CV HW7

B07902072 資工四 陳光裕

Usage

```
python3 main.py
```

corner

```
corner = [[[1,0],[1,-1],[0,-1]],[[0,-1],[-1,-1],[-1,0]],[[-1,0],[-1,1],[0,1]],[[0,1],[1,1],[1,0]]]
```

流程敘述

- 首先跟上次作業一樣，做binary跟down sampling

```
def binary(img):
    width, height = img.shape[:2]
    for i in range(width):
        for j in range(height):
            px = img[i,j]
            if px < 128:
                img[i,j] = 0
            else:
                img[i,j] = 255
    return img

def down_sampling(img):
    width, height = 512, 512
    new_img = np.zeros((64,64))
    for i in range(64):
        for j in range(64):
            new_img[i][j] = img[i*8][j*8] # left top point = new point
    return new_img # new img is a 64*64 array
```

- 對於每次的thinning，先複製一份當作compare的基底，然後按照步驟，對圖片做yokoi operator後做pair relationship operator得到一個marked image

- yokoi:

```
def valid(x, y, w, h):
    return x >= 0 and x < w and y >= 0 and y < h

def yokoi_conn(img, x, y, w, h, cor):
    new_x = x+cor[0]
    new_y = y+cor[1]
    if valid(new_x, new_y, w, h):
        return img[new_x][new_y]
    return 0

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

def yokoi(img, corner):
    width, height = img.shape[:2]
    result = np.zeros((64,64))
    for i in range(width):
        for j in range(height):
            if img[i][j] == 0:
                continue
            else:
                # check every corner's h_equation then check qrs's number
                cnt = []
                for cor in corner:
                    # four corner test
                    b = img[i][j]
                    c = yokoi_conn(img, i, j, width, height, cor[0])
                    d = yokoi_conn(img, i, j, width, height, cor[1])
                    e = yokoi_conn(img, i, j, width, height, cor[2])
                    cnt.append(h_equation(b, c, d, e))
                # r = 4 : 5 , else print q's number
                if cnt.count('r') == 4:
                    result[i,j] = 5
                else:
                    if cnt.count('q') == 0:
                        result[i,j] = 0
                    else:
                        result[i,j] = cnt.count('q')
    return result
```

- pair relationship:

```
def pair_h(a):
    if a == 1:
        return 1
    return 0

def pair_relationship_operator(yokoi):
    result = np.zeros((66,66))
    new_yokoi = np.zeros((66,66))
    for i in range(64):
        for j in range(64):
            new_yokoi[i+1,j+1] = yokoi[i,j]
    for i in range(1,65):
        for j in range(1,65):
            if new_yokoi[i,j] == 1:
                h1 = new_yokoi[i+1,j]
                h2 = new_yokoi[i,j+1]
                h3 = new_yokoi[i-1,j]
                h4 = new_yokoi[i,j-1]
                if pair_h(h1) == 1 or pair_h(h2) == 1 or pair_h(h3) == 1 or pair_h(h4) == 1: # p
                    result[i,j] = 1
                else: #q
                    result[i,j] = 2
            elif new_yokoi[i,j] > 1:
                result[i,j] = 2
    return result
```

- 然後對於marked image裡面為1的點(也就是P)做connected shrink operator，最後跟基底比對，若沒有變則退出迴圈並印出結果

```
def new_h_equation(b, c, d, e):
    if b == c and ( b != d or b != e ):
        return 1
    return 0

def thinning_operate(img, corner):
    width, height = img.shape[:2]
    new_img = np.copy(img)
    while(1):
        compare_base = np.copy(new_img)
        yokoi_res = yokoi(new_img, corner)
        marked = pair_relationship_operator(yokoi_res)
        for i in range(width):
            for j in range(height):
                if marked[i+1,j+1] == 1: # p
                    cnt = []
                    for cor in corner:
                        b = new_img[i][j]
                        c = yokoi_conn(new_img, i, j, width, height, cor[0])
                        d = yokoi_conn(new_img, i, j, width, height, cor[1])
                        e = yokoi_conn(new_img, i, j, width, height, cor[2])
                        cnt.append(new_h_equation(b, c, d, e))
                    if cnt.count(1) == 1:
                        new_img[i][j] = 0
        compare = (new_img == compare_base)
        if compare.all():
            break
    return new_img
```

Result