EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Visualization of Scalar Fields

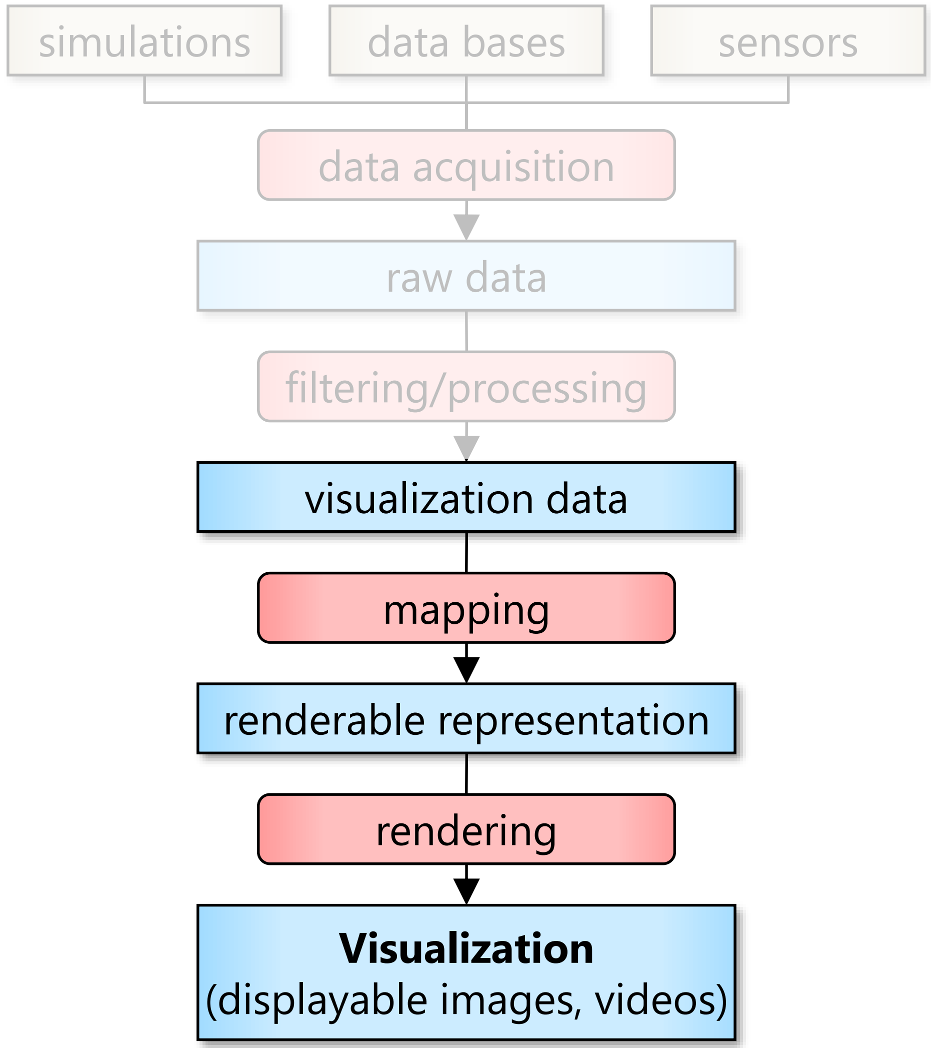
Scientific Visualization – Summer Semester 2021

Jun.-Prof. Dr. **Michael Krone**

Contents

- Basic strategies
- Function plots and height fields
- Isolines
- Color coding
- Volume visualization (overview)
- Classification
- Segmentation
- Volumetric illumination

Focus:
Second step of visualization pipeline



Visualization of Scalar Fields – Basic Strategies

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

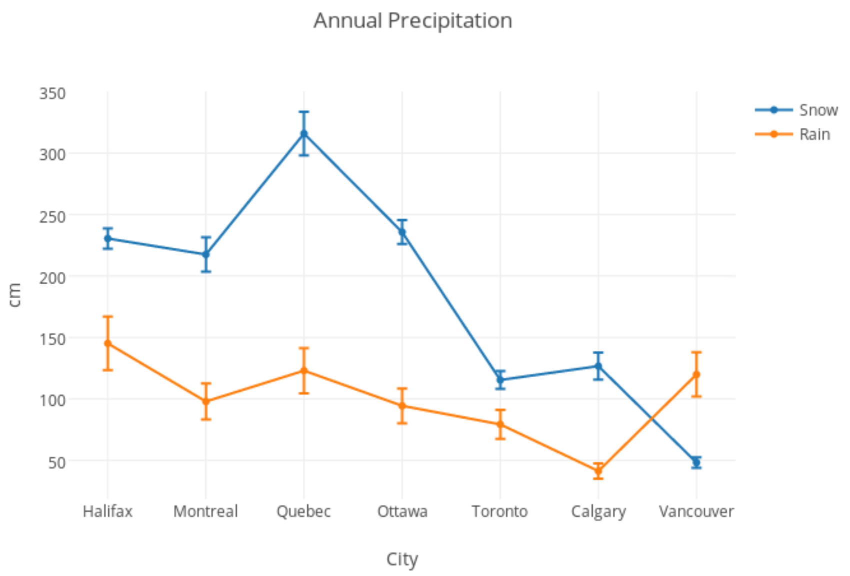
Visualization of Scalar Fields – Basic Strategies

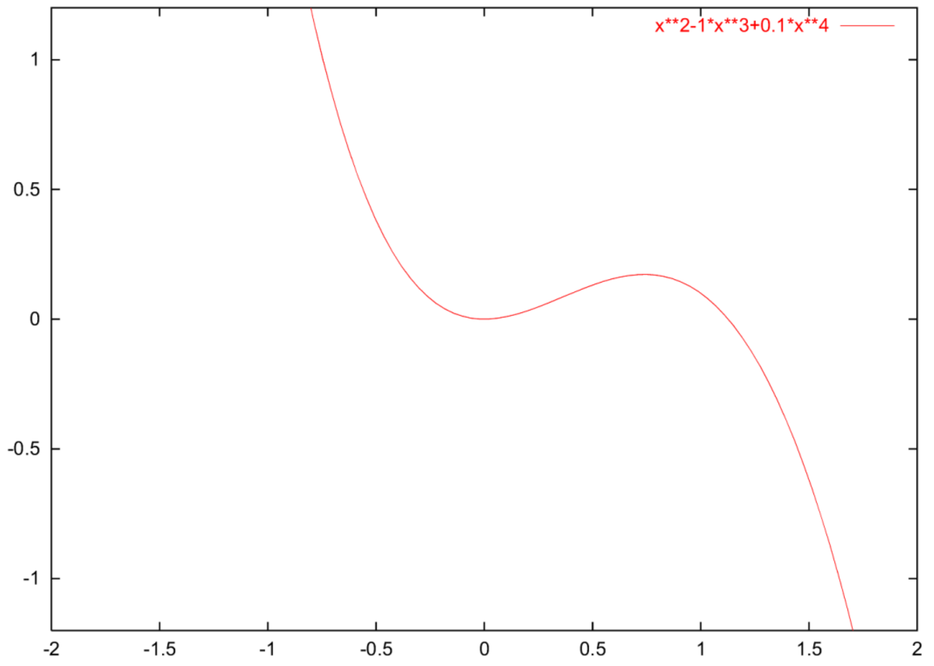
- Mapping to geometry
  - Function plots
  - Height fields
  - Isolines and isosurfaces
- Color coding
- Specific techniques for 3D data
  - Indirect volume visualization
  - Direct volume visualization
  - Slicing

- **Visualization method depends heavily on dimensionality of domain**

Function Plots & Height Fields

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



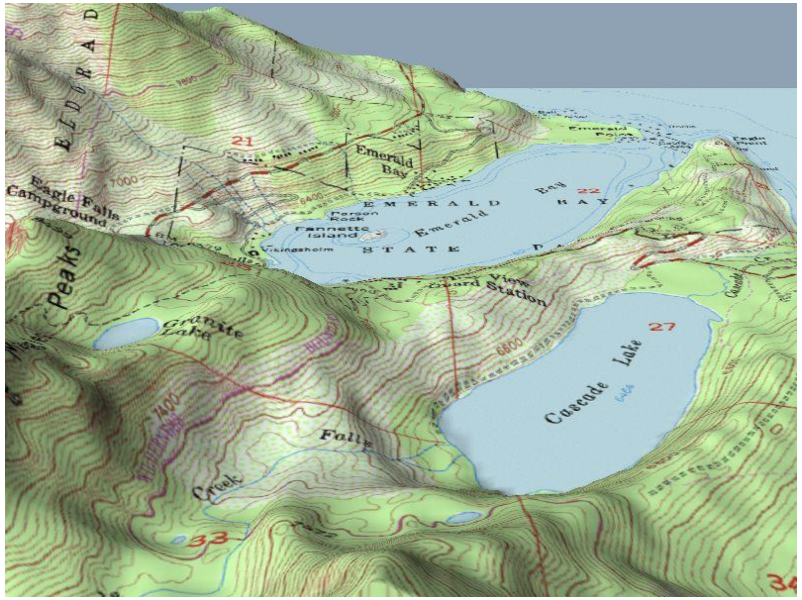


Gnuplot example

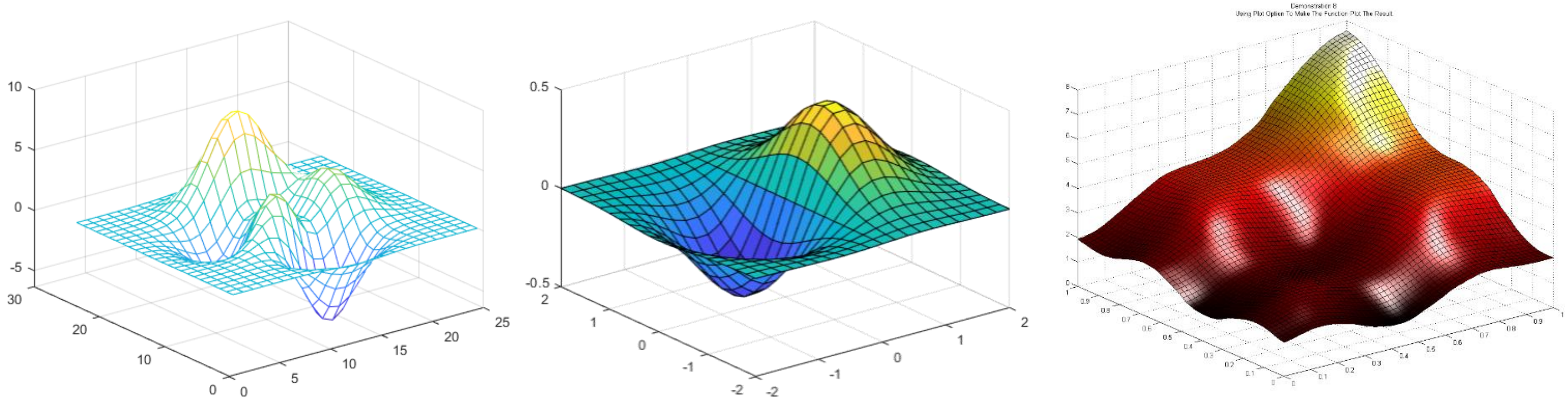
Function Plots & Height Fields

- Function plot for a 2D scalar field
  - Points $\{(s, t, f(s,t)) | (s,t) \in \mathbb{R}^2\}$
  - 2D manifold: surface

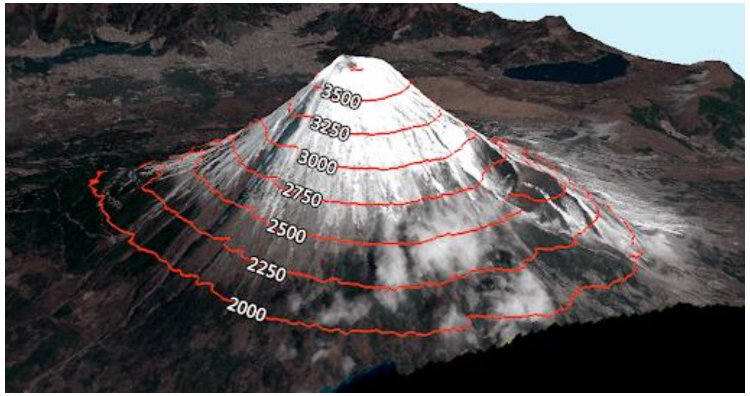
- Surface representations
  - Wireframe
  - Hidden lines
  - Shaded surface



Terrain Rendering

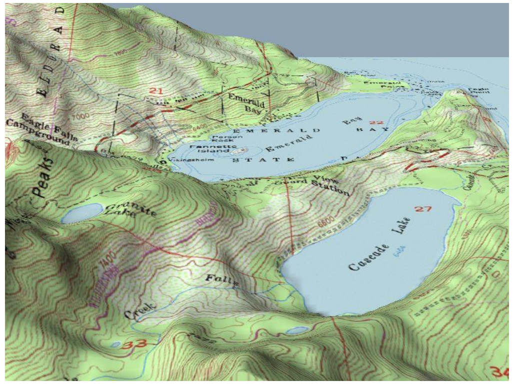


Function Plots



Isolines / Contours

Isolines



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

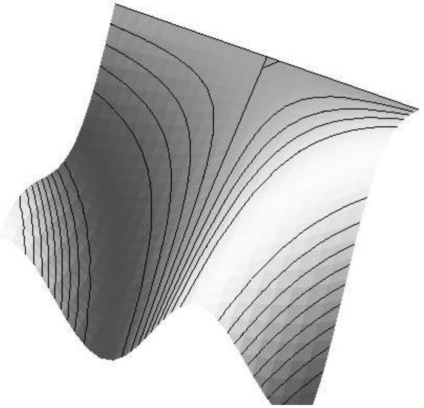
- If $f()$ is differentiable and $grad(f) \neq 0$,
  then isolines are curves

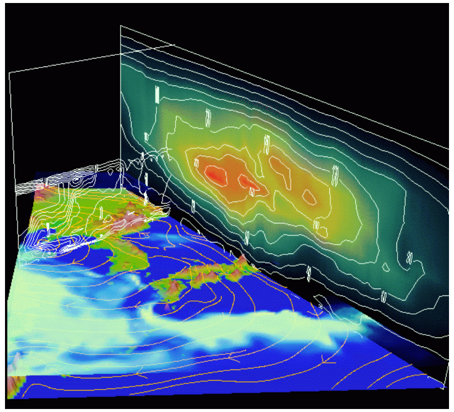
- Contour lines

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

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

- Contour lines





Isolines: Pixel-by-Pixel Contouring

- Straightforward approach: scan all pixels for equivalence with isovalue
- Input
  - $f: (1, \ldots, x_{max}) \times (1, \ldots, y_{max}) \to \mathbb{R}$
  - Isovalues $c_1, \ldots, c_n$ and isocolors $w_1, \ldots, w_n$
- Algorithm:

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

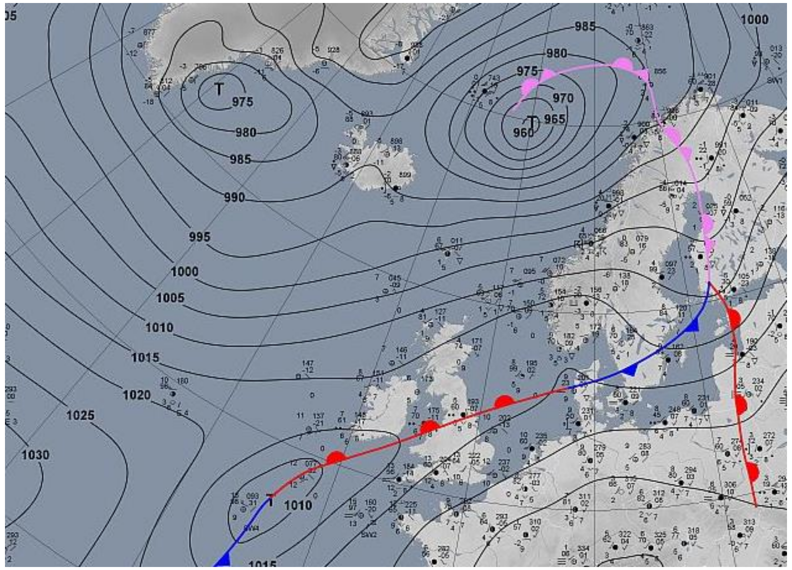
- **Problem:** Isoline can be missed if the gradient of $f()$ is too large (despite range ε), or can be too wide if the gradient is too small

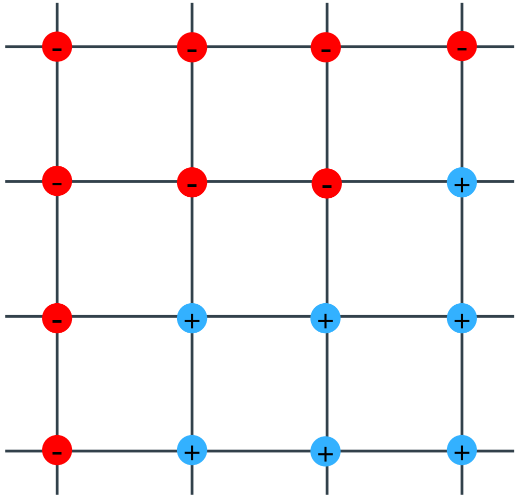
Contours

- Contours: $\{\boldsymbol{x} \in \mathbb{R}^n | f(\boldsymbol{x}) = c\}$
  - Closed (including boundaries), orientable, non-intersecting manifolds
- Isolines (n=2), Isosurfaces (n=3)
- Fast approach for cartesian/rectilinear grids using lookup table:
  - Cell-based approach (contour segments for each cell separately)
  - Lookup table provides part of contour (connectivity) for each cell
  - *Marching Squares* (n=2), *Marching Cubes* (n=3)
  - Also applicable to curvilinear/structured grids (quadrilaterals, hexahedra)
    - Using local coordinates
  - Other variants for unstructured grids

Isolines

- Marching Squares
  - Scalar values are given at each node $f \leftrightarrow f_{ij}$
  - Take into account the interpolation within cells
  - Isolines cannot be missed
  - Cells are processed independently of each other



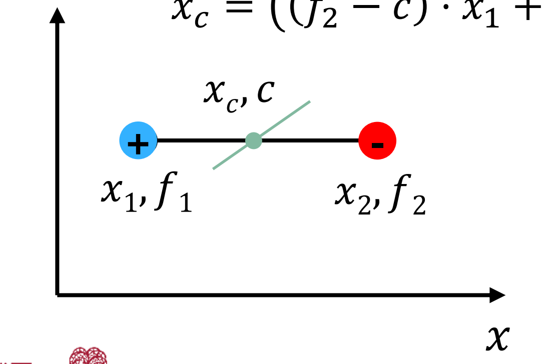


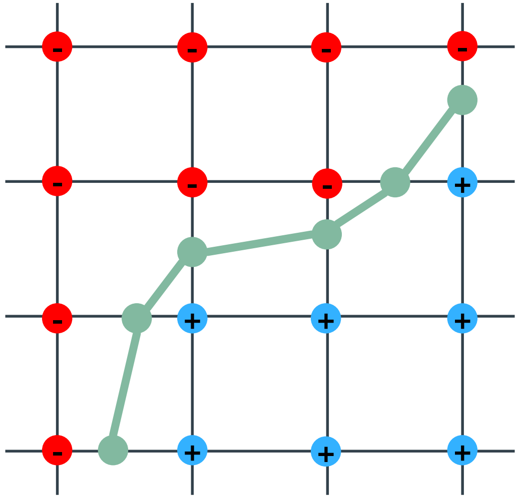
Isolines

- Which cells will be intersected?
  - Initially mark all nodes by + or – , depending on the conditions $f_{ij} \geq c$, $f_{ij} < c$
- No isoline segments inside cells that have same sign at all nodes
  - So we only have to determine the cells with different signs
  - Use look-up table for respective case, depending on marked nodes
  - Find exact position of intersection on edge by linear interpolation

$$f_1 \geq c > f_2$$

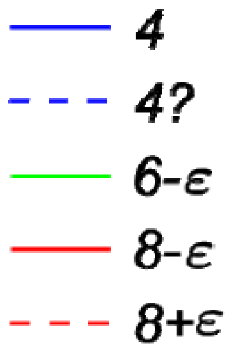
$$x_c = \big((f_2 - c) \cdot x_1 + (c - f_1) \cdot x_2\big)/(f_2 - f_1)$$



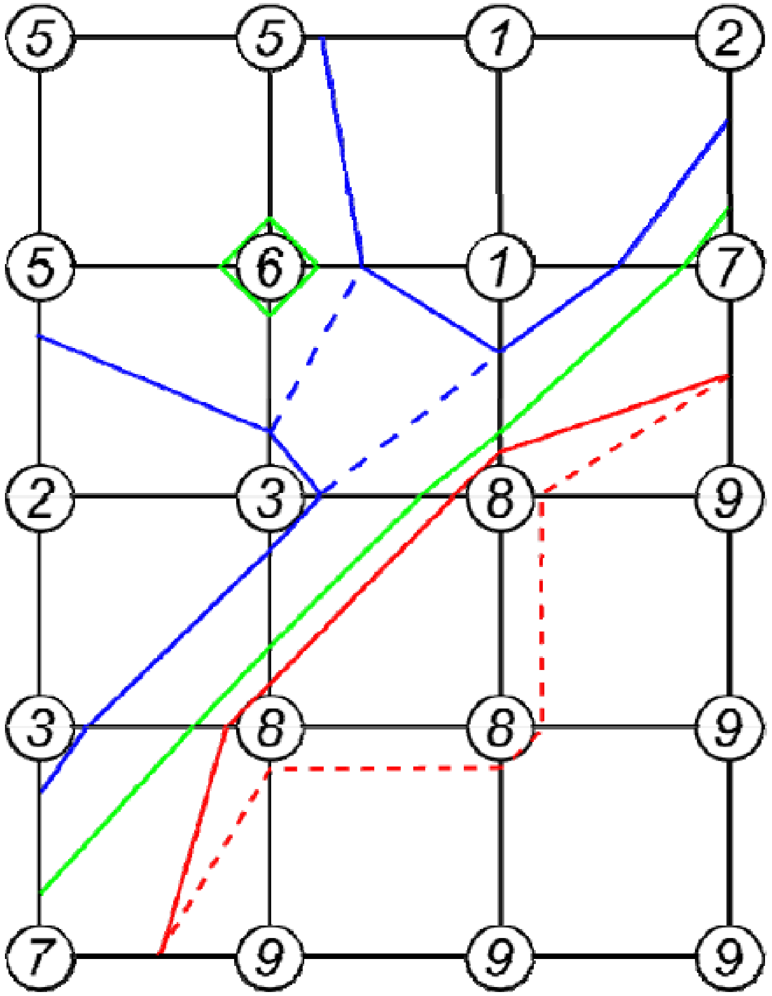


Isolines: Degenerate Cases

- Contour levels:

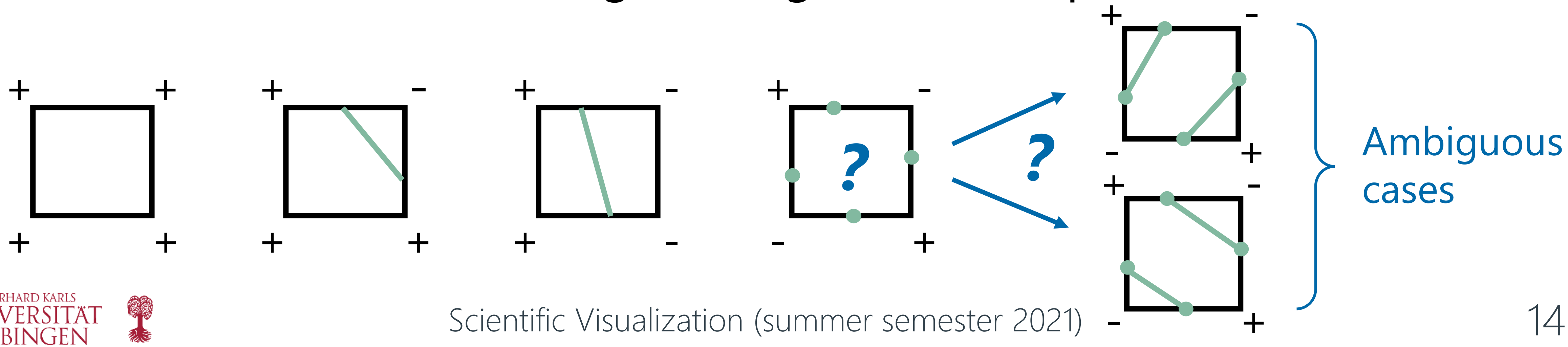


- 2 types of degeneracies
  - Isolated points (c=6)
    - Two possible solutions
  - Flat regions (c=8)
    - **Solution:** perturbation
    - If node value = c use c-ε



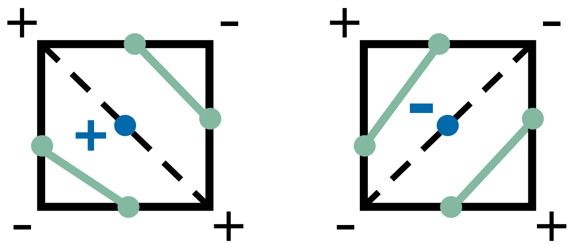
Isolines – Efficient Implementation

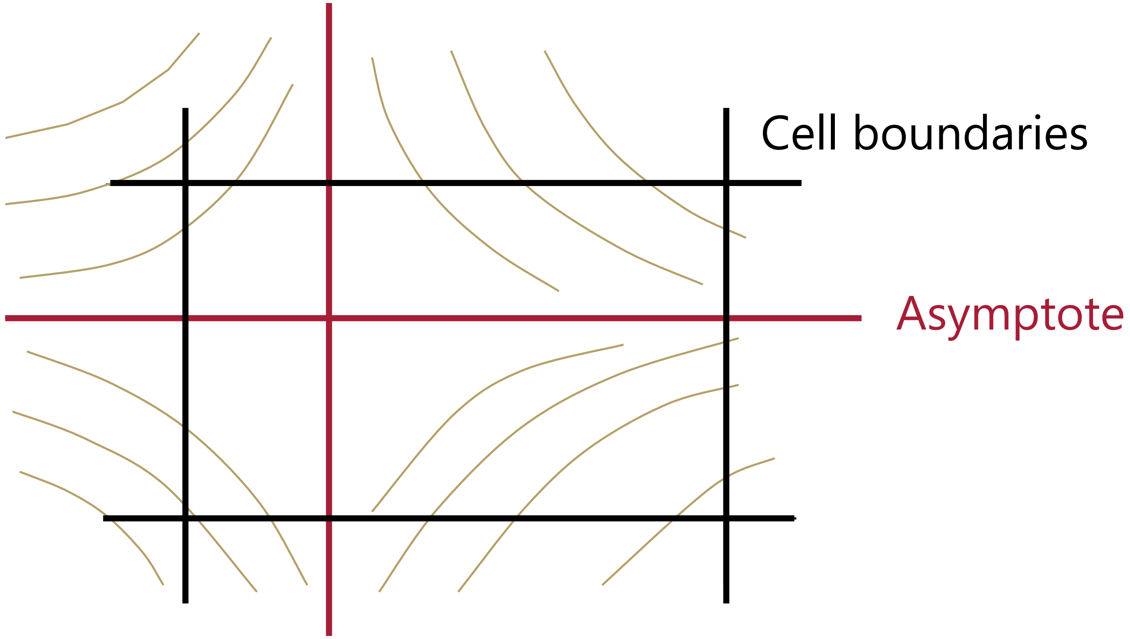
- Construct a binary number (4 bits = 16 cases) from signs at 4 nodes
  - Use a precomputed lookup table (array of 16 cases telling how to connect edge intersections)
- Symmetries: rotation, reflection, change + ↔ -
  - Only 4 different cases (classes) of combinations of signs
- Based on information from lookup table, compute exact intersections between isoline and cell edges using linear interpolation



Isolines

- We can distinguish the ambiguous cases by a decider
- Asymptotic decider
  - Consider the bilinear interpolant within a cell
  - The true isolines within a cell are hyperbolas





Isolines

- Interpolate the function bilinearly

$$f(x,y) = f_{i,j}(1-x)(1-y) + f_{i+1,j}x(1-y) + f_{i,j+1}(1-x)y + f_{i+1j+1}xy$$

$$f(x,y) = Axy + Bx + Cy + D$$

- If A=0, contour equation is:

$$c = Bx + Cy + D$$

→ contours are straight lines

- If A≠0, contour equation is:

$$c = A(x + \frac{C}{A})(y + \frac{B}{A}) + D - \frac{BC}{A}$$

→ contours are hyperbola except for special level $c = D - \frac{BC}{A}$

Isolines

- Contour equation for this special level:

$$0 = A\left(x + \frac{C}{A}\right)\left(y + \frac{B}{A}\right)$$

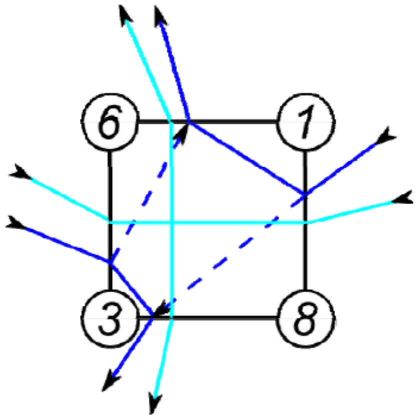
→ contours are axis-aligned straight lines $x = -\frac{C}{A}$ and $y = -\frac{B}{A}$

  - $D - \frac{BC}{A}$ is value at intersection of asymptotes (saddle point)

- Example
  - Contour equation:

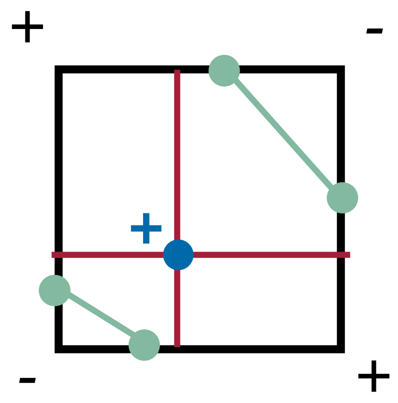
$$c = -10(x - 0.3)(y - 0.5) + 4.5$$

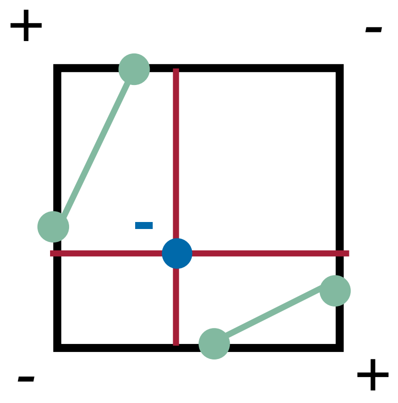
  - Special level $c = 4.5$
  - Saddle point at $(0.3, 0.5)$



Isolines

- If $D - \frac{BC}{A} > c$ we choose the left case, otherwise we choose the right one



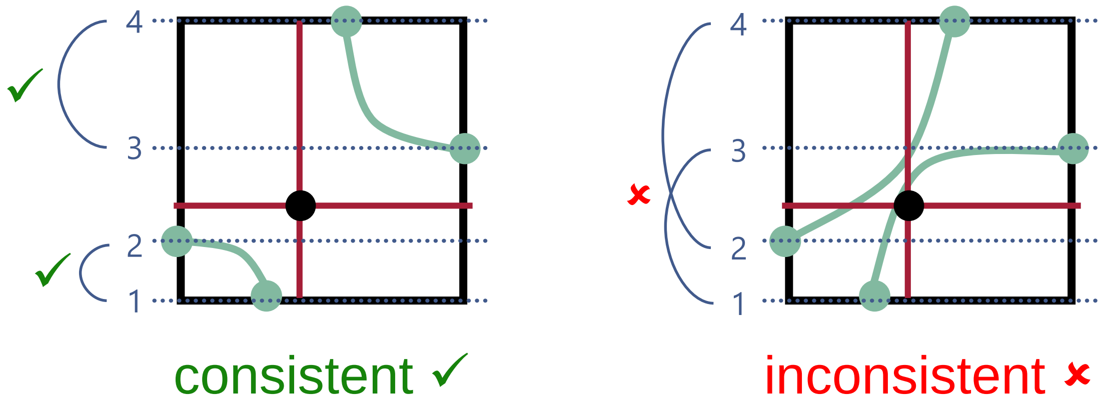


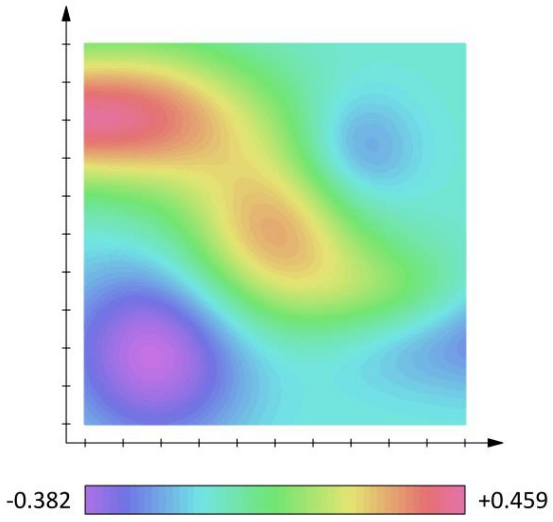
Isolines

- Explicit transformation of $f()$ to

$$f(x, y) = A(x + \frac{C}{A})(y + \frac{B}{A}) + D - \frac{BC}{A}$$

  can be avoided
- **Idea:** investigate order of intersection points either along x or y axis
  - Build pairs of first two and last two intersections





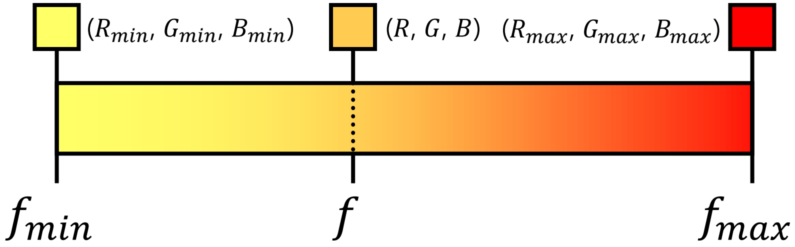
Color Coding

Image Source: https://en.wikipedia.org/wiki/Scalar_field

Color Coding

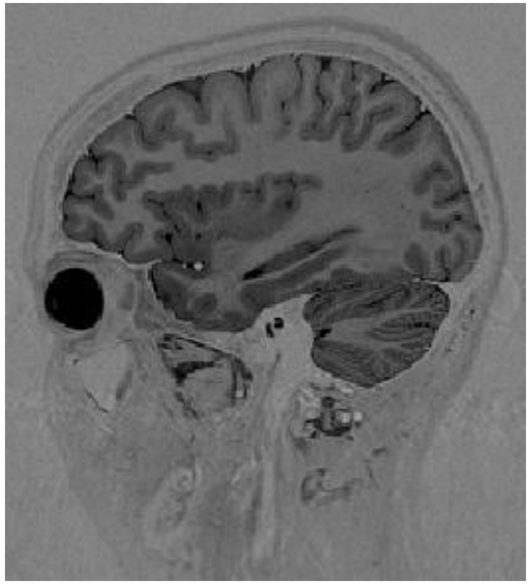
- Easy to apply to 1D and 2D scalar fields
  - Map color to each pixel on 1D or 2D image
- **Example:** Transfer function *(recapitulation of last chapter)*
  - Linear transfer function for color coding
    - Specify colors for f_{min} and f_{max} → $(R_{min}, G_{min}, B_{min})$ and $(R_{max}, G_{max}, B_{max})$
    - Linearly interpolate between them:

$$f \rightarrow \frac{f_{max} - f}{f_{max} - f_{min}}(R_{min}, G_{min}, B_{min}) + \frac{f - f_{min}}{f_{max} - f_{min}}(R_{max}, G_{max}, B_{max})$$

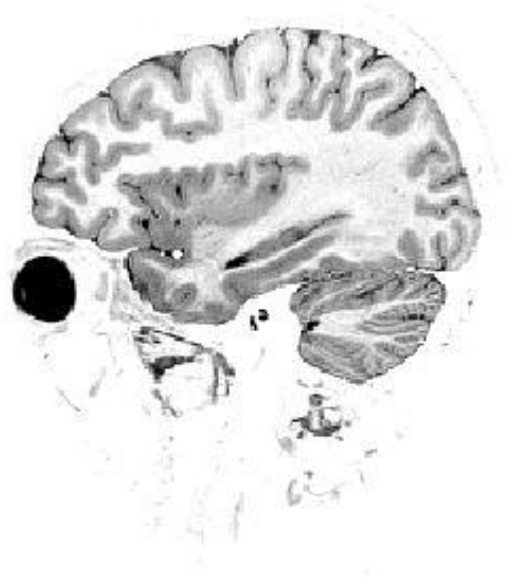


Color Coding

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



Original



Brain

Tissue

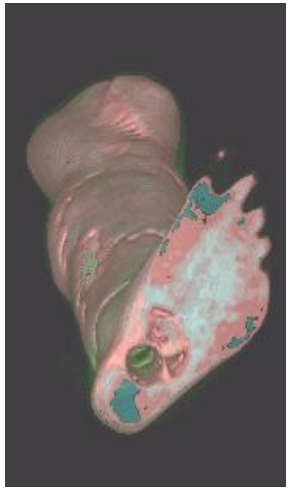
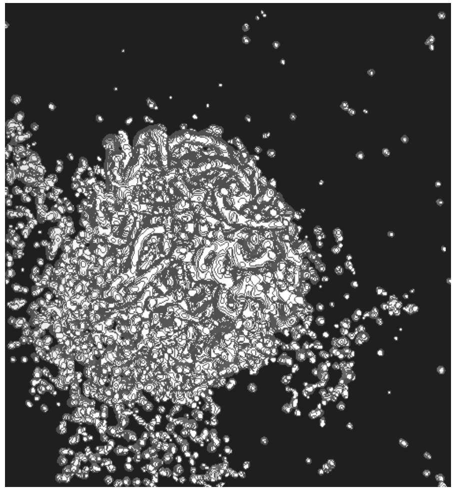
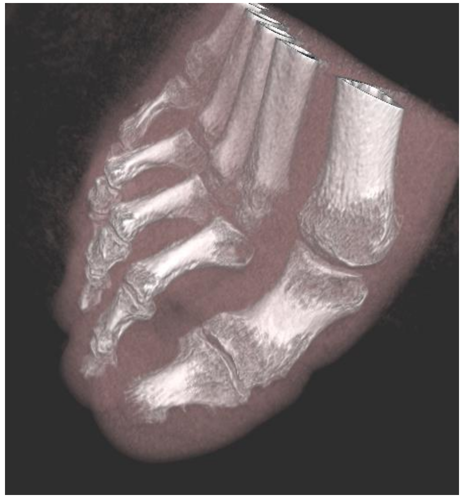
Volume Visualization

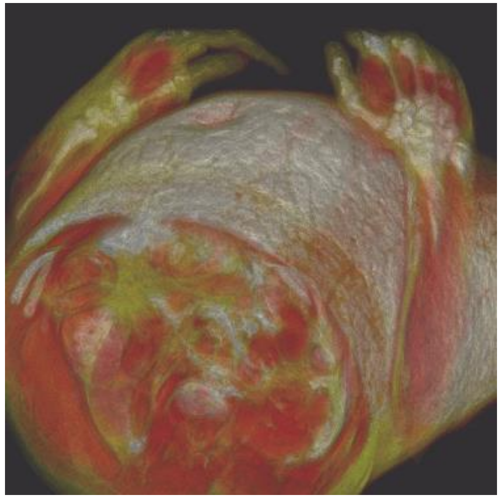
Volume Visualization

- Scalar volume data

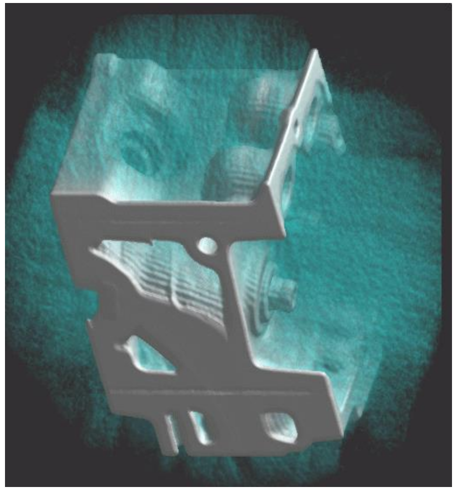
$$\Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}$$

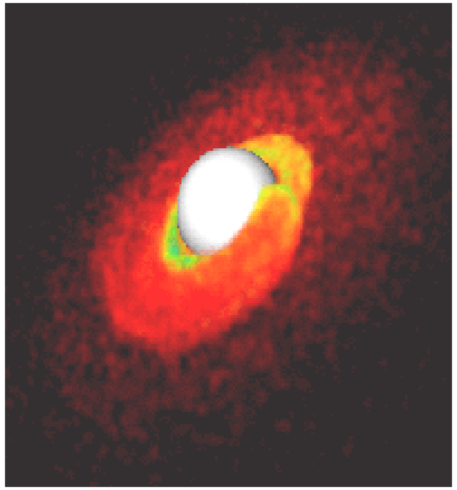
- **Example:** Medical Imaging (CT, MRI, confocal microscopy, ultrasound, etc.)

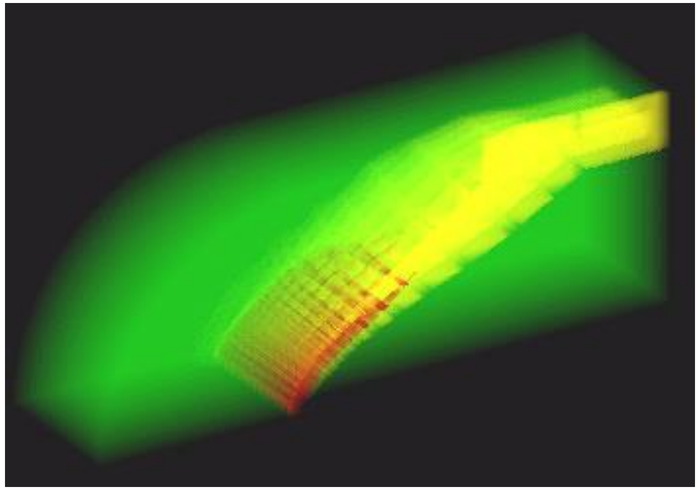





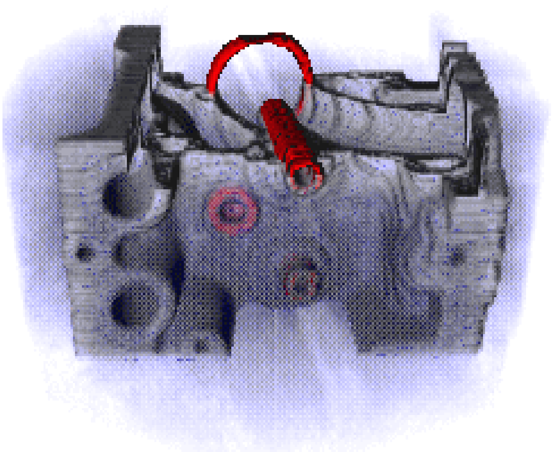


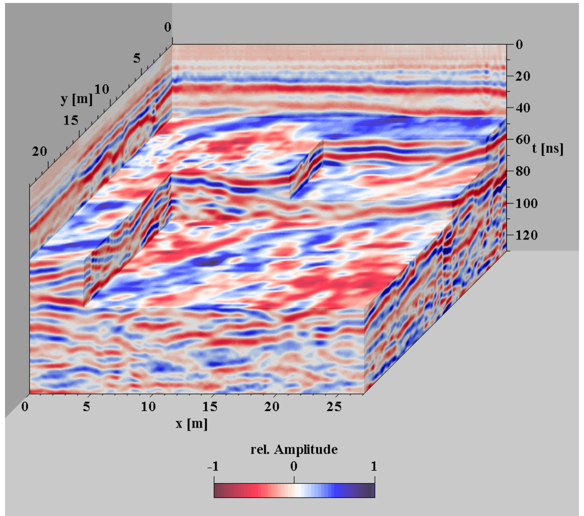
Volume Visualization

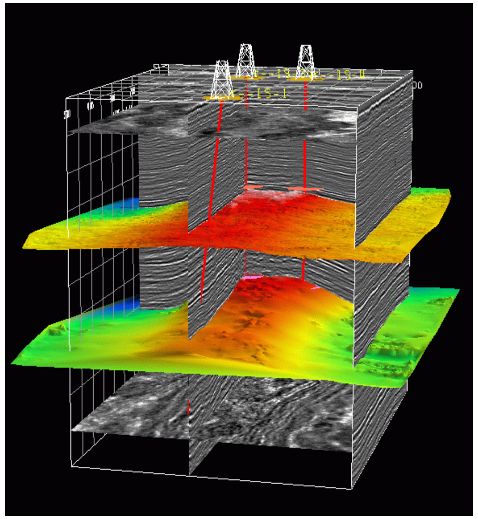








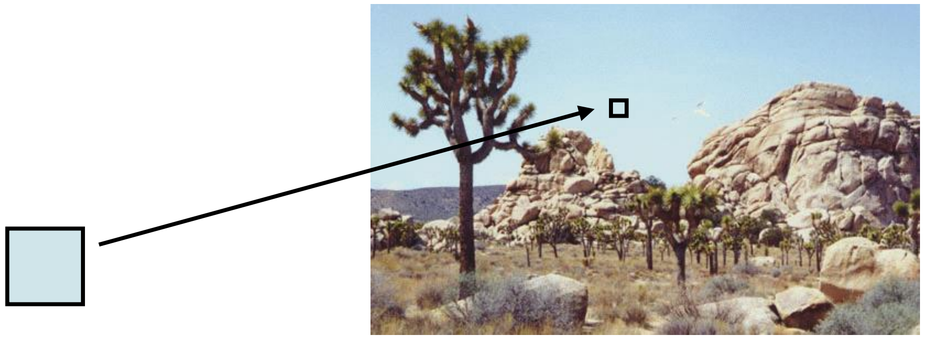




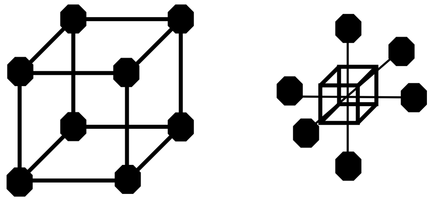
Volume Visualization

- Representation of scalar 3D data set

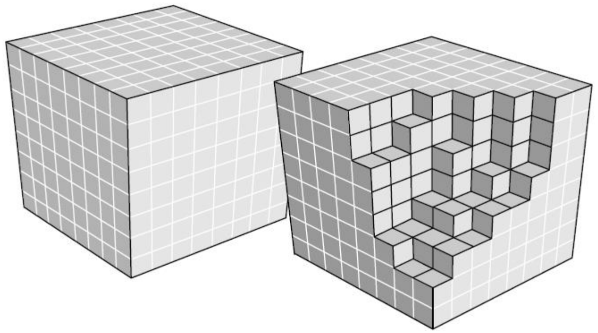
$$\Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}$$



- Analogy in 2D: **pixel** (picture element)



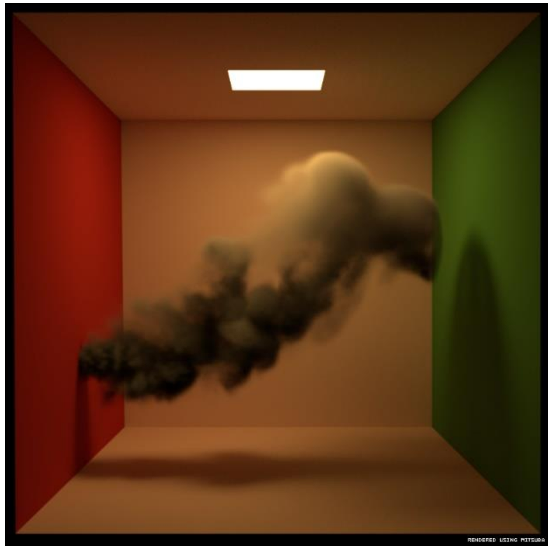
- **Voxel** (volume element), with two interpretations:
  - Cell-based:
    - Values are constant within a region around a grid point
  - Node-based (dual grid):
    - Values between grid points are obtained by interpolation



Volume Visualization

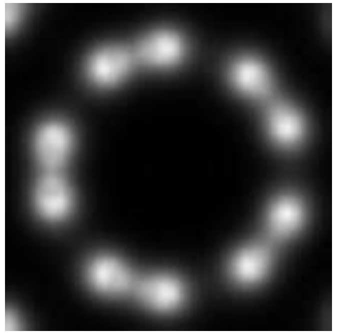
- Challenges
  - Essential information “hidden” in the interior
  - Occlusion?
  - Often data sets cannot be described by geometric (surface) representation
    → fire, clouds, gaseous phenomena,…

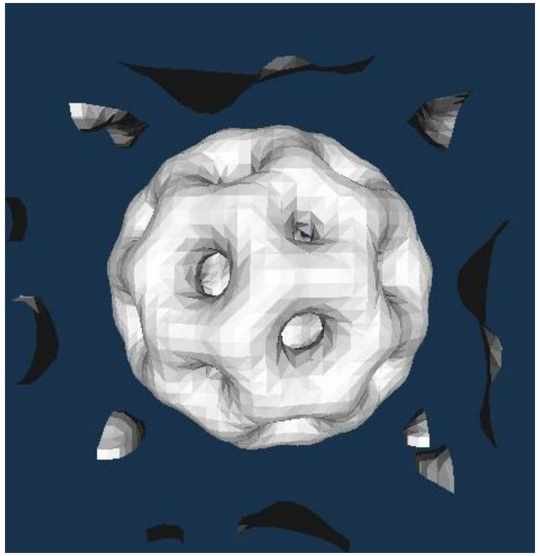


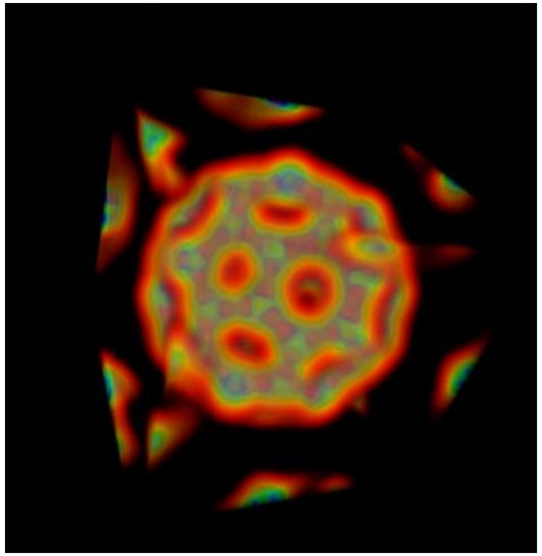



Volume Visualization

- Volume rendering approaches
  - Techniques for 2D scalar fields
    - Transform 3D data set to 2D
    - Then apply 2D methods
  - Indirect volume rendering techniques (e.g. isosurfaces, 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

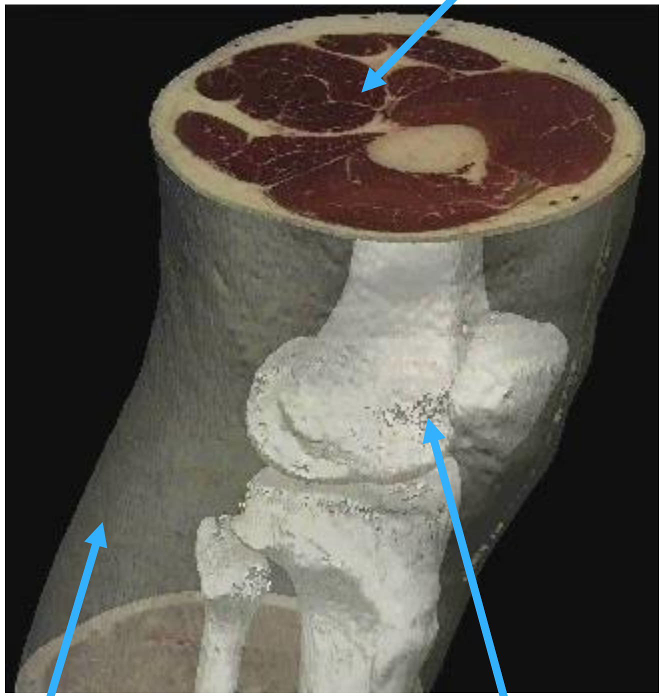




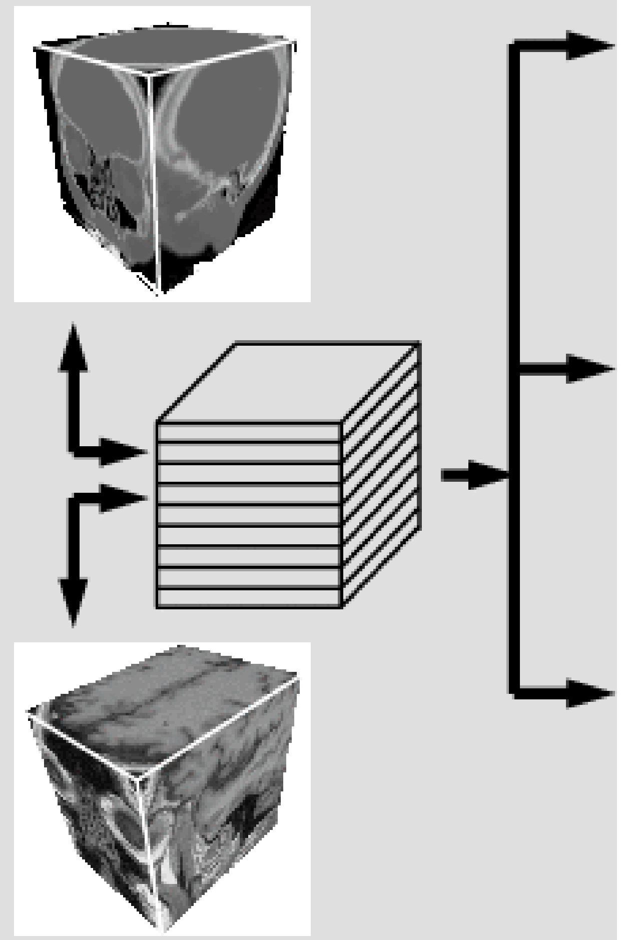


Volume Visualization

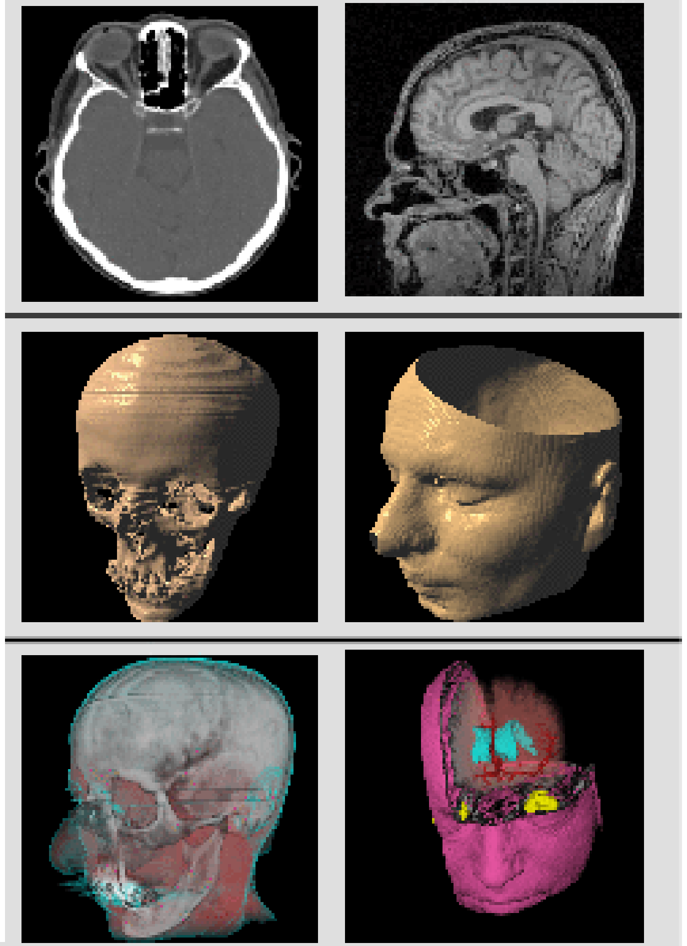
- **Slicing:**
  Display the volume data, mapped to colors, on a slice plane
- **Isosurfacing:**
  Generate opaque/semi-opaque surfaces
- **Transparency effects:**
  Volume material attenuates reflected or emitted light



Volume Visualization



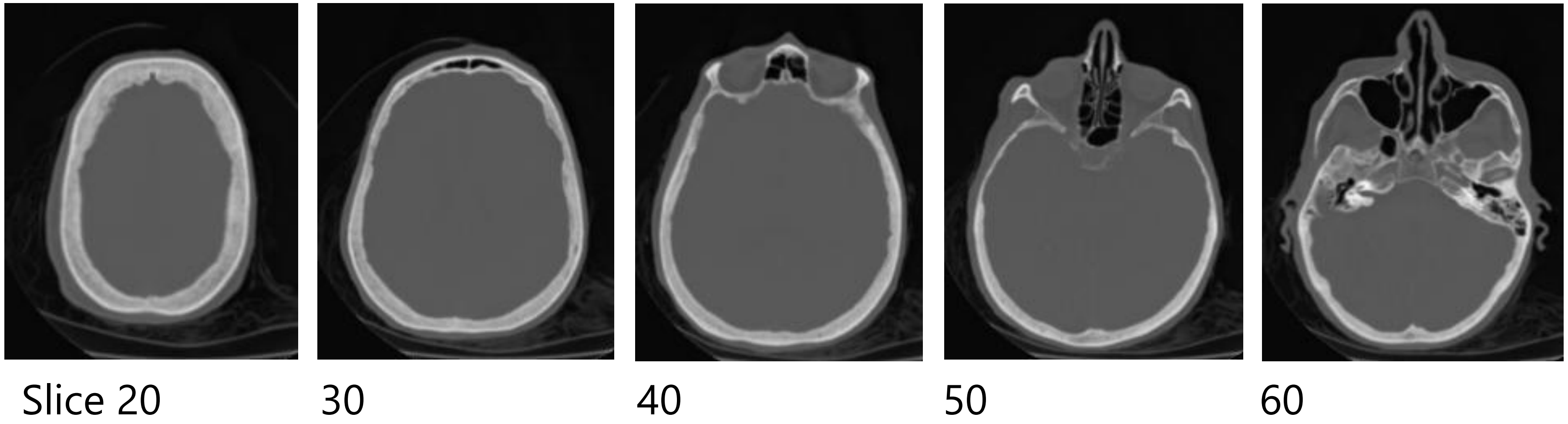
- 2D visualization
  slice images
  (or multi-planar
  reformation, MPR)
- ***Indirect***
  3D visualization
  isosurfaces
  (or surface-shaded
  display SSD)
- ***Direct***
  3D visualization
  (direct volume
  rendering DVR)



Volume Visualization

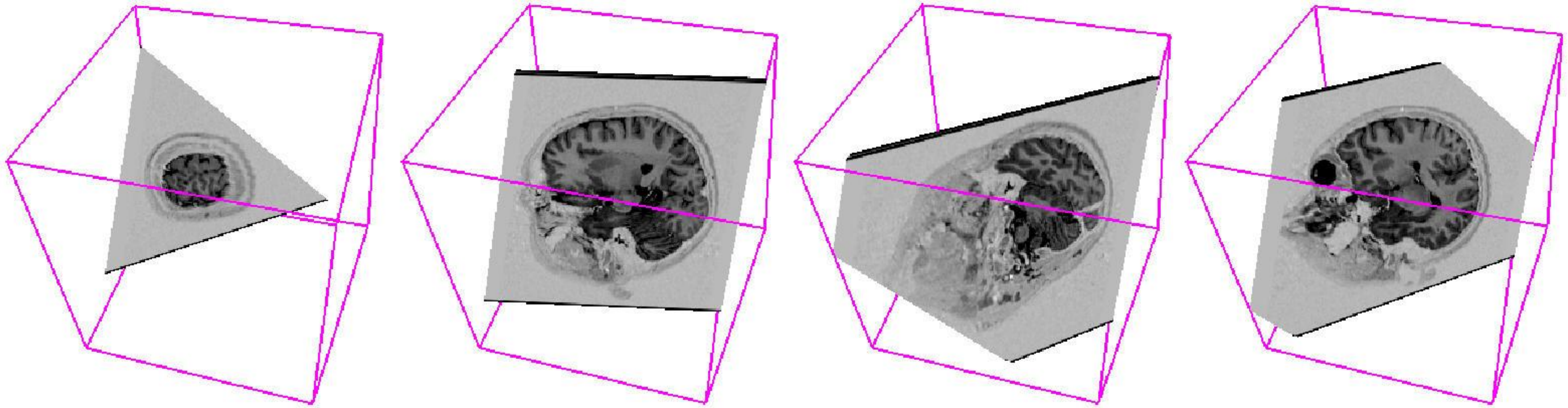
- 2D approach: Orthogonal slicing
  - Interactively resample the data on slices perpendicular to x-,y-,z-axis
  - Use visualization techniques for 2D scalar fields
    - Color coding
    - Isolines
    - Height fields

CT data set

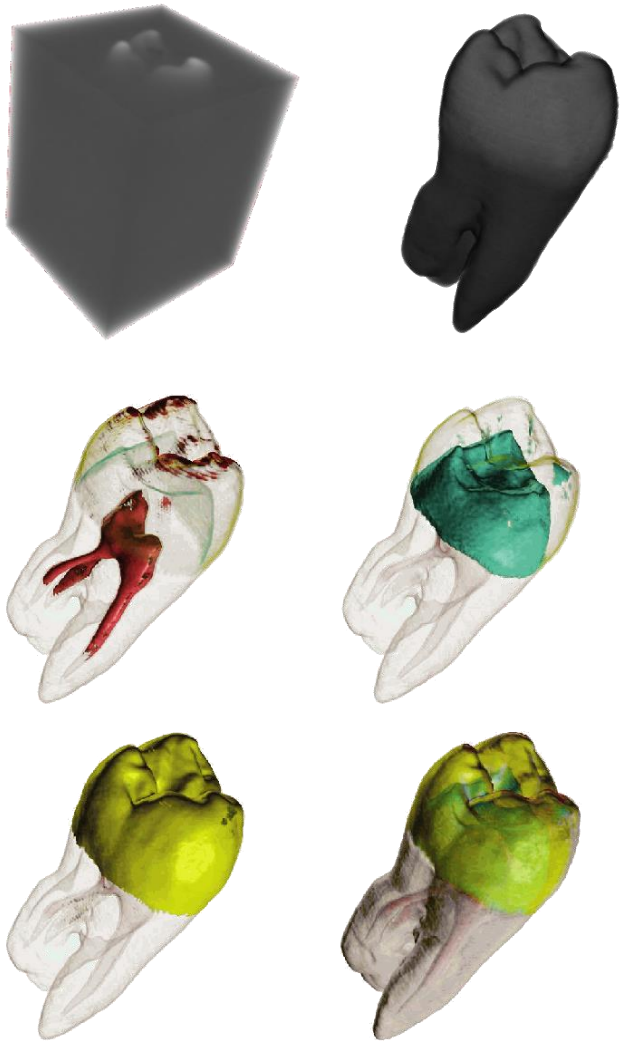


Volume Visualization

- Alternative: Oblique slicing (MPR, multiplanar reformation)
  - Resample the data on arbitrarily oriented slices
    - Resampling on CPU or on graphics hardware (trilinear interpolation)
    - Exploit 3D texture mapping functionality
      - Store volume in 3D texture
      - Compute sectional polygon (clip plane with volume bounding box)
      - Render textured polygon



Classification



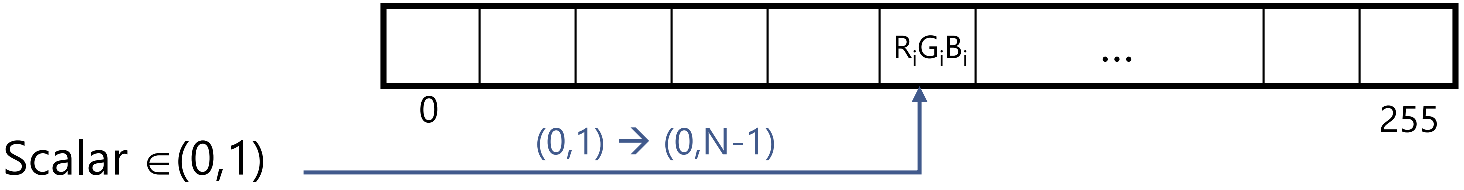
- Goals and issues:
  - Empowers user to select "structures"
  - Extract important features of the data set
  - Classification is non-trivial
  - Histogram can be a useful hint
- Usually needed for volume visualization
- Standard approach: Transfer function
  - Color table for volume visualization
  - Maps raw voxel value to presentable entities: color, intensity, opacity, etc.
  - Often requires interactive manipulation of transfer function

Classification

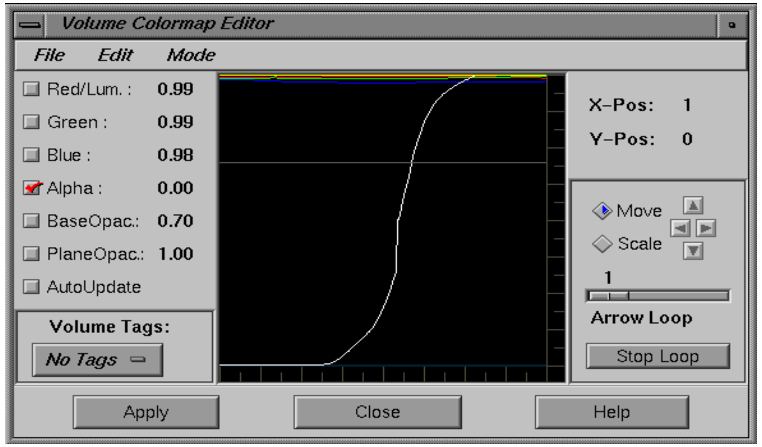
- Most widely used approach for transfer functions:
  - Assign to each scalar value a different color value and opacity
  - Assignment via transfer function T

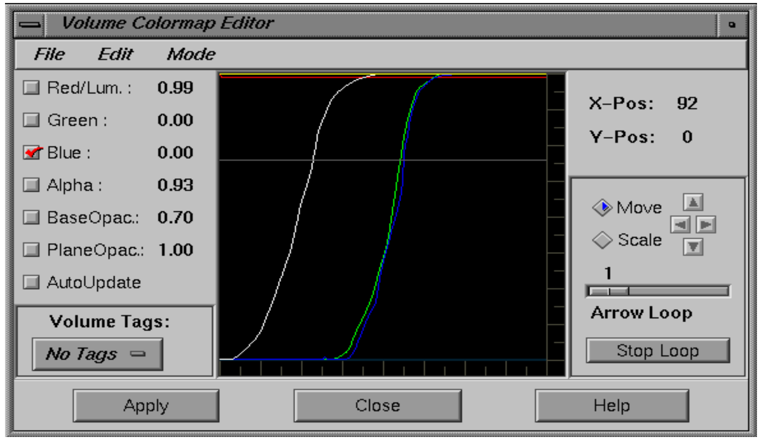
$$T : scalarvalue \rightarrow colorvalue$$

  - Common choice for color representation: RGBA
  - Alpha value is very important, describes opacity
    - A=0.0: fully transparent; A=1.0: opaque
  - Can be stored inside a color lookup table (LUT)
  - On-the-fly update of LUT

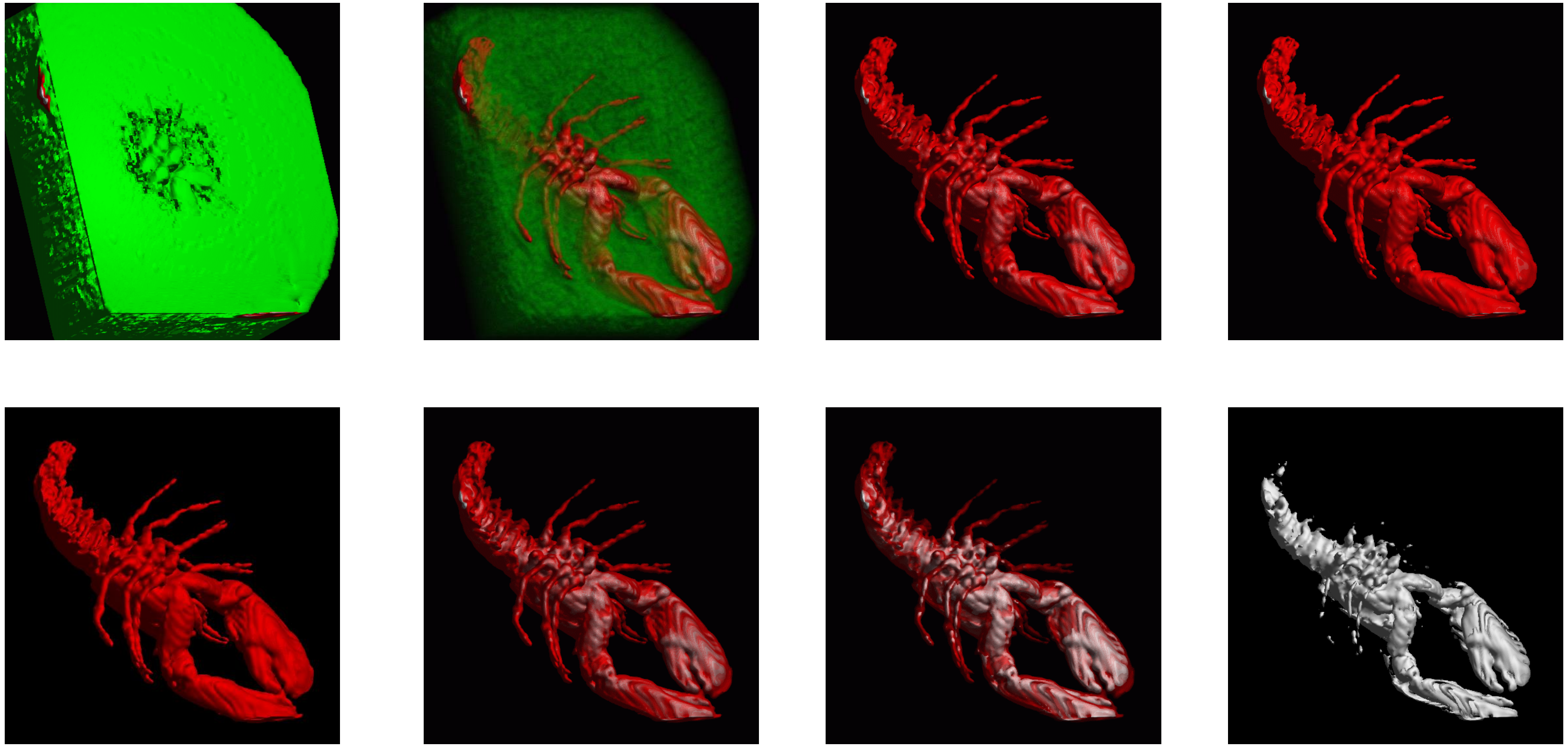


Classification



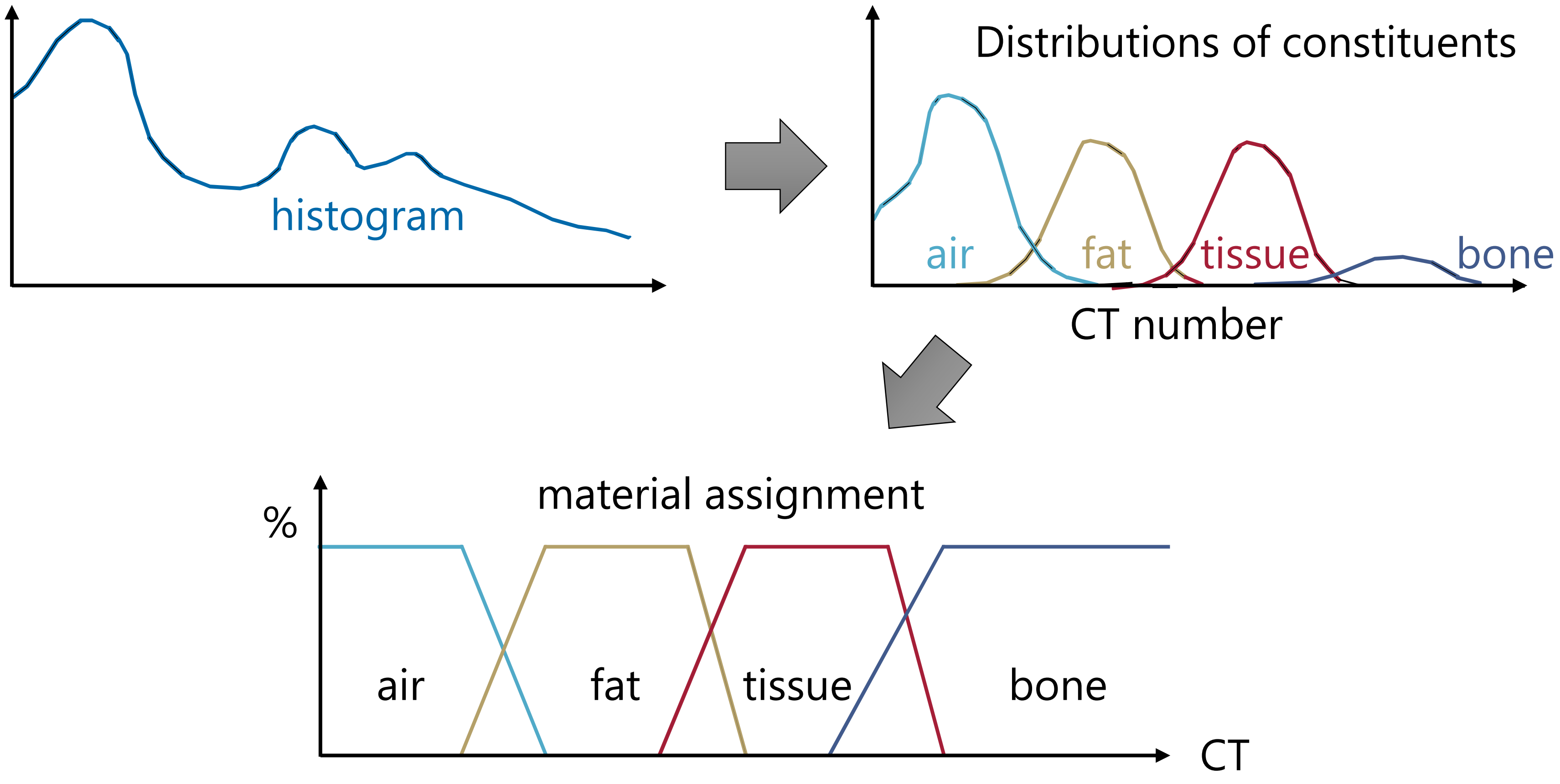


Classification



Classification

- Heuristic approach, based on measurements of many data sets



Classification

- Hounsfield units (HU) for CT data sets
  - Describes x-ray attenuation, i.e., density of material
  - 12-bit CT-measurements
  - Range of values from -1024 to +3071 HU
  - Typical values:
    - Air: -1024
    - Fat: -100 to -20
    - Water: 0
    - Soft tissue such as muscle: +20 to +80
    - Bone: > +500
  - For visualization, 12 bits are often reduced to 8 bits by windowing (loss of dynamic range)

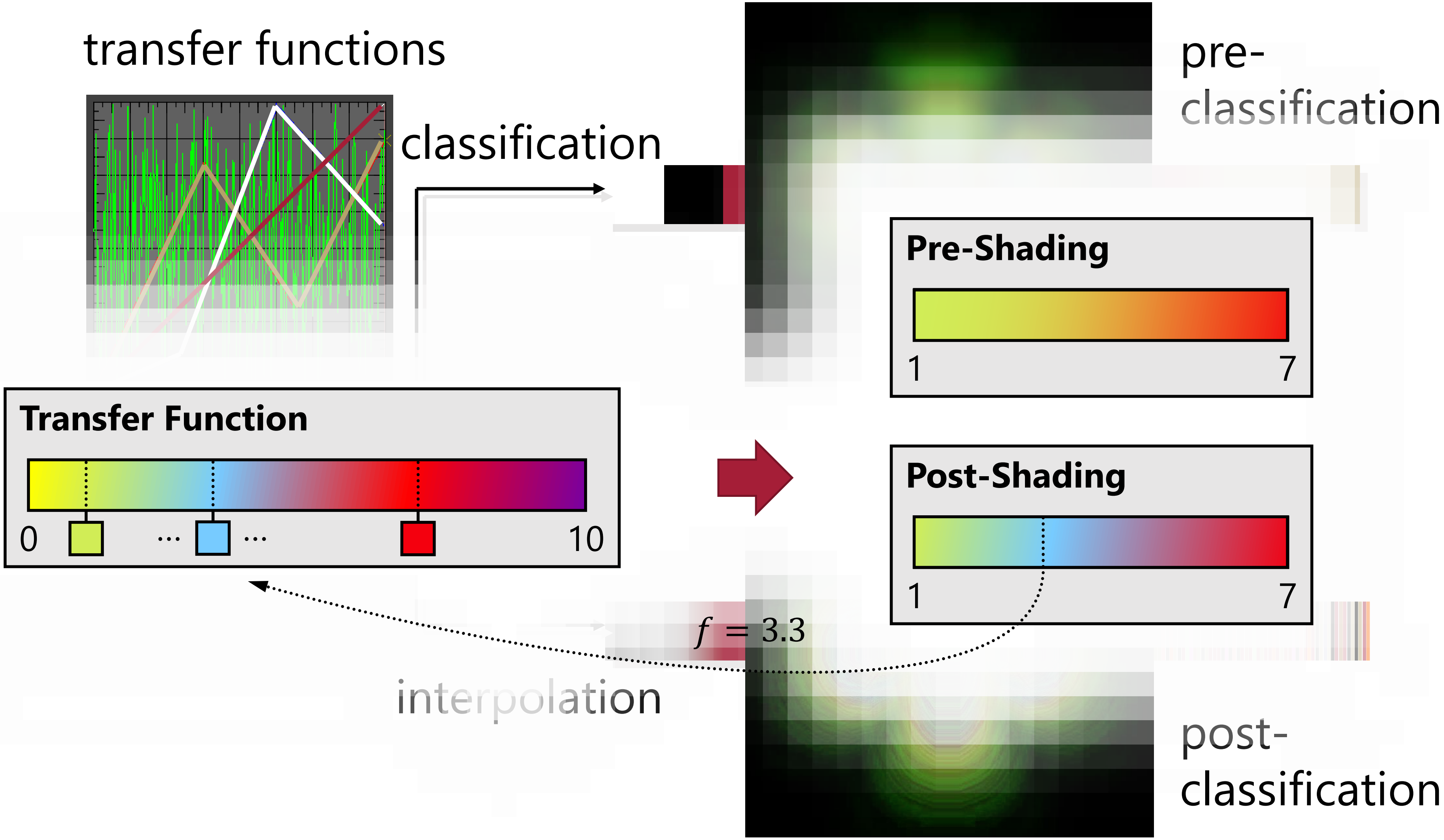
Sir Godfrey N. Hounsfield (1919 – 2004)

→ 1979 Nobel Prize for Medicine (together with A. M. Cormack for developing X-ray CT)

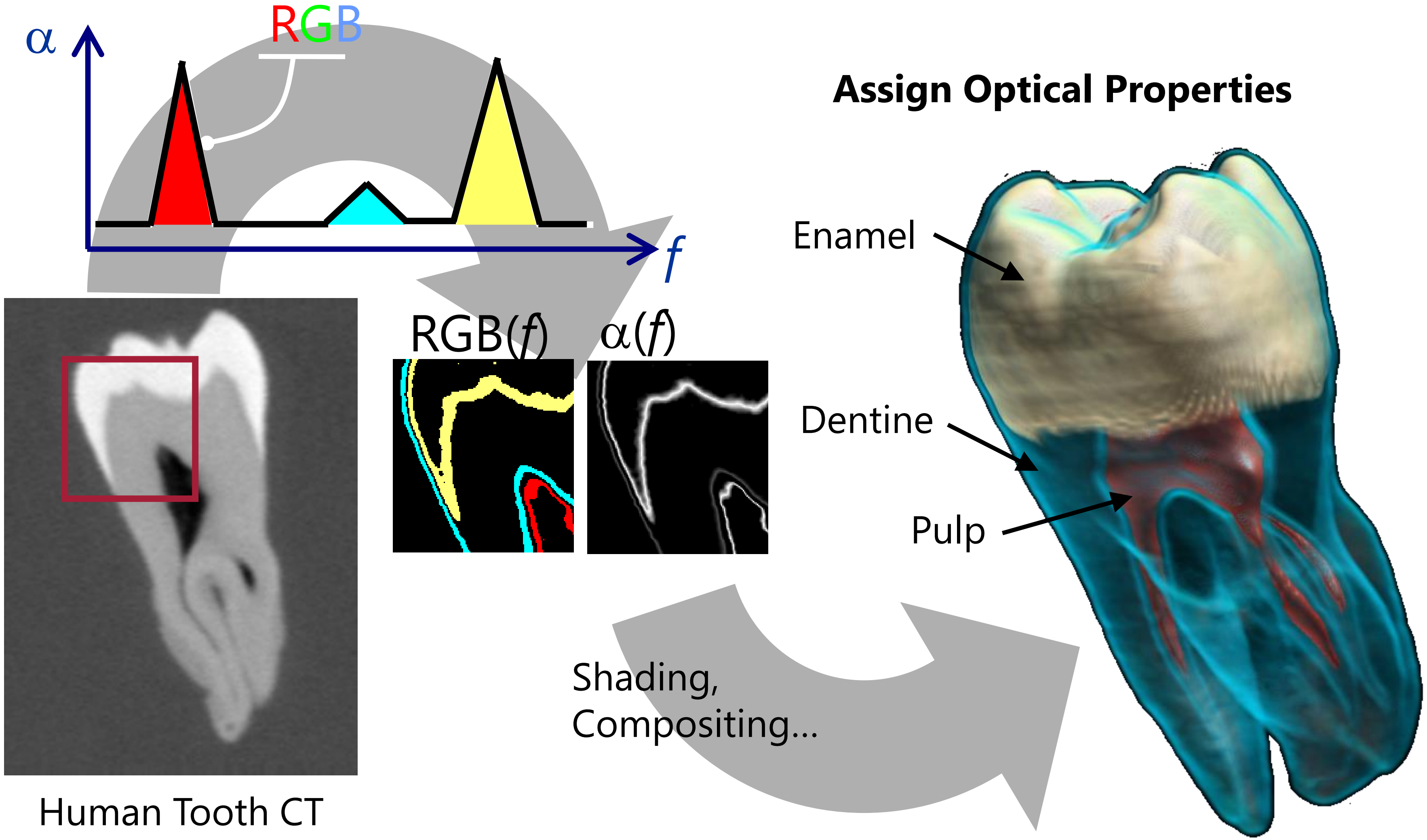
Classification

- Pre-shading
  - Assign color values to original function values
  - Interpolate between color values
- Post-shading
  - Interpolate between scalar values
  - Assign color values to interpolated scalar values

Classification

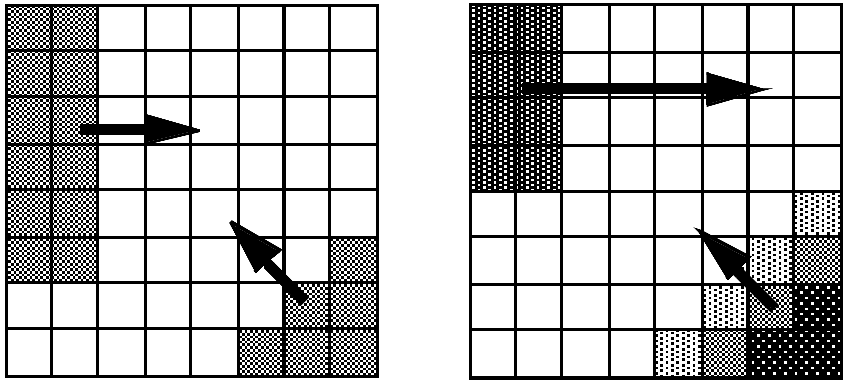


Classification



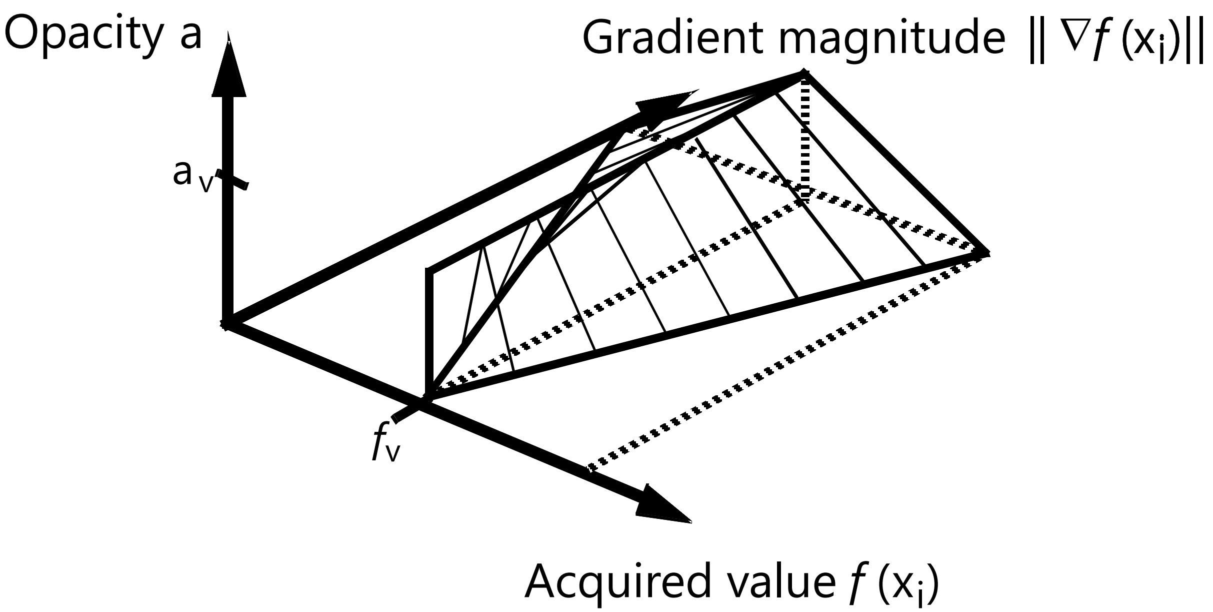
Classification

- Regions of “change” (boundaries) are often most important
- Feature extraction - High value of opacity in regions of change
  - Homogeneous regions less interesting - transparent
- Surface “strength” depends on gradient
- Gradient of the scalar field is taken into account



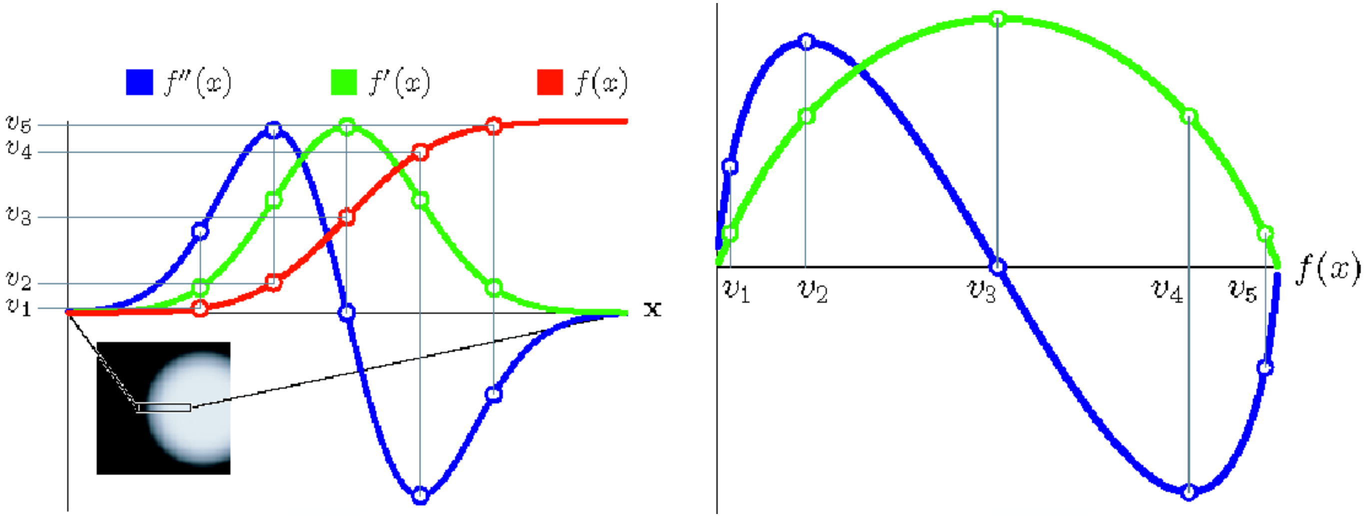
Classification

- Scalar value and gradient of the scalar field in a transfer function to emphasize isosurfaces [Levoy 1988]
- Achieves "isosurfaces" of constant width
- 2D transfer function



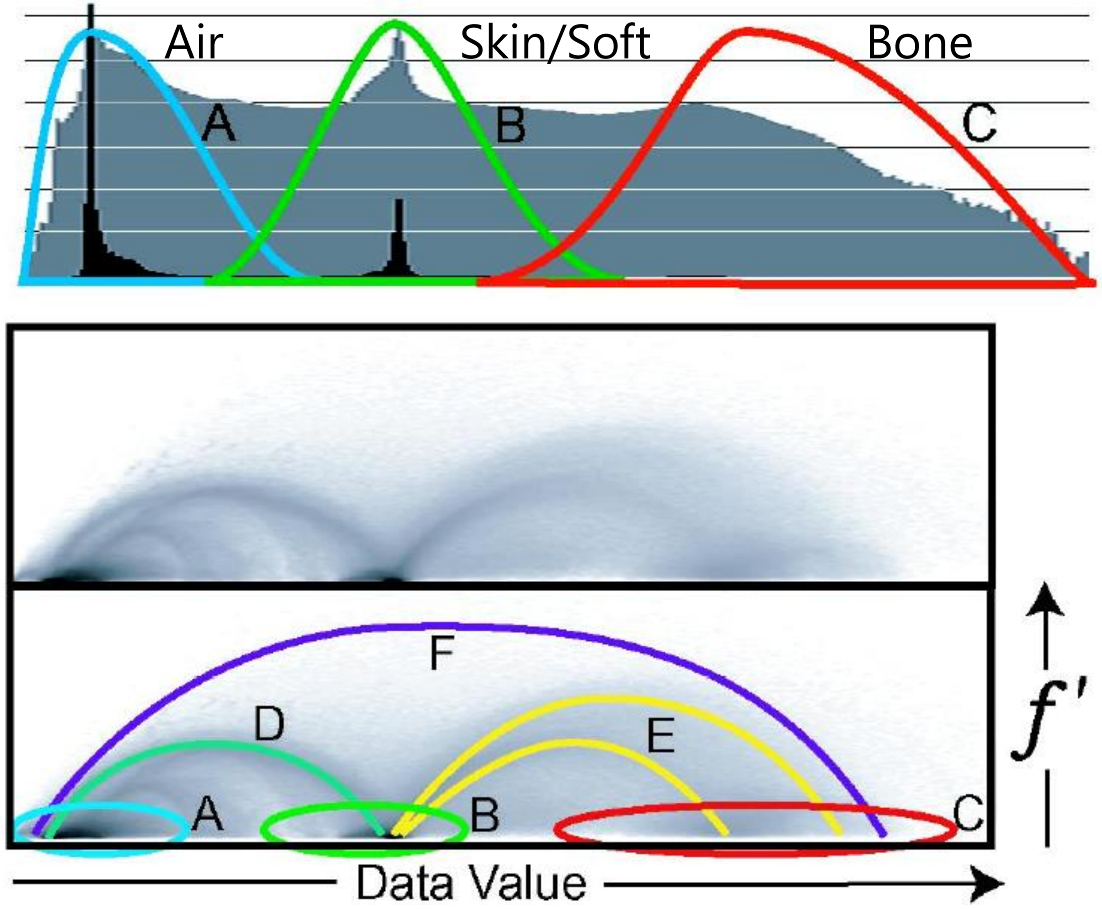
Classification

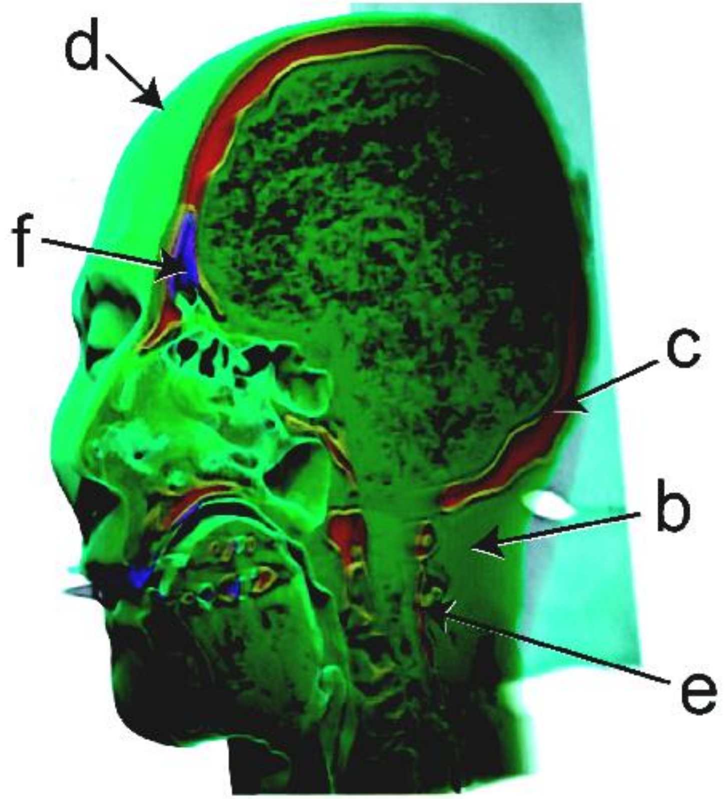
- Multidimensional transfer functions [Kindlmann & Durkin 98], [Kniss, Kindlmann, Hansen 01]
- Problem: How to identify boundary regions/surfaces
- Approach: 2D/3D transfer functions, depending on
  - Scalar value and magnitude of the gradient
  - Possibly also second derivative along the gradient direction



Classification

- Multidimensional transfer functions

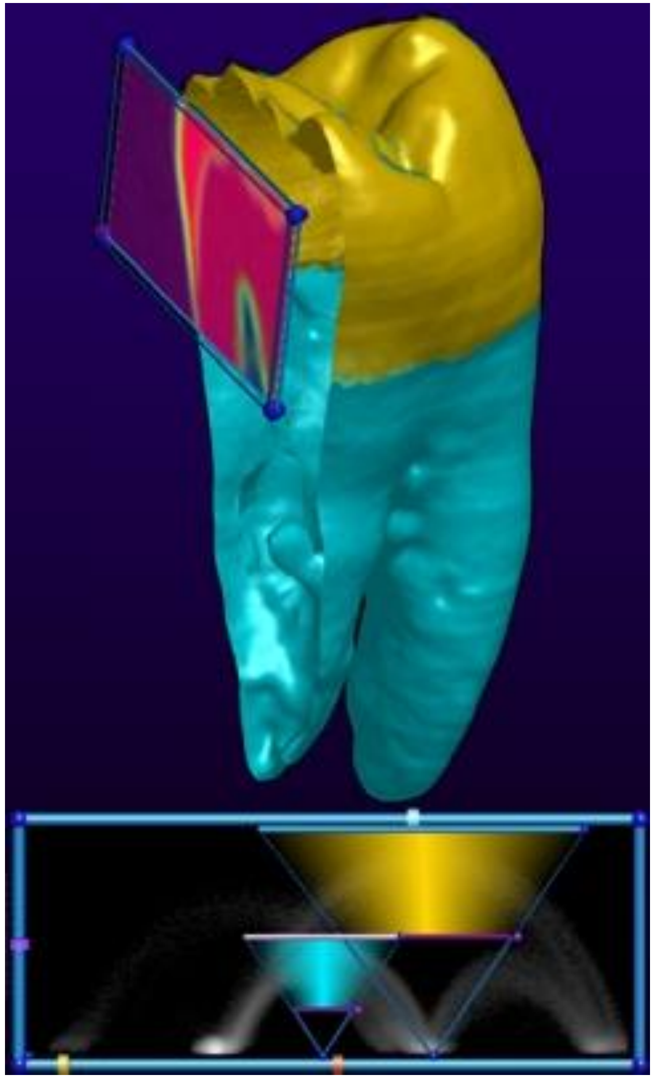




Classification

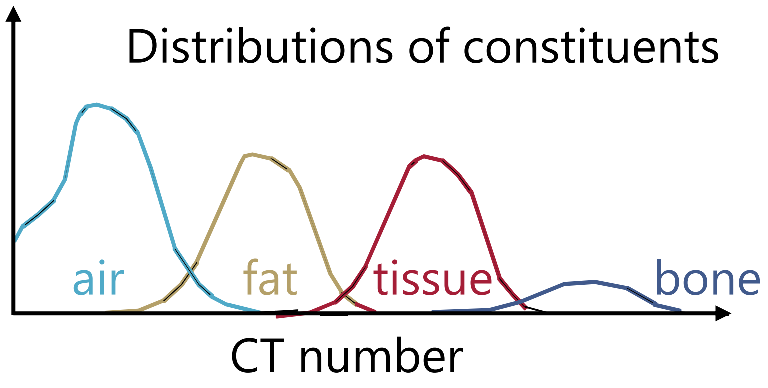
- Multidimensional transfer functions

- Extraction of two boundaries
- Triangle function in histogram



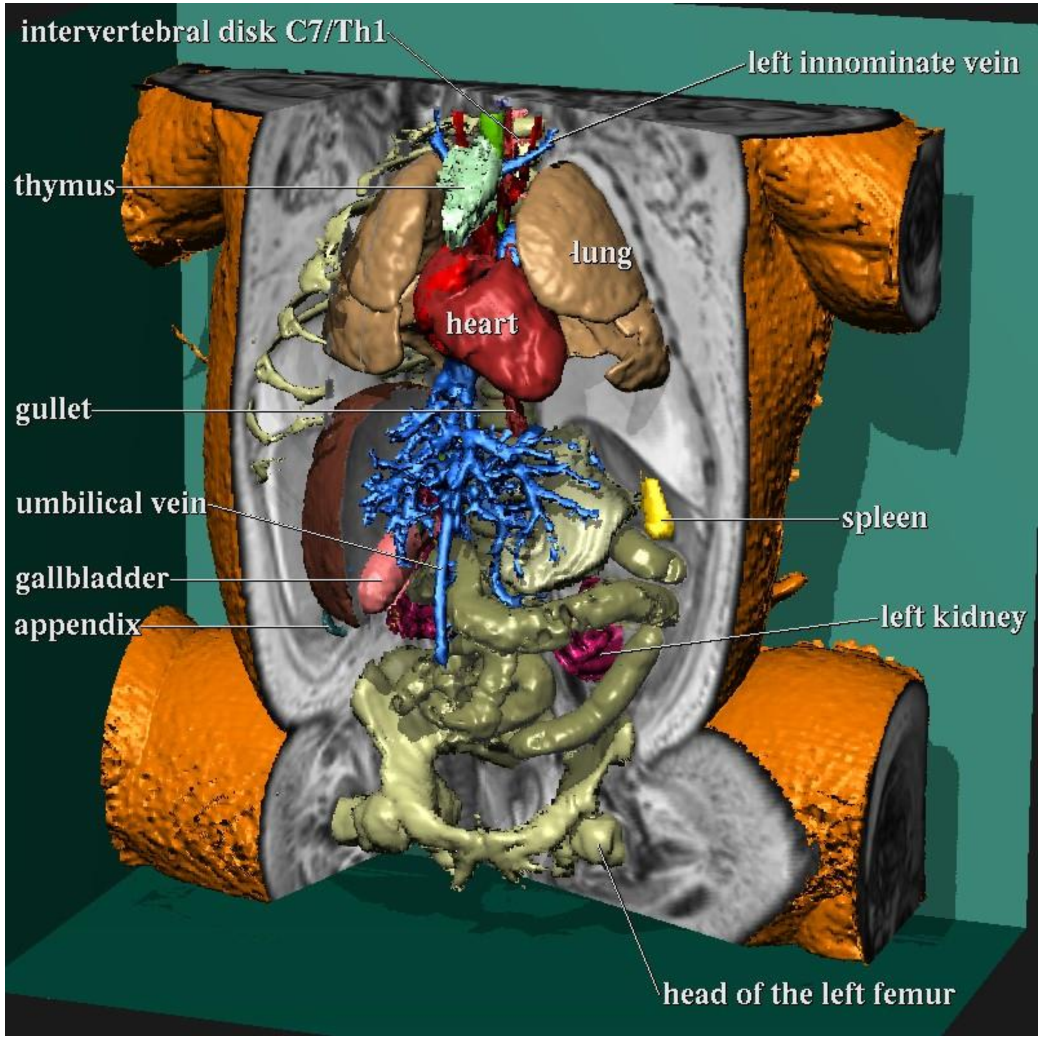
Segmentation

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



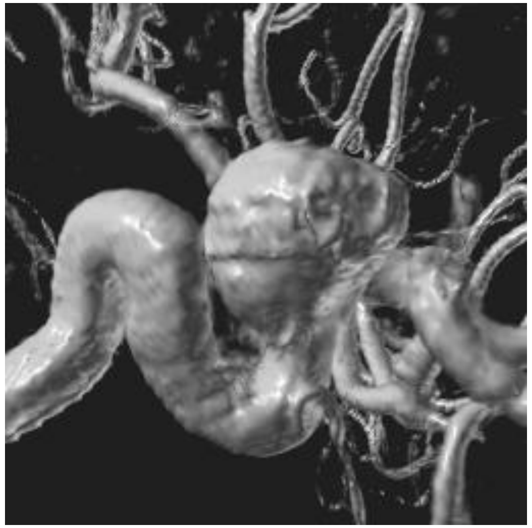
Segmentation

- Anatomic atlas



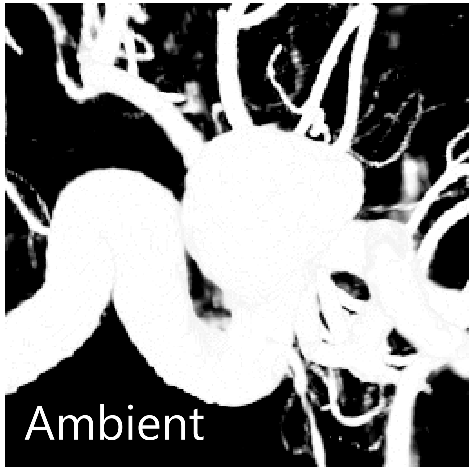
Volumetric Illumination

- Illumination:
  - Simulate reflection of light
  - Simulate effect on color
- We want to make use of the human visual system’s ability to efficiently deal with illuminated objects (shape perception)

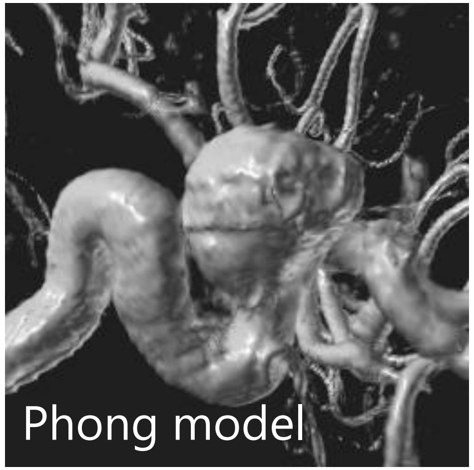


Volumetric Illumination

- Phong illumination model
  - Ambient light + diffuse light + specular light



$k_a = 0.1$
$k_d = 0.5$
$k_s = 0.4$

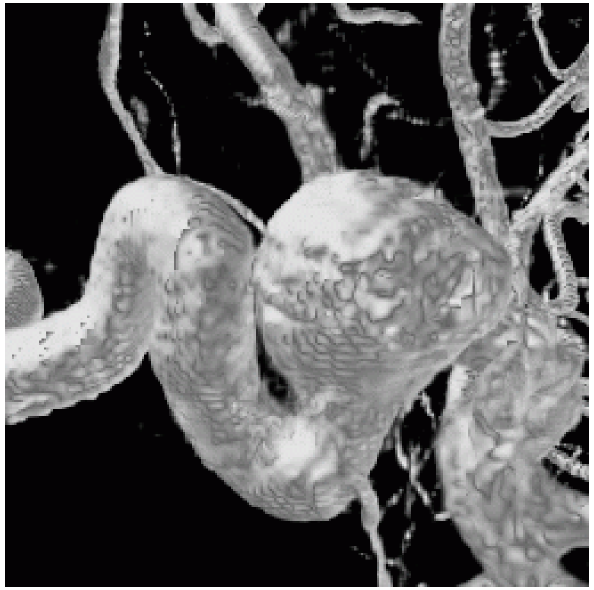


Volumetric Illumination

- What is the normal vector in a scalar field?
- Use the gradient!
- Gradient is perpendicular to isosurface (direction of largest change)

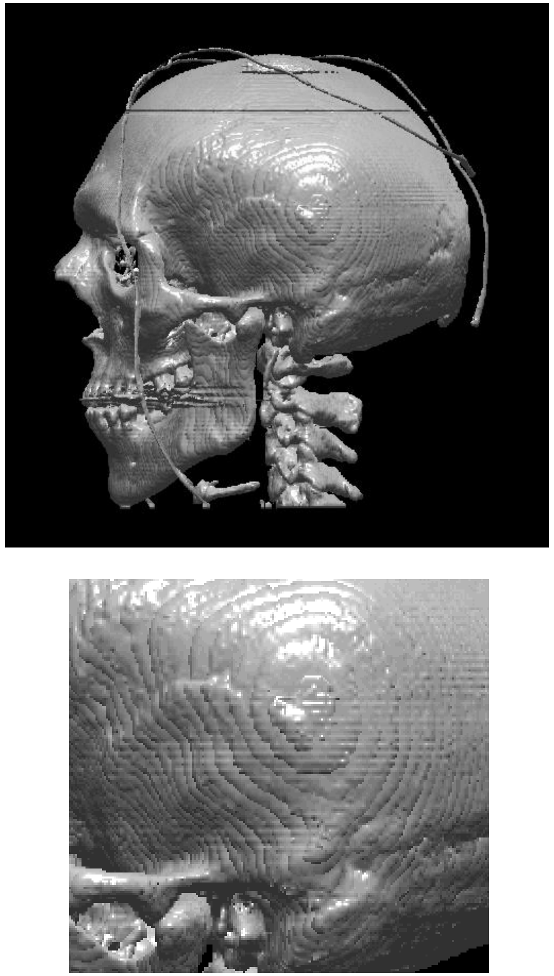
- Numerical computation of the gradient:
  - Central difference
  - Intermediate difference (forward/backward difference)
  - Sobel operator (3×3 kernel for each partial derivative)

Central differences

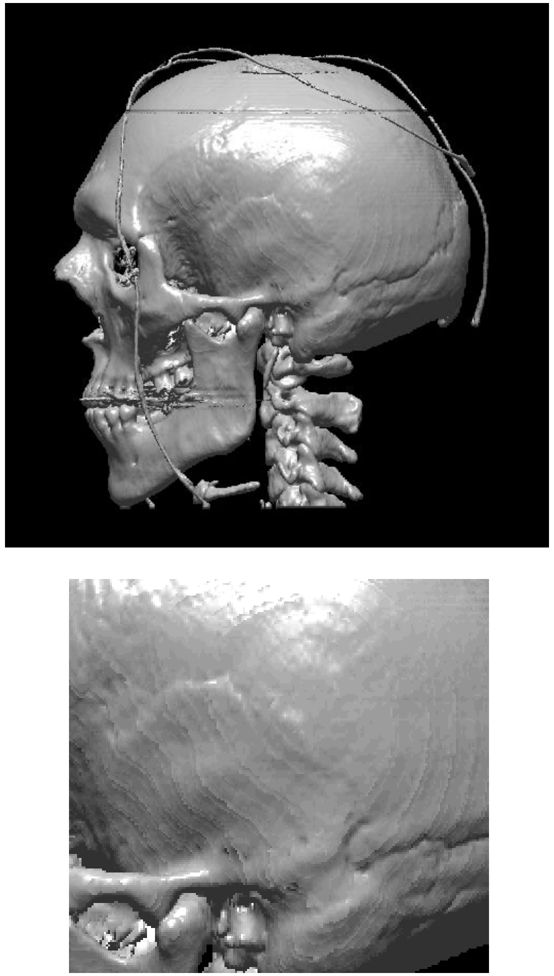


Intermediate differences

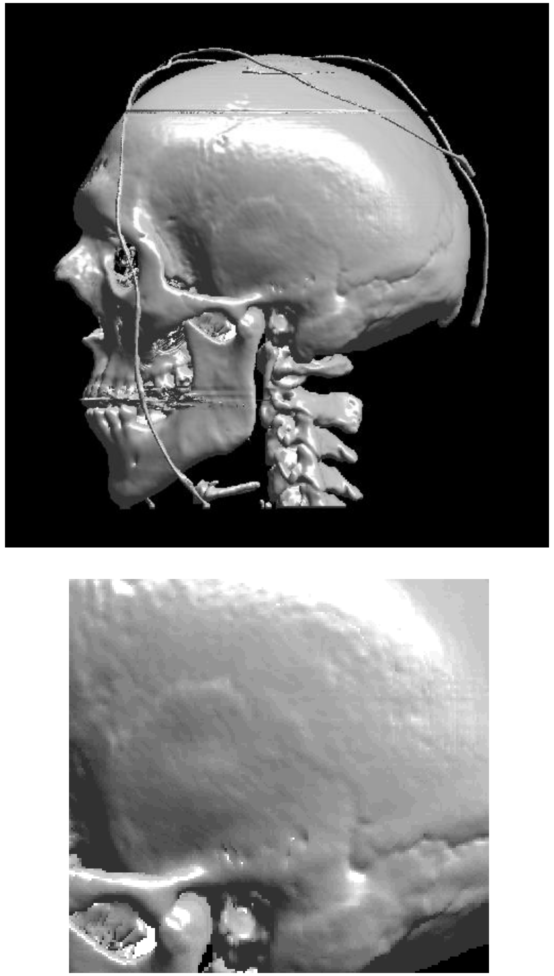
Volumetric Illumination



Intermediate differences



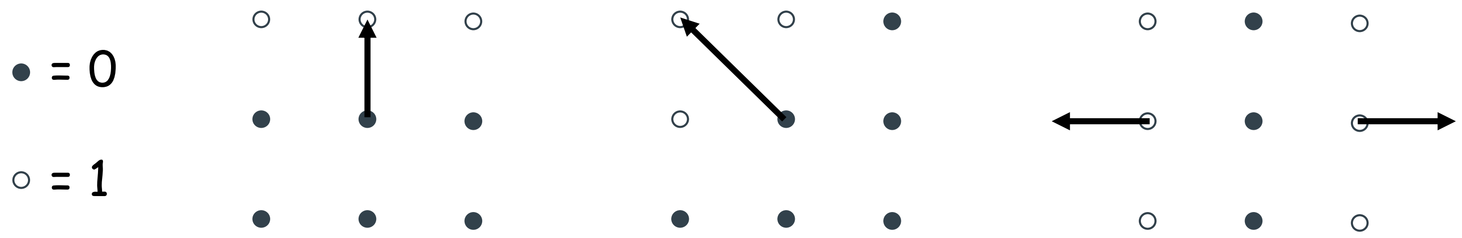
Central differences



Sobel operator

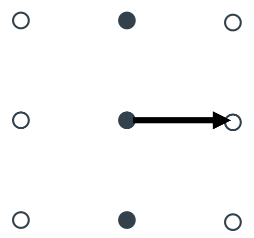
Volumetric Illumination

- Central differences
  - Computation:
$$
\begin{aligned}
G_x &= V_{x+1,y,z} - V_{x-1,y,z} \\
G_y &= V_{x,y+1,z} - V_{x,y-1,z} \\
G_z &= V_{x,y,z+1} - V_{x,y,z-1}
\end{aligned}
$$
  - Convolution kernel: [-1 0 1]
  - High-pass filter
  - Not isotropic; length is 1 to sqrt(3)
  - Needs normalization



Volumetric Illumination

- Intermediate difference (forward / backward)
  - Convolution kernel: [-1 1]
  - Very cheap
  - Noisy data → lower quality
  - Also not isotropic



- Sobel operator
  - Nearly isotropic
  - Rather expensive (multiple multiplications and summations)

prev. slice	this z-slice	next slice	
$\begin{bmatrix} -1 & 0 & 1 \\ -3 & 0 & 3 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -3 & 0 & 3 \\ -6 & 0 & 6 \\ -3 & 0 & 3 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -3 & 0 & 3 \\ -1 & 0 & 1 \end{bmatrix}$	partial derivative along the *x*-axis (other axes by rotation)

NEXT CHAPTER:

Direct Volume Visualization

Focus:
Second step of visualization pipeline