HW5 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，執行 R10922082_HW5.py 會生成 dilation_lena.bmp, erosion_lena.bmp, opening_lena.bmp 和 closing_lena.bmp 四張圖片對應四個小題。

(a) Dilation

Lena 灰階原圖直接做 dilation，方法和 binary 時一樣但做一些定義的擴充，將 kernel 對到的每個點和原圖 pixel 值疊加起來，找尋結果最大值作為該點輸出的 pixel 值，因為題目中使用的 kernel 值都是 0 所以相當於直接選取 kernel 範圍內原圖最大值為輸出。

```
# init a white canvas
nimg = np.zeros(shape=(imgWidth,imgHeight,3))
for i in range(imgHeight):
    for j in range(imgWidth):
        max = np.array([0,0,0]) # temp value for every pixel in order to find max value under kernel
        for a in range(k_h):
            for b in range(k_w):
                if kernel[a][b] ==1 and i+a-centerY < imgHeight and j+b-centerX < imgWidth and i+a-centerY >=0 and j+b-centerX >=0 :
                    if img[i+a-centerY][j+b-centerX][0] > max[0]:
                        max = img[i+a-centerY][j+b-centerX]
        nimg[i][j] =  max
```

(b) Erosion

Lena 灰階原圖直接做 erosion，方法和 binary 時一樣但做一些定義的擴充，將 kernel 對到的每個點和原圖 pixel 值相減，找尋結果最小值作為該點輸出的 pixel 值，正好也是因為題目中的 kernel 值都是 0 所以相當於直接選取 kernel 範圍內原圖最小值為輸出，實作方法就會很像直接使用 dilation 但取的改變 max 為 min。

```
nimg = np.zeros(shape=(imgWidth,imgHeight,3))
for i in range(imgHeight):
    for j in range(imgWidth):
        min = np.array([255,255,255])
        for a in range(k_h):  # kernel operation
            for b in range(k_w):
                if kernel[a][b] ==1 and i+a-centerY < imgHeight and j+b-centerX < imgWidth and i+a-centerY >=0 and j+b-centerX >=0
                    if img[i+a-centerY][j+b-centerX][0] < min[0]:
                        min = img[i+a-centerY][j+b-centerX]
        nimg[i][j] =  min
```

(c) Opening

這部分的作法完全就和 binary 時相同，先 erosion 再 dilation，只是實作方法中的 dilation 和 erosion 就是使用前兩題的新方法。

```
def openOp(img):        # erosion then dilation
    # print("do open")
    kernel =[[0,1,1,1,0],
             [1,1,1,1,1],
             [1,1,1,1,1],
             [1,1,1,1,1],
             [0,1,1,1,0]]
    e_img = erosion(img,kernel,5,5,2,2)
    o_img = dilation(e_img,kernel,5,5,2,2,)
    cv2.imwrite("opening_lena.bmp", o_img) #write the output of image
```

(d) Closing

同樣使用跟 binary 處理時一樣作法，先 dilation 再 erosion。

```
def closeOp(img):       # dilation then erosion
    # print("do close")
    kernel =[[0,1,1,1,0],
             [1,1,1,1,1],
             [1,1,1,1,1],
             [1,1,1,1,1],
             [0,1,1,1,0]]
    d_img = dilation(img,kernel,5,5,2,2)
    c_img = erosion(d_img,kernel,5,5,2,2)
    cv2.imwrite("closing_lena.bmp", c_img) #write the output of image
```