Digital Image Processing Project

BY

Mohammad Saleh (Fa9520)
GRA at Physics department of WSU
Scientist at European Organization for Nuclear Research (CERN)

ECE5690 Final Project

Submitted to Professor Xiaoyan Han

April 26, 2017

TABLE OF CONTENTS

I. Abstract

Image processing is aimed to extract important or needed information from an image.The main source for images in use today is the electromagnetic energy spectrum, and the visible spectrum is the most usable spectrum in the image processing. The latest rapid progress in the technology lead to the appearance of digital image processing. Nowadays, digital image processing (DIP) has a broad spectrum of applications in science and technology from medicals robotics, automotives, ...etc. Digital image processing can be interpreted as the process of improving image for human interpretation or the process for image compression for storage and transfer. In this booklet, we will present a GUI application for image enhancement for better human interpretation and image compression technique for storage and transfer purpose. This project has been done for the fulfillment of the final project of the ECE5690 class at the electric engineering department of Wayne State University.

II. INTRODUCTION

Digital Image

A digital image can be presented as two-dimensional function $f(x, y)$, where x and y are the spatial coordinates, and $f(x, y)$ is the amplitude for the grey level at that coordinates, usually reffered to a pixel which is the most common name. Digital image processing refer to processing image by digital computers. In this project all the processed images are grayscale images in the visible spectrum, so our eye interpretation will be the main interpretation of the proceessed image.

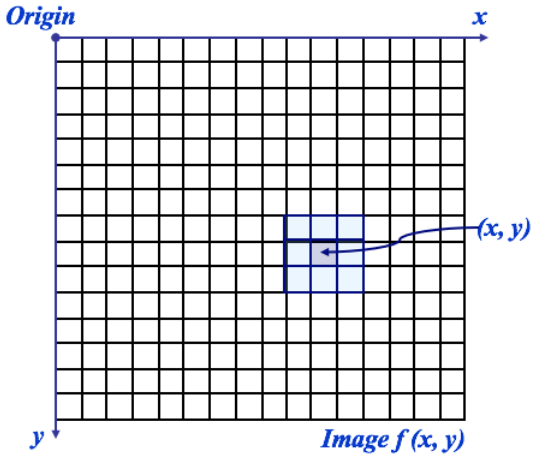

Figure 1: Digital representation of an image

Figure 1 shows the digital representation of an image, where it shows the origin position in the spatial space and the pixel representation. In the following description, we will be explaining image processing procedure on the level of pixels (pixel by pixel). In terms of vector analysis, this means that we will be doing array processes not matrix ones.

We will give a short description how the digital image is obtained. Two process are involved in the digitization process. First, we do sampling by digitizing the coordinate values of the digital signal from the sensor as showed in figure 2. Second, we do the

digitization of the amplitude of the digitized coordinates.

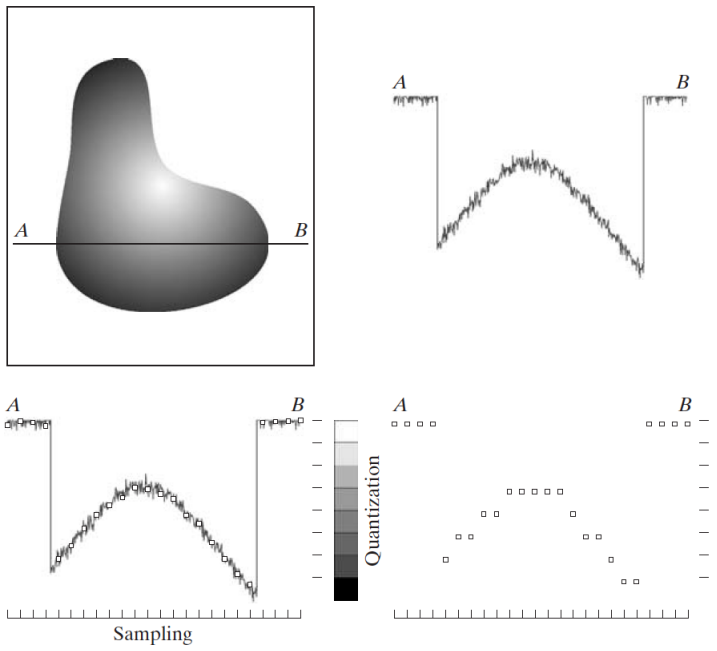

Figure 2: Digitization of an image using sensor scan form A to B. Figure from [1]

We will be talking about different bits of an images. The bit representation of an image is defined as the number of different grey levels in an image. A k-bit image has a range between 0 and $2^k - 1$ in the digital representation $f(x, y)$, as an example 8-bit image has $f(x, y)$ in the range (0, 255). Figure 3 shows the difference between images of different bits. It shows how high bit images exhibit smoother transition in the colors. In the digital image processing, image enhancement is the main goal, this enhancement can be done in the spatial and frequency domain. The frequency domain, is defined as the Fourier transform of the image in the spatial domain ($f(x, y)$). The process is defined as the following:

$$g(x, y) = T[f(x, y)], \tag{1}$$

where $g(x, y)$ is the transformed image and T is defined as the operator on the original image $f(x, y)$. We have different basic intensity transformations. In this project, we

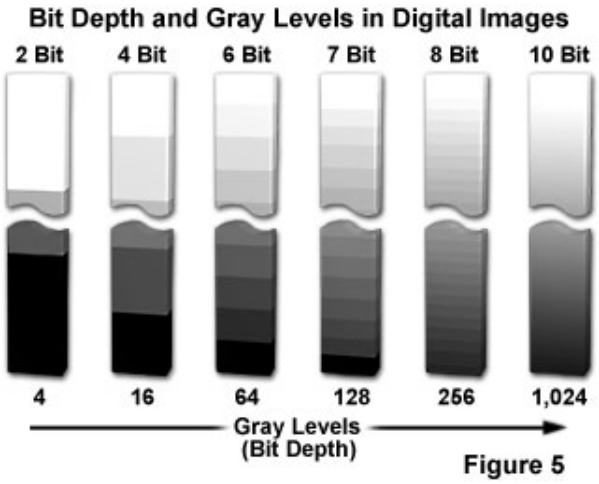

Figure 3: Different bits images

will be including the following basic intensity transformations: the log, power-law, predefined region, the piecewise, and the histogram processing intensity transformation. A more advanced filters can be the smoothening and the sharpening filters in the spatial and frequency domains. We will give a detailed description of these transformations in the upcoming section.

Beside the image enhancement, in one of the sections, we will discuss the process of image compression using a machine learning technique called the principal component analysis. It is one of the most important technique in the image compression, in which most of the image details is conserved using much smaller size than the original size. This section will be for a personal interest, and one of the main reason for going through the class to be able to do this kind of processes on digital images.

In this project, MATLAB was used as the main framework for all the functions and the GUI operations. This project is available on GitHub as an open source project, `https://github.com/salehmoe/Image_processing`, and it can be forked for any contributions for the matter of improving the overall functionality.

III. Functions

In this section we will describe the functions we used and its applicability on some images at the end, we will import 10 images proposed by Professor Han and apply our functions to enhance those images.

A. Intensity Transformation

1. *Importing Image*

The first functionality should be importing the desired image. Images of different extensions can be imported (.jpeg, .tiff, .png, .bmp, ...etc). As we mentioned before, we will be working on grayscale images. One of our first functionality is to convert the imported image to grayscale image in case it is an RGB image. Figure 4 show the two functionality mentioned above.

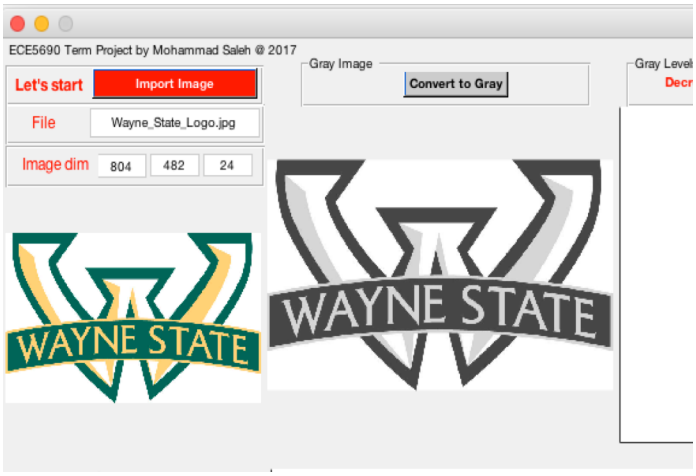

Figure 4: Snapshot of the main GUI for importing and converting the colored image to grayscale image

2. *Decrease the greyscale level*

After importing the image and converting it to a grayscale image, we added the functionality of decreasing the grey scale level of the image. You will have the option

of inserting the factor of which you want the image to be down graded to. As an example, inserting 2 will convert an 8-bit (0-255) image to 7-bit image (0-127). After inserting the factor, a histogram of the image will appear showing the grayscale level range (ex: 0-127) in addition to the mean and the standard deviation of the intensity levels. Figure **??** shows the results of inserting a factor of 4 to an 8-bit images and the histogram of the resulted image. The histogram y-axis is in log-scale for clarity.

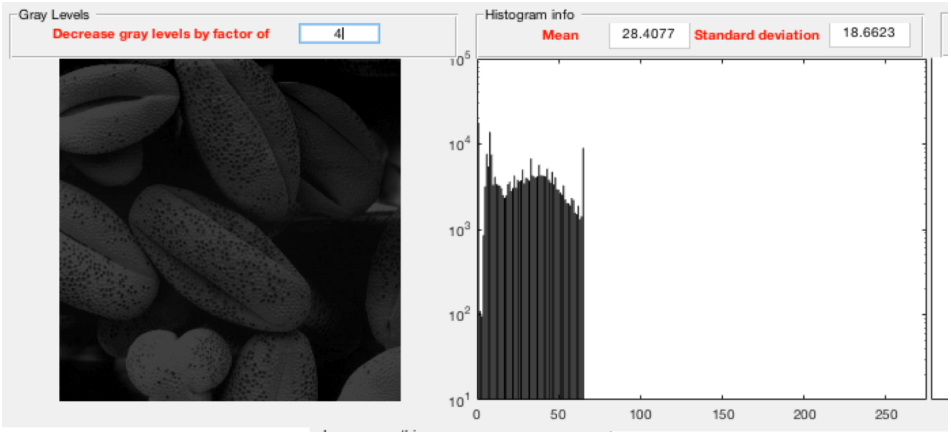

Figure 5: Snapshot of the main GUI for downgrading an 8-bit (0-255) image to 6-bit (0-63) image. Figure from [1].

3. Reseizing Image

After we cintrained how many bits the image is required, we will add a functionality for reseizing the image. To avoid a confusion here, we will give a brief description of the difference between the previous functionality and this functionality. When we worked on downgrading the grey levels, we were working on pixel by pixel and change the amplitude of the pixels. In this functionality, we will increase or decrease the number of pixels in an image. In the GUI, you can insert the factor of reseizing. It is trivial that a number greater than one will increase the number of pixels and a number smaller than one will shrink the image and decrease the number of pixels. We included three options for reseizing the image: nearest-neighborhood, bi-linear

and bi-cubic algorithm. We will describe the three different algorithm.

Nearest Neighbor is best used for categorical data like land-use classification or slope classification. The values that go into the grid stay exactly the same, a 2 comes out as a 2 and 99 comes out as 99. The value of of the output cell is determined by the nearest cell center on the input grid. Nearest Neighbor can be used on continuous data but the results can be blocky [2].

Bi-linear Interpolation uses a weighted average of the four nearest cell centers. The closer an input cell center is to the output cell center, the higher the influence of its value is on the output cell value. This means that the output value could be different than the nearest input, but is always within the same range of values as the input. Since the values can change, Bilinear is not recommended for categorical data. Instead, it should be used for continuous data like elevation and raw slope values [2].

Bi-cubic Convolution looks at the 16 nearest cell centers to the output and fits a smooth curve through the points to find the value. Not only does this change the values of the input but it could also cause the output value to be outside of the range of input values (imagine a sink or a peak occurring on a surface). This method is also not recommended for categorical data, but does an excellent job of smoothing continuous data [2].

Figure **??** shows the GUI for reseizing the image and how you can choose the desired algorithm.

4. Intensity transformation

The intensity transformation play very important role in the enhancement process according to the desired interest. Six different intensity transformation functionality were implemented. We will go through each one and how it works.

Negative intensity transformation is very well known publicly as it is usually used in some public field. The negative of an image with range [0, L-1] is obtained by reversing the intensity levels of an image which usually produce the well known negative

Figure 6: Snapshot of the main GUI resezing an image using different options. Figure from [1].

photographic. This transformation is given by the following expression.

$$s = L - 1 - r. \tag{2}$$

This process is suited for enhancing white or grey details embedded in a dark dominant region. Figure **??** shows a negative of an image shown in previous examples. Log transformation maps a narrow range of low intensity values in the input into a wider range of output levels which it compresses the dynamic ranges of images with large variations in pixel values. The general form of the transformation is defined as:

$$s = c * log(1 + r), \tag{3}$$

where c is a constant, and r>0. In our GUI you can specify the value of c needed. Figure 8 shows an example of log transforamtion for c=1. Power-Law (Gamma) transformation is defined as the following:

$$s = c * r^{\gamma}, \tag{4}$$

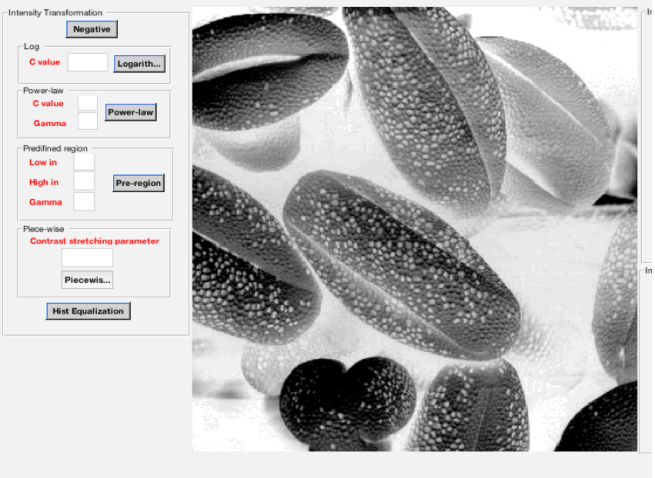

Figure 7: Negative of an image. Figure from [1].

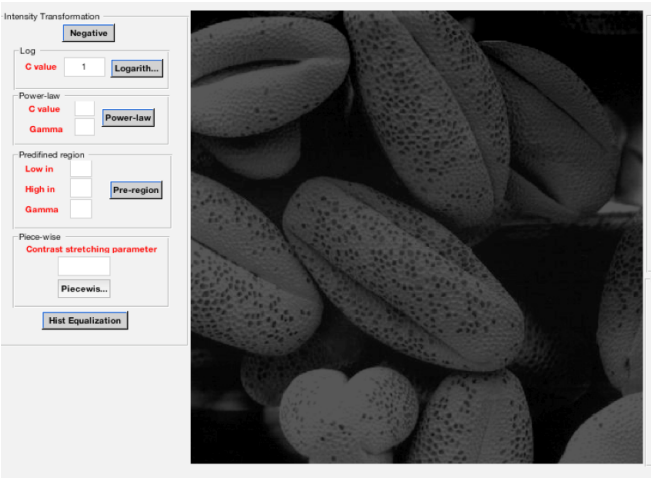

Figure 8: log transformation of an image. Figure from [1].

where c and γ are positive constants. It is more general than the log transformation where a family of possible transformation curve can be obtained by varying γ. It is very well known as the gamma correction. It is widely used for calibrating the monitors. Figure 9 shows an example of power-law transformation for some values of c and γ. This transformation will be very useful when we do enhancement for the assigned images.

The pre-defined region transformation is a simple one where we can choose a certain region of the histogram to be transformed according to certain value of γ.

Figure 9: Power-law transformation of an image for c=1 and γ=0.3. Figure from [1].

Figure10 shows an example of this transformation for region 0-0.4 and γ value 0.8.

Figure 10: Pre-defined region transformation of an image for γ=0.8. Figure from [1].

One of the simplest piece-wise transformation is a contrast-stretching. Contrast stretching is a process that expands the range of intensity levels in an image. This transformation increases the contrast between the darks and the lights. In simple words, the dark becomes darker and the light becomes brighter. Usually this transformation increase the contrast of an image. Figure 11 shows a piece-wise transformation of an image. The last intensity transformation in our GUI is the histogram equalization. This transformation is based on the histogram in the previous GUI

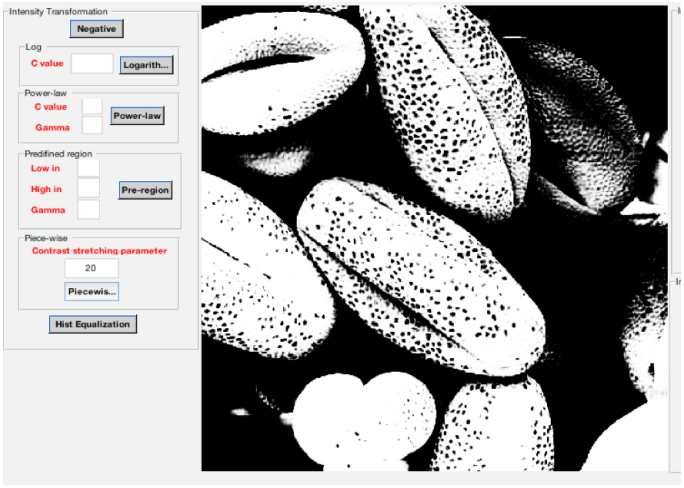

Figure 11: Piece-wise intensity transformation of an image. Figure from [1].

panels. The output image will have a uniform distribution from weighting the original image histogram by a specific weight. After pressing the histogram equalization, the histogram will be updated with the new image histogram, an example of this transformation is shown in figure 12.

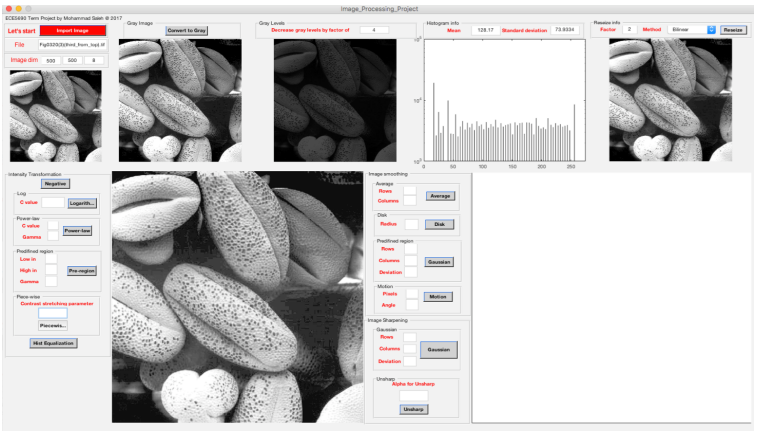

Figure 12: Histogram equalization transformation of an image with the updated histogram. Figure from [1].

B. Filtering in spatial domain

1. *Image smoothing*

Two filters were implemented for image smoothing in spatial domain. The Gaussian and motion filter. In the Gaussian filter, you have the option to specify the mask dimension (3x3, 5x5, ...etc) and how much the deviation needed in the filtering mask from the mean. The Gaussian filter work on the neighbor of each pixel as specified that we can choose the mask dimension. In the other hand, the motion filter is designed to correct the images that acquired smoothening from a motion, the blurness that you get from slow shutter in photography as an example. In the motion blur filtering, you specify the number included in this smoothening and the angle of this smoothening, Figure 13 shows an example of motion smoothening filter.

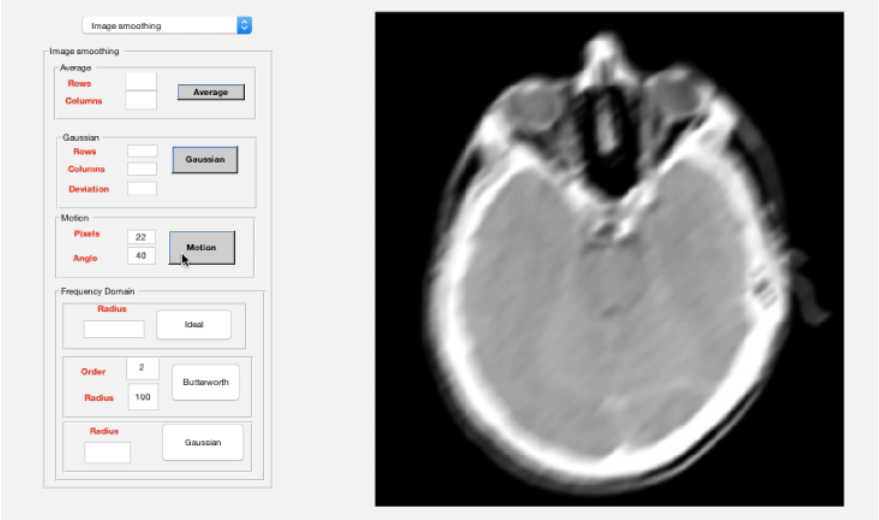

Figure 13: Motion transformation of an image. Figure from [1].

2. *Image sharpening*

Two filter were also implemented and Gaussian is still one of the two filters. The second sharpening filter is the unsharp filtering. The way how the unsharp mask work is by smoothing an image and subtract it from the original image. The obtained image

is called the masked image and by adding this mask to the original image, we get the sharpened image. A snapshot of the GUI for the sharpening is shown in figure 14

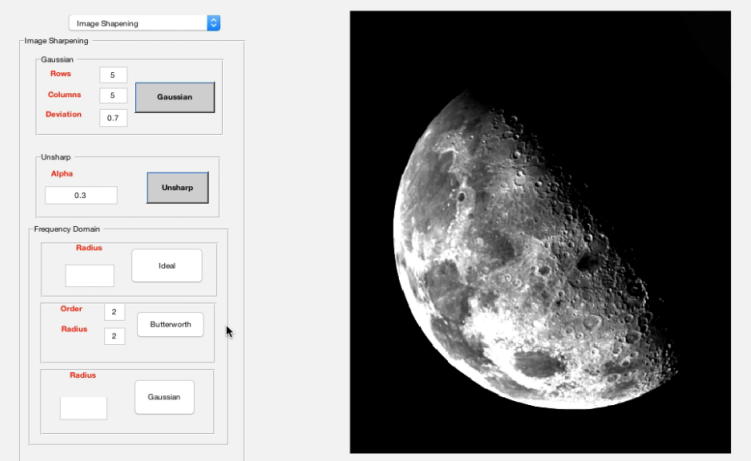

Figure 14: Snapshot of the GUI for the two sharpening functions in spatial domain. Figure from [1].

C. Filtering in frequency domain

Filtering in the frequency domain consists of modifying the Fourier transform of an image and then computing the inverse transform to obtain the processed result. The process is given by:

$$g(x, y) = \mathscr{F}^{-1}[H(u, v)F(u, v)], \tag{5}$$

where $\mathscr{F}^{-1}$ is the IDFT, $F(u, v)$ is the DFT of the input image, $f(x, y)$, $H(u, v)$ is a filter function, and $g(x, y)$ is the output image. the product of $H(u, v)F(u, v)$ is done using array multiplication.

In the frequency domain, filters are categorized in two categories, the low and the high pass filters. Low pass filters refer to smoothening filters, while high pass filters refer to sharpening filters. Low (high) pass filter passes the low (high) frequency and attenuate the others. We implemented three different filters for the smoothening and the sharpening. The Ideal, Gaussian, and the Butterworth filter.

Ideal low pass filter is a 2-D filter that passes without attenuation all frequencies within a circle of radius D_0 from the origin and cut off all the frequencies outside the circle. Knowing this, the filter is formulated as:

$$H(u,v) = 1 \; for \; D(u,v) \leq D_0 \; and \; zero \; otherwise. \tag{6}$$

where D_0 is a constant number entered by the user as shown in figure 15.

Gaussian low pass filter is defined as:

$$H(u,v) = exp(-D^2(u,v)/2\sigma^2), \tag{7}$$

where D_0 is the cutoff frequency and GLPF is down to 0.607 of its maximum value when $D(u,v) = D_0$

Butterworth lowpass filter is defined as:

$$H(u,v) = 1/(1 + [D(u,v)/D_0]^{2n}), \tag{8}$$

where n is the order of the filter and D_0 is the cutoff frequency. In our GUI, you can input the order n and the cutoff frequency D_0 as shown in figure 15

In the other hand, the high frequency domain filters is the opposite of the low pass filters, where

$$H_{HP}(u,v) = 1 - H_{LP}(u,v) \tag{9}$$

D. Noise Reduction

For noise reduction, we implemented three functions, the mean, median, and the adaptive filter.

The mean filter is obtained from the arithmetic mean. Let S_{xy} represent the set of coordinates in a rectangular neighbor of size mxn, the arithmetic mean compute the average value of the corrupted image $g(x,y)$ in the area defined by S_{xy} [1]. The

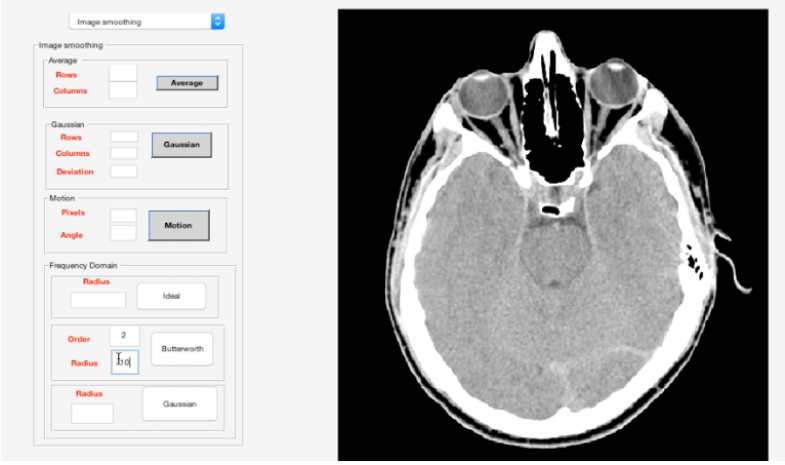

Figure 15: Image smoothening in frequency domain. Figure from [1].

value of the restored image is the arithmetic mean defined as:

$$\hat{f}(x,y) = (1/mn) \sum g(s,t) \tag{10}$$

The adaptive filter in the GUI layout also take the neighborhood dimension as a user input. The median filter, is the best known order-statistic filter which is as the name implies, it replaces the value of the pixel by the median of the intensity levels in the neighborhood of that pixel. In our GUI we implemented the median filter, where you can choose the dimension of the neighborhood as a user input (figure 16).

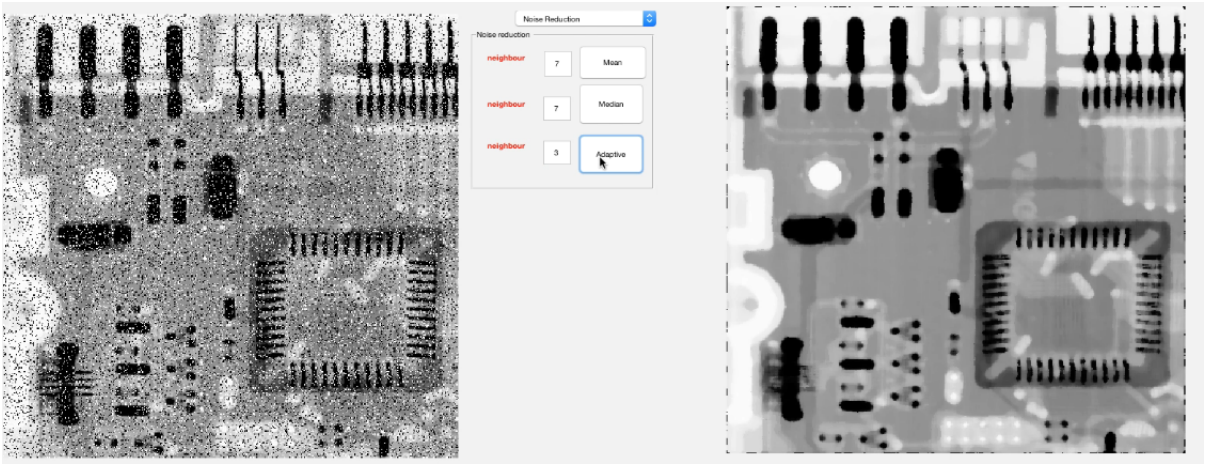

Figure 16: Noise reduction layout in the GUI. Figure from [1].

E. Image Compression

The last step in our GUI is the image compression using the principal component analysis (PCA). PCA consists of projecting the original image to smaller dimension component image. The new dimension characterized as number of component image. Figure 17 shows compressed image by keeping the first two component only, Figure 18 show the compressed image by keeping the first hundred components of the original 500 ones. The new image size is 37 KB with very good quality from the original 373 KB image.

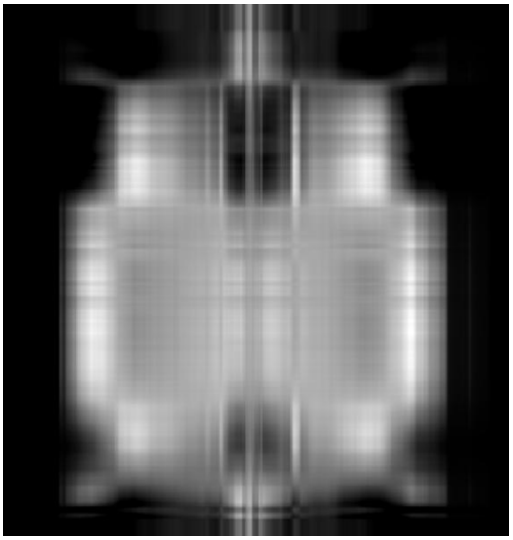

Figure 17: Compressed image by keeping the fist two components. Figure from [1].

Figure 18: Compressed image by keeping the fist hundred components. Figure from [1].

IV. Testing the functions

In this section, we will test our function using ten images provided by Professor Han. For each image, we will show the final proceeded image and the filters added to improve or to get the intended image.

Figure 19 shows the relevant spot that need to be enhanced using our functions. Figure 20 show the enhanced image. It shows a clean image of the intended spot.

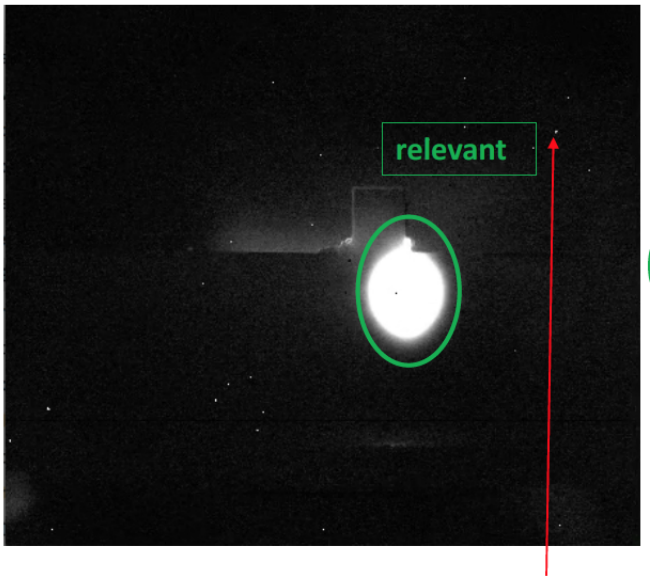

Figure 19: Image 1 of the images that need enhancement. Figure from [1].

The enhancement is well documented using a video included in the package. The image enhamcement is done as the following:
Log transformation (c=5) ===> Median noise reduction filter (45 input value) ===> Power-law transformation (c=3, γ=7) ===> Median noise reduction (5 as input value).

Figure 21 shows the image before enhancement and figure 22 shows the image after enhancement. The filtered applied to this image is the following:
Log transformation (c=10) ===> Image sharpening using Butterworth (order 2, D=4) ===> Log transformation (c=5).

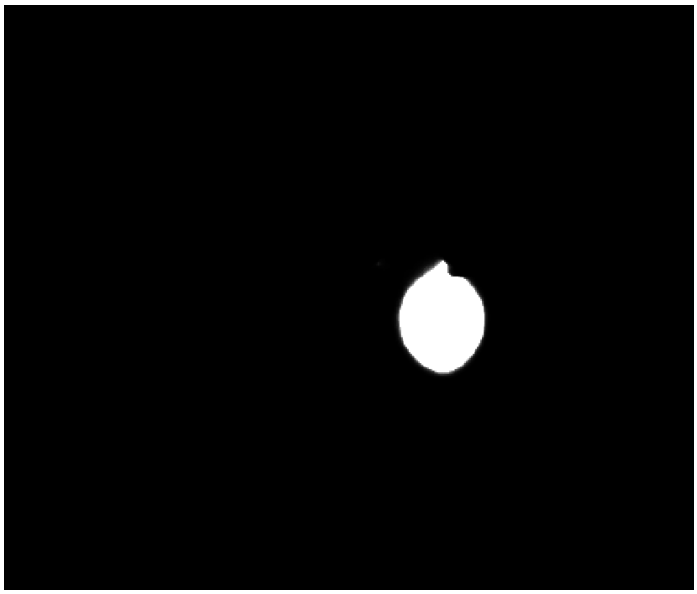

Figure 20: The enhancement of image 1. Figure from [1].

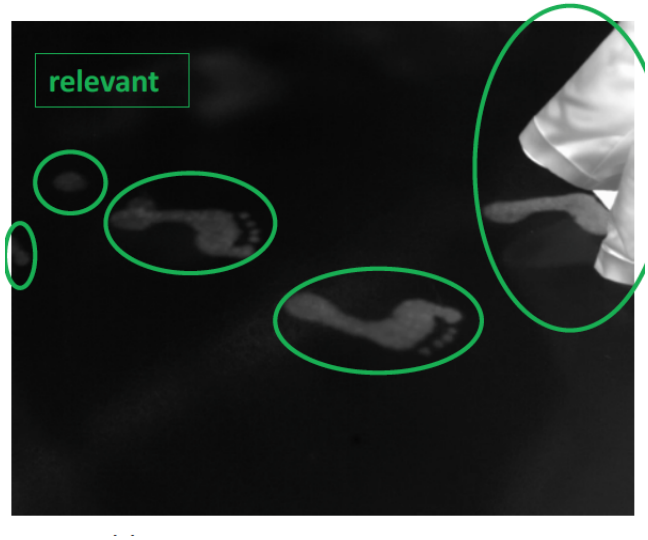

Figure 21: Image 2 of the images that need enhancement. Figure from [1].

For figure 23 we obtained the enhanced image in figure 24. It is clear that we have the noise that needed to be removed. Below is the following filters that applied to the image.

Gaussian sharpening (2x2, deviation=0.9) ===> Power-law (c=2, $\gamma = 5$) ===> Noise Reduction using median filter (5x5 mask).

Figure 22: The enhancement of image 2. Figure from [1].

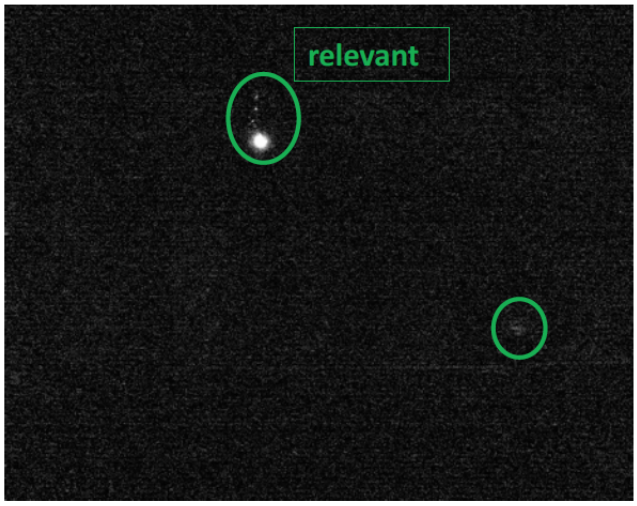

Figure 23: Image 3 of the images that need enhancement. Figure from [1].

For figure 25 we obtained the enhanced image in figure 26. It is clear that we have less noise in the enhanced image but not all the noise removed. Below is the following filters that applied to the image.

Butterworth smoothening (order 2, D_0=60) ===> Power-law (c=2, $\gamma = 2$). Many combinations of filters have been tried but this combination results was better than others.

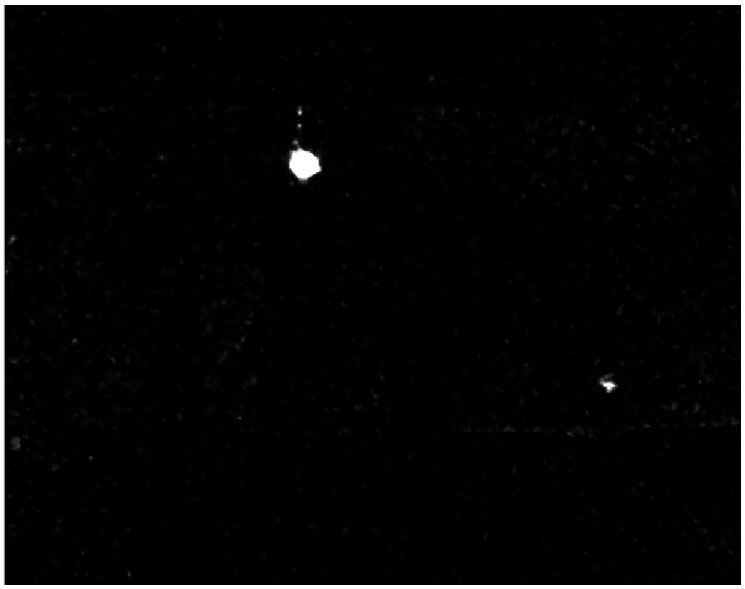

Figure 24: The enhancement of image 3. Figure from [1].

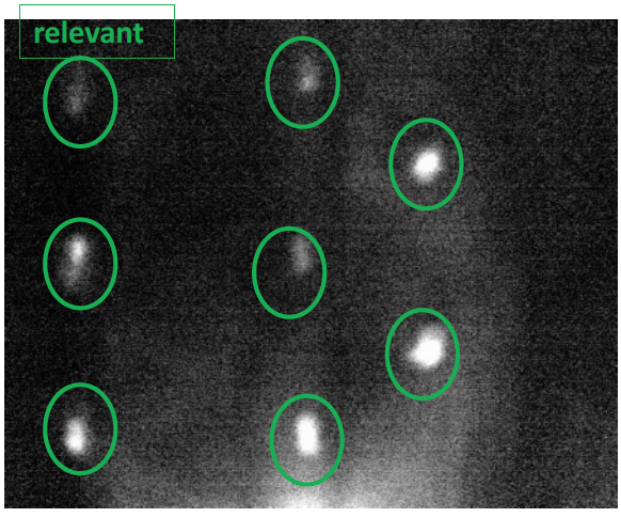

Figure 25: Image 4 of the images that need enhancement. Figure from [1].

Figure 28 show the enhanced image. It shows a clean image of the intended spot. The image enhancement is done as the following:

Gaussian sharpening (4x4, deviation 0.5) ===> Power-law transformation (c=2, γ=12) ===> Median noise reduction (4x4 as input value for mask).

Figure 30 show the enhanced image. It shows more sharper image of the relevant

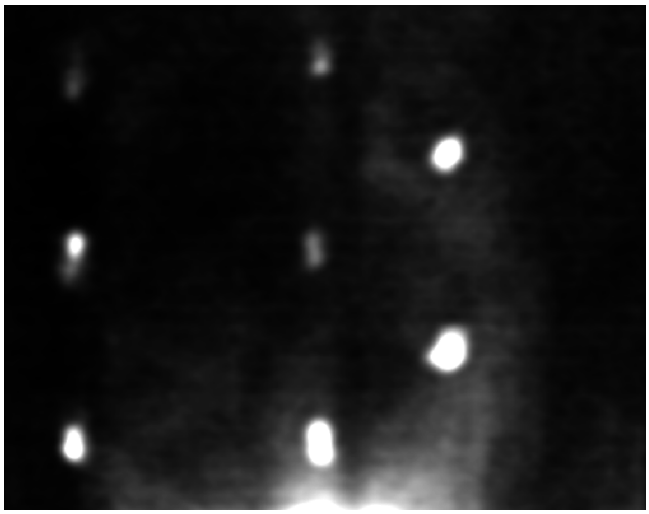

Figure 26: The enhancement of image 4. Figure from [1].

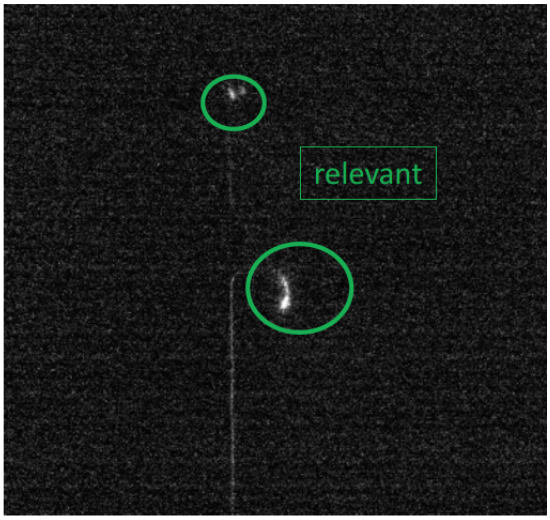

Figure 27: Image 5 of the images that need enhancement. Figure from [1].

spot but the unintended spots still there. The image enhamcement is done as the following:

Butterworth smoothing (order 2, $D_0 = 60$) ===> Subtract the smoothened image from the original ===> Log-transformation with $c = 12$. Probably a 0 1 mask could help by working on the center of the image but this would be as cropping the image and not enhancing it. Figure 32 show the enhanced image. It shows more clearer image for the intended spot but the unintended bright spot still there. The image

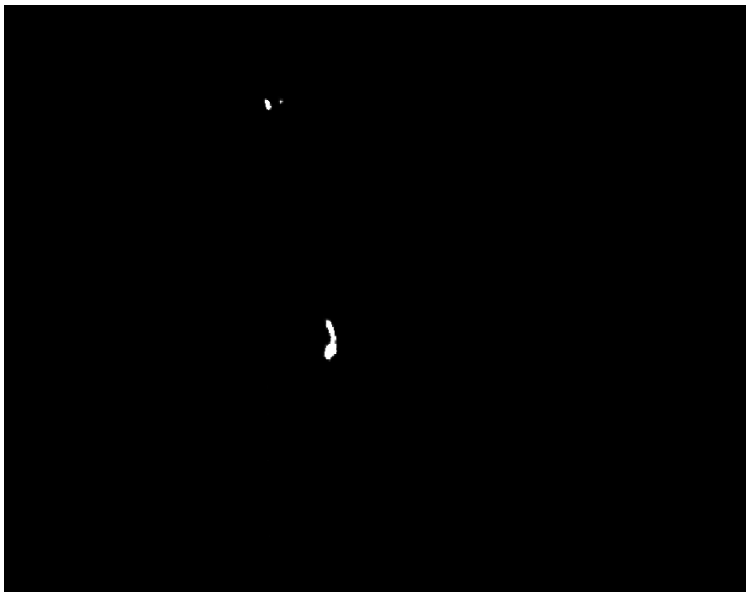

Figure 28: The enhancement of image 5. Figure from [1].

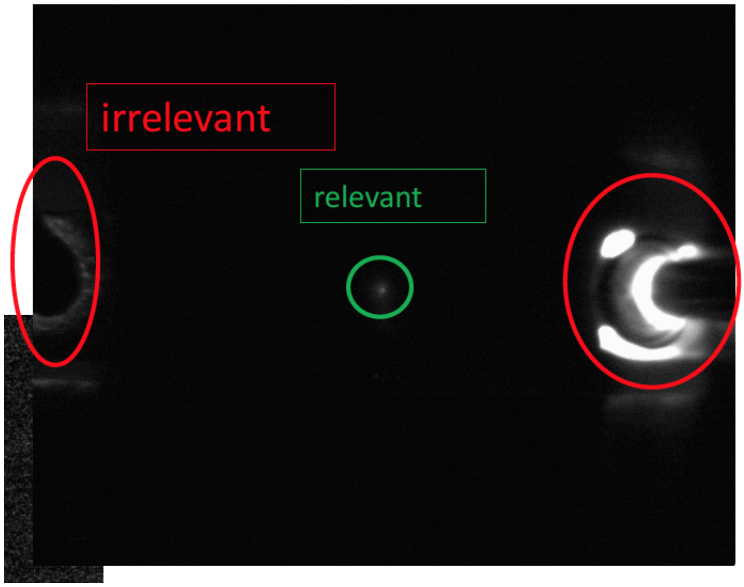

Figure 29: Image 6 of the images that need enhancement. Figure from [1].

enhancement is done as the following:

Butterworth smoothing (order 2, $D_0 = 80$) ===> Power-law transformation with $c = 2$ and $\gamma = 2$.

For figure 33 we obtained the enhanced image in figure 34. It is clear that all the

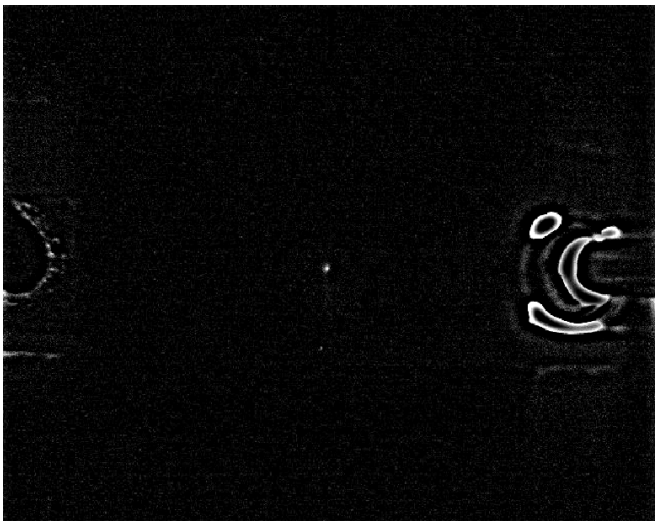

Figure 30: The enhancement of image 6. Figure from [1].

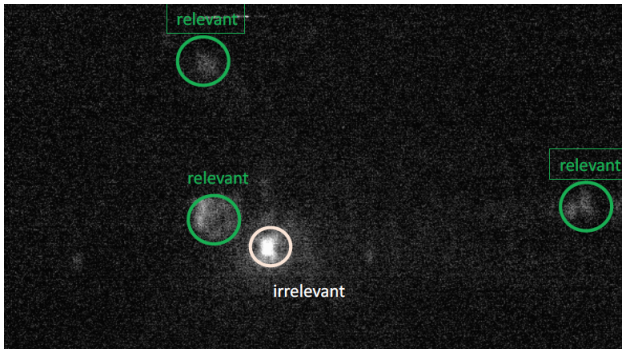

Figure 31: Image 7 of the images that need enhancement. Figure from [1].

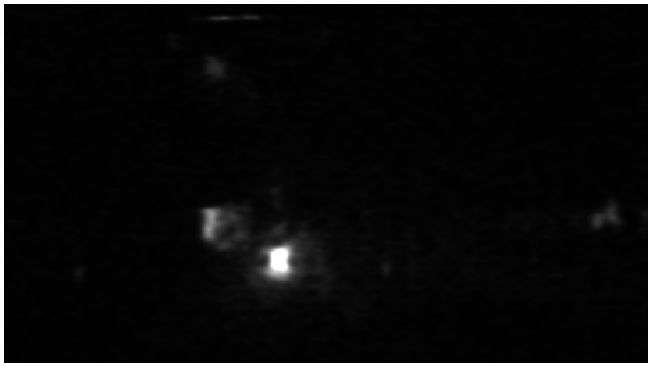

Figure 32: The enhancement of image 7. Figure from [1].

noise were removed form the image. Below is the following filters that applied to the image.

Butterworth smoothening (order 2, D_0=50) ===> Power-law (c=1, $\gamma = 2$). Many combinations of filters have been tried but this combination results was better than others.

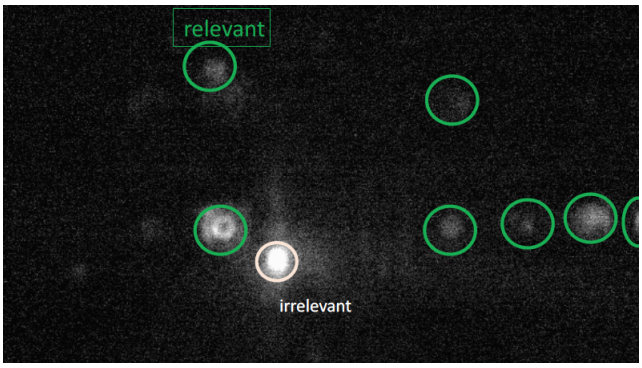

Figure 33: Image 8 of the images that need enhancement. Figure from [1].

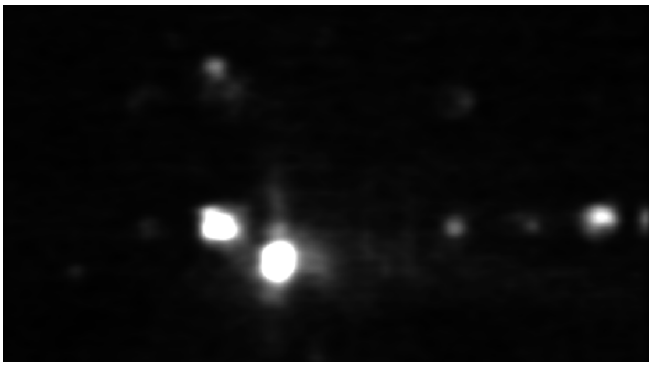

Figure 34: The enhancement of image 8. Figure from [1].

For figure 35 we obtained the enhanced image in figure 36. It is clear that all the noise were removed form the image. Below is the following filters that applied to the image.

Adaptive noise reduction filter (18 as input user) ===> Power-law (c=2, $\gamma = 3$).

For figure 37 we obtained the enhanced image in figure 38. It is clear that all the noise were removed form the image and the unwanted spot got separated from the wanted ones. Below is the following filters that applied to the image.

Power-law (c=2, $\gamma = 22$) ===> Median noise reduction filter (5x5 mask) .

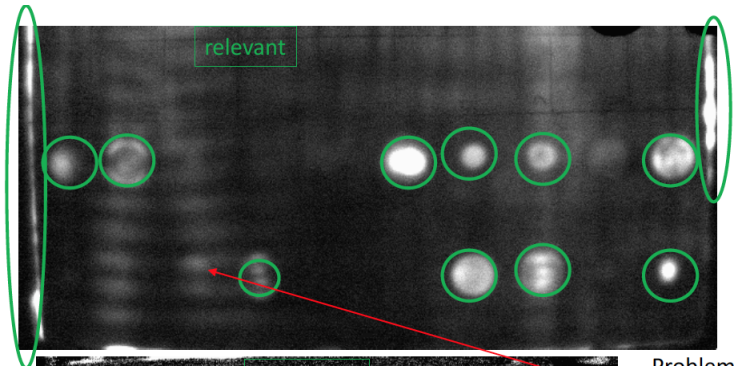

Figure 35: Image 9 of the images that need enhancement. Figure from [1].

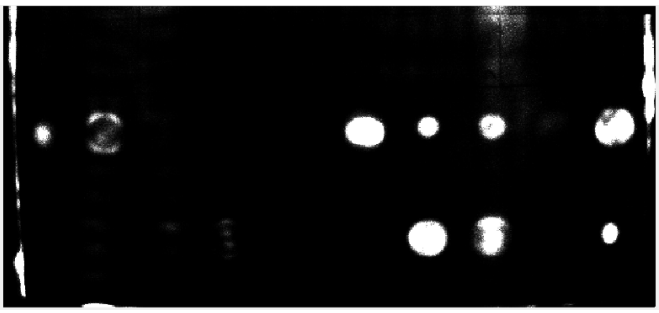

Figure 36: The enhancement of image 9. Figure from [1].

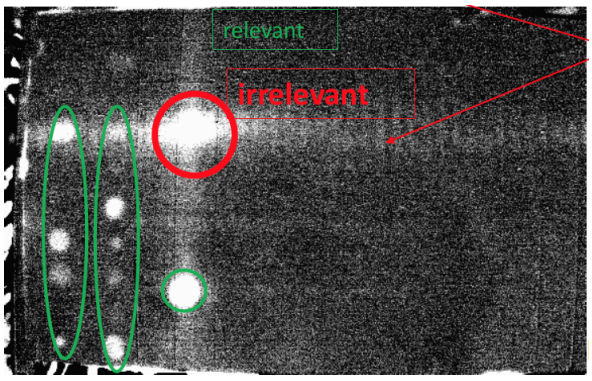

Figure 37: Image 10 of the images that need enhancement. Figure from [1].

Figure 38: The enhancement of image 10. Figure from [1].

V. Conclusion

We have performed the required functions and the testing for these functions as required by the course. This project can be developed more, for faster prediction of the needed filters based on the data of the image. I would like to thanks Professor Han at WSU for the lectures regarding all the functions used in this GUI layout.

VI. Appendices

A. Appendix A (code, the .m file for the project)

```
function varargout = Image_Processing_Project(varargin)
% IMAGE_PROCESSING_PROJECT MATLAB code for Image_Processing_Project.fig
%      IMAGE_PROCESSING_PROJECT, by itself, creates a new IMAGE_PROCESSING_PROJECT or raises the existing
%      singleton*.
%
%      H = IMAGE_PROCESSING_PROJECT returns the handle to a new IMAGE_PROCESSING_PROJECT or the handle to
%      the existing singleton*.
%
%      IMAGE_PROCESSING_PROJECT('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in IMAGE_PROCESSING_PROJECT.M with the given input arguments.
%
%      IMAGE_PROCESSING_PROJECT('Property','Value',...) creates a new IMAGE_PROCESSING_PROJECT or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Image_Processing_Project_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Image_Processing_Project_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 Image_Processing_Project

% Last Modified by GUIDE v2.5 27-Apr-2017 07:16:00

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Image_Processing_Project_OpeningFcn, ...
                   'gui_OutputFcn',  @Image_Processing_Project_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 Image_Processing_Project is made visible.
function Image_Processing_Project_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 Image_Processing_Project (see VARARGIN)

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

% Update handles structure
guidata(hObject, handles);

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


% --- Outputs from this function are returned to the command line.
function varargout = Image_Processing_Project_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 pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename pathname]= uigetfile({'*.jpg';'*.bmp';'*.png';'*.tif'},'File Selector');
handles.image_1= strcat(pathname, filename);
axes(handles.axes1)
imshow(handles.image_1)
InfoImage = imfinfo(handles.image_1);
set(handles.edit7,'String',num2str(InfoImage.Width));
set(handles.edit8,'String',num2str(InfoImage.Height));
set(handles.edit9,'String',num2str(InfoImage.BitDepth));
set(handles.Im_name,'string',filename);
image_2 = imread(handles.image_1);
if size(image_2,3)==1
    handles.gray_image=image_2;
else
```

```
    handles.gray_image=rgb2gray(image_2);
end
handles.grey_1=handles.gray_image;
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function pushbutton1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over text3.
function text3_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to text3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
filename



function Im_name_Callback(hObject, eventdata, handles)
% hObject    handle to Im_name (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 Im_name as text
%        str2double(get(hObject,'String')) returns contents of Im_name as a double


% --- Executes during object creation, after setting all properties.
function Im_name_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Im_name (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 button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cam=webcam;
handles.image_1 = snapshot(cam);
closePreview(cam);
axes(handles.axes1)
imshow(handles.image_1);
handles.gray_image=rgb2gray(handles.image_1);
guidata(hObject,handles);

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
image_2 = imread(handles.image_1);
if size(image_2,3)==1
    handles.gray_image=image_2;
else
    handles.gray_image=rgb2gray(image_2);
end
guidata(hObject,handles);
axes(handles.axes2);
imshow(handles.gray_image);
guidata(hObject,handles);


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

guidata(hObject,handles);



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (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 edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (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.
```

```
195 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
196     set(hObject,'BackgroundColor','white');
197 end
198
199
200
201 function edit3_Callback(hObject, eventdata, handles)
202 % hObject    handle to edit3 (see GCBO)
203 % eventdata  reserved - to be defined in a future version of MATLAB
204 % handles    structure with handles and user data (see GUIDATA)
205
206 % Hints: get(hObject,'String') returns contents of edit3 as text
207 %        str2double(get(hObject,'String')) returns contents of edit3 as a double
208
209
210 % --- Executes during object creation, after setting all properties.
211 function edit3_CreateFcn(hObject, eventdata, handles)
212 % hObject    handle to edit3 (see GCBO)
213 % eventdata  reserved - to be defined in a future version of MATLAB
214 % handles    empty - handles not created until after all CreateFcns called
215
216 % Hint: edit controls usually have a white background on Windows.
217 %       See ISPC and COMPUTER.
218 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
219     set(hObject,'BackgroundColor','white');
220 end
221
222
223 % --- Executes on button press in pushbutton5.
224 function pushbutton5_Callback(hObject, eventdata, handles)
225 % hObject    handle to pushbutton5 (see GCBO)
226 % eventdata  reserved - to be defined in a future version of MATLAB
227 % handles    structure with handles and user data (see GUIDATA)
228
229
230 % --- Executes on slider movement.
231 function slider2_Callback(hObject, eventdata, handles)
232 % hObject    handle to slider2 (see GCBO)
233 % eventdata  reserved - to be defined in a future version of MATLAB
234 % handles    structure with handles and user data (see GUIDATA)
235 handles.a=get(handles.slider2,'Value');
236 %returns position of slider
237 handles.Grey_quant=(handles.gray_image)/(str2double(handles.grayValues));
238 axes(handles.axes4);
239 imshow(handles.Grey_quant)
240 guidata(hObject,handles);
241 axes(handles.axes3);
242 [pixelCount grayLevels]=imhist(handles.Grey_quant);
243 %subplot(1, 2, 2);
244 bar(pixelCount);
245 title('');
246 set(gca,'YScale','log')
247 xlim([0 grayLevels(end)+20]); % Scale x axis manually.
248 yRange = ylim;
249 Im_mean = mean2(handles.Grey_quant);
250 Im_std = std2(handles.Grey_quant);
251 set(handles.edit2,'string',Im_mean);
252 set(handles.edit3,'string',Im_std);
253 maxGrayLevel = max(handles.Grey_quant(:));
254 set(handles.edit10,'String',num2str(maxGrayLevel));
255 guidata(hObject,handles);
256
257 %        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
258
259
260 % --- Executes during object creation, after setting all properties.
261 function slider2_CreateFcn(hObject, eventdata, handles)
262 % hObject    handle to slider2 (see GCBO)
263 % eventdata  reserved - to be defined in a future version of MATLAB
264 % handles    empty - handles not created until after all CreateFcns called
265
266 % Hint: slider controls usually have a light gray background.
267 if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
268     set(hObject,'BackgroundColor',[.9 .9 .9]);
269 end
270
271
272 % --- Executes on button press in pushbutton6.
273 function pushbutton6_Callback(hObject, eventdata, handles)
274 % hObject    handle to pushbutton6 (see GCBO)
275 % eventdata  reserved - to be defined in a future version of MATLAB
276 % handles    structure with handles and user data (see GUIDATA)
277 image_2=imread(handles.image_1);
278 image_Reseize=imresize(image_2,str2double(handles.reseize),handles.reseize_Method);
279 axes(handles.axes5);
280 imshow(image_Reseize)
281
282
283 function edit4_Callback(hObject, eventdata, handles)
284 % hObject    handle to edit4 (see GCBO)
285 % eventdata  reserved - to be defined in a future version of MATLAB
286 % handles    structure with handles and user data (see GUIDATA)
287
288 % Hints: get(hObject,'String') returns contents of edit4 as text
289 %        str2double(get(hObject,'String')) returns contents of edit4 as a double
290 handles.reseize=get(hObject,'String');
291 guidata(hObject,handles);
292
293 % --- Executes during object creation, after setting all properties.
294 function edit4_CreateFcn(hObject, eventdata, handles)
295 % hObject    handle to edit4 (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 edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (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 edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a double
handles.reseize_method=get(hObject,'String');
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (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 button press in radiobutton1.
function radiobutton1_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton1



function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (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 edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as a double
set(handles.edit6,'string',handles.reseize_method);

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (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 selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
contents = get(hObject,'Value')
switch contents
    case 1
        do_none=1;
    case 2
        handles.reseize_Method='nearest';
    case 3
        handles.reseize_Method='bilinear';
    case 4
        handles.reseize_Method='bicubic';
    otherwise
end
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu 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 button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton2



function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (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 edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as a double
set(handles.edit7,'string',size(handles.image_1));
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (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 edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (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 edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as a double


% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (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 selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from listbox1


% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: listbox 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 when uipanel5 is resized.
function uipanel5_SizeChangedFcn(hObject, eventdata, handles)
% hObject    handle to uipanel5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (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 edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as a double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (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 edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (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 edit10 as text
%        str2double(get(hObject,'String')) returns contents of edit10 as a double


% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (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 selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu3
bit_value = get(hObject,'Value')
switch bit_value
    case 1
        handles.bits=16;
    case 2
         handles.bits=8;
    case 3
         handles.bits=4;
    otherwise
end
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu 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 edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

handles.grayValues=get(hObject,'String');
%returns position of slider
handles.Grey_quant=(handles.gray_image)/(str2double(handles.grayValues));
axes(handles.axes4);
imshow(handles.Grey_quant)
guidata(hObject,handles);
axes(handles.axes3);
[pixelCount grayLevels]=imhist(handles.Grey_quant);
%subplot(1, 2, 2);
bar(pixelCount);
title('');
set(gca,'YScale','log')
xlim([0 grayLevels(end)+20]); % Scale x axis manually.
yRange = ylim;
Im_mean = mean2(handles.Grey_quant);
Im_std = std2(handles.Grey_quant);
set(handles.edit2,'string',Im_mean);
set(handles.edit3,'string',Im_std);
maxGrayLevel = max(handles.Grey_quant(:));
guidata(hObject,handles);

%returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of edit11 as a double


% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (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.
```

```
599 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
600     set(hObject,'BackgroundColor','white');
601 end
602
603
604 % --- Executes on selection change in popupmenu4.
605 function popupmenu4_Callback(hObject, eventdata, handles)
606 % hObject    handle to popupmenu4 (see GCBO)
607 % eventdata  reserved - to be defined in a future version of MATLAB
608 % handles    structure with handles and user data (see GUIDATA)
609 intensity = get(hObject,'Value')
610 switch intensity
611     case 1
612         a;
613     case 2
614         handles.low=0;
615         handles.high=1;
616
617     case 3
618         handles.low=0;
619         handles.high=1;
620     otherwise
621 end
622 guidata(hObject,handles);
623 % Hints: contents = cellstr(get(hObject,'String')) returns popupmenu4 contents as cell array
624 %        contents{get(hObject,'Value')} returns selected item from popupmenu4
625
626
627 % --- Executes during object creation, after setting all properties.
628 function popupmenu4_CreateFcn(hObject, eventdata, handles)
629 % hObject    handle to popupmenu4 (see GCBO)
630 % eventdata  reserved - to be defined in a future version of MATLAB
631 % handles    empty - handles not created until after all CreateFcns called
632
633 % Hint: popupmenu controls usually have a white background on Windows.
634 %       See ISPC and COMPUTER.
635 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
636     set(hObject,'BackgroundColor','white');
637 end
638
639
640
641 function edit12_Callback(hObject, eventdata, handles)
642 % hObject    handle to edit12 (see GCBO)
643 % eventdata  reserved - to be defined in a future version of MATLAB
644 % handles    structure with handles and user data (see GUIDATA)
645 g=im2double(handles.gray_image);
646 c=str2double(get(hObject,'String'));
647
648 [M,N]=size(g);
649         for x = 1:M
650             for y = 1:N
651                 m=g(x,y);
652                 z(x,y)=c.*log10(1+m); %#ok<AGROW>
653             end
654         end
655 axes(handles.axes7);
656 imshow(z)
657  %returns contents of edit12 as text
658 %        str2double(get(hObject,'String')) returns contents of edit12 as a double
659
660
661 % --- Executes during object creation, after setting all properties.
662 function edit12_CreateFcn(hObject, eventdata, handles)
663 % hObject    handle to edit12 (see GCBO)
664 % eventdata  reserved - to be defined in a future version of MATLAB
665 % handles    empty - handles not created until after all CreateFcns called
666
667 % Hint: edit controls usually have a white background on Windows.
668 %       See ISPC and COMPUTER.
669 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
670     set(hObject,'BackgroundColor','white');
671 end
672
673
674 % --- Executes on button press in Negative.
675 function Negative_Callback(hObject, eventdata, handles)
676 % hObject    handle to Negative (see GCBO)
677 % eventdata  reserved - to be defined in a future version of MATLAB
678 % handles    structure with handles and user data (see GUIDATA)
679  handles.gray_image=imadjust(handles.gray_image,[0  1],[1  0]);
680  axes(handles.axes7);
681  imshow(handles.gray_image)
682 guidata(hObject,handles);
683
684 % --- Executes on button press in log.
685 function log_Callback(hObject, eventdata, handles)
686 % hObject    handle to log (see GCBO)
687 % eventdata  reserved - to be defined in a future version of MATLAB
688 % handles    structure with handles and user data (see GUIDATA)
689
690
691 % --- Executes on button press in pow.
692
693 % hObject    handle to pow (see GCBO)
694 % eventdata  reserved - to be defined in a future version of MATLAB
695 % handles    structure with handles and user data (see GUIDATA)
696
697
698 % --- Executes on button press in piece.
699 function piece_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to piece (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
g=im2double(handles.gray_image);
m=mean2(g);
E=str2double(handles.pre_value);
[M,N]=size(g);
        for x = 1:M
            for y = 1:N
                gm=g(x,y);
                z(x,y)=1/(1+(m/gm)^E);
            end
        end
axes(handles.axes7);
handles.gray_image=z;
imshow(handles.gray_image)
guidata(hObject,handles);

% Hint: get(hObject,'Value') returns toggle state of piece


% --- Executes on button press in pre.
function pre_Callback(hObject, eventdata, handles)
% hObject    handle to pre (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
gm=str2double(handles.gamma_pre);
low=str2double(handles.reg_Low);
 high=str2double(handles.reg_Hi);
grey_Reg=imadjust(handles.gray_image,[low high],[0 1],gm);
 axes(handles.axes7);
 imshow(grey_Reg);
%handles.grey_image=grey_Reg;
 guidata(hObject,handles);



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

handles.c_pow=get(hObject,'String');
guidata(hObject,handles);

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


% --- Executes during object creation, after setting all properties.
function pow_CreateFcn(hObject, eventdata, handles)
% hObject    handle to pow (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 edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

handles.gamma_pow=get(hObject,'String')
%returns contents of edit14 as text
%        str2double(get(hObject,'String')) returns contents of edit14 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (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 button press in pushbutton15.
function pushbutton15_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
G=str2double(handles.gamma_pow);%Gamma =0.40
C=str2double(handles.c_pow);
a=handles.gray_image;
c=im2double(a(:,:,1));
[m n]=size(c);
for i=1:1:m
    for j=1:1:n
        b(i,j)=C*(c(i,j)^G);
    end
```

```
801 end
802 %display image range [0 255]
803 axes(handles.axes7);
804 handles.gray_image=b;
805 imshow(handles.gray_image);
806 guidata(hObject,handles);
807
808
809
810 function edit16_Callback(hObject, eventdata, handles)
811 % hObject    handle to edit16 (see GCBO)
812 % eventdata  reserved - to be defined in a future version of MATLAB
813 % handles    structure with handles and user data (see GUIDATA)
814
815 handles.reg_Low=get(hObject,'String')
816 guidata(hObject,handles);
817
818 %returns contents of edit16 as text
819 %        str2double(get(hObject,'String')) returns contents of edit16 as a double
820
821
822 % --- Executes during object creation, after setting all properties.
823 function edit16_CreateFcn(hObject, eventdata, handles)
824 % hObject    handle to edit16 (see GCBO)
825 % eventdata  reserved - to be defined in a future version of MATLAB
826 % handles    empty - handles not created until after all CreateFcns called
827
828 % Hint: edit controls usually have a white background on Windows.
829 %       See ISPC and COMPUTER.
830 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
831     set(hObject,'BackgroundColor','white');
832 end
833
834
835
836 function edit18_Callback(hObject, eventdata, handles)
837 % hObject    handle to edit18 (see GCBO)
838 % eventdata  reserved - to be defined in a future version of MATLAB
839 % handles    structure with handles and user data (see GUIDATA)
840
841 % Hints: get(hObject,'String') returns contents of edit18 as text
842 %        str2double(get(hObject,'String')) returns contents of edit18 as a double
843
844 handles.reg_Hi=get(hObject,'String')
845 guidata(hObject,handles);
846
847 % --- Executes during object creation, after setting all properties.
848 function edit18_CreateFcn(hObject, eventdata, handles)
849 % hObject    handle to edit18 (see GCBO)
850 % eventdata  reserved - to be defined in a future version of MATLAB
851 % handles    empty - handles not created until after all CreateFcns called
852
853 % Hint: edit controls usually have a white background on Windows.
854 %       See ISPC and COMPUTER.
855 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
856     set(hObject,'BackgroundColor','white');
857 end
858
859
860
861 function edit19_Callback(hObject, eventdata, handles)
862 % hObject    handle to edit19 (see GCBO)
863 % eventdata  reserved - to be defined in a future version of MATLAB
864 % handles    structure with handles and user data (see GUIDATA)
865
866 handles.gamma_pre=get(hObject,'String')
867 %returns contents of edit19 as text
868 %        str2double(get(hObject,'String')) returns contents of edit19 as a double
869 guidata(hObject,handles);
870
871
872 % --- Executes during object creation, after setting all properties.
873 function edit19_CreateFcn(hObject, eventdata, handles)
874 % hObject    handle to edit19 (see GCBO)
875 % eventdata  reserved - to be defined in a future version of MATLAB
876 % handles    empty - handles not created until after all CreateFcns called
877
878 % Hint: edit controls usually have a white background on Windows.
879 %       See ISPC and COMPUTER.
880 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
881     set(hObject,'BackgroundColor','white');
882 end
883
884
885
886 function edit20_Callback(hObject, eventdata, handles)
887 % hObject    handle to edit20 (see GCBO)
888 % eventdata  reserved - to be defined in a future version of MATLAB
889 % handles    structure with handles and user data (see GUIDATA)
890 handles.pre_value=get(hObject,'String');
891 %returns contents of edit20 as text
892 %        str2double(get(hObject,'String')) returns contents of edit20 as a double
893 guidata(hObject,handles);
894
895
896 % --- Executes during object creation, after setting all properties.
897 function edit20_CreateFcn(hObject, eventdata, handles)
898 % hObject    handle to edit20 (see GCBO)
899 % eventdata  reserved - to be defined in a future version of MATLAB
900 % handles    empty - handles not created until after all CreateFcns called
901
```

```
% 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 button press in pushbutton16.
function pushbutton16_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
hist_En=handles.gray_image;
enhanc= histeq(hist_En);
axes(handles.axes3);
[pixelCount grayLevels]=imhist(enhanc);
%subplot(1, 2, 2);
bar(pixelCount);
title('');
set(gca,'YScale','log')
xlim([0 grayLevels(end)+20]); % Scale x axis manually.
yRange = ylim;
Im_mean = mean2(enhanc);
Im_std = std2(enhanc);
set(handles.edit2,'string',Im_mean);
set(handles.edit3,'string',Im_std);
axes(handles.axes7);
imshow(enhanc);


% --- Executes during object creation, after setting all properties.
function axes3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate axes3



function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

handles.low_average= get(hObject,'String');
guidata(hObject,handles);
%returns contents of edit21 as text
%        str2double(get(hObject,'String')) returns contents of edit21 as a double


% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (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 edit22_Callback(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.high_average= get(hObject,'String');
guidata(hObject,handles);
% Hints: get(hObject,'String') returns contents of edit22 as text
%        str2double(get(hObject,'String')) returns contents of edit22 as a double


% --- Executes during object creation, after setting all properties.
function edit22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit22 (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 edit23_Callback(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.low2_average= get(hObject,'String');
guidata(hObject,handles);
% Hints: get(hObject,'String') returns contents of edit23 as text
%        str2double(get(hObject,'String')) returns contents of edit23 as a double


% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to edit23 (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 edit24_Callback(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.radius_disk= get(hObject,'String');
guidata(hObject,handles);
% Hints: get(hObject,'String') returns contents of edit24 as text
%        str2double(get(hObject,'String')) returns contents of edit24 as a double


% --- Executes during object creation, after setting all properties.
function edit24_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit24 (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 button press in pushbutton17.
function pushbutton17_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
low=str2double(handles.low_average);
high=str2double(handles.high_average);
%smooth_image=handles.gray_image;
W=fspecial('average',[low  high]);
handles.gray_image=imfilter(handles.gray_image,W);
axes(handles.axes8);
imshow(handles.gray_image)
guidata(hObject,handles);


% --- Executes on button press in pushbutton18.
function pushbutton18_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
disk_r=str2double(handles.radius_disk);
smooth_image1=handles.gray_image;
W1=fspecial('disk',disk_r);
Filtered1=imfilter(smooth_image1,W1);
axes(handles.axes8);
handles.gray_image=Filtered1;
imshow(handles.gray_image)
guidata(hObject,handles);


% --- Executes on button press in pushbutton19.
function pushbutton19_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
dev=str2double(handles.gaus_dev);
low2=str2double(handles.low2_average);
high2=str2double(handles.high2_average);
smooth_image2=handles.gray_image;
W2=fspecial('gaussian',[low2  high2],dev);
Filtered2=imfilter(smooth_image2,W2);
axes(handles.axes8);
handles.gray_image=Filtered2;
imshow(handles.gray_image)
guidata(hObject,handles);



function edit25_Callback(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.high2_average= get(hObject,'String');
guidata(hObject,handles);
% Hints: get(hObject,'String') returns contents of edit25 as text
%        str2double(get(hObject,'String')) returns contents of edit25 as a double


% --- Executes during object creation, after setting all properties.
function edit25_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit25 (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.
```

```
1104 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1105     set(hObject,'BackgroundColor','white');
1106 end
1107
1108
1109
1110 function edit26_Callback(hObject, eventdata, handles)
1111 % hObject    handle to edit26 (see GCBO)
1112 % eventdata  reserved - to be defined in a future version of MATLAB
1113 % handles    structure with handles and user data (see GUIDATA)
1114 handles.gaus_dev= get(hObject,'String');
1115 guidata(hObject,handles);
1116 % Hints: get(hObject,'String') returns contents of edit26 as text
1117 %        str2double(get(hObject,'String')) returns contents of edit26 as a double
1118
1119
1120 % --- Executes during object creation, after setting all properties.
1121 function edit26_CreateFcn(hObject, eventdata, handles)
1122 % hObject    handle to edit26 (see GCBO)
1123 % eventdata  reserved - to be defined in a future version of MATLAB
1124 % handles    empty - handles not created until after all CreateFcns called
1125
1126 % Hint: edit controls usually have a white background on Windows.
1127 %       See ISPC and COMPUTER.
1128 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1129     set(hObject,'BackgroundColor','white');
1130 end
1131
1132
1133 % --- Executes on button press in pushbutton20.
1134 function pushbutton20_Callback(hObject, eventdata, handles)
1135 % hObject    handle to pushbutton20 (see GCBO)
1136 % eventdata  reserved - to be defined in a future version of MATLAB
1137 % handles    structure with handles and user data (see GUIDATA)
1138 pixel_motion=str2double(handles.pixel_m);
1139 alpha_filter=str2double(handles.alpha);
1140 smooth_image3=handles.gray_image;
1141 W3=fspecial('motion',pixel_motion,alpha_filter);
1142 Filtered3=imfilter(smooth_image3,W3);
1143 axes(handles.axes8);
1144 handles.gray_image=Filtered3;
1145 imshow(handles.gray_image)
1146 guidata(hObject,handles);
1147
1148
1149 function edit27_Callback(hObject, eventdata, handles)
1150 % hObject    handle to edit27 (see GCBO)
1151 % eventdata  reserved - to be defined in a future version of MATLAB
1152 % handles    structure with handles and user data (see GUIDATA)
1153 handles.alpha=get(hObject,'String');
1154 guidata(hObject,handles);
1155 %returns contents of edit27 as text
1156 %        str2double(get(hObject,'String')) returns contents of edit27 as a double
1157
1158
1159 % --- Executes during object creation, after setting all properties.
1160 function edit27_CreateFcn(hObject, eventdata, handles)
1161 % hObject    handle to edit27 (see GCBO)
1162 % eventdata  reserved - to be defined in a future version of MATLAB
1163 % handles    empty - handles not created until after all CreateFcns called
1164
1165 % Hint: edit controls usually have a white background on Windows.
1166 %       See ISPC and COMPUTER.
1167 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1168     set(hObject,'BackgroundColor','white');
1169 end
1170
1171
1172
1173 function edit28_Callback(hObject, eventdata, handles)
1174 % hObject    handle to edit28 (see GCBO)
1175 % eventdata  reserved - to be defined in a future version of MATLAB
1176 % handles    structure with handles and user data (see GUIDATA)
1177 handles.pixel_m=get(hObject,'String');
1178 guidata(hObject,handles);
1179
1180 % Hints: get(hObject,'String') returns contents of edit28 as text
1181 %        str2double(get(hObject,'String')) returns contents of edit28 as a double
1182
1183
1184 % --- Executes during object creation, after setting all properties.
1185 function edit28_CreateFcn(hObject, eventdata, handles)
1186 % hObject    handle to edit28 (see GCBO)
1187 % eventdata  reserved - to be defined in a future version of MATLAB
1188 % handles    empty - handles not created until after all CreateFcns called
1189
1190 % Hint: edit controls usually have a white background on Windows.
1191 %       See ISPC and COMPUTER.
1192 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1193     set(hObject,'BackgroundColor','white');
1194 end
1195
1196
1197 % --- Executes on button press in pushbutton21.
1198 function pushbutton21_Callback(hObject, eventdata, handles)
1199 % hObject    handle to pushbutton21 (see GCBO)
1200 % eventdata  reserved - to be defined in a future version of MATLAB
1201 % handles    structure with handles and user data (see GUIDATA)
1202 alpha_filter4=str2double(handles.alpha4);
1203 smooth_image4=handles.gray_image;
1204 W4=fspecial('unsharp',alpha_filter4);
```

```
Filtered4=imfilter(smooth_image4,W4)+handles.gray_image;
axes(handles.axes8);
handles.gray_image=Filtered4;
imshow(Filtered4)
guidata(hObject,handles);



function edit29_Callback(hObject, eventdata, handles)
% hObject    handle to edit29 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.alpha4=get(hObject,'String');
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit29 as text
%        str2double(get(hObject,'String')) returns contents of edit29 as a double


% --- Executes during object creation, after setting all properties.
function edit29_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit29 (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 edit30_Callback(hObject, eventdata, handles)
% hObject    handle to edit30 (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 edit30 as text
%        str2double(get(hObject,'String')) returns contents of edit30 as a double
handles.high5_average= get(hObject,'String');
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit30_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit30 (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 edit31_Callback(hObject, eventdata, handles)
% hObject    handle to edit31 (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 edit31 as text
%        str2double(get(hObject,'String')) returns contents of edit31 as a double
handles.low5_average= get(hObject,'String');
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit31_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit31 (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 edit32_Callback(hObject, eventdata, handles)
% hObject    handle to edit32 (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 edit32 as text
%        str2double(get(hObject,'String')) returns contents of edit32 as a double
handles.gaus_dev5= get(hObject,'String');
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit32_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit32 (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 button press in pushbutton22.
function pushbutton22_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
dev5=str2double(handles.gaus_dev5);
low5=str2double(handles.low5_average);
high5=str2double(handles.high5_average);
smooth_image5=handles.gray_image;
W5=fspecial('gaussian',[low5  high5],dev5);
Filtered5=imfilter(smooth_image5,W5)+smooth_image5;
axes(handles.axes8);
handles.gray_image=Filtered5;
imshow(handles.gray_image)
guidata(hObject,handles);


% --- Executes when figure1 is resized.
function figure1_SizeChangedFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton30.
function pushbutton30_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton30 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
contents = get(hObject,'Value')
switch contents
    case 1
        do_none=1;
    case 2
        set(handles.uipanel28, 'Visible','on');
        set(handles.uipanel19, 'Visible','off');
        set(handles.uipanel25, 'Visible','off');
        set(handles.uipanel39, 'Visible','off');
    case 3
        set(handles.uipanel28, 'Visible','off');
        set(handles.uipanel19, 'Visible','on');
        set(handles.uipanel25, 'Visible','off');
        set(handles.uipanel39, 'Visible','off');
    case 4
        set(handles.uipanel28, 'Visible','off');
        set(handles.uipanel19, 'Visible','off');
        set(handles.uipanel25, 'Visible','on');
        set(handles.uipanel39, 'Visible','off');
    case 5
        set(handles.uipanel28, 'Visible','off');
        set(handles.uipanel19, 'Visible','off');
        set(handles.uipanel25, 'Visible','off');
        set(handles.uipanel39, 'Visible','on');
    otherwise
end
guidata(hObject,handles);
%Image_Processing_Project_2(handles);



% --- Executes during object creation, after setting all properties.
function pushbutton30_CreateFcn(hObject, eventdata, handles)
% hObject    handle to pushbutton30 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function im = IdealLowPass(im0,fc)
% fc is the circular cutoff frequency which is normalized to [0 1], that is,
% the highest radian frequency \pi of digital signals is mapped to 1.

[ir,ic,iz] = size(im0);
hr = (ir-1)/2;
hc = (ic-1)/2;
[x, y] = meshgrid(-hc:hc, -hr:hr);

mg = sqrt((x/hc).^2 + (y/hr).^2);
lp = double(mg <= fc);

IM = fftshift(fft2(double(im0)));
IP = zeros(size(IM));
for z = 1:iz
    IP(:,:,z) = IM(:,:,z) .* lp;
end
im = abs(ifft2(ifftshift(IP)));

% --- Executes on button press in pushbutton31.
function pushbutton31_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton31 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

FT=fft2(handles.gray_image);
FT_real=abs(FT);
Fs=fftshift(FT_real);
FsL=log(Fs);
%axes(handles.axes8);
%imshow(FsL,[]);
fourier=IdealLowPass(handles.gray_image,handles.Low_Ideal);
handles.gray_image=fourier;
```

```
axes(handles.axes8);
imshow(handles.gray_image,[])
guidata(hObject,handles);



function edit50_Callback(hObject, eventdata, handles)
% hObject    handle to edit50 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.Low_Ideal=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit50 as text
%        str2double(get(hObject,'String')) returns contents of edit50 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit50_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit50 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function res = blpf(im,thresh,n)

% inputs
% im is the fourier transform of the image
% thresh is the cutoff circle radius (1,2,3...)
% n is the order of the filter (1,2,3...)

%outputs
% res is the filtered image

[r,c]=size(im);
d0=thresh;

d=zeros(r,c);
h=zeros(r,c);

for i=1:r
    for j=1:c
     d(i,j)=  sqrt( (i-(r/2))^2 + (j-(c/2))^2);
    end
end

for i=1:r
    for j=1:c
      h(i,j)=  1 / (1+ (d(i,j)/d0)^(2*n) ) ;
    end
end


for i=1:r
    for j=1:c
    res(i,j)=(h(i,j))*im(i,j);

    end
end




% 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 button press in pushbutton32.
function pushbutton32_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton32 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% simple implementation of frequency domain filters
%read input image
dim=handles.gray_image;
cim=double(dim);
[r,c]=size(cim);

r1=2*r;
c1=2*c;

pim=zeros((r1),(c1));
kim=zeros((r1),(c1));

%padding
for i=1:r
    for j=1:c
   pim(i,j)=cim(i,j);
    end
end

%center the transform
for i=1:r
    for j=1:c
   kim(i,j)=pim(i,j)*((-1)^(i+j));
    end
end


%2D fft
```

```
fim=fft2(kim);

n=handles.butterorder_low; %order for butterworth filter
thresh=handles.butterradius_low; % cutoff radius in frequency domain for filters

% % function call for low pass filters
% him=glp(fim,thresh); % gaussian low pass filter
 him=blpf(fim,thresh,n); % butterworth low pass filter

% % function calls for high pass filters
% him=ghp(fim,thresh); % gaussian low pass filter
%him=bhp(fim,thresh,n);  %butterworth high pass filter

% % function call for high boost filtering
% him=hbg(fim,thresh);  % using gaussian high pass filter
%  him=hbb(fim,thresh,n);  % using butterworth high pass filter


%inverse 2D fft
 ifim=ifft2(him);

for i=1:r1
    for j=1:c1
   ifim(i,j)=ifim(i,j)*((-1)^(i+j));
    end
end


% removing the padding
for i=1:r
    for j=1:c
   rim(i,j)=ifim(i,j);
    end
end

% retaining the ral parts of the matrix
rim=real(rim);
rim=uint8(rim);
handles.gray_image=rim;
axes(handles.axes8);
imshow(handles.gray_image);
guidata(hObject,handles);



function edit51_Callback(hObject, eventdata, handles)
% hObject    handle to edit51 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
 handles.butterorder_low=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit51 as text
%        str2double(get(hObject,'String')) returns contents of edit51 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit51_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit51 (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 edit52_Callback(hObject, eventdata, handles)
% hObject    handle to edit52 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.butterradius_low=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit52 as text
%        str2double(get(hObject,'String')) returns contents of edit52 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit52_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit52 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function res = glp(im,thresh)

% inputs
% im is the fourier transform of the image
% thresh is the cutoff circle radius

%outputs
% res is the filtered image

[r,c]=size(im);
d0=thresh;

d=zeros(r,c);
h=zeros(r,c);

for i=1:r
    for j=1:c
```

```
      d(i,j)=  sqrt( (i-(r/2))^2 + (j-(c/2))^2);
    end
end

for i=1:r
    for j=1:c
      h(i,j)=exp ( -( (d(i,j)^2)/(2*(d0^2)) ) );
    end
end


for i=1:r
    for j=1:c
    res(i,j)=(h(i,j))*im(i,j);

    end
end
% 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




% 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 button press in pushbutton33.
function pushbutton33_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton33 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
dim=handles.gray_image;
cim=double(dim);
[r,c]=size(cim);

r1=2*r;
c1=2*c;

pim=zeros((r1),(c1));
kim=zeros((r1),(c1));

%padding
for i=1:r
    for j=1:c
   pim(i,j)=cim(i,j);
    end
end

%center the transform
for i=1:r
    for j=1:c
   kim(i,j)=pim(i,j)*((-1)^(i+j));
    end
end


%2D fft
fim=fft2(kim);

thresh=handles.gaussradius_low; % cutoff radius in frequency domain for filters

% % function call for low pass filters
him=glp(fim,thresh); % gaussian low pass filter
% him=blpf(fim,thresh,n); % butterworth low pass filter

% % function calls for high pass filters
%him=ghp(fim,thresh); % gaussian low pass filter
%him=bhp(fim,thresh,n);  %butterworth high pass filter

% % function call for high boost filtering
% him=hbg(fim,thresh);  % using gaussian high pass filter
%  him=hbb(fim,thresh,n);  % using butterworth high pass filter


%inverse 2D fft
 ifim=ifft2(him);

for i=1:r1
    for j=1:c1
   ifim(i,j)=ifim(i,j)*((-1)^(i+j));
    end
end


% removing the padding
for i=1:r
    for j=1:c
   rim(i,j)=ifim(i,j);
    end
end

% retaining the ral parts of the matrix
rim=real(rim);
rim=uint8(rim);
```

```
handles.gray_image=rim;
axes(handles.axes8);
imshow(handles.gray_image)
guidata(hObject,handles);



function edit53_Callback(hObject, eventdata, handles)
% hObject    handle to edit53 (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 edit53 as text
%        str2double(get(hObject,'String')) returns contents of edit53 as a double
handles.gaussradius_low= str2double(get(hObject,'String'));
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit53_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit53 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function im = IdealHiPass(im0,fc)
% fc is the circular cutoff frequency which is normalized to [0 1], that is,
% the highest radian frequency \pi of digital signals is mapped to 1.

[ir,ic,iz] = size(im0);
hr = (ir-1)/2;
hc = (ic-1)/2;
[x, y] = meshgrid(-hc:hc, -hr:hr);

mg = sqrt((x/hc).^2 + (y/hr).^2);
lp = double(mg >= fc);

IM = fftshift(fft2(double(im0)));
IP = zeros(size(IM));
for z = 1:iz
    IP(:,:,z) = IM(:,:,z) .* lp;
end
im = abs(ifft2(ifftshift(IP)));

% 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 button press in pushbutton34.
function pushbutton34_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton34 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
fourier=IdealHiPass(handles.gray_image,handles.Low_Ideal);
handles.gray_image=fourier;
axes(handles.axes8);
imshow(handles.gray_image,[])
guidata(hObject,handles);



function edit54_Callback(hObject, eventdata, handles)
% hObject    handle to edit54 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.Low_Ideal=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit54 as text
%        str2double(get(hObject,'String')) returns contents of edit54 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit54_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit54 (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 edit55_Callback(hObject, eventdata, handles)
% hObject    handle to edit55 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.gaussradius_Hi=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit55 as text
%        str2double(get(hObject,'String')) returns contents of edit55 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit55_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit55 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function res = ghp(im,thresh)
```

```
1811
1812 % inputs
1813 % im is the fourier transform of the image
1814 % thresh is the cutoff circle radius
1815
1816 %outputs
1817 % res is the filtered image
1818
1819 [r,c]=size(im);
1820 d0=thresh;
1821
1822 d=zeros(r,c);
1823 h=zeros(r,c);
1824
1825 for i=1:r
1826     for j=1:c
1827      d(i,j)=  sqrt( (i-(r/2))^2 + (j-(c/2))^2);
1828     end
1829 end
1830
1831 for i=1:r
1832     for j=1:c
1833       h(i,j)=1- exp ( -( (d(i,j)^2)/(2*(d0^2)) ) );
1834     end
1835 end
1836
1837
1838 for i=1:r
1839     for j=1:c
1840     res(i,j)=(h(i,j))*im(i,j);
1841
1842     end
1843 end
1844 % Hint: edit controls usually have a white background on Windows.
1845 %       See ISPC and COMPUTER.
1846 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1847     set(hObject,'BackgroundColor','white');
1848 end
1849
1850
1851 % --- Executes on button press in pushbutton35.
1852 function pushbutton35_Callback(hObject, eventdata, handles)
1853 % hObject    handle to pushbutton35 (see GCBO)
1854 % eventdata  reserved - to be defined in a future version of MATLAB
1855 % handles    structure with handles and user data (see GUIDATA)
1856 dim=handles.gray_image;
1857 cim=double(dim);
1858 [r,c]=size(cim);
1859
1860 r1=2*r;
1861 c1=2*c;
1862
1863 pim=zeros((r1),(c1));
1864 kim=zeros((r1),(c1));
1865
1866 %padding
1867 for i=1:r
1868     for j=1:c
1869    pim(i,j)=cim(i,j);
1870     end
1871 end
1872
1873 %center the transform
1874 for i=1:r
1875     for j=1:c
1876    kim(i,j)=pim(i,j)*((-1)^(i+j));
1877     end
1878 end
1879
1880
1881 %2D fft
1882 fim=fft2(kim);
1883
1884 thresh=handles.gaussradius_Hi; % cutoff radius in frequency domain for filters
1885
1886 % % function call for low pass filters
1887 % him=glp(fim,thresh); % gaussian low pass filter
1888 % him=blpf(fim,thresh,n); % butterworth low pass filter
1889
1890 % % function calls for high pass filters
1891 him=ghp(fim,thresh); % gaussian low pass filter
1892 %him=bhp(fim,thresh,n);  %butterworth high pass filter
1893
1894 % % function call for high boost filtering
1895 % him=hbg(fim,thresh);  % using gaussian high pass filter
1896 %  him=hbb(fim,thresh,n);  % using butterworth high pass filter
1897
1898
1899 %inverse 2D fft
1900  ifim=ifft2(him);
1901
1902 for i=1:r1
1903     for j=1:c1
1904    ifim(i,j)=ifim(i,j)*((-1)^(i+j));
1905     end
1906 end
1907
1908
1909 % removing the padding
1910 for i=1:r
1911     for j=1:c
```

```
    rim(i,j)=ifim(i,j);
     end
end

% retaining the ral parts of the matrix
rim=real(rim);
rim=uint8(rim);
handles.gray_image=rim;
axes(handles.axes8);
imshow(handles.gray_image)
guidata(hObject,handles);



function edit56_Callback(hObject, eventdata, handles)
% hObject    handle to edit56 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.butter_order=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit56 as text
%        str2double(get(hObject,'String')) returns contents of edit56 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit56_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit56 (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 edit57_Callback(hObject, eventdata, handles)
% hObject    handle to edit57 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
 handles.butter_radius=str2double(get(hObject,'String'));
guidata(hObject,handles);

% Hints: get(hObject,'String') returns contents of edit57 as text
%        str2double(get(hObject,'String')) returns contents of edit57 as a double


% --- Executes during object creation, after setting all properties.
function edit57_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit57 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
function res = bhp(im,thresh,n)

% inputs
% im is the fourier transform of the image
% thresh is the cutoff circle radius

%outputs
% res is the filtered image

[r,c]=size(im);
d0=thresh;

d=zeros(r,c);
h=zeros(r,c);

for i=1:r
    for j=1:c
     d(i,j)=  sqrt( (i-(r/2))^2 + (j-(c/2))^2);
    end
end

for i=1:r
    for j=1:c
      h(i,j)=  1 / (1+ (d0/d(i,j))^(2*n) ) ;
    end
end


for i=1:r
    for j=1:c
    res(i,j)=(h(i,j))*im(i,j);

    end
end



% 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 button press in pushbutton36.
function pushbutton36_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton36 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
dim=handles.gray_image;
cim=double(dim);
[r,c]=size(cim);

r1=2*r;
c1=2*c;

pim=zeros((r1),(c1));
kim=zeros((r1),(c1));

%padding
for i=1:r
    for j=1:c
   pim(i,j)=cim(i,j);
    end
end

%center the transform
for i=1:r
    for j=1:c
   kim(i,j)=pim(i,j)*((-1)^(i+j));
    end
end


%2D fft
fim=fft2(kim);

thresh=handles.butter_radius; % cutoff radius in frequency domain for filters
n=handles.butter_order;
% % function call for low pass filters
% him=glp(fim,thresh); % gaussian low pass filter
% him=blpf(fim,thresh,n); % butterworth low pass filter

% % function calls for high pass filters
%him=ghp(fim,thresh); % gaussian low pass filter
him=bhp(fim,thresh,n);  %butterworth high pass filter

% % function call for high boost filtering
% him=hbg(fim,thresh);  % using gaussian high pass filter
%  him=hbb(fim,thresh,n);  % using butterworth high pass filter


%inverse 2D fft
 ifim=ifft2(him);

for i=1:r1
    for j=1:c1
   ifim(i,j)=ifim(i,j)*((-1)^(i+j));
    end
end


% removing the padding
for i=1:r
    for j=1:c
   rim(i,j)=ifim(i,j);
    end
end

% retaining the ral parts of the matrix
rim=real(rim);
rim=uint8(rim);
handles.gray_image=rim;
axes(handles.axes8);
imshow(handles.gray_image)
guidata(hObject,handles);


% --- Executes on button press in pushbutton37.
function pushbutton37_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton37 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

Kaverage = filter2(fspecial('average',handles.average_size),handles.gray_image)/255;
axes(handles.axes8);
handles.gray_image=Kaverage;
imshow(handles.gray_image)



function edit58_Callback(hObject, eventdata, handles)
% hObject    handle to edit58 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.average_size=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit58 as text
%        str2double(get(hObject,'String')) returns contents of edit58 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit58_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit58 (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 edit59_Callback(hObject, eventdata, handles)
% hObject    handle to edit59 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.med_size=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit59 as text
%        str2double(get(hObject,'String')) returns contents of edit59 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit59_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit59 (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 button press in pushbutton38.
function pushbutton38_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton38 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Median_noise = medfilt2(handles.gray_image, [handles.med_size handles.med_size]);
handles.gray_image=Median_noise;
axes(handles.axes8);
imshow(handles.gray_image)



function edit60_Callback(hObject, eventdata, handles)
% hObject    handle to edit60 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.wein_size=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit60 as text
%        str2double(get(hObject,'String')) returns contents of edit60 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit60_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit60 (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 button press in pushbutton39.
function pushbutton39_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton39 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Adaptive = wiener2(handles.gray_image,[handles.wein_size handles.wein_size]);
handles.gray_image=Adaptive;
axes(handles.axes8);
imshow(handles.gray_image)


% --- Executes on button press in pushbutton40.
function pushbutton40_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton40 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Start of PCA code,
%Data = imread('butterfly.jpg');
Data_gray = handles.gray_image;
Data_grayD = im2double(Data_gray);

Data_mean = mean(Data_grayD);
[a b] = size(Data_gray);
Data_meanNew = repmat(Data_mean,a,1);
DataAdjust = Data_grayD - Data_meanNew;
cov_data = cov(DataAdjust);
[V, D] = eig(cov_data);
V_trans = transpose(V);
DataAdjust_trans = transpose(DataAdjust);
FinalData = V_trans * DataAdjust_trans;
% End of PCA code
% Start of Inverse PCA code,
OriginalData_trans = inv(V_trans) * FinalData;
OriginalData = transpose(OriginalData_trans) + Data_meanNew;
% End of Inverse PCA code

```

```
% Image compression
PCs=handles.pca_dim;
PCs = b - PCs;
Reduced_V = V;
for i = 1:PCs,
Reduced_V(:,1) =[];
end
Y=Reduced_V'* DataAdjust_trans;
Compressed_Data=Reduced_V*Y;
handles.Compressed_Data = Compressed_Data' + Data_meanNew;
axes(handles.axes8);
imshow(handles.Compressed_Data)
guidata(hObject,handles);
% End of image compression



function edit61_Callback(hObject, eventdata, handles)
% hObject    handle to edit61 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.pca_dim=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit61 as text
%        str2double(get(hObject,'String')) returns contents of edit61 as a double
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function edit61_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit61 (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 button press in pushbutton41.
function pushbutton41_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton41 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
imwrite(handles.Compressed_Data,'output.jpg');



function edit62_Callback(hObject, eventdata, handles)
% hObject    handle to edit62 (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 edit62 as text
%        str2double(get(hObject,'String')) returns contents of edit62 as a double
InfoImage = imfinfo(handles.gray_image);
set(handles.edit62,'String',num2str(InfoImage.size));

% --- Executes during object creation, after setting all properties.
function edit62_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit62 (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 button press in pushbutton42.
function pushbutton42_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton42 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
newimage=imsubtract(handles.grey_1,handles.gray_image);
axes(handles.axes7);
handles.gray_image=newimage;
imshow(newimage)
guidata(hObject,handles);
```

References

[1] Digital Image Processing 2nd Ed. by Gonzalez, Woods, and Eddins.

[2] http://support.esri.com/technical-article/000005606