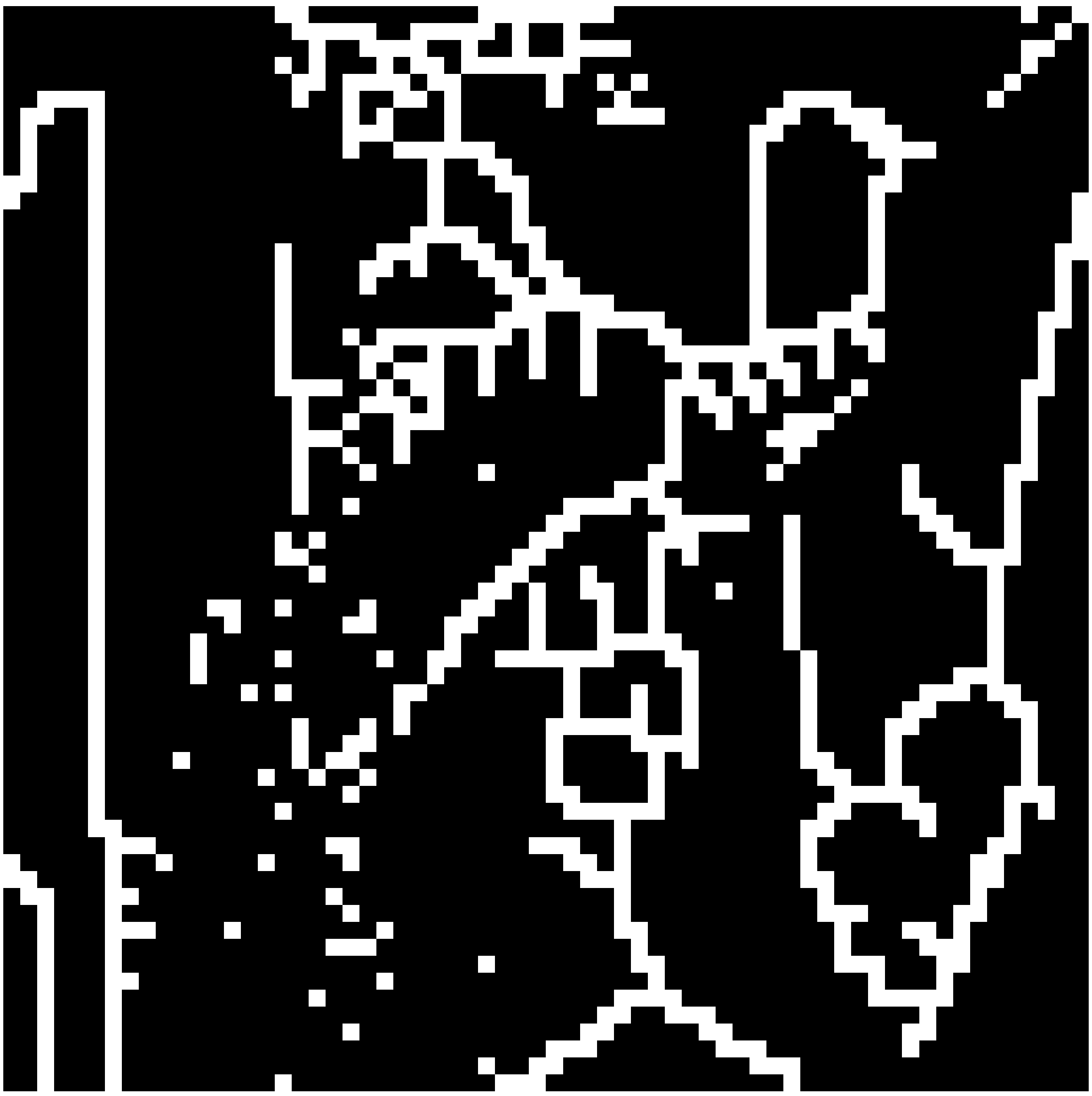
Computer Vision Homework7

by b06902034 黃柏諭

Result



Code

```python
import cv2
import numpy as np

def downSample(img):
    size = int(img.shape[0] / 8)
    res = np.zeros((size, size))
    for i in range(64):
        for j in range(64):
            res[i, j] = img[i * 8, j * 8]
    return res

def getNeighborhood(img, r, c):
    res = []
    for i in range(3):
        for j in range(3):
            nr = r + i - 1
            nc = c + j - 1
            if 0 <= nr < img.shape[0] and 0 <= nc < img.shape[1]:
                res.append(img[nr][nc])
            else:
                res.append(0)
    return [res[x] for x in [4, 5, 1, 3, 7, 8, 2, 0, 6]]

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

def f(lst):
    if lst.count('r') == 4:
        return 5
    return lst.count('q')

def getYokoiMatrix(img):
    res = np.zeros(img.shape)
    for i, j in np.ndindex(img.shape):
        if img[i, j] != 0:
            n = getNeighborhood(img, i, j)
            res[i, j] = f([h(n[0], n[1], n[6], n[2]),\
                           h(n[0], n[2], n[7], n[3]),\
                           h(n[0], n[3], n[8], n[4]),\
                           h(n[0], n[4], n[5], n[1])])
    return res

def getPairRelation(img, yokoi):
    delta = [(1, 0), (0, 1), (-1, 0), (0, -1)]
    res = np.zeros(yokoi.shape)
    for i, j in np.ndindex(yokoi.shape):
        if yokoi[i, j] != 1:
            continue
        for d in delta:
            if 0 <= i + d[0] < yokoi.shape[0] and 0 <= j + d[1] < yokoi.shape[1]\
                and yokoi[i + d[0], j + d[1]] == 1:
                res[i, j] = 1
    return res

def connectShrink(img):
    flg = True
    res = np.copy(img)
    yokoi = getYokoiMatrix(img)
    pair_relation = getPairRelation(img, yokoi)
    for i, j in np.ndindex(res.shape):
        if pair_relation[i, j] > 0:
            n = getNeighborhood(res, i, j)
            count = f([h(n[0], n[1], n[6], n[2]),\
                       h(n[0], n[2], n[7], n[3]),\
                       h(n[0], n[3], n[8], n[4]),\
                       h(n[0], n[4], n[5], n[1])])
            if count == 1:
                res[i, j] = 0
                flg = False
    return flg, res

image = cv2.imread("lena.bmp", cv2.IMREAD_GRAYSCALE)
binary = (image >= 128) * np.full(image.shape, 255)
thinning = downSample(binary)
cv2.imwrite("f_thinning.bmp", thinning)
flg = False
iterate = 0
while flg == False:
    flg, thinning = connectShrink(thinning)
cv2.imwrite("1_thinning.bmp", thinning)
```