HW7 report

R10922082 林育駿

本次作業中使用 python opencv 幫助圖片的 I/O 一些處理部分，在有 lena.bmp 同目錄下執行 R10922082_HW7.py 會生{1 - 7}_iteration_lena.bmp 七張分別是七個 iteration 後 thinning 後的 lena。

Write a program which does thinning on a downsampled image (lena.bmp).

- Binarized & Downsample & Yokoi connectivity number matrix 如上次 HW6 作法相同
- Pair relationship operator

H function: (m="1", means "edge" in Yokoi)

- $h(a,m) = \begin{cases} 1, if\ a = m \\ 0, otherwise \end{cases}$

Output:

- $y = \begin{cases} q, if\ \sum_{n=1}^{4} h(x_n, m) < 1\ or\ x_0 \neq m \\ p, if\ \sum_{n=1}^{4} h(x_n, m) \geq 1\ and\ x_0 = m \end{cases}$

如公式所描述的在計算出的 yokoi connectivity number matrix 上再做運算得出新的一張 q 和 p 的 pair relationship 圖找出用來進行 shrink operation 的邊界。

```python
def pairOp(yokoim): # let q =1 p =2
    (imgHeight, imgWidth) = yokoim.shape[:2]
    nyokoim = np.zeros((imgWidth +2,imgHeight +2)) # for yokoi with padding
    pairm = np.zeros((imgWidth,imgHeight)) # for yokoi with padding
    for i in range(imgHeight):
        for j in range(imgWidth):
            nyokoim[i+1][j+1] = yokoim[i][j]
    # do pair relationship algorithm
    for i in range(imgHeight):
        for j in range(imgWidth):
            if nyokoim[i+1][j+1] != 1:
                pairm[i][j] = 1
            else:
                if nyokoim[i+1][j+1] == nyokoim[i+1+1][j+1] or nyokoim[i+1][j+1] ==nyokoim[i+1][j+1+1] or nyokoim[i+1][j+1] =
                    pairm[i][j] = 2
                else:
                    pairm[i][j] = 1
    return  pairm
```

- Connected shrink operator

H function: (yokoi corner => “q”)

- $$h(b,c,d,e) = \begin{cases} 1, if\ b = c\ and\ (d \neq b\ or\ e \neq b) \\ 0, otherwise \end{cases}$$

Output:

- $$f(a_1,a_2,a_3,a_4,x) = \begin{cases} g, if\ exactly\ one\ of\ a_n = 1, n = 1{\sim}4 \\ x, otherwise \end{cases}$$

如公式所描述以得出的 pair relationship 圖再和原圖做 shrink operation，其中用到的 h 和 f function 和計算 yokoi matrix 時類似但改變一些條件，將整個流程完成為一個 iteration 重複七次得到最後 tinning 的圖。

```python
def shrinkOp(pairm):
    global labelm
    (imgHeight, imgWidth) = pairm.shape[:2]
    # do algorithm
    for i in range(imgHeight):
        for j in range(imgWidth):
            if pairm[i][j] ==2: # is p
                if f2(i+1,j+1):
                    labelm[i+1][j+1] = 0
```

新 h function:

```
def h2(y,x,c,d,e):
    global labelm
    (cx,cy) = index2pos(c)
    (dx,dy) = index2pos(d)
    (ex,ey) = index2pos(e)

    if labelm[y][x] == 1: # center exist
        # checl b != c
        if labelm[y][x] != labelm[y+cy][x+cx]:
            return 0 #s
        else:
            if labelm[y][x] ==labelm[y+ey][x+ex] and labelm[y+dy][x+dx] ==labelm[y][x]:
                return 0 #r
            else:
                return 1 #q
```

新 f function:

```
def f2(y,x):

    a1 = h2(y,x,1,6,2)
    a2 = h2(y,x,2,7,3)
    a3 = h2(y,x,3,8,4)
    a4 = h2(y,x,4,5,1)

    if a1+a2+a3+a4 == 1:
        return True # change to background
    else:
        return False
```