appeared in Proceedings of PACIFIC GRAPHICS 2000, p. 413–414, IEEE Computer Society Press, 2000
PACIFIC GRAPHICS 2000, October 3–5 2000, Hong Kong, China.

Triangular Bézier Clipping

S. H. Martin Roth, Patrick Diezi, Markus H. Gross

Computer Science Department, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland
email: {roth, grossm}@inf.ethz.ch

Abstract

This short paper introduces a new approach to finding ray–patch intersections with triangular Bernstein–Bézier patches of arbitrary degree. Unlike a previous approach based on a combination of hierarchical subdivision and a Newton–like iteration scheme [7], this work extends the concept of Bézier clipping to the triangular domain.

The problem of reporting wrong intersections, inherent to the original Bézier clipping algorithm [5], is investigated and opposed to the triangular case. It turns out that reporting wrong hits is very improbable, even close to impossible, in the triangular set-up.

1 Introduction and related work

The basic requirement for ray tracing of objects is the computation of intersections between a ray and the surface description. For free-form surfaces, this task is referred to as the *ray–patch intersection problem.* This short paper describes a new approach to this problem for triangular Bézier patches of arbitrary degree [3] based on Bézier clipping [5]. For a full version of this paper, please see [6].

In general, the ray–patch intersection problem with free-form surfaces must be solved iteratively. Methods used to find parametric intersections divide into two categories:

- *nested bounding volumes:* bounding spheres, axes-aligned bounding boxes, oriented slabs, parallelepipeds [1], Chebyshev boxing [4], bounding volume hierarchy [2]
- *parameter interval iteration:* Bézier clipping [5, 2]

Only Stürzlinger addressed the problem of ray tracing triangular free-form surfaces [7]. His algorithm must be attributed to the class of nested bounding volumes using skewed triangular prisms as bounding primitives and Newton iteration similar to [1]. The next section presents an extension of Bézier clipping to the triangular domain.

2 Triangular Bézier Clipping

We denote a triangular (barycentric) Bézier patch as:

$$\mathbf{b}^n(r, s) = \sum_{j=0}^{n} \sum_{i=0}^{n-j} \mathbf{b}_{ij} B_{ij}^n(r, s) \qquad (1)$$

with Bernstein polynomials $B_{ij}^n(r, s) = \frac{n!}{i!\,j!\,(n-i-j)!} r^i s^j (1-r-s)^{n-i-j}$ and on the assumption that $i, j \geq 0$.

The ray–patch intersection problem refers to the task of finding the intersections of a ray

$$\mathbf{r}(u) = \mathbf{r}_0 + u \cdot \mathbf{r}_d, \qquad u \in \mathrm{R}^+, \qquad \|\mathbf{r}_d\| = 1\,. \qquad (2)$$

with a Bézier patch according to (1). As do [5] we represent the ray as the intersection of two orthogonal planes given by their normalized implicit equations

$$a_k x + b_k y + c_k z + e_k = 0, \qquad k \in \{0, 1\} \qquad (3)$$

with $a_k^2 + b_k^2 + c_k^2 = 1$ (see Fig. 1a).

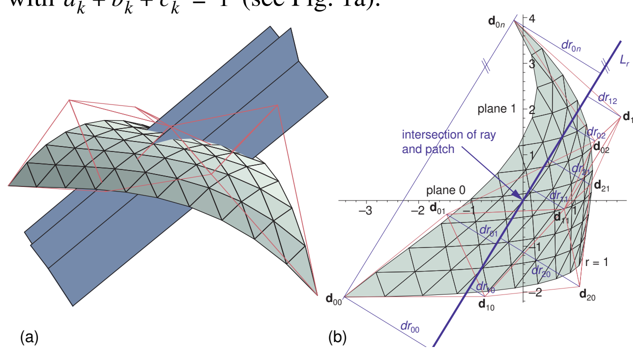

Fig. 1: (a) Representation of a ray by two orthogonal planes
(b) Problem reduction and distances dr_{ij} to L_r

In complete analogy to [5, 2] the problem of finding an intersection $\mathbf{r}(u) = \mathbf{b}^n(r, s)$ can be reduced from three to two dimensions even if the patch $\mathbf{b}^n(r, s)$ is rational. This is accomplished by substituting (1) into (3) which yields

$$\mathbf{d}^n(r, s) = \sum_{j=0}^{n} \sum_{i=0}^{n-j} \mathbf{d}_{ij} B_{ij}^n(r, s) \stackrel{!}{=} \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad (4)$$

with
$$\mathbf{d}_{ij} = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} x_{ij} + \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} y_{ij} + \begin{pmatrix} c_0 \\ c_1 \end{pmatrix} z_{ij} + \begin{pmatrix} e_0 \\ e_1 \end{pmatrix}$$

and the coordinates x_{ij}, y_{ij}, and z_{ij} of the control points of the patch. The components 0 and 1 (in the remainder referred to as x and y) of $\mathbf{d}_{ij}$ geometrically represent the distance of the point to plane 0 and 1, respectively (see Fig. 1b). The problem now reduces to finding the roots of (4).

Bézier clipping basically clips away regions in the parametric domain which are known not to intersect the patch. For tensor product surfaces, Nishita et al. in [5] determine both (u_{min}, u_{max}) and (v_{min}, v_{max}) of the parametric candidate region for an intersection with the ray (see Fig. 2a).

A similar approach on the triangular domain of the barycentric coordinates r, s and t in general yields a complex non-triangular candidate region (see Fig. 2b, striped region). A triangular upper bound of this region can be

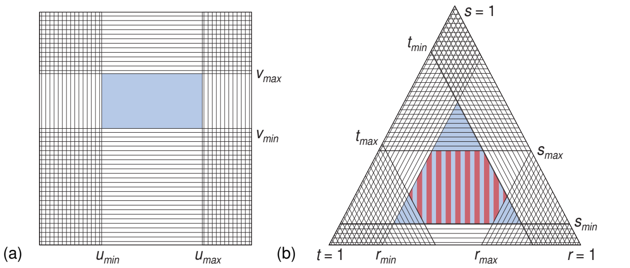

Fig. 2: Parametric candidate region (shaded) of tensor product domain (a) and barycentric domain (b)

found using only r_{min}, s_{min} and t_{min} (see Fig. 2b, shaded region). In the following, the procedure of finding r_{min} on the example of a cubic patch will be illustrated.

Firstly, we determine a line L_r parallel to the vector from $\mathbf{d}_{00}$ to $\mathbf{d}_{0n}$ through the origin. Expressing this line in its implicit form

$$ax + by + c = 0, \qquad a^2 + b^2 = 1 \tag{5}$$

yields the distances dr_{ij} of the control points $\mathbf{d}_{ij} = (x_{ij}, y_{ij})$ to the line L_r as

$$dr_{ij} = ax_{ij} + by_{ij} + c, \qquad 0 \le i + j \le n \tag{6}$$

with $c = 0$ since the line touches the origin (see Fig. 1b).

The distance $dr^n(r, s)$ of arbitrary points to L_r becomes

$$dr^n(r, s) = \sum_{j=0}^{n} \sum_{i=0}^{n-j} dr_{ij} B_{ij}^n(r, s) \tag{7}$$

This distance function $dr^n(r, s)$ can be regarded as a functional surface over the triangular domain with control points $\mathbf{dr}_{ij} = (r_i, s_j, dr_{ij})$ and equidistant $r_i = i/n$ and $s_j = j/n$ for $0 \le i + j \le n$. In a next step, the functional surface is projected along L_r. The clipping value r_{min} can now be found by intersecting the convex hull of the projected distances with the r coordinate axis. The steps for s_{min} and t_{min} follow from symmetry.

For parametric values below those minima there cannot be an intersection due to the convex hull property of Bézier patches. Further, the ray does not intersect the patch if

$$r_{min} + s_{min} + t_{min} > 1 \tag{8}$$

If the parametric candidate domain is not yet small enough to assure sub-pixel accuracy, the patch is subdivided according to the minima found and the clipping procedure continues on the resulting sub-patch. Otherwise, the centroid of r_{min}, s_{min} and t_{min} is taken as the parametric point of intersection.

3 Reporting of wrong hits

The original Bézier clipping algorithm can report wrong intersections whenever the convex hull of projected distances intersects the corresponding parameter axis even if the patch actually does not [2, 6]. A potentially wrong intersection is reported if this happens for both tensor product directions u and v or if the other candidate domain collapses because the projection converts to a line.

There are several reasons why a similar situation for triangular patches is difficult to construct. First and possibly most important, the algorithm only makes use of the minima r_{min}, s_{min} and t_{min}. Second, distances are computed with respect to three coordinate directions, which due to the barycentric setting are linearly dependent and thus to some respect redundant. Consequently, wrong intersections can only occur if both of the following conditions hold:

- In at least one projective view of distances, one non-interpolating control point lies above or below the respective axis and the others do not.
- The minima r_{min}, s_{min} and t_{min} accidentally sum up to approximately but not more than 1.

It is the second condition that makes triangular Bézier clipping far less error-prone. Please see [6] for the details.

4 Results

In order to test the implementation, we converted the Utah teapot from 32 bicubic tensor product patches to 64 sextic triangular Bézier patches. Fig. 3 shows the sextic control net and corresponding ray traced images. Pairs of red and blue triangular patches represent original bicubic patches.

Fig. 3: (a) Control net of Utah teapot made of 64 triangular sextic Bézier patches. (b),(c) Corresponding ray traced images

Since the cost of the clipping operation grows with the cube of the patch's degree, ray tracing of sextic patches using Bézier clipping is roughly ten times slower than using nested bounding volumes. For patches of lower degree, the efficiency of Bézier clipping improves.

References

[1] W. Barth and W. Stürzlinger. "Efficient Ray Tracing for Bézier and B-spline Surfaces." *Computers & Graphics*, 17(4):423–430, 1993.

[2] S. Campagna, P. Slusallek, and H.-P. Seidel. "Ray Tracing of Spline Surfaces: Bézier Clipping, Chebyshev Boxing, and Bounding Volume Hierarchy – a Critical Comparison with New Results." *The Visual Computer*, 13(6):265–282, 1997.

[3] G. Farin. "Triangular Bernstein-Bézier Patches." *Computer Aided Geometric Design*, 3(2):83–128, 1986.

[4] A. Fournier and J. Buchanan. "Chebyshev Polynomials for Boxing and Intersections of Parametric Curves and Surfaces." In *Proc. EUROGRAPHICS'94,* COMPUTER GRAPHICS Forum, (13)3:127–142, 1994.

[5] T. Nishita, T. W. Sederberg, and M. Kakimoto. "Ray Tracing Trimmed Rational Surface Patches." In *Computer Graphics (SIGGRAPH'90 Proc.)*, volume 24, pages 337–345, 1990.

[6] S. H. M. Roth, P. Diezi, and M. H. Gross. "Triangular Bézier Clipping." Technical Report 347, Institute of Scientific Computing, ETH Zurich, 2000.

[7] W. Stürzlinger. "Ray Tracing Triangular Trimmed Free Form Surfaces." *IEEE Transactions on Visualization and Computer Graphics*, 4(3):202–214, 1998.