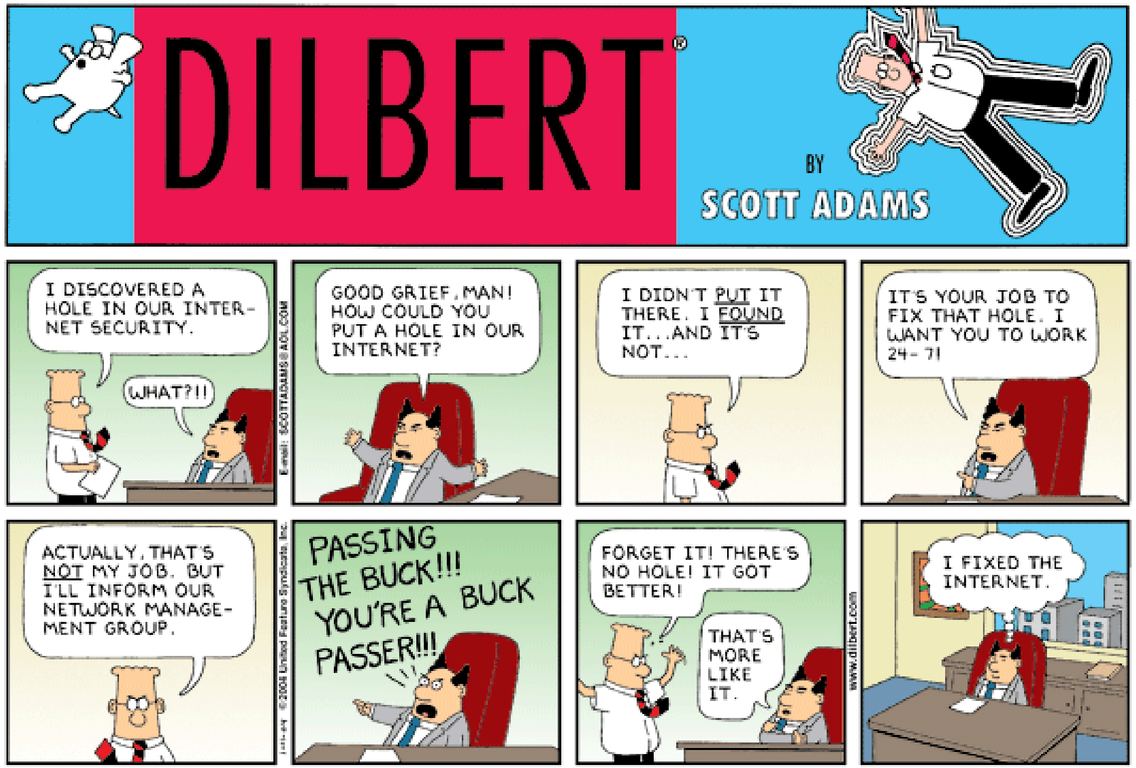
Volume visualization

Iso-Surface Extraction and Marching Cubes



Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

Computer Science
Gwyddor Cyfrifiadur

Marching Cubes

Iso-Surfaces: Generation and Presentation

Surface Rendering

Terminology

Isosurface: "surface made from the level set of a 3D scalar function"

Isovalue: "The scalar value used to generate an isosurface." The VTK

About Surface rendering:

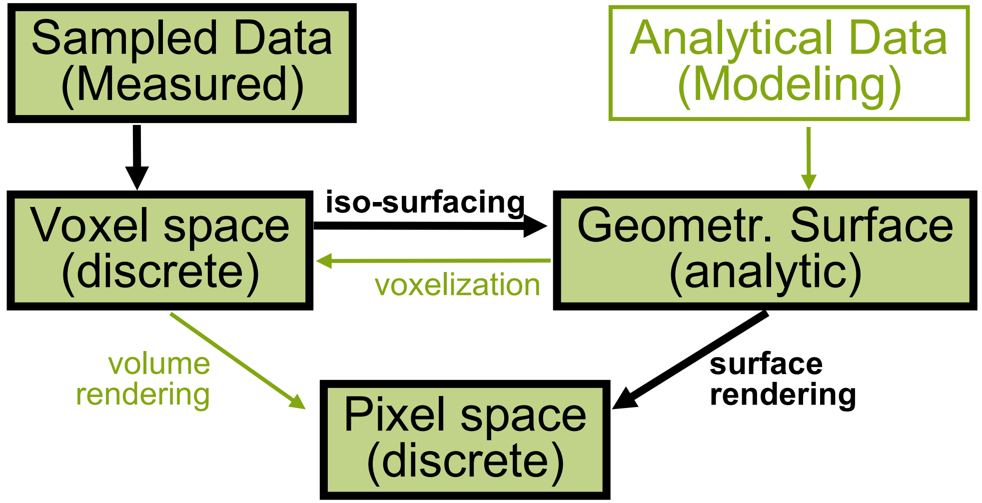
- indirect volume visualization technique
- isosurfaces are computed from 3D volume data

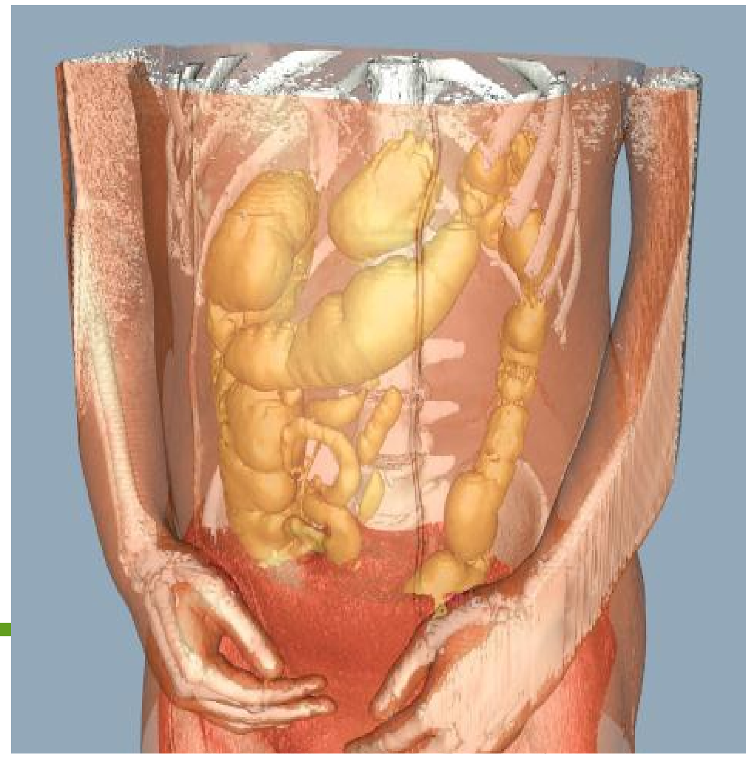
Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

Conceptual VolVis Framework

Example:

- CT measurement
- iso-stack compuation
- iso-surface computation (marching cubes)
- Surface rendering (OpenGL)





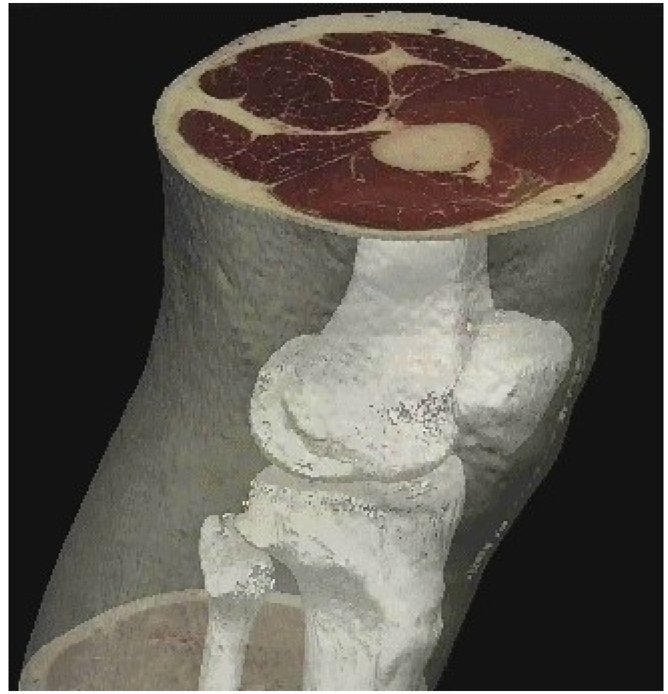
Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

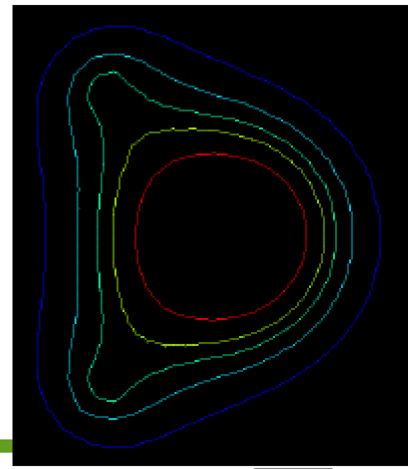
Iso-Surfaces

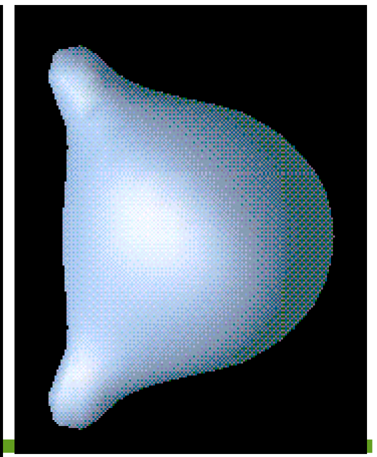
Indirect representation

Characteristics:

- Meaningful isovalue,
- Isovalue separates substances
- Interest in boundary regions
- Very selective (binary choice/filter)
- Uses traditional hardware
- Shading gives a 3D impression





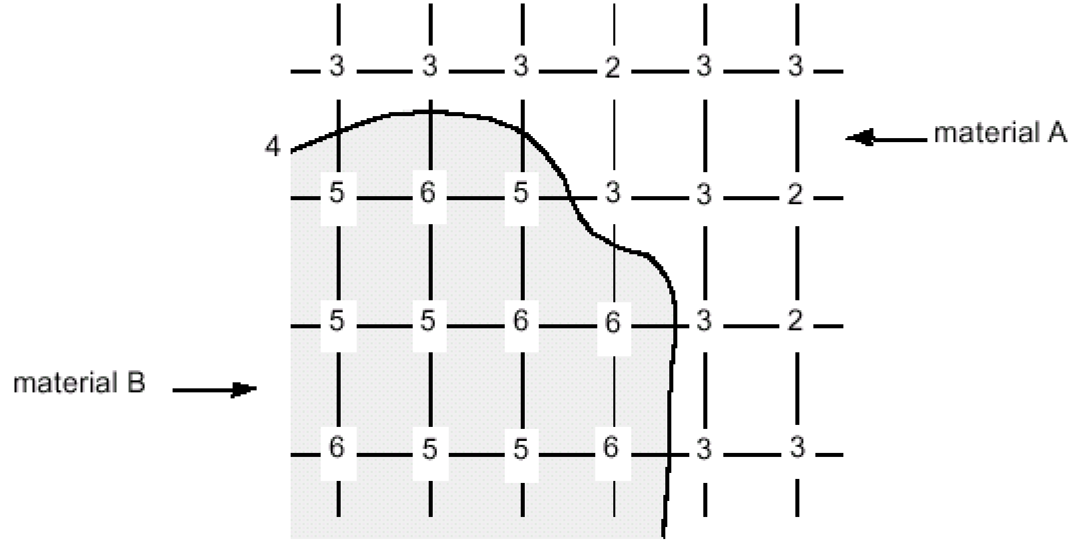


Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

Planar Data and Iso-Contours

Iso-Contours:

- isovalue f_0
- Separate values $> f_0$ from values $\leq f_0$
- Can be approximated from given samples
- Form/Position depends on reconstruction (interpolation)



https://www.youtube.com/watch?v=B_xk71YopsA

Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

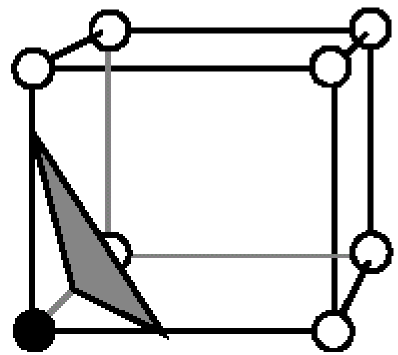
Approximating the Iso-Surface

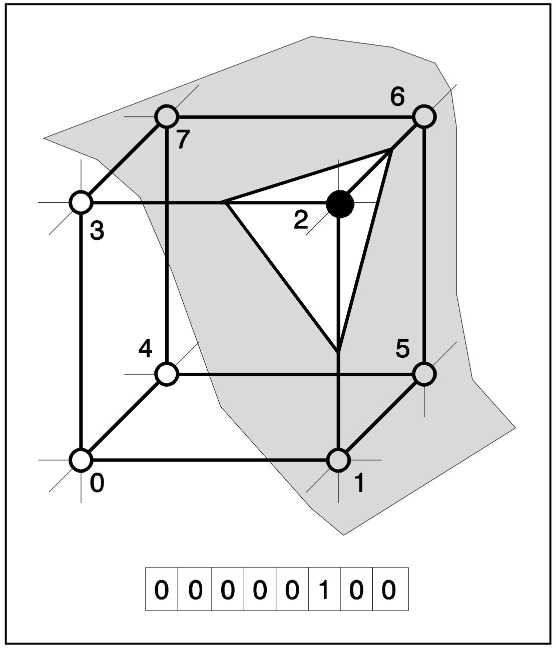
Approach:

- Iso-surface cuts volume = each cell is searched

Idea:

- Divide the Iso-Surface into a per-volume cell representation
- Use triangles





Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

Marching Cubes - Overview

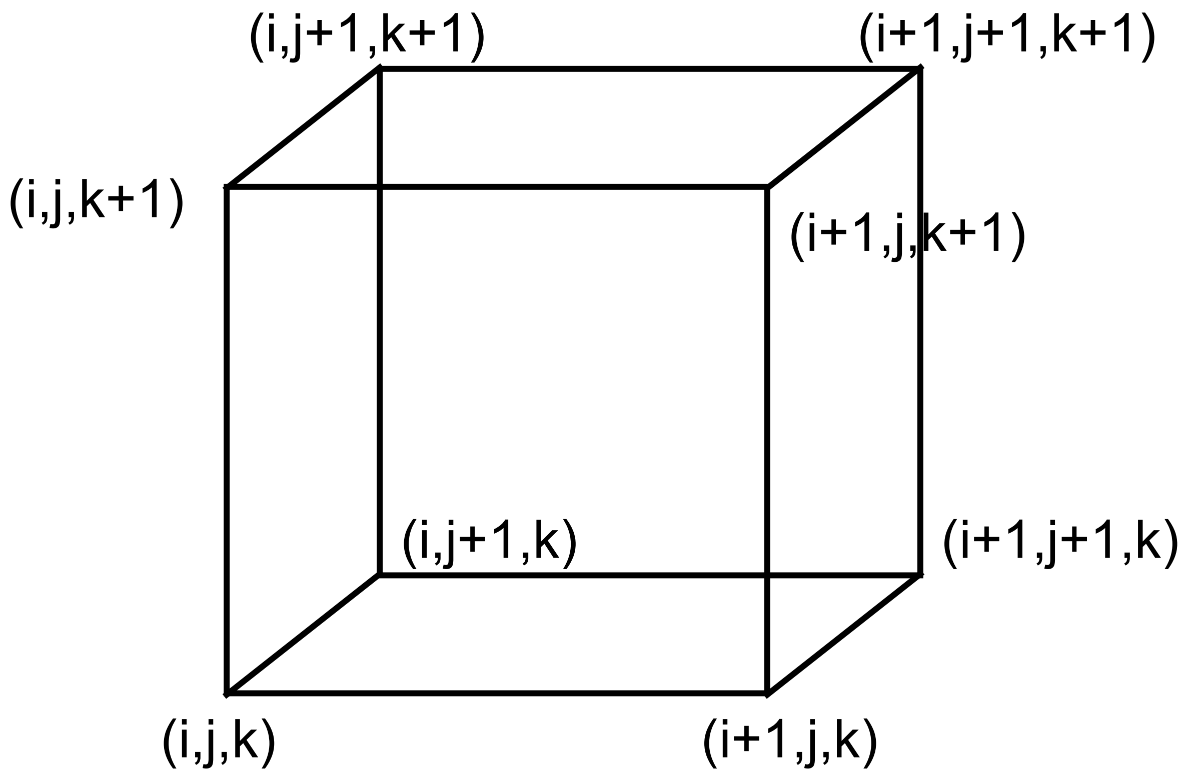
Overview of MC algorithm

- Cell consists of 4(8) pixel (voxel) values:
  (i+[01], j+[01], k+[01])

1. Examine a cell
2. Classify each vertex as inside or outside
3. Build an index
4. Get edge list from table[index]
5. Interpolate the edge location
6. Compute gradients
7. Consider ambiguous cases
8. Go to next cell

Marching Cubes – Step 1

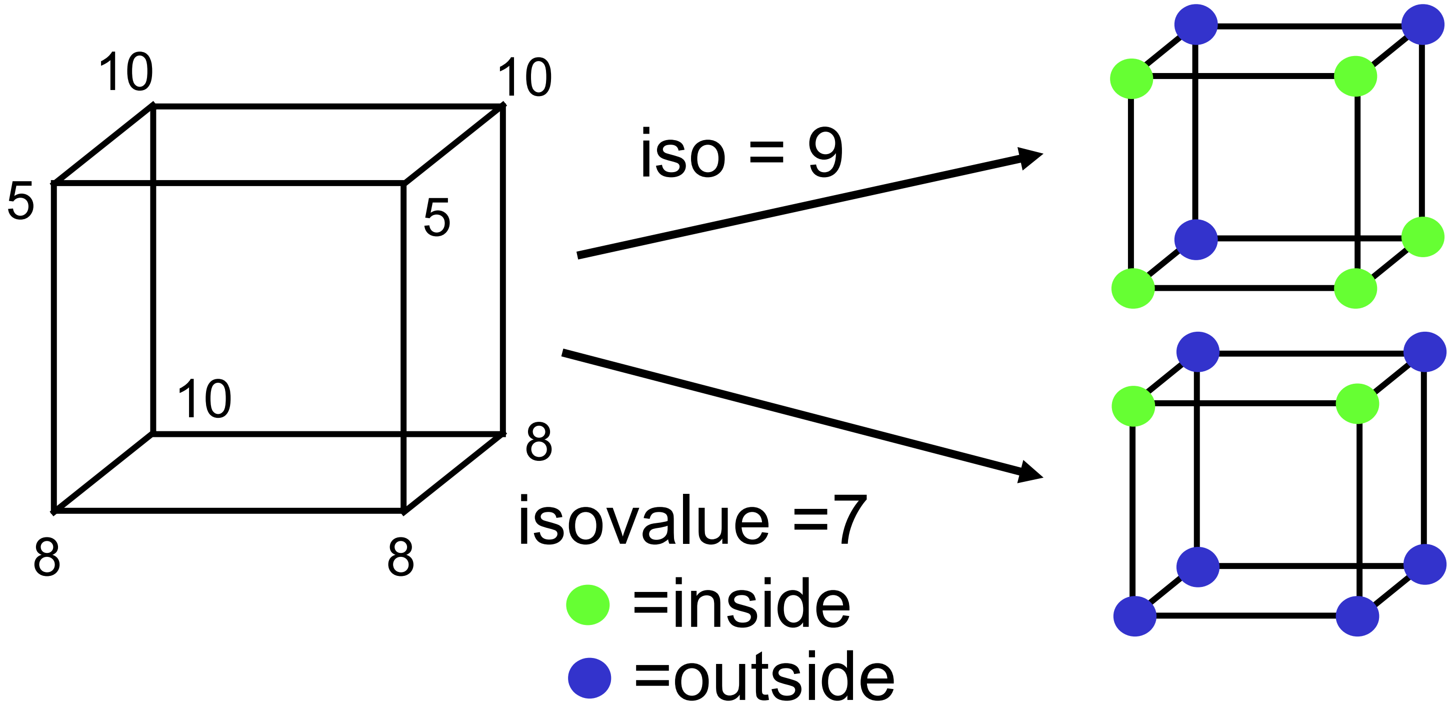
Step 1: Consider a cell defined by eight data values



Marching Cubes – Step 2

Step 2: Classify each voxel according to whether it lies:

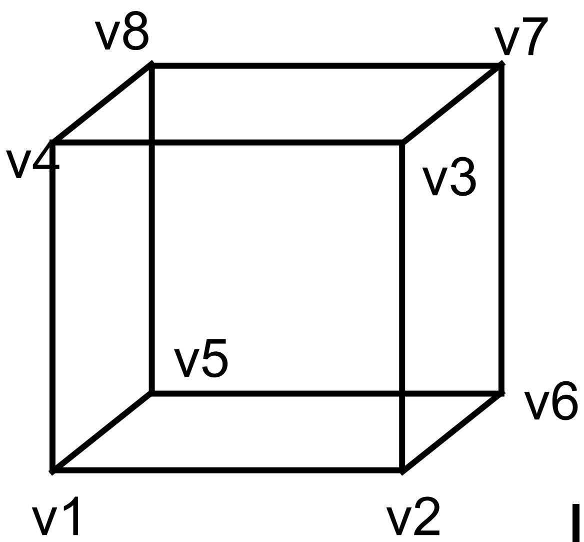
- Outside the surface (value > isosurface value)
- Inside the surface (value <= isosurface value)



Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

Marching Cubes Step 3

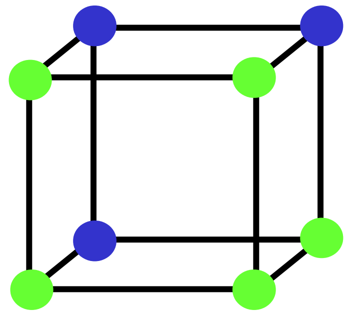
Step 3: Use the binary labeling of each voxel vertex to create an index



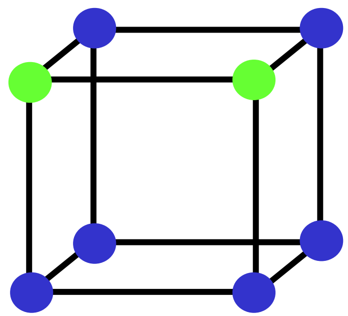
- inside =1
- outside=0

Index:

v1	v2	v3	v4	v5	v6	v7	v8



11110100

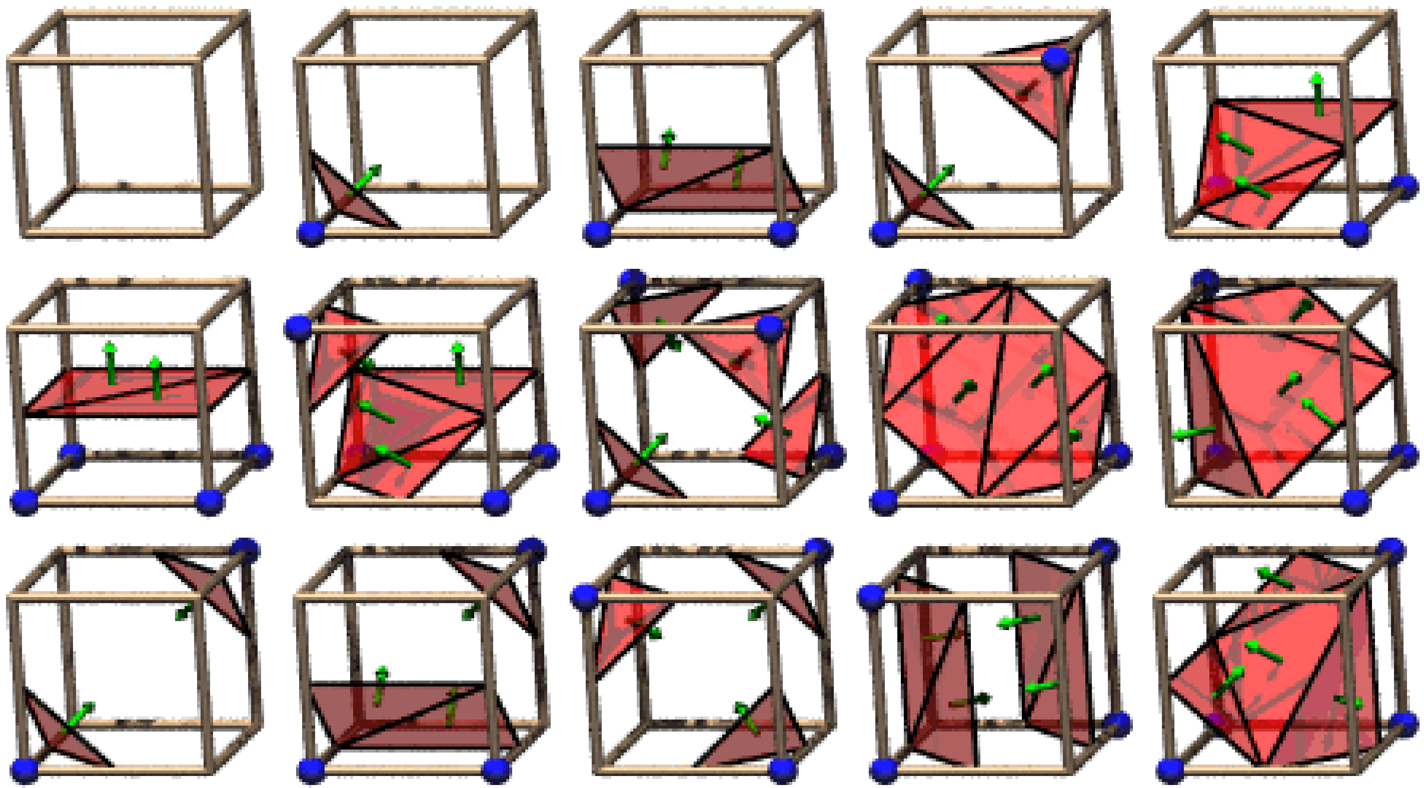


00110000

Marching Cubes – Step 4

Step 4: For a given index, access an array storing a list of edges

All 256 cases can be derived from 1+14=15 base cases due to symmetries



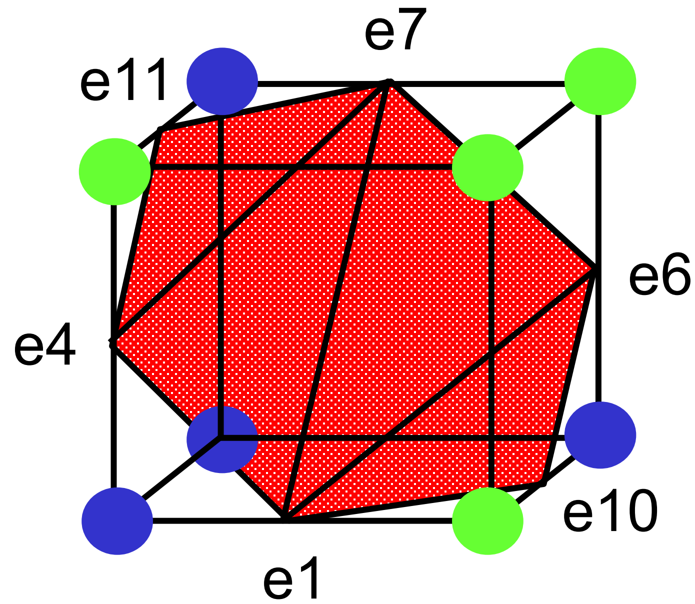
The 15 Cube Combinations

Marching Cubes – Step 4 continued

Step 4 *cont.*: Get edge list from lookup (case) table

- Example for

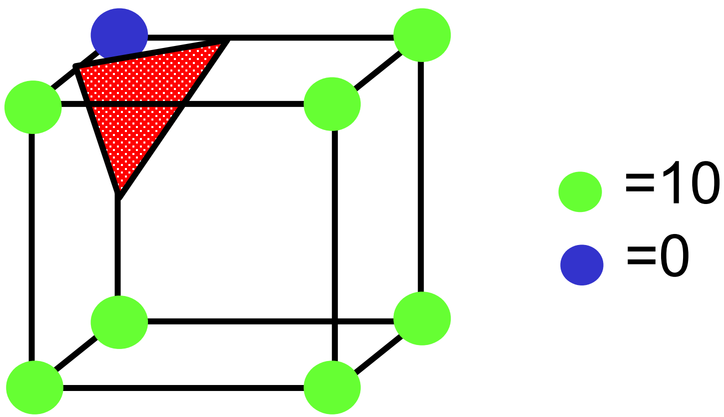
  Index = 10110001
  triangle 1 = e4,e7,e11
  triangle 2 = e1, e7, e4
  triangle 3 = e1, e6, e7
  triangle 4 = e1, e10, e6



Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

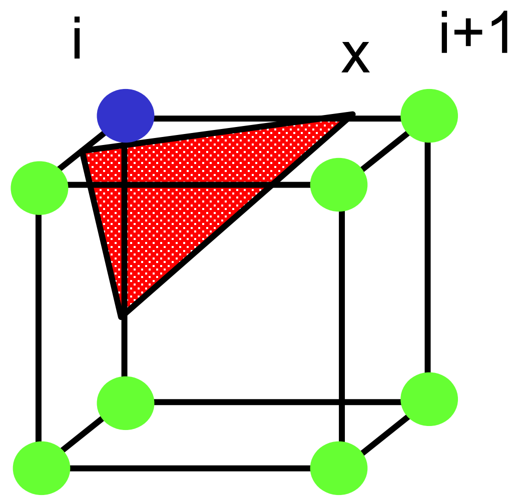
Marching Cubes – Step 5

Step 5: For each triangle edge, find the vertex location along the edge using linear interpolation of the voxel values



T=5

$$x = i + \left(\frac{T - v[i]}{v[i+1] - v[i]} \right)$$



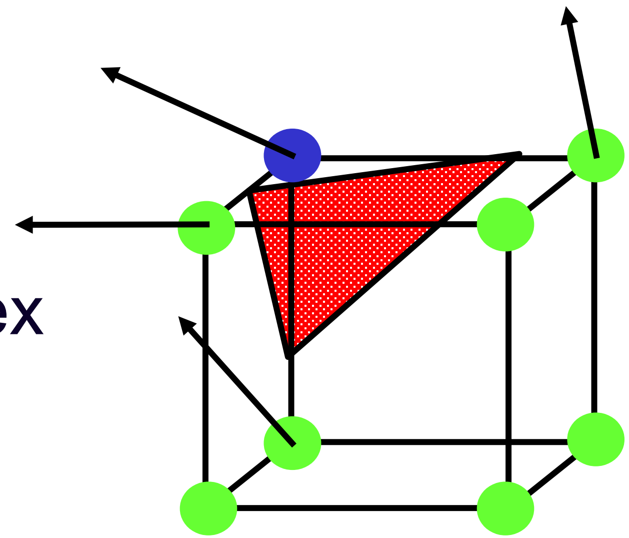
T=8

Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

Marching Cubes – Step 6

Step 6: Calculate the normal at each cube vertex (central differences)

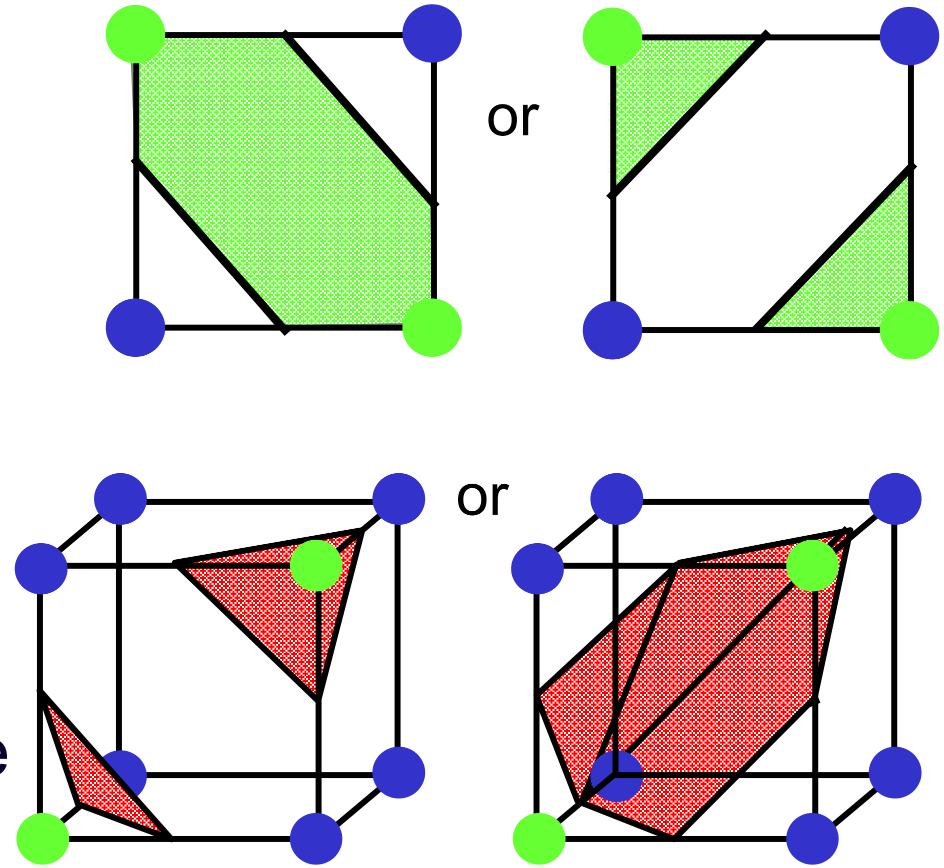
- $G_x = V_{x+1,y,z} - V_{x-1,y,z}$
  $G_y = V_{x,y+1,z} - V_{x,y-1,z}$
  $G_z = V_{x,y,z+1} - V_{x,y,z-1}$
- Use linear interpolation to compute the polygon vertex normal (of the isosurface)



Marching Cubes – Step 7

Step 7: Consider ambiguous cases

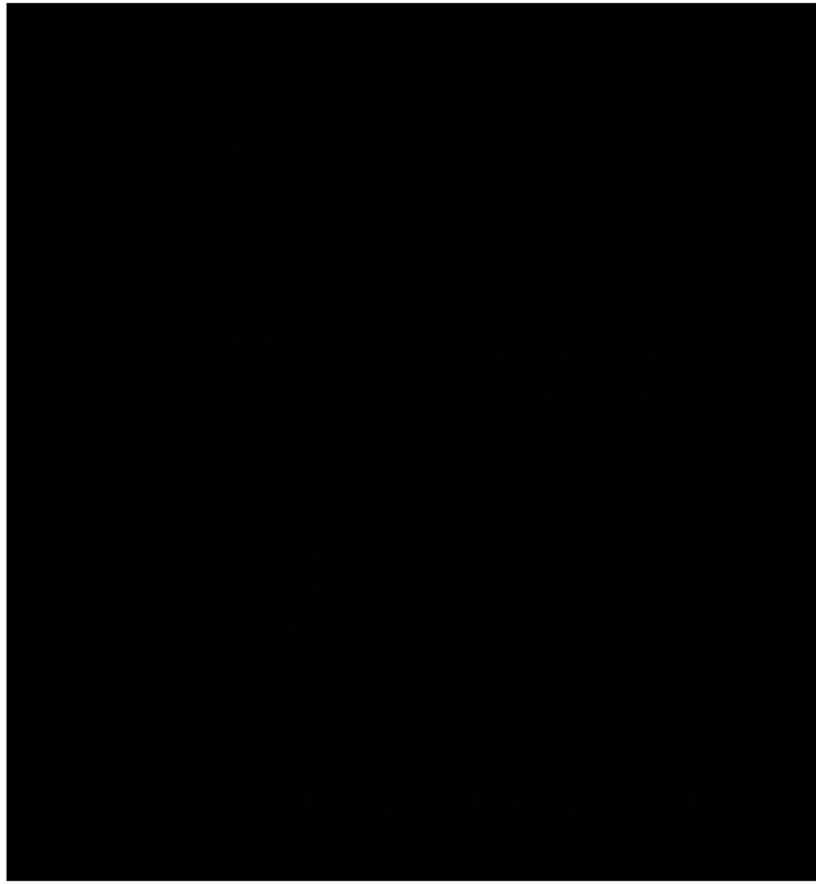
- Adjacent vertices: different states
- Diagonal vertices: same state
- Resolution: choose one case (the right one)



Marching Cubes - Summary

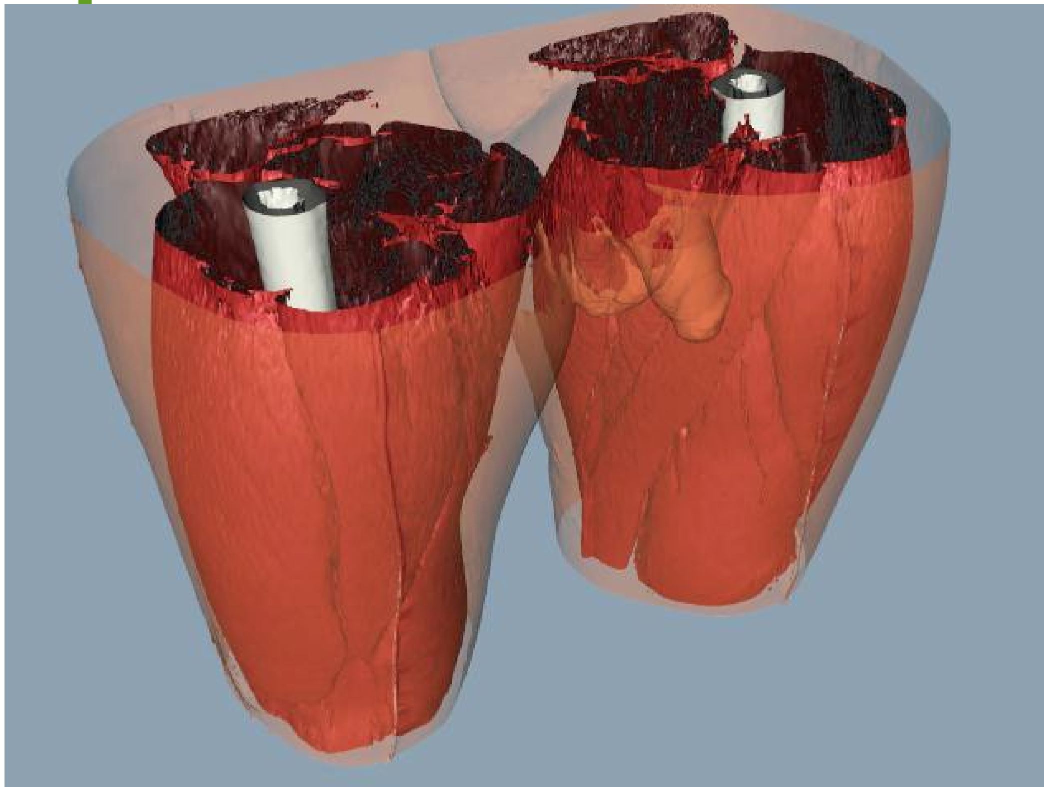
Summary

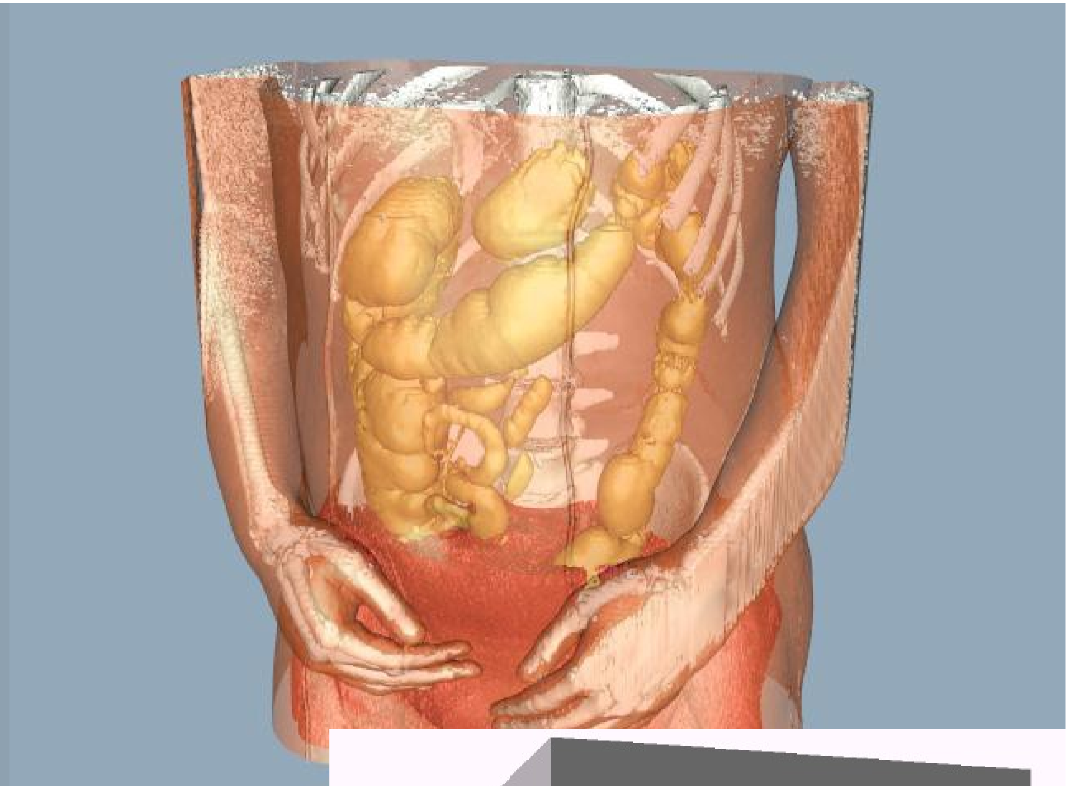
- 256 Cases
- Reduce to 15 cases by symmetry
- Ambiguity in certain cases
- Causes holes if arbitrary choices are made
- Up to 5 triangles per cube

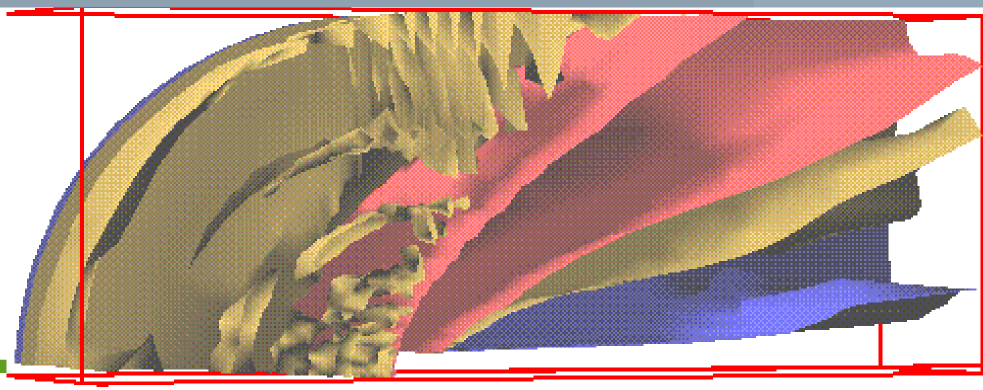


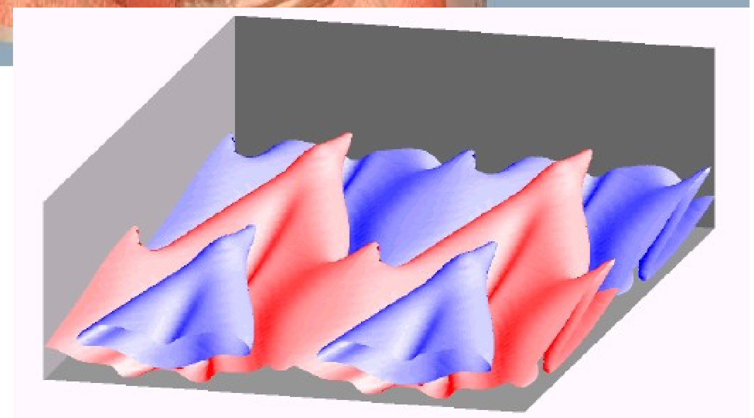
Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

More Examples



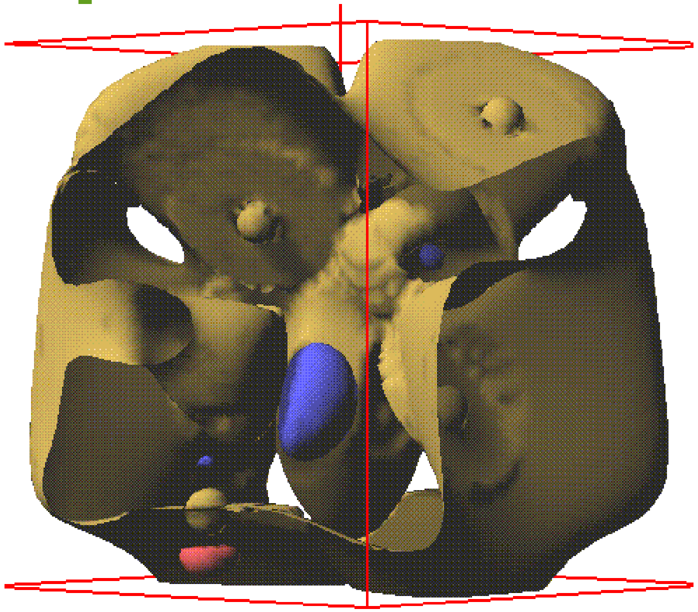


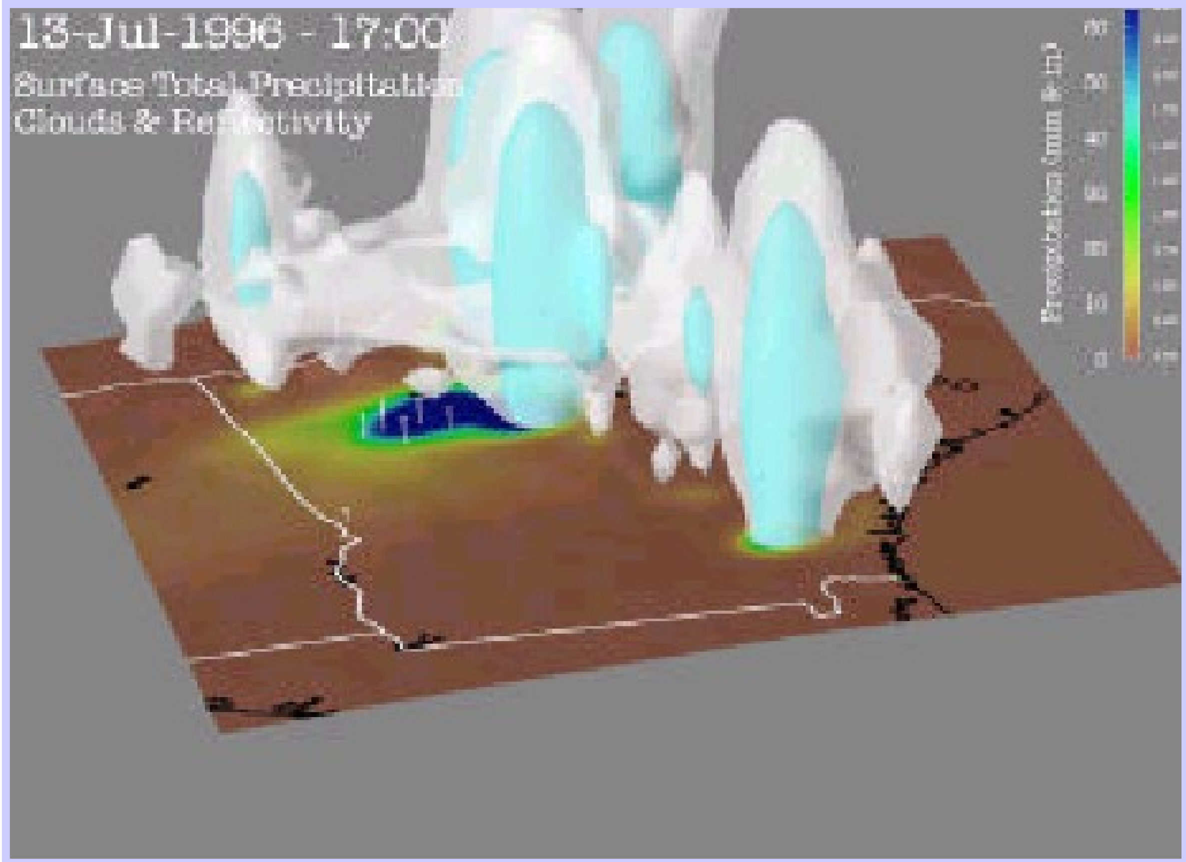


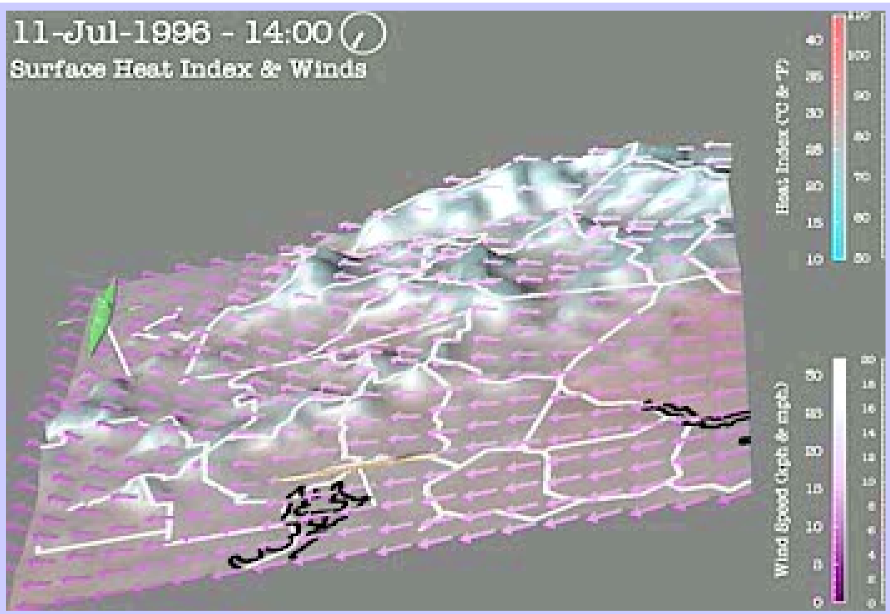


Thomas Torsney-Weir
t.d.torsney-weir@swansea.ac.uk

More Examples







Literature

Paper (more details):

- W. Lorensen & H. Cline: "**Marching Cubes: A High Resolution 3D Surface Construction Algorithm**" in *Proceedings of ACM SIGGRAPH '87 = Computer Graphics*, Vol. 21, No. 24, July 1987

Acknowledgments

For their contribution to the lecture material:

- Michael Meißner
- Roger Crawfis (Ohio State Univ.)
- Hanspeter Pfister
- Meister E. Gröller
- Torsten Möller
- Dirk Bartz
- Markus Hadwiger
- Helwig Hauser
- Daniel Weiskopf