The Visual Computer

A marching method for the triangulation of surfaces

Erich Hartmann

Darmstadt University of Technology,
Deptartment of Mathematics, Schlossgartenstr. 7,
D-64289 Darmstadt, Germany
E-mail: ehartmann@mathematik.tu-darmstadt.de

All surfaces that can be described by collections of equations, especially the parametric ones, can be treated uniformly as implicit surfaces. The idea of numerical implicitization makes this possible. We introduce a marching method for the triangulation of implicit surfaces. The method produces coherent nets of triangles, even for sets of intersecting surface patches.

Key words: Triangulation – Marching method – Numerical implicitization – Parametric surface – Implicit surface – Offset surface – Blend surface

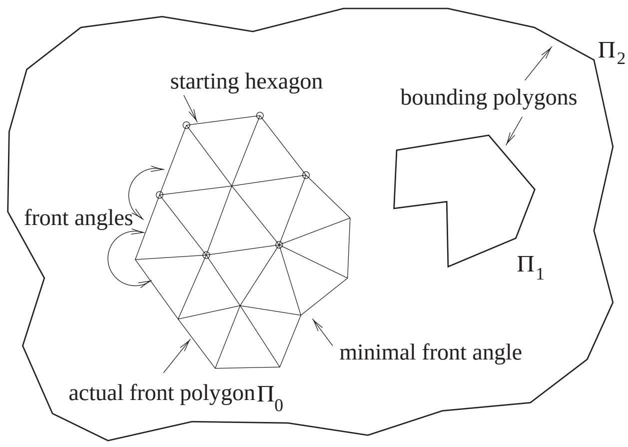
1 Introduction

Surface triangulations are necessary in applying finite element methods for solving mechanical problems and for displaying surfaces by ray tracing or other hidden line algorithms. A parametric surface can be a triangulated by triangulating its (plane) area of definition. However, the images of these triangles in object space may vary unacceptably for the application. Thus we need suitable methods of triangulation even for parametric surfaces. Triangulation algorithms for implicit surfaces are available in the literature. [ALGN'91; BL'88; LO'87; SC'93; WY'86]. All these methods divide the space into suitable polyhedrons (cubes, tetrahedrons) and determine the section of the given implicit surface with the edges of these polyhedrons.
The intention of this paper is to introduce a *marching* method to build a mesh of triangles successively by starting with a point or a prescribed polygon. The triangulation is terminated by several bounding polygons (on the given surface) or a global bounding box. (A similar idea is used in the recently published paper [BAXU'97] on algebraic surfaces.) The method will be established for implicit surfaces. With the idea of *numerical implicitization* introduced in [HA'97], the triangulation is applicable to any surface for which *foot points* (i.e., points of minimal distance to the surface) can be determined.The main advantages of the triangulation presented in this paper are:

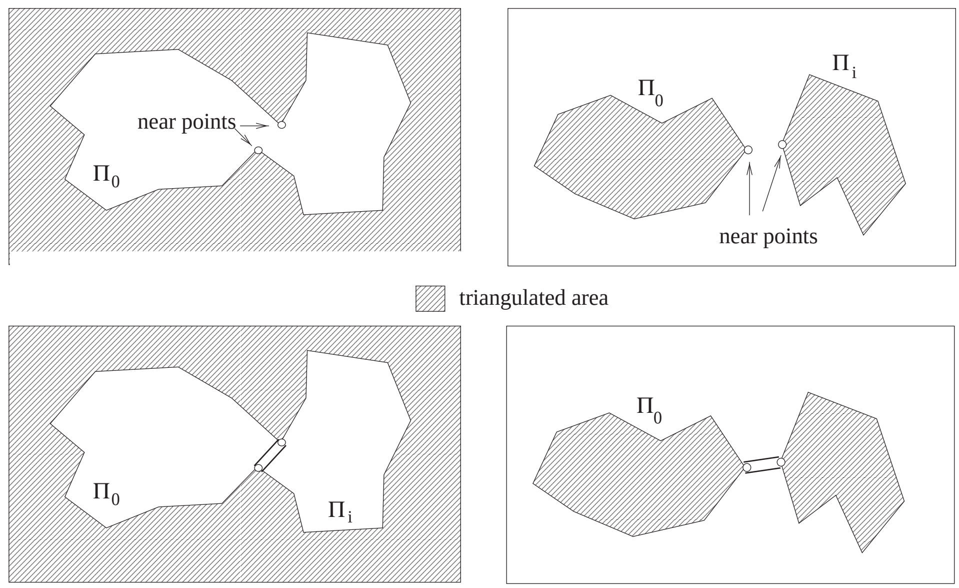
1. The data structure is simple.
2. The termination of the triangulation by prescribed polygons makes it possible for the user to generate a coherent net for intersecting surface patches (Sects. 3.9, 3.11).
3. The method is applicable not only for implicit surfaces, but also for parametric or more general surfaces.

Numerical implicitization of a surface Φ means that a real function f exists such that Φ is implicitly represented by $f=0$, and for a point $\mathbf{x}$ the function value $f(\mathbf{x})$ and the gradient $\nabla f(\mathbf{x})$ can be determined numerically. Usually we choose f so that $f(\mathbf{x})$ is the shortest oriented distance of point $\mathbf{x}$ from the surface Φ and call $f=0$ the *normal form* of Φ analogously to the Hessian normal form of a plane. $\nabla f(\mathbf{x})$ for the oriented distance function is just the unit normal at the corresponding foot point. Especially the normal form of a surface provides an easy representation of its offset surfaces: they are the *level surfaces* $f(\mathbf{x})=c$. Thus, there is even an advantage to implicitizing an implicit

The Visual Computer (1998) 14:95–108



1



2

Fig. 1. Basic notions for the triangulation algorithm

Fig. 2. Dividing (left) and uniting (right) the actual front polygon

surface. The main advantage is that surfaces with different kinds of definitions (parametric, implicit, etc.) can be treated in a uniform way considering intersection, blending, and at least triangulation. The performance of numerical implicitization is described in [HA'98]. It is based on suitable algorithms for determining foot points on surfaces.

2 The triangulation algorithm

The formulation of the triangulation algorithm uses no special representation of the surface to be triangulated. The operations depending on the representation are hidden in the procedure surfacepoint (defined below).

The next subsection gives a survey and basic ideas of the algorithm. Then the procedure surfacepoint and the data structure are introduced, and the steps of the algorithm are explained in detail.

2.1 The idea of the algorithm

Step 0: Choose a point **s** in the neighborhood of the the surface. Determine the corresponding surface

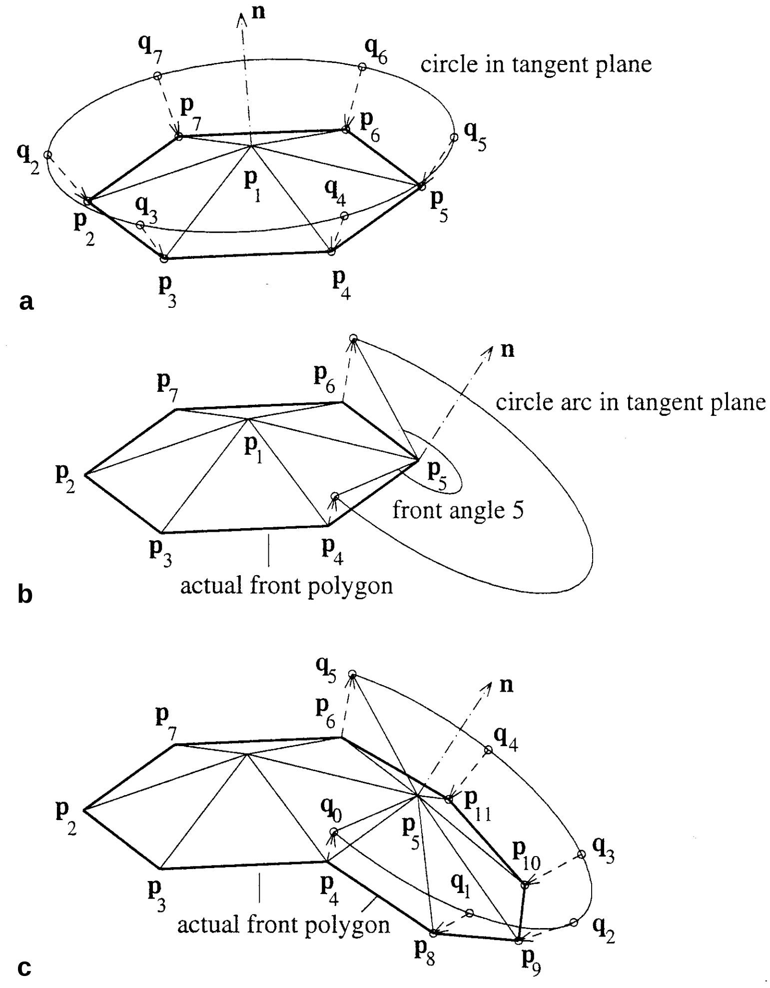


Fig. 3a–c. The first steps of the algorithm

point $\mathbf{p}_1$. Surround $\mathbf{p}_1$ with a regular hexagon $\mathbf{q}_2, \dots, \mathbf{q}_7$ in the tangent plane. With procedure surfacepoint, determine the points $\mathbf{p}_2, \dots, \mathbf{p}_7$ corresponding to the starting points $\mathbf{q}_2, \dots, \mathbf{q}_7$. The triangles of the surface hexagon are the first six triangles of the triangulation (Figs. 1, 3).
We call the ordered array of points $\mathbf{p}_2, \dots, \mathbf{p}_7$ the first *actual front polygon* Π_0. If the triangulation should be limited (not necessary for closed surfaces) by closed surface curves $\Gamma_1, \Gamma_2, \dots$ (c.f. examples below) we determine *bounding* front polygons $\Pi_1, \Pi_2, \dots$ on these curves.
For special surfaces (cylinder, torus, etc.) it might be convenient to start with a prescribed actual front polygon first (c.f. Sect. 3).

Step 1. For every point of the actual front polygon Π_0, we determine the angle of the area till to be triangulated. We call these angles *front angles* (Fig. 1).

Step 2. Check if any point $\mathbf{p}_i$ of the actual front polygon is near
– a point of Π_0 that is different from $\mathbf{p}_i$ and its neighbors or
– a point of any other front polygon Π_k, $k>0$.

In the first case, divide the actual front polygon Π_0 into a smaller one and an additional front polygon (see Figs. 2, 8a, b).
In the second case, if $\mathbf{p}_i$ is near a point of the front polygon Π_m then unite the polygons Π_0, Π_m to a new and larger actual front polygon. Delete Π_m. (see Figs. 2,8d, e)

Step 3. Determine a front point $\mathbf{p}_m$ of the actual front polygon Π_0 with a minimal front angle. Surround $\mathbf{p}_m$ by triangles with angles ≈60°. Delete $\mathbf{p}_m$ from the polygon Π_0 and insert the new points into the actual front polygon Π_0.

Step 4. Repeat steps 1–3 until the actual front polygon Π_0 consists of only three points that generate a new triangle. If there is another (nonempty) front polygon left, it becomes the new actual front polygon Π_0 and steps 1–3 are repeated. If there are no more front polygons, then the triangulation is finished.
If the surface is not bounded, the triangulation should be limited by bounding polygons on the surface or a global bounding box (c.f. Sect. 3).

2.2 The procedure surfacepoint

An essential step of the triangulation algorithm is to determine a surface point $\mathbf{p}$ that is near a given point $\mathbf{q}$ in the vicinity of a surface. $\mathbf{q}-\mathbf{p}$ need not be exactly perpendicular to the surface. Because nearly all surfaces can be numerically implicitized, we give a solution for implicit surfaces.
We start with an implicitly given surface $\Phi{:}f(\mathbf{x})=0$ for which the gradient ∇f exists and is not zero for any point of consideration and a point $\mathbf{q}$ in the neighborhood of the surface.
The following procedure surfacepoint calculates a surface point $\mathbf{p}$, a normal and two tangent vectors at $\mathbf{p}$.

1. (a) $\mathbf{u}_0=\mathbf{q}$
   (b) repeat $\mathbf{u}_{k+1} := \mathbf{u}_k - \frac{f(\mathbf{u}_k)}{\nabla f(\mathbf{u}_k)^2}\nabla f(\mathbf{u}_k)$
   (Newton step for the function $g_k(t){:=}f(\mathbf{u}_k+t\nabla f(\mathbf{u}_k)))$
   until $\|\mathbf{u}_{k+1}-\mathbf{u}_k\|$ is sufficiently small.
   Surface point $\mathbf{p}=\mathbf{u}_{k+1}$.
2. The *surface normal* at surface point $\mathbf{p}$ is $\mathbf{n}{:=}\nabla f(\mathbf{p})/||...||$.
3. For *tangent vectors* we choose
   $\mathbf{t}_1{:=}(n_y, -n_x, 0)/||...||$ if $n_x>0.5$ or $n_y>0.5$
   else $\mathbf{t}_1{:=}(-n_z, 0, n_x)/||...||$
   and $\mathbf{t}_2{:=}\mathbf{n}\times\mathbf{t}_1$ where $(n_x, n_y, n_z){:=}\mathbf{n}$.

2.3 The data structure

For the construction of the triangles, we need a *step length* $\delta_t>0$ that is approximately the length of the edges.
The *points* of the triangulation get current numbers. For any point $\mathbf{p}_i$ we keep the following information:

- The coordinates
- The surface normal $\mathbf{n}$ and tangent vectors $\mathbf{t}_1$, $\mathbf{t}_2$ such that $\mathbf{n}$, $\mathbf{t}_1$, $\mathbf{t}_2$ are orthonormal
- The actual front angle if $\mathbf{p}_i$ is an actual front point
- The boolean variable `angle_changed` with `angle_changed=true` if the actual front angle was changed and has to be recalculated

  the boolean variable `border_point`. It is set to true if point $\mathbf{p}_i$ is on the border of the triangulation and should be ignored in further considerations (recalculation of front angle, distance check (Step 2).

The *triangles* are numbered consecutively. For each triangle, we store the numbers of the vertices. The *front polygons* Π_0, Π_1, ... are represented by the integer arrays of their point numbers.

2.4 Step 0

Let $\mathbf{s}$ be a starting point in the vicinity of the surface. The procedure surfacepoint determines the first point $\mathbf{p}_1$ of the triangulation and the orthonormal system $\mathbf{n}_1$, $\mathbf{t}_{11}$, $\mathbf{t}_{12}$. The following six points $\mathbf{p}_2, \ldots, \mathbf{p}_7$ are the results of procedure surfacepoint applied to

$$\mathbf{q}_{i+2} := \mathbf{p}_1 + \delta_t\cos(i\pi/3)\mathbf{t}_{11} + \delta_t\sin(i\pi/3)\mathbf{t}_{12},$$
$$i := 0, \ldots, 5,$$

which are points of a regular hexagon in the tangent plane at $\mathbf{p}_1$.
We get the first six triangles (Fig. 3a):

$(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$, $(\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_4)$, $(\mathbf{p}_1, \mathbf{p}_4, \mathbf{p}_5)$, $(\mathbf{p}_1, \mathbf{p}_5, \mathbf{p}_6)$, $(\mathbf{p}_1, \mathbf{p}_6, \mathbf{p}_7)$, $(\mathbf{p}_1, \mathbf{p}_7, \mathbf{p}_2)$.

2.5 Step 1

If a point $\mathbf{p}_{0i}$ of the actual front polygon $\Pi_0=(\mathbf{p}_{01}, \mathbf{p}_{02}, \ldots, \mathbf{p}_{0N_0})$ has just been inserted or if a neighbor of $\mathbf{p}_{0i}$ is a new point, then it is necessary to recalculate the actual front angle ω at point $\mathbf{p}_{0i}$. Let

$\mathbf{v}_1 := \mathbf{p}_{0,i-1}$ if $i > 1$ or $\mathbf{v}_1 := \mathbf{p}_{0N_0}$ if $i = 1$,

$\mathbf{v}_2 := \mathbf{p}_{0,i+1}$ if $i < N_0$ or $\mathbf{v}_2 := \mathbf{p}_{01}$ if $i = N_0$ and

(ξ_1, η_1, ζ_1) the coordinates of $\mathbf{v}_1$, (ξ_2, η_2, ζ_2) the coordinates of $\mathbf{v}_2$ in the local orthonormal system $\mathbf{n}$, $\mathbf{t}_1$, $\mathbf{t}_2$ at point $\mathbf{p}_{0i}$,
ω_1:= polar angle of (ξ_1, η_1), ω_2:=polar angle of (ξ_2, η_2). Then the front angle at point $\mathbf{p}_{0i}$ is $\omega=\omega_2-\omega_1$ if $\omega_2 \geq \omega_1$ otherwise $\omega=\omega_2-\omega_1+2\pi$. (Fig. 3b).

2.6 Step 2

In order to prevent new triangles from overlapping existing triangles, we check:

- The distances of pairs of points of the actual front polygon Π_0. If there are points $\mathbf{p}_{0i}$, $\mathbf{p}_{0j}$, $i<j$, that are neither neighbors nor neighbors of neighbors and $\|\mathbf{p}_{0i}-\mathbf{p}_{0j}\|<\delta_t$, then the actual front polygon is split into the new actual front polygon $\mathbf{p}_{01}, \ldots, \mathbf{p}_{0i}, \mathbf{p}_{0j}, \ldots, \mathbf{p}_{0N_0})$ with $N_0-(j-i-1)$ points and a further front polygon $(\mathbf{p}_{0i}, \ldots, \mathbf{p}_{0j})$ with $j-i+1$ points (Figs. 2, 6). $\mathbf{p}_{0i}$ and $\mathbf{p}_{0j}$ must not be involved in later distance checks.
- The distance of the points of the actual front polygon Π_0 to points of all further front polygons Π_k, $k> 0$. If there are points $\mathbf{p}_{0i}\in\Pi_0$ and $\mathbf{p}_{mj}\in\Pi_m$ with $\|\mathbf{p}_{0i}-\mathbf{p}_{mj}\|<\delta_t$, then the polygons

  $\Pi_0=(\mathbf{p}_{01}, \ldots, \mathbf{p}_{0N_0})$ and $\Pi_m=(\mathbf{p}_{m1}, \ldots, \mathbf{p}_{mN_m}$

  are united with the new actual front polygon

  $\Pi_0=(\mathbf{p}_{01}, \ldots, \mathbf{p}_{0i}, \mathbf{p}_{mj}, \ldots \mathbf{p}_{mN_m}, \mathbf{p}_{m1}, \ldots, \mathbf{p}_{mj}, \mathbf{p}_{0i}, \ldots, \mathbf{p}_{0N_0})$

with N_0+N_m+2 points (Fig. 2). The points $\mathbf{p}_{0i}$ and $\mathbf{p}_{mj}$ appear twice! Before any further action is taken, one should determine the front angles of these points when they first appear in the polygon Π_0 and first surround with triangles, see step 3, that point with the smallest angle then the second (Fig. 8e). After this operation, the first appearance of these two points is deleted from the actual front polygon. The points $\mathbf{p}_{0i}$ and $\mathbf{p}_{mj}$ must not be involved in later distance checks.

Remarks

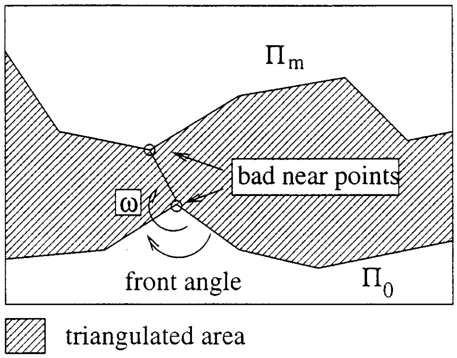
1. For "simple" surfaces, the distance check can be omitted (see Sect. 3.1, 3.7).
2. Before applying the distance check, one should complete points with front angles smaller than (about) 60º.
3. The rare case of "bad" near points $\mathbf{p}_{0i}$, $\mathbf{p}_{mj}$ that are connected by an already triangulated area (Fig. 4) can be detected by calculating the angle ω at point $\mathbf{p}_{0i}$ described in step 1 using $\mathbf{p}_{mj}$ instead of $\mathbf{v}_2$. $\mathbf{p}_{0i}$, $\mathbf{p}_{mj}$ are "bad" near points if ω is greater than the front angle at point $\mathbf{p}_{0i}$.
4. An essential acceleration of the distance check can be achieved by using bounding boxes of the front polygons.

2.7 Step 3

Let $\mathbf{p}_{0m}$ be a point of the actual front polygon Π_0 with a minimal front angle ω. Complete the triangulation at point $\mathbf{p}_{0m}$ in the following way:

1. Determine the neighbors $\mathbf{v}_1$, $\mathbf{v}_2$ of $\mathbf{p}_{0m}$ (c.f. step 1).
2. Determine the number of triangles n_t to be generated.
   Let n_t:=`trunc`$(3\omega/\pi)+1$, $\Delta\omega:=\omega/n_t$
   Correct $\Delta\omega$ for extreme cases.
   If $\Delta\omega<0.8$ and $n_t>1$, then $n_t\rightarrow n_t-1$ and $\Delta\omega=\omega/n_t$ (Fig. 5a).
   If $n_t=1$ and $\Delta\omega>0.8$ and $\|\mathbf{v}_1-\mathbf{v}_2\|>1.2\,\delta_t$ then $n_t=2$ and $\Delta\omega\rightarrow \Delta\omega/2$ (Fig. 5b).
   If $\omega<3$ and ($\|\mathbf{v}_1-\mathbf{p}_{0m}\|\leq 0.5\,\delta_t$ or $\|\mathbf{v}_2-\mathbf{p}_{0m}\|\leq 0.5\,\delta_t$) then $n_t=1$ (Fig. 5c).
3. Generate the triangles.
   If $n_t=1$, we get one new triangle: $(\mathbf{v}_1, \mathbf{v}_2, \mathbf{p}_{0m})$.
   Otherwise let $\mathbf{q}_0$, $\mathbf{q}_{n_t}$ be the orthogonal projection of $\mathbf{v}_1$, $\mathbf{v}_2$ into the tangent plane at point $\mathbf{p}_{0m}$ and let $\mathbf{q}_i$ be the result of a rotation of $\mathbf{p}_{0m}+\delta_t(\mathbf{q}_0-\mathbf{p}_{0m})/\|\mathbf{q}_0-\mathbf{p}_{0m}\|$ by the angle $i\Delta\omega$ around the normal at surface point $\mathbf{p}_{0m}$ for $i=1, \ldots, n_t-1$. (If a global bounding box is valid, the chord $\mathbf{p}_{0m}\mathbf{q}_i$ is truncated and the variable `border_point` of the corresponding new surface point is set to true. Border points will not be considered any more.) Applying procedure `surfacepoint` on $\mathbf{q}_i$, we get the new points $\mathbf{p}_{N+i}$, $i=1, \ldots, n_t-1$, where N is the total number of points already existing and n_t new triangles (see Fig. 3b, c):

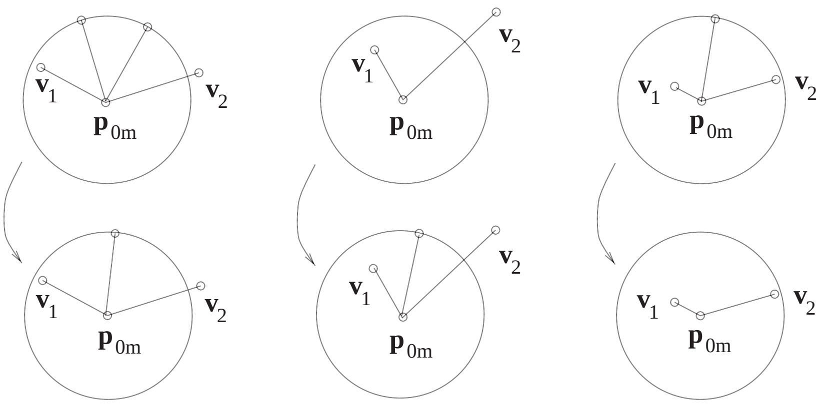
$$(\mathbf{v}_1, \mathbf{p}_{N+1}, \mathbf{p}_{0m}), \quad (\mathbf{p}_{N+1}, \mathbf{p}_{N+2}, \mathbf{p}_{0m}), \ldots,$$
$$(\mathbf{p}_{N+n_t-1}, \mathbf{v}_2, \mathbf{p}_{0m}).$$



4

Fig. 4. Bad near points and their detection

Fig. 5a–c. Corrections for extreme cases



5a 5b 5c

4. Renew the actual front polygon.
   Delete point $\mathbf{p}_{0m}$ and, if $n_t>1$, insert the new points $\mathbf{p}_{N+1}, \ldots, \mathbf{p}_{N+n_t-1}$.
   All boolean variables *angle_changed* at points $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{p}_{N+1}, \ldots, \mathbf{p}_{N+n_t-1}$. are set to true to ensure recalculation of their front angles.

3 Examples

3.1 Sphere

Triangulation of the sphere $x^2+y^2+z^2-4=0$ with a starting point (1, 1, 1) and a step length $\delta_t=0.3$. Figure 6 shows the first four actual front polygons and the situation after the generation of 101 and 1531 triangles. The total triangulation of the sphere involves 1544 triangles.

3.2 Cylinder

Triangulation of the cylinder $x^2+y^2-1=0$

1. With starting point $(1, 0, 0)$ and $\delta_t=0.2$ before and after the first splitting of the actual front polygon. The cylinder is bounded by a bounding box (Fig. 7a–c).
2. With points on the top circle as the starting actual front polygon and points on the basic circle as a bounding front polygon (Fig. 7d).

3.3 Torus

The triangulation of the torus

$$(x^2+y^2+z^2+r^2-a^2)^2-4r^2(x^2+y^2)=0,$$

$$r=1,\ a=0.35$$

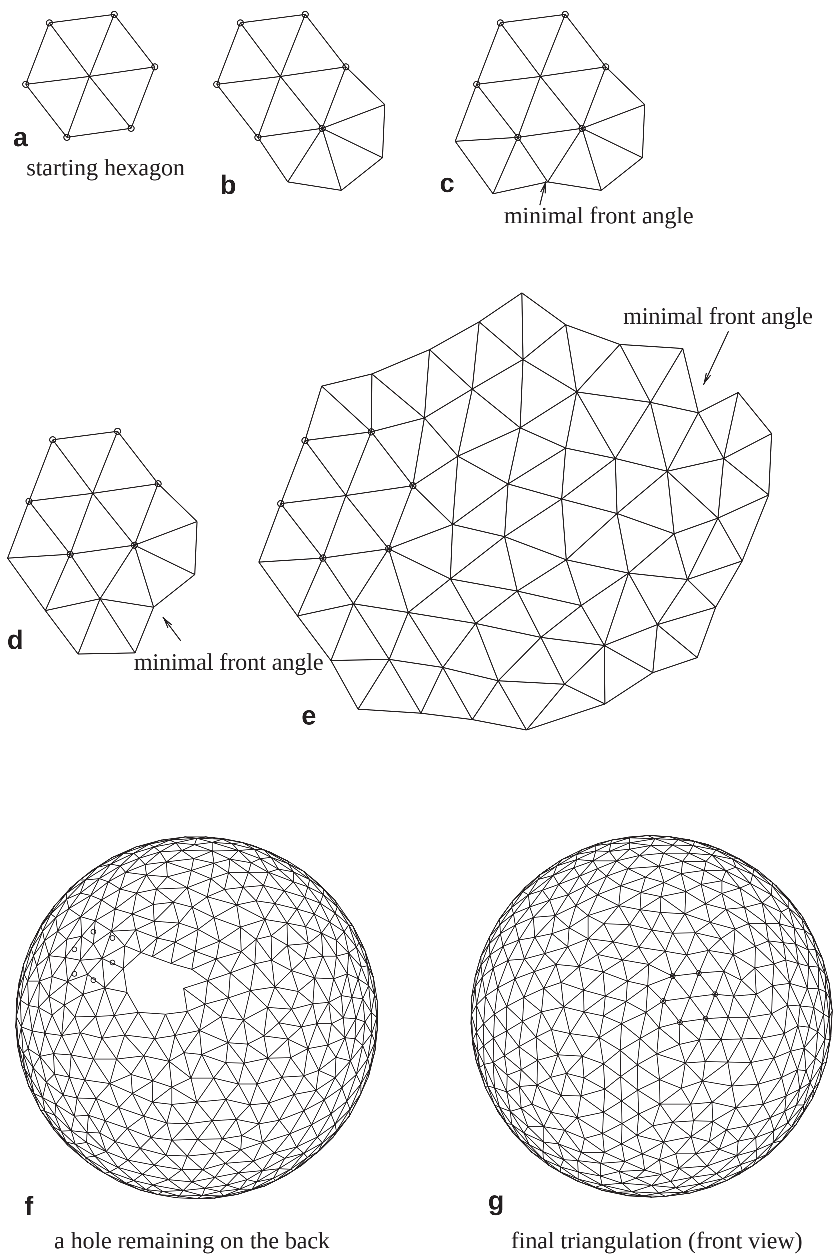


Fig. 6a–g. Triangulation of a sphere

with δ_t=0.1 as shown in Fig. 8 with starting point (1, 0, 0.5). We see the stages before and after the first dividing of the front polygon Π_0 (Fig. 8a, b), before and after uniting front polygon Π_0 with front polygon Π_1 that was generated by the first dividing (Fig. 8c, d), and the complete triangulation. Figure 9a shows a triangulation of a torus part starting with a polygon on a circle as first actual front polygon Π_0 and a bounding polygon on a second circle. It also shows an entire torus where the starting and bounding polygons coincide.

3.4 Six-peak surface

The triangulation of the rather complicated implicit surface

$(3\,x^2-y^2)^2y^2-(x^2+y^2)^4-z^3-0.001\,z=0$

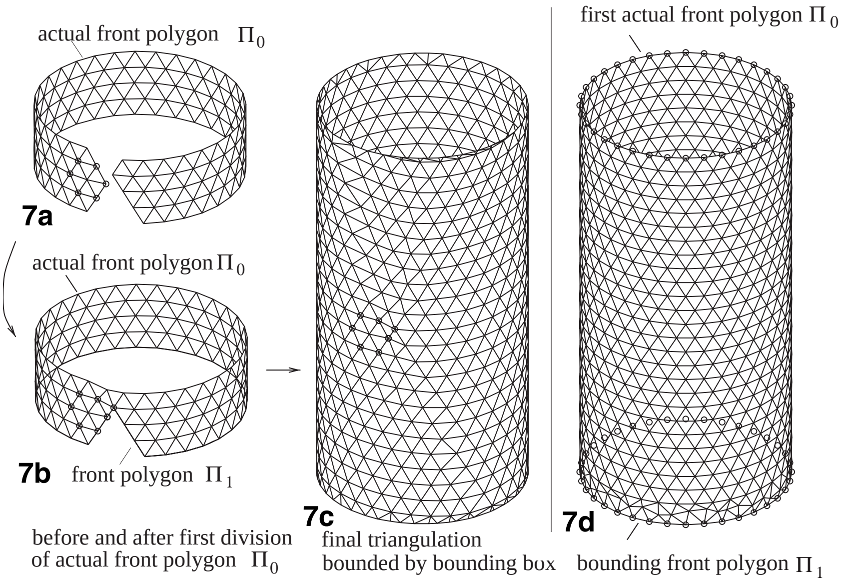


Fig. 7a–d. Triangulation of a cylinder

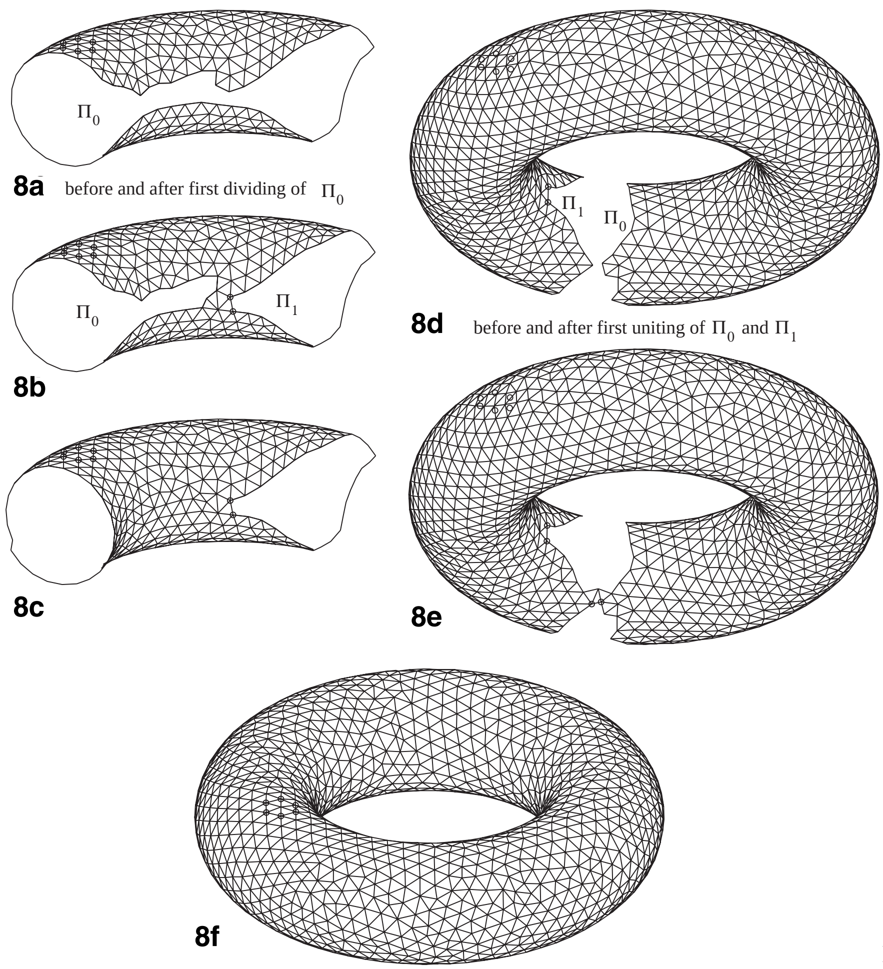
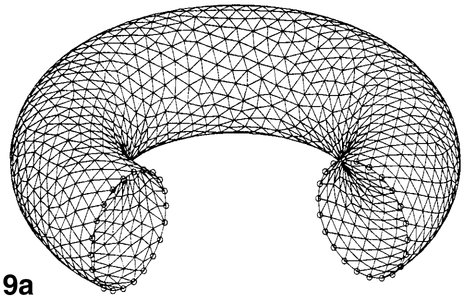
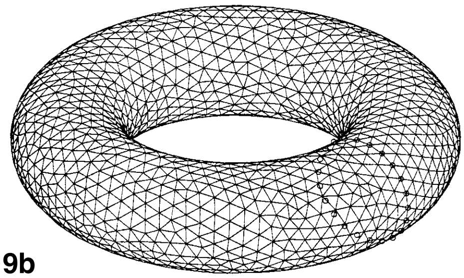


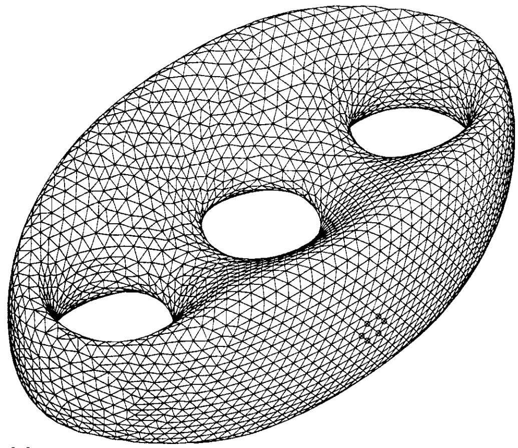
Fig. 8a–f. Triangulation of a torus



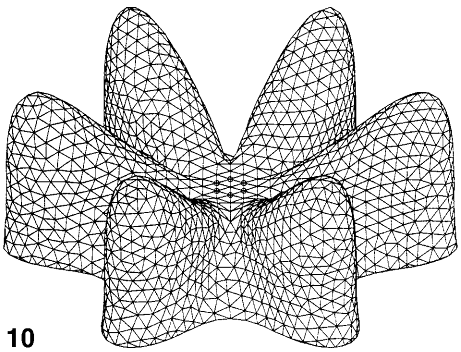
9a



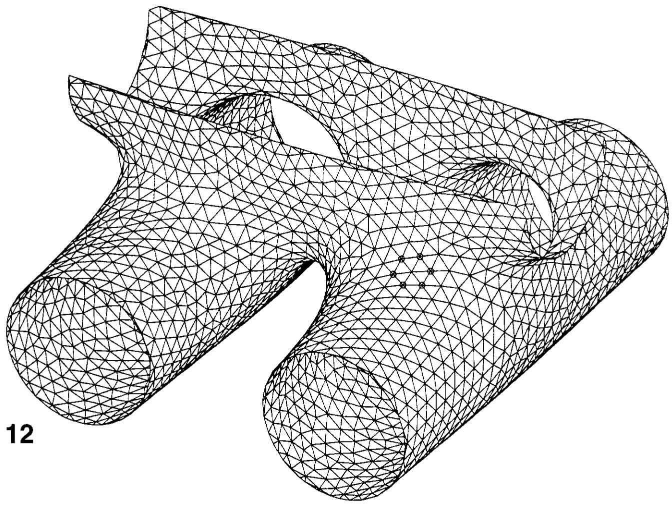
9b



11



10



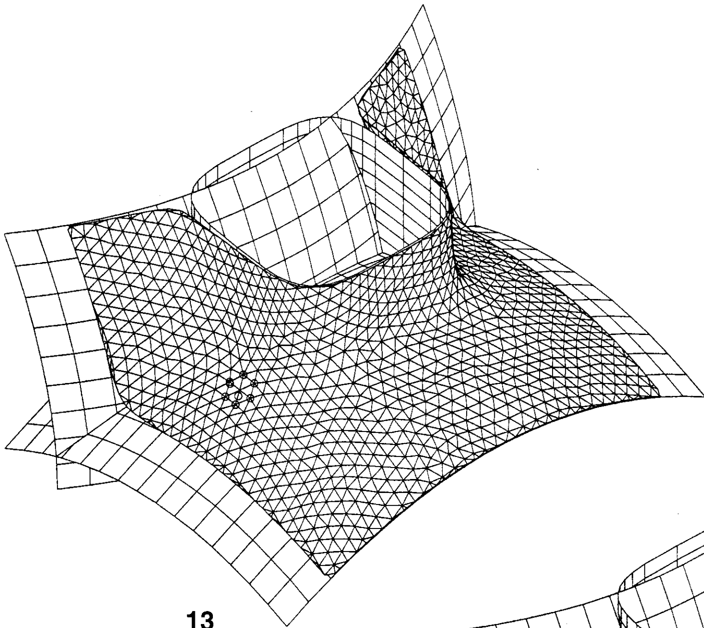
12

Fig. 9a–b. Triangulation of a torus (continued)

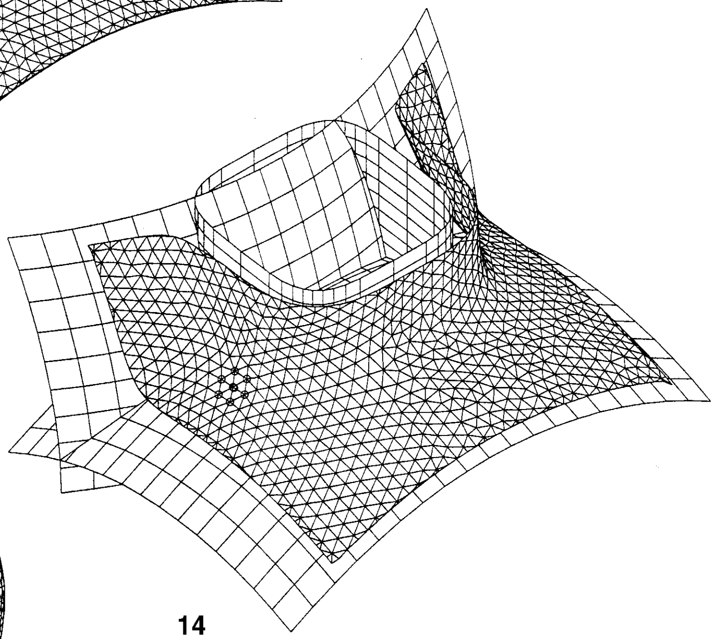
Fig. 10. Triangulation of a six-peak surface

Fig. 11. Triangulation of a surface of genus 3

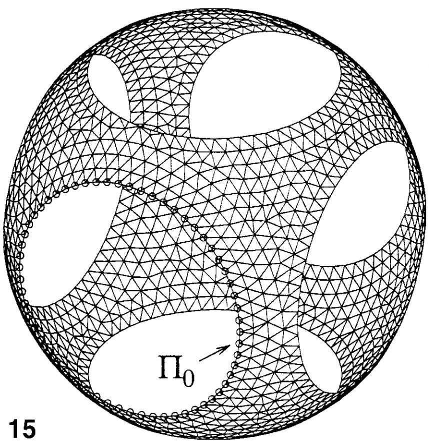
Fig. 12. Triangulation of an approximation of three horizontal cylinders



13



14



15

Fig. 13. Triangulation of an approximation of a set of surfaces

Fig. 14. Triangulation of an approximation of a set of offset surfaces

Fig. 15. Triangulation of a sphere with six holes

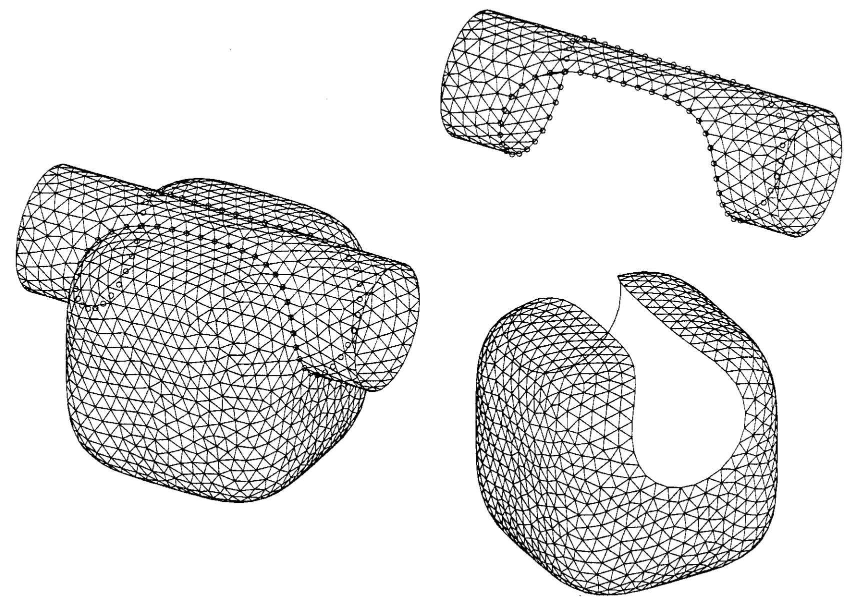


Fig. 16. Triangulation of a set of two intersecting surfaces

is possible without dividing and uniting. For Fig. 10, the starting point is (0, 0, 0.1), the step length δ_t=0.07, and the bounding box $-1.2 \le x, y \le 1.2$, $-0.3 \le z \le 1$.

3.5 Surface of genus 3

Figure 11 shows a triangulation of the implicit surface of genus 3 with the equation

$$r_z^4 z^2 - \left(1 - (x/r_x)^2 - (y/r_y)^2\right) \left((x-x_1)^2 + y^2 - r_1^2\right)\left(x^2+y^2-r_1^2\right) \left((x+x_1)^2 + y^2 - r_1^2\right) = 0$$

and parameters r_x=6, r_y=3.5, r_z=4, r_1=1.2, x_1=3.9, starting point (0, 3, 0), and step length δ_t=0.3. The triangulation consists of 7354 triangles.

3.6 Approximation of three horizontal cylinders

Given are three horizontal cylinders $f_1(\mathbf{x})$: $=x^2+(z-5)^2-4=0$, $f_2(\mathbf{x}):=(y+4)^2+z^2-4=0$, and $f_3(\mathbf{x})$: $=(y-4)^2+z^2-4=0$.

The implicit surface $f:=f_1 f_2 f_3 - c = 0$, $c>0$, is a smooth approximation of the set of three cylinders. Figure 12 shows a triangulation of the approximation surface for c=2287.5587, step length δ_t=0.5, and bounding box $-7 \le x, y, \le 7$, $-7 \le z \le 5$. The triangulation consists of 5297 triangles.

3.7 Approximation of a set of intersecting surfaces

Given are:

1. The implicit surface

   $(x-2)^4 + y^4 - r_1^4 = 0,\ r_1 = 2$

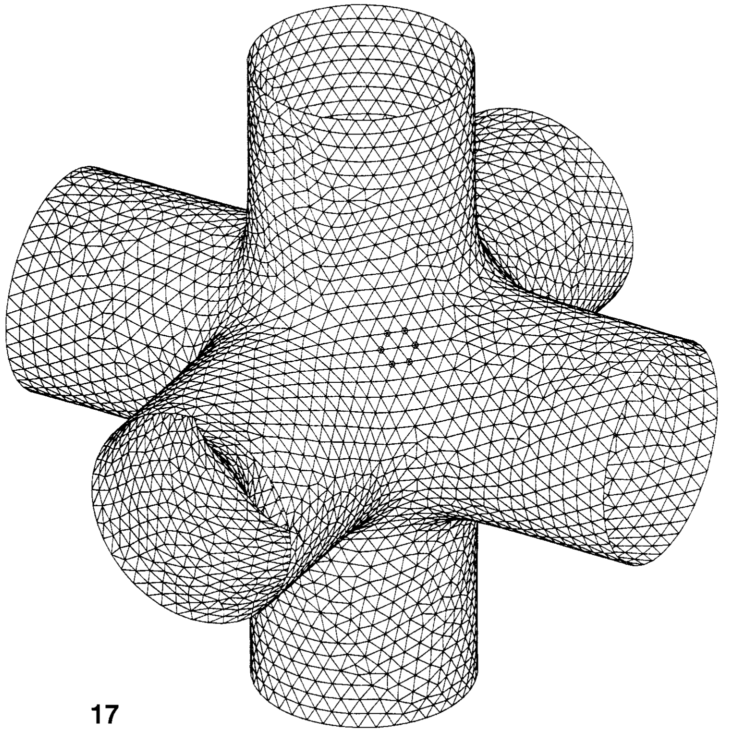
2. The parametric surface patch

   $\mathbf{x}=(10\,v-5,\ 10\,u-5,\ 6(u-u^2+v-v^2))$,
   $0 \le u \le 1$, $0 \le v \le 0.8$

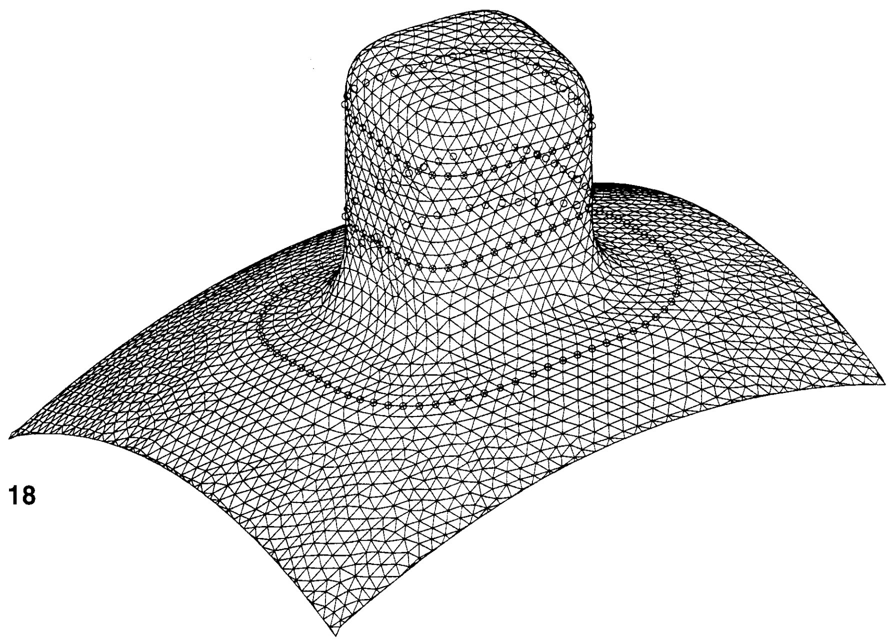
3. The parametric surface patch

   $\mathbf{x}=(6(u-u^2+v-v^2)-5,\ 10\,u-5,\ 10\,v-5)$,
   $0 \le u \le 1$, $0.5 \le v \le 1$.

Let $f_1(\mathbf{x})=c_1, f_2(\mathbf{x})=c_2, f_3(\mathbf{x})=c_3$ be the numerically implicitized pencils of offset surfaces (c.f. [HA'97]) of the first, second, and third surfaces. The given surfaces fulfill the equations f_1=0, f_2=0, f_3=0, respectively. The implicit surface



17



18

Fig. 17: Triangulation of a blending surface of three intersecting cylinders

Fig. 18. Triangulation of a G^2-continuous set of surfaces

$f(\mathbf{x}) := f_1(\mathbf{x})\, f_2(\mathbf{x})\, f_3(\mathbf{x}) = c > 0$ is a smooth approximation of the set of the three given surfaces. Figure 13 shows the triangulation of $f(\mathbf{x}) = c$ for $c = 0.2$. The triangulation is limited by a bounding box.

If we take the equation $(f_1(\mathbf{x}) - c_1)\,(f_2(\mathbf{x}) - c_2)\,(f_3(\mathbf{x}) - c_3) = c > 0$, we get a smooth approximation of the set of offset surfaces $f_1(\mathbf{x}) = c_1$, $f_2(\mathbf{x}) = c_2$, $f_3(\mathbf{x}) = c_3$. Figure 14 shows the case $c_1 = c_2 = c_3 = 0.4$ and $c = 0.2$.

3.8 Sphere with six holes

Figure 15 shows a triangulation of the sphere $x^2+y^2+z^2-r^2$ truncated by the planes $x=\pm a$, $y=\pm a$, $z=\pm a$ with $r=2$, $a=1.6$. The six boundaries (circles) are used for generating the first actual front polygon Π_0 and five bounding front polygons $\Pi_1, \ldots, \Pi_5$. The triangulation uses the step length $\delta_t = a\,\pi/30$ and consists of 2206 triangles.

3.9 Set of two intersecting surfaces

Figure 16 shows a triangulation of the two intersecting surfaces $x^4+y^4+z^4=16$ and $x^2+(z-1.1)^2=1$. A polygon on the intersection curve is used for the first actual front polygons of both the surfaces. The cylinder is truncated by a bounding box.

3.10 G^2-continuous blending of three cylinders

The implicit surface

$$(1-\mu)\,(x^2+y^2-1)\,(x^2+z^2-1)\,(y^2+z^2-1)-\mu(9-x^2-y^2-z^2)^3=0$$

is a G^2-continuous (i.e., curvature continuous) blending surface of the three cylinders $x^2+y^2-1=0$, $x^2+z^2-1=0$, $y^2+z^2-1=0$ (c.f. [LI,HO,-HA'90]). The parameter μ chosen for Fig. 17 is 0.0003.

The triangulation starts with starting point (1, 1, 1) near the surface (c.f. step 0 of the algorithm). It is limited by six bounding polygons $\Pi_1, \ldots, \Pi_6$ on the curves of contact (circles) with the cylinders using step length $\delta_t = \pi/20$ and consisting of 8062 triangles.

3.11 G^2-continuous set of surfaces

Given are:

1. The implicit surface
   $$(x+1.2)^4+(y+1)^4-r_1^4=0, \quad r_1=1.3,$$
2. The parametric surface patch
   $\mathbf{x}=(10\,v-5,\ 10\,u-5,\ 6(u-u^2+v-v^2))$,
   $0 \le u \le 1$, $0 \le v \le 0.8$.

Let $f_1(\mathbf{x}) = c_1$, $f_2(\mathbf{x}) = c_2$ be the numerically implicitized pencils of offset surfaces (c.f. [HA'97]) of the first and second surface. (The given surfaces full fill the equations $f_1 = 0$, $f_2 = 0$).

We establish two blending surfaces (c.f. [LI,HO,-HA'90]):

1. $$f_3(\mathbf{x}) := (1-\mu)\frac{f_1(\mathbf{x})}{c_1}\frac{f_2(\mathbf{x})}{c_2} - \mu\left(1-\frac{f_1(\mathbf{x})}{c_1}-\frac{f_2(\mathbf{x})}{c_2}\right)^3 = 0, \quad 0<\mu<1$$
   (c.f. [HA'97]) is a G^2-continuous blending surface between the given surfaces.
2. $f_4(\mathbf{x}) := (1-\lambda)\, f_1(\mathbf{x}) - \lambda(z-z_4)^3 = 0, \quad 0 < \lambda < 1,$
   is a G^2-continuous closure of the cylinder (top).

The parameters for Fig. 18 are: $c_1=1$, $c_2=1$, $\mu=0.1$, $\lambda=0.8$, $z_4=5$.

4 Conclusion

Each of the pictures shown in this paper was produced within seconds. Thus, a fast and simple method for the triangulation of implicit surfaces has been introduced. Using numerical implicitization the method is applicable to nearly arbitrary surfaces. Further investigation will consider the optimization of triangulations produced by the given method.

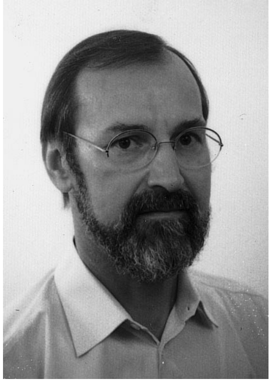
References

[ALGN'91] Allgower EL, Gnutzmann S (1991) Simplicial pivoting for mesh generation of implicitly defined surfaces. Comput Aided Geom Des 8:305–325.

[BAXU'97] Bajaj C, Xu G (1997) Spline approximations of real algebraic surfaces. J Symbolic Comput 23:315–333

[BL'88] Bloomenthal J (1988) Polygonization of implicit surfaces. Comput Aided Geom Des 5:341–355

[GO'96] Görg A (1996) Die Flächenschnittmethode zur Erzeugung GC^n-stetiger Flächenübergänge. (Dissertation) Shaker, Aachen, Germany
[HA'90] Hartmann E (1990) Blending of implicit surfaces with functional splines. Comput Aided Design 22:500–506
[HA'95] Hartmann E (1995) Blending an implicit with a parametric surface. Comput Aided Geom Des 12:825–835
[HA'98] Hartmann E (1998) Numerical implicitization for intersection and G^n-continuous blending of surfaces. Comput Aided Geom Des 15:377–397
[LI,HO,HA'90] Li J, Hoschek J, Hartmann E (1990) G^{n-1}-functional splines for interpolation and approximation of curves, surfaces and solids. Comput Aided Geom Des 7:209–220
[LO'87] Lorensen WE (1987) Marching cubes: a high resolution 3D surface construction algorithm. Comput Graph 21:163–169
[SC'93] Schmidt M (1993) Cutting cubes – visualizing implicit surfaces by adaptive polygonization. Visual Comput 10: 101–115
[VL'90] Vlassopoulos V (1990) Adaptive polygonization of parametric surfaces. Visual Comput 6:291–298
[WY'86] Wyvill G, McPheeters C, Wyvill B (1986) Data structure for soft objects. Visual Comput 2:227–234



ERICH HARTMANN studied physics and received a PhD (1973) in mathematics from the Technical University of Darmstadt where he is professor of mathematics. He has been active in the field of *foundation of geometry* and *geometric algebra.* Since 1987 his research interests include *computer aided geometric design*.