HW3: Filtering in the Frequency Domain

DIP Teaching Stuff, *Sun Yat-sen University*

Welcome to your third DIP homework! The crazy Fourier Transform is coming! Are you ready for the challenge? HW3 may be much more difficult than the first two homeworks, but we believe that you can overcome all obstacles. Please submit a report (in **PDF** format) and all relevant codes as the homework solution. Again: Plagiarism = Fail, and there may be at least 30% penalty for late homework.

1 Exercises

Please answer the following questions in the report.

1.1 Rotation (10 Points)

Fig. 1(b) was generated by:

1. Multiplying Fig. 1(a) by $(-1)^{x+y}$;
2. Computing the discrete Fourier transform;
3. Taking the complex conjugate of the transform;
4. Computing the inverse discrete Fourier transform;
5. Multiplying the real part of the result by $(-1)^{x+y}$.

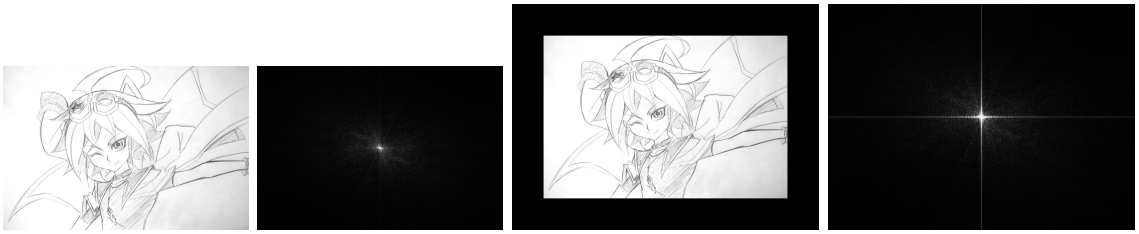
Explain (mathematically) why Fig. 1(b) appears as it does.

(a) (b)

Figure 1: *Rotation*

1.2 Fourier Spectrum (10 Points)

The (centered) Fourier spectrum in Fig. 2(b) corresponds to the original image Fig. 2(a), and the (centered) Fourier spectrum in Fig. 2(d) was obtained after Fig. 2(a) was padded with zeros (shown in Fig. 2(c)). Explain the significant increase in signal strength along the vertical and horizontal axes of Fig. 2(d) compared with Fig. 2(b).



(a) (b) (c) (d)

Figure 2: *Fourier Spectrum*

1.3 Lowpass and Highpass ($5 * 2 = 10$ Points)

1. Find the equivalent filter $H(u, v)$ in the frequency domain for the following spatial filter:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

2. Is $H(u, v)$ a low-pass filter or a high-pass filter? Prove it mathematically.

1.4 Padding (10 Points)

Section 4.7.3 says that, when filtering in the frequency domain, image padding is needed. Considering the following two types of padding:

1. Append zeros to the ends of rows and columns of the image;
2. Surround the image by a border of zeros.

Suppose the total numbers of zeros used in the above two types are the same. Do you think these two types of padding would make a difference for filtering in the frequency domain? Why?

2 Programming Tasks

Write programs to finish the following three tasks, and answer questions in your report. Don't forget to submit all relevant codes.

2.1 Pre-requirement

Input Please download the archive "hw3.zip" (actually same as "hw2.zip"), unzip it and choose the image corresponding to the last two digits of your student ID. This image is the initial input of your programming tasks in HW3. For example, if your student ID is "14110349", then you should take "49.png" as your input. You can convert the image format (to BMP, JPEG, ...) via Photoshop if necessary.

Make sure that you have selected the correct image. Misusing images may result in zero scores.

Language Any language is allowed.

Others There remain some issues that you should pay attention to:

1. You can use third-party packages for operating images. But you should manually implement your programming tasks. For example, though you can use "imread" of Matlab to load an image, you cannot invoke "fft2" of Matlab for Fourier transform.
2. Good UX (User Experience) is encouraged, but will only bring you negligible bonuses. Please don't spend too much time on it, since this is not an HCI course.
3. Keep your codes clean and well-documented. Bad coding styles will result in 20% penalty at most.

2.2 Fourier Transform (30 Points)

Write a function to perform 2-D Discrete Fourier Transform (DFT) or Inverse Discrete Fourier Transform (IDFT). The function prototype is "**dft2d(input_img, flags) → output_img**", returning the DFT / IDFT result of the given input. "flags" is a parameter to specify whether DFT or IDFT is required. You can modify the prototype if necessary.

For the report, please load your input image and use your program to:

1. Perform DFT and manually paste the (centered) Fourier spectrum on your report. (10 Points)
2. Perform IDFT on the result of the last question, and paste the real part on your report. Note: the real part should be very similar to your input image. (Why? Think about it.) (5 Points)
3. Detailedly discuss how you implement DFT / IDFT, i.e., the "dft2d" function, in **less** than 2 pages. Please focus on the algorithm part. Don't widely copy/paste your codes in the report, since your codes are also submitted. (15 Points)

2.3 Bonus: Fast Fourier Transform (50 Points)

This is an **optional** task. Here you are required to manually implement the Fast Fourier Transform (FFT). The function prototype is "**fft2d(input_img, flags) → output_img**", returning the FFT / IFFT result of the given input. "flags" is a parameter to specify whether FFT or IFFT is required.

"fft2d" should produce very similar results in comparison to "dft2d". However, your implementation may be limited to images whose sizes are integer powers of 2. We recommend you to handle this problem by simply padding the given input so as to obtain a proper size.

For the report, please load your input image and use your program to:

1. Perform FFT and manually paste the (centered) Fourier spectrum on your report. (15 Points)
2. Perform IFFT on the result of the last question, and paste the real part on your report. (10 Points)
3. Why does FFT have a lower time complexity than DFT? Explain. (10 Points)
4. Detailedly discuss how you implement FFT / IFFT, i.e., the "fft2d" function, in **less** than 2 pages. (15 Points)

Note: You may observe significant acceleration compared with "dft2d". However, it is also possible that your "fft2d" is very slow if you implement it via pure Python (or Matlab, ...). Never mind.

2.4 Filtering in the Frequency Domain (30 Points)

Write a function that performs filtering **in the frequency domain**. The function prototype is "**filter2d_freq(input_img, filter)** $\rightarrow$ **output_img**", where "filter" is the given filter. Modify the prototype if necessary. According to the convolution theorem, filtering in the frequency domain requires you to apply DFT / FFT to the given image and the given filter, and then multiply them, followed by IDFT / IFFT to get the filtered result. Hence, it should be easy to implement "filter2d_freq" based on "dft2d" (or "fft2d"). Of course, you should pay attention to some details like wraparound errors (see Chapter 4.7 of the textbook).

For the report, please load your input image and use your "filter2d_freq" function to:

1. Smooth your input image with an 7×7 averaging filter. Paste your result on the report. (8 Points)
2. Sharpen your input image with a 3×3 Laplacian filter and then paste the result (using the variant specified in Fig. 3.37(b) of the textbook). (8 Points)
3. Detailedly discuss how you implement the filtering operation, i.e., the "filter2d_freq" function, in **less** than 2 pages. (14 Points)

Note: Your results of the first two questions should be very similar to the corresponding results in HW2 (if your previous implementation for HW2 is correct).

3 Reference

0	1	0	1	1	1
1	−4	1	1	−8	1
0	1	0	1	1	1
0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

a b
c d

FIGURE 3.37
(a) Filter mask used to implement Eq. (3.6-6). (b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.