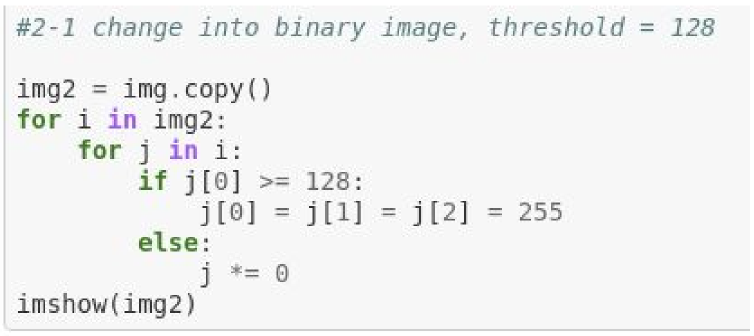
Computer Vision Homework2 report

R08922143 賴振東

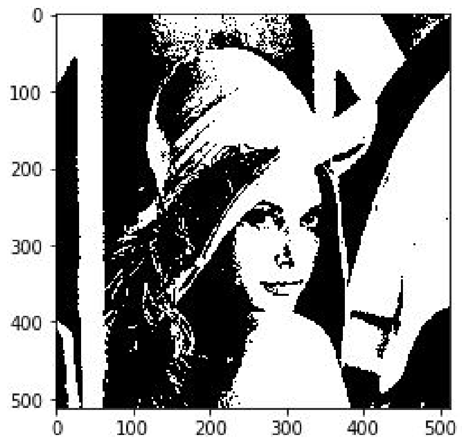
(a) a binary image (threshold at 128)

對圖片中每點的顏色作判斷 >=128 則設為 255，其餘則設為 0 。



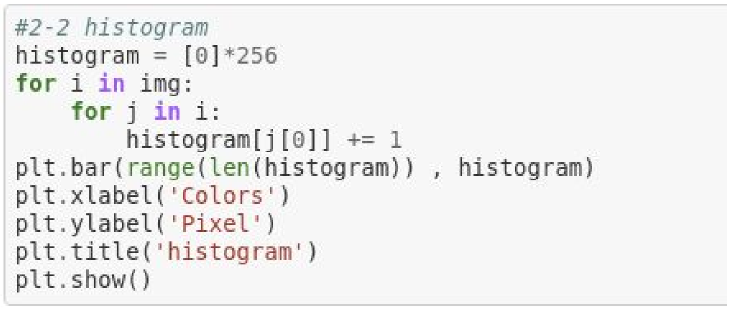
```
#2-1 change into binary image, threshold = 128

img2 = img.copy()
for i in img2:
    for j in i:
        if j[0] >= 128:
            j[0] = j[1] = j[2] = 255
        else:
            j *= 0
imshow(img2)
```

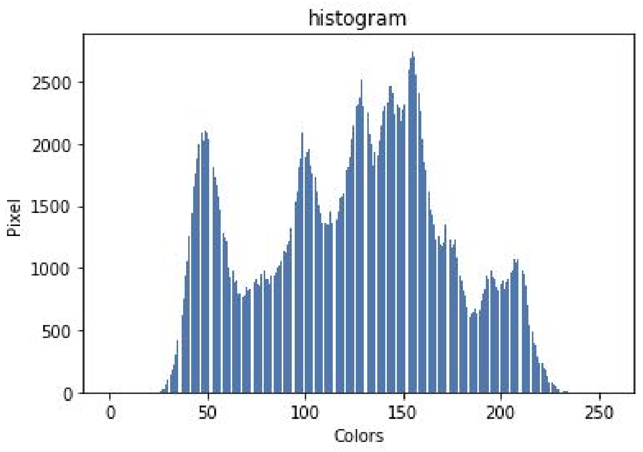


(b) a histogram

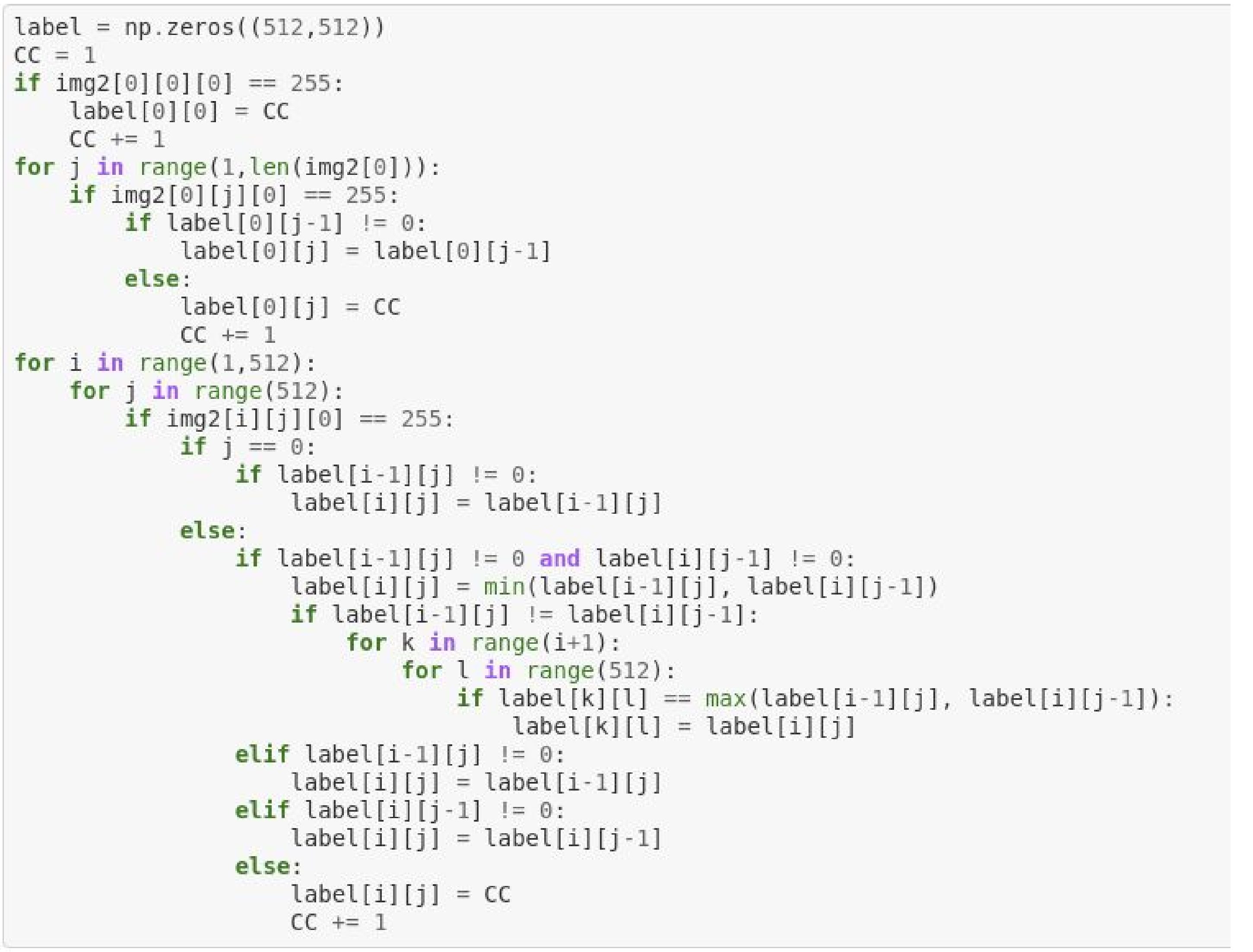
建立一個大小為256的Array紀錄各灰階值出現的次數，並用 pyplot.bar 繪製直方圖，縱軸為 pixel 數，橫軸為灰階值。



```
#2-2 histogram
histogram = [0]*256
for i in img:
    for j in i:
        histogram[j[0]] += 1
plt.bar(range(len(histogram)) , histogram)
plt.xlabel('Colors')
plt.ylabel('Pixel')
plt.title('histogram')
plt.show()
```



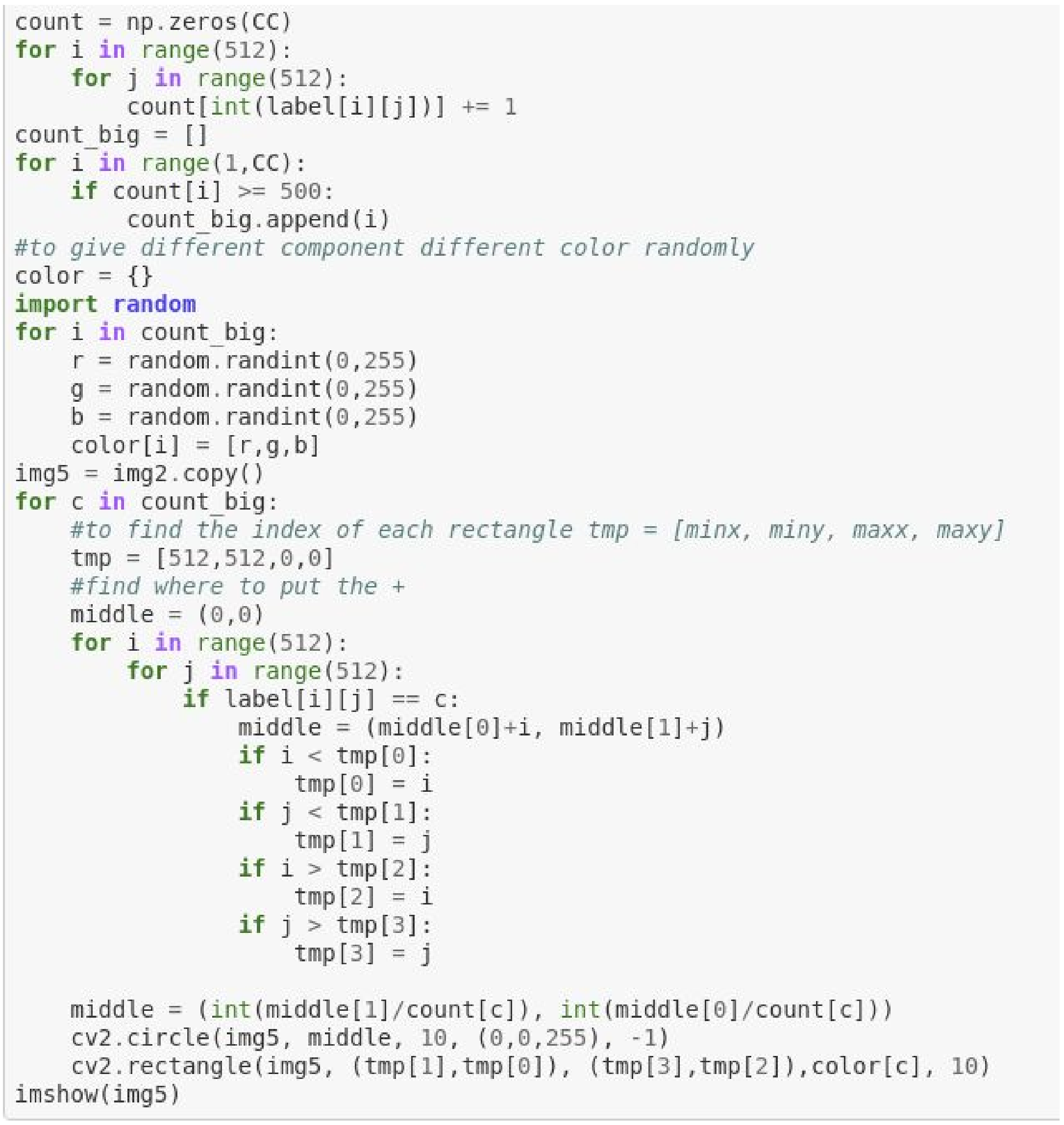
(c) connected components
(regions with + at centroid, bounding box)



```python
label = np.zeros((512,512))
CC = 1
if img2[0][0][0] == 255:
    label[0][0] = CC
    CC += 1
for j in range(1,len(img2[0])):
    if img2[0][j][0] == 255:
        if label[0][j-1] != 0:
            label[0][j] = label[0][j-1]
        else:
            label[0][j] = CC
            CC += 1
for i in range(1,512):
    for j in range(512):
        if img2[i][j][0] == 255:
            if j == 0:
                if label[i-1][j] != 0:
                    label[i][j] = label[i-1][j]
            else:
                if label[i-1][j] != 0 and label[i][j-1] != 0:
                    label[i][j] = min(label[i-1][j], label[i][j-1])
                    if label[i-1][j] != label[i][j-1]:
                        for k in range(i+1):
                            for l in range(512):
                                if label[k][l] == max(label[i-1][j], label[i][j-1]):
                                    label[k][l] = label[i][j]
                elif label[i-1][j] != 0:
                    label[i][j] = label[i-1][j]
                elif label[i][j-1] != 0:
                    label[i][j] = label[i][j-1]
                else:
                    label[i][j] = CC
                    CC += 1
```

這邊採用 4-Connected Component, 演算法與課堂上教的 Iterative algo 相似, 透過從左上方的 pixel(0,0) 一路掃到右下(511,511)結束, 當一 pixel 的灰階值為 255 時, 檢查其左邊與上面的臨點是否有大於0的 Label , 若只有一個有 Label 就將此點標示為該 Label , 若都沒有 Label 就為該點標上一個新的 Label, 當兩個鄰點的 Label 值不一樣時, 則選擇較小的作為此點 Label (Top Down)。但我在實作的時候每當兩相鄰點 Label 值不同時, 就執行將陣列大的 Label 值全部替換成小的 Label (Bottom Up), 而非待掃到最右下的點才執行此動作。
下圖的程式碼用來為各 Connected Component 畫出重心與 bounding box , 首先過濾出面積大於 500 的 Connected Component, 接著為這5個 CC

找出最左、最上、最下、最右的值，用來畫出 boundeing box。並將各相同 Label 的座標相加取平均，找出中心的位置以實心圓標示。



```python
count = np.zeros(CC)
for i in range(512):
    for j in range(512):
        count[int(label[i][j])] += 1
count_big = []
for i in range(1,CC):
    if count[i] >= 500:
        count_big.append(i)
#to give different component different color randomly
color = {}
import random
for i in count_big:
    r = random.randint(0,255)
    g = random.randint(0,255)
    b = random.randint(0,255)
    color[i] = [r,g,b]
img5 = img2.copy()
for c in count_big:
    #to find the index of each rectangle tmp = [minx, miny, maxx, maxy]
    tmp = [512,512,0,0]
    #find where to put the +
    middle = (0,0)
    for i in range(512):
        for j in range(512):
            if label[i][j] == c:
                middle = (middle[0]+i, middle[1]+j)
                if i < tmp[0]:
                    tmp[0] = i
                if j < tmp[1]:
                    tmp[1] = j
                if i > tmp[2]:
                    tmp[2] = i
                if j > tmp[3]:
                    tmp[3] = j

    middle = (int(middle[1]/count[c]), int(middle[0]/count[c]))
    cv2.circle(img5, middle, 10, (0,0,255), -1)
    cv2.rectangle(img5, (tmp[1],tmp[0]), (tmp[3],tmp[2]),color[c], 10)
imshow(img5)
```

Ps: 碼都放在https://github.com/e94046165/2019NTUcourse/tree/master/CV/HW2