Homework 9
DFT & IDFT

Student: Yuanyuan Zhao, 2300013077, School of EECS, zhaoyuanyuan@stu.pku.edu.cn
Lecturer: Yisong Chen

Part 1: DFT & IDFT Algorithm Design

We implement our customized DFT and IDFT algorithm in myDFT2_fast.mlx and myIDFT2_fast.mlx, and the image-processing script is DFT_IDFT.mlx.

(a) **DFT & IDFT Process**

(i) Build the wave basis

$$\phi_{u,v}(x,y) = e^{-i2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\phi_{u,v}(x,y)\phi^*_{u',v'}(x,y) = \begin{cases} MN & \text{if } u = u' \text{and } v = v' \\ 0 & \text{otherwise} \end{cases}$$

Because of the above equation, normalization is required in subsequent IDFT process.

(ii) Calculate the coefficient

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\phi_{u,v}(x,y)$$

For all the pixel (x, y) in the image, accumulate their similarity with frequency $(u/M, v/N)$, that is, $f(x,y)\phi_{u,v}(x,y)$.

(iii) Move low frequencies to the center for visualization

```
F_shifted = fftshift(F);
```

(iv) Show the magnitude, phase, real part and imaginary part of F

$$log_{mag} = \log(1 + |F_{shifted}|)$$

```
magnitude = mat2gray(log_mag);
phase = angle(F);
real_part = real(F);
imag_part = imag(F);
```

(v) Reconstruct the image

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\phi^*(x,y)$$

For all the frequencies $(u/M, v/N)$ as a wave basis, accumulate their contribution to the image, that is, $F(u,v)\phi^*(x,y)$. In the equation, $\frac{1}{MN}$ is the normalization coefficient calculated in (i) and ϕ^* is the complex conjugate of ϕ.

(b) **Implementation 1: For-Loop Version**

To calculate the coefficient F, we need to traverse all pairs of (u, v) where $u = 0, 1, ..., M-1, v = 0, 1, ..., N-1$. And for each (u, v), the coefficient is based on all the pixels (x, y). So, the time complexity of the for-loop algorithm is $O(M^2N^2)$, which is inacceptably time consuming. Specific implementation can be found in myDFT.mlx and myIDFT.mlx.

(c) **Implementation 2: Matrix Multiplication Speedup**

To speed up the process, we calculate the vertical and horizontal DFT and IDFT kernel in advance. Specific implementation can be found in myDFT2_fast.mlx and myIDFT2_fast.mlx.

```
u = (0:M-1)';
x = 0:M-1;
Wx = exp(-1i * 2 * pi * (u * x) / M);

v = (0:N-1)';
y = 0:N-1;
Wy = exp(-1i * 2 * pi * (v * y) / N);

x = (0:M-1)';
u = 0:M-1;
iWx = exp(1i * 2 * pi * (x * u) / M);

y = (0:N-1)';
v = 0:N-1;
iWy = exp(1i * 2 * pi * (y * v) / N);
```

Then the Fourier Transform can be carried out with the weight matrix.

$$F = Wx * f * Wy$$

$$f = (iWx * F * iWy)/MN$$

Part 2: Low & High Frequency Filter

(a) **Build the Filter**

Choose appropriate Cutoff Frequency D_0 and we have

```
H_low = double(D <= D0);
H_high = double(D > D0);
```

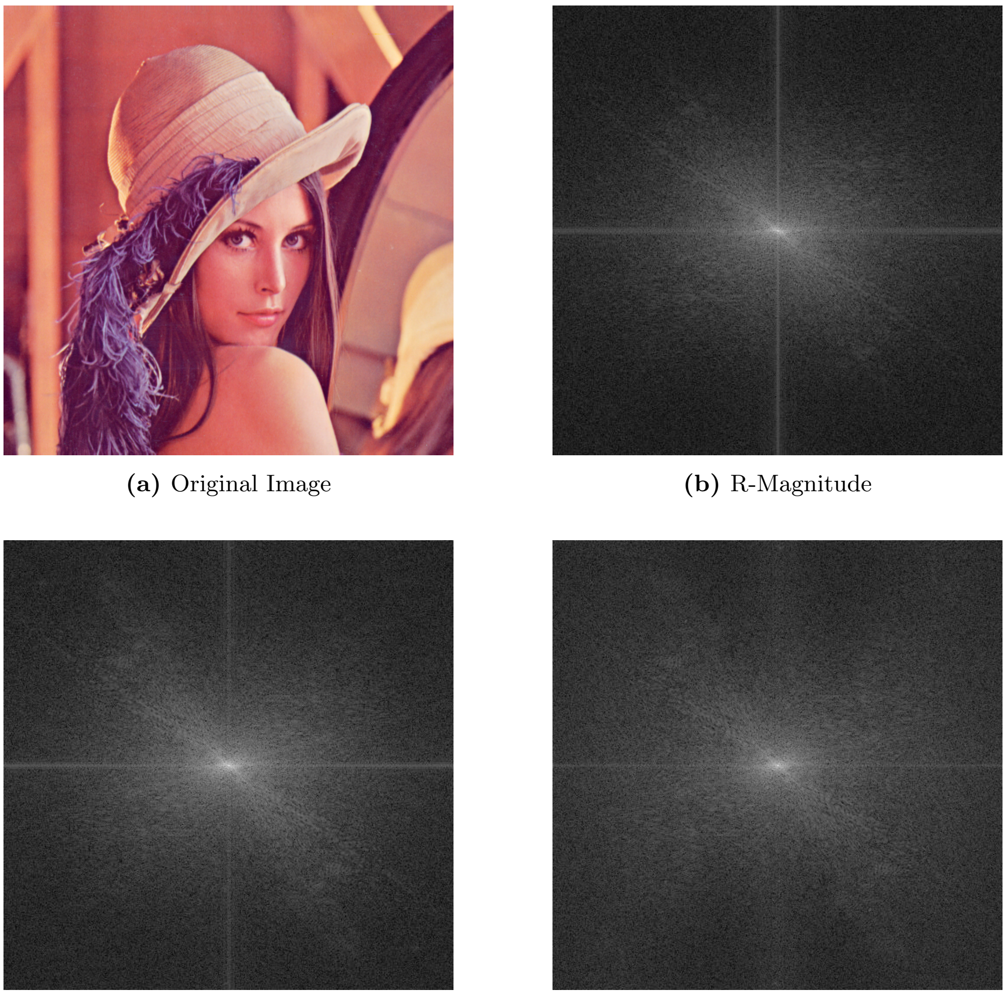
(b) **Calculate the Filtered Coefficient**

```
F_low = F_shifted .* H_low;
```

```
F_high = F_shifted .* H_high;
```

(c) **Reconstruct the Filtered Image**

```
img_low = myIDFT2_fast(F_low);
img_high = myIDFT2_fast(F_high);
```



(a) Original Image **(b)** R-Magnitude

(c) G-Magnitude **(d)** B-Magnitude

Figure 1: Lena: Magnitude in R, G, B channels

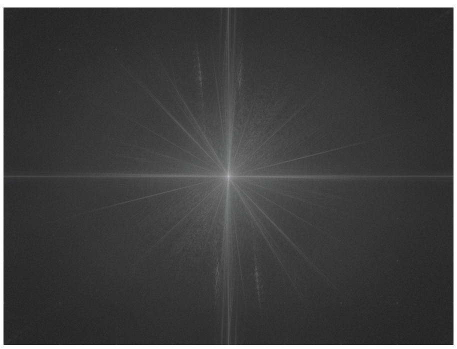
Part 3: Result Analysis

(a) **Reconstructed and Filtered Images**

Figure 2 and 3 respectively show the reconstructed image and the high/low-pass image. i) The reconstruction result is satisfactory, as no obvious distinction can be noticed,

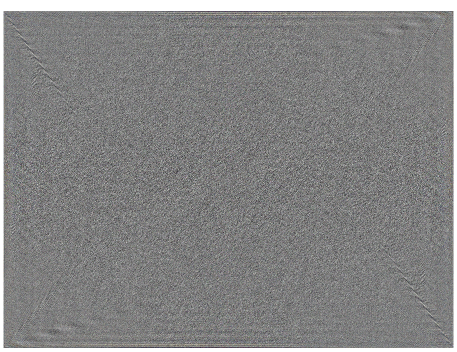
(a) Original Image

(b) Reconstructed Image



(c) Magnitude

(d) Phase

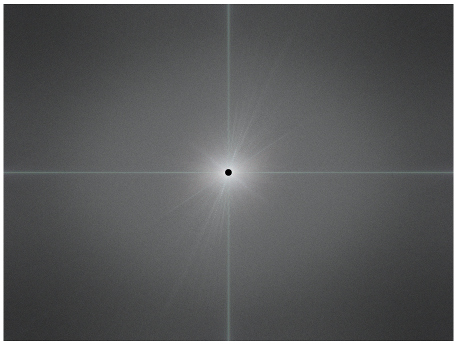


(e) Real Part

(f) Imaginary Part

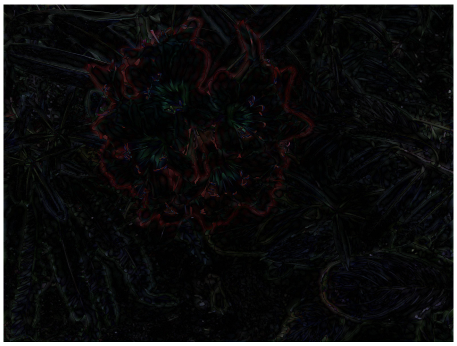
Figure 2: Room: Images generated during the DFT process

(a) Original Image



(b) High-pass Filter

(c) Low-pass Image



(d) High-pass Image

Figure 3: Flower: Comparison between the Original and Filtered Images with D0=30

Image Pairs	Flower	Lena	Room
Error	5.4324e-13	7.546e-14	6.6642e-13

Table 1: Error evaluation for each original and reconstructed image pair

showing that our DFT and IDFT implementation is correct. ii) The high-pass filter succeeds to separate high-frequency components from the low-frequency ones, with the outline of the flower in the high-pass image and the low-pass image becomes blur and reduces noises.

(b) **Reconstructed Error Evaluation** We define the metric below to show the performance of our reconstruction. And results can be found in Table 1. The tiny discrepancy arises from the limited complex calculation precision in DFT and IDFT.

$$error = \frac{\sum_{(x,y)} |img_{ori} - img_{recon}|}{MN}$$

Part 4: Appendix: Detailed Experiment Results

(a) **Reconstructed Image, Magnitude, Phase, Real Part, Imaginary Part**

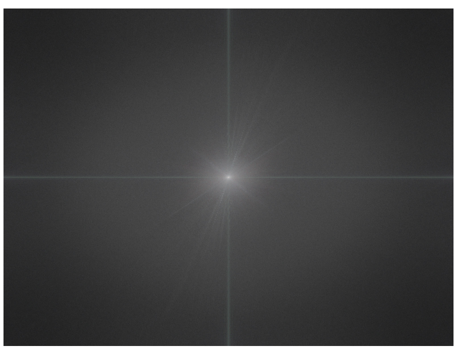
Figure 2, 4, 5

(b) **High-Pass Filter, High-Pass Image, Low-Pass Image**

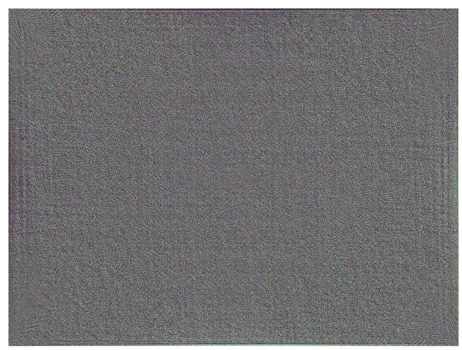
Figure 3, 6, 7

(a) Original Image

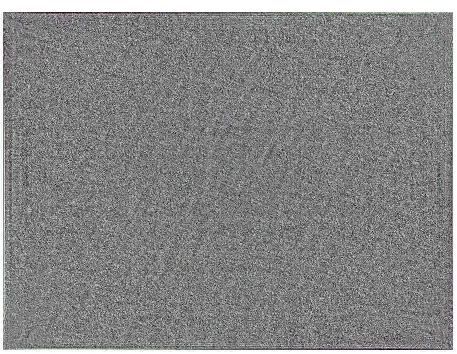
(b) Reconstructed Image



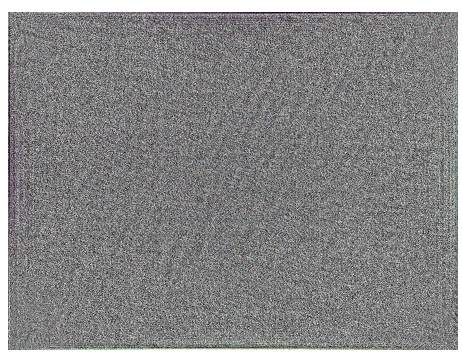
(c) Magnitude



(d) Phase



(e) Real Part

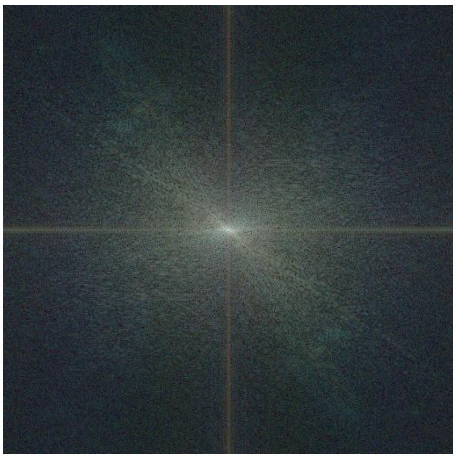


(f) Imaginary Part

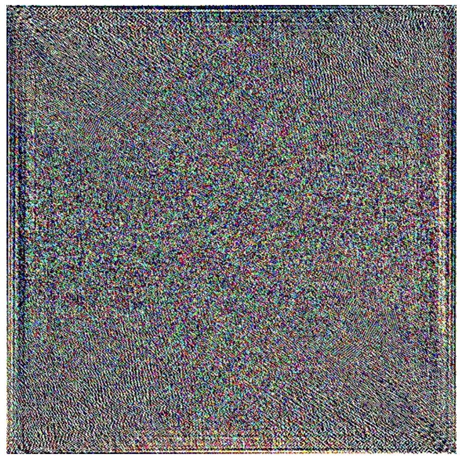
Figure 4: Flower: Images generated during the DFT process

(a) Original Image

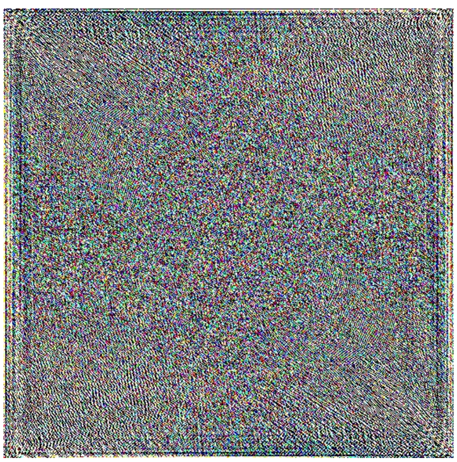
(b) Reconstructed Image



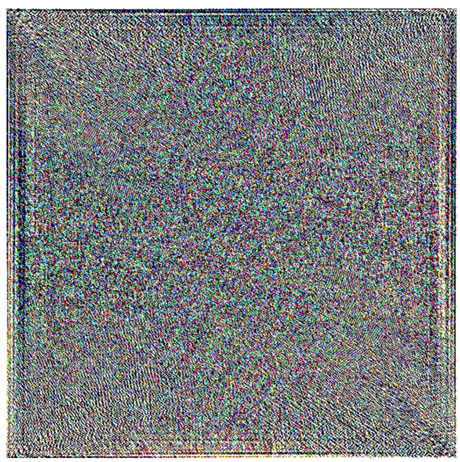
(c) Magnitude



(d) Phase



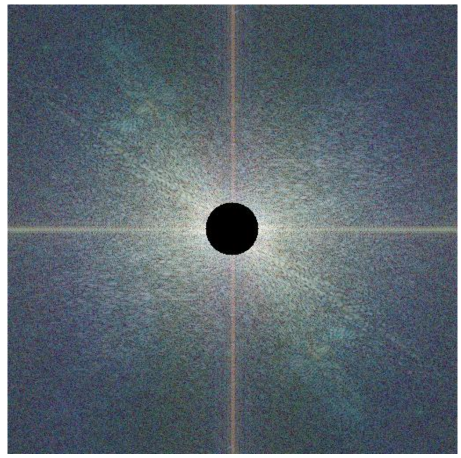
(e) Real Part



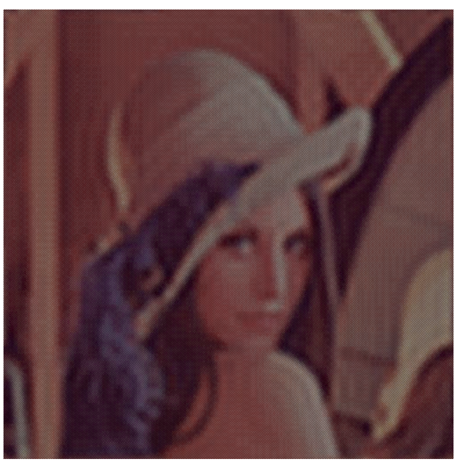
(f) Imaginary Part

Figure 5: Lena: Images generated during the DFT process

(a) Original Image



(b) High-pass Filter

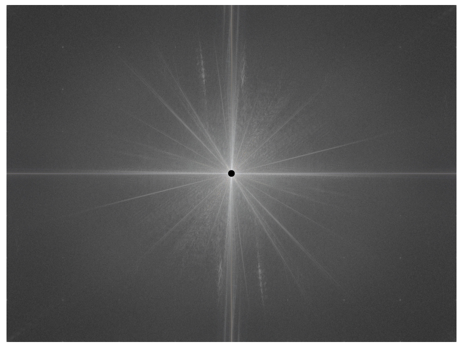


(c) Low-pass Image

(d) High-pass Image

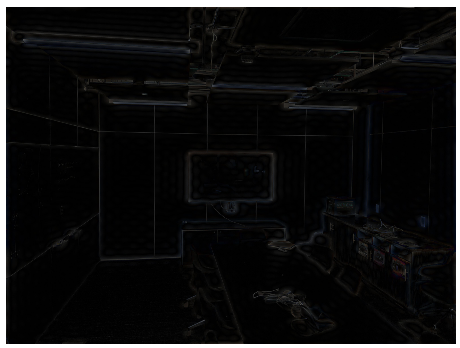
Figure 6: Lena: Comparison between the Original and Filtered Images with D0=30

(a) Original Image



(b) High-pass Filter

(c) Low-pass Image



(d) High-pass Image

Figure 7: Room: Comparison between the Original and Filtered Images with D0=30