تمرین چهارم

آریا ابراهیمی

چکیده

در این سری تمرین، با حوزه فرکانس آشنا شده و به بررسی فیلتر ها و عملیات فیلترینگ و کانولوشن در این حوزه میپردازیم. همچنین طیف عکس های مختلف را نیز در این تمرین به دست آورده و به بررسی آن‌ها می‌پردازیم.
در تمرین اول سه فیلتر داده شده را با استفاده از تبدیل فوریه به حوزه فرکانس برده و طیف و تاثیر آنها را بررسی میکنیم. در تمرین دوم نیز تاثیر تغییر طیف در حوزه فرکانس را با اعمال دو تابع به طیف بررسی کرده و نتایج آنرا مشاهده میکنیم.

۱- شرح تکنیکال

۱.۱) تمرین ٤.۱.۱

در این تمرین خواسته شده است تا به بررسی سه فیلتر داده شده در حوزه فرکانس پرداخته شده و تحلیلی بر عملکرد آنها انجام شود.

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

فیلتر *seprable* طبق تعریف، فیلتری است که از ضرب ماتریسی دو فیلتر دیگر تشکیل شده باشد. برای فیلتر a میتوان مشاهده کرد که از ترکیب دو فیلتر *smoothing* با وزن بیشتر در وسط تشکیل شده است.

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

مشاهده می‌شود در فیلتر های *smooth* کننده پایه در فیلتر a، وزن در وسط بردار بیشتر است که به این دلیل است تا در میانگین گیری، وزن بیشتری به پیکسل مرکزی دهد تا در نتیجه تاثیر بیشتری داشته باشد.

برای اعمال کردن فیلتر بر روی تصویر، ابتدا باید با استفاده از *zero padding* سایز فیلتر و عکس را به دوبرابر سایز عکس برسانیم که در کتاب این سایز ها را P و Q نامیده است. در ادامه تبدیل فوریه را با استفاده از تبدیل فوریه پیاده سازی شده در کتابخانه *numpy* روی عکس اصلی و فیلتر انجام داده و در نهایت با استفاده از *fftshift* ، فرکانس های پایین را به وسط شیفت میدهیم. برای *convolve* کردن در حوزه فرکانس کافی است تا نتایج به دست آمده از تبدیل فوریه عکس و فیلتر را به صورت *elementwise* در یکدیگر ضرب کنیم. (ضرب در حوزه فرکانس و تبدیل، برابر با *convolution* در حوزه مکان است) در نهایت برای بازگشت به حوزه مکان، معکوس شیفت یا همان *ifftshift* در کتابخانه *numpy* را انجام داده و سپس با استفاده از معکوس تبدیل فوریه (*ifft*) به حوزه مکان باز می‌گردیم.

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

فیلتر b یک فیلتر لاپلاسین است که برای پیدا کردن لبه ها استفاده می‌شود. این فیلتر *seprable* نیست برای همین به صورت مستقیم تبدیل فوریه را بر روی آن انجام میدهیم. همچنین برای *convolve* کردن نیز همانند روش مطرح شده در قسمت قبل عمل میکنیم.

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

فیلتر c یک فیلتر شارپ کننده میباشد که ترکیبی از لاپلاسین و یک ماتریس است که تنها در درایه وسط آن ۱ داریم و بقیه درایه ها ۰ می‌باشند. این ترکیب باعث شارپ شدن لبه ها میشود. این فیلتر نیز همانند فیلتر b ، جداپذیر نیست. برای اعمال کردن آن نیز همانند فیلتر های قبلی عمل میکنیم و همان مراحل را طی می کنیم.

۱.۲) تمرین ٤.١.٢

در این تمرین خواسته شده است تا تبدیل فوریه را بر روی ۴ عکس داده شده انجام دهیم و طیف ها را در چهار حالت بدون لگاریتم بدون شیفت، بدون لگاریتم با شیفت، با لگاریتم بدون شیفت و با لگاریتم شیفت داده شده بررسی کنیم.
برای این کار کافی است تنها روی خود عکس، تبدیل فوریه انجام دهیم، سپس با استفاده از *fftshift* فرکانس های پایین را به وسط شیفت میدهیم تا بهتر بتوان طیف را بررسی و مشاهده کرد.
برای این کار تابعی طراحی شده است که عکس را دریافت میکند و سپس خروجی تبدیل و شیفت را در دو متغیر جدا ذخیره میکند. در نهایت ۴ خروجی مورد نظر را با استفاده از همین دو متغیر برمی گرداند.
نتایج را در قسمت بررسی نتایج مورد بحث قرار میدهیم.

۱.۳) تمرین ٤.٢.١

در قسمت اول خواسته شده است تا مراحل رسیدن به نتیجه کانولوشن در حوزه فرکانس برای عکس و فیلتر داده شده بررسی شود.
طبق مراحل کتاب گنزالس، این کار را میتوان در ۸ گام انجام داد.

۱. برای عکس ورودی با ابعاد $M \times N$، ابعاد مورد نیاز برای *padding* را طبق روابط $P = 2M$ و $Q = 2N$ به دست می‌آوریم.
۲. با استفاده از ابعاد به دست آمده در گام اول، *padding* روی تصویر انجام میدهیم به این صورت که تصویر در بالا و سمت چپ قرار میگیرد و بقیه درایه ها برابر با صفر خواهند بود. (در صورت استفاده از *zero padding* این مقدار برابر با صفر است. در این تمرین از *zero padding* استفاده شده است)
۳. در این گام در کتاب شیفت دادن انجام شده است ولی در این تمرین ما در این گام تبدیل فوریه را انجام میدهیم و در گام بعد، طیف را شیفت میدهیم که تفاوتی ندارد. با استفاده از *fft2* پیاده سازی شده در *numpy* تبدیل فوریه را روی عکس انجام می‌دهیم.
۴. همانطور که گفته شد در این گام با استفاده از تابع *fftshift* پیاده سازی شده در *numpy* عملیات شیفت دادن را انجام میدهیم تا فرکانس های پایین در مرکز قرار بگیرند
۵. در این گام در کتاب، ایجاد و آماده کردن فیلتر انجام شده است. بدین منظور در پیاده سازی، گام های ۱ تا ۴ را همانند عکس، برای فیلتر نیز انجام میدهیم. فیلتر را با استفاده از ابعاد P و Q به دست آمده از تصویر، *zero pad* کرده و بعد از اعمال تبدیل فوریه، با استفاده از *fftshift* فرکانس های پایین را به وسط شیفت میدهیم.
۶. میدانیم عملیات *convolution* در حوزه مکان، برابر با ضرب *elementwise* در حوزه فرکانس و تبدیل می‌باشد. تنها کافیست نتیجه به دست آمده از فیلتر و تصویر را در یکدیگر ضرب کنیم.
۷. در گام بعد کافیست نتیجه به دست آمده از گام ۶ را با استفاده از *ifftshift* به حالت ابتدایی بدون شیفت برگردانده و سپس *ifft2* یا همان معکوس تبدیل فوریه را روی آن اعمال کرده تا به حوزه مکان بازگردیم. (باید *magnitude* نتیجه بازگشتی را به عنوان نتیجه حوزه مکان در نظر بگیریم. برای این کار از *abs* در پایتون استفاده میکنیم.)
۸. در گام آخر، از آنجایی که تصویر $M \times N$ بوده است و نتیجه بازگشتی ما $P \times Q$ است و تصویر ما در قسمت بالا و سمت چپ این ماتریس تشکیل شده است، ماتریسی $M \times N$ را از بالا و سمت چپ نتیجه جدا می کنیم که نتیجه اعمال فیلتر بر روی تصویر در حوزه فرکانس و بازگشت به حوزه مکان می‌باشد.

برای تصویر داده شده که ابعاد آن 256×256 است، ماتریس پدینگ ماتریسی 512×512 خواهد بود. فیلتر نیز برای اینکه هم سایز با تصویر پد شده باشد، باید تبدیل به ماتریسی 512×512 شود که تنها ۱۱ سطر و ۱۱ ستون ابتدایی آن مقدار خواهند داشت و بقیه درایه ها برابر با ۰ خواهند بود.
بعد از اعمال تبدیل بر روی عکس و فیلتر، کافی است تا عملیات شیفت را انجام دهیم و نتیجه ها را در هم ضرب کنیم. و سپس طبق گام های گفته شده *ifftshift* و *ifft2* را اعمال کرده و به حوزه مکان بازگردیم.

۱.٤) تمرین ٤.٢.٢

در این تمرین خواسته شده است تا بعد از تبدیل فوریه، قسمت هایی از طیف را صفر کنیم و سپس به حوزه فرکانس برگردیم.
دو تابع و دو ضریب از اندازه عکس داده شده است.

تابع a) نمایانگر صفر کردن مربعی در وسط طیف میباشد و ضرایب اندازه این مربع را تعیین میکنند. اگر از ضریب $\frac{1}{4}$ استفاده کنیم، مربع کوچکتری با اندازه ضلع $\frac{1}{2}N$ خواهیم داشت و اگر از ضریب $\frac{1}{8}$ استفاده کنیم اندازه ضلع مربع برابر با $\frac{3}{4}N$ خواهد شد.

تابع b) نمایانگر ۴ مربع در ۴ گوشه طیف است. اگر از ضریب $\frac{1}{4}$ استفاده شود، ضلع مربع ها برابر با $\frac{1}{4}N$ و اگر از ضریب $\frac{1}{8}$ استفاده شود ضلع مربع ها برابر با $\frac{1}{8}N$ خواهد شد.

۲- شرح نتایج

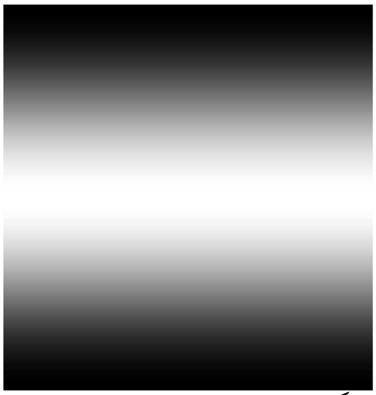
۲.۱) تمرین ٤.١.١

در ابتدا طیف فیلتر های داده شده در حوزه فرکانس را بررسی کرده و سپس نتیجه اعمال آنها را مشاهده و بررسی خواهیم کرد.
فیلتر a) همان طور که در قسمت اول مشاهده کردیم، این فیلتر، فیلتری *seprable* است و میتوان آنرا به صورت حاصل ضرب دو بردار نمایش داد.

$$a_1 = \frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix}$$

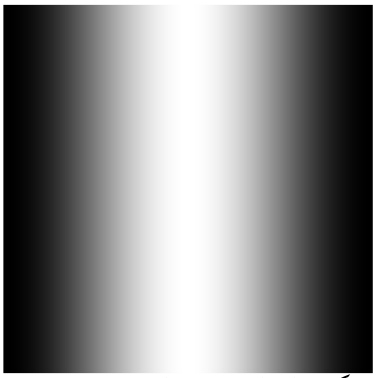
$$a_2 = \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix}$$

بعد از اعمال پدینگ و تبدیل فوریه، به طیف های زیر میرسیم:



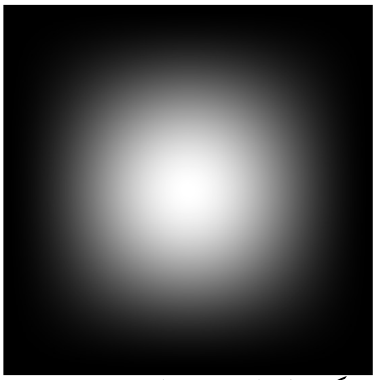
شکل ۱) طیف مربوط به فیلتر a_1 در حوزه فرکانس

همانطور که میتوان مشاهده کرد، این فیلتر که در حوزه مکان به صورت عمودی میانگین گیری میکند و طیف آن در حوزه فرکانس به صورت افقی است.



شکل ۲) طیف مربوط به فیلتر a_2 در حوزه فرکانس

این فیلتر نیز در حوزه مکان به صورت افقی میانگین گیری میکند و در حوزه فرکانس میتوان مشاهده کرد که طیف عمودی دارد.



شکل ۳) طیف مربوط به فیلتر a در حوزه فرکانس

میتوان مشاهده کرد که از ترکیب a_1 و a_2 طیف مربوط به a را میتوان به دست آورد. فیلتر a فیلتری *lowpass* است. میدانیم بعد از شیفت دادن طیف تصویر در حوزه فرکانس، فرکانس های پایین در مرکز قرار میگیرند. میتوان مشاهده کرد که فیلتر a اطراف مرکز را که فرکانس های پایین هستند نگه می‌دارد و فرکانس های بالا که در گوشه ها هستند و نمایانگر لبه ها هستند را حذف میکند. در واقع این فیلتر باعث *smooth* شدن تصویر می‌شود زیرا لبه ها را حذف میکند و کلیات را نگه میدارد.
همان طور که در شکل ۴ مشاهده می‌شود، بعد از اعمال این فیلتر لبه ها کمه *smooth* تر شده اند و از شارپ بودن تصویر کاسته شده است.

شکل ۴) تصویر سمت راست تصویر اصلی *lena* و تصویر سمت چپ نتیجه اعمال فیلتر a می‌باشد.

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

همان‌طور که حدس میزدیم، فیلتر b یک فیلتر لاپلاسین است که فیلتری *highpass* میباشد و فرکانس های پایین را حذف میکند و فرکانس های بالا را نگه میدارد. با حذف فرکانس های پایین، کلیات تصویر را از دست میدهیم و فقط فرکانس های بالا که نشان‌دهنده لبه ها هستند باقی می‌مانند.

شکل ۵) تصویر سمت راست تصویر اصلی *lena* و تصویر سمت چپ نتیجه اعمال فیلتر b می‌باشد.

همان طور که مشاهده میشود، فرکانس های پایین از تصویر حذف شده اند و فقط فرکانس های بالا که همان لبه ها هستند باقی مانده‌اند.

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

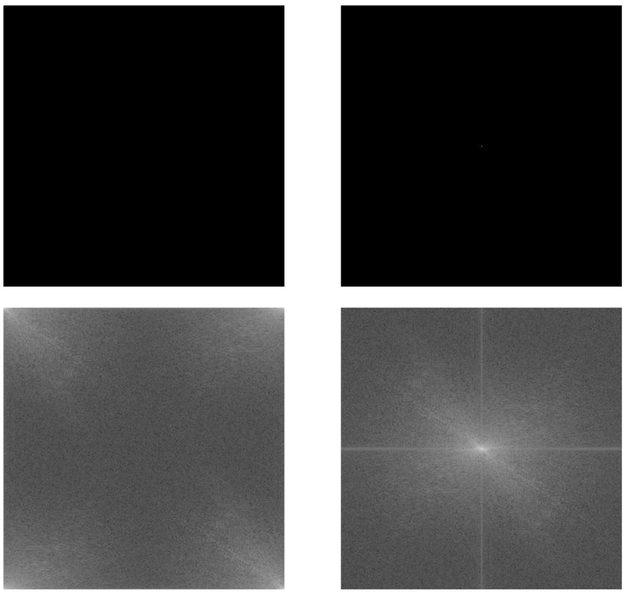
همان طور که گفته شد، فیلتر c که یک فیلتر شارپ کننده لبه است، از ترکیب فیلتر لاپلاسین و یک ماتریس تمام صفر که درایه وسط آن مقدار ۱ دارد تشکیل شده است. این فیلتر باعث میشود تا فرکانس های بالا تقویت شوند و فرکانس های پایین را نیز به طور کلی حذف نمیکند. به این صورت کلیات تصویر را خواهیم داشت و لبه های آن شارپ تر خواهند شد.

شکل ۶) تصویر سمت راست تصویر اصلی *lena* و تصویر سمت چپ نتیجه اعمال فیلتر c می‌باشد.

همان طور که گفته شد، با اعمال فیلتر c، لبه های تصویر شارپ تر شده اند و فرکانس های پایین تصویر که همان کلیات آن هستند نیز باقی مانده‌اند.

۲.۲) تمرین ٤.١.٢

در این تمرین چهار تصویر داده شده را به حوزه فرکانس برده و طیف آن‌ها را در چهار حالت بدون لگاریتم و بدون شیفت، بدون لگاریتم و با شیفت، با لگاریتم و بدون شیفت و با لگاریتم و با شیفت بررسی میکنیم.



شکل ۷) طیف تصویر *Lena* الف) بدون لگاریتم، بدون شیفت. ب) بدون لگاریتم، با شیفت. پ) با لگاریتم، بدون شیفت. ت) با لگاریتم، با شیفت

میتوان مشاهده کرد در تصویر۷- الف که مربوط به نمایش طیف بدون شیفت و لگاریتم است، تقریبا کل صفحه را به صورت سیاه مشاهده میکنیم.

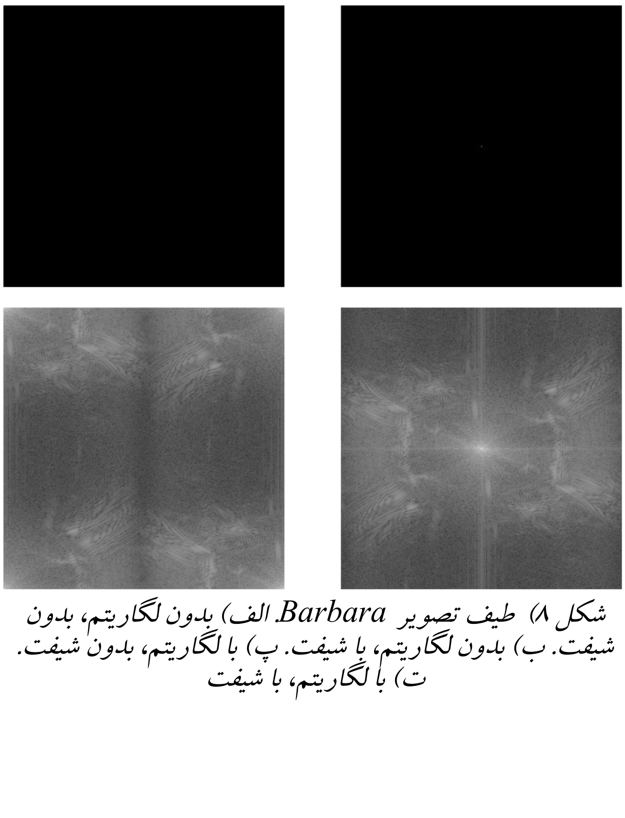
در تصویر ۷–ب ، نقطه ای سفید در وسط طیف مشاهده میشود و بقیه طیف به صورت سیاه نمایش داده میشود. این نتایج به دلیل عدم استفاده از لگاریتم است. تفاوت میان وسط و بقیه قسمت ها زیاد است و باعث میشود در نمایش فقط وسط طیف مقدار داشته باشد و بقیه طیف به صورت سیاه نمایش داده میشود. برای حل این مشکل از لگاریتم استفاده میکنیم تا *scale* مناسب تری ارائه دهد که در آن فاصله میان فرکانس های پایین و فرکانس های بالا کمتر باشد و بتوان طیف را *visualize* کرد.

در تصویر ۷–پ و ۷–ت میتوان طیف را بعد از لگاریتم گرفتن مشاهده کرد. برای بقیه تصاویر هم به همین صورت است. در ورژن های بدون لگاریتم تقریبا صفحه ای سیاه دیده میشود و در ورژن های لگاریتمی، طیف بهتر نمایش داده میشود.

برای شیفت نیز میتوان مشاهده کرد که در تصویر ۷–پ، فرکانس های پایین در گوشه ها قرار دارند و تحلیل کمی سخت تر است ولی در ورژن شیفت داده شده، فرکانس های پایین به مرکز تصویر شیفت داده شده اند و فرکانس های بالا که مربوط به لبه ها هستند به اطراف منتقل شده اند.

همانطور که میدانیم در طیف در حوزه فرکانس، لبه های عمودی در طیف بیانگر لبه افقی در عکس در حوزه مکان هستند و برعکس.

در طیف نمایش داده شده، میتوان مشاهده کرد که طیف مربوط به لبه افقی و عمودی ضرایب مشابهی دارند ولی برای لبه ها با زاویه ۱۳۵ درجه در طیف میتوان مشاهده کرد که مقادیر بیشتری از لبه های ۴۵ درجه دارند که میتواند برای لبه های کلاه در تصویر اصلی باشد.

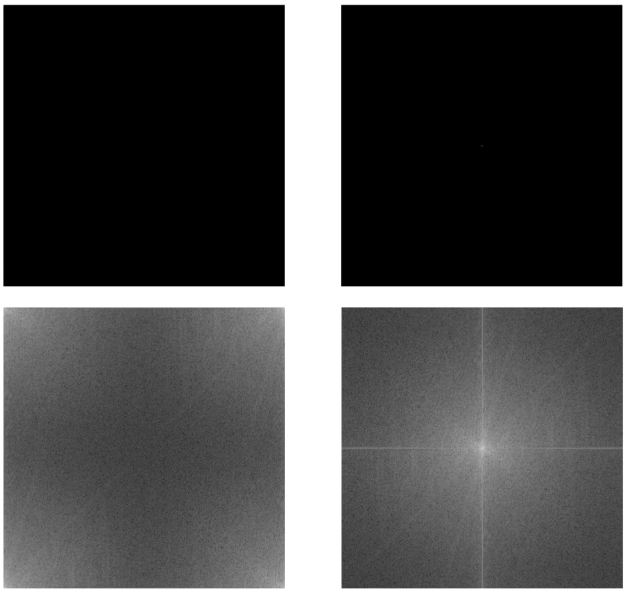


شکل ۸) طیف تصویر Barbara. الف) بدون لگاریتم، بدون شیفت. ب) بدون لگاریتم، با شیفت. پ) با لگاریتم، بدون شیفت. ت) با لگاریتم، با شیفت

مشاهده میشود که در ورژن های بدون لگاریتم، طیف حاصل همانند طیف تصویر *Lena* است و نمیشود اطلاعاتی از آن به دست آورد. و تنها در ۸–ب میتوان حاصل جمع تمامی پیکسل ها که مقدار زیادی شده است را به صورت یک نقطه سفید مشاهده کرد.

شکل ۹) تصویر Barbara.

همانطور که در عکس *Barbara* میتوان مشاهده کرد، لبه های کوچک عمودی بسیار زیادی وجود دارد. با بررسی طیف نیز میتوان به این نتیجه رسید. همانطورکه در طیف مشاهده میشود، در فرکانس های بالای طیف و نزدیک محور افقی تصویر، مقادیر بیشتری مشاهده میشود که نمایانگر فرکانس های بالای افقی میباشد که در حوزه مکان تبدیل به لبه های کوچک عمودی میشود.

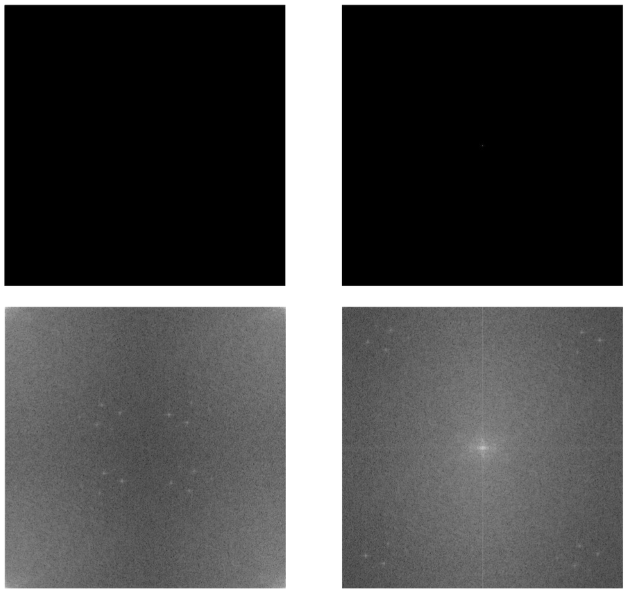


شکل ۱۰) طیف تصویر F16. الف) بدون لگاریتم، بدون شیفت. ب) بدون لگاریتم، با شیفت. پ) با لگاریتم، بدون شیفت. ت) با لگاریتم، با شیفت

در طیف مربوط به تصویر *F16* میتوان مشاهده کرد که مقادیر در محور عمودی و نزدیک مرکز، مقدار بزرگتری دارند که به این

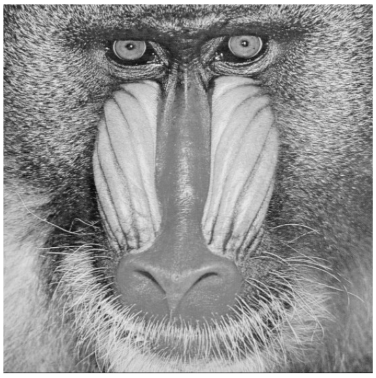
معنی است که لبه های افقی بزرگ در تصویر اصلی بیشتر هستند که با توجه به عکس ۱۱ نیز به نظر درست می‌آید زیرا هواپیما دارای لبه های افقی بزرگی است و لبه های عمودی به مراتب نسبت به لبه های افقی کوچک تر هستند.

شکل ۱۱) تصویر *F16*.



شکل ۱۲) طیف تصویر *Baboon*. الف) بدون لگاریتم، بدون شیفت. ب) بدون لگاریتم، با شیفت. پ) با لگاریتم، بدون شیفت. ت) با لگاریتم، با شیفت

در طیف تصویر *Baboon* میتوان مشاهده کرد که اکثر مقادیر بالا در مرکز طیف تجمیع شده اند و در فرکانس های بالاتر طیف، به جز سه نقطه مشخص شده مقادیر پایینی داریم. این به این دلیل است که در تصویر *Baboon*، لبه های بزرگی داریم و لبه های ریز تقریبا به چند زاویه ثابت در قسمت سبیل و موهای بدن *Baboon* قرار دارند و بقیه لبه ها لبه های بزرگی هستند که در طیف، فرکانس های کوچک نزدیک به مبدا هستند. در سه نقطه از فرکانس های بالا هم که مقادیر بالاتری داریم احتمالا برای وجود لبه های کوچک تر در سبیل ها و موهای *Baboon* است و ازآنجایی که زاویه این لبه ها تقریبا یکسان است، به صورت نقطه ای در طیف به وجود آمده اند.

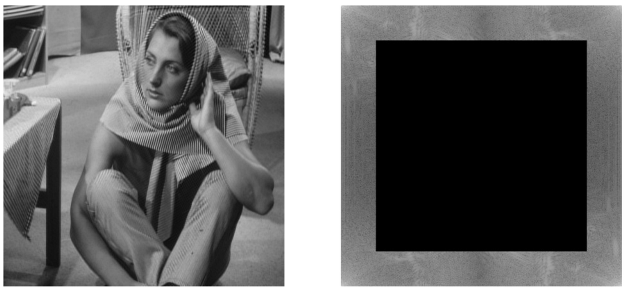


شکل ۱۳) تصویر *Baboon*.

۲.۳) تمرین ۴.۲.۲

در این تمرین همان طور که گفته شد، دو تابع برای صفر کردن مقدار طیف در حوزه فرکانس داده شده است که با دو ضریب از اندازه تصویر باید آنها را اعمال کنیم. نتایج به صورت زیر است:

شکل ۱۴) اعمال تابع a با استفاده از ضریب یک چهارم.

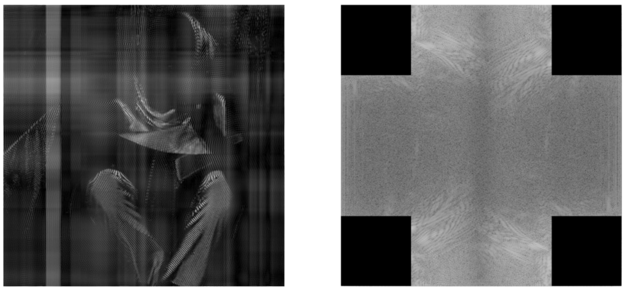


شکل ۱۵)) اعمال تابع a با استفاده از ضریب یک هشتم.

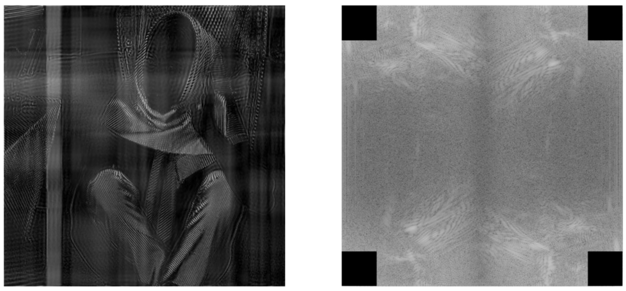
شکل ۱۶) مقایسه دو قسمت از تصویر. سمت چپ با استفاده از ضریب یک هشتم و سمت راست با استفاده از ضریب یک چهارم

ابتدا به بررسی طیف می‌پردازیم. در اینجا چون از *fftshift* استفاده نشده است، در نتیجه در وسط طیف، فرکانس های بالا یعنی لبه های تیز تر را داریم و در گوشه های طیف، فرکانس های پایین که نمایانگر کلیات تصویر هستند. در نتیجه با حذف کردن مقادیر وسط طیف، در واقع فرکانس های بالا را حذف میکنیم و فرکانس های پایین را نگه میداریم که همانند یک فیلتر *low pass* عمل کرده و تصویر را *smooth* میکند.

در تصویر ۱۶ میتوان نتیجه دو ضریب را بررسی کرد. همان طور که مشاهده میشود، تصویری که طیف آن با استفاده از ضریب $\frac{1}{8}$ صفر شده است، تصویر بیشتر *smooth* شده است زیرا فرکانس های بالای بیشتری حذف شده است.



شکل ۱۷) اعمال تابع b با استفاده از ضریب یک چهارم.



شکل ۱۸) اعمال تابع b با استفاده از ضریب یک هشتم.

شکل ۱۹) مقایسه دو قسمت از تصویر. سمت چپ با استفاده از ضریب یک هشتم و سمت راست با استفاده از ضریب یک چهارم

تابع b بر خلاف تابع a، فرکانس های پایین در طیف را حذف میکند و فرکانس های بالا را نگه میدارد. در نتیجه کلیات تصویر از بین میرود و فقط لبه ها که فرکانس های بالا هستند باقی می‌مانند.

میتوان مشاهده کرد که با استفاده از ضریب $\frac{1}{8}$ نسبت به ضریب $\frac{1}{4}$ محدوده کوچکتری از طیف صفر شده است که این بیانگر آن است که فرکانس های بالای کمتری از طیف حذف شده است و در نتیجه لبه های بیشتری در عکس نتیجه باید داشته باشیم. همان طور که انتظار میرود، در شکل ۱۹ میتوان مشاهده کرد که در حذف با ضریب $\frac{1}{8}$ ، لبه های بیشتری نسبت به ضریب دیگر وجود دارد.

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

lena = cv2.imread('Lena.bmp')
lena = cv2.cvtColor(lena, cv2.COLOR_BGR2GRAY)

barbara = cv2.imread('Barbara.bmp')
barbara = cv2.cvtColor(barbara, cv2.COLOR_BGR2GRAY)

f16 = cv2.imread('F16.bmp')
f16 = cv2.cvtColor(f16, cv2.COLOR_BGR2GRAY)

baboon = cv2.imread('Baboon.bmp')
baboon = cv2.cvtColor(baboon, cv2.COLOR_BGR2GRAY)

def show_img(*args, figsize=10, is_gray=True, title=None, fontsize=12):
    if isinstance(figsize, int):
        figsize = (figsize, figsize)
    images = args[0] if type(args[0]) is list else list(args)
    cmap=None
    if not is_gray:
        images = list(map(lambda x: cv2.cvtColor(x, cv2.COLOR_BGR2RGB), images))
    else:
        cmap = 'gray'
    plt.figure(figsize=figsize)
    for i in range(1, len(images)+1):
        plt.subplot(1, len(images), i)
        if title is not None:
            plt.title(title[i-1], fontsize=fontsize)

        plt.imshow(images[i-1], cmap=cmap)
        plt.axis('off')

def pad(x, n, m):
    y = np.zeros((n, m))
    y[0:x.shape[0], 0:x.shape[1]] = x
    return y

#------------------- 4.1.1 -------------------#

def fourier_spectrum(x, n, m):
    x = pad(x, n, m)
    y = np.fft.fft2(x)
    return abs(np.fft.fftshift(y))

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

a_1 = 1/4 * np.array([[1, 2, 1]])

show_img(fourier_spectrum(a_1, 512, 512))

a_2 = 1/4 * np.array([[1], [2], [1]])

show_img(fourier_spectrum(a_2, 512, 512))

show_img(fourier_spectrum(a, 512, 512))

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

show_img(fourier_spectrum(b, 512, 512))

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

show_img(fourier_spectrum(c, 512, 512))

def filter(img, filter):
    x = np.fft.fft2(pad(img, img.shape[0]*2, img.shape[1]*2))
    x_shift = np.fft.fftshift(x)
    f = np.fft.fft2(pad(filter, img.shape[0]*2, img.shape[1]*2))
    f_shift = np.fft.fftshift(f)
    y = x_shift * f_shift
    y = np.fft.ifftshift(y)
    return np.fft.ifft2(y)[0:img.shape[0], 0:img.shape[1]]

f = filter(lena, a)

show_img(abs(f), lena)

f = filter(lena, b)

show_img(abs(f), lena)

f = filter(lena, c)

show_img(abs(f), lena)

#------------------- 4.1.2 -------------------#

def fourier_spectrum_2(x):
    y = np.fft.fft2(x)
    z = np.fft.fftshift(y)
    return abs(y), abs(z), np.log(abs(y)), np.log(abs(z))

i, j, k, l = fourier_spectrum_2(lena)
show_img(i, j, figsize=30)
show_img(k, l, figsize=30)

i, j, k, l = fourier_spectrum_2(barbara)
show_img(i, j, figsize=30)
show_img(k, l, figsize=30)
show_img(barbara)

i, j, k, l = fourier_spectrum_2(f16)
show_img(i, j, figsize=30)
show_img(k, l, figsize=30)
show_img(f16)

i, j, k, l = fourier_spectrum_2(baboon)
show_img(i, j, figsize=30)
show_img(k, l, figsize=30)

show_img(baboon)

#------------------- 4.2.1 -------------------#

#              Don't need to code             #

#------------------- 4.2.2 -------------------#

def normalize(img):
    return ((img-img.min())/(img.max() - img.min()) * 255).astype('uint8')

def filter_2(img, n=0.25, mode='a'):
    o, p = int(n*img.shape[0]), int(n*img.shape[1])
    x = np.fft.fft2(img)
    if mode == 'a':
        x[o:img.shape[0]-o, p:img.shape[1]-p] = 0
    elif mode == 'b':
        x[0:o, 0:p] = 0
        x[img.shape[0]-o:-1, 0:p] = 0
        x[0:o, img.shape[1]-p:-1] = 0
        x[img.shape[0]-o:-1, img.shape[1]-p:-1] = 0

    y = np.fft.ifft2(x)
    x_r = np.log(abs(x))
    x_r[x_r == -math.inf] = 0

    return abs(y), x_r

r1, x1 = filter_2(barbara, 1/4, 'a')
show_img(normalize(r1), x1, figsize=30)

r2, x2 = filter_2(barbara, 1/4, 'b')
show_img(normalize(r2), x2, figsize=30)

r3, x3 = filter_2(barbara, 1/8, 'a')
show_img(normalize(r3), x3, figsize=30)

r4, x4 = filter_2(barbara, 1/8, 'b')
show_img(normalize(r4), x4, figsize=30)
```