Volume Visualization

Part 3 (out of 3)

Hardware-Volume Visualization

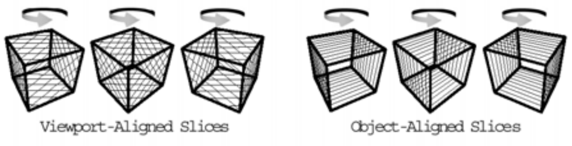
Faster with Hardware?!

Two Approaches

- 3D-textures:
  - Volume data stored in 3D-texture
  - Proxy geometry (slices) parallel to image plane, are interpolated tri-linearly
  - Back-to-front compositing
- 2D-textures:
  - 3 stacks of slices (x-, y- & z-axis), slices are interpolated bi-linearly
  - Select stack (most “parallel” to image plane)
  - Back-to-front compositing

Variation of View Point

- 3D-textures:
  - Number of slices arbitrary
- 2D-textures:
  - Stack change: discontinuity



Special Hardware

- Hardware volume raycasting
  - In vertex and fragment operations of modern graphics cards
- Special Hardware
  - VolumePro board:
    - Special card for PC
    - Calculates shear-warp factorization, incl. compositing
    - Warp-step with "regular" graphics card (OpenGL)

Marching Cubes (MC)

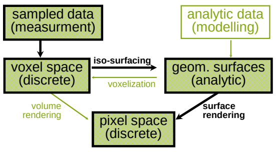
Iso-Surface-Display

Repetition: Volume vs. Surface Rendering

- Volume rendering:
  - Direct volume visualization
  - Usage of transfer functions
  - Pros: look on the interior, semi-transparency
- Surface rendering:
  - Indirect volume visualization
  - Intermediate representation: Iso-surface, “3D”
  - Pros: shading → shape!, hardware rendering

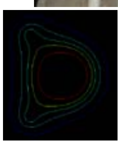
Concepts and Terms

- Example 1:
  - CT measurement
  - Iso-stack-conversion
  - Iso-surface-calculation (marching cubes)
  - Surface rendering (OpenGL)



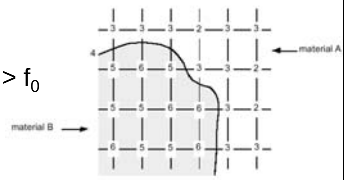
Iso-Surfaces

- Intermediate representation
- Aspects:
  - Preconditions:
    - expressive Iso-value, Iso-value separates materials
    - Interest: in transitions
  - Very selective (binary selection / omission)
  - Uses traditional hardware
  - shading ⇒ 3D-impression!



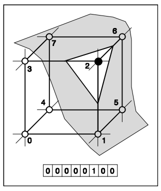
Volume Data ⇔ Iso-Surfaces

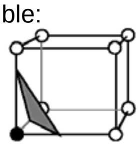
- Iso-Surface:
  - Iso-value f_0
  - separates values > f_0 from values ≤ f_0
  - Often not known →
  - Can only be approximated from samples!
  - Shape / position dependent on type of reconstruction

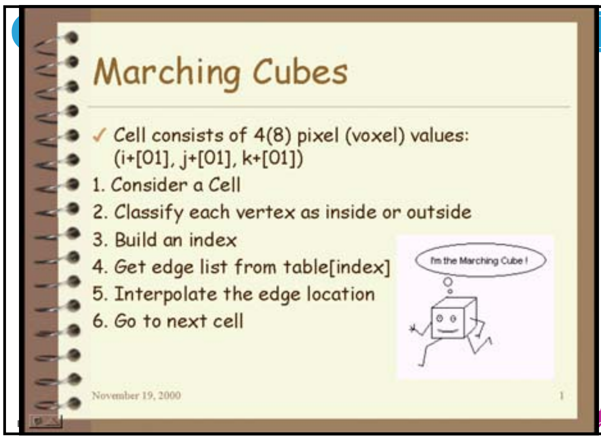


Approximation of Iso-Surface

- Approach:
  - Iso-Surface intersects data volume = set of all cells
- Idea:
  - Parts of iso-surface represented on a(n intersected) cell basis
  - As simple as possible: Usage of triangles

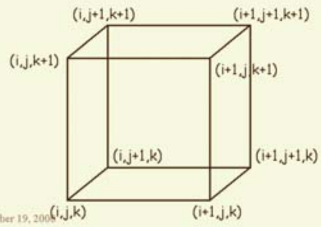






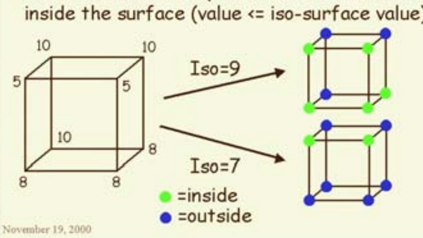
MC 1: Create a Cube

- ✓ Consider a Cube defined by eight data values:



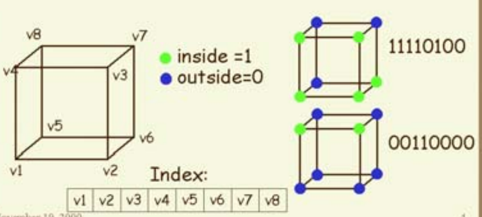
MC 2: Classify Each Voxel

- ✓ Classify each voxel according to whether it lies
  outside the surface (value > iso-surface value)
  inside the surface (value <= iso-surface value)



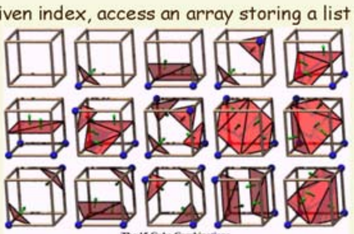
MC 3: Build An Index

- ✓ Use the binary labeling of each voxel to create an index



MC 4: Lookup Edge List

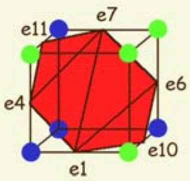
- For a given index, access an array storing a list of edges



- all 256 cases can be derived from 15 base cases

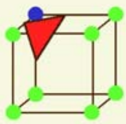
MC 5: Example

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



MC 6: Interp. Triangle Vertex

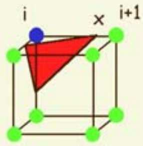
- For each triangle edge, find the vertex location along the edge using linear interpolation of the voxel values



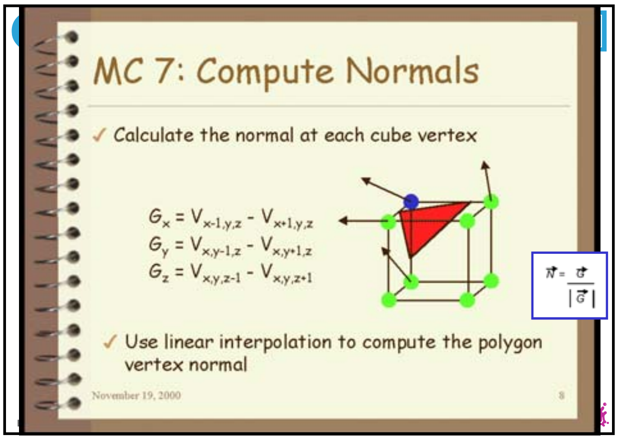
● =10
● =0

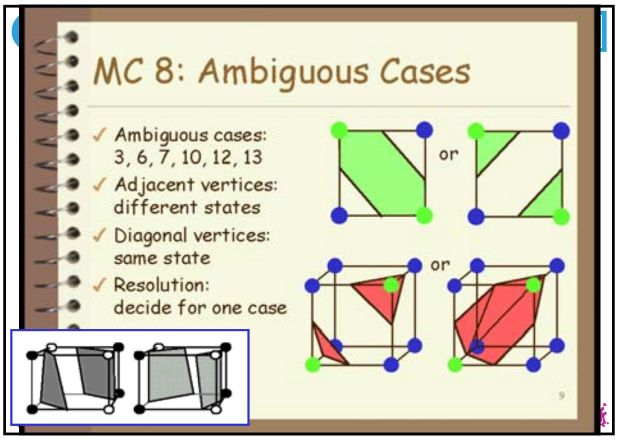
T=5

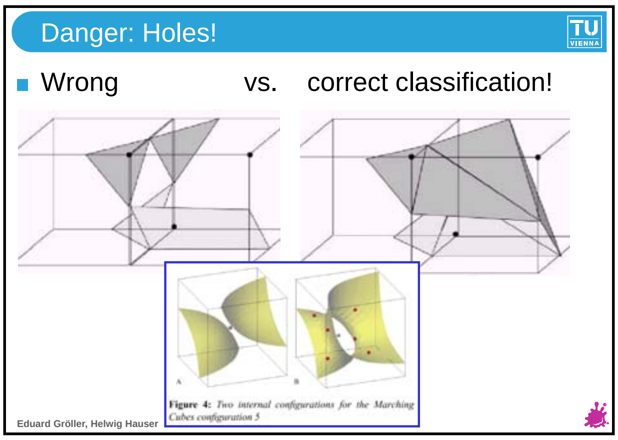
$$x = i + \left(\frac{T - v[i]}{v[i+1] - v[i]} \right)$$



T=8

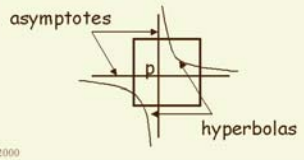

MC 7: Compute Normals
Calculate the normal at each cube vertex
$G_x = V_{x-1,y,z} - V_{x+1,y,z}$
$G_y = V_{x,y-1,z} - V_{x,y+1,z}$
$G_z = V_{x,y,z-1} - V_{x,y,z+1}$
$\vec{N} = \frac{\vec{G}}{|\vec{G}|}$
Use linear interpolation to compute the polygon vertex normal
November 19, 2000
8


MC 8: Ambiguous Cases
Ambiguous cases: 3, 6, 7, 10, 12, 13
Adjacent vertices: different states
Diagonal vertices: same state
Resolution: decide for one case
or
or
9


Danger: Holes!
TU VIENNA
Wrong vs. correct classification!
Figure 4: Two internal configurations for the Marching Cubes configuration 5
Eduard Gröller, Helwig Hauser

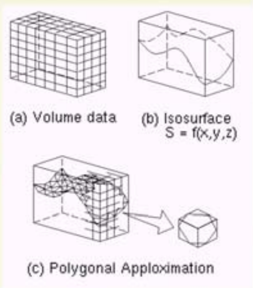
MC 9: Asymptotic Decider

- Assume bilinear interpolation within a face
- hence iso-surface is a hyperbola
- compute the point p where the asymptotes meet
- sign of S(p) decides the connectedness



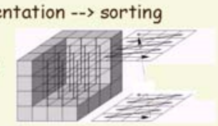
Marching Cubes - Summary 1

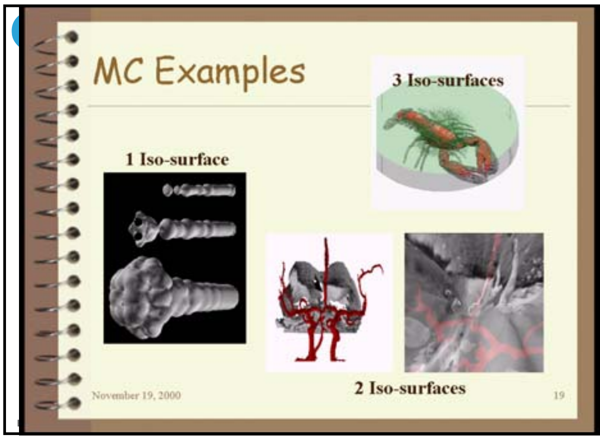
- 256 Cases
- reduce to 15 cases by symmetry
- Complementary cases - (swap in- and outside)
- Ambiguity resides in cases 3, 6, 7, 10, 12, 13
- Causes holes if arbitrary choices are made.



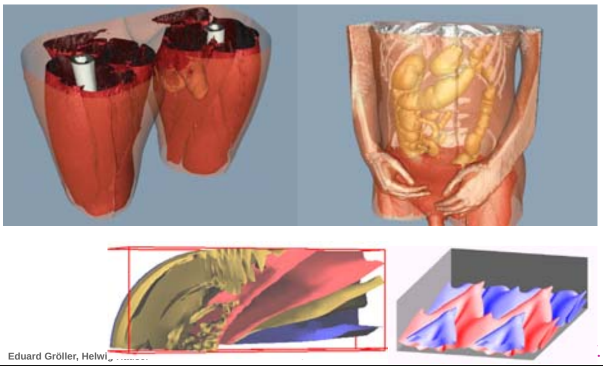
Marching Cubes - Summary 2

- Up to 4 triangles per cube
- Dataset of 512^3 voxels can result in several million triangles (many Mbytes!!!)
- Iso-surface does not represent an object!!!
- No depth information
- Semi-transparent representation --> sorting
- Optimization:
  - Reuse intermediate results
  - Prevent vertex replication
  - Mesh simplification

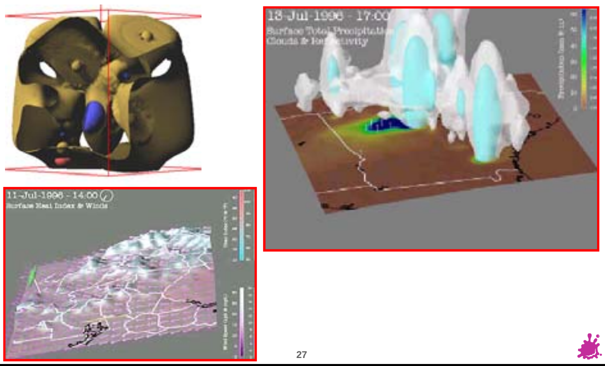




Further Examples



Even Further Examples



Literature

- Paper (more details):
  - W. Lorensen & H. Cline: “**Marching Cubes: A High Resolution 3D Surface Construction Algorithm**” in *Proceedings of ACM SIGRRAPH ’87 = Computer Graphics*, Vol. 21, No. 24, July 1987

Conclusion
Volume Visualization

General Remarks

Surface vs. Volume Rendering

- Surface Rendering:
  - Indirect representation / display
  - Conveys surface impression
  - Hardware supported rendering (fast?!)
  - Iso-value-definition
- Volume Rendering:
  - direct representation / display
  - Conveys volume impression
  - Often realized in software (slow?!)
  - Transfer functions



Conclusion VolVis

- Introduction ⇐ data, simple methods ⇐ DVR vs. surf. fitting
- Direct volume visualization
  - Ray casting ⇐ types of combinations
  - Splatting ⇐ object-order vs. image-order ⇐ speed vs. quality
  - Shear-warp factorization
  - Hardware-based VolVis
- Indirect VolVis ⇐ iso-value selection
  - Marching cubes (iso-surface-visualization)
- Conclusion

Acknowledgments

- For material for this lecture unit:
  - Michael Meißner
  - Roger Crawfis (Ohio State Univ.)
  - Hanspeter Pfister (MERL)
  - Torsten Möller
  - Dirk Bartz
  - Markus Hadwiger