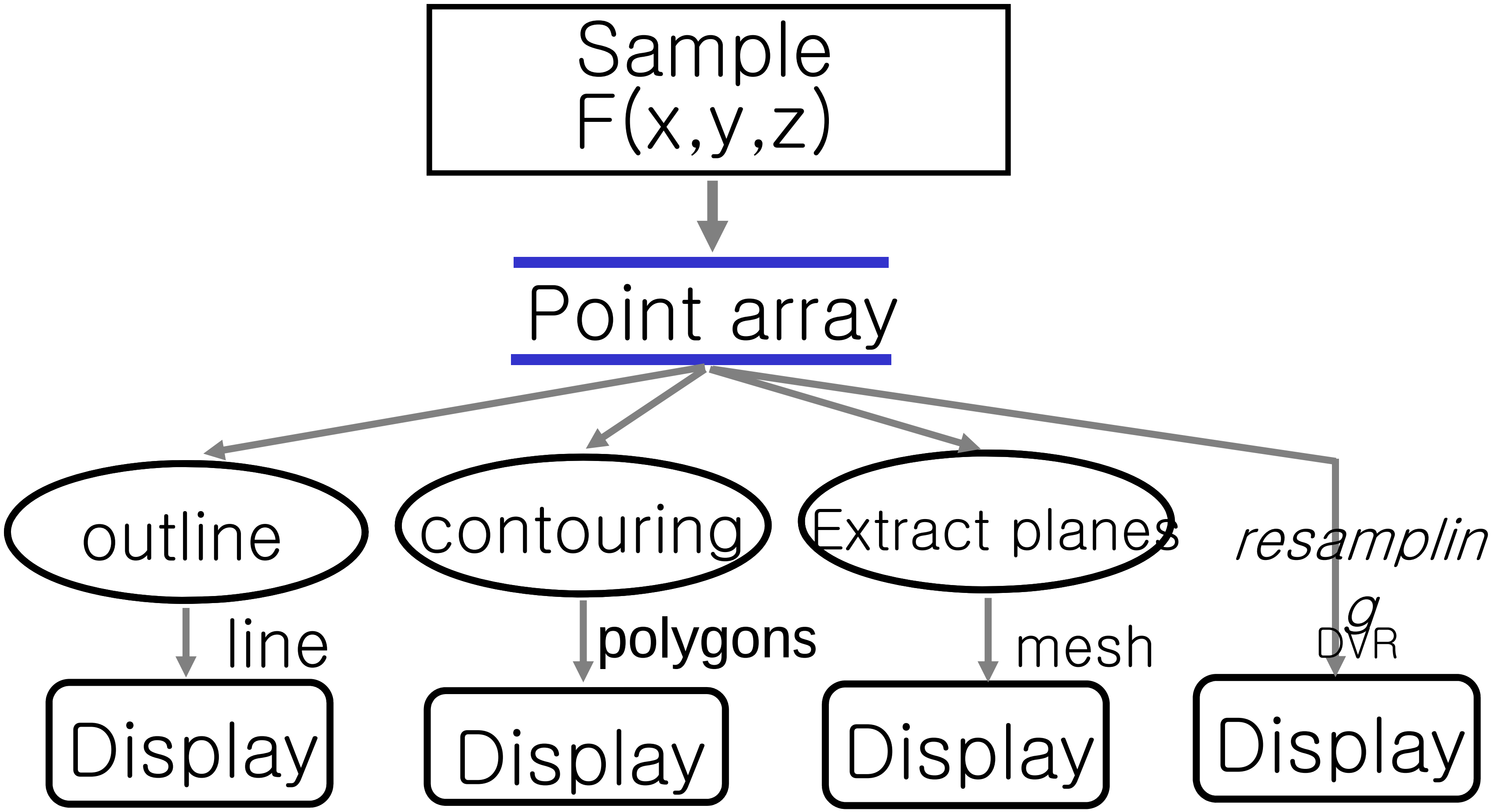
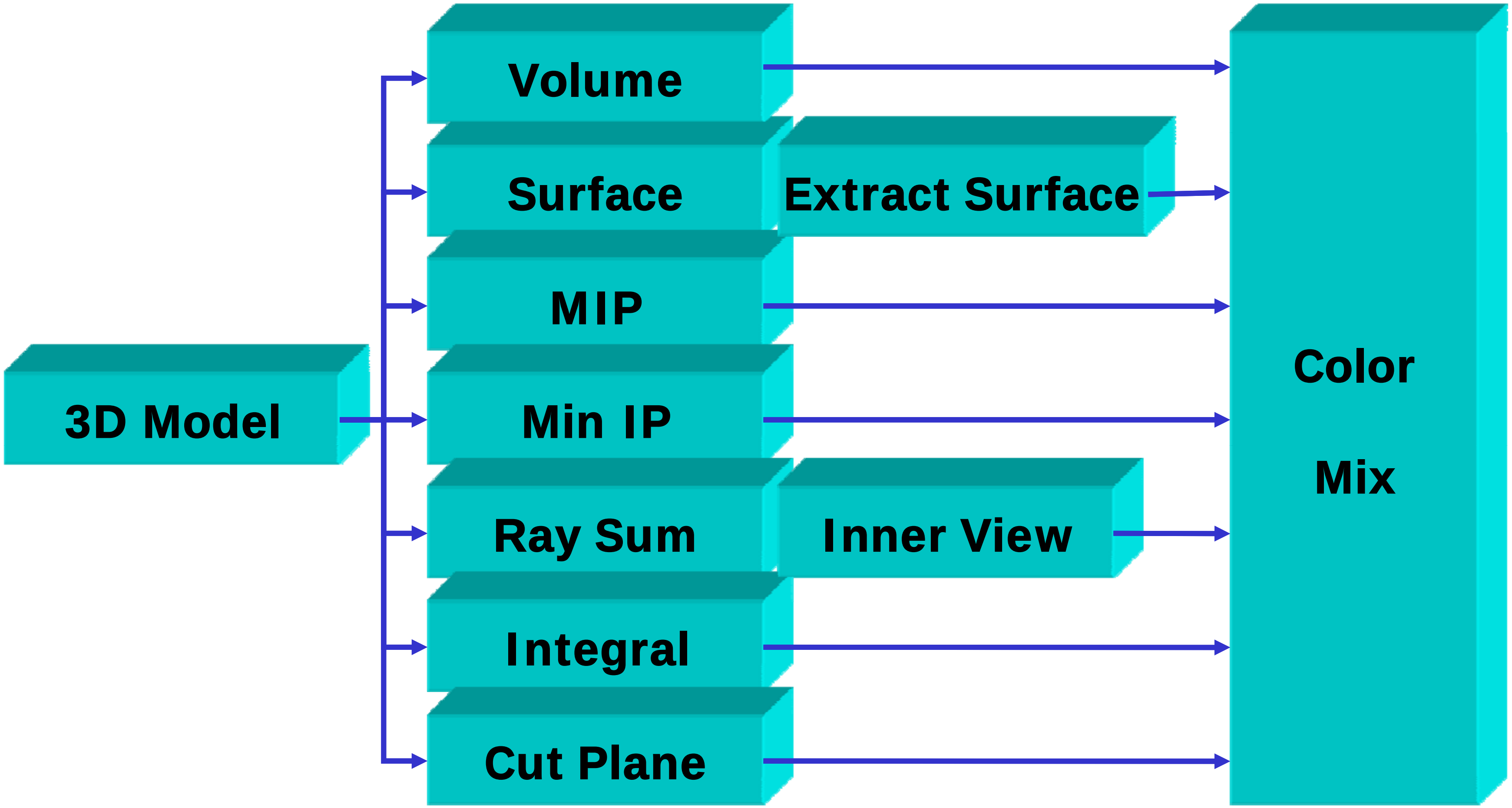
Visualization Methods

- Contour Tracking
- Surface Rendering
- Direct Volume Rendering
- Iso-surface Rendering

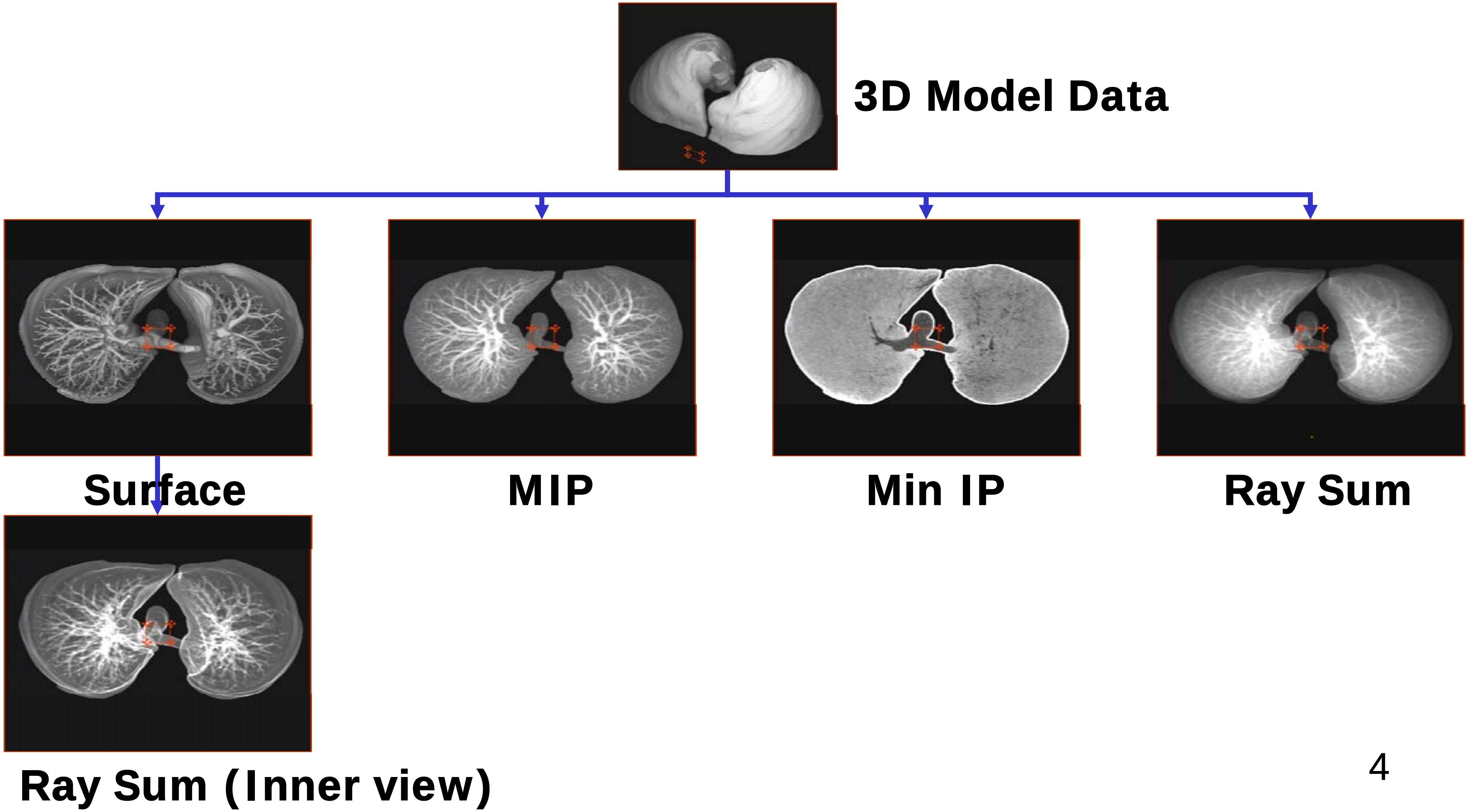
Visualization Methods



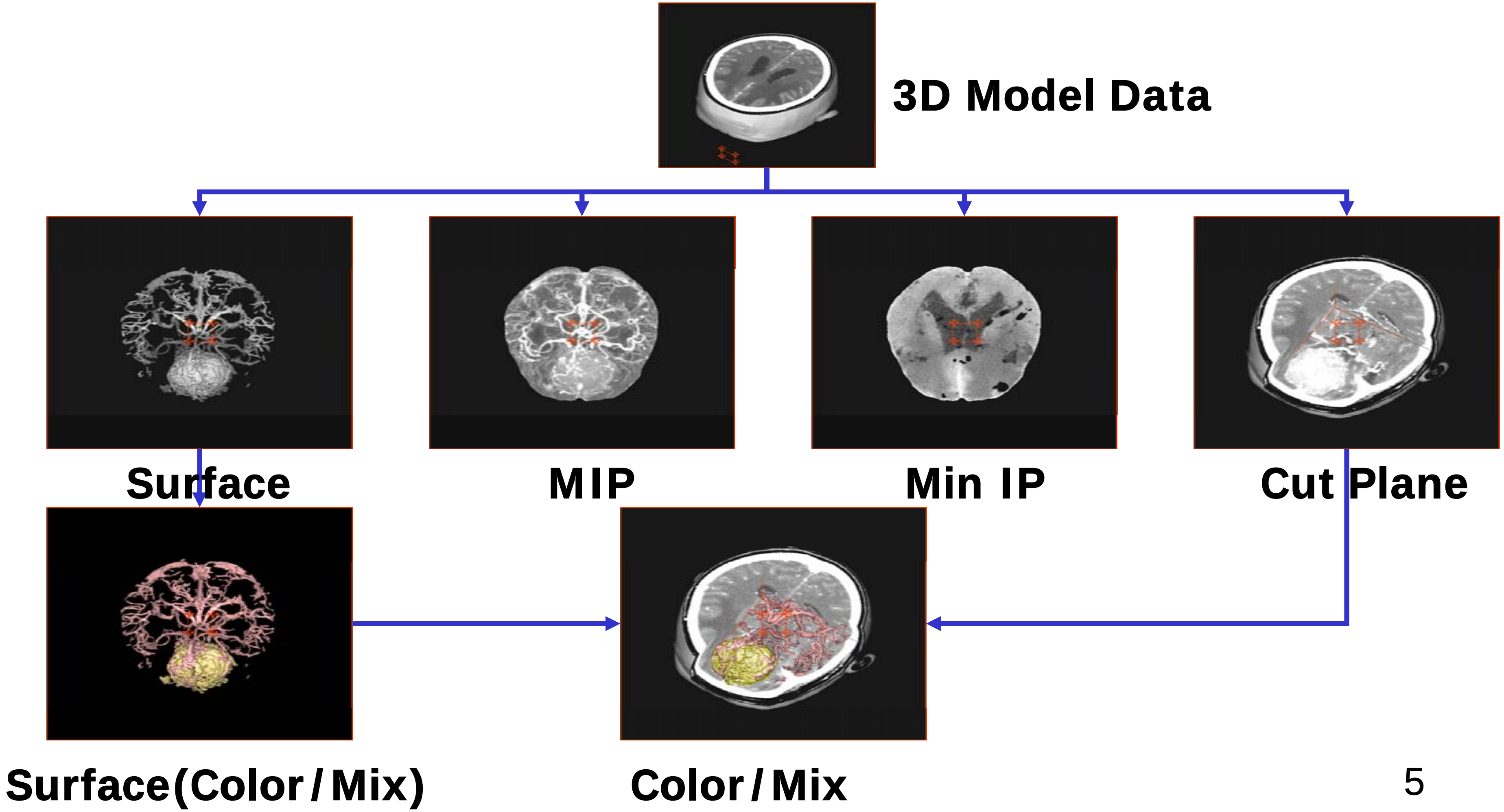
Visualization Methods



Visualization Methods

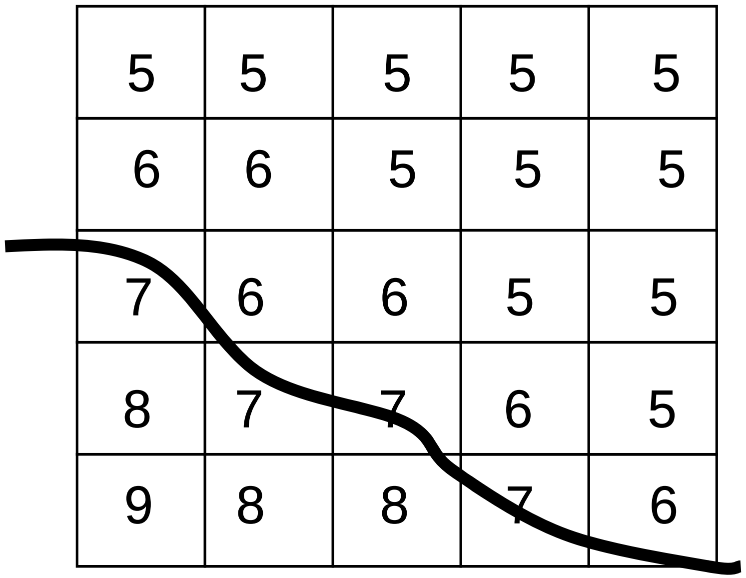


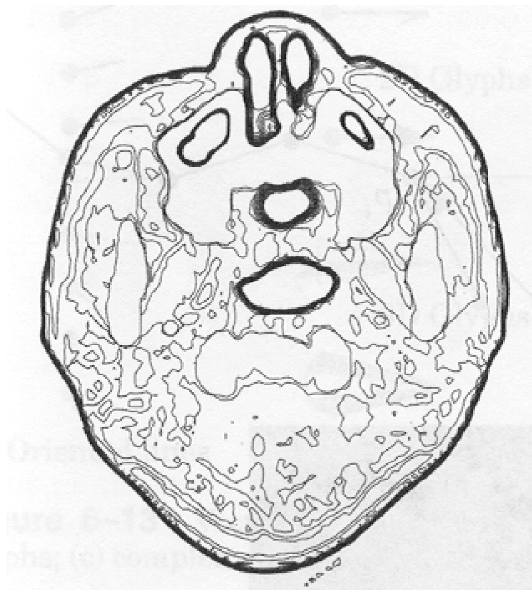
Visualization Methods



Wireframe Contours

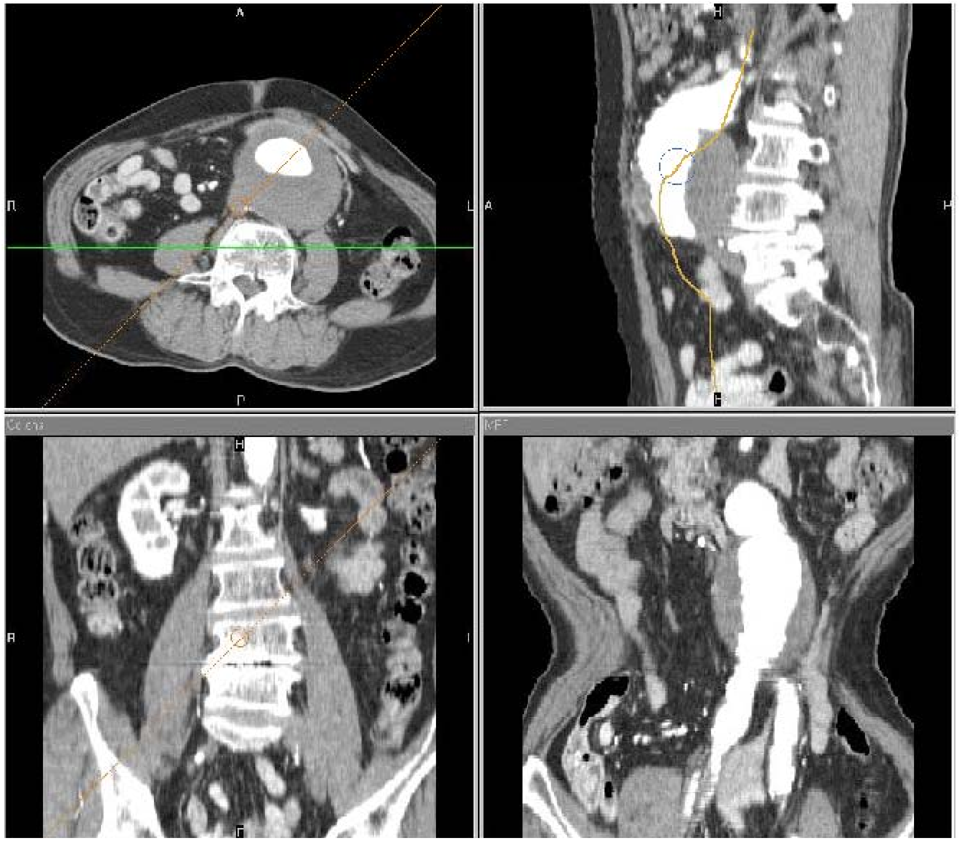
- Build a display list of isovalued lines
- Interactive methods
- Pro : interactive speed
- Con : minimal information in images





Multiplanar Reformation (MPR)

- Pseudo-color exposed extent and oblique planes
- Examples



Multiplanar Reformation

- Interactive viewing
- Reconstruction planes
  - Axial, coronal, sagittal
  - Oblique, double oblique
  - Curved planar reformation
  - Freehand drawing
- Special techniques
  - Display on 3D image
  - 3D MPR
  - Freely moving MPR plane on the volume

MPR

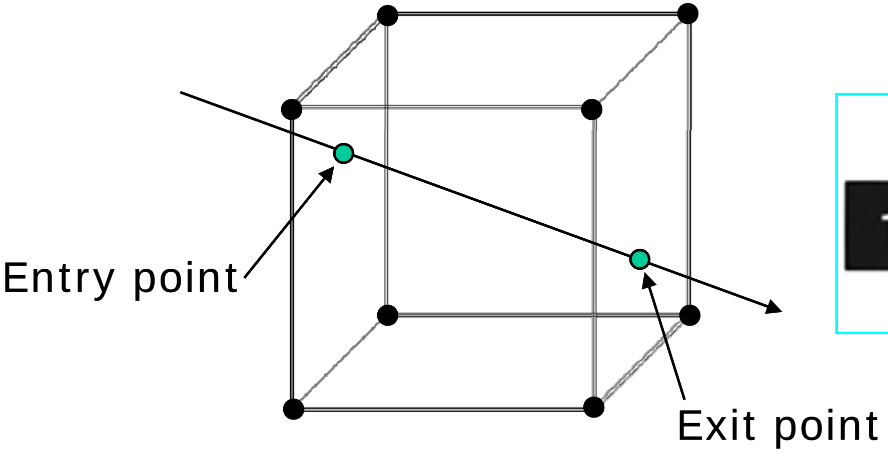
- Pros
  - Real-time, rapid
  - Entire information without loss of data
- Cons
  - 2D image
  - Lack of overall feature
  - Tortuous, branching structures

→ *Compensation with special techniques*

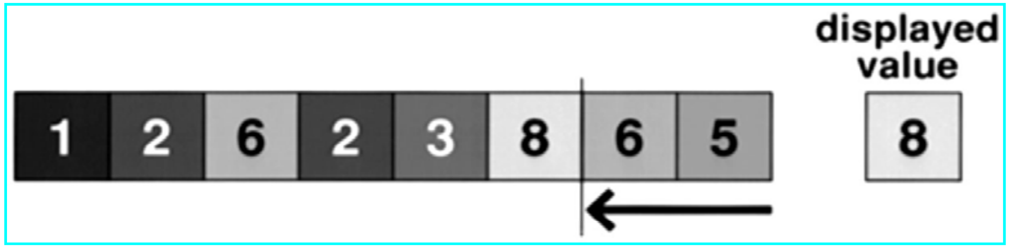
MIP / MinIP

- 2D projection image
- Cine-loop to convey 3D structures
- Useful in blood vessels, calcifications, airways & air-filled GI tract (high contrast with background)
- Necessary to remove overlapping structures

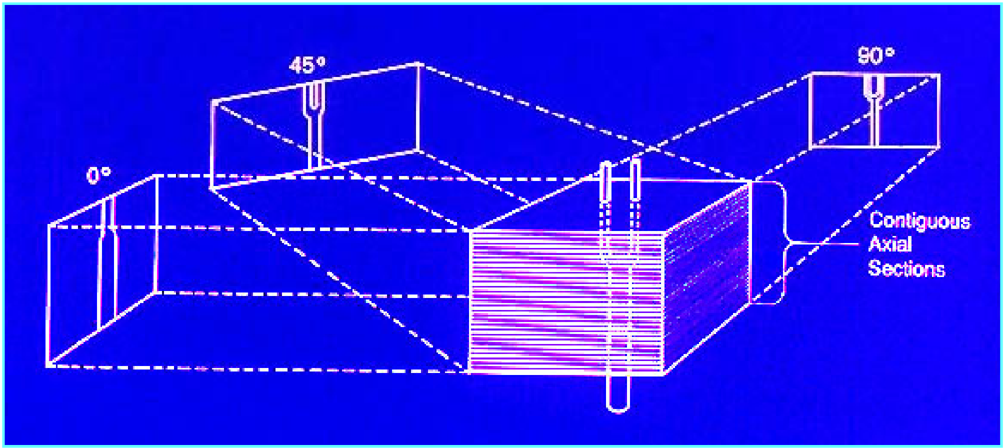
MIP



Entry point
Exit point



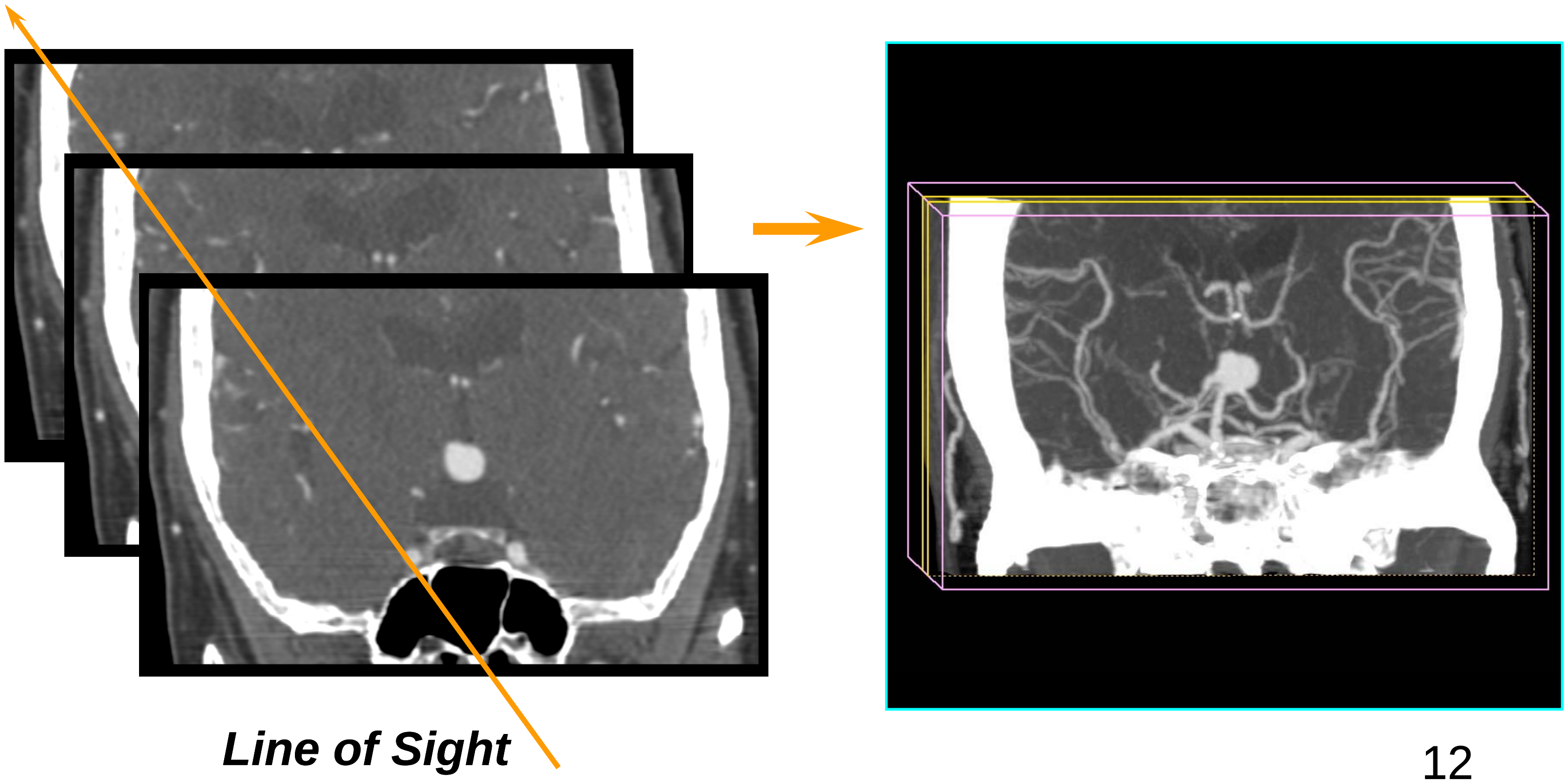
displayed value
1
2
6
2
3
8
6
5
8



45°
90°
0°
Contiguous Axial Sections

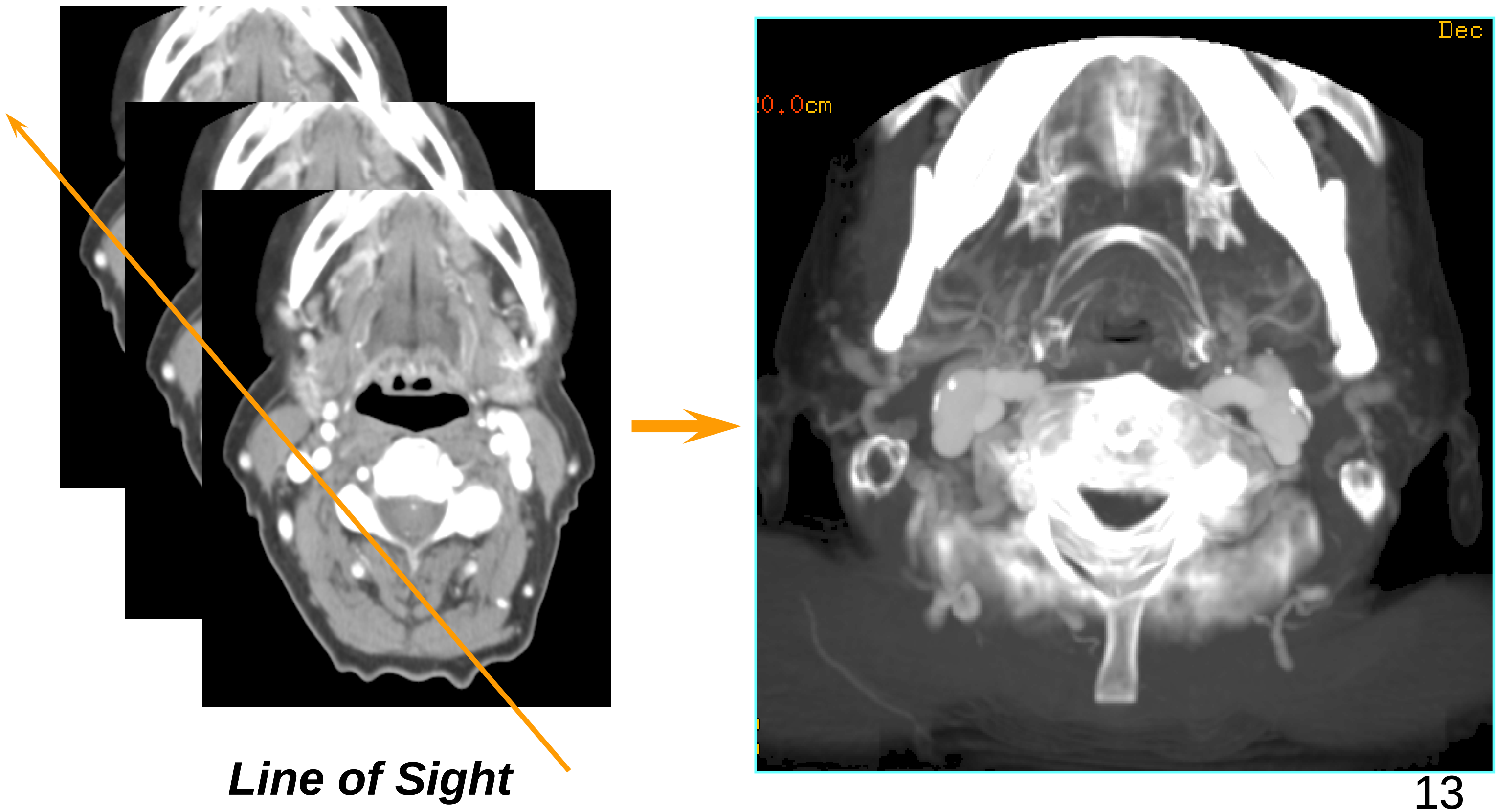
MIP

Maximum attenuation value is mapped to a 2D gray scale image

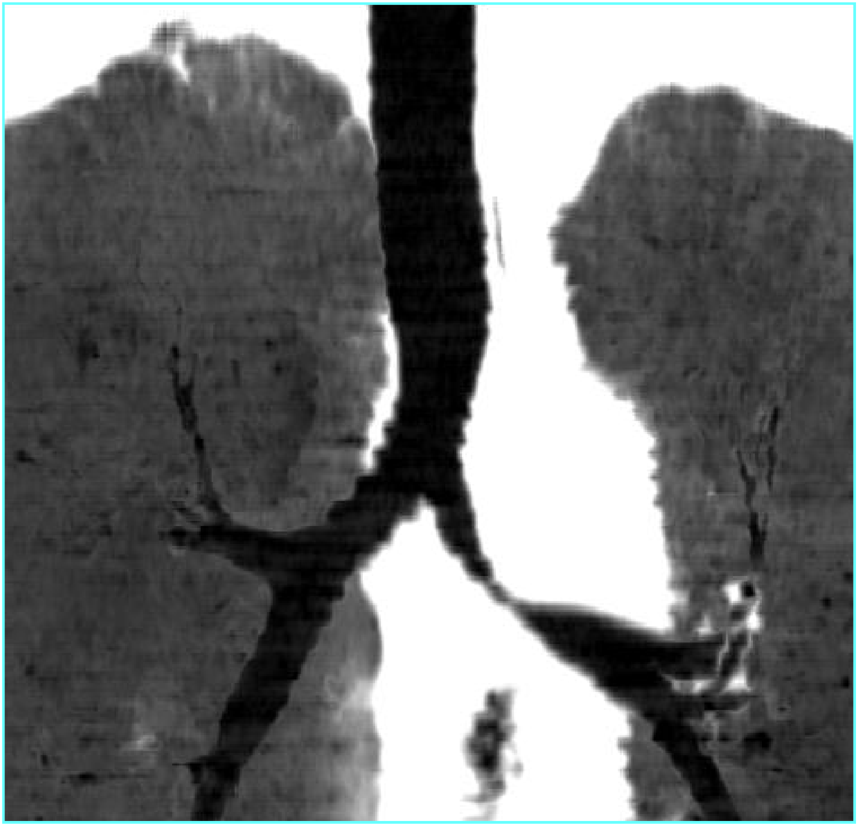


MIP

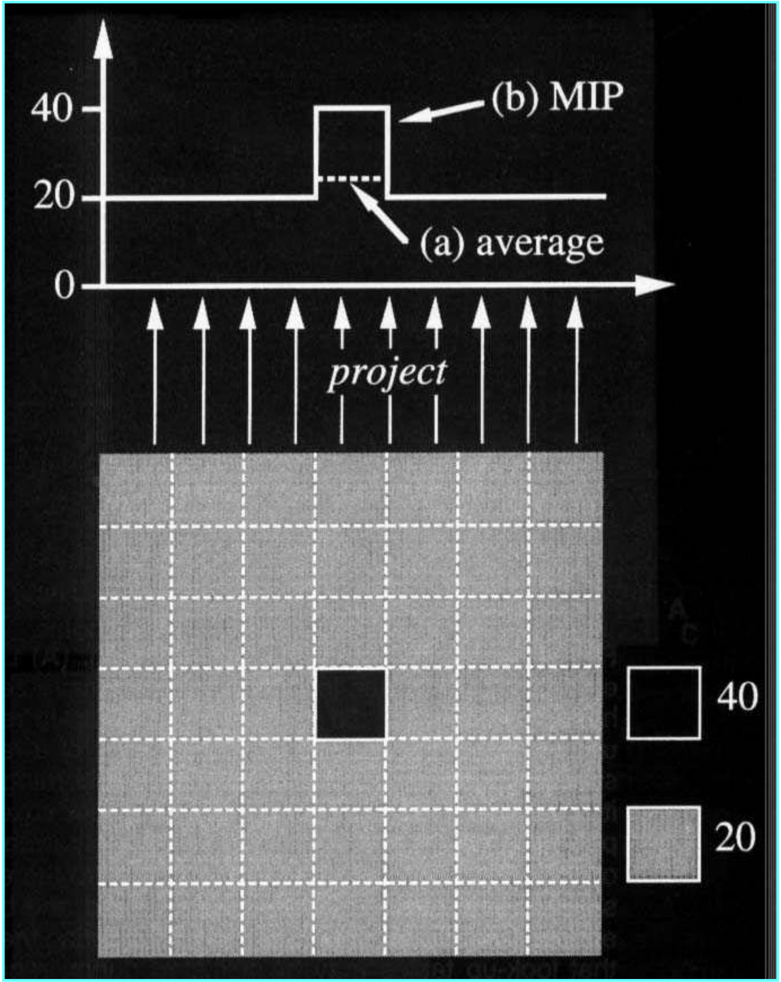
Maximum attenuation value is mapped to a 2D gray scale image

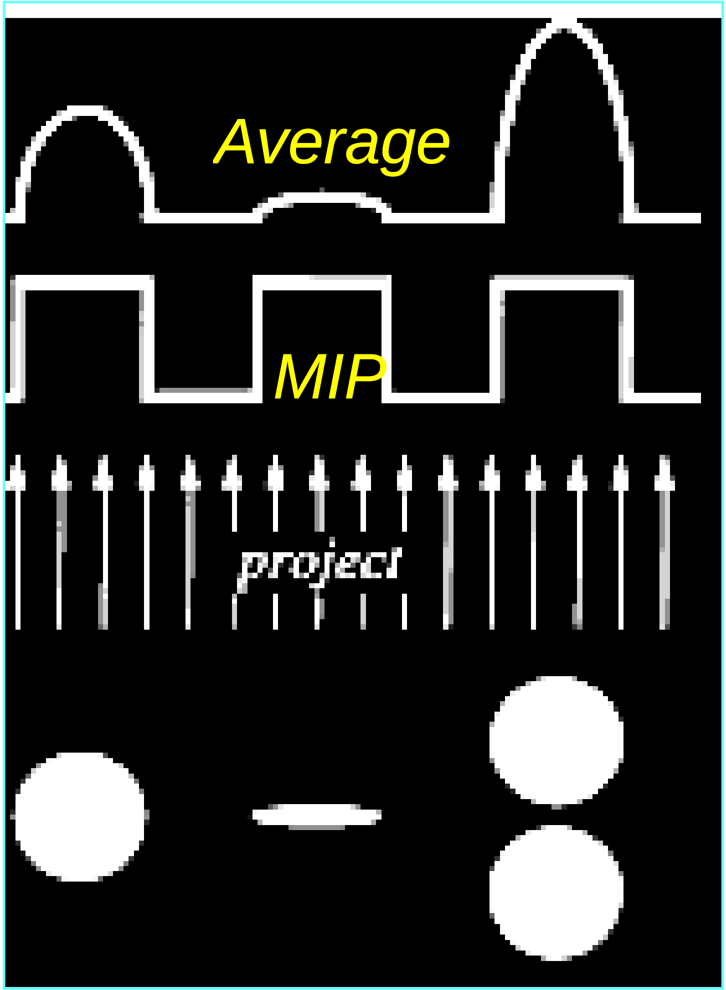


MinIP



MIP vs Average





Surface Rendering (Geometry Rendering)

- Objects are defined in terms of surfaces
- Converts the data into an intermediate surface representation before the rendering
- volume data $\Rightarrow$ geometric primitives

  : surface reconstruction (surface fitting)
- Contouring, Marching cubes, Dividing cubes

Surface Rendering Pros

- Render in hardware
- Changing View/Light(s) requires only rendering
- Compact storage & transmission
- Render in object order or image order
- Good spatial coherence for efficient rendering

Surface Rendering Cons

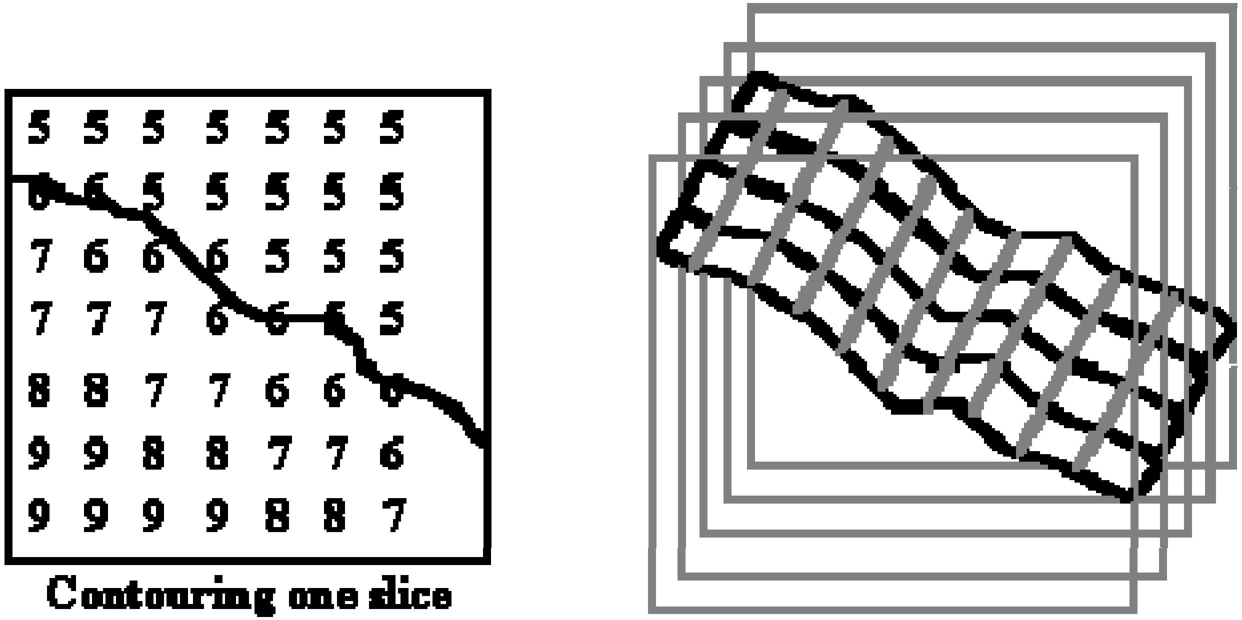
- Requires Binary Classification
- Throws Away Data Between Surfaces
- False Positives and Negatives
- Handles Small Features Poorly
- Not easy to Handle Branching
- User Intervention Sometimes Required
- Amorphous Data Doesn't Have "Thin Surfaces"
- Modeling complexity : 10 minutes to extract
- 500,000 Triangles

Contouring

- A contour is a line drawn on a plan joining all points of the same (OR above or below) intensity value.
- Extract contours at each section and connect them together
- The plan spacing between contour line indicates the steepness of slopes.
  - Closely spaced lines indicates a steep gradient.
  - Widely spaced lines indicate a flatter gradient.

Contouring

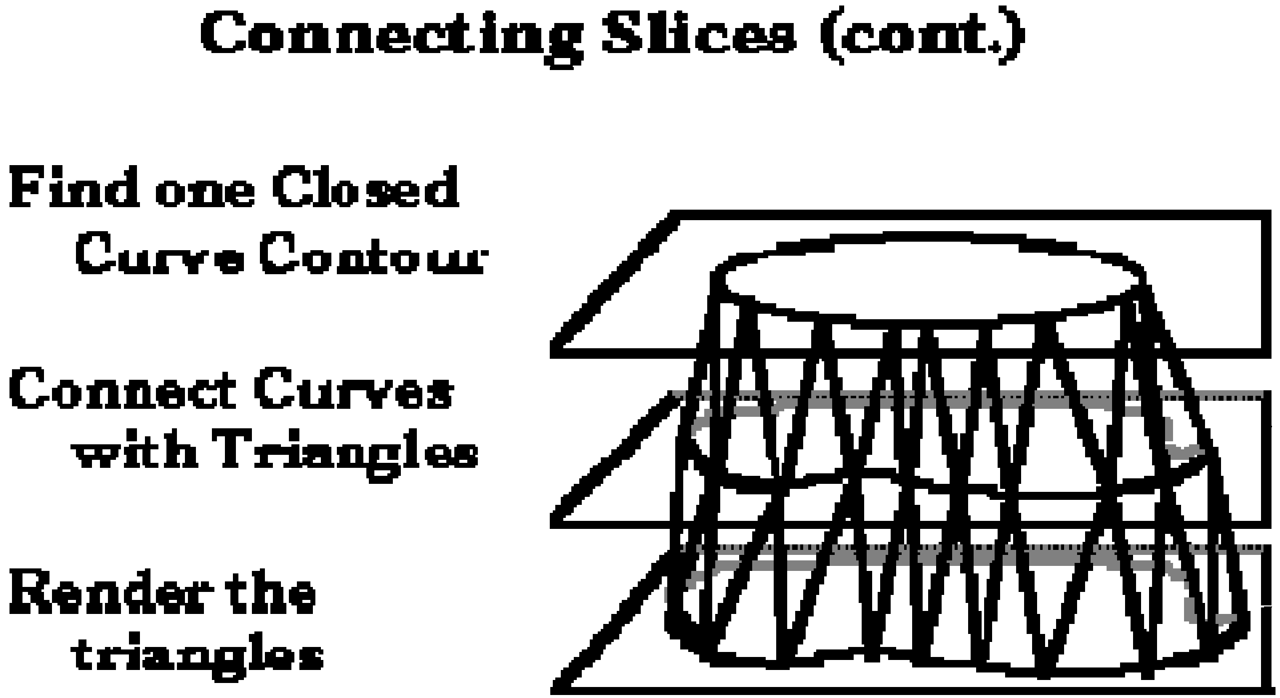
Contouring (object order)



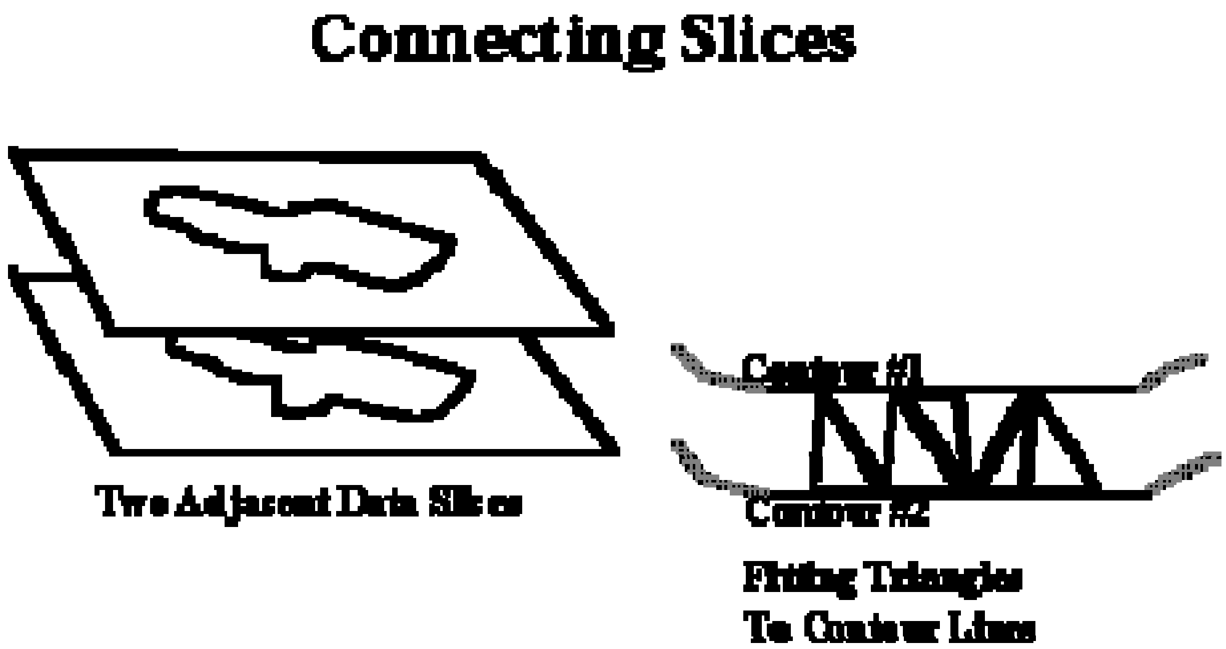
Contouring one slice
T=6

Contouring five identical slices

Contouring


Connecting Slices (cont.)
Find one Closed Curve Contour
Connect Curves with Triangles
Render the triangles

Contouring


Connecting Slices
Two Adjacent Data Slices
Contour #1
Contour #2
Fitting Triangles
To Contour Lines

4 Problems of Contouring

- Correspondence problem
- Tiling problem
- Branching problem
- Surface-fitting problem

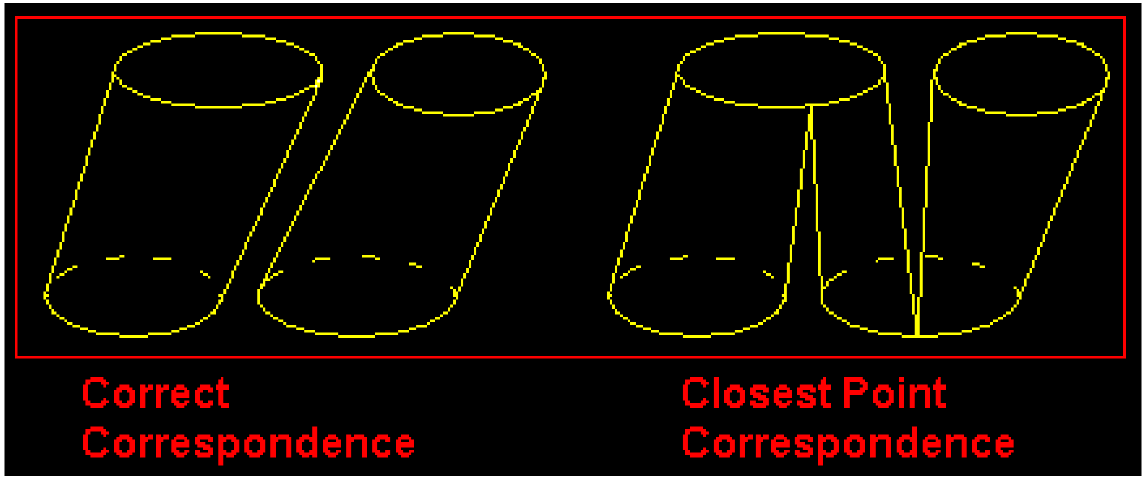
(Meyers, Skinner, and Sloan “*Surfaces from Contours*” 1992)

Correspondence Problem of Contouring

- Which contours should be connected together by the surface?
  - The question arises whenever there are multiple contours in a section
  - Solution: arrange contours into groups organized by which objects they represent
    - Difficult to make automatic without making assumptions because of the ambiguities
    - Usually done by a human in the contour editor

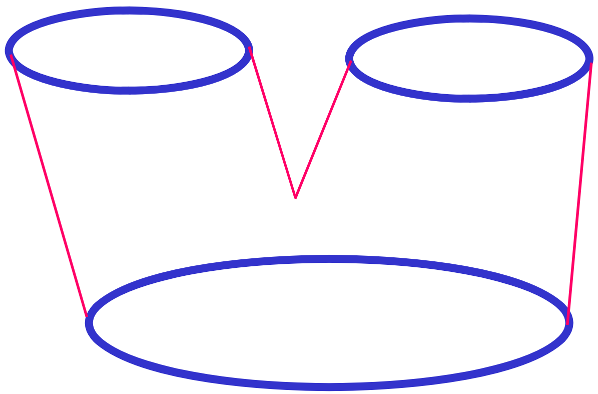
Tiling Problem of Contouring

- How should the contours be connected?



Branching Problem of Contouring

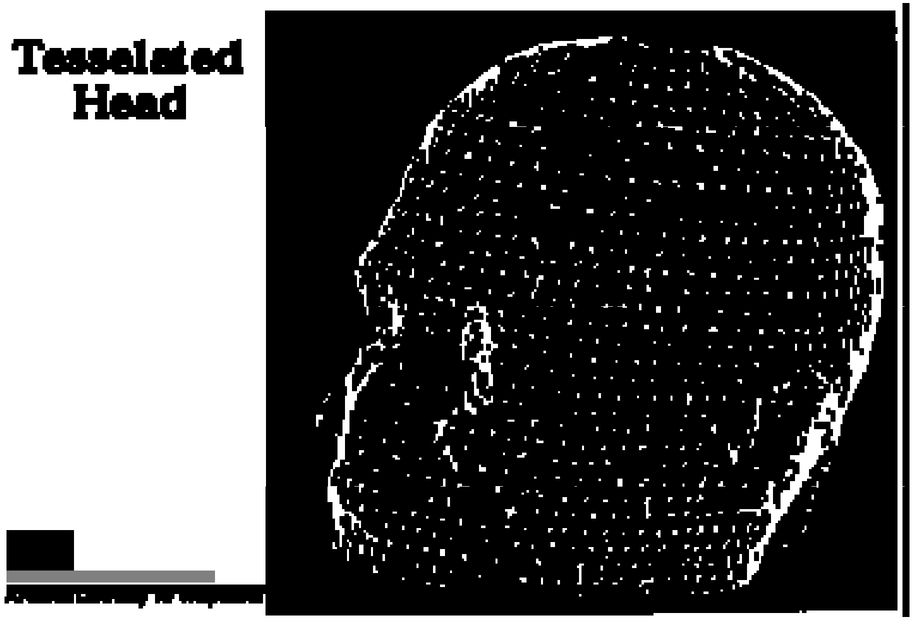
- What do we do when there are branches in the surface?
  – One solution is user intervention

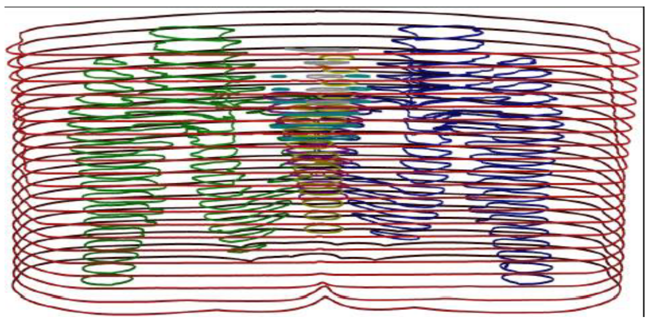


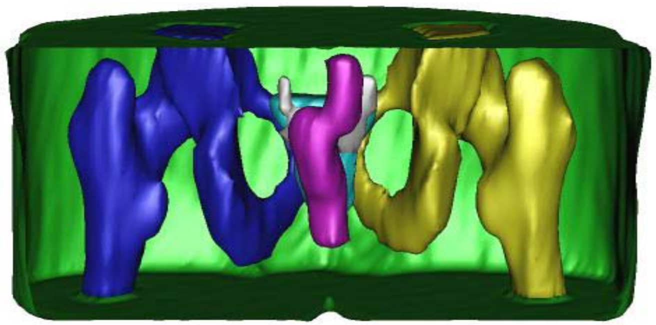
Surface-fitting Problem of Contouring

- Once a triangulated mesh is found, how should a smooth surface be made that approximates or interpolates the vertices of the mesh and maintains the same topology?

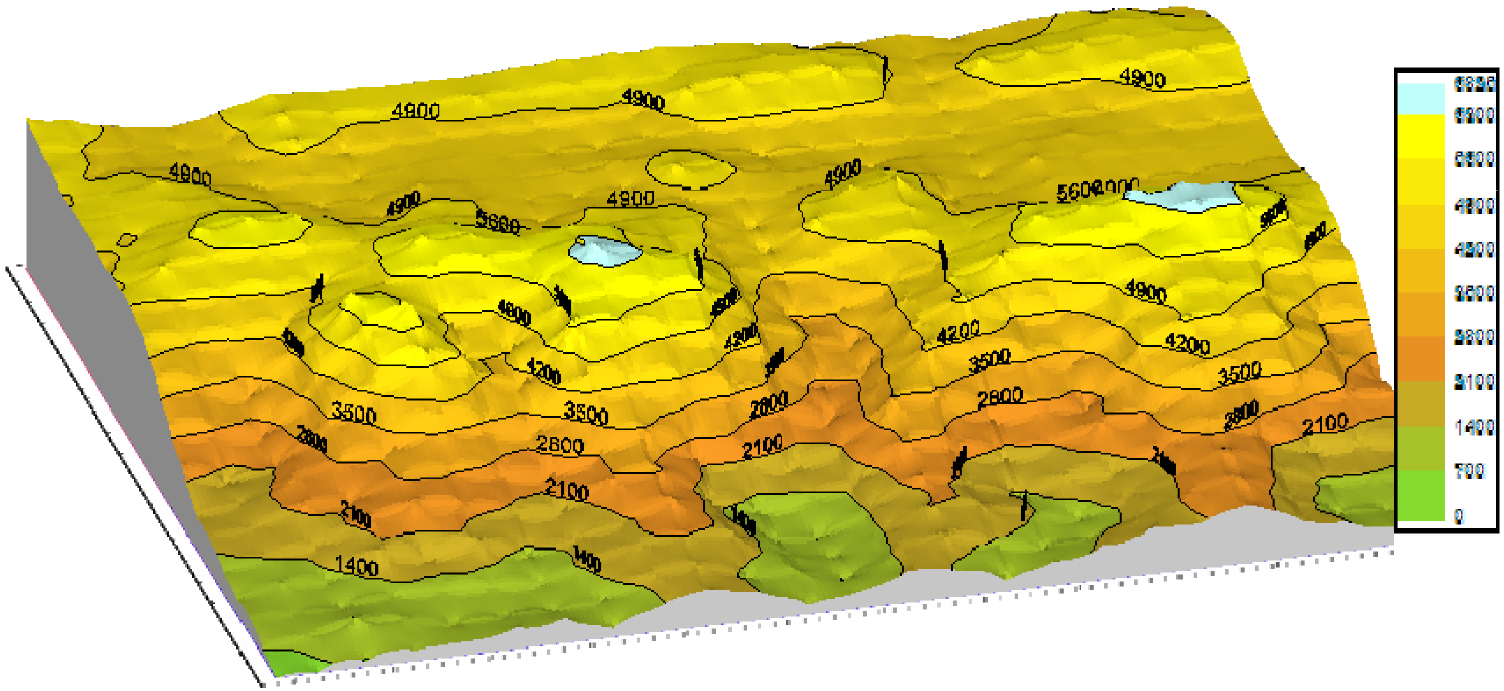
Contouring





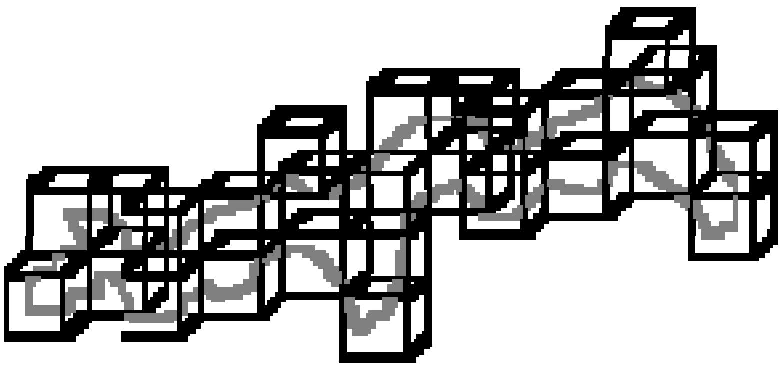


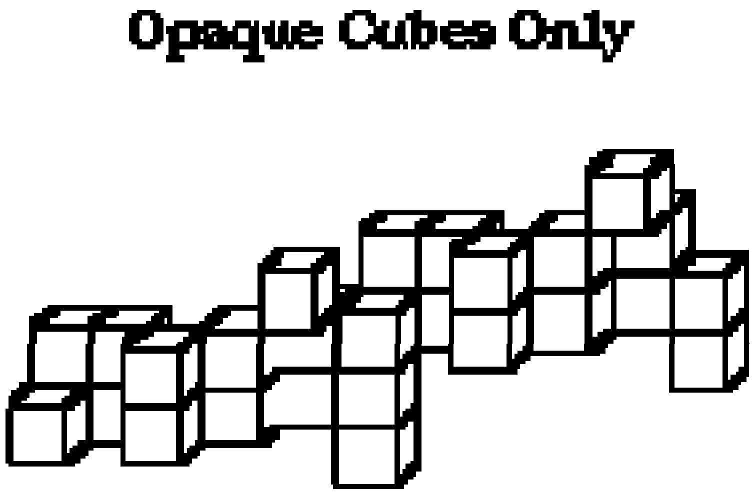
3D Contour Maps



Opaque Cube

- All ON cells are represented as hexahedra
- Render six faces with traditional methods





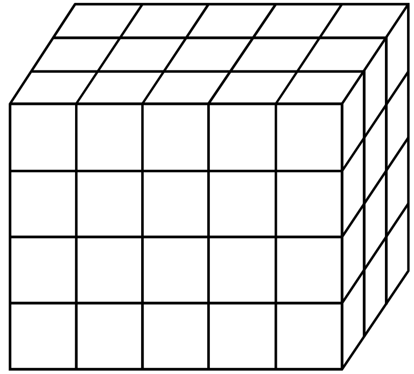
Opaque Cube

- Pros
  - Straight-forward to Implement
  - Data Classification is Easy
  - One Threshold Value
- Cons
  - Jaggies
  - Same Problems as Surface Fitting

Marching Cubes Algorithm

- Developed by William E. Lorensen and Harvey E. Cline, 1987
- Creates a constant density surface from a 3D array of data
- Create a triangular mesh that will approximate the iso-surface
- Can use a general purpose graphics pipeline
- Slow for typical medical applications

  ← too many polygons
- Drawback: Possible holes in the model



Marching Cube - Steps

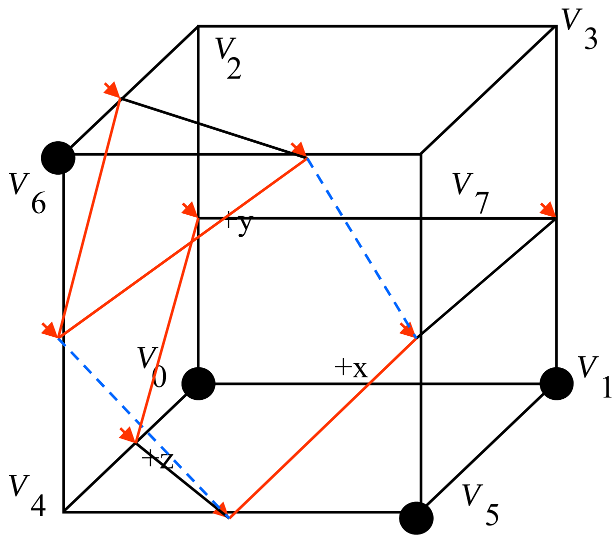
Volume is subdivided into unit cubes, which are analyzed and replaced by an appropriate set of polygons

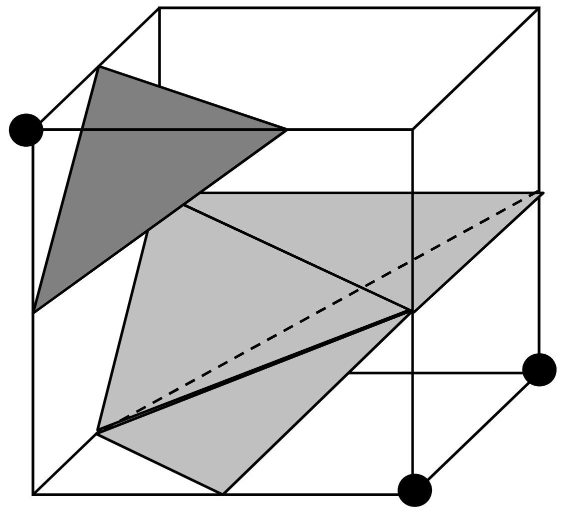
1. Create cells (cubes)
2. Classify each vertex
3. Build an index
4. Get edge list based on table look-up
5. Interpolate triangle vertices
6. Interpolate normals
7. Obtain polygon list and do shading in image space

Marching Cubes Algorithm

- For each voxel, set vertex flags
  - Each vertex of the voxel is assigned a binary (0 or 1) value based on whether scalar value is higher or lower than surface value.

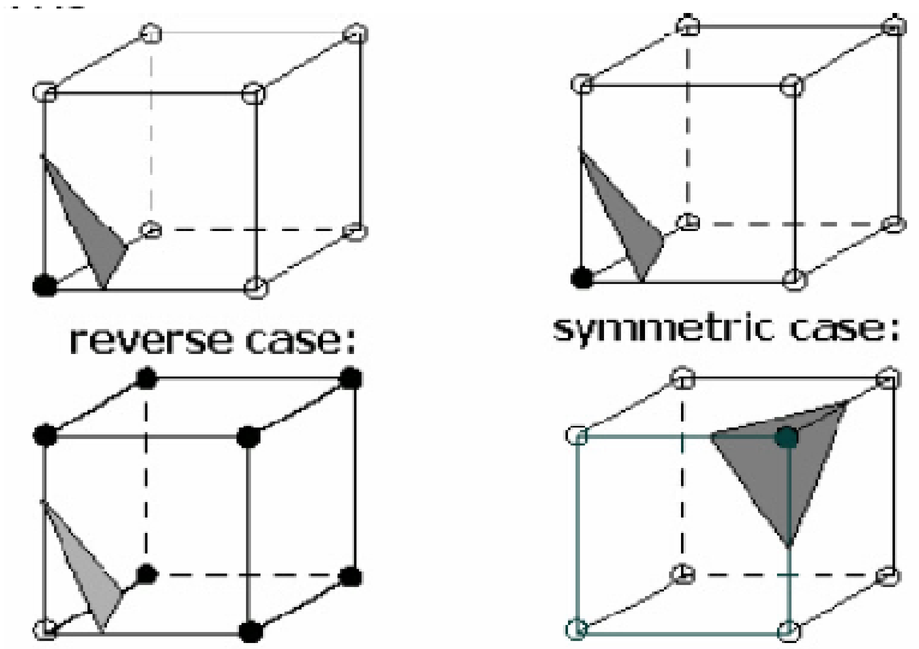
$$\text{if } F(i,j,k) \geq \alpha$$





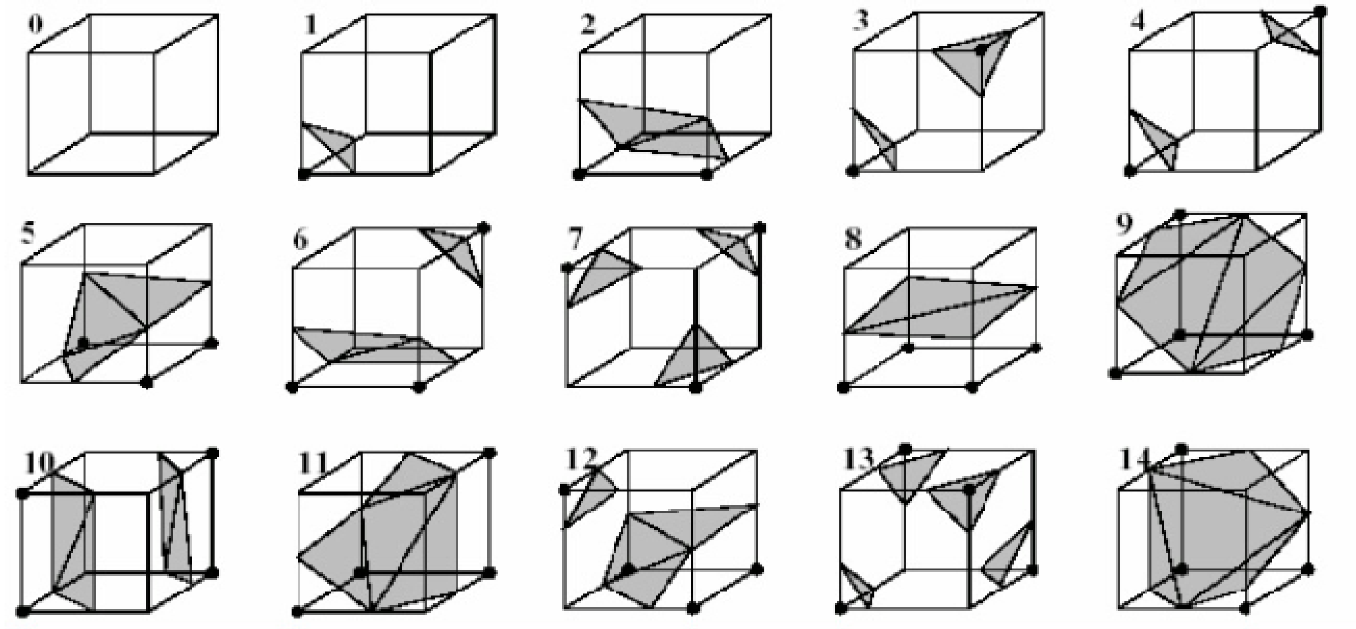
Edge list

For a given vertex flag (➔ index), edge table returns a list of edges that contains a triangle vertex.



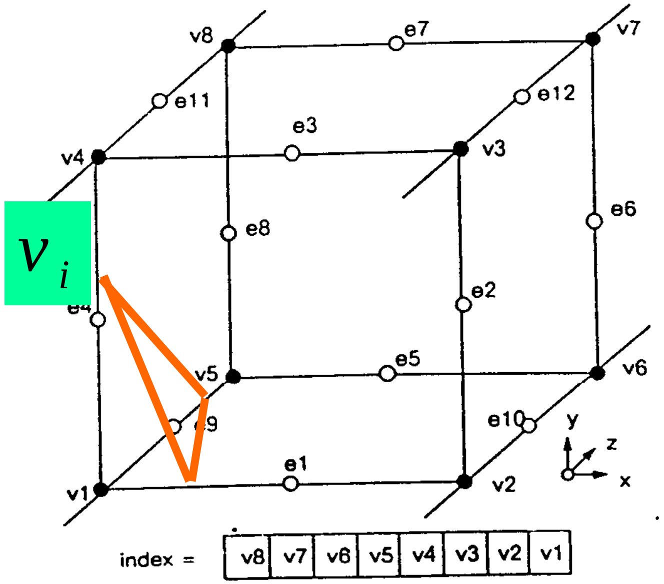
Edge list

Because rotation symmetry and complementary, 256 cases are reduced to 15 cases.



Surface intersection in a cube

- Interpolate surface intersection along each edge



$$v_i = v_1(1-t) + tv_4$$

$$t = \frac{v_1 - v_i}{v_1 - v_4}$$

Calculate Normals in MC

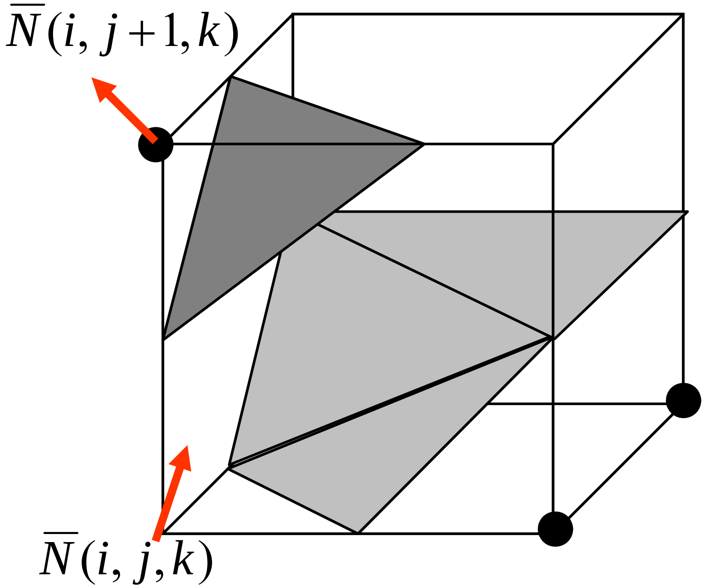
- Normals at Vertices of Cubes using Central difference

$$\overrightarrow{N}(x,y,z) = \nabla \overrightarrow{f}(x,y,z)$$

$$N_x(i,j,k) = \frac{D(i+1,j,k) - D(i-1,j,k)}{\Delta x}$$

$$N_y(i,j,k) = \frac{D(i,j+1,k) - D(i,j-1,k)}{\Delta y}$$

$$N_z(i,j,k) = \frac{D(i,j,k+1) - D(i,j,k-1)}{\Delta z}$$

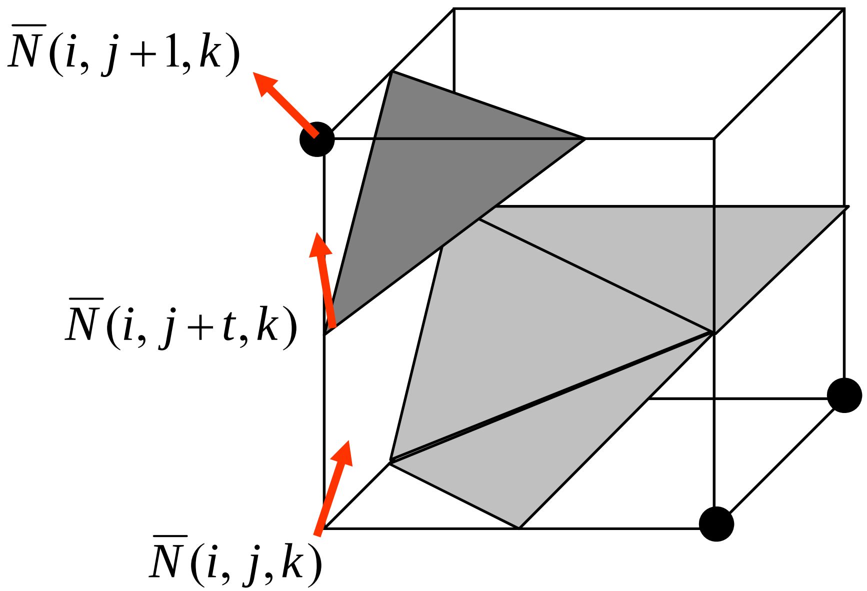


You need to normalize!

$$\overline{N}(i,j,k) = \frac{N(i,j,k)}{\|N(i,j,k)\|}$$

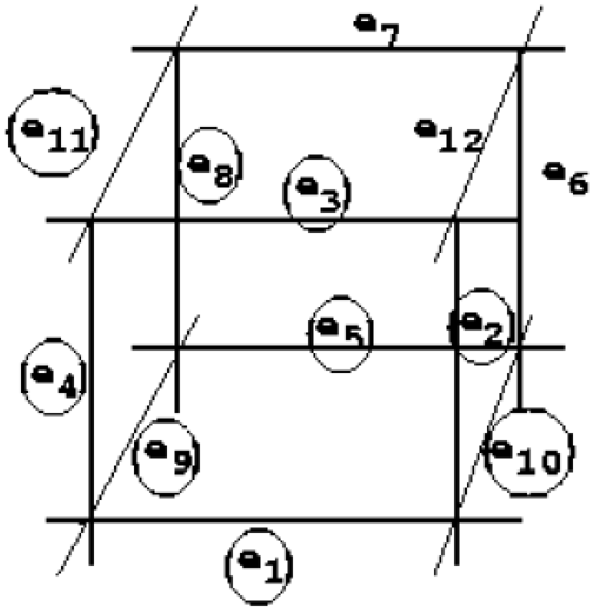
Calculate Normals in MC

- Normals at Vertices of Triangles
  - Linear interpolations of two vertices of cubes
- Keeping 4 slices in memory at once

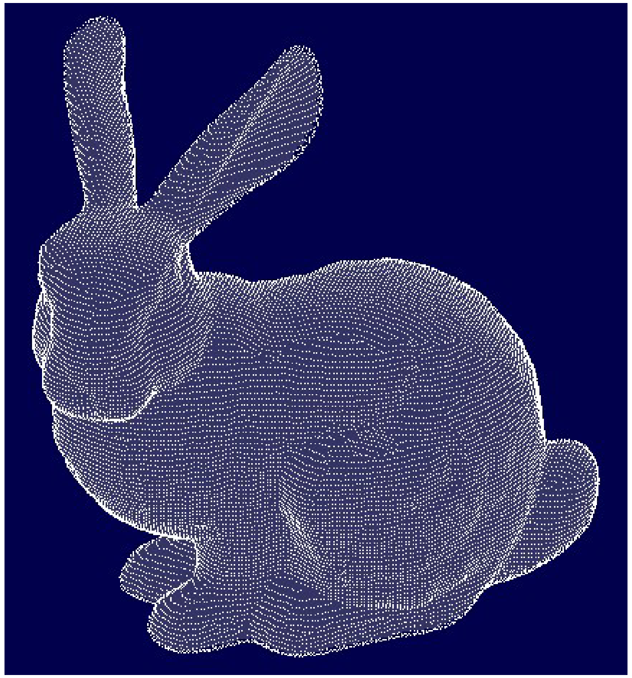


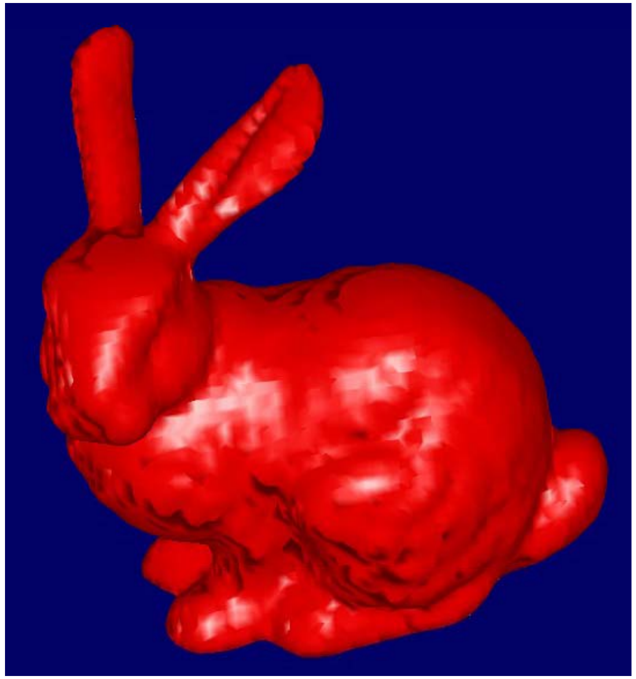
Efficiency Enhancement in MC

- Line-to-line/slice-to-slice coherence
  - For interior cubes, only 3 new edges need to be interpolated for each cube, other 9 edges are obtained from previous slices, lines, or pixels.

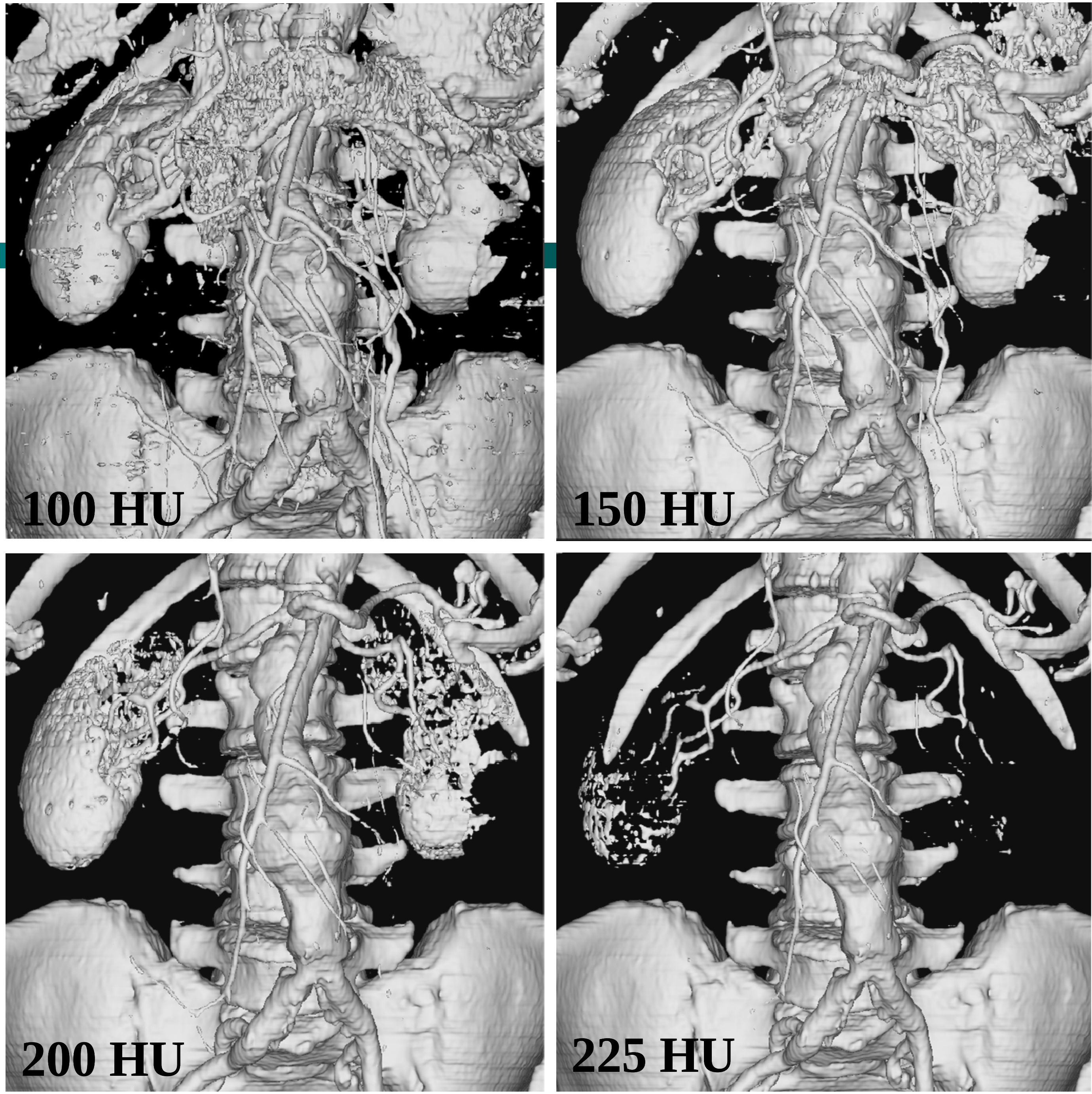


Bunny Example (MC)



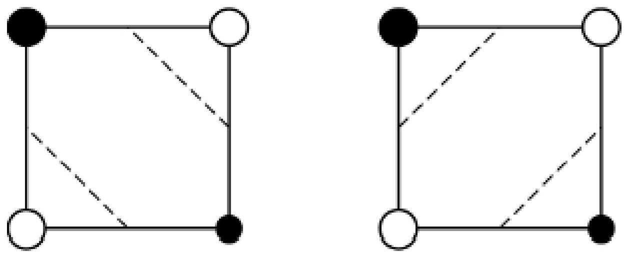


Shaded Surface Display



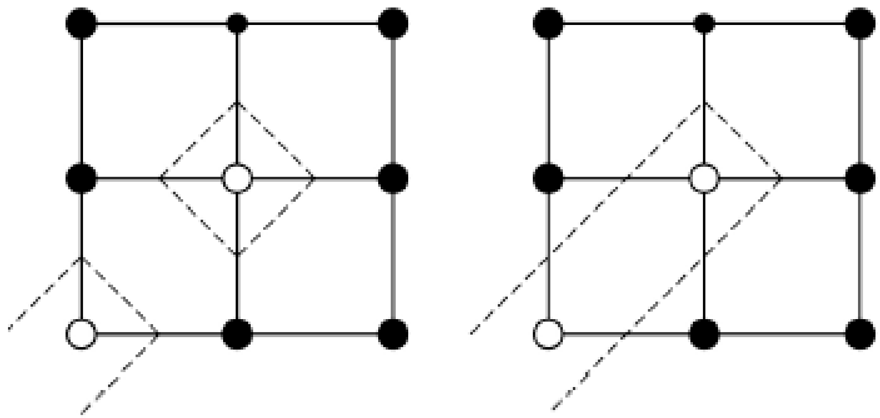
Possible Holes of MC Algorithm

The reason – “ambiguous labels”

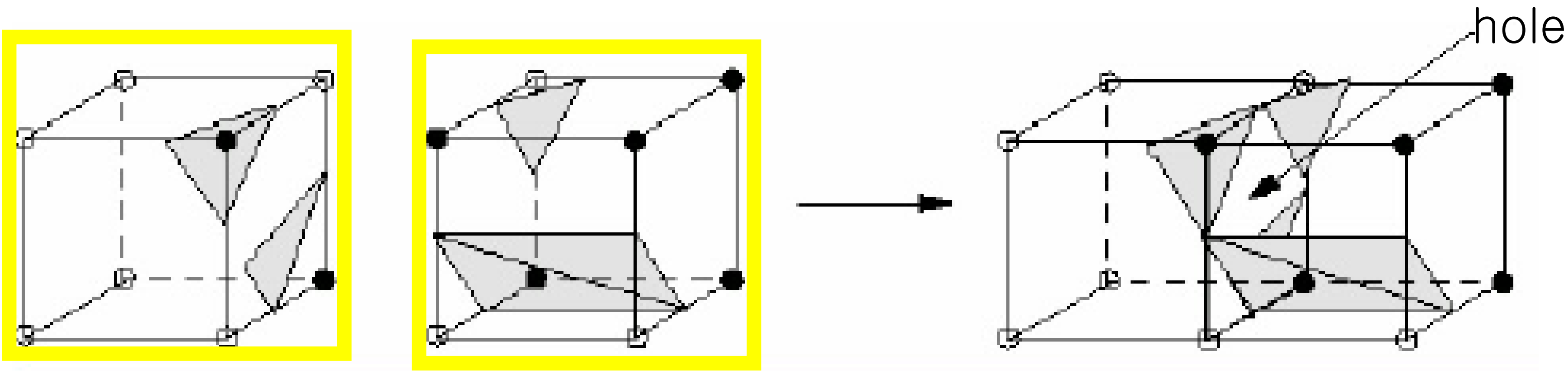


in 2D

Different resulting contours



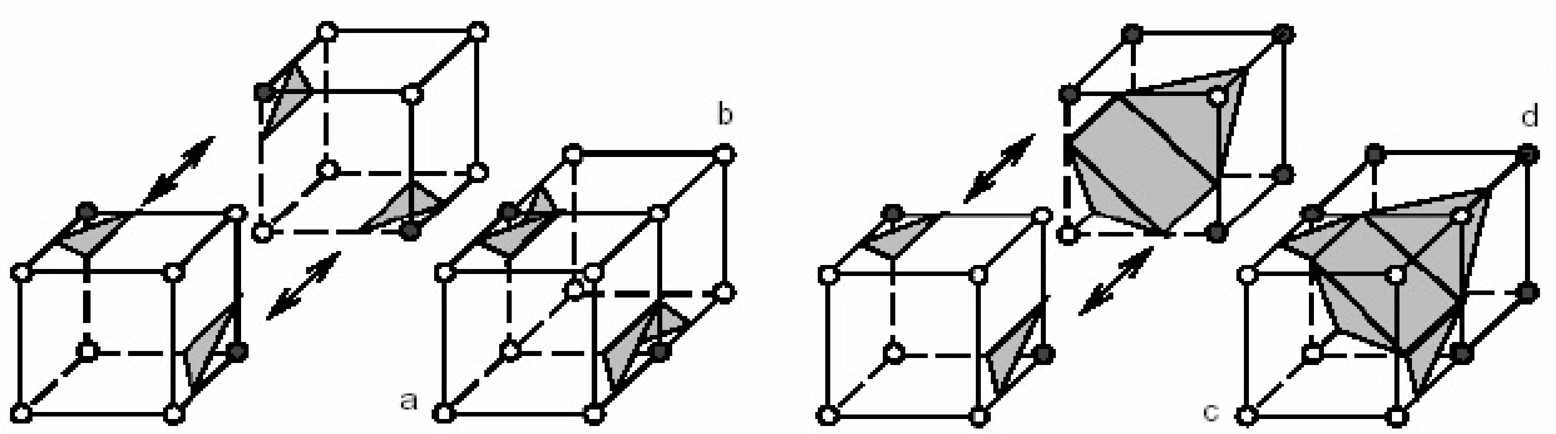
Possible Holes of MC Algorithm



These are: Cases 3, 6, 7, 10, 12, 13

Data Set Threshold	Dolphin 120.2	Dolphin 1125.2	Human 129.5	Aircraft Fin 1.0
# of cells	3,718,093	3,718,093	2,880,405	31,117
Cells with Isosurface	106,612	64,908	364,173	4,036
Ambiguous cells	3.1 %	3.3 %	5.5 %	5.6 %

Possible Holes of MC Algorithm

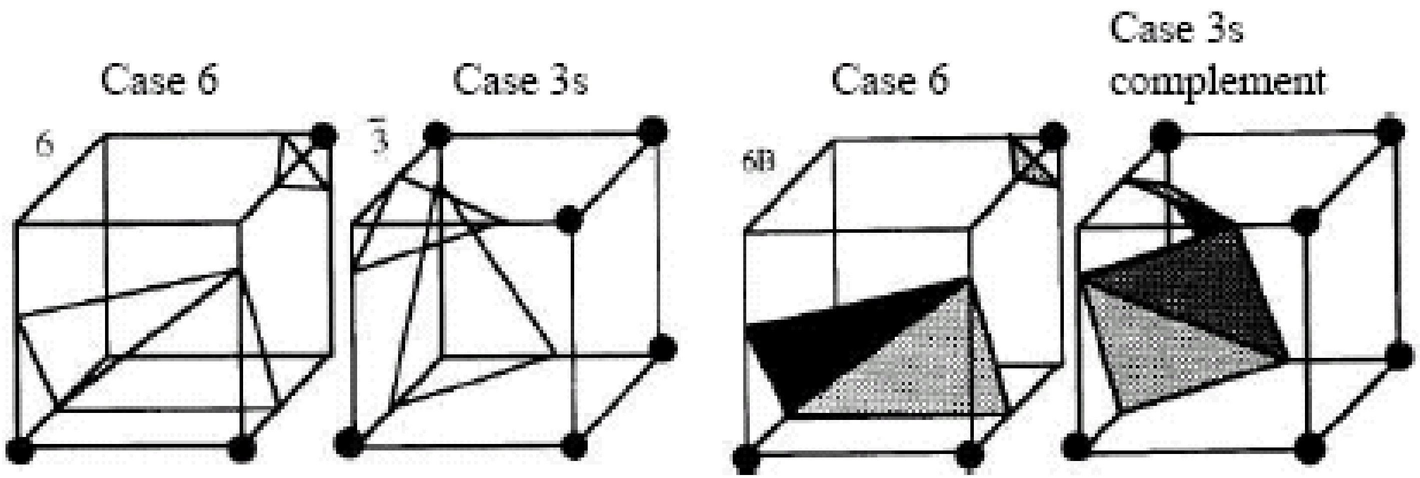


Resolve ambiguities of MC Algorithm

- “Resolving the Ambiguity in Marching Cubes”, Nielson and Hamann, Visualization '91, pages 83-91
- Be consistent!
  - Then resulting surface should contain no ambiguities
  - Store the edge intercepts that you have computed for the cube being considered
  - In the next march step, do not recalculate intercepts for the common face - they have been done.
- Use the triangular facets that would be consistent with the edge intercepts computed for the previous cube

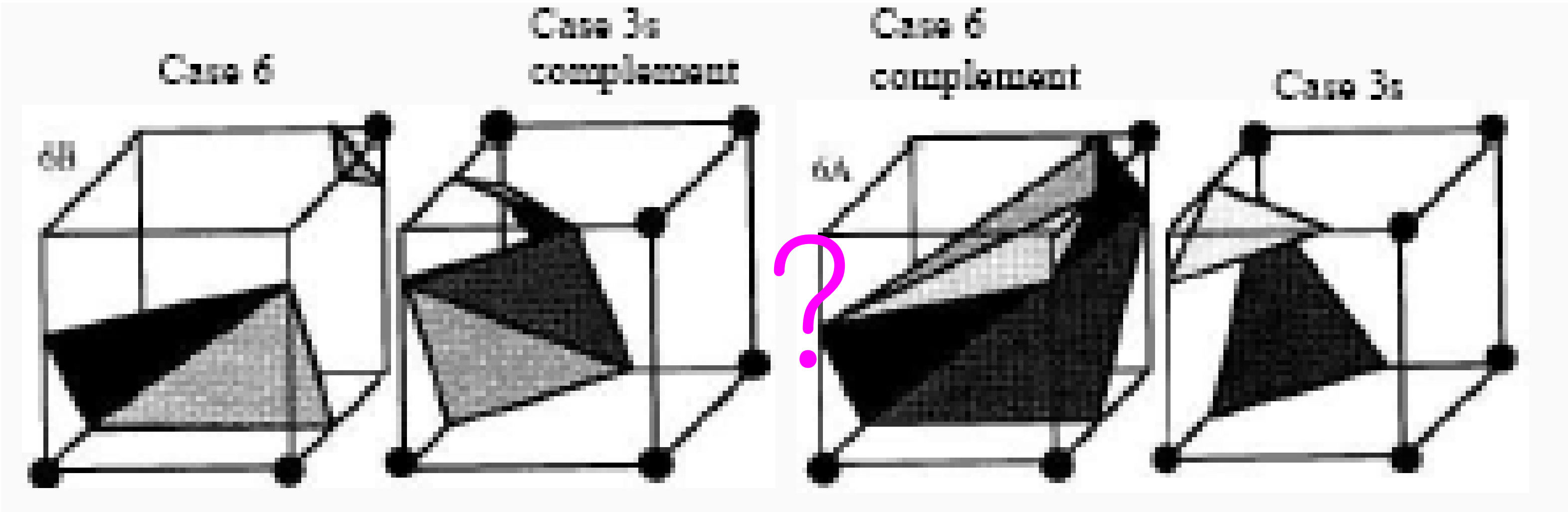
Resolve ambiguities of MC Algorithm

- If topology of the current cell is not consistent with the previous neighbor cell then we should consider taking the complementary topology of the current cell.



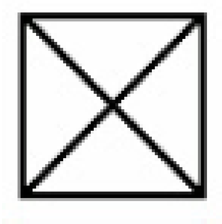
Resolve ambiguities of MC Algorithm

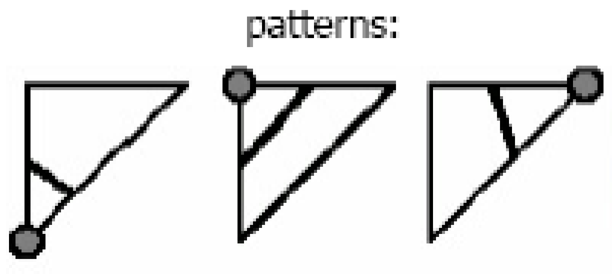
- Using this method, the resultant surface will complete though not necessarily correct.



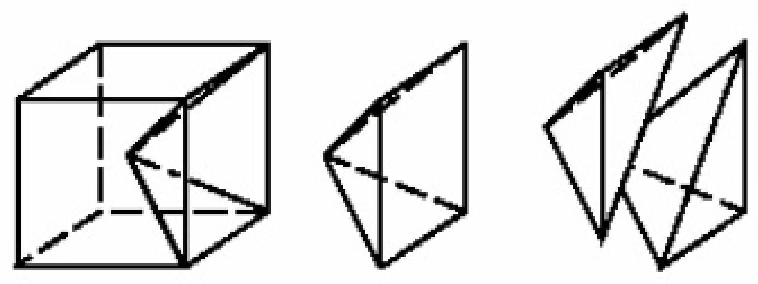
Resolve ambiguities II

- Simple Subdivision
- 2D

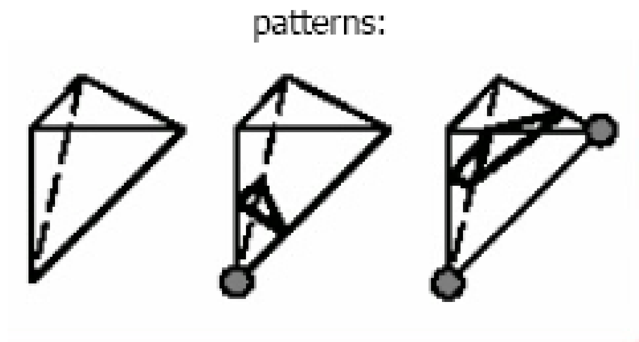




- 3D



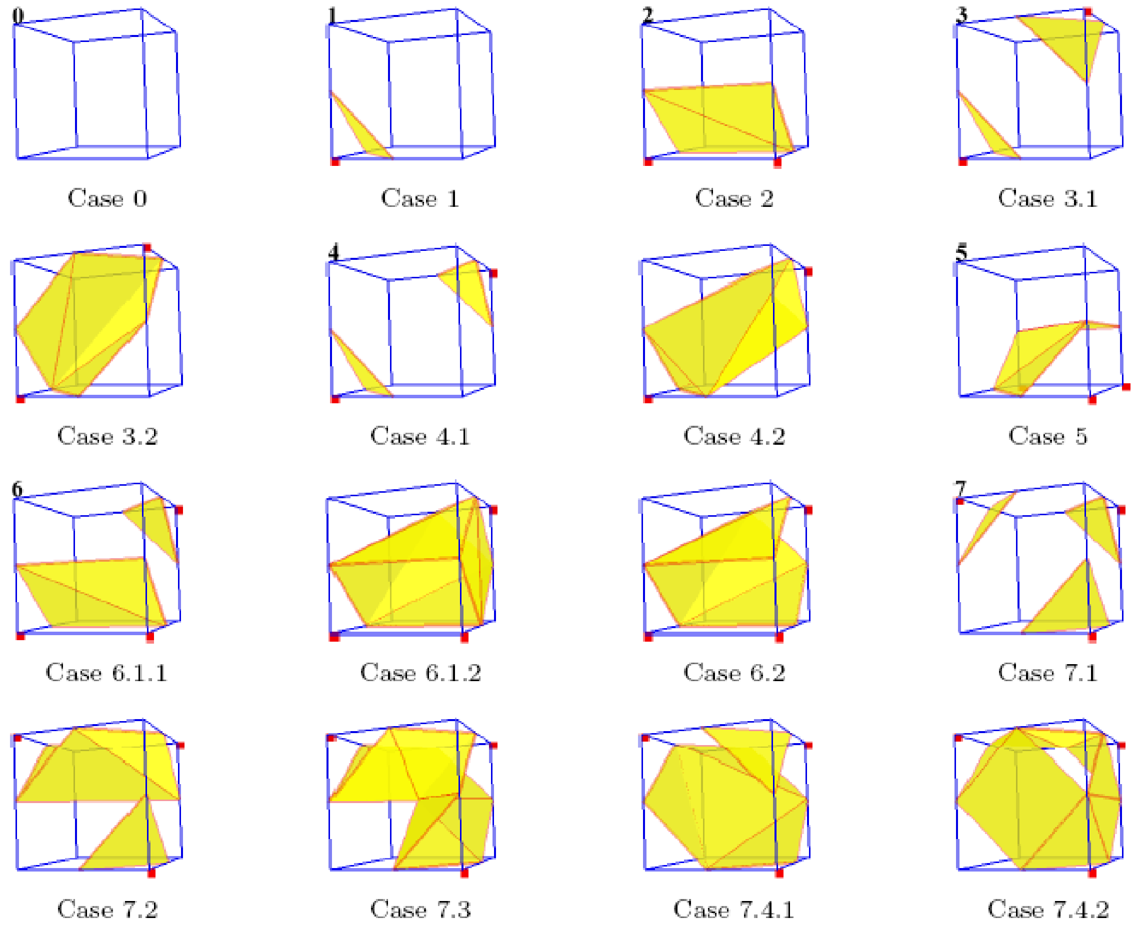
12 tetrahedra in the cube



➔ A disadvantage of the marching tetrahedra is the large number of triangles which are created - slowing down the rendering time

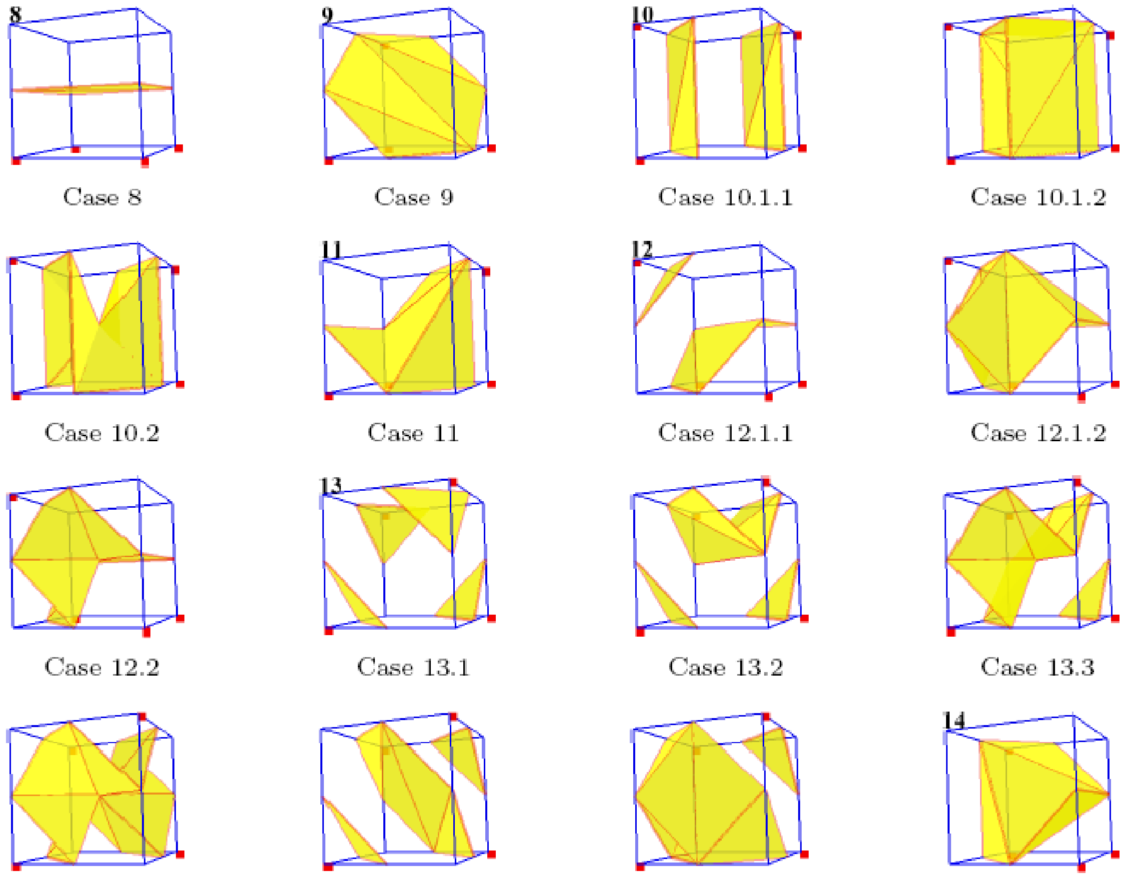
Resolve ambiguities III

- Marching cubes 33 by Evgeni Chernyaev



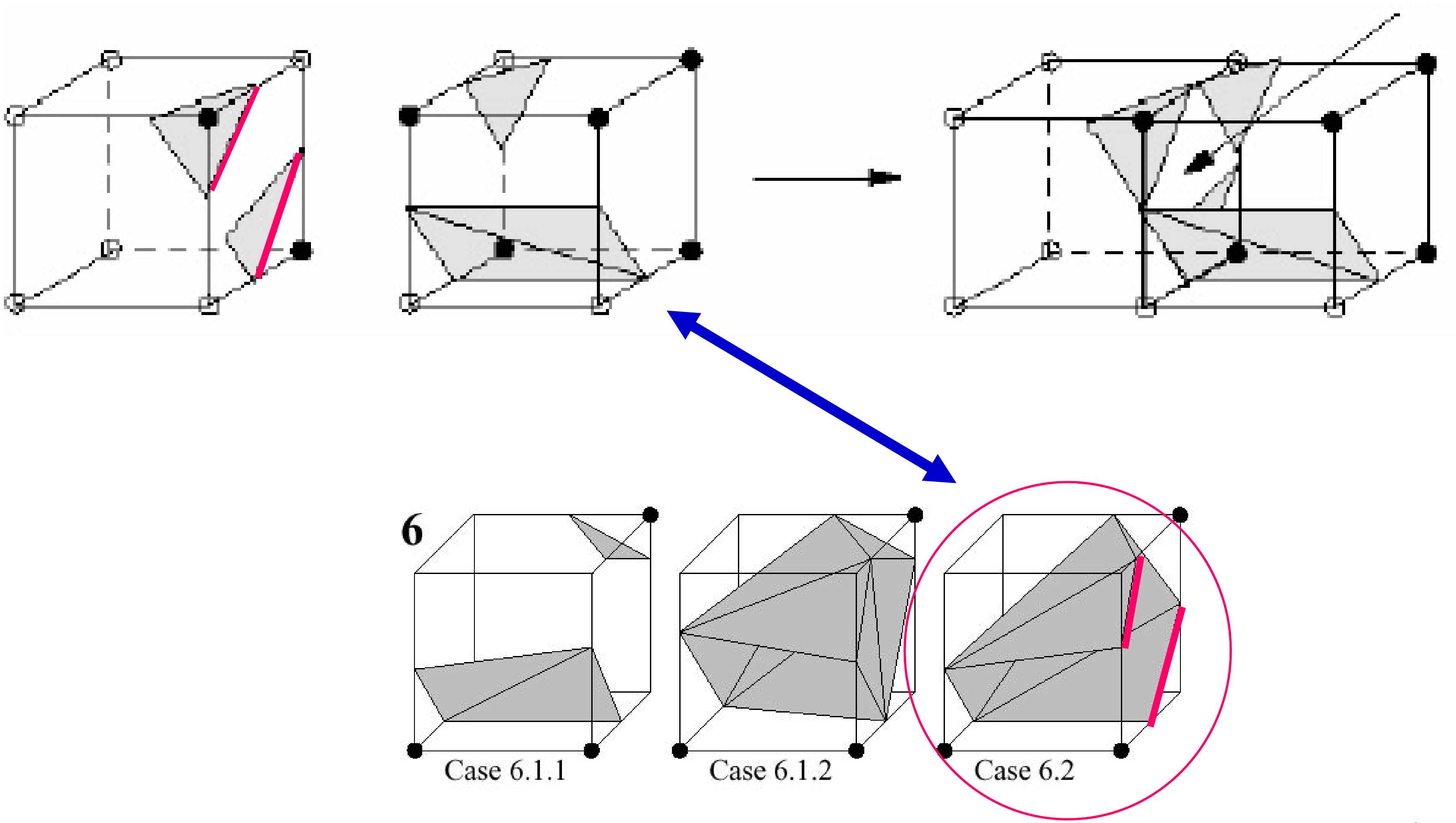
Resolve ambiguities III

- Marching cubes 33 by Evgeni Chernyaev



Case 8 Case 9 Case 10.1.1 Case 10.1.2
Case 10.2 Case 11 Case 12.1.1 Case 12.1.2
Case 12.2 Case 13.1 Case 13.2 Case 13.3
Case 13.4 Case 13.5.1 Case 13.5.2 Case 14

Resolve ambiguities



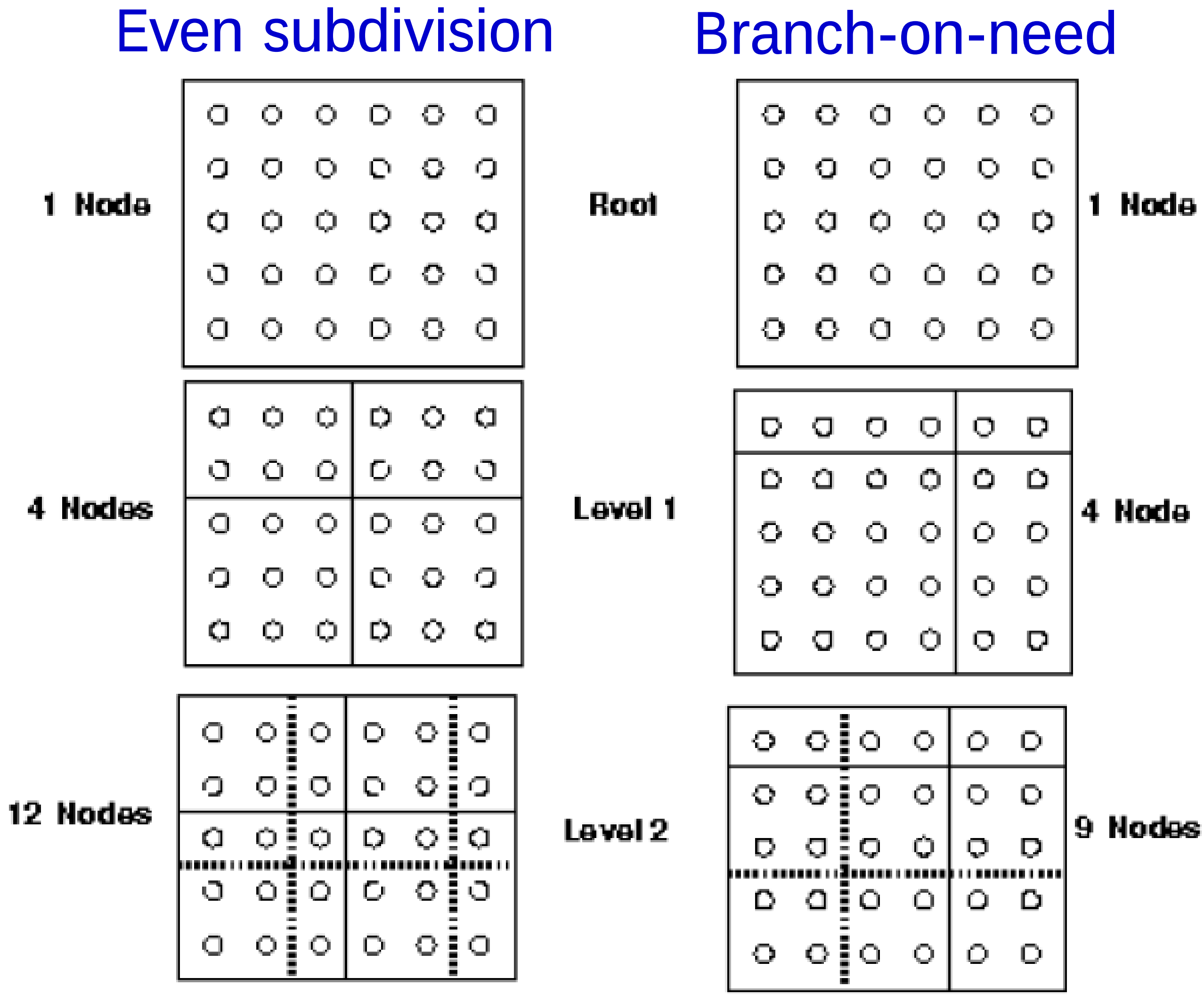
Drawback of MC

- Marches all cells, including empty cells.
  - 30-70% of isosurface generation time was spent in examining empty cells.
- Methods
  - Use hierarchical data structure to summarize volume information
  - Can prevent useless traversal of regions of no interest.
  - Octree approach

Octrees

- Full octree
  - Each node has exactly 8 children.
  - Even-subdivision
- Adaptive Even-subdivision
- Branch-on-need octree :Delay subdivision until absolutely necessary

Octrees


Even subdivision
Branch-on-need
1 Node
Root
1 Node
4 Nodes
Level 1
4 Node
12 Nodes
Level 2
9 Nodes

Model Simplification

Motivation

- Digital elevation data > 10^5 polygons
- 3D digitizers > 10^5 polygons
- Medical imaging > 10^6 polygons

  e.g., 256x256x128 ➔ ~820,000 triangles

→

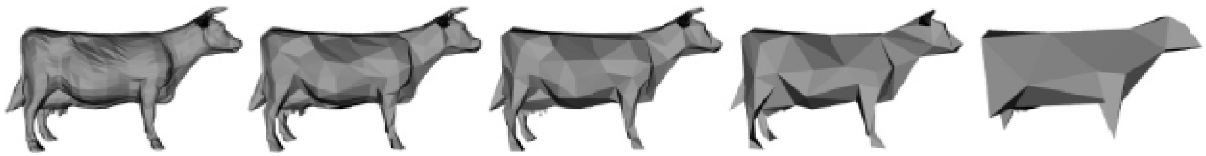
- *Slower rendering speeds*
- *Large memory requirements*
- *More expensive analysis*

Mesh simplification algorithm - Decimation

Full Resolution
(569K Gouraud shaded triangles)

75% decimated
(142K Gouraud shaded triangles)

Mesh simplification algorithm - Decimation



- To reduce the total # of triangles in a triangle mesh while preserving original topology

Classify each point in the mesh

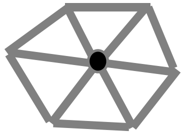
↓

Measure decimation criterion

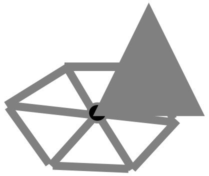
↓

Delete the point and triangulate the resulting hole

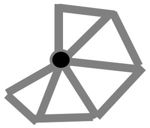
Characterize the vertices



simple : surrounded by a complete cycle of triangles

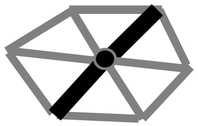


complex : incident triangles are not in the cycle



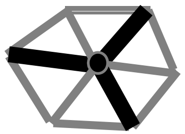
boundary: on the mesh boundary

- A simple vertex may be interior edge vertex or corner vertex



interior edge vertex: two feature edges

Feature edge: dihedral angle > a specified angle



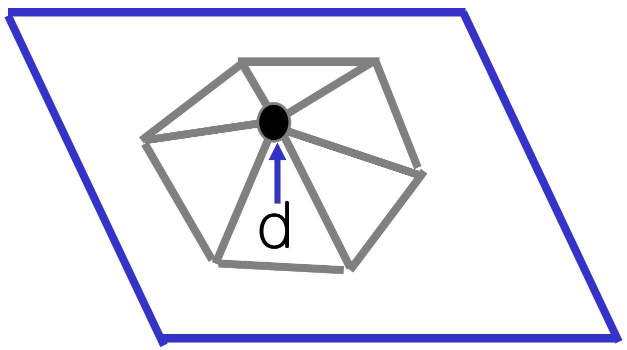
corner vertex : three or more feature edges use the vertex

Evaluate Decimation Criterion

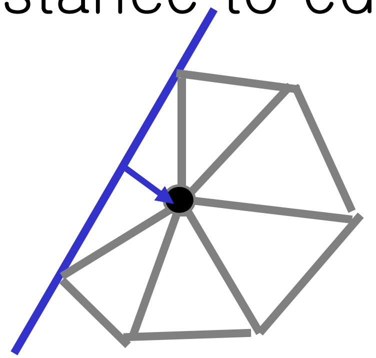
Complex and corner vertices are not deleted;
apply error measure to all simple vertices

[Simple distance based measure]

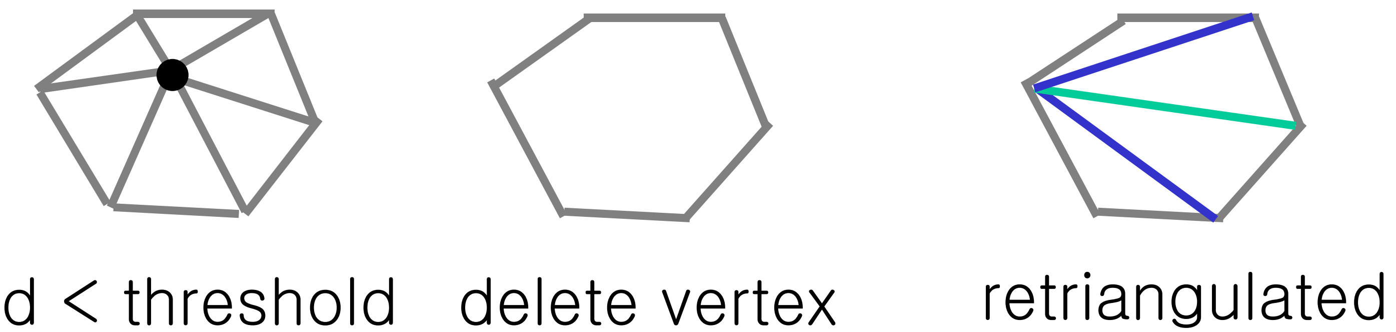
Distance to average plane



Distance to edge

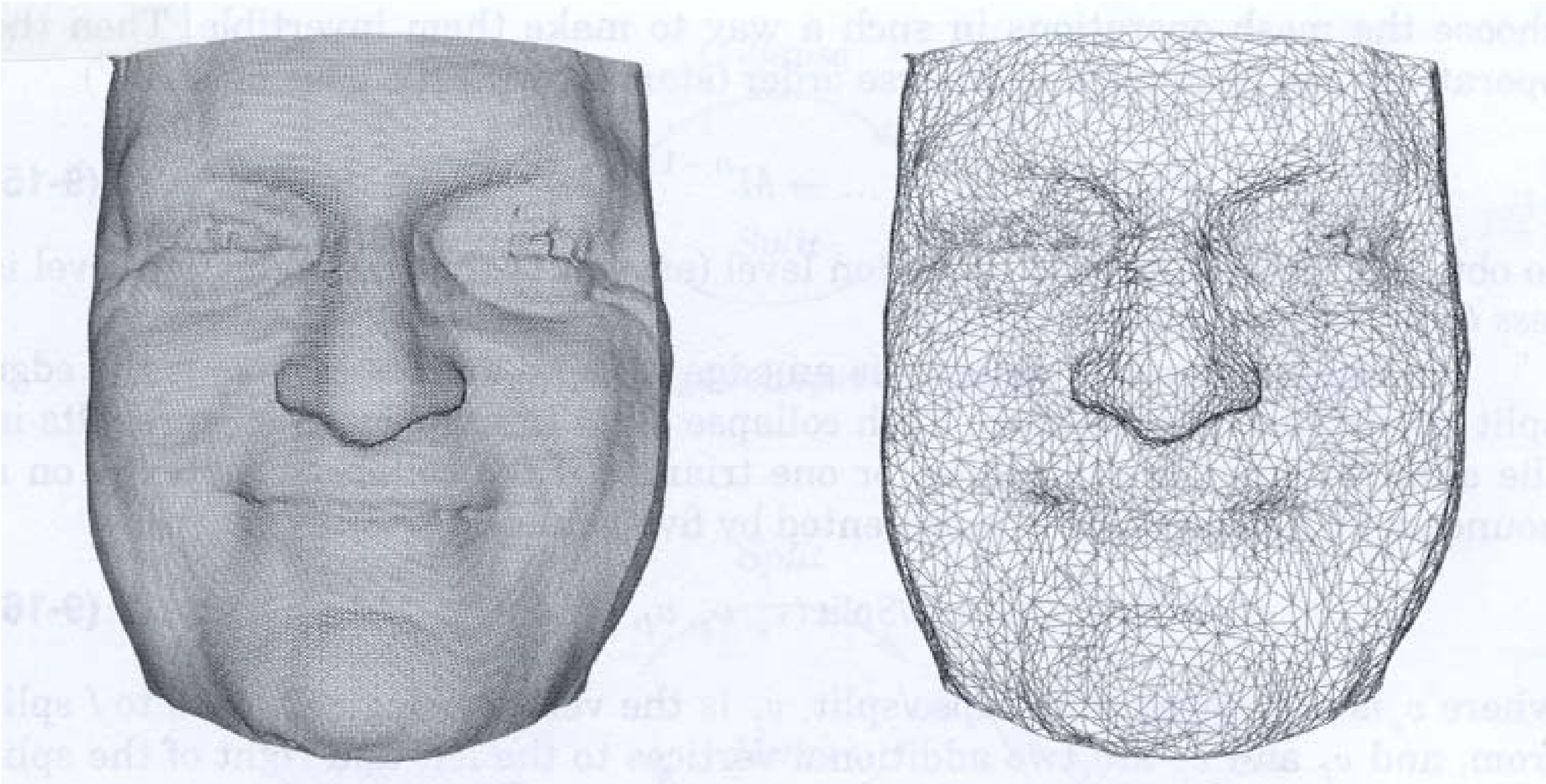


Triangulating Holes



- If the distance *d < a user specified distance*, the vertex is removed, resulting a hole
- Re-triangulation: divide the loop so that two sub-loops have the similar vertices
- Compression rate : upto 100:1

Decimation Example

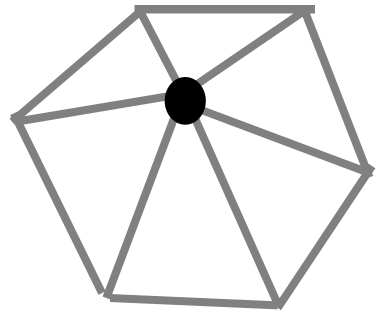


Characteristics of Decimation Method

- Local-topology preserving.
- Preserving sharp edges/corners.
- Vertices of the simplified model are original.
  - Good for maintain the color, normal, texture
- Maintain the shape based on the geometric criterion. The distance error, however, is derived using the current simplified model and the model of the previous iteration. (Cannot achieve error-bound guarantee)

Smoothing

- Improve the appearance of a mesh by adjusting the coordinates of vertices
- Modify the geometry not topology
- A simple Laplacian smoothing



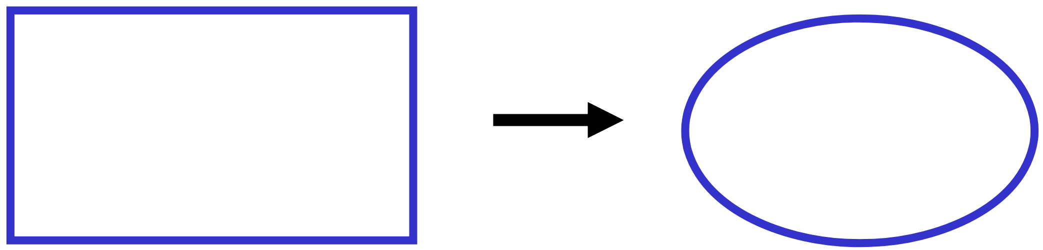
$$\vec{x}_{i+1} = \vec{x}_i + \lambda \vec{V}_{ij} = \vec{x}_i + \lambda \sum (\vec{x}_j - \vec{x}_i)$$

λ : weight $\qquad \forall j : 0 \leq j < n$

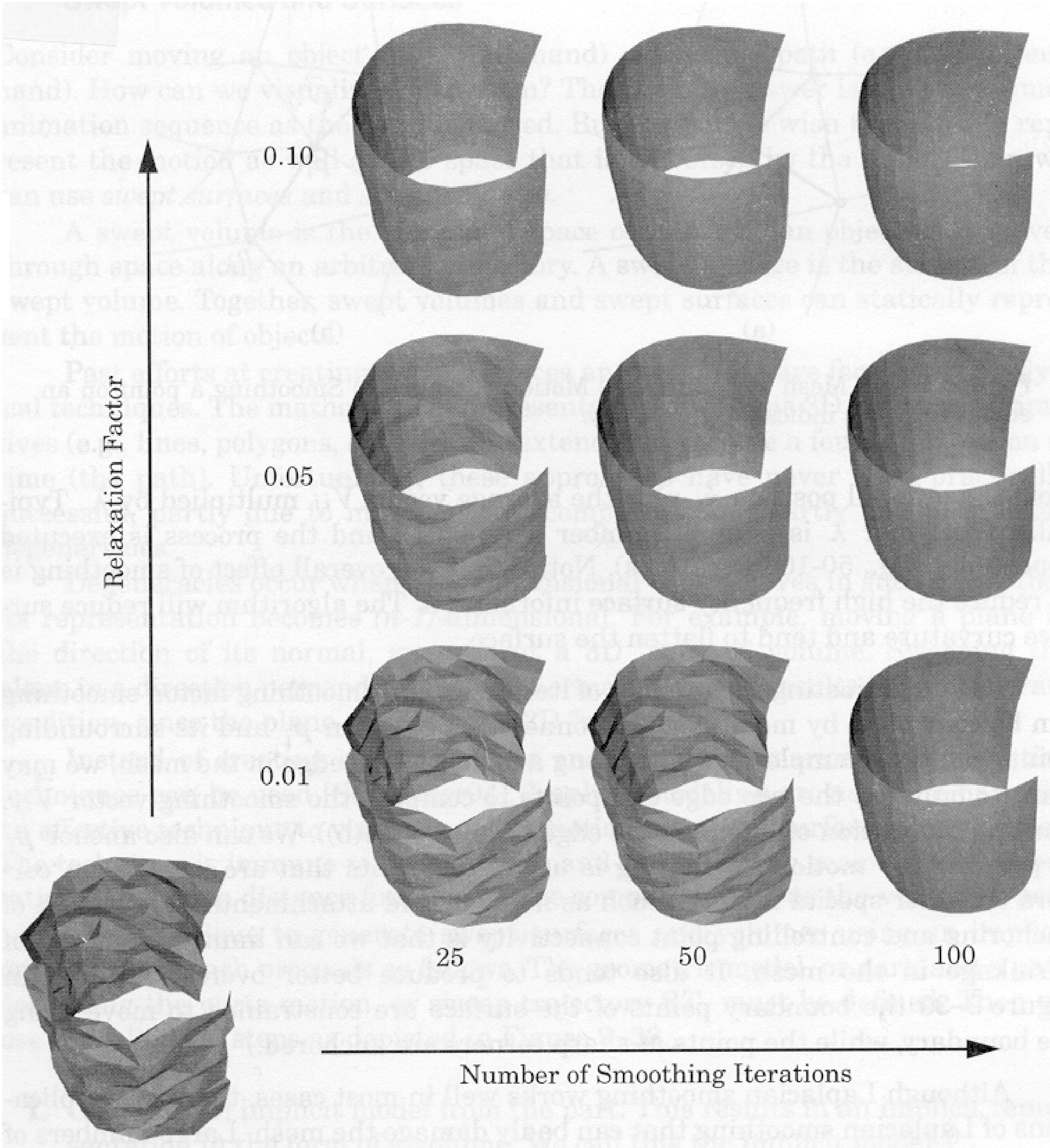
Smoothing

Large number of iteration

→ shrinkage and surface distortion



Relaxation factor limits motion


Relaxation Factor
0.10
0.05
0.01
25
50
100
Number of Smoothing Iterations

Dividing Cubes

Create "Surface Points“ with normals instead of triangles (no topology)

[ALGORITHM]

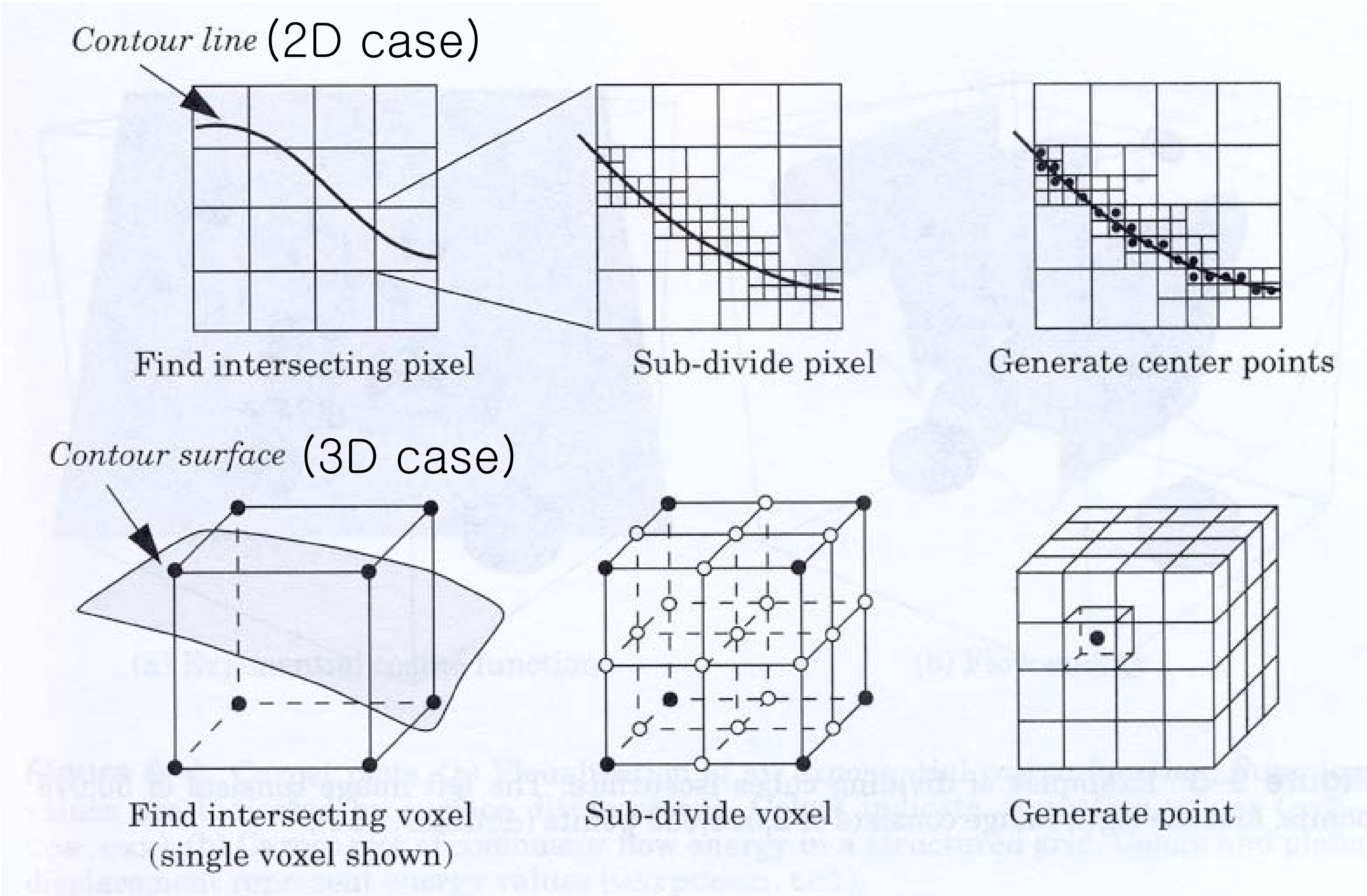
1. Create a cube

   consider a cube defined by eight data values,
   four from slice k and four from slice k+1

2. Classify each cube

   - interior, exterior, or intersecting(surface) cube

Dividing Cube Algorithm

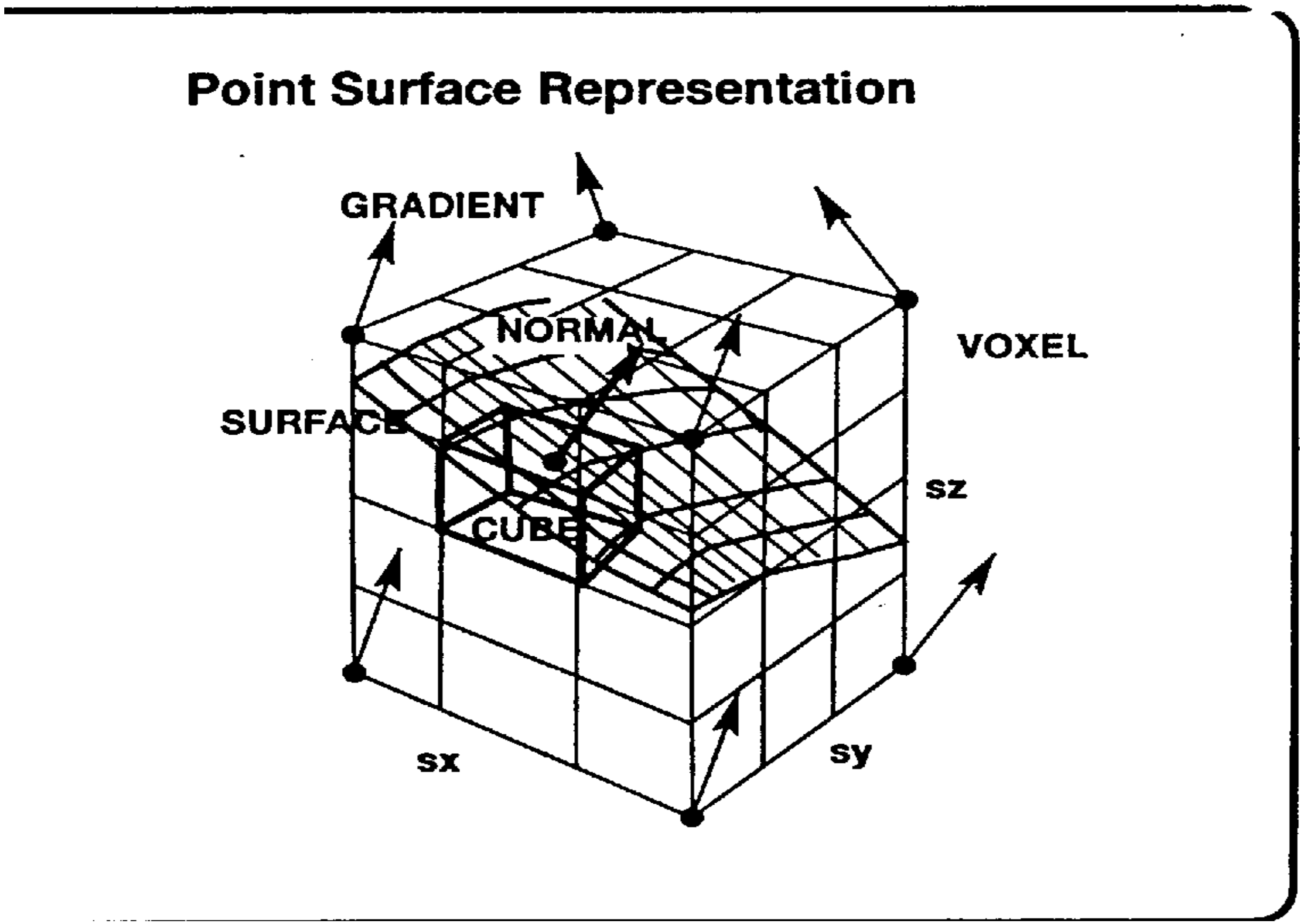


Dividing Cube Algorithm

3. Subdivide intersecting cubes to image resolution
   - In x,y, and z, create small cubes that correspond to final image resolution
   - Calculate intensities for new cubes using tri-linear interpolation
   - For each small cube, test for intersecting cube

4. Calculate normals using central differences
5. Shade Cube

Dividing Cube Algorithm



Dividing Cubes Pro & Cons

- Rendering points is much faster than rendering polygons (but not always true)
- Useful for high-speed or large dataset allocations
  - the direct display of point values bypasses the need to scan-convert the polygons
- Point cloud rendering
- Randomly ordered points
- Magnification can result disconnected surface
- Good for clipping and merging data.