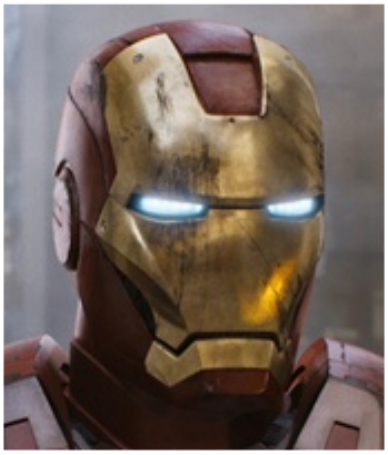
Computer Vision HW1

Chien-Cheng Lai
ChBE

I. Part 1

a) Exercise 1

- Crop the head of the superhero from the image and save as PNG file.
  `(1:200, 230:400, :)`

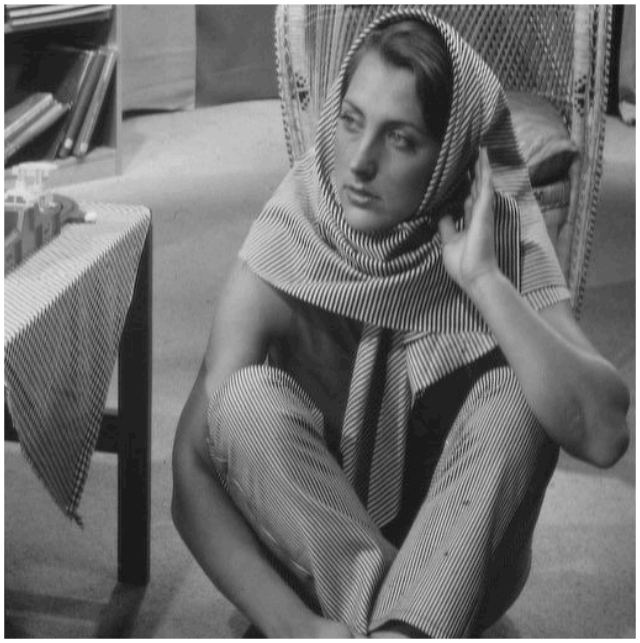


- Display the green component of the cropped image.

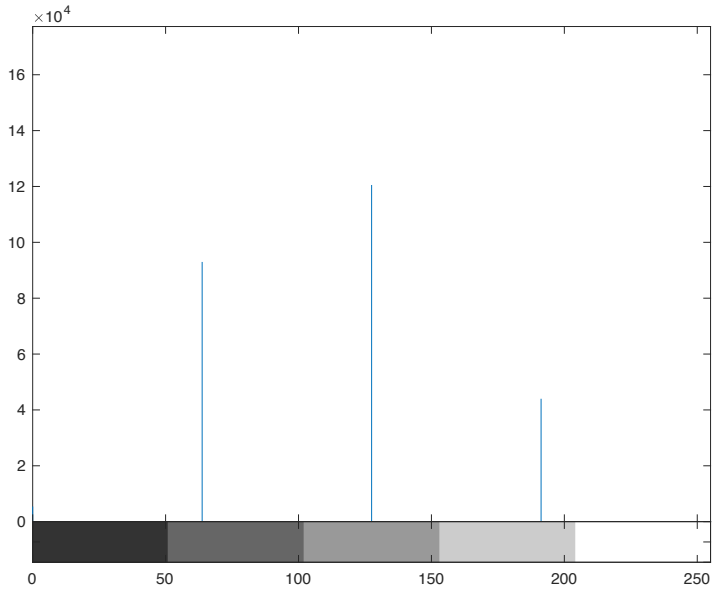
- Change the order of the color components to [Green,Red,Blue] for the original image. (`imshow(im1(:,:,[2,1,3]))`)

b) Exercise 2

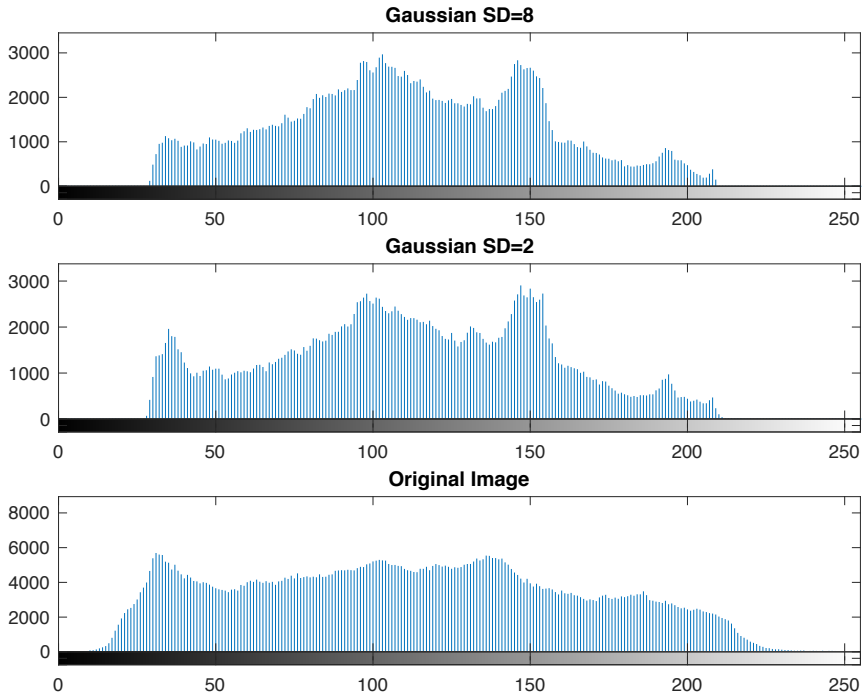
- Use *rgb2gray* to convert the image to gray scale.



- Plot a histogram of the gray scale image with bin-size of 5 pixels.



- Use *fspecial* and *imfilter* to blur the gray scale image by using Gaussian filters of size 15 × 15 with standard deviations 2 and 8. And plot the histogram.



Gaussian will filter out the signal on the both ends of the x-direction. However, the standard deviations increasing from 2 to 8 seems not to affect the signal much except on the small value x.

- Subtract the blurred image obtained using the filter with standard deviation of 2 from the original gray scale image.

- Threshold the resultant image at 5% of its maximum pixel value and display the final image.

There is not much difference between these two pictures. They both show contour of the picture. I think they probably filter out similar signal of the picture.

II. Part 2

a) Filter the following matrices I1 and I2 without using built-in MATLAB functions.

$$I1 = \begin{bmatrix} 120 & 110 & 90 & 115 & 40 \\ 145 & 135 & 135 & 65 & 35 \\ 125 & 115 & 55 & 35 & 25 \\ 80 & 45 & 45 & 20 & 15 \\ 40 & 35 & 25 & 10 & 10 \end{bmatrix}$$

$$I2 = \begin{bmatrix} 125 & 130 & 135 & 110 & 125 \\ 145 & 135 & 135 & 155 & 125 \\ 65 & 60 & 55 & 45 & 40 \\ 40 & 35 & 40 & 25 & 15 \\ 15 & 15 & 20 & 15 & 10 \end{bmatrix}$$

Use *padarray* to pad repeating border elements of array.

PI1 = padarray(I1, [1,1], 'replicate')
PI2 = padarray(I2, [1,1], 'replicate')

I choose to use replicate because I think it is more suitable in this case. They become:

$$I1 = \begin{bmatrix} 120 & 120 & 110 & 90 & 115 & 40 & 40 \\ 120 & 120 & 110 & 90 & 115 & 40 & 40 \\ 145 & 145 & 135 & 135 & 65 & 35 & 35 \\ 125 & 125 & 115 & 55 & 35 & 25 & 25 \\ 80 & 80 & 45 & 45 & 20 & 15 & 15 \\ 40 & 40 & 35 & 25 & 10 & 10 & 10 \\ 40 & 40 & 35 & 25 & 10 & 10 & 10 \end{bmatrix}$$

$$I2 = \begin{bmatrix} 125 & 125 & 130 & 135 & 110 & 125 & 125 \\ 125 & 125 & 130 & 135 & 110 & 125 & 125 \\ 145 & 145 & 135 & 135 & 155 & 125 & 125 \\ 65 & 65 & 60 & 55 & 45 & 40 & 40 \\ 40 & 40 & 35 & 40 & 25 & 15 & 15 \\ 15 & 15 & 15 & 20 & 15 & 10 & 10 \\ 15 & 15 & 15 & 20 & 15 & 10 & 10 \end{bmatrix}$$

Then apply the flowing 3 filters:

- Filter 1: $\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$
- Filter 2: $\frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$
- Filter 3: $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

The finally results:

```
I1_filter1 =

  116.6667  106.6667  105.0000   81.6667   65.0000
  141.6667  138.3333  111.6667   78.3333   45.0000
  121.6667   98.3333   68.3333   38.3333   28.3333
   68.3333   56.6667   36.6667   26.6667   16.6667
   38.3333   33.3333   23.3333   15.0000   10.0000


I2_filter1 =

  126.6667  130.0000  125.0000  123.3333  120.0000
  141.6667  138.3333  141.6667  138.3333  135.0000
   63.3333   60.0000   53.3333   46.6667   41.6667
   38.3333   38.3333   33.3333   26.6667   18.3333
   15.0000   16.6667   16.6667   15.0000   11.6667

I1_filter2 =

  128.3333  118.3333  105.0000   98.3333   38.3333
  130.0000  120.0000   93.3333   71.6667   33.3333
  116.6667   98.3333   78.3333   40.0000   25.0000
   81.6667   65.0000   41.6667   21.6667   16.6667
   53.3333   38.3333   31.6667   13.3333   11.6667


I2_filter2 =

  131.6667  131.6667  135.0000  125.0000  125.0000
  111.6667  108.3333  108.3333  103.3333   96.6667
   83.3333   76.6667   76.6667   75.0000   60.0000
   40.0000   36.6667   38.3333   28.3333   21.6667
   23.3333   21.6667   26.6667   18.3333   11.6667
```

```
I1_filter3 =

  125.0000  117.2222  107.2222   80.5556   58.3333
  126.6667  114.4444   95.0000   66.1111   46.1111
  110.5556   97.7778   72.2222   47.7778   30.0000
   76.1111   62.7778   42.7778   26.6667   18.3333
   48.3333   41.1111   27.7778   18.8889   12.2222


I2_filter3 =

  131.6667  132.7778  130.5556  128.3333  125.0000
  110.5556  109.4444  106.6667  102.7778   98.8889
   81.1111   78.8889   76.1111   70.5556   65.0000
   38.8889   38.3333   34.4444   29.4444   23.8889
   22.7778   23.8889   22.2222   18.8889   13.8889
```

b) Apply following filters on the gray scale image of barbara from Part I.

1. Central difference Gradient filter
2. Sobel filter
3. Mean filter
4. Median filter

The functions I use are below:

```
%%
%Central difference Gradient filter
dx = imfilter(double(grim2)/256, [-1/2 0 1/2]);
dy = imfilter(double(grim2)/256, [-1/2 0 1/2]');
cenbar = (dx.^2 + dy.^2).^(1/2);
```

```
%%
%Sobel filter
sobfilt = fspecial('sobel');
sobbar = imfilter(grim2,sobfilt);
```

```
%%
%Mean filter
meanfilt = fspecial('average');
meanbar = imfilter(grim2,meanfilt);
```

```
%%
%Median filter
medbar = medfilt2(grim2);
```

The result:

Central difference Gradient filter

Sobel filter

Mean filter

Median filter

c) Smoothing

- Average

```
%smoothing_average
im3 = imread('/Users/aaron/Desktop/Computer Vision/Assignment 01/camera_man_noisy.png');
avfilt2 = fspecial('average',[2 2]);
avfilt4 = fspecial('average',[4 4]);
avfilt8 = fspecial('average',[8 8]);
avfilt16 = fspecial('average',[16 16]);

av2cam = imfilter(im3,avfilt2);
av4cam = imfilter(im3,avfilt4);
av8cam = imfilter(im3,avfilt8);
av16cam = imfilter(im3,avfilt16);
```

- Gaussian

```
%smoothing_gaussan

gafilt2 = fspecial('gaussian',8,2);
gafilt4 = fspecial('gaussian',16,4);
gafilt8 = fspecial('gaussian',32,8);
gafilt16 = fspecial('gaussian',64,16);

ga2cam = imfilter(im3,gafilt2);
ga4cam = imfilter(im3,gafilt4);
ga8cam = imfilter(im3,gafilt8);
ga16cam = imfilter(im3,gafilt16);
```

The results:

Average filter size 2x2

Average filter size 4x4

Average filter size 8x8

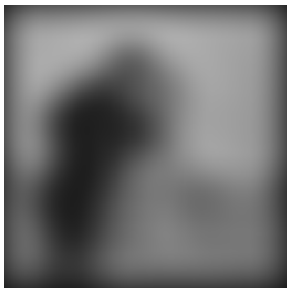
Average filter size 16x16

Gaussian filter SD=2

Gaussian filter SD=4

Gaussian filter SD=8

Gaussian filter SD=16



It’s hard to say which one is better in this case. But I think the Average filter is slightly better than Gaussian filter since it has less effect on the edge of the picture.

When I increase the box filter size, the picture becomes more blur. In the similar way, when I increase the standard deviation of gaussian filter, the picture becomes more blur too.

d) Edge preserving smoothing

Function:

```
%Edge preserving smoothing

cam = double(im3)/255;

% Set bilateral filter parameters.
w     = 5;       % bilateral filter half-width
sigma = [3 0.35]; % bilateral filter standard deviations

% Apply bilateral filter to each image.
bflt_cam = bfilter2(cam,w,sigma);

imagesc(bflt_cam);axis image; colormap gray;
```

Result:

If we choose the large intensity standard deviation, the picture would become very blurred, otherwise, with the small intensity standard deviation, the effect of the filter on the picture becomes small as well.

In this case, I think 0.35 is a good choice, the picture won't be too blurred and also filter out much noise.