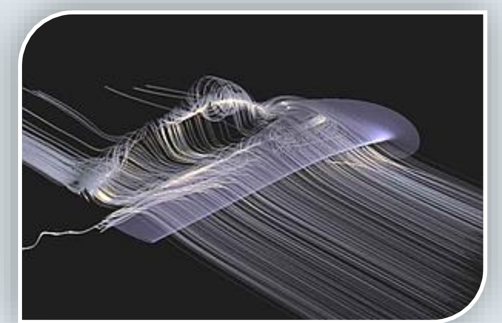



Visual Data Analytics
Flow Visualization II

Dr. Johannes Kehrer

Disclaimer

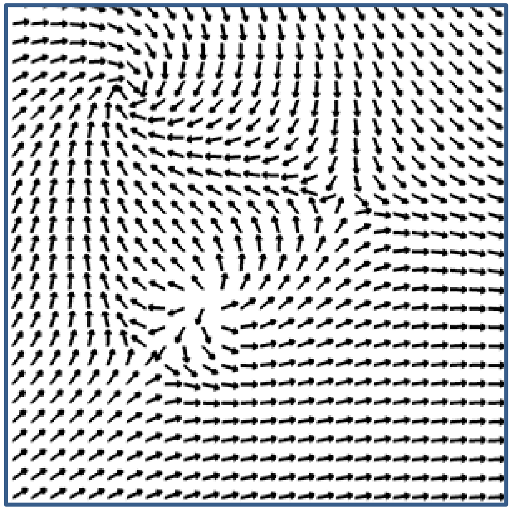
These lecture slides and materials are for personal use by students and are intended for educational purposes only. Distribution or reproduction of the material is prohibited.

The views and opinions expressed by any individual during the course of these lecturers are their own and do not necessarily represent the views, opinions, or positions of Siemens or the Technical University of Munich.

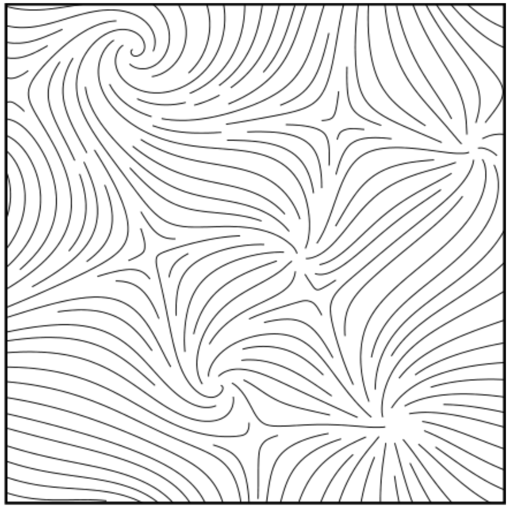
Although reasonable efforts have been made to ensure the information in these lecture slides is current and accurate, we do not assume any responsibility for the content of the posted material.

These slides also contain links to websites of third parties. As the content of these websites is not under our control, we cannot assume any liability for such external content.

Flow visualization – Approaches



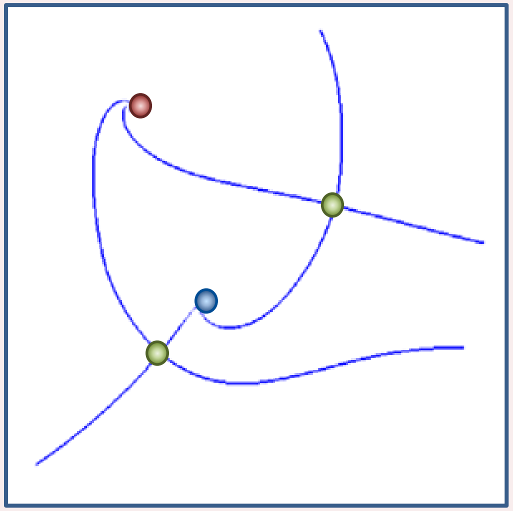
Direct flow visualization (arrows, color coding, …)

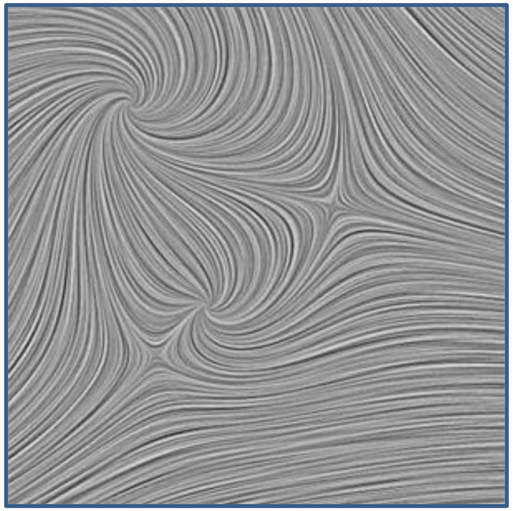


Geometric flow vis. (stream lines/surfaces, …)

Sparse (feature-based) vis.

- Global computation of flow features
- Vortices, shockwaves, vector field topology





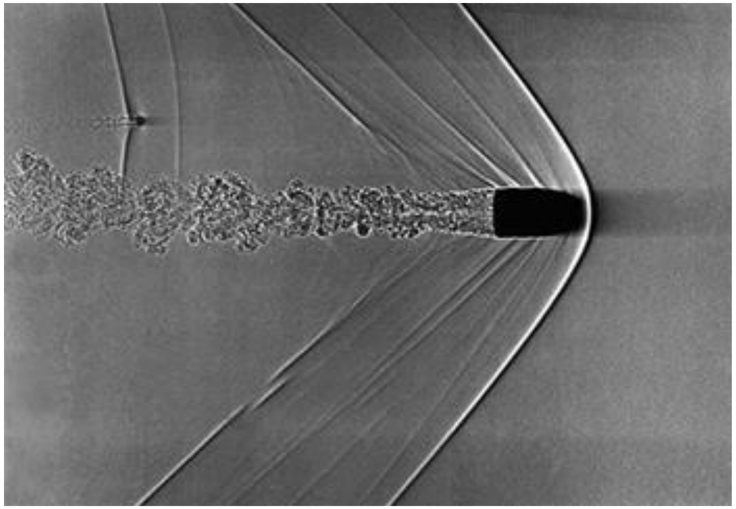
Dense (texture-based)

Flow features

SIEMENS
Ingenuity for life

- Vortices
  - One of most prominent features
  - Important in many applications (turbulent flows)
  - No formal, well accepted definition yet ("something swirling")

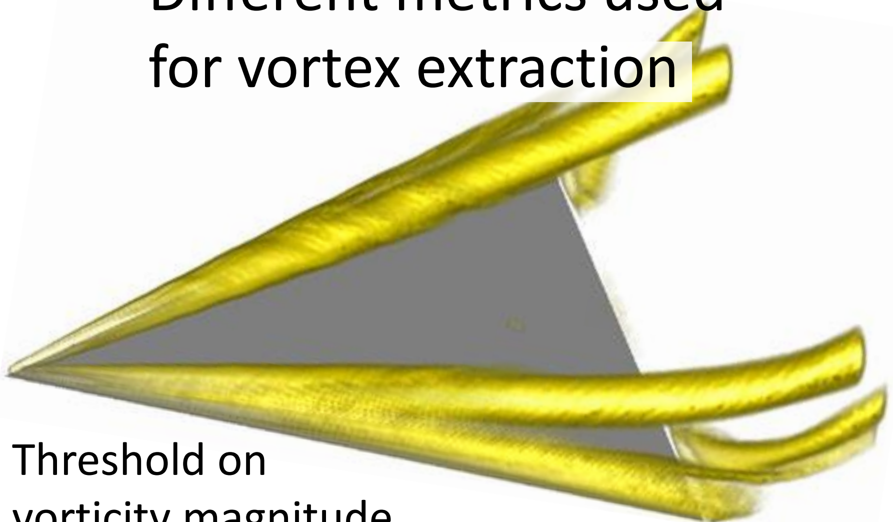
- Shock waves
  - Characterized by sharp discontinuities in flow attributes (pressure, velocity magnitude, …)



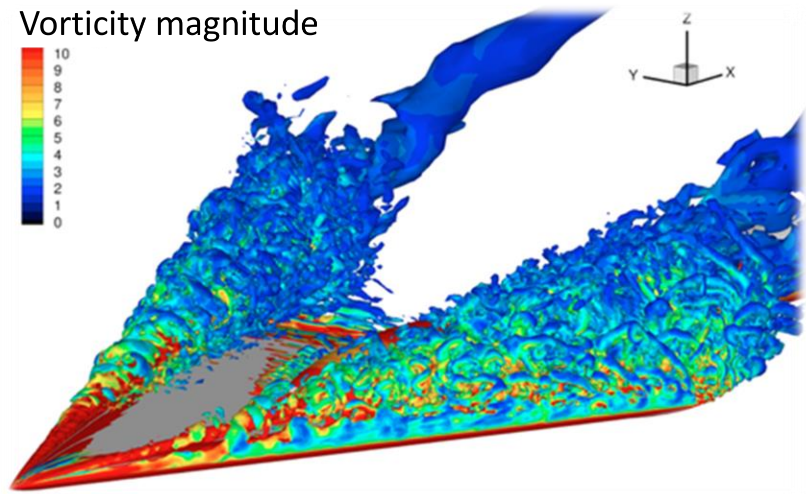
Bullet traveling through air at about 1.5 times sound speed

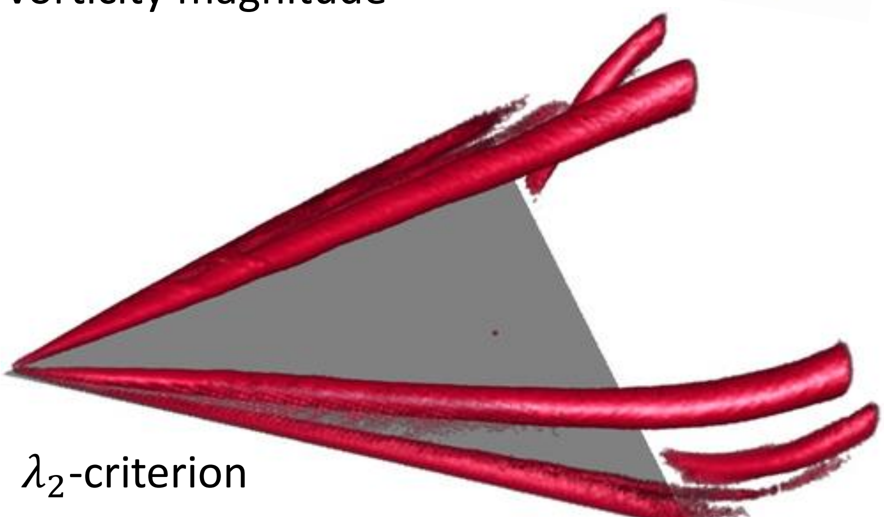
Flow features

- Vortices around delta wing
  - Different metrics used for vortex extraction

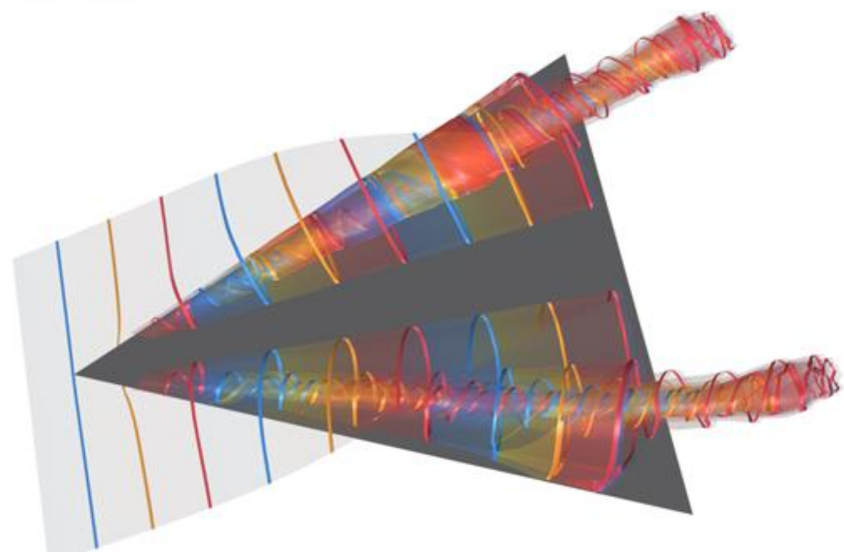


Threshold on vorticity magnitude





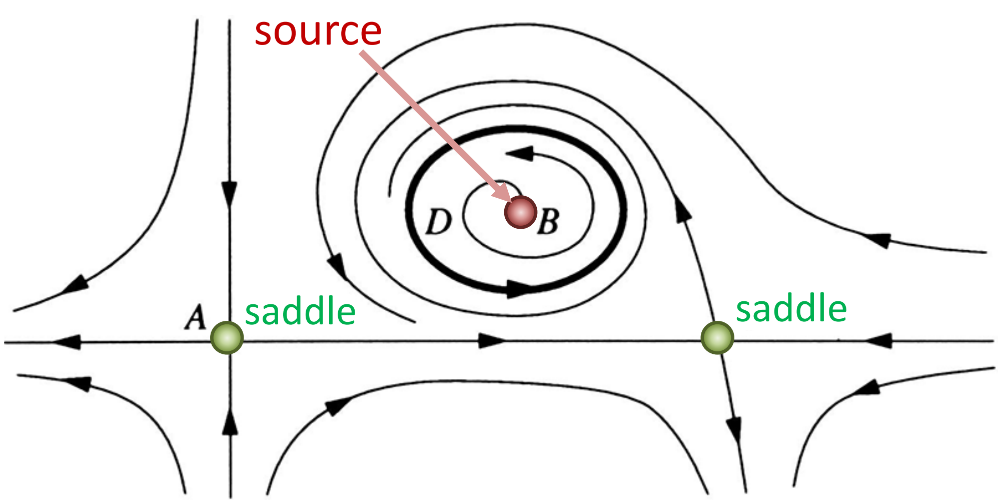
λ_2-criterion

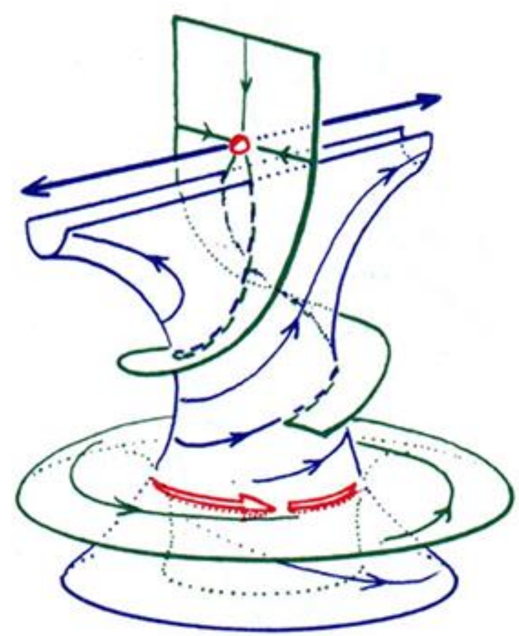


Vector field topology

SIEMENS
Ingenuity for life

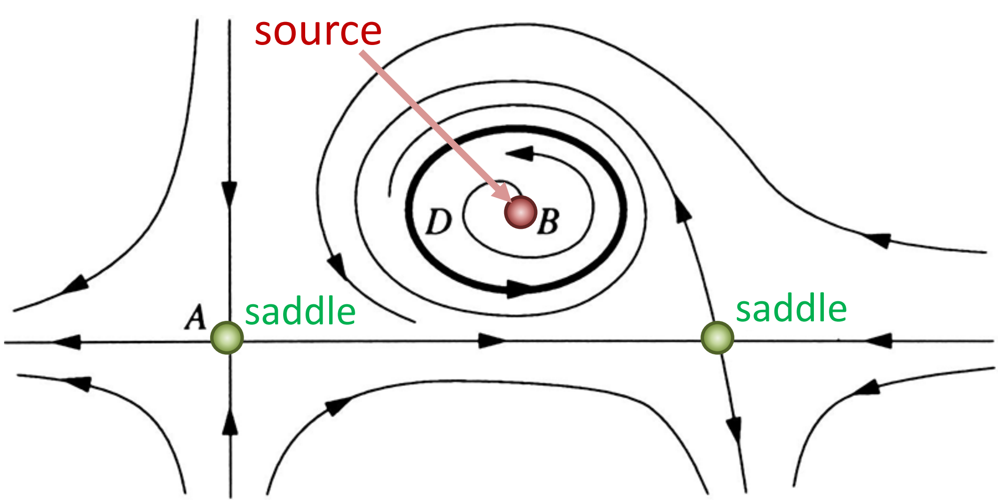
- Idea: Do not draw "all" stream lines, but only the "important" ones
- Show only topological skeleton
  - Connection of critical points
  - Characterization of global flow structures

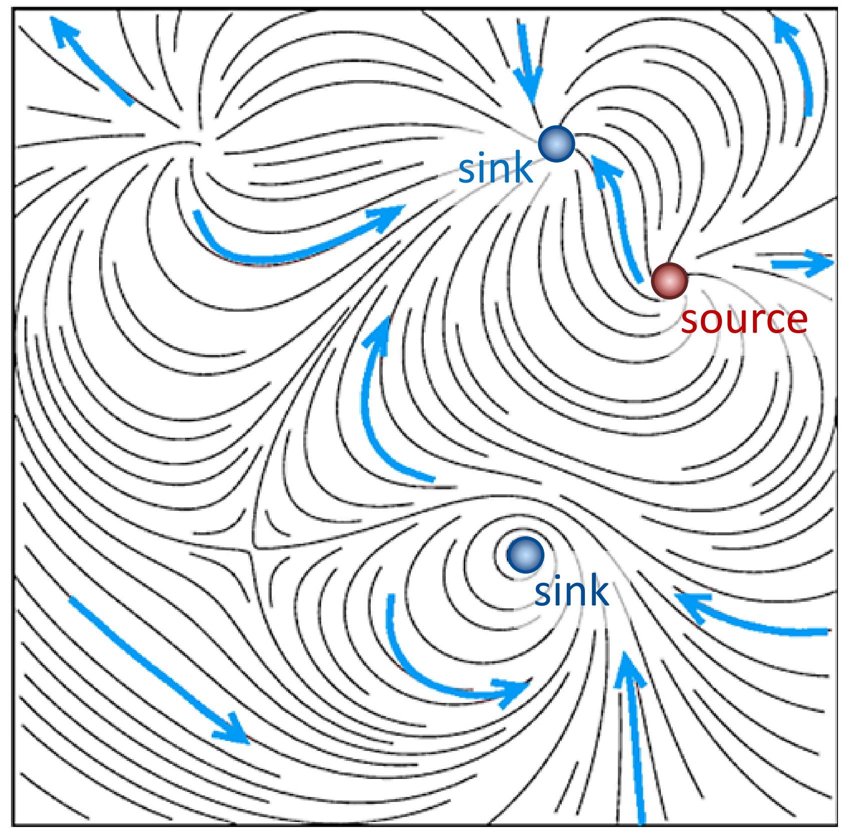




Vector field topology

- Critical points: singularities in vector field such that $\boldsymbol{v}(\boldsymbol{x}^*) = 0$
  - Points where magnitude of vector goes to zero and direction of vector is undefined
  - Stream lines reduced to single point

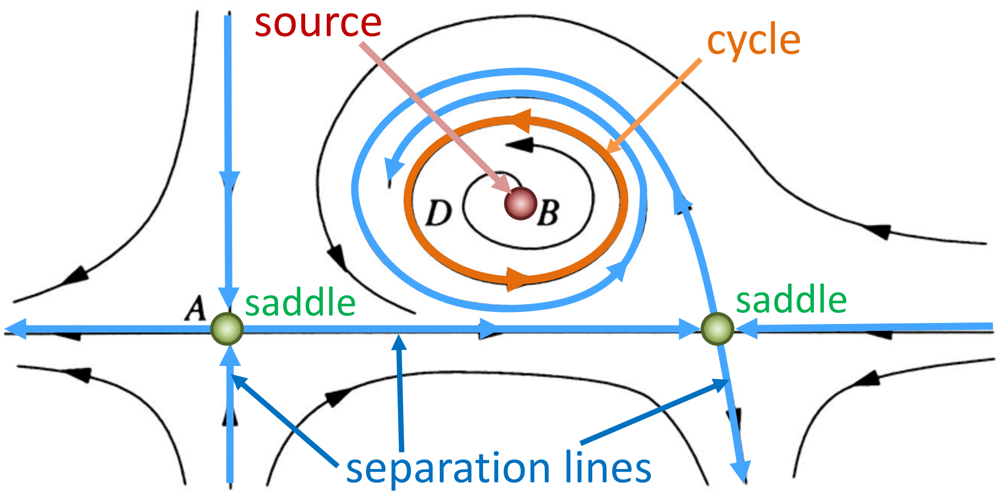


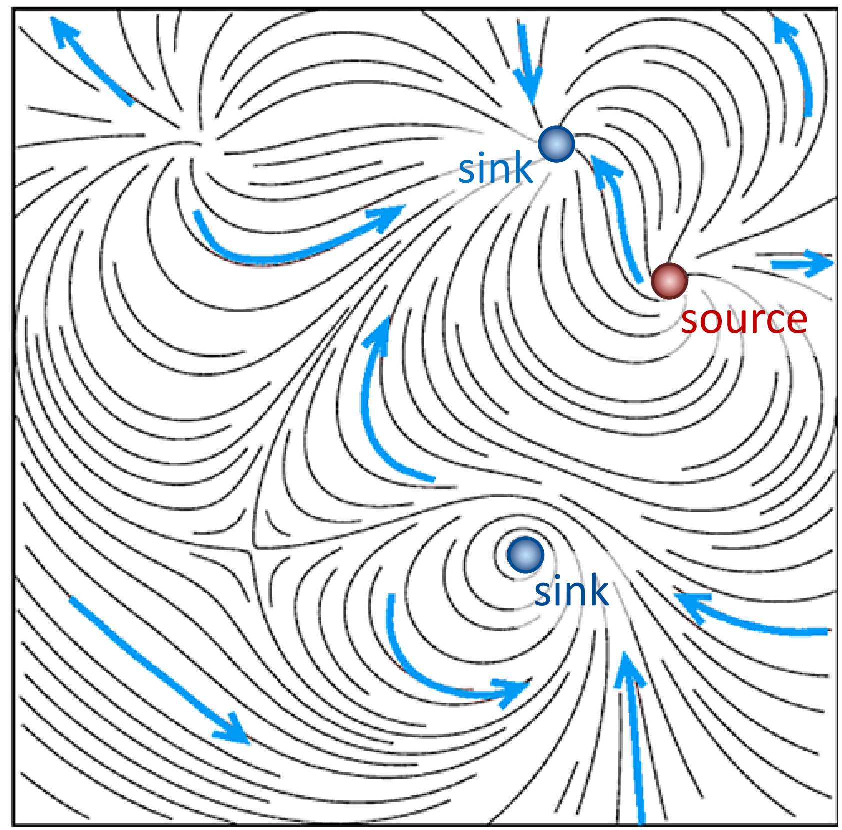


Vector field topology

SIEMENS
Ingenuity for life

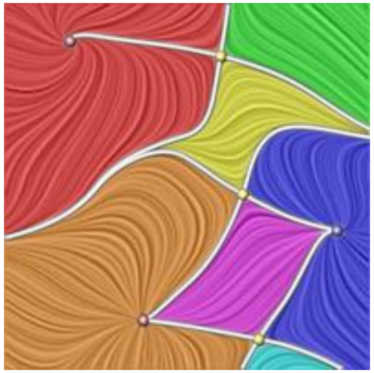
- Critical points: singularities in vector field such that $\boldsymbol{v}(\boldsymbol{x}^*) = 0$
  - Points where magnitude of vector goes to zero and direction of vector is undefined
  - Stream lines reduced to single point
  - Type of critical point determines flow pattern around it

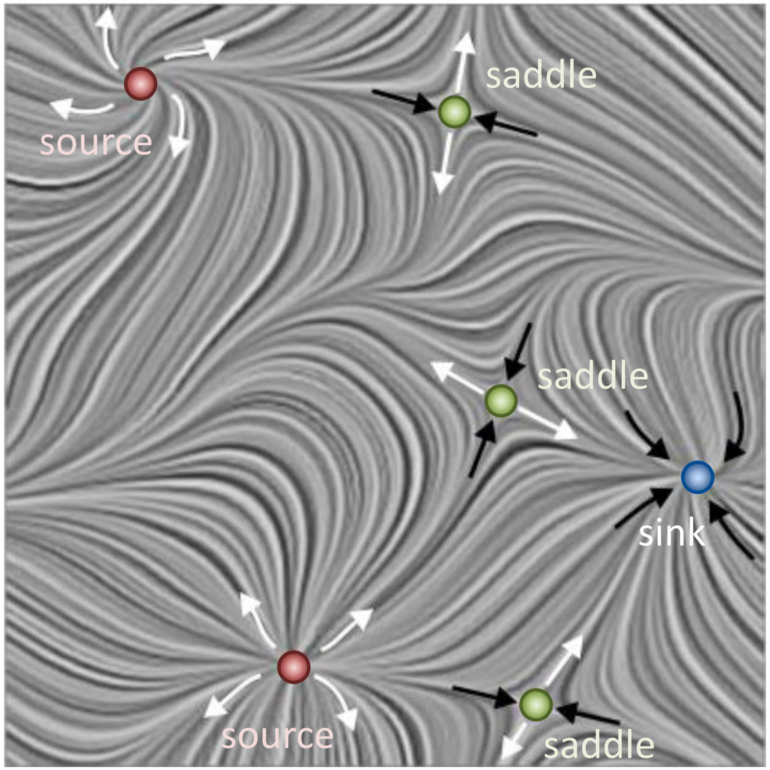




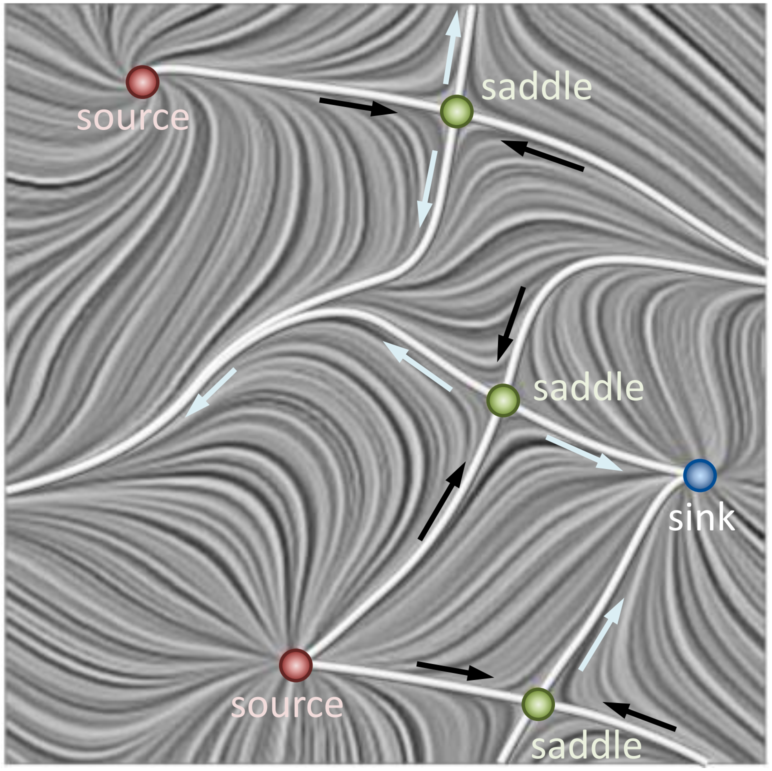
Vector field topology

- Topological skeleton / graph
  - Nodes: critical points
  - Edges: separation lines and cycles
  - Flow divided into regions with similar properties





Critical points



Critical points + separation lines

[Weinkauf]

Vector field topology (2D)

SIEMENS
Ingenuity for life

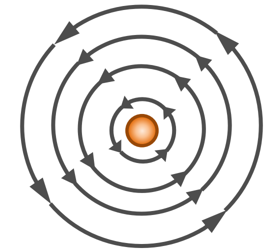
- How to find critical points $\boldsymbol{x}^*$?
  - Points where $\boldsymbol{v}(\boldsymbol{x}^*) = 0$
- How to classify critical points $\boldsymbol{x}^*$?
  - Jacobian matrix $\mathbf{J} = \begin{pmatrix} \frac{\partial}{\partial x}\boldsymbol{v}_x & \frac{\partial}{\partial y}\boldsymbol{v}_x \\ \frac{\partial}{\partial x}\boldsymbol{v}_y & \frac{\partial}{\partial y}\boldsymbol{v}_y \end{pmatrix}$ governs the behavior near $\boldsymbol{x}^*$
  - For each $\boldsymbol{x}^*$, calculate eigenvalues λ_1, λ_2 of $\mathbf{J}$

$$\mathbf{J}\boldsymbol{u} = \lambda\boldsymbol{u}$$

$\mathbf{J}$ ... Jacobian matrix
$\boldsymbol{u}$ ... eigenvector (non-zero)
λ ... eigenvalue

Vector field topology (2D)

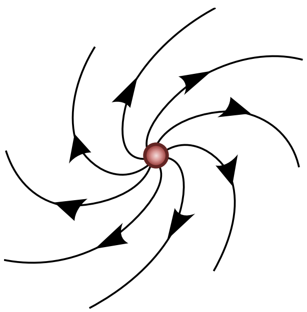
- Classify critical points by eigenvalue analysis of $\mathbf{J}$
  - λ_1, λ_2 both positive → local repulsion (source)
  - λ_1, λ_2 both negative → local attraction (sink)
  - $\lambda_1 \lambda_2 < 0$ → saddle point
  - λ_1, λ_2 both complex → rotation around $\boldsymbol{x}^*$



Center

$\mathrm{Im}(\lambda_{1,2}) \neq 0$

$\mathrm{Re}(\lambda_{1,2}) = 0$

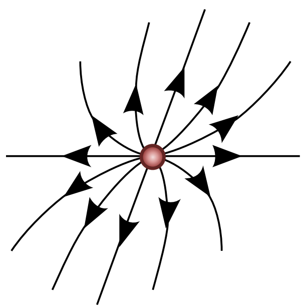


Circulating source
(repelling focus)

$\mathrm{Im}(\lambda_{1,2}) \neq 0$

$\mathrm{Re}(\lambda_{1,2}) > 0$

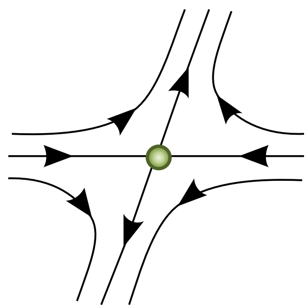
$\lambda_{1,2} = a \pm bi$



Noncirculating source
(repelling node)

$\mathrm{Im}(\lambda_{1,2}) = 0$

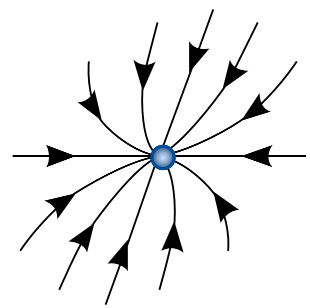
$\mathrm{Re}(\lambda_{1,2}) > 0$



Saddle point

$\mathrm{Im}(\lambda_{1,2}) = 0$

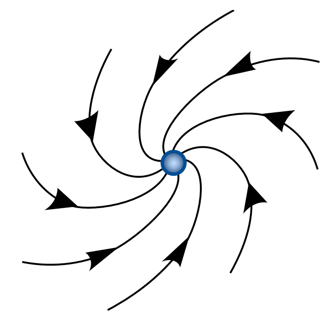
$\lambda_1 \lambda_2 < 0$



Noncirculating sink
(attracting node)

$\mathrm{Im}(\lambda_{1,2}) = 0$

$\mathrm{Re}(\lambda_{1,2}) < 0$



Circulating sink
(attracting focus)

$\mathrm{Im}(\lambda_{1,2}) \neq 0$

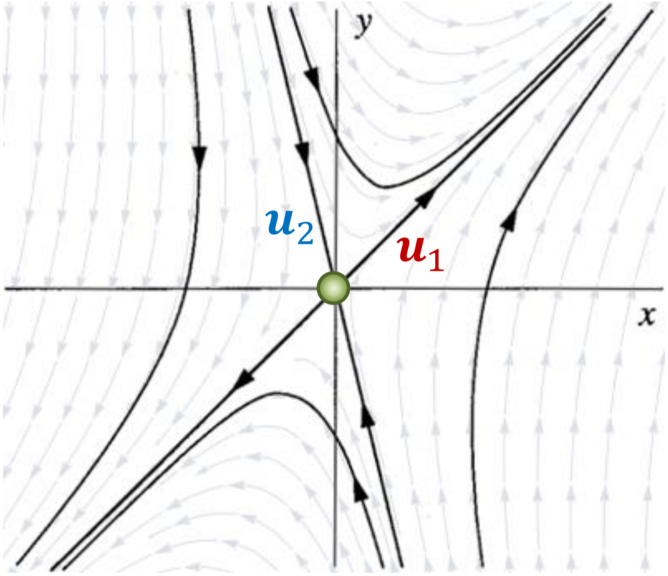
$\mathrm{Re}(\lambda_{1,2}) < 0$

$\lambda_{1,2} = a \pm bi$

Vector field topology (2D)

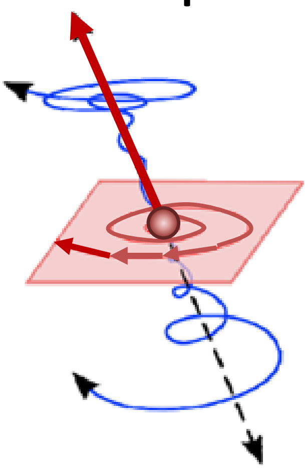
SIEMENS
Ingenuity for life

- Mapping to graphical primitives: streamlines
  - Start streamlines close to critical points
  - Initial direction along the eigenvectors $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ (forward+backward integration)
- End particle tracing at
  - Other "real" critical points
  - Interior boundaries
  - Boundaries of computational domain



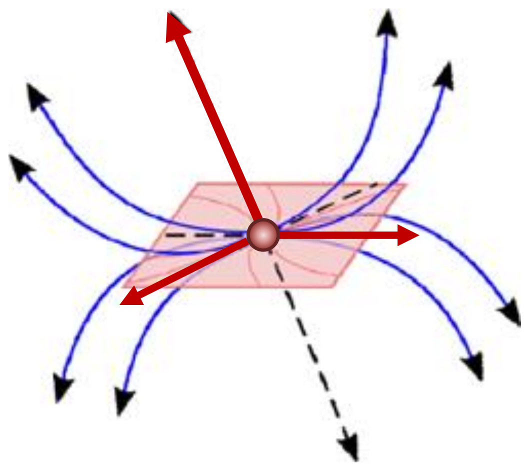
Vector field topology (3D)

- Critical points in 3D
  - More complicated
  - Line and surface separatices exist
- Examples



Spiral source

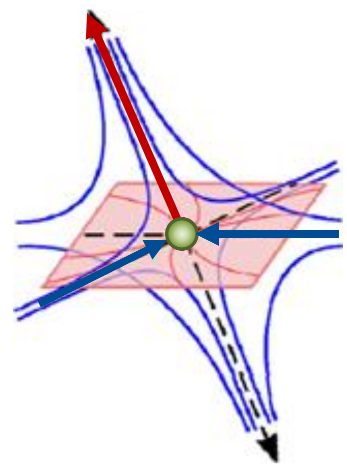
$\mathrm{Im}(\lambda_1) = 0,$
$\mathrm{Im}(\lambda_{2,3}) \neq 0$
$\mathrm{Re}(\lambda_{1,2,3}) > 0$



Noncirculating source

$\mathrm{Im}(\lambda_{1,2,3}) = 0$

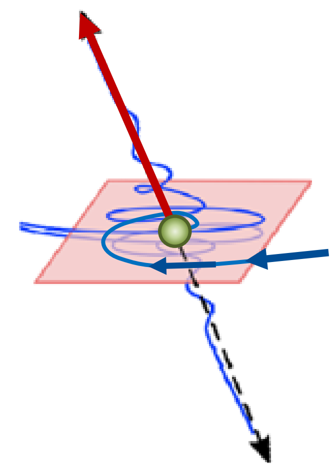
$\mathrm{Re}(\lambda_{1,2,3}) > 0$



Noncirculating saddle

$\mathrm{Im}(\lambda_{1,2,3}) = 0$

$\mathrm{Re}(\lambda_1) > 0,$
$\mathrm{Re}(\lambda_{2,3}) < 0$



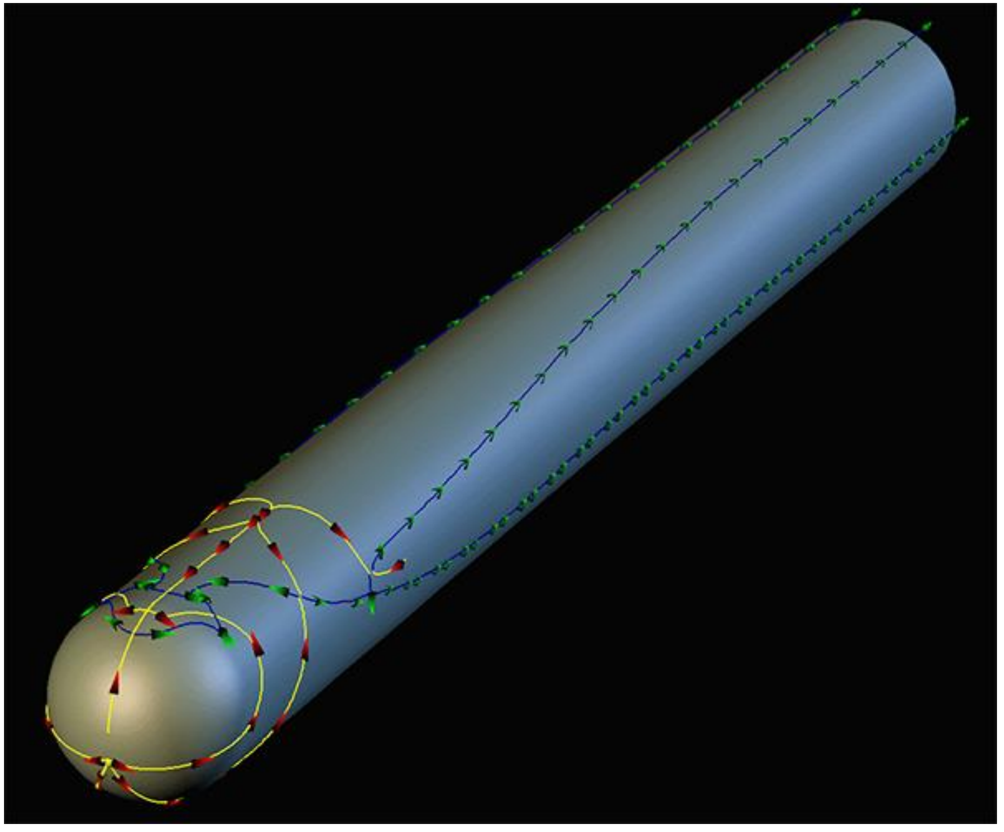
Spiral saddle

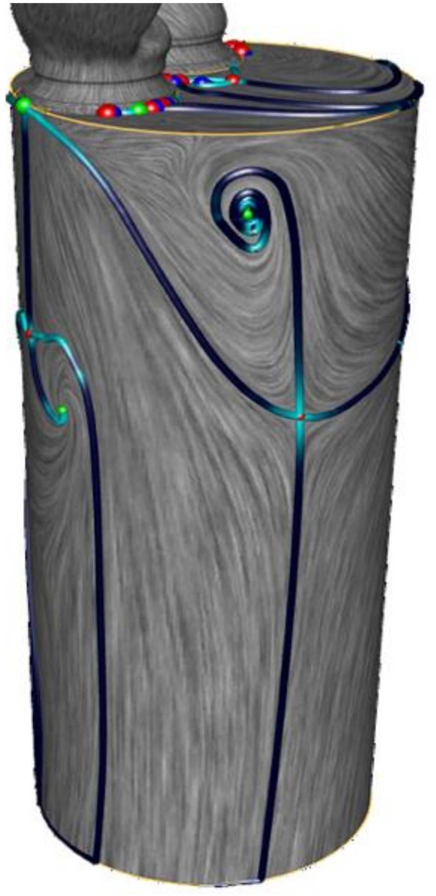
$\mathrm{Im}(\lambda_1) = 0,$
$\mathrm{Im}(\lambda_{2,3}) \neq 0$
$\mathrm{Re}(\lambda_1) > 0,$
$\mathrm{Re}(\lambda_{2,3}) < 0$

[Peikert & Sadlo 10]

Vector field topology (3D)

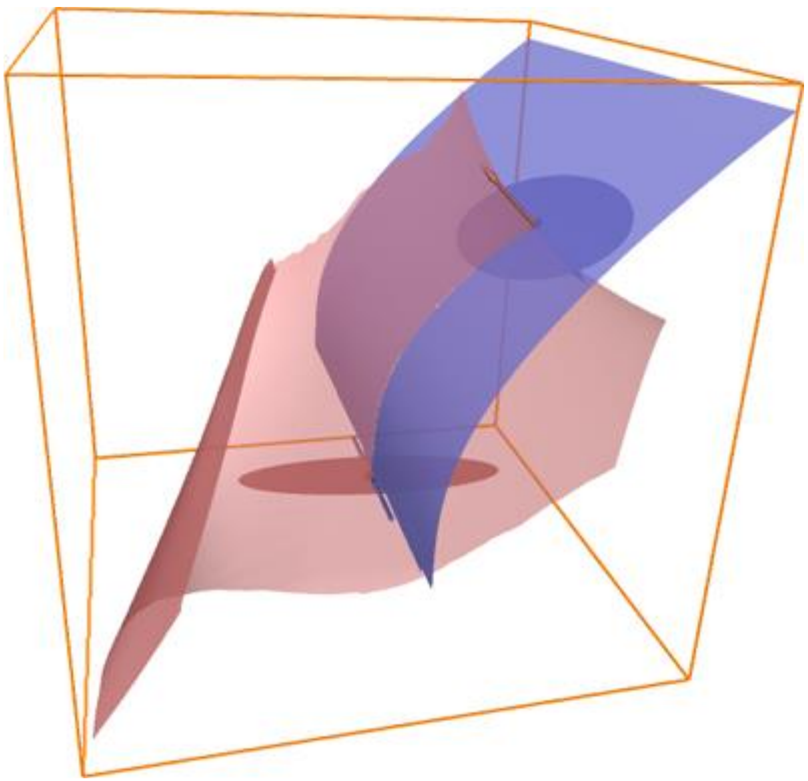
- Topology on surfaces
  - Critical points + separation lines applied to projections of vector field onto polygonal surface



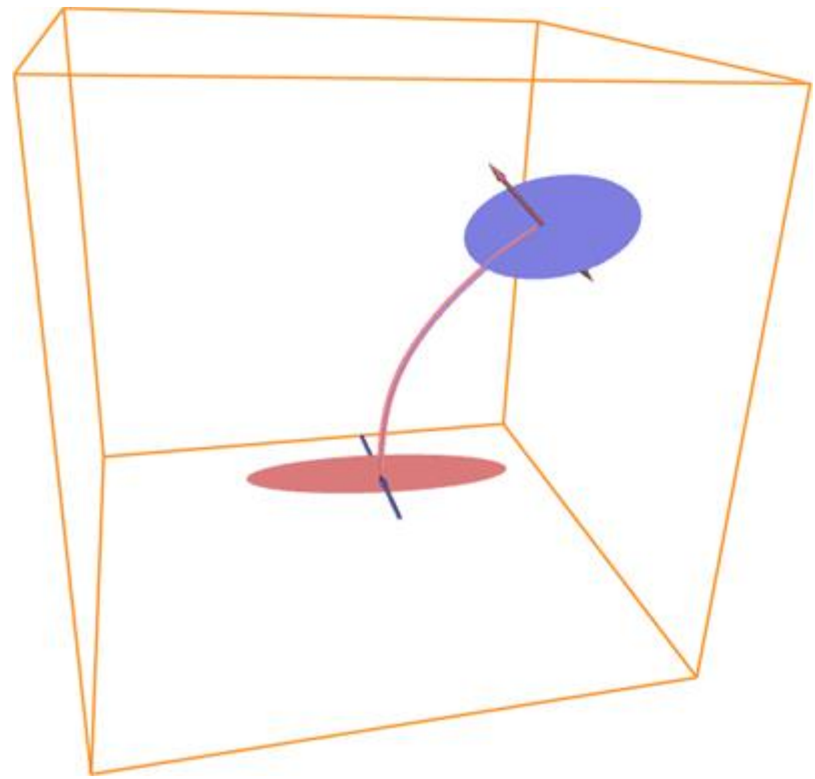


Vector field topology (3D)

- Saddle connectors in 3D



Separation surfaces of two saddles

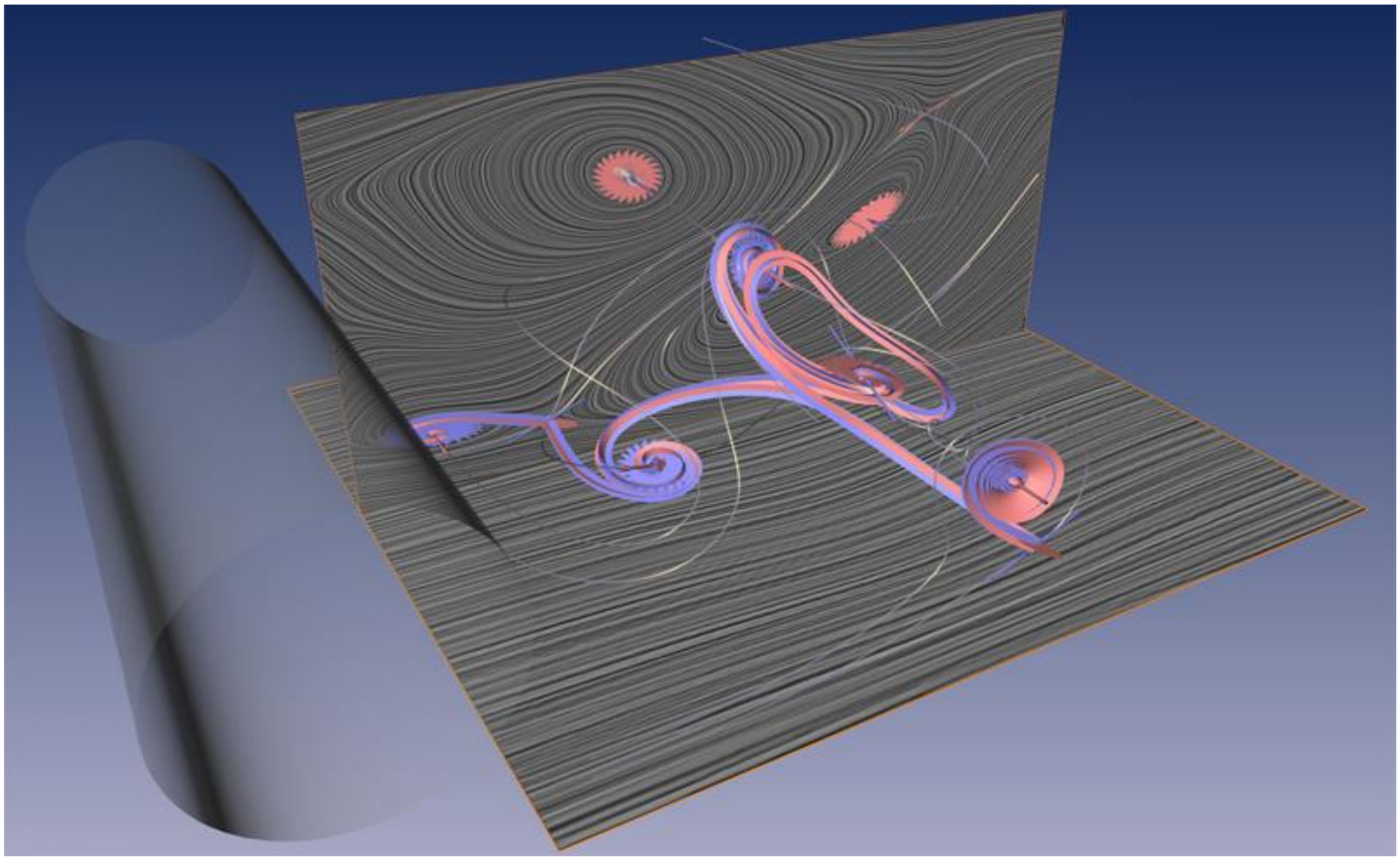


The intersection of the separation surfaces is the saddle connector

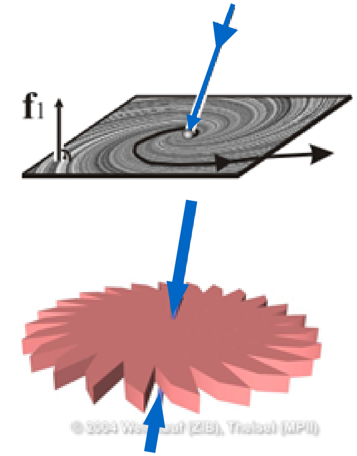
[Weinkauf & Theisel 05]

Vector field topology (3D)

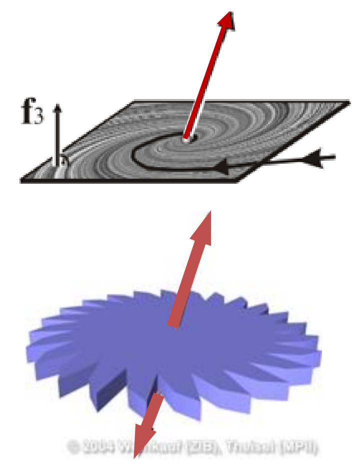
- Saddle connectors in 3D



[Weinkauf & Theisel 05]



Repelling spiral saddle

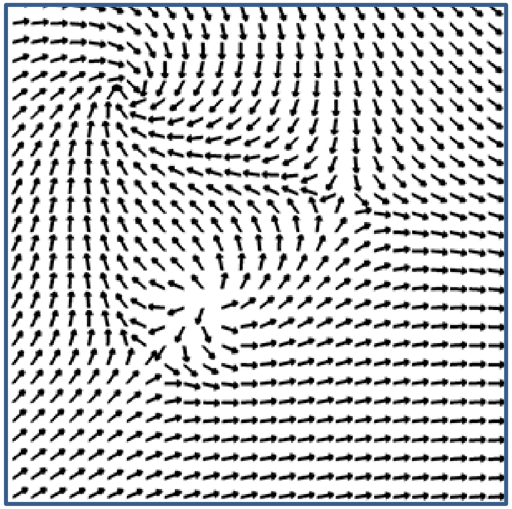


Attracting spiral saddle

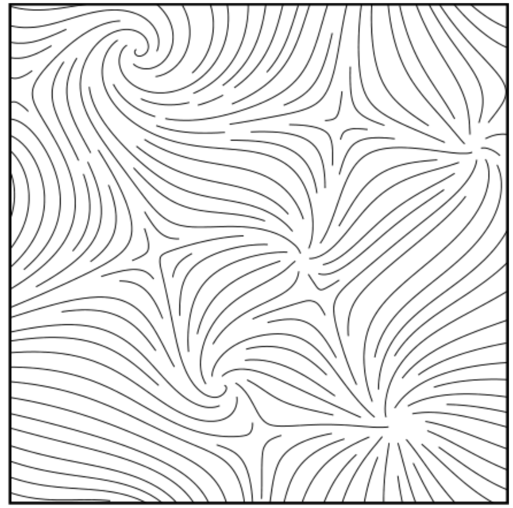
Vector field topology

- Summary
  - Draw only relevant stream lines (topological skeleton)
  - Partition domain into regions with similar flow features
  - Based on critical points
  - Good for 2D steady flows
  - Unsteady flows?
  - 3D?

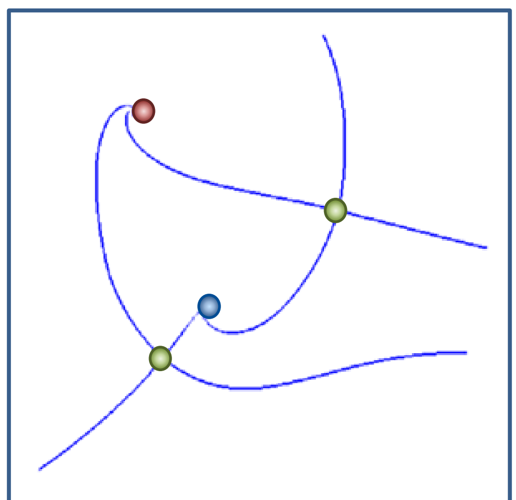
Flow visualization – Approaches



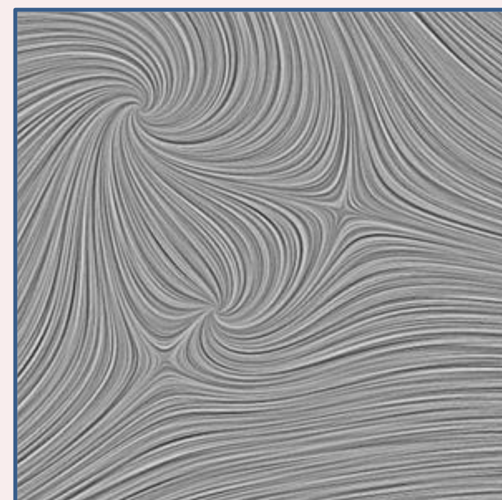
Direct flow visualization
(arrows, color coding, …)



Geometric flow vis.
(stream lines/surfaces, …)



Sparse (feature-based) vis.

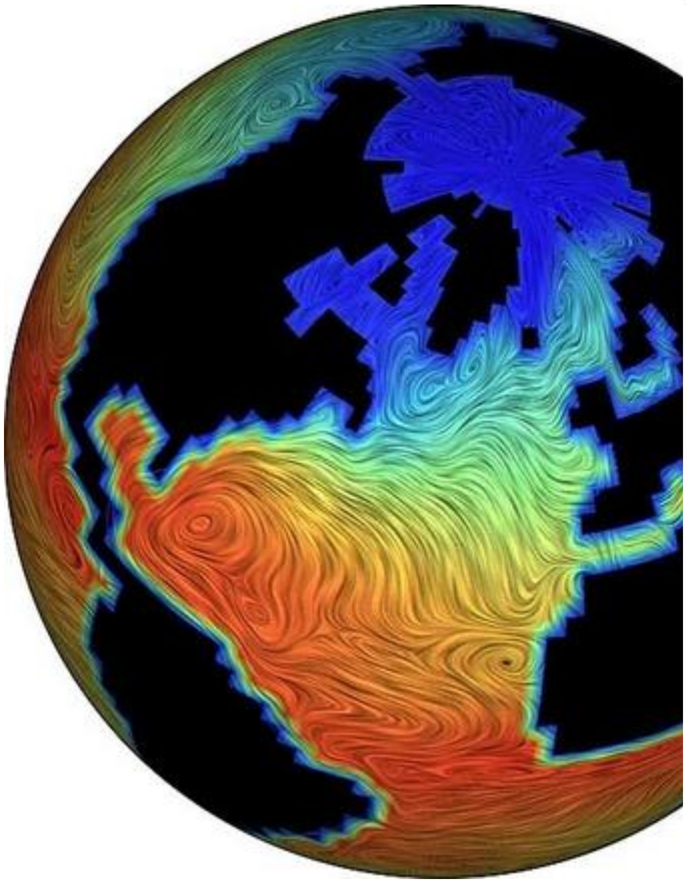
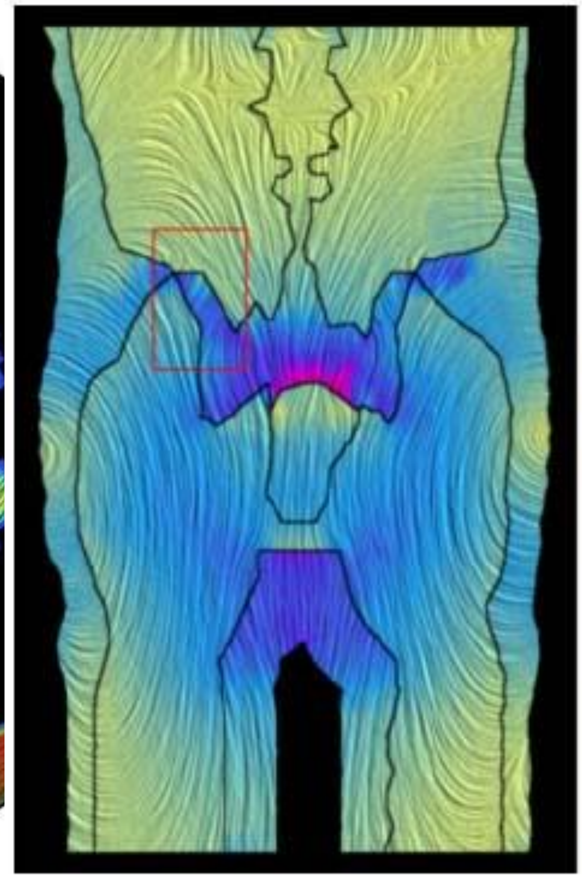


Dense (texture-based)

Texture-based flow visualization

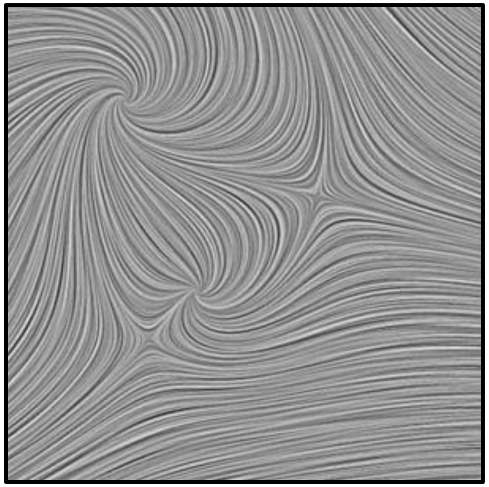
SIEMENS
Ingenuity for life

- Global method to visualize vector fields

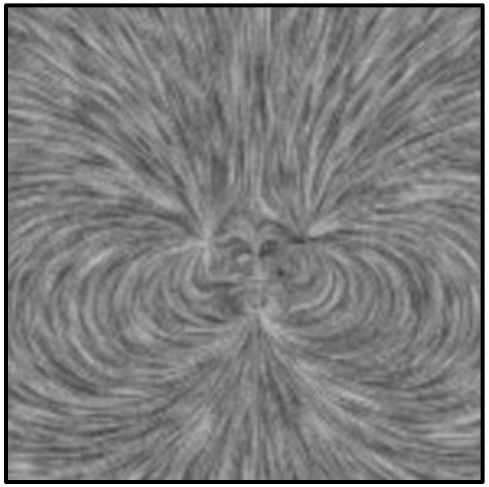



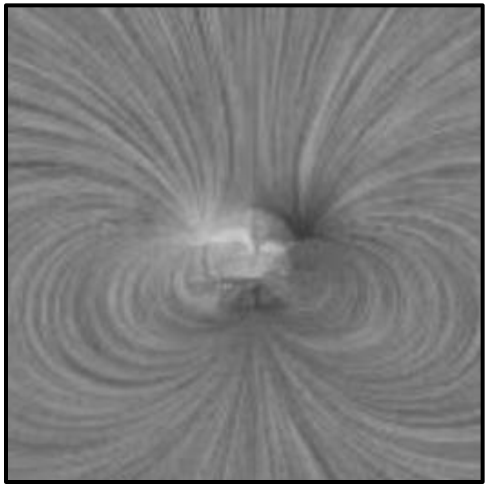

Why texture-based flow vis.

- Dense sampling
  - Better coverage of information
  - Critical point detection and classification
  - (Partially) solved problem of seeding



- Flexibility in visual representation
  - Good controllability of visual style
  - From line-like (crisp) to fuzzy

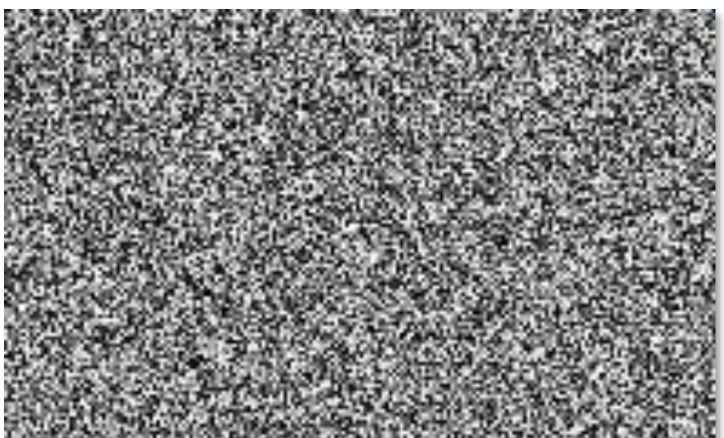




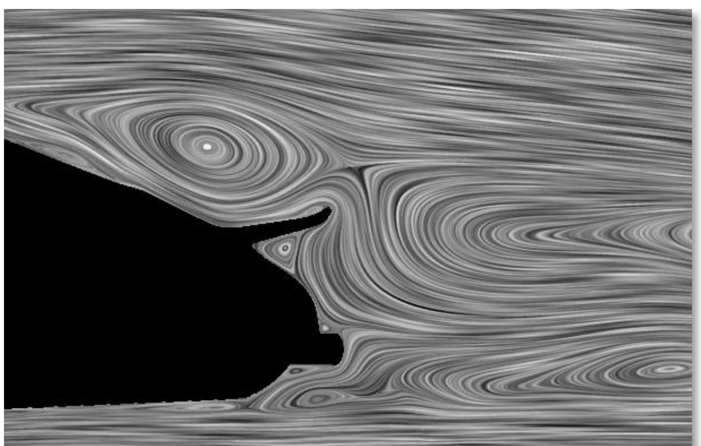
Line Integral Convolution

SIEMENS
Ingenuity for life

- Line Integral Convolution (LIC)
  - Global visualization technique (not only one particle path)
  - Start with a random texture (white noise)
  - Smear out the texture along trajectories of vector field
  - Results in low correlation between neighboring lines but high correlation along them (in flow direction)



White noise (no correlation)

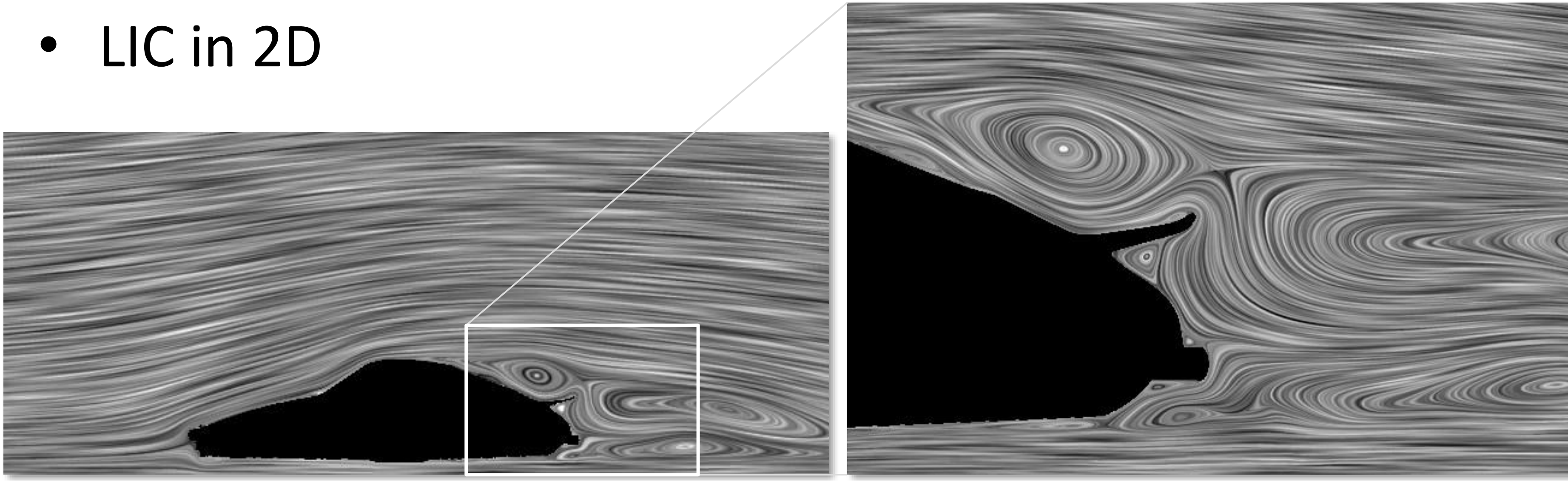


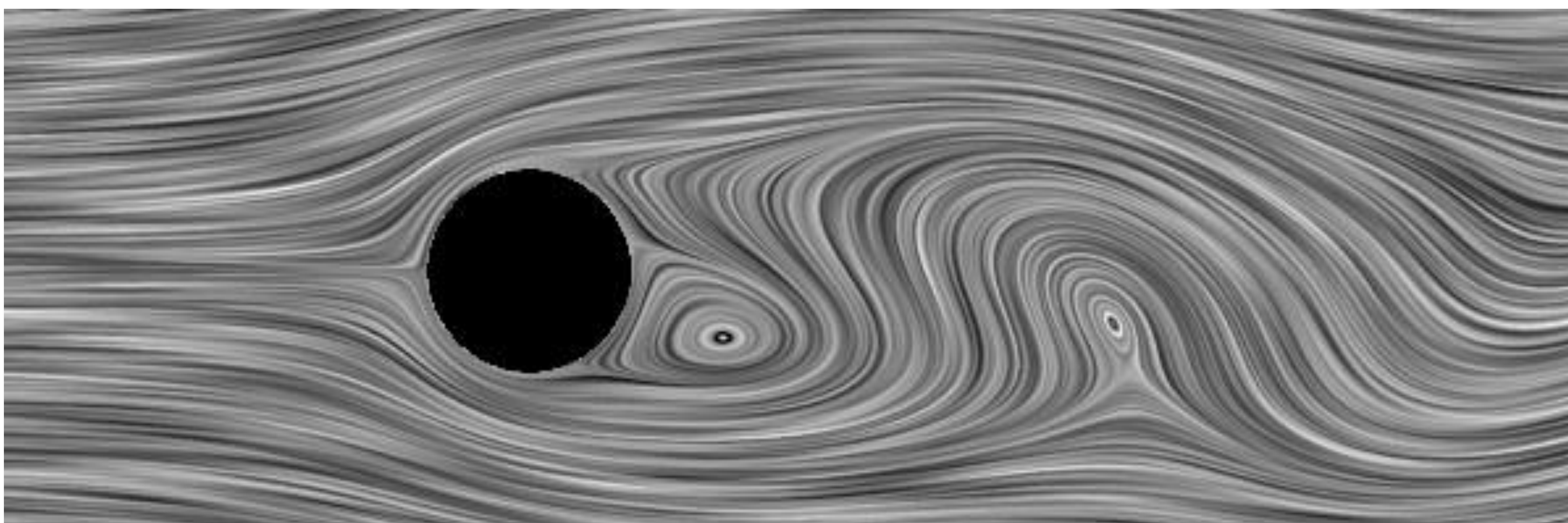
Texture values along trajectories are correlated (visually coherent)

[Cabral & Leedom 93]

Line Integral Convolution

- LIC in 2D

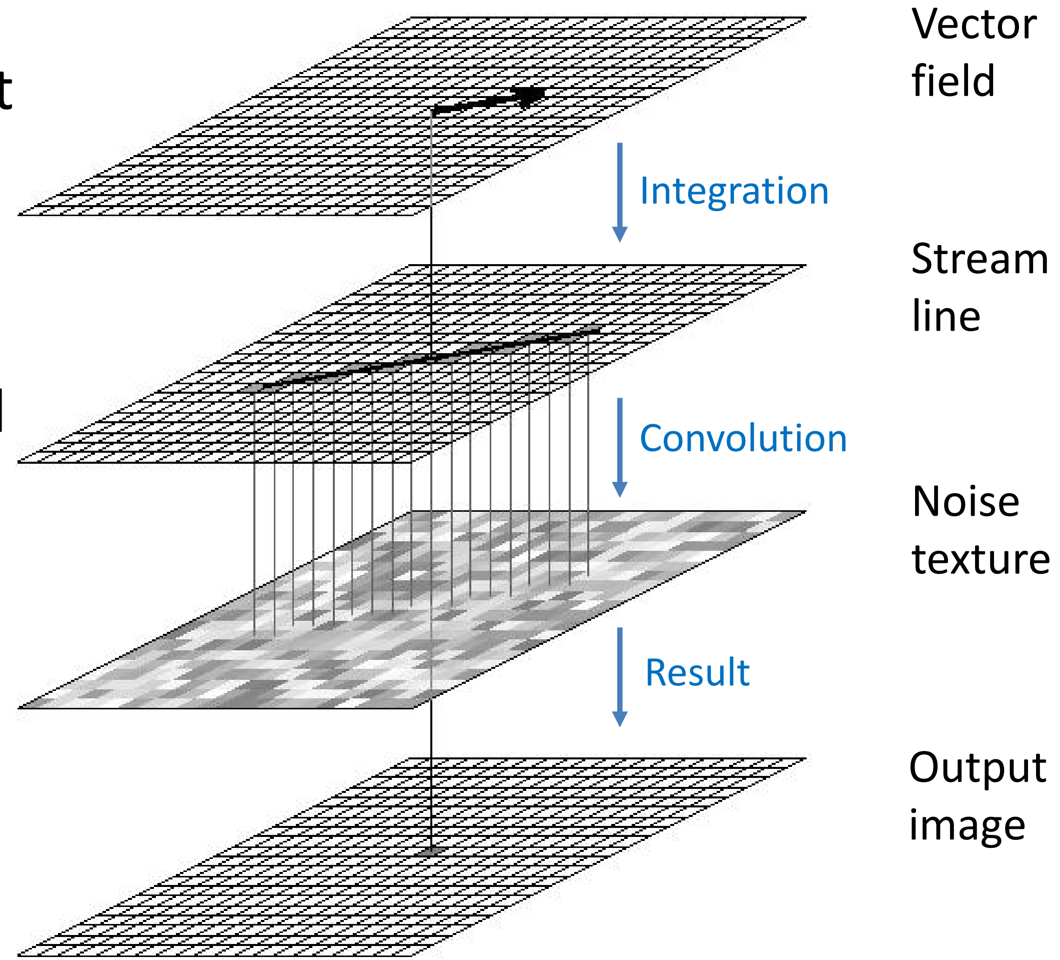




[Cabral & Leedom 93]

Line Integral Convolution

- Algorithm for 2D LIC
  - Look at stream line that passes through a pixel
  - Smear out - convolve - noise texture in direction of vector field (along stream line)



[Cabral & Leedom 93]

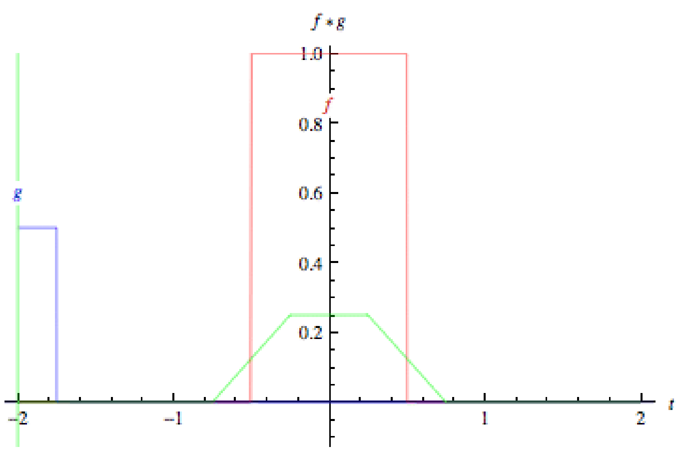
Filtering by convolution

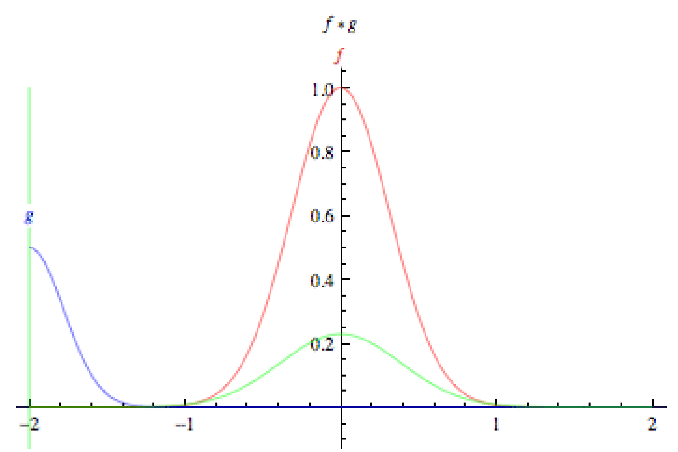
SIEMENS
Ingenuity for life

- Sliding a function $g(x)$ along a function $f(x)$

$$s(x') = [f * g](x') = \int_{-\infty}^{\infty} f(x)g(x' - x)dx$$

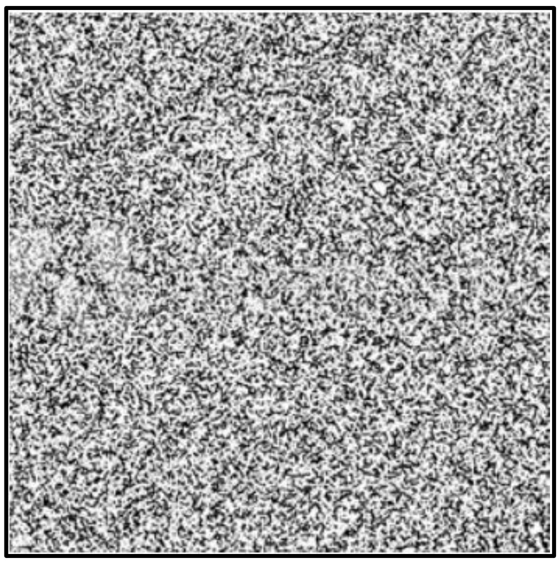
  - Function f is averaged with a weight function g
    - $(x' - x)$ centers g around x'



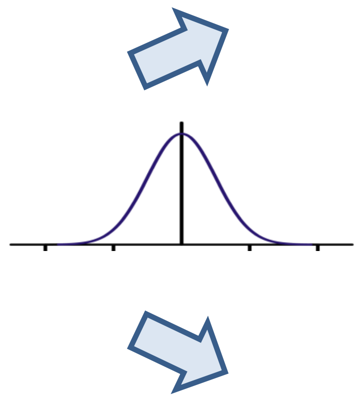


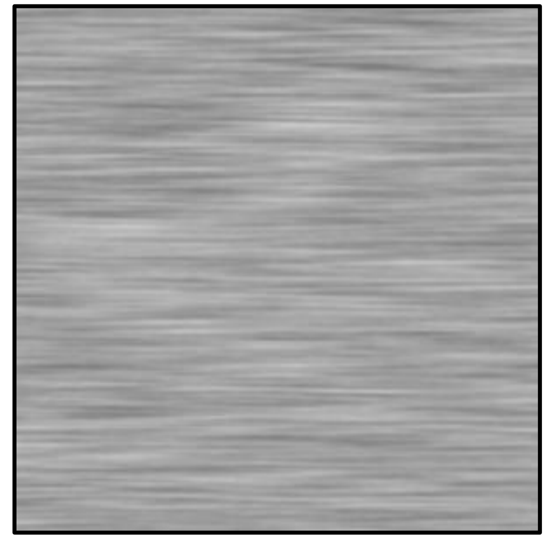
http://mathworld.wolfram.com/Convolution.html

Filtering by convolution

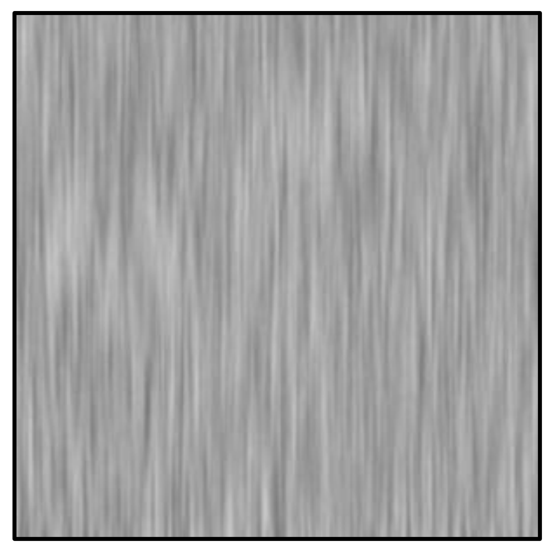


White noise (no correlation)





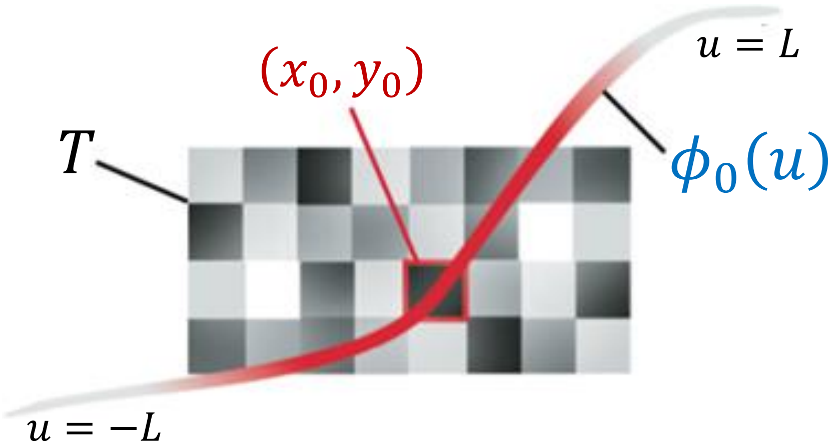
Horizontal Gaussian blur



Vertical Gaussian blur

Line Integral Convolution

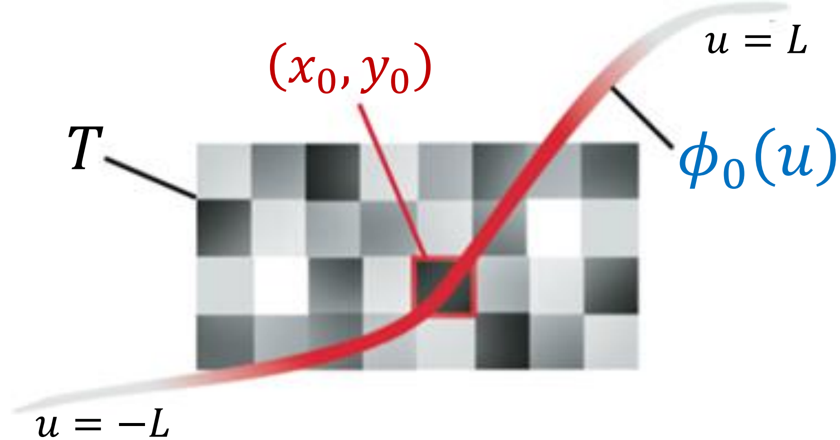
- Algorithm for 2D LIC
  - Let $\phi_0(u)$ be the stream line containing the point (x_0, y_0)
  - $T(x, y)$ is the randomly generated noise texture



[Cabral & Leedom 93]

Line Integral Convolution

- Algorithm for 2D LIC
  - Let $\phi_0(u)$ be the stream line containing the point (x_0, y_0)
  - $T(x, y)$ is the randomly generated noise texture



  - Compute the intensity at (x_0, y_0) as

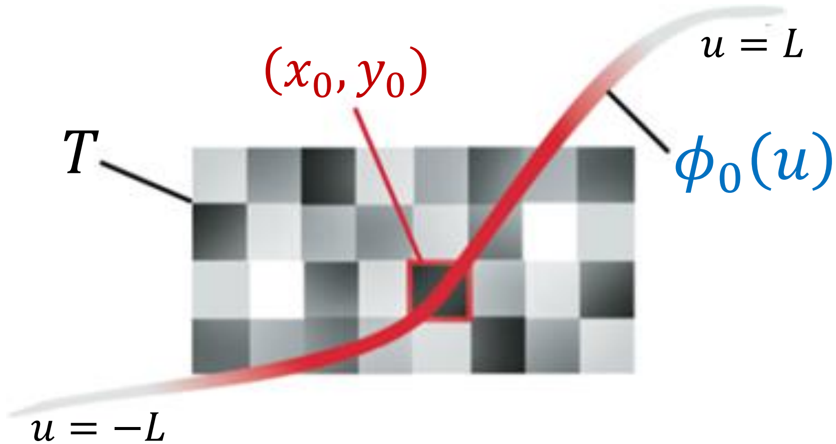
$$I(x_0, y_0) = \int_{-L}^{L} k(u) \cdot T\big(\phi_0(u)\big)\, du$$

convolution with a kernel $k(u)$

[Cabral & Leedom 93]

Line Integral Convolution

- Algorithm for 2D LIC
  - Let $\phi_0(u)$ be the stream line containing the point (x_0, y_0)
  - $T(x, y)$ is the randomly generated noise texture

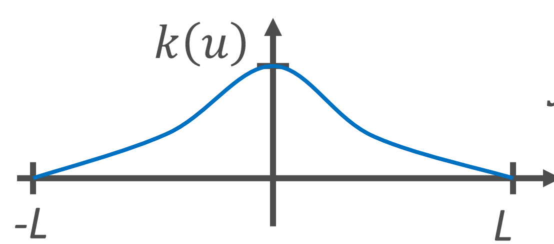


  - Compute the intensity at (x_0, y_0) as

$$I(x_0, y_0) = \int_{-L}^{L} k(u) \cdot T(\phi_0(u))\, du$$

convolution with a kernel $k(u)$

Smoothing filter kernel
- Finite support [-*L*, *L*]
- Normalized, usually symmetric
- E.g., Gaussian or box filter

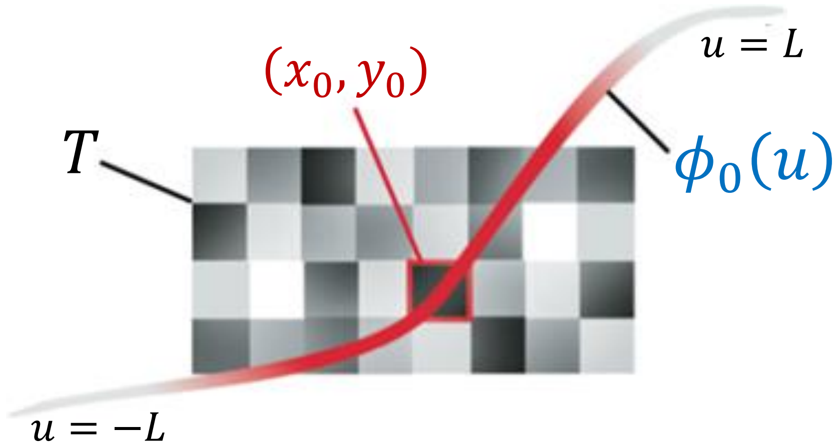


$$\int_{-L}^{L} k(u)\, du = 1$$

[Cabral & Leedom 93]

Line Integral Convolution

- Algorithm for 2D LIC
  - Let $\phi_0(u)$ be the stream line containing the point (x_0, y_0)
  - $T(x, y)$ is the randomly generated noise texture

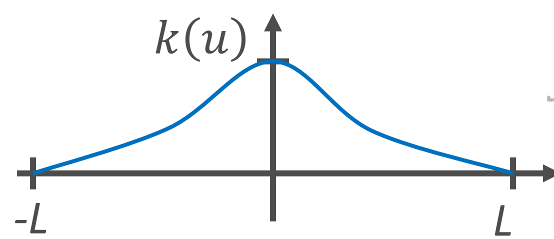


  - Compute the intensity at (x_0, y_0) as

$$I(x_0, y_0) = \int_{-L}^{L} k(u) \cdot T\big(\phi_0(u)\big)\, du$$

convolution with a kernel $k(u)$

- Smoothing filter kernel
  - Finite support [-L, L]
  - Normalized, usually symmetric
  - E.g., Gaussian or box filter



$$\int_{-L}^{L} k(u)\, du = 1$$

[Cabral & Leedom 93]

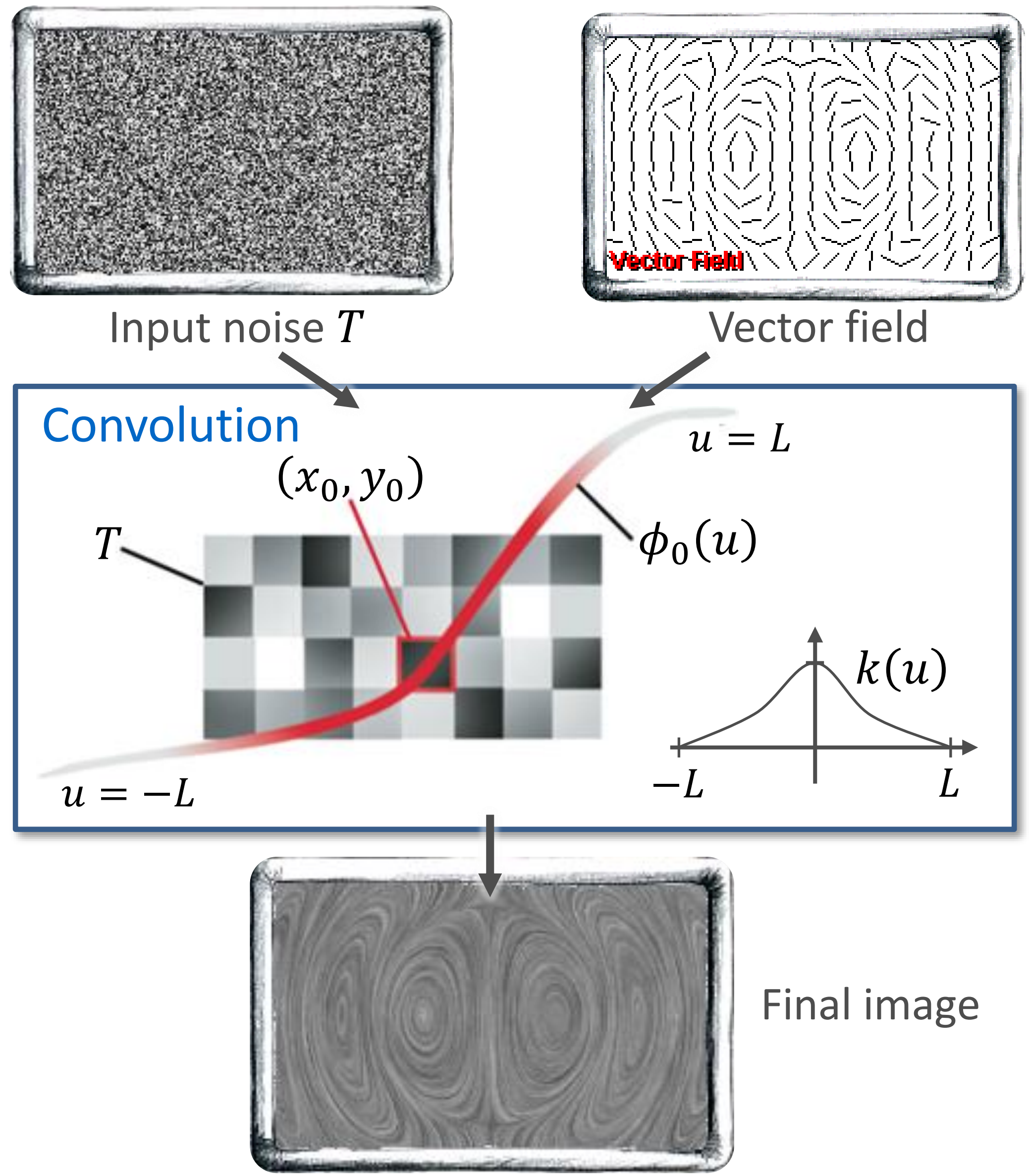
Line Integral Convolution

SIEMENS
Ingenuity for life

$$I(x_0, y_0) = \int_{-L}^{L} k(u) \cdot T\big(\phi_0(u)\big)\, du$$

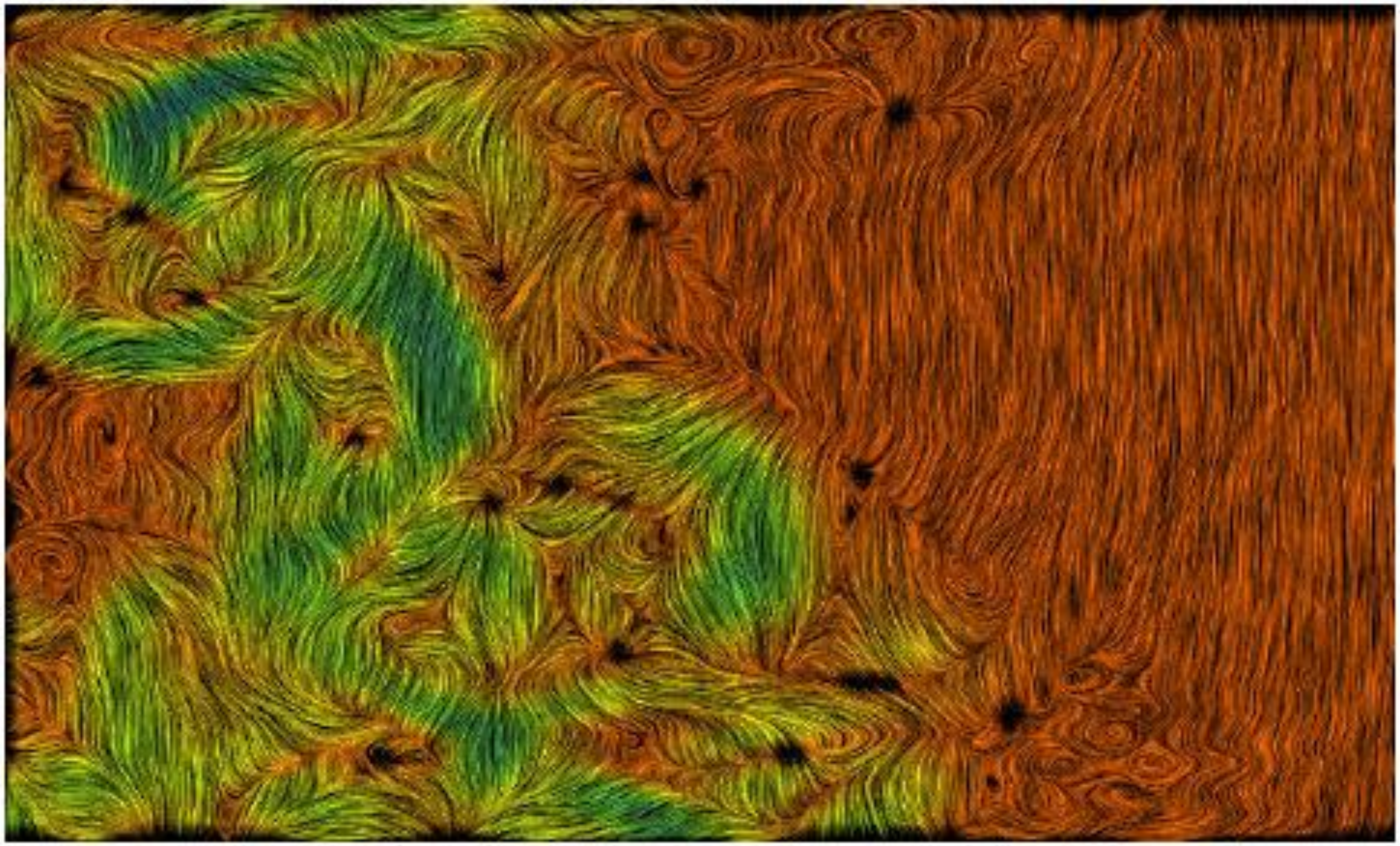
- LIC is a convolution of
  - a noise texture $T(x, y)$
  - and a smoothing filter $k(u)$

- Noise texture values are picked up along the stream line $\phi_0(u)$ through $T\big(\phi_0(u)\big)$



Line Integral Convolution

SIEMENS
Ingenuity for life

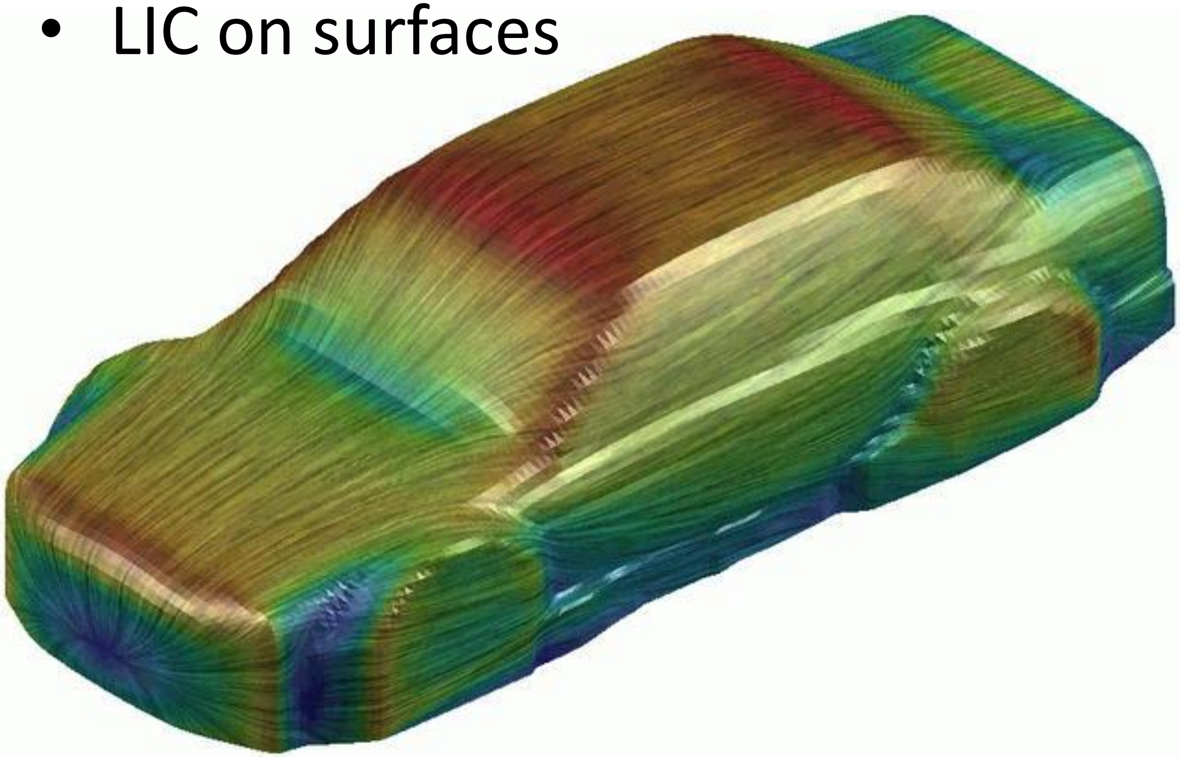


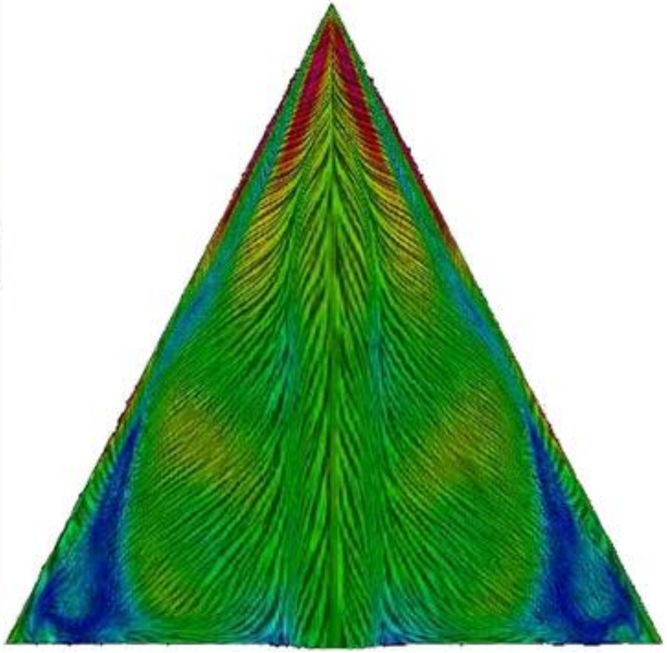
LIC and color coding of velocity magnitude

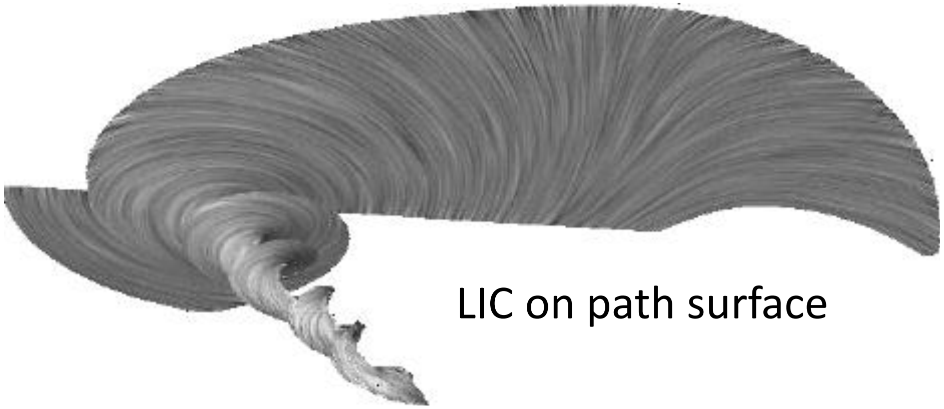
[Cabral & Leedom 93]

Line Integral Convolution

- LIC on surfaces





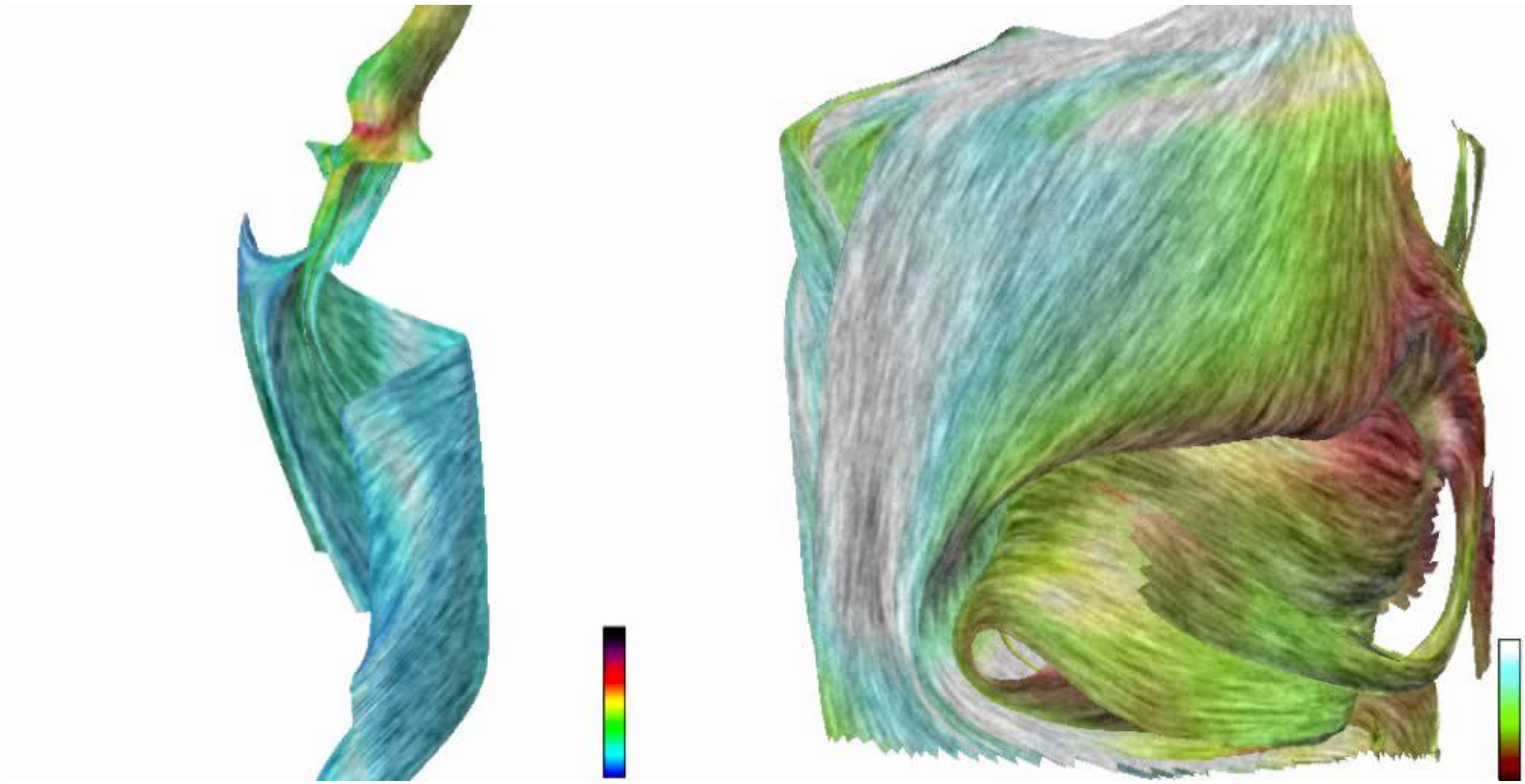


LIC on path surface

Data sources – artificial flows

SIEMENS
Ingenuity for life

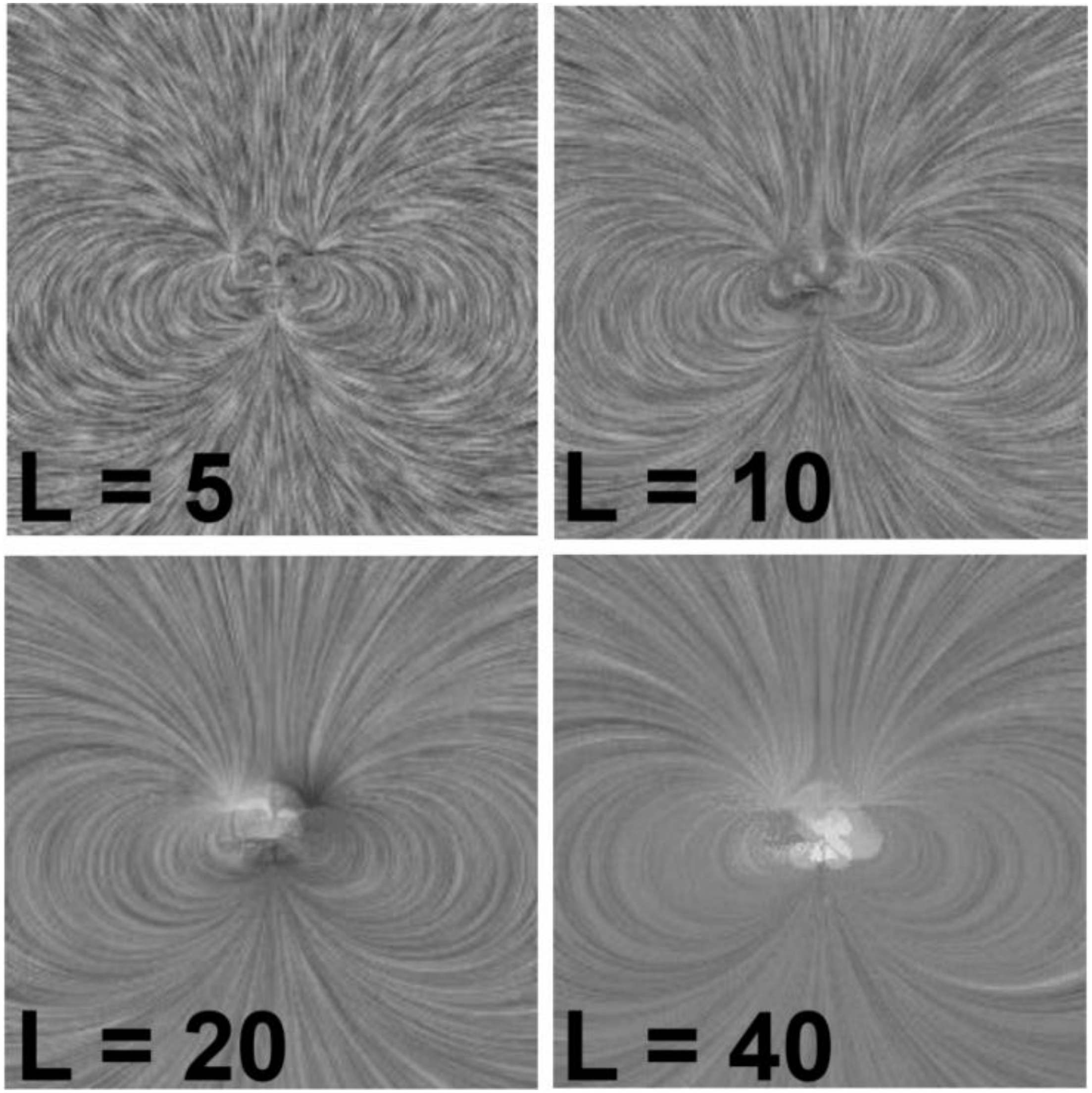
- Visualization of surface flows



[Laramee 2006]

Line Integral Convolution

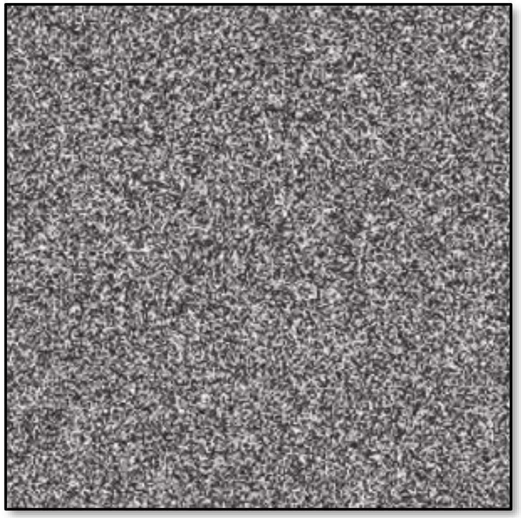
- Influence of filter length



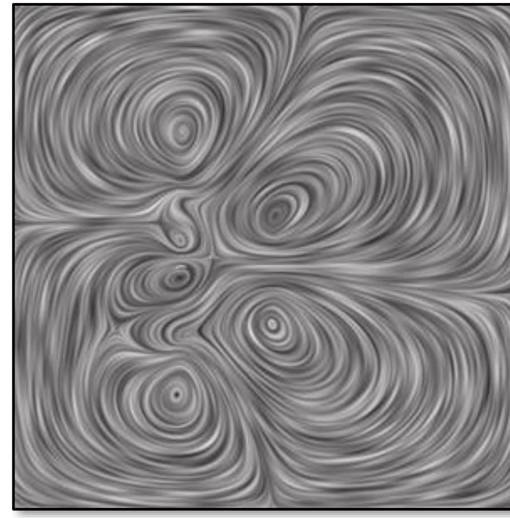
[Schumann & Müller 00]

Line Integral Convolution

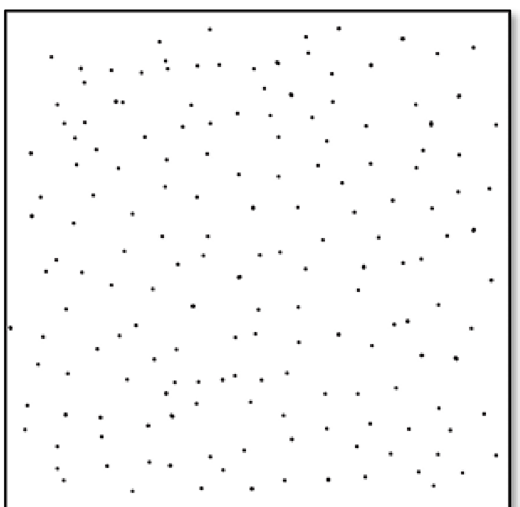
- Influence of input noise



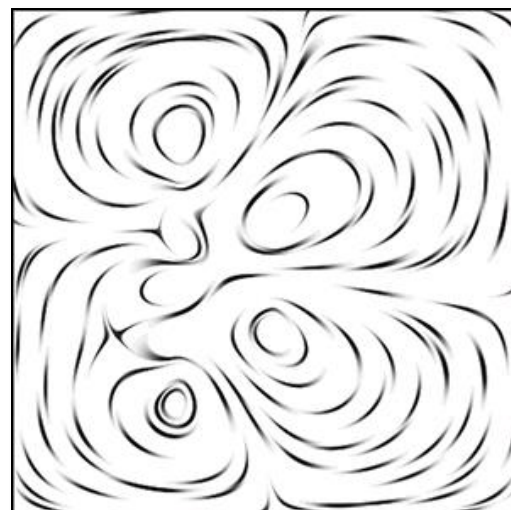
White noise texture



LIC result for white noise



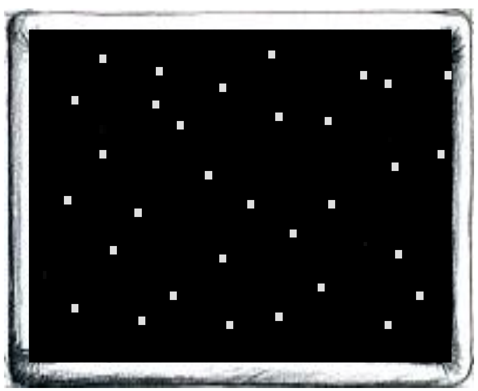
Sparse noise texture

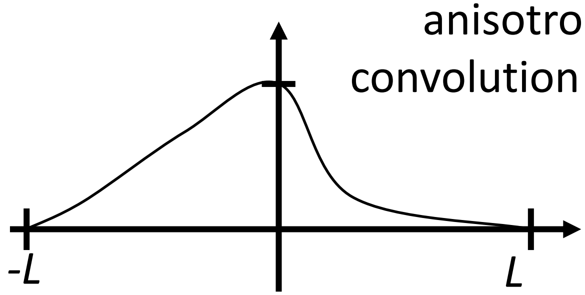


LIC result for sparse noise

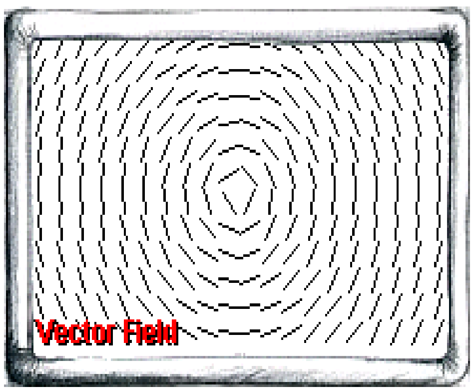
Line Integral Convolution

- Oriented LIC (OLIC)
  - Visualizes orientation (in addition to direction)
  - Uses a sparse texture; i.e. smearing of individual drops
  - Asymmetric convolution kernel





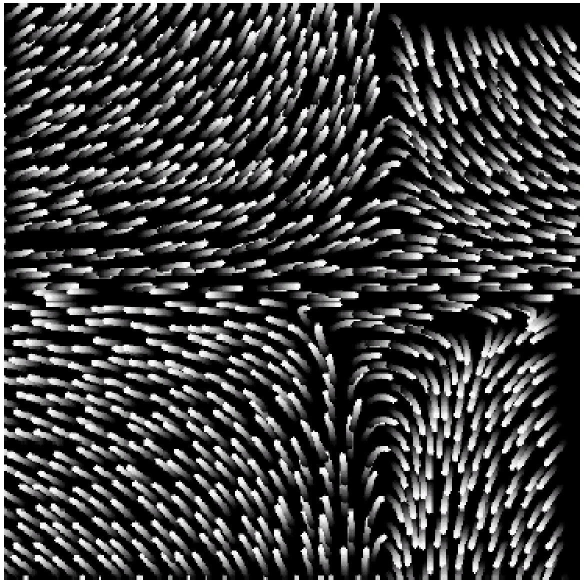
anisotropic
convolution kernel



[Wegenkittel & Gröller 97]

Line Integral Convolution

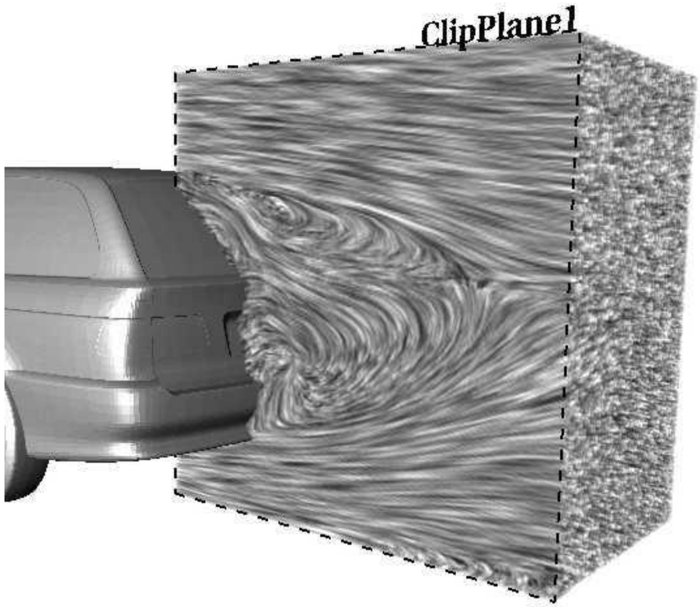
- Oriented LIC (OLIC)

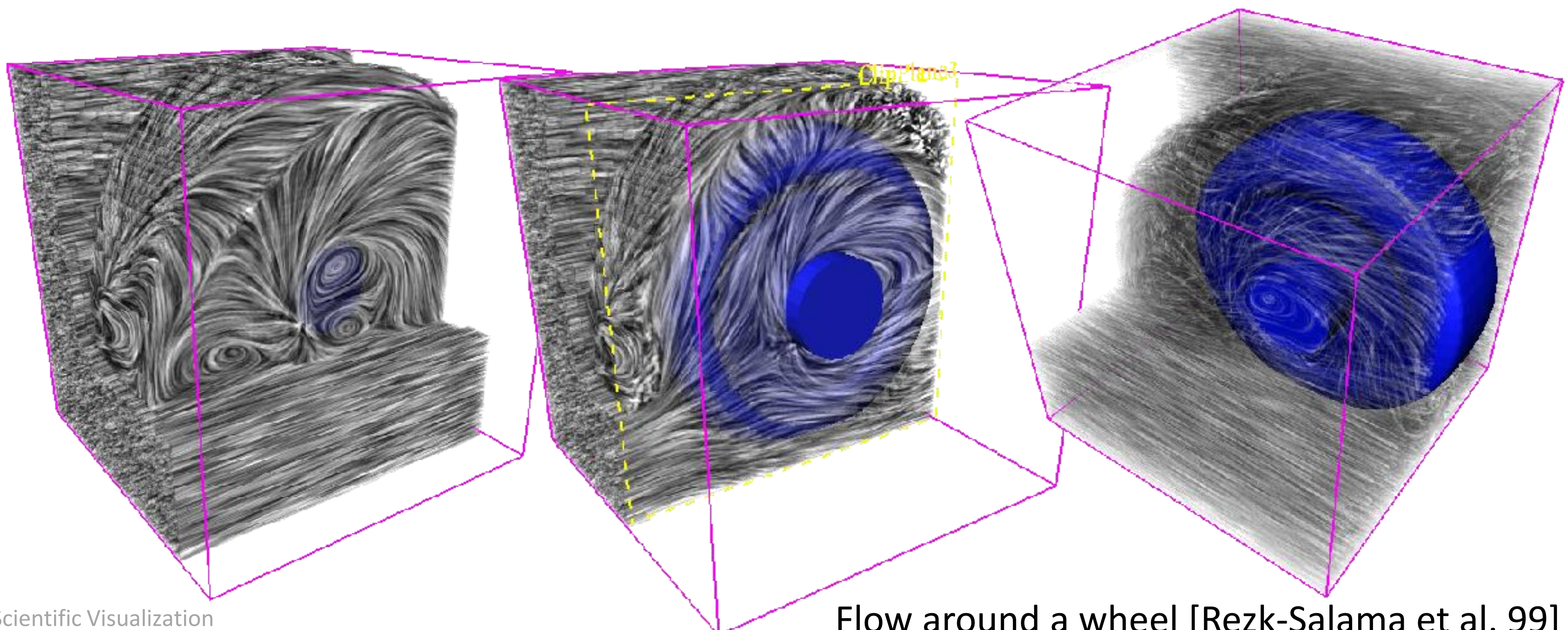


[Wegenkittel & Gröller 97]

Line Integral Convolution

- 3D LIC
  - Only good if non-relevant data is discarded

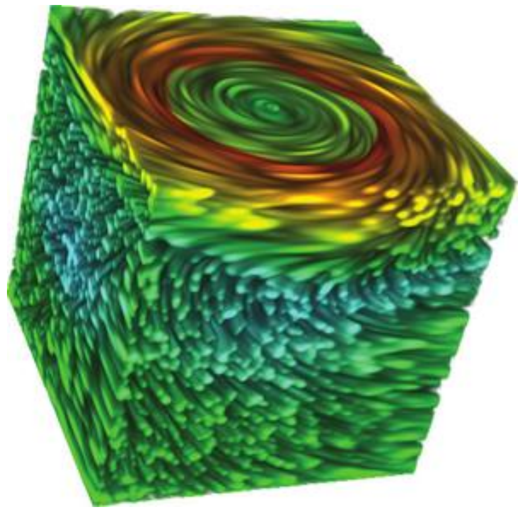




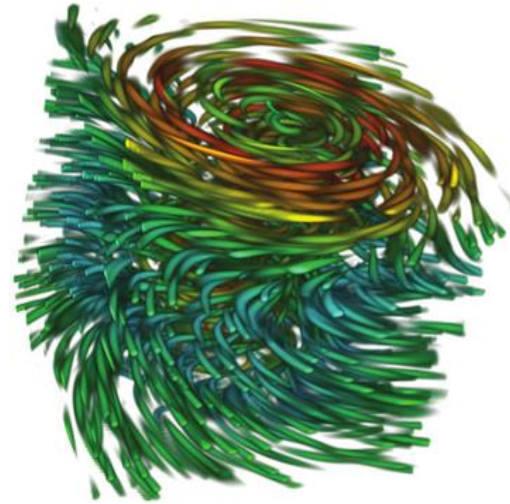
Flow around a wheel [Rezk-Salama et al. 99]

Line Integral Convolution

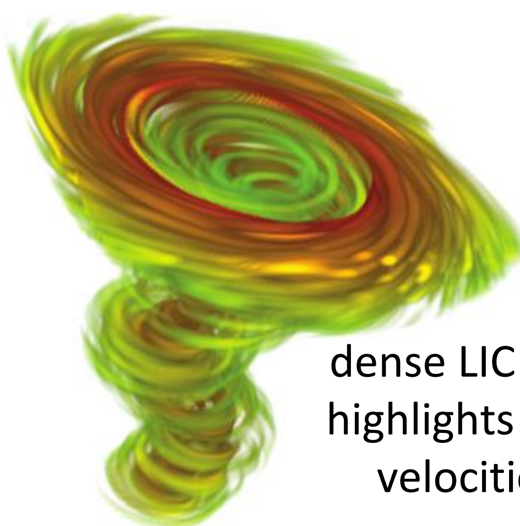
- 3D LIC



dense LIC with
white noise input



sparse LIC with
spares noise



dense LIC that
highlights high
velocities

sparse LIC that
highlights high
velocities

Line Integral Convolution

- Summary
  - Dense representation of flow fields
  - Convolution along characteristic lines
    → correlation along these lines
  - For 2D and (3D flows)

References

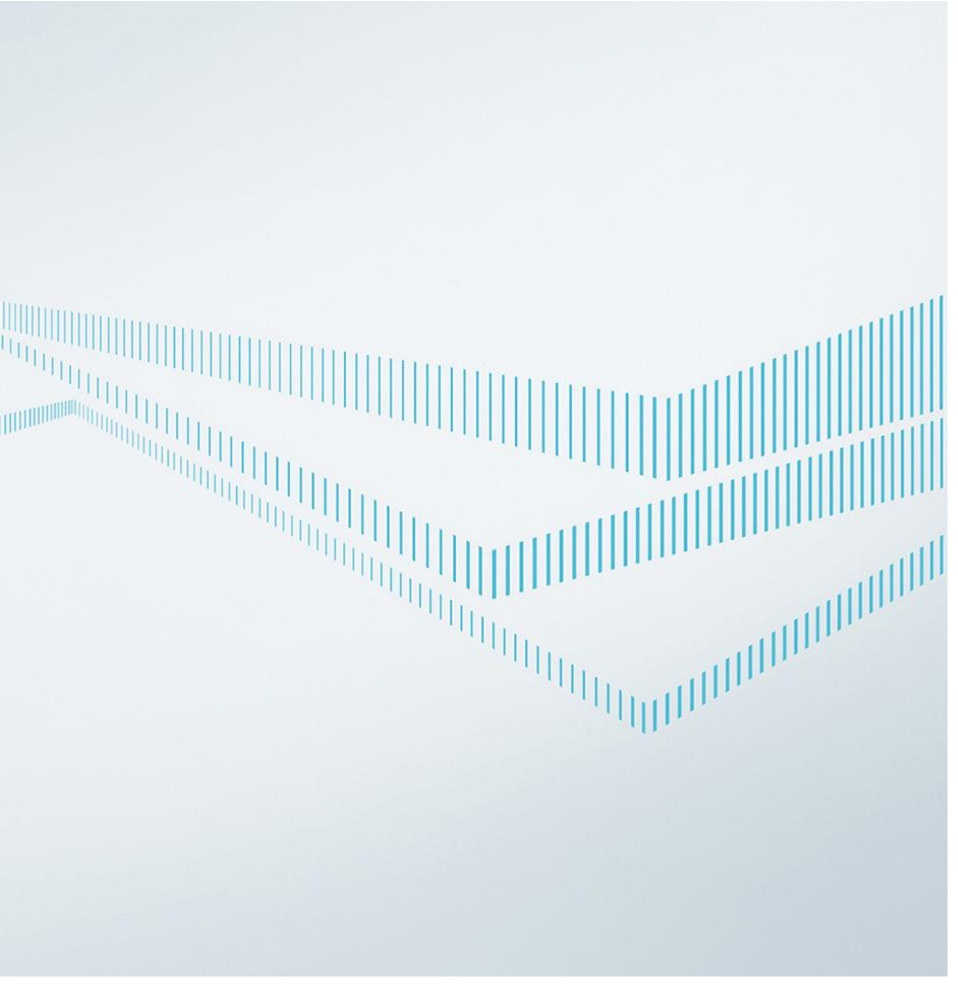
- Jobard & Lefer 97: Creating evenly-spaced streamlines of arbitrary density, In *Visualization in Scientific Computing*, 1997.
- Spencer et al. 09: Evenly-spaced streamlines for surfaces: An image-based approach, *Computer Graphics Forum*, 28(6), 2009.
- Cabral & Leedom 93: Imaging vector fields using line integral convolution. In *Proc. ACM SIGGRAPH*, 1993.
- Wegenkittel & M. Gröller 97: Fast oriented line integral convolution for vector field visualization via the Internet. In *Proc. IEEE Visualization*, 1997.

Acknowledgements

- Helwig Hauser
- Andrea Brambilla
- Daniel Weiskopf
- Ronald Peikert
- Christoph Garth
- Alexandru C. Telea
- Many more

Contact information

SIEMENS
Ingenuity for life



Dr. Johannes Kehrer

Siemens Technology
T RDA BAM IBI-DE
Otto-Hahn-Ring 6
81739 München, Deutschland

E-mail:
kehrer.johannes@siemens.com

Internet
siemens.com/innovation

Intranet
intranet.ct.siemens.com