Vector Visualization

3D Image Processing

Torsten Möller / Alireza Ghane

Overview

- Problem setting
- Vector calculus
- **Characteristic lines**
- Arrows and glyphs
- Particle tracing and mapping methods
- Particle tracing on grids
- **Line integral convolution**
- Texture advection
- **Topology-based visualization**
- 3D vector fields

Readings

- "The Visualization Handbook":
  - Chapter 12 (Overview of Flow Visualization)
  - Chapter 13 (Flow Textures)
  - Chapter 17 (Topological Methods for Flow Visualization)
- "Scientific Visualization":
  - Chapter 14 (Particle Tracing Algorithms for 3D Curvilinear Grids)

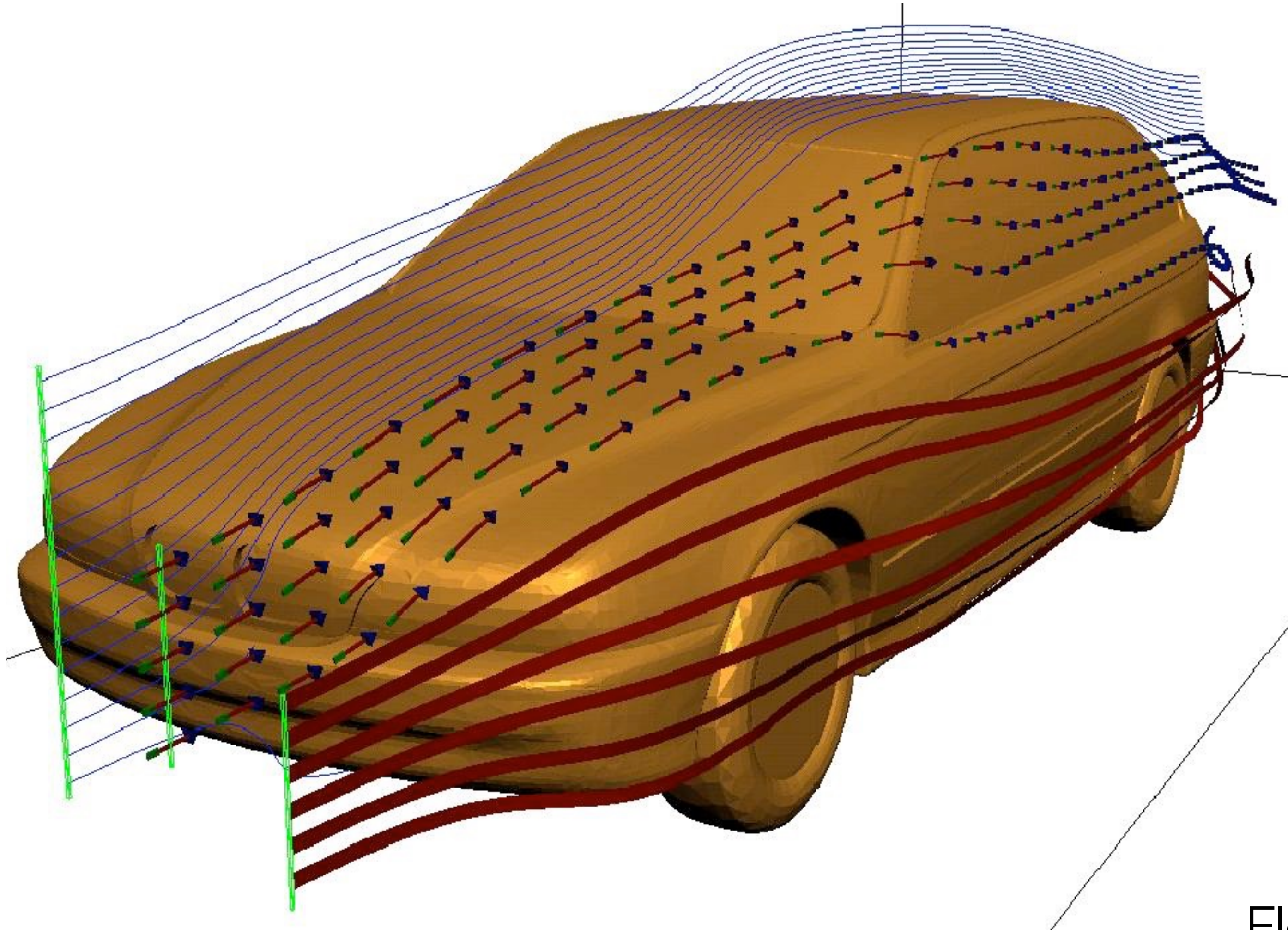
Problem Setting

- Vector data set
- Represent direction and magnitude
- Given by an n-tupel $(f_1,\ldots,f_n)$ with $f_k=f_k(x_1,\ldots,x_n)$, $n \geq 2$ and $1 \leq k \leq n$
- Specific transformation properties
- Typically $n = 2$ or $n = 3$

Problem Setting

- Main application of vector field visualization is flow visualization
  - Motion of fluids (gas, liquids)
  - Geometric boundary conditions
  - Velocity (flow) field $\boldsymbol{v}(\boldsymbol{x},t)$
  - Pressure p
  - Temperature T
  - Vorticity $\nabla\times\boldsymbol{v}$
  - Density ρ
  - Conservation of mass, energy, and momentum
  - Navier-Stokes equations
  - CFD (Computational Fluid Dynamics)

Problem Setting



Flow visualization
based on CFD data

Problem Setting

- Flow visualization – classification
  - Dimension (2D or 3D)
  - Time-dependency: stationary (steady) vs. instationary (unsteady)
  - Grid type
  - Compressible vs. incompressible fluids
- In most cases numerical methods required for flow visualization

Vector Calculus

- Review of basics of vector calculus
- Deals with vector fields and various kinds of derivatives
- Flat (Cartesian) manifolds only
- Cartesian coordinates only
- 3D only

Vector Calculus

- Scalar function $\rho(\boldsymbol{x}, t)$
- Gradient

$$\nabla \rho(\boldsymbol{x}, t) = \begin{pmatrix} \frac{\partial}{\partial x} \rho(\boldsymbol{x}, t) \\ \frac{\partial}{\partial y} \rho(\boldsymbol{x}, t) \\ \frac{\partial}{\partial z} \rho(\boldsymbol{x}, t) \end{pmatrix} = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} \rho(\boldsymbol{x}, t)$$

- Gradient points into direction of maximum change of $\rho(\boldsymbol{x}, t)$
- Laplace $\Delta \rho(\boldsymbol{x}, t) = \nabla \cdot \nabla \rho(\boldsymbol{x}, t)$

$$= \frac{\partial^2}{\partial x^2} \rho(\boldsymbol{x}, t) + \frac{\partial^2}{\partial y^2} \rho(\boldsymbol{x}, t) + \frac{\partial^2}{\partial z^2} \rho(\boldsymbol{x}, t)$$

Vector Calculus

- Vector function ***v(x,t)***
- Jacobi matrix (“Gradient tensor”)

$$\boldsymbol{J} = \nabla \overrightarrow{v}(\boldsymbol{x}, t) = \begin{pmatrix} \frac{\partial}{\partial x} v_x & \frac{\partial}{\partial y} v_x & \frac{\partial}{\partial z} v_x \\ \frac{\partial}{\partial x} v_y & \frac{\partial}{\partial y} v_y & \frac{\partial}{\partial z} v_y \\ \frac{\partial}{\partial x} v_z & \frac{\partial}{\partial y} v_z & \frac{\partial}{\partial z} v_z \end{pmatrix}$$

- Divergence

$$\mathrm{div}\, \overrightarrow{v}(\boldsymbol{x}, t) = \nabla \times \overrightarrow{v}(\boldsymbol{x}, t) = \frac{\partial}{\partial x} v_x + \frac{\partial}{\partial y} v_y + \frac{\partial}{\partial z} v_z$$

Characteristic Lines

- Types of characteristic lines in a vector field:
  - **Streamlines**: tangential to the vector field
  - **Pathlines**: trajectories of massless particles in the flow
  - **Streaklines**: trace of dye that is released into the flow at a fixed position
  - **Time lines** (time surfaces): propagation of a line (surface) of massless elements in time

Characteristic Lines

- Streamlines
  - Tangential to the vector field
  - Vector field at an arbitrary, yet fixed time *t*
  - Streamline is a solution to the initial value problem of an ordinary differential equation:

$$\vec{L}(0) = \vec{x}_0 \quad \frac{d\vec{L}(u)}{du} = \vec{v}(\vec{L}(u), t)$$

initial value
(seed point $\mathbf{x}_0$)

ordinary differential equation

  - Streamline is curve **L**(*u*) with the parameter *u*

Video

- IntroParticles2D
- IntroParticles3D
- IntroStreamlines

Characteristic Lines

- Pathlines
  - Trajectories of massless particles in the flow
  - Vector field can be time-dependent (unsteady)
  - Pathline is a solution to the initial value problem of an ordinary differential equation:

$$\vec{L}(0) = \vec{x}_0 \qquad \frac{d\vec{L}(u)}{du} = \vec{v}(\vec{L}(u), u)$$

Video

- IntroPathlines

Characteristic Lines

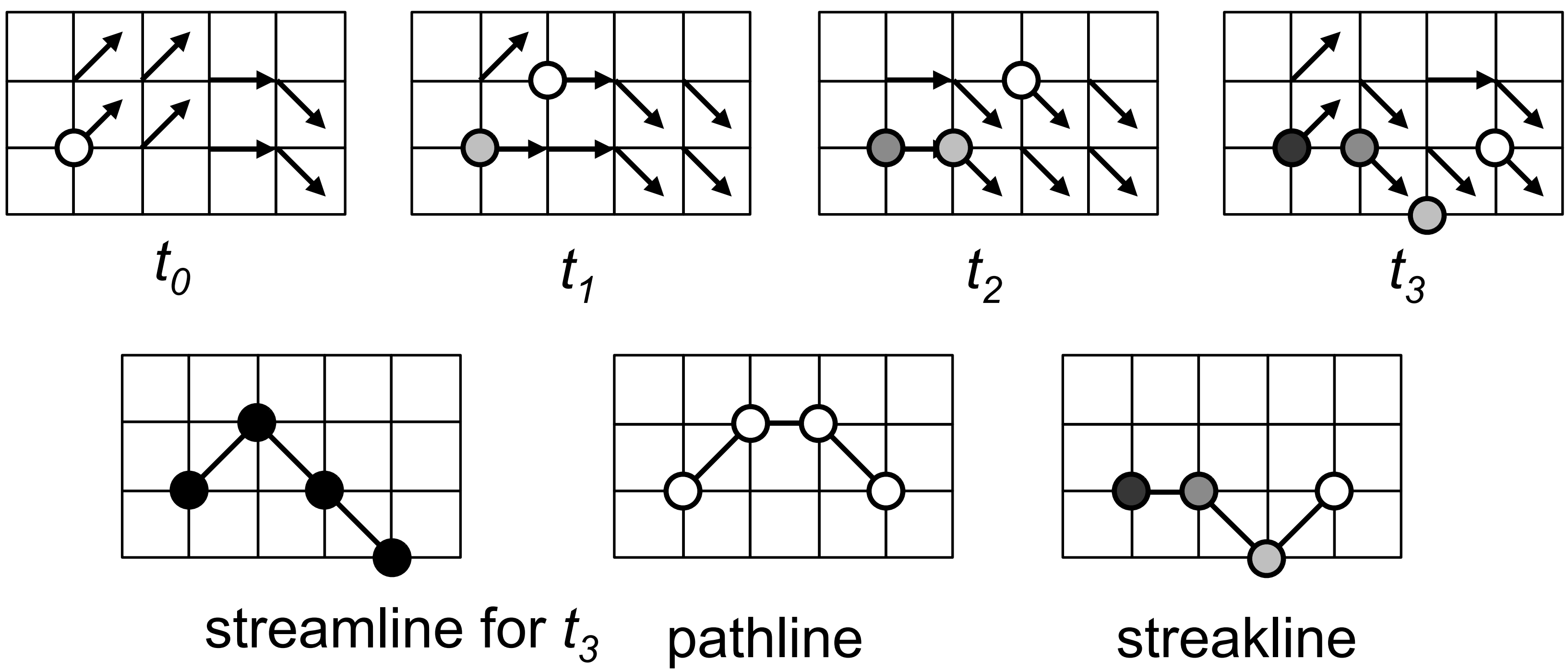
- Streaklines
  - Trace of dye that is released into the flow at a fixed position
  - Connect all particles that passed through a certain position
- Time lines (time surfaces)
  - Propagation of a line (surface) of massless elements in time
  - Idea: “consists” of many point-like particles that are traced
  - Connect particles that were released simultaneously

Video

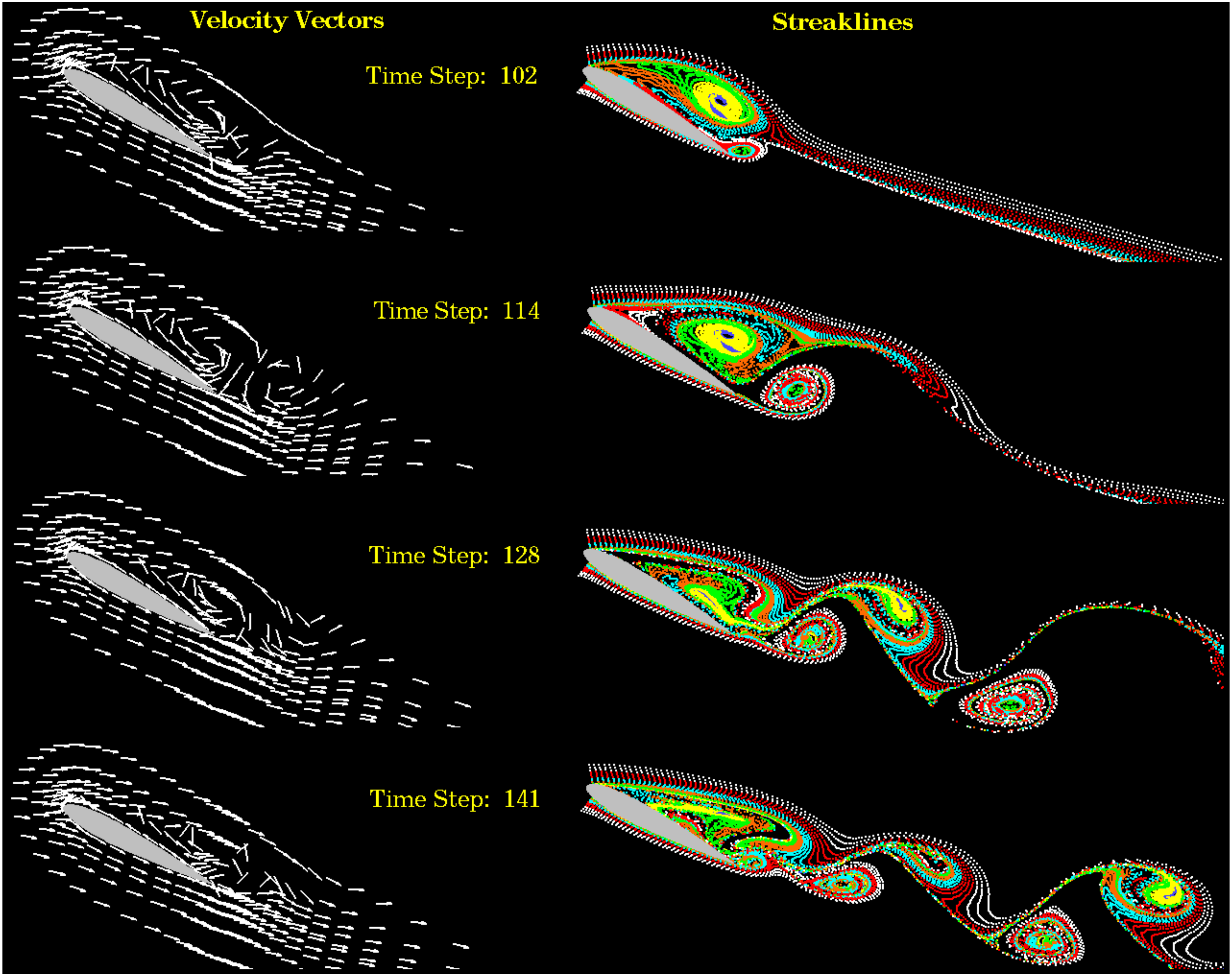
- IntroCylinderStreak
- CylinderStreakOverTau

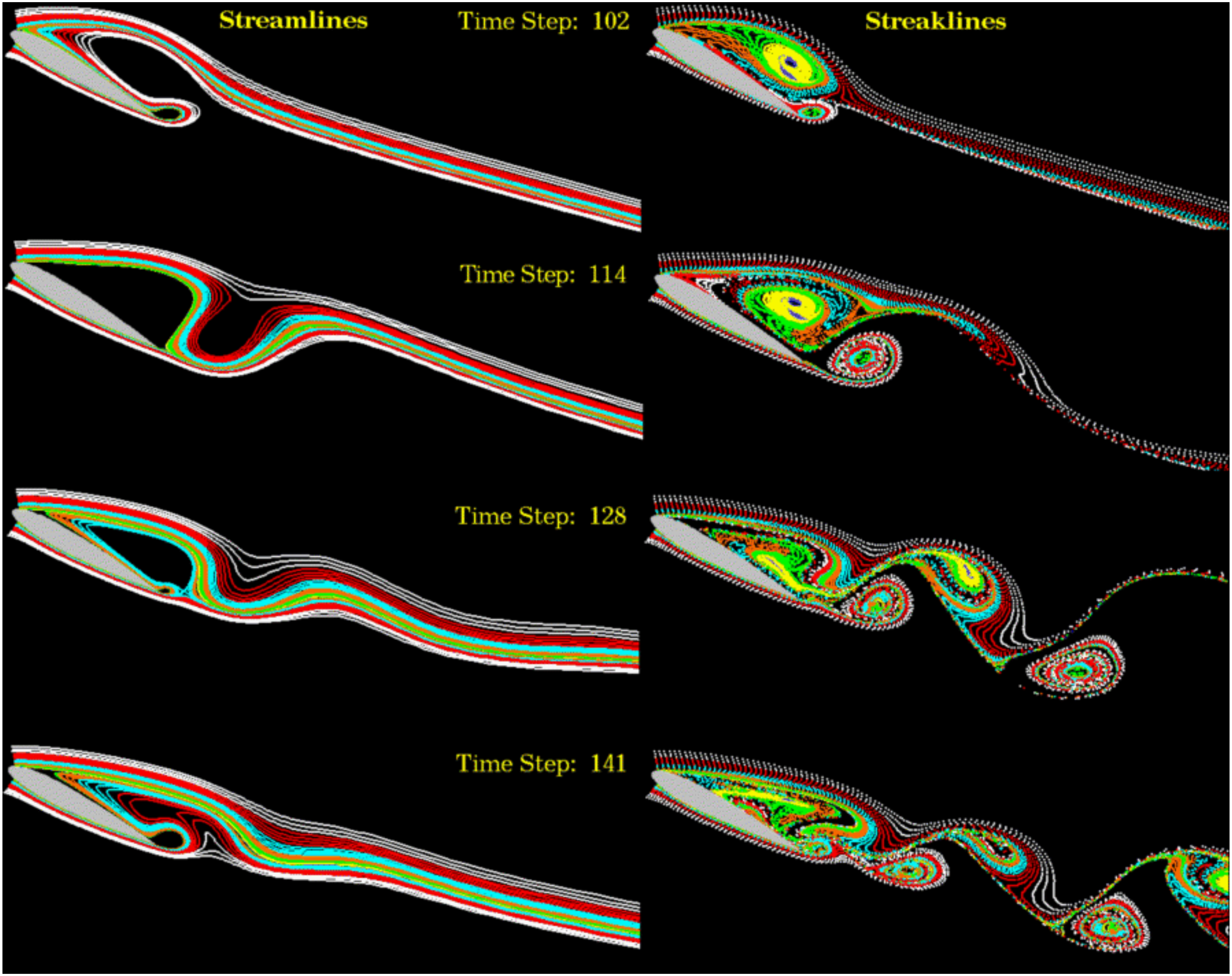
Characteristic Lines

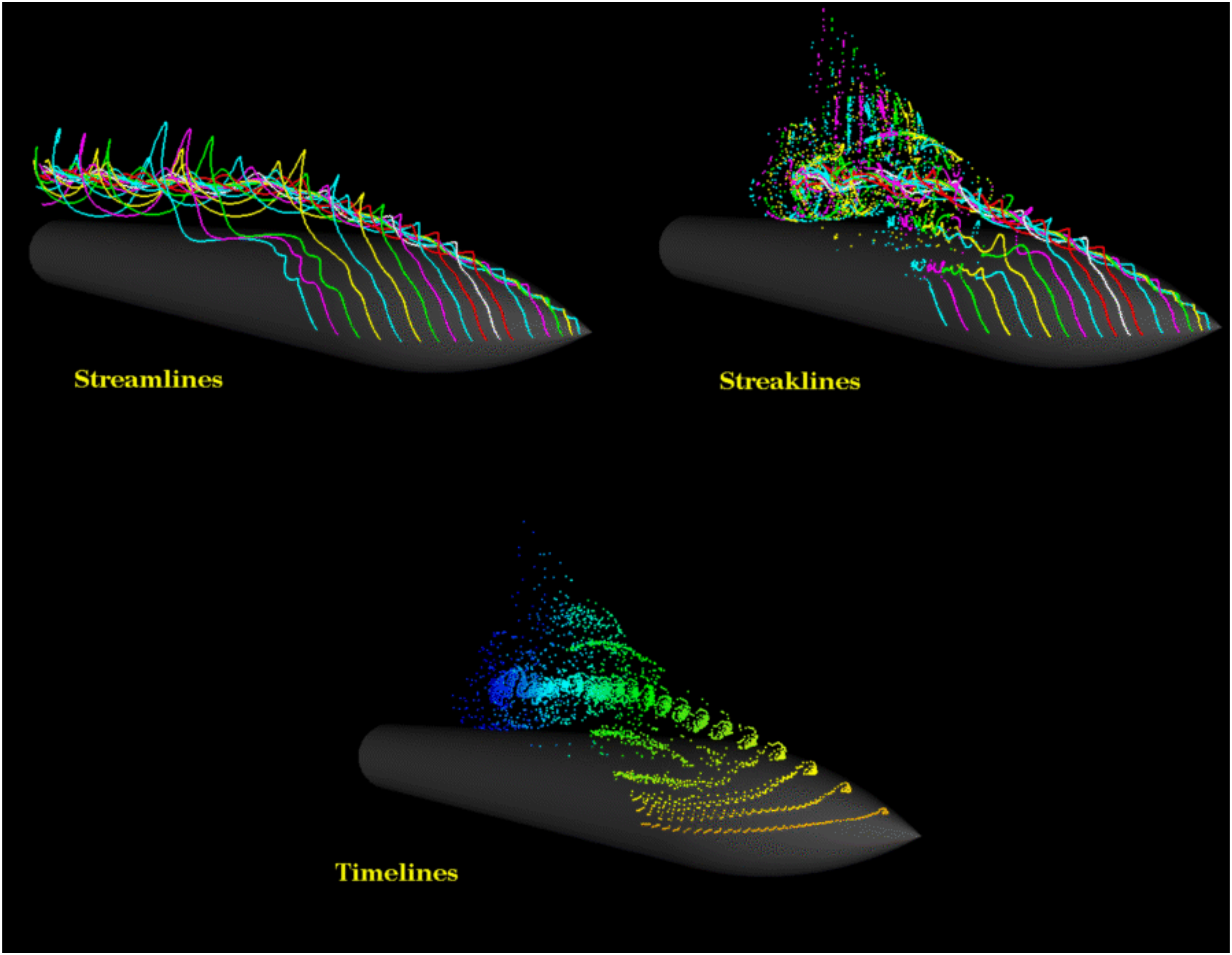
- Comparison of pathlines, streaklines, and streamlines

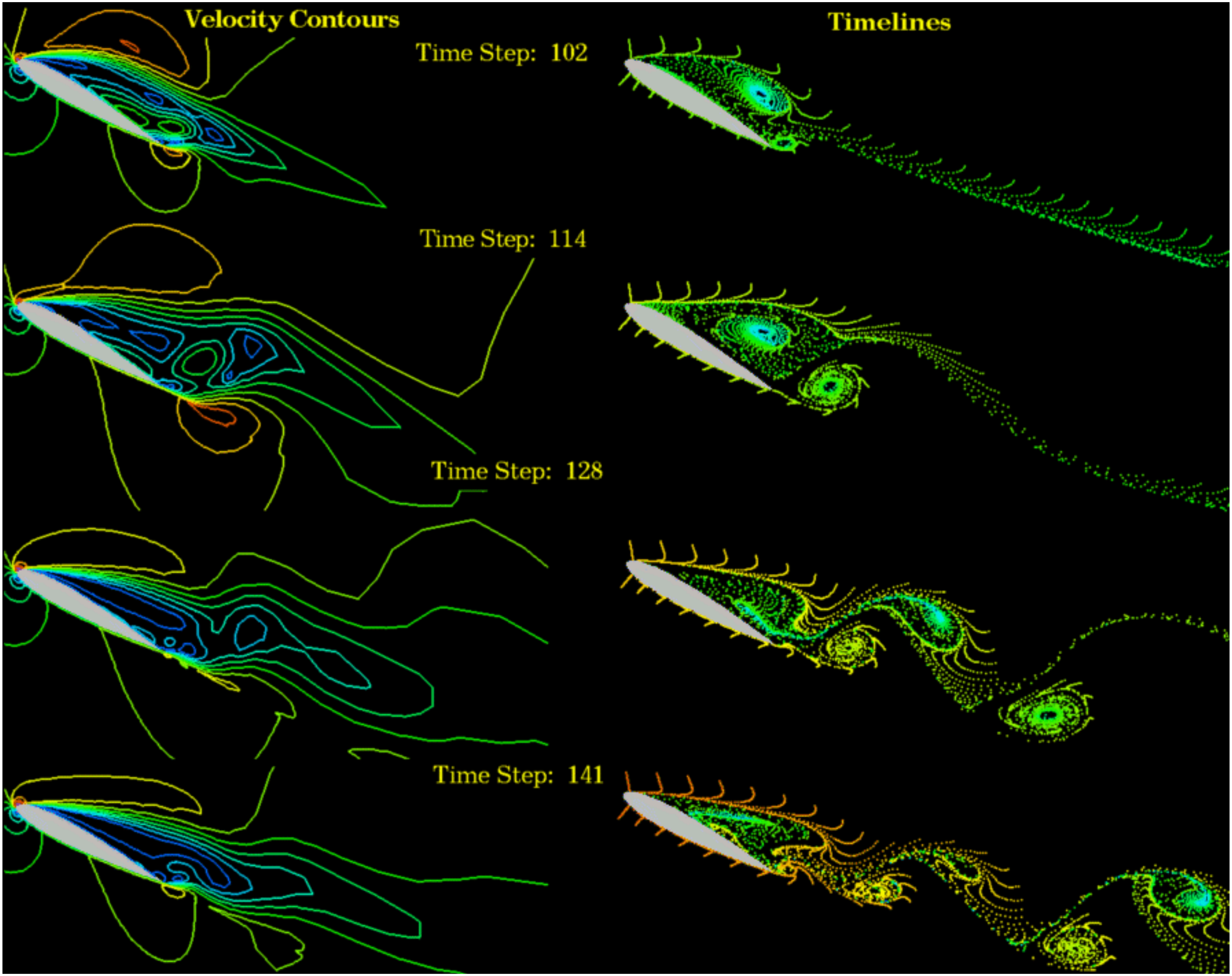


- Pathlines, streaklines, and streamlines are identical for steady flows


Velocity Vectors
Streaklines
Time Step: 102
Time Step: 114
Time Step: 128
Time Step: 141


Streamlines
Time Step: 102
Streaklines
Time Step: 114
Time Step: 128
Time Step: 141


Streamlines
Streaklines
Timelines


Velocity Contours
Timelines
Time Step: 102
Time Step: 114
Time Step: 128
Time Step: 141

Overview

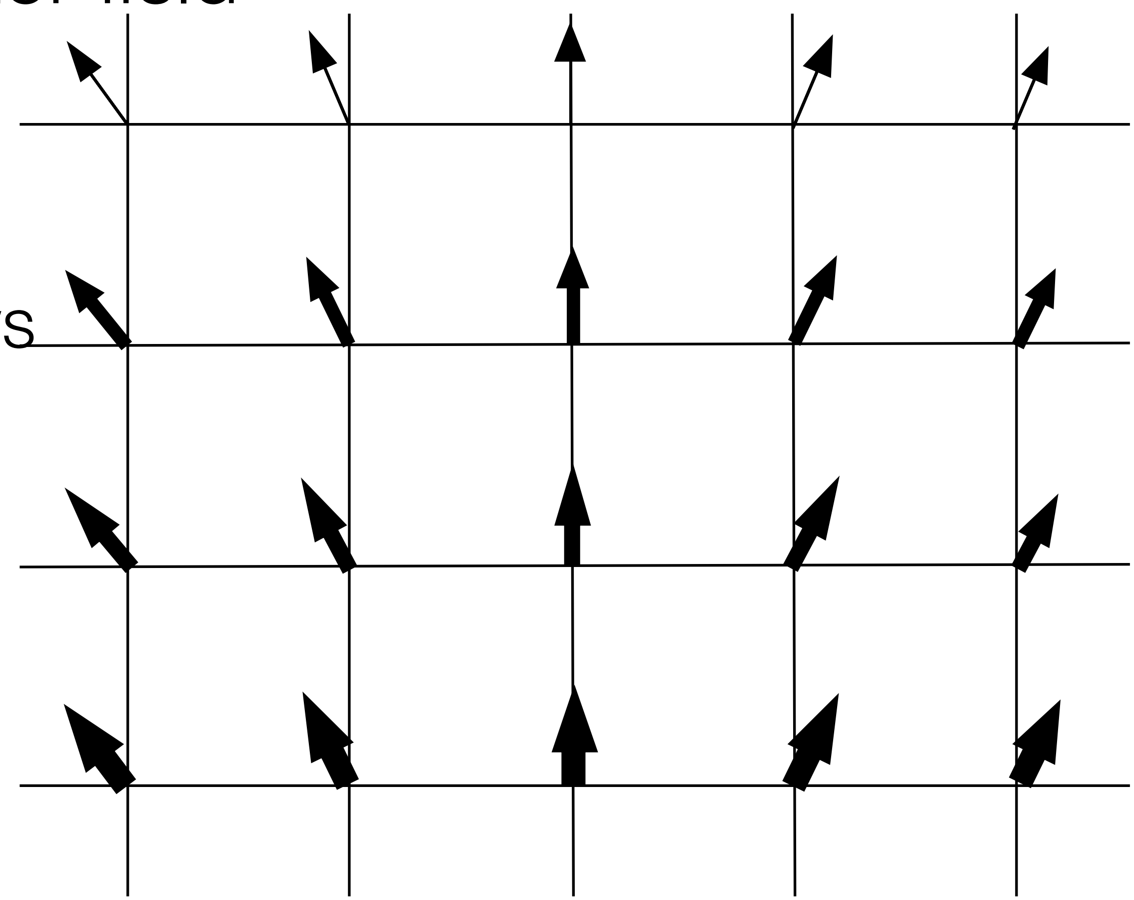
- Problem setting
- Vector calculus
- Characteristic lines
- Arrows and glyphs
- Particle tracing and mapping methods
- Particle tracing on grids
- **Line integral convolution**
- Texture advection
- **Topology-based visualization**
- 3D vector fields

Arrows and Glyphs

- Visualize **local** features of the vector field:
  - Vector itself
  - Vorticity
  - Extern data: temperature, pressure, etc.
- Important elements of a vector:
  - Direction
  - Magnitude
  - Not: components of a vector
- Approaches:
  - Arrow plots
  - Glyphs
- Direct mapping

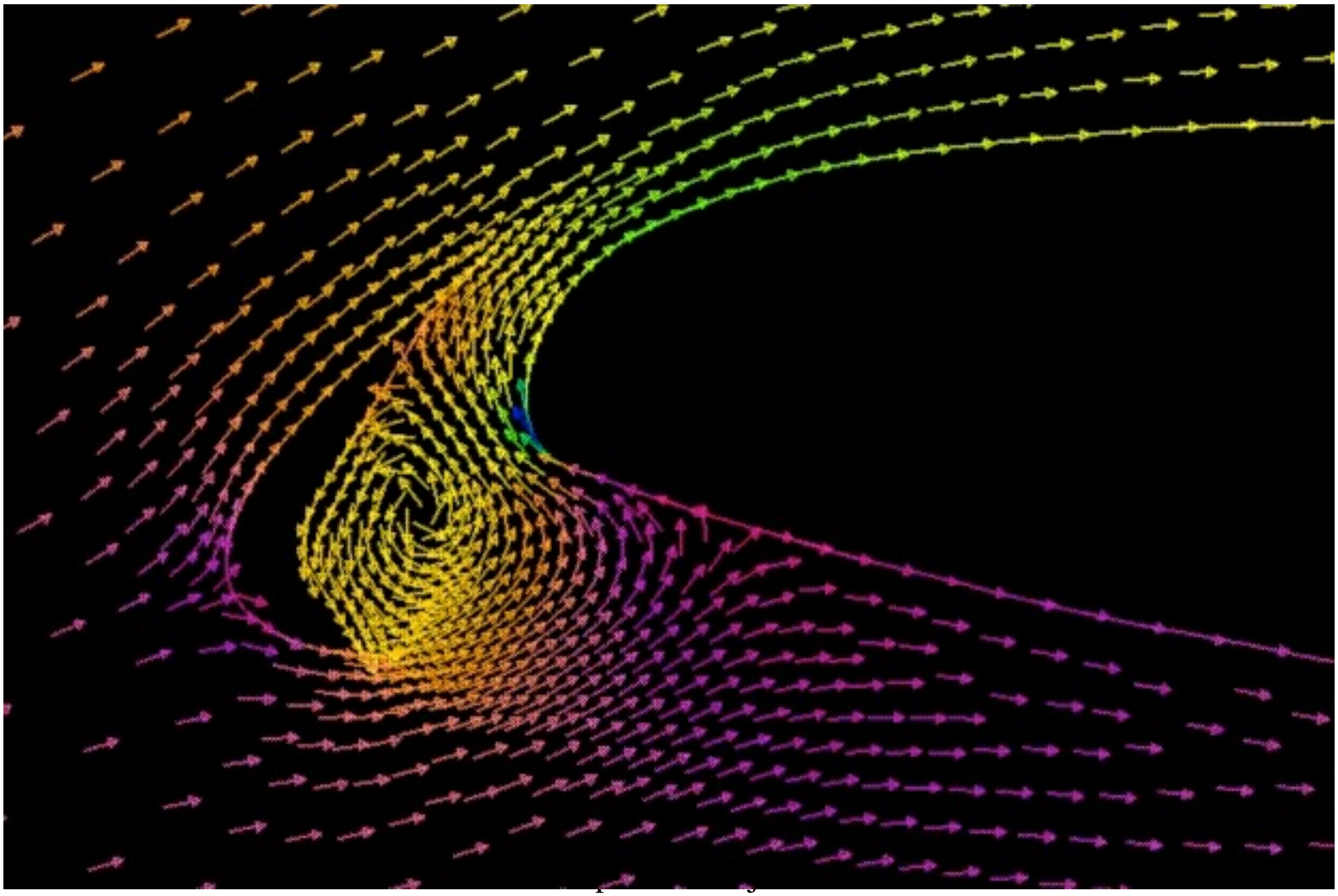
Arrows and Glyphs

- Arrows visualize
  - Direction of vector field
  - Orientation
  - Magnitude:
    - Length of arrows
    - Color coding



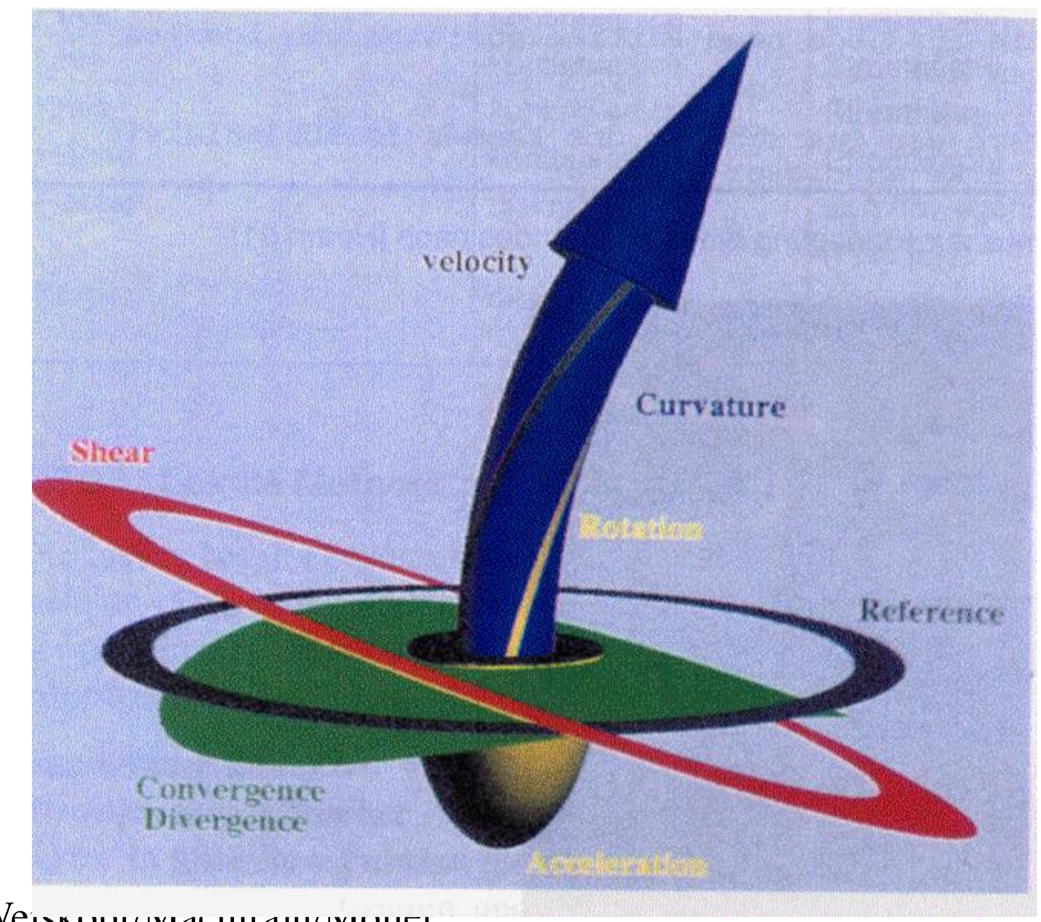
Arrows and Glyphs

- Arrows



Arrows and Glyphs

- Glyphs
  - Can visualize more features of the vector field (flow field)



Arrows and Glyphs

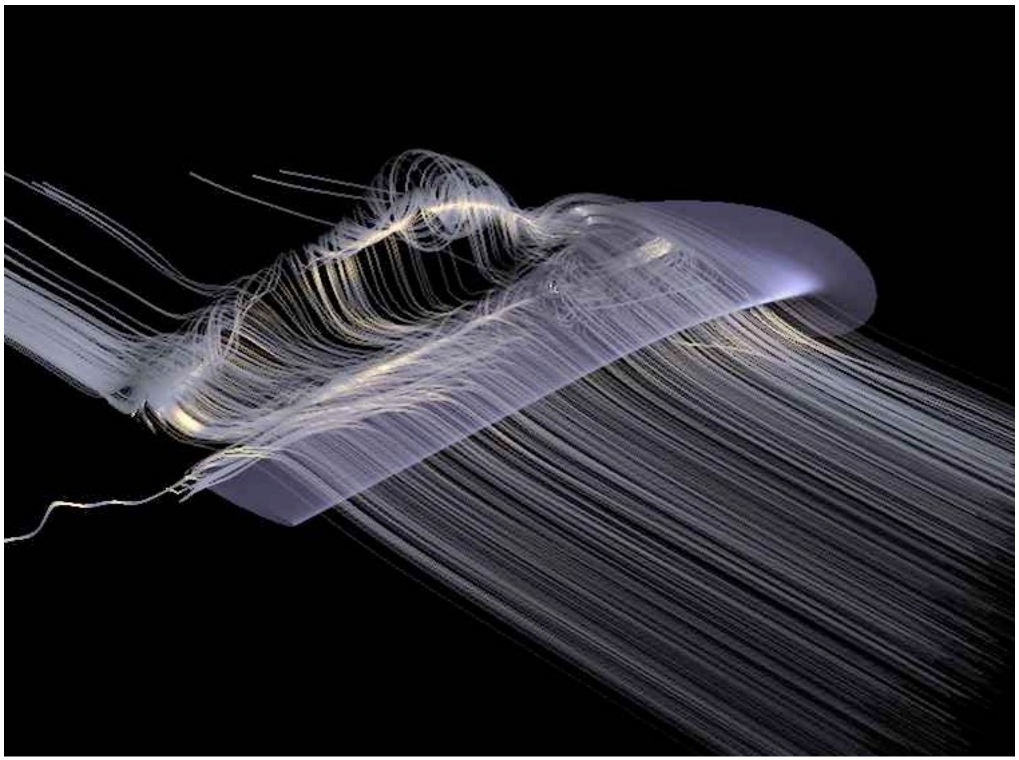
- Advantages and disadvantages of glyphs and arrows:
  - + Simple
  - + 3D effects
  - - Inherent occlusion effects
  - - Poor results if magnitude of velocity changes rapidly (Use arrows of constant length and color code magnitude)

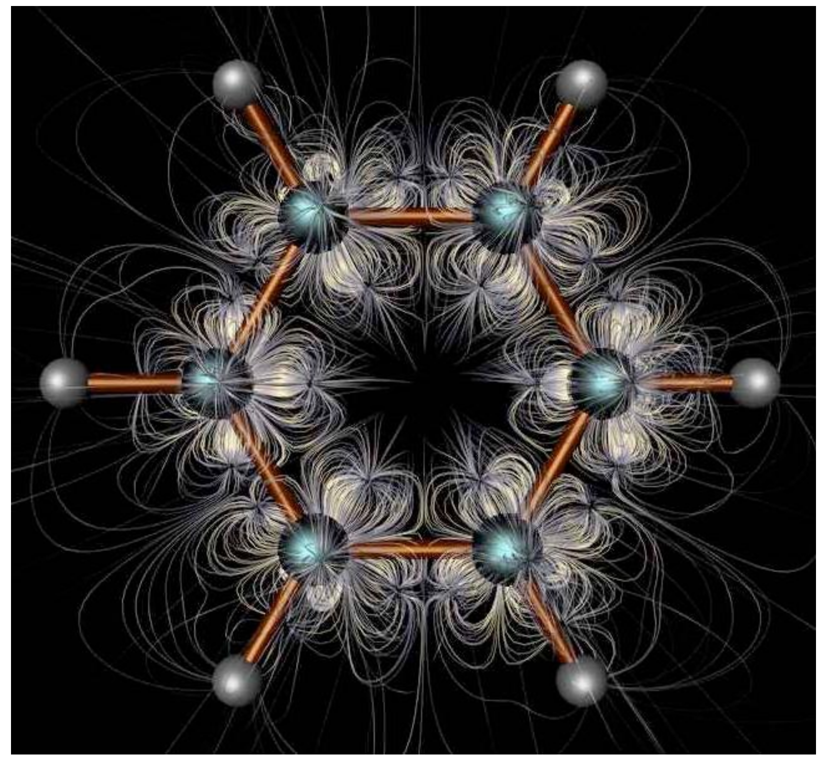
Mapping Methods Based on Particle Tracing

- Basic idea: trace particles
- Characteristic lines
- Mapping approaches:
  - Lines
  - Surfaces
  - Individual particles
  - Texture
  - Sometimes animated
- Density of visual representation
  - Sparse = only a few visual patterns (e.g. only a few streamlines)
  - Dense = complete coverage of the domain by visual structures

Mapping Methods Based on Particle Tracing

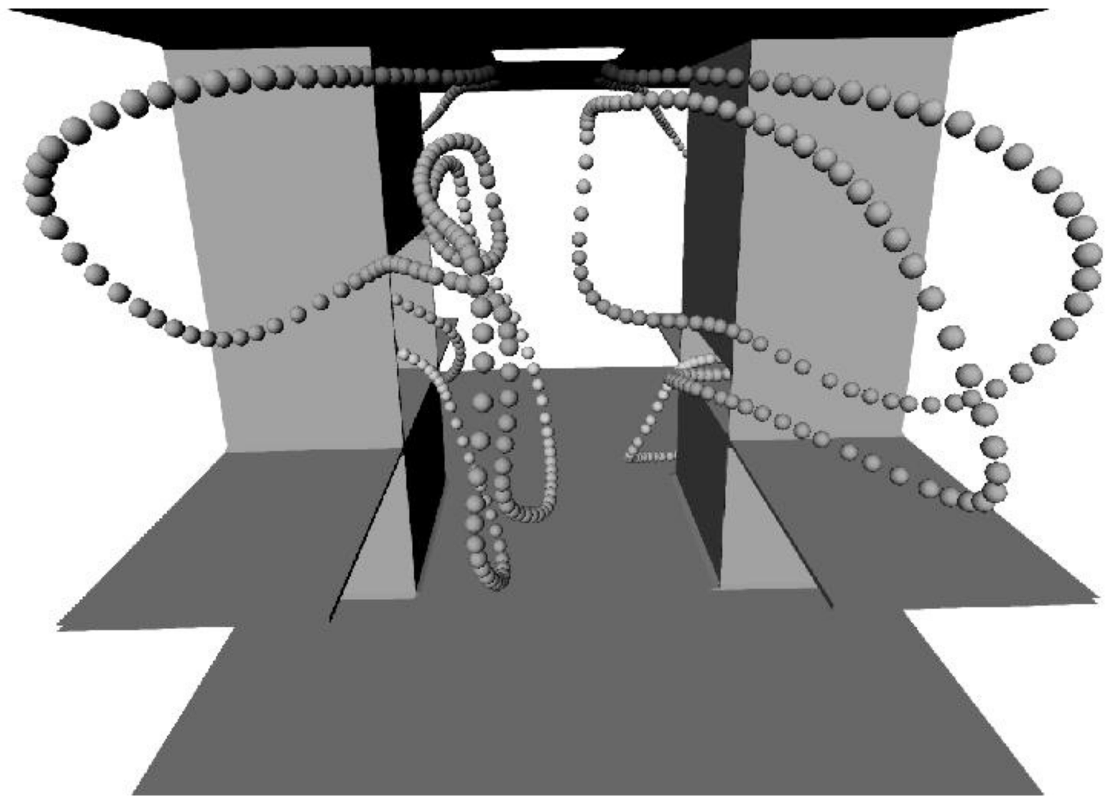
- Pathlines





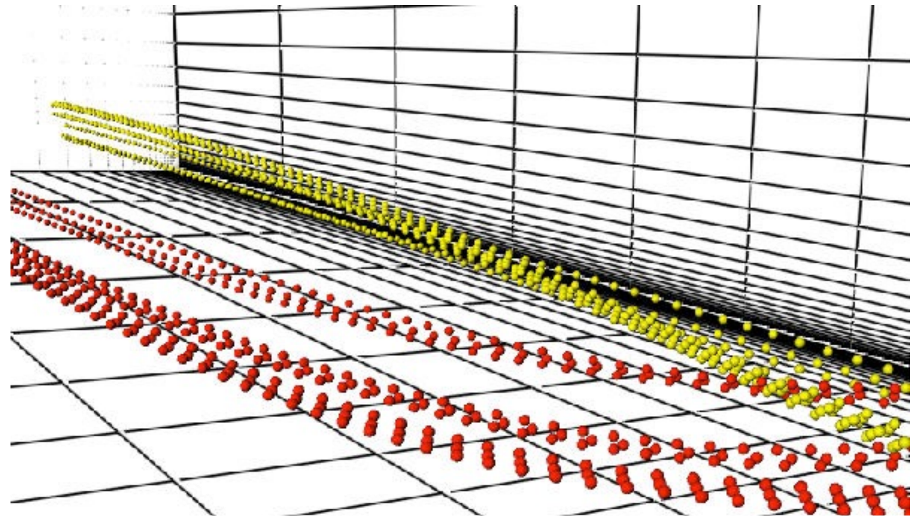
Mapping Methods Based on Particle Tracing

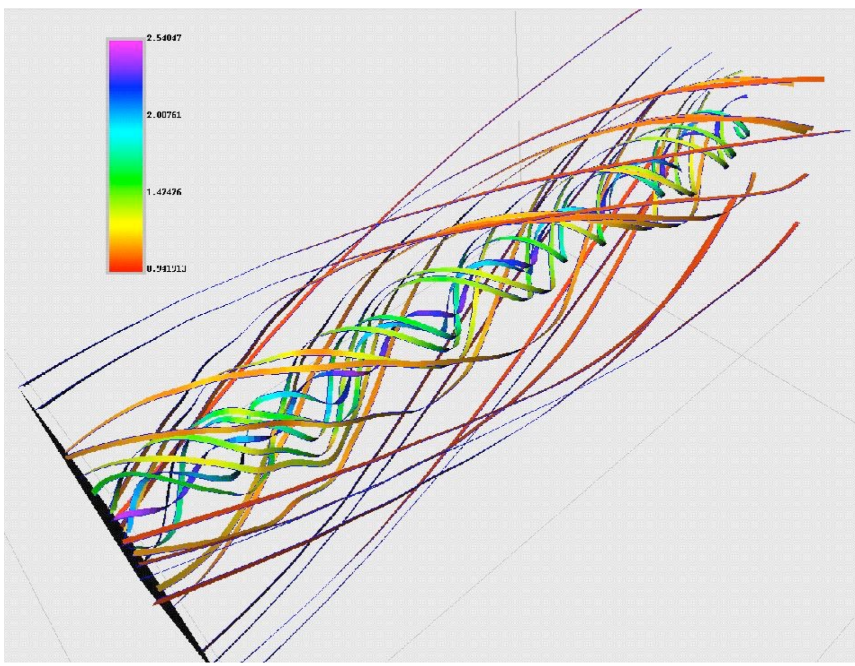
- Stream balls
  - Encode additional scalar value by radius

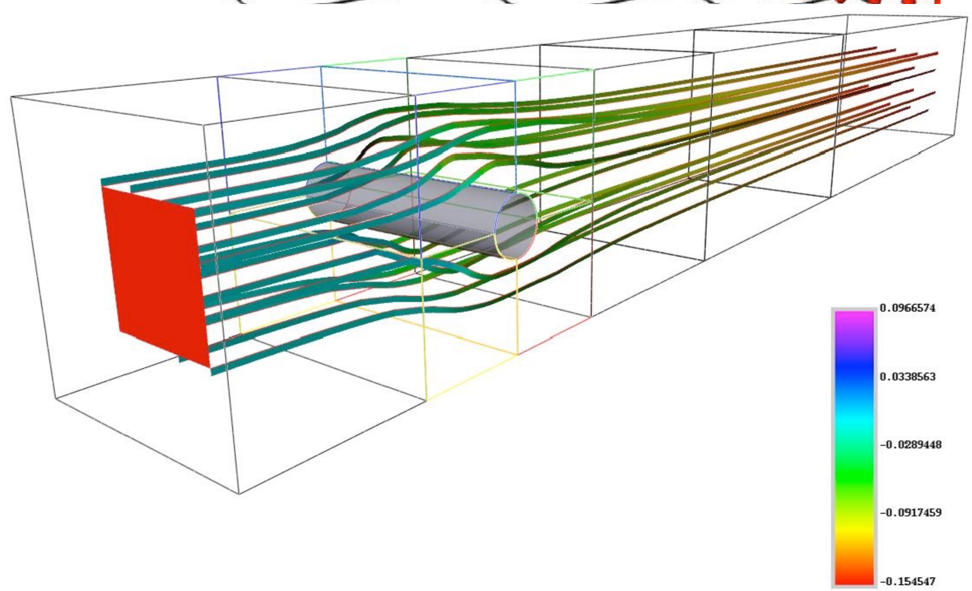


Mapping Methods Based on Particle Tracing

- Streaklines

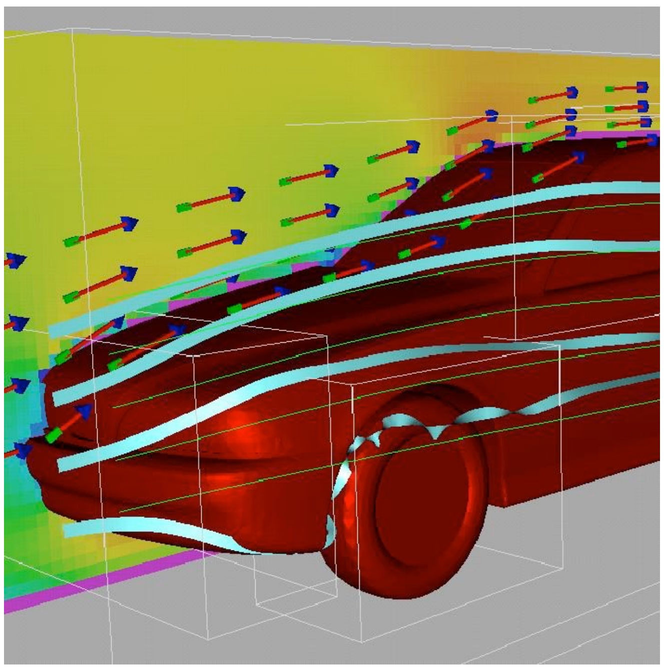


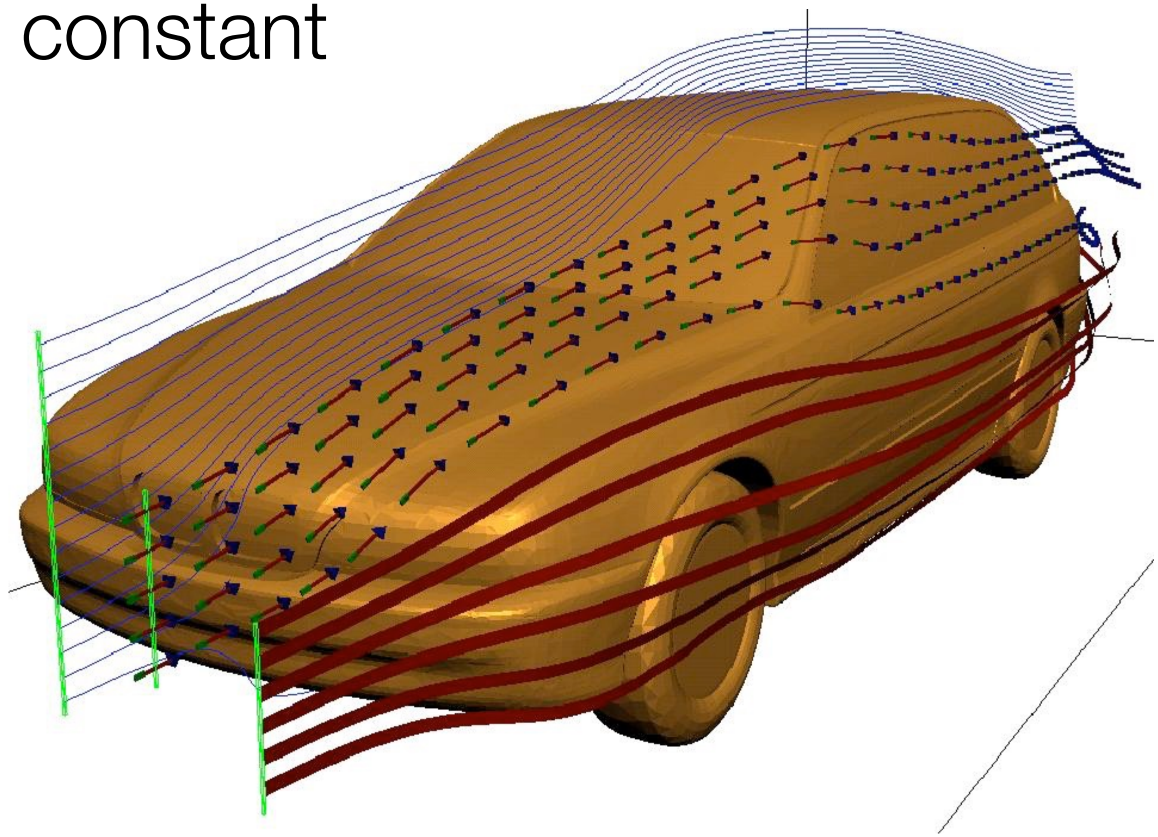




Mapping Methods Based on Particle Tracing

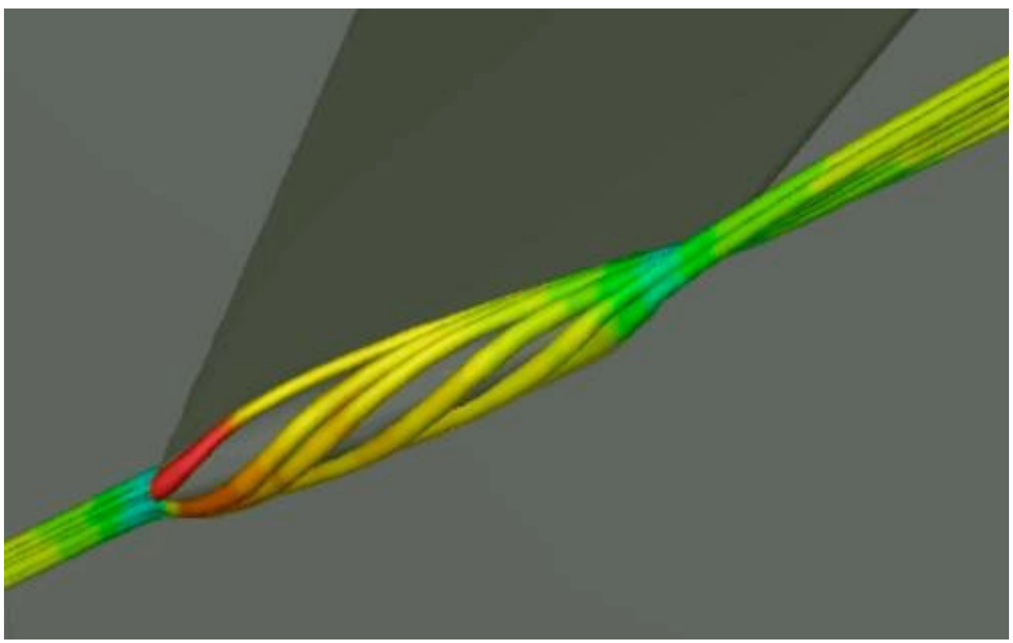
- Stream ribbons
  - Trace two close-by particles
  - Keep distance constant

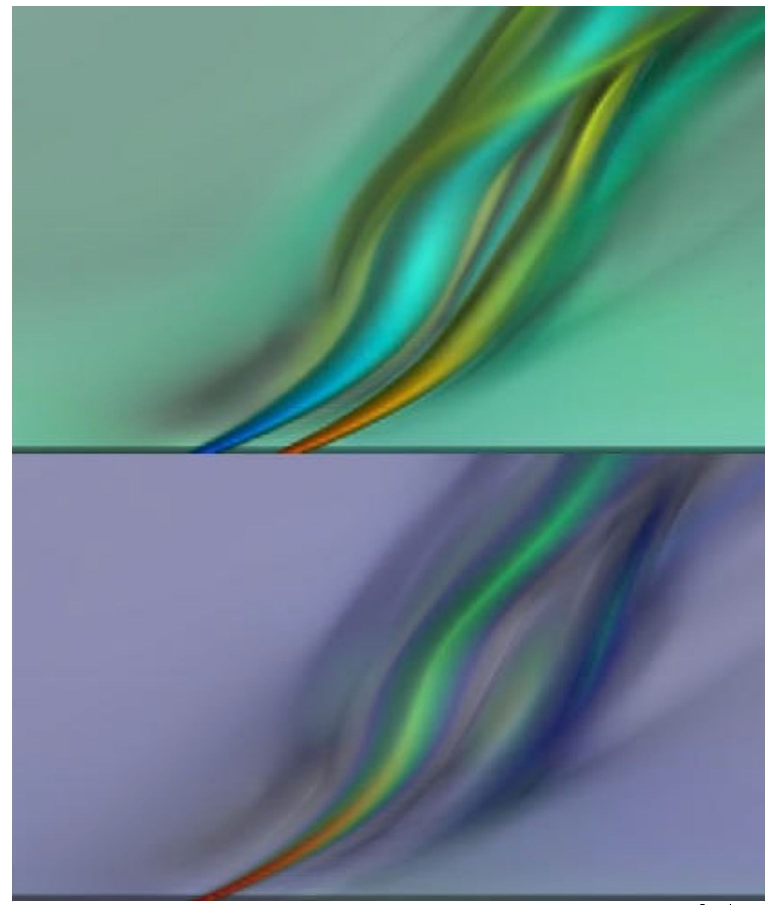




Mapping Methods Based on Particle Tracing

- Stream tubes
  - Specify contour, e.g. triangle or circle, and trace it through the flow





Mapping Methods Based on Particle Tracing

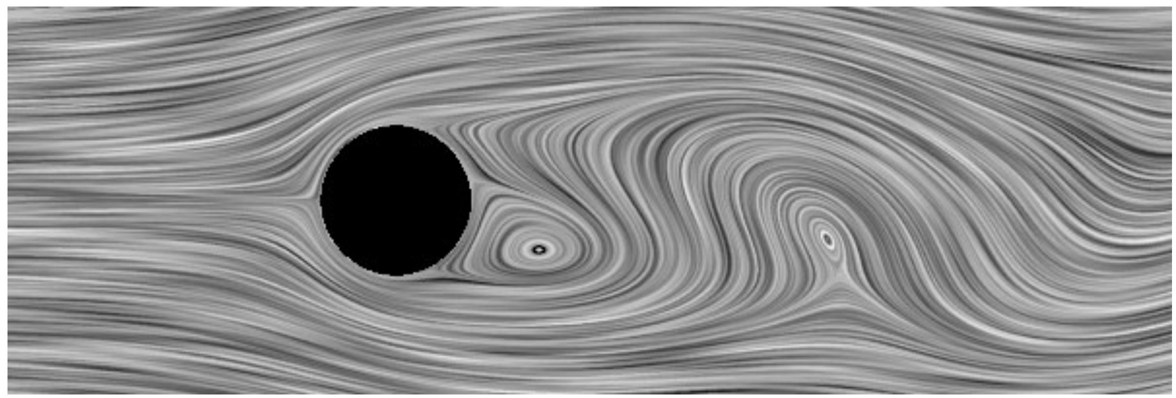
- Motion of individual particles (cavity.avi)

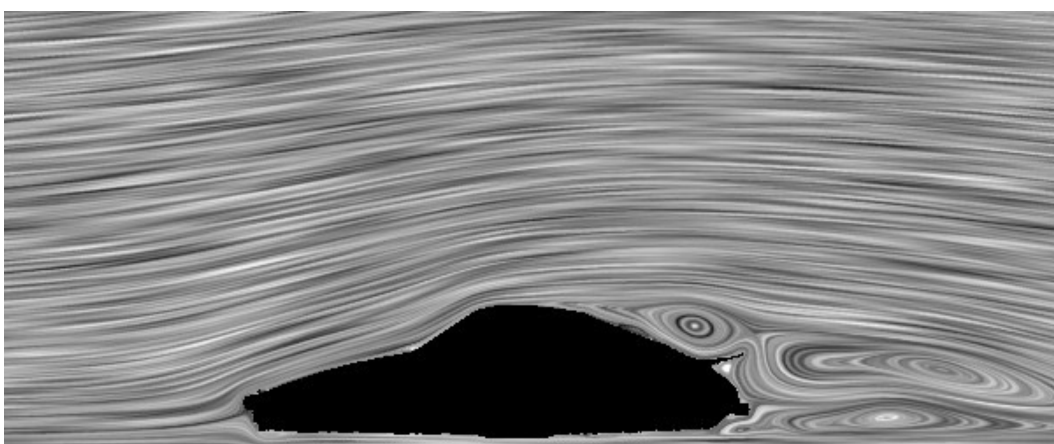
Overview

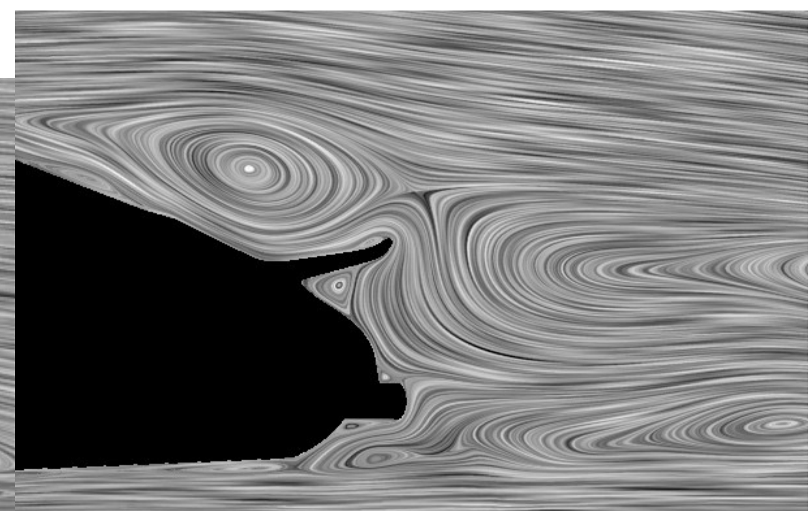
- Problem setting
- Vector calculus
- Characteristic lines
- Arrows and glyphs
- Particle tracing and mapping methods
- Particle tracing on grids
- **Line integral convolution**
- Texture advection
- **Topology-based visualization**
- 3D vector fields

Mapping Methods Based on Particle Tracing

- LIC (Line Integral Convolution)
  - Texture representation
  - Dense

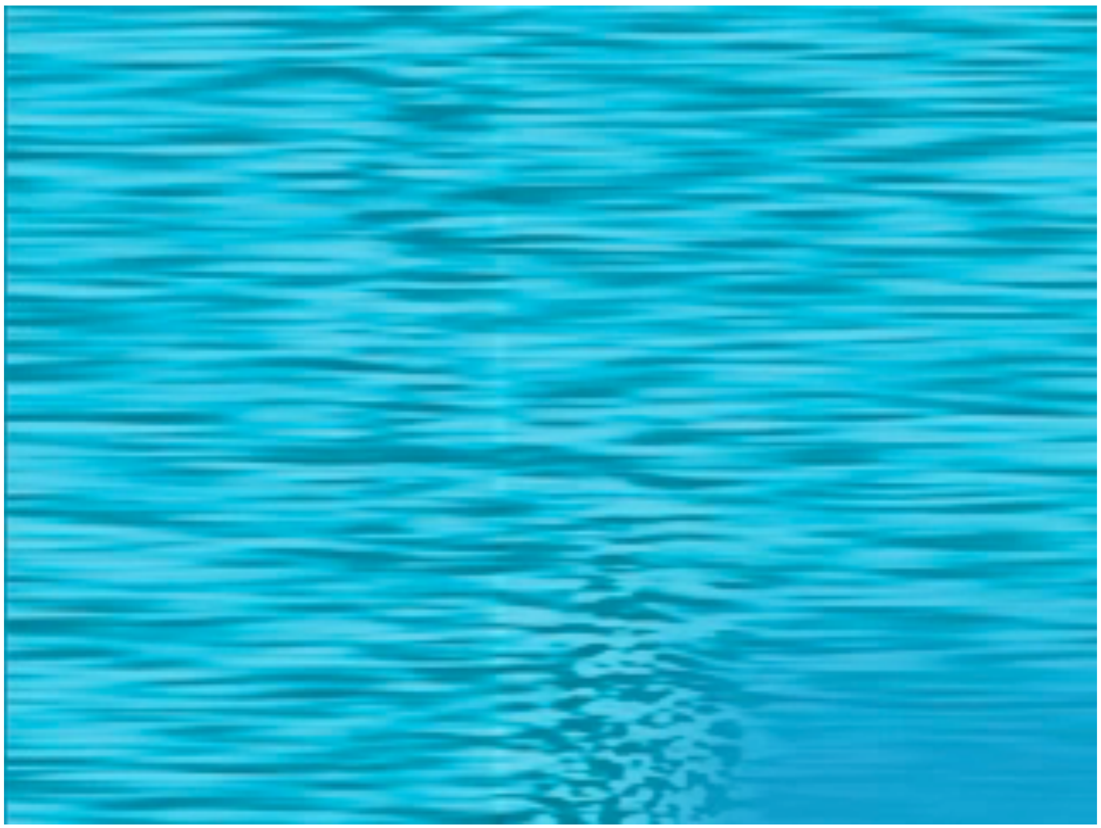






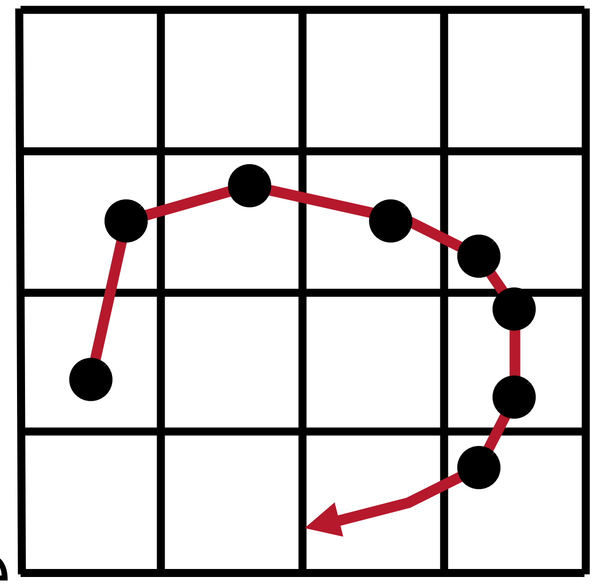
Mapping Methods Based on Particle Tracing

- Unsteady flow advection-convolution
  - Animation



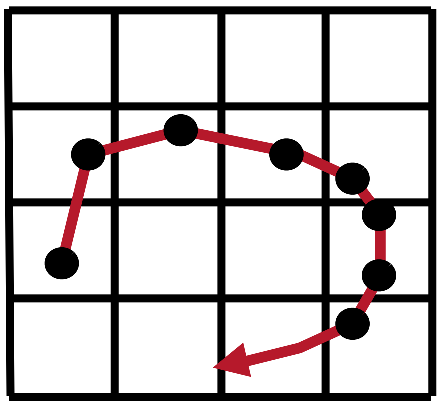
Particle Tracing on Grids

- Vector field given on a grid
- Solve $\mathbf{L}(0) = \mathbf{x}_0$, $\frac{d\mathbf{L}(t)}{dt} = \mathbf{v}(\mathbf{L}(t), t)$ for the pathline
- Incremental integration
- Discretized path of the particle



Particle Tracing on Grids

- Most simple case: Cartesian grid for the pathline
- Basic algorithm:



```
Select start point (seed point)
 Find cell that contains start point
 While (particle in domain) do
     Interpolate vector field at
       current position
     Integrate to new position
     Find new cell
     Draw line segment between latest
       particle positions
 Endwhile
```

point location

interpolation

integration

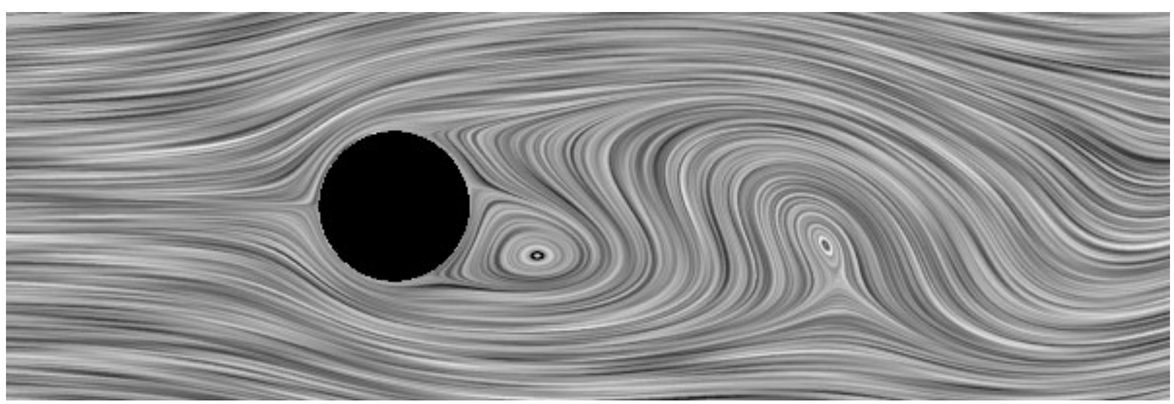
point location

Particle Tracing on Grids

- Point location (cell search) on Cartesian grids:
  - Indices of cell directly from position (x, y, z)
  - For example: $i_x = (x - x_0) / \Delta x$
  - Simple and fast
- Interpolation on Cartesian grids:
  - Bilinear (in 2D) or trilinear (in 3D) interpolation
  - Required to compute the vector field (= velocity) inside a cell
  - Component-wise interpolation
  - Based on offsets (= local coordinates within cell)

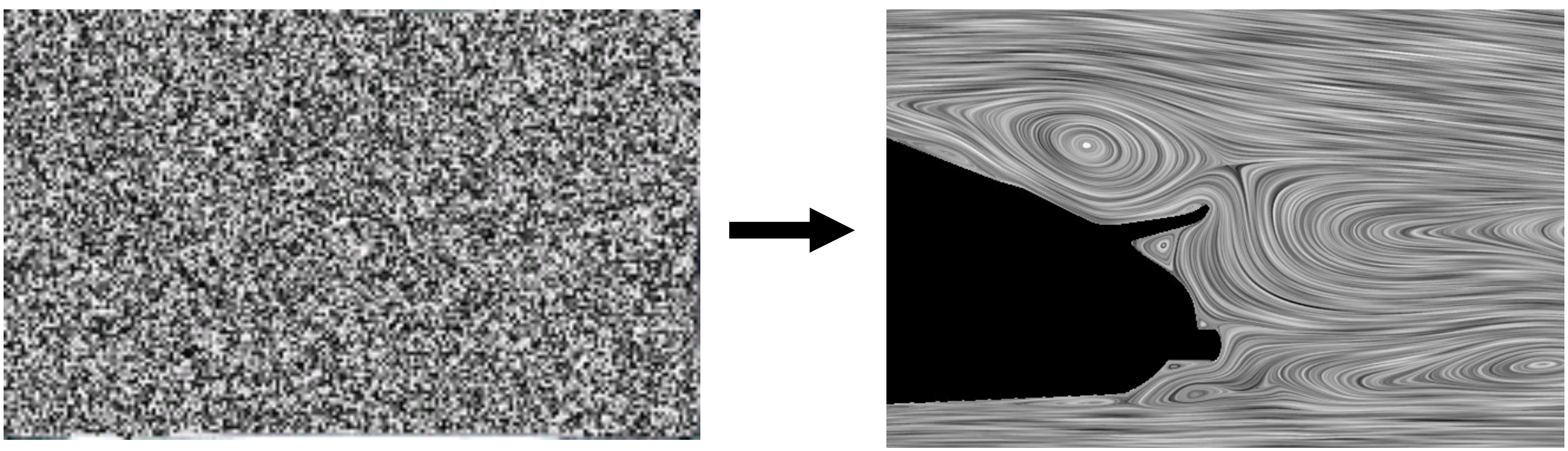
Line Integral Convolution

- Line Integral Convolution (LIC)
  - Visualize dense flow fields by imaging its integral curves
  - Cover domain with a random texture (so called ‚input texture‘, usually stationary white noise)
  - Blur (convolve) the input texture along the path lines using a specified filter kernel
- Look of 2D LIC images
  - Intensity distribution along path lines shows high correlation
  - No correlation between neighboring path lines



Line Integral Convolution

- Idea of Line Integral Convolution (LIC)
  - Global visualization technique
  - Dense representation
  - Start with random texture
  - Smear out along stream lines



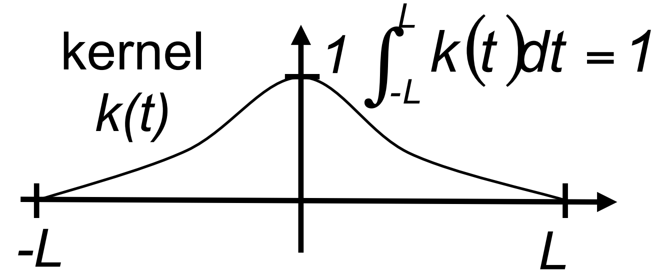
Line Integral Convolution

- Algorithm for 2D LIC
  - Let $t \rightarrow \Phi_0(t)$ be the path line containing the point (x_0, y_0)
  - $T(x,y)$ is the randomly generated input texture
  - Compute the pixel intensity as:

$$I(x_0, y_0) = \int_{-L}^{L} k(t) \times T(\phi_0(t))\, dt$$

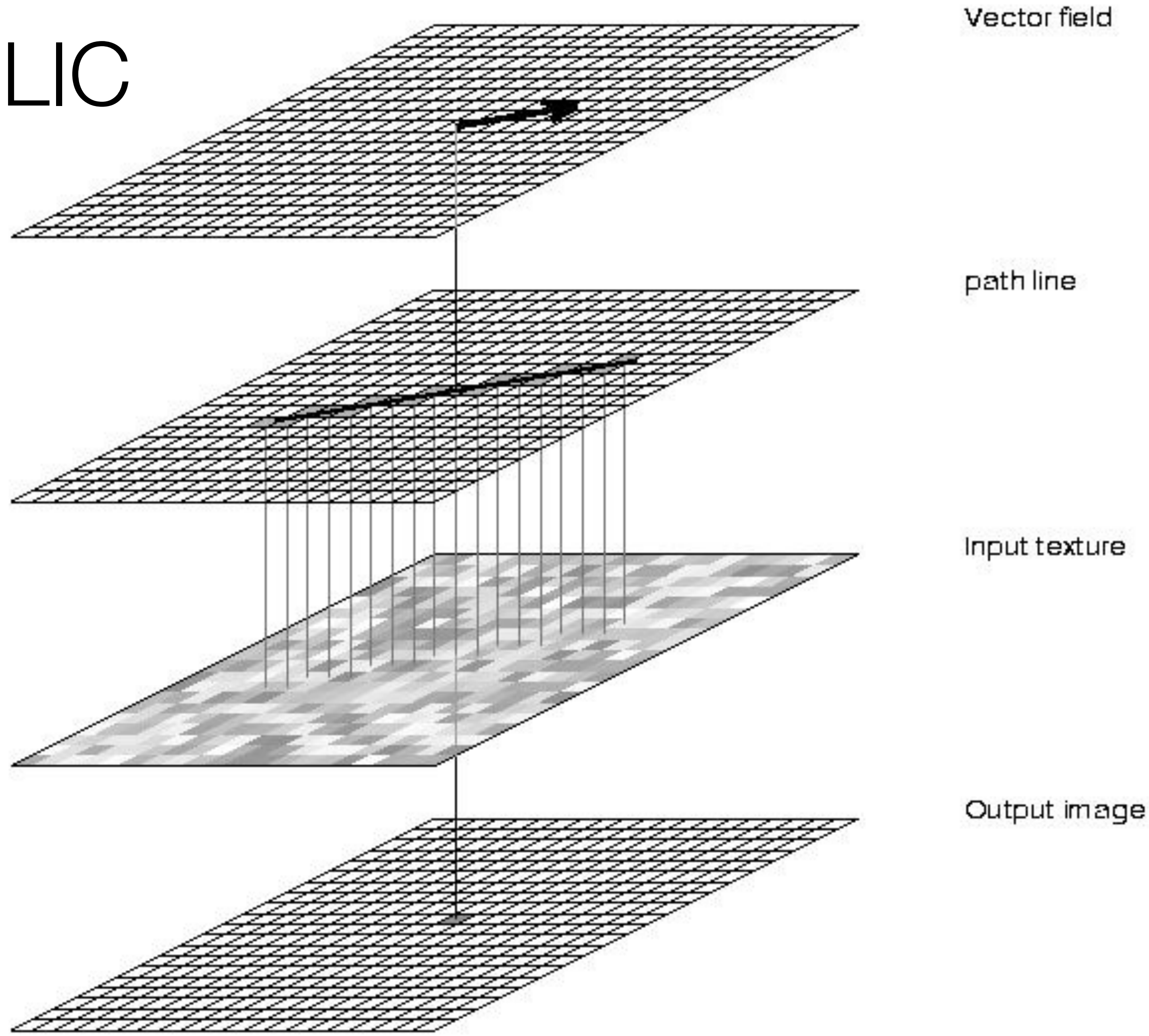
convolution with kernel

- Kernel:
  - Finite support [-*L*,*L*]
  - Normalized
  - Often simple box filter
  - Often symmetric (isotropic)

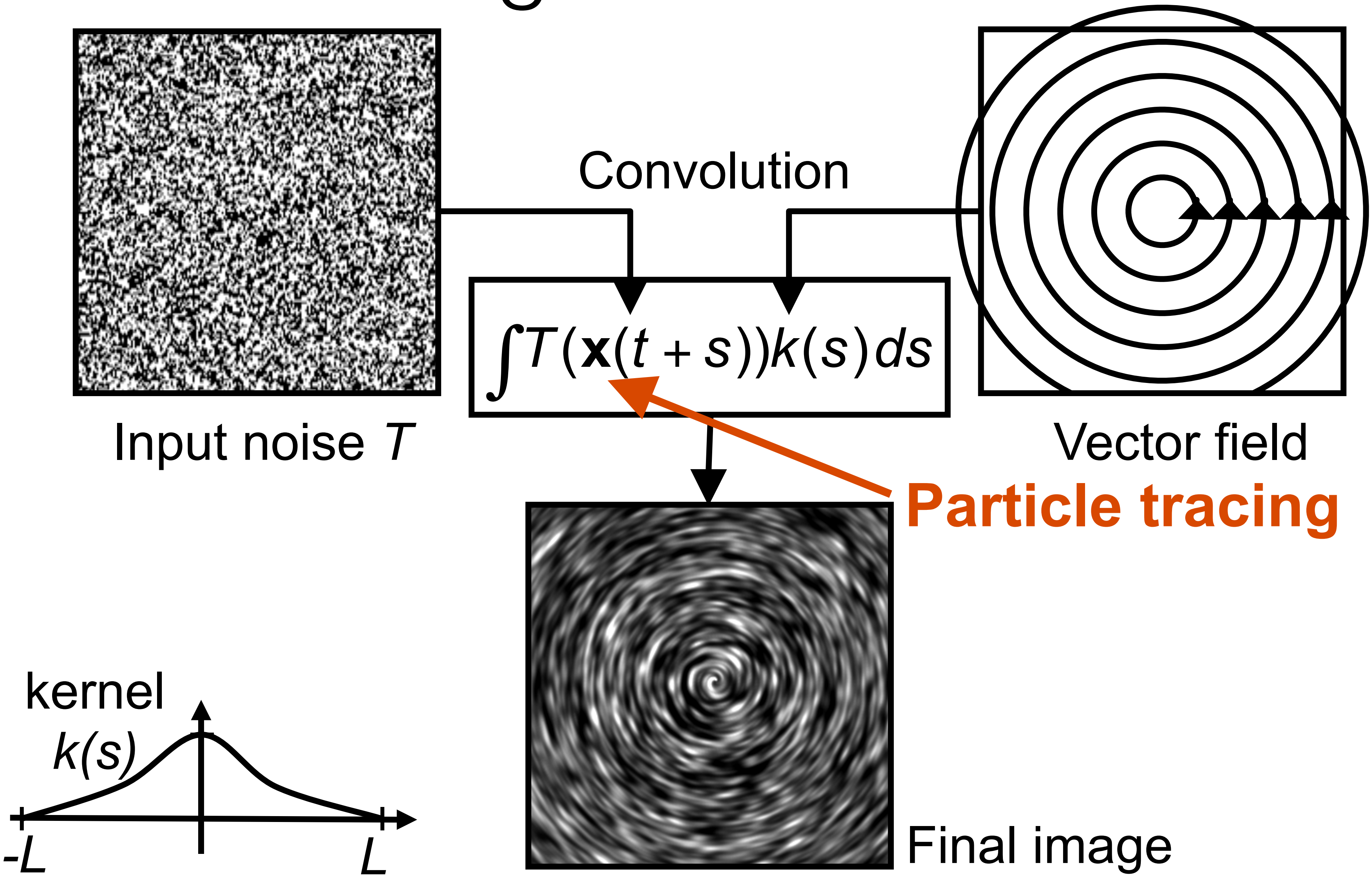


Line Integral Convolution

- Algorithm for 2D LIC
  – Convolve a random texture along the streamlines



Line Integral Convolution

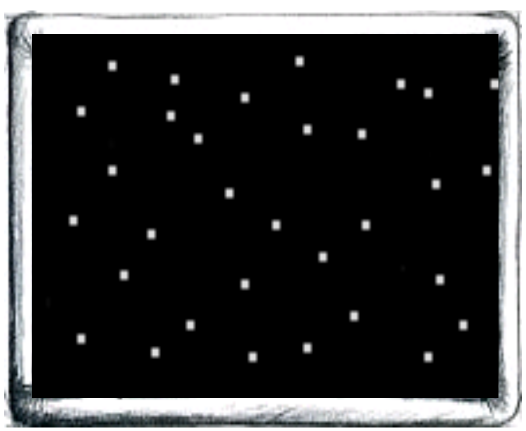


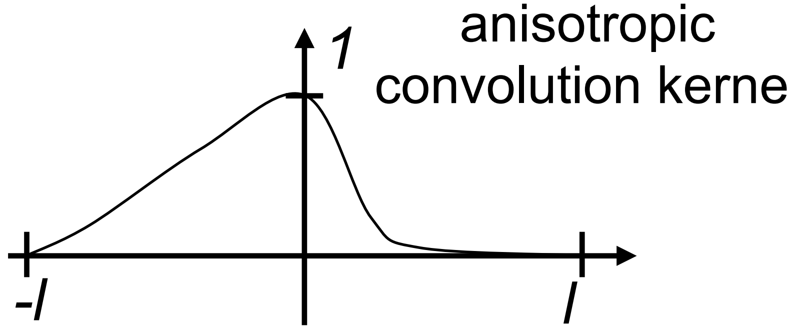
Line Integral Convolution

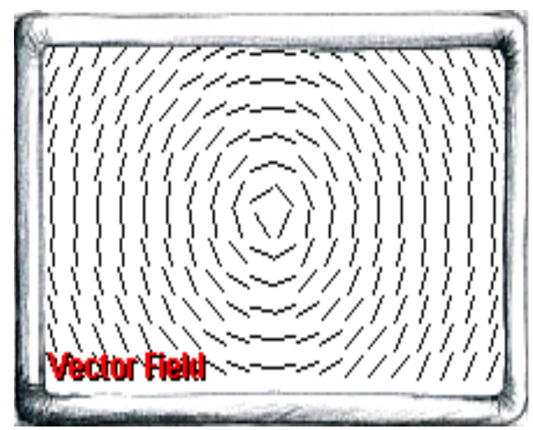
- Fast LIC
- Problems with LIC
  - New streamline is computed at each pixel
  - Convolution (integral) is computed at each pixel
  - Slow
- Idea:
  - Compute very long streamlines
  - Reuse these streamlines for many different pixels
  - Incremental computation of the convolution integral

Line Integral Convolution

- Oriented LIC (OLIC):
  - Visualizes orientation (in addition to direction)
  - Sparse texture
  - Anisotropic convolution kernel
  - Acceleration: integrate individual drops and compose them to final image

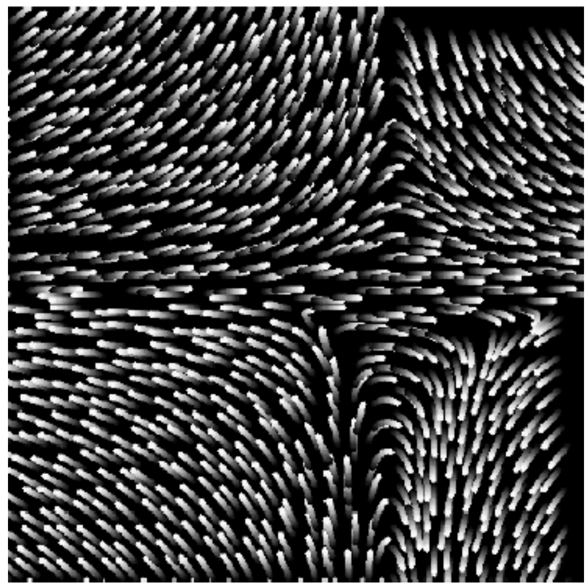


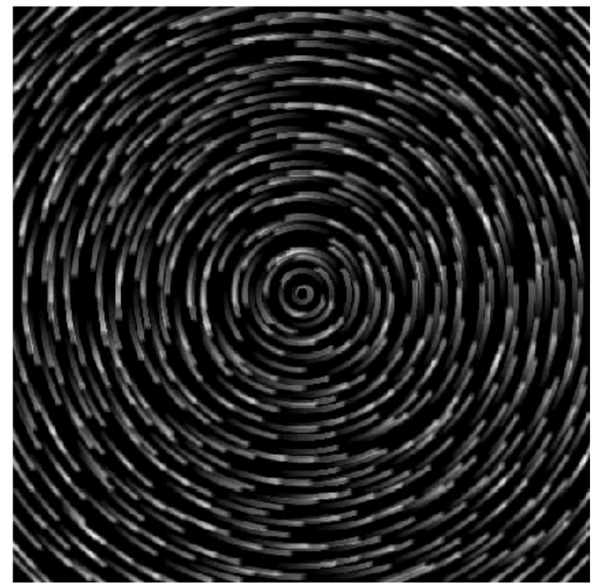




Line Integral Convolution

- Oriented LIC (OLIC)





- Video -- CylinderStreakMovieNoTitleLICAtEnd

Line Integral Convolution

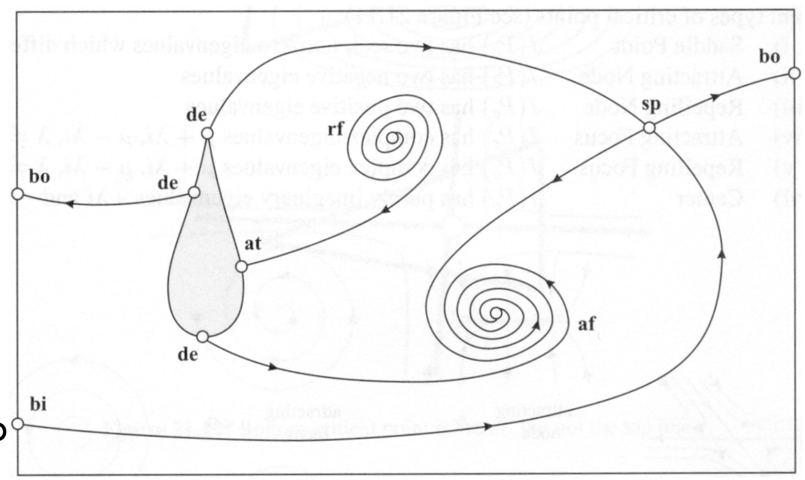
- Outlook
  - GPU LIC for real-time visualization
  - Texture advection (also on GPUs) for an incremental computation
    - Especially useful for time-dependent vector fields
  - Extension to 2.5D and 3D data sets

Overview

- Problem setting
- Vector calculus
- Characteristic lines
- Arrows and glyphs
- Particle tracing and mapping methods
- Particle tracing on grids
- Line integral convolution
- Texture advection
- **Topology-based visualization**
- 3D vector fields

Vector Field Topology

- Idea:
  Do not draw "all" streamlines, but only the "important" streamlines
- Show only topological skeletons
- Important points in the vector field: critical points
- Critical points:
  - Points where the vector field vanishes: v = 0
  - Points where the vector magnitude goes to zero and the vector direction is undefined
  - Sources, sinks, …
- The critical points are connected to divide the flow into regions with similar properties
- Structure of particle behavior for $t \rightarrow \infty$



Vector Field Topology

- Finding “critical” points
- what is critical in a flow?
- Well - when it doesn’t flow anymore!
- I.e - critical points are places without change: $v = 0$!
- Try to
  - find these places
  - classify them

Vector Field Topology

- First we need understand such places!
- two flows can cancel each other out
- center of vortex …
- Look at derivative of v!

$$v_i = v_i^{(0)} + \left(x_j - x_j^{(0)}\right)\frac{\partial v_i}{\partial x_j} + \ldots$$

Vector Field Topology

1. Find critical points

- pretty much iso-value algorithm
- but with a twist - since three components are zero
- find iso-values for each component and then only consider cells where all three intersect
- not enough - sub-divide potential cells until a certain bound is reached.

Vector Field Topology

2. classify critical points

- according to what is happening in the neighborhood - attracting or repelling or a combination thereof
- determined by derivative of velocity
- if positive then things move away
- if negative things come closer
- this is 1D

Vector Field Topology

- One dimension:
- if derivative of velocity is:
  - positive: things move away (repelling)
  - negative: things come closer (attracting)

$$v = v^{(0)} + \left(x - x^{(0)}\right)\frac{\partial v}{\partial x} + \ldots$$

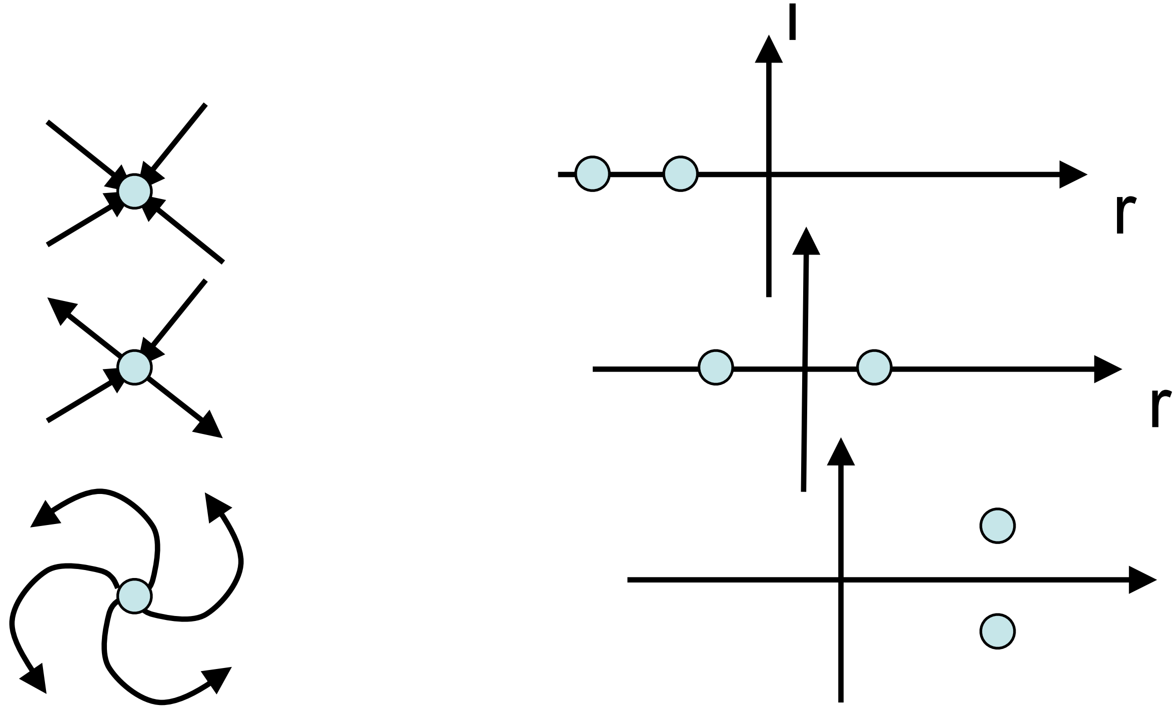
Vector Field Topology

- 2D classification (and higher D):
- according to eigen-values of derivative matrix

Attracting node

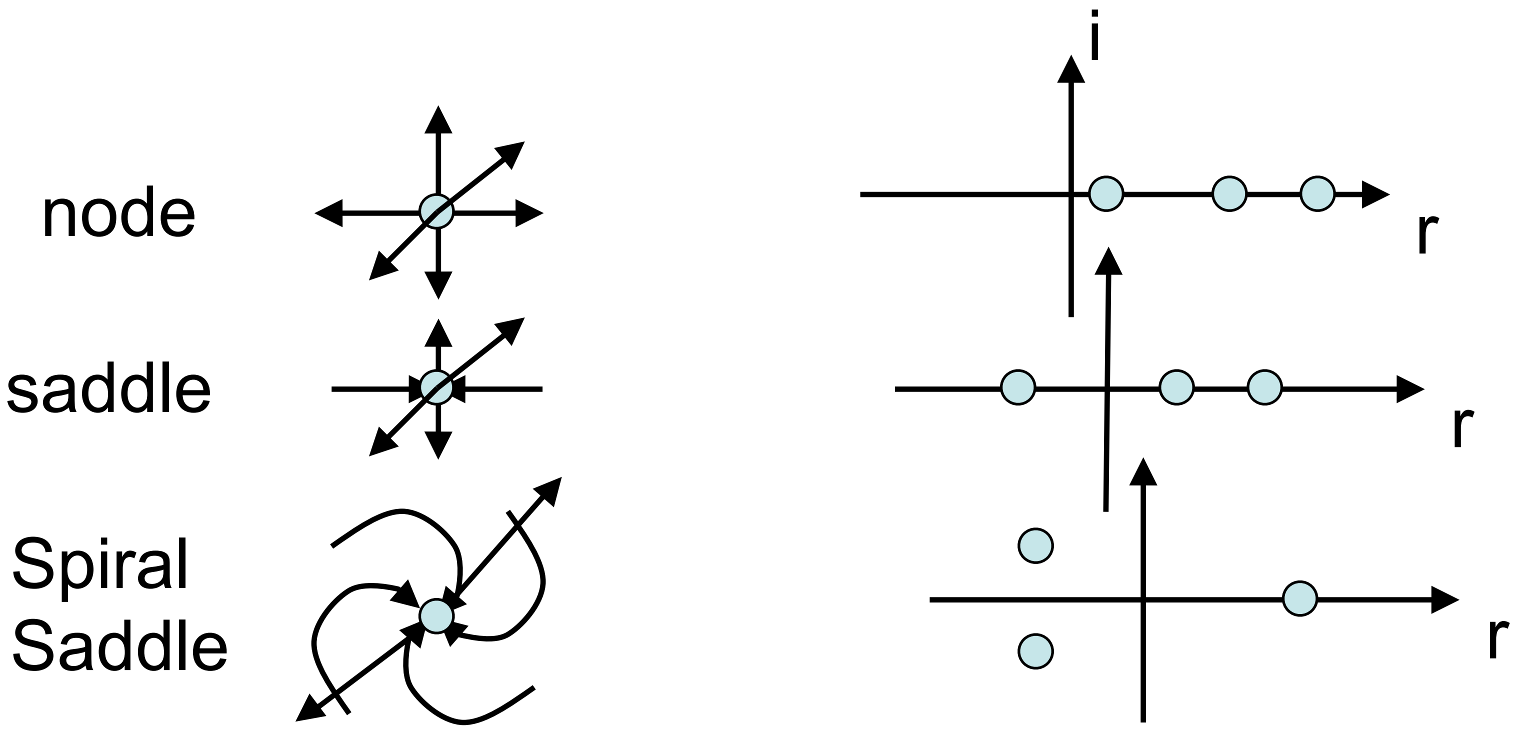
Saddle

Repelling focus



Vector Field Topology

- 3D classification
- more complicated



Vector Field Topology

- Taylor expansion for the velocity field around a critical point $\mathbf{r}_c$:

$$\mathbf{v}(\mathbf{r}) = \mathbf{v}(\mathbf{r}_c) + \nabla\mathbf{v} \cdot (\mathbf{r} - \mathbf{r}_c) + O(\mathbf{r} - \mathbf{r}_c)^2$$
$$\approx \mathbf{J} \cdot (\mathbf{r} - \mathbf{r}_c)$$

- Divide Jacobian into symmetric and anti-symmetric parts

$$\mathbf{J} = \mathbf{J}_s + \mathbf{J}_a = ((\mathbf{J} + \mathbf{J}^T) + (\mathbf{J} - \mathbf{J}^T))/2$$

$$\mathbf{J}_s = (\mathbf{J} + \mathbf{J}^T)/2$$
$$\mathbf{J}_a = (\mathbf{J} - \mathbf{J}^T)/2$$

Vector Field Topology

- The symmetric part can be solved to give real eigenvalues *R* and real eigenvectors

$$\mathbf{J}_s \mathbf{r}_s = R\mathbf{r}_s \qquad R = R_1, R_2, R_3$$

  - Eigenvectors $\mathbf{r}_s$ are an orthonormal set of vectors
  - Describes change of size along eigenvectors
  - Describes flow into or out of region around critical point

Vector Field Topology

- Anti-symmetric part

$$\mathbf{J}_a \times \mathbf{d} = \tfrac{1}{2}\left(\mathbf{J} - \mathbf{J}^T\right) \times \mathbf{d} =$$

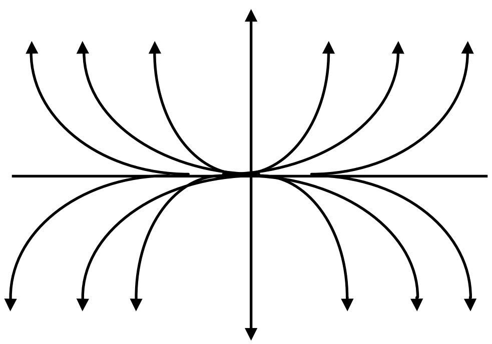
$$\frac{1}{2}\begin{pmatrix} 0 & \frac{\partial \boldsymbol{v}_x}{\partial y} - \frac{\partial \boldsymbol{v}_y}{\partial x} & \frac{\partial \boldsymbol{v}_x}{\partial z} - \frac{\partial \boldsymbol{v}_z}{\partial x} \\ \frac{\partial \boldsymbol{v}_y}{\partial x} - \frac{\partial \boldsymbol{v}_x}{\partial y} & 0 & \frac{\partial \boldsymbol{v}_y}{\partial z} - \frac{\partial \boldsymbol{v}_z}{\partial y} \\ \frac{\partial \boldsymbol{v}_z}{\partial x} - \frac{\partial \boldsymbol{v}_x}{\partial z} & \frac{\partial \boldsymbol{v}_z}{\partial y}\ \frac{\partial \boldsymbol{v}_y}{\partial z} & 0 \end{pmatrix} \times \mathbf{d} = \tfrac{1}{2}\left(\nabla \times \boldsymbol{v}\right) \times \mathbf{d}$$

$$\mathbf{J}_a \mathbf{r}_a = l \mathbf{r}_a \qquad l = l_1, l_2, l_3$$

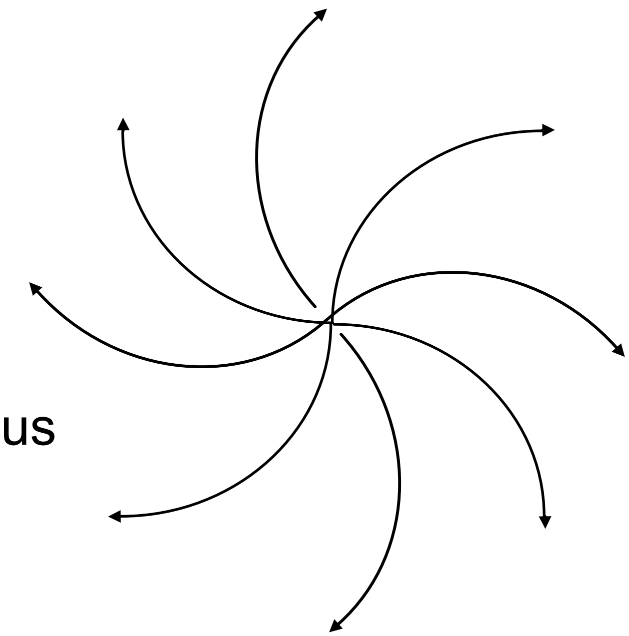
  - Describes rotation of difference vector $\mathbf{d} = (\mathbf{r} - \mathbf{r}_c)$
  - The anti-symmetric part can be solved to give imaginary eigenvalues l

Vector Field Topology

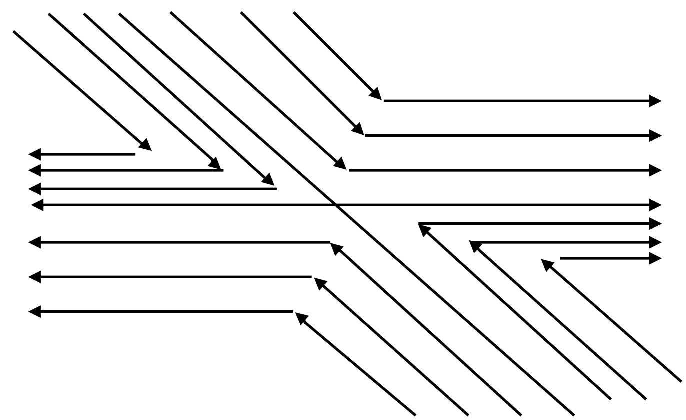
- 2D structure: eigenvalues are ($\boldsymbol{R}_1$, $\boldsymbol{R}_2$) and ($\boldsymbol{I}_1$,$\boldsymbol{I}_2$)



Repelling node
R_1, $R_2 > 0$
I_1,I_2 = 0



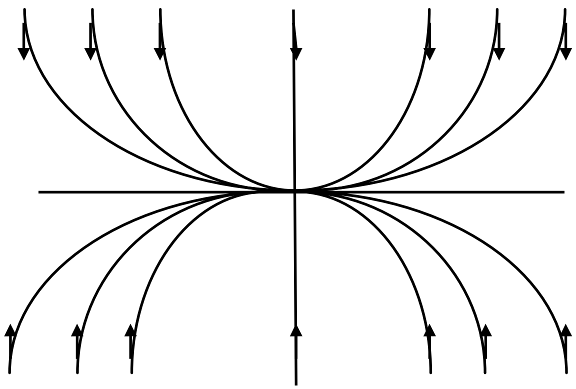
Repelling focus
R_1, $R_2 > 0$
I_1,$I_2 \neq 0$



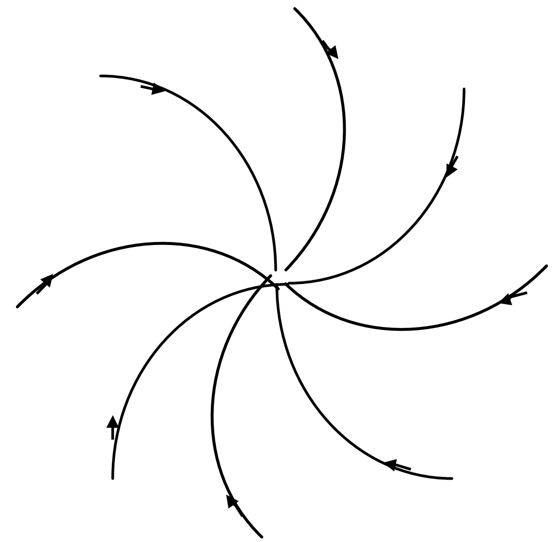
Saddle point
$R_1 * R_2 < 0$
I_1,I_2 = 0

Vector Field Topology

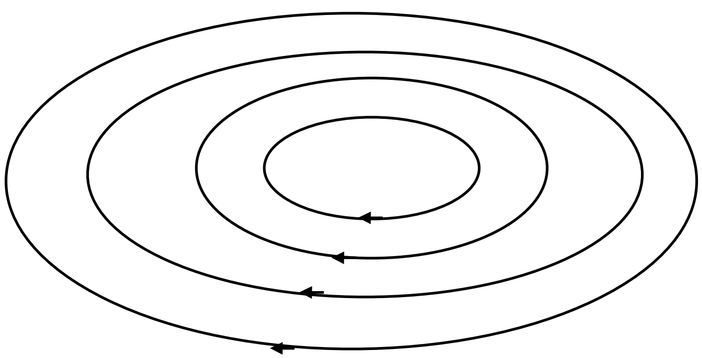
- 2D structure: eigenvalues are (R_1, R_2) and (I_1, I_2)



Attracting node
$R_1, R_2 < 0$
$I_1, I_2 = 0$



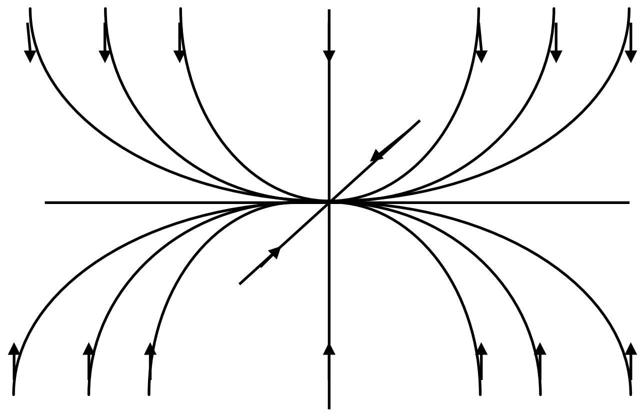
Attracting focus
$R_1, R_2 < 0$
$I_1, I_2 \neq 0$



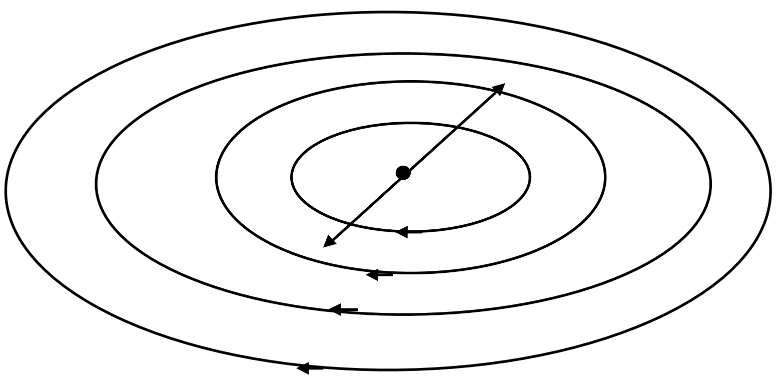
Center
$R_1, R_2 = 0$
$I_1, I_2 \neq 0$

Vector Field Topology

- Also in 3D
  - Some examples



Attracting node
$R_1, R_2, R_3 < 0$
$I_1, I_2, I_3 = 0$



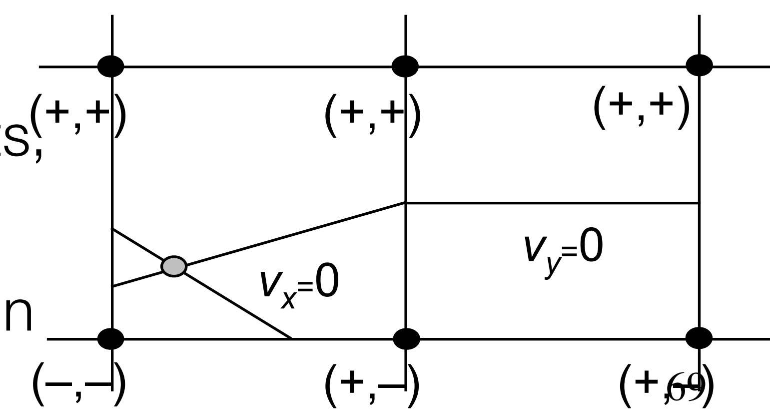
Center
$R_1, R_2 = 0, R_3 > 0$
$I_1, I_2 \neq 0, , I_3 = 0$

Vector Field Topology

- Mapping to graphical primitives: streamlines
  - Start streamlines close to critical points
  - Initial direction along the eigenvectors
- End particle tracing at
  - Other “real” critical points
  - Interior boundaries: attachment or detachment points
  - Boundaries of the computational domain

Vector Field Topology

- How to find critical points
  - Cell search (for cells which contain critical points):
    - Mark vertices by (+,+), (–, –), (+, –) or (–,+), depending on the signs of v_x and v_y
    - Determine cells that have vertices where the sign changes in both components –> these are the cells that contain critical points
  - How to find critical points within a (quad) cell ?
    - Find the critical points by interpolation
    - Determine the intersection of the isolines (c=0) of the two components
    - Two bilinear equations to be solved
    - Critical points are the solutions within the cell boundaries

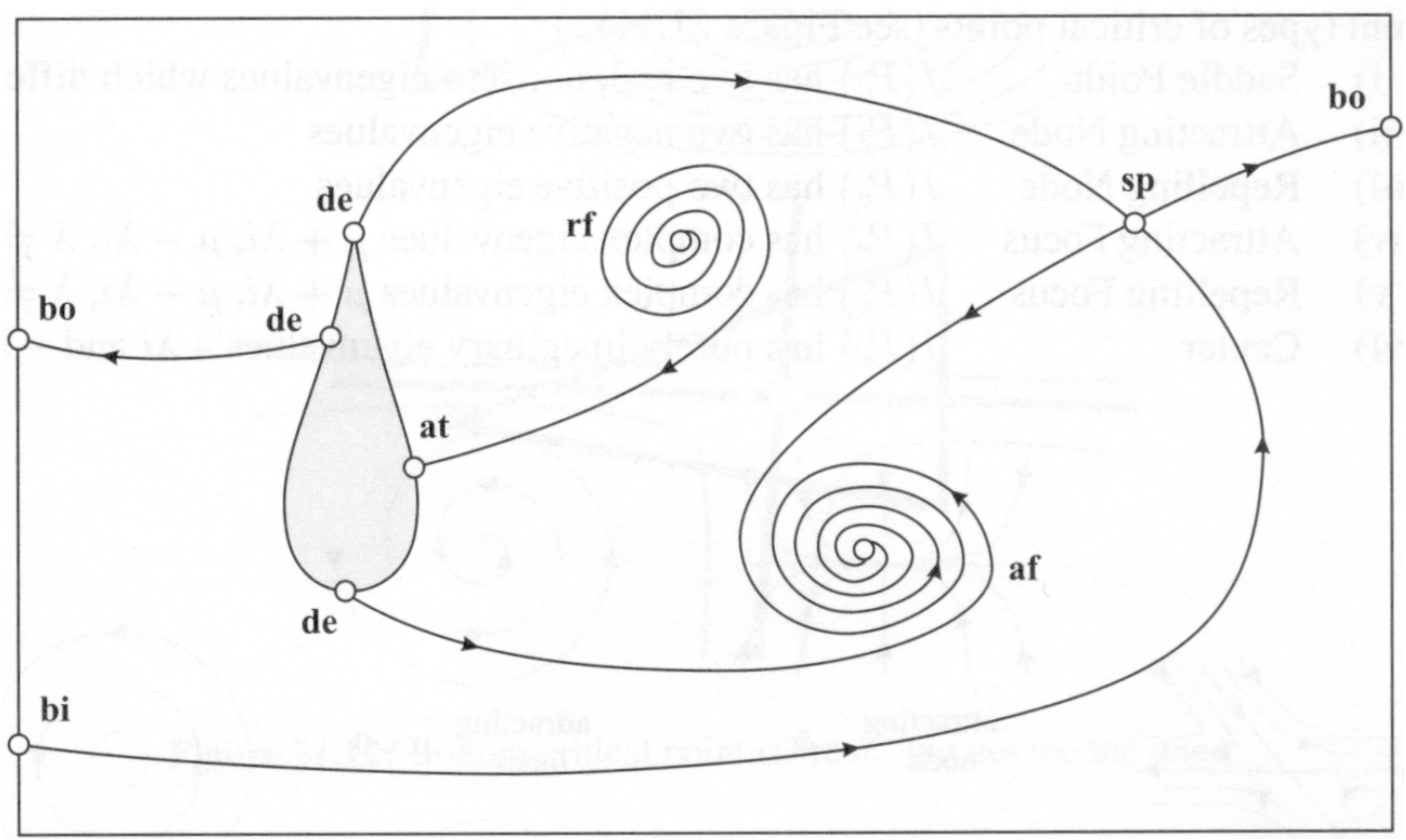


Vector Field Topology

- How to find critical points (cont.)
  - How to find critical points within simplex?
    - Based on barycentric interpolation
    - Solve analytically
  - Alternative method:
    - Iterative approach based on 2D / 3D nested intervals
    - Recursive subdivision into 4 / 8 subregions if critical point is contained in cell

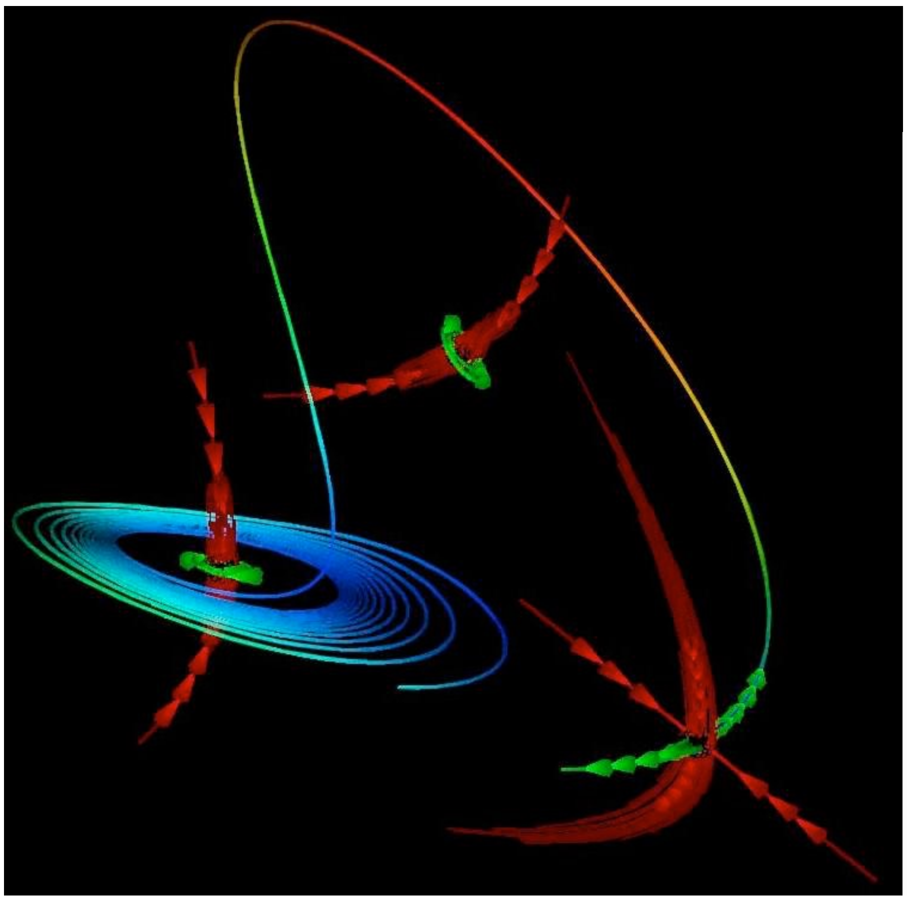
Vector Field Topology

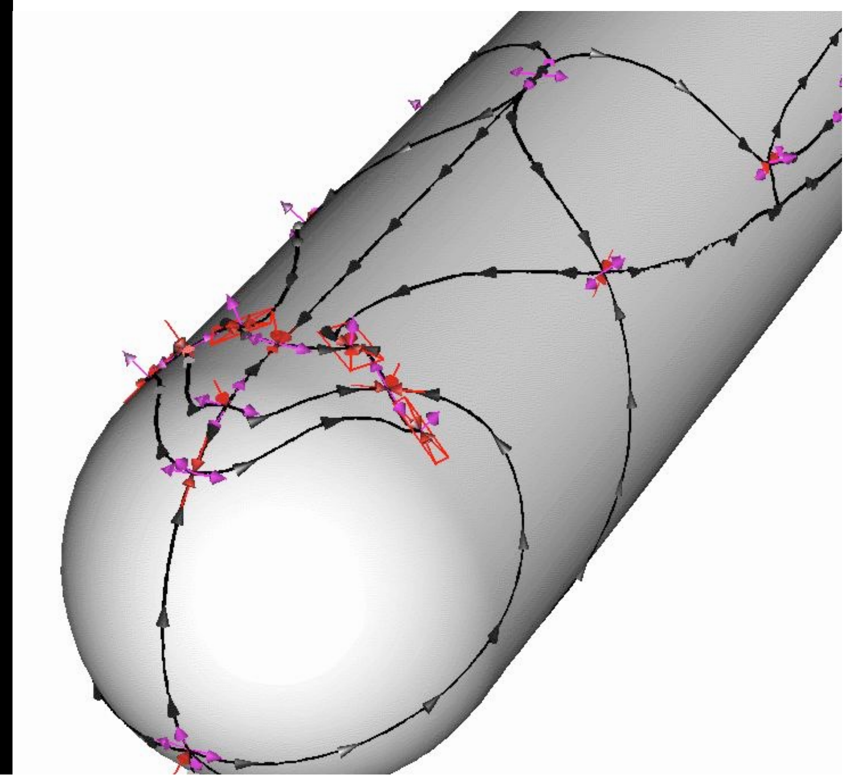
- Example of a topological graph of 2D flow field



Vector Field Topology

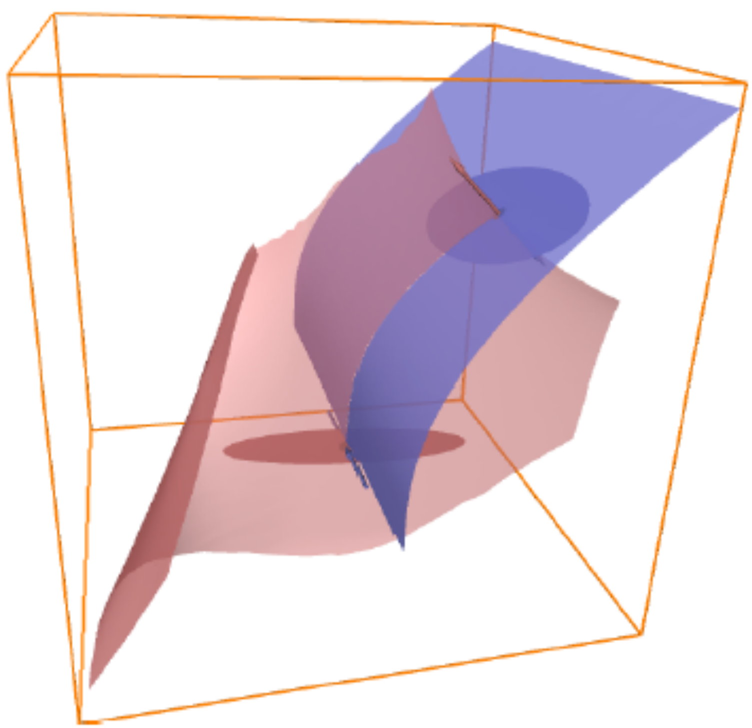
- Further examples of topology-guided streamline positioning



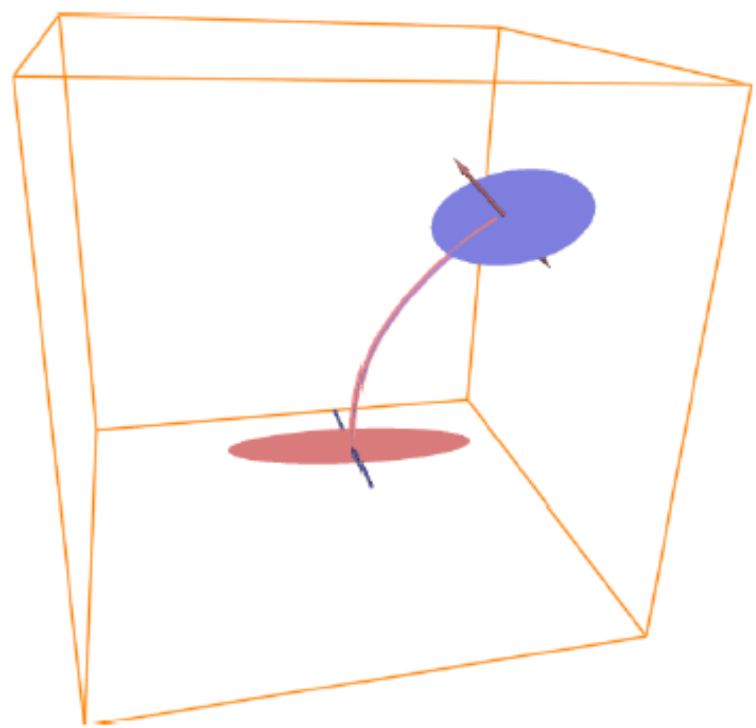


Vector Field Topology

- Saddle connectors in 3D



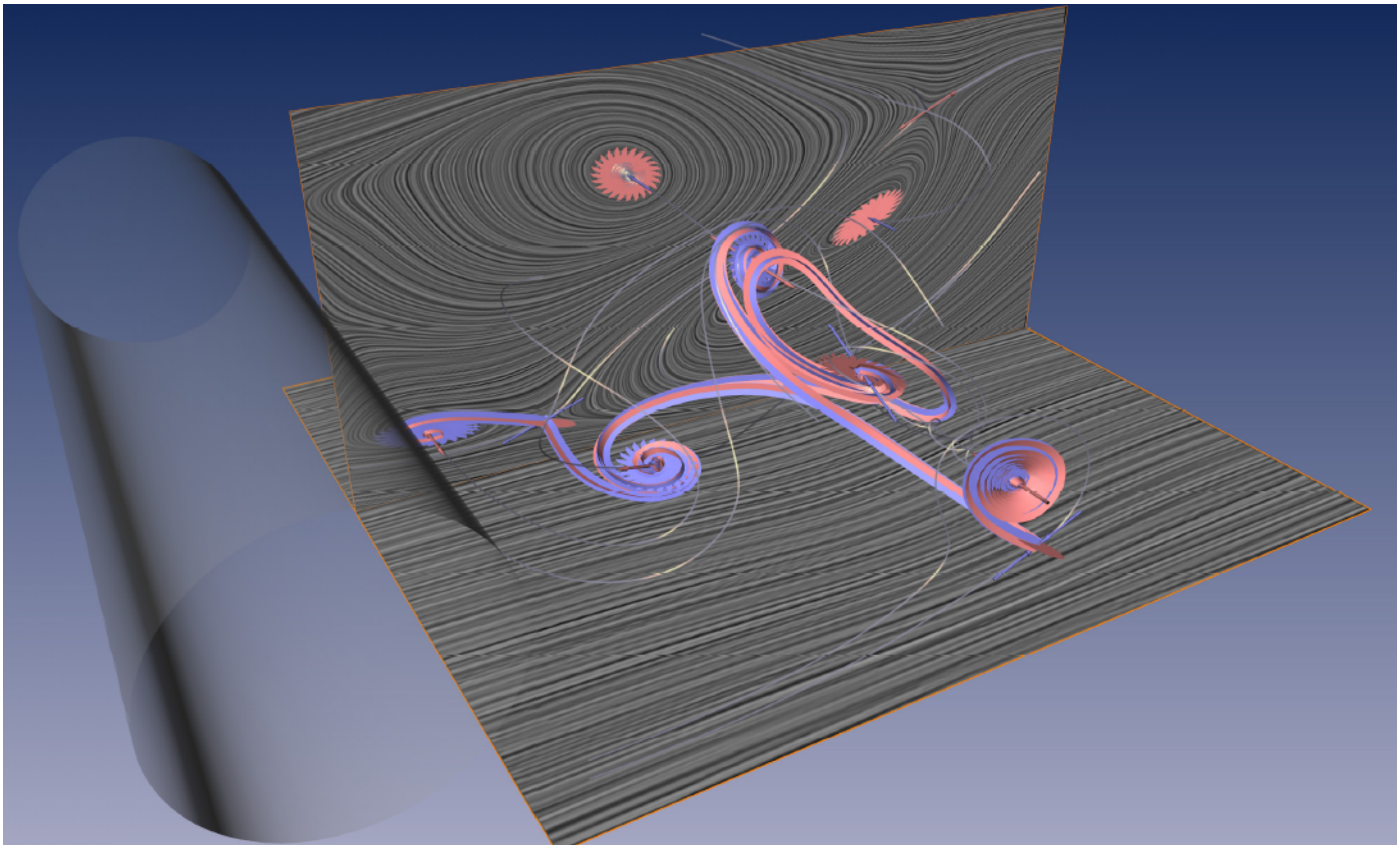
(a) Separation surfaces of the saddles.



(b) The intersection of the separation surfaces is the saddle connector.

Vector Field Topology

- Saddle connectors in 3D



Vector Field Topology

- Summary:
  - Draw only relevant streamlines (topological skeleton)
  - Partition domain in regions with similar flow features
  - Based on critical points
  - Good for 2D stationary flows
  - Unsteady flows?
  - 3D?

3D Vector Fields

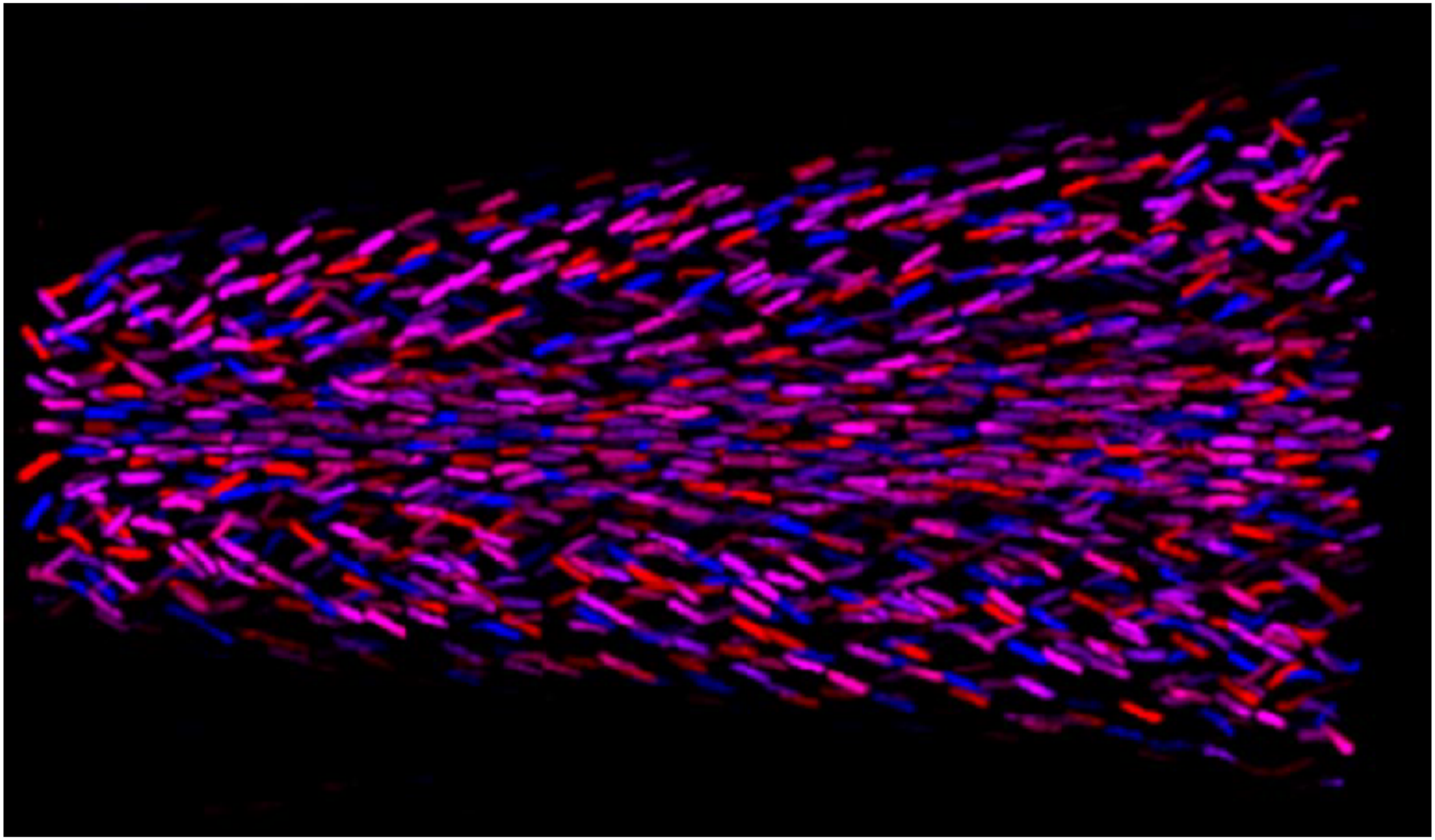
- Most algorithms can be applied to 2D and 3D vector fields
- Main problem in 3D: effective mapping to graphical primitives
- Main aspects:
  - Occlusion
  - Amount of (visual) data
  - Depth perception

3D Vector Fields

- Approaches to occlusion issue:
  - Sparse representations
  - Animation
  - Color differences to distinguish separate objects
  - Continuity
- Reduction of visual data:
  - Sparse representations
  - Clipping
  - Importance of semi-transparency

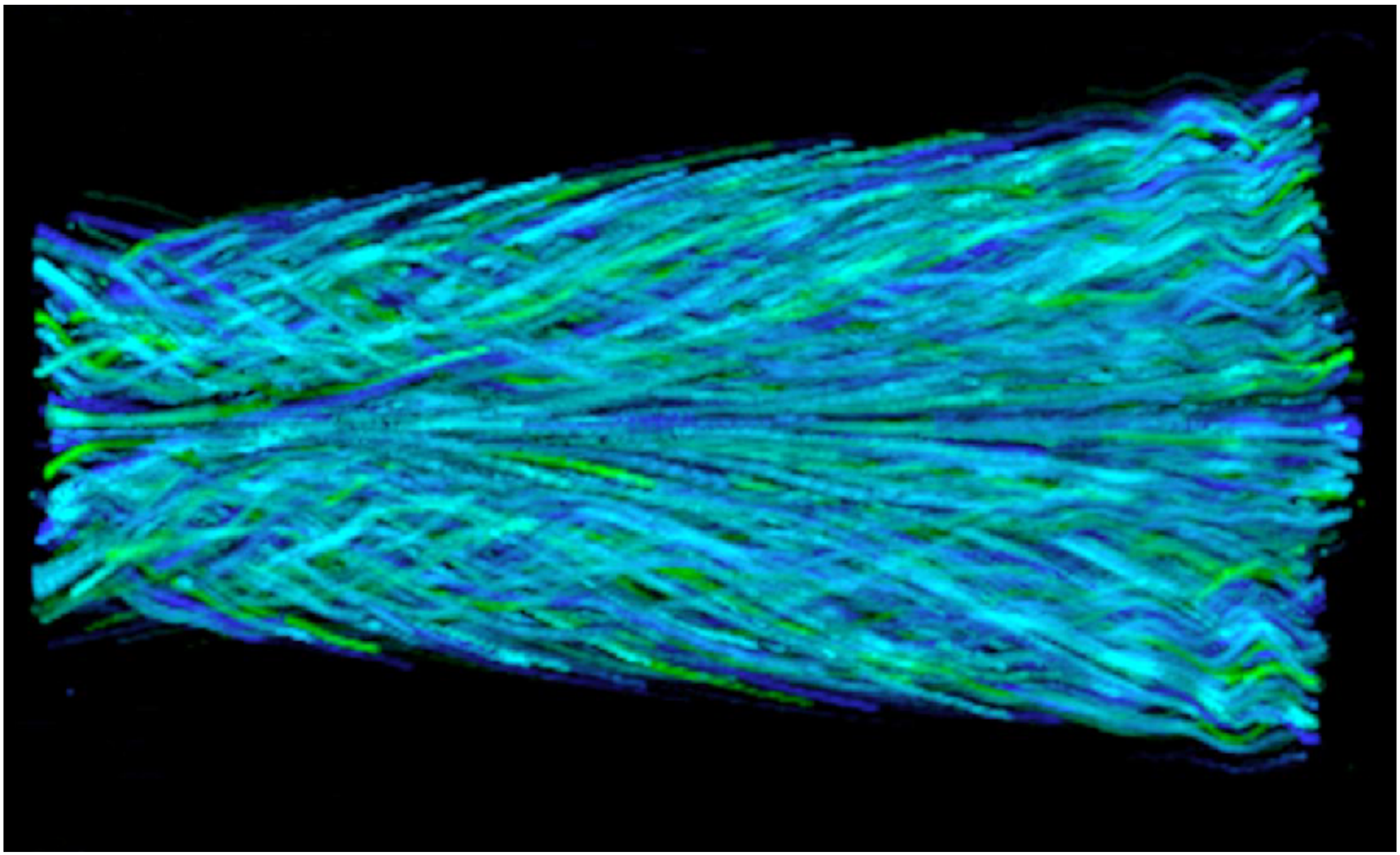
3D Vector Fields

- Missing continuity



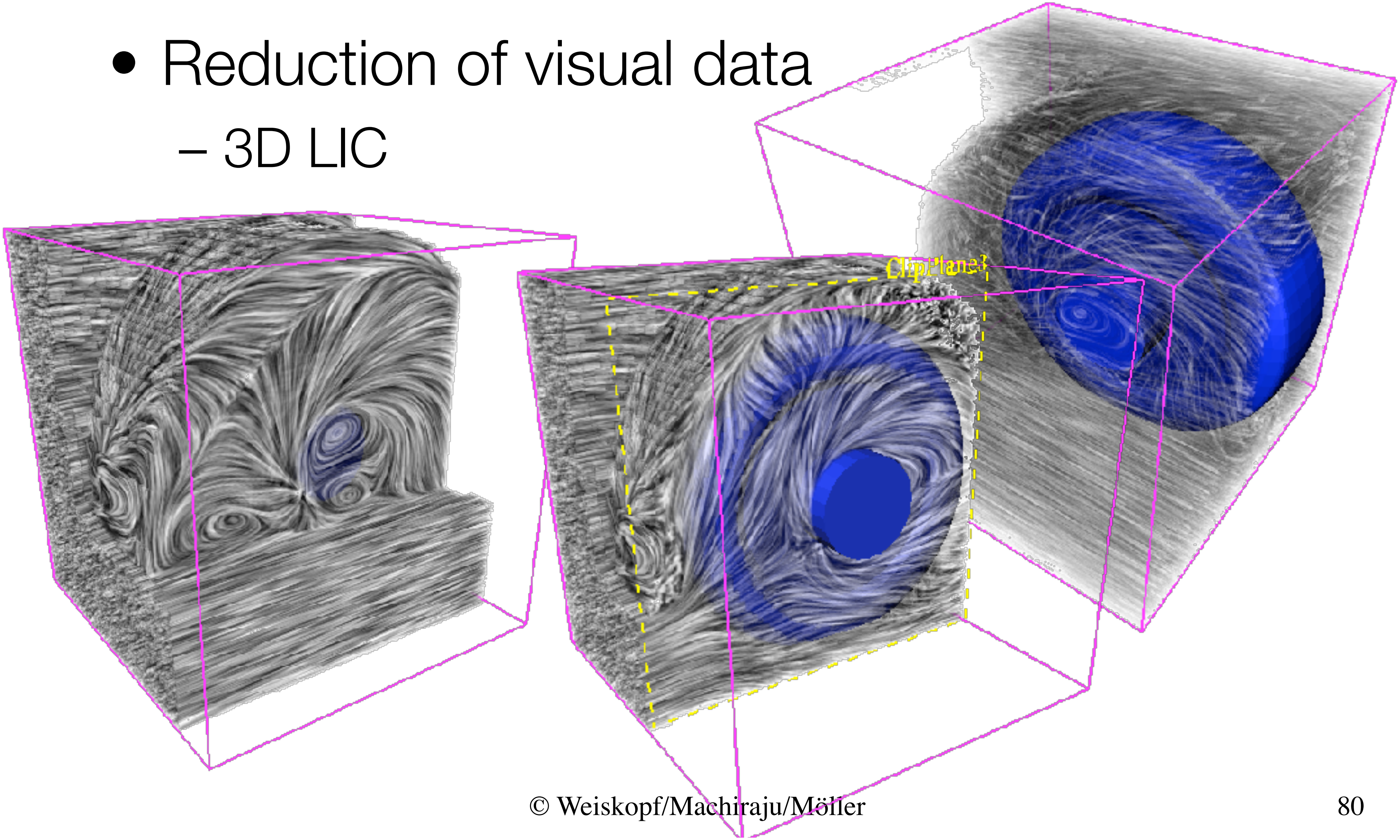
3D Vector Fields

- Color differences to identify connected structures



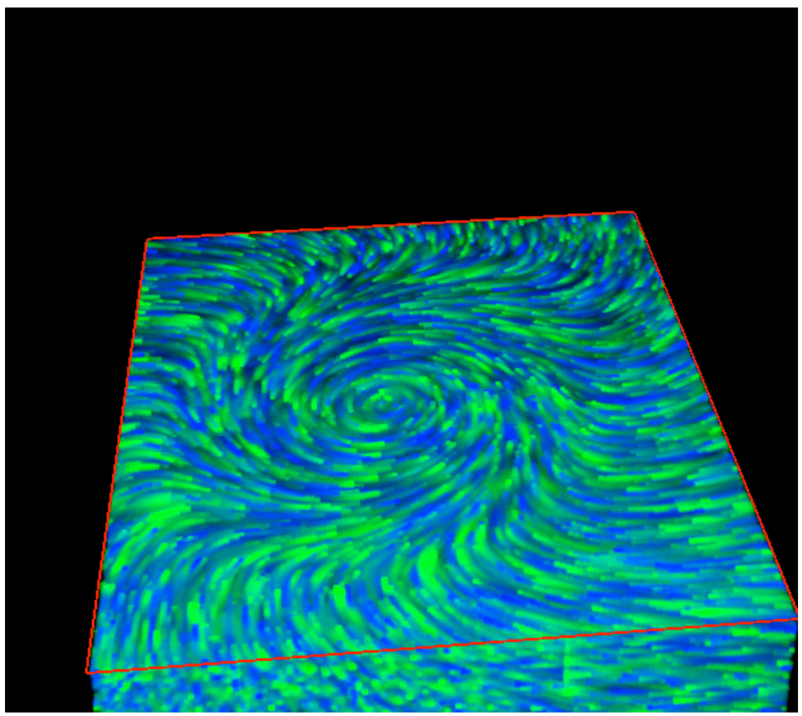
3D Vector Fields

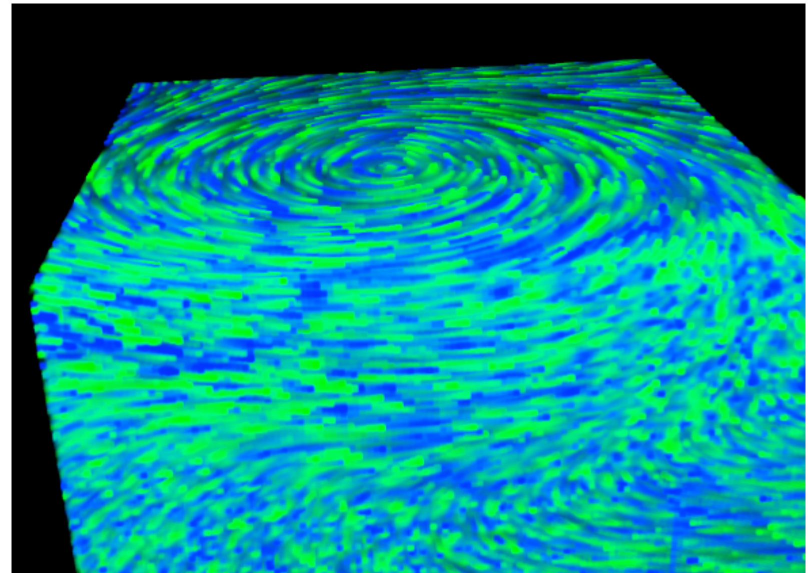
- Reduction of visual data
  - 3D LIC

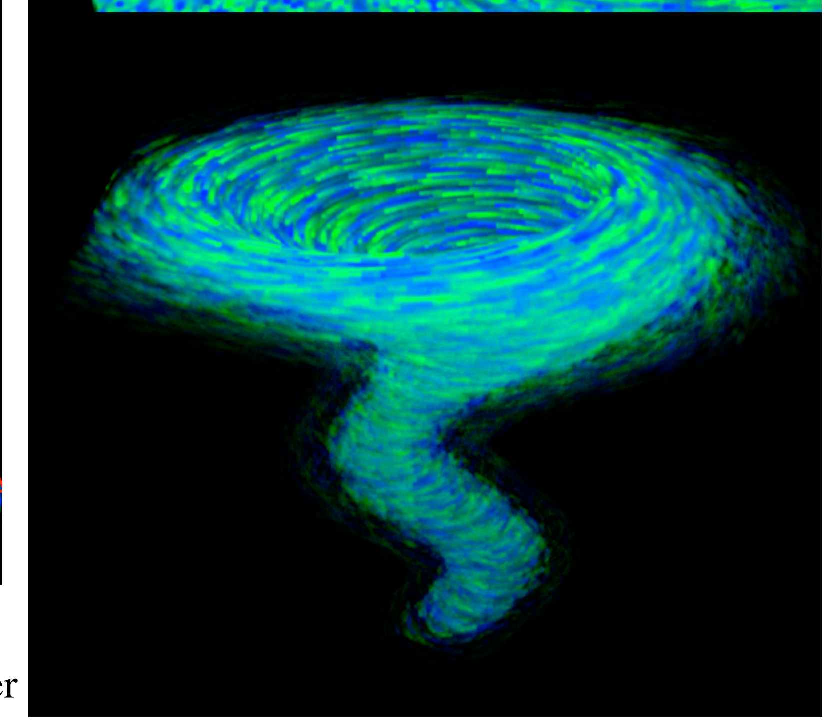


3D Vector Fields

- Reduction of visual data
  - Clipping
  - Masking

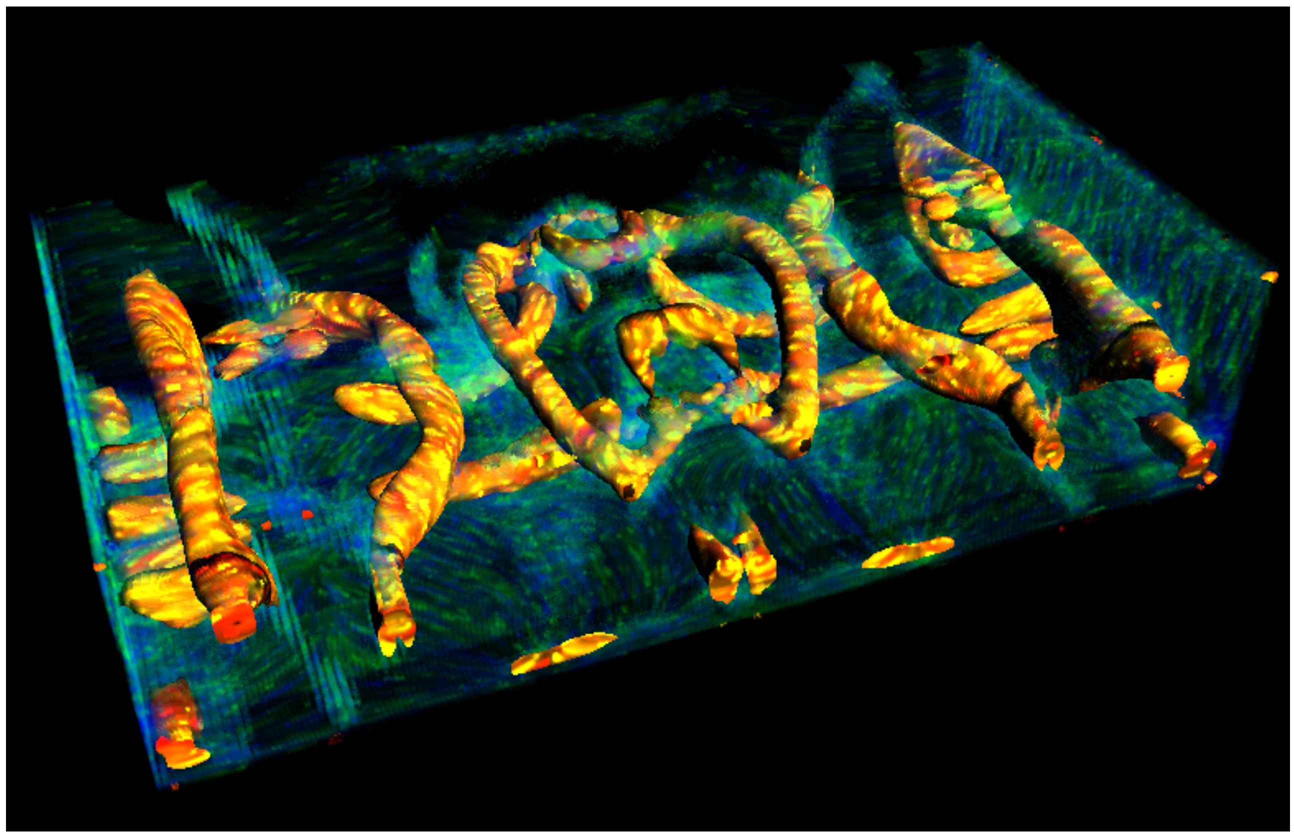






3D Vector Fields

- Reduction of visual data
  - 3D importance function
  - Feature extraction, often interactive



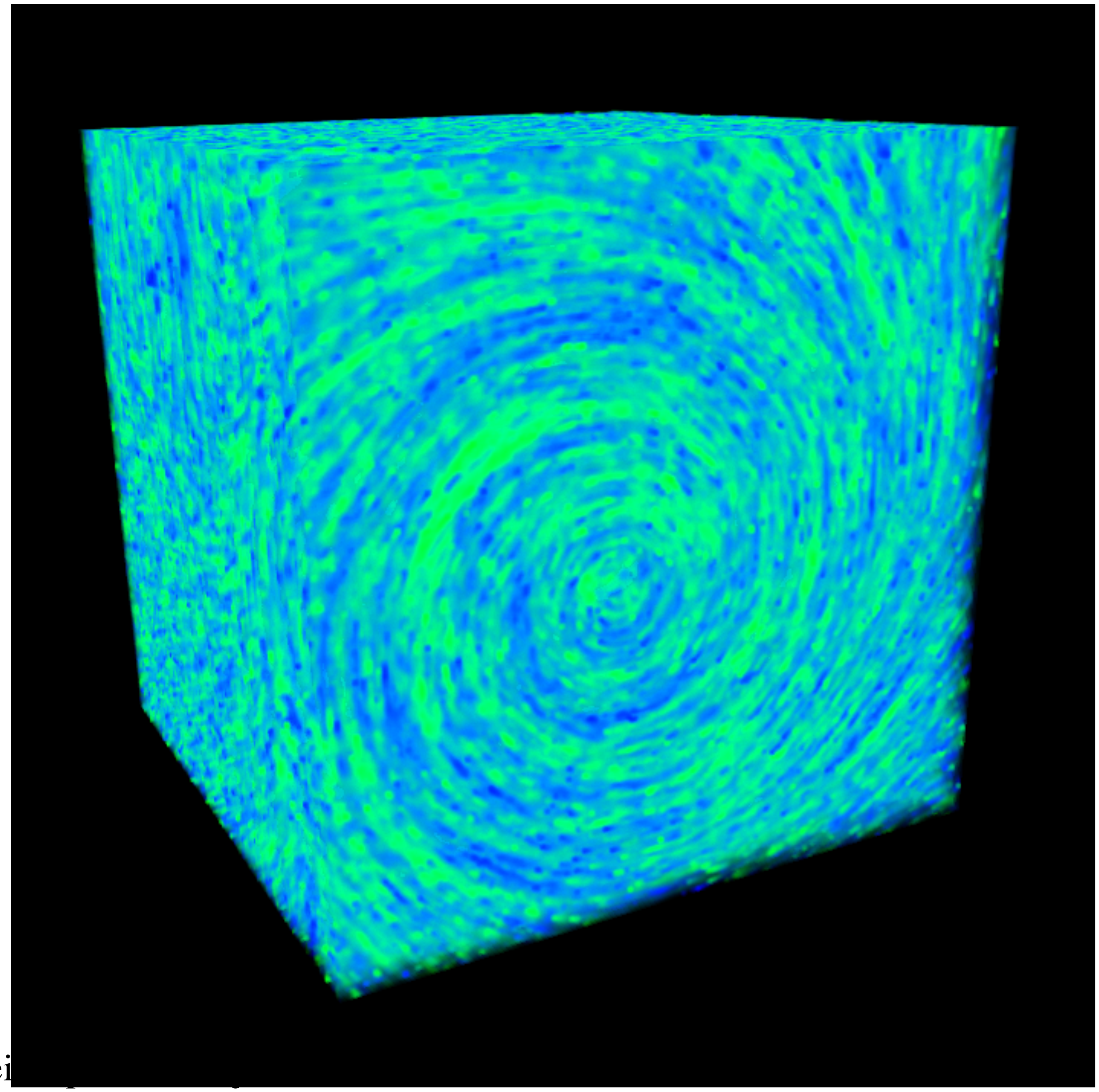
Vortex extraction with λ_2

3D Vector Fields

- Improving spatial perception:
  - Depth cues
    - Perspective
    - Occlusion
    - Motion parallax
    - Stereo disparity
    - Color (atmospheric, fogging)
  - Halos
  - Orientation of structures by shading (highlights)

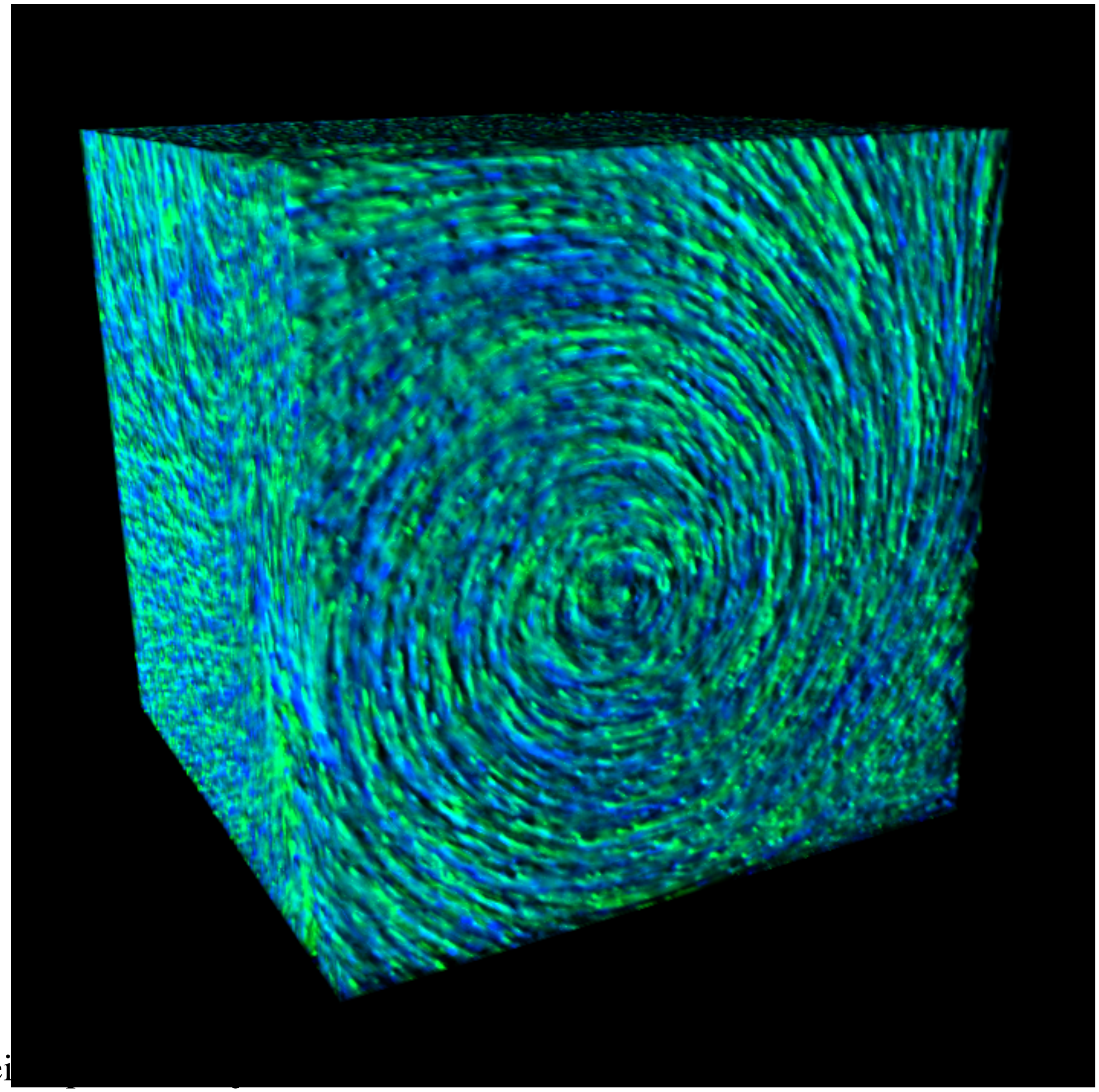
3D Vector Fields

- No illumination



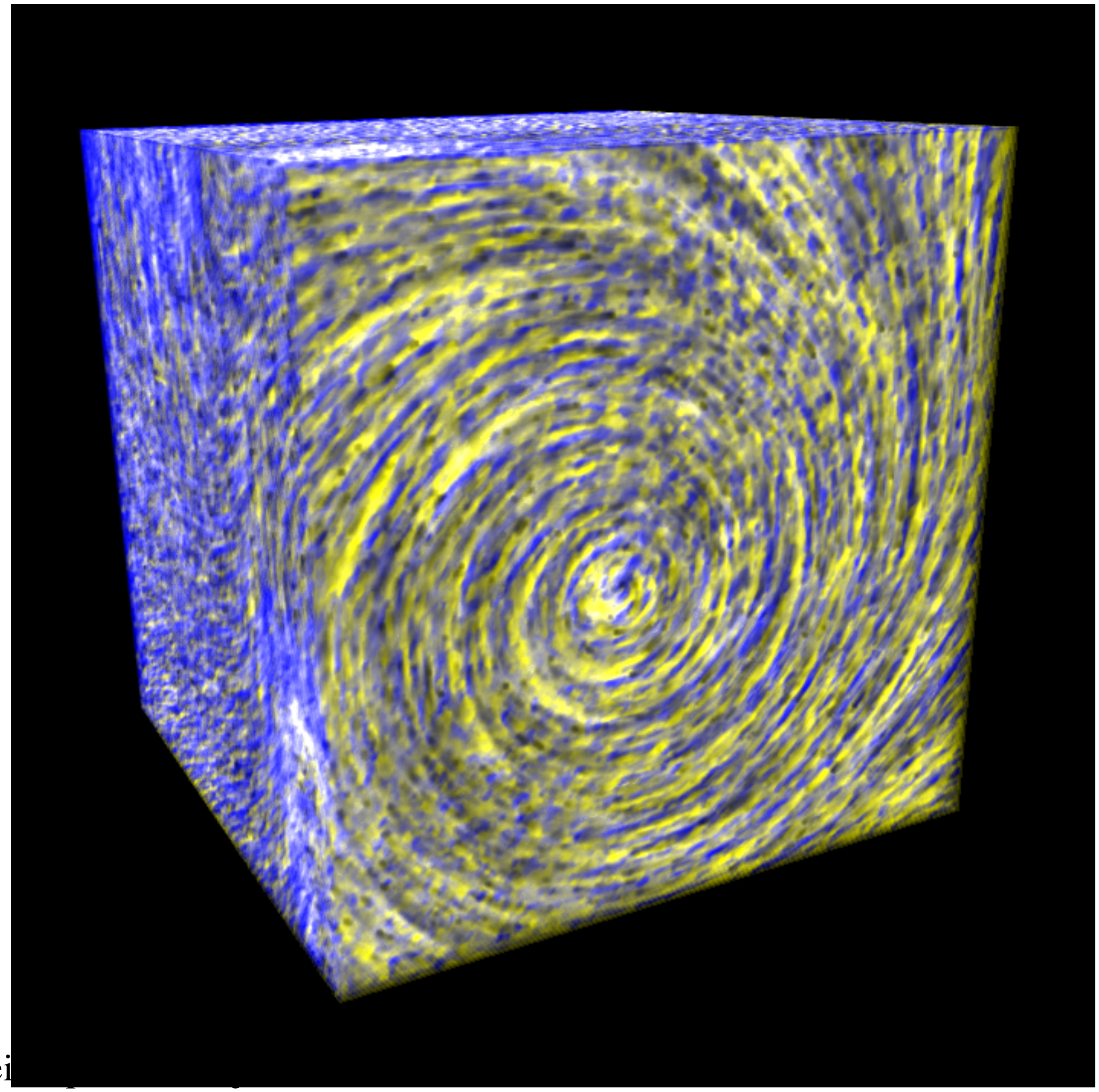
3D Vector Fields

- Phong illumination



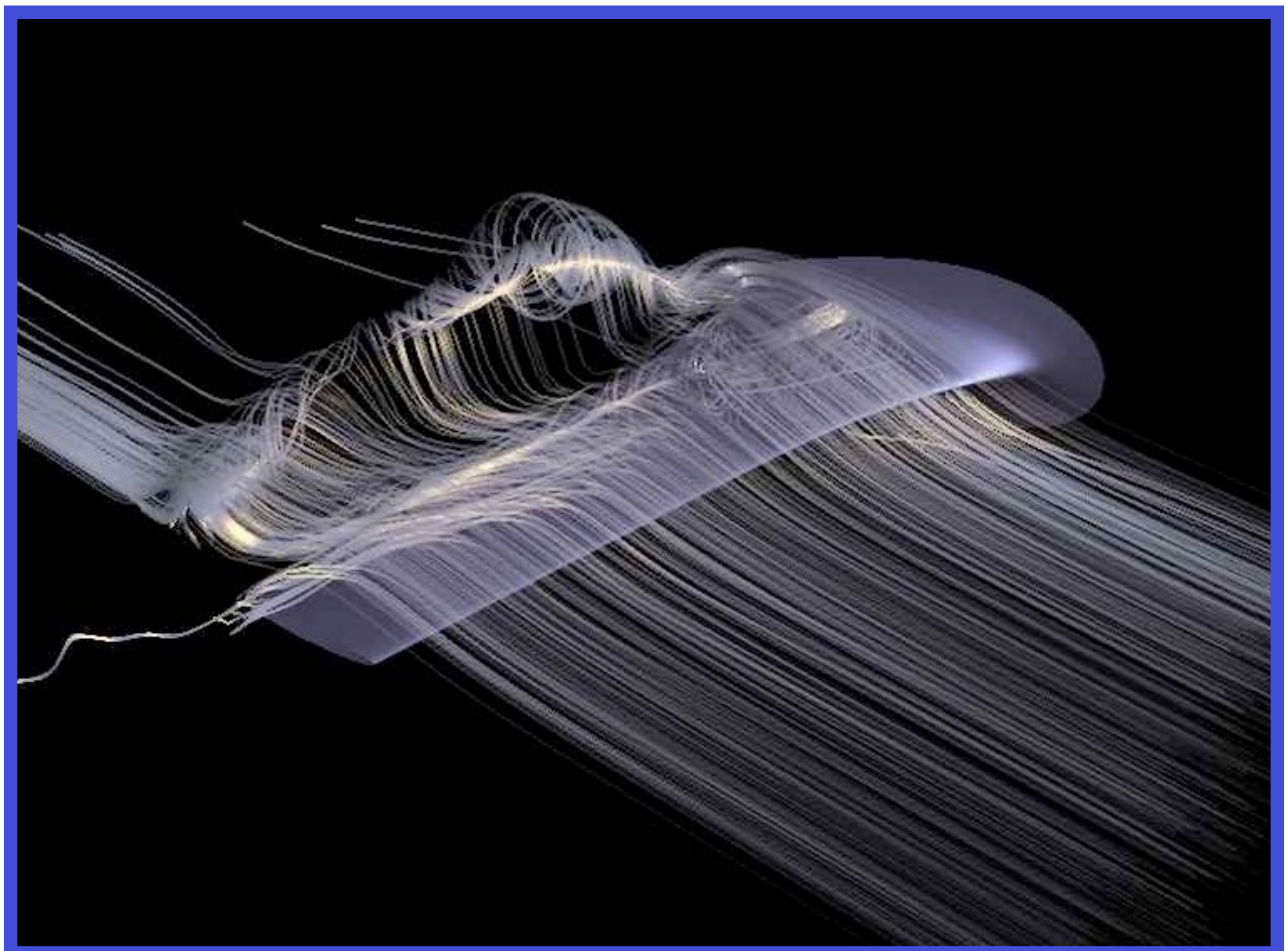
3D Vector Fields

- Cool/warm



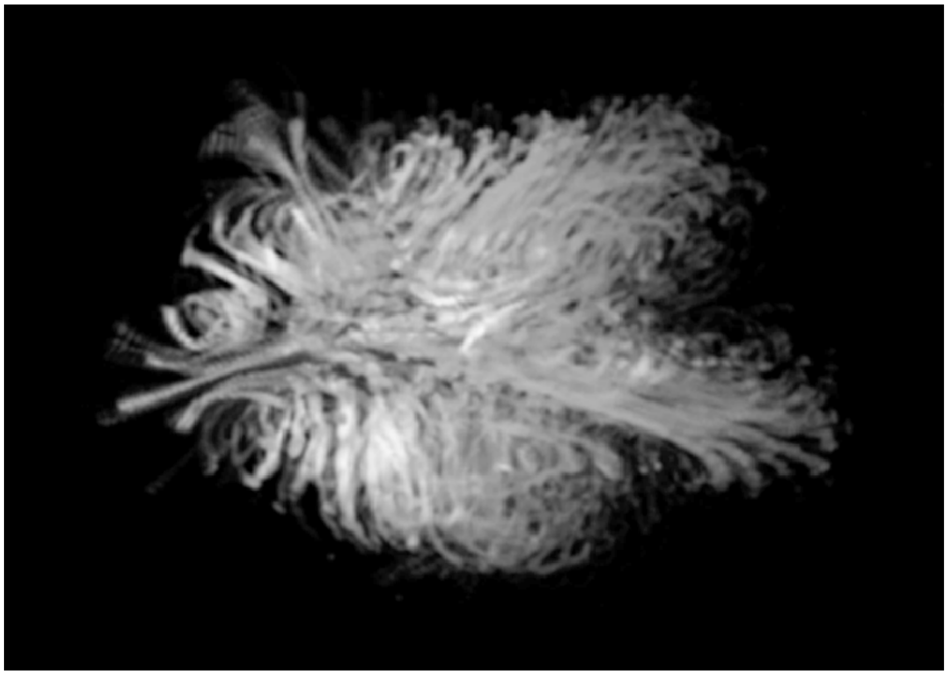
3D Vector Fields

- Illuminated streamlines

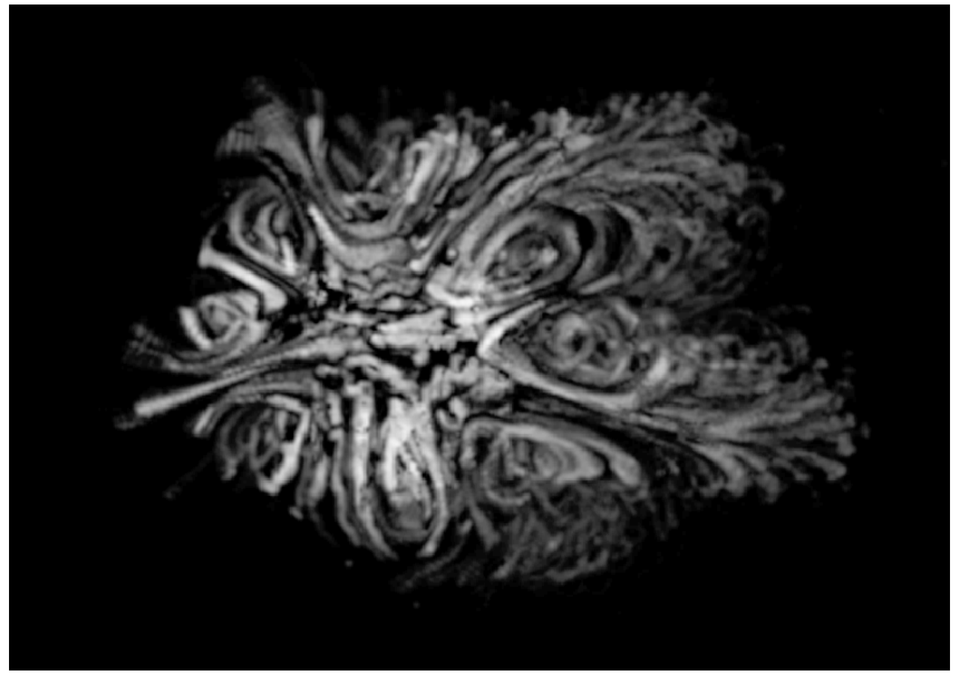


3D Vector Fields

- Halos



Without halos



With halos