مبانی بینایی کامپیوتر

تمرین چهارم

نیکی نزاکتی
98522094

$$F(u,v) = \sum_{x=0}^{1}\sum_{y=0}^{1} f(x,y)\, e^{-j2\pi\left(\frac{ux+vy}{2}\right)}$$

$$= f(0,0)\, e^{-j2\pi(0)} + f(0,1)\, e^{-j2\pi\left(\frac{v}{2}\right)}$$

$$+ f(1,0)\, e^{-j2\pi\left(\frac{u}{2}\right)} + f(1,1)\, e^{-j2\pi\left(\frac{u+v}{2}\right)}$$

$$\Rightarrow F(u,v) = 2 + e^{-j\pi v} + 3e^{-j\pi u} + 4e^{-j\pi(u+v)}$$

$$\rightarrow F(0,0) = 2+1+1+1 = 5$$

$$\rightarrow F(0,1) = 2 + e^{-j\pi} + 1 + 4e^{-j\pi} = 3 + 5e^{-j\pi}$$

$$\rightarrow F(1,0) = 2 + 3e^{-j\pi} + 4e^{-j\pi} = 2 + 7e^{-j\pi}$$

$$\rightarrow F(1,1) = 2 + e^{-j\pi} + 3e^{-j\pi} + 4e^{-2j\pi} = 2 + 8e^{-j\pi}$$

.2

الف) تصویر با تبدیل فوریه به حوزهٔ فرکانس منتقل می‌شود و در تبدیل فوریه ماتریس حاصل یک ماتریس متقارن است:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=1}^{N+1} f(x,y)e^{-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} = \sum_{x=0}^{1}\sum_{y=0}^{1} f(x,y)e^{-i2\pi\left(\frac{vx}{M}+\frac{uy}{N}\right)} = F(v,u)$$

پس چون ماتریس ما متقارن است تعداد مولفه‌های آزاد آن $\frac{n \times n+1}{2}$ است:

تعداد درایه‌ها $\rightarrow \frac{n(n-1)}{2} + n \leftarrow$ مولفه‌های قطر اصلی

ب)

$$F(0,0) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi\left(\frac{0\cdot x}{M}+\frac{0\cdot y}{N}\right)} = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$$

می‌بینیم که نقطهٔ $(0,0)$ تبدیل فوریه تصویر برابر با مجموع همهٔ پیکسل‌های تصویر است و مقدار حقیقی دارد.

.3

الف)

برای پیاده سازی تابع filter_2d ابتدا کرنل ورودی را معکوس کرده، سپس عکس را با توجه به اندازه کرنل pad میکنیم، و در نهایت با ضرب ماتریسی کرنل در عکس بدست آمده به خروجی میرسیم.

```
kernel = np.flipud(np.fliplr(kernel))
image_padded = np.zeros((image.shape[0] + (kernel.shape[0]-1),
        image.shape[1] + (kernel.shape[1]-1)))
image_padded[(kernel.shape[0]//2):-(kernel.shape[0]//2), (kernel.shape[1]//
2):-(kernel.shape[1]//2)] = image
 result[y,x]=(kernel*image_padded[y:y + kernel.shape[0], x:x +
kernel.shape[1]]).sum()
```

ب) کرنل میانگین گیر به صورت زیر پیاده سازی می شود.

```
result = np.ones((size, size))
result = np.divide(result, size*size)
```

خروجی حاصل از اعمال آن:

پ)

تفاوت پیاده سازی کرنل میانه گیر نسبت به میانگین گیر در خط زیر است:

```
result[y,x] = np.median(image_padded[y:y + kernel.shape[0], x:x +
kernel.shape[1]])
```

خروجی حاصل از اعمال آن:

مزایا و معایب فیلتر نسبت به میانگین گیر:

با محاسبه مقدار میانه یک همسایگی به جای فیلتر میانگین گیر، فیلتر میانه گیر دو مزیت اصلی نسبت به فیلتر میانگین گیر دارد:

میانه میانگین گیر robust تری نسبت به میانگین است و بنابراین یک پیکسل غیر نماینده در یک همسایگی تأثیر قابل توجهی بر مقدار میانه نخواهد داشت. علاوه بر این، از آنجایی که مقدار میانه باید در واقع مقدار یکی از پیکسل های همسایگی باشد، فیلتر میانه گیر مقادیر پیکسل غیر واقعی جدیدی را زمانی که فیلتر در یک لبه قرار می گیرد ایجاد نمی کند. به همین دلیل فیلتر میانه گیر در حفظ لبه های تیز بسیار بهتر از فیلتر میانگین گیر است.

عیب فیلتر میانه گیر این است که تحلیل اثر آن بر روی تصویر دشوار است چون error propagation وجود ندارد.

4.

الف)

برای حذف نویز، ابتدا تصویر به کمک ماژول fft2 به حوزه فرکانس می بریم:

```
image_fft = fftpack.fft2(image)
```

سپس با در نظر گرفتن مقداری کوچک به نام keep_fraction و ضرب ماتریسی آن در تصویر، نواحی بین سطر * keep_fraction و سطر * (1-keep_fraction) را برابر صفر قرار می دهیم. به صورت مشابه این عملیات روی ستون ها انجام می شود.

```
image_fft2[int(r*keep_fraction):int(r*(1-keep_fraction))] = 0
image_fft2[:, int(c*keep_fraction):int(c*(1-keep_fraction))] = 0
```

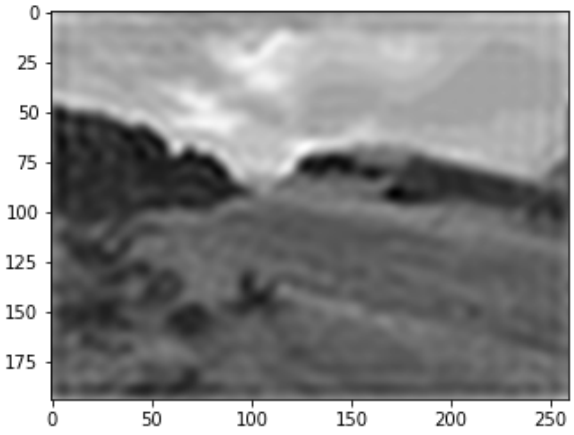
در نهایت با استفاده از ماژول ifft2 تصویر را از حوزه فرکانسی خارج می کنیم.

```
denoised = fftpack.ifft2(image_fft2).real
```

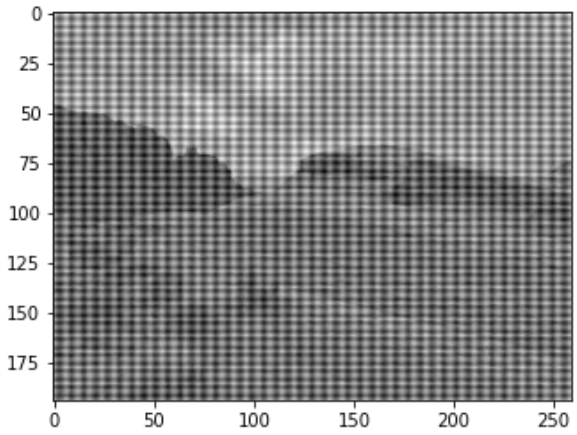
ب)

نسبت سیگنال به نویز پیک (PSNR) نسبت بین حداکثر توان ممکن یک تصویر و قدرت نویز مخرب است که بر کیفیت نمایش آن تأثیر می گذارد. پس هر چه مقدار PSNR بزرگتر باشد تصویر خروجی بهتر است. با توجه به PSNR تصویر اولیه، و افزایش این مقدار بعد از حذف نویز، میتوان به این نتیجه رسید که عملکرد خوبی داشتیم.

```
PSNR between noisy image and original image =  8.122255096865844
PSNR between denoised image and original image =  24.804180799257377
```



Denoised Image



Original Image

پ)

نویز اضافه شده به تصویر از نوع ضرب شونده است. در این نوع نویز، سیگنال نویزی در سیگنال واقعی ضرب می شود. حذف کردن این نویز با تبدیل ماهیت ضربی به افزودنی با استفاده از تبدیل لگاریتمی انجام می شود. با استفاده از تبدیل log، نویز ضربی به نویز افزایشی تبدیل می شود و سپس می توان هر روش فیلتری را برای آن اعمال کرد. و بعداً از لاگ معکوس برای به دست آوردن نتیجه مطلوب استفاده می شود. در مقابل در نویز جمعی، سیگنال نویزی با سیگنال واقعی جمع می شود. تأثیر این نویز بر روی تصویر کمتر از نویز ضربی است و این نویز پترنی با توزیع نرمال دارد.