**Lab 3: Filter Design and Image Restoration in Frequency Domain**

****

Jonathan Tjong 20723414

William Li 20720929

**SYDE 575 - University of Waterloo**

Submitted To:Prof David Clausi

Due Date:Oct 29, 2021

# Introduction

In this third image processing lab, we explore working with images in the frequency domain. Using matlab, we generated images then processed and analyzed fourier spectra of those images. We used the *fft2* and the *fftshift* function in matlab to produce our fourier spectras. We then experimented reconstructing images from the fourier spectra using just the amplitude and phase portions of the fourier spectra. Next we explored noise filtering techniques in the frequency domain. We analyzed fourier spectras and applied low-pass filters and a gaussian filter to the spectra. Transforming the filtered spectra to the spatial domain we analyzed the effects of each filter and calculated the PSNRs. Finally, we examined an image with periodic noise. By analyzing the fourier spectra, we were able to implement a notch filter design to effectively remove this periodic noise from the image.

Fourier Analysis

| White Bar | Fourier Spectra |
| --- | --- |

**1. What can you say about the general distribution of energy in the Fourier spectra? Why?**

The energy distribution is more intense in the low frequencies of the image due to the solid white mass in the spatial image. This means the information within the white bar is unchanging and thus appears in the spectra as a concentrated energy distribution within the low frequencies. The energy distribution is also only present along the u axis, passing through the origin. This is because the solid white bar is the same across the entire y-axis. There is also some energy present along the far ends of the u-axis. This is due to the sharp edges along the shape and the high contrast between the white shape and the black background.

**2. What characteristics about the test image can you infer from the Fourier spectra?**

Since the Fourier spectra has no frequency along the v-axis, I can infer that the spatial domain image does not change along the y-axis. Since the Fourier spectra only contains information along the u-axis itself, I can also infer that the image changes in the same way across the x-axis. Meaning, for every row of spatial data, it will be changing in the exact same manner. Since there is also information along the high ends of the frequency domain, I can infer that there are sharp edges or areas of large contrast in the spatial image.

| White Bar Rotated 45° | Fourier Spectra |
| --- | --- |

**3. How did the Fourier spectra change from the original image (before rotation)?**

The Fourier spectra rotated, with the main distribution of energy occurring along the 45 degree angle between the v and u axis. With the rotated bar, there are now changes along the v and u axis, representing changes along the x and y axis in the rotated image.

Since the image dimensions also changed during rotation, there are now black spaces surrounding the entire rotated bar. This appears in the fourier spectra as additional information along the 90° complement to the original spectra.

**4. What conclusions and observations can be made about image characteristics based on the Fourier spectra of both original image and the rotated image?**

Both the original and rotated spectra images contain most of their energy in the low frequencies. This represents the significant amount of solid intensities in both of the spatial images. Both spectras also show information along the highest frequencies, this represents the sharp edges and contrast between the solid white shapes and the black background.

| Amplitude Reconstruction | Phase Reconstruction |
| --- | --- |

**5. Describe how the reconstructed image from the amplitude component look like? What image characteristics does the amplitude component capture? (Hint: apply log to the result of the inverse fft).**

The resulting image from the reconstruction of the amplitude component looks like a series of splotches or paint brush strokes. The amplitude component by itself doesn’t carry any meaningful information about the image characteristics. Since the amplitude is a result of the square root of the real and imaginary components squared, no information can be regained from just the amplitude. When performing the inverse fourier transform, the amplitude is set to the real component and the imaginary is set to 0. This loss of information means the resulting sinusoids all begin from the same phase. Thus, the reconstructed image bears no resemblance to the original.

**6. Describe how the reconstructed image from the phase component look like? What image characteristics does the phase component capture?**

In the reconstructed image from the phase component, we can see a faint outline of the structure in the original image. This is because the phase component contains vital information about where the sinusoids are beginning and ending. Reconstructing the spatial image from phase alone will still reveal the original structure of the image.

# Noise Reduction in the Frequency Domain

| **Original image** | **Original Spectra** |
| --- | --- |

| **Noisy image** | **Noisy Spectra** |
| --- | --- |

**7. Compare the two Fourier spectra. What are the differences? Where are these differences most visually prominent? Why?**

Looking at the Fourier spectra of the original image, there are some clear frequency domain characteristics that can be observed. There is stronger frequency content in strictly the horizontal and vertical directions, since there is evidently higher gains/intensities on the u and v axes in the spatial frequency domain. However, this is not quite as visible in the Fourier spectra of the noisy image.  
Also, in the spectra of the noise-free image, it is clear to see that the low frequencies near the origin are the most prominent, and as we move outward to higher and higher frequencies, the amplitude diminishes. Higher frequencies in all directions are much weaker in comparison to low frequencies. Again, this trend is not quite as visible in the noisy spectra, as the entire spectra maintains more of a consistent grey colour all over.

These discrepancies can be attributed to the effect that gaussian noise has in the spatial frequency domain. In general, noise is considered high frequency content. Therefore adding gaussian noise to the image adds high frequency content in all directions, which means that the resulting spectra is a lot brighter overall in the areas that used to be darker. The image has stronger high frequencies where it used to be weak, and as well the frequency content in the horizontal and vertical directions are no longer as strong in comparison to the rest of the spectra.

| **Denoised image using ideal LP filter (radius = 60)**    PSNR: 27.8865 |
| --- |

**8. Describe the appearance of the denoised image compared to the original and the noisy images. Why does it look this way? What does the ideal low-pass filter do?**

In comparison to the original image, the denoised image looks much blurrier and less sharp.

In comparison to the noisy image, the denoised image does not have as visually evident gaussian noise.  
An ideal low-pass filter cuts off all high frequency components at a distance greater than radius = 60 from the origin. As a result, both sharp edges as well as noise are severely attenuated. This is why the denoised image looks blurrier and less noisy.

**9. There is a particular artifact present in the restored image. What is it and why does it happen?**

The artifact present in the restored image is ringing. Ringing can be seen as the oscillations leftover near the edges after applying the low-pass filter. For example, at the sharp edge at the back side of Lena’s hat, we see the ringing artifact which is the repeated contour edge of the hat. Ringing occurs because the inverse Fourier transform of a low-pass filter is a sinc function, which has oscillations on both sides. As a result, convolving the sinc function with an edge generates oscillations or ringing around the edge.

| **Denoised image using ideal LP filter (radius = 20)**    PSNR: 23.114 |
| --- |

**10. Compare the denoised image with the denoised image using a cut-off radius of 60. How does the image and the PSNR differ? Why?**

When using a cut-off radius of 20 for the low-pass filter, the image becomes much blurrier in comparison to using a cut-off radius of 60. This is because we are cutting out even more higher frequency components and leaving in only extremely low frequency components. Edges become even less sharp and the image resembles more of a solid shade in areas. As well, ringing becomes more and more evident as the cut-off frequency decreases.  
The denoised image using a cut-off radius of 60 performed better in terms of PSNR since using a cut-off radius of 20 resulted in over blurring. The high amount of blurriness resulted in a poor reconstruction of the original image, thus the PSNR was not as high.

**11. What conclusions can you draw about the relationship between cut-off radius and resulting image after filtering? What is the trade-off in terms of noise reduction?**

As cut-off radius decreases, the resulting image becomes more blurry and less sharp. As well, ringing increases. However, noise becomes more reduced due to the higher smoothing. Therefore, there is a trade-off between noise reduction vs image blurring and ringing.

| **Denoised image using gaussian LP filter (std. dev. = 60)**    PSNR: 29.5764 |
| --- |

**12. Compare the denoised image with the denoised images produced using the ideal low-pass filters. How does the image and the PSNR differ? Is it better or worse? Why? Does it have the same type of image artifacts?**

Using the gaussian low-pass filter arguably outperformed the ideal low-pass filters since the noise was reduced significantly, however there are no ringing artifacts. Visually, the image looks smoothed slightly, however edges are still clearly distinguishable without any oscillatory artifacts. The image resulting from the gaussian filter is clearly of higher quality in comparison to using the ideal filters through both the eye test and as well as PSNR (has the highest PSNR of all three reconstructions). Using the gaussian low-pass filter produced the best reconstruction of the original image.

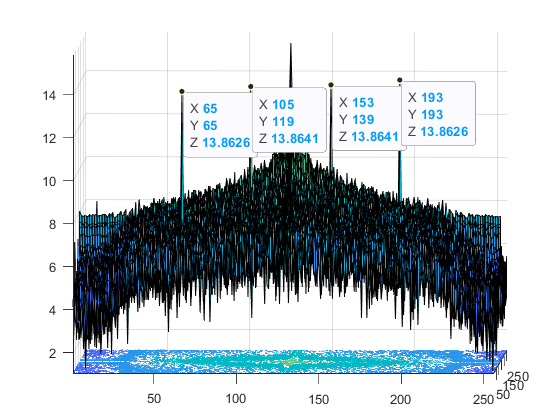
# Filter Design

The given noisy image, frequnoisy.tif, contains evident periodic noise. This periodic noise is seen as consistent diagonal striations that severely obscures the image. Examining the Fourier spectra of the noisy image, we can see the periodic noise appears in the form of four individual white spots/pixels. These four pixels can be considered as two pairs of symmetrical spots. The reason why the periodic noise is observed as such in the spatial frequency domain is because the Fourier transform of a cosine function is the sum of two impulses.

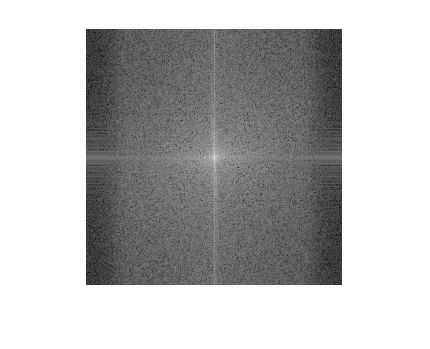
| Original noisy image | Fourier spectra of noisy image |
| --- | --- |

In order to filter out this noise, a simple notch reject filter can be applied in the frequency domain to reject these bright spots in the spectra. This can be done by locating the bright spots/pixels, and creating a filter with values of zero in these locations.

In order to locate the locations of the white pixels in the spatial frequency domain, the spectra was graphed using a surface plot. The four white spots are clearly observable as spikes in the plot, and using the MATLAB figure tools the pixels could be located:



Afterwards, a simple filter was created using ones(256, 256) (256x256 is the same size as the image) and then setting the pixels at the four locations to 0. After applying the filter to the image spectra using element-wise multiplication, the resulting spectra is unaltered except for the four individual pixels, which have now been rejected and set to black.



Thus, the periodic noise has been eliminated from the image. Taking the inverse fourier transform of this filtered spectra gives the reconstructed original image:



Evidently the noise has been significantly reduced thanks to the filter, and the tree in the image is much clearer.

# 

# Conclusion

In this lab, we explored working with images in the frequency domain.

In the first section, we used matlab to generate an image of a solid white bar on a black background. Using matlab’s *fft2* and *fftshift* function we were able to plot the resulting fourier spectra. We then manipulated the original image by rotating it 45° and plotted the new fourier spectra. Analyzing both fourier spectra helped support our understanding of what the fourier spectra represented.

We then experimented reconstructing images from the fourier spectra using just the amplitude and phase portions of the fourier spectra. We used matlab to calculate the discrete fourier transform of an image and then isolated the amplitude and phase. We found the amplitude by taking the absolute value of the fourier transform, and then calculated the phase by taking the original fourier transform and dividing by the amplitude. Using the inverse fourier transform we were able to visualize new spatial images from the amplitude and phase information alone. We found that the phase recreation retained the structural information, but the amplitude recreation had no meaningful data.

Next we explored noise filtering techniques in the frequency domain. We first created a gaussian noised image and analyzed it’s fourier spectra. We then experimented applying low-pass filters of varying sizes and a gaussian filter to the spectra. Transforming the filtered spectras to the spatial domain we analyzed the effects of each filter and calculated the resulting PSNRs. The low-pass filters had various degrees of success depending on the cut-off range of frequencies determined by the size of the filter. However, the low-pass filters created a ringing artifact in the reconstructed images. The gaussian filter was empirically determined to be the best due to its high PSNR value.

Finally, we examined an image with periodic noise. By analyzing the fourier spectra, we were able to identify the locations of this periodic noise. We then implemented a notch-design filter with masks at the exact locations of the periodic noise. Multiplying this filter with the fourier spectra, we were able to perform the inverse fourier transform on the filtered spectra and recover the original image without noise.

# Appendix

**Part 2**

close all;

f = zeros(256,256);

f(:,108:148)=1;

f=imrotate(f, 45);

figure(1)

imshow(f)

fourier\_spectra = fft2(f);

shift = fftshift(fft2(f));

a = abs(fftshift(fft2(f)));

figure(2)

imshow(log(abs(fftshift(fft2(f)))),[])

lena = imread('lena.tiff');

lena = rgb2gray(lena);

amplitude = abs(fftshift(fft2(lena)));

phase = fftshift(fft2(lena))./amplitude;

re = ifft2(ifftshift(amplitude));

m = max(re(:))

%

% figure(1)

% imshow(lena)

%

% figure(2)

% imshow(ifft2(ifftshift(amplitude)), []);

%

% figure(3)

% imshow(ifft2(ifftshift(phase)),[]);

**Part 3**

lena = imread('lena.tiff');

lena = rgb2gray(lena);

lena = double(lena) ./ 255;

noisy = imnoise(lena, 'gaussian', 0, 0.005);

noisy\_freq = (fftshift(fft2(noisy)));

% imshow(noisy\_freq, []);

figure(1)

imshow(noisy);

% figure(2)

% imshow(log(abs(fftshift(fft2(noisy)))),[]);

% figure(3)

% imshow(log(abs(fftshift(fft2(lena)))),[]);

r = 60;

h = fspecial('disk', r);

h(h>0)=1;

height = 512;

width = 512;

h\_freq = zeros(height,width);

h\_freq(height/2-r:height/2+r, width/2-r:width/2+r) = h;

g = fspecial('gaussian', 512, 60);

m = max(g(:))

g = g ./ max(g(:));

figure(3)

imshow(g)

filtered = noisy\_freq .\* g;

re = abs(ifft2(ifftshift(filtered)));

psnr(re, lena)

figure(2)

imshow(re, [])

**Part 4**

close all;

freq\_noise = imread('frequnoisy.tif');

figure;

imshow(freq\_noise);

% Fourier spectra for noisy image

spectra = fftshift(fft2(freq\_noise));

figure;

imshow(log(abs(spectra)), []);

figure;

surfc((log(abs(spectra))));

% % Radius for notch reject filter

% r = 1;

% h = fspecial('disk', r);

% h(h==0) = 1;

% h(h<1) = 0;

%

% % Image size is 256 x 256

% h\_freq = ones(256, 256);

% % Create frequency reject filter

% h\_freq(118-r:118+r, 105-r:105+r) = h;

% h\_freq(139-r:139+r, 152-r:152+r) = h;

% h\_freq(64-r:64+r, 64-r:64+r) = h;

% h\_freq(192-r:192+r, 192-r:192+r) = h;

% figure;

% imshow(h\_freq);

% Image size is 256 x 256

h\_freq = ones(256, 256);

% Create frequency reject filter

h\_freq(119, 105) = 0;

h\_freq(139, 153) = 0;

h\_freq(65, 65) = 0;

h\_freq(193, 193) = 0;

figure;

imshow(h\_freq);

% Apply filter onto spectra

filtered = spectra .\* h\_freq;

figure;

imshow(log(abs(filtered)), []);

% reconstructed

reconstructed = ifft2(ifftshift(filtered));

figure;

imshow(abs(reconstructed), []);