CSE328 Fundamentals of Computer Graphics: Concepts, Theory, Algorithms, and Applications

Hong Qin

Department of Computer Science

State University of New York at Stony Brook (Stony Brook University)

Stony Brook, New York 11794--4400

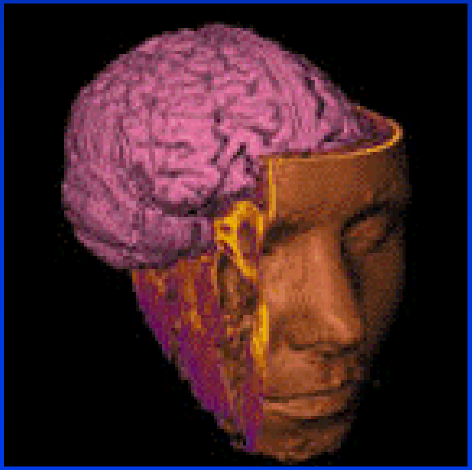
Tel: (631)632-8450; Fax: (631)632-8334

qin@cs.sunysb.edu

http://www.cs.sunysb.edu/~qin

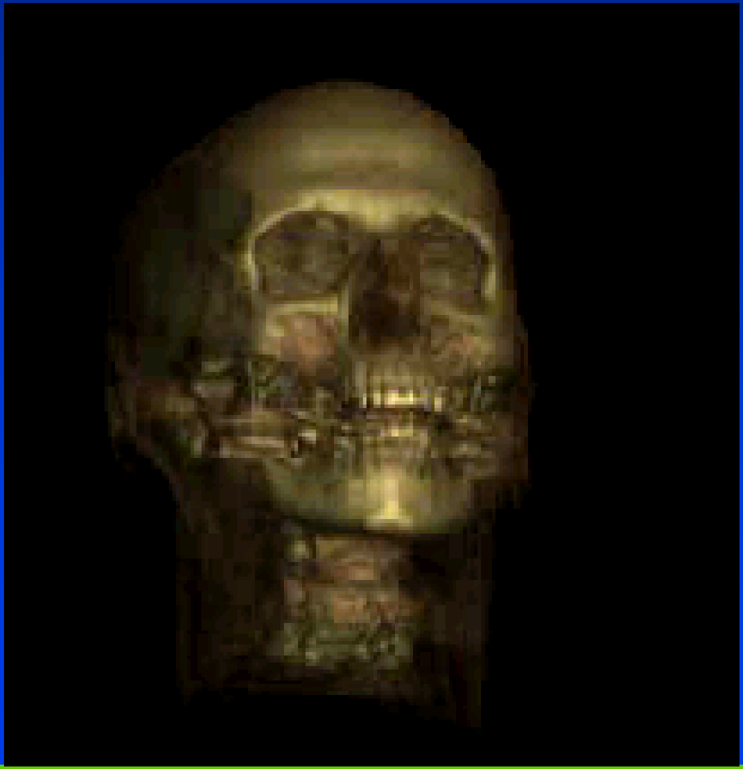
Solid Modeling Basics

- **Represent objects' solid interiors**
  - Surface may not be described explicitly



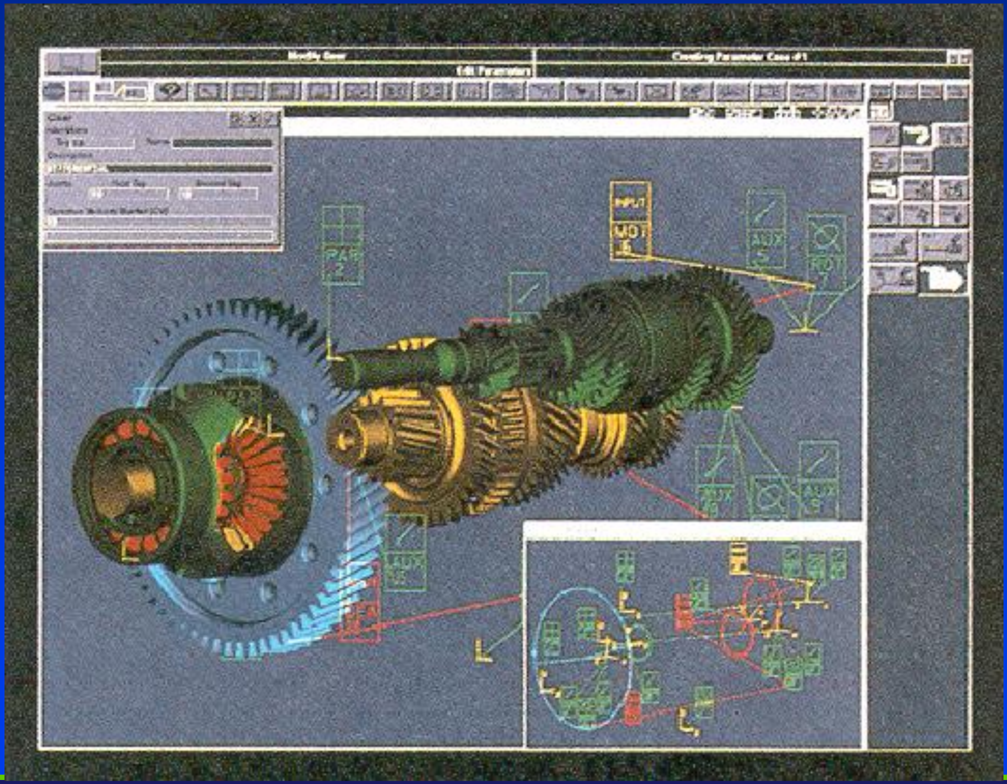
Motivation

- **Some acquisition methods to generate solids**
  - Example: Different medical imaging modalities



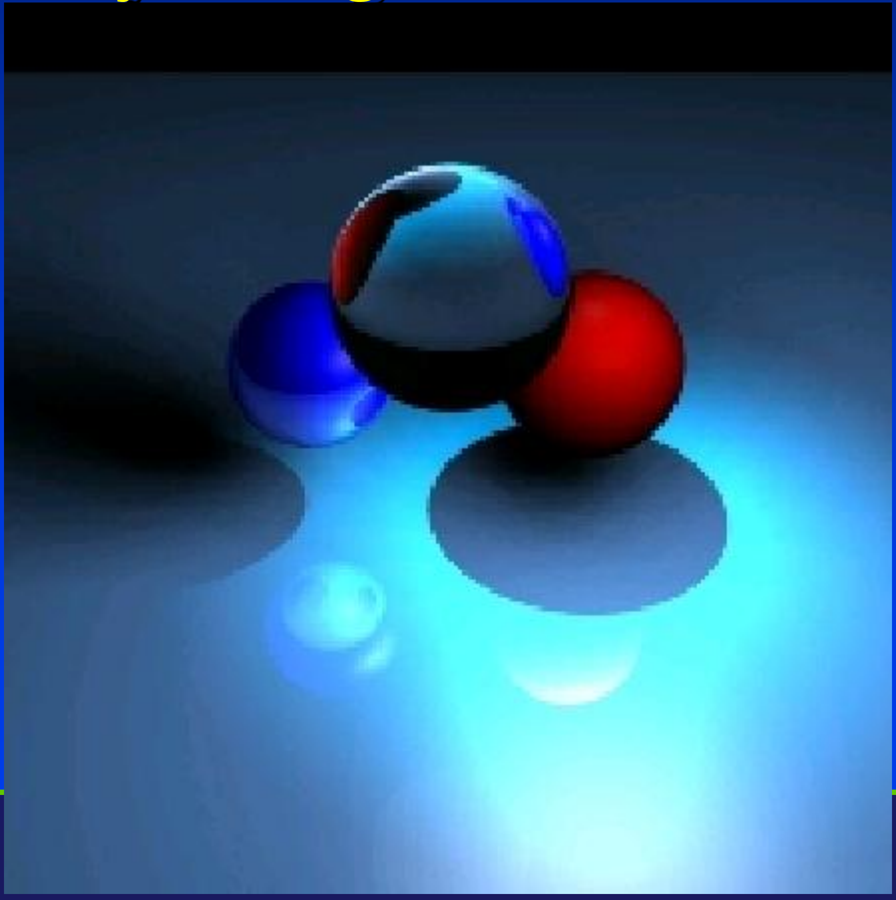
Motivation

- **Some applications to require solids**
  - Example: CAD/CAM/CAE



Motivation

- **Some algorithms to require solids**
  - Example: Ray tracing with refraction

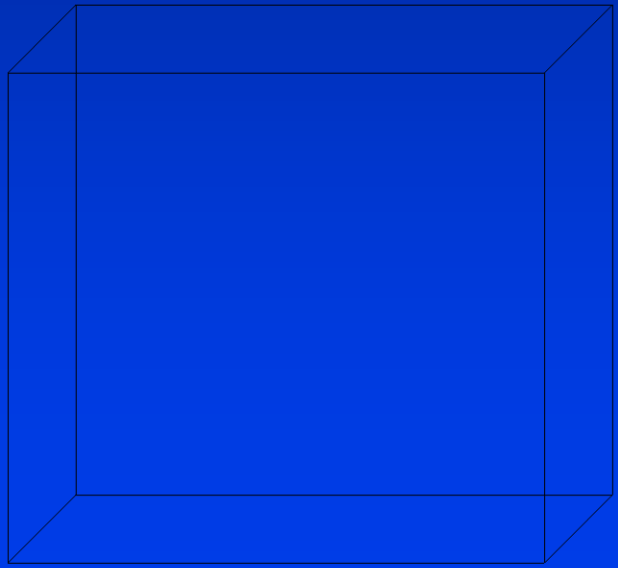


Solid Modeling Representations

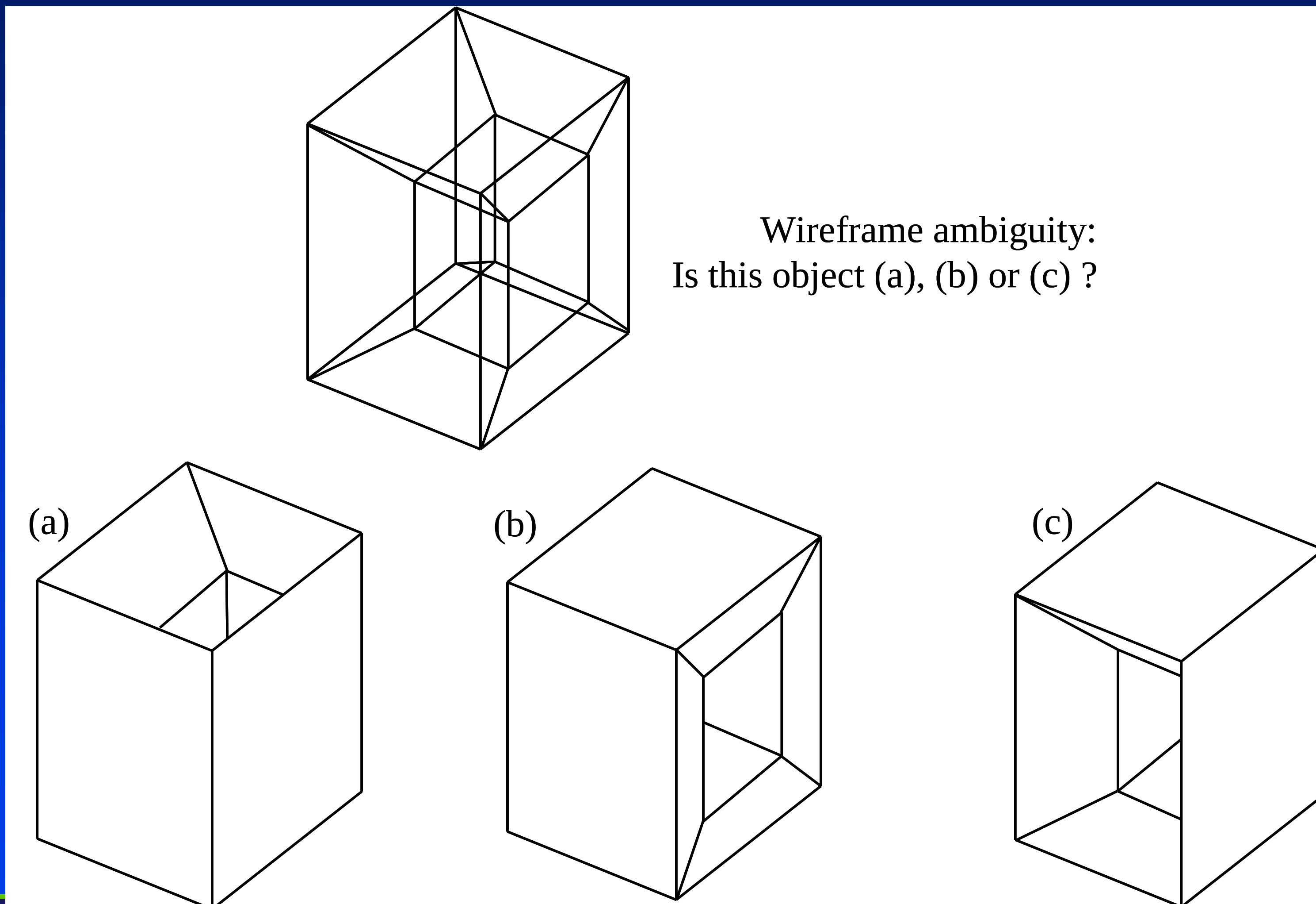
- **Boundary representation (Surface representation)**
- **Constructive Solid Geometry (CAD/CAM/CAE)**
- **Voxels (Medical imaging modalities)**
- **Quadtrees & Octrees (Computational geometry)**
- **Binary Space Partitions (Computational geometry)**

3D and Solid Representation

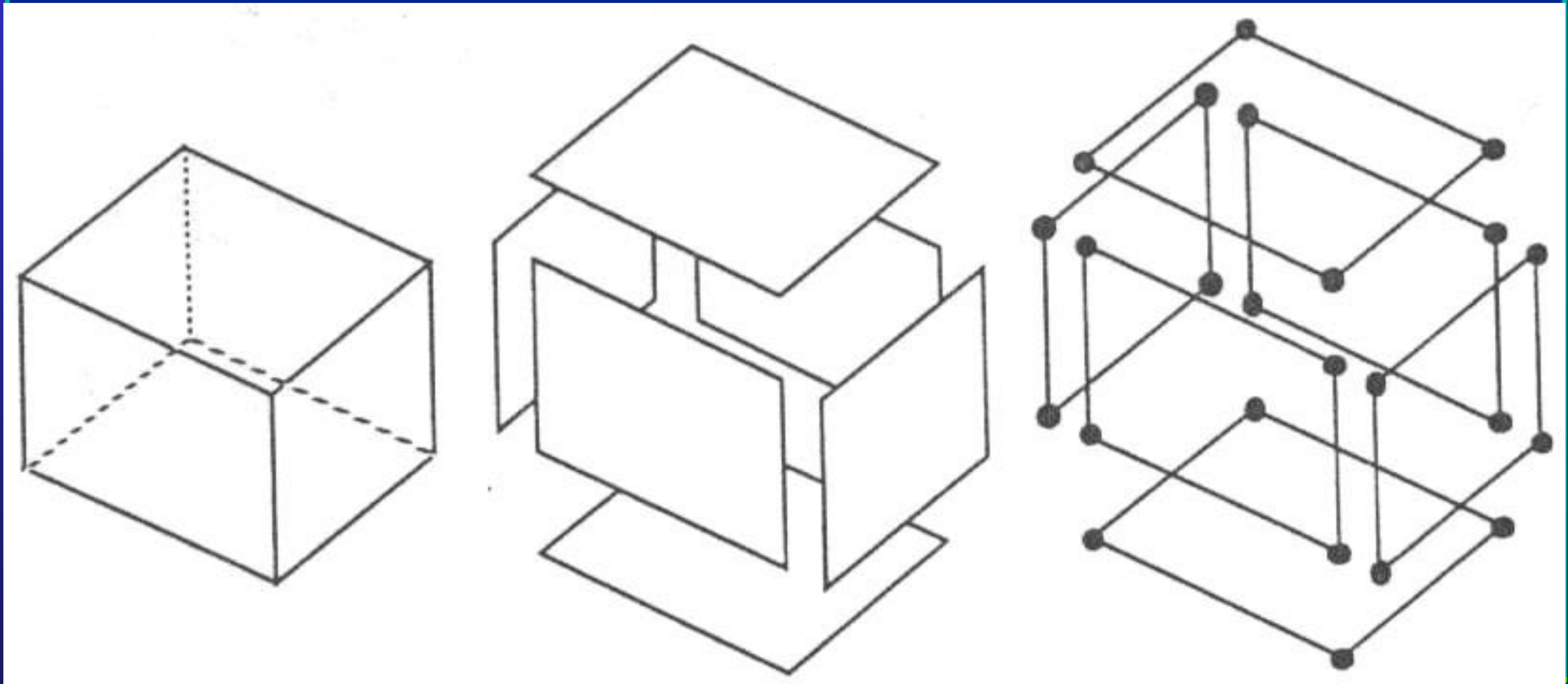
- Wireframe models
- Stores each edge of the object
- Data structure: the vertices (start point, end point)
- The equation of the edge-curve



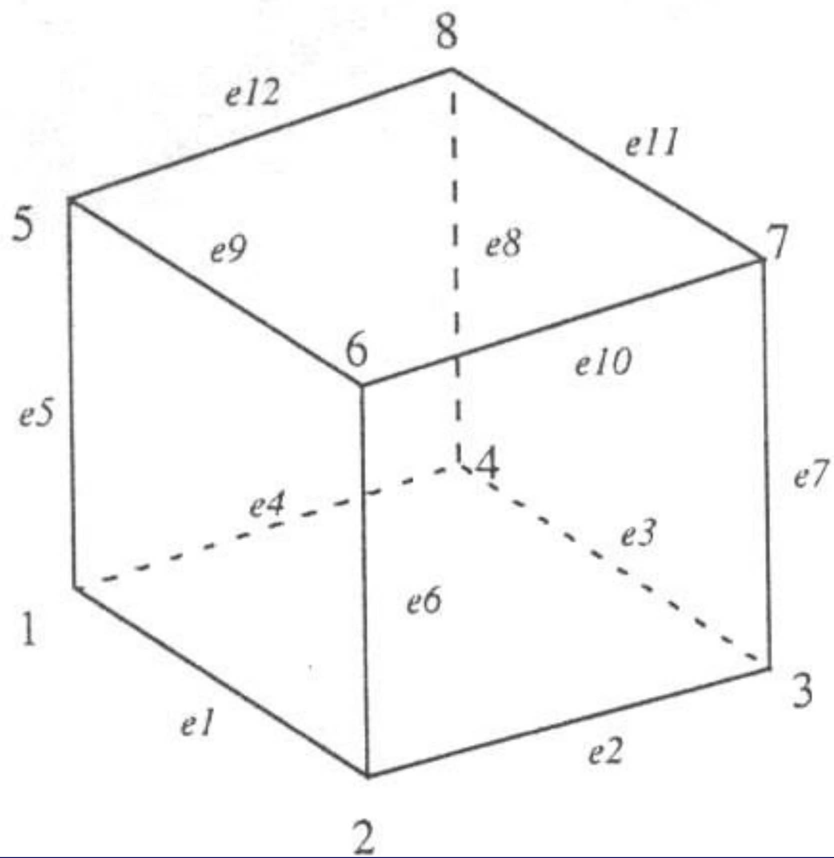
Wireframe Problem: Ambiguity



Boundary Models



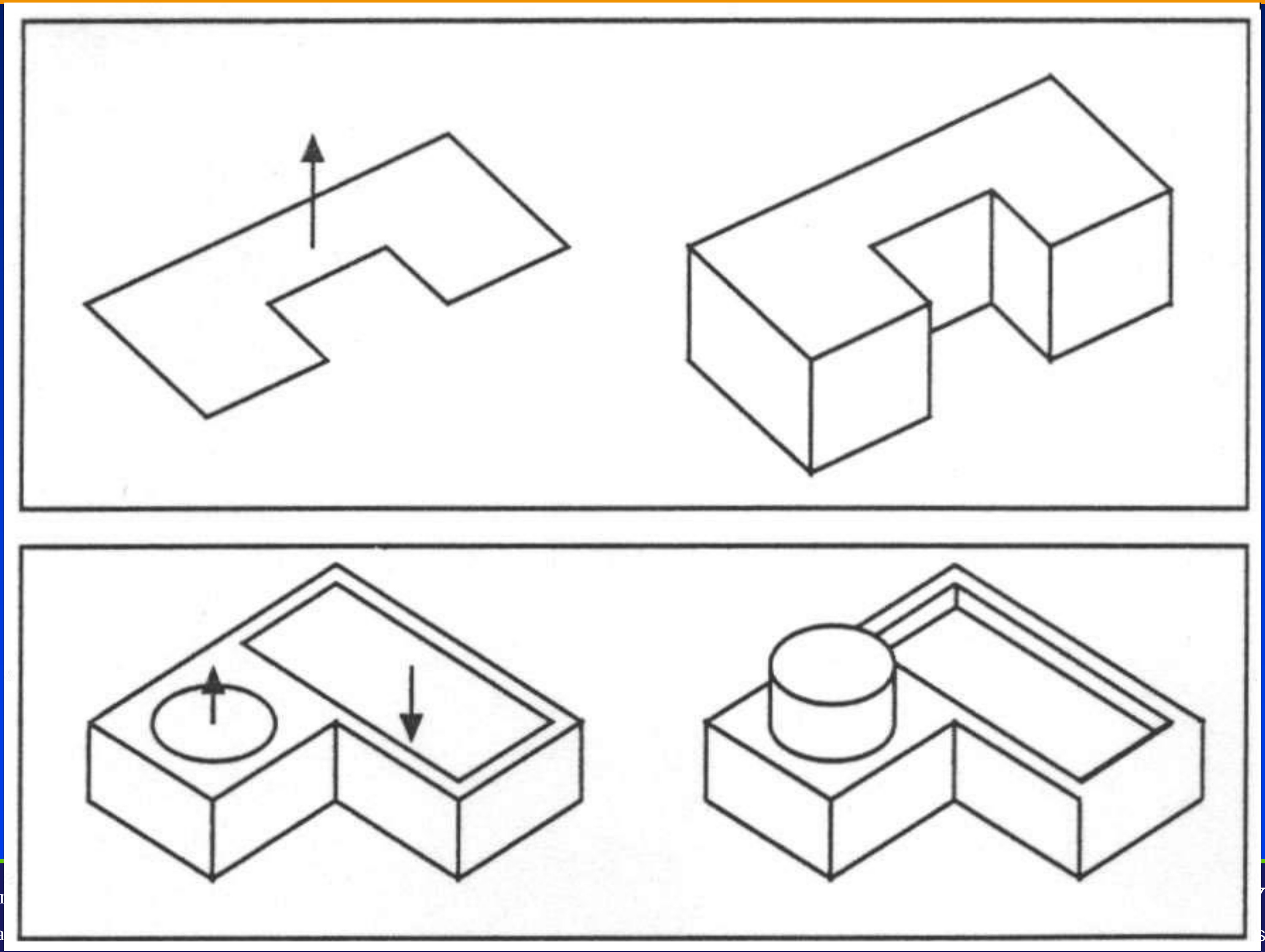
Vertex-Based B-REP



v1	x1	y1	z1
v2	x2	y2	z2
v3	x3	y3	z3
v4	x4	y4	z4
v5	x5	y5	z5
v6	x6	y6	z6
v7	x7	y7	z7
v8	x8	y8	z8

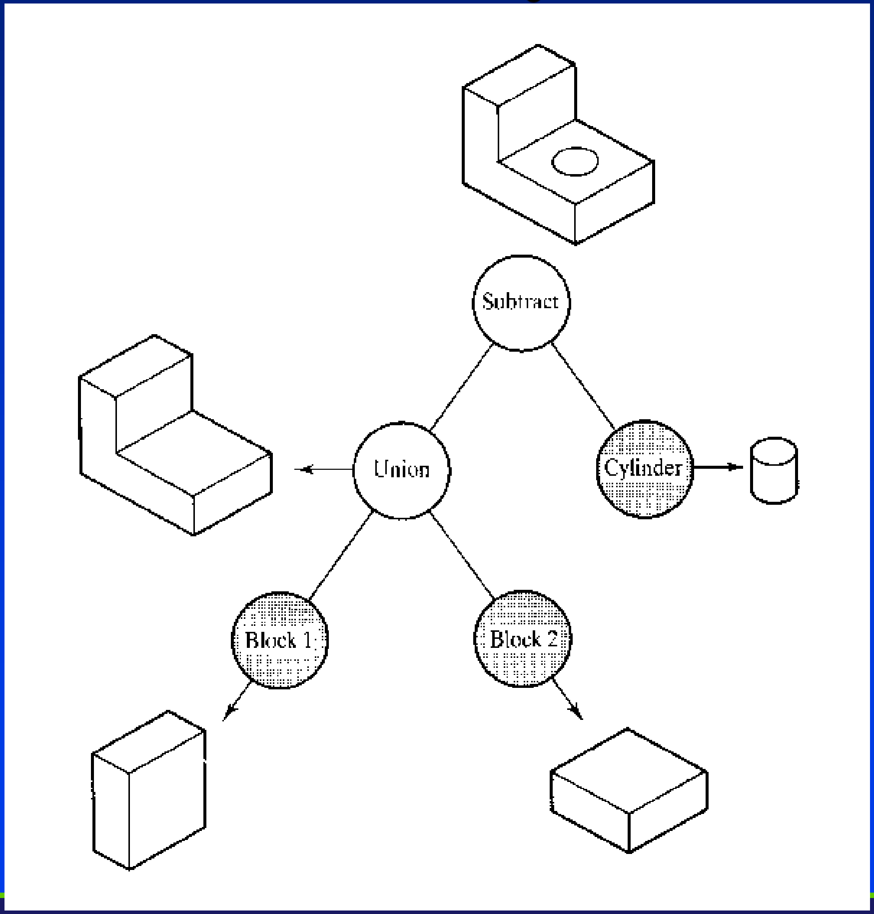
f1	v1	v2	v3	v4
f2	v6	v2	v1	v5
f3	v7	v3	v2	v6
f4	v8	v4	v3	v7
f5	v5	v1	v4	v8
f6	v8	v7	v6	v5

Procedural Models (Sweeping)



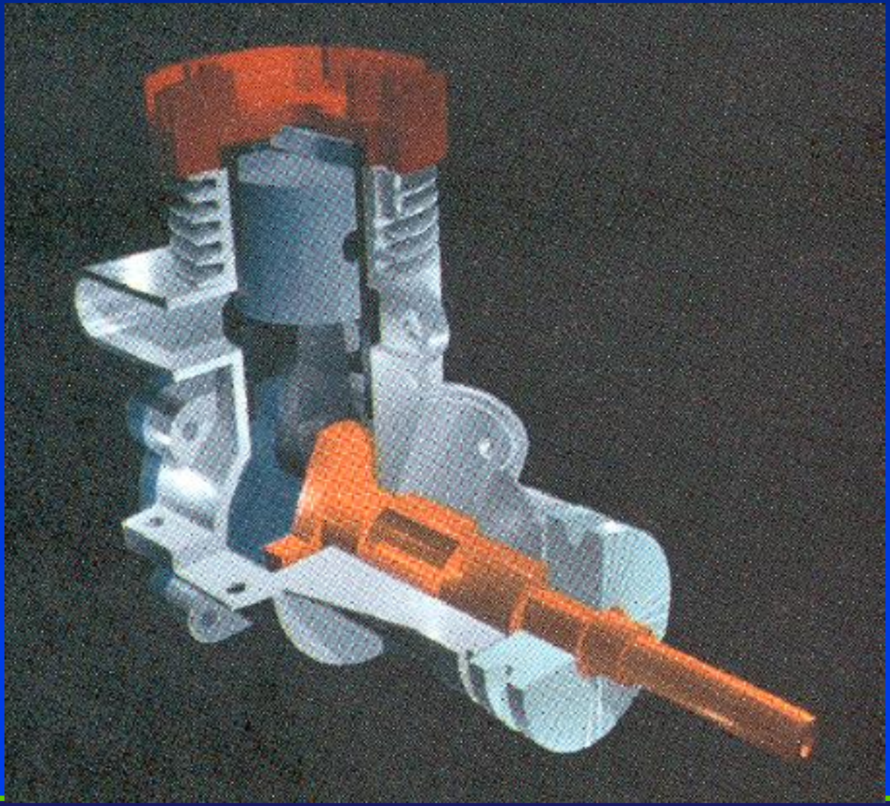
Constructive Solid Geometry (CSG)

- **Represent solid object as hierarchy of Boolean operations**
  - Union
  - Intersection
  - Difference



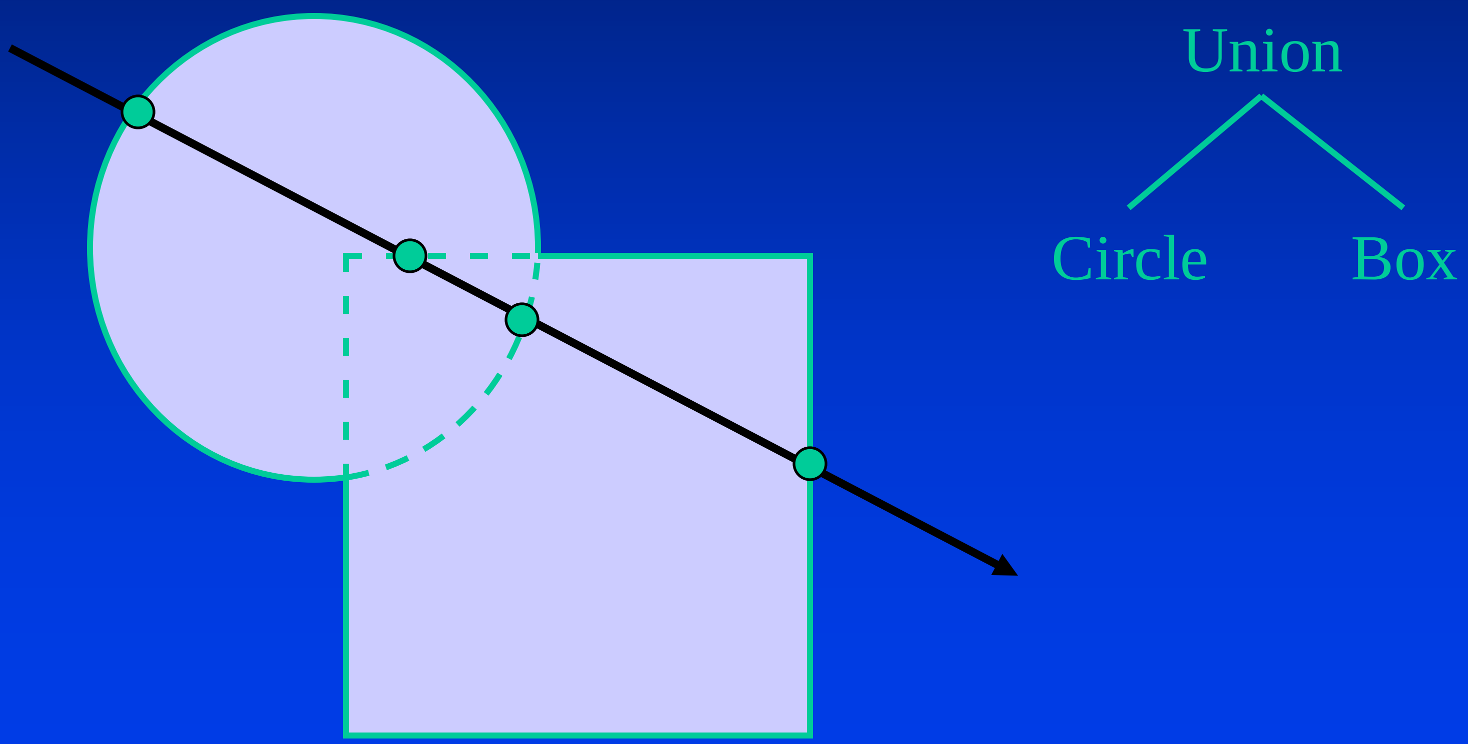
CSG Acquisition

- **Interactive modeling programs**
  - CAD/CAM/CAE

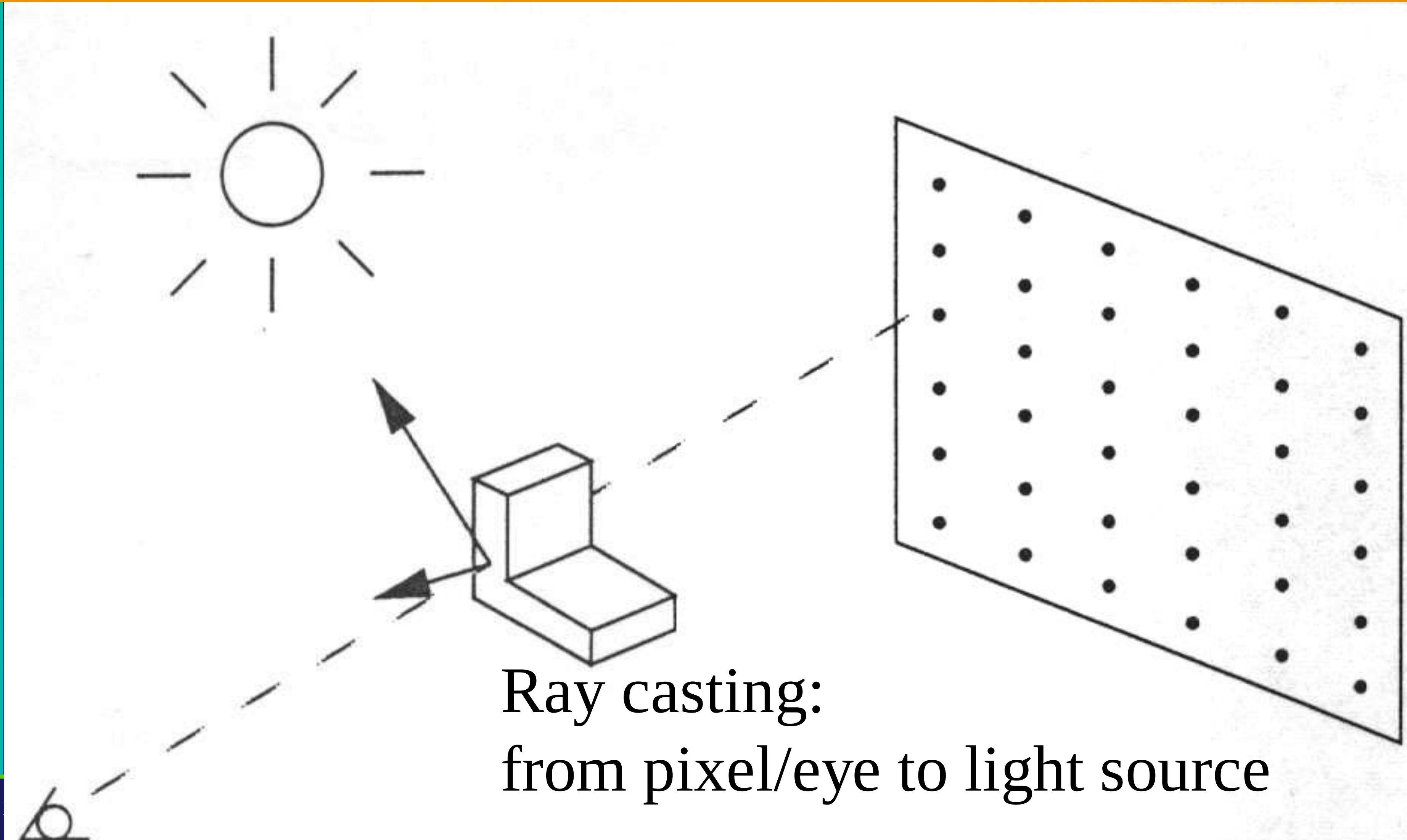


CSG Display & Analysis

- **Ray-casting**

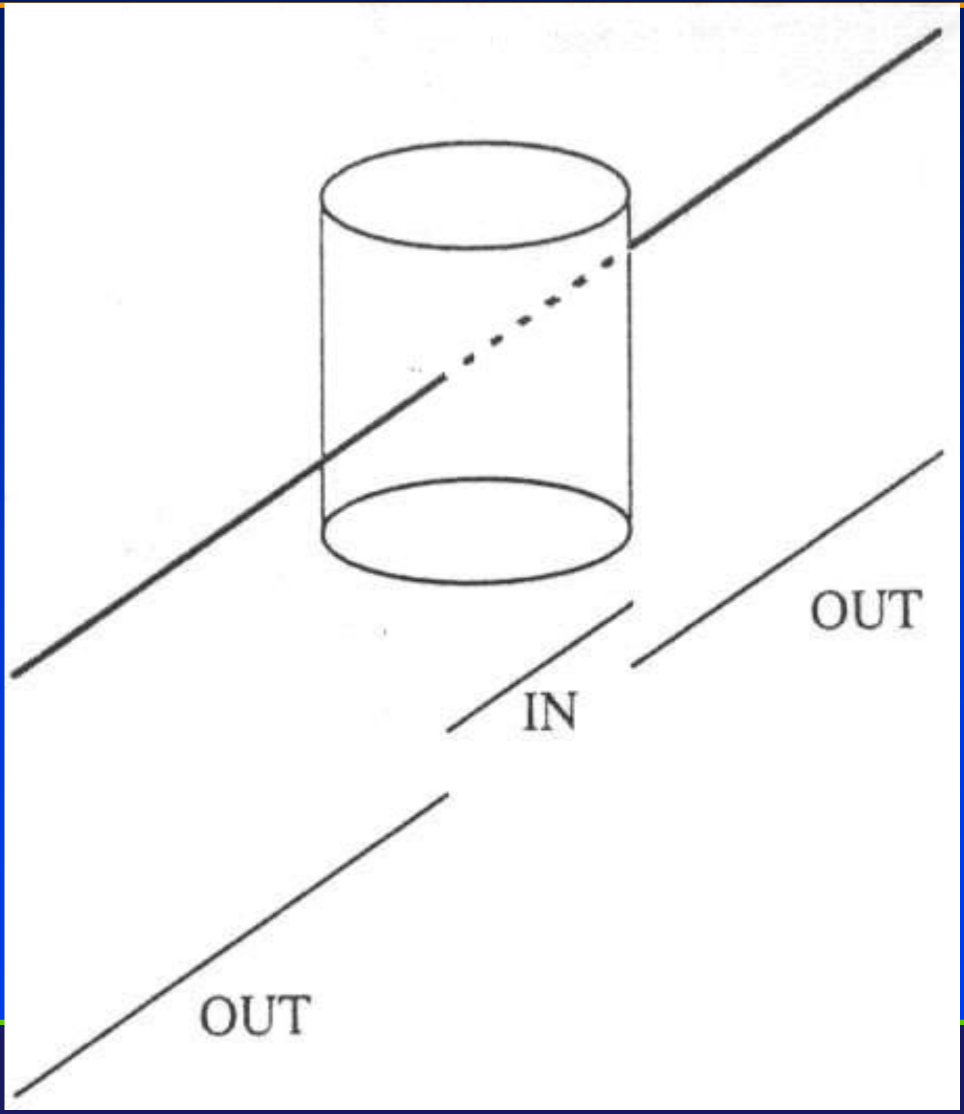


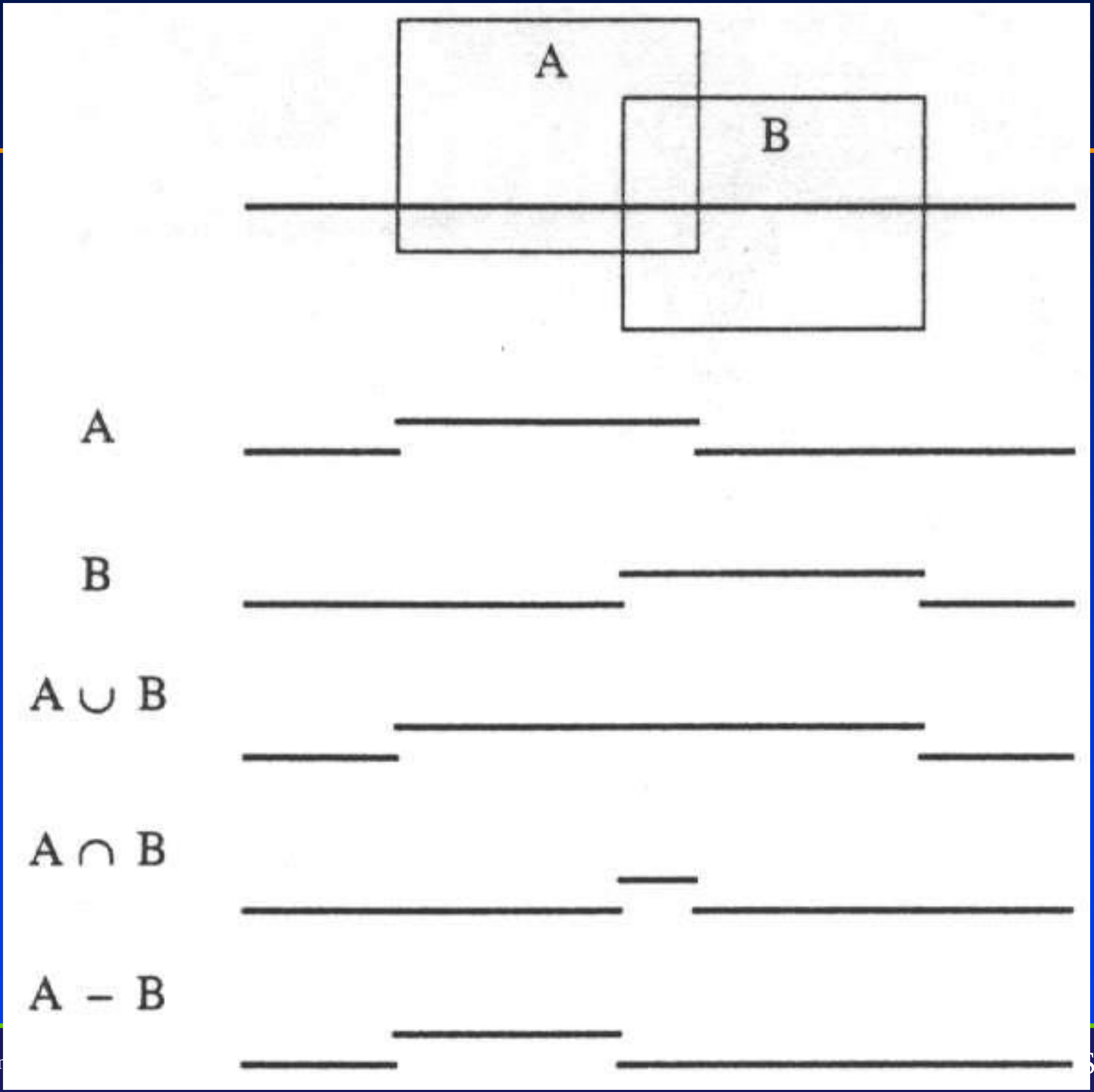
Ray Casting



Ray casting:
from pixel/eye to light source

Ray Classification

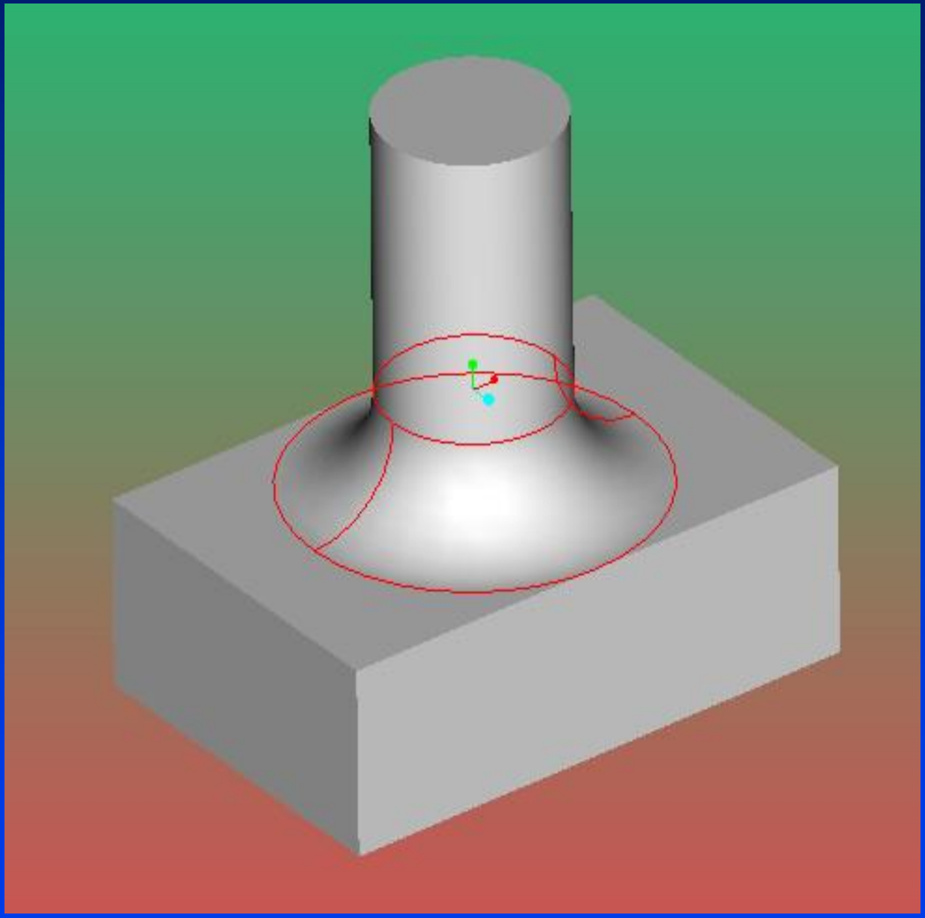



A
B
A
B
A ∪ B
A ∩ B
A − B

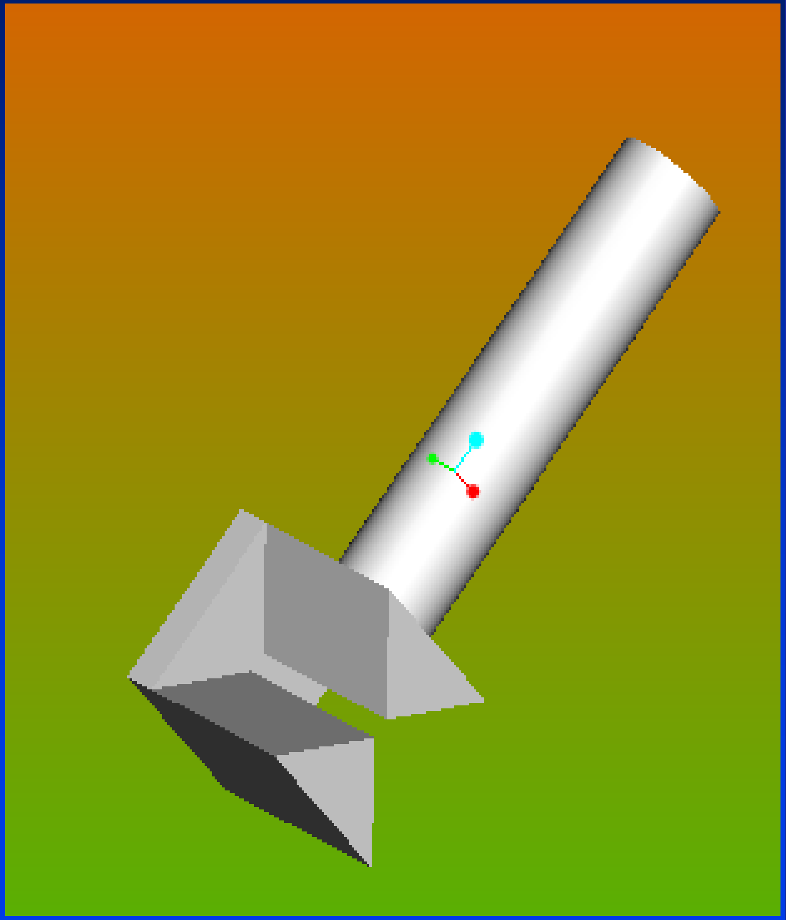
Application: Computing Volume

- Put bounding box around object
- Pick n random points inside the box
  - Determine if each point is inside/outside the CSG Tree
- Volume $\approx \#inside / n$

Examples of Solid Models

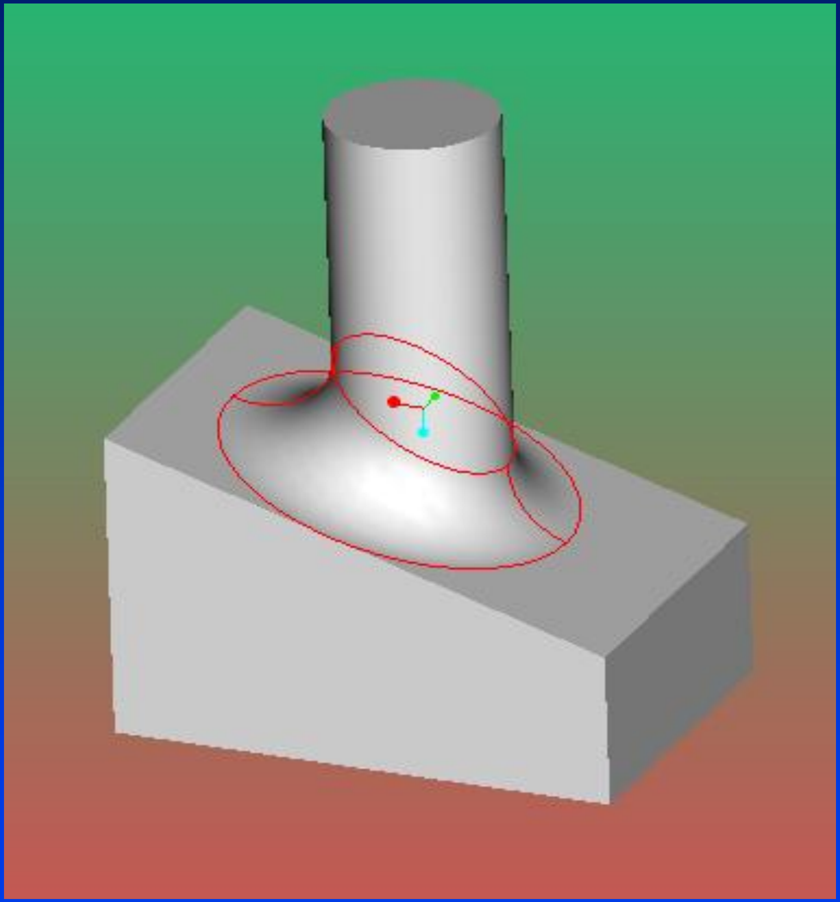


Torus

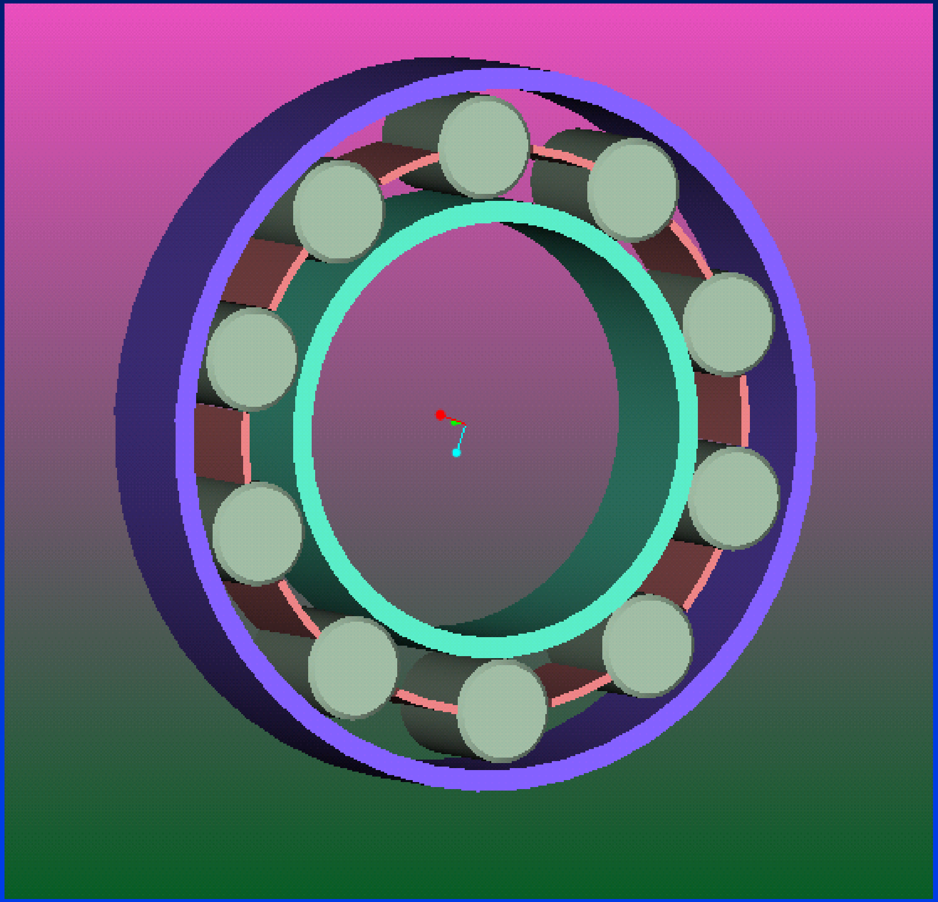


Lock

More Examples

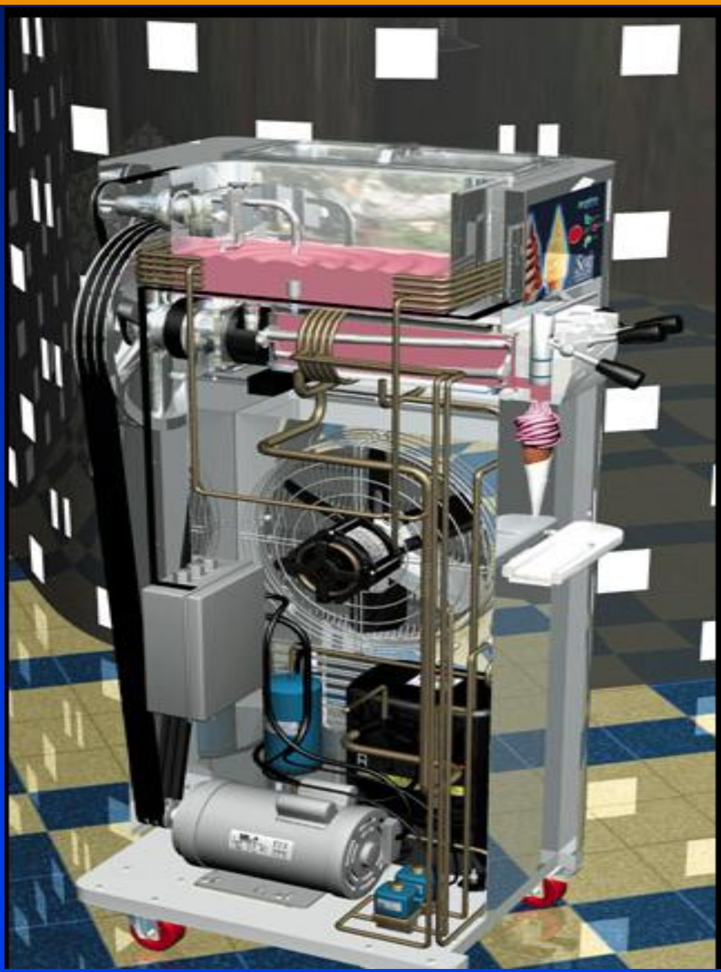


Slanted Torus



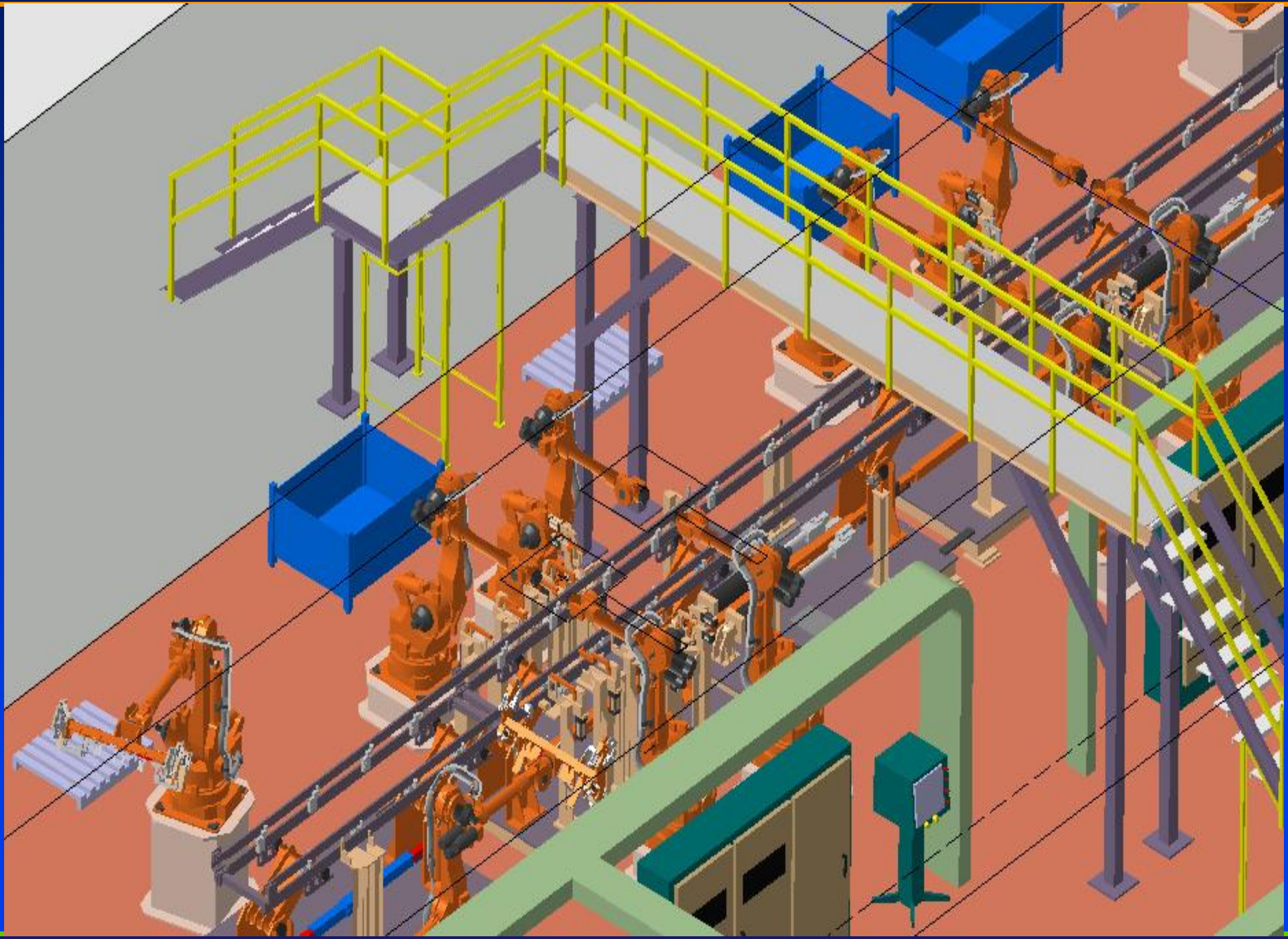
Bearing

Examples

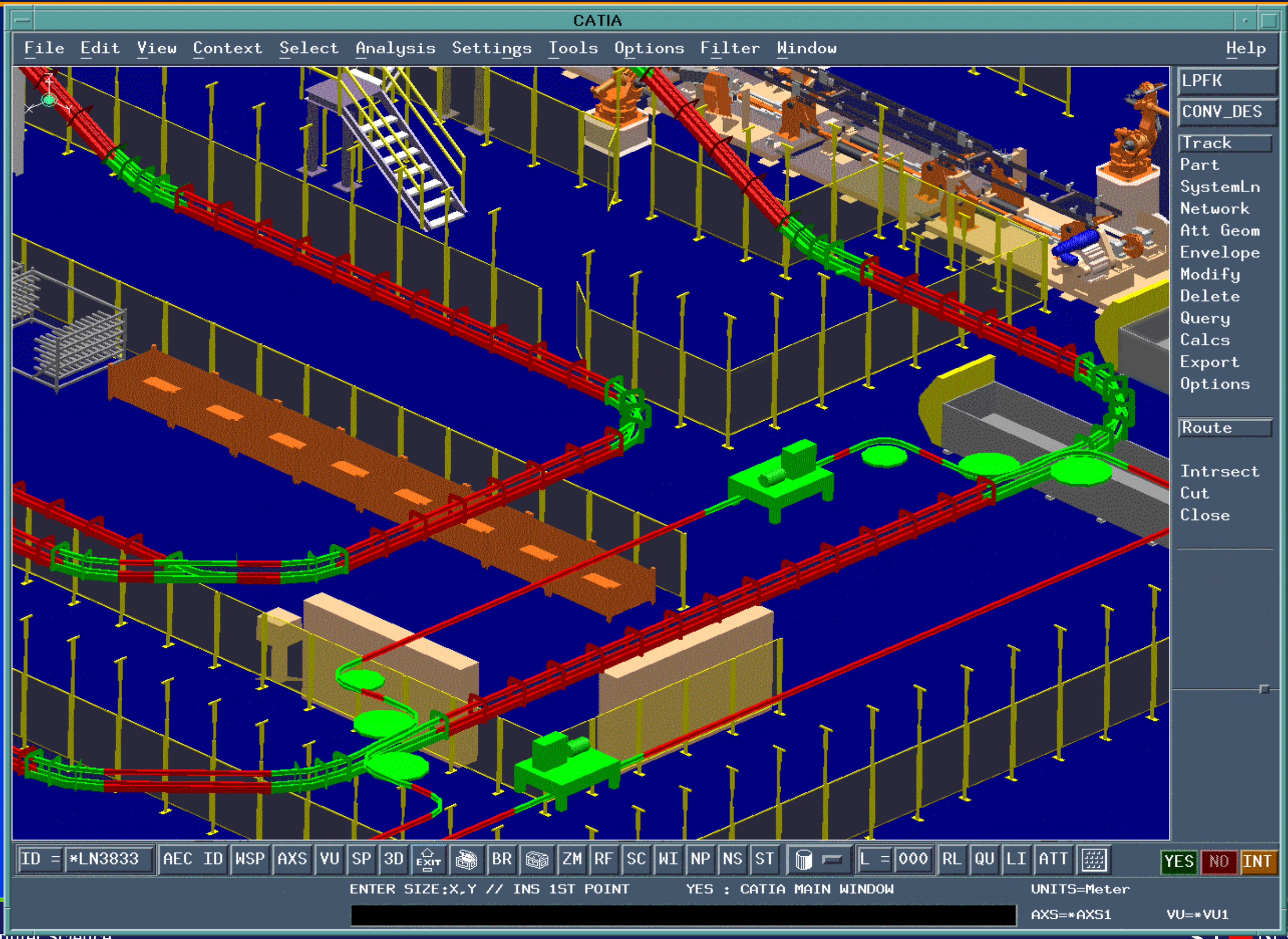


Solid Model of an Ice-Cream Machine

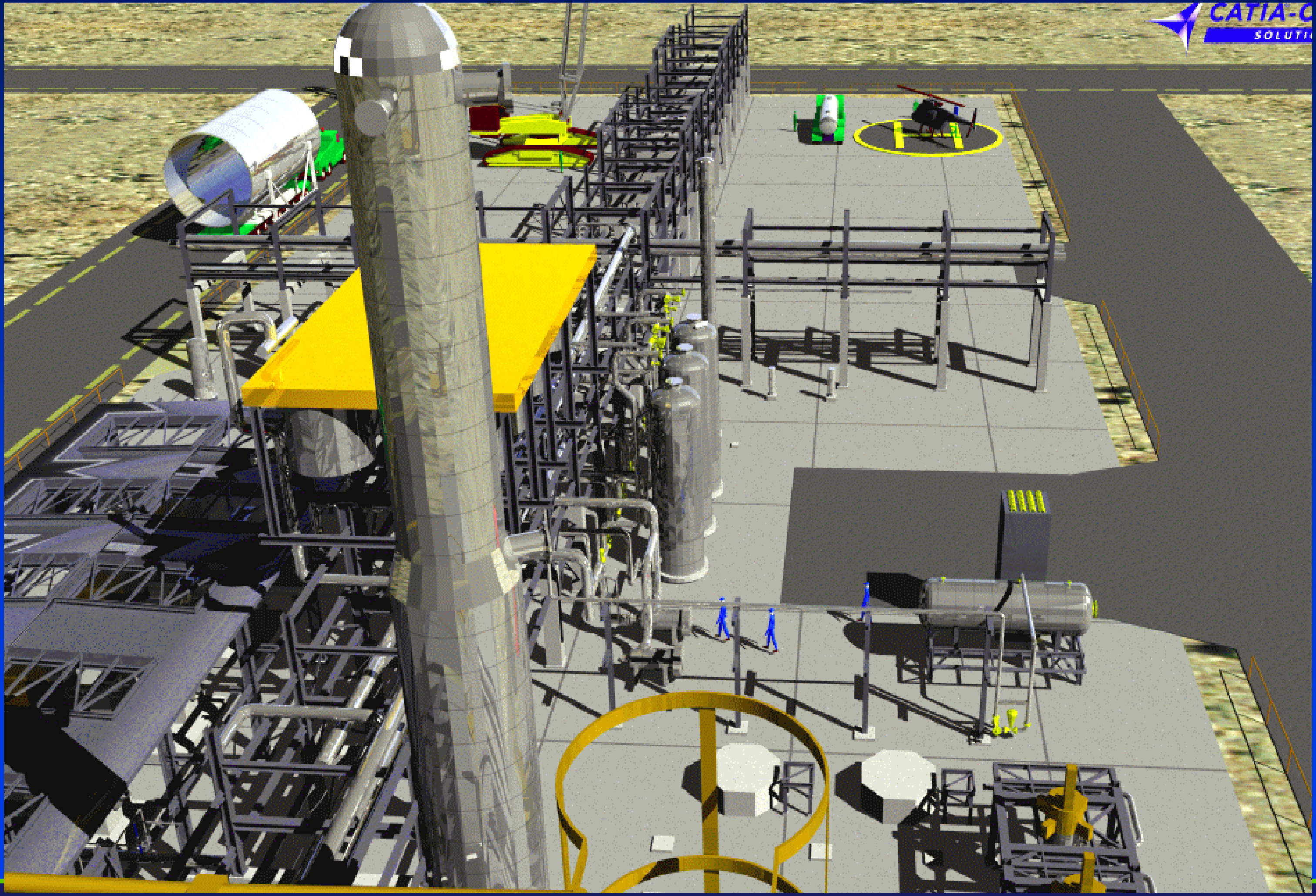
Chemical Plants



Chemical Plants



Chemical Plants (Example)



B-REP (Boundary REPresentation)

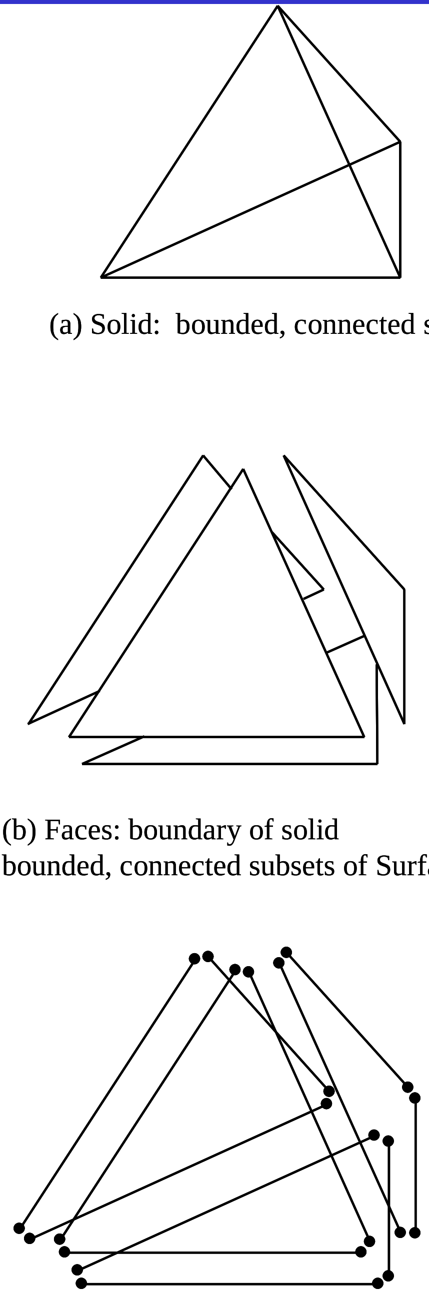
- What entities define the
- Boundary of a solid ?
- Boundary of surfaces?
- Boundary of curves (edges) ?
- Boundary of points ?

B-REP

Boundary of a solid…

Boundary of surfaces…

Boundary of curves (edges)…



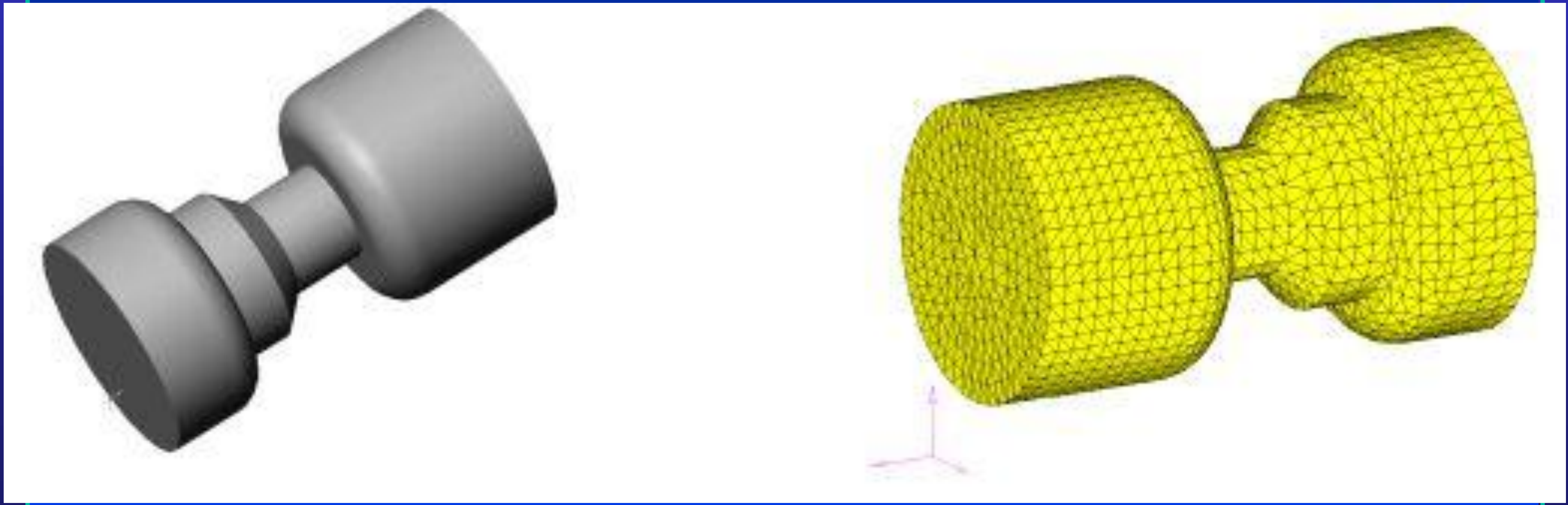
(a) Solid: bounded, connected subset of E^3

(b) Faces: boundary of solid
bounded, connected subsets of Surfaces

(c) Edges: boundary of faces
bounded, connected subsets of curves

Department of Computer Science
Center for Visual Computing

B-REP Polyhedral Models

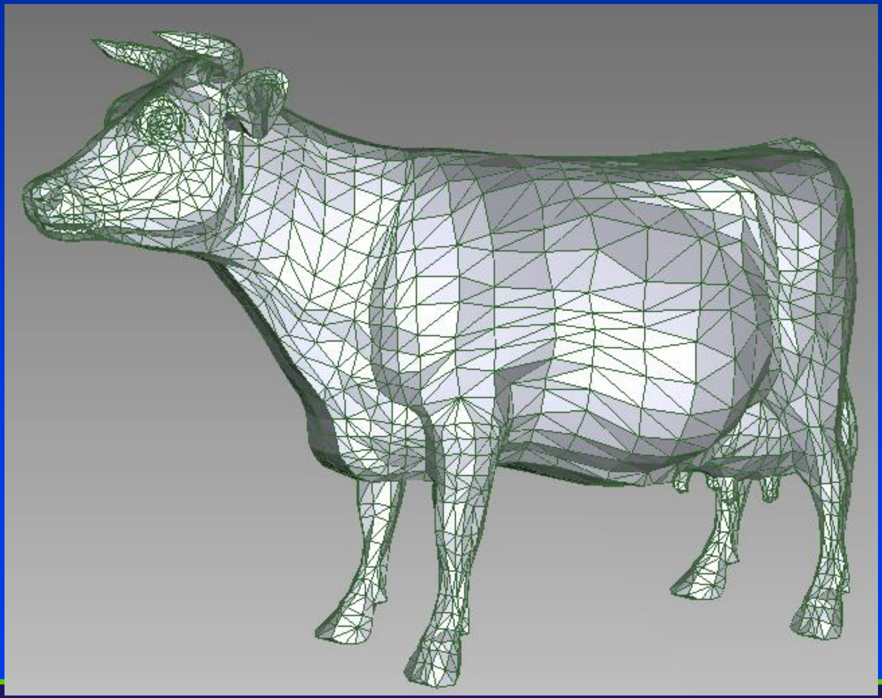


Using a Boundary Model

- Compute volume, weight
- Compute surface area
- Point inside/outside solid
- Intersection of two faces
- …

Boundary Representation

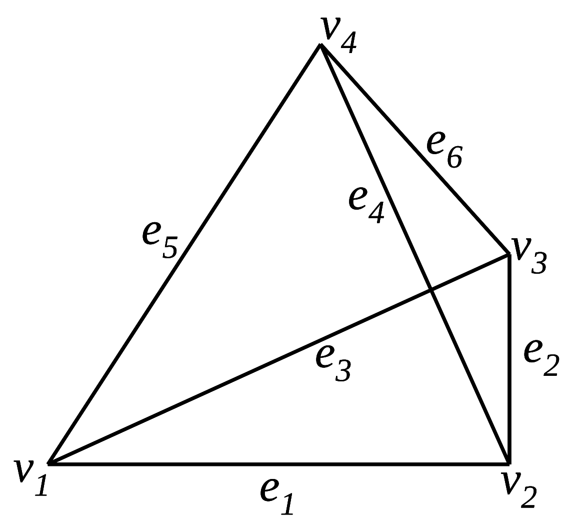
- Stores the boundary of a solid
  - Geometry: vertex locations
  - Topology: connectivity information
    - Vertices
    - Edges
    - Faces



Boundary Representation

- Constant time adjacency information
  - For each vertex,
    - Find edges/faces touching vertex
  - For each edge,
    - Find vertices/faces touching edge
  - For each face,
    - Find vertices/edges touching face

An Edge-Based Model



Faces:

f_1	$e_1\ e_4\ e_5$
f_2	$e_2\ e_6\ e_4$
f_3	$e_3\ e_5\ e_6$
f_4	$e_3\ e_2\ e_1$

Edges:

e_1	$v_1\ v_2$
e_2	$v_2\ v_3$
e_3	$v_3\ v_1$
e_4	$v_2\ v_4$
e_5	$v_1\ v_4$
e_6	$v_3\ v_4$

Vertices:

v_1	$x_1\ y_1\ z_1$
v_2	$x_2\ y_2\ z_2$
v_3	$x_3\ y_3\ z_3$
v_4	$x_4\ y_4\ z_4$
v_5	$x_5\ y_5\ z_5$
v_6	$x_6\ y_6\ z_6$

Boundary Representation

- Advantages
  - Explicitly stores neighbor information
  - Easy to render
  - Easy to calculate volume
  - Nice-looking surface
- Disadvantages
  - CSG very difficult
  - Inside/Outside test hard

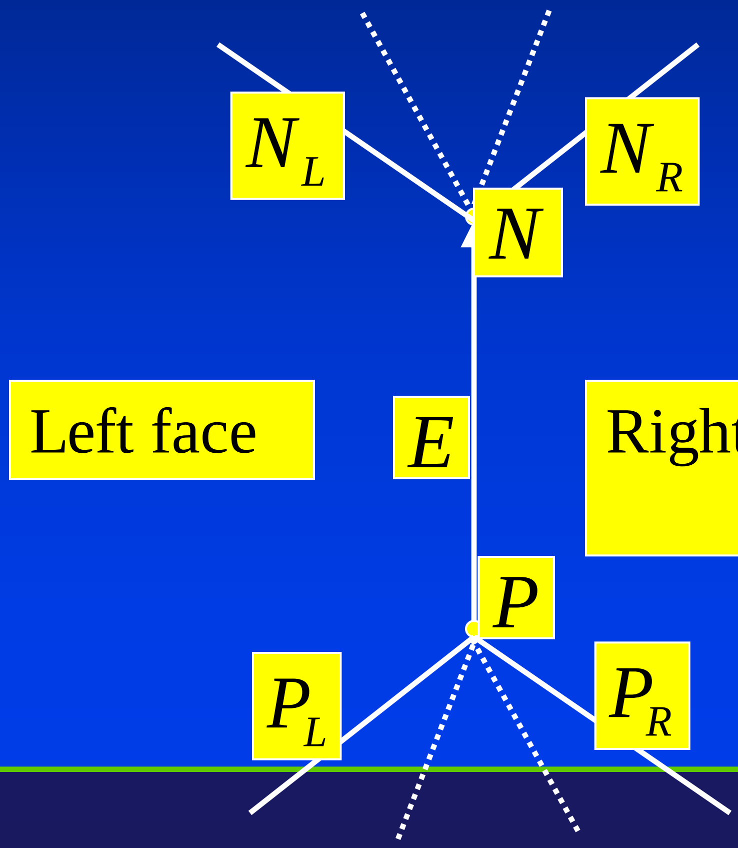
Winged Edge Data Structure

- Efficient implementation of frequently-used algorithms

- Area of face
- Hidden surface removal
- Find neighbor-faces of a face

Winged Edge Data Structure

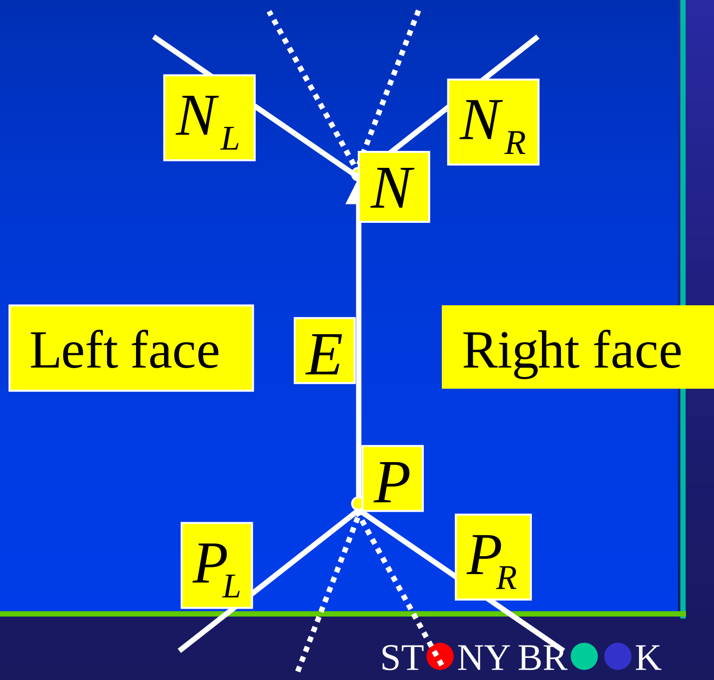
- Each vertex/face points to a single edge containing that vertex/face



STONY BROOK
STATE UNIVERSITY OF NEW YORK

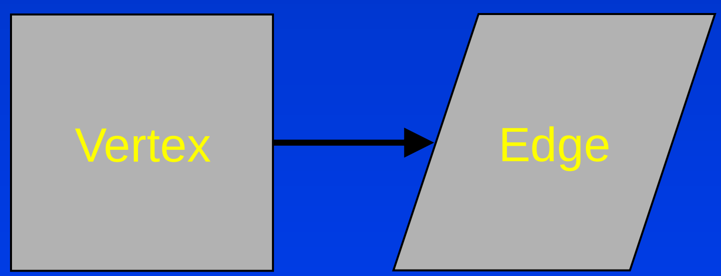
Winged Edge Data Structure

- Given a face, find all vertices touching that face
- Given a vertex, find all edge-adjacent vertices
- Given a face, find all adjacent faces



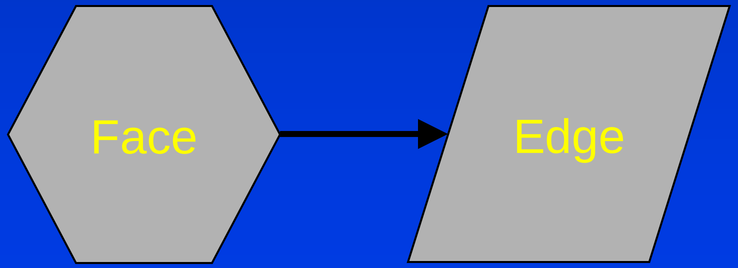
Winged Edge Data Structure

- Vertex record
  - Contains the vertex coordinates
  - Contains a unique number for the vertex
  - Contains a pointer to the record for an edge that ends at that vertex.



Winged Edge Data Structure

- Face record contains a pointer to the edge record of one of its edges

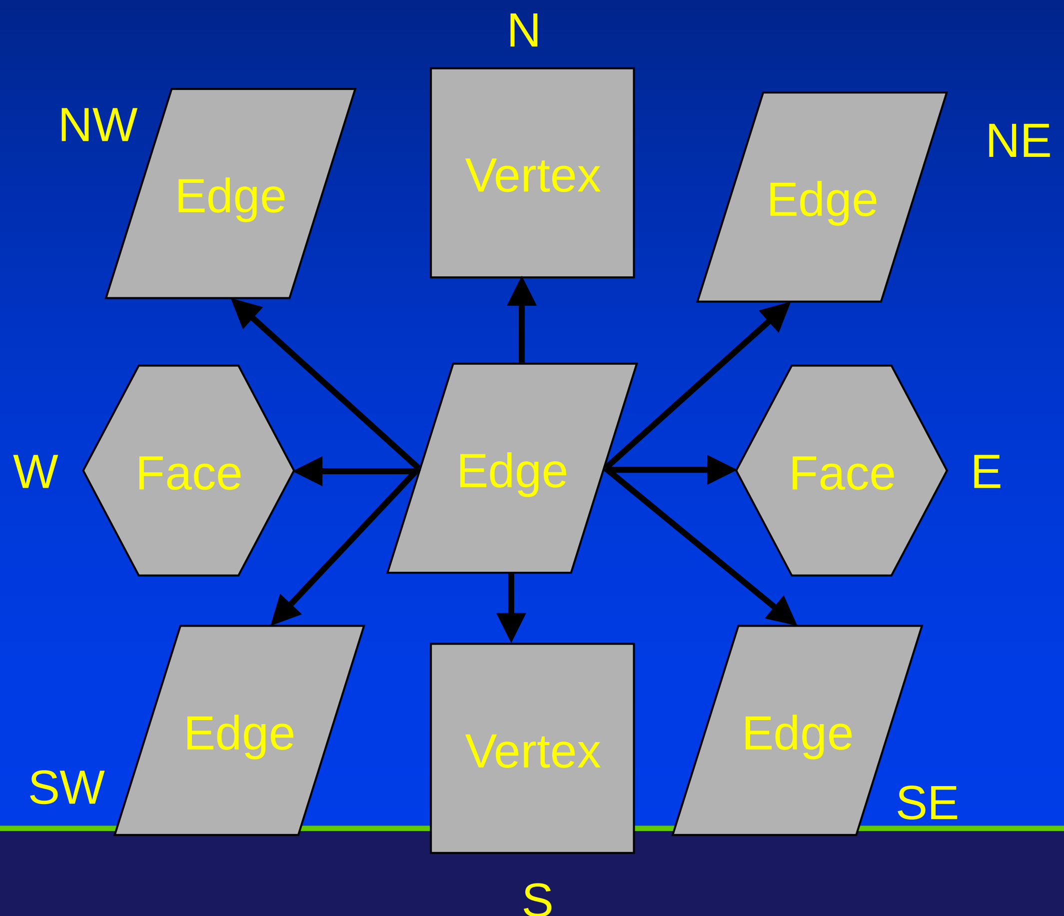


Winged Edge Data Structure

- Edge record
  - Provides most of the connectivity for the mesh.
  - Contains a pointer to each of the vertices at its ends.
  - Contains a pointer to each face on either side of the edge.
  - Contains pointers to the four wing edges that are neighbors in the polygonal mesh.
  - These pointers connect the faces and vertices into a polygonal mesh and allow the mesh to be traversed efficiently, i.e., efficient traversal from edge to edge around a face.

Winged Edge Data Structure

- Edge record - Notation of compass directions is just for convenience; in a polygonal mesh, there is no global sense of direction.



Traversing a Face

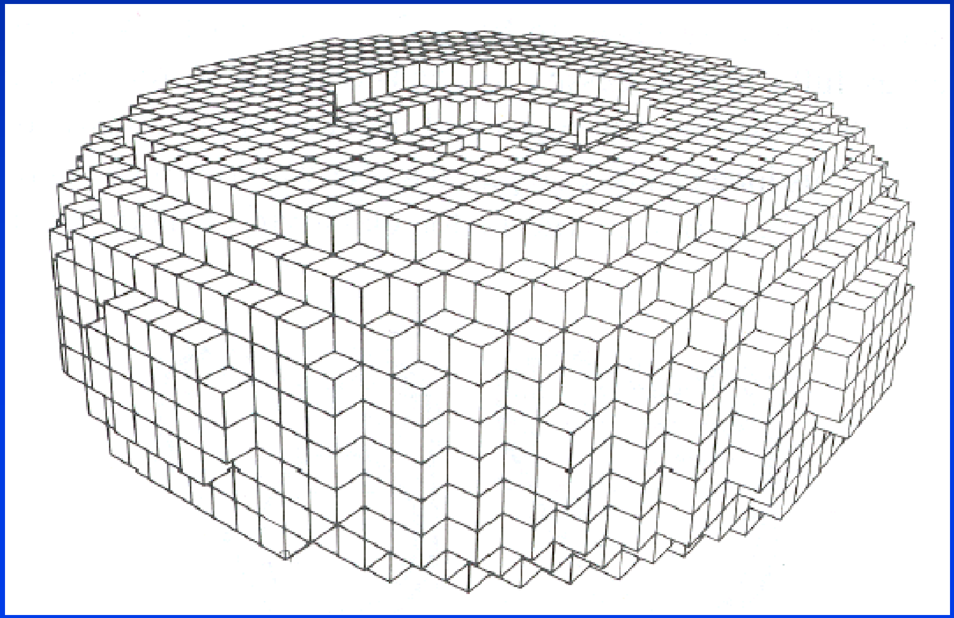
- Start at the edge pointed to by the face record
- For clockwise traversal, follow the northeast wing if the face is east of the edge. Follow the southwest wing if the face is west of the edge.
- For each edge, a check must be performed to determine if the face is east or west of the edge
- Continue until the starting edge is reached

B-REP vs. CSG ?

- Using: CSG is more intuitive
- Computing: BREP is more convenient
- Modern CAD Systems:
- CSG for GUI (feature tree)
- B-REP for internal storage and API's

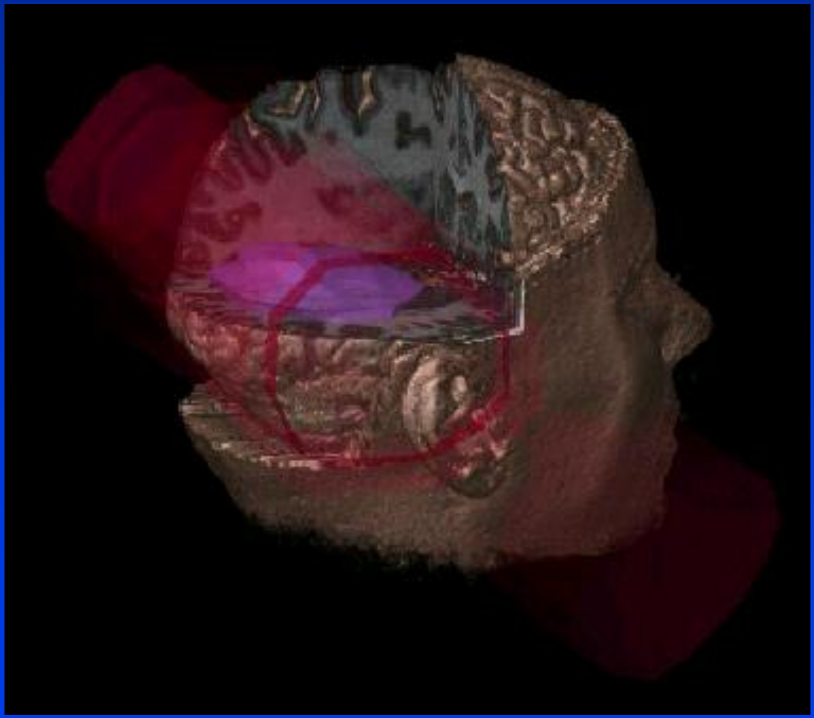
Voxel Representation

- **Partition space into uniform grid**
  - Grid cells are called *voxels* (like pixels)
- **Store properties of solid object with each voxel**
  - Occupancy
  - Color
  - Density
  - Temperature
  - Etc.



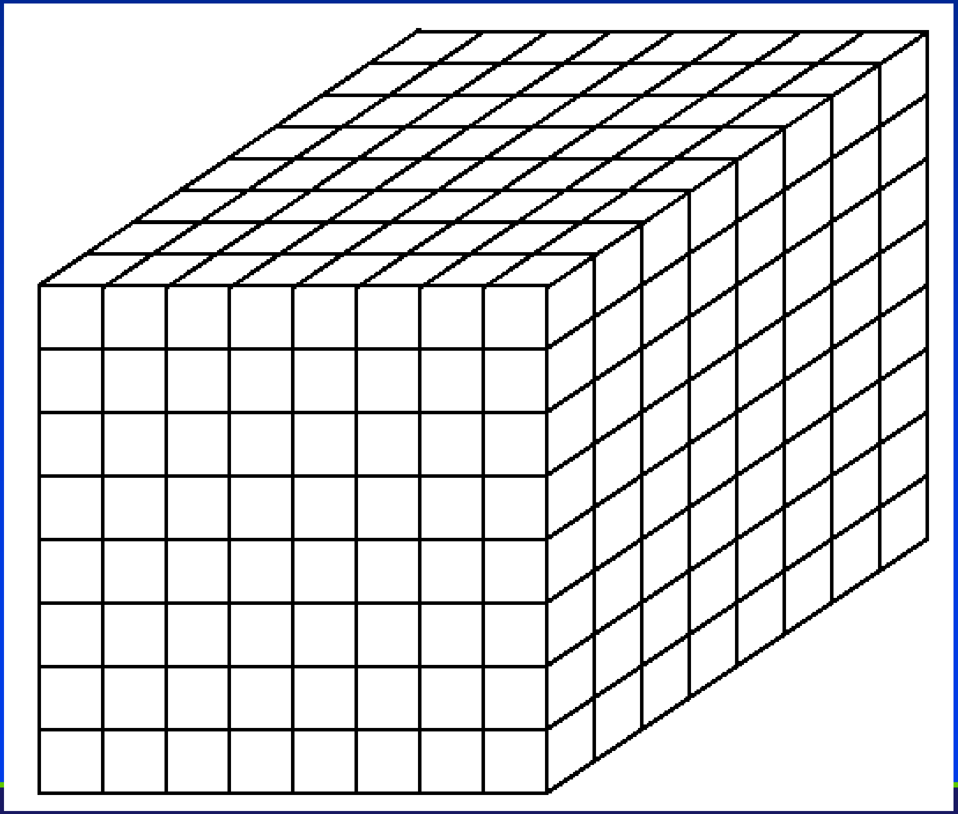
Voxel Acquisition

- **Scanning devices using different medical imaging modalities**
  - MRI
  - CAT
- **Simulation**
  - FEM



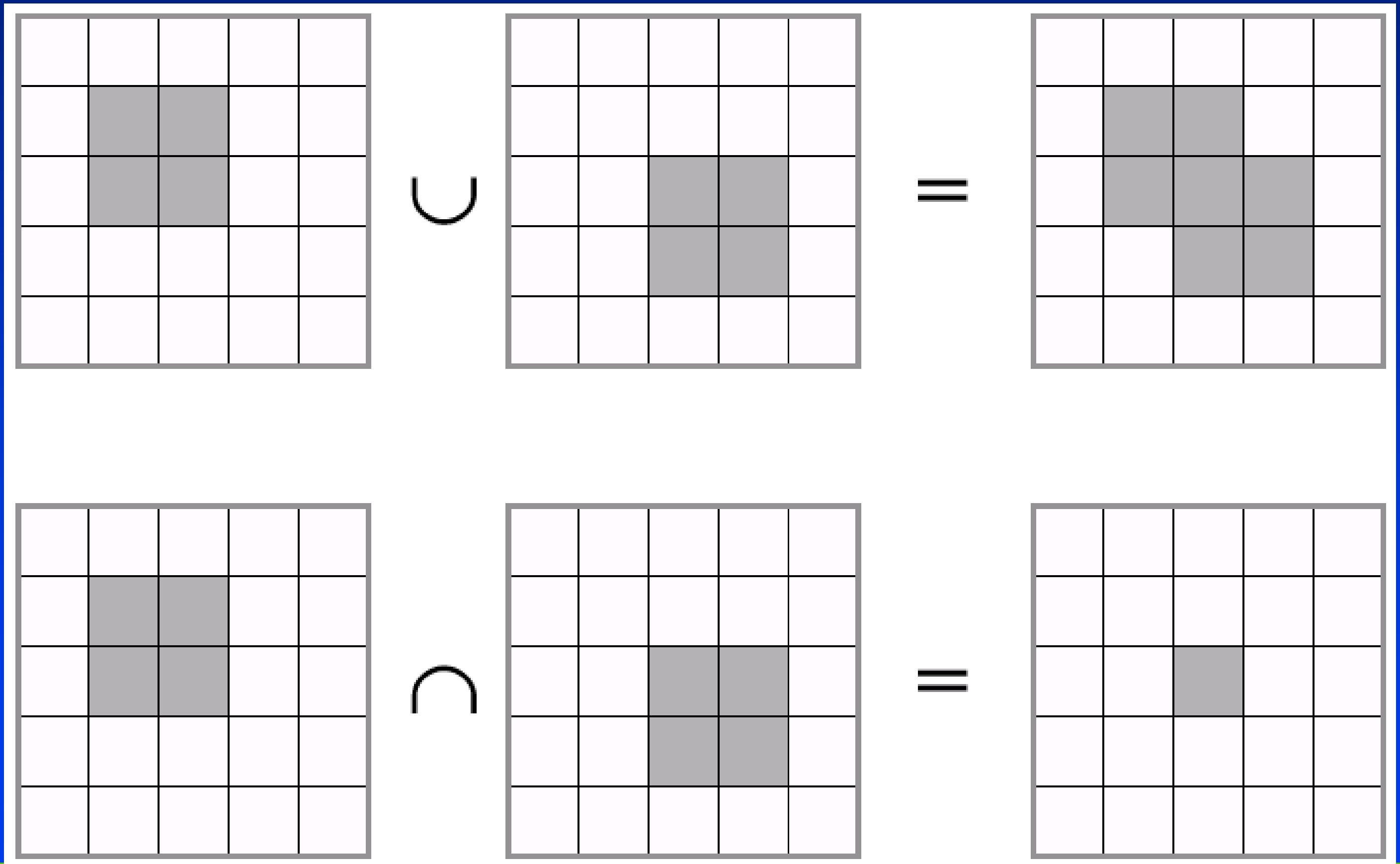
Voxel Storage

- **O(n^3) storage for *n* x *n* x *n* grid**
  - 1 billion voxels for 1000 x 1000 x 1000



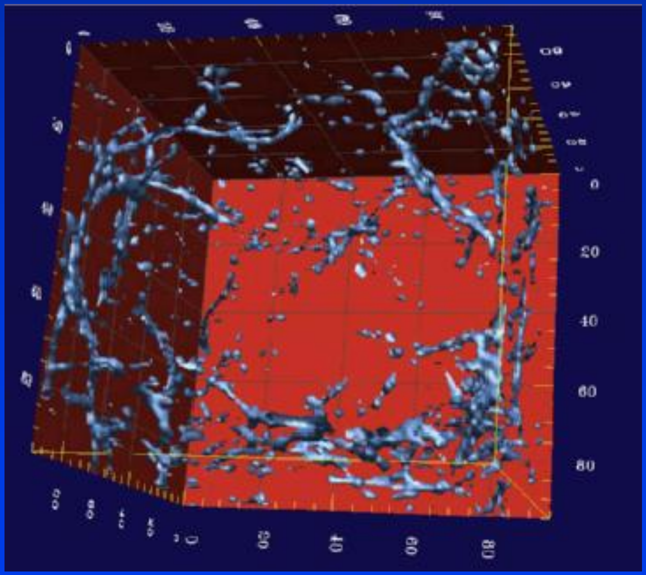
Voxel Boolean Operations

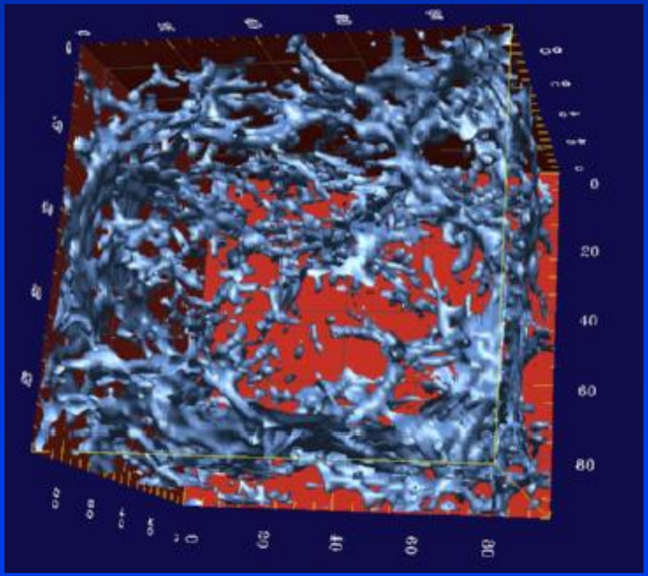
- **Compare objects voxel by voxel**



Voxel Display

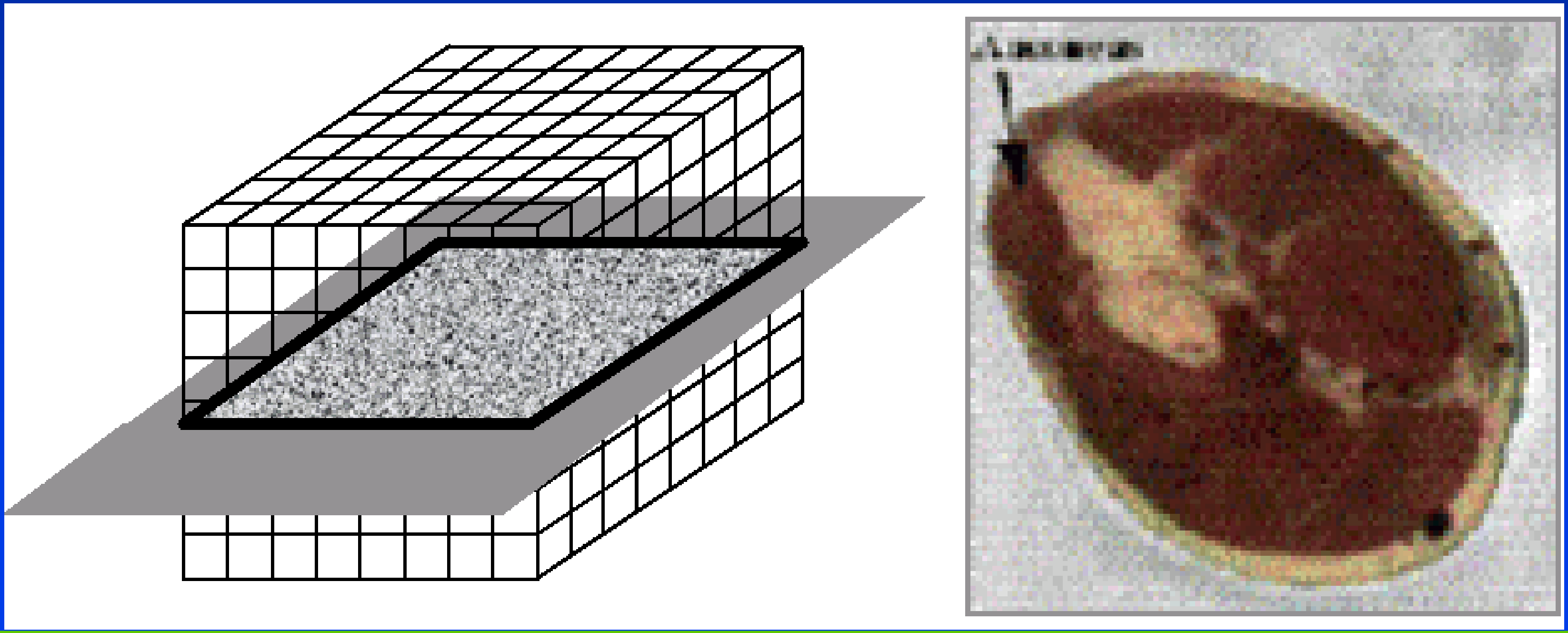
- **Isosurface rendering**
  - Render surfaces bounding volumetric regions of constant value (e.g., density)



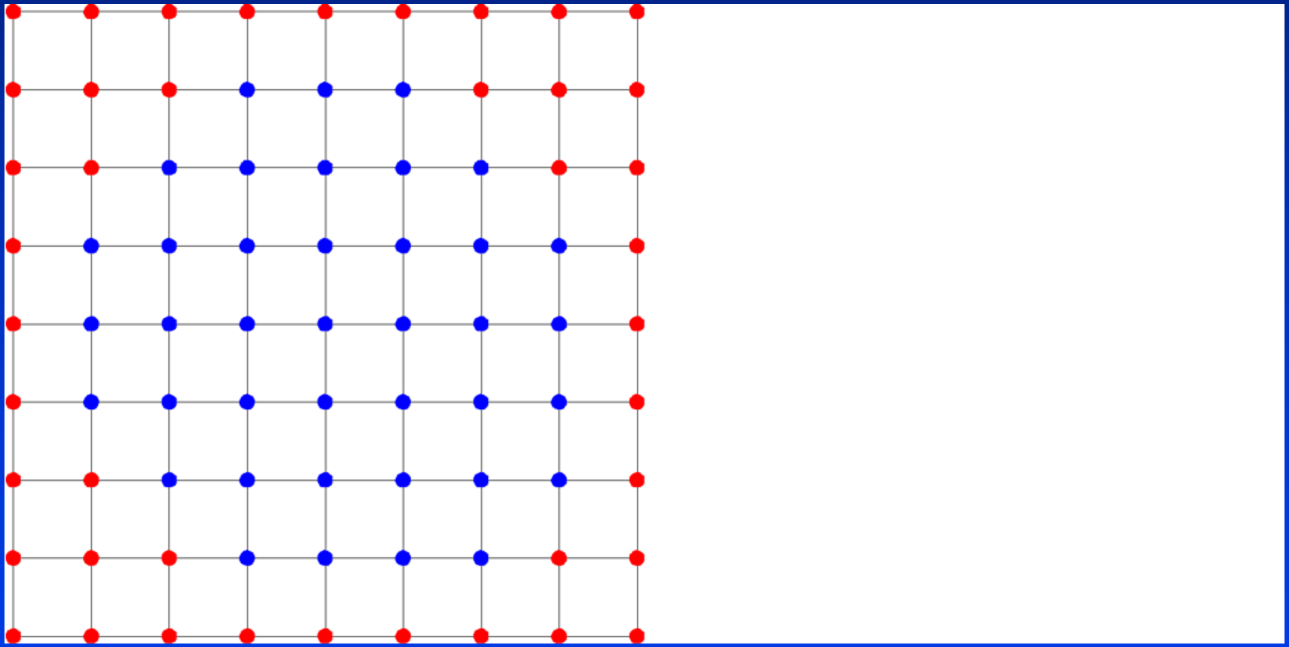


Voxel Display

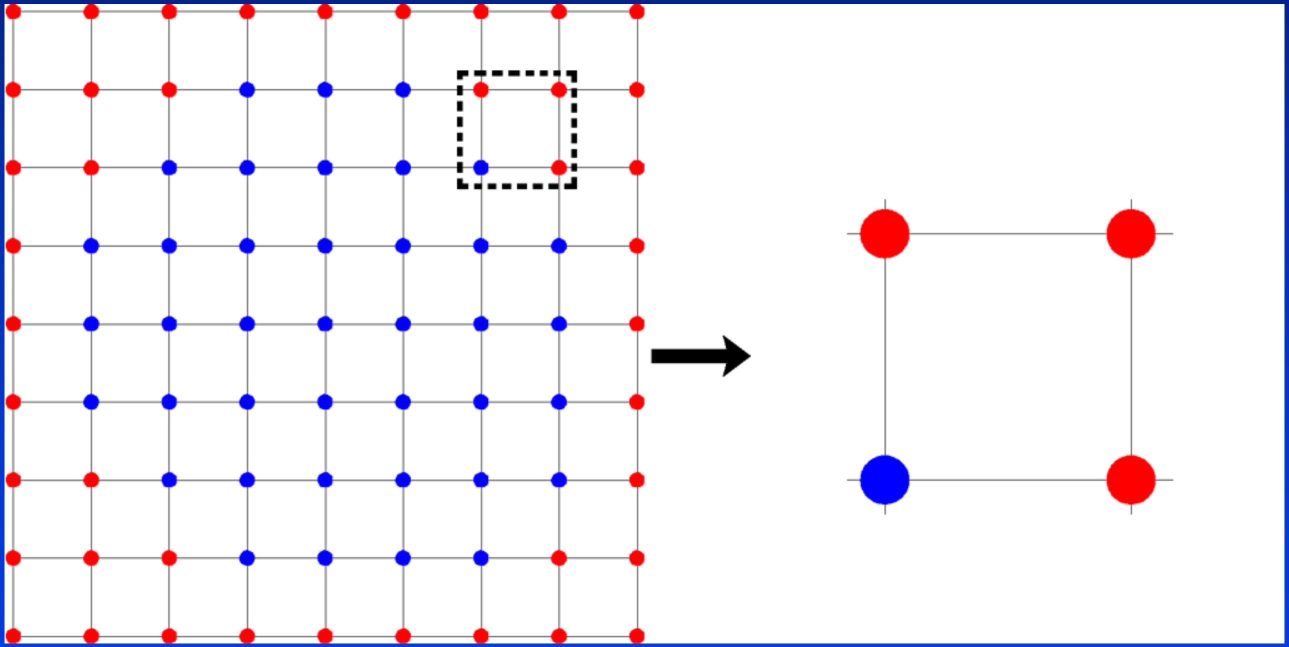
- **Slicing**
  - Draw 2D image resulting from intersecting voxels with a plane



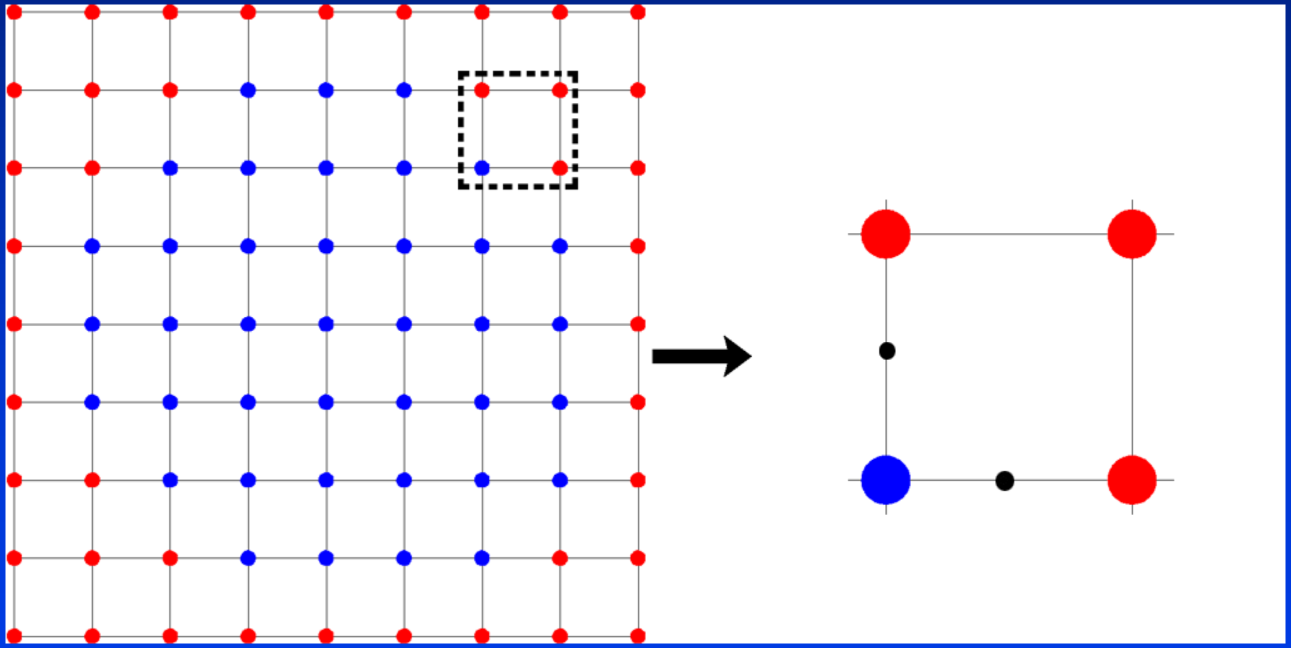
2D Polygon Generation



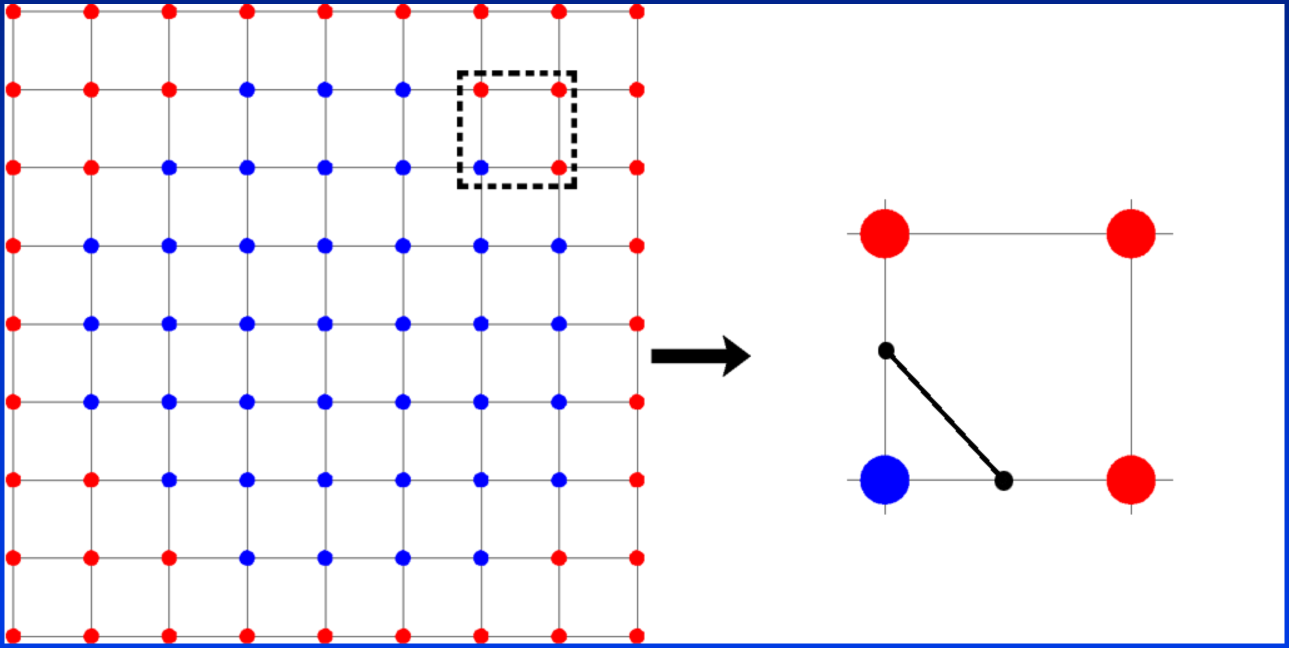
2D Polygon Generation



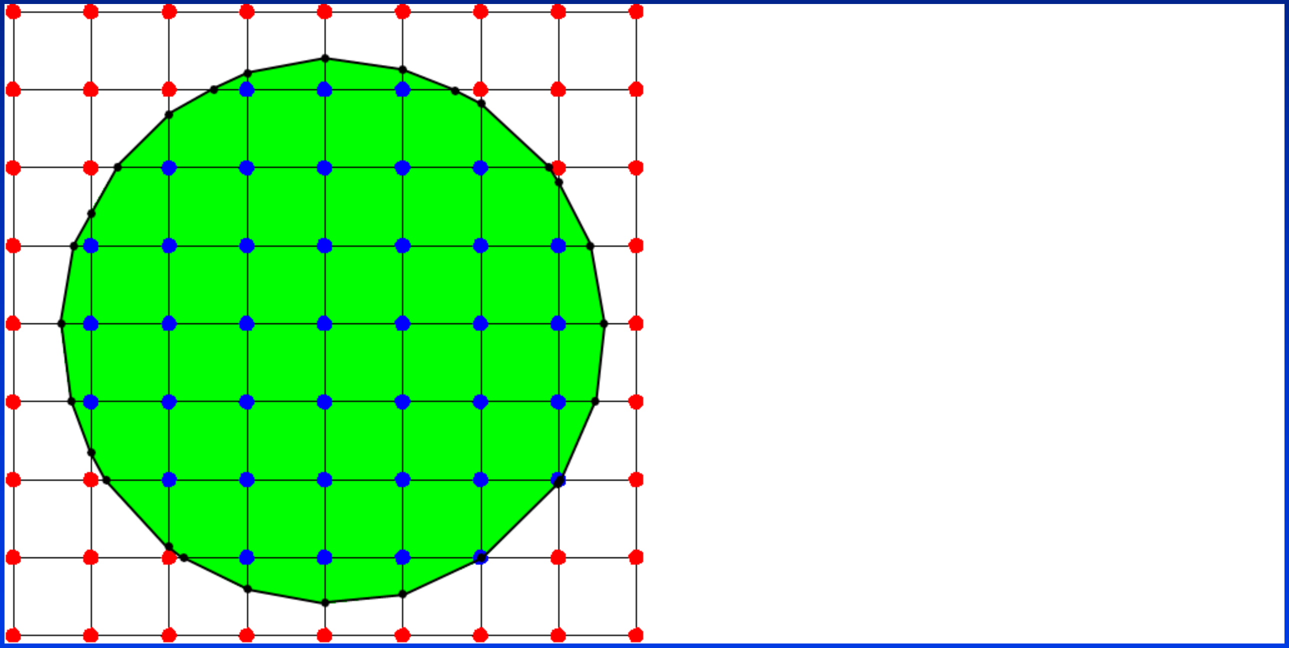
2D Polygon Generation



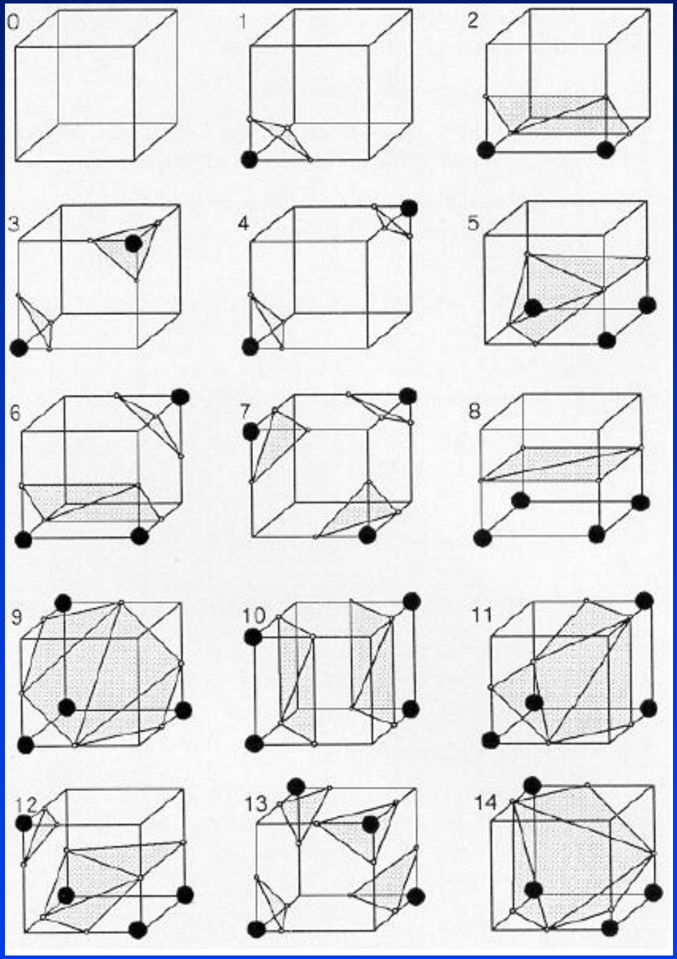
2D Polygon Generation

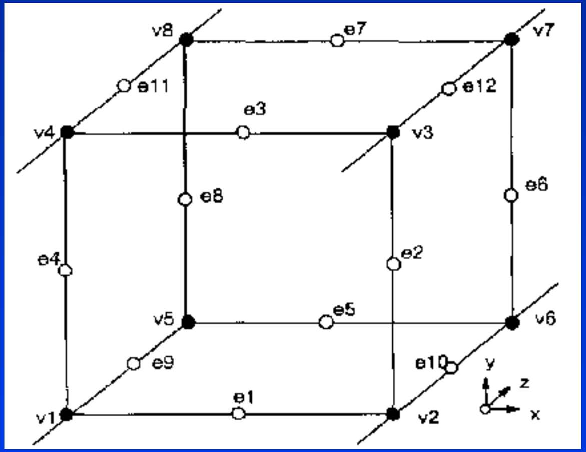


2D Polygon Generation

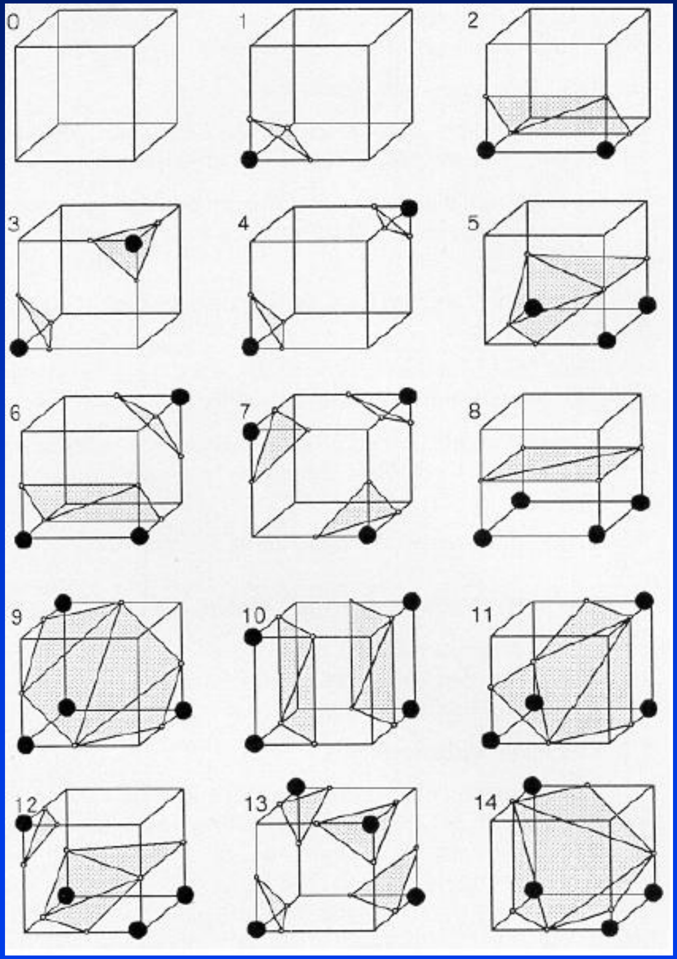


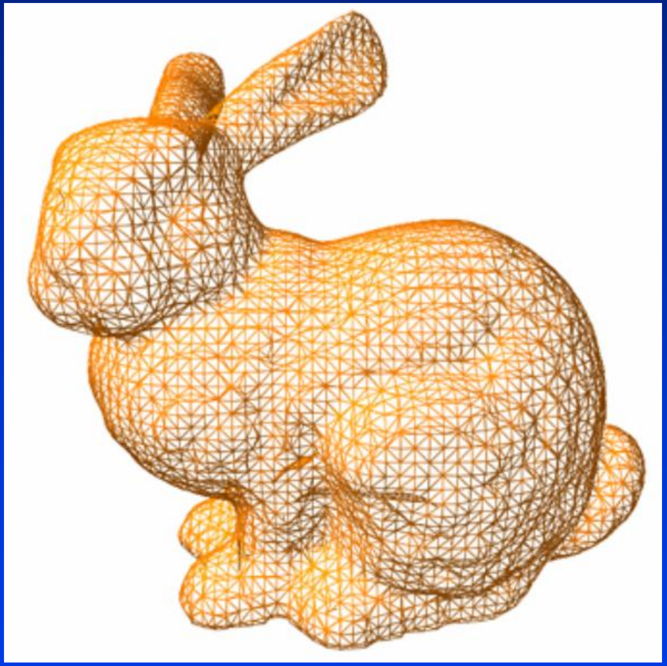
3D Polygon Generation





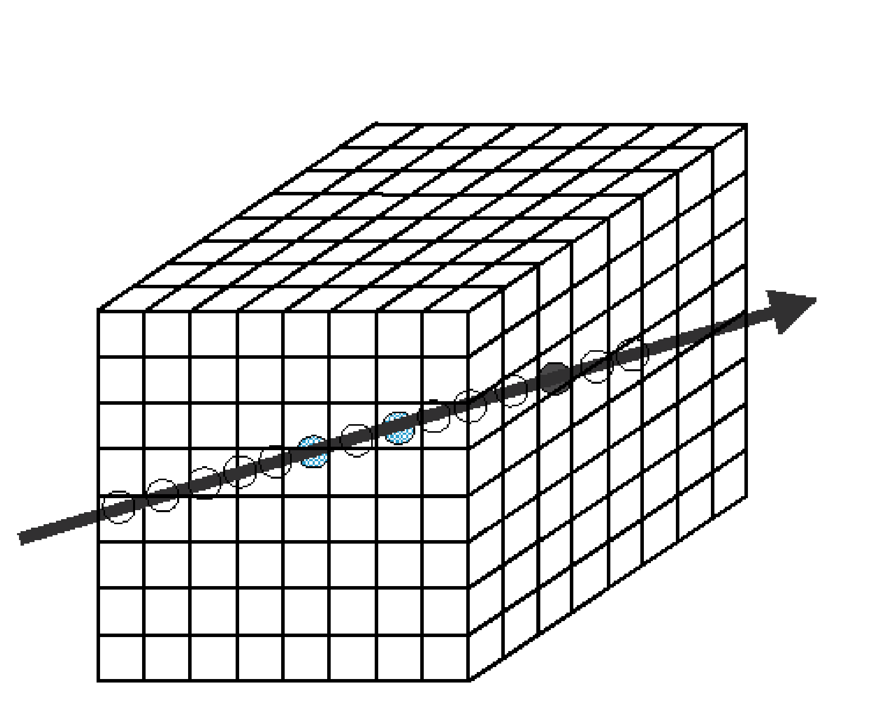
3D Polygon Generation

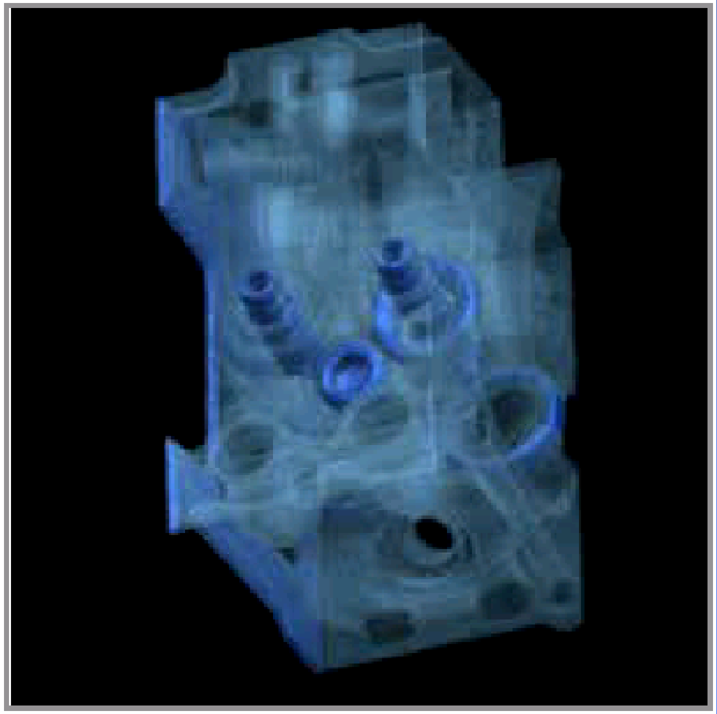




Voxel Display

- **Ray-casting**
  – Integrate density along rays through pixels





Voxels

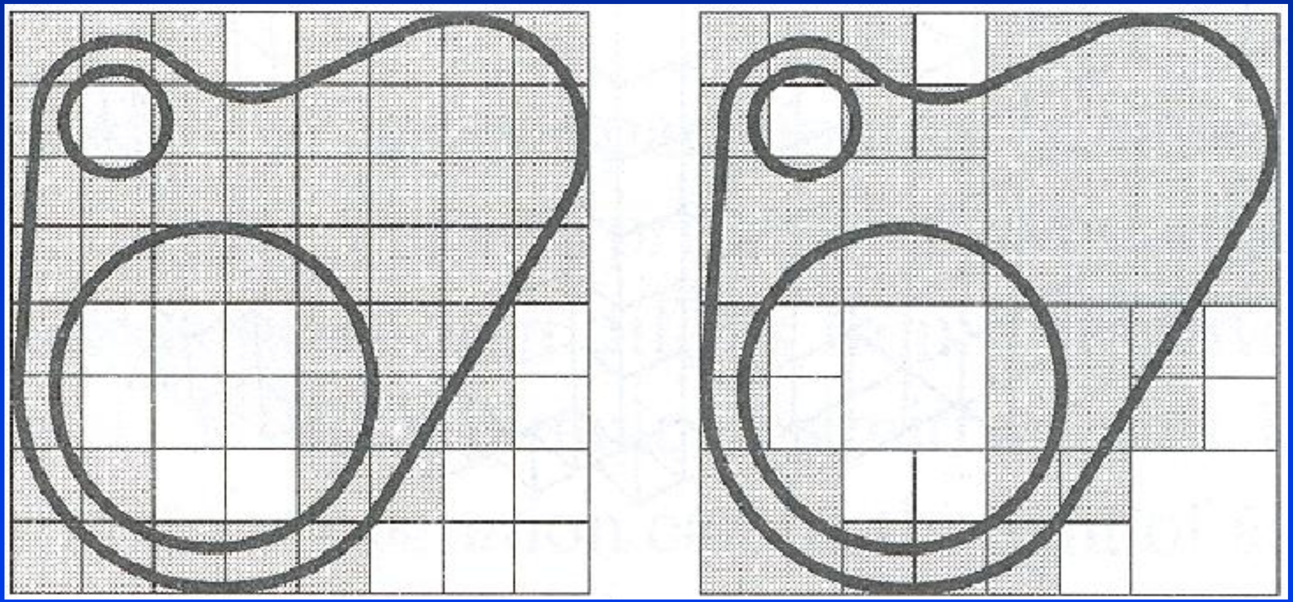
- **Advantages**
  - Simple, intuitive, unambiguous
  - Same complexity for all objects
  - Natural acquisition for some applications
  - Trivial Boolean operations
- **Disadvantages**
  - Approximation, not accurate
  - Large storage requirements
  - Expensive display

Solid Modeling Representation

- **Quadtrees & Octrees**

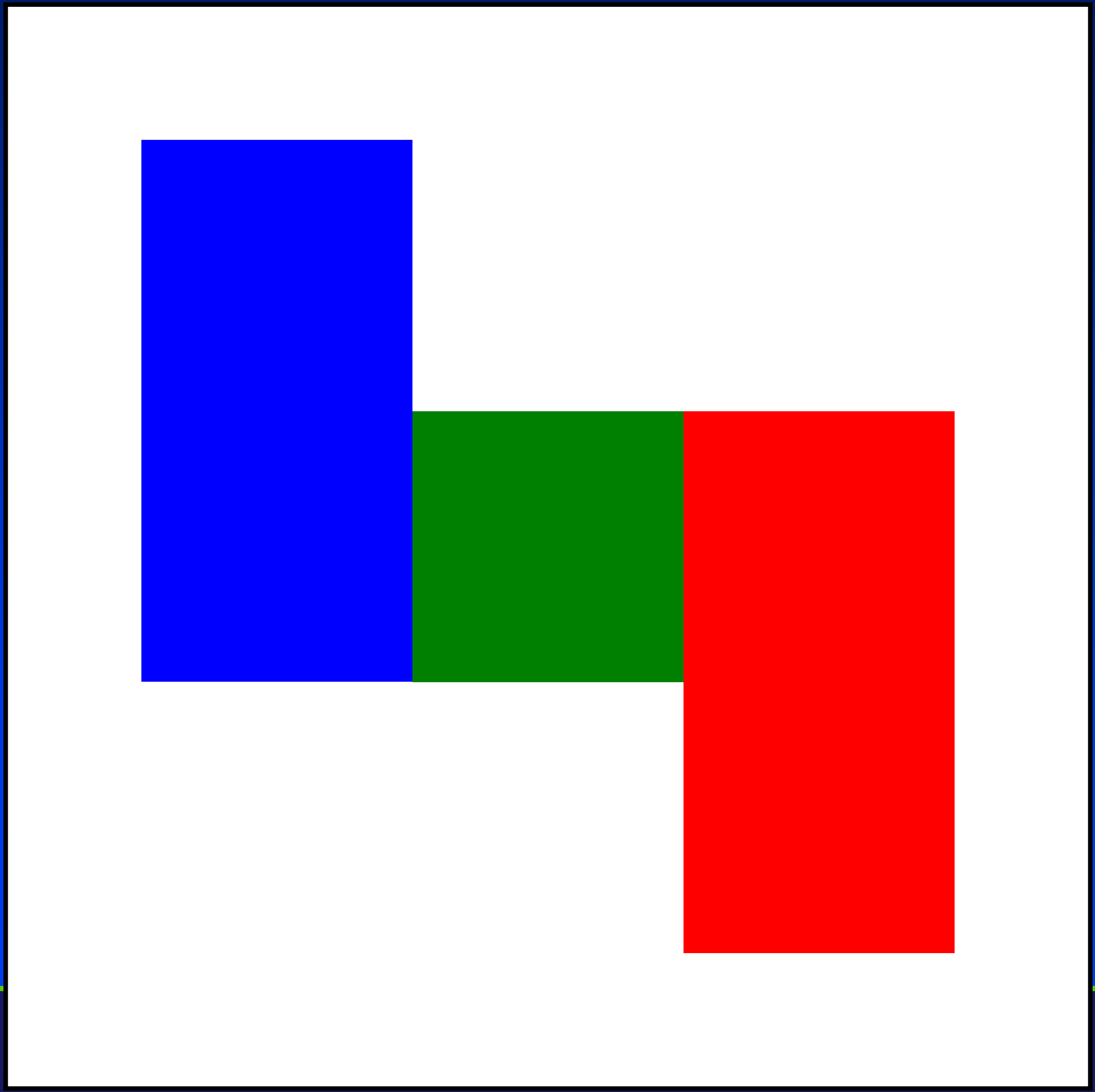
Quadtrees & Octrees

- **Refine resolution of voxels hierarchically**
  - More concise and efficient for non-uniform objects

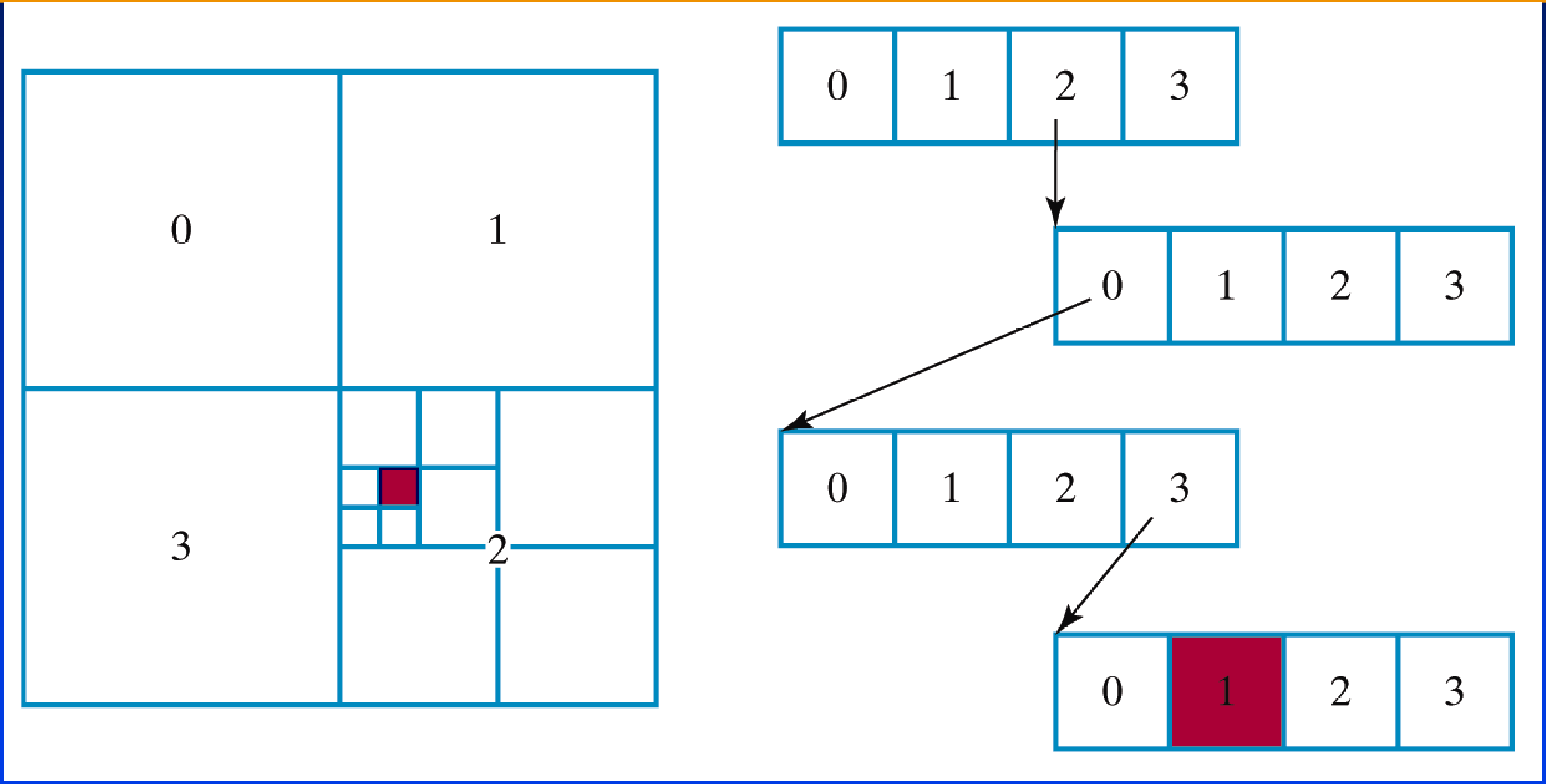


Uniform Voxel Quadtree

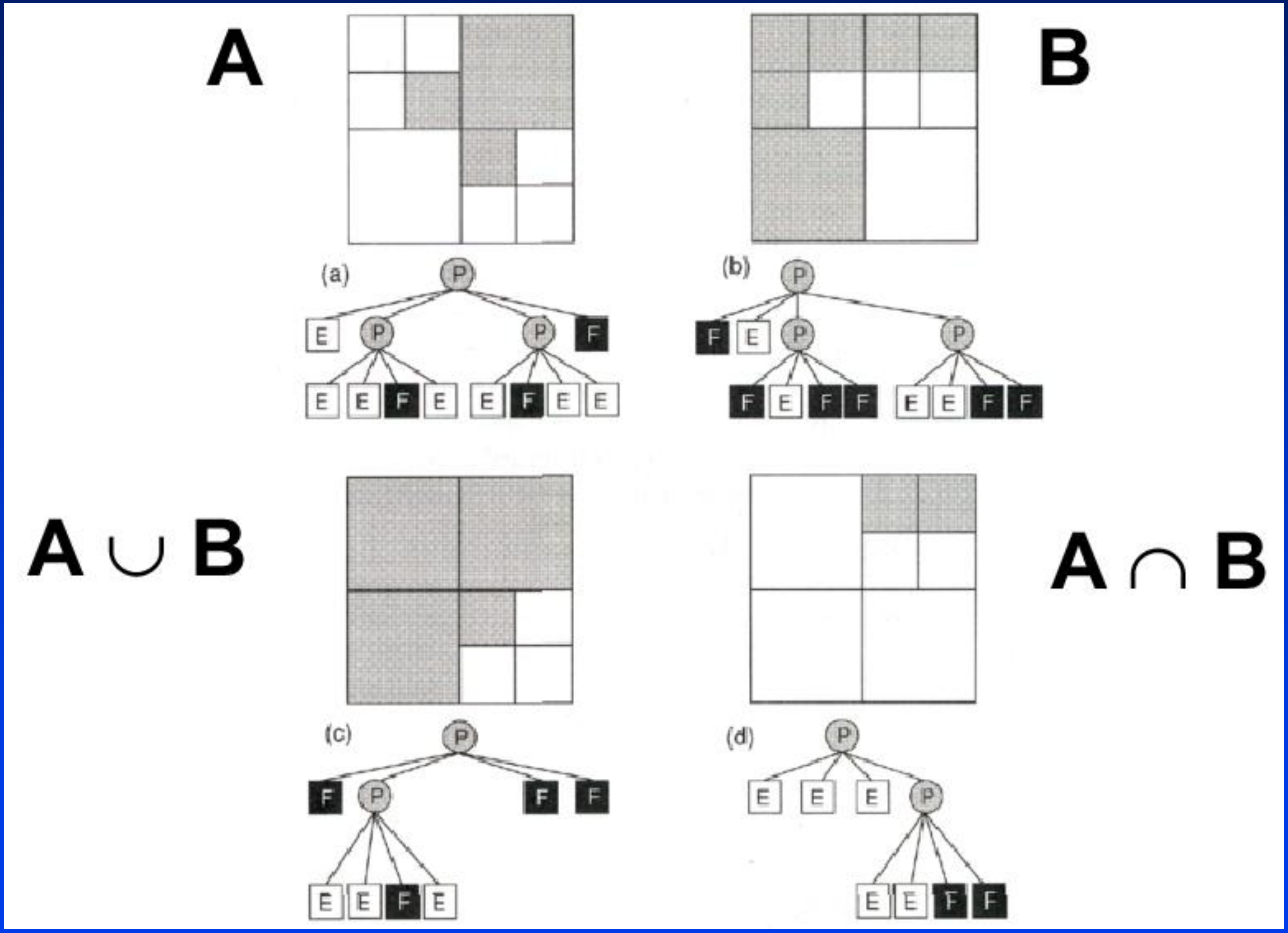
Quadtree



Quadtree



Quadtree Boolean Operations

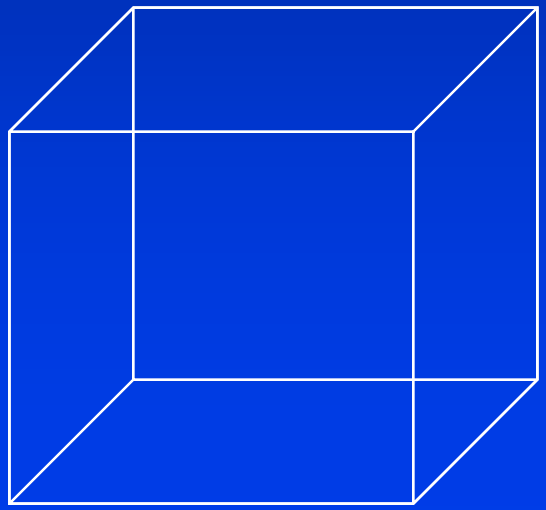


Octrees & Quadtrees

- Octrees are based on a two-dimensional representation scheme called **quadtree** encoding
- Quadtree encoding divides a square region of space into four equal areas until *homogeneous regions* are found
- These regions can then be arranged in a tree

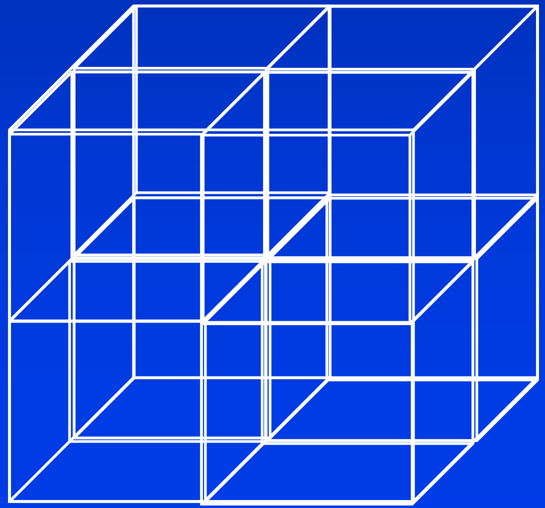
Octrees

- Model space as a tree with 8 children
- Nodes can be 3 types
  - Interior Nodes
  - Solid
  - Empty

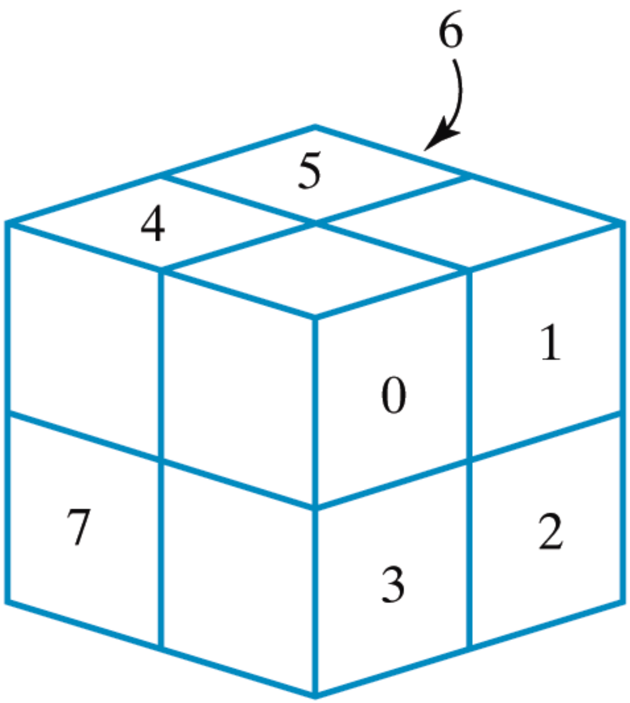


Octrees

- Model space as a tree with 8 children
- Nodes can be 3 types
  - Interior Nodes
  - Solid
  - Empty



Octrees

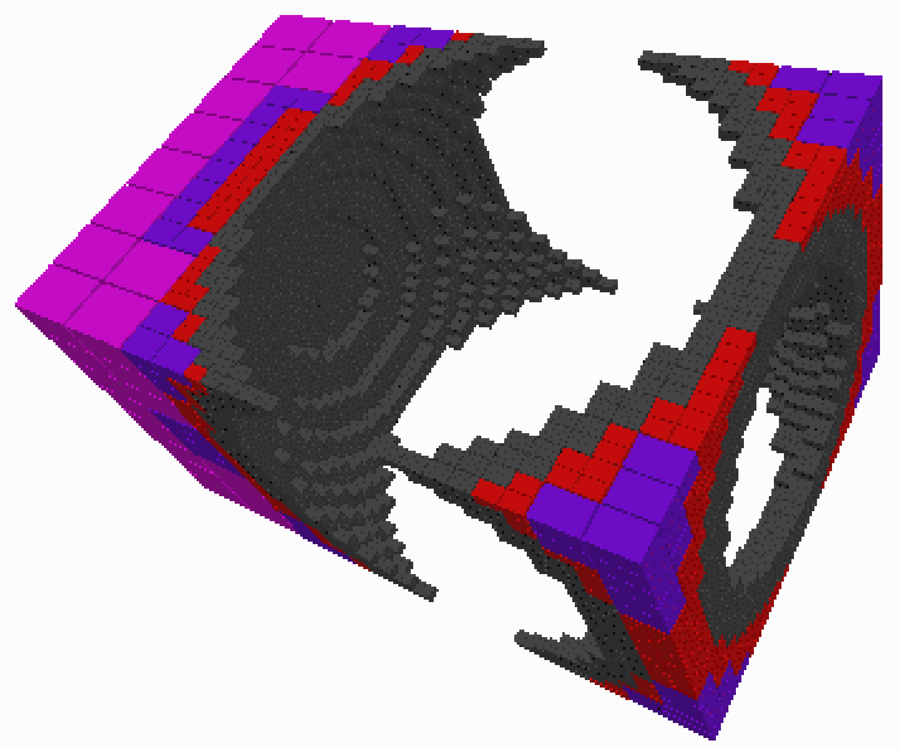


Region of a
Three-Dimensional
Space

Data Elements
in the Representative
Octree Node

Octrees

- Octrees are hierarchical tree structures used to represent solid objects
- Octrees are particularly useful in applications that require cross sectional views – for example medical applications
- Octrees are typically used when the interior of objects is important



Octrees

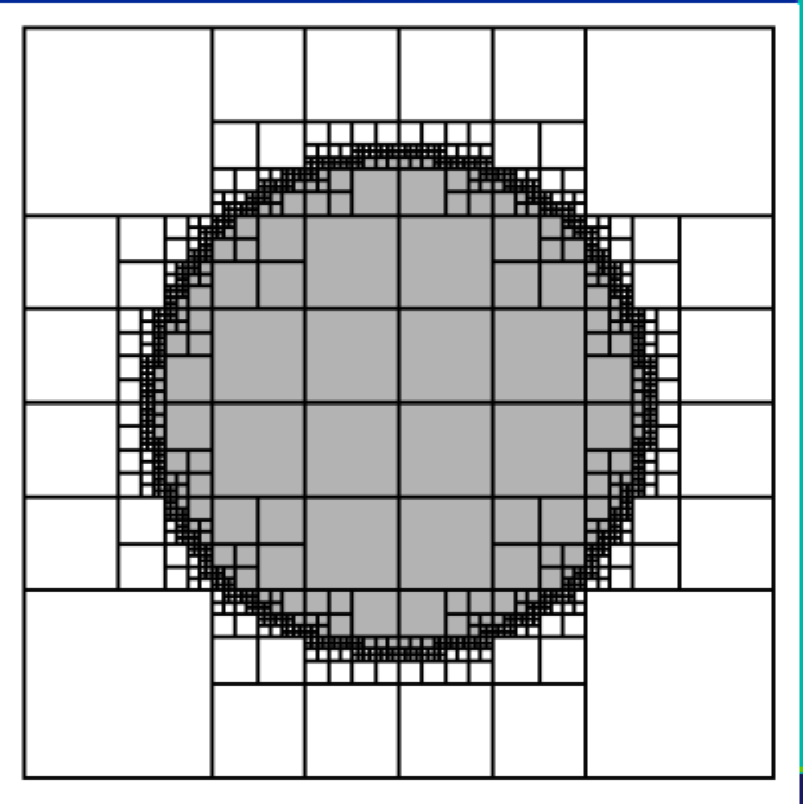
- Quadtree encodings provide considerable savings in storage when large colour areas exist in a region of space
- An octree takes the same approach as quadtrees, but divides a cube region of 3D space into octants
- Each region within an octree is referred to as a **volume element** or **voxel**
- Division is continued until homogeneous regions are discovered

Octrees

- In 3 dimensions regions can be considered to be homogeneous in terms of color, material type, density or any other physical characteristics
- Voxels also have the unique possibility of being *empty*

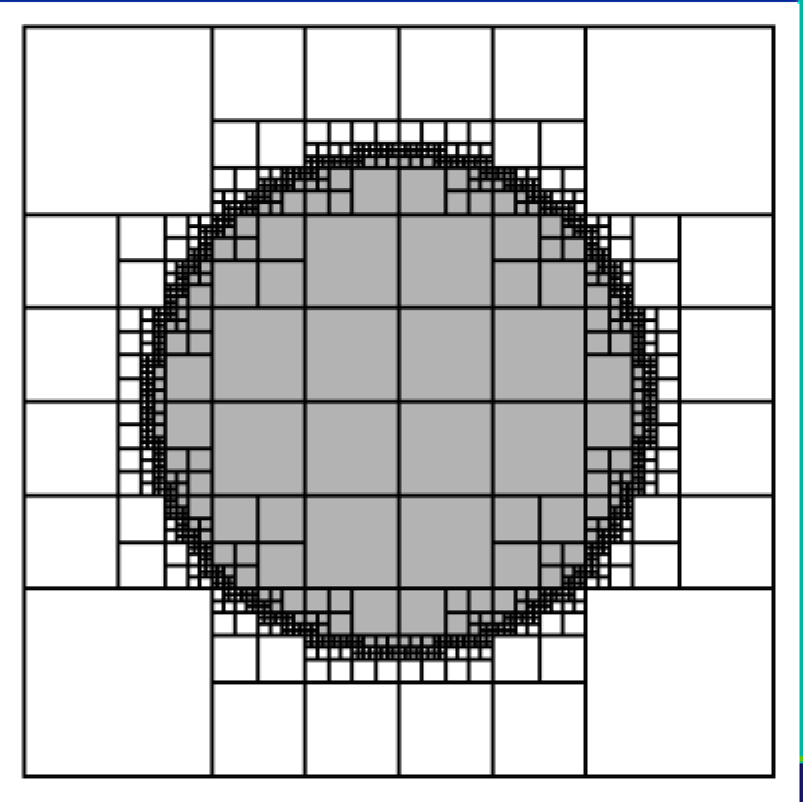
Building Octrees

- If cube completely inside, return solid node
- If cube completely outside, return empty node
- Otherwise recursion until maximum depth reached



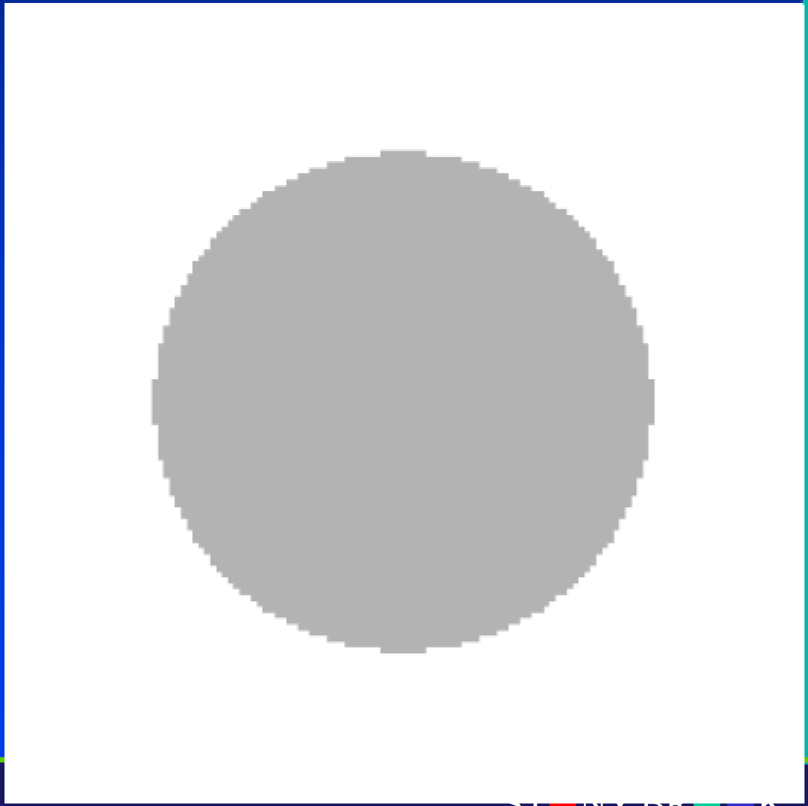
Octrees

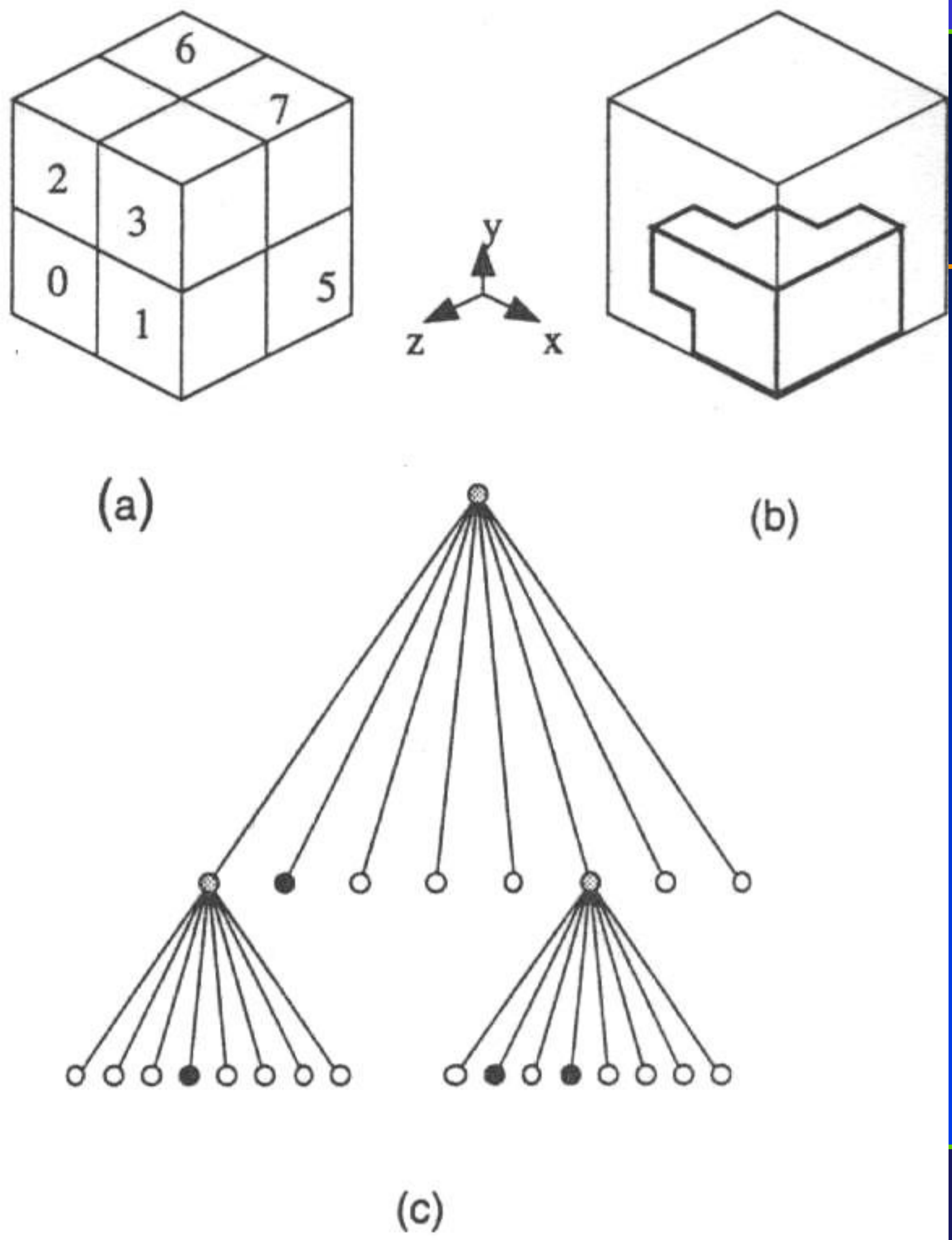
- Advantages
  - Storage space proportional to surface area
  - Inside/Outside trivial
  - Volume trivial
  - CSG relatively simple
  - Can approximate any shape
- Disadvantages
  - Blocky appearance



Octrees

- Advantages
  - Storage space proportional to surface area
  - Inside/Outside trivial
  - Volume trivial
  - CSG relatively simple
  - Can approximate any shape
- Disadvantages
  - Blocky appearance



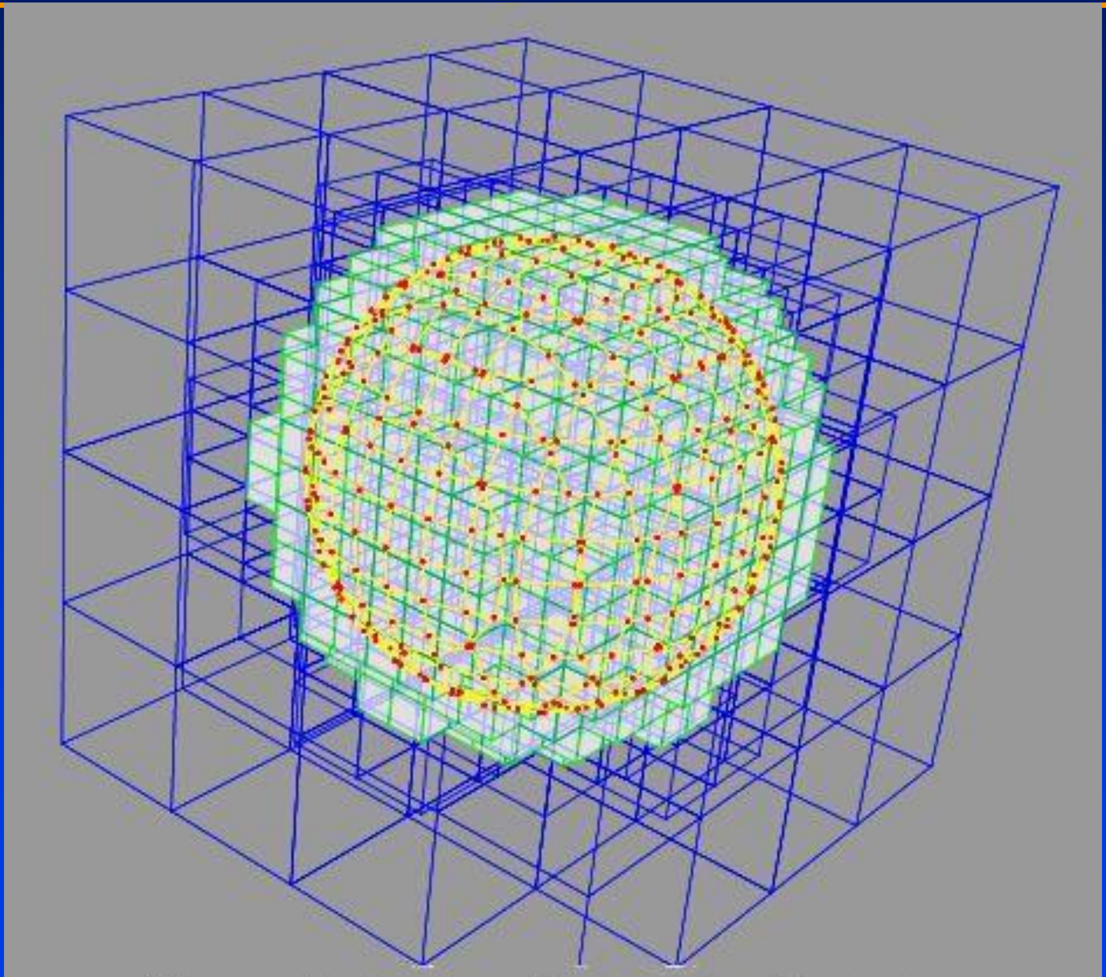


Octree Example

Octree Data Structure

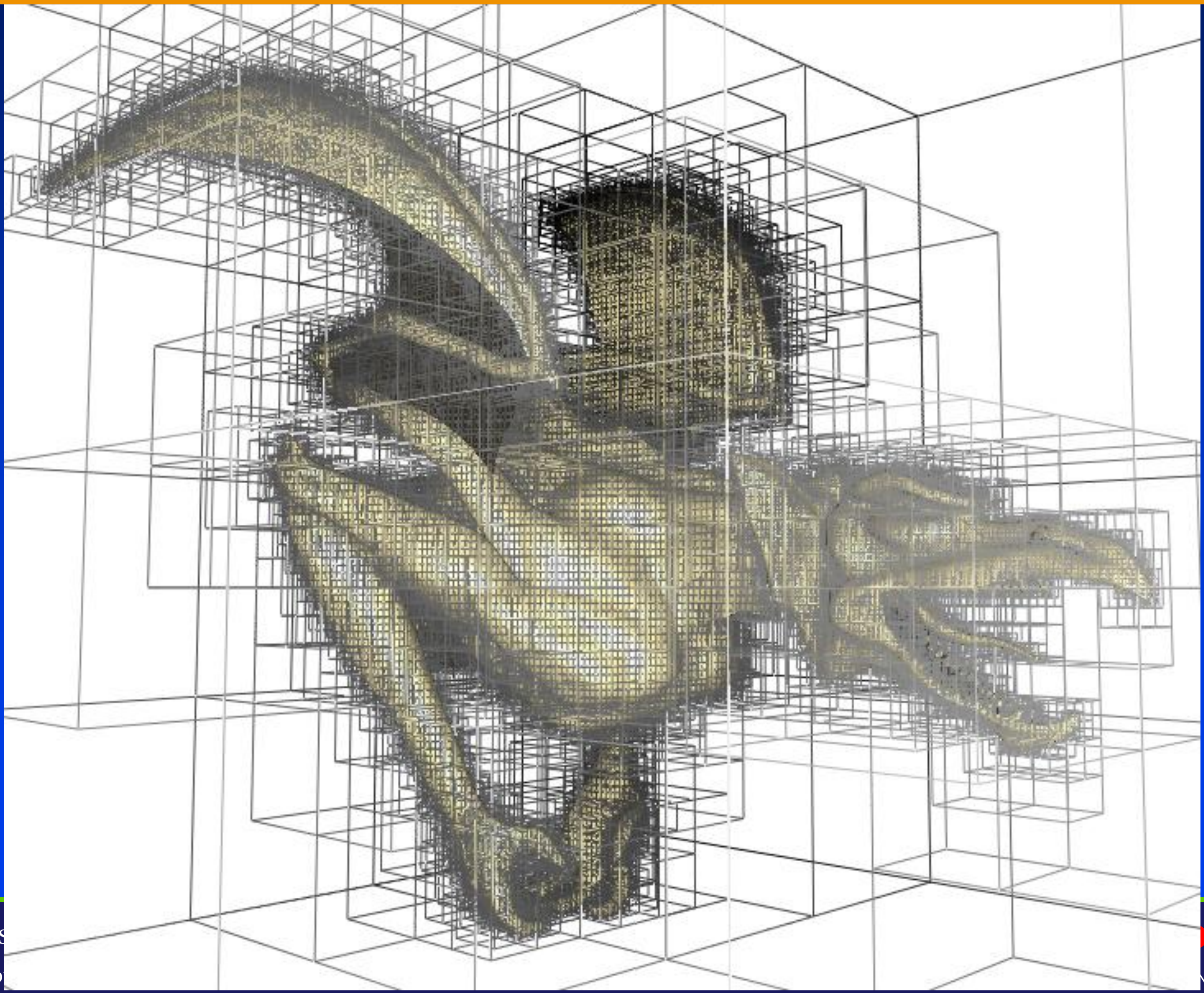
```
struct octreeroot
{
   float xmin, ymin, zmin;     /* space of interest */
   float xmax, ymax, zmax;
   struct octree  *root;/* root of the tree */
};

struct octree
{
   char  code;                 /* BLACK, WHITE, GRAY */
   struct octree  *oct[8];     /* pointers to octants, present if GRAY */
};
```

Octree Examples



Octree containing pieces of an implicitly defined sphere; within each terminal node surface vertices are computed and connected to form a polygon

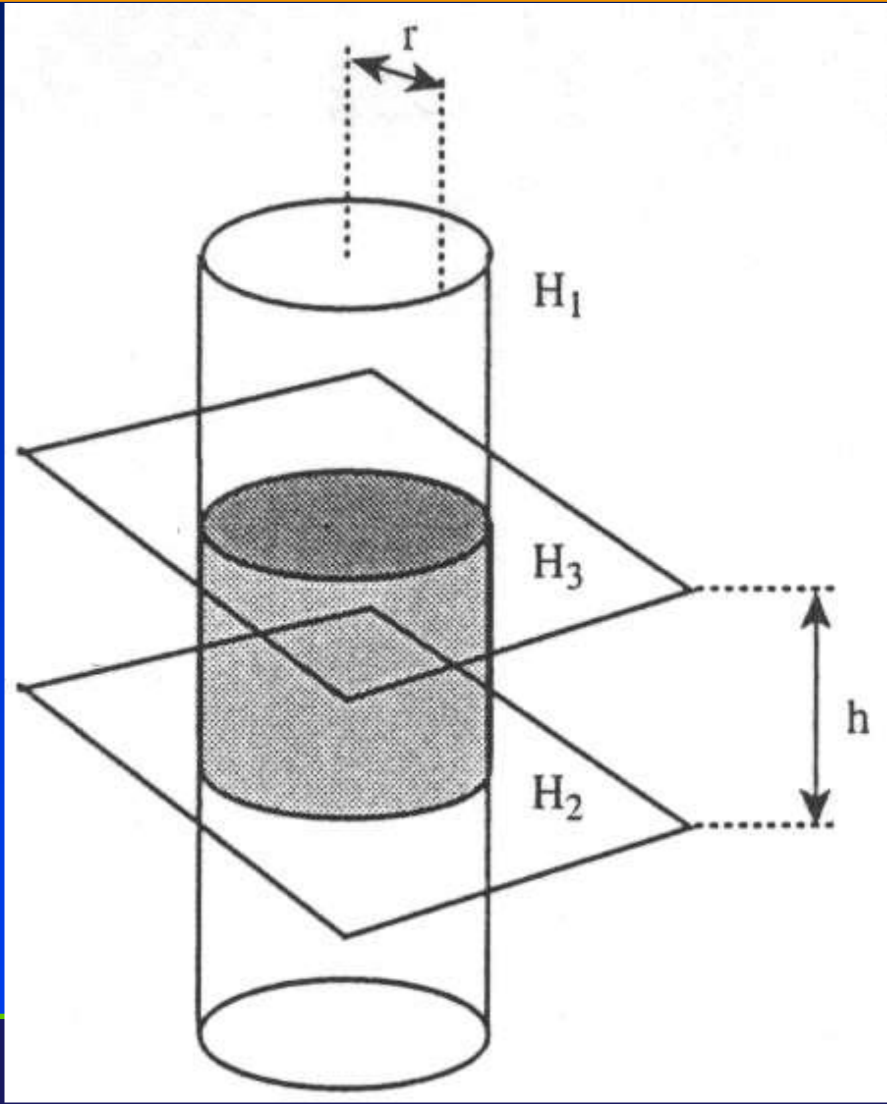
Octree Examples



Solid Modeling Representation

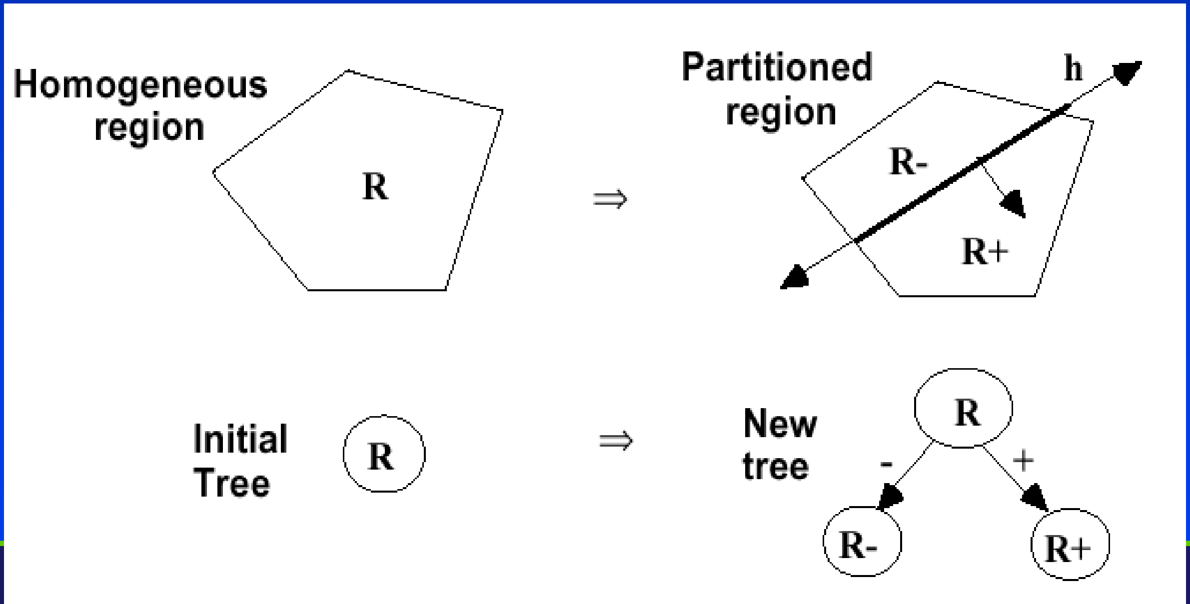
- **Binary Space Partitions**

Half Space Model



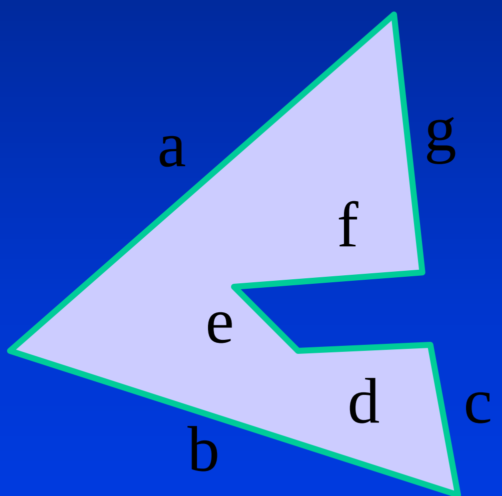
BSP Fundamentals

- **Single geometric operation**
  - Partition a convex region by a hyper-plane
- **Single combinatorial operation**
  - Two child nodes added as leaf nodes

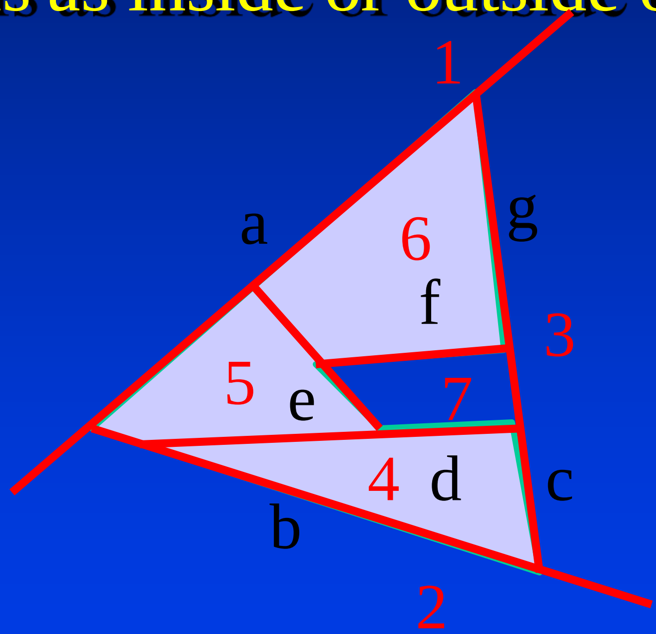


Binary Space Partitions (BSPs)

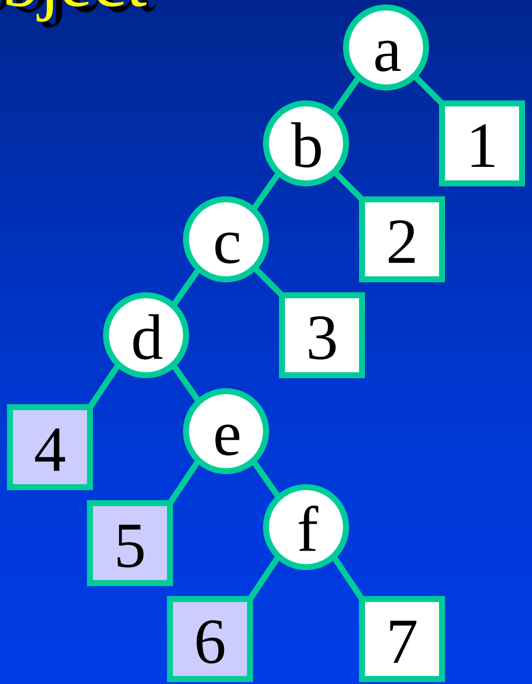
- **Recursive partition of space by Planes**
  - Mark leaf cells as inside or outside object



Object



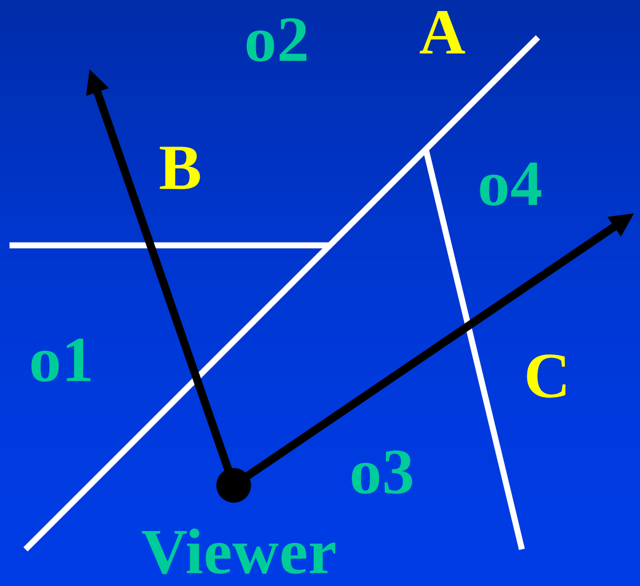
Binary Spatial Partition

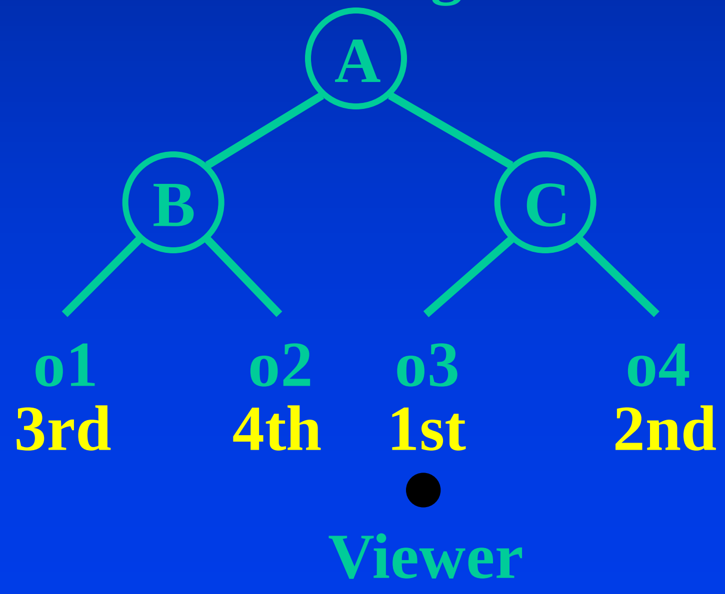


BSP Tree

BSP Display

- **Visibility Ordering**
  - Determine on which side of plane the viewer lies
    - Near-subtree -> polygons on split -> far-subtree





Summary

	Voxels	Octree	BSP	CSG
Accurate	No	No	Some	Some
Concise	No	No	No	Yes
Affine Invariant	No	No	Yes	Yes
Easy Acquisition	Some	Some	No	Some
Guaranteed Validity	Yes	Yes	Yes	No
Efficient Boolean Operations	Yes	Yes	Yes	Yes
Efficient Display	No	No	Yes	No

New Solid Modeling Techniques:
(Sketch-Based Solid Modeling with BlobTrees)

Implicit Representation of Shape

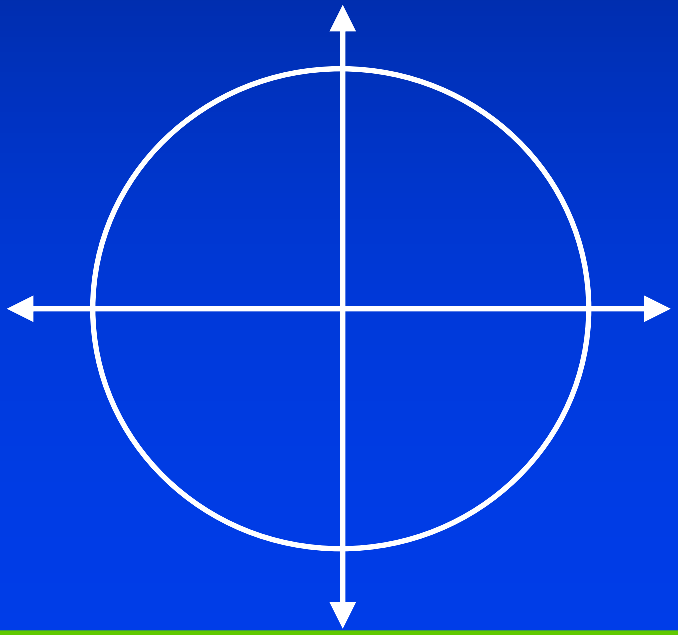
- Shape described by solution to $f(x)=c$

$$f(x, y) = x^2 + y^2 - 9$$

Implicit Representation of Shape

- Shape described by solution to $f(x)=c$

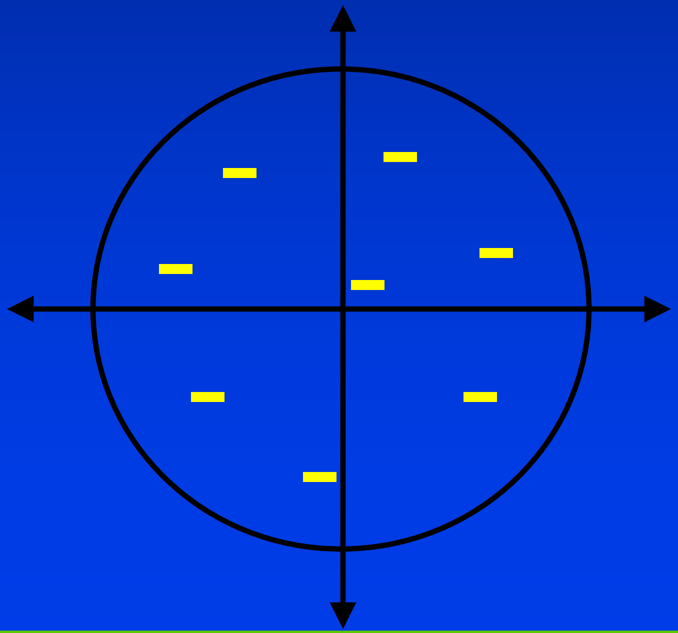
$$f(x,y) = x^2 + y^2 - 9$$



Implicit Representation of Shape

- Shape described by solution to $f(x)=c$

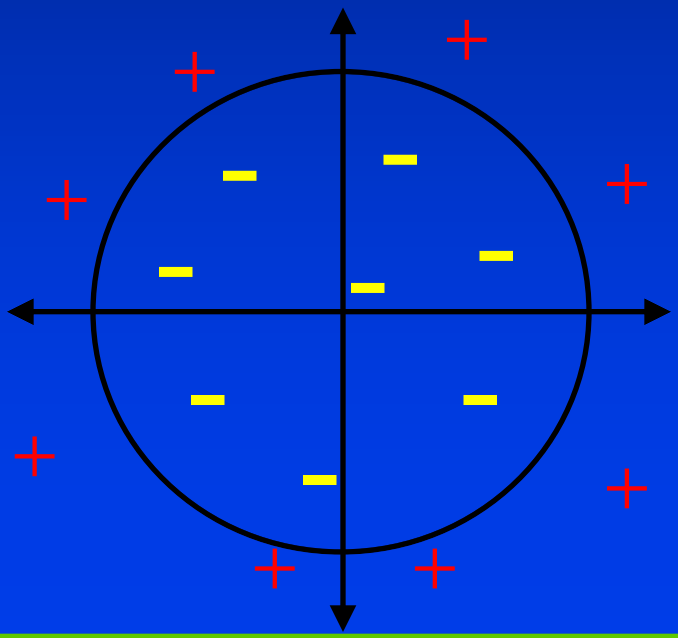
$$f(x,y)=x^2+y^2-9$$



Implicit Representation of Shape

- Shape described by solution to $f(x)=c$

$$f(x,y)=x^2+y^2-9$$

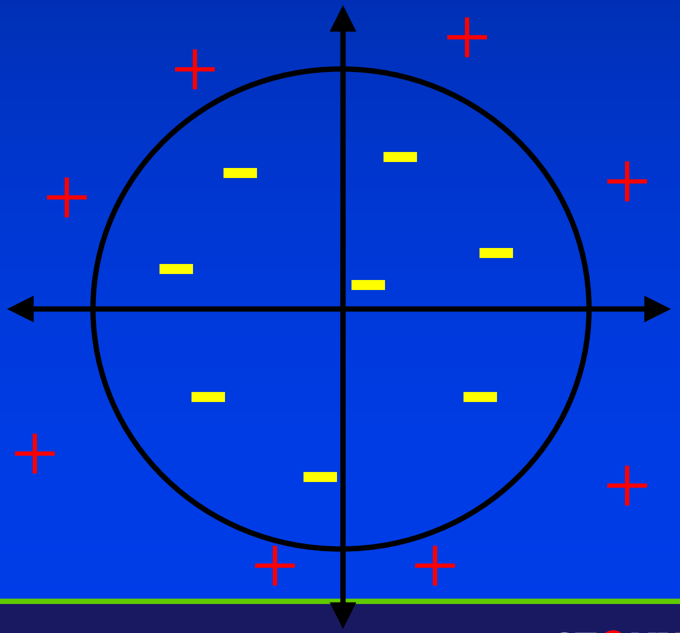


Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations

Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations

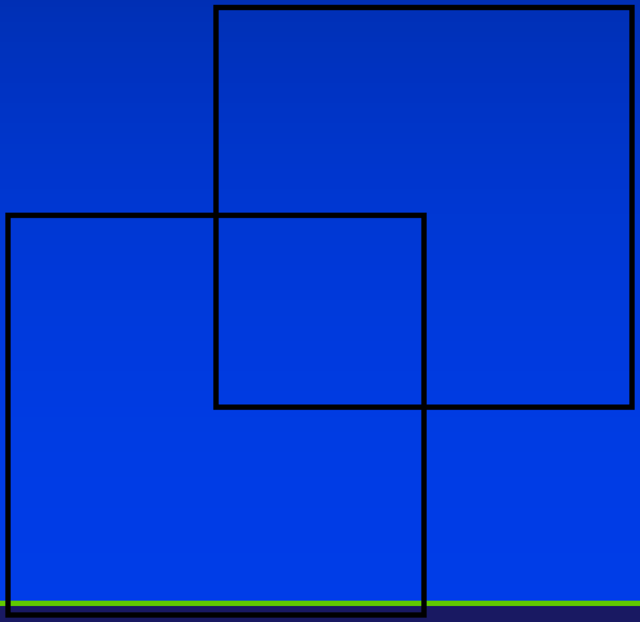


Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations

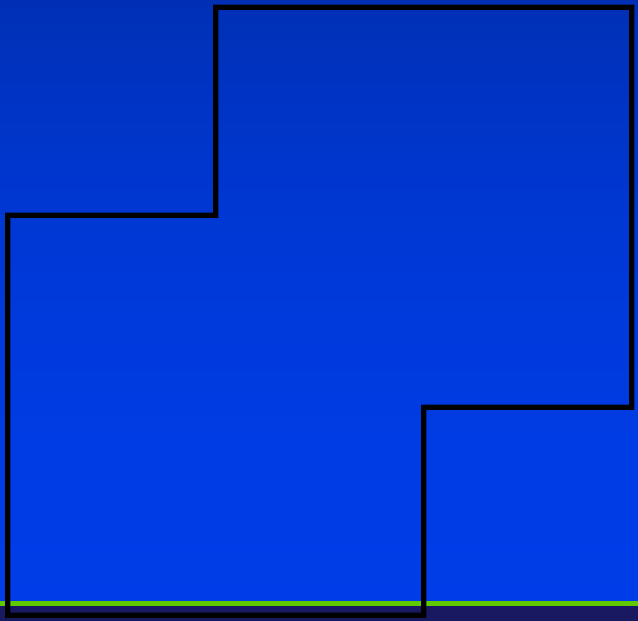
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union



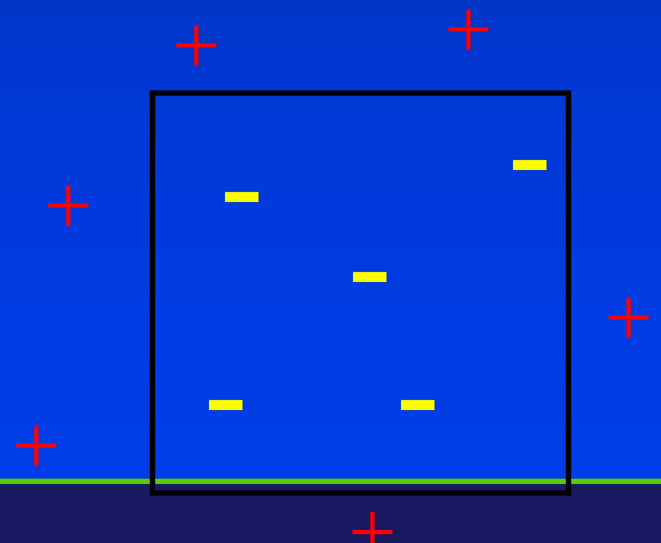
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union



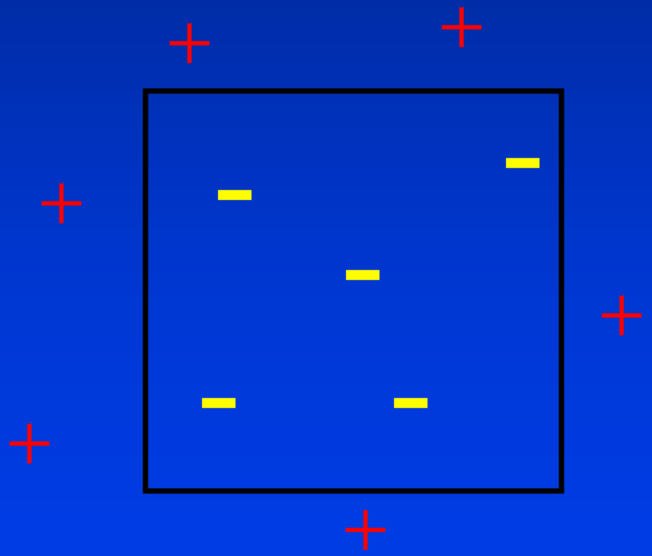
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union



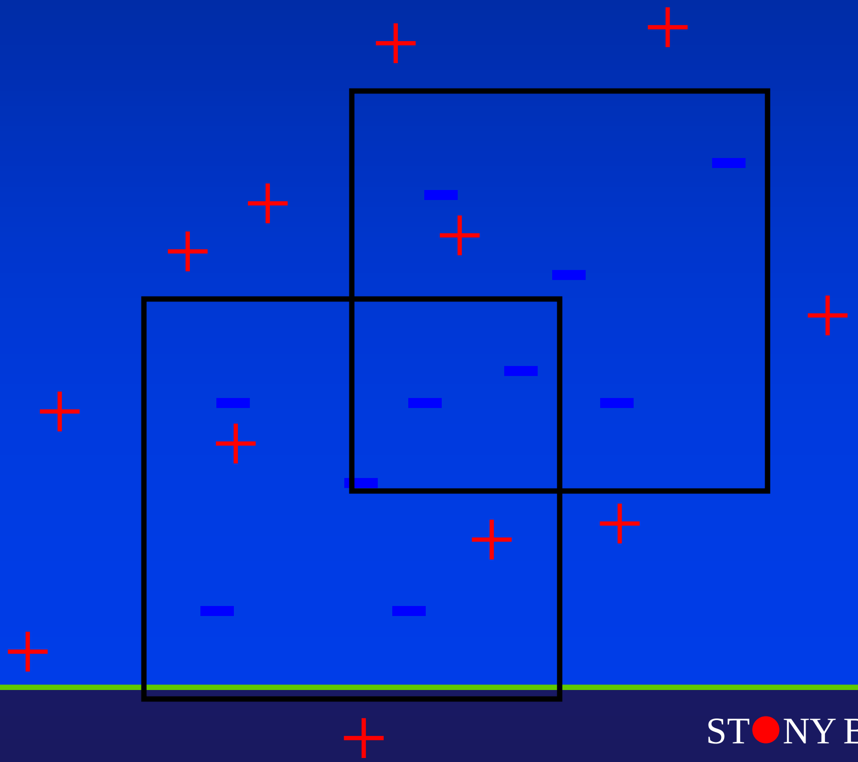
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union



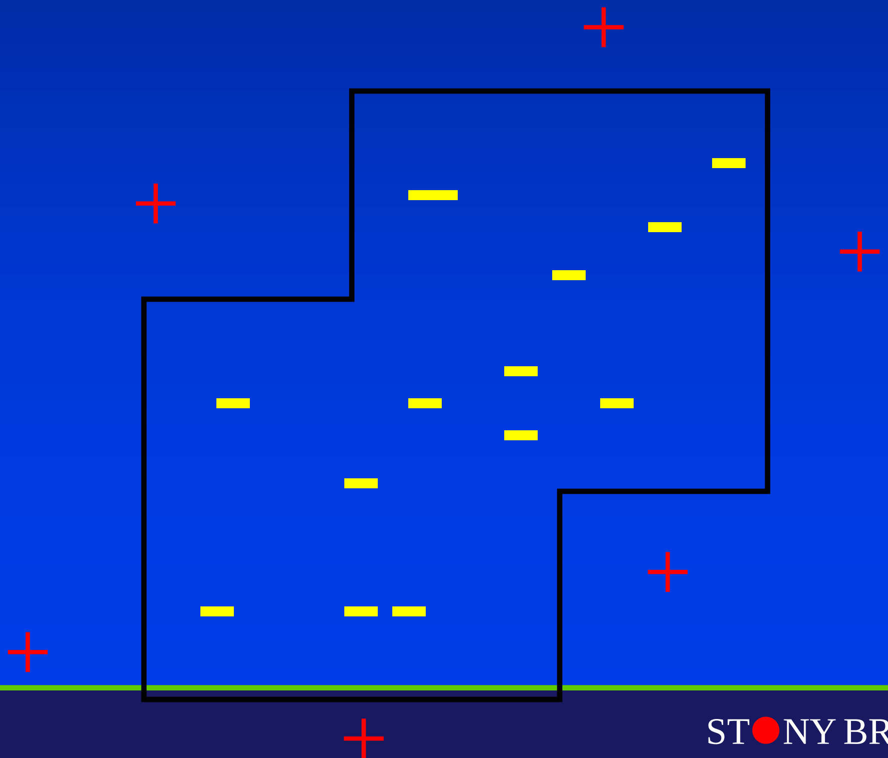
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union



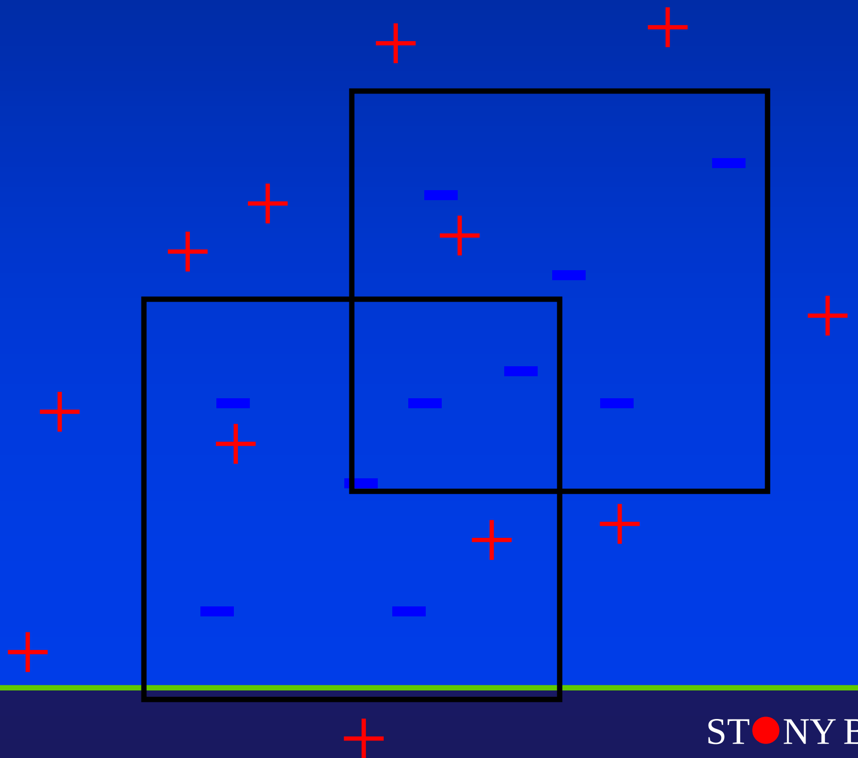
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union



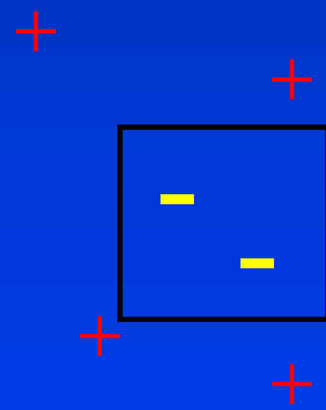
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union
  - Intersection



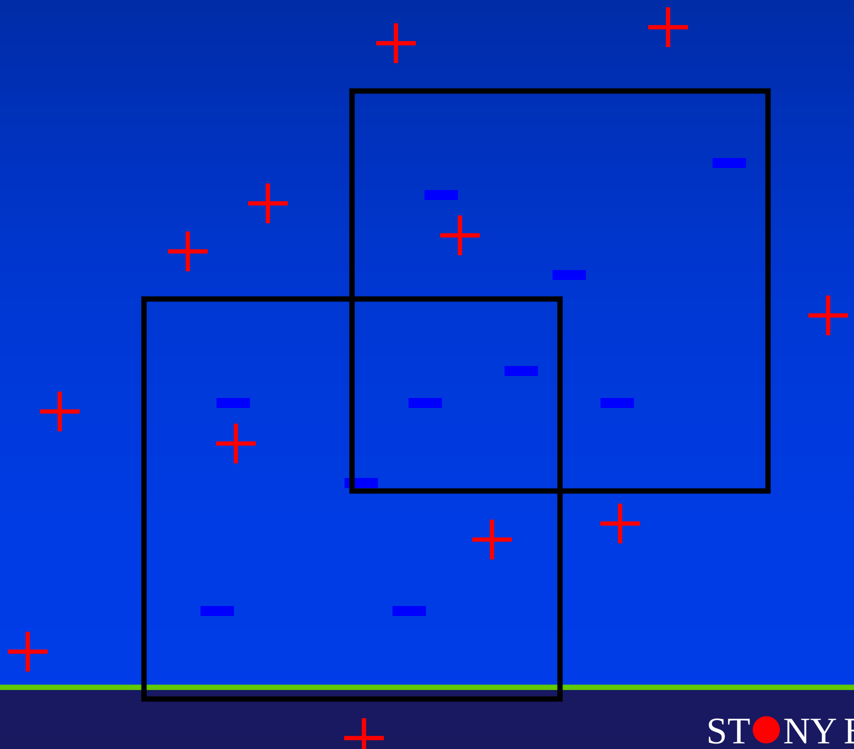
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union
  - Intersection



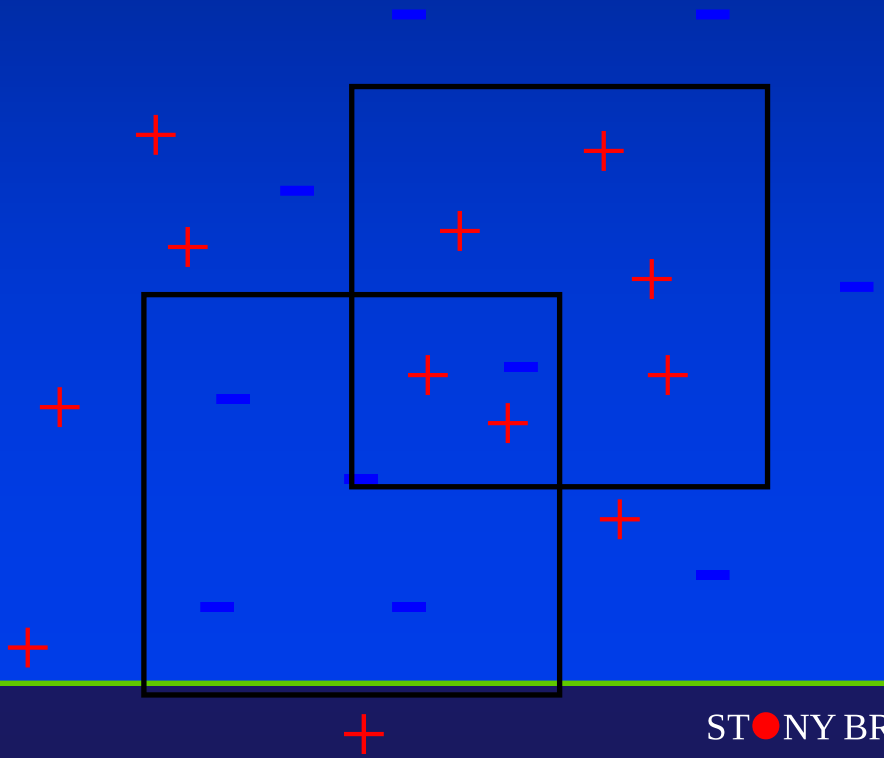
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union
  - Intersection
  - Subtraction



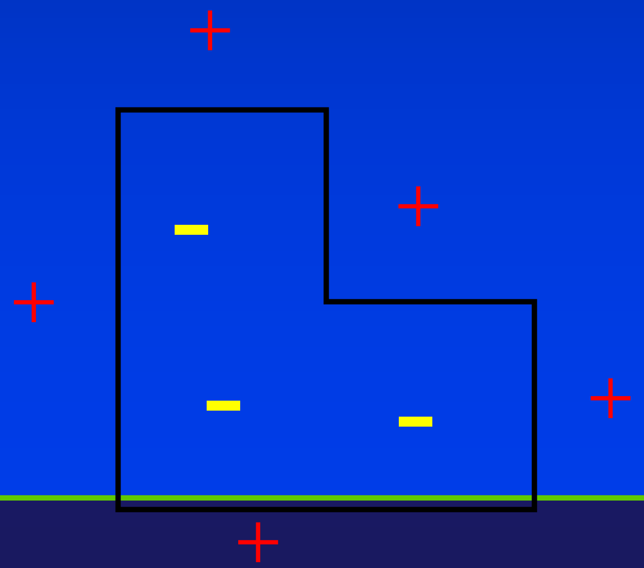
Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union
  - Intersection
  - Subtraction



Advantages

- No topology to maintain
- Always defines a closed surface!
- Inside/Outside test
- CSG operations
  - Union
  - Intersection
  - Subtraction

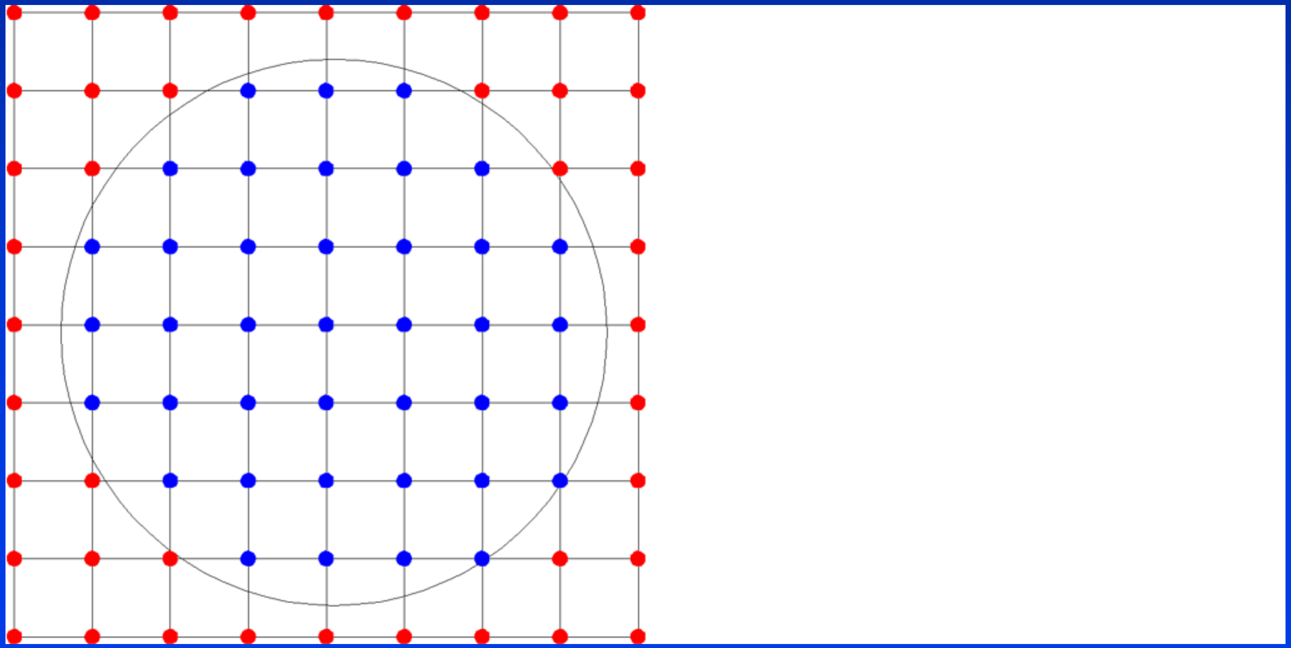


Disadvantages

- Hard to render - no polygons
- Creating polygons amounts to root finding
- Arbitrary shapes hard to represent as a function

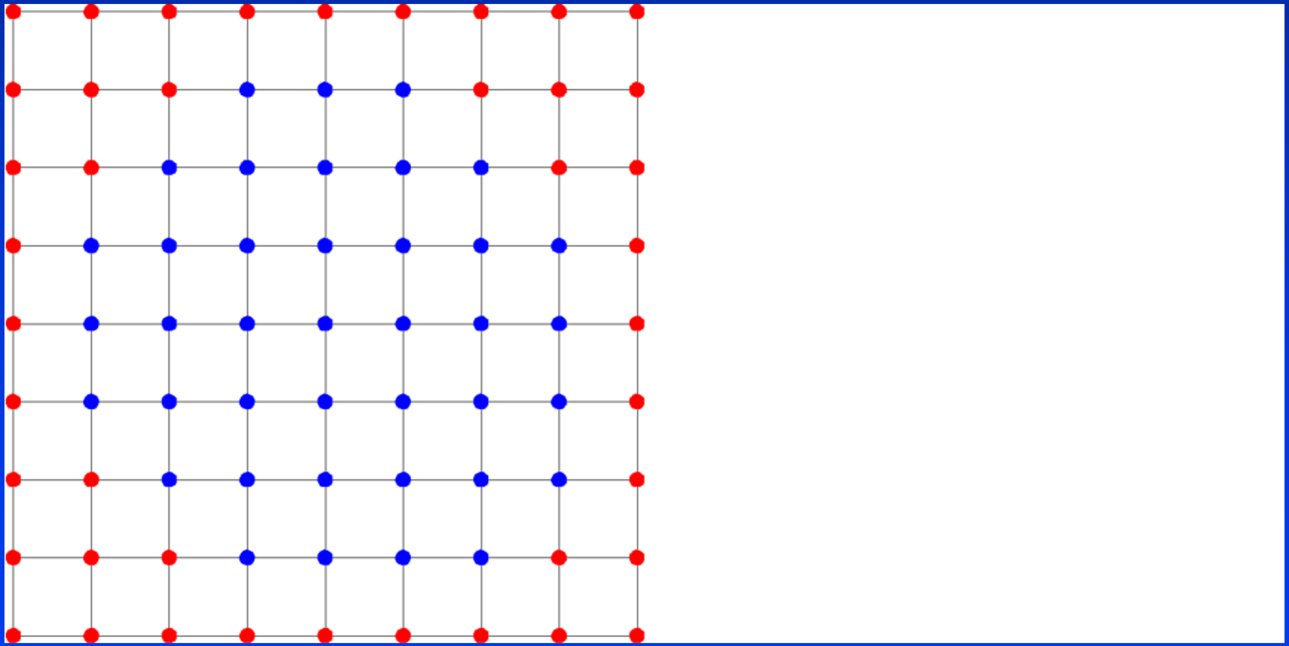
Non-Analytic Implicit Functions

- Sample functions over grids



Non-Analytic Implicit Functions

- Sample functions over grids



Sketch-Based 3D Modeling System ?

Key Concept:

Anyone can create 3D models

Method:

3D modeling from sketched 2D strokes

Technical Challenges

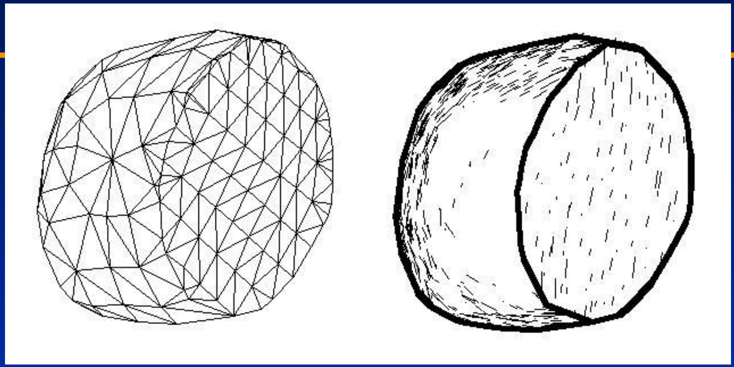
- A sketch-based modeling system
  - Easy
  - Interactive

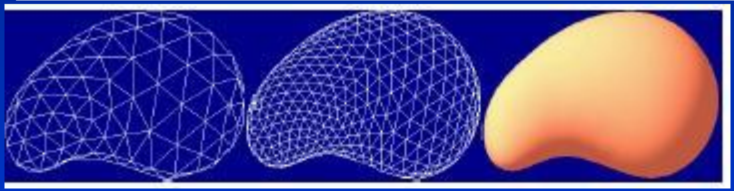
Problem:

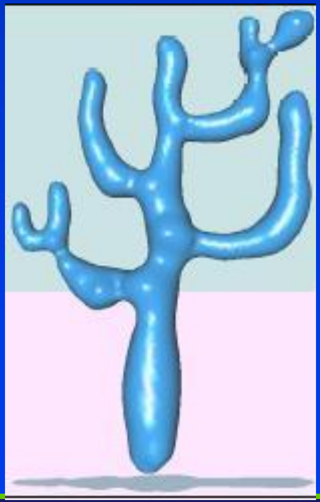
It is difficult to support complex models

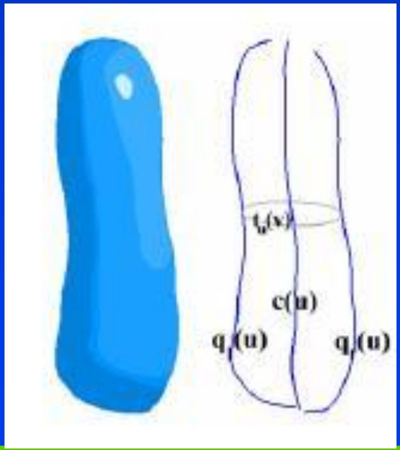
Various Kinds of Sketch-Based Modeling Systems

- Triangle meshes
- Subdivision surfaces
- Implicit surfaces
- Parametric surfaces









Teddy

- Triangle meshes
- Chordal axis

△Low complex models

Implicit Approaches

- Blending operation

△A Large matrix must be solved

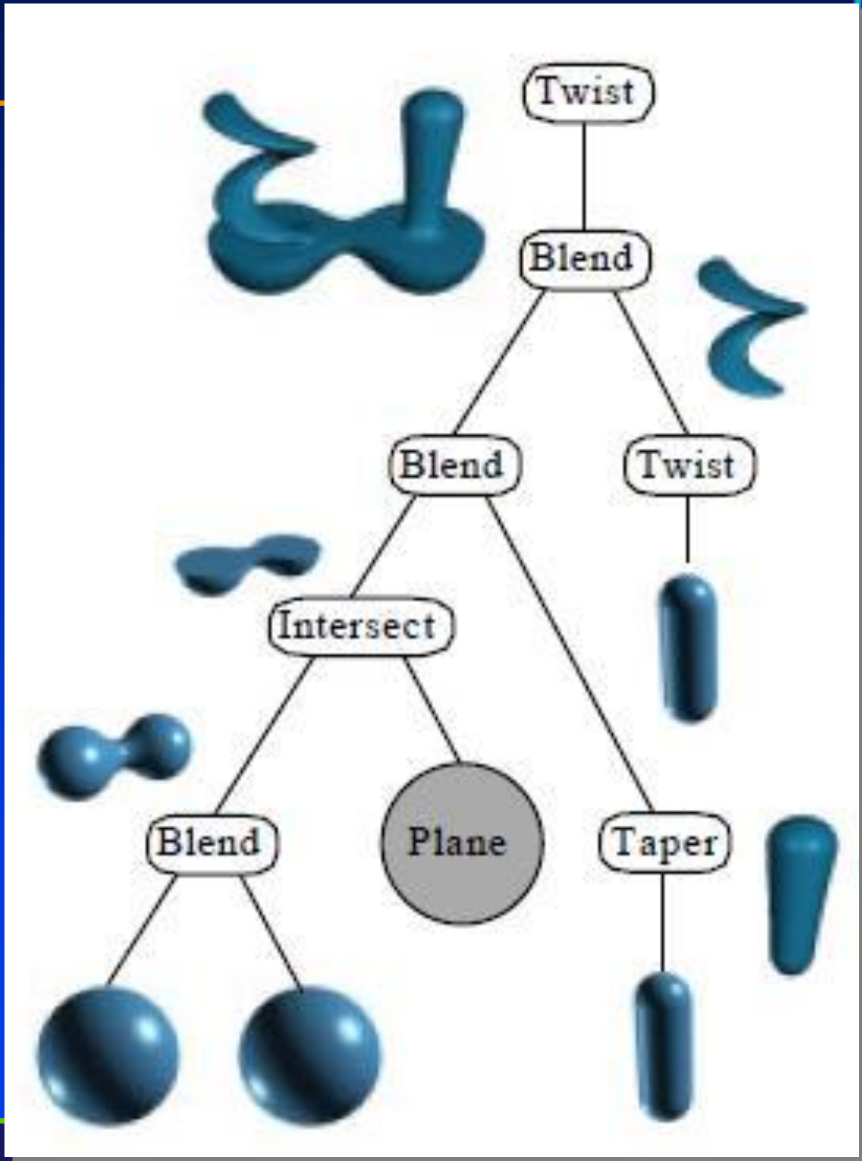
Approach:

BlobTree

(Hierarchical implicit volume Models)

BlobTree

- Leaves:
  Implicit primitives
- Tree nodes: Composition operators
- Complex 3D modeling with skeletal primitives



Why is BlobTree effective?

- Non-linear editing of primitives

→ Complex models can be constructed **easily**

- A hierarchical spatial cashing

→ Complex models can be constructed **Interactively**

Basic Functionalities

- Creating an implicit field from 2D contours defined by sketched strokes
- Converting 2D contours into 3D implicit volumes
- Editing 3D implicit volumes in BlobTree

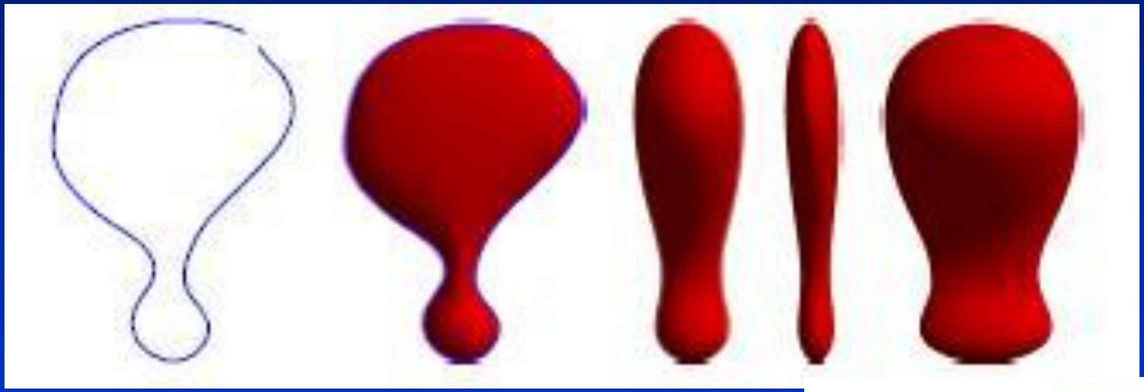
A Sketch-Based Implicit Field

$$g_{wyvill}(x) = (1 - x^2)^3$$

- C^2 Continuity
- $f_M = v_{iso}$ on a 2D stroke

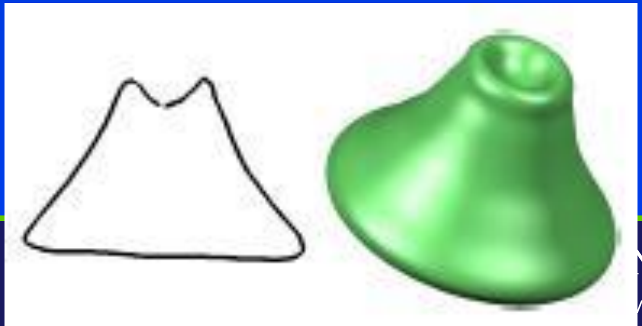
Three Types of Surfaces

- Blobby inflation



- Linear sweeps

- Surfaces of revolution



Operations

- Cutting (CSG)

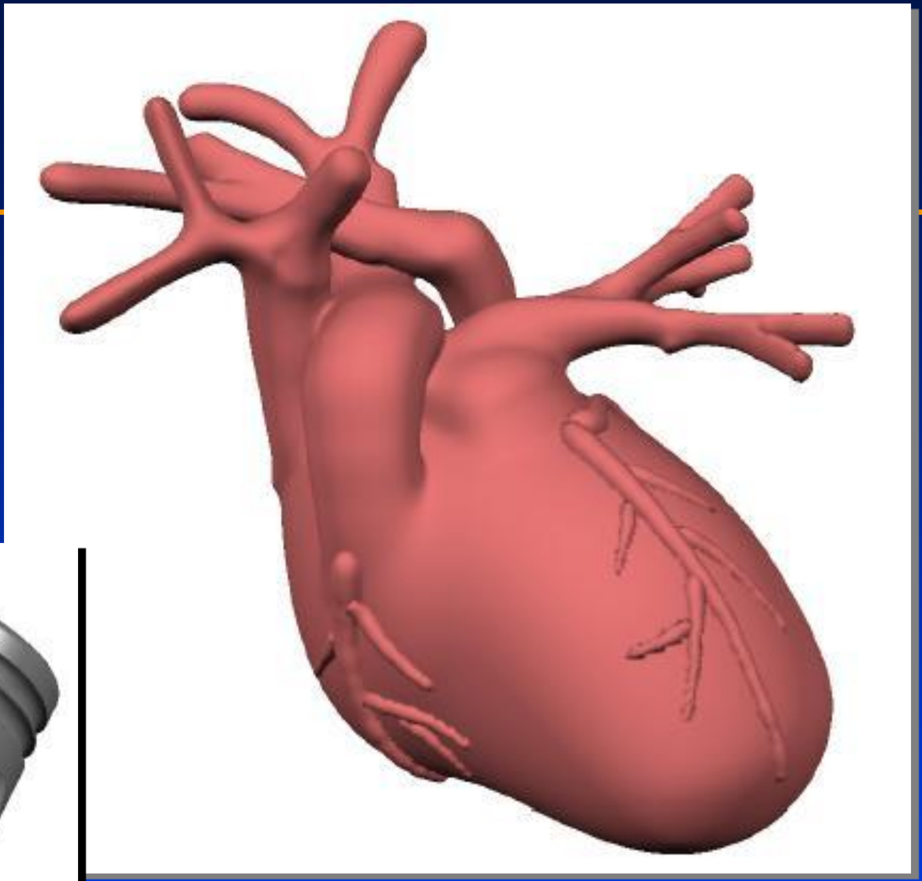
- Blending

BlobTree

- BlobTree has allowed us to create complex 3D models in a sketch-based modeling system

△ The User must understand BlobTree structure

Results



Results