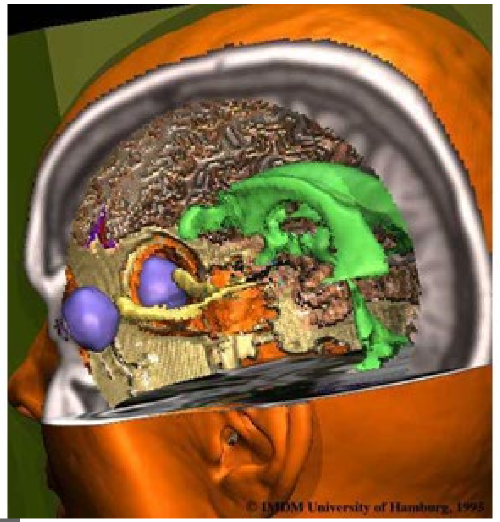
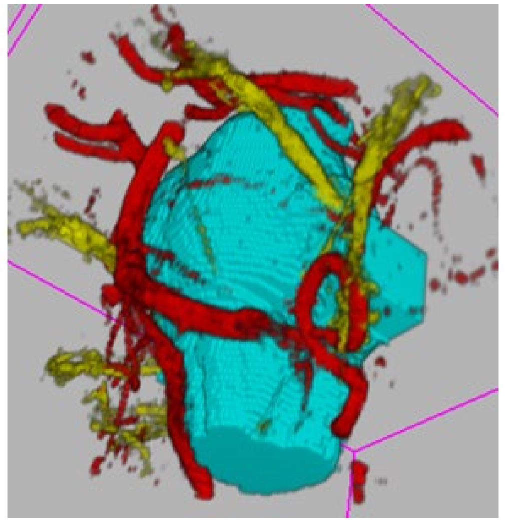
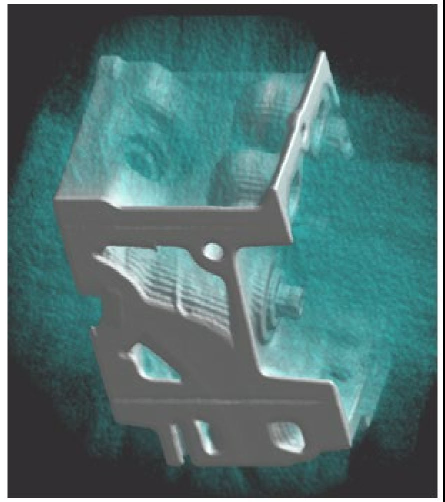
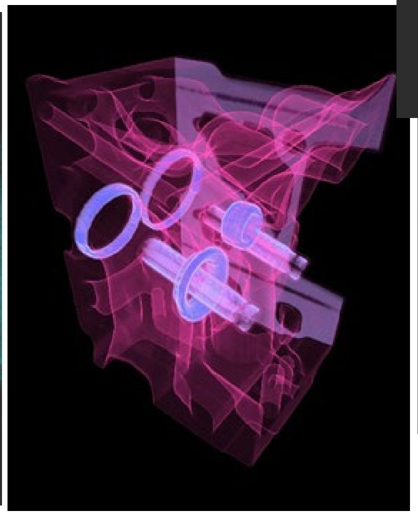
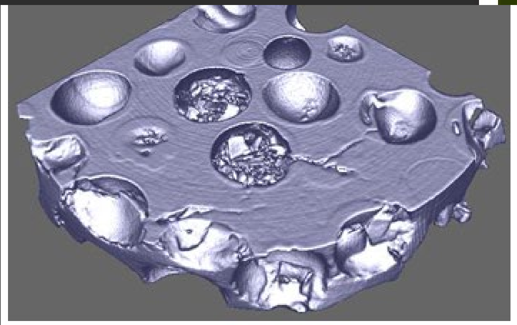
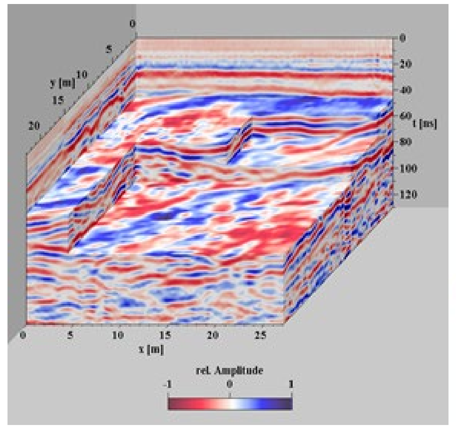
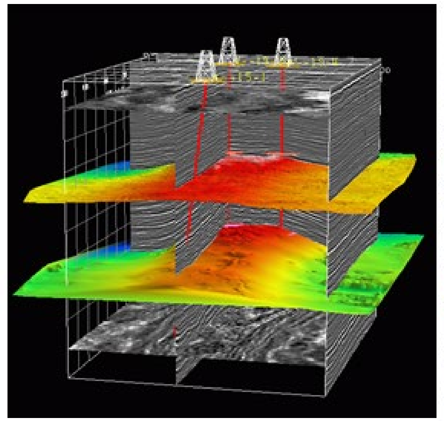
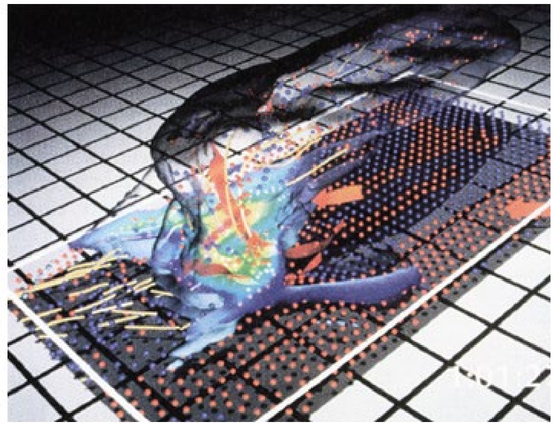
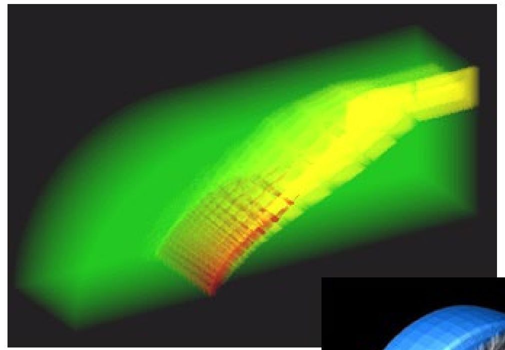
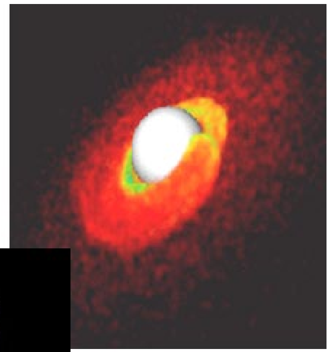
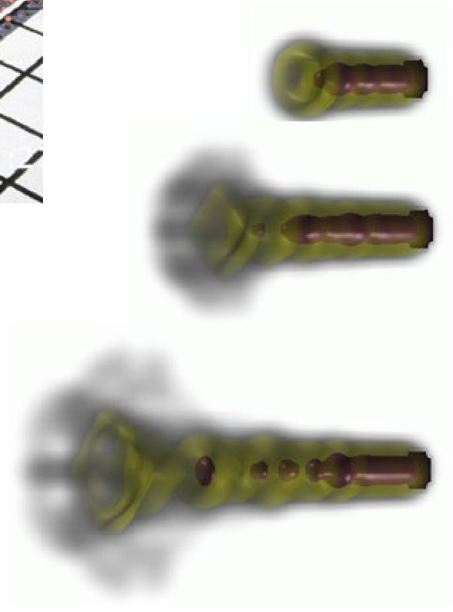
6. 3D Scalar Fields

Introduction

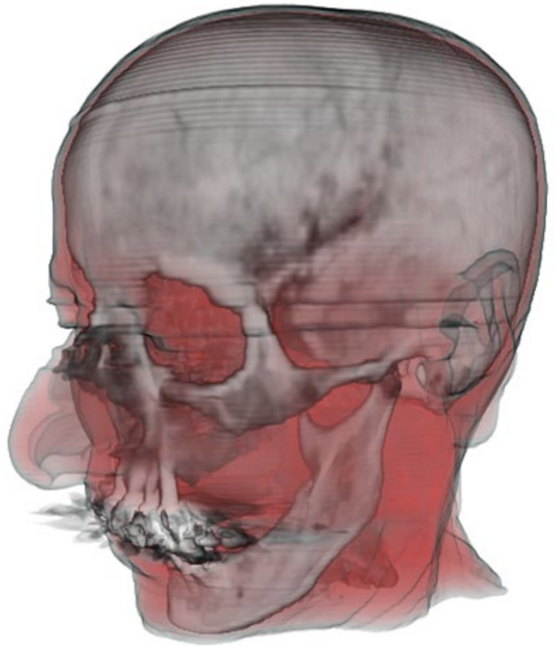
- Medical Applications
  - CT (Computed Tomography)
  - MRI (Magnetic Resonance Imaging)
  - Confocal Microscopy
  - 3D Ultrasound (US is usually 2D)
- Materials sciences
  - Industrial CT
    - Quality control, non-destructive testing
  - E.g. statue, engine block, wheels, etc.
- Geosciences
  - Geoseismic data
  - Exploration of natural resources (oil, gas)
  - Meteorological data
- Numerical simulations
  - Particle simulations
  - Finite element or finite differences methods, etc.

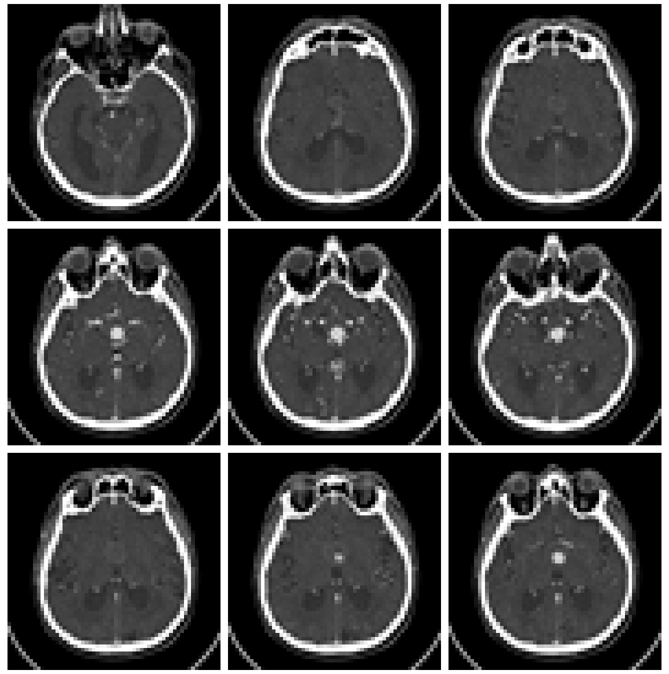








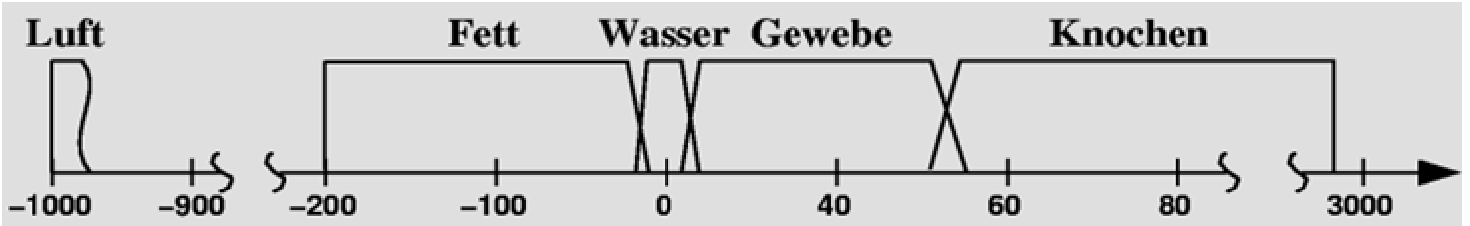
Introduction

Example: typical medical volume data





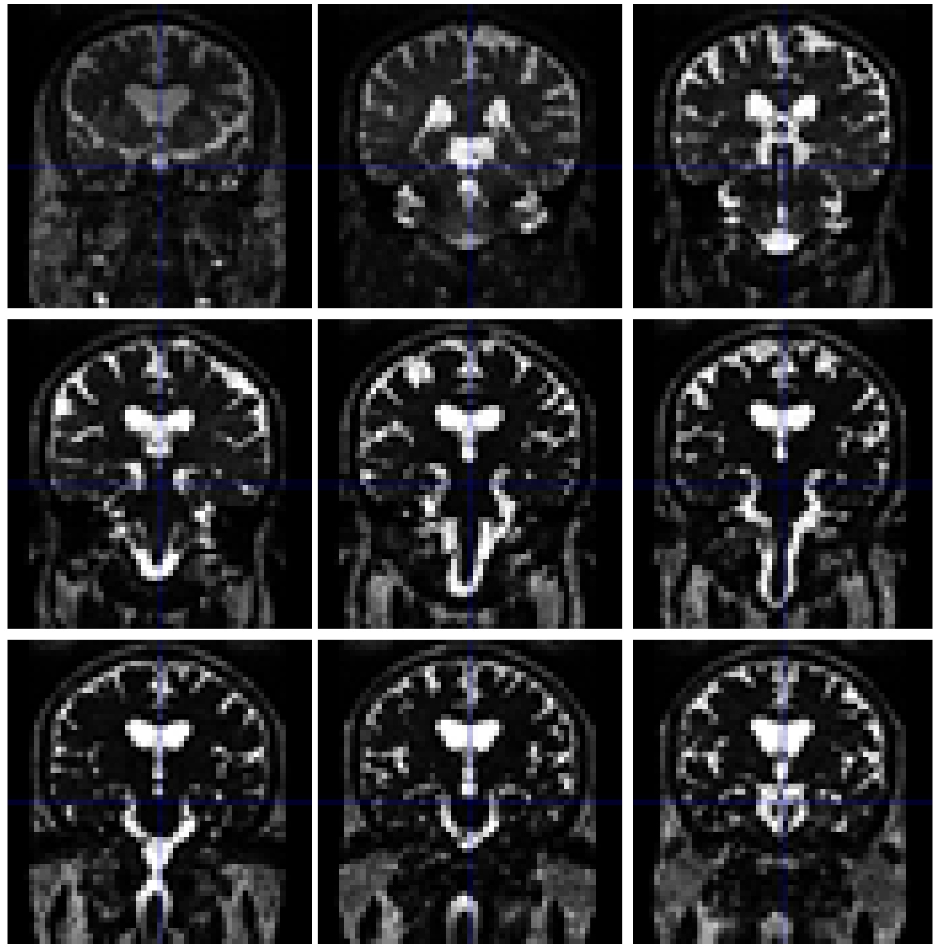
- Computed Tomography (CT)
  - Stack of slice images
    - Number of slice images: 100 - 2000
    - Typical image size: 512 x 512 pixels
    - Data representation: 12 or 16 bits
    - Typical voxel sizes: 0.4x0.4x0.4mm3, 0.7x0.7x1.2mm3
  - Representations (based on X-ray projections)
    - Good contrast between air, fat, muscle, bone
    - Metal objects are "bad" but possible
  - Hounsfield scale: -1000 - 3095 (12 bits)
    - Direct correlation between data values and tissue types

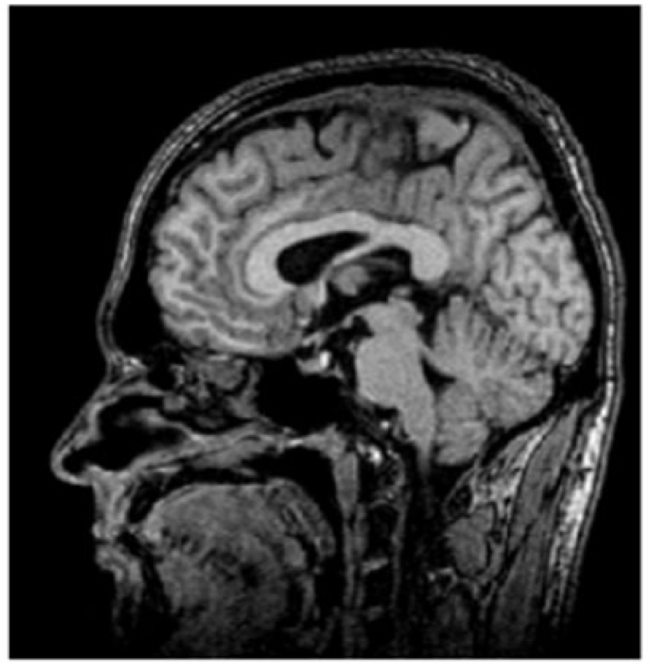


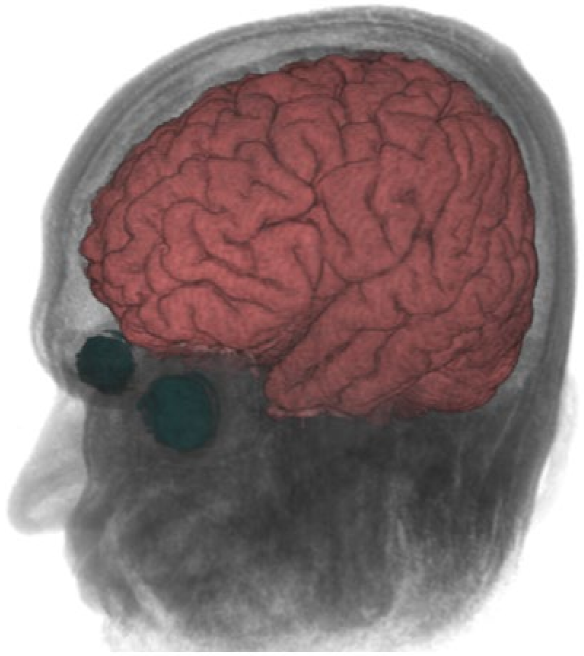
Introduction

Example: typical medical volume data

- Magnetic Resonance Tomography (MRT)
  - Stack of slice images
    - Number of slice images: 40 - 250
    - Typical image size: 512x512 pixels
    - Data representation: 12 or 16 bits
    - Typical voxel sizes: 0.7x0.7x0.7mm3, 1.0x1.0x1.7mm3
  - Representations (no ionizing radiation!)
    - Better contrast for soft tissue, bone is dark
    - Metal object scans are "not possible"
  - Nothing comparable to Hounsfield scale
    - No direct correlation between data values and tissue types

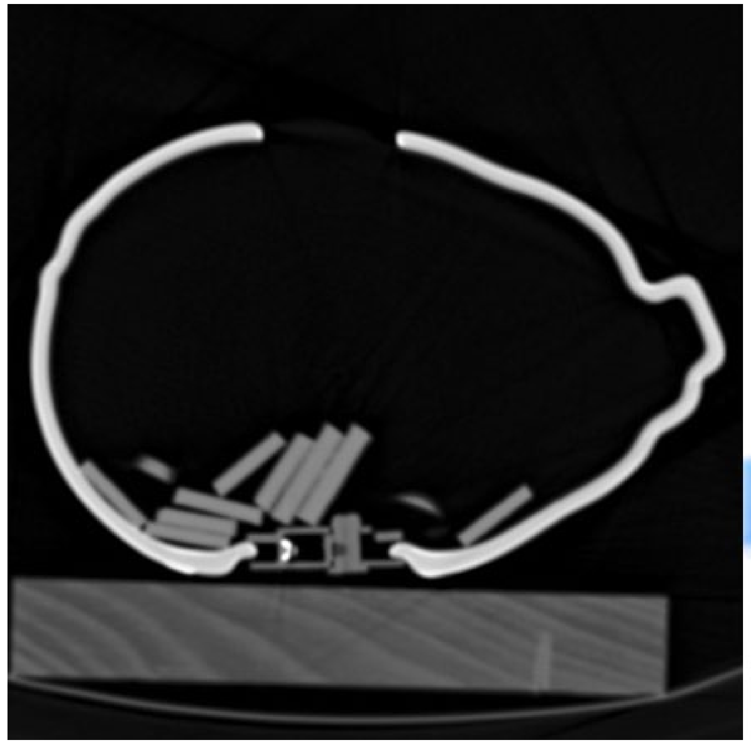


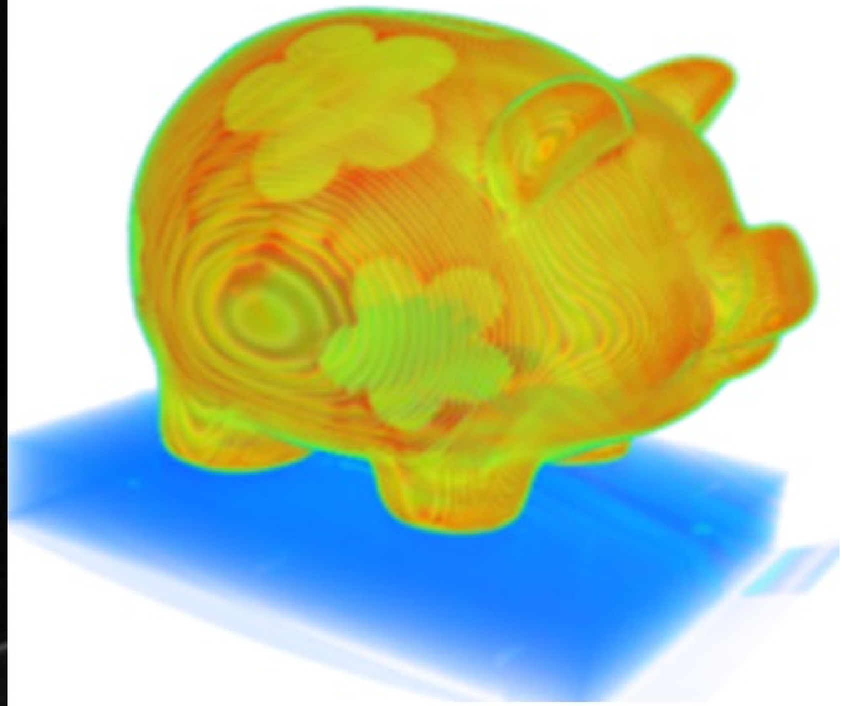


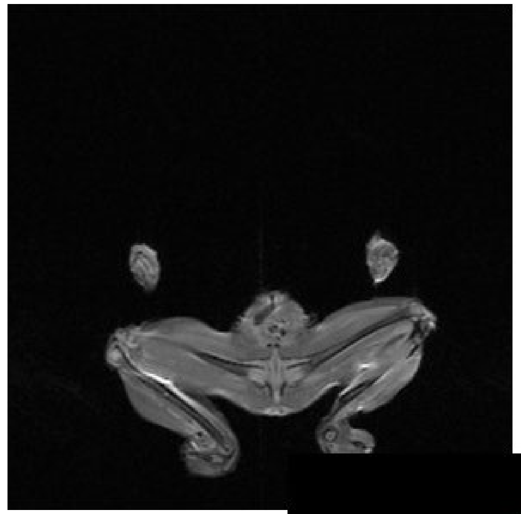


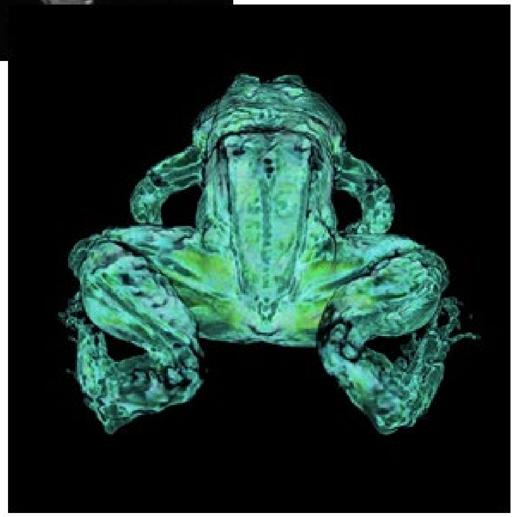
Introduction

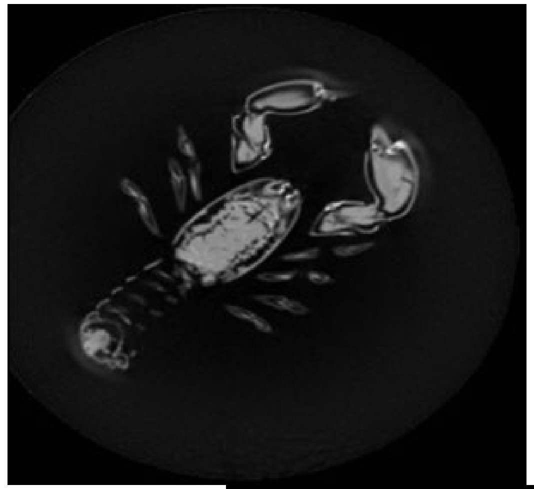
More examples

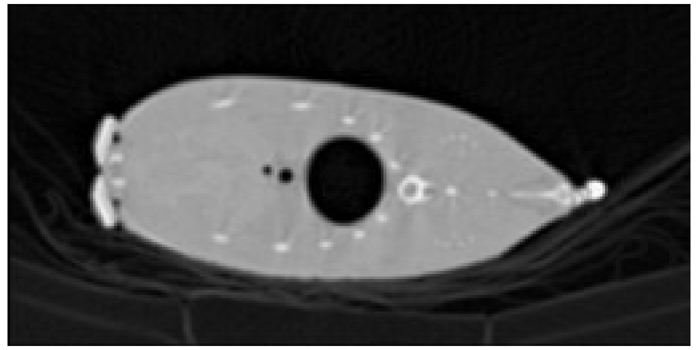












6.1 Volume Visualization

ohm

Volume Visualization

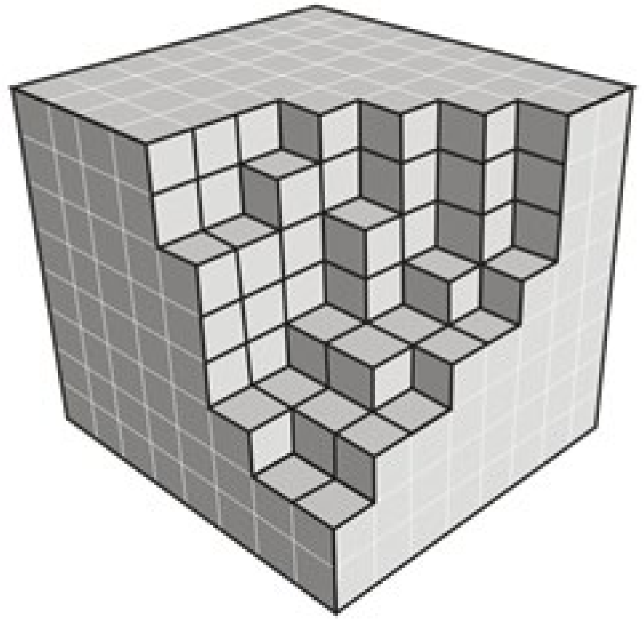
Possible characteristics of volume data

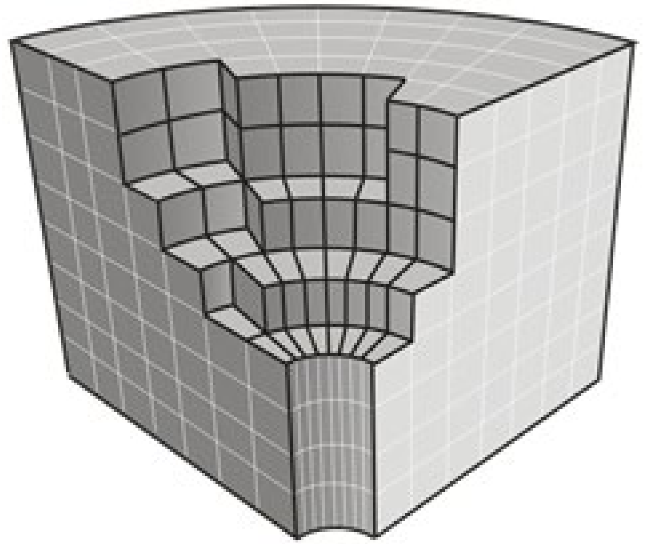
- Essential information in the interior
- Cannot be described by geometric representation
  - Fire, clouds, gaseous phenomena
- Distinguish between shape
  - Given by the geometry of the grid
- and appearance
  - Given by the scalar values
- Even if the data could be described geometrically, there are, in general, too many primitives to be represented

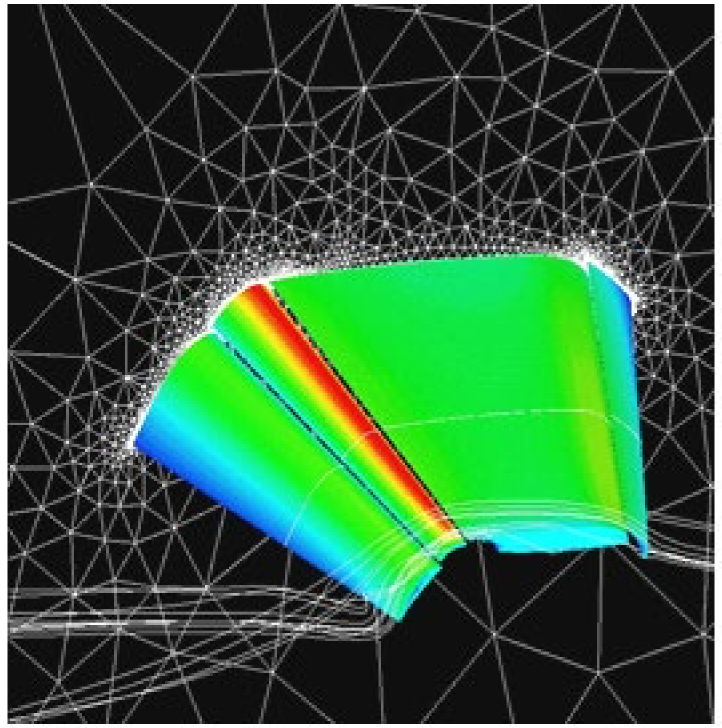
Volume Visualization

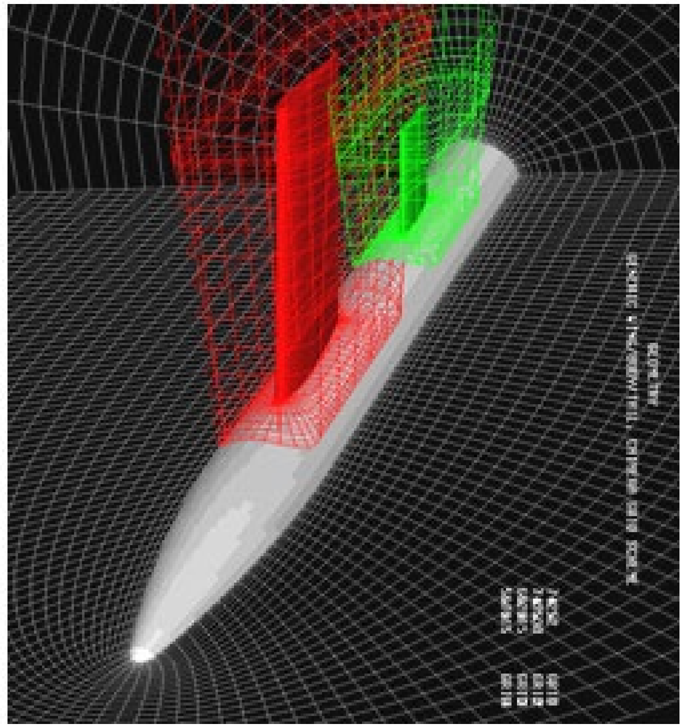
Grid structures

- Structured
  - Uniform
  - Rectilinear
  - Curvilinear
- Unstructured
  - Tetrahedral
  - Mixed elements
    - Hexahedron
    - Tetrahedron
    - Prism
    - Pyramid
  - Scattered data





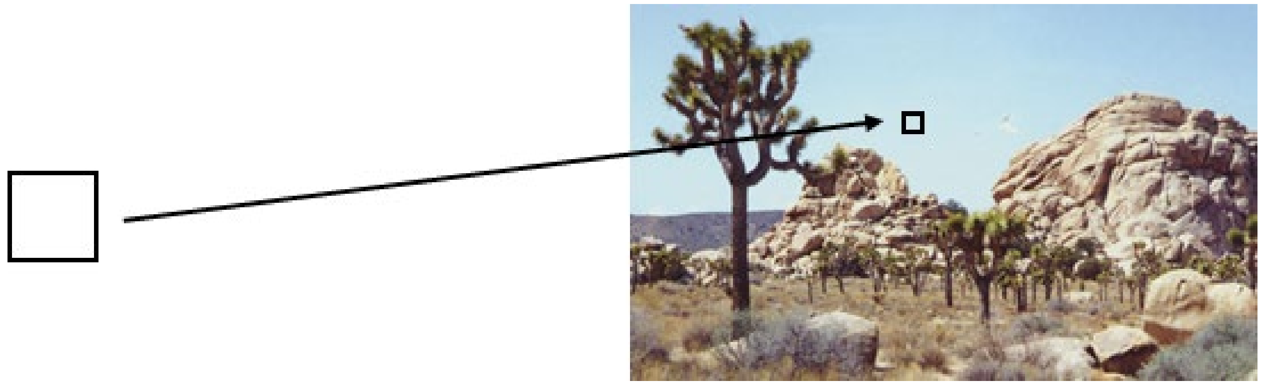




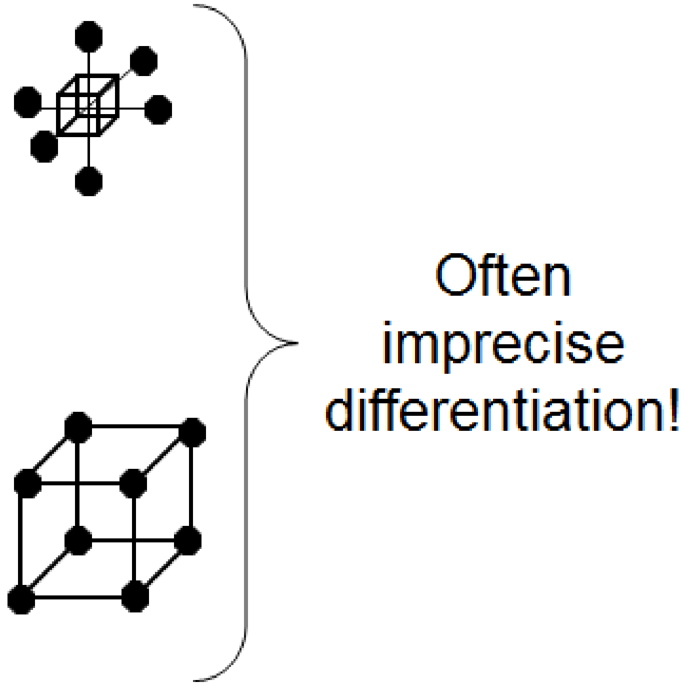
Volume Visualization

Definitions (most common for structured uniform grids)

- Pixel
  - "Picture element"



- Voxel
  - "Volume element"
  - Values are constant within a region around a grid point
- Cell
  - Values between grid points are resampled by interpolation



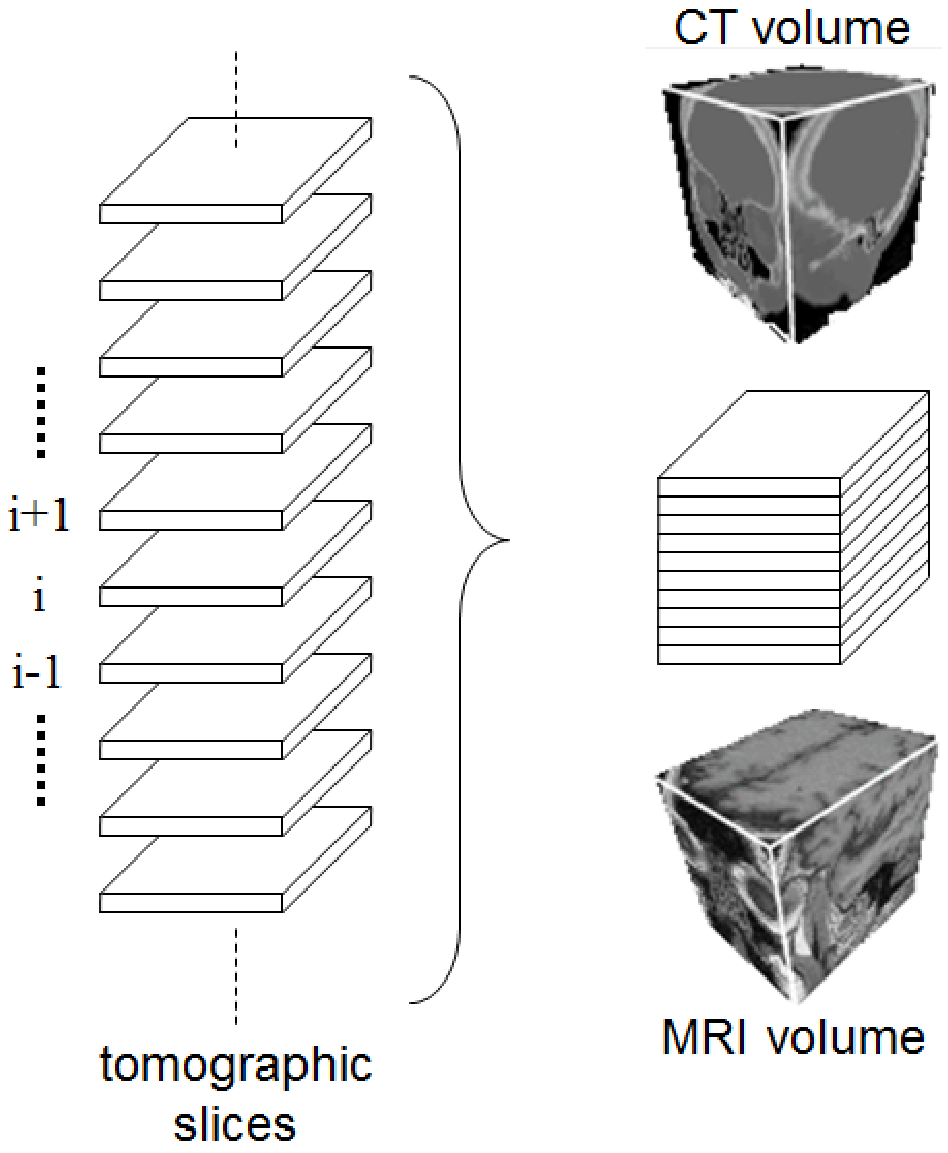
Volume Visualization

Main visualization approaches

- Slicing
  - Standard 2D approach
  - Techniques for 2D scalar fields
- Indirect volume rendering
  - Convert/reduce volume data to surface representation
  - Rendering with traditional techniques
- Direct volume rendering
  - Take volume data as real volumetric object
  - Consider as light-emitting, semi-transparent gel
  - Directly get a 3D representation of the volume data

Stichwort: Isoflächen

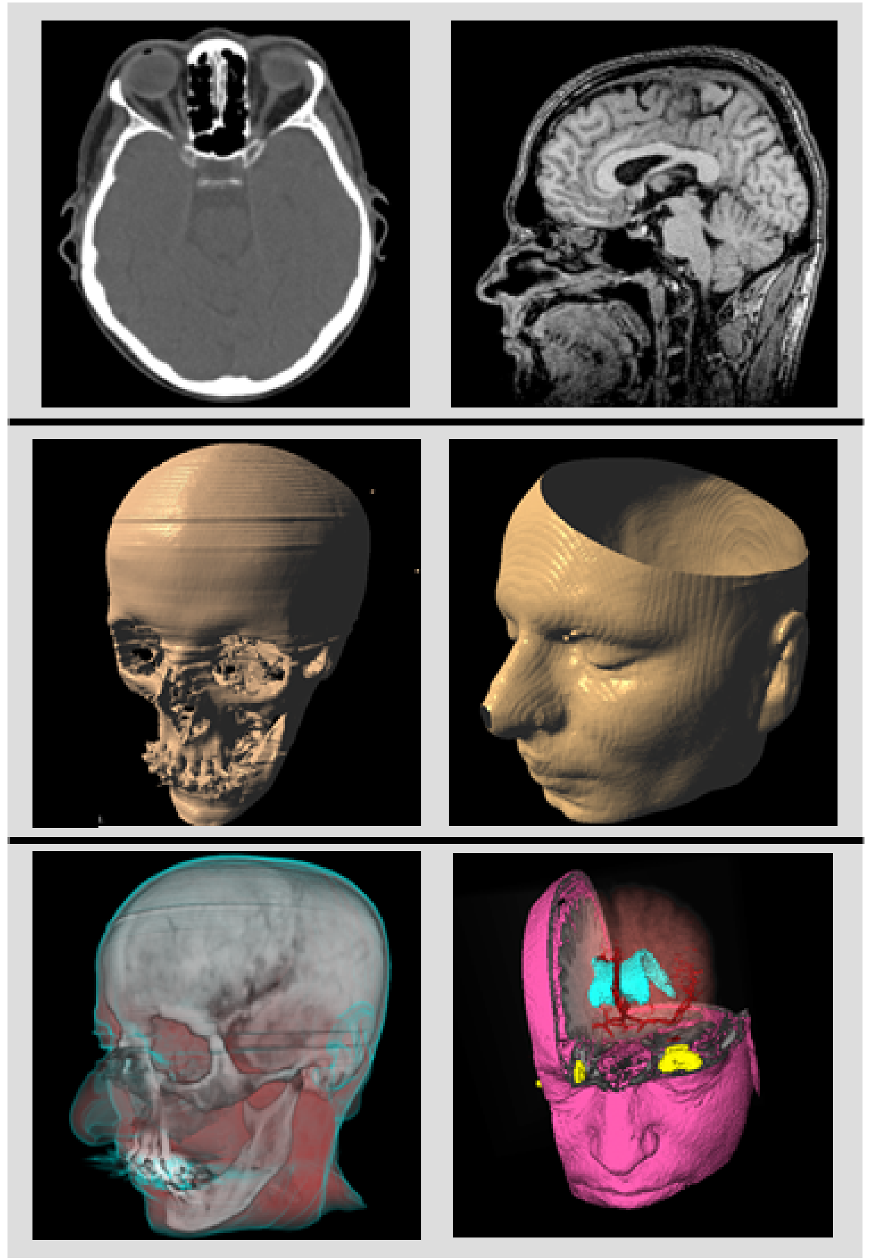
Volume Visualization



Slicing
- 2D visualization

Indirect volume rendering
- Polygonal rendering

Direct volume rendering
- Volume rendering



Volume Visualization

Considerations

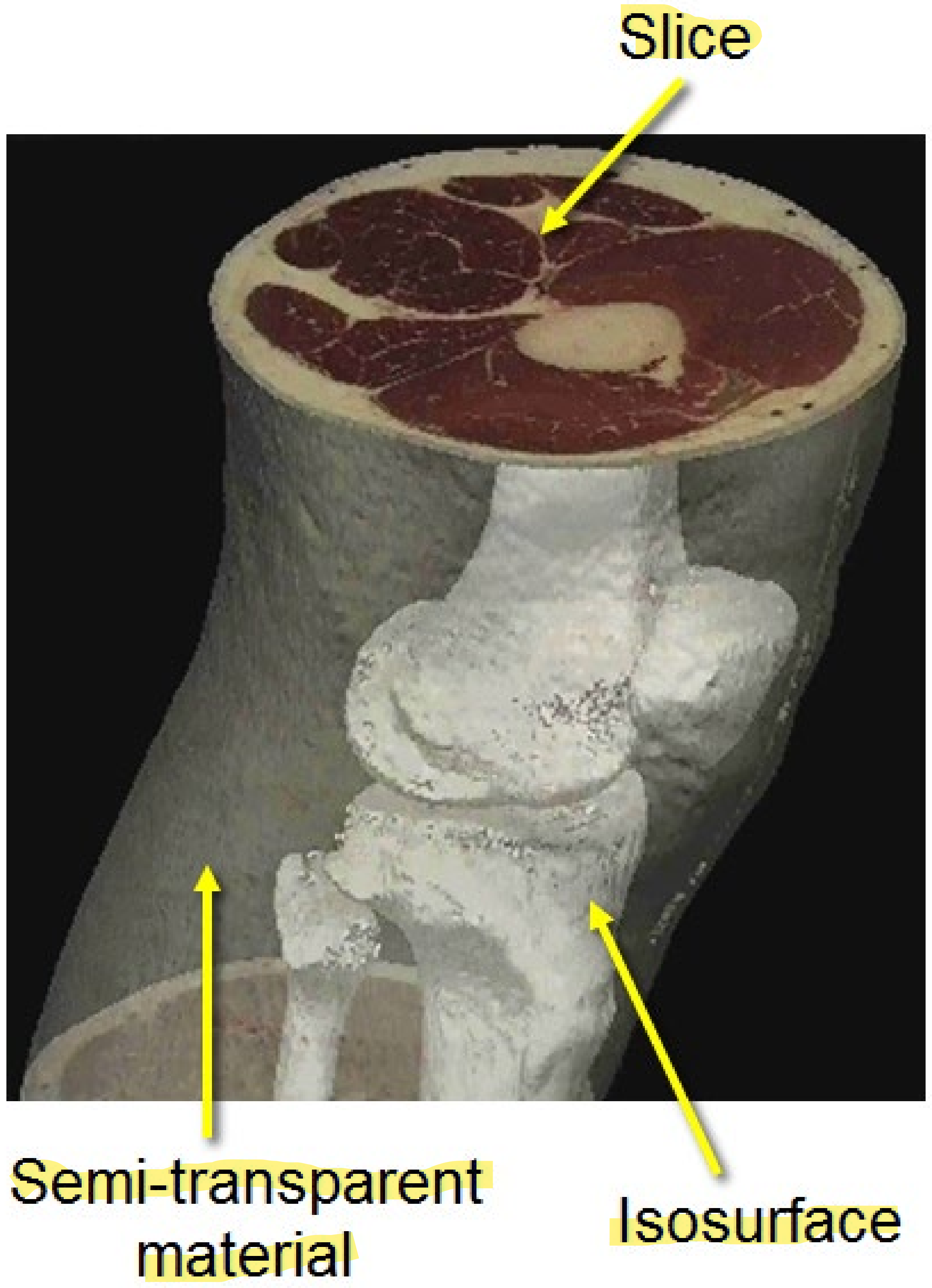
- Indirect volume rendering techniques
  - Often result in complex representations
  - Preprocessing of the surface representation might help
  - Standard graphics hardware for interactive display
- Direct volume rendering techniques
  - "Global" representation integrating physical characteristics
  - Interactive display difficult due to numerical complexity
  - Use graphics hardware for acceleration
- Goal
  - Integration of different techniques for optimal display
  - The most correct method in terms of physical realism may not be the most optimal one in terms of understanding the data

ohm

Volume Visualization

Example

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



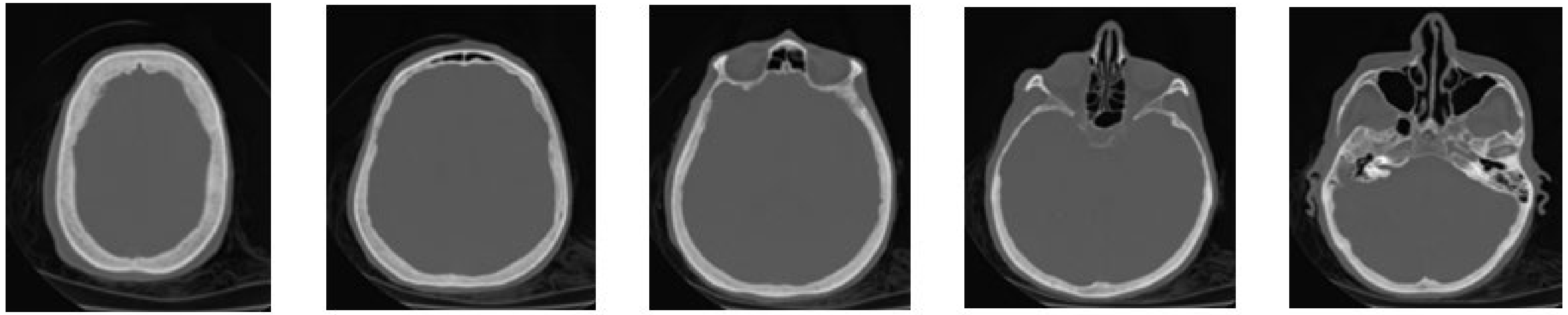
6.2 Slicing of Volume Data

ohm

Slicing of Volume Data

Orthogonal slicing

- Resample the volume data on parallel planes perpendicular to the x-, y-or z-axis
  - Example: z = -25, -24, ..., 23, 24
  - Let the user interactively change the z-value



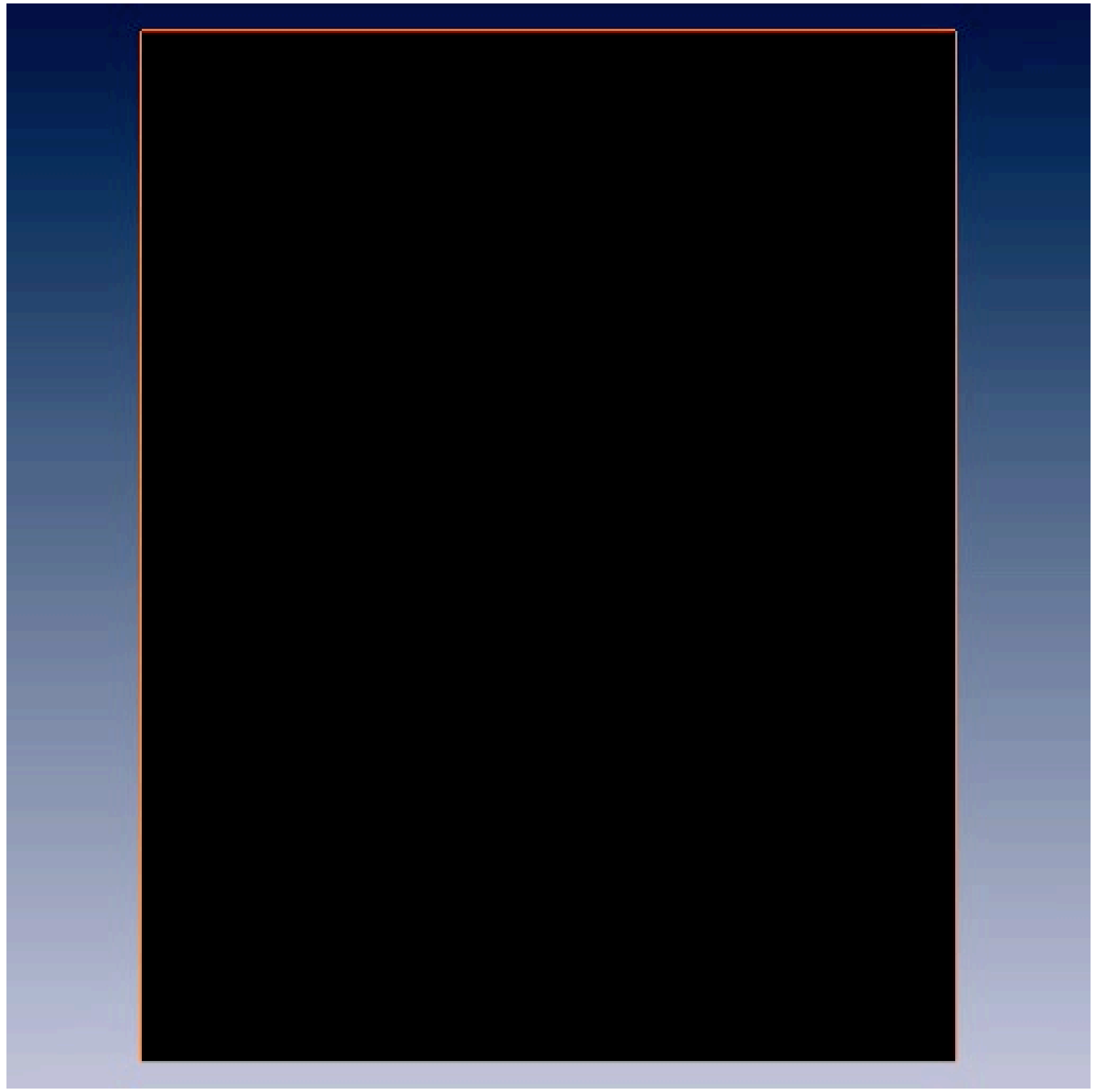
z = 20 z = 30 z = 40 z = 50 z = 60

- Use visualization techniques for 2D scalar fields
  - Color coding, isolines, height fields, ...

ohm

Slicing of Volume Data

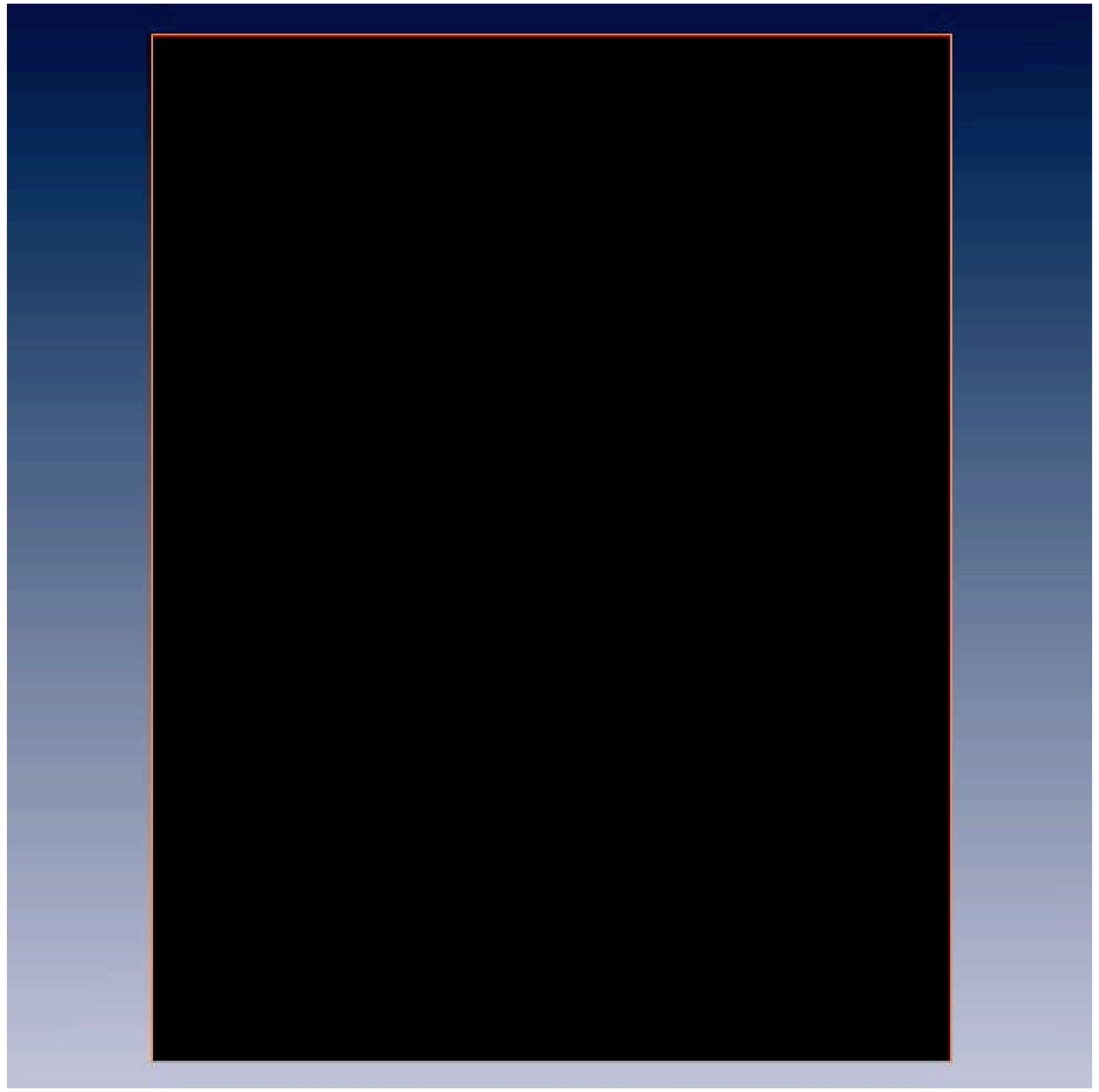
- Example: Visible male



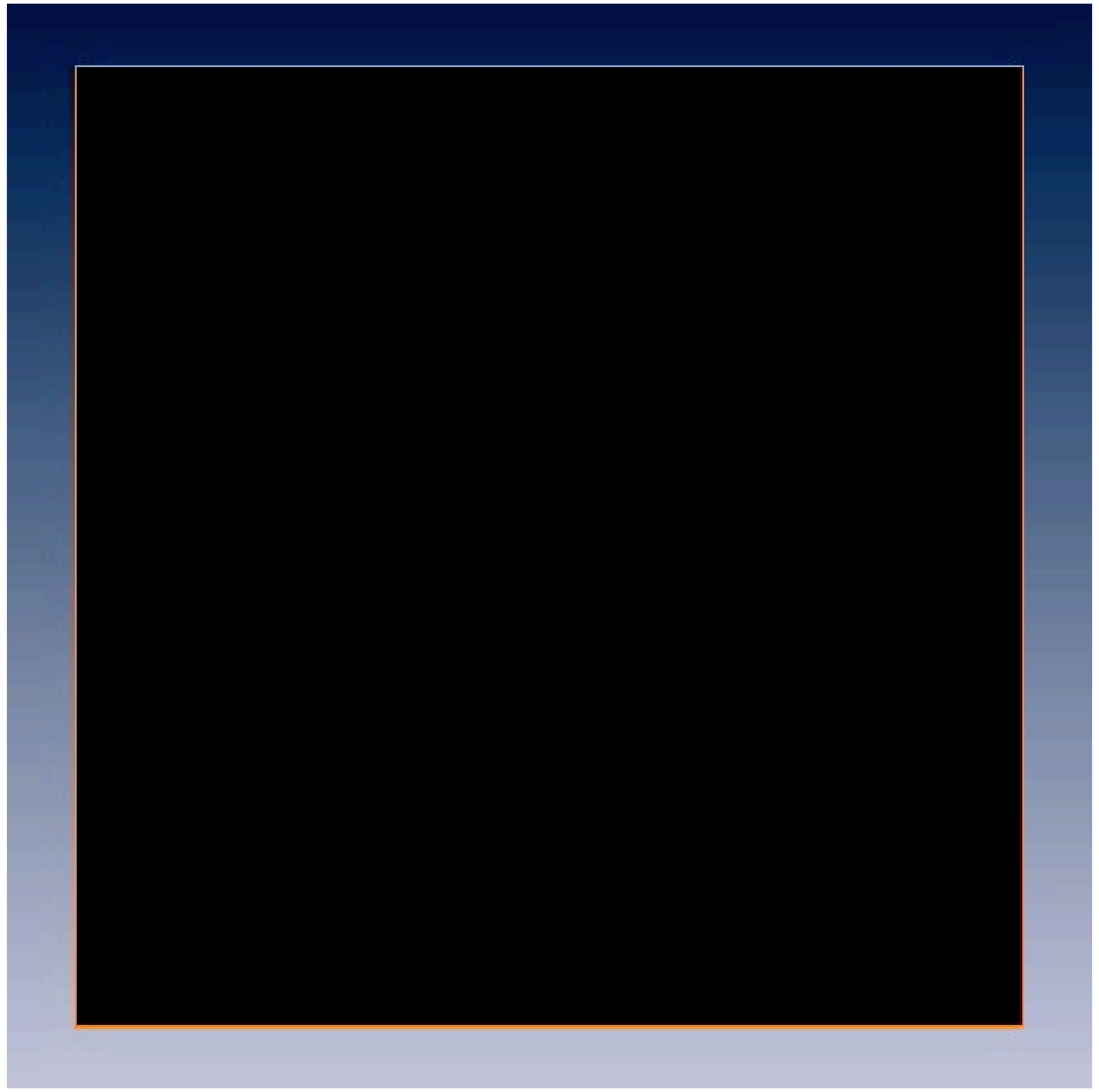
axial

Slicing of Volume Data

- Example: Visible male



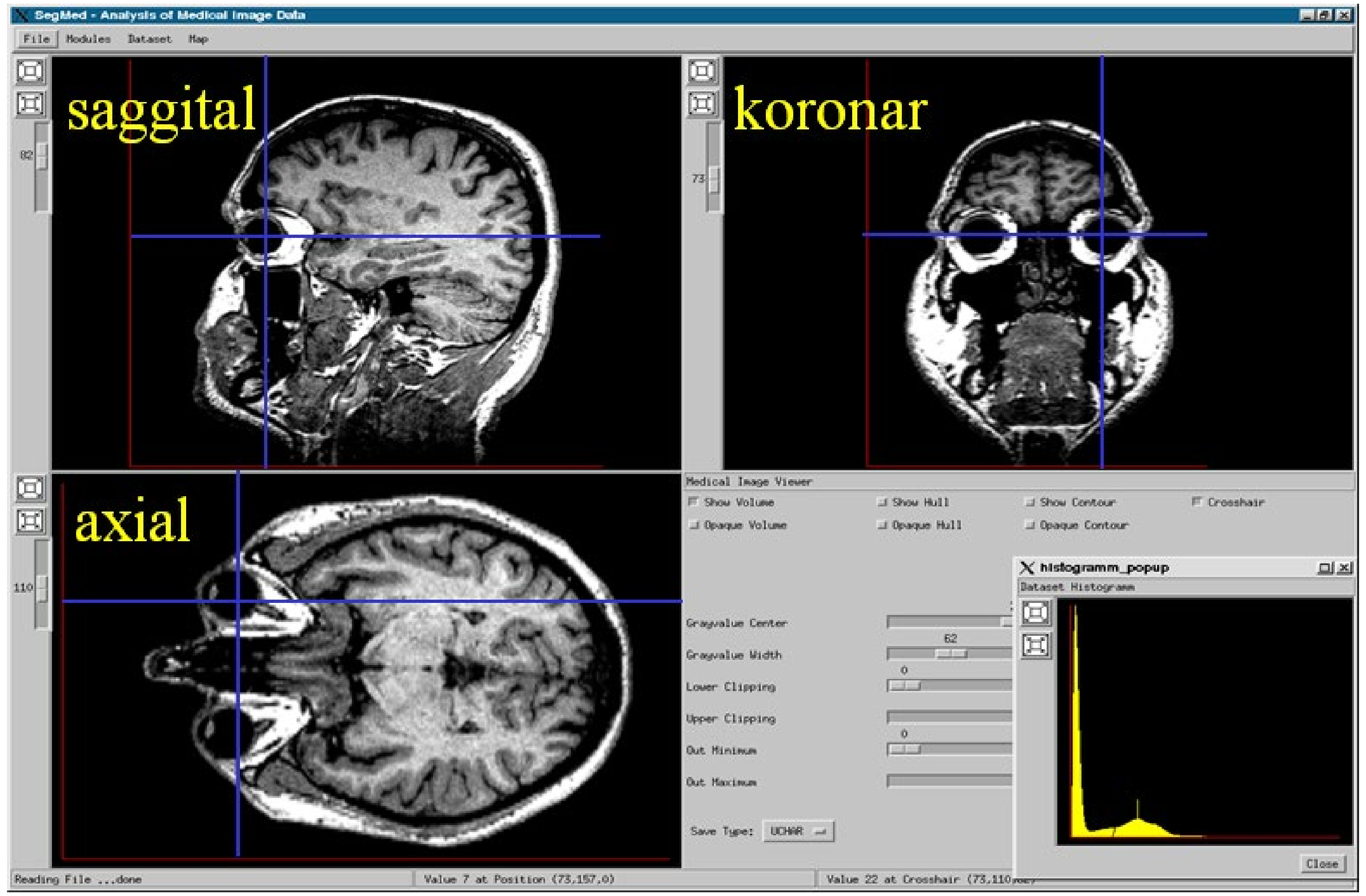
coronar

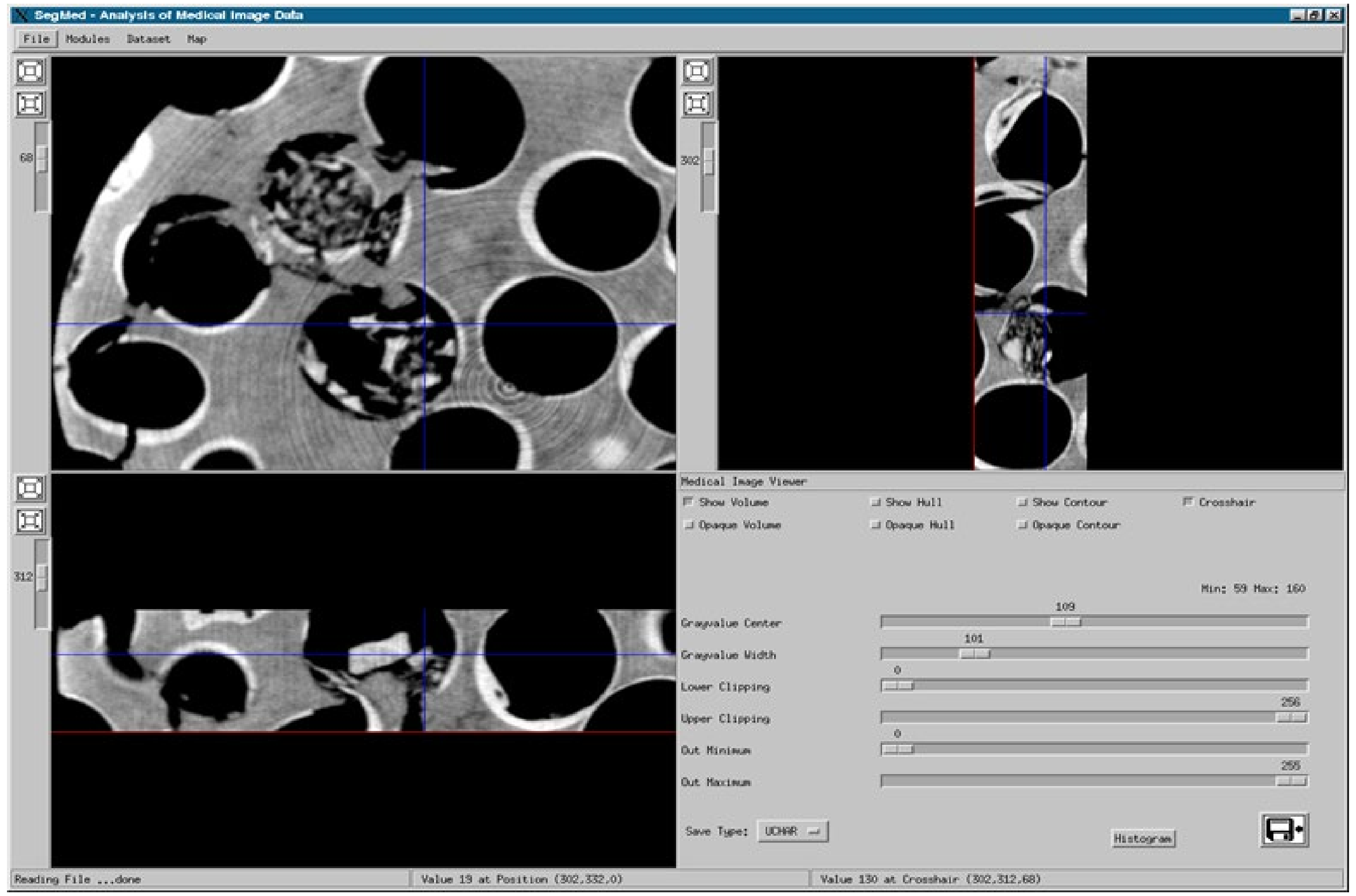


sagittal

Slicing of Volume Data

- Simultaneous slicing in x-, y-or z-axis

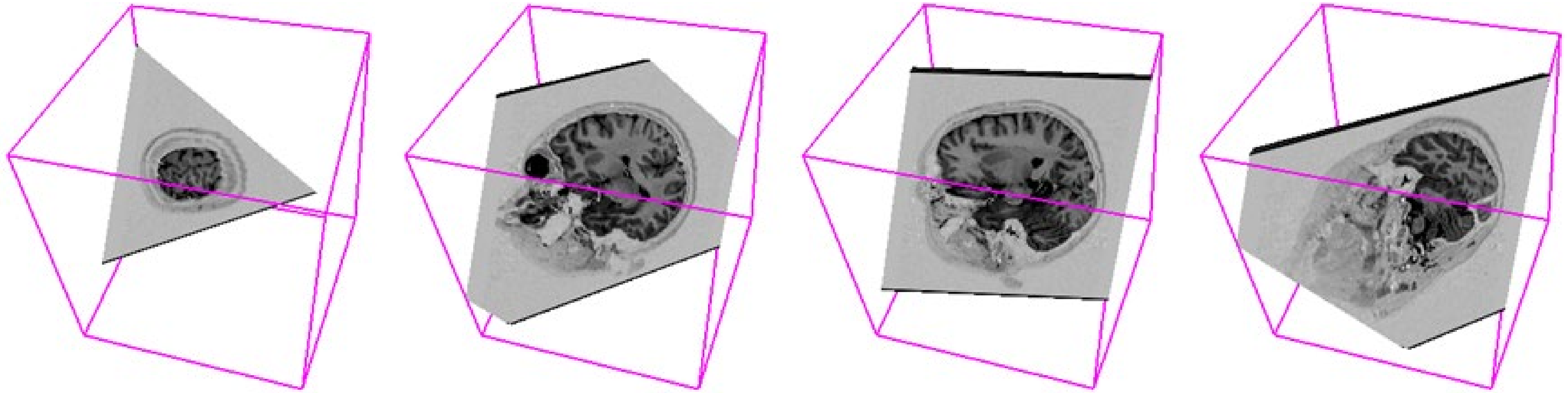




Slicing of Volume Data

Oblique slicing (MPR - multiplanar reformating)

- Resample volume data on arbitrarily oriented slices
- Interpolation in software on CPU
- Exploit graphics hardware
  - Store volume data in 3D texture memory
  - Get sectional polygon (plane clipped with bounding box of volume)
  - Render textured polygon

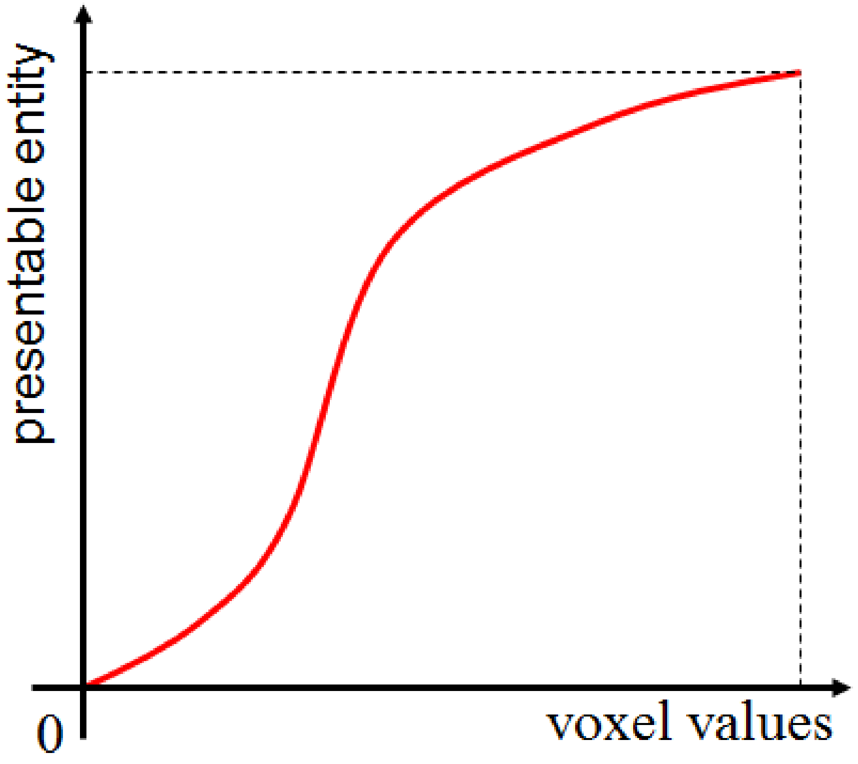


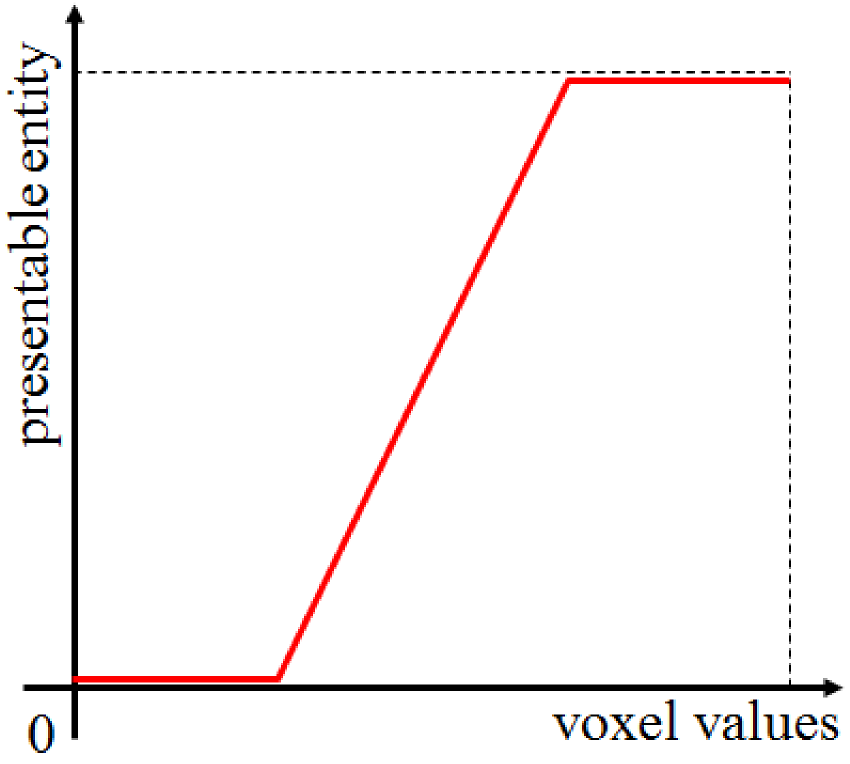
6.3 Classification

Classification

Transfer function

- Map voxel values to representable entities
  - Color, intensity, opacity, etc.



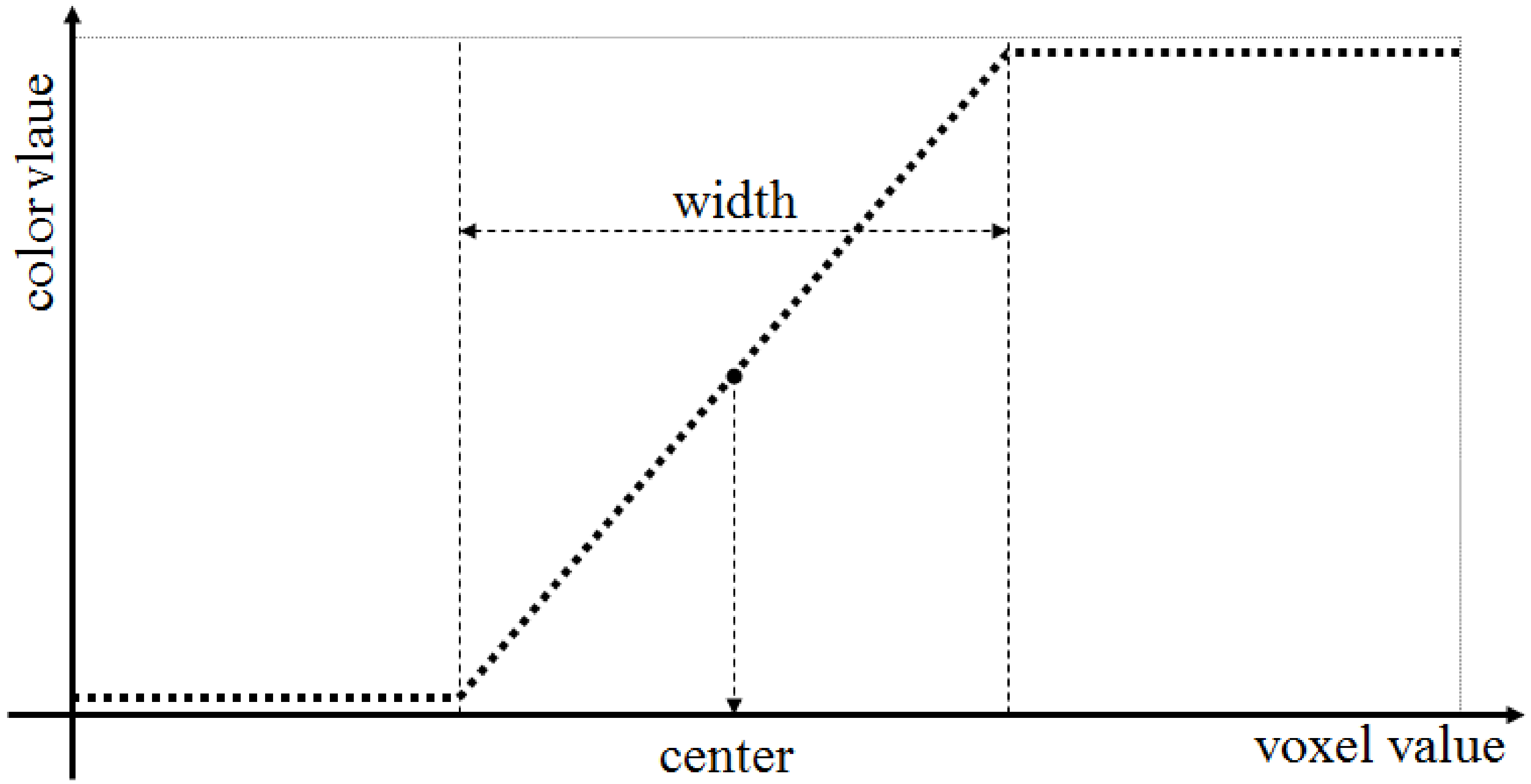


ohm

Classification

Transfer function

- Grey value transformation ("windowing")



Adjustment of width and center

Classification

Goals and issues

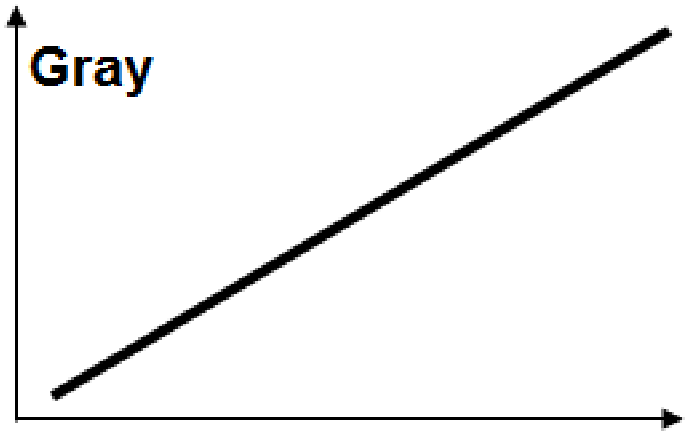
- Empowers user to select "structures"
- Extract important features of the data set
- Classification is non trivial
- Histogram can be a useful hint
- Often interactive manipulation of transfer functions needed

Classification

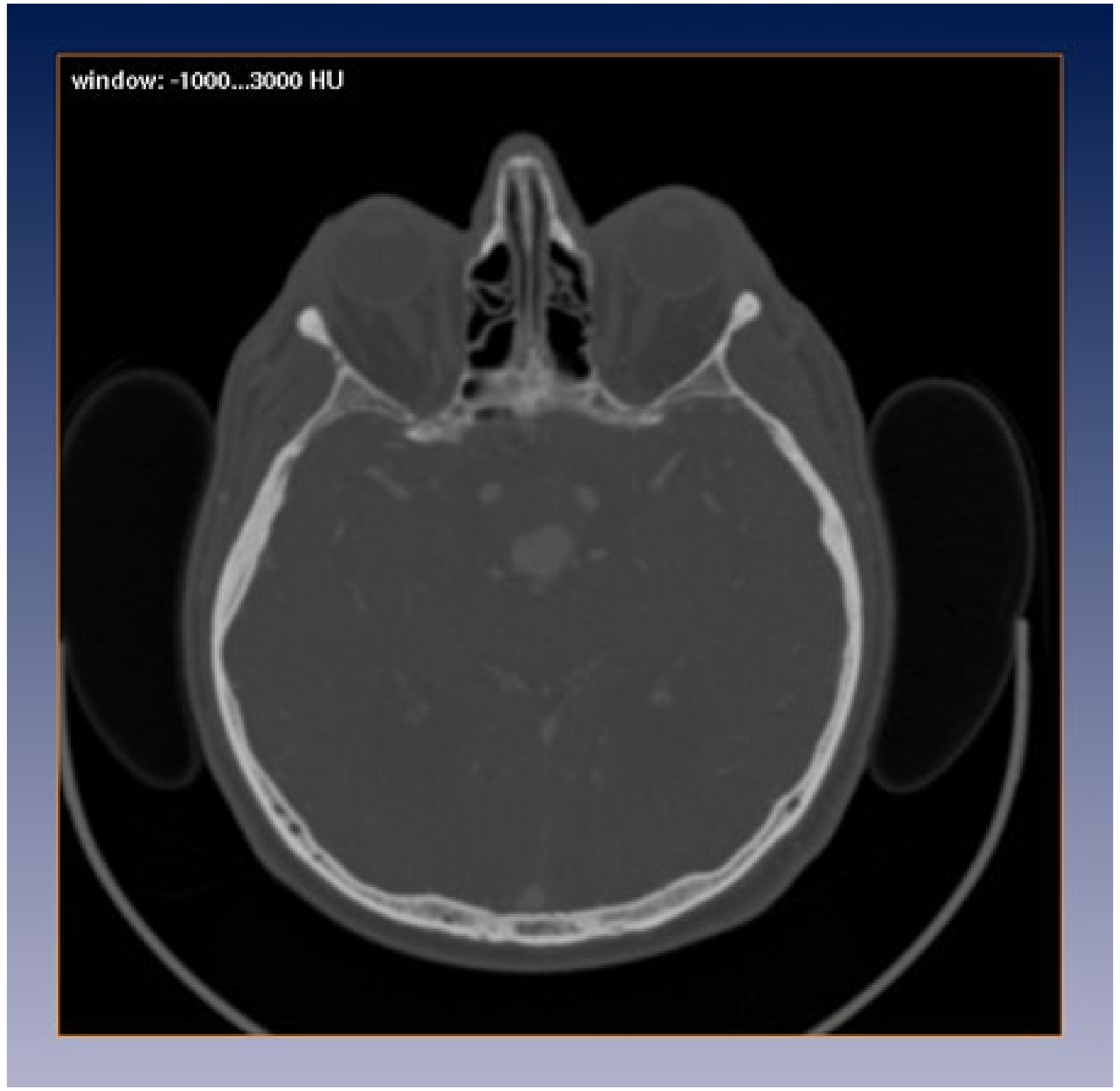
Example - CT data

- Linear ramp transfer function

- f1: [-3000, 1000] → [0, 255]



- All data values visible but no details displayed

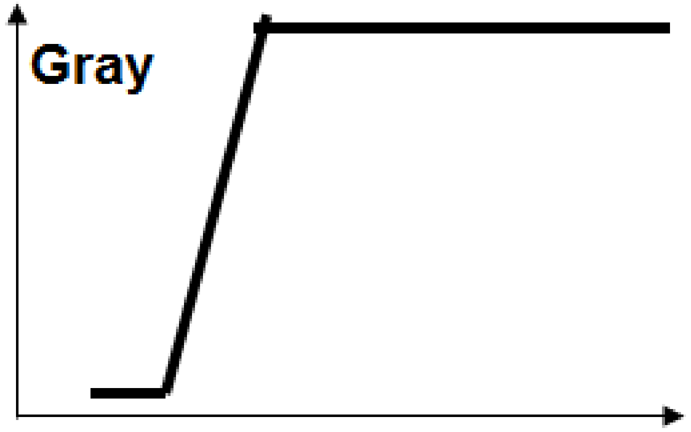


Classification

Example - CT data

- Center: 0 (water)
  Width: 500

- f2: [-250, 250] → [0, 255]



- Blood is highlighted

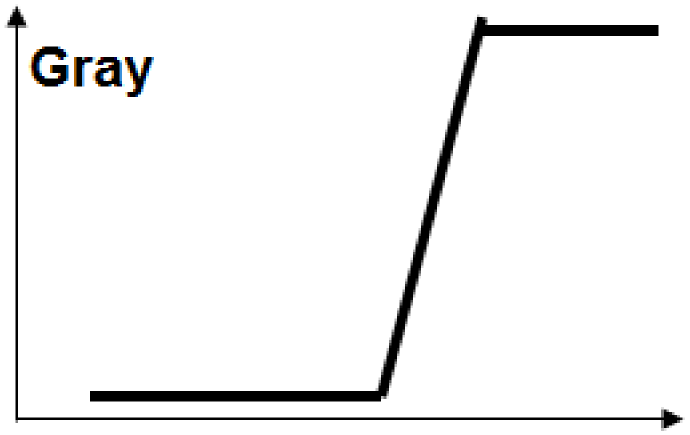
ohm

Classification

Example - CT data

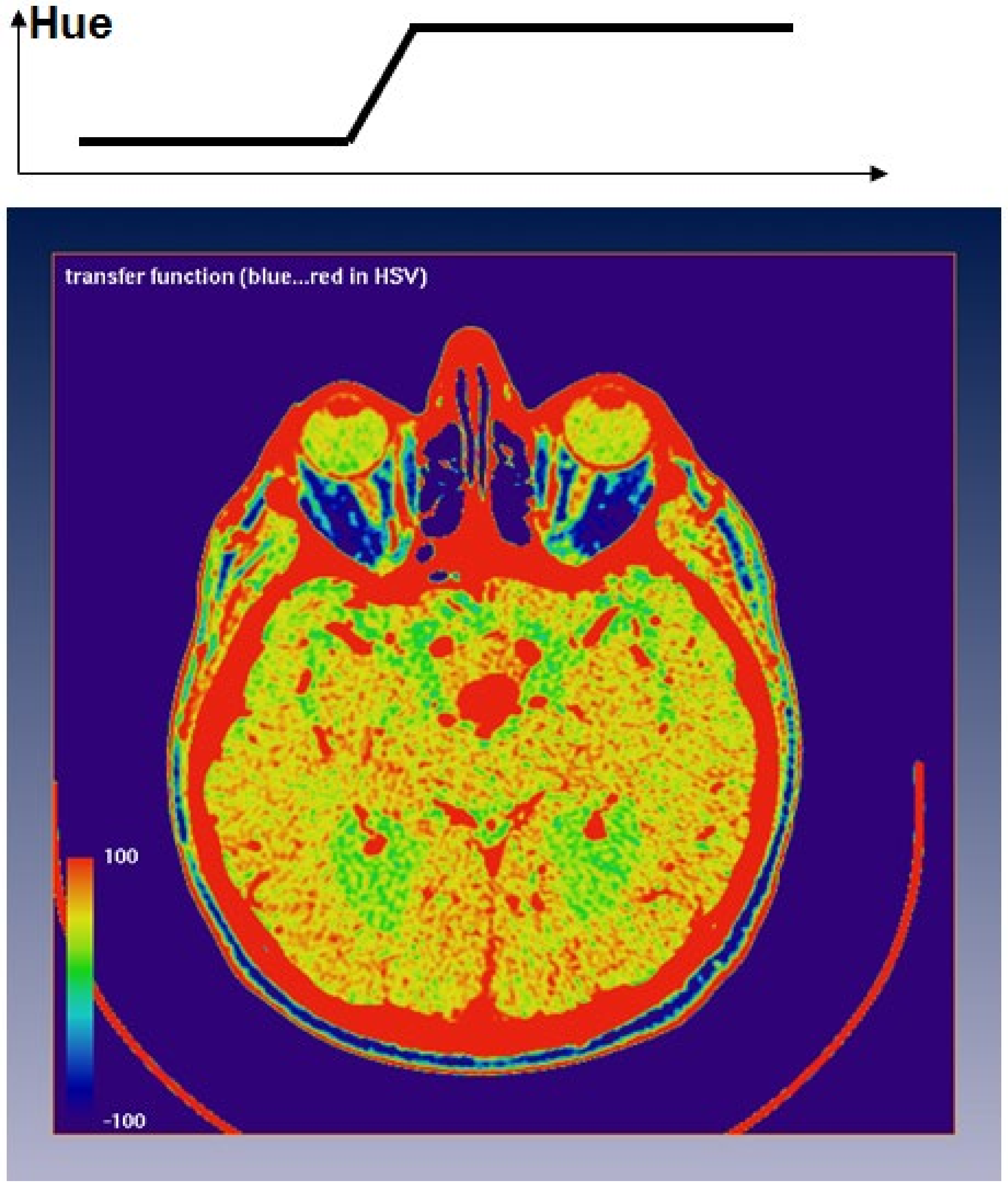
- Center: 1000 (bone)
  Width: 500

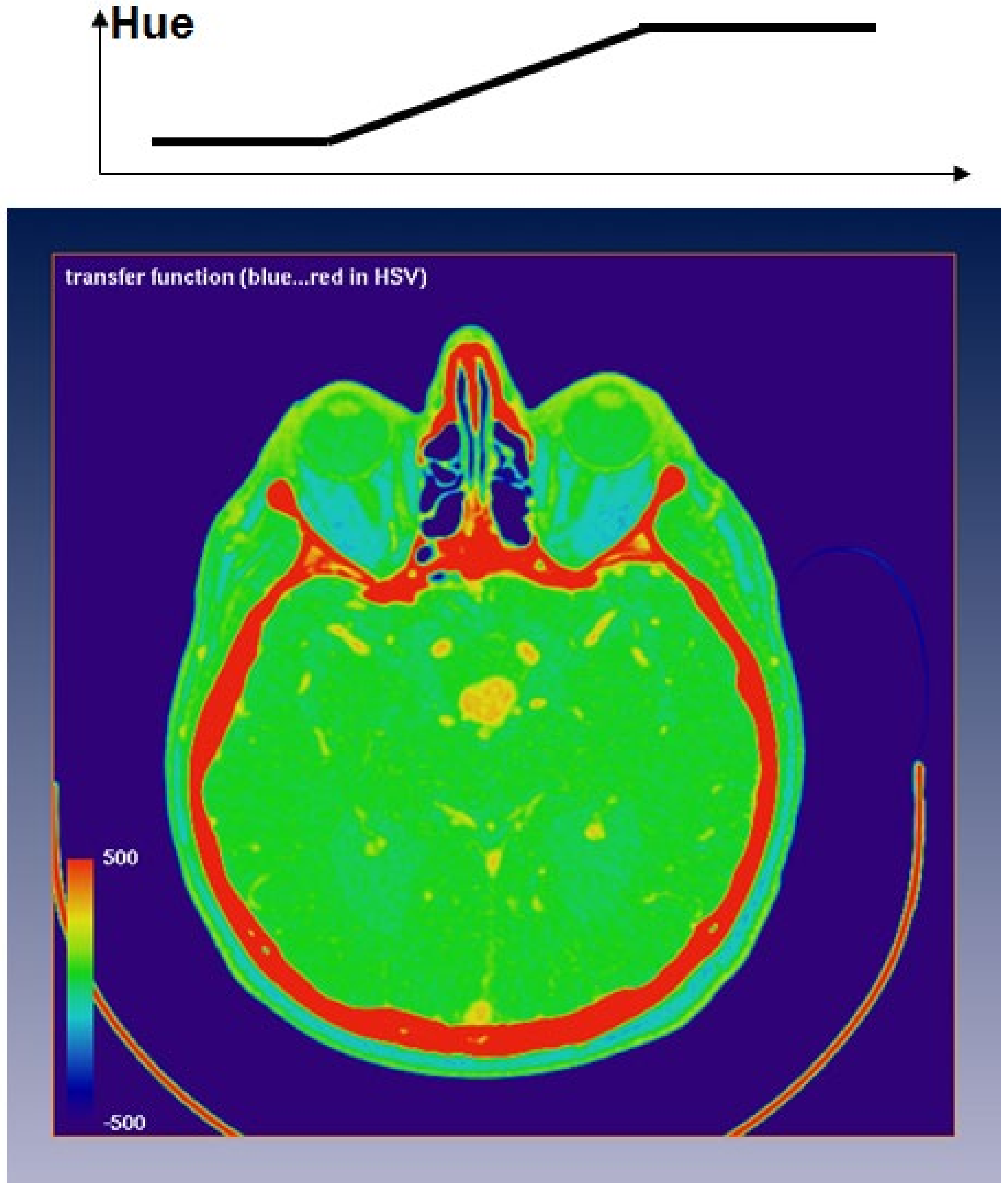
- f2: [750,1250] → [0, 255]



- Only bone is displayed

Classification





6.4 Indirect Volume Rendering

Indirect volume rendering

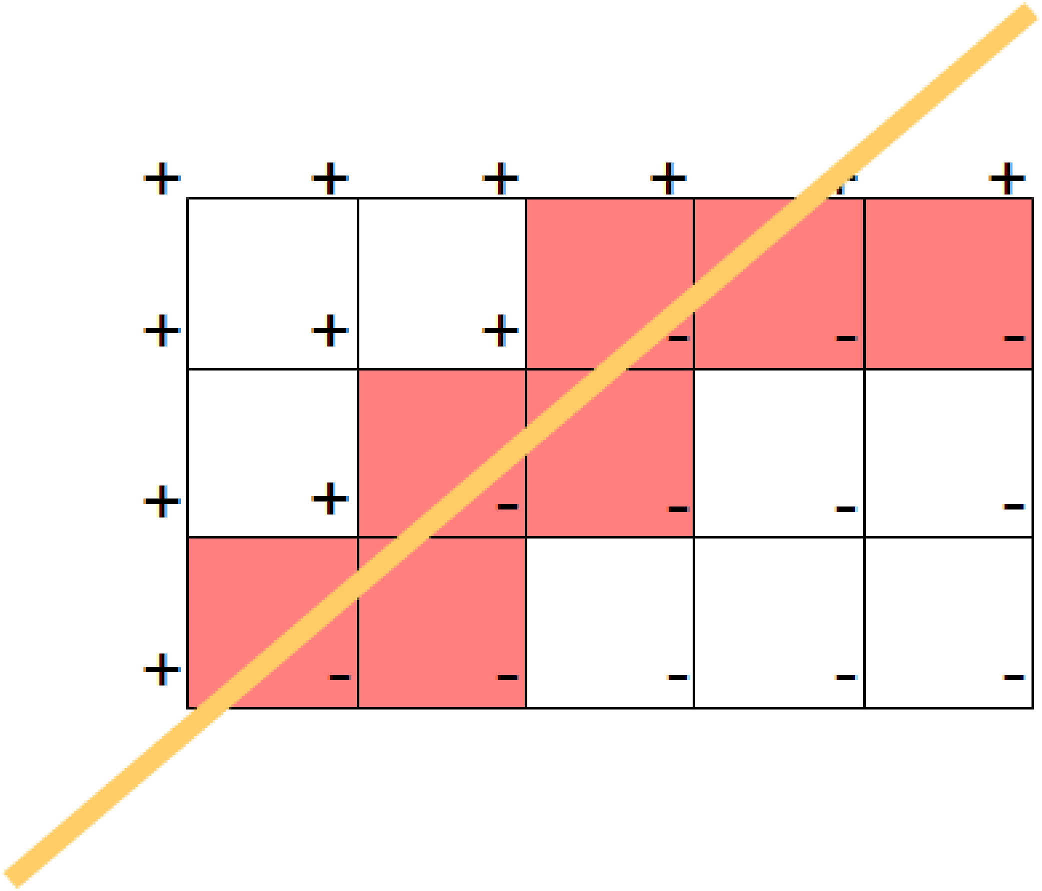
Introduction

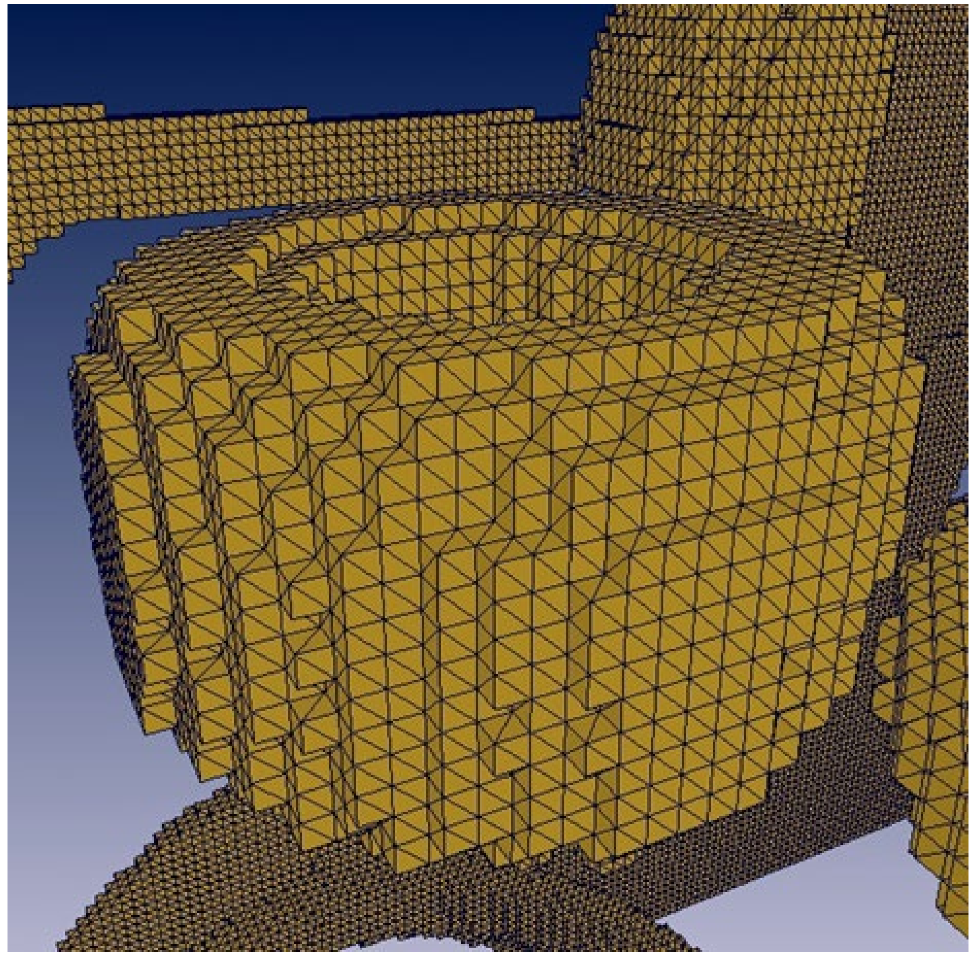
- If f(x,y,z) is differentiable in every point, then the level sets {(x,y,z)|f(x,y,z) = c} are smooth surfaces (denoted iso-surfaces) to the iso-value c
- Techniques to reconstruct isosurfaces
  - Obaque cube (Cuberille method), Dividing cube
  - Marching cube
  - Marching tetrahedra
  - Surfaces from contours (e.g. NUAGES)
- While opaque cube and marching cube methods are restricted to structured grids, the marching tetrahedra method can deal with any kind of mesh

6.4.1 Opaque Cubes

Opaque Cubes

- Approach [Herman, 1979]
  - Volume data on regular grid
    - Detect all cells that intersect the iso-surface f = c by checking vertices
    - If fi,j,k > c mark vertex (i,j,k) with +, otherwise with -
    - Find all boundary front-faces
      - Faces where normal points towards viewpoint (N·V > 0) and where normal points outwards the cell
    - Render these faces as shaded polygons
  - "Voxel" point of view: NO interpolation within cells

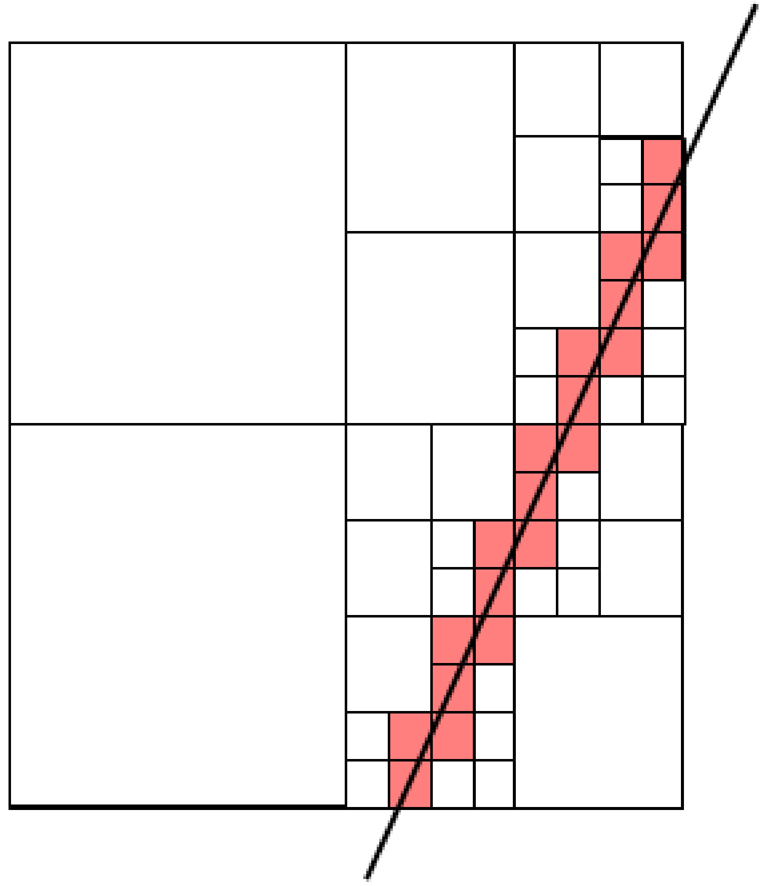




ohm

Opaque Cubes

- Evaluation
  - Method yields blocky surfaces since cell is either opaque or not
  - Improvement through adaptive subdivision
    - Subdivide each marked cube into eight smaller cubes
    - Use trilinear interpolation to determine the function value in the center
    - Repeat the cuberille approach for each new cube until pixel size
    - This is known as "dividing cubes"



6.4.2 Marching Cubes

Marching Cubes

-> Oberflächen

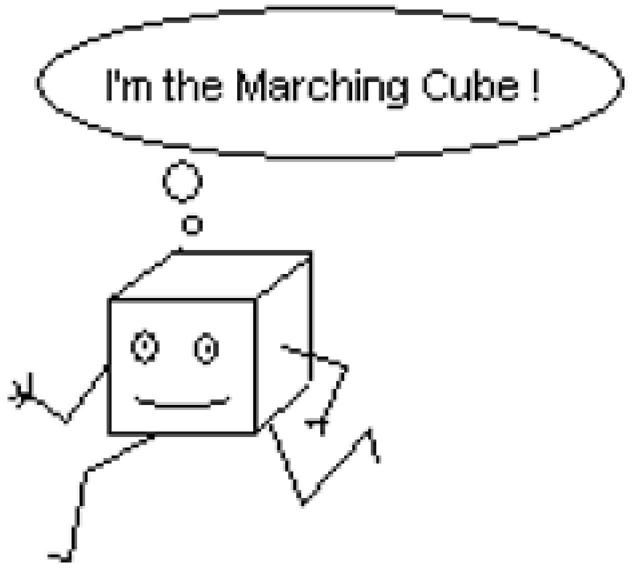
- Approach [Lorensen, Cline, 1987]
  - Better approximation of the "real" iso-surface
    - 3D analog to the isoline construction method (marching squares)
  - Strategy
    - Works on the original data
    - Approximates the surface by a triangle mesh
    - Surface found by linear interpolation along cell edges
    - Uses gradients as the normal vectors of the iso-surface
    - Efficient computation by means of lookup tables

- THE standard algorithm for geometry-based iso-surface extraction

Marching Cubes

- The core algorithm
  - Cell consists of 4 (8) pixel (voxel) values
    - (i+[0/1], j+[0/1], k+[0/1])

  - Consider a cell
  - Classify each vertex as inside or outside
  - Build an index
  - Get edge list from table[index]
  - Interpolate the edge location
  - Compute gradients
  - Consider ambiguous cases
  - Go to next cell

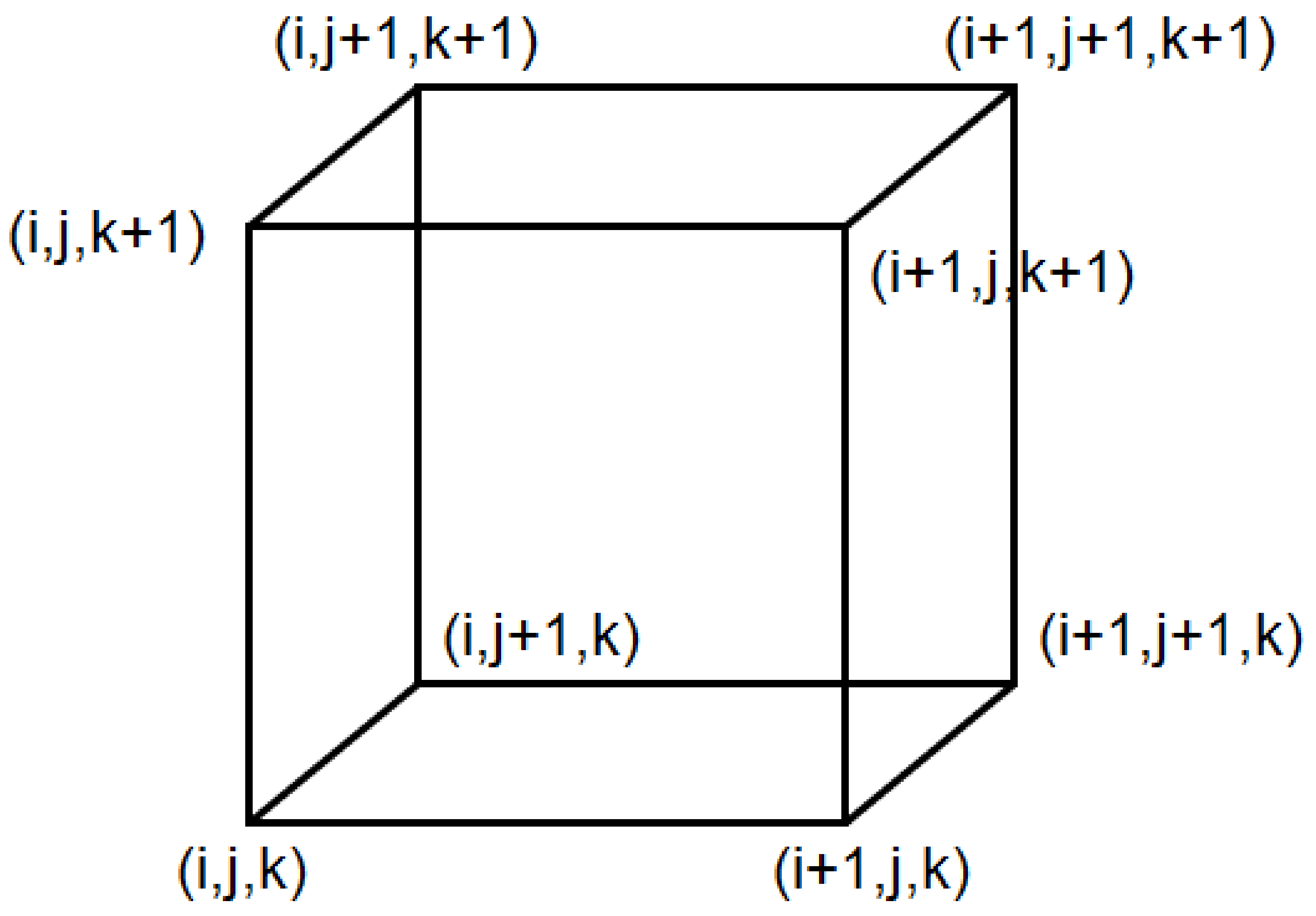


ohm

Marching Cubes

Step 1

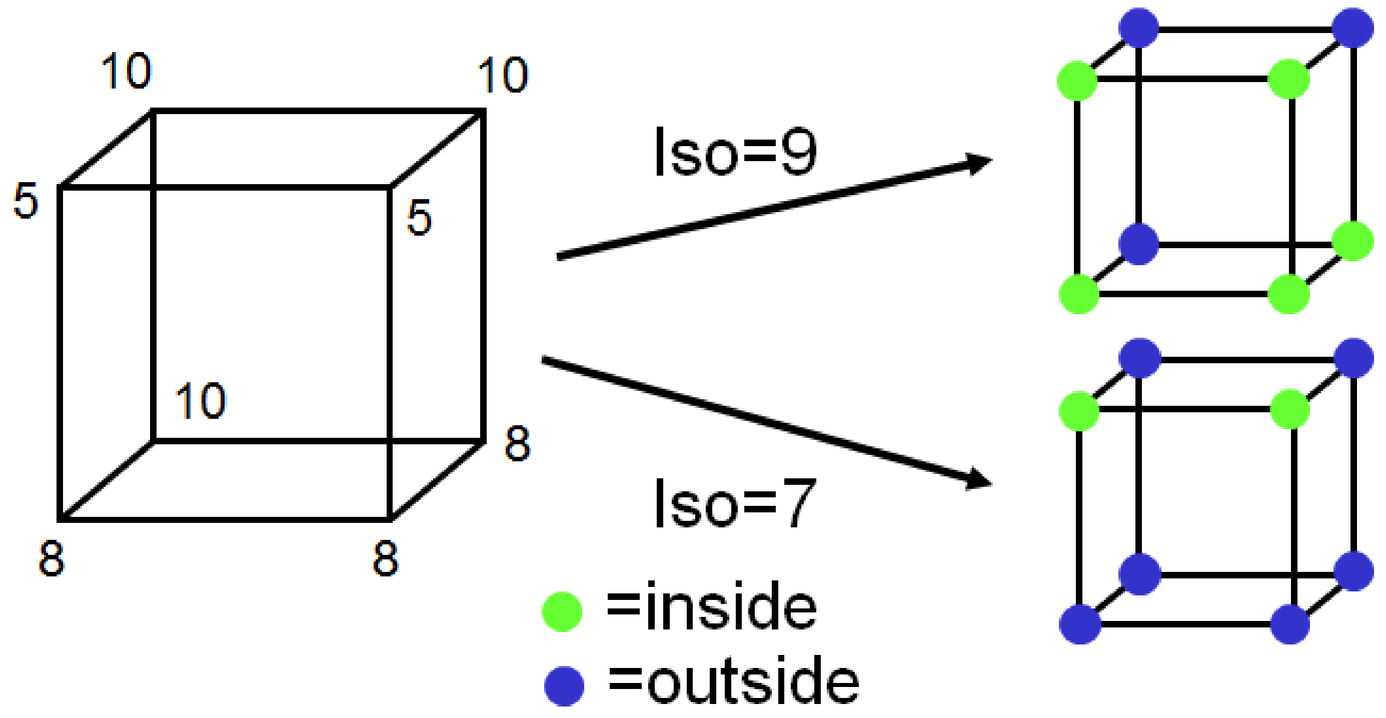
- Consider a cell defined by eight data values



Marching Cubes

Step 2

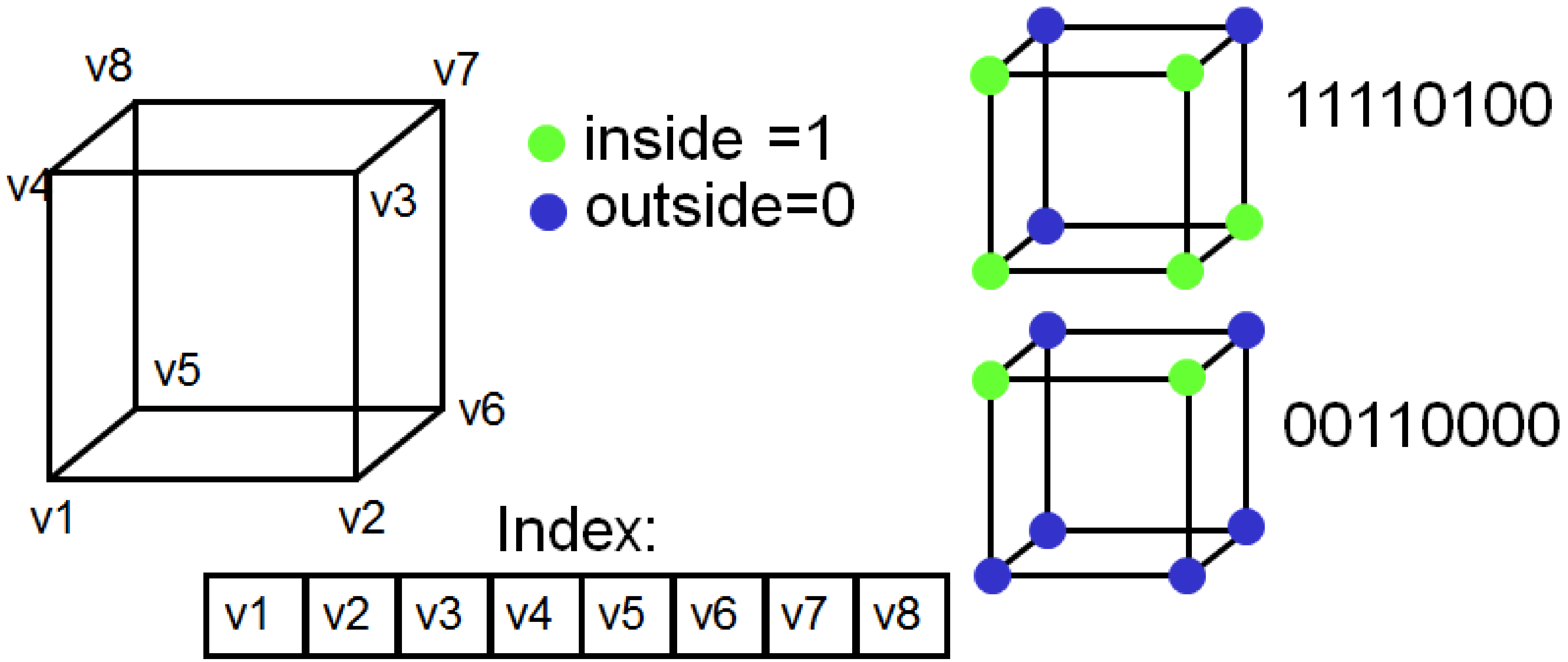
- Classify each voxel according to whether it lies
  - Outside the surface (value > iso-surface value)
  - Inside the surface (value ≦ iso-surface value)



Marching Cubes

Step 3

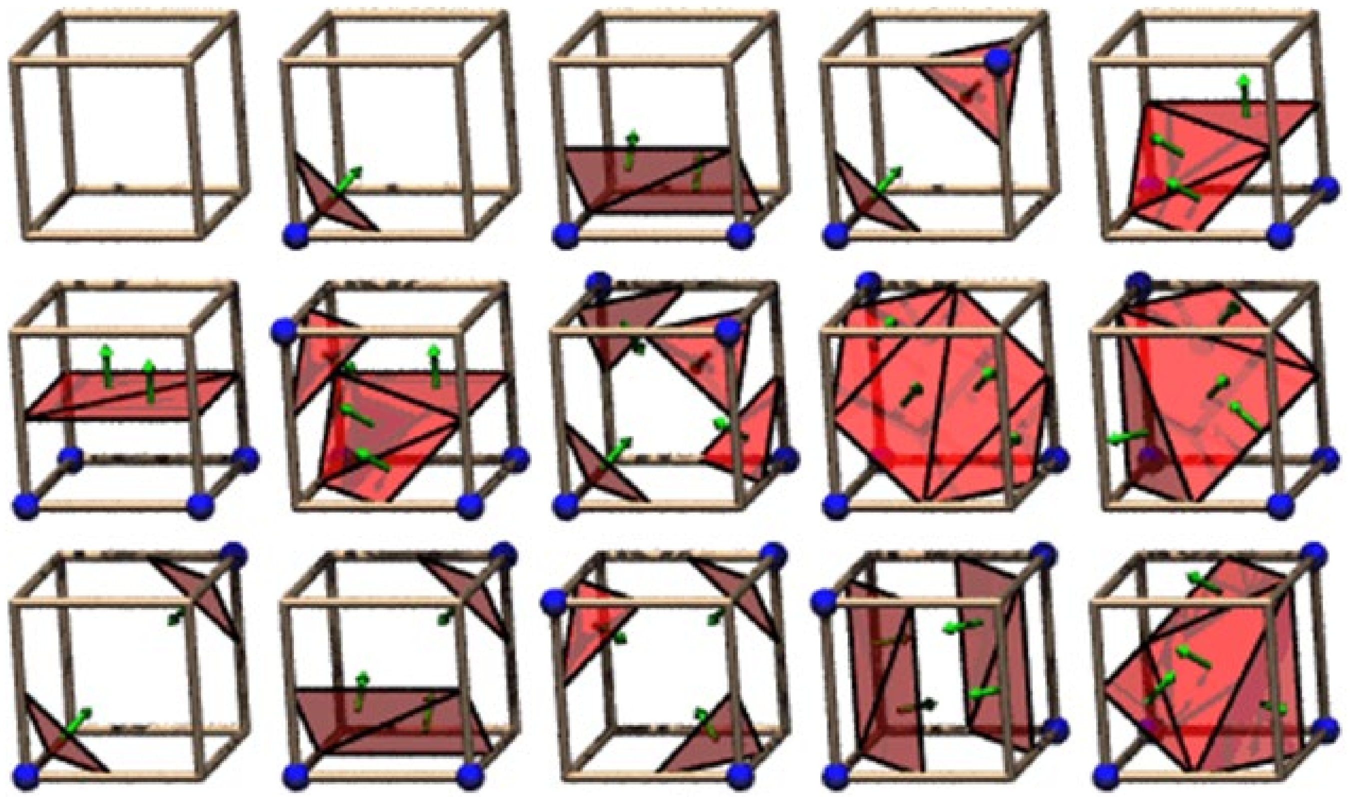
- Use binary labeling of each voxel to create an index



Marching Cubes

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



The 15 Cube Combinations

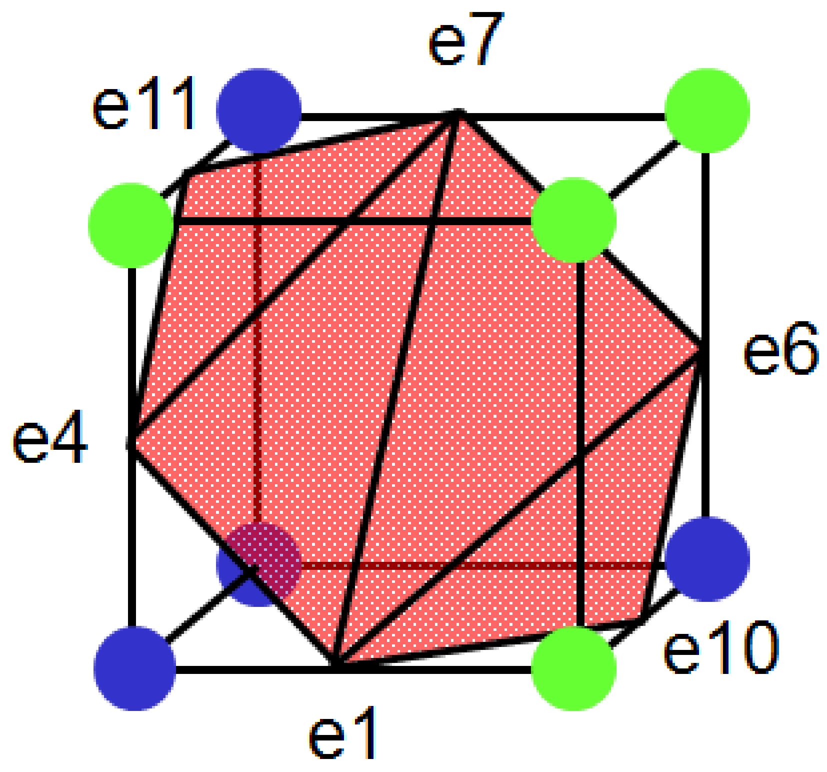
Marching Cubes

Step 4 (cont.)

- Get edge list from table
  - Example for

    Index = 10110001

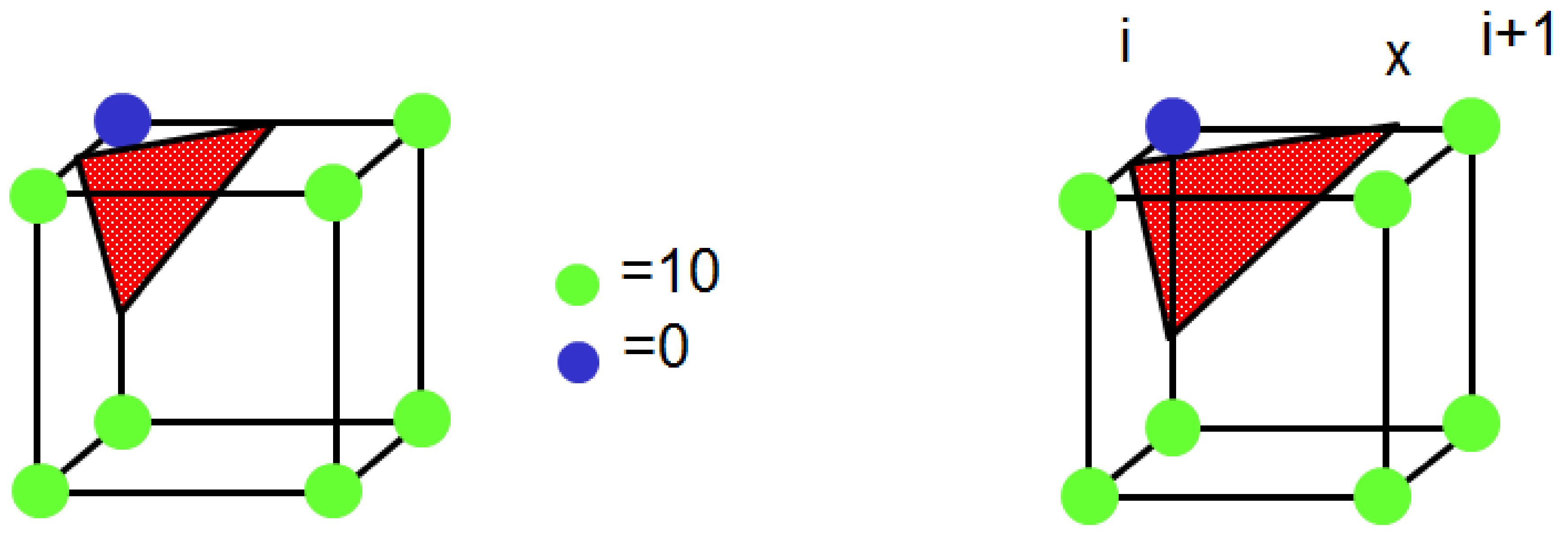
    tri 1 = e4, e7, e11
    tri 2 = e1, e7, e4
    tri 3 = e1, e6, e7
    tri 4 = e1, e10, e6



Marching Cubes

Step 5

- For each triangle edge, find the vertex location along the edge using linear interpolation of the voxel values



T=5

T=8

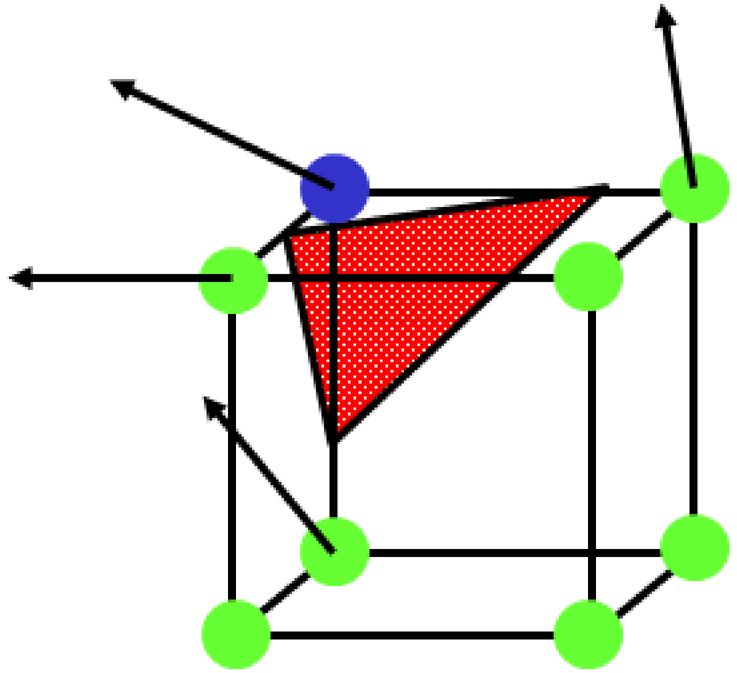
$$x = i + \left(\frac{T - v[i]}{v[i+1] - v[i]} \right)$$

Marching Cubes

Step 6

- Calculate normal at each cube vertex (central differences)
  - $G_x = V(x+1,y,z) - V(x-1,y,z)$
    $G_y = V(x,y+1,z) - V(x,y-1,z)$
    $G_z = V(x,y,z+1) - V(x,y,z-1)$

  - Use linear interpolation to compute the polygon vertex normal (of the iso-surface)

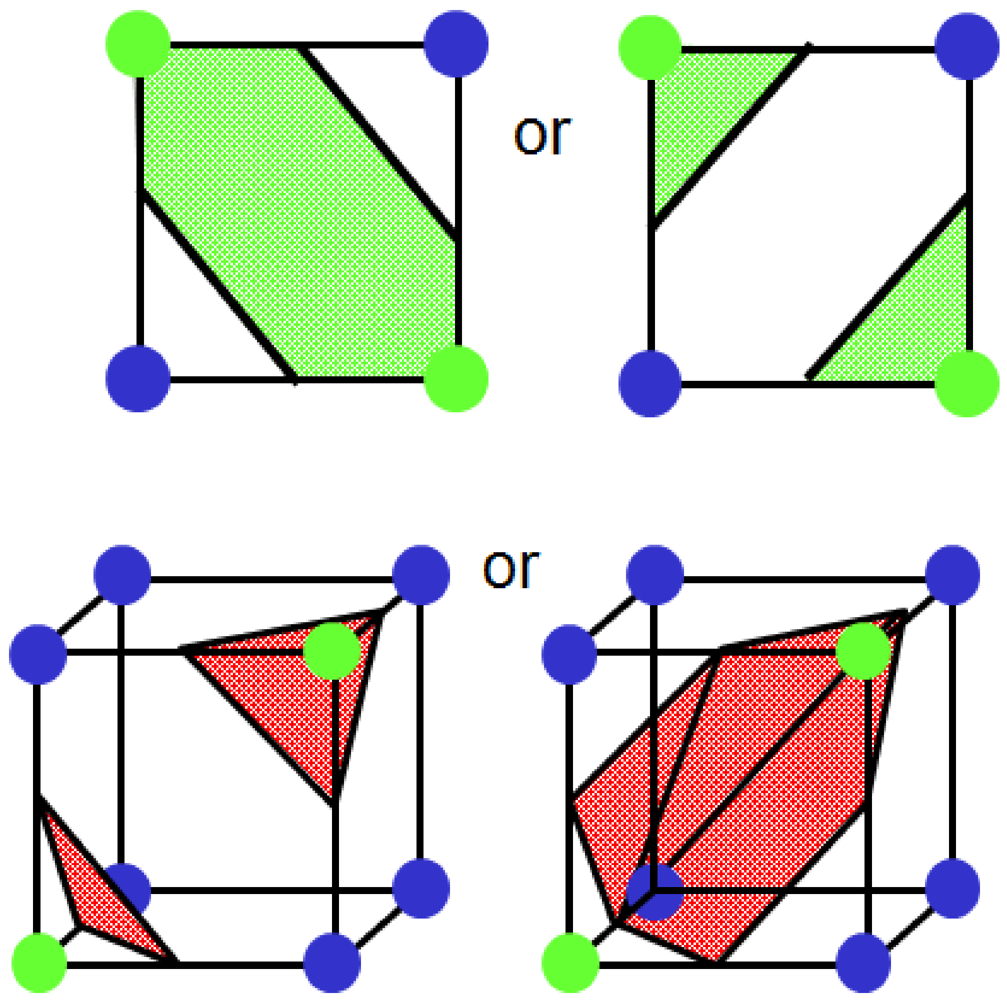


ohm

Marching Cubes

Step 7

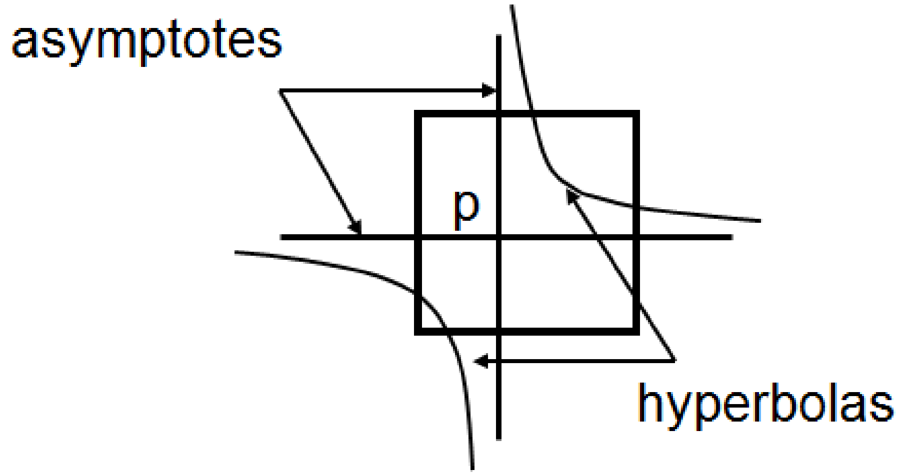
- Consider ambiguous cases
  - Ambiguous cases
    - 3, 6, 7, 10, 12, 13
  - Adjacent vertices:
    - Different states
  - Diagonal vertices:
    - Same state
  - Resolution
    - Choose one case (the right one!)



Marching Cubes

Step 7 (cont.)

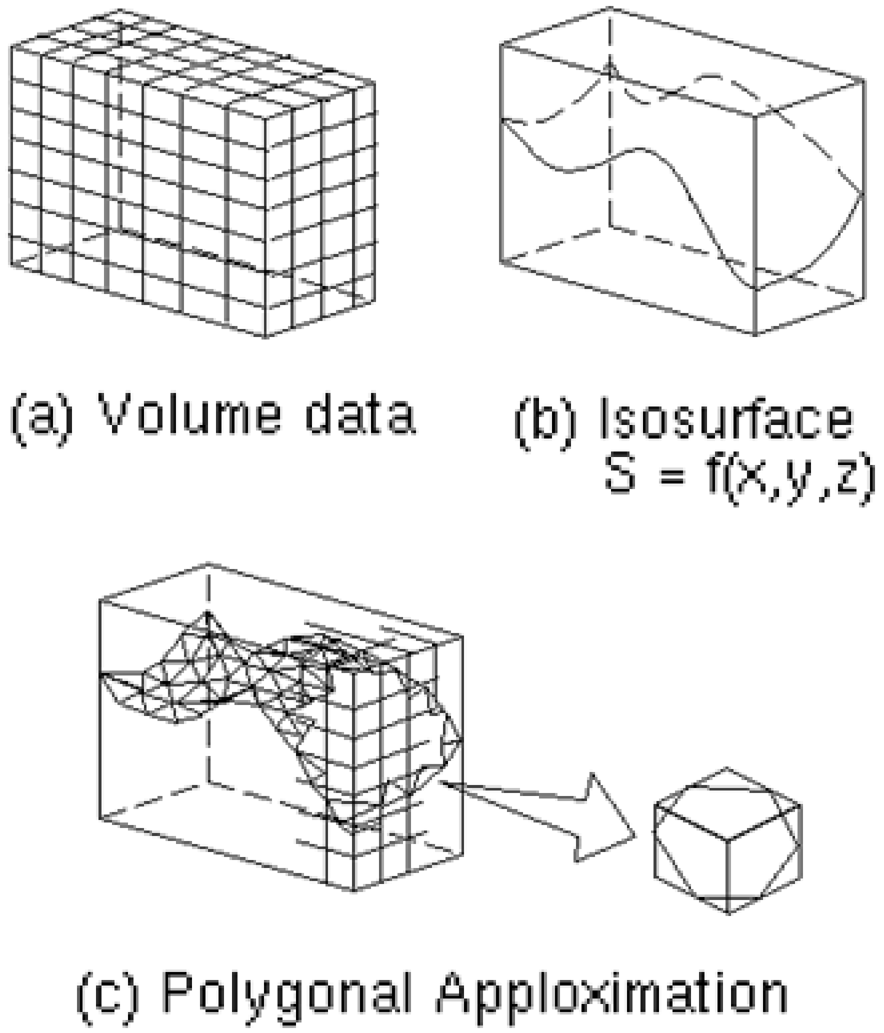
- Consider ambiguous cases
- Asymptotic decider [Nielson, Hamann, 1991] (comp. marching squares!)
  - Assume bilinear interpolation within a face
  - Hence iso-surface is a hyperbola
  - Compute point p where asymptotes meet
  - Sign of S(p) decides on the connectivity
    - This is analog to the 2D case



Marching Cubes

Evaluation

- Up to 5 triangles per cube
- Dataset of 512^3 voxels
  - Can result in several millions of triangles (many Mbytes!)
- Both very big and very small triangles
  - Post-processing is necessary to get a "good" triangular mesh
- Many special cases
  - Ambiguity in cases can cause holes if arbitrary choices are made
  - Special cases at the boundaries of the volume



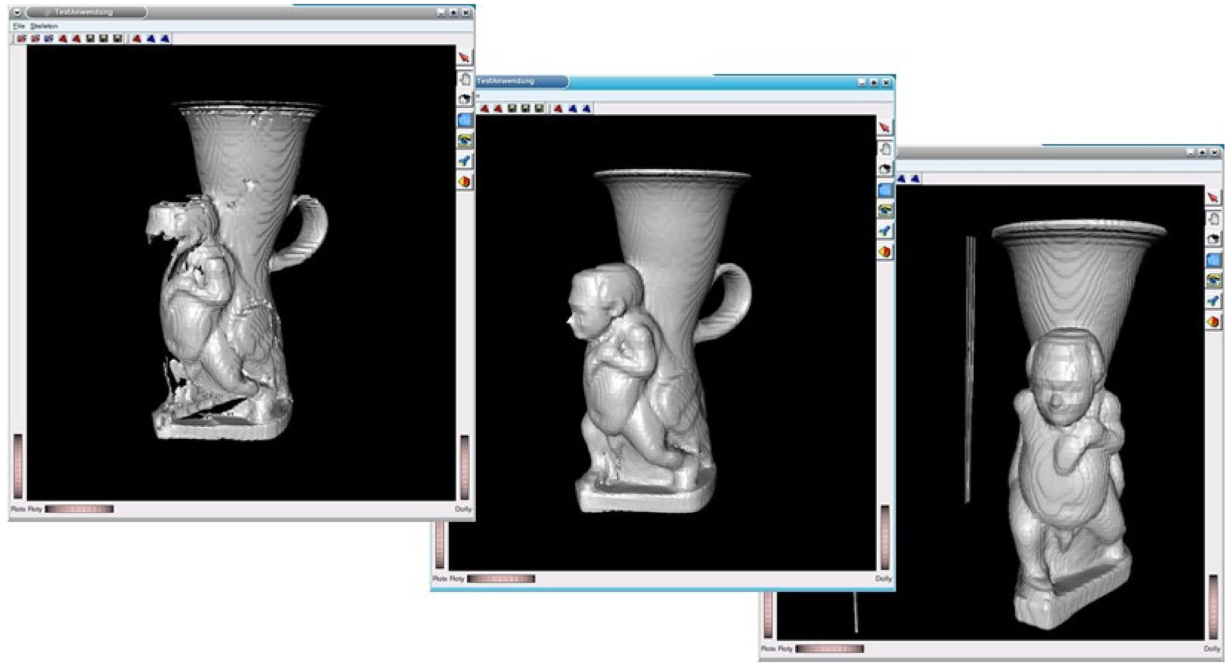
Marching Cubes

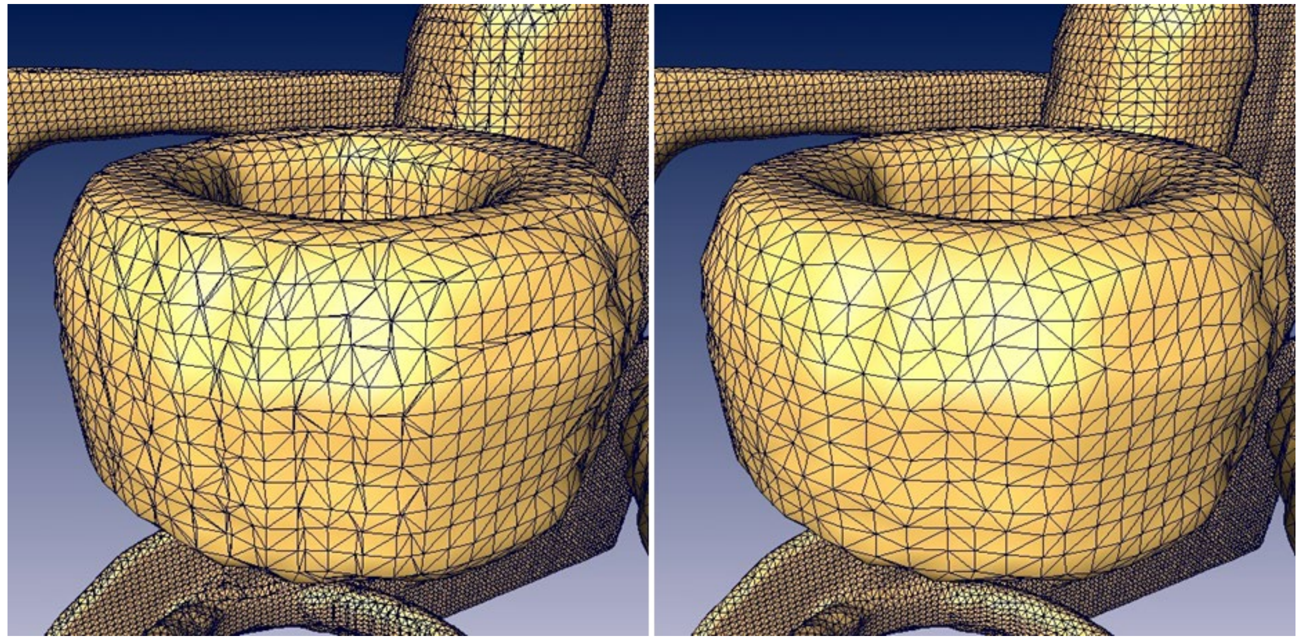
Enhancements

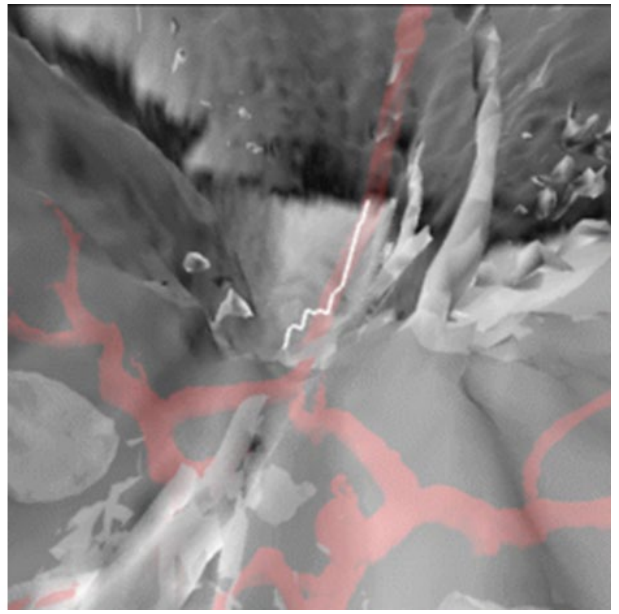
- Semi-transparent representation
  - Requires sorting
- Optimization
  - Reuse values from prior calculations
  - Prevent vertex replication
  - Mesh simplification
- Accelerated display
  - Graphics hardware
- High quality display
  - Ray-tracing algorithm

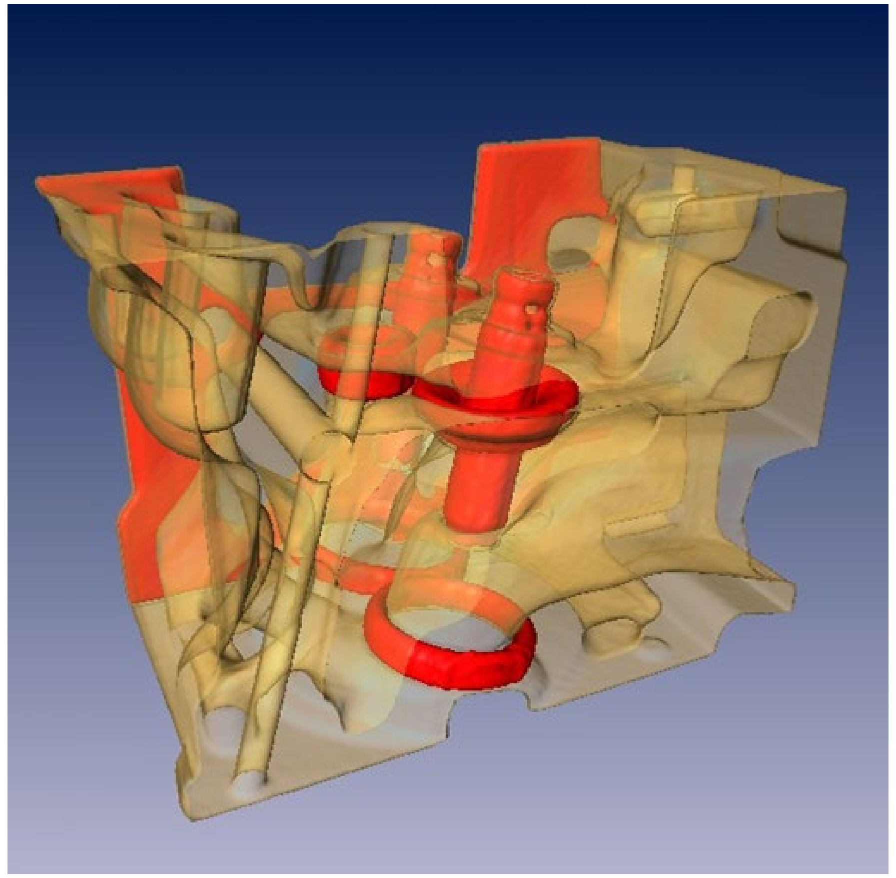
Marching Cubes

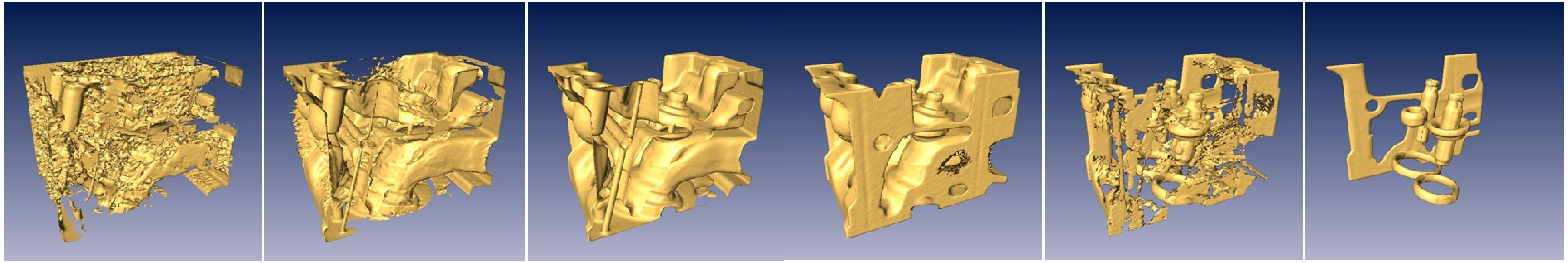
Examples











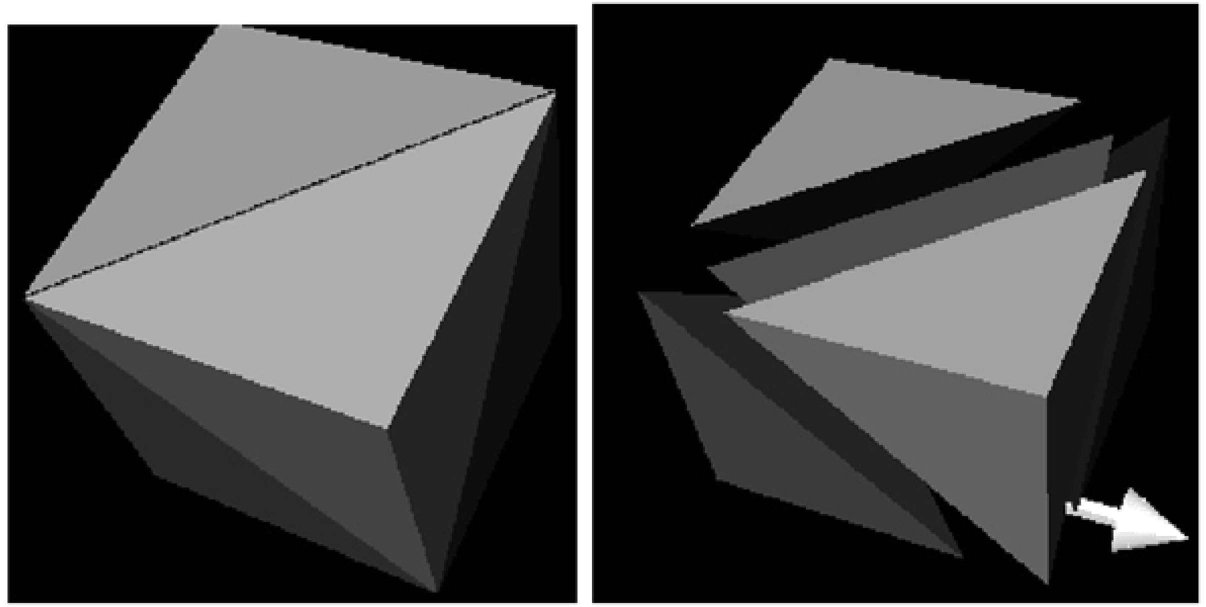
f = 5 (bad) | f = 13 (bad) | f = 45 (good) | f = 122 (good) | f = 146 (bad) | f = 170 (good)

6.4.3 Marching Tetrahedra

Marching Tetrahedra

Alternative to Marching Cubes

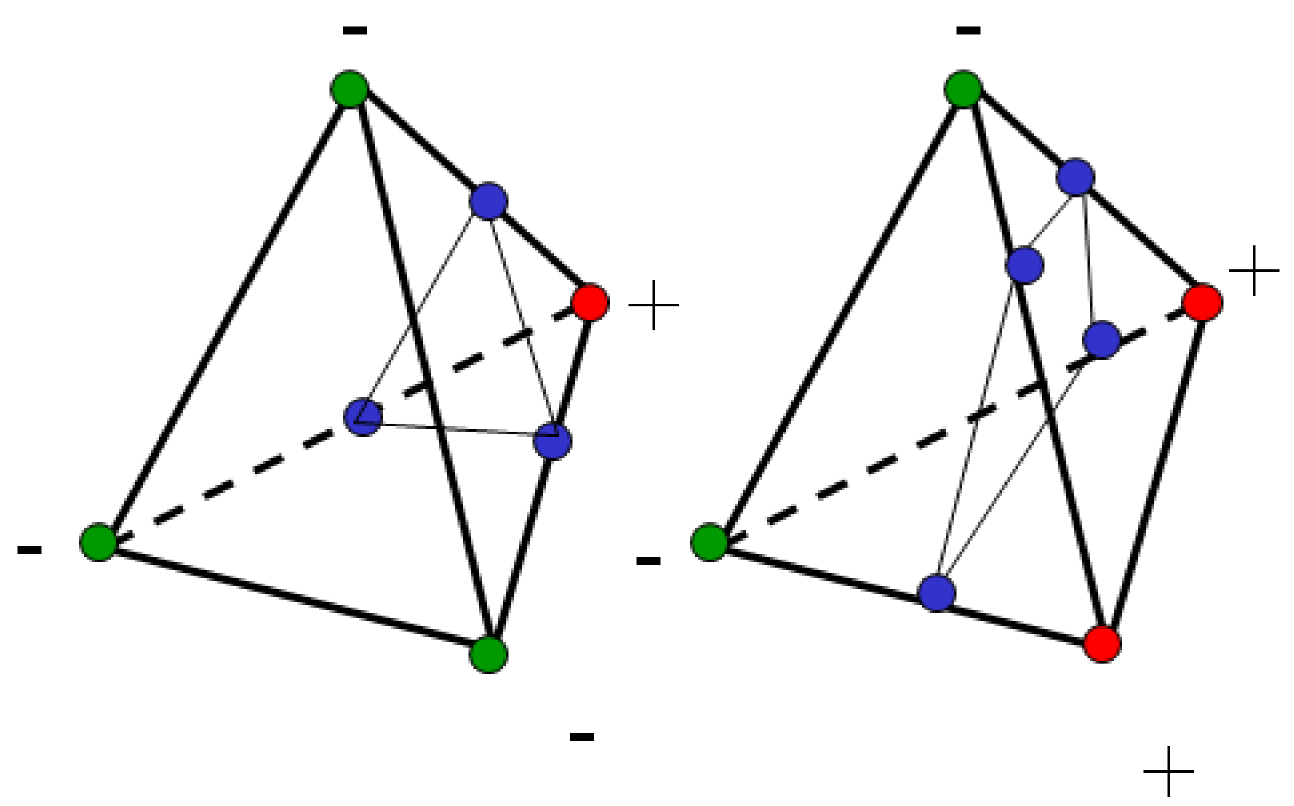
- Split each cube into five or six tetrahedra
  - Depending on the method used for splitting the cube
- Example for 5 tetrahedra



- Yields more triangles and "rougher" surface than MC

Marching Tetrahedra

- Approach
  - Primarily used for unstructured grids
  - Process each cell similarly to the MC-algorithm
    - Mark vertices with + or -, depending on f > c or not
  - Only two possible non-trivial cases for a tetrahedron
    - One "-" and three "+" (or vice versa)
      - Intersection surface is a triangle
    - Two "-" and two "+"
      Intersection surface
      is a quadrilateral
      - Split into two triangles
        using the shorter diagonal



Marching Tetrahedra

- Properties
  - Fewer cases, i.e. 3 instead of 15
    - Linear interpolation within cells
    - No problems with consistency between neighboring cells
  - Also many triangles of different size
    - Number of generated triangles might increase considerably compared to the MC algorithm due to splitting into tetrahedra
    - Huge amount of geometric primitives
  - But, several improvements exist
    - Hierarchical surface reconstruction
    - View-dependent surface reconstruction
    - Mesh decimation

6.4.4 Surfaces from Contours

Surfaces from Contours

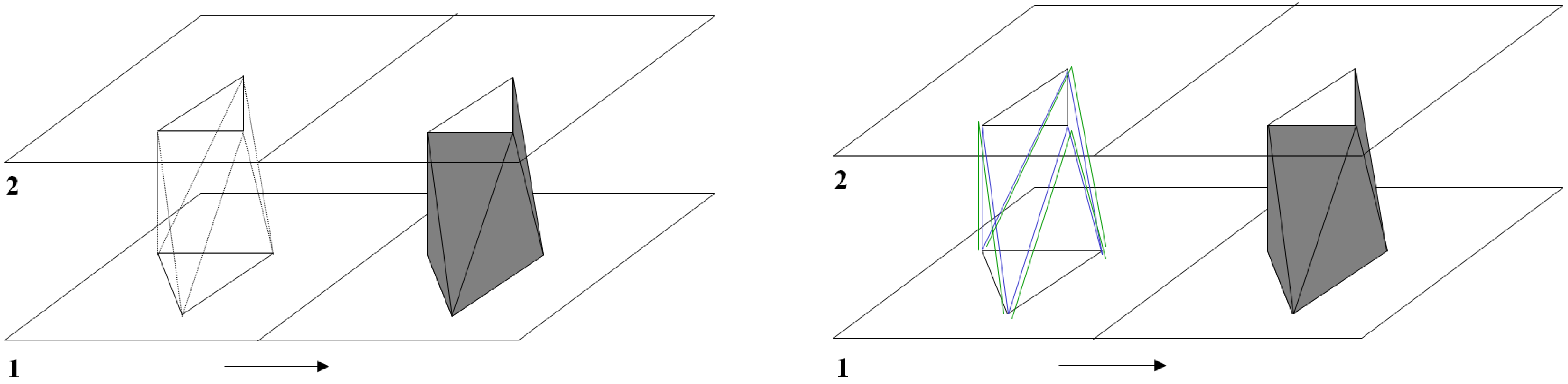
- Approach
  - Explicit segmentation
    - Find closed contours in successive 2D slices
      - Often semi-automatic procedure with user-interaction!
      - Requires a lot of expertise
    - Represent contours as poly-lines
  - Labeling
    - Identify different structures: e.g. brain, vessels, ...
  - Reconstruction
    - Connect contours representing the same object from neighboring slices and form triangles
  - Rendering
    - Display triangles

ohm

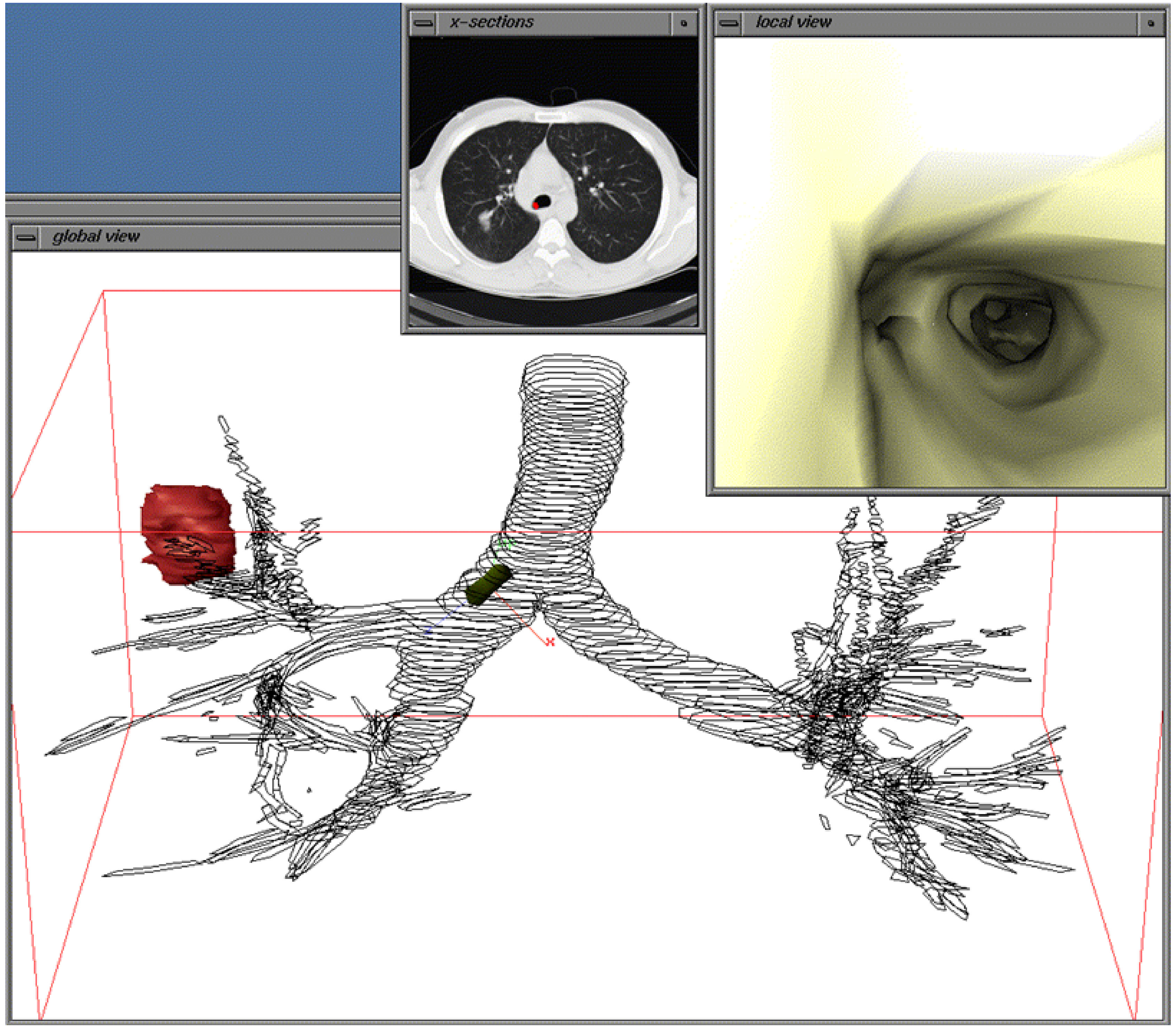
Surfaces from Contours

Example

- Identify for edge on level 2 corresponding vertex on level 1
- Identify for edge on level 1 corresponding vertex on level 2



Surfaces from Contours



Surfaces from Contours

- Problems
  - Many contours in each slice
  - There is a high variation between slices
    - Correspondence between neighboring slices can be difficult
    - Branching has to be handled (not always easy)

- Literature
    - B. Geiger: NUAGES (INRIA France, 1993) - see paper and software http://www-sop.inria.fr/prisme/fiches/Medical/index.html.en
    - F. Cazals, J. Giesen, Delaunay Triangulation Based Surface Reconstruction: Ideas and Algorithms, INRIA, Tech Report 2004
    - D. Wang, O. Hassan, K. Morgan, N. Weatherill, Efficient surface reconstruction from contours based on two-dimensional Delaunay triangulation, Int. J. Numer. Meth. Engng 2006; 65:734–751

Summary

ohm

Summary

- Advantages
  - Unambiguous definition of a surface
  - Clear visual impression (light!)
  - Use standard graphics hardware for rendering
- Disadvantages
  - Explicit segmentation required
    - Difficult and time consuming
  - Exact surface
    - Ill-posed problem in regions of little data value variation
  - Information reduction
  - Too many triangles in case of complex structures