Computer Vision HW2

r12922054 資工所 邱信瑋

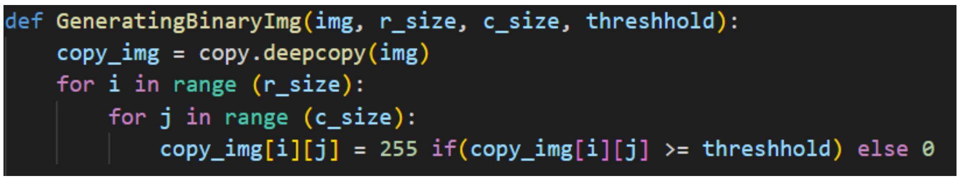
Description

Part 1.

1. A binary image (threshold at 128)

How to implement :

遍歷整個圖片，將大於等於128的值設為255，小於128設為0



```
def GeneratingBinaryImg(img, r_size, c_size, threshhold):
    copy_img = copy.deepcopy(img)
    for i in range (r_size):
        for j in range (c_size):
            copy_img[i][j] = 255 if(copy_img[i][j] >= threshhold) else 0
```

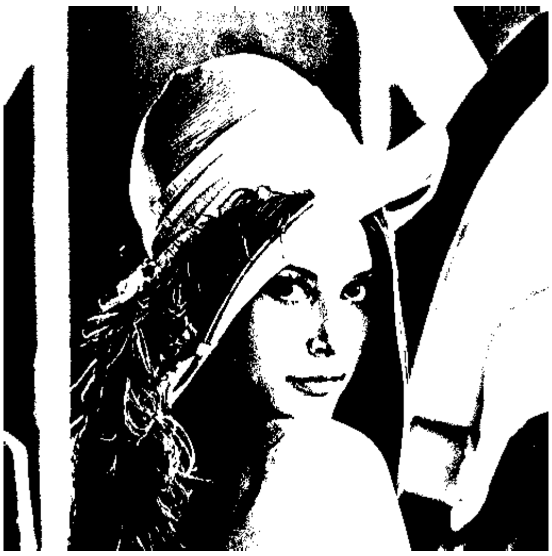
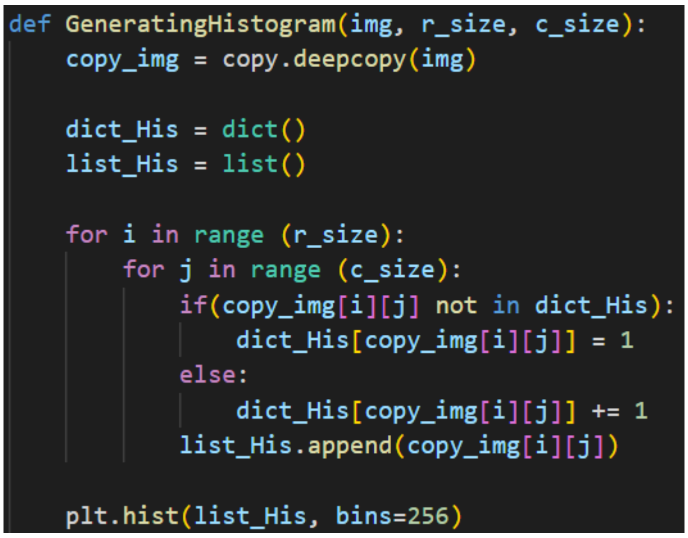


Figure 1: upside_down image.

2. A histogram

How to implement :

透過dictionay，將每個intensity有幾個pixel記錄下來，不過我這裡透過這個dictionary作圖好像只能做出折線圖，因此，後來透過list紀錄，並透matplotlib.pyplot中hist()function完成



```python
def GeneratingHistogram(img, r_size, c_size):
    copy_img = copy.deepcopy(img)

    dict_His = dict()
    list_His = list()

    for i in range (r_size):
        for j in range (c_size):
            if(copy_img[i][j] not in dict_His):
                dict_His[copy_img[i][j]] = 1
            else:
                dict_His[copy_img[i][j]] += 1
            list_His.append(copy_img[i][j])

    plt.hist(list_His, bins=256)
```

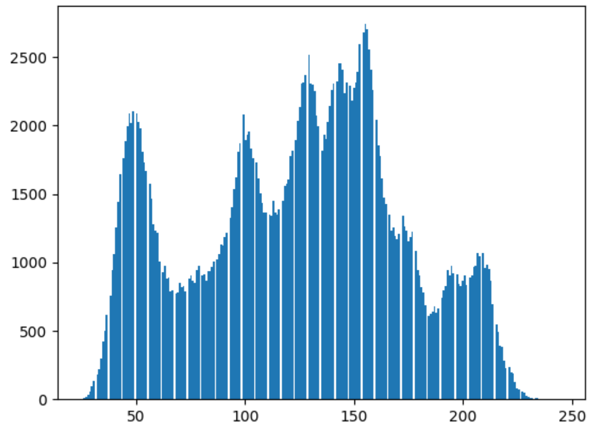


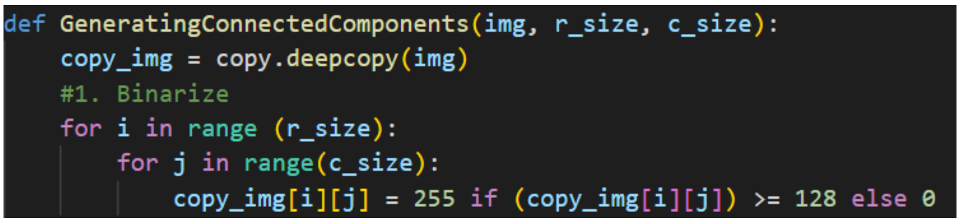
Figure 2: right_side_left image.

3. Connected components

How to implement：

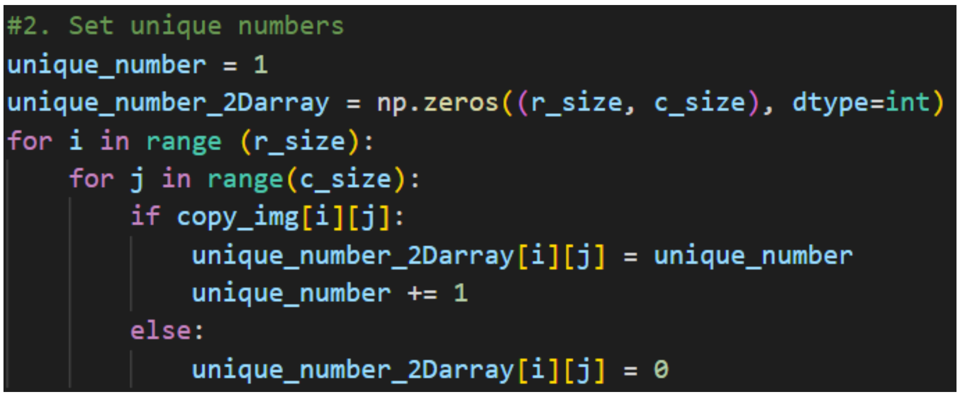
本題我使用Iterative Algorithm，並參考PPT上面的Pseudo code實作

(a) 第一步:二質化圖片



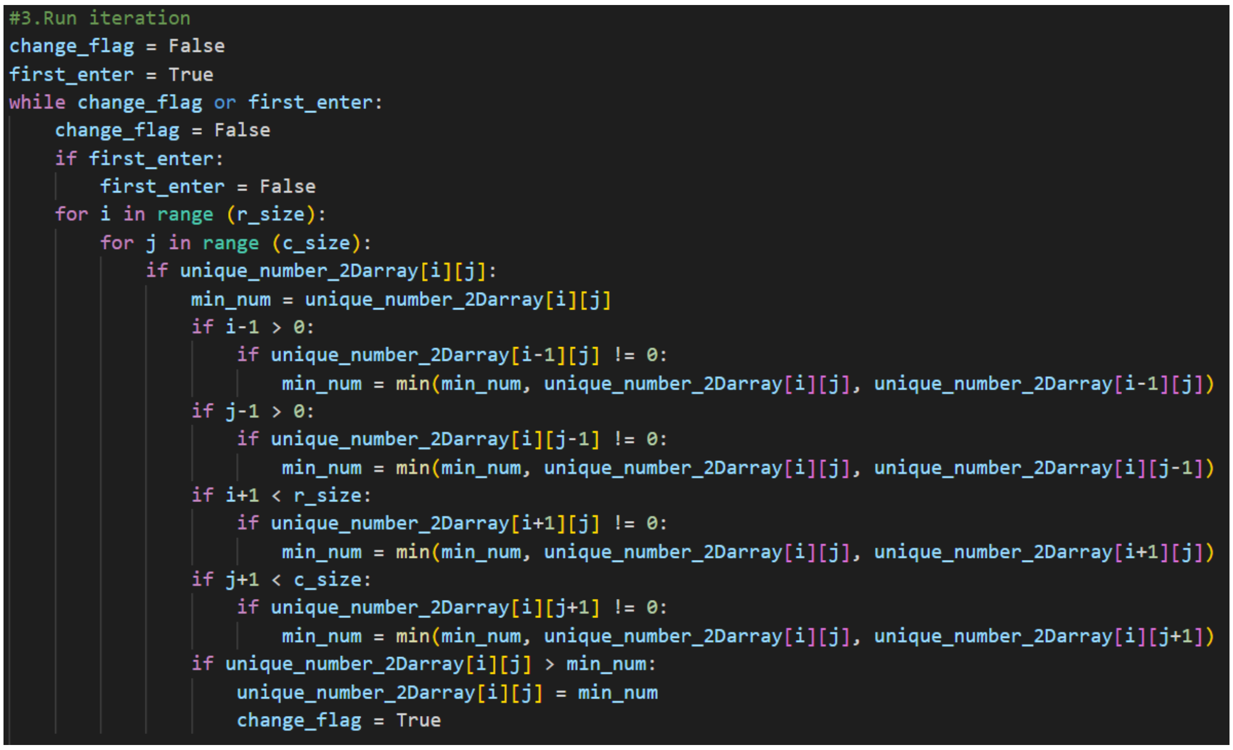
```python
def GeneratingConnectedComponents(img, r_size, c_size):
    copy_img = copy.deepcopy(img)
    #1. Binarize
    for i in range (r_size):
        for j in range(c_size):
            copy_img[i][j] = 255 if (copy_img[i][j]) >= 128 else 0
```

(b) 第二步:設定Unique Numbers

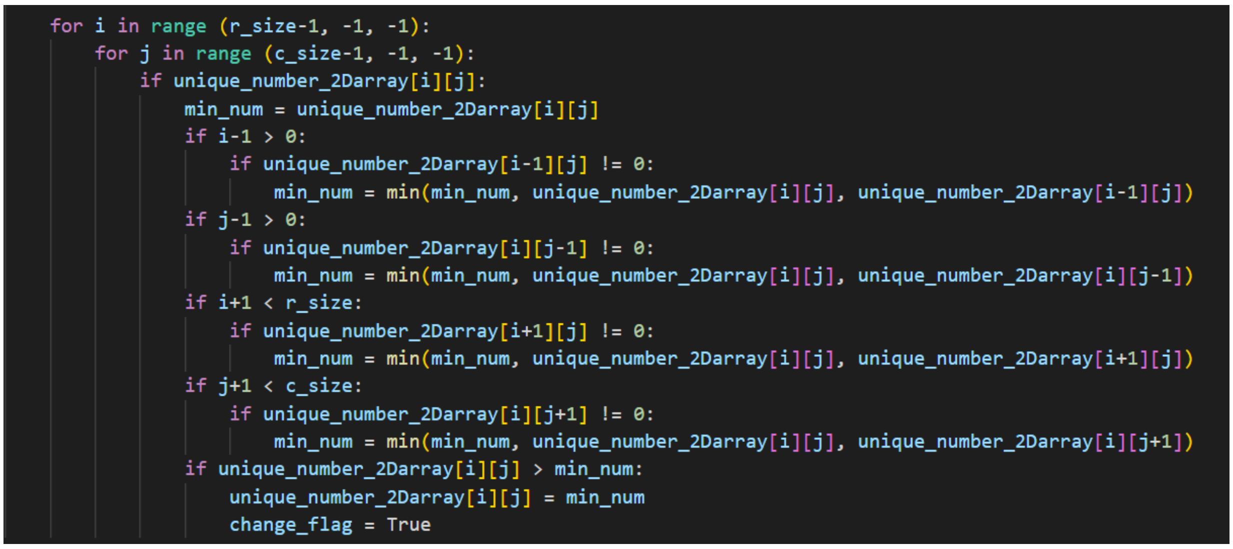


```python
#2. Set unique numbers
unique_number = 1
unique_number_2Darray = np.zeros((r_size, c_size), dtype=int)
for i in range (r_size):
    for j in range(c_size):
        if copy_img[i][j]:
            unique_number_2Darray[i][j] = unique_number
            unique_number += 1
        else:
            unique_number_2Darray[i][j] = 0
```

(c) 第三步:開始Top-Down followed by bottom-up passes，直到unique number不再改變



```python
#3.Run iteration
change_flag = False
first_enter = True
while change_flag or first_enter:
    change_flag = False
    if first_enter:
        first_enter = False
    for i in range (r_size):
        for j in range (c_size):
            if unique_number_2Darray[i][j]:
                min_num = unique_number_2Darray[i][j]
                if i-1 > 0:
                    if unique_number_2Darray[i-1][j] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i-1][j])
                if j-1 > 0:
                    if unique_number_2Darray[i][j-1] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i][j-1])
                if i+1 < r_size:
                    if unique_number_2Darray[i+1][j] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i+1][j])
                if j+1 < c_size:
                    if unique_number_2Darray[i][j+1] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i][j+1])
                if unique_number_2Darray[i][j] > min_num:
                    unique_number_2Darray[i][j] = min_num
                    change_flag = True
```

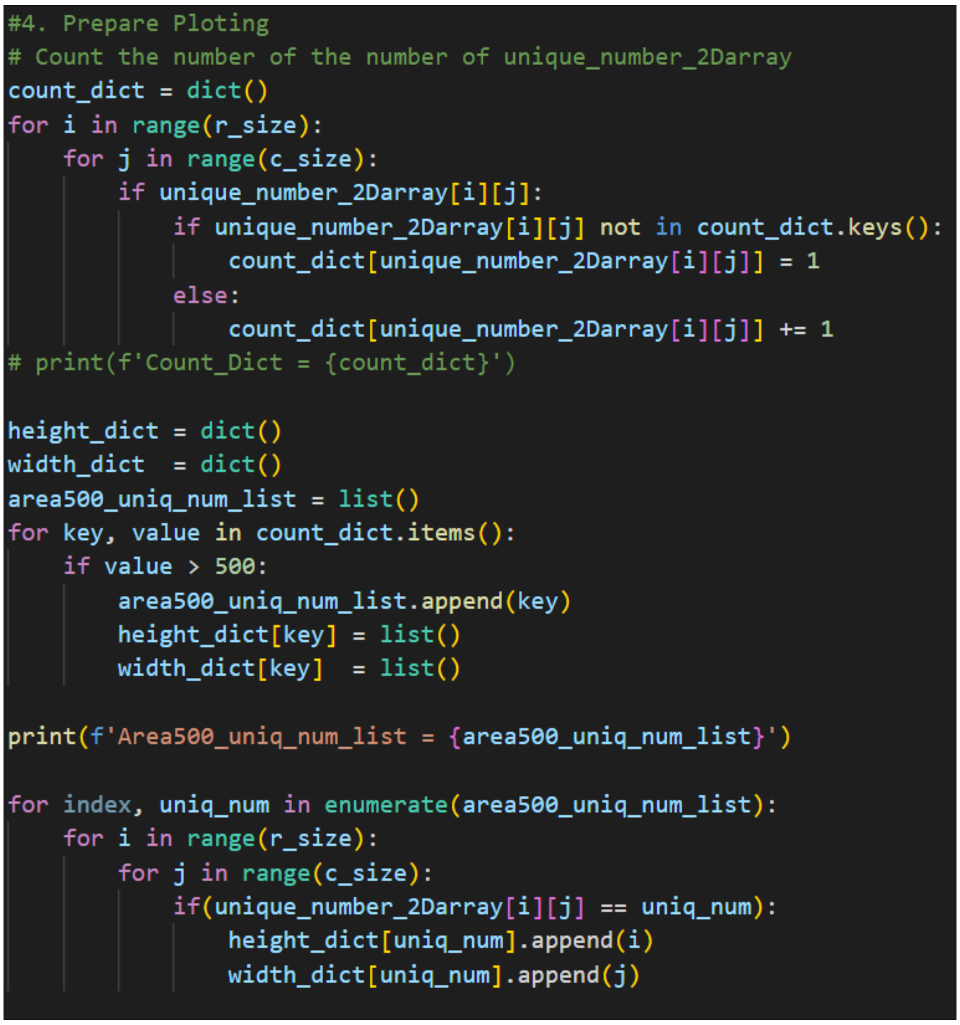


```python
    for i in range (r_size-1, -1, -1):
        for j in range (c_size-1, -1, -1):
            if unique_number_2Darray[i][j]:
                min_num = unique_number_2Darray[i][j]
                if i-1 > 0:
                    if unique_number_2Darray[i-1][j] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i-1][j])
                if j-1 > 0:
                    if unique_number_2Darray[i][j-1] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i][j-1])
                if i+1 < r_size:
                    if unique_number_2Darray[i+1][j] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i+1][j])
                if j+1 < c_size:
                    if unique_number_2Darray[i][j+1] != 0:
                        min_num = min(min_num, unique_number_2Darray[i][j], unique_number_2Darray[i][j+1])
                if unique_number_2Darray[i][j] > min_num:
                    unique_number_2Darray[i][j] = min_num
                    change_flag = True
```

(d) 第四步:準備畫圖前的動作，分別做:

1.用dictionary紀錄每個unique number有幾個(因為本題有要求須要將Area超過500的區塊bounding起來，所以才需要紀錄)

2.遍歷所有點，如果他的unique number是屬於面積超過500的unique number，則記錄其位置(之後做排序就可以知道bounding區域的左上角以及右下角在哪裡了)



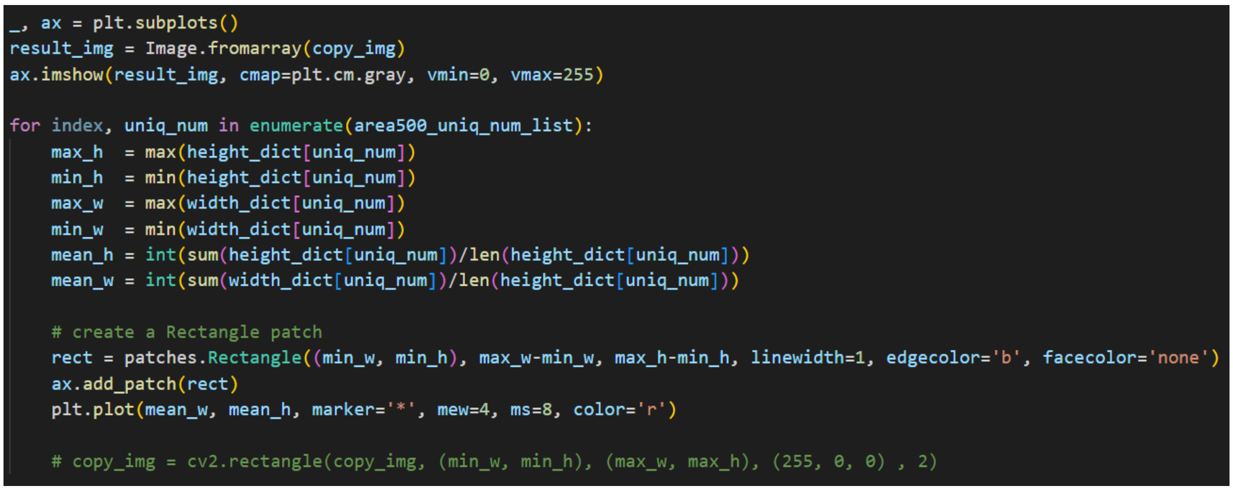
```python
#4. Prepare Ploting
# Count the number of the number of unique_number_2Darray
count_dict = dict()
for i in range(r_size):
    for j in range(c_size):
        if unique_number_2Darray[i][j]:
            if unique_number_2Darray[i][j] not in count_dict.keys():
                count_dict[unique_number_2Darray[i][j]] = 1
            else:
                count_dict[unique_number_2Darray[i][j]] += 1
# print(f'Count_Dict = {count_dict}')

height_dict = dict()
width_dict  = dict()
area500_uniq_num_list = list()
for key, value in count_dict.items():
    if value > 500:
        area500_uniq_num_list.append(key)
        height_dict[key] = list()
        width_dict[key]  = list()

print(f'Area500_uniq_num_list = {area500_uniq_num_list}')

for index, uniq_num in enumerate(area500_uniq_num_list):
    for i in range(r_size):
        for j in range(c_size):
            if(unique_number_2Darray[i][j] == uniq_num):
                height_dict[uniq_num].append(i)
                width_dict[uniq_num].append(j)
```

(e) 第五步:畫圖，透過matplotlib.patches中的patches function來完成矩形的繪製，最後透過matplotlib.pyplot中的plot()function 繪製圖片並將質心位置以* 表示



```python
_, ax = plt.subplots()
result_img = Image.fromarray(copy_img)
ax.imshow(result_img, cmap=plt.cm.gray, vmin=0, vmax=255)

for index, uniq_num in enumerate(area500_uniq_num_list):
    max_h  = max(height_dict[uniq_num])
    min_h  = min(height_dict[uniq_num])
    max_w  = max(width_dict[uniq_num])
    min_w  = min(width_dict[uniq_num])
    mean_h = int(sum(height_dict[uniq_num])/len(height_dict[uniq_num]))
    mean_w = int(sum(width_dict[uniq_num])/len(height_dict[uniq_num]))

    # create a Rectangle patch
    rect = patches.Rectangle((min_w, min_h), max_w-min_w, max_h-min_h, linewidth=1, edgecolor='b', facecolor='none')
    ax.add_patch(rect)
    plt.plot(mean_w, mean_h, marker='*', mew=4, ms=8, color='r')

    # copy_img = cv2.rectangle(copy_img, (min_w, min_h), (max_w, max_h), (255, 0, 0) , 2)
```

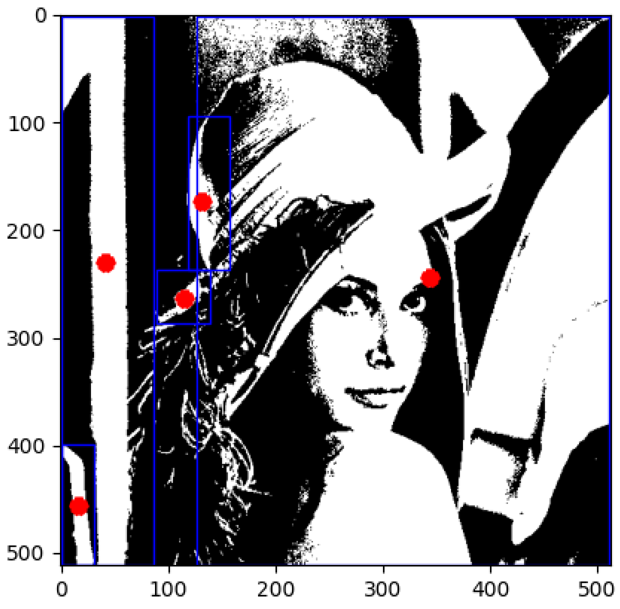


Figure 3: diagonally_flip image.