Scripts, Functions and Graphical User Interfaces

Scripts and ***functions,*** also called *m-files,* are widely used and make life easier while using Matlab. M-files group several commands that are stored in a text file with the extension *.m* and can be later used as any normal Matlab function. For example, a simple function that displays and image with the command *surf*:

```
function outputImage = surfImage (inputImage)
%%this function uses surf to display an image
surf(inputImage);
shading flat;
view(0,-90);
axis tight;
%for example, a different output image
outputImage=[inputImage inputImage/2; inputImage*2 inputImage*3];
```

This function should be saved as a file with the same name, *surfImage.m*. In order to use the function an input image should be provided, the output is not necessary though. Try:

```
>> surfImage(image4);
```

When a set of instructions are to be repeated a number of times, but no input and output parameters are to be specified, these can be saved into an m-file, but without the header, that is, the first line of the previous example. This would be called a *script*. Use *help script / function* to find out more.

Generating a ***Graphical User Interface*** (GUI) in MATLAB is quite simple. GUIs allow the direct manipulation of the used without the need of running a script or function directly. The standard elements of GUIs: *Check boxes, Editable text fields, Frames, List boxes, Pop-up menus, Push buttons, Radio buttons, Sliders, Static text labels, Toggle buttons* . . . are readily available. First of all you have to grab a figure to use as GUI:

```
fig=gcf;            %% Get current figure, opens one if not existing
clf;                %% clears the figure
```

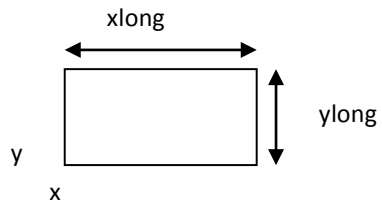
The ***handle*** of the figure is stored in "fig", this handle will be used for any element within the GUI (to see the properties of an image type get(fig)). uicontrol creates uicontrol graphics objects (user interface controls). You implement graphical user interfaces using uicontrols. When selected, most uicontrol objects perform a predefined action.

1 To insert buttons try:

```
pbConvolute = uicontrol(fig,'Style','push','position',[5 50 110 35],...
  'String','Start','CallBack','convCallback');
pbEnd= uicontrol(fig,'Style','push','position',[5 10 110 35],...
  'String','End','CallBack','close');
```

The previous commands create two buttons, one for performing the Convolution "Convolute" and one to close the GUI "End". Some of the properties of these controls are:

```
Style         push, text, slider, popup
Position      [x   y   xlong   ylong]
String        text that appears in the interface
Callback       command executed
```



2 For Sliders and text:

```
sliSize= uicontrol(fig,'Style','slider','position',[125 10 130 35],...
  'Min',1,'Max',10,'Value',1,'CallBack','');
texSize=uicontrol(fig,'Style','text','position',[125 50 100 35],...
  'string','Size of Kernel');
valSize=uicontrol(fig,'Style','text','position',[225 50 30 35],...
  'string',num2str(get(sliSize,'Value')));
```

3 For pop-up menus:

```
texKernel=uicontrol(fig,'Style','text','position',[5 25 95 40],...
  'string','Type of Kernel');
popKernel = uicontrol(fig,'Style','popup',...
  'String','Average|Gaussian|Vertical|Horizontal|45|90|non-linear',...
  'position',[5 10 95 35],...
  'CallBack','');
```

4 To retrieve the values of the controls you can use

```
sizeK=round(get(sliSize,'value'));
typeK=get(popKernel,'value');
```

5 Callbacks perform operations for every event of the control (push the button, move the slider ...). They can be specified in many ways, from just a command in the same line "close", call a new function or programme, or execute a series of commands previously defined, for example:

```
convCallback= ['sizeK=round(get(sliSize,''value''));',...
               'kindK=get(popKernel,''value''); if kindK==1
outputImage=(image1); else outputImage=(image2); end;',...
               'surfImage ( outputImage);'];

pbConvolve = uicontrol(fig,'Style','push','position',[5 50 110 35],...
  'String','Start','CallBack',convCallback);
```

The callback reads parameters and uses them to determine the image to be displayed.

Tasks

1 Study any of the many GUI tutorials for Matlab:

http://www.mathworks.com/access/helpdesk/help/techdoc/creating_guis/bqz6qcd.html
http://www.matlabgui.com/
http://www.intelligent-systems.info/classes/ee509/gui.htm
http://www.mathworks.com/matlabcentral/fileexchange/5439-matlab-gui-tips

Matlab has a very powerful tool to create GUIs called GUIDE, but before using it, familiarise yourself with the programmatic creation of GUIs.

2 Create a GUI in which you incorporate some of the previous knowledge/skills you have learned. For instance you can display an image and select with buttons which colour channel you want to display, you may wish to crop the image or display its Fourier transform in different ways, absolute value, phase, etc.