Computer Vision hw7

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.

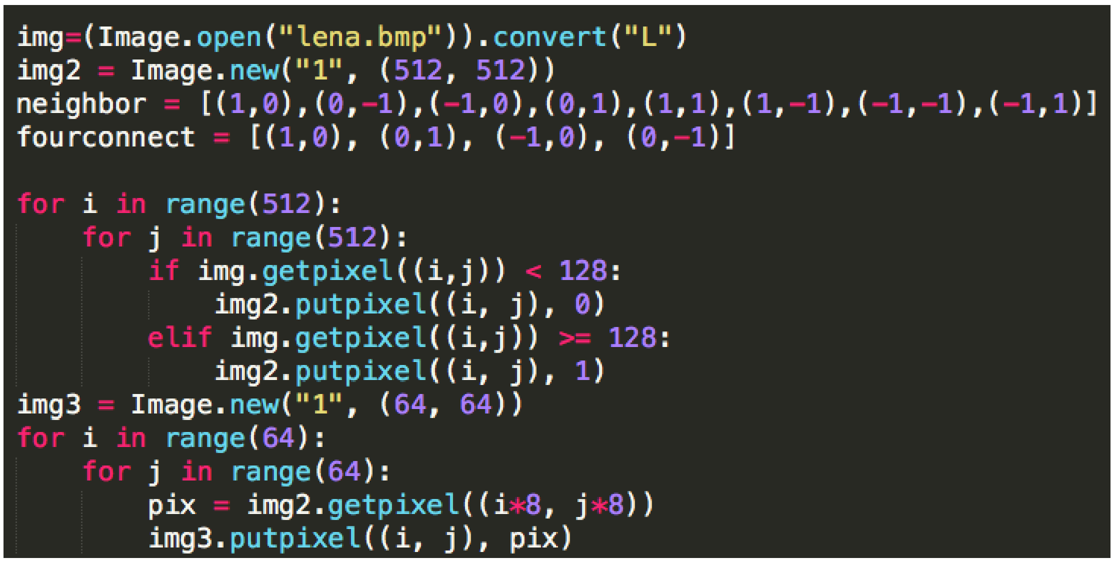
I use 4-connectivity neighborhood to complete thinning operator.

I create a list which represent all 8 points around center (0, 0), and a list represent the 4 points around center (0, 0).

Binarize the benchmark and take the topmost-left pixel as the down-sampled data:

In this part, I use the same method as I used in homework 2, And create a 64*64 image which called img3 and put all the topmost-left pixels of 8*8 block in it.

Principal code fragment:



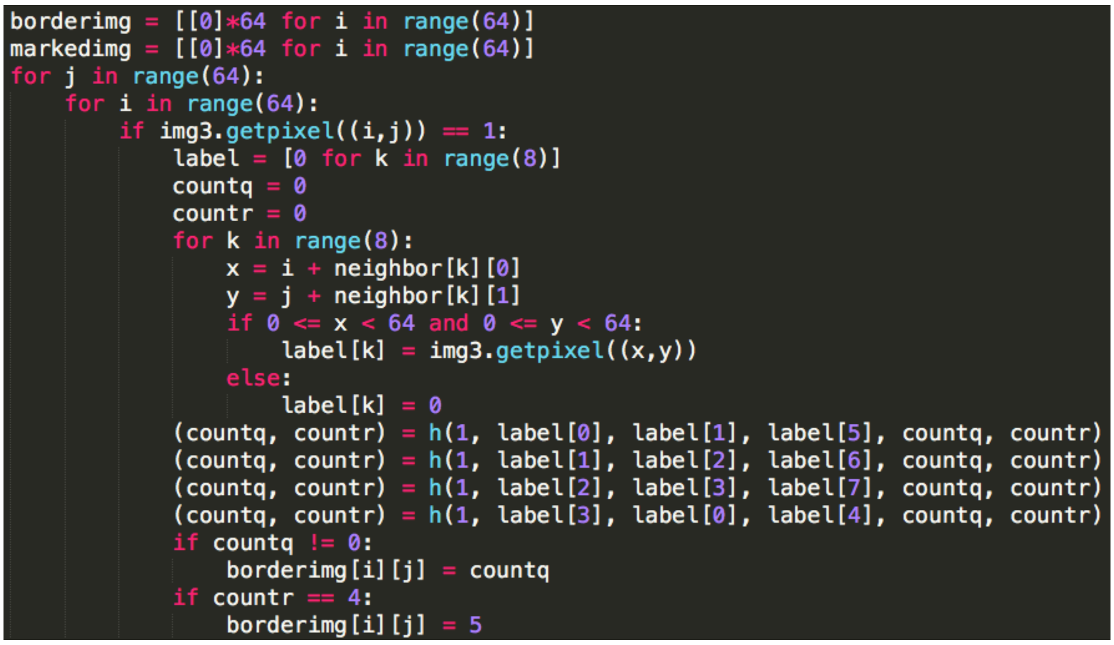
```
img=(Image.open("lena.bmp")).convert("L")
img2 = Image.new("1", (512, 512))
neighbor = [(1,0),(0,-1),(-1,0),(0,1),(1,1),(1,-1),(-1,-1),(-1,1)]
fourconnect = [(1,0), (0,1), (-1,0), (0,-1)]

for i in range(512):
    for j in range(512):
        if img.getpixel((i,j)) < 128:
            img2.putpixel((i, j), 0)
        elif img.getpixel((i,j)) >= 128:
            img2.putpixel((i, j), 1)
img3 = Image.new("1", (64, 64))
for i in range(64):
    for j in range(64):
        pix = img2.getpixel((i*8, j*8))
        img3.putpixel((i, j), pix)
```

Step 1:

I create a 64*64 array that records the border pixels. The value of the array is the yokoi number of the original image.

Principal code fragment:

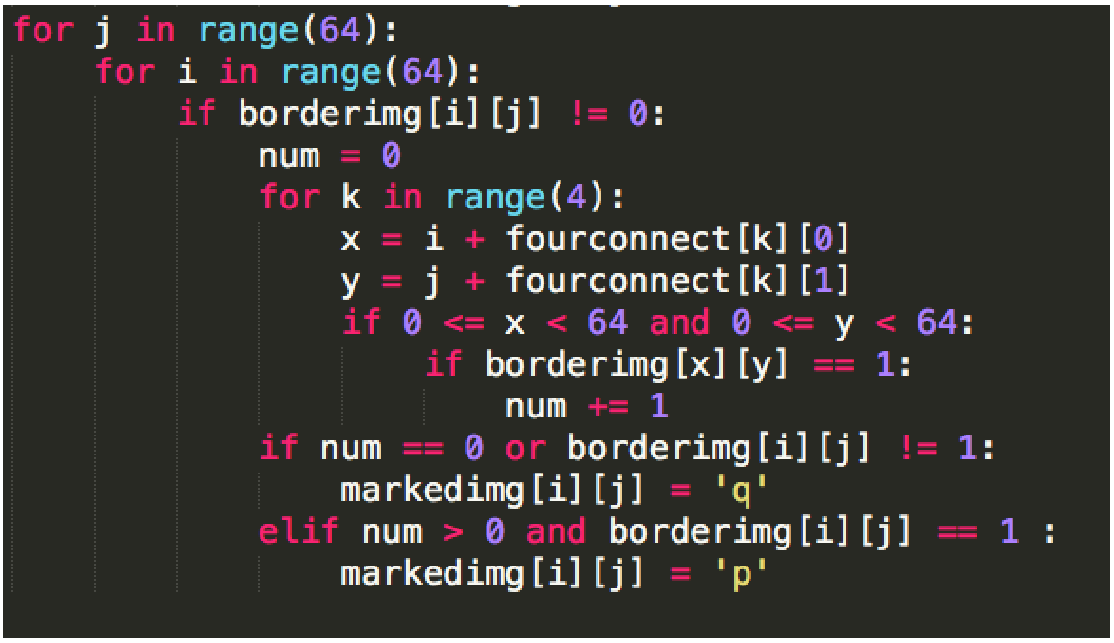


```python
borderimg = [[0]*64 for i in range(64)]
markedimg = [[0]*64 for i in range(64)]
for j in range(64):
    for i in range(64):
        if img3.getpixel((i,j)) == 1:
            label = [0 for k in range(8)]
            countq = 0
            countr = 0
            for k in range(8):
                x = i + neighbor[k][0]
                y = j + neighbor[k][1]
                if 0 <= x < 64 and 0 <= y < 64:
                    label[k] = img3.getpixel((x,y))
                else:
                    label[k] = 0
            (countq, countr) = h(1, label[0], label[1], label[5], countq, countr)
            (countq, countr) = h(1, label[1], label[2], label[6], countq, countr)
            (countq, countr) = h(1, label[2], label[3], label[7], countq, countr)
            (countq, countr) = h(1, label[3], label[0], label[4], countq, countr)
            if countq != 0:
                borderimg[i][j] = countq
            if countr == 4:
                borderimg[i][j] = 5
```

Step 2:

Now we take the border array as input, and create an array that records the output of pair relationship operator. If the pixel yokoi number is 1 and any of its neighborhood is labeled as 1, then we label the pixel as pair 'p'.

Principal code fragment:



```python
for j in range(64):
    for i in range(64):
        if borderimg[i][j] != 0:
            num = 0
            for k in range(4):
                x = i + fourconnect[k][0]
                y = j + fourconnect[k][1]
                if 0 <= x < 64 and 0 <= y < 64:
                    if borderimg[x][y] == 1:
                        num += 1
            if num == 0 or borderimg[i][j] != 1:
                markedimg[i][j] = 'q'
            elif num > 0 and borderimg[i][j] == 1 :
                markedimg[i][j] = 'p'
```

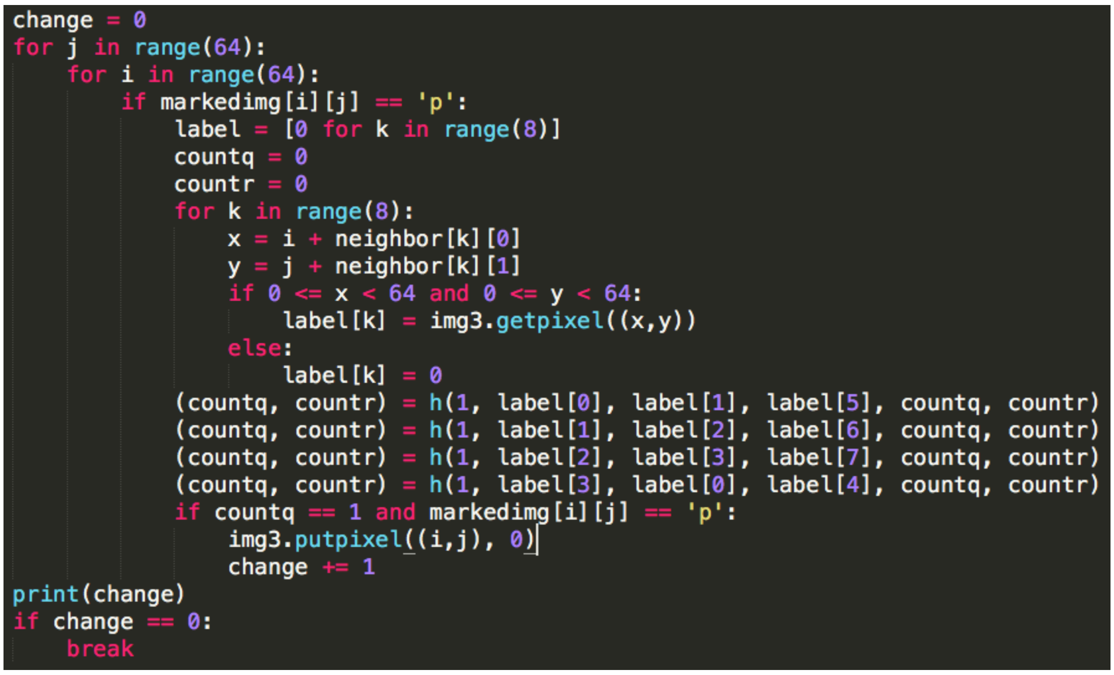
Step 3:

Then we do the connected shrink operator on the original image. If the q value of the pixel is 1 and it’s labeled as ‘p’, then we delete the pixel, the way to do connected shrink operator was told in hw6.

Step 4:

After doing the steps above, we repeat step 1 to 3 until the number of deleted pixels is 0, then the work is done.

Principal code fragment:



```
change = 0
for j in range(64):
    for i in range(64):
        if markedimg[i][j] == 'p':
            label = [0 for k in range(8)]
            countq = 0
            countr = 0
            for k in range(8):
                x = i + neighbor[k][0]
                y = j + neighbor[k][1]
                if 0 <= x < 64 and 0 <= y < 64:
                    label[k] = img3.getpixel((x,y))
                else:
                    label[k] = 0
            (countq, countr) = h(1, label[0], label[1], label[5], countq, countr)
            (countq, countr) = h(1, label[1], label[2], label[6], countq, countr)
            (countq, countr) = h(1, label[2], label[3], label[7], countq, countr)
            (countq, countr) = h(1, label[3], label[0], label[4], countq, countr)
            if countq == 1 and markedimg[i][j] == 'p':
                img3.putpixel((i,j), 0)
                change += 1
print(change)
if change == 0:
    break
```

Output of the thinning operator: