

# حوزه فرکانس

## سحر شیخ الاسلامی

چکیده

اطلاعات گزارش

تصاویر را می توان در دو حوزه فرکانس و مکان بررسی کرد. در هر یک از این دو حوزه تصویر ویژگی های خاصی از خودش را نشان می دهد و هر یک از این حوزه مزایای و بدی های خودش را دارد. در این گزارش نگاهی به تصاویر در حوزه فرکانس می اندازیم و با ویژگی های تصویر در این حوزه و همچنین با اعمال فیلتر در این حوزه آشنا می شویم.

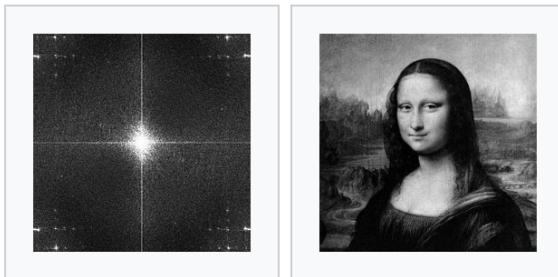
تاریخ: ۲۱/۹/۱۳۹۹

واژگان کلیدی:

Fourier Transform  
Frequency Domain  
Phase  
Magnitude  
Filtering  
DFT

$$F(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t)e^{-i\omega t} dt$$

یک تابع طی تبدیل فوریه از حوزه زمان به حوزه فرکانس انتقال پیدا می کند. با عملیات مشابه، می توانیم یک تابع را از حوزه مکان به حوزه فرکانس انتقال دهیم. همانطور که در مقدمه بررسی شد، با انتقال یک تابع از حوزه مکان به حوزه فرکانس، برخی از ویژگی ها این تابع آشکارتر می شود.



شکل ۱.۱: تصویر تابلو مونالیزا(سمت راست) و نمایش آن در حوزه فرکانس(چپ)  
سه فیلتری که در این سوال تعریف شده اند به شکل زیر هستند:

جزوی فوریه ریاضی دان فرانسوی در سال ۱۷۶۸ در فرانسه متولد شد. او در کتابی به این موضوع اشاره کرد که هر تابع پریودیکی را می توان به صورت مجموع توابع سینوسی نوشت. این نظریه اولیه او مقدمه ای برای یافتن تبدیل فوریه شد، که به جرئت از پرکاربردترین ریاضیات در حوزه مهندسی است. با این تبدیل می توان از حوزه زمان و مکان، به حوزه فرکانس برویم.

یکی از کاربردهای این حوزه فرکانس در پردازش تصویر است. انتقال تصویر به حوزه فرکانس، کاربردهای زیادی همچون فشرده سازی تصویر، اعمال فیلتر و ... دارد.  
می توان با استفاده از تبدیل فوریه عملیات دلخواه خود را در حوزه فرکانس بر روی عکس اعمال کرده، و با استفاده از معکوس تبدیل فوریه به حوزه مکان انتقال پیدا کرد در حالیکه تغییرات مورد نظر را اعمال کرده ایم.

## ۲- شرح تکنیکال

### ۴.۱.۱:

در این قسمت از ابتدا ما خواسته شده است که فیلتر های a,b,c را به حوزه فرکانس ببریم.

میدانیم که تبدیل فوریه، تبدیلی است که در آن هر تابع شبیه  $f(t)$  به یک تابع شبیه  $F(W)$  منعکس می شود.  $F(W)$  را تبدیل فوریه  $f(t)$  می دانیم که از رابطه زیر بدست می آید:

$$w = \mathbf{v}\mathbf{w}^T$$

از میان فیلتر های داده شده، تنها فیلتر  $a$  قابلیت جداسازی و نوشتگی به فرم بالا دارد. این فیلتر به شکل زیر قابل جداسازی است:

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

در ادامه خواسته شده این فیلتر ها را در حوزه فرکانس بر روی تصویر اعمال کنیم.

میدانیم عملیات اعمال یک فیلتر در حوزه مکان را کانونلوشن تعریف می شود، چرا که فیلتر را بر روی تصویر حرکت می دهیم و اعمال می کنیم. از طرف دیگر می دانیم که معادل عملیات کانونلوشن در حوزه مکان، عملیات ضرب در حوزه فرکانس است. پس بعد از انتقال فیلتر و تصویر به حوزه مکان، برای اعمال فیلتر تنها لازم است این دو را در هم ضرب کنیم. همانطور که در درس به آن اشاره شد، مراحل اعمال فیلتر در حوزه فرکانس به شکل زیر است.

۱. بدست آوردن  $P$  و  $Q$  (اندازه جدید تصویر) با توجه

به اندازه  $M$  و  $N$  تصویر  $f$ . عموما  $P=2N$  و  $Q=2M$  باشد.

۲. ساختن تصویر  $f_p$  با عمل افزودن صفر به تصویر  $f$

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

۴. بدست آوردن DFT حاصل

۵. اعمال فیلتر حقیقی با مبدا مرکز  $(P/2, Q/2)$  با ضرب آرایه

۶. انتخاب بخش حقیقی نتیجه (چون با تغییر ضرایب ممکن است بخش موهومی هم به تصویر اضافه شده باشد که قابل صرف نظر کردن است)

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

۸. برداشتن بخشی به اندازه  $M*N$  از گوشه بالای

سمت چپ  $g_p$

باید دقت شود که فیلتر را روی طیف تصویر در حوزه فرکانس اعمال می کنیم، زیر دست زدن به فاز عکس باعث به هم ریختگی شدید آن می شود.

#### قسمت ۴.۱.۲:

در این سوال از ما خواسته شده است که ابتدا عکس های داده شده را با کمک تبدیل فوریه از حوزه مکان به حوزه فرکانس انتقال دهیم.

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

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

برای بردن فیلتر ها از حوزه مکان به فرکانس باید تابع بالا در حالت گسسته را به آنها اعمال کنیم.

پس از انتقال فیلتر به حوزه فرکانس، حال باید مبدا فرکانسی آن را جا به جا کنیم. دلیل انجام این کار این است که هنگام اعمال فیلتر بر عکس در حوزه فرکانس، مهم است که مبدا فرکانسی این دو یکسان باشد. پس باید مبدا فرکانسی فیلتر را به مبدا فرکانسی تصویر انتقال دهیم. این عملیات باعث می شود نمایش بهتر فیلتر نیز می شود.

در ادامه باید از اعداد بدست آمده  $\log$  بگیریم، دلیل انجام این کار این است که پس از بردن از حوزه مکان به حوزه فرکانس، مبدا فرکانسی برابر مجموع تمام پیکسل های تصویر خواهد بود. این امر باعث می شود که طیف از کیفیت خوبی برخوردار نباشد، زیرا مقدار این نقطه اختلاف قابل توجهی با بقیه قسمت ها دارد و اعداد دارای رنج بسیار وسیعی هستند که نمایش آن را بسیار دشوار می کند. با استفاده از تابع  $\log$  میتوانیم این رنج را حد قابل قبولی کاهش دهیم. پس از انتقال به حوزه فرکانس، برای نمایش خروجی باید اعداد را نرمال کنیم به این معنی که حدود اعداد بین ۰-۲۵۵ باشد. به همین دلیل باید آنها را با استفاده از تابع زیر به این حدود بیاوریم:

$$f_m = f - \min(f)$$

$$f_s = K \left[ \frac{f_m}{\max(f_m)} \right]$$

که در این  $K = 255$  می باشد.

در ادامه سوال از خواسته شده است که با استفاده از ضرایب عکس عملکرد فیلتر را توضیح دهیم. در قسمت نتایج به تفصیل در این باره صحبت خواهیم کرد.

پس از آن خواسته شده اگر فیلتری جدا پذیر است، آن را به شکل جدا نوشته و هر بخش را به حوزه فرکانس ببریم. منظور از فیلتر جدا پذیر فیلتری است که بتوان آن را به شکل زیر نوشت:

با شیفت دادن این نقطه را به مرکز انتقال می‌دهیم. به این شکل اطلاعات مهم تصویر را به مرکز انتقال می‌دهیم که باعث می‌شود نمایش بهتری داشته باشیم. بنابراین با انتقال عکس به مرکز صفحه اطلاعات مهمتر در مرکز صفحه قرار میگیرند و طیف تصویر نمایش بهتری خواهند داشت.

#### قسمت ۴.۲.۱:

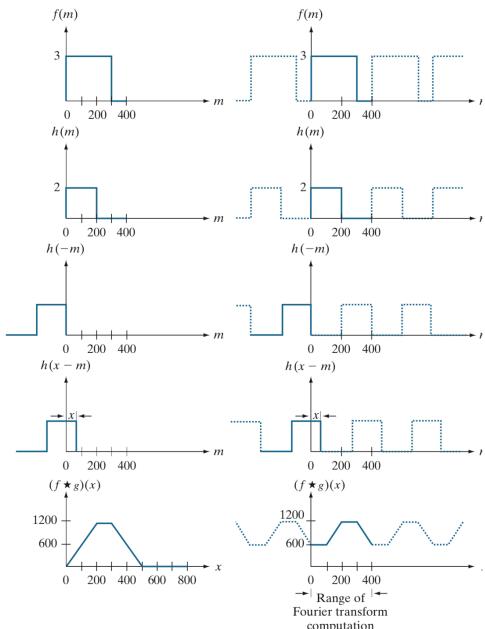
در این قسمت از ما خواسته شده است که مراحل اعمال فیلتر در حوزه فرکانس را بیان کنیم.

يعني هم عکس و هم فیلتر در حوزه مکان هستند و برای اعمال فیلتر در حوزه مکان باید از عملیات کانولوشن استفاده کنیم. به جای استفاده از عملیات کانولوشن، می‌توان دو عکس را در حوزه فرکانس برد و اعمال فیلتر را در این حوزه انجام داد.

مراحل اعمال فیلتر  $H(m,n)$  بر روی عکس  $F(m,n)$  در پایین بیان شده است:

۱) در ابتدا باید عملیات padding را انجام دهیم. همانطور که در تمرین‌های قبل بررسی شد padding انواع مختلفی دارد، اما در اینجا ما از zero padding استفاده می‌کنیم. دلیل این امر است که در هنگام تبدیل فوریه، ما فرض می‌کنیم که سیگنال پریودیک است.

شکل زیر را در نظر بگیرید:



شکل ۱.۲: محاسبه کانولوشن با فرض پریودیک بودن و غیر پریودیک بودن

همانطور که در شکل بالا مشاهده می‌شود، در نمودار های سمت چپ ما از کانولوشن در سیگنال غیر پریودیک استفاده

همانطور که قسمت ۴.۱.۱ توضیح دادیم، پس از استفاده از تابع فوریه ما دو عملیات شیفت دادن و لگاریتم گیری را انجام می‌دهیم. در قسمت از ما خواسته شده است که خروجی را بدون لگاریتم گیری و با لگاریتم گیری و بدون شیفت دادن و شده است.

ابتدا تاثیر لگاریتم گیری را بررسی می‌کنیم: برای بررسی تاثیر لگاریتم گیری بهتر است ابتدا نگاهی به تبدیل فوریه دو بعدی داشته باشیم. میدانیم تبدیل فوریه دو بعدی به شکل زیر است:

$$F(\mu, \nu) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(t, z) e^{-j2\pi(\mu t + \nu z)} dt dz$$

با توجه به فرمول بالا و برای مبدا فرکانسی، یعنی هنگامی که ۶ و ۷ هر دو مقدار صفر را دارند، مقدار  $F(u, v)$  برابر انتگرال  $(\text{یا در حالت گسسته سیگما})$  بر روی کل تصویر خواهد بود (زیرا توان  $e$  در این حالت برابر صفر خواهد بود که کل عبارت توانی مقداری برابر یک خواهد داشت)، به این معنی که مقداری برابر مجموع تمام پیکسل ها خواهد داشت. این در حالی است که برای بقیه مقداری این عدد می‌تواند بسیار کوچک باشد و اختلاف بسیار زیادی بین مقدار این نقطه و مقدادر بقیه نقاط وجود خواهد داشت.

پس محدوده اعداد پس از استفاده از تبدیل فوریه بسیار رنج بزرگی خواهد داشت. این رنج بزرگ باعث می‌شود که حتی پس از نرمال سازی و انتقال اعداد به بازه  $[-255, 255]$ ، با توجه به اختلاف زیاد مبدا فرکانسی با بقیه نقاط، اعداد در بقیه نقاط به هم بسیار نزدیک شده و از دید ناظر مقداری برابر داشته باشند (همه نقاط تقریباً سیاه) ولی مبدا یعنی  $F(0,0)$  مقدار بسیار زیادی دارد که به شکل یک نقطه سفید خود را نشان می‌دهد.

با لگاریتم گیری فاصله اعداد به شکل ملاحظه ای کم می‌شود. به طور مثال  $10^{1000000000}$  و  $10^0$  پس از لگاریتم گیری اختلاف  $10$  را خواهند داشت که قابل مقایسه با قبل از لگاریتم گیری نیست.

حال که اختلاف اعداد کم شده است و بازه آنها کاهش پیدا کرده است، می‌توانیم آنها را به خوبی نمایش دهیم.

حال می‌خواهیم تاثیر شیفت دادن را بررسی کنیم. مبدا فرکانس (منظور  $F(0,0)$  می‌باشد) در حوزه فرکانس قسمت مهمی از کل تصویر در حوزه فرکانس است، این نقطه شامل اطلاعات مهمی از تصویر است. علاوه بر آن نقاط مجاور آن (یعنی نقاط با فرکانس پایین) نیز حاوی اطلاعات مهمی هستند، به طوری که شاید نگه داشتن ضریب بخش کوچکی از فرکانس‌های پایین، اطلاعات زیادی درباره کلیت و شکل عکس به ما بدهد.

قبل از شیفت دادن این نقطه در گوشش  $\circ$  بالا سمت راست تصویر قرار دارد.

با استفاده از padding عملیات اعمال فیلتر در حوزه فرکانس مشابه اعمال فیلتر در حوزه مکان خواهد شد، و فرض پریویدیک بودن سیگنال دیگر مشکلی ایجاد نمی کند. در بیشتر موارد سایز عکس و فیلتر را دو برابر در نظر میگیرند(محض اطمینان). اما به طور کل به اندازه مجموع فیلتر و عکس در بیشتر کاربردها کافی است.

(۲) در مرحله بعد مبدا فرکانسی را هم در تصویر و هم در فیلتر به مرکز صفحه منتقل می کنیم.

برای این کار کافی است که تصویر و فیلتر را در  $y^{x+1}$  ضرب کنیم. این کار فقط برای نمایش بهتر طیف انجام می شود. نکته‌ی قابل توجه در این قسمت این است که فیلتر و تصویر هر دو باید مبدأ یکسان داشته باشند.(یا هر دو بالا سمت چپ و بدون شیفت دادن و یا هر دو در مرکز)

(۳) حال باید تصویر و فیلتر را به حوزه فرکانس انتقال دهیم. این کار را به کمک تبدیل فوریه دو بعدی گستته انجام می دهیم. با اعمال فرمول پایین به تصویر و به فیلتر هر دو به حوزه فرکانس انتقال بپدا می کنیم.

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

(۴) پس از انتقال فیلتر و تصویر به حوزه فرکانس، حال طیف این دو را در هم ضرب آرایه ای می کنیم. می دانیم که فاز عکس بخش مهمی از آن است و با کوچکترین تغییری این فاز به هم میریزد. اعمال فیلتر بر روی فاز عکس باعث به هم ریختن کامل عکس می شود، به همین دلیل ما فیلتر را روی طیف عکس ها اعمال می کنیم. نکته دیگر این است که کانولوشن در حوزه مکان، معادل عملیات ضرب در حوزه فرکانس است.

$$(f \cdot h)(x,y) \Leftrightarrow \frac{1}{MN} (F \star H)(u,v)$$

(۵) پس از انجام ضرب، در واقع فیلتر را در حوزه فرکانس به عکس اعمال کردیم. پس حال با برگرداندن عکس به حوزه مکان عکسی داریم که فیلترینگ بر روی آن انجام شده است. این کار را با استفاده از عملگر فوریه معکوس انجام می دهیم. در حین انجام عملگر فوریه معکوس ممکن است ناخواسته برای تصویر بخش موهومی ایجاد شود. باید این بخش را در نظر نگیریم، چرا که ما فیلتر را روی طیف تصویر اعمال کردیم و حال خروجی باید طیف تصویر باشد. پس از گرفتن خروجی طیف، فاز را که جدا کرده بودیم باید به عکس اضافه کنیم.

(۶) با توجه به اینکه در مرحله مبدا فرکانسی را جا به جا کردیم و به مرکز تصویر انتقال داده ایم، در این مرحله نیز باید عکس

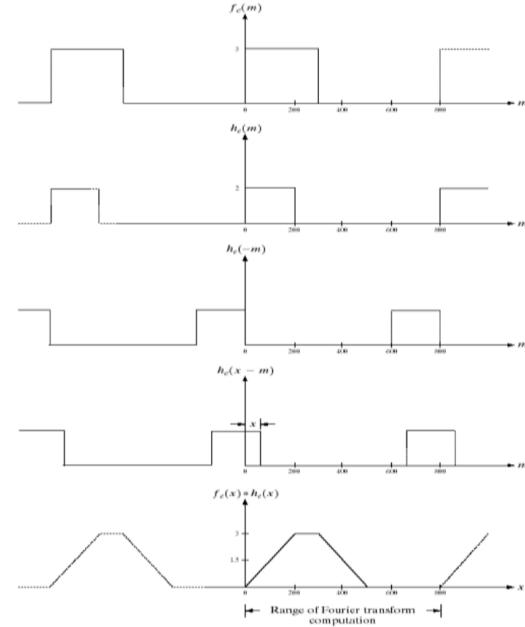
کردیم، یعنی از فرمول پایین برای اعمال فیلتر استفاده کرده ایم:

$$(w \star f)(x,y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s,t) f(x-s, y-t)$$

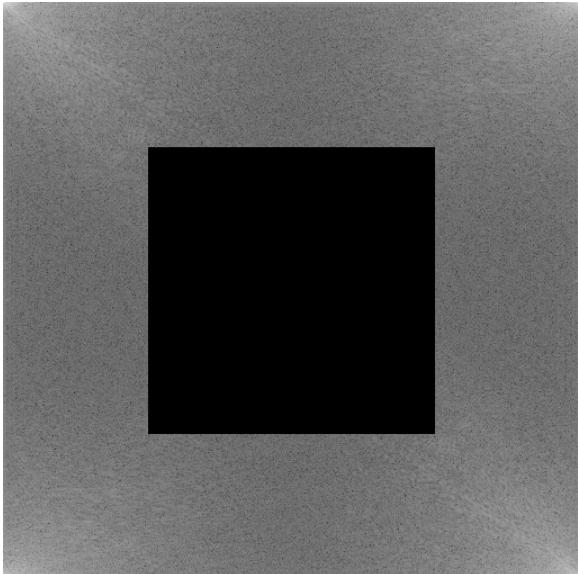
نتیجه در شکل پایین سمت چپ مشخص شده است، در واقع ما به دنبال چنین جوابی هستیم. اما در نمودار های سمت راست، با فرض پریویدیک بودن سیگنال و با استفاده از کانولوشن circular این کار را انجام داده ایم، می بینیم که در شکل سمت راست پایین یک خط بوجود آمده است، که این خط در واقع اشتباه است. با استفاده از zero padding این مشکل حل می شود و در سیگنال های پریویدیک هم نتیجه ای مانند شکل چپ پایین خواهیم گرفت.

این مشکل به این دلیل پیش آمده است که ما از تبدیل فوریه گستته (DFT) دو بعدی استفاده می کنیم و تعداد نمونه های محدودی داریم(در اینجا برابر سایز تصویر و سایز فیلتر)، دوره تناوب سیگنال ما با توجه به تعداد نمونه های ما تعیین می شود.

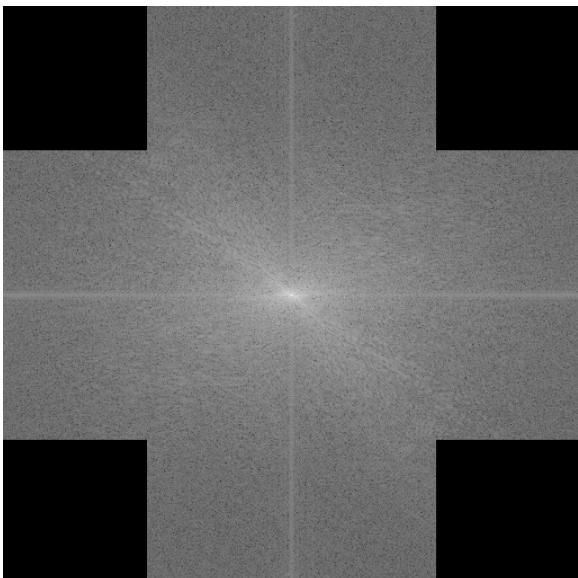
برای حل مشکل کافی است، سایز تصویر و فیلتر را برابر مجموع سایز هر دو قرار دهیم(به جای اینکه فقط سایز فیلتر را برابر اندازه سایز عکس قرار دهیم). در این مثال اگر سایز هر دو را برابر  $267 \times 267$  در نظر بگیریم، این مشکل حل خواهد شد.



شکل ۱.۳: نتیجه استفاده از padding



شکل ۱.۴: نتیجه اعمال فیلتر  $a$  بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و بدون شیفت



شکل ۱.۵: نتیجه اعمال فیلتر  $a$  بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و با شیفت

این کار را انجام دهیم . در واقع باید مرکز را به گوشه سمت چپ و بالا انتقال دهیم. توجه شود که هر دوی این عملیات برای نمایش بهتر نتیجه در حوزه فرکانس و حوزه مکان است.

(۷) با توجه به اینکه سایز تصویر را تغییر داده ایم، در این مرحله با سایز تصویر را به اندازه اصلی برگردانیم. برای انجام این کار باید به مقدار اندازه اصلی عکس(که در اینجا  $256*256$ ) است از گوشه بالا سمت چپ برداریم.

در ادامه سوال از ما خواسته است که نتیجه اعمال فیلتر با استفاده از کانولوشن در حوزه مکان را با نتیجه اعمال مستقیم آن در حوزه فرکانس مقایسه کنیم.

در واقع در اینجا ما از zero-padding استفاده نکردیم، بلکه تصویر و فیلتر را با تبدیل فوریه به حوزه فرکانس بردیم، در آن حوزه ضرب کردیم و به حوزه مکان باز گرداندیم.(مطلوب فرمول زیر)

$$Z = IDFT(DFT(X) \cdot DFT(H))$$

همانطور که بالاتر توضیح دادیم و در شکل ۱.۲ مشخص است نتیجه این عملیات یکسان نیست. همانطور که در شکل ۱.۲ مشاهده می شود، خروجی برای این عملیات در اکثر نقاط تصویر یکسان است اما در لبه ها نتیجه متفاوت است.

#### ۴.۲.۲:

در این سوال از ما خواسته شده از که ضرایب خاصی را از تصویر Lena در حوزه فرکانس صفر کنیم.

برای این قسمت ابتدا با استفاده از تبدیل فوریه دو بعدی گستته ، تصویر را به حوزه فرکانس انتقال بدھیم حال با داشتن ضرایب فرکانسی، کافی است هر کدام از ضرایب که مد نظر سوال است را صفر کنیم. در این قسمت برای هر بخش، شکل فیلتر در حوزه فرکانس نتیجه را جداگانه بررسی میکنیم. نتیجه نهایی در قسمت نتایج آمده است.

#### قسمت : a

در بخش به ازای دو  $T$  مختلف از ما خواسته شده است که ضرایب فرکانس های بین  $TN$  و  $(1-T)N$  را صفر کنیم.

بنابراین در طیف تصویر، باید به اندازه این رنج فرکانسی که شکلی شبیه یک مربع در وسط تصویر دارد، سیاه شود(صفر شود) شکل طیف را در دو حالت بدون شیفت و با استفاده از شیفت مشاهده می کنیم.

اگر به طیف تصویر پس از شیفت دادن نگاه کنیم ، می بینیم که در هر دو حالت  $T = \frac{1}{4}$  و  $T = \frac{1}{8}$  ضریب فرکانس های گوشه طیف که از مبدا فرکانسی دورتر بوده و مقدار آنها بزرگتر است ، صفر شده است. بنابراین می توان گفت  $a$  مانند یک فیلتر low pass عمل کرده ، فرکانسهای کوچک را عبور میدهد و فرکانسهای بزرگ را صفر می کند.

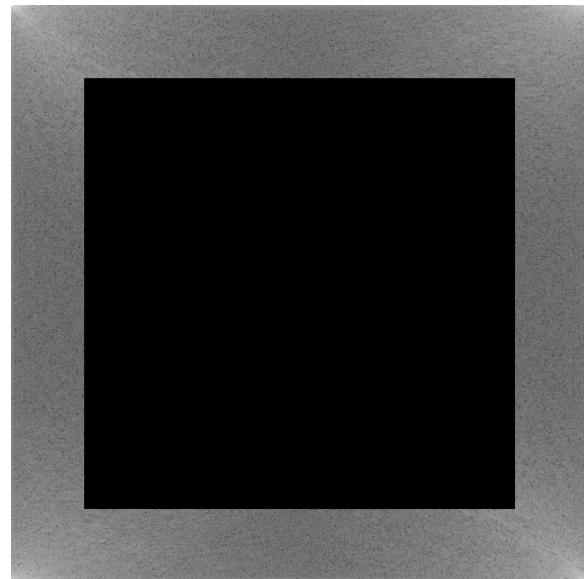
در مورد تفاوت  $T = \frac{1}{4}$  و  $T = \frac{1}{8}$  نیز میتوان گفت که در حالت دوم مریع سیاه بزرگتر بوده و در نتیجه در رنج فرکانسی بزرگتری ، ضرایب صفر میشود و شدت low pass بودن بیشتر است. در واقع فقط فرکانسهای کوچکتر عبور داده میشوند. البته این فیلتر تفاوت مهمی با فیلتر lowpass دارد و آن هم این است که همان طور که در تصاویر بالا میبینیم مقدار تمام فرکانسهای بزرگ را صفر نمیکند بلکه فقط فرکانس های گوشه طیف صفر می شوند. در واقع از میان فرکانسهای بزرگ ، آهایی که در جهت افقی و عمودی هستند نیز عبور داده میشوند.

#### قسمت b:

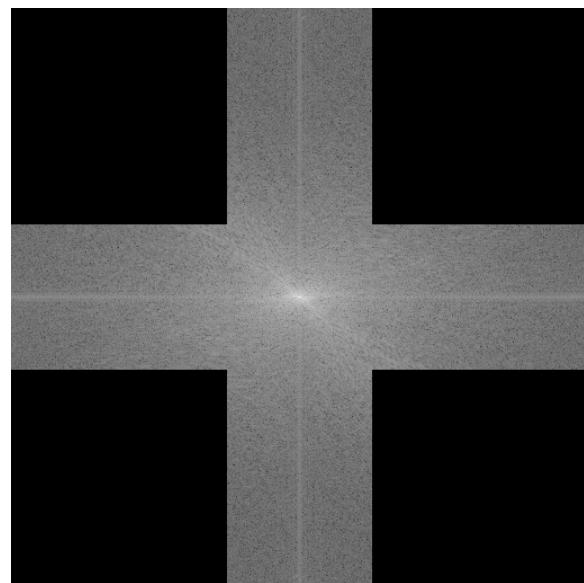
در این حالت برخلاف  $a$  هر دفعه یکی از گوشه های تصویر سیاه میشود . در واقع در هر حالت یک مریع در گوشه تصویر که طول و عرض آن  $T$  برابر طول و عرض تصویر است سیاه شده و ضرایب فرکانسی آن صفر میشود. در حالت  $i$  گوشه بالا سمت چپ ، در  $ii$  بالا سمت راست ، در  $iii$  پایین سمت چپ

و در  $iv$  گوشه پایین سمت راست سیاه شده است. پس از شیفت دادن طیف آنها میبینیم که در واقع مریع سیاه شده در این ۴ حالت در ناحیه چهارم ، سوم ، اول و دوم مثلثاتی قرار می گیرند.

اگر هر ۴ را فیلتر را روی تصویر اعمال کنیم ، در فیلتر نهایی میبینیم که یک مریع سیاه در مرکز تصویر وجود دارد یعنی این فیلتر ، یک فیلتر high pass خواهد بود که فرکانسهای کوچک که در اطراف مبدا فرکانسی هستند را صفر کرده و فرکانسهای بزرگ را که از مبدا دور هستند، عبور میدهد اما اگر هریک از این ۴ حالت را جداگانه بررسی کنیم ، میبینیم که هر کدام یک بخش از فرکانس های کوچک را صفر می کنند و بنابراین فیلتر high pass محسوب نمیشوند. در مورد تفاوت  $T = \frac{1}{4}$  و  $T = \frac{1}{8}$  نیز باید گفت که در حالت دوم که مقدار  $T$  کوچکتر است ، مساحت مریع سیاه کوچک تر است و در نتیجه رنج فرکانسی کوچکتری صفر شده و تصویر به تصویر اصلی نزدیکتر است.



شکل ۱.۶: نتیجه اعمال فیلتر  $a$  بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و بدون شیفت



شکل ۱.۷: نتیجه اعمال فیلتر  $a$  بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و با شیفت

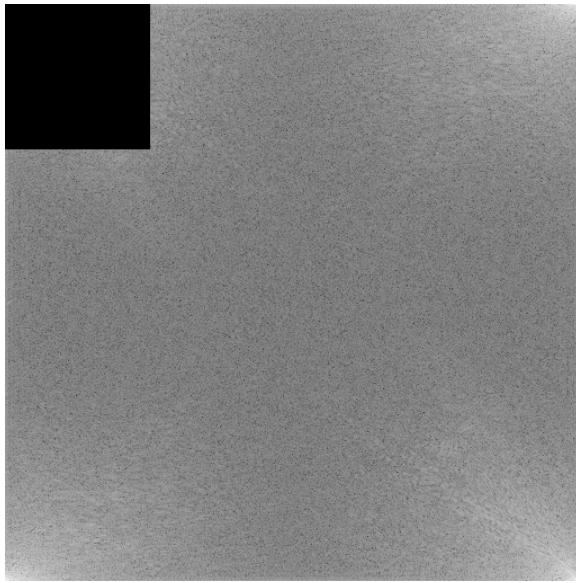
در مورد حالت ۱ ذکر یک نکته ضروری است :

اگر به صورت سوال دقت کنید میبینید که در این حالت ، نقطه [٠,٠] نیز در رنجی قرار میگیرد که مقدار آنها صفر می شود بنابراین مقدار این نقطه که مبدأ فرکانسی بوده و شامل اطلاعات اصلی تصویر است صفر شده وبخش هایی از تصویر از بین میرود . در بخش نتایج که تصاویر نهایی گزارش شده می توانید به خوبی این موضوع و تفاوت فاحش خروجی این بخش و سایر بخشها را ببینید. برای اینکه بهتر بتوان این ۴ حالت را با هم مقایسه کرد ، در بخش نتایج علاوه بر خروجی اصلی حالت ۱ تغییر کوچکی در این فیلتر داده شده و خروجی آن در حالتی که مقدار نقطه [٠,٠] صفر نشده باشد نیز گزارش شده است.

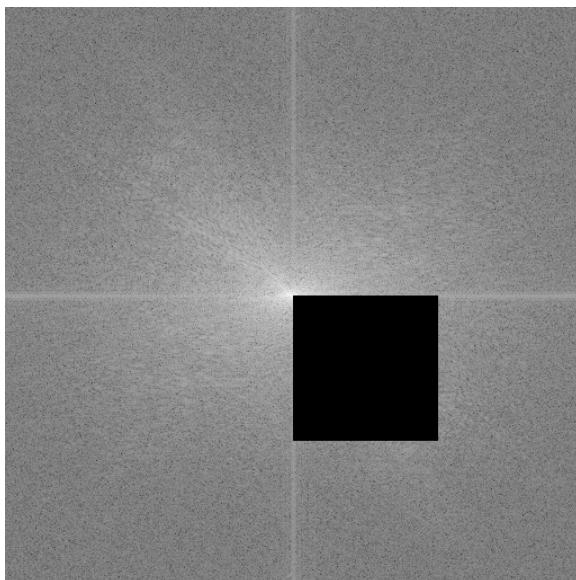
در ادامه نتیجه اعمال ۴ قسمت با هم نیز آمده است. هر کدام از این ۴ قسمت به تنها ی شبهه هیچ فیلتر خاصی عمل نمی کنند. اما اعمال هر ۴ تا همزمان شبهه فیلتر edge detection عمل می کند.

با مشاهده شکل های ۱.۲۶ تا ۱.۲۷ میبینیم که مراکز فرکانسی که نشانگر فرکانس های پایین است ضریب به شدت پایین و نزدیک صفر دارند و فرکانس های بالاتر ضریب بیشتر دارند. با توجه به اینکه فرکانس های بالا نشان گر جزیات تصویر و فرکانس های پایین نشانگر کلیات تصویر است، این فیلتر کلیات تصویر را از بین برده و لبه ها را تقویت می کند، بنابراین این فیلتر edge detection خواهد بود.

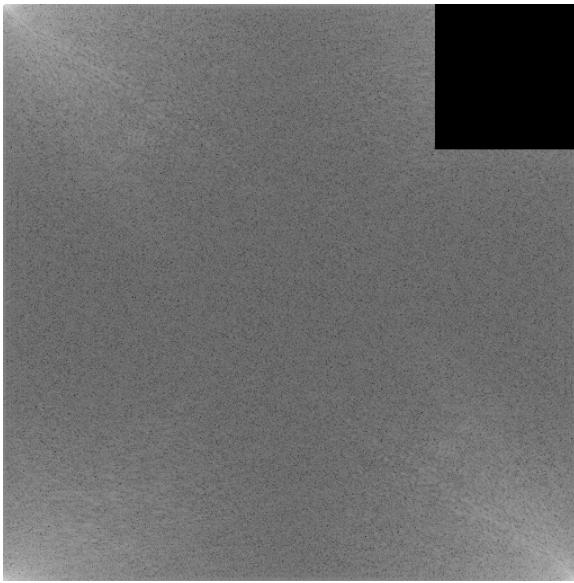
برای فهم بهتر مانند حالت قبل طیف تصویر پس از اعمال تغییرات خواسته شده در هر بخش در زیر گزارش میشود. دقت کنید که پس از اعمال این تغییرات طیف ها شیفت داده شده‌اند که تصویر انها پیش و پس از شیفت در زیر می آید :



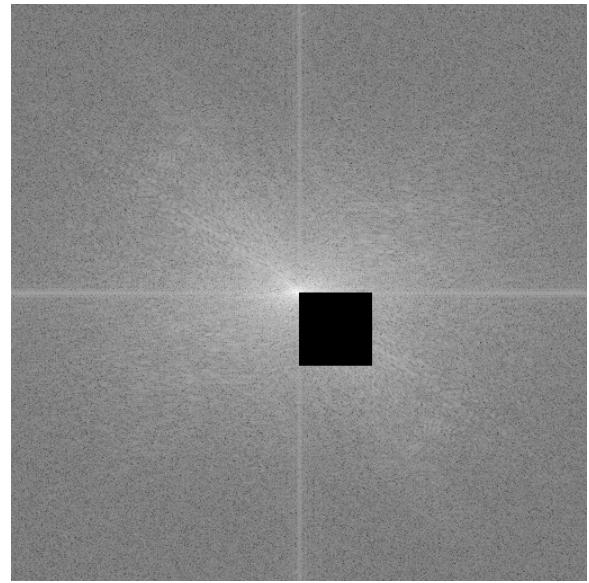
شکل ۱.۸: نتیجه اعمال فیلتر b قسمت ا بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و بدون شیفت



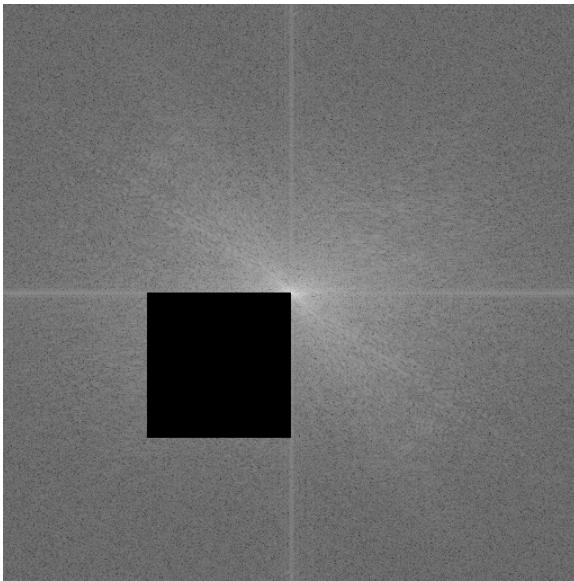
شکل ۱.۹: نتیجه اعمال فیلتر b قسمت ا بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و با شیفت



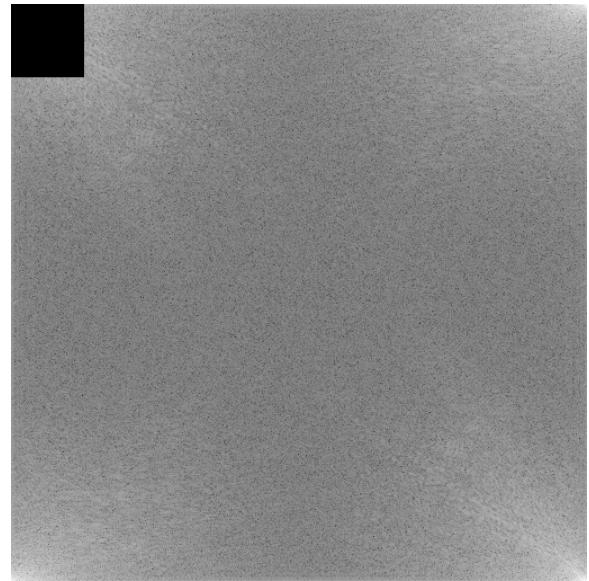
شکل ۱.۱۲: نتیجه اعمال فیلتر b قسمت ii بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و بدون شیفت



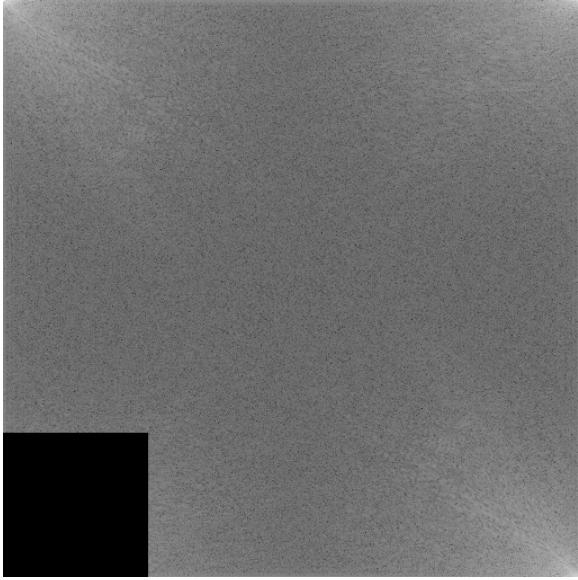
شکل ۱.۱۰: نتیجه اعمال فیلتر b قسمت i بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و بدون شیفت



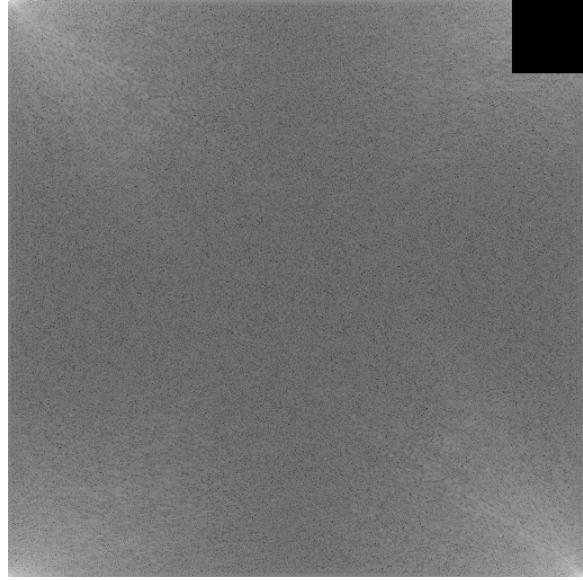
شکل ۱.۱۳: نتیجه اعمال فیلتر b قسمت ii بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و با شیفت



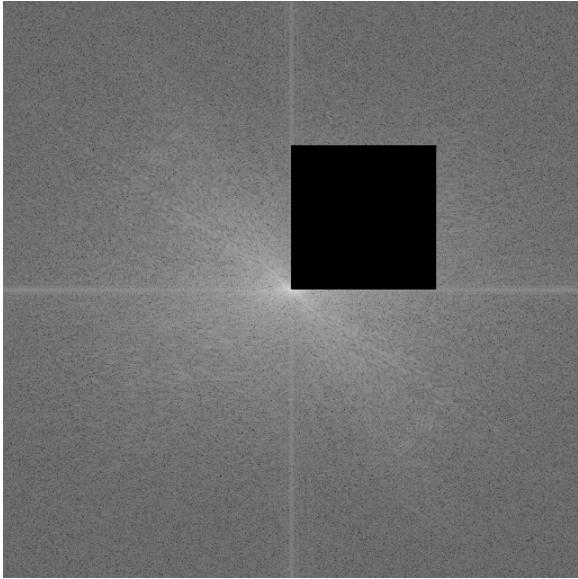
شکل ۱.۱۱: نتیجه اعمال فیلتر b قسمت i بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و با شیفت



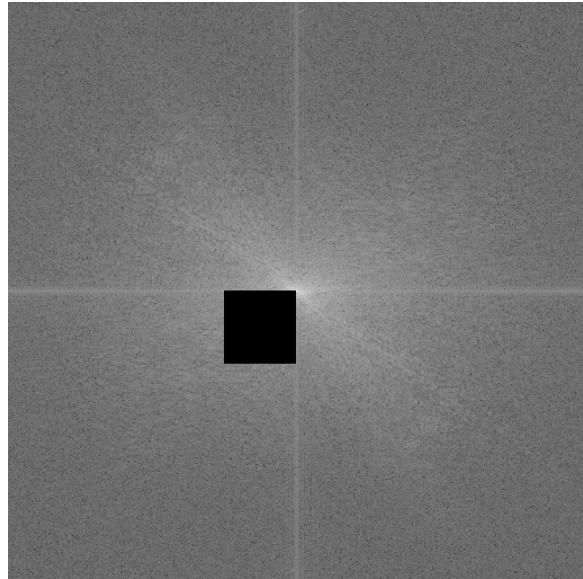
شکل ۱.۱۶: نتیجه اعمال فیلتر b قسمت iii بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و بدون شیفت



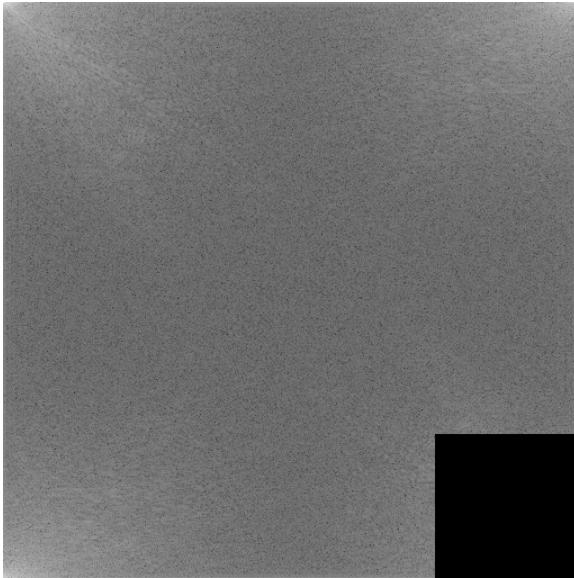
شکل ۱.۱۴: نتیجه اعمال فیلتر b قسمت ii بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و بدون شیفت



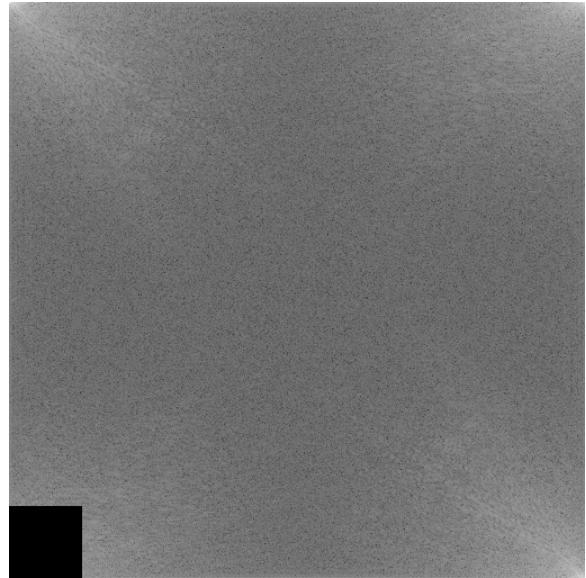
شکل ۱.۱۷: نتیجه اعمال فیلتر b قسمت iii بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و با شیفت



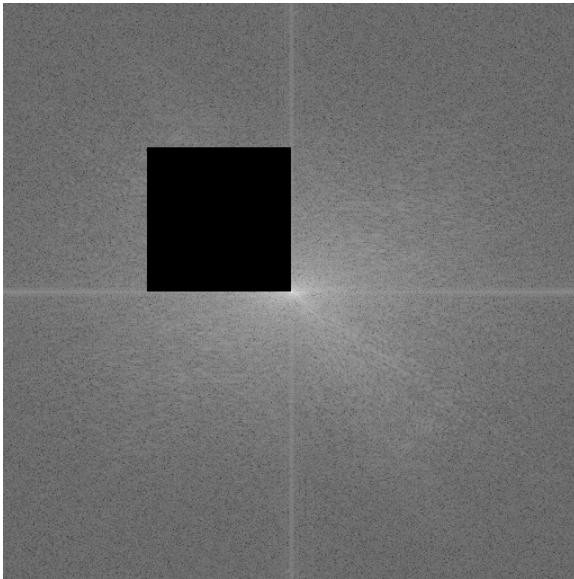
شکل ۱.۱۵: نتیجه اعمال فیلتر b قسمت ii بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و با شیفت



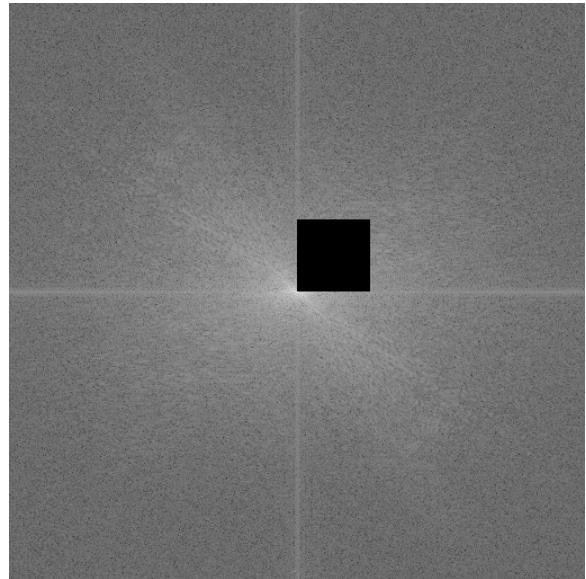
شکل ۱.۲۰: نتیجه اعمال فیلتر b قسمت iv بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و بدون شیفت



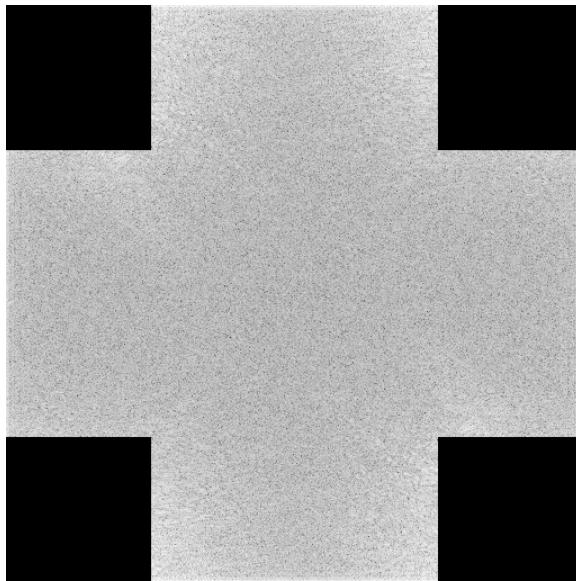
شکل ۱.۱۸: نتیجه اعمال فیلتر b قسمت iii بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و بدون شیفت



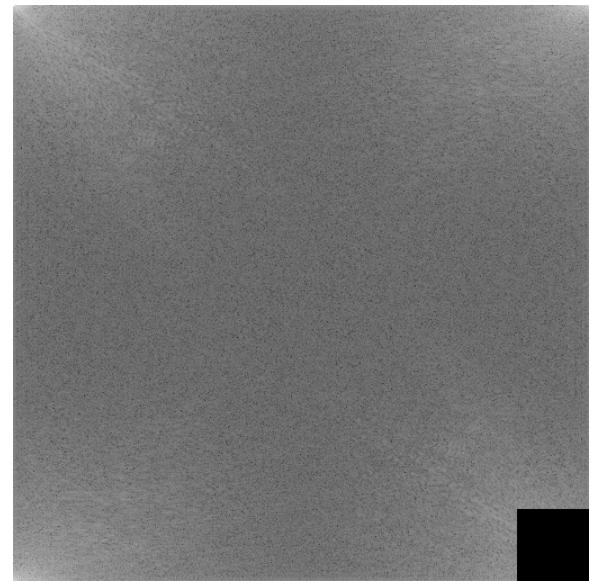
شکل ۱.۲۱: نتیجه اعمال فیلتر b قسمت iv بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و با شیفت



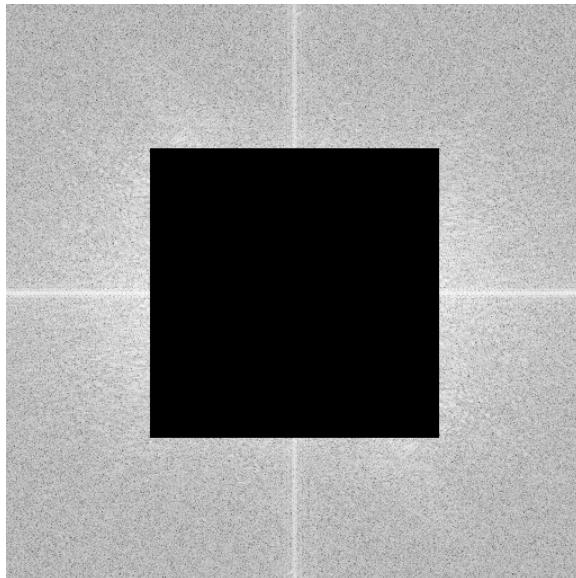
شکل ۱.۱۹: نتیجه اعمال فیلتر b قسمت iii بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و با شیفت



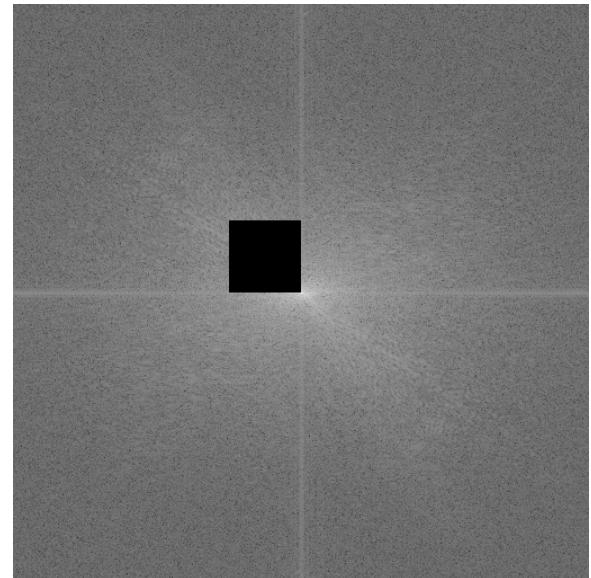
شکل ۱.۲۰: شکل ۱.۲۰: نتیجه اعمال فیلتر b (تمام قسمت ها) بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و بدون شیفت



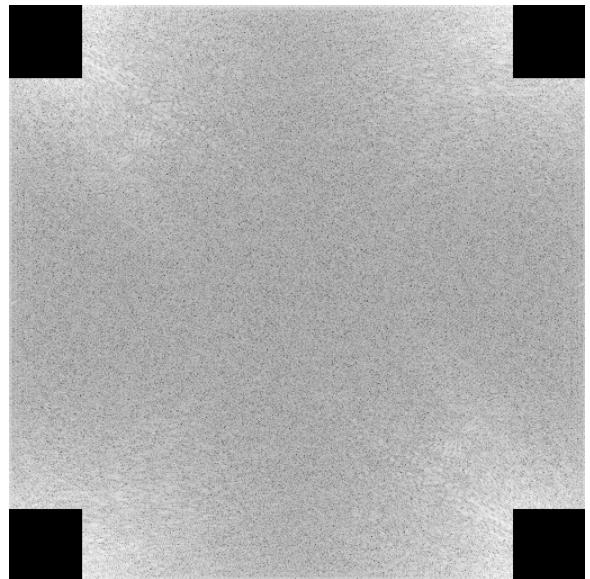
شکل ۱.۲۰: شکل ۱.۲۰: نتیجه اعمال فیلتر b قسمت iv بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و بدون شیفت



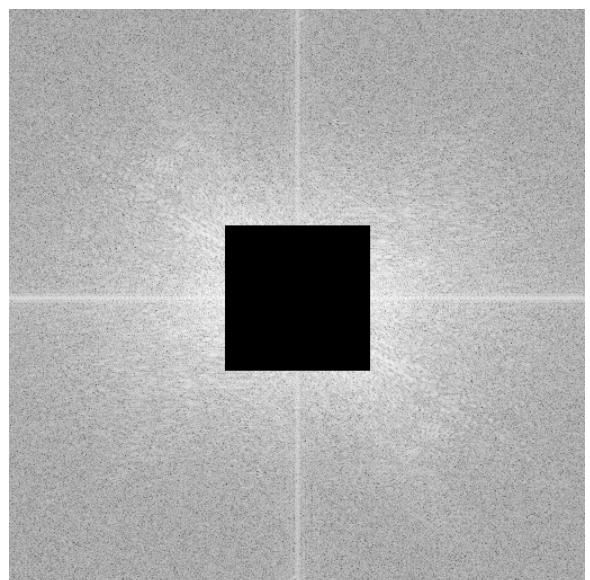
شکل ۱.۲۵: شکل ۱.۲۰: نتیجه اعمال فیلتر b قسمت iv بر روی تصویر Lena با  $T = \frac{1}{4}$  در حوزه فرکانس و با شیفت



شکل ۱.۲۳: شکل ۱.۲۰: نتیجه اعمال فیلتر b قسمت iv بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و با شیفت



شکل ۱.۲۶: شکل ۱.۲۰: نتیجه اعمال فیلتر  $b$  تمام قسمت  
ها بر روی تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و بدون  
شیفت



شکل ۱.۲۷: نتیجه اعمال فیلتر  $b$  تمام قسمت ها بر روی  
تصویر Lena با  $T = \frac{1}{8}$  در حوزه فرکانس و با شیفت

## نتایج:

قسمت ۴.۲.۱:

برای هر فیلتر خروجی را جدا بررسی می کنیم.

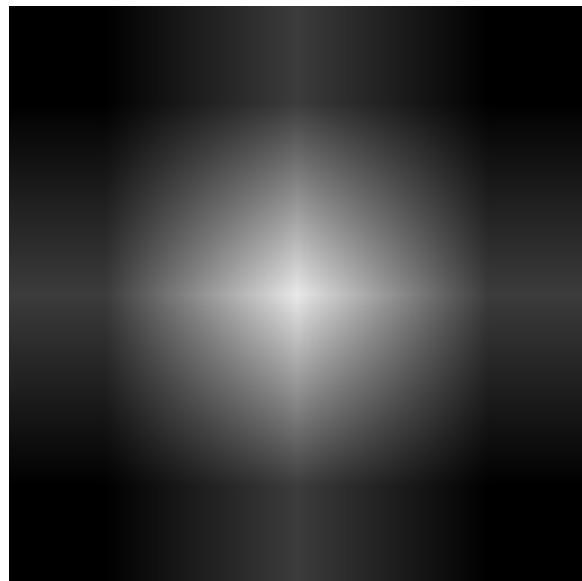
فیلتر a:

فیلتر a به شکل زیر تعریف شده است.

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

با استفاده از ضرایب فیلتر، مشخص است که فیلتر میانگین گیر گوسی است، این فیلتر به این شکل عمل میکند که میانگین وزن داری از یک پیکسل و همسایگی آن بر می گرداند.

این وزن ها با فاصل پیکسل از مرکز نسبت عکس دارد، یعنی سعی شده است که پیکسل هایی که فاصله کمتری دارند و به عکس اصلی نزدیکتر هستند، در اندازه این پیکسل نقش مؤثرتری داشته باشند. میدانیم که فیلتر های میانگین گیر، برای smooth کردن عکس هستند، چرا که لبه های کمرنگ می کنند، و به طور کل جزییات عکس را کمرنگ می کنند. این فیلتر در حوزه فرکانس به شکل خواهد بود:



شکل ۲.۱: فیلتر a در حوزه فرکانس



شکل ۲.۲: تصویر اصلی



شکل ۲.۳: تاثیر اعمال فیلتر a بر روی شکل ۲.۲ در حوزه فرکانس

همانطور که در تصویر مشاهده می شود، مرکز تصویر روشن و اطراف تصویر تیره است. بنابراین اطلاعات مرکز تصویر به خوبی حفظ می شود و اطلاعات گوشه های تصویر را به خوبی حفظ نمی کند و آنها را حدودا حذف می کند.

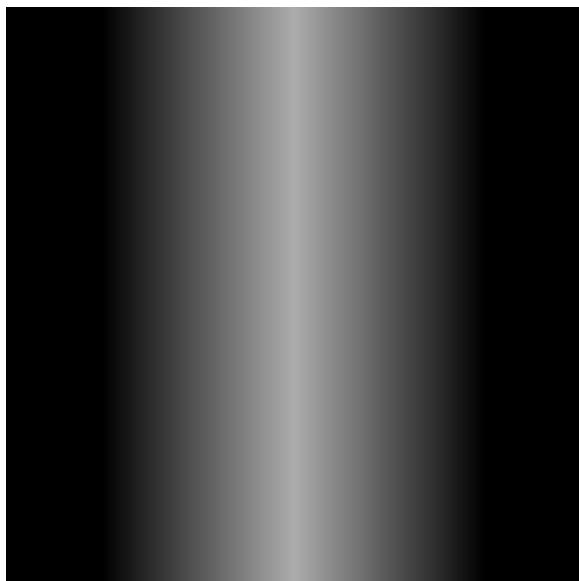
تاثیر اعمال این فیلتر بر عکس به شکل زیر خواهد بود:



شکل ۲.۵: تاثیر اعمال فیلتر شکل ۲.۴ بر تصویر اصلی همانطور که مشاهده می شود، لبه های افقی تصویر کمی تار شده است.

### [1 2 1] :

این فیلتر برخلاف فیلتر قبلی، مقدار پیکسل مرکزی را با همسایه های افقی آن جمع می کند که این امر باعث کاهش اثر لبه های عمودی می شود. این فیلتر در حوزه فرکانس به شکل زیر خواهد بود:



شکل ۲.۶: بخش دیگر فیلتر a در حوزه فرکانس

همانطور که بالاتر توضیح داده شد، تاثیر اعمال این فیلتر عکس را smooth می کند، و از پرنگی لبه ها می کاهد. مقایسه شکل ۲.۲ و ۲.۳ این موضوع آشکار می شود. چرا که شکل ۲.۳ در مقابل شکل ۲.۲ کمی تار تر است.

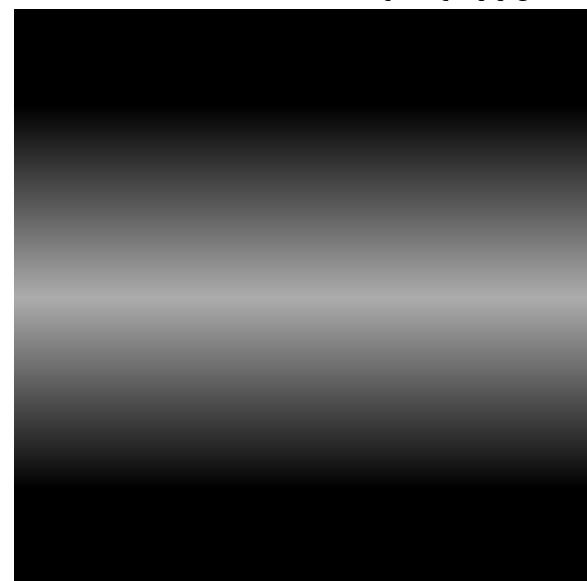
با توجه به آنکه در شرح توضیح دادیم، فیلتر a فیلتری جدا پذیر بود و میتوان آن را به شکل زیر نوشت:

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

حال می خواهیم هر کدام از این فیلتر های یک بعدی را به حوزه مکان برد و تاثیر آن را بر تصویر مشاهده کنیم:

$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} :$$

این فیلتر مقدار پیکسل اصلی را با دو همسایه عمودی اش جمع می کند، و میانگین می گیرد. در واقع این فیلتر باعث ضعیف شدن لبه های افقی می شود. این فیلتر در حوزه فرکانس به شکل زیر خواهد بود.



شکل ۲.۷: یک بخش از فیلتر a در حوزه فرکانس

میدانیم که تغییرات در حوزه فرکانس بر تغییرات بر حوزه مکان عمود است. مشاهده میکنیم این فیلتر در حوزه فرکانس لبه های افقی را حفظ می کند و لبه های عمودی را از بین می برد، با توجه به عمود بودن تغییرات در حوزه مکان و زمان پس این فیلتر دقیقاً لبه های افقی در حوزه مکان را کمزنگ می کند.



شکل ۲.۸: تاثیر اعمال فیلتر شکل ۲.۴ و سپس ۲.۶ بر عکس اصلی

همانطور که مشاهده می شود این عکس مشابه شکل ۲.۳ است که یعنی اعمال این دو فیلتر پشت سر هم، همانند یک بار اعمال فیلتر  $a$  است.

همانطور که مشاهده می شود، این فیلتر در حوزه فرکانس لبه های عمودی را حفظ می کند و لبه های افقی را کمزنگ کرده یا از بین می برد. با توجه به اینکه نکته تغییرات در حوزه زمان و مکان بر هم عمودند، بنابراین حفظ لبه های عمودی و از بین بردن لبه های افقی در حوزه فرکانس به معنی حفظ لبه های افقی و کمزنگ کردن لبه های عمودی در حوزه مکان است.

خروجی پس از اعمال فیلتر شکل ۲.۶ بر تصویر اصلی به شکل زیر است:



شکل ۲.۷: تاثیر اعمال فیلتر شکل ۲.۶ بر تصویر اصلی

همانطور که مشاهده می شود، لبه های عمودی تصویر کمی نرم شده است.

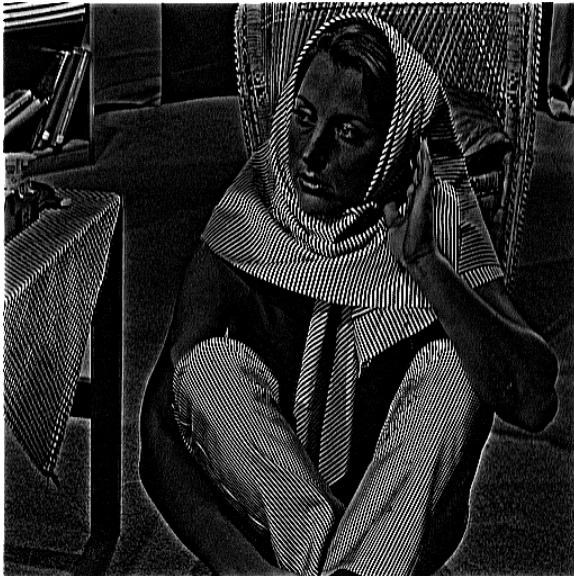
اگر ابتدا فیلتر تضعیف لبه های افقی و سپس فیلتر تضعیف لبه های عمودی را به عکس اعمال کنیم نتیجه مانند شکل زیر خواهد بود:

:فیلتر  $b$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

با نگاه کردن به ضرایب فیلتر، متوجه می شویم که فیلتر بالا یک فیلتر لاپلاسین است. فیلتر لاپلاسین که عملیاتی مشابه مشتق گیری دارد برای تشخیص لبه کاربرد دارد.

این فیلتر اختلاف هر پیکسل را با تمام پیکسل های مجاور بدست می آورد به جای پیکسل اصلی میگذارد. این فیلتر لبه ها در تمام جهات محاسبه می کند، چرا که در تمام اختلاف را در تمام جهات بدست می آورد. شکل زیر این فیلتر را در حوزه فرکانس نشان می دهد:



شکل ۲.۹: تاثیر اعمال فیلتر b بر تصویر اصلی

همانطور که مشاهده می شود، لبه ها در این عکس تقویت شده است و جزیات تصویر سیاه شده است و از بین رفته است.

فیلتر c:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

با مشاهده ضرایب فیلتر، مشخص می شود که این فیلتر نیز یک فیلتر لاپلاسین است که برای تشخیص لبه به کار می رود. این فیلتر اختلاف هر پیکسل را با پیکسلهای مجاور افقی و عمودی اش محاسبه کرده و مجموع این اختلافها را با مقدار پیکسل جمع میکند.

برخلاف فیلتر b که اختلاف هر پیکسل را با تمام همسایه‌ی افقی، عمودی و قطری بدست می آورد، این فیلتر فقط اختلاف یک پیکسل را با همسایه‌ی عمودی و افقی بدست می آورد. فرق مهم دیگر این نکته است که این فیلتر پس از یافتن اختلاف تصویر اصلی را به آن اضافه می کند، بنابراین لبه های عمودی و افقی را بدست آورده و آنها تقویت می کند ولی بقیه عکس تغییری نمی کند.

این فیلتر در حوزه فرکانس به شکل زیر خواهد بود:

شکل ۲.۱۰: فیلتر b در حوزه فرکانس

همانطور که مشاهده می شود، ضرایب این فیلتر برای فرکانس های پایین را کوچک است (که در مرکز تصویر قرار دارند) و فرکانس های بالا را ضرایب بزرگتری دارند (که از مرکز تصویر فاصله دارند).

میدانیم که فرکانس های بالا در حوزه فرکانس، لبه ها در حوزه مکان را نشان می دهند. پس این فیلتر در حوزه فرکانس، فرکانس های بالا را تقویت می کند، که یعنی لبه ها را در حوزه مکان تقویت می کند. با این کار لبه های تصویر تقویت شده و بقیه تصویر سیاه می شود.

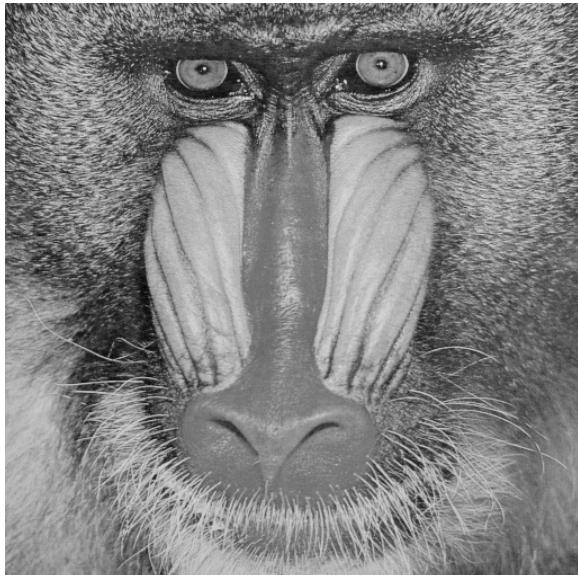
این فیلتر، عملکردی برخلاف فیلتر a دارد. با مشاهده شکل این دو فیلتر در حوزه فرکانس، این موضوع مشخص خواهد شد.

فیلتر a که لبه ها را تضعیف می کند و فیلتر b لبه را تقویت می کند.

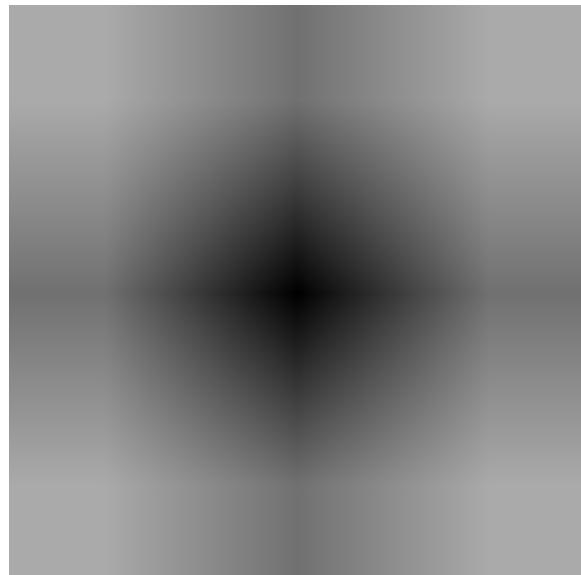
شکل زیر نتیجه اعمال این فیلتر برای عکس اصلی در حوزه فرکانس است.

#### قسمت ۴.۱.۲:

در سوال از ما خواسته سه تصویر زیر را به حوزه فرکانس ببریم و تاثیر عملیات لگاریتم گیری و شیفت را بررسی کنیم. سه تصویر به شکل زیر هستند.



شکل ۲.۱۳: تصویر اصلی baboon



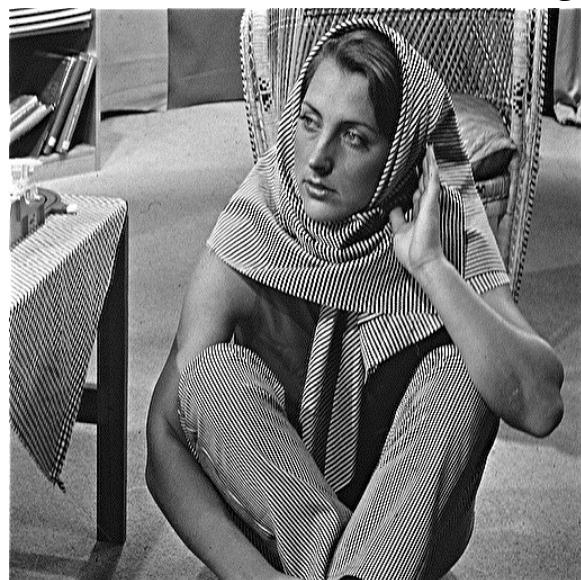
شکل ۲.۱۱: فیلتر C در حوزه فرکانس

همانطور که در شکل ۲.۱۱ مشاهده می شود، این فیلتر در فرکانس های پایین (که در مرکز تصویر قرار دارد) ضرایب کمتری دارد و هر چه از مرکز فاصله می گیریم و به سمت فرکانس های بالاتر می رویم، ضرایب اختصاص داده شده بیشتر خواهد شد.

شکل زیر خروجی را پس از اعمال فیلتر بر تصویر اصلی نشان می دهد:



شکل ۲.۱۴: تصویر اصلی ۱۶F



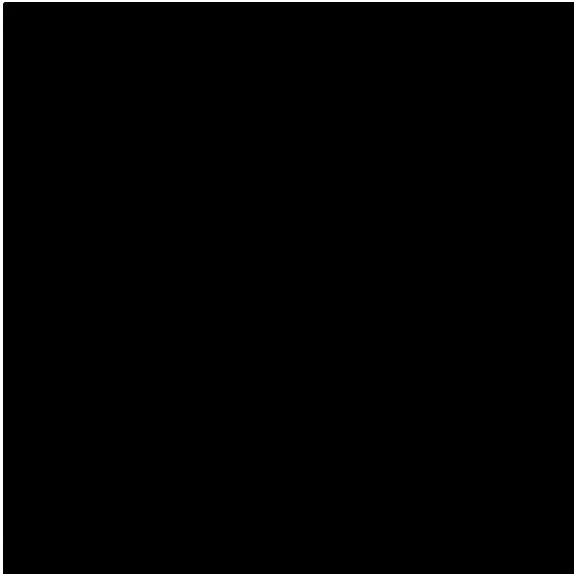
شکل ۲.۱۲: اعمال فیلتر C بر تصویر اصلی

همانطور که در تصویر مشاهده می شود، لبه های تصویر تقویت شده است در حالی که بقیه تصویر ثابت مانده و به ان آسیبی نرسیده است.

به علت اینکه پس زمینه سفید و کوچکی پیکسل سفید، این پیکسل به صورت واضح دیده نمیشود. به همین علت برای این تصویر(baboon) زوم شده‌ی آن را نیز در پایین مشاهده می‌کند(برای بقیه تصاویر نیز مشابه همین خواهد بود).



شکل ۲.۱۷: تصویر زوم شده‌ی آنرا در حوزه فرکانس بدون لگاریتم و شیفت



شکل ۲.۱۸: تصویر ۱۶F در حوزه فرکانس بدون لگاریتم و شیفت



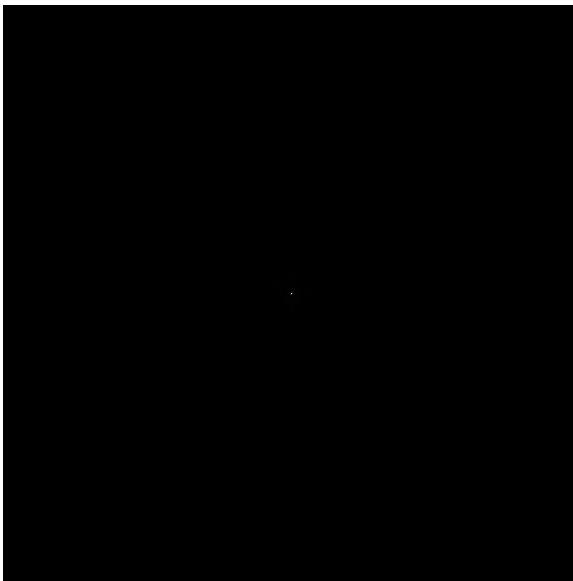
شکل ۲.۱۵: تصویر اصلی Lena

حال سه تصویر را در حوزه فرکانس و ویژگی‌های زیر نشان می‌دهیم:

۱) بدون لگاریتم گیری و بدون شیفت:  
در قسمت شرح توضیح دادیم که در صورتی که لگاریتم گرفته نشود، به علت مقدار زیاد در مبدأ فرکانسی، پس از نرمال کردن بقیه تصویر مقداری بسیار نزدیک به هم و نزدیک به صفر خواهند گرفت. و فقط یک نقطه مبدأ سفید خواهد بود، با توجه با اینکه در این قسمت شیفت هم نداده ایم، مبدأ در گوشه سمت چپ و بالا قرار دارد.



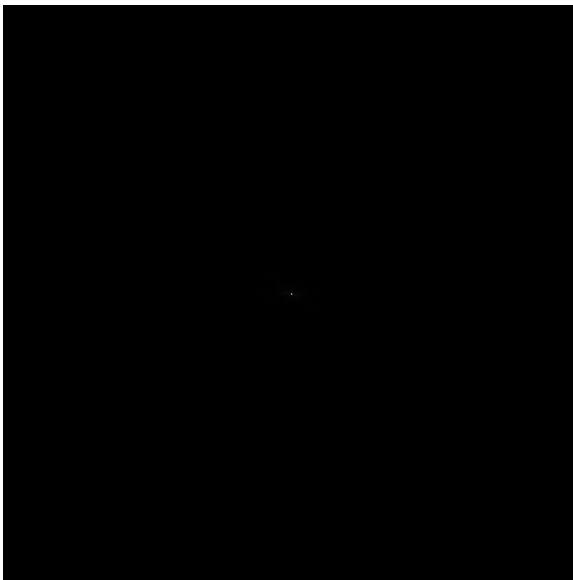
شکل ۲.۱۶: تصویر baboon در حوزه فرکانس بدون لگاریتم و شیفت



شکل ۲.۲۱: تصویر 16F در حوزه فرکانس بدون لگاریتم و با شیفت



شکل ۲.۱۹: تصویر Lena در حوزه فرکانس بدون لگاریتم و با شیفت



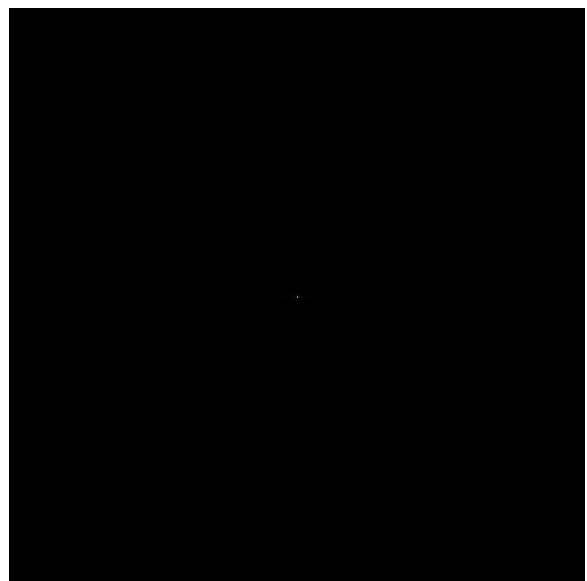
شکل ۲.۲۲: تصویر Lena در حوزه فرکانس بدون لگاریتم و با شیفت

اگر روی تصاویر زوم شود، نقطه سفیدی در همه تصاویر در مرکز وجود دارد.

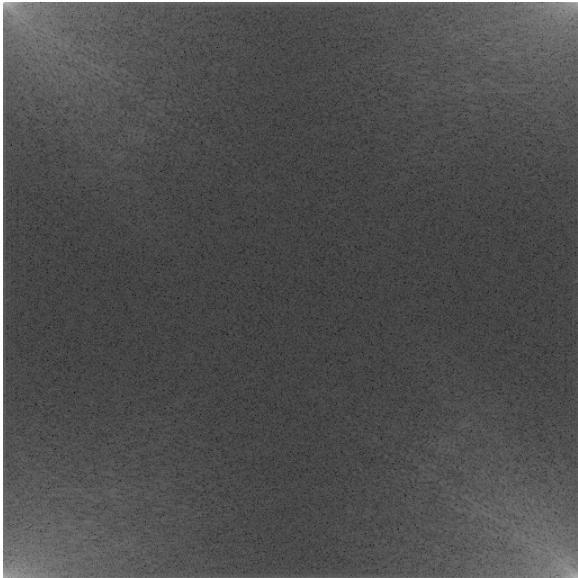
مشاهد کردیم که بدون لگاریتم گیری، تصویر در حوزه فرکانس اطلاعات خاصی در اختیار ما قرار نمی‌دهد، حال با لگاریتم بررسی می‌کیم.

(۲) با لگاریتم گیری و بدون شیفت:

به علت استفاده از لگاریتم، اختلاف به شدت کاهش یافته و ضرایب در حوزه فرکانس در این قسمت واضح هستند و به

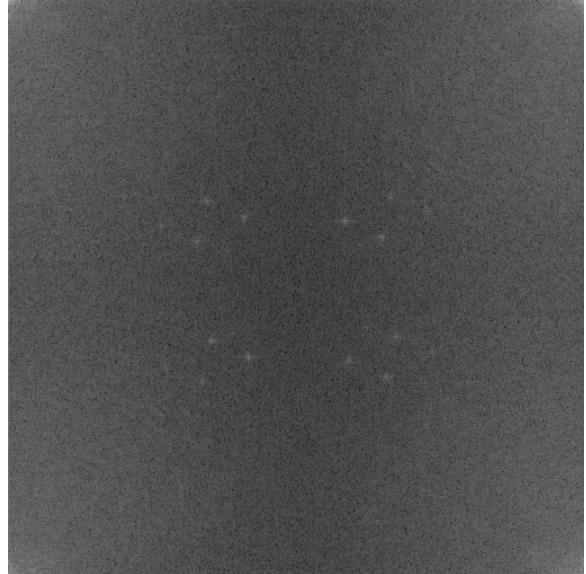


شکل ۲.۲۰: تصویر Baboon در حوزه فرکانس بدون لگاریتم و با شیفت



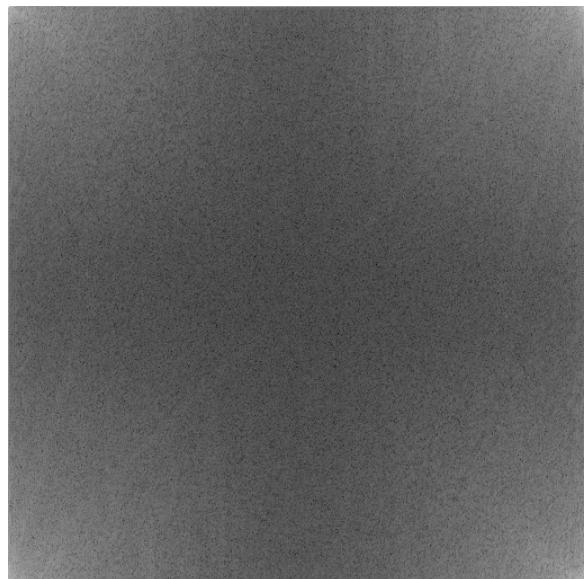
شکل ۲.۲۵: تصویر Lena در حوزه فرکانس با لگاریتم و بدون شیفت

خوبی دیده می شوند. فقط در این حالت مبدا فرکانسی در گوشه چپ بالا تصاویر قرار دارد.

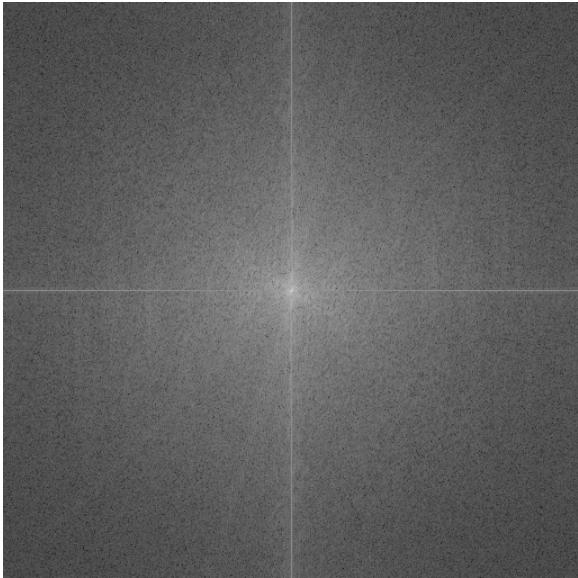


شکل ۲.۲۳: تصویر Baboon در حوزه فرکانس با لگاریتم و بدون شیفت

۴) با لگاریتم و با شیفت:  
به علت استفاده از لگاریتم، ضرایب فرکانسی به خوبی دیده می شوند، به علت استفاده از شیفت نیز مرکز فرکانس به مبدا منتقال پیدا کرده است.  
مشاهده می شود که در این حالت ویژگی های عکس به شکل بارزی دیده می شود. به طور مثال وجود لبه قطری در عکس Lena کاملاً واضح است، در حالیکه در سه حالت قبل چنین چیزی از روی عکس مشخص نبود. بنابراین این حالت بهترین حالت برای نمایش عکس در حوزه فرکانس است.

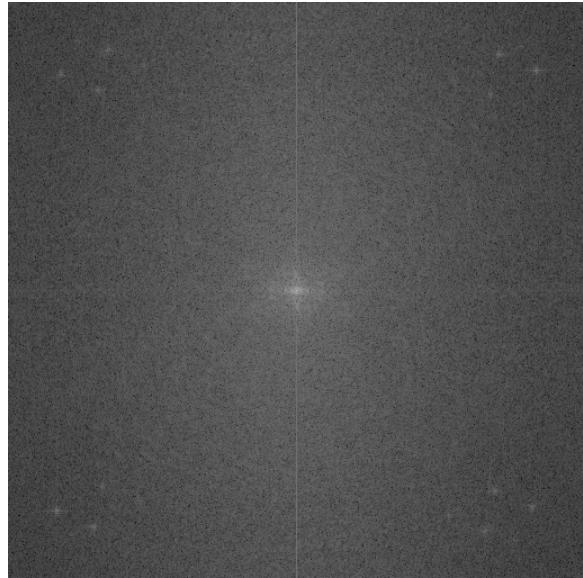


شکل ۲.۲۴: تصویر 16F در حوزه فرکانس با لگاریتم و بدون شیفت



شکل ۲.۲۷: تصویر 16F در حوزه فرکانس با لگاریتم و با شیفت

در 16F با اینکه لبه‌های افقی، عمودی و مورب نسبتاً زیادی در تصویر وجود دارد اما بیشتر قسمت‌های تصویر از جمله مناطق برفی، یکنواخت هستند. به همین دلیل در طیف آن با جود اینکه خطاهای افقی، عمودی و مورب در تصویر وجود دارد، می‌بینیم که ضرایب بزرگ‌تر در مرکز تجمع کرده‌اند.



شکل ۲.۲۶: تصویر Baboon در حوزه فرکانس با لگاریتم و با شیفت

اگر به تصویر Baboon دقیق‌تر بینیم که با وجود اینکه در نگاه اول چند لبه عمودی واضح در آن دیده می‌شود، اما تعداد لبه‌های افقی در آن به مراتب بیشتر است. در واقع در اکثر نواحی تصویر از جمله دور چشمها و صورت حیوان لبه‌های افقی کوچک بسیاری دیده می‌شود و میتوان گفت که تصویر پر از لبه‌ی افقی است. از آنجایی که جهت تغییرات در حوزه فرکانس عمود بر جهت آن در حوزه مکان است، در طیف این تصویر به صورت خطوط عمودی دیده می‌شود.

که نشان دهنده لبه‌های افقی است. به طور کلی baboon یک تصویر نسبتاً غیر یکنواخت است و به همین دلیل همانطور که می‌بینید، ضرایب مرکز خیلی بزرگ نیست و همان طور که گفته شده ضرایب بزرگ‌تر مربوط به فرکانسها در جهت عمود می‌باشد. همچنان در این تصویر چندین لبه‌ی مورب وجود دارد که خود را به شکل نقاطی سفید در فواصل دورتر از مرکز (فرکانس‌های بالا) و در جهت مورب خود را در طیف این تصویر نشان داده‌اند.

### قسمت a:



شکل ۲.۲۹: تصویر پس از اعمال تابع a در حوزه مکان با

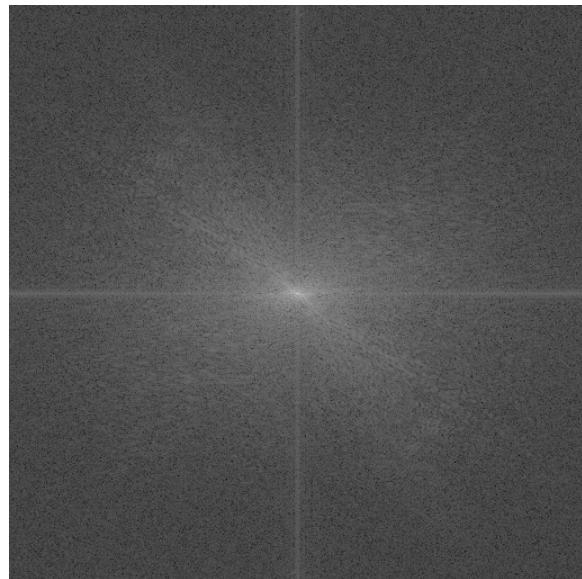
$$T = \frac{1}{4}$$



شکل ۲.۳۰: تصویر پس از اعمال تابع a در حوزه مکان با

$$T = \frac{1}{8}$$

با توجه به توضیحات بیان شده در بخش شرح ، این فیلتر بسیار شبیه فیلتر low pass عمل می کند با این تفاوت که فرکانس‌های بزرگ درجهت عمودی و افقی را از بین نمیرد و فقط ۴ گوشه طیف را سیاه می‌کند و از آنجایی که ضرایب فرکانسی در این نقاط بیشتر است پس از اعمال آن تصویر چندان تغییر نمی‌کند البته در حالت  $T = \frac{1}{8}$  تصویرکمی smooth شده است.



شکل ۲.۲۸: تصویر Lena در حوزه فرکانس با لگاریتم و با شبیت

تصویر Lena یک تصویر نیمه یکتوخت است یعنی لبه‌های زیادی دارد اما بسیاری از قسمت‌های آن نیز یکتوخت و بدون لبه می‌باشد. همان طور که می‌بینید در طیف آن خط‌های افقی و عمودی نشان دهنده‌ی لبه‌های عمودی و افقی هستند و بک خط مورب نیز در جهت خلاف لبه‌های مورب تصویر وجود دارد (بیشتر لبه‌های مورب در تصویر Lena در جهتی هستند که با افق زاویه ۴۵ درجه می‌سازند اما در طیف آن در خلاف جهت بوده و با افق زاویه ۴۵- درجه می‌سازند). در واقع در تصویر Lena ضرایب بزرگ مربوط به فرکانس‌های افقی ، عمودی ، مورب در جهت گفته شده و فرکانس‌های مرکزی که همان فرکانس‌های کوچک هستند ، می‌باشد.

### قسمت ۴.۲.۲:

برای درک بهتر در قسمت شرح نتیجه اعمال تابع ها رو بر حوزه فرکانس نمایش دادیم، در این قسمت نتیجه در حوزه مکان نشان می دهیم.

: قسمت b



شکل ۲.۳۳: تصویر پس از اعمال قسمت a در حوزه مکان با  $T = \frac{1}{4}$  و بدون صفر کردن مرکز فرکانسی



شکل ۲.۳۱: تصویر پس از اعمال قسمت a در حوزه مکان با  $T = \frac{1}{4}$



شکل ۲.۳۴: تصویر پس از اعمال قسمت a در حوزه مکان با  $T = \frac{1}{8}$  و بدون صفر کردن مرکز فرکانسی



شکل ۲.۳۲: تصویر پس از اعمال قسمت a در حوزه مکان با  $T = \frac{1}{8}$

همانطور که گفته شد چون در این قسمت مقدار نقطه، [۰,۰] که مبدا فرکانسی است و شامل بیشتر اطلاعات تصویر است نیز صفر میشود، بخشایی از تصویر از بین رفته است. برای مقایسه این قسمت با ۳ قسمت بعد این فیلتر را طوری تغییر میدهیم که مقدار نقطه‌ی [۰,۰] تغییر نکند.



شکل ۲.۳۷: تصویر پس از اعمال قسمت iii در حوزه مکان

$$T = \frac{1}{4}$$



شکل ۲.۳۵: تصویر پس از اعمال قسمت ii در حوزه مکان با

$$T = \frac{1}{4}$$



شکل ۲.۳۸: تصویر پس از اعمال قسمت iii در حوزه مکان

$$T = \frac{1}{8}$$



شکل ۲.۳۶: تصویر پس از اعمال قسمت ii در حوزه مکان با

$$T = \frac{1}{8}$$



شکل ۲.۴۱: تصویر پس از اعمال چهار قسمت در حوزه مکان با  $T = \frac{1}{4}$



شکل ۲.۴۲: تصویر پس از اعمال چهار قسمت در حوزه مکان با  $T = \frac{1}{8}$



شکل ۲.۳۹: تصویر پس از اعمال قسمت ۷ در حوزه مکان با  $T = \frac{1}{4}$



شکل ۲.۴۰: تصویر پس از اعمال قسمت ۷ در حوزه مکان با  $T = \frac{1}{8}$

همانطور که مشاهده می شود همه این فیلتر تاثیر محسوسی بر روی عکس میگذارند و عکس به شکل قابل ملاحظه ای تغییر می کند اما هیچ کدام از این فیلتر ها عملکرد خاصی مانند شارپ کردن یا تشخیص لبه ندارند.

اما نتیجه اعمال ۴ تای آنها همانطور که در شرح توضیح داده شد، مانند edge detection عمل می کند.

```

    return final_img

out_path = "result/4.1.1/"
barbara_gray =
cv.imread('image/Barbara.bmp',
cv.IMREAD_GRAYSCALE)
cv.imwrite(out_path +
'barbara_gray.jpg', barbara_gray)

# a filter
a = np.array([[1, 2, 1], [2, 4, 2], [1,
2, 1]])
a = a / 16

# separate a
a_separate_c = np.array([1, 2, 1])
a_separate_c = a_separate_c / 4
a_separate_r = np.array([[1, 2, 1]])
a_separate_r = a_separate_r / 4

# b filter
b = np.array([-1, -1, -1], [-1, 8, -
1],[-1, -1, -1])

# c filter
c = np.array([[0, -1, 0], [-1, 5, -1],
[0, -1, 0]])

# compute FT of filters
a_f = np.fft.fft2(a)
b_f = np.fft.fft2(b)
c_f = np.fft.fft2(c)
a_separate_c_f =
np.fft.fft(a_separate_c)
a_separate_r_f =
np.fft.fft(a_separate_r)

# shift filters
a_shift = np.fft.fftshift(a_f)
b_shift = np.fft.fftshift(b_f)
c_shift = np.fft.fftshift(c_f)
a_separate_c_shift =
np.abs(np.fft.fftshift(a_separate_c_f))
a_separate_r_shift =
np.abs(np.fft.fftshift(a_separate_r_f))

# compute logarithm of filters

```

## APPENDIX:

:٤.١.١ قسمت

```

import cv as cv
import numpy as np

def normalize(img):
    minimum = np.amin(img)
    return (img - minimum) * (255 /
np.amax(img))

def filtering(this_img, this_filter):
    img = np.zeros(this_img.shape)
    for i in range(this_img.shape[0]):
        img[i] = this_img[i].copy()

    # compute p and q
    height, width = img.shape
    new_height, new_width = height * 2,
width * 2

    # create fp
    new_img = np.zeros((new_height,
new_width))
    for i in range(height):
        for j in range(width):
            new_img[i][j] = img[i][j]

    img_fft = np.fft.fft2(new_img)
    # DFT and shift
    img_fft = np.fft.fftshift(img_fft)
    # multiply filter
    img_filtering = img_fft *
cv.resize(np.abs(this_filter),
(new_height, new_width))
    # shift
    img_filtering =
np.fft.ifftshift(img_filtering)
    # IDFT and real
    img_ifft =
np.real(np.fft.ifft2(img_filtering))
    # resize image
    final_img = np.zeros((height,
width))
    for i in range(height):
        for j in range(width):
            final_img[i][j] =
img_ifft[i][j]

```

```

final_img_b = filtering(barbara_gray,
b_shift)
cv.imwrite(out_path + 'barbara_b.jpg',
final_img_b)

# apply c filter
final_img_c = filtering(barbara_gray,
c_shift)
cv.imwrite(out_path + 'barbara_c.jpg',
final_img_c)

# apply a separate filters
img_a_separate_c =
filtering(barbara_gray,
a_separate_c_shift)
img_a_separate_r =
filtering(barbara_gray,
a_separate_r_shift)
cv.imwrite(out_path +
'barbara_a_separate_c.jpg',
img_a_separate_c)
cv.imwrite(out_path +
'barbara_a_separate_r.jpg',
img_a_separate_r)

final_img_for_seperate_filter =
filtering(img_a_separate_c,
a_separate_r_shift)
cv.imwrite(out_path +
'barbara_a_separate_final.jpg',
final_img_for_seperate_filter)

```

```

a_log = np.log(np.abs(a_shift) + 1)
b_log = np.log(np.abs(b_shift) + 1)
c_log = np.log(np.abs(c_shift) + 1)
a_separate_c_log =
np.log(a_separate_c_shift + 1)
a_separate_r_log =
np.log(a_separate_r_shift + 1)

# normalize filters
a_log_normalize = normalize(a_log)
b_log_normalize = normalize(b_log)
c_log_normalize = normalize(c_log)
a_separate_c_log_normalize =
normalize(a_separate_c_log)
a_separate_r_log_normalize =
normalize(a_separate_r_log)

# resize filters to show
a_separate_c_log_f =
cv.resize(a_separate_c_log_normalize,
(512, 512))
a_separate_r_log_f =
cv.resize(a_separate_r_log_normalize,
(512, 512))
a_log = cv.resize(a_log_normalize,
(512, 512))
b_log = cv.resize(b_log_normalize,
(512, 512))
c_log = cv.resize(c_log_normalize,
(512, 512))

# write filters
cv.imwrite(out_path + 'a.jpg', a_log)
cv.imwrite(out_path + 'b.jpg', b_log)
cv.imwrite(out_path + 'c.jpg', c_log)
cv.imwrite(out_path +
'a_separate_c.jpg', a_separate_c_log_f)
cv.imwrite(out_path +
'a_separate_r.jpg', a_separate_r_log_f)

# apply a filter
final_img_a = filtering(barbara_gray,
a_shift)
cv.imwrite(out_path + 'barbara_a.jpg',
final_img_a)

# apply b filter

```

#### قسمت ۴.۱.۲:

```

import cv as cv
import numpy as np
def normalize(img):
    minimum = np.amin(img)
    return (img - minimum) * (255 /
np.amax(img))

out_path = "result/4.1.2/"
baboon = cv.imread('image/Baboon.bmp',
cv.IMREAD_GRAYSCALE)

```

```

cv.imwrite(out_path +
'f16_f_shift.jpg',
normalize(f16_f_shift))
cv.imwrite(out_path +
'lena_f_shift.jpg',
normalize(lena_f_shift))

# no shift and log
cv.imwrite(out_path +
'baboon_f_log.jpg',
normalize(baboon_f_log))
cv.imwrite(out_path + 'f16_f_log.jpg',
normalize(f16_f_log))
cv.imwrite(out_path + 'lena_f_log.jpg',
normalize(lena_f_log))

# shift and log
cv.imwrite(out_path +
'baboon_f_shift_log.jpg',
normalize(baboon_f_shift_log))
cv.imwrite(out_path +
'f16_f_shift_log.jpg',
normalize(f16_f_shift_log))
cv.imwrite(out_path +
'lena_f_shift_log.jpg',
normalize(lena_f_shift_log))

# read images
baboon_f = cv.imread(out_path +
'baboon_f.jpg', cv.IMREAD_GRAYSCALE)
f16_f = cv.imread(out_path +
'f16_f.jpg', cv.IMREAD_GRAYSCALE)
lena_f = cv.imread(out_path +
'lena_f.jpg', cv.IMREAD_GRAYSCALE)

baboon_f_shift = cv.imread(out_path +
'baboon_f_shift.jpg', 0)
f16_f_shift = cv.imread(out_path +
'f16_f_shift.jpg', 0)
lena_f_shift = cv.imread(out_path +
'lena_f_shift.jpg', 0)
baboon_f_log = cv.imread(out_path +
'baboon_f_log.jpg', 0)
f16_f_log = cv.imread(out_path +
'f16_f_log.jpg', 0)
lena_f_log = cv.imread(out_path +
'lena_f_log.jpg', 0)

```

```

f16 = cv.imread('image/F16.bmp', 0)
lena = cv.imread('image/Lena.bmp', 0)

cv.imwrite(out_path +
'baboon_gray.jpg', baboon)
cv.imwrite(out_path + 'f16_gray.jpg',
f16)
cv.imwrite(out_path + 'lena_gray.jpg',
lena)

# compute DFT of images
baboon_f = np.abs(np.fft.fft2(baboon))
f16_f = np.abs(np.fft.fft2(f16))
lena_f = np.abs(np.fft.fft2(lena))

# shift
baboon_f_shift =
np.fft.fftshift(baboon_f)
f16_f_shift = np.fft.fftshift(f16_f)
lena_f_shift = np.fft.fftshift(lena_f)

# log on no shifted image
baboon_f_log = np.log(baboon_f + 1)
f16_f_log = np.log(f16_f + 1)
lena_f_log = np.log(lena_f + 1)

# log on shifted image
baboon_f_shift_log =
np.log(baboon_f_shift + 1)
f16_f_shift_log = np.log(f16_f_shift + 1)
lena_f_shift_log = np.log(lena_f_shift + 1)

# write images
# no shift and no log

cv.imwrite(out_path + 'baboon_f.jpg',
normalize(baboon_f))
cv.imwrite(out_path + 'f16_f.jpg',
normalize(f16_f))
cv.imwrite(out_path + 'lena_f.jpg',
normalize(lena_f))

# shift and no log
cv.imwrite(out_path +
'baboon_f_shift.jpg',
normalize(baboon_f_shift))

```

```

        if (maximum >= i >= 0) and
(maximum >= j >= 0) and (i != 0 or j != 0):
            img_copy[i][j] = 0
    return img_copy

def b2(img_fft, t):
    img_copy = img_fft.copy()
    height, width = img_fft.shape
    maximum = t * height
    minimum = (1 - t) * height
    for i in range(height):
        for j in range(width):
            if (maximum >= i >= 0) and
(height - 1 >= j >= minimum):
                img_copy[i][j] = 0
    return img_copy

def b3(img_fft, t):
    img_copy = img_fft.copy()
    height, width = img_fft.shape
    maximum = t * height
    minimum = (1 - t) * height
    for i in range(height):
        for j in range(width):
            if (height - 1 >= i >=
minimum) and (maximum >= j >= 0):
                img_copy[i][j] = 0
    return img_copy

def b4(img_fft, t):
    img_copy = img_fft.copy()
    height, width = img_fft.shape
    minimum = (1 - t) * height
    for i in range(height):
        for j in range(width):
            if (height - 1 >= i >=
minimum) and (height - 1 >= j >=
minimum):
                img_copy[i][j] = 0
    return img_copy

def inverse_transform(img, phase,
name):
    img_i = np.real(np.fft.ifft2(img *
math.e ** (1j * phase)))

```

```

baboon_f_shift_log = cv.imread(out_path +
'baboon_f_shift_log.jpg', 0)
f16_f_shift_log = cv.imread(out_path +
'f16_f_shift_log.jpg', 0)
lena_f_shift_log = cv.imread(out_path +
'lena_f_shift_log.jpg', 0)

```

## قسمت ۴.۲.۲

```

import cv2 as cv
import numpy as np
import math
def normalize(img):
    minimum = np.amin(img)
    return (img - minimum) * (255
/np.amax(img))

def a(img_fft, t):
    img_copy = img_fft.copy()
    height, width = img_fft.shape
    minimum = (1 - t) * height
    maximum = t * height
    for i in range(height):
        for j in range(width):
            if (minimum > i > maximum)
and (minimum > j > maximum):
                img_copy[i][j] = 0
    return img_copy

def b1(img_fft, t):
    img_copy = img_fft.copy()
    height, width = img_fft.shape
    maximum = t * height
    minimum = (1 - t) * height
    for i in range(height):
        for j in range(width):
            if (maximum >= i >= 0) and
(maximum >= j >= 0):
                img_copy[i][j] = 0
    return img_copy

def b1_no_zero(img_fft, t):
    img_copy = img_fft.copy()
    height, width = img_fft.shape
    maximum = t * height
    for i in range(height):
        for j in range(width):

```

```

lena_i_b1_4 =
inverse_transform(lena_b1_4, phase,
'lena_b1_4.jpg')
cv.imwrite(out_path +
'b1_4_fourier.jpg',
normalize(np.log(lena_b1_4 + 1)))
cv.imwrite(out_path +
'b1_4_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b1_4) + 1)))

# b , i , T = 1/8
lena_b1_8 = b1(magnitude, 0.125)
lena_i_b1_8 =
inverse_transform(lena_b1_8, phase,
'lena_b1_8.jpg')
cv.imwrite(out_path +
'b1_8_fourier.jpg',
normalize(np.log(lena_b1_8 + 1)))
cv.imwrite(out_path +
'b1_8_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b1_8) + 1)))

# b , i , not considering [0, 0]
lena_b1_4_no_zero =
b1_no_zero(magnitude, 0.25)
lena_i_b1_4_no_zero =
inverse_transform(lena_b1_4_no_zero,
phase, 'lena_b1_4_no_zero.jpg')

# b , i , not considering [0, 0]
lena_b1_8_no_zero =
b1_no_zero(magnitude, 0.125)
lena_i_b1_8_no_zero =
inverse_transform(lena_b1_8_no_zero,
phase, 'lena_b1_8_no_zero.jpg')

# b , ii , T = 1/4

lena_b2_4 = b2(magnitude, 0.25)
lena_i_b2_4 =
inverse_transform(lena_b2_4, phase,
'lena_b2_4.jpg')
cv.imwrite(out_path +
'b2_4_fourier.jpg',
normalize(np.log(lena_b2_4 + 1)))

```

```

cv.imwrite("result/4.2.2/" + name,
normalize(img_i))
img_final = cv.imread(name, 0)
return img_final

out_path = "result/4.2.2/"
lena_gray = cv.imread('image/Lena.bmp',
0)
cv.imwrite(out_path + 'lena_gray.jpg',
lena_gray)
lena_fft = np.fft.fft2(lena_gray)
magnitude = np.abs(lena_fft)
phase = np.angle(lena_fft)
# cv.imshow('m', normalize(magnitude))

...
# a , T = 1/4
lena_a_4 = a(magnitude, 0.25)
lena_i_a_4 =
inverse_transform(lena_a_4, phase,
'lena_a_4.jpg')

# write magnitude after apply change
cv.imwrite(out_path +
'a_4_fourier.jpg',
normalize(np.log(lena_a_4 + 1)))
cv.imwrite(out_path +
'a_4_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
a_4) + 1)))
# a , T = 1/8
lena_a_8 = a(magnitude, 0.125)
lena_i_a_8 =
inverse_transform(lena_a_8, phase,
'lena_a_8.jpg')
cv.imwrite(out_path +
'a_8_fourier.jpg',
normalize(np.log(lena_a_8 + 1)))

cv.imwrite(out_path +
'a_8_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
a_8) + 1)))

# b , i , T = 1/4
lena_b1_4 = b1(magnitude, 0.25)

```

```

lena_i_b4_4 =
inverse_transform(lena_b4_4, phase,
'lena_b4_4.jpg')
cv.imwrite(out_path +
'b4_4_fourier.jpg',
normalize(np.log(lena_b4_4 + 1)))
cv.imwrite(out_path +
'b4_4_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b4_4) + 1)))

# b , iv , T = 1/8
lena_b4_8 = b4(magnitude, 0.125)
lena_i_b4_8 =
inverse_transform(lena_b4_8, phase,
'lena_b4_8.jpg')
cv.imwrite(out_path +
'b4_8_fourier.jpg',
normalize(np.log(lena_b4_8 + 1)))
cv.imwrite(out_path +
'b4_8_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b4_8) + 1)))
'''

# b , all , T = 1/4
lena_all_4 = b4(b3(b2(b1(magnitude,
0.25) , 0.25) , 0.25) , 0.25)
lena_all_4_s =
inverse_transform(lena_all_4, phase,
'lena_b_all_4.jpg')
cv.imwrite(out_path + 'b_all_4.jpg',
normalize(np.log(lena_all_4 + 1)))
cv.imwrite(out_path +
'b_all_4_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
all_4) + 1)))

# b , all , T = 1/8
lena_all_8 =
b4(b3(b2(b1_no_zero(magnitude, 0.125) ,
0.125) , 0.125) , 0.125)
lena_all_8_s =
inverse_transform(lena_all_8, phase,
'lena_b_all_8.jpg')
cv.imwrite(out_path + 'b_all_8.jpg',
normalize(np.log(lena_all_8 + 1)))

```

```

cv.imwrite(out_path +
'b2_4_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b2_4) + 1)))

# b , ii , T = 1/8
lena_b2_8 = b2(magnitude, 0.125)
lena_i_b2_8 =
inverse_transform(lena_b2_8, phase,
'lena_b2_8.jpg')
cv.imwrite(out_path +
'b2_8_fourier.jpg',
normalize(np.log(lena_b2_8 + 1)))
cv.imwrite(out_path +
'b2_8_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b2_8) + 1)))

# b , iii , T = 1/4
lena_b3_4 = b3(magnitude, 0.25)
lena_i_b3_4 =
inverse_transform(lena_b3_4,
phase,'lena_b3_4.jpg')
cv.imwrite(out_path +
'b3_4_fourier.jpg',
normalize(np.log(lena_b3_4 + 1)))
cv.imwrite(out_path +
'b3_4_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b3_4) + 1)))

# b , iii , T = 1/8
lena_b3_8 = b3(magnitude, 0.125)
lena_i_b3_8 =
inverse_transform(lena_b3_8, phase,
'lena_b3_8.jpg')
cv.imwrite(out_path +
'b3_8_fourier.jpg',
normalize(np.log(lena_b3_8 + 1)))
cv.imwrite(out_path +
'b3_8_fourier_shift.jpg',
normalize(np.log(np.fft.ifftshift(lena_
b3_8) + 1)))

# b , iv , T = 1/4
lena_b4_4 = b4(magnitude, 0.25)

```

```
cv.imwrite(out_path +  
          'b_all_8_shift.jpg',  
          normalize(np.log(np.fft.ifftshift(lena_  
          all_8) + 1)))
```