

# **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

<b>I. Abstract</b>	<b>2</b>
<b>II. INTRODUCTION</b>	<b>3</b>
<b>III. Functions</b>	<b>6</b>
A. Intensity Transformation . . . . .	6
1. Importing Image . . . . .	6
2. Decrease the greyscale level . . . . .	6
3. Resizing Image . . . . .	7
4. Intensity transformation . . . . .	8
B. Filtering in spatial domain . . . . .	13
1. Image smoothing . . . . .	13
2. Image sharpening . . . . .	13
C. Filtering in frequency domain . . . . .	14
D. Noise Reduction . . . . .	15
E. Image Compression . . . . .	17
<b>IV. Testing the functions</b>	<b>19</b>
<b>V. Conclusion</b>	<b>29</b>
<b>VI.Appendices</b>	<b>30</b>
A. Appendix A (code, the .m file for the project) . . . . .	30

## **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 medical 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 referred 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 processed image.

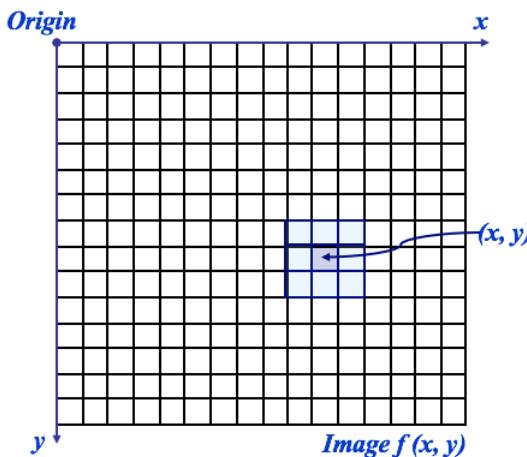


Figure 1: Digital representation of an image

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.

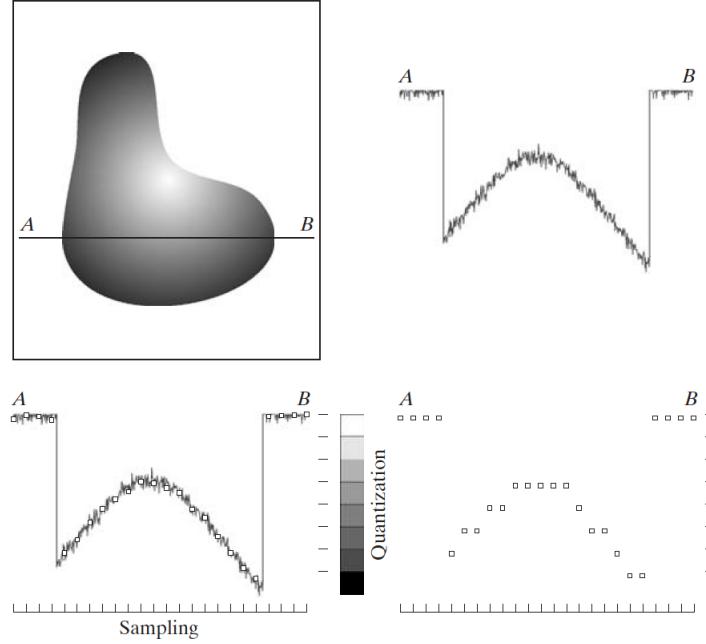


Figure 2: Digitization of an image using sensor scan from A to B. Figure from [1]

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:

$$g(x, y) = T[f(x, y)], \quad (1)$$

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

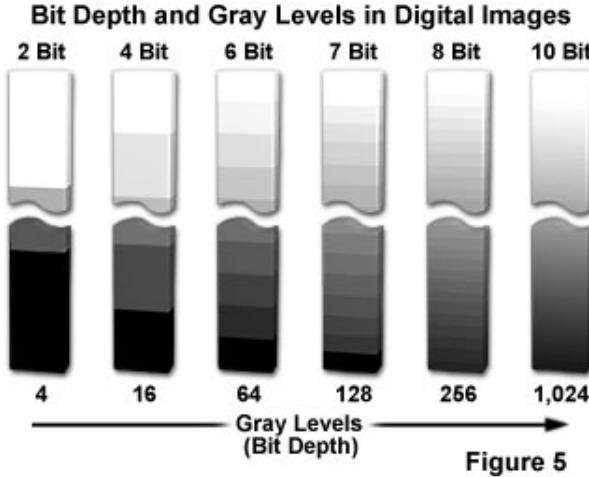


Figure 3: Different bits images

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](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.

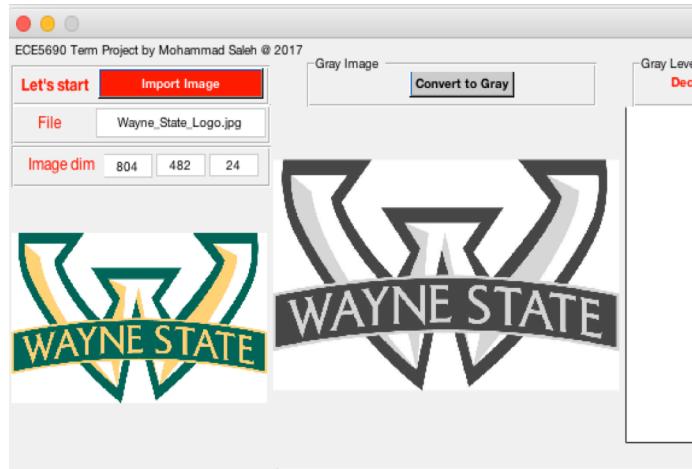


Figure 4: Snapshot of the main GUI for importing and converting the colored image to grayscale image

##### 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.

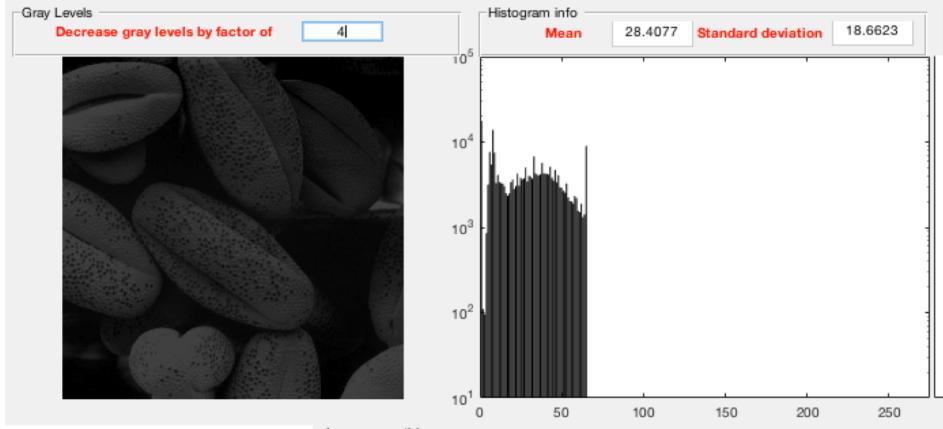


Figure 5: Snapshot of the main GUI for downgrading an 8-bit (0-255) image to 6-bit (0-63) image. Figure from [1].

### 3. Resizing Image

After we constrained how many bits the image is required, we will add a functionality for resizing 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 resizing. 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 resizing 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 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

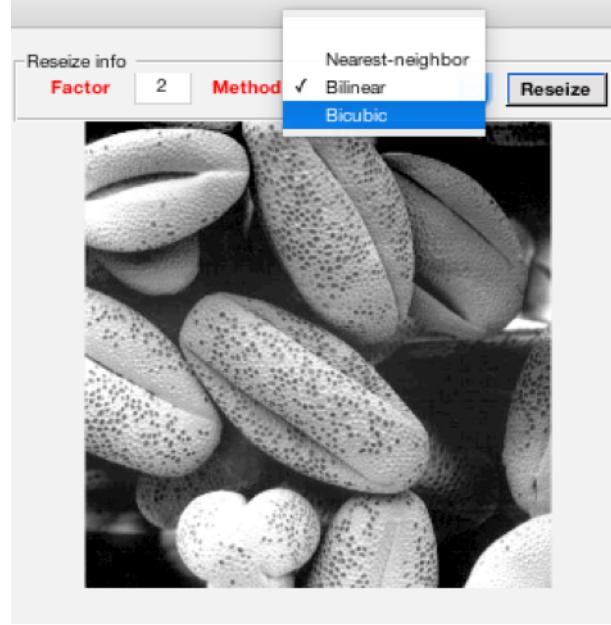


Figure 6: Snapshot of the main GUI resezing an image using different options. Figure from [1].

photographic. This transformation is given by the following expression.

$$s = L - 1 - r. \quad (2)$$

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:

$$s = c * \log(1 + r), \quad (3)$$

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:

$$s = c * r^\gamma, \quad (4)$$

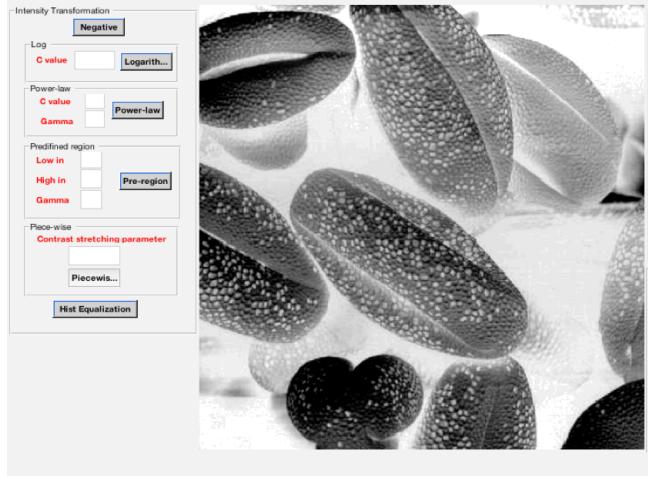


Figure 7: Negative of an image. Figure from [1].

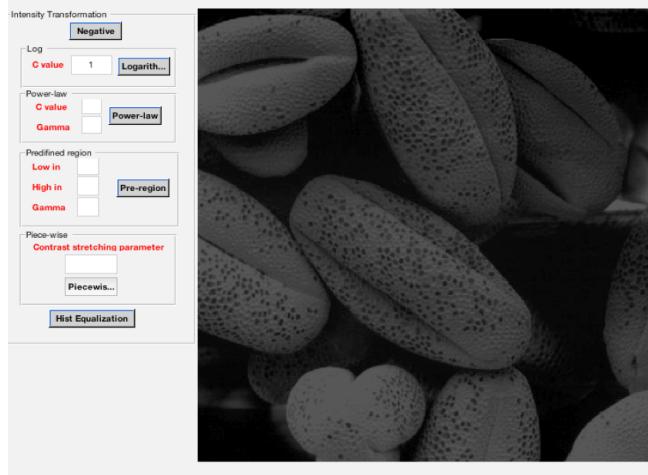


Figure 8: log transformation of an image. Figure from [1].

where  $c$  and  $\gamma$  are positive constants. It is more general than the log transformation where a family of possible transformation curve can be obtained by varying  $\gamma$ . 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  $\gamma$ . 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  $\gamma$ .

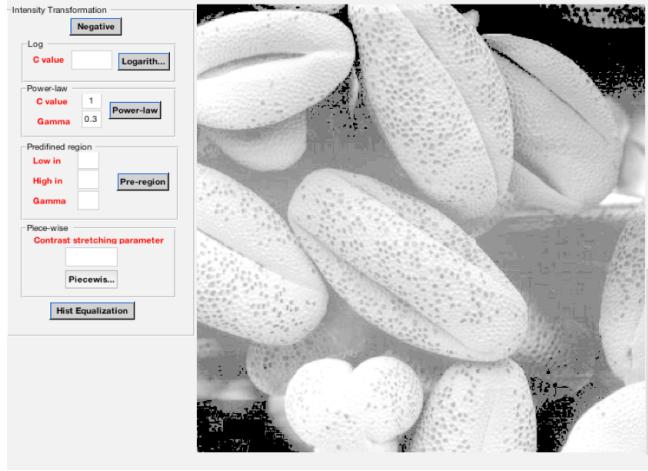


Figure 9: Power-law transformation of an image for  $c=1$  and  $\gamma=0.3$ . Figure from [1].

Figure 10 shows an example of this transformation for region 0-0.4 and  $\gamma$  value 0.8.

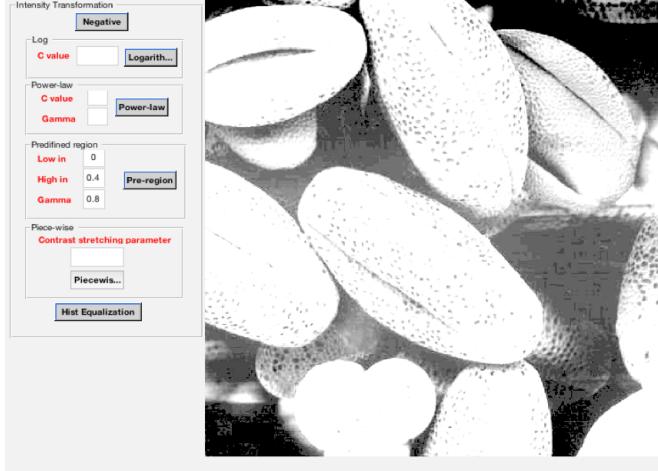


Figure 10: Pre-defined region transformation of an image for  $\gamma=0.8$ . Figure from [1].

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.

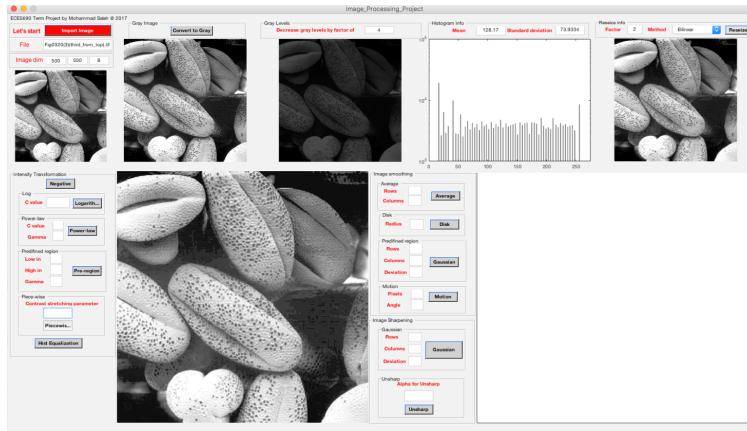


Figure 12: Histogram equalization transformation of an image with the updated histogram. Figure from [1].

## 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.

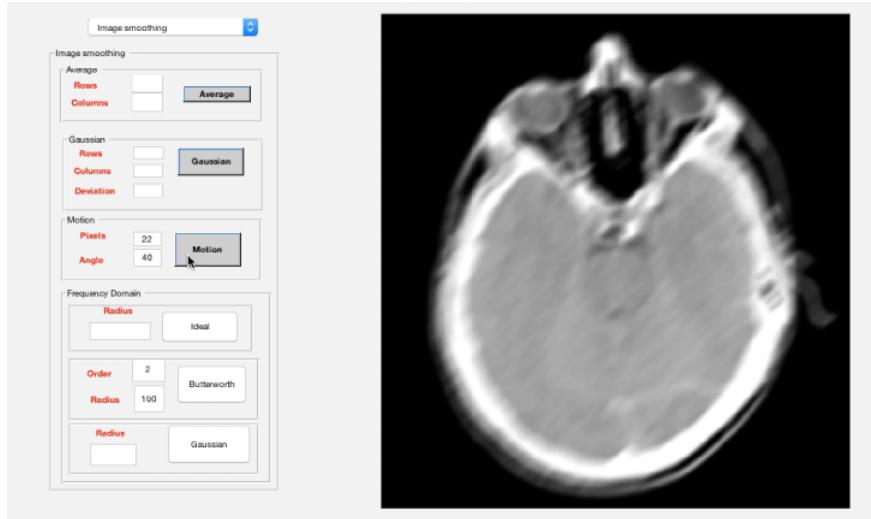


Figure 13: Motion transformation of an image. Figure from [1].

### 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

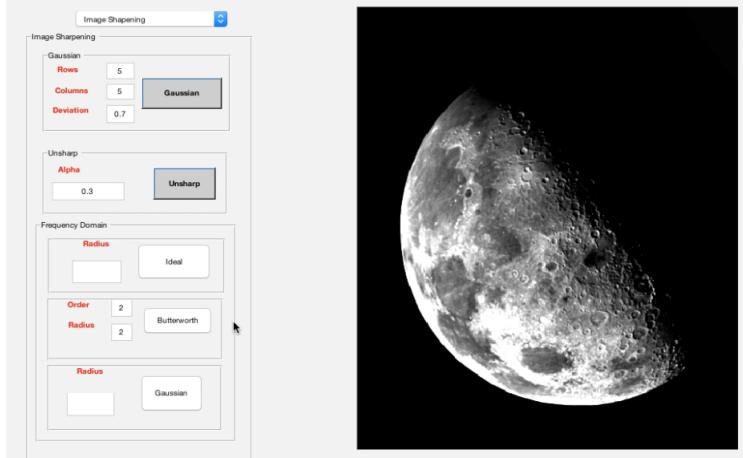


Figure 14: Snapshot of the GUI for the two sharpening functions in spatial domain.

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:

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

where  $\mathcal{F}^{-1}$  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:

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

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

Gaussian low pass filter is defined as:

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

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:

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

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

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

## 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

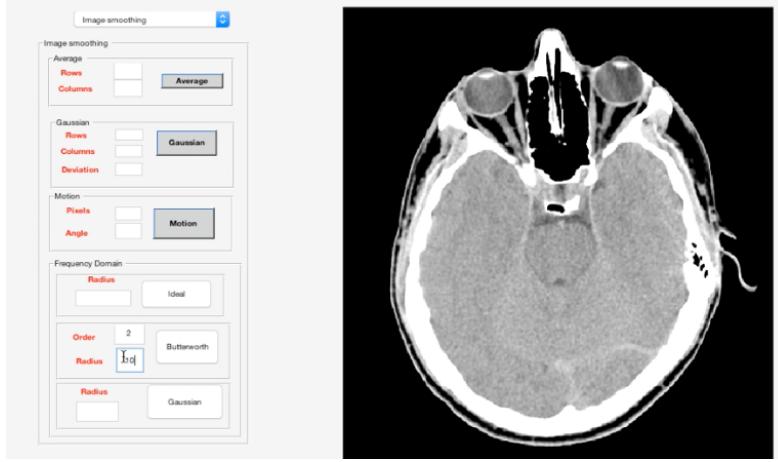


Figure 15: Image smoothening in frequency domain. Figure from [1].

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

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

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).

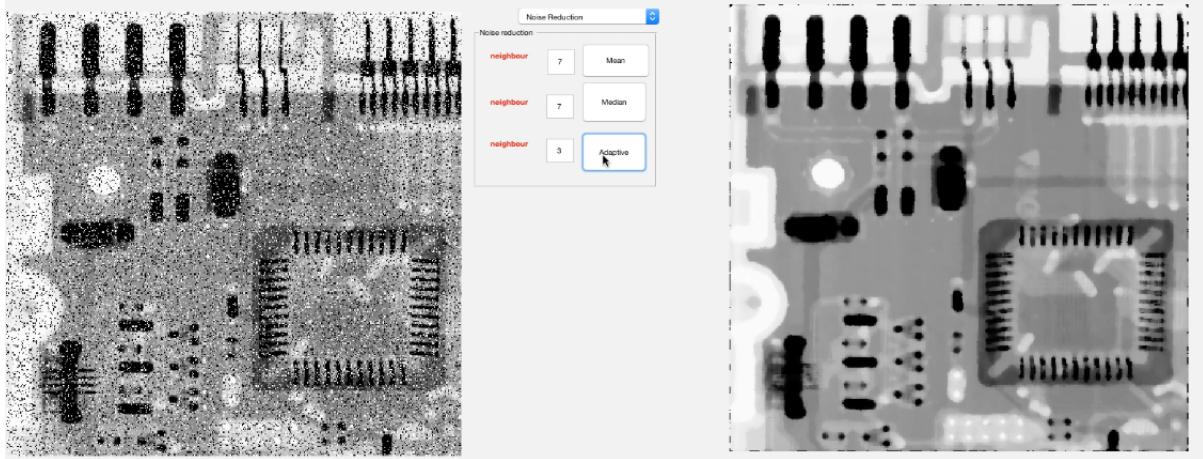


Figure 16: Noise reduction layout in the GUI. Figure from [1].

## 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.

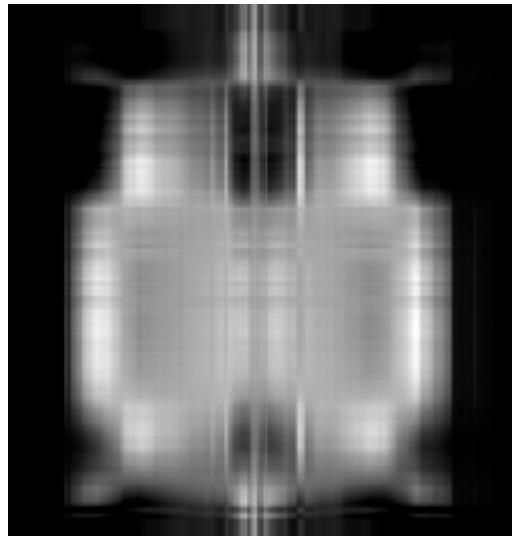


Figure 17: Compressed image by keeping the fist two components. Figure from [1].



Figure 18: Compressed image by keeping the first 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.

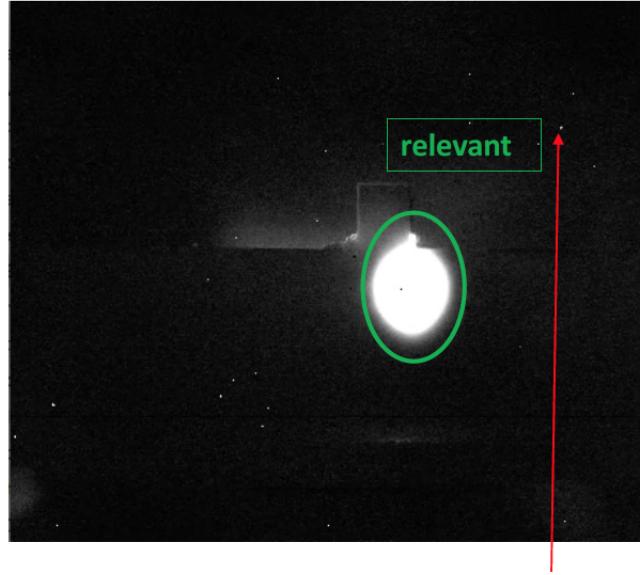


Figure 19: Image 1 of the images that need enhancement. Figure from [1].

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 ( $c=3, \gamma=7$ ) ==> 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 ( $c=5$ ).

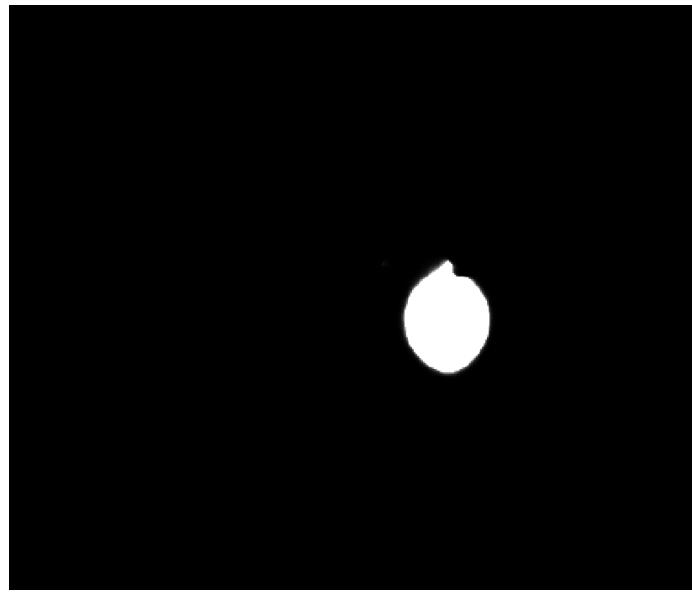


Figure 20: The enhancement of image 1. Figure from [1].

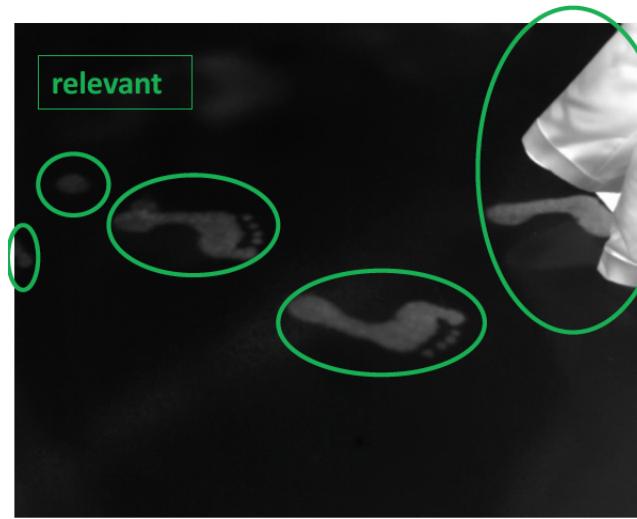


Figure 21: Image 2 of the images that need enhancement. Figure from [1].

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].

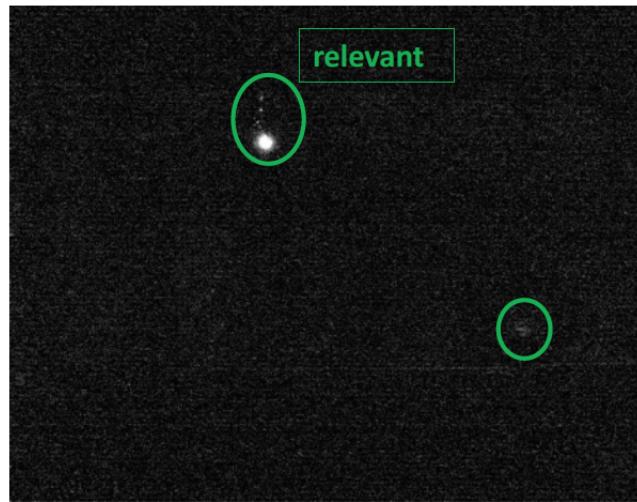


Figure 23: Image 3 of the images that need enhancement. Figure from [1].

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.

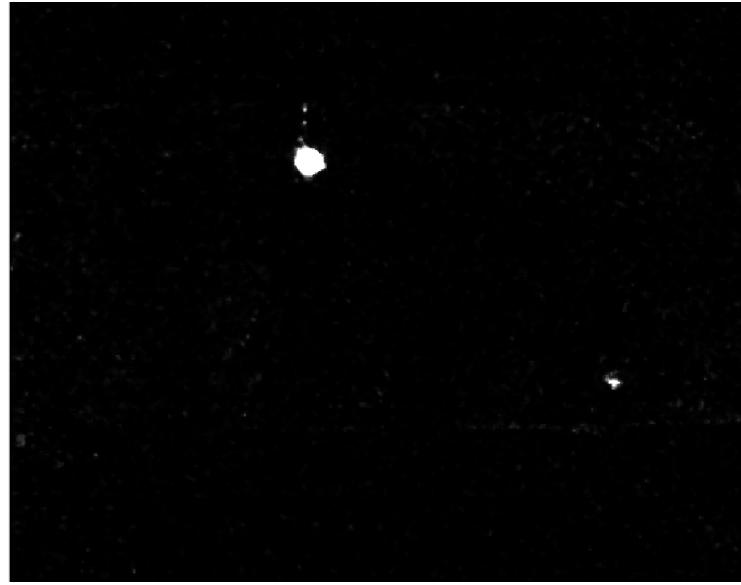


Figure 24: The enhancement of image 3. Figure from [1].

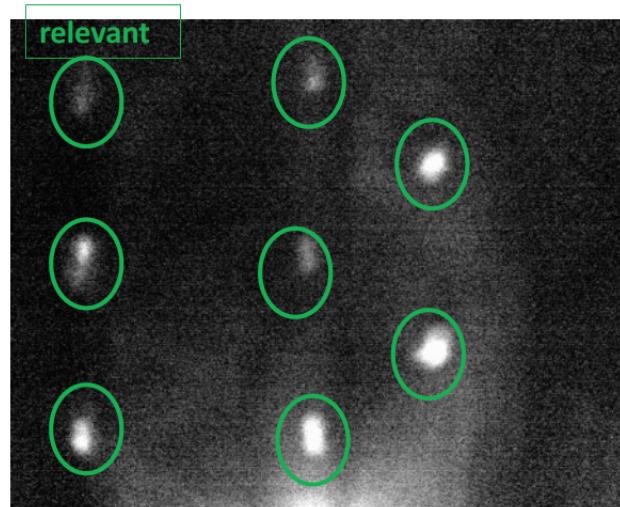


Figure 25: Image 4 of the images that need enhancement. Figure from [1].

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,  $\gamma=12$ ) ==> Median noise reduction (4x4 as input value for mask).

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

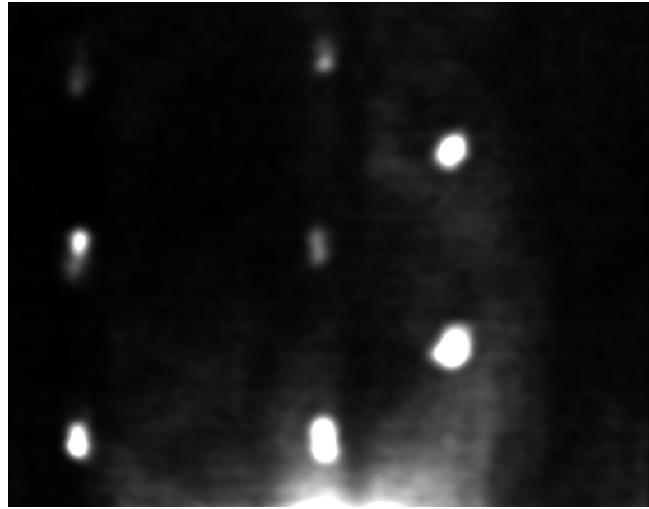


Figure 26: The enhancement of image 4. Figure from [1].

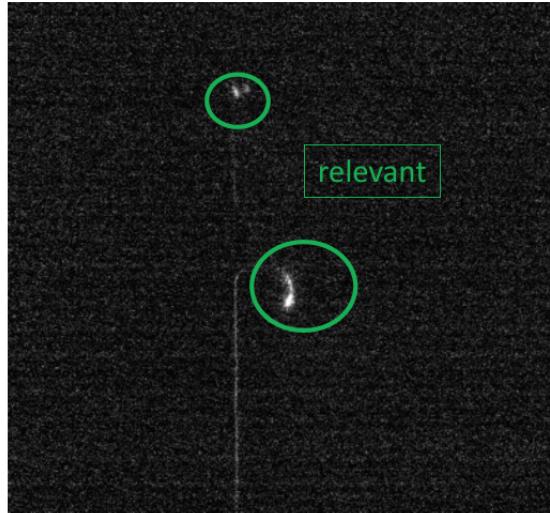


Figure 27: Image 5 of the images that need enhancement. Figure from [1].

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

Butterworth smoothing (order 2,  $D_0 = 60$ ) ==> Subtract the smoothed 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

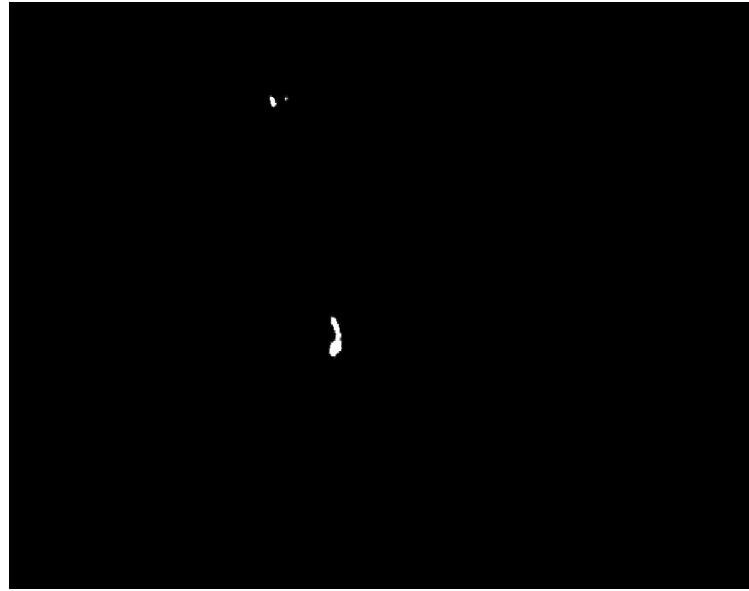


Figure 28: The enhancement of image 5. Figure from [1].

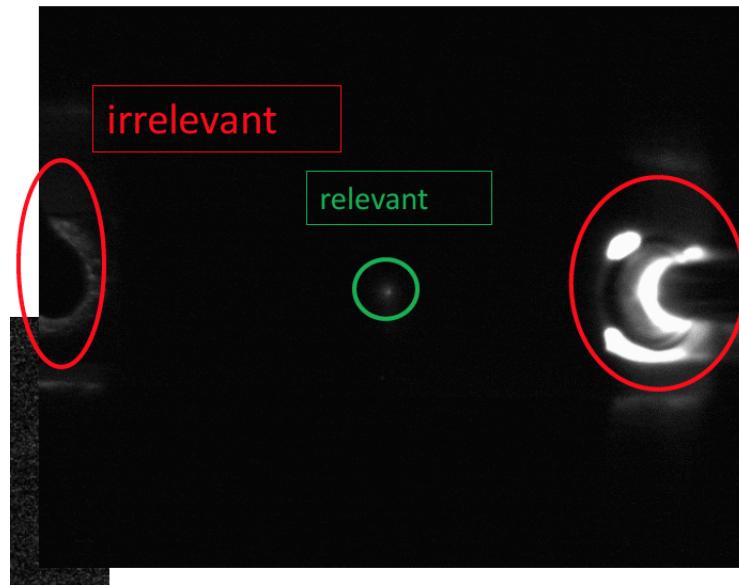


Figure 29: Image 6 of the images that need enhancement. Figure from [1].

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

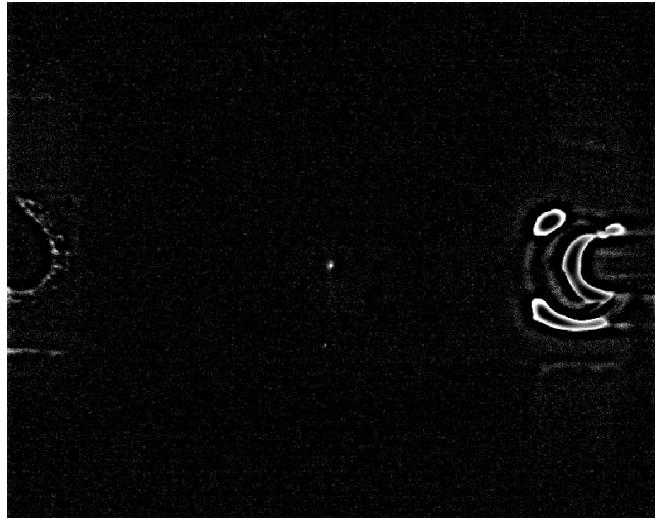


Figure 30: The enhancement of image 6. Figure from [1].

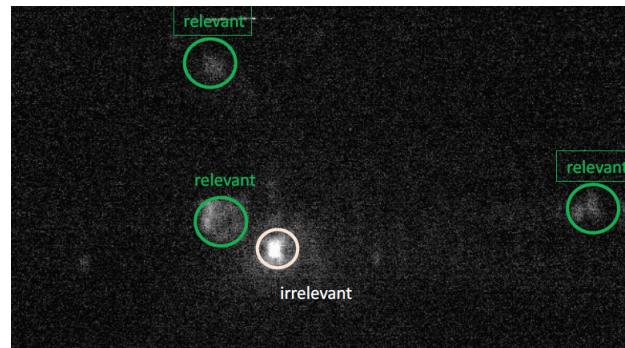


Figure 31: Image 7 of the images that need enhancement. Figure from [1].



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

noise were removed from 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.

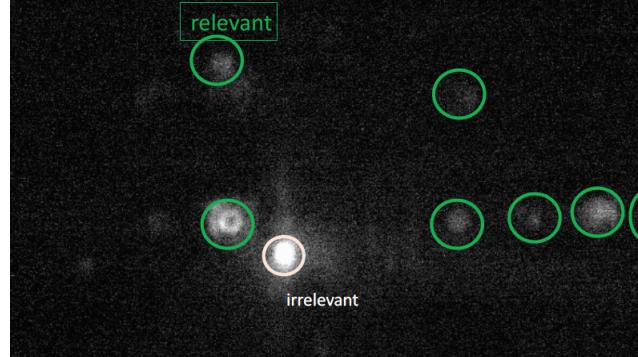


Figure 33: Image 8 of the images that need enhancement. Figure from [1].



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) .

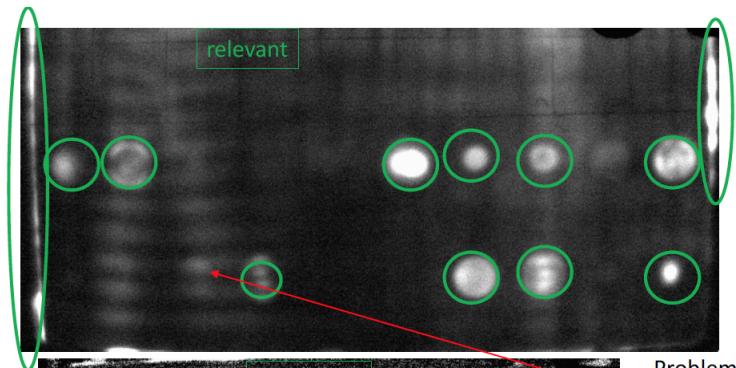


Figure 35: Image 9 of the images that need enhancement. Figure from [1].

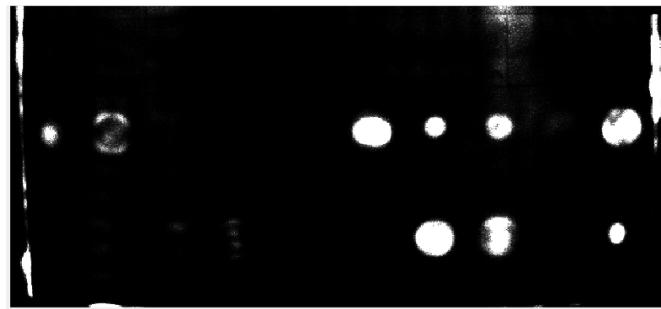


Figure 36: The enhancement of image 9. Figure from [1].

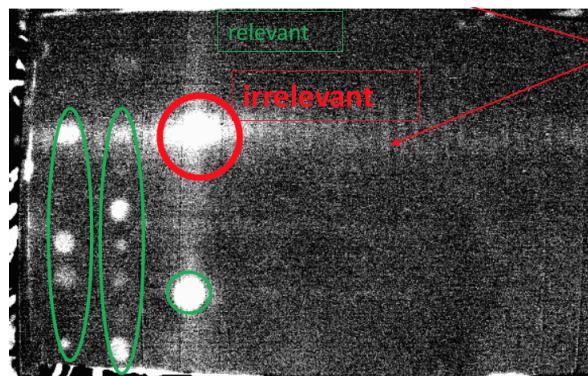


Figure 37: Image 10 of the images that need enhancement. Figure from [1].



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)

```
1 function varargout = Image_Processing_Project(varargin)
2 % IMAGE_PROCESSING_PROJECT MATLAB code for Image_Processing_Project.fig
3 %   IMAGE_PROCESSING_PROJECT, by itself, creates a new IMAGE_PROCESSING_PROJECT or raises the existing
4 %   singleton *.
5 %
6 %   H = IMAGE_PROCESSING_PROJECT returns the handle to a new IMAGE_PROCESSING_PROJECT or the handle to
7 %   the existing singleton*.
8 %
9 %   IMAGE_PROCESSING_PROJECT('CALLBACK', hObject, eventData, handles,...) calls the local
10 %   function named CALLBACK in IMAGE_PROCESSING_PROJECT.M with the given input arguments.
11 %
12 %   IMAGE_PROCESSING_PROJECT('Property','Value',...) creates a new IMAGE_PROCESSING_PROJECT or raises
13 %   the
14 %   existing singleton*. Starting from the left, property value pairs are
15 %   applied to the GUI before Image_Processing_Project_OpeningFcn gets called. An
16 %   unrecognized property name or invalid value makes property application
17 %   stop. All inputs are passed to Image_Processing_Project_OpeningFcn via varargin.
18 %
19 %   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
20 %   instance to run (singleton)".
21 %
22 % See also: GUIDE, GUIDATA, GUIHANDLES
23 %
24 % Edit the above text to modify the response to help Image_Processing_Project
25 % Last Modified by GUIDE v2.5 27-Apr-2017 07:16:00
26 %
27 % Begin initialization code - DO NOT EDIT
28 gui_Singleton = 1;
29 gui_State = struct('gui_Name',         mfilename, ...
30                     'gui_Singleton',    gui_Singleton, ...
31                     'gui_OpeningFcn',   @Image_Processing_Project_OpeningFcn, ...
32                     'gui_OutputFcn',    @Image_Processing_Project_OutputFcn, ...
33                     'gui_LayoutFcn',    [], ...
34                     'gui_Callback',     []);
35 if nargin && ischar(varargin{1})
36     gui_State.gui_Callback = str2func(varargin{1});
37 end
38
39 if nargout
40     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
41 else
42     gui_mainfcn(gui_State, varargin{:});
43 end
44 % End initialization code - DO NOT EDIT
45
46
47 % --- Executes just before Image_Processing_Project is made visible.
48 function Image_Processing_Project_OpeningFcn(hObject, eventdata, handles, varargin)
49 % This function has no output args, see OutputFcn.
50 % hObject    handle to figure
51 % eventdata   reserved - to be defined in a future version of MATLAB
52 % handles    structure with handles and user data (see GUIDATA)
53 % varargin   command line arguments to Image_Processing_Project (see VARARGIN)
54
55 % Choose default command line output for Image_Processing_Project
56 handles.output = hObject;
57
58 % Update handles structure
59 guidata(hObject, handles);
60
61 % UIWAIT makes Image_Processing_Project wait for user response (see UIRESUME)
62 % uiwait(handles.figure1);
63
64
65 % --- Outputs from this function are returned to the command line.
66 function varargout = Image_Processing_Project_OutputFcn(hObject, eventdata, handles)
67 % varargout  cell array for returning output args (see VARARGOUT);
68 % hObject    handle to figure
69 % eventdata   reserved - to be defined in a future version of MATLAB
70 % handles    structure with handles and user data (see GUIDATA)
71
72 % Get default command line output from handles structure
73 varargout{1} = handles.output;
74
75
76 % --- Executes on button press in pushbutton1.
77 function pushbutton1_Callback(hObject, eventdata, handles)
78 % hObject    handle to pushbutton1 (see GCBO)
79 % eventdata   reserved - to be defined in a future version of MATLAB
80 % handles    structure with handles and user data (see GUIDATA)
81 [filename pathname]= uigetfile({ '*.jpg'; '*.bmp'; '*.png'; '*.tif' },'File Selector');
82 handles.image_1= strcat(pathname, filename);
83 axes(handles.axes1)
84 imshow(handles.image_1)
85 InfoImage = imfinfo(handles.image_1);
86 set(handles.edit7,'String',num2str(InfoImage.Width));
87 set(handles.edit8,'String',num2str(InfoImage.Height));
88 set(handles.edit9,'String',num2str(InfoImage.BitDepth));
89 set(handles.Im.name,'string',filename);
90 image_2 = imread(handles.image_1);
91 if size(image_2,3)==1
92     handles.gray.image=image_2;
93 else
```

```

94     handles.gray.image=rgb2gray(image_2);
95 end
96 handles.grey_1=handles.gray.image;
97 guidata(hObject,handles);
98
99 % --- Executes during object creation, after setting all properties.
100 function pushbutton1_CreateFcn(hObject, eventdata, handles)
101 % hObject    handle to pushbutton1 (see GCBO)
102 % eventdata   reserved - to be defined in a future version of MATLAB
103 % handles    empty - handles not created until after all CreateFcns called
104
105
106 % --- If Enable == 'on', executes on mouse press in 5 pixel border.
107 % --- Otherwise, executes on mouse press in 5 pixel border or over text3.
108 function text3_ButtonDownFcn(hObject, eventdata, handles)
109 % hObject    handle to text3 (see GCBO)
110 % eventdata   reserved - to be defined in a future version of MATLAB
111 % handles    structure with handles and user data (see GUIDATA)
112 filename
113
114
115
116 function Im_name_Callback(hObject, eventdata, handles)
117 % hObject    handle to Im_name (see GCBO)
118 % eventdata   reserved - to be defined in a future version of MATLAB
119 % handles    structure with handles and user data (see GUIDATA)
120
121 % Hints: get(hObject,'String') returns contents of Im_name as text
122 %         str2double(get(hObject,'String')) returns contents of Im_name as a double
123
124
125 % --- Executes during object creation, after setting all properties.
126 function Im_name_CreateFcn(hObject, eventdata, handles)
127 % hObject    handle to Im_name (see GCBO)
128 % eventdata   reserved - to be defined in a future version of MATLAB
129 % handles    empty - handles not created until after all CreateFcns called
130
131 % Hint: edit controls usually have a white background on Windows.
132 %       See ISPC and COMPUTER.
133 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
134     set(hObject,'BackgroundColor','white');
135 end
136
137
138 % --- Executes on button press in pushbutton2.
139 function pushbutton2_Callback(hObject, eventdata, handles)
140 % hObject    handle to pushbutton2 (see GCBO)
141 % eventdata   reserved - to be defined in a future version of MATLAB
142 % handles    structure with handles and user data (see GUIDATA)
143 cam=webcam;
144 handles.image_1 = snapshot(cam);
145 closePreview(cam);
146 axes(handles.axes1);
147 imshow(handles.image_1);
148 handles.gray.image=rgb2gray(handles.image_1);
149 guidata(hObject,handles);
150
151 % --- Executes on button press in pushbutton3.
152 function pushbutton3_Callback(hObject, eventdata, handles)
153 % hObject    handle to pushbutton3 (see GCBO)
154 % eventdata   reserved - to be defined in a future version of MATLAB
155 % handles    structure with handles and user data (see GUIDATA)
156 image_2 = imread(handles.image_1);
157 if size(image_2,3)==1
158     handles.gray.image=image_2;
159 else
160     handles.gray.image=rgb2gray(image_2);
161 end
162 guidata(hObject,handles);
163 axes(handles.axes2);
164 imshow(handles.gray.image);
165 guidata(hObject,handles);
166
167
168 % --- Executes on button press in pushbutton4.
169 function pushbutton4_Callback(hObject, eventdata, handles)
170 % hObject    handle to pushbutton4 (see GCBO)
171 % eventdata   reserved - to be defined in a future version of MATLAB
172 % handles    structure with handles and user data (see GUIDATA)
173
174 guidata(hObject,handles);
175
176
177
178 function edit2_Callback(hObject, eventdata, handles)
179 % hObject    handle to edit2 (see GCBO)
180 % eventdata   reserved - to be defined in a future version of MATLAB
181 % handles    structure with handles and user data (see GUIDATA)
182
183 % Hints: get(hObject,'String') returns contents of edit2 as text
184 %         str2double(get(hObject,'String')) returns contents of edit2 as a double
185
186
187 % --- Executes during object creation, after setting all properties.
188 function edit2_CreateFcn(hObject, eventdata, handles)
189 % hObject    handle to edit2 (see GCBO)
190 % eventdata   reserved - to be defined in a future version of MATLAB
191 % handles    empty - handles not created until after all CreateFcns called
192
193 % Hint: edit controls usually have a white background on Windows.
194 %       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)

```

```

296 % eventdata reserved - to be defined in a future version of MATLAB
297 % handles empty - handles not created until after all CreateFcns called
298
299 % Hint: edit controls usually have a white background on Windows.
300 % See ISPC and COMPUTER.
301 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
302     set(hObject,'BackgroundColor','white');
303 end
304
305
306
307 function edit5_Callback(hObject, eventdata, handles)
308 % hObject handle to edit5 (see GCBO)
309 % eventdata reserved - to be defined in a future version of MATLAB
310 % handles structure with handles and user data (see GUIDATA)
311
312 % Hints: get(hObject,'String ') returns contents of edit5 as text
313 % str2double(get(hObject,'String')) returns contents of edit5 as a double
314 handles.resize.method=get(hObject,'String');
315 guidata(hObject,handles);
316
317 % --- Executes during object creation, after setting all properties.
318 function edit5_CreateFcn(hObject, eventdata, handles)
319 % hObject handle to edit5 (see GCBO)
320 % eventdata reserved - to be defined in a future version of MATLAB
321 % handles empty - handles not created until after all CreateFcns called
322
323 % Hint: edit controls usually have a white background on Windows.
324 % See ISPC and COMPUTER.
325 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
326     set(hObject,'BackgroundColor','white');
327 end
328
329
330 % --- Executes on button press in radiobutton1.
331 function radiobutton1_Callback(hObject, eventdata, handles)
332 % hObject handle to radiobutton1 (see GCBO)
333 % eventdata reserved - to be defined in a future version of MATLAB
334 % handles structure with handles and user data (see GUIDATA)
335
336 % Hint: get(hObject,'Value') returns toggle state of radiobutton1
337
338
339
340 function edit6_Callback(hObject, eventdata, handles)
341 % hObject handle to edit6 (see GCBO)
342 % eventdata reserved - to be defined in a future version of MATLAB
343 % handles structure with handles and user data (see GUIDATA)
344
345 % Hints: get(hObject,'String ') returns contents of edit6 as text
346 % str2double(get(hObject,'String')) returns contents of edit6 as a double
347 set(handles.edit6,'string',handles.resize.method);
348
349 % --- Executes during object creation, after setting all properties.
350 function edit6_CreateFcn(hObject, eventdata, handles)
351 % hObject handle to edit6 (see GCBO)
352 % eventdata reserved - to be defined in a future version of MATLAB
353 % handles empty - handles not created until after all CreateFcns called
354
355 % Hint: edit controls usually have a white background on Windows.
356 % See ISPC and COMPUTER.
357 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
358     set(hObject,'BackgroundColor','white');
359 end
360
361
362 % --- Executes on selection change in popupmenul.
363 function popupmenul_Callback(hObject, eventdata, handles)
364 % hObject handle to popupmenul (see GCBO)
365 % eventdata reserved - to be defined in a future version of MATLAB
366 % handles structure with handles and user data (see GUIDATA)
367 contents = get(hObject,'Value')
368 switch contents
369 case 1
370     do_none=1;
371 case 2
372     handles.resize.Method='nearest';
373 case 3
374     handles.resize.Method='bilinear';
375 case 4
376     handles.resize.Method='bicubic';
377 otherwise
378 end
379 guidata(hObject,handles);
380
381 % --- Executes during object creation, after setting all properties.
382 function popupmenul_CreateFcn(hObject, eventdata, handles)
383 % hObject handle to popupmenul (see GCBO)
384 % eventdata reserved - to be defined in a future version of MATLAB
385 % handles empty - handles not created until after all CreateFcns called
386
387 % Hint: popupmenu controls usually have a white background on Windows.
388 % See ISPC and COMPUTER.
389 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
390     set(hObject,'BackgroundColor','white');
391 end
392
393
394 % --- Executes on button press in radiobutton2.
395 function radiobutton2_Callback(hObject, eventdata, handles)
396 % hObject handle to radiobutton2 (see GCBO)

```

```

397 % eventdata reserved - to be defined in a future version of MATLAB
398 % handles structure with handles and user data (see GUIDATA)
399
400 % Hint: get(hObject,'Value') returns toggle state of radiobutton2
401
402
403
404 function edit7_Callback(hObject, eventdata, handles)
405 % hObject handle to edit7 (see GCBO)
406 % eventdata reserved - to be defined in a future version of MATLAB
407 % handles structure with handles and user data (see GUIDATA)
408
409 % Hints: get(hObject,'String') returns contents of edit7 as text
410 % str2double(get(hObject,'String')) returns contents of edit7 as a double
411 set(handles.edit7,'string',size(handles.image-1));
412 guidata(hObject,handles);
413
414
415 % ---- Executes during object creation, after setting all properties.
416 function edit7_CreateFcn(hObject, eventdata, handles)
417 % hObject handle to edit7 (see GCBO)
418 % eventdata reserved - to be defined in a future version of MATLAB
419 % handles empty - handles not created until after all CreateFcns called
420
421 % Hint: edit controls usually have a white background on Windows.
422 % See ISPC and COMPUTER.
423 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
424     set(hObject,'BackgroundColor','white');
425 end
426
427
428
429 function edit8_Callback(hObject, eventdata, handles)
430 % hObject handle to edit8 (see GCBO)
431 % eventdata reserved - to be defined in a future version of MATLAB
432 % handles structure with handles and user data (see GUIDATA)
433
434 % Hints: get(hObject,'String') returns contents of edit8 as text
435 % str2double(get(hObject,'String')) returns contents of edit8 as a double
436
437
438 % ---- Executes during object creation, after setting all properties.
439 function edit8_CreateFcn(hObject, eventdata, handles)
440 % hObject handle to edit8 (see GCBO)
441 % eventdata reserved - to be defined in a future version of MATLAB
442 % handles empty - handles not created until after all CreateFcns called
443
444 % Hint: edit controls usually have a white background on Windows.
445 % See ISPC and COMPUTER.
446 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
447     set(hObject,'BackgroundColor','white');
448 end
449
450
451 % ---- Executes on selection change in listbox1.
452 function listbox1_Callback(hObject, eventdata, handles)
453 % hObject handle to listbox1 (see GCBO)
454 % eventdata reserved - to be defined in a future version of MATLAB
455 % handles structure with handles and user data (see GUIDATA)
456
457 % Hints: contents = cellstr(get(hObject,'String')) returns listbox1 contents as cell array
458 % contents{get(hObject,'Value')} returns selected item from listbox1
459
460
461 % ---- Executes during object creation, after setting all properties.
462 function listbox1_CreateFcn(hObject, eventdata, handles)
463 % hObject handle to listbox1 (see GCBO)
464 % eventdata reserved - to be defined in a future version of MATLAB
465 % handles empty - handles not created until after all CreateFcns called
466
467 % Hint: listbox controls usually have a white background on Windows.
468 % See ISPC and COMPUTER.
469 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
470     set(hObject,'BackgroundColor','white');
471 end
472
473
474 % ---- Executes when uipanel5 is resized.
475 function uipanel5_SizeChangedFcn(hObject, eventdata, handles)
476 % hObject handle to uipanel5 (see GCBO)
477 % eventdata reserved - to be defined in a future version of MATLAB
478 % handles structure with handles and user data (see GUIDATA)
479
480
481
482 function edit9_Callback(hObject, eventdata, handles)
483 % hObject handle to edit9 (see GCBO)
484 % eventdata reserved - to be defined in a future version of MATLAB
485 % handles structure with handles and user data (see GUIDATA)
486
487 % Hints: get(hObject,'String') returns contents of edit9 as text
488 % str2double(get(hObject,'String')) returns contents of edit9 as a double
489
490
491 % ---- Executes during object creation, after setting all properties.
492 function edit9_CreateFcn(hObject, eventdata, handles)
493 % hObject handle to edit9 (see GCBO)
494 % eventdata reserved - to be defined in a future version of MATLAB
495 % handles empty - handles not created until after all CreateFcns called
496
497 % Hint: edit controls usually have a white background on Windows.

```

```

498 % See ISPC and COMPUTER.
499 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
500 set(hObject,'BackgroundColor','white');
501 end
502
503
504
505 function edit10_Callback(hObject, eventdata, handles)
506 % hObject    handle to edit10 (see GCBO)
507 % eventdata   reserved - to be defined in a future version of MATLAB
508 % handles    structure with handles and user data (see GUIDATA)
509
510 % Hints: get(hObject,'String') returns contents of edit10 as text
511 %         str2double(get(hObject,'String')) returns contents of edit10 as a double
512
513
514 % --- Executes during object creation, after setting all properties.
515 function edit10_CreateFcn(hObject, eventdata, handles)
516 % hObject    handle to edit10 (see GCBO)
517 % eventdata   reserved - to be defined in a future version of MATLAB
518 % handles    empty - handles not created until after all CreateFcns called
519
520 % Hint: edit controls usually have a white background on Windows.
521 % See ISPC and COMPUTER.
522 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
523 set(hObject,'BackgroundColor','white');
524 end
525
526
527 % --- Executes on selection change in popupmenu3.
528 function popupmenu3_Callback(hObject, eventdata, handles)
529 % hObject    handle to popupmenu3 (see GCBO)
530 % eventdata   reserved - to be defined in a future version of MATLAB
531 % handles    structure with handles and user data (see GUIDATA)
532
533 % Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3 contents as cell array
534 %         contents{get(hObject,'Value')} returns selected item from popupmenu3
535 bit_value = get(hObject,'Value')
536 switch bit_value
537     case 1
538         handles.bits=16;
539     case 2
540         handles.bits=8;
541     case 3
542         handles.bits=4;
543     otherwise
544 end
545 guidata(hObject,handles);
546
547 % --- Executes during object creation, after setting all properties.
548 function popupmenu3_CreateFcn(hObject, eventdata, handles)
549 % hObject    handle to popupmenu3 (see GCBO)
550 % eventdata   reserved - to be defined in a future version of MATLAB
551 % handles    empty - handles not created until after all CreateFcns called
552
553 % Hint: popupmenu controls usually have a white background on Windows.
554 % See ISPC and COMPUTER.
555 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
556 set(hObject,'BackgroundColor','white');
557 end
558
559
560
561 function edit11_Callback(hObject, eventdata, handles)
562 % hObject    handle to edit11 (see GCBO)
563 % eventdata   reserved - to be defined in a future version of MATLAB
564 % handles    structure with handles and user data (see GUIDATA)
565
566 handles.grayValues=get(hObject,'String');
567 %returns position of slider
568 handles.Grey_Quant=(handles.grayImage)/(str2double(handles.grayValues));
569 axes(handles.axes4);
570 imshow(handles.Grey_Quant);
571 guidata(hObject,handles);
572 axes(handles.axes3);
573 [pixelCount grayLevels]=imhist(handles.Grey_Quant);
574 %subplot(1, 2, 2);
575 bar(pixelCount);
576 title('');
577 set(gca,'YScale','log')
578 xlim([0 grayLevels(end)+20]); % Scale x axis manually.
579 yRange = ylim;
580 Im_mean = mean2(handles.Grey_Quant);
581 Im_std = std2(handles.Grey_Quant);
582 set(handles.edit2,'String',Im_mean);
583 set(handles.edit3,'String',Im_std);
584 maxGrayLevel = max(handles.Grey_Quant(:));
585 guidata(hObject,handles);
586
587 %returns contents of edit11 as text
588 %         str2double(get(hObject,'String')) returns contents of edit11 as a double
589
590
591 % --- Executes during object creation, after setting all properties.
592 function edit11_CreateFcn(hObject, eventdata, handles)
593 % hObject    handle to edit11 (see GCBO)
594 % eventdata   reserved - to be defined in a future version of MATLAB
595 % handles    empty - handles not created until after all CreateFcns called
596
597 % Hint: edit controls usually have a white background on Windows.
598 % 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)

```

```

700 % hObject    handle to piece (see GCBO)
701 % eventdata reserved - to be defined in a future version of MATLAB
702 % handles    structure with handles and user data (see GUIDATA)
703 g=im2double(handles.gray_image);
704 m=mean2(g);
705 E=str2double(handles.pre_value);
706 [M,N]=size(g);
707     for x = 1:M
708         for y = 1:N
709             gm=g(x,y);
710             z(x,y)=1/(1+(m/gm)^E);
711         end
712     end
713 axes(handles.axes7);
714 handles.gray_image=z;
715 imshow(handles.gray_image)
716 guidata(hObject,handles);
717
718 % Hint: get(hObject,'Value') returns toggle state of piece
719
720
721 % ---- Executes on button press in pre.
722 function pre_Callback(hObject, eventdata, handles)
723 % hObject    handle to pre (see GCBO)
724 % eventdata reserved - to be defined in a future version of MATLAB
725 % handles    structure with handles and user data (see GUIDATA)
726 gm=str2double(handles.gamma-pre);
727 low=str2double(handles.reg.Low);
728 high=str2double(handles.reg.Hi);
729 grey_Reg=imadjust(handles.gray_image,[low high],[0 1],gm);
730 axes(handles.axes7);
731 imshow(grey_Reg);
732 %handles.grey_image=grey_Reg;
733 guidata(hObject,handles);
734
735
736
737 function pow_Callback(hObject, eventdata, handles)
738 % hObject    handle to pow (see GCBO)
739 % eventdata reserved - to be defined in a future version of MATLAB
740 % handles    structure with handles and user data (see GUIDATA)
741
742 handles.c_pow=get(hObject,'String');
743 guidata(hObject,handles);
744
745 %returns contents of pow as text
746 %      str2double(get(hObject,'String')) returns contents of pow as a double
747
748
749 % ---- Executes during object creation, after setting all properties.
750 function pow_CreateFcn(hObject, eventdata, handles)
751 % hObject    handle to pow (see GCBO)
752 % eventdata reserved - to be defined in a future version of MATLAB
753 % handles    empty - handles not created until after all CreateFcns called
754
755 % Hint: edit controls usually have a white background on Windows.
756 % See ISPC and COMPUTER.
757 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
758     set(hObject,'BackgroundColor','white');
759 end
760
761
762
763 function edit14_Callback(hObject, eventdata, handles)
764 % hObject    handle to edit14 (see GCBO)
765 % eventdata reserved - to be defined in a future version of MATLAB
766 % handles    structure with handles and user data (see GUIDATA)
767
768 handles.gamma_pow=get(hObject,'String')
769 %returns contents of edit14 as text
770 %      str2double(get(hObject,'String')) returns contents of edit14 as a double
771 guidata(hObject,handles);
772
773
774 % ---- Executes during object creation, after setting all properties.
775 function edit14_CreateFcn(hObject, eventdata, handles)
776 % hObject    handle to edit14 (see GCBO)
777 % eventdata reserved - to be defined in a future version of MATLAB
778 % handles    empty - handles not created until after all CreateFcns called
779
780 % Hint: edit controls usually have a white background on Windows.
781 % See ISPC and COMPUTER.
782 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
783     set(hObject,'BackgroundColor','white');
784 end
785
786
787 % ---- Executes on button press in pushbutton15.
788 function pushbutton15_Callback(hObject, eventdata, handles)
789 % hObject    handle to pushbutton15 (see GCBO)
790 % eventdata reserved - to be defined in a future version of MATLAB
791 % handles    structure with handles and user data (see GUIDATA)
792 G=str2double(handles.gamma_pow);%Gamma =0.40
793 C=str2double(handles.c_pow);
794 a=handles.gray_image;
795 c=im2double(a(:,:,1));
796 [m n]=size(c);
797 for i=1:m
798     for j=1:n
799         b(i,j)=C*(c(i,j)^G);
800     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

```

```

902 % Hint: edit controls usually have a white background on Windows.
903 % See ISPC and COMPUTER.
904 if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
905 set(hObject, 'BackgroundColor', 'white');
906 end
907
908
909 % ---- Executes on button press in pushbutton16.
910 function pushbutton16_Callback(hObject, eventdata, handles)
911 % hObject handle to pushbutton16 (see GCBO)
912 % eventdata reserved - to be defined in a future version of MATLAB
913 % handles structure with handles and user data (see GUIDATA)
914 hist_En=handles.gray.image;
915 enhanc= histeq(hist_En);
916 axes(handles.axes3);
917 [pixelCount grayLevels]=imhist(enhanc);
918 %subplot(1, 2, 2);
919 bar(pixelCount);
920 title('');
921 set(gca, 'YScale', 'log')
922 xlim([0 grayLevels(end)+20]); % Scale x axis manually.
923 yRange = ylim;
924 Im_mean = mean2(enhanc);
925 Im_std = std2(enhanc);
926 set(handles.edit2, 'String', Im_mean);
927 set(handles.edit3, 'String', Im_std);
928 axes(handles.axes7);
929 imshow(enhanc);
930
931
932 % ---- Executes during object creation, after setting all properties.
933 function axes3_CreateFcn(hObject, eventdata, handles)
934 % hObject handle to axes3 (see GCBO)
935 % eventdata reserved - to be defined in a future version of MATLAB
936 % handles empty - handles not created until after all CreateFcns called
937
938 % Hint: place code in OpeningFcn to populate axes3
939
940
941
942 function edit21_Callback(hObject, eventdata, handles)
943 % hObject handle to edit21 (see GCBO)
944 % eventdata reserved - to be defined in a future version of MATLAB
945 % handles structure with handles and user data (see GUIDATA)
946
947 handles.low_average= get(hObject, 'String');
948 guidata(hObject, handles);
949 %returns contents of edit21 as text
950 % str2double(get(hObject, 'String')) returns contents of edit21 as a double
951
952
953 % ---- Executes during object creation, after setting all properties.
954 function edit21_CreateFcn(hObject, eventdata, handles)
955 % hObject handle to edit21 (see GCBO)
956 % eventdata reserved - to be defined in a future version of MATLAB
957 % handles empty - handles not created until after all CreateFcns called
958
959 % Hint: edit controls usually have a white background on Windows.
960 % See ISPC and COMPUTER.
961 if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
962 set(hObject, 'BackgroundColor', 'white');
963 end
964
965
966
967 function edit22_Callback(hObject, eventdata, handles)
968 % hObject handle to edit22 (see GCBO)
969 % eventdata reserved - to be defined in a future version of MATLAB
970 % handles structure with handles and user data (see GUIDATA)
971 handles.high_average= get(hObject, 'String');
972 guidata(hObject, handles);
973 % Hints: get(hObject, 'String') returns contents of edit22 as text
974 % str2double(get(hObject, 'String')) returns contents of edit22 as a double
975
976
977 % ---- Executes during object creation, after setting all properties.
978 function edit22_CreateFcn(hObject, eventdata, handles)
979 % hObject handle to edit22 (see GCBO)
980 % eventdata reserved - to be defined in a future version of MATLAB
981 % handles empty - handles not created until after all CreateFcns called
982
983 % Hint: edit controls usually have a white background on Windows.
984 % See ISPC and COMPUTER.
985 if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
986 set(hObject, 'BackgroundColor', 'white');
987 end
988
989
990
991 function edit23_Callback(hObject, eventdata, handles)
992 % hObject handle to edit23 (see GCBO)
993 % eventdata reserved - to be defined in a future version of MATLAB
994 % handles structure with handles and user data (see GUIDATA)
995 handles.low2_average= get(hObject, 'String');
996 guidata(hObject, handles);
997 % Hints: get(hObject, 'String') returns contents of edit23 as text
998 % str2double(get(hObject, 'String')) returns contents of edit23 as a double
999
1000
1001 % ---- Executes during object creation, after setting all properties.
1002 function edit23_CreateFcn(hObject, eventdata, handles)

```

```

1003 % hObject    handle to edit23 (see GCBO)
1004 % eventdata   reserved - to be defined in a future version of MATLAB
1005 % handles     empty - handles not created until after all CreateFcns called
1006
1007 % Hint: edit controls usually have a white background on Windows.
1008 %         See ISPC and COMPUTER.
1009 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1010     set(hObject,'BackgroundColor','white');
1011 end
1012
1013
1014
1015 function edit24_Callback(hObject, eventdata, handles)
1016 % hObject    handle to edit24 (see GCBO)
1017 % eventdata   reserved - to be defined in a future version of MATLAB
1018 % handles     structure with handles and user data (see GUIDATA)
1019 handles.radius.disk= get(hObject,'String');
1020 guidata(hObject,handles);
1021 % Hints: get(hObject,'String') returns contents of edit24 as text
1022 %         str2double(get(hObject,'String')) returns contents of edit24 as a double
1023
1024
1025 % ---- Executes during object creation, after setting all properties.
1026 function edit24_CreateFcn(hObject, eventdata, handles)
1027 % hObject    handle to edit24 (see GCBO)
1028 % eventdata   reserved - to be defined in a future version of MATLAB
1029 % handles     empty - handles not created until after all CreateFcns called
1030
1031 % Hint: edit controls usually have a white background on Windows.
1032 %         See ISPC and COMPUTER.
1033 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1034     set(hObject,'BackgroundColor','white');
1035 end
1036
1037
1038 % ---- Executes on button press in pushbutton17.
1039 function pushbutton17_Callback(hObject, eventdata, handles)
1040 % hObject    handle to Pushbutton17 (see GCBO)
1041 % eventdata   reserved - to be defined in a future version of MATLAB
1042 % handles     structure with handles and user data (see GUIDATA)
1043 low=str2double(handles.low_average);
1044 high=str2double(handles.high_average);
1045 %smooth_image=handles.gray_image;
1046 W=fspecial('average',[low high]);
1047 handles.gray.image=imfilter(handles.gray.image,W);
1048 axes(handles.axes8);
1049 imshow(handles.gray.image)
1050 guidata(hObject,handles);
1051
1052
1053 % ---- Executes on button press in pushbutton18.
1054 function pushbutton18_Callback(hObject, eventdata, handles)
1055 % hObject    handle to pushbutton18 (see GCBO)
1056 % eventdata   reserved - to be defined in a future version of MATLAB
1057 % handles     structure with handles and user data (see GUIDATA)
1058 disk_r=str2double(handles.radius_disk);
1059 smooth_image1=handles.gray.image;
1060 W1=fspecial('disk',disk_r);
1061 Filtered1=imfilter(smooth_image1,W1);
1062 axes(handles.axes8);
1063 handles.gray.image=Filtered1;
1064 imshow(handles.gray.image)
1065 guidata(hObject,handles);
1066
1067
1068 % ---- Executes on button press in pushbutton19.
1069 function pushbutton19_Callback(hObject, eventdata, handles)
1070 % hObject    handle to pushbutton19 (see GCBO)
1071 % eventdata   reserved - to be defined in a future version of MATLAB
1072 % handles     structure with handles and user data (see GUIDATA)
1073 dev=str2double(handles.gaus_dev);
1074 low2=str2double(handles.low2_average);
1075 high2=str2double(handles.high2_average);
1076 smooth_image2=handles.gray.image;
1077 W2=fspecial('gaussian',[low2 high2],dev);
1078 Filtered2=imfilter(smooth_image2,W2);
1079 axes(handles.axes8);
1080 handles.gray.image=Filtered2;
1081 imshow(handles.gray.image)
1082 guidata(hObject,handles);
1083
1084
1085
1086 function edit25_Callback(hObject, eventdata, handles)
1087 % hObject    handle to edit25 (see GCBO)
1088 % eventdata   reserved - to be defined in a future version of MATLAB
1089 % handles     structure with handles and user data (see GUIDATA)
1090 handles.high2.average= get(hObject,'String');
1091 guidata(hObject,handles);
1092 % Hints: get(hObject,'String') returns contents of edit25 as text
1093 %         str2double(get(hObject,'String')) returns contents of edit25 as a double
1094
1095
1096 % ---- Executes during object creation, after setting all properties.
1097 function edit25_CreateFcn(hObject, eventdata, handles)
1098 % hObject    handle to edit25 (see GCBO)
1099 % eventdata   reserved - to be defined in a future version of MATLAB
1100 % handles     empty - handles not created until after all CreateFcns called
1101
1102 % Hint: edit controls usually have a white background on Windows.
1103 %         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);

```

```

1205 Filtered4=imfilter(smooth_image4,W4)+handles.gray_image;
1206 axes(handles.axes8);
1207 handles.gray.image=Filtered4;
1208 imshow(Filtered4)
1209 guidata(hObject,handles);
1210
1211
1212 function edit29_Callback(hObject, eventdata, handles)
1213 % hObject    handle to edit29 (see GCBO)
1214 % eventdata reserved - to be defined in a future version of MATLAB
1215 % handles    structure with handles and user data (see GUIDATA)
1216 handles.alpha4=get(hObject,'String');
1217 guidata(hObject,handles);
1218
1219 % Hints: get(hObject,'String') returns contents of edit29 as text
1220 %         str2double(get(hObject,'String')) returns contents of edit29 as a double
1221
1222 % --- Executes during object creation, after setting all properties.
1223 function edit29_CreateFcn(hObject, eventdata, handles)
1224 % hObject    handle to edit29 (see GCBO)
1225 % eventdata reserved - to be defined in a future version of MATLAB
1226 % handles    empty - handles not created until after all CreateFcns called
1227
1228 % Hint: edit controls usually have a white background on Windows.
1229 % See ISPC and COMPUTER.
1230 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1231 set(hObject,'BackgroundColor','white');
1232 end
1233
1234
1235
1236
1237
1238 function edit30_Callback(hObject, eventdata, handles)
1239 % hObject    handle to edit30 (see GCBO)
1240 % eventdata reserved - to be defined in a future version of MATLAB
1241 % handles    structure with handles and user data (see GUIDATA)
1242
1243 % Hints: get(hObject,'String') returns contents of edit30 as text
1244 %         str2double(get(hObject,'String')) returns contents of edit30 as a double
1245 handles.high5.average= get(hObject,'String');
1246 guidata(hObject,handles);
1247
1248 % --- Executes during object creation, after setting all properties.
1249 function edit30_CreateFcn(hObject, eventdata, handles)
1250 % hObject    handle to edit30 (see GCBO)
1251 % eventdata reserved - to be defined in a future version of MATLAB
1252 % handles    empty - handles not created until after all CreateFcns called
1253
1254 % Hint: edit controls usually have a white background on Windows.
1255 % See ISPC and COMPUTER.
1256 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1257 set(hObject,'BackgroundColor','white');
1258 end
1259
1260
1261
1262 function edit31_Callback(hObject, eventdata, handles)
1263 % hObject    handle to edit31 (see GCBO)
1264 % eventdata reserved - to be defined in a future version of MATLAB
1265 % handles    structure with handles and user data (see GUIDATA)
1266
1267 % Hints: get(hObject,'String') returns contents of edit31 as text
1268 %         str2double(get(hObject,'String')) returns contents of edit31 as a double
1269 handles.low5.average= get(hObject,'String');
1270 guidata(hObject,handles);
1271
1272 % --- Executes during object creation, after setting all properties.
1273 function edit31_CreateFcn(hObject, eventdata, handles)
1274 % hObject    handle to edit31 (see GCBO)
1275 % eventdata reserved - to be defined in a future version of MATLAB
1276 % handles    empty - handles not created until after all CreateFcns called
1277
1278 % Hint: edit controls usually have a white background on Windows.
1279 % See ISPC and COMPUTER.
1280 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1281 set(hObject,'BackgroundColor','white');
1282 end
1283
1284
1285
1286 function edit32_Callback(hObject, eventdata, handles)
1287 % hObject    handle to edit32 (see GCBO)
1288 % eventdata reserved - to be defined in a future version of MATLAB
1289 % handles    structure with handles and user data (see GUIDATA)
1290
1291 % Hints: get(hObject,'String') returns contents of edit32 as text
1292 %         str2double(get(hObject,'String')) returns contents of edit32 as a double
1293 handles.gaus.dev5= get(hObject,'String');
1294 guidata(hObject,handles);
1295
1296 % --- Executes during object creation, after setting all properties.
1297 function edit32_CreateFcn(hObject, eventdata, handles)
1298 % hObject    handle to edit32 (see GCBO)
1299 % eventdata reserved - to be defined in a future version of MATLAB
1300 % handles    empty - handles not created until after all CreateFcns called
1301
1302 % Hint: edit controls usually have a white background on Windows.
1303 % See ISPC and COMPUTER.
1304 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1305 set(hObject,'BackgroundColor','white');

```

```

1306 end
1307
1308
1309 % ---- Executes on button press in pushbutton22.
1310 function pushbutton22_Callback(hObject, eventdata, handles)
1311 % hObject    handle to pushbutton22 (see GCBO)
1312 % eventdata reserved - to be defined in a future version of MATLAB
1313 % handles    structure with handles and user data (see GUIDATA)
1314 dev5=str2double(handles.gaus-dev5);
1315 low5=str2double(handles.low5-average);
1316 high5=str2double(handles.high5-average);
1317 smooth_image5=handles.gray.image;
1318 W5=fspecial('gaussian',[low5 high5],dev5);
1319 Filtered5=imfilter(smooth_image5,W5)+smooth_image5;
1320 axes(handles.axes8);
1321 handles.gray.image=Filtered5;
1322 imshow(handles.gray.image)
1323 guidata(hObject,handles);
1324
1325
1326 % ---- Executes when figure1 is resized.
1327 function figure1_SizeChangedFcn(hObject, eventdata, handles)
1328 % hObject    handle to figure1 (see GCBO)
1329 % eventdata reserved - to be defined in a future version of MATLAB
1330 % handles    structure with handles and user data (see GUIDATA)
1331
1332
1333 % ---- Executes on button press in pushbutton30.
1334 function pushbutton30_Callback(hObject, eventdata, handles)
1335 % hObject    handle to pushbutton30 (see GCBO)
1336 % eventdata reserved - to be defined in a future version of MATLAB
1337 % handles    structure with handles and user data (see GUIDATA)
1338 contents = get(hObject, 'Value')
1339 switch contents
1340     case 1
1341         do_none=1;
1342     case 2
1343         set(handles.uipanel28, 'Visible','on');
1344         set(handles.uipanel19, 'Visible','off');
1345         set(handles.uipanel25, 'Visible','off');
1346         set(handles.uipanel39, 'Visible','off');
1347     case 3
1348         set(handles.uipanel28, 'Visible','off');
1349         set(handles.uipanel19, 'Visible','on');
1350         set(handles.uipanel25, 'Visible','off');
1351         set(handles.uipanel39, 'Visible','off');
1352     case 4
1353         set(handles.uipanel28, 'Visible','off');
1354         set(handles.uipanel19, 'Visible','off');
1355         set(handles.uipanel25, 'Visible','on');
1356         set(handles.uipanel39, 'Visible','off');
1357     case 5
1358         set(handles.uipanel28, 'Visible','off');
1359         set(handles.uipanel19, 'Visible','off');
1360         set(handles.uipanel25, 'Visible','off');
1361         set(handles.uipanel39, 'Visible','on');
1362     otherwise
1363 end
1364 guidata(hObject,handles);
1365 %Image_Processing_Project_2(handles);
1366
1367
1368
1369 % ---- Executes during object creation, after setting all properties.
1370 function pushbutton30_CreateFcn(hObject, eventdata, handles)
1371 % hObject    handle to pushbutton30 (see GCBO)
1372 % eventdata reserved - to be defined in a future version of MATLAB
1373 % handles    empty - handles not created until after all CreateFcns called
1374 function im = IdealLowPass(im0,fc)
1375 % fc is the circular cutoff frequency which is normalized to [0 1], that is,
1376 % the highest radian frequency \pi of digital signals is mapped to 1.
1377
1378 [ir,ic,iz] = size(im0);
1379 hr = (ir-1)/2;
1380 hc = (ic-1)/2;
1381 [x, y] = meshgrid(-hc:hc, -hr:hr);
1382
1383 mg = sqrt((x/hc).^2 + (y/hr).^2);
1384 lp = double(mg <= fc);
1385
1386 IM = fftshift(fft2(double(im0)));
1387 IP = zeros(size(IM));
1388 for z = 1:iz
1389     IP(:,:,z) = IM(:,:,z) .* lp;
1390 end
1391 im = abs(ifft2(ifftshift(IP)));
1392
1393 % ---- Executes on button press in pushbutton31.
1394 function pushbutton31_Callback(hObject, eventdata, handles)
1395 % hObject    handle to pushbutton31 (see GCBO)
1396 % eventdata reserved - to be defined in a future version of MATLAB
1397 % handles    structure with handles and user data (see GUIDATA)
1398
1399 FT=fft2(handles.gray.image);
1400 FT.real=abs(FT);
1401 Fs=fftshift(FT-real);
1402 FsL=log(Fs);
1403 %axes(handles.axes8);
1404 %imshow(FsL,[]);
1405 fourier=IdealLowPass(handles.gray.image,handles.Low_Ideal);
1406 handles.gray.image=fourier;

```

```

1407 axes(handles.axes8);
1408 imshow(handles.gray_image,[])
1409 guidata(hObject,handles);
1410
1411
1412
1413 function edit50_Callback(hObject, eventdata, handles)
1414 % hObject    handle to edit50 (see GCBO)
1415 % eventdata   reserved - to be defined in a future version of MATLAB
1416 % handles    structure with handles and user data (see GUIDATA)
1417 handles.LowIdeal=str2double(get(hObject,'String'));
1418 % Hints: get(hObject,'String') returns contents of edit50 as text
1419 % str2double(get(hObject,'String')) returns contents of edit50 as a double
1420 guidata(hObject,handles);
1421
1422
1423 % ---- Executes during object creation, after setting all properties.
1424 function edit50_CreateFcn(hObject, eventdata, handles)
1425 % hObject    handle to edit50 (see GCBO)
1426 % eventdata   reserved - to be defined in a future version of MATLAB
1427 % handles    empty - handles not created until after all CreateFcns called
1428 function res = blpf(im,thresh,n)
1429
1430 % inputs
1431 % im is the fourier transform of the image
1432 % thresh is the cutoff circle radius (1,2,3...)
1433 % n is the order of the filter (1,2,3...)
1434
1435 %outputs
1436 % res is the filtered image
1437
1438 [r,c]=size(im);
1439 d0=thresh;
1440
1441 d=zeros(r,c);
1442 h=zeros(r,c);
1443
1444 for i=1:r
1445     for j=1:c
1446         d(i,j)= sqrt( (i-(r/2))^2 + (j-(c/2))^2 );
1447     end
1448 end
1449
1450 for i=1:r
1451     for j=1:c
1452         h(i,j)= 1 / (1+ (d(i,j)/d0)^(2*n) ) ;
1453     end
1454 end
1455
1456
1457 for i=1:r
1458     for j=1:c
1459         res(i,j)=(h(i,j))*im(i,j);
1460     end
1461 end
1462 end
1463
1464
1465
1466
1467
1468 % Hint: edit controls usually have a white background on Windows.
1469 %       See ISPC and COMPUTER.
1470 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1471     set(hObject,'BackgroundColor','white');
1472 end
1473
1474
1475 % ---- Executes on button press in pushbutton32.
1476 function pushbutton32_Callback(hObject, eventdata, handles)
1477 % hObject    handle to pushbutton32 (see GCBO)
1478 % eventdata   reserved - to be defined in a future version of MATLAB
1479 % handles    structure with handles and user data (see GUIDATA)
1480 % simple implementation of frequency domain filters
1481 %read input image
1482 dim=handles.gray_image;
1483 cim=double(dim);
1484 [r,c]=size(cim);
1485
1486 r1=2*r;
1487 c1=2*c;
1488
1489 pim=zeros((r1),(c1));
1490 kim=zeros((r1),(c1));
1491
1492 %padding
1493 for i=1:r
1494     for j=1:c
1495         pim(i,j)=cim(i,j);
1496     end
1497 end
1498
1499 %center the transform
1500 for i=1:r
1501     for j=1:c
1502         kim(i,j)=pim(i,j)*((-1)^(i+j));
1503     end
1504 end
1505
1506
1507 %2D fft

```

```

1508 fim=fft2 (kim);
1509
1510 n=handles.butterorder_low; %order for butterworth filter
1511 thresh=handles.butterradius_low; % cutoff radius in frequency domain for filters
1512
1513 % % function call for low pass filters
1514 % him=glp(fim,thresh); % gaussian low pass filter
1515 him=blpf(fim,thresh,n); % butterworth low pass filter
1516
1517 % % function calls for high pass filters
1518 % him=ghp(fim,thresh); % gaussian low pass filter
1519 %him=bhp(fim,thresh,n); %butterworth high pass filter
1520
1521 % % function call for high boost filtering
1522 % him=hbg(fim,thresh); % using gaussian high pass filter
1523 % him=hbb(fim,thresh,n); % using butterworth high pass filter
1524
1525
1526 %inverse 2D fft
1527 ifim=ifft2 (him);
1528
1529 for i=1:r1
1530   for j=1:c1
1531     ifim(i,j)=ifim(i,j)*((-1)^(i+j));
1532   end
1533 end
1534
1535
1536 % removing the padding
1537 for i=1:r
1538   for j=1:c
1539     rim(i,j)=ifim(i,j);
1540   end
1541 end
1542
1543 % retaining the ral parts of the matrix
1544 rim=real(rim);
1545 rim=uint8 (rim);
1546 handles.gray.image=rim;
1547 axes(handles.axes8);
1548 imshow(handles.gray-image);
1549 guidata(hObject,handles);
1550
1551
1552
1553 function edit51_Callback(hObject, eventdata, handles)
1554 % hObject    handle to edit51 (see GCBO)
1555 % eventdata reserved - to be defined in a future version of MATLAB
1556 % handles    structure with handles and user data (see GUIDATA)
1557 handles.butterorder_low=str2double(get(hObject,'String'))
1558 % Hints: get(hObject,'String') returns contents of edit51 as text
1559 % str2double(get(hObject,'String')) returns contents of edit51 as a double
1560 guidata(hObject,handles);
1561
1562
1563 % --- Executes during object creation, after setting all properties.
1564 function edit51_CreateFcn(hObject, eventdata, handles)
1565 % hObject    handle to edit51 (see GCBO)
1566 % eventdata reserved - to be defined in a future version of MATLAB
1567 % handles    empty - handles not created until after all CreateFcns called
1568
1569 % Hint: edit controls usually have a white background on Windows.
1570 % See ISPC and COMPUTER.
1571 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1572   set(hObject,'BackgroundColor','white');
1573 end
1574
1575
1576
1577 function edit52_Callback(hObject, eventdata, handles)
1578 % hObject    handle to edit52 (see GCBO)
1579 % eventdata reserved - to be defined in a future version of MATLAB
1580 % handles    structure with handles and user data (see GUIDATA)
1581 handles.butterradius_low=str2double(get(hObject,'String'))
1582 % Hints: get(hObject,'String') returns contents of edit52 as text
1583 % str2double(get(hObject,'String')) returns contents of edit52 as a double
1584 guidata(hObject,handles);
1585
1586
1587 % --- Executes during object creation, after setting all properties.
1588 function edit52_CreateFcn(hObject, eventdata, handles)
1589 % hObject    handle to edit52 (see GCBO)
1590 % eventdata reserved - to be defined in a future version of MATLAB
1591 % handles    empty - handles not created until after all CreateFcns called
1592 function res = glp(im,thresh)
1593
1594 % inputs
1595 % im is the fourier transform of the image
1596 % thresh is the cutoff circle radius
1597
1598 %outputs
1599 % res is the filtered image
1600
1601 [r,c]=size(im);
1602 d0=thresh;
1603
1604 d=zeros(r,c);
1605 h=zeros(r,c);
1606
1607 for i=1:r
1608   for j=1:c

```

```

1609      d(i,j)= sqrt( (i-(r/2))^2 + (j-(c/2))^2 );
1610    end
1611
1612  for i=1:r
1613    for j=1:c
1614      h(i,j)=exp ( -( (d(i,j)^2)/(2*(d0^2)) ) );
1615    end
1616
1617 end
1618
1619
1620 for i=1:r
1621   for j=1:c
1622     res(i,j)=(h(i,j))*im(i,j);
1623   end
1624 end
1625
1626 % Hint: edit controls usually have a white background on Windows.
1627 % See ISPC and COMPUTER.
1628 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1629   set(hObject,'BackgroundColor','white');
1630 end
1631
1632
1633
1634
1635 % Hint: edit controls usually have a white background on Windows.
1636 % See ISPC and COMPUTER.
1637 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1638   set(hObject,'BackgroundColor','white');
1639 end
1640
1641
1642 % ---- Executes on button press in pushbutton33.
1643 function pushbutton33_Callback(hObject, eventdata, handles)
1644 % hObject    handle to pushbutton33 (see GCBO)
1645 % eventdata reserved - to be defined in a future version of MATLAB
1646 % handles    structure with handles and user data (see GUIDATA)
1647 dim=handles.gray.image;
1648 cim=double(dim);
1649 [r,c]=size(cim);
1650
1651 r1=2*r;
1652 c1=2*c;
1653
1654 pim=zeros((r1),(c1));
1655 kim=zeros((r1),(c1));
1656
1657 %padding
1658 for i=1:r
1659   for j=1:c
1660     pim(i,j)=cim(i,j);
1661   end
1662 end
1663
1664 %center the transform
1665 for i=1:r
1666   for j=1:c
1667     kim(i,j)=pim(i,j)*((-1)^(i+j));
1668   end
1669 end
1670
1671
1672 %2D fft
1673 fim=fft2(kim);
1674
1675 thresh=handles.gaussradius_low; % cutoff radius in frequency domain for filters
1676
1677 % % function call for low pass filters
1678 him=glp(fim,thresh); % gaussian low pass filter
1679 % him=blpf(fim,thresh,n); % butterworth low pass filter
1680
1681 % % function calls for high pass filters
1682 %him=ghp(fim,thresh); % gaussian low pass filter
1683 %him=bhp(fim,thresh,n); %butterworth high pass filter
1684
1685 % % function call for high boost filtering
1686 % him=hbg(fim,thresh); % using gaussian high pass filter
1687 % him=hbb(fim,thresh,n); % using butterworth high pass filter
1688
1689
1690 %inverse 2D fft
1691 ifim=ifft2(him);
1692
1693 for i=1:r1
1694   for j=1:c1
1695     ifim(i,j)=ifim(i,j)*((-1)^(i+j));
1696   end
1697 end
1698
1699
1700 % removing the padding
1701 for i=1:r
1702   for j=1:c
1703     rim(i,j)=ifim(i,j);
1704   end
1705 end
1706
1707 % retaining the real parts of the matrix
1708 rim=real(rim);
1709 rim=uint8(rim);

```

```

1710 handles.gray_image=rim;
1711 axes(handles.axes8);
1712 imshow(handles.gray_image)
1713 guidata(hObject,handles);
1714
1715
1716
1717 function edit53_Callback(hObject, eventdata, handles)
1718 % hObject    handle to edit53 (see GCBO)
1719 % eventdata reserved - to be defined in a future version of MATLAB
1720 % handles    structure with handles and user data (see GUIDATA)
1721
1722 % Hints: get(hObject,'String') returns contents of edit53 as text
1723 % str2double(get(hObject,'String')) returns contents of edit53 as a double
1724 handles.gaussradius_low=str2double(get(hObject,'String'));
1725 guidata(hObject,handles);
1726
1727 % ---- Executes during object creation, after setting all properties.
1728 function edit53_CreateFcn(hObject, eventdata, handles)
1729 % hObject    handle to edit53 (see GCBO)
1730 % eventdata reserved - to be defined in a future version of MATLAB
1731 % handles    empty - handles not created until after all CreateFcns called
1732 function im = IdealHiPass(im0,fc)
1733 % fc is the circular cutoff frequency which is normalized to [0 1], that is,
1734 % the highest radian frequency \pi of digital signals is mapped to 1.
1735
1736 [ir,ic ,iz] = size(im0);
1737 hr = (ir-1)/2;
1738 hc = (ic-1)/2;
1739 [x, y] = meshgrid(-hc:hc, -hr:hr);
1740
1741 mg = sqrt((x/hc).^2 + (y/hr).^2);
1742 lp = double(mg >= fc);
1743
1744 IM = fftshift(fft2(double(im0)));
1745 IP = zeros(size(IM));
1746 for z = 1:iz
1747     IP(:,:,z) = IM(:,:,z) .* lp;
1748 end
1749 im = abs(ifft2(ifftshift(IP)));
1750
1751 % Hint: edit controls usually have a white background on Windows.
1752 % See ISPC and COMPUTER.
1753 if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
1754     set(hObject, 'BackgroundColor', 'white');
1755 end
1756
1757
1758 % ---- Executes on button press in pushbutton34.
1759 function pushbutton34_Callback(hObject, eventdata, handles)
1760 % hObject    handle to pushbutton34 (see GCBO)
1761 % eventdata reserved - to be defined in a future version of MATLAB
1762 % handles    structure with handles and user data (see GUIDATA)
1763 fourier=IdealHiPass(handles.gray_image,handles.Low_Ideal);
1764 handles.gray_image=fourier;
1765 axes(handles.axes8);
1766 imshow(handles.gray_image,[])
1767 guidata(hObject,handles);
1768
1769
1770
1771 function edit54_Callback(hObject, eventdata, handles)
1772 % hObject    handle to edit54 (see GCBO)
1773 % eventdata reserved - to be defined in a future version of MATLAB
1774 % handles    structure with handles and user data (see GUIDATA)
1775 handles.Low_Ideal=str2double(get(hObject,'String'));
1776 % Hints: get(hObject,'String') returns contents of edit54 as text
1777 % str2double(get(hObject,'String')) returns contents of edit54 as a double
1778 guidata(hObject,handles);
1779
1780
1781 % ---- Executes during object creation, after setting all properties.
1782 function edit54_CreateFcn(hObject, eventdata, handles)
1783 % hObject    handle to edit54 (see GCBO)
1784 % eventdata reserved - to be defined in a future version of MATLAB
1785 % handles    empty - handles not created until after all CreateFcns called
1786
1787 % Hint: edit controls usually have a white background on Windows.
1788 % See ISPC and COMPUTER.
1789 if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
1790     set(hObject, 'BackgroundColor', 'white');
1791 end
1792
1793
1794
1795 function edit55_Callback(hObject, eventdata, handles)
1796 % hObject    handle to edit55 (see GCBO)
1797 % eventdata reserved - to be defined in a future version of MATLAB
1798 % handles    structure with handles and user data (see GUIDATA)
1799 handles.gaussradius_Hi=str2double(get(hObject,'String'));
1800 % Hints: get(hObject,'String') returns contents of edit55 as text
1801 % str2double(get(hObject,'String')) returns contents of edit55 as a double
1802 guidata(hObject,handles);
1803
1804
1805 % ---- Executes during object creation, after setting all properties.
1806 function edit55_CreateFcn(hObject, eventdata, handles)
1807 % hObject    handle to edit55 (see GCBO)
1808 % eventdata reserved - to be defined in a future version of MATLAB
1809 % handles    empty - handles not created until after all CreateFcns called
1810 function res = ghp(im,thresh)

```

```

1811 % inputs
1812 % im is the fourier transform of the image
1813 % 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     end
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

```

```

1912     rim(i,j)=ifim(i,j);
1913   end
1914 end
1915
1916 % retaining the real parts of the matrix
1917 rim=real(rim);
1918 rim=uint8(rim);
1919 handles.gray_image=rim;
1920 axes(handles.axes8);
1921 imshow(handles.gray_image)
1922 guidata(hObject,handles);
1923
1924
1925
1926 function edit56_Callback(hObject, eventdata, handles)
1927 % hObject    handle to edit56 (see GCBO)
1928 % eventdata reserved - to be defined in a future version of MATLAB
1929 % handles    structure with handles and user data (see GUIDATA)
1930 handles.butter_order=str2double(get(hObject,'String'));
1931 % Hints: get(hObject,'String') returns contents of edit56 as text
1932 %         str2double(get(hObject,'String')) returns contents of edit56 as a double
1933 guidata(hObject,handles);
1934
1935
1936 % --- Executes during object creation, after setting all properties.
1937 function edit56_CreateFcn(hObject, eventdata, handles)
1938 % hObject    handle to edit56 (see GCBO)
1939 % eventdata reserved - to be defined in a future version of MATLAB
1940 % handles    empty - handles not created until after all CreateFcns called
1941 % Hint: edit controls usually have a white background on Windows.
1942 % See ISPC and COMPUTER.
1943 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
1944   set(hObject,'BackgroundColor','white');
1945 end
1946
1947
1948
1949 function edit57_Callback(hObject, eventdata, handles)
1950 % hObject    handle to edit57 (see GCBO)
1951 % eventdata reserved - to be defined in a future version of MATLAB
1952 % handles    structure with handles and user data (see GUIDATA)
1953 handles.butter_radius=str2double(get(hObject,'String'));
1954 guidata(hObject,handles);
1955
1956 % Hints: get(hObject,'String') returns contents of edit57 as text
1957 %         str2double(get(hObject,'String')) returns contents of edit57 as a double
1958
1959
1960 % --- Executes during object creation, after setting all properties.
1961 function edit57_CreateFcn(hObject, eventdata, handles)
1962 % hObject    handle to edit57 (see GCBO)
1963 % eventdata reserved - to be defined in a future version of MATLAB
1964 % handles    empty - handles not created until after all CreateFcns called
1965 function res = bhp(im,thresh,n)
1966
1967 % inputs
1968 % im is the fourier transform of the image
1969 % thresh is the cutoff circle radius
1970
1971 %outputs
1972 % res is the filtered image
1973
1974 [r,c]=size(im);
1975 d0=thresh;
1976
1977 d=zeros(r,c);
1978 h=zeros(r,c);
1979
1980 for i=1:r
1981   for j=1:c
1982     d(i,j)= sqrt( (i-(r/2))^2 + (j-(c/2))^2 );
1983   end
1984 end
1985
1986 for i=1:r
1987   for j=1:c
1988     h(i,j)= 1 / (1+ (d0/d(i,j))^(2*n) ) ;
1989   end
1990 end
1991
1992
1993 for i=1:r
1994   for j=1:c
1995     res(i,j)=(h(i,j))*im(i,j);
1996   end
1997 end
1998 end
1999
2000
2001
2002
2003 % Hint: edit controls usually have a white background on Windows.
2004 % See ISPC and COMPUTER.
2005 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
2006   set(hObject,'BackgroundColor','white');
2007 end
2008
2009
2010 % --- Executes on button press in pushbutton36.
2011 function pushbutton36_Callback(hObject, eventdata, handles)
2012 % hObject    handle to pushbutton36 (see GCBO)

```

```

2013 % eventdata reserved - to be defined in a future version of MATLAB
2014 % handles structure with handles and user data (see GUIDATA)
2015 dim=handles.gray_image;
2016 cim=double(dim);
2017 [r,c]=size(cim);
2018
2019 r1=2*r;
2020 c1=2*c;
2021
2022 pim=zeros((r1),(c1));
2023 kim=zeros((r1),(c1));
2024
2025 %padding
2026 for i=1:r
2027     for j=1:c
2028         pim(i,j)=cim(i,j);
2029     end
2030 end
2031
2032 %center the transform
2033 for i=1:r
2034     for j=1:c
2035         kim(i,j)=pim(i,j)*((-1)^(i+j));
2036     end
2037 end
2038
2039
2040 %2D fft
2041 fim=fft2(kim);
2042
2043 thresh=handles.butter_radius; % cutoff radius in frequency domain for filters
2044 n=handles.butter_order;
2045 % % function call for low pass filters
2046 % him=glp(fim,thresh); % gaussian low pass filter
2047 % him=blpf(fim,thresh,n); % butterworth low pass filter
2048
2049 % % function calls for high pass filters
2050 %him=ghp(fim,thresh); % gaussian low pass filter
2051 him=bhp(fim,thresh,n); %butterworth high pass filter
2052
2053 % % function call for high boost filtering
2054 % him=hbg(fim,thresh); % using gaussian high pass filter
2055 % him=hbb(fim,thresh,n); % using butterworth high pass filter
2056
2057
2058 %inverse 2D fft
2059 ifim=ifft2(him);
2060
2061 for i=1:r1
2062     for j=1:c1
2063         ifim(i,j)=ifim(i,j)*((-1)^(i+j));
2064     end
2065 end
2066
2067
2068 % removing the padding
2069 for i=1:r
2070     for j=1:c
2071         rim(i,j)=ifim(i,j);
2072     end
2073 end
2074
2075 % retaining the real parts of the matrix
2076 rim=real(rim);
2077 rim=uint8(rim);
2078 handles.gray_image=rim;
2079 axes(handles.axes8);
2080 imshow(handles.gray_image);
2081 guidata(hObject,handles);
2082
2083
2084 % --- Executes on button press in pushbutton37.
2085 function pushbutton37_Callback(hObject, eventdata, handles)
2086 % hObject handle to pushbutton37 (see GCBO)
2087 % eventdata reserved - to be defined in a future version of MATLAB
2088 % handles structure with handles and user data (see GUIDATA)
2089
2090 Kaverage = filter2(fspecial('average',handles.average_size),handles.gray_image)/255;
2091 axes(handles.axes8);
2092 handles.gray_image=Kaverage;
2093 imshow(handles.gray_image)
2094
2095
2096
2097 function edit58_Callback(hObject, eventdata, handles)
2098 % hObject handle to edit58 (see GCBO)
2099 % eventdata reserved - to be defined in a future version of MATLAB
2100 % handles structure with handles and user data (see GUIDATA)
2101 handles.average_size=str2double(get(hObject,'String'))
2102 % Hints: get(hObject,'String') returns contents of edit58 as text
2103 % str2double(get(hObject,'String')) returns contents of edit58 as a double
2104 guidata(hObject,handles);
2105
2106
2107 % --- Executes during object creation, after setting all properties.
2108 function edit58_CreateFcn(hObject, eventdata, handles)
2109 % hObject handle to edit58 (see GCBO)
2110 % eventdata reserved - to be defined in a future version of MATLAB
2111 % handles empty - handles not created until after all CreateFcns called
2112
2113 % Hint: edit controls usually have a white background on Windows.

```

```

2114 % See ISPC and COMPUTER.
2115 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
2116 set(hObject,'BackgroundColor','white');
2117 end
2118
2119
2120
2121 function edit59_Callback(hObject, eventdata, handles)
2122 % hObject    handle to edit59 (see GCBO)
2123 % eventdata reserved - to be defined in a future version of MATLAB
2124 % handles    structure with handles and user data (see GUIDATA)
2125 handles.med.size=str2double(get(hObject,'String'));
2126 % Hints: get(hObject,'String') returns contents of edit59 as text
2127 % str2double(get(hObject,'String')) returns contents of edit59 as a double
2128 guidata(hObject,handles);
2129
2130
2131 % --- Executes during object creation, after setting all properties.
2132 function edit59_CreateFcn(hObject, eventdata, handles)
2133 % hObject    handle to edit59 (see GCBO)
2134 % eventdata reserved - to be defined in a future version of MATLAB
2135 % handles    empty - handles not created until after all CreateFcns called
2136
2137 % Hint: edit controls usually have a white background on Windows.
2138 % See ISPC and COMPUTER.
2139 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
2140 set(hObject,'BackgroundColor','white');
2141 end
2142
2143
2144 % --- Executes on button press in pushbutton38.
2145 function pushbutton38_Callback(hObject, eventdata, handles)
2146 % hObject    handle to pushbutton38 (see GCBO)
2147 % eventdata reserved - to be defined in a future version of MATLAB
2148 % handles    structure with handles and user data (see GUIDATA)
2149 Median.noise = medfilt2(handles.gray.image, [handles.med_size handles.med_size]);
2150 handles.gray.image=Median.noise;
2151 axes(handles.axes8);
2152 imshow(handles.gray.image)
2153
2154
2155
2156 function edit60_Callback(hObject, eventdata, handles)
2157 % hObject    handle to edit60 (see GCBO)
2158 % eventdata reserved - to be defined in a future version of MATLAB
2159 % handles    structure with handles and user data (see GUIDATA)
2160 handles.wein.size=str2double(get(hObject,'String'));
2161 % Hints: get(hObject,'String') returns contents of edit60 as text
2162 % str2double(get(hObject,'String')) returns contents of edit60 as a double
2163 guidata(hObject,handles);
2164
2165
2166 % --- Executes during object creation, after setting all properties.
2167 function edit60_CreateFcn(hObject, eventdata, handles)
2168 % hObject    handle to edit60 (see GCBO)
2169 % eventdata reserved - to be defined in a future version of MATLAB
2170 % handles    empty - handles not created until after all CreateFcns called
2171
2172 % Hint: edit controls usually have a white background on Windows.
2173 % See ISPC and COMPUTER.
2174 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
2175 set(hObject,'BackgroundColor','white');
2176 end
2177
2178
2179 % --- Executes on button press in pushbutton39.
2180 function pushbutton39_Callback(hObject, eventdata, handles)
2181 % hObject    handle to pushbutton39 (see GCBO)
2182 % eventdata reserved - to be defined in a future version of MATLAB
2183 % handles    structure with handles and user data (see GUIDATA)
2184 Adaptive = wiener2(handles.gray.image,[handles.wein_size handles.wein_size]);
2185 handles.gray.image=Adaptive;
2186 axes(handles.axes8);
2187 imshow(handles.gray.image)
2188
2189
2190 % --- Executes on button press in pushbutton40.
2191 function pushbutton40_Callback(hObject, eventdata, handles)
2192 % hObject    handle to pushbutton40 (see GCBO)
2193 % eventdata reserved - to be defined in a future version of MATLAB
2194 % handles    structure with handles and user data (see GUIDATA)
2195 % Start of PCA code,
2196 %Data = imread('butterfly.jpg');
2197 Data.gray = handles.gray.image;
2198 Data.grayD = im2double(Data.gray);
2199
2200 Data.mean = mean(Data.grayD);
2201 [a b] = size(Data.gray);
2202 Data.meanNew = repmat(Data.mean,a,1);
2203 DataAdjust = Data.grayD - Data.meanNew;
2204 cov_data = cov(DataAdjust);
2205 [V, D] = eig(cov_data);
2206 V.trans = transpose(V);
2207 DataAdjust.trans = transpose(DataAdjust);
2208 FinalData = V.trans * DataAdjust.trans;
2209 % End of PCA code
2210 % Start of Inverse PCA code,
2211 OriginalData_trans = inv(V.trans) * FinalData;
2212 OriginalData = transpose(OriginalData_trans) + Data.meanNew;
2213 % End of Inverse PCA code
2214
```

```

2215 % Image compression
2216 PCs=handles.pca_dim;
2217 PCs = b - PCs;
2218 Reduced_V = V;
2219 for i = 1:PCs,
2220 Reduced_V(:,1) = [];
2221 end
2222 Y=Reduced_V'* DataAdjust_trans;
2223 Compressed_Data=Reduced_V*Y;
2224 handles.Compressed_Data = Compressed_Data' + Data_meanNew;
2225 axes(handles.axes8);
2226 imshow(handles.Compressed_Data)
2227 guidata(hObject,handles);
2228 % End of image compression
2229
2230
2231
2232 function edit61_Callback(hObject, eventdata, handles)
2233 % hObject handle to edit61 (see GCBO)
2234 % eventdata reserved - to be defined in a future version of MATLAB
2235 % handles structure with handles and user data (see GUIDATA)
2236 handles.pca_dim=str2double(get(hObject,'String'));
2237 % Hints: get(hObject,'String') returns contents of edit61 as text
2238 % str2double(get(hObject,'String')) returns contents of edit61 as a double
2239 guidata(hObject,handles);
2240
2241
2242 % ---- Executes during object creation, after setting all properties.
2243 function edit61_CreateFcn(hObject, eventdata, handles)
2244 % hObject handle to edit61 (see GCBO)
2245 % eventdata reserved - to be defined in a future version of MATLAB
2246 % handles empty - handles not created until after all CreateFcns called
2247
2248 % Hint: edit controls usually have a white background on Windows.
2249 % See ISPC and COMPUTER.
2250 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
2251 set(hObject,'BackgroundColor','white');
2252 end
2253
2254
2255 % ---- Executes on button press in pushbutton41.
2256 function pushbutton41_Callback(hObject, eventdata, handles)
2257 % hObject handle to pushbutton41 (see GCBO)
2258 % eventdata reserved - to be defined in a future version of MATLAB
2259 % handles structure with handles and user data (see GUIDATA)
2260 imwrite(handles.Compressed_Data,'output.jpg');
2261
2262
2263
2264 function edit62_Callback(hObject, eventdata, handles)
2265 % hObject handle to edit62 (see GCBO)
2266 % eventdata reserved - to be defined in a future version of MATLAB
2267 % handles structure with handles and user data (see GUIDATA)
2268
2269 % Hints: get(hObject,'String') returns contents of edit62 as text
2270 % str2double(get(hObject,'String')) returns contents of edit62 as a double
2271 InfoImage = imfinfo(handles.gray_image);
2272 set(handles.edit62,'String',num2str(InfoImage.size));
2273
2274 % ---- Executes during object creation, after setting all properties.
2275 function edit62_CreateFcn(hObject, eventdata, handles)
2276 % hObject handle to edit62 (see GCBO)
2277 % eventdata reserved - to be defined in a future version of MATLAB
2278 % handles empty - handles not created until after all CreateFcns called
2279
2280 % Hint: edit controls usually have a white background on Windows.
2281 % See ISPC and COMPUTER.
2282 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
2283 set(hObject,'BackgroundColor','white');
2284 end
2285
2286
2287 % ---- Executes on button press in pushbutton42.
2288 function pushbutton42_Callback(hObject, eventdata, handles)
2289 % hObject handle to pushbutton42 (see GCBO)
2290 % eventdata reserved - to be defined in a future version of MATLAB
2291 % handles structure with handles and user data (see GUIDATA)
2292 newimage=imsubtract(handles.grey_1,handles.gray_image);
2293 axes(handles.axes7);
2294 handles.gray_image=newimage;
2295 imshow(newimage)
2296 guidata(hObject,handles);

```

## References

- [1] Digital Image Processing 2nd Ed. by Gonzalez, Woods, and Eddins.
- [2] <http://support.esri.com/technical-article/000005606>