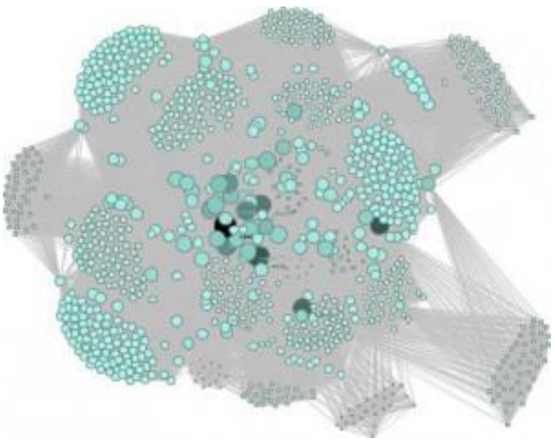
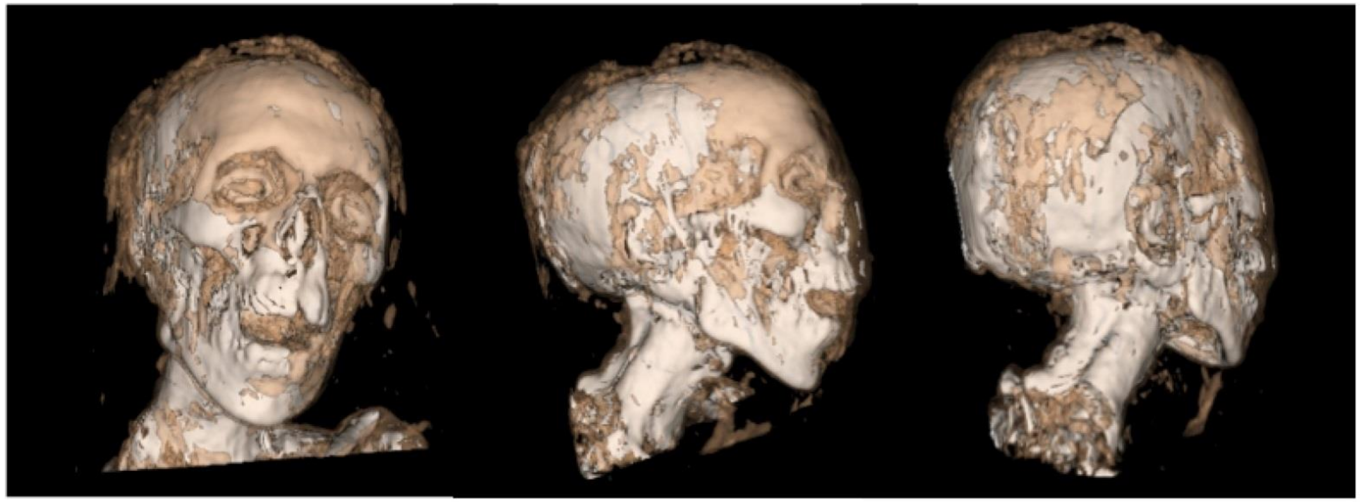
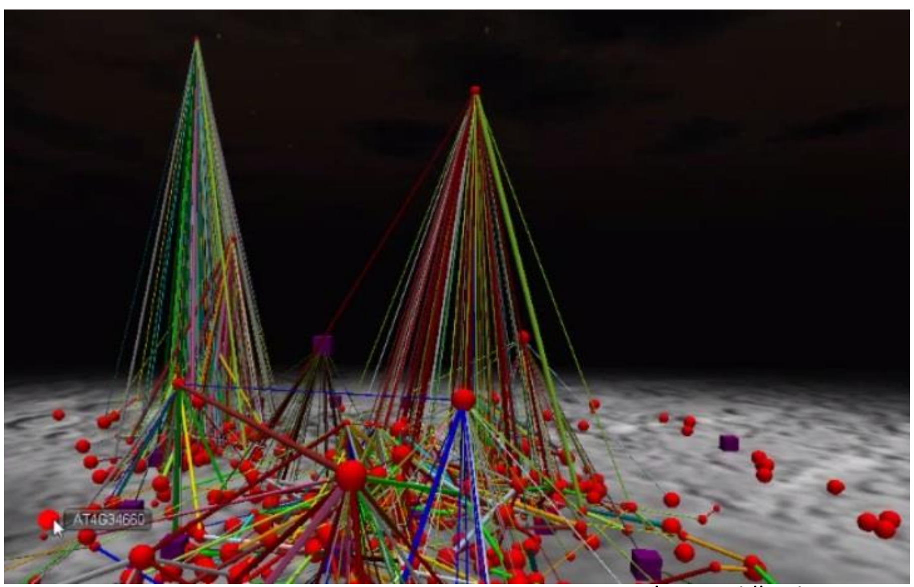


blogs.library.duke.edu

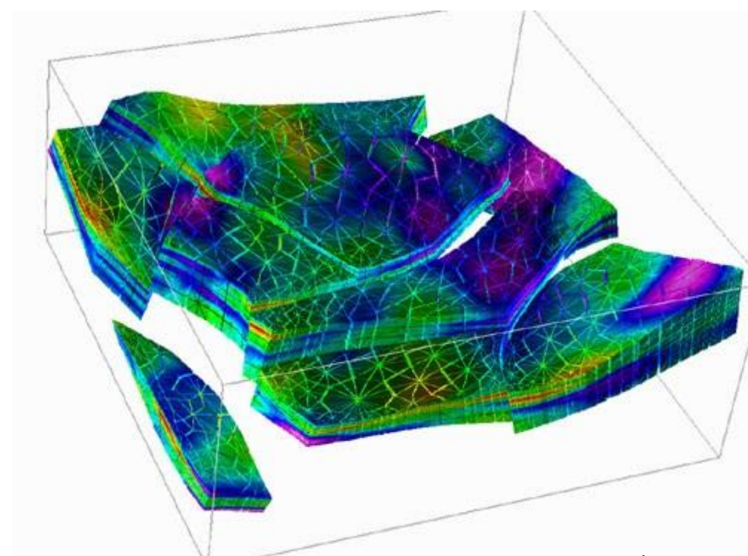


visthis.blogspot.com

4. Spatial data visualization



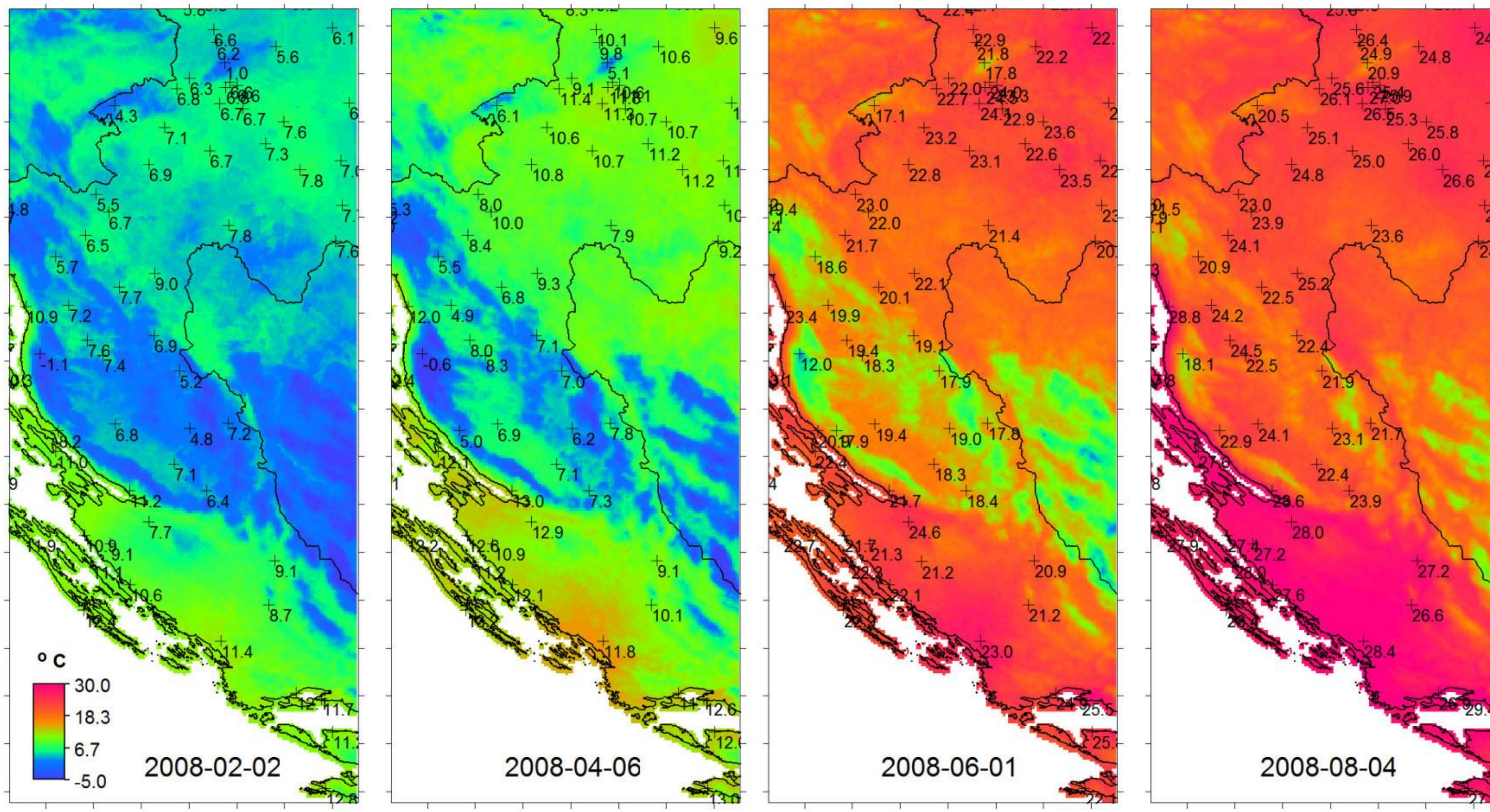
www.hypergridbusiness.com



www.automation-drive.com

Spatial data visualization

- Input data contains spatial or spatio-temporal attributes



Real world vs. screen

- In real world, we are not limited by 2D space, discrete representation, low resolution
- On screen:
  - Exploring data in different scales
  - Dynamic changes of contrast, lighting, resolution
  - Interactive exploration of space inaccessible in real world
  - Interactive adding and removing parts of the data

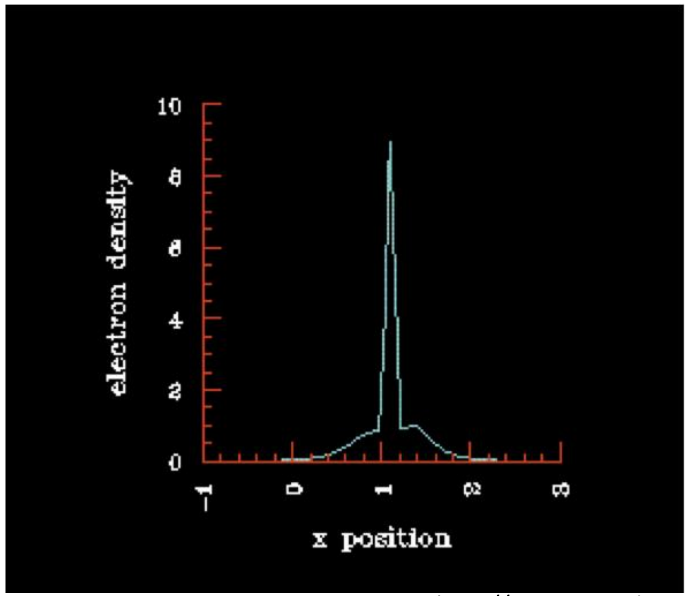
Mapping of attributes

- Phase no. 1:
  - Mapping of spatial attributes of data to spatial attributes of the screen (transformation)

- Phase no. 2:
  - Mapping of the remaining attributes – color, texture, size, shape of graphical entities, …

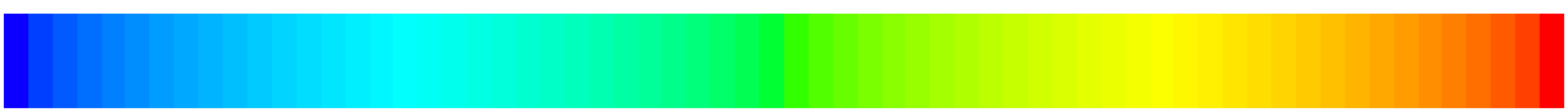
1D data

- Sequence of 1D data with one variable
  - Graph



http://www.opendx.org

  - Color bar



1D graph drawing - algorithm

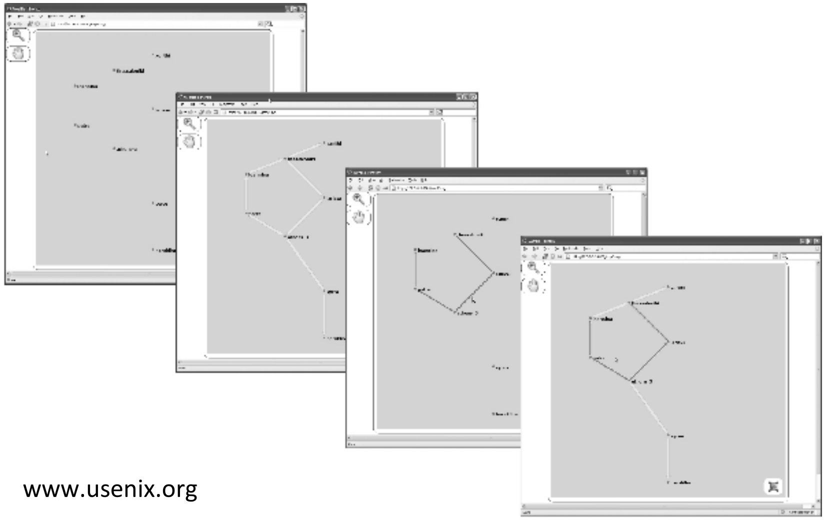
- Input:
  - $data_{min}$, $data_{max}$ – minimal and maximal value in data
  - $data_{count}$ – number of data items to be drawn
  - Screen for data visualization – rectangle (x_{min}, y_{min}, x_{max}, y_{max})

1D graph drawing - algorithm

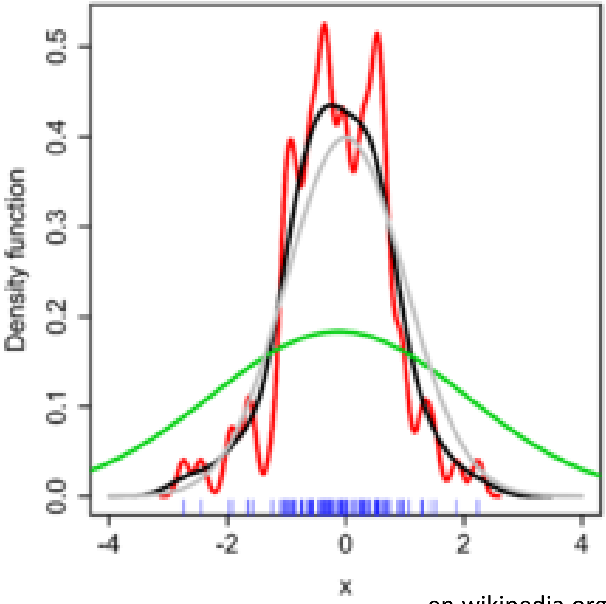
```
DRAW-LINE-GRAPH(data, dataCount, xMin, xMax, yMin, yMax)
    dataMin <- computeMin(data, dataCount)
    dataMax <- computeMax(data, dataCount)
    xFrom <- xMin
    yFrom <- worldToScreenY(data[0], dataMin, datamax, yMin, yMax)
    for i<- 1 to dataCount
          do xTo <- worldToScreenX(i, dataCount, xMin, xMax)
                yTo <- worldToScreenY(data[i], dataMin, dataMax,
                       yMin, yMax)
                drawLine(xFrom, yFrom, xTo, yTo)
                xFrom <- xTo
                yFrom <- yTo

worldToScreenX(index, dataCount, xMin, xMax)
         return (xMin + index * (xMax - xMin)/dataCount)

worldToScreenY(value, dataMin, dataMax, yMin, yMax)
         return (yMin + (value - dataMin) * (yMax - yMin)/(dataMax /
                dataMin))
```

1D multivariate data

- More variables or more values for one data input
- Extension of the previous technique
  - Juxtapositioning
  - Superimpositioning



www.usenix.org



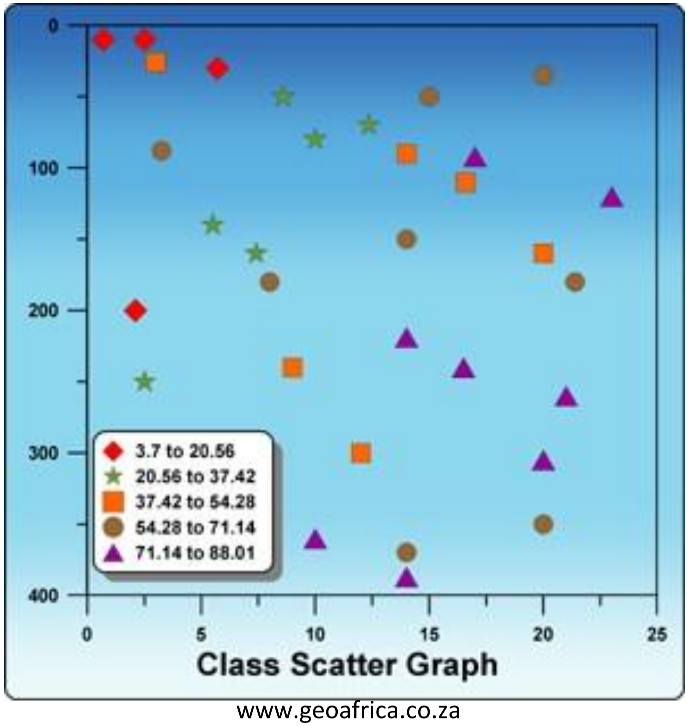
en.wikipedia.org

2D data

- Two spatial dimensions – mapping of spatial data attributes to screen space attributes
- Typical visualizations of 2D data:
  - Scatterplot
  - Map
  - Image
  - Rubber sheet
  - Cityscape
  - Contours, isobars

Scatterplot

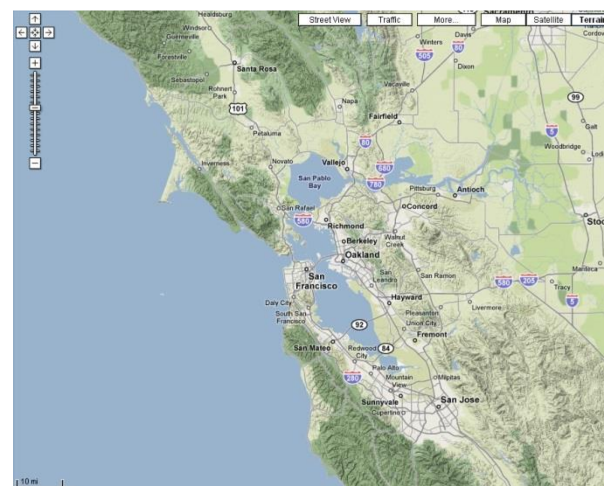
- Each data item influences color, shape, and size of the selected glyph
- No interpolation



www.geoafrica.co.za

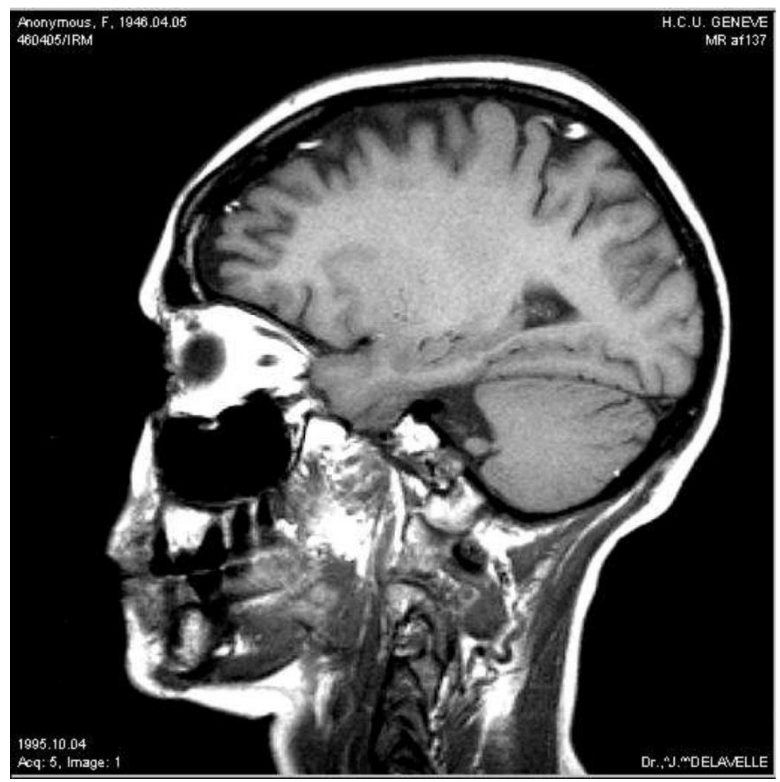
Map

- Linear objects – continuous line segments (rivers, roads)
- Planar objects – closed polygons with color, texture, … (lakes, countries)
- Point objects – specific symbols (school, church)
- Labels



Image

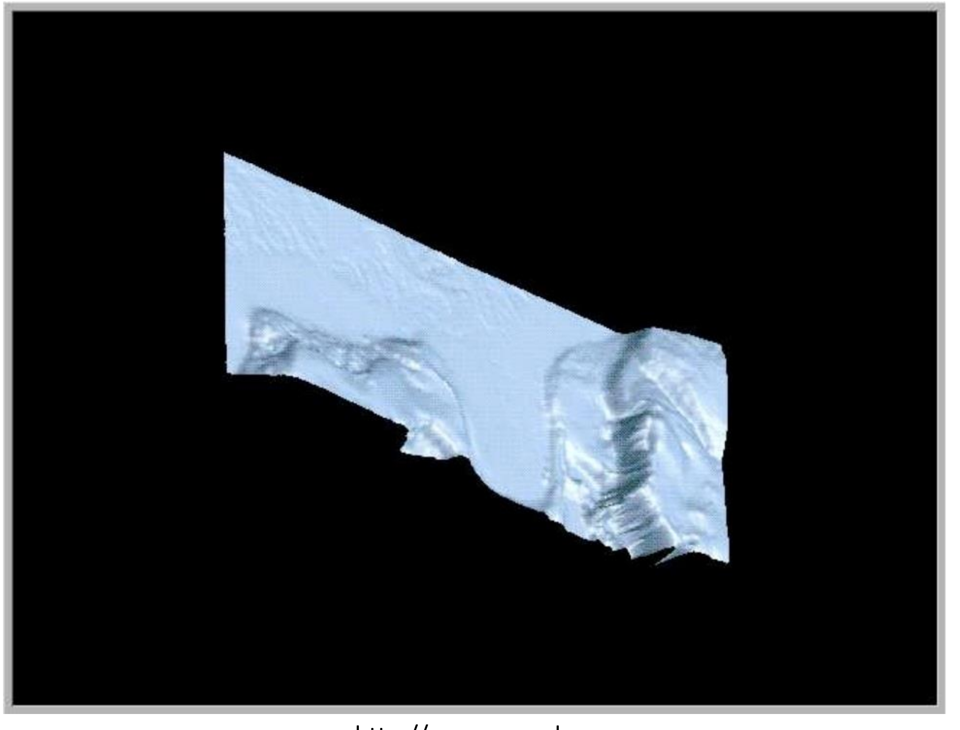
- Data value mapped onto color in given position, color between pixels has to be interpolated



Interactive Data Visualization - Foundations, Techniques and Applications. Matthew Ward

Rubber sheet

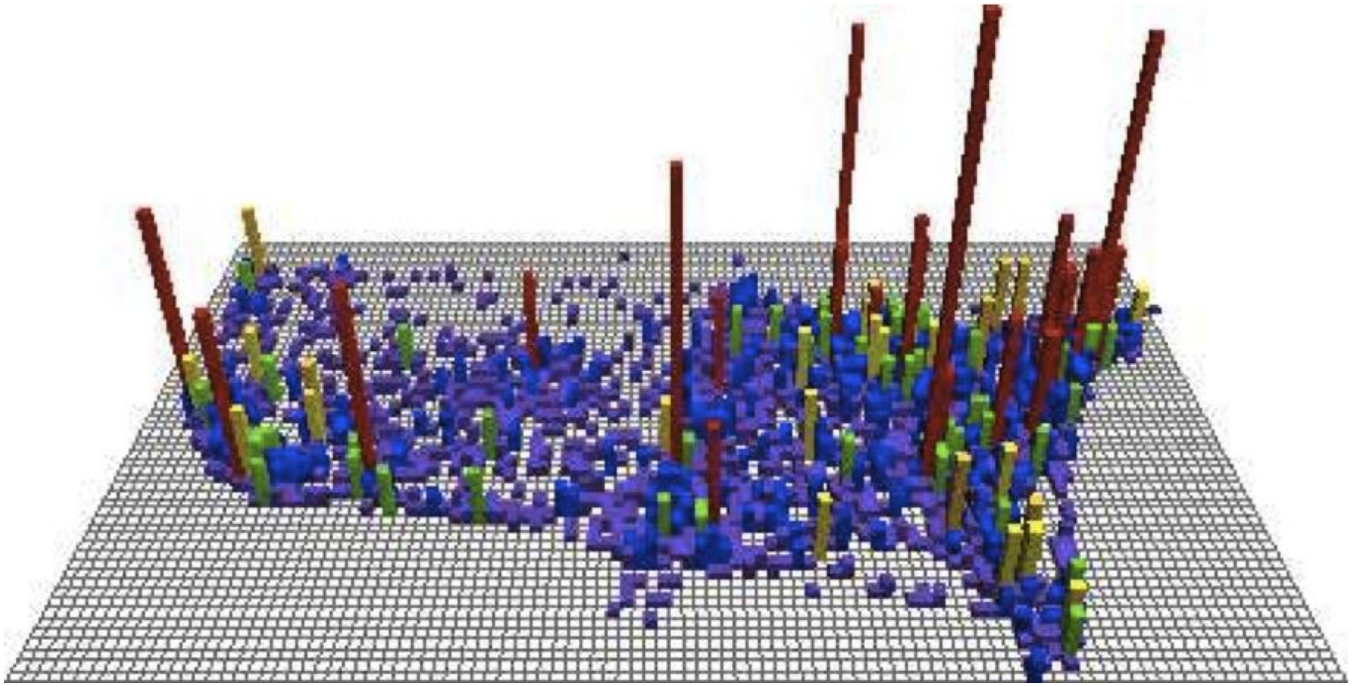
- Data mapped onto the point height in 3D space + triangulation of these points



http://www.opendx.org

Cityscape

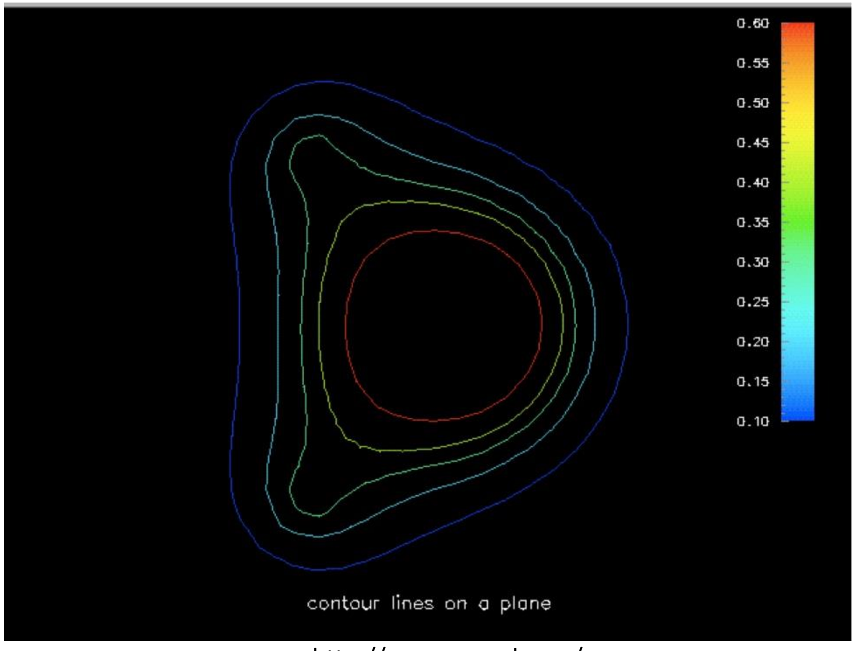
- Drawing 3D blocks in plane, data mapped onto their attributes (height, color, …)



Interactive Data Visualization - Foundations, Techniques and Applications. Matthew Ward

Contours, isobars

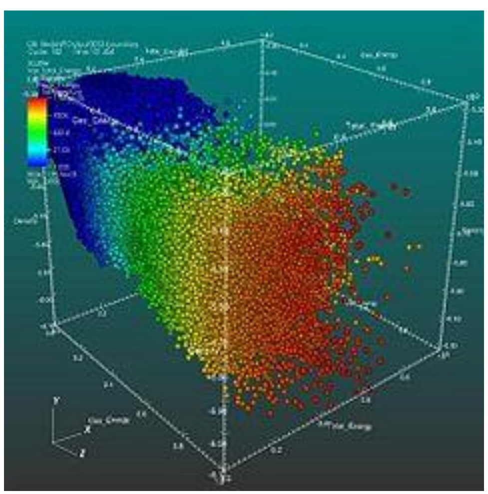
- Border information representing a continuous phenomenon (elevation, temperature)
- Determines the boundary between points with higher and lower values



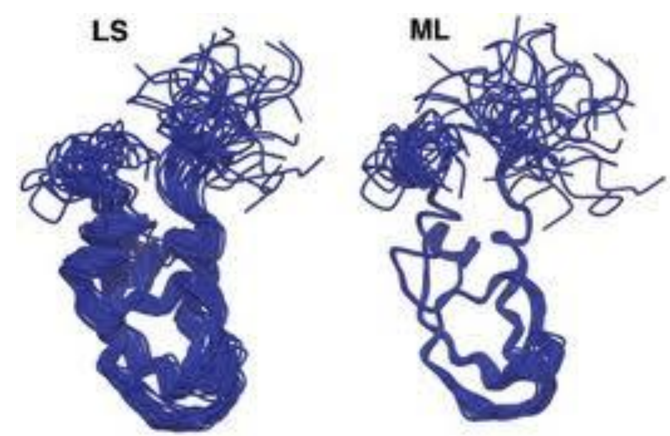
http://www.opendx.org/

2D multivariate data

- Juxtapositioning
  - Stacking of 2D visualizations to 3D
- Superimpositioning
  - Overlapping 2D visualizations
- Both limited by the number of variables



en.wikipedia.org



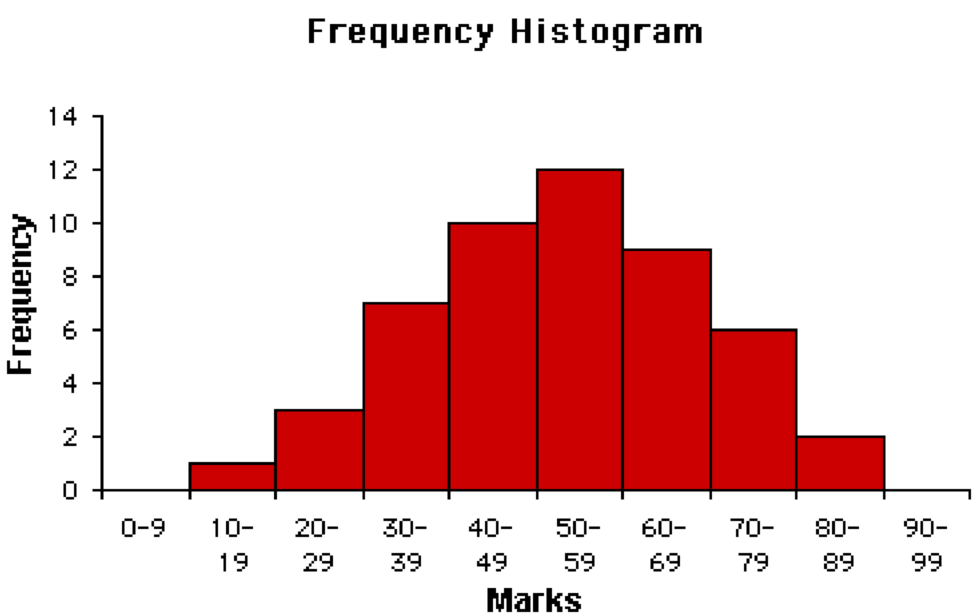
boscoh.com

Studying 2D data

- Simplification of input data – visualization of subsets of the data, projections, summarizations
- Then using previous techniques
- Projection techniques:
  – Frequency histograms
  – Merging rows and columns
  – Linear „probes“

Frequency histograms

- Calculating the frequency in which given values are appearing in the data
- Result is displayed as bar chart



www.microbiologybytes.com

Merging rows and columns

- Localization of regions of interest with high or low variability
- Merging by adding, averaging, calculating median, standard deviation, maximum, minimum
- Color bars, line charts, bar charts

Linear „probes“

- Line (ray probe) passing through the input data
- Using parametric equations and bilinear interpolation
- Defined by two points P_1 and P_2 or by one point and direction vector
- Parametric equation for line:

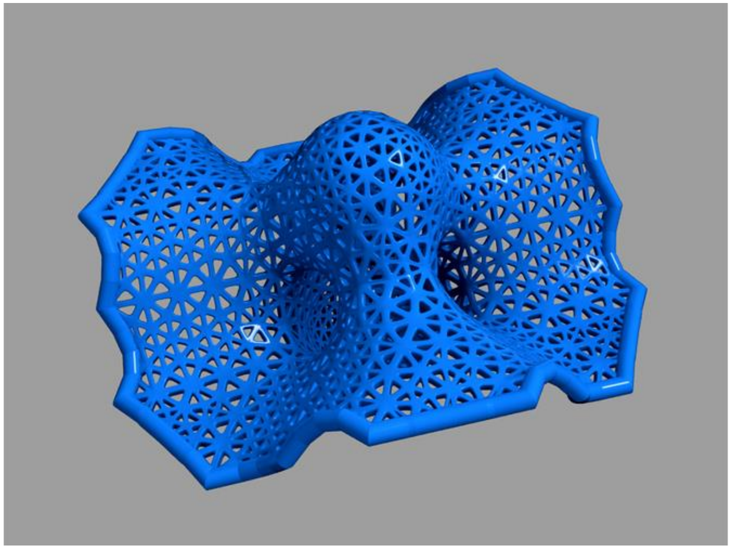
$$P(t) = P_1 + t(P_2 - P_1), \text{ where } 0 \leq t \leq 1.0$$

3D data

- Discrete samples of a continuous phenomenon or set of vertices, edges, and polygons
- Mostly combination of both
- Basic techniques:
  - Visualization of explicit surfaces
  - Volumetric visualization
  - Implicit surfaces

Visualization of explicit surfaces

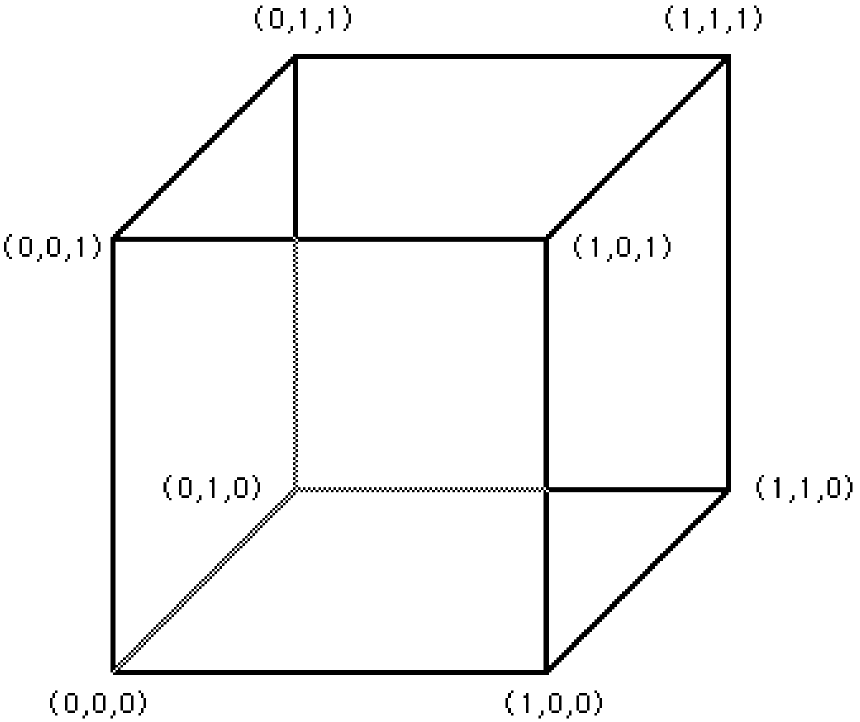
- Explicit surface defined as:
  - List of 3D vertices, edges, planar polygons
  - Set of parametric equations defining x, y, z coordinates of points, along with strategy for their connection (edges, polygons)



www.shapeways.com

Example

vertex[0] = (0., 0., 0.)
vertex[1] = (0., 0., 1.)
vertex[2] = (0., 1., 1.)
vertex[3] = (0., 1., 0.)
vertex[4] = (1., 0., 0.)
vertex[5] = (1., 0., 1.)
vertex[6] = (1., 1., 1.)
vertex[7] = (1., 1., 0.)
edge[0] = (0, 1)
edge[1] = (1, 2)
edge[2] = (2, 3)
edge[3] = (3, 0)
edge[4] = (0, 4)
edge[5] = (1, 5)
edge[6] = (2, 6)
edge[7] = (3, 7)
edge[8] = (4, 5)
edge[9] = (5, 6)
edge[10] = (6, 7)
edge[11] = (7, 4)
face[0] = (0, 1, 2, 3)
face[1] = (8, 9, 10, 11)
face[2] = (0, 5, 8, 4)
face[3] = (1, 6, 9, 5)
face[4] = (2, 7, 10, 6)
face[5] = (3, 4, 11, 7)



Example – unit cylinder in y axis

$y = 1.0, \quad x = \cos \Theta, \quad z = \sin \Theta,$

$0.0 \le \Theta \le 2\pi$ (top base)

$y = 0.0, \quad x = \cos \Theta, \quad z = \sin \Theta,$

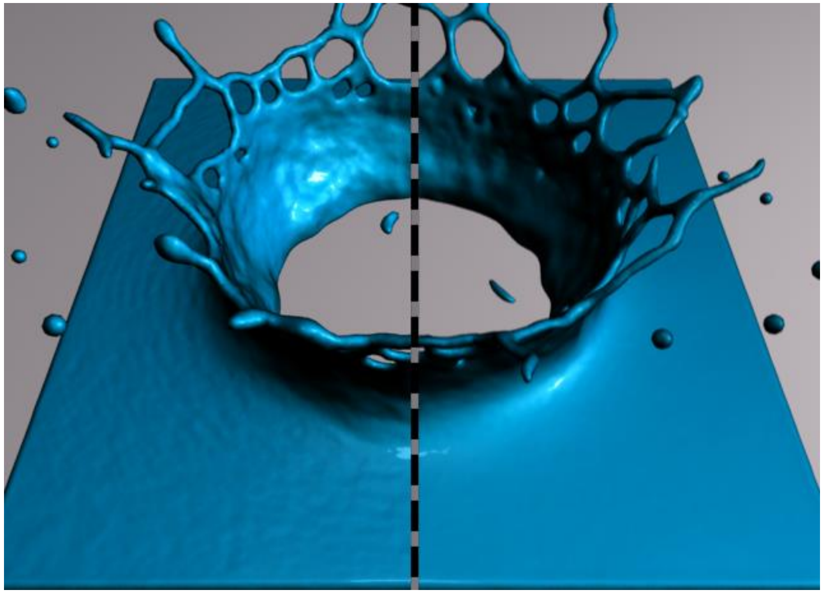
$0.0 \le \Theta \le 2\pi$ (bottom base)

$y = h, \quad x = \cos \Theta, \quad z = \sin \Theta,$

$0.0 \le \Theta \le 2\pi, \ 0.0 \le h \le 1.0$ (middle part)

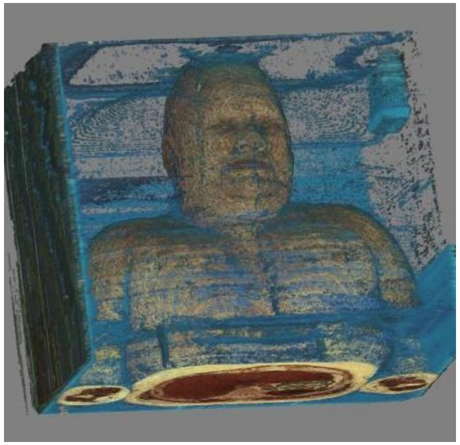
Examples

- Input data associated with:
  - vertices – temperature, weight of vertex
  - edges – strength of chemical bond
  - polygons – map coverage of area



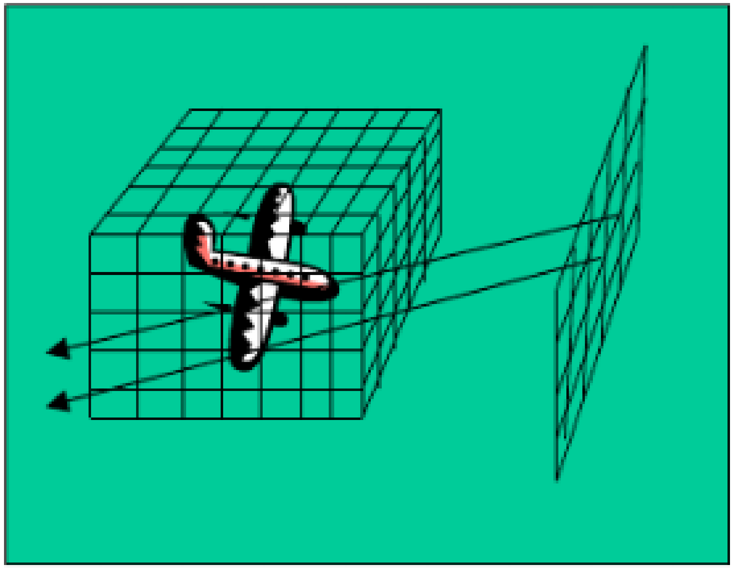
http://pub.ist.ac.at/group_wojtan/projects/meshSPH/index.html

Volumetric visualization



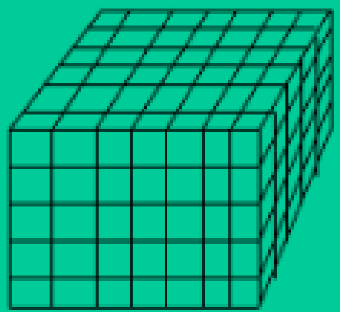
vidi.cs.ucdavis.edu

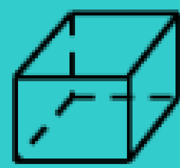
- Using voxels
- Categories:
  - Slicing – using clipping plane
  - Isosurfaces – generating surface
  - Direct volume rendering – ray casting or projecting of voxels to projection plane



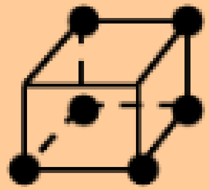
http://www.docstoc.com/docs/92958007/Direct-Volume-Rendering-_DVR_-Ray-casting

Voxel





A voxel is a cubic cell, which
has a single value cover
the entire cubic region



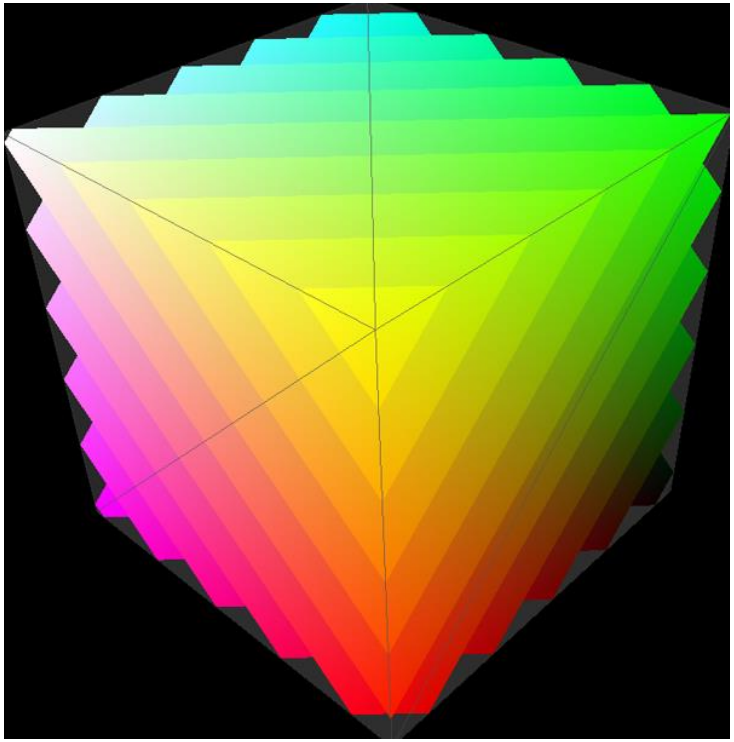
A voxel is a data point
at a corner of the cubic cell
The value of a point inside the
cell is determined by interpolation

Resampling

- Important for most of the volumetric visualization techniques
  – Isosurfaces
  – Slicing
  – Direct volume rendering

Slicing of volumetric data using clip planes

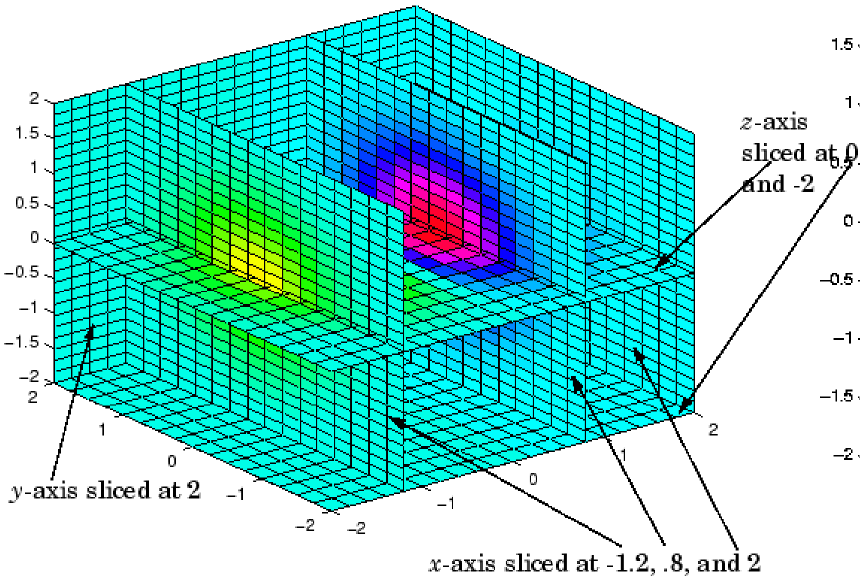
- Creates a subset of input data in lower dimension



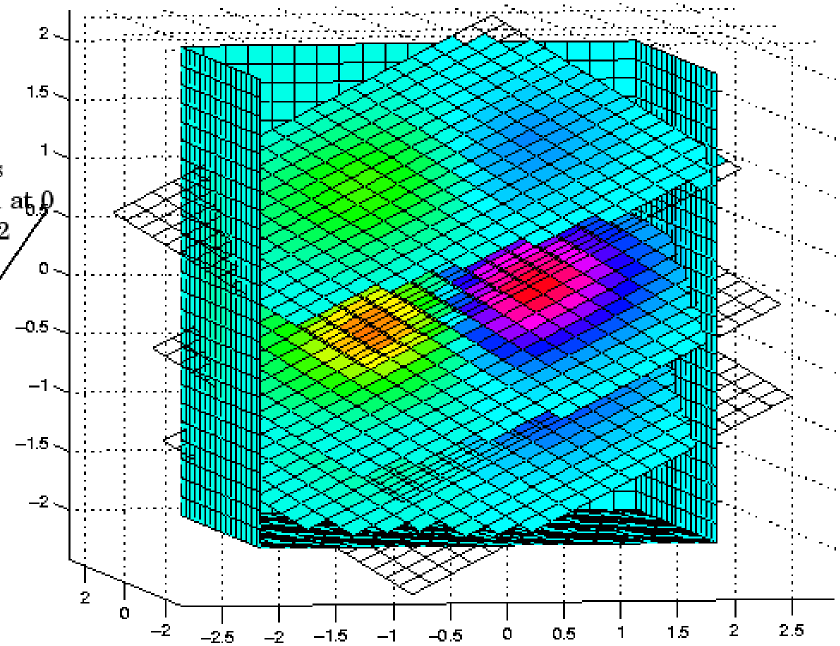
http://doc.instantreality.org/tutorial/volume-rendering/

Slicing of volumetric data using clip planes

- Orientation of clip plane
  - Normal vector of the plane corresponding to one of the main axes
  - Arbitrary orientation



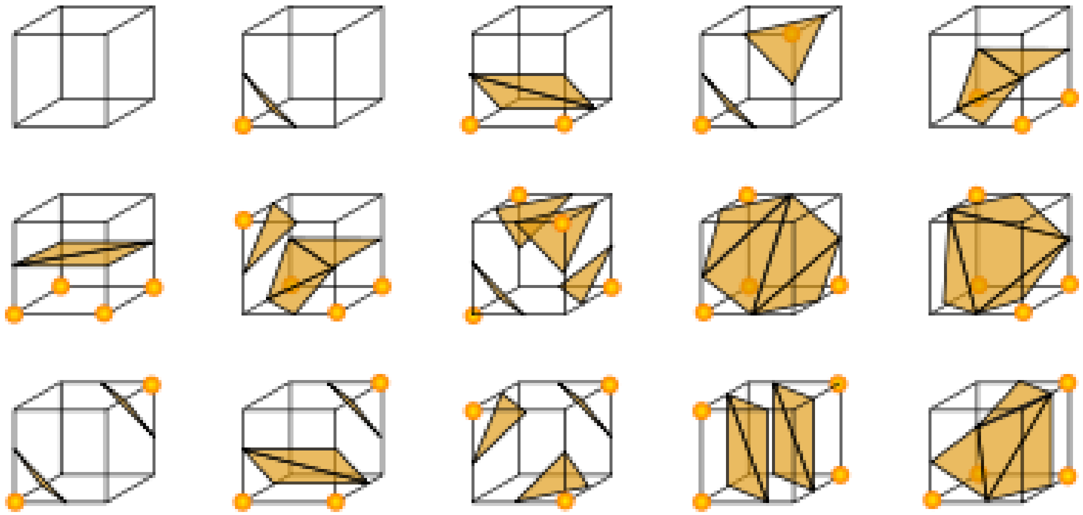
www.mathworks.com



amath.colorado.edu

Generating isosurface using Marching Cubes

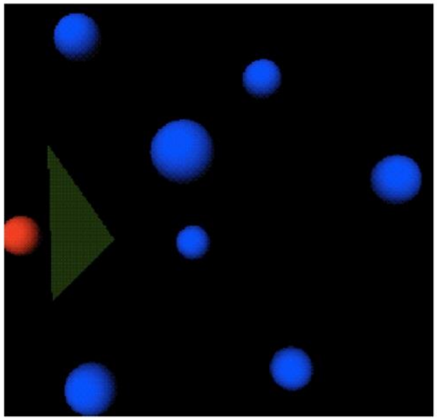
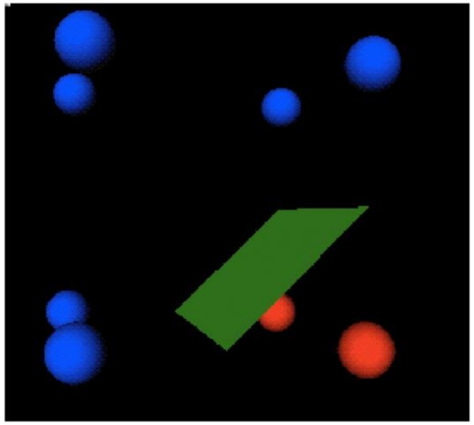
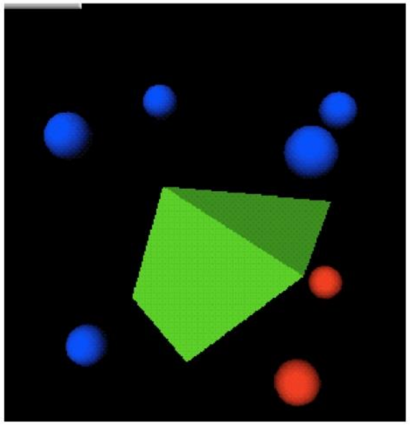
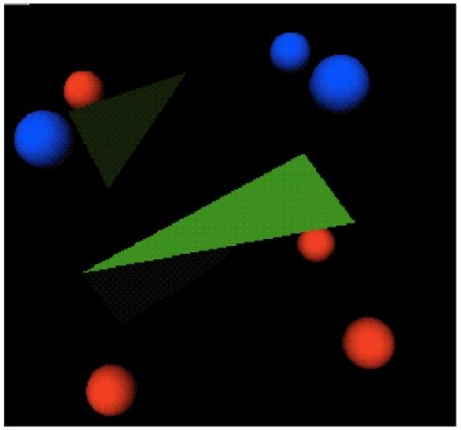
- Lorensen, Cline (1987)
- Voxel = cube with vertices
- Algorithm creates triangles based on the correspondence between vertices and isosurface



en.wikipedia.org

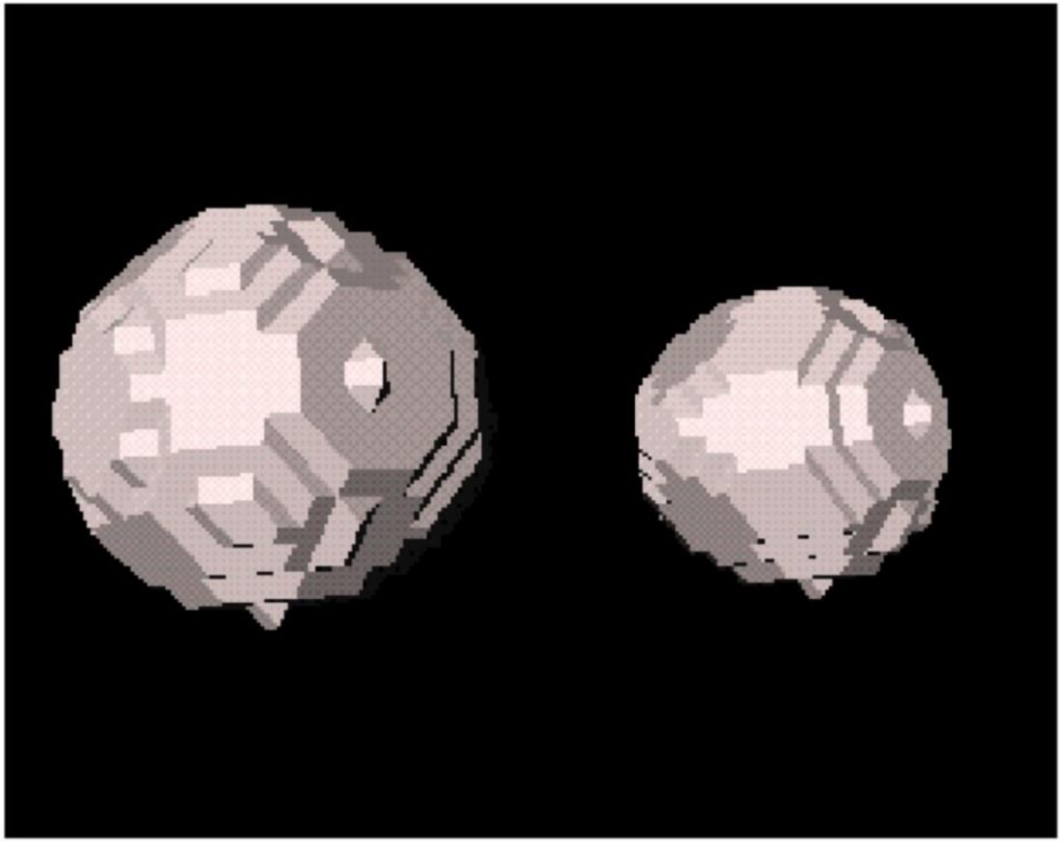
Marching Cubes – details

- 256 configurations, thanks to symmetry only 16 unique (1 = whole cube inside, 1 = whole cube outside)
- Generating corresponding triangles

http://www.opendx.org

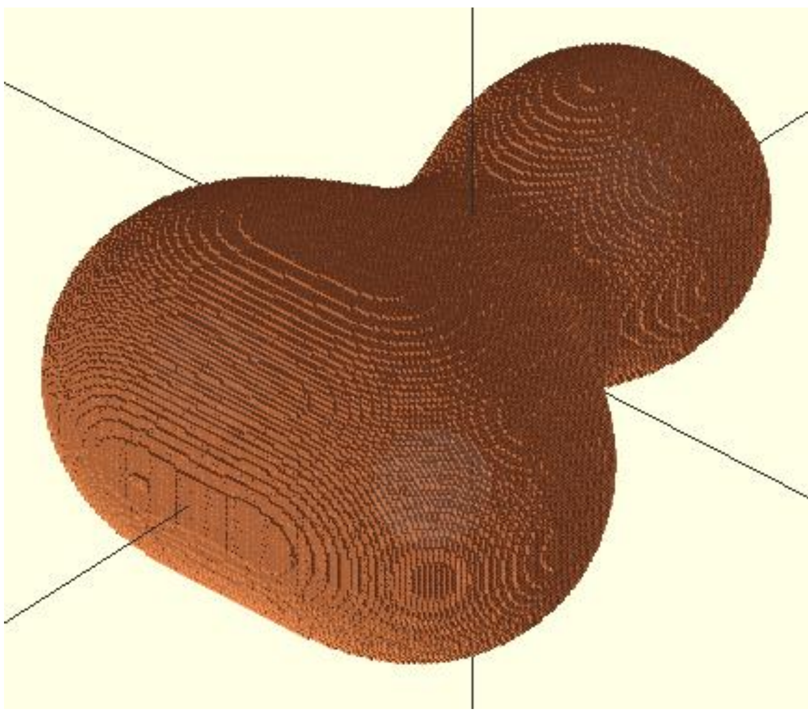
Marching Cubes - details



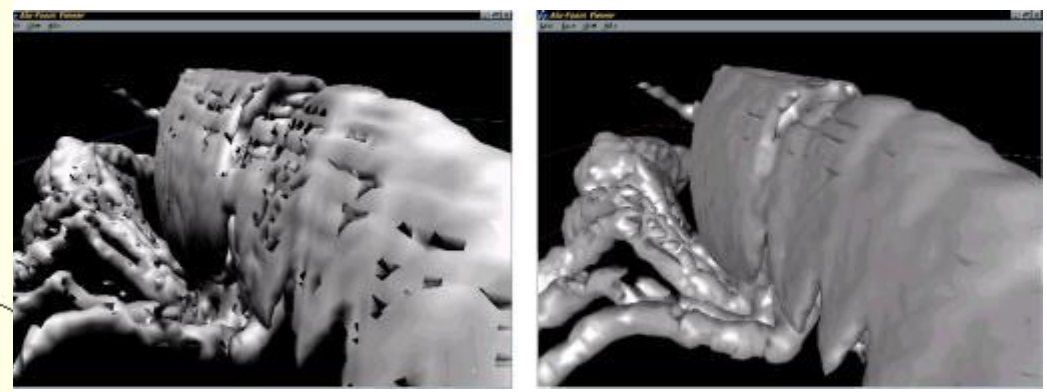
(http://www.opendx.org

Marching Cubes - problems

- High memory requirements
- Holes in data – poor quality of input data



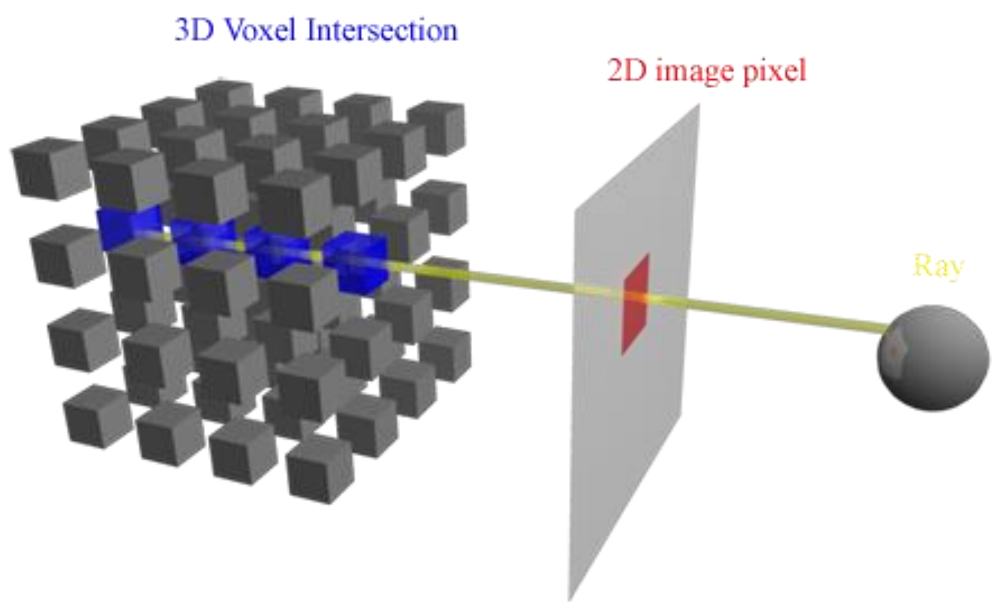
williamaadams.wordpress.com



www.cescg.org

Direct volume visualization

- Pixels of the resulting image computed individually – using ray casting or voxel projection
- Methods:
  - Forward mapping
  - Inverse mapping (ray casting)



www.volviz.com

Forward mapping - problems

- F1: How to deal with pixels which are influenced by more voxels?
- F2: How to deal with pixels without any voxels mapped onto them?
- F3: How to deal with situation when voxels are projected to positions between pixels?

Inverse mapping - problems

- I1: How to choose correct number of points along ray which will be sampled?
- I2: How to calculate the value in these points if they hit the space between voxels?
- I3: How to combine points hit by the ray?

Solution

- F2 and F3: Mapping of each voxel to a region of the projection plane. Each voxel then partially influences values of several neighboring pixels

- I1: Determining the spacing between pixels and setting the sampling frequency to the smaller value than this spacing

Solution

- F1 and I3: Compositing
  - Each voxel has associated the transparency value
  - Voxel i has color c_i and transparency o_i, then its contribution to the resulting pixel value is:

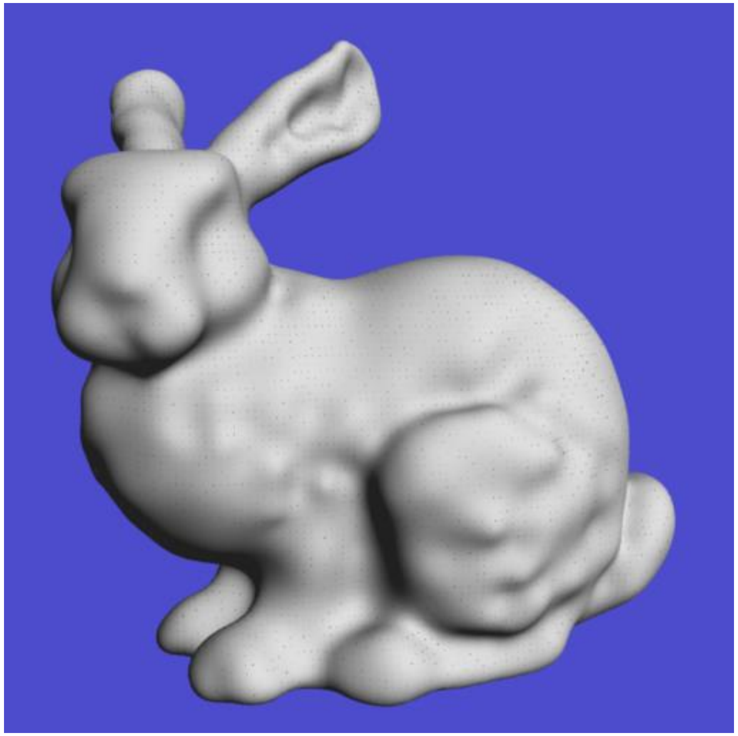
$$c_i * o_i * \prod_{j=0}^{i-1}(1 - o_j)$$

  - Resulting pixel value is then determined as:

$$I(x,y) = \sum_{i=0}^{n} c_i * o_i * \prod_{j=0}^{i-1}(1 - o_j)$$

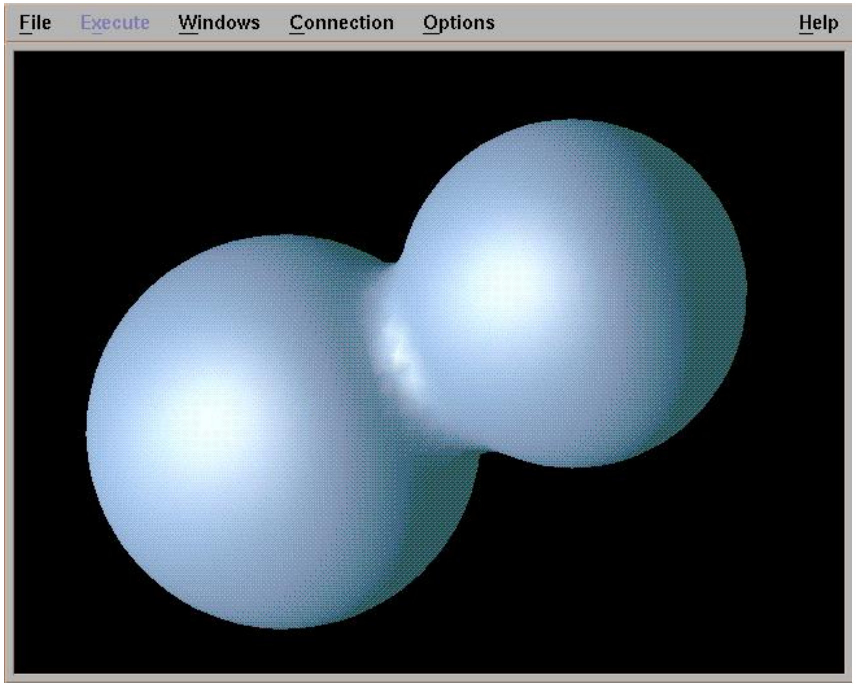
Implicit surfaces

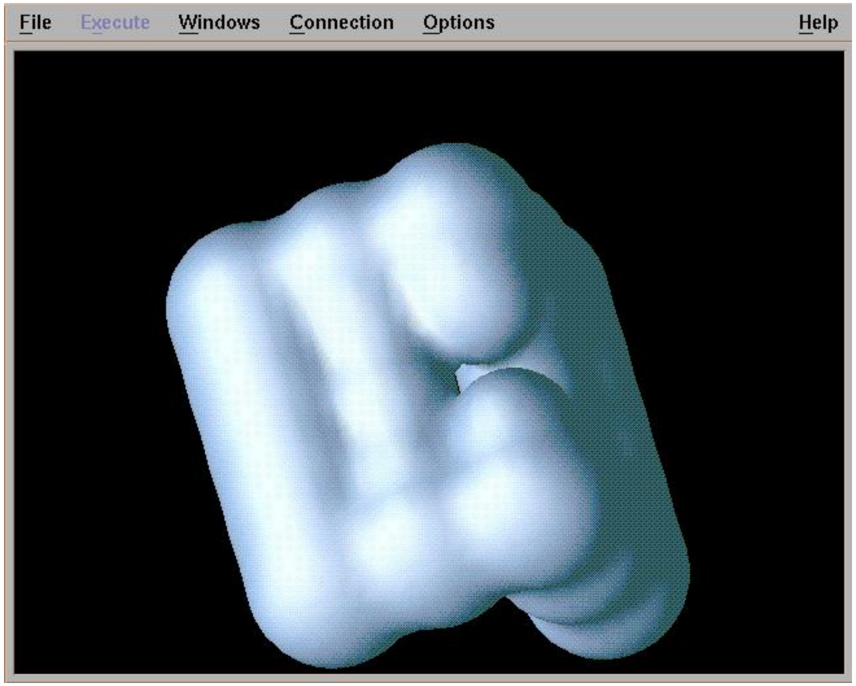
- Surface is defined as zero contour for function with two or three variables



http://www.cs.umd.edu/class/spring2005/cmsc828v/papers/vimp_tog.pdf

Implicit surfaces





http://www.opendx.org

Dynamic data

- Flow visualization – methods for visualizing the dynamic behavior of fluids

www.formula1-dictionary.net

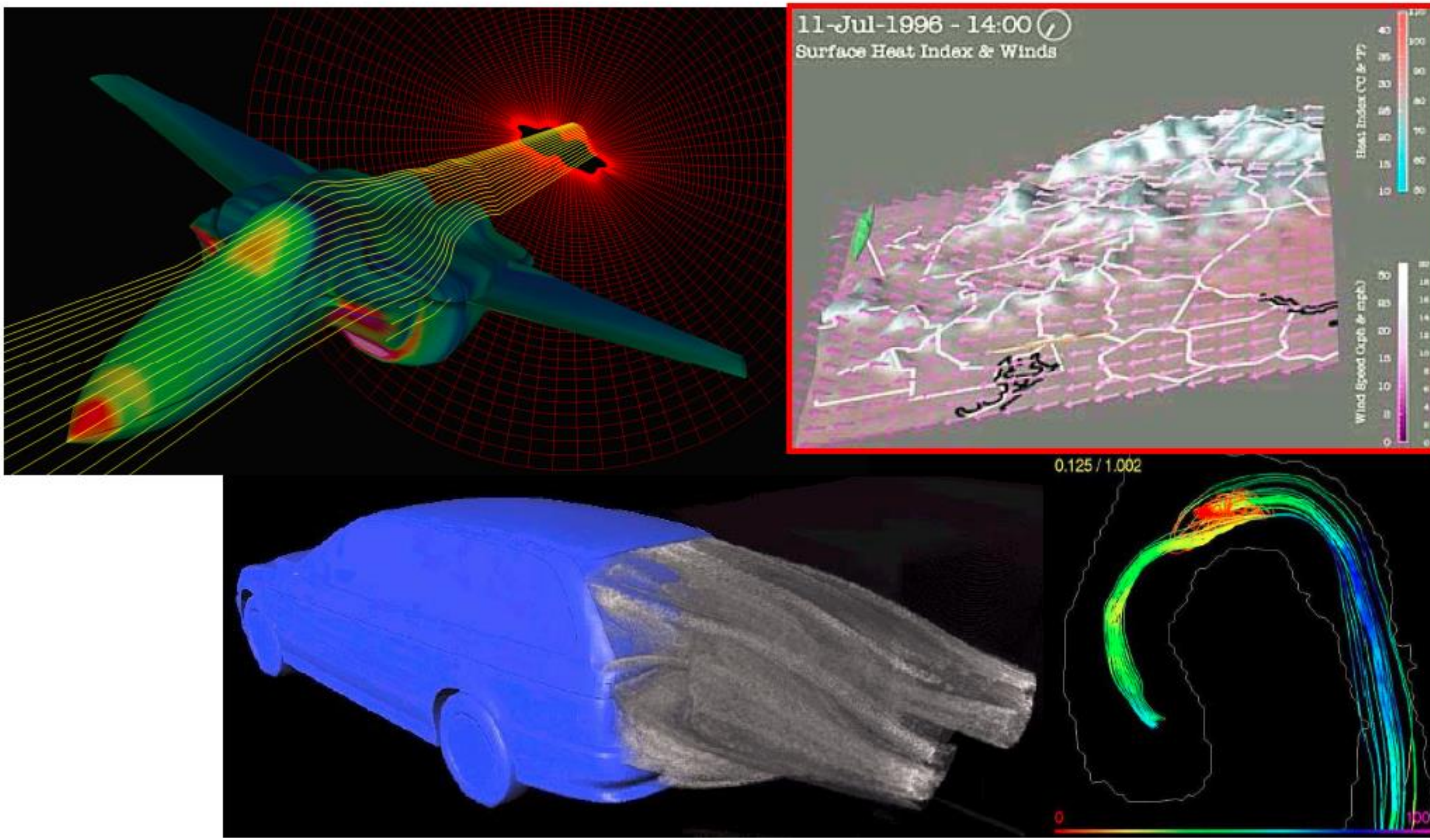
Flow visualization

- Visualization of changes
- Typically more than 3D
- User goals
  - Data overview vs. details

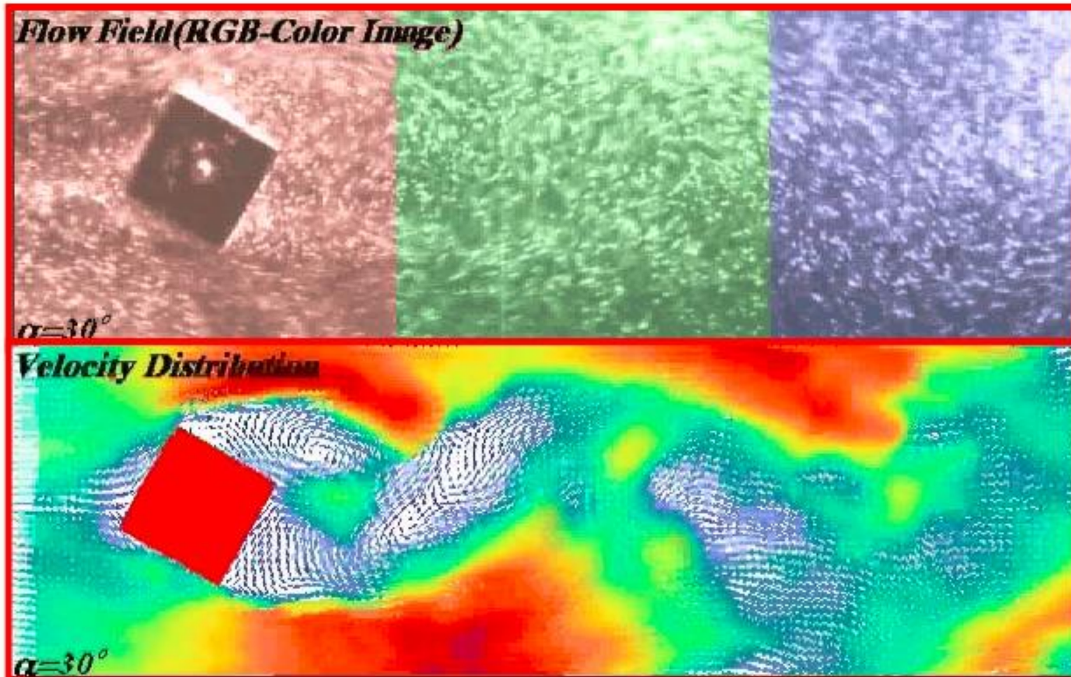
http://en.wikipedia.org/wiki/File:Windkanal.jpg

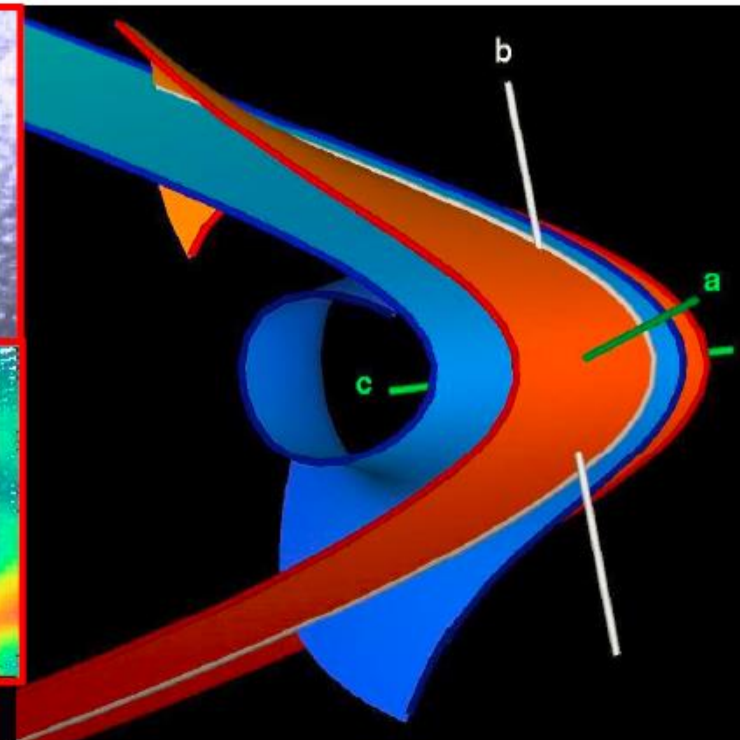
- Input data:
  - Simulation – flight, ship, car industry, weather forecast, medicine (blood flow), …
  - Measurements – wind tunnel (aerodynamics)
  - Models – using differential equations

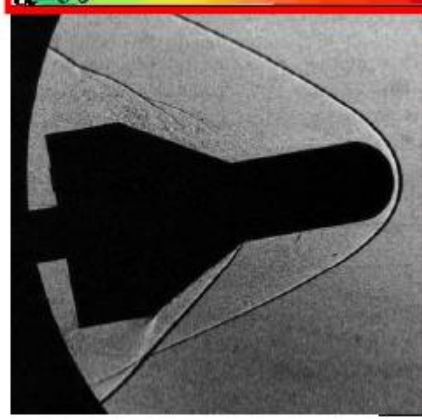
Examples

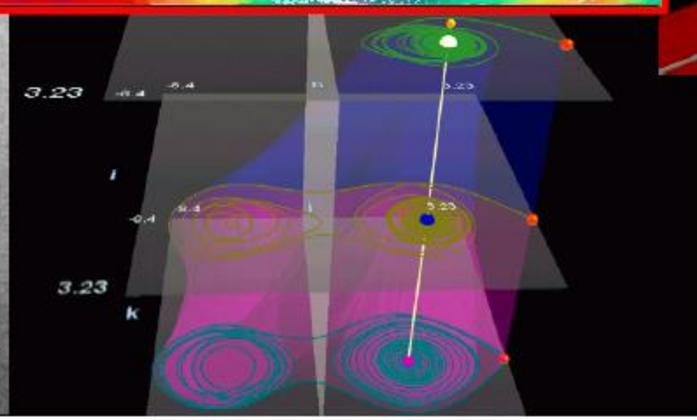


Examples


Flow Field(RGB-Color Image)
α=30°
Velocity Distribution
α=30°

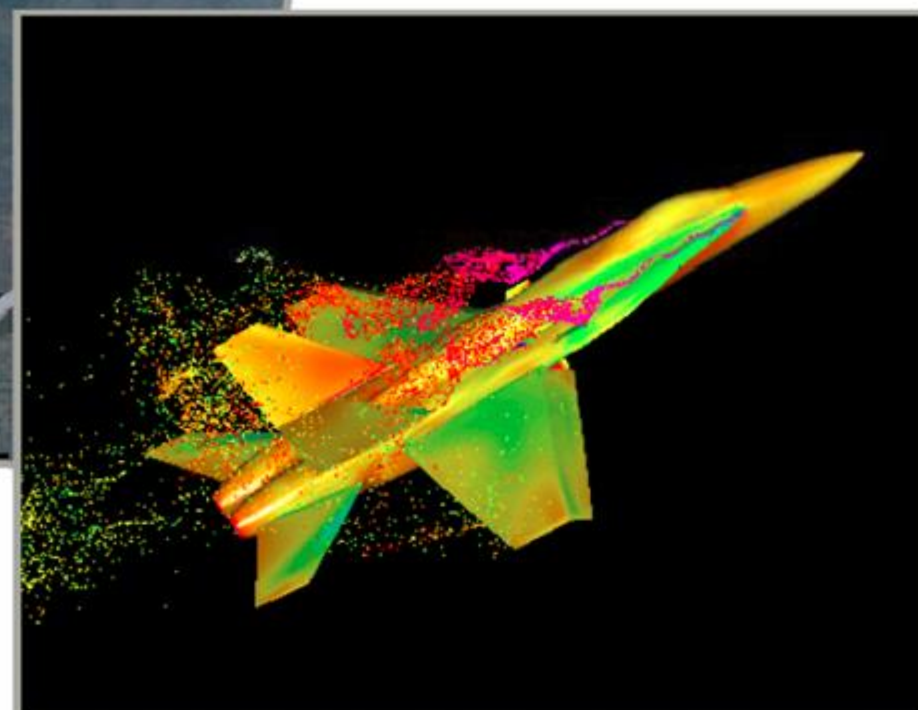

b
a
c




3.23
i
3.23
k

Comparison with real experiments

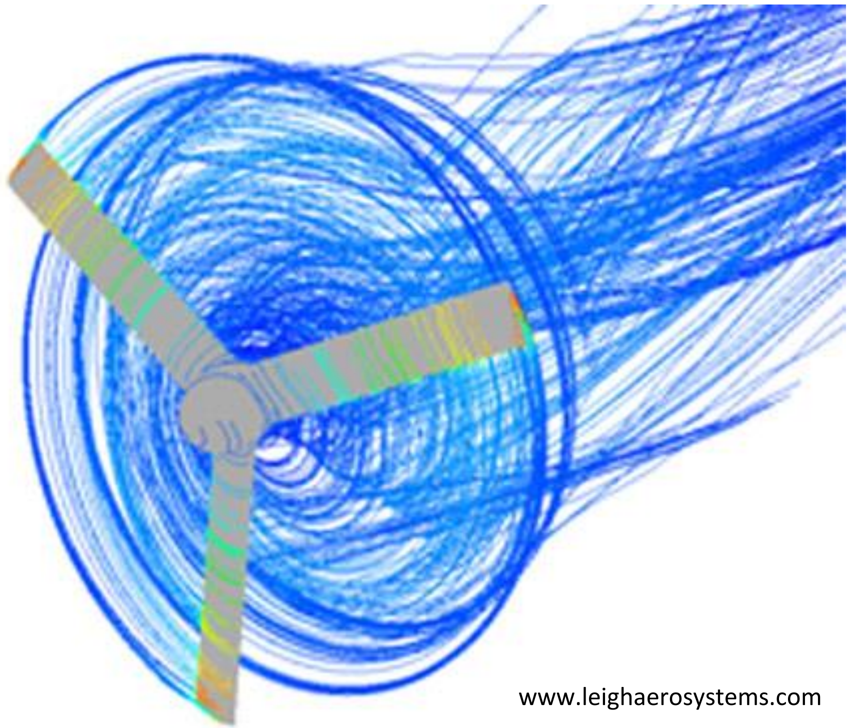
Experiment



Simulation

Dynamic data

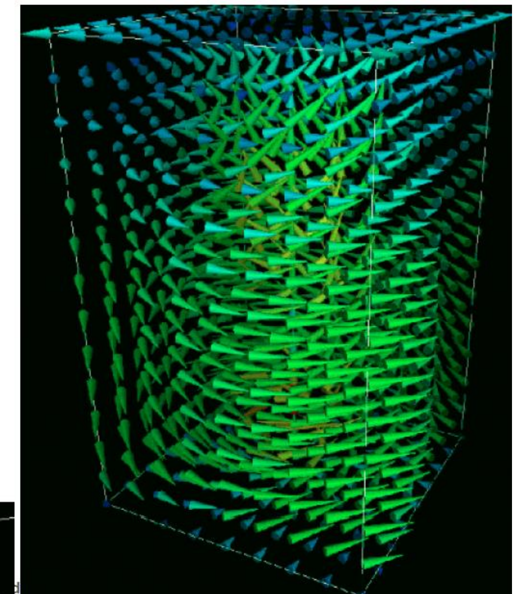
- Computational Fluid Dynamics (CFD) – simulating dynamic behavior of fluids under different conditions
- Two variants of generated data:
  - Static (time independent) field
  - Time-dependent field

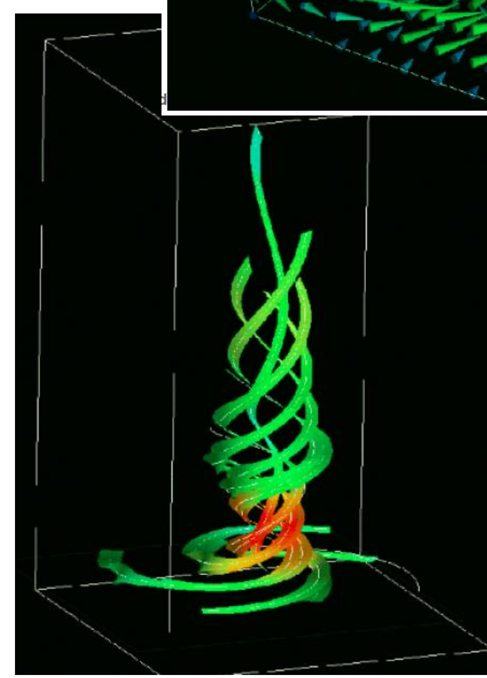


www.leighaerosystems.com

Direct vs. Indirect flow visualization

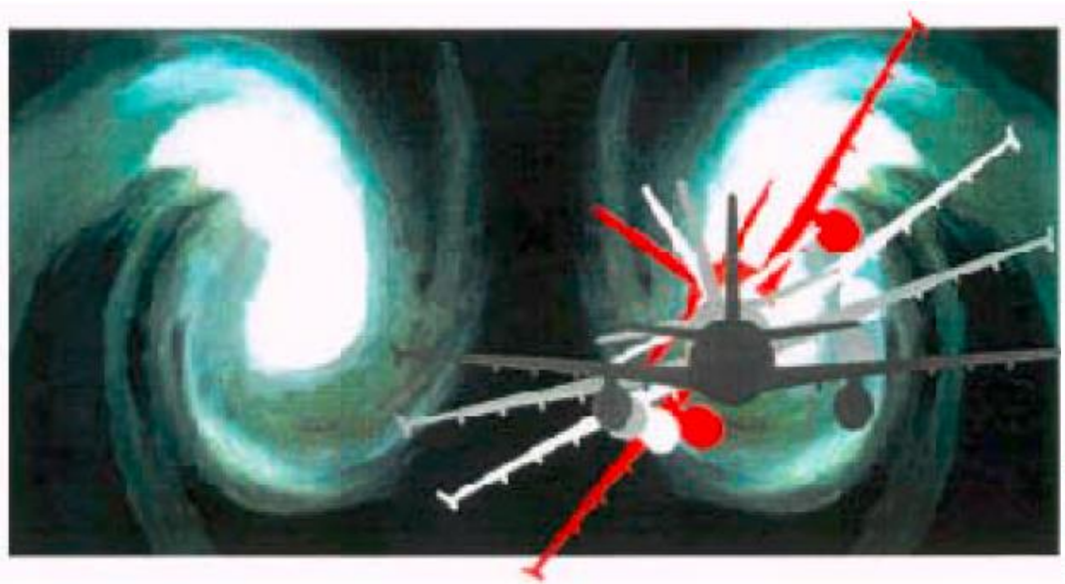
- Direct
  - View onto the current state of the flow
  - Vector field visualization
- Indirect
  - Visualizing the evolution of flow over time
  - Streamlines, streamsurfaces





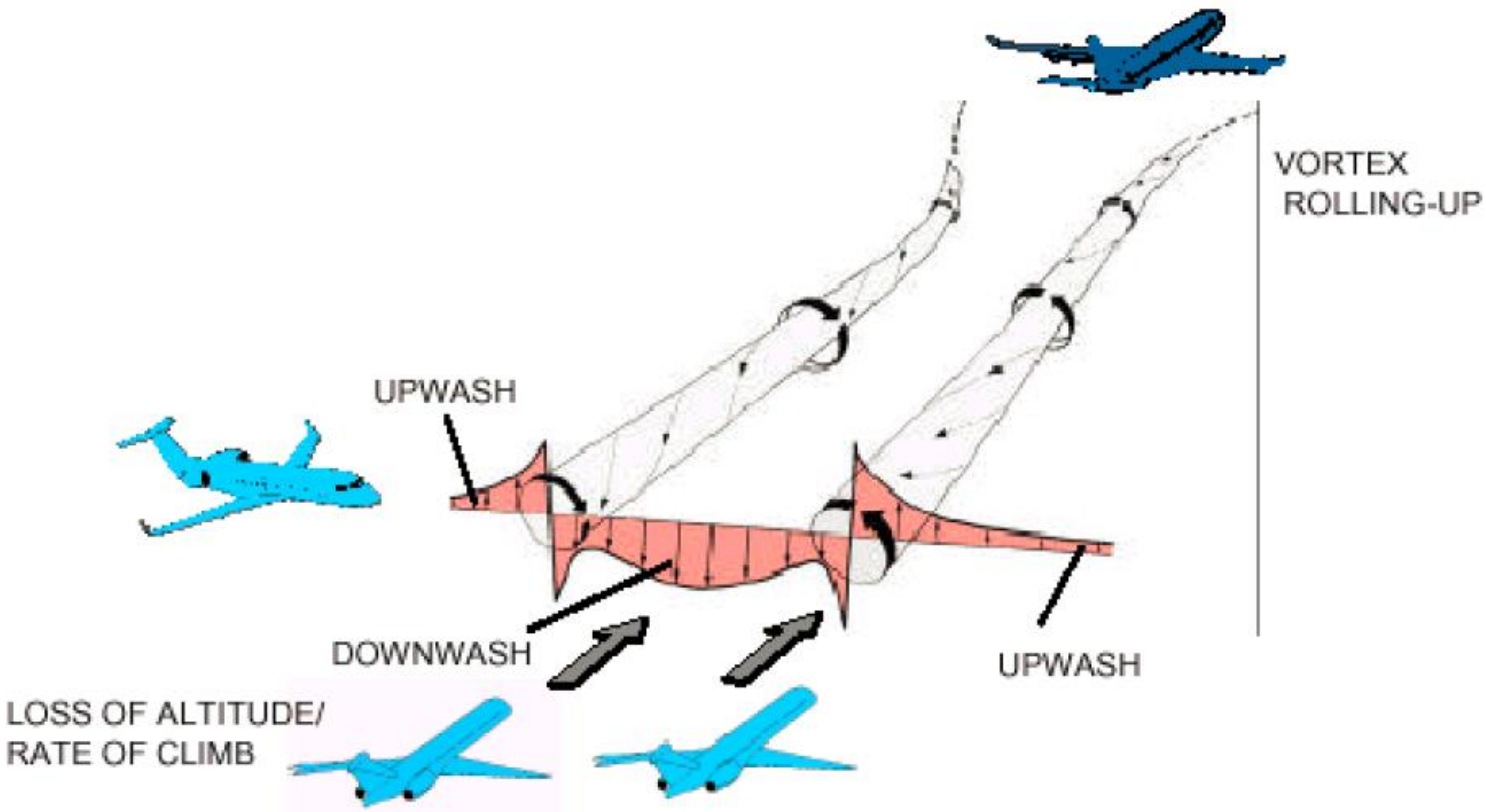
Example – Wind-Tip vortex

- Problem: turbulence behind plane



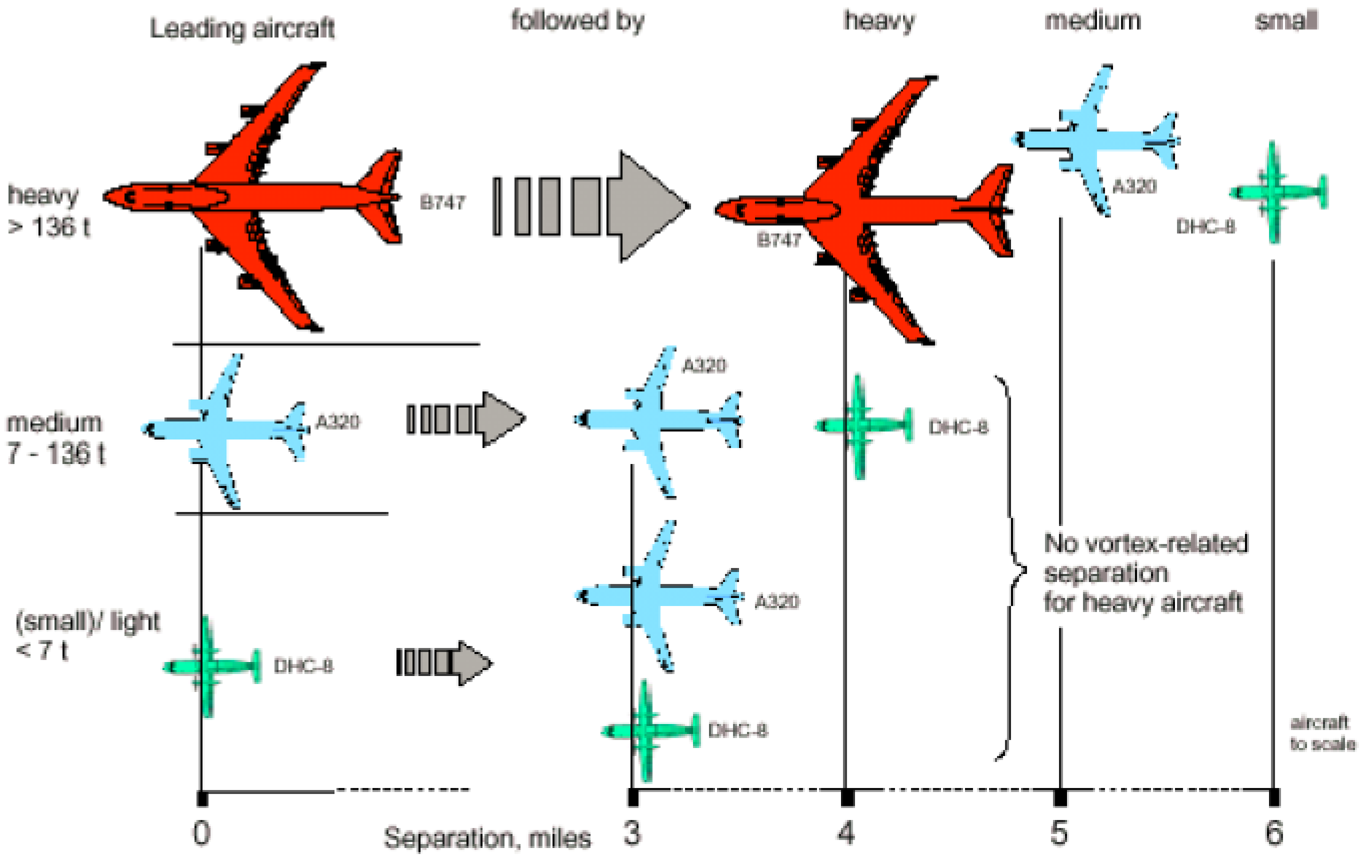
Example – Wind-Tip vortex

- Vortex can be dangerous!



Example – Wind-Tip vortex

- It is crucial to maintain certain distances

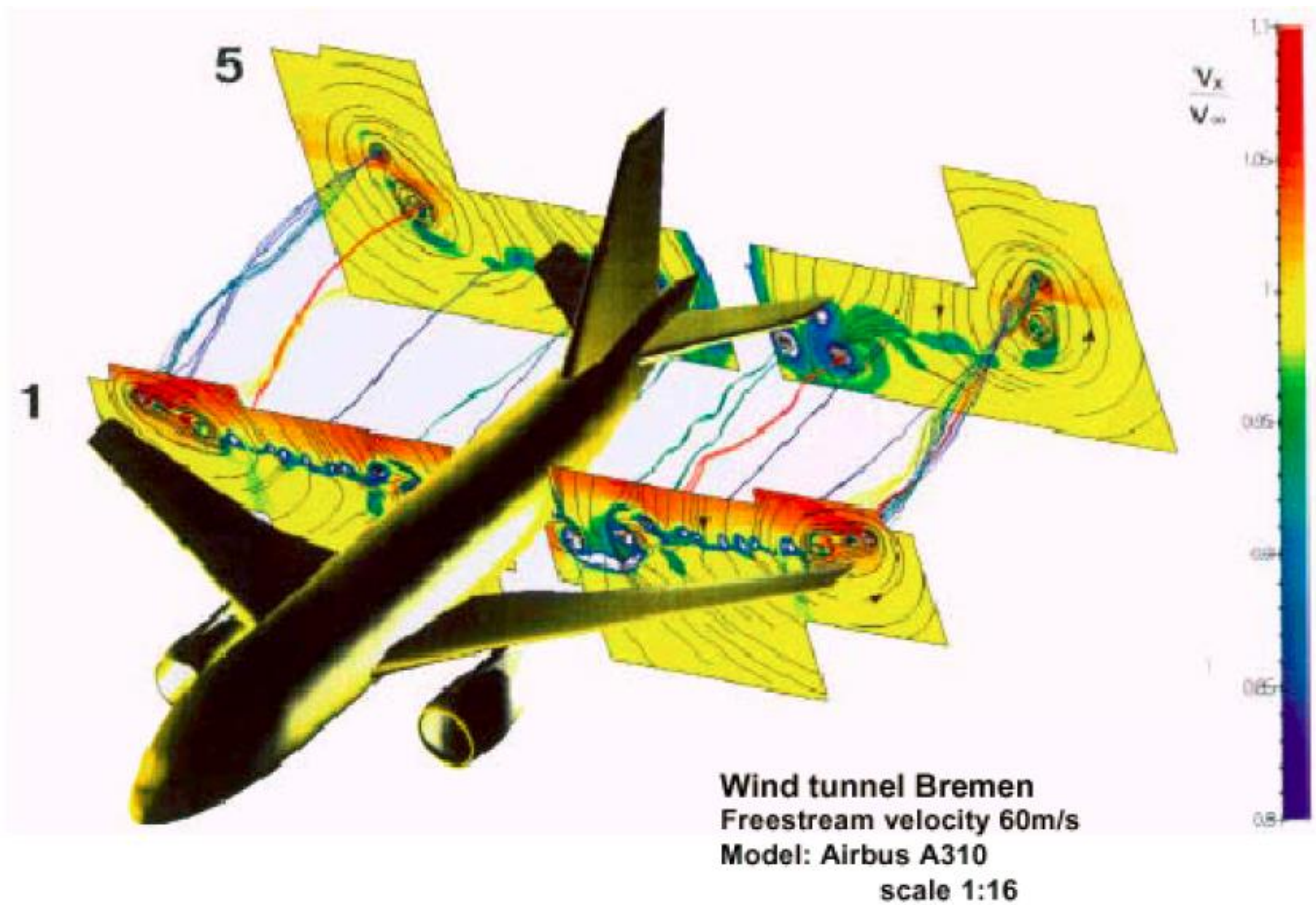


Example – Wind-Tip vortex

- Simulation in wind tunnel

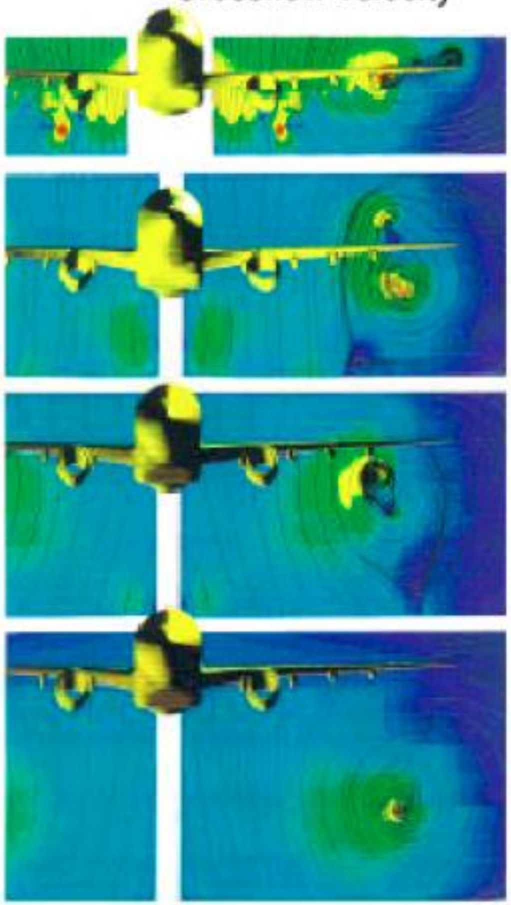
Example – Wind-Tip vortex

- And subsequent visualization

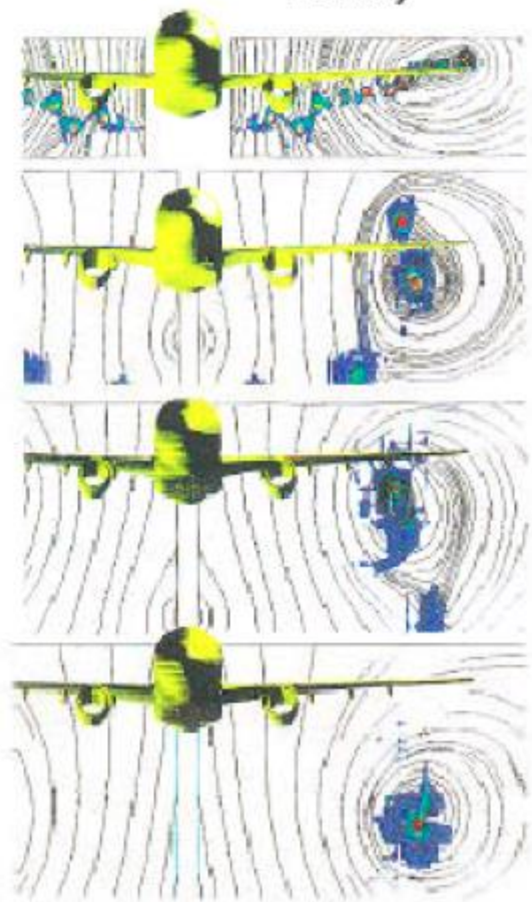


DNW tunnel
Freestream velocity 60m/s

Crossflow velocity



Vorticity

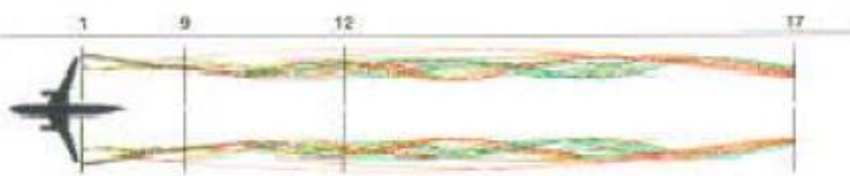


Surveying plane 1
0.03 wing spans behind wing tip

Surveying plane 9
1 wing spans behind wing

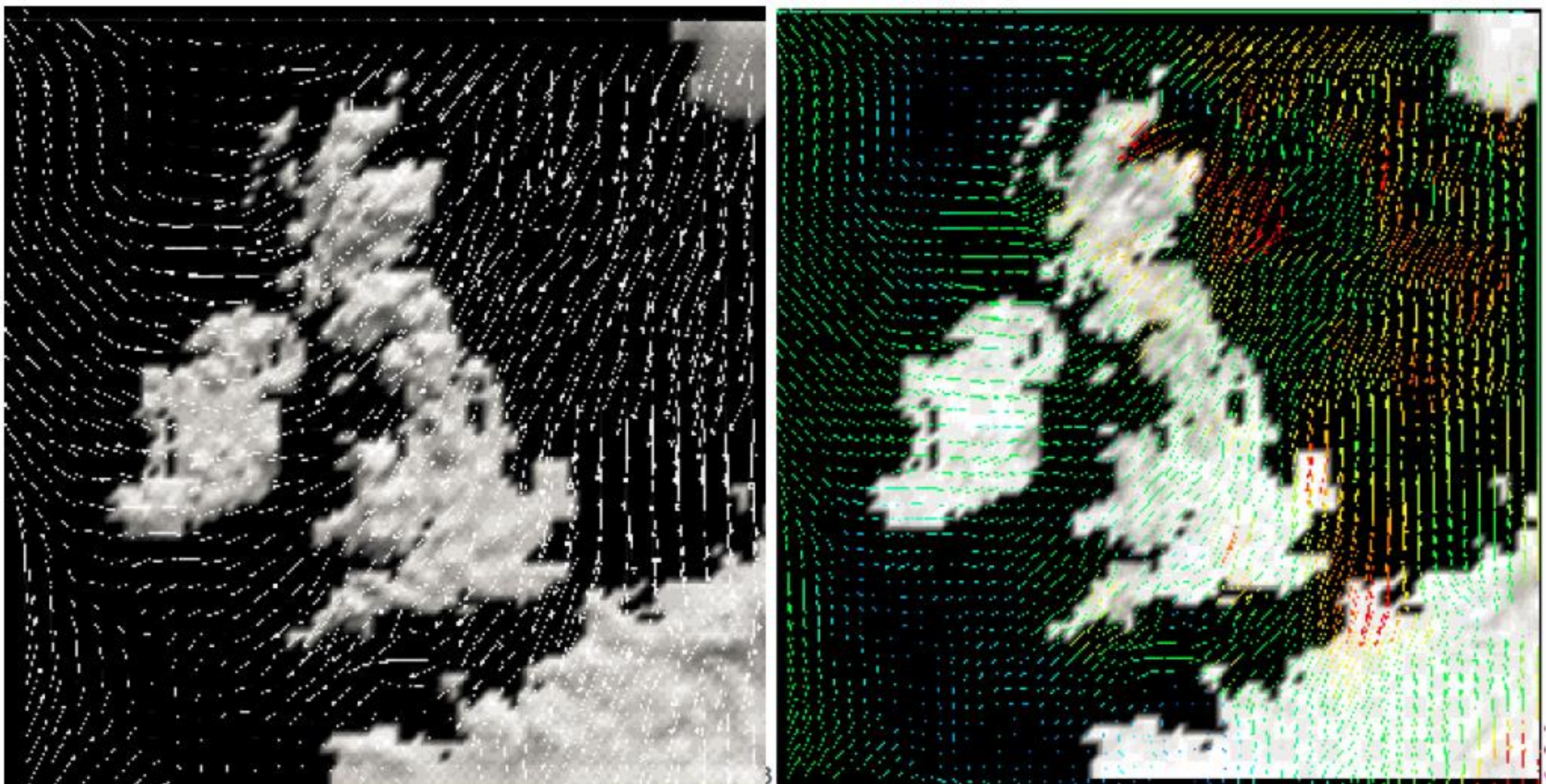
Surveying plane 12
2.5 wing spans behind wing

Surveying plane 17
6.8 wing spans behind wing



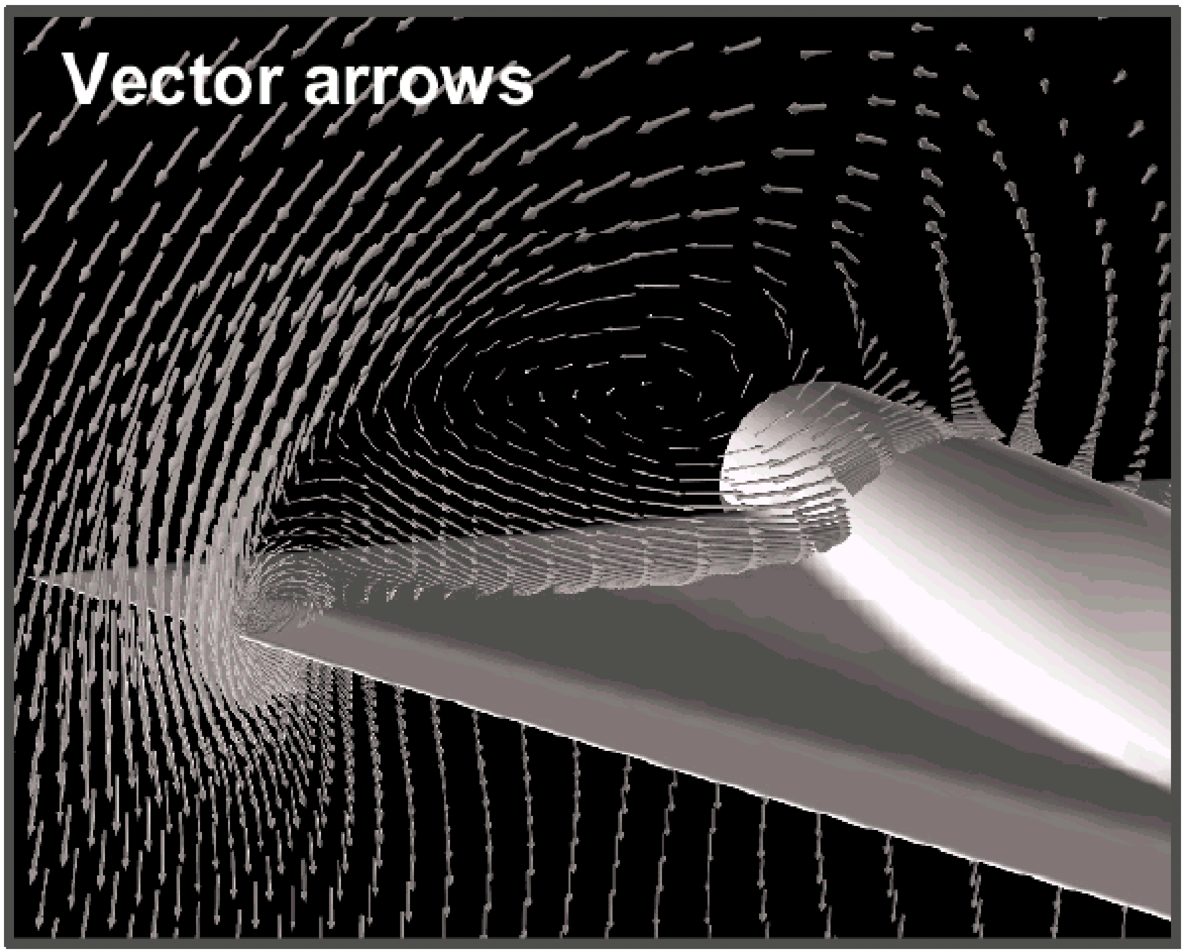
Flow visualization using arrows

- 2D – scaling vs. coloring of arrows



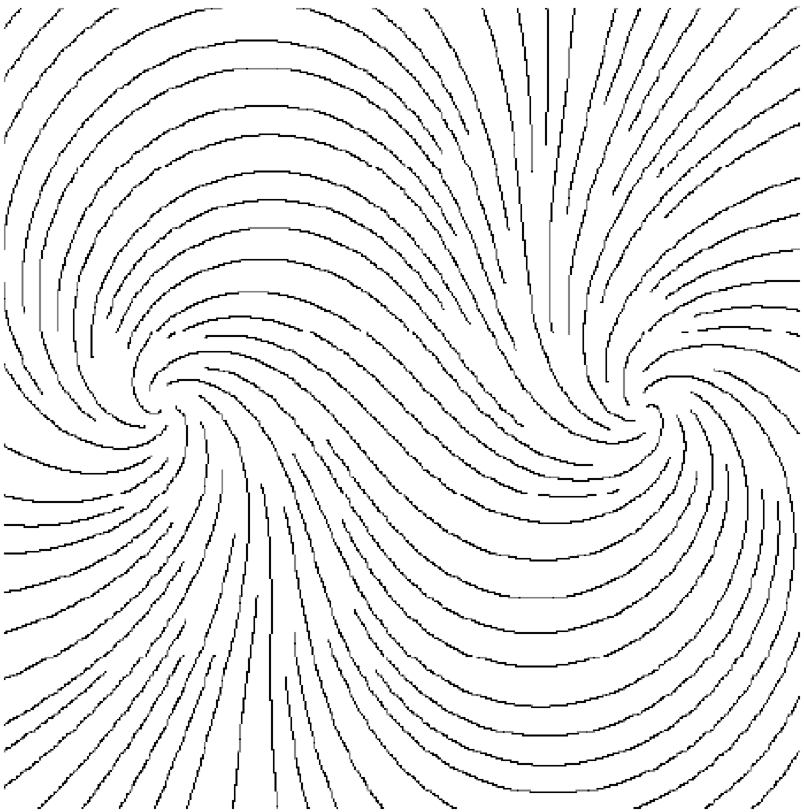
Flow visualization using arrows

- 3D – arrows only in certain „layers“

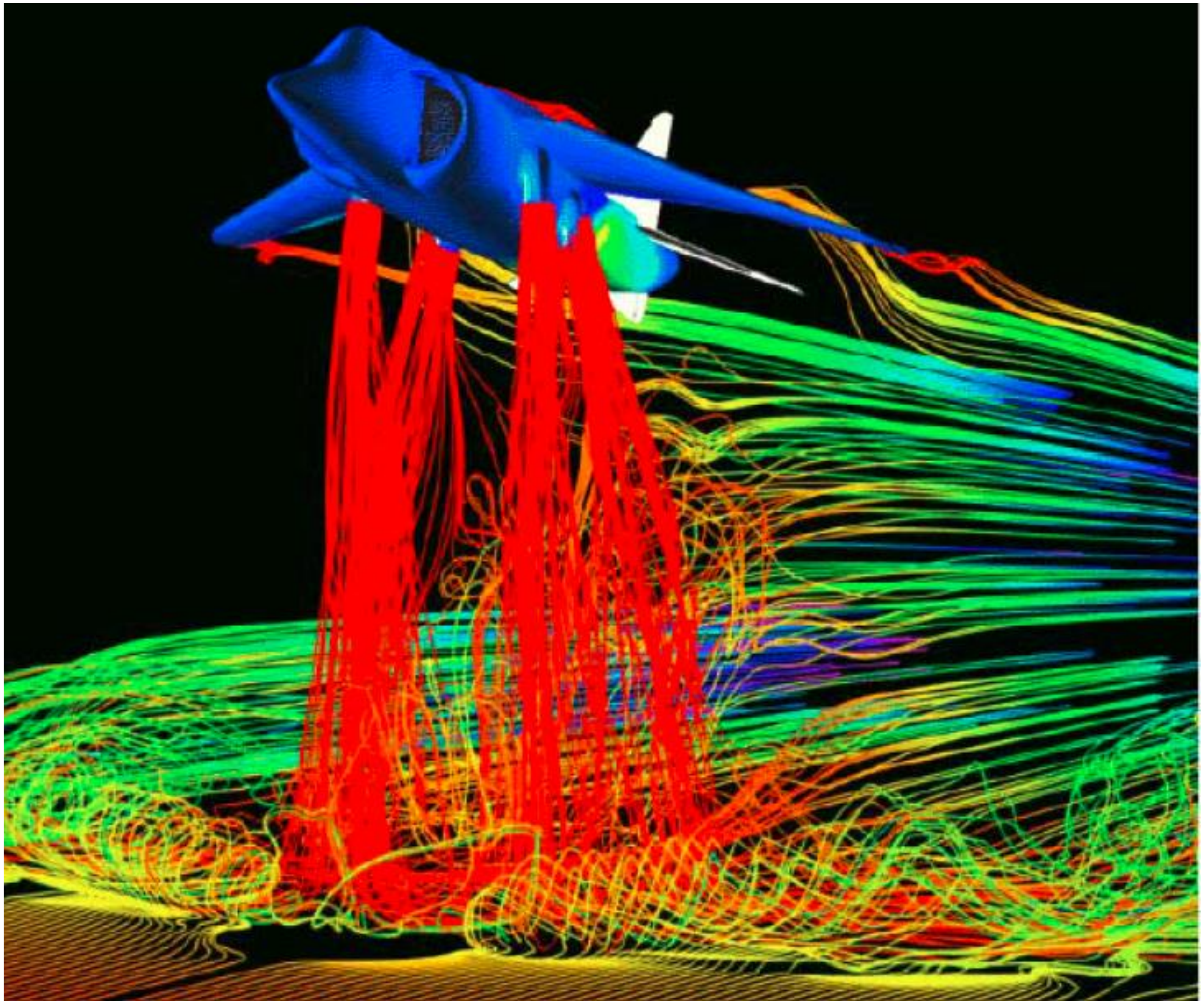


Flow visualization using streamlines

- Streamlines = paths of individual particles in the flow



Streamlines in 3D



Algorithm –positioning of streamlines

- Main idea: streamlines should not be too close to each other
- Principle:
  - Parameters:
    - d_{sep} starting distance
    - d_{test} minimal distance

Algorithm –positioning of streamlines

- Calculate initial streamline, insert it into queue
- Set the initial streamline as active

WHILE not finished DO

TRY get new point in d_{sep} distance from the active streamline

IF found THEN calculate new streamline and insert to queue

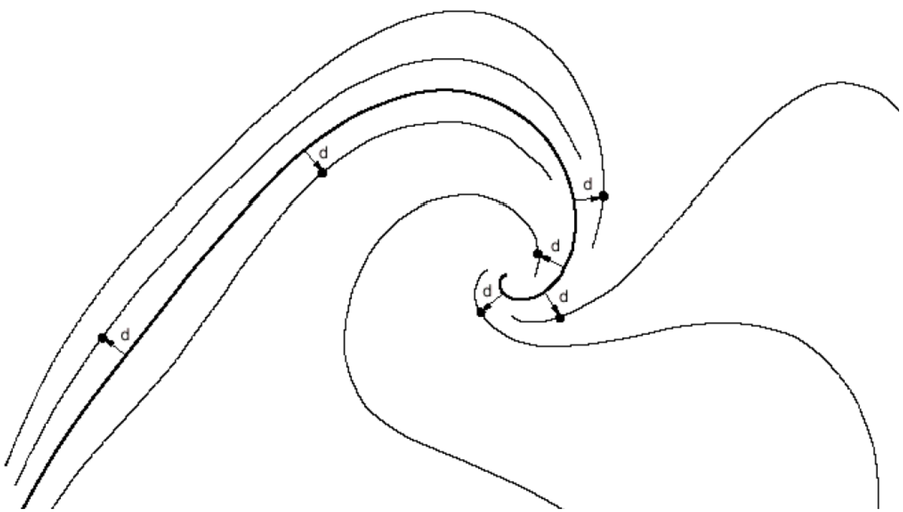
ELSE IF queue is empty

THEN end loop

ELSE next streamline in queue becomes active

Finishing generation of streamlines

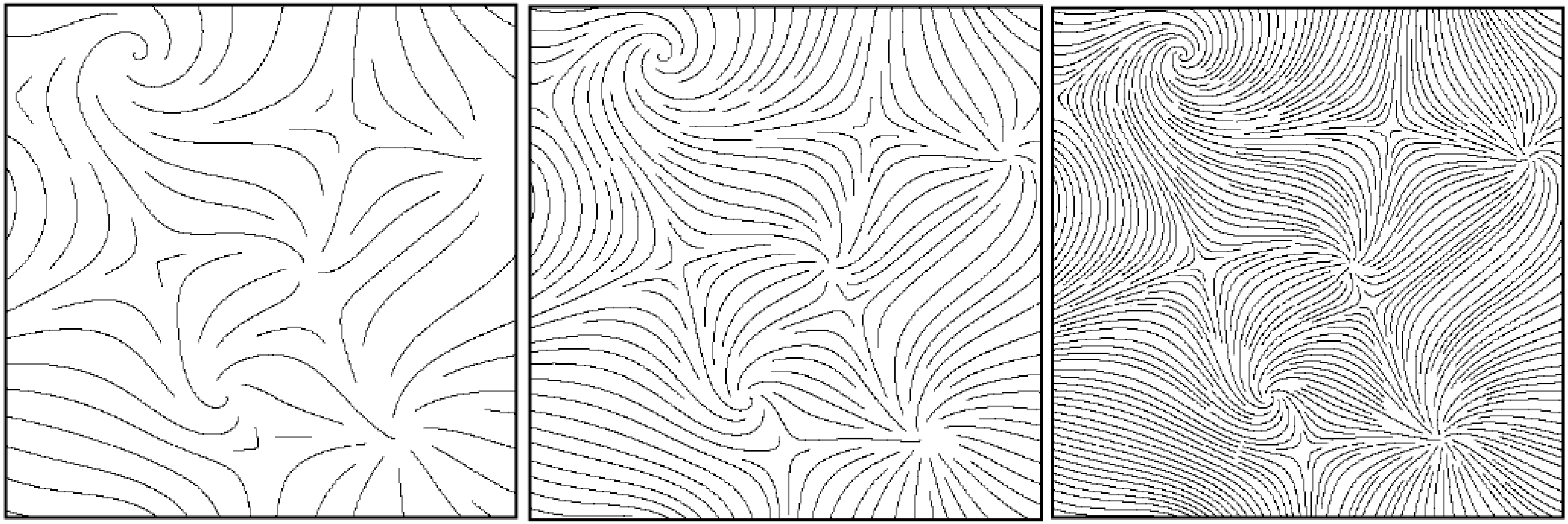
- When the distance to the neighboring streamline ≤ d_{test}
- When the streamline leaves the predefined domain
- When the streamline is too close to itself
- After a predefined number of steps



Streamlines – influence of density by d_{sep}

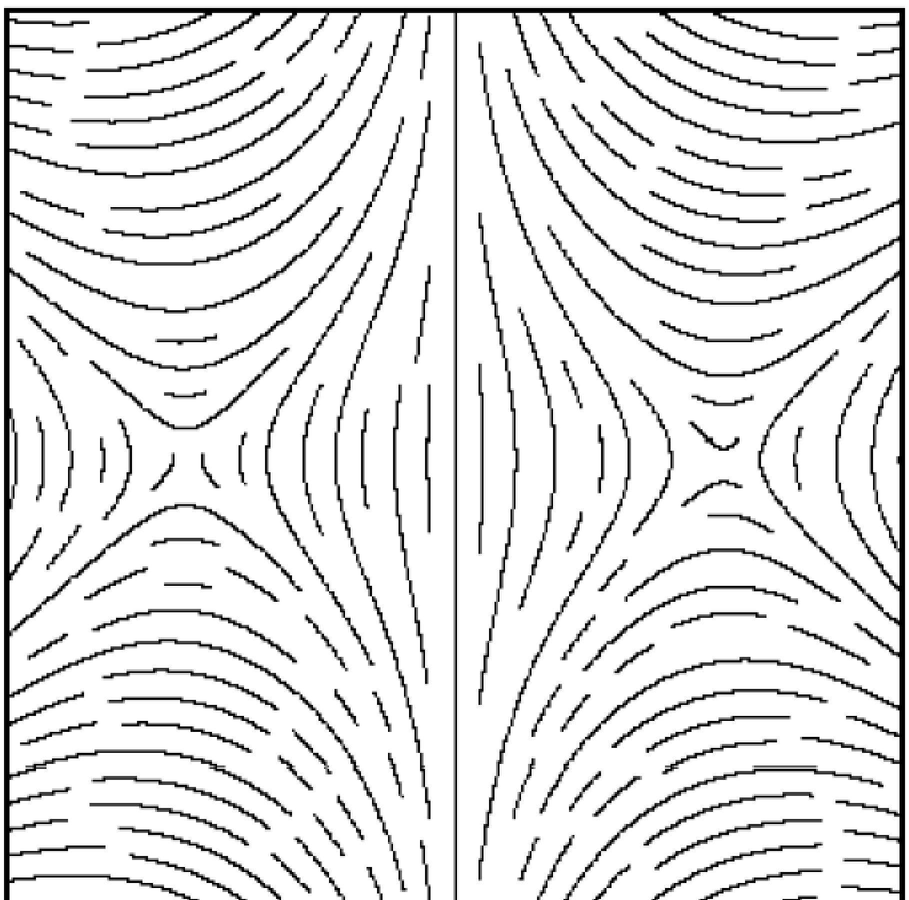
- Relative to the image width:

6% 3% 1.5%

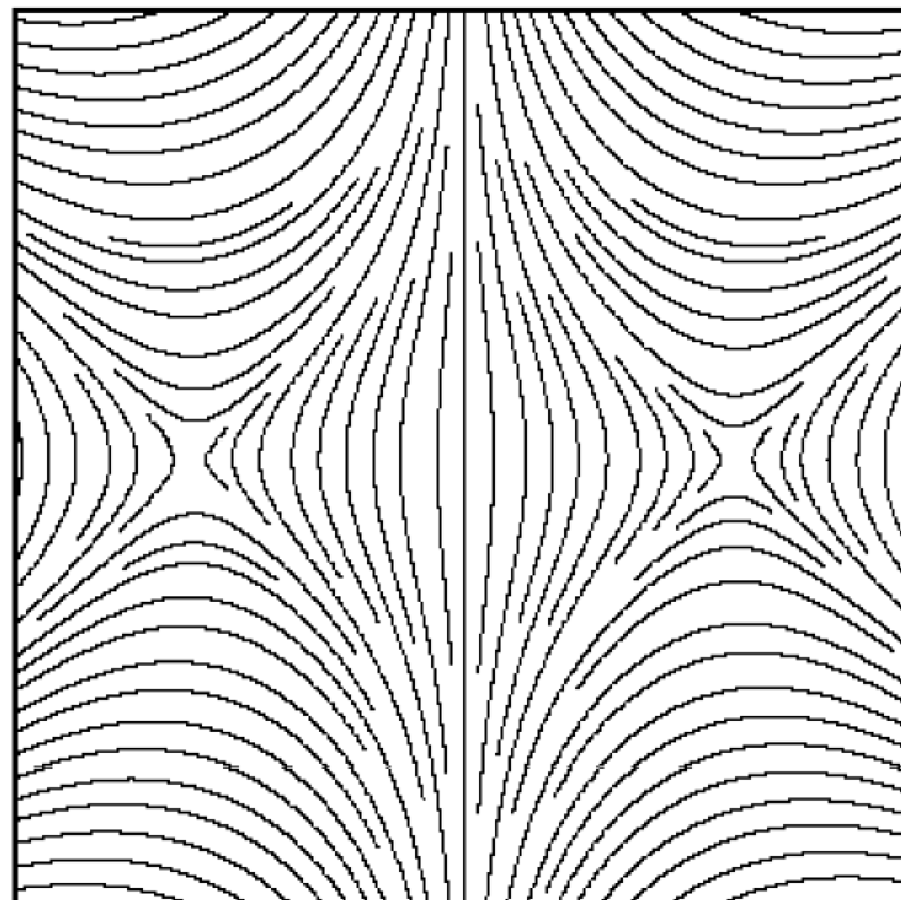


d_{sep} vs. d_{test}

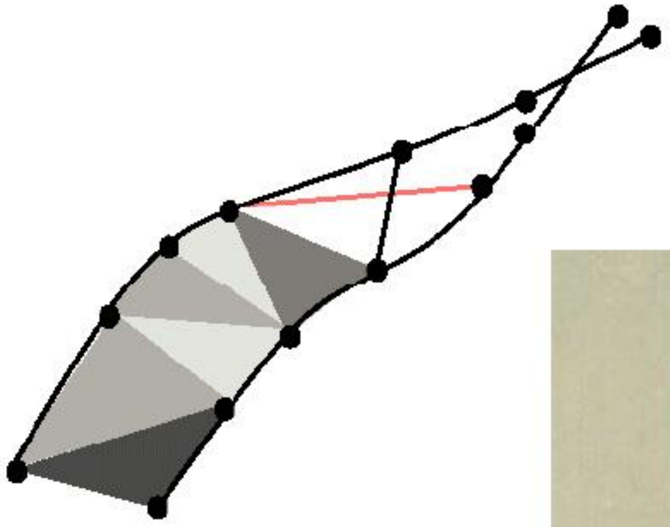
$d_{test} = 0.9 \cdot d_{sep}$

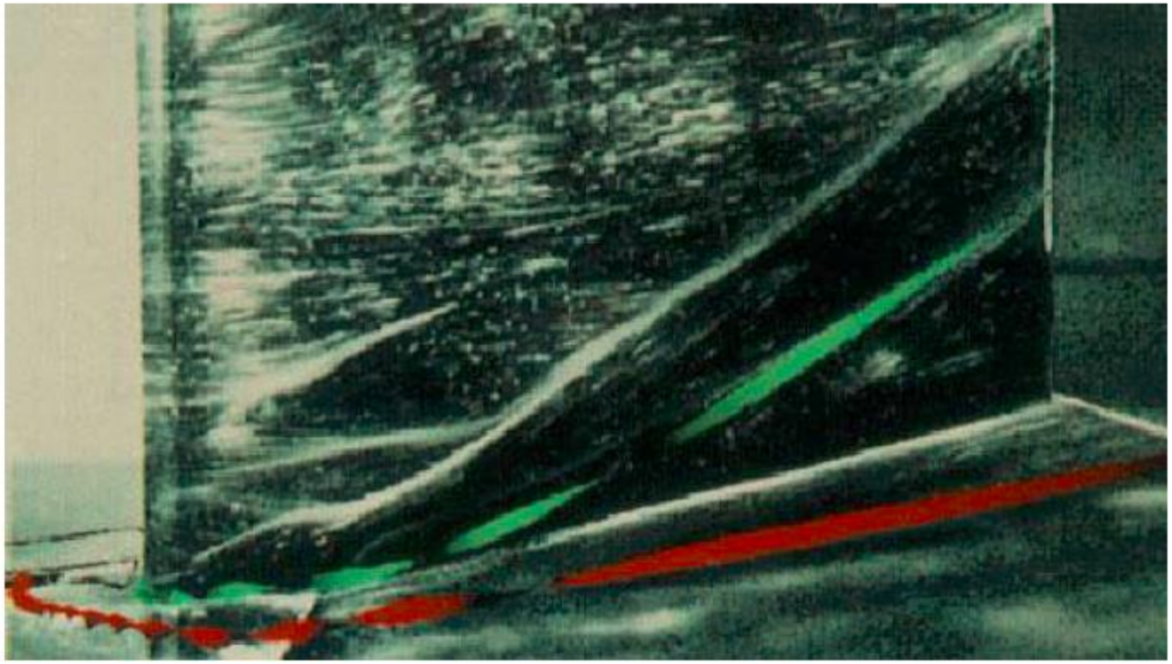


$d_{test} = 0.5 \cdot d_{sep}$



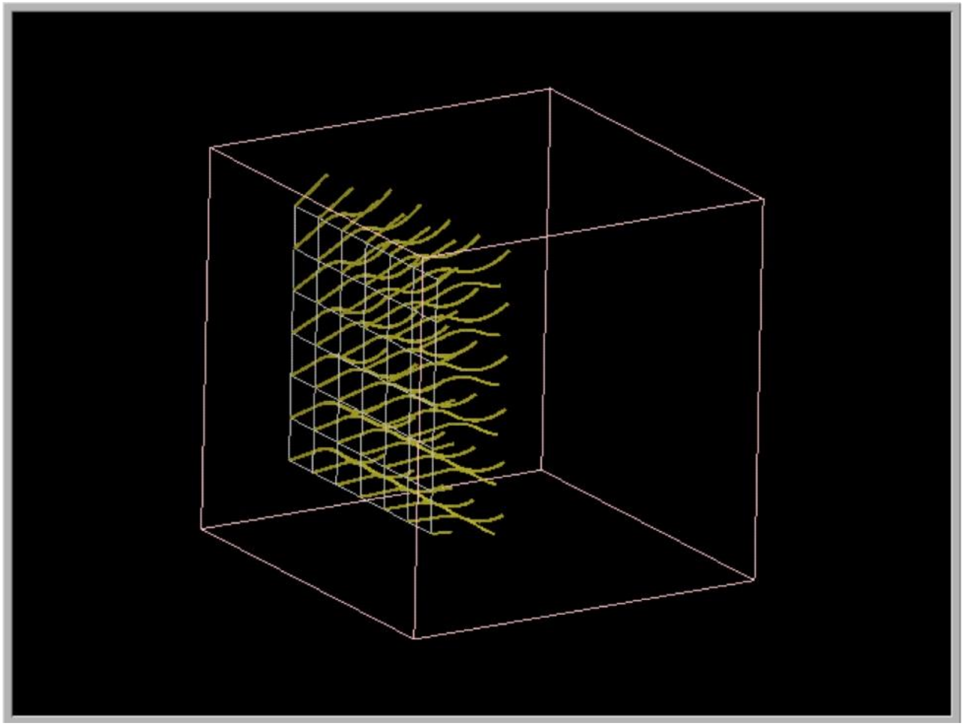
Streamribbons





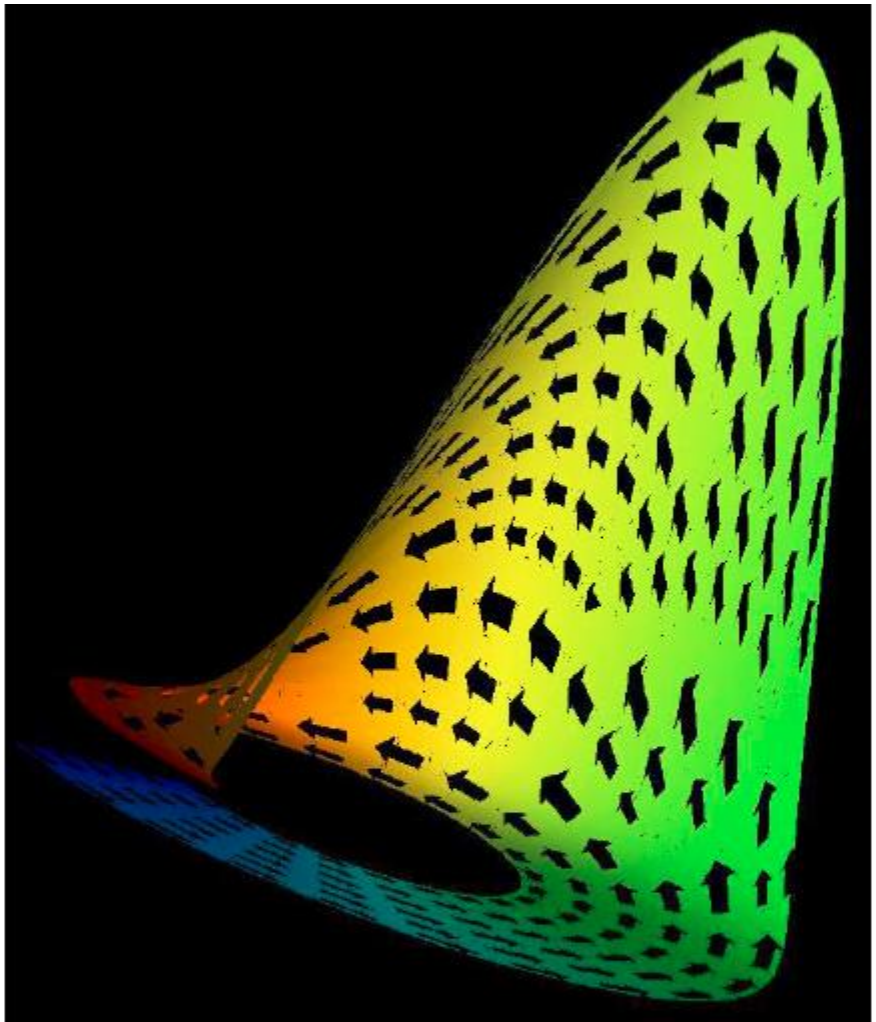
Streaklines

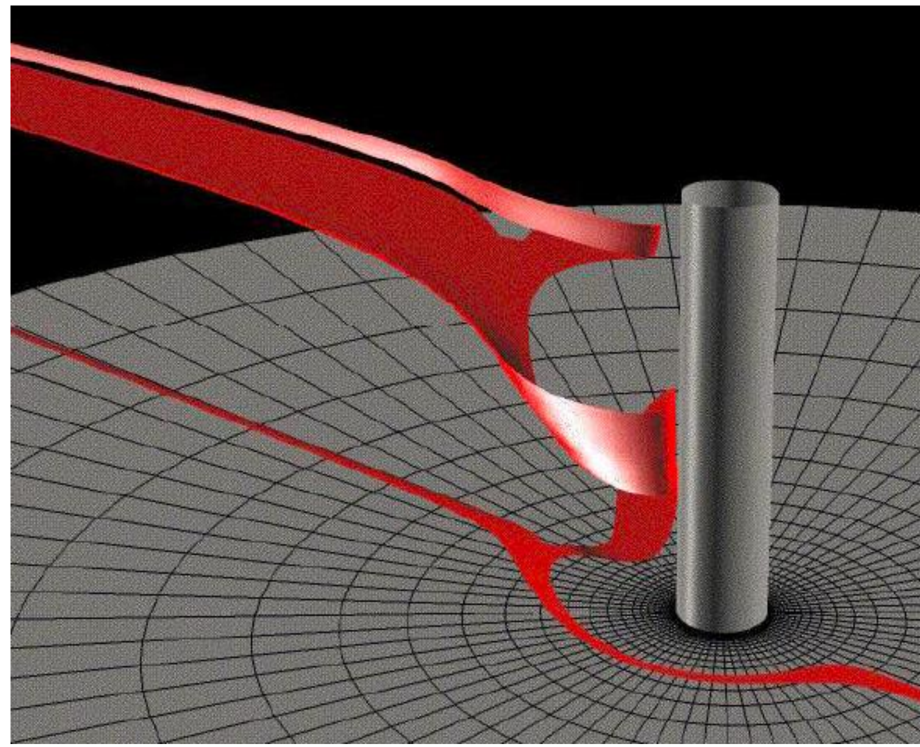
- Continuous flow of particles emitted from a discrete set of points and flowing through a field



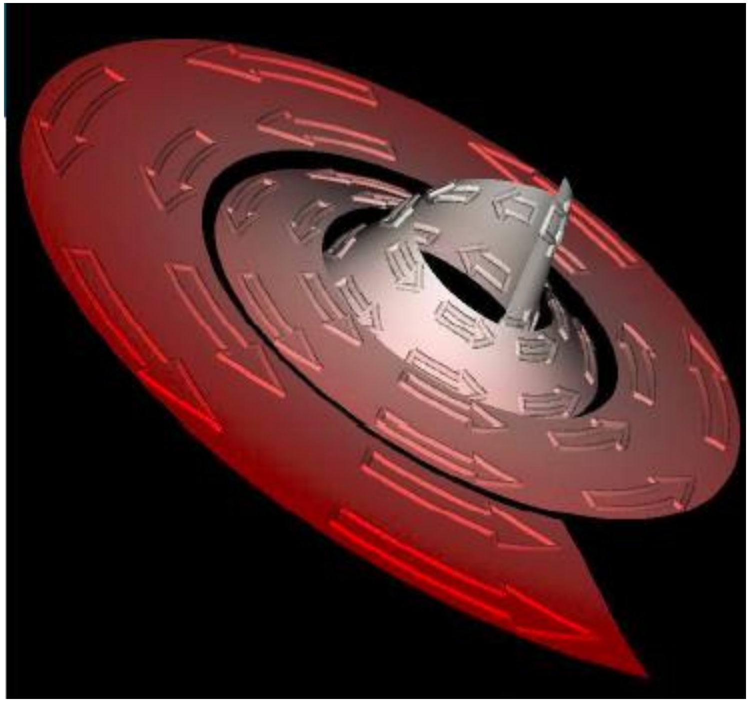
http://www.opendx.org

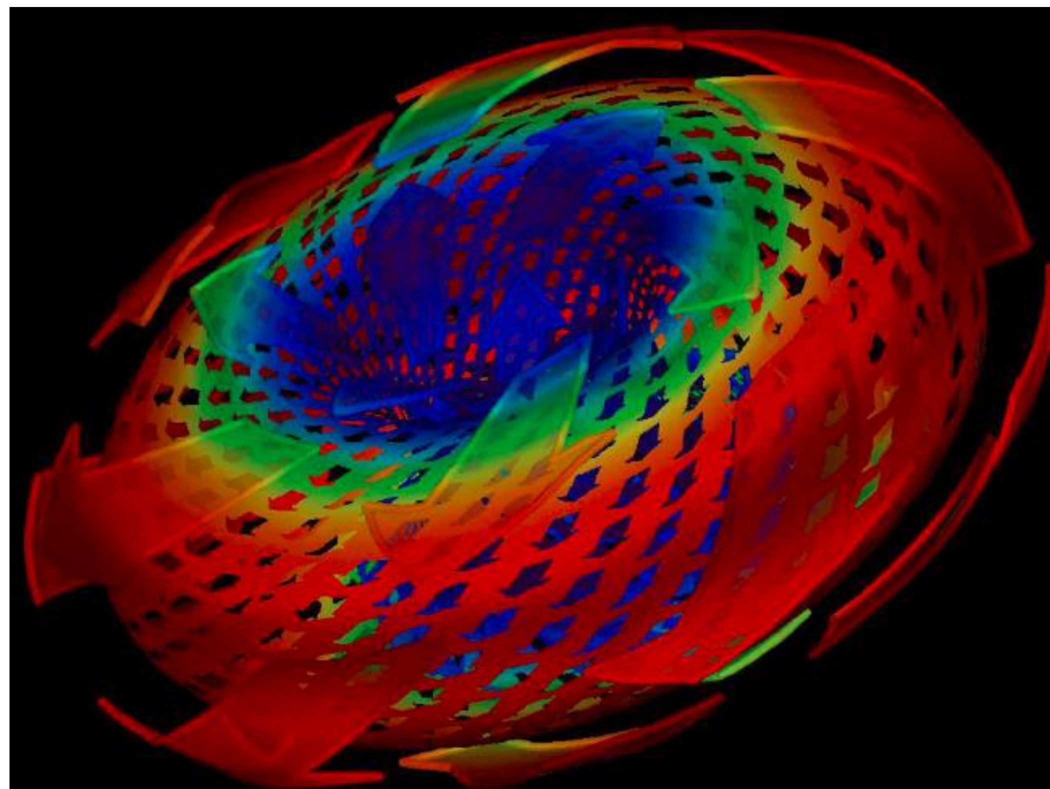
Streamsurfaces





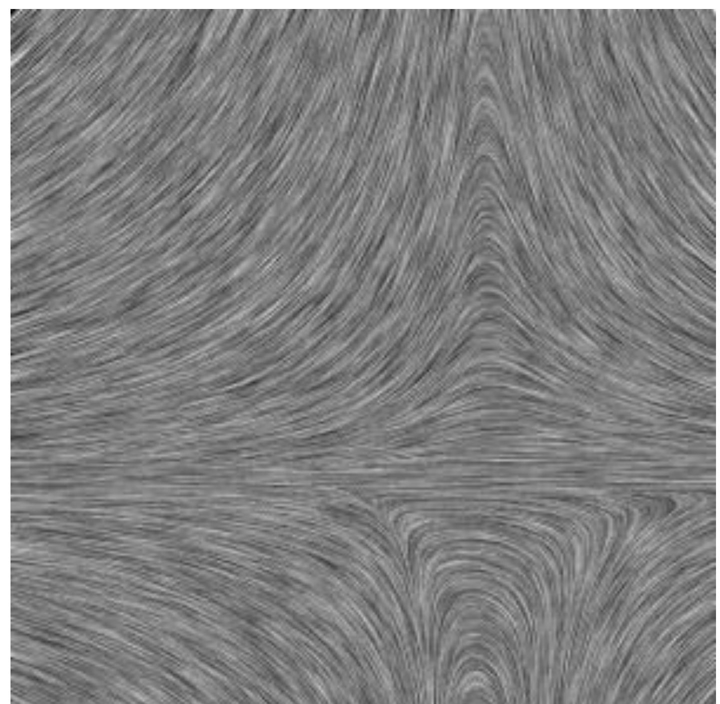
Stream Arrows





Line integral convolution (LIC)

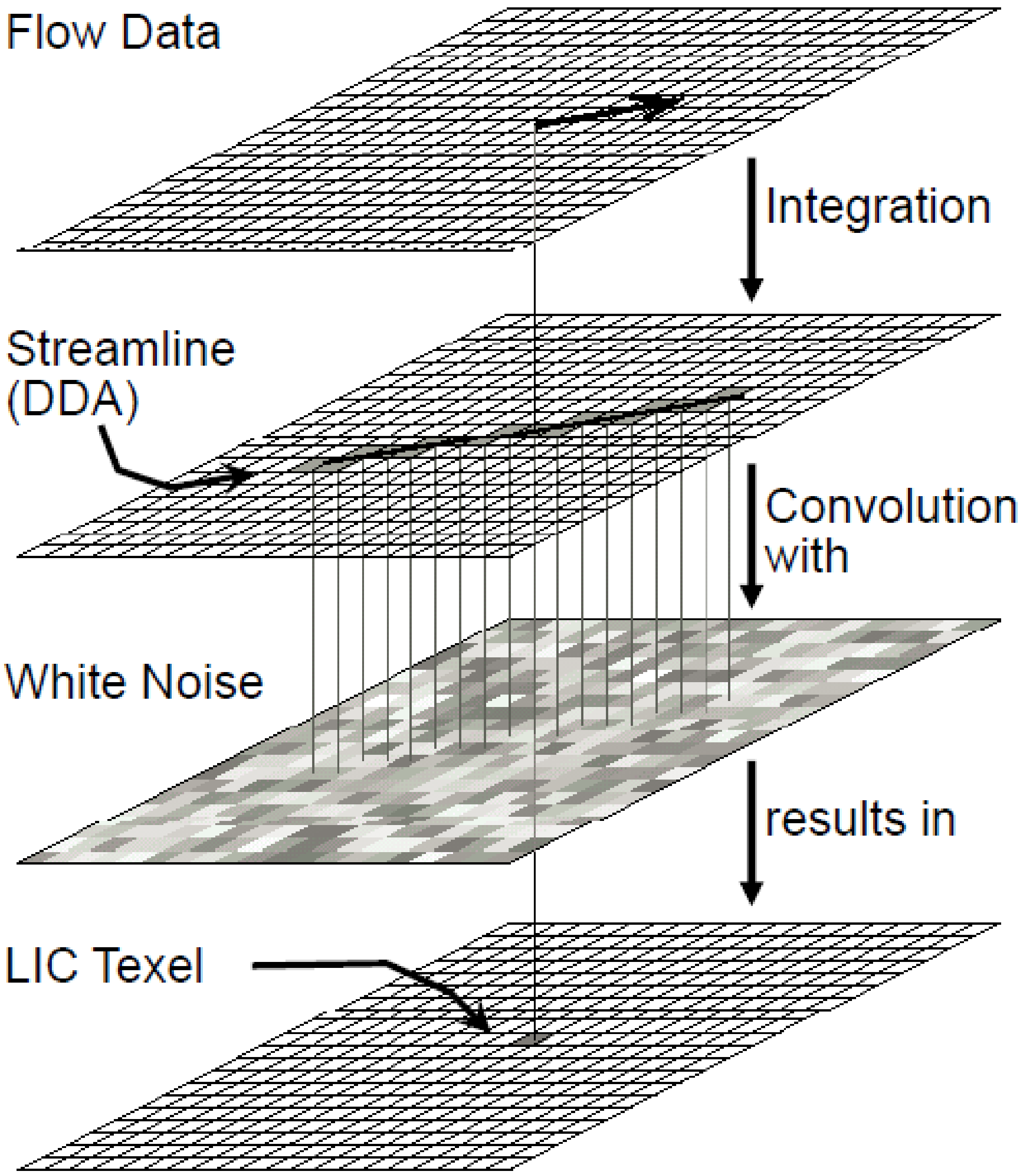
- LIC designed by Cabral a Leedom in 1993
- Random field and vector field of the same height for generating dense flow visualization



www.cg.tuwien.ac.at

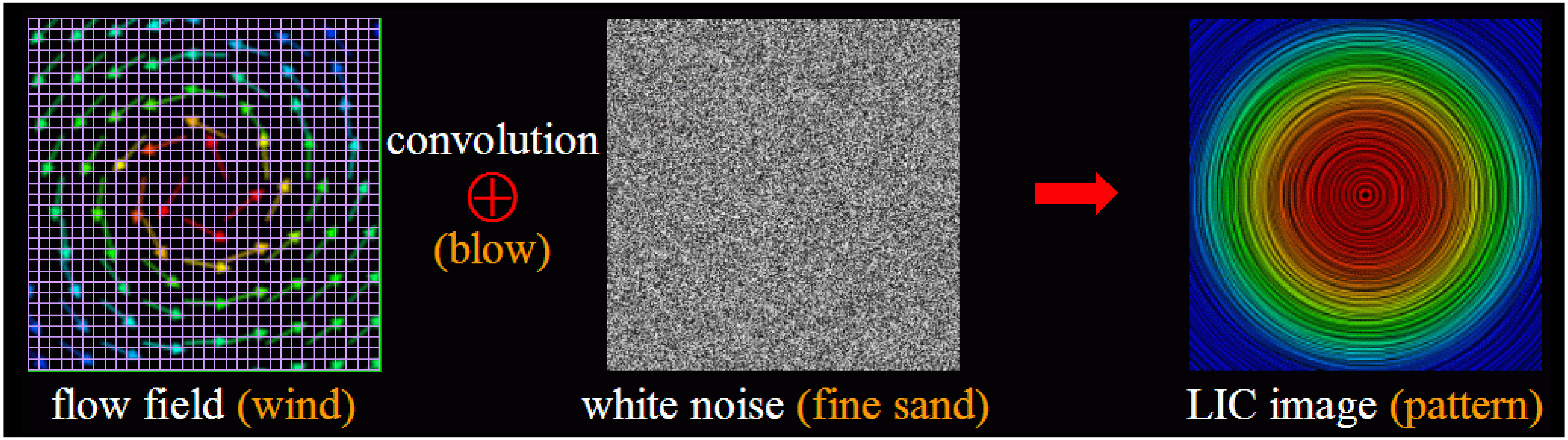
Line integral convolution (LIC)

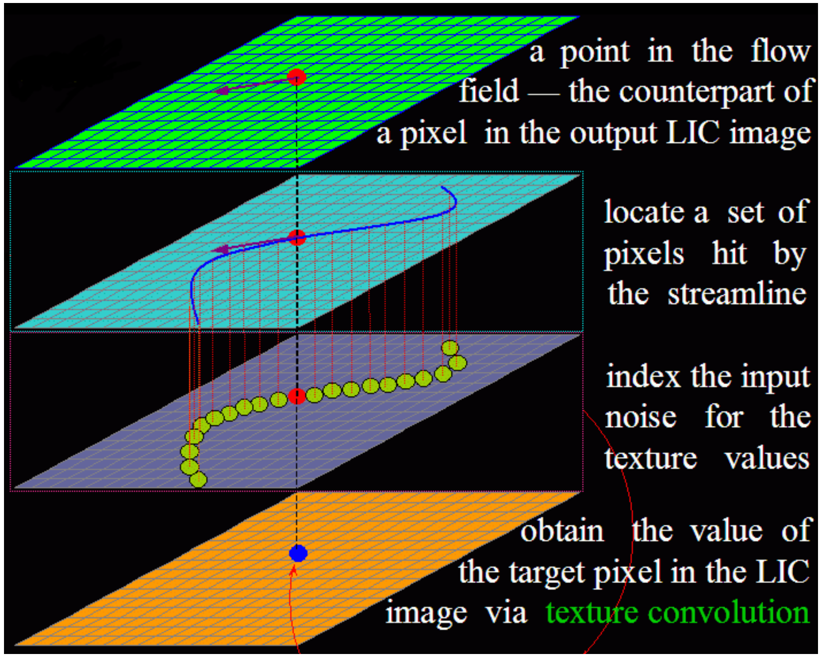
- Uses textures for showing correlation between visualization and flow
- Calculating the texture value
  - View onto streamline from a given point
  - Filtration of white noise along streamline



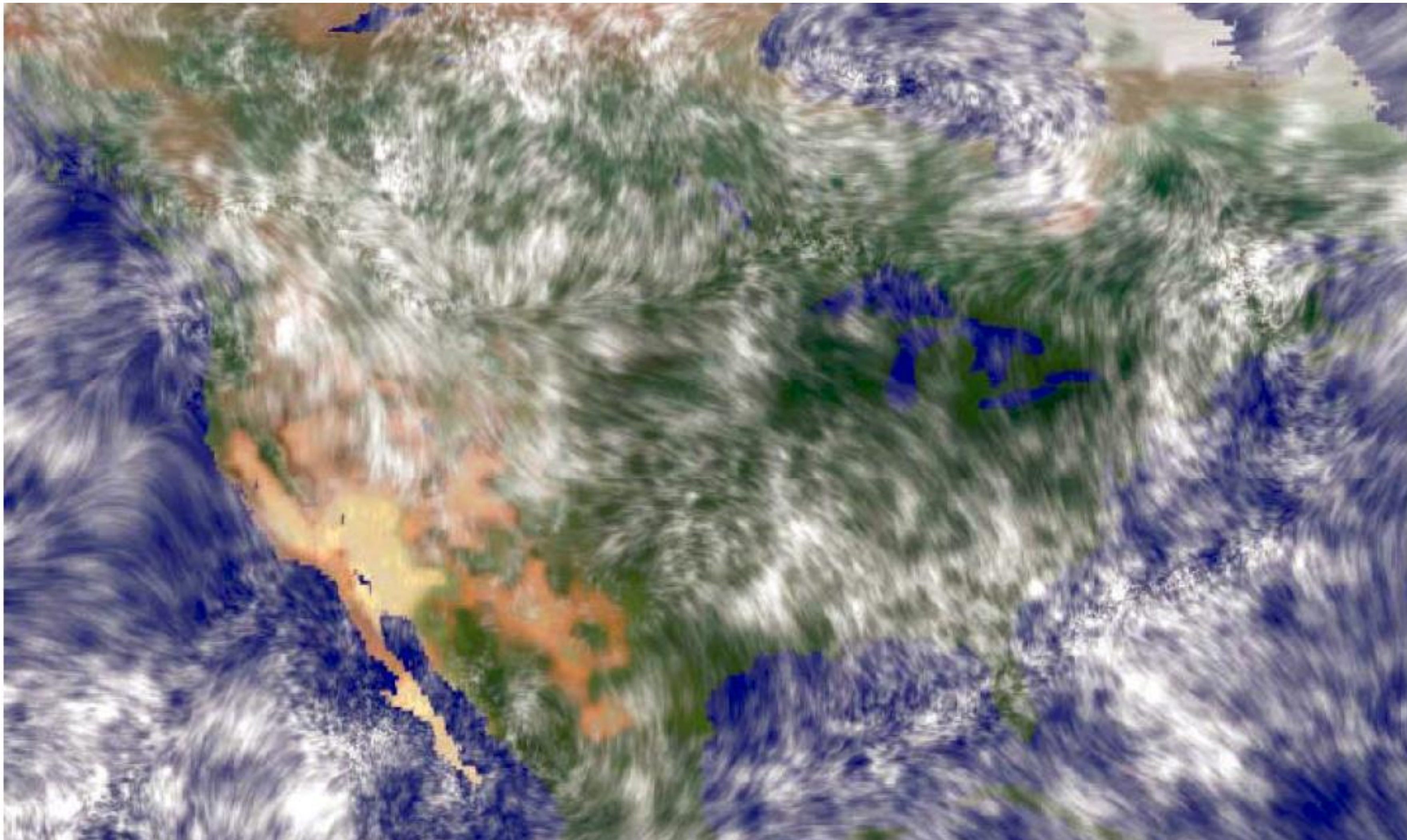
Flow Data
Integration
Streamline
(DDA)
Convolution
with
White Noise
results in
LIC Texel

LIC

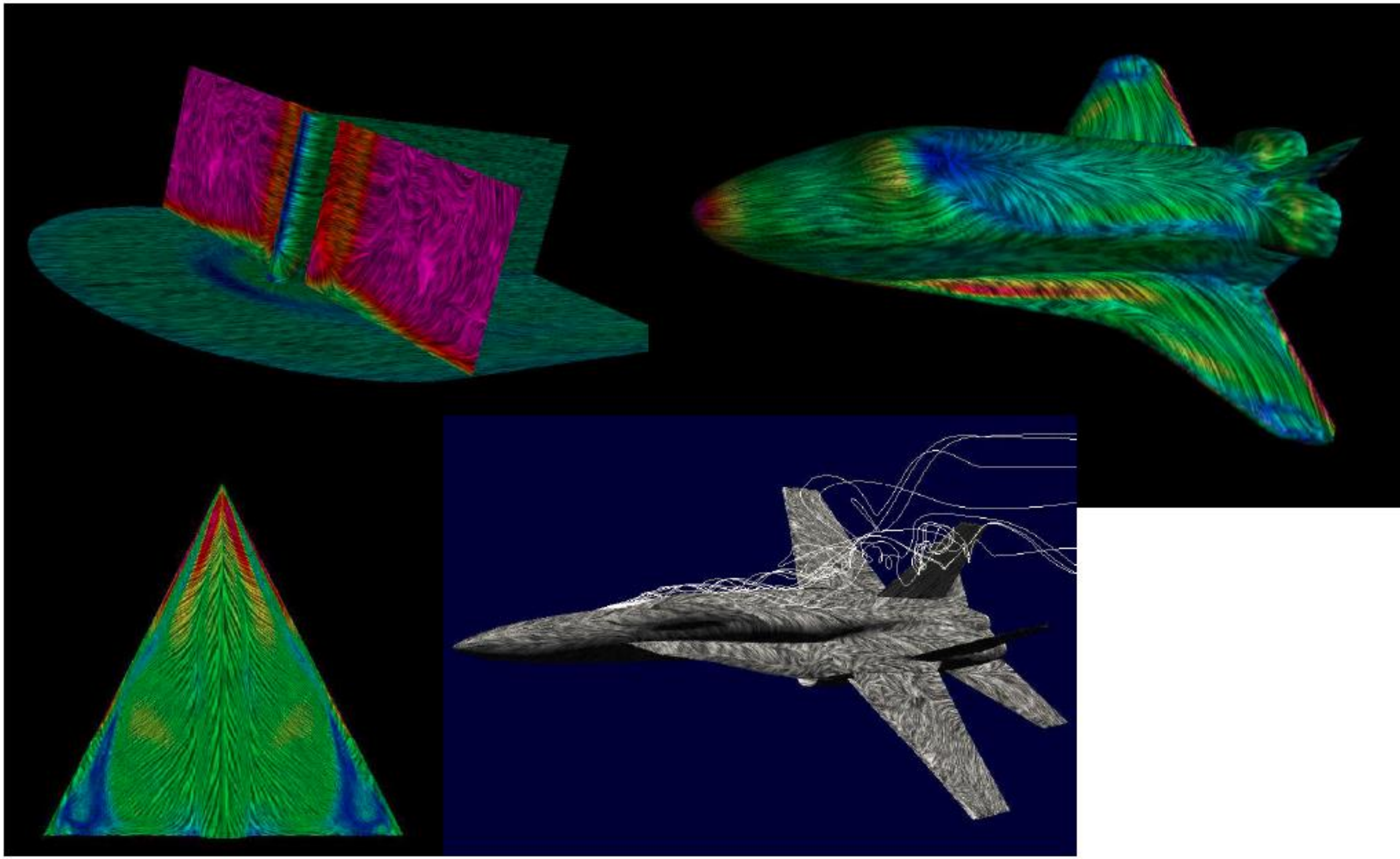

convolution
(blow)
flow field (wind)
white noise (fine sand)
LIC image (pattern)


a point in the flow field — the counterpart of a pixel in the output LIC image
locate a set of pixels hit by the streamline
index the input noise for the texture values
obtain the value of the target pixel in the LIC image via texture convolution

LIC examples

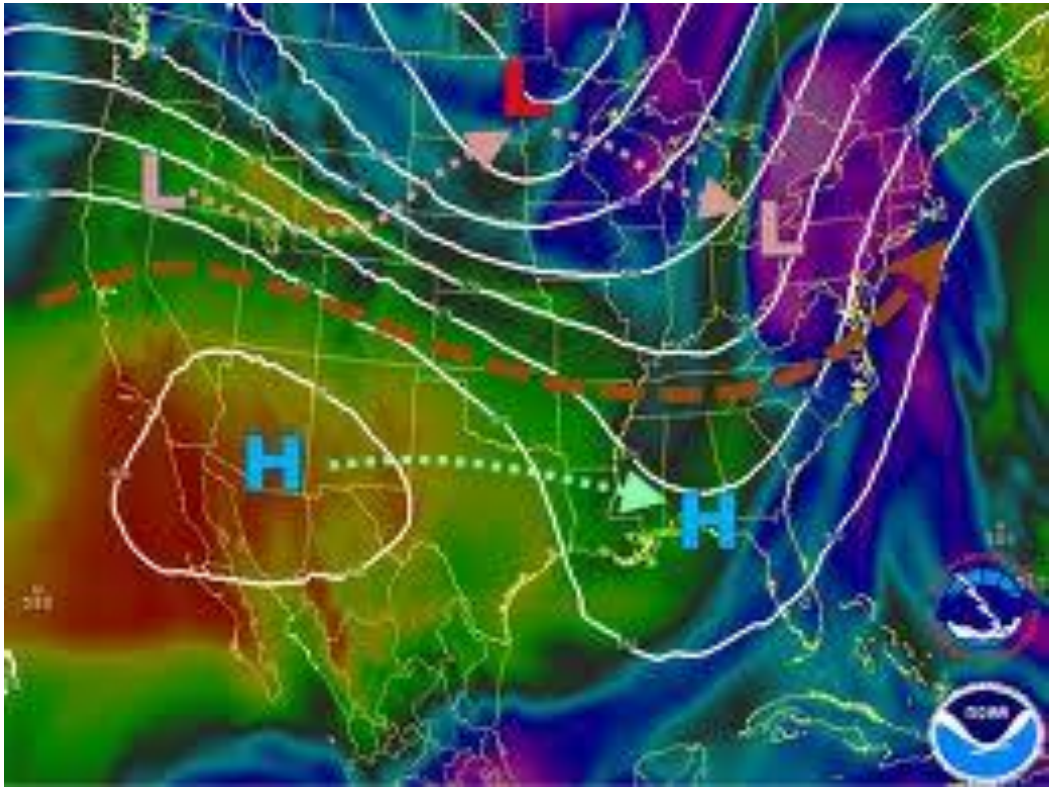


LIC – mapping onto surface



Combined techniques

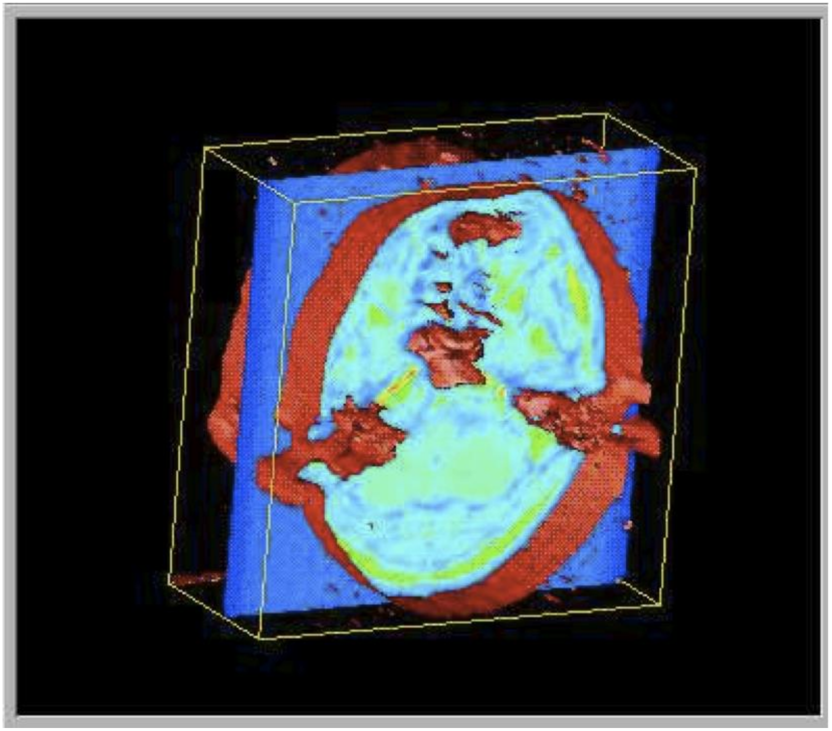
- Combination of techniques enables to highlight their strong points



www.srh.noaa.gov

Slices combined with isosurfaces

- Isosurface of medical data in combination with orthogonal slicing
- Video



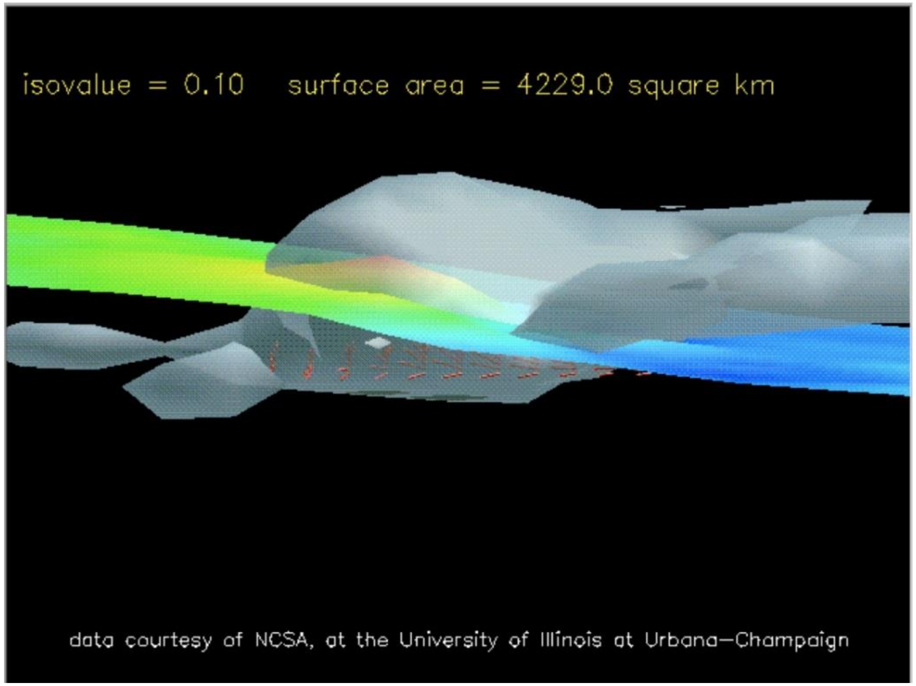
http://www.opendx.org

Slices combined with isosurfaces

- Design of this type of visualization should take into account:
  - Do not support fast changes of isosurface values
  - User-driven position and orientation of slices
  - User-driven position and and orientation of camera
  - Color assignment
  - Hiding individual visualization components or making them transparent

Combining isosurfaces and pictograms

- Isosurfaces for showing details of 3D surface, pictograms for showing size or direction of change in the dataset

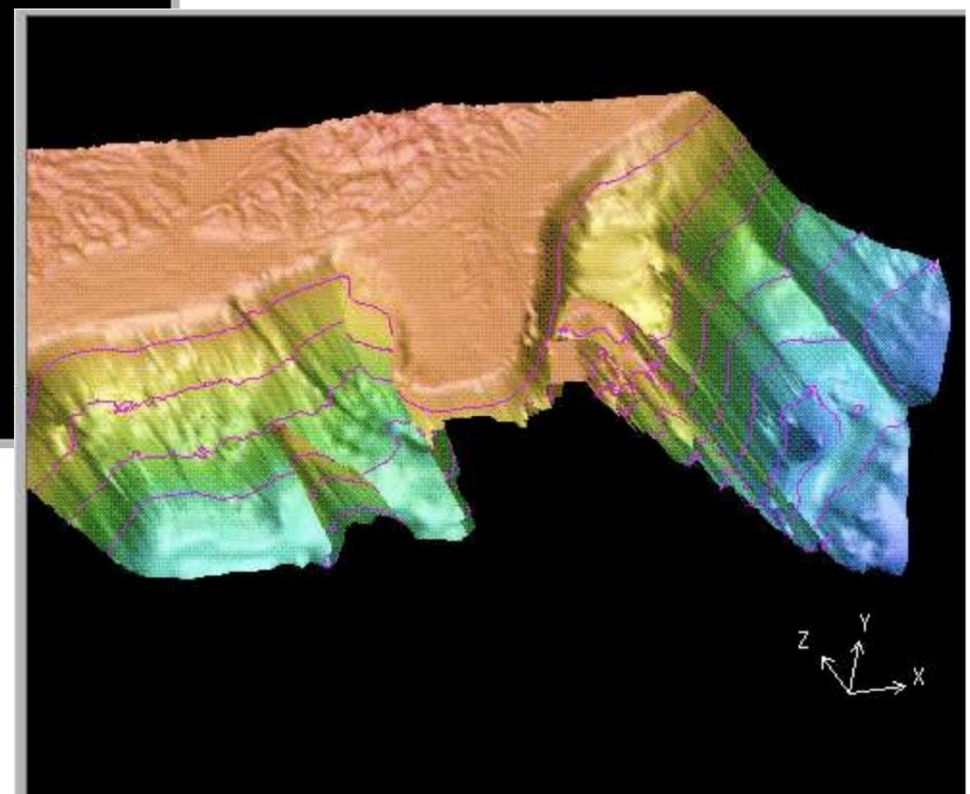


http://www.opendx.org

Combining isosurfaces and pictograms

- The following rules should be kept:
  - Interactive control of parameters
  - Changing the density of pictograms
  - Changing the size of pictograms
  - Different colors of pictograms
  - Calculating basic position of pictograms

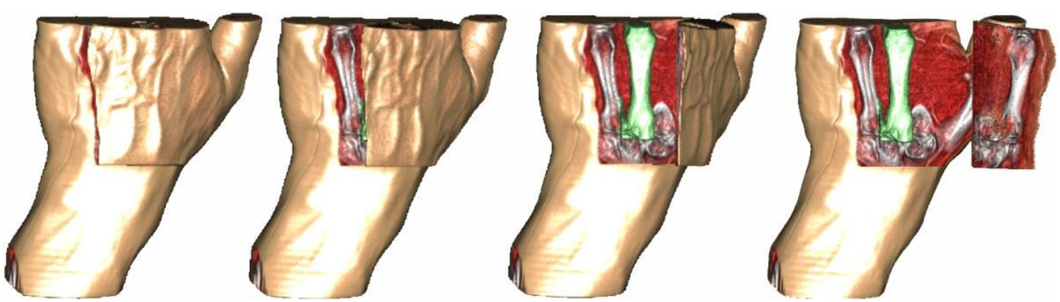
Rubber sheet + contour + color



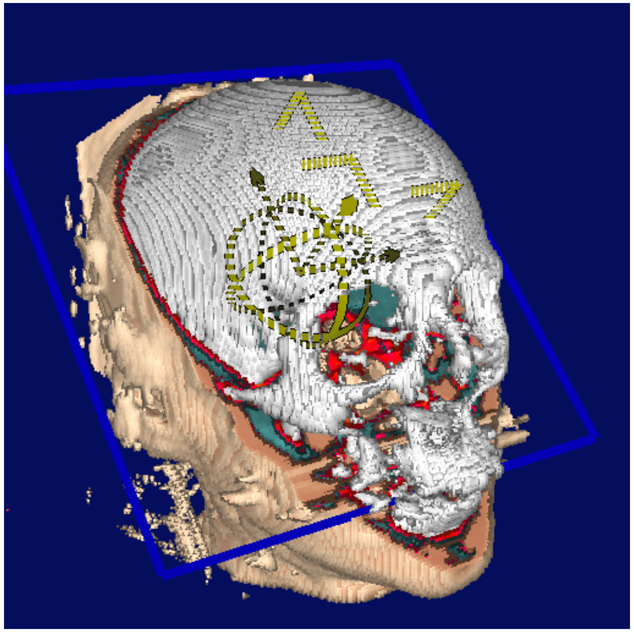
http://www.opendx.org

Summary

- Different techniques for data in different dimensions
- We need to understand pros and cons of the techniques
- Their combination is beneficial



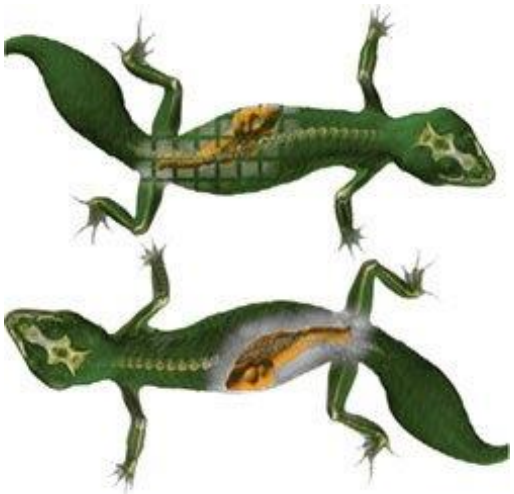
www.ii.uib.no

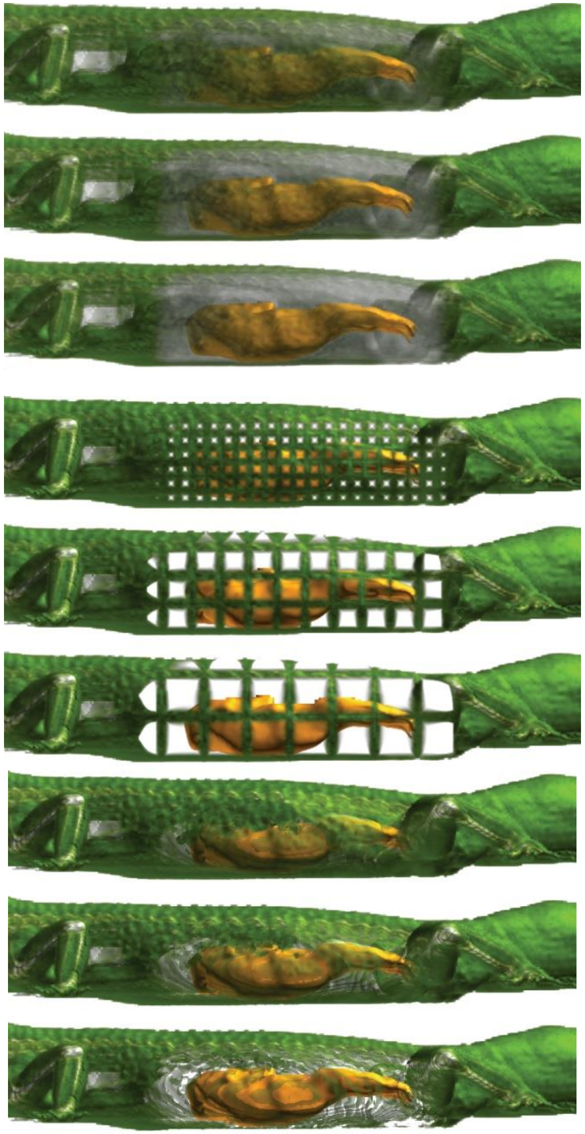


profs.etsmtl.ca

Examples

- Ivan Viola – Importance-Driven Volume Rendering

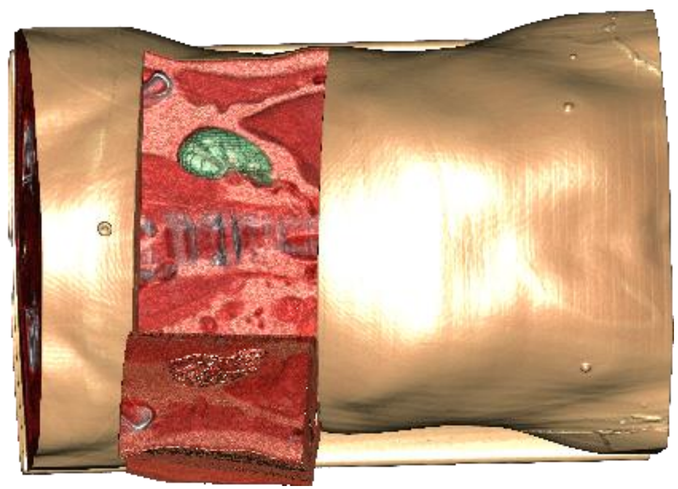
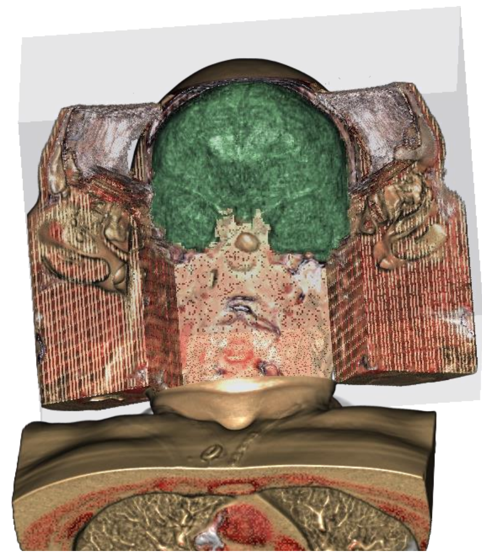
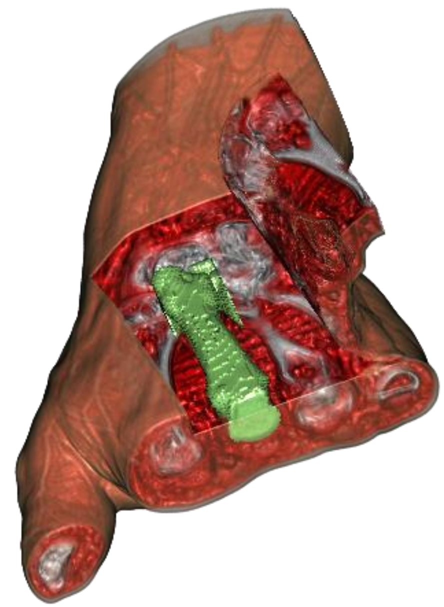




- http://www.cg.tuwien.ac.at/research/publications/2004/Viola-2004-ImpX2/

Examples

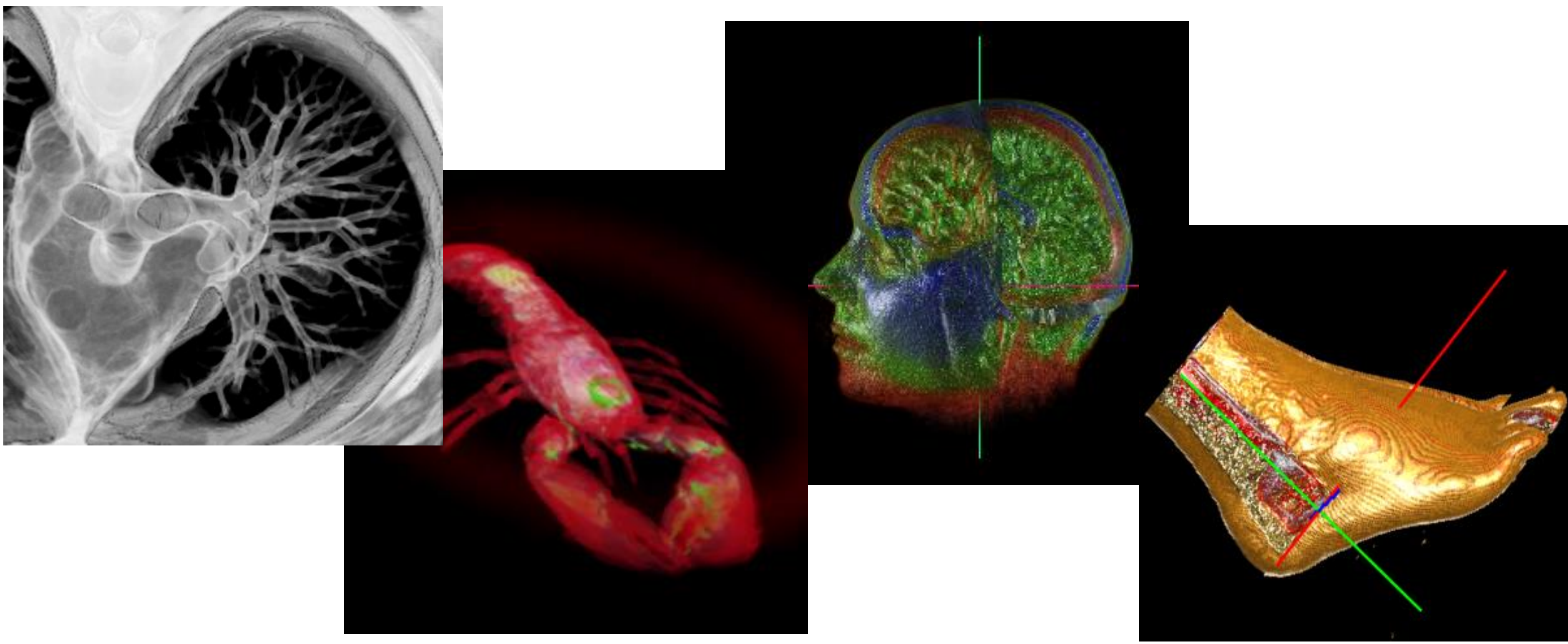
- Åsmund Birkeland - View-Dependent Peel-Away Visualization for Volumetric Data

- http://www.ii.uib.no/vis/teaching/thesis/2008-birkeland/_files/MasterThesisBirkeland2008.pdf

Examples

- Meißner et al., Volume Visualization and Volume Rendering Techniques, EUROGRAPHICS 2000



Videos

- 3D blood flow visualization

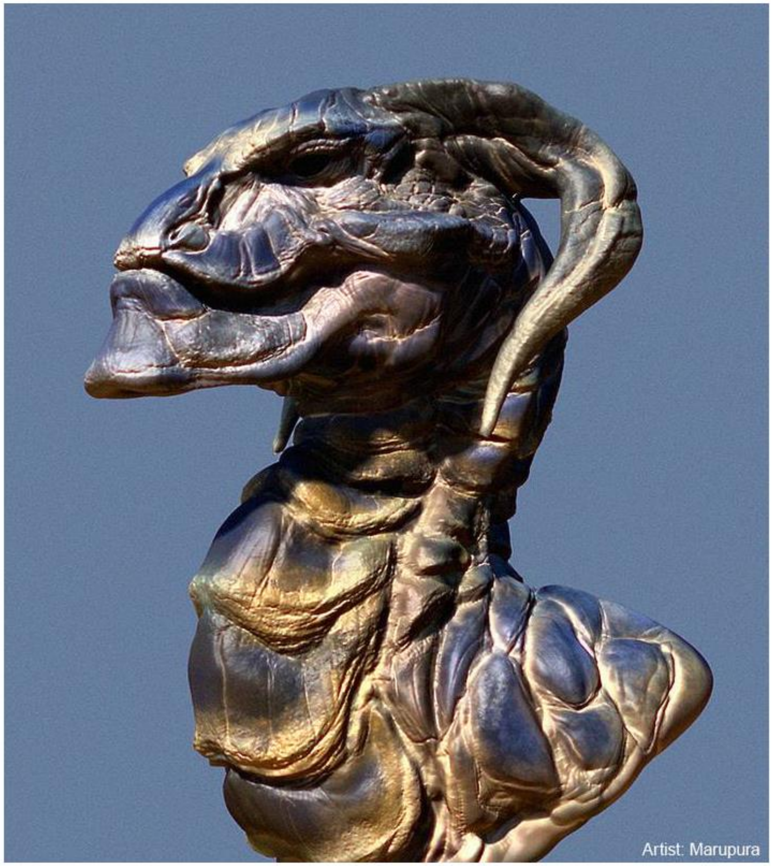
http://www.youtube.com/watch?v=jkjbLk0nSFc

- Flow visualization

http://www.youtube.com/watch?v=DOUfyDHxkYQ

Voxel modeling

- 3D-Coat modeling tool
  - Voxel-based modeling



http://3d-coat.com/voxel-sculpting/