

Computer Vision

HW10

R10921104 王怡堯

December 13, 2021

1 Primary Procedures

1.1 My Convolution

```
1 def my_conv(lena,mask,theshold):
2     m=len(lena)-len(mask)+1
3     n=len(lena[0])-len(mask[0])+1
4     mk=len(mask)
5     nk=len(mask[0])
6     res=[[0]*n for _ in range(m)]
7     for i in range(m):
8         for j in range(n):
9             tmp=0
10            for k in range(mk):
11                for l in range(nk):
12                    tmp+=mask[k][l]*lena[i+k][j+l]
13            if tmp>=theshold:
14                res[i][j]=1
15            elif tmp<=-theshold:
16                res[i][j]=-1
17
18     return res
```

1.2 Laplace Mask1

```
1 def Laplace1(lena,theshold):
2     m=len(lena)
3     n=len(lena[0])
4     mask=[[0, 1, 0],
5           [1, -4, 1],
6           [0, 1, 0]]
7     lena=expan(lena)
8     res1=my_conv(lena,mask,theshold)
9     res1=expan(res1)
10    res2=[[1]*m for _ in range(n)]
11    for i in range(m):
12        for j in range(n):
13            if res1[i+1][j+1]==1:
14                tmp=False
15                for k in range(3):
16                    for l in range(3):
17                        if res1[i+k][j+l]==-1:
18                            tmp=True
19                            break
20                if tmp:
21                    break
22            if tmp:
23                res2[i][j]=0
24    return np.array(res2)*255
```

1.3 Laplace Mask2

```
1 def Laplace2(lena,threshold):
2     m=len(lena)
3     n=len(lena[0])
4     mask=[[1, 1, 1],
5           [1, -8, 1],
6           [1, 1, 1]]
7     lena=expan(lena)
8     res1=my_conv(lena,mask,threshold)
9     res1=expan(res1)
10    res2=[[1]*m for _ in range(n)]
11    for i in range(m):
12        for j in range(n):
13            if res1[i+1][j+1]==1:
14                tmp=False
15                for k in range(3):
16                    for l in range(3):
17                        if res1[i+k][j+l]==-1:
18                            tmp=True
19                            break
20                if tmp:
21                    break
22            if tmp:
23                res2[i][j]=0
24    return np.array(res2)*255
```

1.4 Minimum variance Laplacian

```
1 def Minimum_Variance_Laplacian(lena,threshold):
2     m=len(lena)
3     n=len(lena[0])
4     mask=[[2, -1, 2],
5           [-1, -4, -1],
6           [2, -1, 2]]
7     lena=expan(lena)
8     res1=my_conv(lena,mask,threshold)
9     res1=expan(res1)
10    res2=[[1]*m for _ in range(n)]
11    for i in range(m):
12        for j in range(n):
13            if res1[i+1][j+1]==1:
14                tmp=False
15                for k in range(3):
16                    for l in range(3):
17                        if res1[i+k][j+l]==-1:
18                            tmp=True
19                            break
20                if tmp:
21                    break
22            if tmp:
23                res2[i][j]=0
24    return np.array(res2)*255
```

1.5 Laplace of Gaussian

```
1 def Laplace_of_Gaussian(lena,threshold):
2     m=len(lena)
3     n=len(lena[0])
4     mask=[[0, 0, 0, -1, -1, -2, -1, -1, 0, 0, 0],
5           [0, 0, -2, -4, -8, -9, -8, -4, -2, 0, 0],
6           [0, -2, -7, -15, -22, -23, -22, -15, -7, -2, 0],
7           [-1, -4, -15, -24, -14, -1, -14, -24, -15, -4, -1],
8           [-1, -8, -22, -14, 52, 103, 52, -14, -22, -8, -1],
9           [-2, -9, -23, -1, 103, 178, 103, -1, -23, -9, -2],
10          [-1, -8, -22, -14, 52, 103, 52, -14, -22, -8, -1],
11          [-1, -4, -15, -24, -14, -1, -14, -24, -15, -4, -1],
12          [0, -2, -7, -15, -22, -23, -22, -15, -7, -2, 0],
13          [0, 0, -2, -4, -8, -9, -8, -4, -2, 0, 0],
14          [0, 0, 0, -1, -1, -2, -1, -1, 0, 0, 0]]
15    for i in range(5):
16        lena=expan(lena)
17        res1=my_conv(lena,mask,threshold)
18        res1=expan(res1)
19        res2=[[1]*m for _ in range(n)]
20        for i in range(m):
21            for j in range(n):
22                if res1[i+1][j+1]==1:
23                    tmp=False
24                    for k in range(3):
25                        for l in range(3):
26                            if res1[i+k][j+l]==-1:
27                                tmp=True
28                                break
29                    if tmp:
30                        break
31            if tmp:
32                res2[i][j]=0
33    return np.array(res2)*255
```

1.6 Difference of Gaussian

```
1 def Difference_of_Gaussian(lena, threshold):
2     m=len(lena)
3     n=len(lena[0])
4     mask=[[-1, -3, -4, -6, -7, -8, -7, -6, -4, -3, -1],
5           [-3, -5, -8, -11, -13, -13, -13, -11, -8, -5, -3],
6           [-4, -8, -12, -16, -17, -17, -17, -16, -12, -8, -4],
7           [-6, -11, -16, -16, 0, 15, 0, -16, -16, -11, -6],
8           [-7, -13, -17, 0, 85, 160, 85, 0, -17, -13, -7],
9           [-8, -13, -17, 15, 160, 283, 160, 15, -17, -13, -8],
10          [-7, -13, -17, 0, 85, 160, 85, 0, -17, -13, -7],
11          [-6, -11, -16, -16, 0, 15, 0, -16, -16, -11, -6],
12          [-4, -8, -12, -16, -17, -17, -17, -16, -12, -8, -4],
13          [-3, -5, -8, -11, -13, -13, -13, -11, -8, -5, -3],
14          [-1, -3, -4, -6, -7, -8, -7, -6, -4, -3, -1]]
15     for i in range(5):
16         lena=expan(lena)
17     res1=my_conv(lena,mask,threshold)
18     res1=expan(res1)
19     res2=[[1]*m for _ in range(n)]
20     for i in range(m):
21         for j in range(n):
22             if res1[i+1][j+1]==1:
23                 tmp=False
24                 for k in range(3):
25                     for l in range(3):
26                         if res1[i+k][j+l]==-1:
27                             tmp=True
28                             break
29                 if tmp:
30                     break
31             if tmp:
32                 res2[i][j]=0
33     return np.array(res2)*255
```

2 Result



thresholds:15



thresholds:15



thresholds:10



thresholds:3000



thresholds:1