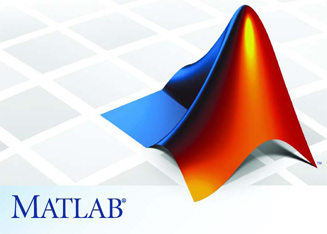
**BUSAT DATA ANALYSIS GUI**

**EK 127 FALL 2012 PROJECT 3**



CARLTON DUFFETT  
SAM CHENEY  
LAB SECTION C6

PROFESSOR ATTAWAY

WE CERTIFY THAT ALL WORK PRESENTED HERE IS OUR OWN AND THAT NO OUTSIDE ASSISTANCE WAS RECEIVED IN THE DEVELOPMENT OF THIS CODE.

CARLTON DUFFETT \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ DATE \_\_\_\_\_\_\_\_\_

SAM CHENEY \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ DATE \_\_\_\_\_\_\_\_\_

HANDED IN TO PROJECT HAND-IN FOLDER, SECTION C6. FOLDER TITLED: “CARLTON DUFFETT SAM CHENEY PROJECT 3”

*LarrysFileRead* runtime: **16.9733** seconds

*loadData run*time: **0.2130** seconds

ZERO\_GUI.m:

function ZERO\_GUI()

%% General User Interface for BUSAT data analysis

% Code developed by:

% Carlton Duffett

% Sam Cheney

% 11 December 2012

clear

% create the figure window

f = figure(...

'Visible','off',...

'Units','normalized',...

'MenuBar','none',...

'NumberTitle','off',...

'Position', [0.1 0.1 0.8 0.8],...

'Name','Accel/Gyro Project 3 GUI');

movegui('center')

% import appropriate file data from directory

data = dir('\*.txt');

% file names:

[fileAname fileBname fileCname] = data.name;

% file sizes:

[fileAsize fileBsize fileCsize] = data.bytes;

% file dates:

[fileAdate fileBdate fileCdate] = data.date;

% File Information:

Ainfo = {'File General Info:' ...

'' ...

sprintf('File name: %s',fileAname) ...

sprintf('File size: %d bytes',fileAsize) ...

sprintf('Date created: %s',fileAdate(1:11))};

Binfo = {'File General Info:' ...

'' ...

sprintf('File name: %s',fileBname) ...

sprintf('File size: %d bytes',fileBsize) ...

sprintf('Date created: %s',fileBdate(1:11))};

Cinfo = {'File General Info:' ...

'' ...

sprintf('File name: %s',fileCname) ...

sprintf('File size: %d bytes',fileCsize) ...

sprintf('Date created: %s',fileCdate(1:11))};

% read file using loadData

[C\_x C\_y C\_z G\_x G\_y G\_z] = loadData(fileAname); % default is fileA

%% Prepare the BUSAT logo watermark

% load image

img = imread('BUlogo.png');

R = img(:,:,1);%separate RGB

G = img(:,:,2);

B = img(:,:,3);

% Replace the yellow background with white

repvecR = find(R == mode(mode(double(R))));% mode of RGB

repvecG = find(G == mode(mode(double(G))));

repvecB = find(B == mode(mode(double(B))));

reppixels = intersect(intersect(repvecR,repvecG),repvecB);% yellow pixels

% replace yellow pixels w/ white

R(reppixels) = 255;

G(reppixels) = 255;

B(reppixels) = 255;

% combine RGB

newimg(:,:,1) = R;

newimg(:,:,2) = G;

newimg(:,:,3) = B;

% adjusted image

newimg = uint8(newimg);

%% Plots

% Accelerometer axis

C\_axis = axes(...

'Parent',f,...

'Visible','off',...

'Units','normalized',...

'Position',[0.2 0.6 0.7 0.3]);

% plot default accelerometer data

hold(C\_axis,'on')

C\_xplot = plot(1:length(C\_x),C\_x,'r','Visible','off');

C\_yplot = plot(1:length(C\_y),C\_y,'g','Visible','off');

C\_zplot = plot(1:length(C\_z),C\_z,'b','Visible','off');

hold(C\_axis,'off')

title('Accelerometer')

xlabel('Sample Number')

ylabel('Acceleration in G')

legend({'X','Y','Z'})

% Gyroscope axis

G\_axis = axes(...

'Parent',f,...

'Visible','off',...

'Units','normalized',...

'Position',[0.2 0.1 0.7 0.3]);

% plot default gyroscope data

hold(G\_axis,'on')

G\_xplot = plot(1:length(G\_x),G\_x,'r','Visible','off');

G\_yplot = plot(1:length(G\_y),G\_y,'g','Visible','off');

G\_zplot = plot(1:length(G\_z),G\_z,'b','Visible','off');

hold(G\_axis,'off')

title('Gyroscope')

xlabel('Sample Number')

ylabel('Acceleration in rads/s^2')

legend({'Pitch','Roll','Yaw'})

%% Radio Buttons

% Accelerometer buttons

X\_plot\_button = uicontrol(...

'Visible','off',...

'Style','radiobutton',...

'String','X data',...

'Value',1,...

'Units','normalized',...

'Position',[0.93 0.81 0.05 0.025],...

'BackgroundColor',[0.8 0.8 0.8],...

'Callback',{@plotbuttonfn,1});

Y\_plot\_button = uicontrol(...

'Visible','off',...

'Style','radiobutton',...

'String','Y data',...

'Value',1,...

'Units','normalized',...

'Position',[0.93 0.77 0.05 0.025],...

'BackgroundColor',[0.8 0.8 0.8],...

'Callback',{@plotbuttonfn,2});

Z\_plot\_button = uicontrol(...

'Visible','off',...

'Style','radiobutton',...

'String','Z data',...

'Value',1,...

'Units','normalized',...

'Position',[0.93 0.73 0.05 0.025],...

'BackgroundColor',[0.8 0.8 0.8],...

'Callback',{@plotbuttonfn,3});

% Gyroscope buttons

Pitch\_plot\_button = uicontrol(...

'Visible','off',...

'Style','radiobutton',...

'String','Pitch data',...

'Value',1,...

'Units','normalized',...

'Position',[0.93 0.31 0.05 0.025],...

'BackgroundColor',[0.8 0.8 0.8],...

'Callback',{@plotbuttonfn,4});

Roll\_plot\_button = uicontrol(...

'Visible','off',...

'Style','radiobutton',...

'String','Roll data',...

'Value',1,...

'Units','normalized',...

'Position',[0.93 0.27 0.05 0.025],...

'BackgroundColor',[0.8 0.8 0.8],...

'Callback',{@plotbuttonfn,5});

Yaw\_plot\_button = uicontrol(...

'Visible','off',...

'Style','radiobutton',...

'String','Yaw data',...

'Value',1,...

'Units','normalized',...

'Position',[0.93 0.23 0.05 0.025],...

'BackgroundColor',[0.8 0.8 0.8],...

'Callback',{@plotbuttonfn,6});

%make all visible

set([...

X\_plot\_button ...

Y\_plot\_button ...

Z\_plot\_button ...

Pitch\_plot\_button ...

Roll\_plot\_button ...

Yaw\_plot\_button ...

C\_axis ...

C\_xplot ...

C\_yplot ...

C\_zplot ...

G\_axis ...

G\_xplot ...

G\_yplot ...

G\_zplot ...

f], ...

'Visible','on')

%% Radio Button Callback Function

function plotbuttonfn(~,~,button) % '~' does not store source/eventdata

switch button

case 1 % X

if get(X\_plot\_button,'Value') == 0

set(C\_xplot,'Visible','off')

else

set(C\_xplot,'Visible','on')

end

case 2 % Y

if get(Y\_plot\_button,'Value') == 0

set(C\_yplot,'Visible','off')

else

set(C\_yplot,'Visible','on')

end

case 3 % Z

if get(Z\_plot\_button,'Value') == 0

set(C\_zplot,'Visible','off')

else

set(C\_zplot,'Visible','on')

end

case 4 % Pitch

if get(Pitch\_plot\_button,'Value') == 0

set(G\_xplot,'Visible','off')

else

set(G\_xplot,'Visible','on')

end

case 5 % Roll

if get(Roll\_plot\_button,'Value') == 0

set(G\_yplot,'Visible','off')

else

set(G\_yplot,'Visible','on')

end

case 6 % Yaw

if get(Yaw\_plot\_button,'Value') == 0

set(G\_zplot,'Visible','off')

else

set(G\_zplot,'Visible','on')

end

end

end

%% Info box

Info\_text = uicontrol(...

'Visible','on',...

'Style','text',...

'Units','normalized',...

'Position',[0.01 0.9 0.14 0.08],...

'BackgroundColor',[1 1 1],...

'String',Ainfo,... % default is file A

'HorizontalAlignment','left');

%% File drop down menu

% drop down menu

DD\_menu = uicontrol(...

'Style','popupmenu',...

'Units','normalized',...

'Position',[0.0125 0.8 0.13 0.025],...

'String',{fileAname fileBname fileCname});

% load pushbutton

Load\_psb = uicontrol(...

'Style','pushbutton',...

'Units','normalized',...

'Position',[0.035 0.75 0.08 0.03],...

'String','Load Data',...

'Callback',@loadpsbfn);

set([DD\_menu Load\_psb],'Visible','on')

% Load\_psb callback function

function loadpsbfn(~,~)

set(Load\_psb,'String','LOADING...','BackgroundColor','white')

whichfile = get(DD\_menu,'Value');

switch whichfile

case 1 % file A

[C\_x C\_y C\_z G\_x G\_y G\_z] = loadData(fileAname);

set(Info\_text,'String',Ainfo)

case 2 % file B

[C\_x C\_y C\_z G\_x G\_y G\_z] = loadData(fileBname);

set(Info\_text,'String',Binfo)

case 3 % file C

[C\_x C\_y C\_z G\_x G\_y G\_z] = loadData(fileCname);

set(Info\_text,'String',Cinfo)

end

sampleSize = round(get(Filter\_slider,'Value'));

% adjusts vectors using a moving average

C\_xA = mvgAverage(C\_x,sampleSize);

C\_yA = mvgAverage(C\_y,sampleSize);

C\_zA = mvgAverage(C\_z,sampleSize);

G\_xA = mvgAverage(G\_x,sampleSize);

G\_yA = mvgAverage(G\_y,sampleSize);

G\_zA = mvgAverage(G\_z,sampleSize);

% Plot new data

if get(nFilterButtongroup,'SelectedObject') == Noise\_button\_1

% if filtering is currently turned on

% Plot adjusted data

% accelerometer

set(C\_xplot,...

'XData',1:length(C\_xA),...

'YData',C\_xA);

set(C\_yplot,...

'XData',1:length(C\_yA),...

'YData',C\_yA);

set(C\_zplot,...

'XData',1:length(C\_zA),...

'YData',C\_zA);

% gyroscope

set(G\_xplot,...

'XData',1:length(G\_xA),...

'YData',G\_xA);

set(G\_yplot,...

'XData',1:length(G\_yA),...

'YData',G\_yA);

set(G\_zplot,...

'XData',1:length(G\_zA),...

'YData',G\_zA);

else

% if filtering is currently turned off

% Plot original data

% accelerometer

set(C\_xplot,...

'XData',1:length(C\_x),...

'YData',C\_x);

set(C\_yplot,...

'XData',1:length(C\_y),...

'YData',C\_y);

set(C\_zplot,...

'XData',1:length(C\_z),...

'YData',C\_z);

% gyroscope

set(G\_xplot,...

'XData',1:length(G\_x),...

'YData',G\_x);

set(G\_yplot,...

'XData',1:length(G\_y),...

'YData',G\_y);

set(G\_zplot,...

'XData',1:length(G\_z),...

'YData',G\_z);

end

pause(.1)

% reset load pushbutton

set(Load\_psb,...

'String','Load Data',....

'BackgroundColor',[0.831373 0.815686 0.784314])

% reset radio buttons

set([...

X\_plot\_button ...

Y\_plot\_button ...

Z\_plot\_button ...

Pitch\_plot\_button ...

Roll\_plot\_button ...

Yaw\_plot\_button],...

'Value',1)

end

%% Filtering

% filter toggle button

nFilterButtongroup = uibuttongroup(...

'Visible','off',...

'Title','Noise Filter',...

'Units','Normalized',...

'Position',[0.037 0.6 0.075 0.1]);

% noise on/off switch

Noise\_button\_1 = uicontrol(...

'Style','Radiobutton',...

'Parent',nFilterButtongroup,...

'String','ON',...

'FontUnits','Normalized',...

'Units','Normalized',...

'Position',[0.1 0.65 0.8 0.3],...

'HandleVisibility','off');

Noise\_button\_2 = uicontrol(...

'Style','Radiobutton',...

'Parent',nFilterButtongroup,...

'String','OFF',...

'FontUnits','Normalized',...

'Units','Normalized',...

'Position',[0.1 0.2 0.8 0.3],...

'HandleVisibility','off');

% default sampleSize

sampleSize = 3;

set(nFilterButtongroup,'SelectionChangeFcn',@filterfn);

set(nFilterButtongroup,'SelectedObject',Noise\_button\_2);

set(nFilterButtongroup,'Visible','on');

% noise filter callback function

function filterfn(~,~)

if get(nFilterButtongroup,'SelectedObject') == Noise\_button\_1

% get sample size

sampleSize = round(get(Filter\_slider,'Value'));

% adjusts vectors using a moving average

C\_xA = mvgAverage(C\_x,sampleSize);

C\_yA = mvgAverage(C\_y,sampleSize);

C\_zA = mvgAverage(C\_z,sampleSize);

G\_xA = mvgAverage(G\_x,sampleSize);

G\_yA = mvgAverage(G\_y,sampleSize);

G\_zA = mvgAverage(G\_z,sampleSize);

% plot new data

set(C\_xplot,...

'XData',1:length(C\_xA),...

'YData',C\_xA);

set(C\_yplot,...

'XData',1:length(C\_yA),...

'YData',C\_yA);

set(C\_zplot,...

'XData',1:length(C\_zA),...

'YData',C\_zA);

set(G\_xplot,...

'XData',1:length(G\_xA),...

'YData',G\_xA);

set(G\_yplot,...

'XData',1:length(G\_yA),...

'YData',G\_yA);

set(G\_zplot,...

'XData',1:length(G\_zA),...

'YData',G\_zA);

% show sample size and recalculation objects

set([Filter\_slider Min\_text Max\_text Title\_text Recalc\_button],...

'Visible','on')

else

% restore original data

set(C\_xplot,...

'XData',1:length(C\_x),...

'YData',C\_x);

set(C\_yplot,...

'XData',1:length(C\_y),...

'YData',C\_y);

set(C\_zplot,...

'XData',1:length(C\_z),...

'YData',C\_z);

set(G\_xplot,...

'XData',1:length(G\_x),...

'YData',G\_x);

set(G\_yplot,...

'XData',1:length(G\_y),...

'YData',G\_y);

set(G\_zplot,...

'XData',1:length(G\_z),...

'YData',G\_z);

% hide sample size and recalculation objects

set([Filter\_slider Min\_text Max\_text Title\_text Recalc\_button],...

'Visible','off')

end

end

% filter slider

Filter\_slider = uicontrol(...

'Style','slider',...

'Visible','off',...

'Units','normalized',...

'Position',[0.025 0.5 0.1 0.025],...

'Value',3,...

'Min',2,...

'Max',10,...

'SliderStep',[(1/9) (1/3)],...

'Callback',@sliderfn);

% slider labels

Min\_text = uicontrol(...

'Style','text',...

'Visible','off',...

'Units','normalized',...

'Position',[0.01 0.497 0.01 0.025],...

'String','2',...

'BackgroundColor',[0.8 0.8 0.8]);

Max\_text = uicontrol(...

'Style','text',...

'Visible','off',...

'Units','normalized',...

'Position',[0.126 0.497 0.02 0.025],...

'String','10',...

'BackgroundColor',[0.8 0.8 0.8]);

Title\_text = uicontrol(...

'Style','text',...

'Visible','off',...

'Units','normalized',...

'Position',[0.03 0.527 0.085 0.025],...

'String','Sample Size: 3',...

'BackgroundColor',[0.8 0.8 0.8]);

% recalculation button

Recalc\_button = uicontrol(...

'Style','pushbutton',...

'Visible','off',...

'Units','normalized',...

'Position',[0.035 0.42 0.08 0.03],...

'String','Recalculate',...

'Callback',@filterfn);

% slider callback function

function sliderfn(~,~)

sampleSize = round(get(Filter\_slider,'Value'));

label = sprintf('Sample Size: %d',sampleSize);

set(Title\_text,'String',label)

end

%% Screen capture

% screenshot button

Screen\_button = uicontrol(...

'Style','pushbutton',...

'Units','normalized',...

'Position',[0.04 0.04 0.075 0.05],...

'String','Screen Capture',...

'BackgroundColor',[0.0625 1.0000 0.9375],...

'Callback',@screenshotfn);

% screen shot callback function

function screenshotfn(~,~)

% screenshot button visibility

set(Screen\_button,'Visible','off')

% center figure window

movegui('center')

saveas(f,'Sshot.png') % save figure window

close(f) % close figure

% play camera sound

[y,Fs] = wavread('camera.wav');

sound(y,Fs)

% new figure window

figure(...

'Visible','on',...

'Units','normalized',...

'Position', [0.1 0.1 0.8 0.8],...

'Name','Screenshot');

% load saved screenshot

windowimg = imread('Sshot.png');

imshow(windowimg)

% mouse location

[Xclick Yclick] = ginput(1);

% error check watermark positioning

[X Y ~] = size(windowimg);

[r c h] = size(newimg);

% requested position exceeds screenshot dimensions

while round(Xclick - c/2) <= 0 || round(Xclick - c/2) > X ...

|| round(Yclick - r/2) <= 0 || round(Yclick - r/2) > Y

disp('ERROR: watermark position exceeds screenshot dimensions')

% get new position

[Xclick Yclick] = ginput(1);

end

% superimpose image in desired location

for i = 1:r

for j = 1:c

for k = 1:h

windowimg(round(Yclick - r/2) + i,...

round(Xclick - c/2) + j,...

k) = newimg(i,j,k);

end

end

end

% display image with watermark

hold on

image(windowimg,'AlphaData',.2)

end

end

mvgAverage.m:

function [ADJUSTEDvector] = mvgAverage(RAWvector, sampleSize)

% This function takes a RAW vector and returns a moving average

% of a given sample size between 2 and 10

% preallocate

RAWmat = zeros(sampleSize, length(RAWvector)); % s x l matrix of zeros

RAWmat(1,:) = RAWvector; % first row of matrix is RAWvector

for i = 2:sampleSize

% creates matrix of shifted elements

RAWmat(i,:) = circshift(RAWvector, [0 -(1\*i - 1)]);

end

% create adjusted vector

ADJUSTEDvector = sum(RAWmat)/sampleSize;

% truncate adjusted vector

ADJUSTEDvector = ADJUSTEDvector(1:end-(sampleSize-1));

end

loadData.m:

function [C\_x C\_y C\_z G\_x G\_y G\_z] = loadData(filename)

% FileRead reads data from a specified file and returns

% xyz vectors of acceleration & gyroscopic data.

fid = fopen(filename);

if fid == -1

disp('ERROR: File Open Failed')

else

% first two lines of file (extraneous)

fgetl(fid);

fgetl(fid);

% read file using textscan

RAWdata = textscan(fid,'%s %f %f %f %f');

% extract data

C\_x = (RAWdata{3}(2:4:end))';

C\_y = (RAWdata{4}(2:4:end))';

C\_z = (RAWdata{5}(2:4:end))';

G\_x = (RAWdata{3}(3:4:end))';

G\_y = (RAWdata{4}(3:4:end))';

G\_z = (RAWdata{5}(3:4:end))';

end

close = fclose(fid);

if close == -1

disp('ERROR: File Close Failed')

end

efficiencyTest.m:

%efficiency test

tic

[C\_x C\_y C\_z G\_x G\_y G\_z] = LarrysFileRead('A\_DataCollection\_2012-08-25.txt');

toc

tic

[C\_x C\_y C\_z G\_x G\_y G\_z] = loadData('A\_DataCollection\_2012-08-25.txt');

toc