Advection methods comparison

Stream-ribbon

- We really would like to see vorticities, i.e. places where the flow twists.
- A point primitive or an icon can hardly convey this.
- Idea: trace neighboring particles and connect them with polygons.
- Shade those polygons appropriately and one will detect twists.

Stream-tube

- Generate a stream-line and connect circular crossflow sections along the stream-line.

Stream Ribbons

- First of two other line icons...
  - Narrow surfaces between two adjacent streamlines.
  - Built from a front with only two particles.
  - Reflect flow divergence through changing width, and vortices are shown in the degree of twist in the ribbon.

Streamribbons

- Vorticity
  - Formally: \( \omega \) - measure of rotation of vector field.
  - Streamwise vorticity:
    \[
    \Omega = \frac{\mathbf{v} \cdot \phi}{|\mathbf{v}| \cdot |\omega|}
    \]
  - Where \( \mathbf{v} \) - instantaneous velocity.
- Flow divergence
  - Measure of the spread of the flow.
Streamtube example
- 6 streamlines
- Streamline added for comparison

Streamtube example
- 6 streamlines
  - Alternative generation
  - Stream polygon
    - Rotation of edges $\Rightarrow$ vorticity
    - Area of polygon $\Rightarrow$ divergence

Stream-balls
- Another way to get around diverging streamlines
- Simply put implicit surface primitives at particle traces - at places where they are close they’ll merge elegantly ...

Streamballs
- Stream balls:
  - Display stream lines as chains of balls and use their radius and color to encode scalar values

Global Texture Techniques
- Two approaches:
  - Advection-based
  - Streamline placement
  - Icon placement or smearing
    - Spot Noise
    - Vector Kernel
    - Line Integral Convolution
    - Line Bundles

2D vector field visualization
- We want to visualize a function
- $F: \Omega \rightarrow \mathbb{R}^2$
- with $F$ given only at certain vertices
  - $F \leftrightarrow F' = \left( \begin{array}{c} F_x' \\ F_y' \end{array} \right)$
Mappings - Hedgehogs, Glyphs

- analogous to tufts or vanes from experimental flow visualization
- clutter the image real quick
- maybe ok for 2D
- not very informative

2D vector field visualization

- First idea
  - Visualize the two scalar fields $F_x$ and $F_y$
  - As the two components are normally not independent, this usually provides no insight.

- Second idea
  - Data is supposed to represent the direction of moving particles
  - Visualize the “flow”
  - Do it by means of glyphs or particle tracing

2D vector field visualization

- Glyphs
  - We display the vectors using arrows as geometric primitives
  - Draw an arrow at every vertex of the grid
    - Length corresponds to magnitude of velocity
    - Direction corresponds to flow direction

2D vector field visualization

- Pros and Cons of glyphs
  - Simple
  - 3D effects
  - Heavy load in the graphics subsystem
  - Inherent occlusion effects
  - Poor results if magnitude of velocity changes fast
  - Use arrows of constant length and color code magnitude

Global techniques

- Display the entire flow field in a single picture
- Minimum user intervention
- Example: Hedgehogs (global arrow plots)

Mappings - Hedgehogs, Glyphs

- Put “icons” at certain places in the flow
  - e.g. arrows - represent direction & magnitude
- Other primitives are possible

- oriented lines
  - vertex
2D vector field visualization

Advection-Based Textures

Streamline Placement

Movie

Icon Placement

Image-filling and randomized

More artistic

Hedgehogs or arrow plots: uniform grid

Image-filling and randomized
Arrows

But in 3D it suffers from perception problems:

Is it this?

or this?

Of course the picture quickly gets cluttered too.

Arrows

Arrows can be used successfully in 3D as follows:
- by slicing the volume, and attaching arrows (with shadow effects) to the slice plane - this gives a hedgehog effect
- by giving more spatial cues - drawing arrows as true 3D objects

but clutter again a problem!

(BTW - Eulerian or Lagrangian?)

Example of a complex Glyph

An image kernel moves in a discrete and jittered path left to right and top to bottom across the image screen.
- The kernel samples the scalar and vector fields.
- The vector field uses a weighted probability of drawing an anti-aliased line across the kernel.
- The line is oriented in the vector field direction.
- The line may also be semi-transparent.
- The probability is proportional to the vector magnitude.

Vector Kernel

An image kernel moves in a discrete and jittered path left to right and top to bottom across the image screen.
- The kernel samples the scalar and vector fields.
- The vector field uses a weighted probability of drawing an anti-aliased line across the kernel.
- The line is oriented in the vector field direction.
- The line may also be semi-transparent.
- The probability is proportional to the vector magnitude.

Spot Noise for Flow Visualization

Spots of random size and intensity drawn in a plane give a texture

Texture defined as an intensity function:

\[ f(\mathbf{x}) = \sum a_i h(\mathbf{x} - \mathbf{x}_i) \]

where \( \mathbf{x}_i \) is random position, \( a_i \) is random scale (zero mean), and \( h \) is the spot function - zero everywhere except for small area (here circular).
Spot Noise for Flow Visualization

- Different textures result from different spot shapes
- Aligning the shape of the spot with the direction of flow gives a good visualization effect
- In direction of flow, scale proportional to \((1 + |v|)\), \(|v| = \text{velocity magnitude}\)
- At 90 degrees to flow, scale proportional to \(1 / (1 + |v|)\)

Spot Noise Example

Spot Noise

- Uses small motion blurred particles to visualize flows on stream surfaces
- Particles represented as ellipses with their long axes oriented along the direction of the flow
- I.e. we multiply our kernel \(h\) with an amplitude and add a phase shift!
- Hence - we convolve a spot kernel in spatial domain with a random sequence (white noise)

Spot Noise Example

Spot Noise

- examples of white noise:
  - set of random values on a grid
  - Poisson point process - a set of randomly scaled delta functions randomly placed (dart throwing)
- variation of the data visualization can be realized via variation of the spot:
  \[ f(x) = \sum_i a_i b(m(d(x_i)), x - x_i) \]
  - \(d\) - data value
  - \(m\) - parameter mapping

Rendering - Spot Noise

Different size
Different profiles

Rendering - Spot Noise

Examples of white noise:
- set of random values on a grid
- Poisson point process - a set of randomly scaled delta functions randomly placed (dart throwing)
- variation of the data visualization can be realized via variation of the spot:
  \[ f(x) = \sum_i a_i b(m(d(x_i)), x - x_i) \]
  - \(d\) - data value
  - \(m\) - parameter mapping
**Rendering - Spot Noise**

- Scalar - use + shape for positive values, x shape for negative values
- Change the size of the spot according to the norm of the gradient
- Vector data - use an ellipse shaped spot in the direction of the flow

**Spot Noise for Flow Visualization**

- If velocity direction varies rapidly, result is not very successful
- Improvement achieved by 'bending' spot along the streamline

**Flow Over a Surface**

Wall friction displayed using oil and paint - wind evaporates oil and paint leaves white traces

**Spot Noise Example**

Numerical simulation of flow visualized using spot noise

**Spot Noise Movie**

Numerical simulation of flow visualized using spot noise

**Learning More about Spot Noise**

- Spot noise has been developed by researchers in the Netherlands
  - van Wijk and de Leeuw
  - see http://www.cwi.nl/~wimc/spotnoise.html
- Thanks to Wim de Leeuw for the images used in these slides
Rendering - LIC

- Similar to spot noise
- Embed a noise texture under the vector field
- Difference - integrates along a streamline

Line Integral Convolution (LIC)

- Essence of method is:
  - Consider a white noise texture, \( T(x,y) \)
  - For each pixel, set its intensity as a function (e.g., average) of values of \( T \) along a short streamline segment through the pixel
  - This has effect of correlating the resulting pixel values along streamlines, so a sense of the flow direction is obtained

LIC Example

Flow over surface of car - from CIRA, Italy
Italian Aerospace Research Centre

LIC Development - Oriented LIC

- Original LIC shows direction of flow but not orientation (i.e., \( \rightarrow \) or \( \leftarrow \))
- Oriented LIC uses a sparse texture and a weighting of samples along streamline to give orientation effect

Learning More about LIC

- Original LIC
- Fast LIC
  - www.zib.de/Visual/projects/vector
  - Includes the Lic Factory - Java applet
- Oriented LIC
  - R. Wegenkittl and E. Groller
  - www.cg.tuwien.ac.at/research/vis/dynsys/frolic/
Algorithm - Cabral & Leedom '93

Rather than deposit energy on each pixel, integrate the icon weights with a white noise function.

2D LIC

What do LIC–images look like?
- smooth change of color along time line, but rough alternation perpendicular to it
- visual impression like a painting
- depicts the directional structure of the vector field but not the magnitude of the flow

LIC Example: 2D Airfoil

Line Integral Convolution (LIC)
**LIC Example: Space Shuttle**

Han Wei Shen

4/30/2003

---

**Enhancements**

- LIC is computationally expensive
  - Fast LIC (Stalling and Hege '95)
  - PLIC (Verma et al '99)
- LIC is primarily for regular Cartesian grids
  - Curvilinear LIC (Forsell '95)
- LIC is primarily 2D
  - 3D Volume LIC (Shen et al '96, Interrante '97, Rezk-Salama '99)

4/30/2003

---

**Fast LIC: an order of magnitude faster**

(Stalling '95)

4/30/2003

---

**3D Volume LIC**

(Rezk-Salaman '99)

4/30/2003

---

**LIC and Stream Surfaces**

http://www.zib.de/Visual/projects/sector

4/30/2003

---

**LIC Factory**

- Cool website
  - LIC Factory
Movement Without Motion

An oscillating filter weight can produce the appearance of motion.

Movement in LIC

The integral weights are shifted to form an oscillation along the streamline.

Global Winds movie of a single timestep.

Changing the Texture

By controlling the white noise to band-limited noise with varying frequencies, different textures can be generated.

Textured Splats

Use graphics hardware at the data points to brush a texture image canvas for that data point. Do this in a depth-sorted order to get a volume rendering.

Texture Splats

Crawfis, Max 1993

Extended splatting to visualize vector fields

Used simple idea of “textured vectors” for visualization of vector fields

Texture Splats - Vector Viz

The splat would be a Gaussian type texture

How about setting this to an arbitrary image?

How about setting this to an image including some elongated particles representing the flow in the field?

Texture must represent whether we are looking at the vector head on or sideways.
Texture Splats

- Texture images
- Appropriate opacities

How do you get them to "move"?
Just cycle over a periodic number of different textures (rows)

More global techniques

- Texture Splats

- See VectorVizNotes

Line Bundles

- For vector fields only, just use a collection of semi-transparent lines.

Line bundles