Scalar Algorithms: Colour Mapping

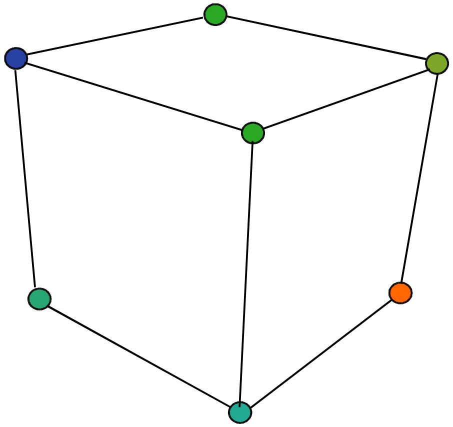
Visualisation – Lecture 5

Taku Komura

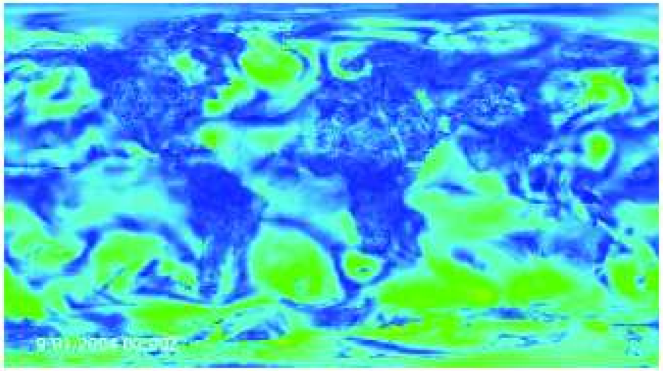
Institute for Perception, Action & Behaviour
School of Informatics

From last lecture ......

- Data representation
  - structure + **value**
  - structure = topology & geometry
  - value = attribute



- Attribute Classification
  - **scalar** (today)
  - vector
  - tensor



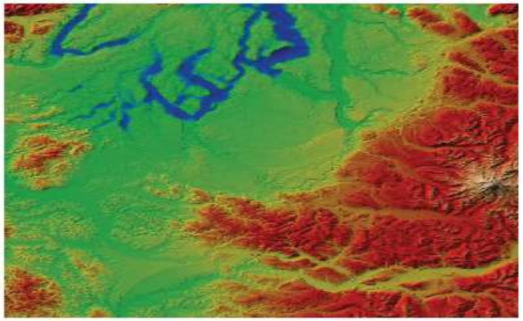
Visualisation Algorithms

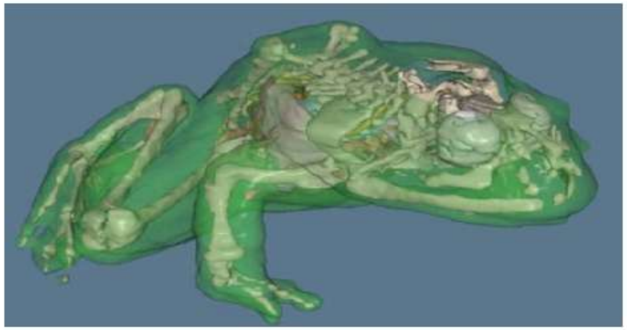
- Generally, classified by attribute type
  - **scalar algorithms** (e.g. colour mapping)
  - vector algorithms (e.g. glyphs)
  - tensor algorithms (e.g. tensor ellipses)

Scalar Algorithms

- **Scalar data : single value** at each location

- Structure of data set may be 1D, 2D or 3D+



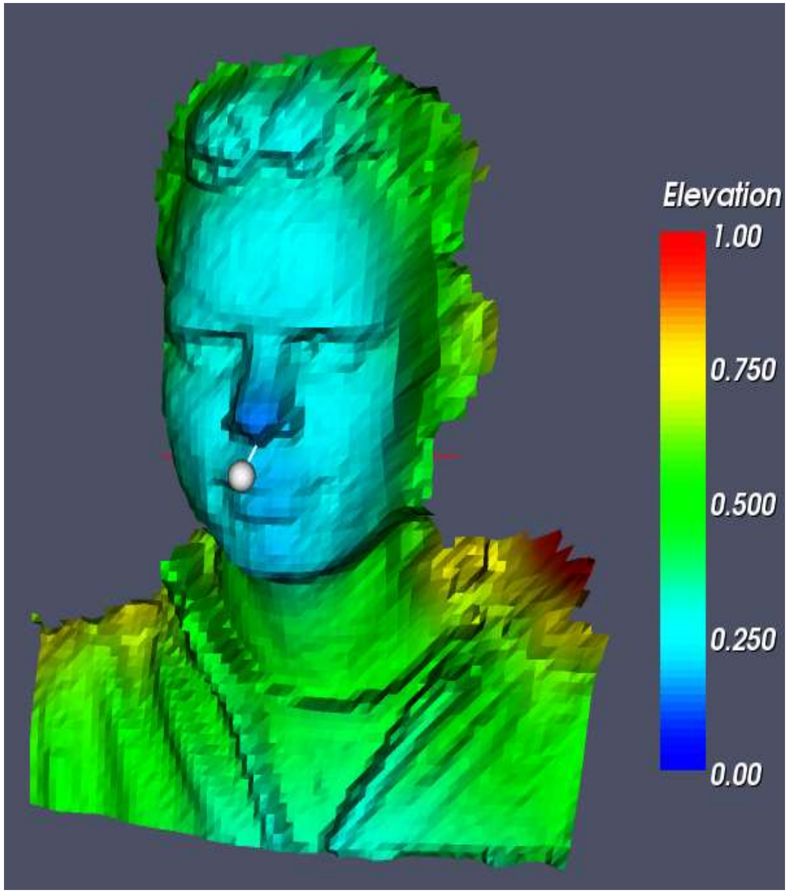


  - we want to visualise the **scaler within this structure**

- Two fundamental algorithms
  - colour mapping (**transformation :** value → colour)
  - contouring (**transformation :** value transition → contour)

Colour Mapping

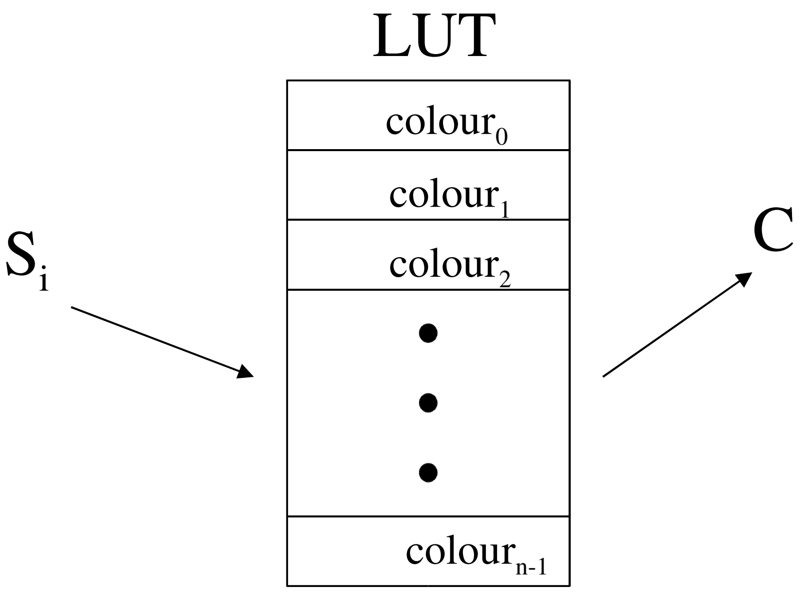
- Map **scalar value to colour range** for display
  - e.g.
    - scalar value = height / max elevation
    - colour range = blue →red



- **Colour Look-up Tables** (LUT)
  - provide scalar to colour conversion
  - scalar values = indices into LUT

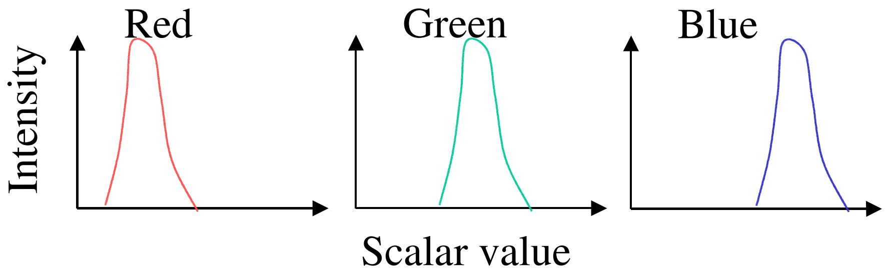
Colour LUT

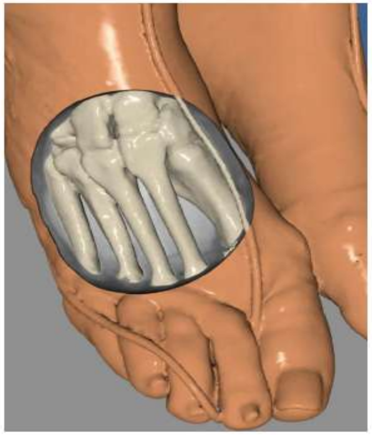
- **Assume**
  - scalar values S_i in range {min→max}
  - n unique colours, {colour_0... colour_{n-1}} in LUT
- **Define mapped colour C:**
  - if S_i < min then C = colour_{min}
  - if S_i > max then C = colour_{max}
  - else



Colour Transfer Function

- More general form of colour LUT
  - scalar value S; colour value C
  - **colour transfer function** : f(S) = C
  - Any functional expression can map scalar value into intensity values for colour components

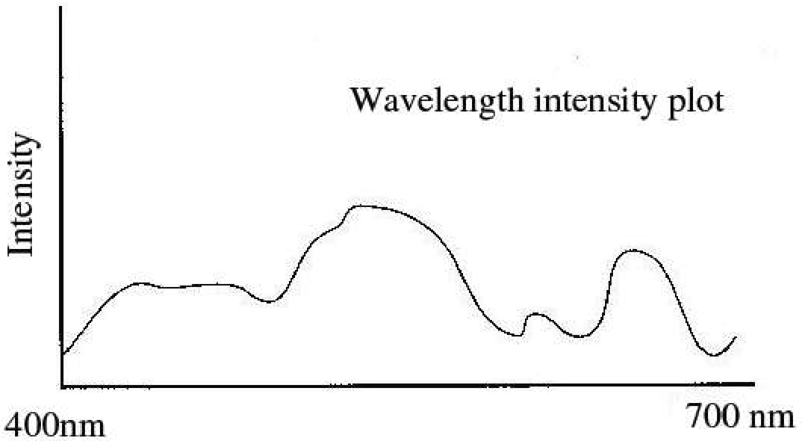


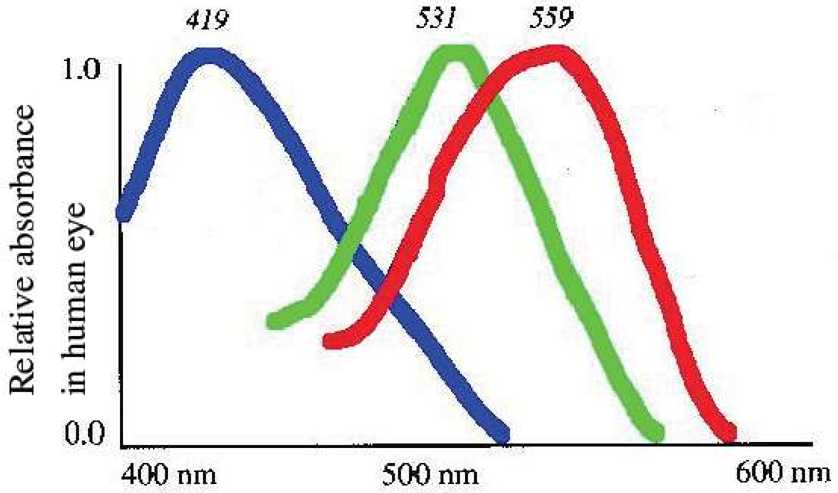


  - e.g. define f() to convert densities to realistic skin/bone/tissue colours

Colour Components

- **EM spectrum** visible to humans
  - continuous range 400-700nm
  - 3 type of receptors (cones) in eye for R, G, B.

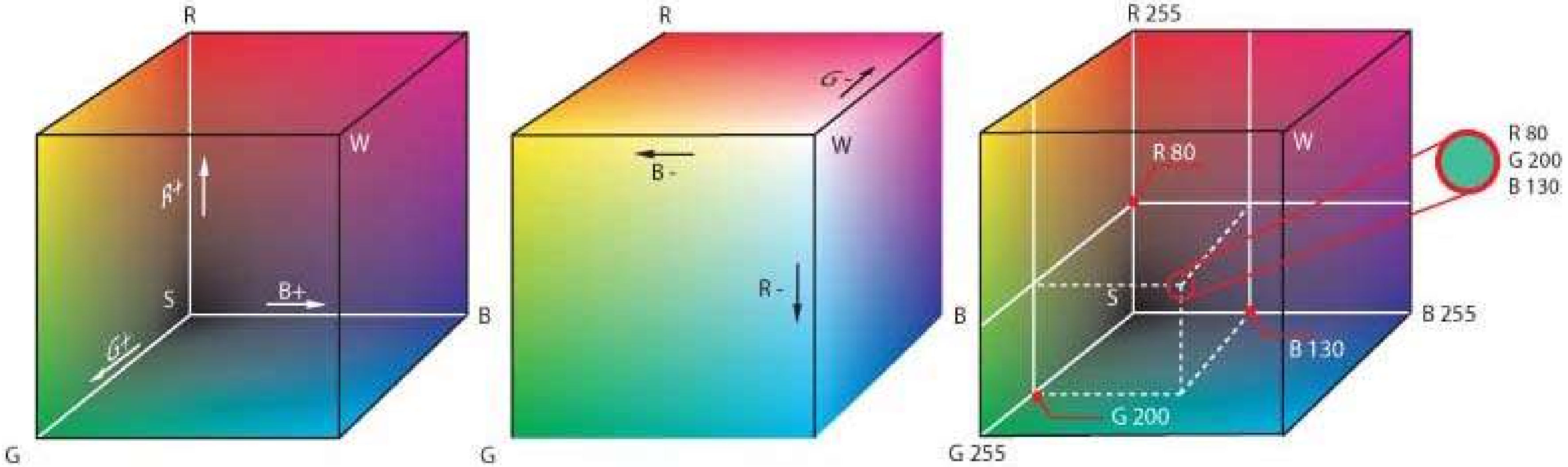




- So we can use the **RGB model in CG** for visualization

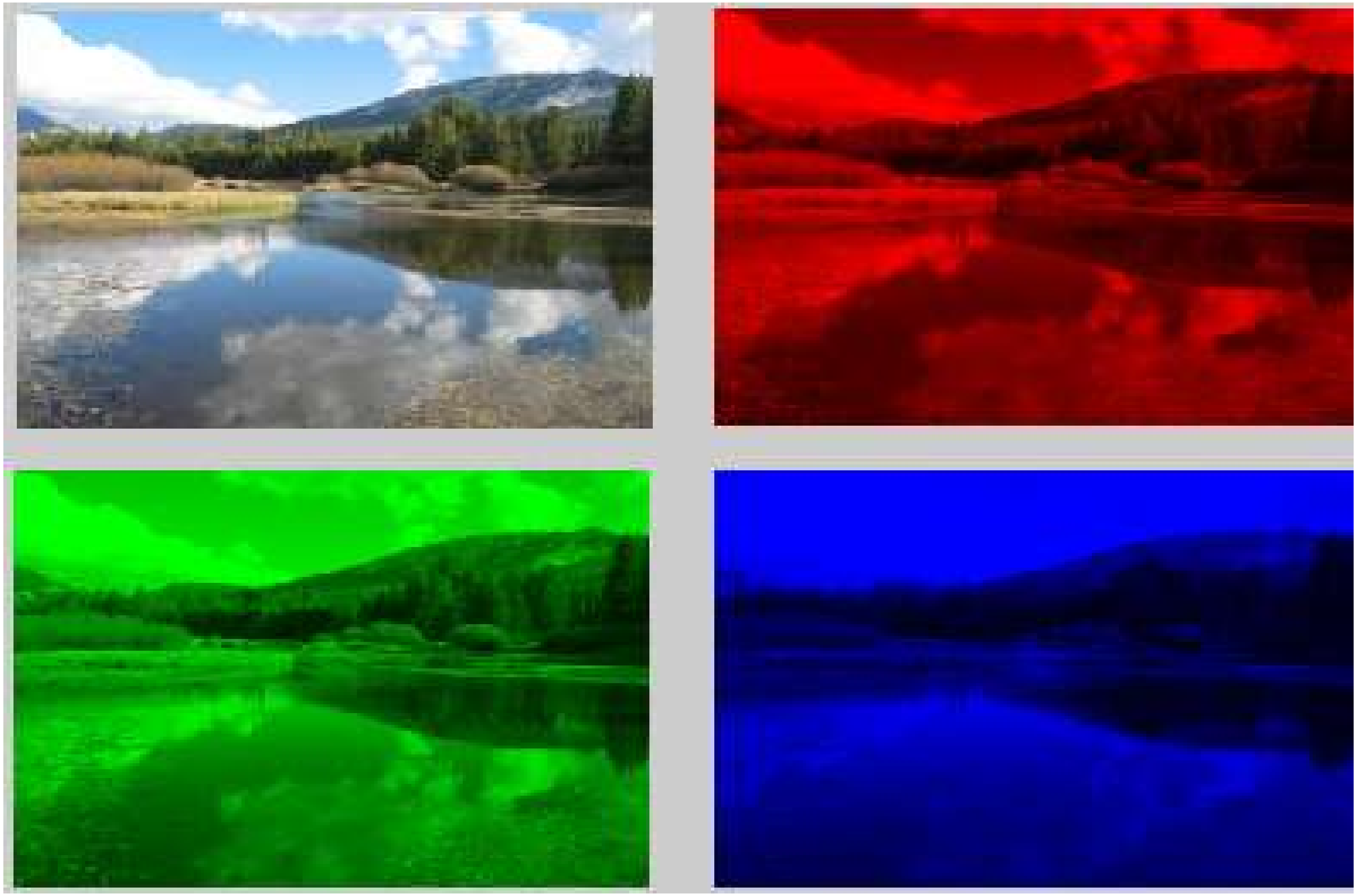
Colour Spaces - RGB

- Colours represented as R,G,B intensities
  - **3D colour space** (cube) with axes R, G and B
  - each axis 0 → 1 (below scaled to 0-255 for 1 byte per colour channel)
  - Black = (0,0,0) (origin); White = (1,1,1) (opposite corner)



  - **Problem :** difficult to map continuous scalar range to 3D space
    - can use subset (e.g. a diagonal axis) but imperfect

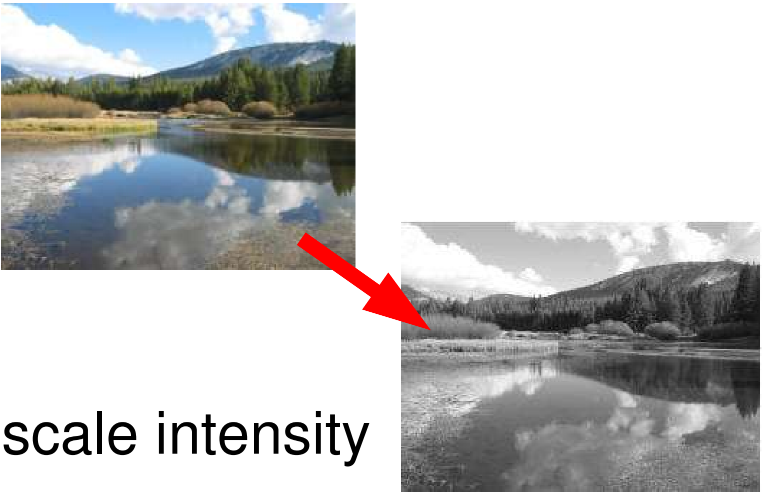
Example : RGB image



RGB Channel Separation

Colour Spaces - Greyscale

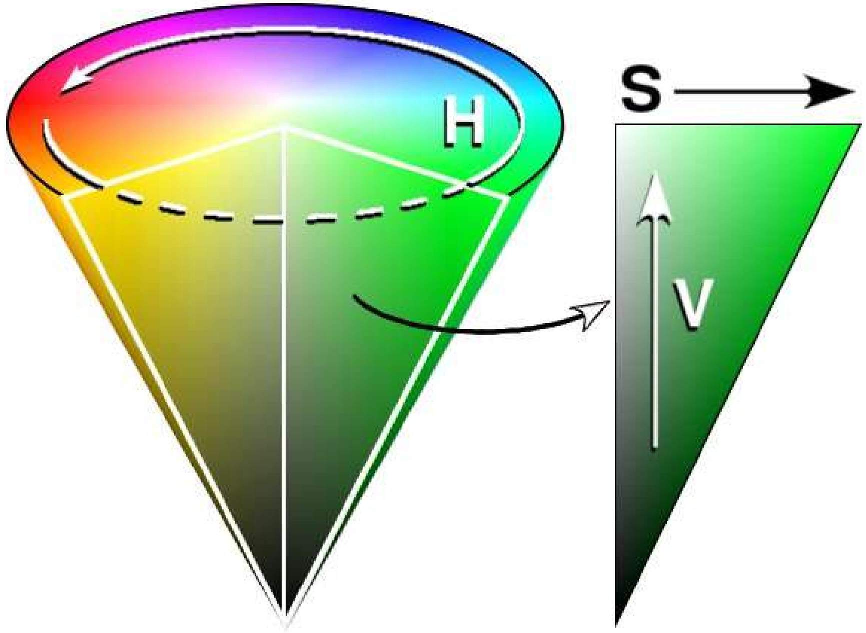
- **Linear combination of R, G, B**
  - `Greyscale = (R + G + B) / 3`



- Defined as linear range
  - easy to map linear scalar range to grayscale intensity
  - can **enhance structural detail in visualisation**
    - The shading effect is emphasized
    - as distraction of colour is removed
  - **not really using full graphics capability**
  - **lose colour associations** : e.g. red=bad/hot, green=safe, blue=cold

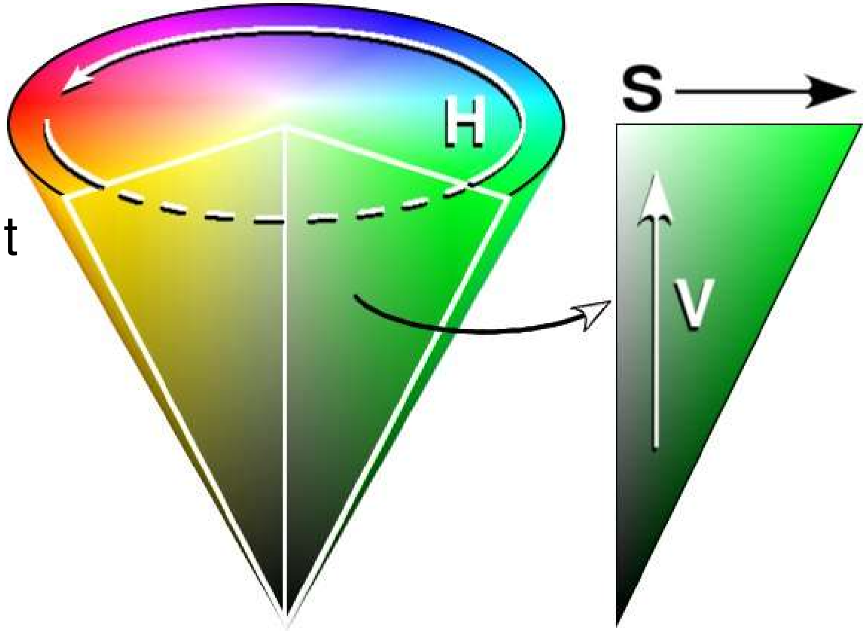
HSV

- HSV encapsulates information about a color in terms that are more familiar to humans:
  - *What color is it?*
  - *How vibrant is it?*
  - *How light or dark is*



Colour spaces - HSV

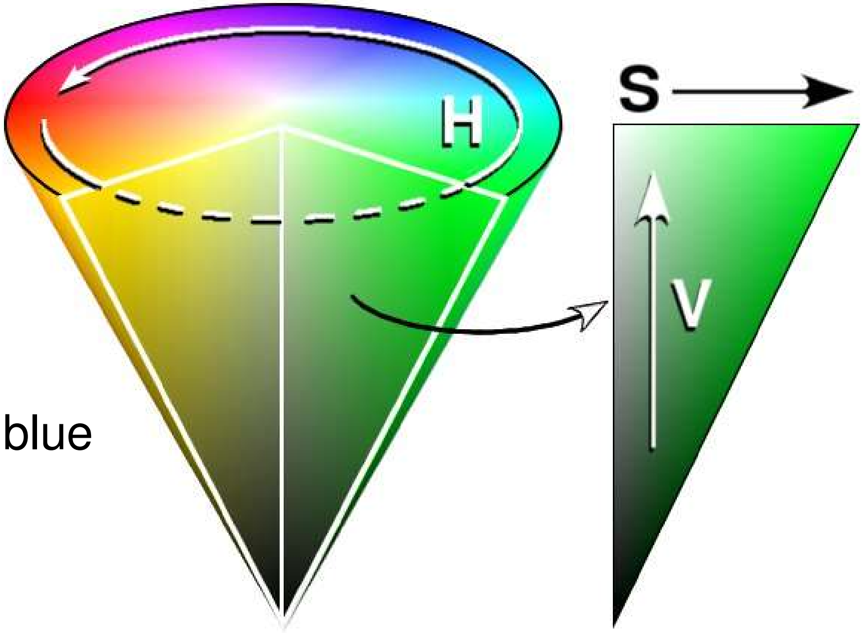
- **Colour represented in H,S,V parametrised space**
  - commonly modelled as a cone
- **H (Hue)** = dominant wavelength of colour
  - **colour type** {e.g. red, blue, green...}
- **S (Saturation)** = amount of Hue present
  - “vibrancy” or **purity of colour**
- **V (Value)** = brightness of colour
  - **brightness** of the colour



Colour spaces - HSV

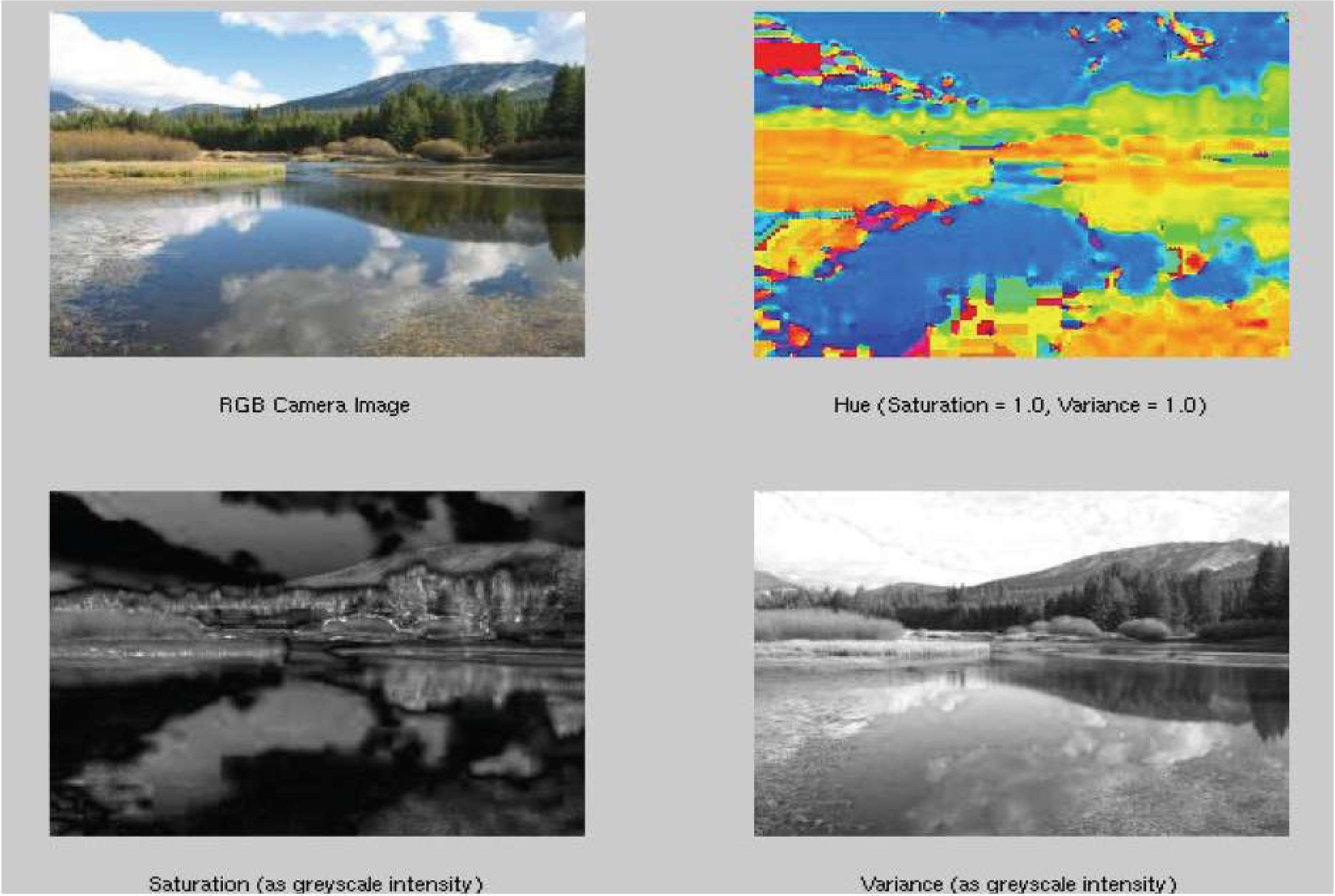
- **HSV Component Ranges**
  - **Hue** = 0 → 360°
  - **Saturation** = 0 → 1
    - e.g. for Hue≈blue

      0.5 = sky colour; 1.0 = primary blue
  - **Value** = 0 → 1 (amount of light)
    - e.g. 0 = black, 1 = bright



- **All can be scaled to 0→100% (i.e. min→max)**
  - use hue range for colour gradients
  - **very useful for scalar visualisation with colour maps**

Example : HSV image components



RGB Camera Image

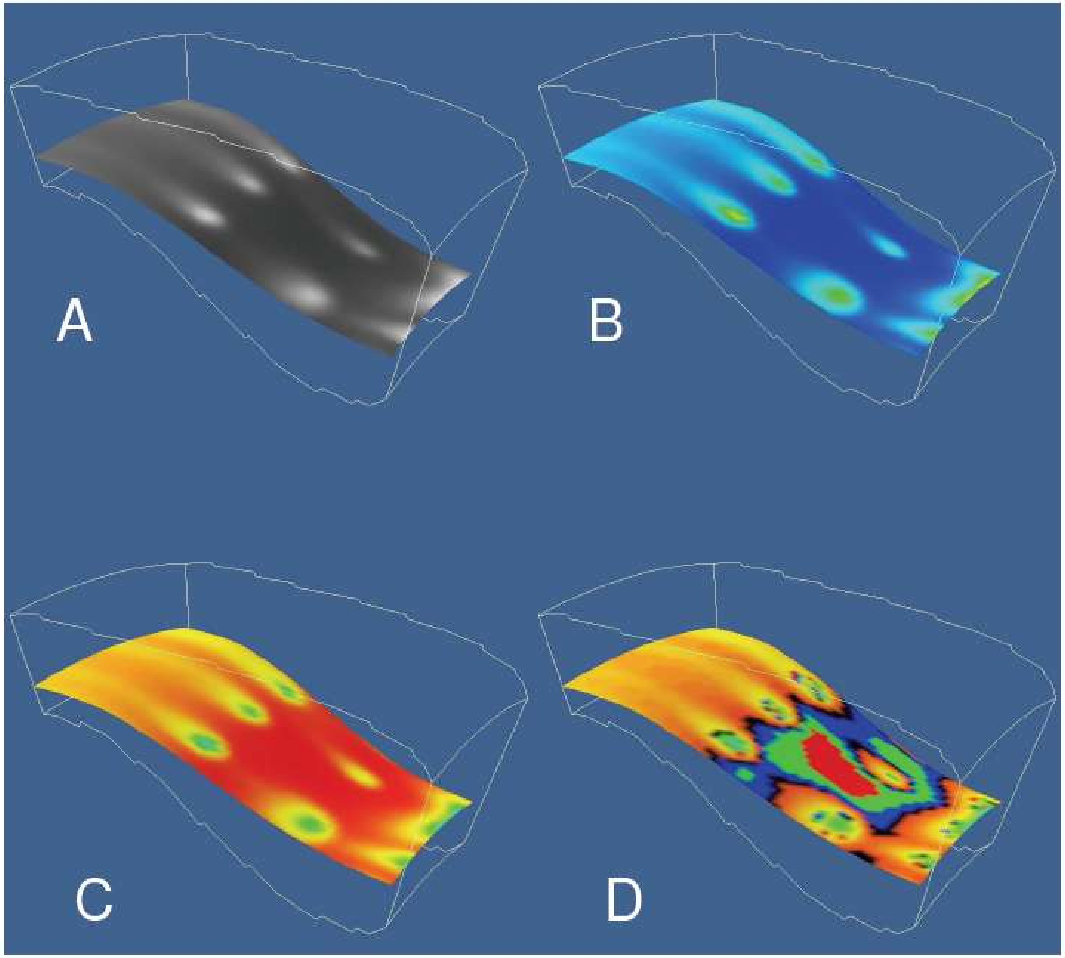
Hue (Saturation = 1.0, Variance = 1.0)

Saturation (as greyscale intensity)

Variance (as greyscale intensity)

Different Colour LUT

- Visualising gas density in a combustion chamber
  - **Scalar** = gas density
  - **Colour Map** =



A: grayscale

B: hue range blue to red

C: hue range red to blue

D: specifically designed transfer function

  - *highlights contrast*
  -

Colour Table Design

- "More of an art that a science"
  - **debate** – *where does visualisation end and art begin?*
- Key focus of colour table design
  - **emphasis important features / distinctions**
  - **minimise extraneous detail**
- Often task specific
  - consider application (e.g. temperature change, use hue red to blue)
  - consider viewer (**colour associations**, colour blindness)
  - ***Rainbow colour maps –*** rapid change in colour hue

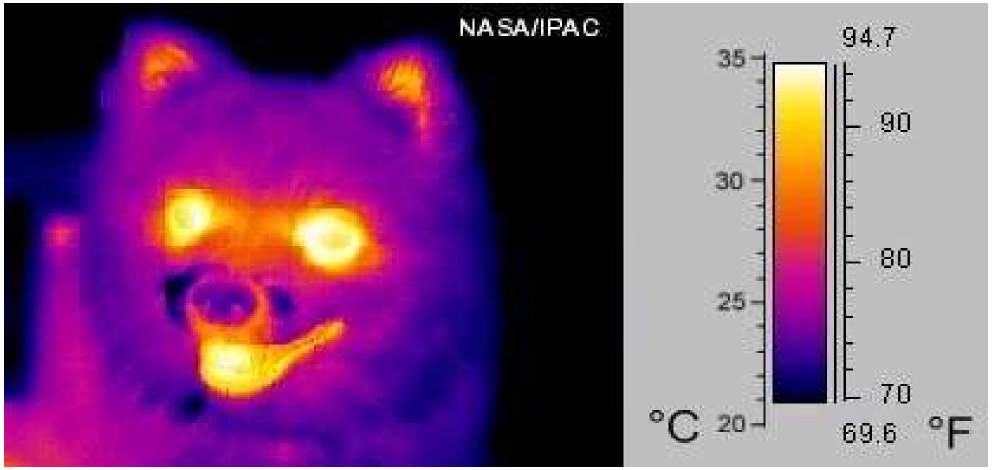
    representing a 'rainbow' of colours.

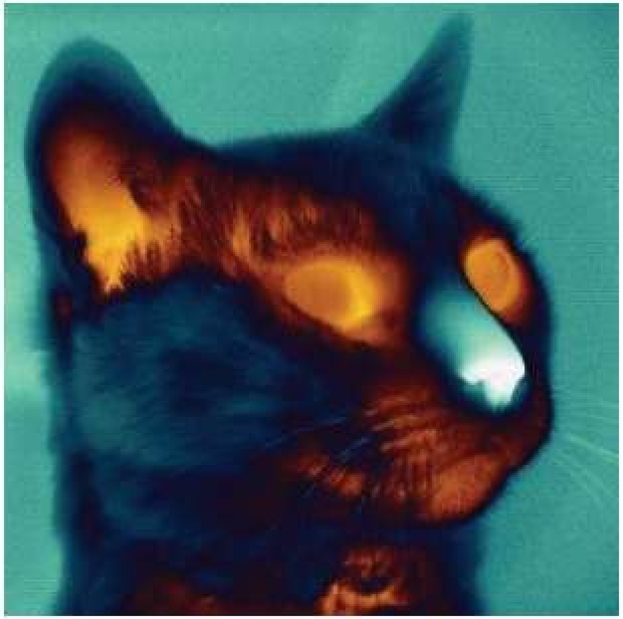
    shows small gradients well as colours change quickly.

Examples – 2D colour images

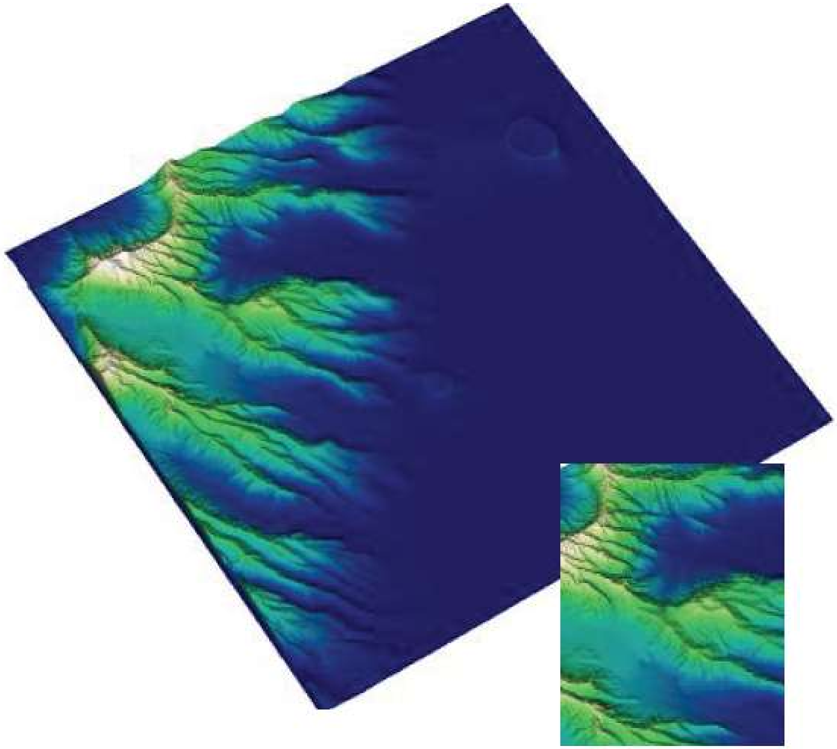
- Infra-red intensity viewed as Hue
  - received from sensor as 2D array of infra-red readings
  - visualise as colour image using colour mapping

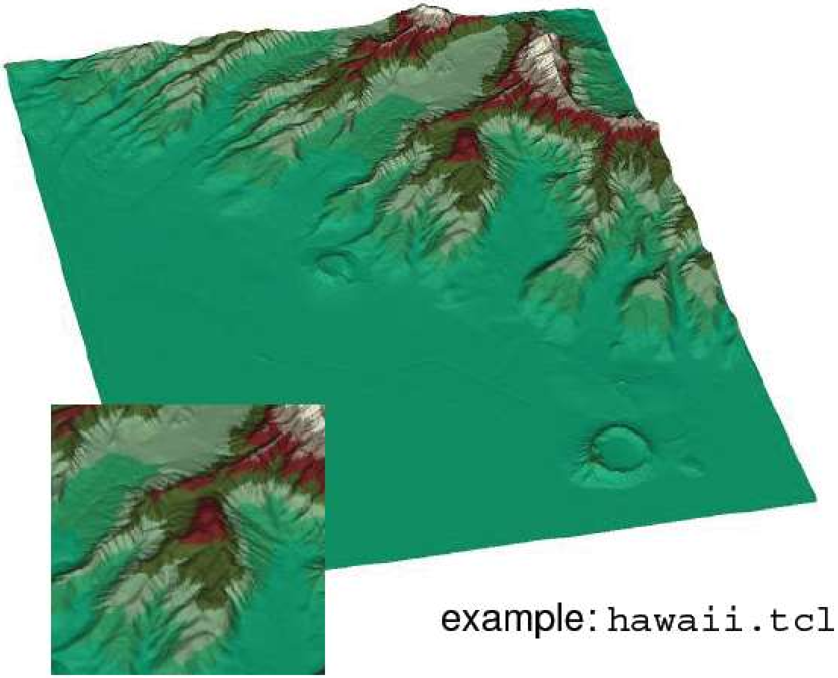
–





Examples – 3D Height Data





example: `hawaii.tcl`

HSV based colour transfer function

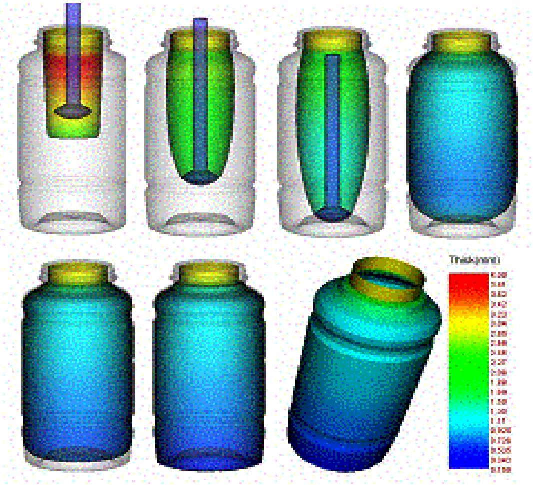
- continuous transition of height represented

8 colour limited lookup table

- discrete height transitions
- **rainbow type effect**

Colour Mapping

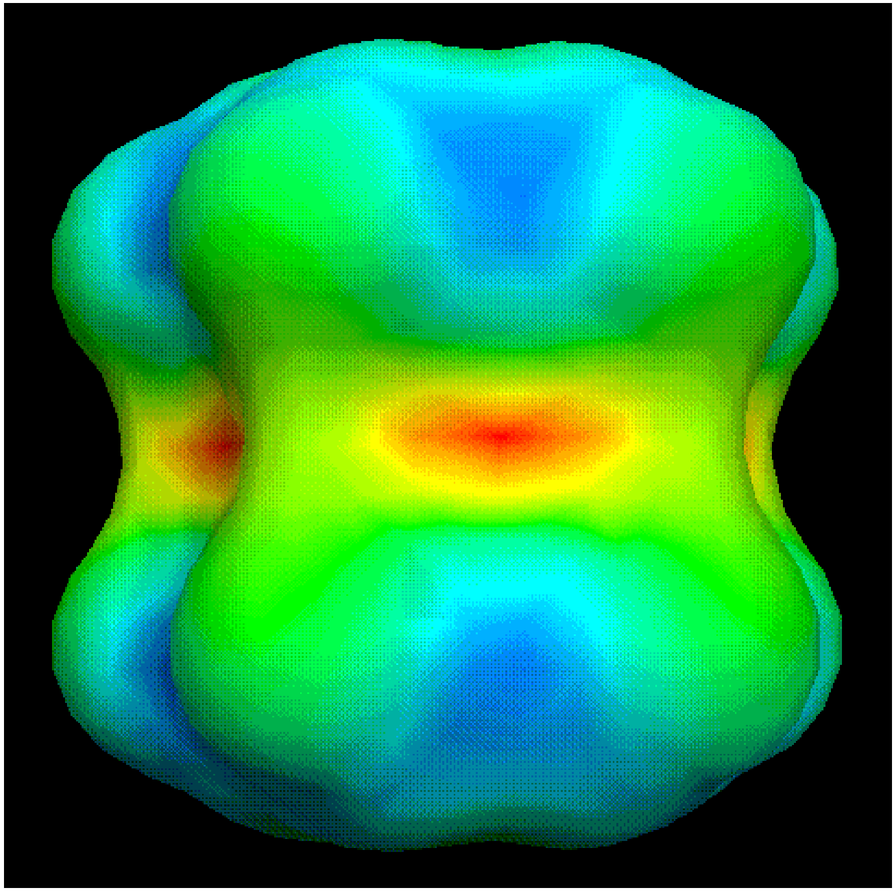
- Linear or **1D mapping process**
- Use to **map colour onto surfaces, images, volumes (>1D)**



Visualisation of a blow-moulding process.
Colour indicates wall thickness.

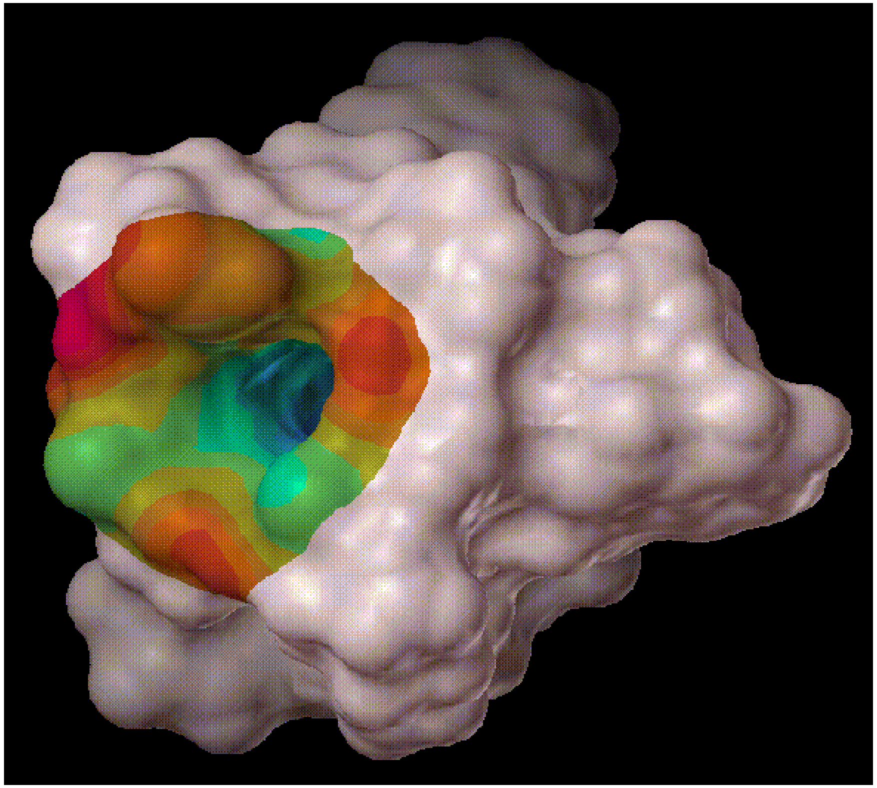
- Theoretically 3 channels of information are available:
  - H, S and V
  - But V (brightness) frequently used for shading, important for visualising 3D shape. Normally H and S only used.

Molecular visualisations



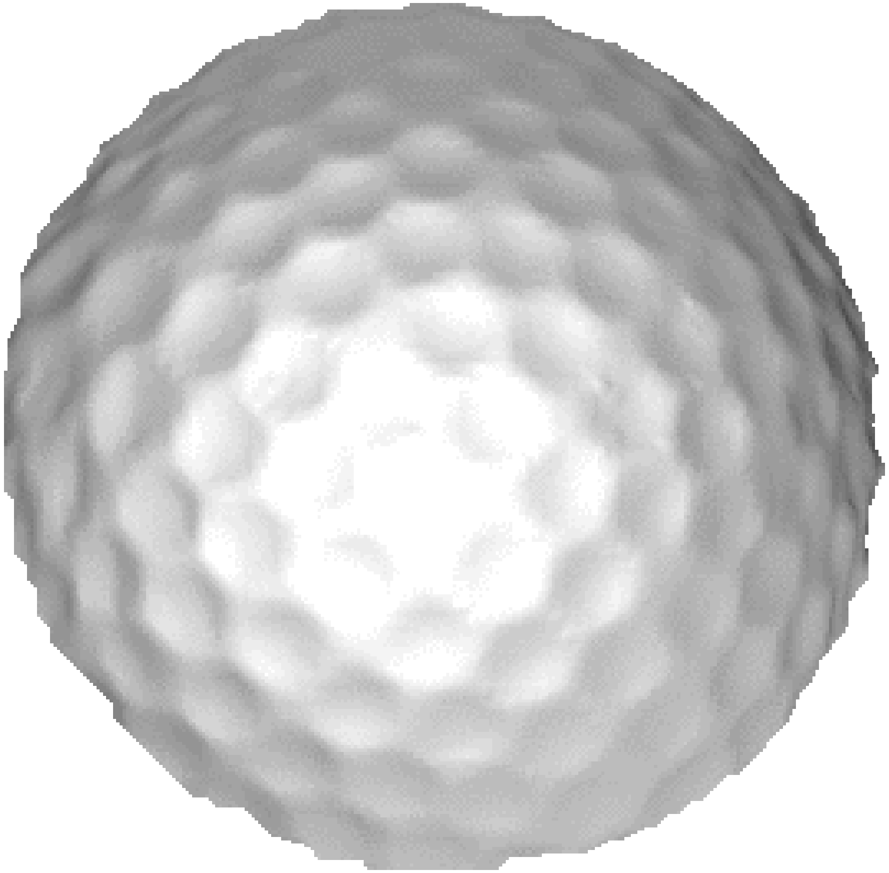
Two variables visualised relating to electric properties

- mapped to **Hue and Saturation**



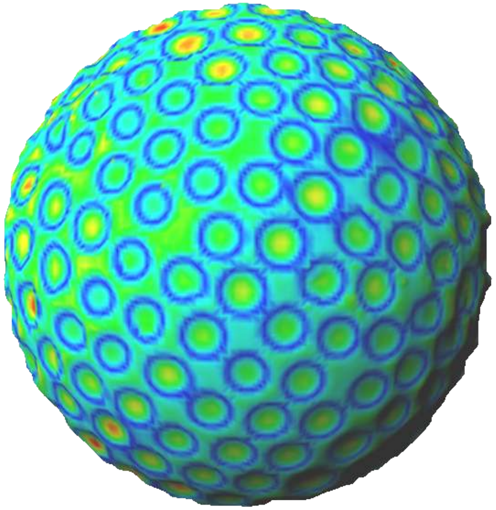
Example : Colour Transfer Function

- **Question :** Are the dimples on this golfball evenly distributed?



Example : Colour Transfer Function

- **Answer :** No. *Why ?* Improves flight characteristics.



- **Visualisation technique** : colour map each point based on distance (scalar) from regular sphere

VTK : Colour Mapping

- To create a new LUT object with a name lut:

```
vtkLookupTable lut
```

- To set the colour range in the HSV colourspace:

```
lut SetHueRange start finish
lut SetSaturationRange start finish
lut SetValueRange start finish
```

  - range = [0,1]

- Also define specific N colour lookup table

see `hawaii.tcl` example

VTK Example : Blood Flow 1

- **Application** : blood flow in the carotid arteries
  - blockages are a common cause of strokes
  - **Data source :** Can measure flow velocities using MR Imaging machine and calculating doppler shift
  - Typical data format for scientific/medicine:
    - 3D regular grid of velocity vectors produced
      - velocity = vector field; speed = scalar field
    - structured points data structure
    - size is 76 x 49 x 45; 168,000 points

VTK Example : Blood Flow 2

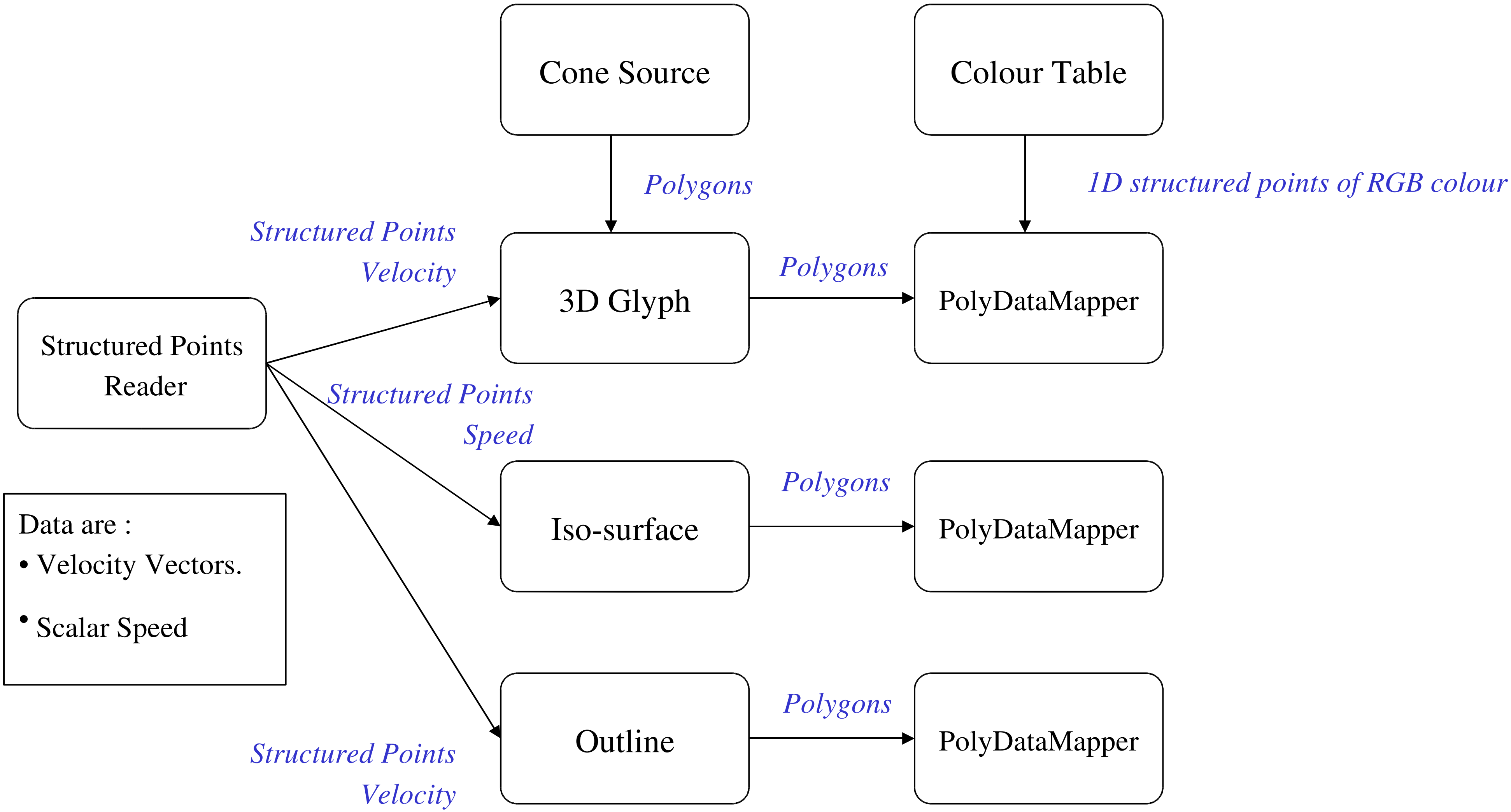
- **Visualisation criteria :**
  - display **flow direction and magnitude** clearly
  - highlight large, **abnormal velocities**
  - show **wall of arteries** for navigation purposes

- **Visualisation solution :**
  - draw little cones (**glyphs**) aligned with the velocities
  - **colour** the cones according to flow magnitude (scalar)
  - show the artery walls as a **polygon surface**
  - draw a **bounding box** around the data to assist 3D navigation

Our VTK Tasks

- **Read the data** in from the file.
  - 2 fields, velocity and speed
- **Create a cone** object (glyph)
- **Place cone** at each of the data points
- **Create colour map** related to speed (scalar)
- **Colour each cone** with the colour map
- **Create surface** at *v*=0 – draw in wireframe
- **Create box** around the data

VTK Example : Blood Flow 3



Cone Source

Colour Table

Polygons

1D structured points of RGB colour

Structured Points
Velocity

3D Glyph

Polygons

PolyDataMapper

Structured Points Reader

Structured Points
Speed

Iso-surface

Polygons

PolyDataMapper

Data are :

- Velocity Vectors.
- Scalar Speed

Outline

Polygons

PolyDataMapper

Structured Points
Velocity

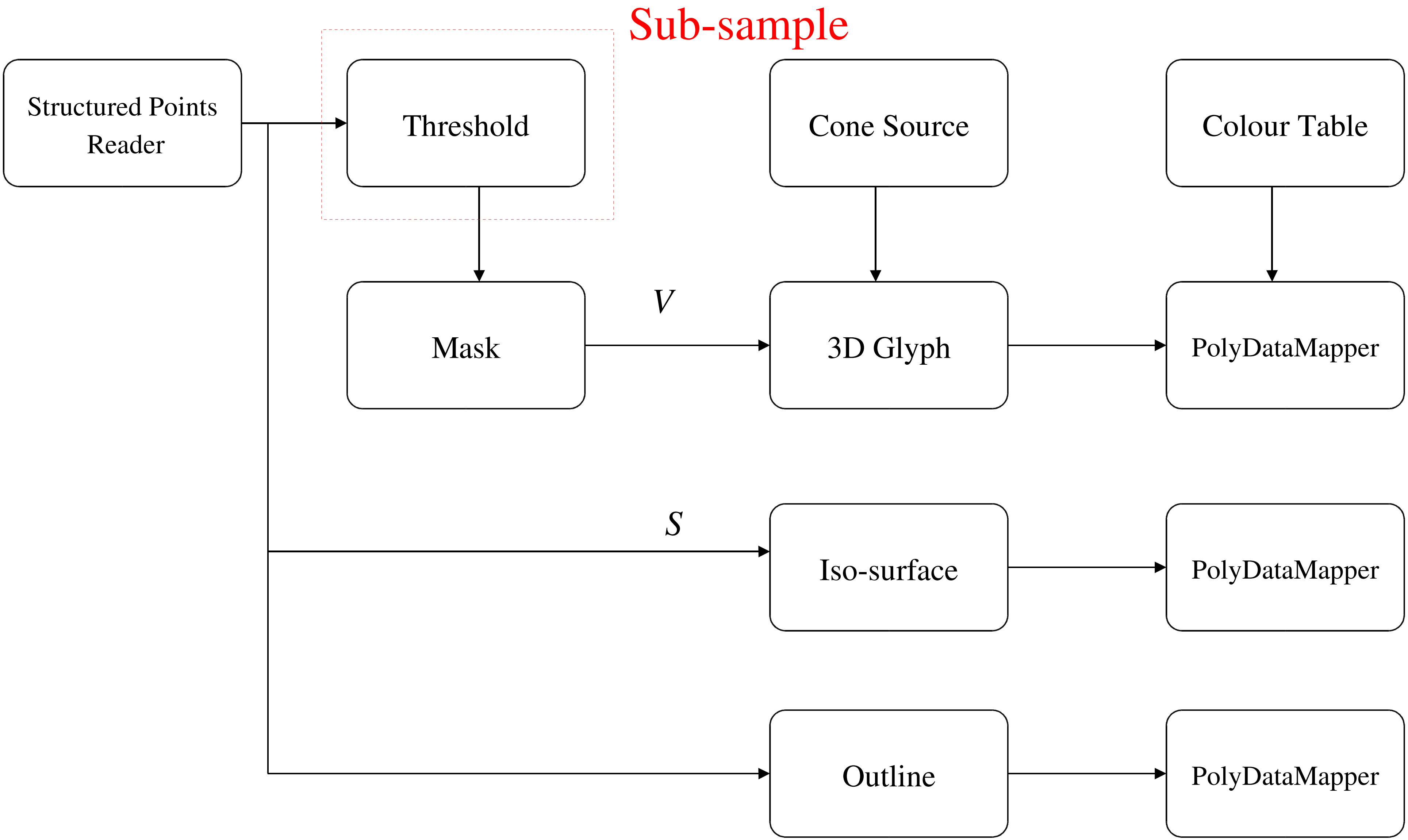
VTK Example : problem

too many cones....

“Can't see the wood for the trees”

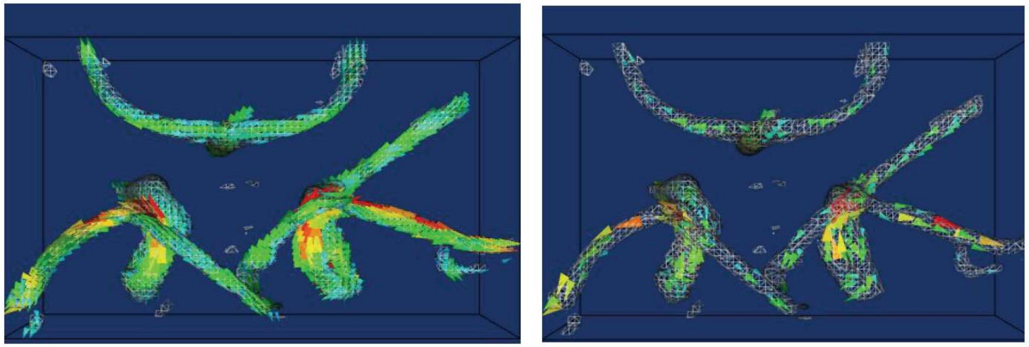
Solution : sub-sample

VTK Example : Blood Flow 3



VTK Example : Problems

- Density of flow : introduce sub-sampling to improve visibility of flow
  - previous slide



- Glyphs take up space
- flow direction and magnitude at a fixed point - visible
  - but cannot see where the blood has come from
- Other methods of flow visualisation
  - later in the course

`thrshldV1.tcl/ thrshldV2.tcl`

Summary

- Introduction to **scalar data**
- **Colour maps**
  - colour **LUT**
  - **colour transfer functions**
  - **RGB** and **HSV** colour spaces
  - design issues
- VTK : colour maps & blood flow example