Flow Visualization
Fluid Simulation

Looking at vector fields...

how?
FIELD VISUALIZATION
METHODS

Physical analogies
- particle advection
- static or dynamic
- ink advection

Issues:
- missing important detail

CS 176 SPRING 2011
Texture Methods

Advection

- advect texture coordinates with velocity; display regular grid with original texture
  - next time step: use previous texture
  - blend with exponential decay
LINE INTEGRAL CONVLTN.

Continuous version

- integral curves of the flow
  \[ \frac{d}{dt} \sigma(t) = v(\sigma(t)) \]
- particles in flow
- follow one traj.
Integrate *noise*

- output pixel:

  \[ I(x_0) = \int_{s_0-L}^{s_0+L} k(s - s_0) T(\sigma(s)) \, ds \]

- high correlation along flow
- low correlation orthogonal to flow

CS 176 Spring 2011
Making it Fast

Pixels along path are correlated

- curve tracing for each output pixel far too expensive
  - step along curve!

\[
I(x_2) = I(x_1) - \int_{s_1-L}^{s_1-L+\Delta s} T(\sigma(s))ds + \int_{s_1+L}^{s_1+L+\Delta s} T(\sigma(s))ds
\]

- accumulate results in pixels crossed
- renormalize at end

CS 176 Spring 2011
Details

Streamline integration

- high order integrator recommended
  - may make LARGE steps
  - need to interpolate along path
- aliasing...
  - step size 1/2 of texture cell
- path length: 1/10 image size
ANIMATION

Slide integration along streamline

- blend multiple images with different phases
- contrast and brightness (renormalize)
Image Based Flow Viz

Treat images as basic primitive
- LIC as blending of advected images
  - setup mesh with advected texture coordinates
  - render and blend; repeat
    \[ F(p_k; k) = (1 - \alpha)F(p_{k-1}; k-1) + \alpha G(p_k; k) \]
- what image?
  - random noise (gives standard LIC)
WHAT IMAGE?

Issues to consider

- aliasing in time and space
- pink, not white noise
- contrast, boundaries
NICE EXAMPLE

van Wijk’s ibfv program
- testbed for visualization
DIRECT VISUALIZATION

Integral curves as shaded lines
- “hair” like analogy
- how to shade lines?
- usually objects co-D 1

New shading model
- objects of co-D 2
- think cylinders of infinitesimal size
**SHADED LINES**

What normal?

- standard model

\[ I = k_a + k_d (L \cdot N) + k_s (V \cdot R)^n \]

\[ L \cdot N = |L_N| = \sqrt{1 - (L \cdot T)^2} \]

\[ V \cdot R = V \cdot (L_T - L_N) \]

\[ = \sqrt{1 - (L \cdot T)^2} \sqrt{1 - (V \cdot T)^2} - (L \cdot T)(V \cdot T) \]
**Implementation**

Shading is function of L, T, V

- texture lookup
- inner products via texture xform

\[
M = \frac{1}{2} \begin{pmatrix}
L_1 & V_1 & 0 & 0 \\
L_2 & V_2 & 0 & 0 \\
L_3 & V_3 & 0 & 0 \\
1 & 1 & 0 & 2
\end{pmatrix}
\]
Implementation

- shading too bright: for diffuse
  \[ I_d = k_d (L \cdot N)^p \]
- add transparency: “wispy” tails
- how to seed stream lines?
  - user driven
  - Monte Carlo; Voronoi
  - divergence… (reseed)