Data Visualization (DSC 530/CIS 568)

Vector Field Visualization

Dr. David Koop
Visualizing Volume (3D) Data

- 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)

[© Weiskopf/Machiraju/Möller]
Volume Rendering vs. Isosurfacing

(a) Direct volume rendered  
(b) Isosurface rendered

[Kindlmann, 1998]
Volume Ray Casting

Image Plane

Data Set

Eye

[Levine]

D. Koop, DSC 530, Spring 2019
Volume Ray Casting

Image Plane

Data Set

Eye

[Levine]
Compositing

Pixel Compositing Schemes

- depth
- max intensity
- accumulate
- average
- first
- intensity
- opacity to show inside
- color to distinguish structures

[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 · c_f + (1-α_f) · α_b · c_b
  - α = α_f + (1-α_f) · 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
Multidimensional Transfer Functions
Projects

- Deadlines:
  - Presentations on May 6, 3-6pm (6-10 minutes per project)
  - Final Reports not late until the end of the day on the 7th

- Suggestions
  - Spend the most time showing your visualization or demoing your work
  - Provide background on dataset and the questions you wanted to answer
  - Comment on the design choices you made: what works well, what would you like to improve?
Examples of Vector Fields
Examples of Vector Fields

Wind [earth.nullschool.net, 2014]
Examples of Vector Fields
Examples of Vector Fields

Earthquake Ground Surface Movement [H. Yu et. al., SC2004]
Examples of Vector Fields

Gradient Vector Fields
Examples of Vector Fields

Wildfire Modeling [E. Anderson]
Visualizing Vector Fields

- Direct: Glyphs, Render statistics as scalars
- Geometry: Streamlines and variants
- Textures: Line Integral Convolution (LIC)
- Topology: Extract relevant features and draw them
Glyphs

- Represent each vector with a symbol
- Hedgehogs are primitive glyphs (glyph is a line)
- Glyphs that show direction and/or magnitude can convey more information
- If we have a separate scalar value, how might we encode that?
- Clutter issues
Streamlines & Variants

• Trace a line along the direction of the vectors
• Streamlines are always tangent to the vector field
• Basic Particle Tracing:
  1. Set a starting point (seed)
  2. Take a step in the direction of the vector at that point
  3. Adjust direction based on the vector where you are now
  4. Go to Step 2 and Repeat
Example

- Elliptical path
- Suppose we have the actual equation
- Given point \((x,y)\), the vector is at that point is \([v_x, v_y]\) where
  - \(v_x = -y\)
  - \(v_y = (1/2)x\)
- Want a streamline starting at \((0,-1)\)
Some Glyphs

\[ (x, y) \rightarrow (-y, \frac{1}{2}x), \text{ Step: 0.5} \]
Streamlines (Step 1)

\[ (x, y) \rightarrow (-y, (1/2)x), \text{ Step: } 0.5 \]
Streamlines (Step 2)

\[ [x, y] \rightarrow [-y, (1/2)x], \text{ Step: 0.5} \]
Streamlines (Step 3)

\[ [x, y] \rightarrow [-y, (1/2)x], \text{ Step: 0.5} \]
Streamlines (Step 4)

\[ [x, y] \rightarrow [-y, (1/2)x], \text{ Step: 0.5} \]
Streamlines (Step 10)

\[ [x, y] \rightarrow [-y, (1/2)x], \text{ Step: 0.5} \]
Streamlines (Step 19)

\[ x, y \rightarrow [-y, (1/2)x], \text{ Step: 0.5} \]
Euler Method

- Seeking to approximate integration of the velocity over time
- Euler method is the starting point for approximating this
- Problems?
Euler Method

• Seeking to approximate integration of the velocity over time
• Euler method is the starting point for approximating this
• Problems?
  - Choice of step size is important
Euler Method

• Seeking to approximate integration of the velocity over time
• Euler method is the starting point for approximating this
• Problems?
  - Choice of step size is important
  - Choice of seed points are important
Euler Method

- Seeking to approximate integration of the velocity over time
- Euler method is the starting point for approximating this
- Problems?
  - Choice of step size is important
  - Choice of seed points are important
- Also remember that we have a field—we don't have measurements at every point (interpolation)
Euler Quality by Step Size

Euler quality is proportional to $dt$ via Levine.
Numerical Integration

• How do we generate accurate streamlines?
• Solving an ordinary differential equation

\[
\frac{dL}{dt} = v(L(t)) \quad L(0) = L_0
\]

where \( L \) is the streamline, \( v \) is the vector field, and \( t \) is “time”

• Solution:

\[
L(t + \Delta t) = L(t) + \int_{t}^{t+\Delta t} v(L(t)) \, dt
\]
Higher-order methods

\[ \int_{t}^{t+\Delta t} v(L(t)) \, dt \]

• Euler method (use single sample)

• Higher-order methods (Runge-Kutta) (use more samples)

[A. Mebarki]
Higher-Order Comparison

Euler vs. Runge-Kutta

- RK-4: pays off only with complex flows
- Here approx. like RK-2

[via Levine]
Streamlines & Variants

- Steady vs. **Unsteady** flows
  - In unsteady flows, the vector field *changes* over time
- Variants: **Pathlines** and **Streaklines**

![Diagram showing pathlines, streaklines, and streamlines at different time steps](image-url)
Streamlines & Variants

- Steady vs. **Unsteady** flows
  - In unsteady flows, the vector field changes over time
- Variants: **Pathlines** and **Streaklines**

All are identical in steady flows!

[T. Möller]
Streamlines vs. Pathlines

Streamlines

Pathlines

[Weinkauf & Theisel, 2010]
Streaklines and timelines

streamlines  pathlines

streaklines  timelines

[via Levine]
Streamline Variants

Streaklines [NASA]

Stream Ribbons [Weiskopf/Machiraju/Möller]

Stream Tubes [Weiskopf/Machiraju/Möller]
Fig. 7. A streak surface in the Ellipsoid dataset as depicted in our interactive visualization tool. The surface is seeded upstream of the ellipsoid in the initial timestep and shows a prominent bubble that precedes the vortex formation.

Top: Overview; a timeline texture provides temporal orientation.
Bottom left: Surface textured with streak ribbons.
Bottom right: Without texturing, spatial and temporal orientation on the surface is lost.

Fig. 8. Evolution of a time surface in the Ellipsoid dataset. The surface is seeded on a rectangle located immediately downstream from the ellipsoid near the temporal beginning of the dataset and illustrates parts of the flow that remain close to the ellipsoid and twist to envelop the nascent vortex system as it forms. A two-dimensional color map helps identify distinct parts of the surface despite heavy overlap.

Fig. 9. Left images: Evolution of a time surface in the delta wing dataset, seeded parallel to the wing tip. The texture provides radial distance stripes to the wing tip for spatial orientation.
Right image: Despite numerical difficulties, the surface mesh remains well-conditioned.

[Krishnan et al., 2009]
2D Vector Field Visualization Techniques

For LIC, we used a box-shaped convolution kernel of width 20 pixels. The convolution was performed on a noise image where each pixel value was set to a uniform random value in the interval \( \frac{1}{C} \). To correct for loss of contrast due to the convolution, we applied an intensity mapping that took intensity \( I \) to \( I \left( \frac{4}{I + 1} \right) \).

For OSTR and GSTR, the code from [12], version 0.5, was modified to allow batch running without a graphical display and to have the optimization process stop after 60 seconds, without requiring manual intervention. OSTR was invoked with \( \text{opt 0.017} \) given to the \( \text{stplace} \) program (the “opt 0.017” parameter invokes optimal streamline placement using the algorithm from reference [12] with a separation choice of 0.017), while GSTR was invoked with \( \text{square 23.2} \) (streamlines 20 percent of the image width each centered on a square grid of 23 points in each direction), and both were plotted with “fancy arrows.” All other options to OSTR and GSTR were left as the defaults.

The tasks we used to evaluate the effectiveness of visualization methods needed to be representative of typical interactions that users perform with visualizations, simple enough that users could perform them enough times for us to calculate meaningful statistics, and able to provide an objective measure of accuracy.

We chose fluid mechanics as our “representative” scientific field because it frequently utilizes vector visualizations as a means of studying physical phenomena. We searched the literature and interviewed fluid mechanics researchers to identify good representative tasks. Two of the tasks, locating critical points and identifying their types, were derived from motivations behind the development of many of the visualization methods that we tested. Critical points are the salient features of a flow pattern; given a distribution of such points and their types, much of the remaining geometry and topology of a flow field can be deduced, since there is only a limited number of ways to join the streamlines. Beyond their importance for the interpretation of vector fields, these tasks are testable: we can measure how accurately a user determines the number, placement, and type of a collection of critical points in a given image.

Fig. 2 shows an example stimulus for locating all the critical points in a vector field. The GSTR method is used in this example. Users indicated the location of each critical point with the mouse and pressed the “Enter” key (or clicked on the “Next Image” button) when finished. Users were not allowed to delete or move their chosen points because editing operations tend to significantly increase the variability of response times, making statistical comparisons more difficult. We realized that this limitation on the user’s interactions might reduce accuracy but we felt that the benefit of more precise timing was an appropriate tradeoff.
Line Integral Convolution

- Goal: provide a global view of a steady vector field while avoiding issues with clutter, seeds, etc.
- Remember convolution?
- Start with random noise texture
- Smear according to the vector field
- Need structured data

[Weiskopf/Machiraju/Möller]
3D LIC

Figure 2: Visualization of a LIC volume: The flow field is explored using a clip plane, which is interactively translated. In the same way, complementary information is visually integrated for better orientation if fusion with another volume is performed. As demonstrated in Figure 15 a 3D–LIC calculation within the aorta is combined with the surrounding anatomy. In contrast, the fully opaque assignment shows the flow information directly at the outer surface of the vector field. According to Figure 2 this is useful if a clip plane is applied in order to explore the LIC volume.

In Figure 3 the setting of the transfer functions for color and opacity values is shown which leads to the visualization presented in Figure 4. Although arbitrary transfer functions are applicable a piecewise linear mapping is sufficient. The arrows indicate the location and the direction of simple manipulation operations which are required to adjust the lookup tables. As an additional orientation the intensity histogram of the volume data is displayed within the diagram. If an opaque representation is envisaged, opacity is set to a constant high value. However, it is useful to decrease it slightly in order to improve the visual continuity and impression. Thereby, stream lines become visible which are directly below the actual surface. Simultaneously, a linear ramp is specified for the luminance values enhancing the contrast of the resulting image. Within the histogram this ramp is positioned in the center of the main peak.

Figure 3: Intensity histogram and transfer functions for the visualization of the LIC volume shown in Figure 4: Setting for the opaque representation (left) — Setting for the semi-transparent representation (right).

The semi–transparent representation (right side) requires to use low opacity values for low data values and high opacity values for high data values. Further on, a linear ramp of high gradient is used in between in order to produce a smooth transition. Depending on the selected background color, the contrast is intensified if there is another linear ramp for opacity values that increases to lower data values. This is of importance if light background colors are chosen. The transfer function for luminance values is positioned within the transition from low to high opacity values. This leads to a good impression of depth, as can be seen on the right side of Figure 4. Moreover, the interactive adjustment of transfer functions is an efficient way to substitute the separate application of sparse noise textures as proposed in [15].

Figure 4: Simulated flow around wheel with different setting of transfer functions: (left) Opaque representation showing details at the surface and (right) semi–transparent representation efficiently substituting the application of sparse noise textures.

5 Clipping Functionality

Additional scalar fields such as density, pressure, or absolute value of velocity are frequently used in order to specify a volume of interest (VOI) to restrict the rendering process to significant parts of the flow. This VOI is usually applied a priori to the input texture or as a postprocess to the resulting 3D–LIC texture. Since this operation modifies the voxel data, it is impossible to change the VOI during the visualization process.

The above mentioned strategy aims at a visualization of the LIC volume as an opaque object extracted by the VOI. Due to the intricate and dense structure of stream lines inside a 3D–LIC texture, higher transparency will result in cluttered displays. In order to explore the interior structures, the use of clip planes is a straightforward approach. However, for a static visualization clip planes are not sufficient when visualizing 3D–LIC, because planar surfaces do not generally follow the direction of the flow, resulting in discontinuous stream lines. However, the interactivity provided is an efficient way to substitute the separate application of sparse noise textures as proposed in [15].

[Rezk-Salama et al., 1999]
Critical Points

• Remember finding min/max for functions?
• Want to understand the general structure of a field, not the exact values
• Find critical points, understand there is a general trend in between
• How?
  - Derivative for functions
  - For fields…gradients
Topology

• The general shape of data
• Visualizations that can be "stretched" to resemble each other are topologically equivalent
• Technically, continuous transformations don't change anything
• Connect critical points to obtain a general picture of the data
• Can talk about topology in both scalar and vector fields
2D Scalar Field Topology
2D Scalar Field Topology
Scalar Field Topology

• Examine the gradient (changes between points on the grid) of the scalar field

• Where the gradient is zero, we have critical points (max, min, saddle)

• Can build Reeb Graph, Contour Tree, or Morse-Smale Complex from this information to show the topology (with some reasonable assumptions about how the scalar field looks)
Scalar Field Topology

Key developments in topological data analysis (TDA):

1. Abstraction of the data: topological structures and their combinatorial representations
2. Separate features from noise: persistent homology

Two types of topological structures:

- Reeb Graph/Contour Tree/Merge Tree
- Morse-Smale Complex

[via Levine]
Vector Field Topology

• Instead of “guessing” correct seed points for streamlines to understand the field, try to identify structure (topology) of the field

Figure 7.1 A phase portrait.
Critical Points

Figure 1: Classification criteria for critical points. $R_1$ and $R_2$ denote the real parts of the eigenvalues of the Jacobian, $I_1$ and $I_2$ the imaginary parts.

[Helman & Hesselink]
Critical Points

- Critical Points
  - Find where the vector field vanishes (the zero vector or undefined)
  - Attracting Nodes (Sinks), Repelling Nodes (Sources), Attracting Foci, Repelling Foci, Saddles, Centers

- How to find such points?
  - Can use a similar idea to Marching Cubes
  - Use the eigenvalues of the Jacobian matrix to classify
Topological Skeleton
More Examples

[Levine]