Geometric ambiguities in boundary representations

Christoph M Hoffmann and John E Hopcroft*

Boundary representations are usually separated into two components, a topological component and a geometric component. The conditions necessary to ensure that the topological component is unambiguous are well understood. However, in algebraic modelling systems (as opposed to polyhedral modelling systems), an unambiguous topological structure can have noncongruent geometric interpretations with identical vertex coordinates, face equations and edge descriptions using the usual representation of edges as the intersection of adjacent faces. This means that a naive conversion of a CSG tree that unambiguously describes a solid may well lead to ambiguous boundary representations unless additional information is retained in the boundary representation. This paper examines the source of these ambiguities.

geometry, boundary representations, ambiguities

Geometric modelling must be based on a flexible, yet unambiguous scheme for representing objects and sets of objects. Among the schemes used in current modelling systems is the familiar boundary representation method, in which the surface of an object is described, e.g. Ansaldi *et al*[1] and Baumgart[2]. This method has been used extensively in computer graphics, yet it is by no means a simple task to lay down a complete set of precise criteria permitting formal verification that a given boundary representation scheme is unambiguous and informationally complete in the sense of Requicha and Voelcker[3].

A number of papers[4–7] have addressed the intrinsic problems of boundary representations, and the field has succeeded in giving precise conditions under which boundary representations of polyhedral objects are correct and unambiguous. Here one of the major issues has been to understand the topological properties of surface descriptions[5]. These topological properties generalize to the curved surface domain. However, geometric problems arise that can lead to ambiguities even when the topology of the surface has been unambiguously specified. These difficulties relate to the following phenomena:

- A curved surface may have singularities. Singular points may be isolated or constitute curves of singularity. For example, the vertex of a cone is an isolated singular point, and the self-intersection curves of certain canal surfaces are curves of singularity, as shown in Figure 1.
- The intersection curve of two faces may be closed, and no intrinsic direction of the edge can be defined naturally. For example, consider the two intersecting spheres of Figure 2. The curve of intersection is a circle that has no intrinsic direction, even with the convention of outward facing normals.
- The intersection curve of two curved faces may have singular points, even though the intersecting surfaces have no singularity. For example, the orthogonal, axial intersection of two circular cylinders of equal radius has two intersecting ellipses as components, and the points of intersection are singular curve points. See also Figure 3.

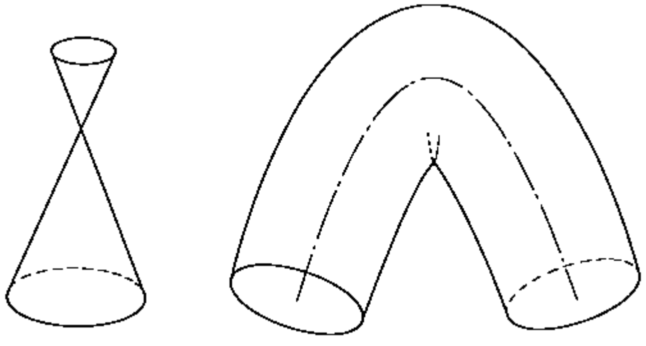

Figure 1. Singular points and curves

It is demonstrated in this paper that even with a number of reasonable and intuitive restrictions, annotations, and conventions, a traditional boundary representation may remain ambiguous. This ambiguity is not due to a lack of topological information. To the contrary, the topology is defined unambiguously by the results of Weiler[5]. Rather, the source of ambiguity lies in the representation of edges as the intersection of adjacent faces.

Representing edges as the intersection of the adjacent faces is perfectly adequate in the polyhedral domain and does not cause ambiguities. In the curved surface domain, this is no longer true. Unfortunately, the intersection curves of curved surfaces are difficult to represent in any other way since many do not possess a rational parametric form. For example, the intersection of two circular cylinders is in general an irreducible degree 4 curve that is not rationally parameterizable. When representing the curve, a choice

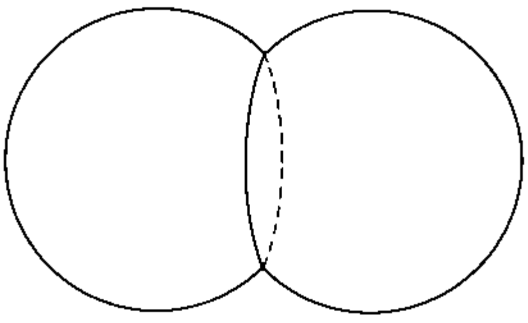

Figure 2. Closed curve without intrinsic direction

Computer Science Department, Purdue University, West Lafayette, IN 47907, USA

*Computer Science Department, Cornell University, Ithaca, NY 14853, USA

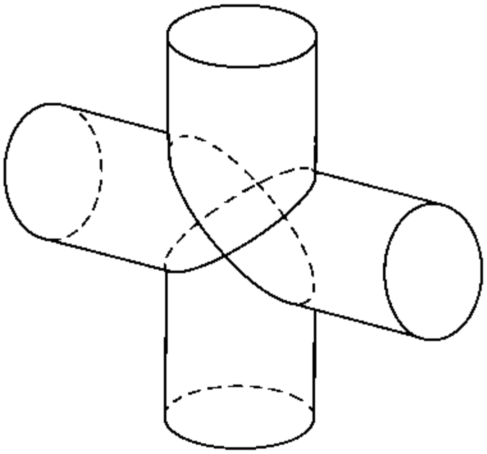

Figure 3. Intersection curve with two components and two singular points

must therefore be made between approximating it to a satisfactory tolerance by a class of rationally parameterizable curves, or finding a nonrational parameterization. Alternatively, the source of ambiguities should be sought and how to circumvent it identified.

Curved surfaces and space curves may be closed, and, furthermore, they may contain singularities. Both situations are unparalleled in the polyhedral domain and are responsible for the geometric difficulties that must be faced. That singularities on edges may cause difficulties has been noted previously[8]. However, what has not been recognized is that there are different, noncongruent interpretations of curved surface boundary representations that are consistent with an unambiguously defined surface topology. Thus even global processing of the boundary representation is insufficient to resolve the ambiguity and additional information must be incorporated into the model. Hence work such as that done by Weiler[6] needs to be complemented with the authors' findings.

The work reported here was done in an ongoing project aimed at extending the geometric coverage of present day modelling systems to curved algebraic surfaces of any degree. It is demonstrated that already the basic question of how to represent an edge becomes an issue in such an undertaking. The paper is structured as follows: in the second section, the usual conventions and properties of boundary representations are reviewed. The third section shows why it is necessary to place additional vertices on certain edges even though from a topological point of view this is unnecessary as maintained[4,7]. In the fourth section the ambiguous object is developed by first giving an unambiguous CSG description clarifying the intended shape. Then the corresponding boundary representation is given and shown to be ambiguous. Thus, straightforward CSG to boundary representation conversion algorithms have to account for the possibility of introducing ambiguities. Finally, the consequences of these findings are discussed, as well as how to cope with them.

PRELIMINARIES

In a boundary representation, the surface of an object is specified given the following information:

- A finite set of vertices, specified by Cartesian coordinates in Euclidean space.
- A set of edges, where each edge is incident to two vertices as specified.
- A set of faces, where each face is bounded by (say) a single cycle of edges.

In addition, edge and face adjacency information is provided.

Typically, an edge is specified by the intersection of two faces, one on the left, the other on the right of the edge. Here left and right are defined relative to the edge direction as seen from the exterior of the object. By convention, the edge (u, v) is considered directed from vertex u to vertex v. Moreover, the cyclical order of incident edges is specified at each vertex, say in clockwise order. Additional conventions are understood, e.g. faces do not self-intersect, two distinct faces intersect at most in edges, etc.

For finite, polyhedral objects it is simple to define edges and faces geometrically: an edge is uniquely specified as the line segment connecting two vertices, and the face plane may be specified by its equation or by three distinct noncollinear face vertices.

In contrast, representing curved faces and edges is more difficult. It is assumed that the surface of which the face is a patch is specified by an implicit algebraic equation. This equation has been adjusted so that the convention of outward-pointing normals is obeyed: for each interior face point p on the surface f, the vector $\nabla f(p)$ locally points to the outside of the object -- an edge is represented by the intersection of two adjacent faces. When the adjacent faces belong to the same surface, an auxiliary surface is used whose intersection fixes the edge.

We assume that the boundary representation describes a solid object. This is not usually made precise, but Requicha[8] contains ways for stating this requirement rigorously. However, there is no accepted mathematical definition: consider the object in Figure 4. Many, but not all, solid modellers based on boundary representation do not consider this object legitimate because one of the edges is adjacent to four faces. For example, in Romulus this object is disallowed[9], and so Romulus is not closed under regularized Boolean operations. The reason Romulus forbids this object seems to be that all object surfaces should be readily recognized manifolds.

In the modelling system by Paoluzzi *et al*[10], the object of Figure 4 is allowed, but the offensive edge is internally represented as two different edges that happen to coincide geometrically yet are topologically distinct. This ensures that all objects have manifold surfaces, although this would not be apparent visually. Of the two possible topologies, shown in exaggeration in Figure 5, the system chooses the one that leads to a surface triangulation with triangles of

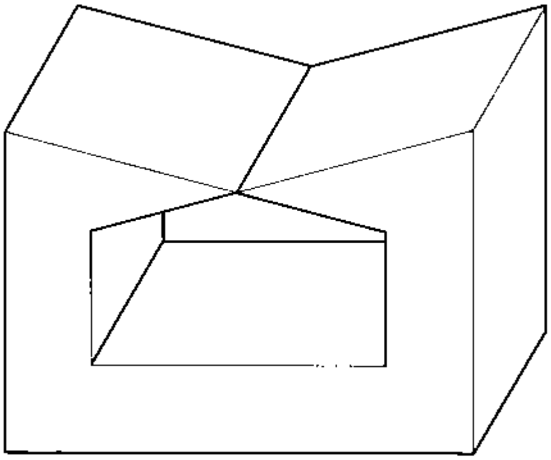

Figure 4. Controversial polyhedral object

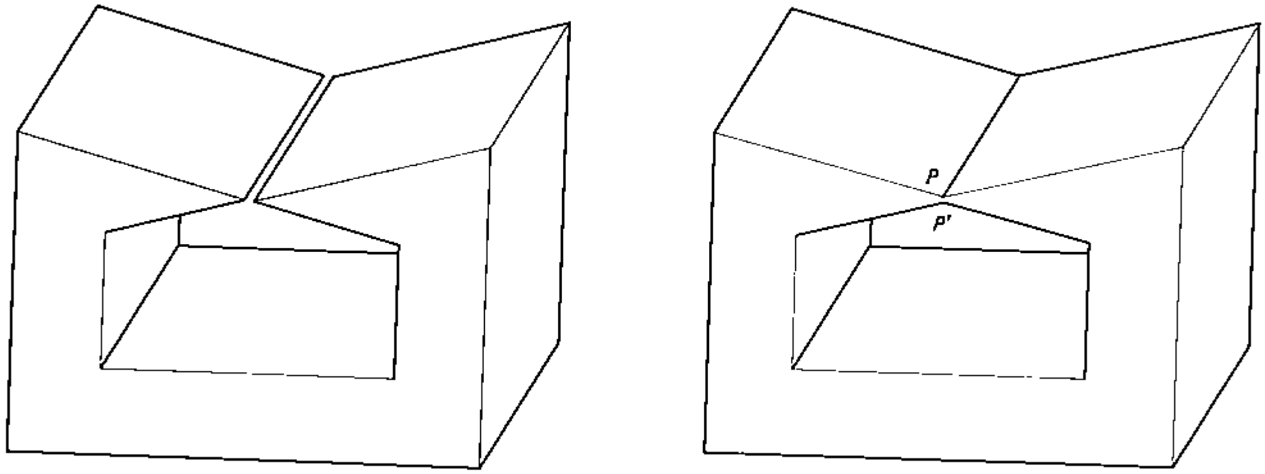

Figure 5. Two possible topologies for the coincident edges

nonzero area. Thus, the left topology would be chosen, for in the right topology every triangulation of the front face must include a triangle two of whose vertices are the (coincident) vertices p and p'. This triangle would enclose geometrically a zero area. Paoluzzi's modelling system is closed under regularized Boolean operations, but objects such as the one shown here necessitate fairly intricate steps in the implementation.

In the GDP modelling system[11], the object of Figure 4 is also legitimate but the edge is represented only once, although this fact is not clearly stated in the paper. The object's surface is therefore not a manifold. It is our experience that the implementation of the modeller becomes much more straightforward when the manifold restriction is dropped.

For the purposes of this paper, this controversy is avoided and solid objects whose surfaces are clearly 2 manifold without having to resort to embedding arguments are used.

A point p is singular on the surface $f = 0$ if the gradient $\nabla f(p)$ is zero. Intuitively this means that at a singular point the position of the tangent plane to the surface is not defined. Likewise, a point p on an algebraic curve is singular if the tangent to the curve at p is not defined. Points which are not singular are said to be regular. On algebraic surfaces, the singular points cannot fill up more than certain individual curves and points[12]. Clearly, faces with singular points are needed in geometric modelling. For example, the vertex of a cone is a singular point that must be modelled in many situations. Let us contain the difficulties introduced by singularities as follows: we require that all singular points of a face be vertices in the boundary description, and that, moreover, all interior edge points are regular, i.e. possess a unique tangent. Hence, the edge (u, v) of Figure 6 is illegitimate because it contains in its interior the singular curve points p and p'.

EDGE CYCLES AS AGAINST LEFT AND RIGHT FACE SPECIFICATIONS

Consider modelling the intersection of two solid spheres with surfaces S_1 and S_2, as shown in Figure 7. The result is a lens shape with an edge $S_1 \cap S_2$, on which we arbitrarily place a vertex u as shown. The edge may then be given as (u, u). Note that closed edges are perfectly legitimate from the point of view of Euler operations[4,7].

In a boundary description of this object, specify that the face on S_1 is to the right of the edge (u, u), and that the face on S_2 is to the left, as seen from the outside of the solid. Then the edge (u, u) must be oriented by the vector drawn at u.

Now the edge $S_1 \cap S_2$ does not have an intrinsic direction. If there were no additional information as part of the boundary representation, then (u, u) could have been oriented by a vector in the opposite direction. In that case, left and right are seen differently, and the same boundary representation now describes the object shown in Figure 8, i.e. the union of the two spheres.

This simple example shows that in the curved surface domain boundary descriptions somehow must specify edge orientation in addition to describing where in space the edge is situated. A simple method is to subdivide the edge by introducing additional vertices. A minimum of three vertices is required to encode the intended edge direction. Such a representation is shown in Figure 9 and no longer is ambiguous. This convention also serves to simplify graph algorithms that operate on the edge graph, e.g. when rendering objects. Note, however, that an edge, say (u, v), must be traversed before it can be decided whether the initial direction of traversal was the correct one. It is shown next that this convention does not solve the problem of unambiguously fixing an edge in space, in general.

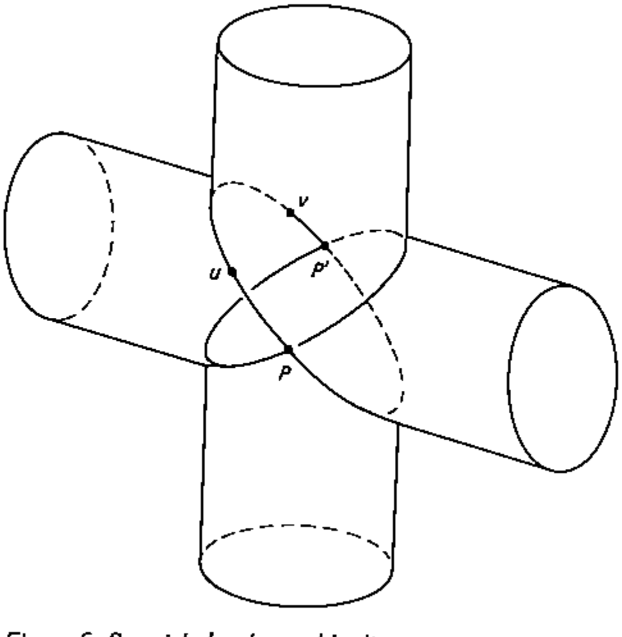

Figure 6. Requicha's edge ambiguity

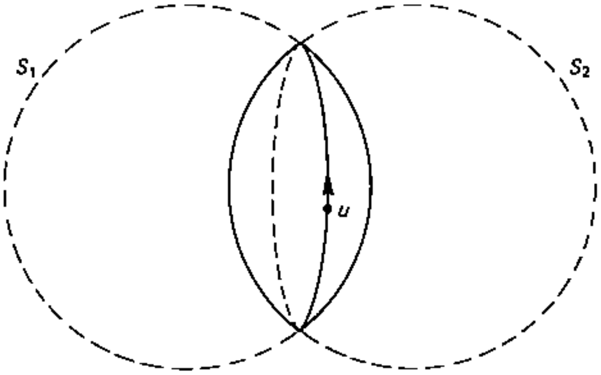

Figure 7. Intended edge orientation

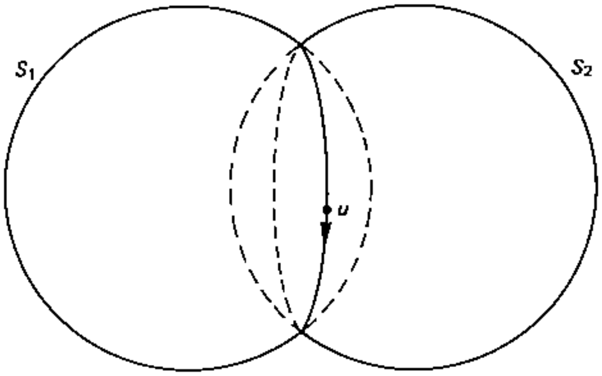

Figure 8. Result of inverting edge orientation

SINGULARITIES

It has been pointed out by Requicha[8] that singular points in the interior of an edge cause uncertainty when traversing the edge. Requicha is referring to nodal singularities, that is, to points where two different branches of the curve intersect. His example is shown in Figure 6. When traversing the edge e from u to v, at the singular points p and p' it cannot be decided locally, in which direction to continue with the traversal. Note that this type of singularity arises with very simple intersecting surfaces, but it can be handled by requiring that there are no singular points on the interior of faces or edges.

It would be nice if isolating singularities at vertices were all that were needed. This is not so, and it is now shown that in the presence of singular vertices, ambiguous boundary descriptions exist that satisfy the following restrictions:

- Every edge cycle consists of at least three edges and every vertex is incident to at least three edges.
- All interior points of an edge are regular points.
- All interior points of a face are regular points.

Moreover, the topology of the object is shown to be unique by the results of Weiler[5].

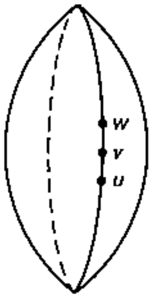

Figure 9. Three vertices to specify edge orientation

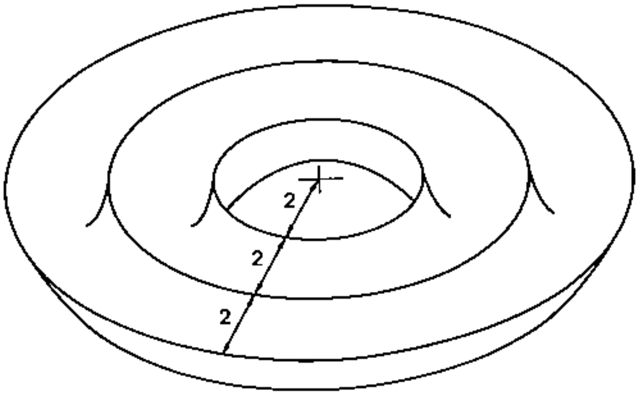

Figure 10. $((T_1 \cap H) - T_2) - T_3$

First, the object is constructed unambiguously using constructive solid geometry. Thereafter, its boundary representation is given and it is shown that this representation has a second interpretation, also defining a different solid. In the CSG definition, each primitive volume is specified by an implicit algebraic equation $f = 0$ and is the closure of the set of all points p for which $f(p) < 0$. Thus we deal with regular sets. The construction gives a step-by-step elaboration of how the resulting shape is structured.

For the construction, the standard set of primitives[3] is used, augmented by the cylinder $C : y^3 + z^3 - 6yz = 0$. Although this cylinder itself is not a manufacturable object, a final object can be constructed from it that can be manufactured without difficulty. The question of whether surfaces such as C ought to be included in solid modelling systems is commented on later in this paper.

The object has quartic faces that are toroidal. First, the object of Figure 10 is built by subtracting tori T_2 and T_3 from the halved torus $T_1 \cap H$, where H is the half space $y = 0$. The corresponding CSG expression is $((T_1 \cap H) - T_2) - T_3$. The tori dimensions are shown in Figure 10. Next, take the cylinder C whose cross cut is the Cartesian Leaf. The cylinder is shown in Figure 11. The positive surface side, that is, the side pointing to the outside of C has been shaded, and the gradients are directed as drawn. Consequently, the intersection $(((T_1 \cap H) - T_2) - T_3) \cap C$ is the object shown in Figure 12. A boundary description of this object is now given. This boundary description could be the result of a conversion algorithm translating CSG trees to boundary representations, akin to the one described in Requicha and Voelcker[13], or the description could have been constructed directly from Figure 12 by an unwary designer.

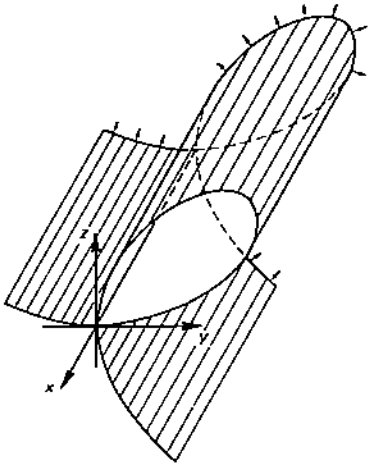

Figure 11. Cylinder $y^3 + z^3 - 6yz = 0$

In the description, it is assumed that the edges are oriented as specified by the vertex pair written. So, edge e_1 is oriented from vertex v_1 to vertex v_3. The opposite direction is indicated by negating the edge symbol, e.g. $-e_1$ denotes the edge e_1 in opposite direction.

Vertices:

v_1: (-6,0,0)
v_2: (-4,0,0)
v_3: (-2,0,0)
v_4: (2,0,0)
v_5: (4,0,0)
v_6: (6,0,0)

Face equations:

$$a: (x^2+y^2+z^2-4)^2+32(z^2-x^2-y^2-4)+256=0$$
$$b: -(x^2+y^2+z^2-1)^2-18(z^2-x^2-y^2-1)-81=0$$
$$c: -(x^2+y^2+z^2-1)^2-50(z^2-x^2-y^2-1)-625=0$$
$$d: y^3+z^3-6yz=0$$
$$e: y^3+z^3-6yz=0$$

Edges with left and right adjacent face:

$e_1: (v_1, v_3)$	(a, d)
$e_2: (v_3, v_2)$	(b, d)
$e_3: (v_2, v_1)$	(c, d)
$e_4: (v_6, v_4)$	(e, a)
$e_5: (v_4, v_5)$	(e, b)
$e_6: (v_5, v_6)$	(e, c)
$e_7: (v_1, v_6)$	(c, a)
$e_8: (v_2, v_5)$	(b, c)
$e_9: (v_3, v_4)$	(a, b)

Edge cycles clockwise about each face:

$a: (-e_1, e_7, e_4, -e_9)$
$b: (-e_2, e_9, e_5, -e_8)$
$c: (-e_3, e_8, e_6, -e_7)$
$d: (e_1, e_2, e_3)$
$e: (-e_4, -e_6, -e_5)$

Incident edges clockwise about each vertex:

$v_1: (e_1, -e_3, e_7)$
$v_2: (e_3, -e_2, e_8)$
$v_3: (e_2, -e_1, e_9)$
$v_4: (e_5, -e_9, -e_4)$
$v_5: (e_6, -e_8, -e_5)$
$v_6: (e_4, -e_7, -e_6)$

Note that every interpretation of this description has the same, fixed topology[5,6].

Since the vertices are singular points on the intersection curves of the surfaces, it is possible to interpret the edge positions differently by following a different branch of the intersection curve of C with the torus. The complete intersection curve of the torus T_1 with C is shown in Figure 13. Four directions may be followed at each vertex, but only two of them are consistent with the convention of outward pointing normals and the topology of the boundary description. For global topological consistency, the direction choices must agree at all vertices, so there can be only two correct interpretations of the above boundary description. The second interpretation is shown in Figure 14. It is easy to verify its consistency with the topology of the boundary description above. Not only are the two interpretations in different position, they are also not congruent. Due to the asymmetry of the two surface branches for the faces d and e, the two interpretations have different volume.

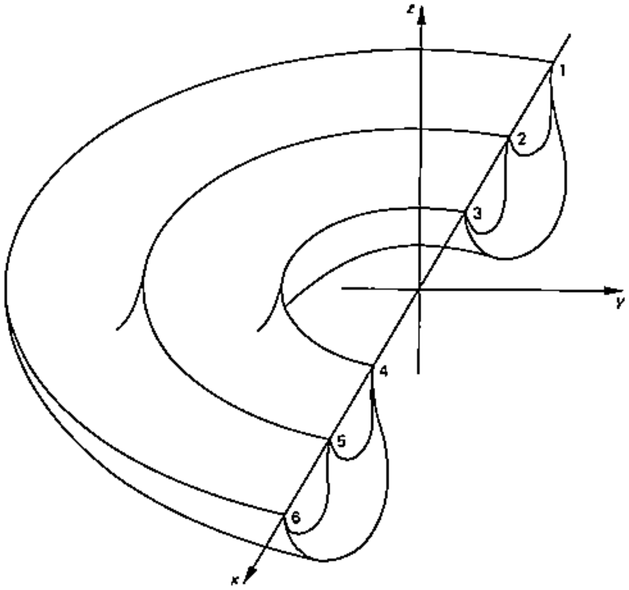

Figure 12. $((T_1 \cap H) - T_2) - T_3) \cap C$

Recall that the construction of the object involves the self-intersecting surface C. Other surfaces could have been chosen, but self-intersection seems to be necessary for obtaining global ambiguity. The next section discusses whether such surfaces should be excluded from modellers, or if instead, adequate edge data structures that resolve such ambiguities should be developed.

CONCLUSIONS

It makes sense to extend CSG type modellers by using as primitive shapes the point sets defined by $\{(x, y, z) \mid f(x, y, z) \leqslant 0\}$, where $f = 0$ is the implicit equation of an irreducible algebraic surface, i.e. f is an irreducible polynomial. For quadric surfaces this has already been

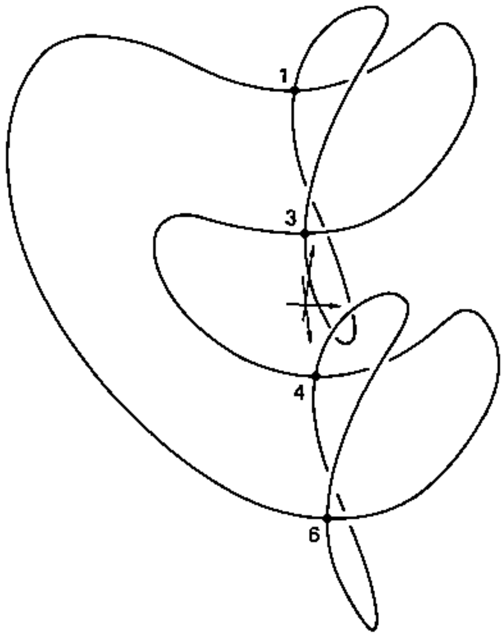

Figure 13. Intersection curve (C, T_1)

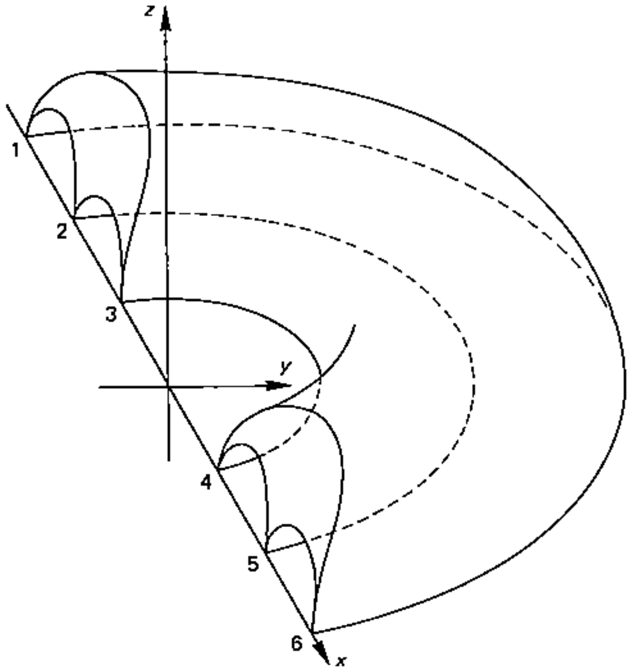

Figure 14. Second interpretation of the boundary description

proposed[14,15]. Manufacturability can be a criterion for the final objects modelled, but should not be imposed as a limitation on intermediate modelling steps.

When extending modellers in this way, reducible surfaces ought to be excluded since their equation $f(x, y, z) = 0$ can be written as $g(x, y, z)\, h(x, y, z) = 0$. Here g and h are two other polynomials, and so f is the union of two simpler surfaces. In addition, one may require that every modelled object, obtained from these primitives by the usual regularized Boolean operations, be finite, although this need not be the case for the primitives themselves.

If the modeller is based on boundary representation, we allow similarly faces that are suitable patches of (irreducible) algebraic surfaces and edges that are algebraic curves on them. Assuming, rather conservatively, that any singularity should be confined to vertices, the demonstrated ambiguity becomes a matter of concern for modellers based on boundary representation, but it also needs to be considered for conversion algorithms from CSG to boundary representation. A purely CSG-based modeller does not seem to have these difficulties.

The question, then, must be whether there are natural restrictions that eliminate ambiguities of the kind we have demonstrated, short of avoiding boundary representation altogether. These restrictions might be one of the following:

- Allow no singularities at all.
- Exclude surfaces that are self-intersecting along a curve.
- Find a suitable representation that fixes edges unambiguously.

Clearly disallowing all singularities is unreasonable, for singularities such as the vertex of a cone are needed. But should we exclude surfaces that self-intersect along a curve? As long as the resulting face does not contain a segment of the self-intersection curve, we believe that such surfaces should not be excluded, as the class includes some very attractive surfaces: for example, the Steiner patch[16] has many desirable properties for solid modelling and is self-intersecting along three lines. The Steiner patch has the following advantages: it is a quartic surface, so the implicit form has moderate degree. It can be parameterized and the parametric representation has degree 2. Finally, it is easy to convert the parametric into the implicit form and *vice versa.* So patches of the Steiner surface are especially easy to work with.

Hence, the third alternative should be explored and edge representations should be investigated with more care. Such an investigation divides into two parts: how do we represent the underlying algebraic curve, and how do we identify a proper segment constituting the edge.

An algebraic space curve is represented in general as the complete intersection of two or more algebraic surfaces. In the case of solid modelling, two surfaces suffice, say $f = 0$ and $g = 0$, so the curve, of which the edge is a segment, can be represented by the pair (f, g). Note, however, that the pair $(f, uf + vg)$ represents the same curve as (f, g), with u and v constants, $v \neq 0$. So we may select among the difference surfaces $uf + vg$ one which simplifies modelling algorithms. For quadric f and g the most convenient choice is a ruled quadric. Given f and g, such surfaces always exist in the set $uf + vg$ and can be found efficiently[14]. For higher degree surfaces, the choice is not obvious and requires further investigation.

Now consider a singular point p of the curve (f, g). At such a point, there are one or more distinct branches of the curve. In general, it is not possible to isolate one of the branches by a better choice of g or by using additional surfaces to intersect with. If singular points are confined to vertices, then the edge segment is homeomorphic to a line, and a possible unambiguous identification of the segment is provided by specifying an interior point for each edge. This is the method suggested by Requicha[8]. The major drawback is its inconvenience for rendering algorithms, for the scheme requires walking the edge, beginning with the interior point, so as to locate the correct branch and direction at each vertex. Instead, it is suggested that the edge should be annotated with directional information at each vertex.

In the simplest case, the desired branch is identified by the tangent director vector at each vertex. This method suffices for all nodal singularities, that is, for singularities at which locally the curve consists of a number of continuous branches that intersect transversally. The singularities of Figures 3, 6 and 10 are all of this type, and the method suffices for most situations the authors have encountered.

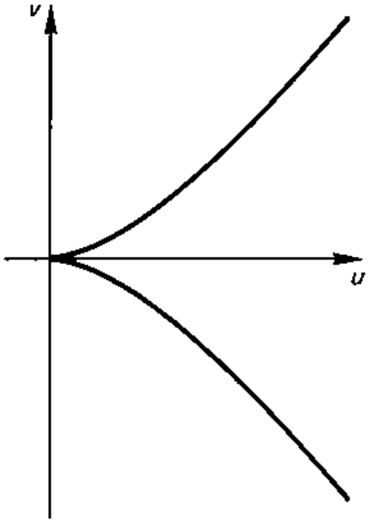

Figure 15. Cuspidal singularity $u^3 - v^2 = 0$

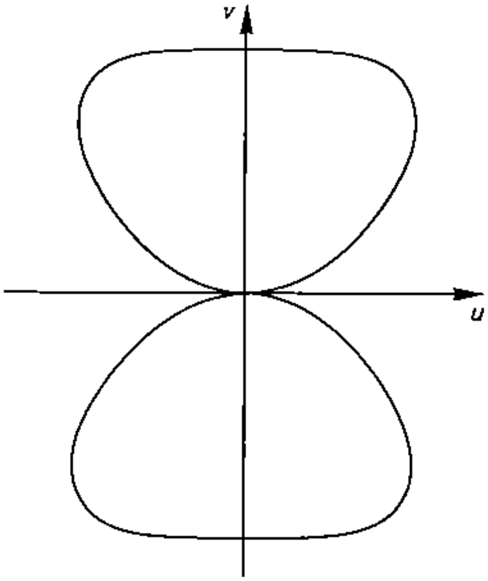

Figure 16. Tacnode $v^2 - u^4 - v^4 = 0$

More difficult singularities are cusps, at which the branch is discontinuous (see Figure 15) and tacnodes at which branches intersects tangentially (see Figure 16). Intuitively, such singularities require information about higher order derivatives best provided by birational transformations for resolving the curve structure at the singularity[17]. These transformations can be found algorithmically, but the procedure is complex and cannot be portrayed simply. Note that the singularities shown in Figures 15 and 16 arise for surfaces at fairly low degree: the cusp of figure 15 is the curve of intersection of $y^2 - x^3 + z = 0$ with the plane $z = 0$. The tacnode of Figure 16 is the intersection of the parabolic cylinder $z - y^2 = 0$ with the surface $z - x^4 - y^4 = 0$, approximately a figure of revolution of the quartic parabola $z - y^4 = 0$. In each case, the intersecting surfaces have no singularities at all.

A large segment of the community restricts itself to parametric surfaces of specific types, e.g. bicubic patches, rational B-splines, etc.[18]. These surfaces are a proper subset of all algebraic surfaces; for example, bicubic patches lie on surfaces of degree at most 18. Does modelling with such surface patches avoid problems arising from singularities? The answer here very much depends on the types of modelling operations supported. If only free-form design is done, then the patches all are fitted to each other explicitly by the user. In that case, each patch has a fixed rectangular or triangular domain and edges between patches are unambiguous. When two such objects are intersected, however, then this nice surface structure can no longer be preserved. In that case, the curves in which two patches intersect are the image of planar algebraic curves that show how the domains intersect. These curves contain in general singular points and require special techniques for analysis[19]. Whether global ambiguities can arise in consequence is not known.

ACKNOWLEDGEMENTS

The work for this paper was supported in part by National Science Foundation Grants ECS 83-12096, DCR 85-02568, DCR 85-12443 and the Office of Naval Research under contract N00014-86-K-0465.

REFERENCES

1 Ansaldi, S, DeFloriani, L and Falcienda, B 'Geometric modelling of solid objects by using face adjacency graph representation' *Comp. Graphics* Vol 19 No 3 (1985) pp 131–139

2 Baumgart, B 'Winged edge polyhedron representation' *Stanford AI Report CS-320* (1972)

3 Requicha, A and Voelcker, H 'Solid modelling: current status and research directions' *IEEE Comp. Graphics and Applic.* Vol 3 (1983) pp 25–37

4 Eastman, C and Weiler, K 'Geometric modelling using the Euler operators' *1st Annual Conf. Comp. Graphics in CAD/CAM* MIT (1979) pp 248–259

5 Weiler, K 'Adjacency relationships in boundary graph based solid models' (*to appear*) (1983)

6 Weiler, K 'Edge based data structures for solid modelling in curved-surface environments' *IEEE Comp. Graphics and Applic.* Vol 5 (1985) pp 21–40

7 Wilson, P 'Euler formulas and geometric modelling' *IEEE Comp. Graphics and Applic.* Vol 5 (1985) pp 24–36

8 Requicha, A 'Representation of rigid solid objects' in Encarnacao, J (ed) *Computer aided design*, Springer-Verlag Lecture Notes CS 89, (1980) pp 2–78

9 Owen, J *Private communication concerning Romulus 7* (1986)

10 Paoluzzi, A, Ramella, M and Santarelli, A 'Un modellatori geometrico su rappresentazioni "triango-alate" *Tech. Report 13.86* Dept. of Inf. and Systems, University of Rome, Rome, Italy (1986)

11 Wesley, M, Lozano-Perez, T, Liberman, L, Lavin, M and Grossman, D 'A geometric modelling system for automated mechanical assembly' *IBM J. Res. and Dev.* No 24 (1980) pp 64–74

12 Hilbert, D and Cohn-Vossen, S *Geometry and the imagination* Chelsea Publishing, New York, NY, USA (1952)

13 Requicha, A and Voelcker, H 'Boolean operations in solid modelling: boundary evaluation and merging algorithms' *Proc. IEEE 73* (1985) pp 30–44

14 Levin, J 'A parametric algorithm for drawing pictures of solid objects composed of quadric surfaces' *Comm. ACM* No 19 (1976) pp 555–563

15 Ocken, S, Schwartz, J and Sharir, M 'Precise implementation of CAD primitives using rational parametrizations of standard surfaces' *Tech. Report 67* Courant Institute, New York University, NY, USA (1983)

16 Sederberg, T and Anderson, D 'Steiner surface patches' *IEEE Comp. Graphics and Applic.* Vol 5 (1985) pp 23–36

17 Walker, R *Algebraic curves* Springer Verlag

18 Mortensen, M *Geometric modelling* Wiley, New York, NY, USA (1985)

19 Farouki, R 'The characterization of parametric surface sections' *Comp. Vision, Graphics and Image Proc.* No 33 (1986) pp 209–236

Application of structural optimization using finite elements

P Ward, D Patel, A Wakeling and R Weeks

This paper describes the theoretical basis of SDRC's redesign package, Optisen. The use of the program is illustrated using three realistic applications. First, the design of a plastic seat tub for strength requirements. Second, the design of an excavator arm for stiffness and strength requirements. Third, the modification of a helicopter for dynamic response.

computer-aided design, finite element analysis, structural optimization

NOTATION

*	— denotes optimum (e.g. Z^*)
i	— refers to the design variable (e.g. Z_i)
j	— refers to the constraint equation (e.g. g_j)
m	— denotes membrane value (e.g. σ_m)
b	— denotes bending value (e.g. σ_b)
$g_j(z)$	— constraint equation as a function of design variables
G	— matrix of constraint gradients
h	— incremental step in design space
H(z)	— diagonal matrix of second derivatives of W with respect to each design variable
K	— structure stiffness matrix
L	— Lagrangean function
M	— mass matrix
r	— deformed shape in a static analysis
R	— load vector in a static analysis
ω	— radian frequency
λ_j	— Lagrange multiplier
σ	— actual stress
∇W	— vector of first derivatives of W with respect to each design variable
Z_i	— design variable, typically thickness of shells
I	— unit matrix
W	— structure mass
V	— weighted forced response of structure
$\bar{\sigma}$	— allowable stress

Until recently, structural optimization has not made a significant impact on the wider analysis community. Several reasons may be advanced, the most prominent in the mechanical engineering field being the requirement for shell bending elements. In addition, the capability has generally been made available as *ad hoc* packages loosely connected to an analysis program. SDRC's Optisen offers integrated analysis and redesign with interactive graphics capabilities for both the specification of the design problem and the interpretation of results.

STRUCTURAL OPTIMIZATION

To illustrate the application of structural optimization, it is necessary to define the commonly used terminology.

The process of optimization implies producing the best design for a structure under prescribed loading conditions. The relative merit of alternative designs is generally evaluated with reference to five factors. First, satisfactory performance: the designer may specify upper and lower limits on displacements, generally referred to as stiffness constraints. In addition, limits may be specified on allowable stresses referred to as strength constraints.

Second, structural mass: of the possible designs satisfying the performance requirements that with minimum mass is defined to be the best. In optimization terminology, mass becomes the objective function.

Third, analysis variables: those structural parameters which are fixed at the outset. Various fields of optimization consider certain parameters fixed. In this paper, material properties and outline geometry are considered fixed. In addition, there may be any number of finite elements within the model of the structure assigned to be analysis elements (i.e. not allowed to change).

Fourth, design variables: those parameters defining a structural system which are allowed to vary. For example, in a finite element model comprised of shell elements the design variable allowed to vary is the thickness of the element. As groups of finite elements tend to correspond to part of a physical component desired to have uniform thickness, the element thicknesses are tied to one free variable for that group. This approach is referred to as design variable linking.

Finally, gauge constraints: in addition to constraints on structural performance upper and lower bounds may be specified on the design variables. The lower gauge constraint

SDRC CAE International, York House, Stevenage Road, Hitchin, Herts, UK

0010-4485/87/030148-09 $03.00 © 1987 Butterworth & Co (Publishers) Ltd