

Lecture 21: B-rep and Boolean Operations ¹

B-rep

A solid can be represented with its boundary. *B-rep* is the short form of such representation. We assume that the solid is a manifold and thus its boundary is a 2-manifold. Such a solid can be represented with the *shells*, i.e., one or more components of the boundary. Each such shell is represented with its list of *faces* = f_1, f_2, \dots, f_k . Each face is a list of *edges* ordered along the boundary. Each edge is a list of two *vertices*. Thus, a solid structure would be defined as

$$\begin{aligned} \textit{solid} &= \{\textit{shell}_1, \dots, \textit{shell}_n\} \\ \textit{shell} &= \{\textit{face}_1, \dots, \textit{face}_k\} \\ \textit{face} &= \{\textit{edge}_1, \dots, \textit{edge}_r \\ &\quad \textit{face - equation}\} \\ \textit{edge} &= \{\textit{vertex}_1, \textit{vertex}_2 \\ &\quad \textit{orientation} \\ &\quad \textit{face}_1, \textit{face}_2\} \\ \textit{vertex} &= \{x, y, z\} \end{aligned}$$

Above definitions use pointer structure. For example, \textit{vertex}_1 and \textit{vertex}_2 in the edge structure are two pointers to the vertex structures. One can maintain more adjacency information. For example, we can maintain the pointers to the edges incident on a vertex.

Nonmanifold solids have nonmanifold boundaries. Thus they allow more than two faces adjacent to an edge. A generalization of the above definition can be used for nonmanifold solids.

Boolean operations

Set theoretic operations such as union, intersection and difference can be defined on solids if we consider them as a set of points. More complex solids can be generated from simpler ones using these boolean operations. We use $A \cap B$, $A \cup B$ and $A - B$ to denote the solids obtained as the intersection, union and difference of two solids A and B respectively.

Polygons

Let us first look at these operations in two dimensions between polygons. We assume that the boundary of the polygons are oriented such that inside lies on the right side of an edge (oriented). To compute the intersection $A \cap B$ of two polygons A and B , we intersect the line containing an edge of A with B for each edge of A . This generate all vertices of intersection. At each vertex two edges intersect to be segmented at the vertex as *inside edge* and *outside edge*. If we sort all the vertices of intersection on an edge together with its two end vertices, then consecutive segments

¹Note by Tamal K. Dey, Ohio State U.

connecting consecutive vertices are designated as inside and outside alternatingly. The first edge is taken as inside or outside depending on if the corresponding end vertex is inside or outside B .

Now the task is to connect the edges in a consistent manner to form the boundaries. We start from an inside edge and walk along its direction to reach an endpoint, say w . From w the walk proceeds on the other inside edge at w and it continues until the first edge is encountered. When the first inside edge is reached, we complete one boundary. But, there may be more than one boundary. So, we start another walk if there is any inside edge left. This procedure collects all boundaries on the intersection. The orientation of the boundaries are inherited from the oriented boundaries of the original.

If we wish to compute the union instead of intersection, we start the boundary collection from outside edges. For difference $A - B$, the designation of inside, outside edges are reversed on the edges on B and then we proceed as the union.

All of the above procedure assume non degeneracy, i.e., edges intersect at points other than their vertices. If degeneracies are allowed more care is necessary.

Polyhedra

In carrying out the boolean operations for polyhedra, we intersect the plane of each face of A with B . Let P be a plane of a face of A . The intersection of faces of B with P creates a set of edges which connect to form the boundaries of polygons of the intersection $P \cap B$. To compute the edges of these polygons, compute the line of intersection of P and the plane of the face of B being considered. Then, intersect this line with the face to generate the vertices of intersection. Again, we sort the vertices on the line of intersection. The inside edges are determined by designating the consecutive edges between consecutive vertices as inside and outside alternatingly. But, this time the first edge is an inside edge starting from a new generated vertex. After generating all such inside edges on all the faces of B we connect them to form the boundaries. We start at an inside edge and walk towards a vertex where we switch to another face of B and continue the walk another inside edge on that face and this continues. When we come back to the first edge, we complete one boundary and then start from another inside edge. After generating all the boundaries on the plane P , we orient them so that inside of the polygons lie on the rightside of the edges. In general, there will be a number of nested boundaries which have to be paired consistently. This nesting can be determined by a sweep algorithm.

After the polygons of intersection have been determined on P , we compute the intersection, union or difference depending on the operation sought for with the face polygon of A which lie on P . The new edges and vertices thus generated are the edges and vertices of the final object on the plane P . We do this for each plane containing faces of A and also the planes containing the faces of B . Finally, all new faces on each such plane are collected and proper adjacencies are recorded in the data structure.

Point inclusion

In modeling point membership is a very basic operation. Given a point p and a polygon F , determine if F contains p . This can be answered by shooting a ray from p to a point in F and extending it to compute its intersections with F . If the number of such intersection is even, then the p is outside and it is inside otherwise. For a polyhedron P , we can follow the same procedure.