ECE 4750: Digital Signal Processing Lab

Project 2: Application on Images

Angel Rodriguez

Utah Valley University

# Introduction

This project involves the application of DSP and filter design for Images. Given a total of 8 noisy images, the objective is to design appropriate filters for each noisy image to reduce/remove the noise from the image. The image includes complex equations, and the noise includes items such as an obstructing line through the image as well as background obstructions. The results show the images after the filters have been applied with the intention of reducing image noise.

# Process

Each of the given images have distortion including: lines through the images, salt and pepper backgrounds in various magnitudes, and diagonal lines running through the background. The plan was to use the movmean() filter from MATLAB to reduce the noise. The idea was that the filter would determine the pixels based on the averages of surrounding pixels and put those average values on a new image. The function syntax used was movmean(A,[5 5]) which computes the mean with a window of length 5+5+1 that includes the element in the current position, 5 elements backward, and 5 elements forward. This seemed to work for some images after only applying the filter once. However, for some images, the filter needed to be applied multiple times using for loops. For example, the results from Figures 5, 6, 7, and 8 were accomplished by applying the filter 3 times. Looking at the original grayscale images, this makes sense since those images have considerably more noise than the rest of the images. For images with more manageable noise such as Figures 1, 2, 3, and 4, the filter was only applied once or twice to minimize the amount of blur that results from applying the movemen() filter. The tradeoff with removing noise is that the readability of the equations lessens since the numbers and symbols become blurrier. This issue of blurriness leads to the second part of fixing the blurriness cause by the movemean() filter: binarization. The imbinarize() function of MATLAB was used to accomplish this with 2 parameters: the first being the images to binarize and the second parameter to define the pixel value cutoff from being converted to a 0 or a 1 where 0 is white and 1 is black. Initially the second parameter was set to 500 which resulted in pure white images, then 250 which resulted in very blurry images, and then 150 which made images less blurry than the movmean() filter alone. Around the 150-value showed better results for all figures. The next step was to record the spectrum of the original image to the spectrum of the new image which is shown in the figures below. The spectrums were obtained using the MATLAB function fft2() which returns the two-dimensional Fourier transform of a matrix using a fast Fourier transform algorithm, and followed by the MATLAB function fftshift(X) which rearranges a Fourier transform X by shifting the zero-frequency component to the center of the array. Since the goal was to remove noise, the filtered images look like each other. Similarly, the spectrums of the filtered images look like each other. A pattern seen in the spectrums is that they become significantly darker. This is due to the binarization making each pixel choose between being white or black with no gray color in between. Results are as shown below

# Figures

Text, letter

Description automatically generated

Text

Description automatically generated

Figure 1:  
Comparison between the original and filtered grayscale/spectrum images of p1.png.

Text, letter

Description automatically generatedA picture containing text, spoon, tableware

Description automatically generated

Figure 2:  
Comparison between the original and filtered grayscale/spectrum images of p1\_2.png.

Text, letter

Description automatically generated

A picture containing text, spoon, tableware

Description automatically generated

Figure 3:  
Comparison between the original and filtered grayscale/spectrum images of p1\_3.png.

A picture containing diagram

Description automatically generated

Text, letter

Description automatically generated

Figure 4:  
Comparison between the original and filtered grayscale/spectrum images of p1\_4.png.

Text, letter

Description automatically generated

Text, letter

Description automatically generated

Figure 5:  
Comparison between the original and filtered grayscale/spectrum images of p1\_5.png.

Diagram

Description automatically generated with medium confidence

Text

Description automatically generated

Figure 6:  
Comparison between the original and filtered grayscale/spectrum images of p1\_6.png.

A picture containing diagram

Description automatically generated

Text, letter

Description automatically generated

Figure 7:  
Comparison between the original and filtered grayscale/spectrum images of p1\_7.png.

Diagram

Description automatically generated with medium confidence

Text

Description automatically generated

Figure 8:  
Comparison between the original and filtered grayscale/spectrum images of p1\_8.png.

# Attachments

MATLAB Code:  
% ECE 4755

% Project 2

clc, close all;

grayImage1 = imread('p1.png');

grayImage2 = imread('p1\_2.png');

grayImage3 = imread('p1\_3.png');

grayImage4 = imread('p1\_4.png');

grayImage5 = imread('p1\_5.png');

grayImage6 = imread('p1\_6.png');

grayImage7 = imread('p1\_7.png');

grayImage8 = imread('p1\_8.png');

% Get the dimensions of the image.

[rows, columns, numberOfColorBands] = size(grayImage2);

% if numberOfColorBands > 1

% % It's not really gray scale like we expected - it's color.

% % Convert it to gray scale by taking only the green channel.

% grayImage2 = grayImage2(:, :, 1); % Take green channel.

% end

grayImage1 = grayImage1(:, :, 1); % Take green channel.

grayImage2 = grayImage2(:, :, 1); % Take green channel.

grayImage3 = grayImage3(:, :, 1); % Take green channel.

grayImage4 = grayImage4(:, :, 1); % Take green channel.

grayImage5 = grayImage5(:, :, 1); % Take green channel.

grayImage6 = grayImage6(:, :, 1); % Take green channel.

grayImage7 = grayImage7(:, :, 1); % Take green channel.

grayImage8 = grayImage8(:, :, 1); % Take green channel.

% Display the original gray scale image.

% subplot(2, 1, 1);

% imshow(grayImage, []);

% imshow(grayImage3, []);

fontSize = 20;

% title('Original Grayscale Image', 'FontSize', fontSize, 'Interpreter', 'None');

% Set up figure properties:

% Enlarge figure to full screen.

% set(gcf, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1]);

% Get rid of tool bar and pulldown menus that are along top of figure.

%set(gcf, 'Toolbar', 'none', 'Menu', 'none');

% Give a name to the title bar.

%set(gcf, 'Name', 'Demo by ImageAnalyst', 'NumberTitle', 'Off')

% Display the original gray scale image.

% subplot(2, 1, 2);

% Imag\_F=fft2(grayImage); % uncentered FFT

% Image\_F\_C=fftshift(log(1+abs(Imag\_F))); % Centered FFT

% imshow(Image\_F\_C,[]);

% title('Spectrum Image', 'FontSize', fontSize, 'Interpreter', 'None');

% Recons = [];

% for g = 1 : size(grayImage,2)

% Recons = [Recons, movmean(grayImage(:,g),[10,1])];

% Z=Recons;

% end

% figure; imshow(Recons,[])

% for h=1:5

% for g = 1 : size(grayImage,2)

% Recons(:,g) = movmean(Recons(:,g),[10,1]);

% Z=Recons;

% end

% figure; imshow(Recons,[])

% end

% Uses median and wiener filters

% Kmedian = medfilt2(grayImage);

% figure;imshowpair(grayImage,Kmedian,'montage')

% Kwi ener = wiener2(grayImage,[5 5]);

% figure;imshowpair(grayImage,Kwiener,'montage')

Kmovemean1 = movmean(grayImage1,[5 5]);

for i = 1:1

Kmovemean1 = movmean(Kmovemean1,[5 5]);

end

Kmovemean1 = imbinarize(Kmovemean1, 150);

figure;imshowpair(grayImage1,Kmovemean1,'montage')

title('Original and New Grayscale Image: p1.png', 'FontSize', fontSize);

Imag\_F1=fft2(grayImage1); % uncentered FFT

Image\_F\_C1=fftshift(log(1+abs(Imag\_F1))); % Centered FFT

Imag\_FNew1=fft2(Kmovemean1); % uncentered FFT

Image\_F\_CNew1=fftshift(log(1+abs(Imag\_FNew1))); % Centered FFT

figure;imshowpair(Image\_F\_C1,Image\_F\_CNew1,'montage')

title('Original and New Spectrum Image: p1.png', 'FontSize', fontSize);

Kmovemean2 = movmean(grayImage2,[5 5]);

for i = 1:1

Kmovemean2 = movmean(Kmovemean2,[5 5]);

end

Kmovemean2 = imbinarize(Kmovemean2, 150);

figure;imshowpair(grayImage2,Kmovemean2,'montage')

title('Original and New Grayscale Image: p1\\_2.png', 'FontSize', fontSize);

Imag\_F2=fft2(grayImage2); % uncentered FFT

Image\_F\_C2=fftshift(log(1+abs(Imag\_F2))); % Centered FFT

Imag\_FNew2=fft2(Kmovemean2); % uncentered FFT

Image\_F\_CNew2=fftshift(log(1+abs(Imag\_FNew2))); % Centered FFT

figure;imshowpair(Image\_F\_C2,Image\_F\_CNew2,'montage')

title('Original and New Spectrum Image: p1\\_2.png', 'FontSize', fontSize);

Kmovemean3 = movmean(grayImage3,[5 5]);

for i = 1:1

Kmovemean3 = movmean(Kmovemean3,[5 5]);

end

Kmovemean3 = imbinarize(Kmovemean3, 150);

figure;imshowpair(grayImage3,Kmovemean3,'montage')

title('Original and New Grayscale Image: p1\\_3.png', 'FontSize', fontSize);

Imag\_F3=fft2(grayImage3); % uncentered FFT

Image\_F\_C3=fftshift(log(1+abs(Imag\_F3))); % Centered FFT

Imag\_FNew3=fft2(Kmovemean3); % uncentered FFT

Image\_F\_CNew3=fftshift(log(1+abs(Imag\_FNew3))); % Centered FFT

figure;imshowpair(Image\_F\_C3,Image\_F\_CNew3,'montage')

title('Original and New Spectrum Image: p1\\_3.png', 'FontSize', fontSize);

Kmovemean4 = movmean(grayImage4,[5 5]);

for i = 1:2

Kmovemean4 = movmean(Kmovemean4,[5 5]);

end

Kmovemean4 = imbinarize(Kmovemean4, 150);

figure;imshowpair(grayImage4,Kmovemean4,'montage')

title('Original and New Grayscale Image: p1\\_4.png', 'FontSize', fontSize);

Imag\_F4=fft2(grayImage4); % uncentered FFT

Image\_F\_C4=fftshift(log(1+abs(Imag\_F4))); % Centered FFT

Imag\_FNew4=fft2(Kmovemean4); % uncentered FFT

Image\_F\_CNew4=fftshift(log(1+abs(Imag\_FNew4))); % Centered FFT

figure;imshowpair(Image\_F\_C4,Image\_F\_CNew4,'montage')

title('Original and New Spectrum Image: p1\\_4.png', 'FontSize', fontSize);

Kmovemean5 = movmean(grayImage5,[5 5]);

for i = 1:3

Kmovemean5 = movmean(Kmovemean5,[5 5]);

end

Kmovemean5 = imbinarize(Kmovemean5, 150);

figure;imshowpair(grayImage5,Kmovemean5,'montage')

title('Original and New Grayscale Image: p1\\_5.png', 'FontSize', fontSize);

Imag\_F5=fft2(grayImage5); % uncentered FFT

Image\_F\_C5=fftshift(log(1+abs(Imag\_F5))); % Centered FFT

Imag\_FNew5=fft2(Kmovemean5); % uncentered FFT

Image\_F\_CNew5=fftshift(log(1+abs(Imag\_FNew5))); % Centered FFT

figure;imshowpair(Image\_F\_C5,Image\_F\_CNew5,'montage')

title('Original and New Spectrum Image: p1\\_5.png', 'FontSize', fontSize);

Kmovemean6 = movmean(grayImage6,[5 5]);

for i = 1:3

Kmovemean6 = movmean(Kmovemean6,[5 5]);

end

Kmovemean6 = imbinarize(Kmovemean6, 150);

figure;imshowpair(grayImage6,Kmovemean6,'montage')

title('Original and New Grayscale Image: p1\\_6.png', 'FontSize', fontSize);

Imag\_F6=fft2(grayImage6); % uncentered FFT

Image\_F\_C6=fftshift(log(1+abs(Imag\_F6))); % Centered FFT

Imag\_FNew6=fft2(Kmovemean6); % uncentered FFT

Image\_F\_CNew6=fftshift(log(1+abs(Imag\_FNew6))); % Centered FFT

figure;imshowpair(Image\_F\_C6,Image\_F\_CNew6,'montage')

title('Original and New Spectrum Image: p1\\_6.png', 'FontSize', fontSize);

Kmovemean7 = movmean(grayImage7,[5 5]);

for i = 1:3

Kmovemean7 = movmean(Kmovemean7,[5 5]);

end

Kmovemean7 = imbinarize(Kmovemean7, 150);

figure;imshowpair(grayImage7,Kmovemean7,'montage')

title('Original and New Grayscale Image: p1\\_7.png', 'FontSize', fontSize);

Imag\_F7=fft2(grayImage7); % uncentered FFT

Image\_F\_C7=fftshift(log(1+abs(Imag\_F7))); % Centered FFT

Imag\_FNew7=fft2(Kmovemean7); % uncentered FFT

Image\_F\_CNew7=fftshift(log(1+abs(Imag\_FNew7))); % Centered FFT

figure;imshowpair(Image\_F\_C7,Image\_F\_CNew7,'montage')

title('Original and New Spectrum Image: p1\\_7.png', 'FontSize', fontSize);

Kmovemean8 = movmean(grayImage8,[5 5]);

for i = 1:3

Kmovemean8 = movmean(Kmovemean8,[5 5]);

end

Kmovemean8 = imbinarize(Kmovemean8, 150);

figure;imshowpair(grayImage8,Kmovemean8,'montage')

title('Original and New Grayscale Image: p1\\_8.png', 'FontSize', fontSize);

Imag\_F8=fft2(grayImage8); % uncentered FFT

Image\_F\_C8=fftshift(log(1+abs(Imag\_F8))); % Centered FFT

Imag\_FNew8=fft2(Kmovemean8); % uncentered FFT

Image\_F\_CNew8=fftshift(log(1+abs(Imag\_FNew8))); % Centered FFT

figure;imshowpair(Image\_F\_C8,Image\_F\_CNew8,'montage')

title('Original and New Spectrum Image: p1\\_8.png', 'FontSize', fontSize);

% Recons = [];

% for g = 1 : size(Kwiener,2)

% Recons = [Recons, movmean(Kwiener(:,g),[10,1])];

% Z=Recons;

% end

% figure; imshow(Recons,[])

% for h=1:20

% for g = 1 : size(Kwiener,2)

% Recons(:,g) = movmean(Kwiener(:,g),[10,1]);

% Z=Recons;

% end

% figure; imshow(Recons,[])

% end

# Conclusion

The adequate process for removing noise appears to be applying the movmean() filter multiple times on a range from 1-3 depending on the noise severity, and then applying the binarize filter. However, there is a tradeoff. While the noise is removed and the complex equations appear easier to read, the elements of the equations become slightly blurred. This blurred comes from the movemean() filter, but the filter is kept since it helps reduce the noise of the images and better prepare these images for the binarize filter.