Computer Graphics 1

Ludwig-Maximilians-Universität München

Summer semester 2020

Prof. Dr.-Ing. Andreas Butz

lecture additions by Dr. Michael Krone, Univ. Stuttgart

https://commons.wikimedia.org/wiki/File:Stanford_bunny_qem.png

Sources

- This lecture was introduced by Michael Krone and is based on the slides of Filip Sadlo for the lecture „*Visualization in Science an Engineering*“
- Course slides make use of selective contributions from
  - Thomas Ertl
  - Daniel Weiskopf
  - Carsten Dachsbacher
  - Oliver Deussen
  - Rüdiger Westermann
  - Stefan Gumbold
  - Dirk Bartz
  - Torsten Möller
  - Ronald Peikert

Chapter 10 – Volume Rendering & Scalar Field Visualization

- Basic strategies
  - Function plots and height fields
  - Isolines
  - Color coding
- Volume data
  - Overview of volume visualization approaches
  - Slicing
  - Indirect volume visualization
  - Direct volume rendering
  - Classification and segmentation

Basic Strategies

- Visualization of 1D, 2D, or 3D scalar fields
  - 1D scalar field: $\Omega \subset \mathbb{R} \to \mathbb{R}$
  - 2D scalar field: $\Omega \subset \mathbb{R}^2 \to \mathbb{R}$
  - 3D scalar field: $\Omega \subset \mathbb{R}^3 \to \mathbb{R}$ → Volume visualization

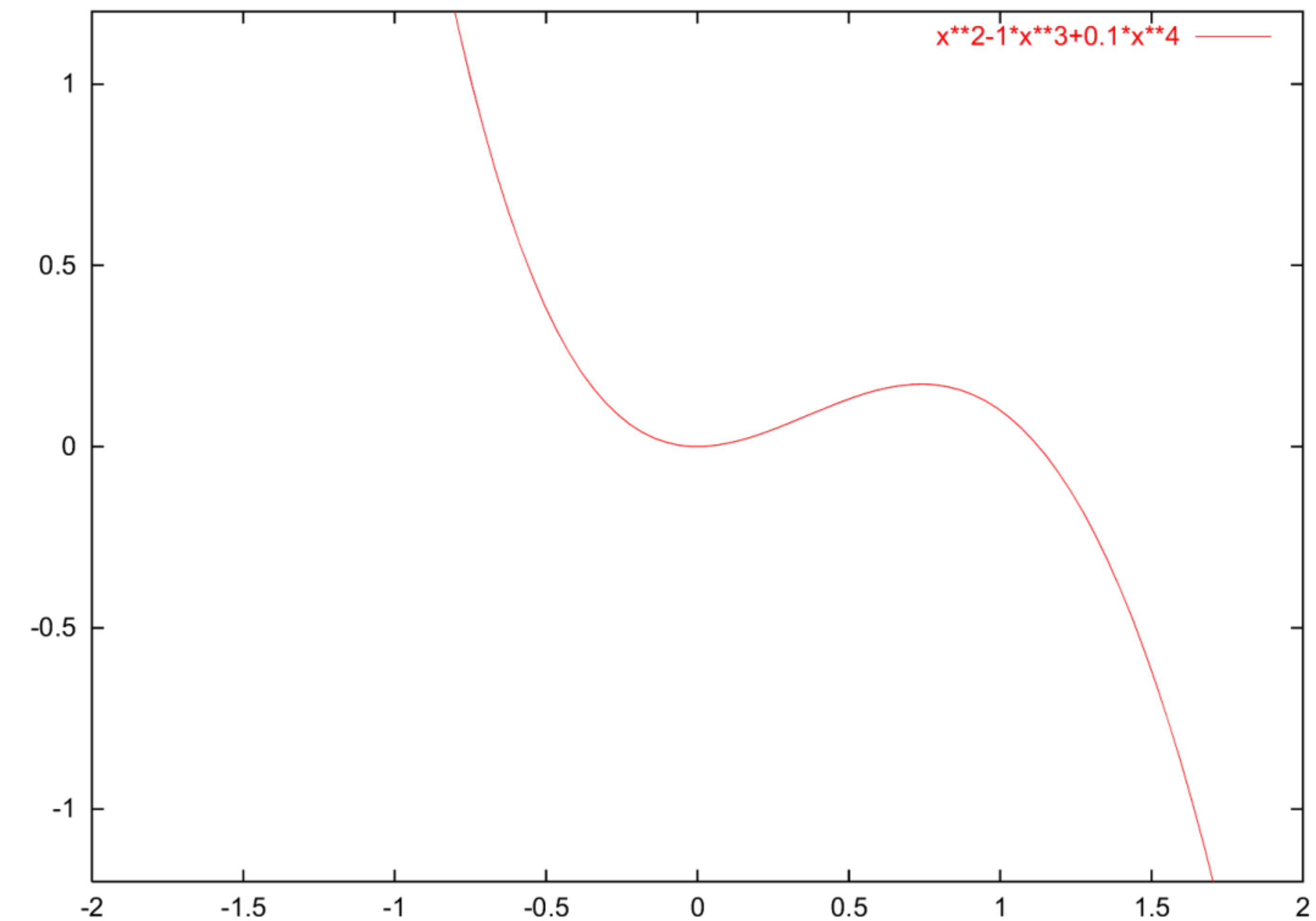
Basic Strategies

- Mapping to geometry
  - Function plots
  - Height fields
  - Isolines and isosurfaces
- Color coding
- Specific techniques for 3D data
  - Indirect volume visualization
  - Direct volume visualization

→ Visualization method depends heavily on dimensionality of domain

Function Plots and Height Fields

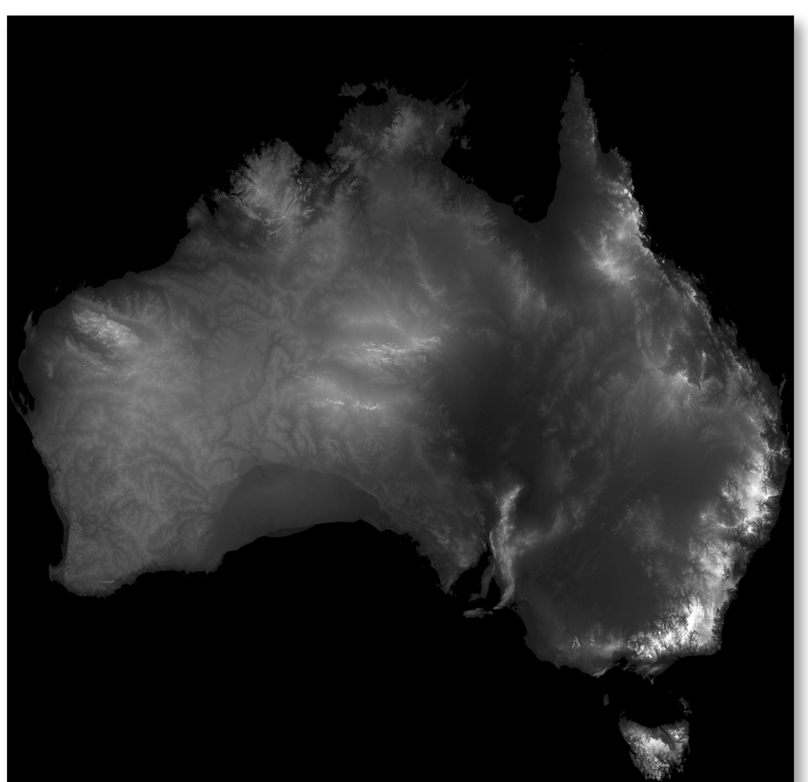
- Function plot for a 1D scalar field
  - Points $\{(s, f(s)) \mid s \in \mathbb{R}\}$
  - 1D manifold: line
  - Error bars possible



Gnuplot example

Function Plots and Height Fields

- Function plot for a 2D scalar field
  - Points $\{(s,t,f(s,t) \mid (s,t) \in \mathbb{R}^2\}$
  - 2D manifold: surface
- Surface representations
  - Wireframe
  - Hidden lines
  - Shaded surface

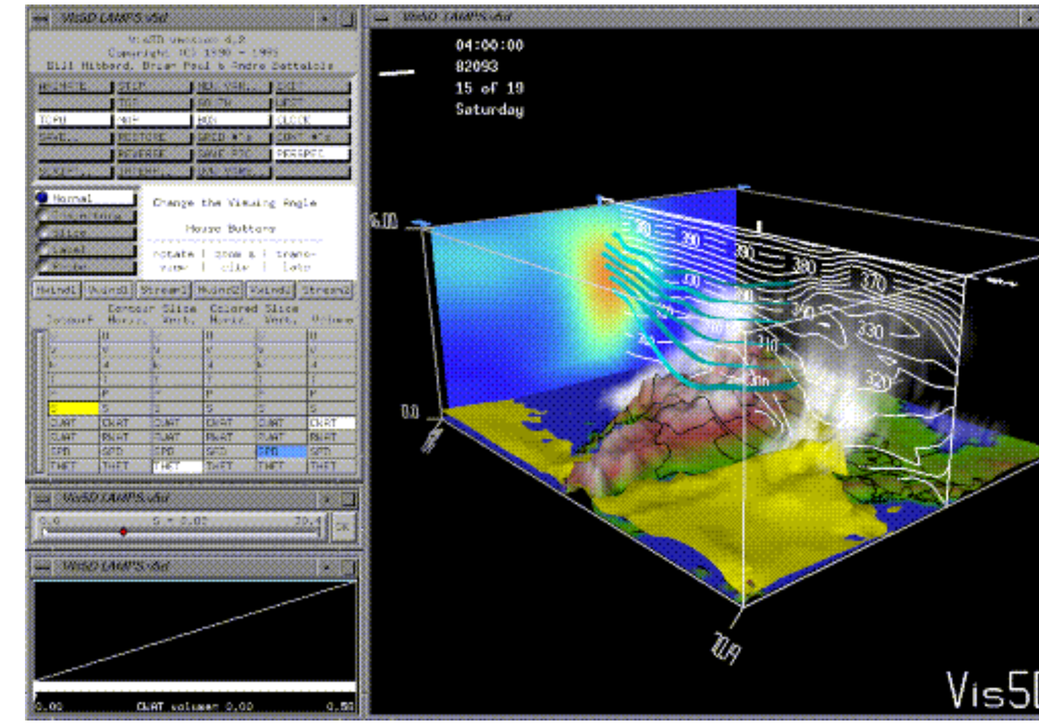


Isolines

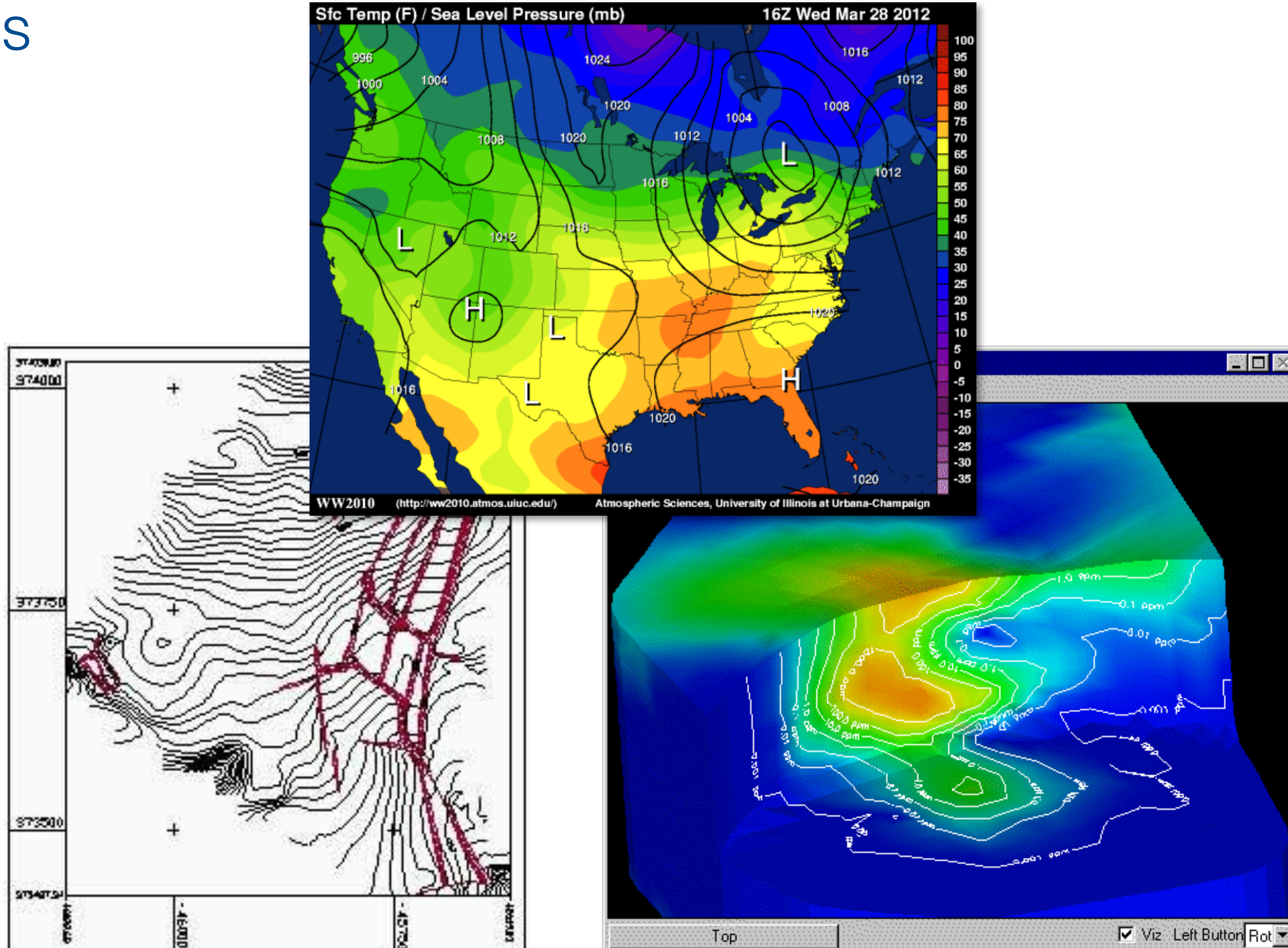
- Visualization of 2D scalar fields
- Given a scalar function $f : \Omega \rightarrow \mathbb{R}$ and a scalar value (isovalue) $c \in \mathbb{R}$
- Isoline consists of points

$$\{(x,y) \mid f(x,y) = c\}$$

- If $f()$ is differentiable and $grad(f) \neq 0$, then isolines are curves
- Contour lines



Isolines



Sfc Temp (F) / Sea Level Pressure (mb)
16Z Wed Mar 28 2012
WW2010
(http://ww2010.atmos.uiuc.edu/)
Atmospheric Sciences, University of Illinois at Urbana-Champaign
Top
Viz
Left Button
Rot

Isolines: Pixel-by-Pixel Contouring

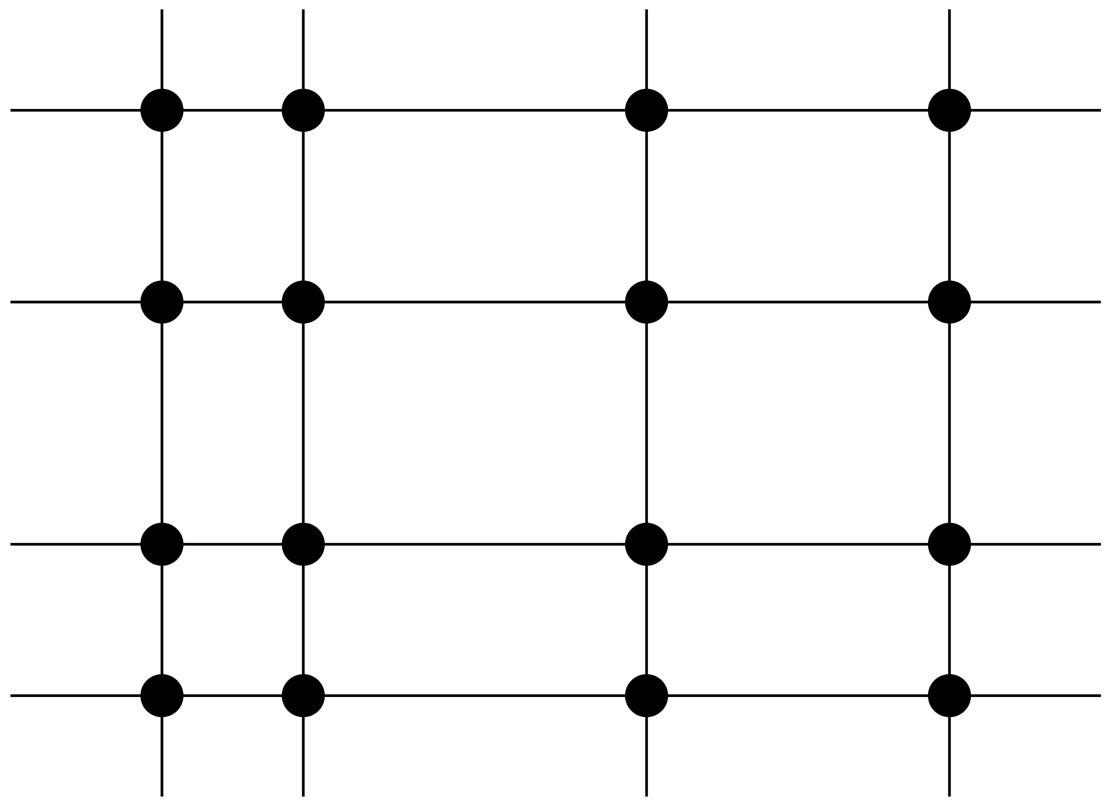
- Straightforward approach:
  scanning all pixels for equivalence with isovalue
- Input
  - $f : (1,...,x_{max}) \times (1,...,y_{max}) \rightarrow \mathbf{R}$
  - Isovalues $I_1,..., I_n$ and isocolors $c_1,...,c_n$
- Algorithm

```
for all (x,y) ∈ (1,...,x_max) x (1,...,y_max) do
        for all k∈ { 1,...,n } do
                if |f(x,y)-I_k| < ε then
                        draw(x,y,c_k)
```

- Problem: Isoline can be missed if the gradient of *f()* is too large (despite range ε)

Isolines: Marching Squares

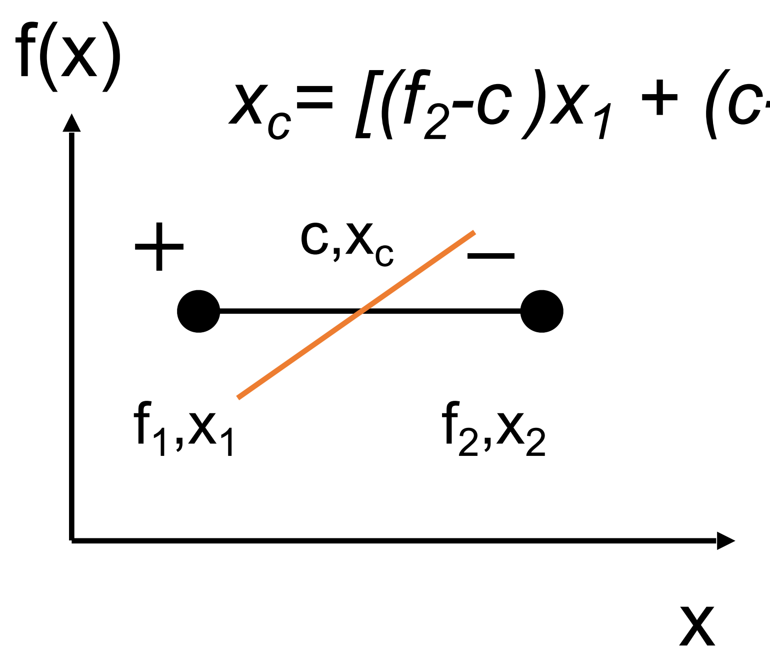
- Representation of the scalar function on a uniform or rectilinear grid
- Scalar values are given at each vertex $f \leftrightarrow f_{ij}$
- Take into account the interpolation within cells
- Consider cells independently of each other

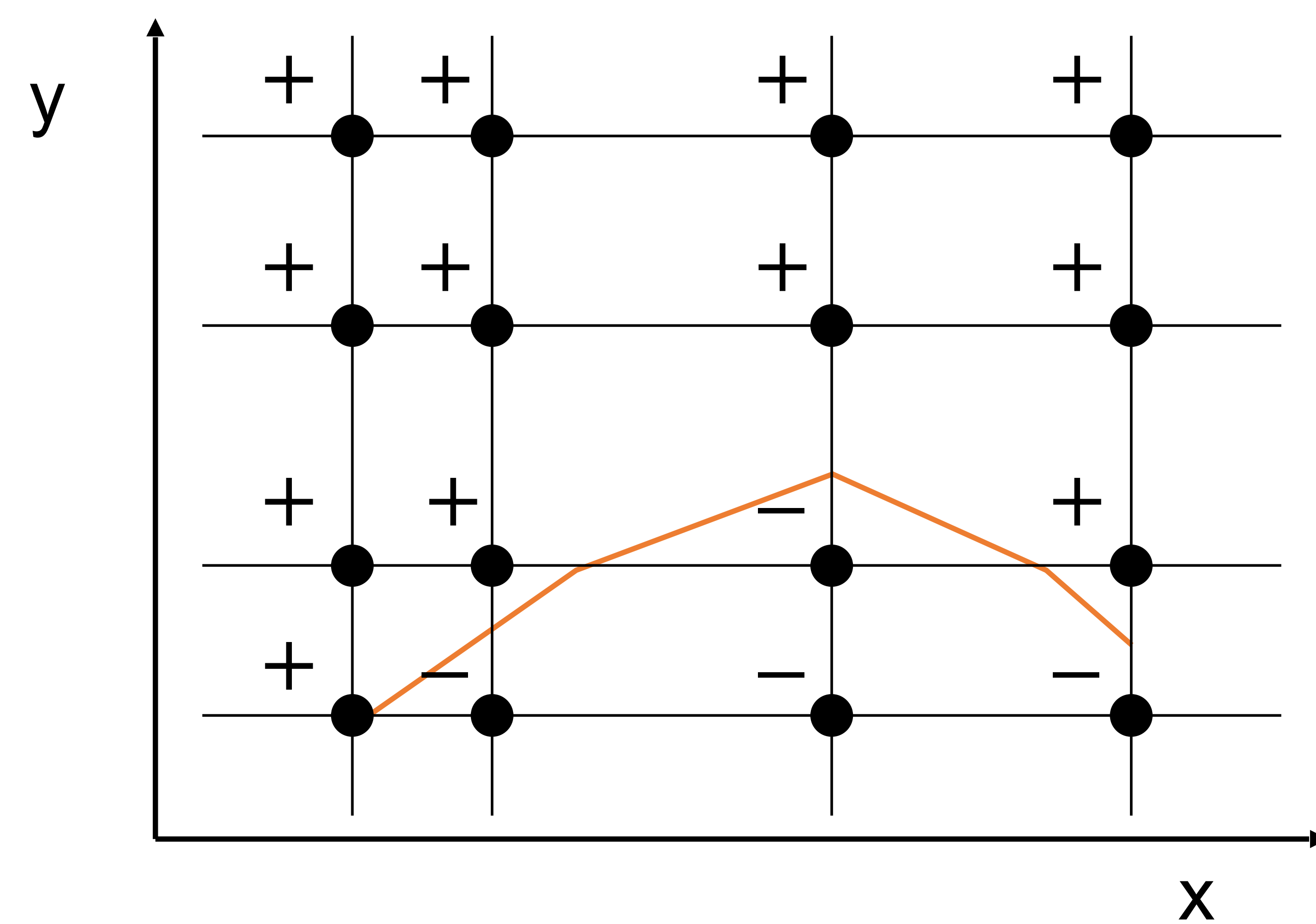


Isolines: Marching Squares

- Which cells will be intersected ?
  - Initially mark all vertices by + or – , depending on the conditions $f_{ij} \geq c$, $f_{ij} < c$
- No isoline passes through cells (=rectangles) which have the same sign at all four vertices
  - So we only have to determine the edges with different signs
  - And find the intersection point by linear interpolation

$$x_c = [(f_2 - c)x_1 + (c - f_1)x_2] / (f_2 - f_1)$$



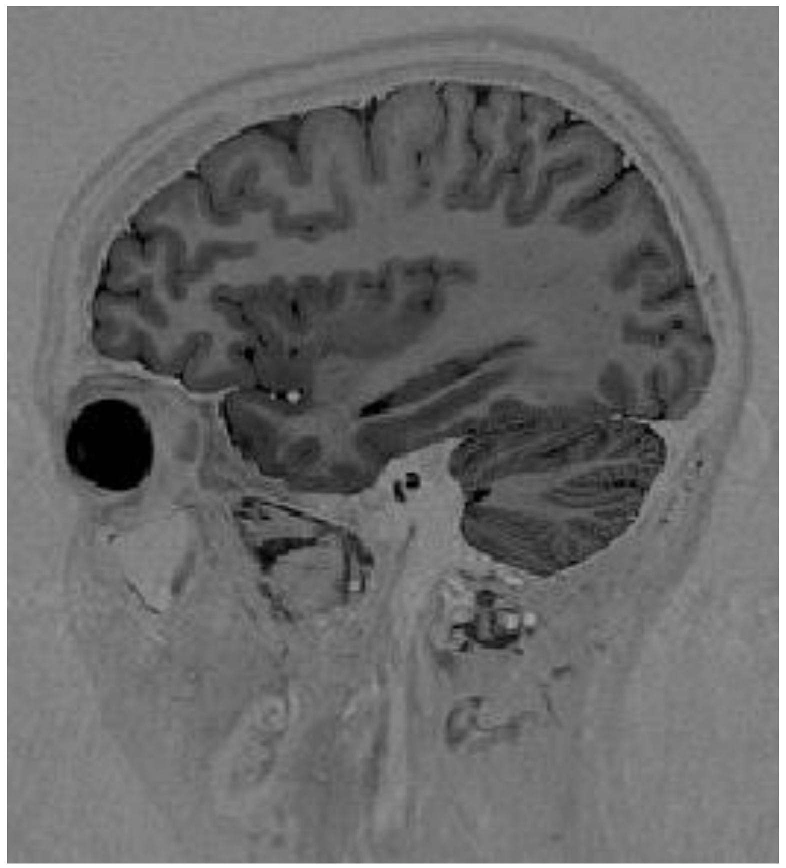


Color Coding

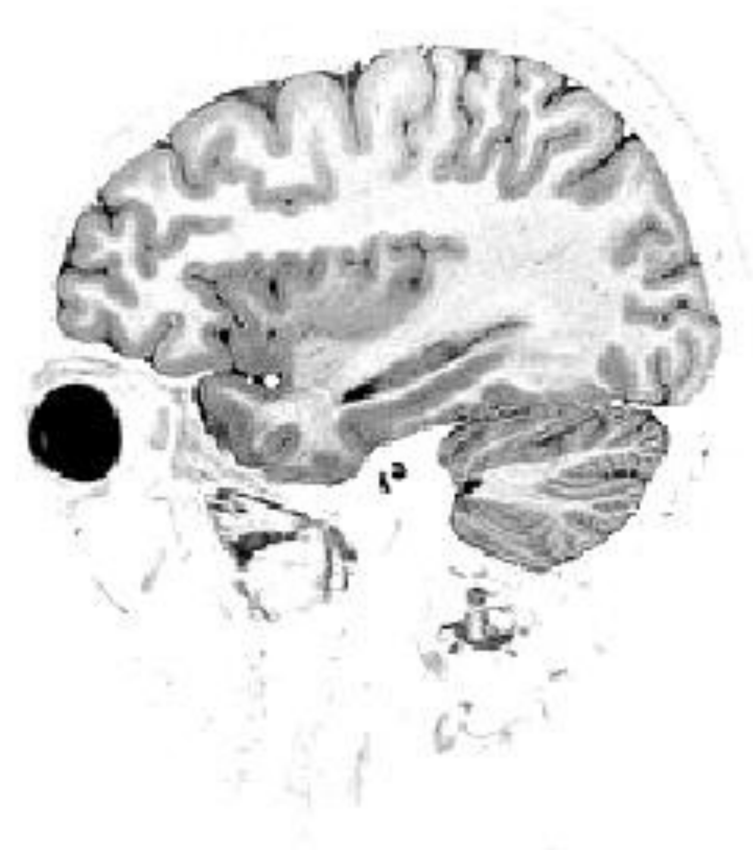
- Easy to apply to 1D and 2D scalar fields
  - Map color to each pixel on 1D or 2D image $\Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$

Color Coding

- Example: Medical images
  - Special color table to visualize the brain tissue
  - Special color table to visualize the bone structure



Original



Brain

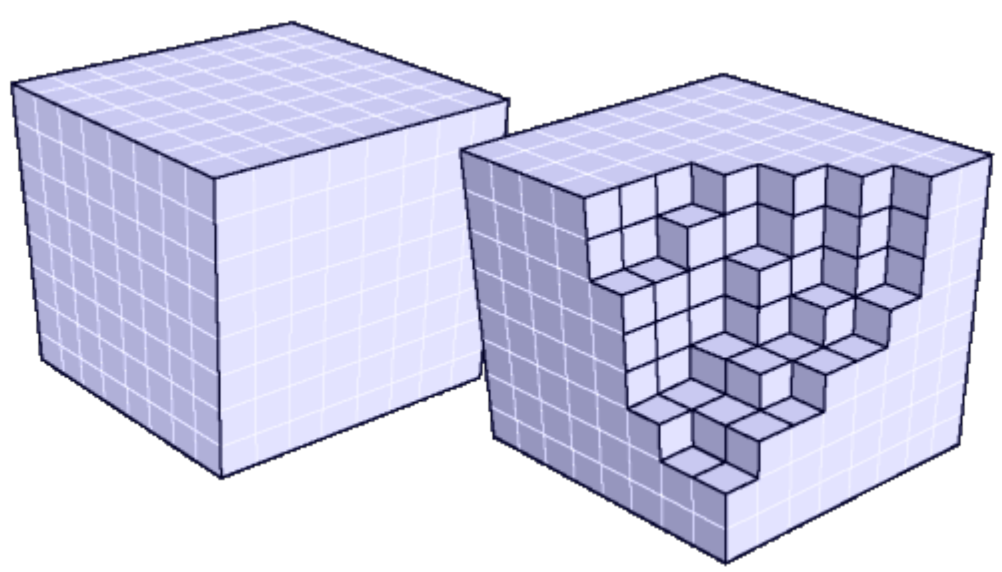
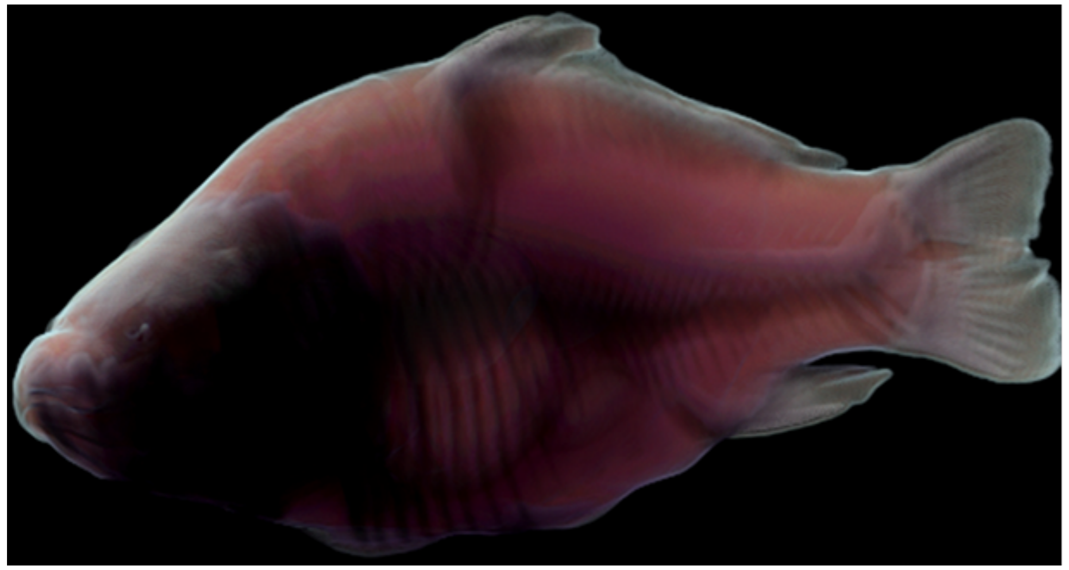
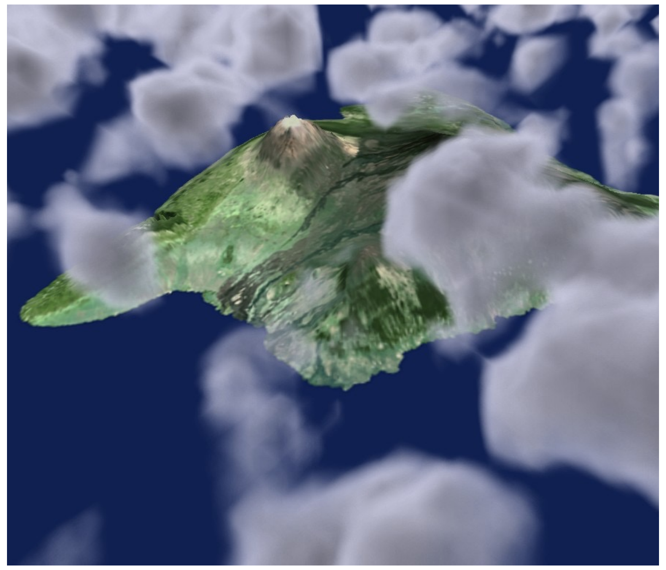
Tissue

Chapter 10 – Volume Rendering & Scalar Field Visualization

- Basic strategies
  - Function plots and height fields
  - Isolines
  - Color coding
- Volume data
  - Overview of volume visualization approaches
  - Slicing
  - Indirect volume visualization
  - Direct volume rendering
  - Classification and segmentation

Volume Data

- Simple case: regular, rectilinear 3D grid with cubic cells
  - Stores one or more values per grid cell
  - Grid cell = voxel (volume pixel)
- Data sources (examples)
  - Measurements, e.g., medical imaging (CT, MRT, 3D ultrasound…)
  - Simulation, e.g., fluid simulations (water, smoke, fog…)
  - Voxelization of 3D models, e.g., write closest distance to a surface to each voxel
  - Mathematical function

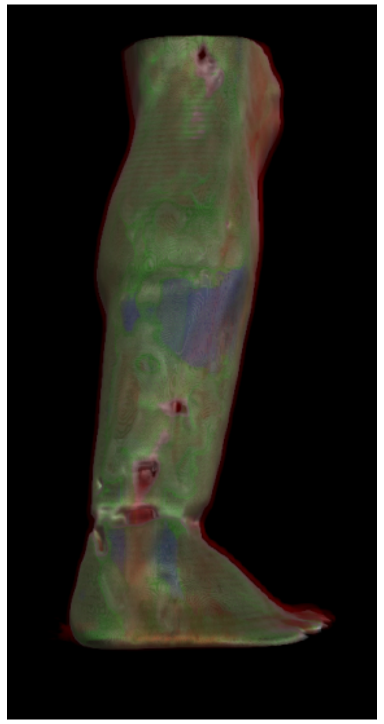
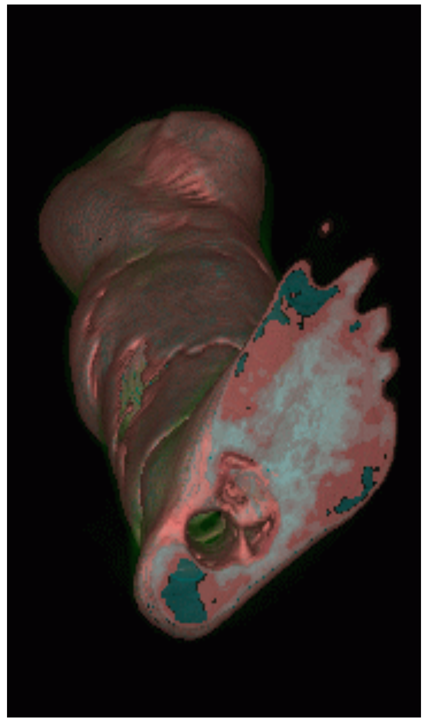
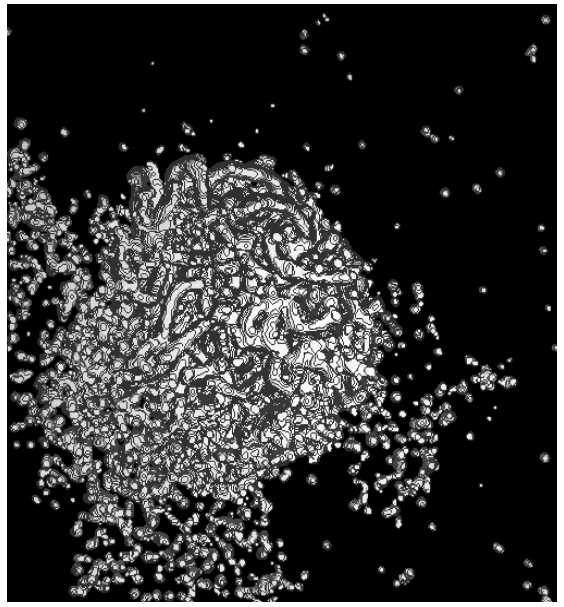
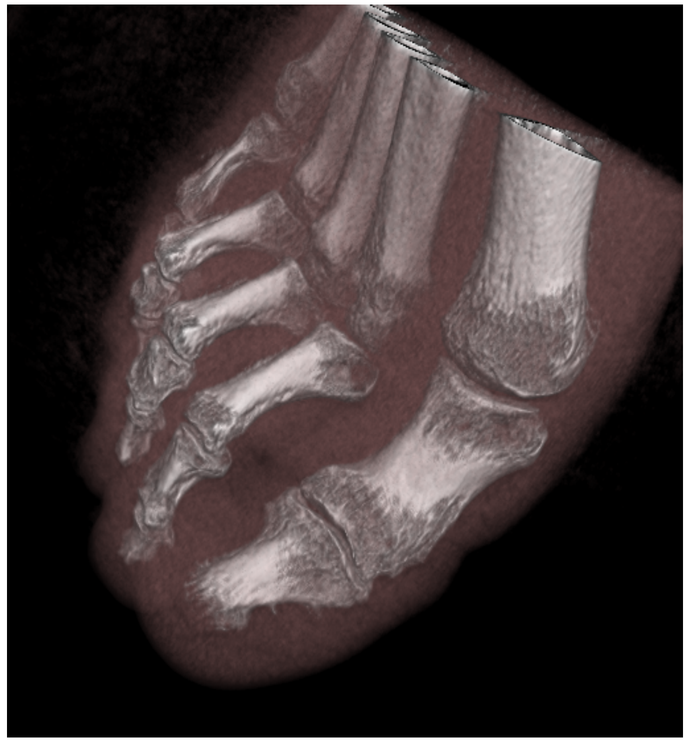




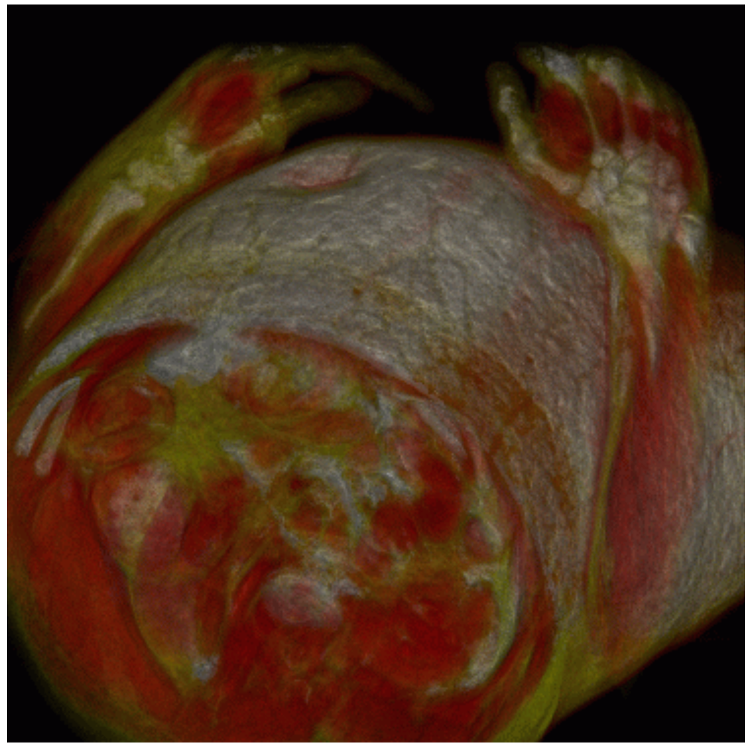
Volume Visualization

- Scalar volume data

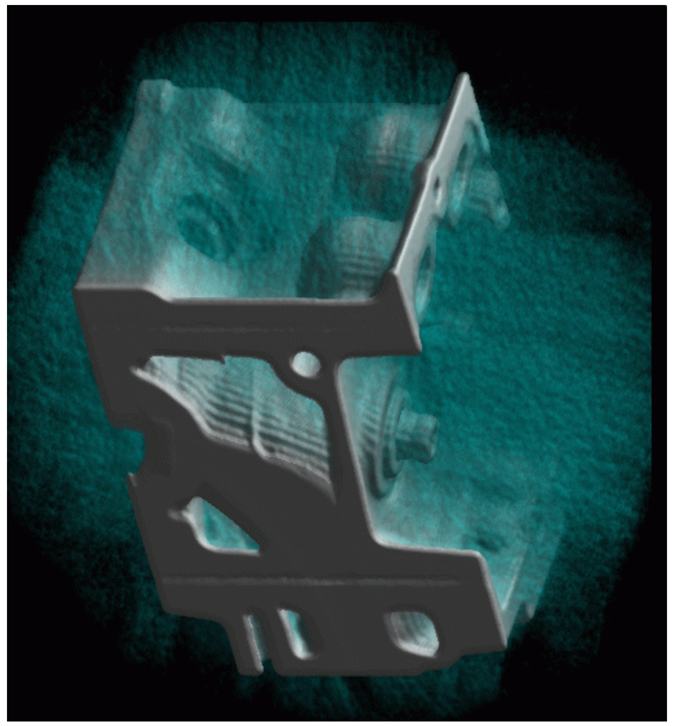
$$\Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}$$

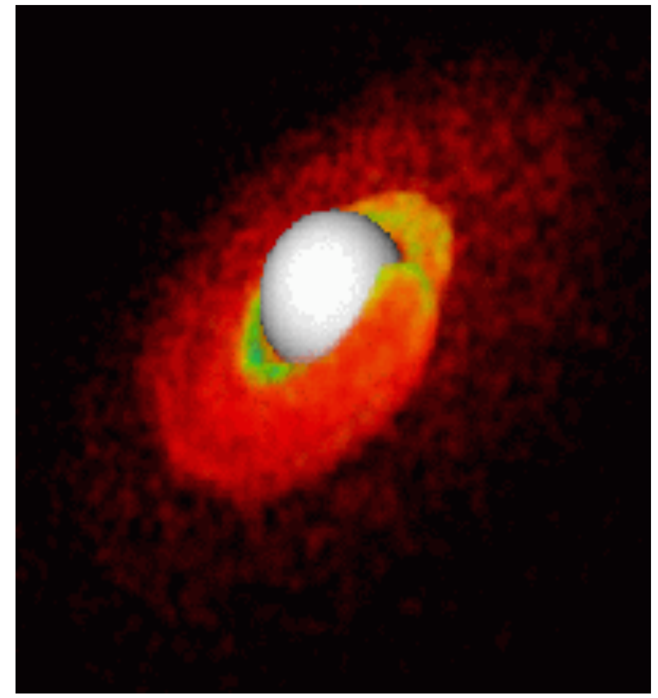
- Medical Applications:
  - CT, MRI, confocal microscopy, ultrasound, etc.

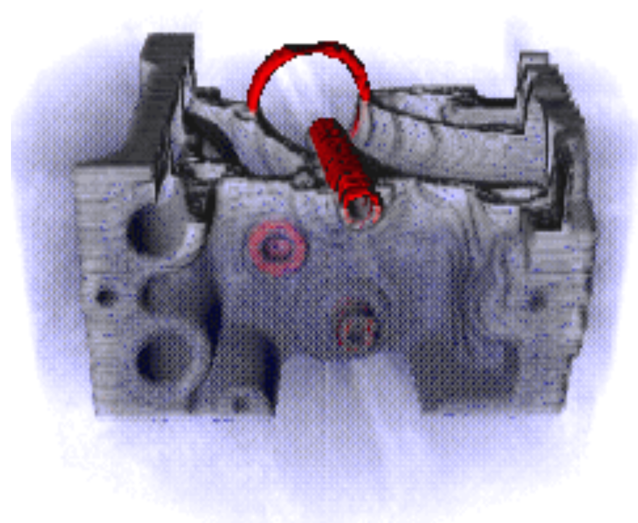





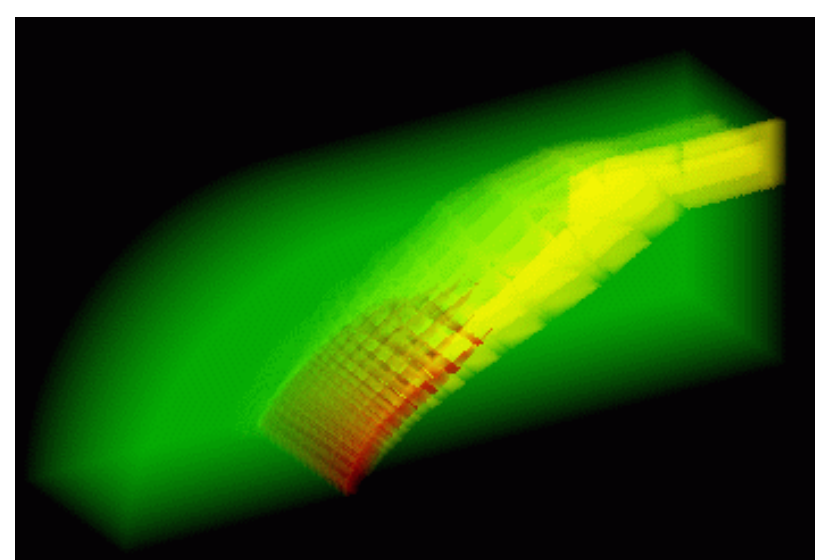


Volume Visualization

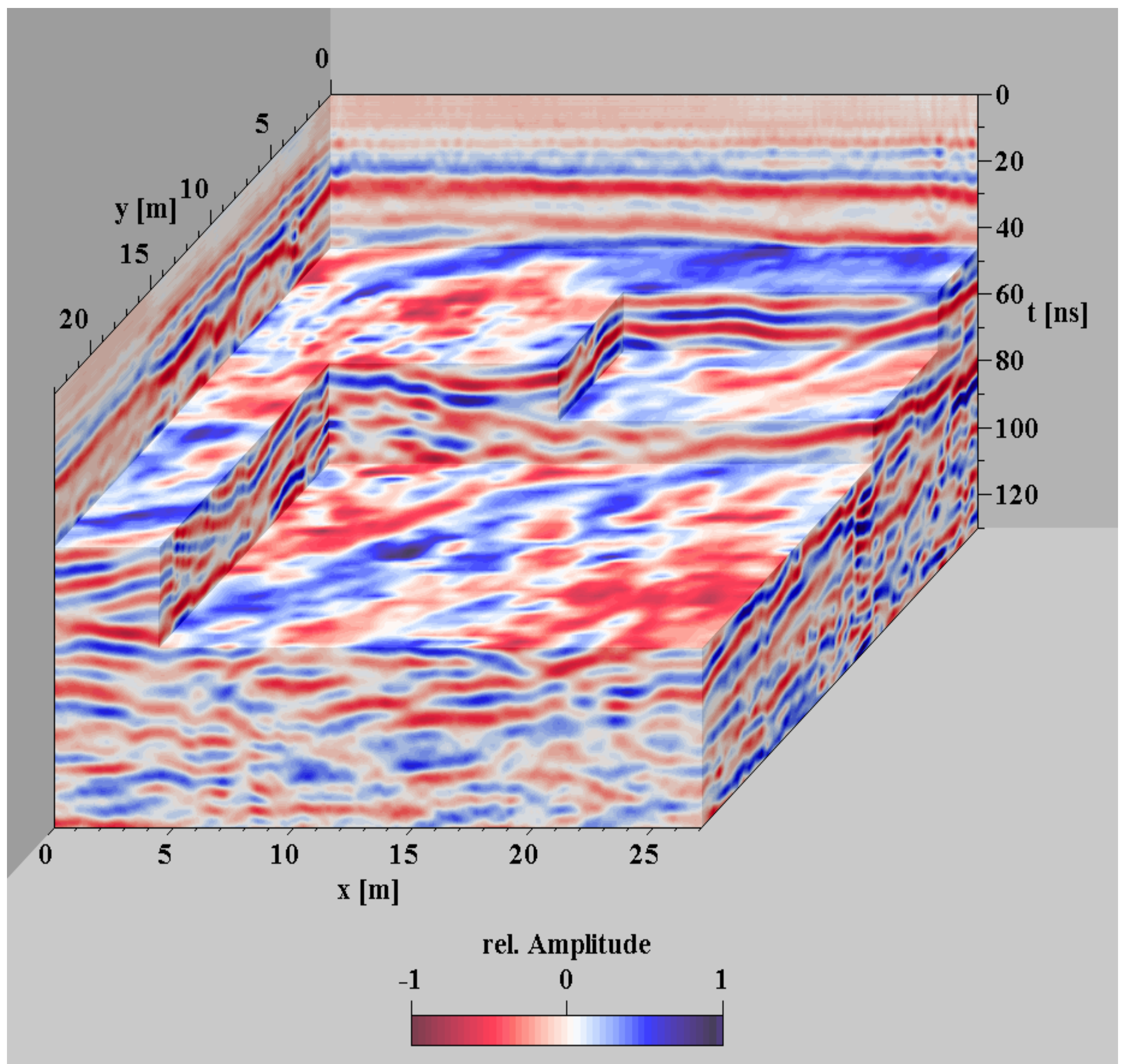


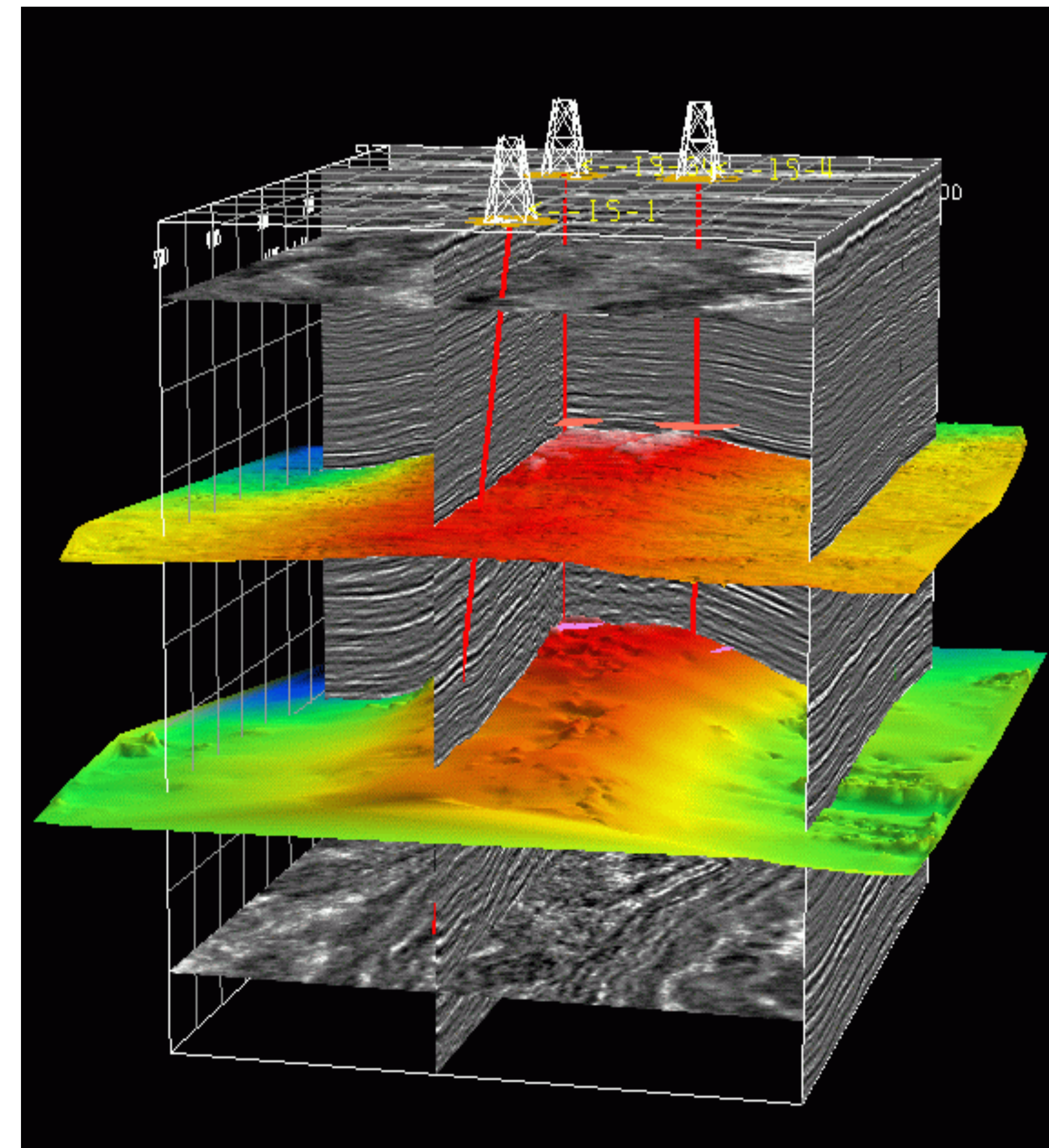






Volume Visualization



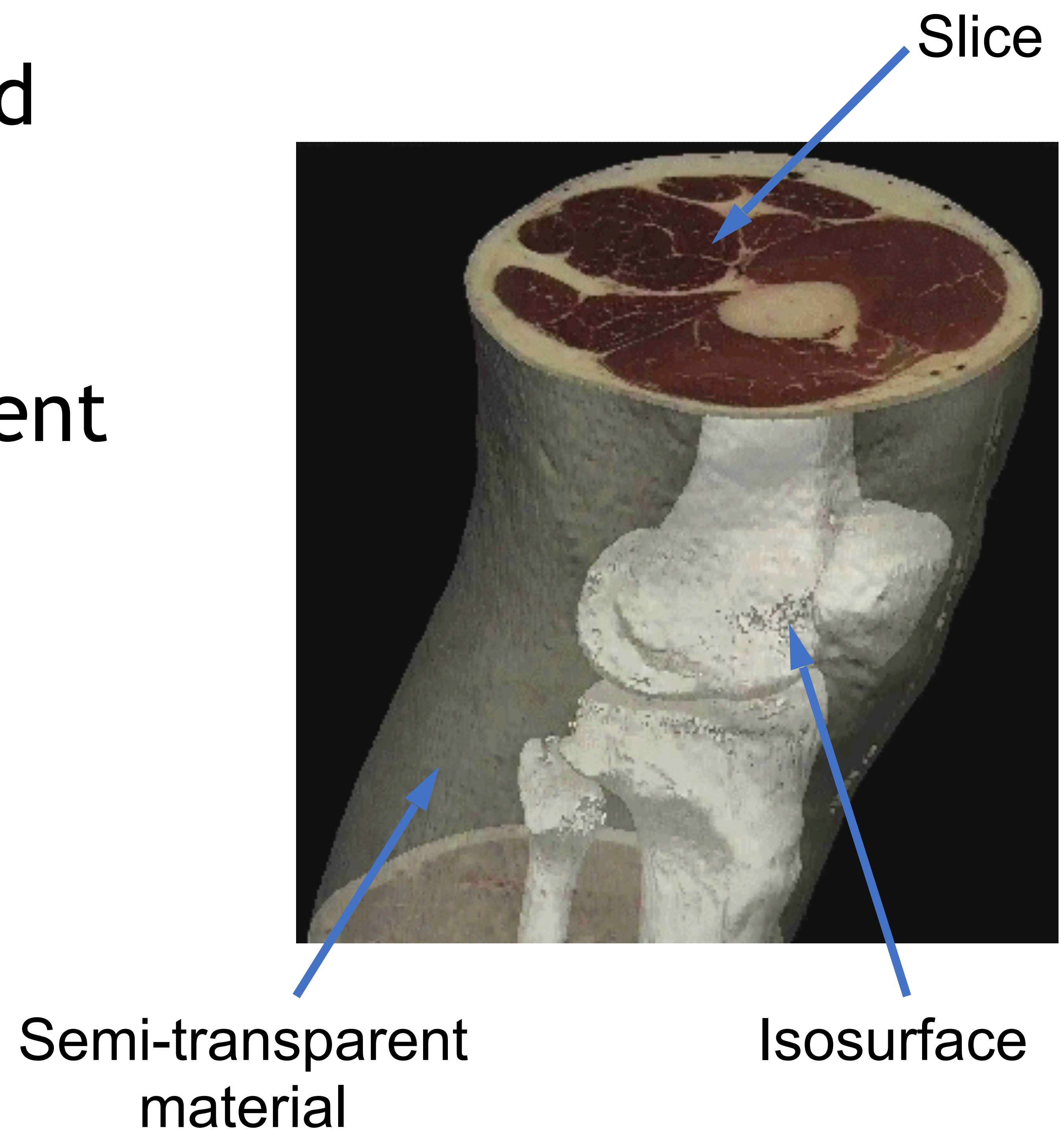


Volume Visualization Approaches

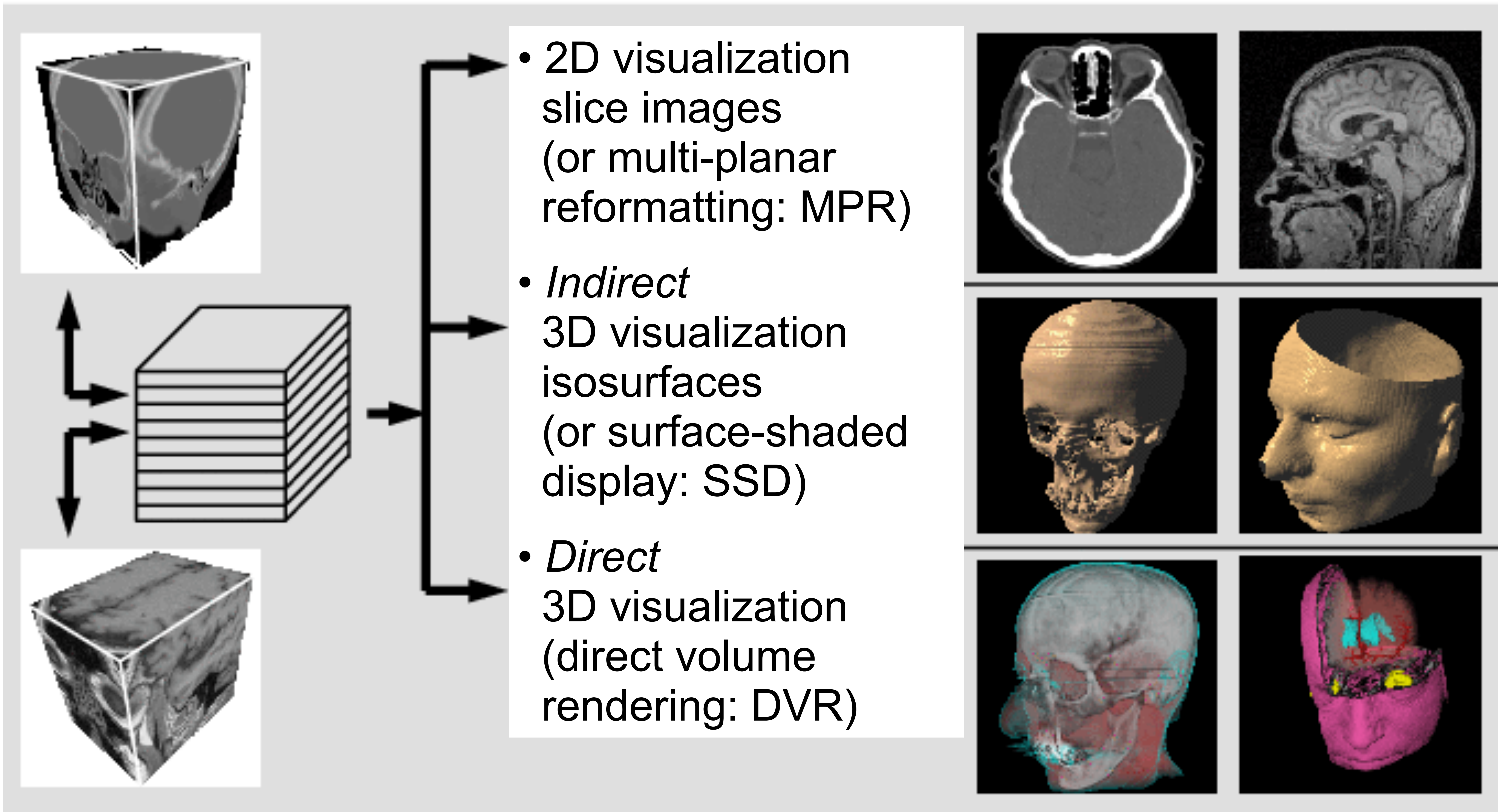
- Techniques for 2D scalar fields
  - Transform 3D data set to 2D
  - Then apply 2D methods
- Indirect volume rendering techniques (e.g. surface fitting)
  - Convert/reduce volume data to an intermediate representation (surface representation), which can be rendered with traditional techniques
- Direct volume rendering
  - Consider the data as a semi-transparent gel with physical properties and directly get a 3D representation of it

Volume Visualization Approaches

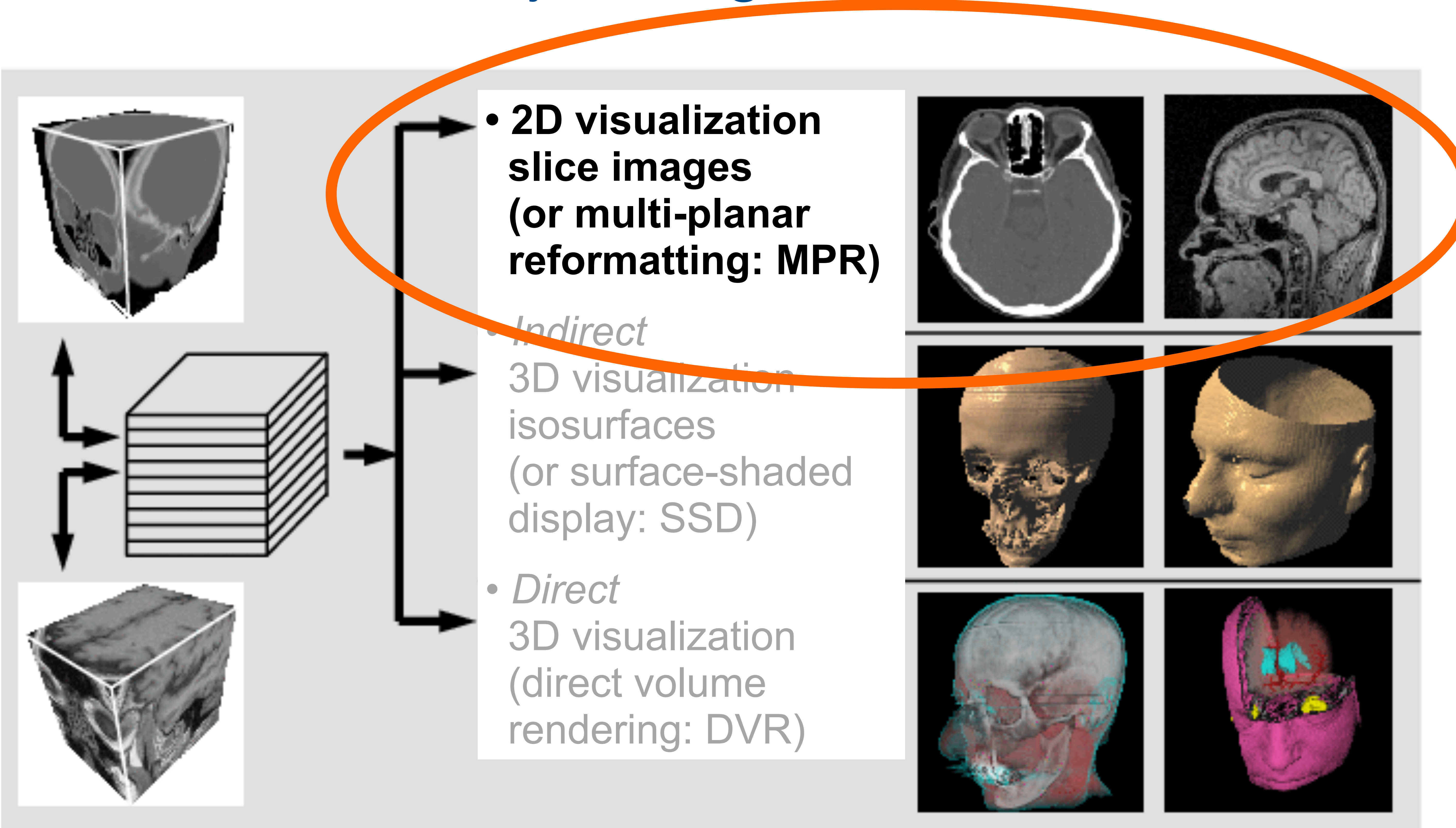
- Slicing:
  Display the volume data, mapped to colors, on a slice plane
- Isosurfacing:
  Generate opaque/semi-transparent surfaces
- Transparency effects:
  Volume material attenuates reflected or emitted light



Volume Visualization Approaches

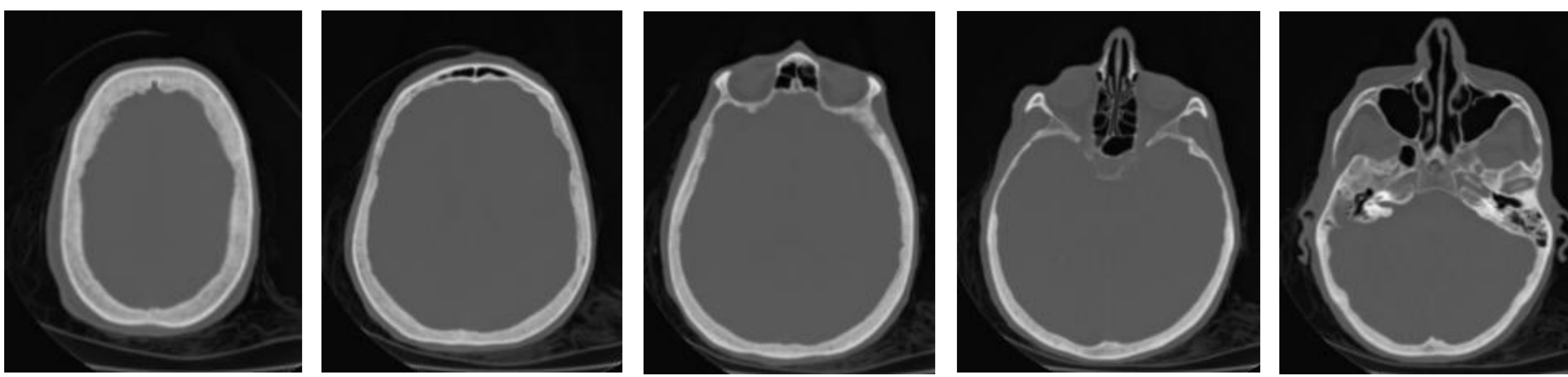


Volume Visualization by Slicing



Volume Visualization by Slicing

- 2D approach: Orthogonal slicing
  - Interactively resample the data on slices perpendicular to the x-,y-, z-axis
  - Use visualization techniques for 2D scalar fields
    - Color coding
    - Isolines
    - Height fields



Slice 20 30 40 50 60

CT data set

Volume Visualization by Slicing

- Alternative: Oblique slicing (MPR multiplanar reformating)
  - Resample the data on arbitrarily oriented slices
    - Resampling (interpolation)
    - e.g., exploit 3D texture mapping functionality of OpenGL/Direct3D…
    - …or compute trilinear interpolation manually

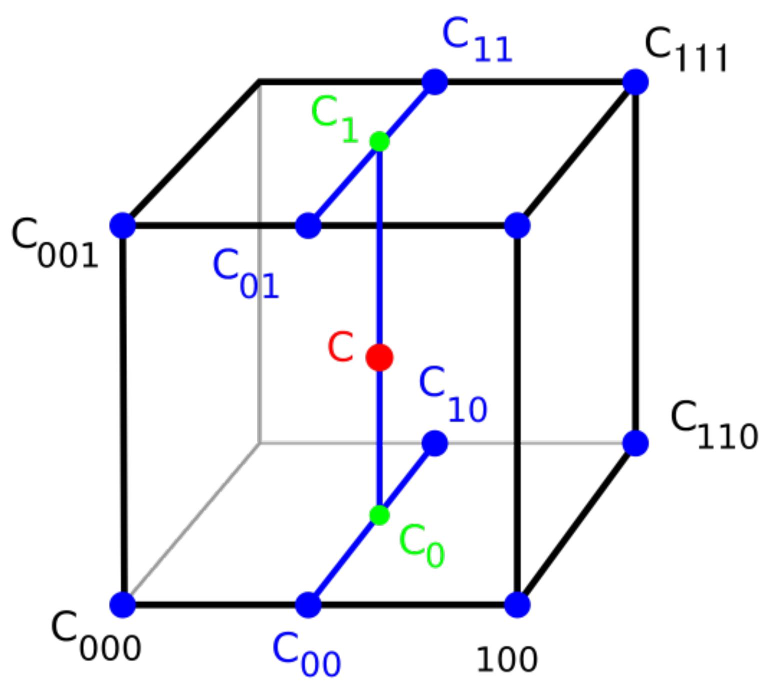
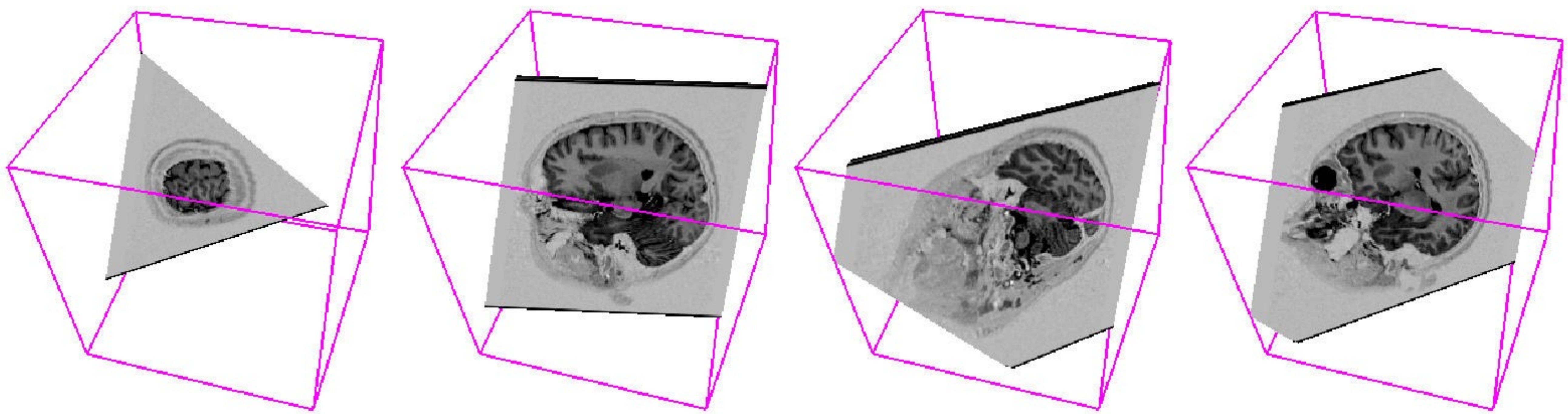
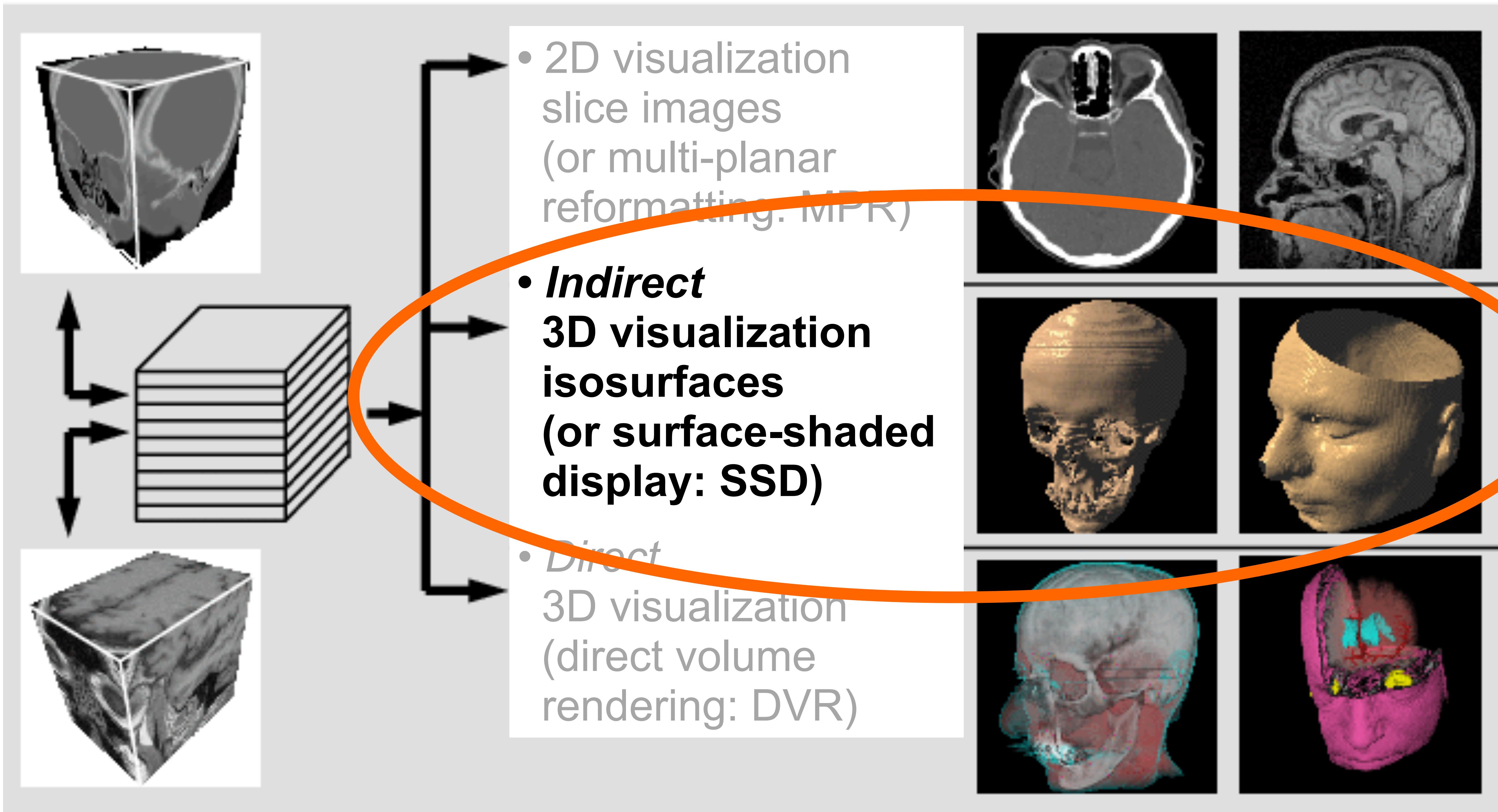


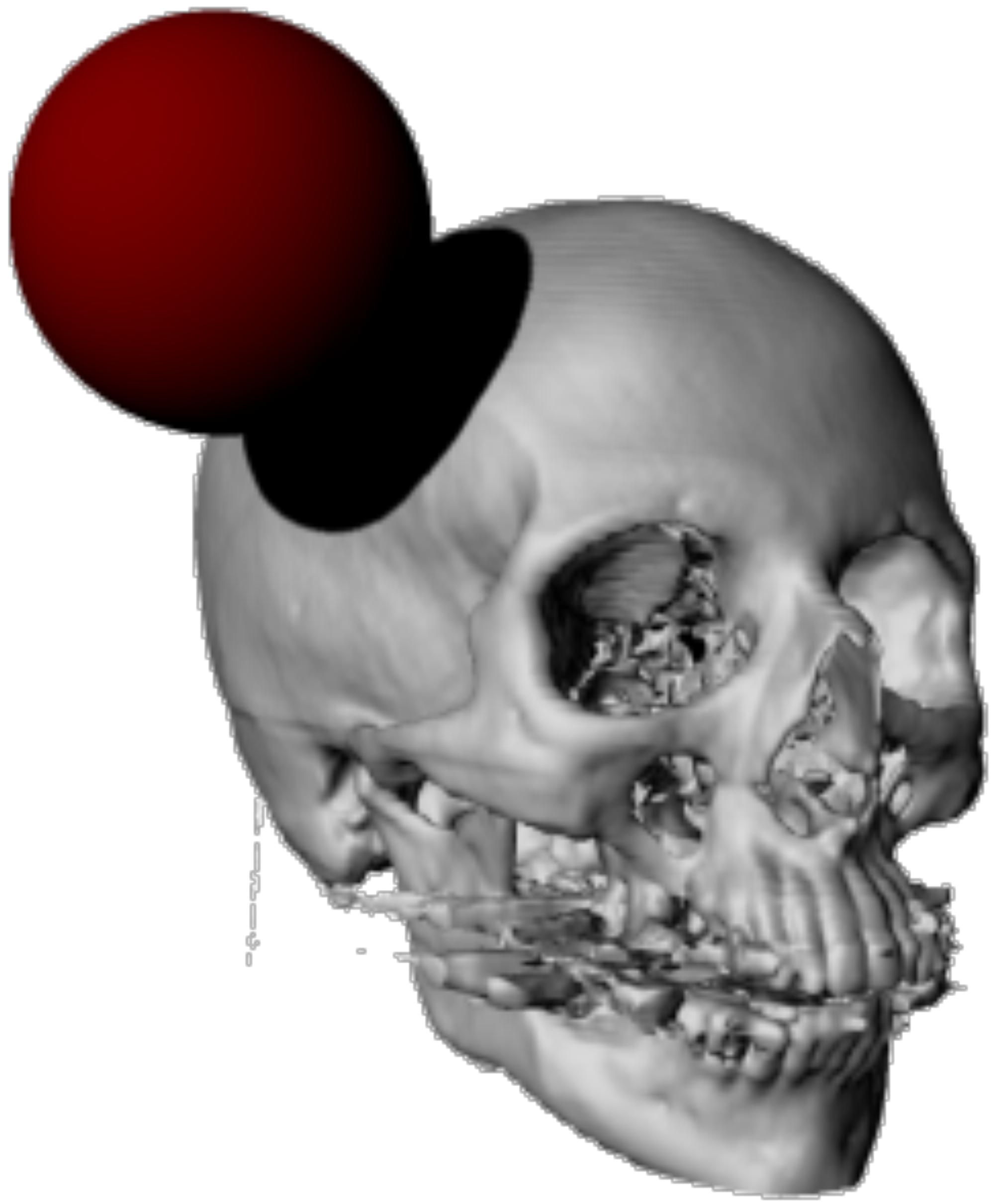
Image source: wikipedia

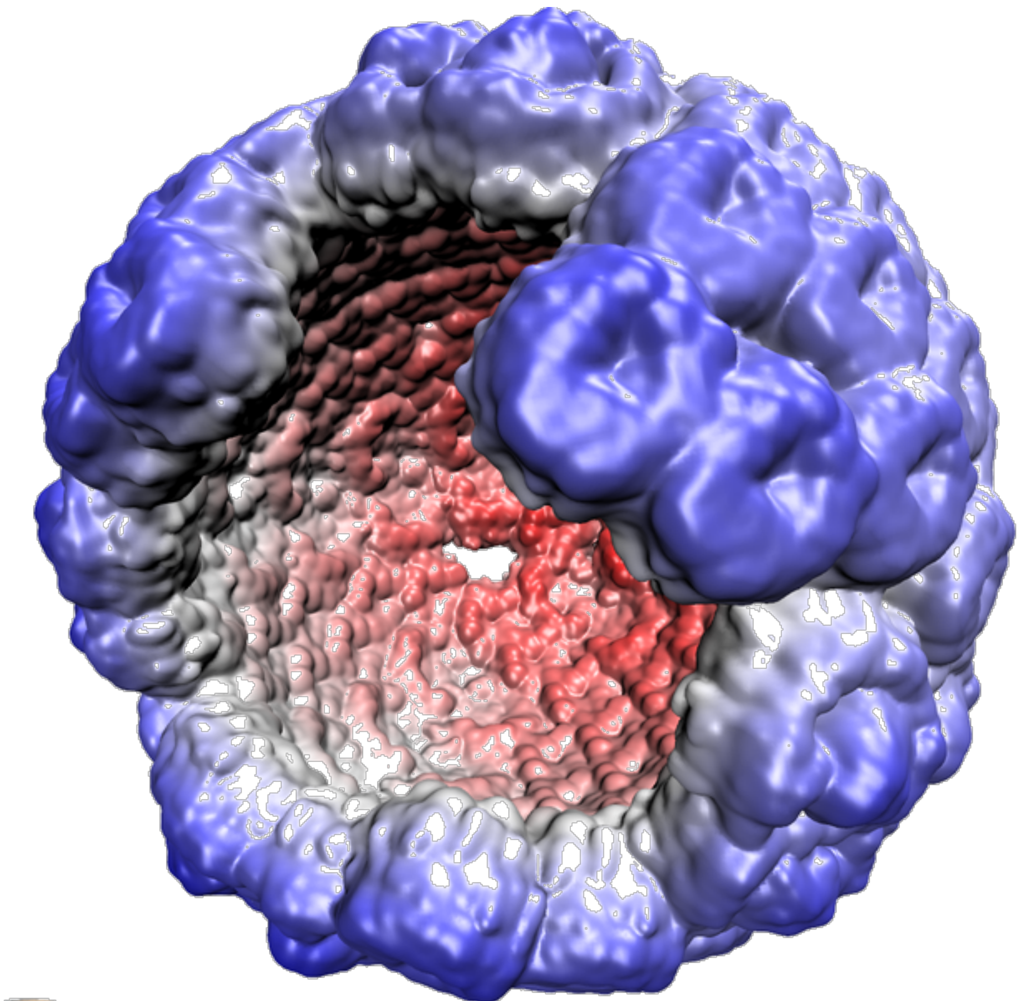


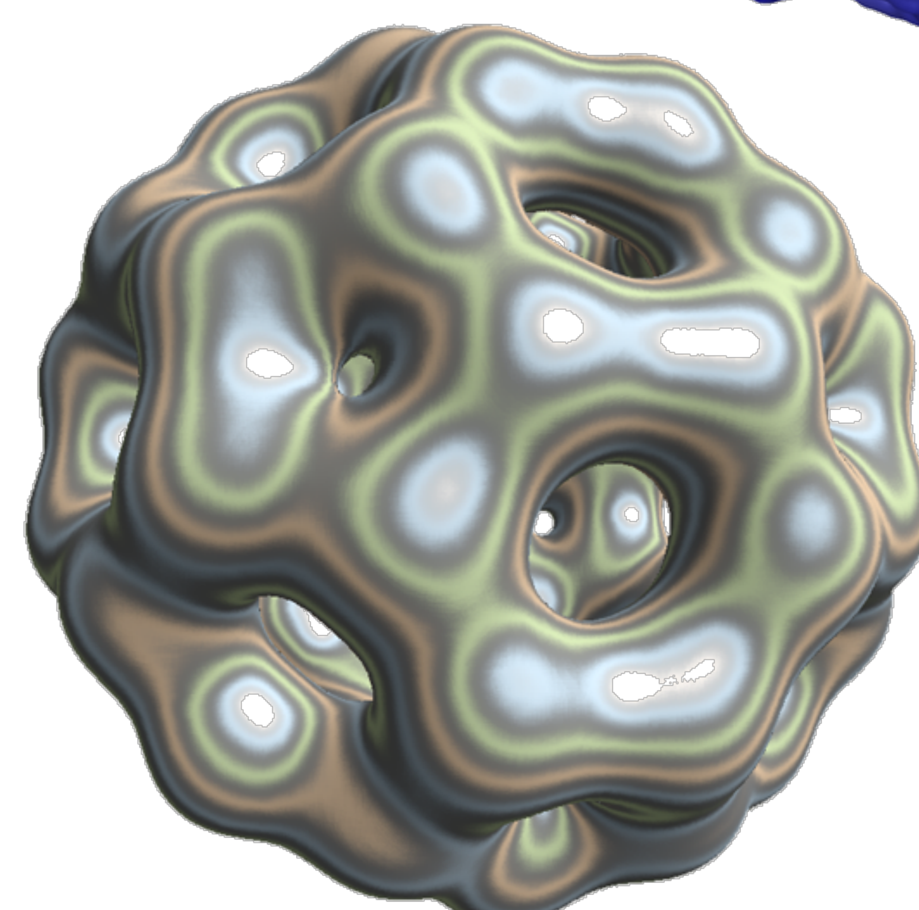
Volume Visualization by Slicing



Isosurfaces: Examples







Marching Cubes

- Isosurface extraction by the Marching-Cubes (MC) algorithm
  [Lorensen, Cline 1987] → one of the most cited papers in the CG field (>12,000)
  - Works on the original data
  - Approximates the surface by a triangle mesh
  - Surface is found by linear interpolation along cell edges
- ***THE*** standard geometry-based isosurface extraction algorithm
- Extension of Marching Squares to 3D
- Assumes a uniform or rectilinear grid

→ Similar for general hexahedral grids

→ Related Marching algorithms for unstructured grids

Marching Cubes: Algorithm

- The core Marching-Cubes algorithm
  - Cell (cube) consists of 8 grid values: (i+[01], j+[01], k+[01])

1. Consider a cell
2. Classify each vertex as inside or outside
3. Build an index
4. Get edge list from table[index]
5. Interpolate the edge location
6. Compute gradients
7. Consider ambiguous cases
7. Go to next cell

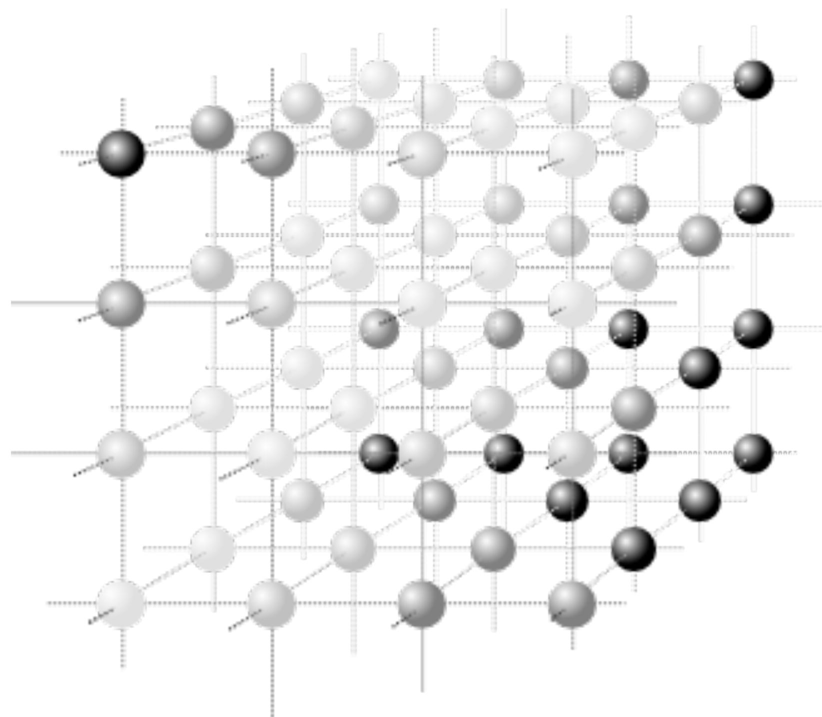
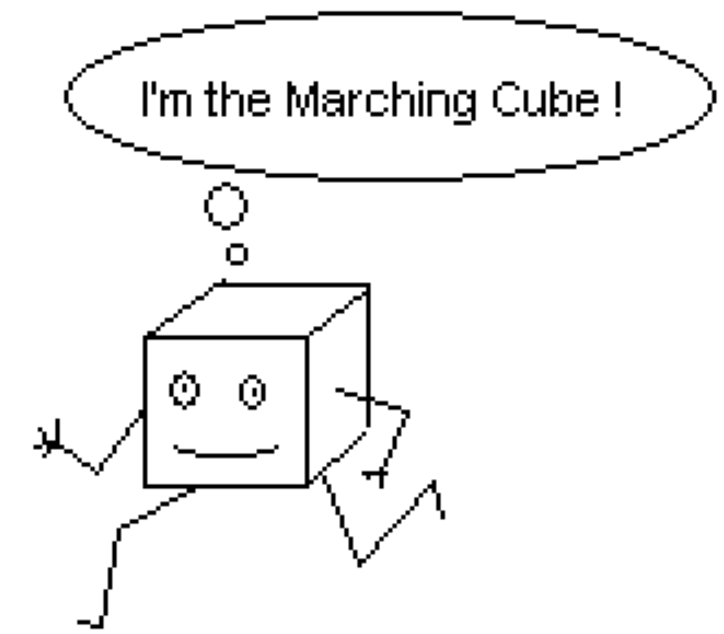
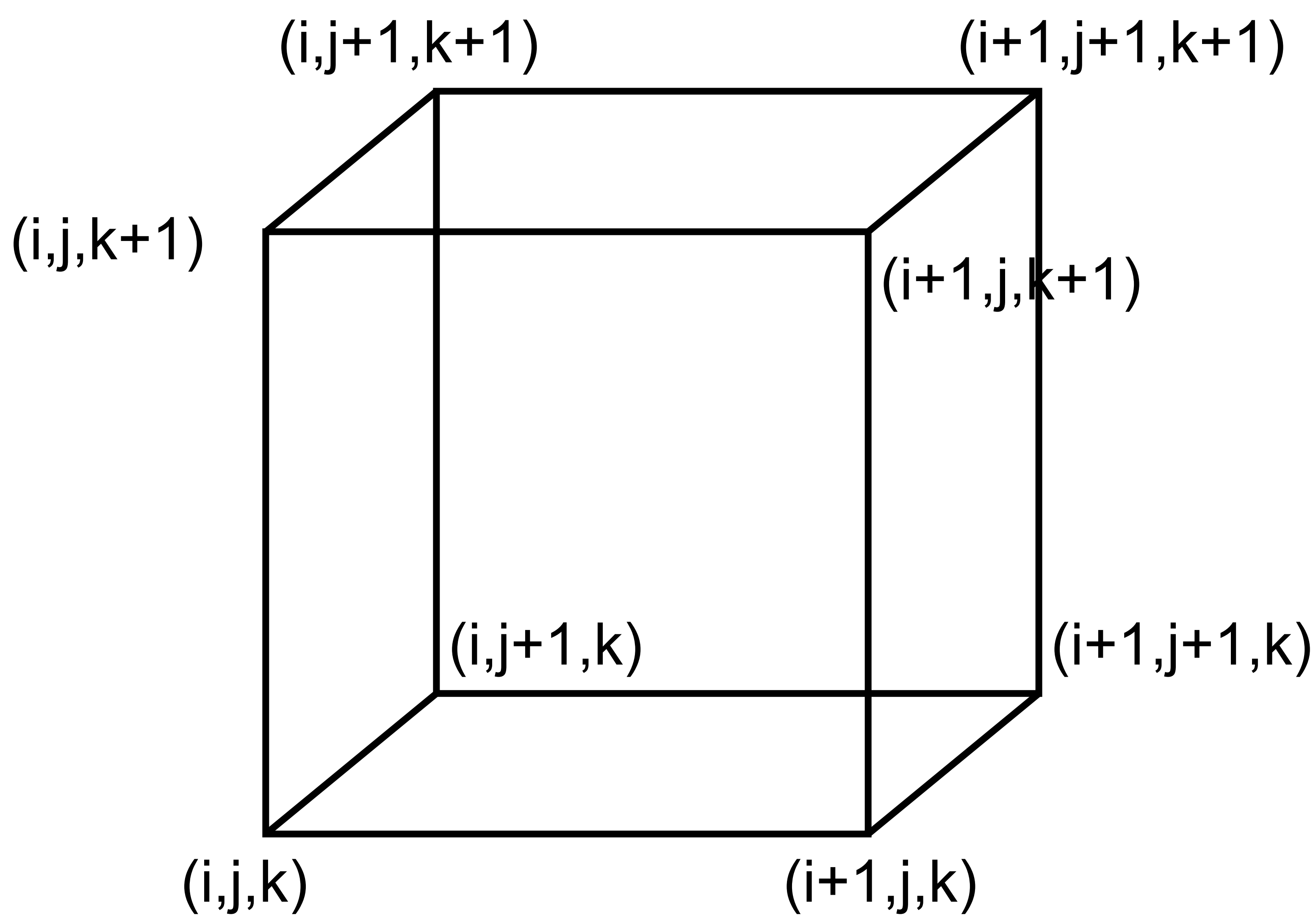


Image source: wikipedia



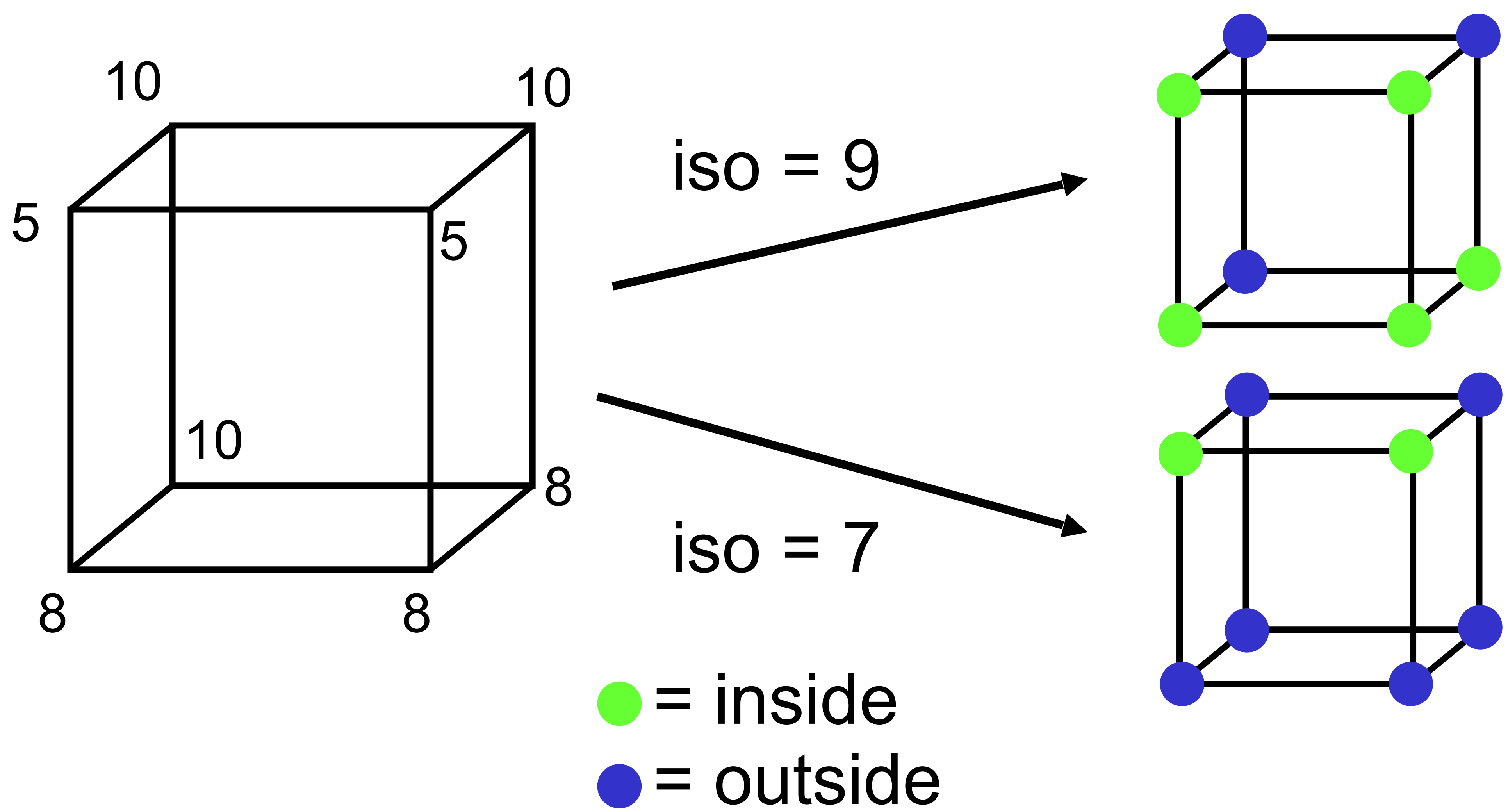
Marching Cubes: Algorithm

- Step 1: Consider a cell defined by eight data values



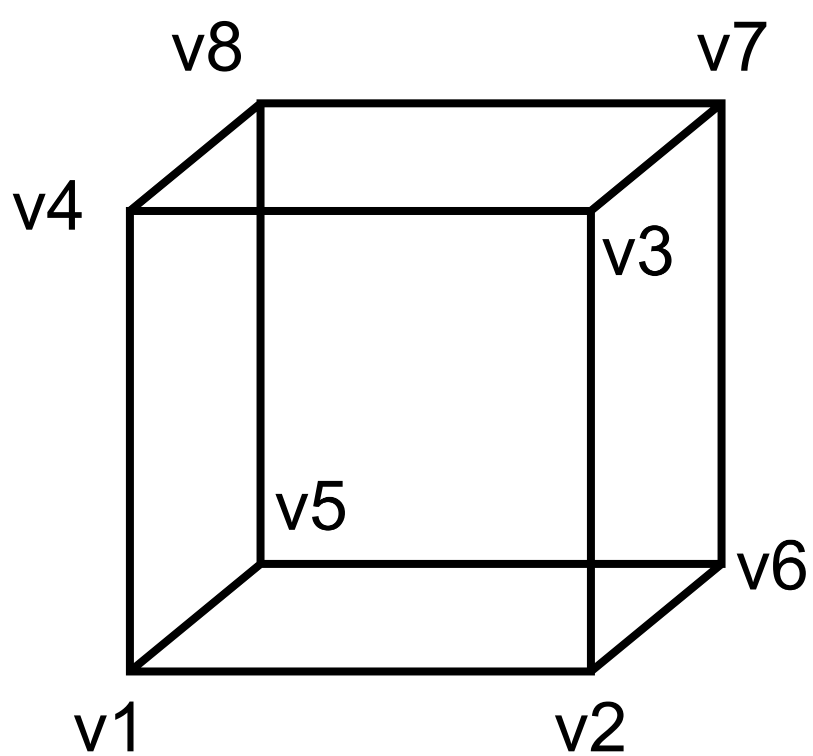
Marching Cubes: Algorithm

- Step 2: Classify each cell according to whether it lies
  - Outside the surface (value > isosurface value)
  - Inside the surface (value <= isosurface value)

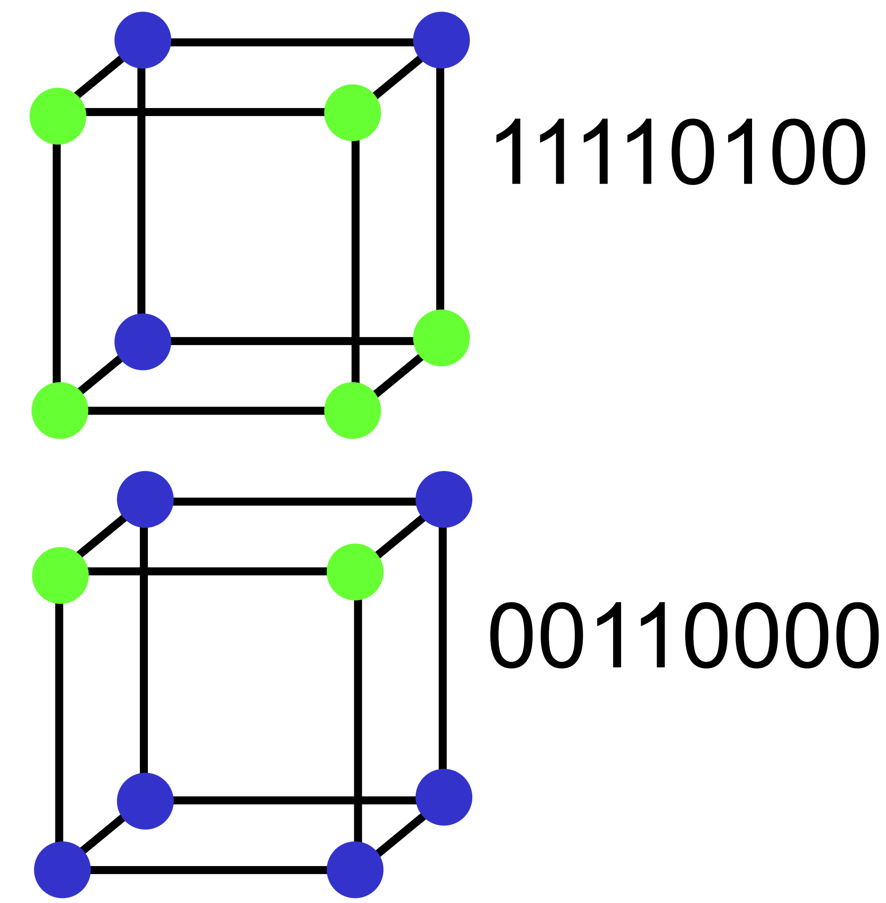


Marching Cubes: Algorithm

- Step 3: Use the binary labeling of each cell to create an index



inside =1
outside=0



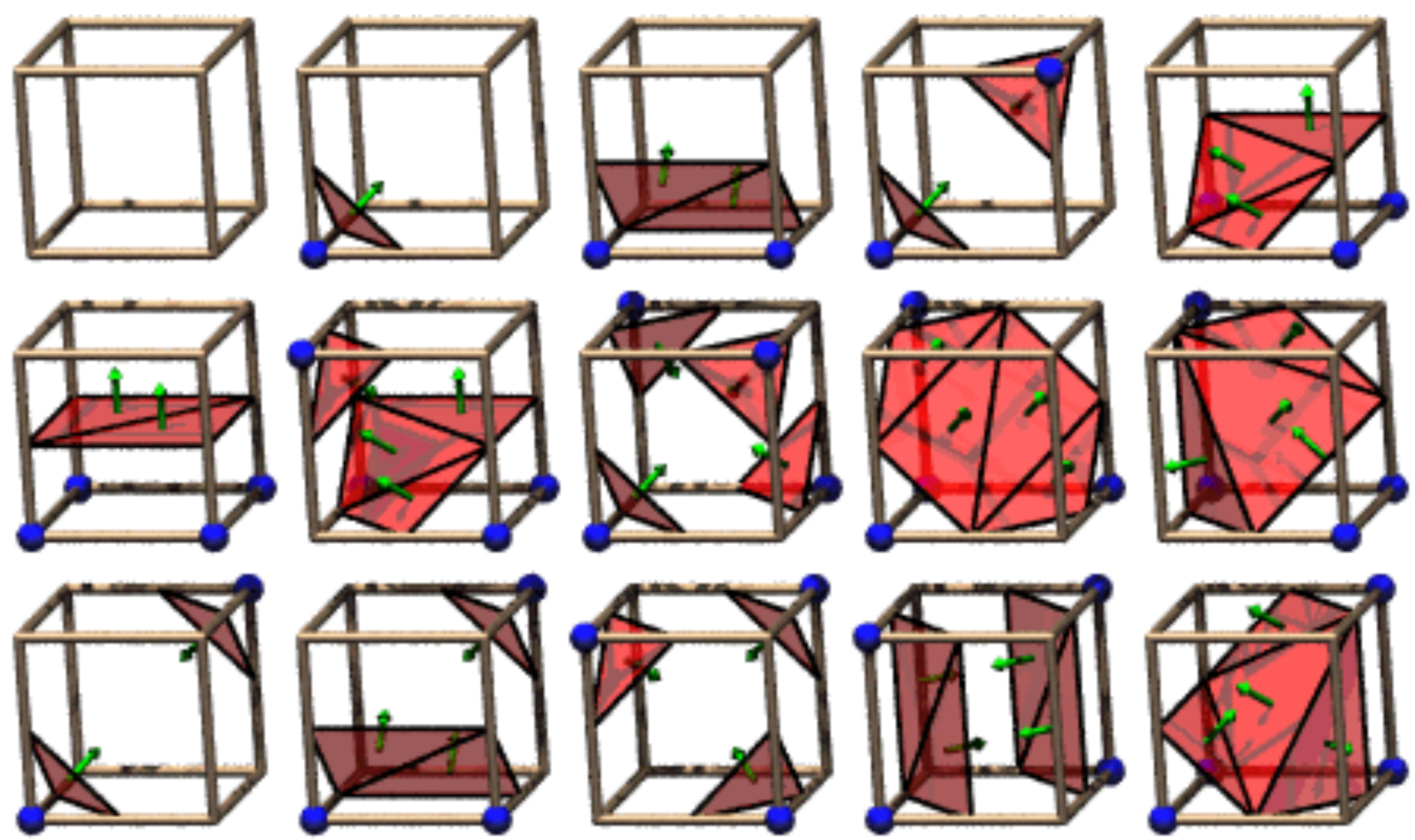
Index:

v1	v2	v3	v4	v5	v6	v7	v8

Binary index number = integer

Marching Cubes: Algorithm

- Step 4: For a given index, access an array storing a list of edges
  - All 256 cases can be derived from 1 + 14 = 15 base cases due to symmetries
  - Each case creates at most 5 triangles (dual cases for inverted signs)



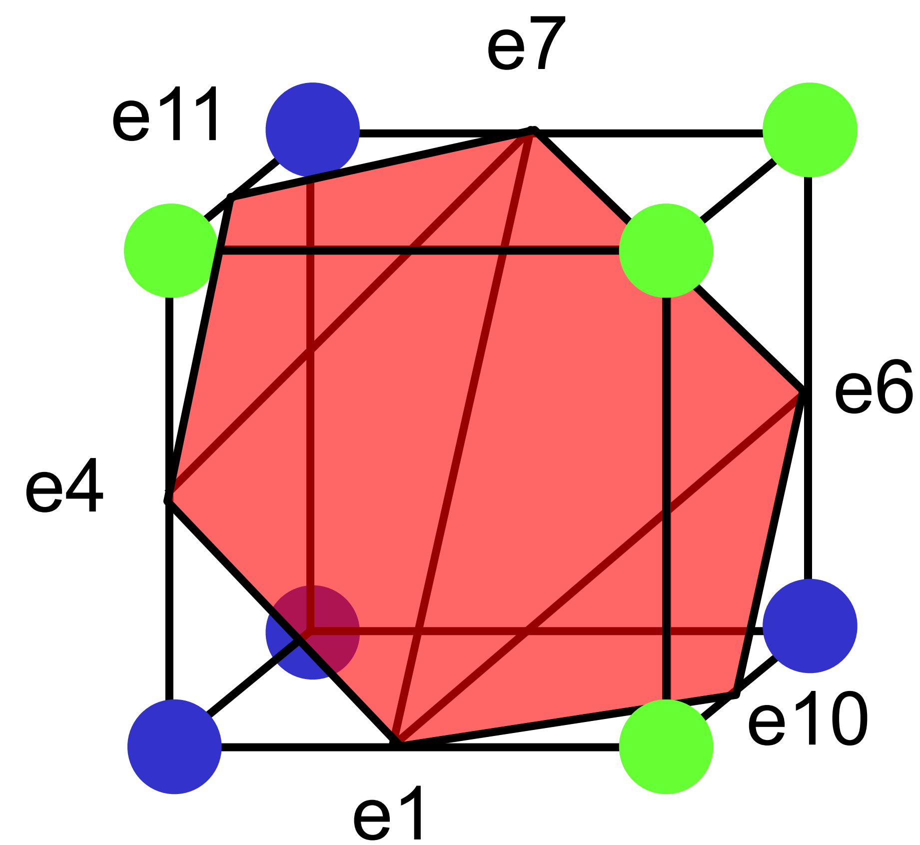
The 15 Cube Combinations

Marching Cubes: Algorithm

- Step 4 *cont.*: Get edge list from table
  - Example:

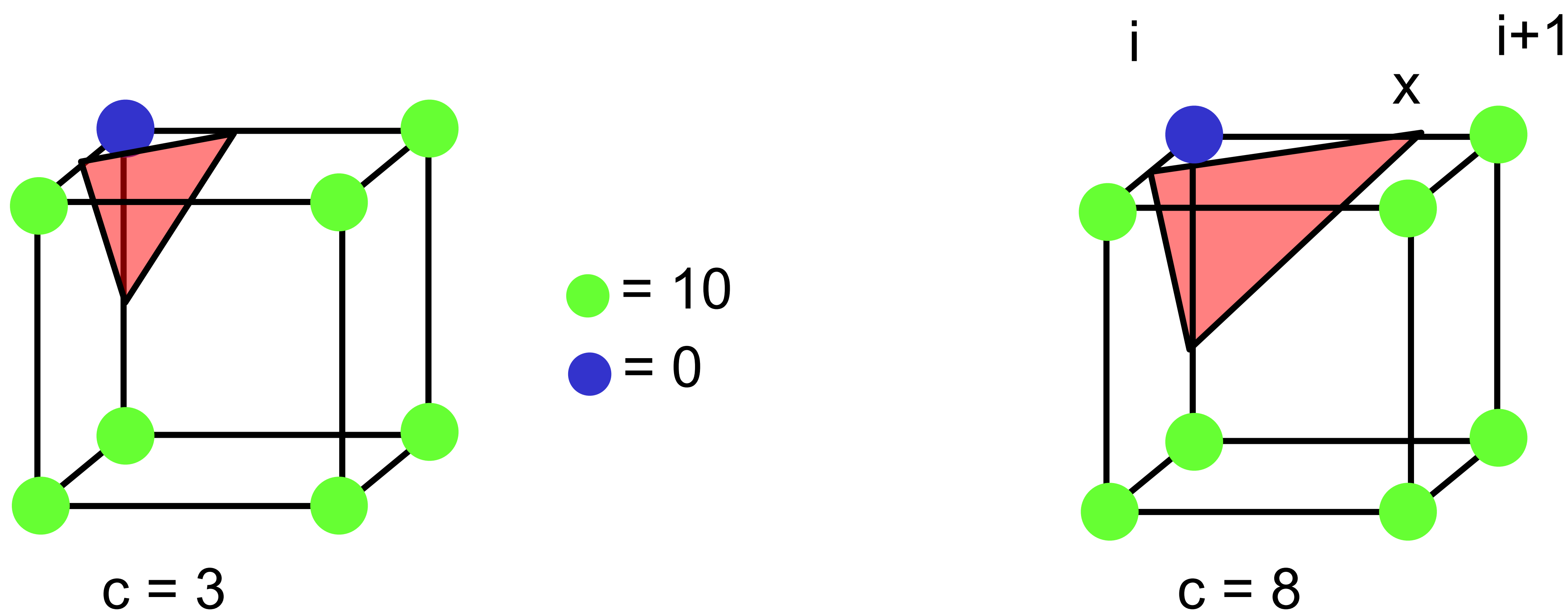
    Index = 10110001
    triangle 1 = e4,e7,e11
    triangle 2 = e1, e7, e4
    triangle 3 = e1, e6, e7
    triangle 4 = e1, e10, e6

  - Face normals encoded implicitly by order of vertices
  - Normal points to higher (or lower) values of the field → inside/outside
  - Vertex normals can be computed by averaging all surrounding face normals



Marching Cubes: Algorithm

- Step 5: For each triangle edge, find the vertex location along the edge using linear interpolation

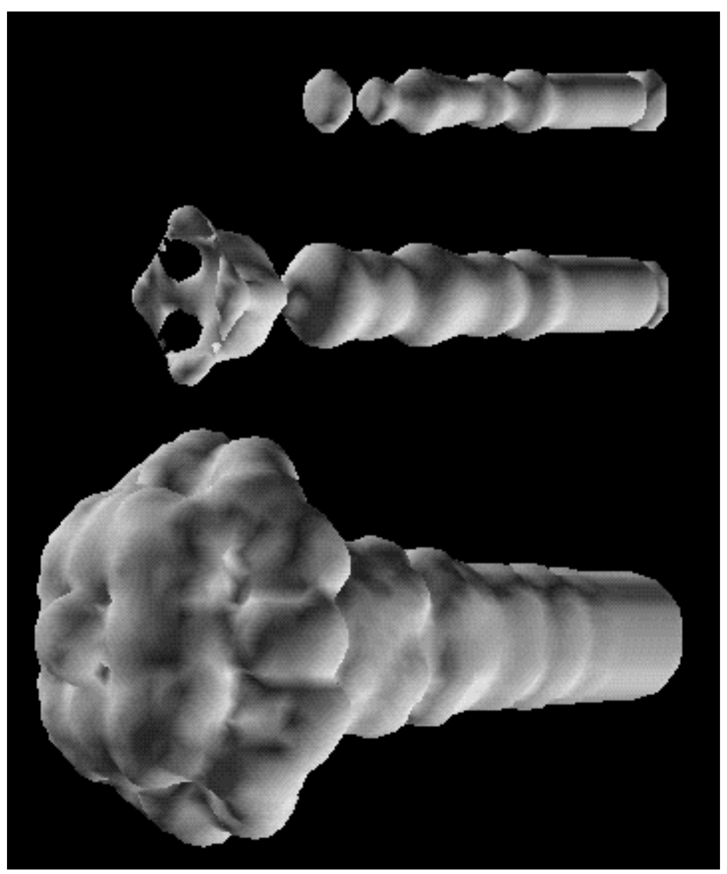


if all edge lengths = 1 $x = i + (c - v[i]) / (v[i+1] - v[i])$

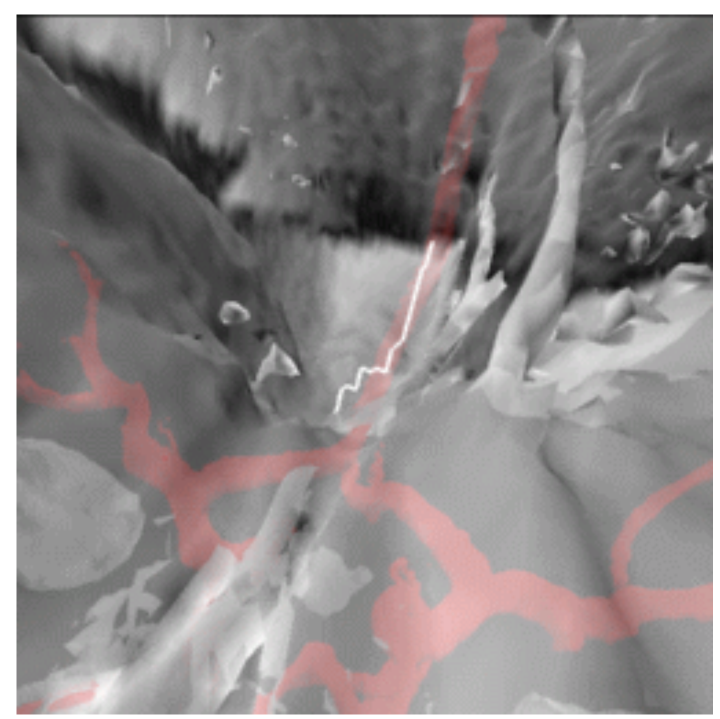
otherwise $x = [(v[i+1] - c)x[i] + (c - v[i])x[i+1]] / (v[i+1] - v[i])$

Marching Cubes: Examples

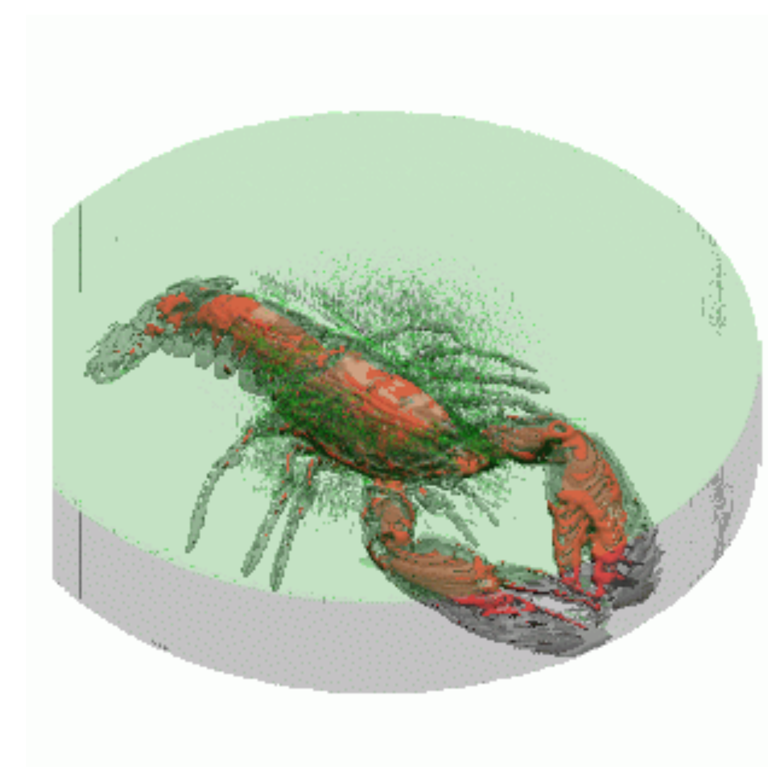
- Possibly overlay of several isosurfaces



1 Isosurface



2 Isosurfaces

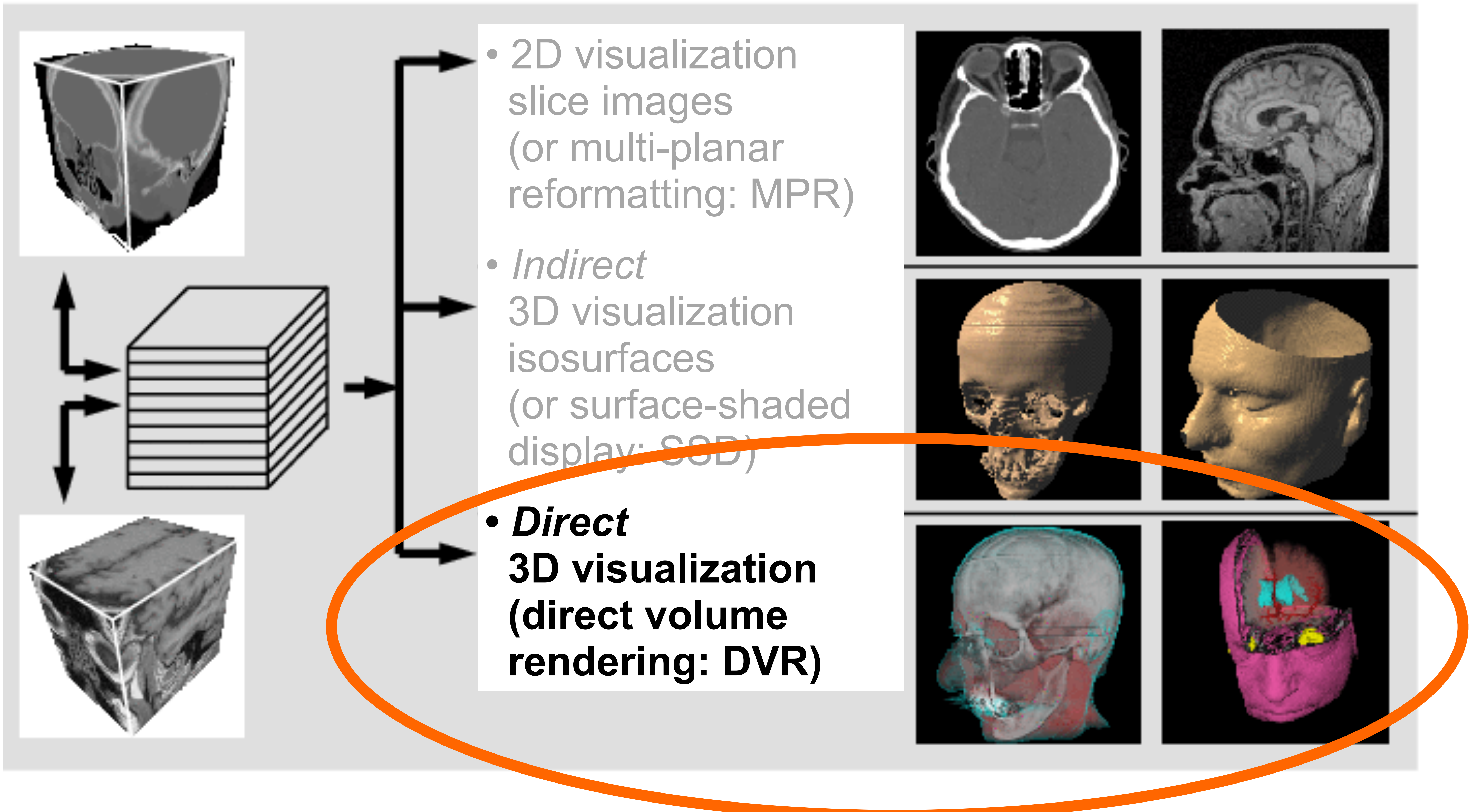


3 Isosurfaces

Marching Cubes: Examples

- Nvidia Cascades Demo for Geforce 8800 (~2008)
  - Real-time generation of volumetric terrain on GPU
  - Marching Cubes isosurface extraction in Geometry Shader

Volume Visualization by Slicing

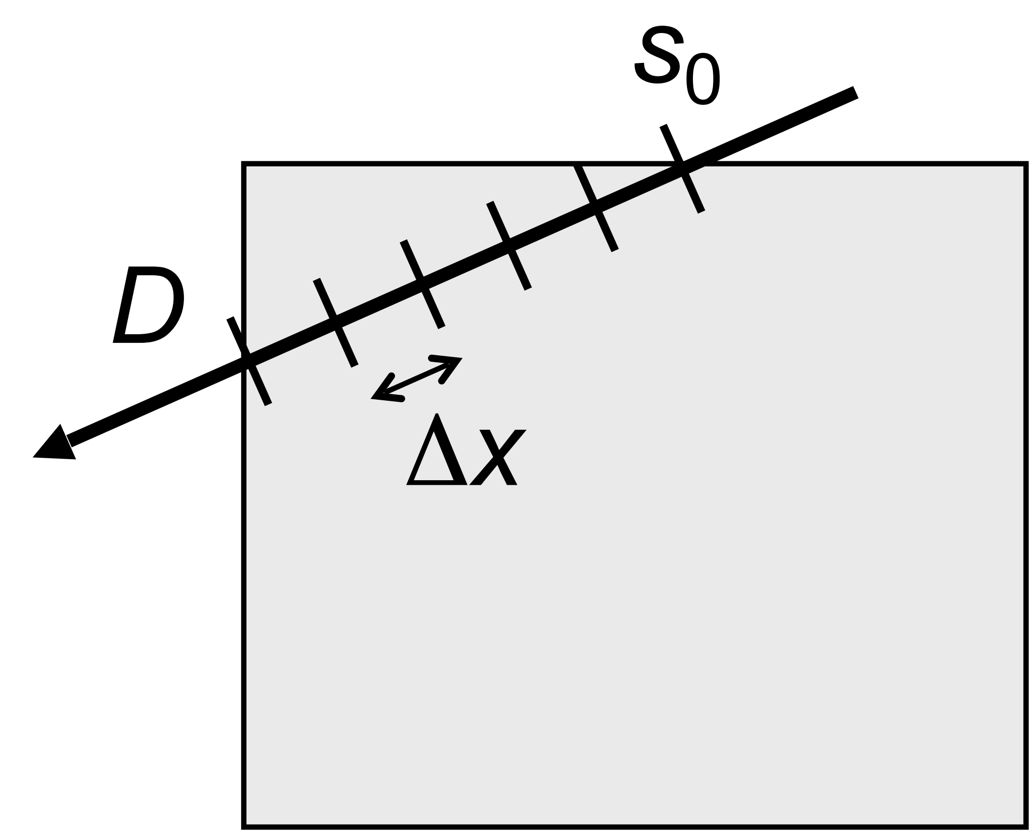


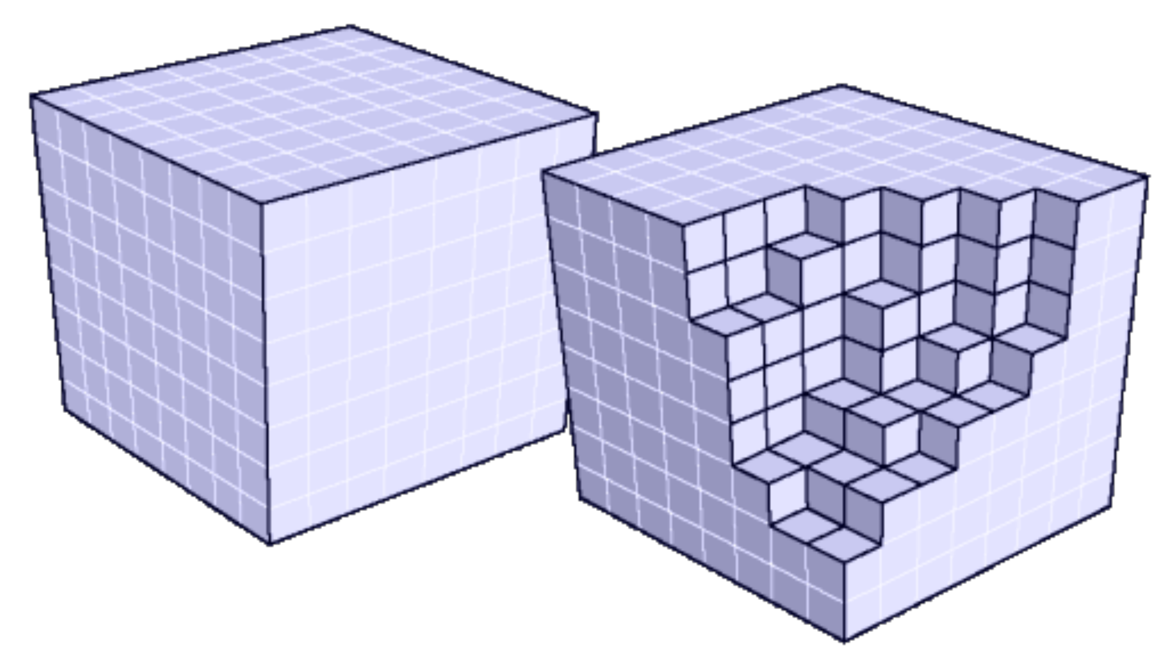
Direct Volume Rendering

- Directly get a 3D representation of the volume data
  - The data is considered to represent a semi-transparent light-emitting medium
  - Approaches are based on the laws of physics (emission, absorption, scattering)
  - The volume data is used as a whole (look inside, see all interior structures)

Direct Volume Rendering

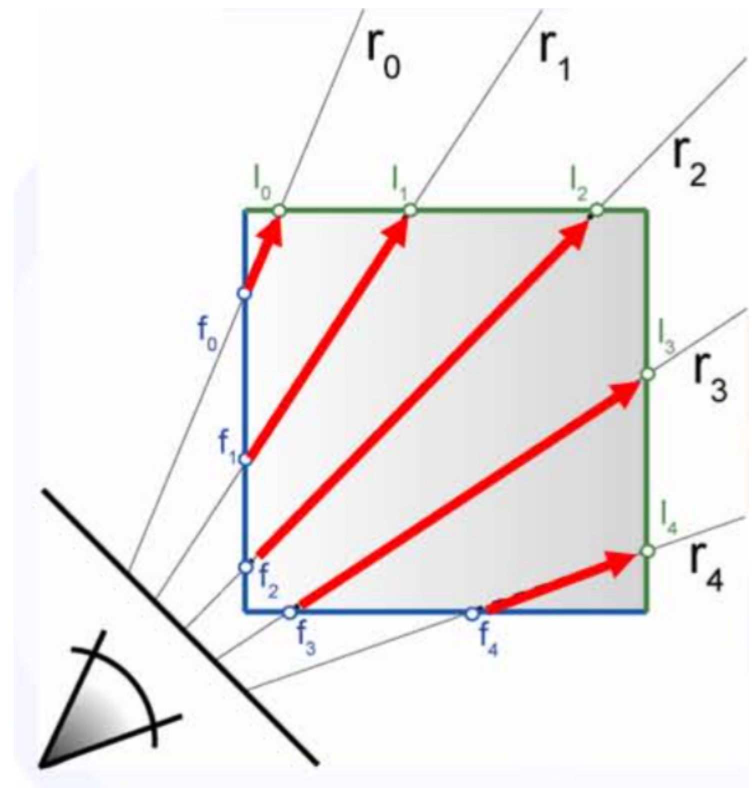
- Optical model:
  volume rendering integral
- Integral approximated by sum along virtual light rays
- Compositing (accumulation of color and opacity) depends on
  - Incoming light (RGB)
  - Opacity (A = alpha = 1-transparency)

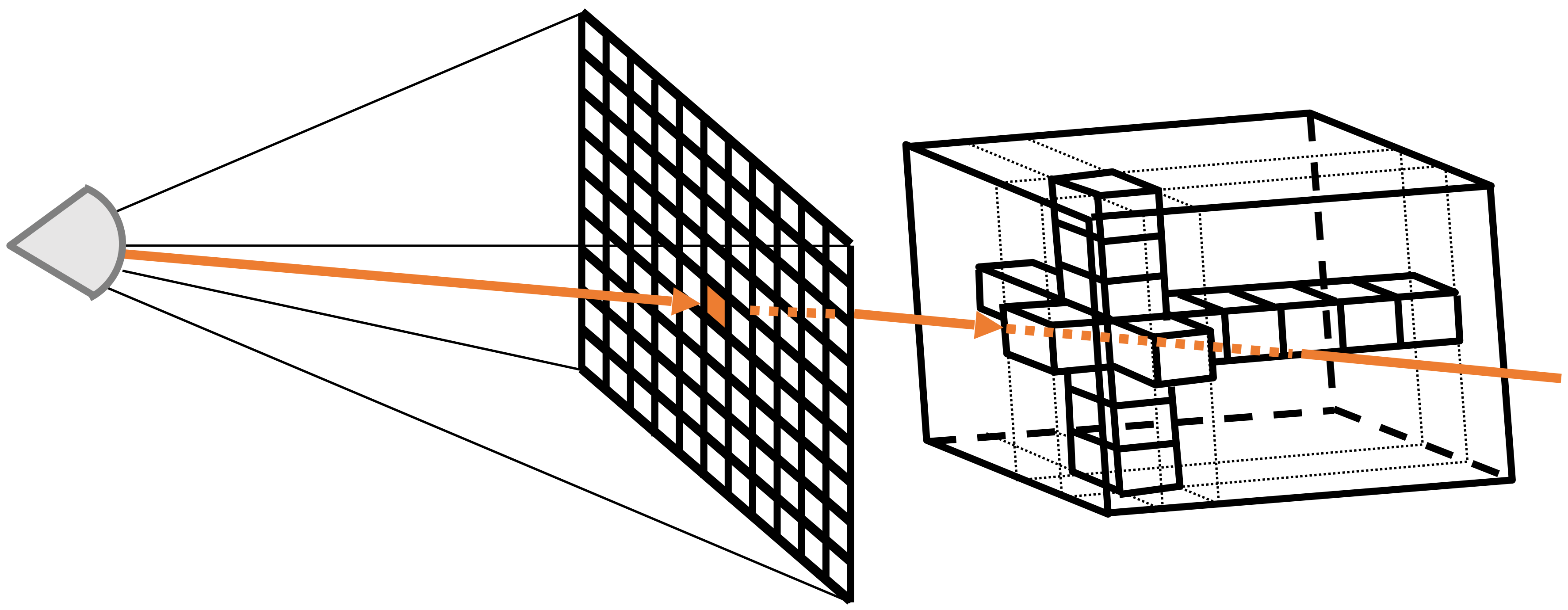




Ray Casting

- Similar to ray tracing in surface-based computer graphics
- In volume rendering we usually only deal with primary rays; hence: *ray casting (or ray marching)*
- Natural **image-order** technique
- Performed pixel-by-pixel



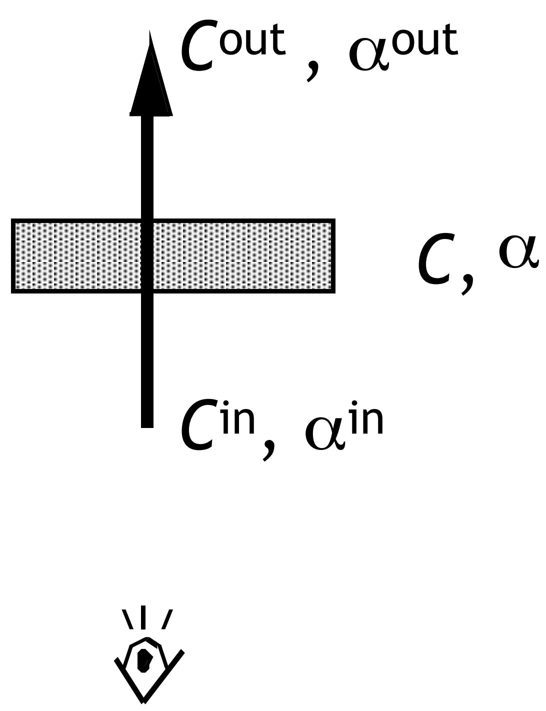


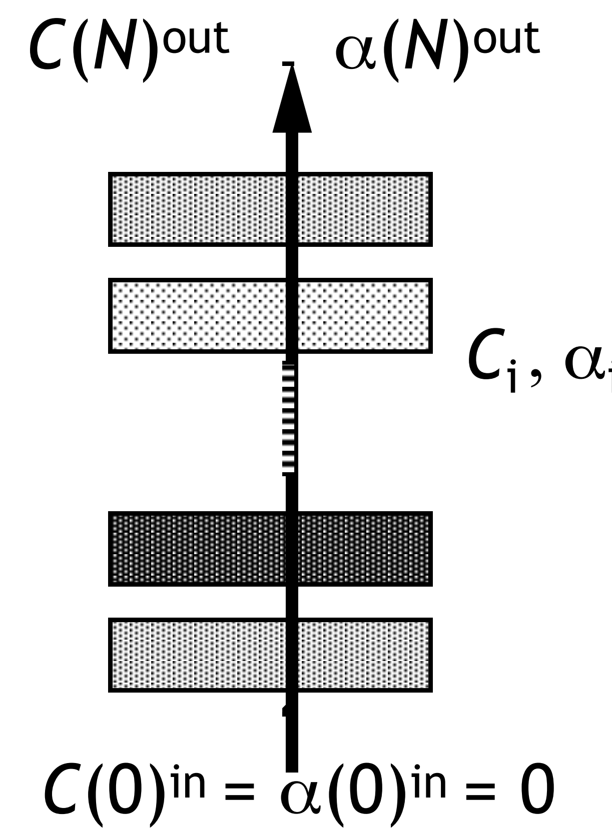
Compositing along Rays

- Front-to-back compositing
  - Combine (sorted) color and opacity values along a ray
  - Most often used in ray casting
  - Allows for early ray termination
- Compositing equation:

$$C^{out} = C^{in} + (1 - \alpha^{in})\, C\, \alpha \qquad C(i)^{in} = C(i-1)^{out}$$

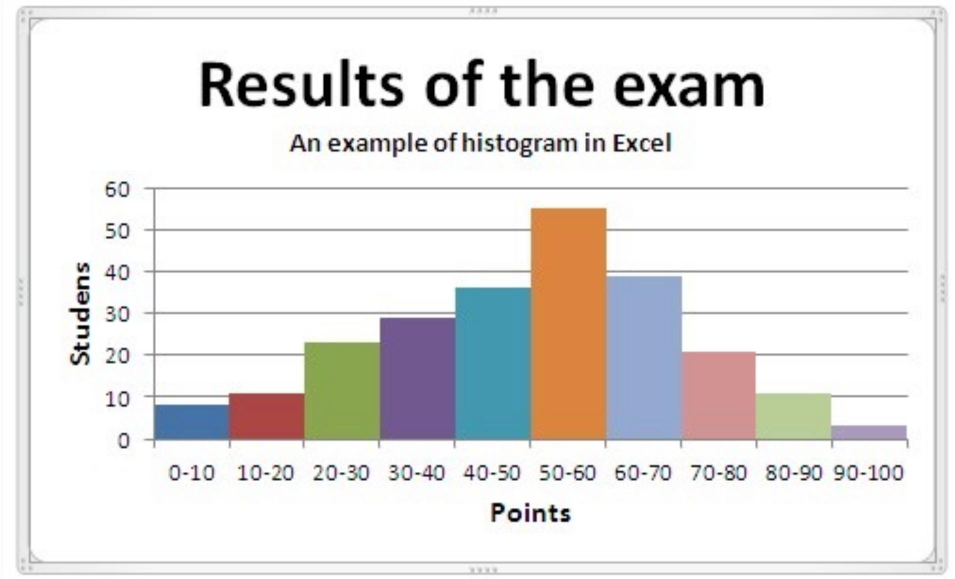
$$\alpha^{out} = \alpha^{in} + (1 - \alpha^{in})\, \alpha \qquad \alpha(i)^{in} = \alpha(i-1)^{out}$$





Problem: Classification

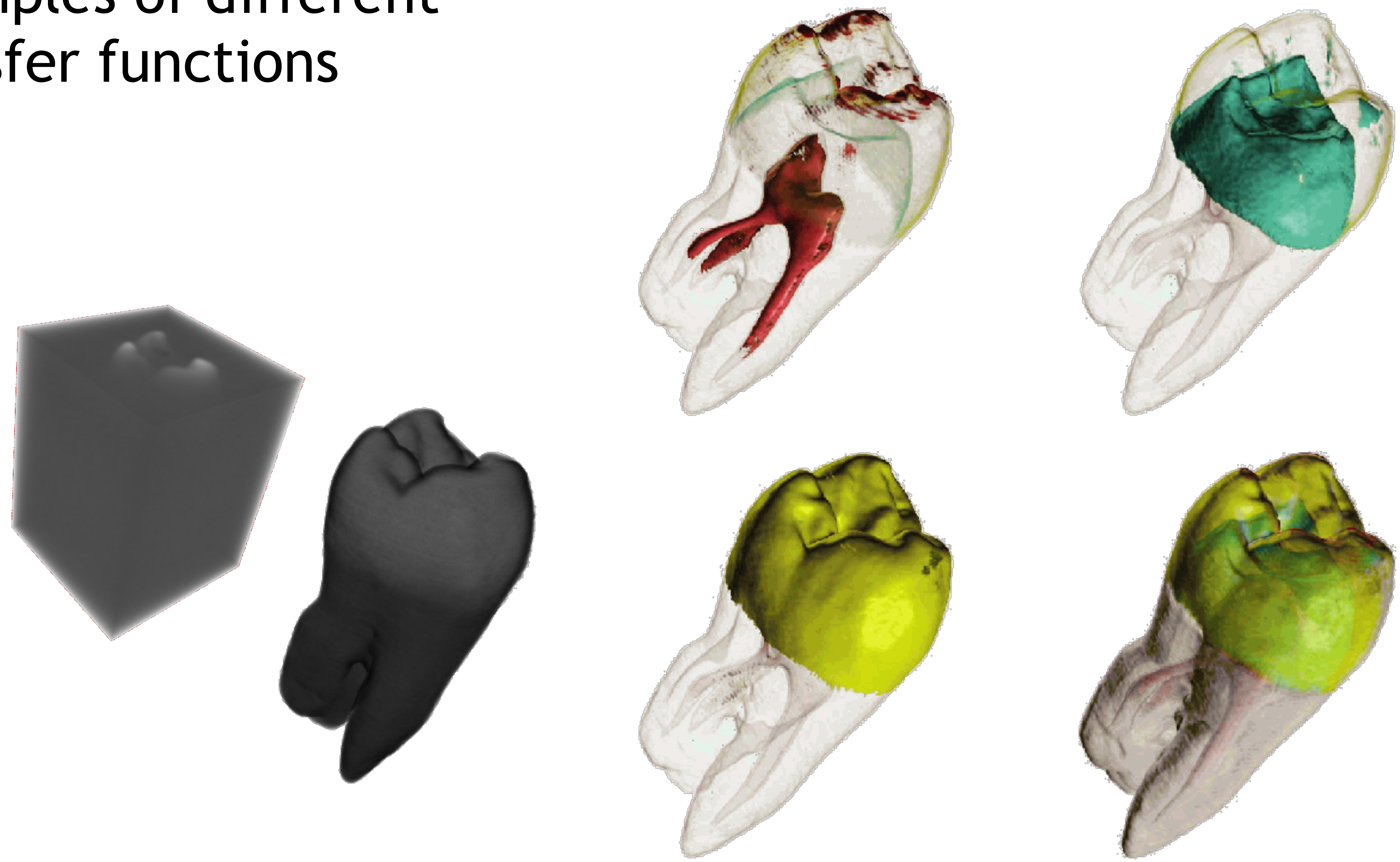
- Missing link so far between
  - Scalar values of the data set and…
  - …color & opacity (RGBA) for volume rendering
- Goals and issues
  - Empowers user to select “structures”
  - Extract important features of the data set
  - Classification is non trivial
  - Histogram can be a useful hint
- Standard approach: Transfer function
  - Color table for volume visualization
  - Maps raw scalar value into presentable entities: color, intensity, opacity, etc.



Histogram
Image source: best-excel-tutorial.com

Classification: Transfer Function

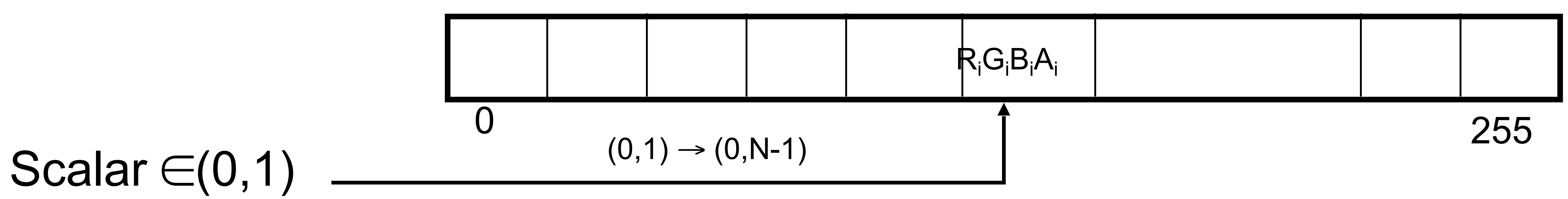
- Examples of different transfer functions



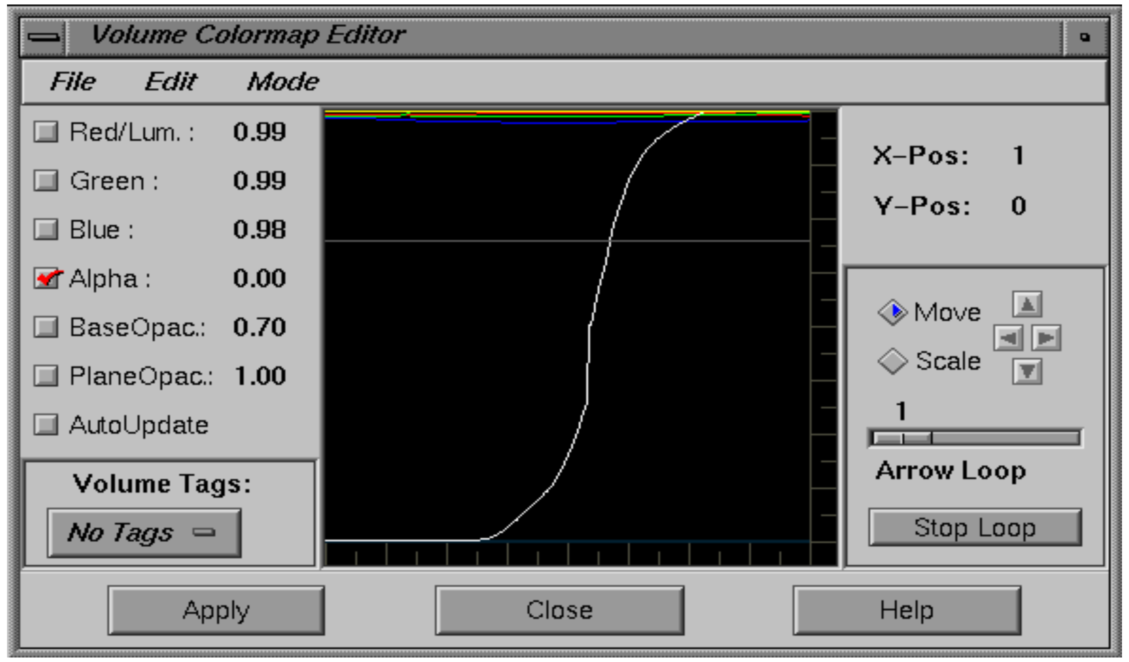
Classification: Transfer Function

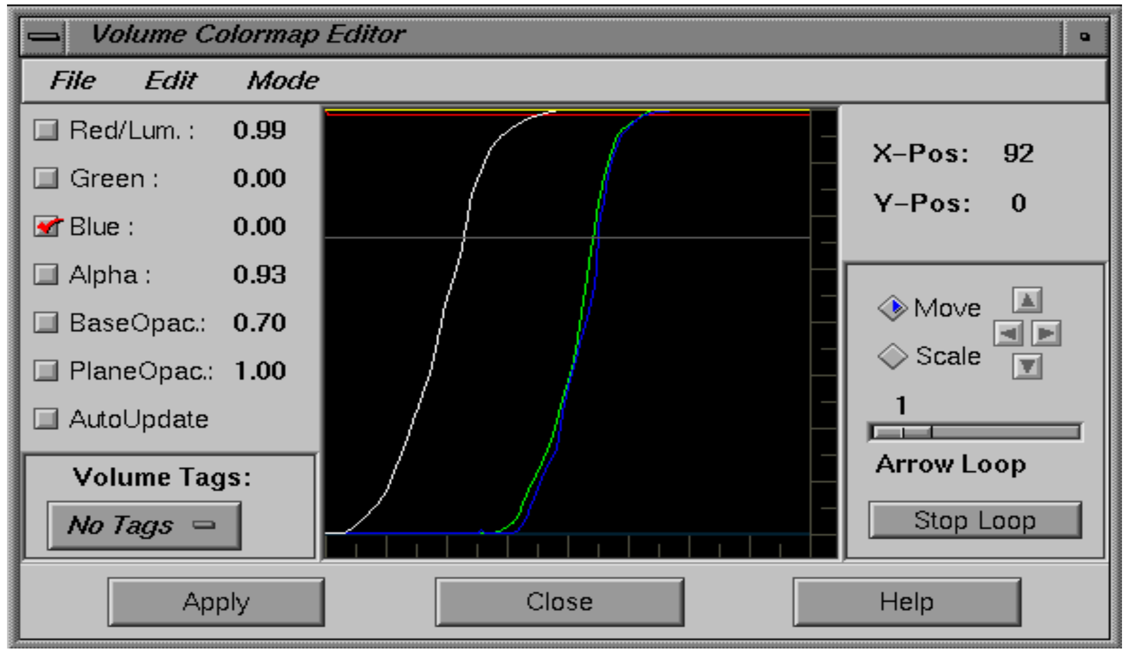
- Most widely used approach for transfer functions:
  - Assign to each scalar value a different color value
  - Assignment via transfer function *T*

    $T : scalarvalue \rightarrow colorvalue$
  - Common choice for color representation: RGBA
  - Code color values into a color lookup table

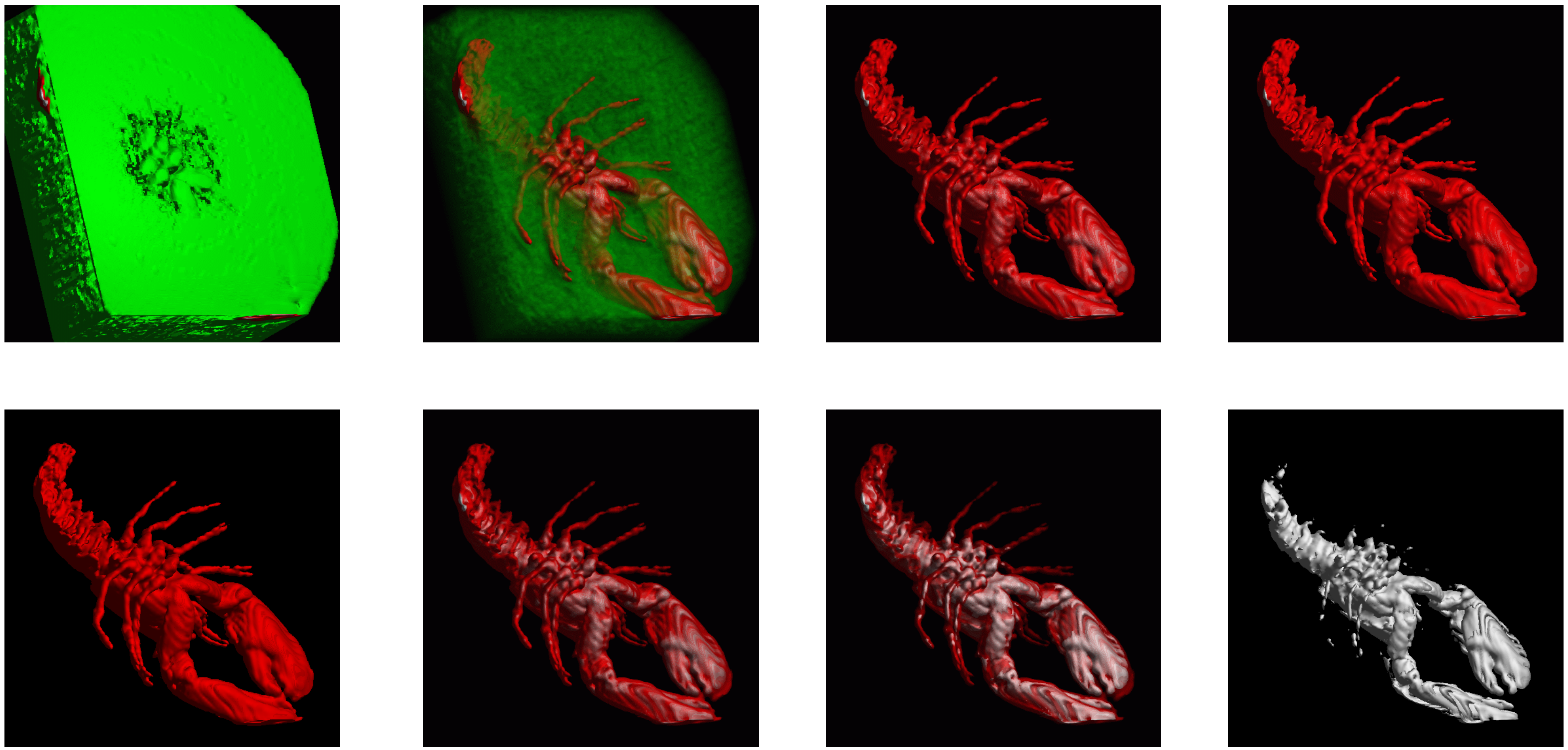


Classification: Example



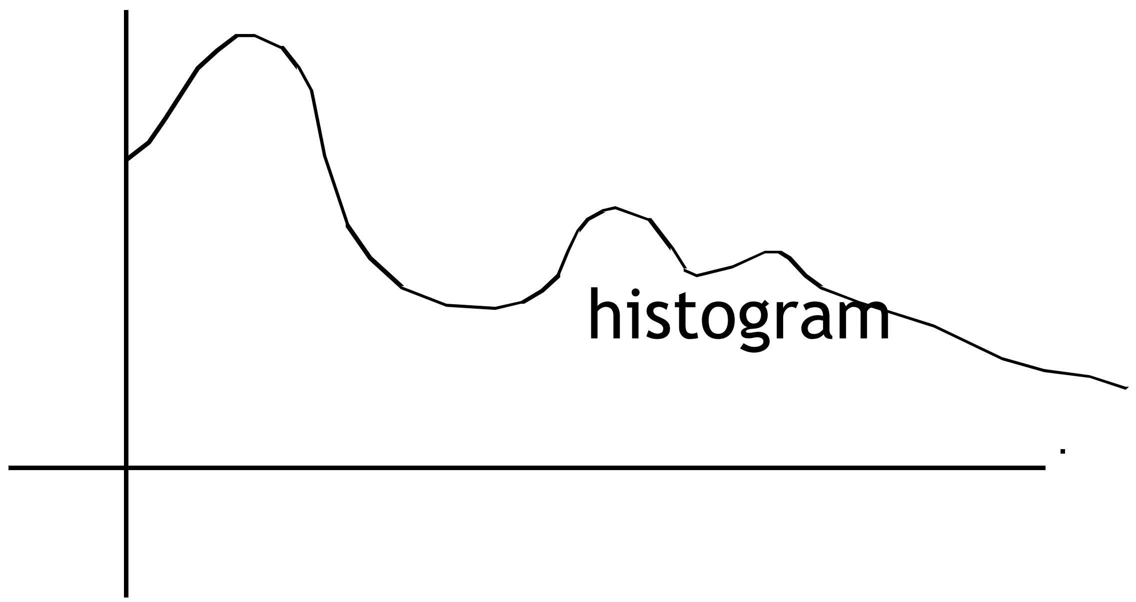


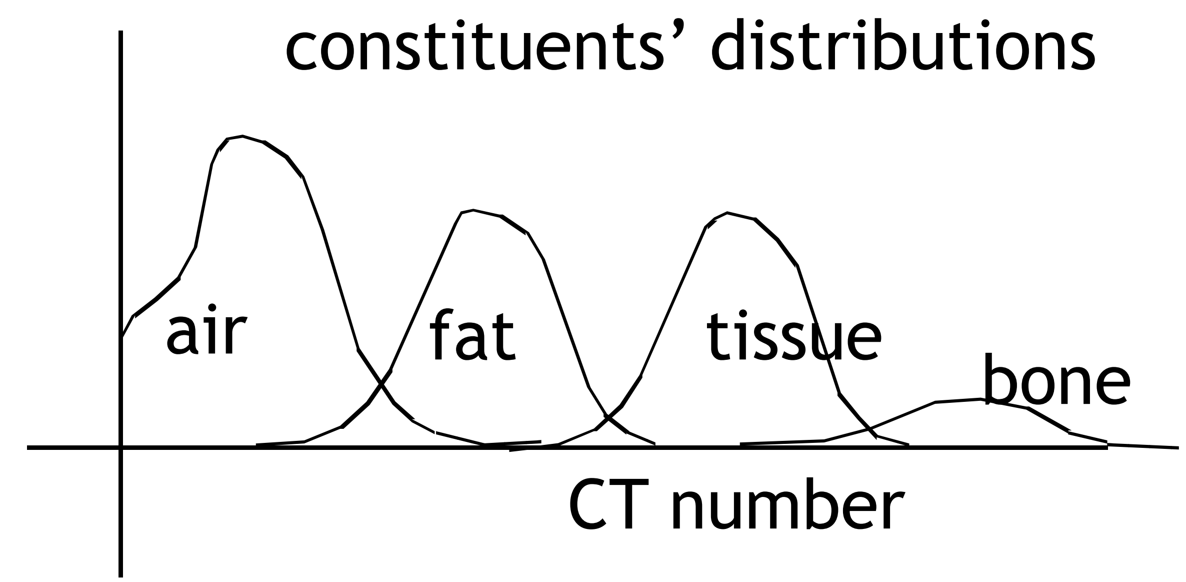
Classification: Example

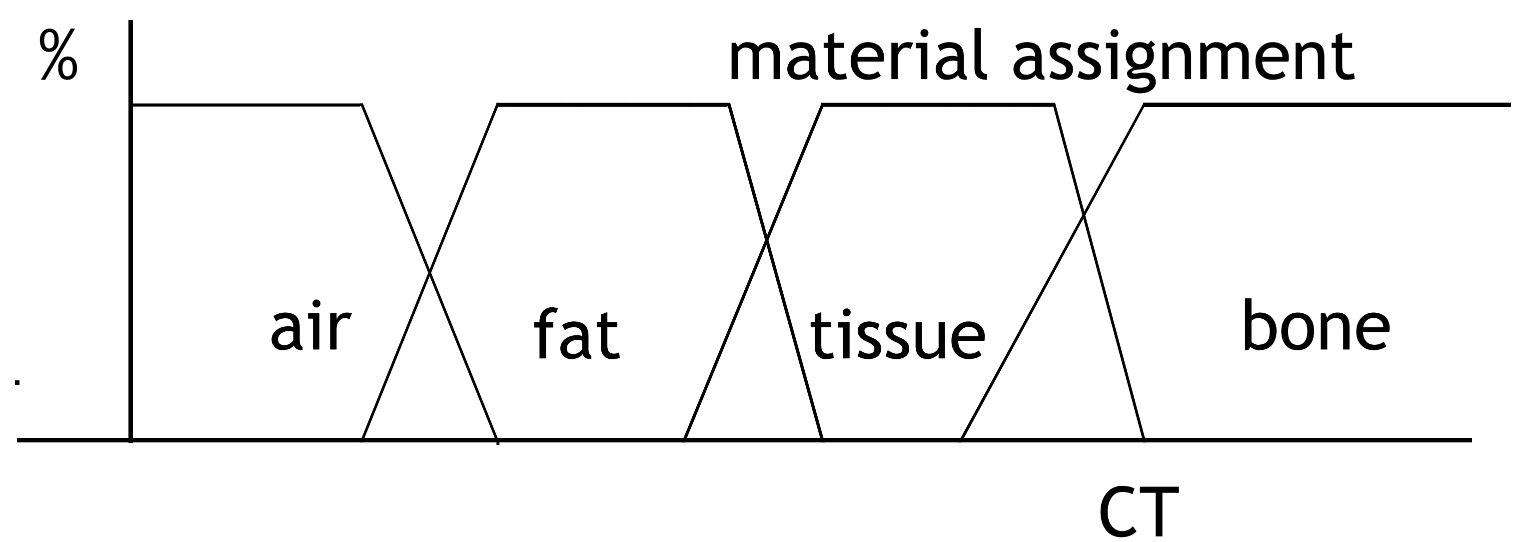


Classification in Medical Imaging

- Heuristic approach, based on measurements of many data sets

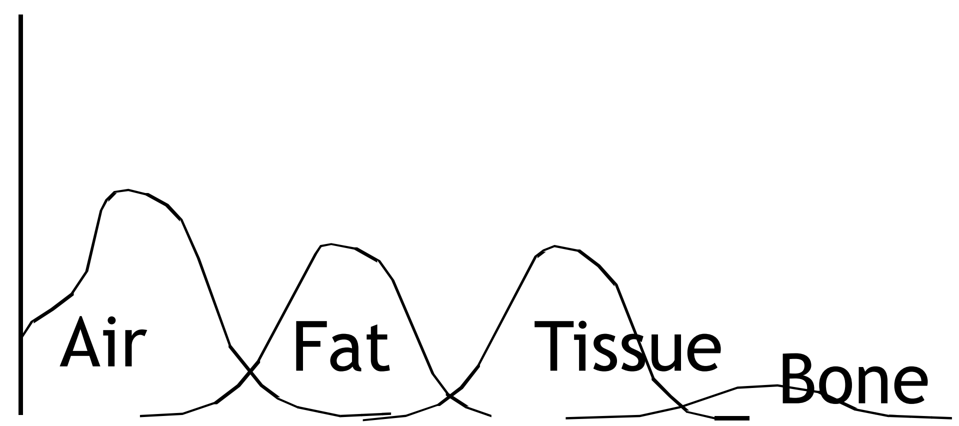






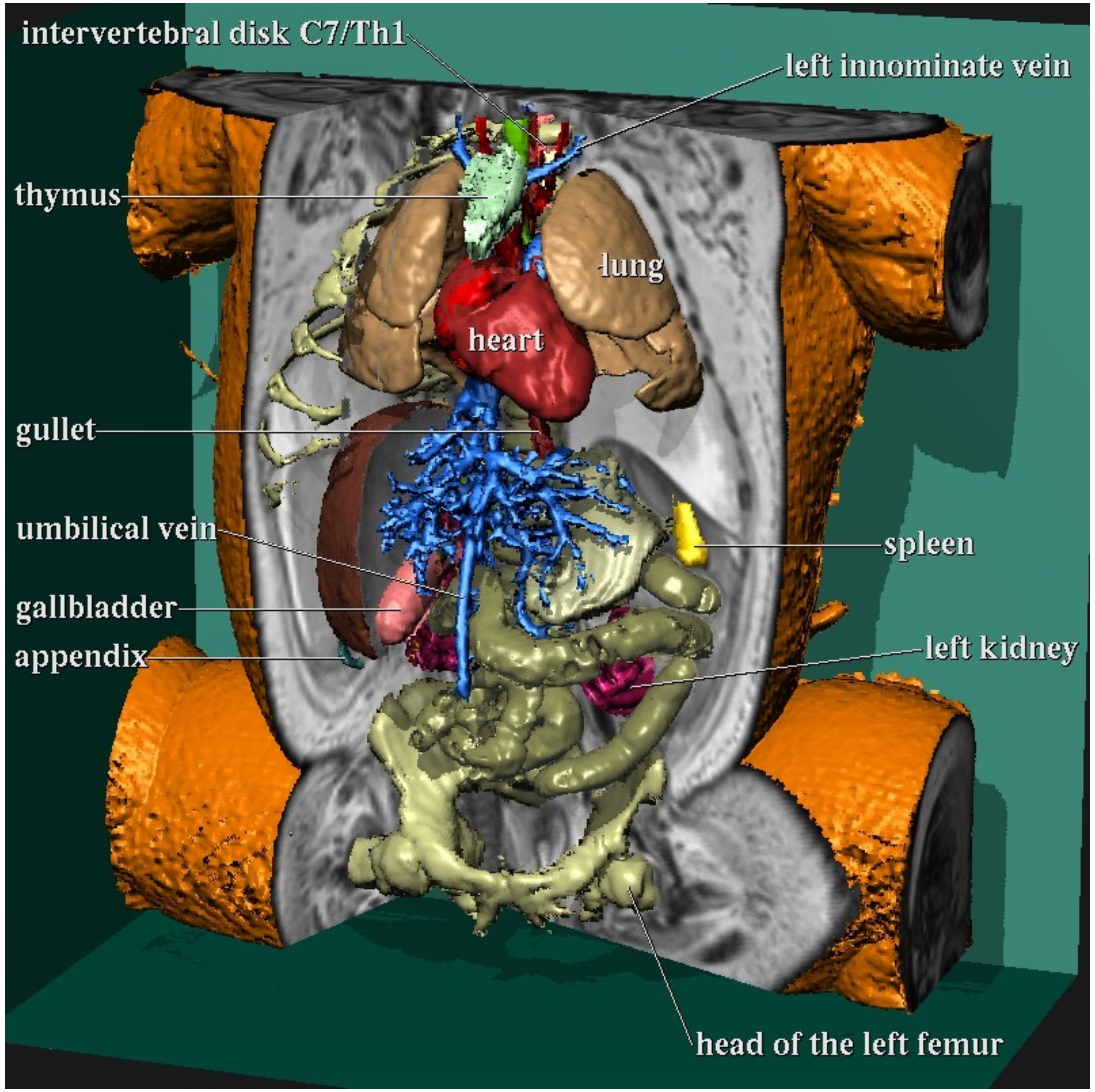
Problem: Segmentation

- Different features with same value
  - Example CT: different organs have similar X-ray absorption
  - Classification cannot be distinguished
- Label grid cells (or grid points) indicating a type
- Segmentation = pre-processing
- Semi-automatic process



→ Vast body of literature in medical imaging, image processing, etc.

Segmentation



Anatomic atlas

Volume Rendering: Examples

- Smoke, fire, clouds, fluid effects etc.

Volume Rendering – Wrap-up

- Direct volume rendering can be implemented on the GPU
  - Ray traversal in fragment shader
  - Efficient scalar value retrieval using 3D textures
- Volume illumination
  - Use e.g. Phong illumination model or secondary rays
  - Often needs normals → gradient of the scalar field

Image source: D. Jönsson et al., CGF 33(19), 2014.