Vector Field Visualisation

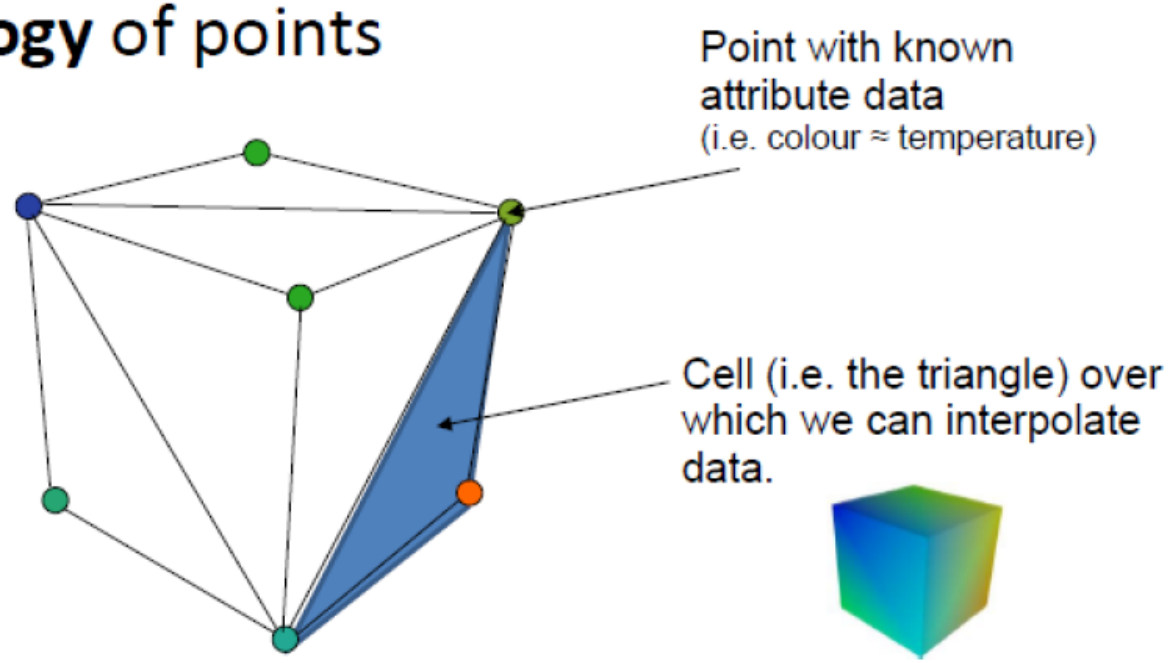
Computer Animation and Visualization
Lecture 11

Taku Komura

Institute for Perception, Action & Behaviour
School of Informatics

Dataset Representation of the Structure

- **Points** specify where the data is known
  - **specify geometry** in $\mathbb{R}^N$
- **Cells** allow us to interpolate between points
  - **specify topology** of points



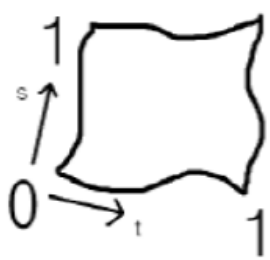
Cells

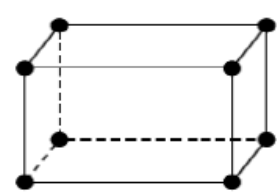
- **Fundamental building blocks of visualisation**
  - *our gateway from discrete to interpolated data*
- Various Cell Types
  - defined by **topological dimension**
  - specified as an ordered point list (connectivity list)
  - primary or composite cells
    - composite : consists of one or more primary cells

Topological Dimension

- **Topological Dimension:** number of independent continuous variables specifying a position within the topology of the data
  - different from **geometric dimension** (position within general space)

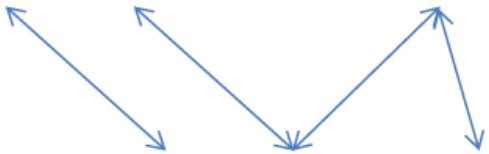
	Topological	Geometric	
point	0D	2D/3D	e.g. 2D (x,y)
curve	1D	2D/3D	e.g. 3D point on curve
surface	2D	3D (in general)	
volume	3D	3D	e.g. MRI or CT scan
Temporal volume	4D	3D (+ t)	

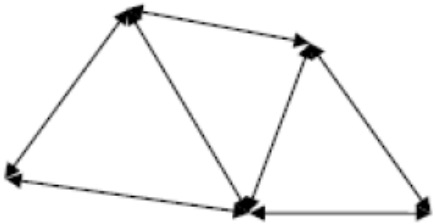


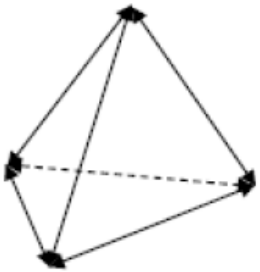


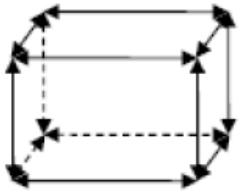
Various Cell types

- **Vertex (0D)**
- **Line (1D)**
- **Triangle, pixel, quad (2D)**
- **Triangle mesh, polygon mesh (2D)**
- **Tetrahedra, voxel (3D)**







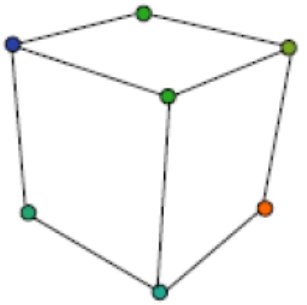


Data Representation

- **Data objects : structure + value**
  - referred to as ***datasets***

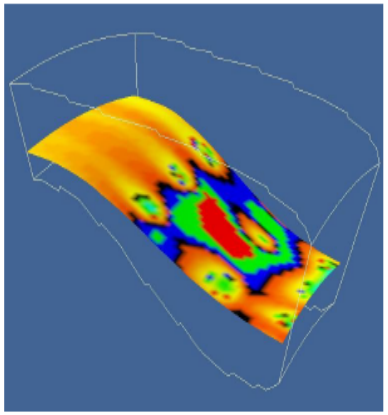
Structure
Points, Cells

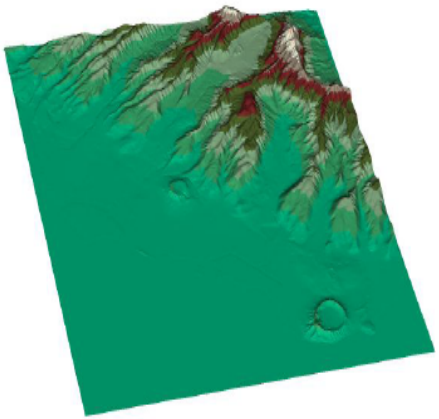
Data Attributes
Scalars, vectors
Normals, texture
coordinates, tensors etc

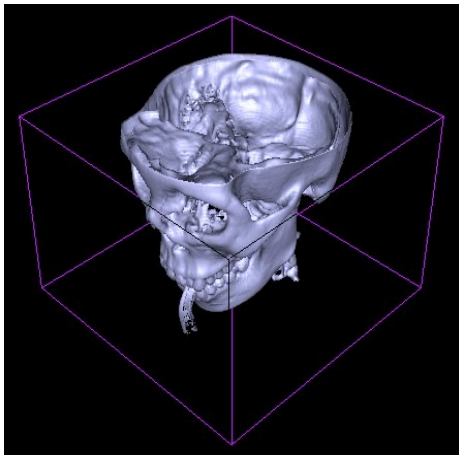


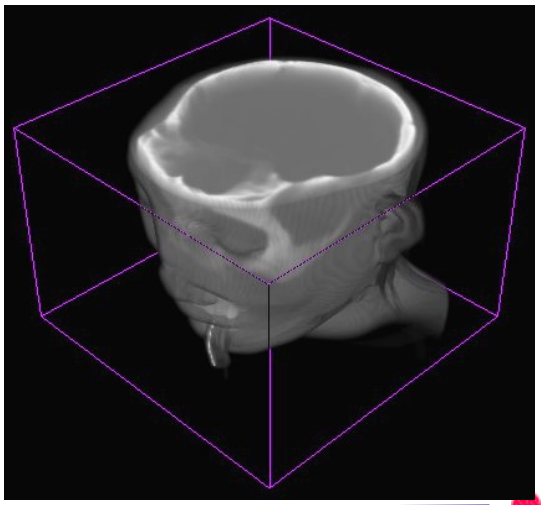
Up-to-now

- Visualising scalar fields : 1D attributes at the sample points



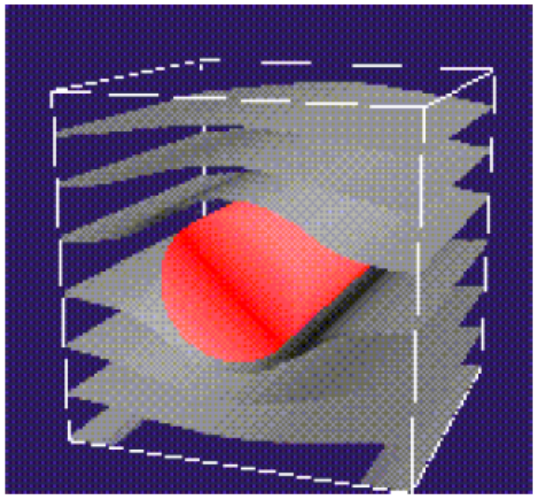




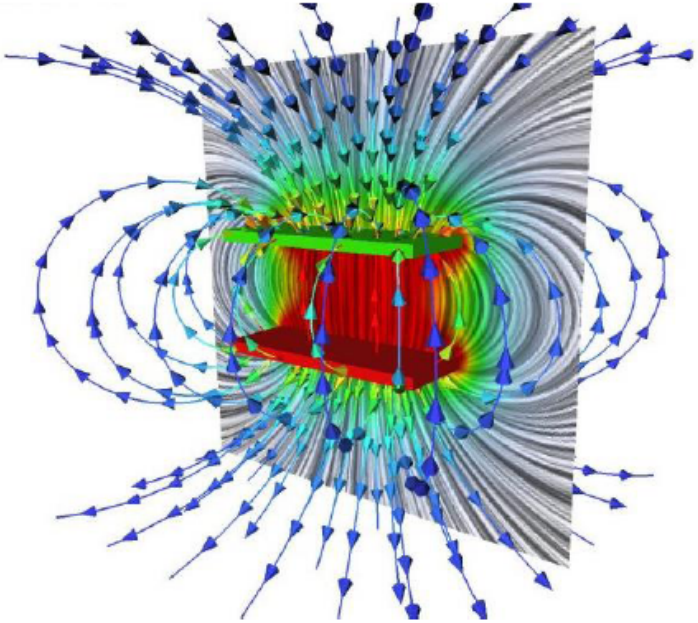


Today: Vector fields

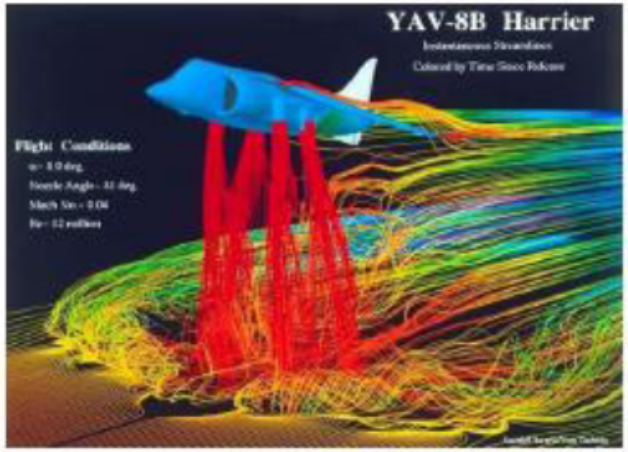
- Visualising vector fields : 2D/3D/nD attributes at the sample points
  - **Magnitude and direction** at each location
    - 3D triplet of values *(i, j, k)*



Force / Displacement



Magnetic Field



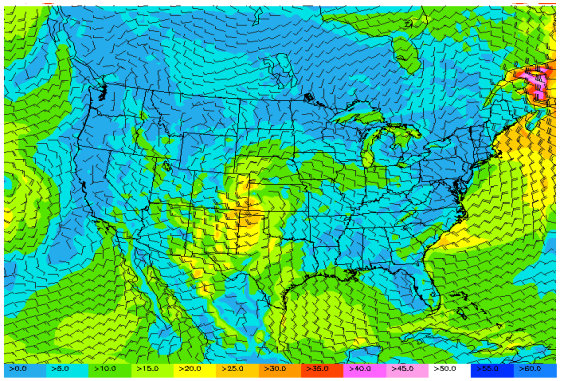
Wind Speed

Overview

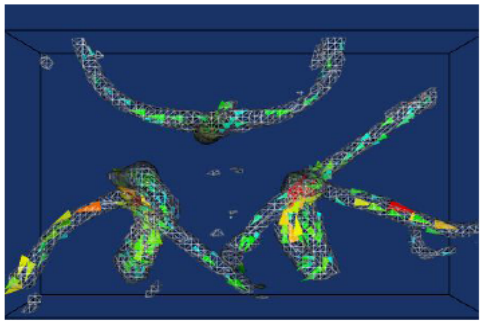
- Local View
  - Warping
  - Glyphs
- Global View
  - Pathline, Streakline, Streamline
  - Integration
  - Stream surface, Stream volume

  - Line Integral Convolution

Visualising Vectors

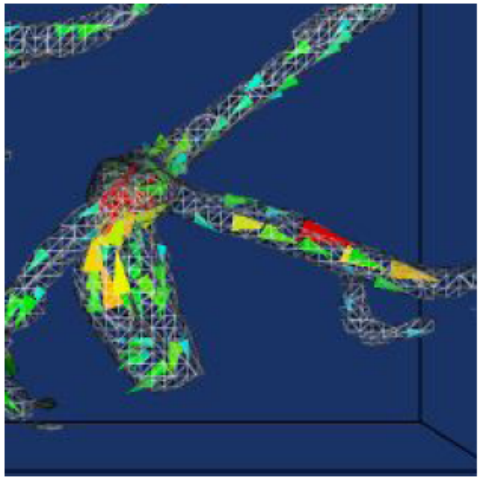


- Examples of vector data:
  - meteorological analyses / simulation
  - medical blood flow measurement
  - Computational simulation of flow over aircraft, ships, submarines etc.
  - visualisation of derivatives of a scalar field



- **Why is visualising these difficult ?**
  - **2 or 3 components per data point, temporal aspects of vector flow, vector density**

Two Methods of Flow Visualisation

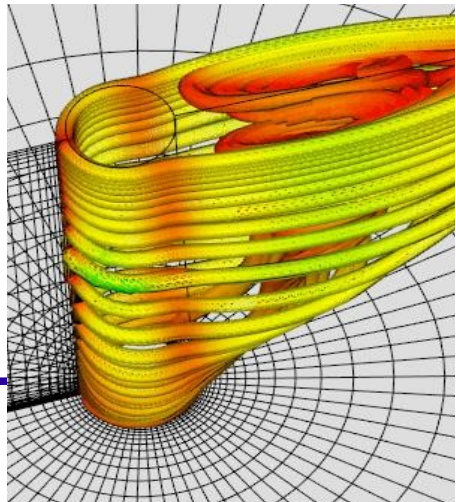


- **Local View** of the vector field
  - Visualise Flow **w.r.t. fixed point**

local direction and magnitude

e.g. for given location, what is the current wind strength and direction

- **Global view of vector field**
  - **Visualise flow as the trajectory of a particles transported by the flow**

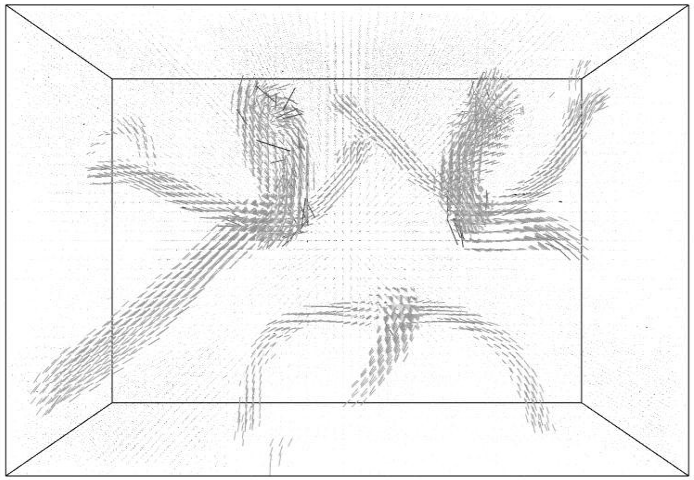


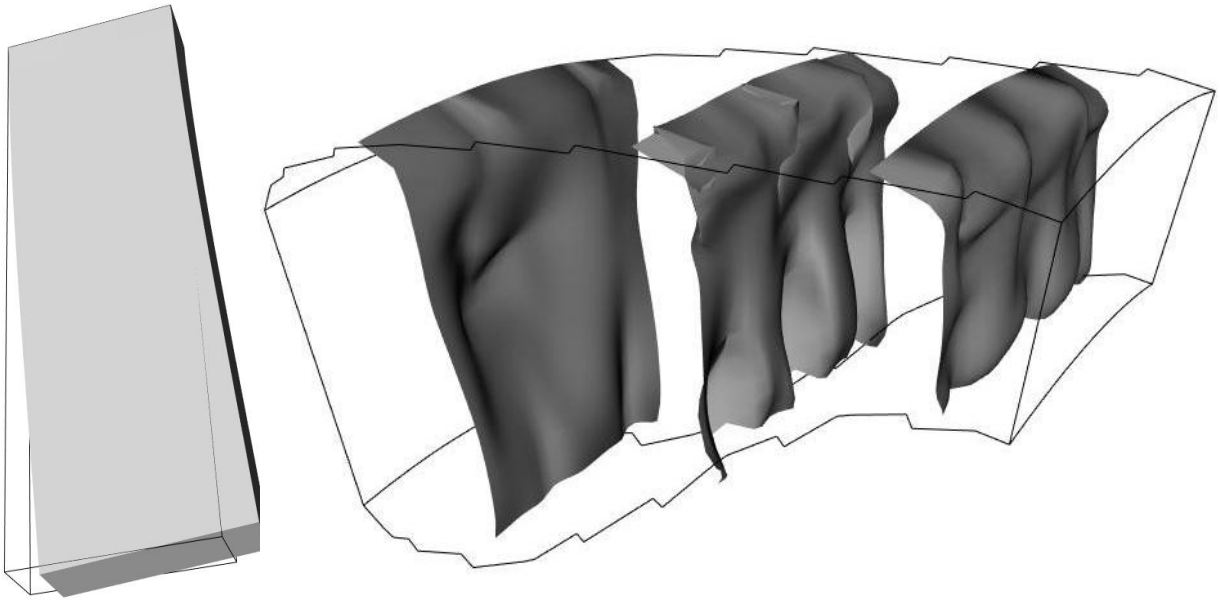
a given location, where has the wind flow come from,

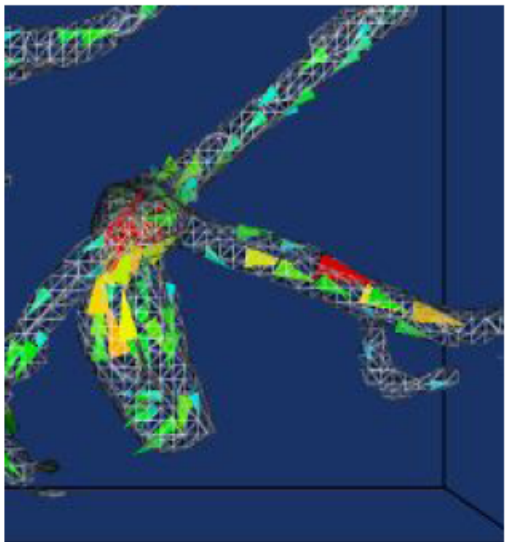
and where will it go to.

Vectors : local visualisation

- Set of basic methods for showing **local view**:
  - **Warping**
  - **Oriented lines, barbs, glyphs**
  - **Can combine with animation**

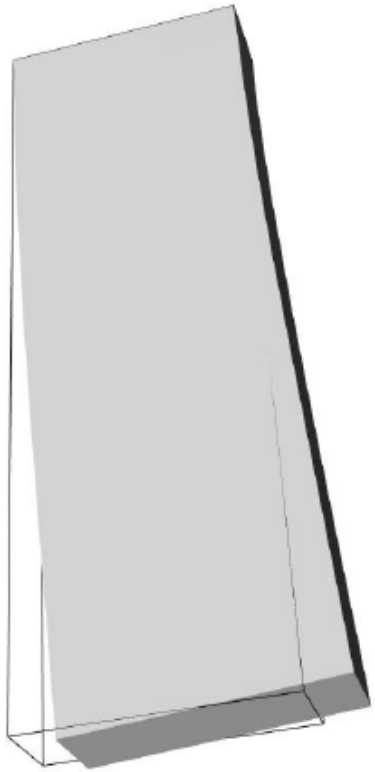






Local vector visualisation : Warping

- **deform geometry** according to the vector field
  - vector fields often associated with motion, or displacement.
  - e.g vibration of a beam.



CAV : Lecture 14

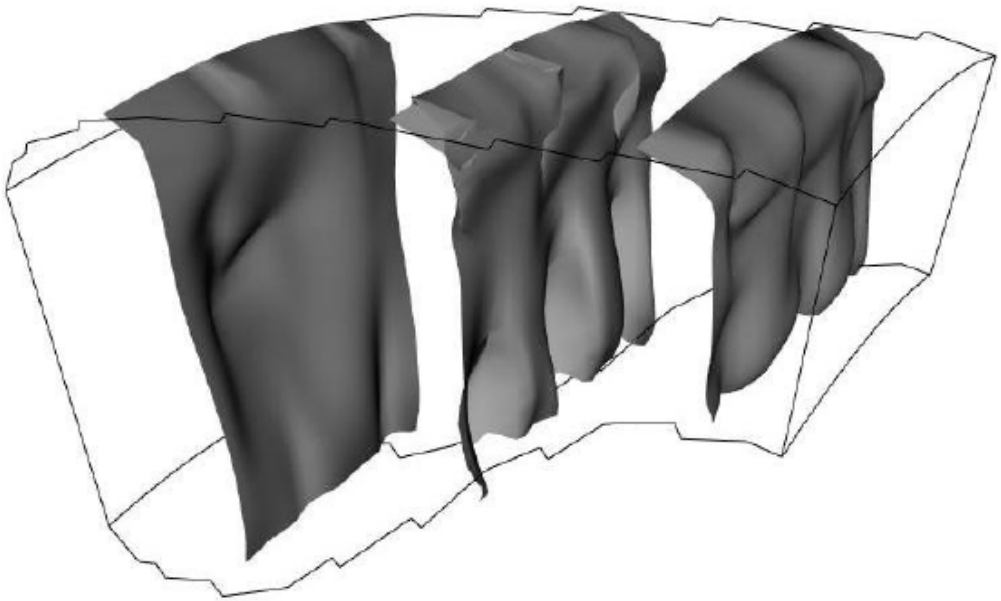
Example : warping

Insert slice planes into the data volume

Displace surface according to flow momentum

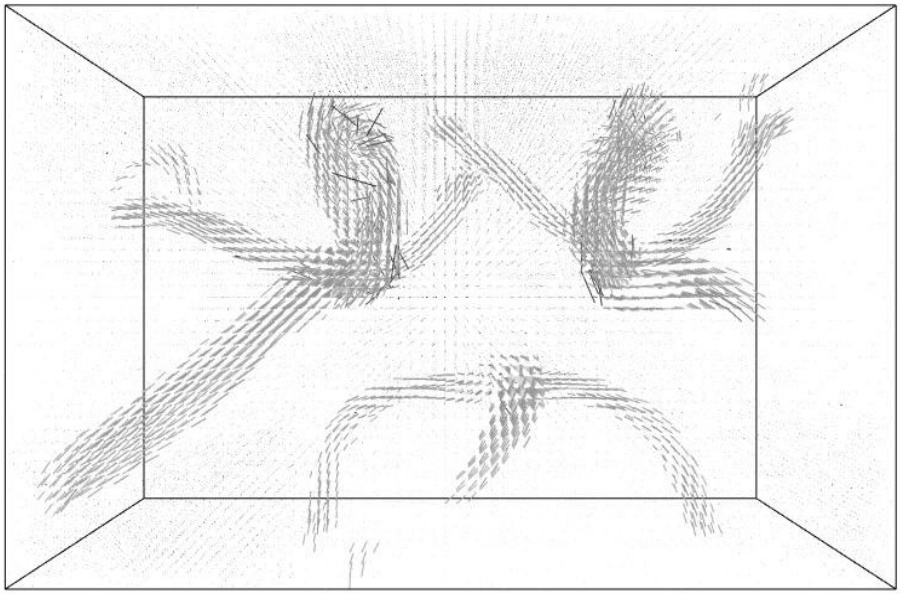
take care with scaling to **avoid excessive geometric distortion**

surfaces may intersect, or even turn inside-out



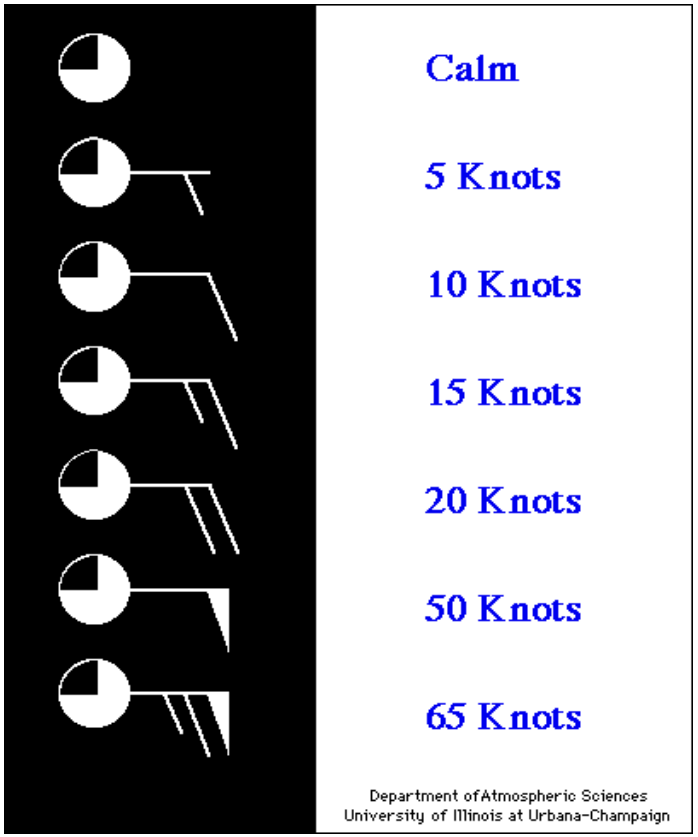
Local vector visualisation : lines

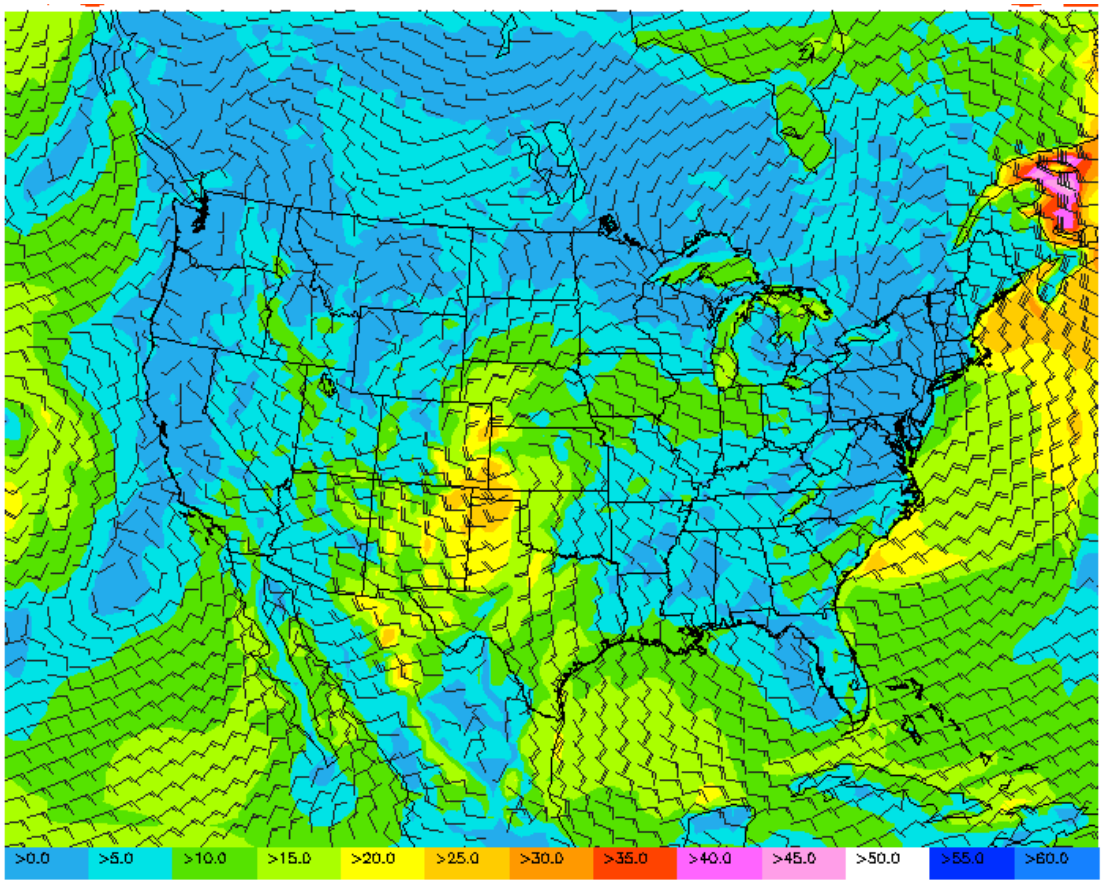
- **Draw line at data point indicating vector direction**
  - scale according to **magnitude**
  - **indicate direction** as vector orientation
- **problems**
  - showing large dynamic range field, e.g. Speed
  - Can result in cluttering
  - Difficult to understand position and orientation in projection to 2D image



- **Option** :
- use colour / barbs to visualise magnitude

Example : Meteorology





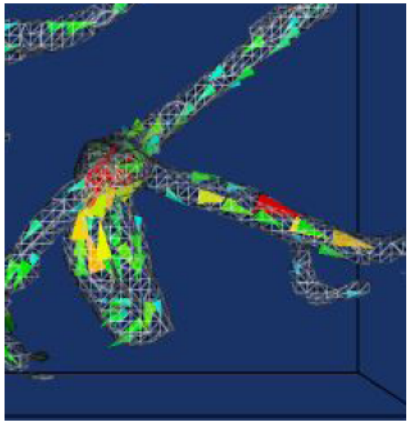
NOAA/FSL

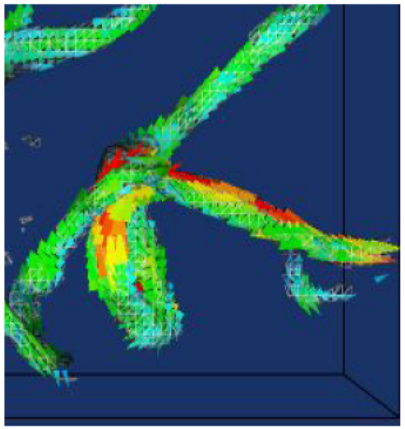
Lines are drawn with constant length, *barbs* indicate wind speed. Also colour mapped scalar field of wind speed.

Local vector visualisation : Glyphs

- **2D or 3D objects**
  - inserted at data point, oriented with vector flow

Need to scale and sample at the appropriate rate otherwise clutter





- e.g. blood flow (reduced data)
  - colourmap shows magnitude in addition to glyph scale

http://www.youtube.com/watch?v=KpURSH_HGB4&feature=related

Overview

Local View

- Warping
- Glyphs

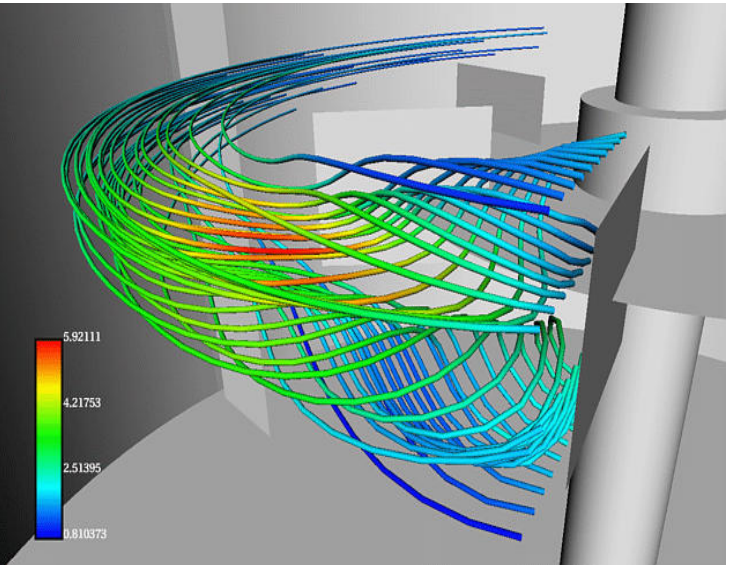
Global View

- **Pathline, Streakline, Streamline**
- Integration
- Stream surface, Stream volume

- Line Integral Convolution

• The Global views of vector fields

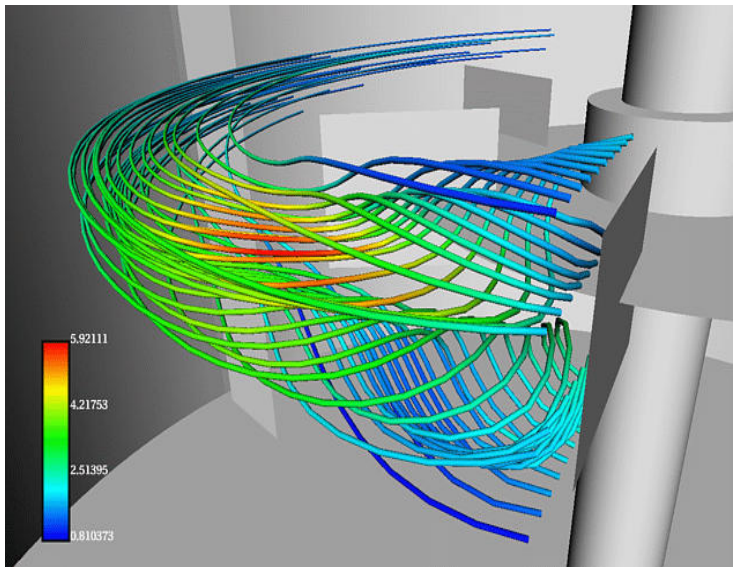
- Visualise where the flow comes from and where it will go
- Visualise flow as the trajectory of a particles transported by the flow
- Pathline, Streamline, Streakline



The Global views of vector fields (2)

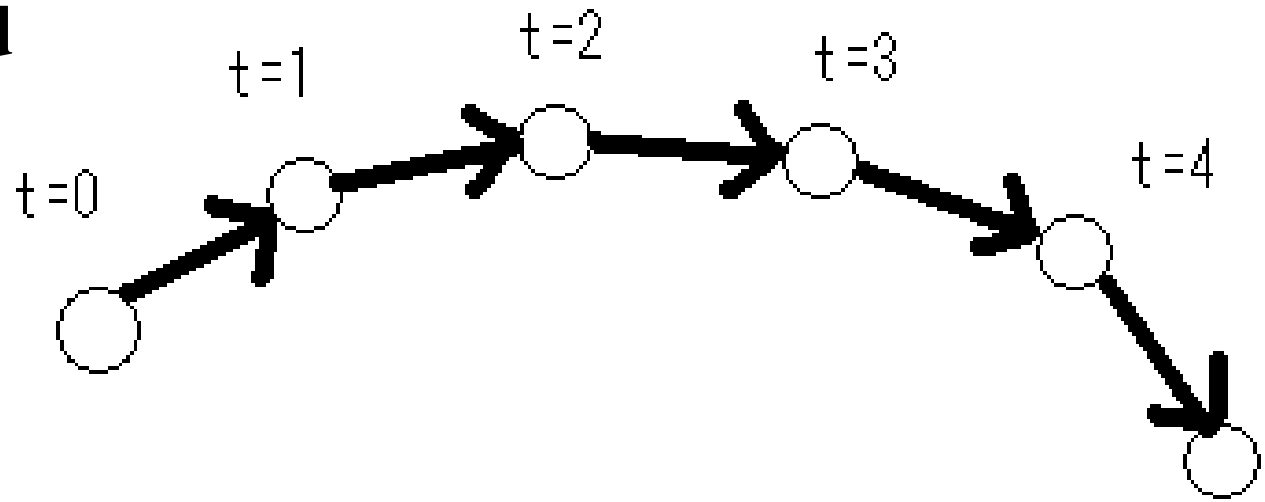
- Shows flow features such as **vortices in flow**
- Only shows **information for intersected points**
- Need to **initialise** in correct place

  - rakes



Pathline

- **Particle trace : the path over time of a massless fluid particle transported by the vector field**
- **The particle's velocity is always determined by the vector field**



$$dx = Vdt$$

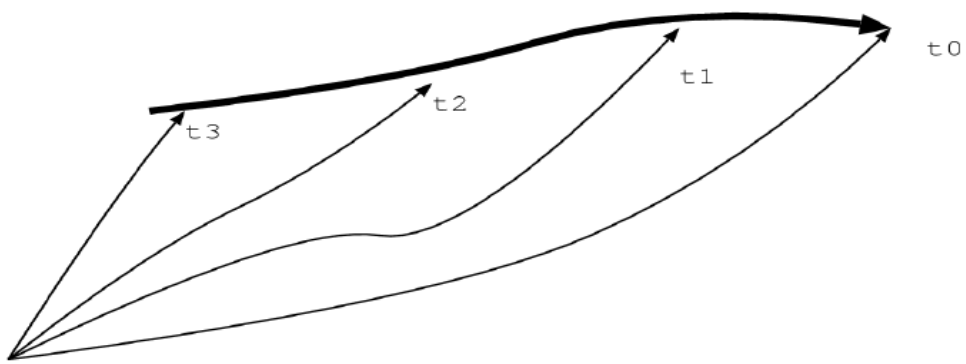
Express in integral form:

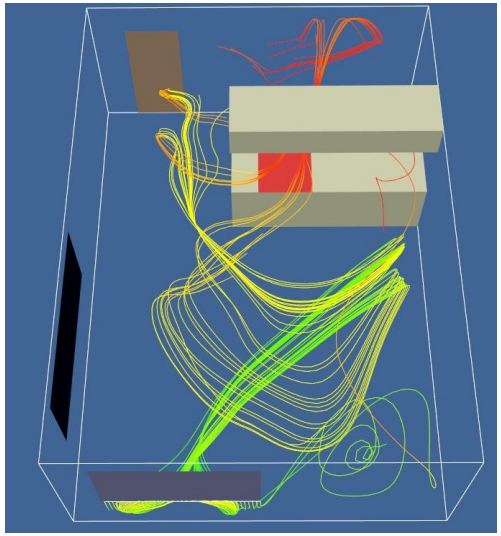
$$x = \int Vdt$$

Solve using numerical integration

Streakline

- The set of points at a particular time that have previously passed through a specific point
  - Path of the particles that were released from a point x_0 at times $t_0 < t < t_f$
  - Dye steadily injected into the fluid at a fixed point extends along a streakline.

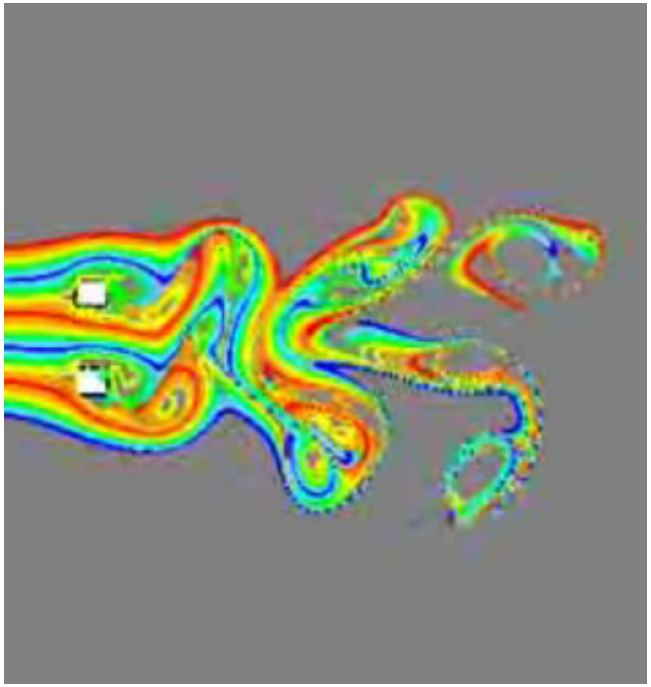




http://www-mdp.eng.cam.ac.uk/web/library/enginfo/aerothermal_dvd_only/aero/fprops/cvanalysis/node8.html

Examples

- Streaklines for 2 square obstruction
- http://www.youtube.com/watch?v=ucetWHDXjAA
- Streaklines exiting from a channel
- http://www.youtube.com/watch?v=tdZ1QafL6MM&feature=channel



Steamline

- **Streamline : integral curves along a curve s satisfying:**

$$s = \int V\, ds, \text{ with } s = s(x,t)$$

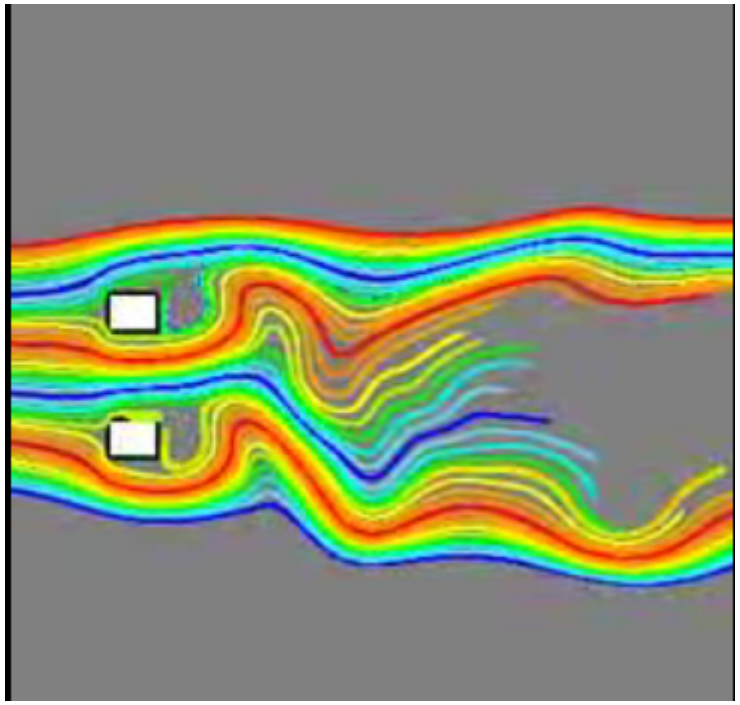
at a fixed time t

  - **Integral in the vector field while keeping the time constant**

Example

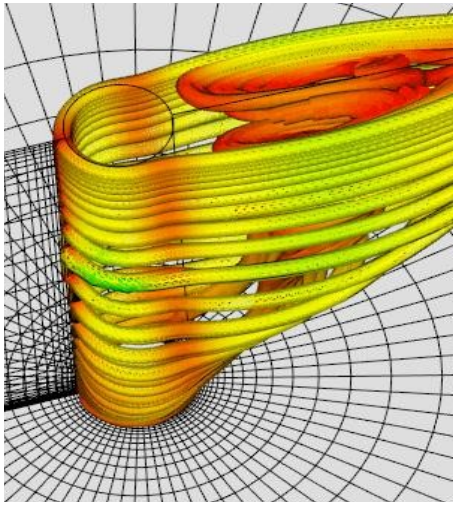
- Streamlines for 2 square obstruction

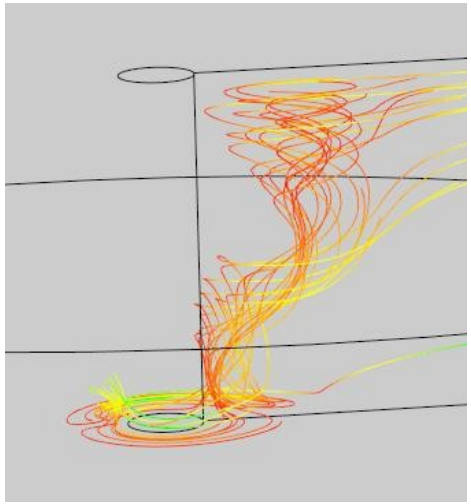
http://www.youtube.com/watch?v=-njBmpInmcU&feature=channel



State of Flow : Steady / Unsteady

- **Steady flow**
  - remains **constant** over time
  - state of **equilibrium** or snapshot
  - **Streamlines==Streaklines**
- **Unsteady flow**
  - **varies** with time
  - streamlines always change the entire shape
  - **Streaklines are more suitable**





Overview

Local View

- Warping
- Glyphs

Global View

- Pathline, Streakline, Streamline
- **Integration**
- Stream surface, Stream volume

- Line Integral Convolution

Integration

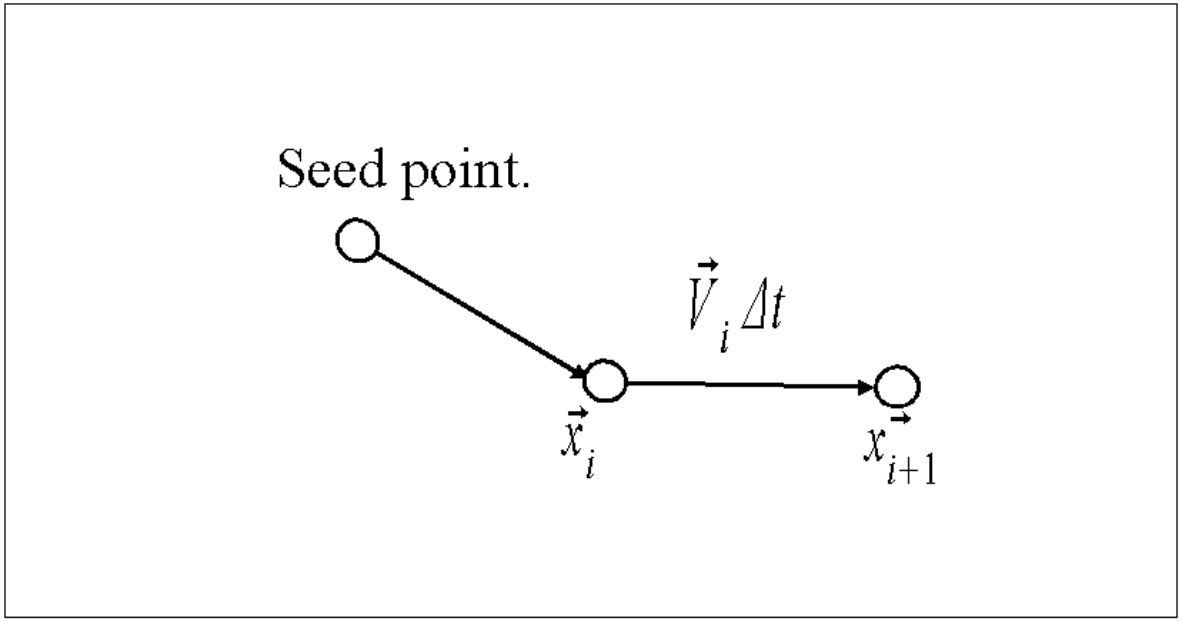
- A scaled, oriented **line is an approximation to a particle's motion in the flow field**
- Need to integrate in order to draw pathlines / streamlines / streaklines
- Integration :
  - Euler method
  - **Implicit Integration**
  - **Verlet** (See notes of Lecture 10)
  - Runge Kutta

Numerical Integration : Euler's Method

$$\vec{x}(t) = \int_t \vec{V}\, dt$$

Euler's method :

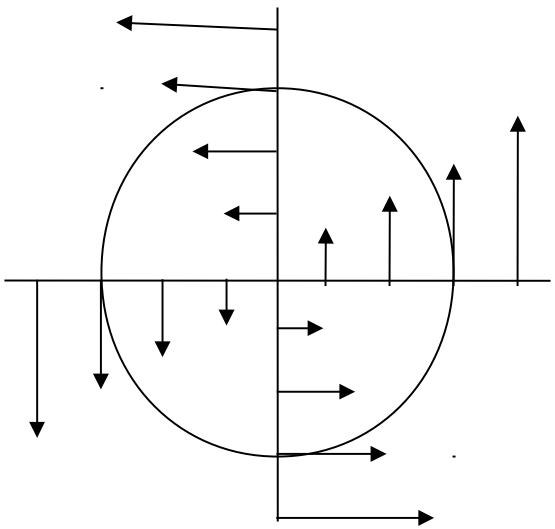
$$\vec{x}_{i+1} = \vec{x}_i + \vec{V}_i \Delta t$$



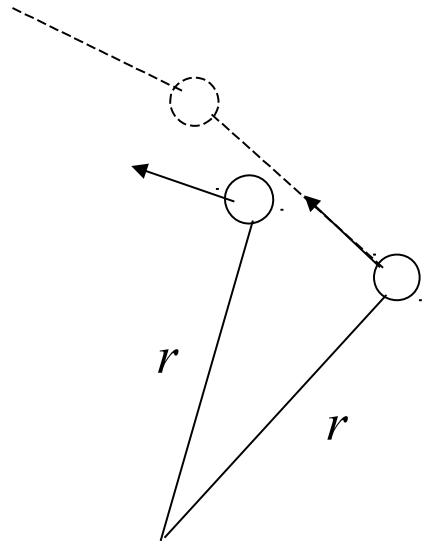
New position $\vec{x}_{i+1}$ = *old position,* $\vec{x}_i$ *plus instantaneous velocity times incremental time step*

Numerical Error is $O(\Delta t^2)$

Problem with Euler's method



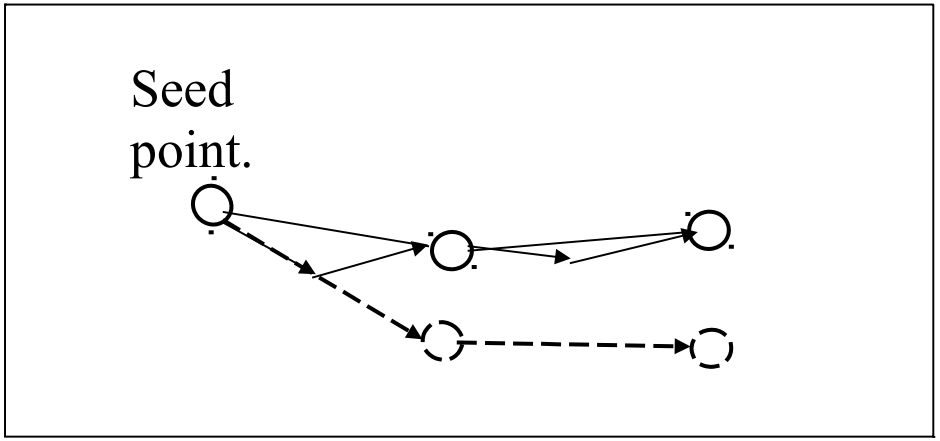
Rotational flow field.



With Euler's method, integrated flow
occurs in a spiral.

- With a rotational flow field – Euler's method wrongly diverges due to error (see also notes of Lecture 10)

Runge-Kutta method, 2nd Order



$$Euler's\ method:\ \vec{x}_{i+1} = \vec{x}_i + \vec{V}_i \Delta t$$

$$V_{i+1} = V_i + a\,\Delta t$$

Runge-Kutta method :

$$\vec{x}_{i+1} = \vec{x}_i + \frac{\Delta t}{2}\left(\vec{V}_i + \vec{V}_{i+1}\right)$$

$\vec{V}_{i+1}$ *is calculated using Euler's method*

Error is $O(\Delta t^3)$

(assuming $0 < \Delta t < 1$)

Runge-Kutta

- Use multiple samples to compute the integral
- Improves accuracy, but more expensive
  - additional function evaluation
- Larger time-step for same error
- 4th Order Runge-Kutta also popular for integration

$$\dot{x} = f(t_i, x_i)$$

$$k_1 = f(t_i, x_i)$$

$$x_{i+1} = x_i + \Delta t \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_2 = f\left(t_i + \frac{1}{2}\Delta t, x_i + \frac{1}{2}\Delta t k_1\right)$$

$$k_3 = f\left(t_i + \frac{1}{2}\Delta t, x_i + \frac{1}{2}\Delta t k_2\right)$$

$$k_4 = f\left(t_i + \frac{1}{2}\Delta t, x_i + \frac{1}{2}\Delta t k_3\right)$$

Overview

Local View

- Warping
- Glyphs

Global View

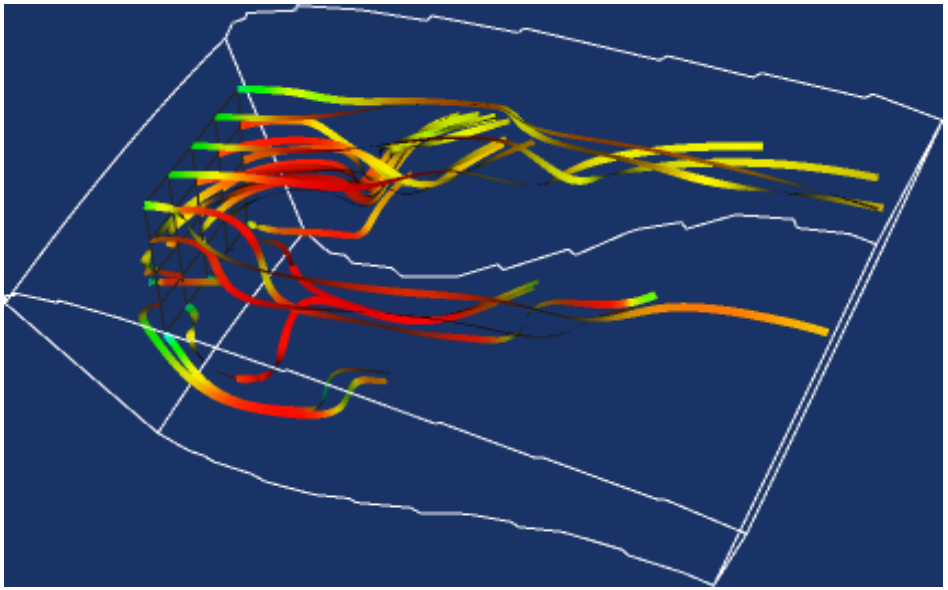
- Pathline, Streakline, Streamline
- Integration
- **Stream surface, Stream volume**

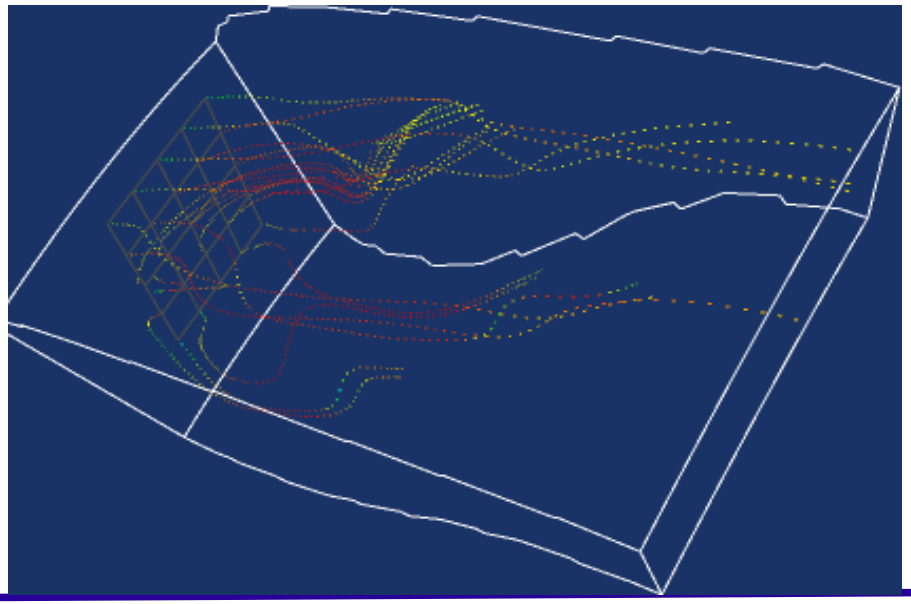
- Line Integral Convolution

Stream Ribbons

Streamribbon : initialise two streamlines together

- **flow rotation**: lines will rotate around each other : can visualize vorticity
- **flow convergence/divergence**: relative distance between lines
- *both not visible with regular separate streamlines*
    - ***Problem if streamlines diverge significantly***

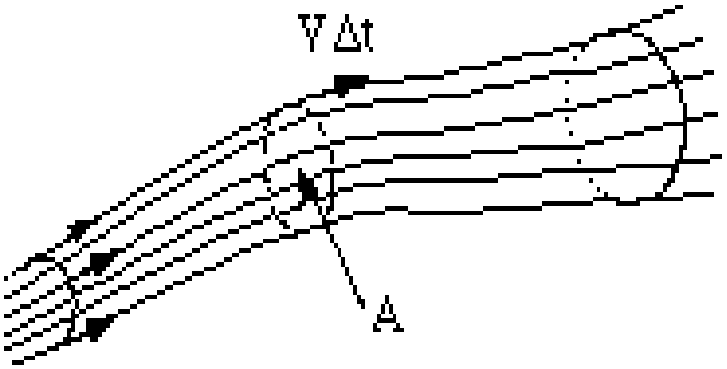


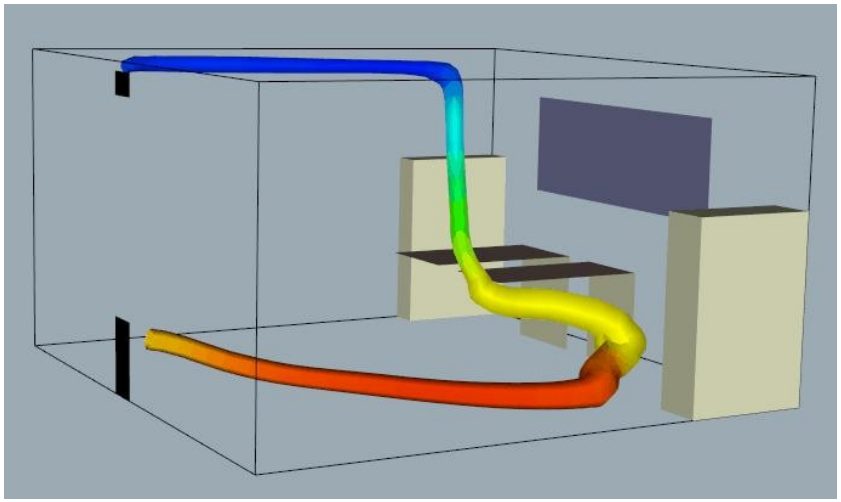


Streamsurface, Streamtube

Initialise multiple streamlines along a base curve or line rake and connect with polygons

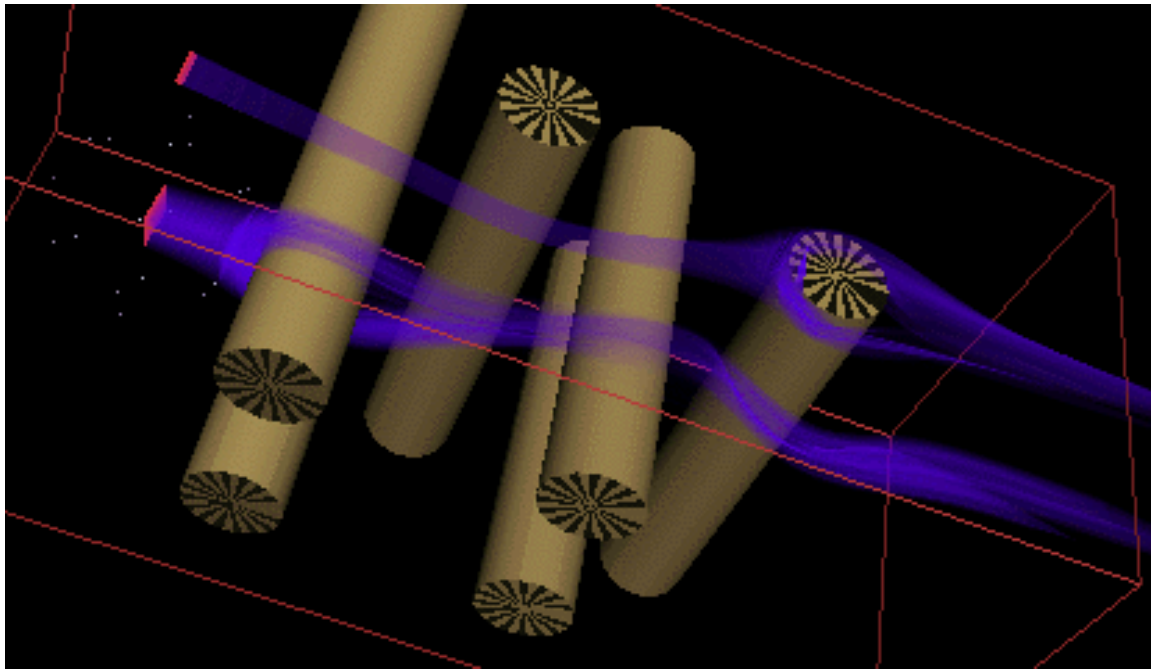
- **Streamtube: A closed stream surface**
- **No substances passing through the tube**
- Properties:
  - surface **orientation at any point on surface tangent to vector field**
  - The amount of substance inside the tube is fixed

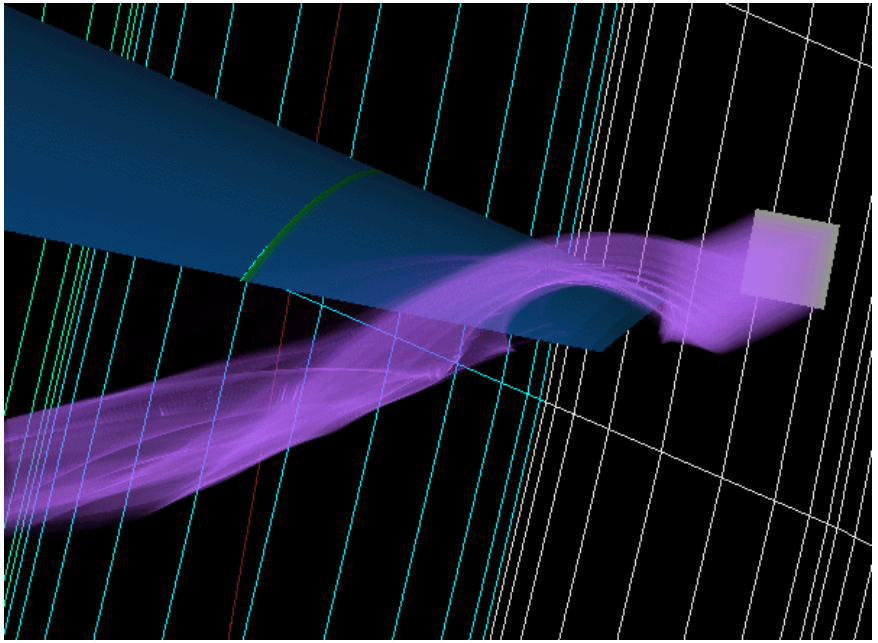




Flow Volumes : simulated smoke

- initialise with a seed polygon – the rake
- calculate streamlines at the vertices.
- split the edges if the points diverge.



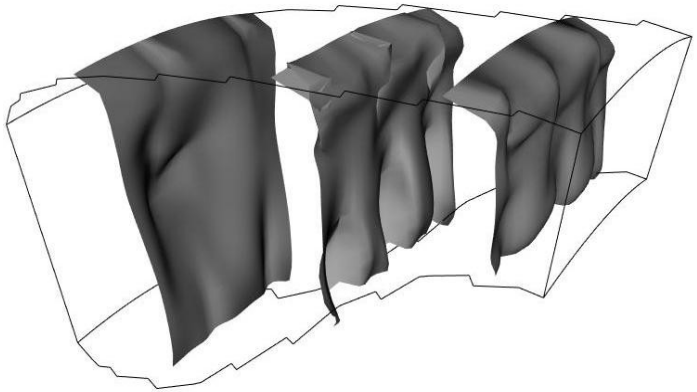


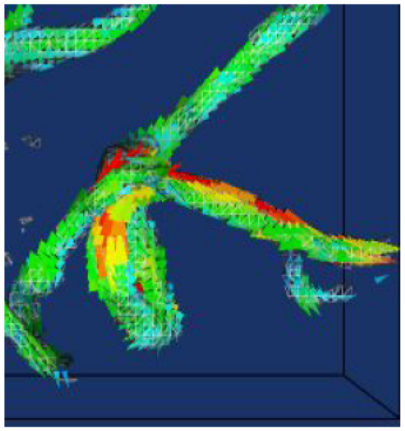
Overview

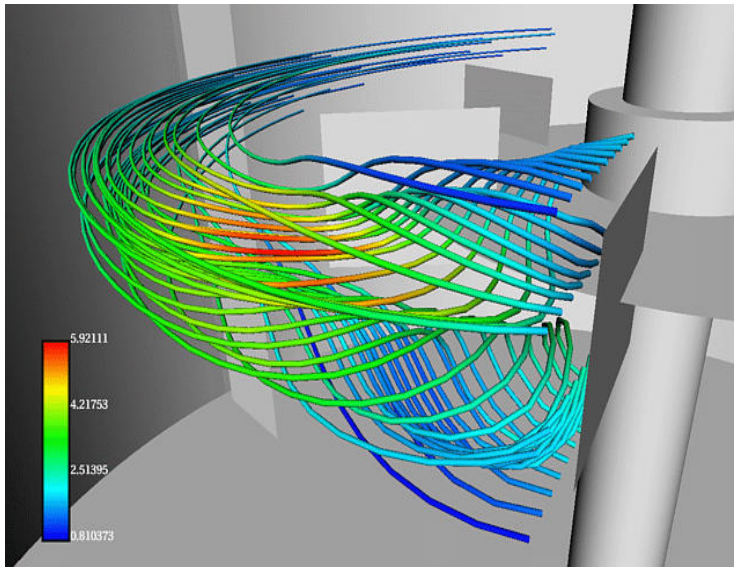
- Local View
  - Warping
  - Glyphs
- Global View
  - Pathline, Streakline, Streamline
  - Integration
  - Stream surface, Stream volume
  - **Line Integral Convolution**

Flow Visualisation Ideals - ?

- **High Density Data** – ability to visualise dense vector fields
- **Effective Space Utilisation** – each output pixel (in rendering) should contain useful information
- **Visually Intuitive** – understandable

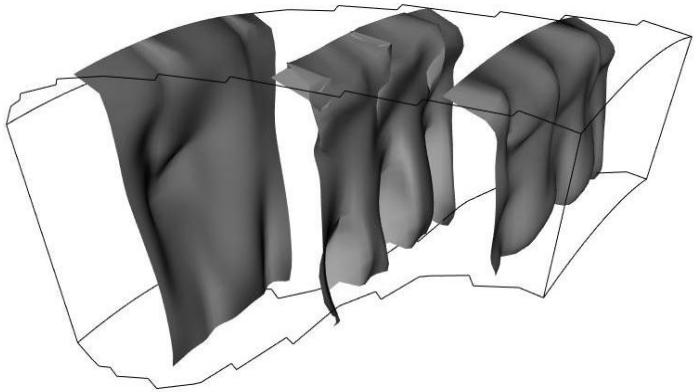


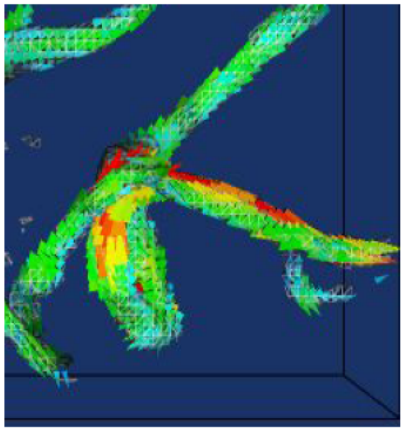


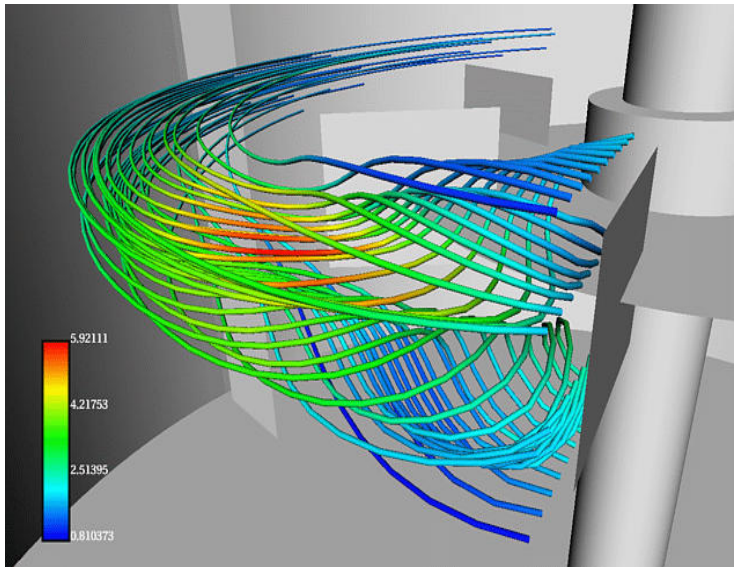


Flow Visualisation Ideals - ? (2)

- **Geometry independent** – not requiring user or algorithmic sampling decisions that can miss data
- **Efficient** – for large data sets, real-time interaction
- **Dimensional Generality** – handle at least 2D & 3D data

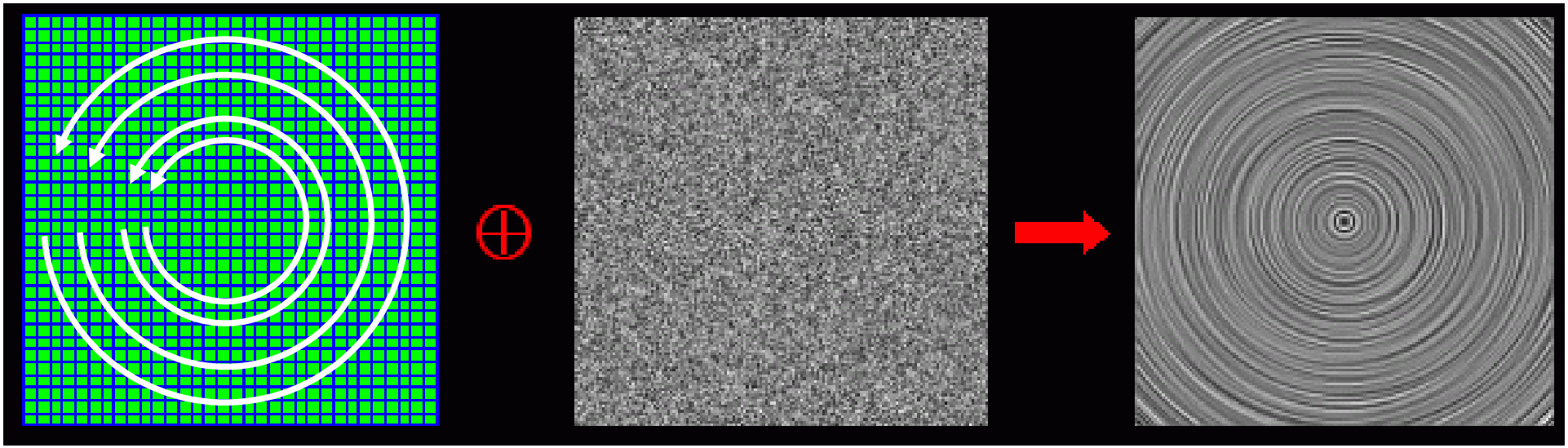






Direct Image Synthesis

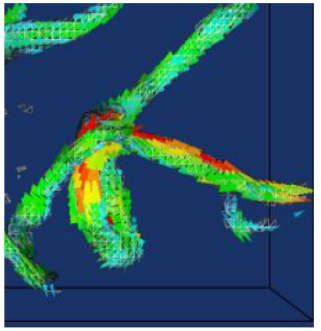
- **Line Integral Convolution (LIC)**
  - **image operator = convolution**



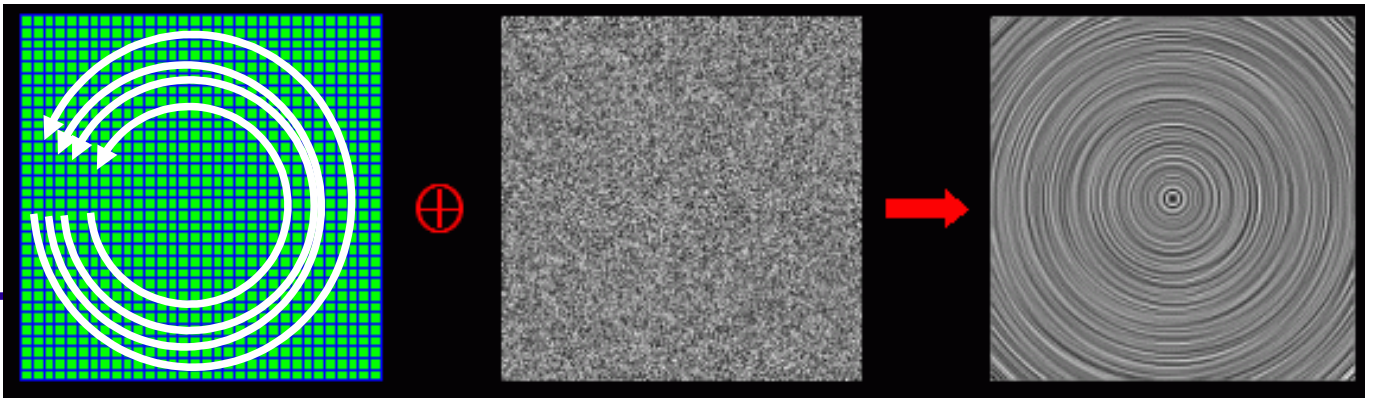
Vector Field White Noise Image LIC Image

Direct Image Synthesis

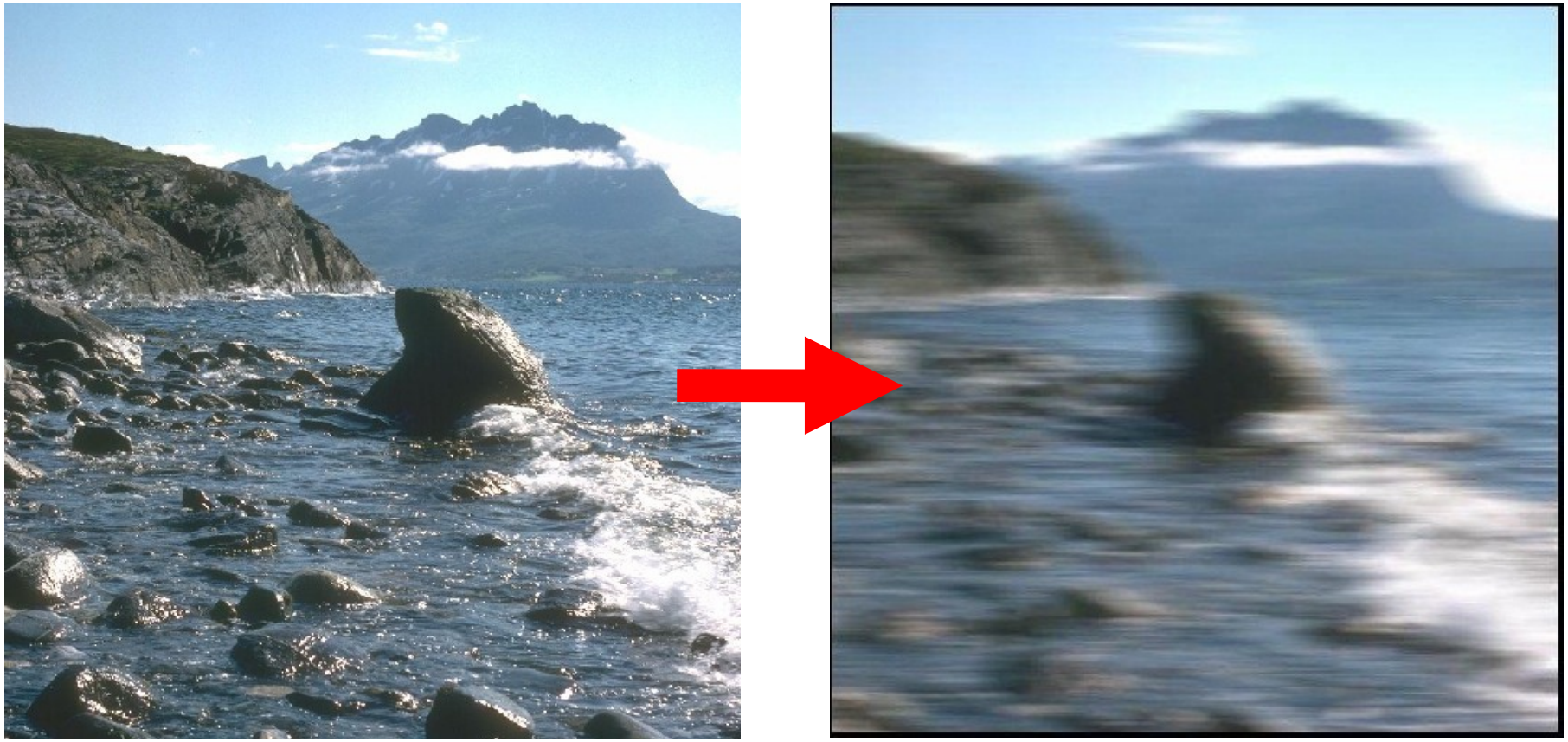
- **Concept** : modify an image directly with reference to the vector flow field
  - **alternative to graphics primitives**
  - ***modified image allows visualisation of flow***
- ***Practice :***
  - ***use image operator to modify image***
  - ***modify operator based on local value of vector field***
  - ***use initial image with no structure***
    - ***e.g. white noise (then modified by operator to create structure)***



Taku

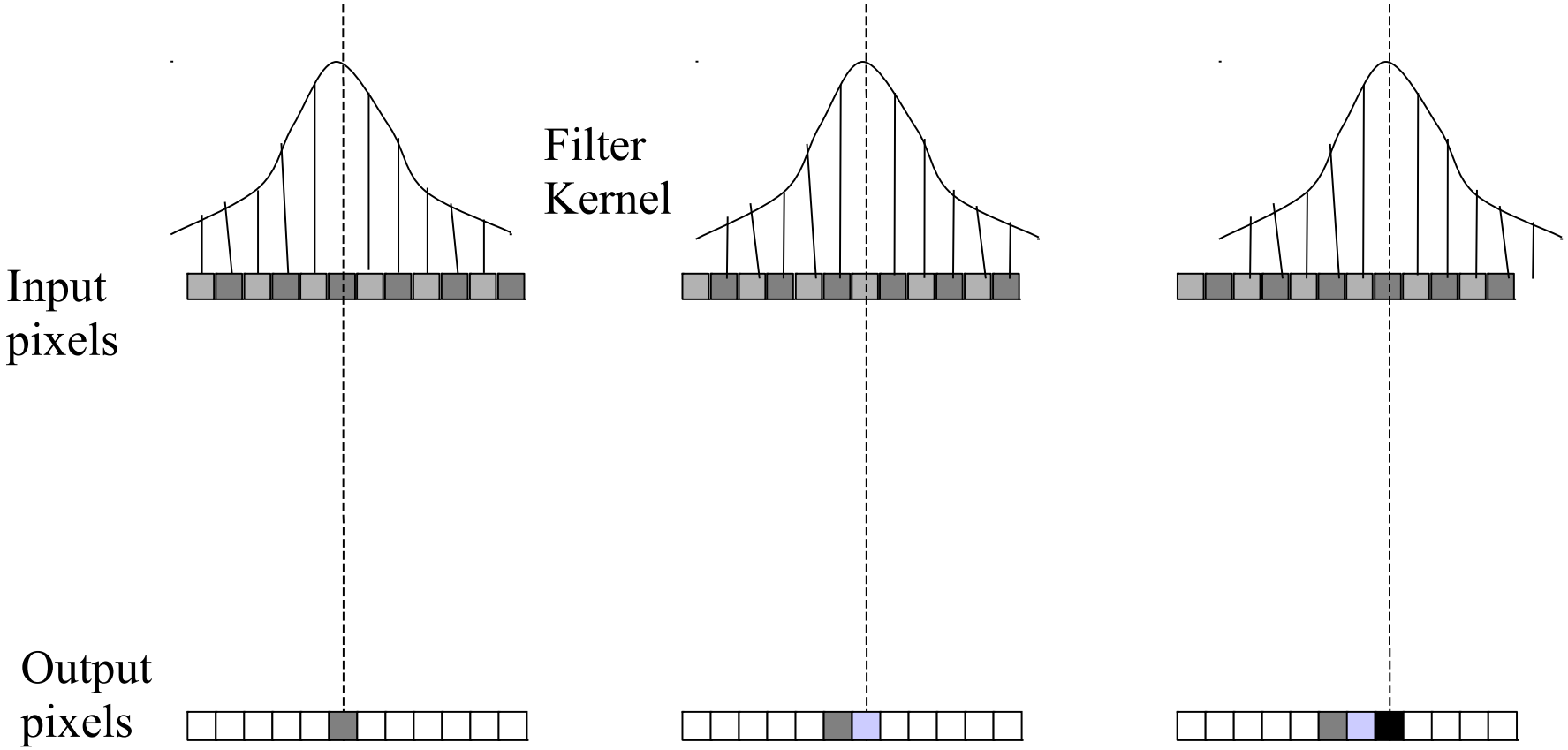


Example : image convolution



- Linear convolution applied to an image
  - linear kernel (causes blurring)

How ? - image convolution



- Each **output pixel p' is computed as a weighted sum of pixel neighbourhood of corresponding input pixel p**
  - weighting / size of neighbourhood defined by kernel filter

LIC : stated formally

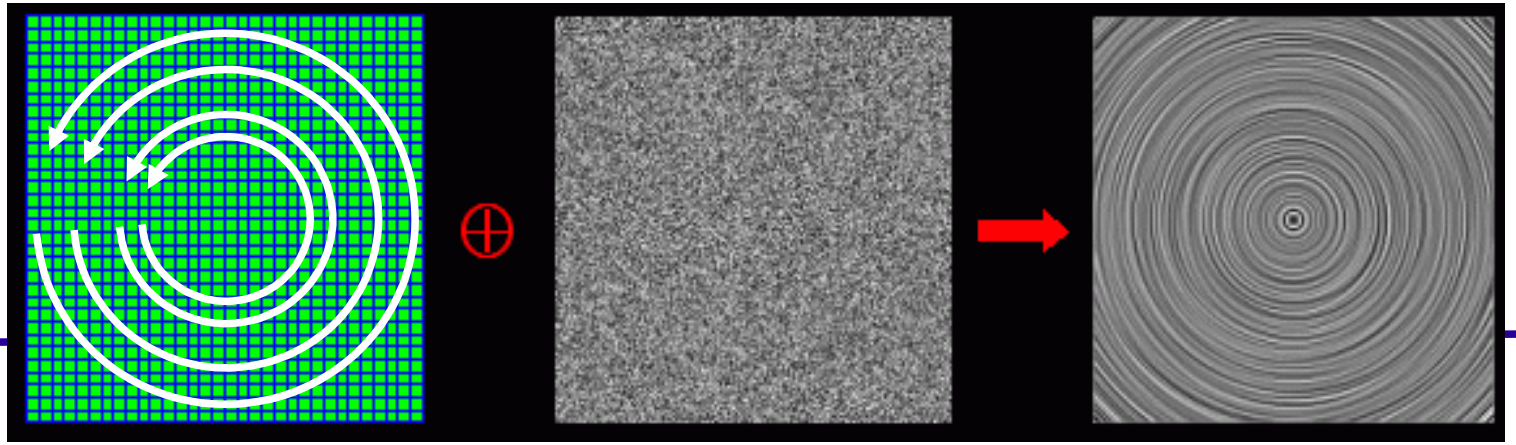
$$F'(p) = \frac{\int_{-L}^{L} F(P(s))\ k(s)\ ds}{\int_{-L}^{L} k(s)\, ds}$$

p is the **image domain**
s is the **parameter along the streamline,** *L* is streamline length
F(p) is the **input image**
F'(p) is the **output LIC image**
P(s) is the **position in the image of a point on the streamline**

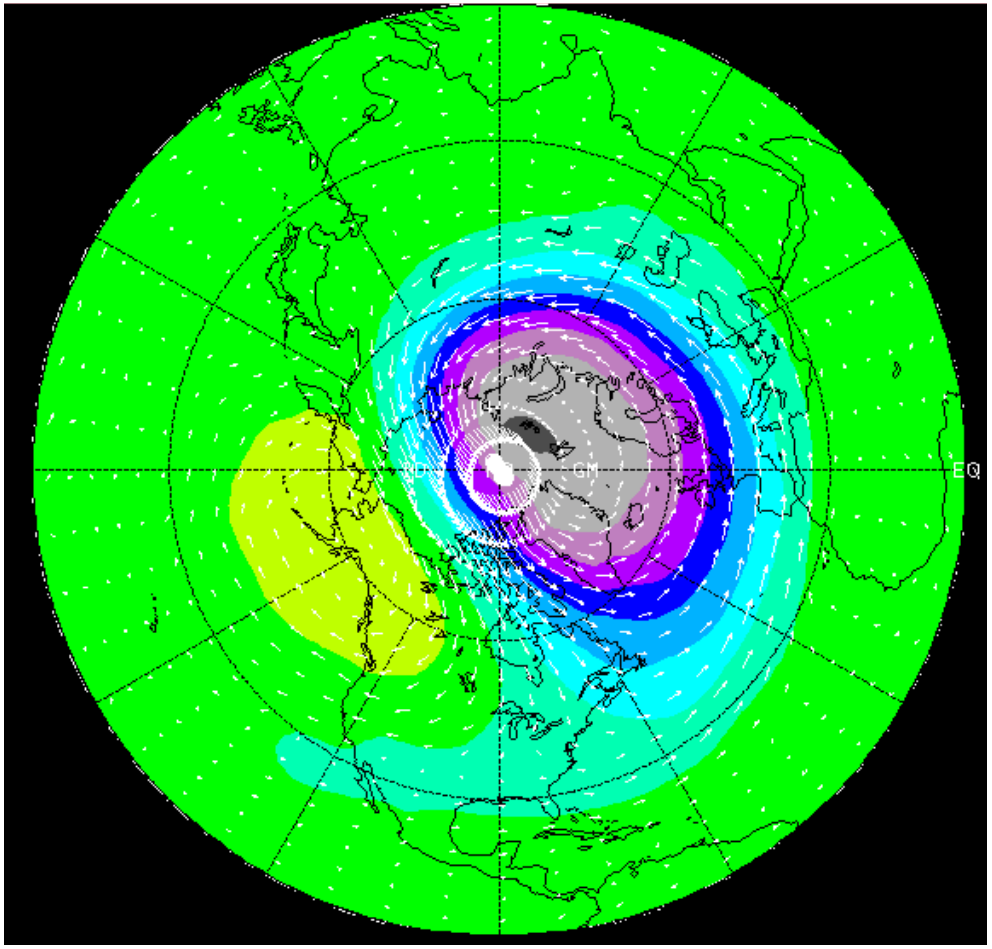
Denominator normalises the output pixel (i.e. maps it back into correct value range to be an output pixel)

Effects of Convolution

- **Convolution 'blurs' the pixels together**
  - amount and direction of blurring defined by kernel
- Perform **convolution in the direction of the vector field**
  - use vector field to define (and modify) convolution kernel
  - produce the effect of **motion blur in direction of vector field**
- For **white noise input** image, convolved output image will exhibit
  - **strong correlation along the vector field streamlines** and
  - **no correlation across the streamlines.**



Example : wind flow using LIC

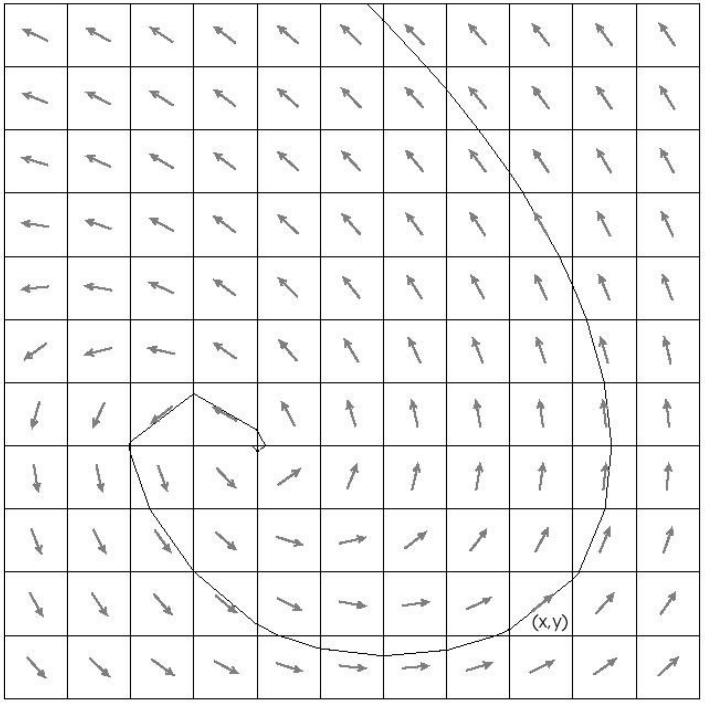


Colour-mapped

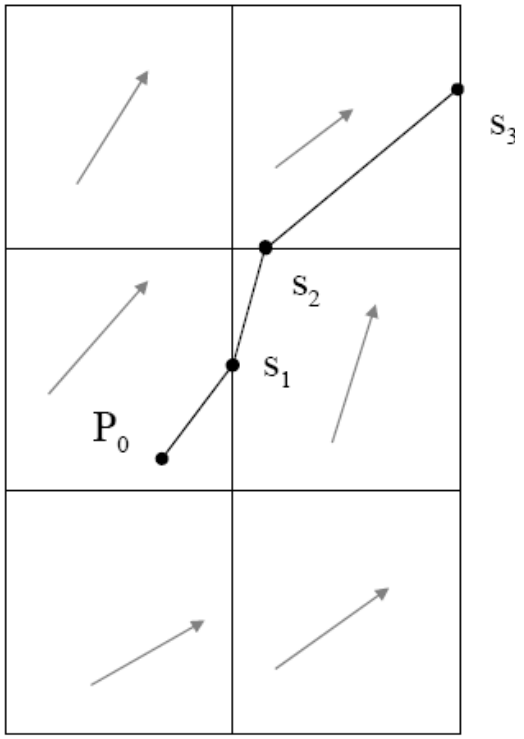
LIC

Data: atmospheric wind data from UK Met. Office Visualisation : G. Watson (UoE)

Streamline Calculation



$$h_i = \int_{s_i}^{s_i+1} k(s)\,ds$$

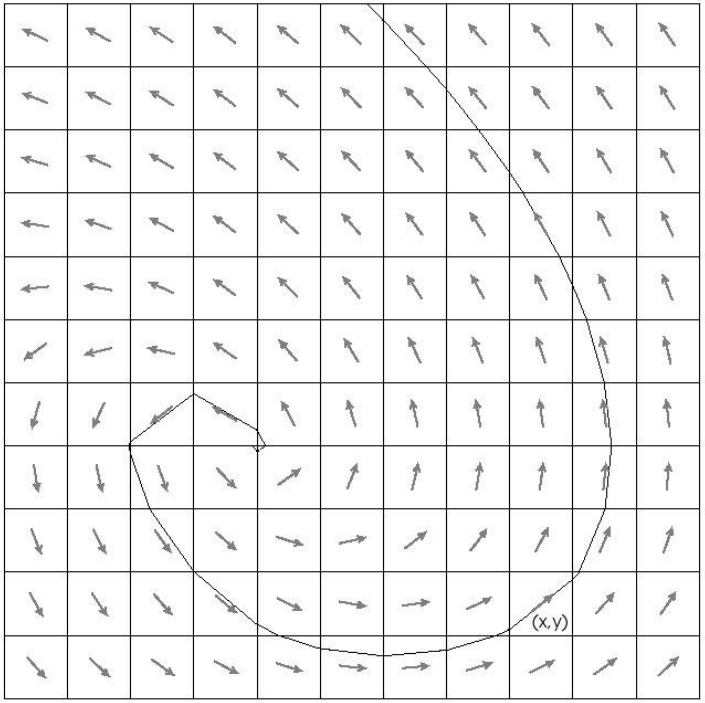


- Constrain the image **pixels to the vector field cells**
  - for each vector field cell, the input white noise image has a corresponding pixel

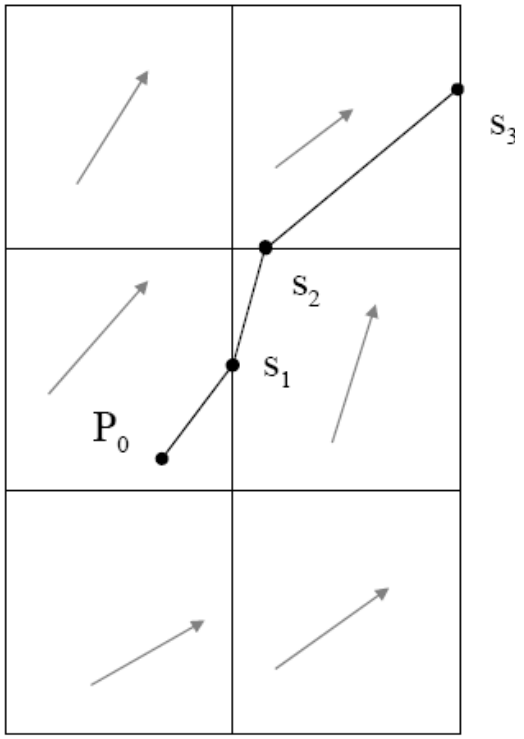
Compute the streamline forwards and backwards in the vector field using **variable-step Euler method.**

- **Compute the parametric endpoints** of each line streamline segment that intersects a cell.

Variable step Euler method



$$h_i = \int_{s_i}^{s_i+1} k(s)\,ds$$



In 2D (lines through cells)

Assume vector is constant across cell.

Calculate closest intersection of cell edge with ray parallel to vector direction using ray-ray intersection.

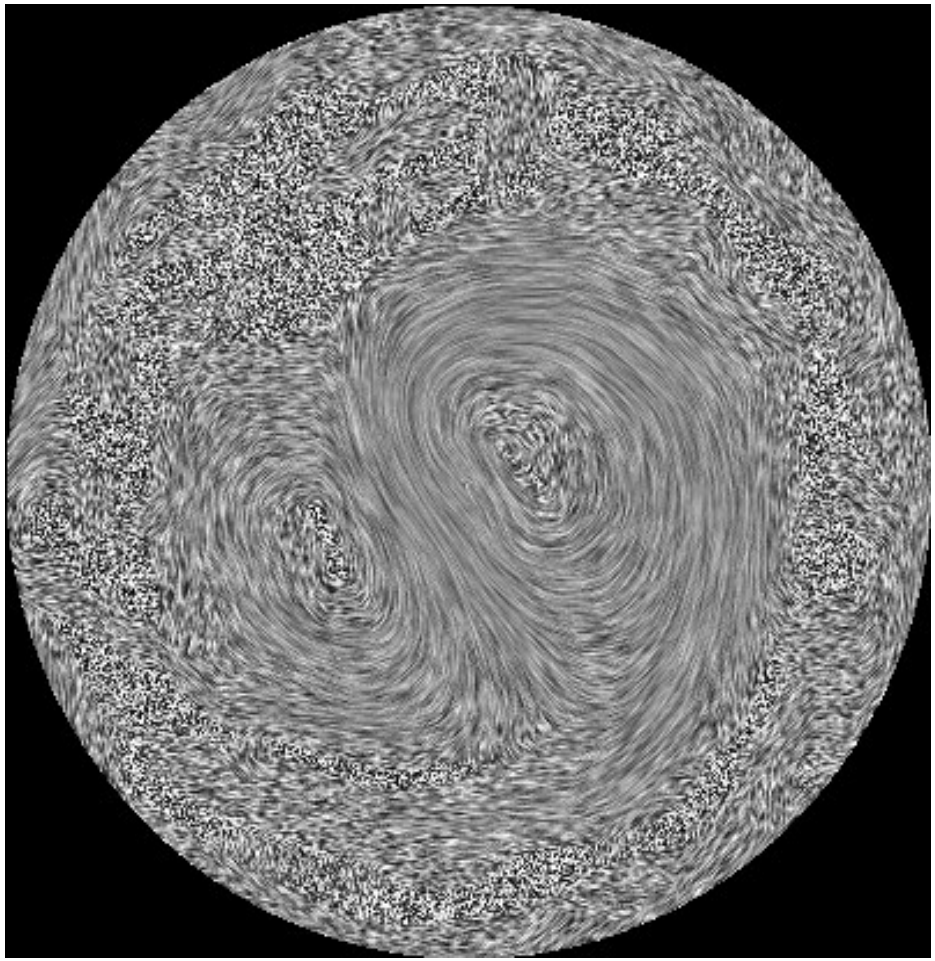
Iterate for next cell position.

What does the LIC show?

- **Constant length convolution kernel**
  - small scale flow features very clear
  - no visualisation of velocity magnitude from vectors
    - can use colour-mapping instead
- **Kernel length proportional to velocity magnitude**
  - large scale flow features are clearer
  - poor visualisation of small scale features
- **Flow direction**
  - **Ambiguity** along line

LIC : 2D results

Variable length Kernel

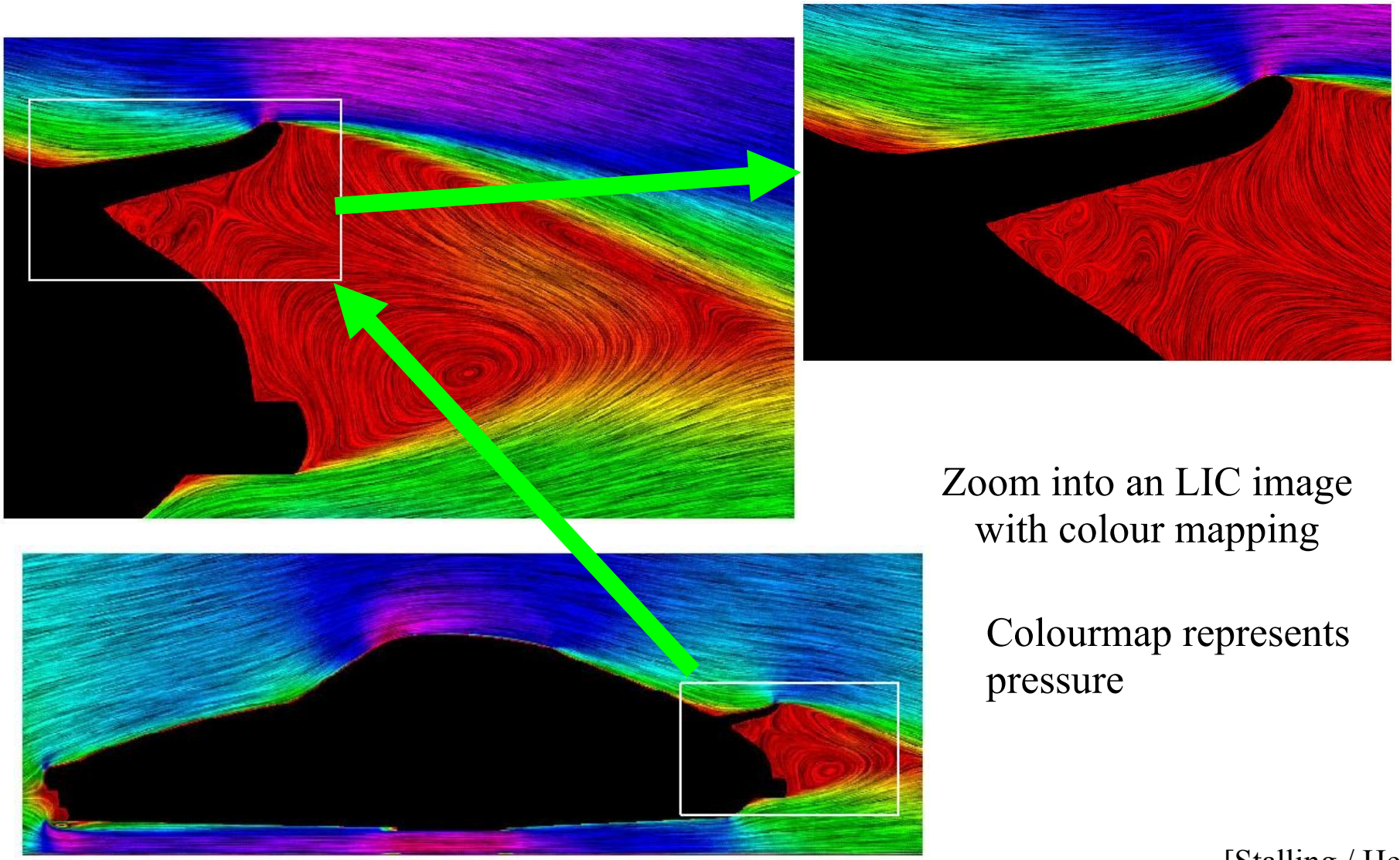


Fixed length Kernel

- Same trade off as with glyph size

images : G. Watson (UoE)

Example : colour-mapped LIC



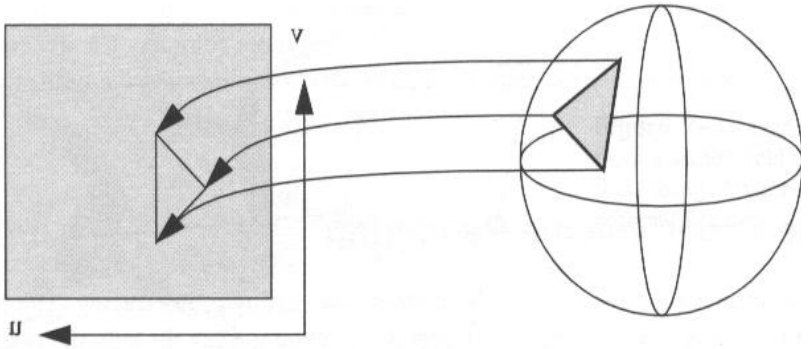
Zoom into an LIC image with colour mapping

Colourmap represents pressure

[Stalling / Hege]

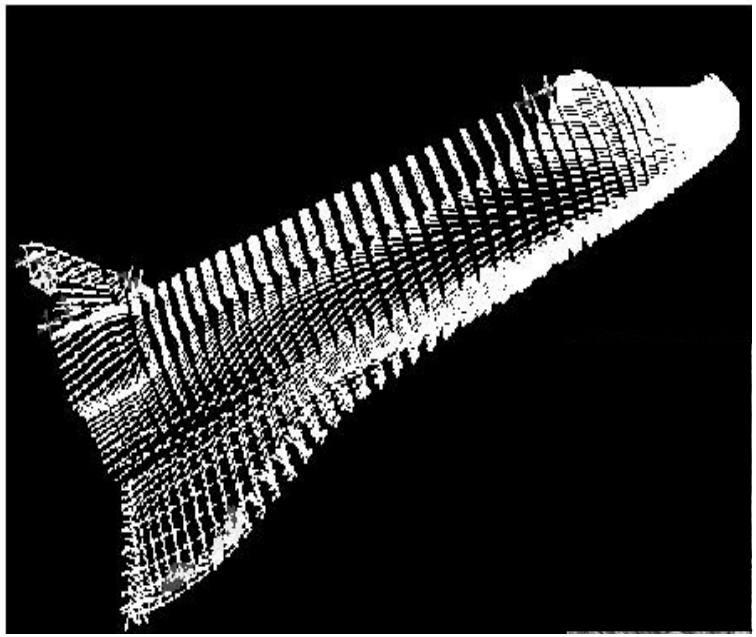
LIC : extension to 3D

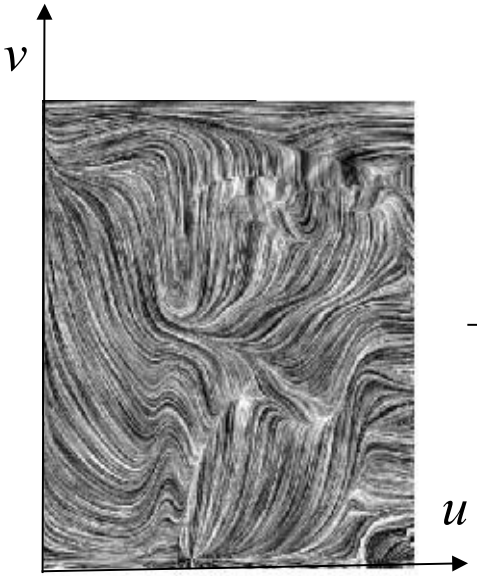
- LIC on *uv* parametric surfaces



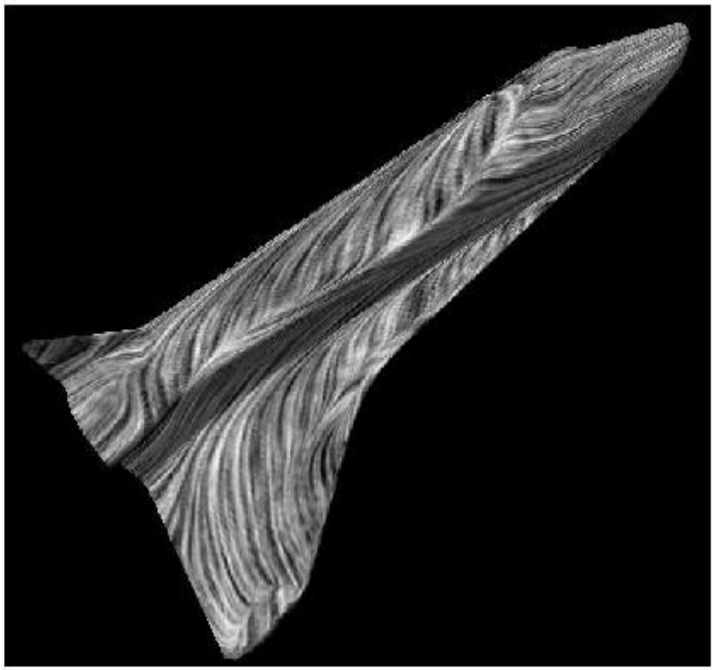
uv coordinates are the same as used in texture coordinates

Vector field on surface to be visualised





Perform LIC in *uv* space



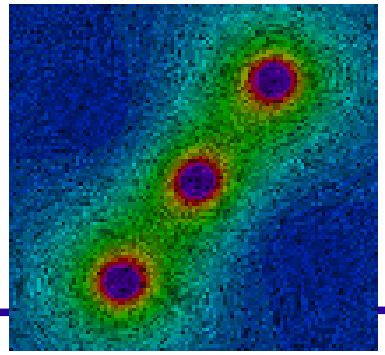
LIC : steady / unsteady flow

- **LIC : Steady Flow** only (i.e. **streamlines**)
  - animate : shift phase of a periodic convolution kernel

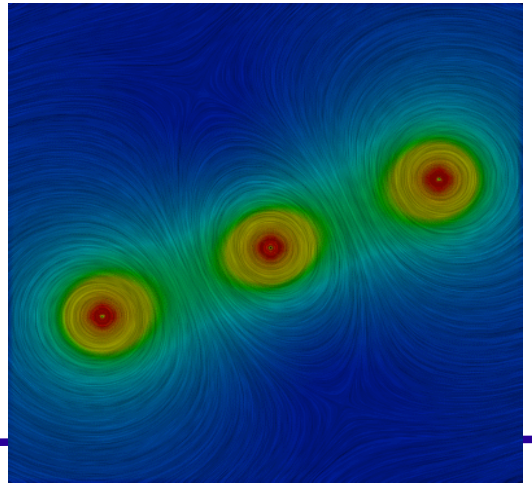
[Cabral/Leedom '93]

- **UFLIC : Unsteady Flow**
  - **Streaklines** are calculated rather than streamlines
  - convolution takes time into account.

[Kao/Shen '97]



[Zhanping Liu]



Summary

- **Local and Global View** of Vector Fields
  - **Local View**
    - **Glyphs, warping, animation**
  - **Global View**
    - visualising **transport**
    - requires **numerical integration**
      - Euler's method
      - Runge-Kutta

  Stream **ribbons and surfaces**:
- **LIC**
  - **steady flow visualisation using direct image synthessis**
  - **convolution with kernel function**
  - **3D & unsteady flow extensions**