Computer Vision Hw4

r07921010 余奇安

1.Binary morphological dilation

We choose 35553 as dilation kernel, and the input image is Lena.jpg thresholded at 128.
Binary image which is threshold at 128:

Dilation image:

We can find that the boundary of objects dilate after dilation, lots of black points disappear.
Code:
I define dil_kernel and dilation function to achieve the function.
The dil_kernal extend the flexibility or code so that I can change to different kind of kernals easily.

```
def dil_kernal(img,row,col):
    for i in range(-2,3):
        for j in range(-2,3):
            if (i==-2 and j==-2) or (i==-2 and j==2)or(i==2 and j==-2)or(i==2 and j==2):
                pass
            else:
                img[row+i][col+j]=255
```

```
def dilation(img):
    row=img.shape[0]
    col=img.shape[1]
    new=np.zeros((row+4,col+4),dtype=np.int)
    res=np.zeros((row,col),dtype=np.int)

    for i in range(row):
        for j in range(col):
            if img[i][j]==255:
                dil_kernal(new,i+2,j+2)
    for i in range(row):
        for j in range(col):
            res[i][j]=new[i+2][j+2]
    return res
```

2.Erosion

Use 35553 kernel the same as dilation.
Erosion image:

We can find that the boundary of objects contract after erosion.
Code:
I define ero_kernel and erosion function.
The reason to define ero_kernel is the same as dia_kernel.

```
def ero_kernal(img,row,col):
    for i in range(-2,3):
        for j in range(-2,3):
            if (i==-2 and j==-2) or (i==-2 and j==2)or(i==2 and j==-2)or(i==2 and j==2):
                pass
            else:
                if img[row+i][col+j]!=255:
                    return False
    return True
```

```
def erosion(img):
    row=img.shape[0]
    col=img.shape[1]
    new=np.zeros((row+4,col+4),dtype=np.int)
    res=np.zeros((row,col),dtype=np.int)
    for i in range(row):
        for j in range(col):
            new[i+2][j+2]=img[i][j]

    for i in range(2,new.shape[0]):
        for j in range(2,new.shape[1]):
            if new[i][j]==255 and ero_kernal(new,i,j):
                res[i-2][j-2]=255
    return res
```

3.Opening

Opening morphological means that do the erosion fast then dilation.
Using the same 35553 kernel.
Image:

We can find that after opening operation, we can eliminate some small connection between objects.
Code:

```
def opening(img):
    ero_im=erosion(img)
    open_im=dilation(ero_im)
    return open_im
```

4.Closing

The closing operation do the dilation first and then do the erosion.

Image:

We can find that lots of small objects get connected after closing.

Code:

```
def opening(img):
    ero_im=erosion(img)
    open_im=dilation(ero_im)
    return open_im
```

5.Hit and miss

We choose upside-down L shape kernel to detect the up right corner of objects.

Image:

Code:

```
def hit_and_miss(img):
    #Ac
    row=img.shape[0]
    col=img.shape[1]
    new=np.zeros((row,col),dtype=np.int)
    Ac=np.zeros((row,col),dtype=np.int)
    new2=np.zeros((row,col),dtype=np.int)
    res=np.zeros((row,col),dtype=np.int)
    for i in range(row):
        for j in range(col):
            if img[i][j]==255:
                Ac[i][j]=0
            else:
                Ac[i][j]=255
    for i in range(row-1):
        for j in range(1,col):
            if img[i][j]==255 and img[i][j-1]==255 and img[i+1][j]==255:
                new[i][j]=255
    for i in range (1,row):
        for j in range(col-1):
            if Ac[i-1][j]==255 and Ac[i-1][j+1] and Ac[i][j+1]:
                new2[i][j]=255
    for i in range(row):
        for j in range(col):
            if new[i][j]==255 and new2[i][j]==255:
                res[i][j]=255
    return res
```

The L shape kernel is easy, so I just to write it in judgement, rather than other kernel function.
We can find that the up right corner is preserved after upside-down L shape kernel after hit and miss operation.