HW4
r10944013 網媒所碩一 馮啟倫

HW4-1:
目標：對照片做dilation

演算法：
給定一個kernel，掃描整張圖片，一旦圖片的某個點的數值為255，就對那張圖片對應的Kernel區域進行補值

Code segment

```
#hw4-1
def dilation(img_binary, kernel):
    print("Process for dilation image...")
    img_dilation = np.zeros((512,512))
    for row in range(img_binary.shape[0]):
        for col in range(img_binary.shape[1]):
            if img_binary[row][col] == 255: #kernel中心點跟img_binary對齊
                for y,x in kernel:
                    if (y+row) >=0 and (y+row) < 512 and (x+col) >= 0 and (x+col) <512:
                        img_dilation[y+row][x+col] = 255
    print("Finish")
    return img_dilation
```

Result picture:

HW4-2:
目標：對照片做erosion

演算法：
給定一個Kernel，掃描整張照片，若當前對應Kernel形狀的區域，每一點都有值(255)的話，就賦予Kernel中心點對應的圖片區域255的值

Code segment:

```python
#hw4-2
def erosion(img_binary, kernel):
    print("Process for erosion image...")
    img_erosion = np.zeros((512,512))
    for row in range(img_binary.shape[0]):
        for col in range(img_binary.shape[1]):
            flag_interset = True
            for y,x in kernel:
                if (y+row) <0 or (y+row) >= 512 or (x+col) < 0 or (x+col) >= 512:
                    flag_interset = False
                    break
                if img_binary[y+row][x+col] != 255:
                    flag_interset = False
                    break
            if flag_interset:
                img_erosion[row][col] = 255
    print("Finish")
    return img_erosion
```

Result picture:

HW4-3:

目標：對照片做opening

演算法：
先對圖片做erosion，再做dilation

Code segment:

```
#hw4-3
#先做erosion, 再做dilation
def opening(img_binary, kernel):
    print("Process for opening image...")
    return dilation(erosion(img_binary, kernel), kernel)
```

Result picture:

HW4-4:

目標：對照片做closing

演算法：
先對圖片做dilation，再做erosion

Code segment:

```
#hw4-4
#先做dilation, 再做erosion
def closing(img_binary, kernel):
    print("Process for closing image...")
    return erosion(dilation(img_binary, kernel), kernel)
```

Result picture:

HW4-5:

目標：對照片做hit and miss

演算法：

根據數學表達式，先取兩個L型的Kernel，分別對binary image與binary image的補集做erosion，再對這兩個image做交集

Code segment:

```
#hw4-5
#hit and miss: L kernel 是foreground, J kernel是backgound
def hitandmiss(img_binary):
    print("Process for hitandmiss image...")
    kernel_J = [[0,-1], [0,0], [1,0]]
    kernel_K = [[-1,0], [-1,1], [0,1]]
    img_c = np.zeros((512,512))
    for row in range(img_binary.shape[0]):
        for col in range(img_binary.shape[1]):
            if img_binary[row][col] == 255:
                img_c[row][col] = 0
            else:
                img_c[row][col] = 255

    img_hit = erosion(img_binary, kernel_J)
    img_miss = erosion(img_c, kernel_K)
    img_hitandmiss = np.zeros((512,512))
    for row in range(img_binary.shape[0]): #intersect
        for col in range(img_binary.shape[1]):
            if img_hit[row][col] == 255 and img_miss[row][col] == 255:
                img_hitandmiss[row][col] = 255
    print("Finish")
    return img_hitandmiss
```

Result picture: