EE211A Digital Image Processing
Winter 2018
Homework 1
(Deadline: 01/29/2018 6 pm)

YAO XIE
UID: 105036239

Problem 1 Implement your own function that performs a convolution operation between a grayscale image and a filter. a) Using your convolution function, filter the test image with Sobel filter. Explain your observations. b) Using your convolution function, filter the test image with Gaussian filter for three different σ (σ = 1, σ=5, σ=15). Explain your observations.

Results:

(a):

This part is to apply Sobel filter to process the image. There're several methods to deal with the boarders. In this task, I flipped the image firstly in both right and left directions and then flipped the image we obtained upside and downside, as shown in Fig. 1. So the image is actually 9 times as large as the original one.

Then I convolve the image with x-sobel filter and y-sobel filter, respectively. It can be known that in the process of x-sobel-filter convolution, the outlines of the image in x direction are stressed (Fig 3.) while in the y-process, the outlines in y direction are stressed, too (Fig 4.). Finally I add them together and after normalization we can know that the process of Sobel-filter convolution can extract the outlines of the image. Thus Sobel filter is mostly used in edge detection.

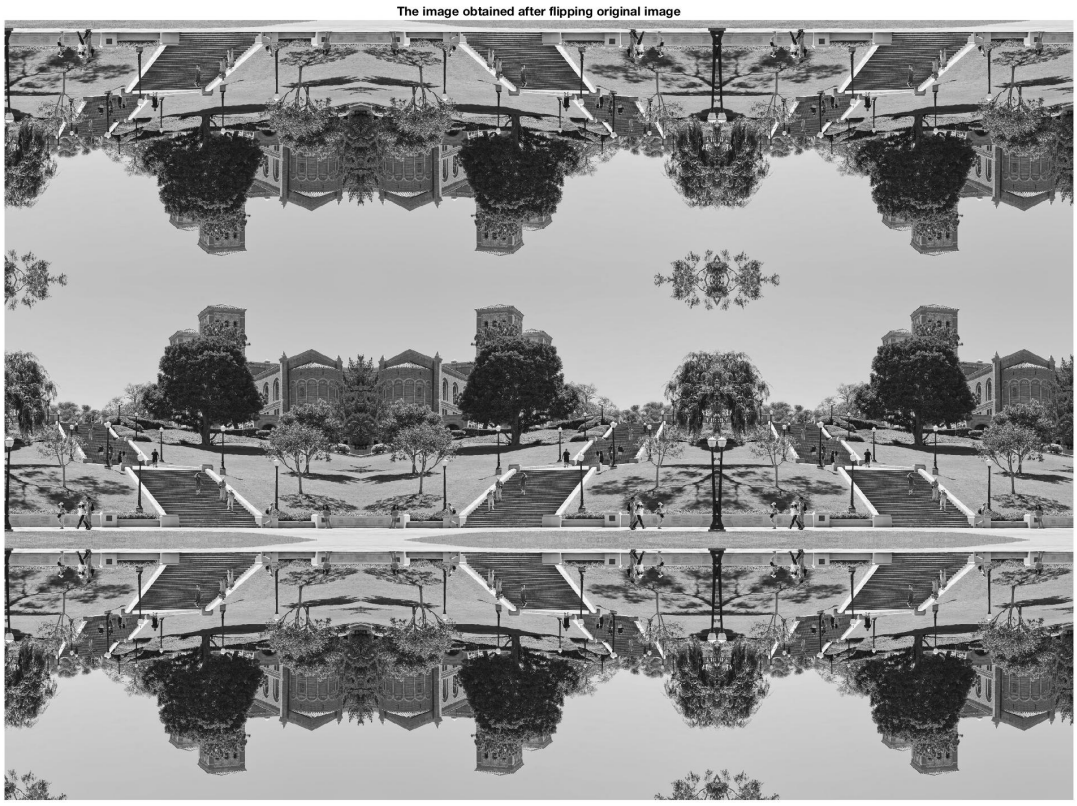

Fig. 1. The image obtained after flipping the original image

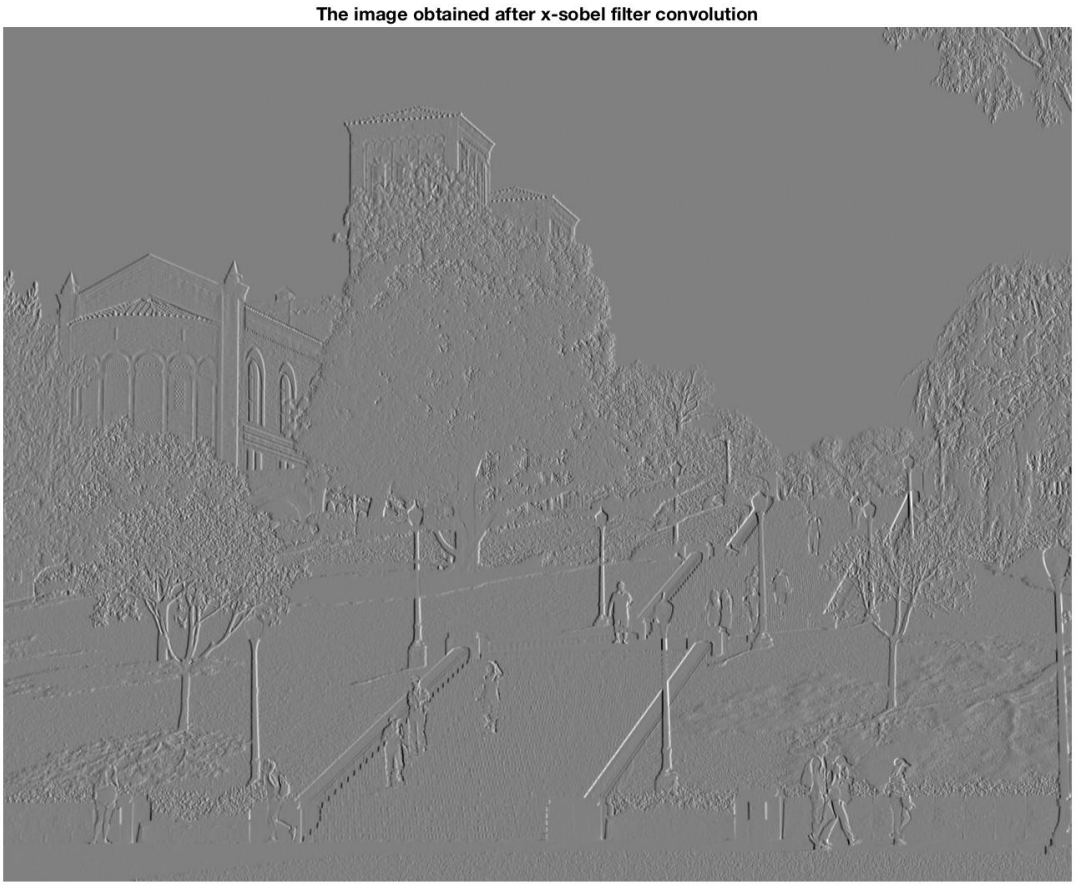

Fig. 2. The image obtained after x-sobel filter convolution

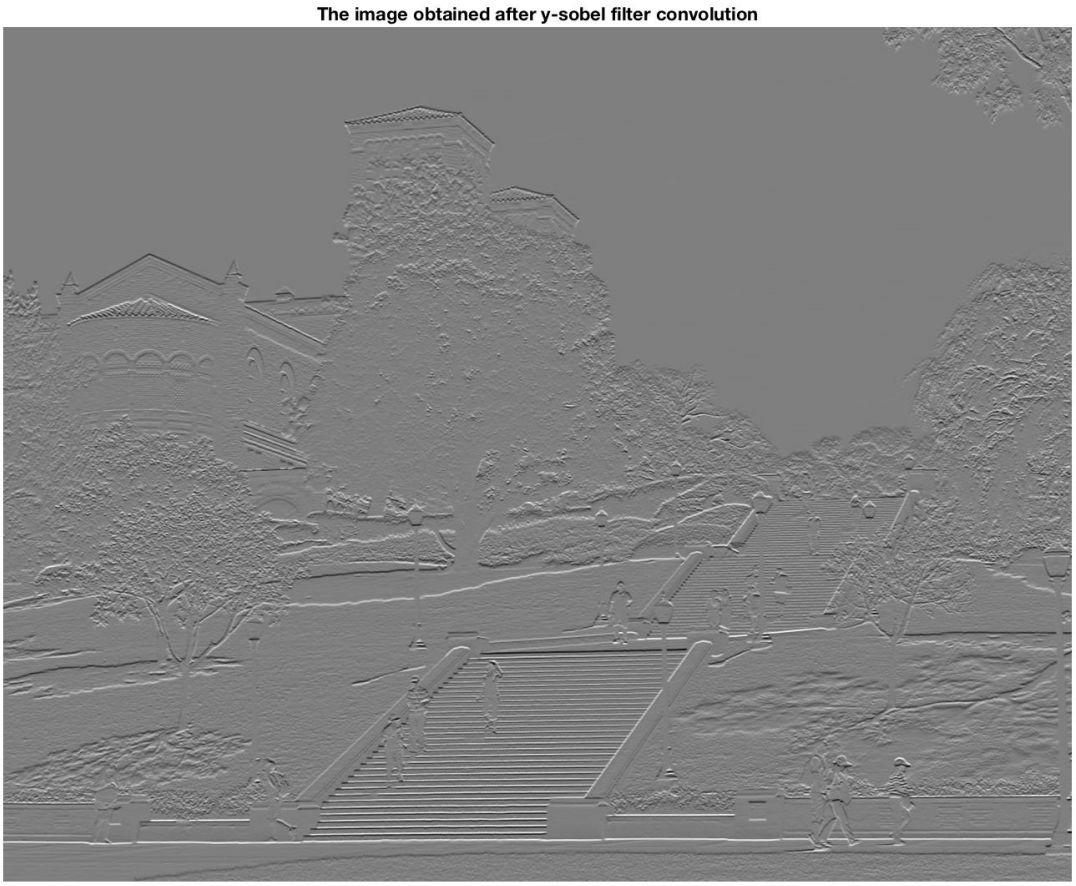

Fig. 3. The image obtained after y-sobel filter convolution

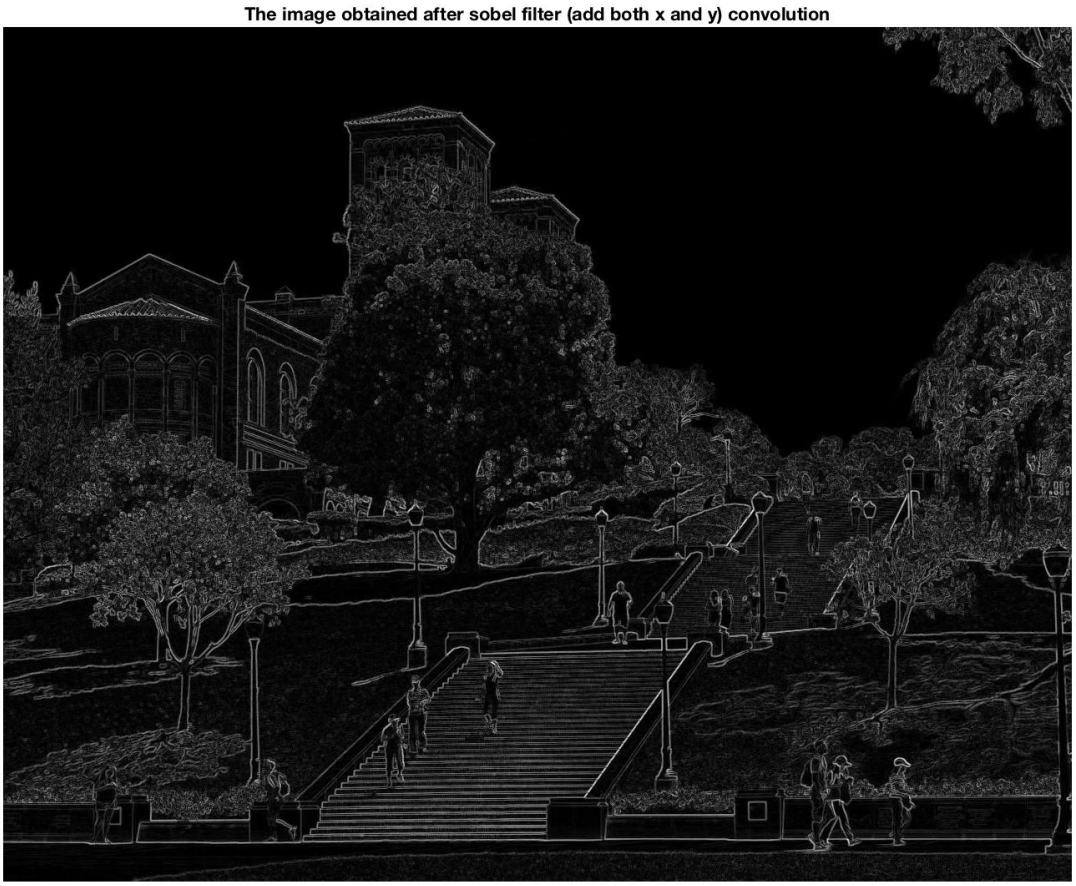

Fig. 4. The image obtained after sobel filter (add both x and y) convolution

Code:

```
pgm_name = 'Test_image.pbm';
data = imread(pgm_name);
data = im2double(data);
figure(1)
imshow(data);
[m n] = size(data);
sobel_x = [-1 0 1;-2 0 2;-1 0 1];
sobel_y = [-1 -2 -1;0 0 0;1 2 1];
data_1 = [data(:,n:-1:1) data(:,:) data(:,n:-1:1)];
newdata = [data_1(m:-1:1,:);data_1(:,:);data_1(m:-1:1,:)];
imshow(newdata);
title('The image obtained after flipping original image')
%sobel_x
data_sobel_x = [];
for i = 1:m
    for j = 1:n
            data_sobel_x(i,j) = (-1)*newdata(m+i-1,n+j-1)+1*newdata(m+i-
1,n+j+1)+(-2)*newdata(m+i,n+j-1)+2*newdata(m+i,n+j+1)+(-1)*newdata(m+i+1,n+j-
1)+1*newdata(m+i+1,n+j+1);
```

```
    end
end
figure(2)
imshow(data_sobel_x,[]);
title('The image obtained after x-sobel filter convolution')
%sobel_y
data_sobel_y = [];
for i = 1:m
    for j = 1:n
            data_sobel_y(i,j) = (-1)*newdata(m+i-1,n+j-1)+(-2)*newdata(m+i-
1,n+j)+(-1)*newdata(m+i-1,n+j+1)+1*newdata(m+i+1,n+j-
1)+2*newdata(m+i+1,n+j)+1*newdata(m+i+1,n+j+1);
    end
end
figure(3)
imshow(data_sobel_y,[]);
title('The image obtained after y-sobel filter convolution')
figure(4)
imshow(abs(data_sobel_x)+abs(data_sobel_y),[]);
title('The image obtained after sobel filter (add both x and y) convolution')
```

(b):

In this part we filter the image with Gaussian filter and the 3 values of standard deviations are tried (σ = 1, σ=5, σ=15). In order to deal with the boarders, I did the same flipping steps as part (a), shown in Fig 5.
From the images obtained after the convolution, it is obvious that the Gaussian filter process can blur the images. In addition, the value of σ (standard deviation) can also effect the results. The larger the σ, the more the Gaussian filter blurs the image. Moreover, the size of the Gaussian filter I apply in this process varies according to the value of σ. The size of the filter is usually 5σ or 6σ. In this problem I choose the sizes as 5*5, 29*29 and 225*225, respectively, which correspond to the σ value of 1, 5 and 15.

Fig. 4. The original image

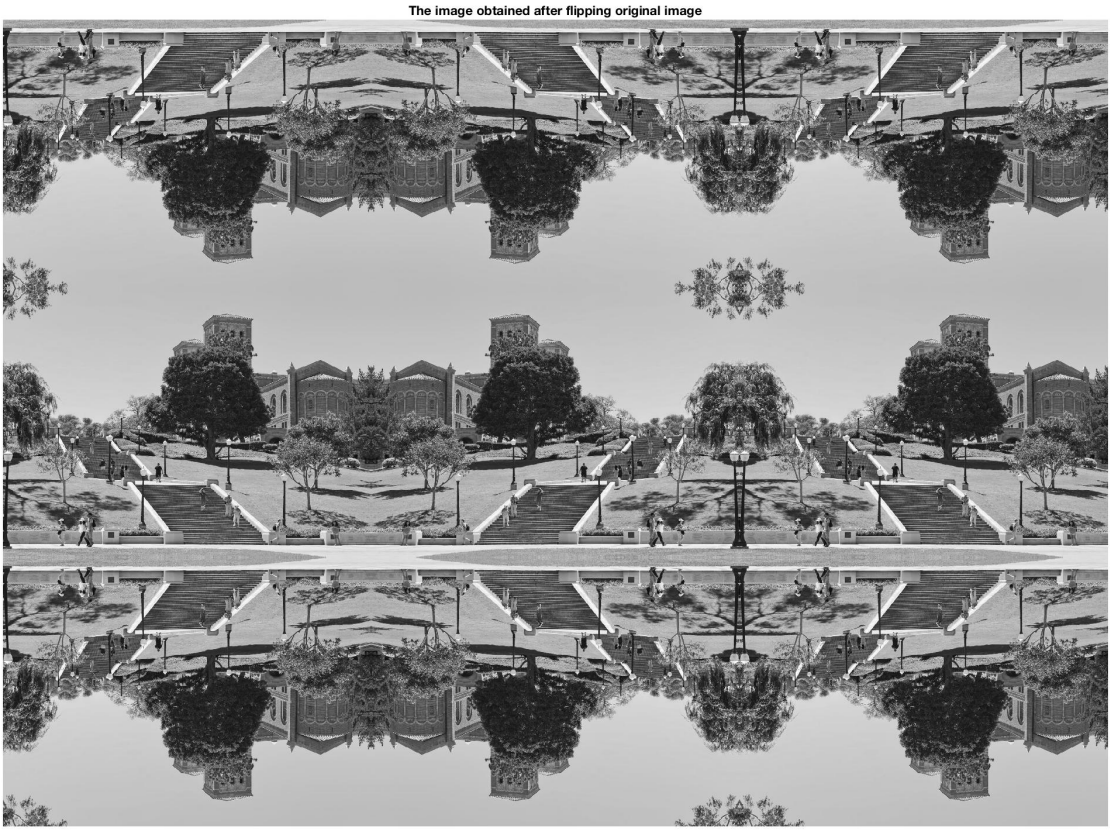

Fig. 5. The image obtained after flipping the original image

Fig. 6. The image obtained after convolution with gaussian filter when signma=1

Fig. 7. The image obtained after convolution with gaussian filter when signma=5

Fig. 8. The image obtained after convolution with gaussian filter when signma=15

Code:

```
pgm_name = 'Test_image.pbm';
data = imread(pgm_name);
data = im2double(data);
figure(1)
imshow(data);
title('The original image')
[m n] = size(data);
%flip the image right and left
data_1 = [data(:,n:-1:1) data(:,:) data(:,n:-1:1)];
%flip the image up and down
newdata = [data_1(m:-1:1,:);data_1(:,:);data_1(m:-1:1,:)];
%generate gaussian filter with sigma=1,size=5*5
gaussian_1=[];
for i=1:5
    for j=1:5
        gaussian_1(i,j) = 1/(2*pi)*exp(-((i-3)^2+(j-3)^2)/2);
    end
end
%generate gaussian filter with sigma=5,size=29*29
gaussian_5=[];
```

```
for i=1:29
    for j=1:29
        gaussian_5(i,j) = 1/(50*pi)*exp(-((i-15)^2+(j-15)^2)/50);
    end
end

%generate gaussian filter with sigma=15,size=225*225
gaussian_15=[];
for i=1:225
    for j=1:225
        gaussian_15(i,j) = 1/(450*pi)*exp(-((i-113)^2+(j-113)^2)/450);
    end
end
%gaussian sigma=1
data_1=[];
for i = 1:m
    for j = 1:n
        conmatrix_1 = newdata(m+i-2:m+i+2,n+j-2:n+j+2).*gaussian_1;
        data_1(i,j) = sum(conmatrix_1(:));
    end
end
figure(2)
imshow(data_1);
title('The image obtained after convolution with gaussian filter, sigma=1')
%gaussian sigma=5
data_5=[];
for i = 1:m
    for j = 1:n
        conmatrix_5 = newdata(m+i-14:m+i+14,n+j-14:n+j+14).*gaussian_5;
        data_5(i,j) = sum(conmatrix_5(:));
    end
end
figure(3)
imshow(data_5);
title('The image obtained after convolution with gaussian filter, sigma=5')
%gaussian sigma=15
data_15=[];
for i = 1:m
    for j = 1:n
        conmatrix_15 = newdata(m+i-112:m+i+112,n+j-112:n+j+112).*gaussian_15;
        data_15(i,j) = sum(conmatrix_15(:));
    end
end
figure(4)
imshow(data_15);
title('The image obtained after convolution with gaussian filter, sigma=15')
```

Problem 2 Implement your own function that performs bilateral filtering to a grayscale image. Using your own function, perform bilateral filtering operation on the test image for the following σs and σr values. Explain your observations.

σ_s =3,σ_r =0.1	σ_s =3,σ_r =0.3	σ_s =3,σ_r =10
σ_s = 10, σ_r =0.1	σ_s=10,σ_r =0.3	σ_s=10,σ_r =10
σ_s=25,σ_r =0.1	σ_s=25,σ_r =0.3	σ_s=25,σ_r =10

Result:
In this part I tried 9 combinations of different σ_s and σ_r. In this task I choose the filter sizes as 6 σ_s. Thus, when σ_s equals to 3, the filter is 17*17 (not 18 in order to make the calculation of mid point more convenient). When σ_s equals to 10, the filter is 59*59 and when σ_s equals to 25, the filter is 149*149.

The results after convolution with such filters are shown below:

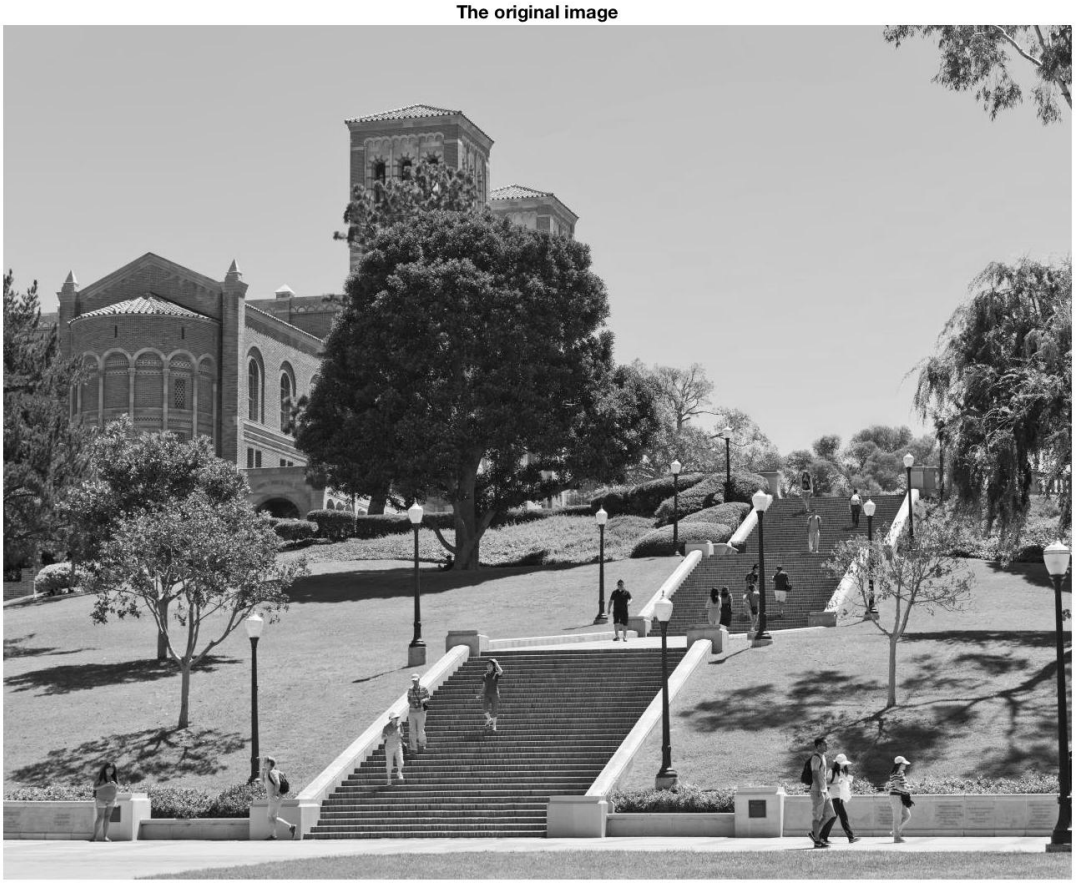

Fig 9. The original image

Fig 10. Bilateral filter when $\sigma_s = 3$, $\sigma_r = 0.1$

Fig 11. Bilateral filter when $\sigma_s = 3$, $\sigma_r = 0.3$

Fig 12. Bilateral filter when $\sigma_s = 3$, $\sigma_r = 10$

Fig 13. Bilateral filter when $\sigma_s = 10$, $\sigma_r = 0.1$

Fig 14. Bilateral filter when $\sigma_s = 10,\ \sigma_r = 0.3$

Fig 15. Bilateral filter when $\sigma_s = 10,\ \sigma_r = 10$

Fig 16. Bilateral filter when $\sigma_s = 25$, $\sigma_r = 0.1$

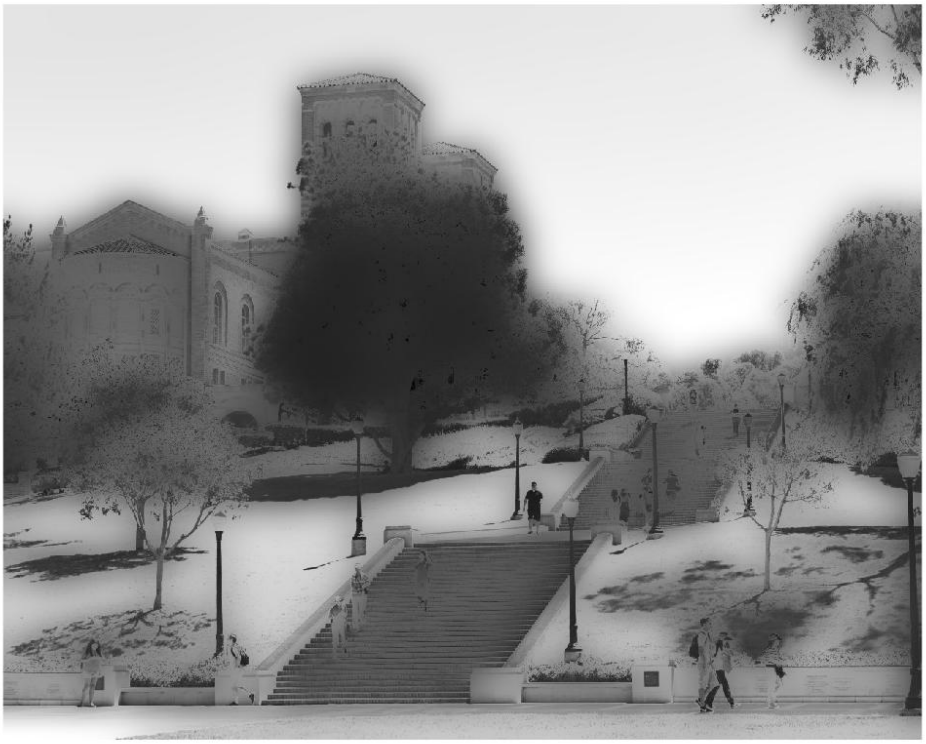

Fig 17. Bilateral filter when $\sigma_s = 25,\ \sigma_r = 0.3$

Fig 18. Bilateral filter when $\sigma_s = 25,\ \sigma_r = 10$

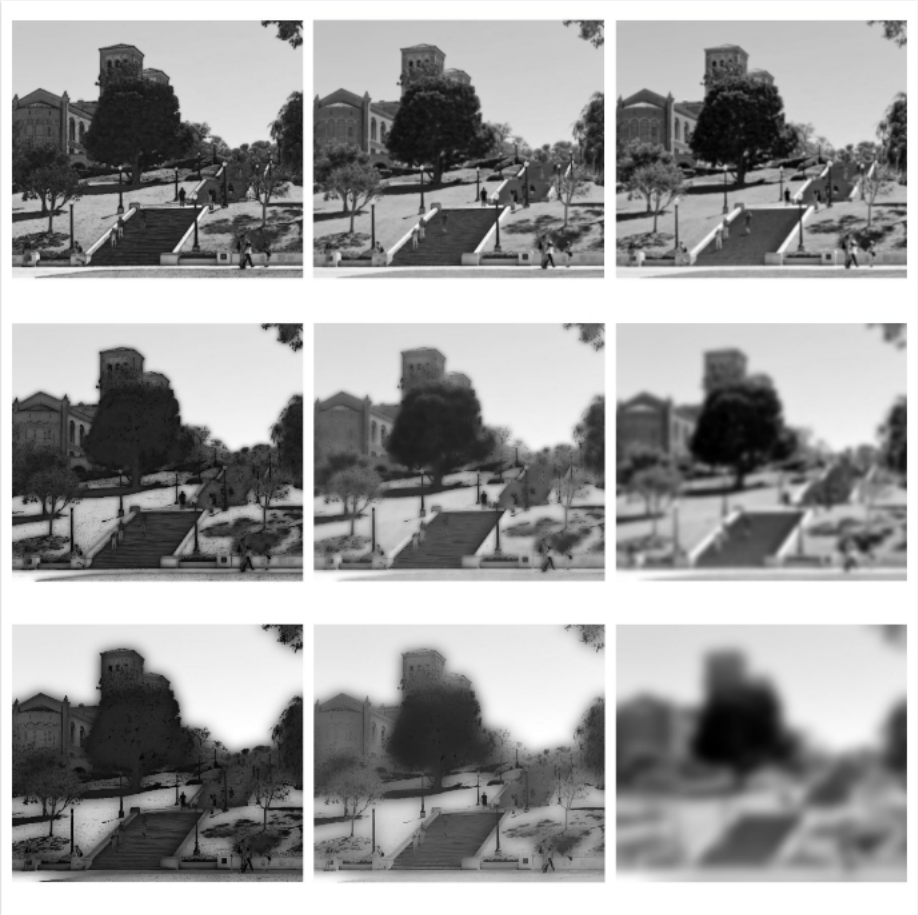

Fig 19. All images corresponding to the order shown in the table

Observation:

From the images we obtained, it can be known that the images after convolution are different according to the values of σs and σr. When is σr and σs are getting larger, the image will be more obscure. If σr is fixed, the image will be darker and more obscure while σs increases. If is σs fixed, the image will be lighter and more obscure while σr increases.
From the equation, we can know that σs influences the Gaussian convolution in geometric aspect, the further a pixel is to the chosen point, the less that filter effects it. Thus we can say that the special Gaussian filter only has obvious impact on pixels near the chosen pixel.
Besides, σr influences the Gaussian convolution in pixel-value difference aspect, the larger the difference between one pixel value to a chosen point's pixel value, the less that filter effects it. Thus we can say that the r-Gaussian filter only has obvious impact on pixels who has similar pixel value as the chosen point's pixel value.

Code:

```
pgm_name = 'Test_image.pbm';
data = imread(pgm_name);
data = im2double(data);
figure(1)
imshow(data);
title('The original image')
[m n] = size(data);
%flip the image right and left
data_1 = [data(:,n:-1:1) data(:,:) data(:,n:-1:1)];
%flip the image up and down
newdata = [data_1(m:-1:1,:);data_1(:,:);data_1(m:-1:1,:)];
%sigma_s = [3 10 25];  %s=3, size=17*17, s=10, size=59*59, s=25, size=149*149
%sigma_r
sigma_r = [0.1 0.3 10];
%generate gaussian filter with sigma_s=3,size=17*17
gaussian_s1=[];
for i=1:17
   for j=1:17
       gaussian_s1(i,j) = 1/(2*pi*9)*exp(-((i-9)^2+(j-9)^2)/(2*9));
   end
end
gaussian_s1=(gaussian_s1)./gaussian_s1(9,9);
%generate gaussian filter with sigma_s=10,size=59*59%gaussian_s2=[]
for i=1:59
   for j=1:59
       gaussian_s2(i,j) = 1/(2*pi*100)*exp(-((i-30)^2+(j-30)^2)/200);
   end
end
gaussian_s2=(gaussian_s2)./gaussian_s2(30,30);
%generate gaussian filter with sigma_s=25,size=17*17
gaussian_s3=[];
for i=1:149
   for j=1:149
       gaussian_s3(i,j) = 1/(2*pi*625)*exp(-((i-75)^2+(j-75)^2)/(2*625));
   end
end
gaussian_s3=(gaussian_s3)./gaussian_s3(75,75);
%%gaussian sigma_r with filter s1,size=17*17
for a = 1:3
    data_1 = [];
    r = sigma_r(a);
    for i = 1:m
        for j = 1:n
            for p = 1:17
                for q = 1:17
                    gaussian_r1(p,q) = 1/(2*pi*r^2)*exp(-(newdata(m+i,n+j)-
newdata(m+i+p-9,n+j+q-9))^2/(2*r^2));
                end
            end
            A = gaussian_s1.*gaussian_r1.*newdata(m+i-8:m+i+8,n+j-8:n+j+8);
            data_1(i,j)=sum(A(:));
        end
    end
    %data_1 = data_1./max(max(data_1));
```

```
    figure
    imshow(data_1,[])
end
%%gaussian sigma_r with filter s2,size=59*59
for a = 1:3
    data_2 = [];
    r = sigma_r(a);
    for i = 1:m
        for j = 1:n
            for p = 1:59
                for q = 1:59
                    gaussian_r2(p,q) = 1/(2*pi*r^2)*exp(-(newdata(m+i,n+j)-
newdata(m+i+p-30,n+j+q-30))^2/(2*r^2));
                end
            end
            A = gaussian_s2.*gaussian_r2.*newdata(m+i-29:m+i+29,n+j-
29:n+j+29);
            data_2(i,j)=sum(A(:));
        end
    end
    %data_2 = data_2./max(max(data_2));
    figure
    imshow(data_2,[])
end
%%gaussian sigma_r with filter s3,size=149*149
for a = 1:3
    data_3 = [];
    r = sigma_r(a);
    for i = 1:m
        for j = 1:n
            for p = 1:149
                for q = 1:149
                    gaussian_r3(p,q) = 1/(2*pi*r^2)*exp(-(newdata(m+i,n+j)-
newdata(m+i+p-75,n+j+q-75))^2/(2*r^2));
                end
            end
            A = gaussian_s3.*gaussian_r3.*newdata(m+i-74:m+i+74,n+j-
74:n+j+74);
            data_3(i,j)=sum(A(:));
        end
    end
    %data_3 = data_3./max(max(data_3));
    figure
    imshow(data_3,[])
end
```