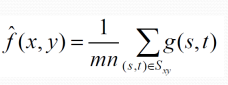
**Digital Image Processing – Homework #6**

**Juan Silva**

**ECE 595**

**October 9, 2019**

1. The white bars in the test pattern shown are 7 pixels wide and 210 pixels high. The separation between bars is 17 pixels. What would this image look like after application of:  
     
   (a) A 3x3 arithmetic mean filter?   
     
   (b) A 7x7 arithmetic mean filter?   
     
   (c) A 9x9 arithmetic mean filter?



**hw5\_q1.m**:  
N = 150;

M = 160;

f=zeros(N,M);

for i = 1:N

for j = 1 : M

if mod(i, 7) == 0 && mod(i, 2) ~= 0

f(7:140,i:i+7) = 1;

end

end

end

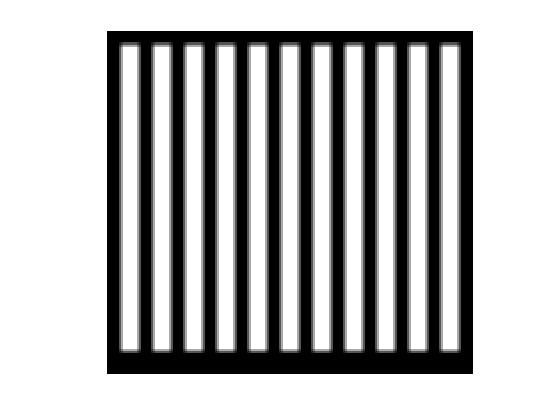
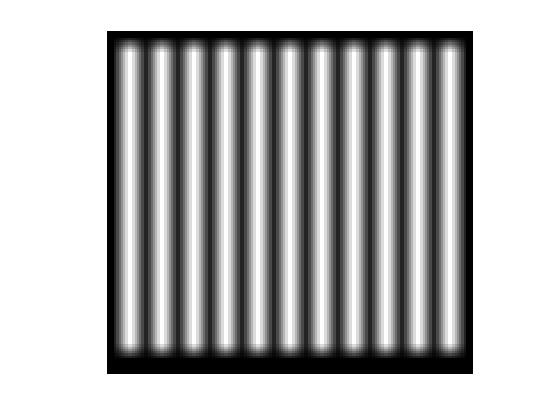
%Arithmetic mean

filter = fspecial('average', 3);

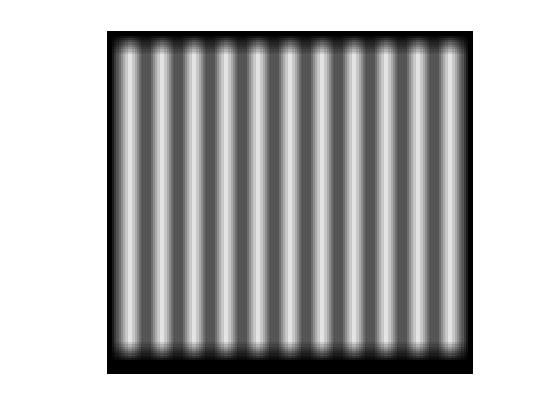
g = imfilter(f,filter);

imshow(g,'InitialMagnification','fit')

**Results**:

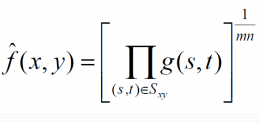
 

3x3 Mean Filter; Some Blurring. 7x7 Mean Filter; More Blurring.



9x9 Mean Filter; Most Blurring.

1. **Geometric Filter**

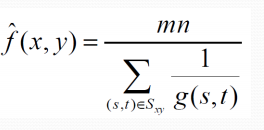


**(a)** When the 3x3 filter is applied, there should be little change to the original image since the weight of the polynomial 1/mn is small.

**(b)** When the 7x7 filter is applied, there should be some detail lost. The details lost here would make the white lines smaller.

**(c)** When the 9x9 filter is applied, significant details would be lost. Additionally, the image would be very blurry similar to the output from the 9x9 arithmetic mean filter. However, even more detail would be lost.

**3. Harmonic Mean Filter**

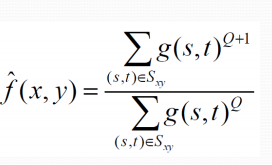


**(a)** When the 3x3 filter is applied, it will affect the white portion of the image but only slightly. This is because the harmonic mean filter handles salt noise well but not pepper.

**(b)** When the 7x7 filter is applied, the failure of pepper is more apparent. More of the white rectangle will be lost.

**(c)** When the 9x9 filter is applied, even more detail is lost. There will be more black than white.

**4. Contraharmonic Mean Filter (Q = 1)**



**(a)** When the 3x3 filter is applied, the white portions of the image will be affected. When Q = 1, the pepper noise will be affected which changes the black portions near the white portions

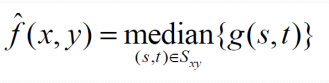
**(b)** When the 7x7 filter is applied, the change between black and white are more noticeable. Less black around the white parts of the image.

**(c)** When the 9x9 filter is applied, the white portions of the image should be larger than the original image.

**5. Contraharmonic Mean Filter (Q = -1)**

When Q = -1, this will act as a harmonic filter.

**6. Median Filter**

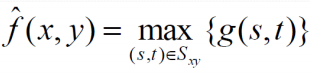


**(a)** When the 3x3 filter is applied, the median filter replaces the pixel values with the intensity levels that are nearby. Therefore, the corners of the white rectangles will smooth out.

**(b)** When the 7x7 filter is applied, the corner smoothing becomes more apparent. There are no rectangular corners because the median around them are larger than the value zero so there are more white than black.

**(c)** When the 9x9 filter is applied, the image should look the same aside from the corners. The smoothing in every corner will smooth out and the rectangular does not have any sharp lines anymore.

**7. Max Filter**

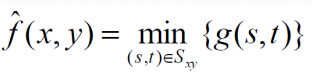
****

**(a)** When the 3x3 filter is applied, the max filter checks for the largest pixel values or the brightest points on the image. Therefore, the white portions of the image are magnified or their intensities levels increase at surrounding neighborhoods.

**(b)** When the 7x7 filter is applied, the white parts of the image become slightly larger due to the larger filter size

**(c)** When the 9x9 filter is applied, the white parts of the image become larger due to the larger filter size.

**8. Min Filter**



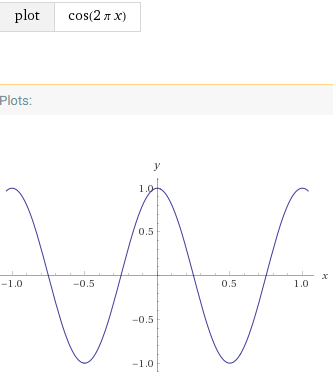
**(a)** When the 3x3 filter is applied, the min filter checks for the smallest pixel values or the darkest points on the image. Therefore, the black portions of the image are magnified or their intensities levels at surrounding neighborhoods.

**(b)** When the 7x7 filter is applied, the dark parts of the image become slightly smaller due to the larger filter size

**(c)** When the 9x9 filter is applied, the dark parts of the image become smaller due to the larger filter size.



(b)



,



The function G(u,v) will peak at 1 when U = M/2 and dips at -1 when U = M.

1. The following Matlab instructions create a white rectangle in the center of black square.   
     
   f=zeros(30,30);   
   f(5:24,13:17)=1;   
   imshow(f,'InitialMagnification', 'fit')   
     
   Write a sequence of Matlab instructions to create the Fourier transform of the given figure. Use shift property to place the highest frequency at the center of the figure.

**hw5\_q2.m:**

clear, clc

format short, format compact

%Generate centered white rectangle

f = zeros(30,30);

f(5:24, 13:17) = 1;

imshow(f, 'InitialMagnification', 'fit');

%apply fast fourier transform

F = fft2(f);

F\_abs = abs(F);

%Display Fourier-transformed image

figure

imshow(F\_abs,[],'InitialMagnification', 'fit');

%Apply padding to filter

F = fft2(f, 256, 256);

F\_abs = abs(F);

%Display edges

figure

imshow(F\_abs, []);

%shift towards center

F\_shift = fftshift(F);

F\_abs = abs(F\_shift);

figure

imshow(F\_abs,[]);

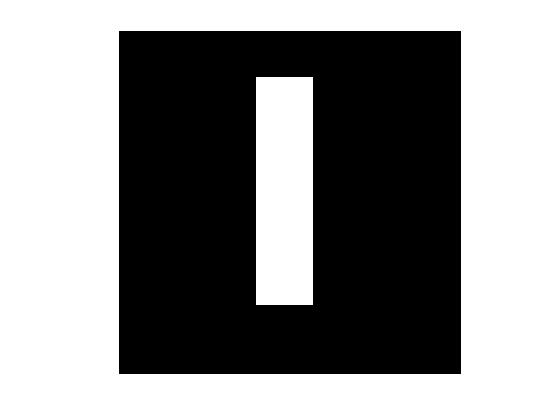
%reduce contrasting using log function

F\_cntr = log(1 + F\_abs);

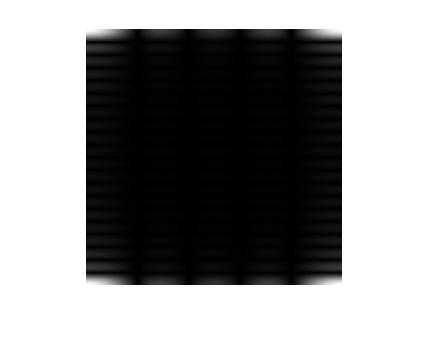
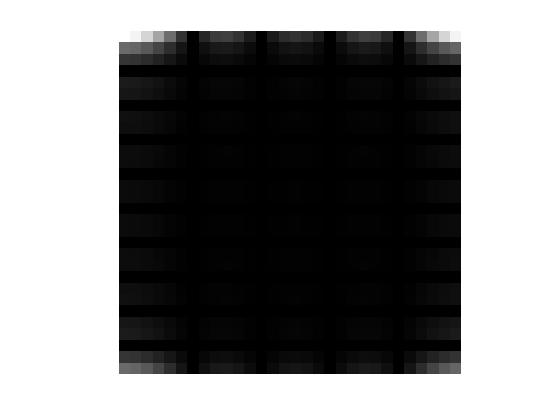
figure

imshow(F\_cntr,[]);

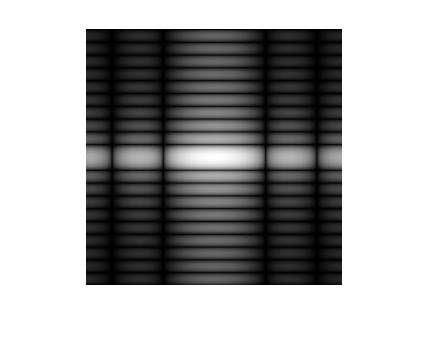
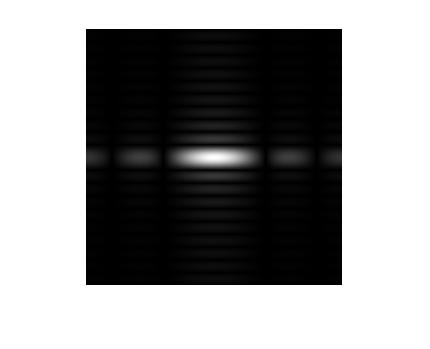
**Results**



Generated White Square



Fast Fourier Transform Padding Applied



Shift to Center Contrast Reduced

1. The file “noisy.png” from the course website is an image with a certain amount of periodic noise. The noise looks like a pattern of close lines running diagonally from bottom left to top right. You can verify this by zooming in the image. Using the method of notch filtering, remove this noise. Show:   
   a. The Matlab code   
   b. The mask you used   
   c. The resulting image.

clear, clc

format compact, format short

%Open image

I = imread('noisy.png');

imshow(I, []);

%Apply fourier transform

Itemp = fft2(double(I));

%Apply shifting to center fourier

Ishift = fftshift(Itemp);

I = abs(Ishift);

%Display noise

figure

imshow(log(I), []);

%Find the peaks via inspection

Y = [125, 175; 125, 425; 400, 100; 400, 380];

%upper left, upper right, lower left, lower right

%Create a mask of zeros where

M = ones(size(I,1), size(I,2));

size(I); %image is 512x512

N = 15;

z = zeros(N,N); %

%Apply a mask

for i = 1:length(Y)

M(Y(i,2) - floor(N/2) : Y(i,2) + floor(N/2), Y(i,1) - ...

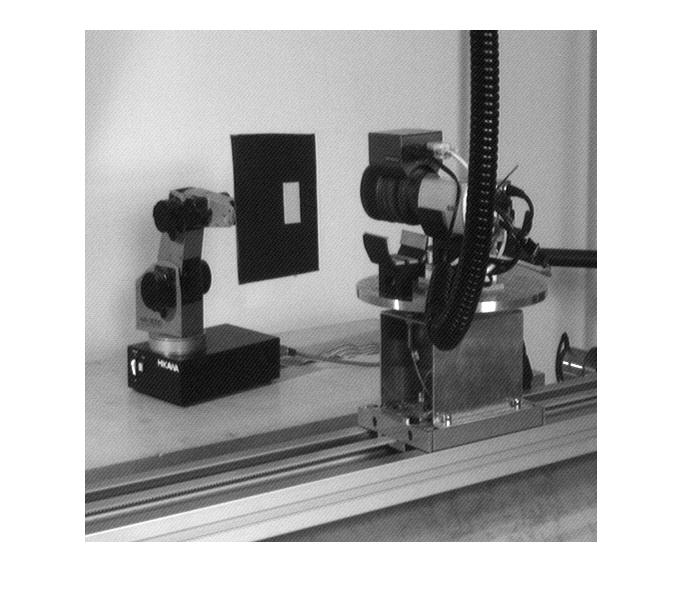
floor(N/2): Y(i,1) + floor(N/2)) = z(:,:);

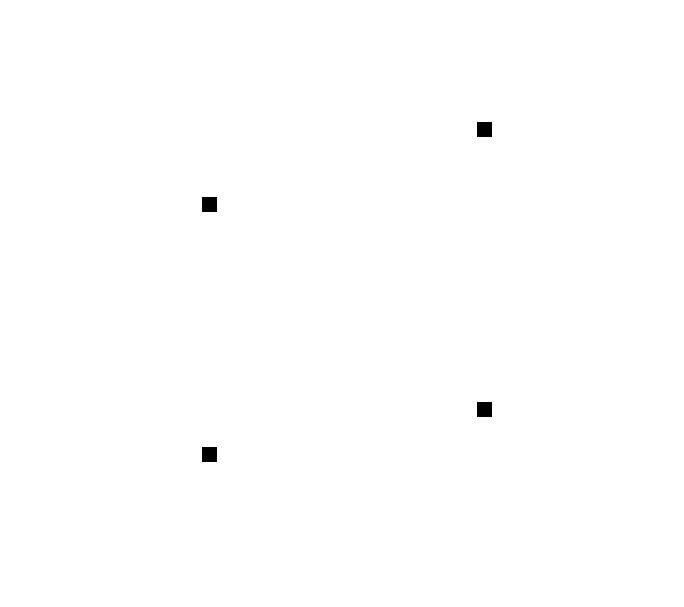
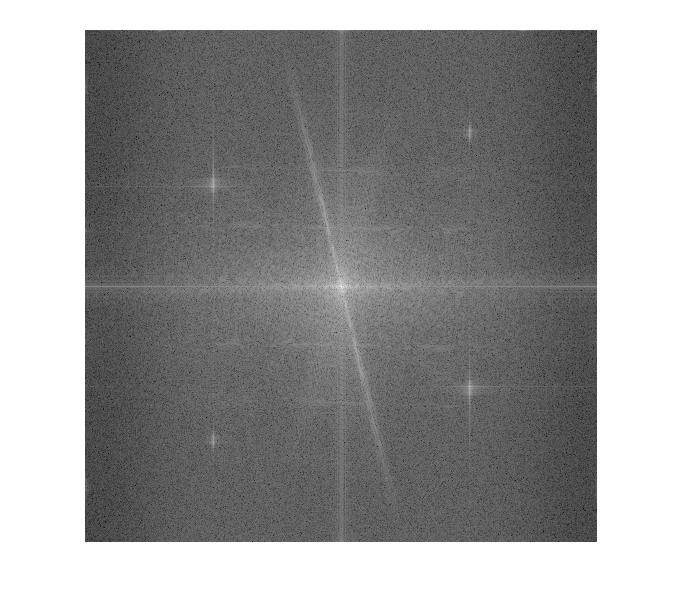
end

figure

imshow(M, []);

**Results**

  
Original Noisy Image



Fourier Transform Identified Peaks