Title

A tessellation algorithm for the representation of trimmed NURBS surfaces with arbitrary trimming curves

Journal

Computer-Aided Design, 28

Authors

Hamann, Bernd
Tsai, P.-Y.

Publication Date

1996

Peer reviewed

ELSEVIER

0010-4485(95)00043-7

Computer-Aided Design, Vol. 28, No. 6/7, pp. 461–472, 1996
Published by Elsevier Science Ltd
Printed in Great Britain
0010-4485/96 $15.00 + 0.00

A tessellation algorithm for the representation of trimmed NURBS surfaces with arbitrary trimming curves

Bernd Hamann and Po-Yu Tsai*

In most CAD systems, a trimmed parametric surface is defined by two things: the control information of the surface itself and the control information of its trimming curves, which are usually defined as parametric curves in the parameter space of the surface. Often, trimmed parametric surfaces cause problems in the context of data exchange between CAD systems, surface evaluation/rendering, and grid/mesh generation. This paper describes a new approach for representing a trimmed surface. The approach is based on the idea of decomposing the valid part of the parameter space of a trimmed surface by a set of planar, ruled surfaces, which are defined in parameter space and whose union defines the entire valid part. A generalized Voronoï diagram is constructed for the set of trimming curves in parameter space. This diagram defines tiles around each trimming curve, which are further tessellated by utilizing a scan line technique for identifying and filling the interior of polygons. Here, the scan line technique is used to extract non-horizontal segments of the given trimming curves. Pairs of these segments are then linearly interpolated in horizontal parameter direction, thus defining a set of planar, ruled surfaces. The set of all these planar, ruled surfaces defines the entire valid part of a trimmed surface exactly. Published by Elsevier Science Ltd

Keywords: bisector, NURBS, scan-line algorithm, trimmed surface, Voronoï diagram

INTRODUCTION

This paper is concerned with the conversion of parametric surfaces containing trimming curves to a set of approximating surfaces that do not contain any trimming curves. Trimmed surfaces arise frequently in real-world applications. Typically, they are the result of surface–surface intersection (SSI). Complex geometries are defined in terms of thousands of parametric surfaces which might intersect each other. The intersection curves are usually defined in the parameter space of the surfaces, e.g. by a set of planar Bézier, B-spline, or NURBS (non-uniform rational B-spline) curves.

Many CAD systems cannot represent a trimmed parametric surface *implicitly*, i.e. as a parametric surface with the trimming curves defined in its parameter space. This causes a problem when exchanging trimmed surface data between CAD systems. A solution to this problem is the representation of the valid part (i.e. the part that remains when disregarding all holes implied by the trimming curves) by a set of *basic* parametric surfaces, i.e. parametric surfaces without trimming curves. Furthermore, the representation of a trimmed surface by a set of *basic* surfaces simplifies grid/mesh generation and surface rendering.

This paper introduces a new approach for the representation of trimmed surfaces. The complement of the part that is 'cut out' by the trimming curves is defined by means of decomposing the valid part of the parameter space into a set of four-sided regions. In the following, only tensor product surfaces will be considered. They are denoted by

$$\mathbf{s}(u,v) = (x(u,v), y(u,v), z(u,v)) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{d}_{i,j} \phi_i(u)\, \psi_j(v), \quad u,v \in [0,1] \qquad (1)$$

where $\mathbf{d}_{i,j} = (x_{i,j}, y_{i,j}, z_{i,j})$ and $\phi_i(u)$ and $\psi_j(v)$ can be Bernstein–Bézier polynomials, B-spline basis functions, or even rational B-spline basis functions[1,2]. It is assumed that $\mathbf{s}$ is C^0 continuous.

Department of Computer Science, University of California, Davis, CA 95616-8562, USA
* NSF Engineering Research Center for Computational Field Simulation, Mississippi State University, PO Box 6176, Mississippi State, MS 39762, USA
Paper received: 28 February 1995. Revised: 30 April 1995

The 2D closed trimming curves in parameter space are denoted by

$$\mathbf{c}_k(t) = (u(t), v(t)) = \sum_{i=0}^{n_k} \mathbf{d}_i^k \, \phi_i(t),$$

$$t \in [0,1], \; k = 0, \ldots, K \tag{2}$$

where $\mathbf{d}_i^k = (u_i^k, v_i^k)$ is one of the $(n_k + 1)$ control points of trimming curve $\mathbf{c}_k$, and $\mathbf{c}_k(0) = \mathbf{c}_k(1)$. It is assumed that the rotation number of all trimming curves is $+1$, which implies that they have the same orientation. Each trimming curve must be at least C^0 continuous but can contain tangent discontinuities. A trimming curve must not intersect another trimming curve and must not self-intersect. In most practical applications, there is one trimming curve enclosing the region that contains all the other trimming curves, which is assumed to be $\mathbf{c}_0$. If this enclosing trimming curve is not explicitly defined, the boundary of the parameter space is chosen to be $\mathbf{c}_0$ (i.e. $\mathbf{c}_0$ is the piecewise linear curve given by the four line segments $v = 0$, $u = 1$, $v = 1$, and $u = 0$). *Figure 1* shows the trimming curves of a trimmed surface in physical and parameter space.

The approach described in this paper is similar to the construction of planar Voronoï and power diagrams in the sense that a tessellation of the valid part of the parameter space of a trimmed surface is computed. These diagrams are described in detail in References 3–6. This paper presents a new method for the construction of the curved boundaries (bisecting curves) defining the tiles associated with the trimming curves. A computationally efficient technique is used for generating a finite set of points on each bisecting curve. The technique is based on shortest distance computations for a finite set of points on a rectilinear grid in parameter space.

Much work has been done regarding the generalization of Voronoï diagrams. Most of the generalizations in 2D deal with the construction of Voronoï diagrams for sets of points, line segments, polygons, circles, and more general planar curves. The construction of Voronoï diagrams for such sets is discussed in References 7–14. Some of these papers address the relation between Voronoï diagrams and the medial axis transform (MAT). The efficient construction of a generalized Voronoï diagram for a set of trimming curves given in parameter space is an integral part of the method described in this paper.

An algorithm for rendering trimmed surfaces by using quadrilateral and triangular elements is described in Reference 15. In Reference 16, a 2D mesh generation algorithm is described that automatically discretizes 2D regions containing trimming curves based on the identification of certain geometrical features of the trimming curves, e.g. slope discontinuities. A technique utilizing a combined 'triangulation–quadrangulation' strategy of the valid part of the parameter space of a trimmed surface is presented in Reference 17. A method for representing a trimmed NURBS (non-uniform rational B-spline) surface by a set of Bézier patches is discussed in Reference 18 the problem of data exchange is addressed; trimmed rational surfaces are approximated by non-rational surfaces.

Recently, an elegant algorithm for the polygonal approximation of trimming curves and the approximation of the valid part of a trimmed surface by a set of triangles has been introduced in Reference 19. The algorithm determines the set of triangles — defining a piecewise linear surface approximation — based on a user-specified tolerance ϵ, which ensures that the approximation, in physical space, does not deviate by more than ϵ from a given trimmed NURBS surface.

The algorithm discussed in Reference 19 must consider errors introduced by the piecewise linear approximation of a given trimmed surface, while the algorithm presented in this paper exactly represents the valid part of a trimmed surface as a union of so-called parameter surfaces, i.e. analytical, planar NURBS surfaces $(u(\xi, \eta), v(\xi, \eta))$. These parameter surfaces can be viewed as an additional 'layer' used for the indirect representation of the valid part of a trimmed NURBS surface. Both algorithms utilize a scan line algorithm for the generation of data inside the valid part of a trimmed surface and the generation of the boundary curves of parameter surfaces, respectively.

The underlying concepts of parametric curve and surface design used in this paper are covered in References 1, 2 and 20. Various solutions to the SSI problem are described in Reference 21. A survey of SSI algorithms is provided in Reference 22. Recent developments in surface and volume grid generation are surveyed in References 23 and 24, which also address the problems that arise when having to generate grids for trimmed surfaces.

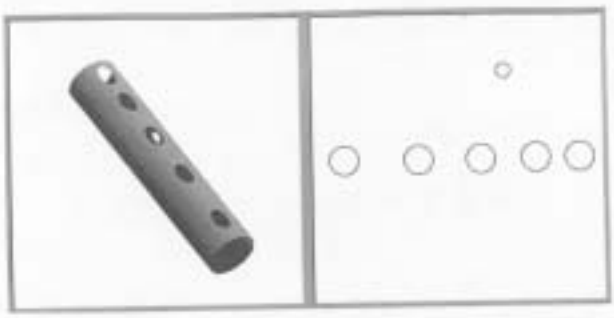

Figure 1 Trimming curves in physical (left) and parameter (right) space

PROBLEM STATEMENT AND DEFINITIONS

The trimming curves define a simply connected region in parameter space. The goal is to represent this region by a set of planar, four-sided surfaces whose union is the valid part of the parameter space. In the following, such a surface will be referred to as a *parameter surface*. It is denoted by

$$\mathbf{u}_l(\xi, \eta) = (u_l(\xi, \eta), v_l(\xi, \eta)) = \sum_{i=0}^{m_l} \sum_{j=0}^{n_l} \mathbf{d}_{i,j}^l \, \overline{\phi}_i(\xi) \overline{\psi}_j(\eta), \quad \xi, \eta \in [0,1] \tag{3}$$

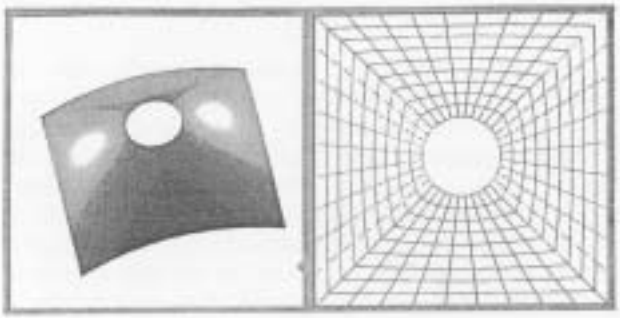

Figure 2 Concept of parameter surfaces

where $\mathbf{d}^l_{i,j} = (u^l_{i,j}, v^l_{i,j})$ Thus, the part of a surface $\mathbf{s}$ that is implied by the parameter surface $\mathbf{u}_l$ is given by

$$\mathbf{s}(u_l) = \mathbf{s}(u_l(\xi,\eta), v_l(\xi,\eta)) = \sum_{i=0}^{m}\sum_{j=0}^{n} \mathbf{d}_{i,j}\phi_i(u_l(\xi,\eta))\phi_j(v_l(\xi,\eta)), \quad \xi,\eta \in [0,1] \tag{4}$$

Figure 2 shows a trimmed surface with one hole represented by four parameter surfaces that are shaded in different grey scales.

The main problem to be solved is the generation of the boundary curves of the parameter surfaces. This problem can be solved using a generalization of the *Voronoï diagram* of a point set. When dealing with trimmed surfaces, the trimming curves define the set for which a tessellation, a generalized Voronoï diagram must be computed. The tile boundaries in a planar Voronoï diagram implied by a point set are obtained from the perpendicular bisectors of all possible point pairs[4,6]. Generalizations of this 'standard' Voronoï diagram are obtained when the elements for which a tessellation is to be constructed are points, line segments, circles, polygons, and more general curves. *Figure 3* shows Voronoï diagrams for a set of points and a set of circles[3,5].

Voronoï diagrams introduce *tiles* around each element (points, line segments, circles etc.) according to some distance measure. A tile is defined as the region that contains all the points being closer to a particular element than any other element. The tile boundary can be used to subdivide a tile into a set of four-sided planar surfaces whose union represents the area between the element and the element's tile boundary.

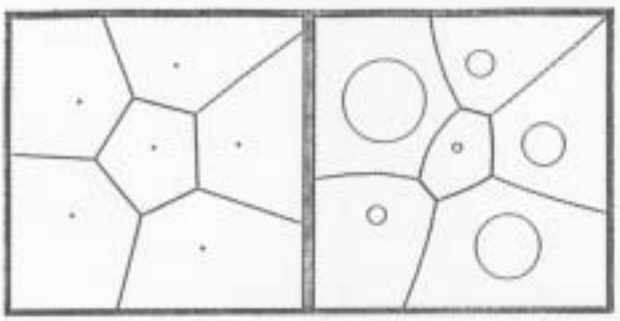

Figure 3 Voronoï diagrams for set of points and set of circles

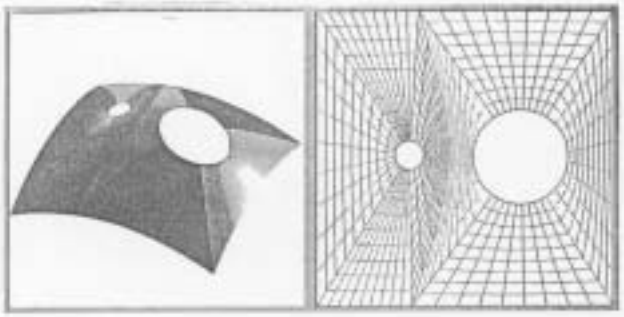

Figure 4 Four-sided parameter surfaces in tiles around several trimming curves

Each four-sided surface can be constructed by subdividing the tile boundary curve into segments and generating additional curves connecting end points of the tile boundary segments and points on the element. This principle is shown to *Figure 4*.

The following definitions are needed when using the Voronoï diagram for the representation of trimmed parametric surfaces.

Definition 1

Given a set of planar, closed, pairwise non-intersecting curves with rotation number $+1$ such that no curve lies in the interior of any other curve, the locus of points that have a smaller shortest (Euclidean) distance of curve $\mathbf{c}_k$ than to any other curve is called the *tile* T of $\mathbf{c}_k$, denoted by $T(\mathbf{c}_k)$.

If all curves $\mathbf{c}_k$ have a continuous tangent, the smallest shortest distance is equal to the *perpendicular distance*. Assuming that there are K such curves, $T(\mathbf{c}_k)$ is obtained as the intersection of $K-1$ half-spaces, i.e.

$$T(\mathbf{c}_k) = \bigcap_{\substack{l=1 \\ l \neq k}}^{K} H(\mathbf{c}_k, \mathbf{c}_l), \tag{5}$$

where $\mathbf{H}(\mathbf{c}_k, \mathbf{c}_L)$ is the half-space containing all points that have a smaller shortest distance to $\mathbf{c}_k$ than to $\mathbf{c}^6_l$.

Definition 2

Given $\mathbf{K}$ curves as in Definition 1, the K tiles $T(\mathbf{c}_k)$ define the *generalized Voronoï diagram*. The curve separating the two half-spaces $H(\mathbf{c}_k, \mathbf{c}_l)$ and $H(\mathbf{c}_l, \mathbf{c}_k)$ is called the *bisector* of $\mathbf{c}_k$ and $\mathbf{c}_l$. The intersection points of bisectors are called *Voronoï vertices*.

Figure 5 shows the generalized Voronoï diagram for five arbitrary curves in the plane.

Once the generalized Voronoï diagram (referred to as Voronoï diagram in the following) has been computed for a set of trimming curves all tiles must be represented by sets of four-sided parameter surfaces. The boundary curves of the parameter surfaces are line segments, segments of the bisectors, and segments of the trimming curves. It is also possible to use segments of the curves defining the *medial axis* (or *skeleton*) of

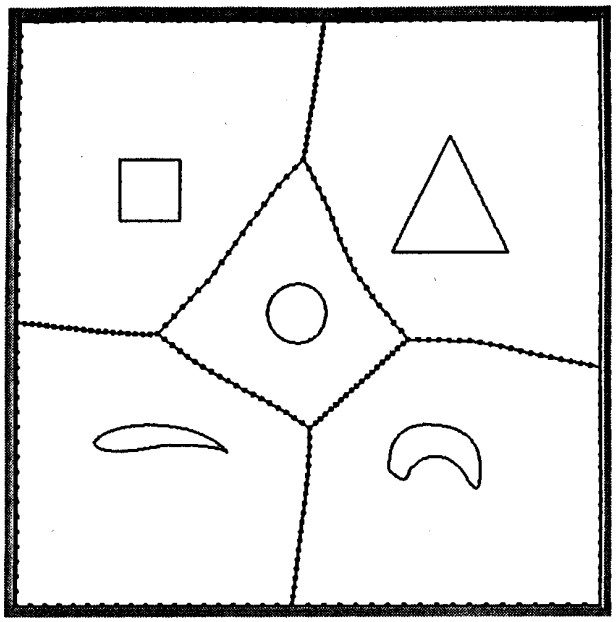

Figure 5 Generalized Voronoï diagram for five curves

the tiles as boundary curves of the parameter surfaces[25,26]. It is pointed out later why this can cause problems for the general case. The MAT has been used in computer vision and pattern recognition in the context of *thinning algorithm*[25-28].

Definition 3

Let R be a simply connected 2D region and B be its boundary. A point $\mathbf{p}$ in R's interior belongs to the *medial axis* or *skeleton* of R if there are (at least) two points on the boundary B having the same distance from $\mathbf{p}$ and being boundary points with shortest distance from $\mathbf{p}$.

The medial axis of a tile can be used for the construction of parameter surfaces. Certain segments of the medial axis can be used to define the boundary curves of parameter surfaces that start on a bisector and end on a trimming curve. Once the boundary curves of each parameter surface are generated its control points $\mathbf{d}^l_{i,j}$ (see Equation 3) will be obtained by interpolating its four boundary curves. The chosen interpolation method must ensure that the parameter surface lies completely inside the region enclosed by its four boundary curves. *Figure 6* shows the Voronoï diagram for a set of arbitrary trimming curves and the medial axes inside each tile.

COMPUTING THE VORONOI DIAGRAM FOR A SET OF TRIMMING CURVES

An efficient algorithm is needed for the generation of the tile boundaries around each trimming curve in parameter space. First, tiles are constructed around the trimming curves $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$,..., and $\mathbf{c}_K$. The final tiles are obtained by intersecting the tile boundary curves in the Voronoï diagram for $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$,..., and $\mathbf{c}_K$ with the enclosing trimming curve $\mathbf{c}_0$. This general configuration of closed trimming curves is characteristic for all real-world cases of practical relevance.

Initially, the trimming curve $\mathbf{c}_0$ is not considered in the construction of the Voronoï diagram. The generation of the Voronoï diagram is based on the computation of the intersections of bisectors with the edges in a triangulation of the parameter space $[0,1] \times [0,1]$. The vertices in this triangulation are labelled according to the index of the closest trimming curve. The labels are used to determine whether there is an intersection

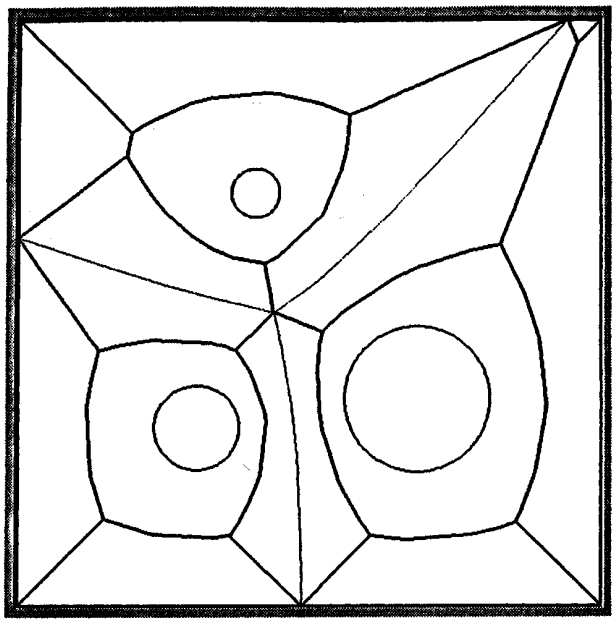

Figure 6 Voronoï diagram (light) for set of trimming curves and medial axes (dark) in each tile

between bisectors and edges in the triangulation. The intersections between the edges and the bisectors are computed and properly connected, thus defining an initial approximation of the Voronoï diagram.

Multiple bisectors can intersect the same edge in the triangulation, and multiple bisectors can intersect in the interior of a triangle. These cases are addressed by the algorithm described below. The algorithm does *not* consider the case of one bisector intersecting the same edge multiple times. One does not need to consider this case due to the fact that only a discrete point (line segment) approximation of the bisectors of the Voronoï diagram is to be computed. The approximation of the Voronoï diagram requires these steps:

(i) Construction of a triangulation of the parameter space $[0,1] \times [0,1]$,
(ii) Extraction of all triangles whose three vertices all lie in the valid part of the parameter space,
(iii) Labelling each vertex in the triangulation with the index of the closest trimming curve,
(iv) Computation of intersections between bisectors and edges in the triangulation using a recursive subdivision strategy,
(v) Computation of intersections of bisectors,
(vi) Computation of intersection between bisectors and curve $\mathbf{c}_0$,
(vii) Generation of piecewise linear and cubic B-spline approximations of all tile boundaries in the Voronoï diagram.

These steps are now described in greater detail. Denoting the minimal distance of all possible pairs of trimming curves by $d_{\min}$, the initial triangulation of the parameter space only contains edges that are shorter than $d_{\min}/2$. This is accomplished by subdividing the parameter square $[0,1] \times [0,1]$ into squares whose diagonal is shorter than $d_{\min}/2$ and splitting each square into two triangles. Only triangles whose three vertices all lie in the valid part of the parameter space are considered for the following computations.

Each vertex in the triangulation is labelled according to the closest trimming curve. The square of the distance d between a vertex with coordinate vector (x, y) and a trimming curve $\mathbf{c}_k(t)$ is given by $d^2(t) = (x - u_k(t))^2 + (y - v_k(t))^2$, $t \in [0,1]$. The critical points of $d^2(t)$ are identified, and the associated distances are computed. In addition, one computes the distances to those points on $\mathbf{c}_k$ where slope discontinuities occur. The index of the trimming curve that has minimal distance to (x, y) is used as the label for this vertex. It turns out that the case of multiple trimming curves all having minimal distance to (x, y) does not require a special case treatment.

The labels at each vertex in the triangulation are used to identify edges which are intersected by at least one bisector. It is possible that the three labels at a triangle's vertices are the same, that there are two different labels, or that they are all different. In the first case, it is assumed that no bisector intersects the triangle. In the second case, it is assumed that there are bisectors intersecting the two edges whose end points have different labels. In the third case, it is assumed that there are bisectors intersecting all three edges.

Points lying on bisectors in the Voronoï diagram are computed based on Algorithm 1 for the recursive subdivision of the parameter space triangulation:

Algorithm 1 (*Computation of points on bisectors in Voronoï diagram*)

Input:
- trimming curves $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \ldots, \mathbf{c}_K$,
- triangulation of parameter space $[0,1] \times [0,1]$,
- label $I \in \{1, 2, 3, \ldots, K\}$ at each vertex in triangulation referring to closest trimming curve $\mathbf{c}_I$,
- tolerance ϵ;

Output:
- set of points on bisectors in Voronoï diagram;

FOR all triangles in the parameter space triangulation DO

IF there are at least two different labels among I_1, I_2 and I_3 associated with the triangle's vertices $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$

/* at least one bisector intersects this triangle; compute points on bisector(s) */

{
- compute the midpoints $\mathbf{m}_{1,2}$, $\mathbf{m}_{2,3}$, and $\mathbf{m}_{3,1}$ of the three edges $e_{1,2} = \overline{\mathbf{v}_1\mathbf{v}_2}$, $e_{2,3} = \overline{\mathbf{v}_2\mathbf{v}_3}$, and $e_{3,1} = \overline{\mathbf{v}_3\mathbf{v}_1}$;

FOR the two (or three) edges among $e_{1,2}$, $e_{2,3}$, and $e_{3,1}$ whose end points have different labels DO

/* determine whether these edges are intersected by one or more bisectors */

{
- compute, for each of these two (or three) edges, the points $\mathbf{p}_{i,j}$ on $e_{i,j}$ having the same distance to the trimming curves indicated by the labels;

IF there is no trimming curve that is closer to $\mathbf{p}_{i,j}$ than either of the two trimming curves indicated by the labels

/* $\mathbf{p}_{i,j}$ is a point on a bisector */

- store $\mathbf{p}_{i,j}$ as a point on the bisector of the two trimming curves indicated by the different labels;

ELSE

/* $\mathbf{p}_{i,j}$ is not a point on a bisector; many bisectors intersect this edge */

- replace the value of $\mathbf{m}_{i,j}$ by the value of $\mathbf{p}_{i,j}$;

}

IF one has found at least one edge whose midpoint is closer to a trimming curve that is different from either of the two trimming curves indicated by its labels

{ /* subdivision of triangle necessary */

```
    • split the triangle into the four subtriangles
      given by the point triples (v1, m1,2, m3,1), (v2,
      m2,3, m1,2), (v3, m3,1, m2,3), and (m1,2, m2,3,
      m3,1);
    • recursively compute intersections of bisectors
      with edges of subtriangles;
      /* a triangle is no longer subdivided when
      each of its edges is intersected by at most
      one bisector or when its longest edge is
      shorter than ε; if a triangle is obtained
      whose longest edge is shorter than ε, then
      the triangle's centroid is viewed as the
      intersection of bisectors */
  }
}
```

Figure 7 shows the results of Algorithm 1 when applied to two different configurations of trimming curves.

A piecewise linear approximation of the Voronoï diagram is obtained by connecting the points resulting from Algorithm 1. If exactly two edges of a triangle each contain one point on a bisector, denoted by $\mathbf{p}_1$ and $\mathbf{p}_2$, then $\mathbf{p}_1$ and $\mathbf{p}_2$ are connected. If all three edges of a triangle each contain one point on a bisector, denoted by $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$, then $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$ are assumed to be lying on three different bisectors. In the second case, each of the three points is connected with the point $\mathbf{q}$ that is the intersection of three bisectors. An iterative method is used to approximate the coordinates of $\mathbf{q}$.

The centroid of $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$, dentoed by $\mathbf{q}^0$, is used as the initial approximation of $\mathbf{q}$. Subsequent approximations $\mathbf{q}^i$ are obtained in the following way: Given an approximation $\mathbf{q}^i$, one computes its closest point $\mathbf{x}_1$ on $\mathbf{c}_1$, closest point $\mathbf{x}_2$ on $\mathbf{c}_2$, and closest point $\mathbf{x}_3$ on $\mathbf{c}_3$, assuming that the curves $\mathbf{c}_1$, $\mathbf{c}_2$, and $\mathbf{c}_3$ are the ones closest to $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$, respectively. One computes the critical points of the squared Euclidean distance between $\mathbf{q}^i$ and the three trimming curves (using Newton's method) and identifies the three closest points $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$. Considering a single trimming curve, there might be multiple points on it having minimal distance to $\mathbf{q}^i$. It is not important which point is chosen. If the centre of the circle passing through $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ lies in the interior (or on one of the edges) of the triangle $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$, the centre is chosen as the value of $\mathbf{q}^{i+1}$. If the centre of this circle lies outside the triangle $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$, the unique point on the triangle's edges having minimal distance to the circle's centre is chosen as the value of $\mathbf{q}^{i+1}$. Due to the fact that there must be an intersection point of three bisectors in the triangle, the method will converge. The method terminates when the Euclidean distance between $\mathbf{q}^i$ and $\mathbf{q}^{i+1}$ is smaller than ϵ.

Whenever Algorithm 1 leads to a triangle whose longest edge is shorter than ϵ (one of the termination criteria of the algorithm) the centroid of such a triangle is considered to be the intersection of bisectors. Eventually, one obtains a piecewise linear approximation of all bisectors in the Voronoï diagram. *Figure 8* shows the piecewise linear approximation of the Voronoï diagram for a set of trimming curves. This example has an enclosing trimming curve $\mathbf{c}_0$. The piecewise linear approximation of the Voronoï diagram is intersected with the enclosing trimming curve $\mathbf{c}_0$, and the resulting curve segments on $\mathbf{c}_0$ are used for the definition of the tile boundary curves around $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$, ..., and $\mathbf{c}_K$.

Based on the piecewise linear approximation of the Voronoï diagram and the curve segments of $\mathbf{c}_0$, a cubic B-spline approximation is constructed for all tile boundary curves. In the following, the tile boundary curve associated with the trimming curve $\mathbf{c}_i$ is denoted by $\bar{\mathbf{c}}_i$. The cubic B-spline representation of $\bar{\mathbf{c}}_i$ is based on a chord length parametrization defined by the lengths of the line segments in the piecewise linear approximation[3]. *Figure 9* shows the cubic B-spline curves approximating the tile boundaries for the example shown in *Figure 8*.

Remark 1. In practical applications, one is often concerned with interior trimming curves that are not closed and intersect the outer trimming curve $\mathbf{c}_0$. If this is the case, the intersections between those trimming curves and $\mathbf{c}_0$ are computed, and the resulting curve segments on $\mathbf{c}_0$ — in combination with the trimming curves that are not closed — define a new enclosing trimming curve $\mathbf{c}_0$. Once this new curve $\mathbf{c}_0$ has been

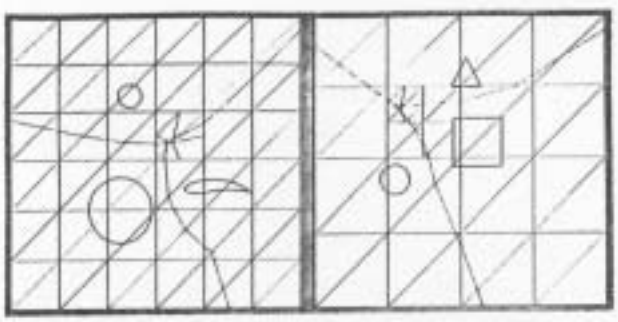

Figure 7 Recursive subdivision of parameter space triangulations for generation of points on bisectors in Voronoï diagram

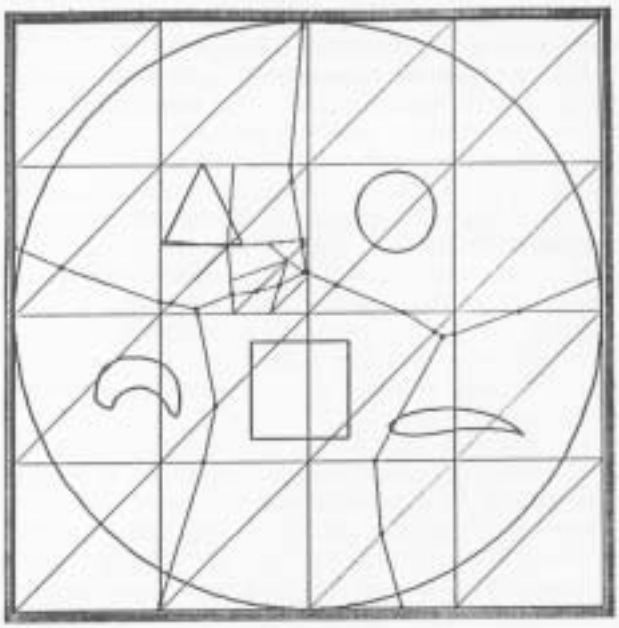

Figure 8 Piecewise linear approximation of Voronoï diagram for circular, non-convex, and slope-discontinuous trimming curves

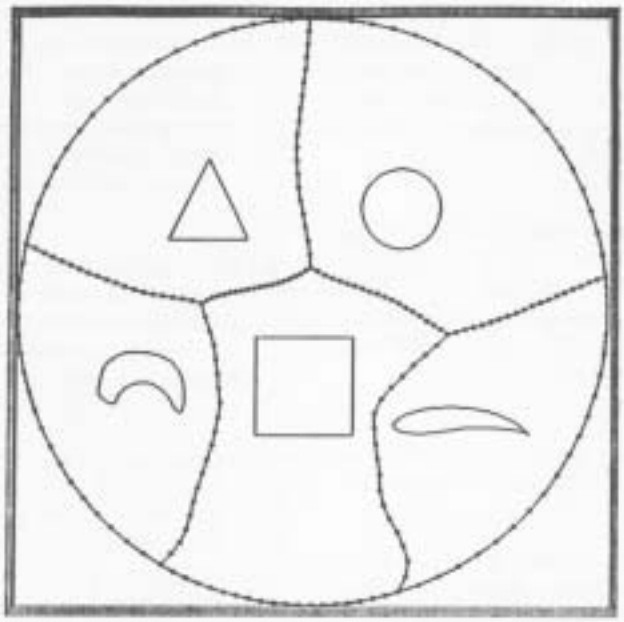

Figure 9 Cubic B-spline approximation of Voronoï diagram for circular, non-convex, and slope-discontinuous trimming curves

computed, the construction of the Voronoï diagram follows the same principle as discussed above. *Figure 10* shows a configuration in which the trimming curve $\mathbf{c}_0$ is intersected by four trimming curves. Trimmed periodic surfaces (i.e. surfaces for which $\mathbf{s}(0, v) = (\mathbf{s}(1, v)$ and/or $\mathbf{s}(u, 0) = \mathbf{s}(u, 1)$) do not require an additional case distinction.

Once the Voronoï diagram is known, each tile can independently be decomposed into parameter surfaces. Thus, all tiles can be processed in parallel, and only two curves ($\mathbf{c}_i$ and $\bar{\mathbf{c}}_i$) have to be considered for the construction of the parameter surfaces inside a particular tile.

COMPUTING THE BOUNDARY CURVES FOR PARAMETER SURFACES

The next step is the decomposition of the tiles in the Voronoï diagram (bounded by the outer trimming curve $\mathbf{c}_0$) into parameter surfaces. The union of all parameter surfaces defines the valid part of the trimmed surface. The computation of an approximation of the medial axis of a tile can be extremely expensive. Furthermore, even if one knows the medial axis of a tile, it is not quite clear how to extract segments from it to be used for the construction of the boundary curves of parameter surfaces and how to interpolate the boundary curves.

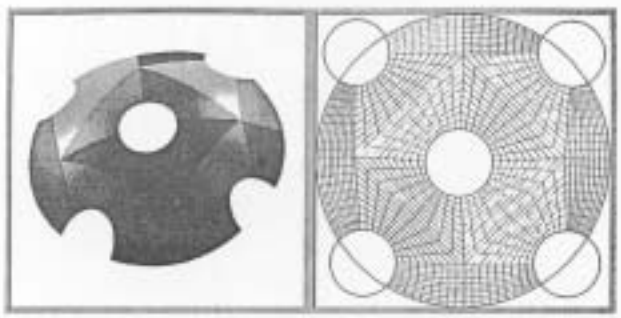

Figure 10 Trimming curves intersecting enclosing trimming curve

Therefore, the algorithm used for the construction of the parameter surfaces is based on decomposing a tile into a set of surfaces that have two horizontal boundary curves. The algorithm is similar to the *scan line* (or *scan conversion*) algorithm used for filling the interior of 2D polygons[29] and to the construction of the *Reeb graph* representing the topology 'skeleton' of manifolds with holes[30].

The basic idea for the construction of the parameter surfaces inside the tile associated with a particular trimming curve $\mathbf{c}$ is as follows: The valid part of the parameter space, i.e. the region between the trimming curve $\mathbf{c}$ and the tile boundary curve $\bar{\mathbf{c}}$, is represented by as set of *ruled parameter surfaces*. They are obtained by identifying local extrema in v-direction (and horizontal line segments) on $\mathbf{c}$ and $\bar{\mathbf{c}}$, computing the intersections between horizontal lines passing through these local extrema (and the horizontal line segments) and $\mathbf{c}$ and $\bar{\mathbf{c}}$, and constructing ruled parameter surfaces from the resulting horizontal line segments inside the tile and non-horizontal curve segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$.

Assuming that the trimming curves are C^0 continuous, the generation of the boundary curves of the parameter surfaces inside the tile associated with trimming curve $\mathbf{c}$ (having tile boundary curve $\bar{\mathbf{c}}$) requires these steps:

- Determining extrema in v-direction and end points of horizontal line segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$:

(i) Computing all points $\mathbf{p} = (u, v)$ on $\mathbf{c}$ and $\bar{\mathbf{c}}$ representing local extrema in v-direction.
(ii) Computing all end points $\mathbf{q} = (u_1, v)$ and $\mathbf{r} = (u_2, v)$, $u_1 < u_2$, of all horizontal line segments $\overline{\mathbf{qr}}$ on $\mathbf{c}$ and $\bar{\mathbf{c}}$.

- Computing intersection points between horizontal semi-infinite lines (passing through the local extrema on $\mathbf{c}$ and $\bar{\mathbf{c}}$) and $\mathbf{c}$ and $\bar{\mathbf{c}}$; computing intersection points between horizontal semi-infinite lines (defined by the horizontal line segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$) and $\mathbf{c}$ and $\bar{\mathbf{c}}$;

(iii) Computing all intersection points between the two semi-infinite lines $\mathbf{L}(t) = p + (-t - 0)$ and $\mathbf{R}(t) = \mathbf{p} + (t, 0)$, $t \in [0, \infty]$, and $\mathbf{c}$ and $\bar{\mathbf{c}}$.

(iv) Computing all intersection points between the two semi-infinite lines $\mathbf{L}(t) = q + (-t, 0)$ and $\mathbf{R}(t) = \mathbf{r} + (t, 0)$, $t \in [0, \infty]$, and $\mathbf{c}$ and $\bar{\mathbf{c}}$.

- Selecting those intersection points to be used as corner vertices of the ruled parameter surfaces:

(v) Determining the intersection point $\mathbf{x}$ on $\mathbf{L}(t)$ closest to $\mathbf{p}$ and the intersection point $\mathbf{y}$ on $\mathbf{R}(t)$ closest to $\mathbf{p}$ ($\mathbf{x}$ and $\mathbf{y}$ being different from all points $\mathbf{p}$, $\mathbf{q}$ and $\mathbf{R}$), provided that both $\mathbf{L}(t)$ and $\mathbf{R}(t)$ start at a local extremum; discarding all other intersection points $\mathbf{x}$ on $\mathbf{L}(t)$ and $\mathbf{y}$ on $\mathbf{R}(t)$.

(vi) Determining the intersection point $\mathbf{x}$ on $\mathbf{L}(t)$ closest to $\mathbf{q}$ and the intersection point $\mathbf{y}$ on $\mathbf{R}(t)$ closest to $\mathbf{r}$ ($\mathbf{x}$ and $\mathbf{y}$ being different from all points $\mathbf{p}$, $\mathbf{q}$, and $\mathbf{r}$), provided that $\mathbf{L}(t)$ starts at the left and $\mathbf{R}(t)$ starts at the right end point of a horizontal line segment; discarding all other intersection points $\mathbf{x}$ on $\mathbf{L}(t)$ and $\mathbf{y}$ on $\mathbf{R}(t)$.

(vii) Determining those points $\mathbf{x}(\mathbf{y})$ for which the horizontal line segments $\overline{\mathbf{xp}}$ and $\overline{\mathbf{xq}}$ ($\overline{\mathbf{py}}$ and $\overline{\mathbf{ry}}$, respectively) lie in the interior (including the boundary curves) of the tile associated with $\mathbf{c}$; discarding all other intersection points $\mathbf{x}$ and $\mathbf{y}$.

- Constructing a graph (vertices and edges) for the set of local extrema, end points of horizontal line segments, and the selected intersection points, assuming there are L scan lines and N_k points (considering local extrema, end points of horizontal line segments, and selected intersection points) on the kth scan line:

(viii) Ordering the points $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$, $\mathbf{x}$, and $\mathbf{y}$ resulting from Steps (i)–(vii) according to their u- and v-coordinates (i.e. ordering them according to the scan lines they lie on and from left to right on each scan line) by constructing the ordered vertex set $\{\mathbf{v}_{j,k} = (u_{j,k}, v_{j,k}) | k = 1, \ldots, L, j = 1, \ldots, N_k\}$, where the point $\mathbf{v}_{j,k}$ precedes the point $\mathbf{v}_{r,s}$ provided that $\mathbf{v}_{j,k}$'s v-coordinate is smaller than $\mathbf{v}_{r,s}$'s v-coordinate or provided that $\mathbf{v}_{j,k}$'s v-coordinate is equal to $\mathbf{v}_{r,s}$'s v-coordinate but $\mathbf{v}_{j,k}$'s u-coordinate is smaller than $\mathbf{v}_{r,s}$'s u-coordinate, i.e. $\mathbf{v}_{j,k} < \mathbf{v}_{r,s} : \Leftrightarrow (v_{j,k} < v_{r,s}) \vee (v_{j,k} = v_{r,s} \wedge u_{j,k} < u_{r,s})$.

(ix) Constructing a graph G with vertices $\mathbf{v}_{j,k}$ and horizontal edges $e_{j,j+1,k} = \overline{\mathbf{v}_{j,k}\mathbf{v}_{j+1,k}}$, $k = 1, \ldots, L$, $j = 1, \ldots, N_k - 1$, where only those edges $e_{j,j+1,k}$ are stored that lie in the interior of the tile associated with $\mathbf{c}$ (the boundary curves are considered part of the interior).

(x) Ordering the points $\mathbf{v}_{j,k}$ on the curve $\mathbf{c} = \mathbf{c}(t)$ (and on the curve $\bar{\mathbf{c}} = \bar{\mathbf{c}}(\bar{t})$) with respect to increasing parameter value t ($\bar{t}$, respectively) and — according to this order — adding an edge $e_{j,k,r,s}$ to G for each pair of successive points $\mathbf{v}_{j,k}$ and $\mathbf{v}_{r,s}$ on $\mathbf{c}$ (and on $\bar{\mathbf{c}}$), where $\mathbf{v}_{j,k}$ and $\mathbf{v}_{r,s}$ must not be end points of a horizontal line segment on $\mathbf{c}$ (or $\bar{\mathbf{c}}$).

- Ordering the horizontal edges $e_{j,k,r,s}$ of G from left to right on each scan line and generating ruled parameter surfaces by applying linear interpolation to pairs of associated non-horizontal curve segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$:

(xi) Considering all edges $e_{j,k,r,s}$, $s > k$, $k = 1, \ldots, L - 1$, with end points $\mathbf{v}_{j,k}$ all lying on one scan line and end points $\mathbf{v}_{r,s}$ all lying on another scan line, ordering these edges in u-direction $(e_{j,k,r,s} < e_{J,k,R,s} :\Leftrightarrow (u_{j,k} \le u_{J,k} \wedge u_{r,s} < u_{R,s}) \vee u_{j,k} < u_{J,k} \wedge u_{r,s} \le u_{R,s}))$.

(xii) Considering the set of edges in $\{e_{j_0,k,r_0,s} < e_{j_2,k,r_2,s} < e_{j_4,k,r_4,s} < \ldots < e_{j_E,k,r_E,s}\}$, E being the even number of non-horizontal boundary curves of ruled parameter surfaces having one end point on the kth scan line and the other end point on the sth scan line, generating and storing the B-spline control information for each ruled parameter surface defined by the two horizontal line segments $\overline{\mathbf{V}_{j_{2l-1},k}\mathbf{V}_{j_{2l},k}}$ and $\overline{\mathbf{V}_{r_{2l-1},s}\mathbf{V}_{r_{2l},s}}$ and the left and right boundary curves, which are the exactly extracted curve segments between $\mathbf{V}_{j_{2l-1},k}$ and $\mathbf{V}_{r_{2l-1},s}$ and between $\mathbf{V}_{j_{2l},k}$ and $\mathbf{v}_{r_{2l},s}$, $l = 1, \ldots, E/2$.

The local extrema in the v-direction on $\mathbf{c}$ and $\bar{\mathbf{c}}$, computed in Step (i), are not always characterized by a horizontal curve tangent. They can coincide with points where tangent discontinuities occur. Step (ii) actually considers all 'nearly horizontal' line segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$, i.e. line segments whose absolute slope is smaller than some tolerance ϵ. *Figure 11* shows the local extrema in v-direction (solid points) and the end points of horizontal line segments (circles) on two simple curves $\mathbf{c}$ and $\bar{\mathbf{c}}$ defining a Voronoï tile.

When computing the intersections between semi-infinite horizontal lines and $\mathbf{c}$ and $\bar{\mathbf{c}}$ (Steps (iii) and (iv)),

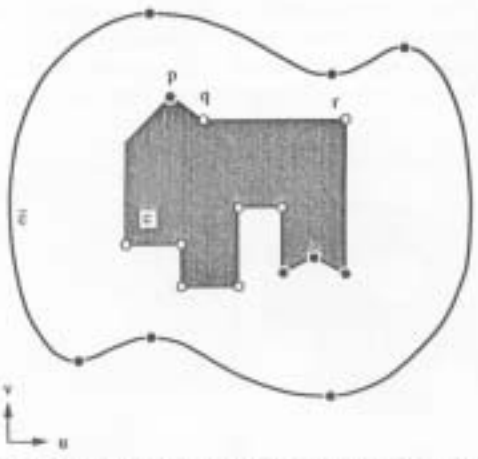

Figure 11 Local extrema in v-direction and end points of horizontal line segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$

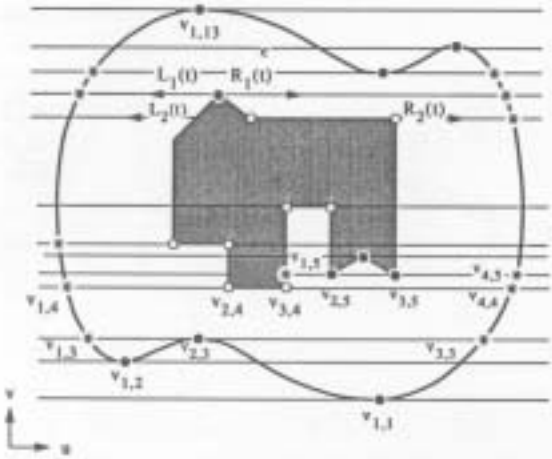

Figure 12 Selected intersection points between semi-infinite horizontal lines and $\mathbf{c}$ and $\bar{\mathbf{c}}$

one must consider the existence of 'nearly horizontal' curve segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$ that lie inside an ϵ-strip to both sides of one of the semi-infinite horizontal lines. If such 'nearly horizontal' curve segments are found, they are not considered for the computation of intersections. The intersection points determined in Steps (v)–(vii) completely define all ruled parameter surfaces. These points lie on the boundary curves of ruled parameter surfaces. *Figure 12* shows the intersection points (stars) remaining after Step (vii) (same example as shown in *Figure 11*).

Steps (vii)–(x) define a set of (non-intersecting) edges/segments connecting local extrema, end points of horizontal curve segments, and selected intersection points. These edges/segments are horizontal line segments inside the tile, horizontal line segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$, and non-horizontal curve segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$. By ordering the non-horizontal edges $e_{j,k,r,s}$ in u-direction (Step (xi)) one generates the topological information necessary to define the four boundary curves of each ruled parameter surface. The four boundary curves of a ruled parameter surface are two horizontal line segments, given by successive horizontal line segments on two different horizontal lines and two non-horizontal curve segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$. *Figure 13* indicates the single horizontal line segments (solid lines) and non-horizontal curve segments (dashed curves) for the example shown in *Figure 11*.

The final boundary curves for each parameter surface are assembled in Step (xii). Since the two non-horizontal boundary curves of each ruled parameter surface are curve segments on $\mathbf{c}$ and $\bar{\mathbf{c}}$, one must represent these curve segments as single curves, e.g. as NURBS curves. If one chooses to use a NURBS representation, it is necessary to represent the two non-horizontal boundary curves using the same degree and the same knots. This is achieved by elevating the degree of the lower-degree boundary curve to the degree of the higher-degree boundary curve and by 'merging' the knots of the two curves[31,32]. Eventually, the ruled parameter surface $\mathbf{u}_j(\xi,\eta)$ is given by

$$\begin{aligned}\mathbf{u}_j(\xi,\eta) &= (u_j(\xi,\eta), v_j(\xi,\eta)) \\ &= (1-\xi)e_{j,k,r,s}(\eta) + \xi\, e_{j,k,R,s}(\eta) \\ &= (1-\xi)\sum_{i=0}^{n_j} \mathbf{d}_i^j \bar{\psi}_i(\eta) + \xi \sum_{i=0}^{n_j} \bar{\mathbf{d}}_i^j \bar{\psi}_i(\eta), \\ &\xi,\eta \in [0,1]\end{aligned} \tag{6}$$

where $\mathbf{d}_i^j = (u_i^j, v_i^j)$, $\bar{\mathbf{d}}_i^j = (\bar{u}_i^j, \bar{v}_i^j)$ and $e_{j,k,r,s}(\eta)$ and $e_{j,k,R,s}(\eta)$ are the analytical representations (e.g. the NURBS representations) of its two non-horizontal boundary curves (see Equation 3). The curves $e_{j,k,r,s}(\eta)$ and $e_{j,k,R,s}(\eta)$ have the same degree and knots.

Figure 14 illustrates some of the curvilinear grids obtained when evaluating parameter surfaces $\mathbf{u}_j$ uniformly (same example as shown in *Figure 11*).

Remark 2. There are two main reasons why ruled surfaces are being used for the definition of the parameter surfaces $\mathbf{u}_j(\xi,\eta)$. First, the scan line algorithm used to define the upper and lower boundary curves of each parameter surface leads, in a natural way, to a ruled surface defined by the left and right boundary curves. Higher-order interpolation schemes would only lead to redundant control information. Second, the construction of the left and right boundary curves of each parameter surface leads to curves that do not contain any local extrema in v-direction nor any horizontal line segments; therefore, performing linear interpolation in u-direction leads to a parameter surface that is a one-to-one mapping from (ξ,η)- to (u,v)-space. This

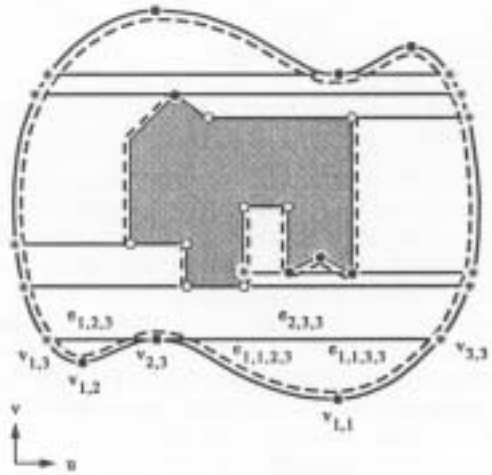

Figure 13 Horizontal line segments and non-horizontal curve segments defining boundary curves of ruled parameter surfaces

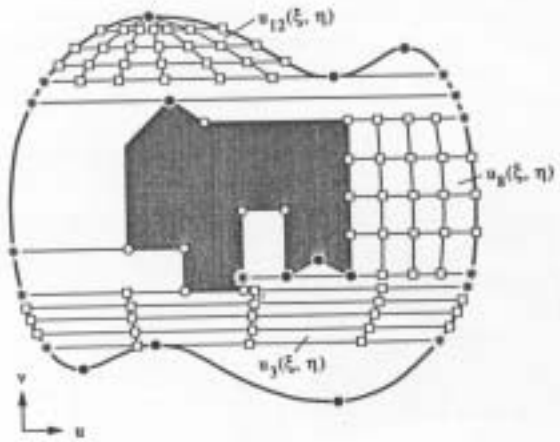

Figure 14 Curvilinear grids of parameter surfaces resulting from uniform evaluation

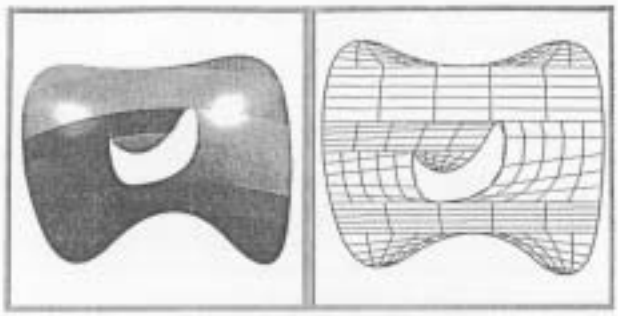

Figure 15 Trimmed surface with one trimming curve

property is important in order to more easily generate grids that do not contain intersecting grid lines[33].

Remark 3. The union of all parameter surfaces inside a particular tile locally represents the valid part of the parameter space without error. This is due to the fact that the trimming curves are given as NURBS curves—defined in parameter space—and that the left and right boundary curves of each parameter surface are obtained by extracting segments from the given trimming curves. Using knot insertion and degree elevation algorithms allows the exact representation of these segments. This, in turn, implies that the valid part of the parameter space is represented exactly.

Remark 4. Figure 14 clearly illustrates that a uniform evaluation of the parameter surfaces in their respective parameter spaces ((ξ, η)-spaces) does not lead to a uniform grid point distribution in (u, v)-space throughout a tile nor to grids whose constituting grid lines match in a continuous fashion along the shared boundary curves of two parameter surfaces. *This is not the goal of the described method.* The method presented here is purely used to define the valid part of a trimmed surface in an analytical, indirect manner. The connectivity information of all parameter surfaces and so-called point distribution functions are used in a later grid generation step to obtain a certain point distribution and continuous grid lines[33–35]. Grid generation methods have to take care of those parameter surfaces having edges that degenerate to points.

EXAMPLES

The described method has been tested for various cases, including various real-world configurations. The method is robust and is completely automatic. *Figures 15–18* show four examples of trimmed surfaces (left: shaded version of trimmed surfaces; right: parameter

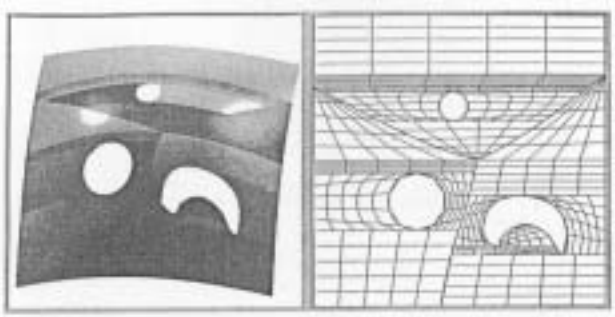

Figure 16 Trimmed surface with multiple trimming curves

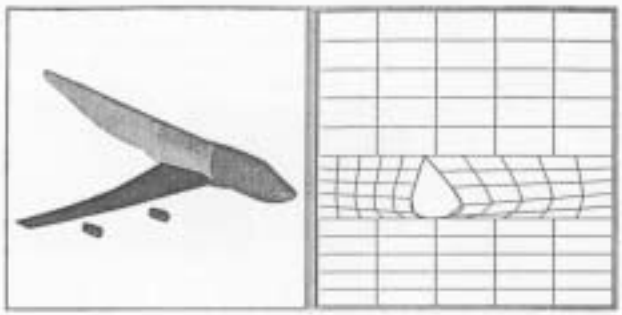

Figure 17 Trimmed surface (fuselage) with one trimming curve

space configuration). The trimming curves, the tile boundary curves of the generalized Voronoï diagrams, and the resulting curvilinear grids in parameter space are shown. The parameter spaces shown in *Figures 17* and *18* (on the right side in each figure) correspond to the fuselage, and the trimming curves indicate the fuselage-wing intersection(s).

Figures 15 and *16* show the same cubic B-spline defined by 4×4 control points. The two trimming curves in *Figure 15*, one of them being an outer enclosing trimming curve, are two cubic B-spline curves defined by 11 (inner trimming curve) and 9 control points (outer trimming curve). The three trimming curves in *Figure 16* are cubic B-spline curves defined by 9, 9, and 11 control points. *Figures 17* and *18* show the same cubic B-spline surface defined by 133×445 control points (fuselage). The one trimming curve in *Figure 17* is a cubic B-spline curve defined by 631 control points. The three trimming curves in *Figure 18* are cubic B-spline curves defined by 631, 9, and 9 control points.

Remark 5. The method has been implemented in C on an SGI Indigo2 Extreme graphics workstation. The examples shown in *Figures 15–18* require approximately 3 s, 3.5 s, 1 s, and 5 s, respectively, for the generation of the control information of all parameter surfaces. Due to the fact that the example shown in *Figure 15* contains an outer enclosing trimming curve, the times required for the examples shown in *Figures 15* and *16* are nearly the same. For grid generation purposes, the additional layer provided by the parameter surfaces is stored and used later for the generation of grids for the valid part of a trimmed surface. In general, this tessellation method is not meant to be used for interactive applications but meant to be used purely for the conversion of given trimmed parametric surfaces to a set of surfaces without trimming curves by means of the additional parameter surface layer.

Remark 6. For each ruled parameter surface, one must store only one knot vector, which is the knot vector shared by the left and right boundary curve, and the NURBS control points of the left and right boundary curve. Provided that all trimming curves are originally defined as cubic B-spline curves, which is usually the case, one does not need to perform degree elevation. Assuming that the original knot vectors of the left and right boundary curves (= segments of cubic B-spline curves) of each parameter surface are entirely different, the number of control points required to represent the left and right boundary curves of all parameter surfaces is approximately twice the number of control

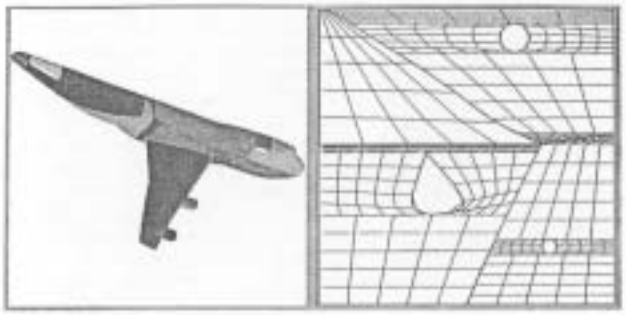

Figure 18 Trimmed surface (fuselage) with multiple trimming curves

points defining the original trimming curves. This fact is independent of the number of parameter surfaces needed to represent the valid part of a trimmed surface.

CONCLUSIONS

A method for representing the region between the trimming curves in the parameter space of a trimmed surface has been presented. A generalized Voronoï diagram is computed for the set of trimming curves, and the resulting tiles associated with each trimming curve are then further subdivided into ruled parameter surfaces whose boundary curves are defined by horizontal line segments and non-horizontal curve segments of the trimming curves and tile boundary curves.

Alternatively, one might want to interpolate between segments on the tile boundary curves and segments on the trimming curves once the generalized Voronoï diagram has been computed. If one decides to do this, one must ensure that the chosen interpolation method guarantees a one-to-one mapping between the (ξ, η)-space (the parameter space of a parameter surface) and the (u, v)-space (the parameter space of a trimmed surface). It will be investigated in which cases a tile can directly and easily be decomposed into parameter surfaces using transfinite interpolation[1] or conformal mappings[37].

The strategy used for representing a single tile by a set of ruled parameter surfaces can also be viewed as a strategy for generating the block boundaries (vertices and edges) of a multi-block decomposition of a 2D region[37]. It is planned to extend this strategy for generating the block boundaries (vertices, edges, and faces) of a multi-block decomposition of a 3D region. Most likely, many of the blocks resulting from the strategy will lead to unacceptable grids. Therefore, one must subdivide the blocks further. This aspect will be investigated in the future.

ACKNOWLEDGEMENTS

This work was supported by the National Grid Project consortium and the National Science Foundation under contract EEC-8907070 to Mississippi State University. Special thanks go to the members of the research and development team of the National Grid Project, which was performed at the NSF Engineering Research Center for Computational Field Simulation, Mississippi State University. The reviewers' comments have improved the paper significantly.

REFERENCES

1 Farin, G *Curves and Surfaces for Computer Aided Geometric Design* (3rd Ed.) Academic Press, San Diego, CA (1993)
2 Hoschek, J and Lasser, D *Fundamentals of Computer Aided Geometric Design* A K Peters, Wellesley, MA (1993)
3 Aurenhammer, F 'Power diagrams: algorithms, and applications' *SIAM J. Computing* Vol 16 (1987) pp 78–96
4 Edelsbrunner, H *Algorithms in Combinatorial Geometry* Springer-Verlag, New York, NY (1987)
5 Facello, M A 'Implementation of a randomized algorithm for Delaunay and regular triangulations in three dimensions' *Comput. Aided Geom. Des.* (1995) (to appear)
6 Preparata, F P and Shamos, M L *Computational Geometry* (3rd Ed.) Springer-Verlag, New York, NY (1990)
7 Farouki, R T and Johnstone, J K 'The bisector of a point and a plane parametric curve' *Comput. Aided Geom. Des.* Vol 11 No 2 (1994) pp 117–151
8 Lee, D T and Drysdale, R L 'Generalization of Voronoï diagrams in the plane' *SIAM J. Computing* Vol 10 No 1 pp 73–87
9 Leven, D and Sharir, M 'Intersection and proximity problems and Voronoï diagrams' *in* Schwartz, J and Yap, C K (Eds.) *Advances in Robotics* Vol I Lawrence Erlbaum, London, UK (1986) pp 187–228
10 Sharir, M 'Intersection and closest-pair problems for a set of discs' *SIAM J. Computing* Vol 14 No 2 (1985) pp 448–468
11 Srinivasan, V and Nackman, L R 'Voronoï diagram for multiply-connected polygonal domains I: Algorithm' *IBM J. Res. Develop.* Vol 31 No 3 (1987) pp 361–372
12 Srinivasan, V, Nackman, L R, Tang, J M and Meshkat, S N 'Automatic mesh generation using the symmetric axis transformation of polygonal domains' *Proc. IEEE* Vol 80 No 9 (1992) pp 1485–1501
13 Yap, C K 'An O (n log n) algorithm for the Voronoï diagram of a set of simple curve segments' *Dis. Comput. Geom.* Vol 2 (1987) pp 365–393
14 Rockwood, A P, Heaton, K and Davis, T 'Real time rendering of trimmed surfaces' *ACM Trans. Graph.* Vol 23 (1989) pp 107–116
15 Klein, R, Mehlhorn, K and Meiser, S 'Randomized incremental construction of abstract Voronoï diagrams' *Comput. Geom. Theor. Appl.* Vol 3 (1993) pp 157–184
16 Baehmann, P L, Wittchen, S L, Shephard, M S, Grice, K R and Yerry, M A 'Robust, geometrically based, automatic two-dimensional mesh generation' *Int. J. Num. Meth. Eng.* Vol 24 (1987) pp 1043–1078
17 Vries-Baayens, A E and Seebregts, C H 'Exact conversion of a trimmed nonrational Bézier surface into composite or basic nonrational Bézier surfaces' *in* Hagen, H (Ed.) *Topics in Surface Modeling* SIAM, Philadelphia, PA (1992) pp 115–143
18 Hoschek, J and Schneider, F J 'Spline conversion for trimmed rational Bézier and B-spline surfaces' *Comput.-Aided Des.* Vol 22 (1990) pp 580–590
19 Piegl, L A and Richard, A M 'Tessellating trimmed NURBS surfaces' *Comput.-Aided Des.* Vol 27 No 1 (1995) pp 16–26
20 Boehm, W and Prautzsch, H *Numerical Methods* A K Peters, Wellesley, MA (1994)
21 Barnhill, R E *Geometry Processing for Design and Manufacturing* SIAM, Philadelphia, PA (1992)
22 Patrikalakis, N M 'Surface-to-surface intersection' *IEEE Comput. Graph. Appl.* Vol 13 No 1 (1993) pp 89–95
23 Thompson, J F and Weatherill, N P 'Aspects of numerical grid generation: current science and art' *Proc. 11th AIAA Applied Aerodynamics Conference* Monterey, CA, August (1993)
24 Weatherill, N P, Hassan, O, Marcum, D L and Marchant, M J 'Grid generation by the Delaunay triangulation' von Karman Institute for Fluid Dynamics, 1993–1994 Lecture Series, January (1994)
25 Blum, H 'Biological shape and visual science (part I)' *J. Theor. Biol.* Vol 38 (1973) pp 205–287
26 Marr, D 'Representing visual information' *AI. Memo 415*, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA (May 1977)
27 Pavlidis, T *Structural Pattern Recognition* Springer-Verlag, New York, NY (1977)

28 Rosenfeld, A and Kak, A C *Digital Picture Processing* Academic Press, New York, NY (1976)

29 Foley, J D, van Dam, A, Feiner, S K and Hughes, J F *Computer Graphics* (2nd Ed.) Addison Wesley, Reading, MA (1990)

30 Kunnii, T L and Shinagawa, Y 'Visualization: new concepts and techniques to integrate diverse application areas' *in* Patrikalakis, N M (Ed.) *Scientific Visualization of Physical Phenomena* Springer-Verlag, New York, NY (1991) pp 3–25

31 Piegl, L A 'On NURBS: a survey' *IEEE Comput. Graph. Appl.* Vol 11 No 1 (1991) pp 55–71

32 Piegl, L A 'Rational B-spline curves and surfaces for CAD and graphics' *in* Rogers, D F and Earnshaw, R A (Eds.) *State of the Art in Computer Graphics* Springer-Verlag, New York, NY (1991) pp 225–269

33 Thompson, J F, Warsi, Z U A and Mastin, C W *Numerical Grid Generation* North-Holland, New York, NY (1985)

34 George, P L *Automatic Mesh Generation* Wiley, New York, NY (1991)

35 Knupp, P M and Steinberg, S L *Fundamentals of Grid Generation* CRC Press, Boca Raton, FL (1993)

Bernd Hamann is an associate professor in the Department of Computer Science at the University of California, Davis, an adjunct professor in the Department of Computer Science and an adjunct research faculty member at the NSF Engineering Research Center for Computational Field Simulation at Mississippi State University, USA. His current research and teaching interests are scientific visualization, computer graphics, and computer-aided geometric design. He received a BS in computer science, a BS in mathematics, and an MS in computer science from the Technical University of Braunschweig, Germany. He received his PhD in computer science from Arizona State University, USA, in 1991.

Po-Yu Tsai is a PhD student in computational engineering at Mississippi State University, USA. His current research interests are computer-aided geometric design and scientific visualization. He received a BS in economics and management information systems from Fu-Jen Catholic University, Taiwan, Republic of China. He received an MS in computer science from Mississippi State University, USA, in 1992.