13.472J/1.128J/2.158J/16.940J
COMPUTATIONAL GEOMETRY

Lecture 1

Nicholas M. Patrikalakis

Massachusetts Institute of Technology
Cambridge, MA 02139-4307, USA

Contents

Lecture 1

Introduction and classification of geometric modeling forms

1.1 Motivation

Geometric modeling deals with the mathematical representation of curves, surfaces, and solids necessary in the definition of complex physical or engineering objects. The associated field of *computational geometry* is concerned with the development, analysis, and computer implementation of algorithms encountered in geometric modeling. The objects we are concerned with in engineering range from the simple mechanical parts (machine elements) to complex sculptured objects such as ships, automobiles, airplanes, turbine and propeller blades, etc. Similarly, for the description of the physical environment we need to represent objects such as the ocean bottom as well as three-dimensional scalar or vector physical properties, such as salinity, temperature, velocities, chemical concentrations (possibly as a function of time as well).

Sculptured objects play a key role in engineering because the shape of such objects (e.g. for aircraft, ships and underwater vehicles) is designed in order to reduce drag or increase the thrust (eg. for propeller blades). At the same time these objects need to satisfy other design constraints to permit them to fulfill certain design requirements (e.g. carry a certain payload, be stable in perturbations, etc). Similarly, there are objects which have significant aesthetic requirements, eg. cars, yachts, consumer products.

Typically, engineers deal with the definition of complex shapes such as engines, automobiles, aircraft, ships, submarines, underwater robots, offshore platforms, etc. The shape of these objects is usually not fully known in advance (except when a baseline design is available). Consequently, the usual design procedure is iterative, involving:

- *Shape creation* based on certain design requirements;
- *Analysis* to evaluate the performance of the object; and,
- *Shape modification* to improve the shape, followed by analysis (and so on in an iterative loop) until a satisfactory (and in simple cases, an optimal) design is reached, which satisfies all the design requirements and minimizes a certain *cost* function.

Geometric modeling attempts to provide a complete, flexible, and unambiguous representation of the object, so that the shape of the object can be:

- Easily visualized (rendered)
- Easily modified (manipulated)
- Increased in complexity
- Converted to a model that can be analyzed computationally
- Manufactured and tested

Computer graphics is an important tool in this process as visualization and visual inspection of the object are fundamental parts of the design iteration. Computer graphics and geometric modeling have evolved into closely linked fields within the last 30 years, especially after the introduction of high-resolution graphics workstations, which are now pervasive in the engineering environment.

The remainder of this lecture introduces many of the different approaches to geometric modeling representations that have evolved over the last four decades.

1.2 Geometric modeling forms

Several different geometric modeling forms have evolved over the last forty years. For the definition of *model*, we can say that an abstract entity M is a model of an object O if M can be used to answer specific questions about O.

Different forms of geometric modeling can be distinguished based on exactly what is being represented, the amount and type of information directly available without derivation, and what other information can and cannot be derived.

1.2.1 Wireframe modeling

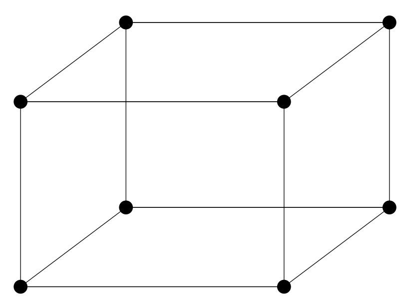

Figure 1.1: Wire frame model of a cube.

Wireframe modeling, developed in the early 1960's, is one of the earliest geometric modeling techniques. It represents objects by edge curves and vertices on the surface of the object, including the geometric equations of these entities (and also possibly but not always adjacency information), as shown in Figure 1.1. The traditional drawings of a ship's lines (Figure 1.2 [4]) is a form of a wireframe model of a ship hull. It is created by intersecting the hull surface with three sets of orthogonal planes. Usually the hull surface is taken as the molded hull surface which is the inner side of the hull plating. Intersections of the hull surface with vertical planes (from bow to stern) are called *buttock lines*. Intersections of the hull surface with horizontal planes (parallel to keel) are the *waterlines*, while intersections with transverse vertical planes are called *sections*. Wireframes are rather incomplete and possibly ambiguous representations that were superseded by surface models.

1.2.2 Surface modeling

Surface modeling techniques, developed in the late 1960's, go one step further than wireframe representations by also providing mathematical descriptions of the shape of the surfaces of objects, as shown in Figure 1.3.

Surface modeling techniques allow graphic display and numerical control machining of carefully constructed models, but usually offer few integrity checking features (e.g. closed volumes). The surfaces are not necessarily properly connected and there is no explicit connectivity information stored. These techniques are still used in areas where only the visual display is required, e.g. flight simulators.

1.2.3 Solid modeling

Solid modeling, first introduced in the early 1970's, explicitly or implicitly contains information about the *closure* and *connectivity* of the volumes of solid shapes. Solid modeling offers a number of advantages over previous wireframe and surface modeling techniques. In principle, it guarantees *closed* and *bounded* objects and provides a fairly *complete* description of an object modelled as a rigid solid in 3D space [7, 6, 8].

Figure 1.4 illustrates that for a *boundary based* solid model of a single homogeneous object, every surface boundary is always directly adjacent to one other surface boundary, guaranteeing a closed volume. Solid models, unlike surface models, enable a modeling system to distinguish the outside of a volume from the inside. This capability, in turn, allows integral property analysis for the determination of volume, center of volume or gravity, moments of inertia, etc.

An example is Baumgart's winged edge data structure [1, 2], where every edge has a start and end point, a face on either side, and at least two edges from each vertex bounding the faces. This information can be put in tabular form (perhaps using a relational database) or in a graph like data structure and used to ensure adjacency.

Typical solid modeling systems also offer tools for the creation and manipulation of complete solid shapes, while maintaining the integrity of the representations.

Solid modeling techniques exclude the two previous modeling forms (wireframe and surface modeling). The reason is that the solid modeling forms are traditionally constrained to work only with two-manifold solids.

In a *two-manifold solid representation*, every point on the surface has a neighborhood on the surface which is topologically equivalent to a two-dimensional disk. In other words, even though the surface exists in three dimensional space, it is topologically *flat* when the surface is examined closely in a small enough area around any given point, as illustrated by the cube in Figure 1.5.

1.2.4 Non-two-manifold modeling

Non-two-manifold modeling [1, 9, 5, 10] is a new modeling form which removes constraints associated with two-manifold solid modeling forms by embodying all of the capabilities of

Figure 1.2: Wire frame model of a ship hull.

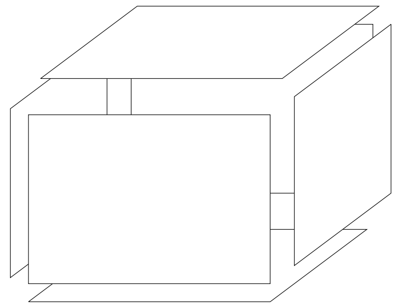

Figure 1.3: Surface model of a cube.

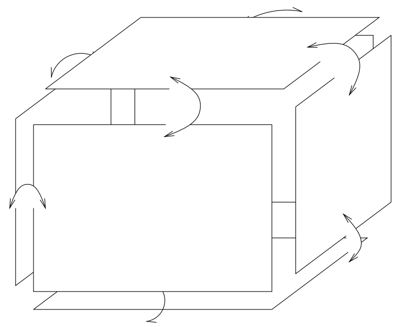

Figure 1.4: Solid model of a cube.

the previous three modeling forms in a unified representation. The following diagrams (in Figure 1.6) demonstrate non-two-manifold situations.

In an environment which allows non-two manifold situations, the surface area around a given point on a surface might not be *flat* in the sense that the neighborhood of the point need not be equivalent to a simple two-dimensional disk. This allows topological conditions such as a cone touching another surface at a single point, more than two faces meeting along a common edge, and wire edges emanating from a point on a surface. A non-two-manifold representation therefore allows a general wire frame mesh with surfaces and enclosed volumes embedded in space. Overall, non-two-manifold representations have superior flexibility, can represent a larger variety of objects, and can support a wider variety of applications than two-manifold representations, but at a cost of a *larger size* and *more complex* data structure.

Applications of the non-two-manifold representation include:

- Distinguish between two different solids, such as a beam welded to a plate (Figure 1.7).
- Represent a solid volume with a cutout and the volume that was cut out (Figure 1.8).
- Distinguish between the components of a composite plate (Figure 1.9).
- Represent a finite element mesh embedded in a solid object (Figure 1.10).
- Represent different dimensions simultaneously, such as a volume with a cut plane and an axis of revolution (Figure 1.11).

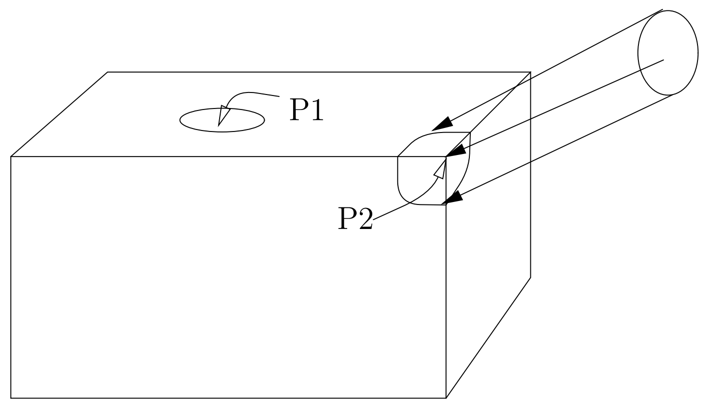

Figure 1.5: The cube is a two manifold object.

1.3 Basic classification of solid modeling methods

Current computer-aided design and manufacturing (CAD/CAM) systems used for solid object representation are generally based on three different types of modeling methods:

1. *Decomposition models* that represent solids in terms of a subdivision of space. - p.7
2. *Constructive models* that represent solids by Boolean (set) operations on primitive solids such as rectangular boxes, cylinders, spheres, cones, torii (appropriately sized, positioned and oriented). - p.14
3. *Boundary models* that represent solids in terms of their bounding faces, which are themselves bounded by edges and the edges by vertices. - p.16

A more detailed description of these models follows.

1.3.1 Decomposition models

Exhaustive enumeration

Exhaustive enumeration is a representation by means of cubes of uniform size, orientation, and which are nonoverlapping, see Figure 1.12. An object is represented by a three dimensional Boolean array. Each cell represents a cubic volume of space. If a cell intersects with the region of interest it has a true value. Otherwise, the value is false. This can be pictured as a box divided into 3D cubical pixels, with 0 assigned if empty and 1 assigned if full. This representation involves:

- Regular subdivision of space.
- It stores just one corner of each cube.
- For fixed space of interest we need just a 3-D array, C_{ijk} of binary data, and overall box/space coordinates:

$$C_{ijk} = \begin{cases} 1 & \text{if the cube } i,j,k \text{ intersects the solid} \\ 0 & \text{if the cube } i,j,k \text{ is empty} \end{cases} \tag{1.1}$$

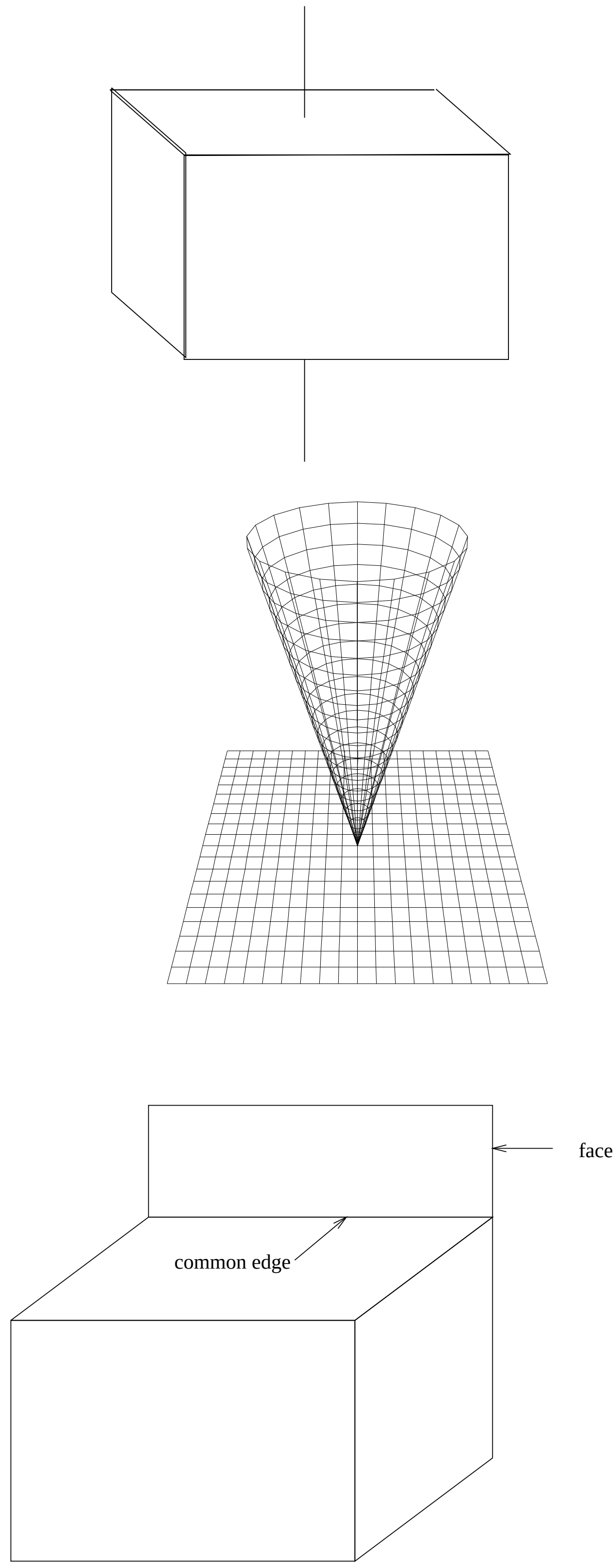

Figure 1.6: Examples of non-two manifold models.

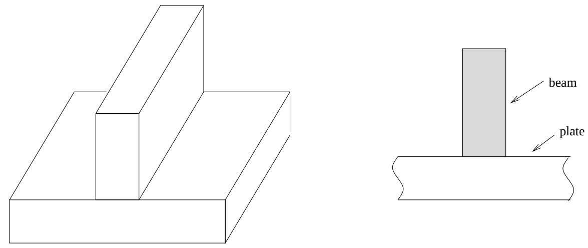

Figure 1.7: Beam welded to a plate.

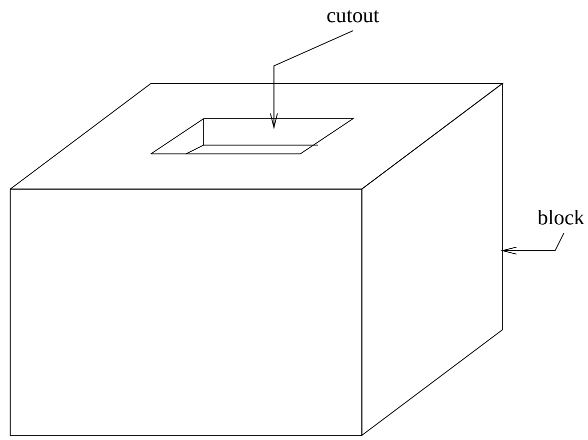

Figure 1.8: Block with cutout.

Applications of exhaustive enumeration methods include:

- Underwater environment representation.
- Finite elements meshing (first step in an algorithm to build such a mesh).
- Medical 3D data representation.
- Preprocessing representation for speeding up operations on other representations (eg. approximating integral properties such as volume, center of gravity, moments of inertia, distance transforms).

Properties of exhaustive enumeration methods include:

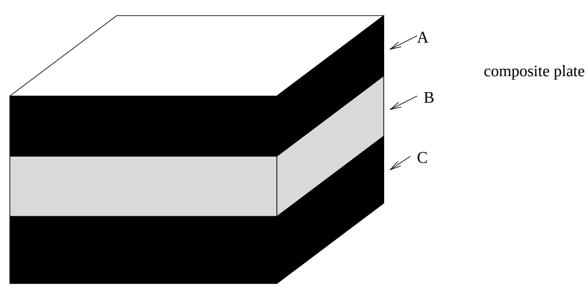

Figure 1.9: Composite plate.

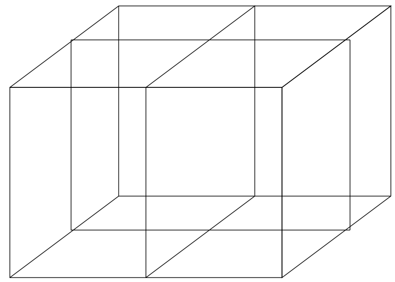

Figure 1.10: Finite element mesh.

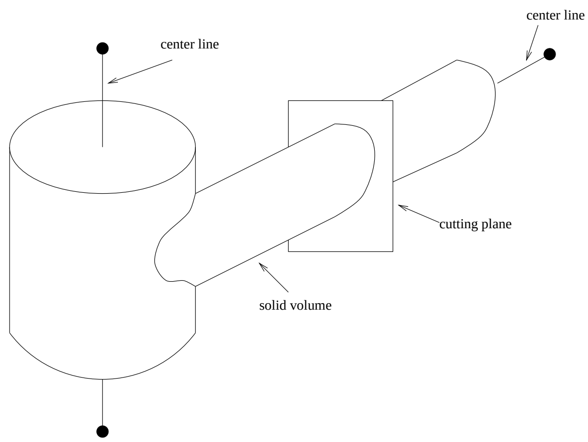

Figure 1.11: Representation of dimensions.

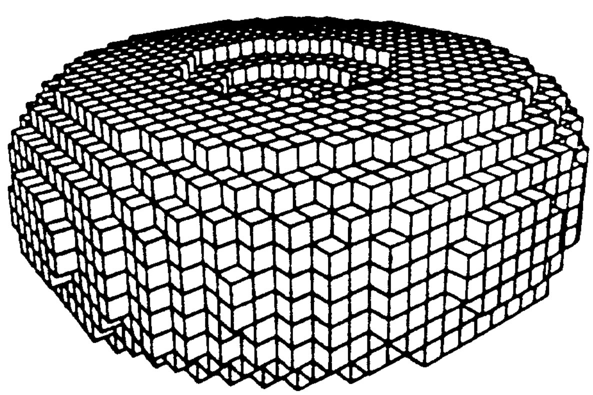

Figure 1.12: Exhaustive enumeration.

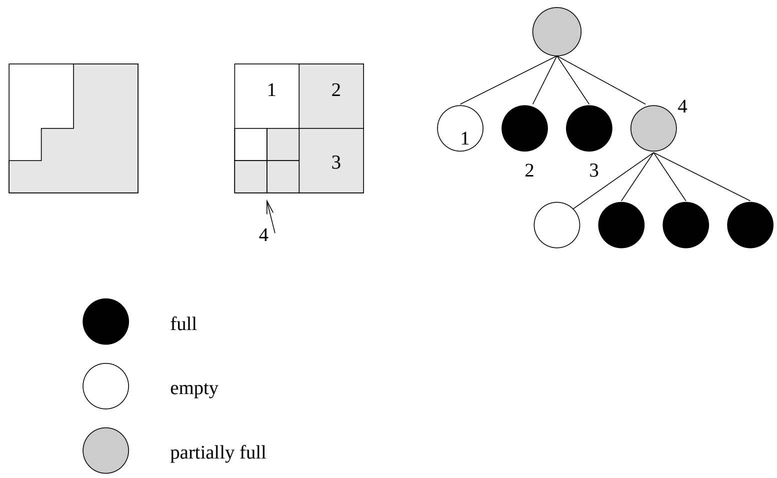

Figure 1.13: Quadtree representation.

- Expressive power: approximation scheme.
- Unambiguous and unique for fixed space and resolution. There do not exist different representations for the same object.
- Memory intensive: eg. $256^3 \rightarrow 16M$ bits and this is a bare minimum.
- Closure[1] of operations (eg. Booleans).
- Computational ease for algorithms: VLSI implementation for volume rendering. However, for high resolution the algorithm slows down.

Boundary cell enumeration

This is a boundary based version of the above technique. Only the cells that intersect region boundaries have true values.

Space subdivision

Some of the motivations behind space subdivision methods include:

- Smaller memory requirements if adaptive subdivision is used;
- Octree/quadtree representations lead to a recursive subdivision into 8 octants (or 4 quadrants) that can be represented as an 8-ary tree (or 4-ary tree).

In an octree representation a solid region is represented by hierarchically decomposing a usually cubic volume of space into successively smaller cubes (8 of them). Hierarchical division and cube orientation usually follows the spatial coordinate system. An example of quadtree, the two dimensional analogue, is shown Figure 1.13.

[1]Closure means that an operation such as Boolean results in an object of the same topological type that can be represented by the same type of data structure.

This is a trivial example. The method can continue to many more levels for a much more complex model. Some tolerance for minimum size blocks is required. In addition, this very concise representation would become very large if the coordinate system was changed; for example, rotated 45 degrees.

This method leads to a quick way to compute the area and other integral properties of a region. It is often used in data analysis in fields such as medical applications and sonar imaging.

To create an octree, we apply a classification procedure to a given solid (represented in Boundary Representation, Constructive Solid Geometry, Exhaustive Enumeration, etc.) and decide if a given node of the octree is:

- Exterior to solid (white);
- Interior to solid (black);
- Partially interior to solid (grey).

The classification procedure is used recursively. It could be based on Boolean solid operations, especially intersection. Figure 1.14 provides an example of octree representation.

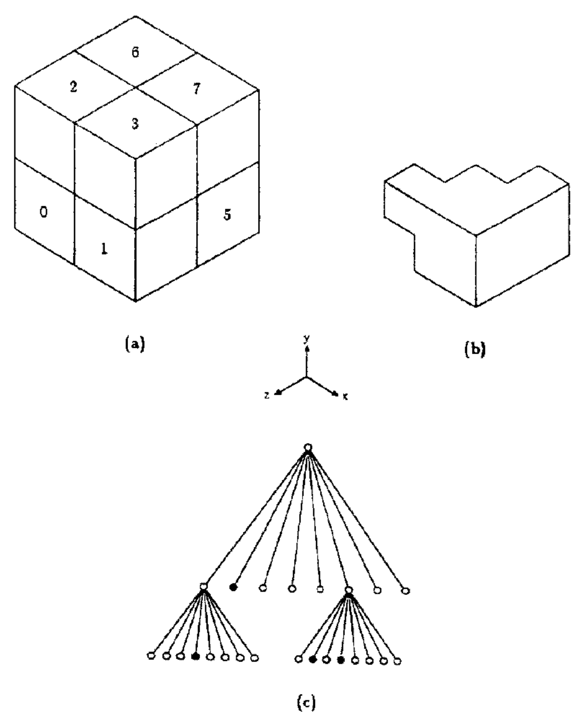

Figure 1.14: An octree model.

Some of the properties of octrees include:

- Expressive power: they are an approximation scheme;
- Validity: if no special connectivity is required, all octrees are valid;

- Unambiguous and unique: for a fixed resolution there is only one compacted[2] octree;
- Memory: not as large as Exhaustive Enumerations, yet much larger than Boundary Representation and Constructive Solid Geometry models;
- Closure of operations: for example Boolean operations and transformations;
- Computational ease: somewhat more complex than exhaustive enumeration.

Cell decompositions

The motivation for cell decomposition methods is:

- Use of elements other than cubes, see Figure 1.15 for an example.
- Application: finite element method, scientific visualization.
- Cells are parametrized instances of a generic cell type, eg. a cell bounded by quadratic curves and surfaces.
- Cells are homeomorphic to spheres.
- Cells meet at a vertex, edge, face otherwise the representation is invalid.
- Cells are disjoint and non-overlapping.
- Cells may belong to different cell types, eg. box-like, tetrahedra-like, etc.

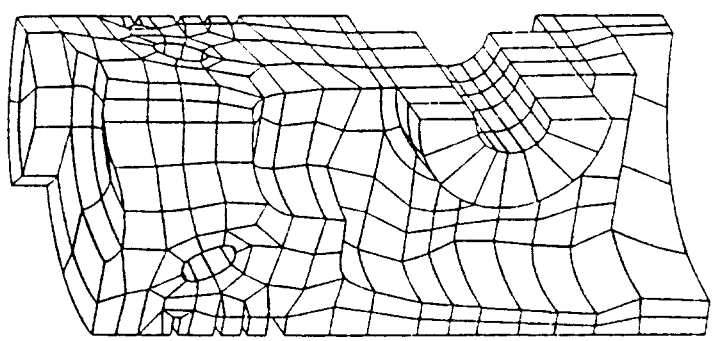

Figure 1.15: A cell decomposition (finite element mesh).

A cell decomposition can be represented using the *cell-tuple data structure* [3]. See Figure 1.16 for a 2D example.

The properties of cell decomposition methods are:

- Expressive power: very general and accurate;

[2]Algorithms such as set operations can create octrees with unnecessary nodes (eg. an internal nodes whose children are all black). Such nodes can be removed with a relatively simple tree traversal algorithm.

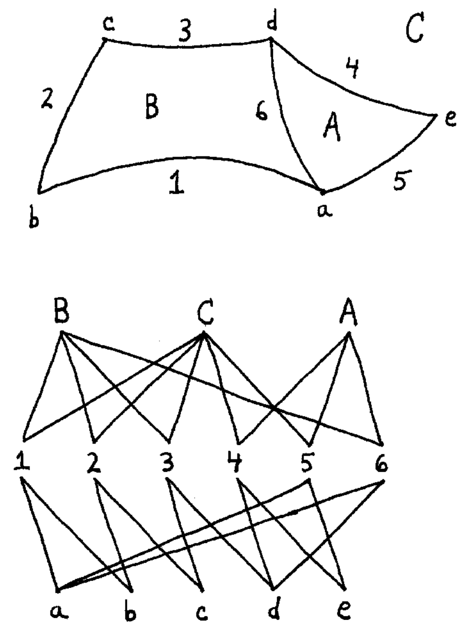

Figure 1.16: Cell data structure.

- Validity: requires an intersection test for verification;
- Unambiguous representation;
- Nonunique: Similar to the Constructive Solid Geometry method we will see below, the same object can be represented at different resolutions or with different types of mesh (eg. hexahedral, tetrahedral, etc.);
- Generation: by conversion from other representations;
- Concise: memory utilization is less than octrees, yet more than Boundary Representation;
- Applicability: finite element meshing, multimaterial non-homogeneous objects, visualization of fields, etc.

1.3.2 Constructive solid geometry (CSG)

Constructive Solid Geometry (CSG) is the Boolean combination of primitive volumes that include the surface and the interior. For example, primitives including rectangular box, sphere, cylinder, cone and torus can be combined using intersection, union and difference operators to form complex solids. Positioning operators (position, orientation) and size operators are applied to the primitives before the Boolean operators are invoked, see Figure 1.17 for an example.

Terminal nodes on the binary tree are primitive volumes; other nodes are Boolean operators. This representation has a direct manufacturing analogue, where difference indicates drilling or machining and union indicates for example welding.

Another example of a related representation is *sweeps*, where more general primitives are obtained by sweeping a solid along a space curve or sweeping a planar curve through a revolution

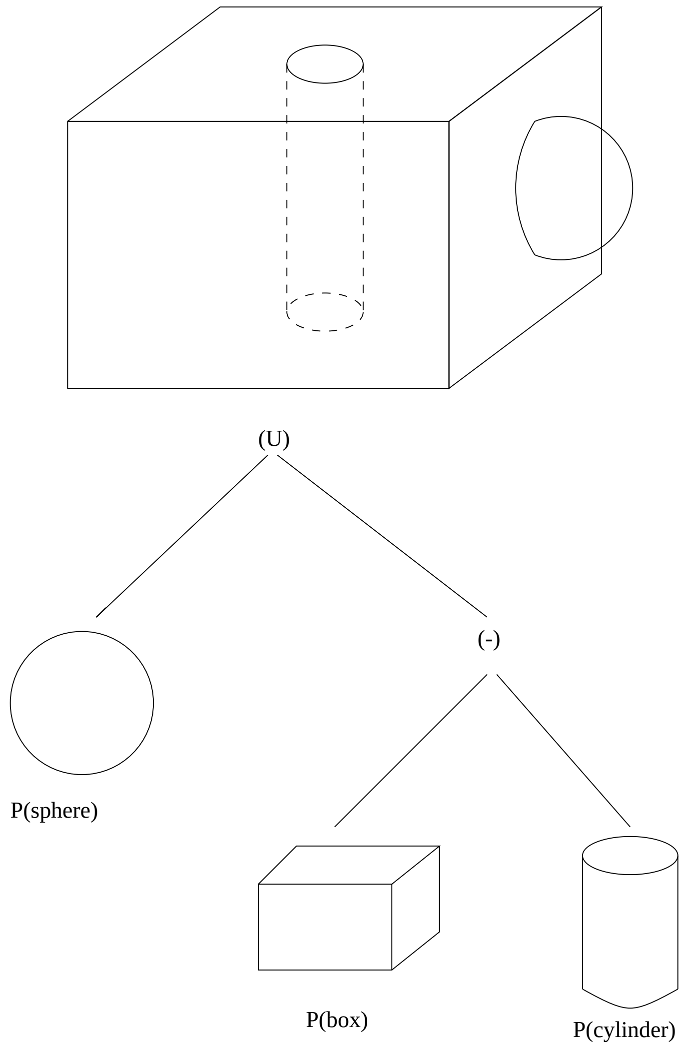

Figure 1.17: Boolean operations and primitives.

about an axis in its plane. Sweeps are useful in the representation of blends, volumes swept by machine tools, and in robotics.

In a survey of machine elements, 90 to 95% of parts could be represented accurately using the CSG method with the above simple primitive solids.

1.3.3 Boundary representation (B-Rep)

Objects are represented in terms of their boundary elements (e.g. vertices, edges, faces) which are related through *adjacency*.[‡] This is the most generally used representation today due to its flexibility. In these notes, we develop the theory of curves and surfaces which form the edges and faces of B-Rep models.

Figures 1.18 and 1.19 show an example of a tetrahedron and its B-Rep model.

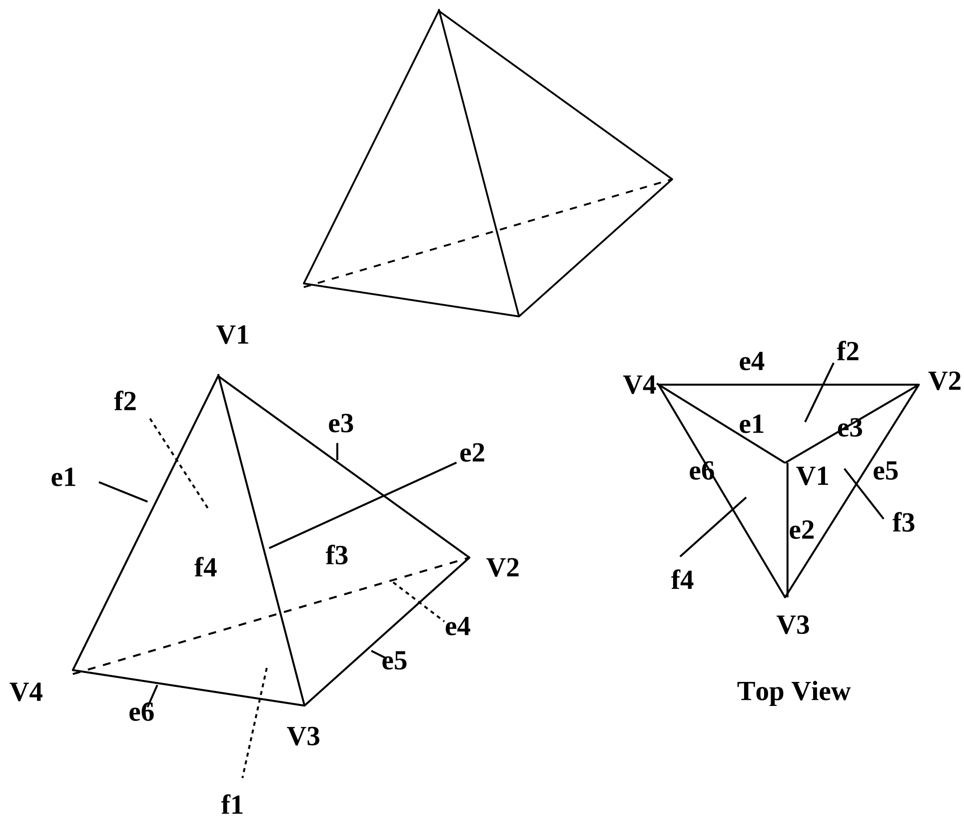

Figure 1.18: A tetrahedron

[‡]Two boundary elements, which are bounded by next lower dimension boundary elements, are called *adjacent*, if they share one common next lower dimension boundary element. For instance, two surfaces having a common edge are adjacent, or two edges having a common vertex are adjacent.

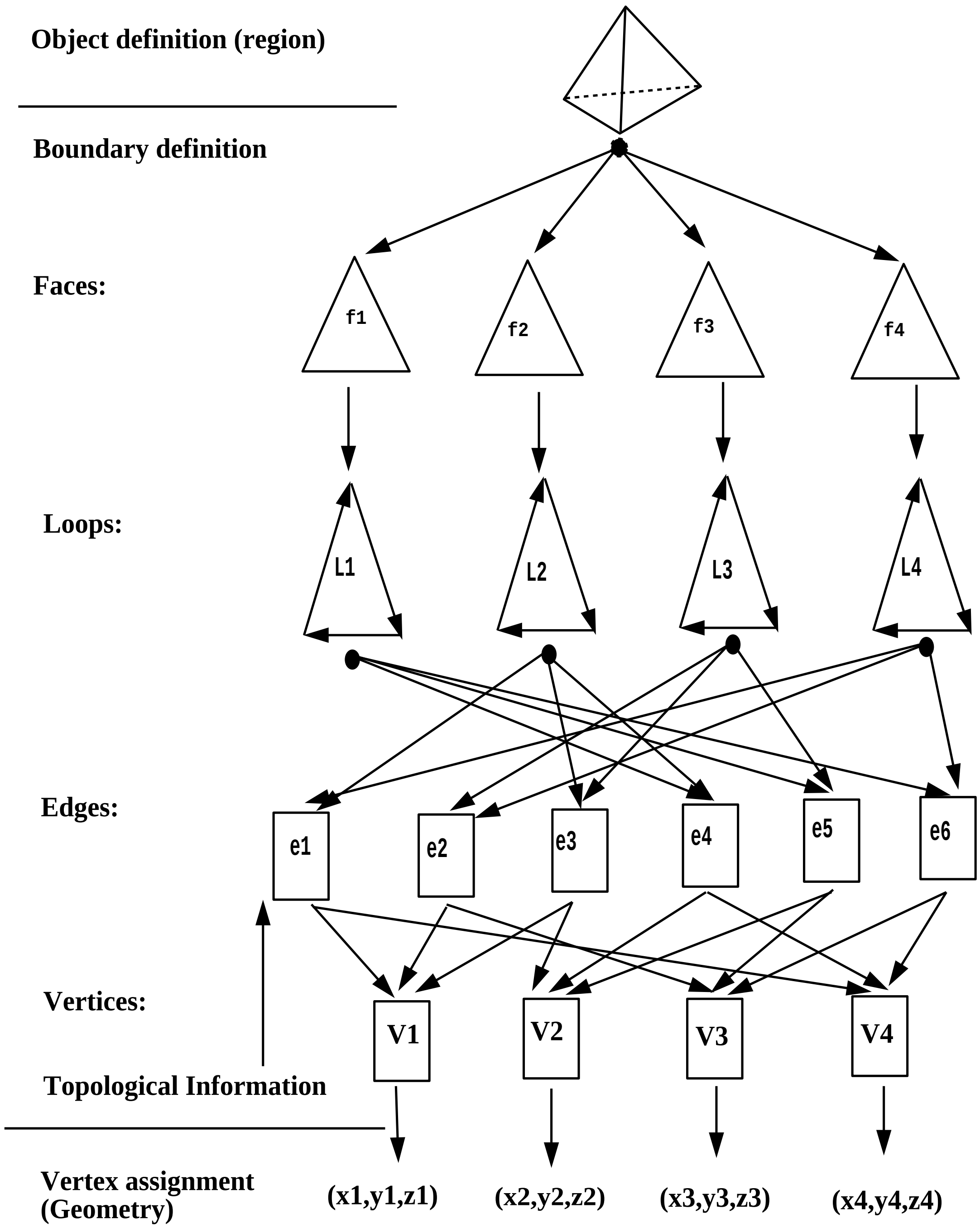

Figure 1.19: Boundary representation model for a tetrahedron

1.4 Alternate classification of geometric modeling forms

1.4.1 Introduction

The wide variety of representation techniques developed (many of which were identified above) can be differentiated on the basis of at least three independent criteria concerning the representation:

- boundary based or volume based
- object based or spatially based
- evaluated or unevaluated in form

A representation is *boundary based* if the solid volume is specified by its surface boundary. If the solid is specified directly by its volume it is *volume based*.

A representation is *object based* if it is fundamentally organized according to the characteristics of the actual geometric shape itself. It is *spatially based* when the representation is organized around the characteristics of the spatial coordinate system it uses.

Evaluated or *unevaluated* characterization is roughly a measure of the amount of work necessary to obtain information about the objects being represented with respect to a specific goal.

What is best depends on the application! A good system should support multiple representational techniques to ensure their efficiency over a broad range of applications.

We have three different criteria with two choices, so eight categories result. The following Table 1.1 gives examples in each category:

Unevaluated Class

	boundary based	volume based
spatial based	Half Space	Octree
object based	Euler Operators	CSG

Evaluated Class

	boundary based	volume based
spatial based	Boundary Cell Enumeration	Cell Enumeration
object based	Boundary Representation	Non-parametric Primitives

Table 1.1: Classification of geometric modeling forms

1.4.2 Unevaluated representation systems

Unevaluated representation systems require some form of procedural interpretation to be used with respect to the specified application.

Spatial, boundary based: half space technique A solid is represented by successively dividing space in half with usually infinite surface descriptions and selecting the half space on a specified side of the surface, eventually enclosing the solid region. The intersection of the half spaces represents the solid. Only convex regions can thus be described unless unions are also employed. Figure 1.20 demonstrates the half space technique in a two dimensional format.

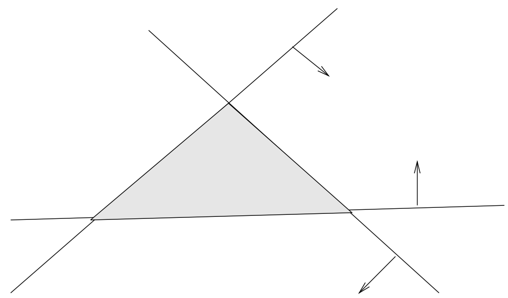

Figure 1.20: Half space technique of model representation.

This technique is classified as spatial based because the surface descriptions are positioned in spatial coordinate space rather than being relative to the object.

Spatial, volume based: octree representation A solid region is represented by hierarchically decomposing a usually cubic volume of space into successively smaller cubes (8 of them). Hierarchical division and cube orientation usually follows the spatial coordinate system.

Object, boundary based: Euler operators The object is procedurally described as a sequence of "Euler Operators," as in Figure 1.21. An (amorphous) topological sphere is topologically modified using the Euler Operators such as:

- msv = make shell, vertex
- mev = make edge, vertex
- mef = make edge, face

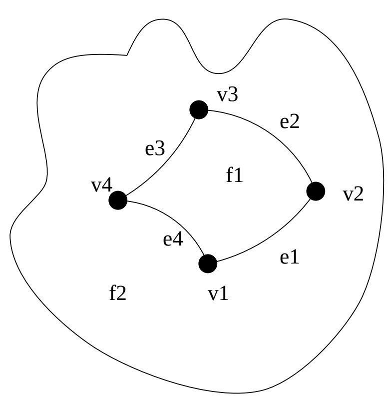

Figure 1.21: Boundary based representation Using Euler operations.

These operators ensure that Euler's Formula is always satisfied:

$$V - E + F - L_i = 2(S - G)$$

where:

- V = number of vertices
- E = number of edges
- F = number of faces
- L_i = number of internal loops
- S = number of surfaces (shells)
- G = genus (number of handles or through holes)

Object, volume based: constructive solid geometry (CSG) Constructive Solid Geometry (CSG) is the Boolean combination of primitive volumes that include the surface and the interior. For example, a box, sphere, cylinder, torus and cone can be combined using intersection, union and difference operators to form many surfaces. In addition, positioning operators such as position, orientation and size are applied to the primitives before the Boolean operators are applied.

Another example of an object, volume based representation is sweeps, where more general primitives are obtained by sweeping a solid along a space curve or sweeping a planar curve through a revolution.

1.4.3 Evaluated representation systems

Evaluated representation systems usually require substantially less computation to be useful in specific applications.

Spatial, volume based: cell enumeration An object is represented by a three dimensional Boolean array. Each cell represents a cubic volume of space. If a cell intersects with the region of interest it has a true value. Otherwise, the value is false. This can be pictured as a box divided into pixels, with 0 assigned if empty and 1 assigned if full.

Spatial, boundary based: boundary cell enumeration This is a boundary based version of the above technique. Only the cells which intersect region boundaries have true values.

Object, Boundary Based: Boundary Representation (b-rep) Objects are represented in terms of their boundary elements (e.g. vertices, edges, faces) which are related through incidence and adjacency. This is the most generally used representation today, and will be discussed in detail in further lectures.

Object, volume based: non-parametric primitives Simple fixed position objects. This is not a particularly flexible representation.

Bibliography

[1] L. Bardis and N. M. Patrikalakis. Topological structures for generalized boundary representations. MITSG 94-22, MIT Sea Grant College Program, Cambridge, Massachusetts, September 1994.

[2] B. G. Baumgart. Winged edge polyhedron representation. Technical Report STAN-CS-320, Computer Science Department, Stanford University, 1972.

[3] E. Brisson. Representing geometric structures in d dimensions: Topology and order. *Discrete and Computational Geometry*, 9:387–426, 1993.

[4] J. P. Comstock, editor. *Principles of Naval Architecture*. The Society of Naval Architects and Marine Engineers, New York, 1967.

[5] E. L. Gürsöz and F. B. Prinz. Node-based representation of non-manifold surface boundaries in geometric modeling. In M. Wozny, J. Turner, and K. Preiss, editors, *Geometric Modeling for Product Engineering*. North-Holland, 1989.

[6] C. M. Hoffmann. *Geometric and Solid Modeling: An Introduction*. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1989.

[7] M. Mäntylä. *An Introduction to Solid Modeling*. Computer Science Press, Rockville, Maryland, 1988.

[8] M. E. Mortenson. *Geometric Modeling*. John Wiley and Sons, New York, 1985.

[9] J. Rossignac and M. O'Connor. SGC: A dimension-independent model for pointsets with internal structures and incomplete boundaries. In M. Wosny, J. Turner, and K. Preiss, editors, *Geometric Modeling for Product Engineering*, pages 145–180. North-Holland, 1989.

[10] K. Weiler. Edge-based data structures for solid modeling in curved surface environments. *IEEE Computer Graphics and Applications*, 5(1):21–40, January 1985.