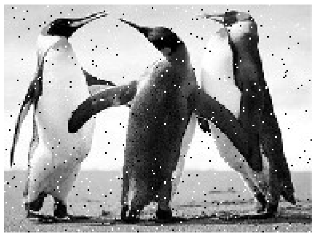
Homework 1
Ryan English
CSE-627-50
09/11/2021

1) **Select two (real) images of your choice that are good candidates to illustrate the concepts of low pass and high pass filtering.**
    a. **Explain why these 2 images are good candidates to illustrate these concepts.**
    b. **Filter each image with an ideal lowpass filter with cutoff frequencies set at radii values of 5 and 20. Show the input image, filter, and output image in the spatial domain and in the frequency domain. Explain the process and analyze the results**
    c. **Repeat (b) with an ideal high-pass filter.**

a)

Penguins

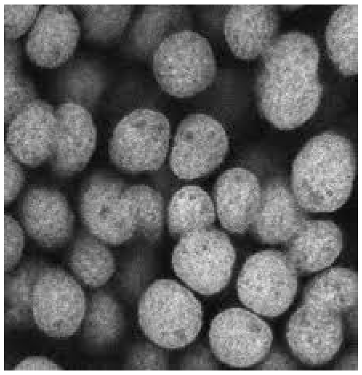


Low pass

This image was chosen as the low pass filtering due to the noise throughout the image. When applying the low pass filter to the image the noise will be removed and replaced with the ringing affect; the "hardness" of the affect is determined by the cutoff frequency, so the higher the cutoff the more ringing and the lower the cutoff the blurrier the image will be.

NOTE: I got this image from an example about low pass filters, as seen here: MATLAB - Ideal Lowpass Filter in Image Processing - GeeksforGeeks

Microbes



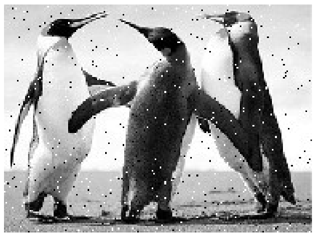
High pass

This image was chosen as the high pass filtering due to the microbes being a bit blurry and by utilizing the high pass filter we can get a bit more detail on the image by sharpening the details.

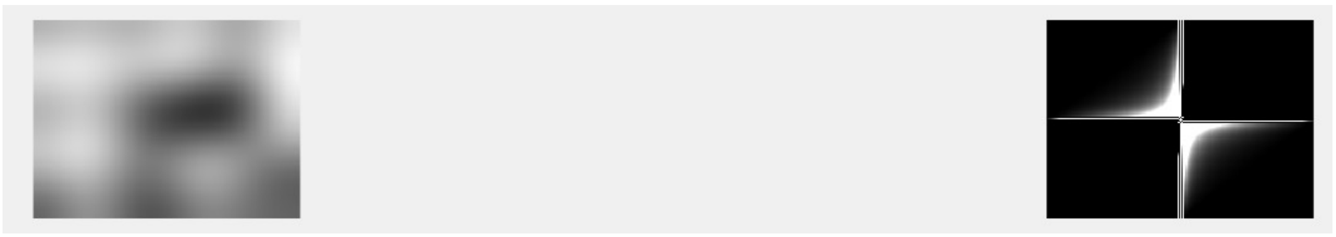
b)

Low Pass Filter

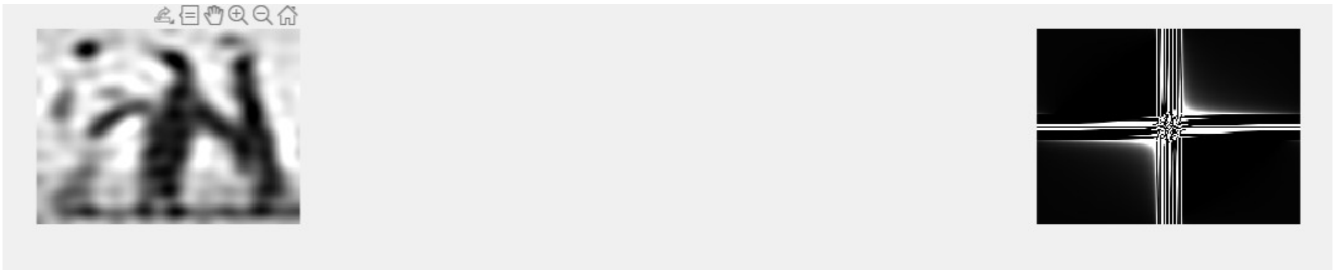
Original Image


Spatial Domain

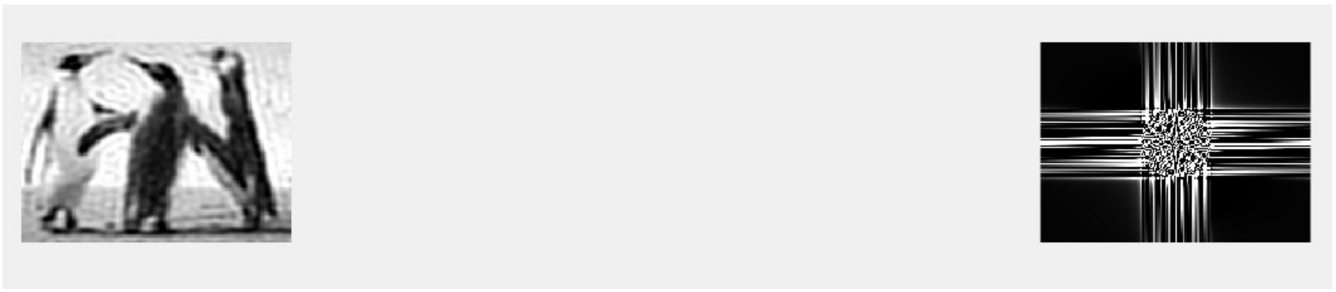
D0 = 5


Spatial Domain/Frequency Domain

D0 = 20

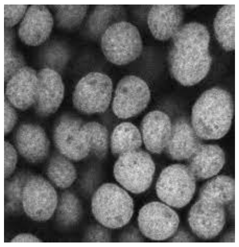

Spatial Domain/Frequency Domain

D0 = 50

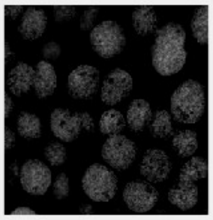
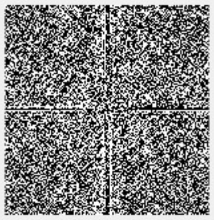

Spatial Domain / Frequency Domain

High Pass Filter

Original Image

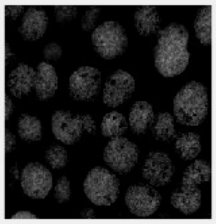
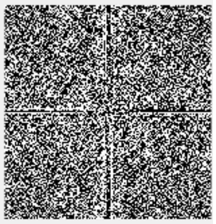


D0 = 5

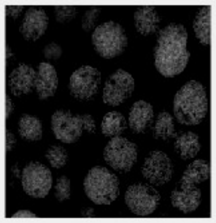
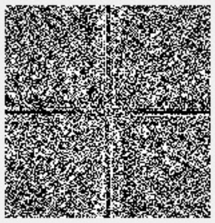
Spatial Domain/Frequency Domain

D0 = 20

Spatial Domain/Frequency Domain

D0 = 50

Spatial Domain/Frequency Domain

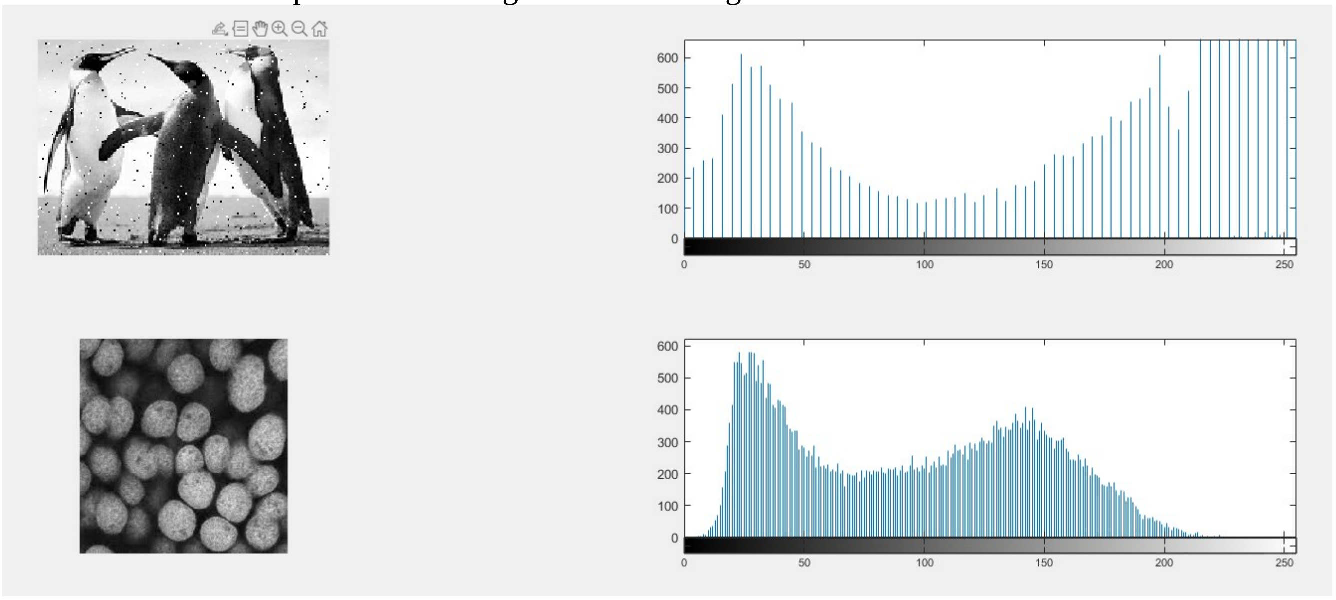
Analysis

In order to generate the low pass filter we begin with the original image padded with zeros. Then we generate the low pass filter with a frequency cutoff point of 5 (which means we end up with [5 1; 4 1; 4 1; 4 1; 3 1;] 1s in each of the four corners). This low pass filter (low pass because the 1s are in the corner – taking these values – and the zeros are in the center – ignoring these) is then convoluted over the image, generating the output image (LEFT) you see above. This process is then repeated for the other cut off points. As you can see the lower the cutoff the more blurry the image, since it its only taking the frequencies in the very center of the frequency domain (RIGHT). As the cutoff point gets larger the less blurry the image and the more "ringing" the image has (see D0 = 50).

High pass filter follows the exact format as the low pass filter, the difference being in the filter itself. The high pass filter is an inverse of the low pass filter so the 0s are on the corners and 1s are in the center; which causes the points of interest to be a bit sharper and the background to go black. The high pass filter perseveres the center of the frequency domain (RIGHT) keeping the more frequent components.

I find the frequency domain images of the microbes really interesting. It really made me question the frequency domain and what it really is portraying. After revisiting the book and online resources, I have come to the conclusion the vast difference between the penguins and microbes frequency domain images is due to the microbes massive variation in the intensity of the all the pixels.

To illustrate this concept I ran the histogram of each images which enforces this ideal:

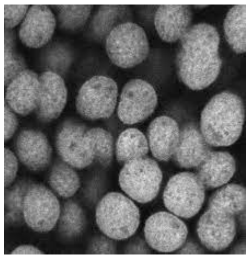


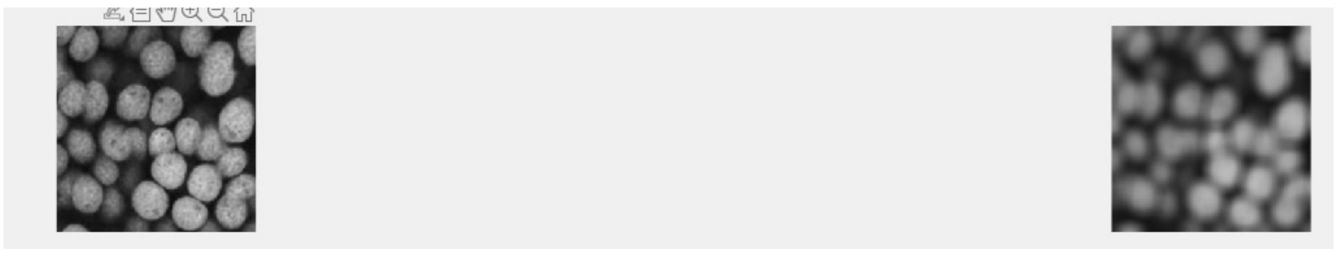
You can clearly see that the penguin image has a fraction of change in the intensities as the microbes image.

2) **Apply 2 different smoothing filters (with different sizes and different weights), in the spatial domain, to the two images in problem 1. Compare the results of the different filters and explain their differences and similarities.**

Average Filter

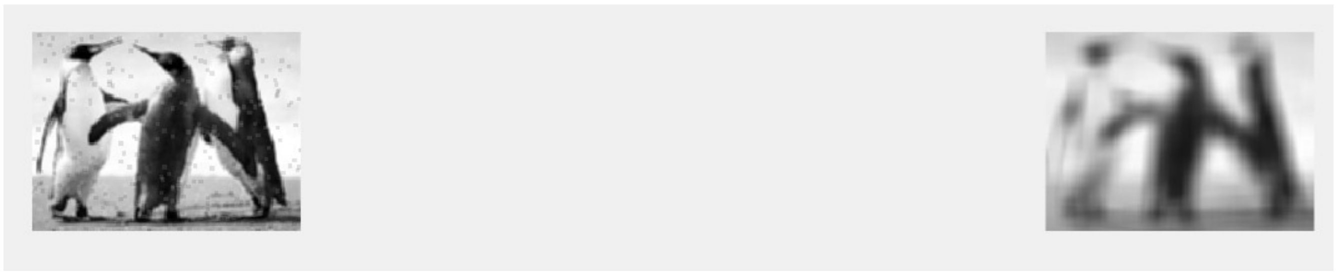
Original Image





[3 3] Mask / [15 15] Mask

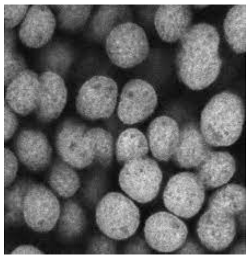
Original Image

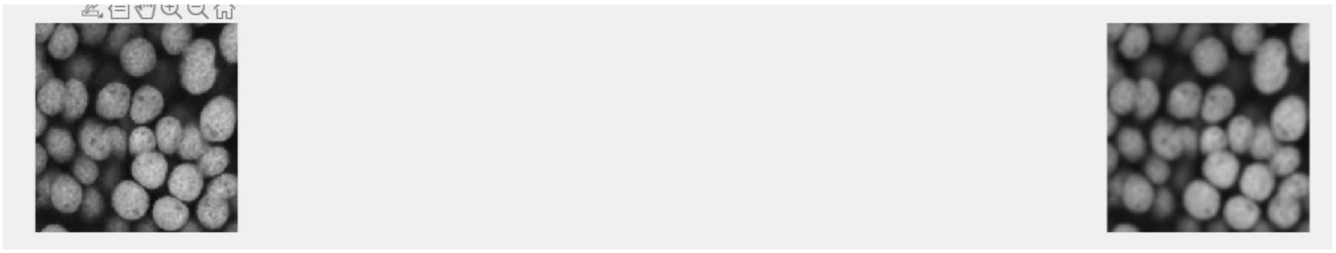


[3 3] Mask / [15 15] Mask

Gaussian

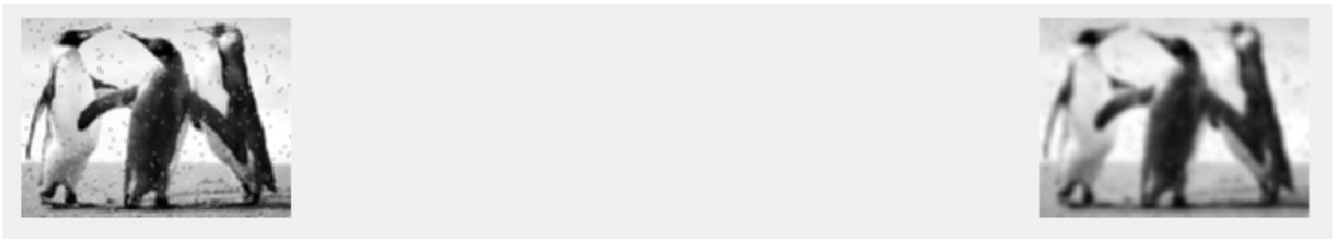
Original Image





[3 3] Mask 5.0 sigma / [8 8] Mask 2.0 sigma

Original Image



[3 3] Mask 5.0 sigma / [8 8] Mask 2.0 sigma

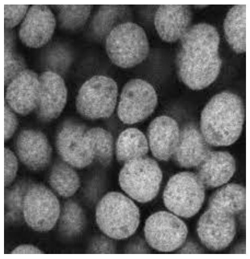
Analysis

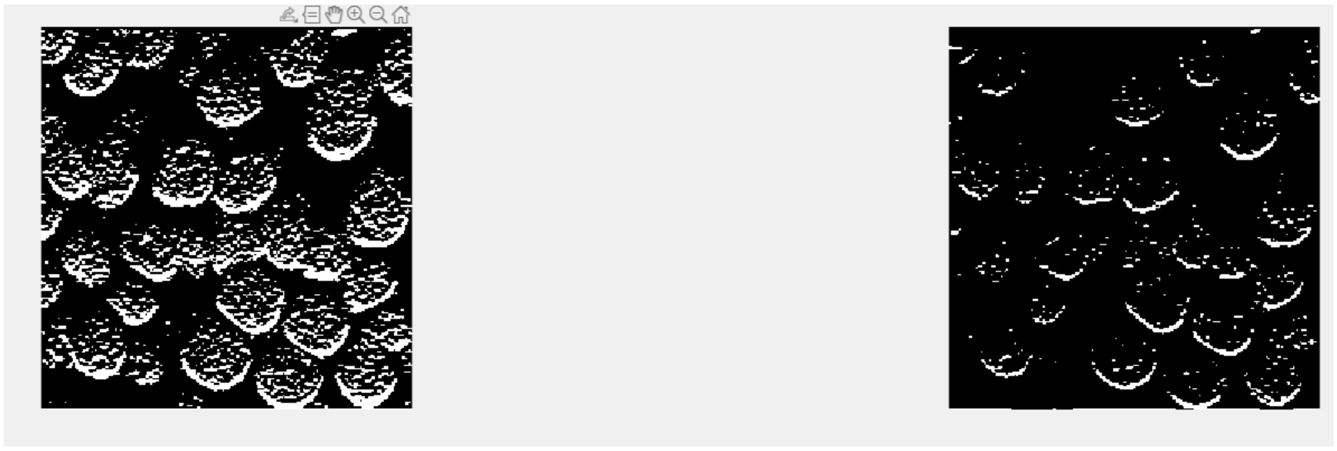
Average filter, being one of the simplest smoothing filters, creates a mask that has an average of its size. For example the [3 3] mask has a value of 1/9 since there are nine slots and each take up one position and for the [15 15] it would be 1/15. This mask then is convoluted across the original image and averages out the neighborhood of the pixels. So as one would expect the higher the mask the more pixels are being averaged and the blurrier the image will be. This is shown in both images that get the [15 15] mask which are so blurry they lost most of their detail – like the penguins noise is non existent.

Gaussian filter, on the other hand, has the heaviest weights in the middle and lowers the weights as it moves from the center location based on the sigma. As with the average filter the larger the mask the more blurring the output image will have, however, with the Gaussian the sigma has a much higher blur rate, as you can see in the above images. The 2.0 sigma created a much heavier blurred imaged, but if you look at the penguins it still has a smooth enough surface you can make out the penguins just fine.

3) **For each of the two images in Problem 1, use spatial filtering techniques to detect the edges. Discuss the detected edges, their strength as a function of their orientations, and any interesting observations (e.g., double edges, disconnected edges, weak edges, etc.). You may need to threshold the detected edges as binary images to support your observations.**

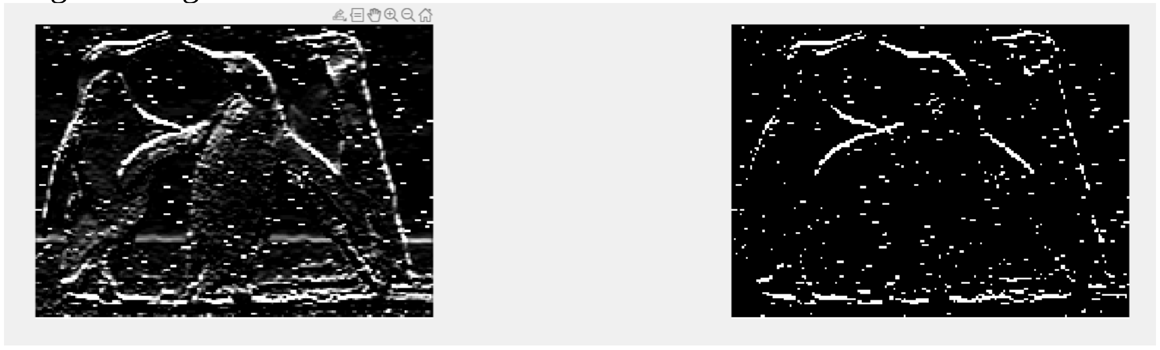
Original Image





Sobel 0.2 Threshold / Sobel 0.5 Threshold

Original Image



Sobel No Threshold / Sobel 0.9 Threshold

Analysis

Utilizing a sobel mask for edge detection, without transposing in order to get the horizontal edges of the original image. The sobel mask generates the edge detection because it detects the hard changes in pixels by having the zeros in the horizontal center and positive values on the top and negative values on the bottom, so when the mask is convoluted on the original image it sets the hard changes (edges) to zero.

In the penguins image it really outlines the noise in the image because if you view the original image the noise consists of a hard black ~4 pixels horizontal/vertical lines. Therefore, when the sobel mask is applied given it is only a [3 3] matrix it will detect these as edges.

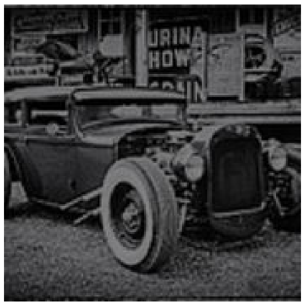
In the microbes image the horizontal edge detection only gets about half the edges due them being circular in nature, so in order to fully detect the edges around the microbes one would have to get both the horizontal and vertical filtered images and combine them. The microbes suffer the same problem as the penguins however, where the center of the microbes has those hard changes in pixels so I applied a threshold to remove as much as I could from the center without loosing too much in the edge of the microbes.

4) **Select one (real) image of your choice that is a good candidate to illustrate the concepts of histogram equalization**

   a. **Explain why this image is a good candidate to illustrate this concept.**
   b. **Perform histogram equalization with 256 and 32 levels**
   c. **Compare the input image and the two histogram equalized images. You should include the histograms of the 3 images to support your analysis.**

a)

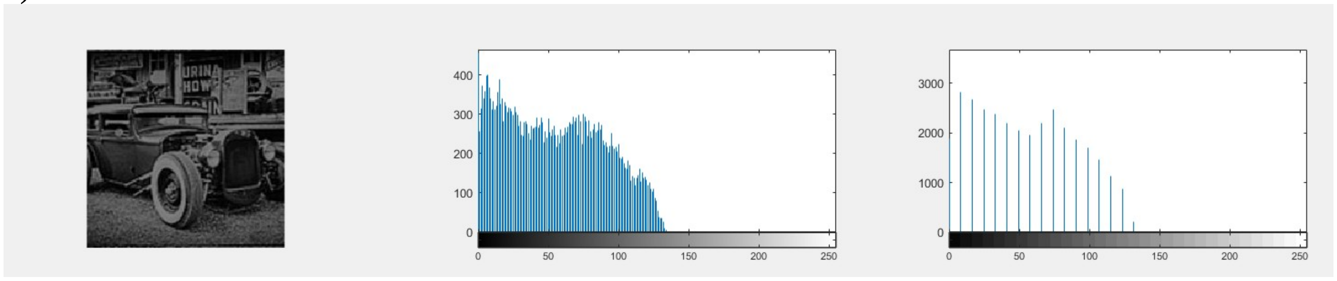
Original Image



I took an original image of a car and opened it in paint to darken it by painting over it by 0.5 black alpha. This created a much deeper black image; thus, giving me the perfect candidate for histogram equalization.

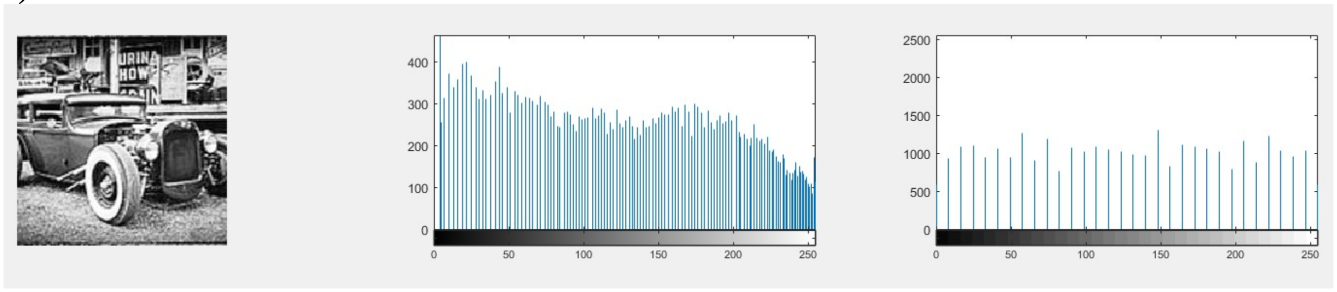
I chose this image to portray the histogram equalization since it is much darker it will have a much heavier histogram below ~180 intensity (see B). When we apply the histogram equalization it will spread the intensity across the 256 intensity spectrum allowing more of the detail to be visible and almost returning the image to its original state before I altered it.

b)

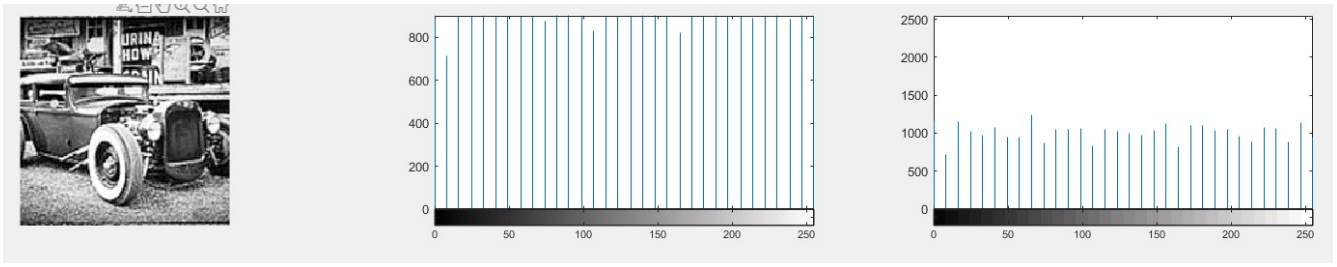


Original Image, Histogram 256, and Histogram 32

c)



256 Histogram Equalized Image, Histogram 256, and Histogram 32



32 Histogram Equalized Image, Histogram 256, and Histogram 32

Analysis

There is not a huge difference in the equalization of the image between 256 and 32 intensities, because the intensity is being spread out across the spectrum in the same means. The colors are “close enough” so to us it gives the image about the same look.

You can really see this, when you compare the 32 Histograms of each. The histograms are almost identical even when the equalized in vastly different domains.

Code

problem1_lowpass.m

```
img = imread('input17.png');

padded = paddedsize(size(img));

filter = lpfilter('ideal', padded(1), padded(2), 5);

img1 = dftfilt(img, filter, 'symmetric');

F = ifftshift(fft2(img1));

filter2 = lpfilter('ideal', padded(1), padded(2), 80);

img2 = dftfilt(img, filter2, 'symmetric');

F2 = ifftshift(fft2(img2));

filter3 = lpfilter('ideal', padded(1), padded(2), 50);

img3 = dftfilt(img, filter3, 'symmetric');

F3 = ifftshift(fft2(img3));

subplot(4,2,1), imshow(img)
subplot(4,2,3), imshow(img1)
subplot(4,2,4), imshow(F)
subplot(4,2,5), imshow(img2)
subplot(4,2,6), imshow(F2)
subplot(4,2,7), imshow(img3)
subplot(4,2,8), imshow(F3);
```

problem1_highpass.m

```
img = imread('microboes.jpg');
img = rgb2gray(img);

padded = paddedsize(size(img));

filter = hpfilter('ideal', padded(1), padded(2), 5);

img1 = dftfilt(img, filter, 'symmetric');

F = ifftshift(fft2(img1));

filter1 = hpfilter('ideal', padded(1), padded(2), 20);
```

```
img2 = dftfilt(img, filter, 'symmetric');

F2 = ifftshift(fft2(img2));

filter2 = hpfilter('ideal', padded(1), padded(2), 50);

img3 = dftfilt(img, filter, 'symmetric');

F3 = ifftshift(fft2(img3));

subplot(4,2,1), imshow(img)
subplot(4,2,3), imshow(img1)
subplot(4,2,4), imshow(F)
subplot(4,2,5), imshow(img2)
subplot(4,2,6), imshow(F2)
subplot(4,2,7), imshow(img3)
subplot(4,2,8), imshow(F3);
```

problem1_hist.m

```
penquins = imread('input17.png');

micro = imread('microboes.jpg');
micro = rgb2gray(micro);

subplot(3,2,1), imshow(penquins)
subplot(3,2,2), imhist(penquins, 256)
subplot(3,2,3), imshow(micro)
subplot(3,2,4), imhist(micro, 256);
```

problem2_avg.m

```
imga = imread('microboes.jpg');
imga = rgb2gray(imga);

imgb = imread('input17.png');

padded = paddedsize(size(imga));

ga = imfilter(imga, fspecial('average', [3 3]), 'corr', 'symmetric', 'same');
ga1 = imfilter(imga, fspecial('average', [15 15]), 'corr', 'symmetric', 'same');

gb = imfilter(imgb, fspecial('average', [3 3]), 'corr', 'symmetric', 'same');
gb1 = imfilter(imgb, fspecial('average', [15 15]), 'corr', 'symmetric', 'same');

subplot(4,2,1), imshow(imga)
subplot(4,2,3), imshow(ga)
subplot(4,2,4), imshow(ga1)
subplot(4,2,5), imshow(imgb)
subplot(4,2,7), imshow(gb)
subplot(4,2,8), imshow(gb1);
```

problem2_gaussian.m

```
imga = imread('microboes.jpg');
imga = rgb2gray(imga);

imgb = imread('input17.png');

padded = paddedsize(size(imga));

ga = imfilter(imga, fspecial('gaussian', [3 3], 5.0), 'corr', 'symmetric', 'same');
ga1 = imfilter(imga, fspecial('gaussian', [8 8], 2.0), 'corr', 'symmetric', 'same');

gb = imfilter(imgb, fspecial('gaussian', [3 3], 5.0), 'corr', 'symmetric', 'same');
gb1 = imfilter(imgb, fspecial('gaussian', [8 8], 2.0), 'corr', 'symmetric', 'same');

subplot(4,2,1), imshow(imga)
subplot(4,2,3), imshow(ga)
subplot(4,2,4), imshow(ga1)
subplot(4,2,5), imshow(imgb)
subplot(4,2,7), imshow(gb)
subplot(4,2,8), imshow(gb1);
```

problem3.m

```
mask = fspecial('sobel');

micro_img = imread('microboes.jpg');
micro_img = rgb2gray(micro_img);

micro_gs = imfilter(micro_img, mask);

micro_gs2 = abs(micro_gs) > 0.2*abs(max(micro_gs(:)));
micro_gs5 = abs(micro_gs) > 0.5*abs(max(micro_gs(:)));

pen_img = imread('input17.png');

pen_gs = imfilter(pen_img, mask);

pen_gs2 = abs(pen_gs) > 0.9*abs(max(pen_gs(:)));

subplot(2,2,1), imshow(micro_gs2)
subplot(2,2,2), imshow(micro_gs5)
subplot(2,2,3), imshow(pen_gs)
subplot(2,2,4), imshow(pen_gs2);
```

problem4.m

```
img = imread('car.jpg');
img = rgb2gray(img);

twofiftysix = histeq(img, 256);
thirtytwo = histeq(img, 32);

subplot(3,3,1), imshow(img)
subplot(3,3,2), imhist(img, 256)
subplot(3,3,3), imhist(img, 32)
subplot(3,3,4), imshow(twofiftysix)
subplot(3,3,5), imhist(twofiftysix, 256)
subplot(3,3,6), imhist(twofiftysix, 32)
subplot(3,3,7), imshow(thirtytwo)
subplot(3,3,8), imhist(thirtytwo, 256)
subplot(3,3,9), imhist(thirtytwo, 32);
```