Assignment 1 EE 610 (Image Processing)

Abstract:

This project is to give an individual a basic idea of elementary image editing techniques, and it's applications in the modern world. This basic GUI built on MATLAB allows for many editing techniques and is built on GUIDE, a platform for building apps.

Introduction

The assignment was to make an image editor GUI using matlab/python. I used matlab as it was easier to build the gui and write the associated code to build the application. I built the basic gui first and then approached the problem, function by function.

Background read

Went through the book "Digital Image Processing" and went through basic GUI building techniques on Youtube. After that, went through Mathworks and other websites for debugging whenever needed.

Approach used to build the software application:

Used MATLAB GUIDE to build the GUI for the application. Chose MATLAB as it was easier to build GUI and write suitable code for building the application, as compared to Python. Tried to make the GUI and the code as simple as possible so that understanding and editing the code when required would be easy.

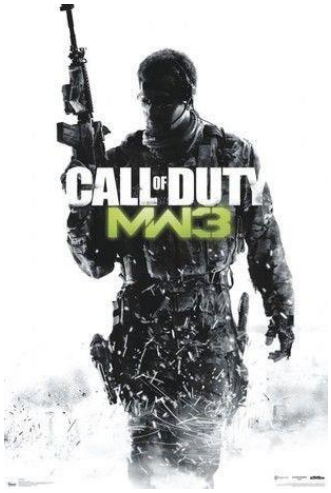
I tried my best to solve problems like making a slider for blurring work, or like trying to make multiple effects work on a single picture, or making undo and redo options possible. Some I managed to accomplish, and some I failed.

Main challenges faced:

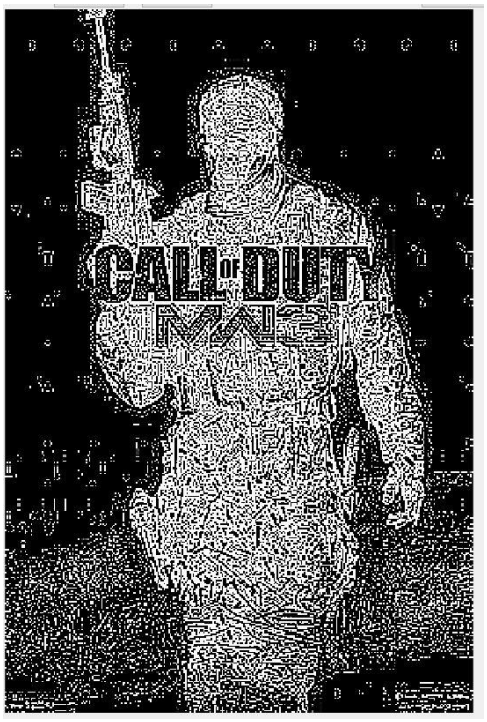
- Understanding the concept of color images and how it is stored in matlab.
- Making all effects work together on the image
- Making a working system to undo and redo images
- Trying to make a slider for blurring (trying to decrease the extent of blurring when slider steps down)
- Switching between color images and grayscale images while editing.
- Working with sliders.

Results on some image(s) that I found interesting

- **Original Image**



After taking it's negative and after multiple sharpening, it becomes (the periodic spots):



- **Its interesting to see the effect of histogram equalization on images that are hazy or have noises that are like translucent screens of color on them:**

INPUT IMAGE

AFTER HISTOGRAM EQUALISATION

References:

https://www.mathworks.com

https://www.youtube.com/
https://stackoverflow.com
https://dsp.stackexchange.com

CODE:

```
function varargout = IPAssignment(varargin)
% IPASSIGNMENT MATLAB code for IPAssignment.fig
%      IPASSIGNMENT, by itself, creates a new IPASSIGNMENT or raises
the existing
%      singleton*.
%
%      H = IPASSIGNMENT returns the handle to a new IPASSIGNMENT or
the handle to
%      the existing singleton*.
%
%      IPASSIGNMENT('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in IPASSIGNMENT.M with the given input
arguments.
%
%      IPASSIGNMENT('Property','Value',...) creates a new IPASSIGNMENT
or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before IPAssignment_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to IPAssignment_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help IPAssignment

% Last Modified by GUIDE v2.5 26-Aug-2018 23:24:17

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @IPAssignment_OpeningFcn, ...
                   'gui_OutputFcn',  @IPAssignment_OutputFcn, ...
```

```
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before IPAssignment is made visible.
function IPAssignment_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to IPAssignment (see VARARGIN)

% Choose default command line output for IPAssignment
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes IPAssignment wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = IPAssignment_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in LoadImage.
function LoadImage_Callback(hObject, eventdata, handles)
% hObject    handle to LoadImage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
iml={};
```

```
[path, user_cance]=imgetfile();
if user_cance
    magbox(sprintf('Error'),'Error','Error');
    return
end
im=imread(path);
im=im2double(im);
im2=im;
iml{1}=im2;
imlen=1;
axes(handles.axes1);
imshow(im);
```

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

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range
of slider
global im im2 iml imlen imx
it=0.5*get(hObject,'Value')-0.5;
im2=iml{imlen};
imbri=im2+it;
axes(handles.axes1);
imshow(imbri);
```

```
% --- Executes during object creation, after setting all properties.
function Brightness_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Brightness (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes on button press in HistogramEqualisation.
function HistogramEqualisation_Callback(hObject, eventdata, handles)
% hObject    handle to HistogramEqualisation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
if(size(im2,3)==3)
    J=I;
    for j=1:3
        k=J(:,:,j);
        J(:,:,j)= histeq(k);
    end
else
    J = histeq(I);
end


axes(handles.axes1);
imshow(J);
im2=im2double(J);
imlen=imlen+1;
iml{imlen}=im2;


% --- Executes on button press in Gammacorrection.
function Gammacorrection_Callback(hObject, eventdata, handles)
% hObject    handle to Gammacorrection (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
c= str2double(get(handles.Log_tx, 'String'));
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
if(size(im2,3)==3)
    g=rgb2gray(I);
else
    g=I;
end
J = imadjust(g,[],[],c);
axes(handles.axes1);
imshow(J);
im2=im2double(J);
imlen=imlen+1;
iml{imlen}=im2;



% --- Executes on button press in Logtx.
function Logtx_Callback(hObject, eventdata, handles)
% hObject    handle to Logtx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
c= str2double(get(handles.Log_tx, 'String'));
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
if(size(im2,3)==3)
    g=rgb2gray(I);
else
    g=I;
end
[M,N]=size(g);
        for x = 1:M
            for y = 1:N
                m=double(g(x,y));
                z(x,y)=c.*log10(1+m);
            end
        end
axes(handles.axes1);
imshow(z);
im2=im2double(z);
imlen=imlen+1;
iml{imlen}=im2;


% --- Executes on button press in Blur.
function Blur_Callback(hObject, eventdata, handles)
% hObject    handle to Blur (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
if(size(im2,3)==3)
    g=rgb2gray(I);
else
    g=I;
end
intImage = integralImage(g);
avgH = integralKernel([1 1 7 7], 1/49);
J = integralFilter(intImage, avgH);
J = uint8(J);
axes(handles.axes1);
imshow(J);
im2=im2double(J);
imlen=imlen+1;
iml{imlen}=im2;

% --- Executes on button press in Sharpen.
function Sharpen_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to Sharpen (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
if(size(im2,3)==3)
    g=rgb2gray(I);
else
    g=I;
end

lap = [1 1 1; 1 -8 1; 1 1 1];
resp = uint8(filter2(lap, g, 'same'));
imsharp = imsubtract(g, resp);


axes(handles.axes1);
imshow(imsharp);
im2=im2double(imsharp);
imlen=imlen+1;
iml{imlen}=im2;

% --- Executes on button press in Undo.
function Undo_Callback(hObject, eventdata, handles)
% hObject    handle to Undo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
imi=iml{imlen};
for i=1:imlen-1
    iml{imlen-i+1}=iml{imlen-i};
end
iml{1}=imi;
axes(handles.axes1);
im2=iml{imlen};
imshow(im2);


% --- Executes on button press in Undoall.
function Undoall_Callback(hObject, eventdata, handles)
% hObject    handle to Undoall (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
axes(handles.axes1);
imshow(im);
im2=rgb2gray(im);
iml=[im2];
```

```
imlen=1;

% --- Executes on button press in Blackandwhite.
function Blackandwhite_Callback(hObject, eventdata, handles)
% hObject    handle to Blackandwhite (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
imblack=im2;
imblack=1-im2;
axes(handles.axes1);
imshow(imblack);
im2=imblack;
imlen=imlen+1;
iml{imlen}=im2;

% --- Executes on button press in Grayscale.
function Grayscale_Callback(hObject, eventdata, handles)
% hObject    handle to Grayscale (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
gsim=(im2(:,:,1)+im2(:,:,2)+im2(:,:,3))/3;
axes(handles.axes1);
imshow(gsim);
im2=gsim;
imlen=imlen+1;
iml{imlen}=im2;


% --- Executes on button press in Redo.
function Redo_Callback(hObject, eventdata, handles)
% hObject    handle to Redo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
imi=iml{1};
for i=1:imlen-1
    iml{i}=iml{i+1};
end
iml{imlen}=imi;
axes(handles.axes1);
imshow(iml{imlen});

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

```
global im im2 iml imlen
im2=iml{imlen};
[path, user_cance]=imputfile();
imwrite(im2, path, 'png');



function Log_tx_Callback(hObject, eventdata, handles)
% hObject    handle to Log_tx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Log_tx as text
%        str2double(get(hObject,'String')) returns contents of Log_tx
as a double


% --- Executes during object creation, after setting all properties.
function Log_tx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Log_tx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function gamma_text_Callback(hObject, eventdata, handles)
% hObject    handle to gamma_text (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of gamma_text as text
%        str2double(get(hObject,'String')) returns contents of
gamma_text as a double


% --- Executes during object creation, after setting all properties.
function gamma_text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to gamma_text (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on slider movement.
function blur_sli_Callback(hObject, eventdata, handles)
% hObject    handle to blur_sli (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range
of slider
global im im2 iml imlen
im2=iml{imlen};
it=double(get(hObject,'Value'));
disp(it);
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
if(size(im2,3)==3)
    g=rgb2gray(I);
else
    g=I;
end
intImage = integralImage(g);
avgH = integralKernel([1 1 it it], 1/(it*it));
J = integralFilter(intImage, avgH);
J = uint8(J);
axes(handles.axes1);
imshow(J);
im2=im2double(J);
imlen=imlen+1;
iml{imlen}=im2;


% --- Executes during object creation, after setting all properties.
function blur_sli_CreateFcn(hObject, eventdata, handles)
% hObject    handle to blur_sli (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on button press in DFT.
```

```
function DFT_Callback(hObject, eventdata, handles)
% hObject    handle to DFT (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global im im2 iml imlen
im2=iml{imlen};
it=double(get(hObject,'Value'));
disp(it);
imwrite(im2, 'temp_img.tif');
I=imread('temp_img.tif');
J=fft2(I);
axes(handles.axes1);
imshow(J);
im2=im2double(J);
imlen=imlen+1;
iml{imlen}=im2;
```

Assignment by:
Satyaprajna Sarthak Sahoo (16D170026)
Aryan Agal (16D170004)