

## Fundamental Visualization Algorithms

1

## 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

2

## 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

3

## 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

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

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

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

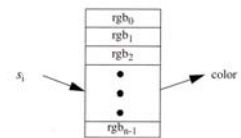


Figure 6-1 Mapping scalars to colors via a lookup table.

4

## Transfer Functions

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

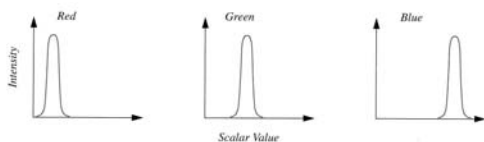
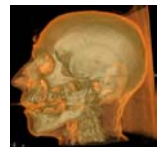
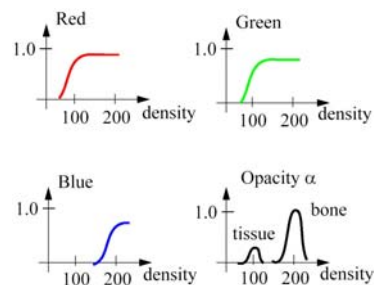


Figure 6-2 Transfer function for color components red, green, and blue as a function of scalar value.

5

## Transfer Functions



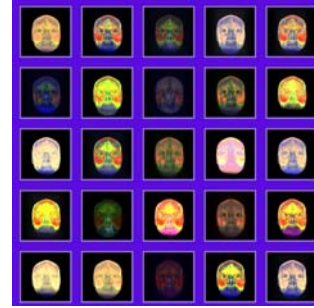
6

## 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

7

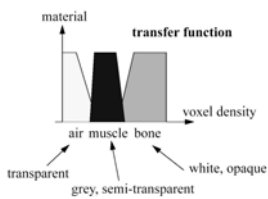
## Transfer Function Design Galleries



8

## Transfer Functions

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



9

## Transfer Functions

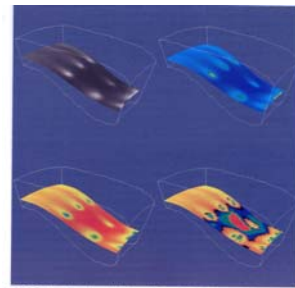


Figure 6-3 Three density colored with different lookup tables. Top-left: grayscale. Top-right: random (blue to red). Lower-left: random (red to blue). Lower-right: large contrast to a color (red to blue).

10

## 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

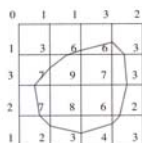


Figure 6-4 Contouring a 2D structured grid with contour line value = 5.

11

## 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

12

## Marching Squares Cases

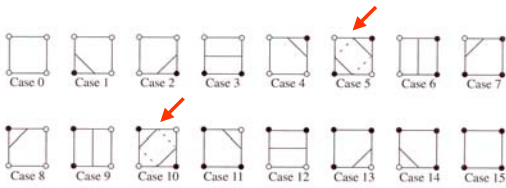


Figure 6-5 Sixteen different marching squares cases. Dark vertices indicate scalar value is above contour value. Cases 5 and 10 are ambiguous.

13

## Marching Squares Ambiguous Case



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

14

## 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  $\rightarrow$  really only 15 cases to consider

15

## Marching Cubes Cases

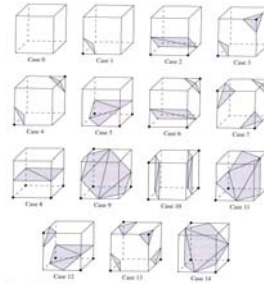


Figure 6-6 Marching cubes cases for 3D isosurface generation. The 256 possible cases have been reduced to 15 cases using symmetry. Dark vertices are greater than the selected isosurface value.

16

## Marching Cubes Ambiguous Cases

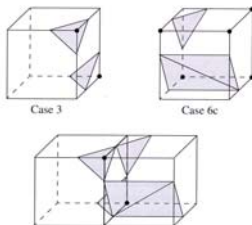


Figure 6-9 Arbitrarily choosing marching cubes cases leads to holes in the isosurface.

17

## Marching Cubes Complementary Cases Used to Avoid Holes

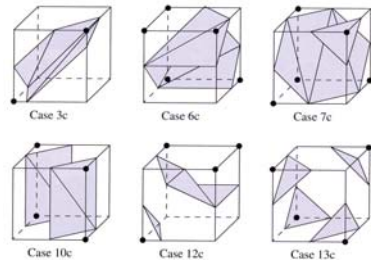


Figure 6-10 Marching cubes complementary cases.

18

## 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

19

## Contouring Examples

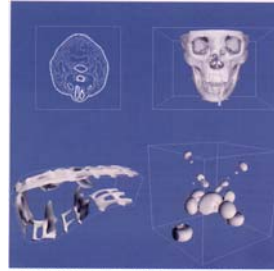


Figure 6-11 Contouring examples. (a) Marching squares used to generate contour lines. (b) (c) (d) Marching cubes surface of human bone (data). Bone. (c) (d) Marching cubes surface of flow density (contour lines). (e) (f) Marching cubes surface of non-point cloud (data). (g) (h)

20