CV hw4 / 電機所R06921082 陳與賢

Description:

利用python來處理bmp檔，進行dilation、erosion、opening、closing、hit and miss共5種操作。

（一）Dilation

- *Algorithm*

首先會把kernel存成與原點的相對距離，比方說octonal 3-5-5-5-3 kernel 會存成如下：

```
68 kernel = []
69 for i in range(5):
70     for j in range(5):
71         temp = [i-2, j-2]
72         kernel.append(temp)
73 kernel.remove([-2,-2])
74 kernel.remove([-2,2])
75 kernel.remove([2,-2])
76 kernel.remove([2,2])
```

再來先把輸出圖的pixels都填0(黑色)，開始跑dilation
在原圖上的pixels值若是255(白色)才需要檢查，首先先看kernel是否超出圖的大小，若沒有則將kernel的位置都填255(白色)，code如下：

```
# initial
for x in range(dilation_output.width):
    for y in range(dilation_output.height):
        dilation_pixels[x, y] = 0

for x in range(img_input.width):
    for y in range(img_input.height):
        if pixels_input[x, y] == 255:
            flag = True
            # check boundry
            for z in range(len(kernel)):
                x1 = x + kernel[z][0]
                y1 = y + kernel[z][1]
                if x1 < 0 or x1 > 511 or y1 < 0 or y1 > 511:
                    flag = False
                    break
            if flag == True:
                for z in range(len(kernel)):
                    x1 = x + kernel[z][0]
                    y1 = y + kernel[z][1]
                    dilation_pixels[x1, y1] = 255
```

- *Result*

（二）Erosion

- *Algorithm*

首先會把kernel存成與原點的相對距離，同dilation的做法。
再來開始跑erosion，檢查kernel是否超出範圍以及kernel對應到原圖位置的pixels值是否為255(白色)，若沒超出範圍且對應的點均為白色，則原點塗白，code如下：

```
for x in range(img_input.width):
    for y in range(img_input.height):
        flag = True
        # check boundry
        for z in range(len(kernel)):
            x1 = x + kernel[z][0]
            y1 = y + kernel[z][1]
            if x1 < 0 or x1 > 511 or y1 < 0 or y1 > 511:
                flag = False
                break
            if pixels_input[x1, y1] != 255:
                flag = False
                break
        if flag == True:
                erosion_pixels[x, y] = 255
```

- *Result*

（三）Opening、Closing

- *Algorithm*

Opening就是先做erosion，發現侵蝕過頭了所以再做dilation
而Closing就事先做dilation，發現膨脹過頭了所以再做erosion
因為我erosion跟dilation都寫成function的型式，所以直接call即可
code如下：

```
output = dilation(kernel, "binary.bmp")
output.save("dilation.bmp")
output = erosion(kernel, "binary.bmp")
output.save("erosion.bmp")
output = dilation(kernel, "erosion.bmp")
output.save("opening.bmp")
output = erosion(kernel, "dilation.bmp")
output.save("closing.bmp")
```

- *Result*

opening

closing

（四）Hit and miss

- *Algorithm*

首先直接根據原圖來判定要給255還是0來把lena的補圖做出來，
code如下：

```
img_input = Image.open("binary.bmp")
pixels_input = img_input.load()
complement_output = Image.new(img_input.mode, img_input.size)
complement_pixels = complement_output.load()
for x in range(img_input.width):
    for y in range(img_input.height):
        if pixels_input[x, y] == 0:
            complement_pixels[x, y] = 255
        else:
            complement_pixels[x,y] = 0
complement_output.save("complement.bmp")
```

再來kernel一樣是直接給好和原點的相對位置

```
kernel = [[0,0],[-1,0],[0,1]]
kernel2 = [[0,-1],[1,-1],[1,0]]
```

接著各自call erosion function

```
output1 = erosion(kernel, "binary.bmp")
output1_pixels = output1.load()
output2 = erosion(kernel2, "complement.bmp")
output2_pixels = output2.load()
```

最後根據兩個圖的pixel值做and運算來決定輸出圖的pixels值要不要給255

```
for x in range(output1.width):
    for y in range(output1.height):
        if output1_pixels[x, y] == 255 and output2_pixels[x, y] == 255:
            hnm_pixels[x, y] = 255
```

- *Result*