

Computer Vision Homework 6

R06922152 袁晟峻

Yokoi Connectivity Number

1. Binarize the benchmark image Lena as in HW2 by 128

```
def binarize(image):  
    width, height = image.size  
    binaryImage = Image.new('1', (width, height), 'white')  
    pixels = image.load()  
  
    for j in range(height):  
        for i in range(width):  
            # threshold 128  
            if pixels[i, j] < 128:  
                binaryImage.putpixel((i, j), 0)  
            else:  
                binaryImage.putpixel((i, j), 1)  
    return binaryImage
```

2. using 8x8 blocks as a unit, take the topmost-left pixel as the down-sampled data, Down-sample Lena from 512x512 to 64x64

```
def downSampling(image):  
    output = Image.new('1', (64, 64))  
    width, height = image.size  
    pixels = image.load()  
  
    for j in range(64):  
        for i in range(64):  
            output.putpixel((i, j), pixels[i*8, j*8])  
    return output
```

取每 8x8 矩形的左上角存到新的圖，每次跳 8 格到下一個矩形。

3. Count the Yokoi connectivity number

```
def countYokoiNumber(image):
    width, height = image.size
    output = Image.new('I', (width, height), 0)
    pixels = image.load()

    # initialize a bigger image
    bigImage = []
    for j in range(height+2):
        tmp = [0] * (width+2)
        bigImage.append(tmp)

    for i in range(1, height+1):
        for j in range(1, width+1):
            bigImage[i][j] = pixels[j-1, i-1]
```

這一部分我先建一個大一點的圖(66x66)，將原本
64x64 的 pixels 值存在裡面，外圍一圈 assign 成 0 以方
便接下來的計算。

```
for i in range(1, height+1):
    for j in range(1, width+1):
        qCounter = 0
        rCounter = 0
        # a1
        if bigImage[i][j] == 1:
            if bigImage[i][j] == bigImage[i+1][j] and bigImage[i][j] == bigImage[i][j-1] and bigImage[i][j] == bigImage[i+1][j-1]:
                rCounter += 1
            elif bigImage[i][j] == bigImage[i+1][j]:
                qCounter += 1
            # a2
            if bigImage[i][j] == bigImage[i][j-1] and bigImage[i][j] == bigImage[i-1][j-1] and bigImage[i][j] == bigImage[i-1][j]:
                rCounter += 1
            elif bigImage[i][j] == bigImage[i][j-1]:
                qCounter += 1
            # a3
            if bigImage[i][j] == bigImage[i-1][j] and bigImage[i][j] == bigImage[i][j+1] and bigImage[i][j] == bigImage[i-1][j+1]:
                rCounter += 1
            elif bigImage[i][j] == bigImage[i-1][j]:
                qCounter += 1
            # a4
            if bigImage[i][j] == bigImage[i][j+1] and bigImage[i][j] == bigImage[i+1][j] and bigImage[i][j] == bigImage[i+1][j+1]:
                rCounter += 1
            elif bigImage[i][j] == bigImage[i][j+1]:
                qCounter += 1

        if rCounter == 4:
            output.putpixel((j-1, i-1), 5)
        elif qCounter != 0:
            output.putpixel((j-1, i-1), qCounter)

return output
```

這邊真正計算 Yokoi number，對每一個 pixel 值為 1 的

pixel，其 connect number 由四個小方形決定，所以分成 a1, a2, a3, a4 4 個 if else，然後每一個去看是 r 或 q(或 s，但因為藉由 r 跟 q 就能判斷，故沒特別計算 s)，四個小方形結果若得到 4 個 r，則 connectivity number = 5，否則則看得到幾個 q，number 就是多少。

4. Result of this assignment is a 64x64 matrix.
Please **align** the matrix within 1 single A4 page (using 4-connected).

11111111	12111111111122322221	1111111111111	
15555551	115555555511 2 11 11	1155555555511	
15555551	1 2115555112 21112221	155555555551	21
15555551	1 2 155112 2221511	1555555555511	1
15555551	22 2112 22 121	15555555555511	
15555551	1 2 21 2 1 1	15555555555551	
15555551	12 1 121111 1321	155555555555511	
15111551	1322 1155551111	155555555555551	
111 1551	1 12155555511	155555555555511	
11 1551	21155555511	15511155555511	
21 1551	2 15555555111	1551 11555511	
1 1551	2 155555555511	1551 115551	1
1551	1121155555555551	1551 15511	12
1551	15555555555555511	1551 1111	111
1551	1 2221155555555555511	1151 11	1151
1551	2 22 1 15555555555555511	151 11111	1551
1551	2 1 115555555555555551	151 115551	11551
1551	2 11555555555555555111511155511		115551
1551	12 1155555555555555555555555551		155551
1551	11 221555555555555555555555555112		1155551
1551	111 22 1555555555555555555555555551 1		1555551
1551	1511 1 1251121111112111555555555111		11555551
1551	15521 1 121 1 11 1 15555555111		15555551
1551	1151 132 2 1155555111		115555551
1551	151 322 115555111 121		155555551
1551	1221 2 1555551 131		1155555551
1551	2 1 115555511 1		1155555551
1551	2 1155555551		1 15555551
1551	2 11555555551		2115555551
1551	1 115555555551		1555555551
1551	1 11511115555521 1		115555555551
1551	1 1 11111 1155511 2		155555555551
1551	131 111 15111 2		155555555551
1551	121 1121 1 111 1 2		11555555555551
1551	11 111 1 221 11 1 2		15555555555551
1551	12 1 21 121 11 1111 2		15555555555551
1551	1 12 22 151111111551 2		11555555555551
1551	1 2 1555551115511 1		15555555555551
1551	2 22 12555551 15551 1		15555555555551
1551	1 1 1555511 11511 2		1155555555555551
1551	21 155551 1 151 2		1555555555555551
1551	2 15555112 151 2		1555555555555551
1551	1 1 1 1155555511111 2		1555555555555551
1551	2 22 111511111212 211555555555555551		211555555555555551
1551	1 12 151 2 1 15555555111555551		1555555511555551
1551	1111 121 155555551 1555551		1555555551 1555551
1551	11111111 155555551 1555551		1555555551 1555551
1551	115551 155555551 1555511		211111111 155511
1551	15551 211111111 155511		2 11 115511
11521	1 12 122155511 2		11 115511
1 151	1 1 155555111 2111		15511
22 1511	1 15555555111 155111		1511
22 1511	1 15555555551 155551		1151
2 151	1 111555555555511 155511		1511
2 1521	1 155555555555511 15551 12151		
2 151	121 155555555555551 155511 1551		
2 1511	1555555555555551 115551 1511		
21 1511	11 1555555555555551 111111151		
11 151	115555555555555511 111511		
11 151	155555555555555551 151		
11 151	115555555555555551 211		
11 151	1155555555555555511 1		
11 151	155555555555555551		
11 111	1211111111111111111		