High-Quality Rendering of Parametric Surfaces by Using a Robust Scanline Algorithm

Tomoyuki Nishita, Kazufumi Kaneda, and Eihachiro Nakamae

ABSTRACT

Displaying objects with high accuracy is seriously required not only for CAGD (Computer Aided Geometric Design) but also for the synthesis of photo-realistic images. Traditionally, polygonal approximation methods have been employed to display curved surfaces. They bring on low accuracy of display not only in shape but also in intensity of objects. In this paper a scanline algorithm to directly display surface patches, expressed by Bezier surfaces, without polygonal approximation is proposed. In our proposal, curved surfaces are subdivided into subpatches with curved edges intersecting with a scanline, and the intersections of every subpatch and the scanline are calculated. This method is extremely robust for calculation of the intersections, which can be obtained with only a few iterations. Furthermore, the greater the number of patches, the more effective the method is regarding required memory and calculation time. Anti-aliased images with shadows and texture mapping are given to show the effectiveness of the method proposed.

Key Words: Bezier Surfaces, Scanline algorithm, Robustness, High-quality rendering, Surface trimming, Silhouette detection, Shadowing

1 Introduction

Raytracing algorithms are a useful tool for rendering realistic images, but require extensive calculation time, while scanline algorithms can save calculation time. Traditionally, polygonal approximation methods have been employed to display curved surfaces. They can save calculation time and be easily implemented, but the displayed shape is not so accurate to a defined curved surface. To solve these problems some scanline algorithms rendering bicubic surfaces directly from parametric description have been proposed.

Blinn [Blinn 78] and Whitted [Whitted 78] employed the Newton-Raphson method to calculate the intersections of a scanline and curved surfaces. This method needs an initial guess and is not robust, and moreover a weakness in Whitted's approach is the lack of generality in finding silhouettes. To find silhouettes robustly, Schweitzer and Cobb [Schweitzer 82] proposed a method of dividing curved surfaces into polygons consisting of the boundary curves which are monotonic in y. In this method extraction of silhouettes is fairly complicated because several points on a silhouette need to be calculated by using normals of a curved surface approximated to cubic surfaces.

Lane, Carpenter [Lane 80] and Clark [Clark 79] rendered curved surfaces after subdividing them into small polygons. That is, curved surfaces are subdivided into subpatches until flat enough, then the subpatches are reckoned as polygons, and finally a polygon-oriented scanline algorithm is employed. In Clark's method curved surfaces are subdivided into subpatches before scan-conversion, while in Lane-Carpenter's method curved surfaces are dynamically subdivided for every scanline. Lane-Carpenter's method has the disadvantage that gaps arise between approximated polygons because of the difference between the approximated polygons and the original surface patches. This

problem is caused by straight lines consisting of the approximated polygons. In the subdivision methods taking into account surface flatness, only when the tolerance of surface flatness is within one pixel size, the silhouettes become smooth.

Griffiths [Griffiths 84] dealt with curved surfaces in the parametric plane. In this method, a curved surface is decomposed into grid cells, and some of them intersecting with a scanline are further divided. Then, linear interpolation is employed to get their depths and intensities. In this method silhouettes are extracted by using normal vectors stored in the grid cells. Pueyo and Brunet [Pueyo 87] improved upon Griffiths' method; they proposed that curved surfaces are decomposed into grid cells beforehand, and intersections of the restricted scanline and curved surfaces are calculated without interpolation (making use of the y-coordinates stored in the grid cells), and interpolation in parametric space is employed to intersections of the other scanlines. Silhouettes are detected by using the z-components of normal vectors stored in the grid points. One of the disadvantages of the previous two methods is missing those silhouettes formed by a smaller loop than the grid span.

Important elements for rendering curved surfaces are robustness (in the detection of silhouettes), accuracy, required memory, image quality (i. e., anti-aliasing and shadows), and calculation time. In all the above-mentioned methods, there are some problems, such as lack of robustness, inaccuracy caused by the approximations, large memory caused by beforehand subdivision. Our proposal overcomes all these problems.

The proposed method has the following advantages, although it is partially based on Lane-Carpenter's method. (a) Subpatches on a scanline are effectively obtained, and no gaps arise between the subpatches. (b) Proposed root finder using curve clipping is accurate and robust for calculation of intersections of the scanline and subpatches. (c) This method is especially effective for displaying scenes with a number of patches because only the curved surfaces intersecting with the active scanline are subdivided. (d) Anti-aliased images with shadows can be rendered.

2 Outline of the Algorithm

Bezier surfaces are used in this paper, because almost all surfaces, such as B-splines and NURBS, can be converted into Bezier surfaces. In this paper, curved surfaces are subdivided into subpatches on a scanline. These subpatches are processed as polygons with curved edges, and intersections of every subpatch and the scanline are calculated. Finally, traditional polygon-oriented scanline algorithms represented by Watkins's algorithm [Watkins 70] are employed to display images.

Outline of the proposed algorithm is as follows:

Step 1: Calculate a bounding box for each surface patch after the perspective transformation of every control point of each surface patch.

Step 2: Find surface patches intersecting with the scanline, and subdivide them into subpatches on the scanline.

Step 3: Calculate every span (called a scan segment) where subpatches intersect with the scanline.

Step 4: Find visible parts on each scan segment.

Step 5: Calculate shadow sections on each visible part.

Step 6: Display intensity of each pixel by using Phong's smooth shading algorithm.

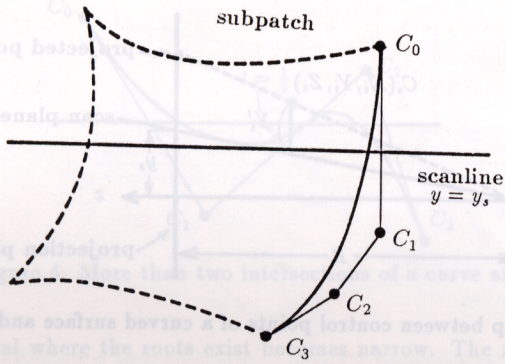


Figure 1: Intersection test of a boundary curve and a scanline.

In **step 1**, a convex hull property of each Bezier surface is useful to determine its bounding box. That is, the bounding box is determined by using the minimum and maximum values of the coordinates (x, y) of control points. In **step 2**, each subpatch is recursively subdivided until a desired degree of surface flatness is achieved. In **step 3**, the spans of scan segments are calculated by using intersections of a curved boundary of each subpatch and the scanline.

The proposed method basically belongs to polygon-oriented scanline algorithms, but **steps 2**, **3**, and **5** are different from them. In the following sections, these steps are discussed.

3 Intersections of Curves and a Scanline

In the proposed method, after subdivision of curved surfaces into subpatches on each scanline, subpatches are scan-converted into scan segments. For the convenience of explanation of this method, first the calculation method of the intersections between curves and the scanline in **step 3** is described, although it follows **step 2**.

Let's assume that a curved surface is defined by Bezier surface of degree n, and the boundary curves of the surface, $P(u, v)$, are expressed by $P(u, 0)$, $P(u, 1)$, $P(0, v)$, and $P(1, v)$. These curves are also expressed by Bezier curves of degree n. Then, each boundary curve, $C(t)$, in a space, is defined by the following.

$$C(t) = \sum_{i=0}^{n} C_i B_i^n(t), \tag{1}$$

where $C_i(X_i, Y_i, Z_i)$ $(i = 0, 1, \cdots, n)$ are control points and B is the Bernstein basis polynomial:

$$B_i^n(t) = \binom{n}{i} (1-t)^{n-i} t^i.$$

The calculation on the intersections of a projected curve $C(t)$ and a scanline, $y = y_s$ (see Fig. 1) is discussed in the following. Assumed that as shown in Fig.2 control points, C_i, are defined by the eye coordinate systems whose origin is set to the viewpoint and the z axis is perpendicular to the projection plane of which distance is R from the viewpoint, the points, (x, y, R), on a curve projected onto the projection plane can be defined by the following rational Bezier function.

$$\begin{aligned} x(t) &= \textstyle\sum_{i=0}^{n} X_i B_i^n(t)/z(t) \\ y(t) &= \textstyle\sum_{i=0}^{n} Y_i B_i^n(t)/z(t) \\ z(t) &= \textstyle\sum_{i=0}^{n} Z_i B_i^n(t)/R. \end{aligned} \tag{2}$$

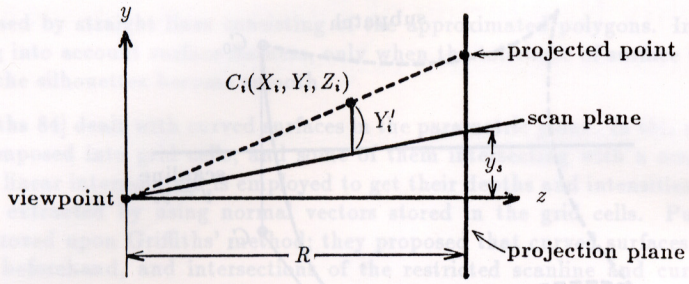


Figure 2: Relationship between control points of a curved surface and a scan plane.

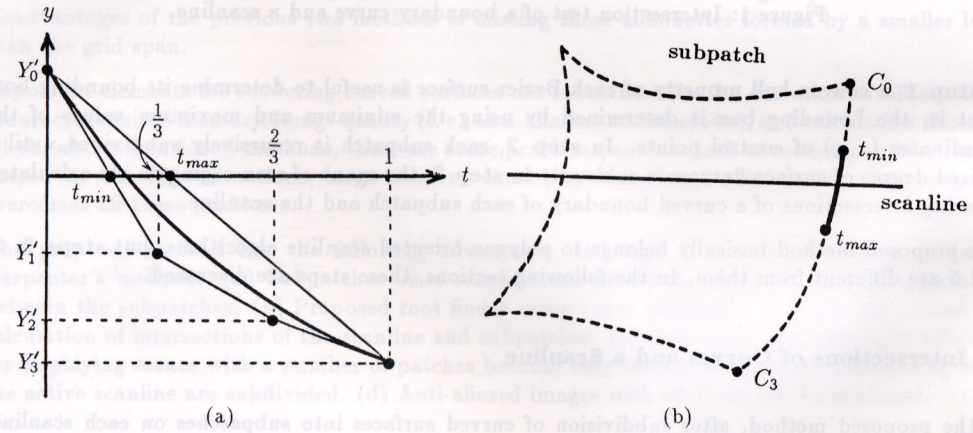


Figure 3: Calculation of intersections in parametric space and clipping a boundary curve.

where $z(t)$ is the depth defined as $z = 1$ at every point on the projection plane. As the line equation of the scan line is expressed by $y - y_s = 0$, the following equation is derived by substituting this into Eq. 2.

$$\sum_{i=0}^{n} Y_i' B_i^n(t) = 0, \tag{3}$$

where $Y_i' = Y_i - (y_s/R)Z_i$.

Y_i' corresponds to the distance in the y direction between the scan plane and a control point C_i as shown in Fig. 2. If a parameter, t, which satisfies Eq. 3 is calculated, then the depth, z, and x coordinates of the intersection on the projection plane can be derived.

The parameter t in Eq. 3 can be solved by using the Newton-Raphson method, because the equation is a polynomial of degree n, however, the calculation is not always robust. To overcome the problem, the authors propose the following method. When the curve intersects with the scanline, the root of Eq. 3 always exists between the intersections of the convex hull determined by the vertexes, $(i/n, Y_i')$ $(i = 0, 1, \cdots, n)$, and the t-axis, because the curve defined by Eq. 3 is non-parametric function (see Fig. 3 (a)), while the curve does not intersect with the scanline when Y_i' $(i = 0, 1, \cdots, n)$ are positive for all i or negative for all i. Only when Y_i' has both positive and negative value (in this condition the roots exist), the outside of the interval $[t_{min}, t_{max}]$ of the curve is clipped away as shown in Fig. 3 (a). As the intersections of the convex hull formed by the control points of the clipped curve (as shown in Fig. 3 (b)) and the t-axis are recursively

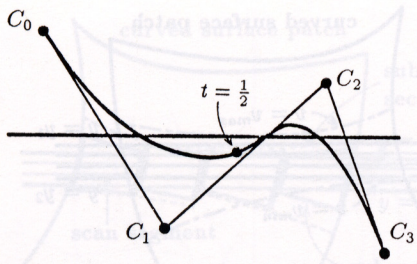


Figure 4: More than two intersections of a curve and a line.

calculated, the interval where the roots exist becomes narrow. The root of t converges with the advance of the clipping process. If Eq. 3 has more than two roots, the ratio of the convergency of the interval becomes small. In this case the curve is divided into two curves (divided at the point $t = 1/2$ in Fig. 4), and the same process as mentioned above is continued for each divided curve; from then, all of the roots can be calculated. This clipping method consists of two processes, that is, calculation of the interval of t and clipping a curve. Generally speaking, the latter process needs extensive calculation time because curves in a space have three components, x, y, and z, while in our method, calculation time can be saved because only one component, Y', is used to clip a curve; de Casteljau's subdivision algorithm is employed for the clipping.

4 Clipping a Curved Surface

The method generating for subpatches intersecting with the scanline is described here. In Lane and Carpenter's method, a curved surface is subdivided into four subpatches. The subpatches on the scanline are extracted, and the subdivision is recursively continued until surface flatness is satisfied. In this method, not only the subdivision process but also the extracting of the subpatches existing on the scanline are necessary. Authors propose a more efficient method of generating subpatches on the scanline. An area of a curved surface is recursively subdivided by clipping away where the surface does not intersect with the scanline. In many cases a subpatch intersects with several scanlines. In this paper, subpatches intersecting with every few scanlines (e. g., every three or four scanlines) are generated, and the intersections of the boundary curves of subpatches and the scanline are calculated.

Assumed that the coordinates of a control point are (X_{ij}, Y_{ij}, Z_{ij}) in the eye coordinate systems, Bezier surfaces of degree n are defined as follows:

$$\begin{aligned} x(u,v) &= \sum_{i=0}^{n} \sum_{j=0}^{n} X_{ij} B_i^n(u) B_j^n(v) \\ y(u,v) &= \sum_{i=0}^{n} \sum_{j=0}^{n} Y_{ij} B_i^n(u) B_j^n(v) \\ z(u,v) &= \sum_{i=0}^{n} \sum_{i=0}^{n} Z_{ij} B_i^n(u) B_i^n(v). \end{aligned} \tag{4}$$

When the range for generating for subpatches is a band between the two scanlines, y_1 and y_2 $(y_1 > y_2)$ as shown in Fig. 5, equations of the scan planes determined by each scanline and the viewpoint are as follows:

$$\begin{aligned} y - (y_1/R)z &= 0 \\ y - (y_2/R)z &= 0. \end{aligned} \tag{5}$$

If the functions, f_1 and f_2, are expressed as following:

$$\begin{aligned} f_1(y,z) &= y - (y_1/R)z \\ f_2(y,z) &= y - (y_2/R)z, \end{aligned}$$

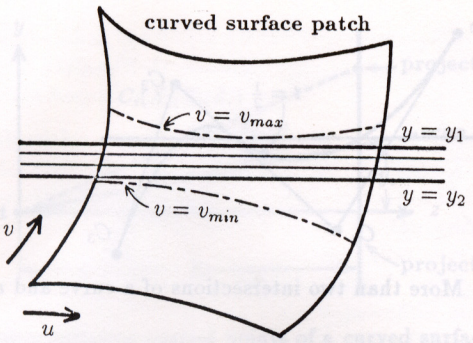


Figure 5: Intersection of a group of scanlines and a surface patch.

the region formed by these two scanlines satisfies the condition, $f_1 \leq 0$ and $f_2 \geq 0$. From this, the interval of u and v intersecting with the scan planes defined by Eq. 5 satisfies the following conditions:

$$\begin{aligned} f_1 &= \sum_{i=0}^{n} \sum_{j=0}^{n} (Y_{ij} - (y_1/R) Z_{ij}) B_i^n(u) B_j^n(v) \leq 0 \\ f_2 &= \sum_{i=0}^{n} \sum_{j=0}^{n} (Y_{ij} - (y_2/R) Z_{ij}) B_i^n(u) B_j^n(v) \geq 0. \end{aligned} \tag{6}$$

That is,

$$\begin{aligned} \sum_{i=0}^{n} \sum_{j=0}^{n} f_1(Y_{ij}, Z_{ij}) B_i^n(u) B_j^n(v) \leq 0 \\ \sum_{i=0}^{n} \sum_{j=0}^{n} f_2(Y_{ij}, Z_{ij}) B_i^n(u) B_j^n(v) \geq 0. \end{aligned} \tag{7}$$

The clipping described in the previous section is also available to calculate the interval of u and v in Eq. 7.

A curved surface is clipped taking into account the obtained interval of u and v. If flatness of the clipped surface, a subpatch, is not enough, the subpatch is divided into two subpatches and this process is continued until every subpatch satisfies the flatness tolerance given in advance. Finally, several subpatches intersecting with the scanline are generated as shown in Fig. 6. Surface flatness is measured by the maximum distance between the curved surface and the plane determined by three corner control points [Whitted 78] [Lane 80].

5 Hidden Surface Removal and Shading

After generating subpatches, a line segment between intersections of a subpatch and each scanline located between $y = y_1$ and $y = y_2$ is calculated by using the method described in the previous section. This line segment is called a scan segment. Scan segments can be reckoned as straight lines because subdivided subpatches are flat enough. Except for the original boundaries, boundary curves consisting of subpatches are classified into two types. One is a boundary caused by clipping away the outside of the region determined by the interval of u or v obtained by Eq. 7, and the other is a boundary caused by dividing a subpatch into two subpatches. In the calculation of intersections of subpatches and the scanline, it is enough to examine only the latter's boundaries (usually, two edges as shown in bold lines in Fig. 6) because the former never intersect with the scanline.

After the calculation of scan segments, visible parts of each scan segment are determined by using a traditional scanline algorithm, and Phong's shading algorithm is employed to display curved surfaces. In this paper penetrations of each curved surface are allowed; hidden surfaces are removed by taking account of the intersection of scan segments in the depth direction.

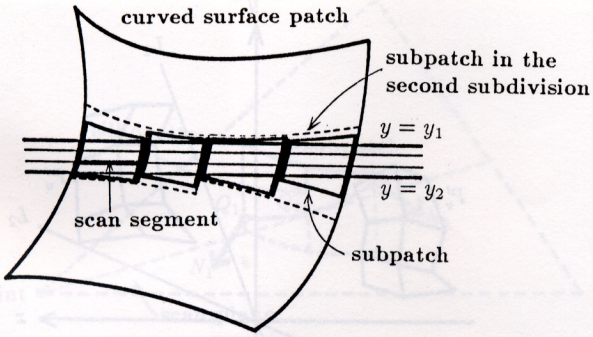


Figure 6: Subpatches on a scanline.

For the visibility test and smooth shading mentioned above, a depth, z, and a normal vector at both endpoints of a scan segment are required. Normal vectors at both endpoints of a scan segment are only slightly different because of flatness of the subpatch. When the difference between the normal vectors at both endpoints is greater than a threshold, the subpatch is divided again. Therefore, accurate intensities can be obtained, even if a linear interpolation is employed to calculate the normal vectors on the scan segment.

To discuss accuracy of shading proposed here, some previous methods are referred here. In Schweitzer's method [Schweitzer 82], a cubic interpolation is employed to calculate normal vectors on a scan segment. However, the accuracy of the normal vectors is not enough to faithfully display an original curved surface, because the normal vectors on the scan segment are interpolated by using only the normal vectors at the intersections of the scanline and a fairly large subpatch (and/or a silhouette.) In Pueyo's method [Pueyo 87], a curved surface is decomposed (equidistantly in parametric space) into grid cells, and linear interpolation is employed to the normal vectors on the grid cells. As the difference between the normal vectors at the adjacent grid points depends upon the size of grid cells, the size of grid cells should be small enough in order to obtain accurate normal vectors. In order to get more precise coordinates and normals at pixels within the scan segments, marching technique, such as Satterfield and Rogers' method[Satter 85], may be effective, even though their method is developed for generating contour lines from a B-spline after triangular mesh approximation.

6 Silhouettes of a Curved Surface

The detection of accurate silhouettes of a curved surface is one of the important elements for displaying realistic curved surfaces. As described in the introduction, traditional calculation methods are not always robust nor so accurate for silhouette edges [Blinn 78] [Whitted 78] [Schweitzer 82]. Even some methods addressing these problems still have some disadvantages, such as very complicated process and missing silhouettes formed by relatively small loops. In the subdivision methods taking into account surface flatness [Lane 80] [Clark 79], only when the tolerance of surface flatness is within one pixel, the silhouettes become smooth; in this case every curved surface needs to be excessively subdivided even though it dose not always have silhouettes. To solve these problems, the authors propose the following efficient subdivision method.

It is useful to classify all curved surfaces into two types – those which probably have silhouettes, and those which never have them before scan-conversion because all the curved surfaces do not necessarily have silhouettes. After subdividing curved surfaces into subpatches on the scanline, only the subpatches probably having the silhouettes are examined as to whether they really have

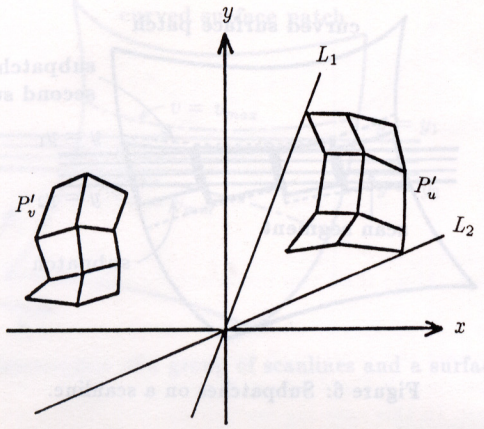


Figure 7: Hodograph for calculating directions of normals.

silhouettes or not. If they have, the subdivision process is applied until the flatness is satisfied. (The lower tolerance of flatness should be set to the subpatches with silhouette.) This method never leaves any silhouette undetected. In the following, how to examine whether or not a curved surface probably has a silhouette is discussed.

First, the classification into a front face, a back face, or the other (i. e., surfaces probably having silhouettes), is necessary. A front face has a normal vector toward the viewpoint throughout the surface (an inner product of the viewing vector and the normal vector is positive), while a back face has a normal vector away from the viewpoint.

The normal vector at a point (u, v) is defined by a vector product of $P_u(u, v)$ and $P_v(u, v)$ derived from the derivative of $P(u, v)$ with regard to u and v, respectively. Let's assume that the normal vector, $P_u \times P_v$, is defined as the direction toward the outside of the surface. If the condition, $P(u, v) \cdot P_u(u, v) \times P_v(u, v) < 0$, is satisfied for all u $(0 \leq u \leq 1)$ and v $(0 \leq v \leq 1)$ in the eye coordinate systems, then the curved surface is a front surface toward the viewpoint. To find out the curved surfaces having silhouettes this condition may be used; however, the calculation cost is probably very high. To address this problem, in this paper the projected control points of curved surfaces (so called 'wedge test') is used.

After perspective transformation the direction of a surface can be tested by using only the signs of the z-component of the normal vectors. That is, only the x and y components of P'_u and P'_v are used for the test where P' is a projected curved surface. Derivatives of Bezier surfaces can be obtained from the control points geometrically. For example, the control points $P'_{ui,j}$ of P'_u is expressed by $n(P'_{i+1,j} - P'_{i,j})$ $(i = 0, 1, \cdots, n-1,\ j = 0, 1, \cdots, n)$. The range of P'_u and P'_v can be obtained by hodograph [Sederberg 88] as shown in Fig. 7. The direction of the tangent with respect to the parameter u exists in the wedge area intercepted by the lines L_1 and L_2 which holds P'_u as shown in Fig. 7. The equations of the lines L_1 and L_2 are defined by

$$\begin{aligned} f_1(x, y) &= a_1 x + b_1 y = 0 \\ f_2(x, y) &= a_2 x + b_2 y = 0. \end{aligned} \tag{8}$$

Every control point P'_{uij} belonging to P'_u locates in the positive side of the line L_1 while P'_{uij} in the negative side of the line L_2. That is, the straight lines satisfy the following conditions.

$$\begin{aligned} &\forall P_{uij} : f_1(x_{uij}, y_{uij}) \geq 0 \\ &\forall P_{uij} : f_2(x_{uij}, y_{uij}) \leq 0. \end{aligned} \tag{9}$$

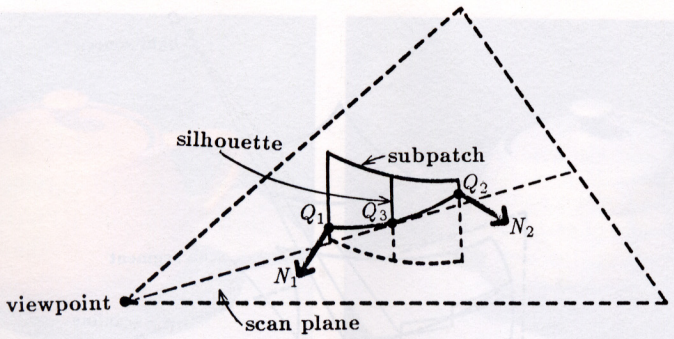


Figure 8: Subpatch with a silhouette.

Therefore,

$$\begin{array}{l} \text{(a) If } \forall P_{vij} : f_1(x_{vij}, y_{vij}) \geq 0 \text{ and } f_2(x_{vij}, y_{vij}) \geq 0 \text{ then a front face.} \\ \text{(b) If } \forall P_{vij} : f_1(x_{vij}, y_{vij}) \leq 0 \text{ and } f_2(x_{vij}, y_{vij}) \leq 0 \text{ then a back face.} \\ \text{(c) Other than the above, a twisted face.} \end{array} \quad (10)$$

In other words, if any P_v overlaps with the wedge intercepted by L_1 and L_2, then the curved surface is a twisted surface. Note that the curved surface classified into a twisted surface does not always have a silhouette. For a more detailed test the curved surface must be further subdivided, but, it is enough to find out only the probability of having silhouettes. In this process of face classification, if L_1 and L_2 cannot be obtained, the surface probably has silhouettes.

Curved surfaces requiring scan-conversion can be reduced in number by culling back faces in the step of the wedge test, because of the invisibility of back faces. (15 percent of the total CPU time was cut down by culling back faces in our experiment.)

A bisectional method is employed to calculate the intersection of silhouettes and each scanline, because the interval between the parameters at the endpoints of a scan segment is very small. That is, if normal vectors N_1 and N_2 at the endpoints (Q_1 and Q_2 in Fig. 8) of a segment have an opposite direction with respect to the direction of the viewpoint, subdivision of the subpatch is executed, because the scanline intersects with the silhouette (at Q_3 in Fig. 8.) The subdivision process is continued until the viewpoint direction component of both normal vectors at the endpoints becomes smaller than a specified tolerance.

7 Shadowing

Even though the papers referred to here use scanline algorithms ([Blinn 78] [Whitted 78] [Lane 80] [Clark 79] [Schweitzer 82] [Griffiths 84] [Pueyo 87]) none of them mention shadowing. Shadows are one of the important elements for displaying realistic images. Calculation method for shadowed sections on each scan segment is discussed here. Shadows are invisible sections when a light source is reckoned as a viewpoint. Therefore, a shadow cast by a curved surface can be calculated as follows. If a triangle formed by both endpoints of a scan segment and the viewpoint (see Fig. 9) intersects with a curved surface, the shadows due to the curved surface are cast on at least a part of the scan segment. By reckoning the light source as a viewpoint, the triangle formed by the scan segment and the light source is treated as a scan plane; the scanline algorithm mentioned before can be employed. That is, the shadow sections on the scan segment can be determined by using the intersection test between the scanline and curved surfaces when viewed from the light source.

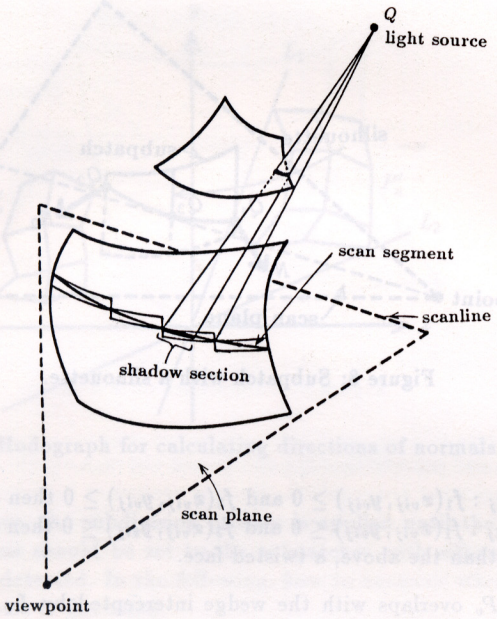


Figure 9: Shadow area on a scan segment.

Table 1: Relationship between the tolerance, the average iterations, and the CPU time (in the case of (a)).

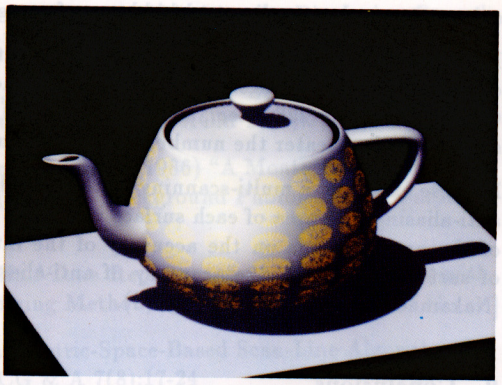
tolerance	average iterations	CPU time (sec.)
10^{-3}	1.6	27.3
10^{-5}	2.1	28.9
10^{-7}	2.4	29.1

8 Examples

Four examples are shown in Fig. 10. Fig. (a) shows a tea pot as an example of standard data. It consists of 32 patches and the CPU time was 27.3 sec. (144.6 sec. in case with shadows) by using IRIS-4D/120GTX and the size of screen is 500×500.

Tab. 1 shows relationships among the tolerance of the parameter t regarding the calculation of intersections of a scanline and subpatches, the average number of iterations, and the CPU time. It is evident that iterations are fairly few, because of quick convergence due to the fact that the shapes of boundaries of every subdivided subpatch are close to straight segments. When the tolerance is set at 10^{-3}, even one patch covering all of the screen size, 1000×1000, is displayed with the accuracy within one pixel. Therefore, it is set at 10^{-3} in the examples. Even when the resolution of the screen becomes higher and the tolerance is set at 10^{-7}, the average number of iterations increases by only 0.8, and the CPU time increases by less than 1 percent. Calculation time for generating subpatches takes 60 percent of all calculation time, and for calculating intersections of

(a)



(b)

(c)

(d)

Figure 10: Examples.

subpatches and a scanline and hidden surface removal 22 percent.

Fig. (b) shows an example of texture mapping to Fig. (a). Fig. (c) shows a base and an ash tray consisting of 133 patches. In case Fig. (d) depicting hot air balloons considered foggy effect, the CPU time is only 228.8 sec., although all of the balloons consist of 1064 patches. These examples show that the greater the number of patches, the more effective the method is.

In these examples a multi-scanning method [Nishita 84] developed by the authors was employed for anti-aliasing; the area of each surface in a pixel is calculated by trapezoidal integral; the precision of the area depends on the accuracy of the intersections between sub-scanlines and boundaries of surfaces. That is, the accuracy of anti-aliasing is improved. Anti-aliased image composition [Nakamae 86] were used for Fig. (d).

9 Conclusions

A robust and fast method for rendering parametric surfaces is proposed. The method has the following advantages:

1. Boundaries of displayed objects are quite faithful to the defined curved surfaces, and therefore extremely smooth, because intersections of subpatches and each scanline are calculated accurately and robustly. No gap arises between subpatches. Intersections can be calculated by few iterations (about 2 iterations.)
2. Even in the area where the changing of normal vectors is drastic, the intensity is accurate, because curved surfaces are sampled in proportion to curvature of the surfaces.
3. Shadowed boundaries are always smooth, because shadows faithful to the defined curved surfaces can be displayed.
4. In terms of calculation time it is low cost especially when a number of patches exist in a scene.
5. The accuracy of anti-aliasing is improved because of calculating the accurate intersections.

The method proposed here is applied for Bezier surfaces. Developing the scanning method for rendering directly from the other types of surfaces, such as B-spline, is expected, even though almost all surfaces can be converted into Bezier surfaces.

Acknowledgment

We would like to thank Dr. Thomas W. Sederberg for his discussion about the method for calculating intersections by surface clipping when the first author stayed at Brigham Young University. We are also grateful to Mr. Mikio Munetomo for his coding anti-aliasing and texture mapping.

REFERENCES

[Blinn 78] Blinn JF (1978) "Simulation of Wrinkled Surfaces," Computer Graphics 12(3):286-292

[Clark 79] Clark JH (1979) "A First Scan-Line Algorithm for Rendering Parametric Surfaces," Computer Graphics 13(2):174

[Griffiths 84] Griffiths JG (1984) "A Depth-Coherence Scanline Algorithm for Displaying Curved Surfaces," CAD 16(2):91-101

[Lane 80] Lane JM, Carpenter LC, Whitted T, and Blinn JF (1980) "Scan Line Methods for Displaying Parametrically Defined Surfaces," Comm. ACM 23(1):23-34

[Nakamae 86] Nakamae E, Harada K, Ishizaki T, and Nishita T (1986) "A Montage: The Overlaying of the Computer Generated Images onto a Background Photograph," Computer Graphics 20(4):207-214

[Nishita 84] Nishita T and Nakamae E (1984) "Half-Tone Representation of 3-D Objects with Smooth Edges by Using a Multi-Scanning Method," Trans. IPSJ 25(5):703-711

[Pueyo 87] Pueyo X and Brunet P (1987) "A Parametric-Space-Based Scan-Line Algorithm for Rendering Bicubic Surfaces," IEEE CG & A 7(8):17-24

[Satter 85] Satterfield SG and Rogers DF (1985) "A Procedure for Generating Contour Lines From a B-Spline Surface," IEEE CG & A 5(4):71-75

[Schweitzer 82] Schweitzer D and Cobb ES (1982) "Scanline Rendering of Parametric Surfaces," Computer Graphics 16(3):265-271

[Sederberg 88] Sederberg TW and Meyers RJ (1988) "Loop Detection in Surface Patch Intersections," CAGD 5(2):161-171

[Watkins 70] Watkins GS (1970) "A Real-Time Visible Surface Algorithm," University of Utah Compt. Sc. Dept. UTEC-CSC-70-101, NTIS AD-762 004

[Whitted 78] Whitted T (1978) "A Scan Line Algorithm for Computer Display of Curved Surfaces," Computer Graphics 12(3):26

Tomoyuki Nishita is an associate professor in the department of Electronic and Electrical Engineering at Fukuyama University, Japan. He was on the research staff at Mazda from 1973 to 1979 and worked on design and development of computer-controlled vehicle system. He joined Fukuyama University in 1979. He was an associate researcher in the Engineering Computer Graphics Laboratory at Brigham Young University from 1988 to the end of March, 1989. His research interests involve computer graphics including lighting model, hidden-surface removal, and antialiasing.
Nishita received his BE, ME and Ph. D in Engineering in 1971, 1973, and 1985, respectively, from Hiroshima University. He is a member of ACM, IPS of Japan and IEE of Japan.
Address: Faculty of Engineering, Fukuyama University, Sanzo, Higashimura-cho, Fukuyama, 729-02 Japan.

Kazufumi Kaneda is a research associate in Faculty of Engineering at Hiroshima University. He worked at the Chugoku Electric Power Company Ltd., Japan from 1984 to 1986. He joined Hiroshima University in 1986. His research interests include computer graphics and image processing.
Kaneda received the BE and ME in 1982 and 1984, respectively, from Hiroshima University. He is a member of IEE of Japan, IPS of Japan and IEICE of Japan.
Address: Faculty of Engineering, Hiroshima University, Saijo-cho, Higashi-hiroshima, 724 Japan.
E-mail: kin@eml.hiroshima-u.ac.jp

Eihachiro Nakamae is a professor at Hiroshima University where he was appointed as research associate in 1956 and a professor in 1968. He was an associate researcher at Clarkson College of Technology, Potsdam, N. Y., from 1973 to 1974. His research interests include computer graphics and electric machinery.
Nakamae received the BE, ME, and DE degrees in 1954, 1956, and 1967 from Waseda University. He is a member of IEEE, IEE of Japan, IPS of Japan and IEICE of Japan.
Address: Faculty of Engineering, Hiroshima University, Saijo-cho, Higashi-hiroshima, 724 Japan.
E-mail: naka@eml.hiroshima-u.ac.jp