GEOMETRIC MODELING: A First Course

5. Solids

This chapter discusses 3-D solids. First we address mathematical modeling issues, and then representations. We consider only single objects that are rigid and made of homogeneous materials. Assemblies of several components and inhomogeneous objects raise additional issues, which are briefly discussed in the section on Further Explorations.

Solids may be represented by using the various methods discussed in Chapter 3. Solidity does not raise significant new issues for most of these methods. However, BReps of solids deserve a more detailed treatment than that provided in Chapter 3, and are the main focus of our representational discussion in this chapter.

5.1 Mathematical Models for Rigid and Homogeneous Solids

Most of the physical objects we encounter in real life are solids. Sometimes they can be modeled as surfaces or curves. For example, in stress analysis, thin shells are usually analyzed as if their thickness was effectively zero. But full 3-D models are required for many applications. In addition, we often need to model not only solid objects but also operations on them. For example, fabrication processes such as machining or welding are important in CAD/CAM. The geometrical aspects of machining may be modeled as the (regularized) difference between the initial state of the workpiece and the volume swept by the cutter in its motion. Welding and other additive processes may be modeled by set union.

Computationally-useful mathematical models for rigid solids should exhibit the following properties.

Rigidity – This is easily achieved since the distances and angles among points of a set in Euclidean space are fixed. Rigid motions preserve distances and angles. Therefore all the instances of a set obtained from one another by rigid motions can be used to model a rigid object in all its poses.

Finiteness – A physical object should have a finite extent. To ensure finiteness all we need is to require that our sets be *bounded.*

Solidity – A model for a solid should be homogeneously 3-D, without dangling faces or edges. We saw earlier that this requirement is met by *regular* sets in 3-space.

Closure under Boolean operations – Boolean operations applied to solids should produce other solids. This has two important advantages. First, the results of a Boolean operation can be used as inputs to other Booleans, and a solid model can therefore be incrementally constructed by successive Boolean operations (perhaps interspersed with other operations).

Second, subtractive and additive manufacturing-process models are guaranteed to produce solids.

Finite describability – Point sets used to model solids should be describable by a finite amount of data, to ensure that they can be represented in computers, which have finite memories. (Here we assume that real numbers can be represented exactly in computers, i.e., we use the Real Random Access Machine model of computation.) Polyhedra can be described finitely by the coordinates of their vertices plus connectivity information that specifies how vertices define edges and faces. But polyhedral models have too small a domain. What we need are models akin to polyhedra, but with curved faces.

Boundary determinism – A solid should be modeled by a set that is unambiguously defined by its boundary. This may seem a truism, but actually there are examples, such as the Lakes of Wada discussed in [Hocking & Young 1988], in which three or more bounded sets all have the same boundary. Sets that exhibit such behavior are counter-intuitive and poor models for physical objects. In addition, without boundary determinism we will not be able to use BReps, which are one of the most popular schemes for representing solids.

It can be shown that sets that are bounded, regular and semi-algebraic possess all the desired properties, and therefore provide appropriate models for solids. These sets are usually called simply *r-sets.* Intuitively, r-sets are curved polyhedra with faces lying on algebraic surfaces. A more precise characterization follows.

A *semi-algebraic half space* is a set of points that satisfy an algebraic inequality

$$\{\mathbf{p} : f(\mathbf{p}) \leq 0\}$$

where f is a polynomial. For example, the inequality

$$ax + by + cz + d \leq 0$$

defines a planar half space, i.e., the portion of 3-space which lies to one side of the plane defined by the equation

$$ax + by + cz + d = 0 \ .$$

A *semi-algebraic set* is the result of a finite number of (standard, unregularized) set-theoretic operations on semi-algebraic half spaces. For example, a finite solid cylinder is the intersection of three semi-algebraic half spaces. One of these is cylindrical and the other two are planar, as shown schematically in Figure 5.1.1. (Each half space itself is unbounded; only bounded portions of the half space boundaries are shown in the figure.)

Because $-f$ is also a polynomial, and $-f \leq 0$ is equivalent to $f \geq 0$, we could have defined semi-algebraic half spaces with inequalities of the form $f \geq 0$. Furthermore, the intersection of the two halfspaces $f \geq 0$ and $f \leq 0$ is the set defined by the equation $f = 0$, and this is an algebraic set, or algebraic variety, defined earlier in Chapter 4. Therefore algebraic sets are special cases of semi-algebraic sets.

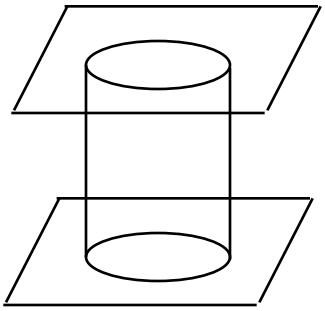

Figure 5.1.1 – A finite cylinder is the intersection of a cylindrical and two planar half spaces

It can be shown that the interior, boundary and closure of a semi-algebraic set are also semi-algebraic. Therefore, a finite number of regularized Boolean operations on semi-algebraic sets produces another semi-algebraic set. This implies that a set defined by CSG on semi-algebraic half space primitives also is semi-algebraic. Furthermore, if the primitives are r-sets, the result also is an r-set, because regularized Booleans preserve boundedness, regularity and semi-algebraicity. This implies that CSG representations in the domain of r-sets are always valid.

A polyomial has a finite number of coefficients, and a semi-algebraic set is the result of a finite number of (non-regularized) Boolean operations on a finite number of half spaces defined by polynomial inequalities. Therefore, a semi-algebraic set is always finitely describable.

It is also true, but not trivially proved, that a bounded semi-algebraic set in 3-space is determined uniquely by its boundary, which is semi-algebraic as well. In addition, semi-algebraic sets do not exhibit certain pathological behaviors found in more general classes of sets. For example, there are sets such as “Alexander’s horned sphere” or “Antoine’s necklace” [Hocking & Young 1988], that are homeomorphic to spheres, and yet the portion of 3-space which lies inside each of them is not a topological ball. Such sets are said to be “wildly imbedded” in Euclidean space. Semi-algebraic sets cannot be wildly imbedded; they are always “tamely imbedded”.

In summary, r-sets are bounded, regular, semi-algebraic sets. They are rigid, finite, solid, closed under Boolean operations, finitely-describable, and uniquely determined by their boundaries. Therefore, they provide suitable models for rigid, homogeneous solid objects.

Some authors and systems prefer a more restricted class of models: r-sets that are bordered, connected 3-manifolds. Such sets have boundaries that are themselves closed 2 manifolds. Manifold models have computational advantages. For example, one can be sure that an edge is shared by only two faces, and this simplifies certain algorithms. However, manifolds are not closed under Boolean operations, as shown by the glued-cube examples of Figure 4.1.3.2.

R-sets need not be connected. For example, two disjoint cubes are an r-set. These cubes are considered rigidly linked, and move together when rigid motions are applied to them. In typical manifold-based systems, the two cubes are considered two distinct objects that can be “assembled”, although they need not even be near each other. The two objects can be moved together by applying a rigid motion to the assembly. Solids or assemblies made out of components that are not in contact with each other are physically counter intuitive, but cause no significant mathematical or computational difficulties.

5.2 Boundary Representations for Solids

Boundary schemes are the most widely used representations for solids. The following sections discuss basic concepts in boundary representations, and then focus on validity issues, and how to help construct valid representations by means of so-called Euler operators.

5.2.1 Boundary Graphs

Essentially, a BRep is a graph structure with nodes corresponding to faces, edges and vertices in a cell decomposition of the solid's boundary. Links between the nodes express connectivity information. Figure 5.2.1.1 provides a simple example. The BRep graph is shown only partially.

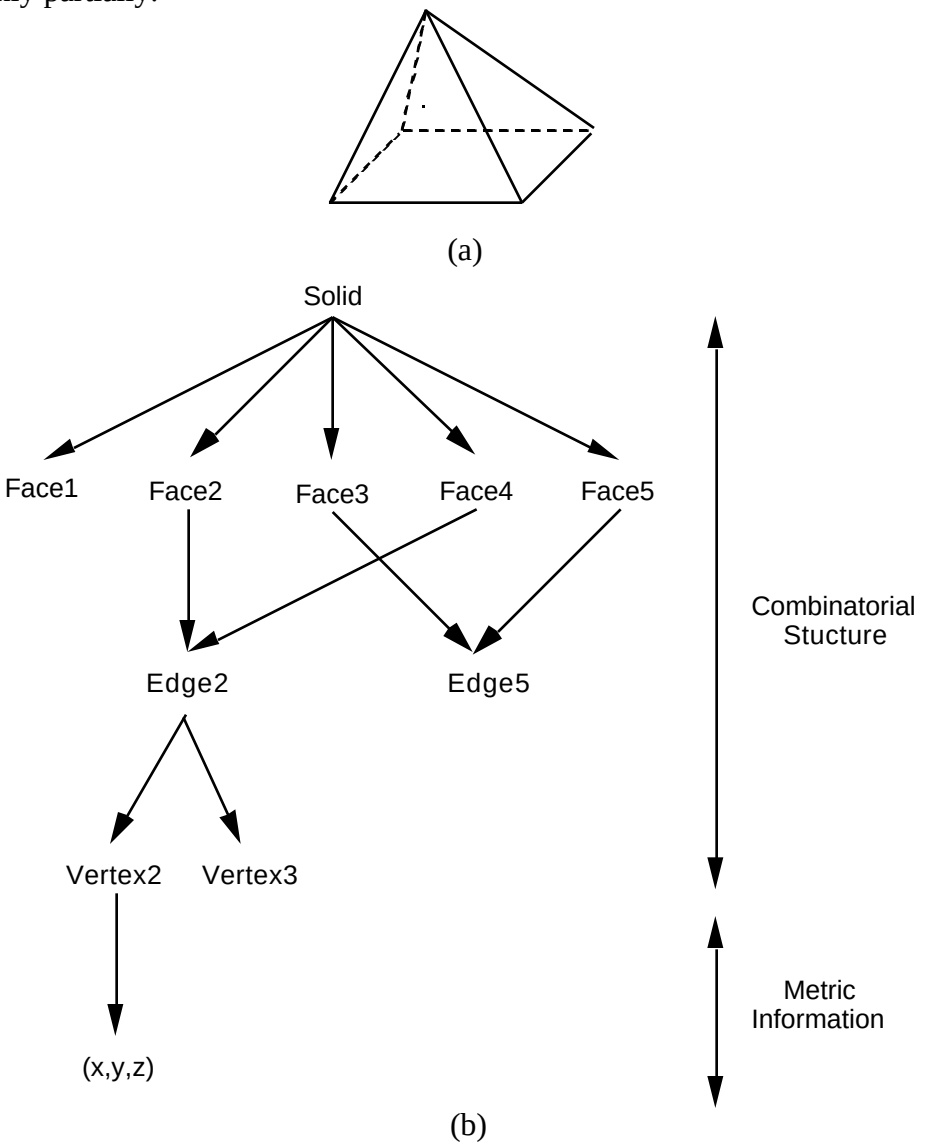

Figure 5.2.1.1 – A pyramid (a) is represented by a graph (b) containing face, edge and vertex nodes.

Observe that the top levels of the graph in the figure contain only connectivity information. Together they constitute the *combinatorial structure* of the representation. The vertex coordinates are the *metric information* associated with the representation. In the geometric modeling jargon, the combinatorial structure is often called the *topology*, and the metric information the *geometry.*

In the representation shown in the figure, the faces are all simple polygons, without holes. When the polygons are all of the same type, for example, all quadrangles, or all triangles, the BRep is called a *tessellation*. Tessellations with triangular faces are called *triangulations*. Tesselations are used extensively in hardware accelerators for rendering and other applications.

Faces with holes are used by many BRep modelers. They normally are represented by introducing an additional level in the combinatorial structure between faces and edges. Edges are grouped into closed circuits or *loops*, and face nodes point to loop nodes, which in turn are linked to their associated edge nodes.

For polyhedral objects it is usually clear what is meant by a face or an edge. But for curved objects intuition alone does not suffice. For example, what are the faces and edges of the object shown in Figure 5.2.1.2? For solids bounded by free form or sculptured surfaces, it is usually impossible to determine by inspection which portions of the boundary are the actual faces in the underlying representation.

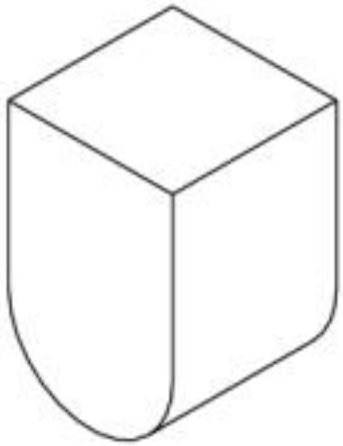

Figure 5.2.1.2 – What are the faces?

Faces and edges are best viewed as representational artifacts that must be carefully defined by the designers of BRep schemes. First, let us assume that there is a set of *primitive surfaces* (for example, planes and natural quadrics) for a boundary scheme. Then, typical face definitions should satisfy the following conditions.

1. A face is a subset of the solid's topological boundary.
2. The union of all the faces equals the boundary of the solid.
3. Each face is a subset of only one primitive surface instance.
4. A face is homogeneously 2-D, it has no dangling vertices or edges.
5. A face is a connected se
6. Faces are quasi-disjoint, i.e., meet only at edges or vertices.
7. A face is the largest subset of the boundary that satisfies all the previous conditions

It can be shown that, under mild conditions, faces that satisfy these conditions are defined uniquely [Silva 19??]. This does not imply that the corresponding BReps are unique. Two representations for the same object may still differ simply because of a different ordering of entities (permutational non-uniqueness), or different poses for the object (positional non-uniqueness). The faces just defined are sometimes called c-faces, because they are connected. Not all schemes use c-faces. For example, the PADL-1 system used so-called b-faces which could have overlaps, and PADL-2 used m-faces, which need not be connected.

Faces of curved objects are represented as discussed in Chapter 4. Typically this involves a representation for a host surface or patch, a set of bounding edges, and neighborhood information. Host surface representation involves additional metric information (or "geometry") beyond vertex coordinates.

Edges also must be defined precisely. Typical BRep schemes use edges that satisfy the following conditions.

1. An edge is a subset of a face's boundary.
2. The union of all the edges associated with a face equals the face's boundary.
3. Each edge is a subset of the intersection of two primitive surface instances.
4. An edge is a compact, connected 1-manifold. (Some schemes require that an edge be homeomorphic to a line segment.)
5. Edges are quasi-disjoint, i.e., meet only at vertices.
6. An edge is the largest subset of the boundary of a face that satisfies all the previous conditions.

Often the intersection of two primitive surfaces crosses itself and therefore is not a 1-manifold. Then, the intersection must be segmented to satisfy the conditions just listed.

Normally, edges are parameterized and represented as explained in Chapter 4. Approximations, for example by splines, are often used. This can cause delicate numerical problems, because points on the edge do not necessarily lie in the surfaces whose intersection creates the edge. Approximate representations in the (u,v) parametric spaces of the surfaces involved are even more pernicious, because each edge is represented twice, and the two representations do not coincide exactly.

Many BReps contain additional information that is redundant but important for algorithm efficiency. For example, face normals may be stored, or edges may have back pointers to the faces which share them. Additional links are used to facilitate graph traversal. An example of a heavily linked BRep is the *winged edge* structure due to Baumgart in the mid 1970s, which is the intellectual root of many of the data structures in use today.

We describe the basic winged edge scheme for manifold objects with the help of the example of Figure 5.2.1.3. Edges are oriented by (arbitrarily) selecting for each a start vertex and an end vertex. Edges are depicted in the figure as vectors, to indicate their orientation. By convention, we orient each face clockwise, as seen by an observer outside the solid. The orientation of faces f_1 and f_2 are shown by the curved arrows in the figure. Since the solid is a manifold, each edge will belong exactly to two faces. In one of these faces, called the clockwise face (*cw*), the edge orientation agrees with the face orientation. In the other adjacent face, called the counter-clockwise face (*ccw*), the edge and face orientations are opposite. Therefore, the *cw* face of e_1 is f_1, and its *ccw* face is f_2. Following the orientation of f_1, the next edge to e_1 in its *cw* face is e_2. This edge is called the *next clockwise* edge (*ncw*) of e_1. Similarly, the *next counter-clockwise* edge (*nccw*) of

e_1 in its *ccw* face f_2 is e_4. Each edge in the winged edge structure points to its *ncw* and *nccw* edges. It also points to its two adjacent faces (not shown in Figure 5.2.1.4).

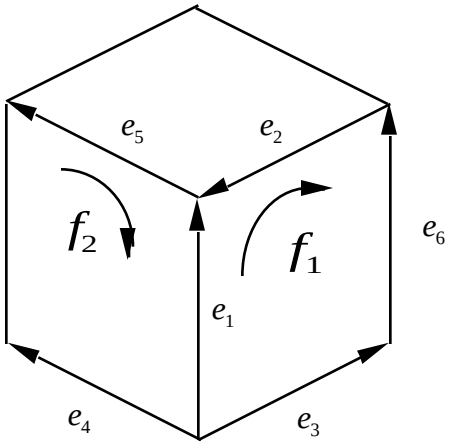

Figure 5.2.1.3 – Edge and face orientations for a cube.

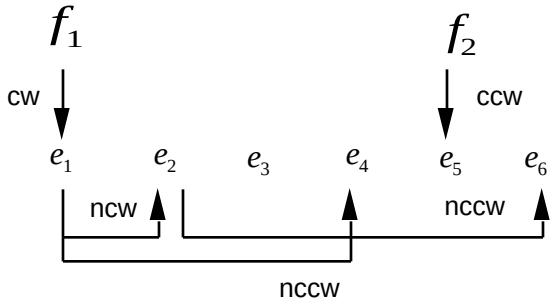

Figure 5.2.1.4 – A portion of the winged edge structure for a cube.

A face in the winged edge structure points to only one of its edges, as shown in Figure 5.2.1.4. The face-edge pointer is labelled to indicate if the face is the *cw* or *ccw* face of the edge. This information is sufficient to efficiently retrieve all the edges in a face. For example, start with edge e_1 of face f_1. The next edge, e_2, of f_1 is found by following the *ncw* pointer of e_1. To get the next edge, observe that f_1 is the *ccw* face of e_2. Therefore, we follow the *nccw* pointer of e_2 and obtain e_6. Continuing this procedure we get all the edges of f_1 in their correct order. Winged edge structures are attractive because they are relatively concise, and provide efficient means for traversing the boundary by following pointers.

The winged edge structure was designed originally for manifold objects and for simply-connected faces (i.e., faces without holes). It can easily be generalized to multiply-connected faces by introducing in the structure another level that corresponds to the separate edge loops that bound the faces. It can also be generalized to non-manifold objects, although it becomes considerably more complicated.

5.2.2 Validity of BReps

The validity of BReps is a complicated issue. For concreteness and simplicity, we will consider a specific representation scheme: boundary triangulation for polyhedral objects. (The discussion is easy to generalize to other BRep schemes.) The domain of the scheme is the set of all polyhedral r-sets. The faces are triangles and the edges are line segments. In addition, faces have the face properties 1-7, and edges have the edge properties 1-6 of Section 5.2.1. This BRep scheme is a simplified version of the scheme illustrated in Figure 5.2.1.1. To convert the representation in the figure to the triangulation scheme, the base of the pyramid would have to be split into two triangles by introducing an additional edge. The validity conditions for the boundary triangulation scheme are the following:

1. Each face must have exactly 3 edges, otherwise it will not be a triangle.
2. Each edge must have exactly two vertices, otherwise it will not be a line segment.
3. The edges associated with a face must form a loop or closed circuit, to ensure that they enclose a 2-D area. This condition is satisfied if and only if each vertex in a face belongs exactly to two of the face's edges.
4. The faces must form one or more closed surfaces or *shells*, to ensure that they enclose a 3-D volume. This condition is satisfied if and only if each edge belongs to an even number of faces. (Had we restricted the domain to manifold polyhedra, an edge would have to belong exactly to two faces.)
5. Each 3-tuple of coordinates must correspond to a distinct point in 3-space.
6. Edges must either be disjoint or intersect at a common vertex, otherwise there would be missing vertices in the representation.
7. Similarly, faces must either be disjoint or intersect at a common edge or vertex.

These conditions are easy to establish intuitively, and can be derived mathematically. Conditions 1-4 are *combinatorial*. They are easy to check algorithmically by counting nodes or links in the boundary graph. In contrast, conditions 5-7 are *metric*, i.e., they involve coordinates of vertices and equations of lines and planes. They are computationally expensive to check, because they require intersection tests.

We conclude that validity checking for BReps is not computationally attractive, and should be avoided. Most geometric modeling systems attempt to imbed the required validity conditions in the algorithms used to construct the representations, instead of testing representational validity after the BReps are built. In the next section we discuss a set of constructors that help ensure BRep validity.

Some modeling systems also provide users with operations, sometimes called *tweaking*, that manipulate boundaries directly. For example, in Figure 5.2.2.1 a sharp edge is *chamfered*, i.e., replaced by a face. This operation can be implemented by Boolean subtraction, but many BRep systems strive for higher processing speed, and simply modify the boundary graph directly, by adding a new face, with associated edges and vertices. The mathematical meaning of tweaking operations is not always well defined, and many systems do not ensure that such operations produce valid solids.

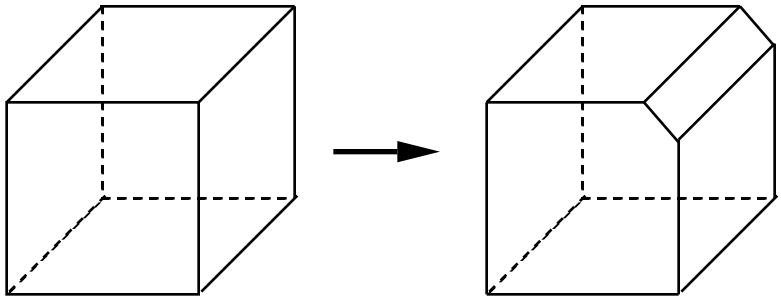

Figure 5.2.2.1 – A chamfering operation.

5.2.3 Euler Operators

The *Euler operators* for BRep manipulation were introduced in the geometric modeling literature by Baumgart in the mid 1970s. They were originally used in mathematics textbooks to prove *Euler's theorem*, which states that the Euler characteristic χ of a closed, connected, compact, orientable 2-manifold satisfies the equation

$$\chi = f - e + v = 2 - 2h,$$

where f, e and v denote the number of faces, edges and vertices, and h is the number of holes or handles of the manifold. This expression assumes that faces are simply connected, i.e., have no holes. If there are s connected surfaces, or *shells*, Euler's theorem can be applied to each connected component:

$$\begin{aligned} f_1 - e_1 + v_1 &= 2 - 2h_1 \\ f_2 - e_2 + v_2 &= 2 - 2h_2 \\ &\vdots \end{aligned}$$

Summing all of these expressions yields

$$f - e + v = 2s - 2h,$$

where f, e, v and h denote total numbers. The Euler characteristic is a topological property of the surface. It is invariant under homeomorphisms, independent of how the surface is decomposed (under mild conditions that decompositions must satisfy), and depends only on the number of shells and handles.

To discuss Euler operations we need to introduce a BRep scheme more general than the boundary triangulation scheme of the last section. Now faces and edges may be curved, and they are open, in the sense that a face does not include its bounding edges, and an edge does not include its bounding vertices. We require that open faces and edges be homeomorphic to open disks and open line segments, respectively. Otherwise, faces and edges have the properties listed in Section 5.2.1, just like their counterparts in the triangulation scheme. The new representation is called a *cell decomposition* of the boundary.

Euler operators manipulate cell decompositions. To describe what they do, it is convenient to draw a cell decomposition of a closed surface in the plane. The result is called a *Schlegel diagram*, and can be constructed as follows. Suppose that we have a cubical surface made of an elastic material. We can imagine a balloon in which we drew the edges of a cube. We cut a slit in the back face, between two points **a** and **b**, as shown in Figure 5.2.3.1, and then we open the surface and force it to lie in a plane.

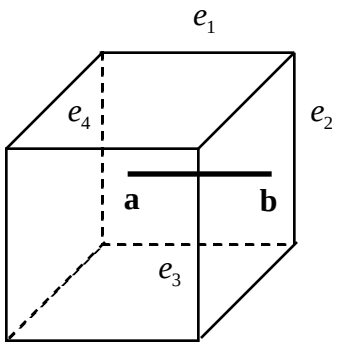

Figure 5.2.3.1 – The surface of a cube with a slitted back face.

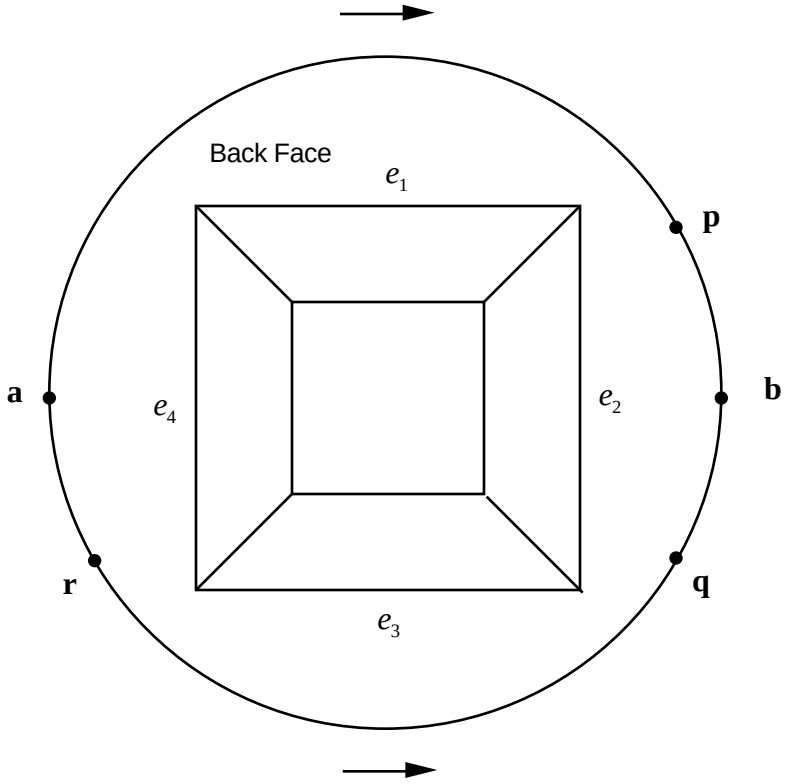

Figure 5.2.3.2 – Schlegel diagram for the surface of a cube.

The result is shown in Figure 5.2.3.2. Observe that the top and bottom curves *both* correspond to **ab**, and therefore must be *identified*. This means that points **p** and **q** in the figure are actually the *same* point. The arrows in the figure serve to indicate how point

correspondences are established. Thus, if the arrows were drawn in opposite directions, **p** would be identified with **r** instead of with **q**. The back face of the cube is mapped onto the region bounded by the two circular arcs and the four labelled edges. After the cut, we deformed the surface elastically, and therefore the planar diagram is homeomorphic to the surface with a slit. If we identify points as indicated above, the Schlegel diagram is homeomorphic to the *original*, uncut surface. The outer, circular edges ofen are not drawn in a Schlegel diagram, and the back face is then understood to be the remainder of the plane.

Now we will successively remove elements of this cell decomposition, as shown in the following figures. In Figure 5.2.3.3 we remove the bottom, front edge and merge the front and the bottom face. Note that the resulting, merged face is no longer planar. Intuitively, we can think of these operations as erasing lines drawn on a balloon. The operation illustrated in this figure is called *kfe*, for "kill face and edge".

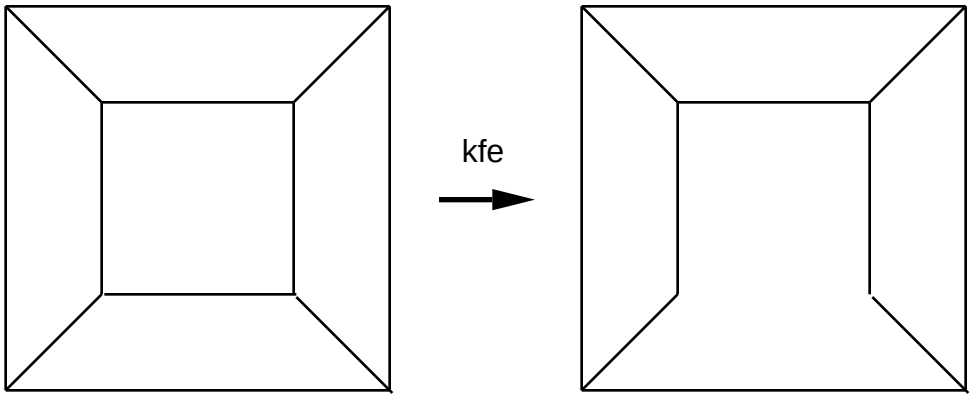

Figure 5.2.3.3 – Removal of an edge and a face

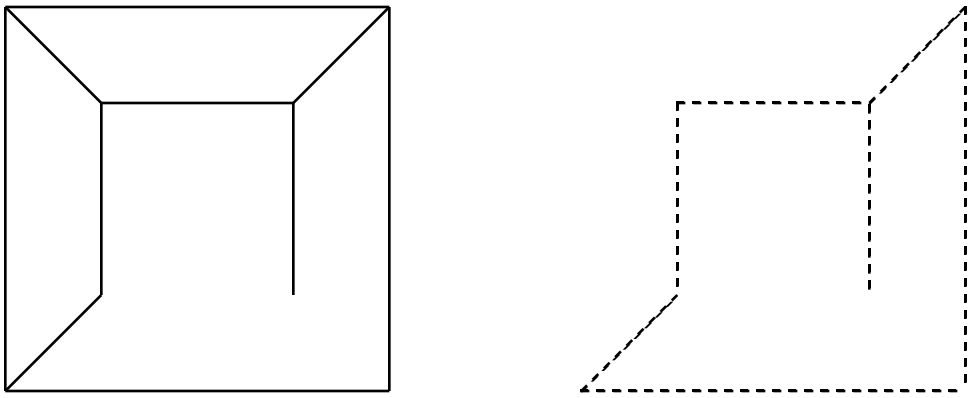

Figure 5.2.3.4 – The result of another *kfe* (left) and a resulting face (right).

Another application of *kfe* yields the result shown on the left, in Figure 5.2.3.4. The resulting cell decomposition contains an unusual face, shown on the right in the figure. The points on the dashed edges do not belong to the face. We can open up the face at the

vertical, intruding edge, and deform the face homeomeorphically into an open disk. Therefore it is a valid cell.

Two more *kfe* operations produce the result shown on the left of Figure 5.2.3.5. Now we need a new operation, termed *kev*, for "kill edge, vertex", which produces the decomposition shown on the right in the figure.

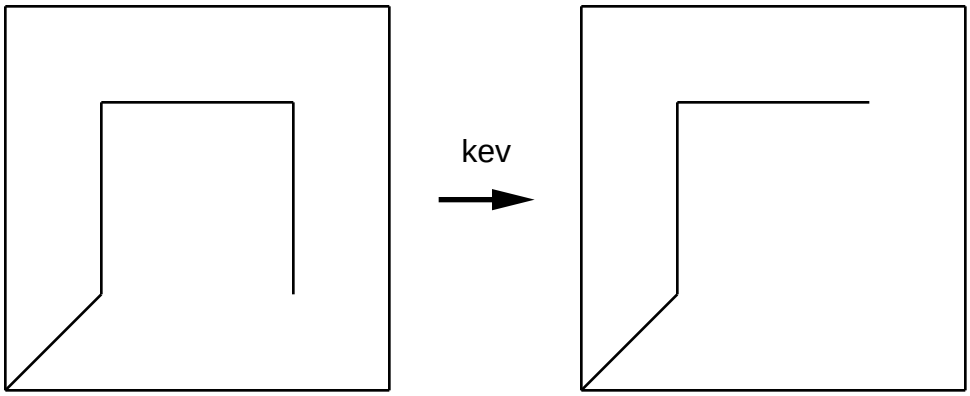

Figure 5.2.3.5 – Applying a *kev* operation.

Three more *kev* operations produce the result shown on the left of Figure 5.2.3.6. Observe that we are drawing explicitly the back face, to emphasize that there are two faces in the decomposition. Applying *kfe* we produce a decomposition with a single face, as shown on the right in the figure.

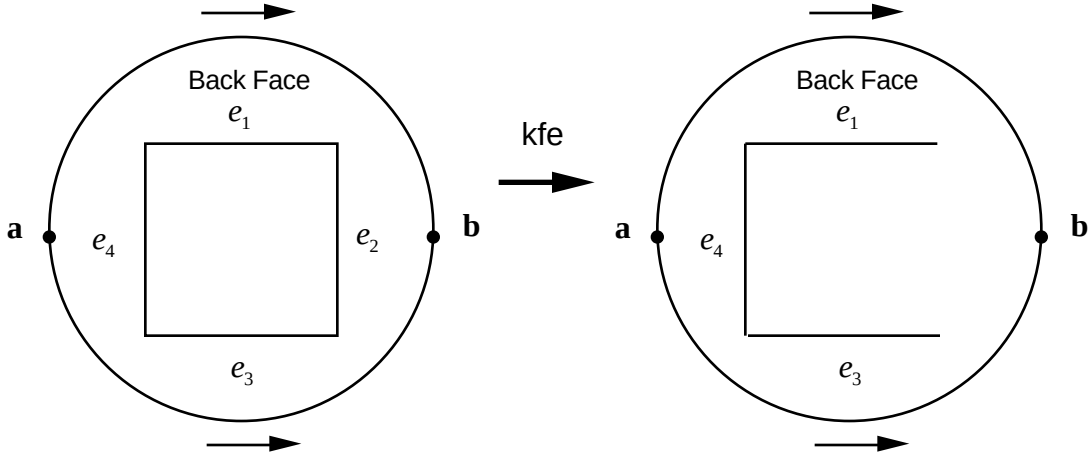

Figure 5.2.3.6 – Applying *kfe* to end out with a single face.

Now we remove two more edges and vertices via *kev* operations and generate the decomposition shown on the left of Figure 5.2.3.7. Here we show explicitly the two vertices associated with the remaining edge.

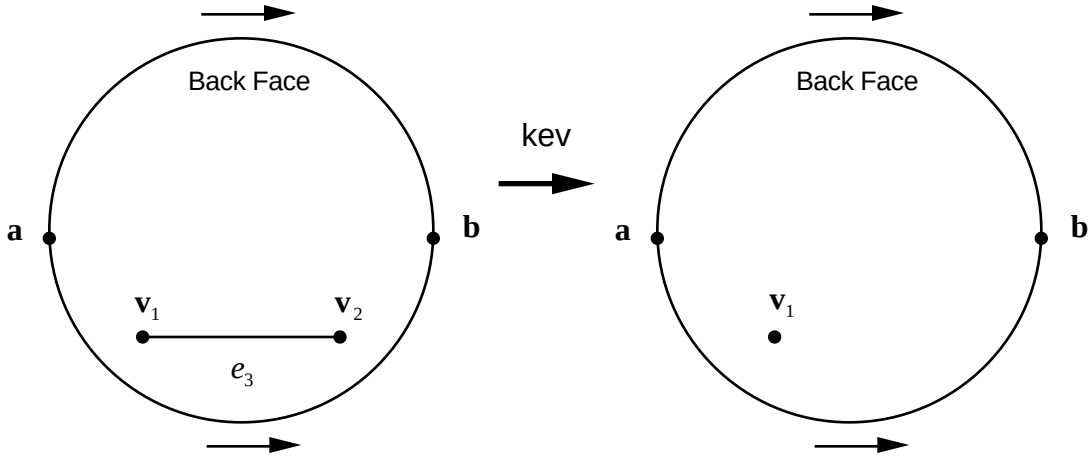

Figure 5.2.3.7 – Generating a single face and single vertex decomposition.

If we remove this edge and one vertex, via *kev*, we end out with a decomposition that contains only one face and one vertex, as shown on the right. This diagram corresponds to a spherical balloon in which only one vertex has been drawn. The face therefore is topologically equivalent to a sphere minus one point, which is a valid cell because it is homeomorphic to an open disk. To see that this is true, project a sphere onto a plane by using the North pole as the center of projection. Figure 5.2.3.8 illustrates the analogous procedure in 2-D. It is clear that points on the sphere map one-to-one onto points on the plane, and so do open sets. Therefore the mapping is a homeomorphism, and the sphere minus a point (the North pole) is topologically equivalent to the whole plane, which is itself equivalent to an open ball. If this last statement is not obvious, map the interval [0,1) through the function $f(x) = x/(1-x)$, which is a homeomeorphism since it establishes a one-to-one correspondence between points and also between open sets—see Figure 5.2.3.9. The result is a semi-infinite line starting at the origin. Now, if we apply a similar mapping in every direction emanating from the origin, the image of the unit open disk centered at the origin is the entire plane.

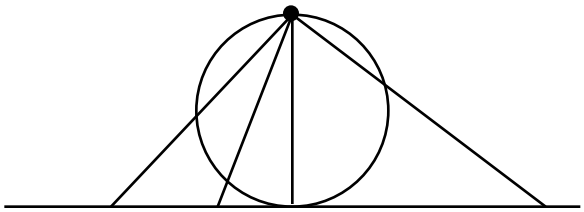

Figure 5.2.3.8 – Projecting a sphere on a plane using the North pole as the viewpoint.

Finally, we can make the single-face, single-vertex object didappear completely by invoking the *ksfv* operator, which kills a face a vertex and a shell, *i.e.*, a connected surface.

The destructive operators introduced above have constructive inverses. We reverse the destruction procedure described earlier, and build the cubic surface decomposition by beginning with the empty set and applying *msfv*, "make shell, face and vertex". Then we apply *mev*, "make edge and vertex", to obtain the decompositions shown in Figure 5.3.2.7. We continue with several applications of *mev* and *mfe*, "make face and edge" until we obtain the original decomposition.

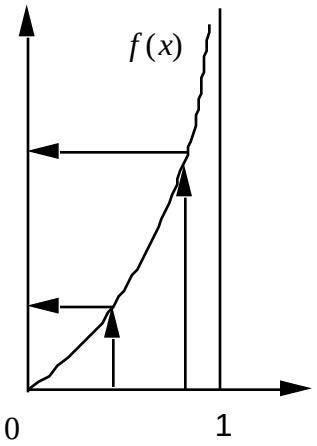

Figure 5.2.3.9 – Mapping a line segment into a semi-infinite line.

The *make* and *kill* operators are called *Euler operators* . Observe that the operators do not change the Euler characteristic of the surface. For example, *mfe* increases the number of faces *and* the number of edges by one. Thefore, the expression $\chi = f - e + v$ remains invariant. (It is this property that makes the operators useful in the proof of Euler's theorem.)

To be able to construct tori and other surfaces with holes we need an additional pair of operators. Consider the two blocks shown on the left in Figure 5.2.3.10. We can glue them as shown on the right in the figure. To do this we kill the bottom face of the top block (but keep the edges that bound that face), and introduce an extra edge on the top face of the bottom block. The extra edge is needed to ensure that the new (open) face of the joined cubes is homeomorphic to an open disk, and therefore is a valid cell. In addition, we change the number of shells from 2 to 1, which causes a decrease of 2 in the generalized Euler's formula $\chi = f - e + v = 2s - 2h$. Increasing the number of holes by one has the same effect on the Euler characteristic, and the operator is usually called *kfmeh*, for "kill face and make edge and hole", for reasons that will be apparent after we present below another example of its application.. It corresponds to a mathematical operation on manifolds called the *connected sum*, which glues two manifolds to produce another manifold.

The operator *kfmeh* can also be applied to a single shell, for example to connect two opposite faces of a cube, as shown in Figure 5.2.3.11. Initially we draw the two small shaded faces within the top and bottom face of the cube (with the necessary extra edges). Then we kill the small faces and connect the associated edge loops by a new vertical face, analogous to a cylinder. We need to add an extra edge to the new face to ensure that it is homeomorphic to an open disk. The outcome of the operation has one less face than the original, one more edge, and a through hole. In this case, *kfmeh* does precisely what its name implies.

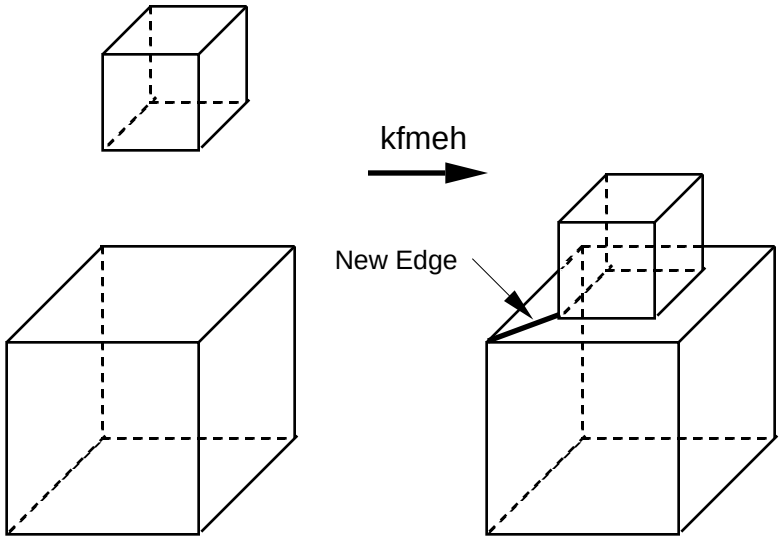

Figure 5.2.3.10 – Glueing two shells.

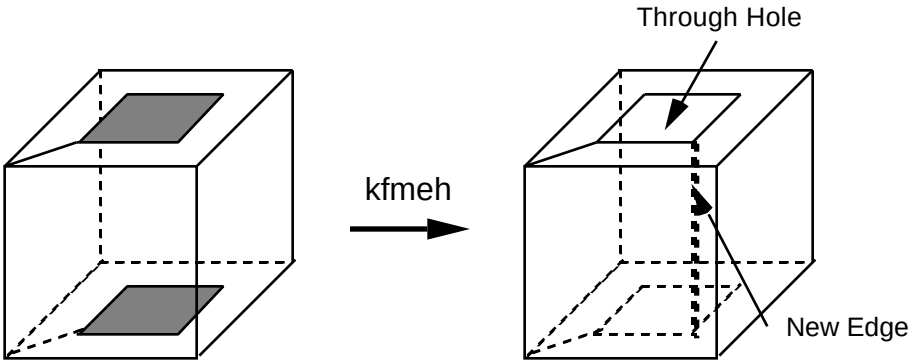

Figure 5.2.3.11 – Making a through hole.

The Euler operators provide convenient means for constructing complex BReps, and help ensure that they are valid. The early literature on geometric modeling stated erroneously that the Euler operators guarantee the validity of BReps. It is easy to see that this claim is false. Assume that the dimpled cube shown in Figure 5.2.3.12 was built trhough Euler operations. By simply changing the coordinates of one vertex we obtain the set of self-intersecting faces shown on the right, which does not correspond to a valid BRep. Therefore, a representation constructed with Euler operators may or may not be valid, depending on the specific values of the metric data assigned to the faces, edges and vertices.

The most important properties of Euler operators were established by Mäntylä [Mäntylä 1984]. He showed that it is always possible to assign metric data to a BRep constructed via Euler operators so that the BRep is valid in the domain of orientable manifold objects. Conversely, any valid manifold BRep can be constructed by a sequence of Euler operations. In essence, Euler operators ensure that the *combinatorial* conditions for validity

are satisfied. The resulting BRep may or may not be valid, depending on the metric data associated with it. Unfortunately, metric conditions are expensive to check.

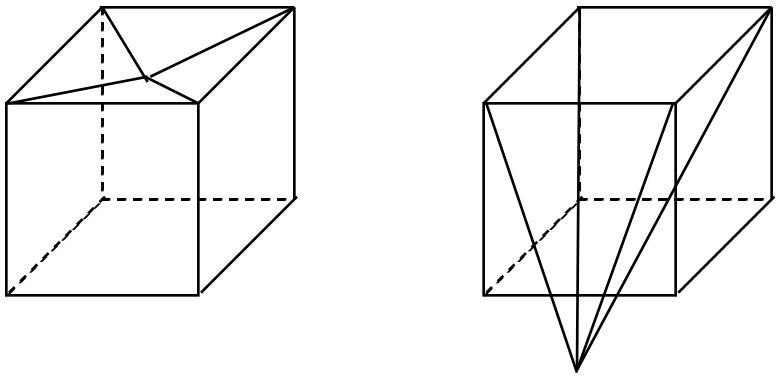

Figure 5.2.3.12 – Changing the coordinates of one vertex may invalidate a BRep.