N92-24416

SINGULARITY CLASSIFICATION AS A DESIGN TOOL FOR MULTIBLOCK GRIDS

Alan K. Jones
Boeing Computer Services
Bellevue, WA

INTRODUCTION

A major stumbling block in interactive design of three-dimensional multiblock grids is the difficulty of visualizing the design as a whole. One way to make this visualization task easier is to focus, at least in early design stages, on an aspect of the grid which is inherently easy to present graphically, and to conceptualize mentally, namely the nature and location of singularities in the grid. The topological behavior of a multiblock grid design is determined by what happens at its edges and vertices. Only a few of these will be in any way exceptional. These exceptional behaviors lie along a singularity graph, which is a one-dimensional construct embedded in three-dimensional space. The varieties of singular behavior are limited enough to make useful symbology on a graphics device possible. Furthermore, some forms of block design manipulation that appear appropriate to the early conceptual-modelling phase can be accomplished on this level of abstraction.

The present work is an overview of a proposed singularity classification scheme and selected examples of corresponding manipulation techniques, drawn mostly from (ref. 1). Mathematical details can be found in (ref. 2) or (ref. 3). It will not treat practical issues of how a computer interface based on such principles would look. Nor will it consider in detail the correspondence between singularity graphs and the resulting block designs.

SINGULARITY CLASSIFICATION AS A DESIGN TOOL FOR MULTIBLOCK GRIDS

For the purposes of this discussion, a multiblock grid is a decomposition of a region in space into blocks, each of which is the image of a cube in parameter space under a smooth, invertible coordinate map. Neighboring cubes intersect only in complete faces, edges or vertices. This last condition is by no means universal in multiblock applications, but it will simplify the analysis greatly.

For numerical applications, decompositions should ideally be conformal in the sense that the blocks meeting at each vertex can be assembled into a consistent local coordinate system. This is analogous to the condition in two dimensional gridding that four blocks, each subtending a right angle in parameter space, should meet to fill out the 360° angle at each vertex. Special numerical techniques are generally required at points which are not conformal in this sense. See, for example, Section IV.3 of (ref. 4). In three dimensions, a conformal vertex is one where eight blocks meet, in an arrangement equivalent to the standard configuration of one block per octant. Similarly, four blocks should meet along each interior edge. Any failure of conformality is called a combinatorial singularity. In many cases, it is possible to remove combinatorial singularities by relaxing the conditions on the coordinate maps. For example, an entire face of a block in computational space may be collapsed to a point or a line in real space, or two adjacent faces may be unfolded to become coplanar. Such analytical singularites are really just an alternative realization of the same underlying phenomenon. However we choose to represent them, singularities are an unavoidable fact of life. Many interesting domains, in both two and three dimensions, simply do not admit a completely nonsingular blocking. A discussion of commonly used three-dimensional grid designs and their singularities, and the computational consequences of these singularities, can be found in (ref. 5).

An interior edge is singular if other than four blocks meet along it. Since there is in effect only one degree of freedom in the location of the blocks, it is enough to count them.The singularity graph is is just the union of all the singular edges. With appropriate bookkeeping conventions, it is always possible to represent the singularity graph as the union of a set of pentagonal curves (strings of edges along which five blocks meet) and a set of triangular curves (strings of edges along which three blocks meet). These curves do not terminate in the interior of a domain; they must either be closed cycles or open curves, with endpoints on the domain boundary. However, the curves need not be disjoint. For example, two pentagonal curves may coincide to give a string of edges along which six blocks meet. Where singularity curves cross, necessarily at vertices, we get complicated behavior. Thus, a simple but useful display of the topological nature of a block design would be to draw the singularity graph, with each edge color coded according to its type, and intersections clearly noted.

We can make the display more useful by encoding information about what happens at singularity curve intersections. Behavior at an interior vertex can be visualized by drawing a certain convex polyhedron, which is in effect just a picture of a small neighborhood of the vertex. To see this, fix a vertex v, and consider all the blocks that meet at v. Together, these form a neighborhood of v which we call U, topologically equivalent to a three-dimensional ball. Unfortunately, to understand the blocking of this neighborhood, it is not enough just to count the blocks. We need to draw its face adjacency graph, denoted $G(v)$. The nodes of G are the blocks that meet v. An edge of G connects two nodes if and only if their corresponding blocks share a face. The graph G has a very concrete geometrical meaning. It is the one-dimensional skeleton of the bounding surface of U. For example, the graph at a non-singular vertex is just the graph of edges of the 3-cube, and the neighborhood *U* does in fact look just like a 3-cube.

Figure 1 shows the six most common polyhedra, corresponding to singular internal vertices adjacent to at most 12 blocks. Two faces of the N=12 case are shaded to aid in visualization. Not shown is the only non-singular case, the 8-block polyhedron which is an ordinary cube. (It is an interesting fact that N, the number of blocks, must be even.) From now on, we adopt the convention that curves are labelled with their total net singularity, i.e., *b-4*, where *b* is the number of blocks meeting along the curve. Curves with negative net singularity will be shown in grey, while curves with positive net singularity are shown in black.

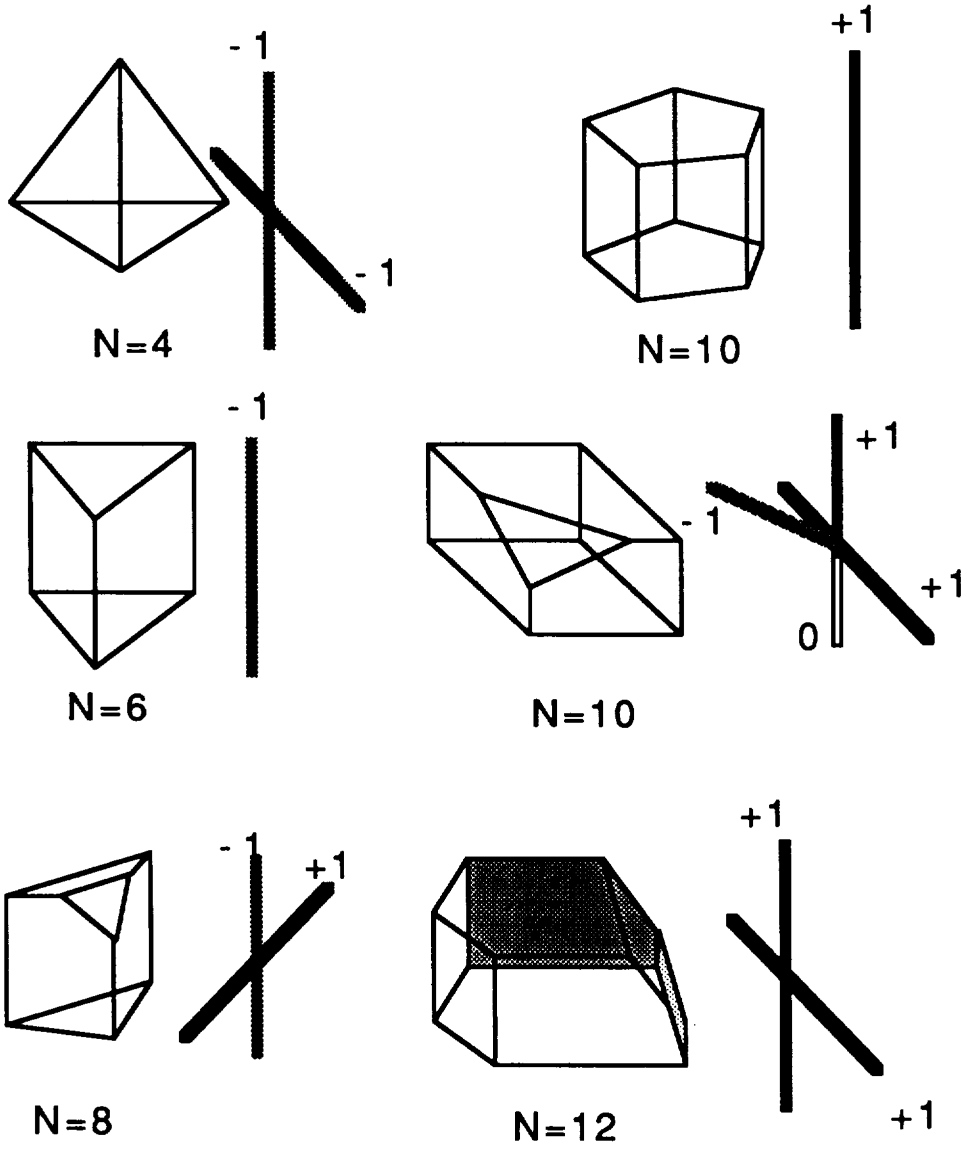

Figure 1. Singular polyhedra corresponding to common crossing patterns.

We will discuss two types of grid manipulations that can be done at the level of singularity graphs. The first type allows us to avoid the more exotic behaviors cataloged in figure 1. (The second type has to do with behavior on the boundary, and we will treat that later.) For example, figure 2 shows how a blocking scheme can be modified to break the connection between two intersecting curves of triangular singularities, by introducing new blocks so that a polyhedral neighborhood that began as a tetrahedron becomes instead a union of two triangular prisms. Figure 3 is the analogous construction for two pentagonal singularity curves.

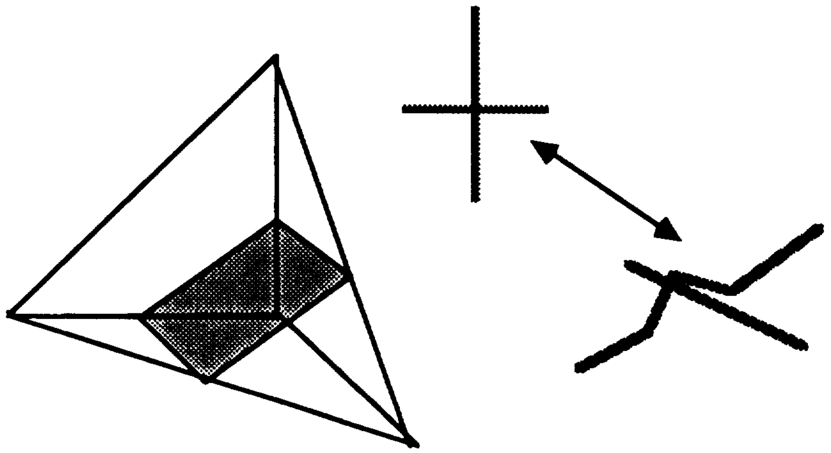
Figure 2. Removing an intersection between triangular curves.

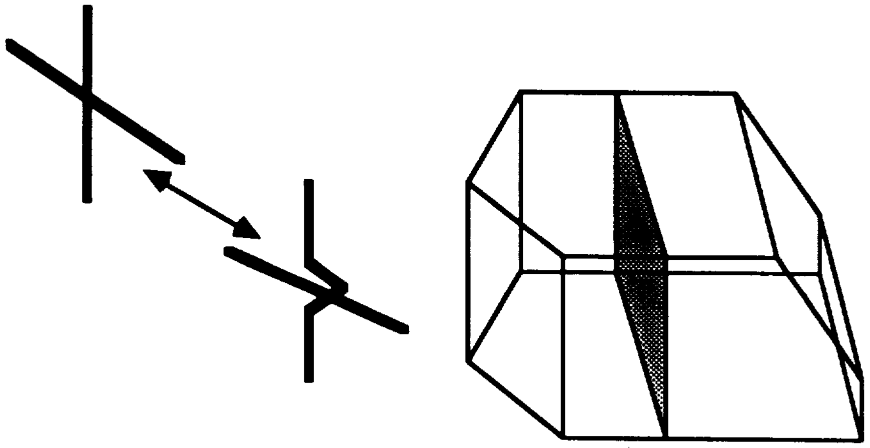
Figure 3. Removing the intersection of two pentagonal curves.

Similarly, figure 4 shows how to remove the intersection of a pentagonal and a triangular curve. The situation in figure 5 is more delicate. The most useful interpretation of this diagram is that one triangular and one pentagonal curve can coalesce when they cross a second pentagonal curve and leave the intersection as a non-singular curve, i.e., one along which four blocks meet. This construction will turn out to be crucial in the manipulation of boundary creases and corners.

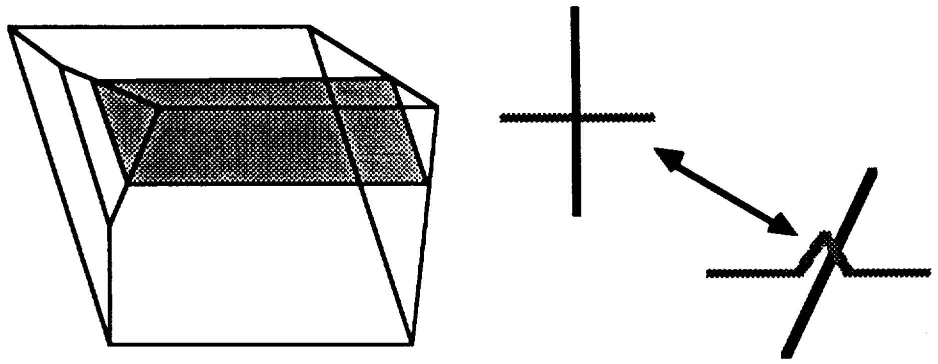
Figure 4. Removing an intersection between curves of mixed type.

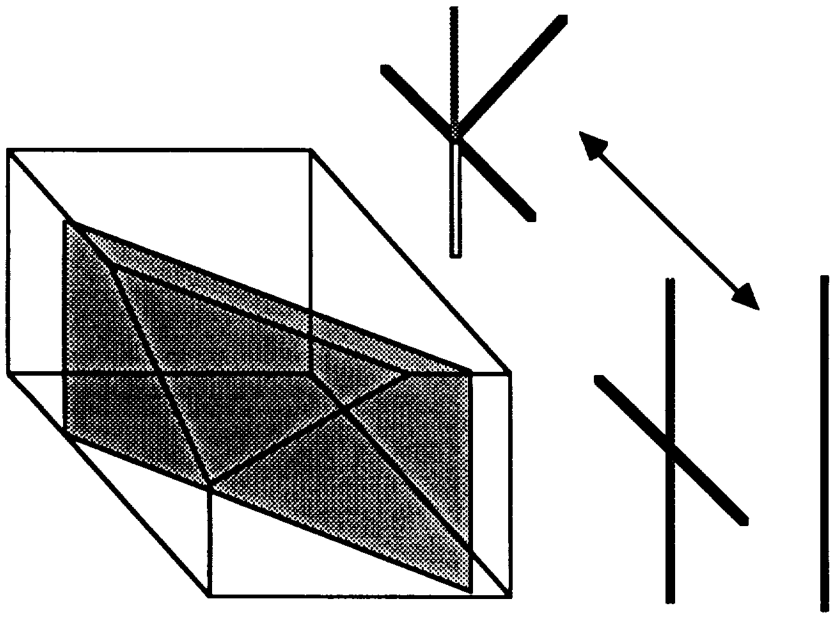
Figure 5. Coalescing curves of opposite type at a pentagonal curve.

Now we wish to put these constructions to use in constructing block designs. Consider first the simplest case, a computational domain which is a solid ball with smooth boundary. Each octant of the ball can be parametrized smoothly by a tetrahedron, which has the singularity graph shown in figure 6. Pasting eight of these graphs together gives the complete singularity graph of the solid ball, as shown in figure 7.

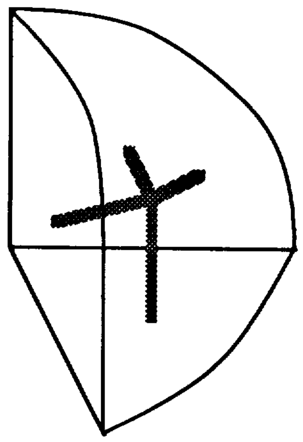

Figure 6. The singularity graph of one octant of a ball.

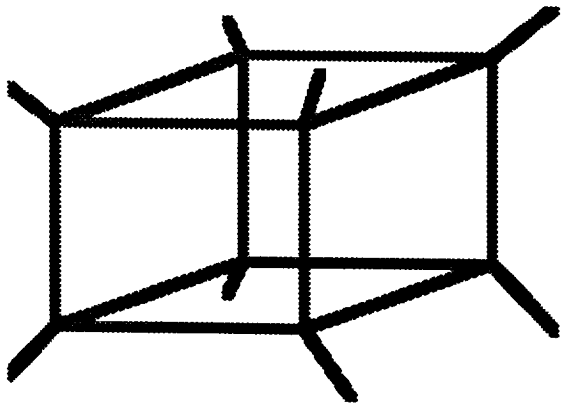

Figure 7. The complete singularity graph of the solid ball.

Figure 8 shows four alternative ways of viewing the singularity graph of a ball. In the upper left hand corner, the graph from figure 7 is reproduced. This graph is connected but is not a manifold. In the upper right hand corner, the four vertical curves have been pulled off the graph, leaving two closed cycles and four open curves. In the lower left hand corner, four different open curves have been pulled off, leaving only one cycle. Finally, on the lower right, we have four open curves and no cycles at all. Evidently, simple counting is not enough to characterize this topology. The only obvious invariant is the total number of open singularity curve ends that terminate on the boundary of the region. For a smooth manifold, this is always -4 χ, where χ is the Euler characteristic. Thus, for a smooth spherical boundary, we must have a net count of -8 singularity curve ends. See (ref. 3) for details.

The first two alternatives are probably the most useful in practice because they reflect the symmetries of real computational domains. The connected arrangement of figure 7 will be referred to as the local version because in it the singularites can be kept arbitrarily close to the boundary. The other arrangements are referred to as global versions, because the open singularity curves are typically threaded through the whole length of the domain.

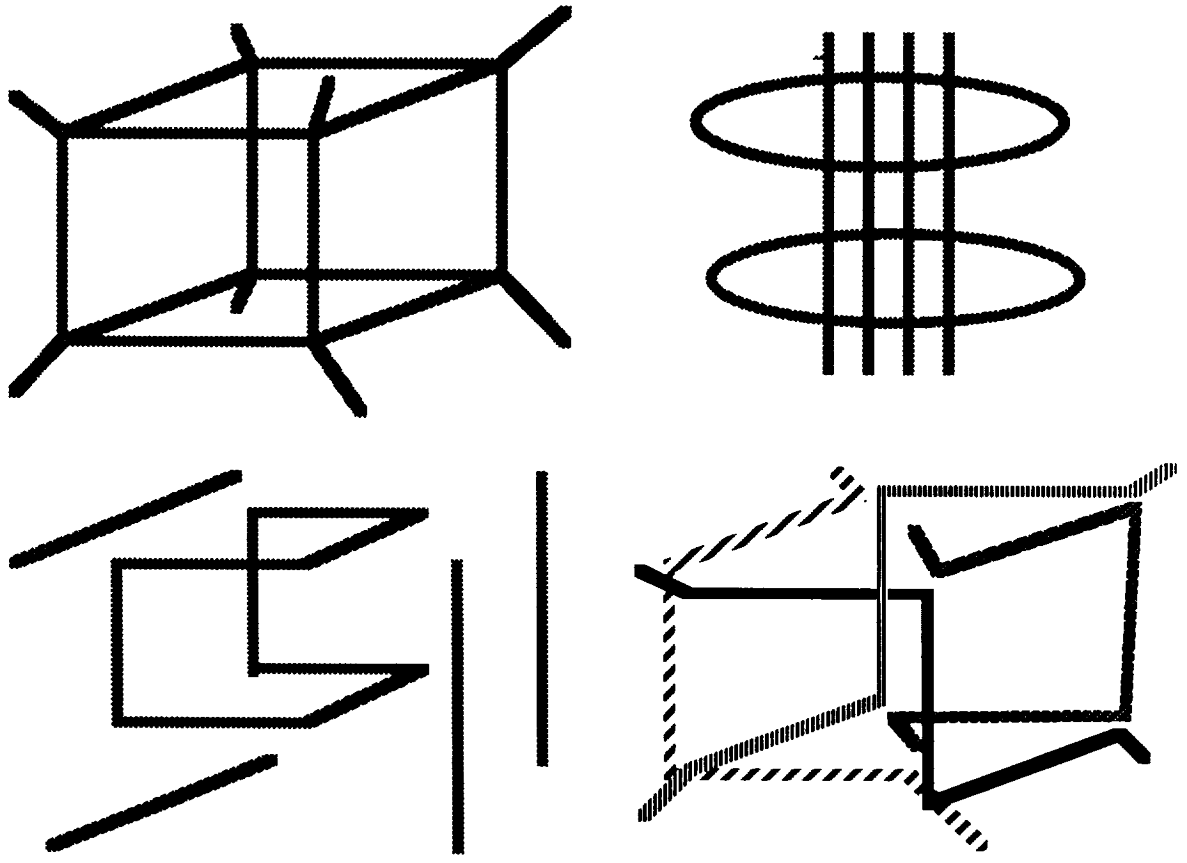

Figure 8. Separating a singularity graph into disjoint curves.

Next, we consider the dual case of a spherical void inside a cubical domain. One octant of this domain looks like the 10-sided polyhedron shown in figure 5. Pasting eight of these together gives figure 9, which is the standard local scheme for a smooth, simply connected void. Using the subdivision of figure 5 allows us to separate the four vertical pentagonal curves from the singularity graph and arrive at the global scheme of figure 10. In this case, we have a total of four pentagonal curves connecting the outer boundary to itself, and a total of eight triangular curves connecting the outer boundary to the inner boundary. This is consistent with the total singularity count of -8 for the smooth spherical inner boundary, and 0 for the outer boundary, which is assumed to be a non-smooth rectangular prism. See (ref. 1) or (ref. 3) for details.

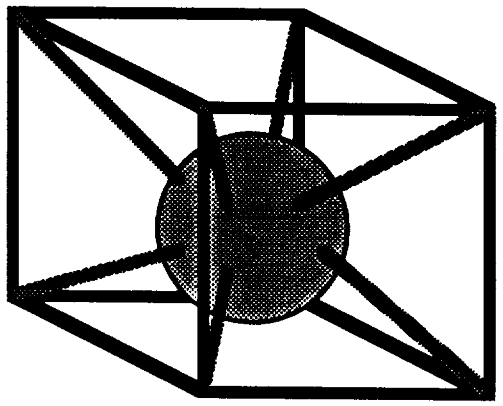

Figure 9. The complete local form for the complement of a ball.

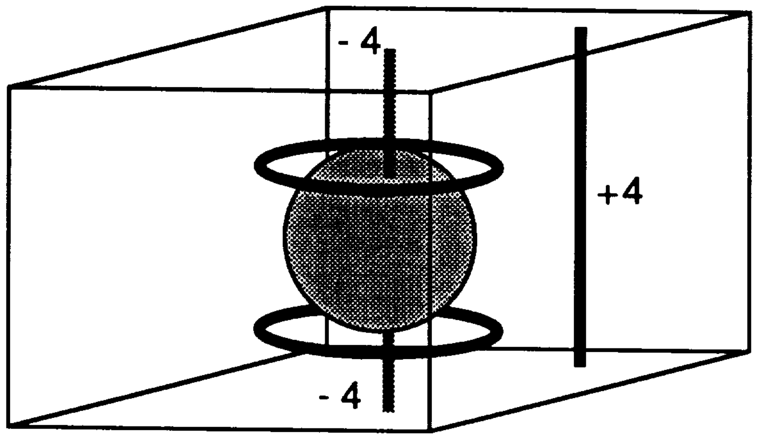

Figure 10. A global form for the complement of a ball.

The shapes of the singularity graphs in figures 7 and 9 are suggestive. In each case, what we see is the edge graph of the 3-cube, connected to the smooth boundary of the region by triangular edge "stingers" at all eight corners. The only difference is that, for the smooth outer boundary of figure 7 the 3-cube graph consists of triangular curves, while for the smooth inner boundary of figure 9 it consists of pentagonal curves. This is not accidental. In each case, we may start with a boundary which is just a rectangular prism, and smooth away its corners and creases. (A crease is a boundary edge along which the coordinate species of the outward pointing normal changes. A corner is where three or more creases meet.) This is the second major class of singularity manipulation techniques. The process is shown in figure 11. The operation of removing a crease consists of adding an extra block along each boundary face (the shaded faces in figure 11). Blocks corresponding to edge-adjacent boundary faces are made face-adjacent. (In figure 11, these faces are crosshatched.) Thus, a convex crease, such as we find on an external boundary, corresponds to a triangular singularity curve, and a concave crease, such as we find on an internal boundary, to a pentagonal curve.

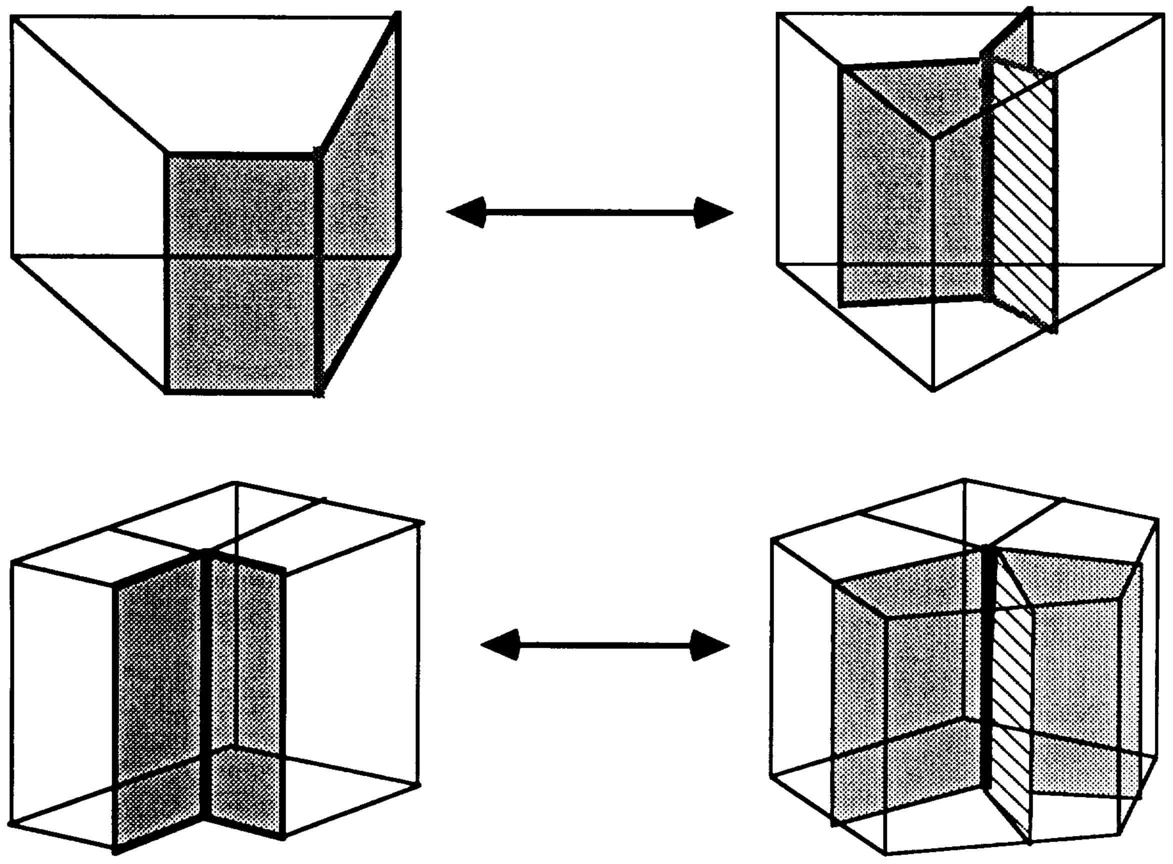

Figure 11. Equivalence of boundary creases and internal singularity curves.

The situation at corners is somewhat more delicate, as shown in figure 12. In the left hand column are typical arrangements of blocks on the boundary of a domain, with boundary faces shaded, and the corner of interest marked by a black dot. In the right hand column are the corresponding polyhedra. The new blocks added along shaded faces correspond to the vertices marked with open squares in the polyhedra; the existing blocks correspond to vertices marked with shaded circles. We see that a convex corner, like that on an outer boundary, corresponds to a tetrahedral singularity, and a concave corner, like that on an inner boundary, to the exceptional N=10 case shown in figure 5. In either case, the internal singularity is connected to the new boundary by a triangular singularity curve because the polygonal face formed by the new boundary blocks is triangular. The third row in figure 12 shows the behavior at a saddle corner, which we would find, for example, at a wing-body join. The internal singularity is the same as for a concave corner, but it is now joined to the boundary by a pentagonal curve, rather than a triangular one.

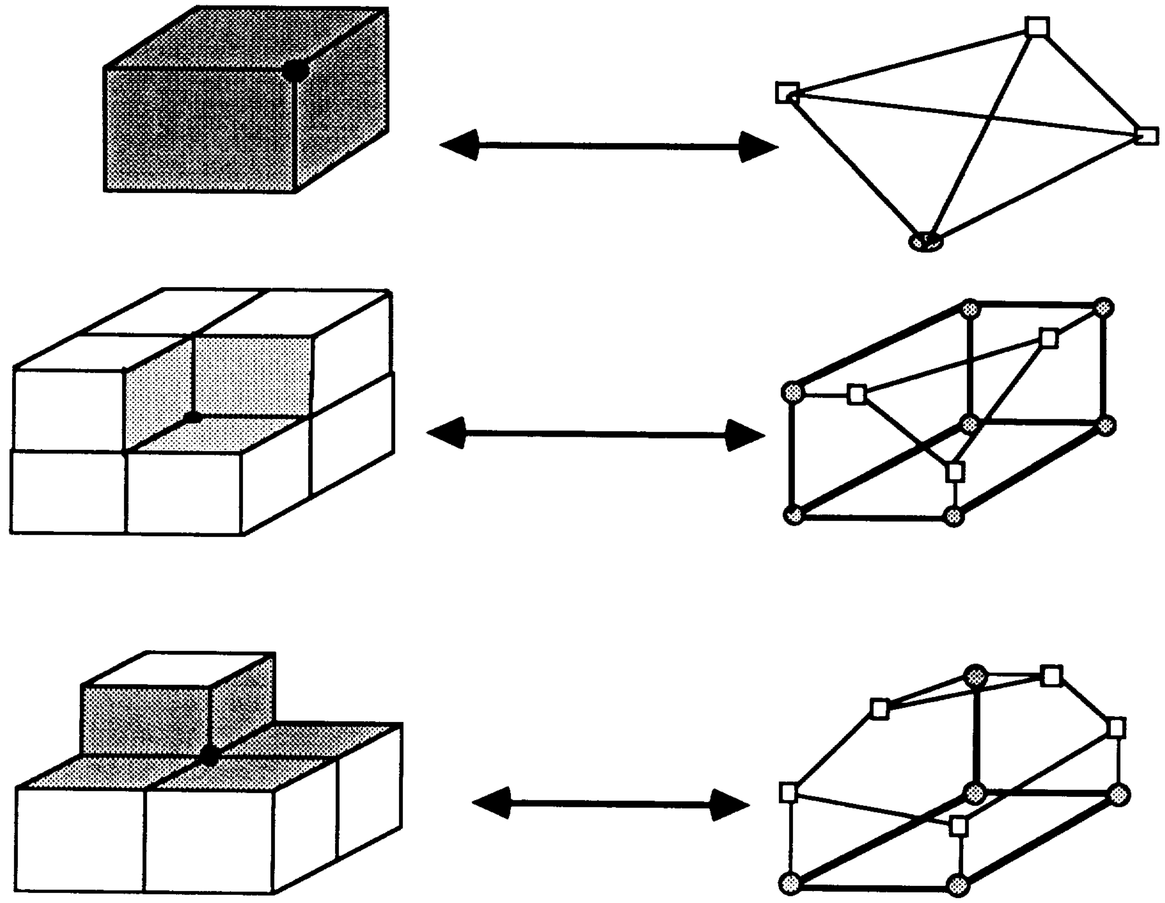

Figure 12. Equivalence of boundary corners and internal singularity vertices.

As a practical illustration of this smoothing procedure, consider blocking in the neighborhood of a rounded wingtip, as shown in figure 13. Here, for the sake of visual clarity, we have changed conventions. It is the four unshaded faces of the square wingtip which are smoothed. The resulting pair of triangular curves that terminate on the wingtip surface corresponds to a parabolic singularity in the terminology of (ref. 5).

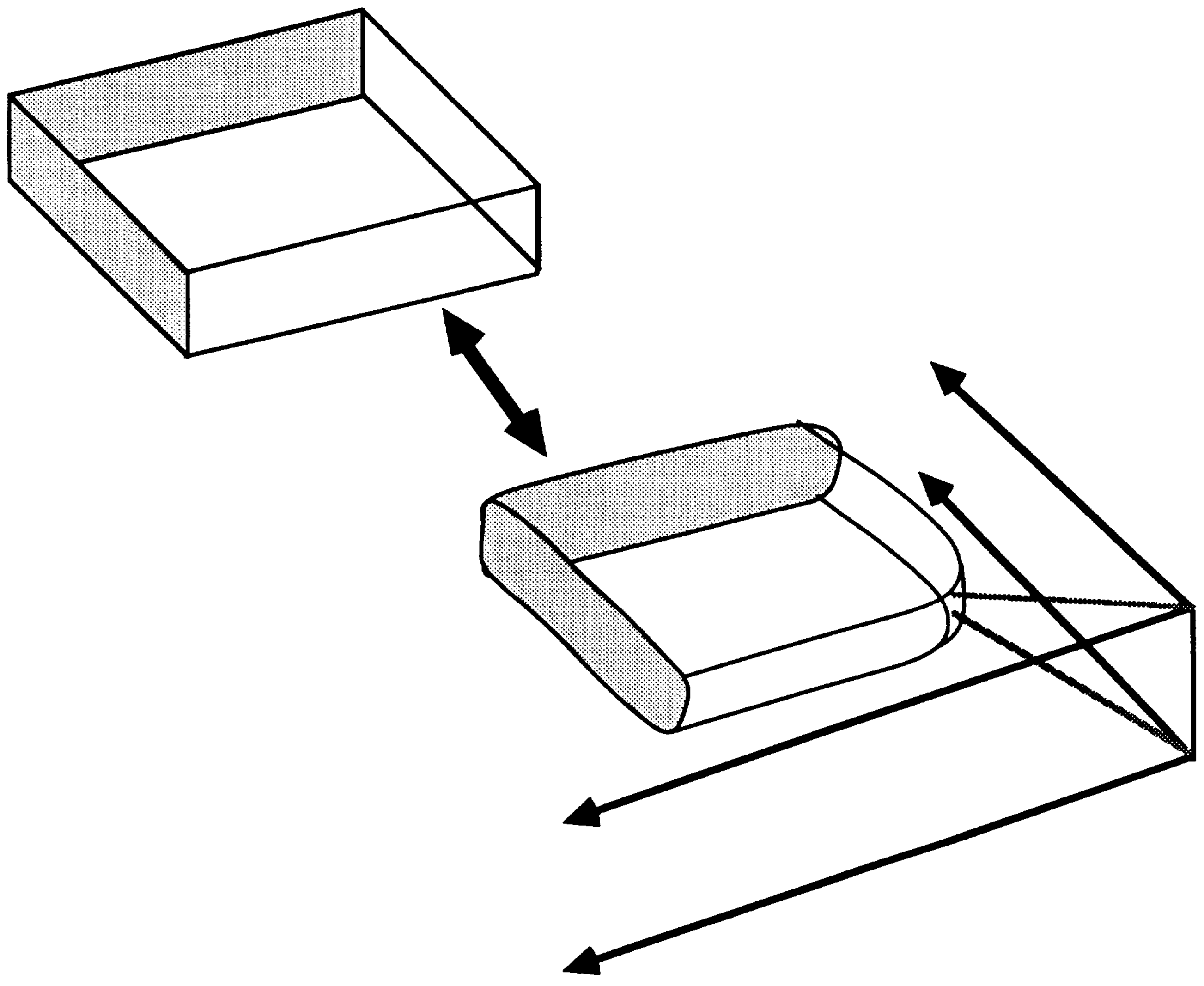

Figure 13. Blocking near a rounded wingtip.

For a second, example, consider how the singularity graphs behave under a basic operation of constructive solid modeling. View a landing gear assembly as the union of two cylinders, as shown in figure 14. We want a block design for the complement of the union of these cylinders. The first step is to find a blocking for the complement of one cylinder. We start with the block design of figure 15 for one octant of this complement, and, as usual, paste eight copies together to get the singularity graph of figure 16. For clarity, note that the individual singularity curves are drawn as though collapsed into groups of four each, although this would probably not be done in practice. For concreteness, figure 16 also shows in top view the block boundaries that would be seen with this design.

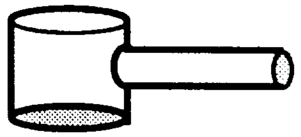

Figure 14. Landing gear modelled as the union of two cylinders.

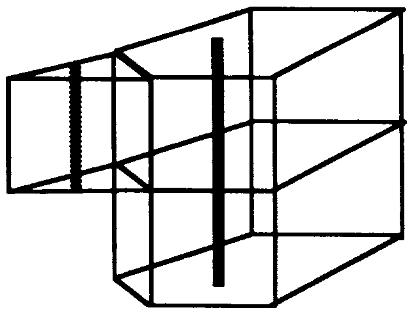

Figure 15. Blocking around one octant of a finite cylinder.

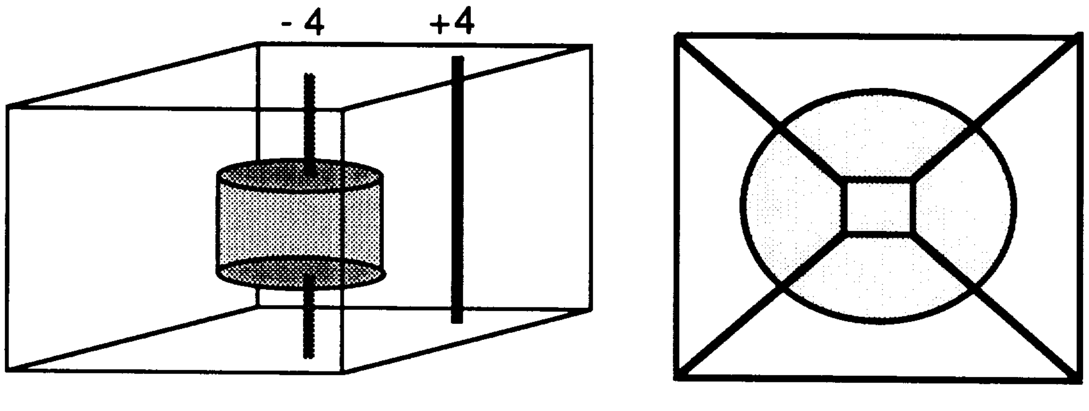

Figure 16. Singularity graph and top view of blocking around a finite cylinder,

If the two cylinders were disjoint, then the singularity graph we seek would be two disjoint copies of figure 16. However, they do intersect, and in a non-symmetric way, with one cylinder pierced by the other. The effect of this is that one set of pentagonal singularity curves now terminates on the surface of the pierced cylinder, where two-dimensional pentagonal singular vertices are needed to block around the circular hole cut by the piercing cylinder. See (ref. 3) for details. This is shown in figure 17.

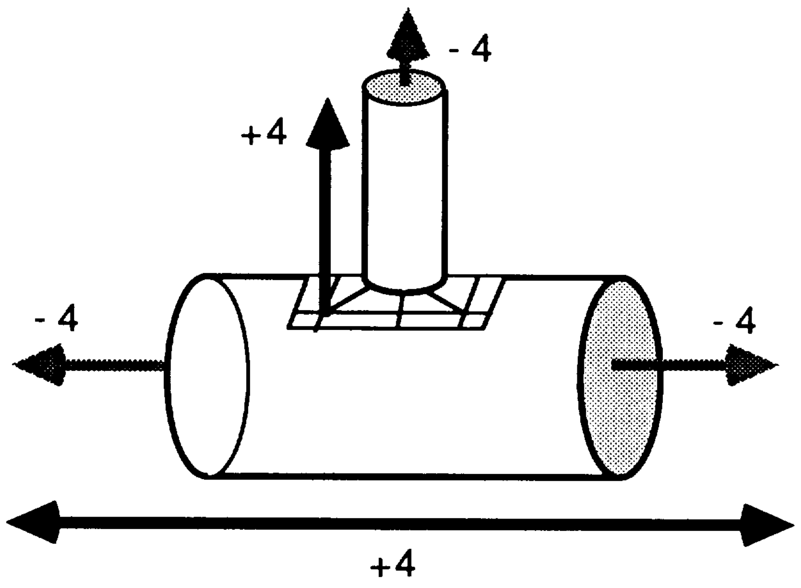

Figure 17. Singularity graph around the union of two cylinders.

When intuition fails in cases like this, the best algorithmic approach is to replace all solids by rectilinear forms, as in figure 18, construct a blocking around them, which can always be made non-singular, and then pull creases and corners off the boundary until the smooth case is recovered. The pentagonal curves terminating on the landing gear surface are then seen to be consequences of the saddle corners contained in the curve of intersection between the two rectangular prisms, marked with black dots in figure 18. Recall the third row of figure 12.

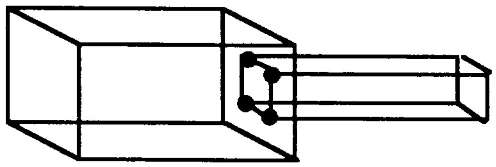

Figure 18. The rectilinear analog of figure 14, showing four saddle vertices.

Finally, consider a complete half-body-wing-pylon-nacelle assembly. A schematic section view is shown in figure 19, along with a typical two-dimensional block arrangement. The large black dots mark two-dimensional vertex singularities, vertices where other than four blocks meet. There are six pentagonal vertices in this scheme, and indeed it can be shown (ref. 2) that any two-dimensional blocking of this domain must have a net singularity count of +6 . Sweeping this two-dimensional design along the aircraft's longitudinal axis gives the set of three-dimensional singularity curves shown in figure 15. Note that, as in figure 16, if the cylindrical body and nacelle do not extend to infinity, triangular singularity curves arise as shown, connecting the ends to the boundary of the computational domain. At this stage, both the wing and the pylon have been left as rectilinear slabs.

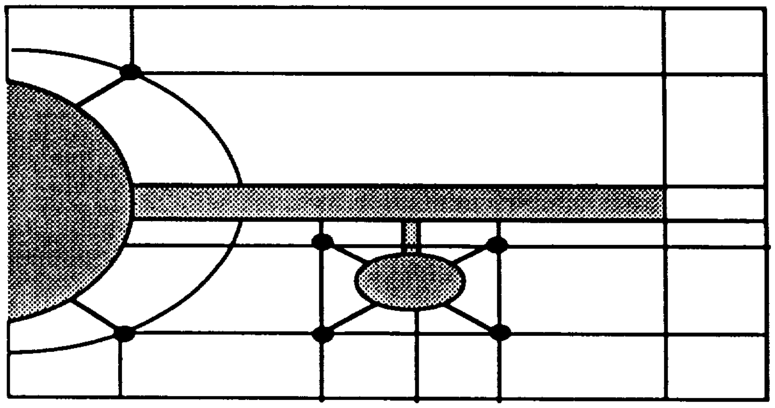

Figure 19. Two dimensional block design for half-body-wing-pylon-nacelle.

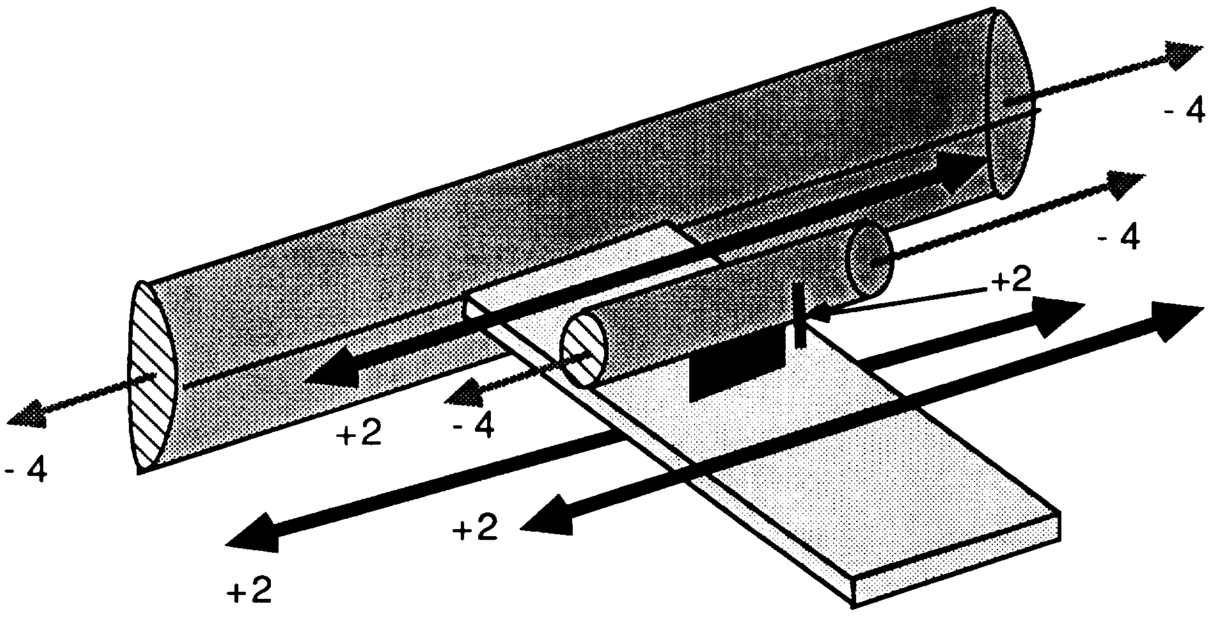

Figure 20 Three-dimensional singularity graph from sweeping figure 14.

Now we selectively pull creases and corners off of the airplane surface into the interior of the domain. If the nose and tail of the body and the leading edge of the wing and pylon are rounded, the result is the graph shown in figure 21. Additional modifications could be performed. Consider, for example, rounding the wingtip as in figure 13, or adding a flow-through nacelle.

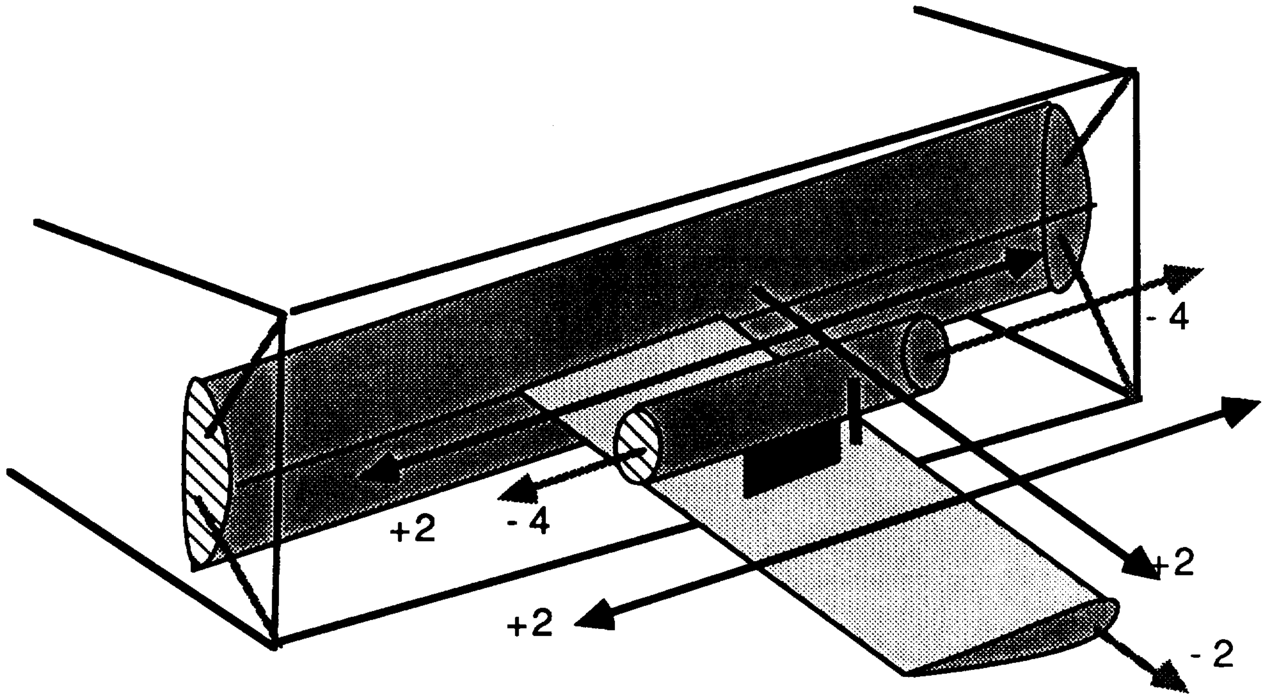

Figure 21. Selective rounding of inner boundary edges.

REFERENCES

1. Jones, Alan K.: Mathematical tools for 3D multiblock grid design. Tech. Report ETA-TR-121, Boeing Computer Services, June 1989.

2. Jones, Alan K.: Grid singularities in the plane. Tech. Report ETA-TR-82, Boeing Computer Services, March 1988.

3. Jones, Alan K.: Grid singularities in three-dimensional domains. Tech. Report ETA-TR-83, Boeing Computer Services, March 1988.

4. Thompson, J.F., Warsi, Z.U.A., and Mastin, C.W.: *Numerical Grid Generation,* North-Holland, 1985.

5. Eriksson, L.E.: Practical three-dimensional mesh generation using transfinite interpolation. *SIAM J Sci Stat Comp,* vol. 6, 1985, pp. 712-741.