Introduction to Solid Modeling

Yogesh Kulkarni

February 1999

1 Introduction

- **What are models ?** - Model is an abstraction. It is an artificially constructed object for better understanding, observation.
- **What are the types ?** -
  - **Physical Models** - share relative dimensions, and general appearance.
  - **Molecule Models** - share relative arrangement of molecules with few other properties.
  - **Mathematical Models** - represents behaviour in terms of numerical data and equations.
  - **Engineering drawings** - convey 3D info with 2D representation.
  - **Computer Models** - use data stored in files.
- **Advantages** - Easier for study than the real counterpart as
  - object may not exist,
  - may not be observable,
  - may not be observable in controlled fashion.

2 Geometric Modeling

Its a type of Computer Model with aim to solve problems which are inherently *geometric*. Here, we try to represent the geometric shape of the object adequately by the equations. *Solid Modeling* is a branch of *Geometric Modeling* that emphasizes the general applicability of the models, and insists on creating only "complete" representation of physical solid objects, i.e. representation that are adequate for answering arbitrary geometric questions *algorithmically* (without interaction of user).

3 Requirements of Solid representation

- **Completeness** - Engineering drawing may not always get interpreted into meaningful object. Even 3D wireframe models may have several interpretations in terms of solidity. Completeness demands calculating answers to arbitrary geometric questions.
- **Integrity** - 3D Models are constructed from planes rather than lines. But this does not rule out possibility of self-intersecting polyhedral models which are physically invalid. Integrity avoids generation of incorrect models.
- **Complexity** - Relatively simple object should have relatively simpler representation. The difficulty of dealing with solid geometry increases with complexity of mathematical formulation used.
- **Computability** - Operations available in the Solid modeler should constitute a *closed system* where all the operations are guaranteed to maintain the correctness of the underlying model.e.g. Dual Entry System (debit-credit) in an account statement ensures balance at every entry.

4 Mathematical Models of the Solids

What is **"Solidity"** ?
We can have characterization based on two approaches, one stressing on the " three dimensional solidity" of things, and the other concentrating on the boundedness by surfaces.

4.1 Point-Set Models

Consider 3D Euclidian [1] Space E^3. A solid is a bounded,closed subset of E^3. This definition dose capture essence of "Solidity" but is far too large to be really useful in terms of computer storage.

- **Rigidity** - A rigid object is an equivalence class of point-set of E^3 spanned by following equivalance relation $\circ$: Let A, B be subsets of E^3. Then $A \circ B$ holds iff A can be mapped to B with a rigid transformation.
- **Regularity** - We expect a solid to be "all material"; it is not impossible that a single point, or a line or even a 2D area would be missing from a solid. Likewise, a "solid" point-set must not contain "isolated points","isolated items",or "isolated faces" that do not have any material around them.

  The regularization of a point-set $A, r(A)$, is defined by

$$r(A) = c(i(A)) \tag{1}$$

[1] The one that can be represented by 3 numbers

where $c(A)$ and $i(A)$ denote the closure and interior of A. Sets that satisfy $r(A) \equiv A$ are said to be regular[2]

- **Finiteness** - You can not store all the point-set data. Indirect but finite methods are adopted like a line is stored as two endpoints. Usually we must require that the surface of a solid is "smooth" [3]

4.2 Surface based Models - Plane Models

Its possible to characterize modeling spaces just based on properties of surfaces. The surface based characterization of the solids looks at the boundary of a solid object. The boundary is considered to consist of a collection of "faces" that are "glued" together so that they form a complete, closing "skin" around the object.

- **2-Manifolds** - A surface may be regarded as a subset of E^3 which is essentially "two-dimensional": every point of the surface is surrounded by a "two-dimensional" region of points belonging to the surface. A 2-manifold M is a topological space where every point has a neighbourhood topologically equivalent to an open disk of E^2. Surface of earth is a 2-manifold.
  Limitations of Manifold models - If a 2-manifold M and the boundary of A, $b(A)$, are topologically equivalent, we say that A is a **realization** of M in E^3. All objects are not "realizable". If the surface touches itself in a point or at a curve segment, the neighbourhood of such points is not a simple disk but more than one disks. So such objects are considered as the rigid combination of two components that just happen to touch each other at a point.

- **Plane Model of 2-Manifold** - Let P be a set of polygons, and let $a_1, a_2, ...$ be a collection of edges of these polygons. These edges are termed "identified"[4] when a new topology is defined on P as follows:

  - Each edge is assigned an orientation from one end point to the other with initial point at 0 and other point at 1.
  - The points on edges with same correspondig values are considered as single points.
  - The neighbourhood of P has disks entirely contained in a single polygon plus the union of half disks diameters are matching intervals around corresponding points on the edges.

  A plane model is a planer directed graph $\{N, A, R\}$ with finite number of vertices $N = \{n_1, n_2, ...\}$, edges $A = \{a_1, a_2, ...\}$ and polygons $R = \{r_1, r_2, ...\}$ bounded by edges and vertices. Each polygon has a certain orientation around its edges and vertices. Plane models are "realizable" if

[2]This tears off all "isolated" parts of a point set, covers it completely with a tight skin and fills the result up with the material

[3]Fractals are the exeptions as they are non-smooth planes and still have indirect, recursive representation.

[4]common

- Every edge is identified with exactly one other edge.
- For each collection of identified vertices, the polygons identified at that collection can be arranged in cycle such that each consecutive pair of polygons in the cycle is identified at an edge adjacent to a vertex from the collection.

Orientability - There are 2-Manifolds that do not have any physical counterpart in E^3 like klein bottle. These non-realizable 2-Manifolds are distinguished from realizable ones by the concept of "orientability".

A plane model is "orientable" if the directions of its polygons can be chosen so that for each pair of identical edges, one edge occurs in it positive orientation in the direction chosen for its polygon, and the other one in its negative direction. This condition is called **Möbius' rule**.

- **Euler Characteristics** - Let the surface S be given as a plane model and let v, e, and f denote the number of vertices, edges and faces in the plane model. Then the sum $v - e + f$ is a constant independent of the manner in which S is divided up to form the plane model. This constant is called "Euler Characteristics" of the surface and is denoted as $\chi(S)$. The theory of **homology** gives

$$\chi = h_0 - h_1 + h_2 \tag{2}$$

where, h_0, h_1, and h_2 are called the **Betti numbers**.

  - h_0 is number of connected pieces called "componentes". Hence of the cube. $h_0 = 1$. Also called s for "shells".[5]
  - h_1 is largest number of closed curves that can be drawn without cutting the solid into two or more parts. Hence of the cube, $h_1 = 0$ and doughnut has $h_1 = 2$. $h_1/2$ denotes h, the number of holes.
  - h_2 reveals the orientability of the surface and equals to h_0 for the kinds of surface we are looking for.

  Finally, we get

$$v - e + f = 2(s - h) \tag{3}$$

- **Surfaces with Boundary** - Sometimes its necessary to have surfaces that do not bound to get solid. In case of Cylinder, some edges (boundary) are not "identified" to some other edge. These edges form a boundaries of the surface; hence the term "Surfaces with Boundary". Theory of closed surfaces can be carried over to surfaces with boundary. To get cylinder, we have to cut two holes at corner of a elliptical shape and "mend" it to get cylinder. Its like "development" in case of sheet metals. Euler characteristics of the mended surface is

$$\chi = v - e + f + b \tag{4}$$

  Hence for a surface with b boundary components, the Euler-Poincaré formula can be written as

$$v - e + f = 2(s - h) - b \tag{5}$$

[5] Addition of a loop on the face makes a new shell.

5 Representation Schemes

- **Primitive Instancing** - Modeling using predefined libary parts. For instance, we might model mechanical components by generating a sufficiantly large collection of "Library" parts. Models are simply instances of them, created with an instantiation operation. Instantiation requires actaul values for parameters variables. Primitive Instancing is often used as an *auxiliary* scheme in modelers based on other representations in order to ease the description of often-needed parts e.g design of electrical instrumentaion or piping.

- **Exhaustive Enumeration** - Solid is supposed to be composed of say tiny cubes which are nonoverlapping and uniform in size and orientaion, i.e. they form *regular subdivision* of space. Exhaustive enumeration is little used as it is extremely expensive in memory terms, so space subdivision methods such as octree methods are more frequently used as they can result in a huge space saving at a small increase in algorithmic complexity. Consider a component roughly 200mm long by 100mm wide by 200mm high. If we were required to model this part to an accuracy of only 0.1mm If we use a single bit for each voxel then, at eight (8) bits to the byte we require approximately 500Mb of space, so exhaustive enumeration is not compact. Exhaustive Enumeration is used in modeling special applications, such as modeling of the cell particles.

- **Octree Representaion** - Recursive subdivision of space of interest into eight *octants* that are arranged into 8-ary In this octant numbering system, octant 0 is the $x, y, z > 0$ octant and the octants are numbered anticlockwise around the z axis, followed by those in the $z < 0$ halfspace. The octree representation then given is equivalent to the object shown. The initial, top level octants are centred on the origin. If we took our object $200x200x100mm^3$ then to model to an accuracy of 1 part in 2000 (or 0.1mm) we would need an octree up to 11 levels deep (212=2048). However, we would make great space savings with all the areas that were either entirely within the object or entirely without the object. One particular application of octtrees is for pre-segmenting a B-rep model to allow rapid ray casting of the model by photorealistic renderers. An octree is made of the B-rep model to some suitable resolution and the nodes of the octree are tagged with those faces of the model that can be found in the appropriate region. This allows rapid ray-firing as precise intersections need only be found for those nodes the ray passes through containing interesting surfaces.

- **Cell Decomposition** - Has a certain variety of basic cell types and a single operator *glue*. Indivisual cells are instances of parametrized cell types.

- **Half Space Models** - To start from sufficiently simple point sets that can be represented directly, and model other point sets in terms of very general combinations of the simple sets. A analytical function f of x, y, z can be defined in such a way that all points $X = (x, y, z), f(X) \geq 0$ are considered to belong to the point set, while $f(X) < 0$ defines its complement. Because $f(X) = 0$ divides the whole space into two subspaces, the two sets are defined as *half spaces*. Half Space models are

constructed by combining instances of half-spaces with *Bollean set operations* such as union ($\cup$), intersection ($\cap$) and set difference ($\setminus$).

- **Constructive Solid Model (CSG)** - Its easier to operate on bounded primitives than un-bounded half spaces. CSG modeler operates only on parametrized instances of *solid primitives* and boolean set operations on them.
  $< CSGtree > ::= < primitive > | < CSGtree >< setoperations >< CSGtree >$
  $| < CSGtree >< rigidmotion >$
  The regularized set operations $union^*, intersection^*$, and $setdifference^*$, denoted by $\cup^*, \cap^*$ and $\setminus^*$ are defined as

$$A \cup^* B = c(i(A \cup B)) \tag{6}$$

$$A \cap^* B = c(i(A \cap B)) \tag{7}$$

$$A \setminus^* B = c(i(A \setminus B)) \tag{8}$$

- **Boundary representaion** - Represents solid indirectly through a representation of its bounding surface.

6 Boundary Models

Boundary models are based on the surface-oriented view to solid modeling. They represent solid object by dividing its surface into a collection of *faces*. A face may have several bounding curves provided they define a connected object. In turn, the bounding curves of faces are represented through a division into *edges* which in turn are chalked out in terms of *vertices*.

6.1 Data Structure

The basic three object types *face,edge* and *vertex*, along with the constituent geometric information forms a boundary model. It also has information about how the faces, edges are related to each other, better known as "topology". Many choices as to what information is stored explicitly and what information is left implicit (i.e. to be computed as needed) must be made when implementing a bounadry model.

- **Polygon-Based Boundary Models** - Have only planer faces. A solid consists of a collection of faces. *Graphical metafiles* have this kind of information.

- **Vertex-Based Boundary Models** - Vertices are introduced as independant entries of the boundary data structure. Faces are represented in terms og global vertex identifiers.

- **Edge-Based Boundary Models** - If curved surfaces are present in a boundary model, it becomes useful to include also edge nodes explicitly in the boundary data structure to be able to strore information of intersection of curves of faces. Represents a face boundary in terms of a closing sequence of edges, or *loop* for short. Vertices of the faces are represented through edges.

- **Winged-Edge Data Structure** - Its elaborated edge-based geometry with introduction of *loop* information in edge nodes. Because each edge e appears in exactly two faces, exactly two other edges $e\prime$ and $e\prime\prime$ appear after e in these faces. So each edge has this new infomration of "next" edges, denoted by ncw and $nccw$ for "next clockwise" and "next counter clockwise"; in particular, ncw identifies the next edge in the face where the edge occurs in its positive orientation and $nccw$ the next edge in the other face. The full winged-edge data structure includes the identifiers fcw and $fccw$ of its neighbour faces, and analogously to ncw and $nccw$, the identifiers pcw and $pccw$ for previous edges. To include faces with several boundaries (holes) each face has list of loops.

6.2 Validity of Boundary Models

A boundary model is *valid* if it defines the boundary of "reasonable" solid object. This includes following conditions

- The set of faces of the boundary model "closes," i.e. forms the complete "skin" of the solid with no missing parts.(also called "topological integrity").
- Faces of the model do not intersect each other except at the common vertices or the edges.
- The boundaries of the faces are simple polygons that do not intersect themselves.

Unfortunately, the "geometric integrity" by above conditions can not be enforced just by structural means; by assigning inappropriate geometric data its possible to create invalid model with sound "topological integrity".

A Appendix

The concept of nearness [6] is fundamental in that it forms basis of defining other topological concepts.

- A set is **closed** if it contains all its near points. The **closure** of a set A denoted by $c(A)$, is the set of points near A.
- A set is **bounded** if it is contained in a neighbourhood.
- A set is **compact** if it is closed and bounded.
- A set is **open** if no point of A is near the complement of A.
- The **interior** of a set A, denoted by $i(A)$, is the set of points $\in A$ that are not near the complement of A.
- The **boundary** of a set A, denoted by $b(A)$,is the set of points that are near to both A and its complement.

[6] Point X is said to be near set A, if every neighbourhood of X contains A.

References

[1] Martti Mantyla. *Introduction to Solid Modeling*. Computer Science Press 1988, ISBN 07167-8015-1