Frequency domain

رضا عباس‌زاده

اطلاعات گزارش

تاریخ: ۱۴۰۰/۲/۱۷

واژگان کلیدی:
Frequency domain
Filtering
DFT
Phase
Magnitude
Smoothing
Edge detection

چکیده

حوزه فرکانس یکی از حوزه‌های پردازش تصویر است. در این تمرین با استفاده از تبدیل فوریه تصاویر را به حوزه فرکانس می‌بریم، فیلتر های مختلف را روی آن اعمال می‌کنیم و نتایج را بررسی و مقایسه می‌کنیم و درنهایت مجدد تصویر به حوزه مکان (spatial) بازمیگردانیم.

۱-مقدمه

نوشتار حاضر، مفهوم حوزه فرکانس در تصاویر را بررسی می‌کند و با استفاده از تبدیل فوریه تصاویر را به این حوزه می‌برد. سپس با عملیات مانند فیلتر کردن تصاویر در این حوزه آشنا می‌شویم و نتایج را با حوزه مکان مقایسه می‌کند.

۲-شرح تکنیکال

۱-۱-۴ تبدیل فوریه

تبدیل فوریه ابزاری مهم در زمینه پردازش تصویر است. از این تبدیل برای تجزیه تصاویر به عناصر سینوسی و کسینوسی استفاده میشود. این تبدیل تصویر را به حوزه فرکانسی میبرد.

تبدیل فوریه در تصاویر دیجیتال انواع و پیکربندی های مختلفی دارد که از بین آنها دو نوع DFT وFFT کاربرد بیشتری دارند. تبدیل های فوریه گسسته یا DFT نظیر به نظیر می باشند و حجم بالایی از محاسبات را شامل می شوند. اما به منظور کاهش محاسبات و ایجاد یک الگوریتم بهینه تر شما می تاونید از تبدیل های فوریه سریع یا FFT استفاده کنید.

یکی از کاربردهای اصلی این تبدیل، فیلترنیگ تصاویر می‌باشد. همانطور که در تمارین گذشته بررسی شده، عمل فیلترینگ در حوزه مکان با استفاده از کانولوشن انجام می‌گرفت. در حوزه فرکانس با عمل ضرب عنصر به عنصر تصویر در حوزه فرکانس و فیلتر منتقل شده به حوزه فرکانس قابل انجام است.

برای انتقال تصویر f به حوزه فرکانس (F) از رابطه زیر استفاده می‌کنیم:

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

بعد از انجام عملیات فیلترینگ روی تصویر، با استفاده از رابطه زیر می‌توان تصویر را به حوزه مکان بازگرداند:

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

در این تمرین برای انجام تبدیل فوریه از کتابخانه numpy استفاده شده است. با استفاده از تابع fft2 تصویر را به حوزه فرکانس می‌بریم و با استفاده از تابع ifft2 تصویر را به حوزه مکان می‌بریم.

نتایج توابع فوق اعداد complex هستند که دارای بخش حقیقی R و مجازی I هستند. برای محاسبه طیف و فاز از روابط زیر استفاده می‌کنیم:

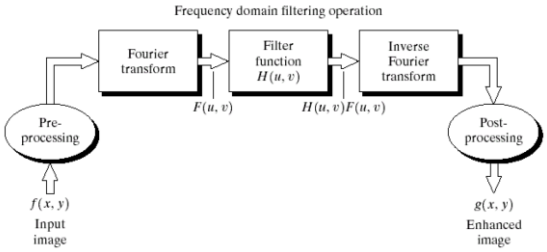
$$\text{Amplitude: } A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

$$\text{Phase: } \phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

در پردازش تصاویر فقط با طیف در حوزه فرکانس کار می‌کنیم و تغییرات را روی آن اعمال ی‌کنیم، چراکه هرگونه تغییر روی فاز تصویر باعث از بین رفتن اطلاعات تصویر می‌شود.

فیلترینگ

به طور کلی مراحل فیلترینگ در حوزه فرکانس به این ترتیب است:



۱. بدست آوردن P و Q با توجه به اندازه MxN تصویر f . عموماً P= 2 M و Q= 2 N انتخاب می شود.

۲. ساختن تصویر f_p با عمل افزودن صفر به تصویر f. همانطور که می‌دانید در حوزه فرکانس سیگنال‌ها را متناوب درنظر می‌گیریم. بنابراین با افزودن padding به تصویر نرخ نمونه برداری را افزایش داده‌ایم و درنتیجه سیگنال‌های متوالی رو هم اثر نخواهند داشت و نتیجه مطلوب‌تری داریم.

۳. ضرب کردن f_p در $(-1)^{x+y}$ برای انتقال مبدا به مرکز

۴. بدست آوردن DFT حاصل با استفاده از تابع fft2

نکته: می‌توان به جای انجام مرحله شماره ۳، مبدا نتیجه مرحله ۴ را با استفاده از تابع fftshift به مرکز تصویر منتقل کرد.

۵. اعمال فیلتر حقیقی با مبدا مرکز (P/ 2 و Q/ 2) با ضرب آرایه‌ای. توجه کنید که اگر فیلتر مورد نظر از حوزه مکان بود، ابتدا باید اطراف آن zero padding اضافه کنیم تا ابعاد آن با تصویر برابر شود و سپس آن را با استفاده از تبدیل فوریه به حوزه فرکانس منتقل کنیم و سپس مبدا آن را به مرکز تصویر منتقل کنیم.

۶. بدست آوردن IDFT حاصل. نتیجه تابع ifft2 ممکن است شامل بخش مجازی باشد. در این صورت تنها بخش حقیق آن را انتخاب می‌کنیم.

۷. ضرب کردن نتیجه در $(-1)^{x+y}$ برای بدست آوردن g_p.

نکته: مانند مرحله ۳ و ۴ در اینجا نیز می‌توان به جای انجام مرحله ۷، قبل از انتقال به حوزه مکان از تابع ifftshift استفاده کرد و مبدا فرکانس را به نقطه (0,0) منتقل کرد.

۸. برداشتن بخشی به اندازه MxN از گوشه ی بالای سمت چپ g_p به منظور حذف padding ایجاد شده در مرحله شماره ۲.

فیلتر جداپذیر:

اگر فیلتر را به عنوان یک ماتریس درنظر بگیریم، فیلتری که رتبه آن یک باشد یک فیلتر جداپذیر است. به عبارت دیگر می‌توان آن را به صورت حاصلضرب یک بردار ستونی در یک بردار سطری نوشت.

۴-۱-۲

در این تمرین ابتدا هر تصویر را با استفاده از تابع fft2 به حوزه فرکانس می‌بریم. سپس طیف آن را جدا می‌کنیم و نمایش می‌دهیم. این طیف معمولا یک تصویر سیاه است.

چرا که مقدار فرکانس در مبدا بسیار زیاد است و باعث می‌شود بقیه نقاط در مقایسه با آن مقدار ناچیزی داشته باشند و سیاه نمایش داده شوند. برای رفع این مشکل لگاریتم تصویر را محاسبه می‌کنیم. البته به دلیل وجود مقدار صفر در تصویر ابتدا یک عدد به همه مقادیر تصویر اضافه می‌کنیم. سپس نتیجه را به رنج ۰ تا ۲۵۵ نرمال‌سازی می‌کنیم.

در نهایت با استفاده از تابع fftshift مبدا فرکانسی را به مرکز تصویر شیفا می‌دهیم تا دید بهتری داشته باشیم.

در این تصویر مرکز تصویر شامل فرکانس‌های پایین می‌باشد و هرچه از مرکز فاصله بگیریم فرکانس‌های بالاتر را مشاهده می‌کنیم.

۴-۲-۱ فیلترها

فیلتر پایین‌گذر

این فیلترها، فرکانس‌های پایین را تغییر نمی‌دهند اما فرکانس‌های بالا را تضعیف یا حذف می‌کنند. فرکانس‌های بالا معمولا شامل اطلاعات ریز تصویر از جمله لبه‌ها هستند که با حذف آن‌ها شاهد تصویری smooth تر با لبه‌های آرام‌تر خواهیم بود.

فیلتر بالاگذر

این فیلترها برعکس فیلترهای پایین‌گذر، فرکانس‌های پایین را تضعیف و حذف می‌کنند. با حذف فرکانس‌های پایین، اطلاعات کلی تصویر از بین می‌رود و لبه‌ها باقی می‌مانند. بنابراین از این فیلترها برای شناسایی لبه‌ها استفاده می‌شود.

۴-۲-۲

این بخش مشابه با بخش اول مربوط به فیلترها است با این تفاوت‌ها:

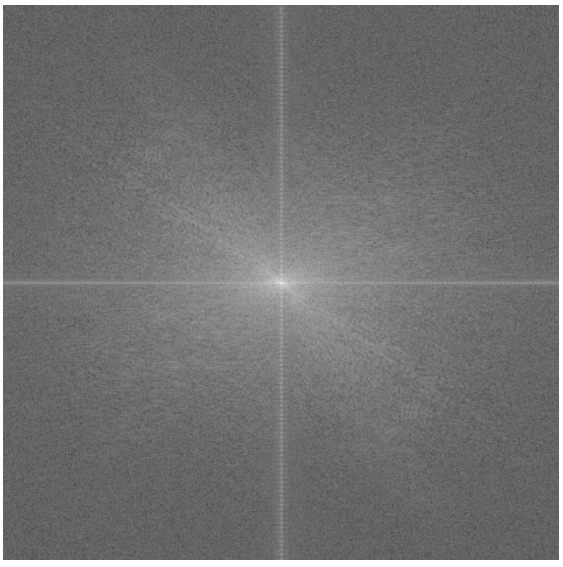
۱. در این قسمت مبدا فرکانسی را به مرکز تصویر شیفت نمی‌دهیم. چرا مبدا فرکانسی فیلترها هم در گوشه ها قرار دارد و مرکز فیلتر نیست.

۲. فیلترها از ابتدا در حوزه فرکانس هستند و نیاز به تبدیل نیست.

۳-شرح نتایج و نتیجه گیری

۴-۱-۱

تصویر اصلی



تصویر اصلی در حوزه فرکانس

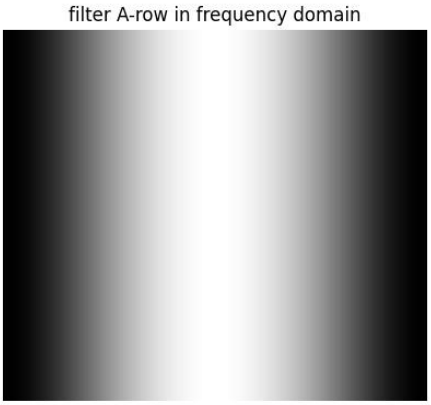
فیلتر a

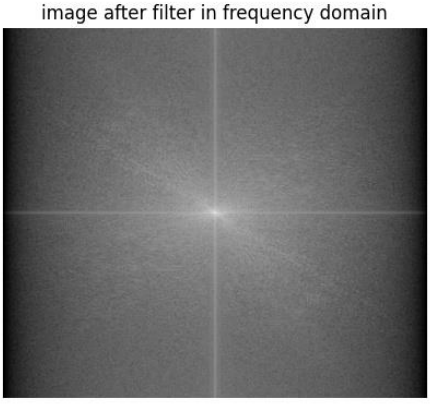
این فیلتر یک فیلتر جداپذیر است زیرا می‌توان آن را به صورت زیر نوشت:

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} = \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix} * \frac{1}{4}\begin{bmatrix}1\\ 2\\ 1\end{bmatrix}$$

بتدا هر یک از دو فیلتر یک بعدی را بررسی می‌کنیم و سپس فیلتر a را بررسی می‌کنیم.

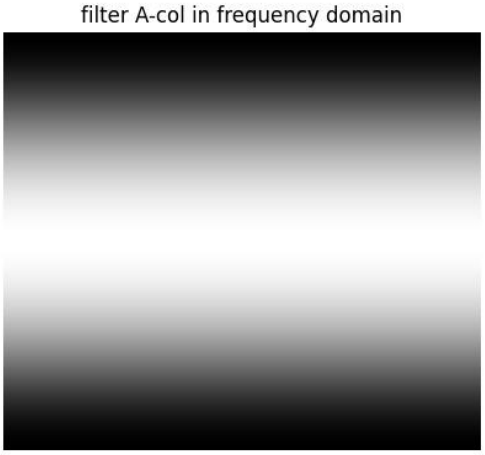
فیلتر $\frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix}$

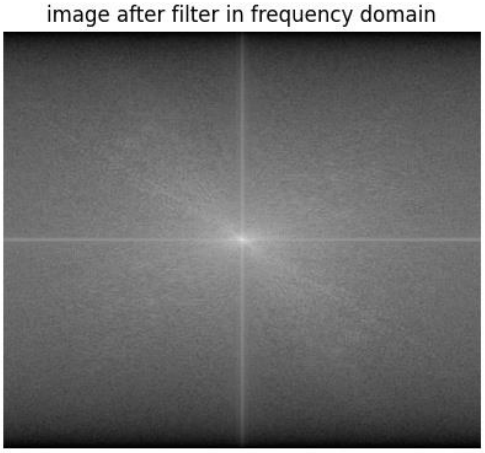




این فیلتر لبه‌های افقی را smoothتر کرده است. چرا که در حوضه فرکانس جزییات حذف شده (لبه‌های سیاه) به صورت عمودی هستند و می‌دانیم که تغییرات در حوضه فرکانس و مکان عمود برهم هستند.

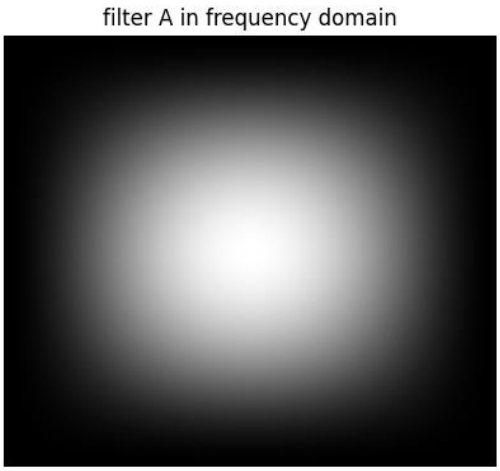
فیلتر $\frac{1}{4}\begin{bmatrix}1\\ 2\\ 1\end{bmatrix}$:

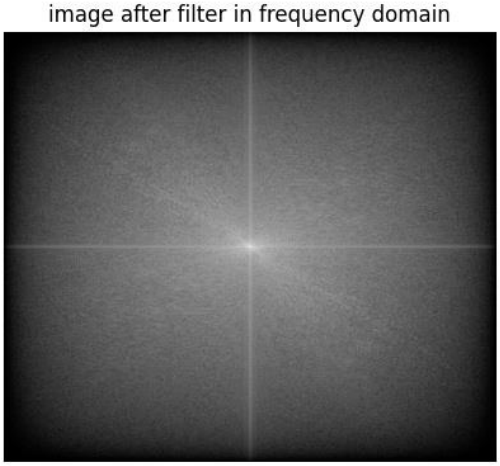




این فیلتر برخلاف فیلتر قبل، لبه‌های عمودی را smooth کرده است.

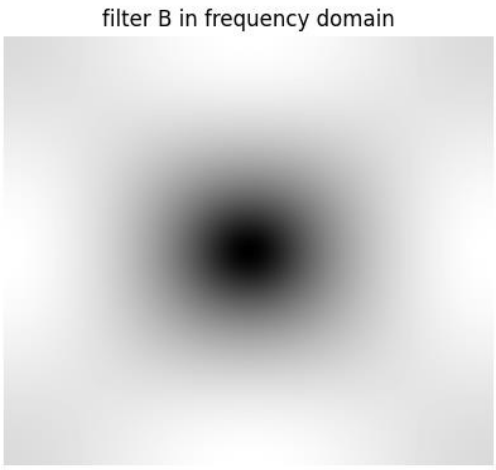
فیلتر A:

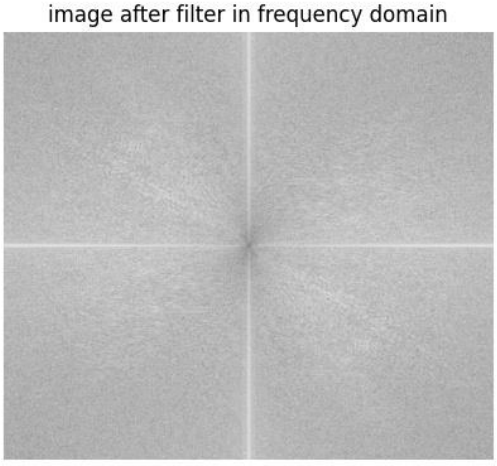


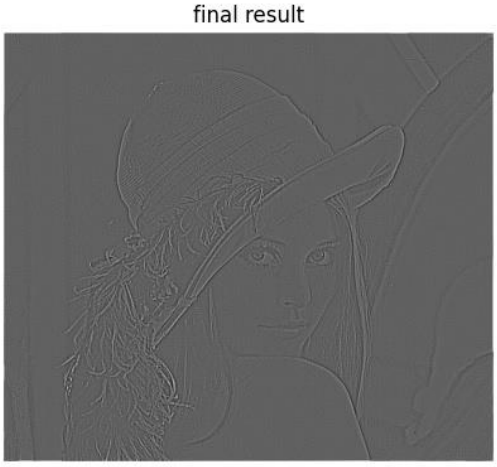


این فیلتر ترکیب دو فیلتر قبل می‌باشد و همانطور که از تصویر این فیلتر در حوزه فرکانس مشخص است، این فیلتر یک فیلتر پایین‌گذر می‌باشد و باعث تضعیف فرکانس‌های بالا و لبه‌ها شده است.

فیلتر B:

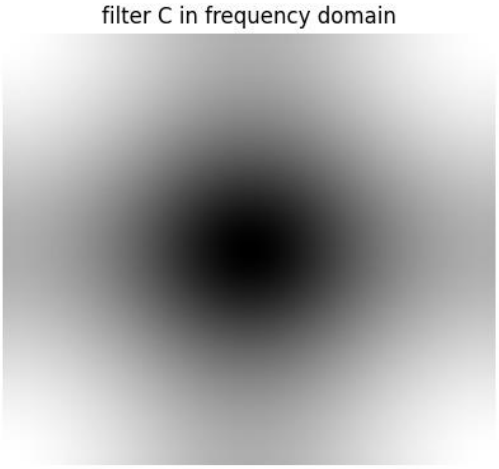


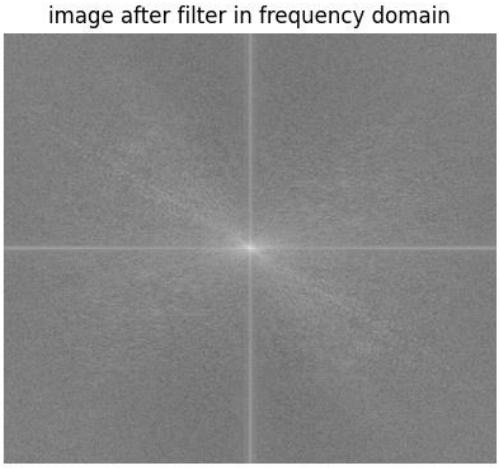




همانطور که در اولین تصویر مشخص است، این فیلتر یک فیلتر بالاگذر می‌باشد و فرکانس‌های پایین شامل کلیات تصویر را از بین برده است و فقط لبه‌ها باقی مانده‌اند. به عبارت دیگر، این فیلتر یک فیلتر شناسایی کننده لبه ها است.

فیلتر C:

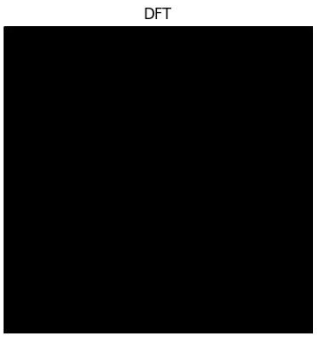


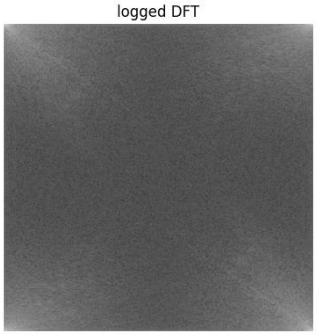


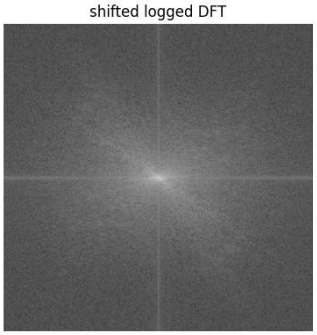
این فیلتر باعث تقویت لبه‌ها در تصویر شده است و درواقع برای edge enhancement به کار می‌رود.

۴-۱-۲

تصویر Lena:

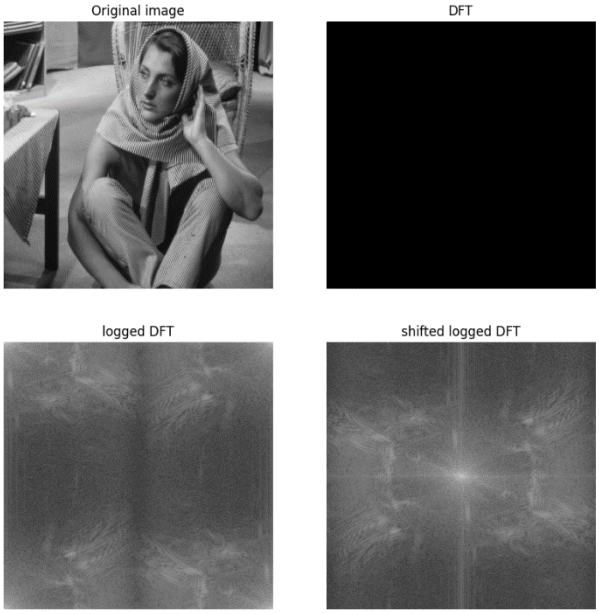






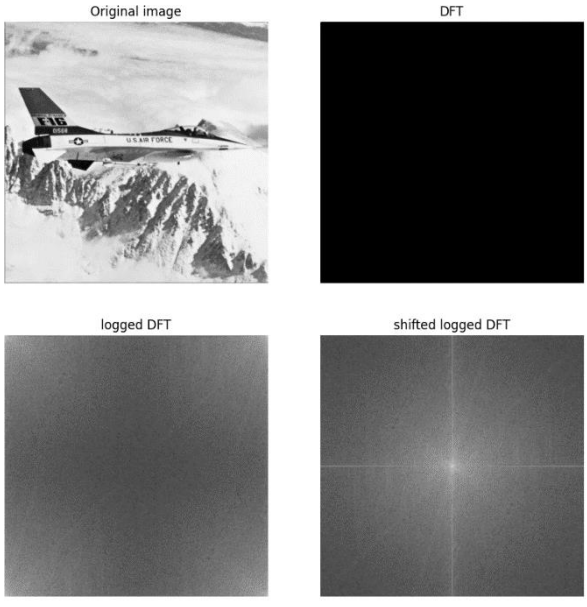
در تصویر نهایی در حوزه فرکانس، دوخط عمودی و افقی در وسط تصویر مشاهده می‌شود که نشان‌دهنده وجود لبه-هایی عمود بر این دو در حوزه مکان این تصویر است.
همچنین می‌بینیم که بیشتر فرکانس‌های موجود، فرکانس-های پایین و نزدیک به مرکز هستند و تصویر دارای جزییات زیادی نمی‌باشد (فرکانس‌های بالا)

تصویر Barbara:



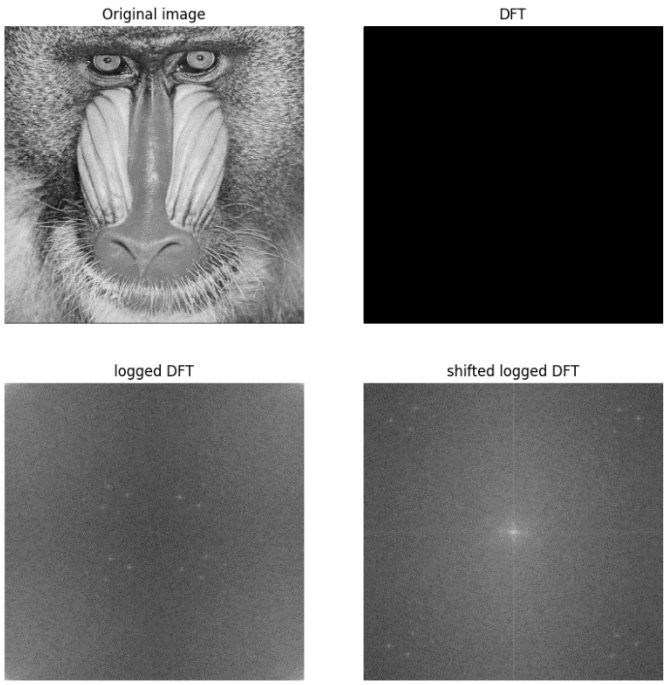
همانطور که مشاهده می‌کنید فرکانس‌های بالا به شکل تقریبا نامنظمی در این تصویر وجود دارند که نشان دهنده لبه‌های زیاد در تصویر به شکل نامنظم است.

تصویر F16:



می‌بینیم که در این تصویر فرکانس‌ها پخش‌شوندگی بالاتری نسبت به تصاویر قبل دارند. همچنین دو خط عمودی و افقی داریم که به ترتیب نشان دهنده وجود لبه-های افقی مانند بدنه هواپیما و لبه های عمودی مانند کوه است.

تصویر baboon:



در هر یک از چهار گوشه تصویر ۴ نقطه مشاهده می‌کنیم که نشان‌دهنده وجود یک الگوی منظم در فرکانس‌های بالا و جزییات تصویر می‌باشد که احتمالا مربوط به جزییات مربوط به صورت این بابون می‌باشد.

۴-۲-۱

از آنجایی که اندازه تصویر ۲۵۶*۲۵۶ می‌باشد، باید به تصویر zero padding اضافه کنیم تا ابعاد آن دو برابر یعنی ۵۱۲*۵۱۲ شود. همچنین باید اطراف فیلتر پدینگ اضافه کنیم تا ابعاد آن با ابعاد تصویر ۵۱۲*۵۱۲ برابر شود.

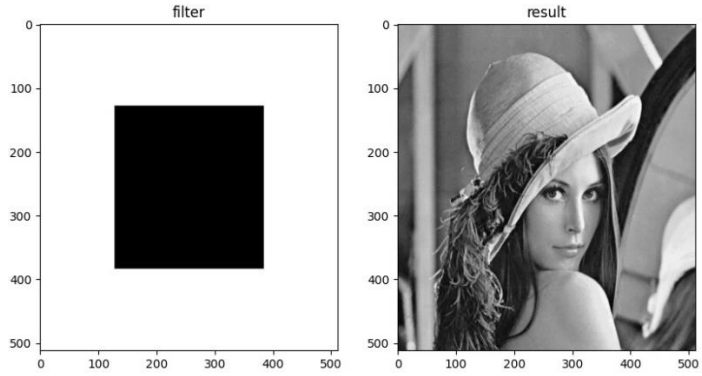
برای بخش دوم سوال، تمام بازه‌ای که شامل عکس اصلی است و شامل پدینگ اضافه شده نیست، در هر دو روش مقدار برابری خواهند داشت. یعنی به ازای مقادیر زیر:

$$0 \leq m \leq 255 \quad \text{و} \quad 0 \leq n \leq 255$$

۴-۲-۲

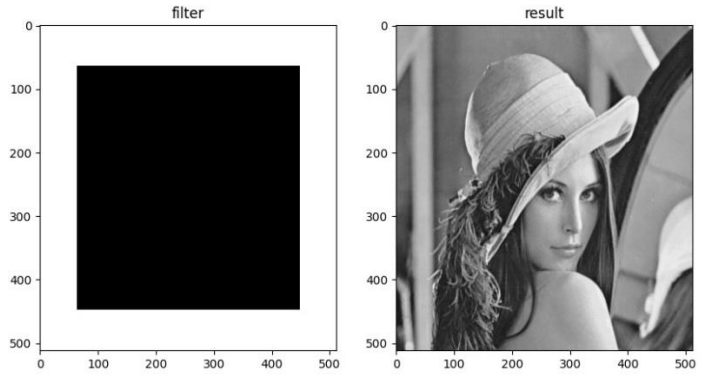
توجه داشته باشید که در این بخش مبدا فرکانسی در گوشه‌های تصویر قرار دارند نه در مرکز.

Filter a, T = ¼



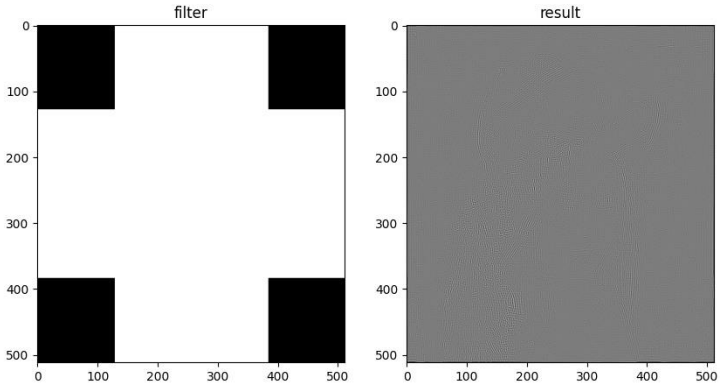
از تصویر فیلتر متوجه می‌شویم که این فیلتر فرکانس‌های بالا را حذف می‌کند در نتیجه باعث حذف لبه ها و smoothشدن تصویر می‌شود.

Filter a, T = 1/8



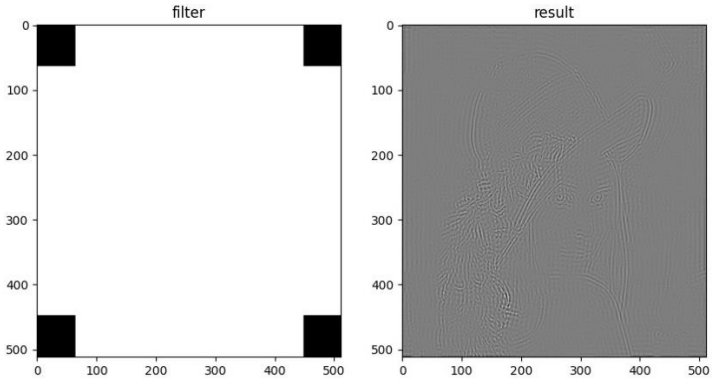
این فیلتر مشابه فیلتر قبل است با این تفاوت که باره بزرگتر از فرکانس‌های بالا حذف شده‌اند. در نتیجه تصویری تارتر داریم.

Filter b, T = ¼



همانطور که مشاهده می‌کنید این فیلتر فرکانس‌های پایین را حذف می‌کند و از آنجایی که بازه نسبتا بزرگی از آن‌ها را حذف کرده است، تقریبا تصویر از بین رفته است و تنها مقدار کمی از لبه‌ها باقی مانده است.

Filter b, T = 1/8



این فیلتر مشابه فیلتر قبل است و به دلیل اینکه بازه مقادیر فرکانس حذف شده کوچکتر بوده است، تصویر واضح تر است و تقریبا تمام لبه‌ها باقی مانده‌اند.

۴-پیوست

تابع مربوط به تبدیل فضای rectangular به polar

```
def rect_to_polar(img_ft):
    amp = np.absolute(img_ft)
    phase = np.angle(img_ft)
    return amp, phase
```

تابع مربوط به تبدیل فضای polar به rectangular

```
def polar_to_rect(amp, phase):
    return amp * np.exp(1j * phase)
```

تابع گرفتن لگاریتم از طیف تصویر و نرمال‌سازی آن

```
def log_ft(ft_img):
    res = np.log(ft_img + 1)
    res = normalize(res)
    return res
```

تابع نرمال‌سازی تصویر به ۰ تا ۲۵۵

```
def normalize(img):
    min = np.min(img)
    max = np.max(img)
    normalized = (img - min) / (max - min) * 255
    return normalized
```

سوال 4-1-1

```
from math import floor

import cv2
import numpy as np
import matplotlib.pyplot as plt

from utils import rect_to_polar, log_ft, polar_to_rect

img = cv2.imread('Lena.bmp', cv2.IMREAD_GRAYSCALE)
m = img.shape[0]
n = img.shape[1]
p = 2 * m
q = 2 * n
fp = np.zeros((p, q))
fp[0:m, 0:n] = img
img_ft = np.fft.fft2(fp)
img_amp, img_phase = rect_to_polar(img_ft)
img_amp = np.fft.fftshift(img_amp)
cv2.imwrite('4.1.1/Lena-FT.png', log_ft(img_amp))

a = np.asarray([
    [1, 2, 1],
    [2, 4, 2],
    [1, 2, 1]
])
a_row = np.asarray([[1, 2, 1]])
a_col = np.asarray([[1], [2], [1]])
b = np.asarray([
```

```
    [-1, -1, -1],
    [-1, 8, -1],
    [-1, -1, -1]
])
c = np.asarray([
    [0, -1, 0],
    [-1, 5, -1],
    [0, -1, 0]
])

filters = [a_row, a_col, a, b, c]
filters_coe = [1 / 4, 1 / 4, 1 / 16, 1, 1]
filter_names = ['A-row', 'A-col', 'A', 'B', 'C']
for idx, filter in enumerate(filters):
    fig, axs = plt.subplots(3, 1, figsize=(5, 15))

    padded_filter = np.zeros((p, q))
    padded_filter[
        floor(p / 2) - floor(filter.shape[0] / 2):floor(p / 2) +
floor(filter.shape[0] / 2 + 1),
        floor(q / 2) - floor(filter.shape[1] / 2):floor(q / 2) +
floor(filter.shape[1] / 2 + 1)
    ] = filters_coe[idx] * filter

    H = np.fft.fft2(padded_filter)
    H = np.fft.fftshift(H)
    HA, HP = rect_to_polar(H)

    axs[0].imshow(log_ft(HA), cmap='gray', aspect='auto')
    axs[0].set_title('filter {} in frequency
domain'.format(filter_names[idx]))
    axs[0].axis('off')

    res_ft = np.multiply(img_amp, HA)

    axs[1].imshow(log_ft(res_ft), cmap='gray', aspect='auto')
    axs[1].set_title('image after filter in frequency domain')
    axs[1].axis('off')

    res_ft = np.fft.ifftshift(res_ft)
    res_ft = polar_to_rect(res_ft, img_phase)
    res = np.fft.ifft2(res_ft)
    res = np.real(res[0:m, 0:n])

    axs[2].imshow(res, cmap='gray', aspect='auto')
    axs[2].set_title('final result')
    axs[2].axis('off')

    fig.savefig('4.1.1/{}.jpg'.format(filter_names[idx]))
```

سوال 4-1-2

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

from utils import rect_to_polar, normalize

images = ['Lena', 'Barbara', 'F16', 'Baboon']

for img_name in images:
    fig, axs = plt.subplots(2, 2, figsize=(10, 10))

    img = cv2.imread(img_name + ".bmp", cv2.IMREAD_GRAYSCALE)
    axs[0][0].imshow(img, cmap='gray', aspect='auto')
    axs[0][0].set_title('Original image')
    axs[0][0].axis('off')

    img_ft = np.fft.fft2(img)
    img_amp, img_phase = rect_to_polar(img_ft)
    axs[0][1].imshow(img_amp, cmap='gray', aspect='auto')
    axs[0][1].set_title('DFT')
    axs[0][1].axis('off')

    logged = normalize(np.log(img_amp + 1))
    logged = normalize(logged)
    axs[1][0].imshow(logged, cmap='gray', aspect='auto')
    axs[1][0].set_title('logged DFT')
    axs[1][0].axis('off')

    shifted = np.fft.fftshift(logged)
    axs[1][1].imshow(shifted, cmap='gray', aspect='auto')
    axs[1][1].set_title('shifted logged DFT')
    axs[1][1].axis('off')
    fig.savefig('4.1.2/{}-FT.png'.format(img_name))
```

سوال 4-2-2

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

from utils import rect_to_polar, log_ft, normalize, polar_to_rect

def generate_filter(f_name, T, N):
    filter = np.ones((N, N))
    if f_name == 'a':
        filter[int(T * N):int((1 - T) * N), int(T * N):int((1 - T) * N)] =
0
    elif f_name == 'b':
        filter[0:int(T * N), 0:int(T * N) + 1] = 0
        filter[0:int(T * N), int((1 - T) * N):N] = 0
        filter[int((1 - T) * N):N, 0:int(T * N) + 1] = 0
        filter[int((1 - T) * N):N, int((1 - T) * N):N] = 0
    return filter


img = cv2.imread('Lena.bmp', cv2.IMREAD_GRAYSCALE)
```

```
N = img.shape[0]

img_ft = np.fft.fft2(img)
img_amp, img_phase = rect_to_polar(img_ft)
cv2.imwrite('Lena-Ft.jpg', log_ft(img_amp))
Ts = [1 / 4, 1 / 8]
filter_names = ['a', 'b']

for f_name in filter_names:
    for t in Ts:
        fig, axs = plt.subplots(1, 2, figsize=(10, 5))

        H = generate_filter(f_name, t, N)

        axs[0].imshow(normalize(H), cmap='gray', aspect='auto')
        axs[0].set_title('filter')

        RES = img_amp * H

        RES = polar_to_rect(RES, img_phase)
        res = np.fft.ifft2(RES)
        res = res.real

        axs[1].imshow(normalize(res), cmap='gray', aspect='auto')
        axs[1].set_title('result')

        fig.savefig('4.2.1/H.{}-T.{}.jpg'.format(f_name, t))
```