Flow Visualization

Overview: Flow Visualization (1)

- Introduction, overview
  - Flow data
  - Simulation vs. measurement vs. modelling
  - 2D vs. surfaces vs. 3D
  - Steady vs time-dependent flow
  - Direct vs. indirect flow visualization
- Experimental flow visualization
  - Basic possibilities
  - PIV (Particle Image Velocimetry) + Example

Overview: Flow Visualization (2)

- Visualization of models
- Flow visualization with arrows
- Numerical integration
  - Euler-integration
  - Runge-Kutta-integration
- Streamlines
  - In 2D
  - Particle paths
  - In 3D, sweeps
  - Illuminated streamlines
- Streamline placement

Overview: Flow Visualization (3)

- Flow visualization with integral objects
  - Streamribbons,
  - Streamsurfaces, stream arrows
- Line integral convolution
  - Algorithm
  - Examples, alternatives
- Glyphs & icons, flow topology

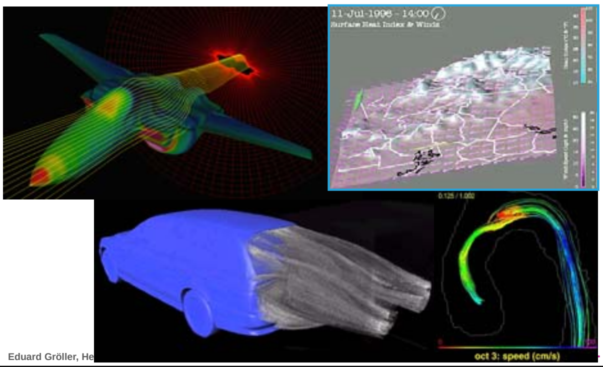
Flow Visualization

- Introduction:
  - FlowVis = visualization of flows
    - Visualization of change information
    - Typically: more than 3 data dimensions
    - General overview: even more difficult
  - Flow data:
    - $nD \times nD$ data, $1D^2/2D^2/nD^2$ (models), $2D^2/3D^2$ (simulations, measurements)
    - Vector data (nD) in nD data space
  - User goals:
    - Overview vs. details (with context)

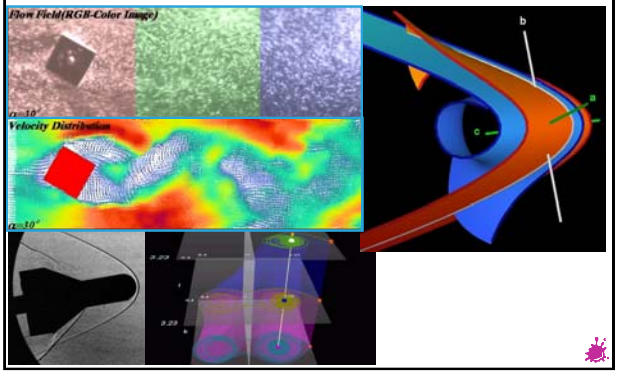
Flow Data

- Where do the data come from:
  - Flow simulation:
    - Airplane- / ship- / car-design
    - Weather simulation (air-, sea-flows)
    - Medicine (blood flows, etc.)
  - Flow measurements:
    - Wind tunnel, fluid tunnel
    - Schlieren-, shadow-technique
  - Flow models:
    - Differential equation systems (ODE) (dynamical systems)

Data Source – Examples 1/2

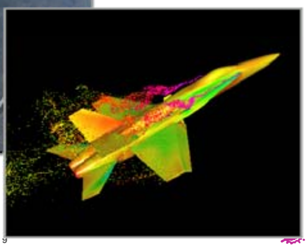


Data Source – Examples 2/2



Comparison with Reality

Experiment



Simulation

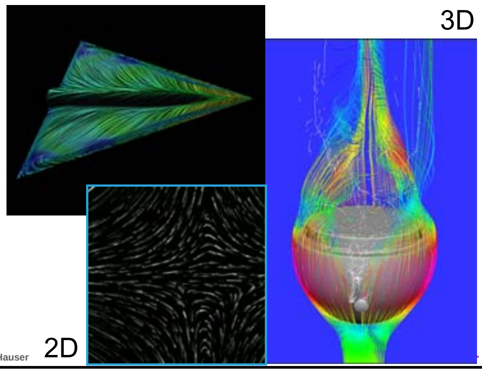
2D vs. Surfaces vs. 3D

- 2D-Flow visualization
  - 2D×2D-Flows
  - Models, slice flows (2D out of 3D)
- Visualization of surface flows
  - 3D-flows around "obstacles"
  - Boundary flows on surfaces (2D)
- 3D-Flow visualization
  - 3D×3D-flows
  - Simulations, 3D-models

2D/Surfaces/3D – Examples

3D

Surface

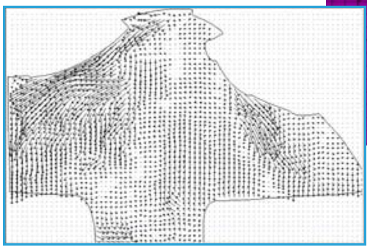


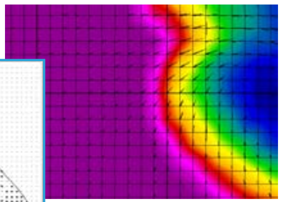
2D

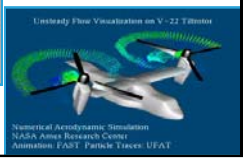
Steady vs. Time-Dependent Flows

- Steady (time-independent) flows:
  - Flow static over time
  - $\mathbf{v}(\mathbf{x})$: $R^n \rightarrow R^n$, e.g., laminar flows
  - Simpler interrelationship
- Time-dependent (unsteady) flows:
  - Flow itself changes over time
  - $\mathbf{v}(\mathbf{x},t)$: $R^n \times R^1 \rightarrow R^n$, e.g., turbulent flows
  - More complex interrelationship

Time-Dependent vs. Steady Flow



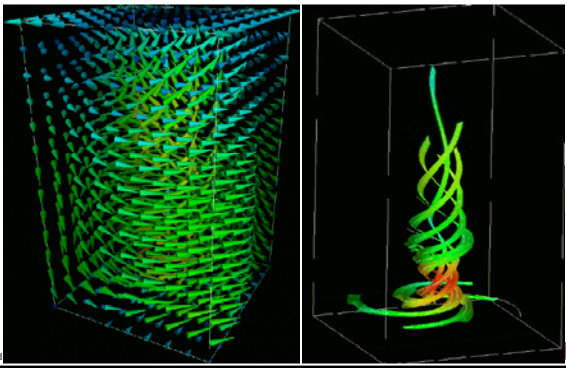




Direct vs. Indirect Flow Visualization

- Direct flow visualization:
  - Overview on current flow state
  - Visualization of vectors
  - Arrow plots, smearing techniques
- Indirect flow visualization:
  - Usage of intermediate representation: vector-field integration over time
  - Visualization of temporal evolution
  - Streamlines, streamsurfaces

Direct vs. Indirect Flow Vis. – Example

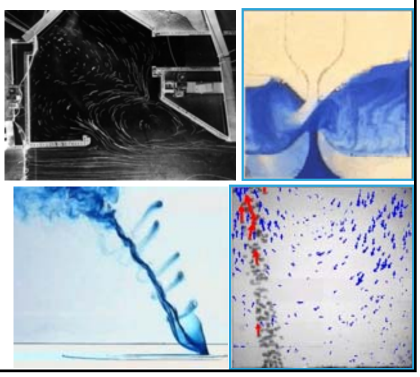


Experimental Flow Visualization

Optical Methods, etc.

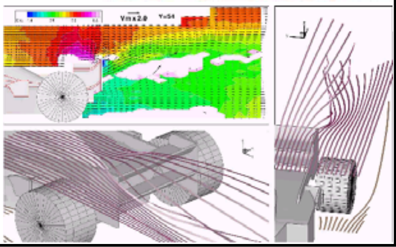
With Smoke rsp. Color Injection

- Injection of color, smoke, particles
- Optical methods:
  - Schlieren, shadows



Example: Car-Design

- Ferrari-model, so-called five-hole probe (no back flows)

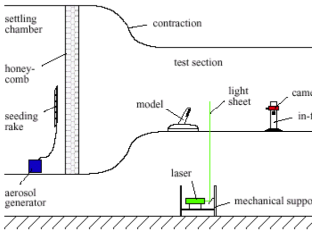


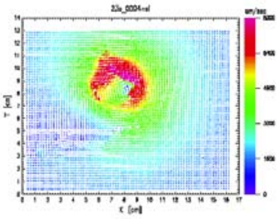
PIV: Particle Image Velocimetry

- Laser + correlation analysis:
  - Real flow, e.g., in wind tunnel
  - Injection of particles (as uniform as possible)
  - At interesting locations: 2-times fast illumination with laser-slice
  - Image capture (high-speed camera), then correlation analysis of particles
  - Vector calculation / reconstruction, typically only 2D-vectors

PIV - Measurements

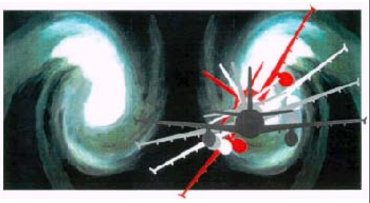
- Setup and typical result:

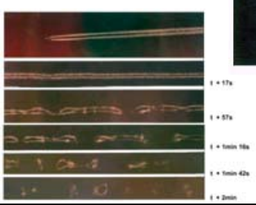




Example: Wing-Tip Vortex

- Problem: Air behind airplanes is turbulent



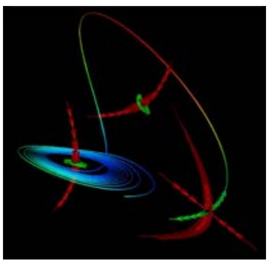


Visualization of Models

Dynamical Systems

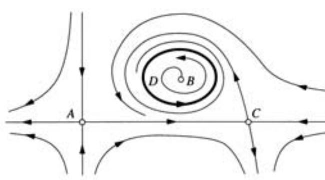
Dynamical Systems Visualization

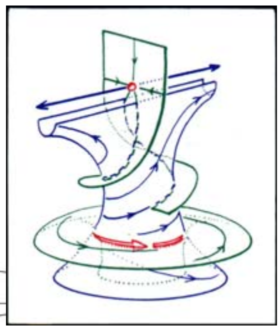
- Differences:
  - Flow analytically def.: $d\mathbf{x}/dt = \mathbf{v}(\mathbf{x})$
  - Navier-Stokes equations
  - E.G.: Lorenz-system: $dx/dt = \sigma(y-x)$ $dy/dt = rx-y-xz$ $dz/dt = xy-bz$
  - Larger variety in data:
    - 2D, 3D, nD
    - Sometimes no natural constraints like non-compressibility or similar



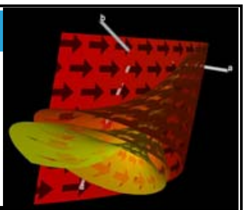
Visualization of Models

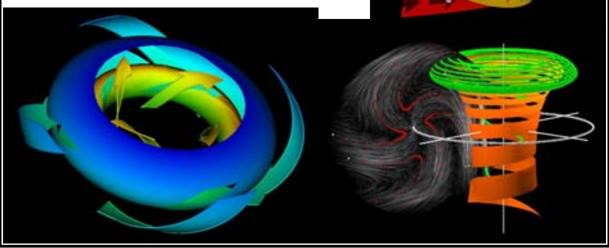
- Sketchy, "hand drawn"





Visualization of 3D Models



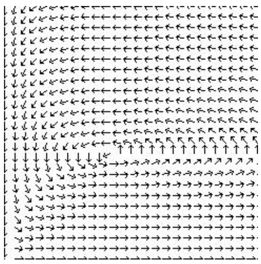


Flow Visualization with Arrows

Hedgehog plots, etc.

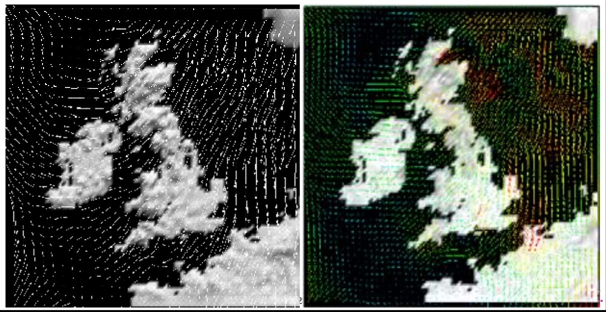
Flow Visualization with Arrows

- Aspects:
  - Direct Flow Visualization
  - Normalized arrows vs. scaling with velocity
  - 2D: quite usable, 3D: often problematic
  - Sometimes limited expressivity (temporal component missing)
  - Often used!



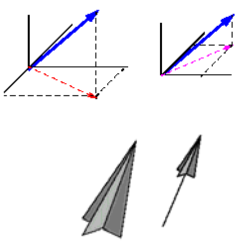
Arrows in 2D

- Scaled arrows vs. color-coded arrows



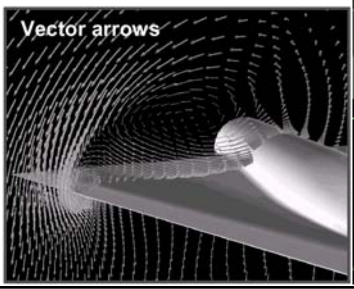
Arrows in 3D

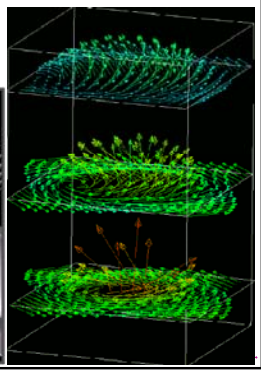
- Following problems:
  - Ambiguity
  - Perspective Shortening
  - 1D-objects in 3D: difficult spatial perception
  - Visual clutter
- Improvement:
  - 3D-arrows (help to a certain extent)



Arrows in 3D

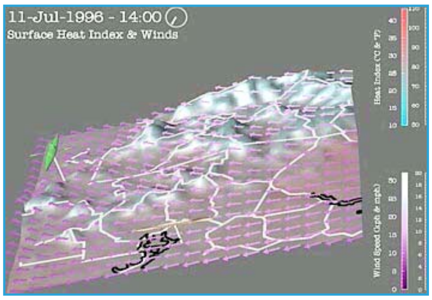
- Compromise: Arrows only in slices





Arrows in 3D

- Well integrable within "real" 3D:



Integration of Streamlines

Numerical Integration

Streamlines – Theory

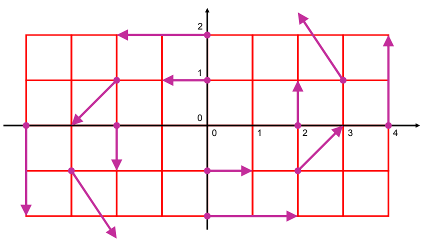
- Correlations:
  - flow data **v**: derivative information
  - d**x**/dt = **v**(**x**);
    spatial points $\mathbf{x} \in R^n$, time $t \in R$, flow vectors $\mathbf{v} \in R^n$
  - streamline **s**: integration over time,
    also called trajectory, solution, curve
  - $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \le u \le t} \mathbf{v}(\mathbf{s}(u))\, du$;
    seed point $\mathbf{s}_0$, integration variable u
  - difficulty: result **s** also in the integral ⇒ analytical solution usually impossible!

Streamlines – Practice

- Basic approach:
  - theory: $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \le u \le t} \mathbf{v}(\mathbf{s}(u))\, du$
  - practice: numerical integration
  - idea:
    (very) locally, the solution is (approx.) linear
  - Euler integration:
    follow the current flow vector $\mathbf{v}(\mathbf{s}_i)$ from the current streamline point $\mathbf{s}_i$ for a very small time (dt) and therefore distance
  - Euler integration: $\mathbf{s}_{i+1} = \mathbf{s}_i + dt \cdot \mathbf{v}(\mathbf{s}_i)$,
    integration of small steps (dt very small)

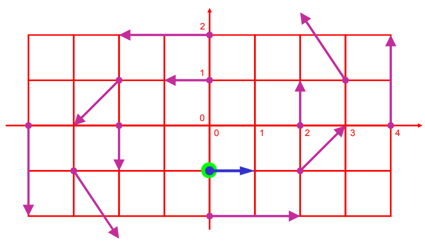
Euler Integration – Example

- 2D model data: $v_x = dx/dt = -y$
  $v_y = dy/dt = x/2$
- Sample arrows:
- True solution: ellipses!



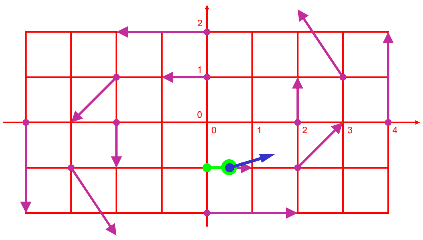
Euler Integration – Example

- Seed point $\mathbf{s}_0 = (0|-1)^T$;
  current flow vector $\mathbf{v}(\mathbf{s}_0) = (1|0)^T$;
  dt = 1/2



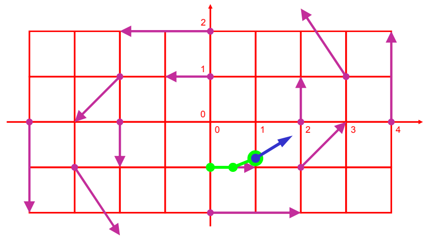
Euler Integration – Example

- New point $\mathbf{s}_1 = \mathbf{s}_0 + \mathbf{v}(\mathbf{s}_0) \cdot dt = (1/2\,|\,-1)^T$;
  current flow vector $\mathbf{v}(\mathbf{s}_1) = (1\,|\,1/4)^T$;



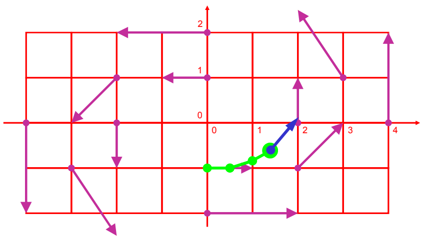
Euler Integration – Example

- New point $\mathbf{s}_2 = \mathbf{s}_1 + \mathbf{v}(\mathbf{s}_1) \cdot dt = (1\,|\,-7/8)^T$;
  current flow vector $\mathbf{v}(\mathbf{s}_2) = (7/8\,|\,1/2)^T$;



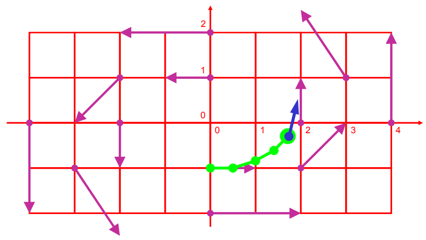
Euler Integration – Example

- $\mathbf{s}_3 = (23/16\,|\,-5/8)^T \approx (1.44\,|\,-0.63)^T$;
  $\mathbf{v}(\mathbf{s}_3) = (5/8\,|\,23/32)^T \approx (0.63\,|\,0.72)^T$;



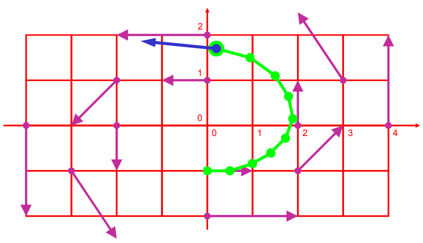
Euler Integration – Example

- $\mathbf{s}_4 = (7/4\,|\,-17/64)^T \approx (1.75\,|\,-0.27)^T$;
  $\mathbf{v}(\mathbf{s}_4) = (17/64\,|\,7/8)^T \approx (0.27\,|\,0.88)^T$;



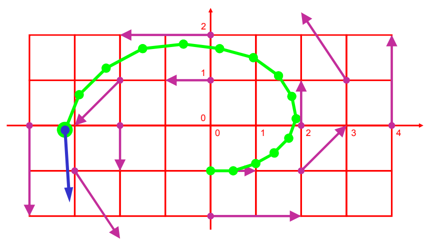
Euler Integration – Example

- $\mathbf{s}_9 \approx (0.20\,|\,1.69)^T$;
  $\mathbf{v}(\mathbf{s}_9) \approx (-1.69\,|\,0.10)^T$;



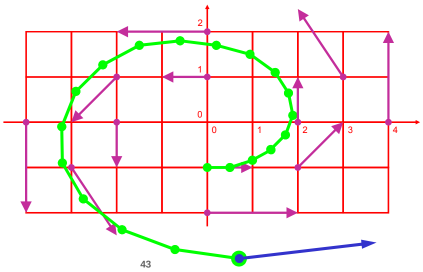
Euler Integration – Example

- $\mathbf{s}_{14} \approx (-3.22\,|\,-0.10)^T$;
  $\mathbf{v}(\mathbf{s}_{14}) \approx (0.10\,|\,-1.61)^T$;



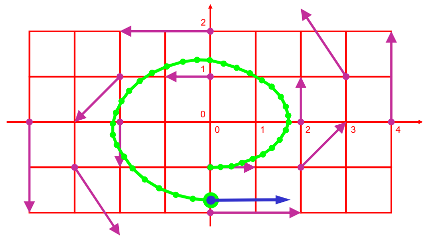
Euler Integration – Example

- $\mathbf{s}_{19} \approx (0.75\,|\,\text{-}3.02)^T$; $\mathbf{v}(\mathbf{s}_{19}) \approx (3.02\,|\,0.37)^T$; clearly: large integration error, dt too large! 19 steps



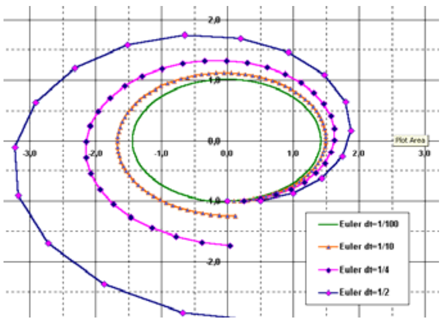
Euler Integration – Example

- dt smaller (1/4): more steps, more exact! $\mathbf{s}_{36} \approx (0.04\,|\,\text{-}1.74)^T$; $\mathbf{v}(\mathbf{s}_{36}) \approx (1.74\,|\,0.02)^T$;
- 36 steps



Comparison Euler, Step Sizes

Euler is getting better propor-tionally to dt



Euler Example – Error Table

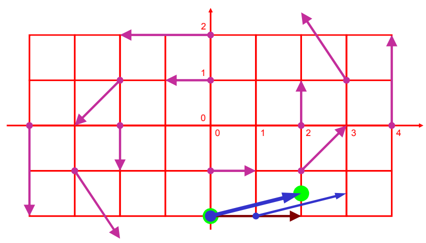
dt	#steps	error	
1/2	19	~200%	
1/4	36	~75%	
1/10	89	~25%	
1/100	889	~2%	
1/1000	8889	~0.2%	✓

Better than Euler Integr.: RK

- Runge-Kutta Approach:
  - theory: $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0\le u\le t} \mathbf{v}(\mathbf{s}(u))\,du$
  - Euler: $\mathbf{s}_i = \mathbf{s}_0 + \Sigma_{0\le u<i}\,\mathbf{v}(\mathbf{s}_u)\cdot dt$
  - Runge-Kutta integration:
    - idea: cut short the curve arc
    - RK-2 (second order RK):
      1.: do half a Euler step
      2.: evaluate flow vector there
      3.: use it in the origin
    - RK-2 (two evaluations of **v** per step): $\mathbf{s}_{i+1} = \mathbf{s}_i + \mathbf{v}(\mathbf{s}_i + \mathbf{v}(\mathbf{s}_i)\cdot dt/2)\cdot dt$

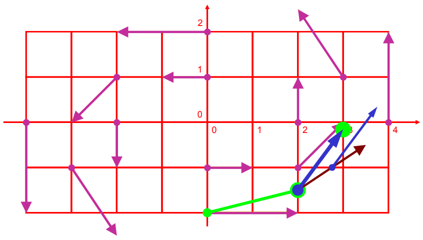
RK-2 Integration – One Step

- Seed point $\mathbf{s}_0 = (0\,|\,\text{-}2)^T$; current flow vector $\mathbf{v}(\mathbf{s}_0) = (2\,|\,0)^T$; preview vector $\mathbf{v}(\mathbf{s}_0 + \mathbf{v}(\mathbf{s}_0)\cdot dt/2) = (2|0.5)^T$; $dt = 1$



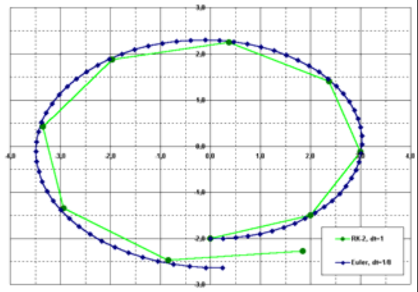
RK-2 – One more step

- Seed point $\mathbf{s}_1 = (2|-1.5)^T$;
  current flow vector $\mathbf{v}(\mathbf{s}_1) = (1.5|1)^T$;
  preview vector $\mathbf{v}(\mathbf{s}_1+\mathbf{v}(\mathbf{s}_1)\cdot dt/2) \approx (1|1.4)^T$;
  $dt = 1$



RK-2 – A Quick Round

- RK-2: even with dt=1 (9 steps) better than Euler with dt=1/8 (72 steps)



Integration, Conclusions

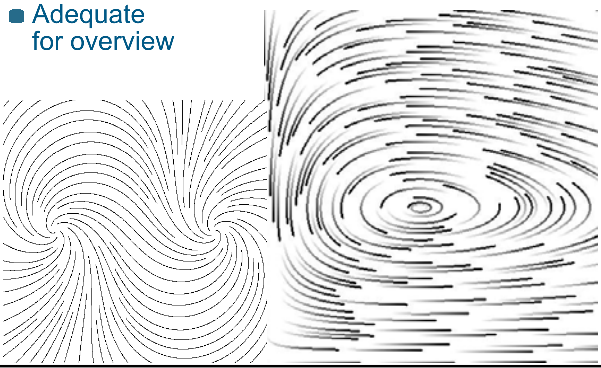
- Summary:
  - analytic determination of streamlines usually not possible
  - hence: numerical integration
  - several methods available (Euler, Runge-Kutta, etc.)
  - Euler: simple, imprecise, esp. with small dt
  - RK: more accurate in higher orders
  - furthermore: adaptive methods, implicit methods, etc.

Flow Visualization with Streamlines

Streamlines,
Particle Paths, etc.

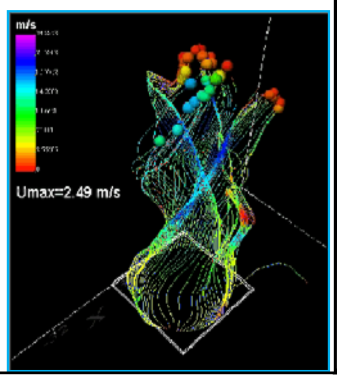
Streamlines in 2D

- Adequate for overview



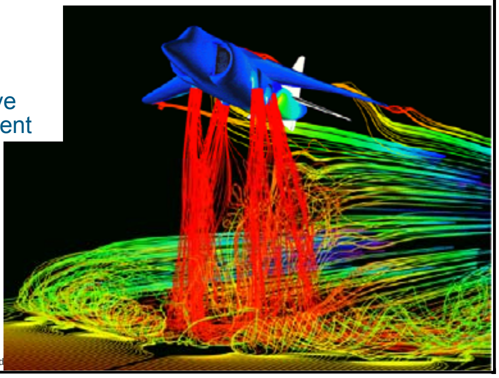
Visualization with Particles

- Particle paths = streamlines (steady flows)
- Variants (time-dependent data):
  - streak lines: steadily new particles
  - path lines: long-term path of one particle



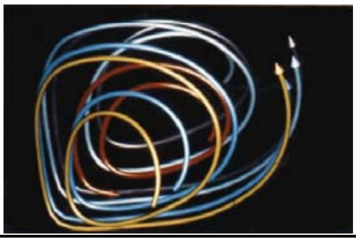
Streamlines in 3D

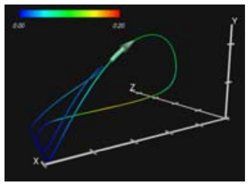
- Color coding: Speed
- Selective Placement

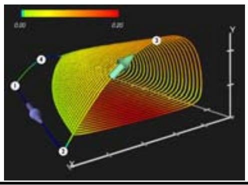


3D Streamlines with Sweeps

- Sweeps: better spatial 3D perception

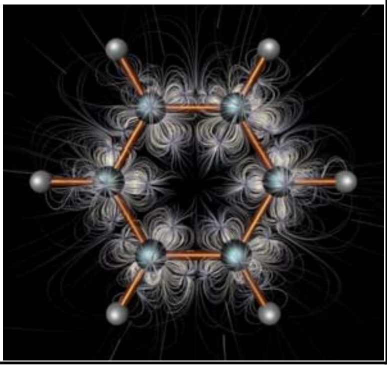






Illuminated Streamlines

- Illuminated 3D curves ⇒ better 3D perception!

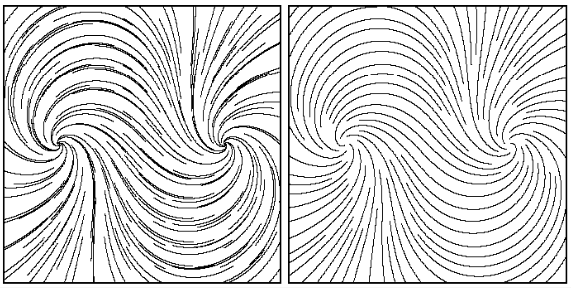


Streamline Placement

in 2D

Problem: Choice of Seed Points

- Streamline placement:
  - If regular grid used: very irregular result



Overview of Algorithm

- Idea: streamlines should not get too close to each other
- Approach:
  - choose a seed point with distance d_{sep} from an already existing streamline
  - forward- and backward-integration until distance d_{test} is reached (or …).
  - two parameters:
    - d_{sep} … start distance
    - d_{test} … minimum distance

Algorithm – Pseudocode

- Compute initial streamline, put it into a queue
- Initial streamline becomes current streamline
- WHILE not finished DO:

TRY: get new seed point which is d_{sep} away from current streamline

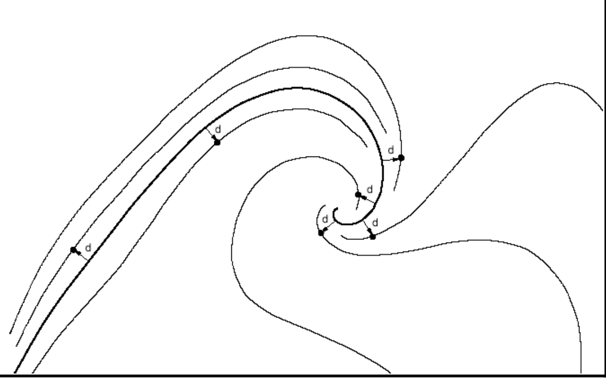
IF successful THEN compute new streamline and put to queue

ELSE IF no more streamline in queue
THEN exit loop
ELSE next streamline in queue becomes current streamline

Streamline Termination

- When to stop streamline integration:
  - when dist. to neighboring streamline ≤ d_{test}
  - when streamline leaves flow domain
  - when streamline runs into fixed point (**v**=0)
  - when streamline gets too near to itself
  - after a certain amount of maximal steps

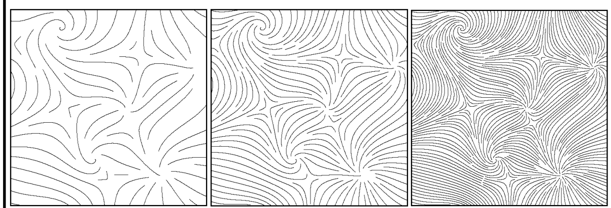
New Streamlines



Different Streamline Densities

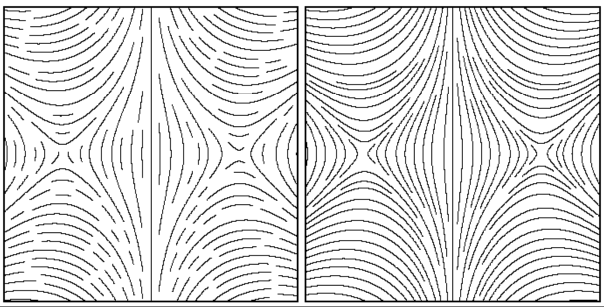
- Variations of d_{sep} in rel. to image width:

6% 3% 1.5%



d_{sep} vs. d_{test}

$d_{test} = 0.9 \cdot d_{sep}$ $d_{test} = 0.5 \cdot d_{sep}$

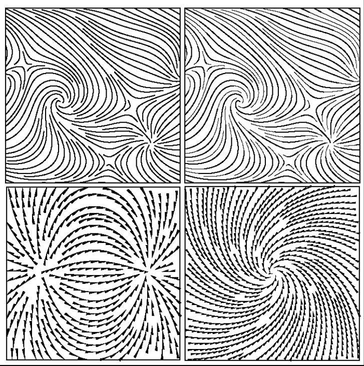


Tapering and Glyphs

- Thickness in rel. to dist.

$$\begin{cases} 1.0 & \forall d \geq d_{sep} \\ \frac{d - d_{test}}{d_{sep} - d_{test}} & \forall d < d_{sep} \end{cases}$$

- Directional glyphs:

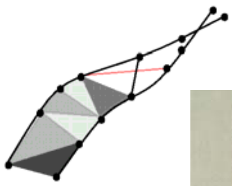


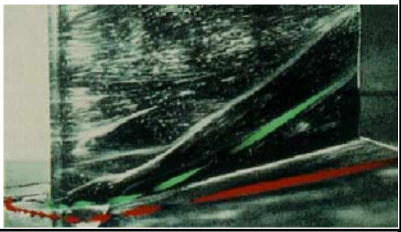
Flow Visualization with Integral Objects

Streamribbons,
Streamsurfaces,
etc.

Integral Objects in 3D 1/3

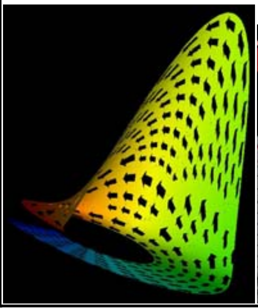
- Streamribbons

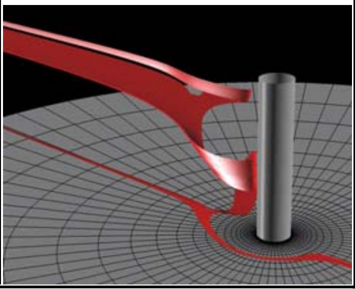




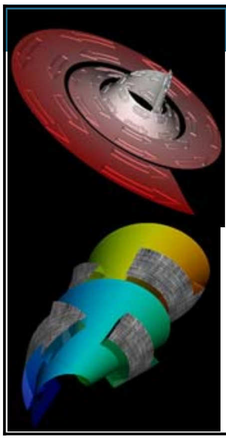
Integral Objects in 3D 2/3

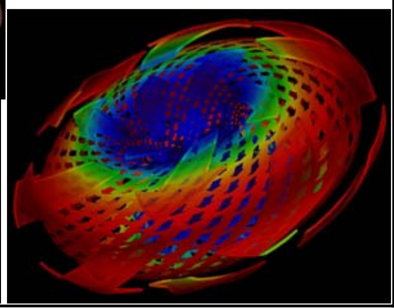
- Streamsurfaces





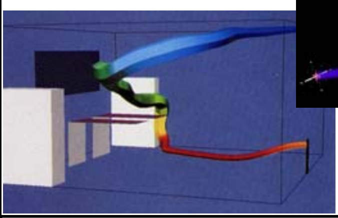
Stream Arrows

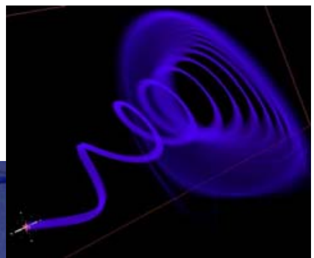




Integral Objects in 3D 3/3

- Flow volumes …
- vs. streamtubes
  (similar to streamribbon)





Relation to Seed Objects

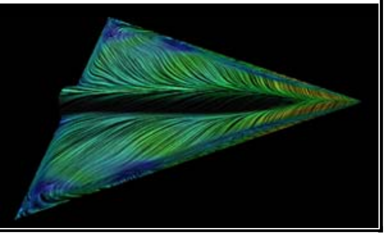
IntegralObj.	**Dim.**	**SeedObj.**	**Dim.**
Streamline,…	1D	Point	0D
Streamribbon	1D++	Point+pt.	0D+0D
Streamtube	1D++	Pt.+cont.	0D+1D
Streamsurface	2D	Curve	1D
Flow volume	3D	Patch	2D

Line Integral Convolution

Flow Visualization
in 2D or on surfaces

LIC – Introduction

- Aspects:
  - goal: general overview of flow
  - Approach: usage of textures
  - Idea: flow ⇔ visual correlation
  - Example:

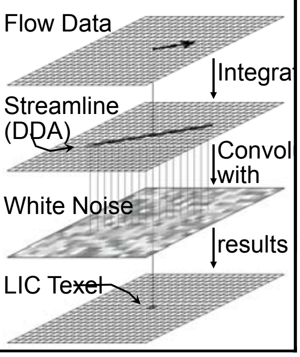


LIC – Approach

- LIC idea:
  - for every texel: let the texture value…
    - … correlate with neighboring texture values along the flow (in flow direction)
    - … *not* correlate with neighboring texture values across the flow (normal to flow dir.)
  - result:
    along streamlines the texture values are correlated ⇒ visually coherent!
  - approach: “smudge” white noise (no a priori correlations) along flow

LIC – Steps

- Calculation of a texture value:
  - look at streamline through point
  - filter white noise along streamline



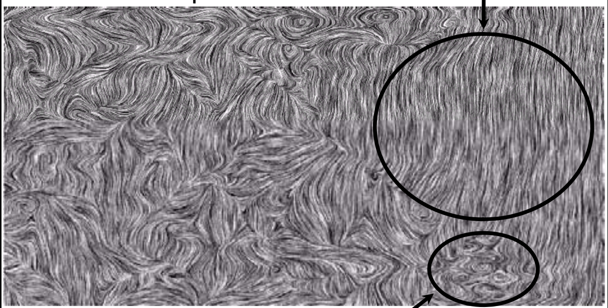
LIC – Convolution with Noise

- Calculation of LIC texture:
  - input 1: flow data $\mathbf{v}(\mathbf{x})$: $R^n \to R^n$,
    analytically or interpolated
  - input 2: white noise $n(\mathbf{x})$: $R^n \to R^1$,
    normally precomputed as texture
  - streamline $\mathbf{s}_\mathbf{x}(u)$ through $\mathbf{x}$: $R^1 \to R^n$,
    $\mathbf{s}_\mathbf{x}(u) = \mathbf{x} + \mathrm{sgn}(u) \cdot \int_{0 \le t \le |u|} \mathbf{v}(\mathbf{s}_\mathbf{x}(\mathrm{sgn}(u) \cdot t))\, dt$
  - input 3: filter $h(t)$: $R^1 \to R^1$, e.g., Gauss
  - result: texture value $\mathrm{lic}(\mathbf{x})$: $R^n \to R^1$,
    $\mathrm{lic}(\mathbf{x}) = \mathrm{lic}(\mathbf{s}_\mathbf{x}(0)) = \int n(\mathbf{s}_\mathbf{x}(u)) \cdot h(-u)\, du$

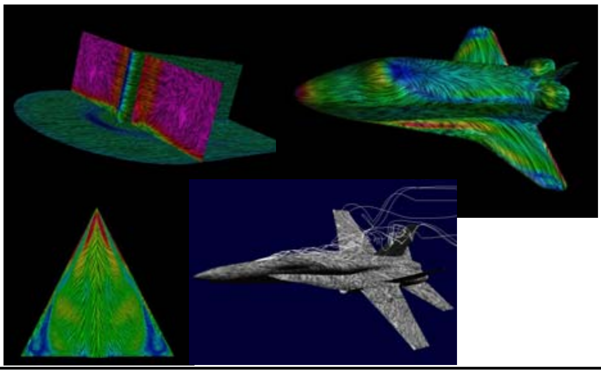
More Explanation

- So:
  - LIC – $\mathrm{lic}(\mathbf{x})$ – is a convolution of
    - white noise n (or …)
    - and a smoothing filter h (e.g. a Gaussian)
  - The noise texture values are picked up along streamlines $\mathbf{s}_\mathbf{x}$ through $\mathbf{x}$

LIC – Example in 2D

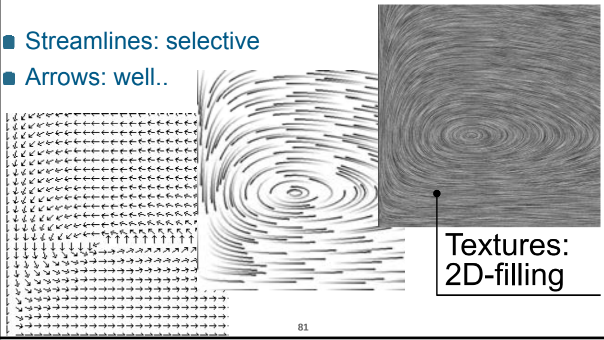


LIC – Examples on Surfaces



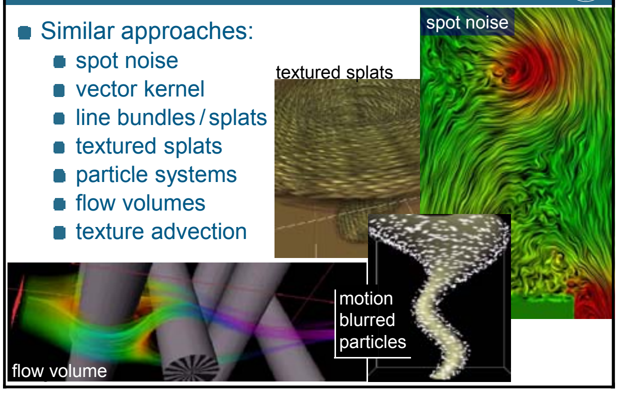
Arrows vs. StrLines vs. Textures

- Streamlines: selective
- Arrows: well..



Alternatives to LIC

- Similar approaches:
  - spot noise
  - vector kernel
  - line bundles / splats
  - textured splats
  - particle systems
  - flow volumes
  - texture advection

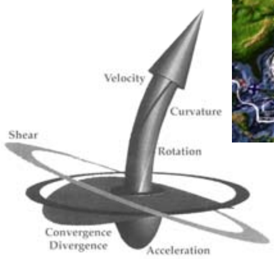


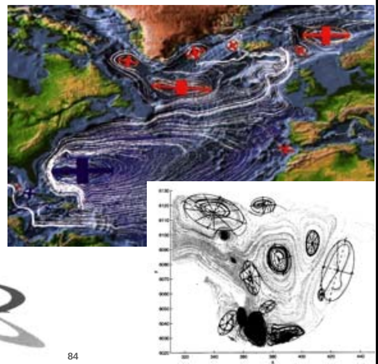
Flow Visualization dependent on local props.

Visualization of ∇**v**

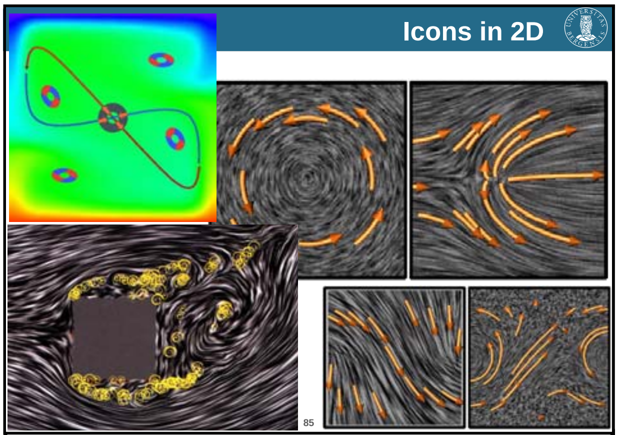
Glyphs resp. Icons

- Local / topological properties

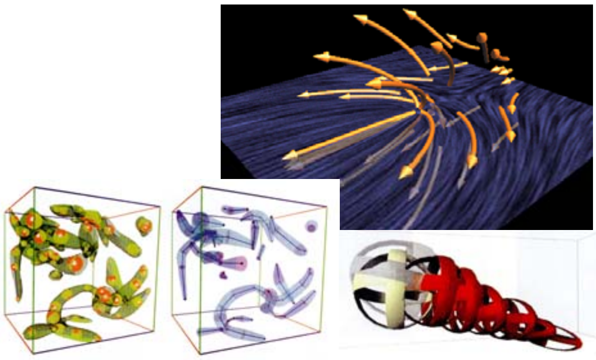




Icons in 2D

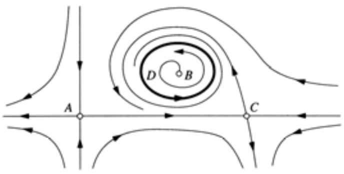


Icons & Glyphs in 3D



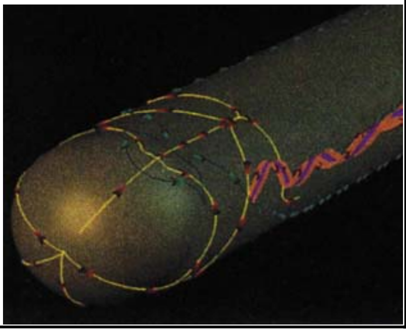
Flow Topology

- Topology:
  - abstract structure of a flow
  - different elements, e.g.:
    - checkpoints, defined through **v**(**x**)=**0**
    - cycles, defined through $\mathbf{s}_\mathbf{x}(t+T)=\mathbf{s}_\mathbf{x}(t)$
    - connecting structures (separatrices, etc.)



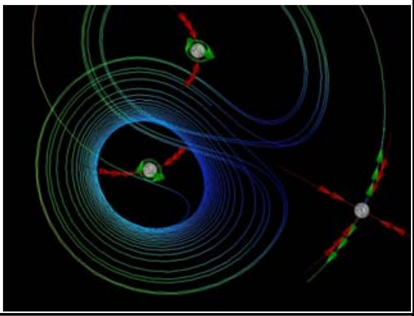
Flow Topology in 3D

- Topology on surfaces:
  - fixed points
  - separa-trices



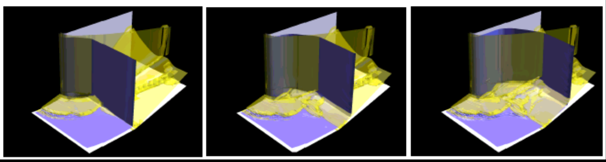
Flow Topology in 3D

- Lorenz system:
  - 1 saddle
  - 2 saddle foci
  - 1 chaotic attractor



Timesurfaces

- Idea:
  - start surface, e.g. part of a plane
  - move whole surface along flow over time
  - time surface: surface at one point in time



Literature, References

- B. Jobard & W. Lefer: "**Creating Evenly-Spaced Streamlines of Arbitrary Density**" in *Proceedings of 8th Eurographics Workshop on Visualization in Scientific Computing*, April 1997, pp. 45-55
- B. Cabral & L. Leedom: "**Imaging Vector Fields Using Line Integral Convolution**" in *Proceedings of SIGGRAPH '93* = Computer Graphics 27, 1993, pp. 263-270
- D. Stalling & H.-C. Hege: "**Fast and Resolution Independent Line Integral Convolution**" in *Proceedings of SIGGRAPH '95* = Computer Graphics 29, 1995, pp. 249-256

Acknowledgements

- For material for this lecture unit
  - Hans-Georg Pagendarm
  - Roger Crawfis
  - Lloyd Treinish
  - David Kenwright
  - Terry Hewitt
  - Bruno Jobard
  - Malte Zöckler
  - Georg Fischel
  - Helwig Hauser
  - Bruno Jobard
  - Jeff Hultquist
  - Lukas Mroz, Rainer Wegenkittl
  - Nelson Max, Will Schroeder et al.
  - Brian Cabral & Leith Leedom
  - David Kenwright
  - Rüdiger Westermann
  - Jack van Wijk, Freik Reinders, Frits Post, Alexandru Telea, Ari Sadarjoen