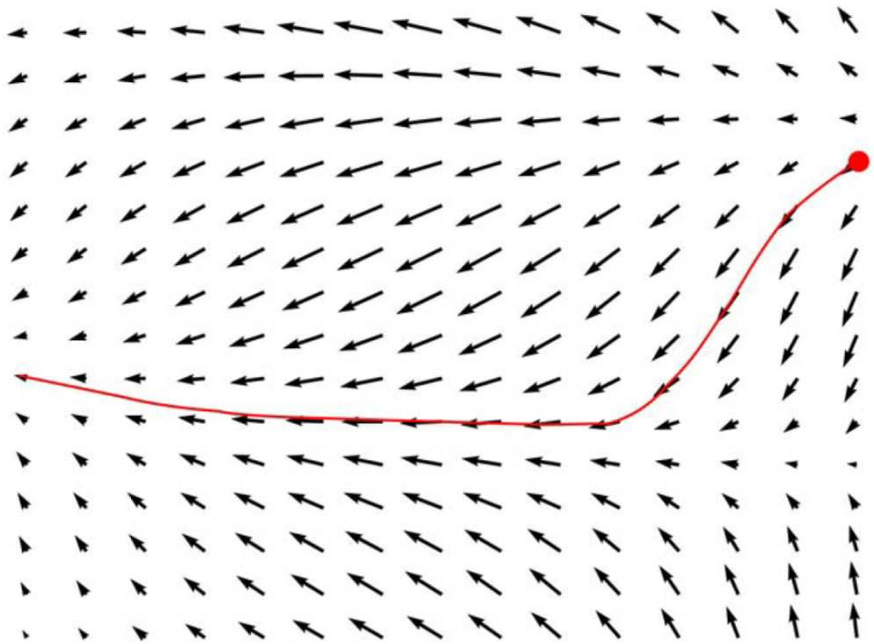
Vector Field Visualization: Introduction

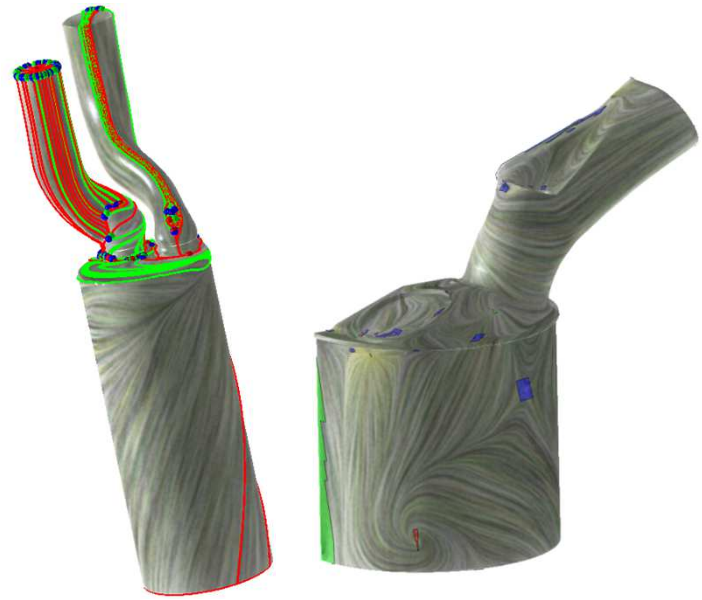
What is a Vector Field?



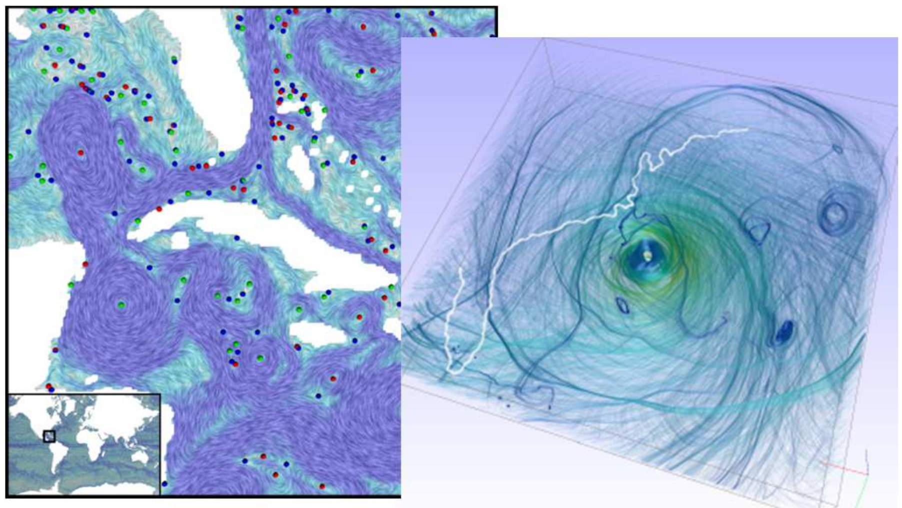
$$\frac{d\varphi(x)}{dt} = V(x)$$

Why It is Important?

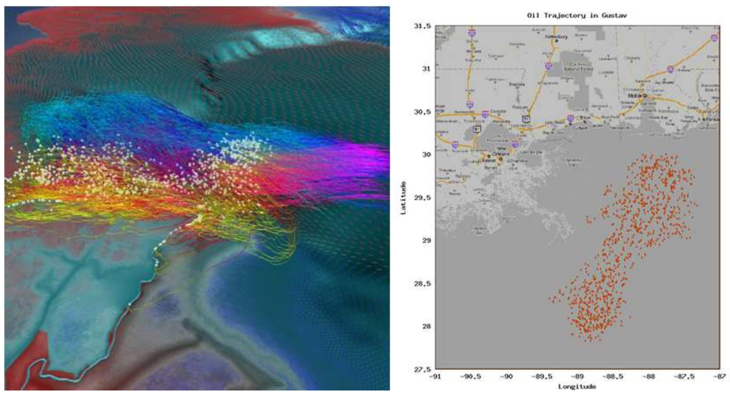
Vector Fields in Engineering and Science



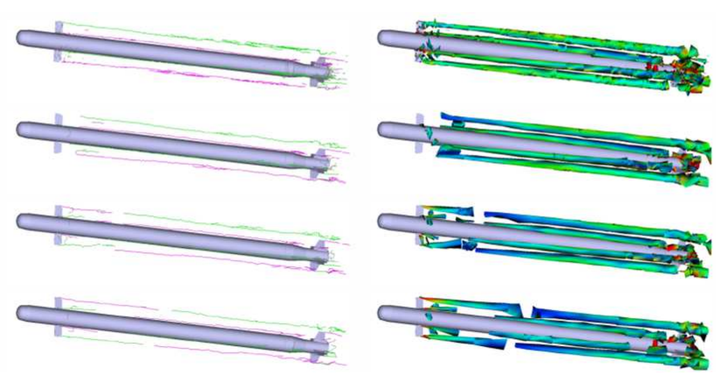
Automotive design
[**Chen** et al. TVCG07,TVCG08]



Weather study [Bhatia and Chen et al. TVCG11]

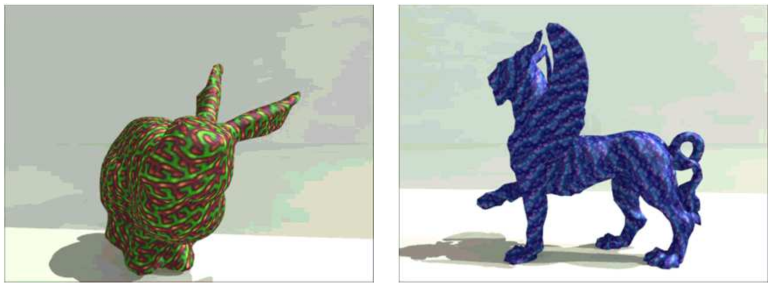


Oil spill trajectories [Tao et al. EMI2010]

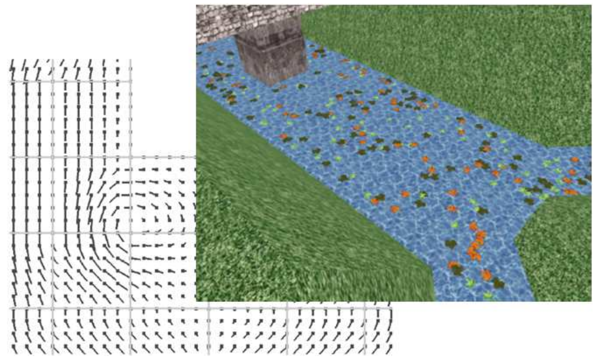


Aerodynamics around missiles [Kelly et al. Vis06]

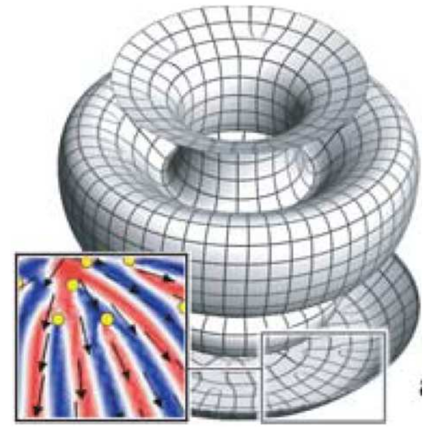
Vector Field Design in Computer Graphics



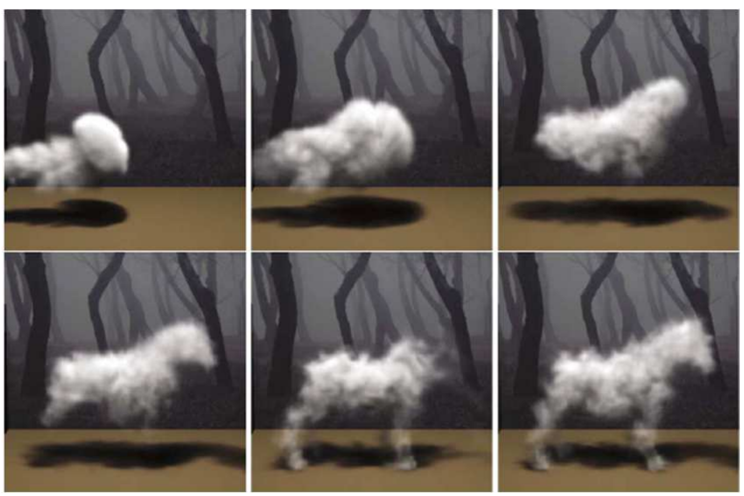
Texture Synthesis [**Chen** et al. TVCG11b]



River simulation [Chenney SCA2004]

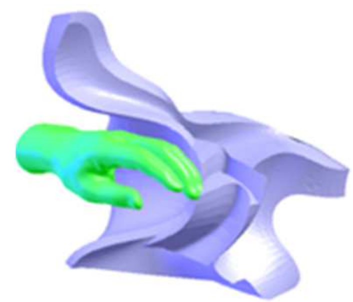


Parameterization
[Ray et al. TOG2006]



Smoke simulation [Shi and Yu TOG2005]

Painterly Rendering [Zhang et al. TOG2006]

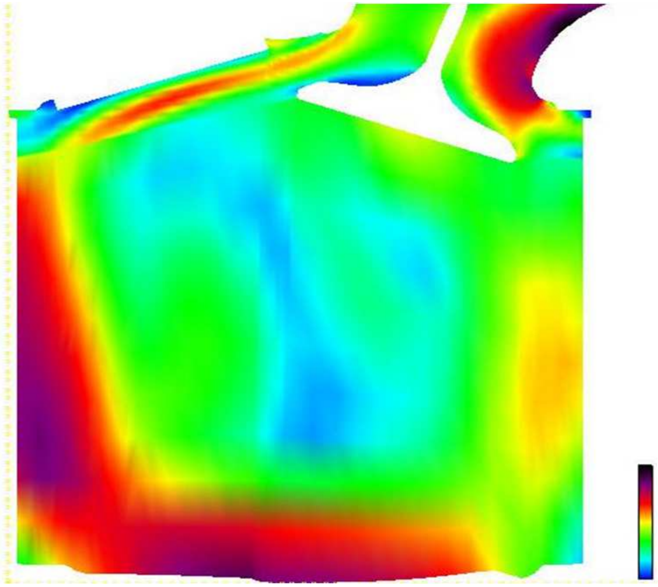


Shape Deformation
[von Funck et al. 2006]

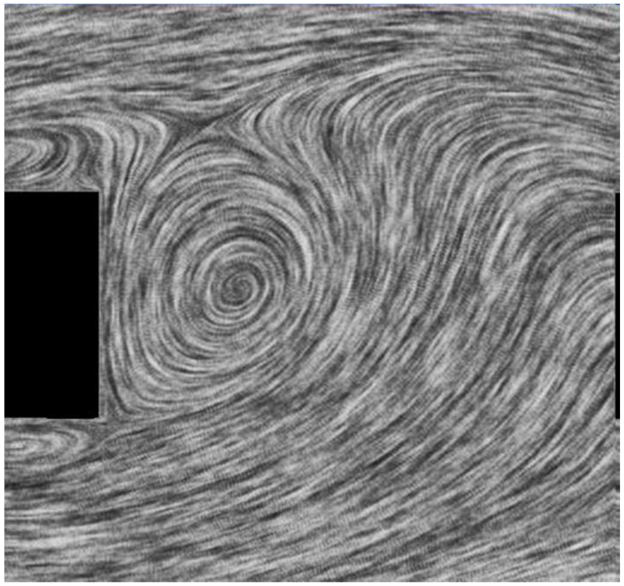
Why is It Challenging?

- to effectively visualize both *magnitude + direction,* often simultaneously
- large data sets
- time-dependent data

magnitude only

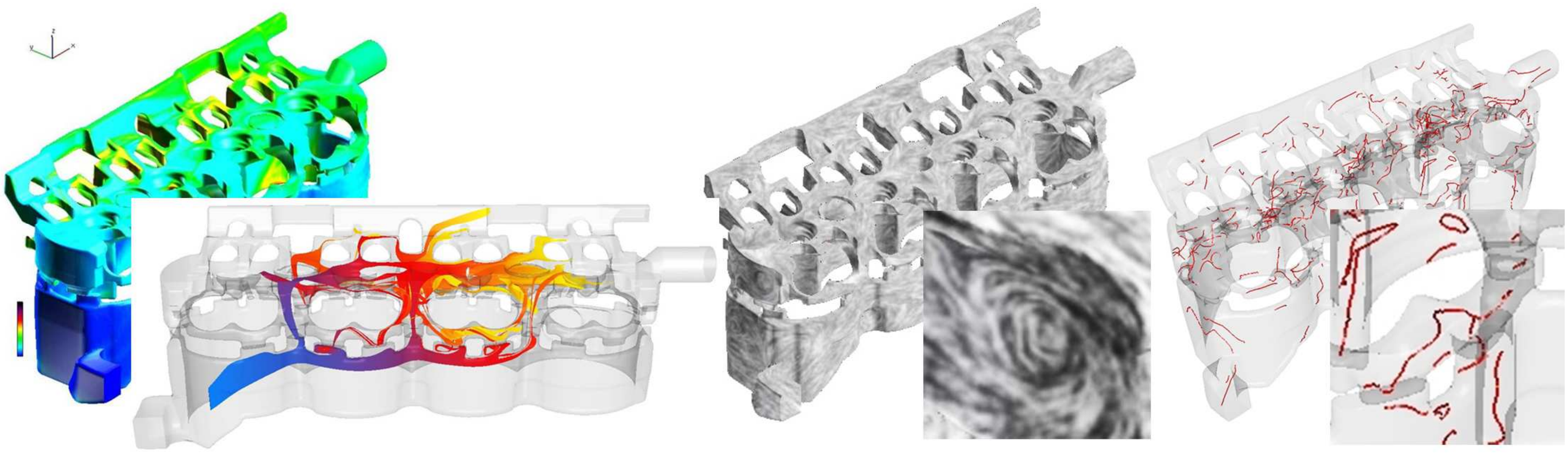


direction only



Classification of Visualization Techniques

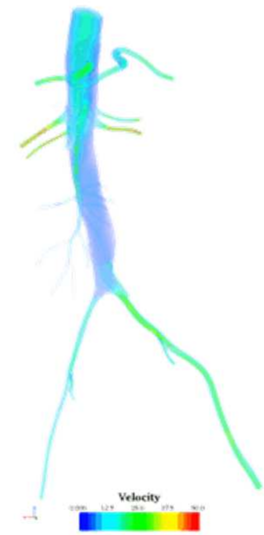
- **Direct:** overview of vector field, minimal computation, e.g. glyphs, color mapping
- **Texture-based:** covers domain with a convolved texture, e.g., Spot Noise, LIC, ISA, IBFV(S)
- **Geometric:** a discrete object(s) whose geometry reflects flow characteristics, e.g. streamlines
- **Feature-based:** both automatic and interactive feature-based techniques, e.g. flow topology



Flow Data

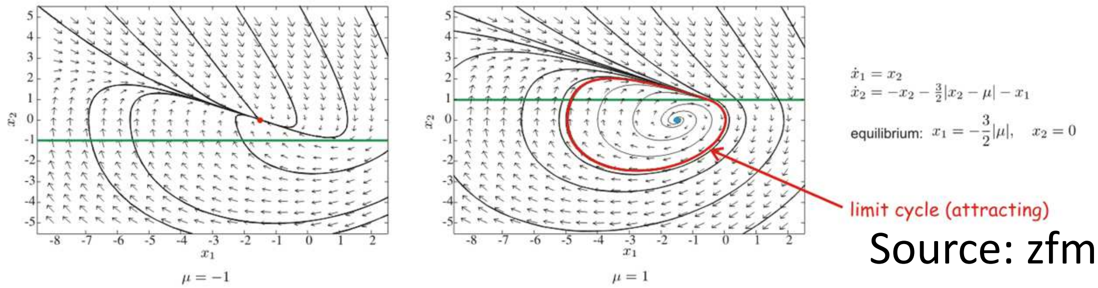
Data sources:

- flow simulation:
  - airplane- / ship- / car-design
  - weather simulation (air-, sea-flows)
  - medicine (blood flows, etc.)
- flow measurement:
  - wind tunnels, water channels
  - optical measurement techniques
- flow models (analytic):
  - differential equation systems
    (dynamic systems)



Source: simtk.org

Source: speedhunter.com



Source: zfm.ethz.ch

Flow Data

Simulation:

- flow: estimate (partial) differential equation systems
- set of samples (n-dims. of data), e.g., given on a curvilinear grid
- most important primitive: tetrahedron and hexahedron (cell)
- could be adaptive grids

Analytic:

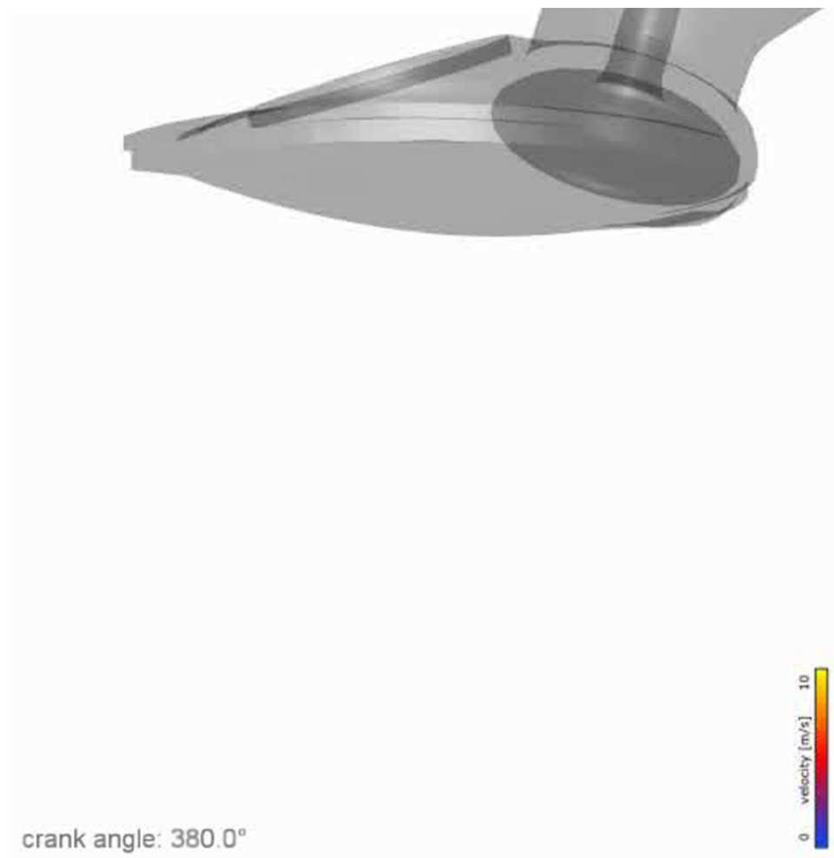
- flow: analytic formula, differential equation systems d**x**/dt (dynamical system)
- evaluated where ever needed

Measurement:

- vectors: taken from instruments, often computed on a uniform grid
- optical methods + image recognition, e.g.: PIV (particle image velocimetry)

Notes on Computational Fluid Dynamics

- We often visualize Computational Fluid Dynamics (CFD) simulation data
- CFD is the discipline of predicting flow behavior, quantitatively
- data is (often) the result of a simulation of flow through or around an object of interest



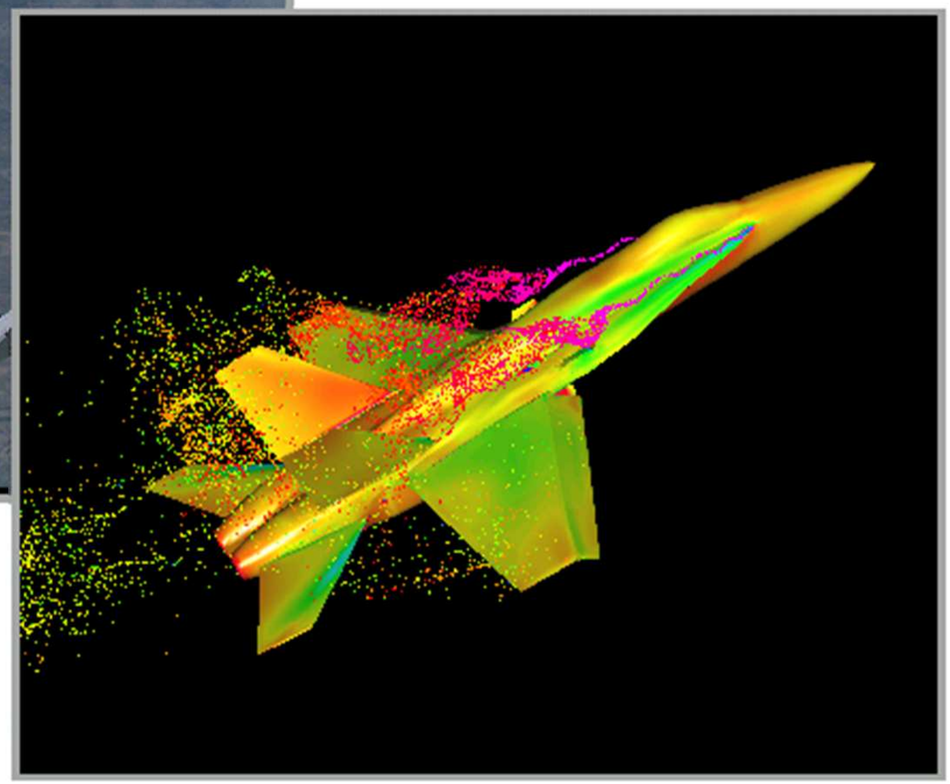
some characteristics of CFD data:

- large, often gigabytes
- Unsteady, i.e. time-dependent
- unstructured, adaptive resolution grids
- Smooth field

Comparison with Reality

Experiment

Simulation



2D vs. 2.5D Surfaces vs. 3D

2D flow visualization

- 2Dx2D flows
- models, flow layers (2D section through 3D)

2.5D, i.e. surface flow visualization

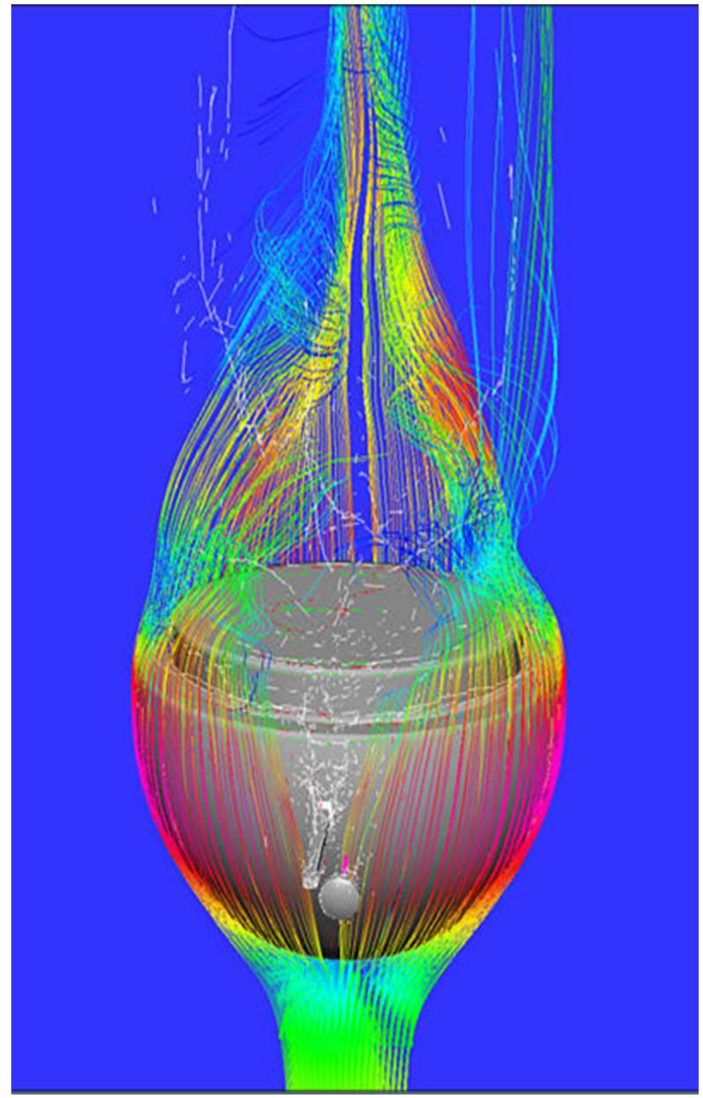
- 3D flows around obstacles
- boundary flows on manifold surfaces (locally 2D)

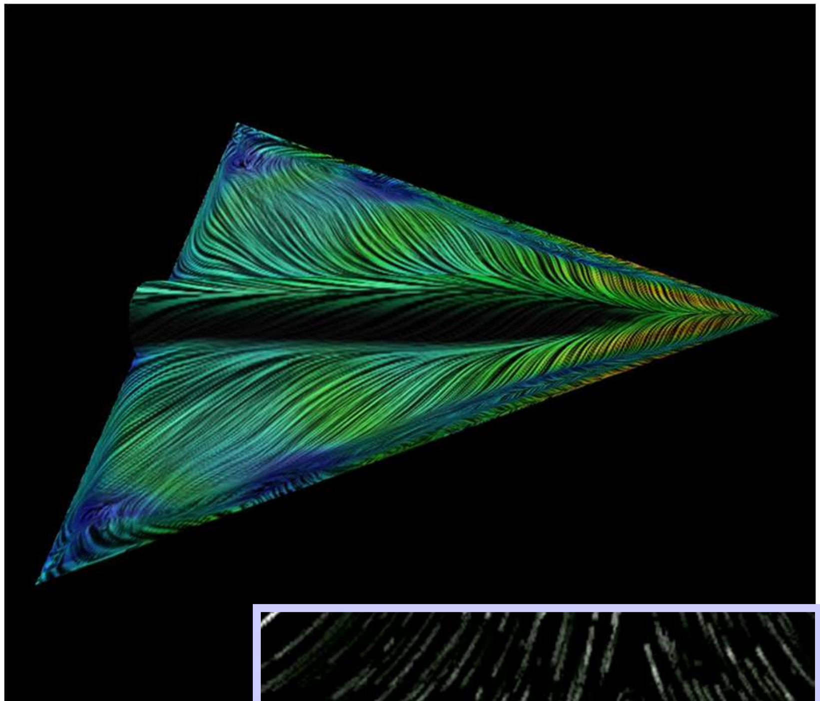
3D flow visualization

- 3Dx3D flows
- simulations, 3D models

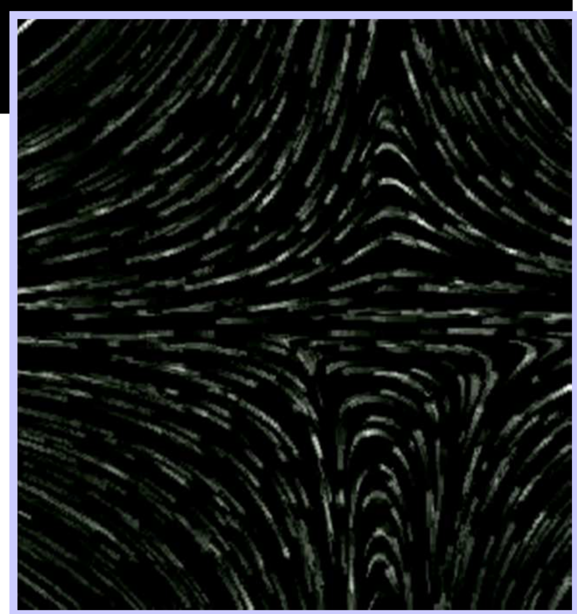
2D/Surfaces/3D – Examples

3D





Surface



2D

Steady vs. Time-dependent

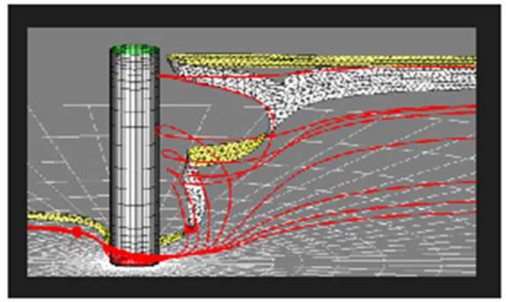
Steady (time-independent) flows:

- flow itself constant over time
- $\mathbf{v}(\mathbf{x})$, e.g., laminar flows
- well understood behaviors
- simpler case for visualization

Time-dependent (unsteady) flows:

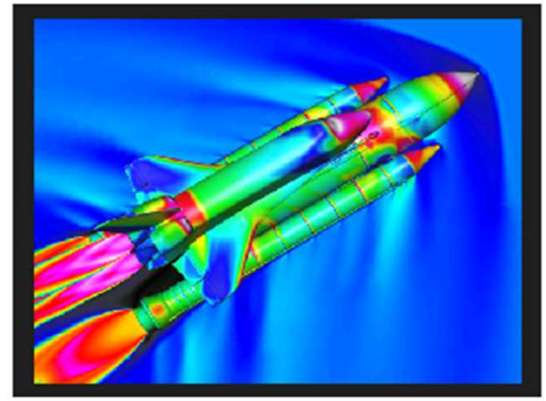
- flow itself changes over time
- $\mathbf{v}(\mathbf{x},t)$, e.g., combustion flow, turbulent flow
- more complex cases
- no uniform theory to characterize them yet!

Time-independent (steady) Data



Single Zone
100K Nodes
4 MB

(1985)



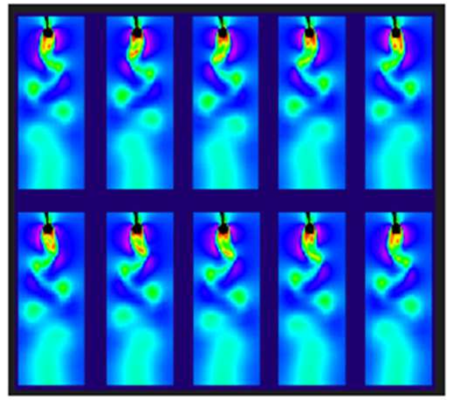
128 Zones
30M Nodes
1080 MB

(1996)

- Dataset sizes over years:

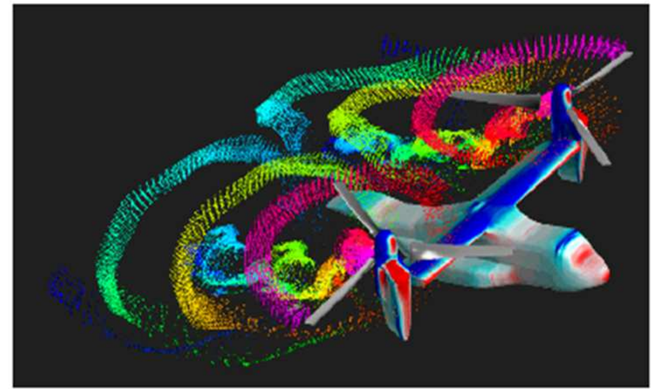
Data set name and year	Number of vertices	Size (MB)
McDonnell Douglas MD–80 '89	230,000	13
McDonnell Douglas F/A–18 '91	900,000	32
Space shuttle launch vehicle '90	1,000,000	34
Space shuttle launch vehicle '93	6,000,000	216
Space shuttle launch vehicle '96	30,000,000	1,080
Advanced subsonic transport '98	60,000,000	2,160
Army UH–60 Blackhawk '99	100,000,000	~4,000

Time-dependent (unsteady) Data



Single Zone
128K Nodes
1 GB

(1990)



25 Zones (9 Moving)
2.8M Nodes
300 GB

(1996)

- Dataset sizes over time:

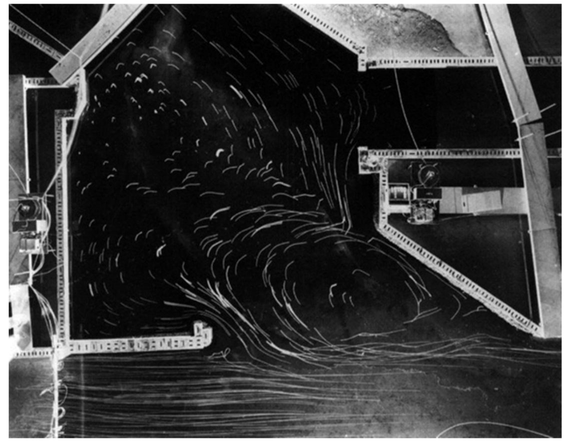
Data set name and year	# vertices	# time steps	size (MB)
Tapered Cylinder '90	131,000	400	1,050
McDonnell Douglas F/A–18 '92	1,200,000	400	12,800
Descending Delta Wing '93	900,000	1,800	64,800
Bell–Boeing V–22 tiltrotor '93	1,300,000	1,450	140,000
Bell–Boeing V–22 tiltrotor '98	10,000,000	1,450	600,000

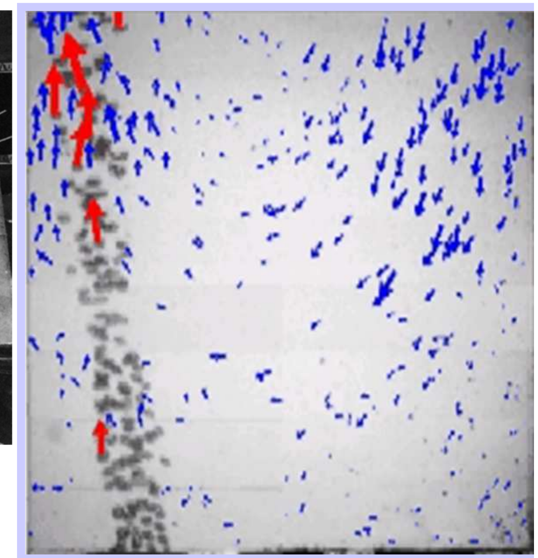
Experimental Flow Visualization

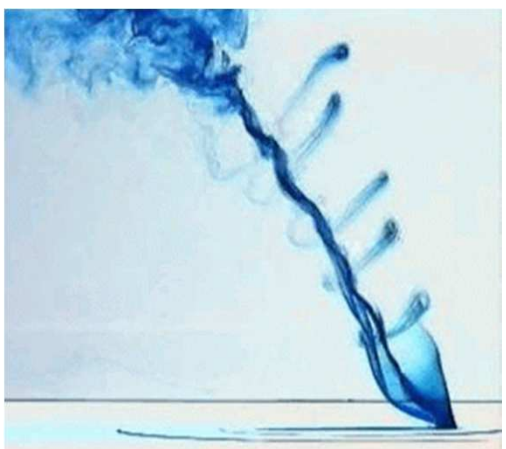
Optical Methods, etc.

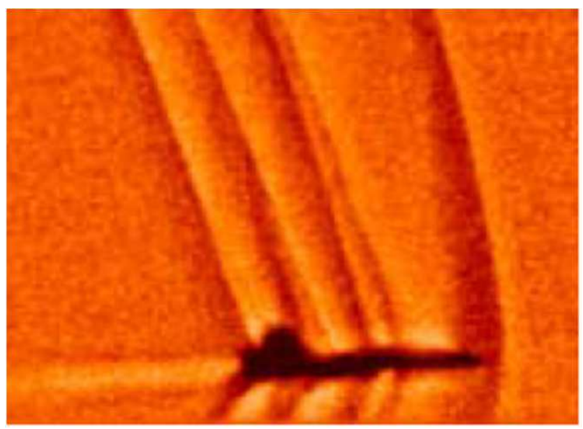
With Smoke or Dye

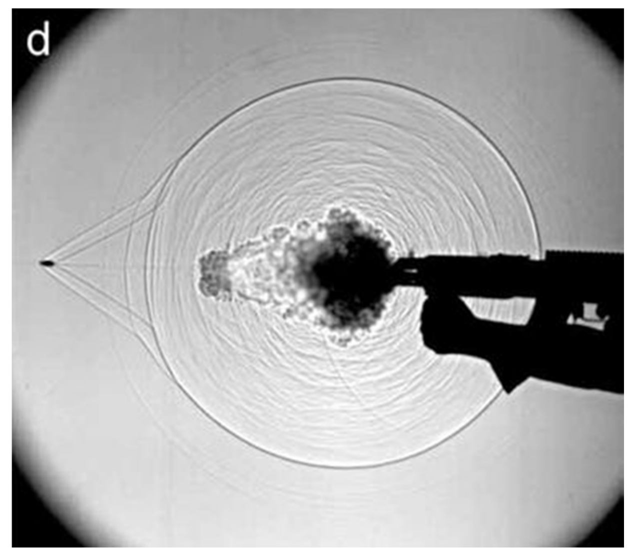
- Injection of dye, smoke, particles
- Optical methods:
  - transparent object with complex distribution of light refraction index
- Streaks, shadows



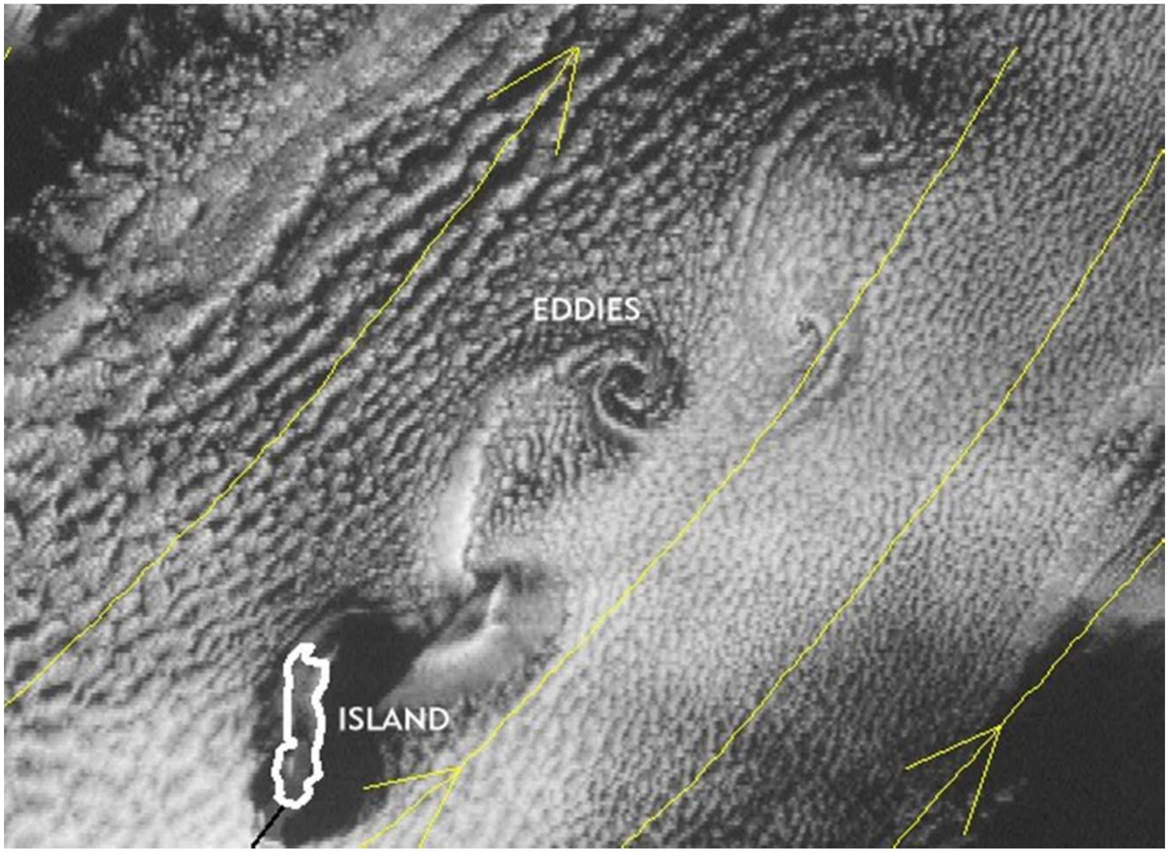








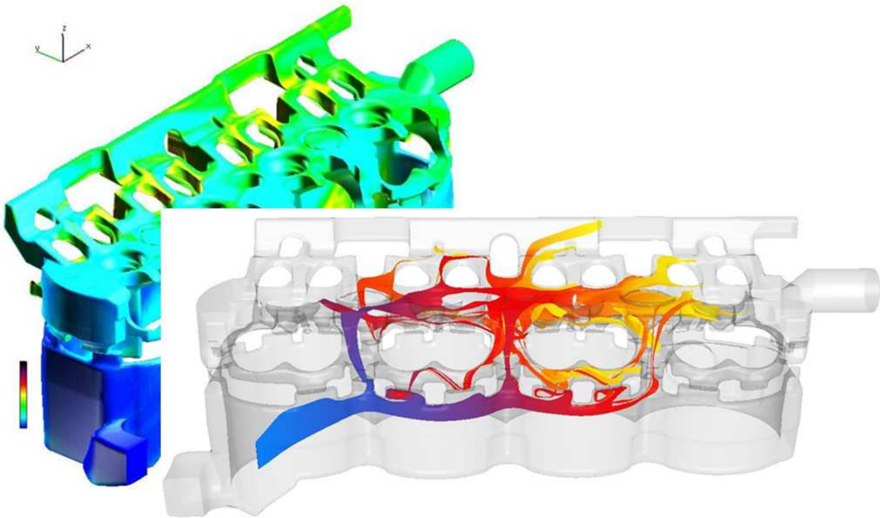
Large Scale Dying

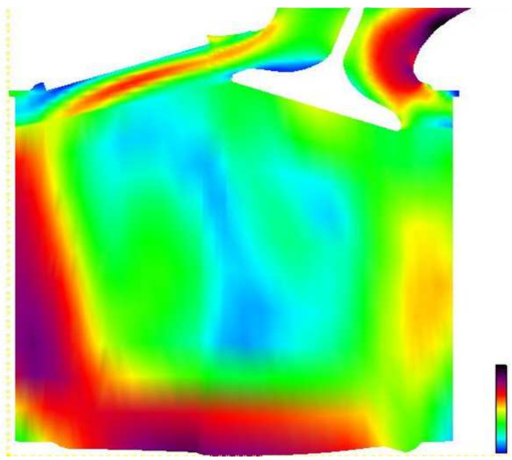


Source: ishtarsgate.com

Source: weathergraphics.com

Direct Methods

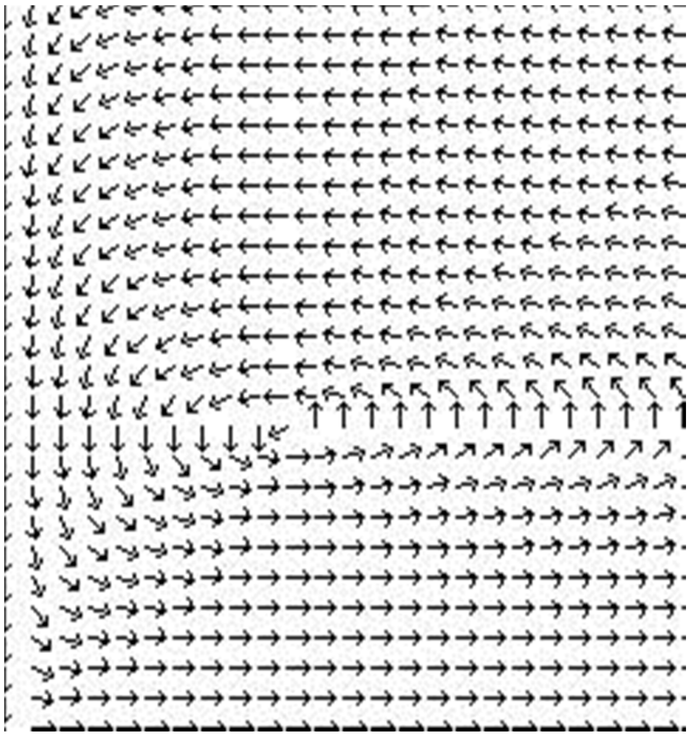




Direct FlowVis with Arrows

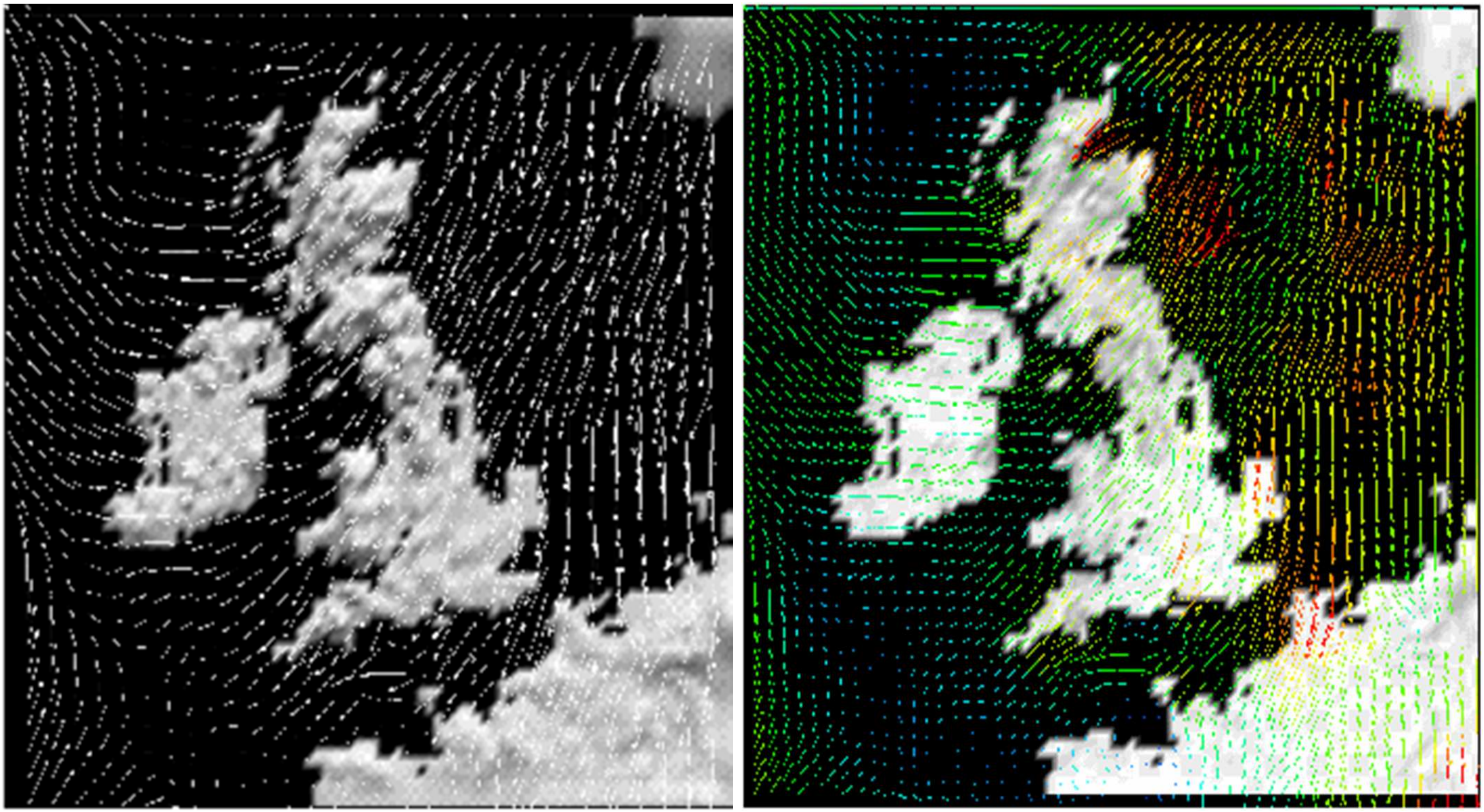
Properties:

- direct FlowVis
- frequently used!
- normalized arrows vs. velocity coding
- 2D: quite useful, 3D: often problematic
- often difficult to understand, mentally integrate (time component missing)



Arrows in 2D

Scaled arrows vs. color-coded arrows



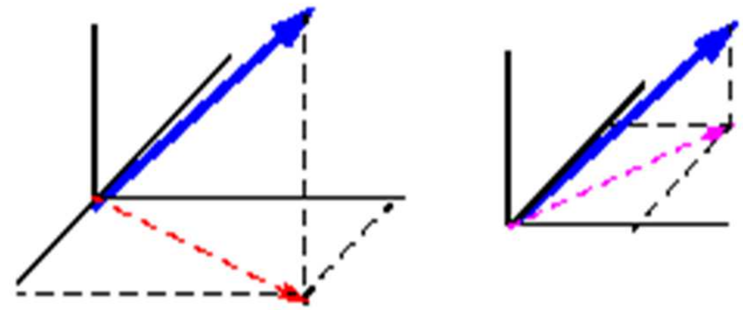
Arrows in 3D

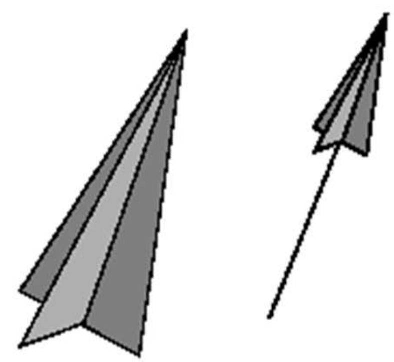
Common problems:

- ambiguity
- Perspective shortening
- 1D objects generally difficult to grasp in 3D

Remedy:

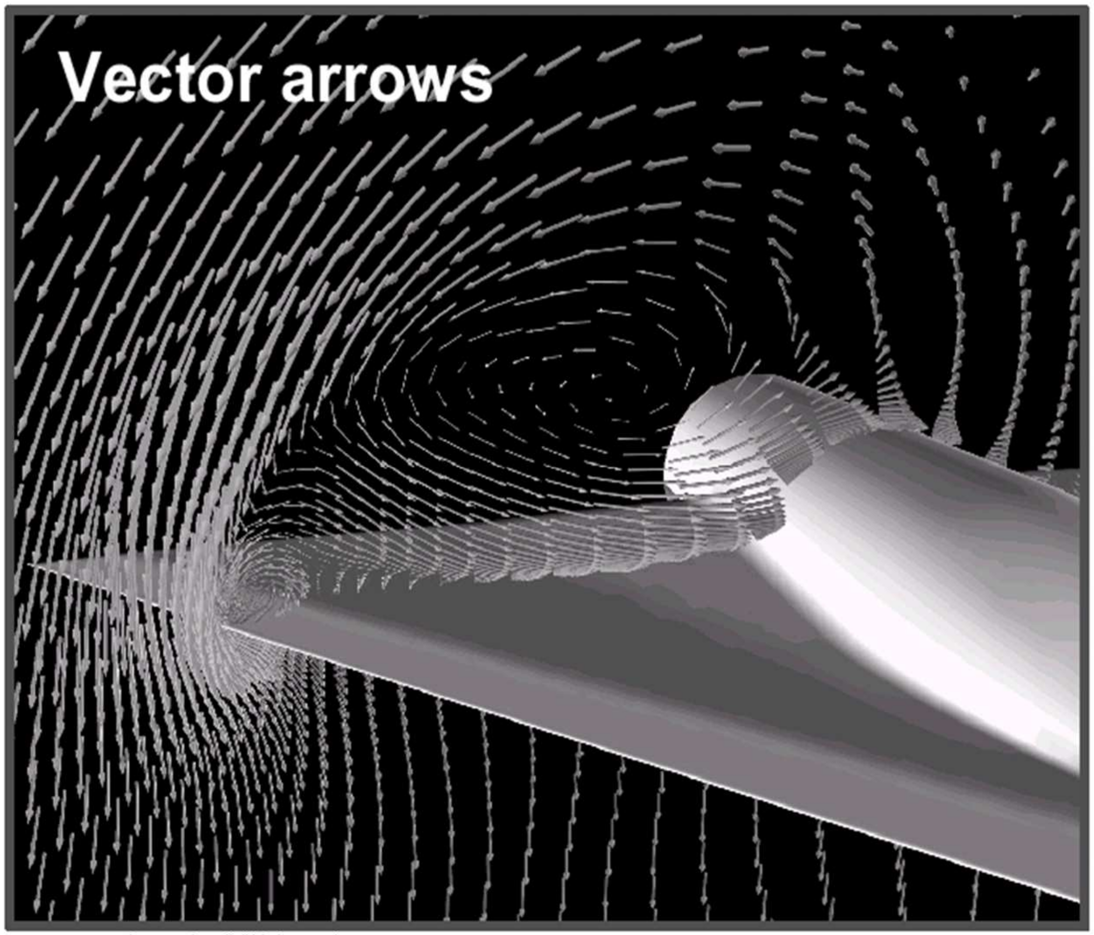
- 3D-Arrows
  (are of some help)



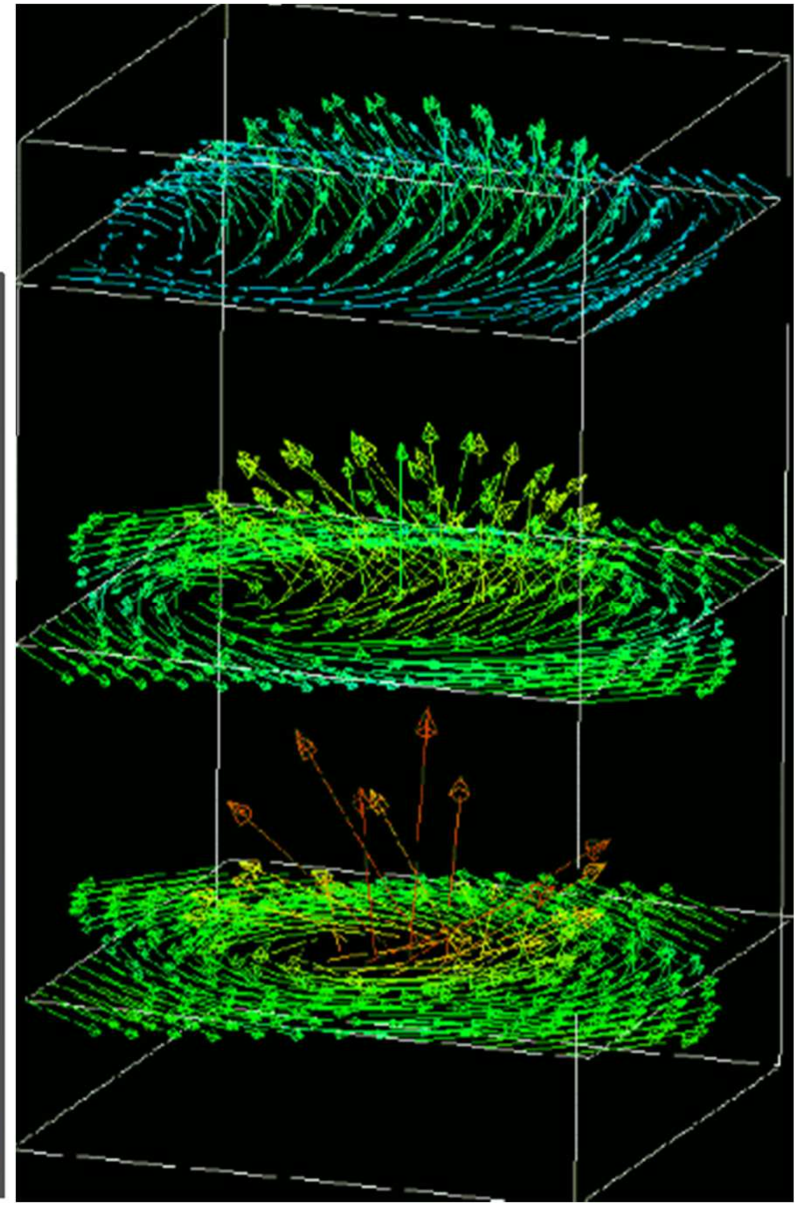


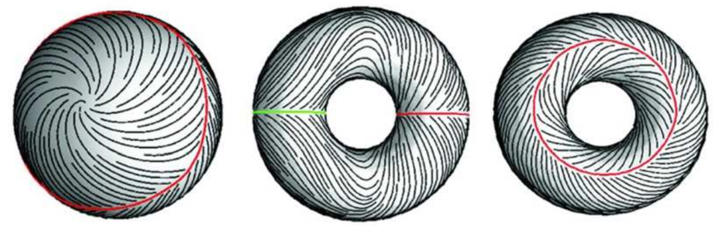
Arrows in 3D

Compromise:
arrows only in layers

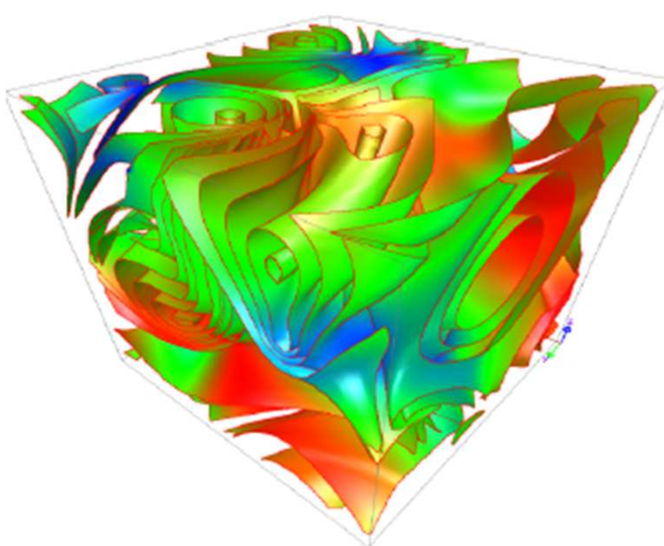


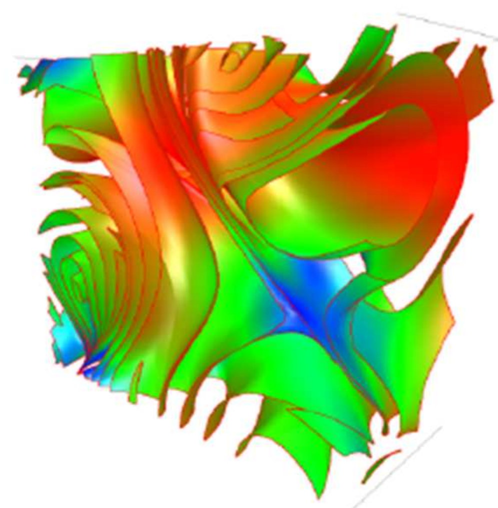
aching/csMU7-vis/



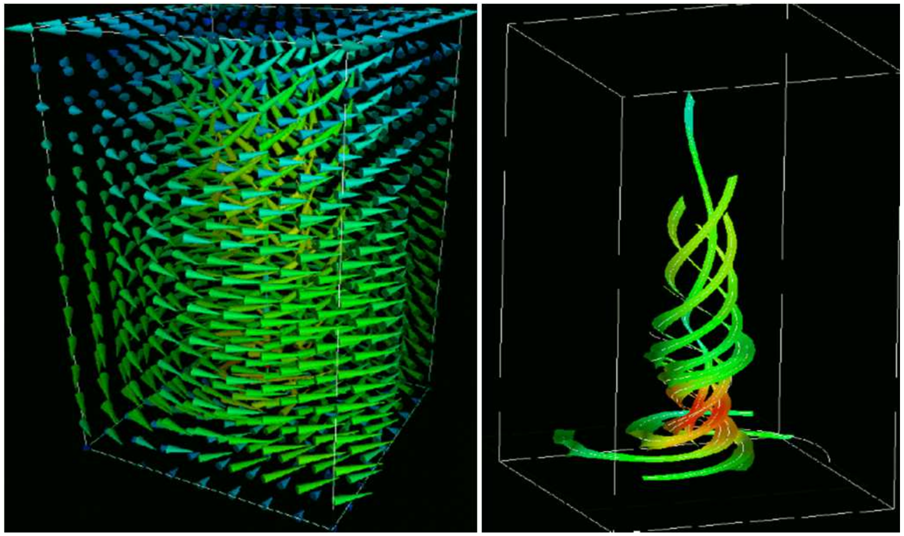


Geometric-based Methods: Integral curves and surfaces





Direct vs. Geometric FlowVis



Direct flow visualization:

- overview of current state of flow
- visualization with vectors popular
- arrows, icons, glyph techniques

Geometric flow visualization:

- use of intermediate objects,
  e.g., after vector field integration over time
- visualization of development over time
- streamlines, stream surfaces
- analogous to indirect (vs. direct) volume visualization

Streamlines – Theory

Correlations:

- flow data **v**: derivative information
  - $d\mathbf{x}/dt = \mathbf{v}(\mathbf{x})$;
    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
  - $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \leq u \leq t} \mathbf{v}(\mathbf{s}(u))\, du$;
    seed point $\mathbf{s}_0$, integration variable u

- Property:
  - uniqueness

- difficulty: result **s** also in the integral $\Rightarrow$ analytical solution usually impossible.

Streamlines – Practice

Basic approach:

- theory: $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \le u \le t} \mathbf{v}(\mathbf{s}(u))\, \mathrm{d}u$
- 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: $\mathbf{s}_{i+1} = \mathbf{s}_i + \mathbf{v}(\mathbf{s}_i) \cdot dt$,
integration of small steps (dt very small)

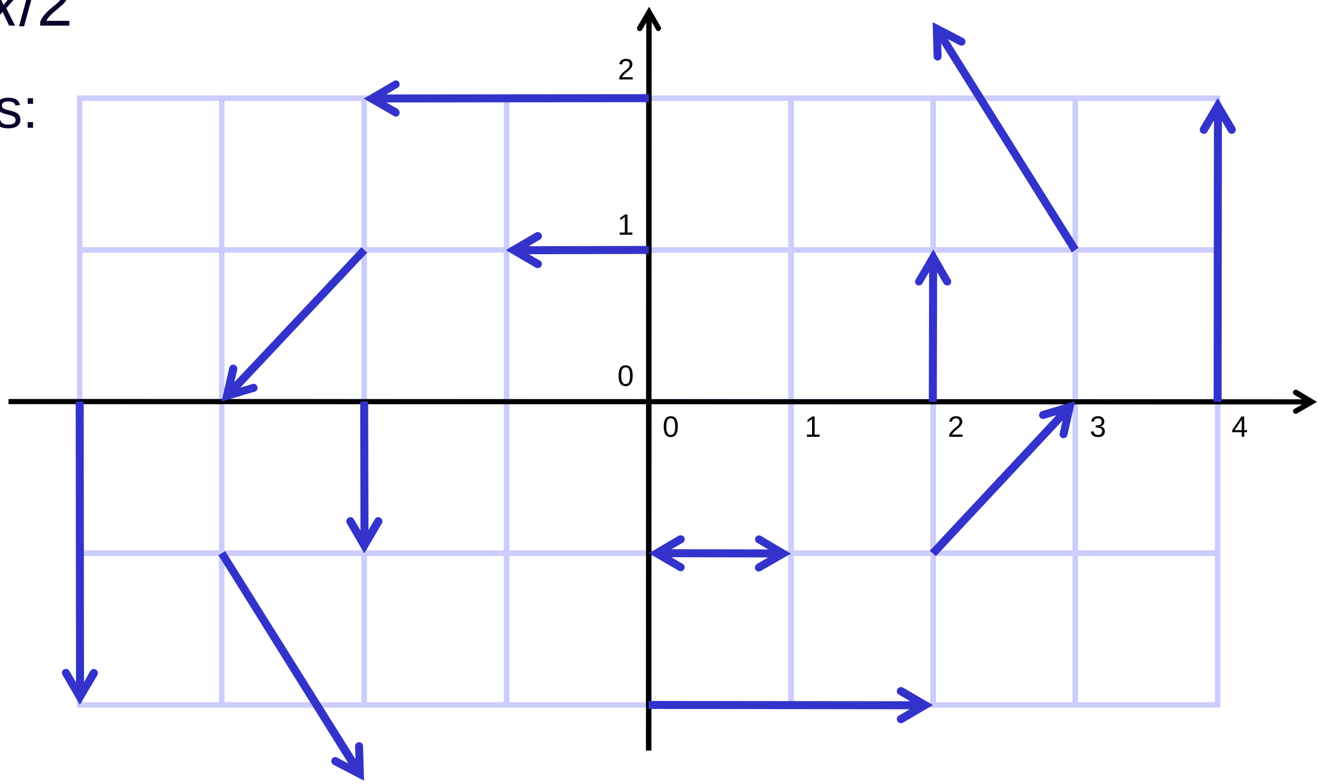
Euler Integration – Example

2D model data:

$v_x = dx/dt = -y$

$v_y = dy/dt = x/2$

Sample arrows:



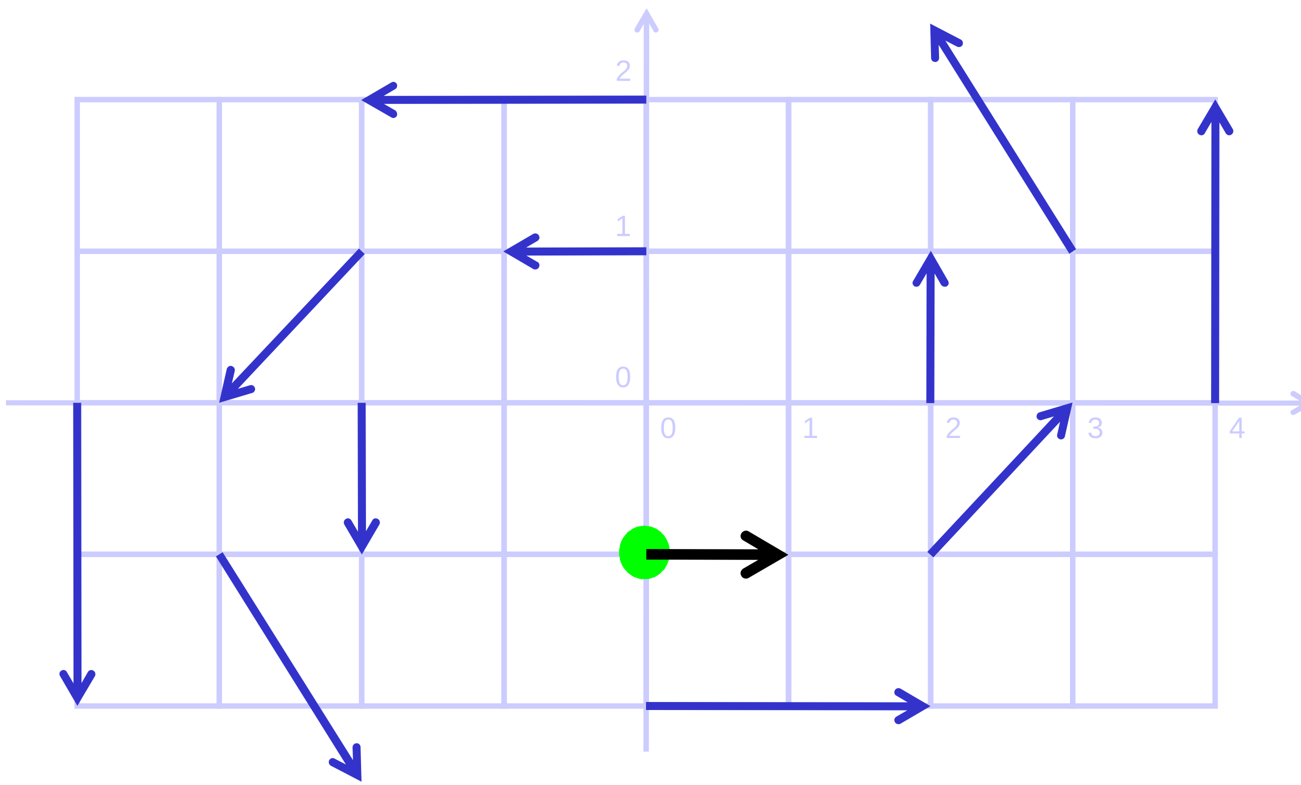
True solution: ellipses.

Euler Integration – Example

- Seed point $\mathbf{s}_0 = (0|-1)^T$;
current flow vector $\mathbf{v}(\mathbf{s}_0) = (1|0)^T$;
$\mathrm{d}t = ½$

$v_x = \mathrm{d}x/\mathrm{d}t = -y$

$v_y = dy/dt = x/2$

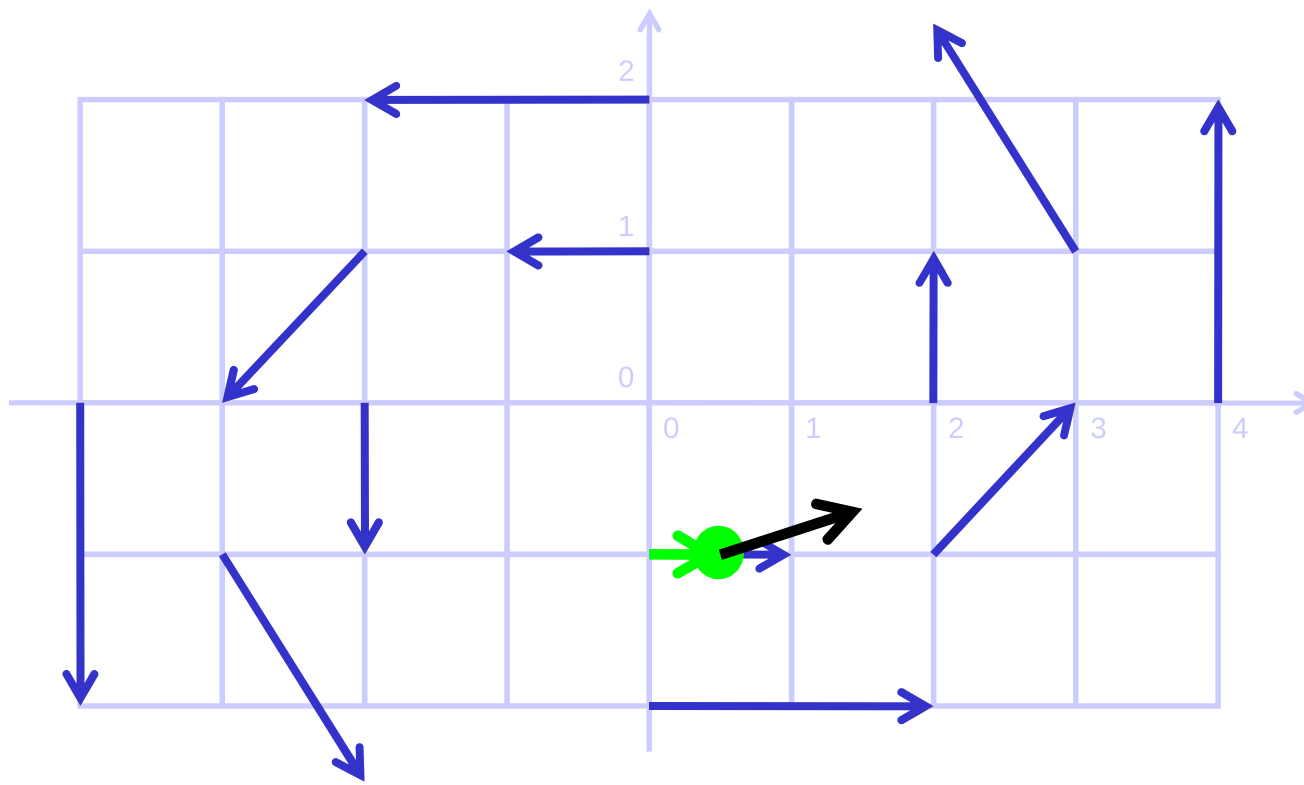


Euler Integration – Example

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

$v_x = dx/dt = -y$

$v_y = dy/dt = x/2$

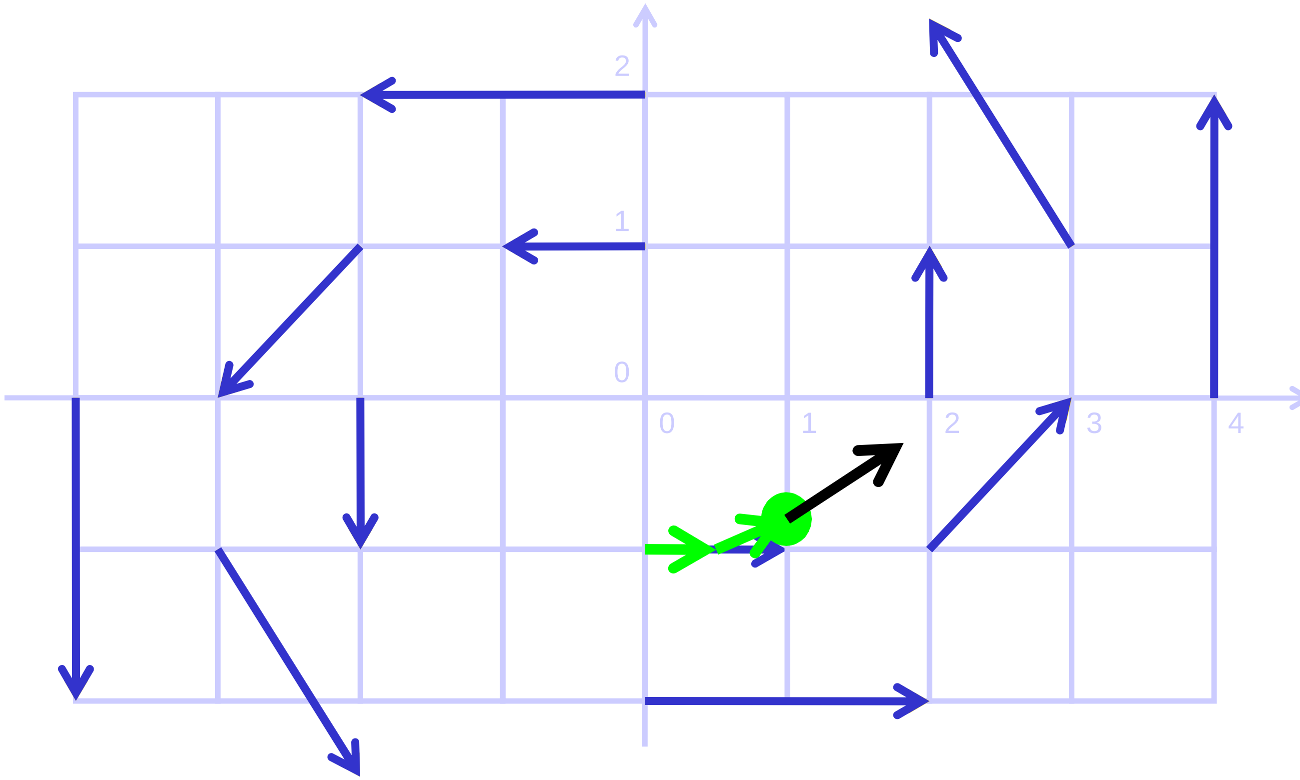


Euler Integration – Example

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

$v_x = dx/dt = -y$

$v_y = dy/dt = x/2$



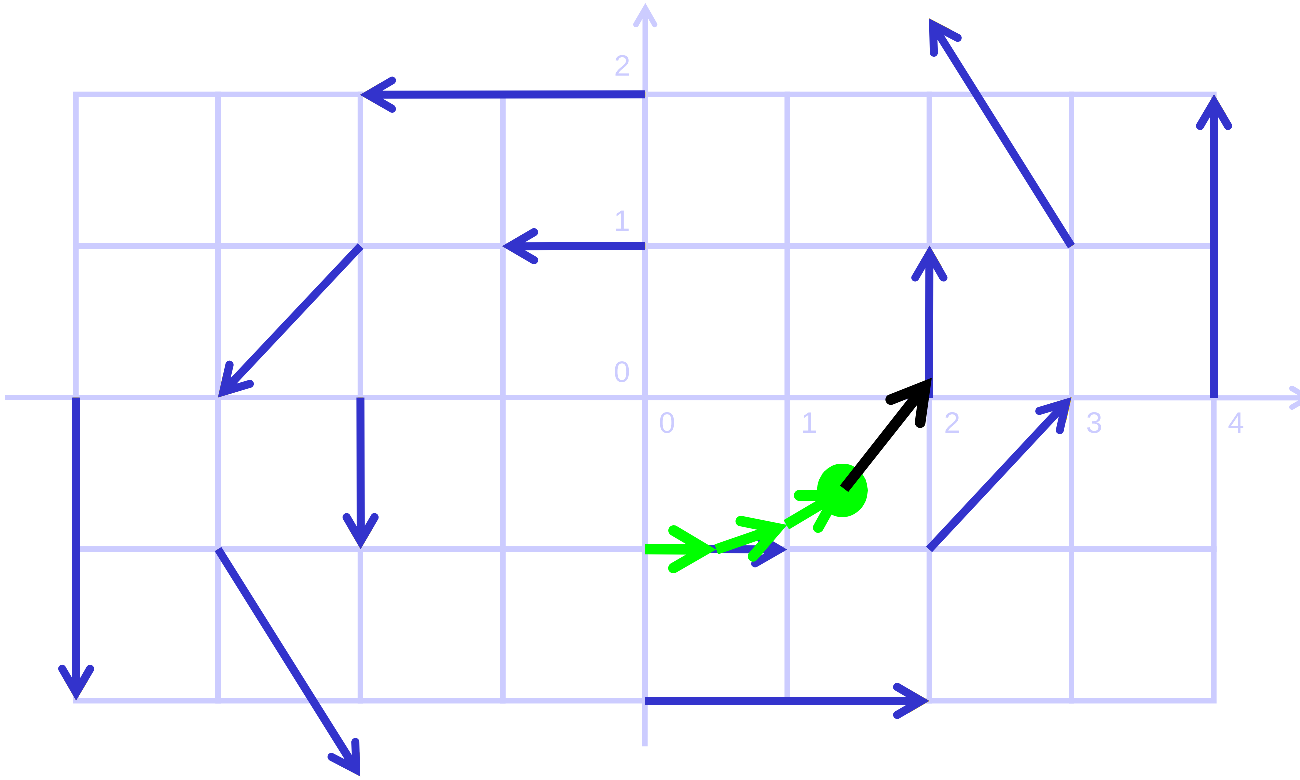
Euler Integration – Example

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

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

$v_x = \mathrm{d}x/\mathrm{d}t = -y$

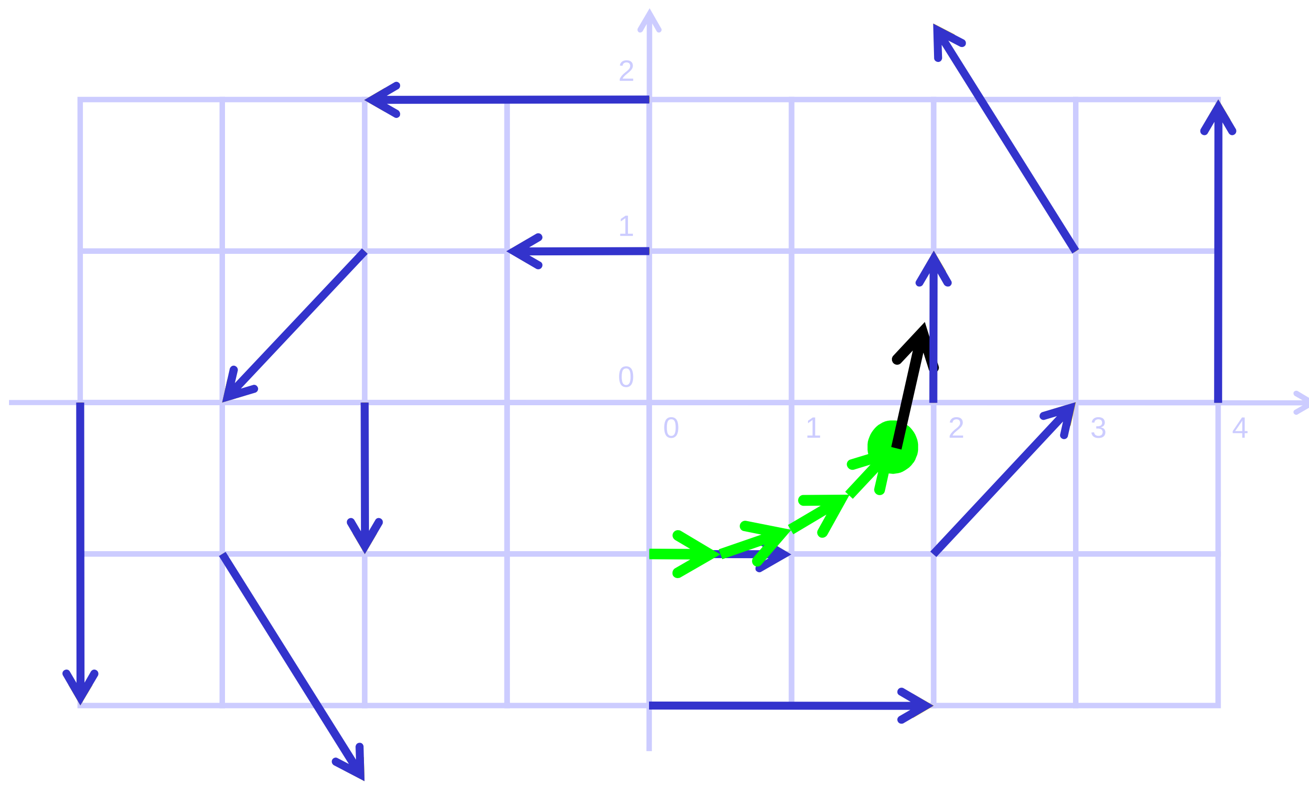
$v_y = dy/dt = x/2$



Euler Integration – Example

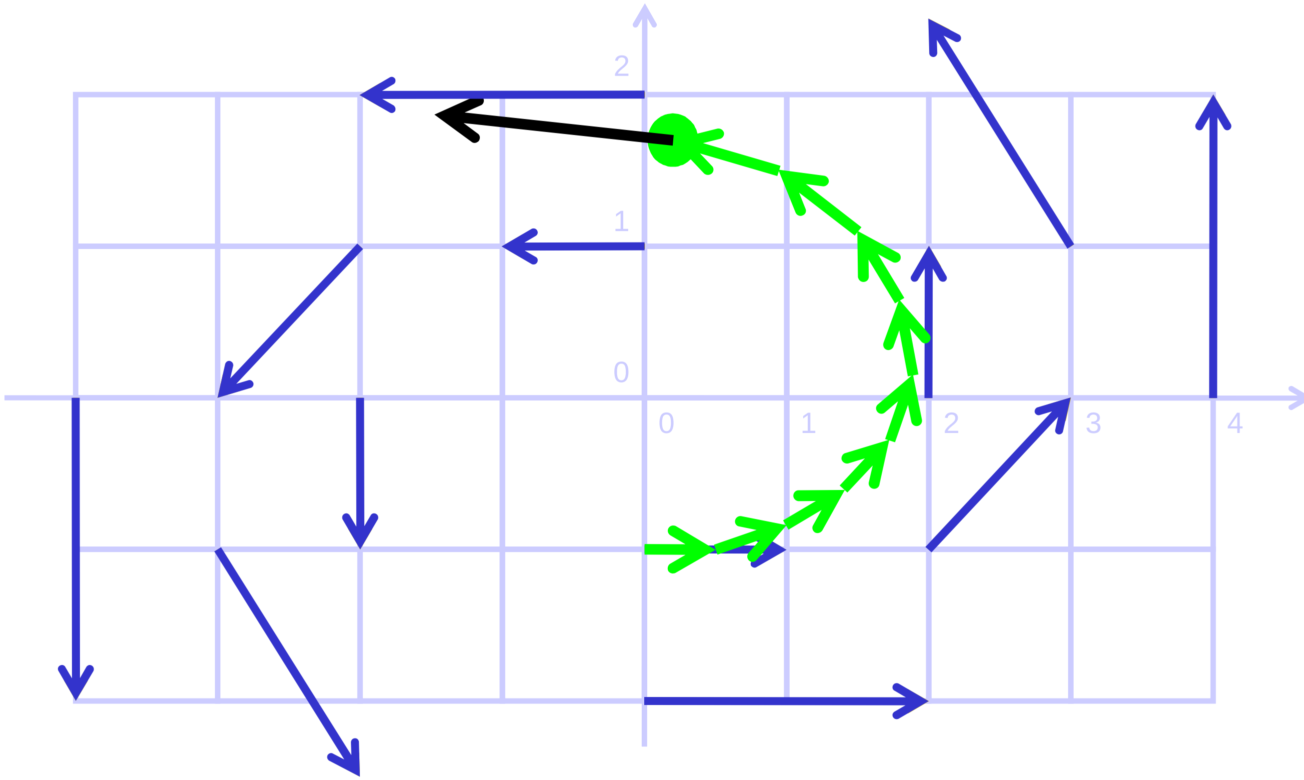
- $\mathbf{s}_4 \quad = (7/4|-17/64)^T \approx (1.75|-0.27)^T;$

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



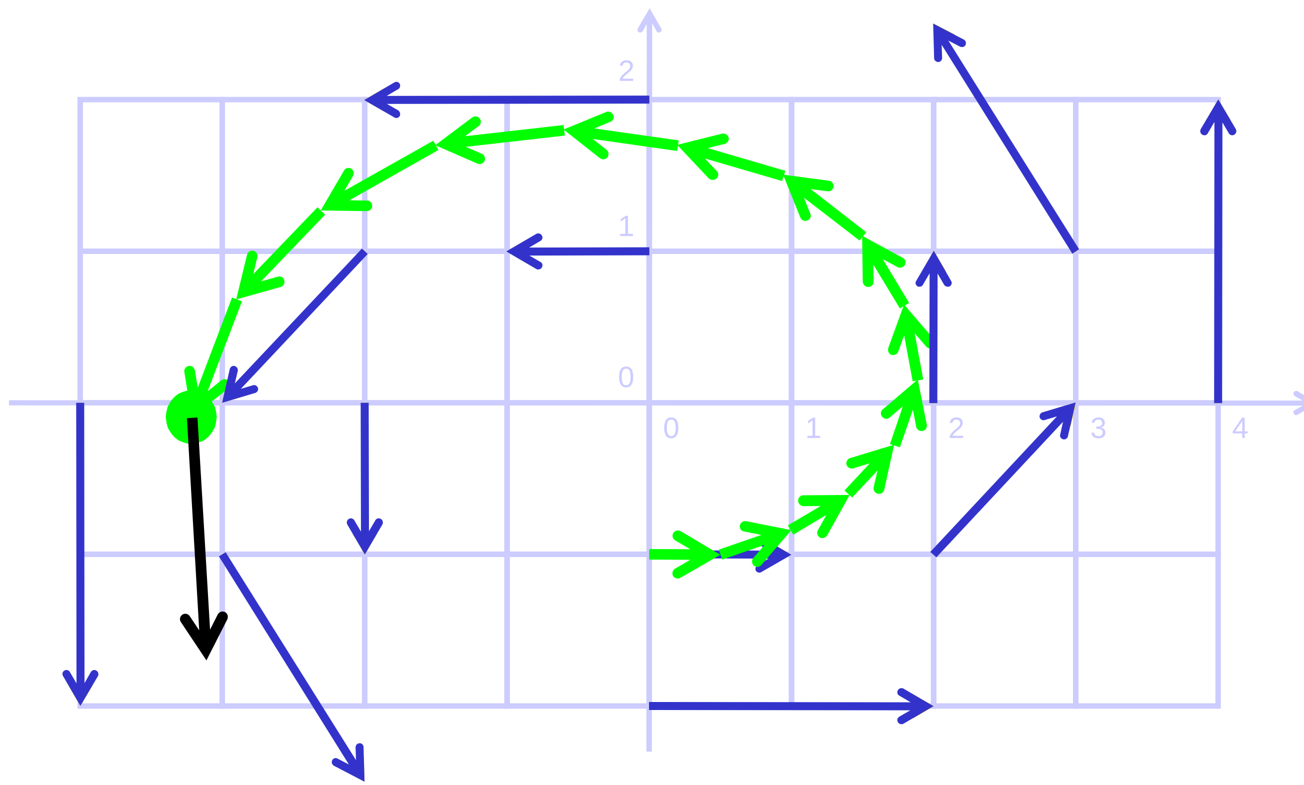
Euler Integration – Example

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



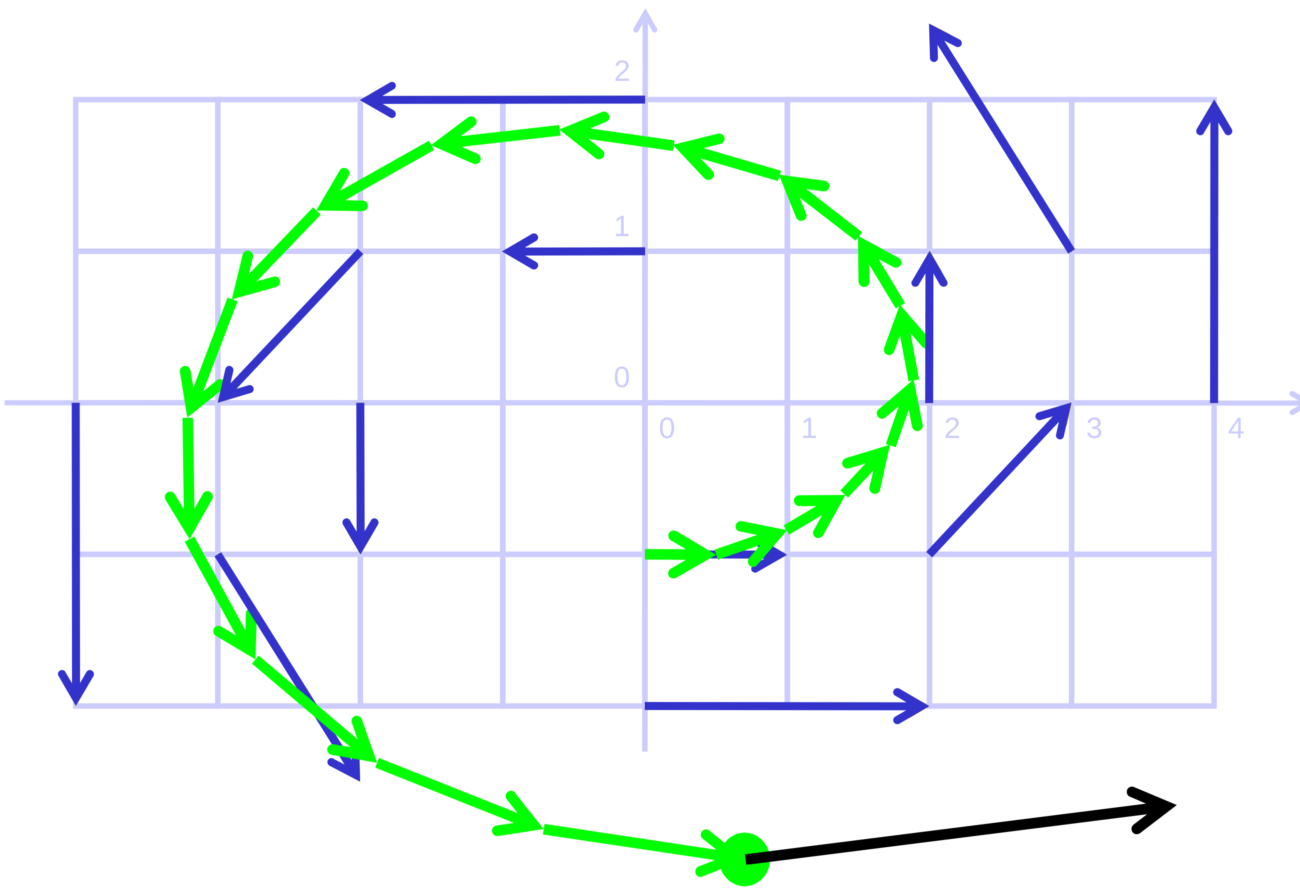
Euler Integration – Example

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



Euler Integration – Example

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

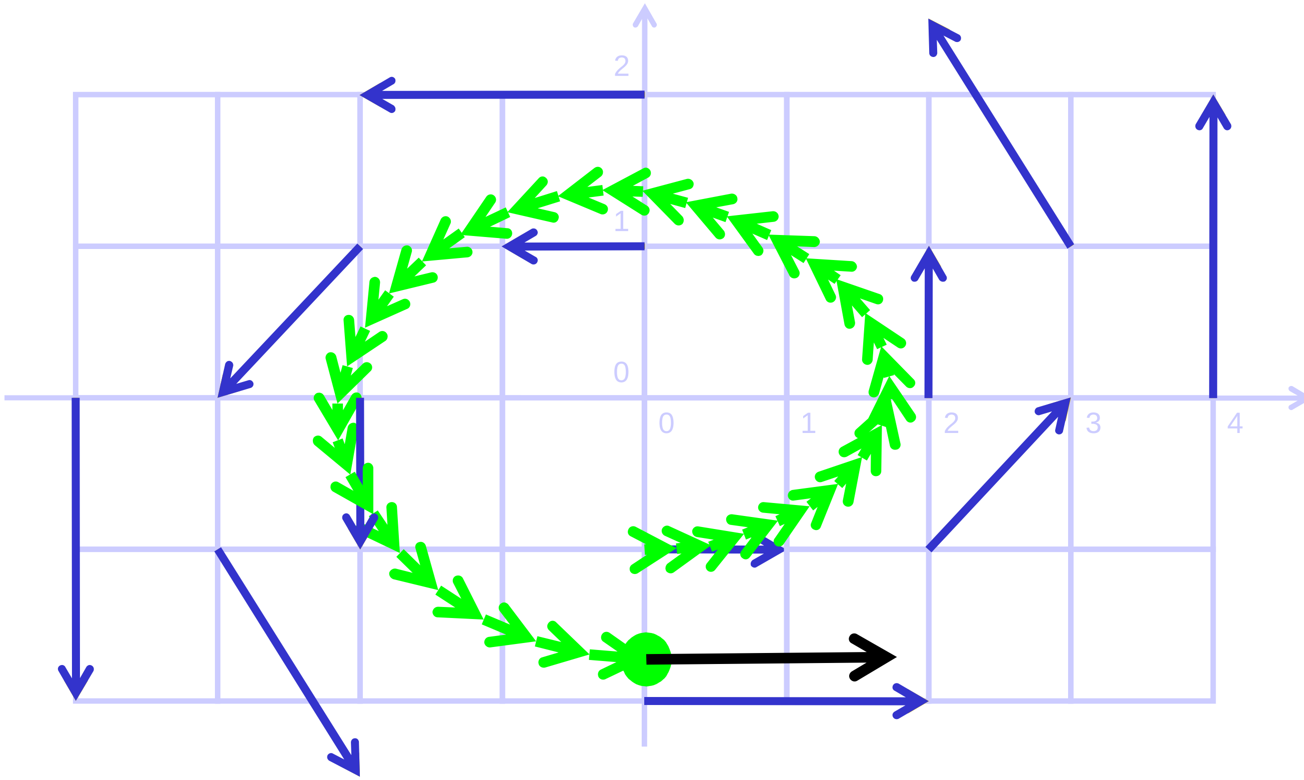


Euler Integration – Example

- d*t* smaller (1/4): more steps, more exact.

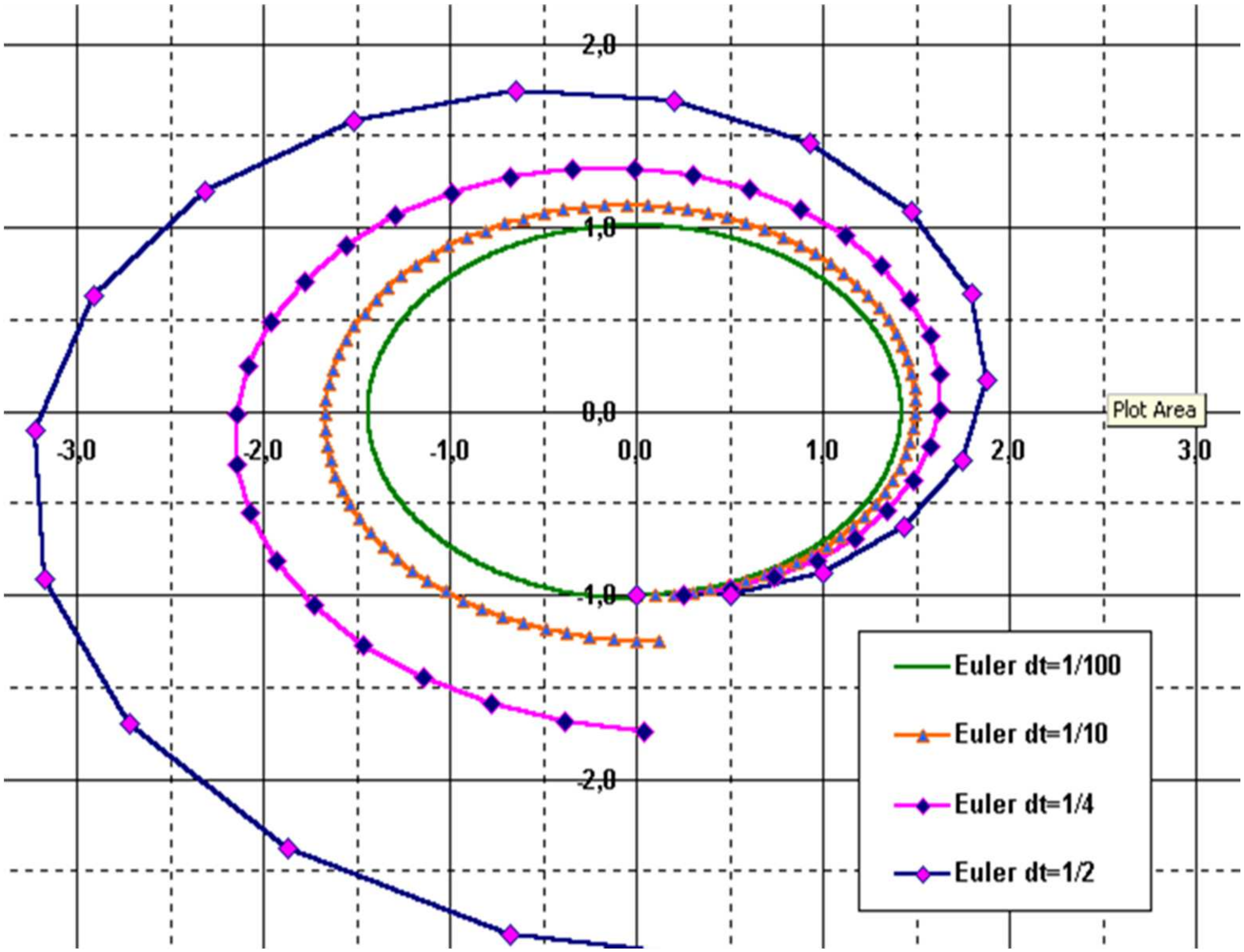
$\mathbf{s}_{36} \approx (0.04 | -1.74)^T$; $\mathbf{v}(\mathbf{s}_{36}) \approx (1.74 | 0.02)^T$;

- 36 steps



Comparison Euler, Step Sizes

Euler
quality is proportional to d*t*

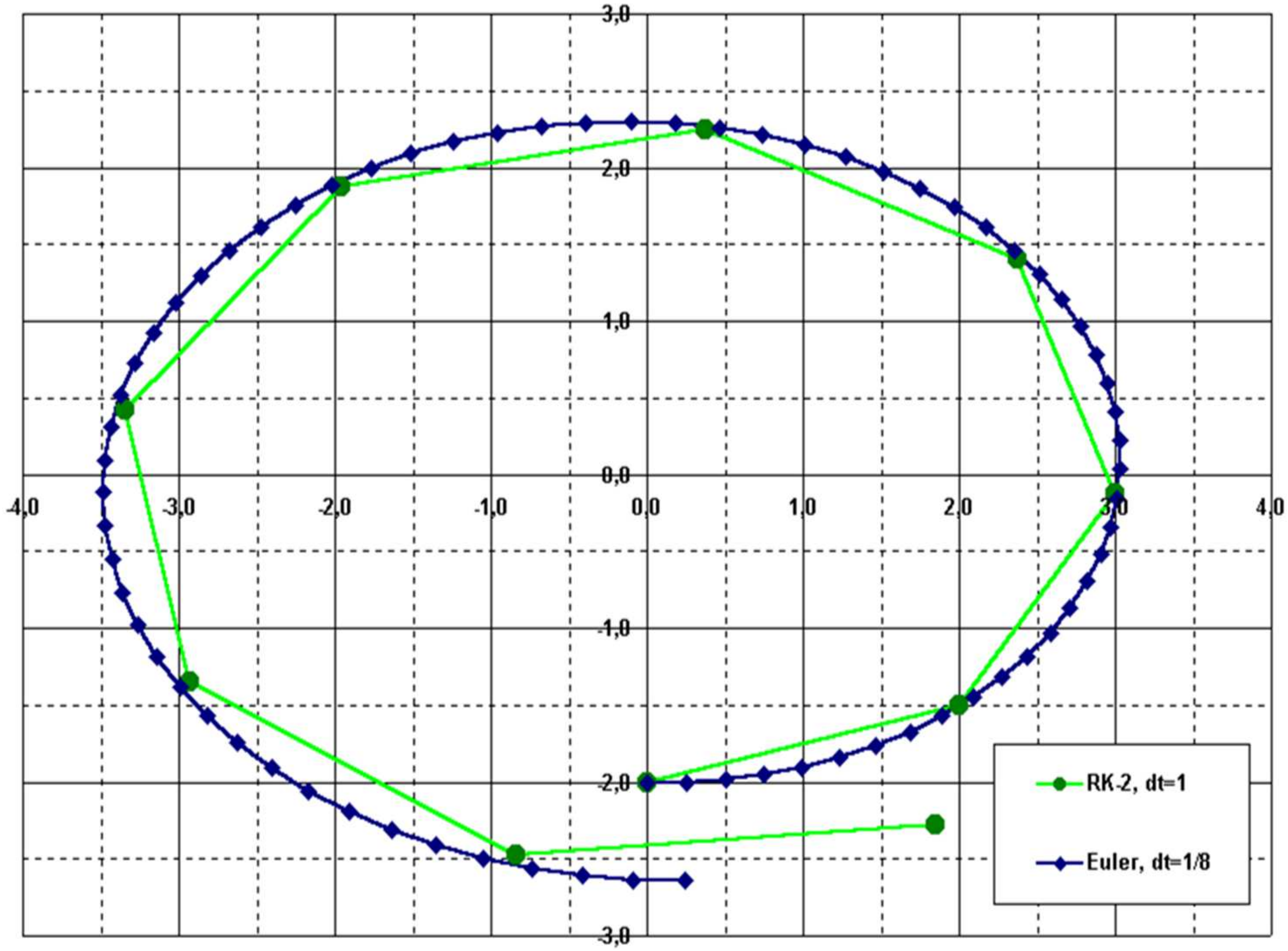


Euler Example – Error Table

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

RK-2 – A Quick Round

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



RK-4 vs. Euler, RK-2

Even better: fourth order RK:

- four vectors **a**, **b**, **c**, **d**
- one step is a convex combination:
  $\mathbf{s}_{i+1} = \mathbf{s}_i + (\mathbf{a} + 2\cdot\mathbf{b} + 2\cdot\mathbf{c} + \mathbf{d})/6$
- vectors:

  $\mathbf{a} = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i)$ … original vector

  $\mathbf{b} = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i+\mathbf{a}/2)$ … RK-2 vector

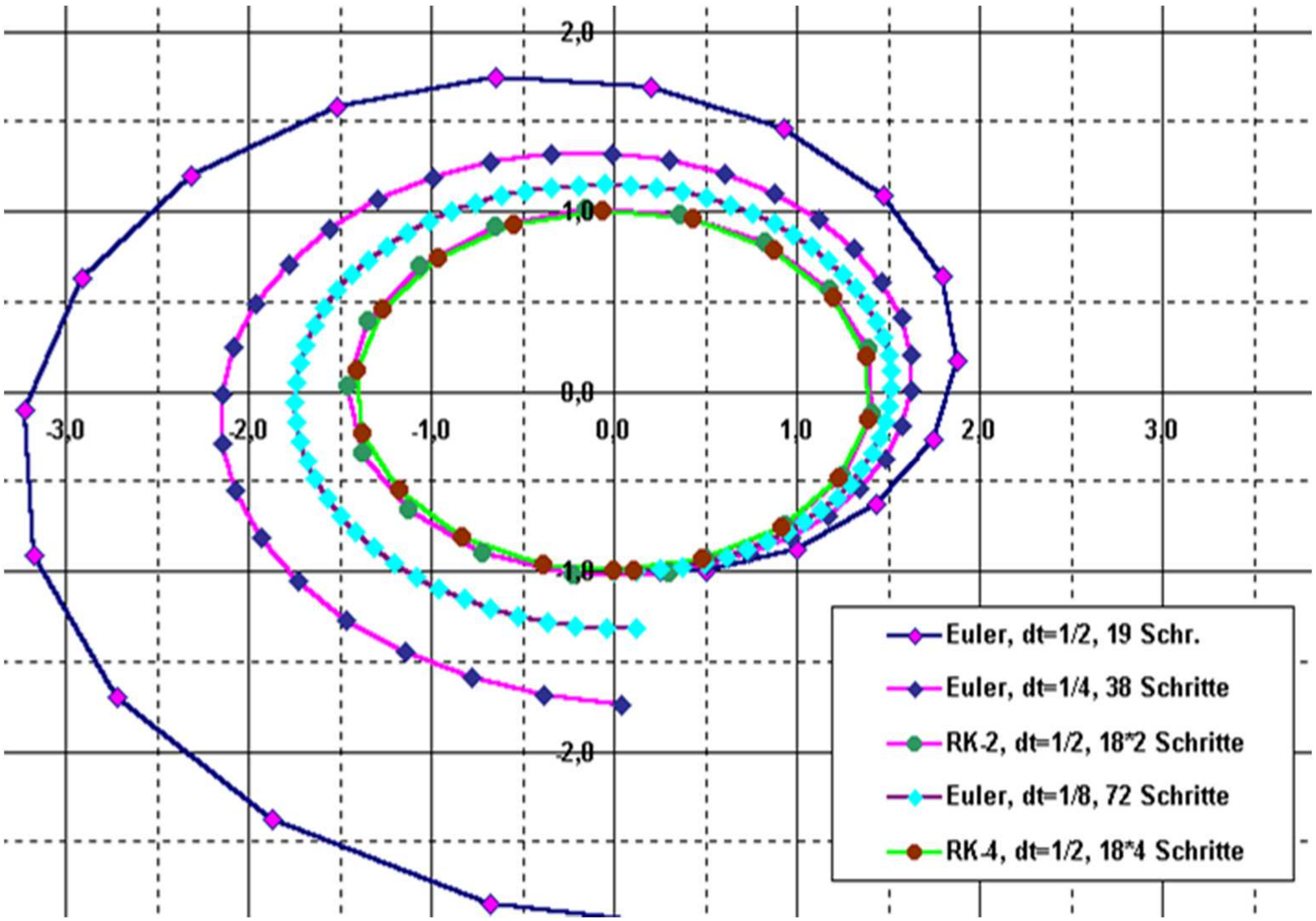
  $\mathbf{c} = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i+\mathbf{b}/2)$ … use RK-2 …

  $\mathbf{d} = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i+\mathbf{c})$ … and again

Euler vs. Runge-Kutta

RK-4: pays off only with complex flows

Here approx. like RK-2



Integration, Conclusions

Summary:

- analytic determination of streamlines usually not possible
- hence: numerical integration
- various 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.

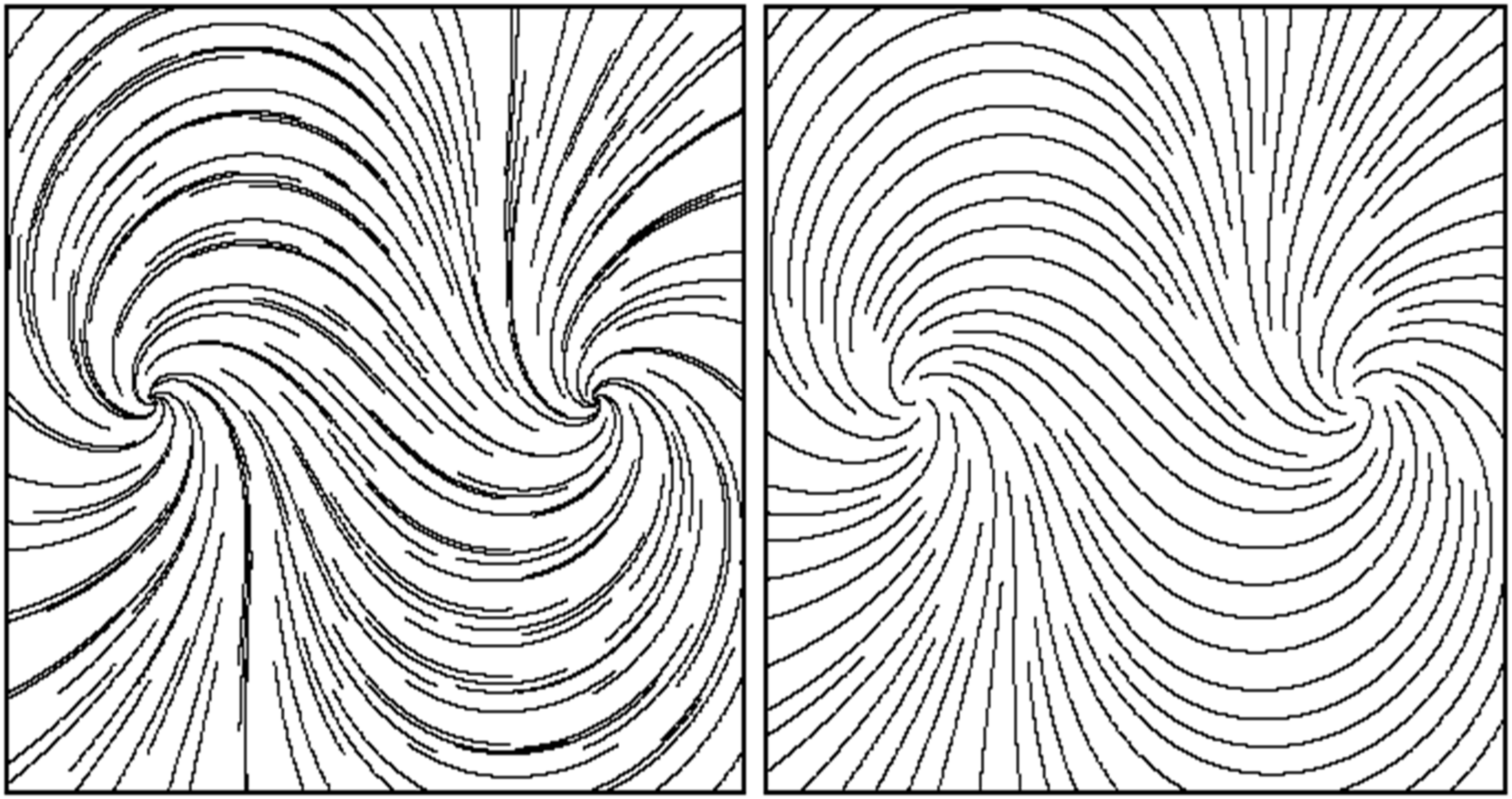
Streamline Placement

in 2D

Problem: Choice of Seed Points

Streamline placement:

- If regular grid used: very irregular result



Overview of Algorithm

Idea: streamlines should not lie too close to one another

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 – Pseudo-Code

- Compute initial streamline, put it into a queue
- current streamline = initial 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

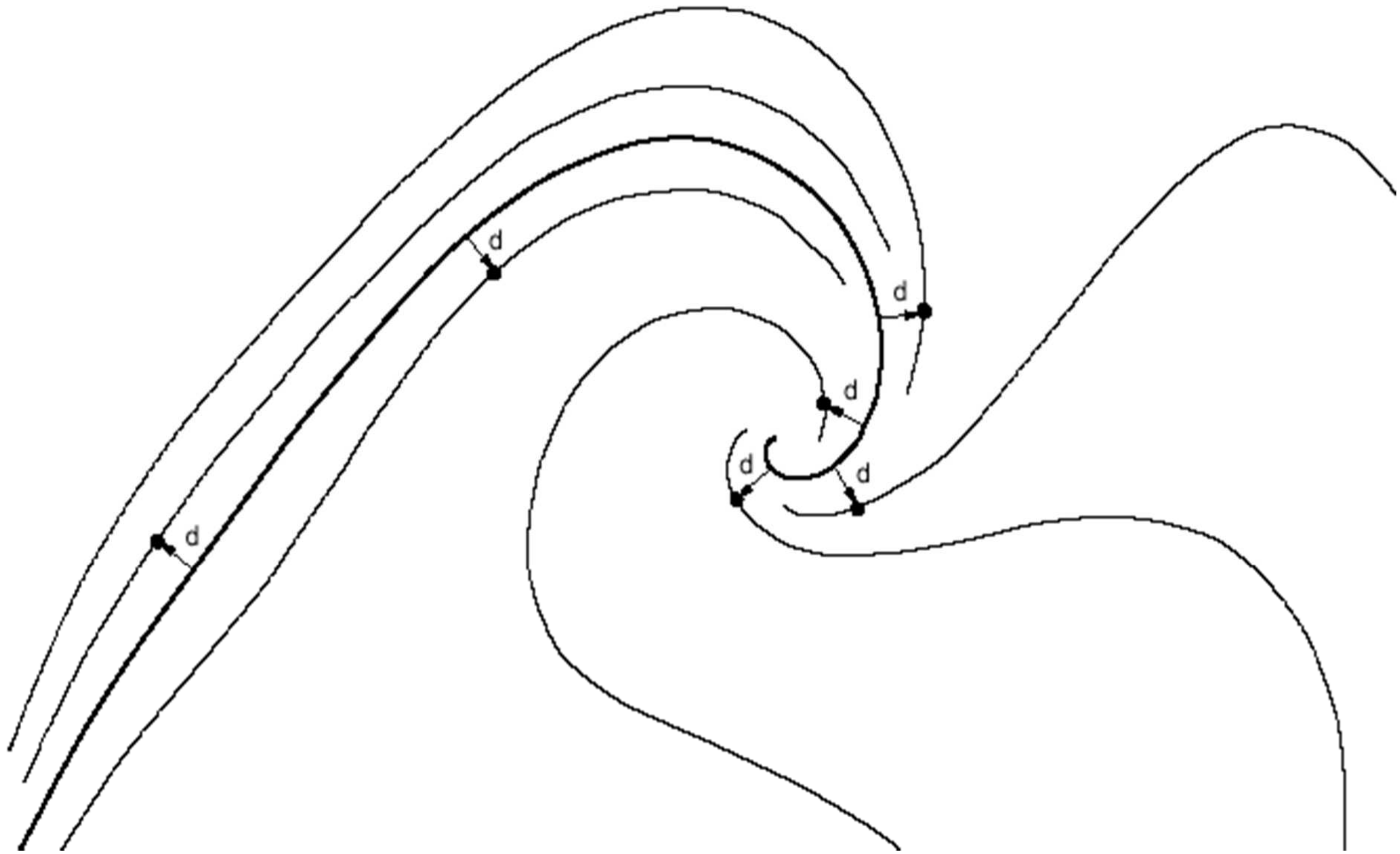
  ELSE next streamline in queue becomes current streamline

Streamline Termination

When to stop streamline integration:

- when distance to neighboring streamline ≤ d_{test}
- when streamline leaves flow domain
- when streamline runs into fixed point ($\mathbf{v} = 0$)
- when streamline gets too near to itself (loop)
- after a certain amount of maximal steps

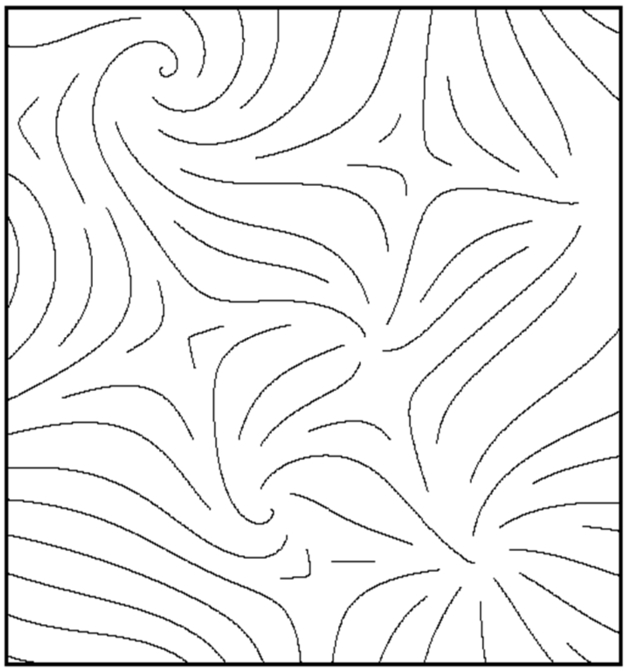
New Streamlines



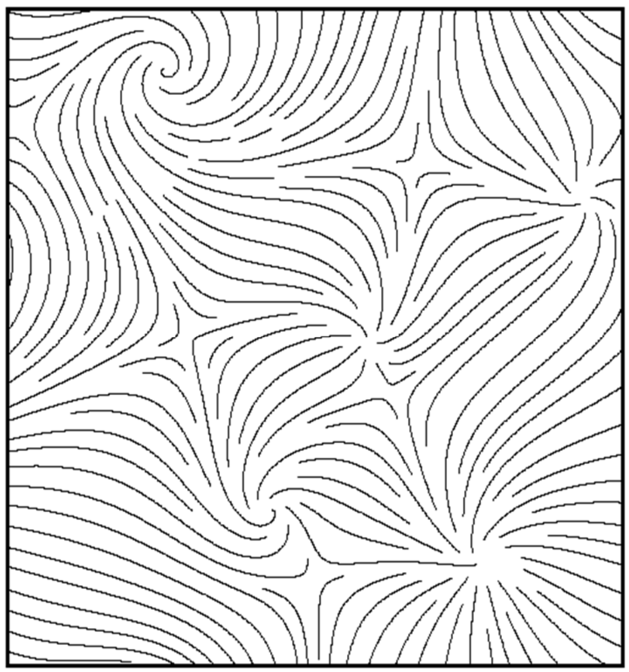
Different Streamline Densities

Variations of d_{sep} relative to image width:

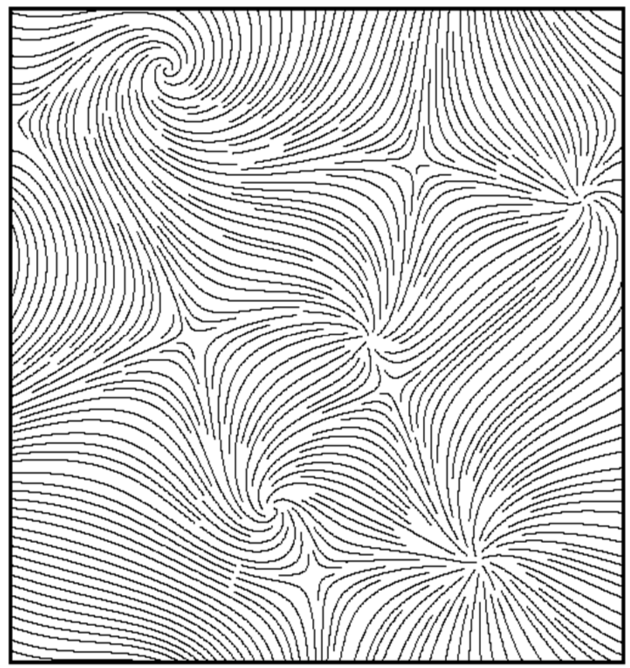
6%



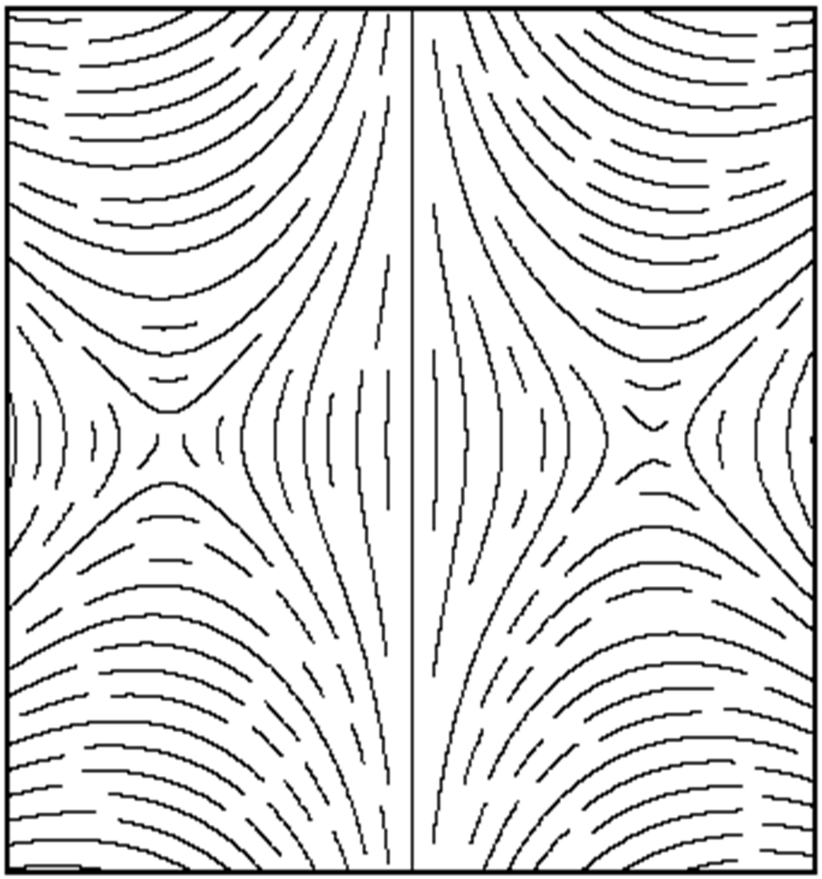
3%

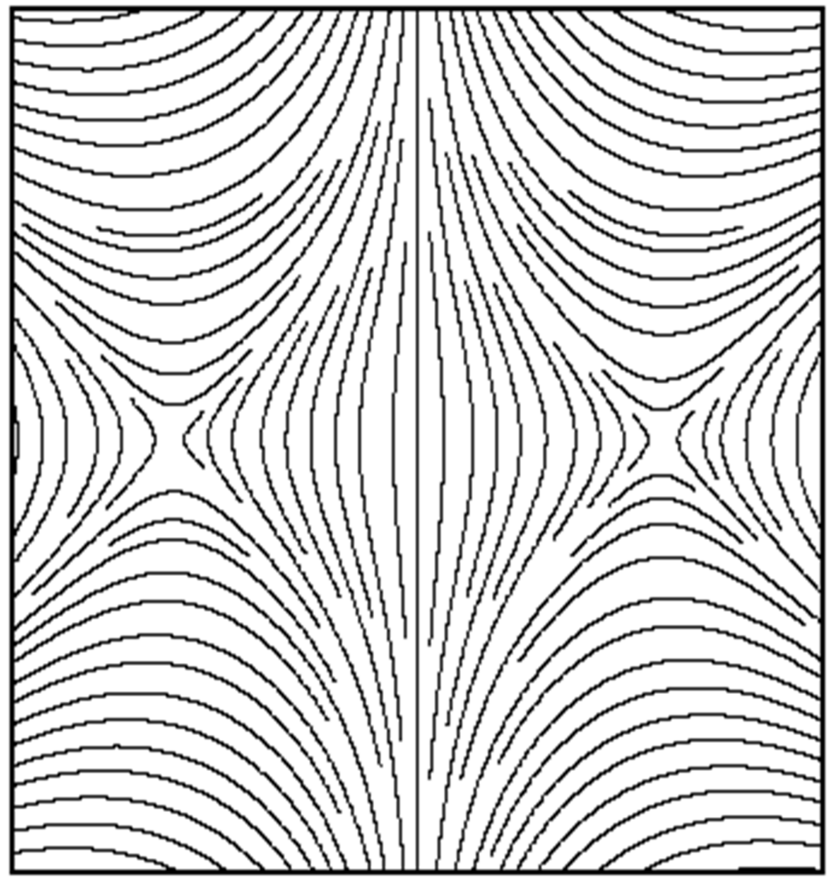


1.5%



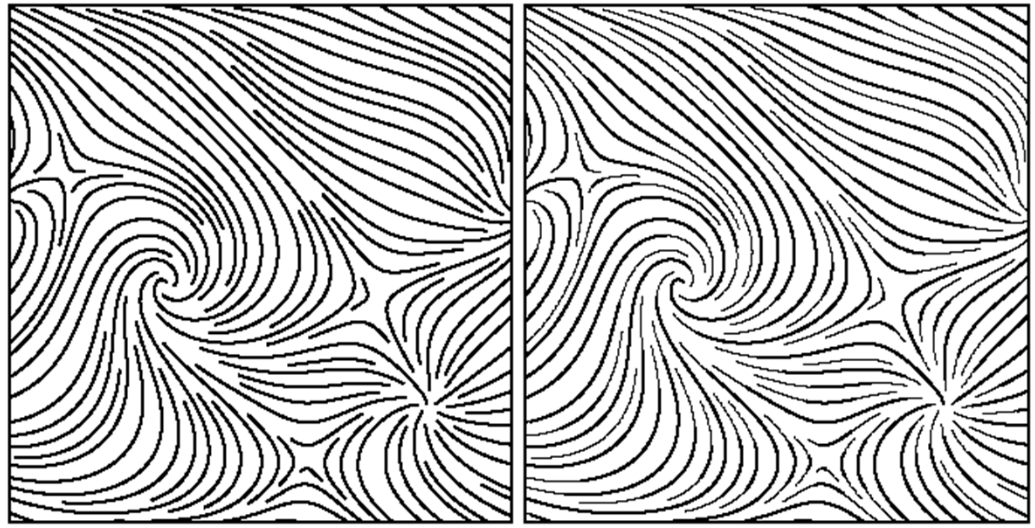
d_{sep} vs. d_{test}



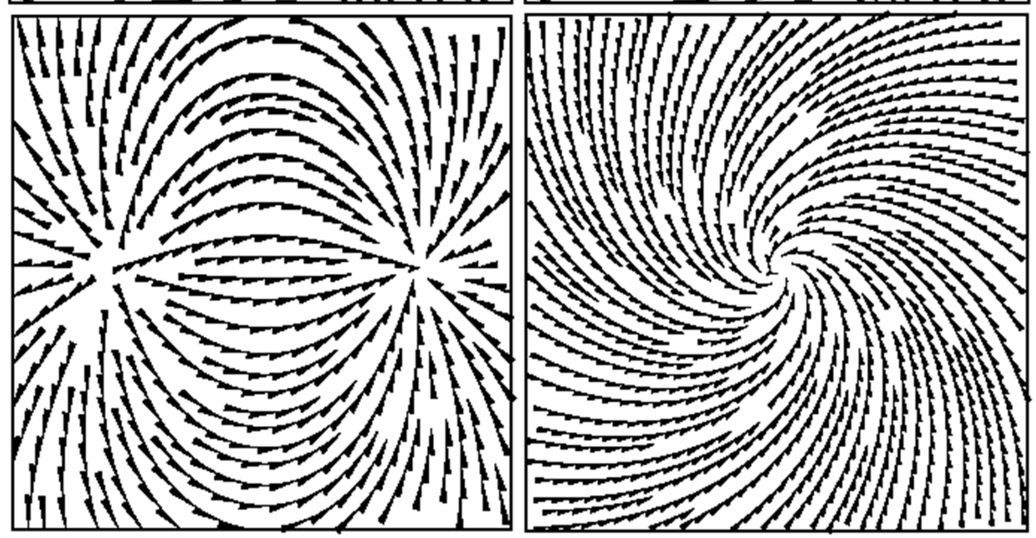


Tapering and Glyphs

Thickness in relation to distance



Directional glyphs:



Literature

For more information, please see:

- 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
- **Data Visualization: Principles and Practice, Chapter 6: Vector Visualization** by A. Telea, AK Peters 2008

FlowVis Videos available on Bob's web page.

Acknowledgment

Thanks for the materials

- Prof. Robert S. Laramee, Swansea University, UK