CSCI 420 Computer Graphics
Lecture 22

Visualization

Height Fields and Contours
Scalar Fields
Volume Rendering
Vector Fields
[Angel Ch. 11]

Oded Stein
University of Southern California

Scientific Visualization

- Generally do not start with a 3D triangle model
- Must deal with very large data sets
  - MRI, e.g. 512 x 512 x 200 = 50MB points
  - Visible Human 512 x 512 x 1734 = 433 MB points
- Visualize both real-world and simulation data
- User interaction
- Automatic search for relevant data

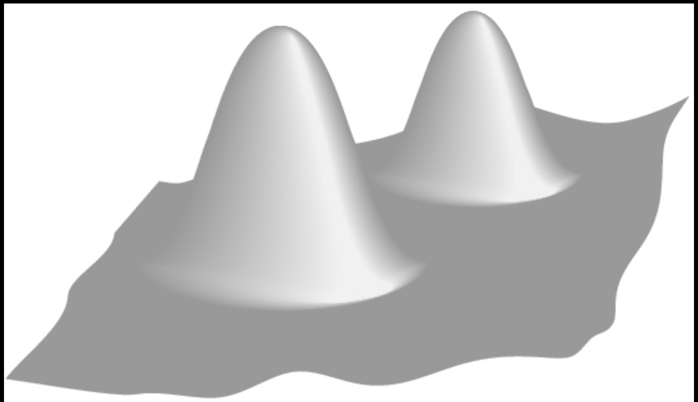
Types of Data

- Scalar fields (3D volume of scalars)
  - E.g., x-ray densities (MRI, CT scan)
- Vector fields (3D volume of vectors)
  - E.g., velocities in a wind tunnel
- Tensor fields (3D volume of tensors [matrices])
  - E.g., stresses in a mechanical part
- Static or dynamic through time

Height Field

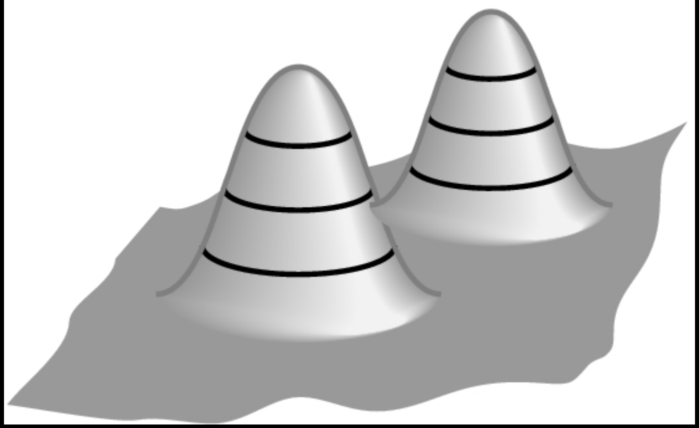
- Visualizing an explicit function

$$z = f(x,y)$$



- Adding contour curves

$$f(x,y) = c$$



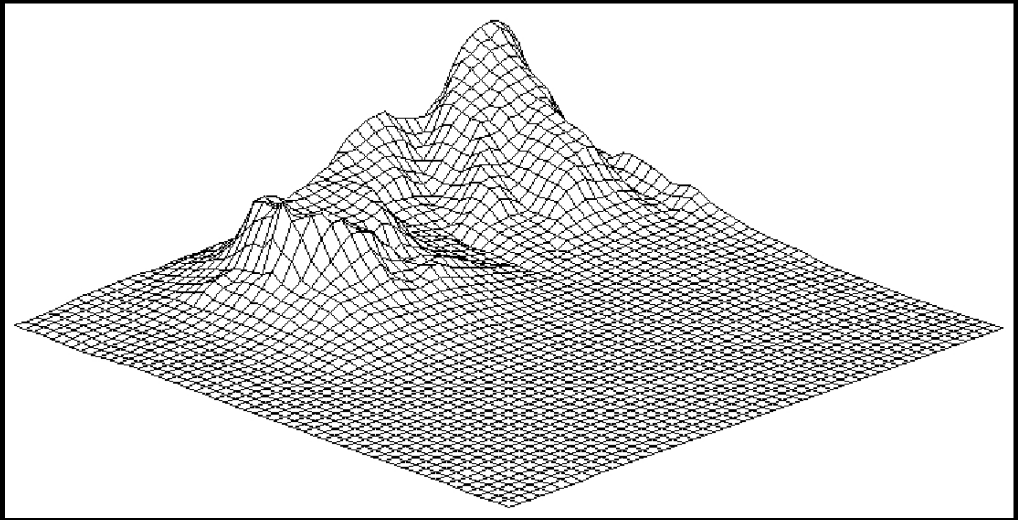
Visualizing the Height Field: Meshes

- Function is sampled (given) at x_i, y_j, $0 \leq i, j \leq n$
- Assume equally spaced

$$x_i = x_0 + i\Delta x$$
$$y_j = y_0 + j\Delta y$$

$$z_{ij} = f(x_i, y_j)$$

- Generate quadrilateral or triangular mesh
- [Assignment 1]



Visualizing the Height Field: Contour Curves

- Recall: implicit curve f(x,y) = 0
- $f(x,y) < 0$ inside, $f(x,y) > 0$ outside
- Here: contour curve at f(x,y) = c
- Implicit function f sampled at regular intervals for x,y

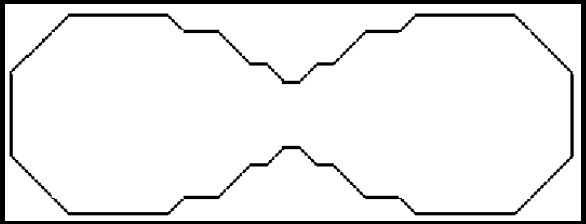
$$x_i = x_0 + i\Delta x$$
$$y_j = y_0 + j\Delta y$$

- How can we draw the curve?

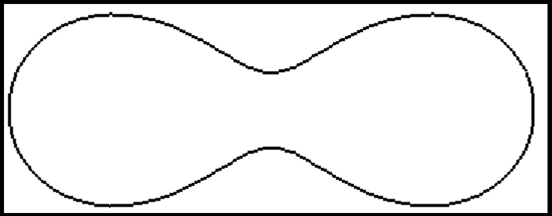
Contour Curves Examples

- Ovals of Cassini, 50 x 50 grid

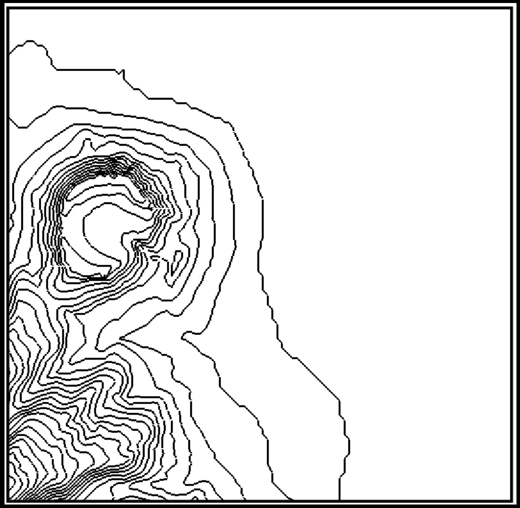
$$f(x,y) = (x^2 + y^2 + a^2)^2 - 4a^2x^2 - b^4$$
$$a = 0.49, b = 0.5$$



Midpoint



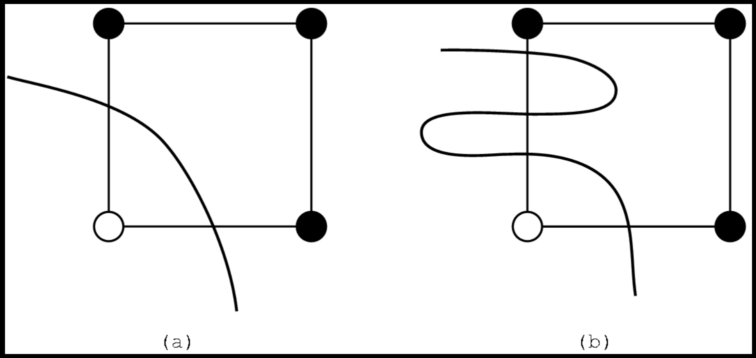
Interpolation



Contour plot of Honolulu data

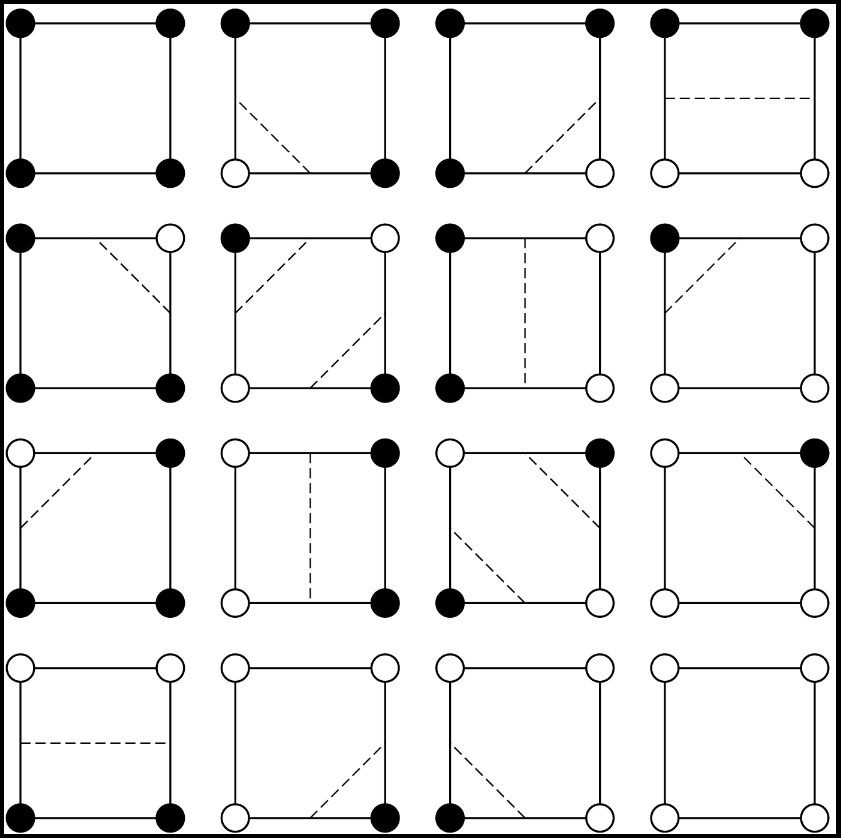
Marching Squares

- Sample function f at every grid point x_i, y_j
- For every point $f_{i\,j} = f(x_i, y_j)$ either $f_{i\,j} \le c$ or $f_{i\,j} > c$
- Distinguish those cases for each corner x
  - White: $f_{i\,j} \le c$
  - Black: $f_{i\,j} > c$
- Now consider cases for curve
- Assume “smooth”

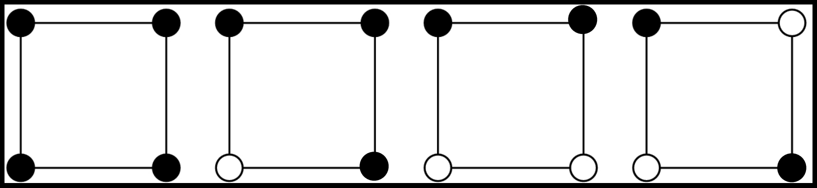


Cases for Vertex Labels

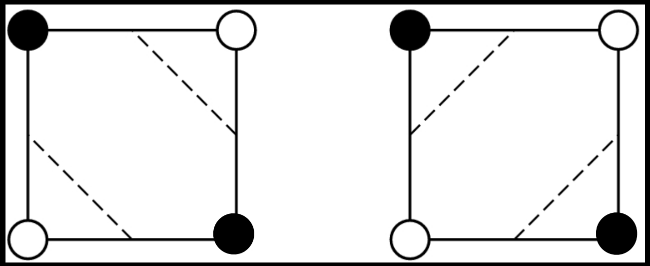
16 cases for vertex labels



4 unique cases modulo symmetries

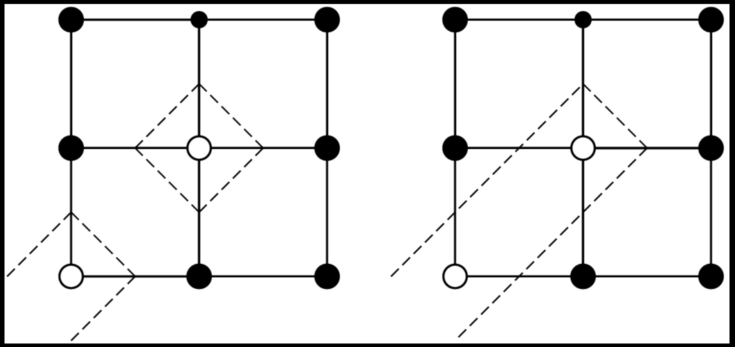


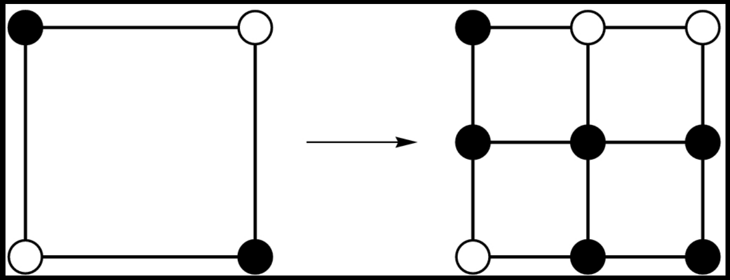
Ambiguities of Labelings



Ambiguous labels

Different resulting contours





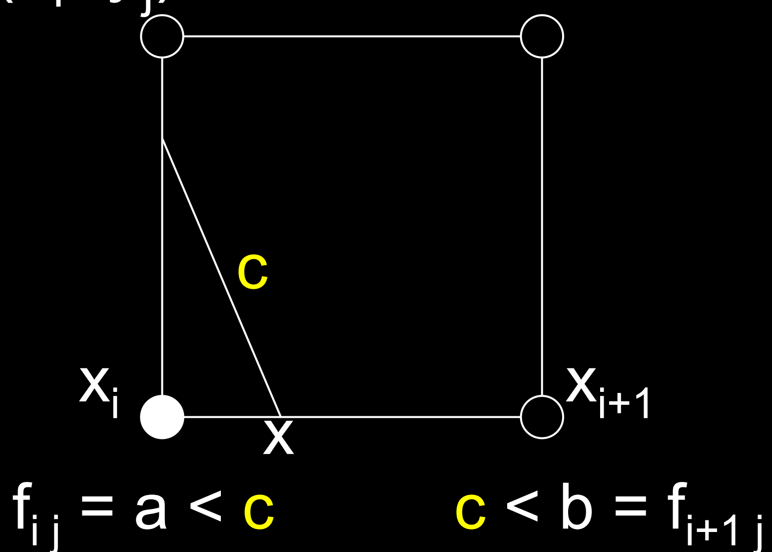
Resolution by subdivision
(if such higher resolution available / possible)

Interpolating Intersections

- Approximate intersection
  - Midpoint between x_i, x_{i+1} and y_j, y_{j+1}
  - Better: interpolate
- If $f_{i\,j} = a$ is closer to c than $b = f_{i+1\,j}$ then intersection is closer to (x_i, y_j):

$$\frac{x - x_i}{x_{i+1} - x} = \frac{c - a}{b - c}$$

- Analogous calculation for y direction



Outline

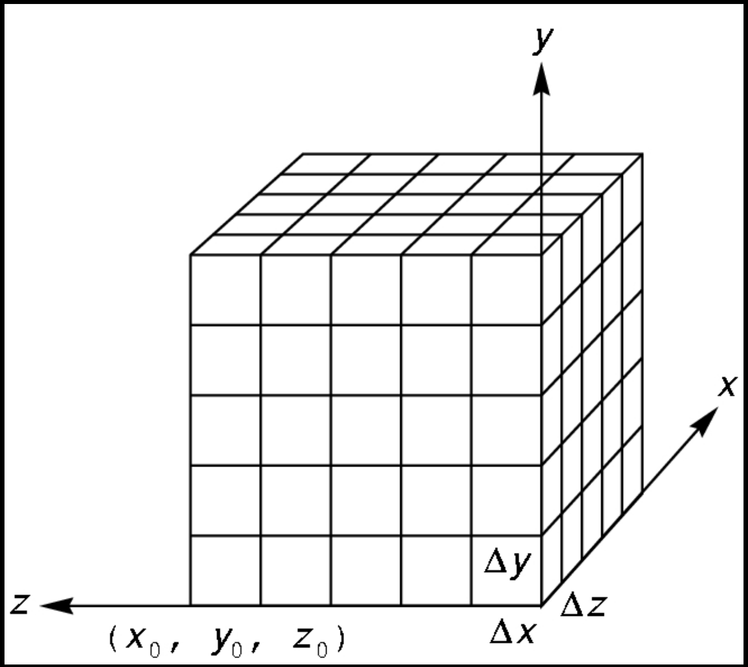
- Height Fields and Contours
- Scalar Fields
- Volume Rendering
- Vector Fields

Scalar Fields

- Volumetric data sets
- Example: tissue density
- Assume again regularly sampled

$$x_i = x_0 + i\Delta x$$
$$y_j = y_0 + j\Delta y$$
$$z_k = z_0 + k\Delta z$$

- Represent as voxels

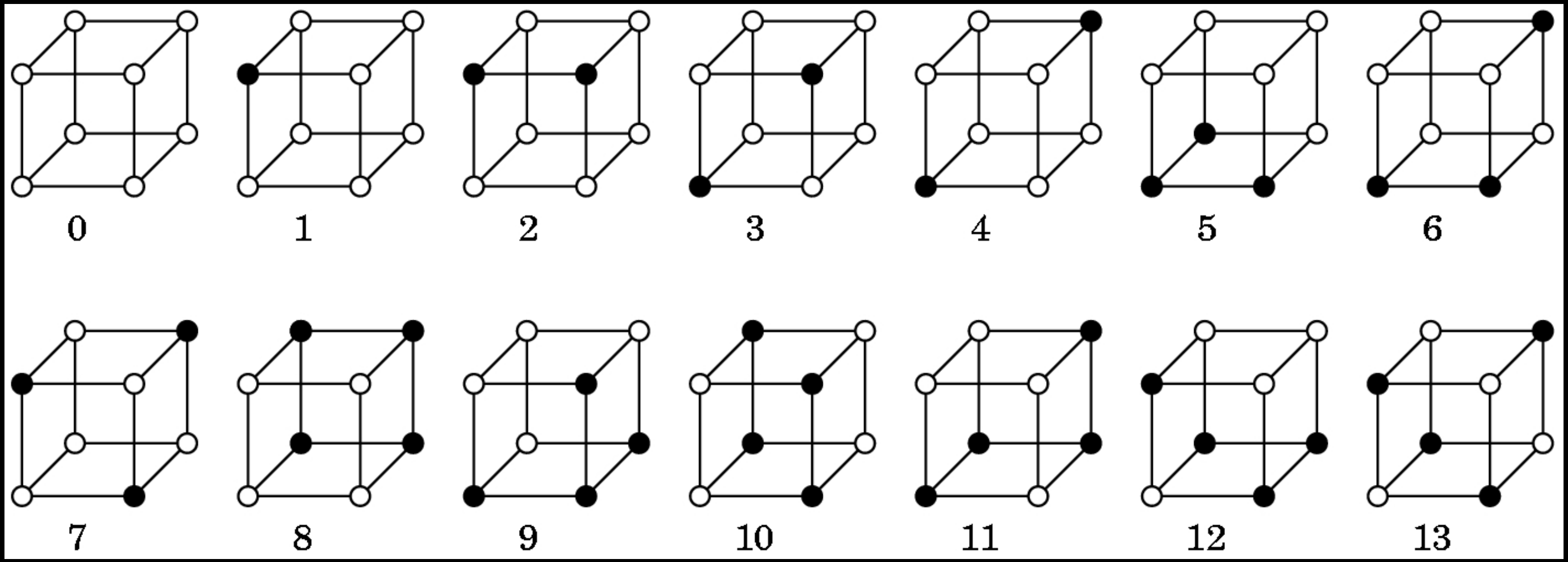


Isosurfaces

- $f(x,y,z)$ represents volumetric data set
- Two rendering methods
  - Isosurface rendering
  - Direct volume rendering (use all values [next])
- Isosurface given by $f(x,y,z) = c$
- Recall implicit surface $g(x, y, z)$:
  - $g(x, y, z) < 0$ inside
  - $g(x, y, z) = 0$ surface
  - $g(x, y, z) > 0$ outside
- Generalize right-hand side from 0 to c

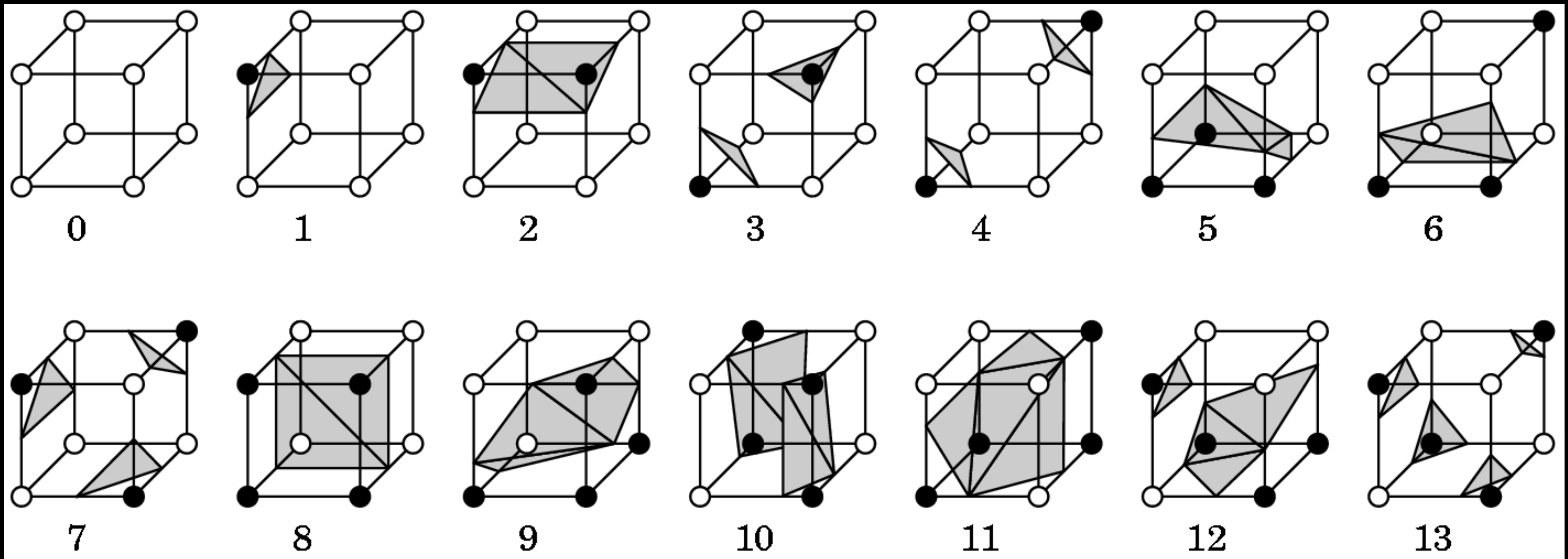
Marching Cubes

- Display technique for isosurfaces
- 3D version of marching squares
- 14 cube labelings (after elimination of symmetries)



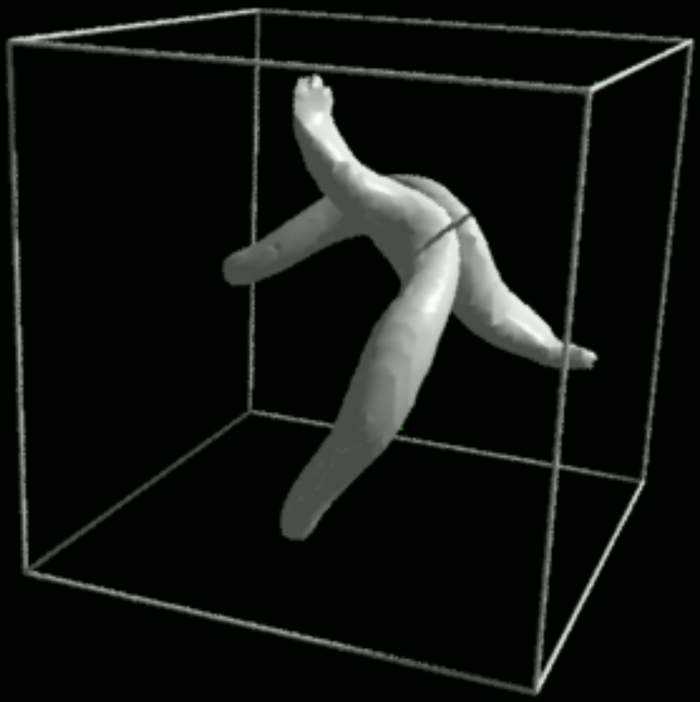
Marching Cube Tessellations

- Generalize marching squares, just more cases
- Interpolate as in 2D
- Ambiguities similar to 2D

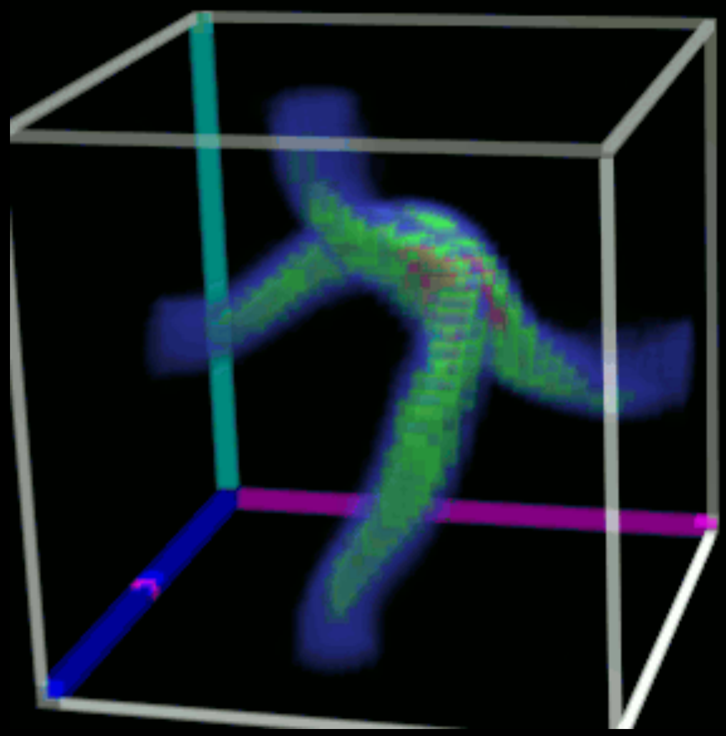


Volume Rendering

- Sometimes isosurfaces are unnatural or do not give sufficient information
- Use all voxels and transparency (α-values)



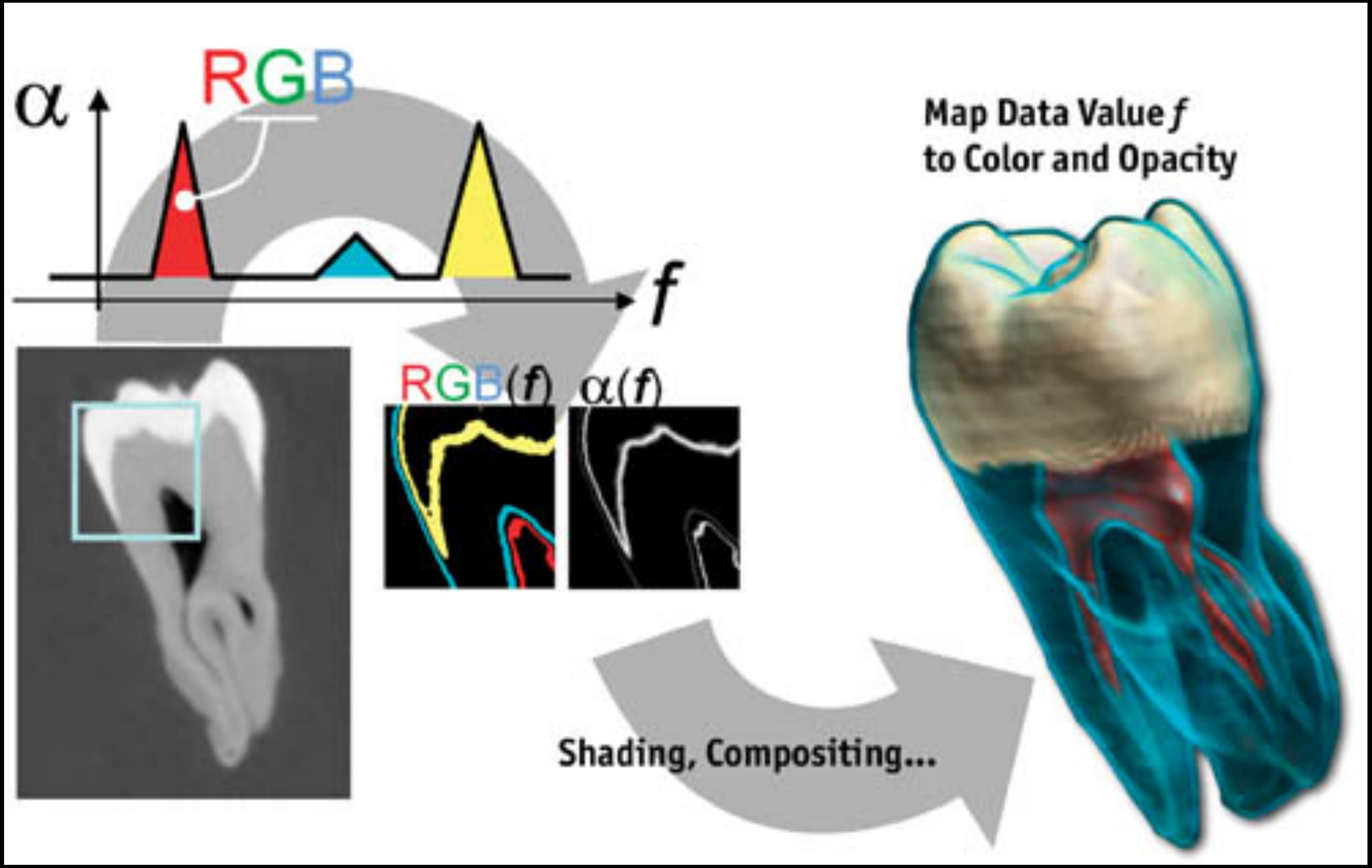
Ray-traced isosurface



Volume rendering

Volume Rendering Example

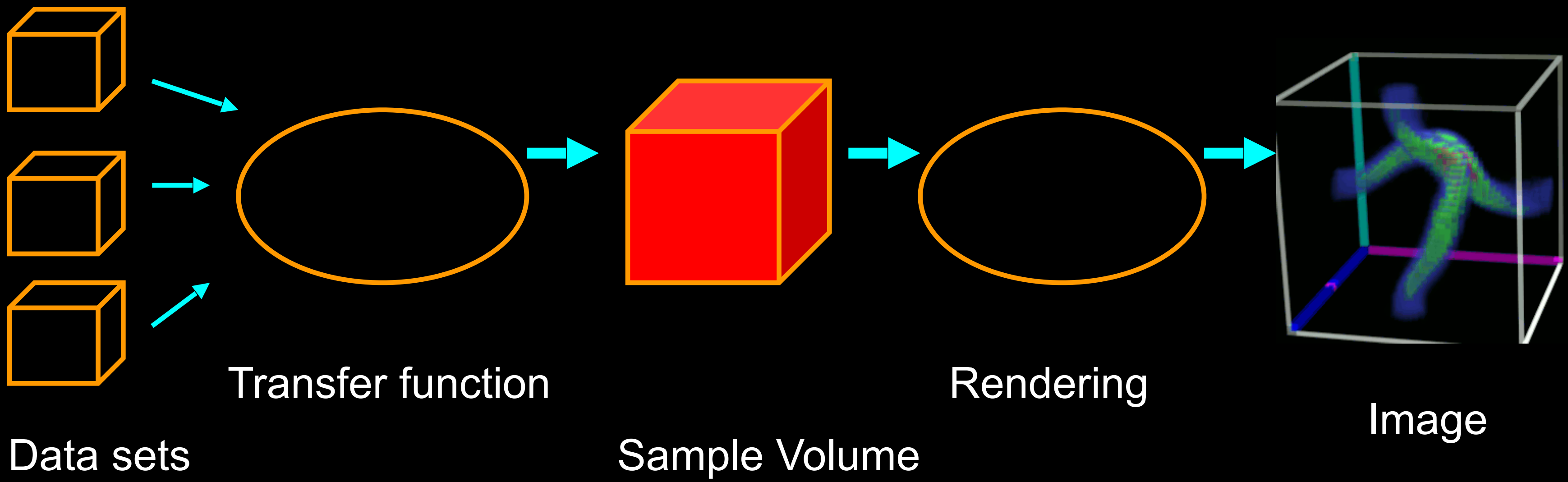
Rendering 3d tooth volume data



Source:
Nvidia

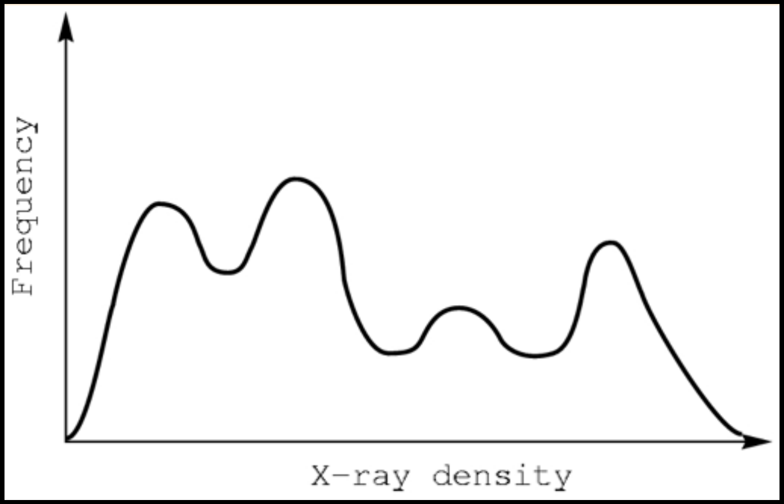
Volume Rendering Pipeline

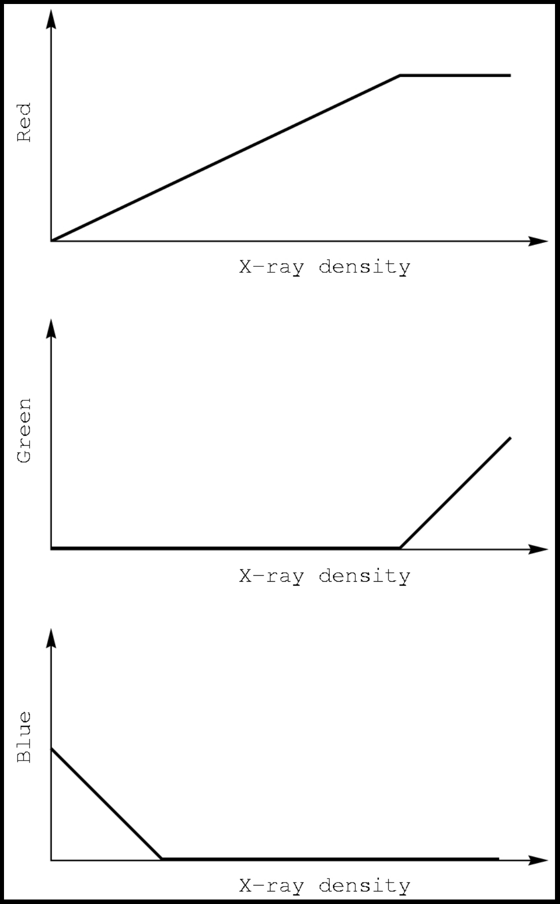
- Transfer function: converts input data set to colors and opacities
  - Example input: 256 x 256 x 256 x 8 bytes = 128 MB
  - Convert to 24 bit color, 8 bit opacity



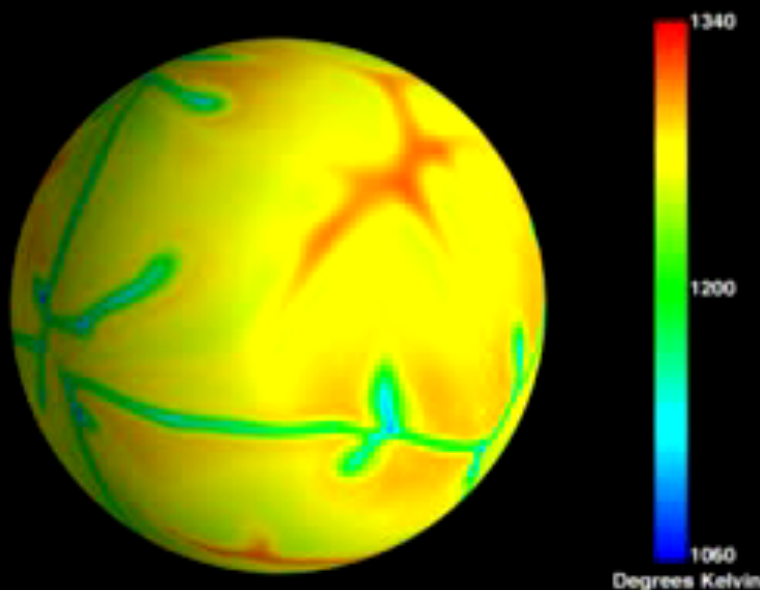
Transfer Functions

- Transform scalar data values to RGBA values
- Apply to every voxel in volume
- Highly application dependent
- Start from data histogram
- Opacity for emphasis

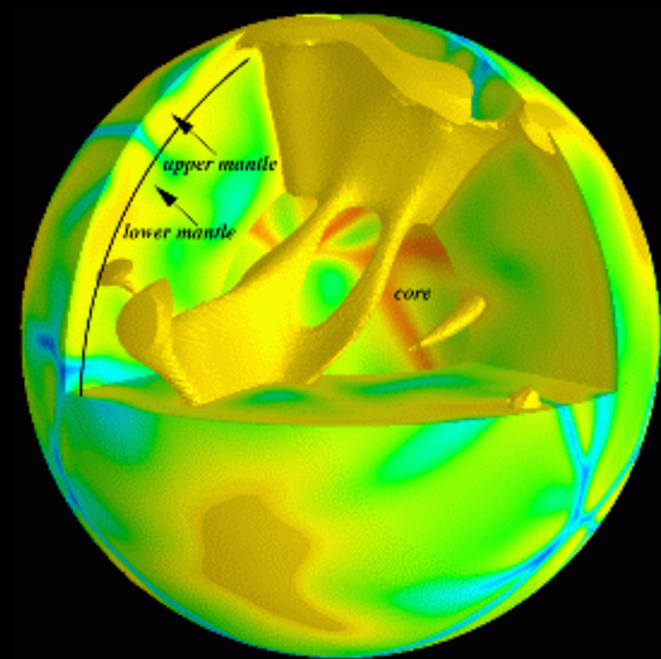




Transfer Function Example

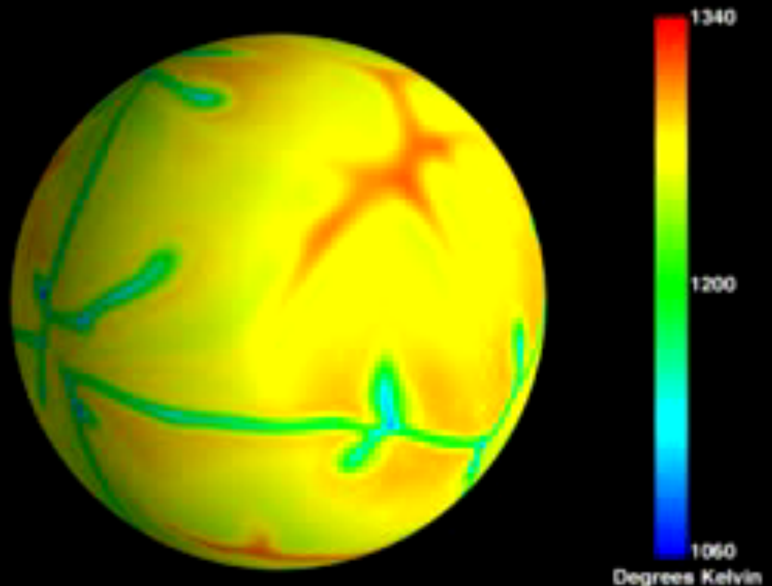


Mantle Heat Convection



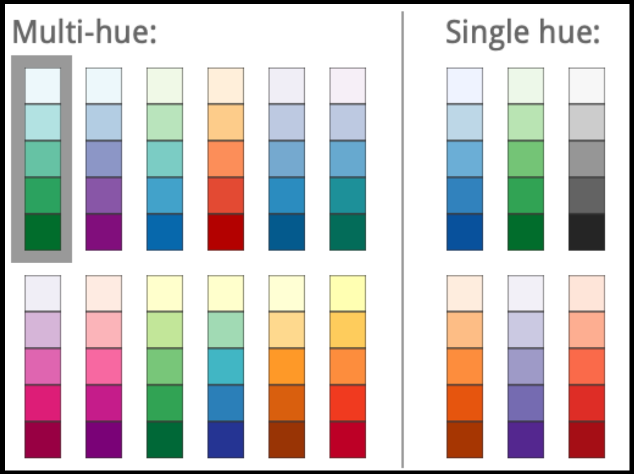
Scientific Computing and Imaging (SCI)
University of Utah

Side note on colormaps



You should not use the jet colormap

Modern, perceptual colormaps at colorbrewer.org (or in matplotlib)

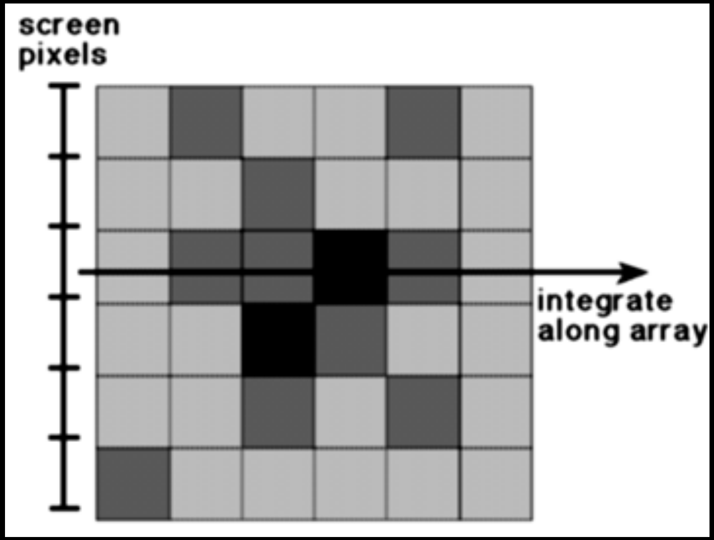


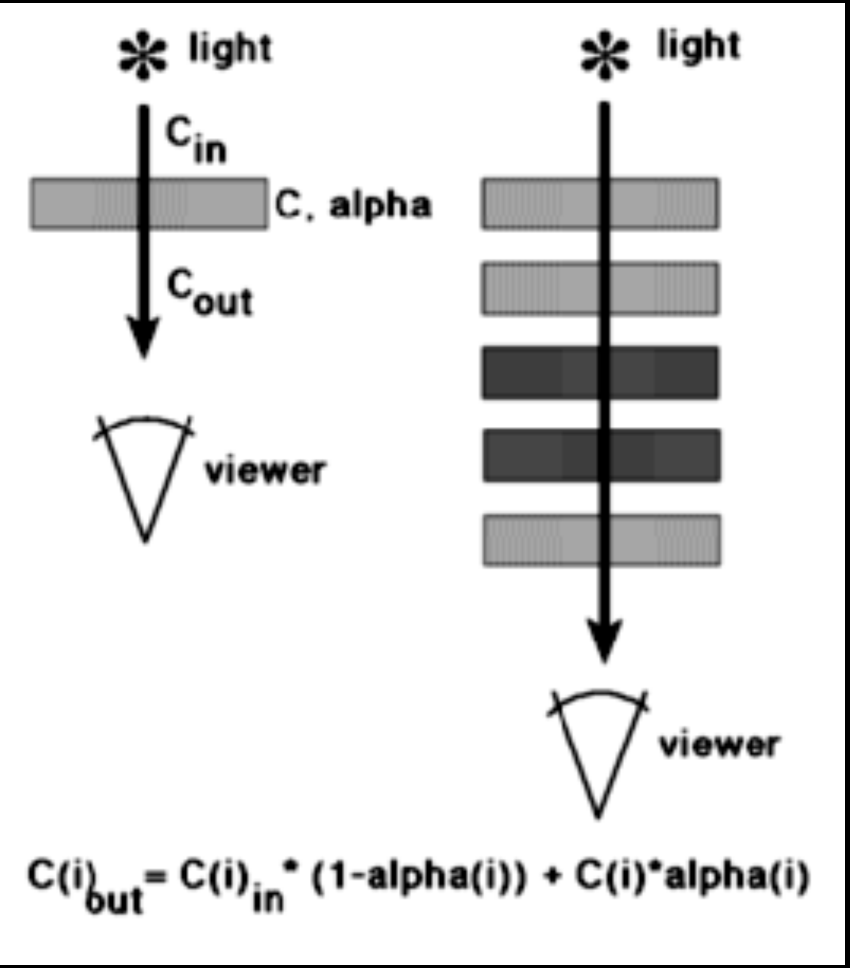
Volume Ray Casting

- Three volume rendering techniques
  - Volume ray casting
  - Splatting
  - 3D texture mapping
- Ray Casting
  - Integrate color through volume
  - Consider lighting (surfaces?)
  - Use regular x,y,z data grid when possible
  - Finite elements when necessary (e.g., ultrasound)
  - 3D-rasterize geometrical primitives

Accumulating Opacity

- α = 1.0 is opaque
- Composite multiple layers according to opacity
- Use local gradient of opacit for enhanced
  display of boundaries





Surface vs. Volume Rendering

- 3D model of surfaces
- Convert to triangles
- Draw primitives
- Lose or disguise data
- Good for opaque objects

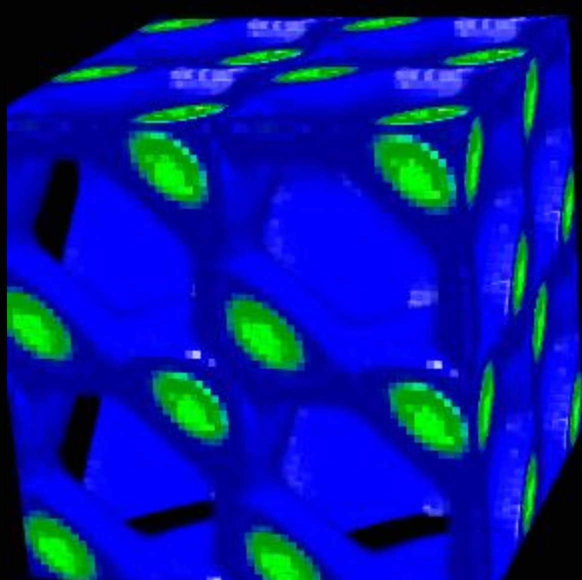
- Scalar field in 3D
- Convert it to RGBA values
- Render volume “directly”
- See data as given
- Good for complex objects

Sample Applications

- Medical
  - Computed Tomography (CT)
  - Magnetic Resonance Imaging (MRI)
  - Ultrasound
- Engineering and Science
  - Computational Fluid Dynamic (CFD)
  - Aerodynamic simulations
  - Meteorology
  - Astrophysics

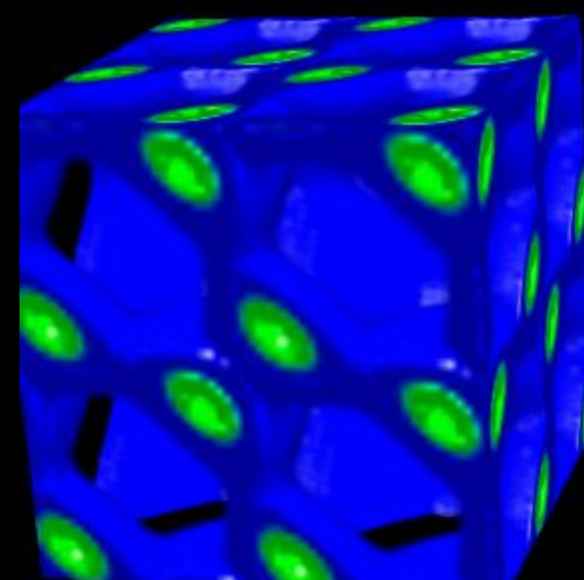
Trilinear Interpolation

- Interpolate to compute RGBA away from grid
- Nearest neighbor yields blocky images
- Use trilinear interpolation
- 3D generalization of bilinear interpolation



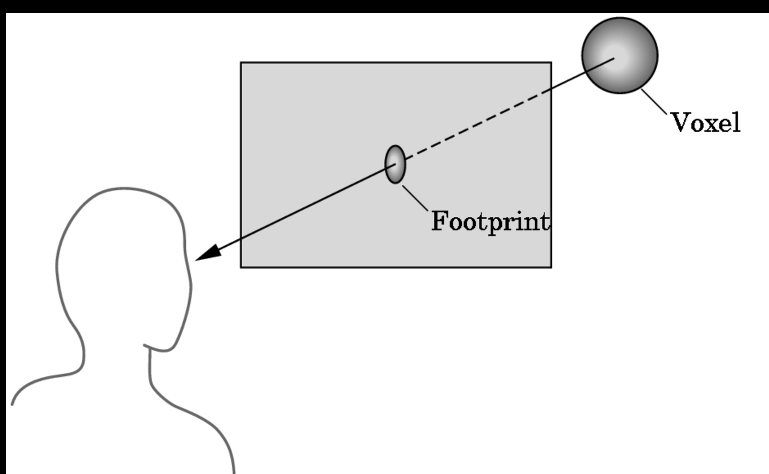
Nearest neighbor

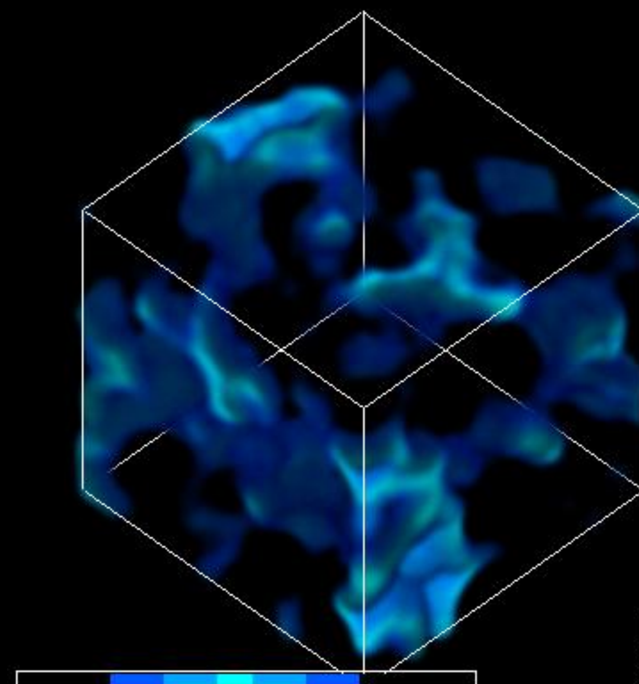
Trilinear interpolation



Splatting

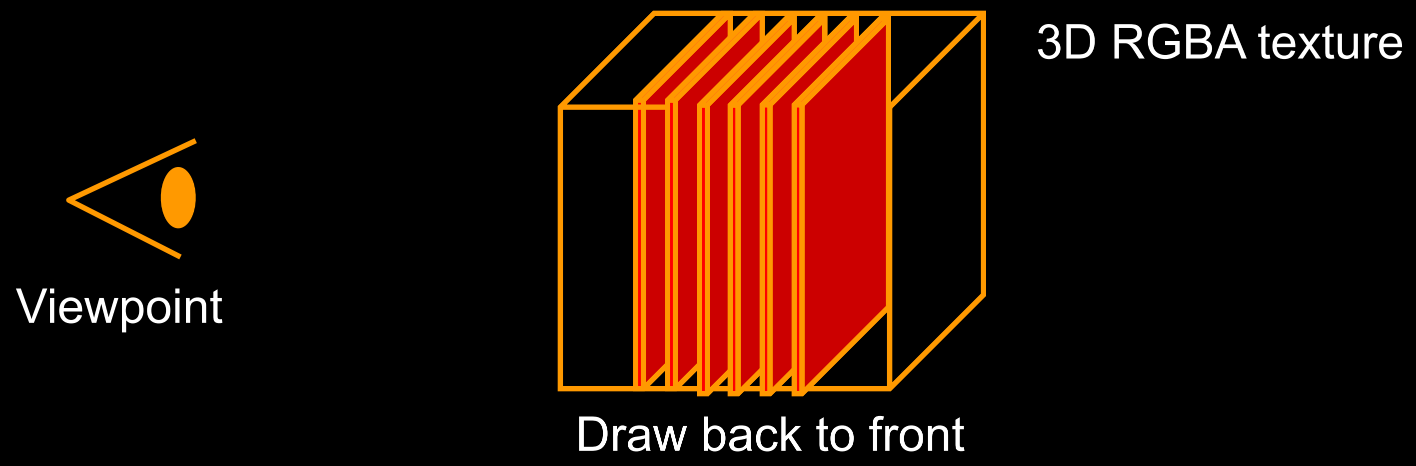
- Alternative to ray tracing
- Assign shape to each voxel (e.g., Gaussian)
- Project onto image plane (splat)
- Draw voxels back-to-front
- Composite (α-blend)



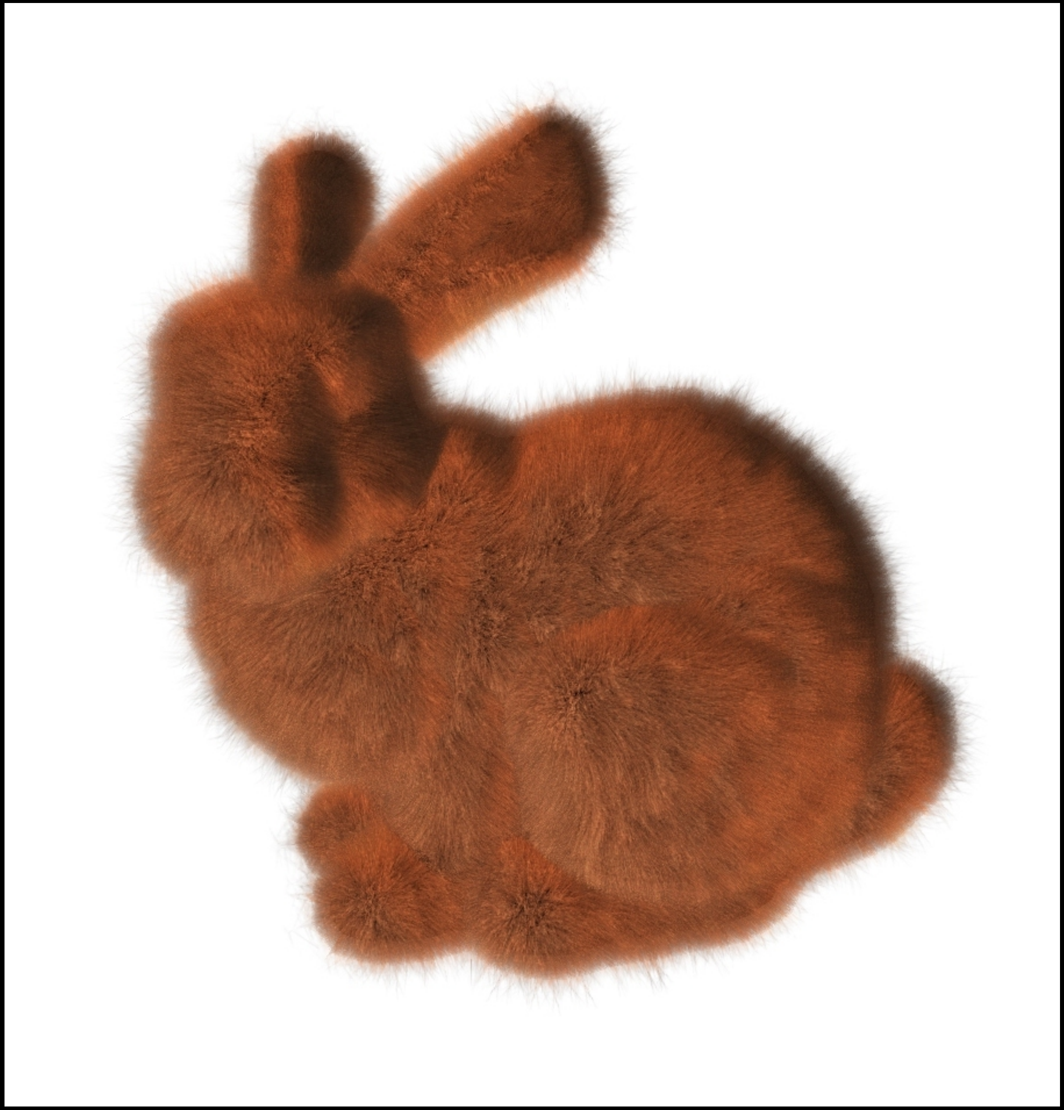


3D Textures

- Alternative to ray tracing, splatting
- Build a 3D texture (including opacity)
- Draw a stack of polygons, back-to-front
- Efficient if supported in graphics hardware
- Few polygons, much texture memory



Example: 3D Textures

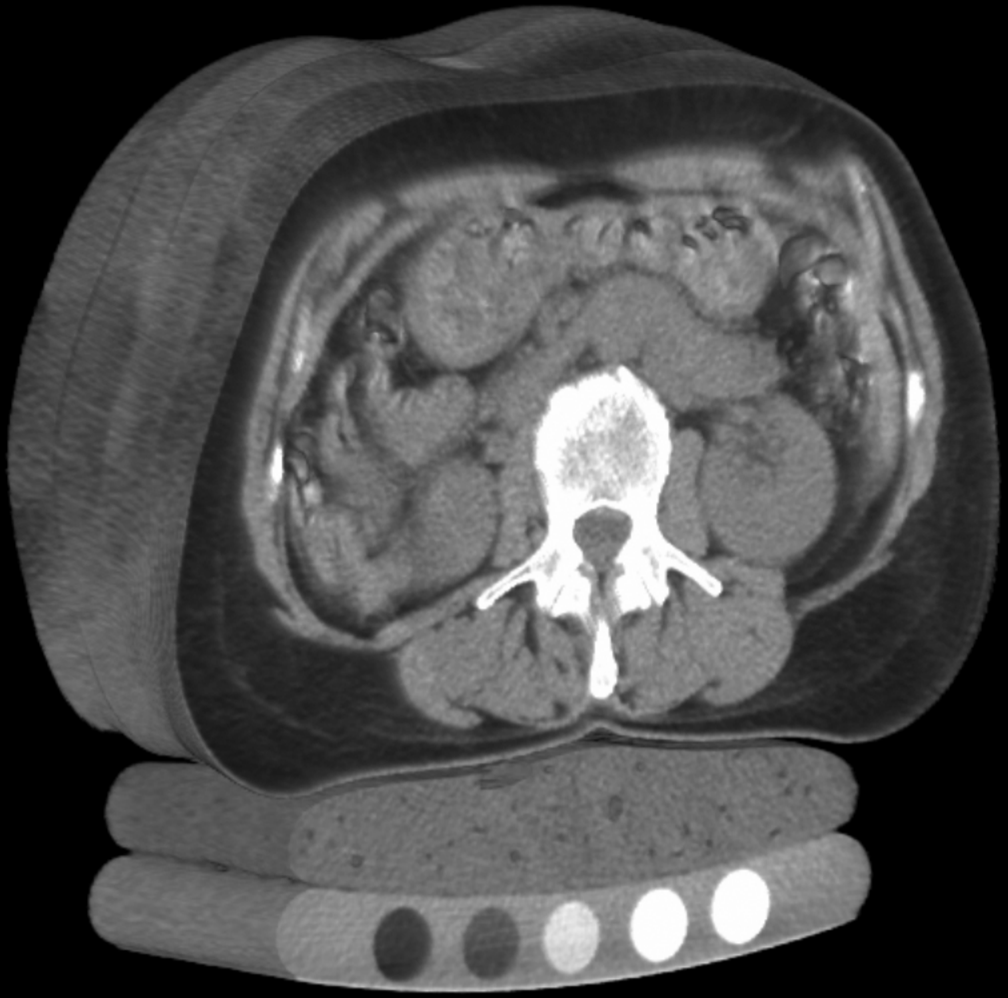


Emil Praun'01

Other Techniques

- Use CSG for cut-aways

Human torso



Source: Wikipedia

Acceleration of Volume Rendering

- Basic problem: Huge data sets
- Must program for locality (cache)
- Divide into multiple blocks if necessary
  - Example: marching cubes
- Use error measures to stop iteration
- Exploit parallelism

Outline

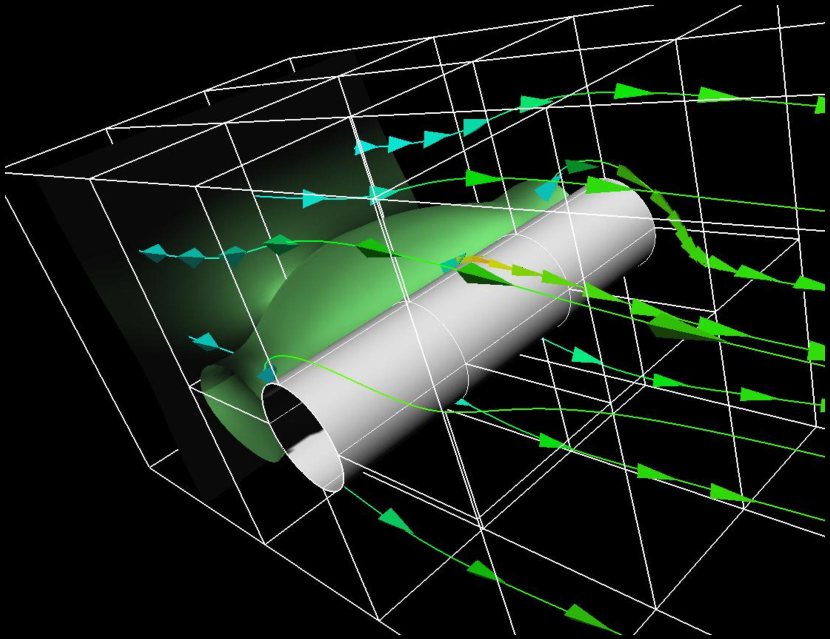
- Height Fields and Contours
- Scalar Fields
- Volume Rendering
- Vector Fields

Vector Fields

- Visualize vector at each (x,y,z) point
  - Example: velocity field
  - Example: hair
- Hedgehogs
  - Use 3D directed line segments (sample field)
  - Orientation and magnitude determined by vector
- Animation
  - Use for still image
  - Particle systems

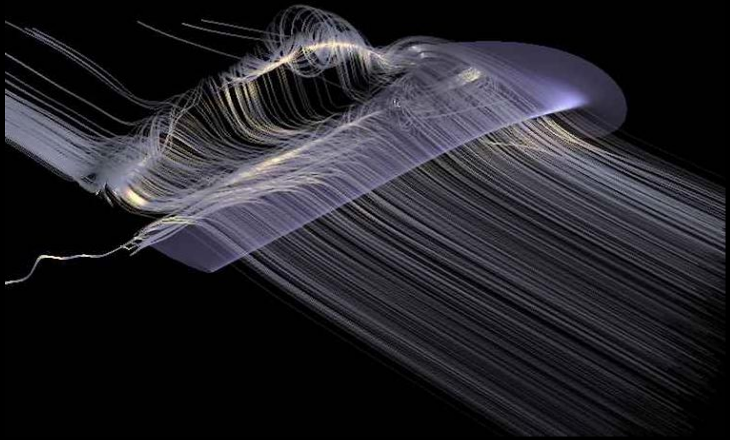
Blood flow in
human carotid artery

Using Glyphs and Streaklines

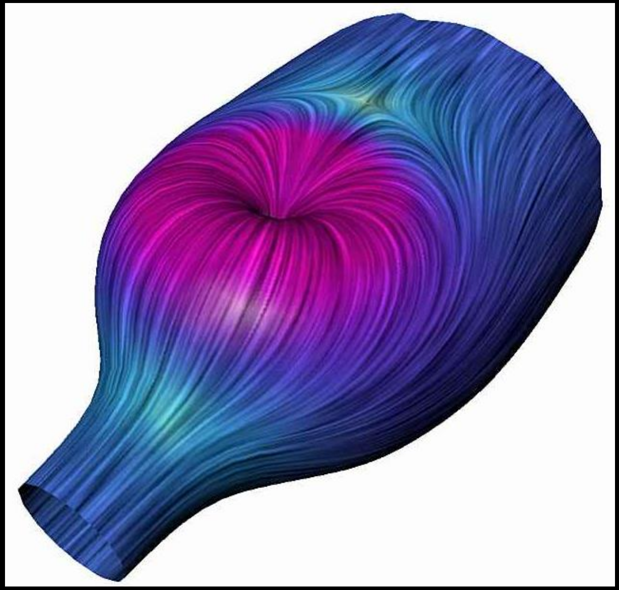


Glyphs for air flow

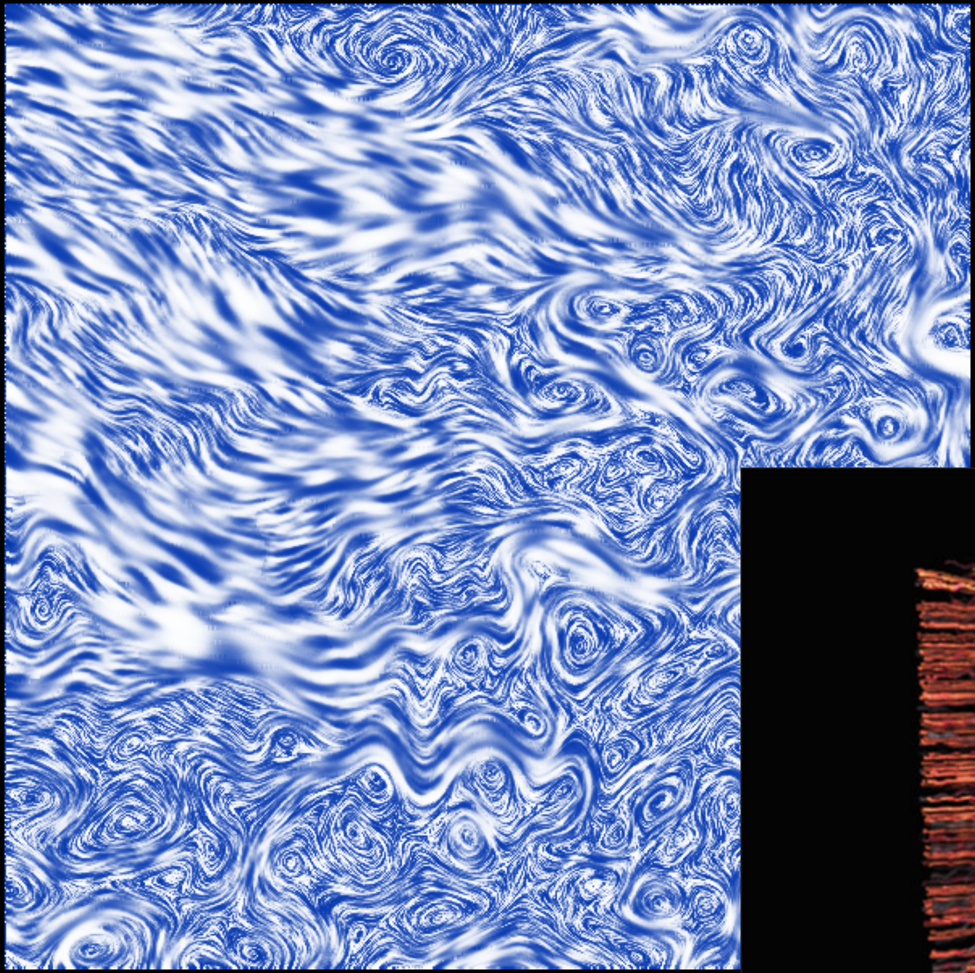
University of Utah



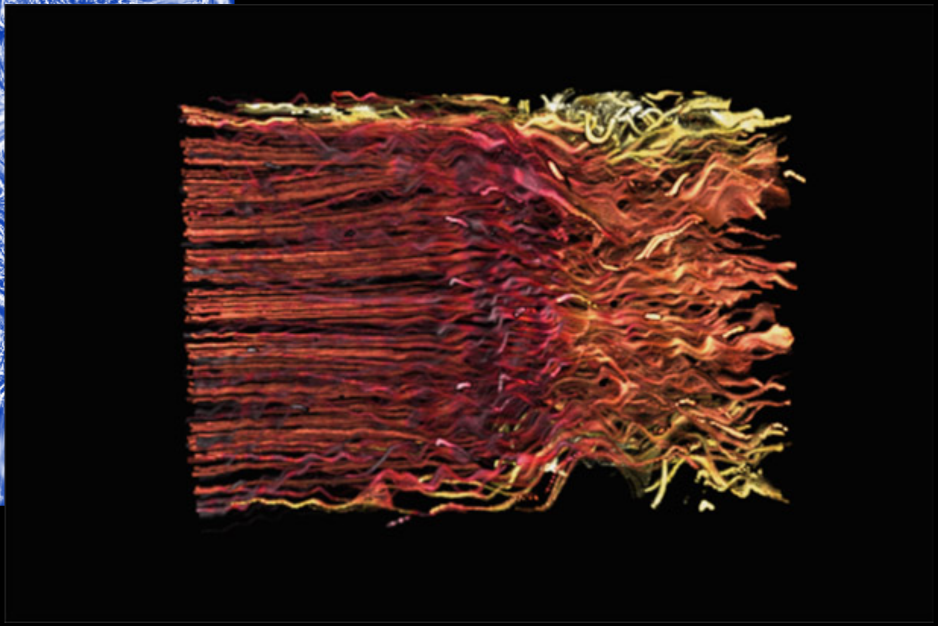
Glyph = marker (for example, an arrow) used for data visualization



More Flow Examples



Banks and Interrante



Summary

- Height Fields and Contours
- Scalar Fields
  – Isosurfaces
  – Marching cubes
- Volume Rendering
  – Volume ray tracing
  – Splatting
  – 3D Textures
- Vector Fields
  – Hedgehogs
  – Animated and interactive visualization