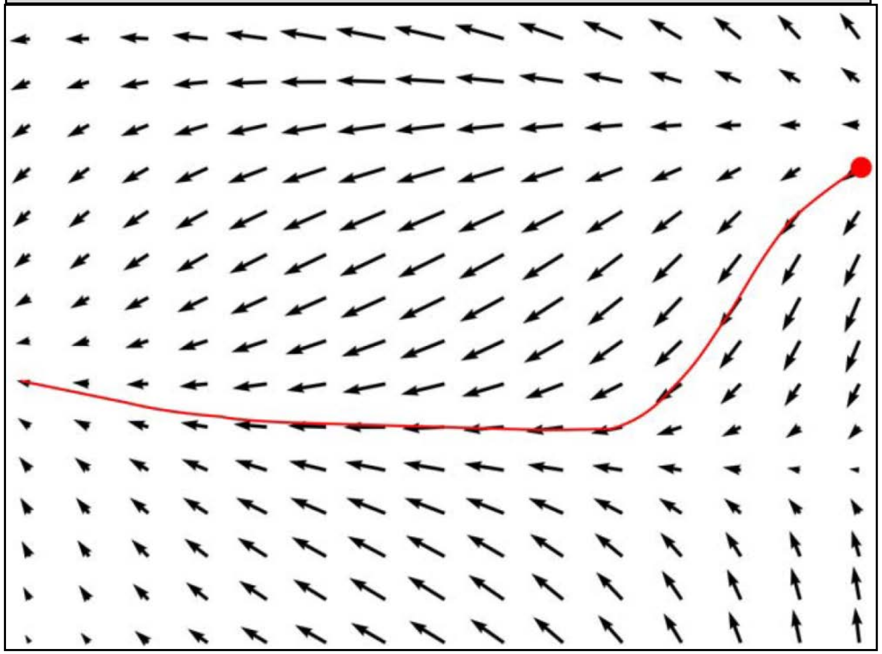
# Vector Field Visualization: Introduction

Goal: understand what is a vector field and where it is from, why visualizing vector fields is challenging, what are the typical visualization techniques for vector field data, what is the direct visualization of vector field, how to compute streamlines

# What is a Vector Field?

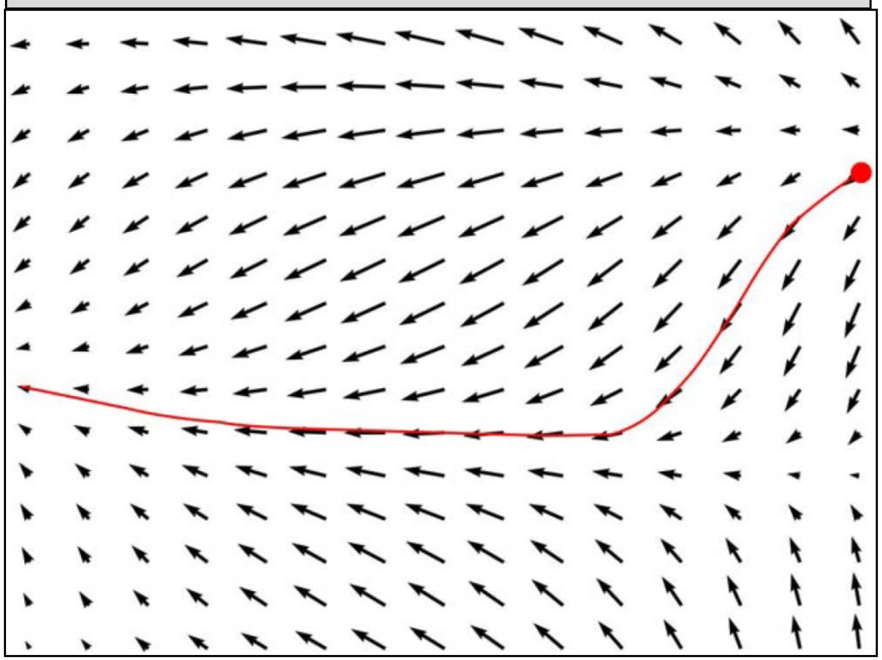
A simple 2D **steady** vector field



**A vector-valued function** that assigns a vector (with direction and magnitude) to any given point.

# What is a Vector Field?

A simple 2D **steady** vector field



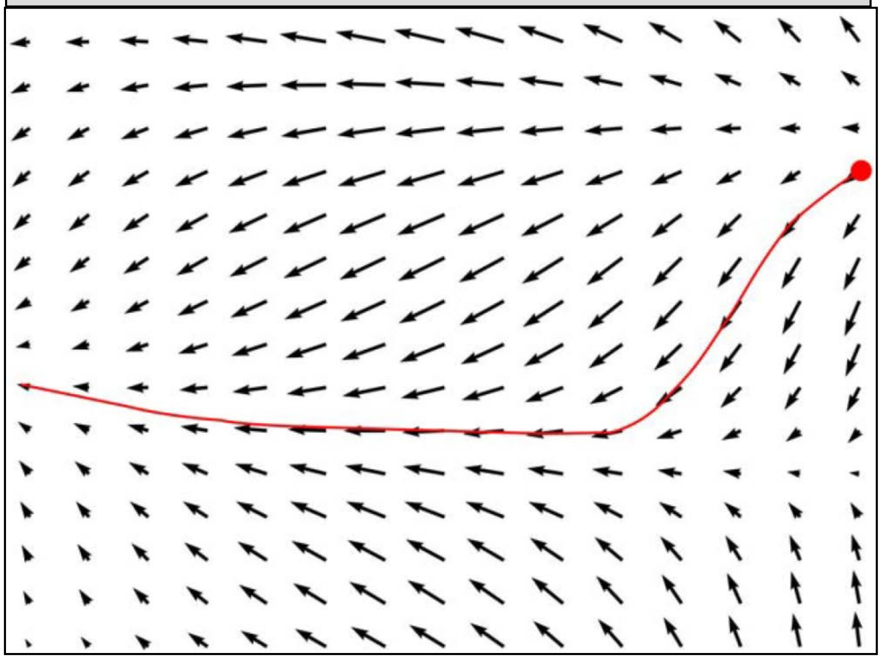
**A vector-valued function** that assigns a vector (with direction and magnitude) to any given point.

It typically can be expressed as an ordinary differential equation (ODE).

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

# What is a Vector Field?

A simple 2D **steady** vector field

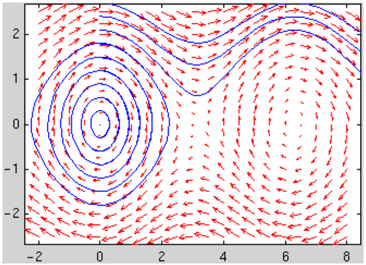


**A vector-valued function** that assigns a vector (with direction and magnitude) to any given point.

It typically can be expressed as an ordinary differential equation (ODE).

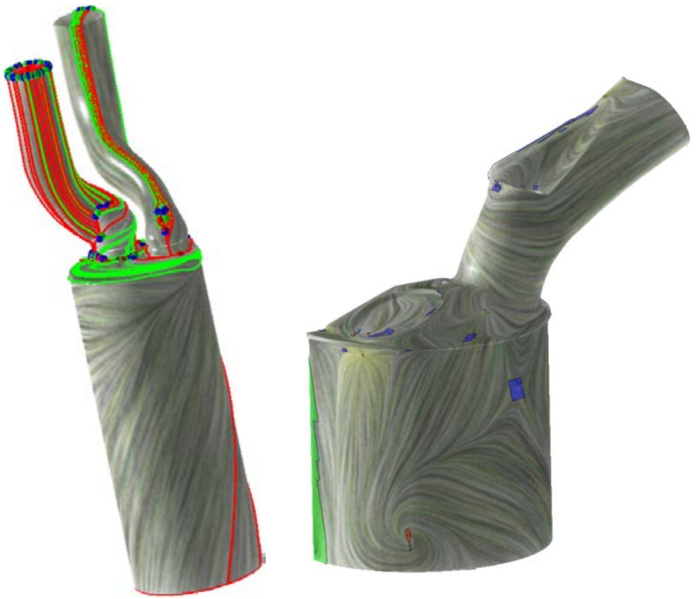
$$\frac{d\varphi(x)}{dt} = V(x)$$

Its solution gives rise to a "**flow**", which consists of densely placed particle **trajectories** (i.e., the red curve shown in the left example).

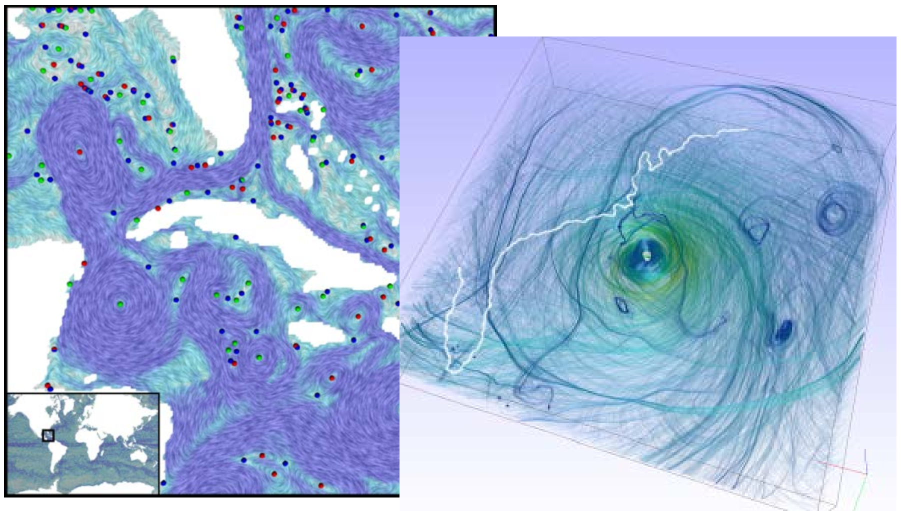


# Why Is It Important?

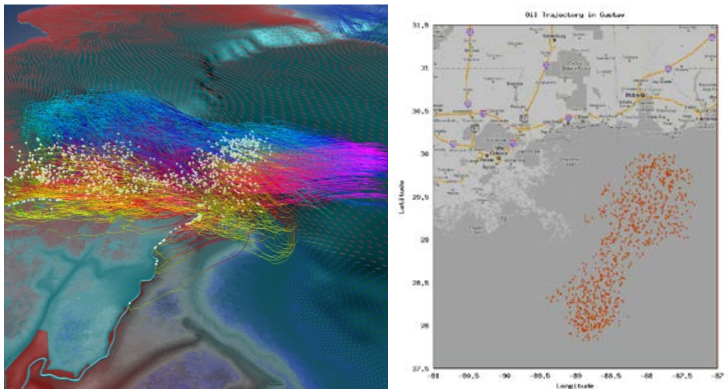
# Applications in Engineering and Science



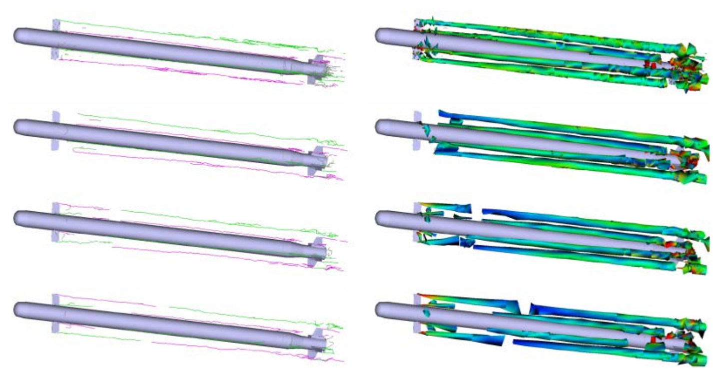
**Automotive design**
[**Chen** et al. TVCG07,TVCG08]



**Weather study** [Bhatia and Chen et al. TVCG11]

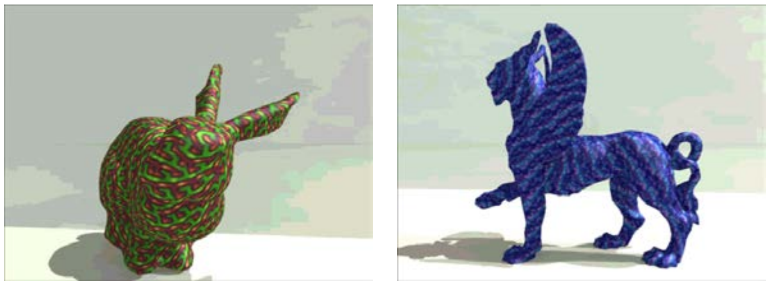


**Oil spill trajectories** [Tao et al. EMI2010]

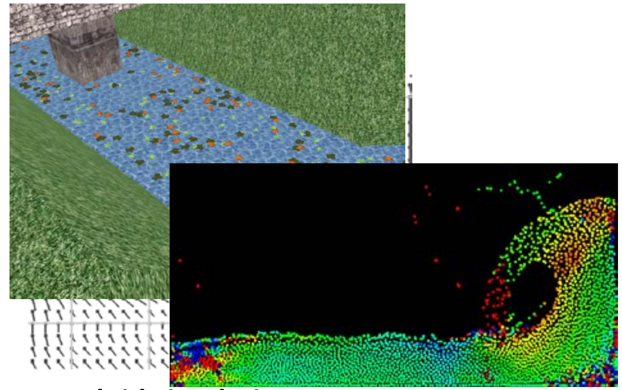


**Aerodynamics around missiles** [Kelly et al. Vis06]

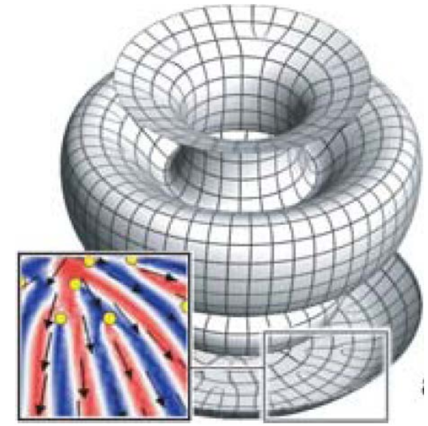
# Applications in Computer Graphics



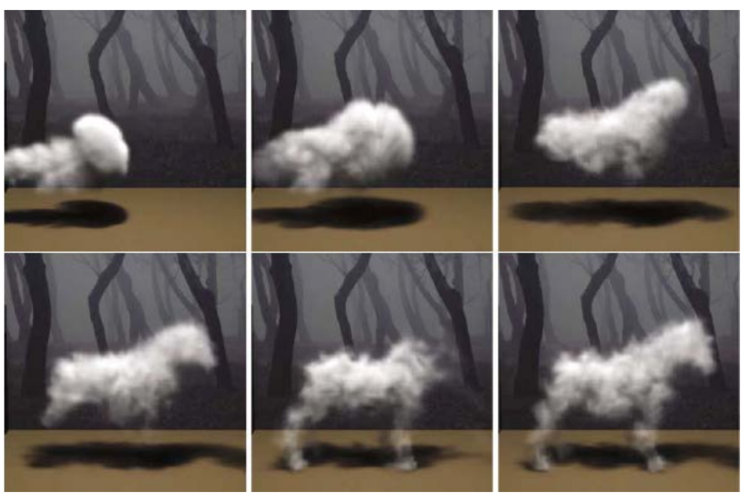
**Texture Synthesis** [**Chen** et al. TVCG12b]



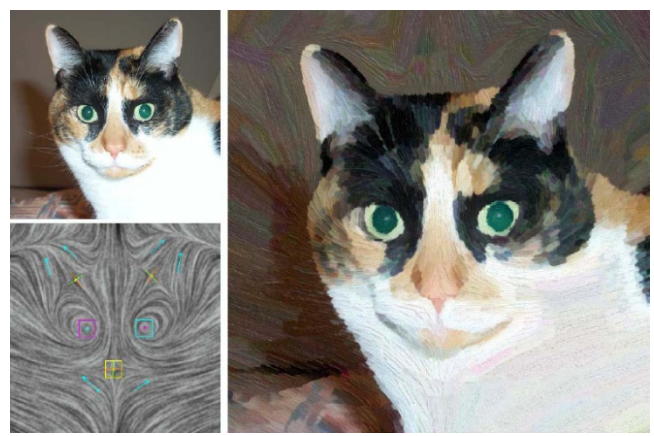
**Fluid simulation** [Chenney SCA2004, Cao&Chen 2013]



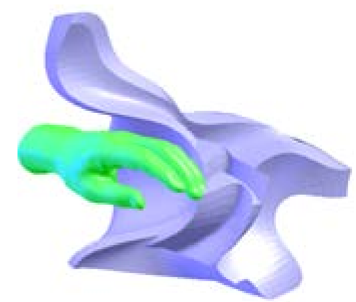
**Parameterization** [Ray et al. TOG2006]



**Smoke simulation** [Shi and Yu TOG2005]



**Painterly Rendering** [Zhang et al. TOG2006]

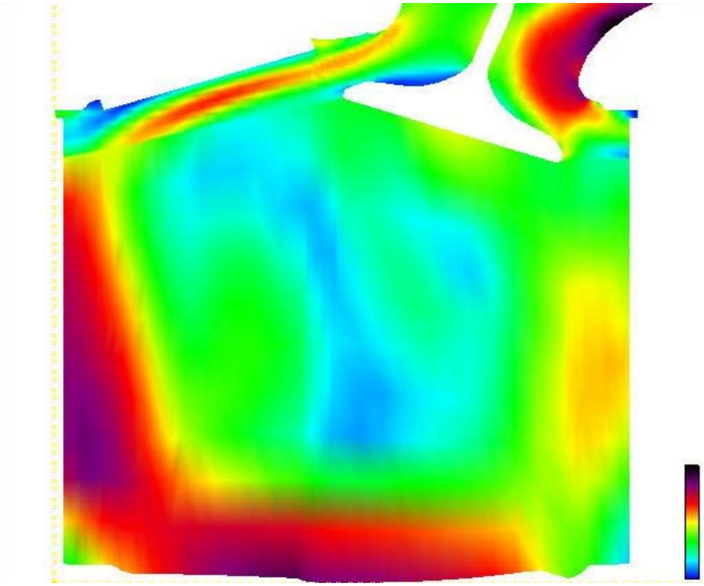


**Shape Deformation** [von Funck et al. 2006]

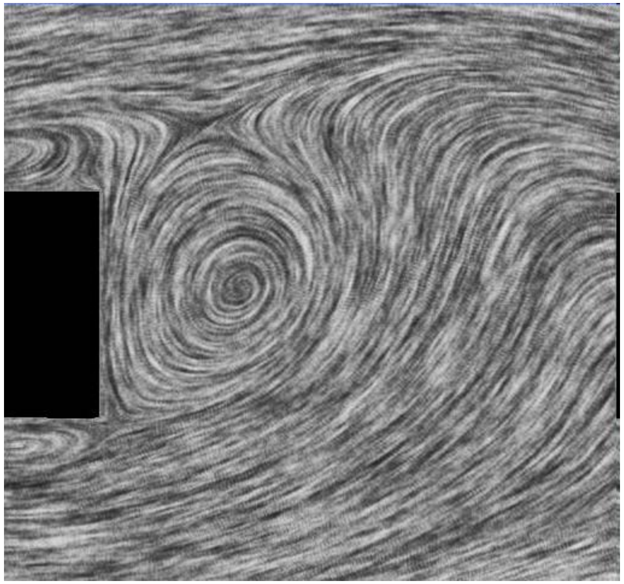
# Why Is It Challenging to Process?

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

*magnitude only*

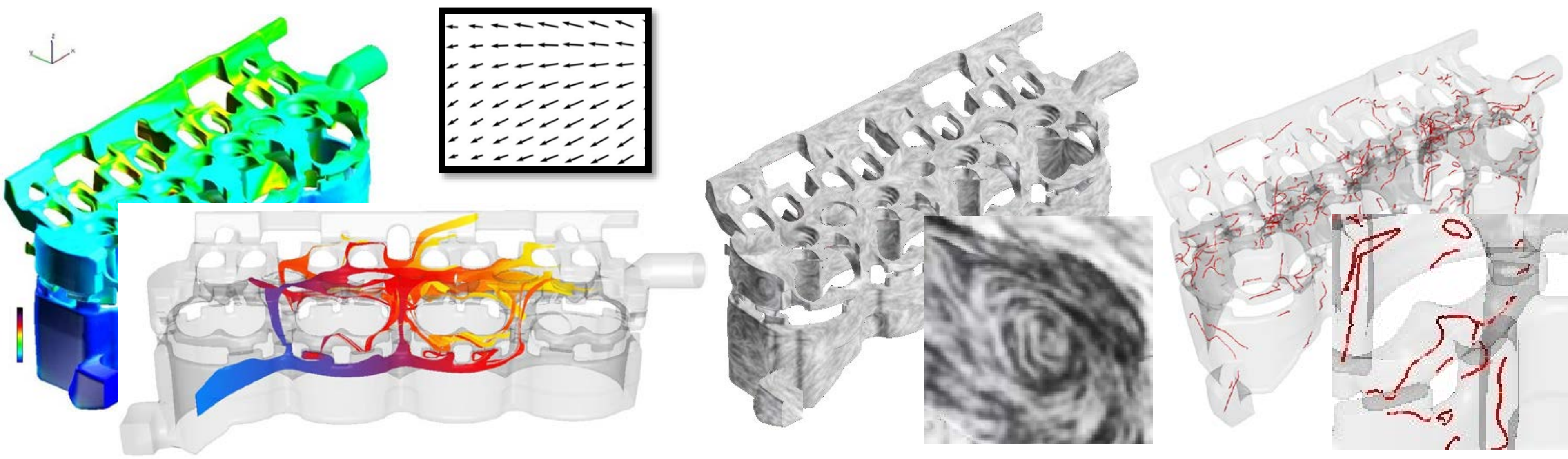


*direction only*



# Classification of Visualization Techniques

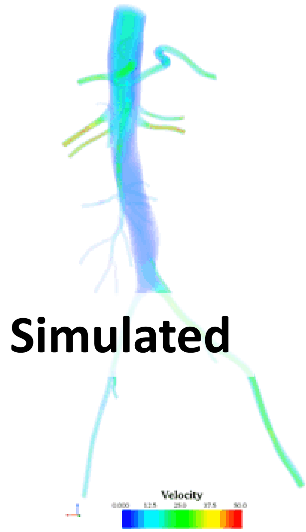
- **Direct method:** overview of vector fields, minimal computation, e.g., glyphs, color mapping.
- **Geometric:** a discrete object(s) whose geometry reflects (e.g., tangent to) flow characteristics, e.g., integral curves.
- **Texture-based:** covers domain with a <u>convolved</u> texture, e.g., Spot Noise, LIC, LEA, ISA, IBFV(S).
- **Feature-based:** both automatic and interactive feature-based techniques, e.g., flow topology, vortex core structure, coherent structure, LCS, etc.



# Flow Data

Data sources:

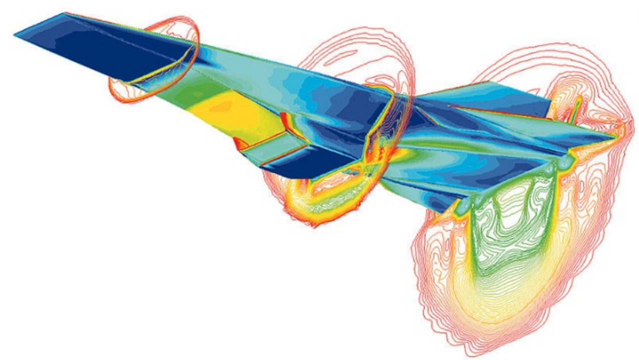
- flow simulation:
  - airplane- / ship- / car-design
  - weather simulation (air-, sea-flows)
  - medicine (blood flows, etc.)

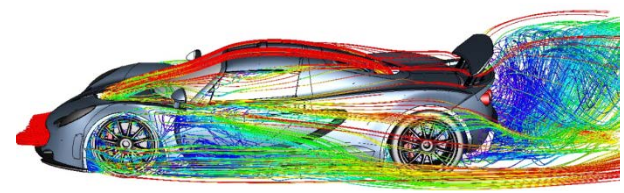



Source: simtk.org

# 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 terabytes
- Unsteady, i.e.. time-dependent
- unstructured, adaptive resolution grids

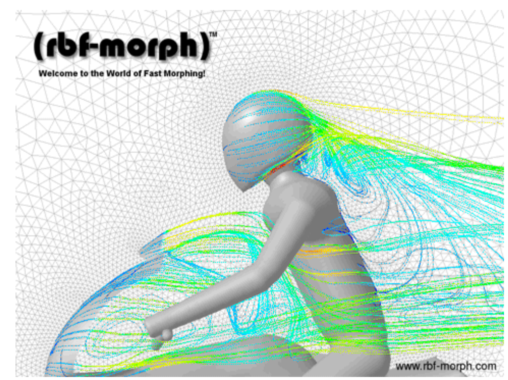
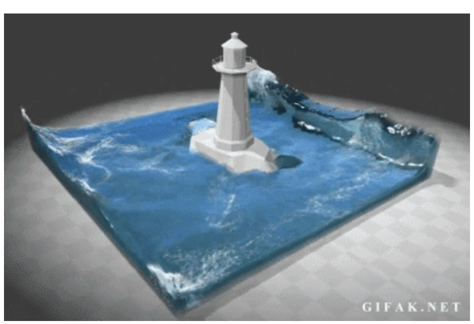
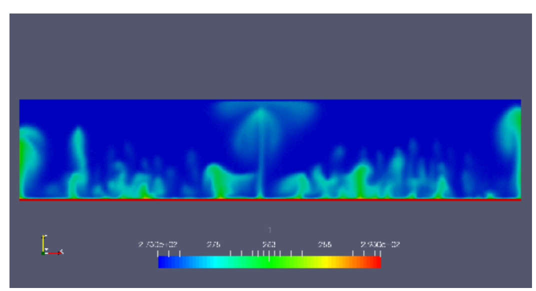



Image source: Google images

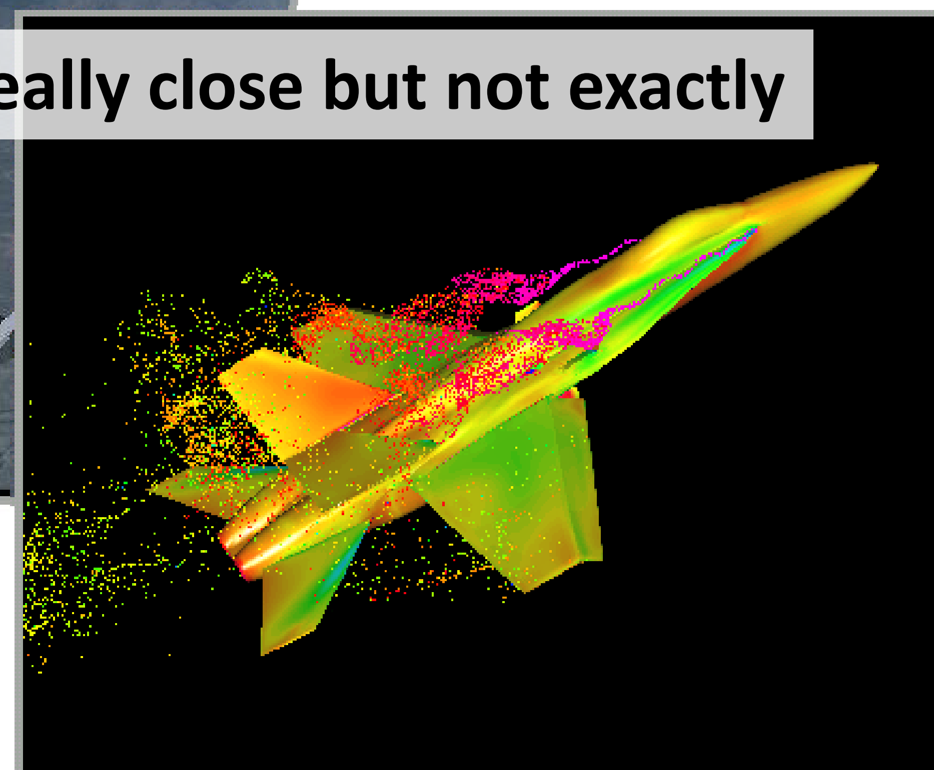


# Comparison with Reality


Experiment

**Really close but not exactly**



Simulation

# 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




Source: simtk.org


Source: speedhunter.com

# 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)

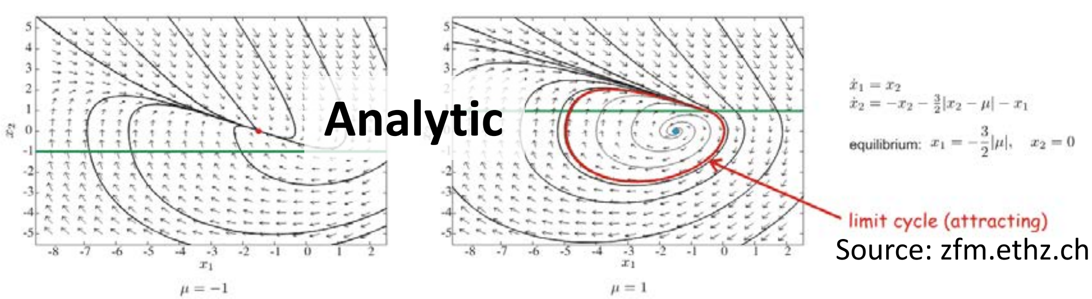



Source: simtk.org




Source: speedhunter.com






Source: zfm.ethz.ch

# Flow Data

**Simulation:**

- flow: estimate (partial) differential equation systems (e.g., a physical model)
- set of samples (3/4-dims. of data), e.g., given on a curvilinear grid
- most important primitive: tetrahedron and hexahedron (cell)
- could be adaptive grids

**Measurement:**

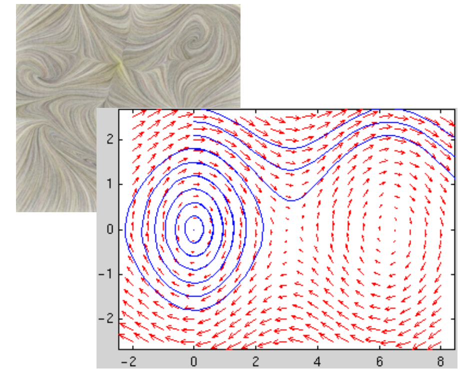
- vectors: taken from instruments, often estimated on a uniform grid
- optical methods + image recognition, e.g.,: PIV (particle image velocimetry)

**Analytic:**

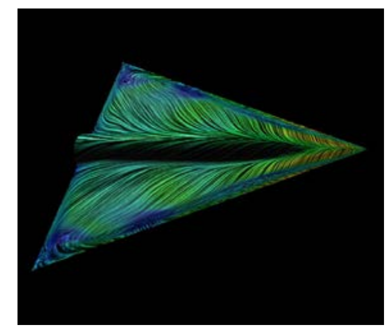
- flow: analytic formula, differential equation systems d**x**/dt (dynamical system)
- evaluated where-ever needed (e.g., making plots of flow in MatLab)

# Types of the vector field data

# 2D vs. 2.5D Surfaces vs. 3D

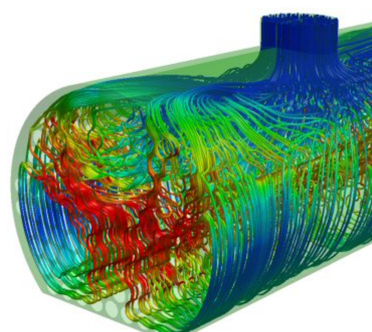


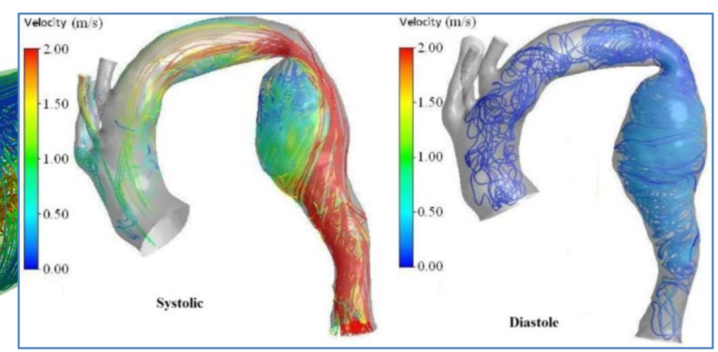

## 2D flow visualization
- $R^2$ flows
- Planes, or flow layers (2D cross sections through 3D)



## 2.5D, i.e. surface flow visualization
- 3D flows around obstacles
- boundary flows on manifold surfaces (locally 2D)

## 3D flow visualization
- $R^3$ flows
- simulations, 3D domains

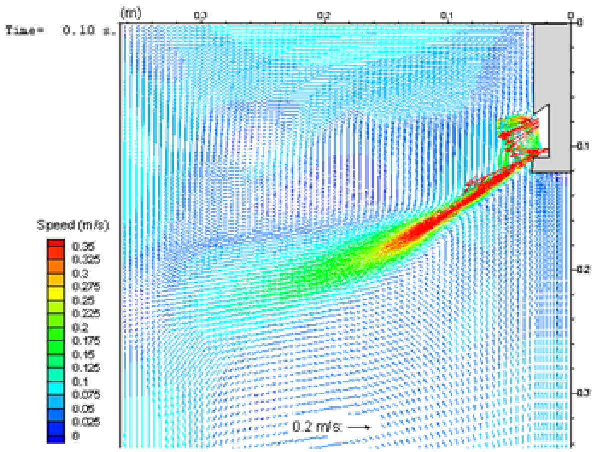




# Steady vs. Time-dependent

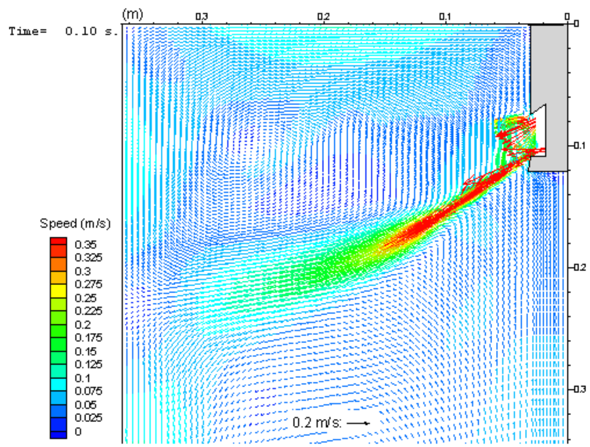
Steady (time-independent) flows:

- flow itself constant over time
- **v**(**x**), e.g., laminar flows
- simpler case for visualization
- well understood behaviors and features

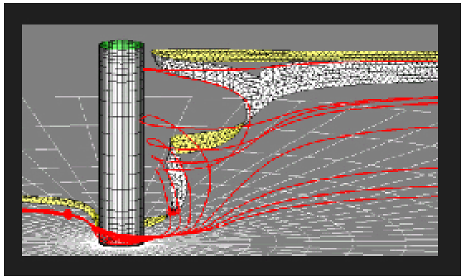



Time-dependent (unsteady) flows:

- flow itself changes over time
- **v**(**x**,***t***), e.g., combustion flow, turbulent flow, wind field
- more complex cases
- *no unified theory to characterize them yet!*

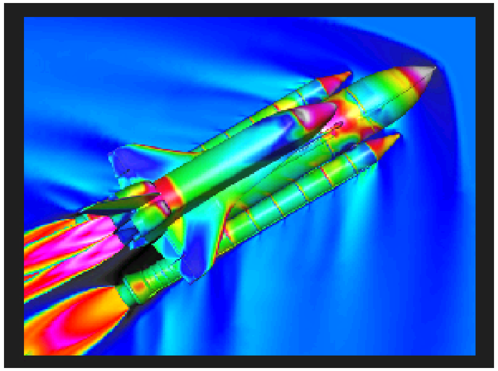


# Time-independent (steady) Data



Single Zone
100K Nodes
4 MB

(1985)

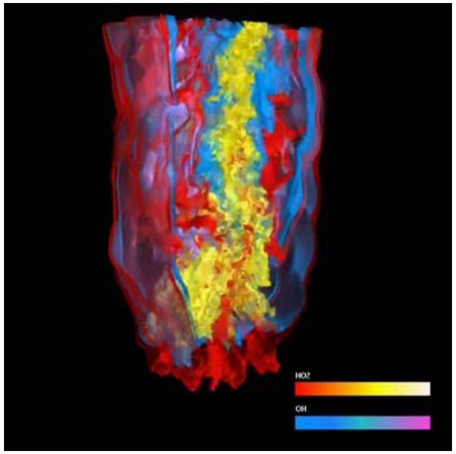


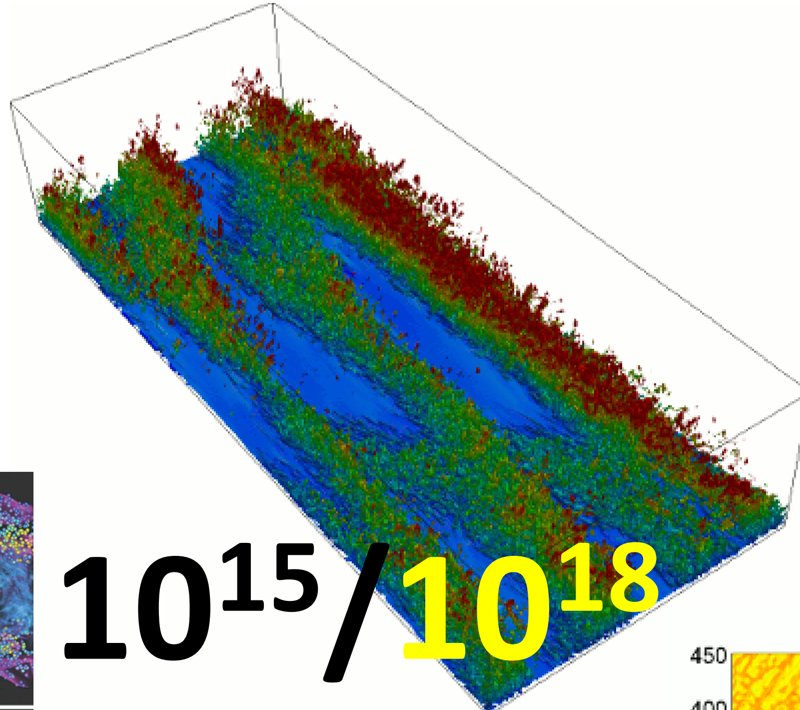
128 Zones
30M Nodes
1080 MB

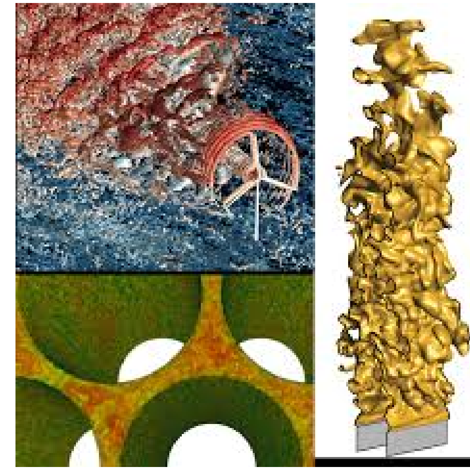
(1996)

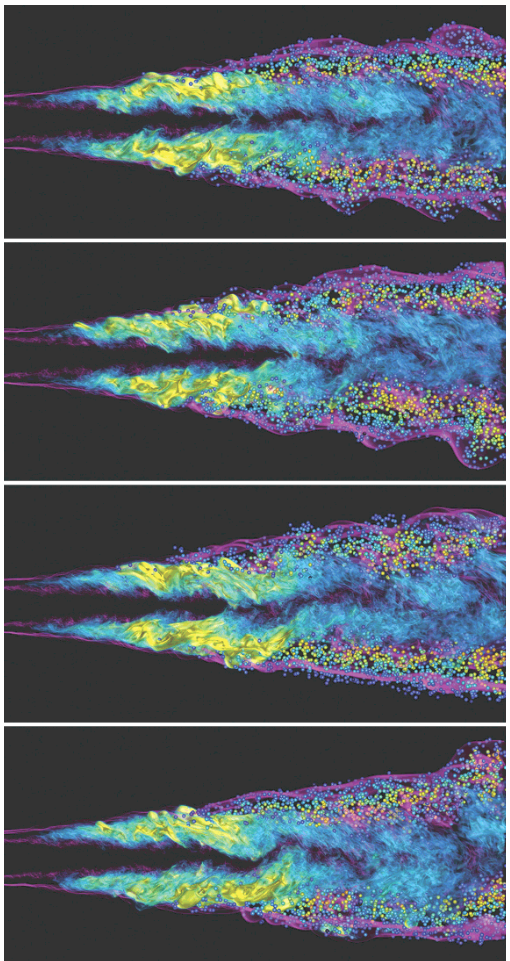
- Dataset sizes over years (old data):

| 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 |



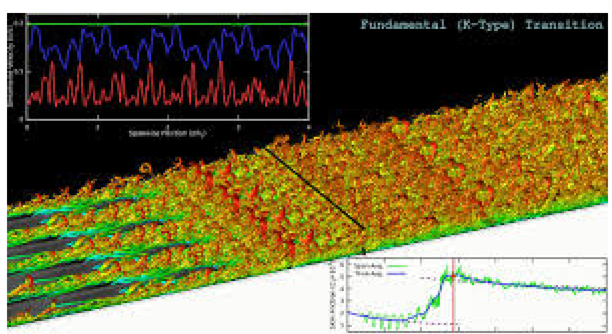


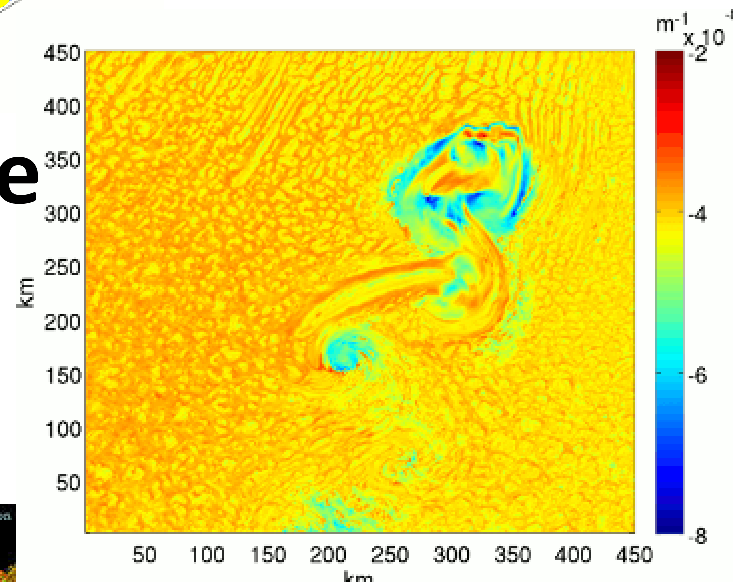




# $10^{15}$/$10^{18}$

# Peta/exa-scale turbulence simulations




# Time-dependent (unsteady) Data

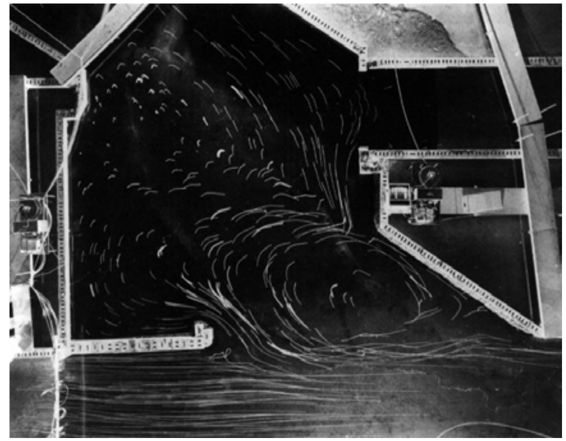
# Experimental Flow Visualization

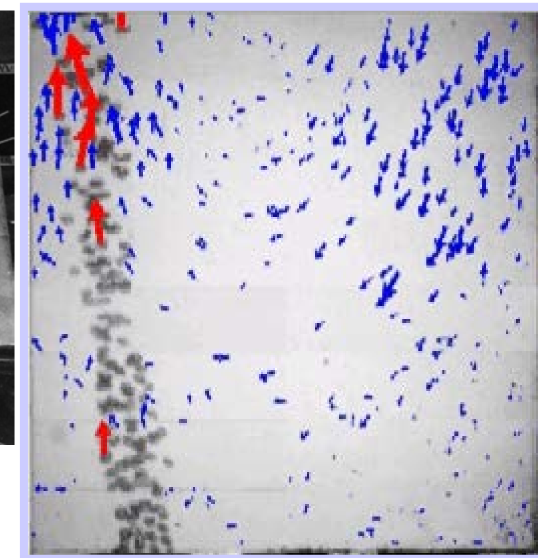
Typically, optical Methods.

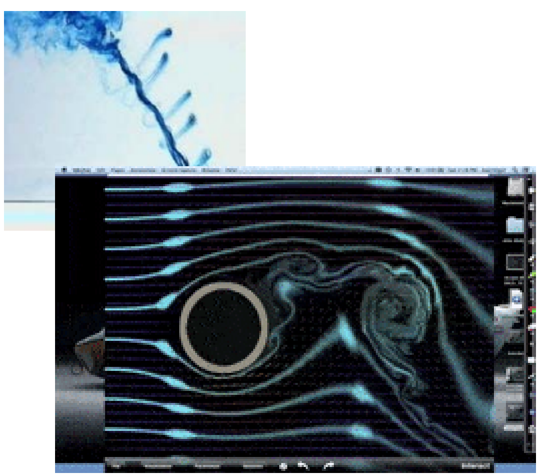
Understanding this experimental methods will help us understand why certain visualization approaches are adopted.

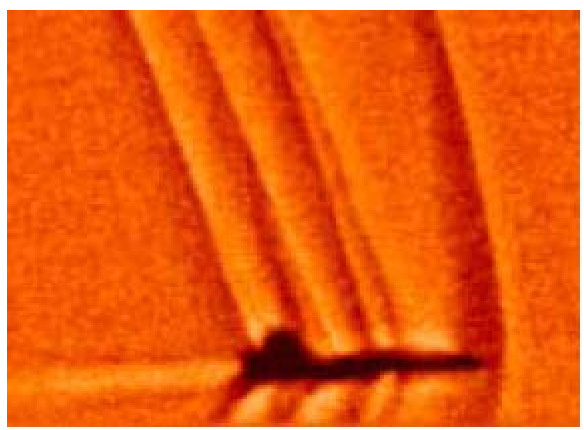
# With Smoke or Dye

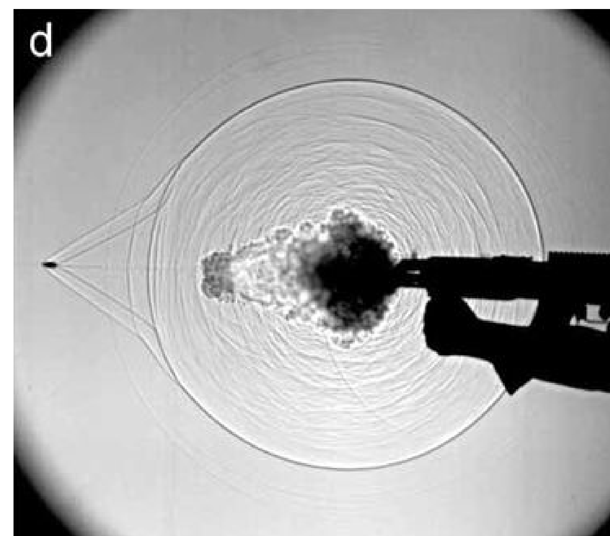
- Injection of dye, smoke, particles
- Optical methods:
  - transparent object with complex distribution of light refraction index
- Streaks, shadows



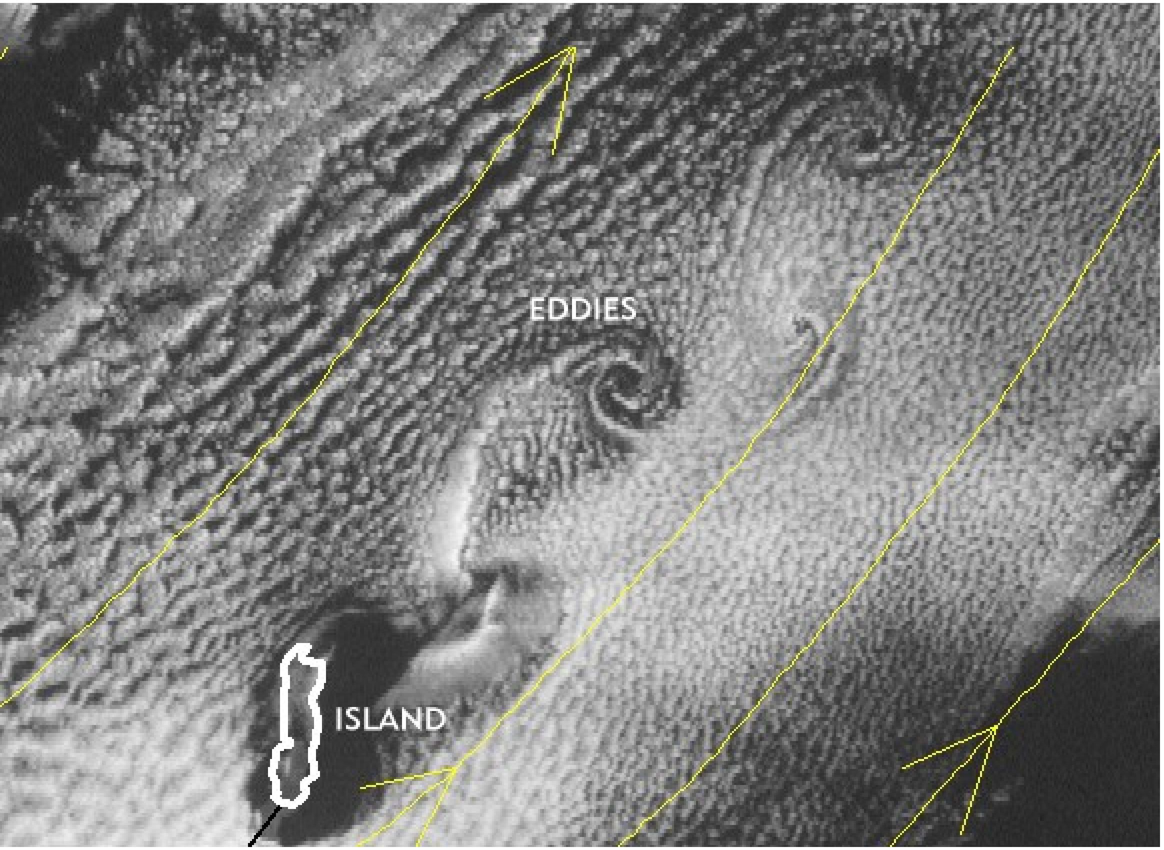








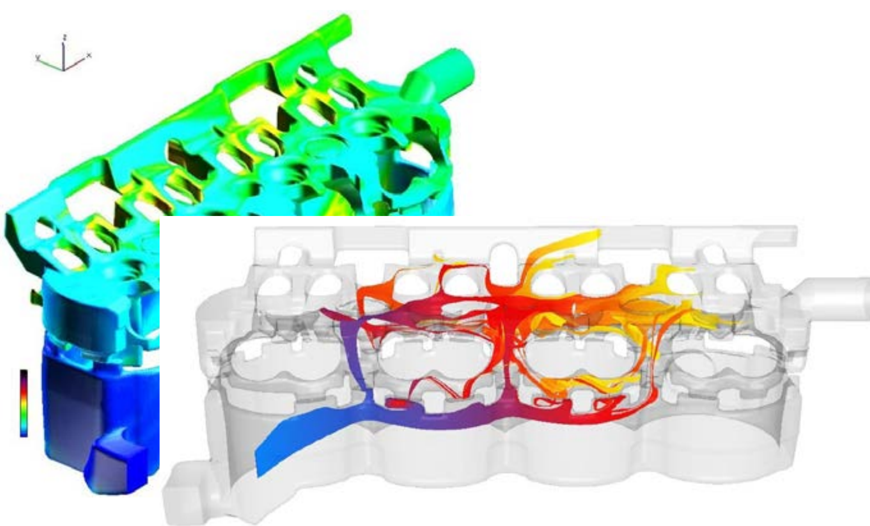
# Large Scale Dying

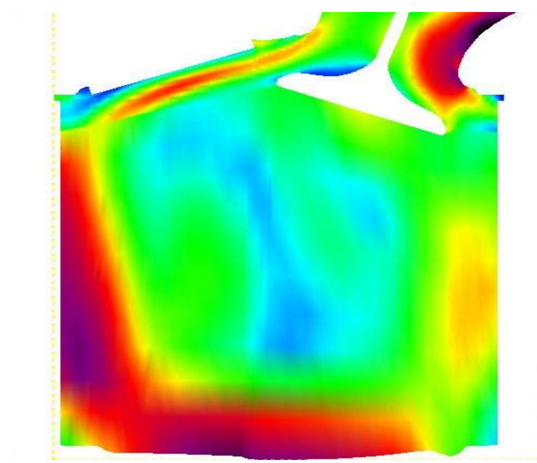



Source: weathergraphics.com

Source: ishtarsgate.com

# Direct Methods

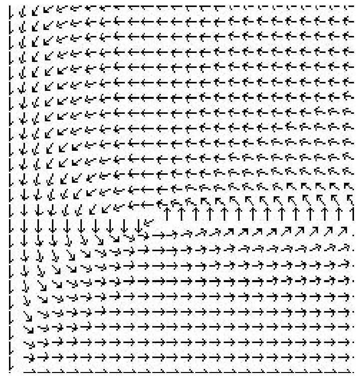


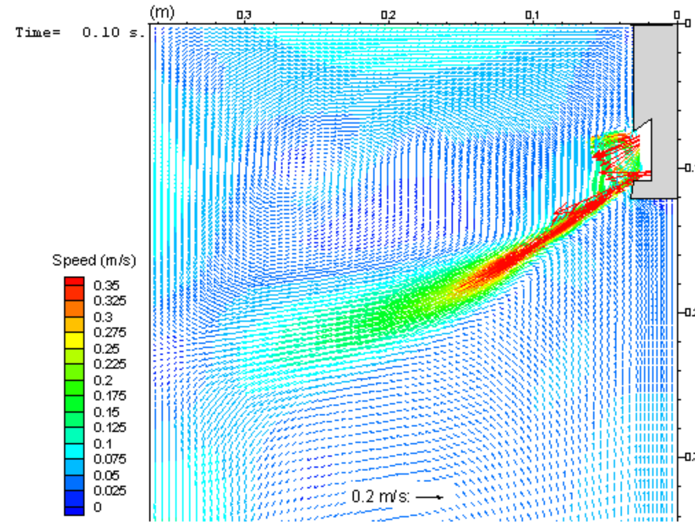


# Direct FlowVis with Arrows

**Properties:**

- direct FlowVis
- <u>frequently used!</u>
- *normalized* arrows vs. velocity coding
- **2D: quite useful, 3D: often problematic**
- often difficult to understand in complex cases, mentally integrate to reconstruct the flow





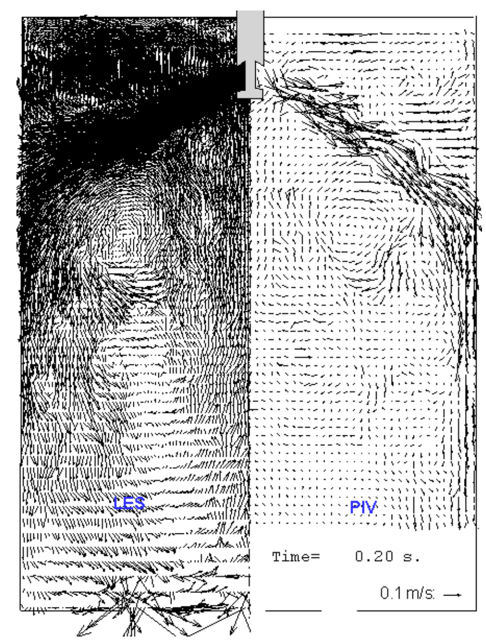




Image source: tms.org

# Issues of Arrows in 3D

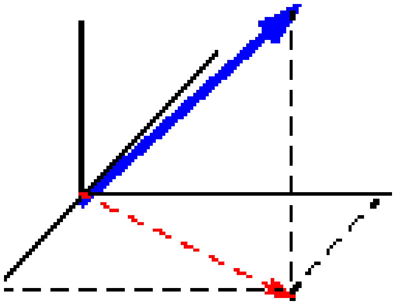
**Common problems:**

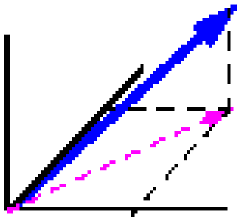
- Ambiguity
- Perspective shortening
- 1D objects generally difficult to grasp in 3D

# Issues of Arrows in 3D

**Common problems:**

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

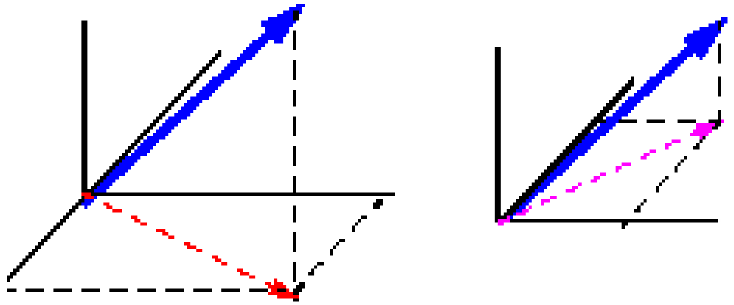




# Issues of Arrows in 3D

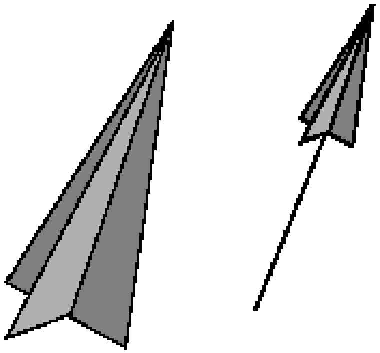
**Common problems:**

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



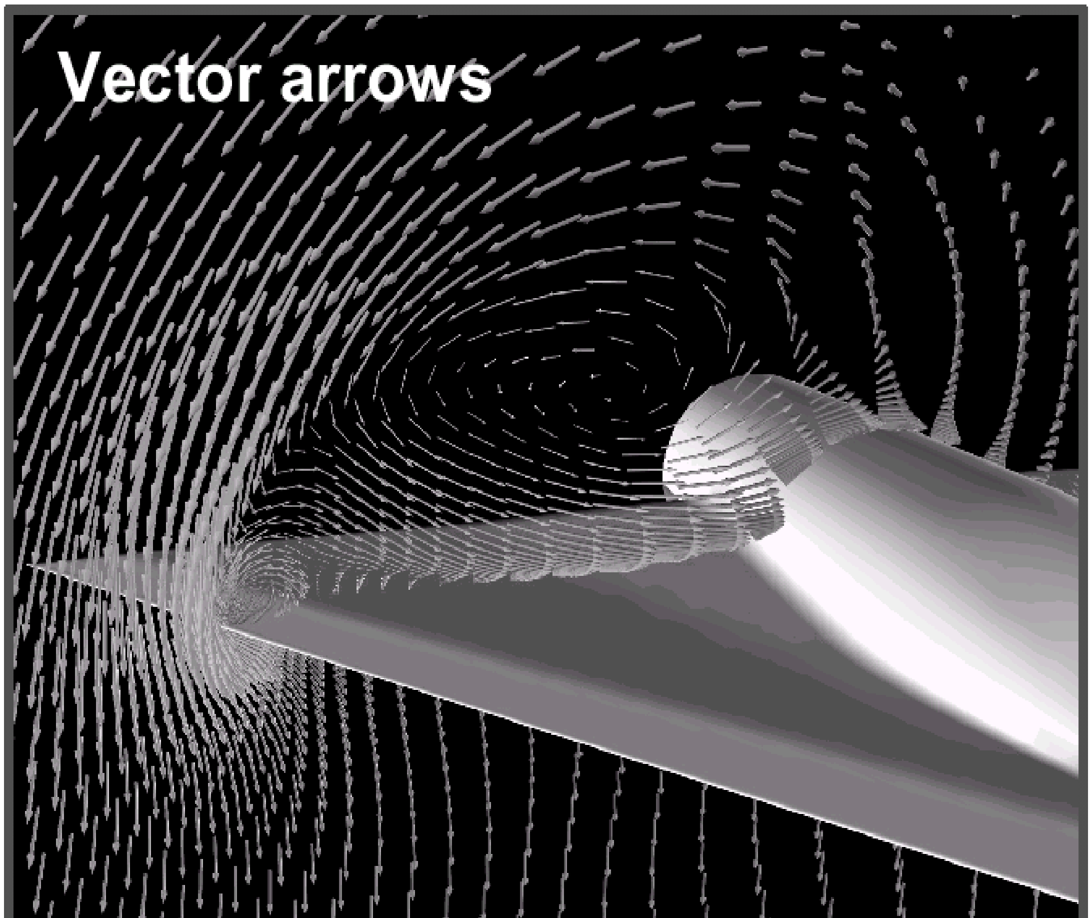
**Remedy:**

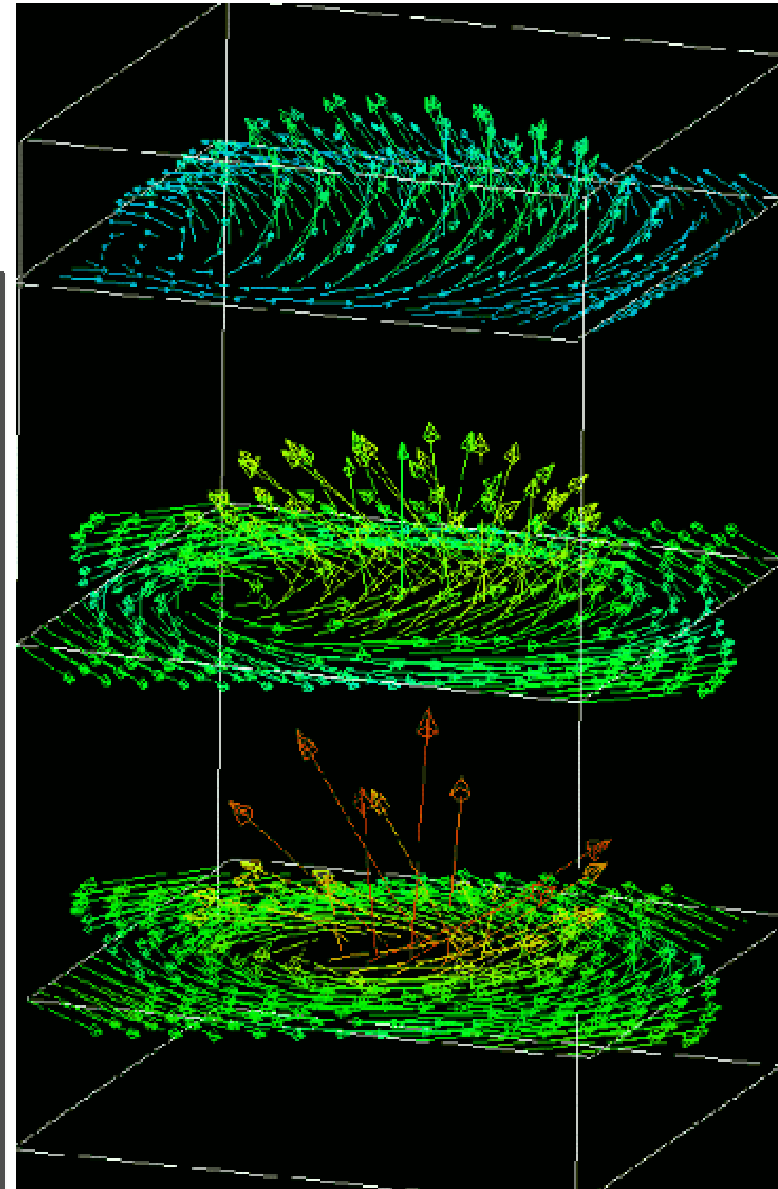
- **3D**-Arrows
  (are of some help)



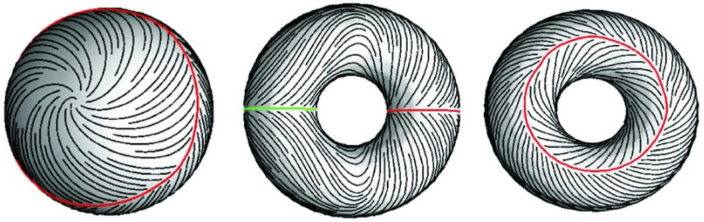
# Arrows in 3D – Examples

Compromise:
  arrows only in layers

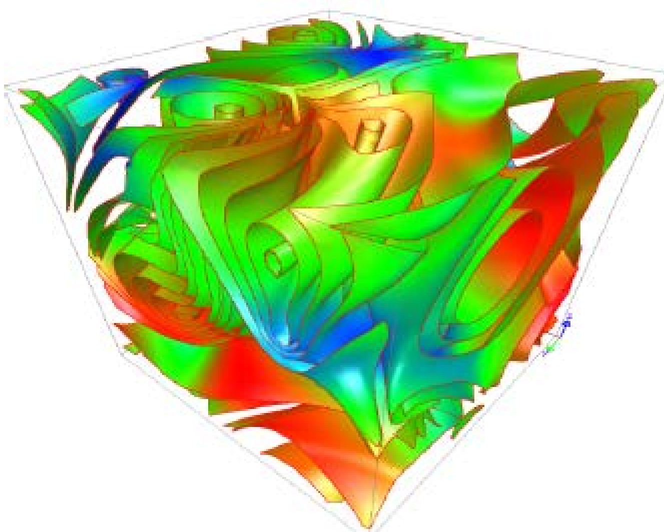


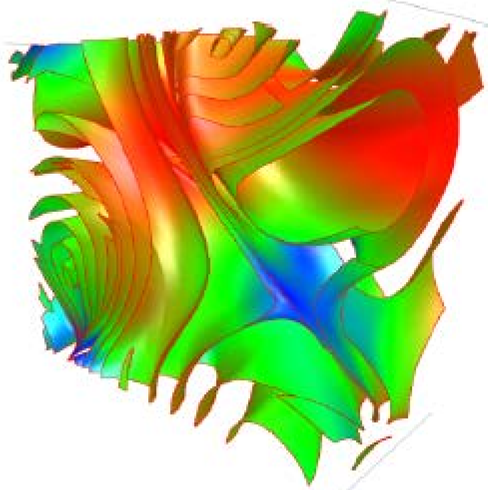



aching/csMU7-vis/

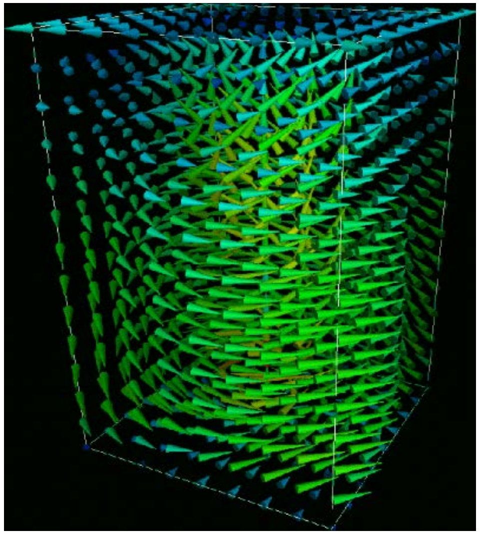


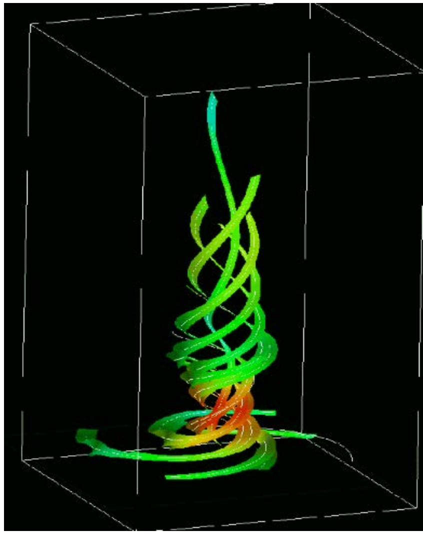
# Geometric-based Methods: Integral curves and surfaces





# Direct vs. Geometric FlowVis



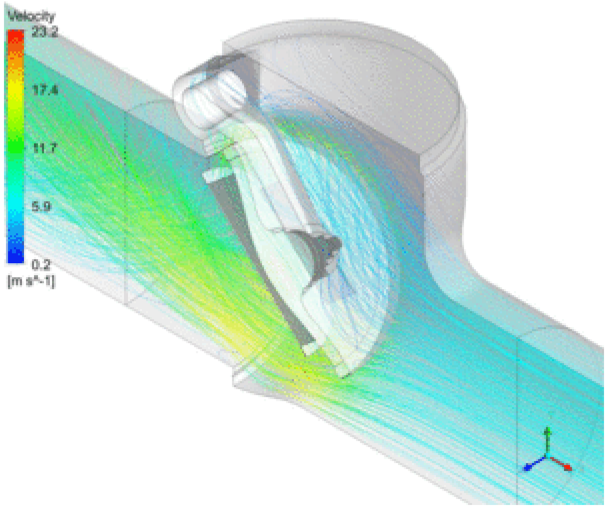


## Direct flow visualization:

- 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*



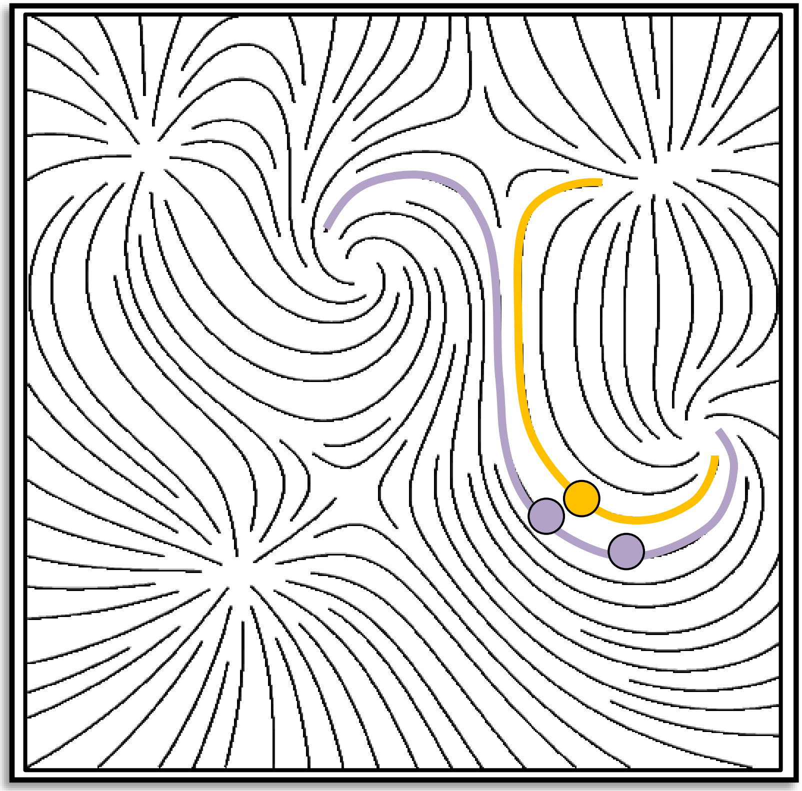
# Streamlines – Theory

- flow data $\mathbf{v}$: derivative information
  - $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{v}(\mathbf{x})$;
    spatial points $\mathbf{x} \in \mathrm{R}^n$, Time $t \in \mathrm{R}$, flow vectors $\mathbf{v} \in \mathrm{R}^n$

- streamline $\mathbf{s}$: integration over time, also called trajectory, solution, curve
  - $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \leq u \leq t} \mathbf{v}(\mathbf{s}(u))\, \mathrm{d}u$;
    seed point $\mathbf{s}_0$, integration variable $u$

# Streamlines – Theory

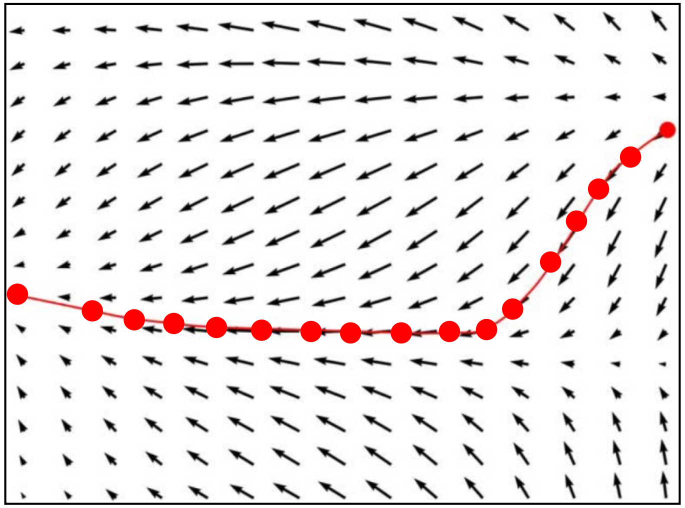
- flow data $\mathbf{v}$: derivative information
  - $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{v}(\mathbf{x})$;
    spatial points $\mathbf{x} \in \mathrm{R}^n$, Time $t \in \mathrm{R}$, flow vectors $\mathbf{v} \in \mathrm{R}^n$
- streamline $\mathbf{s}$: integration over time, also called trajectory, solution, curve
  - $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \leq u \leq t} \mathbf{v}(\mathbf{s}(u))\, \mathrm{d}u$;
    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:

- Mathematical expression: $\mathbf{s}(t) = \mathbf{s}_0 + \int_{0 \le u \le t} \mathbf{v}(\mathbf{s}(u))\,\mathrm{d}u$
- 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 ($\mathrm{d}t$) and therefore distance

Euler integration: $\mathbf{s}_{i+1} = \mathbf{s}_i + \mathbf{v}(\mathbf{s}_i) \cdot dt$,
integration of small steps ($\mathrm{d}t$ very small)

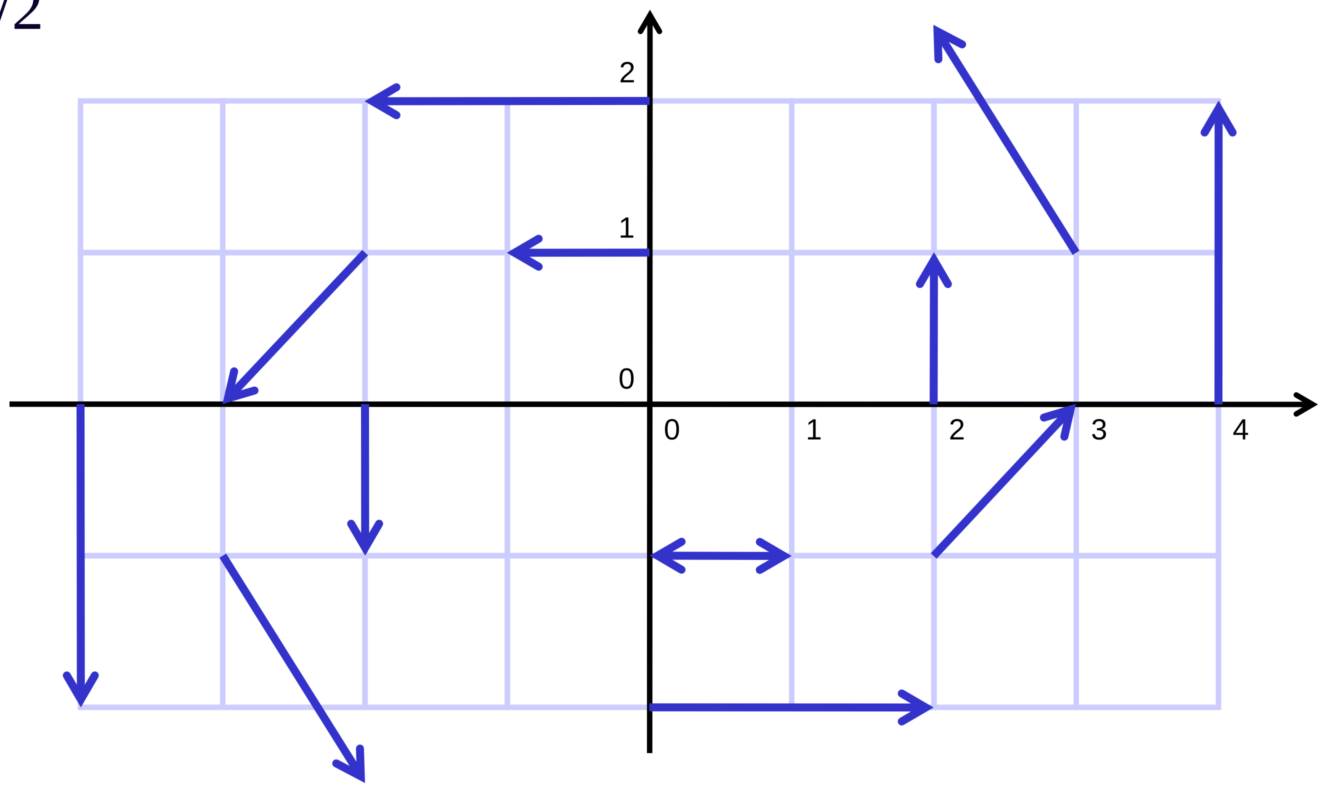
# Euler Integration – Example

2D model data:

$v_x = dx/dt = -y$

$v_y = dy/dt = x/2$

Sample arrows:



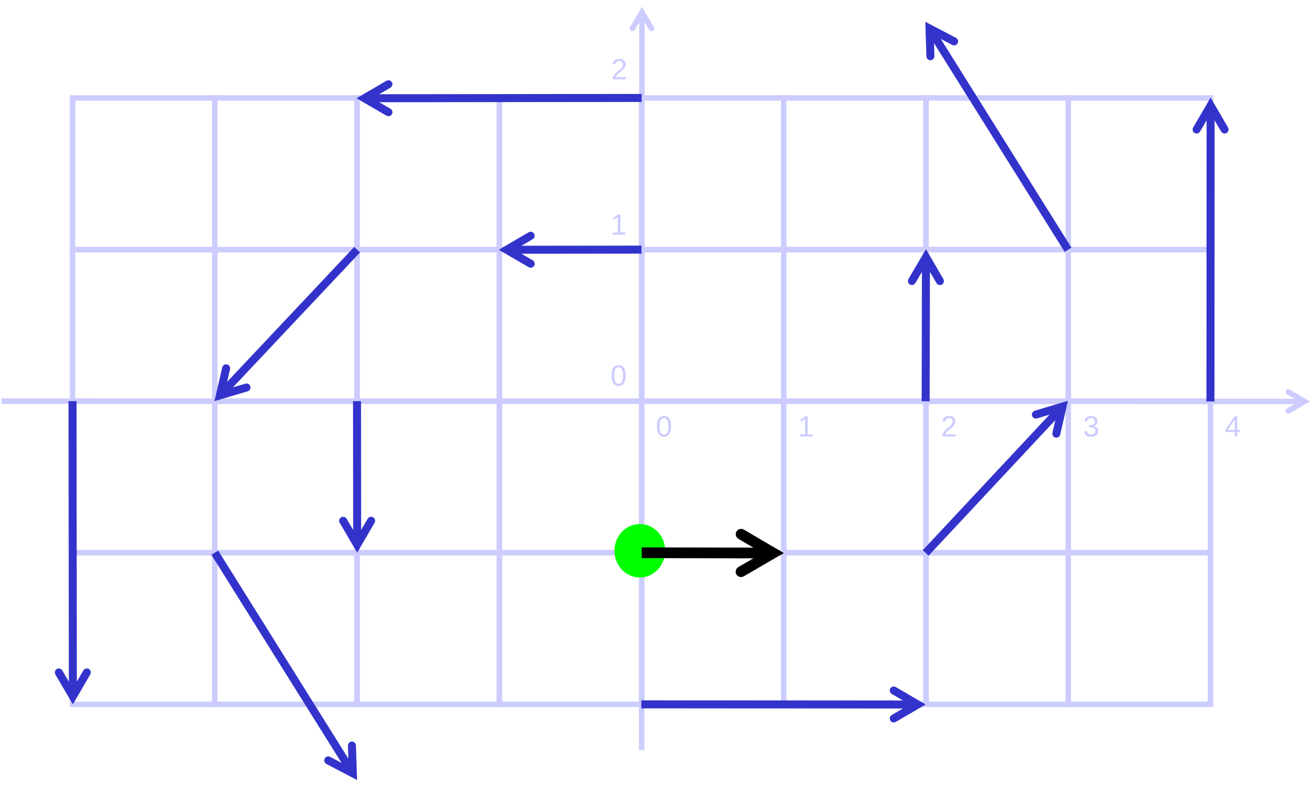

True solution: ellipses.

# Euler Integration – Example

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

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

$v_y = dy/dt = x/2$

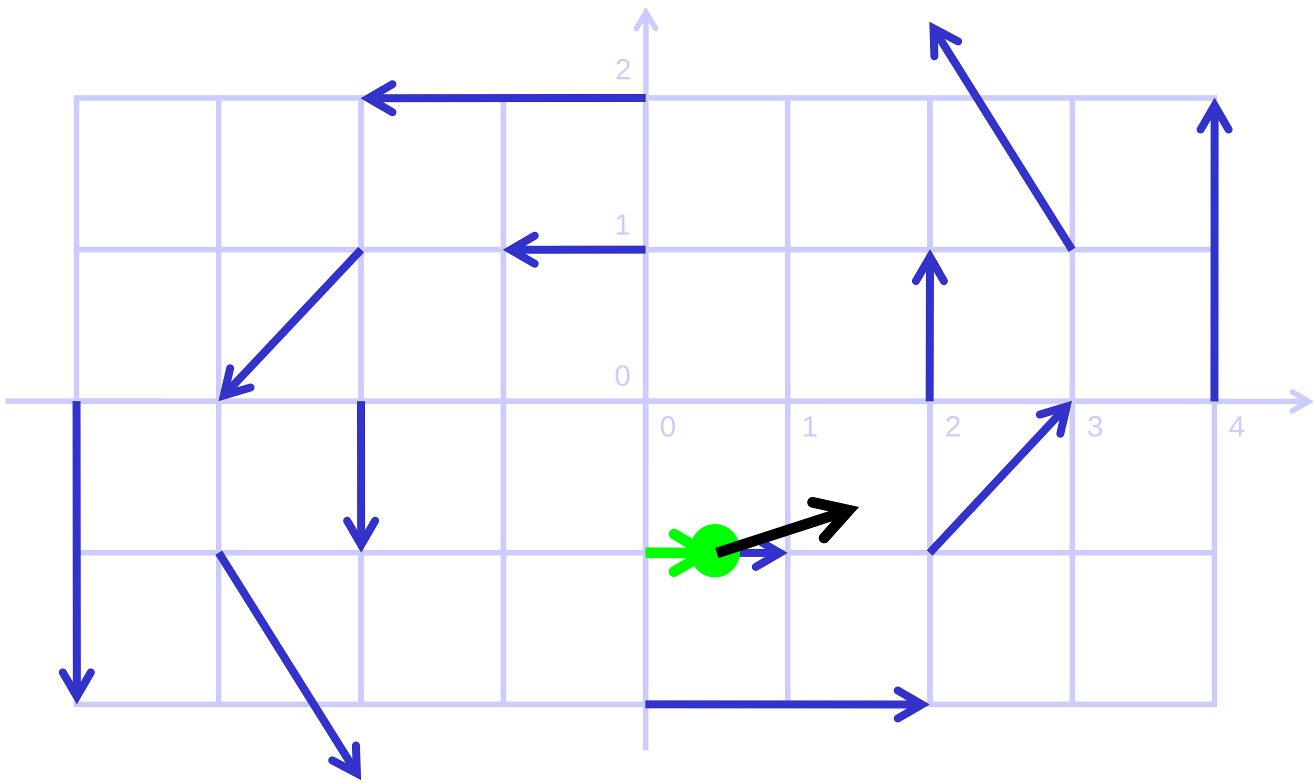


# Euler Integration – Example

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

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

$v_y = dy/dt = x/2$

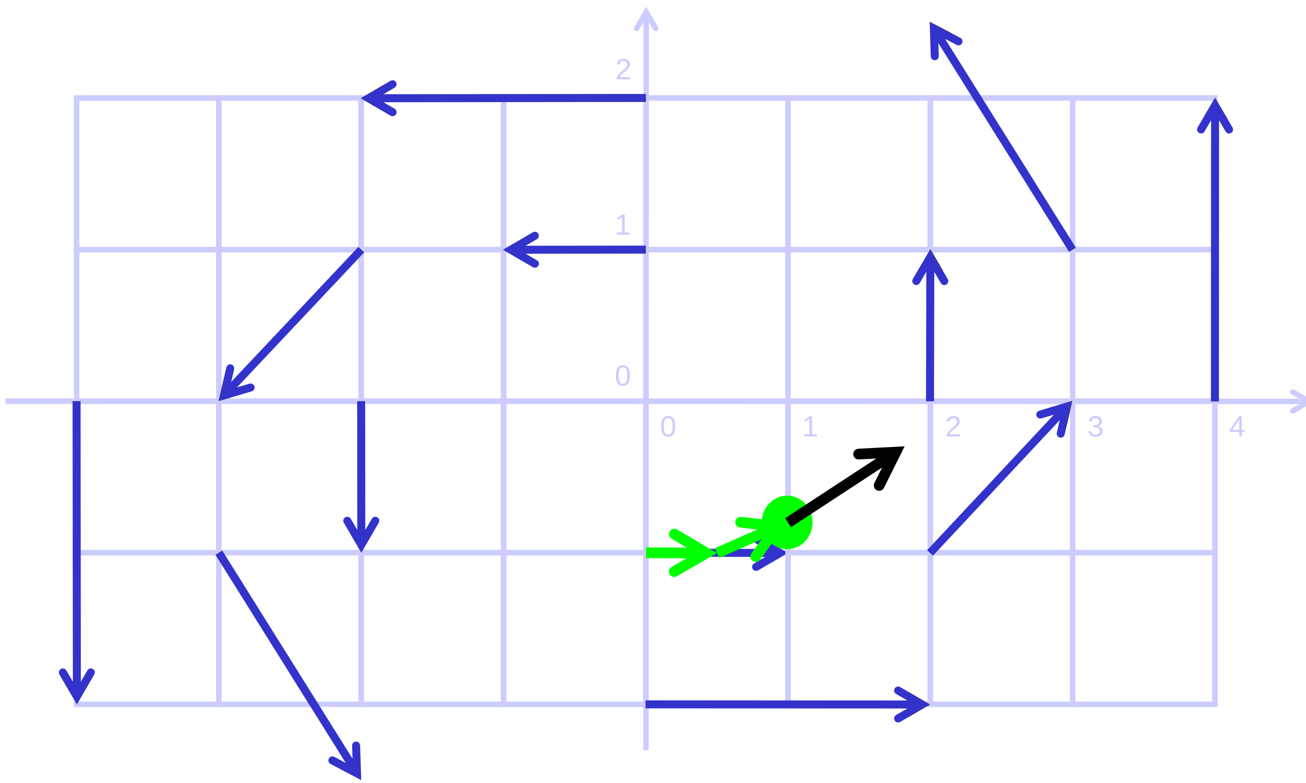


# Euler Integration – Example

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

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

$v_y = dy/dt = x/2$



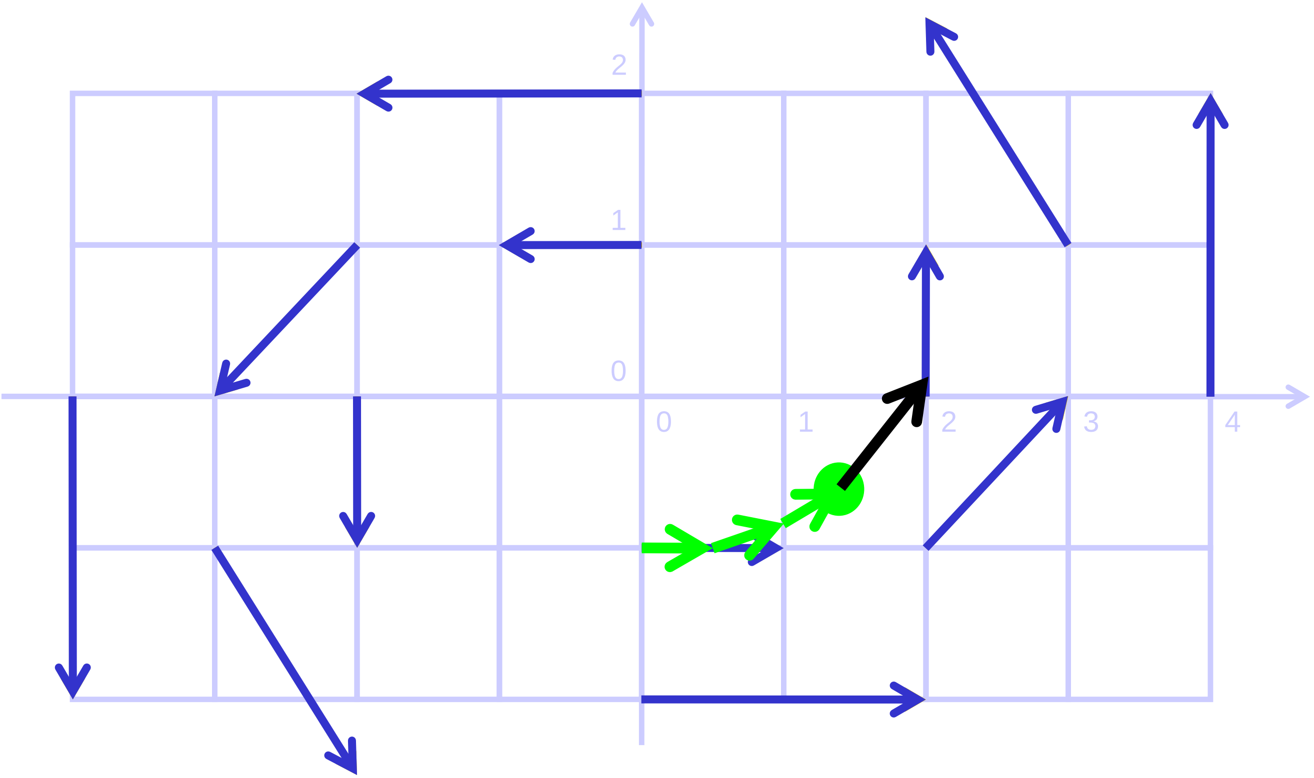
# Euler Integration – Example

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

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

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

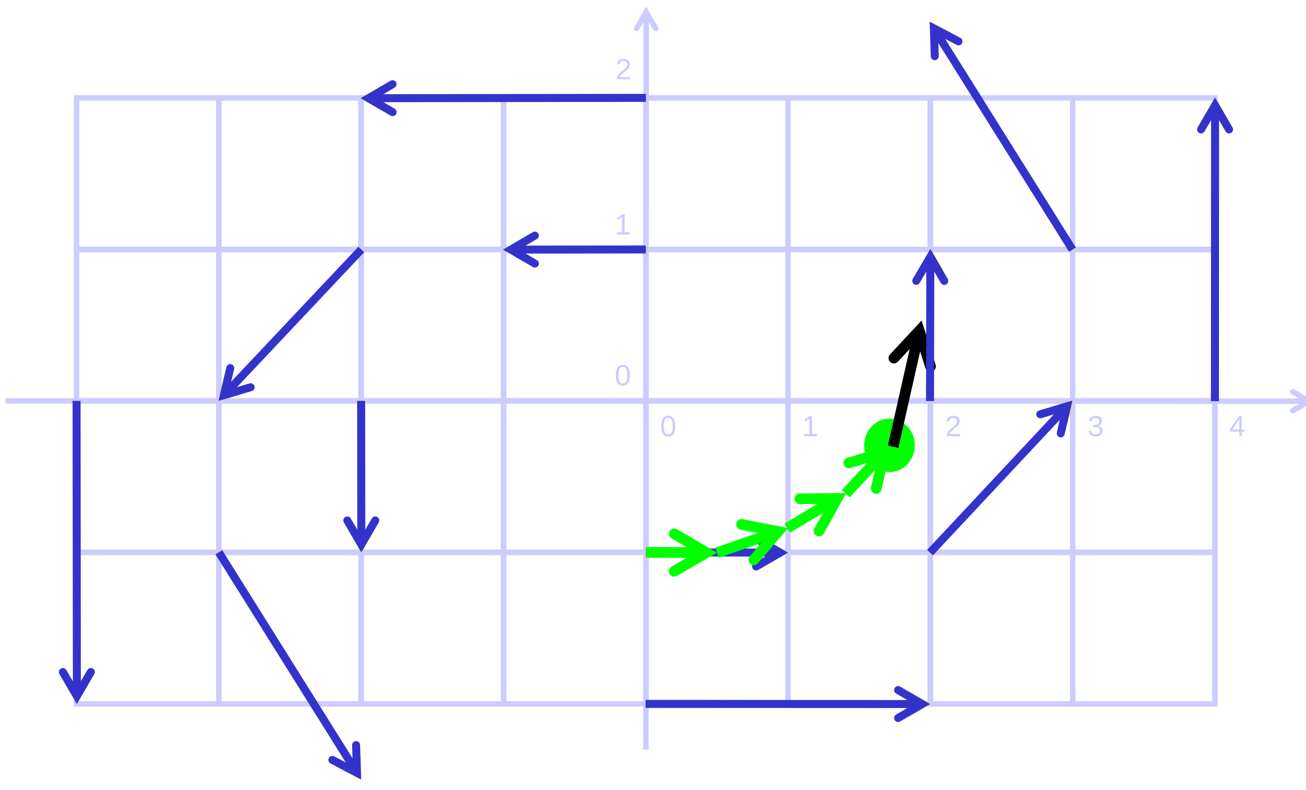
$v_y = dy/dt = x/2$



# Euler Integration – Example

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

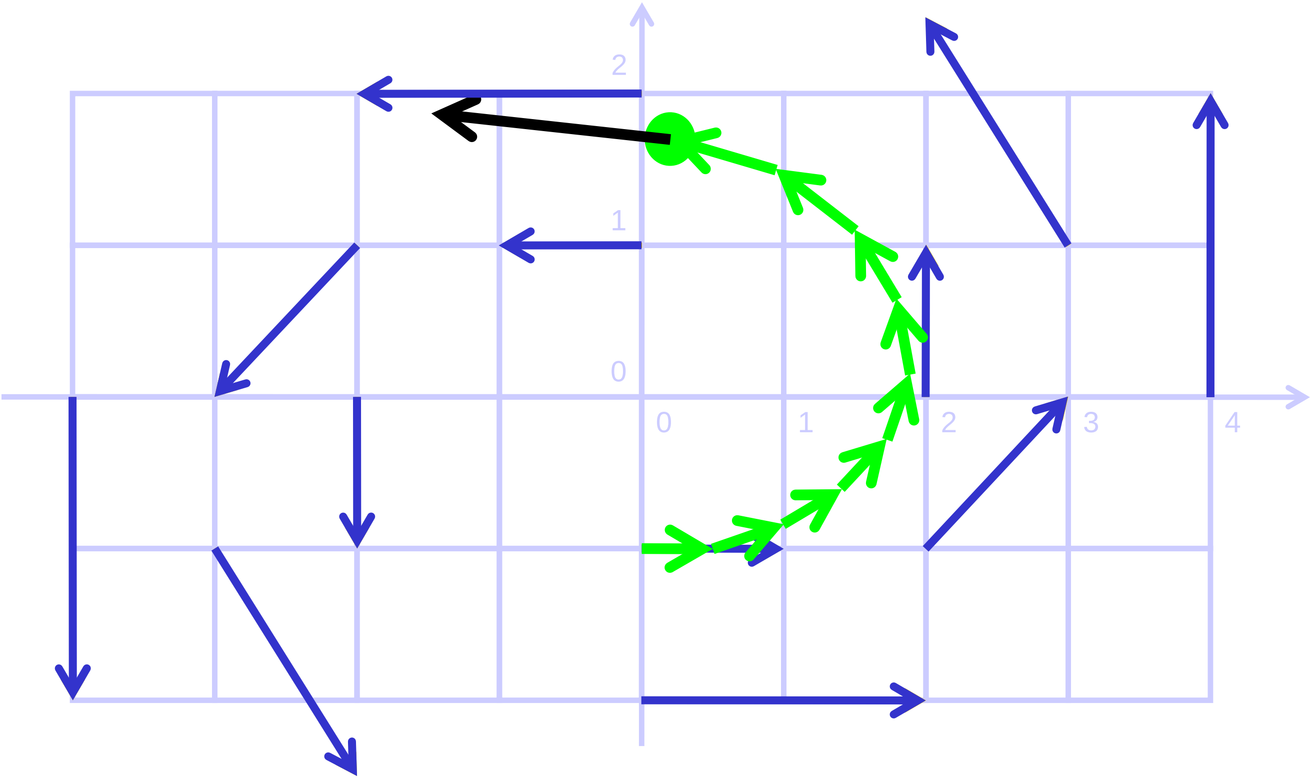
$\mathbf{v}(\mathbf{s}_4) \quad = (17/64 \mid 7/8)^T \quad \approx (0.27 \mid 0.88)^T;$



# Euler Integration – Example

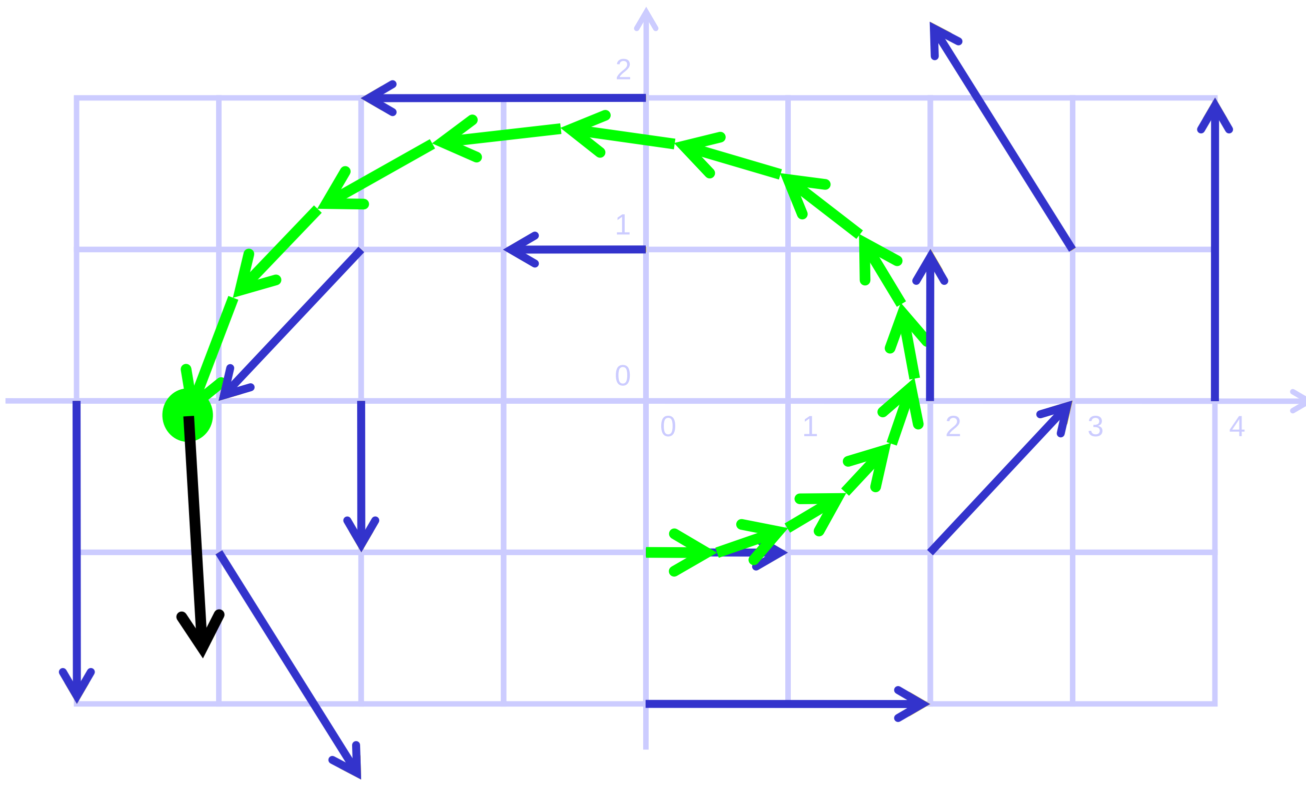
- $\mathbf{s}_9 \approx (0.20 | 1.69)^T$;

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



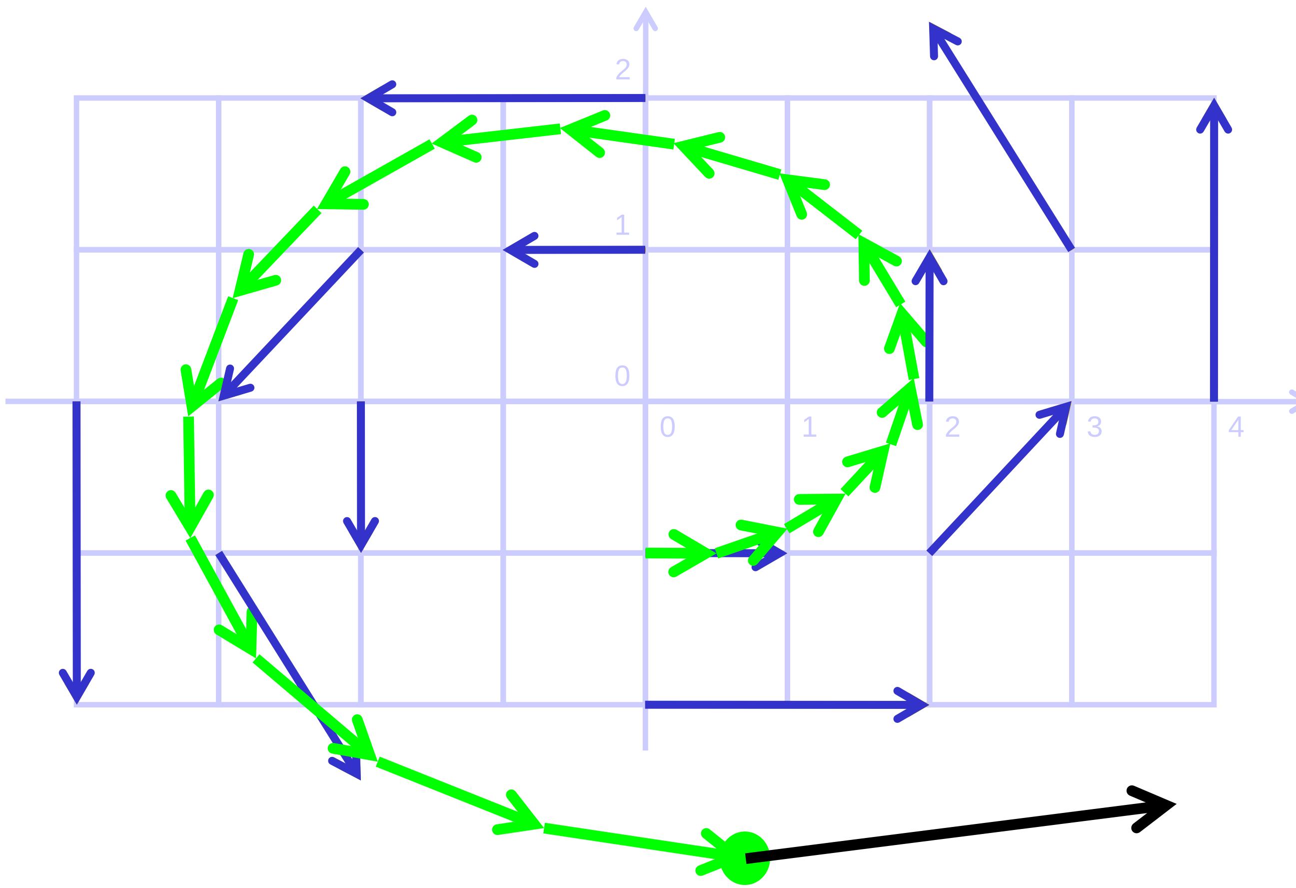
# Euler Integration – Example

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



# 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

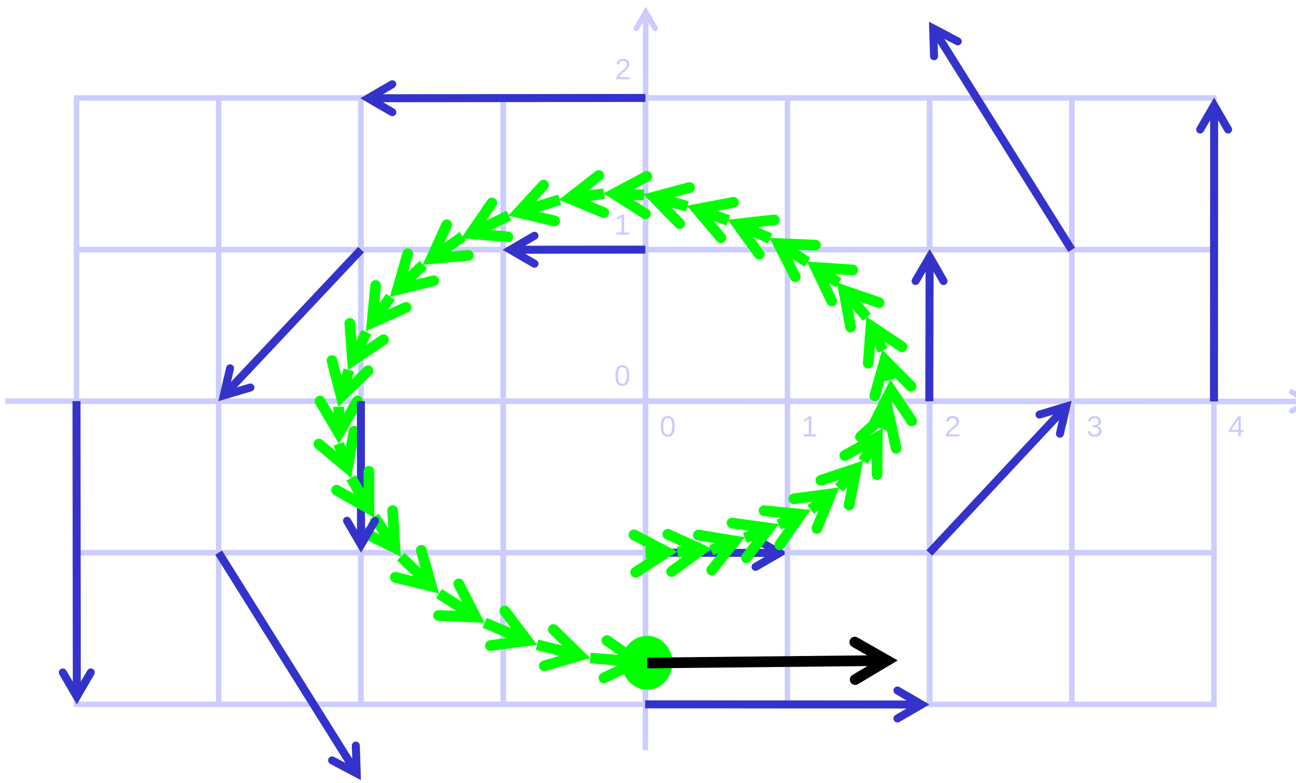


# Euler Integration – Example

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

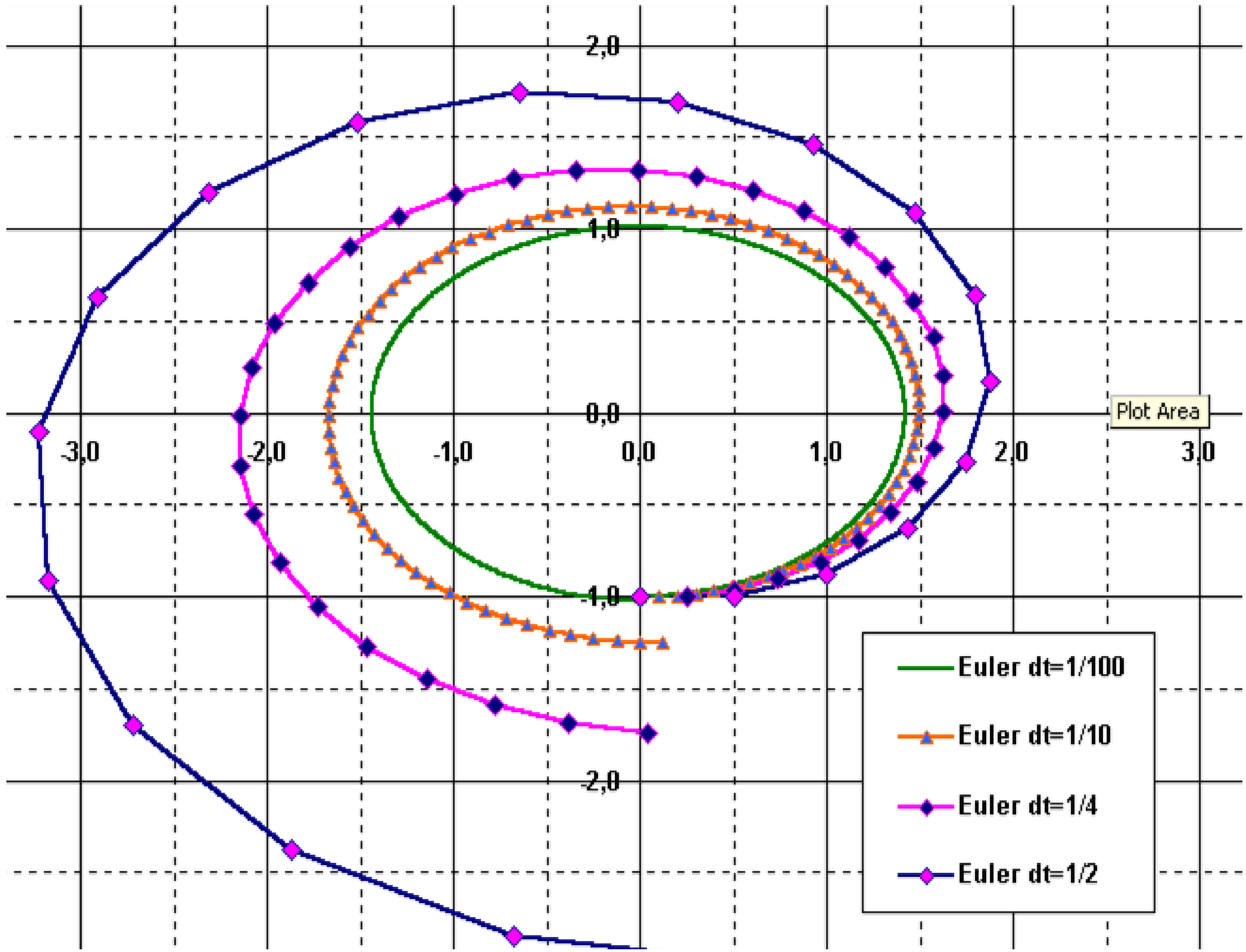
$\mathbf{s}_{36} \approx (0.04 | \text{-}1.74)^T$; $\mathbf{v}(\mathbf{s}_{36}) \approx (1.74 | 0.02)^T$;

- 36 steps



# Comparison Euler, Step Sizes

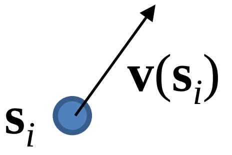
Euler
quality is proportional to d$t$



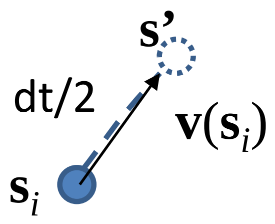
# Euler Example – Error Table

| 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% |

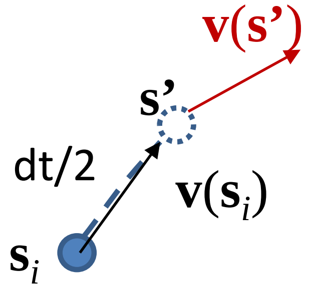
# RK-2 – A Quick Round



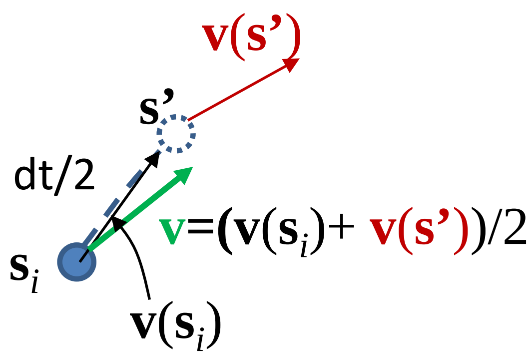
# RK-2 – A Quick Round



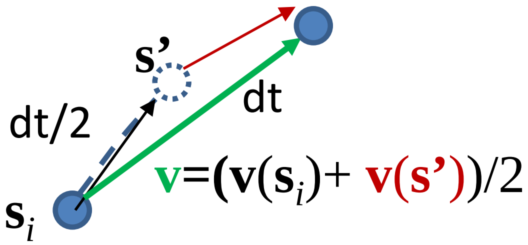
# RK-2 – A Quick Round



# RK-2 – A Quick Round



# RK-2 – A Quick Round



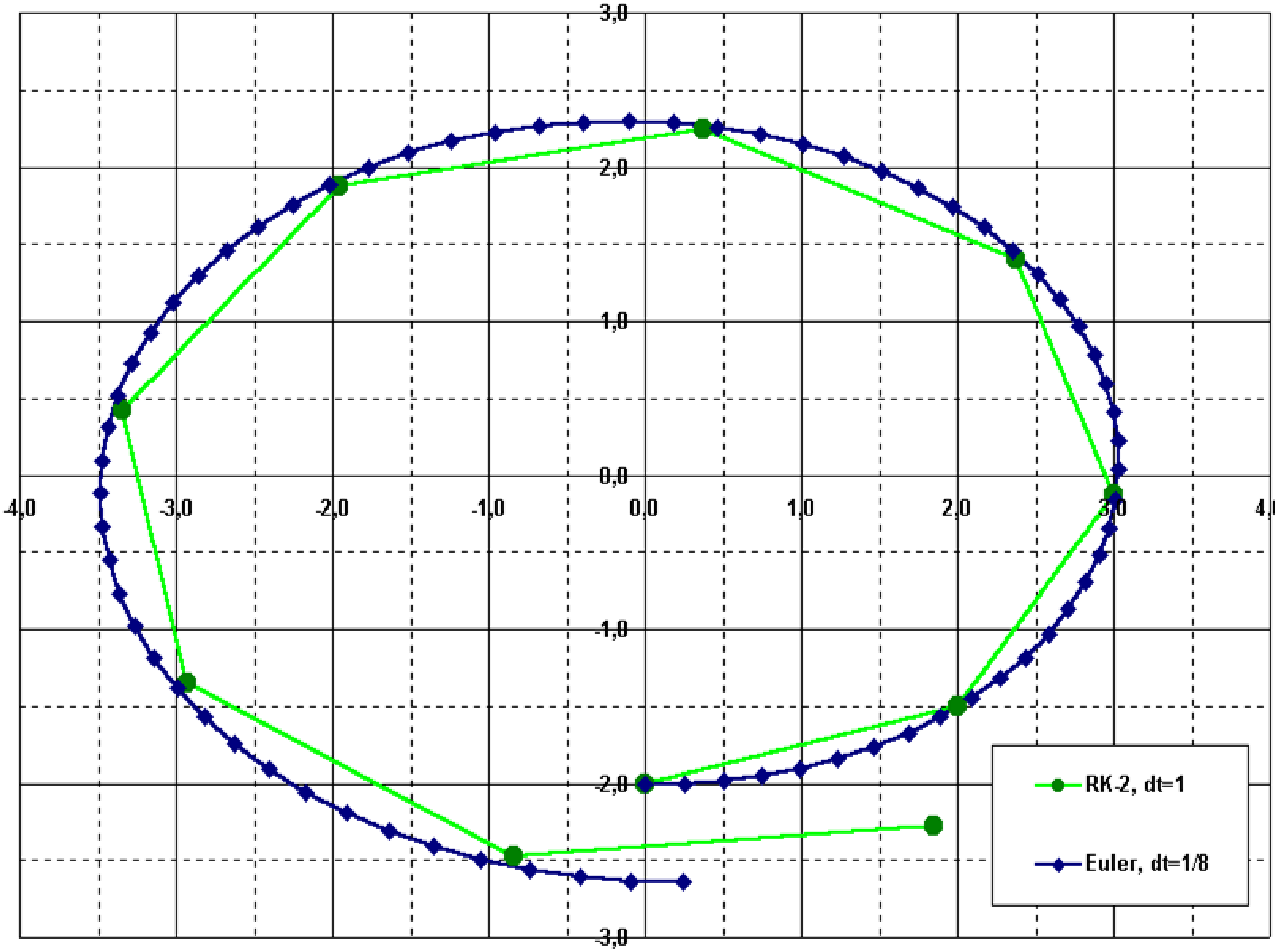

$\mathbf{s}' = \mathbf{s}_i + (\mathbf{dt}/2)*\mathbf{v}(\mathbf{s}_i)$

$\mathbf{v} = (\mathbf{v}(\mathbf{s}_i) + \mathbf{v}(\mathbf{s}'))/2$

$\mathbf{s}_{i+1} = \mathbf{s}_i + \mathbf{dt}*\mathbf{v}$

# 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:

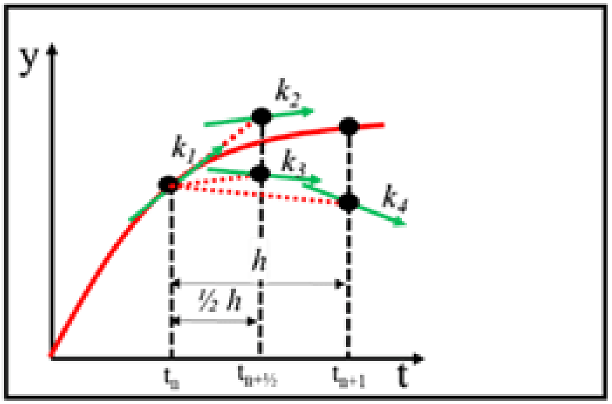
- four vectors $\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3, \mathbf{k}_4$
- one step is a convex combination:
  $\mathbf{s}_{i+1} = \mathbf{s}_i + (\mathbf{k}_1 + 2\cdot \mathbf{k}_2 + 2\cdot \mathbf{k}_3 + \mathbf{k}_4)/6$
- vectors:

  $\mathbf{k}_1 = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i)$ … original vector

  $\mathbf{k}_2 = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i + \mathbf{k}_1/2)$ … RK-2 vector

  $\mathbf{k}_3 = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i + \mathbf{k}_2/2)$ … use RK-2 …

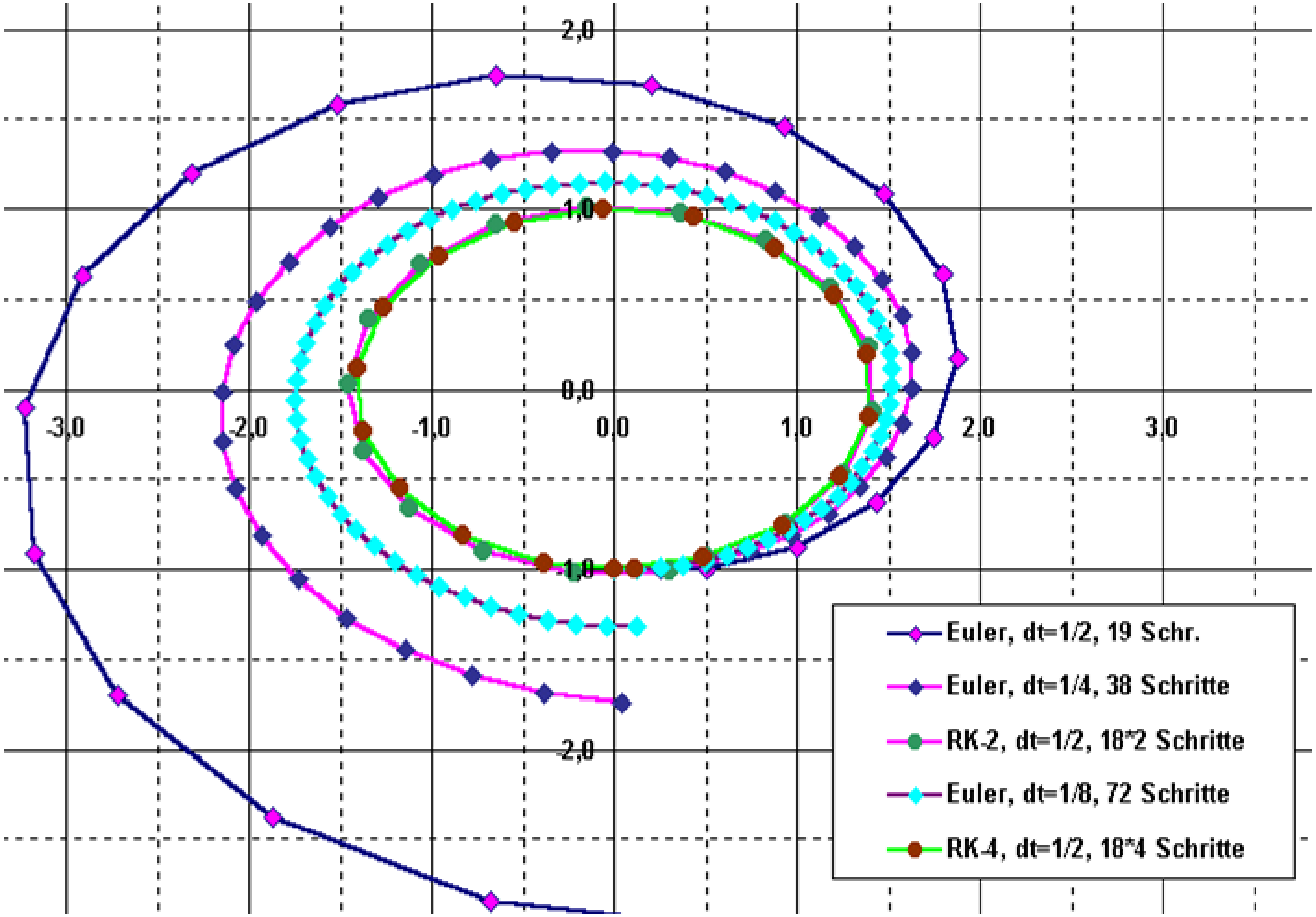
  $\mathbf{k}_4 = \mathrm{d}t\cdot\mathbf{v}(\mathbf{s}_i + \mathbf{k}_3)$ … and again



# Euler vs. Runge-Kutta

RK-4: pays off only with complex flows

Here approx. like RK-2



Taylor expansion

$f(x+a)=$

$$f(a)+\frac{f'(a)}{1!}(x-a)+\frac{f''(a)}{2!}(x-a)^2+\frac{f'''(a)}{3!}(x-a)^3+\cdots,$$

# 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 $\mathrm{d}t$
- RK: more accurate in higher orders
- furthermore: adaptive methods, implicit methods, etc.

# Streamline Termination

When to stop streamline integration (termination condition):

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

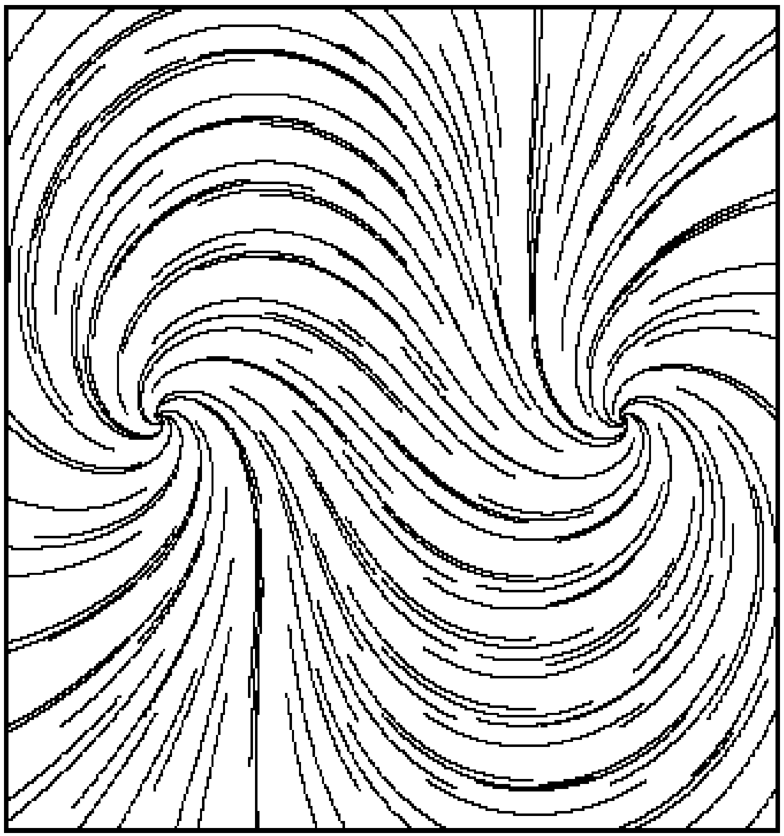
# Streamline Placement

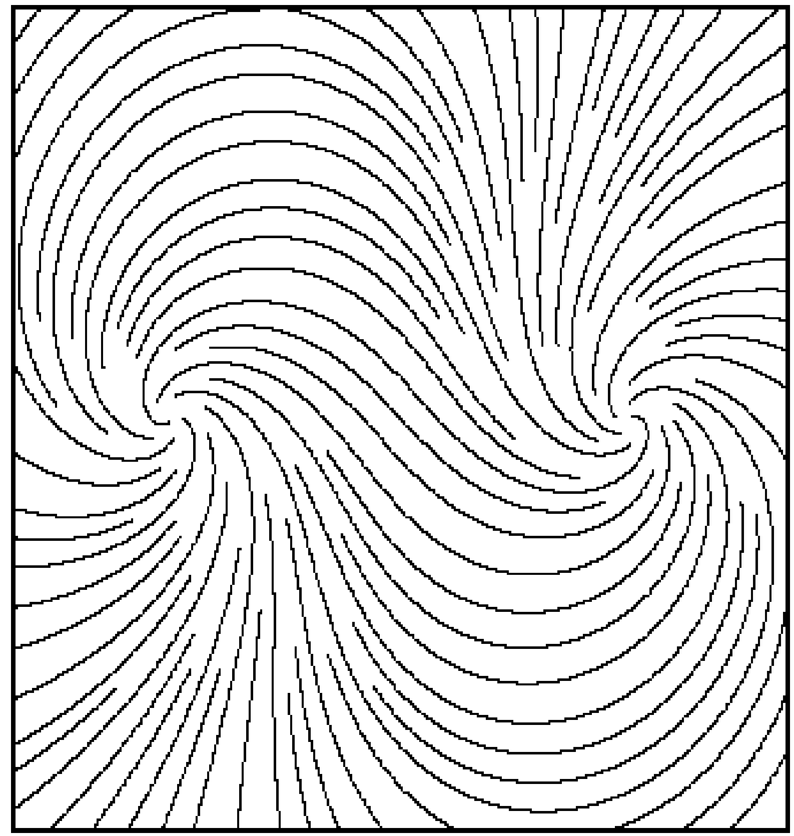
in 2D

# 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 $d_{\text{sep}}$ 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

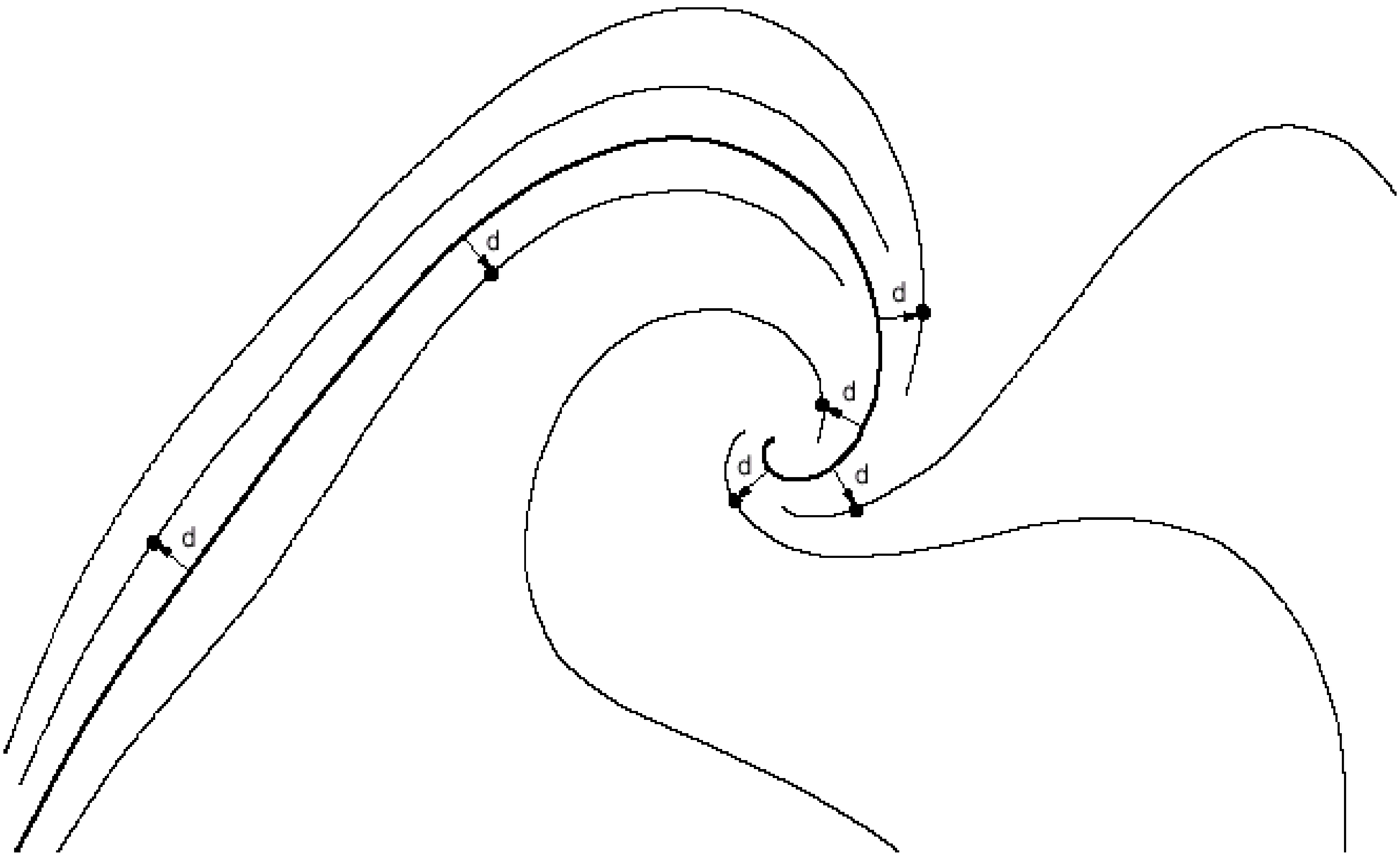
  ELSE next streamline in queue becomes current streamline

# Streamline Termination

When to stop streamline integration (termination condition):

- when distance to neighboring streamline $\leq d_{\text{test}}$
- 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

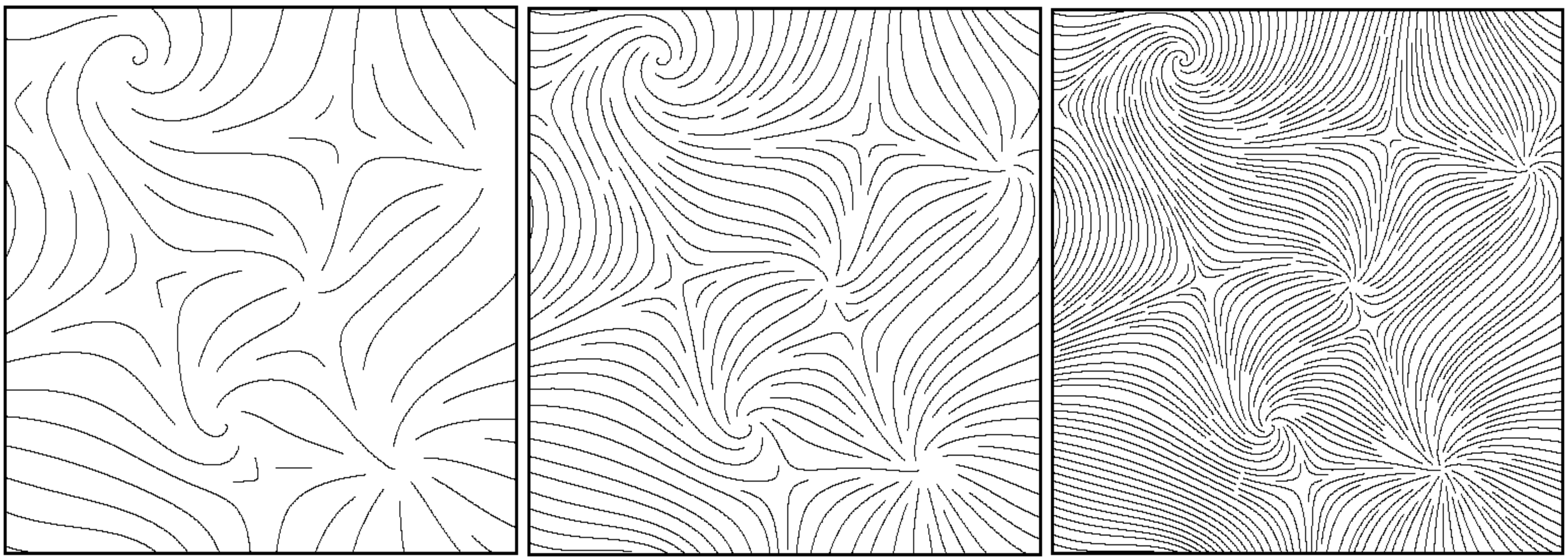
# New Streamlines



# Different Streamline Densities

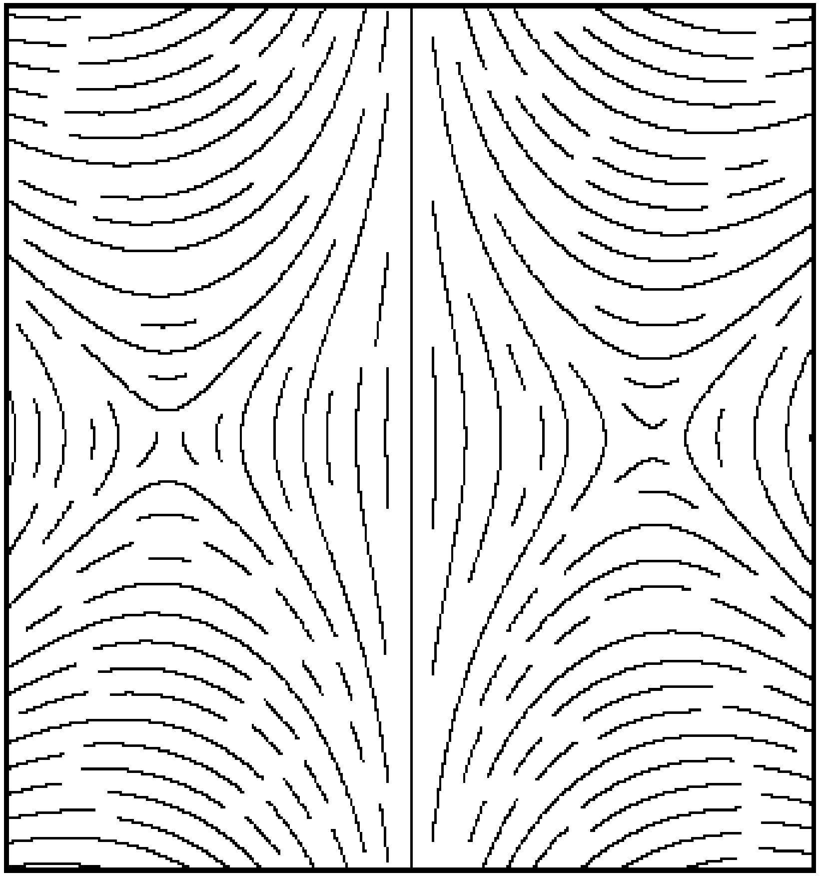
Variations of $d_{sep}$ relative to image width:

6% 3% 1.5%

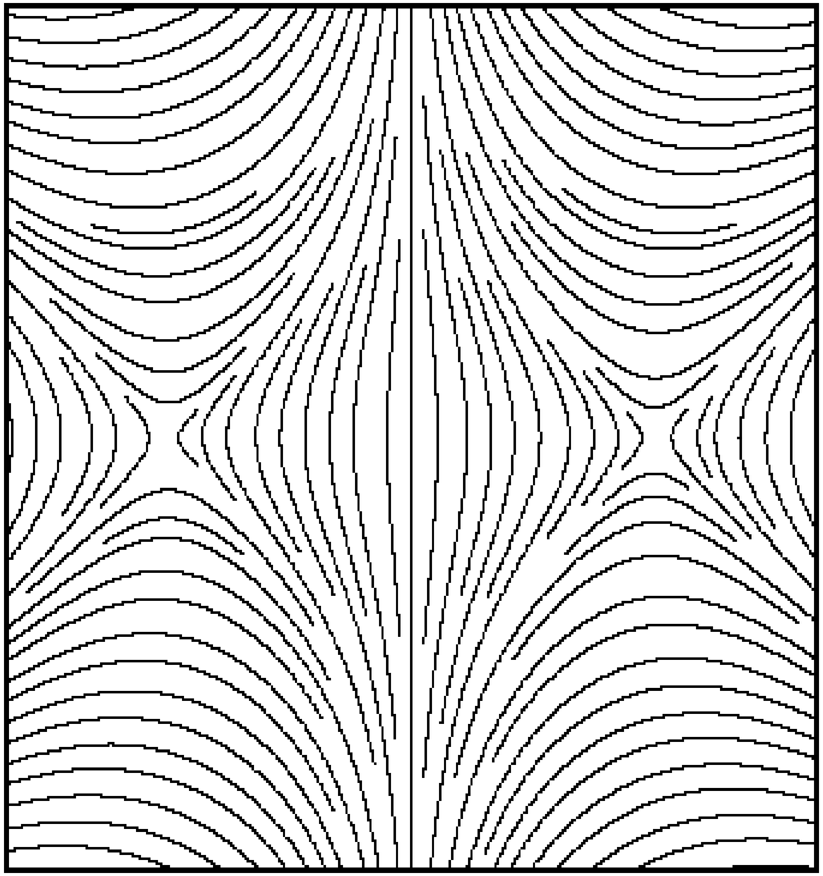


# $d_{sep}$ vs. $d_{test}$

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

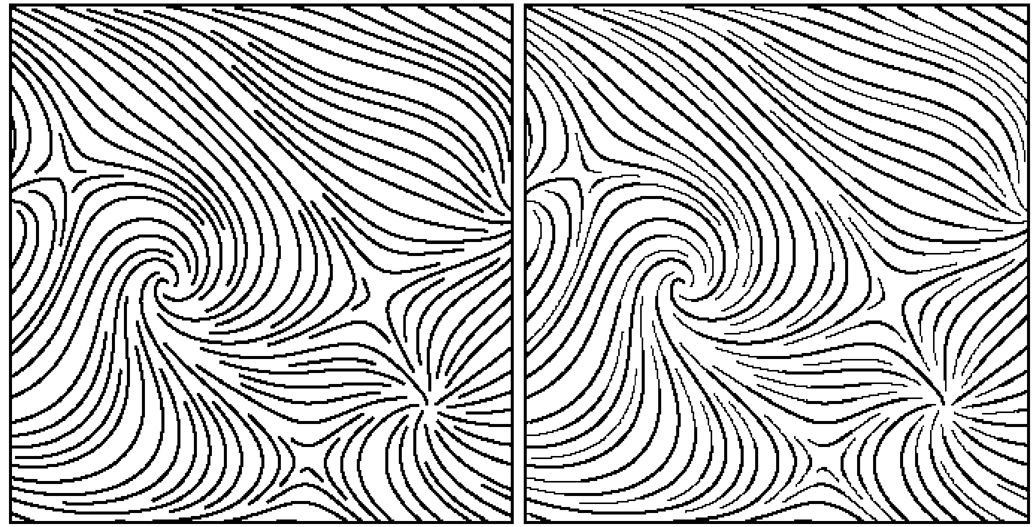


$d_{test} = 0.5 \cdot d_{sep}$

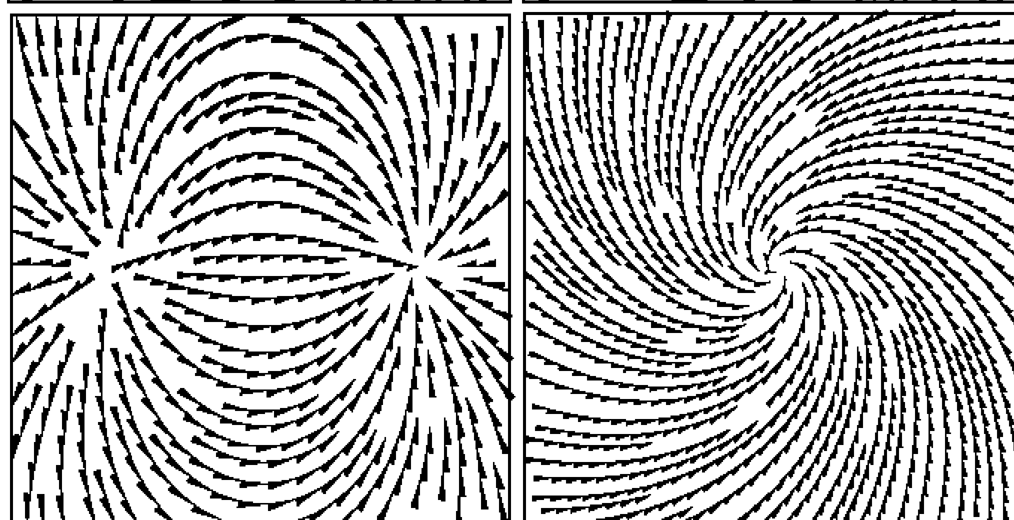


# Tapering and Glyphs

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

# Acknowledgment

Thanks for the materials

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