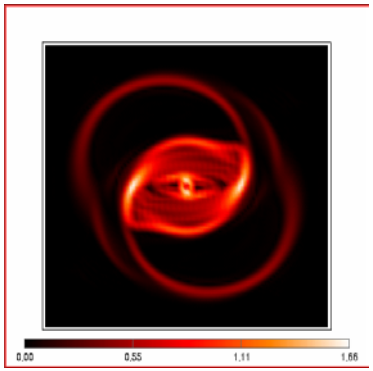
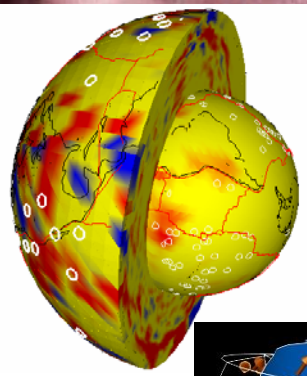
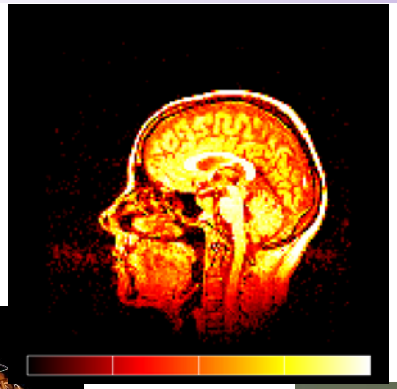
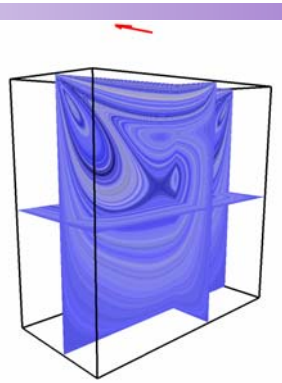
Manchester Computing

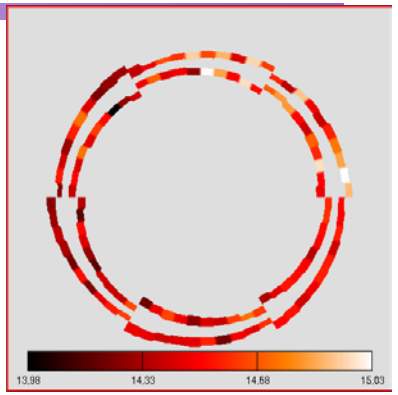
Visualization

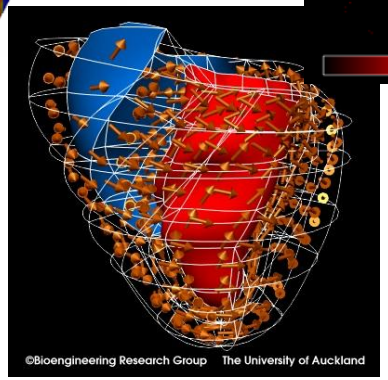












Joanna Leng

MRCCS/NSF Summer School 2003
High Performance Computing in FEA
University of Manchester

THE UNIVERSITY of MANCHESTER

What is Visualization?

- It uses computer graphics technology to make digital data easier to understand.
- Historically visualization has been divided into three disciplines:
  - Scientific visualization
  - Data visualization
  - Information visualization (data mining)

Why Visualize?

- **Because…….**

it allows you to do something that you could not do any other way

What Does Visualization Let You Do

- Speeds up knowledge discovery.
- Allows the scientist or engineer to understand their own analyses and possibly debug their simulation. Here we want to be able to interactively explore all the data even if it looks messy.
- Allows dissemination and communication with others - particularly non-experts. For them pretty colours, photo-realistic rendering, stereoscopy count for a lot.
- What you plot depends on the media, the reason for analysis and who the target audience is.

Why Not Visualize?

- **Because…..**

  it disrupts your current working practise

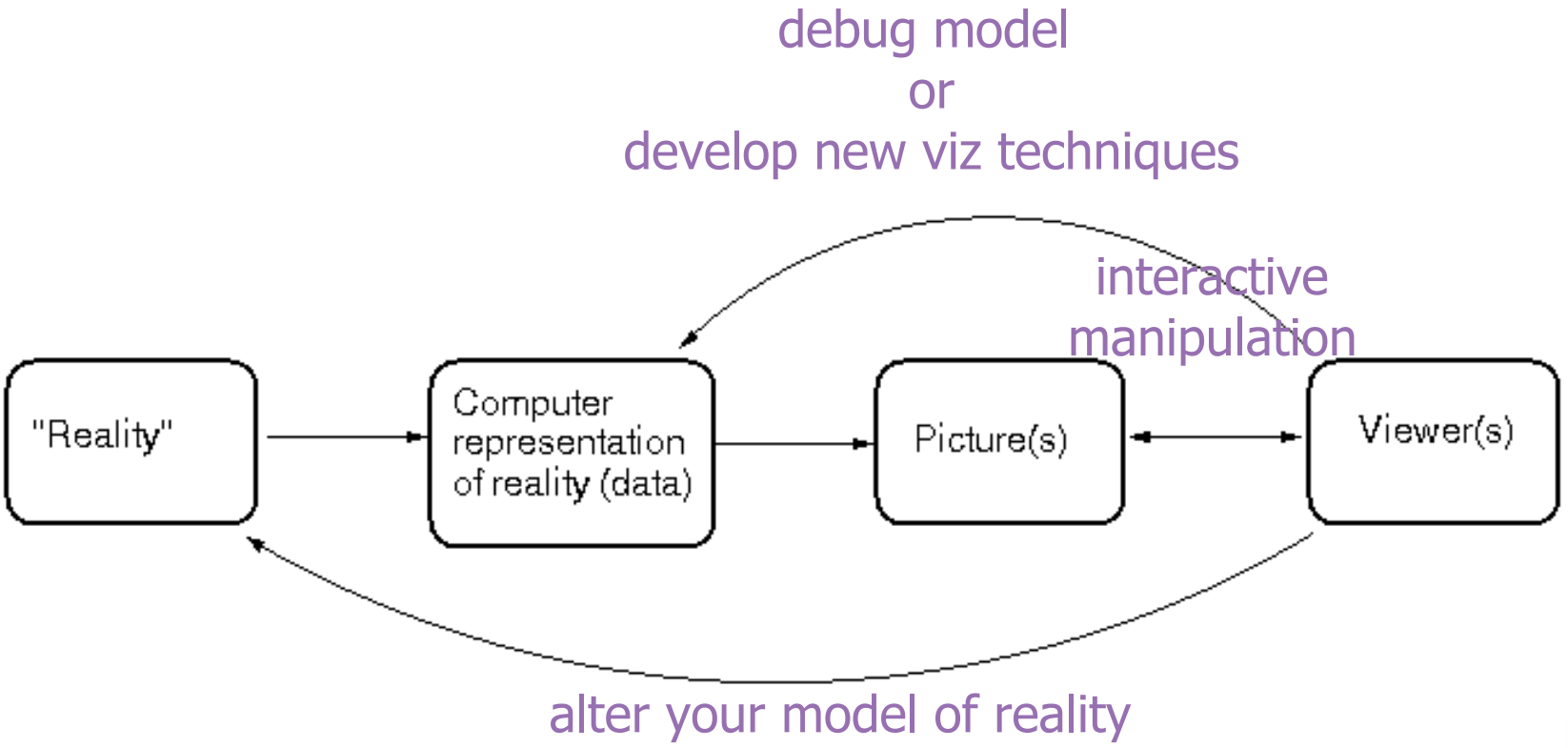
- **Because…..**

  you do not know how

  you do not know which package to use

The General Visualization Pipeline

Designed for scientific visualization but the pipeline is the basic architecture of all visualization systems.

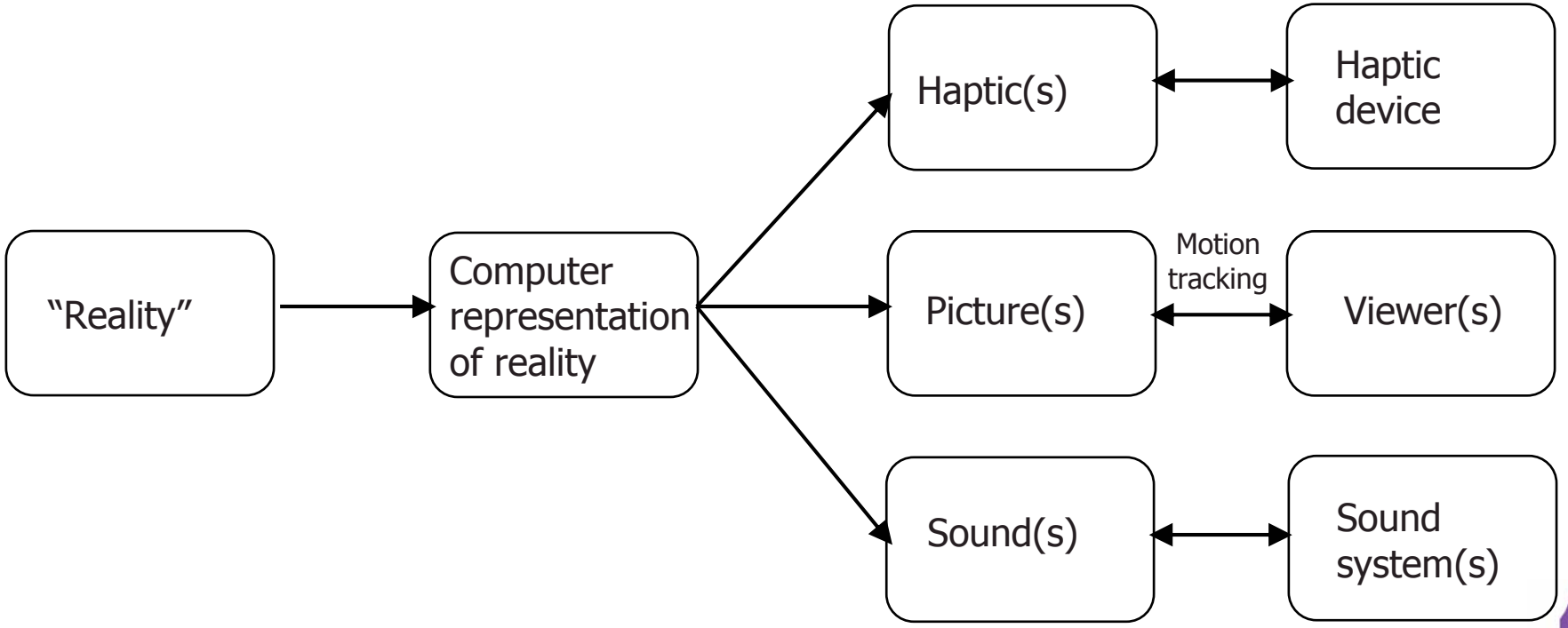


How is VR Different?

- Uses computer interface technology to enhance humans' natural capabilities.
- Encourages users to feel as if they are immersed in their data rather than passive viewers.
- Interfaces should respond in "real time" and often have feedback.
- Interfaces include:
  - Visual displays
  - Tracking systems
  - Input devices
  - Haptic devices
  - Sound systems
  - Graphics and computing hardware

Connection between VR and visualization

- Visualization and VR are not mutually exclusive
- To control more interfaces and use large data you need more processing power – typically a visual supercomputer

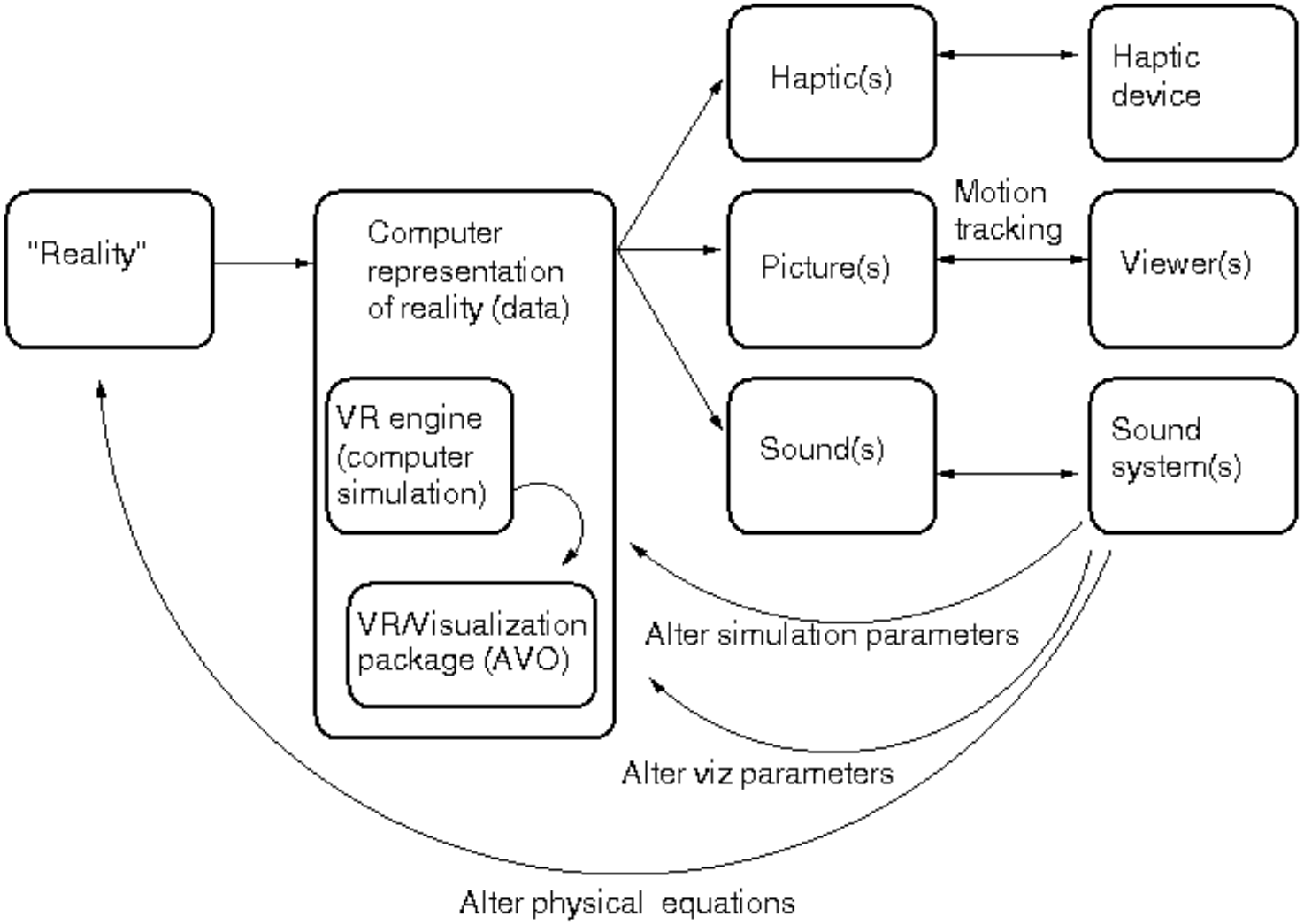


Computational Steering

- Allows the user to see what parameters there are in a simulation and alter their values interactively.

- To the computational scientist computational steering allows them to visually analyse the results of the simulation interactively, while the simulation runs. Checking for convergence/divergence is often important.
- To the VR developer computational steering allows them to use simulation code as a VR engine. Interactively here means with a reasonable frame rate (10-30 frames/s).

Connection between computational simulation, VR and visualization



Case Studies

- Validation of a theoretical model
  - Solar physics
- Validation of engineering codes
  - Large FE Simulations
- Analysis to aid debugging
  - Finite Elements in Biomechanics

Solar Physics

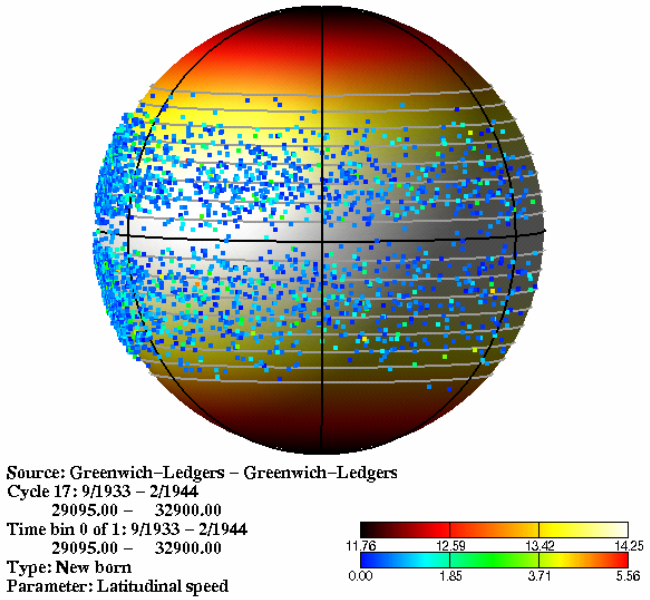
John Brooke and Dave Moss from the University of Manchester have a gyroscopic model of the internal activity in the Sun that needs validation.

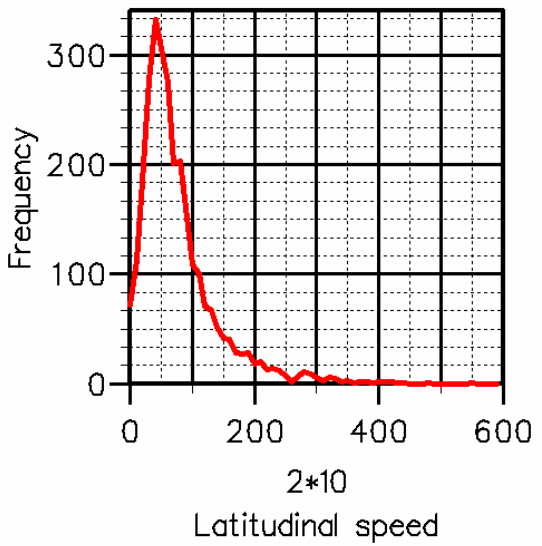
Validation is done by comparing real data to theoretical data

Real data is sunspot observations.

Theoretical data is simulation data.

1. A virtual Sun shows the effects of observational bias/error



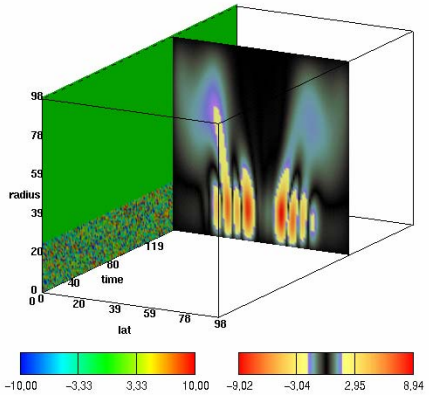


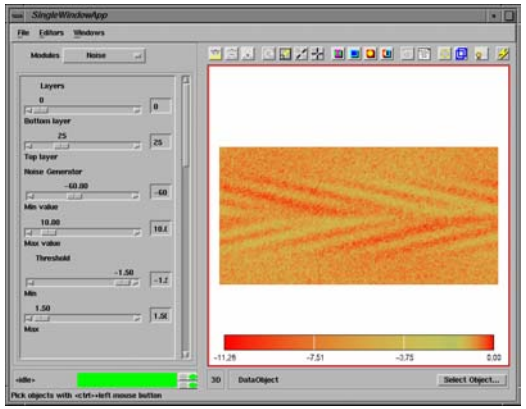
g service

THE UNIVERSITY of MANCHESTER

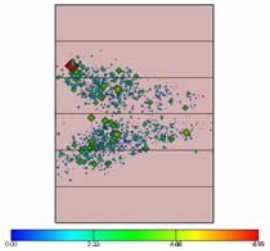
Solar Physics

1. A conversion tool puts stochastic noise in simulation data produced by Andy Phillips to manufacture hypothetical sunspots

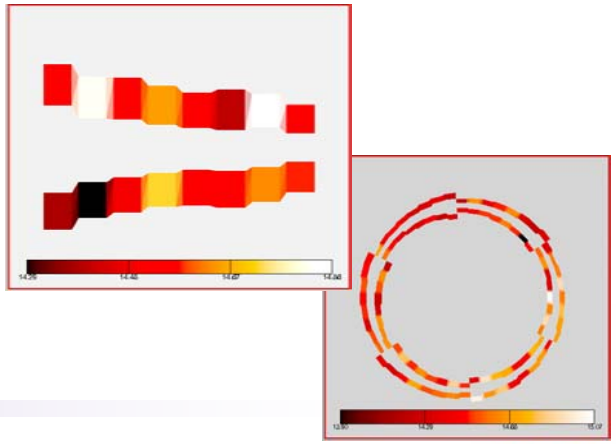




2. Analysis of sunspots observations so that real and hypothetical observations can be compared



new butterfly plots



Analysis of Large FE Simulations

- Ian Smith (UMIST) and Lee Margetts (University of Manchester) have produced a library of parallel routines for finite element analysis that are callable from FORTRAN90 (funded by EPSRC)
- Libraries cover 10 different algorithms taken from the book by Smith and Griffiths
- The codes have proven capability scaling.
- The project focused on the equation solution and parallel algorithms.
- **No time** was earmarked for **visualizing** and processing the **results**.

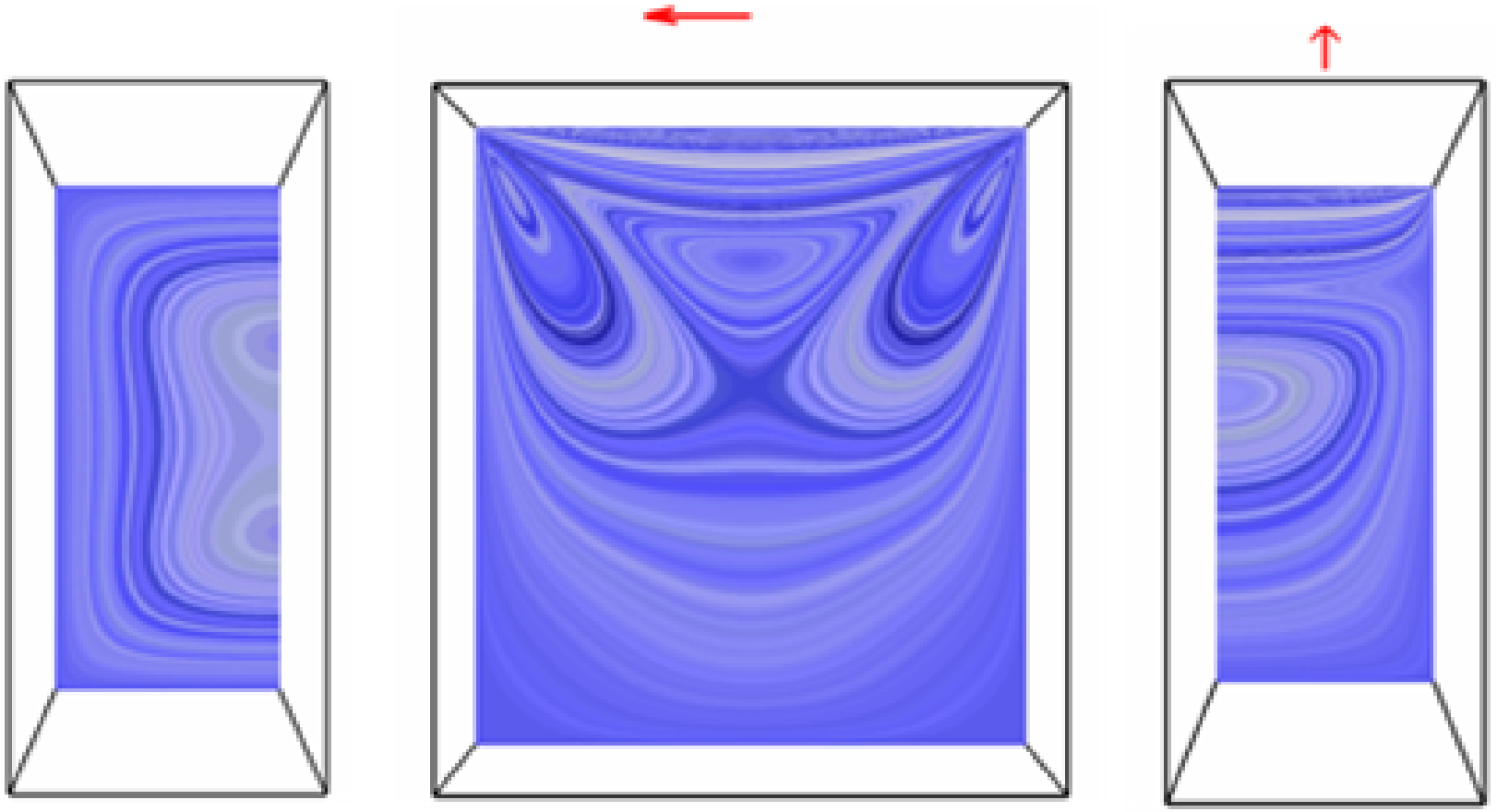
The Lid-driven Cavity

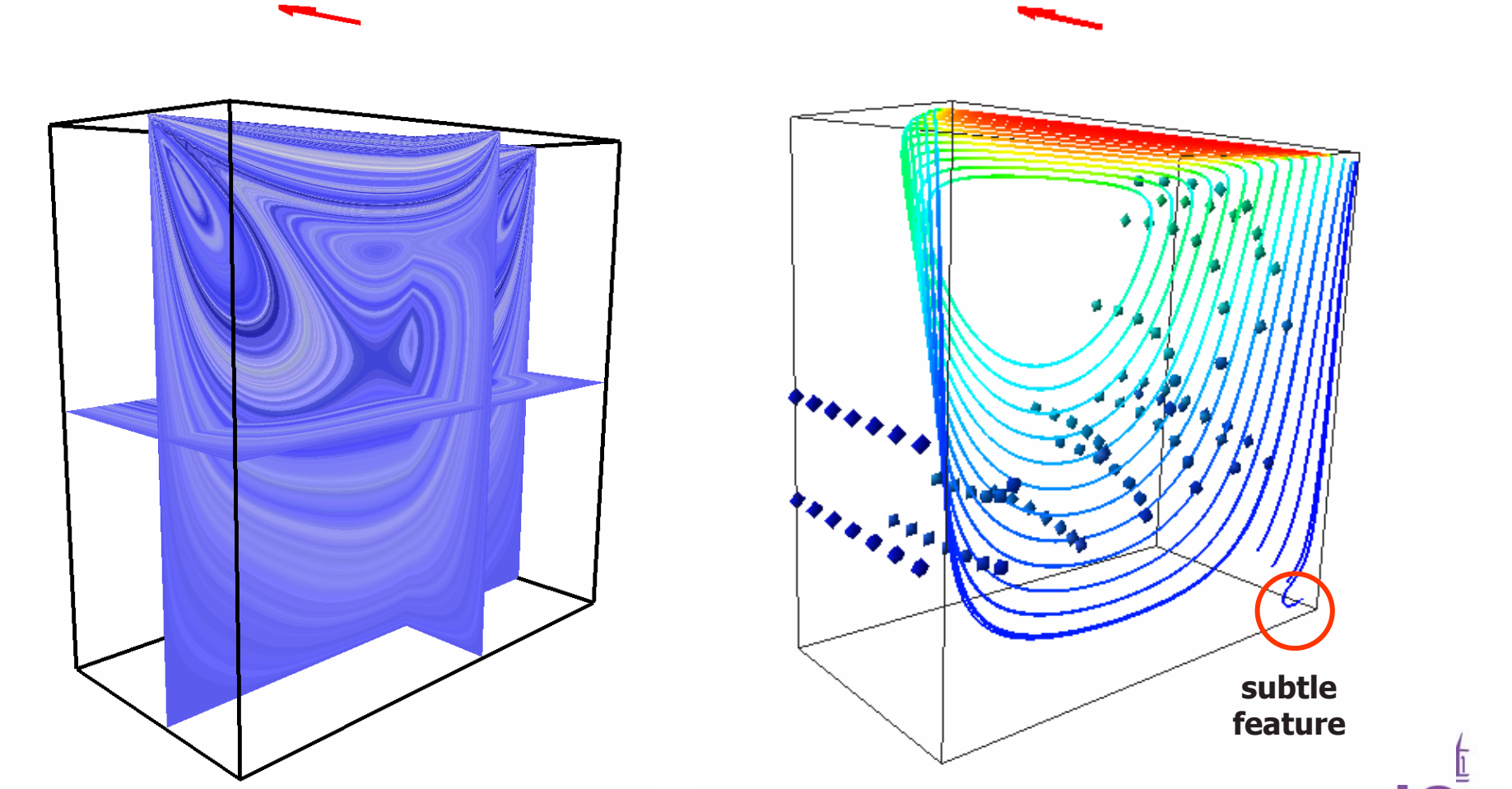
- Well documented test case for computational fluid dynamics (CFD) algorithms
- Top surface of the cavity or ‘lid’ is driven at a constant velocity
- A steady state solution is sought

- Parallel library routine was developed for the full Navier Stokes solution (with no simplifications) of fluid dynamics problems
- To prove the code was valid it was run at high resolution
  - 1 million grid points
  - 4 unknowns at each point
- Need to see the known major features of the flow
- Want to see subtle features as well

- Initial work applied to low resolution data (~5000 points)
  - Just interactive on a O2
  - Could see major features
  - Helped engineer set up high resolution runs
- High resolution data (1 million grid points)
  - Only interactive using the multipipe on a SGI Onyx 300 visualization system with 6 graphics pipes

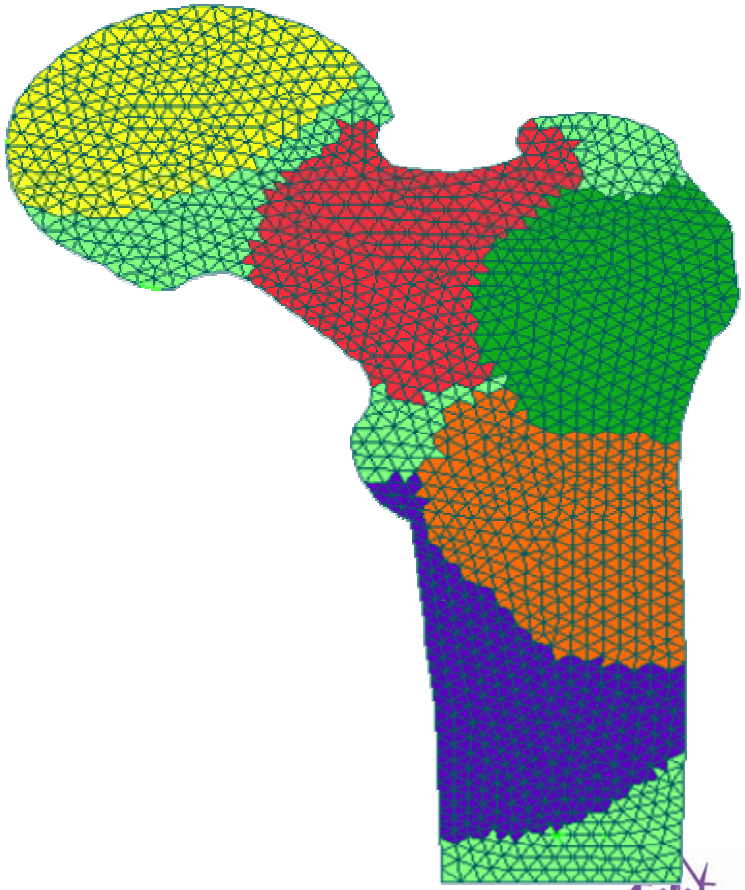
Flow Profiles




subtle
feature

Finite Elements in Biomechanics

- Dan Kidger used FE methods in biomechanics. He used visualization to display features that only matter to him the analyst.
- When simulating a real world object, artefacts are always introduced
- The picture shows a Femur FE mesh
- The dark lines show the finite element mesh.
- The colours shows the domain decomposition across multiple processors.
- Neither should need to appear on presentation to others.

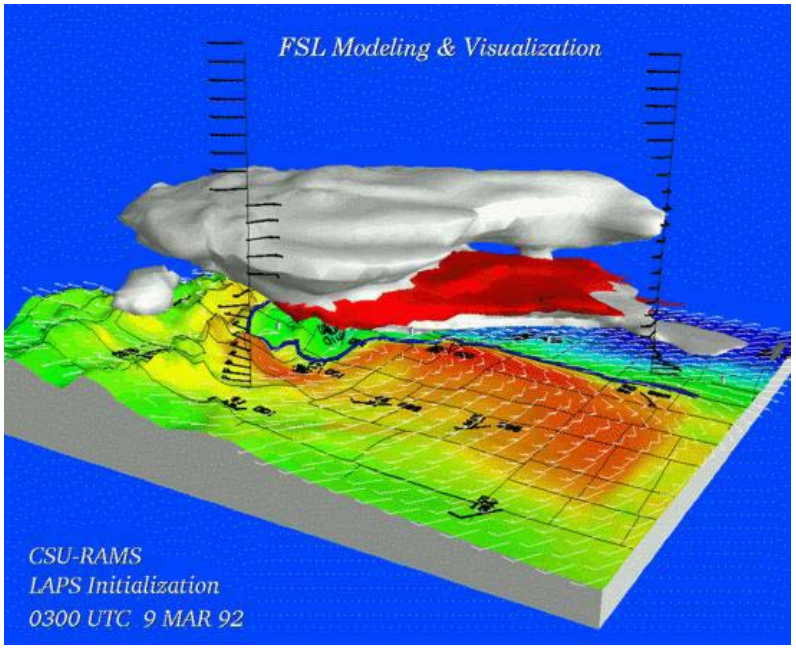


Visualization Techniques

- The big three techniques:
  - Surface rendering
  - Direct rendering
  - Cut planes
- Other important techniques:
  - Segmentation/feature extraction
  - Flow visualization
  - Use of colour

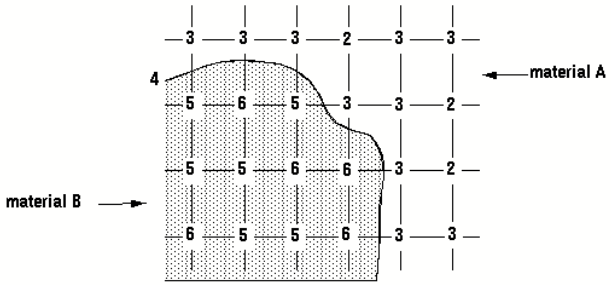
Surface Rendering

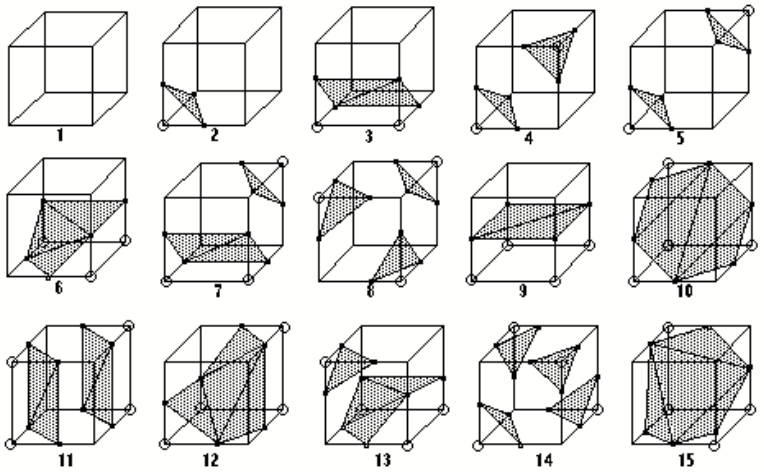
- Defined geometric shapes.
- Lights are modelled and shading is put on the objects' surfaces.
- Can extract surfaces from data – Isosurfacing.



Isosurfacing – Marching Cubes

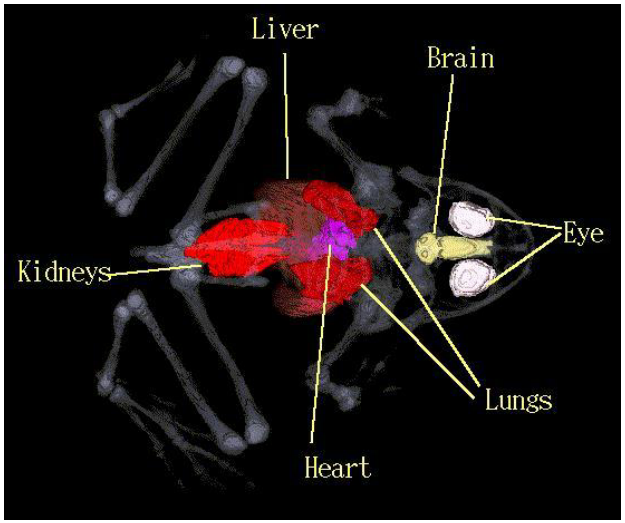
- An isosurface is a surface that goes through all points of the same value.
- Algorithm goes through each voxel and classifies if it is inside or outside.
- Puts the appropriate triangular mesh in place.

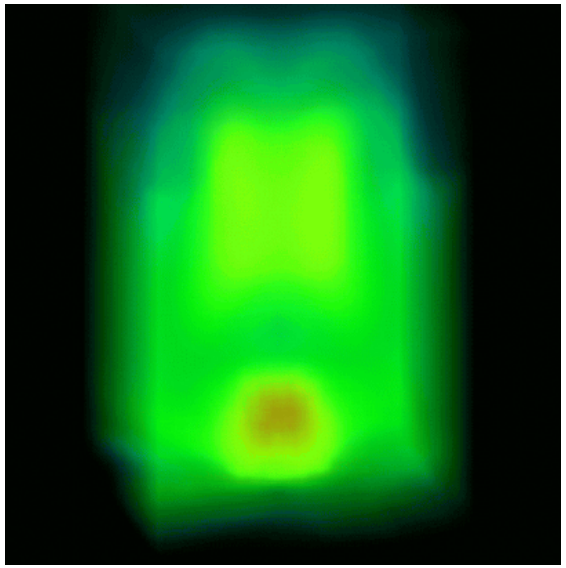




Direct Rendering (Volume Rendering)

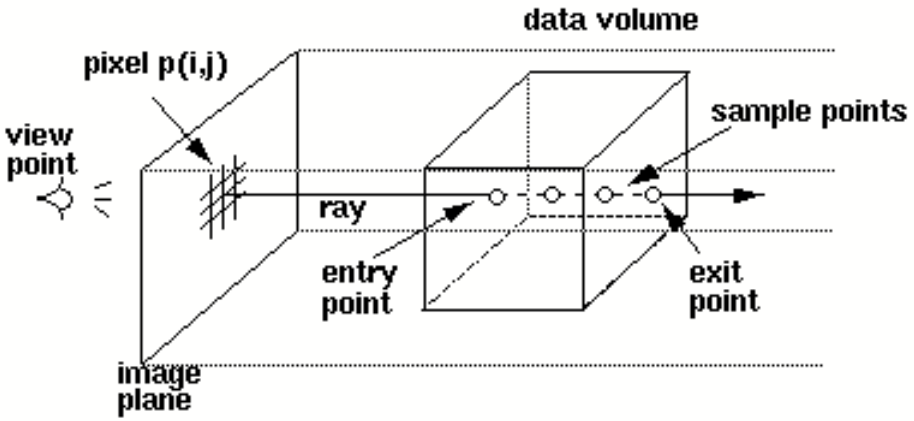
- Originally only uniform/regular array data
- Renders an object directly without calculating any intermediate geometry
- Weak and fuzzy surfaces can be rendered
- By using transparency multiple surfaces are rendered.





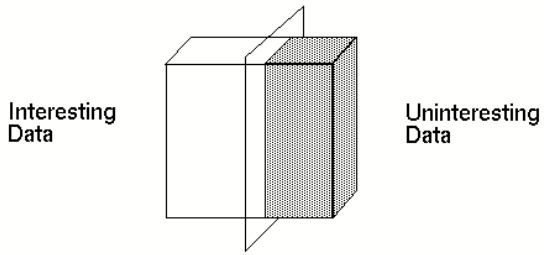
Ray Casting

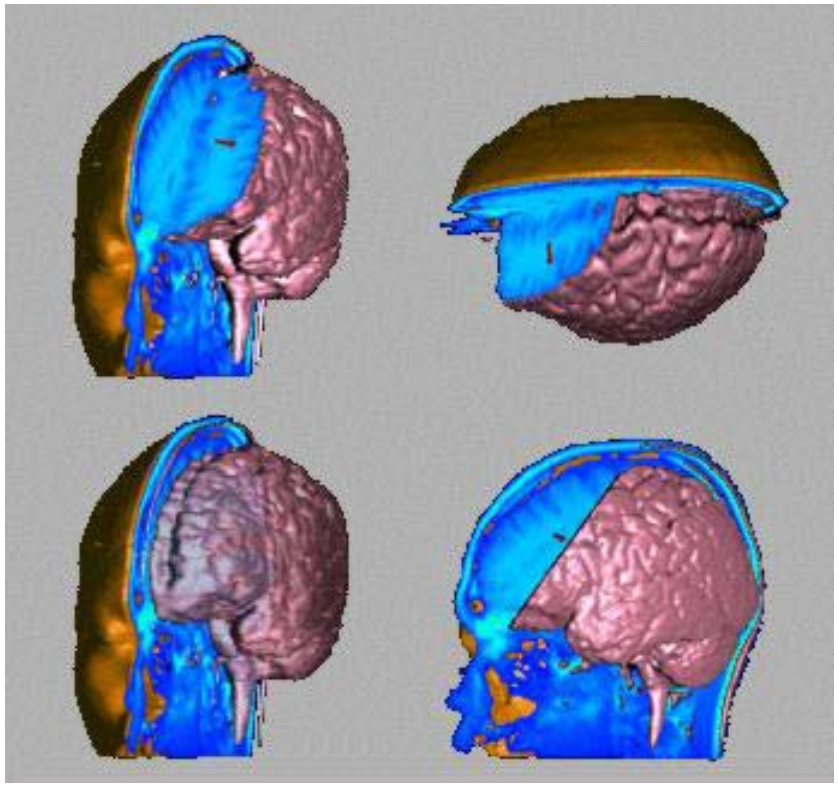
- An algorithm that produces an image similar to an X-Ray
- Cast a ray through a volume of data and calculate colour and transparency values for sampled points along the path.



Cut Planes

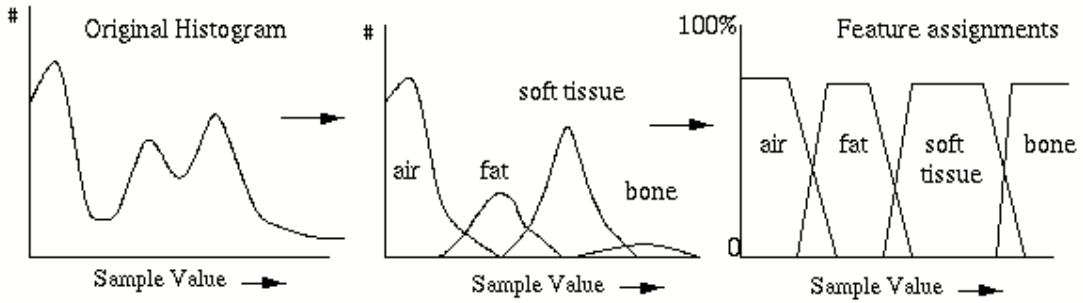
Extent Planes/ Cut Planes/ Excavate





Segmentation/Feature extraction

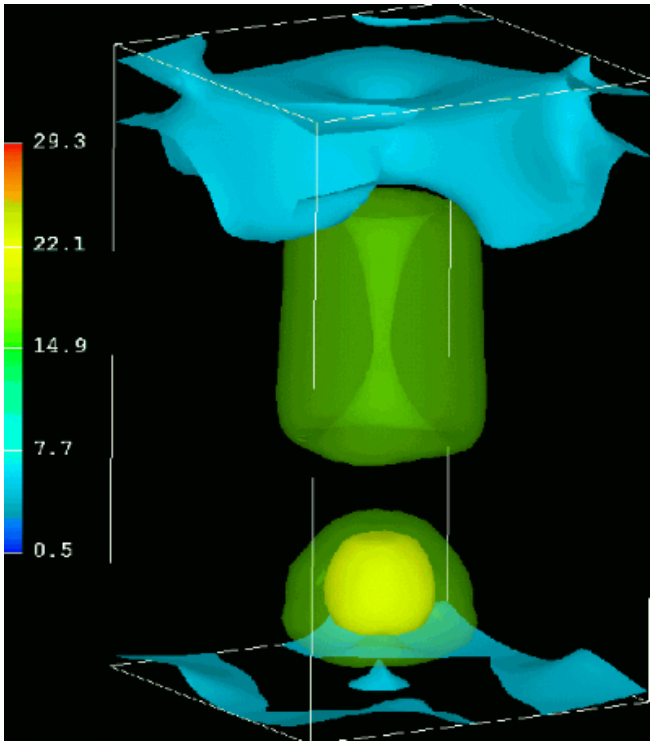
- All viz techniques can be said to extract features.
- Filtering – only looks at data values not the position of the data value.

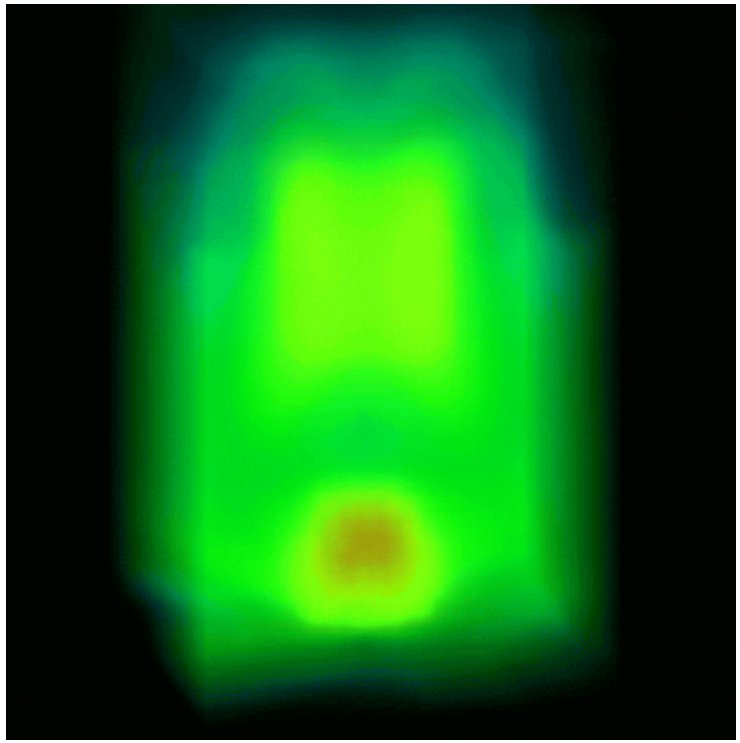


- Mapping – only looks at location (cut planes)
- Segmentation identifies a feature by location and data value (isosurfacing)

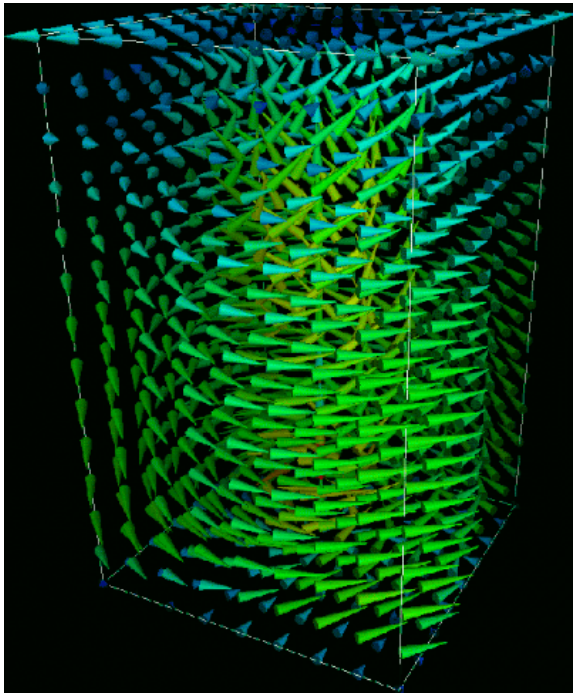
Flow Visualization

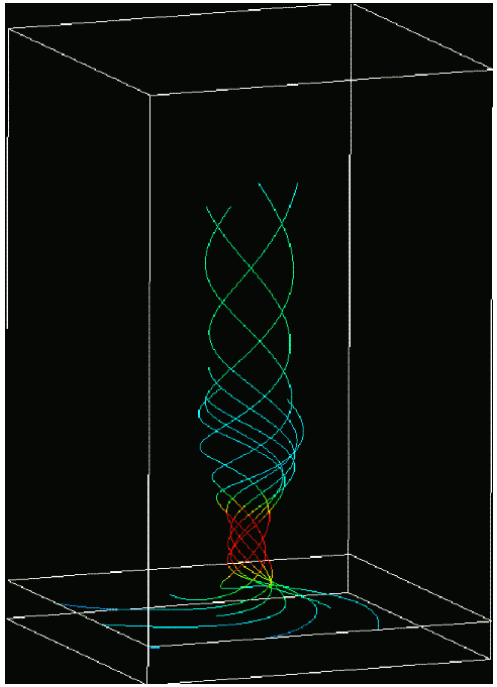
- The data is vector.
- If data is converted into scalar values representing magnitude then isosurfacing and volume rendering can be used.

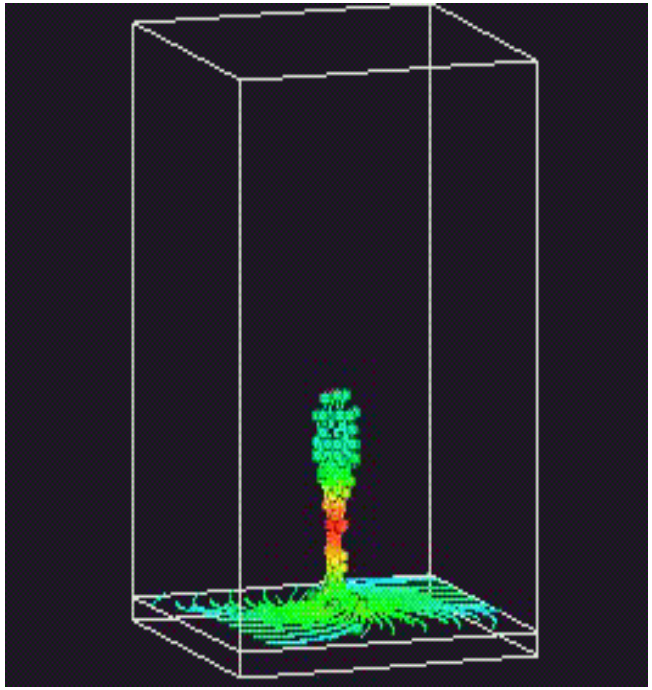




- Vectors can be visualization – glyphs, stream lines and advection



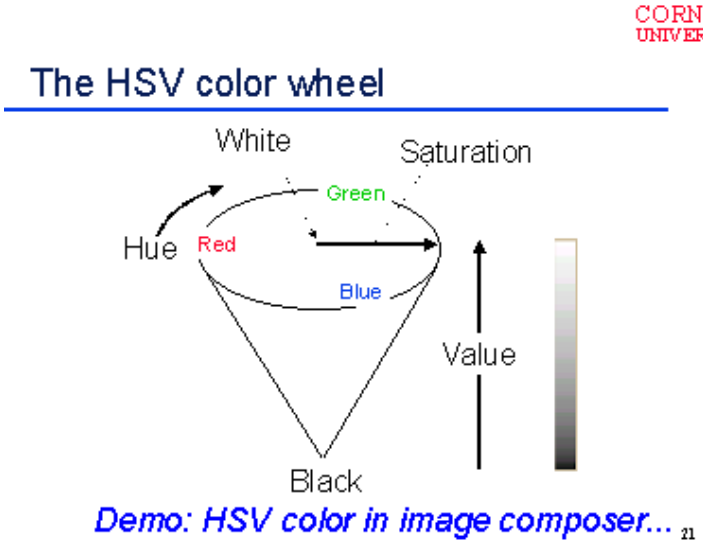


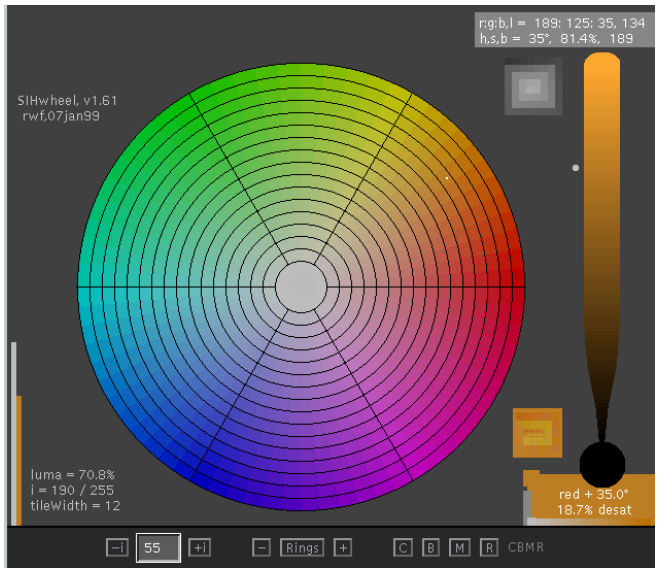


- The ‘orchard’ effect is where the regular placement of glyphs fools the eye into seeing trends along the lines where arrows are placed – interpolation can be used to reduce this effect.

Colour

- The colour wheel and the HSV (hue, saturation, value) model.

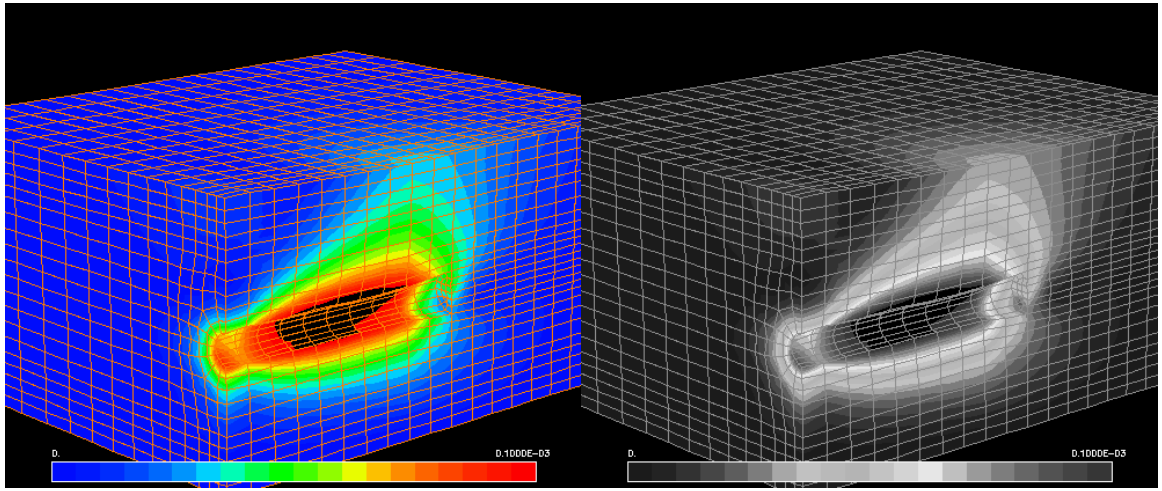




- http://home.att.net/~rocq/SIHwheel.html
- RGB – screens
- CMY(K) - printers

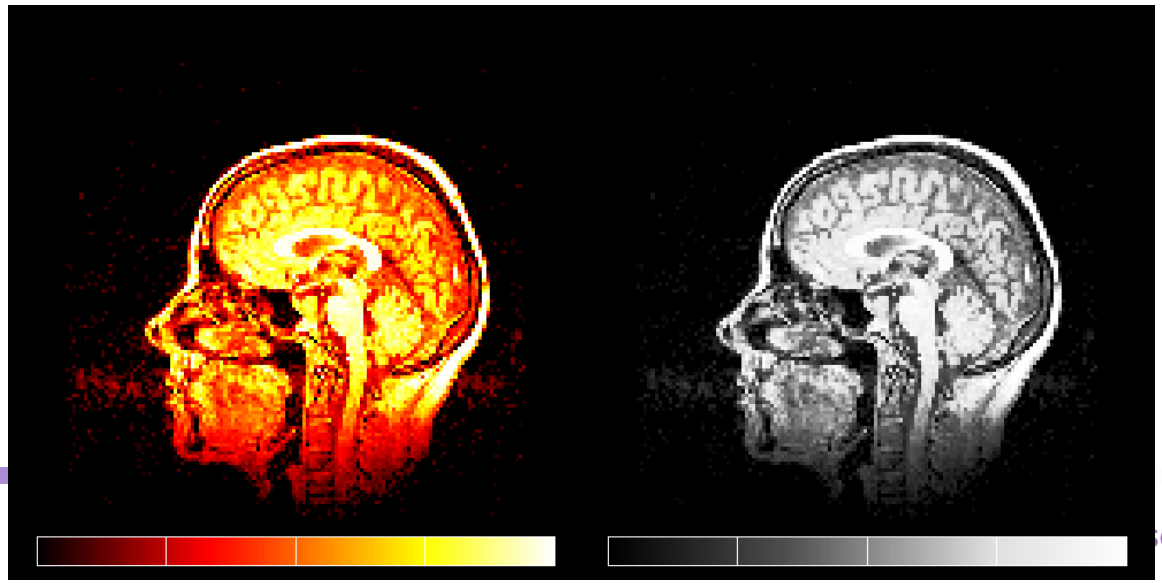
Colour Mapping (Using pseudo-colour)

- Light spectrum (rainbow): *blue is “cool” and red is “hot”.*
- *Often the default colour map but not necessarily the best:*
  - When printed in b+w it is not monotonic
  - Consider ‘Hot-iron’ instead



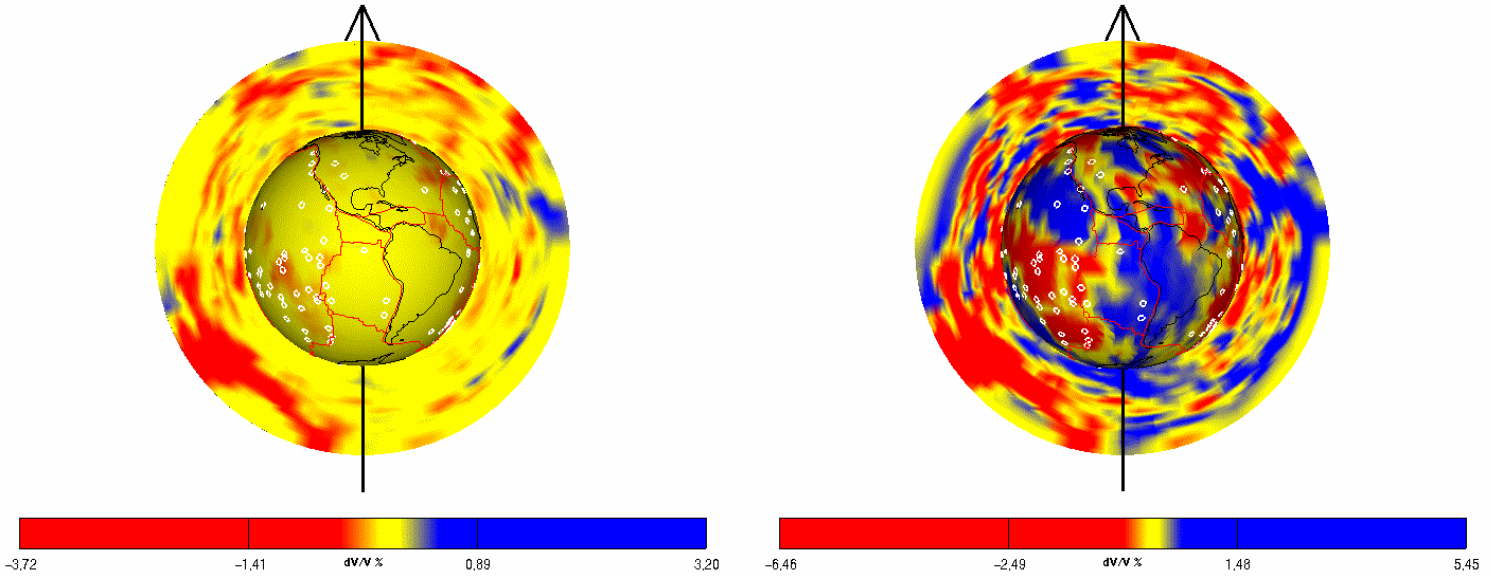
Colour Mapping (Using pseudo-colour)

- Modified light spectrum (hot metal): *Looks good in colour or black and white.*
- A colour scale: black->red->yellow->white
- in each third ramp on one of the primaries:
  - first red, then green and finally blue
  - no ambiguity about high and low
  - is a greyscale on mono printers.



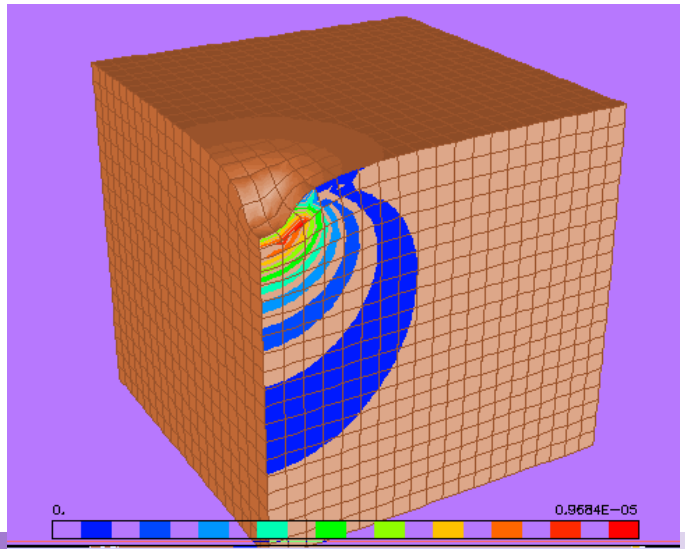
Colour Mapping (Using pseudo-colour)

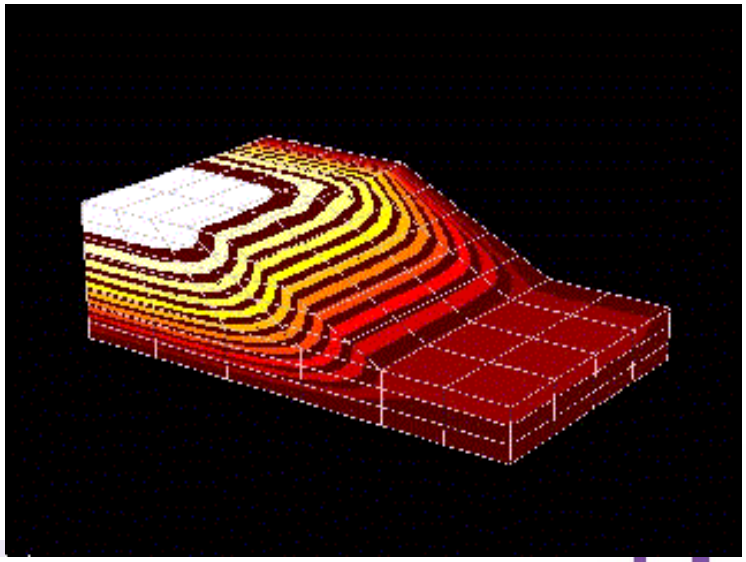
- Contrasting colours at extremes with small colour band in the middle: *Ignores mid range values and is good for showing the deviation from the mean.*



Colour Mapping (Using pseudo-colour)

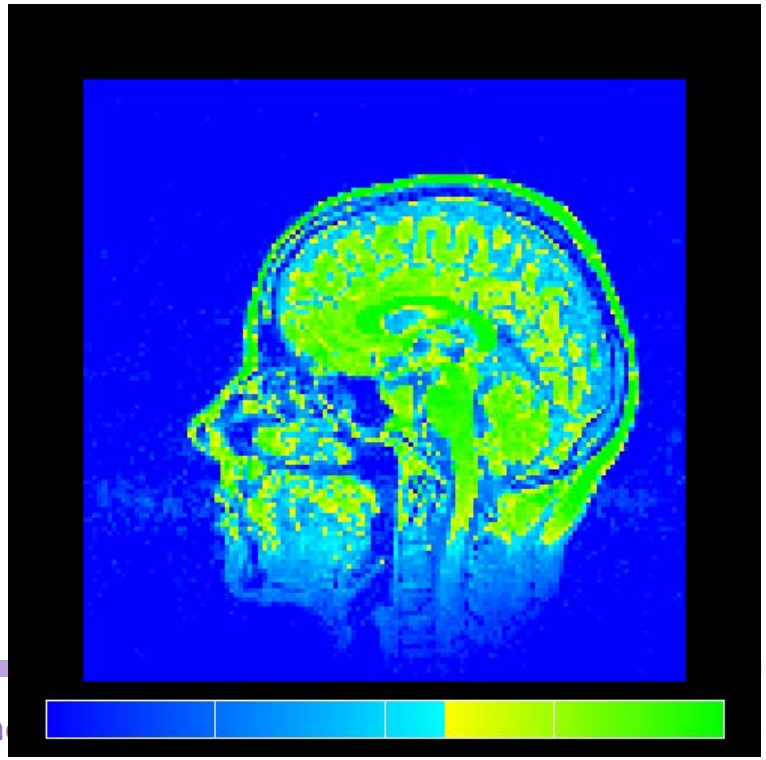
- Colour bands: *Contouring that helps quantify range values.*
- Interleave a continuous scale with a another colour
- three choices:
  - a single colour: e.g. black
  - the same scale but shifted left or right
  - use a texture map





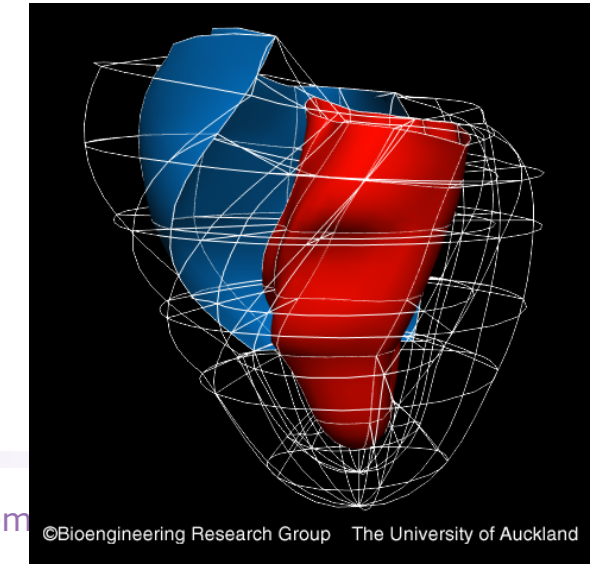
Colour Mapping (Using pseudo-colour)

- Discontinuous colour: *shows values to either side of a threshold.*
- Similar to threshold but shows structures both above and below the threshold value.
- Good if there is a significant value for example zero.



Colour schemes

- Minimise the number of selected colours
- Use contrast to emphasize the main objects
- Use neutral colours for the background. They should provide reasonable contrast to the main objects and may alter with the display environment.
- Use like colours to suggest meaning.
- To distinguish classes of objects use complementary colours evenly distributed on the colour wheel.
- Use additional cues to show shape.



- Heart modelling image produced by the Bioengineering Group, The University of Auckland

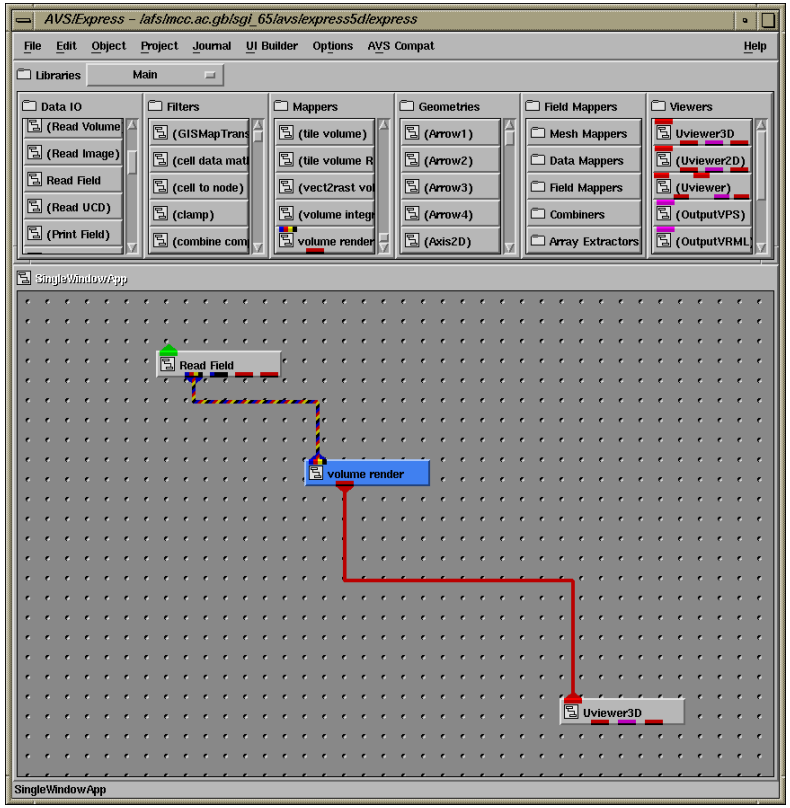
- Use adjacent colours on the colour wheel to show subtle differences.
- Gradually increase colour saturation to show a continuum of change e.g., density.
- Use pastels to show continuity.
- Use a sudden colour change to mark a critical level.
- Be wary of potentially misleading colour illusions or effects.

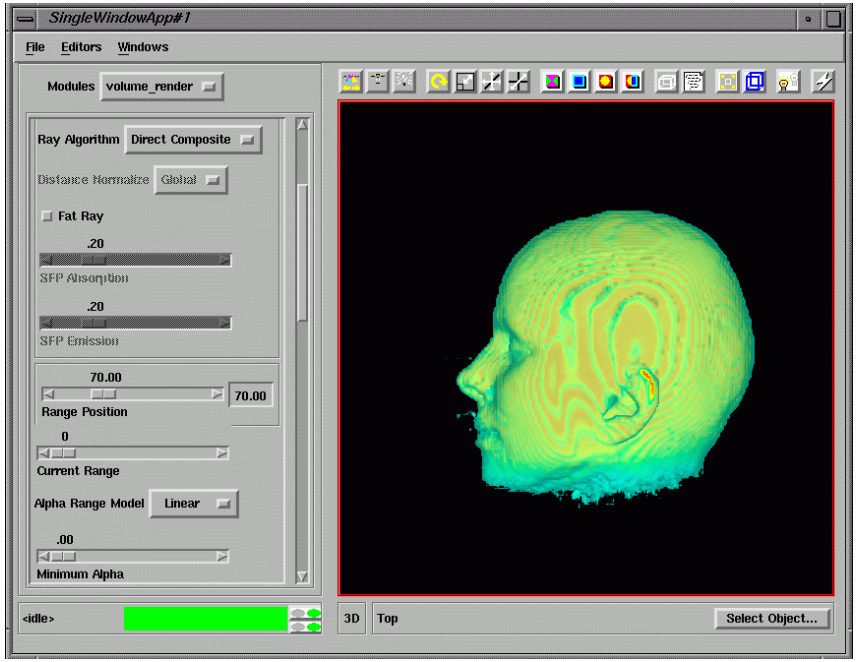
Visualization Systems

- Some computational steering systems like Sci-run have good visualization functionality
- However most visualization system were designed as post-processing systems
- Modular Visualization systems have large GUI which are easy to learn with and play with, e.g. AVS, Iris explorer
- Visualization libraries like vtk can be very comprehensive but are difficult to get started with and play. VTK has good mail groups.
- Graphics libraries like postscript and VRML can be useful.

AVS/Express a Modular Visualization Systems

- A visual programming environment
- An associated view of the data:

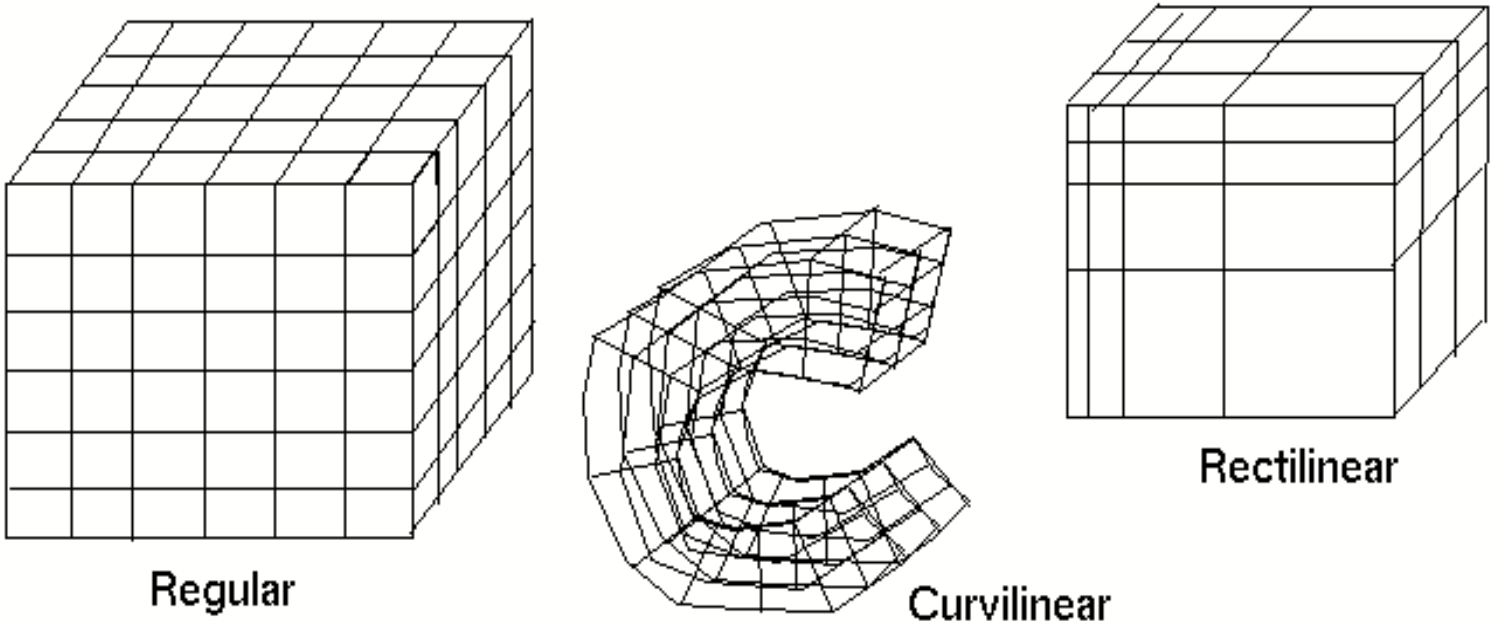




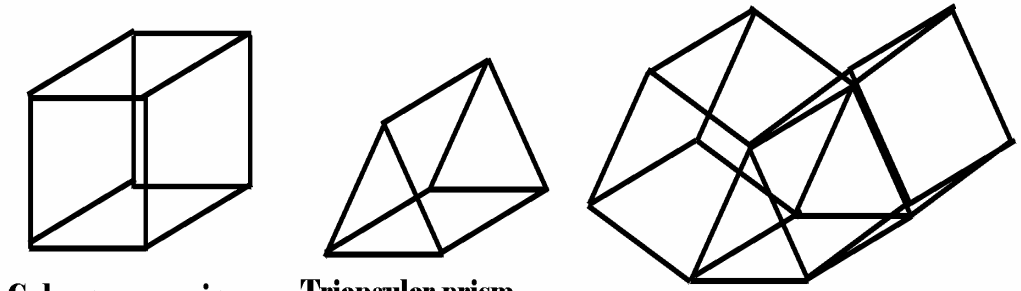
Data in Visualization Systems

- All visualization systems have ways of grouping all the arrays of data and information necessary to process and render 2D and 3D blocks of Data
- AVS/Express calls these fields

"Array Data"



"Cell Data" is FEA data



Problems:

- No standard FEA data format
- In a graphics system each cell has nodes (apexes) and connectivity
  1. uses more memory than "array data"
  2. cells may appear inside out.
- Interpolation to an array based grid may be difficult and may loose information

Thanks To

- All members of Supercomputing, Visualization and eScience

In particular:

John Brooke
Nigel John
Lee Margetts
Dan Kidger
David Moss
Ian Smith
Andy Phillips
George Leaver
Paul Lever
James Perrin
Mary McDerby
Yien Kwok

Fiona Cook
David Bulivant
Terry Hewitt

Manchester Computing

Manchester Computing

Europe's Premier Academic Computing Service

www.mc.man.ac.uk
mc-info@man.ac.uk

THE UNIVERSITY of MANCHESTER