TONGJI UNIVERSITY
SCHOOL OF SOFTWARE ENGINEERING

Homework #1

Student: *Le Yang 1751894*

Course: *Digital Image Processing (42033501)* – Professor: *Dr. Qingjiang Shi*
Due date: *March 28th, 2025*

1. Denosing for Astrophotography

a) To generate a single denoised image from each video, compute a running average of the frames f^t(t = 1, 2, ...) in the video without frame alignment, according to the following update rule:

$$f^1_{average} = f^1$$
$$f^1_{average} = \frac{t-1}{t} f^{t-1}_{average} + \frac{1}{t} f^t$$

Display and submit $f^t_{average}$ at $t = 30$ for each video. Comment on how effectively the noise is reduced and how much the sharp features are blurred by the averaging operation.

Figure 1: origin 30rd frame of sky_1

Figure 2: origin 30rd frame of sky_2

Answer: First, I generate the two origin 30rd frames of the two videos for comparision.The corresponding Matlab code is **origin.m**. This code generate two images: **sky1origin_30.jpg, sky2origin_30.jpg**. They are shown as **Figure 1** and **Figure 2**.
Then I use the update rule in the question to denoise the image. My Matlab code is followed and is saved as **without_align.m**, this code generate two $f^t_{average}$ at 30rd frame of each video which are shown as **Figure 3** and **Figure 4** below.

```
%without_align.m
clc,clear;
vidobj_1=VideoReader("hw1_sky_1.avi");
numFrames_1=vidobj_1.NumberOfFrames;

vidobj_2=VideoReader("hw1_sky_2.avi");
numFrames_2=vidobj_2.NumberOfFrames;

for i=1:numFrames_1
    frame_1=im2double(read(vidobj_1,i));
    image_name_1=strcat('result\sky1woalign_',num2str(i));
    image_name_1=strcat(image_name_1,'.jpg');

    frame_2=im2double(read(vidobj_2,i));
    image_name_2=strcat('result\sky2woalign_',num2str(i));
    image_name_2=strcat(image_name_2,'.jpg');
    if(i==1)
```

```
            f_average_1=frame_1;
            f_average_2=frame_2;
        else
            f_average_1=(i-1)/i*f_average_1+frame_1/i;
            f_average_2=(i-1)/i*f_average_2+frame_2/i;
        end

        if(i==30)
            woalign_1=f_average_1;
            imwrite(woalign_1,image_name_1,'jpg');
            woalign_2=f_average_2;
            imwrite(woalign_2,image_name_2,'jpg');
            break;
        end
    end
```

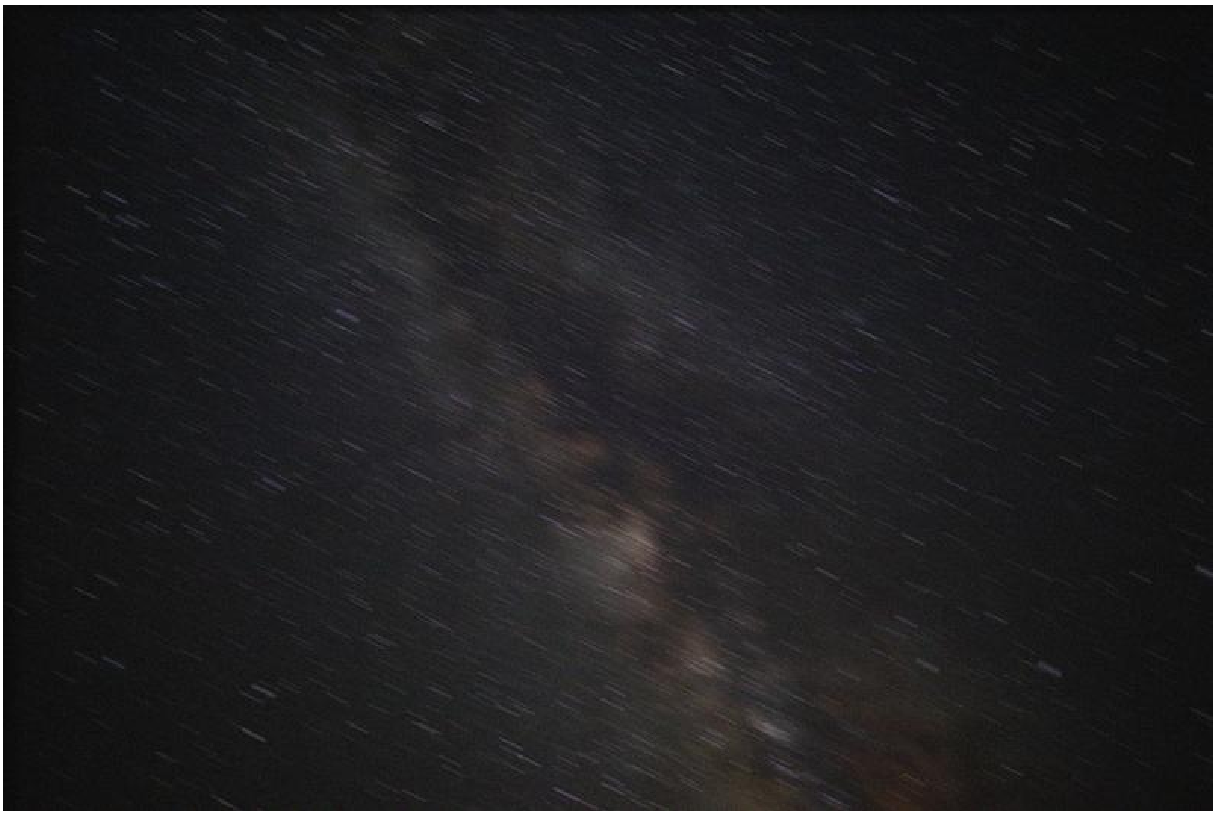

Figure 3: $f^t_{average}$ at 30rd frame of sky_1

Figure 4: $f^t_{average}$ at 30rd frame of sky_2

From the results above, we can see that we reduces the noise in the video to some extent by computing the running average of the frames. It is more easy to recognize in the sky2 video with the moon in the sky because the noise spots mix with stars in the sky1 video. Compared to **Figure 2**, we can see that in **Figure 4**, all the noise spots around the moon are removed. However, because we don't utilize frame alignment, we can that many sharp features are blurred by the averaging operation like the edges of the moon in sky2 and stars in sky1. As a result, it is hard to see some details.

b) Now, compute a running average of the frames with frame alignment, according to the following update rule:

$$f^1_{average} = f^1$$
$$f^1_{average} = \frac{t-1}{t} f^{t-1}_{average} + \frac{1}{t} Align(f^t, f^{t-1}_{average}, t = 2, 3, \ldots$$

Display and submit $f^t_{average}$ at $t = 30$ for each video. Compare to the result in (a) and comment on how effectively noise is reduced while sharp features are better preserved.

Answer: I use the update rule in the question to denoise the image with frame alignment. My Matlab code is followed and is saved as **with_align.m**, this code generate two $f^t_{average}$ at 30rd frame of each video which are shown as **Figure 5** and **Figure 6** below.

```
%with_align.m
clc,clear;
vidobj_1=VideoReader("hw1_sky_1.avi");
numFrames_1=vidobj_1.NumberOfFrames;

vidobj_2=VideoReader("hw1_sky_2.avi");
numFrames_2=vidobj_2.NumberOfFrames;

for i=1:numFrames_1
    frame_1=im2double(read(vidobj_1,i));
    image_name_1=strcat('result\sky1wialign_',num2str(i));
    image_name_1=strcat(image_name_1,'.jpg');
    frame_2=im2double(read(vidobj_2,i));
    image_name_2=strcat('result\sky2wialign_',num2str(i));
    image_name_2=strcat(image_name_2,'.jpg');
    if(i==1)
        f_average_1=frame_1;
        f_average_2=frame_2;
    else
        f_average_1=(i-1)/i*f_average_1+Align(frame_1,f_average_1)/i;
        f_average_2=(i-1)/i*f_average_2+Align(frame_2,f_average_2)/i;
    end

    if(i==30)
        wialign_1=f_average_1;
        imwrite(wialign_1,image_name_1,'jpg');
        wialign_2=f_average_2;
        imwrite(wialign_2,image_name_2,'jpg');
        break;
    end
end
```

```
%Align.m
function aligned = Align(f,g)
%ALIGN
%
search = 10;
minMSE = Inf;
[height, width, channels] = size(f);
for dx = -search:search
    for dy = -search:search
        A=[1 0 0
        0 1 0
        dx dy 1];
        tform = maketform('affine', A);
        frameTform = zeros(size(f));

        frameTform = imtransform(f, tform,'bilinear','XData', [1 ...
            width], 'YData', [1 height], 'FillValues', ...
            zeros(channels,1));


```

```
                % Calculates the MSE
                mse = norm( reshape(frameTform, [], 1) - reshape(g, [], 1));

                if(mse < minMSE)
                minMSE = mse;
                aligned = frameTform;

                end
            end
        end
        end
```

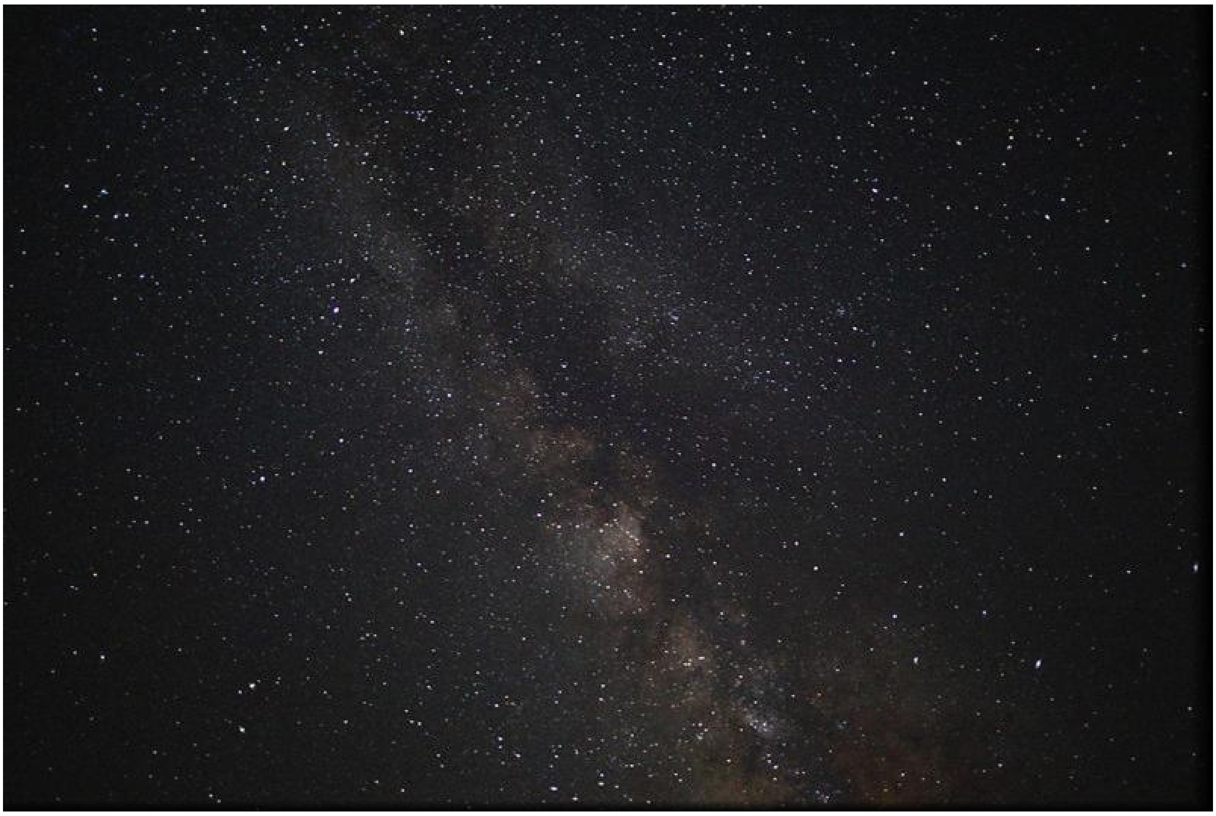

Figure 5: $f^t_{average}$ at 30rd frame of sky_1

Figure 6: $f^t_{average}$ at 30rd frame of sky_2

From the results above, we can see that with frame alignment the bluriness is reduced and we also remove the noise effectively.

2. Nighttime Road Contrast Enhancement

a) Plot and submit the histogram (MATLAB function: imhist) of the original images grayscale values. Briefly comment on the shape of each histogram.

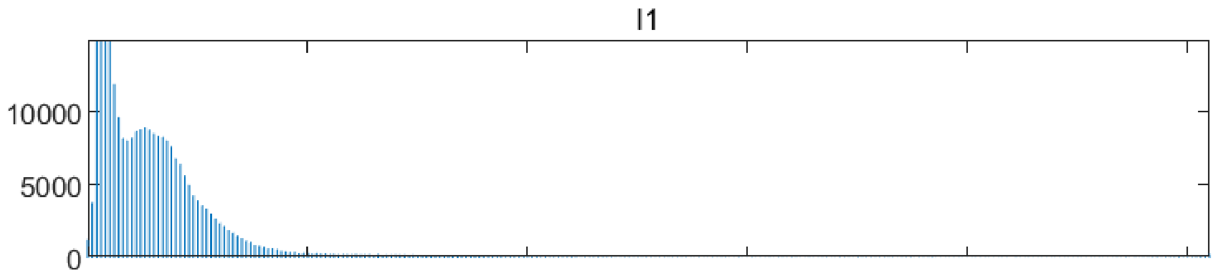

Figure 7: histgram of road1

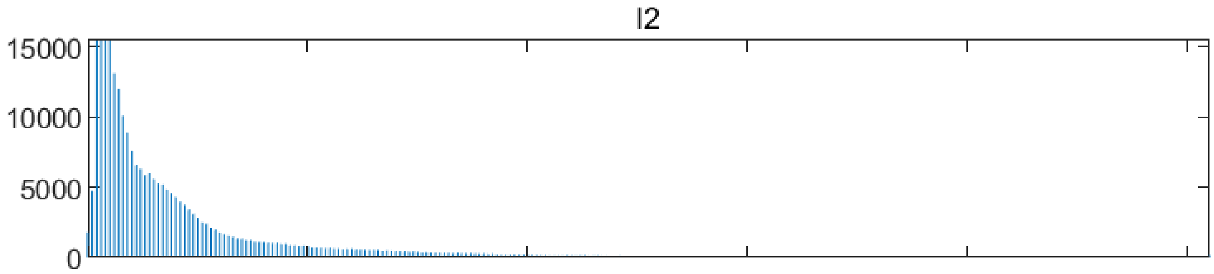

Figure 8: histgram of road2

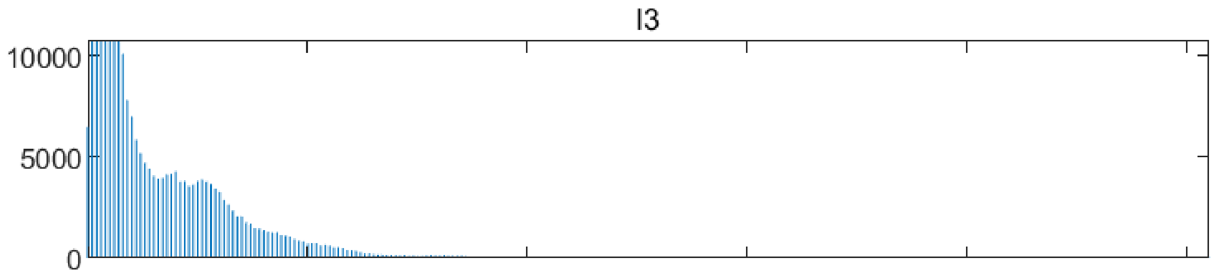

Figure 9: histgram of road3

Answer. The three images above are the histograms of the three origin images grayscale values. We can see that the contrast of the picture is very low, The components of the histogram are concentrated at the low (dark) end of the gray level, and the width is narrow, and the distribution of the pixels is uneven.
In the histograms of road1 the grayscale values are concentrated at the low(dark) levels. There are two peaks.
In the histograms of road2 the grayscale values are concentrated at the low(dark) levels. There is only one peak.
In the histograms of road3 the grayscale values are concentrated at the low(dark) levels. There are one top peak and a few short peaks.
My Matlab code is followed and is saved as **a.m**.

```
clc,clear;
I1=imread("hw1_dark_road_1.jpg");
I2=imread("hw1_dark_road_2.jpg");
I3=imread("hw1_dark_road_3.jpg");
figure('name','histogram','NumberTitle','off');
subplot(3,1,1);
imhist(I1);          %display the origin image
title("I1");
subplot(3,1,2);
imhist(I2);          %display the origin hist
title("I2")
subplot(3,1,3);
imhist(I3);          %display the origin hist
title("I3")
```

b) Apply global histogram equalization to the original image(MATLAB function: histeq is not allowed; just implement it by yourself and compare yours with Matlabs histeq). Display and submit the modified image. Plot and submit the histogram of the modified images grayscale values. Comment on visually desirable/undesirable regions in the modified image.

Figure 10: image of road1 after apply global histogram equalization

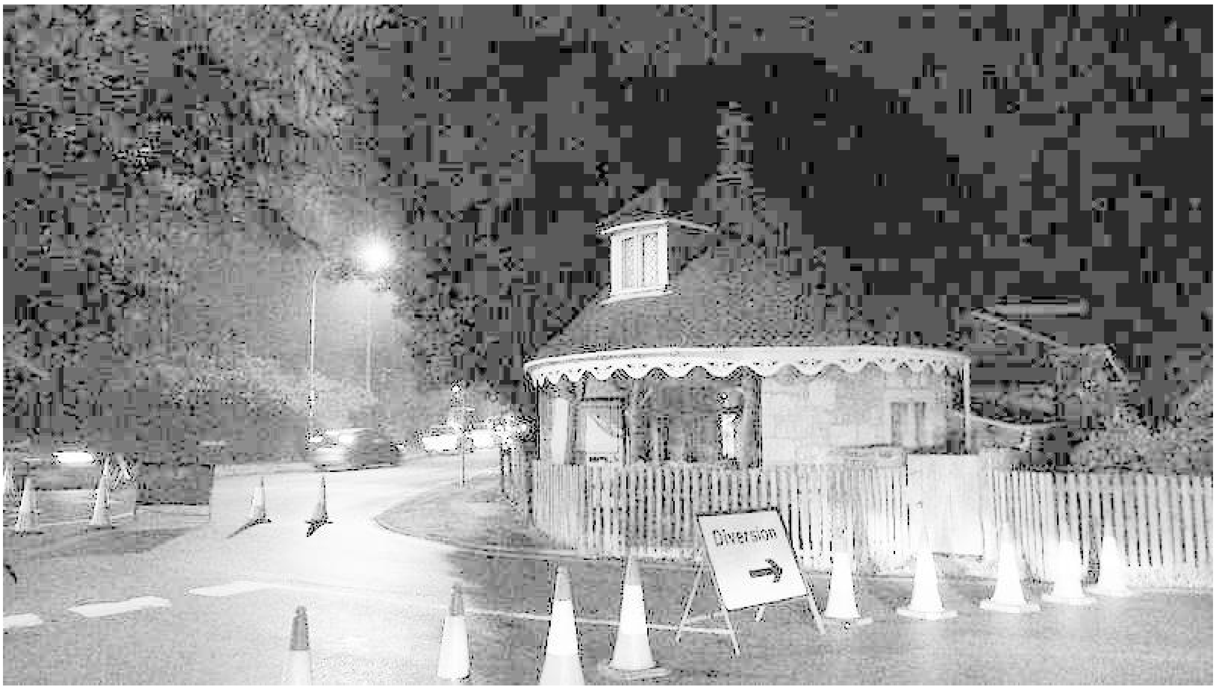

Figure 11: image of road2 after apply global histogram equalization

Figure 12: image of road3 after apply global histogram equalization

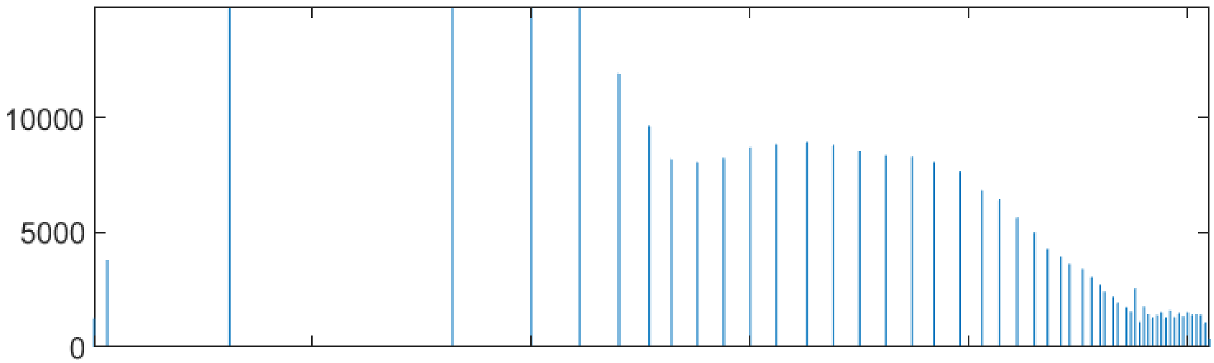

Figure 13: histogram of road1 after apply global histogram equalization

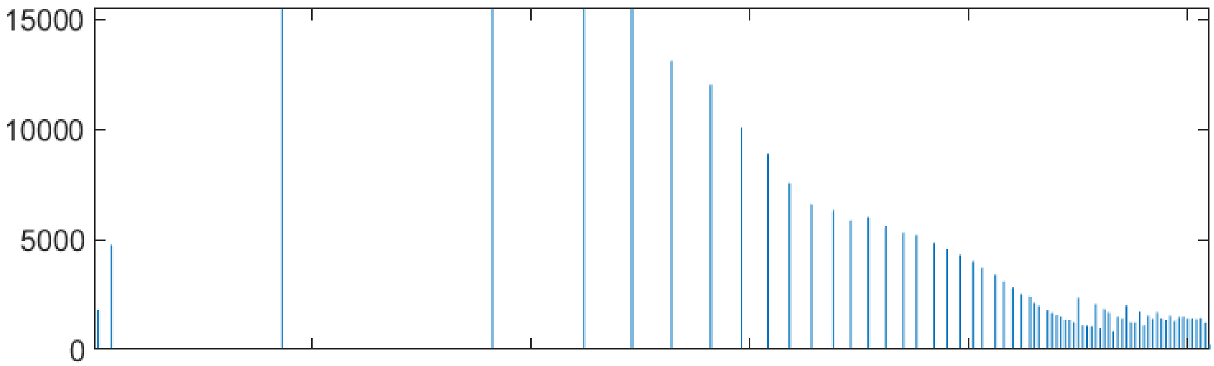

Figure 14: histogram of road2 after apply global histogram equalization

Figure 15: histogram of road3 after apply global histogram equalization

Figure 16: image of road1 after apply histeq function

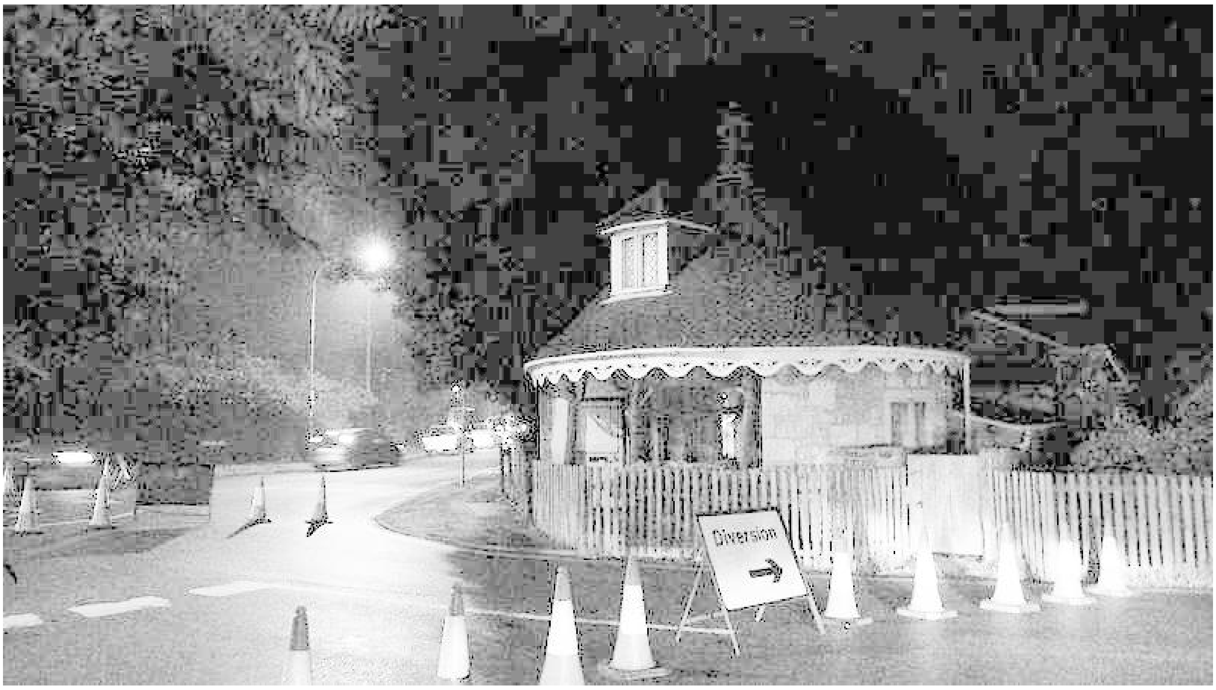

Figure 17: image of road2 after apply histeq function

Figure 18: image of road3 after apply histeq function

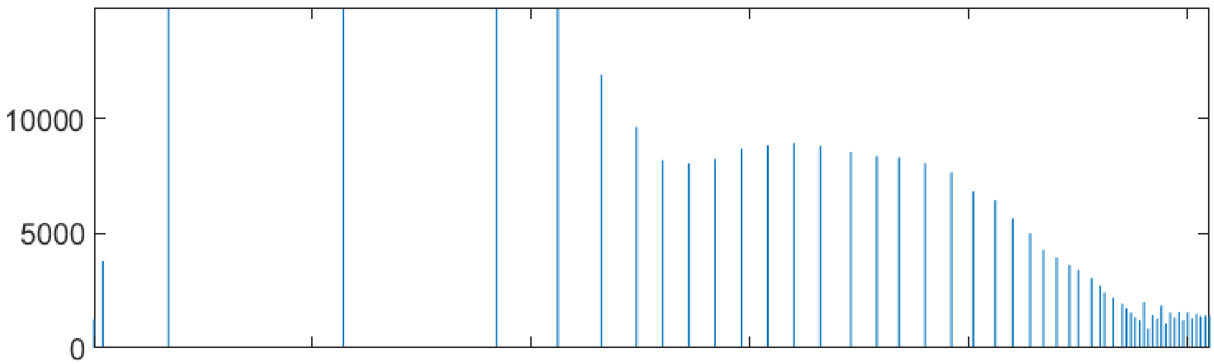

Figure 19: histogram of road1 after apply histeq function

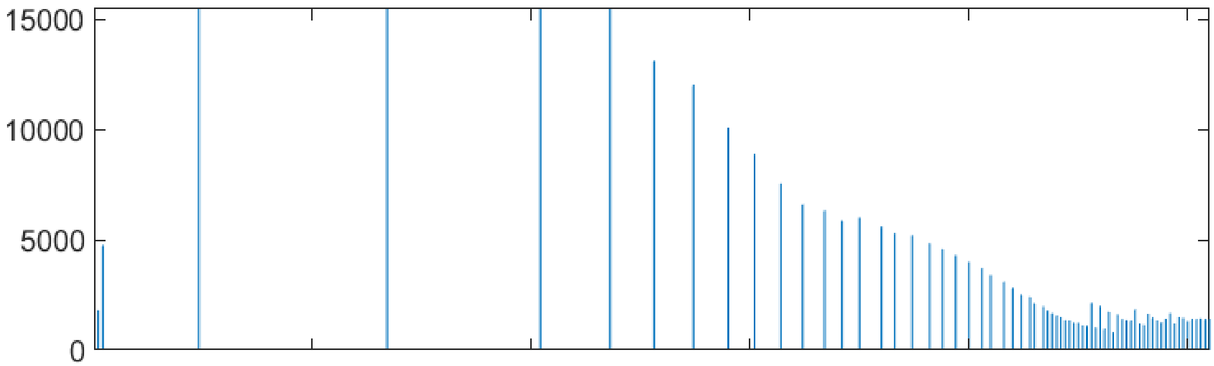

Figure 20: histogram of road2 after apply histeq function

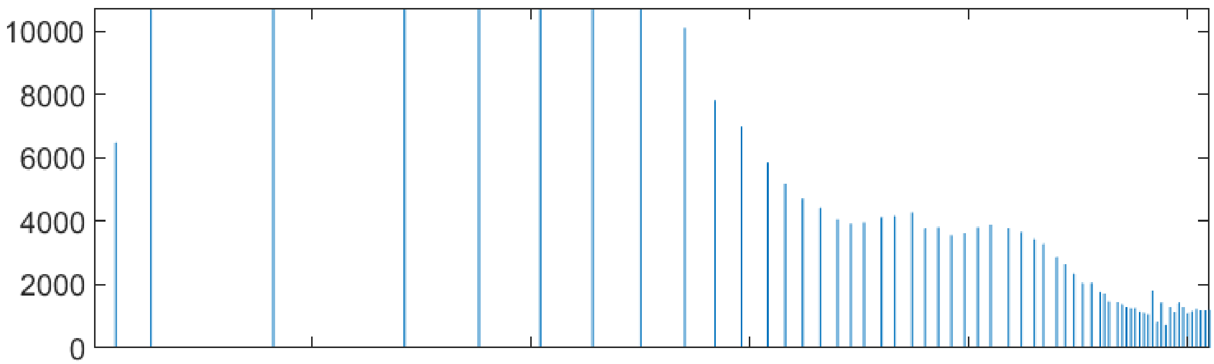

Figure 21: histogram of road3 after apply histeq function

Answer. **Figure 10-12** are the images after applying global histogram equalization to the origin images.
We can see that The contrast of the image is greatly enhanced. For Example in the image of road1, we can see the wall, the lamp adn the car; in the image of road2, we can clearly see the car adn the house; in the picture of the road3, we can see the bushes and the department.
But there are many pacthes in the image because there are jumps between different grayscale values.
Figure 10-12 are the images after applying histeq function in Matlab to the origin images, by comparision we can see that the results are almostly same. My Matlab code is followed and is saved as **b.m** and **myhisteq.m**

```
%b.m
clc,clear;
I1=imread("hw1_dark_road_1.jpg");
I2=imread("hw1_dark_road_2.jpg");
I3=imread("hw1_dark_road_3.jpg");
I1_process=myhisteq(I1);
I2_process=myhisteq(I2);
I3_process=myhisteq(I3)
I1_process_2=histeq(I1,256);
I2_process_2=histeq(I2,256);
I3_process_2=histeq(I3,256);

```

```
    figure('name','road1','NumberTitle','off');

    subplot(2,1,1);
    imshow(I1_process);          %display the MY process image
    imwrite(I1_process,"result/I1_myglobal.jpg","jpg");

    subplot(2,1,2);
    imhist(I1_process);          %display the MY process hist

    figure('name','road2','NumberTitle','off');

    subplot(2,1,1);
    imshow(I2_process);          %display the MY process image
    imwrite(I2_process,"result/I2_myglobal.jpg","jpg");

    subplot(2,1,2);
    imhist(I2_process);          %display the MY process hist

    figure('name','road3','NumberTitle','off');

    subplot(2,1,1);
    imshow(I3_process);          %display the MY process image
    imwrite(I3_process,"result/I3_myglobal.jpg","jpg");

    subplot(2,1,2);
    imhist(I3_process);          %display the MY process hist

    figure('name','road1','NumberTitle','off');

    subplot(2,1,1);
    imshow(I1_process_2);          %display the process image
    imwrite(I1_process_2,"result/I1_global.jpg","jpg");

    subplot(2,1,2);
    imhist(I1_process_2);          %display the process hist

    figure('name','road2','NumberTitle','off');

    subplot(2,1,1);
    imshow(I2_process_2);          %display the process image
    imwrite(I2_process_2,"result/I2_global.jpg","jpg");

    subplot(2,1,2);
    imhist(I2_process_2);          %display the process hist

    figure('name','road3','NumberTitle','off');

    subplot(2,1,1);
    imshow(I3_process_2);          %display the process image
    imwrite(I3_process_2,"result/I3_global.jpg","jpg");

    subplot(2,1,2);
    imhist(I3_process_2);          %display the process hist
```

```
    %myhisteq.m
    function I_process = myhisteq(I)
    [height,width]=size(I);  %
```

```
    r=zeros(1,256);             %
    for i=1:height
        for j=1:width
            r(I(i,j)+1)=r(I(i,j)+1)+1;%
        end
    end

    s=zeros(1,256);
    s(1)=r(1);
    for i=2:256
        s(i)=s(i-1)+r(i);        %CDF
    end

    for i=1:256
        s(i)=floor(255*s(i)/(height*width));        %L-1255
    end

    I_process=I;
    for i=1:height
    for j=1:width
        I_process(i,j)=s(I(i,j)+1);    %
    end
    end
    end
```

c) Apply locally adaptive histogram equalization to the original image. Display and submit the modified image. Plot and submit the histogram of the modified images grayscale values. Choose and report the number of tiles and the clipping limit for attaining higher contrast while avoiding the generation of noisy regions and the amplification of nonuniform lighting effects. Comment on the subjective quality of the modified image compared to the result in (b).

Figure 22: image of road1 after apply local adaptive histogram equalization

Figure 23: image of road2 after apply local adaptive histogram equalization

Figure 24: image of road3 after apply local adaptive histogram equalization

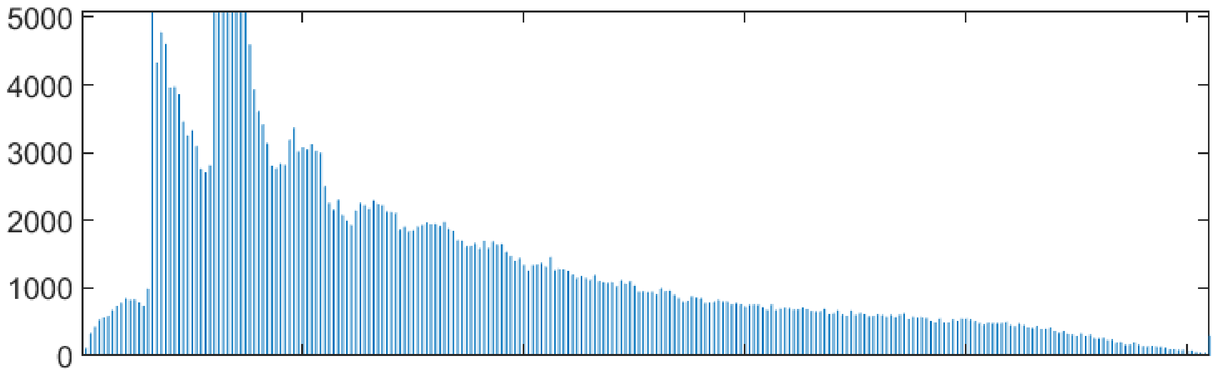

Figure 25: histogram of road1 after apply local adaptive histogram equalization

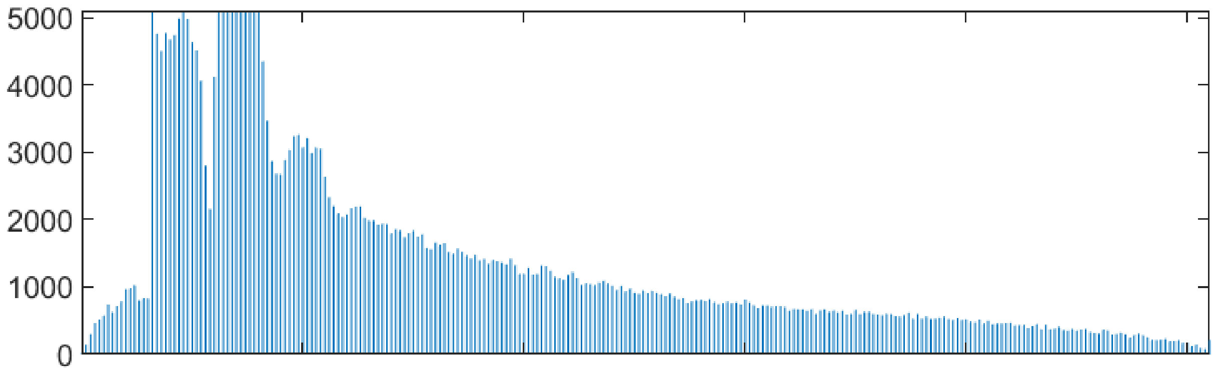

Figure 26: histogram of road2 after apply local adaptive histogram equalization

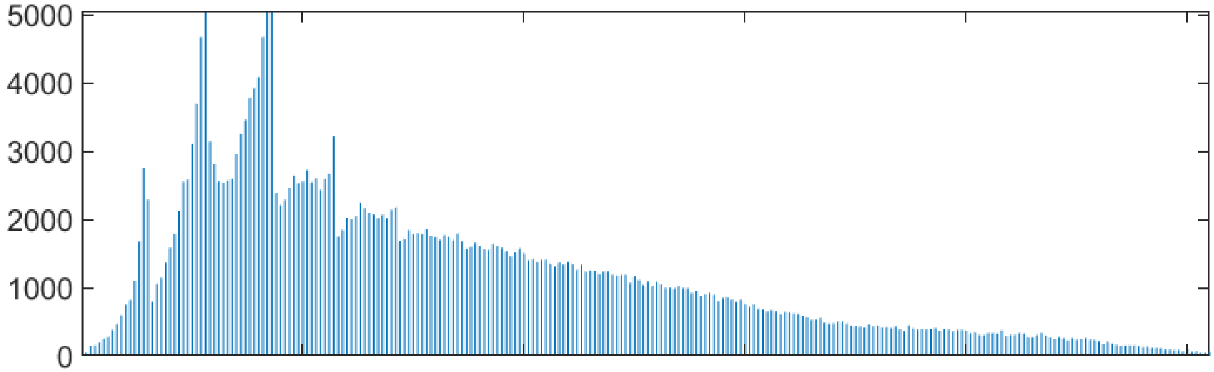

Figure 27: histogram of road3 after apply local adaptive histogram equalization

Answer. Figure 16-18 are the images after applying local adaptive histogram equalization to the origin images. We can see that the effect is much better than the case using global histgram equalization, There are not much patches, which makes us feel we can see more details in the images.
In the image of road1: tiles = 25*25, clipping limit =0.05;
In the image of road2: tiles = 25*25, clipping limit =0.05;
In the image of road3: tiles = 25*25, clipping limit =0.05;
My Matlab code is followed and is saved as **c.m**.

```
clc,clear;
I1=imread("hw1_dark_road_1.jpg");
I2=imread("hw1_dark_road_2.jpg");
I3=imread("hw1_dark_road_3.jpg");

I1_process=adapthisteq(I1,'NumTiles',[25 25],'ClipLimit',0.05);
I2_process=adapthisteq(I2,'NumTiles',[25 25],'ClipLimit',0.05);
I3_process=adapthisteq(I3,'NumTiles',[25 25],'ClipLimit',0.05);

figure('name','road1','NumberTitle','off');

subplot(2,1,1);
imshow(I1_process);          %display the process image
imwrite(I1_process,"result/I1_local.jpg","jpg");

```

```
    subplot(2,1,2);
    imhist(I1_process);         %display the process hist

    figure('name','road2','NumberTitle','off');

    subplot(2,1,1);
    imshow(I2_process);         %display the process image
    imwrite(I2_process,"result/I2_local.jpg","jpg");

    subplot(2,1,2);
    imhist(I2_process);         %display the process hist

    figure('name','road3','NumberTitle','off');

    subplot(2,1,1);
    imshow(I3_process);         %display the process image
    imwrite(I3_process,"result/I3_local.jpg","jpg");

    subplot(2,1,2);
    imhist(I3_process);         %display the process hist
```

d) Apply a γ-nonlinearity mapping to each image to perform contrast enhancement, show the new image, and submit the displayed image. For each image, find and report a value of γ that allows you to see more details.

Figure 28: image of road1 after apply γ-nonlinearity mapping

Figure 29: image of road2 after apply γ-nonlinearity mapping

Figure 30: image of road3 after apply γ-nonlinearity mapping

Answer. The images above are processed images after apply γ-nonlinearity mapping, the value of γ is 0.4., c is 1.
I try from 0.1 to 1 and compare the results. I find the 0.4 is the best, so I only report this. My Matlab code is followed and is saved as **d.m** and **gama.m**

```
%d.m
clc,clear;
I1=imread("hw1_dark_road_1.jpg");
I2=imread("hw1_dark_road_2.jpg");
I3=imread("hw1_dark_road_3.jpg");

I1_process=gama(im2double(I1),0.5);
figure('name','','NumberTitle','off');
imshow(I1_process);          %display the process image
imwrite(I1_process,"result/I1_gama.jpg","jpg");

I2_process=gama(im2double(I1),0.5);
figure('name','','NumberTitle','off');
imshow(I2_process);          %display the process image
imwrite(I2_process,"result/I2_gama.jpg","jpg");

I3_process=gama(im2double(I1),0.5);
figure('name','','NumberTitle','off');
imshow(I3_process);          %display the process image
imwrite(I3_process,"result/I3_gama.jpg","jpg");
```

```
%gama.m
function process_img = gama(img,x)

process_img=img.^x;

end
```