Solid Modeling

Dr. Scott Schaefer

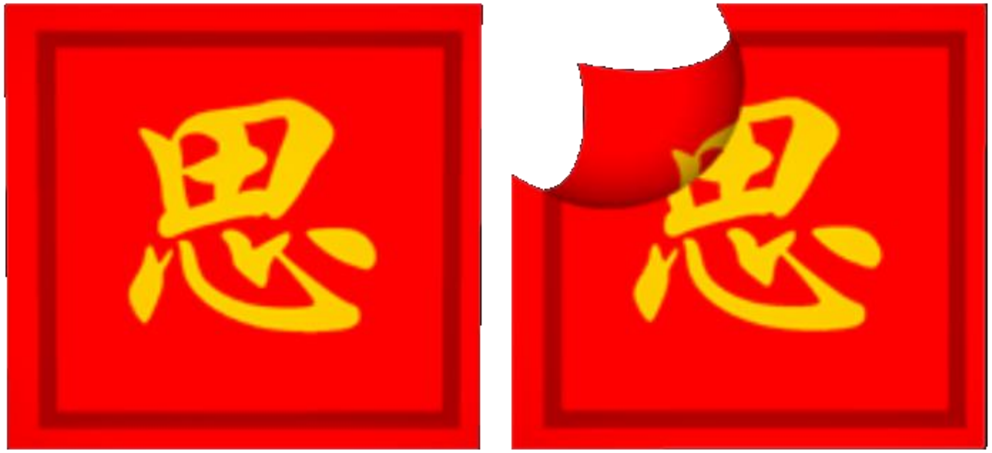
TAMU Computer Science

Solid Modeling Representations

- Constructive Solid Geometry
- Octrees
- Boundary Representations
- Implicit Representations

Constructive Solid Geometry

- Combine simple primitives together using set operations
  - Union, subtraction, intersection
- Intuitive operations for building more complex shapes



Constructive Solid Geometry

- Combine simple primitives together using set operations
  - Union, subtraction, intersection
- Intuitive operations for building more complex shapes

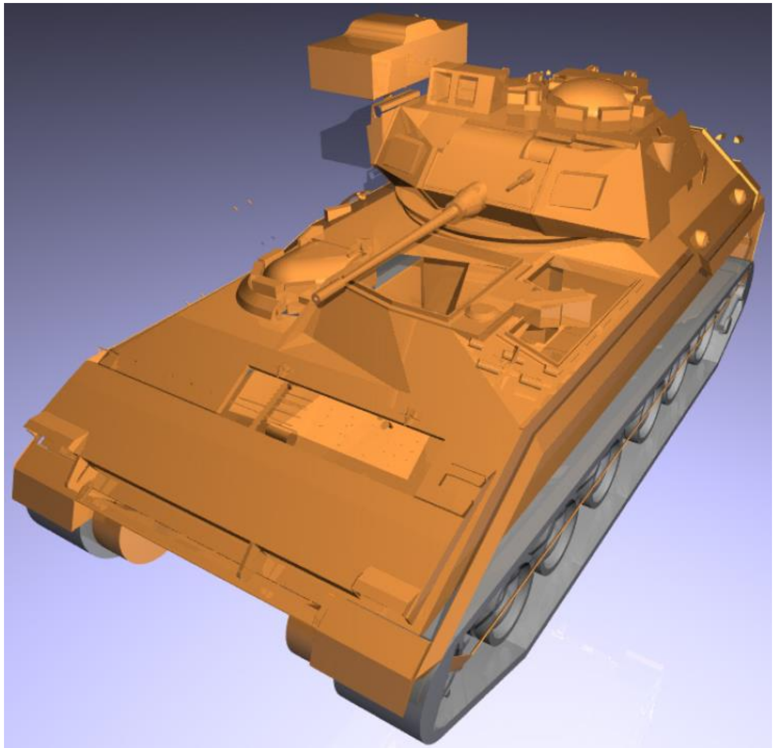
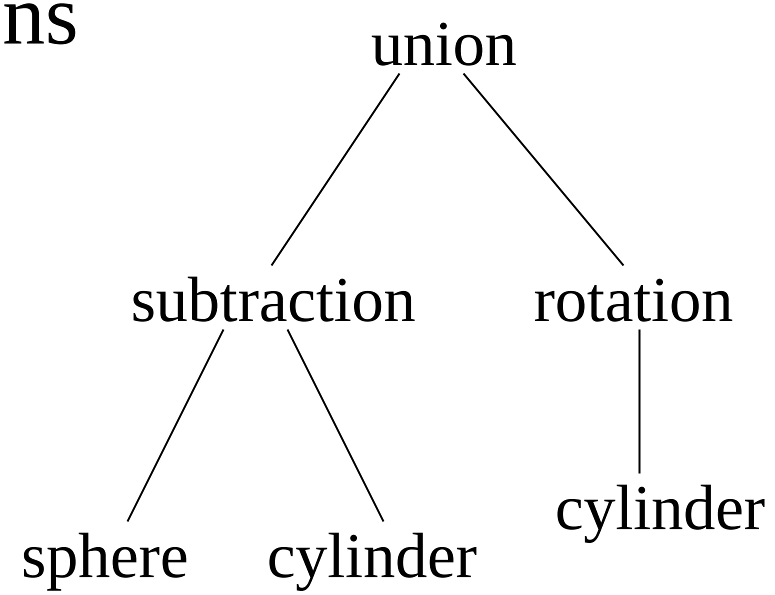


Image taken from "Feature-Sensitive Subdivision and Isosurface Reconstruction"

Constructive Solid Geometry

- Typically represented as binary tree
- Leaves store solids (sphere, cylinder, …)
- Interior nodes are operations (union, subtraction, …) or transformations



Ray Tracing CSG Trees

- Assume we have a ray R and a CSG tree T

- If T is a solid,
  - compute all intersections of R with T
  - return parameter values and normals
- If T is a transformation
  - apply inverse transformation to R and recur
  - apply inverse transpose of transformation to normals
  - return parameter values
- Otherwise T is a boolean operation
  - recur on two children to obtain two sets of intervals
  - apply operation in T to intervals
  - return parameter values.

- Display closest intersection point

Inside/Outside Test for CSG Trees

- Given a point p and a tree T, determine if p is inside/outside the solid defined by T

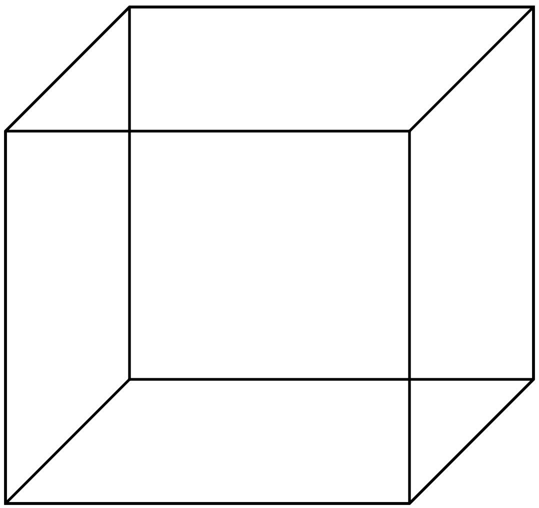
- If T is a solid
  - Determine if p is inside T and return
- If T is a transformation
  - Apply the inverse transformation to p and recur
- Otherwise T is a boolean operation
  - Recur to determine inside/outside of left/right children
  - If T is Union
    - If either child is inside, return inside, else outside
  - If T is Intersection
    - If both children are inside, return inside, else outside
  - If T is Subtraction
    - If p is inside left child and outside right child, return inside, else outside

Application: Computing Volume

- Monte Carlo method
- Put bounding box around object
- Pick n random points inside the box
  - Determine if each point is inside/outside the CSG Tree
- Volume $\approx vol(box)\,{}^{\#inside}\!/_{n}$

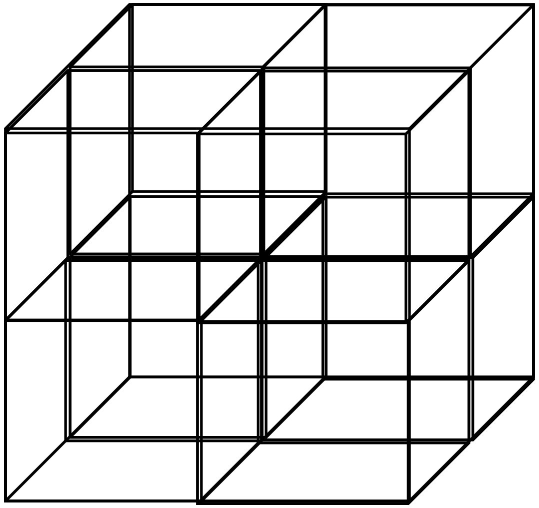
Octrees

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



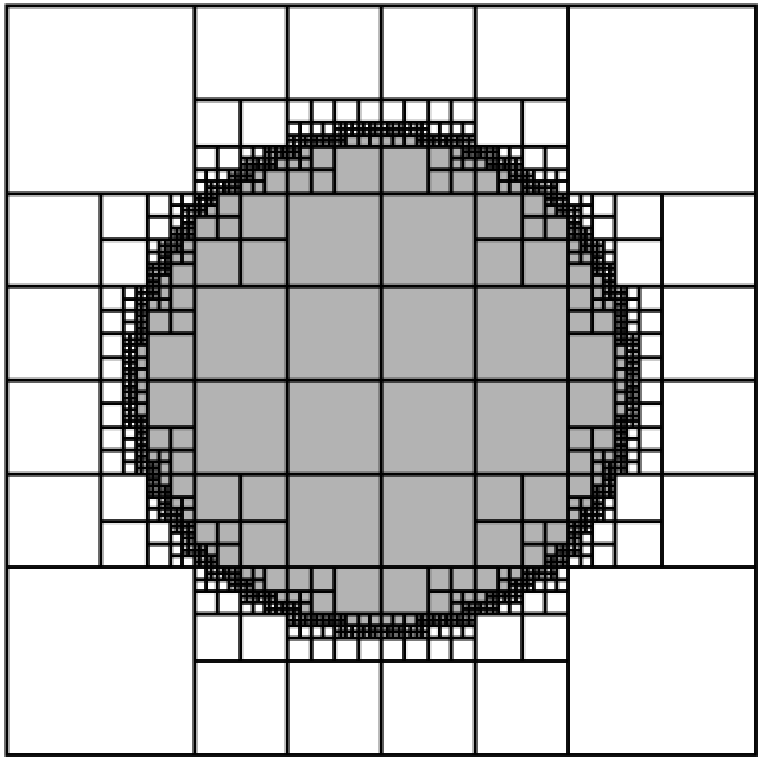
Octrees

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



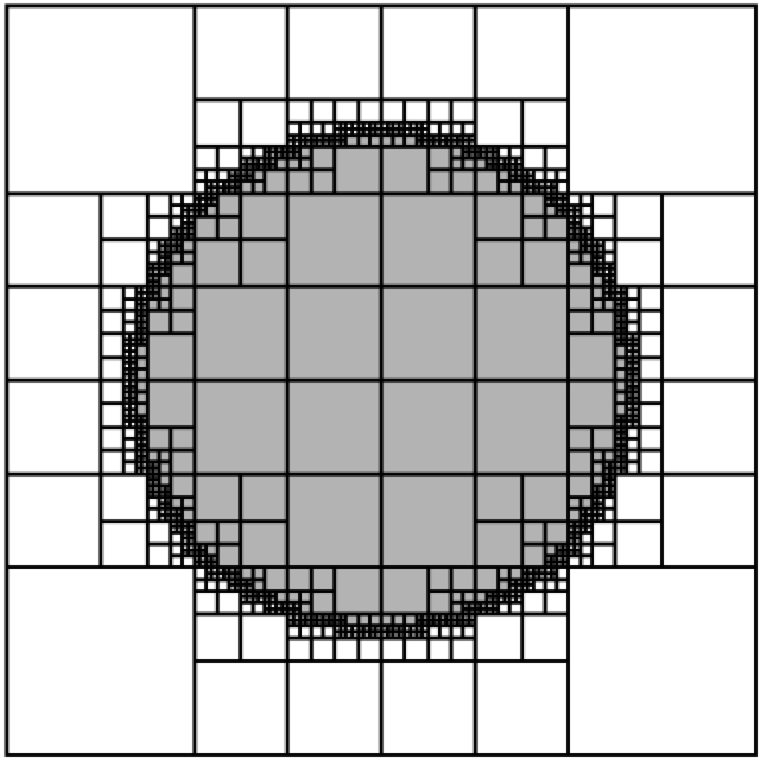
Building Octrees

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



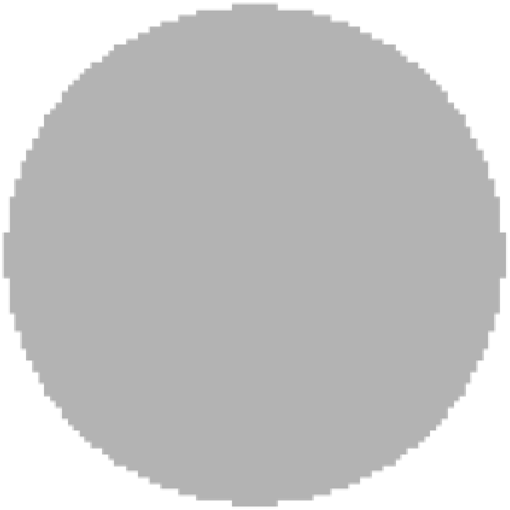
Octrees

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



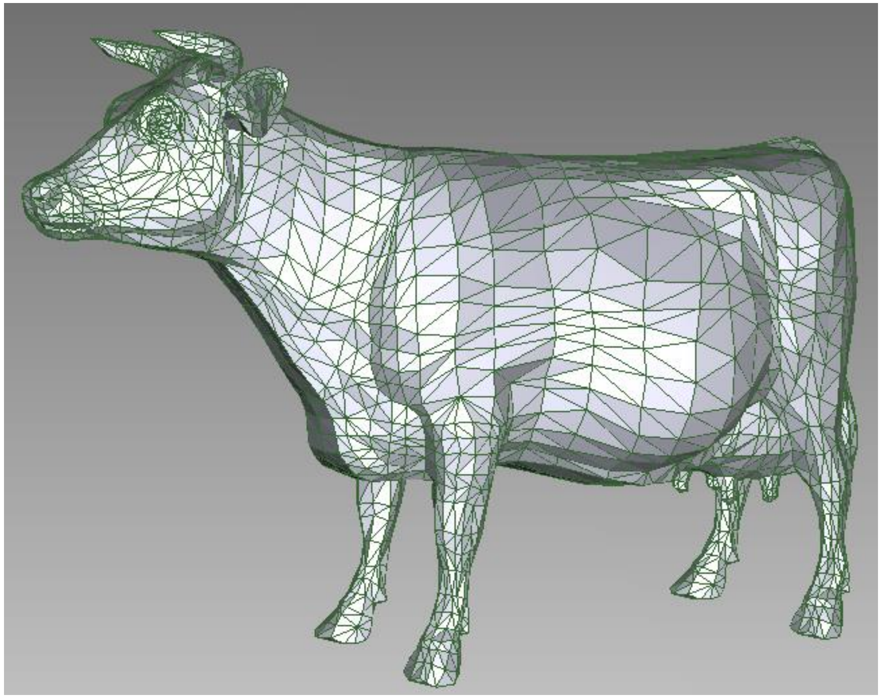
Octrees

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



Boundary Representations

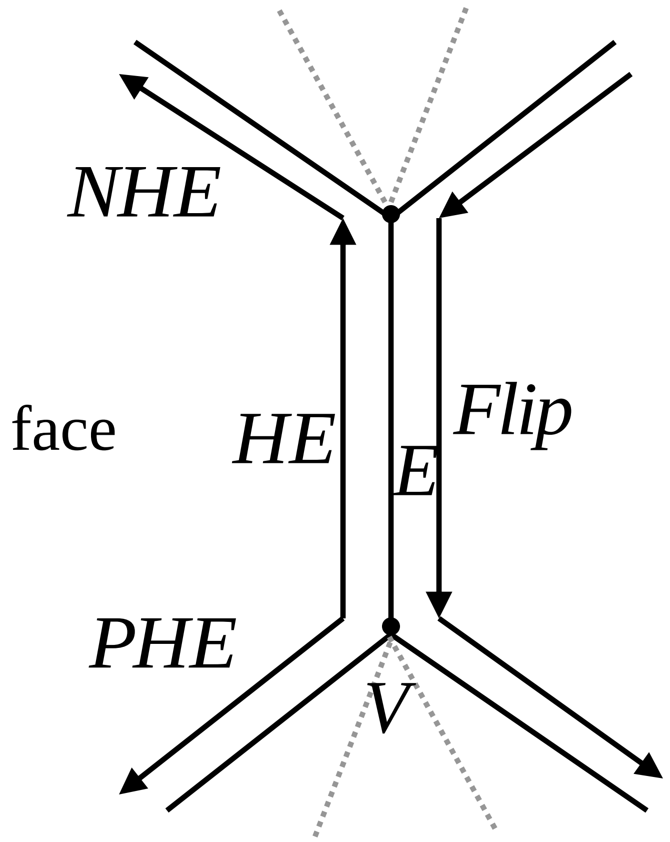
- Stores the boundary of a solid
  - Geometry: vertex locations
  - Topology: connectivity information
    - Vertices
    - Edges
    - Faces



Boundary Representations

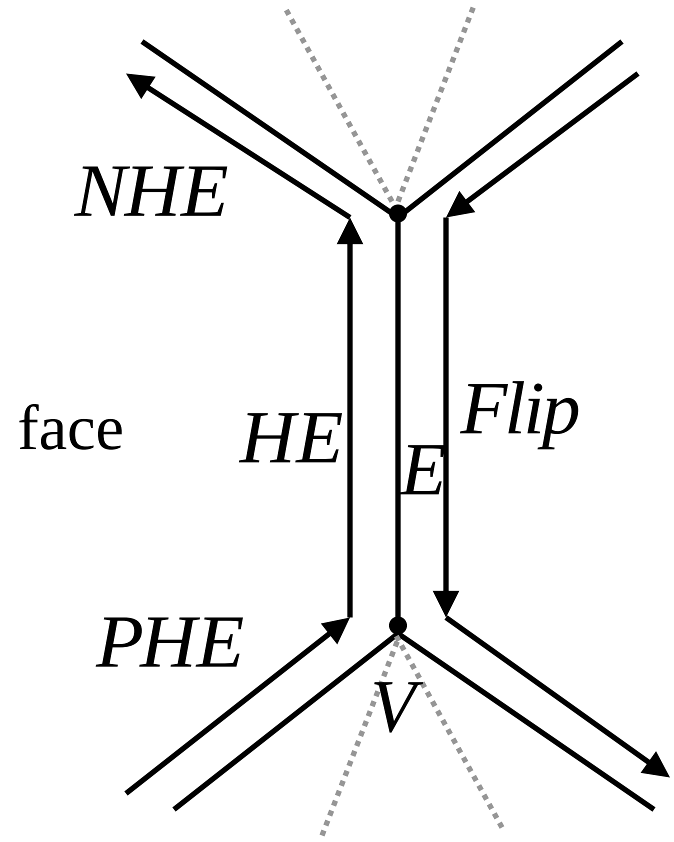
- Constant time adjacency information
  - For each vertex,
    - Find edges/faces touching vertex
  - For each edge,
    - Find vertices/faces touching edge
  - For each face,
    - Find vertices/edges touching face

Half Edge Data Structure



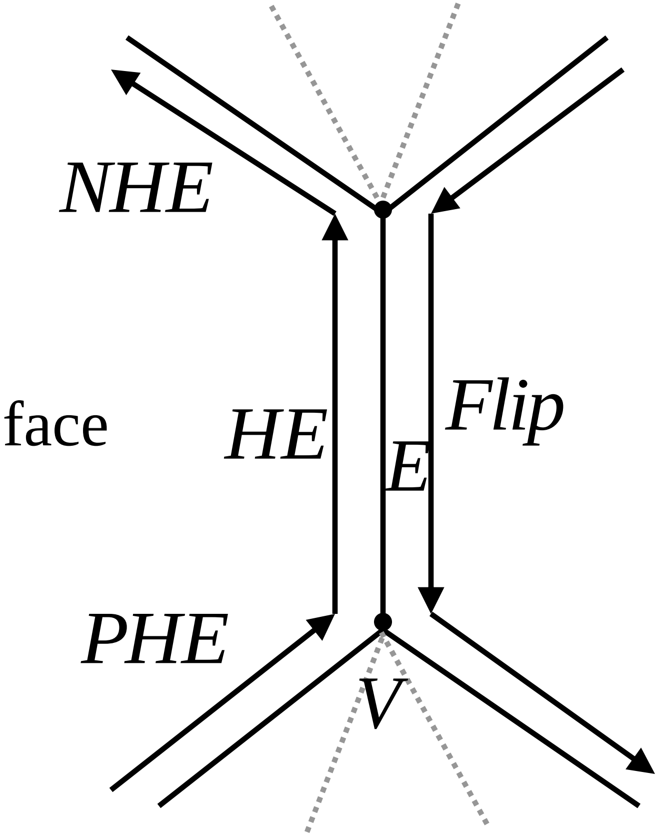
Half Edge Data Structure

```
HalfEdge {
    HalfEdge next, prev, flip;
    Face face;
    Vertex origin;
    Edge edge;
}
Face {
    HalfEdge edge; // part of this face
}
Vertex {
    HalfEdge edge;  // points away
}
Edge {
    HalfEdge he;
}
```



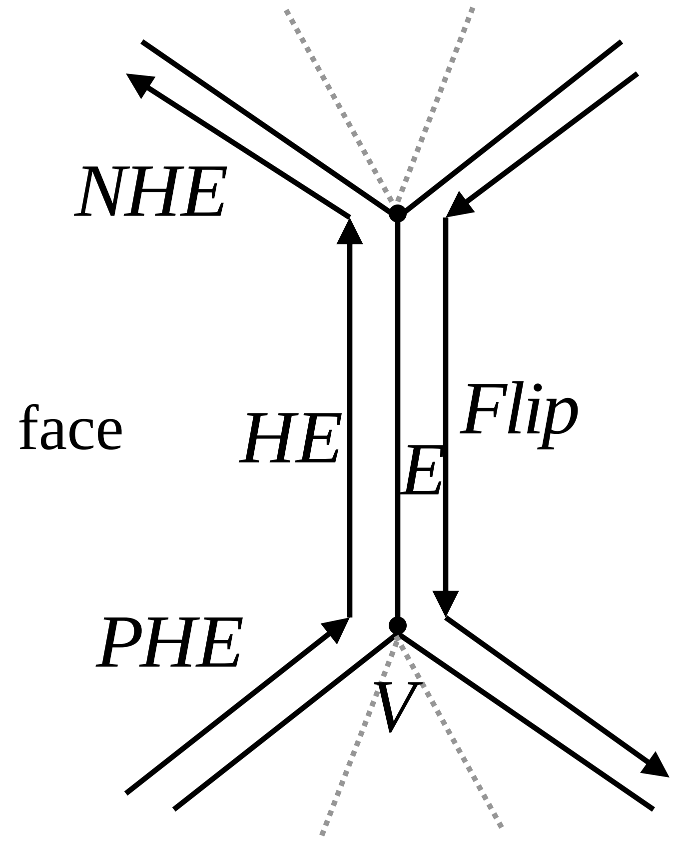
Half Edge Data Structure

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



Building a Topological Data Structure

- Must connect adjacent edges/faces/vertices
- Edges are critical in most data structures
- Use a hash table indexed by two vertices



Boundary Representations

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

Implicit Representations of Shape

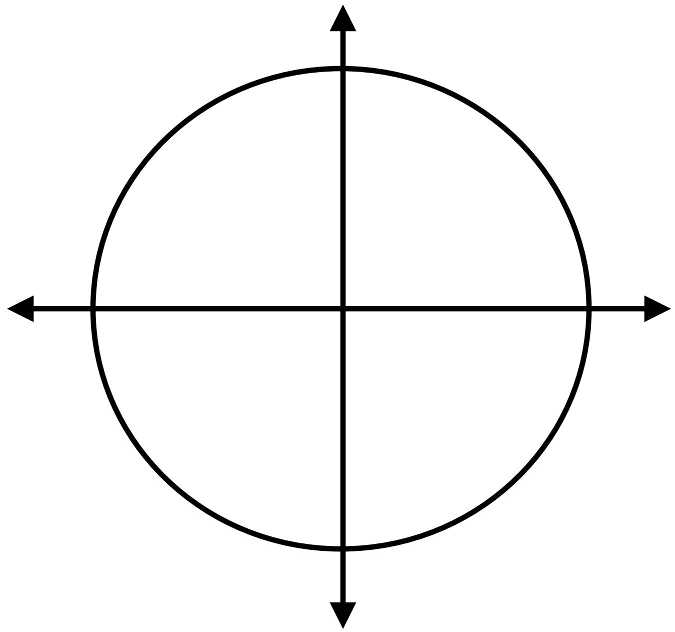
- Shape described by solution to $f(x)=c$

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

Implicit Representations of Shape

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

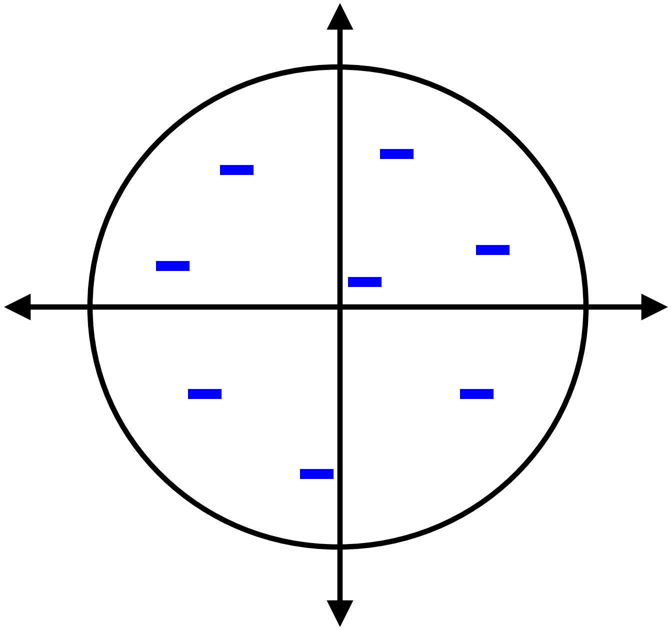
$$f(x, y) = x^2 + y^2 - 9$$



Implicit Representations of Shape

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

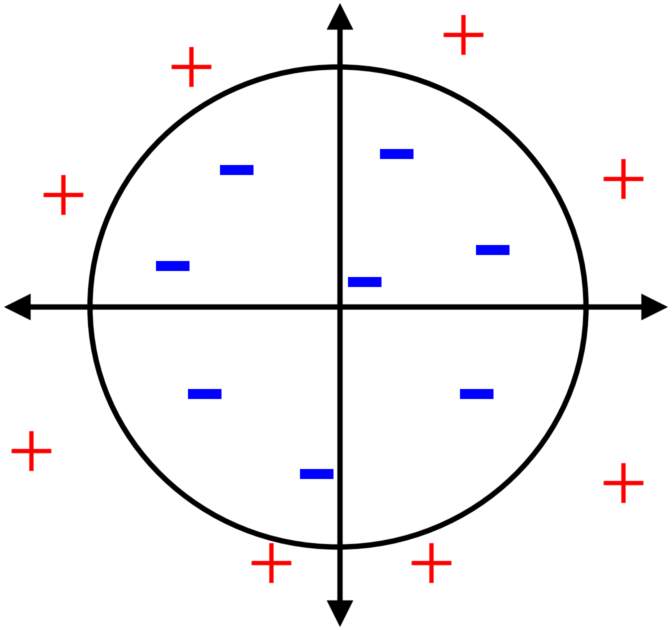
$$f(x,y) = x^2 + y^2 - 9$$



Implicit Representations of Shape

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

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

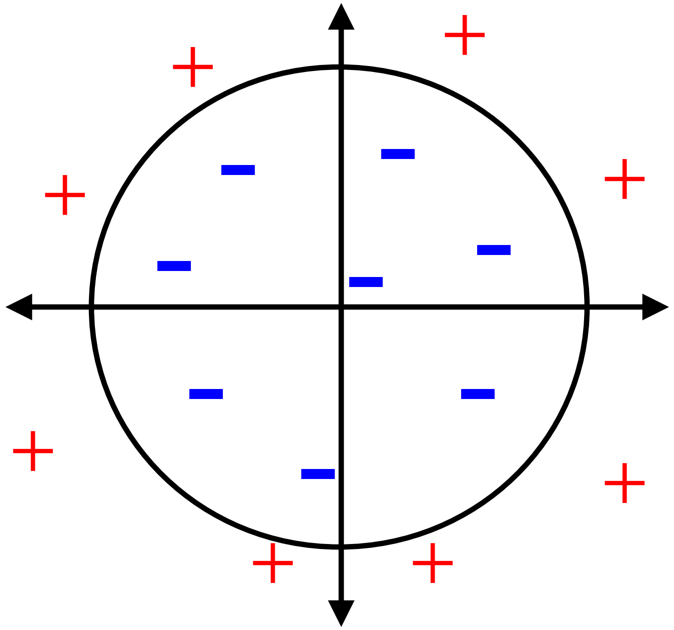


Advantages

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

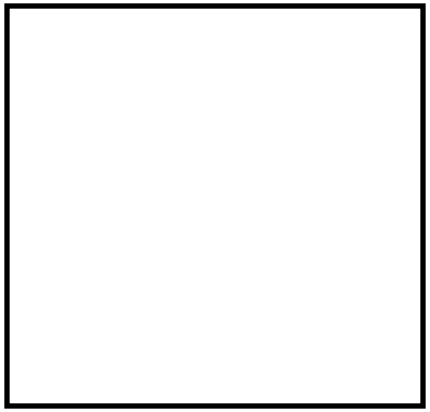
Advantages

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



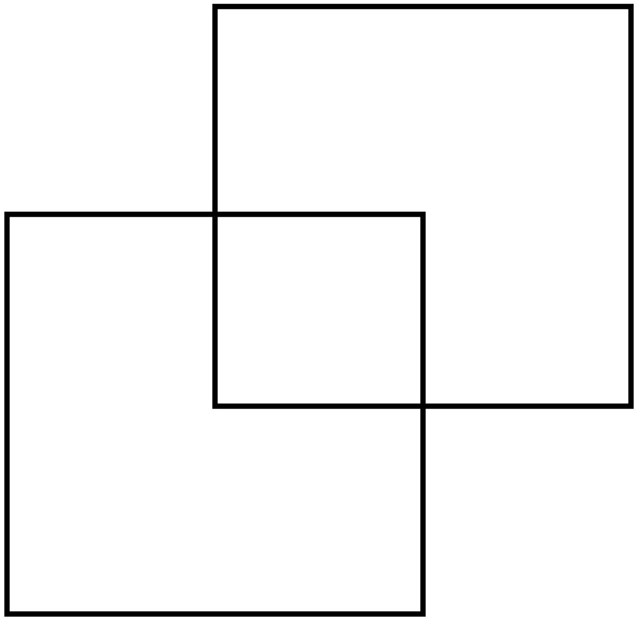
Advantages

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



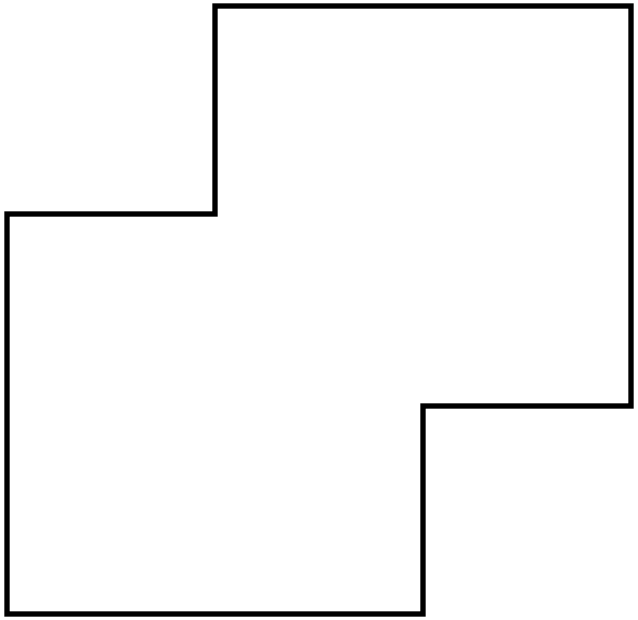
Advantages

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



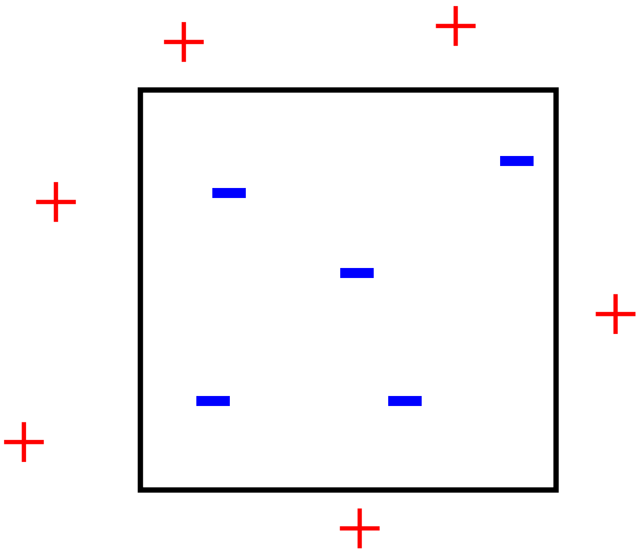
Advantages

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



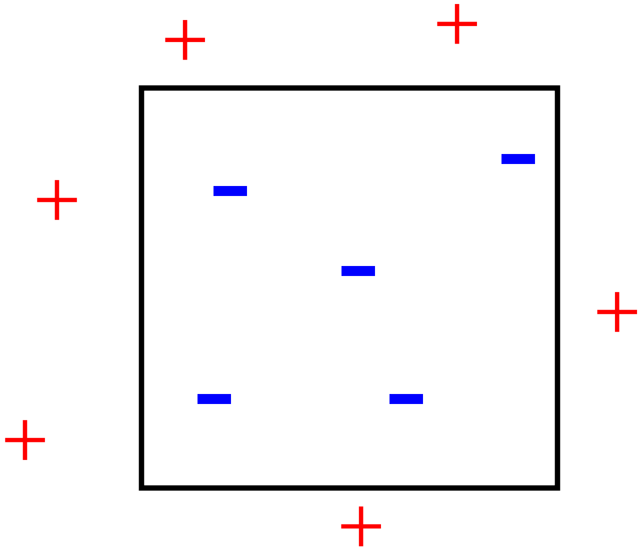
Advantages

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



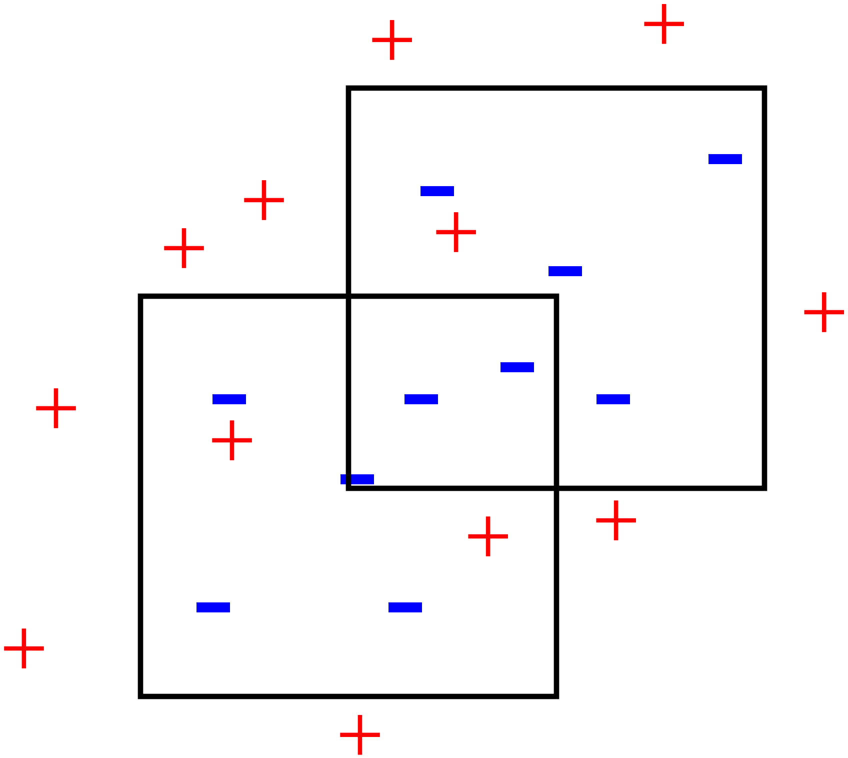
Advantages

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



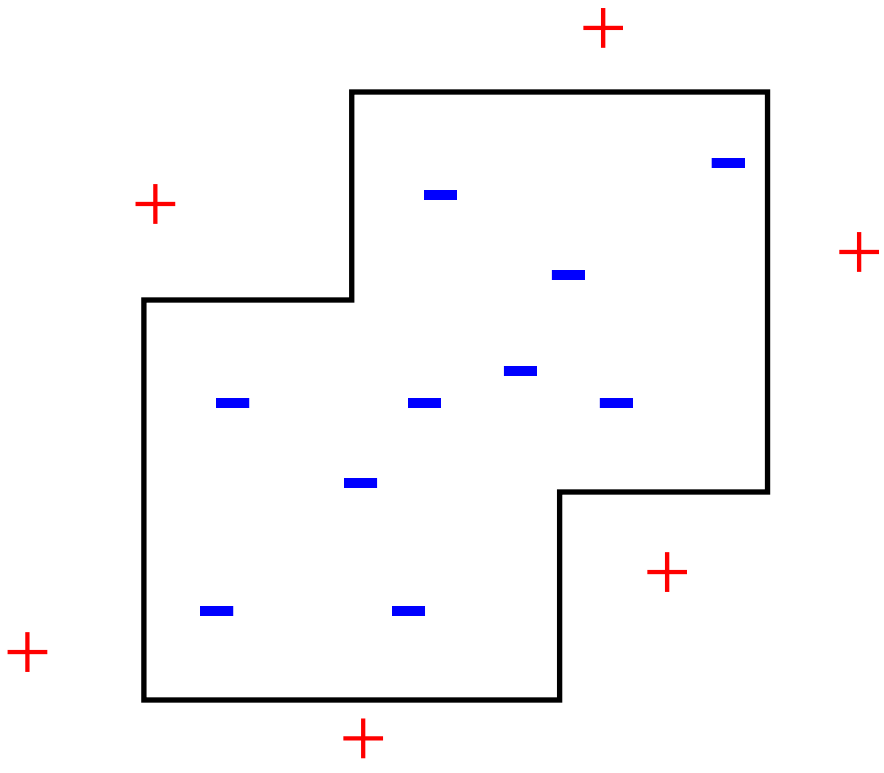
Advantages

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



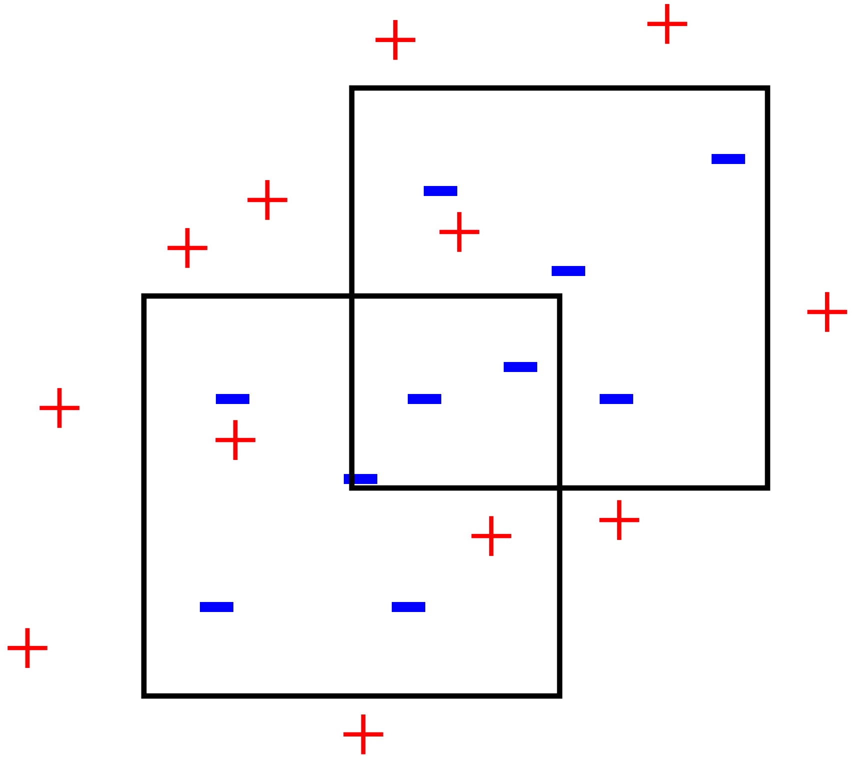
Advantages

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



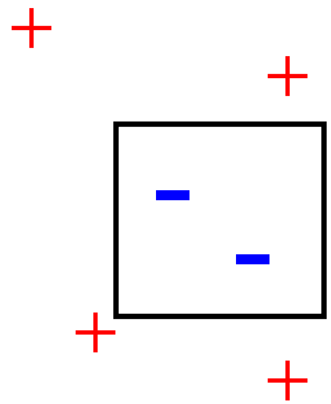
Advantages

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



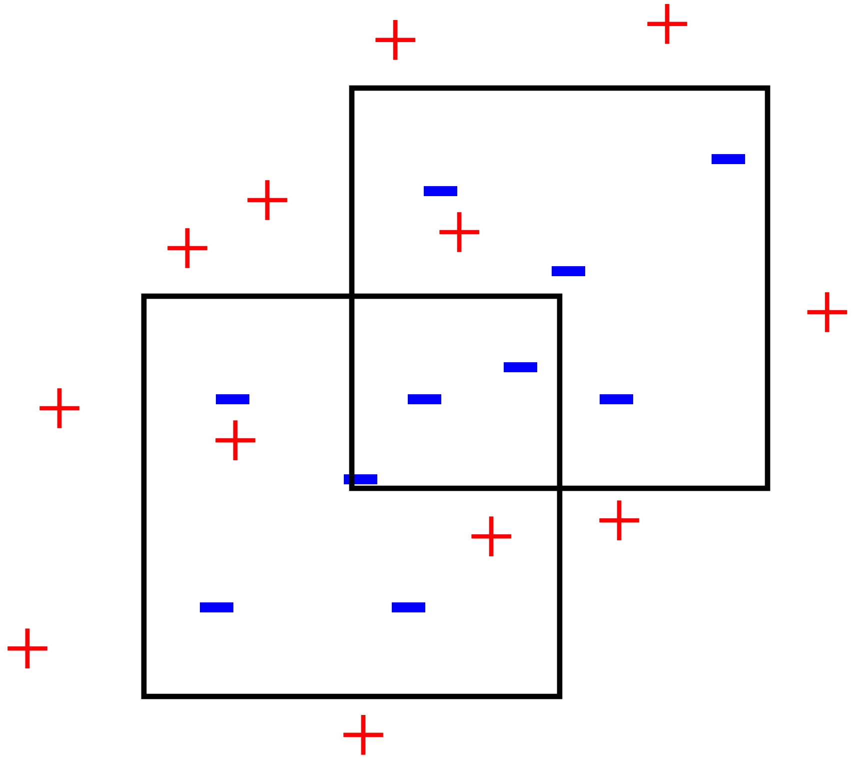
Advantages

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



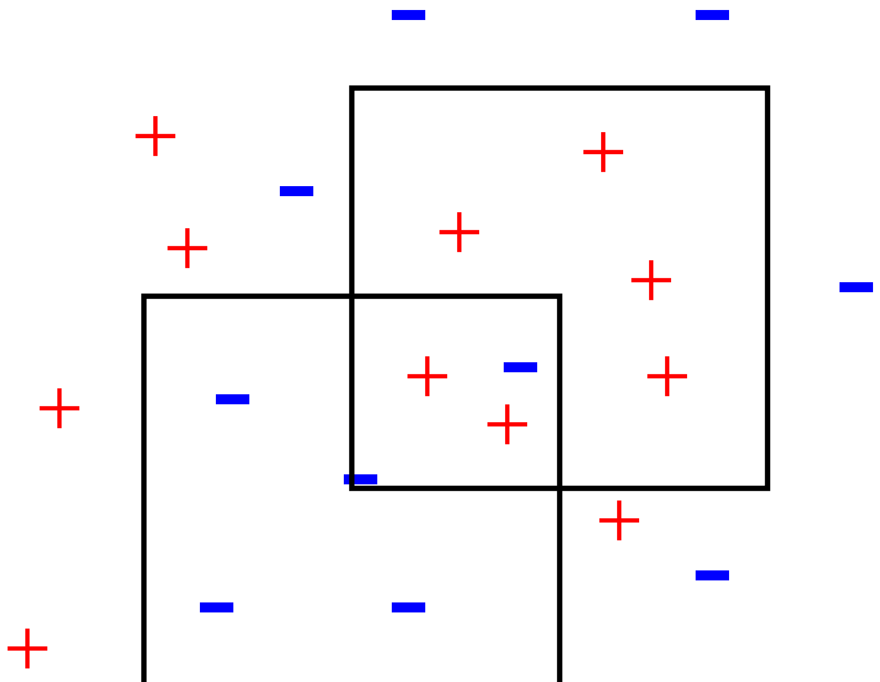
Advantages

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



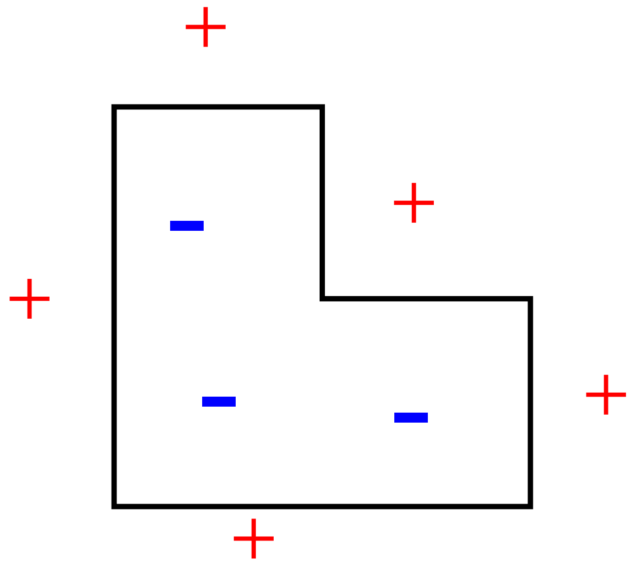
Advantages

- 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

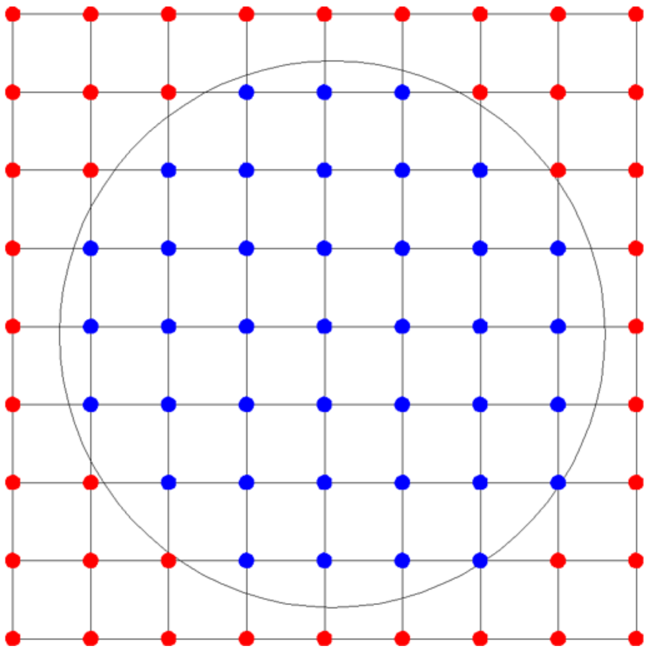


Disadvantages

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

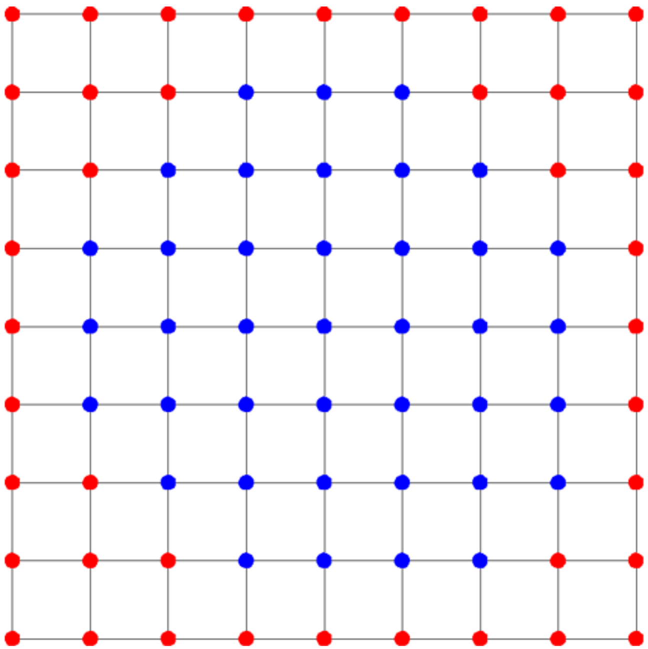
Non-Analytic Implicit Functions

- Sample functions over grids

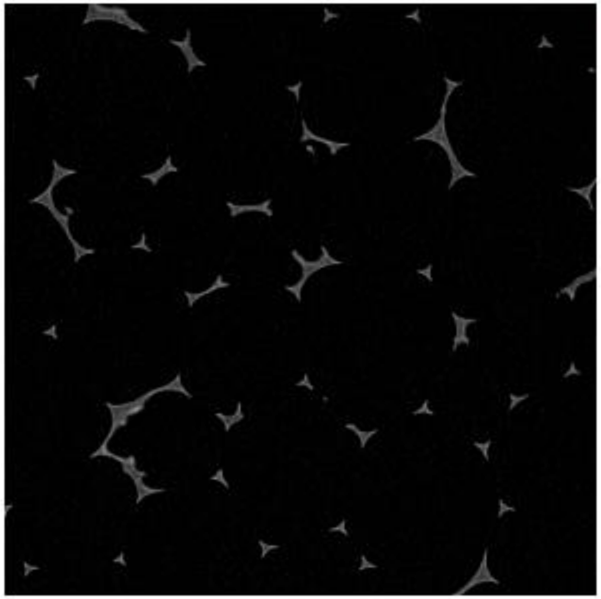


Non-Analytic Implicit Functions

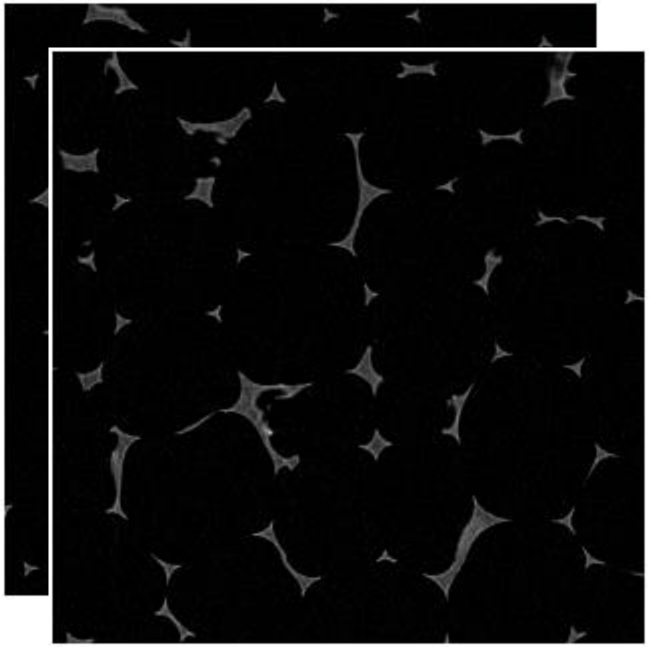
- Sample functions over grids



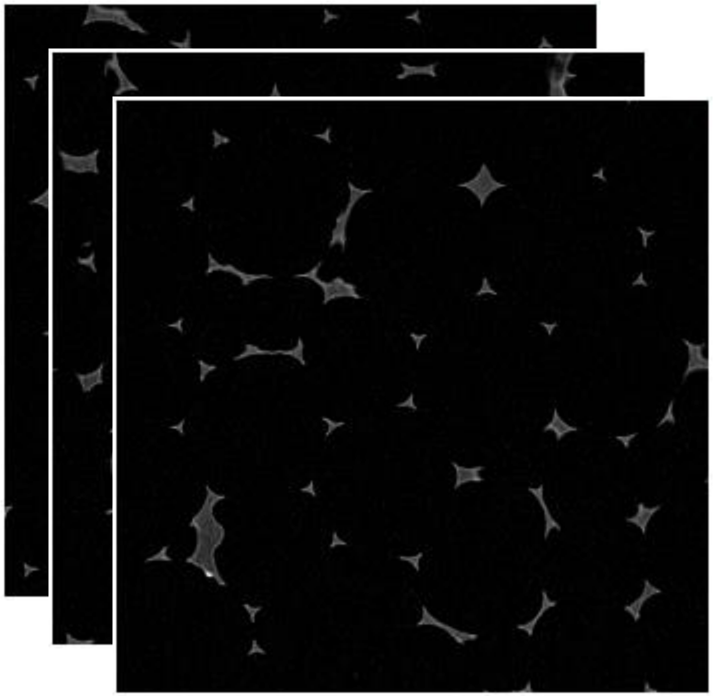
Data Sources



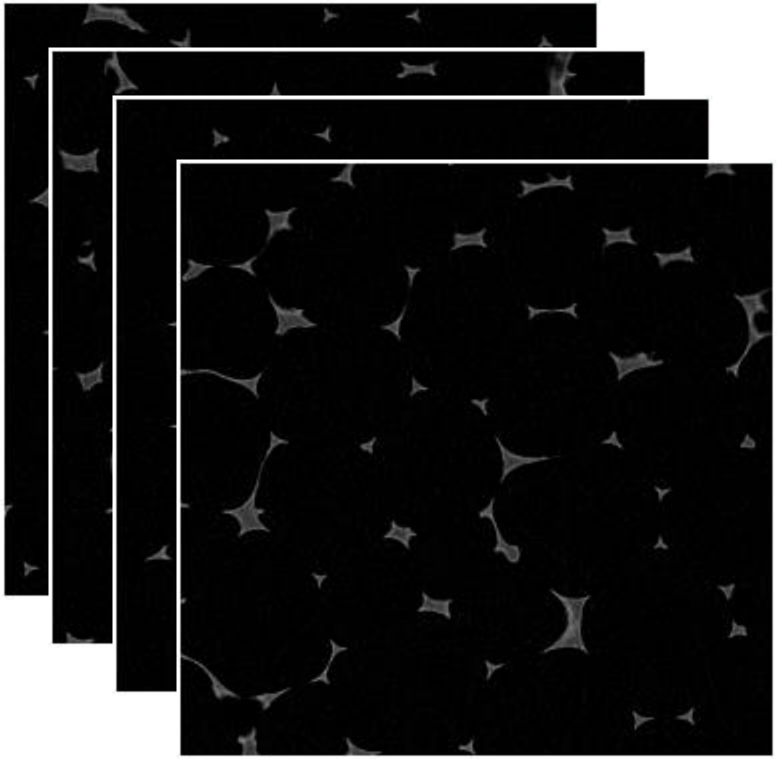
Data Sources



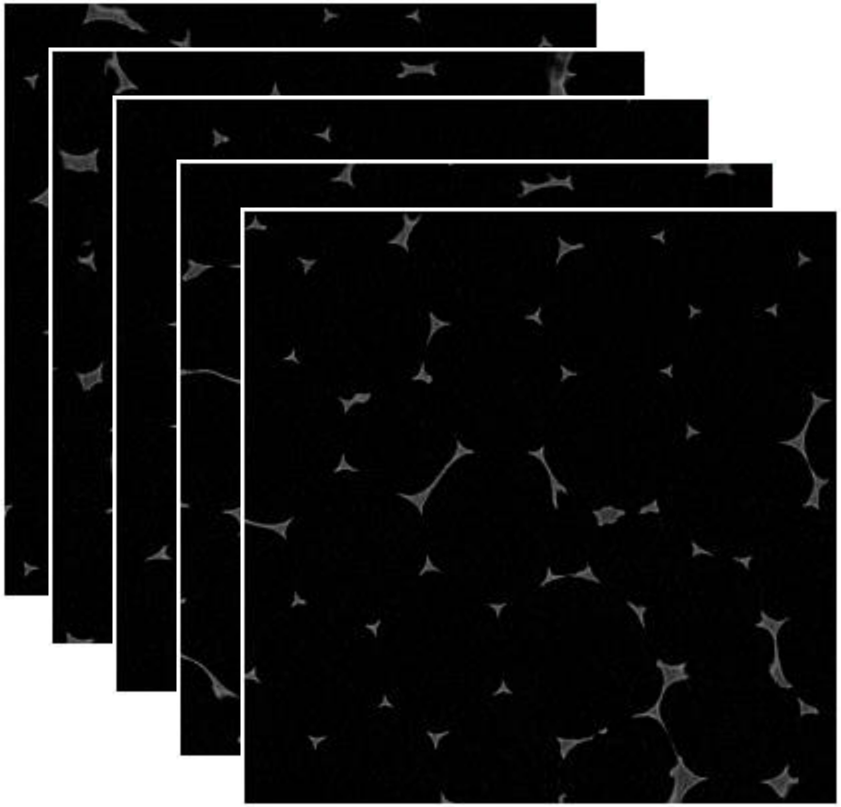
Data Sources



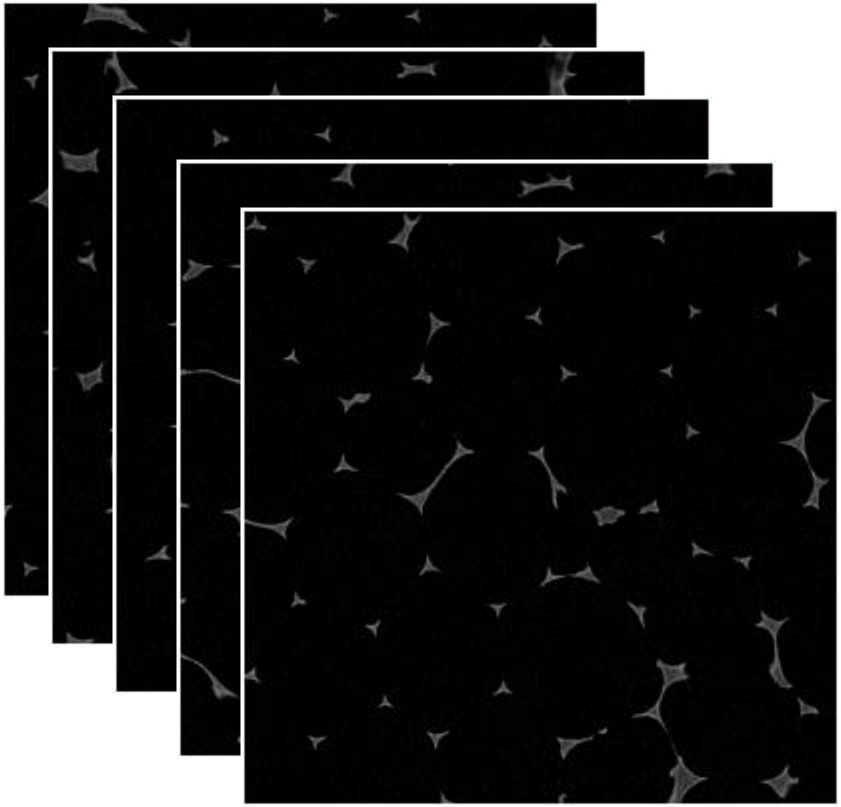
Data Sources

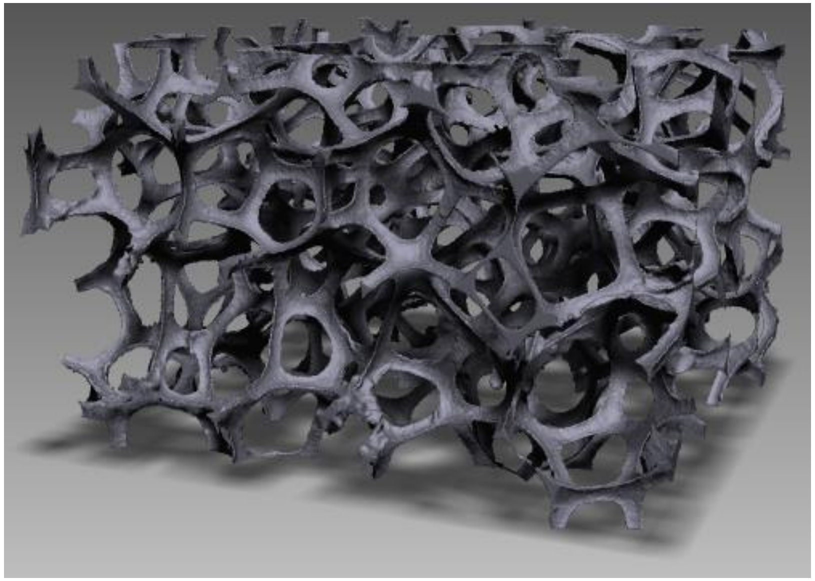


Data Sources

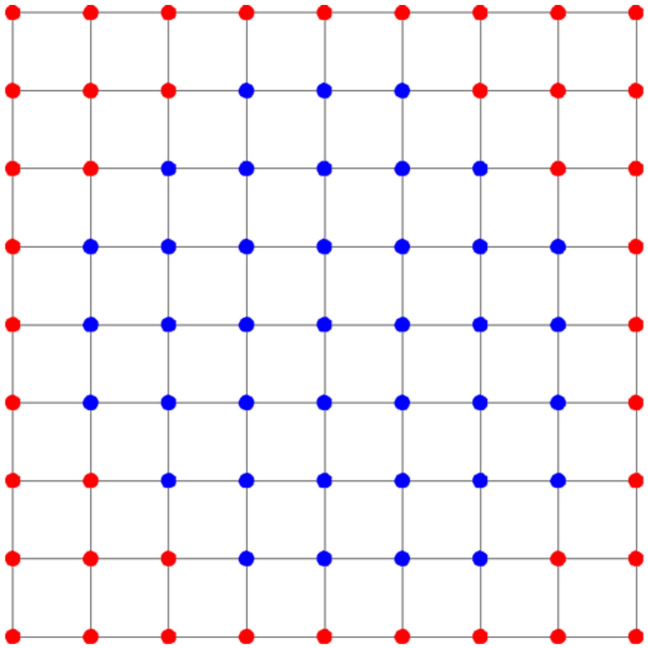


Data Sources

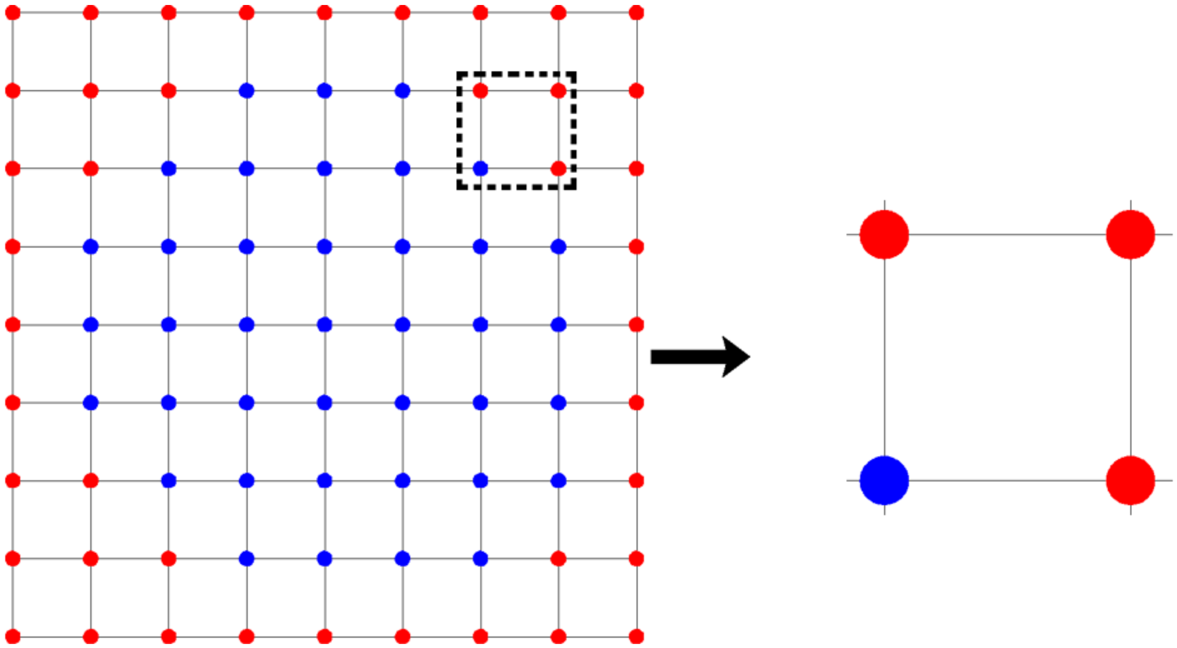




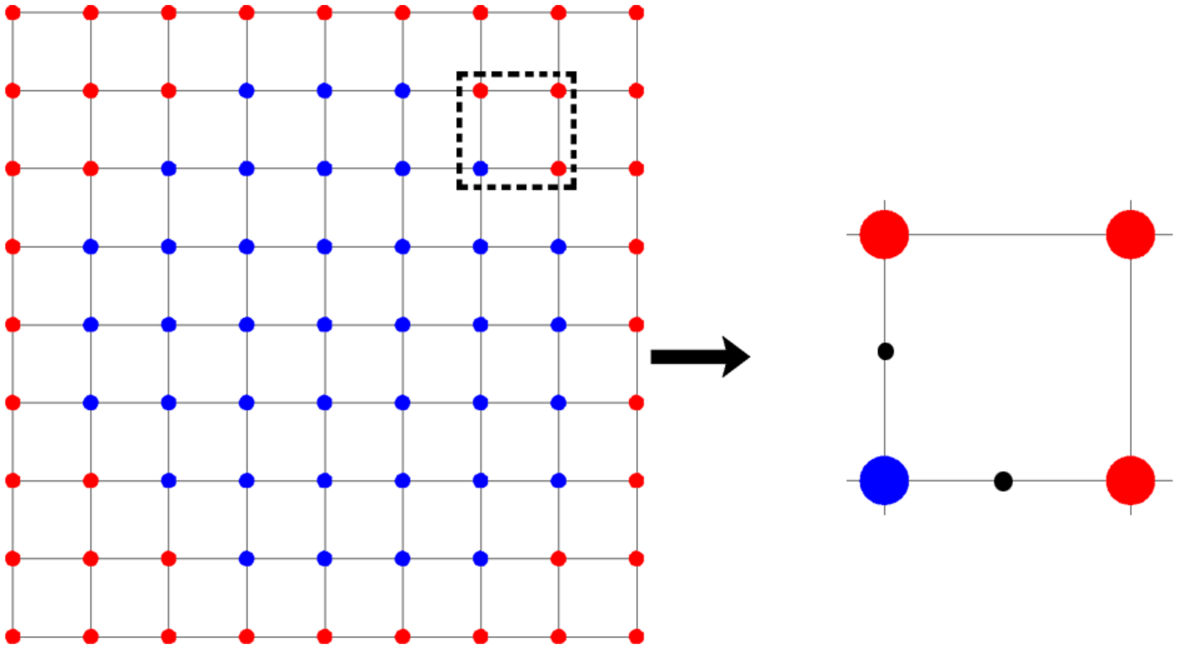
2D Polygon Generation



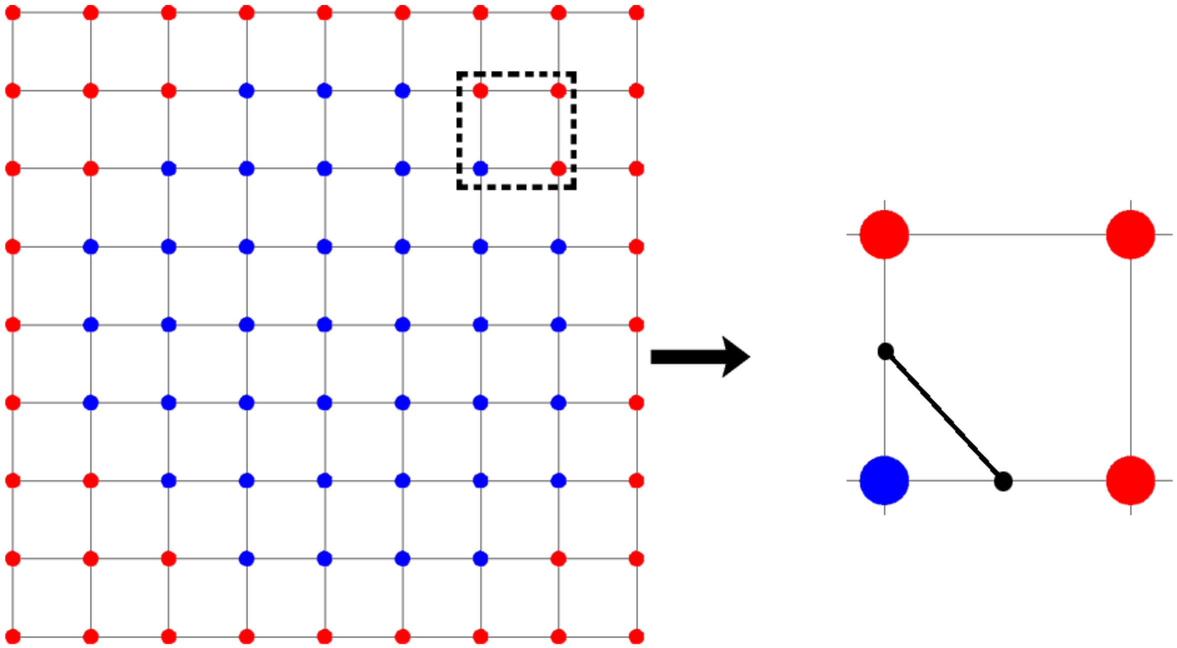
2D Polygon Generation



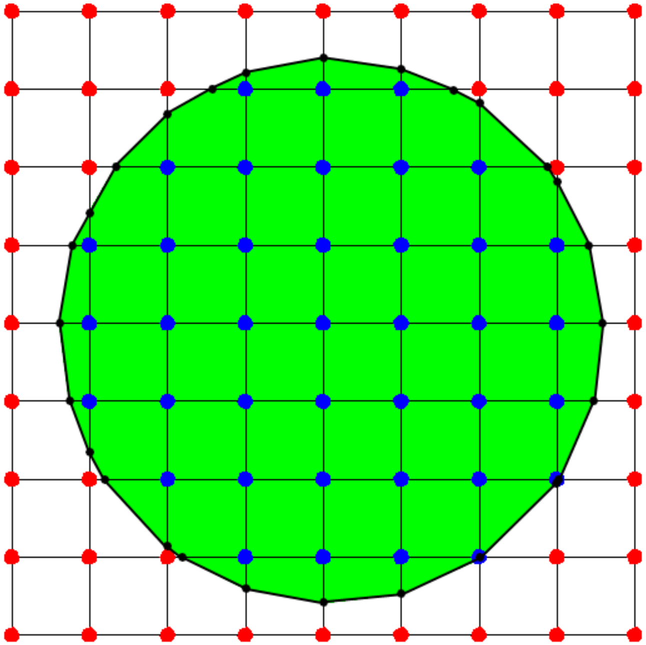
2D Polygon Generation



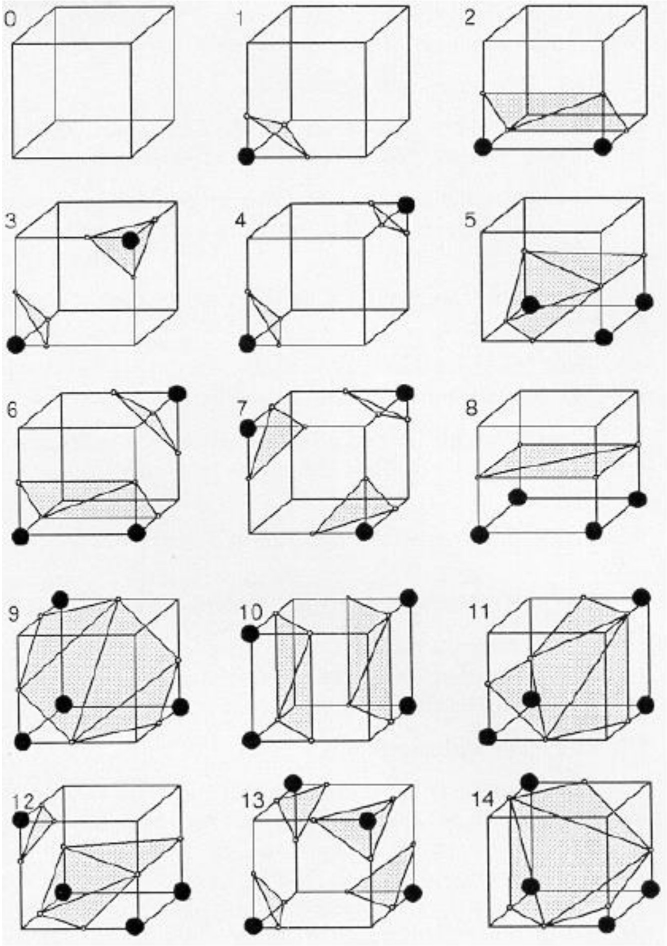
2D Polygon Generation

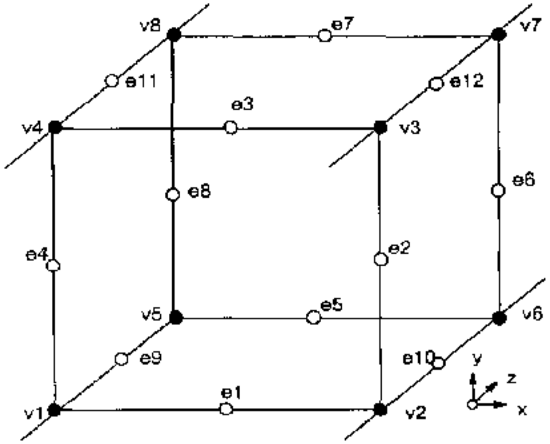


2D Polygon Generation

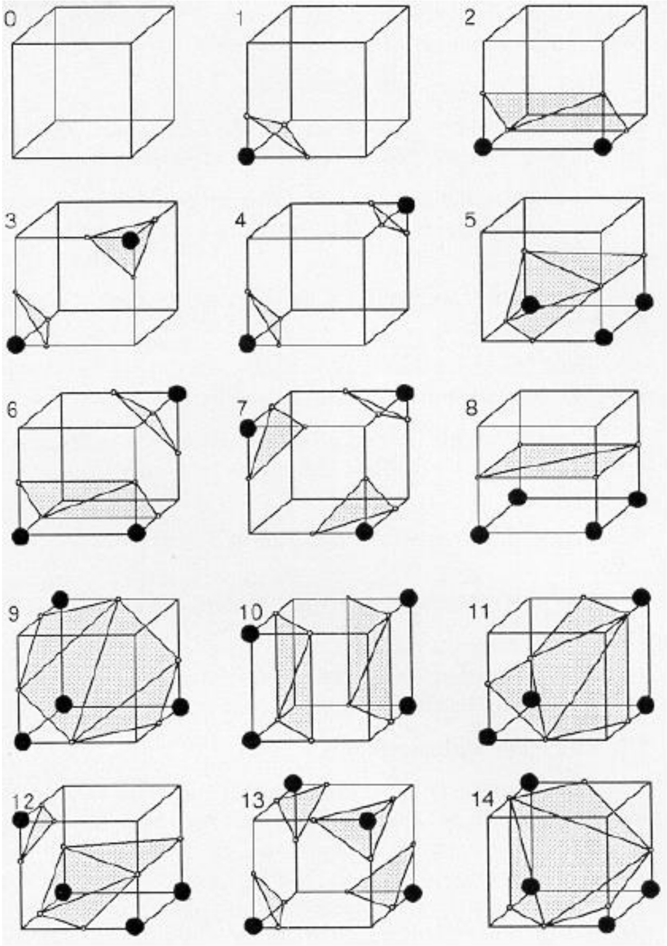


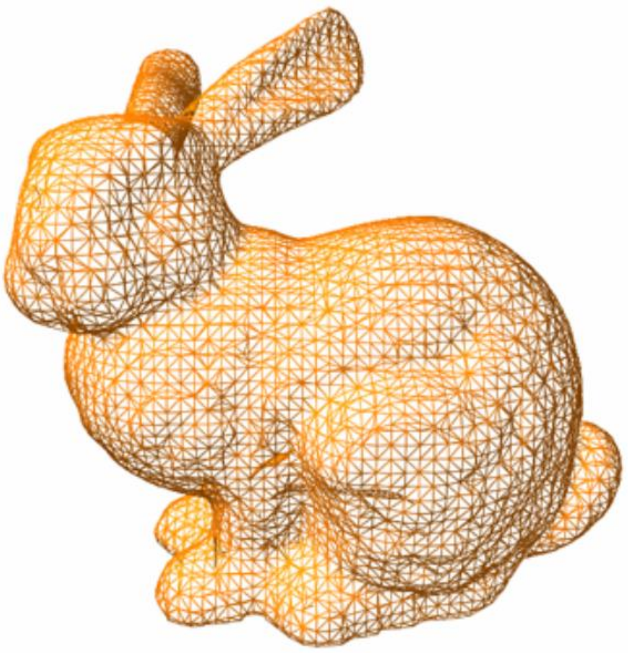
3D Polygon Generation





3D Polygon Generation





Fun Examples

Fun Examples

Fun Examples

Fun Examples