Lecture 21
Working with Diagnostic Medical Images (in MATLAB)
Lecture 21 #goals

Medical Images
- Imaging for Medical Diagnostic
- DICOM Data Format
- Commercial Software
- MATLAB DICOM tools (Image Processing Toolbox)

MATLAB Graphics Properties
- Property Browser
- read/write properties

3D Data Visualization
- Plotting Slices
- Plotting Isosurfaces
Imaging for Medical Diagnostics

Many types of diagnostic medical imaging machines are used in practice. The most common examples include:

- X-ray
- computed tomography (CT)
- magnetic resonance imaging (MRI)
- positron emission tomography (PET)
- ultrasound
Imaging for Medical Diagnostics

CT scans generate ‘volumetric’ information by applying numerical computations (‘backpropagation’) on a series of x-ray images taken at different angles and locations.

http://xrayphysics.com/ctsim.html
Imaging for Medical Diagnostics

Captured “images” can be 2D, 3D, 4D, ...

All medical images have key characteristics:
- subject orientation/coordinates
- image resolution/accuracy
- tissue density to image intensity mapping
- total contrast/dynamic range
- patient information (sensitive metadata)
Digital Imaging and Communications in Medicine (DICOM) Standard

Joint project by the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA)

• v1.0 published in 1985
  • X-Rays
  • CT scans
• v3.0 is the current version of the standard & supports numerous medical imaging data types
  • MRI scans
  • Dentistry
  • Surgical planning
  • 3D ultrasound
  • 3D printed instrument files
  • and many more...

https://www.dicomstandard.org/current/
DICOM Standard Images

Pixel intensity values are different for each imaging type

- X-ray (intensity = gamma ray absorption level)
  - Bones => light
  - Soft tissues => dark
- MRI (intensity = frequency response of an RF pulse)

Image contrast, or the intensity difference between tissue types, is needed to show features of interest

Each file contains a 2D “slice” of imaging data

- 8-bit (0, 255) or 16-bit (0, 65,535) levels
- RGB (MxNx3) or Grayscale (MxNx1)
Commercial DICOM Software

Many commercial software packages exist for viewing and editing DICOM standard images

- These packages are primarily developed for the medical community (radiologists, general practitioners, patients, etc.)

Examples:

- OsiriX for OS X
  - free “Lite” version
- MicroDicom for Windows
  - free viewer
DICOM Images in MATLAB

MATLAB supports DICOM import/export since 2006
- Part of the Image Processing Toolbox (optional installation package)
- Reads in DICOM standard files
- Writes 3 types of DICOM files (Secondary capture, MRI, CT)

DICOM ultrasound “snapshots” plotted in MATLAB
Source: https://www.mathworks.com/help/images/ref/dicomread.html
DICOM Images in MATLAB

`dicomdisp(filename)` - Displays DICOM file structure on command line

`dicominfo(filename)` - Reads metadata from DICOM file into struct

`X = dicomread(filename)` - Reads a DICOM image

`[X, colormap, alpha, overlaps] = dicomread(filename)`  
 Also reads the colormap, alpha channel, and any overlays, if they exist

`dicomwrite(X, filename)` - Write images as DICOM files

*Note: dicomreadVolume exists, but don’t use it for homework.*
DICOM Images in MATLAB

Once the DICOM image is read into a variable, we can treat it as any normal image

```matlab
>> x = dicomread('IM-0002-0003.dcm');
>> imagesc(x)

>> whos x
Name      Size            Bytes  Class      Attributes
x         512x512         524288  int16      
```
MATLAB Graphics Properties

```matlab
[X,Y] = meshgrid(-8:.5:8);
R = sqrt(X.^2 + Y.^2) + eps;
Z = sin(R)./R;
figure
mesh(X,Y,Z)
```
MATLAB Graphics Objects

Every graphics object (figure, axis, surface, line, light, etc.) has a set of properties.

These properties can be viewed or modified using the Property Inspector.
MATLAB Graphics Objects

Alternatively, we can access properties as object attributes

```matlab
>> fh = figure;
>> fh.Color = [1 1 1] % set background to white

>> x = plot(randn(1000,1));
>> x.LineStyle = ':';
>> x.LineWidth = 2;

In older releases of MATLAB we can use ‘get’ and ‘set’ commands.
>> x = get(fh, 'XScale');
>> set(fh, 'XScale', 'log')
```
MATLAB Graphics Objects

Without the object handle, you will need to use ‘gca’, ‘gcf’, or ‘gco’ (“Get current (axis|figure|object”)). These commands retrieve the handle of the most recently accessed axis, figure, or object.

```matlab
>> x = get(gco, 'XScale');
>> set(gcf, 'Position', [1 1 640 480])
```
3D Data Visualization
x = flow(50);

%% plotting X-Y axis
fh = figure;
for n = 1:size(x,3)
    xslice = x(:,:,n);
    contourf(xslice);
    xlabel('x')
    ylabel('y')
    title(sprintf('Slice #%d',n))
    pause(0.1)
end
3D Volumetric Data in MATLAB

```matlab
x = flow(50);

%% plotting X-Z axis
fh = figure;
for n = 1:size(x,2)
    xslice = squeeze(x(:,n,:));
    contourf(xslice);

    xlabel('x')
    ylabel('z')
    title(sprintf('Slice #%d',n))

    pause(0.1)
end
```

`squeeze` is needed to reduce dimensionality: Mx1xN => MxN
“slicing” into 3D Matrices

Selecting sub-volumes can be done with standard indexing

```matlab
data = flow; % load ‘flow’ dataset
xIdx = 20:40;
yIdx = 5:2:end;
zIdx = 10:35;

subdata = data(xIdx,yIdx,zIdx);
```
3D Volumetric Data in MATLAB

```matlab
>> load mri % load built-in MRI data set
>> whos
Name      Size         Bytes  Class     Attributes
D          128x128x1x27 442368  uint8
map        89x3         2136   double
siz        1x3          24     double

>> D = squeeze(D); % remove 'singleton' dimension
>> D(:,1:60,:) = []; % delete half of data about y-axis
>> whos
Name      Size         Bytes  Class     Attributes
D          128x68x1x27 235008  uint8
map        89x3         2136   double
siz        1x3          24     double
```
3D Volumetric Data in MATLAB

%% plot volumetric surfaces

% create 'patch' objects for each surface
thresh = 5; % pick a threshold level
p1 = patch(isosurface(D, thresh), 'FaceColor', 'red', 'EdgeColor', 'none');
p2 = patch(isocaps(D, thresh), 'FaceColor', 'interp', 'EdgeColor', 'none');
3D Volumetric Data in MATLAB

%% manipulate view
view(3)
axis tight
daspect([1,1,.4])

default view

manipulated view
3D Volumetric Data in MATLAB

%% modify color and lighting effects
colormap(gray(100))
camlight left
camlight
lighting gouraud
3D Volumetric Data in MATLAB

%% smooth out isosurface by computing normals from data
isonormals(D,p1)

default isonormals (‘Triangular’)  modified isonormals (computed)

isonormals generally creates a smoother surface finish
3D Volumetric Data in MATLAB

FV = \texttt{isosurface}(X,Y,Z,V,\text{ISOVALUE}) \text{ computes isosurface geometry} \text{ for data V at isosurface value ISOVALUE.} 

FVC = \texttt{isocaps}(X,Y,Z,V,ISOVALUE) \text{ computes isosurface end cap geometry} \text{ for data V at isosurface value ISOVALUE.} 

N = \texttt{isonormals}(X,Y,Z,V,\text{VERTICES}) \text{ computes the normals of isosurface vertices VERTICES by using the gradient of the data V.}
3D Volumetric Data in MATLAB

Generate the 3D coordinate system (or assume default unit spacing)

```matlab
>> [XX,YY,ZZ] = meshgrid(...)
```

MATLAB can also handle non-uniform grids, but requires user to create 3D grid manually
Transformations of 3D Matrices

Affine transformations (translation and rotation of data) can be performed by modifying the x, y, z coordinates

Translation along x-axis

\[ X = X + \text{delta}X; \]

Translation along y-axis

\[ Y = Y + \text{delta}Y; \]

Translation along z-axis

\[ Z = Z + \text{delta}Z; \]
Transformations of 3D Matrices

Rotation about X-axis:

\[ X = X; \]
\[ Y = Y \cdot \cos(\theta) - Z \cdot \sin(\theta); \]
\[ Z = Y \cdot \sin(\theta) + Z \cdot \cos(\theta); \]

Rotation about Y-axis:

\[ X = X \cdot \cos(\theta) + Z \cdot \sin(\theta); \]
\[ Y = Y; \]
\[ Z = Z \cdot \cos(\theta) - X \cdot \sin(\theta); \]

Rotation about Z-axis:

\[ X = X \cdot \cos(\theta) - Y \cdot \sin(\theta); \]
\[ Y = X \cdot \sin(\theta) + Y \cdot \cos(\theta); \]
\[ Z = Z; \]
3D Volumetric Data in MATLAB

Building a 3D volumetric image from 2D medical images requires stacking each image. The (x,y,z) coordinates will need to be tracked carefully.

1) pre-initialize the 3D matrix (optional, for efficiency)
   \[ V = \text{zeros}(M,N,\text{numel}(\text{files})); \]
2) import each 2D medical image
   \[ x = \text{dicomread}(\text{filename}); \]
3) save each image to a slice in the 3D matrix
   \[ V(:,,:,n) = x; \]

\[ V(x,y,z) \Rightarrow \text{single pixel (“voxel” in 3D) at index x,y,z} \]

Note: verify the size of each image to make sure it is M x N!
#puppiesrule