

CV Homework 7

Abstract

本次作業中，我們將二值化的圖片取樣為 64x64 大小，並在使用 4-connected neighborhood detection的情況下，對其以 yokoi connectivity number + pair relationship operator 的方式施作thining。以上操作除輸入、輸出以外，不得直接套用現成套件。

Implementation

- Programming Language: Python3
- Python Package: opencv-python, matplotlib, numpy, copy
- Execution: python3 hw7-main.py

在執行時，請務必將 lena.bmp 放置在和 hw7-main.py 相同的檔案目錄下，並確認檔名相同。opencv-python用於讀取和寫出圖片，matplotlib僅用於runtime時即時顯示和儲存圖片，須配合jupyter使用，在此不贅述，而最後的copy和numpy則用於純計算上。

Pair Relationship Operator

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

def pair_relationship(yona):
    after = np.zeros((64, 64))
    blk = [(0, 1), (-1, 0), (0, -1), (1, 0)]
    for r in range(after.shape[0]):
        for c in range(after.shape[1]):
            ps = []
            for ofst in blk:
                if (r+ofst[0] >= 0 and r+ofst[0] < yona.shape[0]) and (c+ofst[1] >= 0 and c+ofst[1] < yona.shape[1]):
                    ps.append(yona[r + ofst[0]][c + ofst[1]])
                else:
                    ps.append(0)

            ps = [ pair_h(p, 1) for p in ps]
            if (ps.count(1) < 1 or yona[r][c] != 1):
                after[r][c] = 1 #q
            else:
                after[r][c] = 0 #p
    return after
```

H function: ($m=1$, means “edge” in Yokoi)

$$\bullet \quad h(a, m) = \begin{cases} 1, & \text{if } a = m \\ 0, & \text{otherwise} \end{cases}$$

Output:

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

此處的 pair relationship operator 是搭配 yokoi connectivity number 使用。本實作主要參考作業說明的PDF中所附的公式實作，遍歷以計算出yokoi number圖片中的所有像素，並和其的四連通鄰近像素進行pair_h函數操作(即H function)，若四個pair_h函數的結果有至少一者為1且中心像素的yokoi number也為1便回傳p(此處以0表示)，否則回傳q(此處以1表示)。

Thinning Operator

```

def thinning(downa):
    original = copy.deepcopy(downa)
    while True:
        yona = yokoi(original)
        prna = pair_relationship(yona)
        thna = np.copy(original)
        for r in range(original.shape[0]):
            for c in range(original.shape[1]):
                if thna[r][c] == 255:
                    if get_yokoi(thna, r, c) == 1 and prna[r][c] == 0:
                        thna[r][c] = 0

        if (thna == original).all():
            break

        original = copy.deepcopy(thna)
    return original

```

- H function: (yokoi corner => “q”)

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

- Output:

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

對sample後的圖片計算yokoi number, 並將得出的結果再丟入pair_relationship中取得marked image。我們複製一份original的備份來進行處理。此後我們逐一對此備份的像素計算其yokoi number, 若其為1 (edge)且該點為marked(p, 此處以0表示), 則將此備份中該像素轉為背景色即黑色(0)。掃完整張備份後跟original進行比較, 若無任何變化表示處理完畢, 若有變化則以此備份覆蓋掉original, 並再來一次整套流程。處理完後的結果貼於下方, 因排版因素(主要是圖片太小)故有調整顯示尺寸。

