Data Visualization (CSCI 627/490)

Volume Rendering

Dr. David Koop
Visualizing Volume (3D) Data

- **2D visualization**
  - slice images (or multi-planar reformatting MPR)
- **Indirect**
  - 3D visualization isosurfaces (or surface-shaded display SSD)
- **Direct**
  - 3D visualization (direct volume rendering DVR)
Generating Isolines (Isovalue = 5)

(a) Scalar grid. (b) The +/- grid. (c) Midpoint vertices. (d) Isocontour.

Figure 2.4.
Marching Squares

Figure 2.10. Red, positive regions and blue, negative regions for each square configuration. The green isocontour is part of the positive region. Black vertices are positive.

Proof of Properties 1 & 2:
The Marching Squares isocontour consists of a finite set of line segments, so it is piecewise linear. These line segments intersect only at their endpoints and thus form a triangulation of the isocontour. The endpoints of these line segments lie on the grid edges, confirming Property 2. □

Property 3.
The isocontour intersects every bipolar grid edge at exactly one point.

Property 4.
The isocontour does not intersect any negative or strictly positive grid edges.

Proof of Properties 3 & 4:
Each isocontour edge is contained in a grid square. Since the grid squares are convex, only isocontour edges with endpoints (vertices) on the grid edge intersect the grid edge. If the grid edge has one positive and one negative endpoint, the unique location of the isocontour vertex on the grid edge is determined by linear interpolation. Thus the isocontour intersects a bipolar grid edge at only one point.

If the grid edge is negative or strictly positive, then no isocontour vertex lies on the grid edge. Thus the isocontour does not intersect negative or strictly positive grid edges. □

Within each grid square the isocontour partitions the grid square into two regions. Let the positive region for a grid square \( c \) be the set of points which can be reached by a path \( \zeta \) from a positive vertex. More precisely, a point \( p \) is in the positive region of \( c \) if there is some path \( \zeta \subset c \) connecting \( p \) to a positive vertex of \( c \) such that the interior of \( \zeta \) does not intersect the isocontour. A point \( p \) is in the negative region of \( c \) if there is some path \( \zeta \subset c \) connecting \( p \) to a negative vertex of \( c \) such that \( \zeta \) does not intersect the isocontour. Since any path \( \zeta \subset c \) from a positive to a negative vertex must intersect the isocontour, the positive and negative regions form a partition of the square \( c \).

Figure 2.10 illustrates the positive and negative regions, colored red and blue, respectively, for each square configuration.

[R. Wenger, 2013]
Ambiguous Configurations

- Either works for marching squares, this isn't the case for 3D
3D: Marching Cubes

- Same idea, more cases [Lorensen and Cline, 1987]

Figure 2.16. Isosurfaces for twenty-two distinct cube configurations.
Multiple Isosurfaces

- Topographical maps have multiple isolines to show elevation trends
- Problem in 3D? **Occlusion**
- Solution? Transparent surfaces
- Issues:
  - Think about color in order to make each surface visible
  - Compositing: how do colors "add up" with multiple surfaces
  - How to determine good isovalues?

[J. Kniss, 2002]
Assignment 5

- Best-Selling Musical Artists
  - Multiple Views
  - Adjacency Matrix + Line Plot
  - Linked Highlighting
  - Filtering
- Due Wednesday, Nov. 23
Projects

• Keep working on implementation
• Be creative
• Think about interaction
• Presentations on the last two days of class (Nov. 29 & Dec. 1)
  - Submit current visualization code (or a link) to Blackboard
  - Presentation preferences (Tuesday or Thursday)
  - Upload full code to Blackboard beforehand in case of technical issues
Volume Rendering
Volume Rendering vs. Isosurfacing

(a) Direct volume rendered  
(b) Isosurface rendered

[Kindlmann, 1998]
(Direct) Volume Rendering

• Isosurfacing: compute a surface (triangles) and use standard computer graphics to render the triangles

• Volume rendering: compute the pixels shown directly from the volume information

• Why?
  - No need to figure out precise isosurface boundaries
  - Can work better for data with noise or uncertainty
  - Greater control over appearance based on values
Types of Volume Rendering Algorithms

- Ray casting
  - Similar to ray tracing, but use rays from the viewer
- Splatting:
  - Object-order, voxels splat onto the image plane
- Shear Warp:
  - Object-space, slice-based, parallel viewing rays
- Texture-Based:
  - 2D Slices: stack of texture maps
  - 3D Textures
Volume Ray Casting

Eye → Image Plane → Data Set

[Levine]
Volume Ray Casting

Image Plane

Data Set

Eye

[Levine]
How?

- Approximate volume rendering integral: light absorption & emission
- Sample at regular intervals along each ray
- Trilinear interpolation: linear interpolation along each axes (x,y,z)

- Not the only possibility, also "object order" techniques like splatting or texture-based and combinations like shear-warp
Compositing

- Need **one pixel** from all values along the ray
- Q: How do we "add up" all of those values along the ray?
- A: Compositing!

Different types of compositing
- First: like isosurfacing, first intersection at a certain intensity
- Max intensity: choose highest value
- Average: mean intensity (density, like x-rays)
- Accumulate: each voxel has some contribution

[Levine and Weiskopf/Machiraju/Möller]
Volume Ray Casting

Image Plane

Data Set

Eye

[Levine]
Volume Ray Casting

Image Plane

Data Set

Eye

For each pixel {
calculate color of the pixel
}
Compositing

• Need **one pixel** from all values along the ray
• Q: How do we "add up" all of those values along the ray?
• A: Compositing!
• Different types of compositing
  - First: like isosurfacing, first intersection at a certain intensity
  - Max intensity: choose highest value
  - Average: mean intensity (density, like x-rays)
  - Accumulate: each voxel has some contribution

[Levine and Weiskopf/Machiraju/Möller]
Types of Compositing

- max intensity
- accumulate
- average
- first

Intensity vs. Depth

[Levine and Weiskopf/Machiraju/Möller]
Types of Compositing

- max intensity
- accumulate
- average
- first

[Levine and Weiskopf/Machiraju/Möller]
Types of Compositing

- max intensity
- accumulate
- average
- first

[Levine and Weiskopf/Machiraju/Möller]
Types of Compositing

- **max intensity**
- **accumulate**
- **average**
- **first**

[Levine and Weiskopf/Machiraju/Möller]
Accumulation

- If we're not just calculating a single number (max, average) or a position (first), how do we determine the accumulation?
- Assume each value has an associated color (c) and opacity (α)
- Over operator (back-to-front):
  \[ c = α_f \cdot c_f + (1-α_f) \cdot α_b \cdot c_b \]
  \[ α = α_f + (1-α_f) \cdot α_b \]
- Order is important!

Blue Last

Blue First
Transfer Functions

• Where do the colors and opacities come from?
• Idea is that each voxel emits/absorbs light based on its scalar value
• …but users get to choose how that happens
• x-axis: color region definitions, y-axis: opacity

[Kindlmann]
Transfer Function Design

• Transfer function **design** is non-trivial!

• Lots of tools to help visualization designers to create good transfer functions

• Histograms, more attributes than just value like gradient magnitude
Multidimensional Transfer Functions
Multidimensional Transfer Functions
ParaView Examples