Table of Contents

```
%Marat Purnyn and Leonard Knittle
%Section 61
%Group 9

%==========================================================================
%=========== Accelerometer Magnitude Skeleton Code ======================
%==========================================================================
%This skeleton script does the following:
% 1. Specifies the COM port that the Arduino board is connected to
% 2. Initializes the Serial Port - setupSerial() (not to be altered)
% 3. Runs a calibration routine if needed - calibrate() (not to be altered)
% 4. Opens a new figure and customizes it by adding start/stop and close
%    serial buttons
%    - A different stop call
% 5. Initializes a rolling plot
% 6. Runs a loop that continually reads the accelerometer values
%    readAcc() - (not to be altered)
%    The accelerometer data is placed in the variables [gx gy gz].
%    Updates the data on the rolling plot
```

1. Specifies the COM port that the Arduino board is connected to

```
comPort = 'COM4';%This can be found out using the device manager (Windows)
                  %On a mac type ls /dev/tty* in Terminal and
                  %  identify the device that is listed as usbmodem
                  %  Example for a MAC comPort = '/dev/tty.usbmodemfa131';

%comPort = '/dev/tty.usbmodemfd121';
```

2. Initialize the Serial Port - setupSerial() (not to be altered)

```
%connect MATLAB to the accelerometer
if (~exist('serialFlag','var'))
 [accelerometer.s,serialFlag] = setupSerial(comPort);
end
```

3. Run a calibration routine if needed - calibrate() (not to be altered)

```
%if the accelerometer is not calibrated, calibrate now
if(~exist('calCo', 'var'))
    calCo = calibrate(accelerometer.s);
end
```

4. Open a new figure - add start/stop and close serial buttons

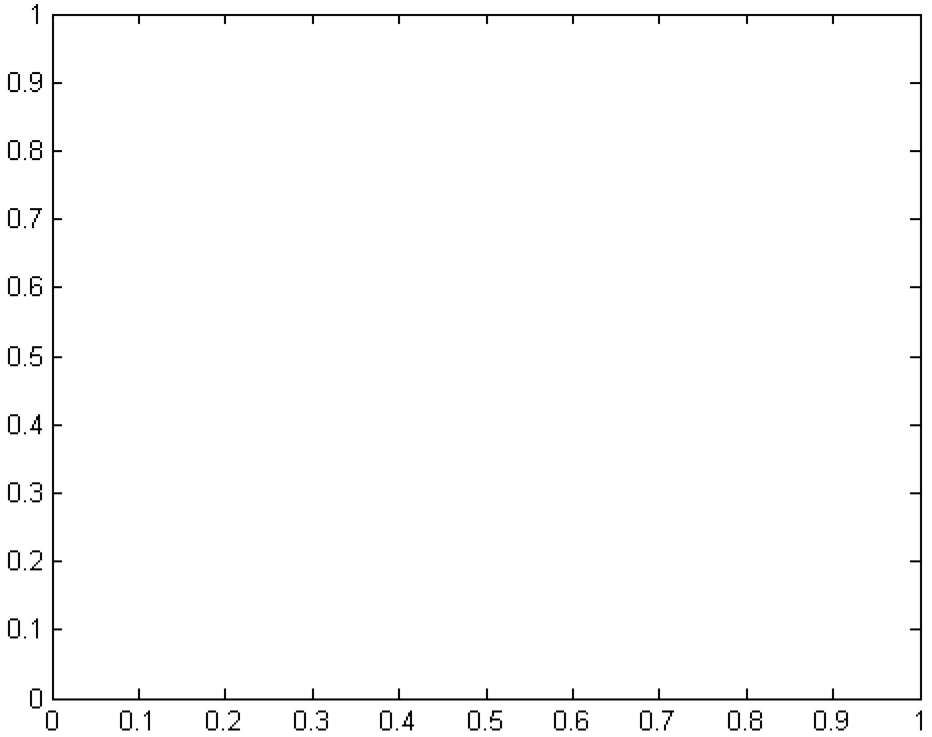
```
%initalize the figure that we will plot in if it does not exist
if(~exist('h', 'var') || ~ishandle(h))
    h = figure(1);
    ax = axes('box','on');
end

%add a start/stop and close serial button inside the figure
%Keep in mind the 'stop_call_wk3' function that this button calls everytime
%it is pressed

if(~exist('button','var'))
    button = uicontrol('Style','pushbutton','String','Stop',...
                    'pos',[0 0 50 25],'parent',h,...
                    'Callback','stop_call_magnitude','UserData',1);
end

%Keep in mind the 'close_call' function that this button calls everytime
%it is pressed

if(~exist('button2','var'))
    button2 = uicontrol('Style','pushbutton','String','Close Serial Port',...
                    'pos',[250 0 150 25],'parent',h,...
                    'Callback','closeSerial','UserData',1);
end
```



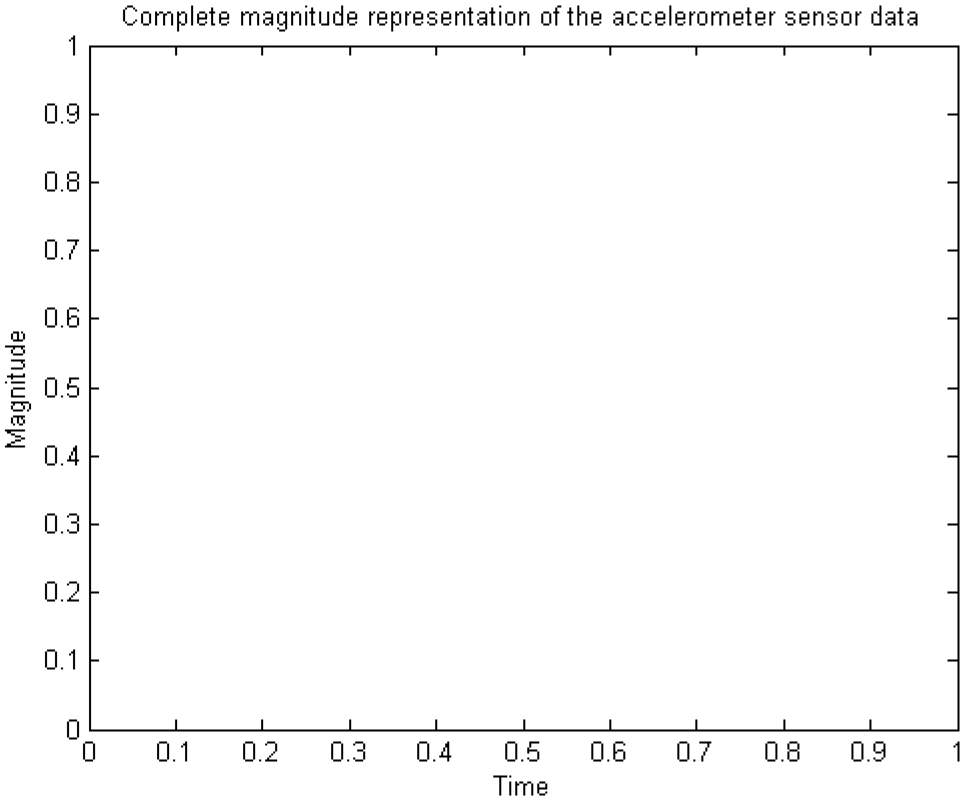
5. Initializing the Rolling Plot

Again, the rolling plot works like the last code. We set it up in the same manner.

```
buf_len = 200;
% create variables for all the three axis and the resultant

gxdata = zeros(buf_len,1);
gydata = zeros(buf_len,1);
gzdata = zeros(buf_len,1);
index = 1:buf_len;

% Display x and y label ad title.
xlabel('Time');
ylabel('Magnitude');
title('Complete magnitude representation of the accelerometer sensor data');
```



6. Data Collection and Plotting.

While the figure window is open.

```
while(get(button, 'UserData'))

    % Get the new values from the accelerometer
    [gx gy gz] = readAcc(accelerometer,calCo);

    % Calculate the magnitude of the accelerometer axis readings
    %Students calculate the magnitude here

    % Append the new reading to the end of the rolling plot data. Drop the
    % first value

    gxdata = [gxdata(2:end) ; gx];
    gydata = [gydata(2:end) ; gy];
    gzdata = [gzdata(2:end) ; gz];
    gmdata = gxdata+gydata+gzdata;
    % Update the rolling plot

    % subplot for resultant maginitude
    subplot(2,1,1);
    %Students plot the magnitude here
    %--->
    plot(index,gmdata,'black');
    axis([1 buf_len -3.5 3.5]);
    xlabel('time');
    ylabel('Magnitude of the resultant acceleration');

    % subplot for x y z magnitude
    subplot(2,1,2);
    plot(index,gxdata,'r', index,gydata,'g', index,gzdata,'b');
    axis([1 buf_len -3.5 3.5]);
    xlabel('time');
    ylabel('Magnitude of individual axes acceleration');

    drawnow;

end
```

```
Error using handle.handle/get
Invalid or deleted object.

Error in wk3_magnitude (line 93)
while(get(button, 'UserData'))
```

Published with MATLAB® R2013a