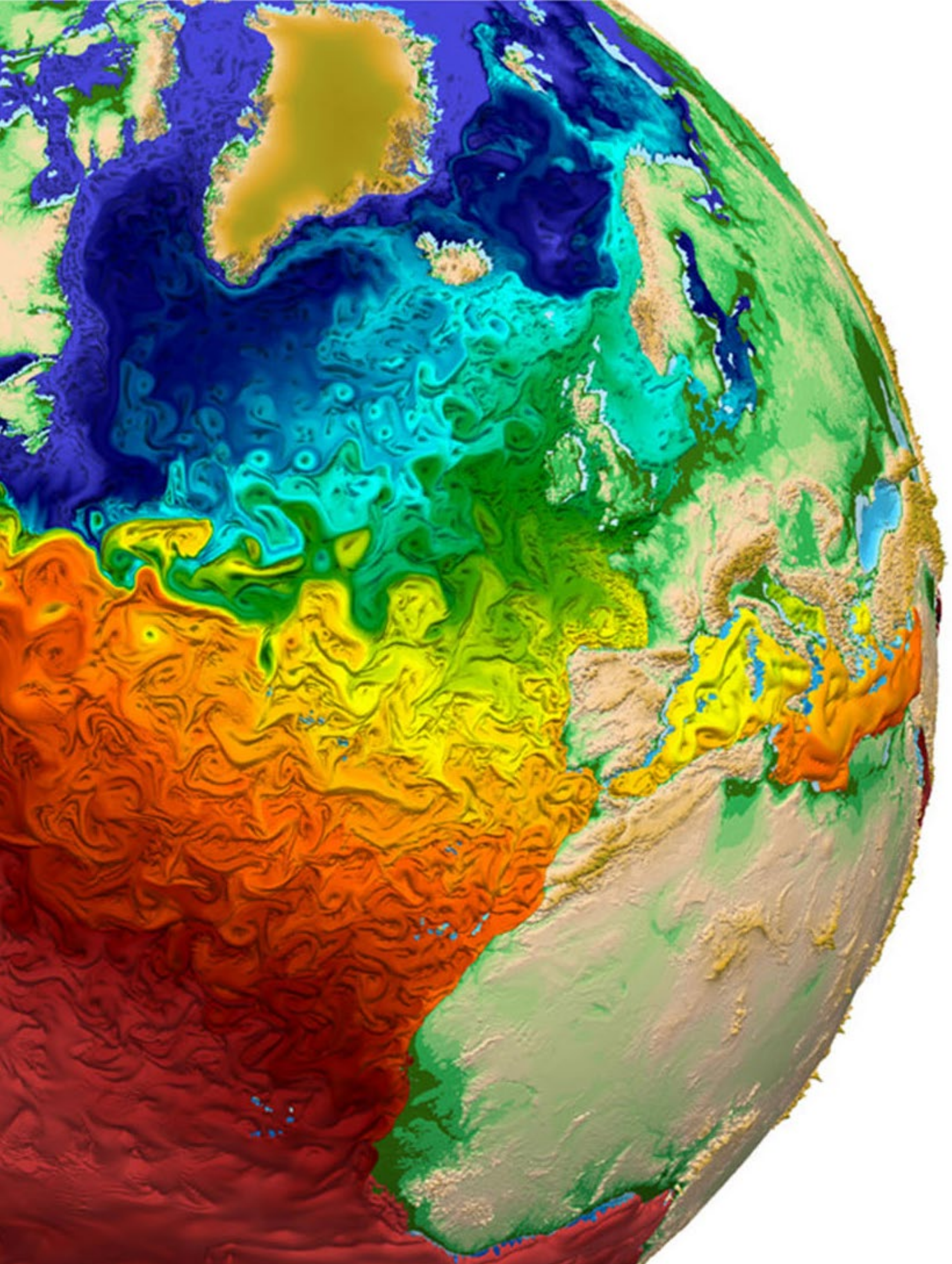


Vector Visualization

Stream Objects

Scientific Visualization

Professor Eric Shaffer

ILLINOIS

Stream Objects

Main Idea

- think of the vector field $\mathbf{v} : D$ as a flow field
- choose some 'seed' points $s \in D$
- move the seed points s in $\mathbf{v}$
- show the trajectories

Streamlines

- assume that $\mathbf{v}$ is not changing in time (steady-states)
- for each seed $p_o \in D$
  - the streamline S seeded at p_o is given by

$$S = \{p(\tau), \tau \in [0, T]\}, p(\tau) = \int_{t=0}^{T} \mathbf{v}(p)dt, \quad \text{where } p(0) = p_0$$

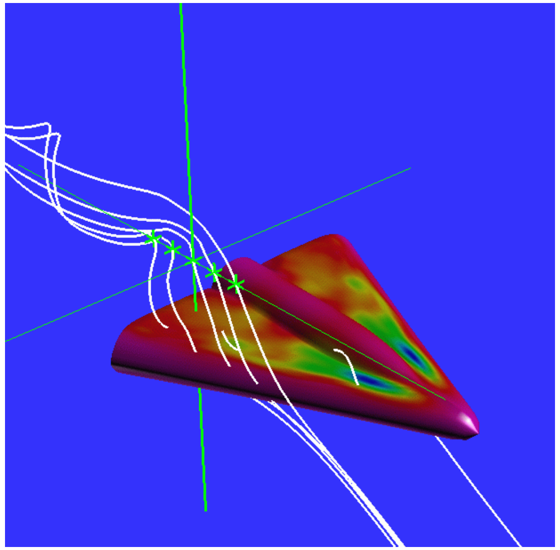
integrate p_o in vector field $\mathbf{v}$ for time T

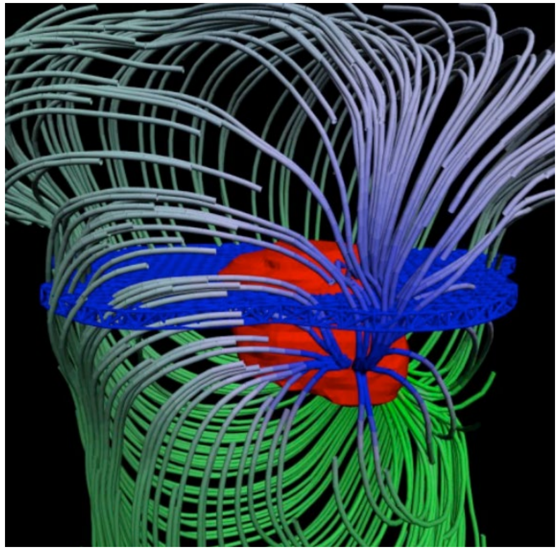
- if v is time dependent $\mathbf{v}=\mathbf{v}(\mathrm{t})$, streamlines are called particle traces

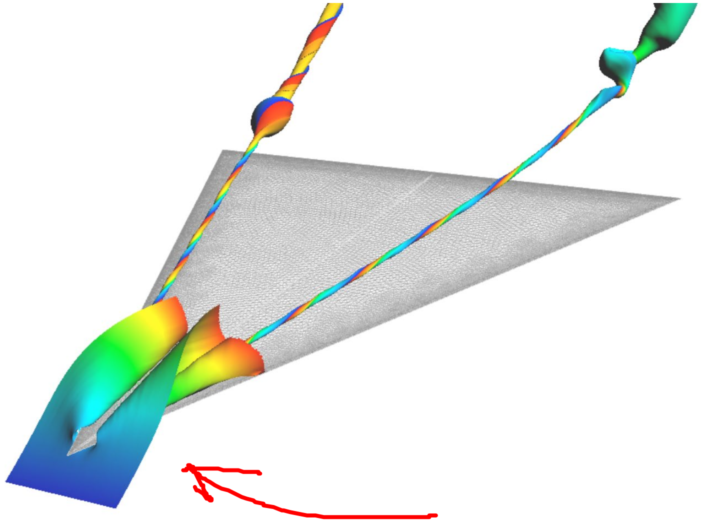
Stream Objects

Stream-{lines | tubes | ribbons | polygons |…}

- Vector glyph plots show the **trajectories over a short time** of trace particles released in the vector fields
- Stream objects show the **trajectories for longer time intervals** for a given vector field



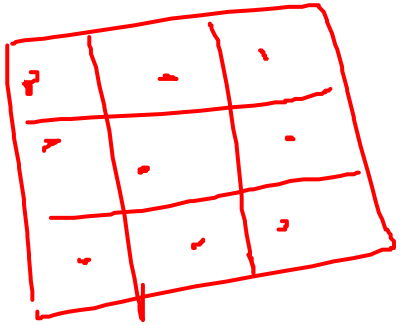


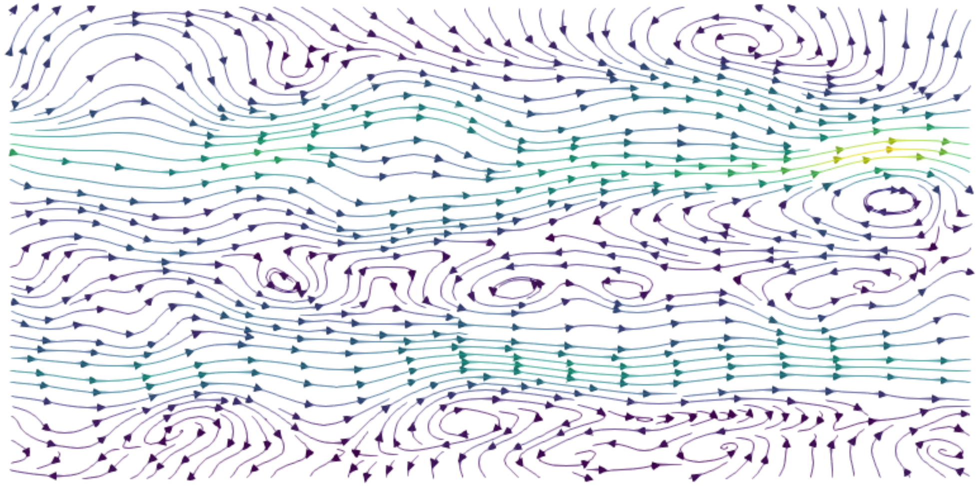


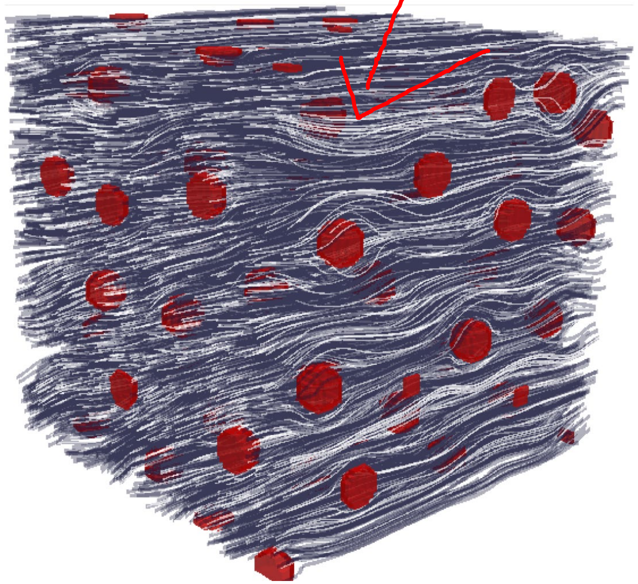
Stream Objects

Stream-{lines | tubes | ribbons | polygons |…}

- Displaying streamlines is a local technique
- You can only visualize the flow directions initiated from a few particles
- Too many streamlines and the scene becomes cluttered
- Location of the seed particles is crucial decision







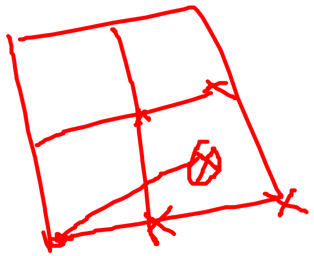
ILLINOIS

Stream Objects

Construction

- numerically integrate

$$S = \{p(\tau), \tau \in [0,T]\}, p(\tau) = \int_{t=0}^{\tau} \mathbf{v}(p)dt, \quad \text{where } p(0) = p_0$$



- discretizing time yields

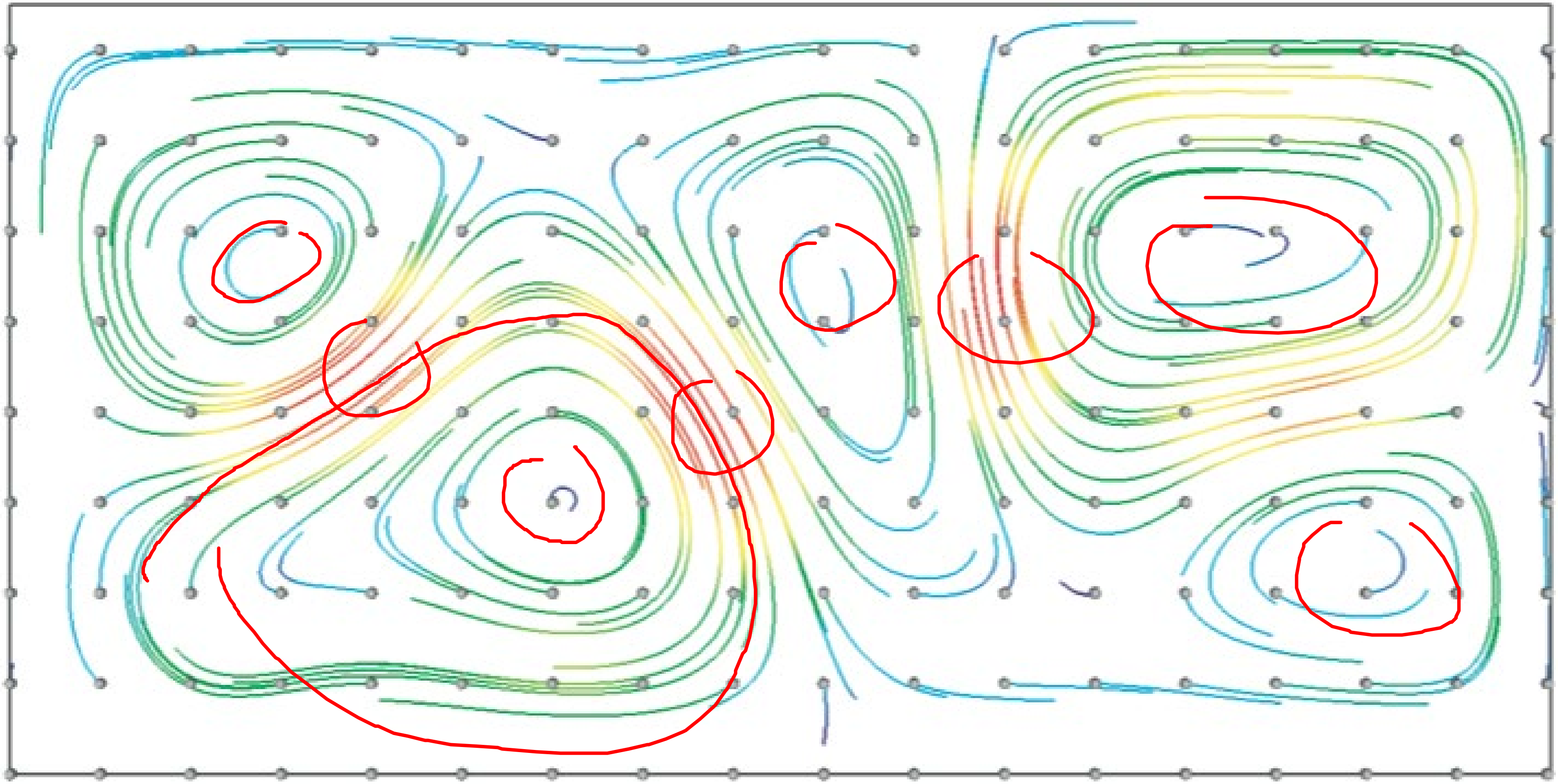
$$\int_{t=0}^{\tau} \mathbf{v}(p)dt = \sum_{i=0}^{\tau/\Delta t} \mathbf{v}(p_i)\Delta t \quad \text{where } p_i = p_{i-1} + \mathbf{v}_{i-1}\Delta t$$

(simple Euler integration)

RK4 would be better

- Euler integration
  - we consider $\mathbf{v}$ constant between two sample points p_i and $p_{\mathrm{i+1}}$
  - we compute $\mathbf{v}(p)$ by linear interpolation within the cell containing p
  - variant: use $\mathbf{v}(p)/\|\mathbf{v}(p)\|$ instead of $\mathbf{v}(p)$ in integral
  - S will be a polyline, $\mathrm{S} = \{p_\mathrm{i}\}$
- stop when $\tau = T$ or $\mathbf{v}(p) = 0$ or $p \notin D$
  - what does $\tau = T$ mean when we use $\mathbf{v}(p)/\|\mathbf{v}(p)\|$?

Example: Streamlines

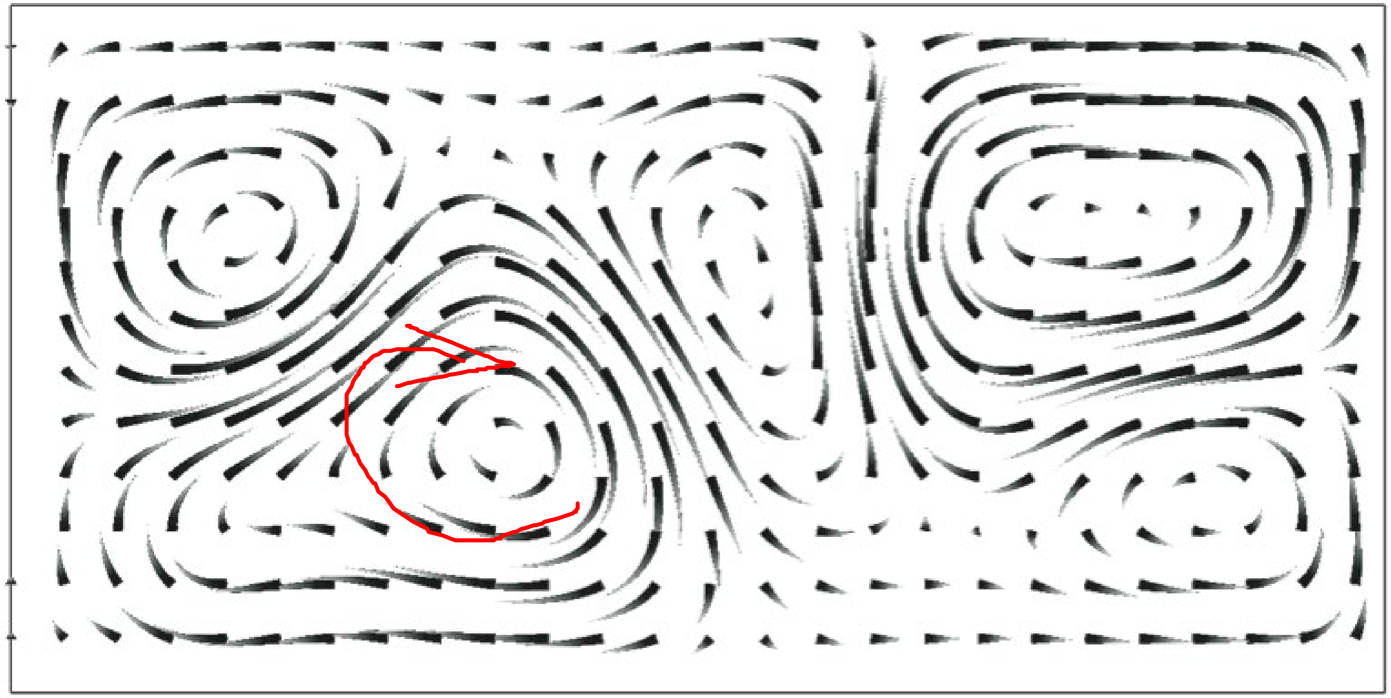


streamlines: seeds from regular grid; use un-normalized $\mathbf{v}$ for integration; color by $||\mathbf{v}||$

Stream Tubes

Can modulate tube thickness by

- data (hyperstreamlines)
- integration time – we obtain nice tapered arrows

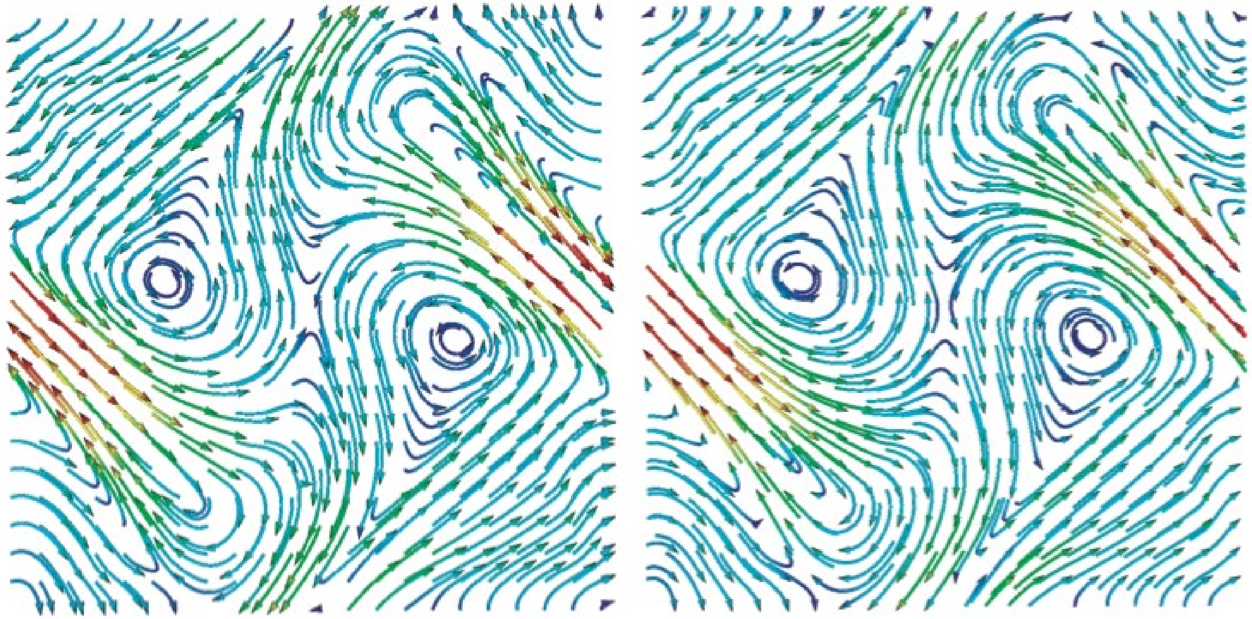


stream tubes – radius *and* opacity decrease with integration time

Stream Tubes

Like stream objects, but 3D

- compute 1D stream objects (e.g. streamlines)
- sweep (circular) cross-section along these
- visualize result with shading



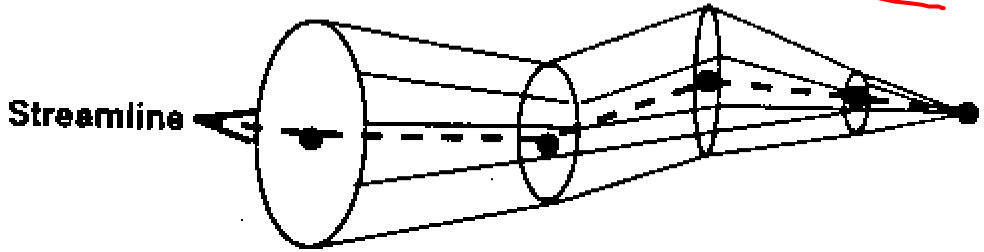
stream tubes, backward integration\

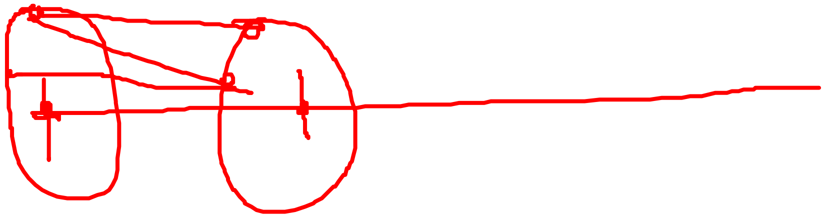
- in 2D they are a nicer option than hedgehog/glyph plots

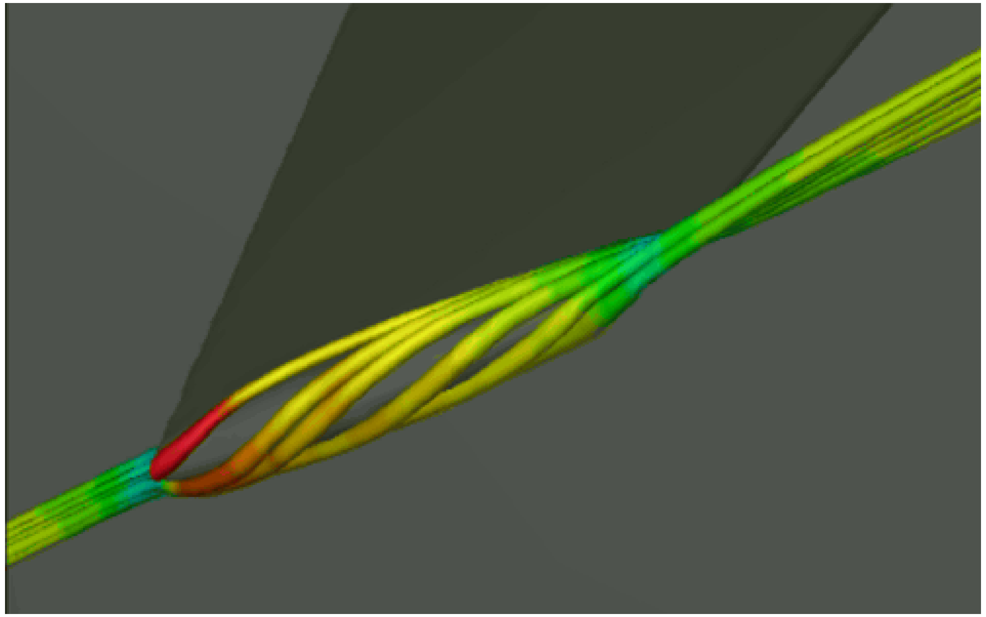
ILLINOIS

Stream Tube

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

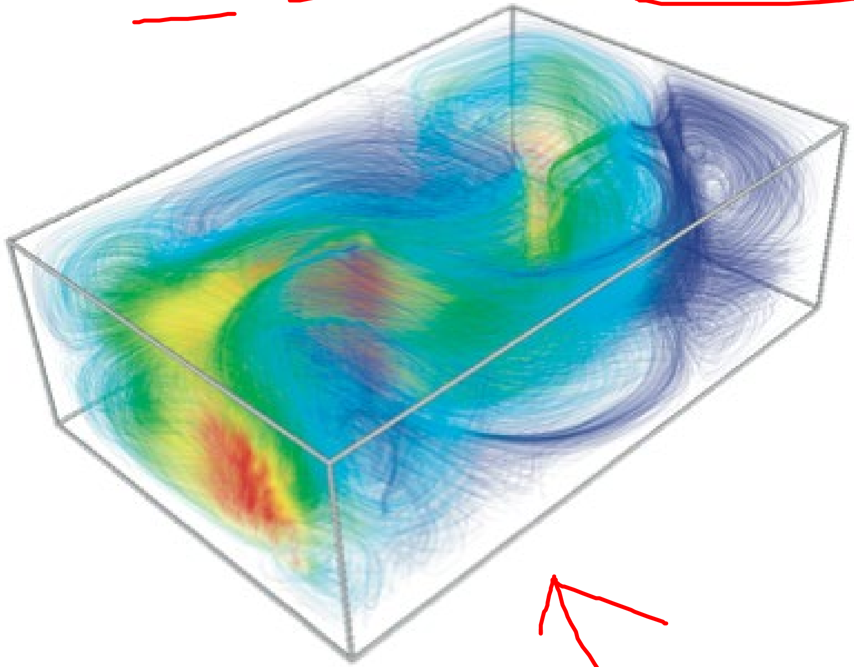




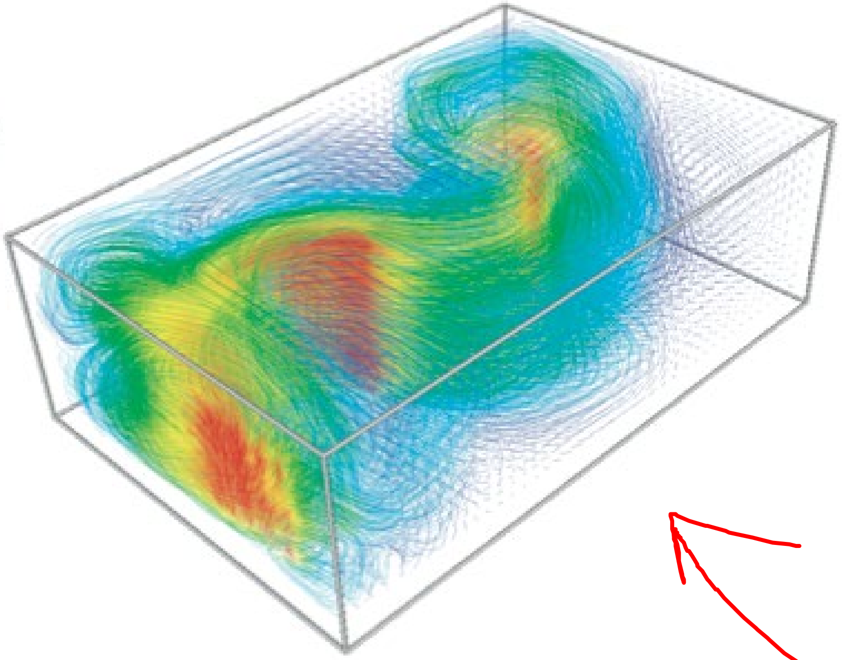


Streamlines in 3D

Can vary opacity, ~~seeding density~~, integration time

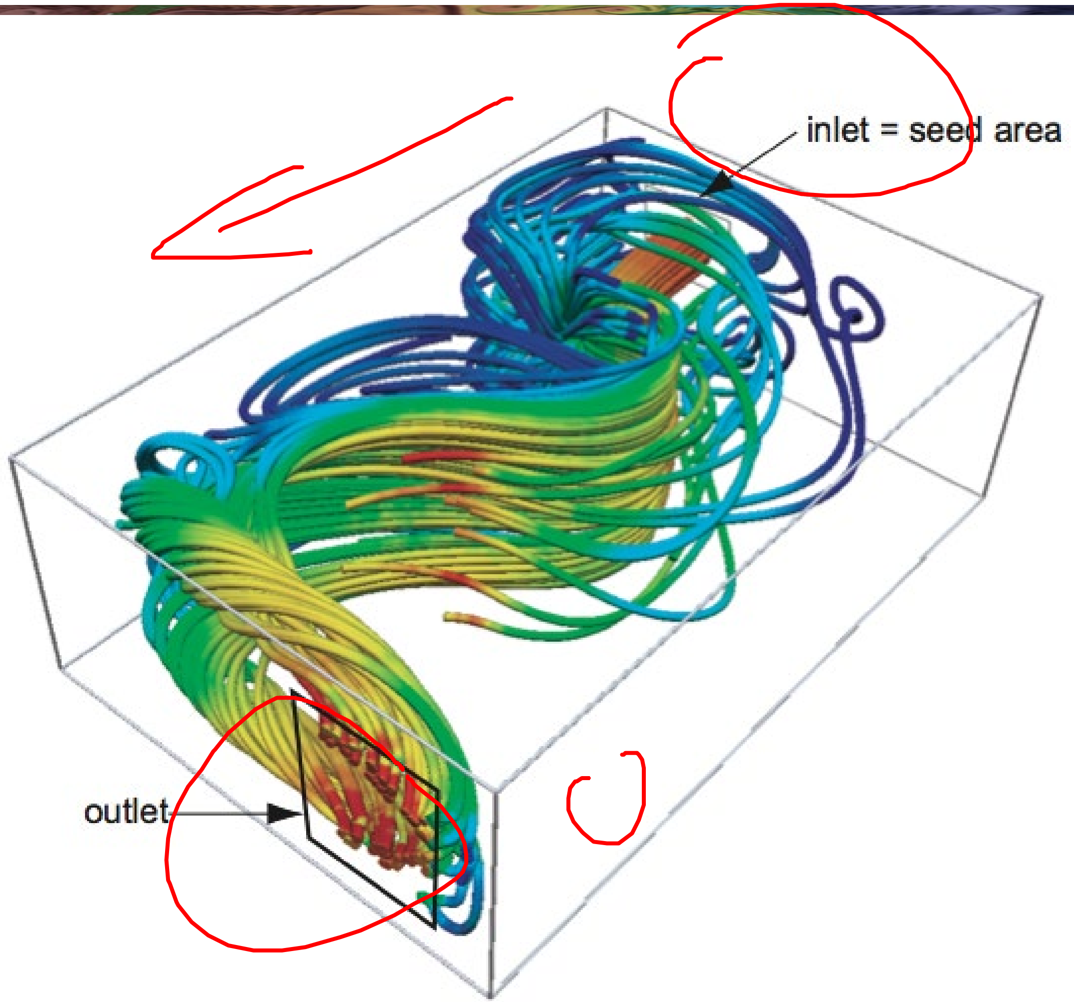


undersampling, opacity
- less occlusion
- good coverage



undersampling, shorter time
- even less occlusion
- but less continuity

Stream Tubes in 3D



stream tubes traced from inlet to outlet

- show where incoming flow arrives at
- color by flow velocity
- shade for extra occlusion cues

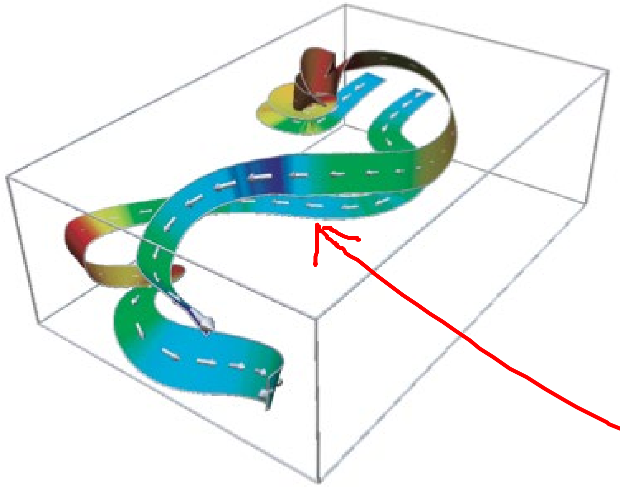
- even higher occlusion problem than for 3D streamlines
- must reduce number of seeds

ILLINOIS

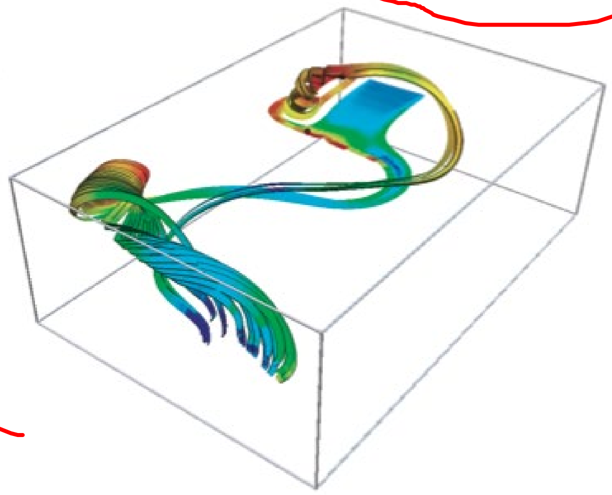
Stream Ribbons

- Visualize how the vector field 'twists' around itself as it advances in space

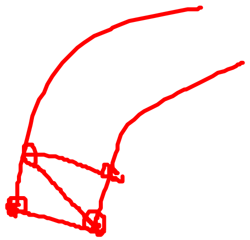
- Visualizes the *helicity* of a vector field $h = \frac{1}{2}\mathbf{v} \cdot \mathrm{rot}\,\mathbf{v}$



stream ribbons: two thick ribbons



stream ribbons: 20 thin ribbons



Algorithm

- define pairs of close seeds (p_a, p_b)
- trace streamlines S_a, S_b from (p_a, p_b)
- construct strip surface connecting closest points on S_a, S_b

ILLINOIS

Stream Objects for Unsteady Flows

- Streamline
  particle trajectory in steady (unchanging) vector field
- Pathline
  trajectory of particle in an unsteady flow
- Timeline
  connect particles released simultaneously at discrete time-steps
- Streakline
  continuously inject particles at a point, connect consecutive particles

ILLINOIS

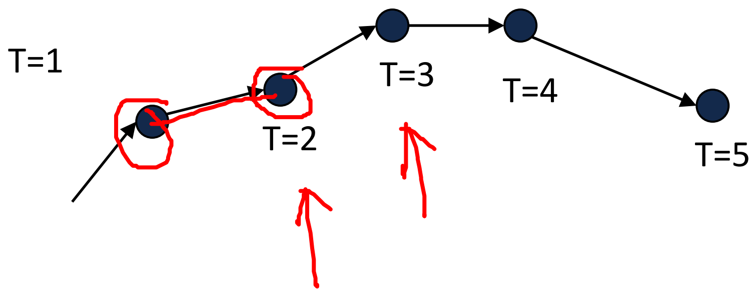
Pathlines

Extension of streamlines for time-varying data (unsteady flow)

Insert a particle into the flow

Connect positions over time

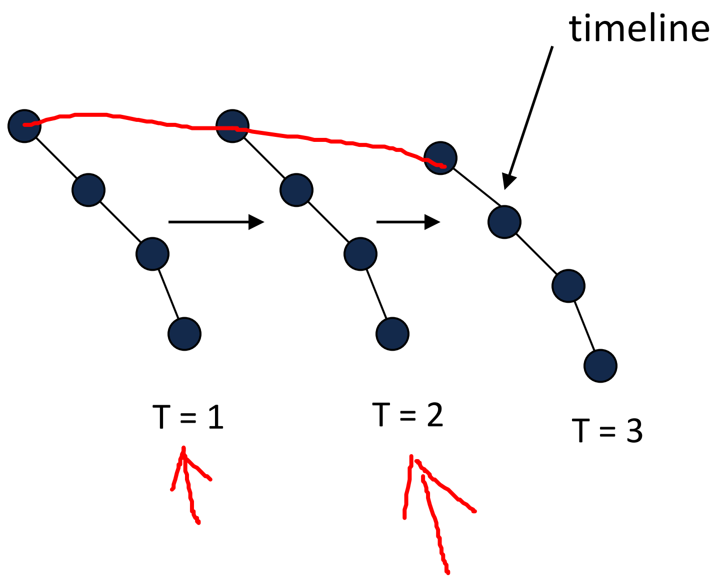
Difference from streamline is that the vector field is changing each time step



Timelines

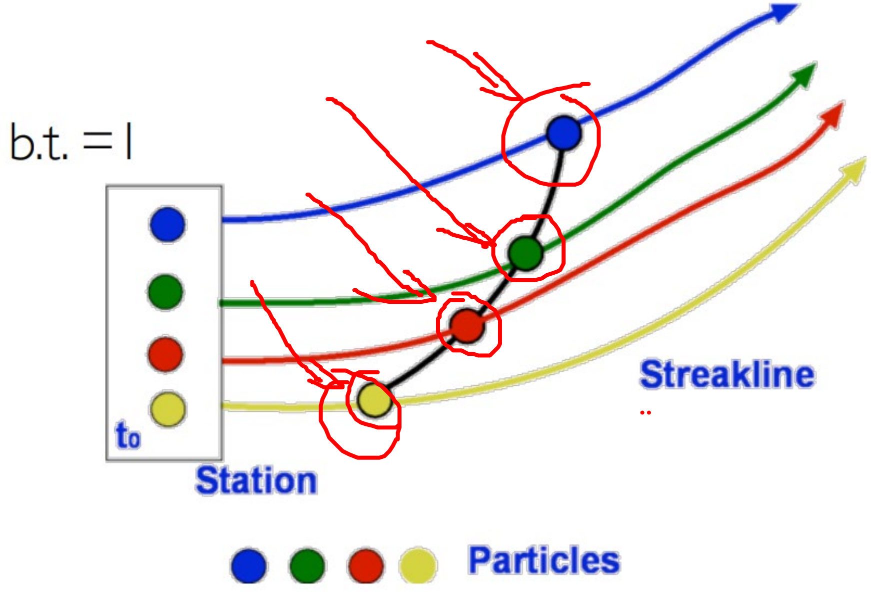
Extension of streamlines for time-varying data (unsteady flows)

Timeline draws a line through adjacent particles in flow at any instant of time

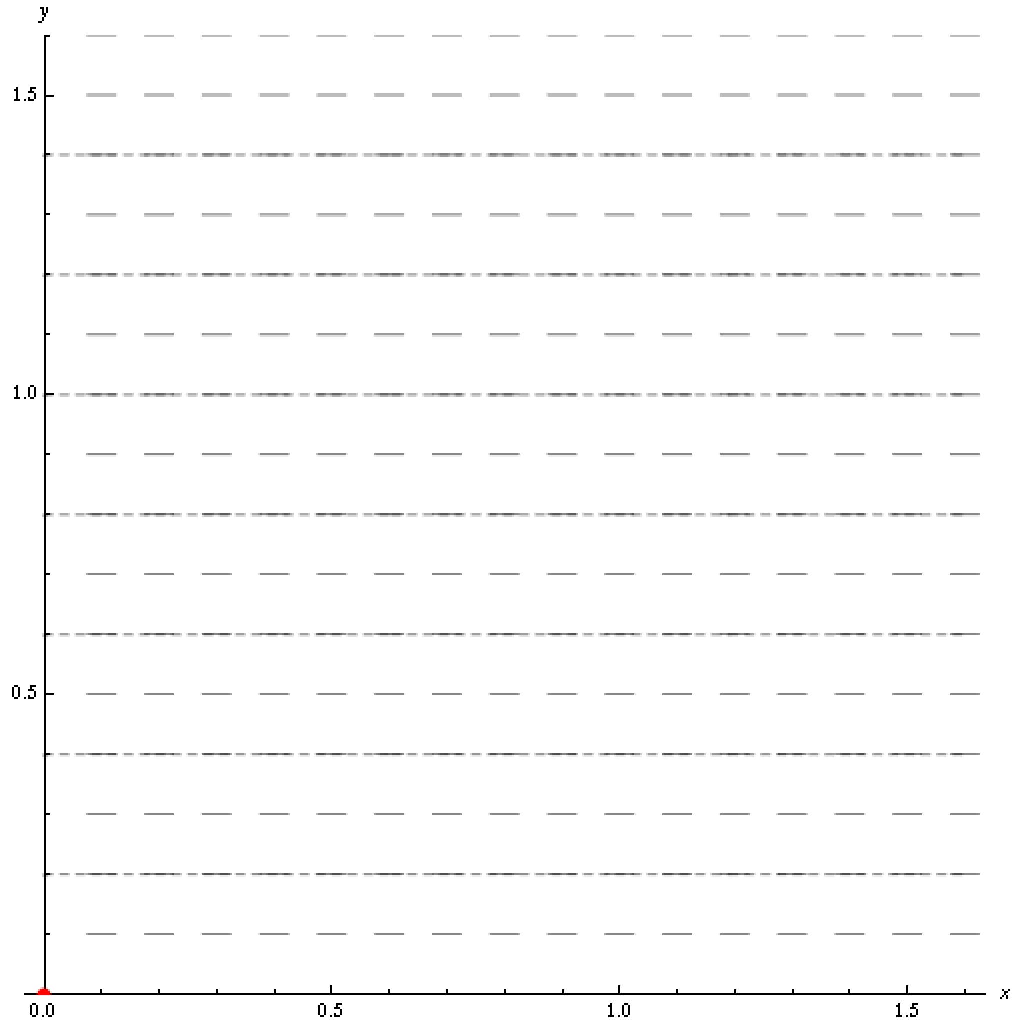


Streaklines

- For unsteady flows
- Continuously injecting a new particle at each time step
- Advecting all the existing particles and connect them together into a *streakline*
- i.e. connecting particles that have gone through a fixed point in the domain



Pathlines and Streaklines



The red particle moves in a flowing fluid; its pathline is traced in red; the tip of the trail of blue ink released from the origin follows the particle, but unlike the static pathline (which records the earlier motion of the dot), ink released after the red dot departs continues to move up with the flow. (This is a streakline.) The dashed lines represent contours of the velocity field (streamlines), showing the motion of the whole field at the same time.

- Wikipedia