Review of Flow Vis for Lower Dimensional Flow Data

- **Direct**: overview of vector field, minimal computation, e.g. glyphs (arrows), color mapping
- **Texture-based**: covers domain with a convolved texture, e.g., Spot Noise, LIC, ISA, IBFV(S)
- **Geometric**: a discrete object(s) whose geometry reflects flow characteristics, e.g. streamlines
- **Feature-based**: both automatic and interactive feature-based techniques, e.g. flow topology
Vector Field Visualization in 3D
Review of Data Structure

Regular (uniform), rectilinear, and structured grids

Alternative:

tetrahedral volume elements:
unstructured
Direct Method (Arrow Plot)

Source:
http://docs.enthought.com/mayavi/mayavi/mlab.html
Direct Method (Arrow Plot)

Source:
http://docs.enthought.com/mayavi/mayavi/mlab.html
Direct Method – Volume Rendering of Certain Scalar Characteristics

Acceleration

Local shearing

Q
Issues of Arrows in 3D

Common problems:

• Ambiguity
• Perspective shortening
• 1D objects generally difficult to grasp in 3D

Remedy:

• 3D-Arrows (are of some help)
Texture-Based Method

Volume LIC

- Victoria Interrante and Chester Grosch (*IEEE Visualization 97*).
- A straightforward extension of LIC to 3D flow fields.
- Low-pass filters *volumetric noise* along 3D streamlines.
- Uses *volume rendering* to display resulting 3D LIC textures.
- Very time-consuming to generate 3D LIC textures.
- Texture values offer no useful guidance for transfer function design due to lack of *intrinsic physical info* that can be exploited to distinguish components.

⇒ Very challenging to clearly show *flow directions and interior structures through a dense texture volume.*
3D IBFV

for $i = 0$ to $N-1$
\{ 
  if ($i>0$) 
    do 1D Z-axis advection from $S_{i-1}$ to $S_i$ 
  if ($i<N-1$) 
    do 1D Z-axis advection from $S_{i+1}$ to $S_i$ 
  do 2D IBFV-based advection in the slice $S_i$
\} 

[Telea and van Wijk Vis03]
Recent Advances in 3D Texture-based Method

Codimension-2 illumination

Different seeding strategies

[Falk and Weikopf 2008]
Streamlines:

Theory: \( \mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \leq u \leq t} \mathbf{v}(\mathbf{s}(u)) \, du \)

Practice: Numerical integration such as Euler, RK2, RK4, etc.

Important: interpolation scheme, **seeding!!**
3D Seed Placement

• The placement of seeds directly determines the visualization quality
  – Too many: scene cluttering
  – Too little: no pattern formed

• It has to be in the right place and in the right amount

A bad seeding example
Some Existing Work

• 3D flow topology-guided [Ye et al. 2005]

• Image-based streamline placement [Li and Shen 2007]

• Priority streamlines [Schlemmer et al. 2007]

• Entropy-guided seed placement [Xu et al. 2010]
Open Issues

• Seed placement in 3D (occlusion and clarity)
• Techniques for handling big data
• Flow field navigation and interaction
• Human perception and user evaluation
Streamline filtering and/or selection techniques
Streamline Bundling

[Yu et al. 2012]
Streamline Bundling

[Yu et al. 2012]
View-dependent streamline selection

initial pool

initial pool

initial pool

selected streamlines

selected streamlines

selected streamlines

[Tao et al. 2013]
Streamline rendering techniques
Illuminated Streamlines

Use lighting to improve spatial perception of lines in 3D.

This can to some extent reduce the 3D cluttering issue.

Open Source: http://www.scivis.ethz.ch/research/projects/illuminated_streamlines

[Zockler et al. 96, Mallo et al. 2005]
Opacity Optimization for 3D Line Fields

Figure 1: Applications of our interactive, global line selection algorithm. Our bounded linear optimization for the opacities reveals user-defined important features, e.g., vortices in rotorcraft flow data, convection cells in heating processes (Rayleigh-Bénard cells), the vortex core of a tornado and field lines of decaying magnetic knots (from left to right).

(a) Given is a set of polylines.
(b) Discretize polylines into $n$ segments (here: $n = 6$).
(c) Compute per-segment opacity $\alpha_i$ by energy minimization.
(d) Interpolate opacities between adjacent segments for final rendering.

[Gunthe et al. 2013]
Other Geometric-Based Methods

Streamribbons, Streamtubes, Stream surfaces, Flow volumes
**streamribbon:**
a ribbon (surface of fixed width) always tangent to the vector field
shows rotational (or twist) properties of the 3D flow
Streamribbon generation:

- Start with a 3D point $\mathbf{x}_{i=0}$ and a 2nd one $\mathbf{y}_{i=0}$ in a particular dist. $d$, i.e. $|\mathbf{x}_i-\mathbf{y}_i|^2 = d^2$
- Loop:
  - Integrate from $\mathbf{x}_i$ to yield $\mathbf{x}_{i+1}$
  - Do an integration step from $\mathbf{y}_i$ to yield $\mathbf{z}$
    renormalize the distance between $\mathbf{x}_{i+1}$ & $\mathbf{z}$ to $d$, i.e. $\mathbf{y}_{i+1} = \mathbf{x}_{i+1} + d \cdot (\mathbf{z} - \mathbf{x}_{i+1}) / |\mathbf{z} - \mathbf{x}_{i+1}|$
- End streamribbon integration if necessary
What about **Stream Surfaces**?

- The computation of stream surfaces is similar to streamribbon.
- However, now the seeding points are typically more than two.
- **Also, during the integration, we may need to adaptively add or remove seeds (i.e. handling divergence, convergence, and shear).**
- Triangulating the stream surface between neighboring streamlines is easy to achieve.
- What is the other challenge?
Where to put seeds to start the integration?

Seeding along a straight-line
Allow user exploration
[Weiskopf et al. 2007]

Seeding along the direction that is perpendicular to the flow leads to stream surface with large coverage
[Edmunds et al. EuroVis2012]
How about automatic evenly-spaced stream surface placement?

Where to start?  How to proceed?  Render

[Edmunds et al. TPCG 2012]
Rendering of stream surfaces

- Stream arrows (Löffelmann et al. 1997)

- Texture advection on stream surfaces (Laramee et al. 2006)
Rendering of stream surfaces

Illustrative visualization
• Using transparency and surface features such as silhouette and feature curves.

Abraham/Shaw’s illustration, 1984

[Hummel et al. 2010]

[Born et al. Vis2010]
Geometric FlowVis in 3D

**flow volume:** a volume whose surface is everywhere tangent to the flow

**streamtube:** shows convergence and divergence of flow (similar to streamribbon)
# Relation to Seed Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Seed Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streamline,...</td>
<td>1D 0D (point)</td>
</tr>
<tr>
<td>Streamribbon</td>
<td>2.5D 1D (line segment)</td>
</tr>
<tr>
<td>Streamtube</td>
<td>2.5D 1D (circle)</td>
</tr>
<tr>
<td>Stream surface</td>
<td>2.5D 1D (curve)</td>
</tr>
<tr>
<td>Flow volume</td>
<td>3D 2D (patch)</td>
</tr>
</tbody>
</table>
Feature-Based Methods

Topology of 3D Steady Flows
3D Flow Topology

• Fixed points

node-source

spiral-sink

saddle-spiral

saddle-node

• Can be characterized using 3D Poincaré index

• Both line and surface separatrices exist
3D Cycles

• Similar principle as in 2D
  – Isolate closed cell chain in which streamline integration appears captured
  – Start stream surface integration along boundary of cell-wise region
  – Use flow continuity to exclude reentry cases

Challenging to \textit{strange attractor}

https://en.wikipedia.org/wiki/Lorenz_system
3D Topology Extraction

• Cell-wise fixed point extraction:
  – Compute root of linear / trilinear expression
  – Compute Jacobian at found position
  – If type is saddle compute eigenvectors

• Extract closed streamlines
• Integrate line-type separatrices
• Integrate surface separatrices as stream surfaces
Saddle Connectors

Topological representations of the Benzene data set.
(left) The topological skeleton looks visually cluttered due to the shown separation surfaces.
(right) Visualization of the topological skeleton using connectors.
Source: Weinkauf et al. VisSym 2004
Additional Readings


Acknowledgment

Thanks for the materials

• Prof. Robert S. Laramee, Swansea University, UK
• Dr. Christoph Garth, University of Kaiserslautern, Germany