Trimming Implicit Surfaces

Alexander Pasko

Hosei University, Tokyo, Japan, E-mail: pasko@k.hosei.ac.jp

Galina Pasko

IT Institute, Kanazawa Institute of Technology, Tokyo, Japan, E-mail: gpasko@iti.kanazawa-it.ac.jp

ABSTRACT

Algorithms of trimming implicit surfaces yielding surface sheets and stripes are presented. These two-dimensional manifolds with boundaries result from set-theoretic operations on an implicit surface and a solid or another implicit surface. The algorithms generate adaptive polygonal approximation of the trimmed surfaces by extending our original implicit surface polygonization algorithm. The presented applications include modeling several spiral shaped surface sheets and stripes (after M. Escher's art works) and extraction of ridges on implicit surfaces. Another promising application of the presented algorithms is modeling heterogeneous objects as implicit complexes.

1. INTRODUCTION

This work was inspired by art works of M.C. Escher namely "Sphere Spirals" (1958), "Bond of Union" (1956), and "Rind" (1955), showing spiral shaped surface sheets cut of a sphere and human head surfaces. These art works raise two questions:

1) How does one define a geometric model for a surface sheet of this type?

2) How does one visualize such a geometric model?

Modeling heterogeneous objects is becoming an important research topic in different application areas of shape modeling. Such objects are heterogeneous from the points of view of their internal structure and dimensionality. Dimensionally heterogeneous objects include elements of different dimension (points, curves, surfaces, solids) combined into a single entity from the geometric point of view (point set) and the topological point of view (cellular complex) [7]. The recently introduced cellular-functional model [1] allows for representing a heterogeneous object as a cellular complex with explicit cells (BRep, parametric curves, wireframes, point lists) and implicit cells (implicit surface patches, intersections of implicit surfaces). An implicit surface patch can be defined in different ways, for example, as an implicit

surface trimmed by an intersecting solid. Some applications such as physical simulation based on finite-element meshes require conversion of an implicit complex to the pure cellular representation. Such a conversion of a trimmed implicit surface to a polygonal mesh is a subject of this paper.

It was proposed in [12] to use CSG (Constructive Solid Geometry) solids for trimming of boundary faces of CSG models to isolate, in a primitive's face, its contribution to the boundary of the solid. Such a formulation was shown to present considerable computational advantages over trimming curves commonly used in boundary representations and to have applications in rendering and point-membership classification. In our work, we pursue this idea to model implicit surface sheets and stripes.

A surface sheet or stripe can be mathematically defined as a two-dimensional manifold with boundary (or simply a 2-manifold). Modeling and visualization of the above-mentioned spiral type 2-manifolds using parametric surfaces seems to be a difficult task. The alternative is to use isosurfaces of functions of three variables (so-called implicit surfaces). A surface sheet can be represented as a set-theoretic difference between some initial implicit surface $f_A(x,y,z) = 0$ and a trimming solid $f_B(x,y,z) \geq 0$ (see Fig. 1, lower left, for modeling Escher's "Sphere Spirals"). In the case of the intersection with another implicit surface $f_B(x,y,z) = 0$, the intersection curves can be presented by thin surface stripes (Fig. 1, lower right). In this paper we discuss problems of defining initial surfaces and trimming solids/surfaces, and polygonization of the trimmed surface with the mesh adaptation to the surface-surface intersection curves composing the manifold boundary.

If we define the initial surface A as $f_A(x,y,z) = 0$ and the trimming solid B as $f_B(x,y,z) \geq 0$, the trimmed surface $T = A \setminus B$ can be defined as $f_T(x,y,z) = 0$ with

$$f_T = -f_A^2 \,\&\, (-f_B),$$

where symbol & denotes $min(f_1, f_2)$ or some other R-function corresponding to the set-theoretic intersection [13, 11]. The idea proposed by Rvachev [13] is to represent the surface as a solid with $-f_A^2 \geq 0$ and further to apply set-theoretic operations to it. A similar approach is used in the modern solid modeller Svlis [5]. The serious disadvantage of this approach is that it becomes not possible to

distinguish two sides of the initial surface represented as $-f_A^2 \geq 0$. Therefore, it is not possible to apply conventional polygonization algorithms based on inside-outside point classification [4,9,10].

Bloomenthal and Ferguson [3] proposed a general polygonization algorithm for non-manifold surfaces. The algorithm can polygonize both 2-manifolds with boundaries and non-manifold surfaces. Because of the algorithm complexity, it is difficult to implement its adaptive version. A more simple algorithm processing only 2-manifolds with boundaries can be obtained by extending conventional polygonization algorithms.

To polygonize a boundary of a CSG solid, Wyvill and van Overveld [14] applied the uniform subdivision algorithm to the resulting surface and then enhanced the mesh with an iterative numerical procedure taking into account intersection curves of primitive surfaces. In general, the problem we deal with in this paper is related to the intersection of two implicit surfaces. There are two main numerical approaches to such implicit surface-surface intersection:

- Start with some intersection point found analytically or numerically. Trace the intersection curve by solving a differential equation (see, for example, [6]).
- Approximate both surfaces by polygons and intersect two obtained polyhedrons.

Note that the above-mentioned methods treat both surfaces equally. On the other hand, one could observe in Fig. 1 that for a simple initial surface the function defining a trimming solid can be quite complex. It means that evaluation times for these two functions can differ drastically. Therefore, an algorithm aiming to decrease the number of evaluations of the more complex function can substantially decrease the overall computation time.

In this work we propose an alternative approach based on the polygonization of the initial surface using the extension of existing implicit surface polygonization algorithms. The extension includes mesh adaptation to the surface-surface intersection curves composing the 2-manifold boundary and testing obtained polygons against the trimming solid or another intersecting surface. The polygon subdivision is

used here to remove trimmed parts of the initial implicit surface and more precisely approximate the trimmed surface boundary as a side effect.

2. BASIC POLYGONIZATION ALGORITHM

The problem solved by a polygonization algorithm is to approximate a given implicit surface by a set of polygons. In this section, we briefly describe the basic polygonization algorithm used in this work [9,10]. The algorithm is based on the uniform bounding box subdivision, trilinear interpolation inside an elementary cell, using hyperbolic arcs for resolving topological ambiguities, and edge connectivity graph tracing for generation of polygons (see Fig. 2).

A parallelepiped is given in three-dimensional Euclidean space and supposed to contain the implicit surface. By introducing a regular point grid we represent the parallelepiped as a set of elementary cells. The polygonal approximation of the surface can be obtained by visiting each cell and approximation of surface patches belonging to a cell. In the general case a cell can contain more than one surface patch (Fig. 2a). Using conventional enumeration of cell edges, the edges connectivity graph can be constructed. Surface patches correspond to simple cycles of this graph (Fig. 2b).

The algorithm of a cell processing follows:

1) evaluation of function values in eight cell vertices;
2) search for intersection points with the surface for edges with different function signs in edge vertices;
3) connectivity graph construction;
4) graph tracing for revealing cycles and patches construction.

The intersection point of an edge and the surface can be found by the linear interpolation of function values in vertices or by a more sophisticated root search algorithm on the edge. The procedure of the graph construction involves independent inspection of six cell faces and listing of graph branches between corresponding nodes. In the most difficult case all four edges of a face intersect the surface (Fig. 2c). The bilinear interpolation of the function on the cell face produces hyperbolic arcs on the face plane. Calculation of coordinates of the hyperbola centre provides unambigous choice of the intersection points connection by two hyperbolic arcs. To obtain the polygonized surface, the hyperbolic arcs are replaced by line segments.

The procedure of graph tracing involves:

1) search for a graph node i connected with another node j by the branch E_{ij};

2) transition to the node j with the branch E_{ij} removal;

3) transition to the node k by the branch E_{jk} with removal of latter.

Steps 2-3 are repeated until a cycle under consideration is closed and then step 1 is executed again. Additional branches can be introduced in the graph to produce quadrilateral or triangular grid. This procedure stops when the graph is empty. The algorithm produces a set of polygons approximating the initial surface. This algorithm is free of heuristics and ambiguities essential to other algorithms of this kind.

3. TRIMMING ALGORITHM

We propose an algorithm for implicit surface trimming based on the extension of existing implicit surface polygonization algorithms. The idea is to adaptively polygonize the initial surface near its intersection curves with the trimming surface and to test the obtained polygons against the trimming solid/surface. The proposed algorithm is based on the implicit surface polygonization method described in the previous section. In fact, any conventional polygonization algorithm can be extended in similar way.

The algorithm includes the following steps:

1. Define the initial surface as $f_A(x,y,z) = 0$ and the trimming solid as $f_B(x,y,z) \geq 0$. In the case of intersection with another implicit surface, it is defined as $f_B(x,y,z) = 0$.

2. Introduce a sparse spatial grid in (x,y,z) space.

3. Calculate $f_A(x,y,z)$ values at the grid points.

4. Obtain the initial triangulation for the surface $f_A = 0$ following a conventional polygonization algorithm providing correct topology with the given sparse grid (see Section 2).

5. For each triangle, calculate $f_B(x,y,z)$ values in its vertices and check the following adaptation criteria (see Fig. 3):

- Different signs of the function f_B values in the vertices: the triangle intersects the trimming surface (Fig. 3a);

- Evaluate f_B in the barycenter of the triangle. If the sign is different from the signs in the vertices, the trimming surface penetrates the triangle but all vertices are outside or inside the surface (Fig. 3b). Fig. 3c shows the worst case when the trimming surface intersects the triangle but the sign in the barycenter is the same as signs in the vertices. Possible improvements of the algorithm aimed to avoidance of such cases are discussed below.

- If the absolute value of f_B in a vertex is less than some given ε, the triangle is close to the trimming surface with the possible surface-surface intersection (Fig. 3d,e).

6. If one of the adaptation criteria is satisfied, start recursive subdivision of the triangle in four triangles by introducing new vertices in the middle of its edges. Place newly introduced vertices on the initial surface using a search in the normal direction. If a triangle adjacent to an edge is not subdivided, an additional triangle has to be inserted to prevent cracks (Fig. 4). Repeat the previous step for the four new triangles.

7. If no one of the adaptation criteria is satisfied for the current triangle or the given level of subdivision is achieved, classify the triangle against the trimming solid. Check the values of the function f_B in the vertices. Three cases are possible:

1) Three positive values. The triangle is completely inside the trimming solid and is not included in the resulting mesh.

2) Three negative values. The triangle is completely outside the trimming solid and is included in the resulting mesh.

3) Different signs. The endpoints of the intersection line segment can be found by the linear interpolation along the corresponding edges. However, more sophisticated root search along the edge can produce more precise intersection point and therefore decrease the overall number of triangles. The triangle is split along this segment and the part lying outside the trimming solid is included in the resulting mesh. In the case of the intersection with another implicit surface, the intersection line segment

is used to generate a patch of a thin stripe by stepping from each endpoint in both directions along the corresponding edge of the triangle.

Note that the proposed algorithm starts with the sparse mesh and then invokes its adaptation in the neighborhood of the intersection curves. Moreover, the function f_B is evaluated only in the vertices of the initial surface mesh but not in all 3D grid points. This helps to decrease the number of time-consuming function evaluations while providing required accuracy of the 2-manifold boundary extraction.

The polygonization algorithm based on the uniform space subdivision can result in missing features of the initial surface such as sharp edges and vertices, spikes and other small elements. As it was mentioned before, any polygonization algorithm can be applied here. The recently introduced polygonization algorithm with the mesh optimization [8] can be used to improve the quality of the initial surface mesh. The algorithm modification for adaptation to the trimming surface features can help avoid the unwanted cases such as shown in Fig. 3c.

4. APPLICATIONS

4.1 Modeling Escher's spirals

Initial algebraic surface (Fig. 1), procedural blobby model (Fig. 7), and surface of a constructive solid (Fig. 8) were trimmed using the proposed algorithm. The trimming solid for these examples was modeled using sweeping, offsetting and union operations [11] (see Fig.6). For the trimmed sphere shown in Fig. 1, top and bottom critical points of the trimming solid are placed exactly on the initial sphere with the trimming surface tangent to the sphere in the critical points. This is necessary to check the proposed adaptive polygonization criteria. Fig. 5 illustrates the difference between non-adaptive and adaptive polygonization near the top critical point on the sparse grid with 13×13×9 nodes and four recursion levels of adaptation. The proposed polygonization algorithm can be applied to any initial surface and a trimming solid defined by continuous real functions. Fig. 7 shows an adaptive polygonal mesh and a trimmed surface of a blobby object. Its adaptive polygonization on the 20×20×20 initial grid took 40 seconds on a SG Indigo^2 workstation. Its non-adaptive polygonization on the corresponding 128×128×128 grid takes about 25 minutes. The "Constructive Rind" example in Fig. 8 was produced

after “Rind” (1955) by M. Escher. The initial solid was constructed using set operations with R-functions on algebraic primitives, soft objects, and convolution solids.

4.2 Extraction of ridges on implicit surfaces

Surface creases provide important information about the shape and can be intuitively defined as curves on a surface along which the surface bends sharply. The mathematical definition of such cureves proposed in [2] is based on extrema of the principal curvatures along their curvature lines. Thus, a ridge is defined as local positive maxima of the maximal principal curvature along its associated curvature line. It was shown in [2] that ridges on an implicit surface $F(x,y,z) = 0$ are intersection curves of this surface and another implicit surface, which is locus of the extrema of the principal curvatures along their principal directions $T(x,y,z) = 0$ with

$$T(x,y,z) = \frac{\partial k_{\max}}{\partial s_{\max}} \cdot \frac{\partial k_{\min}}{\partial s_{\min}},$$

where $\frac{\partial k_{\max}}{\partial s_{\max}}$ is the first derivative of the maximal principal curvature along its associated curvature line and $\frac{\partial k_{\min}}{\partial s_{\min}}$ is the first derivative of the minimal principal curvature along its associated curvature line.

We applied the proposed trimming algorithm to extract the intersection curves of these two surfaces and to generate corresponding surface stripes for visualization purposes (see Fig. 9). As numerical calculations of finite differences approximating derivatives of F involved in the definiton of the surface T are very time-consuming, we polygonized surface F and used surface T as a trimming surface. The proposed algorithm starts with the sparse mesh and then invokes its adaptation in the neighborhood of the intersection curve. The function T defining the ridges is evaluated only in the vertices of the initial surface mesh but not in the 3D grid points. This helps to decrease the number of time-consuming function T evaluations while providing required accuracy of the intersection line extraction. Note that for the purpose of visualization, for each triangle of the initial surface polygonization we extract not only the intersection line segment, but a surface stripe by stepping from each endpoint of the segment in both directions along the corresponding triangle edge.

In the test shown in Fig. 9, we used the rounded octahedron

$$F(x,y,z) = x^4+y^4+z^4+5x^2y^2+5y^2z^2+5z^2x^2-1 = 0$$

shown in Fig. 9a. The trimming surface is shown in Fig. 9b. The initial sparse mesh has 12 × 12 × 12 nodes with four levels of recursion. The processing time for extracting ridges and generation of corresponding surface stripes (Fig. 9c) is 140 seconds. It gives approximately 70 times speed-up if compared with 2.5 hours of calculation time for the non-adaptive run with equivalent 190 × 190 × 190 resolution. Piecewise approximation of the surface stripes allows local coloring of the different type ridges (Fig. 9d) according to [2].

5. CONCLUSION

A new algorithm of polygonization of two-dimensional manifolds with boundaries is proposed. The algorithm extends conventional polygonization algorithms by including a trimming solid or surface and adapting the polygonal mesh to the 2-manifold boundary. The proposed adaptive solution significantly accelerates polygonization. Depending on the initial surface and the trimming solid/surface complexity, we obtained 20 to 70 times speed-up if compared with the non-adaptive algorithm. Applications of the algorithm include computer art, computer-aided design, solid and heterogeneous object modeling, surface-surface intersection, and ridges detection for surface analysis.

ACKNOWLEDGEMENTS

The authors would like to thank Dr. Alexander Belyaev for proposing an application example and for his help with its implementation. The comments by an anonymous reviewer helped improve the paper.

REFERENCES

[1] Adzhiev V., Kartasheva E., Kunii T., Pasko A., Schmitt B., *Cellular-functional modeling of heterogeneous objects*, 7th ACM Symposium on Solid Modeling and Applications, ACM Press (2002) 192-203.

[2] Belyaev A., Pasko A., Kunii T., *Ridges and ravines on implicit surfaces*, Computer Graphics International '98 (June 22-26 1998, Hannover, Germany), IEEE Computer Society (1998) pp.530-535.

[3] Bloomenthal J., Ferguson K., *Polygonization of non-manifold implicit surfaces*, SIGGRAPH'95, Computer Graphics Proceedings (1995) 309-316.

[4] Bloomenthal J. et al., *Introduction to Implicit Surfaces*, Morgan Kaufmann, 1997.

[5] Bowyer A., *SVLIS Set-Theoretic Kernel Modeller. Introduction and User Manual*, Information Geometers, Winchester, UK, 1994.

[6] Hosaka M., *Modeling of Curves and Surfaces in CAD/CAM*, Springer-Verlag, Berlin, 1992.

[7] Kunii T., *Valid computational shape modeling: design and implementation*, International Journal of Shape Modeling 5 (1999), 123-133.

[8] Y. Ohtake, A. Belyaev, A. Pasko, *Dynamic meshes for accurate polygonization of implicit surfaces with sharp features*, Shape Modeling International 2001, IEEE Computer Society (2001) 74-81.

[9] Pasko A., Pilyugin V., Pokrovskiy V., *Geometric modeling in the analysis of trivariate functions*, Communications of Joint Institute of Nuclear Research, P10-86-310, Dubna, USSR, 1986 (in Russian).

[10] Pasko A., Pilyugin V., Pokrovskiy V., *Geometric modeling in the analysis of trivariate functions*, Computers and Graphics 12 (1988) 457-465.

[11] Pasko A., Adzhiev V., Sourin A., Savchenko V., *Function representation in geometric modeling: concepts, implementation and applications*, The Visual Computer 11 (1995) 429-446.

[12] Rossignac J., *CSG formulations for identifying and for trimming faces of CSG models*, CSG 96 Set-theoretic Solid Modeling: Techniques and Applications, Information Geometers, UK (1996) 1-14.

[13] Rvachev V.L., *Theory of R-functions and Some Applications*, Naukova Dumka, Kiev, 1987 (in Russian).

[14] Wyvill B., van Overveld K., *Polygonization of implicit surfaces with Constructive Solid Geometry*, International Journal of Shape Modeling, 2 (1997) 257-273.

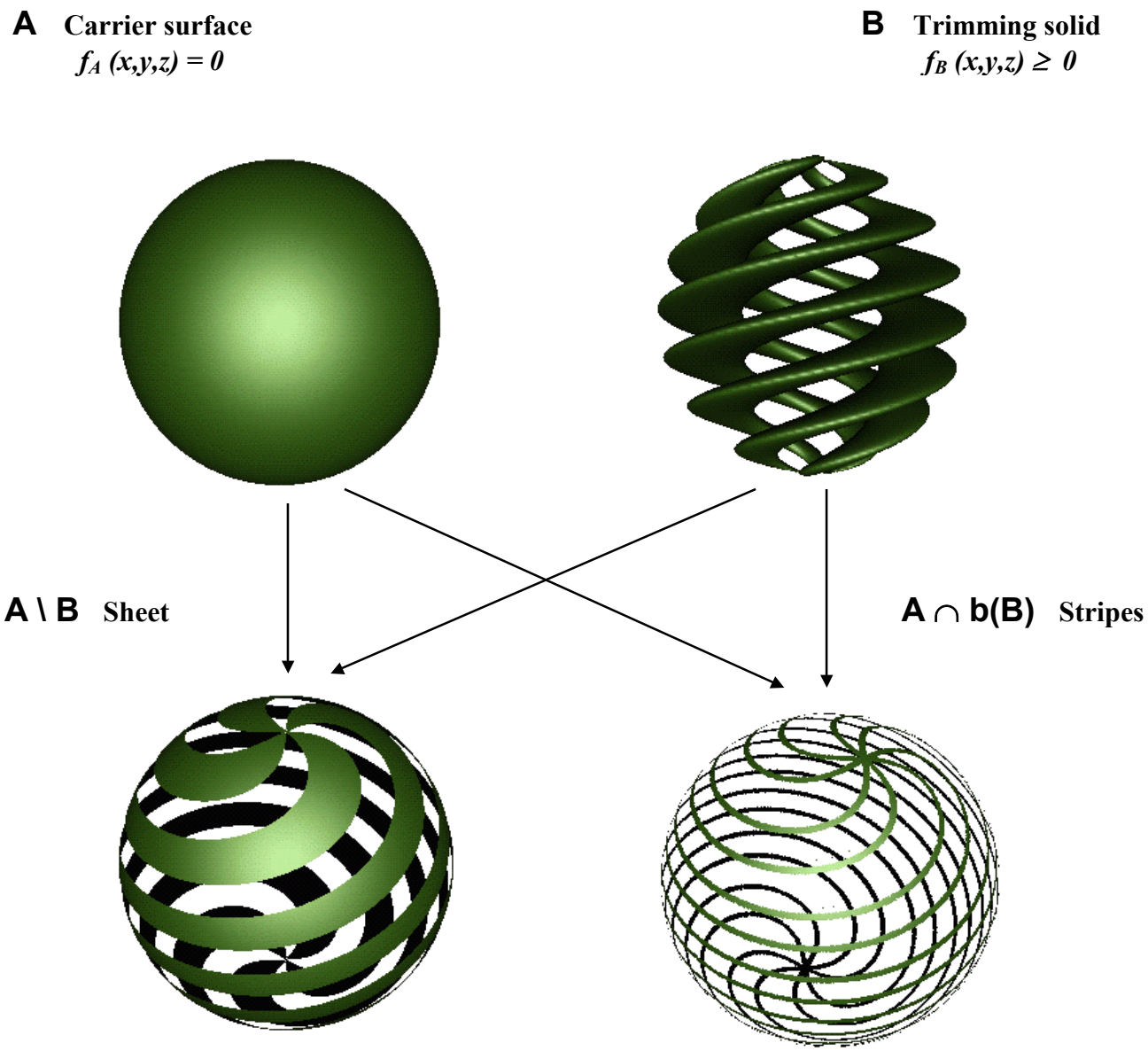
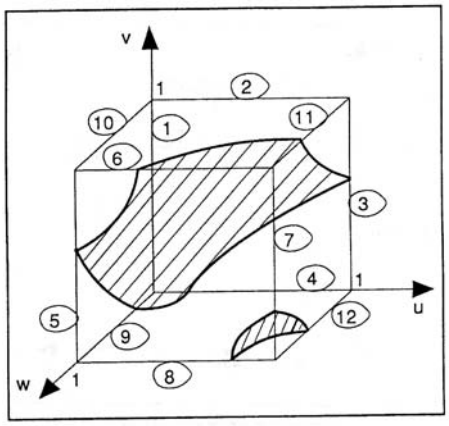
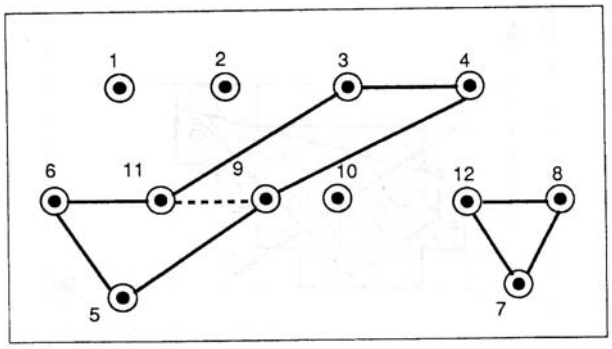


Figure 1. “Sphere Spirals”. Modeling 2-manifolds with boundary:

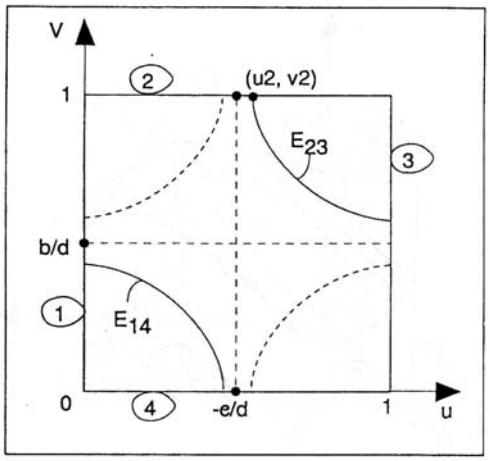
1) surface sheet (lower left) as a set-theoretic difference A\B between an initial surface A and a trimming solid B;
2) surface stripes (lower right) as an offset of the intersection curves A ∩ b(B) between the initial surface A and the trimming surface b(B).



(a)



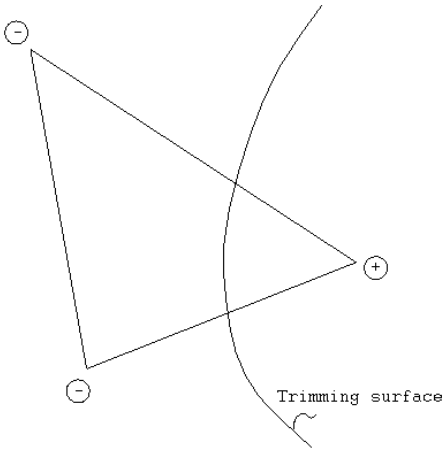
(b)



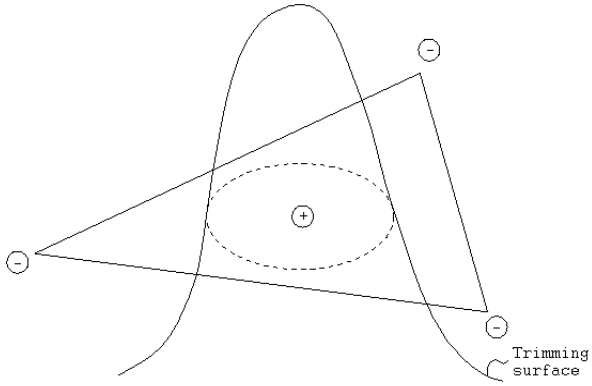
(c)

Figure 2. Basic polgonization algorithm:

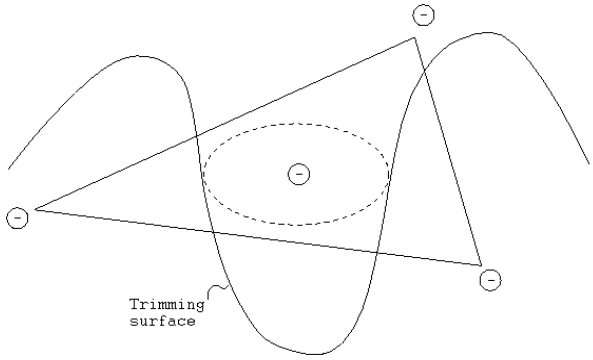
a) example of surface patches inside a cell with hyperbolic arc boundaries on the cell faces;
b) cycles in the connectivity graph corresponding to surface patches;
c) case of four intersection points of cell face edges with the isosurface.



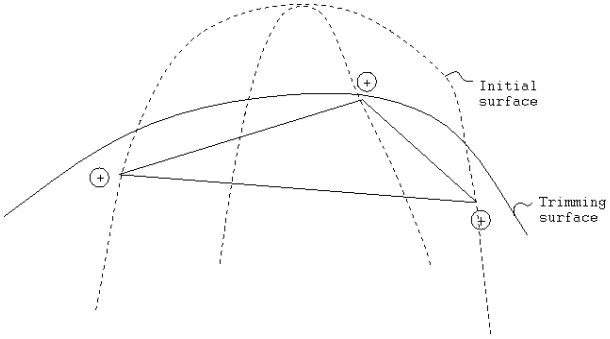
a) One or two vertices inside the trimming solid



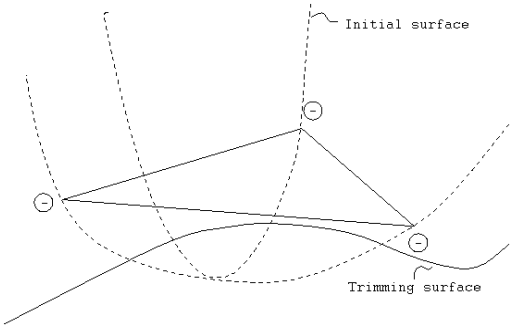
b) Center inside the trimming solid



c) Intersection is not detected



d) Triangle inside and near the trimming surface



e) Triangle outside and near the trimming surface

Figure 3. Adaptation criteria for a polygon of the initial surface mesh.

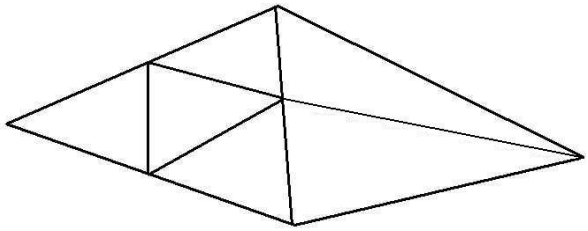
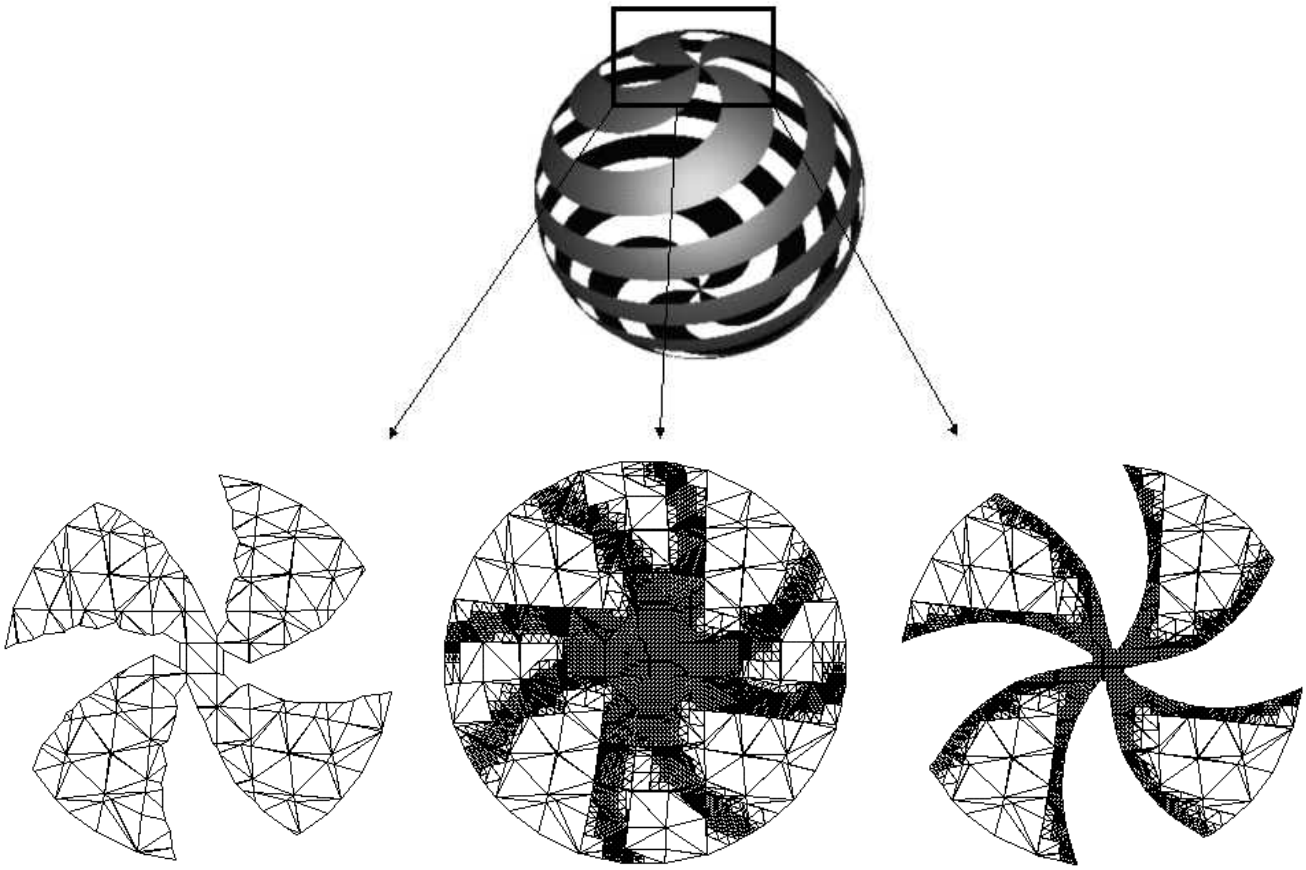


Figure 4. Avoidance of cracks: subdivision (left triangle) and splitting (right triangle).



(a) (b) (c)

Figure 5. Adaptive trimming of a sphere (top part) by a complex solid:

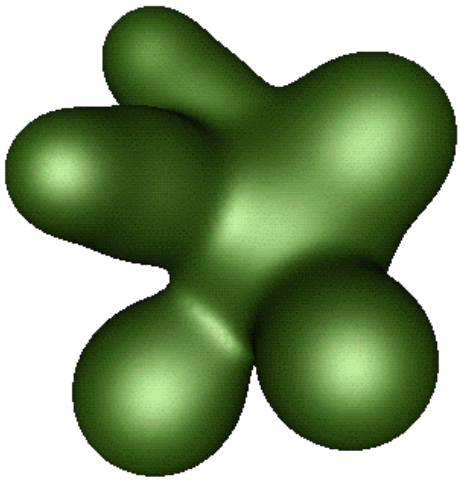
(a) result of nonadaptive trimming with polygonization on a sparse grid 13×13×9;

(b) mesh obtained by the adaptive polygonization (4 levels of subdivision are used for illustration);

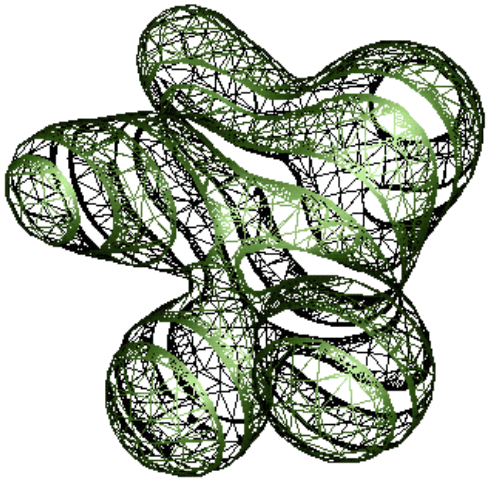
(c) result of adaptive trimming.

```
F(x,y,z) = f(x,y,z,phi1) |          ← R-union
           f(x,y,z,phi2) |
           f(x,y,z,phi3);

f(x,y,z,phi)
{
     z2 = z*z;
     R = sqrt(100.-z2);
     x0 = R*cos(0.5*z+phi);
     y0 = R*sin(0.5*z+phi);
     xt = x-x0;
     yt = y-y0;
     r = 2.- z2*0.02;
     ftrim = r*r - xt*xt -yt*yt;    ← sweep
     offset = 10 - z2*0.1;
     f = ftrim + offset;            ← offset

}
```

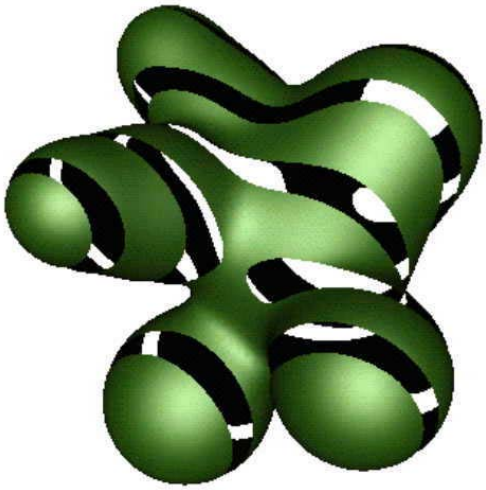
Figure 6. Definition of the trimming solid as $F(x,y,z) \geq 0$ for $x, y, z \in [-10, 10]$. The pseudocode shows the procedural definition of the function *f(x,y,z,phi)* for one of three spiral solid components.



(a)

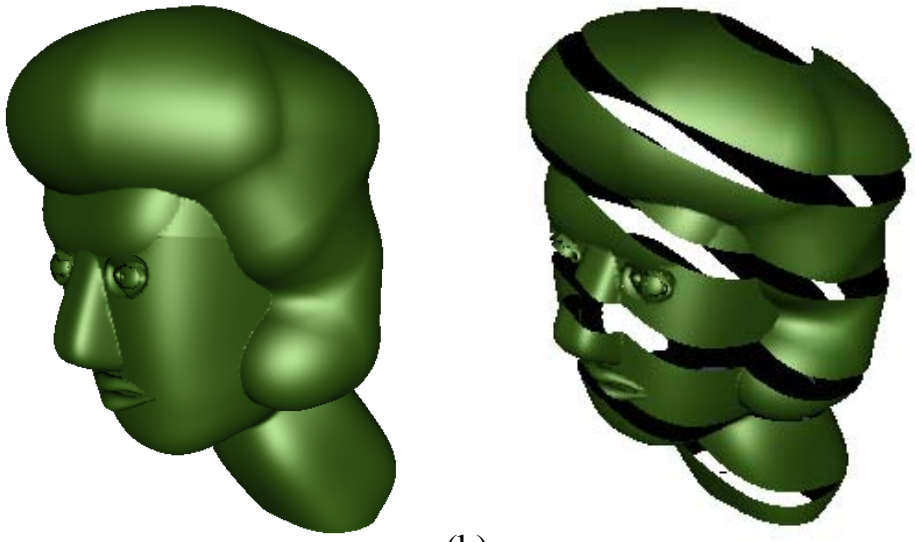


(b)



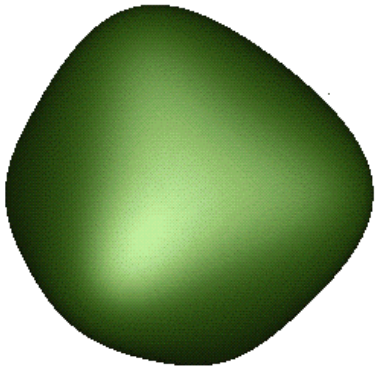
(c)

Figure 7. ”Blobby Spiral”: a) initial blobby object
b) adaptive polygonal mesh
c) surface of the blobby object trimmed by a spiral solid

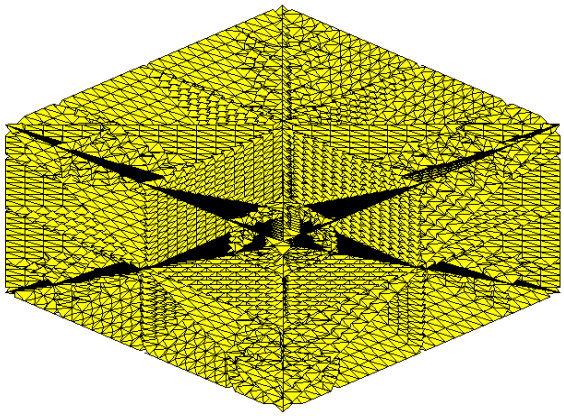


(a) (b)

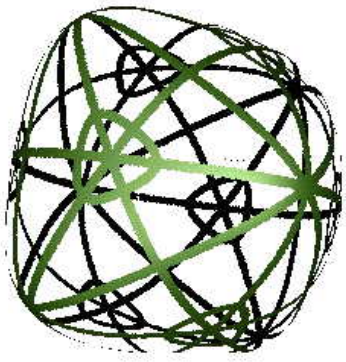
Figure 8. “Constructive Rind”: an initial constructive solid (a) and its surface trimmed by the spiral solid (b).



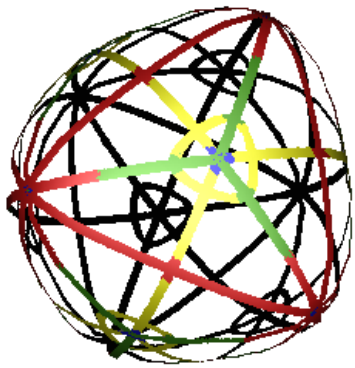
(a)



(b)



(c)



(d)

Figure 9. Extraction of ridges on an implicit surface:

a) initial rounded octahedron;
b) trimming surface defining the ridges;
c) surface stripes representing the intersection curves;
d) local coloring of different type ridges.