HW2

- **A discription of your homework**

  Programming language used: Python 2.7

  Library used: Numpy, PIL, Scipy.misc, matplotlib.pyplot, matplotlib.patches

  Plotting tool: Excel 2013

- **Your parameters**

  i: row

  j: column

  tem: 用於儲存每個像素的灰階數值 0~255

  num: 用於儲存各灰階數值的像素數

  label: 物件標號

  corner: 用於儲存水平連接且為垂直連接的像素的位置

  wrong, large: 需要被代換掉的 label

  right, small: 應該被改成的 label

  value: 統計各 label 的 pixel 數

  leftvalue: 多於(或等於)500 pixels 的 label

  xpos, ypos: 某物件的所有像素的 j(或 i)總和

  xcen, ycen: x(或 y)方向的重心

  xmin, ymin: x(或 y)方向的最小值

  xmax, ymax: x(或 y)方向的最大值

- **The algorithm you used**

1. a binary image (threshold at 128)

   Threshold: 128

   灰階像素 >= 128 → 白(1)

   灰階像素 < 128 → 黑(0)

2. a histogram

   (1) 計算各灰階數值的像素數:

   k：灰階值，num[k]：灰階值為 k 的像素數

   使用迴圈跑過該圖片所有像素，該像素為 k 者，則在 num[k]加一

   (2) 輸出成.txt 檔

   (3) 將資料放入 excel 畫長條圖

3. connected components

使用 4-connected 定義 connected components

標號物件的方法：local table

(1) 由左上往右下標號

(2) 只有水平連接者：

延續左邊的標號

(3) 只有垂直連接者：

延續上面的標號

(4) 既為水平連接又為垂直連接者：

(a) 選擇左邊和上面標號中較小者

(b) 記錄左邊和上面的標號，較小者新增到 right，較大者新增到 wrong

(c) 若該列其他像素的標號，為剛剛新增到 wrong 的編號，需改為剛剛新增到 right 的編號

(5) 若 right 裡某元素和 wrong 裡某元素一樣，則更改 right 裡的標號，為方便說明請見下例：

原本標號為 7 者應該改成 5，但因為 5 要改成 2，所以標號 7 應該要改成 2 而非 5。

right	wrong
2	5
~~5~~ 2	7

(6) 從右下往左上倒著回來把該改的標號改好

(7) 計算各 label 的像素數，方法如 2. (1)，只保留多於(或等於)500 像素的 label

(8) 重新標號多於(或等於)500 像素的 label

(9) 計算各物件的重心及邊界，再畫 bounding box

重心：$x_{center} = \frac{1}{n}\sum_{i=1}^{n} x_i$，$y_{center} = \frac{1}{n}\sum_{i=1}^{n} y_i$

邊界：找 x,y 最大最小值

- **Principal code fragment**

```
def bi(x):
    tem = np.zeros(x.shape)
    for i in range(x.shape[0]):
        for j in range(x.shape[1]):
            if x[i][j]<128:
                tem[i][j] = 0
            else:
                tem[i][j] = 1
    return tem

def his(x):
    num = np.zeros([256])
    for i in range(x.shape[0]):
        for j in range(x.shape[1]):
            num[int(x[i][j])] = num[int(x[i][j])]+1
    f = open('his.txt','w')
    for k in range(len(num)):
        f.write(str(k)+' '+str(num[k])+'\n')
    f.close()
```

```
def con(x):
    tem = np.zeros(x.shape)
    label = 0
    corner = []
    right = []
    wrong = []
    for i in range(x.shape[0]):
        for j in range(x.shape[1]):
            if x[i][j]==1: # white area -> object
                if label!=0: # not first label
                    if i!=0 and j==0: # not first row but first column
                        if x[i-1][j] ==1: # connected
                            tem[i][j] = tem[i-1][j]
                        else: # isolated
                            label = label+1
                            tem[i][j] = label

                    if j!=0 and i==0: # not first column but first row
                        if x[i][j-1] ==1: # connected
                            tem[i][j] = tem[i][j-1]
                        else: # isolated
                            label = label+1
                            tem[i][j] = label

                    if i!=0 and j!=0: # neither first row nor first column
                        # only horizontally-connected
                        if x[i-1][j] !=1 and x[i][j-1] ==1:
                            tem[i][j] = tem[i][j-1]
                        # only vertically-connected
                        elif x[i-1][j] ==1 and x[i][j-1] !=1:
                            tem[i][j] = tem[i-1][j]
                        # horizontally-connected and vertically-connected
                        elif x[i-1][j] ==1 and x[i][j-1] ==1:
                            small = min(tem[i-1][j],tem[i][j-1])
                            large = max(tem[i-1][j],tem[i][j-1])
                            tem[i][j] = small
                            if small!=large:
                                for y in range(j):
                                    if tem[i][y]==large:
                                        tem[i][y]=small
                                if  ([small,large] in corner)==False:
                                    corner.append([small,large])
                                    right.append(small)
                                    wrong.append(large)

                        else: # isolated
                            label = label+1
                            tem[i][j] = label
                else: # first label
                    label = label+1
                    tem[i][j] = label
```

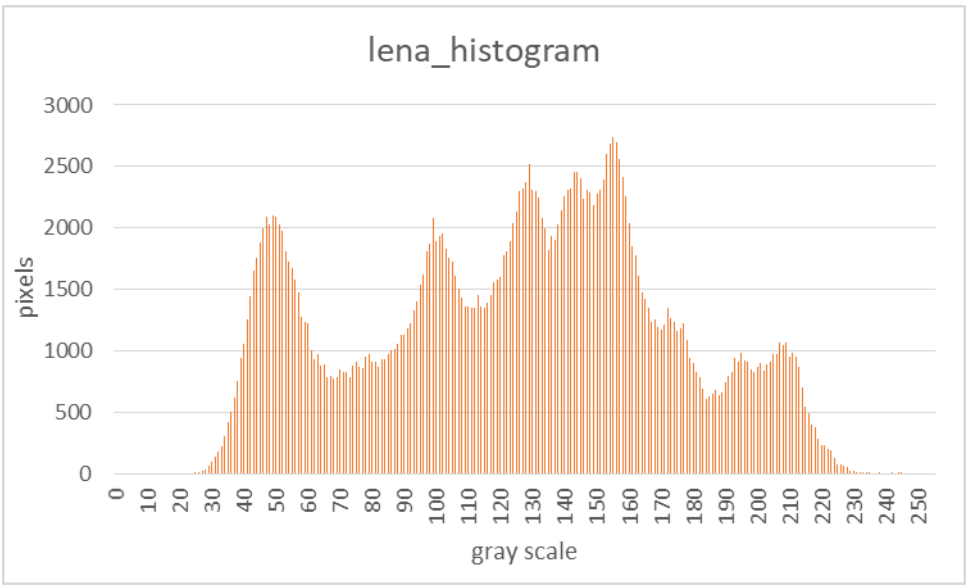
```
# substitute label in corner
for k in range(len(right)):
    for m in range(len(right)):
        if wrong[k]==right[m]:
            right[m]=right[k]

# substitute label in tem
for i in range(x.shape[0]-1,-1,-1):
    for j in range(x.shape[1]-1,-1,-1):
        if (tem[i][j]in wrong)==True:
            tem[i][j]=right[wrong.index(tem[i][j])]

# count pixels of each label
value = np.zeros([label+1])
for i in range(x.shape[0]):
    for j in range(x.shape[1]):
       value[int(tem[i][j])]=value[int(tem[i][j])]+1

# only save areas with more than 500 pixels
leftvalue = []
for k in range(len(value)):
    if value[k]>=500 and k!=0:
        leftvalue.append(k)

# relabel with consecutive integers
for i in range(x.shape[0]):
    for j in range(x.shape[1]):
        if (tem[i][j] in leftvalue)==True:
            tem[i][j] = leftvalue.index(tem[i][j])+1
        elif tem[i][j]!=0:
            tem[i][j] = 0

# centroid & boundary
for k in range(len(leftvalue)):
    xmax.append('n')
    xmin.append('n')
    ymax.append('n')
    ymin.append('n')
for k in range(len(leftvalue)):
    for i in range(x.shape[0]):
        for j in range(x.shape[1]):
            if tem[i][j]==k+1:
                xpos[k] = xpos[k]+j
                ypos[k] = ypos[k]+i
                xnum[k] = xnum[k]+1
                ynum[k] = ynum[k]+1
                if xmax[k]=='n':
                    xmax[k] = j
                elif j>xmax[k]:
                    xmax[k] = j
                if xmin[k]=='n':
                    xmin[k] = j
                elif j<xmin[k]:
                    xmin[k] = j
                if ymax[k]=='n':
                    ymax[k] = i
                elif i>ymax[k]:
                    ymax[k] = i
                if ymin[k]=='n':
                    ymin[k] = i
                elif i<ymin[k]:
                    ymin[k] = i

for k in range(len(leftvalue)):
    xcen[k] = xpos[k]/xnum[k]
    ycen[k] = ypos[k]/ynum[k]
```

完整內容請看 hw2.py

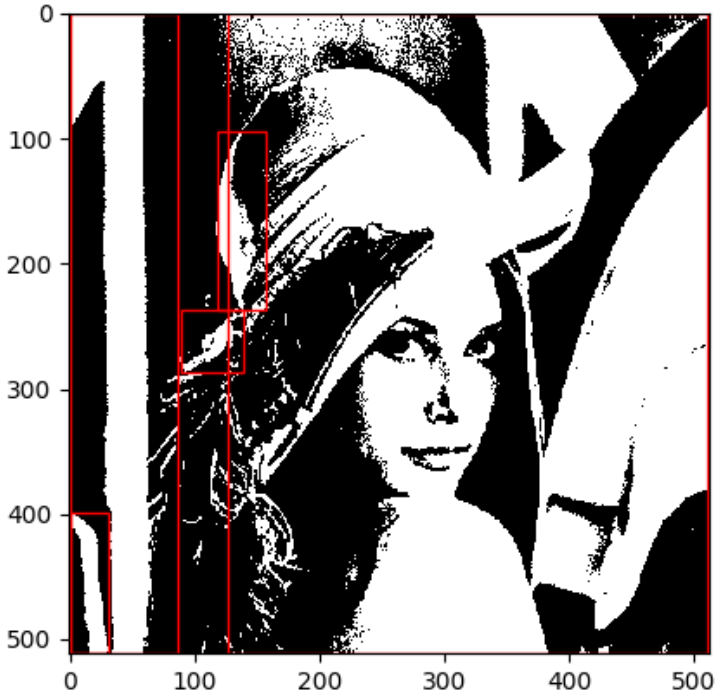
- **Resulting images**

1. a binary image (threshold at 128)

2. a histogram



3. connected components



label	(xmin,ymin)	(xmax,ymax)	centroid	pixels
1	(0,0)	(87,511)	(42,229)	18362
2	(127,0)	(511,511)	(344,245)	106045
3	(118,94)	(157,237)	(132,175)	2048
4	(89,237)	(139,287)	(117,264)	644
5	(0,399)	(31,511)	(18,458)	1490

cetroid=(xcen,ycen)：四捨五入至個位數