EECE 253 IMAGE PROCESSING

LABORATORY ASSIGNMENT 4

Jack Minardi
20 Oct, 2011

Abstract

This paper reports the results of experiments done to explore convolutions and other frequency domain manipulations on images in Matlab. Moving into the frequency domain can be helpful in extracting useful information from an image. In the lab two different methods for convolution were implemented (one in the time domain and the other in frequency) and then tested in a variety of situations. We also explored the basics of Fourier transforms by inspecting the Fourier transform of a small number of impulses. The relationship between impulses and sine gradients was noted. Gaussian lowpass and bandpass filters were also explored as a means of image manipulation. And finally image sharpening though weighted adding of the same image through a high-pass filter, and through subtraction of a low pass filtered image.

Table of Contents

Introduction

Using Fourier transforms to manipulate images give us great insight into how frequency space works and what it represents. The goals of this laboratory are to learn about the Fourier transform of an impulse, how to perform convolutions on both the spatial and frequency domain, and how to filter images via convolution. It is found that convolution in the frequency domain is much faster for all but the smallest of weight matrices. Filtering in the frequency domain can also produce some interesting results, like high and low pas filters which can be used to sharpen or blur an image.

Description of Experiments

There were five experiments performed in this lab. Matlab was used for all the computation. All the scripts that were written can be found in the appendix and in the zip file this report came in. The results were somewhat subjective and discussions can be found in the report below.

Experiment 1. Convolution Programs and Computation Speed.

Write two programs to convolve an arbitrary image, I, with a weight matrix, h. The weight matrix should of type double, m × n, single band. Both programs should work on either 3-band (truecolor) or 1-band (grayscale or monochrome) images of type uint8 or double. Each band of a 3-band image should be convolved with the same weight matrix. The output should be of

type double. (If you need a uint8 image from the convolution routine, convert it inline. E.g. J = uint8(ConvSpace(I,h));)

(a) Write a program to perform the convolution in the spatial domain. Use the shift- multiply-accumulate approach described in the lecture notes.

(b) Write a program to perform the convolution in the frequency domain by multi- plying the Fourier Transforms of the image and the weight matrix and taking the inverse transform of the result. Use Matlab's fft2 function. Recall that within your program you must allocate a temporary image the same size as I, and copy h into it so that the center of h coincides with the center of the temporary image. You must apply ifftshift to the image that contains h but not to I prior to computing the Fourier transforms. Why is that? See EECE253_06_FourierTransform.ppt, slides 63-65, for the definition of the Mat- lab image center and for examples of fftshift and ifftshift. Remember to take only the real part of the inverse transformed result. (The imaginary part may exist but it will contain only extremely low amplitude noise when filtering a real image with a real convolution mask.)

(c) Which is faster, spatial or frequency domain convolution? Write a simple test program to compare the computation times of your two programs. I suggest that you create a small constant image such as I=ones(512,512). Within the main loop of the test program, generate convolution masks of size n × n (e.g., h=ones(n,n)). Convolve I with h using both programs over a range of integers such as n = 1,2,3,...,33 and record the time it takes each one to complete. For example, to compare the computation times of programs named ConvSpace and ConvFreq over convolution masks ranging in size from 1 × 1 to 33 × 33 one could define an array t=zeros(32,2). Then within the main loop: tic; J = ConvSpace(I,h); t(n,1) = toc; tic; J = ConvFreq(I,h); t(n,2) = toc; Plot the two time sequences on the same graph. Describe how time required by the two algorithms changes as a function of the weight matrix dimensions. If the curves cross each other, indicate the values of n where that happens. What function should you use use for a given convolution mask size?

Experiment 2. Sinusoids and FTs of impulses.

(a) Find and load an R×C×3 (truecolor) image, I. Create an R×C×3 zero image, h. Place an impulse – a 1 – in all three bands at some location, (r1,c1) in the interior of h, at neither the center nor close to a corner. Convolve I with h using your Fourier transform convolution program. Display the result. Describe as precisely as possible what has happened to the image.

(b) Create a 512 × 512 × 1 image of all zeros. Place a unit impulse (a one) at the center of the image. Apply ifftshift to the image, take its Fourier transform, and apply fftshift to the result. Find the maximum and minimum values of the FT of the image. [Hint: Find the maximum

and minimum values of the four components, the real, imaginary, magnitude, and phase of the FFT using real(F), imag(F), abs(F), and angle(F).] Describe the results – what are the real, imaginary, magnitude and phase of the FT of an impulse at the origin? It should not be necessary to display these to understand what has happened. Do the same thing for an image of size 512 × 512 × 1 with value one at each location. Be sure to look at the numbers in the vicinity of the origin of the FT. Comment on the relationship between the impulse and the constant image in both the spatial and Fourier domains.

(c) Now create another 5122 image of zeros and place an impulse at some location, (r0, c0), other than the center. Find the maximum and minimum values of the FT of the image as in part (b). Contrast the results – how do the results differ? In particular, how do the power spectra differ and how do the phases differ? What mathematical property of the Fourier transform does this illustrate? Include the corresponding equation in your report. Hint: Try displaying the four components using, for example, figure,imagesc(real(F2c)),colormap(gray(256)),truesize; If it is difficult to see what is going on try placing the impulse at location (r,c) wherer=c=p+1wherep=512·2−n forsomeintegern≥1.

(d) Create a rectangular, one-band image, I, such that I has dimensions R × C × 1, where R ≥ 384 and C = 3R/2. Let r0 be any single integer from the set {8,9,...,64} and set c0 = r0. Place an impulse in I at locations (r0,c0) and (−r0, −c0) with respect to the centered origin of I. Calculate the frequency, wave- length, and orientation of the sinusoid represented by the location (r0,c0). Compute the inverse Fourier transform of I (after applying ifftshift) and dis- play its real component. Verify through direct measurement that the sinusoid has the predicted frequency, wavelength, and orientation.

(e) Create another image like the one in problem 2d. Use the same values of r0 and c0. Place four impulses in the image, one each at at locations (r0,c0), (r0,−c0), (−r0,−c0), and (−r0,c0) with respect to the centered origin of I. Compute the inverse Fourier transform of I (after applying ifftshift) and display its real component. What is the angle in degrees between the two sinusoidal components? For a given r0 what must c0 be so that the angle between the two components is 90◦ ? Generate that image and display its real component.

Experiment 3. Gaussian lowpass, highpass, and bandpass filters.

(a) Find a highly textured 24-bit, 3-band color image with dimensions of at least 512 × 512 to use throughout this problem. Display it in your report.

(b) Write a program to generate a space-domain (SD) symmetric Gaussian filter with arbitrary variance centered on a type double image of arbitrary size. Generate it from the equation at

the bottom of slide 33 in Lecture 8. The inputs to the program should be the variance of the gaussian and the size of the image to be output. Be sure that the Gaussian is centered on the Matlab fftshift center Normalize the result so that the output image sums to one.

(c) Use the program you wrote for the previous part (problem 3b) to generate three SD Gaussian filters, each with a different variance and size R × C. Space the variances so that one is small, one is medium, and large, The largest one should be on the order of 1/4 the size of the smaller of the two dimensions R and C. Apply each one to the image you selected in part and describe the results.

Experiment 4.

Generate a highpass and a bandpass filter in the FD from the Gaussians in 3c. It is most straightforward to do so in the FD and then to take the inverse FFT of the result. Compute the sums of each filter in the SD. What should the sums be? If they are not what they should be, adjust the filters to make them so. Take the real part of the inverse FFT of each and find the support of of each one in the space domain. The support is the set of pixels on which the value of the filter is greater than or equal to (1/(3.5σ))(1/256). If can find that in Matlab by displaying the image S = 255(abs(h)>(1/(3.5*sigma*256)));, where h is the space domain filter. Given the results of problem 1 determine the best domain in which to perform the filtering. Apply each filter to the image you selected in part and describe the results.*

Experiment 5. Image Sharpening Use the image from problem 4 in this exercise.

(a) Direct Sharpening. Convolve the image with a 3 × 3 Laplacian (See p. 59 of Lecture 7)to create a highpass image. Sharpen the original image by adding intensity-scaled versions of the highpass image to it. Use small, intermediate, and large scale factors between 0 and 1 and display the results. What scale factor results in a sharper image that looks natural and is not very noisy? How can you modify the HPF to get the same results with a single convolution?

(b) Unsharp Masking. Unsharp masking (USM) sharpens an image, I, by adding to it a scaled highpass image created by subtracting a Gaussian lowpass image from the original. There are two control parameters involved: the standard deviation, σ, of the Gaussian, and a scale factor, α. Let G(σ) be a Gaussian filter. Let B(σ) = I ⋆G(σ) be the lowpass filtered I. Let D (σ) = I − B (σ) be the difference image. Then the unsharp masked image is J = (1+α)D(σ)+B(σ) = D(σ)+B(σ) +αD(σ) = I+αD(σ). (1) 4 Experiment with various values of α and σ to find a natural looking, but clearly sharper version of of your image. Include versions of the USM image to show the effect

on it of varying the parameters.

Results of Experiments

Experiment 1

b)

c) The program I wrote is called testSpeed.m and it is stored in the folder along with this report. Below is a chart showing run time for each version of the algorithm (freq domain vs space domain) as the weight matrix gets larger. As you can see the spatial domain method is only faster for small weight matrices (less than about 4x4) and after that the frequency domain becomes the faster algorithm.

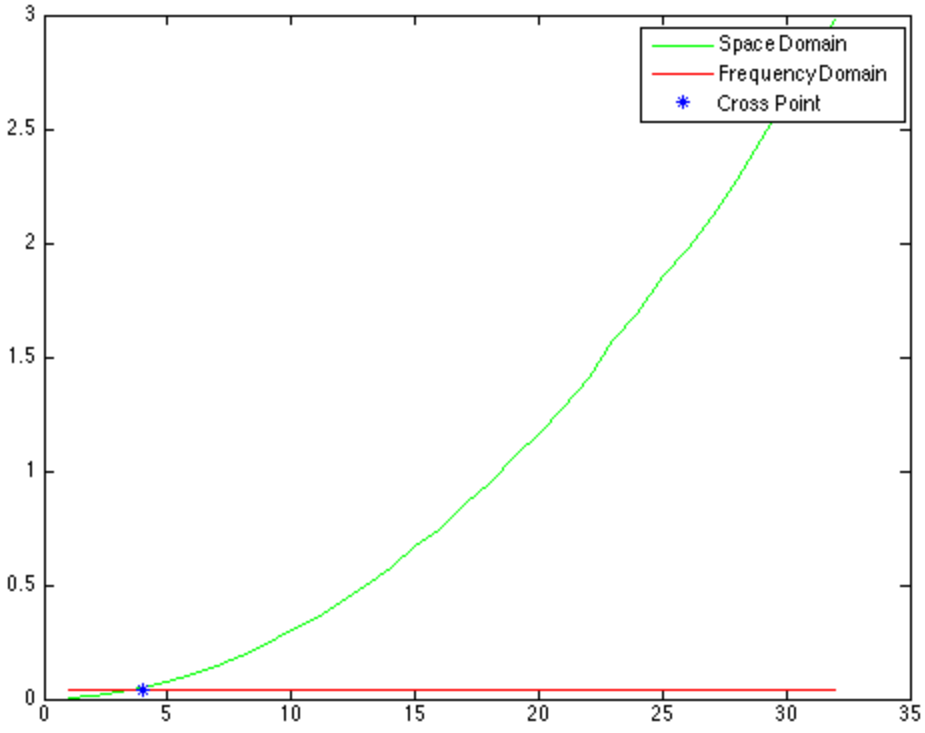

Figure 1. Graph showing run times for freq and time domain convolutions over a series of weight matrices

Experiment 2

Below you can see the image used in this report.

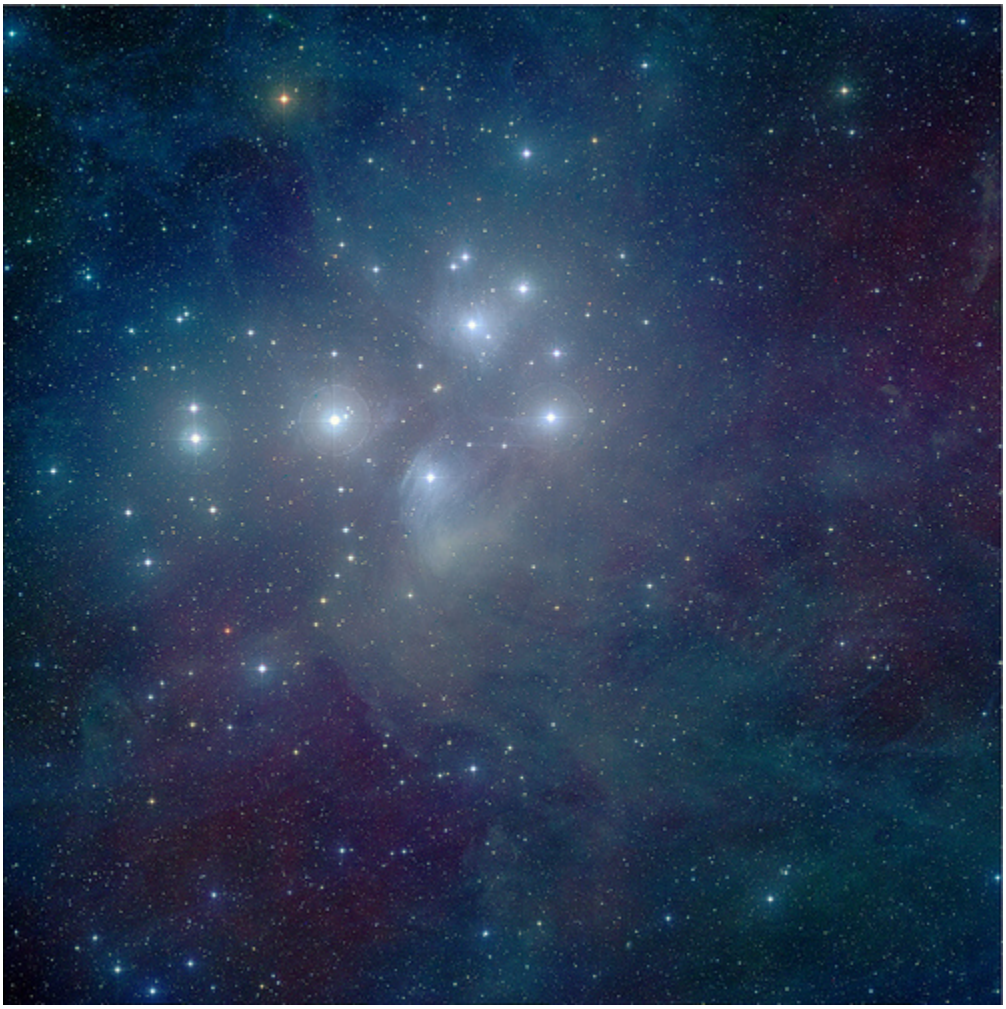

Figure 2. The image used throughout this report.
http://docakilah.files.wordpress.com/2011/06/benefitsfruit01.jpg

a) The following image was convolved with a weight matrix of the same size with all zeros except for ones at (100,150) on each of the three bands. This has the effect of shifting the image up and to the left.

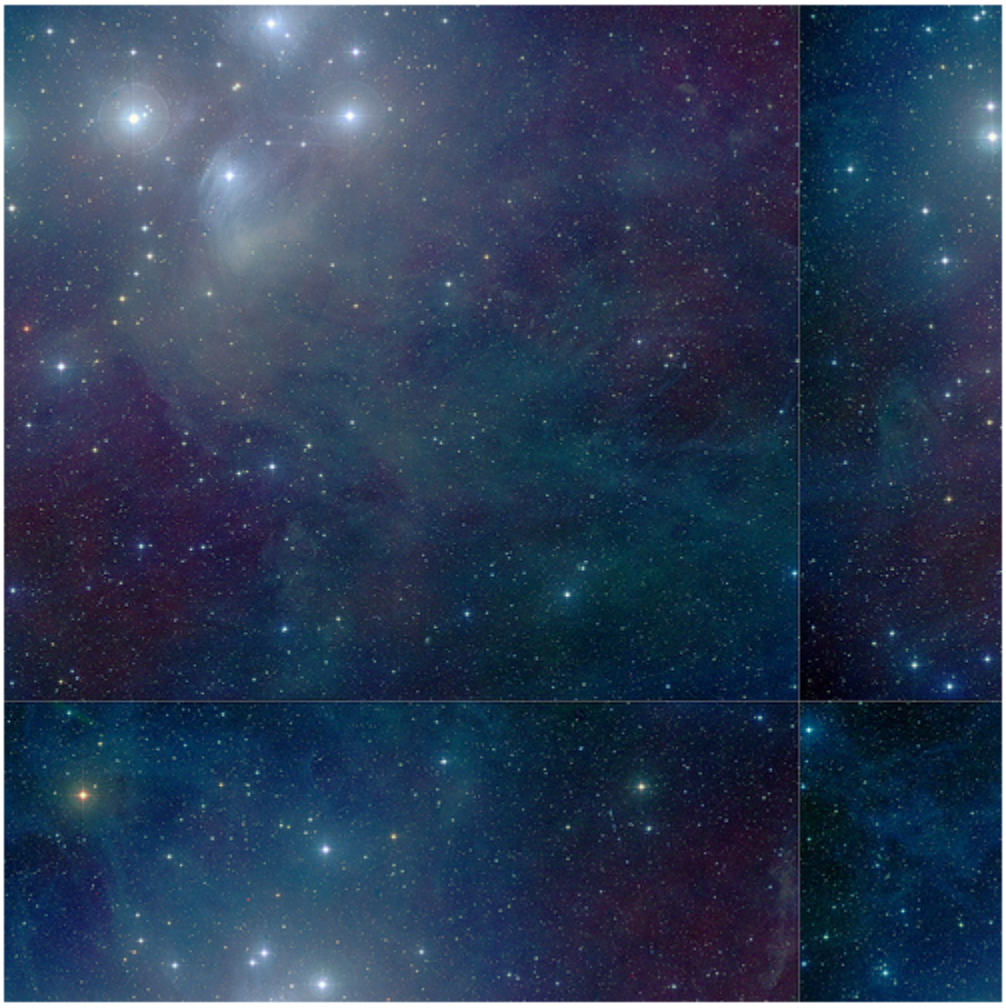

Figure 3. Image convolved with one impulse co-located on each of the bands.

b) This section involves taking the Fourier transform of an image with one impulse exactly at the center. The center value is equal to a DC value (constant) so the resulting image is pure grey. The following are the min and max of the real, imaginary, absolute, and angle.

maxreal = 1
minreal = 1
maximag = 0
minimag = 0
maxabs =1

minabs = 1
maxangl = 0
minangl = 0

With all ones (impulses everywhere) the Fourier transform of the image is one impulse exactly in the center. Again the makes sense because the Fourier transform of a DC plane is an impulse at the origin. The impulse at the origin represents 0 frequency which is DC (flat)

c) In this experiment the resulting image actually had a sinusoidal gradient. The is because one impulse not in the center represents some specific frequency and wavefront direction. The equation for finding the wavefront direction is as follows:

Theta = tan^-1(vC/uR)

This can be derived theoretically.

d) The following image is the Fourier transform of an image with two impulse at (10,10) and (-10,-10) with respect to the centered origin.

lambda_u = C/u = 577/10 = 57.7
lambda_v = R/v = 385/10 = 38.5
lambda_wf = sqrt((577/10)^2 + (385/10)^2) = 69.3653

theta_wf = (3/2)*tan^-1((10*577)/(10*385)) = 1.4736

Note the 3/2 to account for the image not being square. Below is the actual Fourier transform of the image. The above calculations can be verified in the image.

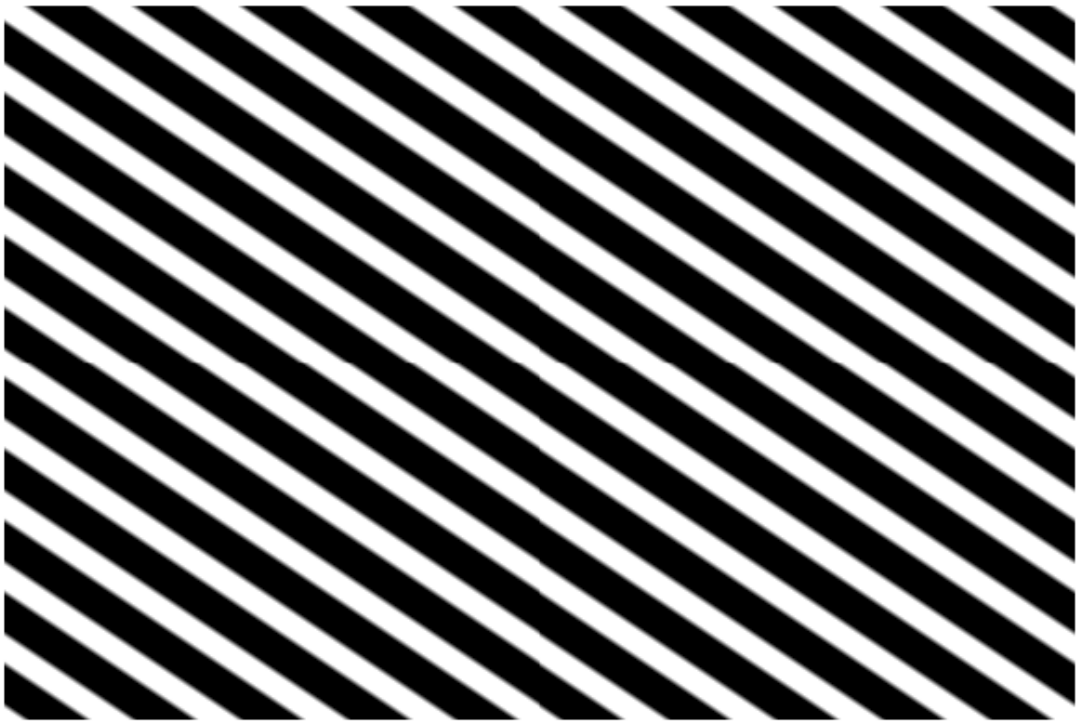

Figure 4. Fourier transform of two impulses at (10,10) and (-10,-10)

d) The following image is the real component of the Fourier transform of an image with four impulses located at (10,10), (-10,10), (-10,-10), and (10,-10).

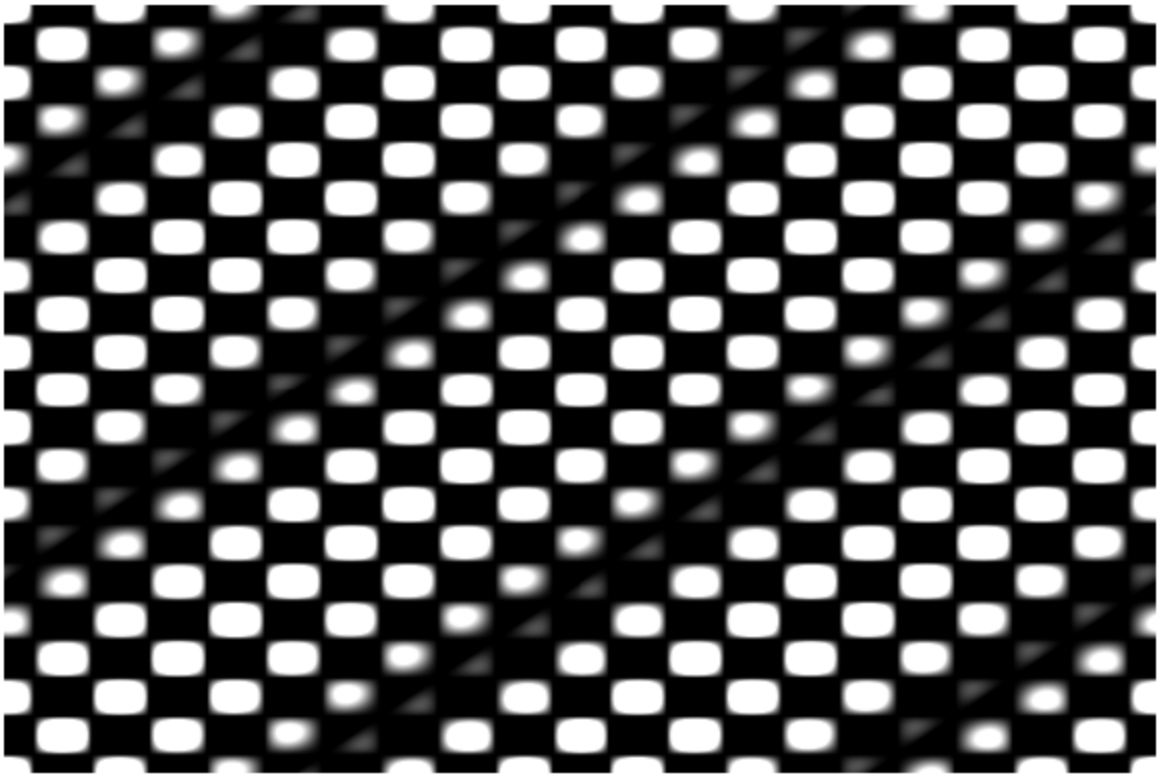

Figure 5. Fourier transform of an image with 4 impulses.

For a given r, c must be equal to it for the angle between the two components to be 90 degrees.

Experiment 3

c) The following images are 2 dimensional Gaussian distributions with STDEVs of 32, 64, and 128.

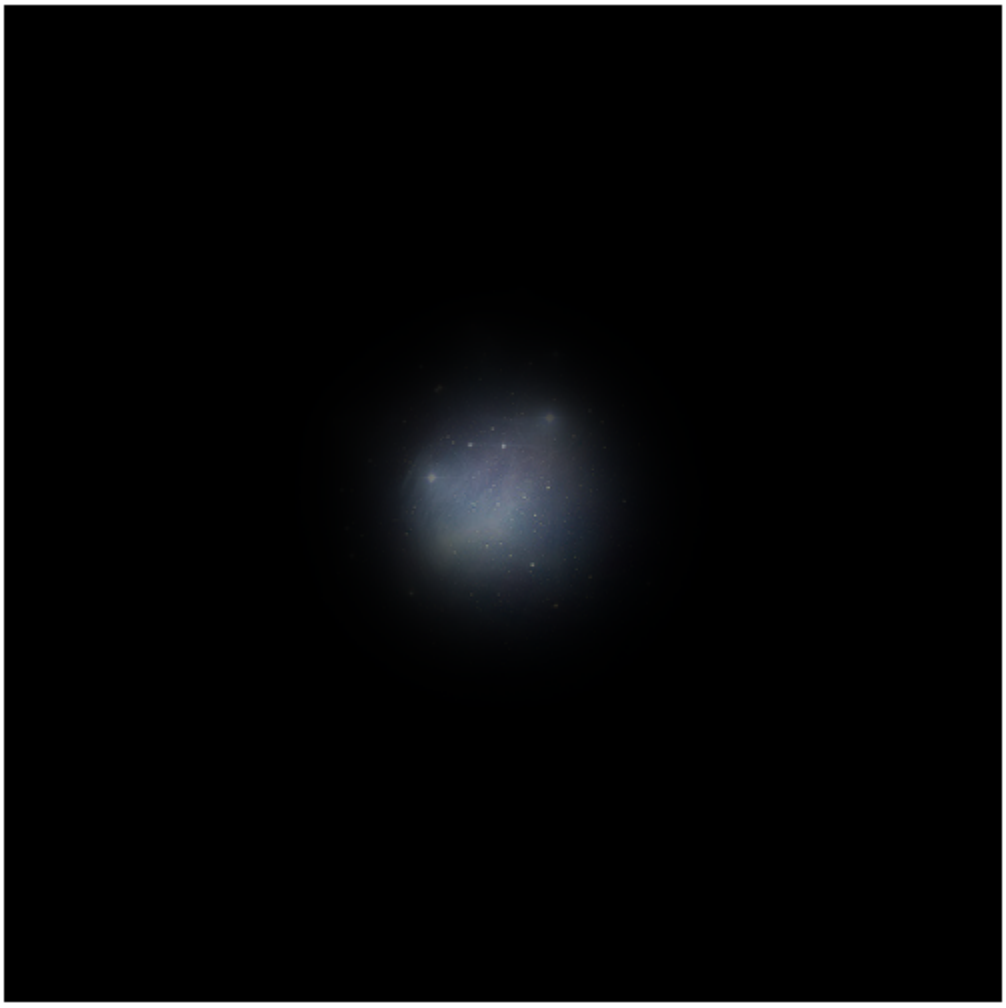

Figure 6. 2d Gaussian distribution with STDV of 32

Figure 7. 2d Gaussian distribution with STDV of 64

Figure 8. 2d Gaussian distribution with STDV of 128

Experiment 4

This experiment involved taking a highpass and bandpass filter and applying it in the frequency domain. The gaussians generated in 3c were used. The sums of each filter should be one, but they were not. Each filter was scaled to account for this. This filtering can be applied in either the frequency domain or the time domain. Given the results of experiment one, with images of this size it would be best to work in the frequency domain as it is much faster.

When the high pass filter is applied to the image is separates out only the high

frequencies. In these images that is the stars.

Experiment 5

a) This experiment had to do with the sharpening of images. The following three images are the sum of the original image and the original image convolved with a 3x3 lapacian. This has the effect of adding back in the high frequency portions of the image.

Figure 9. HF content added back in at .1 * value

Figure 10. HF content added back in at .3 * value

Figure 11. HF content added back in at .7 * value

With this image the smallest value of .1 looks the best. However this image only has every low frequencies (background color blobs) and very high frequences (stars) so other images might produce different results. If you want these same results, multiple the image mask (the laplacian in this case) by .1 before you do the convolution.

b) This section involved unsharp masking. I was not able to produce suitable results that worked in the manner described.

Conclusion

Each of these experiments explored the properties of Fourier space. The Fourier transform of an impulse was shown to simple be a sinusoidal gradient with varying frequency depending on where the impulse is placed. This makes sense as one impulse represents one specific grating. Going into the frequency domain also greatly speeds up most convolutions by simple transforming them into a multiplication, which Matlab is greatly optimized for. These convolutions are then used to filter images by either blurring them or using a high pass filter to sharpen an image.