $\mathbf{P}_{k,l}$ be $[x_{k,l},y_{k,l},z_{k,l}]^{\mathrm{t}}$. By taking the constant term and the terms involving first partial derivatives in the Taylor expansion of $\bar{\boldsymbol{\Psi}}(\bar{s},\bar{t})$ with respect to (s,t) and $\mathbf{P}_{k,l}$, we obtain the following approximation of $\bar{\Psi}(\bar{s},\bar{t})$

$$\bar{\Psi}(\bar{s},\bar{t}) \approx \Psi(s,t) + \Psi_u(s,t)(\bar{s}-s) + \Psi_v(s,t)(\bar{t}-t) + \boldsymbol{\Psi}_{k,l}(s,t)\cdot\mathbf{V}_{k,l} \tag{16}$$

where

$$\Psi_u(u,v) \equiv \frac{\partial\Psi(u,v)}{\partial u} = \{\mathbf{H}\times[\mathbf{S}_{uu}(u,v)\times\mathbf{S}_v(u,v) + \mathbf{S}_u(u,v)\times\mathbf{S}_{uv}(u,v)]\}\cdot[\mathbf{A}_i - \mathbf{S}(u,v)] - [\mathbf{H}\times\mathbf{N}(u,v)]\cdot\mathbf{S}_u(u,v),$$

$$\Psi_v(u,v) \equiv \frac{\partial\Psi(u,v)}{\partial v} = \{\mathbf{H}\times[\mathbf{S}_{uv}(u,v)\times\mathbf{S}_v(u,v) + \mathbf{S}_u(u,v)\times\mathbf{S}_{vv}(u,v)]\}\cdot[\mathbf{A}_i - \mathbf{S}(u,v)] - [\mathbf{H}\times\mathbf{N}(u,v)]\cdot\mathbf{S}_v(u,v),$$

$$\boldsymbol{\Psi}_{k,l}(u,v) \equiv \left[\frac{\partial\Psi(u,v)}{\partial x_{k,l}}, \frac{\partial\Psi(u,v)}{\partial y_{k,l}}, \frac{\partial\Psi(u,v)}{\partial z_{k,l}}\right]^t,$$

and $\mathbf{V}_{k,l}$ is defined in Eq. (9), with

$$\frac{\partial\Psi(u,v)}{\partial x_{k,l}} = \left\{\mathbf{H}\times\left[[F_{k,l}(u,v)\mathbf{S}_u(u,v) - E_{k,l}(u,v)\mathbf{S}_v(u,v)]\times\begin{bmatrix}1\\0\\0\end{bmatrix}\right]\right\}\cdot[\mathbf{A}_i - \mathbf{S}(u,v)] - R_{k,l}(u,v) \times[\mathbf{H}\times\mathbf{N}(u,v)]\cdot\begin{bmatrix}1\\0\\0\end{bmatrix},$$

$$\frac{\partial\Psi(u,v)}{\partial y_{k,l}} = \left\{\mathbf{H}\times\left[[F_{k,l}(u,v)\mathbf{S}_u(u,v) - E_{k,l}(u,v)\mathbf{S}_v(u,v)]\times\begin{bmatrix}0\\1\\0\end{bmatrix}\right]\right\}\cdot[\mathbf{A}_i - \mathbf{S}(u,v)] - R_{k,l}(u,v) \times[\mathbf{H}\times\mathbf{N}(u,v)]\cdot\begin{bmatrix}0\\1\\0\end{bmatrix},$$

and

$$\frac{\partial\Psi(u,v)}{\partial z_{k,l}} = \left\{\mathbf{H}\times\left[[F_{k,l}(u,v)\mathbf{S}_u(u,v) - E_{k,l}(u,v)\mathbf{S}_v(u,v)]\times\begin{bmatrix}0\\0\\1\end{bmatrix}\right]\right\}\cdot[\mathbf{A}_i - \mathbf{S}(u,v)] - R_{k,l}(u,v) \times[\mathbf{H}\times\mathbf{N}(u,v)]\cdot\begin{bmatrix}0\\0\\1\end{bmatrix},$$

where $R_{k,l}(u,v)$, $E_{k,l}(u,v)$ and $F_{k,l}(u,v)$ are defined in Eqs. (13)–(15).

With (s,t), $\Psi_u(s,t)$, $\Psi_v(s,t)$, and $\boldsymbol{\Psi}_{k,l}(s,t)\cdot\mathbf{V}_{k,l}$ being constants, $\bar{\Psi}(\bar{s},\bar{t})$ and $\Psi(s,t)$ being zero, Eq. (16) is actually a line

$$0 = \Psi_u(s,t)(\bar{s}-s) + \Psi_v(s,t)(\bar{t}-t) + \boldsymbol{\Psi}_{k,l}(s,t)\cdot\mathbf{V}_{k,l}. \tag{17}$$

Any point on this line can be chosen as $(\bar{s},\bar{t})$. Nevertheless, the point that is closest to (s,t) certainly is a better choice because it gives a smaller error for the truncated Taylor expansion (16). The coordinates of such a point can be computed as follows

$$\bar{s} = s - \boldsymbol{\Gamma}\cdot\mathbf{V}_{k,l} \text{ and } \bar{t} = t - \boldsymbol{\Delta}\cdot\mathbf{V}_{k,l},$$

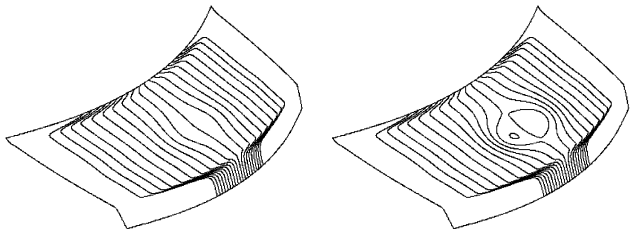
where

$$\boldsymbol{\Gamma} \equiv \frac{\Psi_u(s,t)}{\Omega}\boldsymbol{\Psi}_{k,l}(s,t) \text{ and } \boldsymbol{\Delta} \equiv \frac{\Psi_v(s,t)}{\Omega}\boldsymbol{\Psi}_{k,l}(s,t)$$

with

$$\Omega \equiv [\Psi_u(s,t)]^2 + [\Psi_v(s,t)]^2.$$

$\boldsymbol{\Gamma}$ and $\boldsymbol{\Delta}$ are independent of the control points. Their values can be computed between frames and stored with the values of (s,t) in the linked list of highlight line nodes.



(a) (b)

Fig. 5. Highlight line model of a trunk hood before (a) and after (b) the movement of a control point.

Hence, one can obtain each of the new parameters $\bar{s}$ and $\bar{t}$ in one subtraction and one vector inner product. The values of (s,t) in the linked list of highlight line nodes are then replaced with $(\bar{s},\bar{t})$. $(\bar{s},\ \bar{t})$ usually will not be an edge point of the tessellation mesh. The values of $\bar{\mathbf{S}}(\bar{s},\bar{t})$ and $\bar{\mathbf{N}}(\bar{s},\bar{t})$ can be approximated by bilinear interpolation using values of $\bar{\mathbf{S}}(u,v)$ and $\bar{\mathbf{N}}(u,v)$ at the vertices of the mesh face that contains $(\bar{s},\bar{t})$. If a highlight line model is all that is needed in the rendering process of the deforming NURBS surface, such as the ones shown in Fig. 5, then the value of $\bar{S}(\bar{s},\bar{t})$ can be computed using the following truncated Taylor expansion

$$\bar{\mathbf{S}}(\bar{s},\bar{t}) \approx \mathbf{S}(s,t) + \mathbf{S}_u(s,t)(\bar{s}-s) + \mathbf{S}_v(s,t)(\bar{t}-t) + R_{k,l}(s,t)\mathbf{V}_{k,l}$$

where $R_{k,l}(u,v)$ is defined in Eq. (13). The values of $\mathbf{S}_u(s,t)$, $\mathbf{S}_v(s,t)$ and $R_{k,l}(s,t)$ are independent of the control point $\bar{\mathbf{P}}_{k,l}$. So they can also be computed between frames and stored with (s,t) in the linked list of highlight line nodes. In either case, we can update both the parameters (s,t) and the surface coordinates $\mathbf{S}(s,t)$ for each highlight line node in the highlight line family in linear time. One then generates the next frame by redrawing both the NURBS surface and the highlight line family.

4. Highlight line model re-generation

The above highlight line updating process, using a Taylor expansion technique, works well when the topology of the highlight line model does not change and when deformation of the NURBS surface is small. Small deformation is to ensure small error in the Taylor expansion. If the topology of the highlight line model changes or if the accumulated error is relatively large, it is necessary to re-generate the highlight lines. Since estimating the error term of a Taylor expansion is a well studied problem and the deformation process is local, we need to focus our effort on the study of the highlight line model topology problem only.

Analyzing the topology of a highlight line model on an arbitrary surface is a complex and difficult process. It is even more so when one needs to analyze conditions for the topological change of a highlight line model. Singular cases increase its complexity. Note that a highlight line could contain a patch of a surface, or even the entire surface. For example, if a light source passes through the center point of a sphere, the corresponding highlight line is the entire sphere. If the light source coincides with the axis of a revolution surface, then the corresponding highlight line is the entire surface. In the following, we assume that the light sources are arranged properly so that singular cases do not occur, and discuss the conditions for the highlight line model topology to change when a control point is modified.

4.1. Change of highlight line model topology

A control point of a degree $p \times q$ NURBS surface affects the shape of at most $(p+1)\times(q+1)$ patches of the surface. Therefore, a change of the highlight line model topology can only occur within those patches. Without loss of generality, we shall assume that all of those patches are part of the NURBS surface being considered. Thus, the outside boundary of these patches remains unchanged when that control point is moved. If the endpoints of a highlight line segment are on this boundary, they will remain unchanged when that control point is moved.

The topology of a highlight line model changes when the intersection situation of the highlight lines changes, or when members of the highlight line model change. If the intersection situation of the highlight lines changes but members of the highlight line model remain the same, the above highlight line updating method would still work because the method works on the basis of individual highlight lines, it makes no difference if a highlight line intersects other highlight lines or not. Therefore, in this case, highlight line model re-generation is not necessary.

If members of the highlight line model change (getting new ones or removing old ones), one needs to re-generate the highlight lines. New highlight lines can not have intersection points with the outside boundary of those $(p+1)\times(q+1)$ patches, they must be loops inside those $(p+1)\times(q+1)$ patches. See Fig. 5. Similarly, highlight lines to be removed can only be loops inside those $(p+1)\times(q+1)$ patches as well. Therefore, what one needs here is a mechanism that can detect if a new a highlight line loop is going to emerge or an existing highlight line loop is going to disappear so that the highlight line re-generation process can be evoked promptly.

4.2. Detecting change of highlight line loops

Each highlight line loop of the NURBS surface corresponds to a closed intersection curve of the signed distance surface $D(u,v)$ with a plane $D = ic$. Such an intersection curve occurs around a local maximum or local minimum of the distance surface. Therefore, highlight line loops occur around points of the NURBS surface where the distance surface has local extremes. A parameter space point (u,v) of the NURBS surface is called a *distance extreme point* if the distance surface has a local maximum or local minimum at that point. Each distance extreme point is usually surrounded by a cluster of highlight line loops.

The number of distance extreme points determines the number of clusters of highlight line loops of the NURBS surface. The locations of these points determine the layout of the highlight line loops. Within each cluster, the number of highlight line loops is related to the signed distance of the enclosed distance extreme point. Therefore, distance extreme points are key in detecting change of highlight line loops.

Assume that, before the movement of the control point $\mathbf{P}_{k,l}$, the NURBS surface $\mathbf{S}(u,v)$ has r distance extreme points in the region $(u_k, u_{k+p+1}) \times (v_l, v_{l+q+1})$. Let the parameters of these points be

$$\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_r \qquad (\mathbf{t} \equiv (u(t_i), v(t_i)))$$

and let their signed distances be

$$D_1, D_2, \ldots, D_r \qquad (D \equiv D(\mathbf{t}_i)). \tag{18}$$

After $\mathbf{P}_{k,l}$ is moved to $\bar{\mathbf{P}}_{k,l}$, we have a new NURBS surface $\bar{\mathbf{S}}(u,v)$ and a new signed distance surface $\bar{D}(u,v)$. Suppose the new NURBS surface has $\bar{r}$ distance extreme points in the affected region $(u_k, u_{k+p+1}) \times (v_l, v_{l+q+1})$, whose parameters are

$$\bar{\mathbf{t}}_1, \bar{\mathbf{t}}_2, \ldots, \bar{\mathbf{t}}_{\bar{r}} \qquad (\bar{\mathbf{t}}_i \equiv (u(\bar{t}_i), v(\bar{t}_1)))$$

and whose signed distances are

$$\bar{D}_1, \bar{D}_2, \ldots, \bar{D}_r \qquad (\bar{D}_i \equiv \bar{D}(\bar{\mathbf{t}}_i)). \tag{19}$$

If $r \neq \bar{r}$, with the new NURBS surface having a different number of highlight line loop clusters, obviously the layout of the highlight loops possibly changes.

If $r = \bar{r}$, an indexing scheme can be used to detect if there is a change of the highlight line loops. Without loss of generality, we shall assume that $\mathbf{t}_i$ is moved to $\bar{\mathbf{t}}_i$, for $i = 1, 2, \ldots, r$, after the movement of the control point. Let

$$D_i = J_i \times c + R_i \text{ and } \bar{D}_i = \bar{J}_i \times c + \bar{R}_i \qquad \text{for } i = 1, 2, \ldots, r,$$

where D_i and $\bar{D}_i$ are defined in Eqs. (18) and (19), respectively, c is the distance between two neighboring linear light sources, and $0 \leq R_i, \bar{R}_i < c$ for $i = 1, 2, \ldots, r$. If J_i is different from $\bar{J}_i$ for any i between 1 and r, then there is a change of the highlight line loops: either the number of highlight line loops around the ith distance extreme point becomes different, or those highlight line loops correspond to different light sources.

This method is capable of detecting all possible changes of the highlight line loops. However, this is a very time-consuming approach; one needs to find all the local extremes of the distance surfaces $D(u,v)$ and $\bar{D}(u,v)$. The complexity of this method is $O(N)$ where N is the number of sampling points at the affected region. A control points based, more efficient, approach will be given below.

4.3. A control points based technique

We assume the NURBS surface is located on one side of the light source plane, so, there is no intersection between the NURBS surface and the light source plane. The technique to be presented in this section uses information on the shape of the NURBS surface directly.

Recall that $\mathbf{P}_{k,l}$ is the adjusted control point. $\mathbf{P}_{k,l}$ would affect the shape of $(p+1) \times (q+1)$ patches. These patches are controlled by $\mathbf{P}_{i,j}$, $(i,j) \in \mathbf{I}_1$, where

$$\mathbf{I}_1 = \left\{ (i,j) \middle| \begin{matrix} i = k-p, k-p+1, \ldots, k+p \\ j = l-q, l-q+1, \ldots, l+q \end{matrix} \right\}.$$

For each $(i,j) \in \mathbf{I}_1$, define

$$\begin{cases} \mathbf{P}_1 = \mathbf{P}_{i,j} \\ \mathbf{P}_2 = \frac{1}{4}\mathbf{P}_{i-1,j-1} + \mathbf{P}_{i-1,j+1} + \mathbf{P}_{i+1,j-1} + \mathbf{P}_{i+1,j+1} \end{cases}$$

$\mathbf{P}_2$ is the average of the four control points diagonally adjacent to $\mathbf{P}_{i,j}$. Let $\mathbf{P}_3$ be the intersection point of the line $\mathbf{P}_1\mathbf{P}_2$ with the light source plane. If $\mathbf{P}_3$ and $\mathbf{P}_1$ are on the same side of $\mathbf{P}_2$, and $\|\mathbf{P}_3\mathbf{P}_2\|$ is larger than $\|\mathbf{P}_1\mathbf{P}_2\|$, then the shape of the surface is concave-down around $\mathbf{P}_1$ with respect to the light source plane. $\mathbf{P}_{i,j}$ is called a *convex control point* in this case. Otherwise, the surface is concave-up and $\mathbf{P}_{i,j}$ is called a concave control point accordingly. The case that $\mathbf{P}_3$ does not exist is treated as a concave-up case.

A highlight line would form a loop in a region of a surface only when the surface contains concave-up patch in that region with respect to the light source plane. Therefore, if a surface in a region has no concave-up area both before and after the movement of the control point, there is no highlight line loop enclosed in that region at all. Hence, one only needs to consider regions where the surface has concave-up patch with respect to the light source plane.

Let $\mathbf{I}_2$ be a subset of $\mathbf{I}_1$ whose elements are indices of concave control points before the movement of the control point

$$\mathbf{I}_2 = \{(i,j) | (i,j) \in \mathbf{I}_1 \text{ and } \mathbf{P}_{i,j} \text{ is a concave control point}\}.$$

After the movement of $\mathbf{P}_{k,l}$, the status of a concave control point could change. Let $\bar{\mathbf{I}}_2$ be the counterpart of $\mathbf{I}_2$ after the movement of $\mathbf{P}_{k,l}$

$$\bar{\mathbf{I}}_2 = \{(i,j) | (i,j) \in \mathbf{I}_1 \text{ and } \bar{\mathbf{P}}_{i,j} \text{ is a concave control point}\}.$$

If $\mathbf{I}_2$ and $\bar{\mathbf{I}}_2$ are both empty, since this indicates the affected region of the NURBS surface is concave-down with respect to the light source plane both before and after the movement of the control point, no loop would emerge in this case. Therefore, the status of the highlight line loops in the affected region would not change. If one of $\mathbf{I}_2$ and $\bar{\mathbf{I}}_2$ is empty and the other one is not, then since either new concave-up areas will appear or old concave-up areas will disappear after the movement of the control points, the status of the highlight line loops in the affected region is likely to change.

Now suppose that $\mathbf{I}_2$ and $\bar{\mathbf{I}}_2$ are both non-empty. In this case, we need information on the distance surfaces $D(u,v)$ and $\bar{D}(u,v)$ because extreme points of the NURBS surface

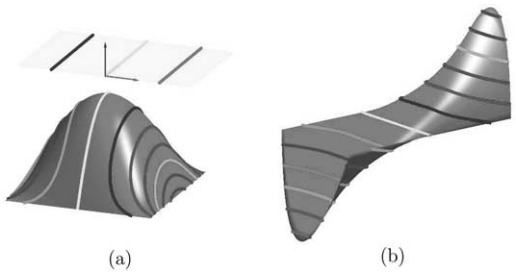


Fig. 6. Correspondence between (a) highlight lines of a NURBS surface and (b) intersection curves of the distance surface with planes D=ic.

are in general not extreme points of the distance surface. They usually locate on the boundary regions that partition the highlight line loop clusters. For instance, the extreme point of the NURBS surface in Fig. 6(a) corresponds to a reflection point of the distance surface in Fig. 6(b). We do not check on the entire NURBS surface, but on some sample points only, namely, Greville abscissae points (u_i^*, v_i^*) for $(i,j) \in \mathbf{I}_1$ [5].

$$\begin{cases} u_i^* = \dfrac{u_{i+1} + u_{i+2} + \cdots + u_{i+p}}{p}, \\ v_j^* = \dfrac{v_{j+1} + v_{j+2} + \cdots + v_{j+q}}{q}, \end{cases} \qquad (i,j) \in \mathbf{I}_1$$

Let (s_1^*, t_1^*) and (s_2^*, t_2^*) satisfy

$$\begin{cases} D(s_1^*, t_1^*) = \max\{D(u_i^*, v_j^*) | (i,j) \in \mathbf{I}_1\}, \\ D(s_2^*, t_2^*) = \min\{D(u_i^*, v_j^*) | (i,j) \in \mathbf{I}_1\}. \end{cases}$$

Similarly, after the movement of the control point, find $(\bar{s}_1^*, \bar{t}_1^*)$ and $(\bar{s}_2^*, \bar{t}_2^*)$ such that

$$\begin{cases} \bar{D}(\bar{s}_1^*, \bar{t}_1^*) = \max\{\bar{D}(u_i^*, v_j^*) | (i,j) \in \mathbf{I}_1\}, \\ \bar{D}(\bar{s}_2^*, \bar{t}_2^*) = \min\{\bar{D}(u_i^*, v_j^*) | (i,j) \in \mathbf{I}_1\} \end{cases}.$$

Let

$$D(s_i^*, t_i^*) = J_i \times c + R_i \text{ and } \bar{D}(\bar{s}_i^*, \bar{t}_i^*) = \bar{J}_i \times c + \bar{R}_i, \text{ for } i = 1,2,$$

where c is the distance between two neighboring linear light sources, and $0 \le R_i, \bar{R}_i < c$. If

$$J_i \neq \bar{J}_i, \qquad \text{for any } i = 1,2,$$

then some of the highlight line loops are likely to change.

4.4. Highlight line re-generation

The indexing technique used in the above testing process can also be used to improve the highlight line generation process. Recall that after the movement of the control point $\mathbf{P}_{k,l}$, an incremental method is used to compute new values of $\bar{\mathbf{S}}(u,v)$, $\bar{\mathbf{S}}_u(u,v)$ and $\bar{\mathbf{S}}_v(u,v)$ for the vertices of the tessellation mesh (Section 3.1).

With the new values of $\bar{\mathbf{S}}(u,v)$, $\bar{\mathbf{S}}_u(u,v)$ and $\bar{\mathbf{S}}_v(u,v)$, one can compute new value of the normal vector $\bar{\mathbf{N}}(u,v)$ and, consequently, new value of the distance surface $\bar{D}(u,v)$ for vertices of the tessellation mesh. Actually, for each vertex of the tessellation mesh, we also calculate a quotient $\bar{I}$ and a remainder $\bar{R}$ so that

$$\bar{D}(u,v) = c \times \bar{I} + \bar{R}, \qquad 0 \le \bar{R} < c$$

where c is the distance between two neighboring light sources. $\bar{I}$ is called the highlight line index of the vertex. Suppose $\mathbf{v}_1$ and $\mathbf{v}_2$ are the vertices of a mesh edge and $\bar{I}_i$ and $\bar{R}_i$, $i = 1,2$, are their highlight line indices and remainders, respectively. If $\bar{I}_1 = \bar{I}_2$ then the interior of this edge has no highlight line nodes or the entire edge is part of a highlight line. If $\bar{I}_1 < \bar{I}_2$, the interior of the edge has

$$\begin{cases} \bar{I}_2 - \bar{I}_1 - 1, & \text{if } R_2 = 0 \\ \bar{I}_2 - \bar{I}_1, & \text{if } R_2 \neq 0 \end{cases}$$

highlight line nodes. The case that $\bar{I}_1 > \bar{I}_2$ can be handled similarly. Therefore, with the help of the highlight line indices, one can immediately tell how many highlight line nodes there are on an edge.

A linear interpolation can then be used to calculate locations of the highlight line nodes on this edge. Each of these nodes will be assigned a highlight line index. These indices form a uniformly distributed increasing (or decreasing) sequence between indices of the edge's endpoints. We connect two highlight line nodes to form a highlight line segment when they have the same index and are in the same tessellation triangle. Hence, highlight line nodes in the interior of two adjacent edges can only be connected in the way shown in Fig. 7(a). If no highlight line nodes have the same indices in the interior of two adjacent edges, then there is no connection between those two edges (Fig. 7b). If a vertex of the tessellation triangle is a highlight line node, the highlight line segment incident to this vertex in this triangle has only two possible cases: connected to its adjacent vertex (Fig. 7c), or connected to a highlight line node on its opposite edge (Fig. 7d).

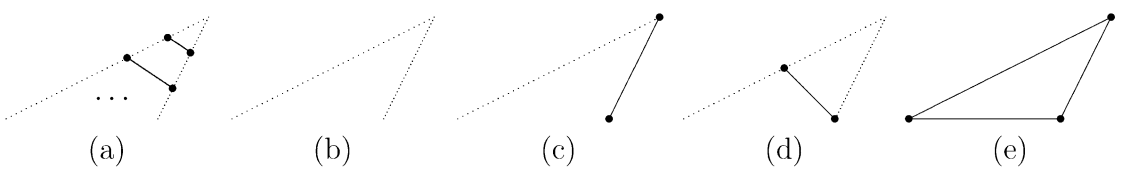


Fig. 7. Possible connection of highlight line nodes in a tessellation triangle.

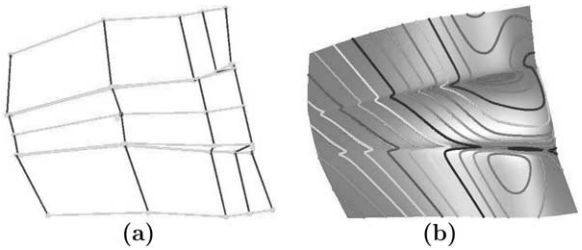


Fig. 8. Example I- Front hood surface of a car: (a) a control point (red dot) is moved to a new location (blue dot), (b) comparison of highlight lines generated by M_t (dotted blue) and by M_c or M_r or (solid curves).

Another advantage of this new indexing technique is that it can deal with the case when the entire tessellation triangle is part of a highlight line. Since the indices of the three vertices are the same, no interior points of the edges would be adopted as highlight line nodes. As shown in Fig. 7(e), its boundary becomes a part of the highlight line instead of the whole triangle. After highlight line nodes are connected into highlight line segments, those highlight line segments with the same indices are connected to form polylines.

5. Implementation

The presented methods have been implemented on a SUN Ultra 10 workstation to test their performance. OpenGL is used as the supporting graphics system. Some of the test results are shown in this section. In the following, for convenience, we shall use M_c, M_r and M_t to refer to Beier and Chen's method [1], the unified distance surface based approach (see Section 2), and the Taylor expansion based scheme (see Section 3), respectively. While M_c and M_r generate the entire highlight line model no matter how small the deformation is, M_t would only update those parts that are affected by the deformation. Since M_t is based on truncating items of high degrees in the Taylor expansion of the distance surface, it works properly when the error item in the Taylor expansion is relatively small, or, when the displacement of the deformation is relatively small. Quantitatively this is achieved when the distance between the original location of the control point, $\mathbf{P}_{k,l}$, and the new location of the control point, $\bar{P}_{k,l}$, satisfies the following condition

$$\|\bar{\mathbf{P}}_{k,l} - \mathbf{P}_{k,l}\|_\infty < 1.$$

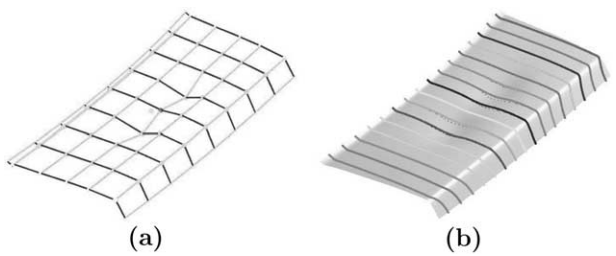


Fig. 9. Example II- Trunk hood surface of a car: (a) a control point (red dot) is moved to a new location (blue dot), (b) comparison of highlight lines generated by (dotted blue) and by or (solid curves).

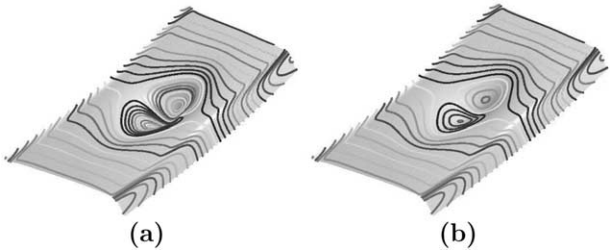


Fig. 10. Topology of highlight line model changes after moving a control point: (a) before deformation; (b) after deformation.

In the test cases shown in Fig. 8(a) front hood surface of a car) and Fig. 9 (a trunk hood surface of a car), a control point of the surface is moved from an old location (red dot) to a new location (blue dot) (Figs. 8(a) and 9(a)), and the corresponding highlight lines are shown in Figs. 8(b) and 9(b). In both cases, the highlight lines generated by M_c (or, M_r) are shown in solid curves and the ones generated by M_t are shown in dotted blue. As can be seen from these examples that the results of M_t are very close to those generated by M_c or M_r with the given displacement of the deformation.

The technique presented in Section 4.2 is used to predict if the topology of a highlight line model would change when a deformation is performed. In the examples shown in Figs. 8 and 9, since $J_i = \bar{J}_i$, for $i = 1, 2$, the topology of the highlight line model does not change after the deformation in either case. An example where the topology of the highlight line model changes after the deformation is shown in Fig. 10. In this example, $J_1 = 6$, $J_2 = -6$, $\bar{J}_1 = 5$ and $\bar{J}_2 = -5$, hence, $J_1 \neq \bar{J}_1$ and $J_2 \neq \bar{J}_2$.

Performance data of these methods on the examples shown in Figs. 8 and 9 are included in Table 1. The data are

Table 1
Performance results of M_c, M_r and M_t

N_s	M_c(s)	M_r(s)	$M_r(s)/M_c(s)$	M_t(s)	$M_t(s)/M_c(s)$
Example I					
861	0.77	0.06	0.083	0.008	0.010
1891	1.66	0.12	0.072	0.014	0.0086
3321	2.93	0.19	0.064	0.023	0.0079
5151	4.51	0.27	0.059	0.032	0.0072
7381	6.49	0.36	0.056	0.045	0.0069
10011	8.81	0.49	0.056	0.057	0.0064
Example II					
961	0.90	0.05	0.0101	0.007	0.0074
2401	2.37	0.12	0.0086	0.014	0.0057
3721	3.48	0.15	0.0079	0.020	0.0056
5329	4.98	0.25	0.0072	0.026	0.0053
8281	7.66	0.37	0.0069	0.039	0.0051
11881	10.93	0.50	0.0064	0.054	0.0050

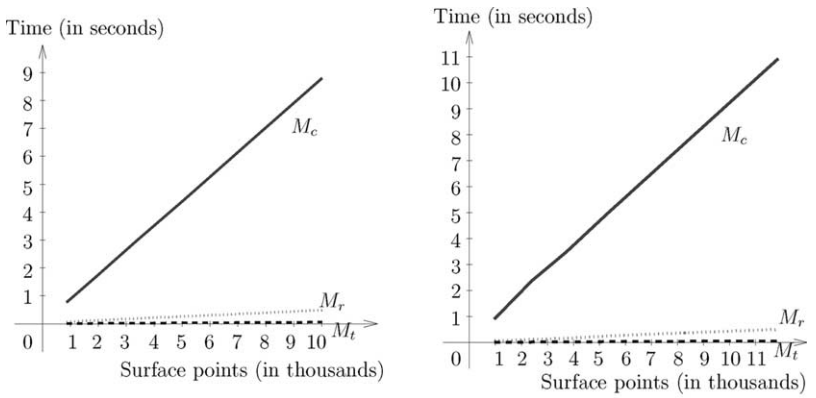


Fig. 11. Performance charts: (solid line), (dotted line), and (dashed line).

also illustrated in two charts in Fig. 11 to make the comparison easy to follow. In Table 1, the numbers below N_s are the numbers of grid points used in the computation process and the numbers below M_c, M_r and M_t are the time (in seconds) used to compute the highlight lines, position and normal vectors of the surface at the specified grid points for these methods. The numbers below M_r/M_c and M_t/M_c are the ratios of M_r to M_c and M_t to M_c, respectively. As shown in the table, in average, M_r requires about $\frac{1}{18}$ of the time used by M_c, and M_t requires about $\frac{1}{148}$ of the time used by M_c only. Since the computation used in predicting if the highlight line model topology would change is required only at a few points, its time cost is very small (0.009 s for Example I and 0.004 s for Example II). This time is independent of the number of grid points.

6. Concluding remarks

A method to dynamically update a NURBS surface and the highlight lines during a local deformation process (with the light sources fixed) is presented. The method takes into consideration of the properties of a moving NURBS surface in the development of the updating process and, consequently, is suitable for an interactive design environment. The method generates the new highlight lines by updating the current highlight lines using a Taylor expansion technique when members of the highlight line model remain the same. Otherwise, it uses a unified distance surface and an indexing technique to facilitate the highlight line re-generation process. The updating process takes only one subtraction and one vector inner product to get the new parameters for each new node. Using the indexing technique, one can also predict when the members of highlight line model would change during the deforming process. An assembly line concept is used here to speed up the computation process, i.e., information that is required in the updating of the surface and the highlight lines is made available at the vertices of the tessellation grid so that computation can be performed efficiently. Possible Improvement of the current algorithm is being studied in several areas. One possible area is to use image based modeling technique in the dynamic updating process of the surface.

Acknowledgements

The authors thank the anonymous reviewers for several helpful comments and suggestions which brought the paper to its current form. Work of the first two authors is supported by NSF (INT-9722728, DMI-9912069).

References

[1] Beier KP, Chen Y. Highlight-line algorithm for realtime surface-quality assessment. Comput-Aid Des 1994;26(4):268–77.
[2] Chen Y. Highlight lines for surface quality control and shape manipulation. PhD Thesis, Department of Naval Architecture and Marine Engineering, the University of Michigan; 1993.
[3] Chen Y, Beier K-P, Papageorgiou D. Direct highlight line modification on NURBS surfaces. Comput Aid Geometr Des 1997; 14(6):583–601.
[4] Cheng F, Luken WL. Computing step sizes for the tessellation of trimmed NURBS surfaces. Technical Report RC18499, IBM Report; 1992.
[5] Farin G. Curves and surfaces for computer aided geometric design: a practical guide. San Diego: Academic Press; 1997.
[6] Kaufmann E, Klass R. Smoothing surfaces using reflection lines for families of splines. Comput-Aid Des 1988;20:312–6.
[7] Klass R. Correction of local surface irregularities using reflection lines. Comput-Aid Des 1980;12(2):73–7.
[8] Li F, Lau R. Incremental polygonization of deforming NURBS surfaces. J Graph Tools 1999;4(4):37–50.
[9] Li F, Lau R. Real-time rendering of deformable parametric free-form surfaces. Proceeding of ACM Symposium on Virtual Reality Software and Technology; 1999. p. 131–8.
[10] Li FWB, Lau RWH, Green M. Interactive rendering of deforming NURBS surfaces. Comput Graph Forum 1997;16(3):C47–C56.
[11] Lien S-L, Shantz M, Pratt V. Adaptive forward differencing for rendering curves and surfaces. Comput Graph 1987;21(4):111–8.
[12] Luken WL. Tessellation of trimmed NURB surfaces. Comput Aid Geometr Des 1996;13(2):163–77.
[13] Luken WL, Cheng F. Rendering trimmed NURBS surfaces. Technical Report RC18669, IBM Report; 1993.
[14] Luken WL, Cheng F. Comparison of surface and derivative evaluation methods for the rendering of NURB surfaces. ACM Trans Graph 1996;15(2):153–78.

[15] Nishita T, Sederberg TW, Kakimoto M. Ray tracing trimmed rational surface patches. Comput Graph 1990;24(4):337–45.
[16] Piegl L, Tiller W. The NURBS book. Berlin: Springer; 1995.
[17] Poeschl T. Detecting surface irregularities using isophotes. Comput Aid Geometr Des 1984;1(2):163–8.
[18] Rockwood AP. A generalized scanning technique for display of parametrically defined surface. IEEE Comput Graph Appl 1987;7(8): 15–26.
[19] Rockwood AP, Heaton K, Davis T. Real-time rendering of trimmed surfaces. Comput Graph 1989;23(3):107–16.
[20] Saunders P. Conceptual design: understanding and communicating form. Proceedings of game developers conference, San Jose, CA; March 20–24 2001.
[21] Shantz M, Chang S-L. Rendering trimmed NURBS with adaptive forward differencing. Comput Graph 1988;22(4):189–98.
[22] Snyder JM, Barr AH. Ray tracing complex models containing surface tessellations. Comput Graph 1987;21(4):119–28.
[23] Zhang C, Cheng F. Removing local irregularities of NURBS surfaces by modifying highlight lines. Comput-Aid Des 1998; 30(12):923–30.

Jun-Hai Yong is a post doctoral fellow in the Department of Computer Science at the University of Kentucky since November 1, 2000. He received his BS and PhD in computer science from the Tsinghua University, China, in 1996 and 2001, respectively. His research interests include computer-aided design, computer graphics and standardization of product data.

Fuhua (Frank) Cheng is Professor of Computer Science and Director of the Graphics and Geometric Modeling Lab at the University of Kentucky where he joined the faculty in 1986. He is also an Adjunct Professor of Applied Mathematics at the Shandong University of Technology, Jinan, China. He received a BS and an MS in mathematics from the National Tsing Hua University in Taiwan in 1973 and 1975, respectively, an MS in mathematics, an MS in computer science, and a PhD in applied mathematics and computer science from the Ohio State University, in 1978, 1980, and 1982, respectively. Cheng has held visiting positions at the University of Tokyo and the University of Aizu, Japan. His research interests include computer aided geometric modeling, computer graphics, parallel computing in geometric modeling and computer graphics, and collaborative CAD.

Yifan Chen is a senior research engineer at Manufacturing Systems Department of Ford Research Lab. His research interests include Computer-Aided Geometric Design and Computer-Aided Manufacturing. His current research is focused on finite element mesh deformation tools and methodologies. Mr. Chen graduated from the University of Michigan in 1993 with a Ph.D. degree in Naval Architecture and Marine Engineering.

Paul Stewart is a Systems Design Engineer from the University of Waterloo. He graduated from the University of Michigan in 1991 with a PhD in Naval Architecture and Marine Engineering. He has spent the last eleven years with the Ford Research Laboratory working on CAD mathematics and design processes. Significant portions of his work have concentrated on effective shape design with NURBS and tessellated surfaces as well as fundamental research in the area of haptics (force-feedback robotics) and it's application to simulation for the automotive design process.

Kenjiro T. Miura is an associate professor of mechanical engineering at the Shizuoka University, Japan, where he joined the faculty in 1997. He received a BEng and an MEng in precision machinery engineering from the University of Tokyo, Japan, in 1982 and 1984, respectively, and a PhD in mechanical engineering from Cornell University in 1991. His research interests include computer aided geometric design, computer graphics, mesh generation, and image processing.