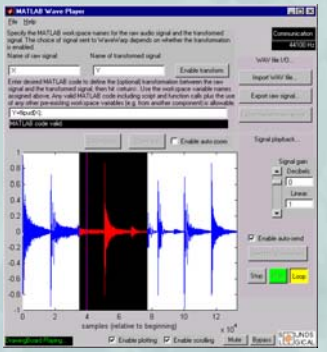
Technion – Electrical Engineering Department

May 2004

Signal and Image Processing Laboratory

GUI with Matlab



Written By:
Yair Moshe

GUI with Matlab - Outline

1. Basic Graphics
2. Animation
3. Handle Graphics Objects
4. Creating GUI using GUIDE

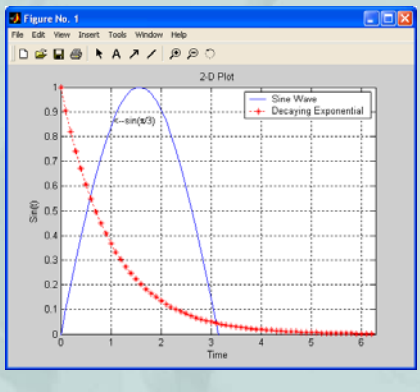
1. Basic Graphics

- 2-D Plotting
- The Figure Window
- Data Statistics & Curve Fitting
- Subplots & Scales for Axes
- Specialized Plotting Routines
- 3-D Plotting
- Images

3

2-D Plotting

```
x=0:.1:2*pi;
y=sin(x);
plot(x,y)
grid on
hold on
plot(x, exp(-x), 'r:*')
axis([0 2*pi 0 1])
title('2-D Plot')
xlabel('Time')
ylabel('Sin(t) ')
text(pi/3, sin(pi/3), '<--sin(\pi/3) ')
legend('Sine Wave',
       'Decaying Exponential')
```



4

Line Characteristics

Specifier	Line Color	Specifier	Marker Style
b	blue	.	point
g	green	o	circle
r	red	x	x-mark
c	cyan	+	plus
m	magenta	*	star
y	yellow	s	square
k	black	d	diamond
		v	triangle down
Specifier	**Line Style**	^	triangle up
-	solid	<	triangle left
:	dotted	>	triangle right
-.	dashdot	p	pentagram
--	dashed	h	hexagram

5

The Figure Window

The figure window contains useful actions in its menu and toolbars:

- Zooming in and out
- Rotating 3-D axes (and other camera actions)
- Copying & pasting
- Plot Edit Mode
- Property Editor
- Saving & Exporting
  - Figures can be saved in a binary .fig file format
  - Figure can be exported to many standard graphics file formats etc., GIF, PNG, JPEG, TIFF, BMP, PCX, EPS.
- Printing

6

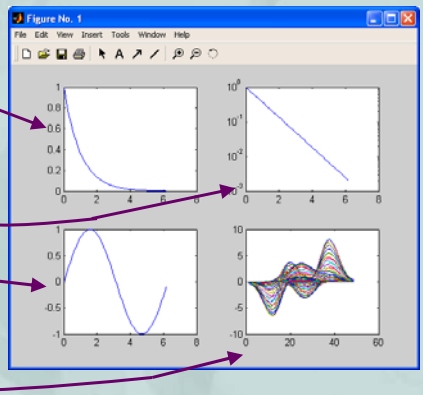
Data Statistics & Curve Fitting

- The Data Statistics Tool:
  - Calculates basic statistics about the central tendency and variability of data plotted in a graph
  - Plots any of the statistics in a graph
- The Basic Fitting Interface:
  - Fits data using a spline interpolant, a shape-preserving interpolant, or a polynomial up to degree 10
  - Plots multiple fits simultaneously for a given data set
  - Examines the numerical results of a fit
  - Annotates the plot with the numerical fit results and the norm of residuals

7

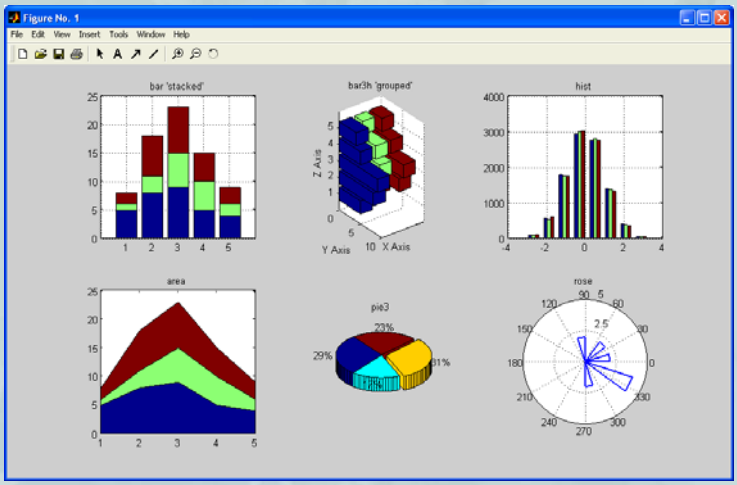
Subplots & Scales for Axes

```
subplot(2,2,1)
x=0:.1:2*pi;
plot(x, exp(-x))

subplot(2,2,2)
semilogy(x, exp(-x))

subplot(2,2,3)
x=0:.1:2*pi;
plot(x, sin(x))

subplot(2,2,4)
plot(peaks)
```



8

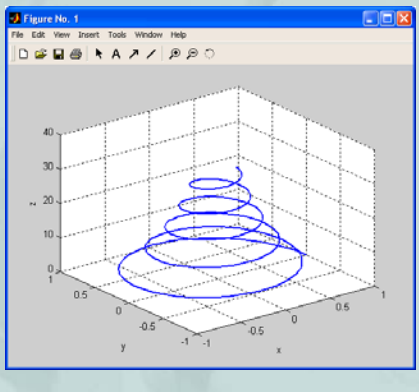
Specialized Plotting Routines



9

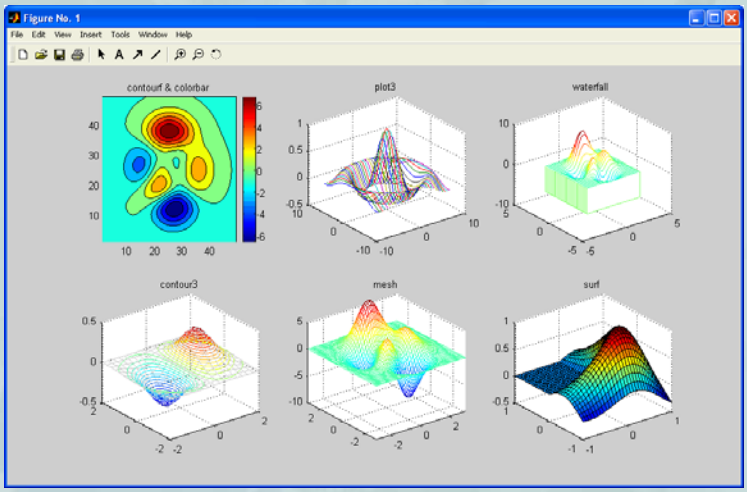
3-D Plotting

```
z = 0:0.1:10*pi;
x = exp(-z/20).*cos(z);
y = exp(-z/20).*sin(z);
plot3(x,y,z,'LineWidth',2)
grid on
xlabel('x')
ylabel('y')
zlabel('z')
```



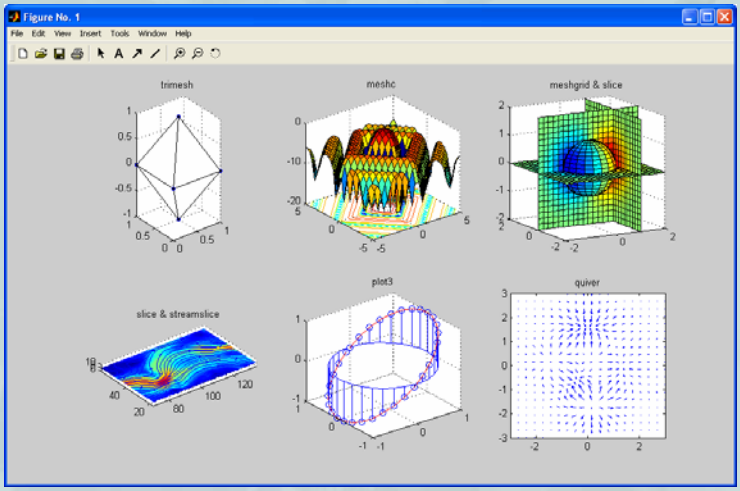
10

3-D Meshes and Surfaces



11

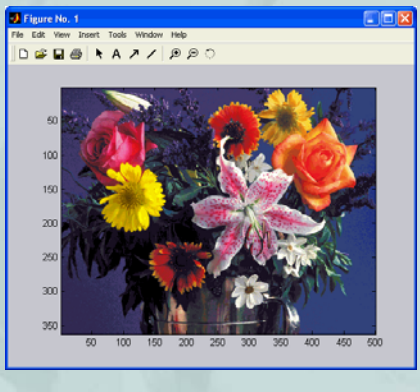
3-D Meshes and Surfaces



12

Images

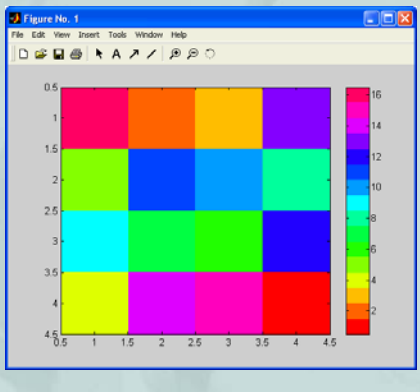
```
[x,map]=imread('flowers.tif');
image(x)
colormap(map)
```



13

Images

```
a = magic(4);
image(a);
map = hsv(16);
colormap(map)
colorbar
```



14

2. Animation

- On the Fly Animation
- Frame by Frame Animation

15

Animation

Matlab provides two ways of generating moving, animated graphics:

1. On the fly - Continually erase and then redraw the objects on the screen, making incremental changes with each redraw.
2. Frame by frame capture and playback - Save a number of different pictures and then play them back as a movie.

16

On the Fly Animation

Example for on the fly animation:

```
t = 0:0.01:10*pi;  x = t.*sin(t);  y = t.*cos(t);
axislimits = [min(x) max(x) min(y) max(y) min(t) max(t)];

line_handle = plot3(x(1), y(1),t(1), 'ko', ...
   'MarkerFaceColor',[.49 1 .63], 'MarkerSize',12);
set(line_handle, 'erasemode','xor');
axis(axislimits);
grid on
for i = 2:length(x)
   set(line_handle, 'xdata',x(i), 'ydata', y(i), 'zdata', t(i));
   drawnow;
end
```

17

Frame by Frame animation

Example for frame by frame movie creation and playing:

```
[x,y] = meshgrid([-10:0.5:10]);
for j = 1:15
   z = bessel(0, (j-1)*0.2 + sqrt(x.^2 +y.^2));
   surf(x,y,z)
   axis([-10 10 -10 10 -.5 1])
   M(j) = getframe;
end

frame_order = [1:15 14:-1:1];
number_repeats = 5;
movie(M, [number_repeats frame_order]);
```

18

3. Handle Graphics

- Handle Graphics Objects
- Graphics Objects Hierarchy
- Obtaining an Object's Handle
- Modifying Object Properties

19

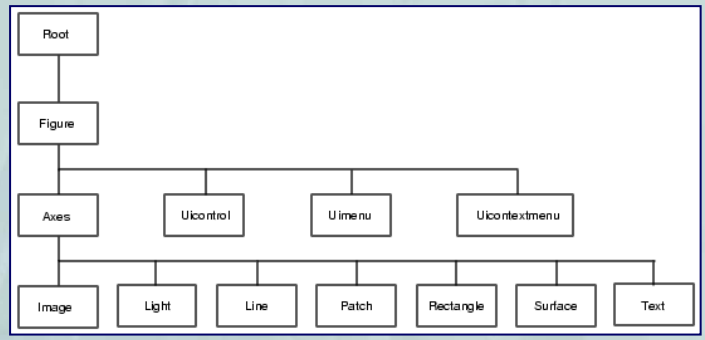
Handle Graphics Objects

- Handle Graphics is an object-oriented structure for creating, manipulating and displaying graphics
- Graphics in Matlab consist of *objects*
- Every graphics objects has:
  - a unique identifier, called *handle*
  - a set of *properties* which define its appearance

20

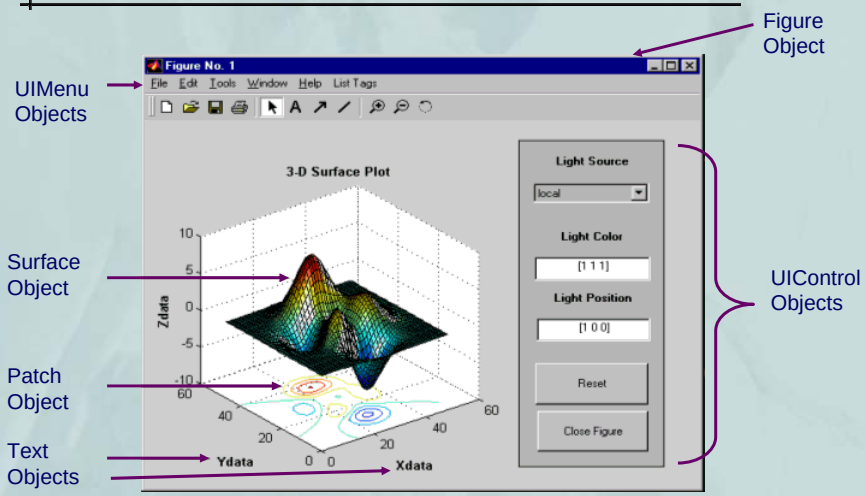
Graphics Objects Hierarchy

Objects are organized into a tree-structure *hierarchy*:



21

Graphics Objects Hierarchy



22

Obtaining an Object's Handle

- Upon creation, for example:
  - h = plot(x_data, y_data, …)
- Using utility functions:
  - 0 - root object handle (the screen)
  - gcf – returns the handle for current figure
  - gca - returns the handle for current axes
  - gco - returns the handle for current object
  - gcbf - returns the handle for callback figure
  - gcbo - returns the handle for callback object

23

Modifying Object Properties

- Return a list of all object properties and their current values:
  - get(handle)
- Return a list of all user-settable object properties and their current values:
  - set(handle)
- Return current value of an object property:
  - get(handle, 'PropertyName')
  - Example: get(gcf, 'Color')
- Return a list of all possible values for an object property:
  - Set(handle,'PropertyName')
  - Example: set(gca, 'XDir')
- Set an object property to a new value:
  - set(handle, 'PropertyName', 'NewValue')
  - Example: set(gca, 'XDir', 'Reverse')
- All the above can also be done (but not at runtime) using the Property Editor

24

Example – Figure Position

```
space = 5;
top_space = 80;
scn_size = get(0,'ScreenSize');

pos1  = [space, 2/3*scn_size(4) + space,...
          scn_size(3)/2 - 2*space, scn_size(4)/3 - (top_space + space)];
pos2 = [pos1(1) + scn_size(3)/2, pos1(2),...
          pos1(3), pos1(4)];

h1 = figure(1);
peaks;
h2 = figure(2);
membrane;

set(h1, 'Position', pos1)
set(h2, 'Position', pos2)
```

25

4. Creating GUI using GUIDE

- What is GUIDE?
- Creating a GUI
- The Layout Editor
- Hands-On GUIDE Example
- Writing Callbacks

26

What is GUIDE?

- GUIDE is Matlab's Graphics User Interface (GUI) Design Environment
- GUIDE stores GUIs in two files, which are generated the first time you save or run the GUI:
  - .fig file - contains a complete description of the GUI figure layout and the components of the GUI
    - Changes to this file are made in the Layout Editor
  - .m file - contains the code that controls the GUI
    - You can program the callbacks in this file using the M-file Editor

27

Creating a GUI

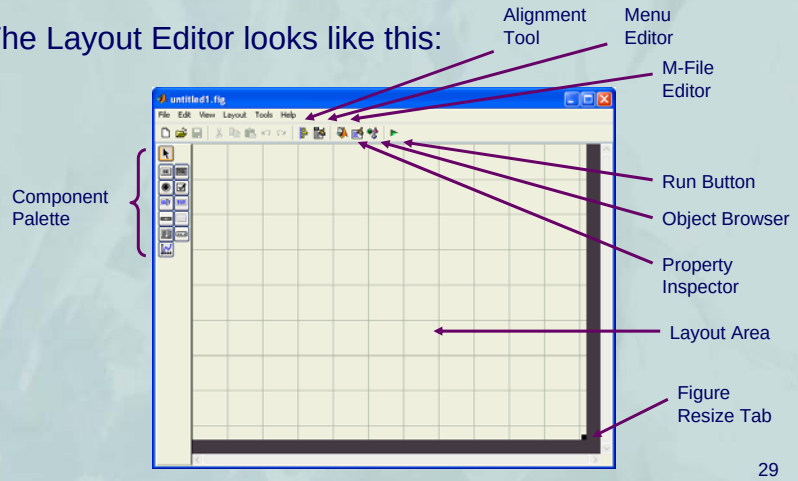
Typical stages of creating a GUI are:

1. Designing the GUI
2. Laying out the GUI
   - Using the Layout Editor
3. Programming the GUI
   - Writing callbacks in the M-file Editor
4. Saving and Running the GUI

28

The Layout Editor

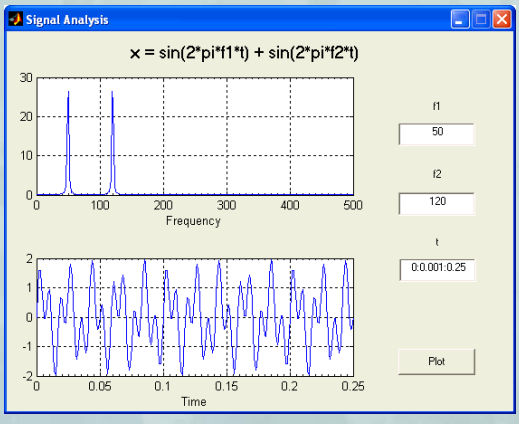
The Layout Editor looks like this:



29

Hands-On GUIDE Example

Example:



30

Writing Callbacks

A *callback* is a sequence of commands that are execute when a graphics object is activated

- Stored in the GUI's M-file
- Is a property of a graphics object (e.g. CreateFnc, ButtonDwnFnc, Callback, DeleteFnc)
- Also called *event handler* in some programming languages

A callback is usually made of the following stages:

1. Getting the handle of the object initiating the action (the object provides event / information / values)
2. Getting the handles of the objects being affected (the object that whose properties are to be changed)
3. Getting necessary information / values
4. Doing some calculations and processing
5. Setting relevant object properties to effect action

Writing Callbacks

Callback from example:

```
% --- Executes on button press in plot_button.
function plot_button_Callback(hObject, eventdata, handles)
% hObject    handle to plot_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get user input from GUI
f1 = str2double(get(handles.f1_input,'String'));
f2 = str2double(get(handles.f2_input,'String'));
t = eval(get(handles.t_input,'String'));

% Calculate data
…

% Create frequency plot
axes(handles.frequency_axes) % Select the proper axes
plot(f,m(1:257))
xlabel('Frequency');
set(handles.frequency_axes,'XMinorTick','on')
grid on
…
```

Further Information

1. The Mathworks, Using Matlab Graphics
2. The Mathworks, Creating Graphics User Interfaces
3. Marchand Patrick, Holland Thomas O., Graphics and GUIs with Matlab, 3ed, 2003, CRC Press

The documents from Mathworks could be found on:
http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml
Most of this info can also be found in Matlab's help

33

34