CMPEN/EE455: Digital Image Processing I

Computer Project # 3

Filtering using Frequency-Domain Analysis

Ziheng Fu, Mei Ya, Songmeng Wang

Date: 10/27/2019

Objectives

The objective of this project is to get familiar with Fast Fourier Transform (FFT) and Discrete Fourier Transform (DFT). Students could see the relations between the filter and generated image through frequency domain analysis. Similar operations are required to filter a corrupted image. Though this project, students would obtain the basic knowledge of DFT based frequency analysis and image recovering.

Methods

Part 1

(a) F (0,0) is set to zero in order to see the difference brought by such changes. To Do this, we have to first get the frequency figure *F(u,v)* of the original image *f(x,y)* by DFT, setting the point *F(1,1) = 0* (since MATLAB start with 1, so *F(0,0)* in MATLAB denoted as *F(1,1)*). Then we could get a new image *g(x,y)* by inverse DFT of the modified frequency figure *F(u,v).* The MATLAB environment provides the functions *fft()* and *ifft()* to compute the discrete fourier transform and its inverse. With 2-D images, *F = fft2(f, M, N)* is used DFT and *f = ifft2(F, M, N)* is used for inverse DFT (M,N denoted for the size of the image M*N).

(b) First, a gaussian lowpass filter *H(u,v)* is used to filter the original image *f(x,y)* with pass width = 15. To build this lowpass filter, a function *lpfilter(type, M, N, sig)* is created, where type denoted for types of filter including gaussian, ideal; M and N denoted for the size of the image *M*N*; sig denoted for cutoff frequency. We first build a M*N array for the filter by creating another function *dftuv(M,N)*. This array should be a M*N size array containing two vectors, we denote it as *[V,U]*. We use the meshgrid function: *[V,U] = meshgrid(v,u).* Here returns a 2D grid coordinates based on the coordinates contained in vector v and u. V is a matrix where each row is a copy of v and U is a matrix where each column is a copy of u. The range of v,u is limited by the given M and N. For gaussian lowpass filter, we have the equation $G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$ where where *x* is the distance from the origin in the

horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution.[1] To set up the gaussian filter, we have the filter width sig. Our filter *H(u,v)* = $e^{(-\frac{\sqrt{|u|^2+|v|^2}}{2*sig^2})}$. Mesh function is used to give a clear 3-D filter of the filter *H(u,v)* with solid edge color. The value of *H(u,v)* is plot as height and u,v is used as coordinates. Filter *H(u,v)* is applied to the original image *f(x,y)* using multiplication of the DFT. *F(u,v)* is denoted as the DFT frequency figure of original image *f(x,y)*. The filter *H(u,v)* is then used to multiply each *(u,v)* value. The new frequency figure *G(u,v)* is computed with *G(u,v) = F(u,v) * H(u,v)*. The new image *g(x,y)* could easily be derived using inverse DFT. The Fourier-transform magnitude figure, denoted as |*G(u,v)*|, is displayed by modulation (multiplication) and scaling since |*G(u,v)*| itself would be too small to see. Equation used to display the magnitude figure is: $log(1 + |DFT\ of\ (g(x,y) * (-1)^{x+y})|)$.
Thus, both new image *g(u,v)* and magnitude figure |*G(u,v)*| could be displayed.

(c) Zero-padding extend the frequency figure with zeros. To get this done, we double the image size by multiplying 2 to M and N. With the new M and N, operations in part (b) is repeated in order to display the zero-padding figures.

Part 2

(a) For digital image $c(x,y) = f(x,y) + 32 \cdot \cos(\frac{2\pi 32y}{N})$, the noise has the form as shown in (1):

$$n(x,y) = 32\cos\left(\frac{2\pi 32y}{N}\right) \tag{1}$$

Where N is 512.

We need to design a notch filter H(u, v) to filter out the noise $32 \cdot \cos(\frac{2\pi 32y}{N})$ with N=512.

First of all, we write the cos() function in its exponential form as shown in (2):

$$32 \cdot \cos\left(\frac{2\pi 32y}{N}\right) = 16e^{j\frac{2\pi}{N}32y} + 16e^{-j\frac{2\pi}{N}32y} \tag{2}$$

Now, $e^{-j\frac{2\pi vy}{N}}$ is periodic in v with period N = 512, or:

$$e^{-j\frac{2\pi vy}{N}} = e^{j\frac{2\pi Ny}{N}}e^{-j\frac{2\pi vy}{N}} = e^{j\frac{2\pi(N-v)y}{N}}, where\ e^{j\frac{2\pi Ny}{N}} = 1 \tag{3}$$

Therefore, the noise can be rewritten as:

$$n(x,y) = 32\cos\left(\frac{2\pi 32y}{N}\right) + 32\cos\left(\frac{2\pi 480y}{N}\right) \quad (4)$$

It's apparent that the noise has two frequency components: 32 and 480. When we design the notch filter, we first create a 2-D array with for loops, then set H(u, v) = 0 whenever v = 32 or v=480; H(u, v) = 1 for all other v values. Note the noise only involves v direction, so we do not need to consider the u values. Further, since the index of MATLAB starts at 1, we have to add 1 to the frequencies, that is:

$$H(u,v) = \begin{cases} 0 & if\ v = 33\ or\ v = 481 \\ 1 & otherwise \end{cases} \quad (5)$$

To get the filtered image, we multiply the Fourier Transform of the noisy image c(x, y) with the notch filter H(u, v), then get its inverse Fourier Transform using the code "real (ifft2(·))".

b)

In order to display c(x, y), we need to clip [limit] c(x, y)'s values to the [0,255] range. To achieve this goal, we write for loops to create a 2-D array, then use if statements to set c(x, y) = 0 if c(x, y) < 0 and c(x, y) = 255 if c(x, y) >255. To get the Fourier Transform, we use the code "fft2($double(\cdot)$)". Further, when we plot the DFT magnitudes, we first modulate the function by multiplying it by $(-1)^{(x+y)}$, then get its Fourier Transform and scale its Fourier Transform using log(1+|$Fourier\ Transform$|).

Result

Part 1:

a):

The effect of setting F(0,0) to be zero is shown by figure 1 and figure2:

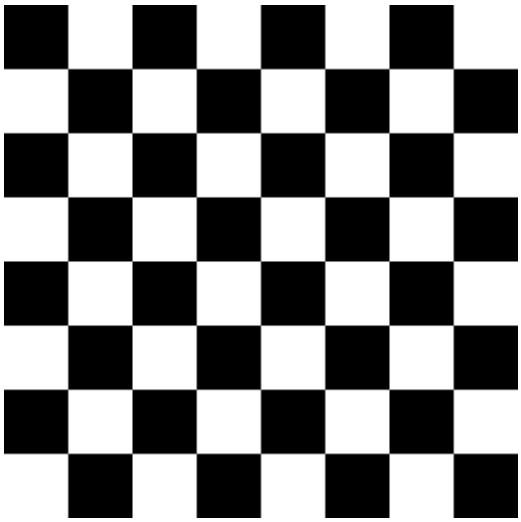

Figure 1: The original image of checker

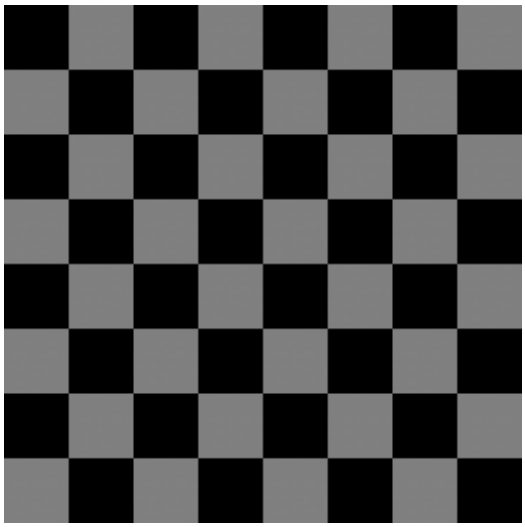

Figure 2: F(0,0) is set to be 0

Figure 2 is darker than figure 1 and the white squares whose gray value is 1 in figure 1 becomes gray squares whose gray scale value is 0.5 in figure 2 and the gray scale value of the black squares in figure also decreased by 0.5.

To interpret the effect of setting F(0,0) to be zero, the DC component of figure 1 is set to be zero and other frequency components are retained. As a result, the DC components of figure 1 is lost and g cannot be a completed reconstructed image of f.

b):

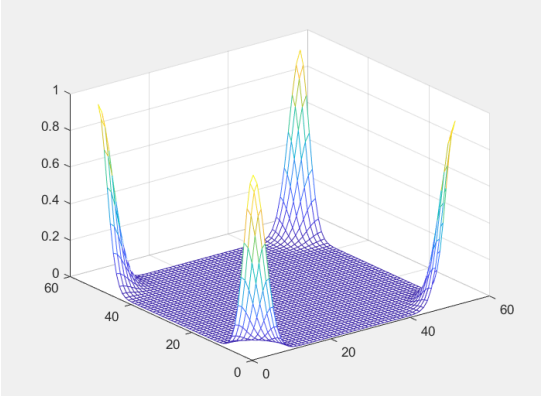

Figure 3: 3-D plot of |H(u,v)|

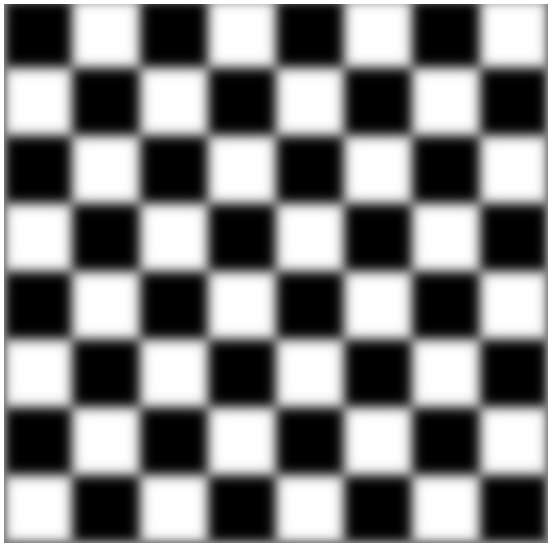

Figure 4: g(x,y)

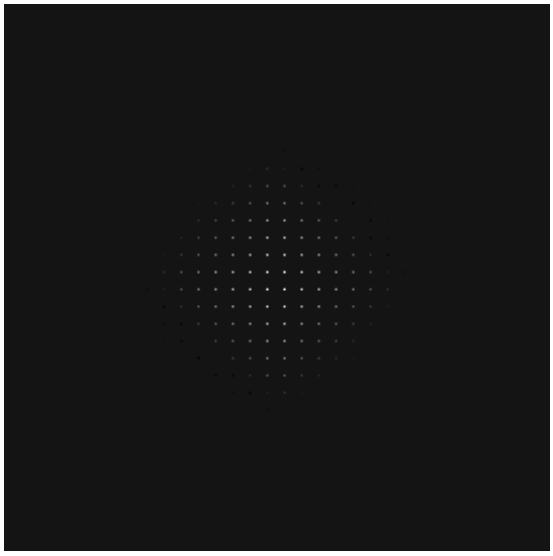

Figure 5: G(u,v)

c):

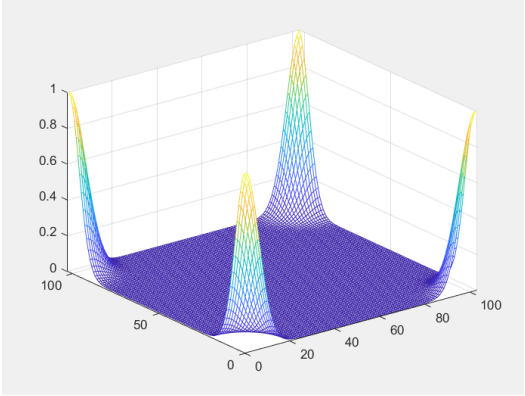

Figure 6: 3-D plot of |H(u,v)|

Figure 7: g(x,y) with zero padding

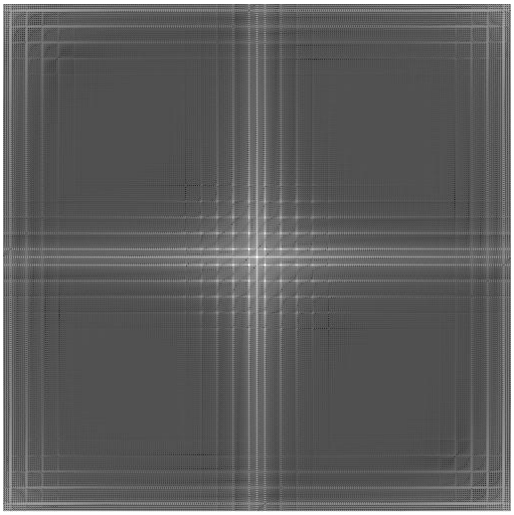

Figure 8: G(u,v) with zero padding

d):

From the results, it can be shown that the low-pass filter blurs the image in its space domain. The modulation leads to:

$$f(x,y)(-1)^{(x+y)} -- \ F(u-\frac{N}{2}, v-\frac{N}{2})$$

And the DFT will be on the middle of the screen.

The scaling operation is to concentrate the DFT image near F(0,0) and swap all other frequency components so that the image could be clear and concentrated.

The wrapped error is observed when the image is zero padded as shown in figure 4. If the image is properly zero padded, there will not exist any interference between adjacent components

since the zero period will not result to any substantial effect on the filtered image, such that the wrapped error could be eliminated.

Part 2:

Figure 9: The original lake image f

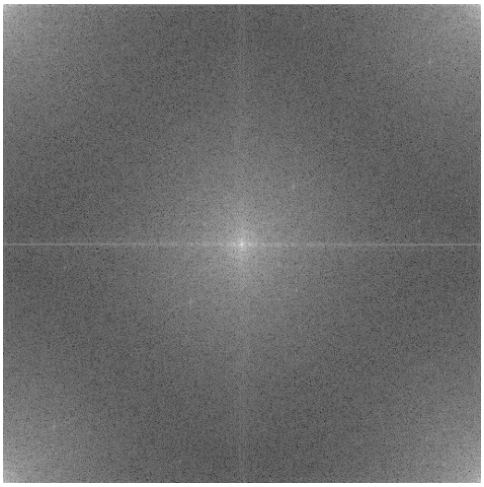

Figure 10: The DFT of f

Figure 11: The noised image c

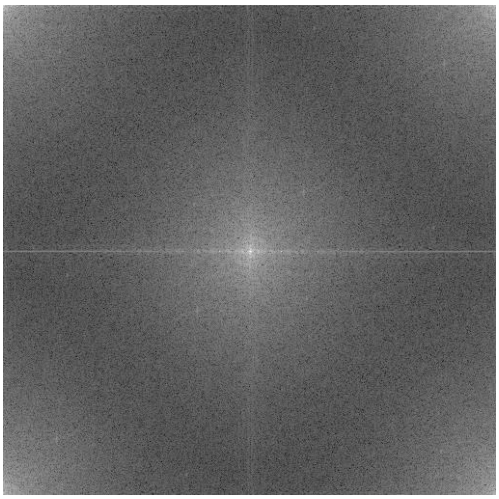

Figure 12: the DFT of c

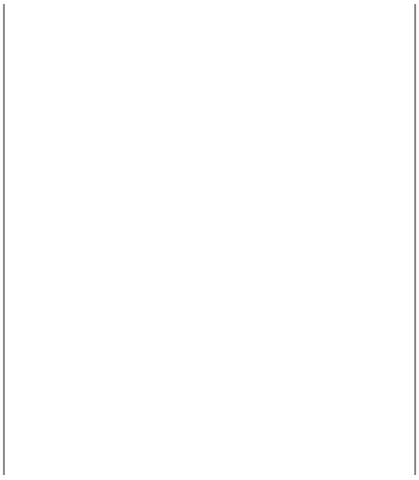

Figure 13: the image of filter H(u,v)

Figure 14: The image of g

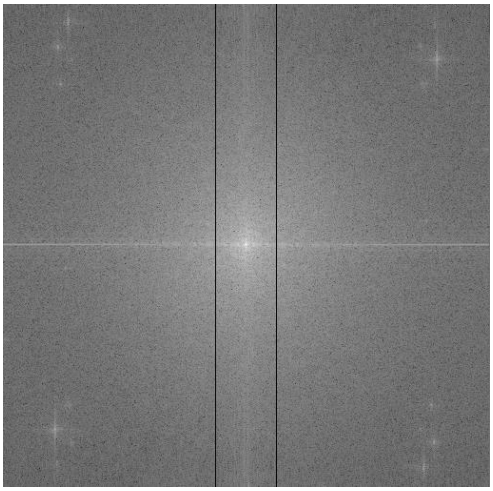

Figure 15: The image of G

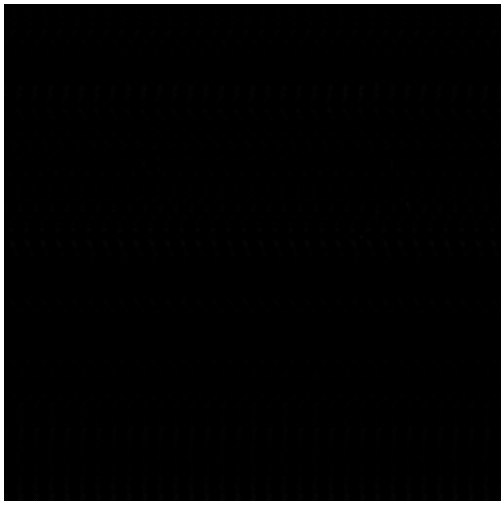

Figure 16: The image of f(x,y) - g(x,y)

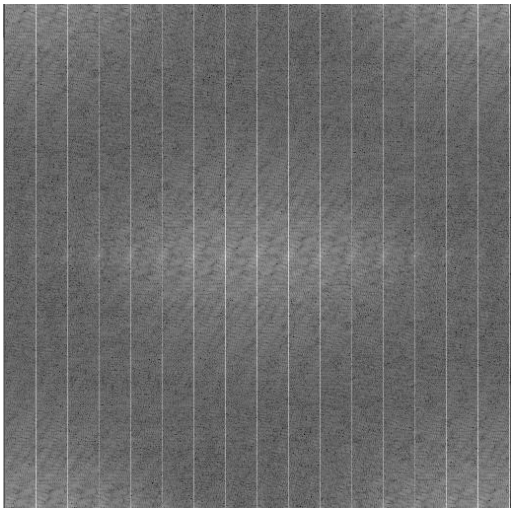

Figure 17: The DTF of f(x,y) - g(x,y)

c):
NO,
As the noise is out of the image, the frequency components with the same frequency which the noise has are also eliminated, as a result, the information of the original image is lost and cannot be completely recovered if the notch filter is applied.

Conclusion

This project shows the effects of filters as the low-pass-filter is applied in part 1 to show how it blurs the image and the notch filter is applied in part 2 to show how the notch filter is used to eliminate the noise given a specific frequency.

The frequency domain of the image is of great importance for analysis. As shown in figure 1 and figure 2, the F(0,0) is set to zero and the gray value of the image in space domain are decreased by 0.5. When the low-pass-filter is applied on the checker image, the image becomes apparently blurry since the image is reconstructed by only the information in high-frequency components with frequency higher than cut-off frequency. However, the wrapped error also appears in the image. To avoid this, the image and the filter need to be zero-padded so that the there would not exist any wrapped intervals.

The filter is a practical tool to eliminate the noise as shown in figure 11 and 14 if a notch filter is properly designed. The notch filter could eliminate the corresponding frequency components from the image and the image can be then reconstructed. It need to be emphasized that the frequency component, whose frequency is same with the noise, of the original image is also lost and cannot be completely recovered if the notch filter is applied.

Reference

[1] https://en.wikipedia.org/wiki/Gaussian_blur