Volumetric Visualization

Outline

- Announcements
  - PS III due Friday
  - Last day for self-motivated assignment
- Belated CookiePresentation
- What is VV?
- Slices
- Isosurfaces
- Movies

What is VV?

- $y = f(x) \Rightarrow$ lines
- $z = f(x, y) \Rightarrow$ surfaces
- $v = f(x(s), y(s), z(s)) \Rightarrow$ surfaces with color $\neq z$
- $v = f(x, y, z) \Rightarrow$ true 3D data?
Representing $V = f(x,y,z)$

- we have a value of $V$ for every point in a volume
- $V$ is a cube of data (m-by-n-by-p)
- We need $X$, $Y$, $Z$ of the same size to indicate positions of data
  - Typically, we have a regular grid defined by vectors $x$, $y$, $z$
  - $[X,Y,Z]=\text{meshgrid}(x,y,z)$ produces 3D arrays needed by Matlab’s VV functions
  - $X$, $Y$, $Z$ are m-by-n-by-p
  - for all $j$ & $k$, $X(j,:,k)=x$, $Y(:,j,k)=y$, $Z(j,k,:)=z$

Example: CT data

- CT scans
  - "Computerized Tomography" a.k.a. CAT scans) produce a series of cross-sectional X-rays
- Several slices can be stacked together to form a volume

Example: CT data

- CT scans of head and thorax from dogs provided by Dr. Ned Dykes at NYSCVM
- Each slice is a separate tiff file
  - loaded each tiff with imread
  - stacked into array head
- Thinned the data set
  - $[X,Y,Z,\text{Heads}]=\text{reducevolume}(X,Y,Z,\text{Head},[4,4,1])$
- Cropped data
  - $\text{Head\_reduce4\_4\_1\_crop.mat}$
**Visualizing V**

- Simplest way is to look at a particular row/column/layer of V
  - `pcolor(x,y,V(:,,:,k))`—layer k
  - `pcolor(x,z,squeeze(V(:,k,:)))`—column k
  - `pcolor(y,z,squeeze(V(k,:,:)))`—row k
- `squeeze` removes singleton dimensions
  - `v(k,:,:)` is 1-by-n-by-p
  - `squeeze(v(k,:,:))` is n-by-p

**General Slicing**

- `h=slice(X,Y,Z,V,xs,ys,zs)`
  - slices V at multiple planes
  - `slice(X,Y,Z,V,[20 30],[],[])` produces 3 slices:
    - x=20, x=30, z=10
  - What if a slice falls between a row or column?
- `h=slice(X,Y,Z,V,Xsurf,Ysurf,Zsurf)`
  - slices V with a surface defined by `Zs=f(Xs,Ys)`

**Slicing the dog**

- `clearslice(Xs,Ys,Zs,Heads,xs,ys,zs,thresh)`
  - same as `slice(Xs ...)` but values of Heads below the threshold are set to clear
Isosurfaces

- Before perspective plots and color mapping, people plotted $z = f(x,y)$ with contours:
  - curves of constant $z$
- Isosurfaces are analogous methods in 3D
  - find $X,Y,Z$ s.t. $f(X,Y,Z) = v$

Isosurfaces

- $fv = \text{isosurface}(X,Y,Z,V,v)$;
- $fv$ is a struct describing a patch (or surface) object on a triangular mesh
- $fv$.vertices(j,:)=position of jth vertex $[x, y, z]$
- $fv$.faces(j,:)=[1-by-3]--index to 3 vertices forming triangle (like tri)
- $h = \text{patch}(fv)$ will display the surface
  - set(h,'edgecolor','none','facecolor',Colorspec,'facelighting','phong')

Isosurfacing the Dog

- $Fv = \text{isosurface}(Xs,Ys,Zs,Heads,25)$;
- visiso($fv$)--makes a pretty isosurface, adds a light source
Animations

- Animations are extremely easy:
  1. Make an image
  2. Change it
  3. Repeat

Animations in Matlab

- You can do this with a for-loop
  - For j=1:n
    - Make image n
  - End
- Problem: Matlab does this too fast
  - Solution: insert pause command
    - pause; %waits until user hits a key
    - pause(t); %pauses for t seconds

Creating AVI files

- Problems with previous scheme
  - Not portable (only in Matlab)
  - Not efficient: must render each image every time
- Solution: save to a standard movie format
  - AVI is a simple video format which is easy to create with Matlab
Creating AVI files

- Procedure is similar to before:
  - First, open a file:
    - `mov = avifile(name);` %opens file called name
  - Set any options
    - `mov.Quality=100;` %quality of images
    - `mov.Compression='None';` %compression
    - `mov.Fps=fps;` %frames per second
  - Create an image as before
  - Then, capture it:
    - `F = getframe(gca);` %capture the frame
    - `mov = addframe(mov,F);` %add it to the movie
  - Repeat
  - Close the movie
    - `mov = close(mov);`

avislice.m

- `avislice.m` uses `clearslice` to slice along a dimension of data
  - `avislice(Xs,Ys,Zs,Heads,3,30,[20 200],2,'dogslicemovie.avi');"