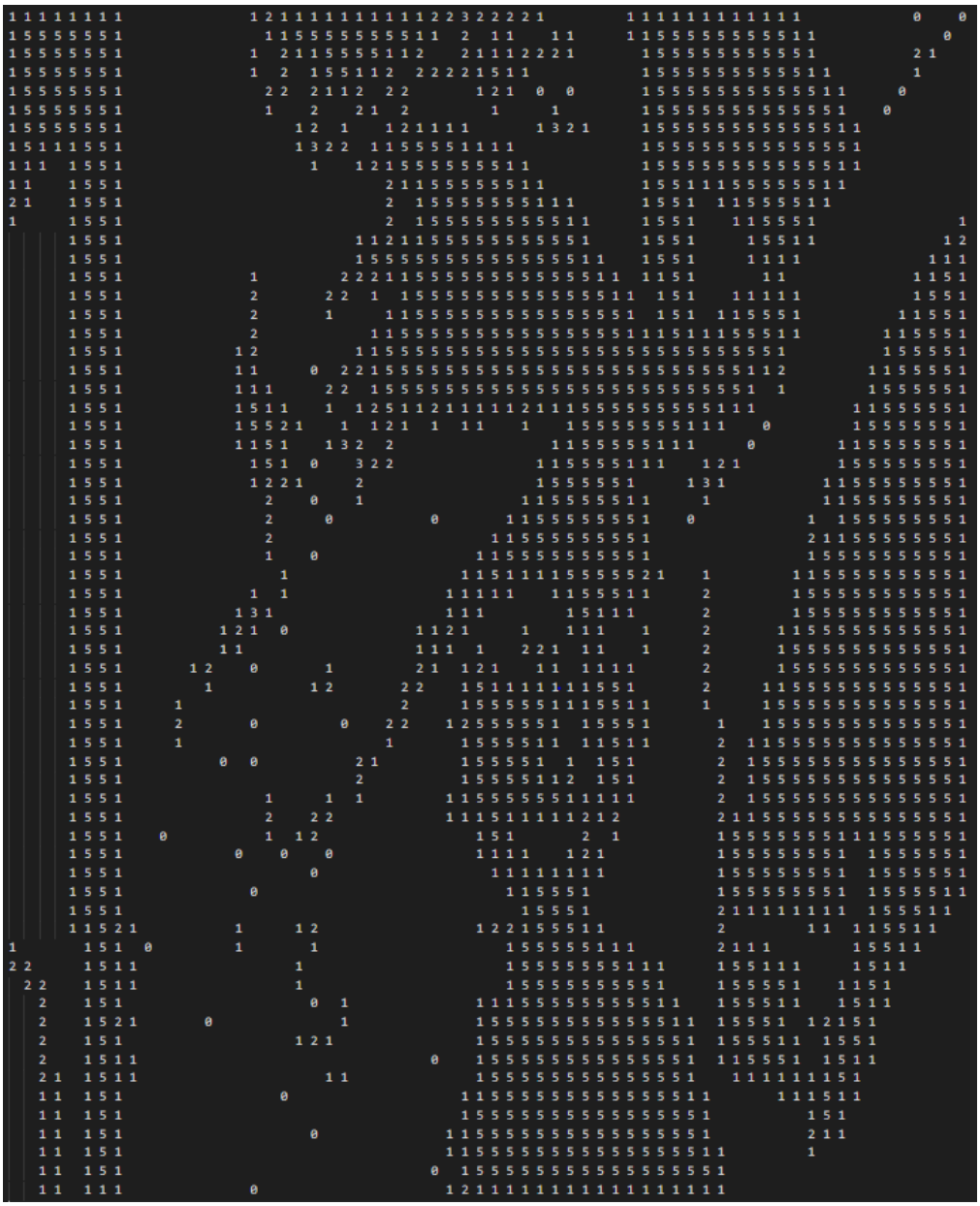
Computer Vision HW6

R08922a27 資工系 人工智慧碩士班 李吉昌

I use python 3.7 to implement all image processing requirements. Reading .bmp file by **PIL**, and then processing through **NumPy** array.

- ## 1. Results



- 2. Code fragment

```
def h(b, c, d, e):
    if b == c and (d != b or e != b):
        return 'q'
    if b == c and (d == b and e == b):
        return 'r'
    return 's'

def YokoiConnectivityNumber(bin_img, i, j):
    if i == 0:
        if j == 0:
        # top-left
            x7, x2, x6 = 0, 0, 0
            x3, x0, x1 = 0, bin_img[i][j], bin_img[i][j + 1]
            x8, x4, x5 = 0, bin_img[i + 1][j], bin_img[i + 1][j + 1]
        elif j == bin_img.shape[1] - 1:
        # top-right
            x7, x2, x6 = 0, 0, 0
            x3, x0, x1 = bin_img[i][j - 1], bin_img[i][j], 0
            x8, x4, x5 = bin_img[i + 1][j - 1], bin_img[i + 1][j], 0
        else:
        # top-row
            x7, x2, x6 = 0, 0, 0
            x3, x0, x1 = bin_img[i][j - 1], bin_img[i][j], bin_img[i][j +
1]
            x8, x4, x5 = bin_img[i + 1][j - 1], bin_img[i + 1][j],
bin_img[i + 1][j + 1]
    elif i == bin_img.shape[0] - 1:
        if j == 0:
        # bottom-left
            x7, x2, x6 = 0, bin_img[i - 1][j], bin_img[i - 1][j + 1]
            x3, x0, x1 = 0, bin_img[i][j], bin_img[i][j + 1]
            x8, x4, x5 = 0, 0, 0
        elif j == bin_img.shape[1] - 1:
        # bottom-right
            x7, x2, x6 = bin_img[i - 1][j - 1], bin_img[i - 1][j], 0
            x3, x0, x1 = bin_img[i][j - 1], bin_img[i][j], 0
            x8, x4, x5 = 0, 0, 0
        else:
        # bottom-row
            x7, x2, x6 = bin_img[i - 1][j - 1], bin_img[i - 1][j],
bin_img[i - 1][j + 1]
            x3, x0, x1 = bin_img[i][j - 1], bin_img[i][j], bin_img[i][j +
1]
            x8, x4, x5 = 0, 0, 0
    else:
        if j == 0:
            x7, x2, x6 = 0, bin_img[i - 1][j], bin_img[i - 1][j + 1]
            x3, x0, x1 = 0, bin_img[i][j], bin_img[i][j + 1]
            x8, x4, x5 = 0, bin_img[i + 1][j], bin_img[i + 1][j + 1]
        elif j == bin_img.shape[1] - 1:
            x7, x2, x6 = bin_img[i - 1][j - 1], bin_img[i - 1][j], 0
            x3, x0, x1 = bin_img[i][j - 1], bin_img[i][j], 0
```

```
            x8, x4, x5 = bin_img[i + 1][j - 1], bin_img[i + 1][j], 0
        else:
            x7, x2, x6 = bin_img[i - 1][j - 1], bin_img[i - 1][j],
bin_img[i - 1][j + 1]
            x3, x0, x1 = bin_img[i][j - 1], bin_img[i][j], bin_img[i][j +
1]
            x8, x4, x5 = bin_img[i + 1][j - 1], bin_img[i + 1][j],
bin_img[i + 1][j + 1]

    a1 = h(x0, x1, x6, x2)
    a2 = h(x0, x2, x7, x3)
    a3 = h(x0, x3, x8, x4)
    a4 = h(x0, x4, x5, x1)

    if a1 == 'r' and a2 == 'r' and a3 == 'r' and a4 == 'r':
        return 5
    else:
        return sum(np.array([a1, a2, a3, a4]) == 'q')
```

- ## 3. Brief Description

  The preprocessing processes, downsampling image from 512x512 to 64x64, and the binarization at the threshold 128 were conducted. Then, the Yokoi connectivity numbers were counted by using 4-connected. Those implementation details of the counting algorithm follow the course's lecture slides. The **h** function has been shown in the **code fragment** part.