

Computer Vision

HW8

R10921104 王怡堯

November 23, 2021

1 Primary Procedures

For Opening and Closing , I'll use the code in HW5.

1.1 Gaussian Noise

```
1 def gaussian_noise(lena, amplitude):
2     m=len(lena)
3     n=len(lena[0])
4     res=[[0]*n for i in range(m)]
5     for i in range(m):
6         for j in range(n):
7             res[i][j]=max(0,min(255,lena[i][j]+amplitude*random.gauss(0,1)))
8     return res
```

1.2 Salt and Pepper Noise

```
1 def s_and_p_noise(lena, threshold):
2     m=len(lena)
3     n=len(lena[0])
4     res=[[0]*n for i in range(m)]
5     for i in range(m):
6         for j in range(n):
7             tmp=random.random()
8             if tmp<threshold:
9                 res[i][j]=0
10            elif tmp>1-threshold:
11                res[i][j]=255
12            else:
13                res[i][j]=lena[i][j]
14    return res
```

1.3 Box Filter (3*3)

```
1 def box_filter3(lena):
2     m=len(lena)
3     n=len(lena[0])
4     lena=expn(lena)
5     res=[[0]*n for i in range(m)]
6     for i in range(m):
7         tmp=lena[i][0]+lena[i+1][0]+lena[i+2][0]
8         tmp+=lena[i][1]+lena[i+1][1]+lena[i+2][1]
9         tmp+=lena[i][2]+lena[i+1][2]+lena[i+2][2]
10        res[i][0]=tmp//9
11        for j in range(1,n):
12            tmp-=lena[i][j-1]+lena[i+1][j-1]+lena[i+2][j-1]
13            tmp+=lena[i][j+2]+lena[i+1][j+2]+lena[i+2][j+2]
14            res[i][j]=tmp//9
15    return res
```

1.4 Box Filter (5*5)

```
1 def box_filter5(lena):
2     m=len(lena)
3     n=len(lena[0])
4     lena=expan(lena)
5     lena=expan(lena)
6     res=[[0]*n for i in range(m)]
7     for i in range(m):
8         tmp=lena[i][0]+lena[i+1][0]+lena[i+2][0]+lena[i+3][0]+lena[i+4][0]
9         tmp+=lena[i][1]+lena[i+1][1]+lena[i+2][1]+lena[i+3][1]+lena[i+4][1]
10        tmp+=lena[i][2]+lena[i+1][2]+lena[i+2][2]+lena[i+3][2]+lena[i+4][2]
11        tmp+=lena[i][3]+lena[i+1][3]+lena[i+2][3]+lena[i+3][3]+lena[i+4][3]
12        tmp+=lena[i][4]+lena[i+1][4]+lena[i+2][4]+lena[i+3][4]+lena[i+4][4]
13        res[i][0]=tmp//25
14        for j in range(1,n):
15            tmp-=lena[i][j-1]+lena[i+1][j-1]+lena[i+2][j-1]+lena[i+3][j-1]+lena[i+4][j-1]
16            tmp+=lena[i][j+4]+lena[i+1][j+4]+lena[i+2][j+4]+lena[i+3][j+4]+lena[i+4][j+4]
17            res[i][j]=tmp//25
18    return res
```

1.5 Median Filter (3*3)

```
1 def med_filter3(lena):
2     m=len(lena)
3     n=len(lena[0])
4     lena=expan(lena)
5     res=[[0]*n for i in range(m)]
6     for i in range(m):
7         for j in range(n):
8             res[i][j]=sorted(lena[i][j:j+3]+lena[i+1][j:j+3]+lena[i+2][j:j+3])[4]
9     return res
```

1.6 Median Filter (5*5)

```
1 def med_filter5(lena):
2     m=len(lena)
3     n=len(lena[0])
4     lena=expan(lena)
5     lena=expan(lena)
6     res=[[0]*n for i in range(m)]
7     for i in range(m):
8         for j in range(n):
9             tmp=sorted(lena[i][j:j+5]+lena[i+1][j:j+5]+lena[i+2][j:j+5]+lena[i+3][j:j+5]+lena[i+4][j:j+5])
10            res[i][j]=tmp[12]
11    return res
```

1.7 Computing Signal Noise Rate (SNR)

```
1 def SNR(origin,lena):
2     m=len(origin)
3     n=len(origin[0])
4     mu=0
5     mu_n=0
6     for i in range(m):
7         for j in range(n):
8             mu+=origin[i][j]
9             mu_n+=lena[i][j]-origin[i][j]
10    mu/=m*n
11    mu_n/=m*n
12    vs=0
13    vn=0
14    for i in range(m):
15        for j in range(n):
16            vs+=(origin[i][j]-mu)**2
17            vn+=(lena[i][j]-origin[i][j]-mu_n)**2
18    vs/=m*n
19    vn/=m*n
20    return 20*np.log10((vs**0.5)/(vn**0.5))
```

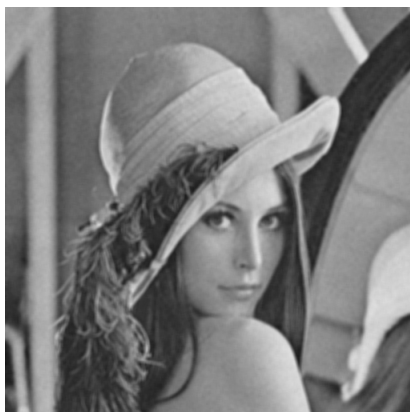
2 Result



Gaussian Noise (amplitude=10)
SNR=13.6130506044



Box Filter 3*3
SNR=15.4130851200



Box Filter 5*5
SNR=14.870670493



Median Filter 3*3
SNR=17.6862701444



Median Filter 5*5
SNR=15.9874514287



Opening-then-Closing
SNR=6.4587138667



Closing-then-Opening
SNR=5.29204632897



Gaussian Noise (amplitude=30)
SNR=4.1657734748



Box Filter 3*3
SNR=12.5972301851



Box Filter 5*5
SNR=13.3031734220



Median Filter 3*3
SNR=11.0682457441



Median Filter 5*5
SNR=12.8832447983



Opening-then-Closing
SNR=7.0260368184



Closing-then-Opening
SNR=3.85076320620



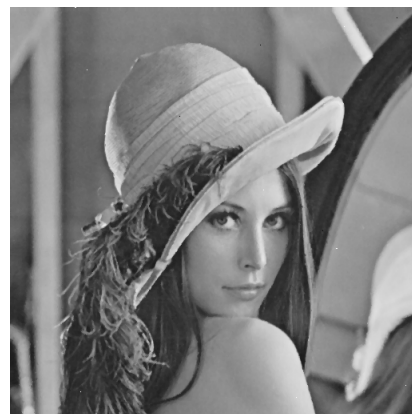
Salt and Pepper Noise (threshold=0.05)
SNR=0.9043074295



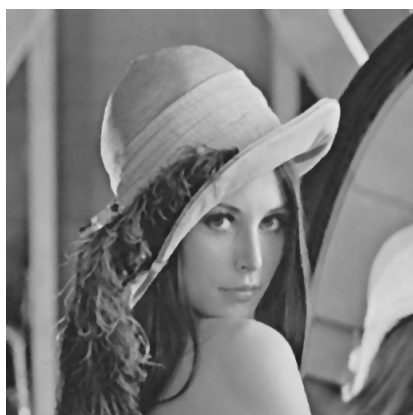
Box Filter 3*3
SNR=9.4436221358



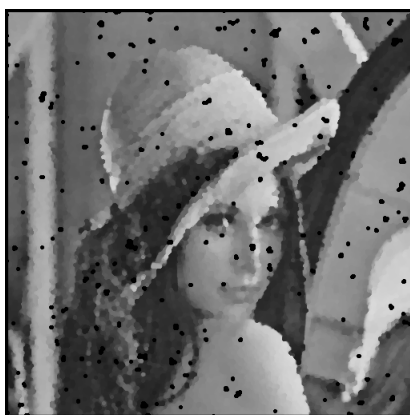
Box Filter 5*5
SNR=11.120614263



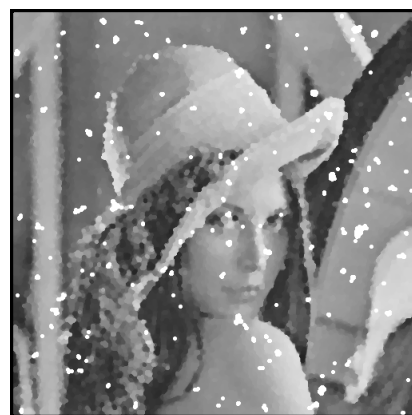
Median Filter 3*3
SNR=19.311863677



Median Filter 5*5
SNR=16.393028419



Opening-then-Closing
SNR=3.64708160609



Closing-then-Opening
SNR=2.75776652710



Salt and Pepper Noise (threshold=0.1)
SNR=-2.0969583441



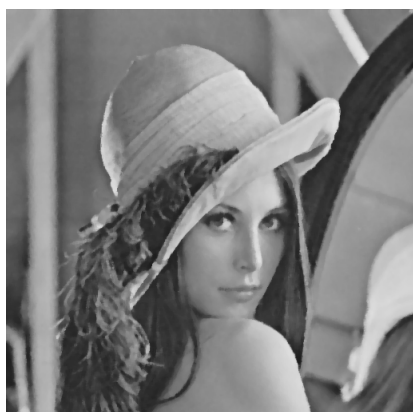
Box Filter 3*3
SNR=6.352908846



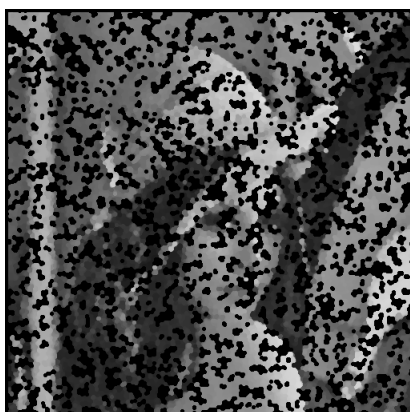
Box Filter 5*5
SNR=8.5315483297



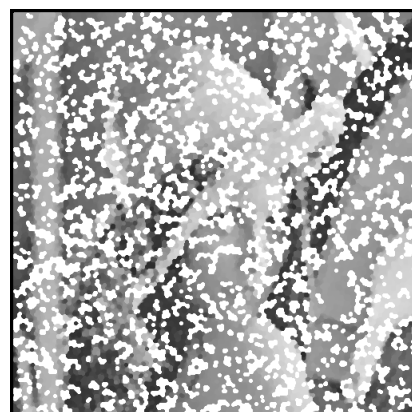
Median Filter 3*3
SNR=15.0759910952



Median Filter 5*5
SNR=15.745725733



Opening-then-Closing
SNR=-2.3192932549702623



Closing-then-Opening
SNR=-3.2127249492740066