Computer Vision hw4

B05902050

黃子源

I use PIL to complete the homework. In my program, I use function getpixel() and putpixel() to get the value of every pixel.

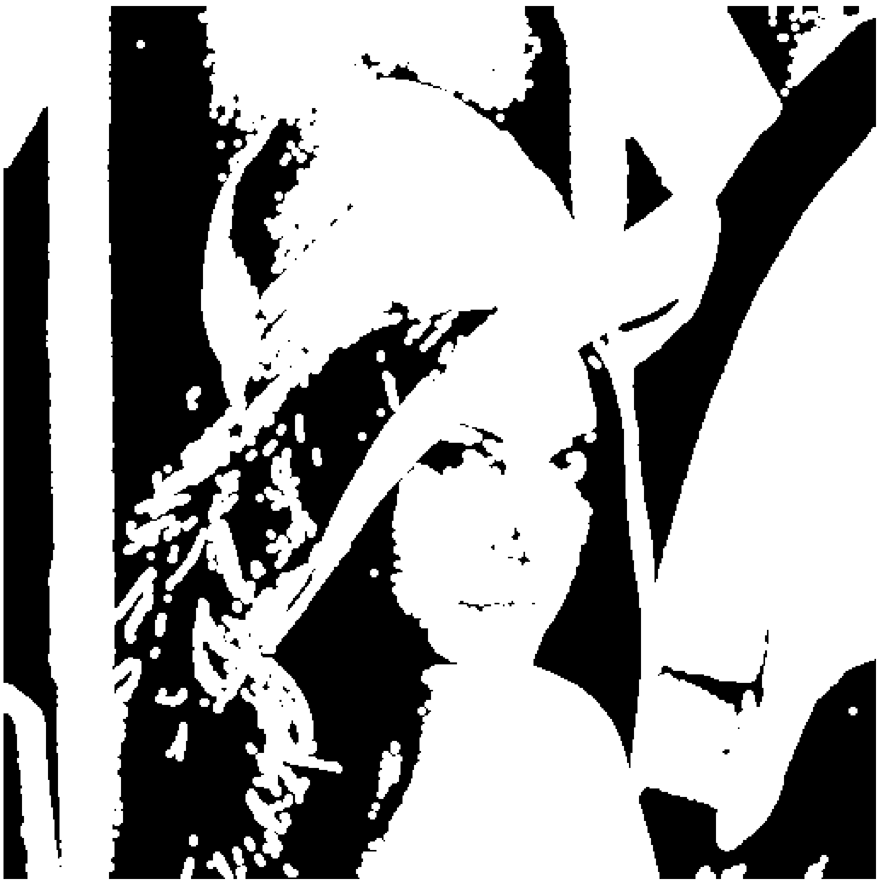
For the 3-5-5-5-3 octogonal, I create a list which represent an octagon whose original is in the center.

Dilation:

First, I read the value of pixels on lena.bmp. Suppose the value of pixel (x, y) is 1, then for the octagon whose center is (x,y), we set all the pixels in octagon into 1.

Principal code fragment.

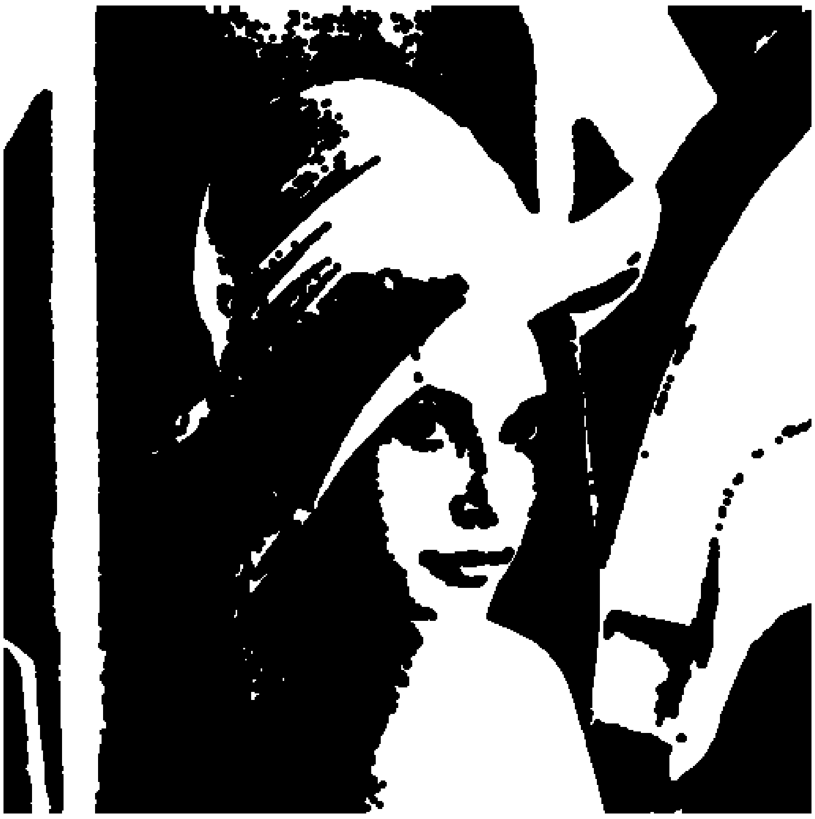
The image after dilation:



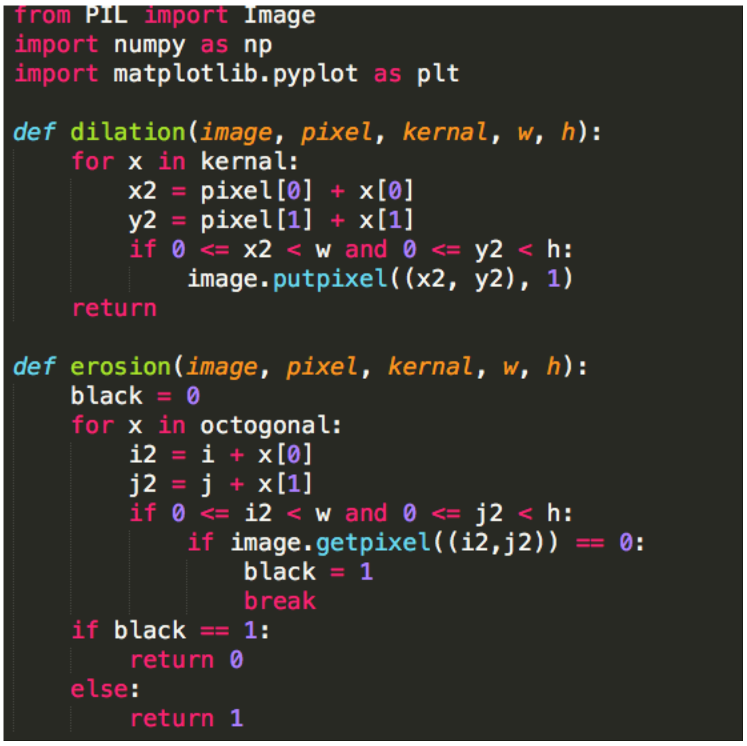
Erosion:

In this part, for all the pixels (x, y) whose value are 1, I check their pixels within 3-5-5-5-3 octogonal , if there is any pixel whose value is 1, then we turn the value of (x, y) into 0.

The image after erosion:



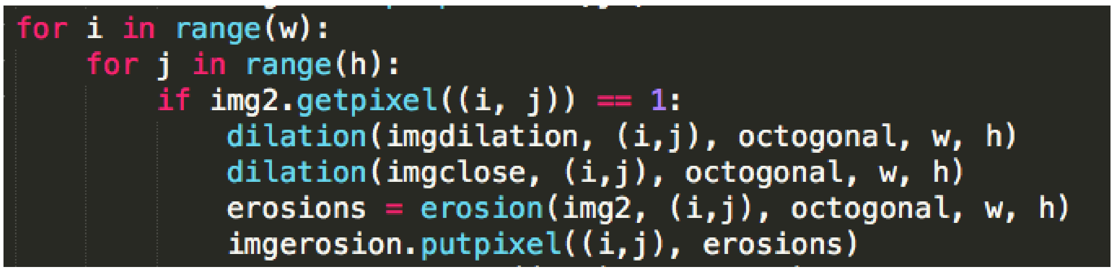
Principal code fragment of dilation and erosion:



```
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

def dilation(image, pixel, kernal, w, h):
    for x in kernal:
        x2 = pixel[0] + x[0]
        y2 = pixel[1] + x[1]
        if 0 <= x2 < w and 0 <= y2 < h:
            image.putpixel((x2, y2), 1)
    return

def erosion(image, pixel, kernal, w, h):
    black = 0
    for x in octogonal:
        i2 = i + x[0]
        j2 = j + x[1]
        if 0 <= i2 < w and 0 <= j2 < h:
            if image.getpixel((i2,j2)) == 0:
                black = 1
                break
    if black == 1:
        return 0
    else:
        return 1
```



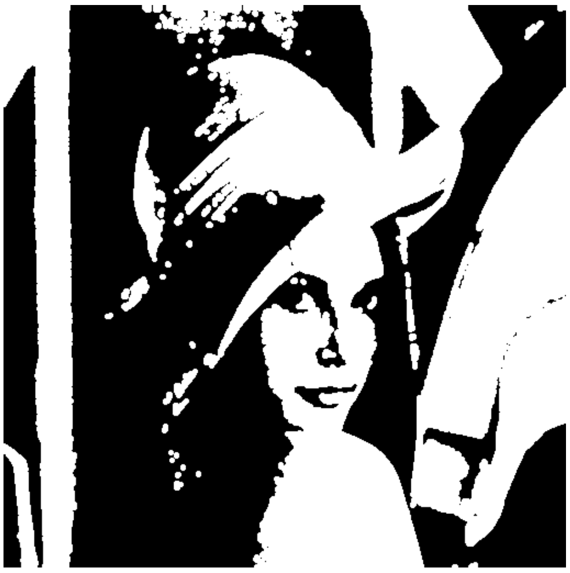
```
for i in range(w):
    for j in range(h):
        if img2.getpixel((i, j)) == 1:
            dilation(imgdilation, (i,j), octogonal, w, h)
            dilation(imgclose, (i,j), octogonal, w, h)
            erosions = erosion(img2, (i,j), octogonal, w, h)
            imgerosion.putpixel((i,j), erosions)
```

Opening and closing:

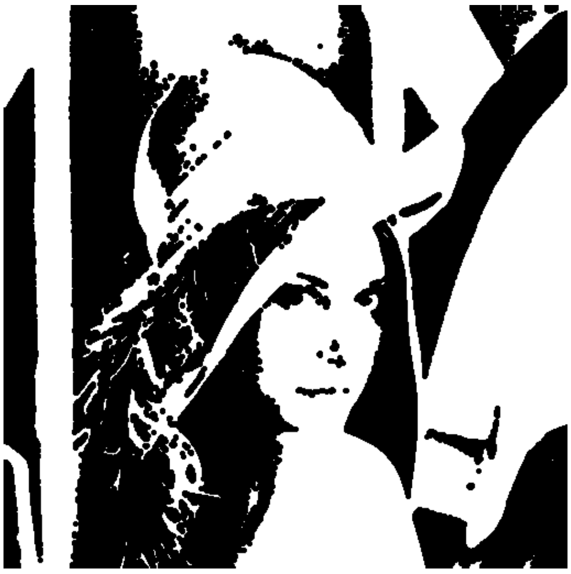
This part is quite simple. For opening, just do the erosion first and do the dilation;

And for closing do the dilation after the erosion.

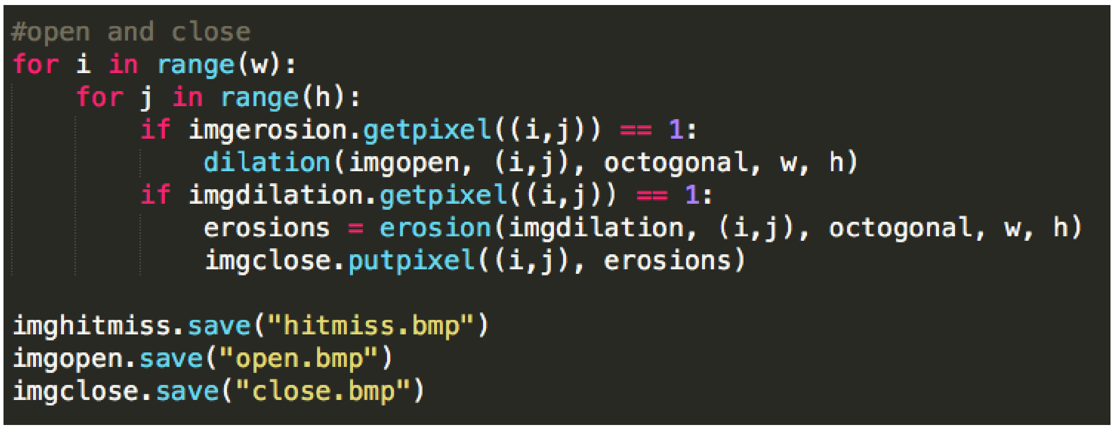
The image after opening:



The image after closing:



Principal code fragment of opening and closing:



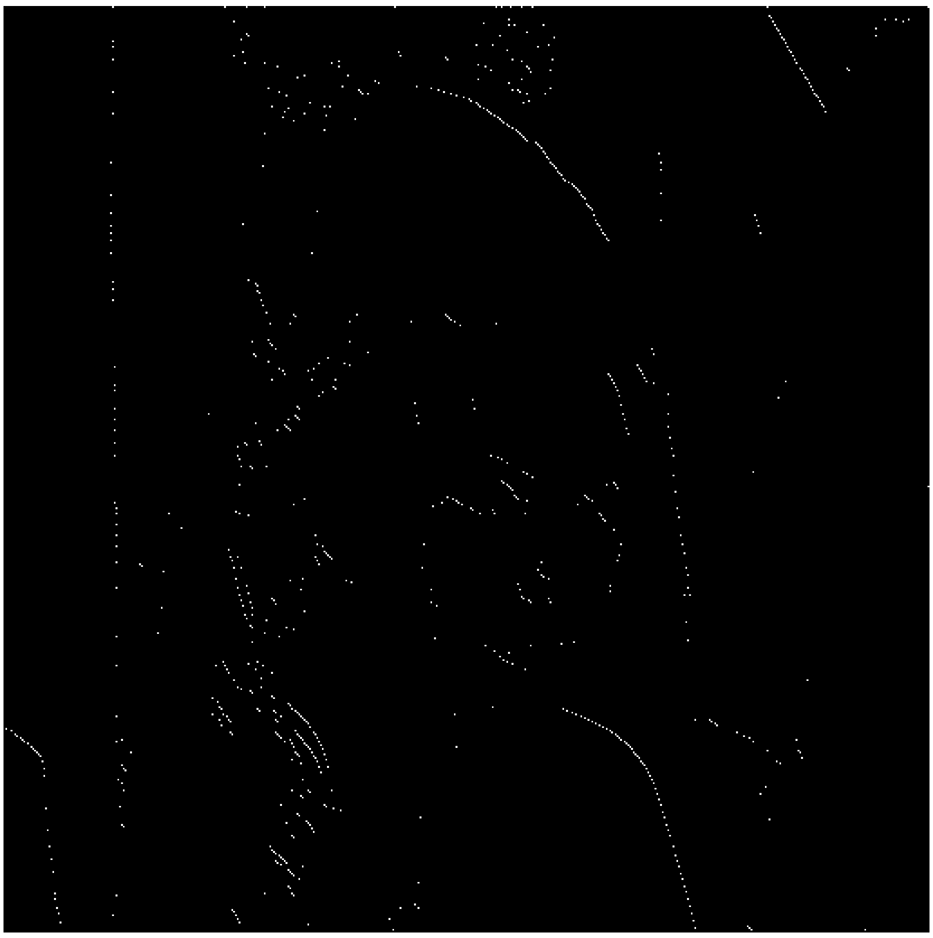
```
#open and close
for i in range(w):
    for j in range(h):
        if imgerosion.getpixel((i,j)) == 1:
            dilation(imgopen, (i,j), octogonal, w, h)
        if imgdilation.getpixel((i,j)) == 1:
            erosions = erosion(imgdilation, (i,j), octogonal, w, h)
            imgclose.putpixel((i,j), erosions)

imghitmiss.save("hitmiss.bmp")
imgopen.save("open.bmp")
imgclose.save("close.bmp")
```

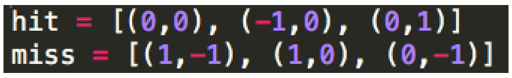
Hit and miss:

I create two list: hit and miss, which hit is the position of pixels that should be 1 and miss is the position of pixels that should be 0. After that, for all the pixels whose value is 1, check whether their value of hit list are all 1 and miss are all 0, if not, then turn the value of the pixel into 0.

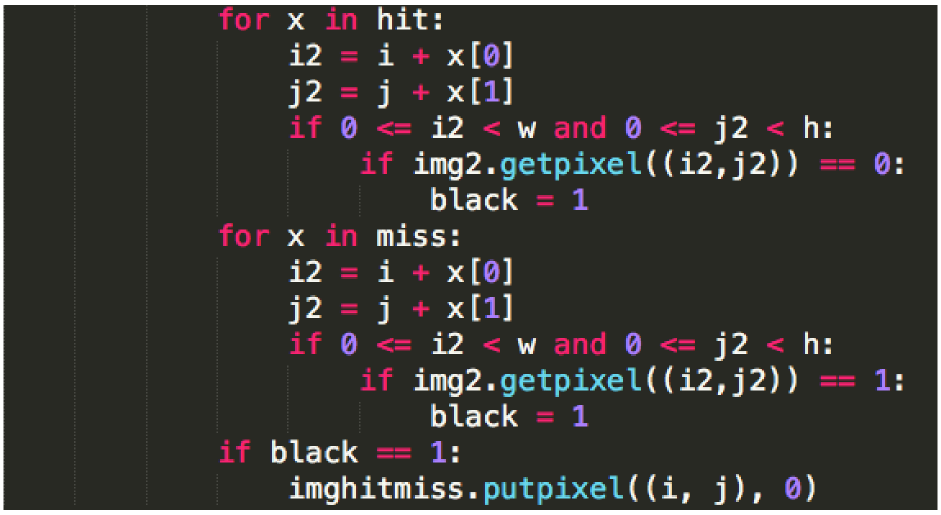
The image after hit and miss:



Principal code fragment of hit and miss:



```
hit = [(0,0), (-1,0), (0,1)]
miss = [(1,-1), (1,0), (0,-1)]
```



```
            for x in hit:
                i2 = i + x[0]
                j2 = j + x[1]
                if 0 <= i2 < w and 0 <= j2 < h:
                    if img2.getpixel((i2,j2)) == 0:
                        black = 1
            for x in miss:
                i2 = i + x[0]
                j2 = j + x[1]
                if 0 <= i2 < w and 0 <= j2 < h:
                    if img2.getpixel((i2,j2)) == 1:
                        black = 1
            if black == 1:
                imghitmiss.putpixel((i, j), 0)
```