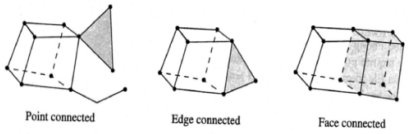
More Advanced Algorithms

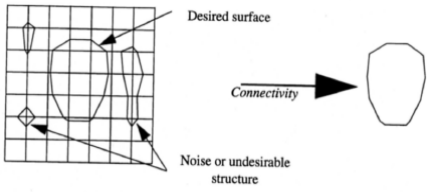
- Connectivity
  - Filter extracts a connected region using inter-cell connectivity (topological property)
  - Can extract the largest connect region encountered
  - Can specify a region (seed point or cell) and extract connected cells that share common features (effective for separating out the parts of a dataset)

Connected Cells

- Cells are connected when they share boundary features such as points, edges, or face



Isosurface Extraction Using Connectivity

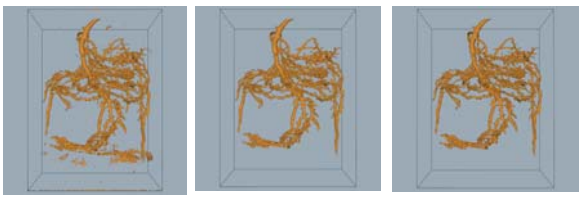


Undesirable portions of an isosurface can be eliminated via connectivity algorithms

Connectivity Algorithm

- To create a connected "surface"
  - Choose an arbitrary cell and mark it as visited
  - Find the cell's neighbors based on a specified criteria (point, edge, or face connected) and mark them as visited
  - Repeat the process recursively for each of the neighbor cells (essentially finding the transitive closure of those cells connected to the first cell that was selected)
  - Assign the surface a unique number
- Identify additional connected surfaces by choosing another "unvisited" cell
- Continue the process until all the cells in the dataset are visited
- Optionally, set a maximum recursion depth to prematurely terminate the recursion to save computation time for large datasets (tradeoff)

Connectivity & Decimation



351,118 triangles
Note the many small, disconnected surfaces due to noise

299,480 triangles
consisting of the largest topologically connected surface

81,111 triangles
after decimation

```
vtkMCubesReader reader
  reader SetFileName "$env(VTK_DATA_ROOT)/Data/pineRoot/pine_root.tri"

vtkPolyDataConnectivityFilter connect
  connect SetInput [reader GetOutput]
  connect SetExtractionModeToLargestRegion

vtkPolyDataMapper isoMapper
  isoMapper SetInput [connect GetOutput]
  isoMapper ScalarVisibilityOff

vtkActor isoActor
  isoActor SetMapper isoMapper
  eval [isoActor GetProperty] SetColor $raw_sienna

vtkOutlineFilter outline
  outline SetInput [reader GetOutput]
vtkPolyDataMapper outlineMapper
  outlineMapper SetInput [outline GetOutput]
vtkActor outlineActor
  outlineActor SetMapper outlineMapper
  [outlineActor GetProperty] SetColor 0 0 0
```

connPineRoot.tcl

```
vtkMCubesReader reader
  reader SetFileName "$env(VTK_TEXTBOOK_DATA)/pineRoot/pine_root.tri"
  reader FlipNormalsOff
vtkDecimate deci
  deci SetInput [reader GetOutput]
  deci SetTargetReduction 0.9
  deci SetAspectRatio 20
  deci SetInitialError 0.0005
  deci SetErrorIncrement 0.001
  deci SetMaximumIterations 6
  deci SetInitialFeatureAngle 30
vtkConnectivityFilter connect
  connect SetInput [deci GetOutput]
  connect SetExtractionModeToLargestRegion
vtkDataSetMapper isoMapper
  isoMapper SetInput [connect GetOutput]
  isoMapper ScalarVisibilityOff
vtkActor isoActor
  isoActor SetMapper isoMapper
  eval [isoActor GetProperty] SetColor $raw_sienna

vtkOutlineFilter outline
  outline SetInput [reader GetOutput]
vtkPolyDataMapper outlineMapper
  outlineMapper SetInput [outline GetOutput]
vtkActor outlineActor
  outlineActor SetMapper outlineMapper
  [outlineActor GetProperty] SetColor 0 0 0
```

decimate first

deciPineRoot.tcl

vtkConnectivityFilter

- vtkConnectivityFilter behavior can be modified by turning on the boolean ivar ScalarConnectivity
  - If this flag is on, the connectivity algorithm is modified so that cells are considered connected only if 1) they are geometrically connected (share a point) and 2) the scalar values of one of the cells' points falls in the scalar range specified
  - ScalarConnectivity can be used to implement a simple "connected segmentation" algorithm. For example, by using a seed voxel (i.e., cell) on a known anatomical structure, connectivity will pull out all voxels "containing" the anatomical structure which can then be contoured or processed by other visualization filters

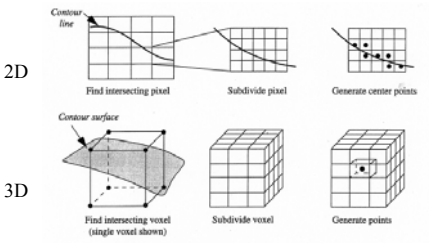
vtkPolyDataConnectivityFilter

- This filter is specialized and optimized for polygonal data
  - This means it runs faster and is easier to construct visualization networks that process polygonal data

Dividing Cubes

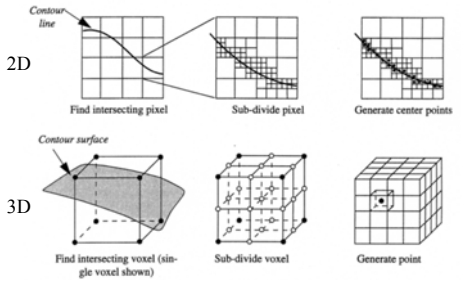
- Similar to marching cubes but generates points rather than triangles (3D) or lines (2D marching squares)
- Generally, rendering points is much faster than rendering polygons (but may not always be significant depending on the hardware)
- Simplifies some geometric operations such as clipping and merging by using points rather than polygons
- May result is disconnected surfaces that may be noticeable depending on the resolution (tradeoff)

Dividing Cubes Algorithm

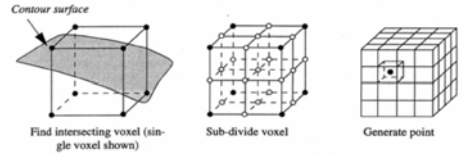


- Voxels through which the contour passes are subdivided into subvoxels
- The number of divisions (n_i) along axes x_i is controlled: $n_i = w_i/R$ in which w_i is voxel width and R is the screen resolution
- A center point is created if the contour passes through a subvoxel

Recursive Dividing Cubes



The sub-divide process repeats until the size of the subvoxel is ≤ screen resolution



- Don't subdivide once, but create a subdivision of 8 subvoxels resulting in 19 new points (12 midedge, 6 midface, and 1 midvoxel point)
- Repeat subdivision for each voxel in which the contour passes until the size of the subvoxel ≤ screen resolution
- Advantage is that the recursive approach will terminate early in those regions of the voxel where the contour cannot pass; however, more points can be generated

vtkDividingCubes (patented?)

- vtkDividingCubes is a filter that generates points lying on a surface of constant scalar value (i.e., an isosurface)
- Dense point clouds (i.e., at screen resolution) will appear as a surface. Less dense clouds can be used as a source to generate streamlines or to generate "transparent" surfaces
- The density of the point cloud is controlled by the Distance instance variable. This the approximate distance between points

THE UNIVERSITY OF MEMPHIS.

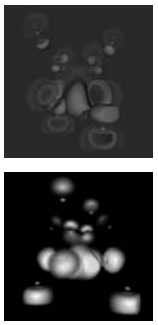
```
vtkStructuredPointsReader reader
   reader SetFileName "$VTK_DATA_ROOT/Data/ironProt.vtk"

vtkDividingCubes iso
   iso SetInput [reader GetOutput]
   iso SetValue 128
   iso SetDistance 1
   iso SetIncrement 1

vtkPolyDataMapper isoMapper
   isoMapper SetInput [iso GetOutput]
   isoMapper ScalarVisibilityOff

vtkActor isoActor1
   isoActor1 SetMapper isoMapper
   eval [isoActor1 GetProperty] SetDiffuseColor $banana
   eval [isoActor1 GetProperty] SetDiffuse .7
   eval [isoActor1 GetProperty] SetSpecular .5
   eval [isoActor1 GetProperty] SetSpecularPower 30

vtkActor isoActor2
   isoActor2 SetMapper isoMapper
   eval [isoActor2 GetProperty] SetDiffuseColor $banana
   eval [isoActor2 GetProperty] SetDiffuse .7
   eval [isoActor2 GetProperty] SetSpecular .5
   eval [isoActor2 GetProperty] SetSpecularPower 30
   eval [isoActor2 GetProperty] SetPointSize 5
```



THE UNIVERSITY OF MEMPHIS.