R09922190 王祥任

CV Homework 5

Abstract

本次作業中，我們要使用 octagonal 3-5-5-5-3 kernel 來對原圖實作 Gray Scale 的 (1) Dilation (2) Erosion (3) Opening (4) Closing，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 hw5-main.py`

在執行時，請務必將 lena.bmp 放置在和 hw5-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。

Dilation and Erosion

```python
def dilation(lena, kernel):
    after = copy.deepcopy(lena)
    for r in range(lena.shape[0]):
        for c in range(lena.shape[1]):
            neigh = []
            for k in kernel:
                if (r - k[0] >= 0 and r - k[0] < lena.shape[0] and
                    c - k[1] >= 0 and c - k[1] < lena.shape[1]):
                    neigh.append(lena[r - k[0]][c - k[1]])
            after[r][c] = np.max(neigh)
    return after
dina = dilation(lena, oct_kernel)
plt.imshow(dina, cmap='gray', vmin=0, vmax=255)
cv2.imwrite('dilation-lena.bmp',dina)
cv2.imwrite('dilation-lena.png',dina)

def erosion(lena, kernel):
    after = copy.deepcopy(lena)
    for r in range(lena.shape[0]):
        for c in range(lena.shape[1]):
            neigh = []
            for k in kernel:
                if (r + k[0] >= 0 and r + k[0] < lena.shape[0] and
                    c + k[1] >= 0 and c + k[1] < lena.shape[1]):
                    neigh.append(lena[r + k[0]][c + k[1]])
            after[r][c] = np.min(neigh)
    return after
dina = erosion(lena, oct_kernel)
plt.imshow(dina, cmap='gray', vmin=0, vmax=255)
cv2.imwrite('erosion-lena.bmp',dina)
cv2.imwrite('erosion-lena.png',dina)
```

兩種做法的邏輯基本相同，對圖片上的某點像素套用Kernel範圍內的所有鄰像素(只考慮邊界內)，並找出這群像素中最大(Dilation)/最小(Erosion)的值，則該點像素套用變形後的值就是該最大值(Dilation)/最小值(Erosion)。依此作法遍歷整張圖片即可。最後成果貼於報告最後的圖組中。

Opening and Closing

```python
def opening(lena, kernel):
    return dilation(erosion(lena, kernel), kernel)
opna = opening(lena, oct_kernel)
plt.imshow(opna, cmap='gray', vmin=0, vmax=255)
cv2.imwrite('opening-lena.bmp',opna)
cv2.imwrite('opening-lena.png',opna)

def closing(lena, kernel):
    return erosion(dilation(lena, kernel), kernel)
clna = closing(lena, oct_kernel)
plt.imshow(clna, cmap='gray', vmin=0, vmax=255)
cv2.imwrite('closing-lena.bmp',clna)
cv2.imwrite('closing-lena.png',clna)
```

$$f \circ k = (f \ominus k) \oplus k \qquad f \bullet k = (f \oplus k) \ominus k$$

有了Dilation和Erosion後，我們便可套用公式來實作Opening和Closing。Opening便是先作Erosion後再使用同樣的kernel對Erosion後的結果作dilation，而Closing則是順序相反過來。成果同樣貼於報告最後的圖組中。

Result

底下圖組為套用變形後的實作成果，因排版因素，有略為調整圖片的尺寸。

Dilation

Erosion

	
Opening	Closing