3D image tools

Anthony P. Reeves
School of Electrical and Computer Engineering

Cornell University

Topics

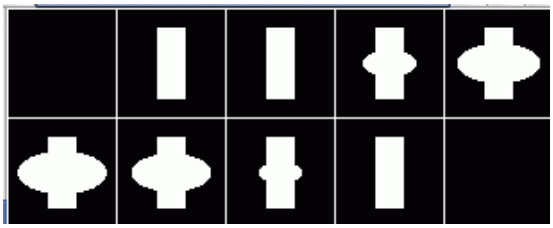
- Generation of 3D synthetic images
- Converting to polygon surface representation
- Filtering polygon representations
- Viewing 3D polygon models

A simple 3D Synthetic test object

```
#!/bin/sh
vgenim s=60 z=50 e=25,12,12 of=im1
vgenim s=60 z=50 r=8,20,20 of=im2
vop -or im1 im2 -o im3
vdim -c im3 -o im4
rm -f im5
vclip im4 f=1 -o  im5
#  make a visualization of the image
for i in 6 11 16 21 26 31 36 41 46
do
   echo "i is is $i"
   vclip im4 f=$i | vxfile of=im5
done
vtile im5 -ib -xb n=5,2 -o im6
```

Im5 is the test object image

For human review:
Im6 is every 5th image of im5 in a tiled single-image presentation

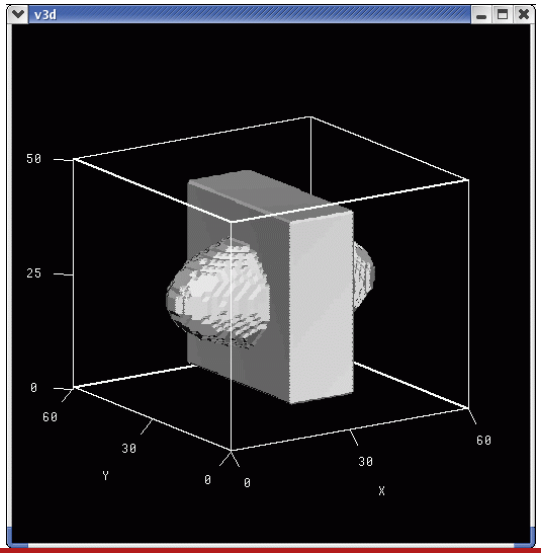


3D Object visualization of the test image

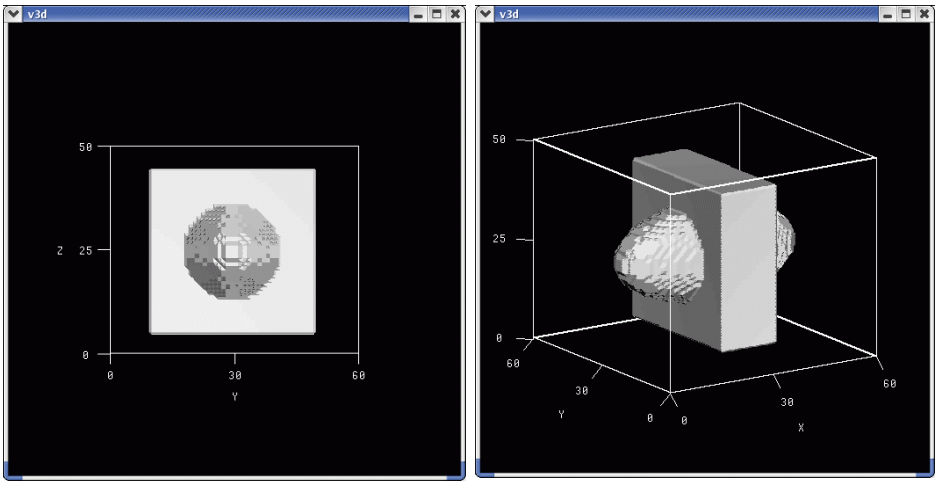
vpol –t im3 –o pol1
v3d pol1

To visualize the surface of a 3D Binary region use vpol to convert to a (triangular) polygon representation and the use v3d to display

Use the interactive controls of v3d to select grey light shaded visualization

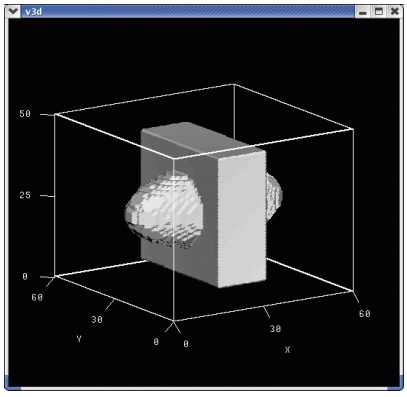


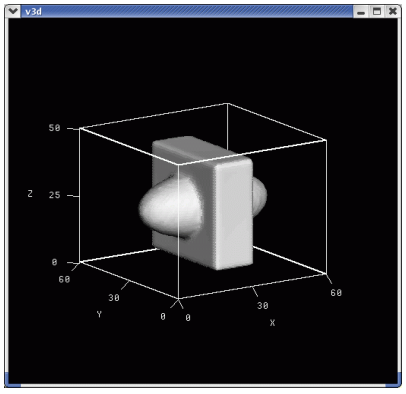
Different viewing directions



The interactive controls fro v3d provide for viewing from different directions

Polygon smoothing



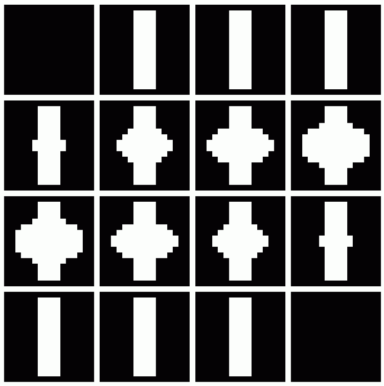


v3pfilt -a pol1 -o pol2
v3pfilt -a pol2 | v3pfilt -a of=pol3

pol3 is shown above, v3pfilt smooths the polygon surface for improved viewing

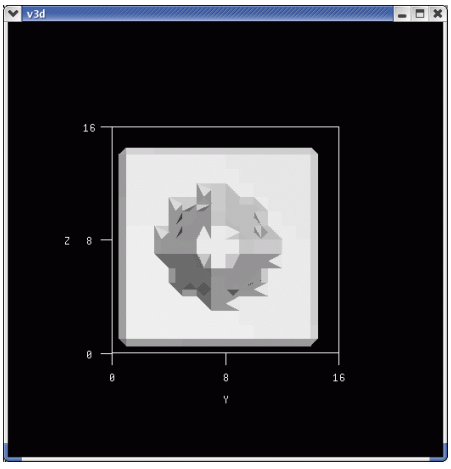
A smaller 3D test image

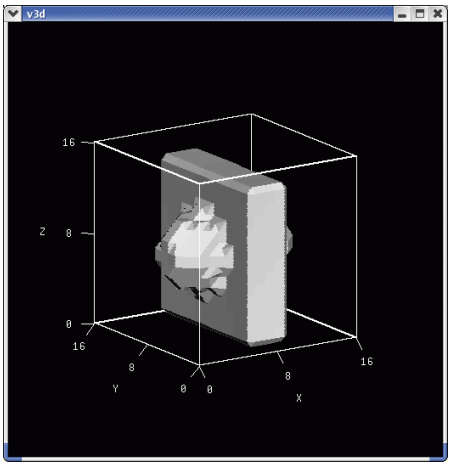
```
#!/bin/sh
vgenim s=16 z=16 e=6,4,4 of=im1
vgenim s=16 z=16 r=2,7,7 of=im2
vop -or im1 im2 -o im3
vdim -c im3 -o im4
# make visualization image
vtile im4 -ib -xb n=4,4 -o im6
#
v3pol -t im3 -o pol1
v3pfilt -a pol1 -o pol2
v3pfilt -a pol2 | v3pfilt -a of=pol3
```



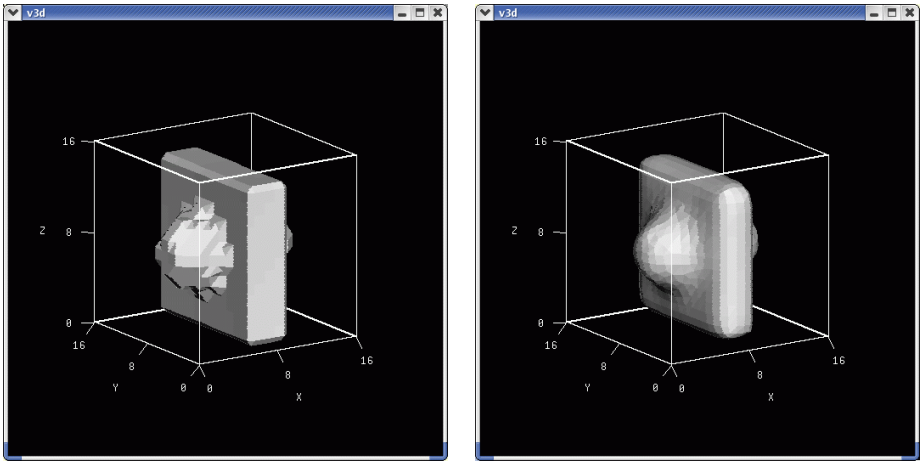
The modified script recreates all the previous images with a smaller size. By making the image smaller we can visualize all the 2D images and better see the effects of polygon smoothing

Small image without smoothing

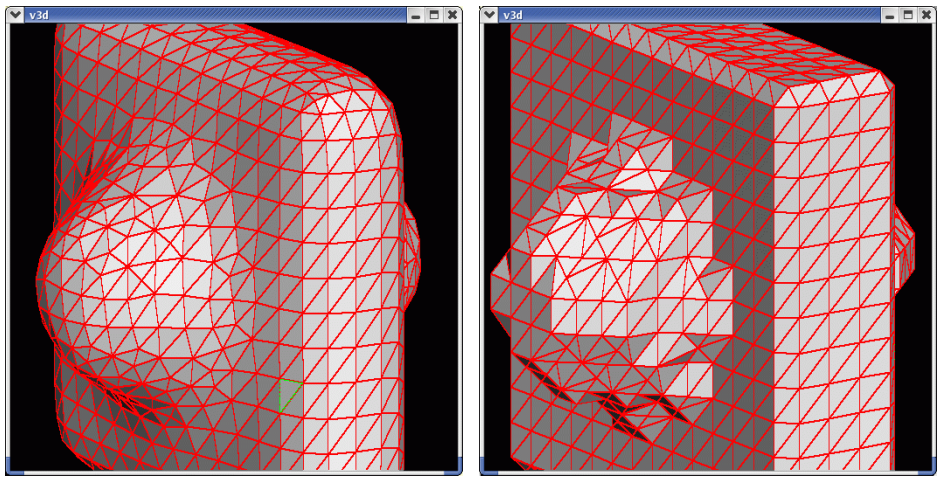




Small image with polygon smoothing



Details of the triangular surface



The Third Image Dimension

Anthony P. Reeves
School of Electrical and Computer Engineering

Cornell University

3D Computer Vision Methods

- Thresholding
- Split and Merge Region Growing
- Image filtering (isotropic)
- Gradient Measurement (isotropic)
- The Method of Moments
- Feature Classification (all methods)

3D Algorithms with Issues

- Image Filtering (anisotropic)
  - how to define an anisotropic kernel?
- Edge Detection (anisotropic)
  - Non-maximum suppression
  - Thresholding with hysteresis
  - complex neighborhood decisions

3D Lines and 3D Surfaces

- B-Splines are well defined for 3D lines and 3D surfaces
- Splines that pass through knot points can be defined for 3D lines and 3D surfaces
- How to decompose a volume to a surface representation?
  - computer graphics: surface polygon models and volume models

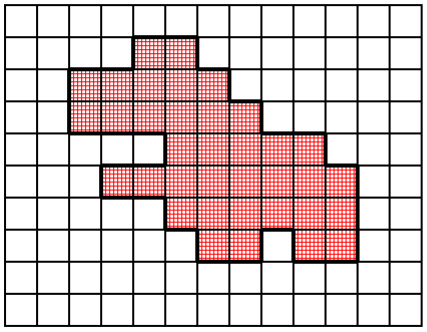
Polygon Representation

- Marching cubes: convert a thresholded 3D image to a set of 3D surface Polygons
  - Concept similar to boundary tracing
- 3D Polyline Algorithm?

2D Boundary Representation and 3D Surface Representation

- Crack Edge (2D and 3D)
- Chain Code (2D)
- Marching cubes (2D version)
- 3D Marching cubes

Crack Edge Model

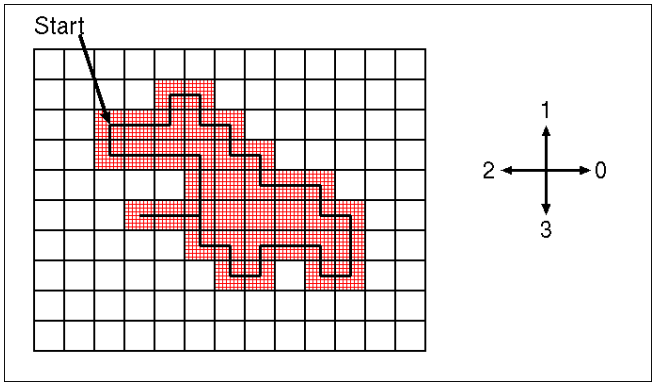


The crack edge model may be extended to 3D where each "crack" is a square polygon having the size of one voxel face

e.g. Brice and Fennema, Boundary melting

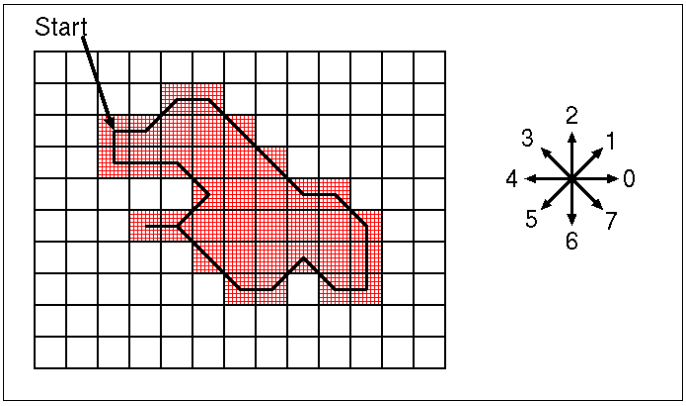
Chain Code

Trace the boundary of a region using a boundary following algorithm and record the trajectory from pixel to pixel with an efficient direction code.



0,0,1,0,3,0,3,0,3,0,0,3,0,3,3,2,1,2,2,3,2,1,2,1,2,2,0,0,1,1,2,2,2,1

Eight-Direction Chain Code



0,1,0,7,7,7,0,7,6,6,4,3,5,4,3,3,4,0,1,3,4,4,2

- For the example image: 4-Dir. 34x2 = 68 bits 8-Dir. 23x3 = 69 bits
- A 3D volume could be represented by set of CCs one for each "slice"

Marching Cubes

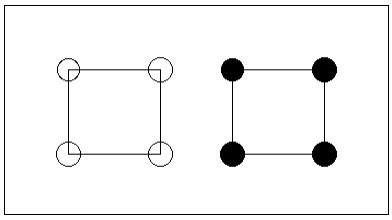
How to convert from a binary volume image to a smooth polygon surface?

- Consider partitioning the space between 1 voxels and 0 voxels
- Local 2x2x2 partitioning algorithm produces a global solution

2D Version (Marching Squares?)

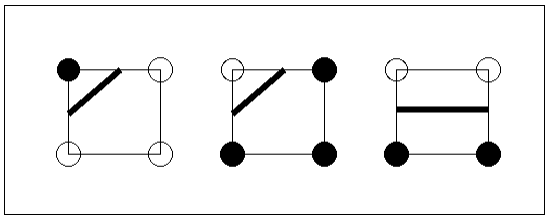
- Partition the space between each 2x2 group of adjacent pixels
- There are sixteen possible arrangements of 2x2 binary pixels

All pixels have the same value



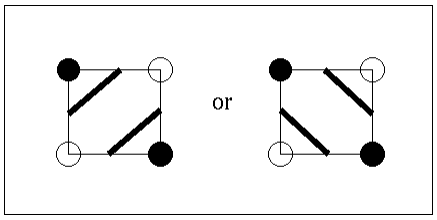
- 2 arrangements
  - all one ⇒ no partition
  - all zero ⇒ no partition

Simple partition arrangements



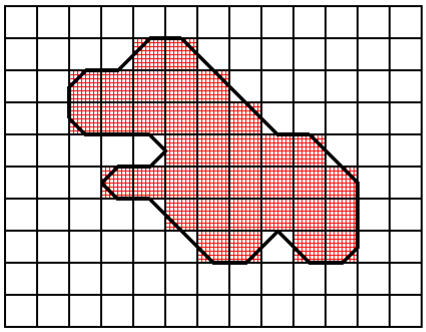
- There are 4 different orientations for each of the above arrangements
- 12 arrangements in total

One final challenging arrangement

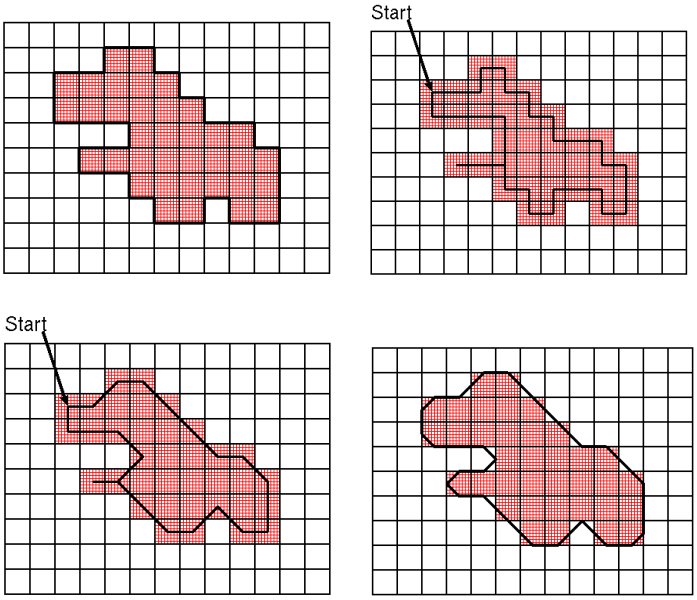


- There are two orientations of the above arrangement and one of two possible partitions must be selected
- This selection determines the connectivity convention (4 or 8)

Marching Squares Boundary



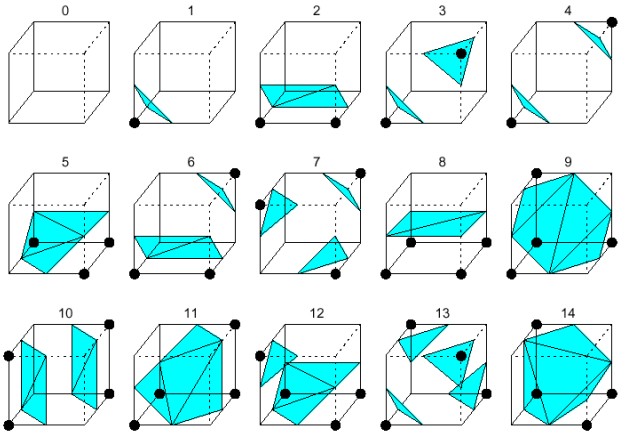
2D Boundary Codes



Marching Cubes (3D)

- There are 256 2x2x2 arrangements
- Partitioning is achieved by polygons
- Frequently, for convenience, polygons are restricted to triangles

Marching Cubes



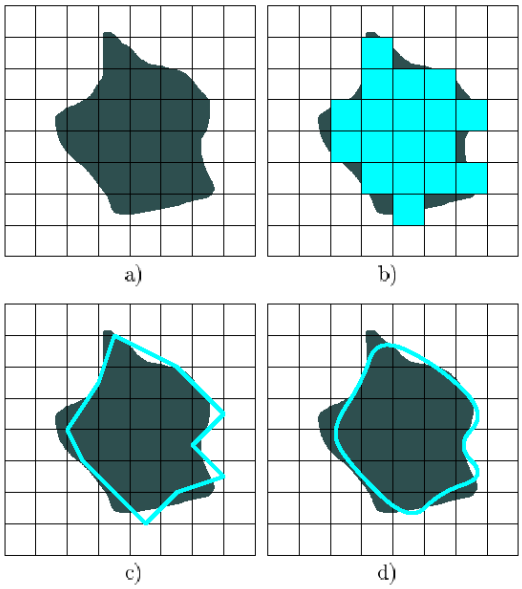
- 14 different cube arrangements and their triangular polygon partitioning
- Several arrangements require 4 triangles for the partition

Summary: Boundary and Surface Representation

- A 2D boundary can be efficiently represented by a chain code (4-dir or 8-dir)
- The surfaces of the crack edges (square planes) could be used for volume surface representation
- Marching cubes provides a better solution for volume surface representation

Boundary Location? (2D)

(a) region
(b) Pixel Representation
c) Marching Cubes
d) Smoothed Marching Cubes



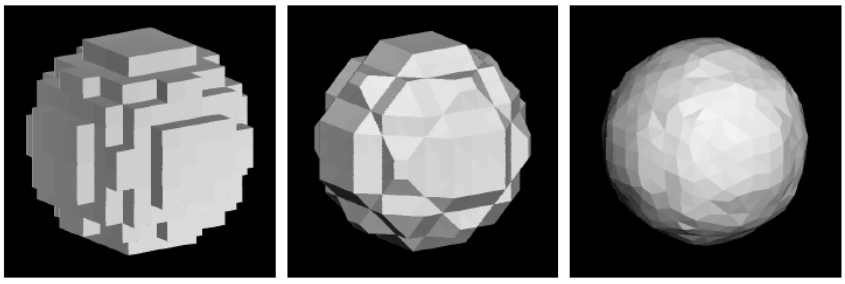
Pixel resolution limits the accuracy and quality of a boundary representation

3D Region Visualization

How to create a 3D model ?

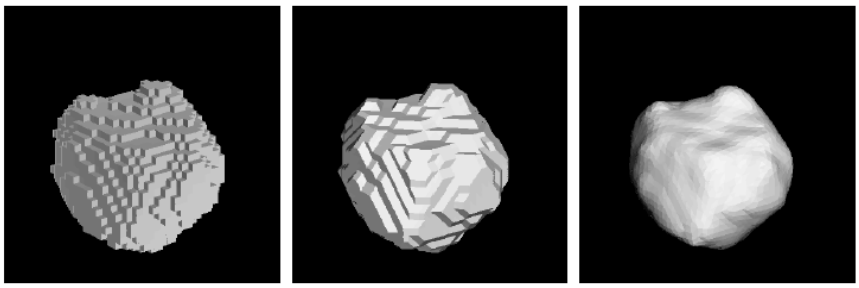
- Threshold 3D image
  - obtain 3D binary image
- Marching Cubes algorithm
  - obtain 3D polygon representation from 3D image
- Smoothing
  - remove voxel quantization artifacts

3D Surface Representations of a Sphere



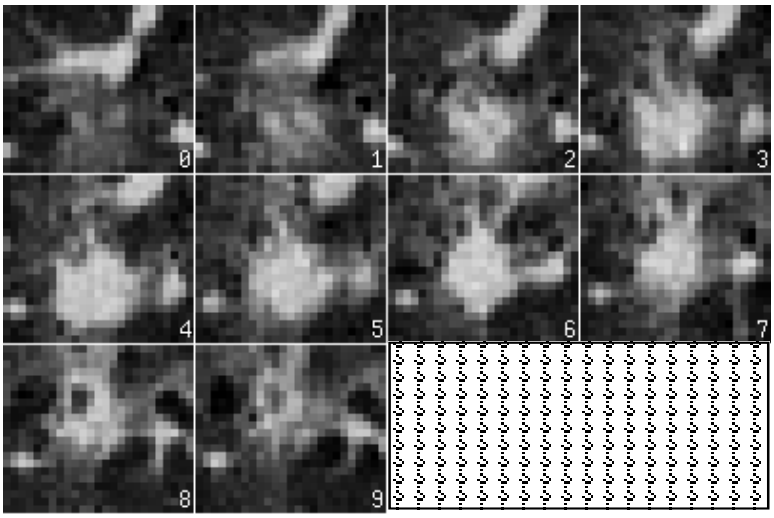
(a) Crack Edge (b) marching cubes (c) Smoothed MC

3D Surface Representations of a pulmonary nodule

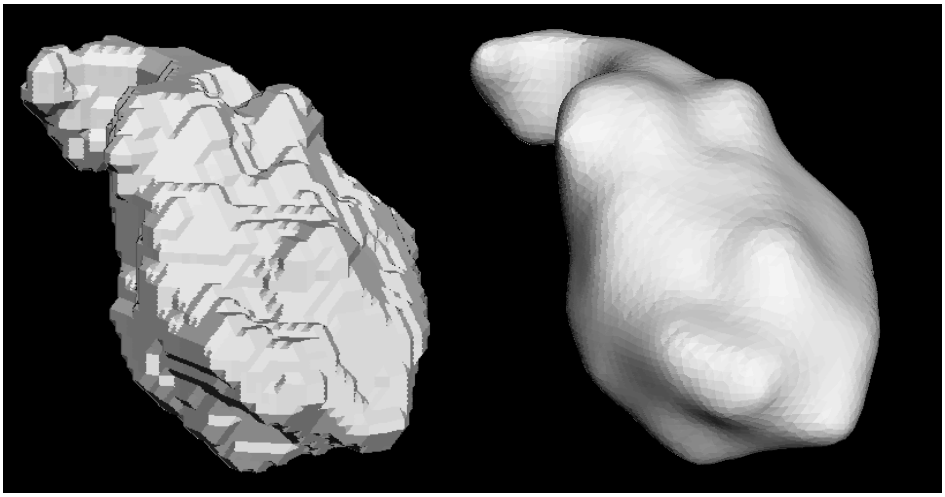


(a) Crack Edge (b) marching cubes (c) Smoothed MC

A Pulmonary Nodule: The CT Slices



Shaded Light Marching Cubes and Smoothing



Smoothing Methods

- 2D Boundary smoothing by local filtering
  - Replace each boundary point by a weighted sum of its neighbors
- 3D Polygon smoothing
  - Replace each vertex by a weighted sum of the neighboring vertices
- Repeat process several times for a Gaussian like filter function
- Issues: what weights to use and how many repetitions; i.e., how much smoothing

2D Boundary Smoothing

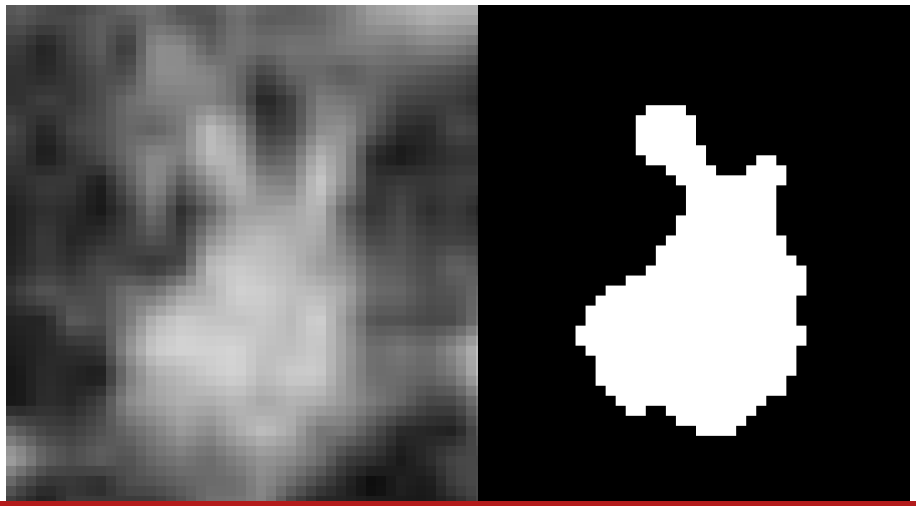
- Given a boundary vector

$$p[i] = (x[i], y[i])$$

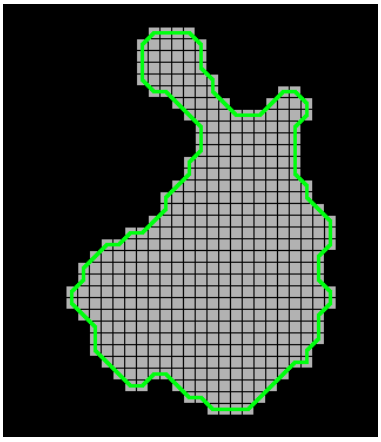
- Replace each element with a weighted fraction of its two adjacent neighbors

$$p'[i] = \alpha\left(\frac{p[i-1] + p[i+1]}{2}\right) + (1-\alpha)\,p[i]$$

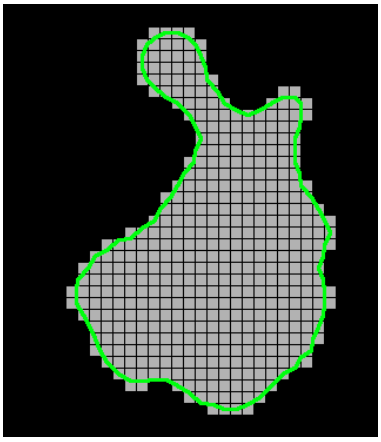
2D CT Image and Segmentation of a Pulmonary Nodule



Chain code and weight filter



8-direction Chain code

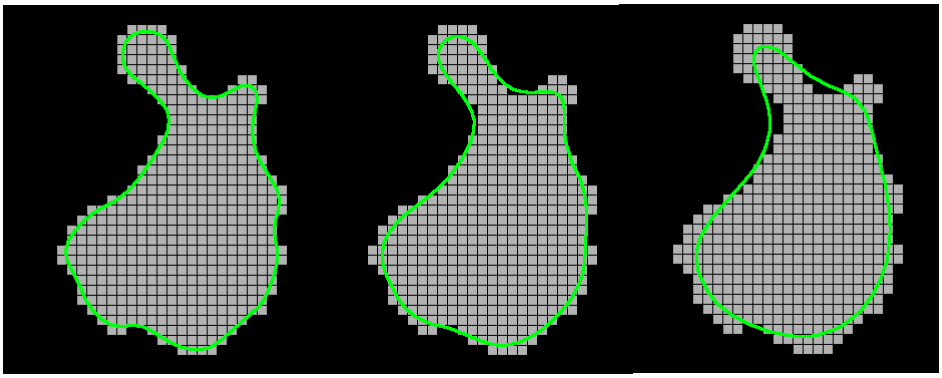


0.9 weight filter

Repeated Operation

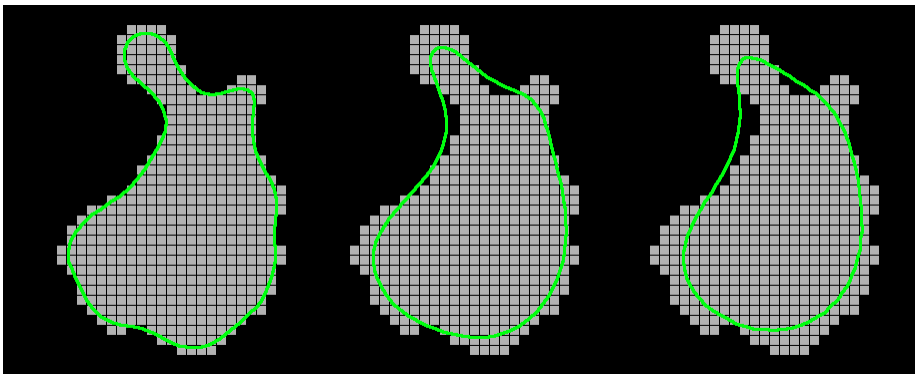
- Simple weighted smoothing will only change boundary location by up to 1 pixel.
- Repeat the weighted smoothing operation *n* times; the smoothing function approximates Gaussian weighting of the near neighbors

Boundary smoothing α = 0.5



n = 3 *n* = 10 *n* = 30

Boundary smoothing n = 30



α = 0.1 α = 0.5 α = 0.99

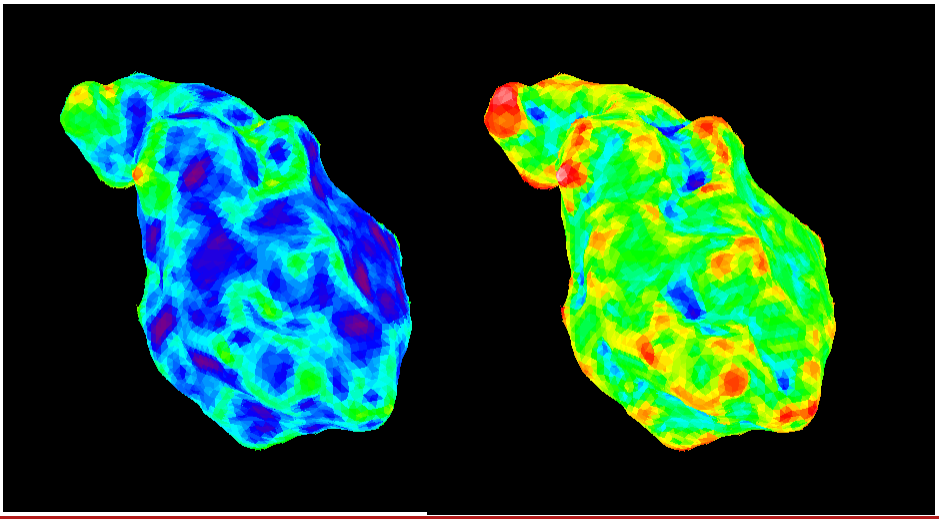
3D Surface Properties

Why bother with a 3D polygon representation (beyond a pretty picture) ?

- Computation of object properties:
  - volume, surface area, center of mass
- Characterization of surface
  - surface curvature
- Object manipulation and comparison
  - scale, rotation, and translation

Curvature

Magnitude | Magnitude + Direction



Polygon Shading

- Light model
  - Lambertian surface, single light source
- Range shading (radar)
  - distance to the viewer
- Dimension shading
  - shade across one of the three dimensions
- Wire frame
  - outlines of the polygons
- Custom shading
  - shade using some object property

Summary: Surface Visualization Methods

- Smoothing
  - reduce voxel quantization and marching cube artifacts
- Rendering Methods
  - range, light, axis --- for depth cues
  - polygon boundaries --- for depth cues
  - outside surface --- for context
  - inside section --- for density distribution
  - gradient --- for surface characterization
  - animation --- additional views and depth cues