Lecture 18
Isosurfaces
Oct. 24, 2014
Agenda

- Lab 04 Questions?
Last Time
• Visualization of 1D, 2D, or 3D scalar fields
  – 1D scalar field: \( \Omega \in \mathbb{R} \rightarrow \mathbb{R} \)
  – 2D scalar field: \( \Omega \in \mathbb{R}^2 \rightarrow \mathbb{R} \)
  – 3D scalar field: \( \Omega \in \mathbb{R}^3 \rightarrow \mathbb{R} \)
  \( \rightarrow \) **Volume visualization!**
Grids (Meshes)

- Meshes combine positional information (geometry) with topological information (connectivity).
- Mesh type can differ substantial depending in the way mesh cells are formed.

scattered  uniform  rectilinear  structured  unstructured
Interpolation

http://en.wikipedia.org/wiki/Bicubic_interpolation
Bilinear Interpolation

- In rectangle

\[ P = (1 - v)Q_b(u) + vQ_t(u) \]
\[ = (1 - u)R_l(v) + uR_r(v) \]
Trilinear Interpolation

- In a cuboid (axis parallel)
  - general formula
    \[ \phi(x, y, z) = axyz + bxy + cxz + dyz + ex + fy + gz + h \]
  - with local coordinates
    \[ P = P_1 + u(P_2 - P_1) + v(P_4 - P_1) + w(P_5 - P_1) + uv(P_1 - P_2 + P_3 - P_4) + uw(P_1 - P_2 + P_6 - P_5) + vw(P_1 - P_4 + P_8 - P_5) + uvw(P_1 - P_2 + P_3 - P_4 + P_5 - P_6 + P_7 - P_8) \]
What is “Correct” Interpolation?
2D Scalar Field Visualization
Volume Visualization

- **2D visualization**
  - slice images (or multi-planar reformating MPR)

- **Indirect**
  - 3D visualization isosurfaces (or surface-shaded display SSD)

- **Direct**
  - 3D visualization (direct volume rendering DVR)
Volume Visualization

- **2D visualization** slice images (or multi-planar reformating MPR)
- **Indirect** 3D visualization isosurfaces (or surface-shaded display SSD)
- **Direct** 3D visualization (direct volume rendering DVR)
Techniques for 2D Scalar Field Vis

- Geometry-based:
  - Height fields, surface plots
  - Contours

- Color-based
  - Transfer Function / LUT Selection
Color Mapping

- Display scalar value through a color map, transfer function, color scale, or lookup table (LUT)
- Map interval on the real line to a path through the color space.

\[ f : R \rightarrow \{\text{RGB}, \text{HSV}\} \]
In Visualization, we use the concept of a **Transfer Function** to set color as a function of scalar value.

Scalar values -> [0,1] -> Colors

\[
Hue = 240 - 240 \times \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}}
\]

In OpenGL, the mapping of 1D texture.
Use the Right Transfer Function Color Scale to Represent a Range of Scalar Values

- Gray scale
- Intensity Interpolation
- Saturation interpolation
- Two-color interpolation
- Rainbow scale
- Heated object interpolation
- Blue-White-Red
A Gallery of Color Scales
• Example
  – Special color table to visualize the brain tissue
  – Special color table to visualize the bone structure

Original  Brain  Tissue
More Examples

Figure 1. Grey scale.
Figure 2. Saturation scale.
Figure 3. Spectrum scale.
Figure 4. Limited spectrum scale.
Figure 5. Redundant hue/lightness scale.
Figure 6. Heated-object scale.
Height Fields

• We use height in 1D plots, let’s use it in 2D plots
  • Direct intuition of the topography
  • Let the geometry convey the data
Contour Lines

- Draw lines of constant value.
- These bound regions of contiguous hues
  - Loops or lines through end of the dataset
- Usually best to use multiple contours
  - Why?
Compare
Isosurfaces
Volume Visualization

- 2D visualization
  slice images (or multi-planar reformating MPR)

- **Indirect**
  3D visualization
  isosurfaces (or surface-shaded display SSD)

- **Direct**
  3D visualization
  (direct volume rendering DVR)
Mount Kilimanjaro, Tanzania
Mount Kilimanjaro, Tanzania
Other examples
Other Examples
Colored Isosurfaces
Contours in 2D
Properties

• **Preimage** of a single scalar value:
  - Concept generalizes to any dimension

• **Closed** except at boundaries

• **Nested**: isocontours of different isovalues do not cross
  - Can consider the zero-set case (generalizes)
    - \( f(x,y) = v \iff f(x,y) - v = 0 \)

• Normals given by gradient vector of \( f() \)
Where are the data values?

Two solutions:
- Interpolate to get the “right” answer
  - Subsampling or raycasting
  - Dividing Cubes
- Approximate to get a “good” answer
  - Geometric primitives
  - Go cell by cell

Data value $f$ defined on grid points only

$S_v$ But we want a continuous, closed surface
Approach to Contouring in 2D

- **Idea**: Assign geometric primitives to individual cells
  - Will use line segments

- **Method**: Consider the “sign” of the values at vertices relative to if they are above or below the isovalue
  - Intersections MUST occur on edges with sign change

- Determine exact position of intersection by interpolate along grid edges
Approach to Contouring in 2D

• Contour must cross every grid line connecting two grid points of opposite sign

Get cell

Identify grid lines w/cross

Find crossings

Interpolate along grid lines

Primitives naturally chain together
# Cases

<table>
<thead>
<tr>
<th>Case</th>
<th>Polarity</th>
<th>Rotation</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Crossings</td>
<td>x2</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Singlet</td>
<td>x2</td>
<td>x4</td>
<td>8</td>
</tr>
<tr>
<td>Double adjacent</td>
<td>x2</td>
<td>x2 (4)</td>
<td>4</td>
</tr>
<tr>
<td>Double Opposite</td>
<td>x2</td>
<td>x1 (2)</td>
<td>2</td>
</tr>
</tbody>
</table>

\[ 16 = 2^4 \]
Ambiguities

• How to form lines?
Ambiguities

- Right or Wrong?
The Asymptotic Decider: Resolving the Ambiguity in Marching Cubes

Gregory M. Nielson  Bernd Hamann

Computer Science
Arizona State University
Tempe, AZ 85287-5406

Abstract

A method for computing isovalue or contour surfaces of a trivariate function is discussed. The input data are values of the trivariate function, \( F_{ijk} \), at the cuberille grid points \((x_i, y_j, z_k)\) and the output is a collection of triangles representing the surface consisting of all points where \( F(x, y, z) \) is a constant value. The method described here is a modification that is intended to correct a problem with a previous method.

1.0 Introduction

The purpose of this paper is to describe a method for computing contour or isovalue surfaces of a trivariate function \( F(x, y, z) \). It is assumed that the function is continuous and that samples over a cuberille grid (see Figure 1) are available. These values are denoted by \( F_{ijk} = F(x_i, y_j, z_k); i = 1, ..., N_x, j = 1, ..., N_y, k = 1, ..., N_z \). The problem is to compute the isovalue or contour surface

\[ S_\alpha = \{ (x, y, z) : F(x, y, z) = \alpha \}. \]
Ambiguities Occur when the Bilinear Interpolant Has Both Hyperbolic Arcs in the Grid Cell

Figure 6. Contours of bilinear interpolant
Ambiguities Occur when the Bilinear Interpolant Has Both Hyperbolic Arcs in the Grid Cell

Figure 6. Contours of bilinear interpolant
By Comparing Isovalue With the Value At Intersection of Asymptotes, Can Determine Which Pair of Contours Should be Connected
A Case Table Can Be Used To Implement The Algorithm
Isosurfaces of 3D Scalar Fields
Isosurfacing

- You’re given a big 3D block of numbers
- Make a picture
- Slicing shows data, but not its 3D shape
- Isosurfacing is one of the simplest ways
A little math

• **Dataset:** $v = f(x,y,z)$
• $f: \mathbb{R}^3 \rightarrow \mathbb{R}$
• **Want to find** $S_v = \{(x,y,z) \mid f(x,y,z) = v\}$
• **All the locations where the value of f is v**
• $S_v$ : **isosurface of f at v**
  • In 2D: isocontours (some path)
  • In 3D: isosurface

• **Why is this useful?**
Marching Cubes: A High Resolution 3D Surface Construction Algorithm

William E. Lorensen
Harvey E. Cline

General Electric Company
Corporate Research and Development
Schenectady, New York 12301

10,887 citations on Google Scholar
Compare with Soft Objects?

Medical data. Using a 3D volume, we construct a case table that defines triangle topology. The algorithm processes the 3D medical data in scan-line order and calculates triangle vertices using linear interpolation. We find the gradient of the original data, normalize it, and use it as a basis for shading the models. The detail in images produced from the generated surface models is the result of maintaining the inter-slice connectivity, surface data, and gradient information present in the original 3D data. Results from computed tomography (CT), magnetic resonance (MR), and single-photon emission computed tomography (SPECT) illustrate the quality and functionality of marching cubes. We also discuss improvements that decrease processing time and add solid modeling capabilities.

CR Categories: 3.3, 3.5

Additional Keywords: computer graphics, medical imaging, surface reconstruction

Existing 3D algorithms lack detail and sometimes introduce artifacts. We present a new, high-resolution 3D surface construction algorithm that produces models with unprecedented detail. This new algorithm, called marching cubes, creates a polygonal representation of constant density surfaces from a 3D array of data. The resulting model can be displayed with conventional graphics-rendering algorithms implemented in software or hardware.

After describing the information flow for 3D medical applications, we describe related work and discuss the drawbacks of that work. Then we describe the algorithm as well as efficiency and functional enhancements, followed by case studies using three different medical imaging techniques to illustrate the new algorithm's capabilities.
Marching Cubes

- “The” isosurface algorithm
- Lorensen + Cline (’87), Wyvill et al. (’86)
- Approximate, Efficient
- Involves many pre-computed tables
- Easy to understand, mostly easy to implement
- The foundation of how most people do isosurfacing
• The core MC algorithm
  – Cell consists of 4(8) pixel (voxel) values:
    \( (i+[01], j+[01], k+[01]) \)

1. Consider a cell
2. Classify each vertex as inside or outside
3. Build an index
4. Get edge list from table[index]
5. Interpolate the edge location
6. Compute gradients
7. Consider ambiguous cases
8. Go to next cell
• Step 1: Consider a cell defined by eight data values
• Step 2: Classify each voxel according to whether it lies
  – Outside the surface (value > isosurface value)
  – Inside the surface (value <= isosurface value)
• Step 3: Use the binary labeling of each voxel to create an index

Index:

\[
\begin{array}{cccccccc}
v1 & v2 & v3 & v4 & v5 & v6 & v7 & v8 \\
\end{array}
\]

\[
\begin{array}{cccccccc}
11110100 & 00110000 \\
\end{array}
\]
• 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
Case Table
8 Above
0 Below

1 case
1 case
6 Above

2 Below

3 cases
5 Above
3 Below

3 cases
4 Above
4 Below
7 cases
• Step 4 cont.: Get edge list from table
  
  - Example for

  Index = 10110001
  triangle 1 = e4, e7, e11
  triangle 2 = e1, e7, e4
  triangle 3 = e1, e6, e7
  triangle 4 = e1, e10, e6
• Step 5: For each triangle edge, find the vertex location along the edge using linear interpolation of the voxel values

\[ x = i + \left( \frac{T - v[i]}{v[i+1] - v[i]} \right) \]
• Step 6: Calculate the 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 isosurface)
• 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!)
• Summary
  – 256 Cases
  – Reduce to 15 cases by symmetry
  – Ambiguity in cases 3, 6, 7, 10, 12, 13
  – Causes holes if arbitrary choices are made
• Up to 5 triangles per cube
• Several isosurfaces
  – Run MC several times
  – Semi-transparency requires spatial sorting
• Examples

1 Isosurface

2 Isosurfaces

3 Isosurfaces

2 Isosurfaces
Challenges

- Ambiguities
- Requires examining every voxel
  - Can be fixed? How?
- Isovalue selection: what is a good isovalue?
- Poorly shaped, non-adaptive triangles
Challenge: Examining Fewer Voxels
The Span Space
Livnat, Shen, Johnson 96

• **Given:** Data cells in 8D

• **Past (active list):**
  Intervals in a 1D Value space

• **New:**
  • Points in the 2D Span Space
  • Benefit: Points do not exhibit any spatial relationships
Challenge: Mesh Quality
Dynamic Particles for Adaptive Sampling of Implicit Surfaces
Dynamic Particle Systems

Robust Particle Systems for Curvature Dependent Sampling of Implicit Surfaces
Robust Particle Systems for Curvature Dependent Sampling of Implicit Surfaces
182k triangles
41 minutes
0.18 min rr
0.94 avg rr

Topology, Accuracy, and Quality of Isosurface
Meshes Using Dynamic Particles.
M. Meyer et al., Vis 2007.
Lattice cleaving: Conforming tetrahedral meshes of multimaterial domains with bounded quality

Bronson, Levine, Whitaker
Comparison

CGAL
min: 0.01°

BioMesh3D
min: 0.18°

Cleaver
min: 6.49°
Boundary Surfaces on Frog Dataset

Input Size: 260x245x150
Dihedral angles: 6.06° -154.28°
14.8 million tets, ~70 sec.
Lec18
Required Reading
Volume Illustration: Non-Photorealistic Rendering of Volume Models

David Ebert  Penny Rheingans
Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore MD 21250
[ebert | rheingan]@cs.umbc.edu

Abstract
Accurately and automatically conveying the structure of a volume model is a problem not fully solved by existing volume rendering approaches. Physics-based volume rendering approaches create images which may match the appearance of translucent materials in nature, but may not embody important structural details. Transfer function approaches allow flexible design of the volume appearance, but generally require substantial hand tuning for each new data set in order to be effective. We introduce the volume illustration approach, combining the familiarity of a physics-based illumination model with the ability to enhance important features using non-photorealistic rendering techniques. Since features to be enhanced are defined on the basis of local volume characteristics rather than volume sample value, the application of volume illustration techniques requires less manual tuning than the design of a good transfer function. Volume illustration provides a flexible unified framework for enhancing structural perception of volume models through the amplification of features and the addition of illumination effects.


Keywords: Volume rendering, non-photorealistic rendering, illustration, lighting models, shading, visualization.

1 Introduction
For volume models, the key advantage of direct volume rendering is to convey that value distribution clearly and explicitly computed and included. The key challenge of direct volume rendering is to provide a flexible unified framework for enhancing structural perception of volume models through the amplification of features and the addition of illumination effects.

Traditionally, volume rendering has employed one of two approaches. The first attempts a physically accurate simulation of the volume are not to be completely obscured. Opacity and clarity is impossible if volume samples in the rear of the volume are not to be completely obscured. In particular, showing each volume sample with full opacity would result in the amplification of features and the addition of illumination effects.

For volume models, the key advantage of direct volume rendering is to provide a flexible unified framework for enhancing structural perception of volume models through the amplification of features and the addition of illumination effects.

The volume illustration approach combines the benefits of the two traditional volume rendering approaches in a flexible and parameterized manner. It provides the ease of interpretation resulting from familiar physics-based illumination and accumulation mechanisms with the flexibility and ease of interpretation of the more physics-based approach.

We propose a new approach to volume rendering: the augmentation of a physics-based rendering process with non-photorealistic rendering (NPR) techniques. NPR draws inspiration from such fields as art and technical illustration to develop automatic methods to synthesize images with an illustrated look from geometric surface models. Non-photorealistic rendering research has effectively addressed both the illustration of surface shape and the visualization of 2D data, but has virtually ignored the rendering of volume models. We describe a set of NPR techniques specifically for the visualization of volume data, including both the adaptation of existing NPR techniques to volume rendering and the development of new techniques specifically suited for volume models. We call this approach volume illustration.

The volume illustration approach combines the benefits of the two traditional volume rendering approaches in a flexible and parameterized manner. It provides the ease of interpretation resulting from familiar physics-based illumination and accumulation mechanisms with the flexibility and ease of interpretation of the more physics-based approach.

We propose a new approach to volume rendering: the augmentation of a physics-based rendering process with non-photorealistic rendering (NPR) techniques. NPR draws inspiration from such fields as art and technical illustration to develop automatic methods to synthesize images with an illustrated look from geometric surface models. Non-photorealistic rendering research has effectively addressed both the illustration of surface shape and the visualization of 2D data, but has virtually ignored the rendering of volume models. We describe a set of NPR techniques specifically for the visualization of volume data, including both the adaptation of existing NPR techniques to volume rendering and the development of new techniques specifically suited for volume models. We call this approach volume illustration.

The volume illustration approach combines the benefits of the two traditional volume rendering approaches in a flexible and parameterized manner. It provides the ease of interpretation resulting from familiar physics-based illumination and accumulation mechanisms with the flexibility and ease of interpretation of the more physics-based approach.

We propose a new approach to volume rendering: the augmentation of a physics-based rendering process with non-photorealistic rendering (NPR) techniques. NPR draws inspiration from such fields as art and technical illustration to develop automatic methods to synthesize images with an illustrated look from geometric surface models. Non-photorealistic rendering research has effectively addressed both the illustration of surface shape and the visualization of 2D data, but has virtually ignored the rendering of volume models. We describe a set of NPR techniques specifically for the visualization of volume data, including both the adaptation of existing NPR techniques to volume rendering and the development of new techniques specifically suited for volume models. We call this approach volume illustration.

The volume illustration approach combines the benefits of the two traditional volume rendering approaches in a flexible and parameterized manner. It provides the ease of interpretation resulting from familiar physics-based illumination and accumulation mechanisms with the flexibility and ease of interpretation of the more physics-based approach.
Direct volume rendering is a key technology for visualizing large 3D data sets from scientific or medical applications. Transfer functions are particularly important to the quality of direct volume-rendered images. A transfer function assigns optical properties, such as color and opacity, to original values of the data set being visualized. Unfortunately, finding good transfer functions proves difficult. Pat Hanrahan called it one of the top 10 problems in volume visualization in his inspiring keynote address at the 1992 Symposium on Volume Visualization. And it seems that today, almost a decade later, there are still no good solutions at hand. Or are there?

In a panel discussion at the Visualization 2000 conference, we pitched four of the currently most promising approaches to transfer function design against each other. The four approaches and their advocates are

- trial and error, with minimum computer aid (Will Schroeder);
- data-centric, with no underlying assumed model (Chandrajit Bajaj);
- data-centric, using an underlying data model (Gordon Kindlmann);
- image-centric, using organized sampling (Hanspeter Pfister).

Ahead of time, each of the four panelists received three volume data sets from Bill Lorensen. The data are static 3D scalar volumes sampled on rectilinear grids. The panelists' task was to create meaningful volume renderings using their respective approaches to transfer function design. During the panel session, each panelist presented a summary of the method and results of the visualization, including visual results (images and animations), performance (timings and memory use), and observations (how easy or hard it was, what the findings were, and so on). At the end of the panel session, Bill Lorensen discussed the content of the volume data, what an experienced visualization practitioner would have hoped to find, and how well the panelists' methods achieved this goal. Bill also announced a winner.

This was a unique event: alternative approaches to a pressing research problem went head-to-head, on multiple real-world data sets, and with an objective quality metric (Bill Lorensen). The panel took place in an atmosphere of lighthearted fun, but with a serious goal, namely to emphasize the importance of further research in transfer function design. This article presents the four methods in more detail and answers such questions as: How well did they do? Which method works best? And who won the bake-off?