Solid Modeling:to build polyhedrons and corresponding computations

Kai Cao

Oct 2012

1. Introduction

   Solid modeling is a consistent set of principles for mathematical and computer modeling of three-dimensional solids. Solid modeling is distinguished from related areas of geometric modeling and cmputer graphics by its emphasis on physical fidelity.Together, the principles of geometric and solid modeling form the foundation of computer-aided design and in general support the creation, exchange ,visualization, animation, interrogation, and annotation of digital models of physical objects.

2. What is included:

   (a) HalfEdge data structure recording the structure of solids like polyhedons. All the cpp files are in the core library. Every halfedge has its mate and we can find a "neighbor" of a face through the mate of a halfedge.Every halfedge belongs to a loop and every loop belongs to a face. A face contains at least a loop, which is its outer loop,and some inner loops wheneverthe solid contains a hole.

   (b) Computations on HalfEdge solid models:

      i. Check the relative postion of two solids.For Points, Lines,Planes and polyhedrons.
      We have different type of relative positions. For example, suppose we have two polyhedrons,then one can be inside, outside or intersect the other. That is not enough, we have to think deeply:suppose we have polyhedron A and B, if A is inside B, A can be "clearly" inside B without any intersection (even a point) with B,or can "touch" B. Here "touch" means that you cannot find a polyhedron inside the intersection of A and B, which means that the intersection of A and B can be a point , an edge and a face. And we have to think about "outside" situation based on the same classifier. The last situation is the most interesting, we have to find out whether the intersection should be "into","in-touch" or "cross". Obviously, "into" is like that A break into

B without breaking out.So for A we have two parts:the intersection and the remain. the "in-touch" situation is like "into" but A "inside-touch" B. And "cross" means that A was separated into three parts, an intersection and two parts remained. Relative position is the previous stage of computing the intersections. Because to compute the intersection of two polyhedrons, we have to decide the relative location of two polyhedrons,we have to find out whether B cuts A into two parts or three parts.

ii. Intersections This is the main part and the final implementation of our corresponding compuatations. Obviously, computing the intersection is not trivial. We will start from convex polyhedrons, which means that there is not any holes and voids and the intersection should be at most one polyhedron. Obviously intersections are more complecated than relative positions. Now what I am trying to do is to compute the intersection of a polygon and a polyhedron,which should be a polygon and a face of the intersection of the polyhedrons,let us call this polygon Q. Then we can generate the intersection polyhedron from that polygon,because other "nearby" faces of the intersection, which is a polygon, should have a common edge with Q. And this common edge should either be part of an edge of one polygon or the intersection with two polygons (not necessarily coplanar) respectively on A and B. In the solid modling structure, we can easily find the "mate" of a HalfEdge and find the face sharing the edge. So if we are facing the first situation, the next face R which should be used is the face containing the "mate". After that we can generate the "neighbor" by computing the intersection of R and B or going the othe other way compute the intersection of A and another face sharing an edge of B.

(c) Communication with the core library. To compute the intersections, we need to compute the 3-D coordinates which requires exact computation. So we would rather build a connection between our solid modeling geometric workbench (gwb) with the core library, especially the geom2d and geom3d programs. We will add intersection functions into geom2d and geom3ds and come up with the result in sympolic representations. We can draw the polyhedron in the gwb with machine number coordinates.