Xi'an Jiaotong-Liverpool University
西交利物浦大學

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

EEE330 lab report 2

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Engineering

Student Name : Lingxuan Kong
Student ID : 1405867

Contents

1 Task1

The psnr could be consider as an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. And it most easily defined via the mean squared error (MSE), given the image pixel is $m \times n$, the two input images are I and K, so the MSE could be defined as:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2$$

The PSNR (in dB) is defined as:

$$psnr = 10log_{10}(\frac{MAX^2}{MSE})$$

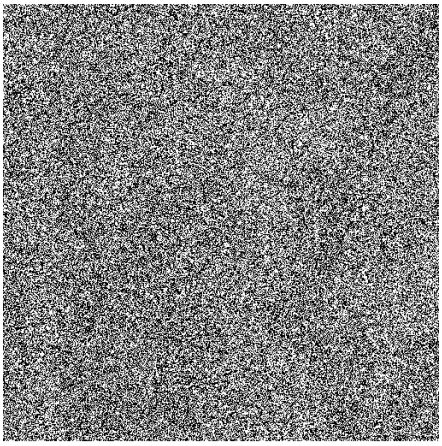
Then, we write a function in matlab to calculate psnr:

```
function [PSNR, MSE]=psnr(m1,m2)%m1,m2 are the two inputs
[m1,m2] = size(I);%obtain the size
Diff = double(m1)-double(m2);%change the data type to double
MSE = sum(Diff(:).^2)/numel(I);%calculate MSE
PSNR=10*log10(255^2/MSE);%calculate psnr
end
```

2 Task2

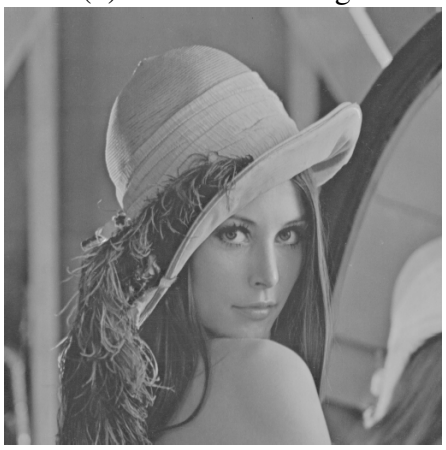
In this task, we first add Gaussian white noise with zero mean and variance 10 to the image. Then, add salt & pepper noise with noise density 10% to the image, and display them:

(a) Reference image



(b) White noise image

(c) Salt & Pepper noise histogram



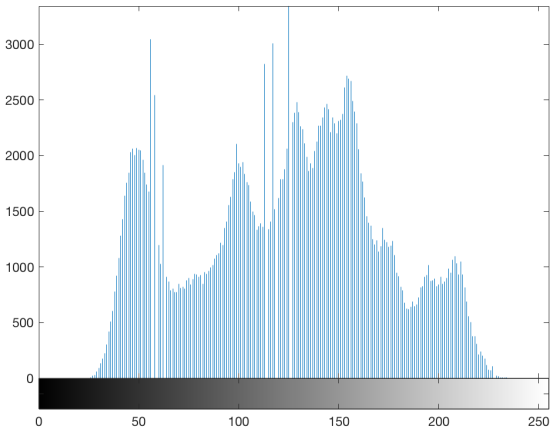
(d) Low dynamic range image

Figure 1: The original image and three images with noise

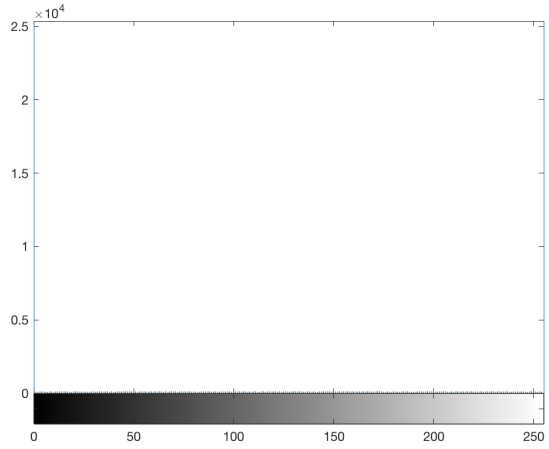
Applying the previous knowledge, we could calculate the PSNR:

Table 1: PSNR of previouse images

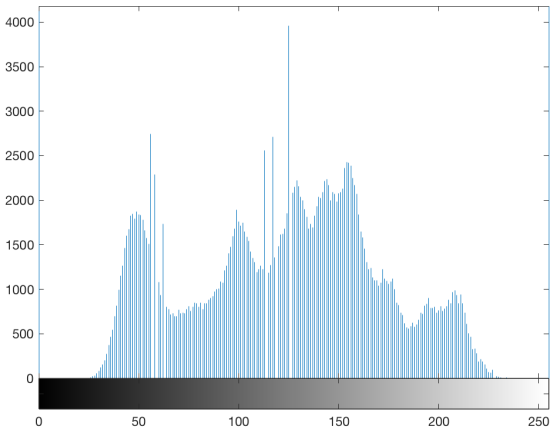
Image	im_wn	im_SP	im_low_dynamic_range
PSNR(dB)	5.92	15.43	23.99



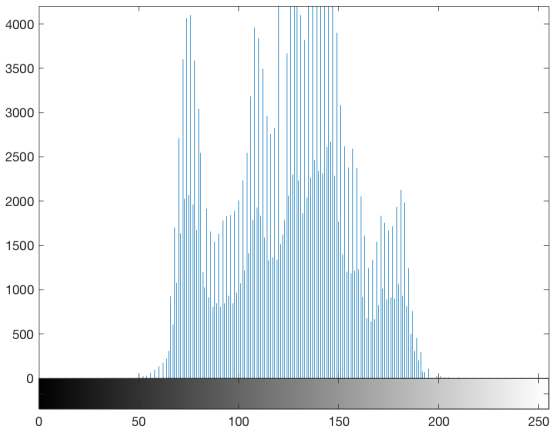
(a) Reference image histogram



(b) White noise histogram



(c) Salt & Pepper noise histogram



(d) Low dynamic range histogram

Figure 2: The histogram of all previous photos

Based on the previous images histogram, we could draw some important conclusions:

(a) From the reference photo, we could that the gray range should be [30,230], and it covers the entire range of possible gray levels, tend to distribute them uniformly. Therefore, we call the reference image as 'balanced' image, which gives more pleasant look and reveals rich details.

(b) In this image, we could find that there is no gray level in the image and histogram. In another word, this image only includes noise without any useful information. So, noise full this image, because of high variance of white noise. Additional, its PSNR is very small, which means it lost too much details and information.

(c) According to the Salt & Pepper noise image and histogram, we could find its gray range is the same as reference image. However, although it has the same gray level as the reference, it has a lower gray pixels, which means it lost some details and information. In another word, because its less image pixels, there will be lots of noise on this image. In addition, from its PSNR and photo, we could find that this image it clearer than white noise, but its PSNR also not high enough.

(d) This image is low dynamic range. When we compare it with the reference image, we can find that it level range should be [50,200], which is smaller than the reference image. Then we look at this image carefully, it is easy to find that this image seems covered with 'a layer of yarn'. Then we observe the histogram, low dynamic range. It is not light in somewhere should light, and not dark in somewhere should be dark. In another word, it looks not clear enough, and has a low level contrast. The PSNR is higher than the previous two images, and it looks much more clear than the previous two images, because it has smaller noise.

3 Task 3

3.1 x=[50,100] & y=[30,180]

In this section, the students should learn how to operate piecewise-Linear transformation functions, which includes two significant parts, one is contrast enhancing, and the other is gray level slicing. In this task, I will focus on contrast enhancing by linear change. Based on the knowledge on the lecture, piece-wise linear function could be designed based on the histogram of the original image, we could operate on any interval that we interested.

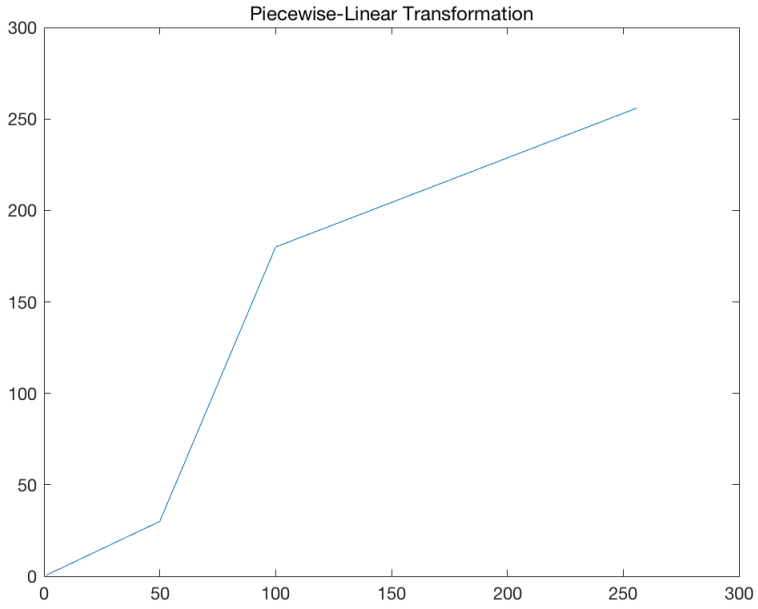


Figure 3: Operating interval

Based on this figure, we can find that:

- The slope from 50 to 100 should be:

$$K_{op} = \frac{180-30}{100-50} = \frac{150}{50} = 3$$

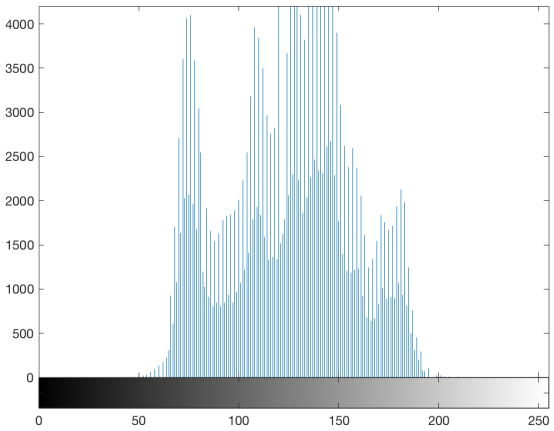
  Contrast enhancing will make sense on this interval.

- The slope of the other interval should be:

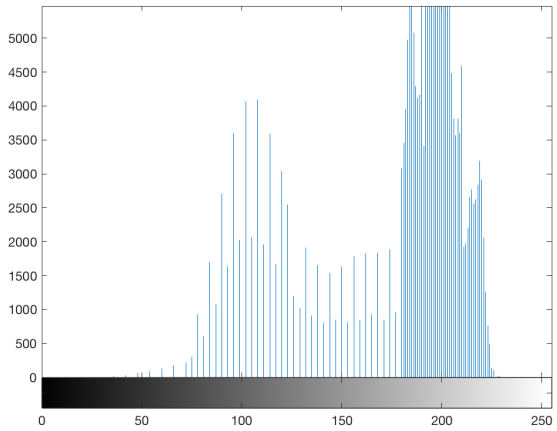
$$K_{non-op} = \frac{30}{50} = 0.6$$

(a) Low dynamic range image

(b) Contrast enhancing image



(c) histogram of low dynamic range



(d) histogram of Contrast enhancing image

According to the previous four images and histogram, we can obtain some important conclusions:

(1) When the students observe the contrast enhancing image carefully, it is easy to find that this image is lighter than the reference one. Additional, this image seems lost a great deal of details. By the same theory, this image seems 'Overexposed' than the reference one.

(2) Then we observe the histogram of contrast enhancing image, we can find that on curve [30,180], there are much less pixels than the reference histogram. Thus, most gray pixels all concentrate on the interval [180,230], which is light gray level. Therefore, this image looks much light than the reference one, because most of its pixels points distribute on the interval [180,230], which will display the light gray color.

(3) The PSNR of this image is:

$$psnr = 12.89$$

3.2 x=[120,180] & y=[50,200]

In the second part of this task, we will apply contrast enhancing on another interval:

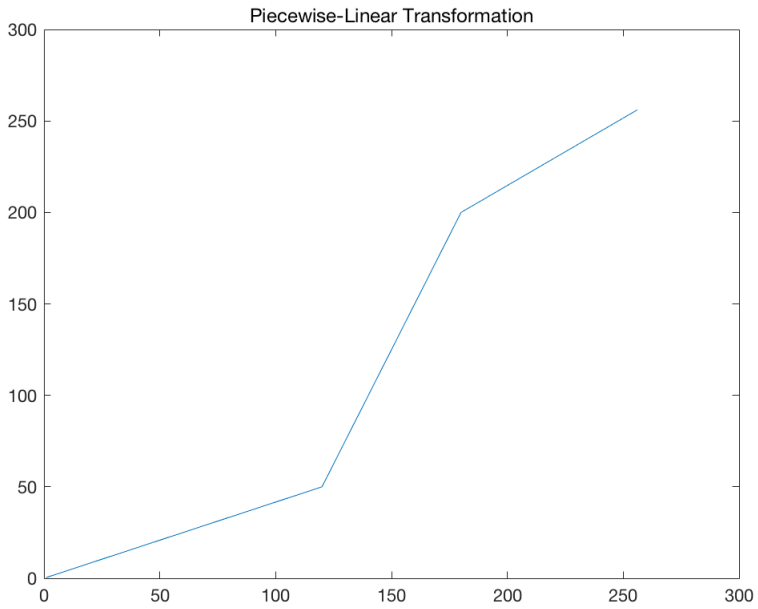


Figure 4: Operating interval

- The slope of contrast enhancing interval should be:

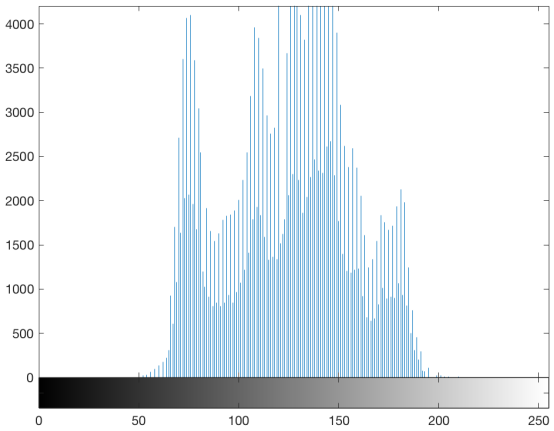
$$K_{op} = \frac{200-50}{180-120} = \frac{150}{60} = 2.5$$

- The slope of the other part is:

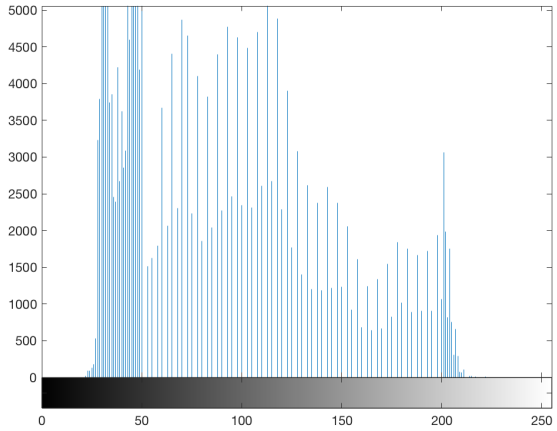
$$K_{non-op} = \frac{50}{120} \approx 0.42$$

(a) Low dynamic range image

(b) Contrast enhancing image



(c) histogram of low dynamic range



(d) histogram of Contrast enhancing image

Based on these four images and previous part, we can draw some important conclusions:

(1) The image after contrast enhance is much darker than the reference one. In somewhere dark, we could find much more details than reference one. However, in somewhere light, this image lost much details. Then we view in its entirety, it display a high contrast and much details in dark points, but also lost much details in the light.

(2) Then we observe its histogram: In the interval [50,230], the gray pixels are less than the reference one. However, in the curve [30,50], there are much more pixels than reference image. This phenomenon will cause the whole image looks dark, because there are much dark gray level pixel points than the light points.

(3) The PSNR of this image should be:

$$psnr = 14.77$$

3.3 Best intensity mapping function

After contrast enhancing on two different interval, we can obtain some conclusions:

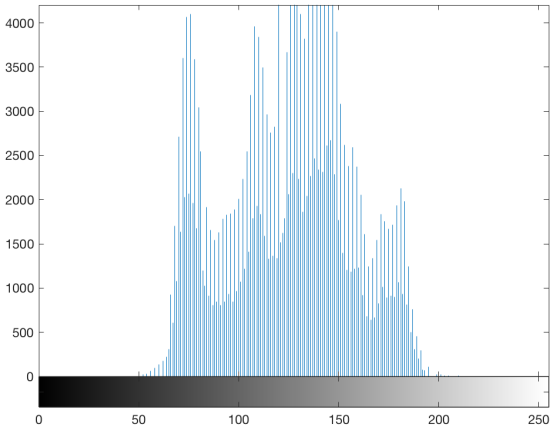
- The best intensity mapping function interval should be [80,130], which include most of significant pixel points. And it will increase the contrast, which will make this image looks much clear, and it will drop the least information and details in this interval.
- After calculate and compare psnr, we can still ensure that interval [80,130] will bring the best intensity mapping function.
- The contrast enhancing can highlights the meaningful message, which are useful for both human and mechine analysis. In addition, it can restrain the useless information, and improve the value of the image. However, the image after enhanced not always necessarily true.

4 Task 4

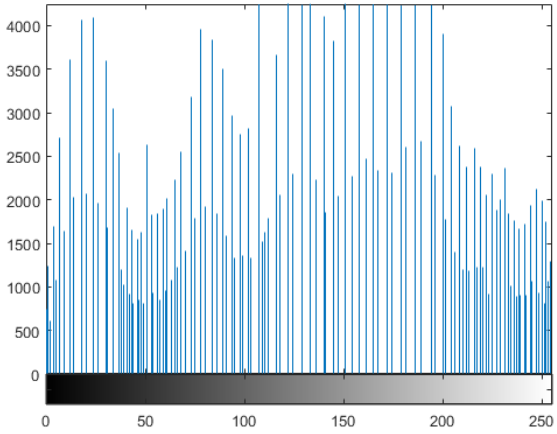
In this task, I will write a function of histogram equalization to enhance the contrast of the image,

(a) Low dynamic range image

(b) histogram equalization image



(c) histogram of low dynamic range



(d) histogram equalization image

Based on these four images, we could draw some important conclusions:

(1) The histogram equalization could enhance the contrast of the image, it makes the original low dynamic image much more colorful and light. The visual contrast of human eyes has also increased based on this enhancement.

(2) According to the histogram of these tow images, we could find that histogram equalization increase the dynamic range of the reference image, from 0 to 255. The gray range of low dynamic range is only from 50 to 200.

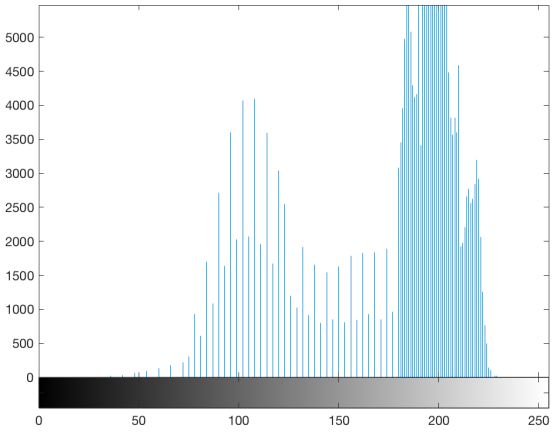
(3) The psnr of this method is:

$$psnr = 15.57$$

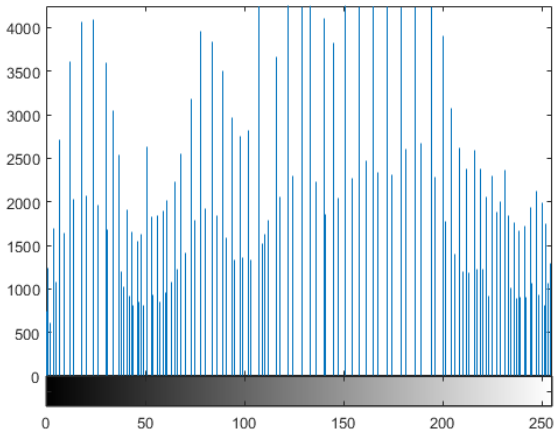
Then we compare the current result with task 3,

(a) piece-wise linear mapping transform

(b) histogram equalization



(c) histogram of piece-wise linear



(d) histogram of histogram equalization

(1) The histogram equalization looks very nature, and it also have a high contrast. But the piece-wise linear mapping transform looks very light, and the contrast of this image is not high enough.

(2) Then we observe their histogram, it is easy to find that the gray dynamic range of piece-wise linear mapping transform not high enough, most of their pixels concentrates on (180,230), so this image looks very light. However, if we observe the histogram of histogram equalization image, we can find that its gray level range average distributed on

(0,255). In another word, the pixels of this images distribute all range, it will not make this image too light or dark. It increase the whole quality of this image.

(3) Therefore, piece-wise linear mapping transform can enhance the contrast of images, but if we choose a unsuitable dynamic range, it will make the image too light or too dark. Histogram equalization can improve the contrast of the whole picture, and make the image looks much more comfortable. However, it could not change the special part of the image. Additional, the histogram equalization can improve the gray level dynamic range.

5 Task 5

5.1 Median filter

We first define a function that can apply median filter, which apply the salt & pepper noise image, in order to reduce the noise.
We first apply the median filter with a 3×3 window:

Figure 5: Salt & Pepper noise image

Figure 6: Median filter 3×3

The PSNR of this image is:

$$psnr = 15.58$$

Then we apply the median filter with 5×5 window:

Figure 7: Salt & Pepper noise image

Figure 8: Median filter 5×5

The PSNR of this image is:

$$psnr = 15.48$$

According to the previous images and psnr, we can obtain some significant conclusions:

(1) The median filter could consider as a technique, which have ability to control and reduce the noise on image effective, based on calculating the mean value of nearby pixel points, and display a new image that includes the new pixel points.

(2) After observe the previous two images, we could find that the image after filter is much more clear and with less noise. However, the image after filter looks too smooth to lose the information. In another word, the noise image after median filter, looks a little white and not nature enough.

(3) Both 3×3 and 5×5 window will play an important role for the median filter. The size of window can decide the final mean value. For example, for 3×3 window, the output pixel will be the median value of the near 3×3 pixel. As for 5×5, the output value will be near 5×5. Thus, the size of window will make a difference on the output pixels.

(4) The noise will be reduced after the median filter, based on calculating the median value of nearby pixels. However, for different window size, the result will be different. For 3×3, we can easy find that the output is better than 5×5, the image is clear and with less noise. In addition, there is still some noise at the edge of the image. This image is not suitable for the window 5×5.

(5) From the psnr of the previous two images, $psnr_{3\times3} > psnr_{5\times5}$, we can sure that window size 3×3 is more suitable for this salt & pepper noise image.

5.2 Average filter

Apply the average filter with 3×3 window, display the image:

Figure 9: Salt & Pepper noise image

Figure 10: Average filter 3×3

$$PSNR_{av-filter} = 16.02$$

After applying these two different type of filters, we can draw the conclusions:

- The average filter, also called linear filter, calculating the average value of the nearby pixels and output the new average pixels. The average filter can effectively suppress additive noise, but it is easy to cause image blur.
- The median filter, also called nonlinear filter, calculating the median value of the nearby pixels, it can make the image much more smooth.
- As for this Salt & Pepper noise image, median filter is much suitable than the average filter, because it can control and reduce the noise effectively, and make this image looks clear. Additional, sharp edges and features in the image are preserved after the median filter.

- The median filter is not sensitivity for the abnormal value, especially for the SP noise. Thus, the median filter can reduce the noise, still with high contrast. However, the median filter is not suitable for line and spike images. It can not reduce the noise at the edge of image.
- As for average filter, it is sensitivity for the abnormal value, also includes the high frequency components. The average filter can filter the high frequency components, it can also reduce the fluctuation of gray level in the image. However, average filer will reduce the contrast and sometimes make image not clear enough.

6 Codes

6.1 Task2

```
im=imread('lenna512.bmp');
im_wn=imnoise(im,'gaussian',0,10);%add gaussian noise
figure,imshow(im_wn);
im_SP=imnoise(im,'salt & pepper',0.1);%add sp noise
figure,imshow(im_SP);
im_low_dynamic_range=imread('lenna512_low_dynamic_range.bmp');
psnr_im_wn=psnr(im_wn,im);%calculate psnr of gaussian
psnr_im_SP=psnr(im_SP,im);%psnr of SP noise
psnr_im_low_dynamic_range=psnr(im_low_dynamic_range,im);%psnr of LDR
figure,imhist(im);
figure,imhist(im_wn);
figure,imhist(im_SP);
figure,imhist(im_low_dynamic_range);
imwrite(im_SP,'im_SP.bmp');
```

6.2 Task 3

```
function dst_img=LinearEnhance(src_img,fa,fb,ga,gb)
[height,width] = size(src_img);%input the height and width
dst_img=uint8(zeros(height,width));
src_img=double(src_img);%change uint8 to double
%Slopes of three linear
k1=ga/fa;
k2=(gb- ga)/(fb- fa);
k3=(255- gb)/(255- fb);
for i=1:height
    for j=1:width
            if src_img(i,j) ≤ fa
               dst_img(i,j)= k1*src_img(i,j);
            elseif fa < src_img(i,j) && src_img(i,j) ≤ fb
                dst_img(i,j)= k2*( src_img(i,j)- fa)+ ga;
            else
                dst_img(i,j)= k3*( src_img(i,j)- fb)+ gb;
            end
    end
```

```
end
dst_img=uint8(dst_img);
```

```
I=imread('lenna512_low_dynamic_range.bmp');
[m,n,k]=size(I);
figure (1)
imshow('lenna512_low_dynamic_range.bmp');title('Original image');
mid=mean(mean(I));%calculate mean value
%horizontal axis
fa=120; fb=180;
%vertical axis
ga=50; gb=200;
J=LinearEnhance(I,fa,fb,ga,gb);
figure (2)
imshow(J);title('After Contrast enhancing');
pixel_f=1:256;
pixel_g=zeros(1,256);

%Three slopes
k1=double(ga/fa);
k2=(gb- ga)/(fb- fa);
k3=(256- gb)/(256- fb);
for i=1:256
    if i ≤ fa
        pixel_g(i)= k1*i;
    elseif fa < i && i ≤ fb
        pixel_g(i)= k2*( i- fa)+ ga;
    else
        pixel_g(i)= k3*( i - fb)+ gb;
    end
end
figure (3)
plot(pixel_f,pixel_g);
title('Piecewise-Linear Transformation');
PSNR_linear=psnr(J,I);
figure (4),imhist(I);
figure (5), imhist(J);
```

6.3 Task 4

```
I=imread('lenna512_low_dynamic_range.bmp');
[m,n,k]=size(I);
figure (1)
imshow('lenna512_low_dynamic_range.bmp');title('Original image');
mid=mean(mean(I));%calculate mean value
%horizontal axis
fa=120; fb=180;
%vertical axis
ga=50; gb=200;
J=LinearEnhance(I,fa,fb,ga,gb);
figure (2)
imshow(J);title('After Contrast enhancing');
pixel_f=1:256;
pixel_g=zeros(1,256);

%Three slopes
k1=double(ga/fa);
k2=(gb- ga)/(fb- fa);
k3=(256- gb)/(256- fb);
for i=1:256
    if i ≤ fa
        pixel_g(i)= k1*i;
    elseif fa < i && i ≤ fb
        pixel_g(i)= k2*( i- fa)+ ga;
    else
        pixel_g(i)= k3*( i - fb)+ gb;
    end
end
figure (3)
plot(pixel_f,pixel_g);
title('Piecewise-Linear Transformation');
PSNR_linear=psnr(J,I);
figure (4),imhist(I);
figure (5), imhist(J);
```

6.4 Task 5

6.4.1 Median filter

```
function d=mid_filter(x,n)
[height, width]=size(x);   %input image,p>n,q>n
x1=double(x);  %change data type to double
x2=x1;
for i=1:height-n+1
    for j=1:height-n+1
        c=x1(i:i+(n-1),j:j+(n-1)); %from(i,j),n row, n column
        e=c(1,:);      %matrix first row
        for u=2:n
            e=[e,c(u,:)];
        end
        mm=median(e);      %take the median value
        x2(i+(n-1)/2,j+(n-1)/2)=mm;   %exchange the pixel with new ...
            median value
    end
end
d=uint8(x2);
```

```
a=imread('im_SP.bmp');%load the image
figure,imshow(a);
b=3;%define the size of window
mid=mid_filter(a,b);
figure,imshow(mid);
PSNR_mid=psnr(mid,a);%calculate the psnr
```

6.4.2 Average filter

```
function d=av_filter(x,n)
a(1:n,1:n)=1;   %a=n n matrix
[hight, width]=size(x);   %input matrix, and hight>n,width>n
x1=double(x);
x2=x1;
for i=1:hight-n+1
    for j=1:width-n+1
        c=x1(i:i+(n-1),j:j+(n-1)).*a; %take the elements from c, ...
            times with a
        s=sum(sum(c));                %sum all elements in c
        x2(i+(n-1)/2,j+(n-1)/2)=s/(n*n); %exchange the pixels with ...
            new calculation results
    end
end
d=uint8(x2);
```

```
a=imread('im_SP.bmp');%load image
figure,imshow(a);
b=3;%defien the size of window
av=av_filter(a,b);%call function
figure,imshow(av);%show image after average filter
PSNR_av=psnr(av,a);%calculate psnr
```