Other Fundamental Visualization Algorithms

Visualization Algorithms

- "Algorithms that transform data are the heart of visualization"
- Algorithms classified according to **structure** and **type** of data
- **Geometric transformations** change geometry but not topology
- Examples: translation, rotation, scaling
- **Topological transformations** change topology but not geometry
- Example: convert from regular to irregular grid

Visualization Algorithms

- **Attribute transformations** convert or create attributes in data
- Example: convert vector to scalar
- **Combined transformations** change data structure and attributes
- Algorithms that change data type include **scalar algorithms, vector algorithms, tensor algorithms,** and **modeling algorithms**
- **Volume visualization** and **vector visualization** have their own special algorithms

Contouring

- **Isocontour** and **isosurface extraction** can reveal structure of data (e.g., isobars on weather maps)
- Separate data into regions
- Isocontours: connected line segments
- Isosurfaces: triangular meshes

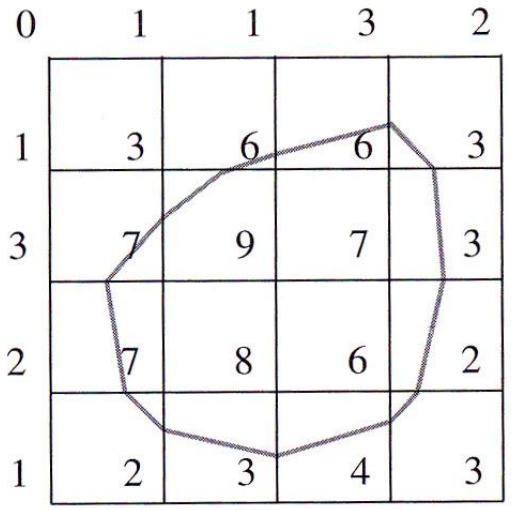


Figure 6–4 Contouring a 2D structured grid with contour line value = 5.

Contouring

- Isolines cross cell boundaries
- Use **interpolation** to compute crossing point
- **Marching squares** algorithm processes each quadrilateral cell independently
- Each vertex may be inside or outside (or on) contour
- How many cases must we consider?
- Ambiguous cases

Marching Squares Cases

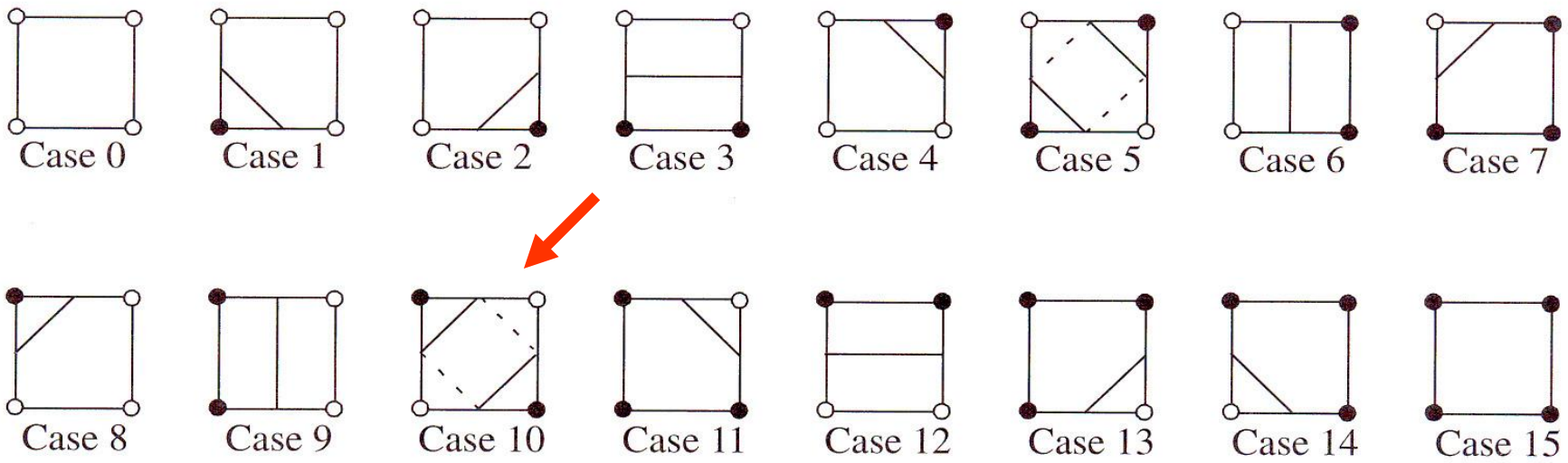
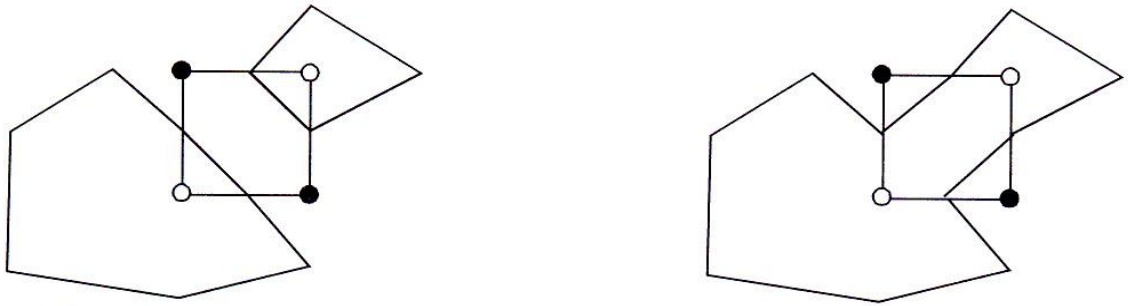


Figure 6–5 Sixteen different marching squares cases. Dark vertices indicate scalar value is above contour value. Cases 5 and 10 are ambiguous.

Marching Squares Ambiguous Case



(a) Break contour (b) Join contour

Figure 6–8 Choosing a particular contour case will break (a) or join (b) the current contour. Case shown is marching squares case 10.

Marching Cubes

- **Marching cubes** algorithm extracts isosurfaces from 3D rasters
- Very famous algorithm
- How many cases of hexahedral cells must we consider?
- Each of 8 vertices may be inside or outside
- $2^8 = 256$
- Lots of symmetry → really only 15 cases to consider

Marching Cubes Cases

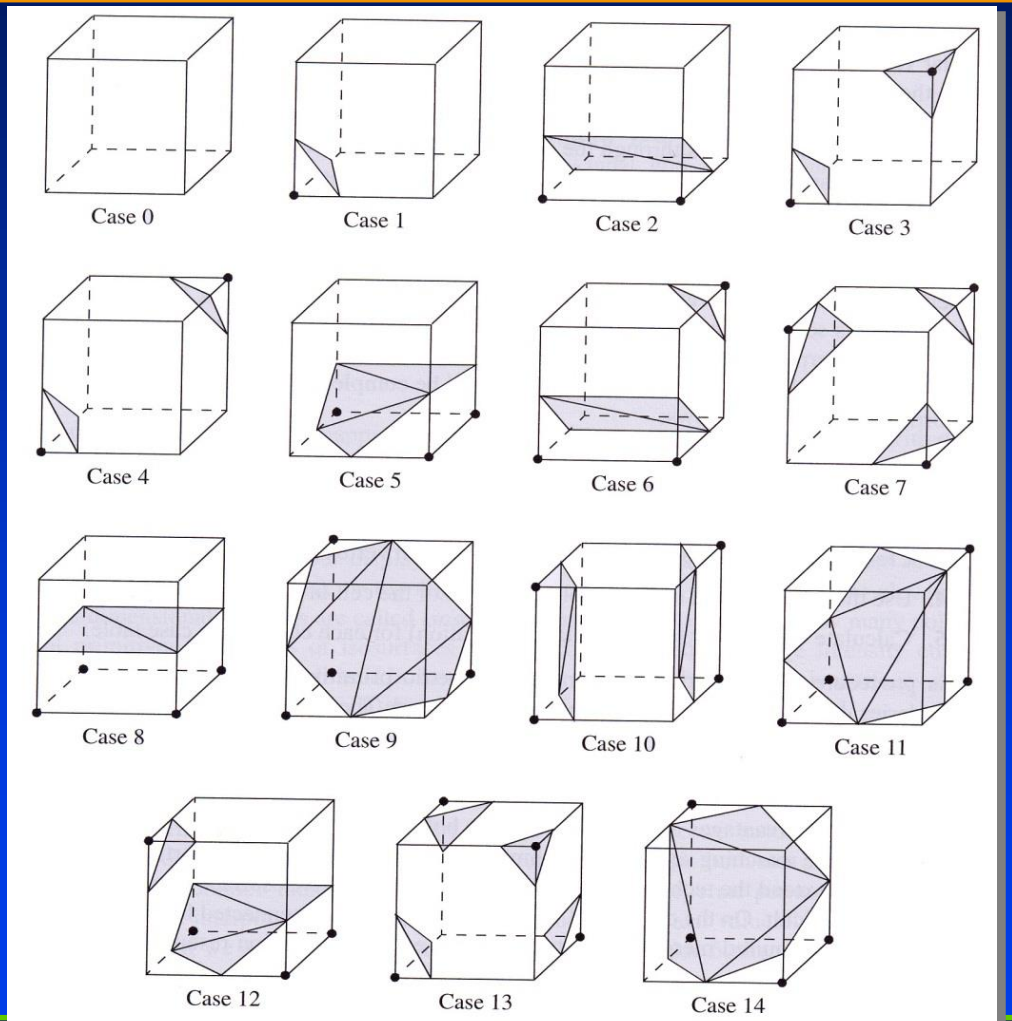
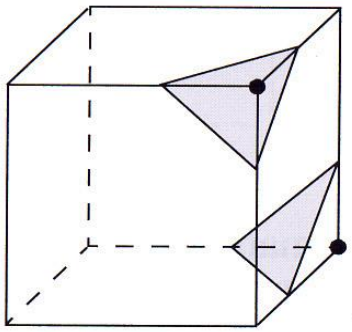
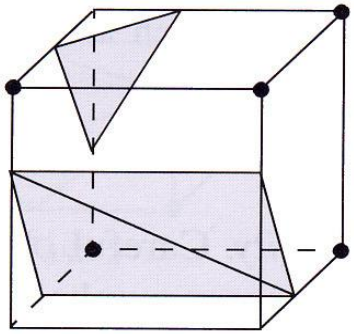


Figure 6–6 Marching cubes cases for 3D isosurface generation. The 256 possible cases have been reduced to 15 cases using symmetry. Dark vertices are greater than the selected isosurface value.

Marching Cubes Ambiguous Cases



Case 3



Case 6c

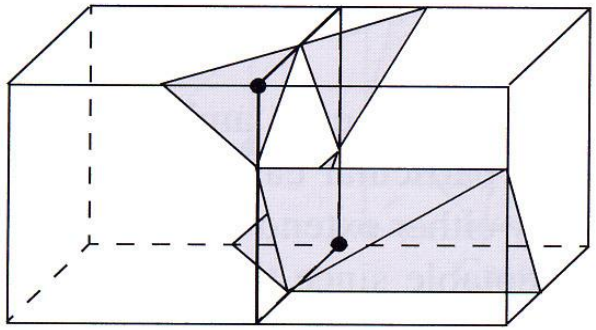
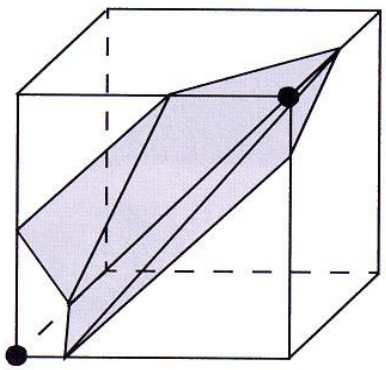
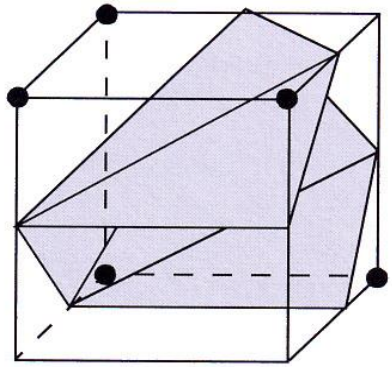


Figure 6–9 Arbitrarily choosing marching cubes cases leads to holes in the isosurface.

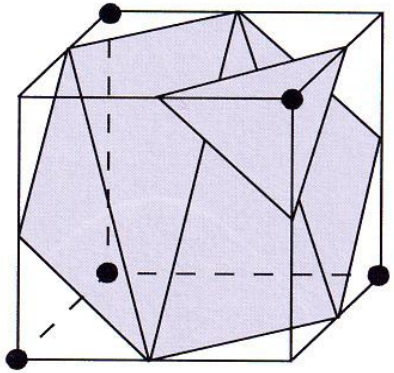
Marching Cubes Complementary Cases Used to Avoid Holes



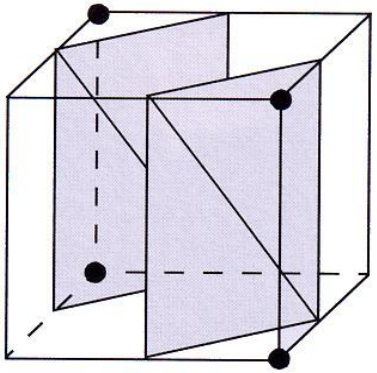
Case 3c



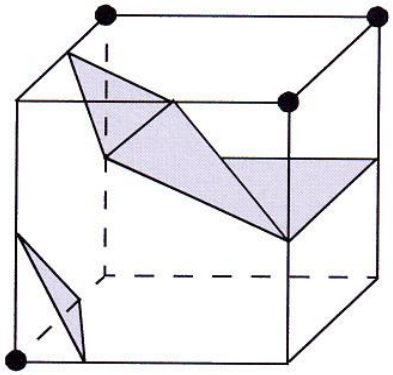
Case 6c



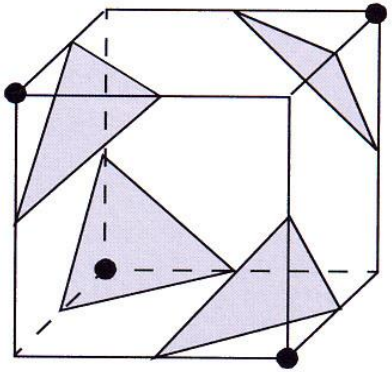
Case 7c



Case 10c



Case 12c



Case 13c

Figure 6–10 Marching cubes complementary cases.

Marching Triangles & Tetrahedra

- Can extend marching squares to *marching triangles*, and marching cubes to *marching tetrahedra*
- Divide squares into triangles, cubes into tetrahedra (how?) and then run different algorithms
- Tradeoff for both algorithms: simplicity vs. memory usage

Contouring Examples

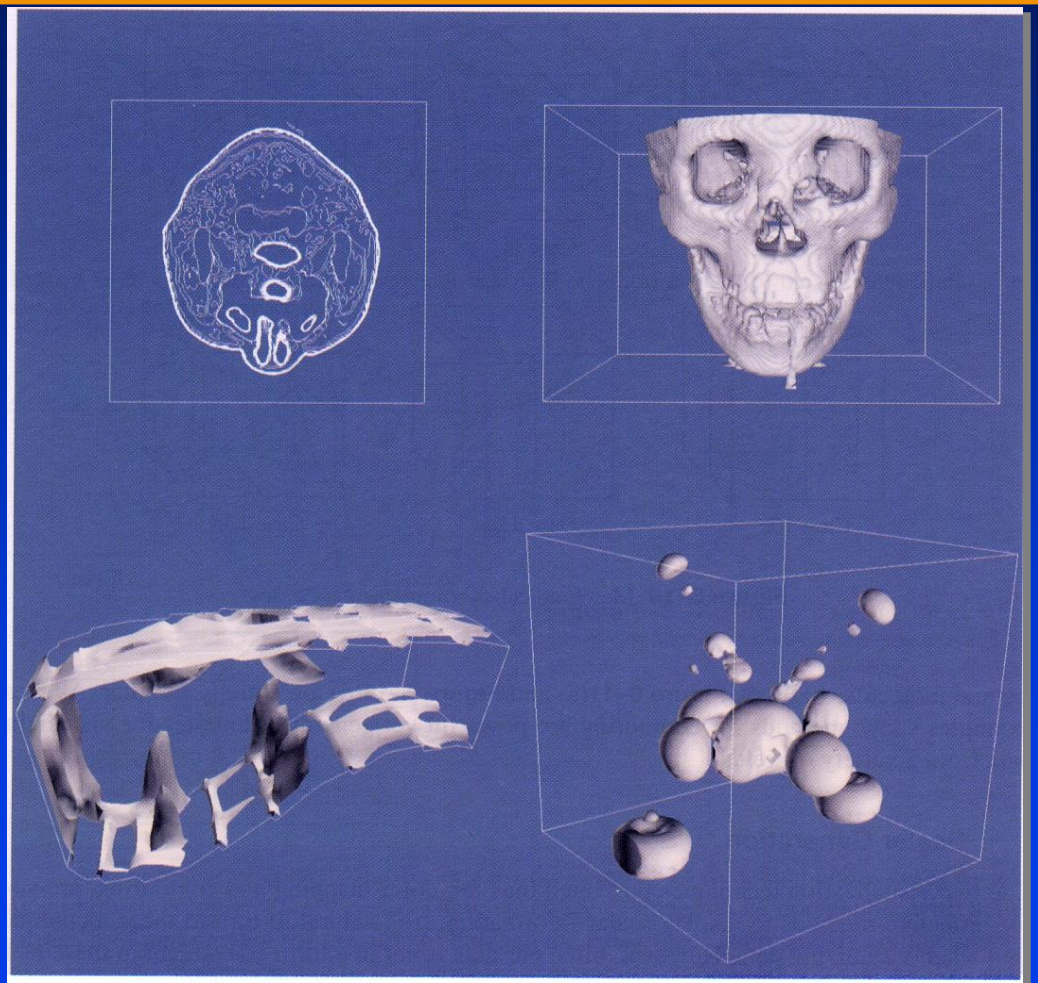
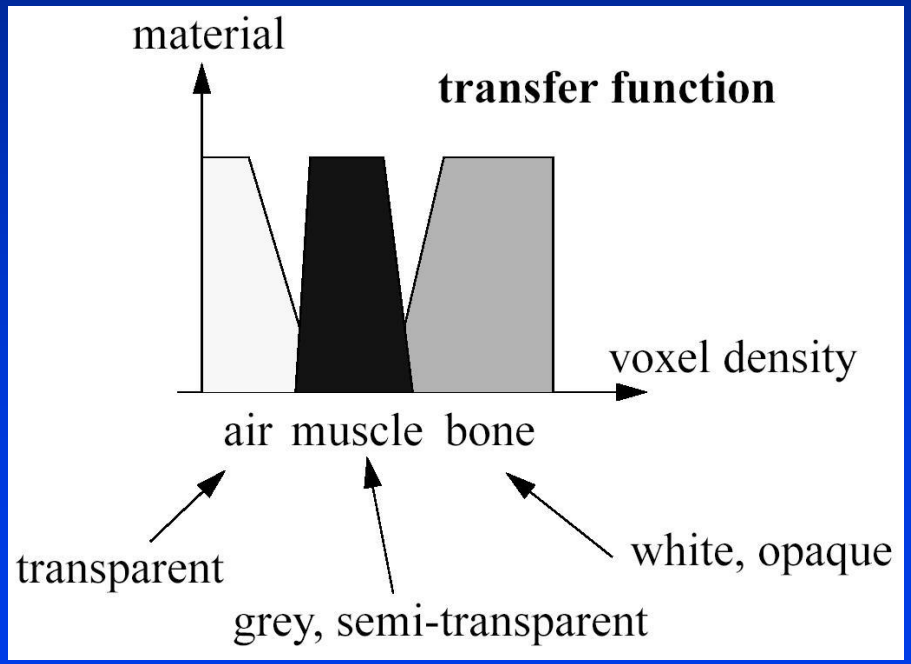


Figure 6–11 Contouring examples. (a) Marching squares used to generate contour lines (`headSlic.tcl`); (b) Marching cubes surface of human bone (`headBone.tcl`); (c) Marching cubes surface of flow density (`combIso.tcl`); (d) Marching cubes surface of iron-protein (`ironPIso.tcl`).

Transfer Functions

- The assignment of color and transparency to density is also called **classification**



Scalar Algorithms

- **Color mapping** – map scalar data to colors
- Why scalars?
- How would you map a vector to a color?
- **Color lookup table (LUT)** – attributes inside particular range are mapped to color

$$s_i < \min\ ,\ i = 0$$

$$s_i > \max\ ,\ i = n - 1$$

$$i = n\left(\frac{s_i - \min}{\max - \min}\right)$$

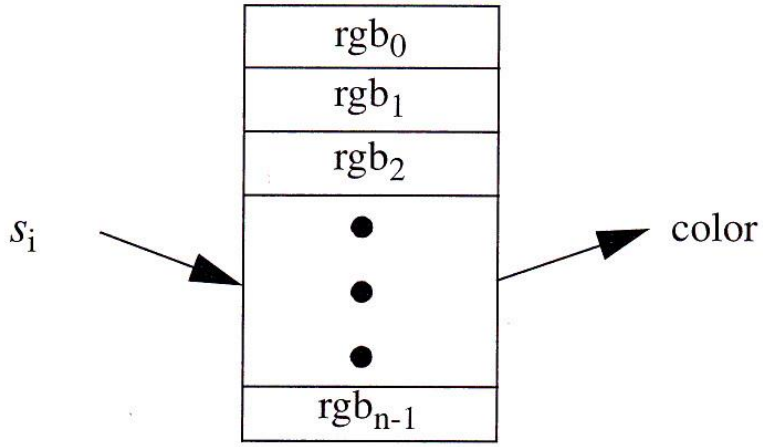


Figure 6–1 Mapping scalars to colors via a lookup table.

Transfer Functions

- More general form of lookup table
- Can map data to color as well as transparency
- Usually expressed as actual functions

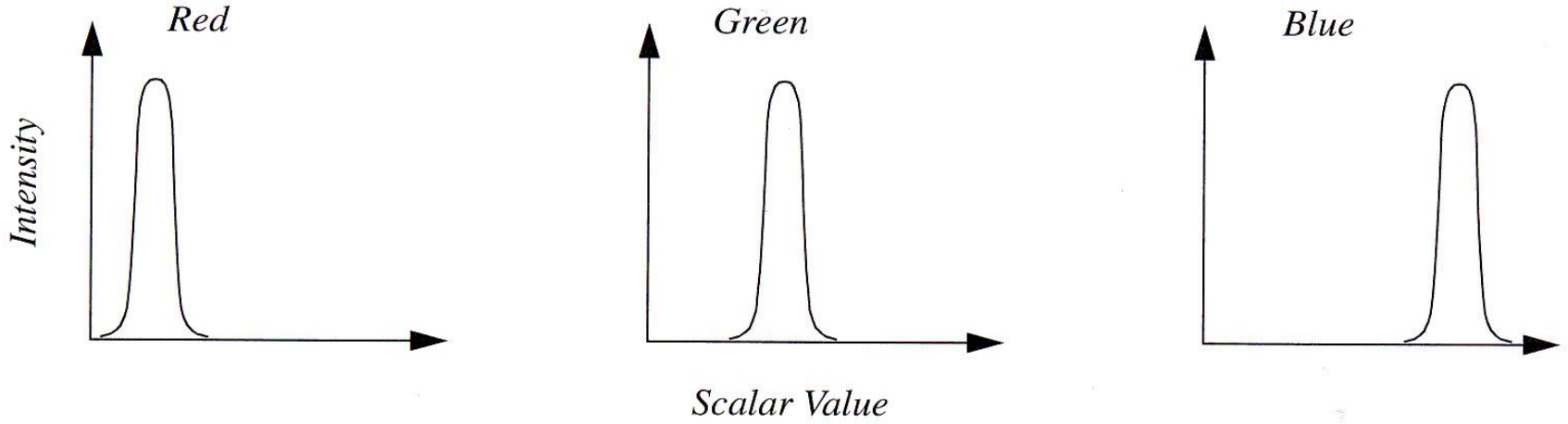
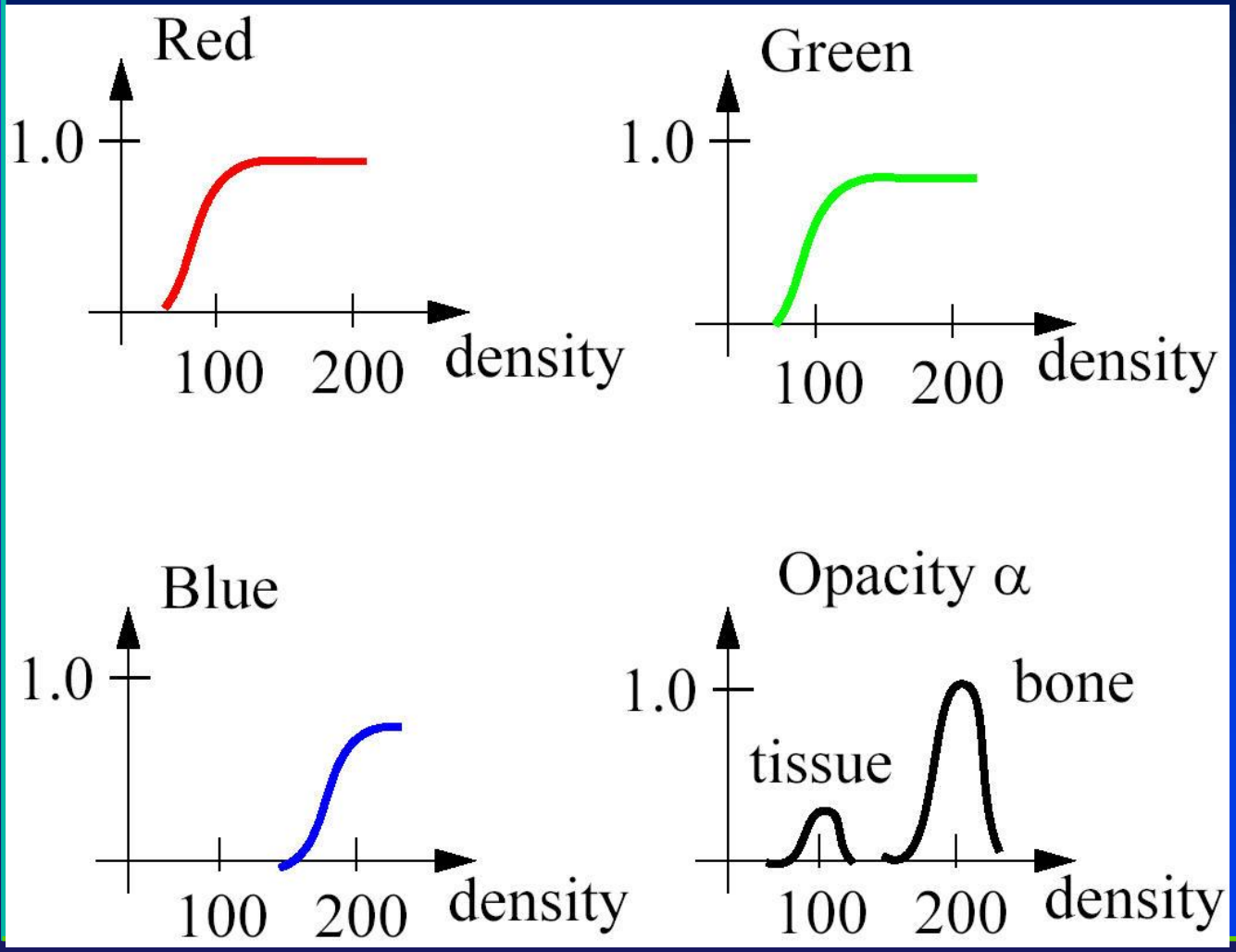
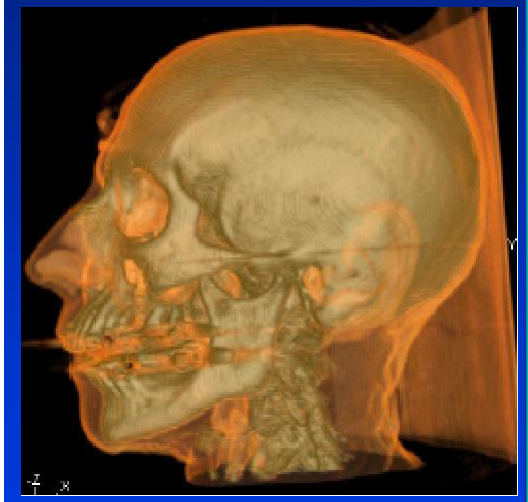


Figure 6–2 Transfer function for color components red, green, and blue as a function of scalar value.

Transfer Functions

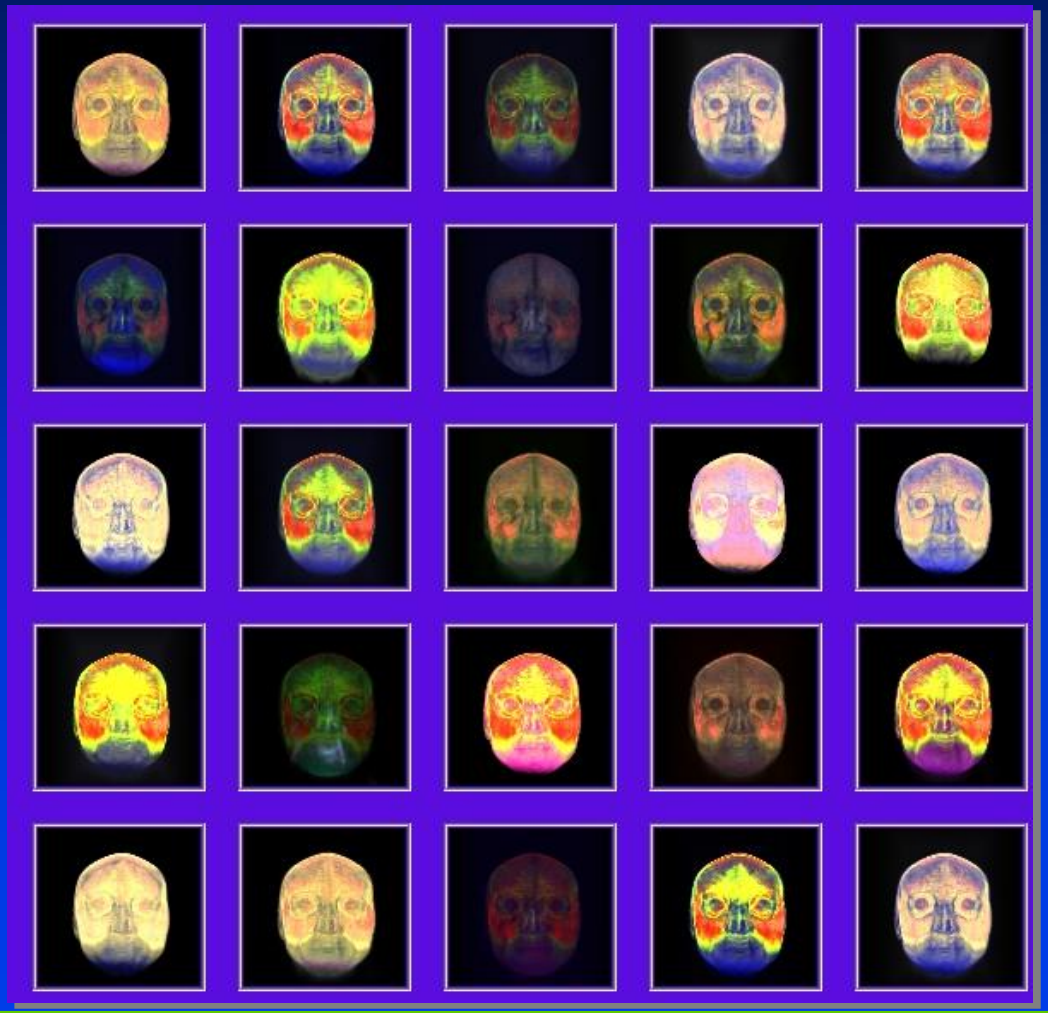




Transfer Functions

- Difficult to design
- Semi-automatic systems exist: **transfer function design galleries**
- Idea: generate random transfer functions, user selects ones he likes, system *mutates* them using a genetic algorithm to create new ones

Transfer Function Design Galleries



Transfer Functions

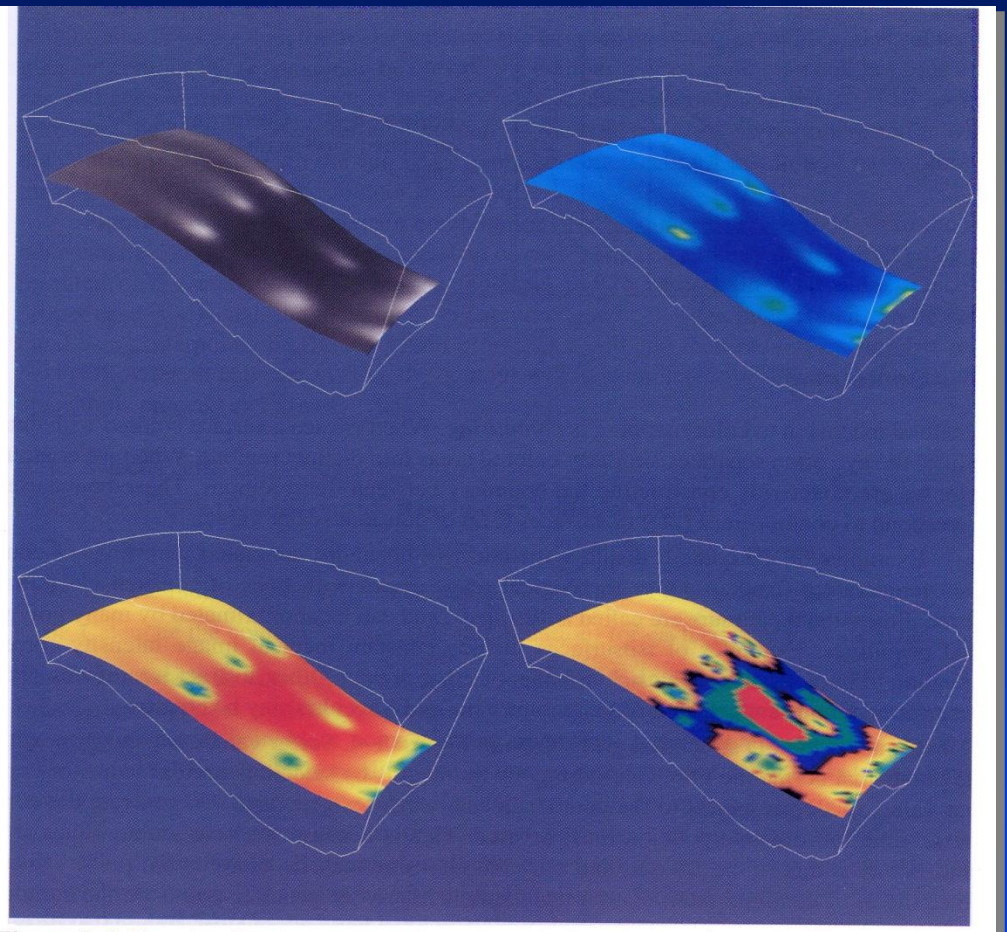


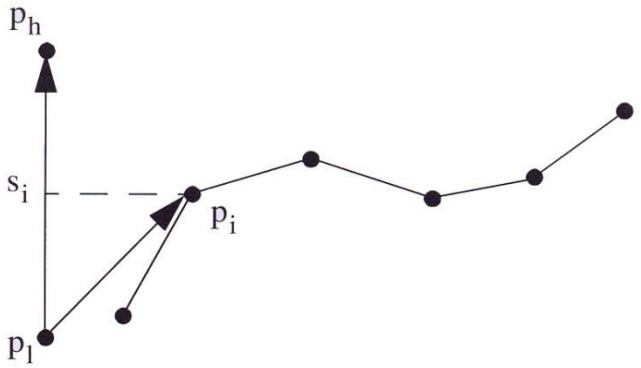
Figure 6–3 Flow density colored with different lookup tables. Top-left: grayscale; Top-right rainbow (blue to red); lower-left rainbow (red to blue); lower-right large contrast (`rainbow.tcl`).

Scalar Generation

- Vectors and other n-D quantities can be turned into scalars
- Example: taking magnitude of vector
- Example: Hawaii terrain visualization created by projecting vector onto vertical
- Normalize vectors to give maximum magnitude of 1.0
- Steepest slope mapped to brightest color

Scalar Generation

$$s_i = \frac{(p_i - p_l) \cdot (p_h - p_l)}{|p_h - p_l|^2}$$



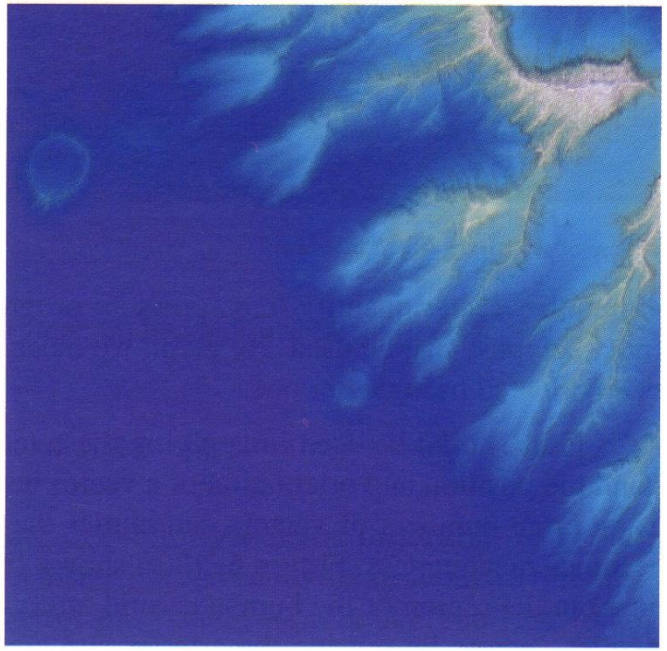


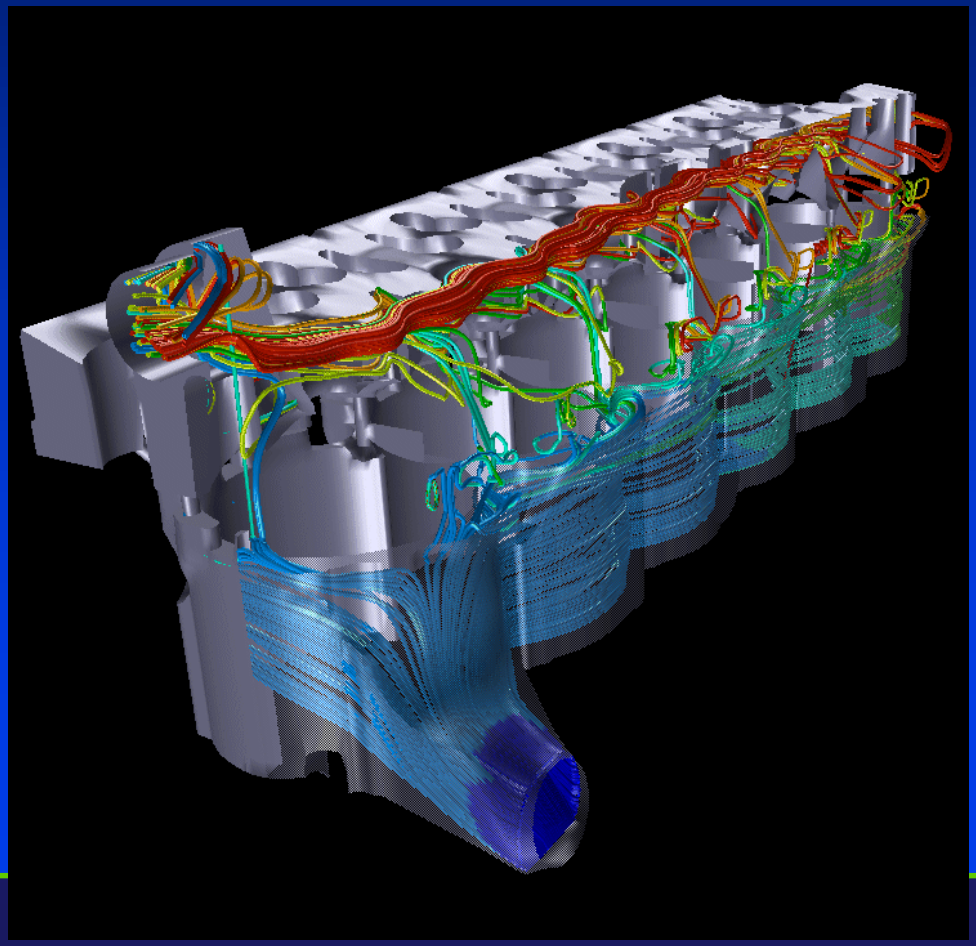
Figure 6–12 Computing scalars using normalized dot product. Bottom half of figure illustrates technique applied to terrain data from Honolulu, Hawaii (`hawaii.tcl`).

Vector Field Visualization

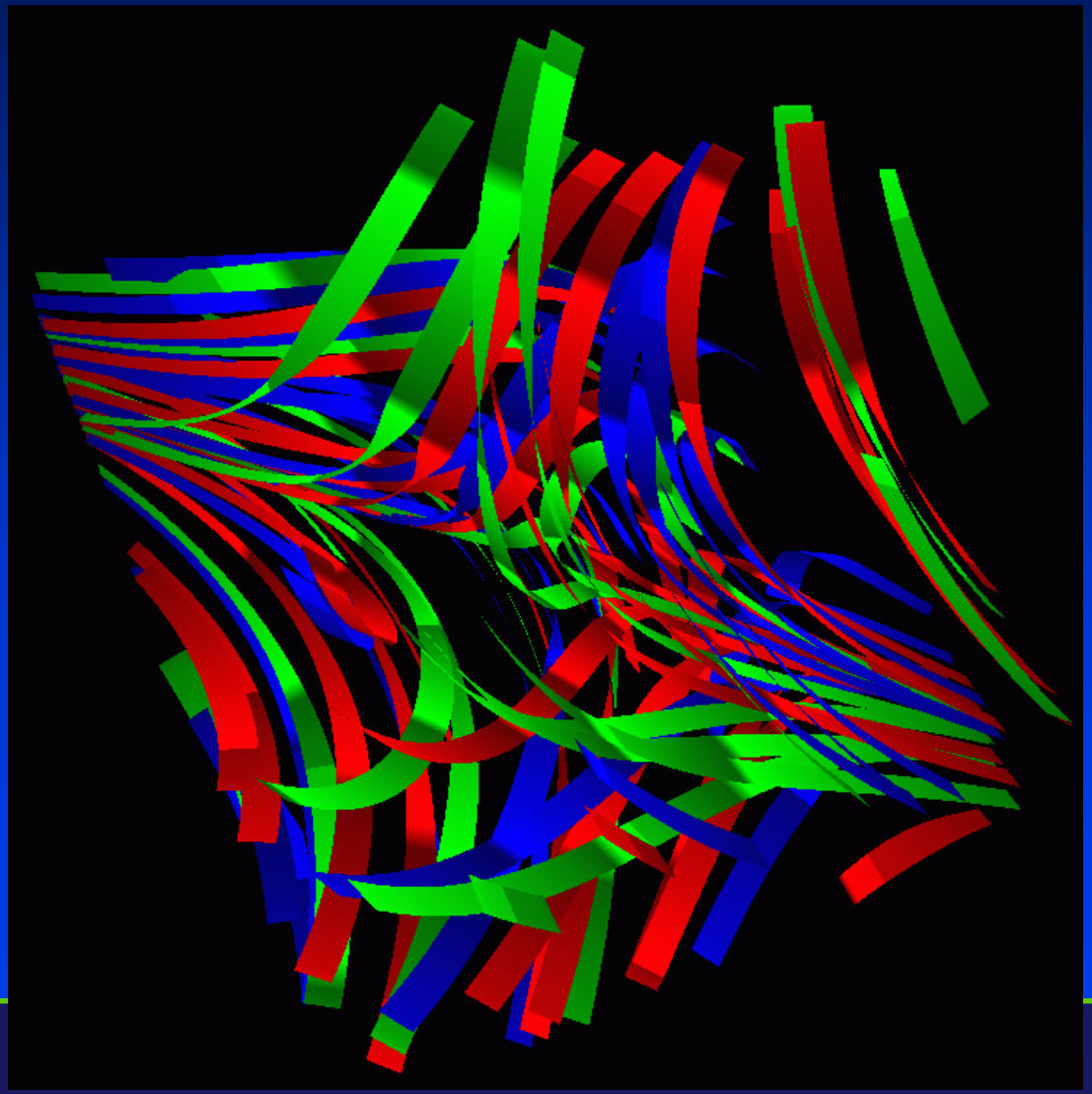
- Streamlines
  - Integration through vector field
- Stream ribbons
  - Connect two streamlines
- Streamtubes
  - Connect three or more streamlines
- Stream surfaces
  - Sweep line segment through vector field

Streamlines Example

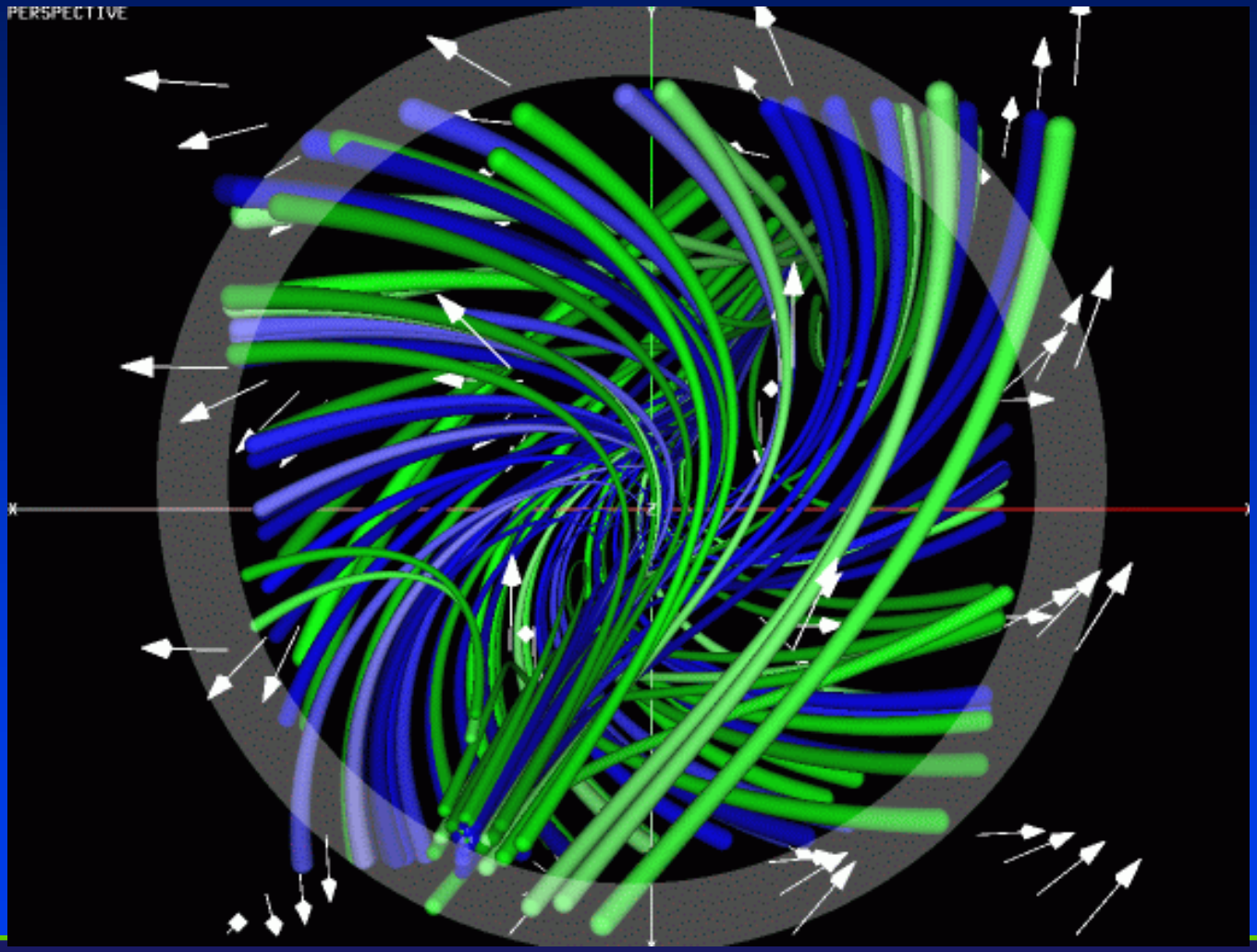
Color indicates temperature of air flowing through engine



Streamribbons Example



Streamtubes Example



Streamesurfaces Example