DIP homework3

16327109 谢昆成

1 Exercises

1.1

1.1 Rotation (10 Points) Figure. 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.

Proof:

Computing the discrete Fourier transform

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

taking the complex conjugate of the transform, applying the conjugate symmetry

$$F^*(u-M/2,v-N/2)=F(-u+M/2,-v+N/2)$$

computing the IDFT

$$\begin{aligned}&F(-u+M/2,-v+N/2)\\=&F((-u+M)-M/2,(-v+N)-N/2)\\=&F(-u-M/2,-v-N/2)\\\Leftrightarrow&f(-x+M,-y+N)(-1)^{x+y}\end{aligned}$$

multiplying the real part of the result by $(-1)^{x+y}$, then we get the reconstructed image $\hat{f}$

$$\begin{aligned}&\hat{f}(x,y)\\=&f(-x+M,-y+N)(-1)^{x+y}(-1)^{x+y}\\=&f(-x+M,-y+N)\end{aligned}$$

so, the reconstructed image is center-symmetric about(M/2,N/2)

1.2

For being padded with zeros, the original image add a black edging, which is different from the original white edging. Thus the image change more intensively and brings the significant increase in signal strength along the vertical and horizontal axes of image.

1.3

(1)

$$g(x,y) = f(x,y-1) + f(x-1,y) - 4f(x,y) + f(x+1,y) + f(x,y+1)$$

$$\begin{aligned} G(u,v) =& F(u,v)(e^{-j2\pi v/N} + e^{-j2\pi u/M} + e^{j2\pi u/M} + e^{j2\pi v/N} - 4) \\ =& F(u,v)(cos(2\pi v/N) - jsin(2\pi v/N) + cos(2\pi u/M) - jsin(2\pi u/M) \\ & + cos(2\pi u/M) + jsin(2\pi u/M) + cos(2\pi v/N) + jsin(2\pi v/N) - 4) \\ =& 2F(u,v)(cos(2\pi v/N) + cos(2\pi u/M) - 2) \end{aligned}$$

$$\therefore H(u,v) = 2(cos(2\pi v/N) + cos(2\pi u/M) - 2)$$

(2)

when (u,v)->(0,0), H(u,v)->0; when (u,v) is far from (0,0), |H(u,v)| gets greater.

So H(u,v) is a high-pass filter.

2 Programming Tasks

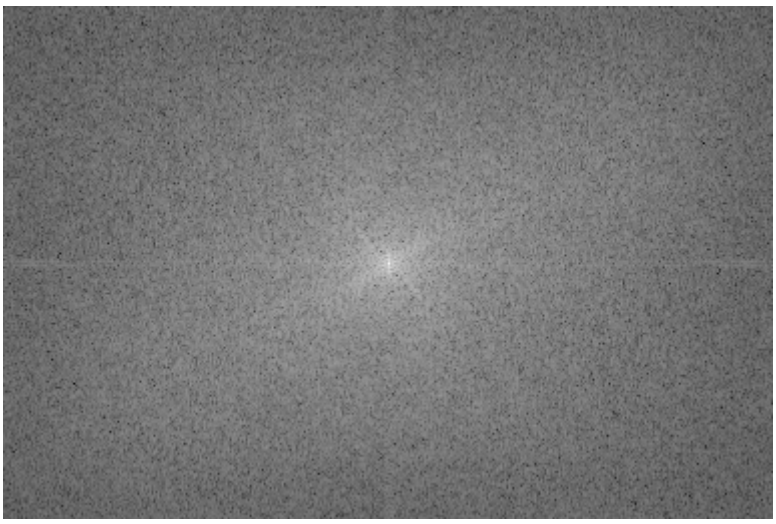
My student ID is 16327109, so I choose "09.png" as my input.

2.1

1. Perform DFT and manually paste the (centered) Fourier spectrum on your report. (10 Points)

   original image

   DFT (centered and log transform)



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.) (10 Points)

   IDFT

For Fourier transform is reversible, we can get original image only containing real part. For DFT, we can get approximately original image which is very similar to input image.

3. Detailed discuss how you implement DFT / IDFT 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.

To reduce time complexity of DFT/IDFT, we have to separate the 2-D DFT into 1-D transforms. We can write as

$$F(u,v) = \sum_{x=0}^{M-1} e^{-j2\pi ux/M} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi vy/N} = \sum_{x=0}^{M-1} F(x,v) e^{-j2\pi ux/M}$$

By doing this, we reduce the time complexity of $O(M^2N^2)$ to $O(MN(M+N))$.

We first apply 1-D DFT/IDFT of a row of f(x,y). By varying x from 0 to M-1, we compute a set of 1-D DFTs for all rows of f(x,y). Then we apply 1-D transforms of the columns of F(x,v).

DFT and IDFT are only different on the sign of $e^{j2\pi x/M}$, so we can only change the sign of exponent to do DFT or IDFT.

run method

```
python dft.py inputFilename outputFilename mode
```

0 for DFT, 1 for IDFT

for instance

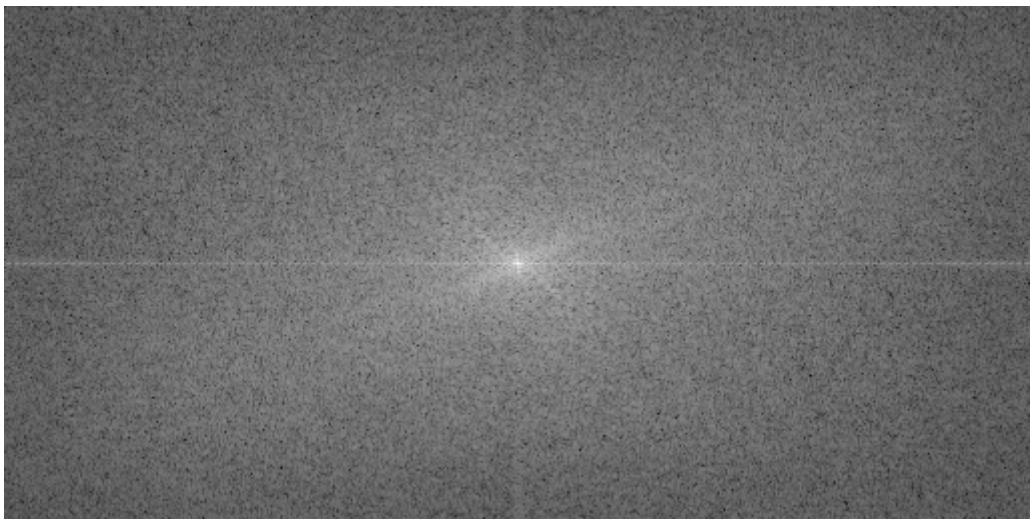
```
python dft.py ../pic/09.png ../pic/09dft.png 0
```

2.2

1.

zero padding

FFT (centered and log transform)



2.

IFFT

3.

FFT exploits the symmetry of DFT

$$F(u) = F_{even}(u) + F_{odd}(u)W_{2K}^{u}$$

$$F(u+K) = F_{even}(u) - F_{odd}(u)W_{2K}^{u}$$

An M-point transform can be computed by dividing the original expression into two parts. Computing the first half of F(u) requires evaluation of the two (M/2)-point transforms of F_{even} and F_{odd}. The resulting values of $F_{even}(u)$ and $F_{odd}(u)$ are then to obtain F(u) for u =0,1,2, ..., (M/2-1). The other half then follows directly from $F(u+K)$ without additional transform evaluation.

4.

We first apply 1-D FFT/IFFT of a row of f(x,y). By varying x from 0 to M-1, we compute a set of 1-D FFTs for all rows of f(x,y). Then we apply 1-D transforms of the columns of F(x,v).

FFT and IFFT are only different on the sign of $e^{j2\pi x/M}$, so we can change the sign of exponent to do FFT or IFFT.

run method

```
python fft.py inputFilename outputFilename mode
```

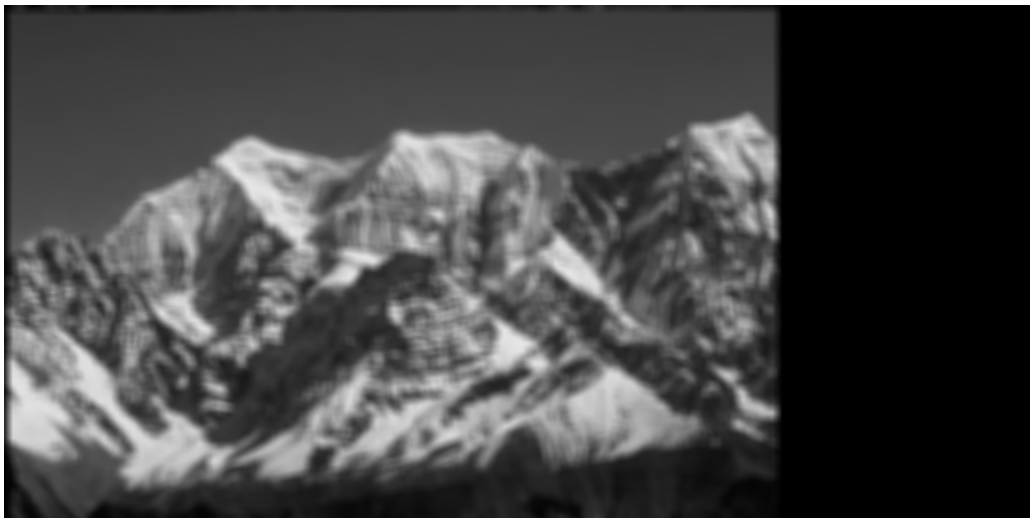
0 for DFT, 1 for IDFT

for instance

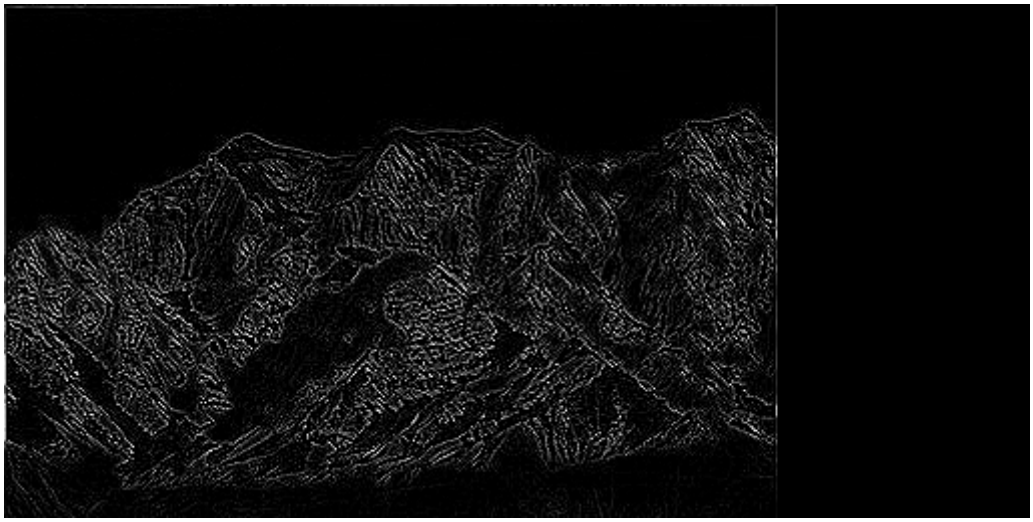
```
python fft.py ../pic/09.png ../pic/09fft.png 0
```

2.3

1.



2.



3.

According to the convolution theorem, filtering in the frequency domain is equivalent to apply FT to the given image and filter, and then multiply them, followed by IDFT / IFFT to get the filtered result.

So I first use zero-padding to pad the filter and image to the same size, then use FFT to get Flourier transform of each. Multiply them and use IFFT to get the filtered result at last.

run method

```
python filter.py inputFilename outputFilename mode
```

0 for smooth, 1 for sharpen

for instance

```
python filter.py ../pic/09.png ../pic/09smooth.png 0
```