Highlight-line algorithm for realtime surface-quality assessment

Klaus-Peter Beier and Yifan Chen

A new method of surface-quality assessment is presented. The model of highlight lines is introduced, and its properties and applications in surface-quality evaluation are demonstrated. The differential equation of the highlight-line model is derived, and the difficulty involved in seeking its analytical solutions is discussed. Alternatively, the creation of highlight lines is formulated as a surface–plane intersection problem, and solved using surface-contouring techniques. A fast highlight-line algorithm is developed utilizing an efficient traced contouring technique. The algorithm is robust, fully automatic, and, therefore, well suited for realtime quality-assessment tasks.

Keywords: computer-aided geometric design, surface-smoothness indicators, highlight lines, contouring

The design of surfaces can be guided by various objectives, ranging from the satisfaction of geometric conditions or manufacturing constraints to aerodynamic, hydrodynamic or aesthetic principles. The common goal is a smooth surface which captures the intent of the designer and satisfies constraints and performance criteria.

To facilitate the design of smooth surfaces, visual-smoothness indicators such as curvature plots[1] have been used in the past. In recent years, isophotes[2], reflection lines[3,4], and similar techniques have become increasingly popular, especially in automotive-body surface design. A new smoothness indicator called a highlight line has been developed by the authors, and it falls into the same category. These indicators are not only effective in detecting surface irregularities but also share the common attempt to simulate surface-reflection characteristics which can be observed by the human eye; in other words, they demonstrate more intuitive, instead of abstract, features, and are easier to comprehend.

The calculation of highlight lines, isophotes, and reflection lines involves solving systems of partial differential equations[2,5]. In general, it is impossible to find the analytic solution, owing to the inherent complexity of the problem. This paper explores reliable, as well as efficient, numerical procedures that allow the realtime manipulation of highlight lines for smoothness evaluation and quality assessment.

We begin our discussion by introducing the model of highlight lines and highlight bands. We then show properties and applications of highlight lines. We derive the fundamental differential equation of highlight lines, and demonstrate the difficulty involved in seeking analytical solutions. Alternatively, we investigate the computational aspect of finding highlight lines by intersecting an auxiliary surface with a plane, and subsequently formulate a surface–plane intersection problem. We then propose our solution method using surface-contouring techniques. We develop the technique of constructing the auxiliary surface, and present a traced contour-line algorithm that is based on an edge-oriented data structure. Finally, we provide a performance analysis based on our implementation results.

DEFINITION OF HIGHLIGHT LINE/BAND

A highlight line is created by an assumed linear light source idealized by a straight line with an infinite extension. The light source is positioned somewhere above the surface under consideration. As shown in *Figure 1*, the *imprint* of the light source on the surface is the collection of all the surface points for which the extended surface normal passes through the light source.

Department of Naval Architecture and Marine Engineering, University of Michigan, 2600 Draper, NA&ME Building, Ann Arbor, MI 48109-2145, USA

Paper received: 30 May 1993. Revised: 15 October 1993

0010-4485/94/04/0268–10 © 1994 Butterworth-Heinemann Ltd

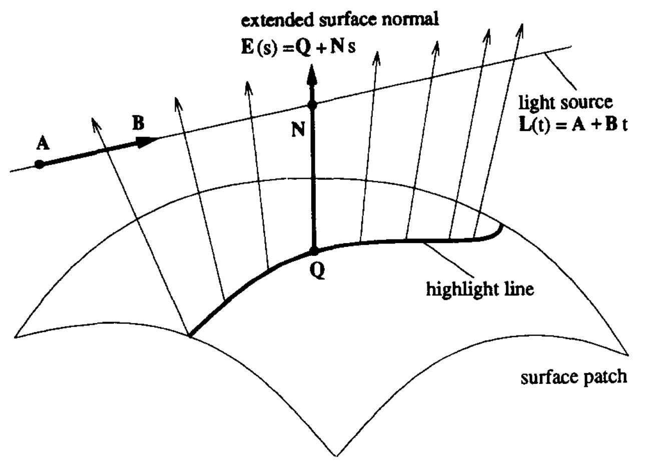


Figure 1 Definition of highlight line

This imprint is the *highlight line*, and it is defined as follows:

Definition 1: A highlight line is a set of points on a surface for which the perpendicular distance between the surface normal and a linear light source is zero.

The condition for a surface point belonging to a highlight line can be formulated as follows. Assume the linear light source $\mathbf{L}(t)$ (see *Figure 1*) is given by

$$\mathbf{L}(t) = \mathbf{A} + \mathbf{B}t \tag{1}$$

where **A** is a point on $\mathbf{L}(t)$ and **B** is a vector defining the direction of $\mathbf{L}(t)$. For a given surface point **Q**, let **N** be the directional vector of the corresponding surface normal. The extended surface normal $\mathbf{E}(s)$ at **Q** is, therefore, a line passing through **Q** in the direction of **N**, and it can be defined as follows:

$$\mathbf{E}(s) = \mathbf{Q} + \mathbf{N}s \tag{2}$$

Point **Q** belongs to the highlight line if both lines $\mathbf{L}(t)$ and $\mathbf{E}(s)$ intersect, that is, if the perpendicular distance d between both lines (see, for example, Reference 6) given by

$$d = \frac{|(\mathbf{B} \times \mathbf{N}) \cdot (\mathbf{A} - \mathbf{Q})|}{\|(\mathbf{B} \times \mathbf{N})\|} \tag{3}$$

is zero.

The highlight-line model can be generalized by replacing the linear light source with a cylinder of a given radius, thereby simulating a more realistic light source. The corresponding imprint on the surface is a *highlight band* with a varying bandwidth that is determined by the cylinder's radius and by the change in surface-normal directions. A sample highlight band is shown in *Colour Plate 1*. Highlight bands are defined as follows:

Definition 2: A highlight band is a set of points on a surface for which the perpendicular distance between the surface normal and the centreline of a cylindrical light source is less than or equal to the radius of the light source.

In other words, a point **Q** on the surface belongs to a highlight band if the extended surface normal at **Q** passes through the cylinder. Definition 1 of the light source is still valid, but it now applies to the centreline of the cylinder. A point **Q** on the surface belongs to a highlight band if

$$d \leqslant r \tag{4}$$

where d is defined in Equation 3, and r is the radius of the light-source cylinder.

The equals sign in Equation 4 holds for all the surface points for which the extended surface normal is tangential to the light-source cylinder. The curve formed by this collection of points is called the *highlight-band boundary*.

Definition 3: A highlight-band boundary is a set of points on a surface for which the perpendicular distance between the surface normal and the centreline of a cylindrical light source is equal to the radius of the light-source cylinder.

Note that, when r becomes zero, the highlight-band boundary becomes the highlight line. This property is used in the development of the highlight-line algorithm.

The creation of colour-coded or grey-shaded highlight bands, as shown in *Colour Plate 1*, is discussed in Reference 7. This initial work by the authors was based on an exhaustive search method, and it involved vector and parallel-processing techniques to achieve realtime response. This paper, in contrast, presents an efficient method for the finding of the highlight-band boundary curves and the highlight lines, and it achieves realtime response with a conventional hardware architecture.

FORMS, PROPERTIES, AND APPLICATIONS

A highlight line can assume a variety of forms including closed or intersecting curves, depending on the shape of the surface and the location and orientation of the light source. A highlight line is usually a single curve if the light source is placed off the convex side of a surface, as shown in *Figure 2*. Disconnected curves, loops, and intersections are typically found if the light source is moved to the concave side of the surface as shown in *Figures 3* and *4*.

In an interactive-graphics environment, the user can inspect the entire surface by translating or rotating the light source, and thereby sweeping the highlight line over the surface. As an extension, a single light source can be replaced by an array of parallel light sources. This creates a family of highlight lines covering the surface. An example is given in *Colour Plate 2*. As an alternative extension, a cylindrical light source with a uniformly increasing sequence of radii can be used to create a family

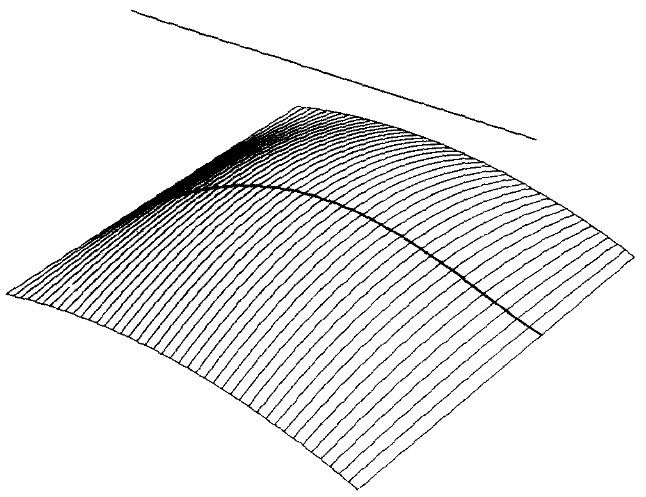


Figure 2 Highlight line resulting from light source located off convex side of surface

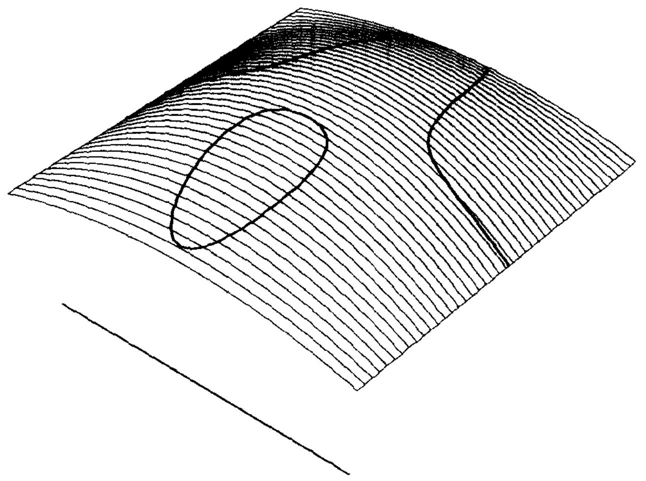


Figure 3 Disconnected highlight lines, including loop resulting from light source located off concave side of surface in *Figure 2*

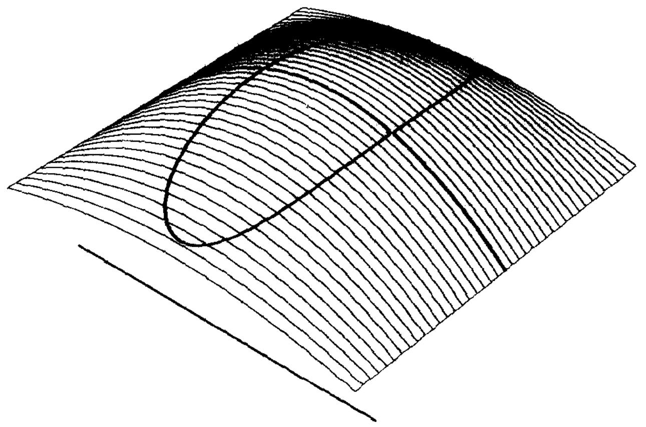


Figure 4 Intersecting and circular-shaped highlight lines resulting from slight manipulation of light source in *Figure 3*

of highlight-band boundary curves as shown in *Colour Plate 3*.

Two important properties of the highlight-line model facilitate surface-quality assessment and apply to the highlight-band model in a similar way.

Property 1: A highlight line is viewer-independent.

The highlight line is a simplified reflection model in the sense that it is viewer-independent. In contrast to reflection lines[3], the path of a highlight line does not change if the viewer changes his/her viewpoint. The surface and light source may be rotated or translated together to facilitate the inspection of the highlight line. Since the highlight line remains at the same location, no calculations are required to update the highlight line during viewing operations.

Property 2: A discontinuity on a surface is magnified by an order of one in the highlight line.

The highlight-line model involves surface normals, and it is sensitive to the change of normal directions. For a parametric surface $\mathbf{Q}(u, v)$, a normal can be defined as the crossproduct of the two first partial derivatives (tangent vectors) at a given point. If these first derivatives of the surface are discontinuous (tangent discontinuity), the highlight line exposes a positional discontinuity, as shown in *Figure 5*. A 2nd-order discontinuity of a surface results in a tangent discontinuity in the highlight line, as shown in *Figure 6*, and so forth. Highlight bands and highlight-band boundaries have similar properties.

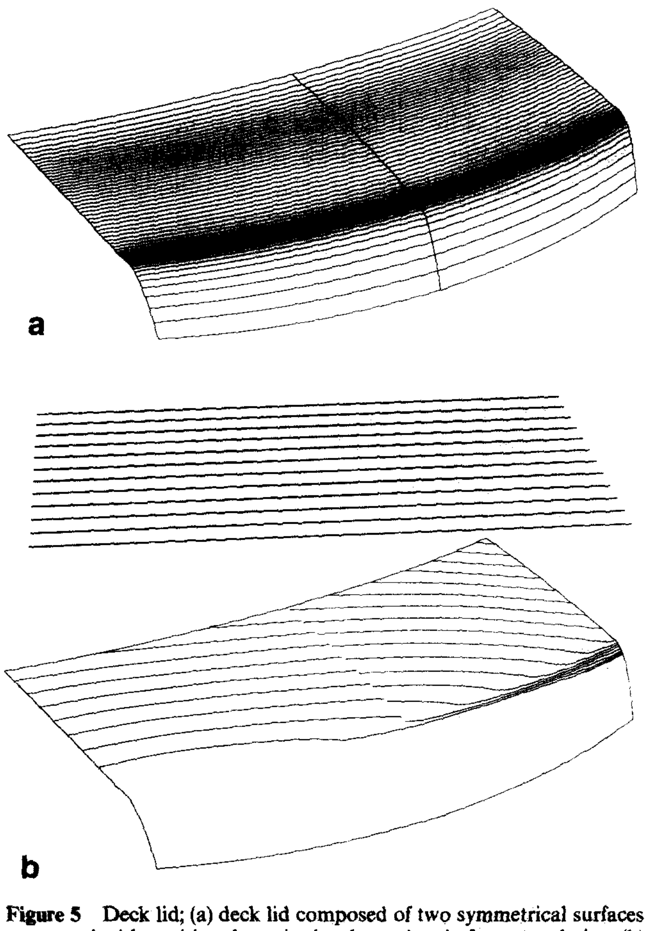


Figure 5 Deck lid; (a) deck lid composed of two symmetrical surfaces connected with positional continuity shown in wireframe rendering, (b) family of highlight lines resulting from array of parallel light sources

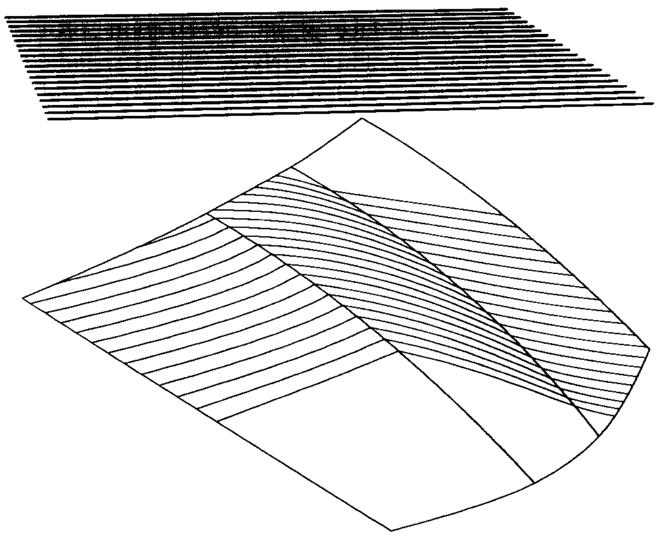


Figure 6 Surface with two C^1 internal patch boundaries
[The highlight lines are connected along these boundaries with only C^0 continuity.]

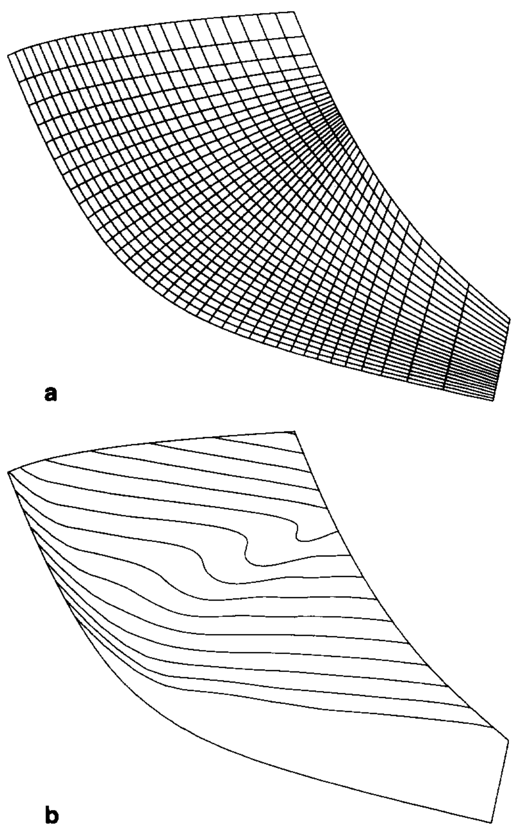


Figure 7 Surface; (a) B-spline surface in wireframe, (b) pattern of highlight lines created by array of parallel light sources, revealing small dent

Therefore, a family of highlight lines derived from an array of parallel light sources (see *Colour Plate 2*) can be replaced by a family of highlight-line boundaries created from a single light source with multiple cylinder radii (see *Colour Plate 3*). The pattern of band boundaries delivers similar information, and can be used instead.

The sensitivity of the highlight lines allows for the detection of the small surface irregularities often found in mathematically smooth surfaces. The wireframe representation of a surface in *Figure 7a* seems to indicate a smooth surface. The highlight lines in *Figure 7b,* however, clearly reveal a small dent in the surface through 'glitches' in the pattern of the highlight-line family.

HIGHLIGHT-LINE/BAND BOUNDARY ALGORITHM

Fundamental differential equations

In theory, a highlight line or band boundary can be defined by a system of nonlinear ordinary differential equations. Consider a highlight line or band boundary in the uv space of a parametric surface $\mathbf{Q}(u, v)$ as follows:

$$\mathbf{H}_{uv}(t)=\begin{cases} u(t) \\ v(t) \end{cases} \qquad \alpha \leqslant t \leqslant \beta \tag{5}$$

where t is the parameter and α and β the lower and upper bounds of t. For a given surface and light source, the distance d of all points on $\mathbf{H}_{uv}$ can be written as a function of $u(t)$, $v(t)$, i.e. $d(u(t), v(t))$, and it must be equal to the light radius r. Thus, we have the following distance equation:

$$d(u(t), v(t))=r \tag{6}$$

where $r=0$ for the highlight line, and $r>0$ for the band boundary. Note that the absolute sign involved in the distance formula of Equation 3 can reduce the continuity of the distance equation to C^0, causing discontinuities in its 1st- and higher-order derivatives. This problem can be solved by simply eliminating the absolute sign, resulting in a *signed distance function** denoted by d_s as follows:

$$d_s=\frac{(\mathbf{B}\times\mathbf{N})\cdot(\mathbf{A}-\mathbf{Q})}{\|(\mathbf{B}\times\mathbf{N})\|} \tag{7}$$

Consequently, the distance Equation 6 can be modified into

$$d_s(u(t), v(t))=\pm r \tag{8}$$

By differentiating both sides of Equation 8 with respect to t, we obtain

$$\nabla\, d_s(u, v)^{\mathrm{T}}\begin{pmatrix} u'(t) \\ v'(t) \end{pmatrix}=0 \tag{9}$$

which provides one differential equation for $u(t)$, $v(t)$. A second one can be made available by adopting the arc-length parameterization for $u(t)$, $v(t)$, thereby yielding

*In the literature of surface–surface intersection, the signed distance function is also called the oriented distance function.

the following equation:

$$(u'(t), v'(t))\begin{pmatrix} u'(t) \\ v'(t) \end{pmatrix} = 1 \qquad (10)$$

Equations 9 and 10 form a fundamental system of 1st-order nonlinear ordinary differential equations for the highlight line.

In general, it is difficult to find $u(t)$, $v(t)$ by solving the differential-equation system analytically. For instance, a bicubic Bézier surface has the following distance equation:

$$d_s(u, v) = \frac{\sum_{i=0}^{8} \sum_{j=0}^{8} s_{i,j} u^i v^j}{\left(\sum_{i=0}^{10} \sum_{j=0}^{10} q_{i,j} u^i v^j \right)^{\frac{1}{2}}} = \pm r \qquad (11)$$

where $s_{i,j}$, $q_{i,j}$ are the polynomial coefficients determined through the surface and the given light source. Differentiating both sides of Equation 11 yields the following:

$$\frac{\sum_{i=0}^{17} \sum_{j=0}^{18} a_{i,j} u^i v^j}{\left(\sum_{i=0}^{10} \sum_{j=0}^{10} q_{i,j} u^i v^j \right)^{\frac{3}{2}}} u' + \frac{\sum_{i=0}^{18} \sum_{j=0}^{17} b_{i,j} u^i v^j}{\left(\sum_{i=0}^{10} \sum_{j=0}^{10} q_{i,j} u^i v^j \right)^{\frac{3}{2}}} v' = 0 \qquad (12)$$

where $a_{i,j}$, $b_{i,j}$ are the coefficients calculated from $s_{i,j}$, $q_{i,j}$. This equation, together with Equation 10, forms a nonlinear system, which is impossible to solve analytically.

Formulation as surface–plane intersection problem

The finding of highlight lines as well as highlight-band boundaries on a parametric surface $\mathbf{Q}(u, v)$ can be formulated as a surface–plane intersection problem performed on an auxiliary surface. The perpendicular distance d between a surface-normal direction and the linear light source, as given in Equation 3, can be calculated for any surface point $\mathbf{Q}(u, v)$ and, therefore, treated as a continuous function $d(u, v)$. This distance function $d(u, v)$ is a single-valued surface over the uv domain as shown in *Figure 8a*. When intersecting this auxiliary surface with a plane $d = c$ parallel to the uv plane, we obtain one or more intersection curves $\mathbf{H}_{uv}$ which can be projected in the uv plane as shown in *Figure 8b*. Using a surface evaluator for $\mathbf{Q}(u, v)$, we can map the intersection curves into the object space of the surface $\mathbf{Q}(u, v)$, as shown in *Figure 8c*, and we obtain a highlight line for $c = 0$ (lines of constant distance zero) or the highlight-band boundaries for $c = r$ (lines of constant distance r).

Use of contouring algorithms

The surface–plane intersection problem can be solved numerically using a contouring algorithm. These algorithms are used to find curves of constant height (isolines or contour lines) on a single-valued surface. The surface is defined by discrete values given at the vertices of a mesh; in our application, this is any mesh defined in the 2D uv space.

Contouring algorithms are robust and fully automatic. The mesh used for the rendering of the surface $\mathbf{Q}(u, v)$ can be fully utilized for the numerical definition of the distance surface $d(u, v)$. The accuracy of the contouring process can be controlled through the mesh density.

Figure 9 shows the distance surface $d(u, v)$ derived from the example given in *Colour Plate 1*. As a mesh, a rectangular uv grid is used, and the numerical values of d are defined at the vertices of the mesh. This surface can be intersected with a plane $c = r$ to find the highlight-band

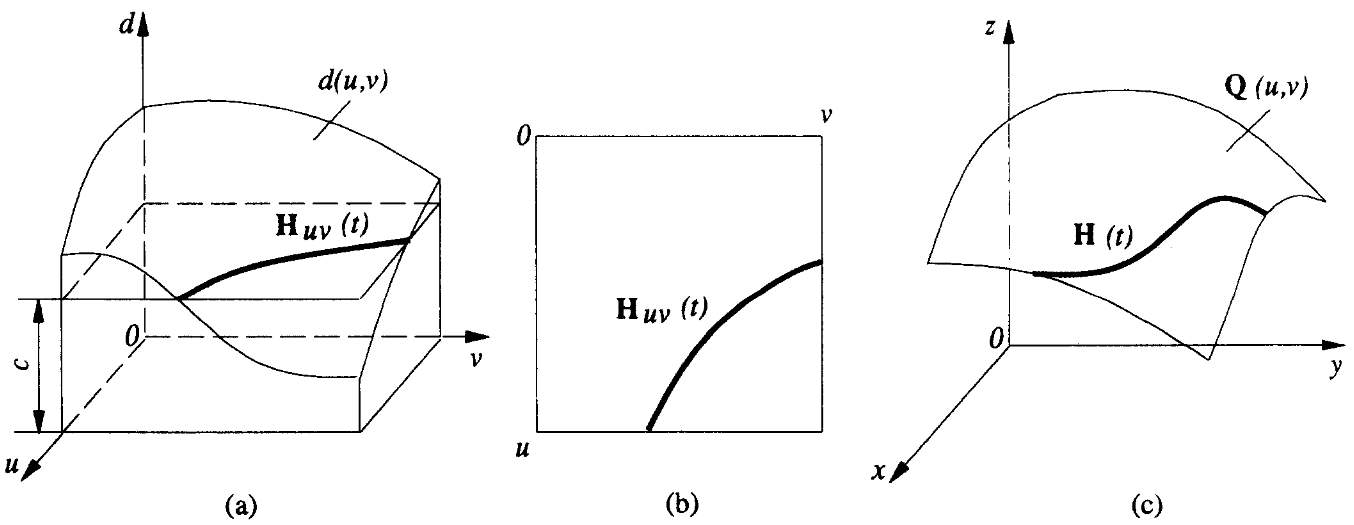
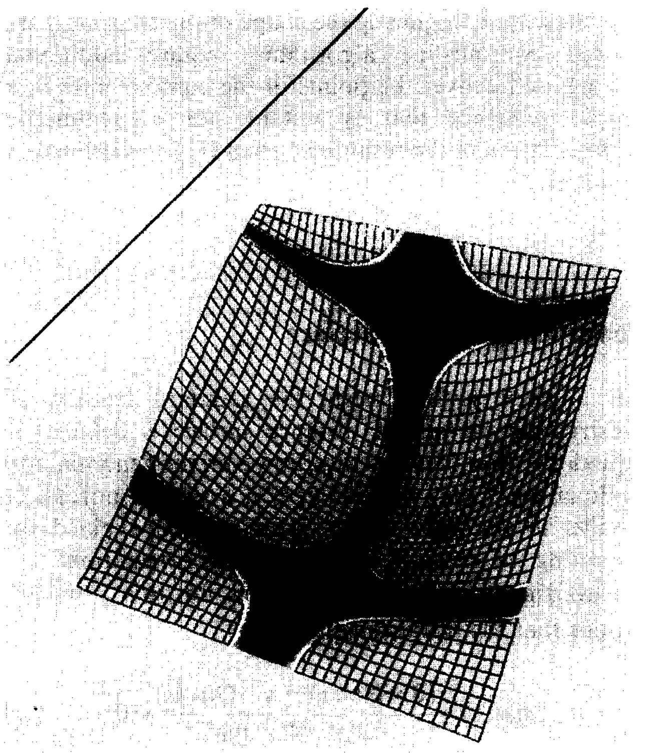
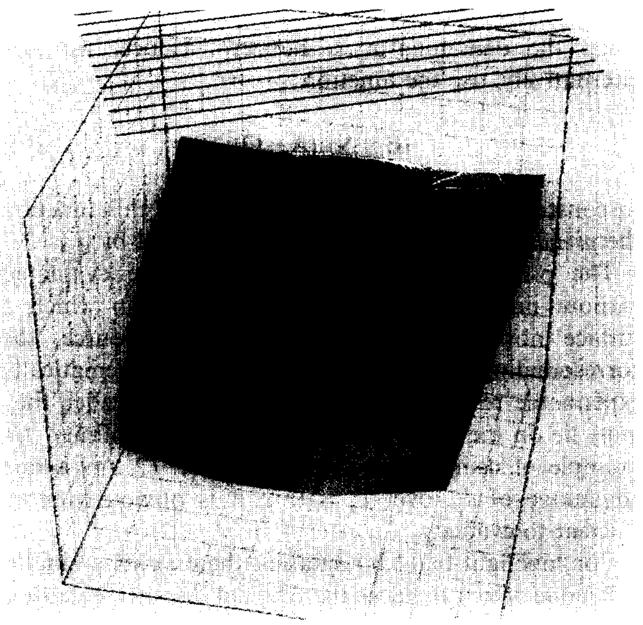
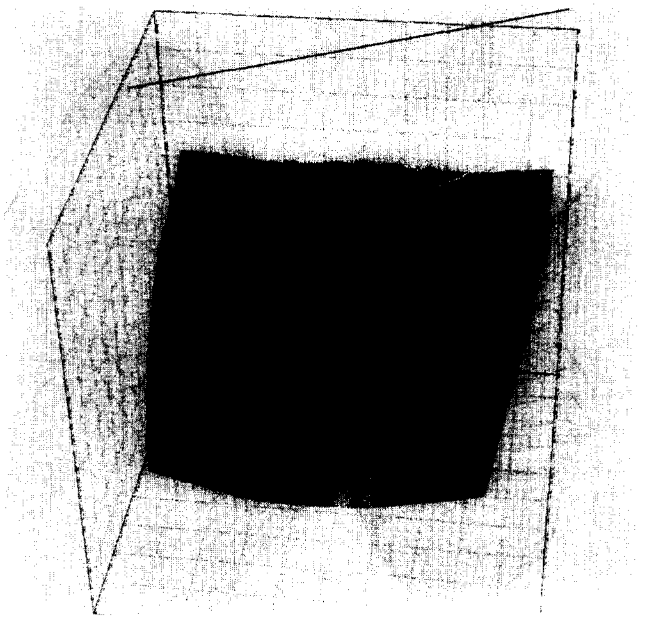


Figure 8 Surface–plane intersection approach; (a) $d(u, v)$ surface intersecting plane $d = c$ in auxiliary space, (b) resulting highlight-band boundary projected in uv space, (c) highlight-band boundary mapped into object space


Colour Plate 1 Example of highlight band.
[The colour of the band is determined by the distance d; the darker the colour, the smaller the d.]


Colour Plate 2 Family of highlight lines created from array of parallel light sources

boundaries. However, finding the highlight line by intersecting with the plane $c=0$ leads to numerical problems, since d is always greater than or equal to zero, and the distance surface does not cross the uv plane $c=0$. Replacing the distance function $d(u, v)$ by the signed distance function $d_s(u, v)$ as defined in Equation 7 solves


Colour Plate 3 Family of multiple highlight-band boundaries created from cylindrical light source with multiple radii

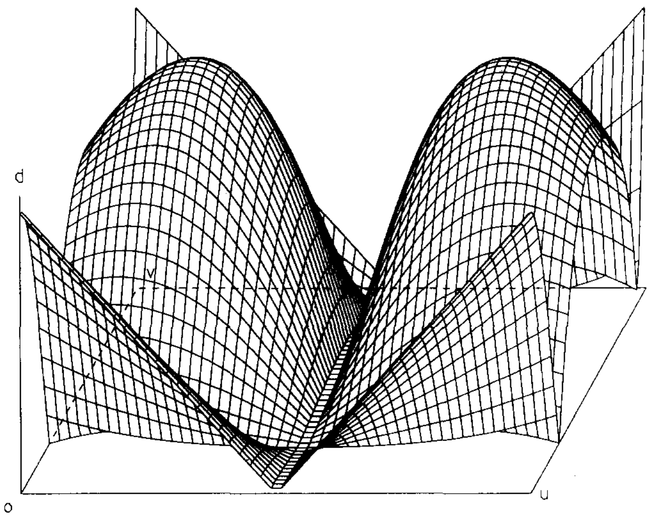


Figure 9 Distance surface corresponding to example in *Colour Plate 1*

this problem. As shown in *Figure 10*, $d_s(u, v)$ extends to both sides of the uv plane, and allows for the intersection with $c=0$ as well as with $c=\pm r$. *Figure 11* shows the resulting highlight-band boundaries and the centred highlight line in uv space. Mapped into the object space, the corresponding patterns can be recognized in *Colour Plate 1*.

The auxiliary surface for the signed distance function $d_s(u, v)$ is valid only for a specific location and orientation of the light source. To find a family of highlight lines for an array of parallel light sources (see, for example, *Colour Plate 2*) requires the calculation of a new surface d_s for each of the light sources. A family of highlight-line boundary curves derived from a single light source with a varying sequence of cylinder radii, however, can be

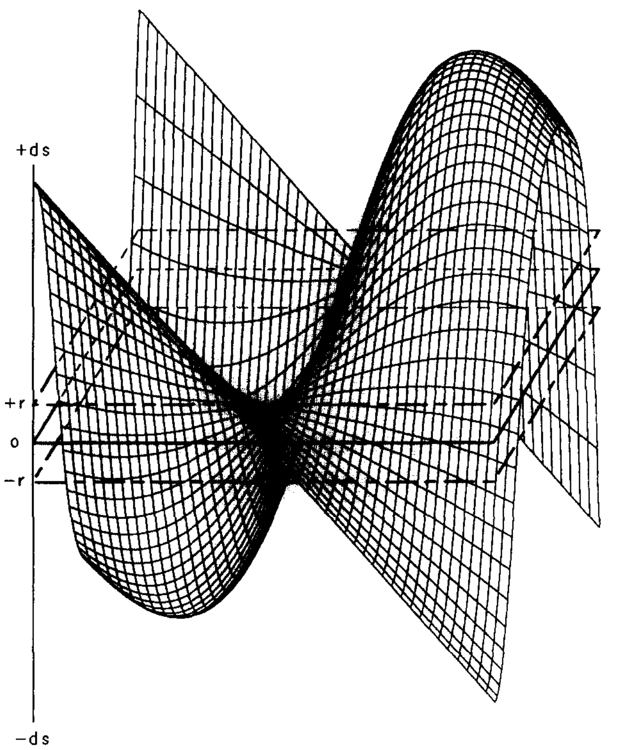


Figure 10 Signed distance surface derived from distance surface in *Figure 9*

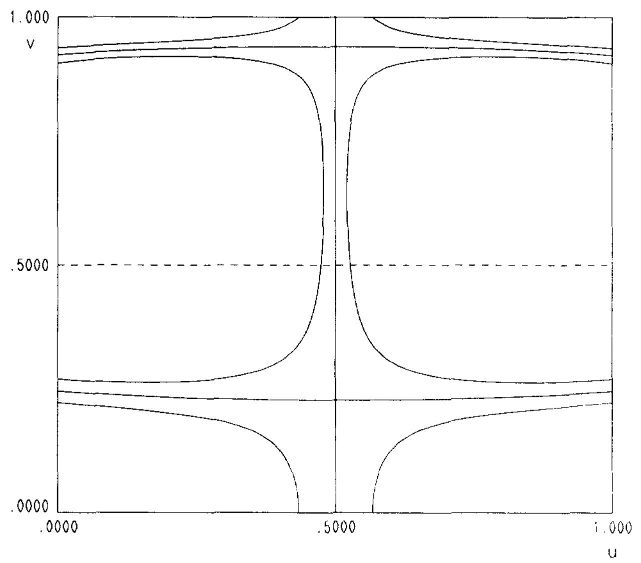


Figure 11 Result of intesecting signed distance surface shown in *Figure 10* with planes $c=0$ and $c=\pm r$ in uv space
[Isolines for $d=+r$, $d=-r$ and $d=0.0$.]

found from a single d_s surface by intersecting d_s with a sequence of planes $c=\pm r$. This, obviously, decreases the computational load significantly and still delivers similar information, as demonstrated in *Colour Plate 3*.

Numerical contouring algorithms can be classified into two types. Piecewise contour-line algorithms intersect the plane c with one face of the mesh at a time, and deliver an unordered set of line segments. These disconnected line segments, when displayed, blend visually together and allow for recognizing the shape of the contour lines. Traced contour-line algorithms require additional sorting, and deliver the points of the contour lines in a proper sequence that is suitable for a parametric representation of the resulting curves $\mathbf{H}_{uv}(t)$ as defined in Equation 5.

Constructing distance surface

For a given surface mesh, the standard method of constructing the signed distance surface is defined in Equation 7. In some rare cases, however, the denominator in Equation 7 may become zero if a surface normal is parallel to the light source. This can be avoided by restricting the range of the light-source orientation.

Nevertheless, there is a solution for this problem. Notice that the distance equation

$$d_s(u,v)=\frac{(\mathbf{B}\times\mathbf{N}(u,v))\cdot(\mathbf{A}-\mathbf{Q}(u,v))}{\|(\mathbf{B}\times\mathbf{N}(u,v))\|}=0 \tag{13}$$

which analytically defines the highlight line, is equivalent to

$$(\mathbf{B}\times\mathbf{N}(u,v))\cdot(\mathbf{A}-\mathbf{Q}(u,v))=0 \tag{14}$$

when the denominator is nonzero. Hence, we can, alternatively, use the function

$$(\mathbf{B}\times\mathbf{N})\cdot(\mathbf{A}-\mathbf{Q}) \tag{15}$$

to construct the distance surface. We call this function the *pseudo signed distance function*, denoted by d_{ps}.

The pseudo signed distance d_{ps} can be evaluated without numerical problems. In the case in which a surface normal is parallel to the light source, the corresponding d_{ps} is zero. This is a mathematically explainable result: since the two lines are parallel, they must lie on the same plane and intersect at infinity. An example of using d_{ps} is given in *Figure 12*. An added advantage of d_{ps} over d_s is that it is obviously more efficient to evaluate.

For highlight-band boundaries, there exists a similar technique which replaces the original distance equation

$$\frac{(\mathbf{B}\times\mathbf{N}(u,v))\cdot(\mathbf{A}-\mathbf{Q}(u,v))}{\|(\mathbf{B}\times\mathbf{N}(u,v))\|}=\pm r \tag{16}$$

with the alternative equation

$$d_a=(\mathbf{B}\times\mathbf{N})\cdot(\mathbf{A}-\mathbf{Q})-(\pm r)\|(\mathbf{B}\times\mathbf{N})\|=0 \tag{17}$$

where d_a is called the *alternative function* to d_s for the generation of highlight-band boundaries. Similarly, d_a

can be evaluated without numerical problems. Note that the adoption of d_a leads to the use of the plane $c=0$ instead of the plane $c=\pm r$ to intersect the distance surface for the creation of highlight-band boundaries.

Contouring offset surface

Numerical contouring algorithms are frequently used in computer-graphics applications, and they are well covered in the literature (see, for example, References 8–11). For the creation of highlight lines, the contouring algorithm is the 'workhorse', and it is instrumental in achieving realtime response. Therefore, a speed-optimized contouring algorithm called NISO has been developed at the University of Michigan, USA. The algorithm is unique in its design, and is briefly described in the following.

NISO generates contour lines of a single-valued continuous surface in linear time using the traced contour-line scheme. The surface is represented by its offset values at the vertices of a mesh which consists of quadrilateral faces, triangular faces, or both types of face. We exemplify the principles of NISO using the quadrilateral faces of a rectangular grid.

To trace the path of contour lines efficiently, NISO creates the set of interrelated tables shown in *Figure 13*. The vertex table lists all the vertices of the mesh with coordinates u, v and surface offset values d_s. The edge table contains all the edges of the mesh. Each edge is defined by the pointers to the beginning and ending vertices V_B, V_E. In addition, pointers to the face table indicate the two faces F_l, F_r that the edge belongs to. The ISDT column in the edge table is initially zero, and is used during processing. The face table defines all the faces of the mesh through pointers to their bounding edges (there are four edges for a quadrilateral mesh).

With the vertex, edge and face tables in place, the algorithm finds the contour lines for a given isovalue c in two steps:

- *Step 1:* All the edges in the edge table are intersected with the plane c using information from the vertex table. If an edge intersects the plane, the intersection point is registered in an intersection table with a reference pointer E to the edge. A backwards pointer I is stored in the edge table in the ISDT column.
- *Step 2:* A tracing algorithm is used to find the connectivity between the intersection points, and it delivers a set of point sequences, each representing the path of a continuous contour-line segment.

The principle of the tracing algorithm is briefly illustrated in *Figure 14*. The intersection point I_{in} on edge E_1 leads the contour line from face F_1 into face F_2. This information is available from the edge table through the pointers F_l and F_r. To find the next intersection point I_{out} from the intersection table, the three remaining edges from face F_2 are inspected for intersections. The face table provides the pointers to these three edges; the edge table

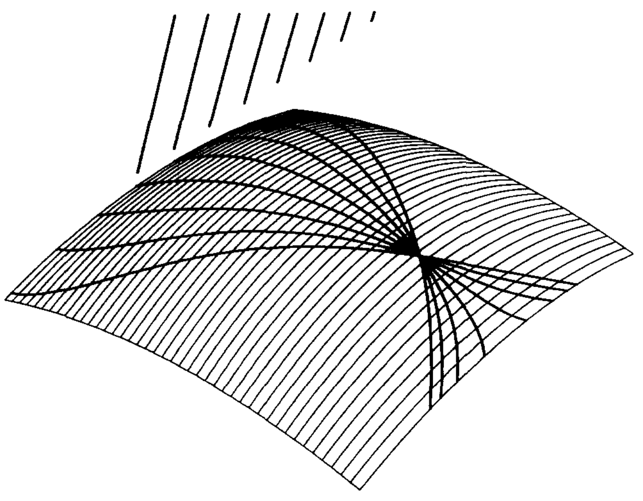


Figure 12 Generation of multiple highlight lines using the pseudo signed distance function. The intersection of highlight lines indicates the location where the surface normal is parallel to the light source direction

Vertex Table

V	VX()	VY()	VF()
1	u	v	d_s
2	u	v	d_s
V_{max}	u	v	d_s

Edge Table

E	ISD(4,)				ISDT()
1	V_B	V_E	F_l	F_r	I
2	V_B	V_E	F_l	F_r	I
E_{max}	V_B	V_E	F_l	F_r	I

Face Table

F	IFDS(4,)			
1	E_1	E_2	E_3	E_4
2	E_1	E_2	E_3	E_4
F_{max}	E_1	E_2	E_3	E_4

Intersection Table

I	XB()	YB()	IB()
1	u	v	E
2	u	v	E
I_{max}	u	v	E

u,v: Coordinates of vertex.
d_s : Surface value at vertex.

V: Pointer to vertex table.
F: Pointer to face table.
I: Pointer to intersection table.
(I=0: no intersection)

E: Pointer to edge table.

u,v: Coordinates of intersection.
E : Pointer to edge table.

Figure 13 Implementation of edge-oriented data structure in NISO

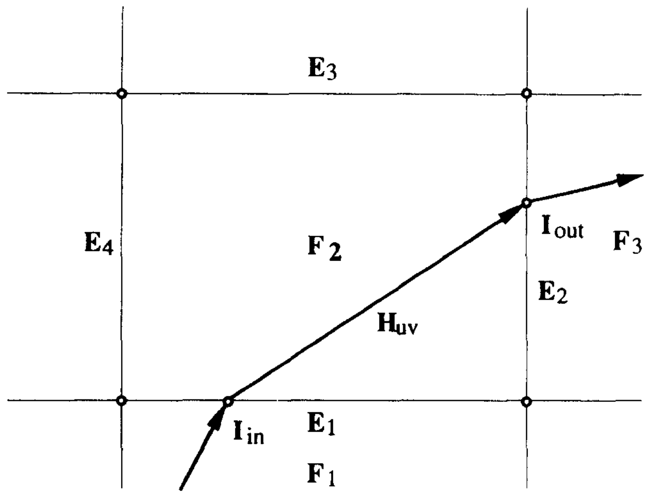


Figure 14 Principle of tracing

indicates in pointer I whether the edge has an intersection or not. Once I_{out} has been located in the intersection table, the process is repeated for the face F_3 in *Figure 14*. I_{out} now serves as I_{in} for the next face.

Steps 1 (intersecting) and Step 2 (tracing) have linear time complexity, and are very efficient. Creating the edge table, however, may be time-consuming. The edge and face tables contain only topological information about the mesh. Both tables can be created before the surface offsets d_s are evaluated and registered in the vertex table. When moving the light source to a new position (new d_s values in the vertex table), the same edge and face tables can be used as long as the same mesh is maintained.

Assessment of algorithm

The algorithm is an efficient computational algorithm for realtime surface-quality assessment. The efficiency is based on the linear-time-complexity nature of this algorithm. The total time consumption of the algorithm can be divided into two major fractions: the time needed to perform numerical computations, and the time needed to display results on the screen. The computational part of the algorithm includes three major steps: constructing an offset surface, contouring the offset surface, and mapping the intersection curve(s) back into the object space; all of them are of linear complexity. Although the complexity of the graphics-display part varies with the hardware used, it is usually linear in nature as well. Therefore, the overall time complexity of the algorithm is linear.

The algorithm was implemented on a Silicon Graphics Personal Iris workstation. To verify the time complexity and speed of the algorithm, the wall clock time* was measured in relation to the manipulation of a light source.

*Wall clock time is the only meaningful criterion for the end user of a system.

The surface used in our measuring environment is a bicubic B-spline surface, as shown in *Figure 2*, and its grid resolution ranges from 32×32 to 100×100 (corresponding to a total number of 1 024 to 10 000 points). *Figure 15* shows the response times for the manipulation of a highlight line through the rotation of the light source. In addition, *Figure 15* also indicates the time savings obtained by using the pseudo signed distance function.

From *Figure 15*, we observe the following:

- The response time is close to linear, which verifies the linear time complexity of the algorithm.
- For the given range of grid resolution, a light-source manipulation can be completed within, approximately, 0.050–0.193 s. This allows the screen to be updated at a corresponding rate of 20.0–5.2 frames/s. The result provides the user with an adequate realtime response for interactive surface-quality assessment.
- The use of the pseudo signed distance function saves an average of 15% of the total response time.

CONCLUSIONS

The highlight-line model is a simplified reflection model. The elimination of the viewpoint decouples the viewing operation from the manipulation of highlight lines, and, therefore, improves the performance of the model. The highlight-line model inherits the fundamental properties of reflection lines and isophotes, such as the continuity-reduction property, and it has proved to be intuitive and effective in detecting local surface imperfections.

The formulation of finding highlight lines as a surface–plane intersection problem allows us to solve the problem efficiently using a traced-isoline algorithm, which is fully automatic and robust. In addition, it allows the use of a low surface-grid resolution to produce a satisfactory rendering quality. As a result, our algorithm

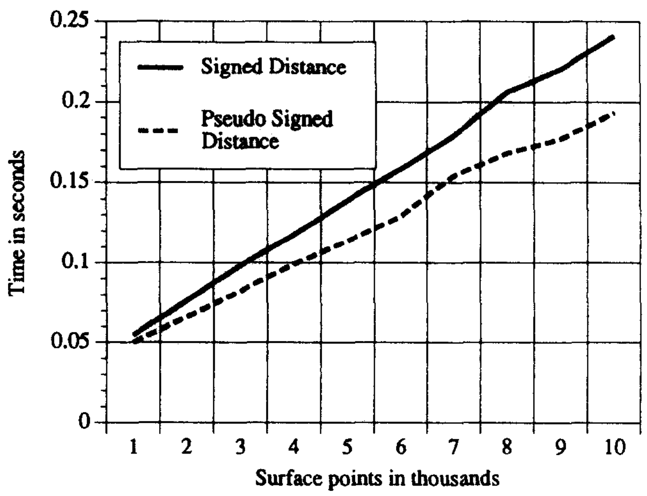


Figure 15 Wall clock time in relation to light-source rotation

is capable of providing the realtime response for the manipulation of highlight lines without excessive loss of accuracy, considered from the quality-visualization point of view.

Finally, the alternative formulation, i.e. the multiple highlight lines *versus* multiple highlight band boundaries, provides the user with the maximum flexibility of use. The alternative formulation is not available in the reflection-line and isophote models, but it certainly can act as an example to expand the context of these models as well.

ACKNOWLEDGEMENT

The authors would like to thank Electronic Data Systems and General Motors Corporation for sponsoring this study through generous research grants.

REFERENCES

1 Dill, J C and Rogers, D F 'Color graphics and ship surface curvature' *Proc. 4th Conf. Computer Application in the Automation of Shipyard Operation & Ship Design (ICCAS '82)* North-Holland (1982) pp 197–205

2 Poeschl, T 'Detecting surface irregularities using isophotes' *Comput. Aided Geom. Des.* Vol 1 No 2 (1984) pp 163–168

3 Klass, R 'Correction of local surface irregularities using reflection lines' *Comput.-Aided Des.* Vol 12 No 2 (1980) pp 73–76

4 Kaufman, E and Klass, R 'Smoothing surfaces using reflection lines for families of splines' *Comput.-Aided Des.* Vol 20 No 6 (1988) pp 312-316

5 Hagen, H, Schreiber, T and Gschwind, E 'Methods for surface interrogation' *Proc. 1st IEEE Conf. Visualization (Visualization '90)* IEEE Computer Society Press (1990) pp 187-193

6 Korn, G A and Korn, T M (Eds.) *Mathematical Handbook for Scientists and Engineers* (2nd Ed.) McGraw–Hill (1968) pp 71–72

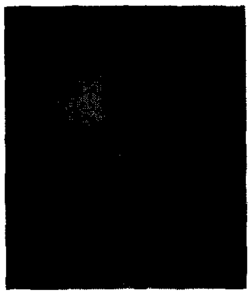
7 Beier, K-P and Chen, Y 'The highlight band, a simplified reflection model for interactive smoothness evaluation' *in* Sapidis, N (Ed.) *Designing Fair Curves and Surfaces* SIAM (to be published)

8 Chen, Y J and Ravani, B 'Offset surface generation and contouring in computer-aided design' *J. Mechanism, Transmissions & Automat. Des.* Vol 109 (1987) pp 133–142

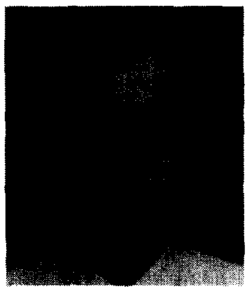
9 Preusser, A 'TRICP: a contour plot program for triangular meshes' *ACM Trans. Math. Soft.* Vol 10 (1984) pp 473–475

10 Palmer, J A B 'An economical method of plotting contours' *Austral. Comput. J.* Vol 2 No 1 (1970) pp 27–31

11 Sutcliffe, D C 'Contouring over rectangular and skewed rectangular grids — an introduction' *in* Brodlie, K W (Ed.) *Mathematical Methods in Computer Graphics and Design* Academic Press (1980) pp 39–62



Dr-Ing Klaus-Peter Beier is a research scientist with the College of Engineering at the University of Michigan, USA, and an adjunct associate professor of naval architecture and marine engineering. He joined the faculty in Ann Arbor in 1984. His current research activities include scientific visualization, free-form shape design, and virtual reality in industrial applications. K-P Beier received a PhD from the Technical University of Berlin, Germany, in 1976.



Yifan Chen received a BS and an MSE in naval architecture and ocean engineering from Shanghai Jiao Tong University, China, in 1985 and 1988, respectively. He pursued further graduate studies at the University of Michigan, USA, from 1989 to 1993, during which he received an MSE in industrial and operations engineering, and an MSE and PhD in naval architecture and marine engineering. He is currently an engineering specialist with the Ford Research Laboratory at the Ford Motor Company. His research interests include computer-aided geometric design, computer-aided manufacturing, and computer graphics.