R02944010 李柏緯

# Computer Vision Homework 2

## How to run it

```
python image.py lena.bmp
```

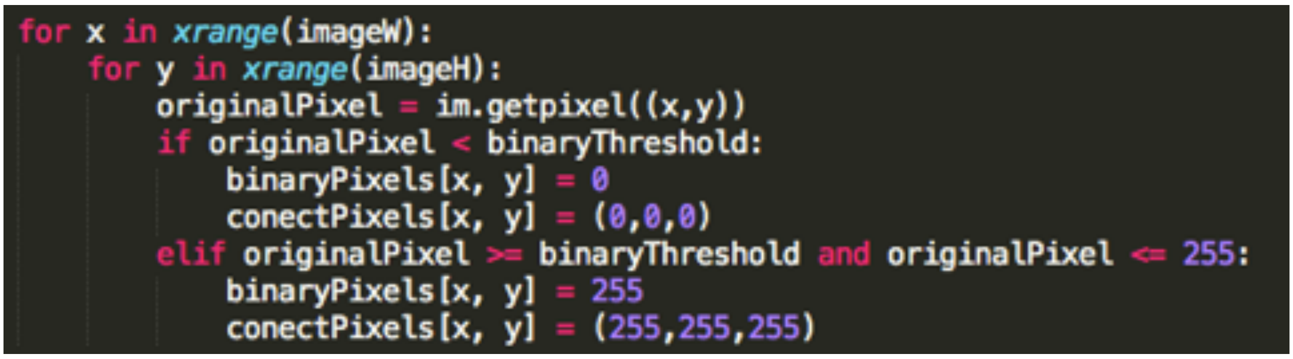
## Binarizing

### Description

I use Python Image Library to do image I/O. When I get image raw pixel data I do these computing.

- Set pixel as white when original pixel small than threshold.
- Set pixel as black when original pixel larger than threshold

### Principal code fragment



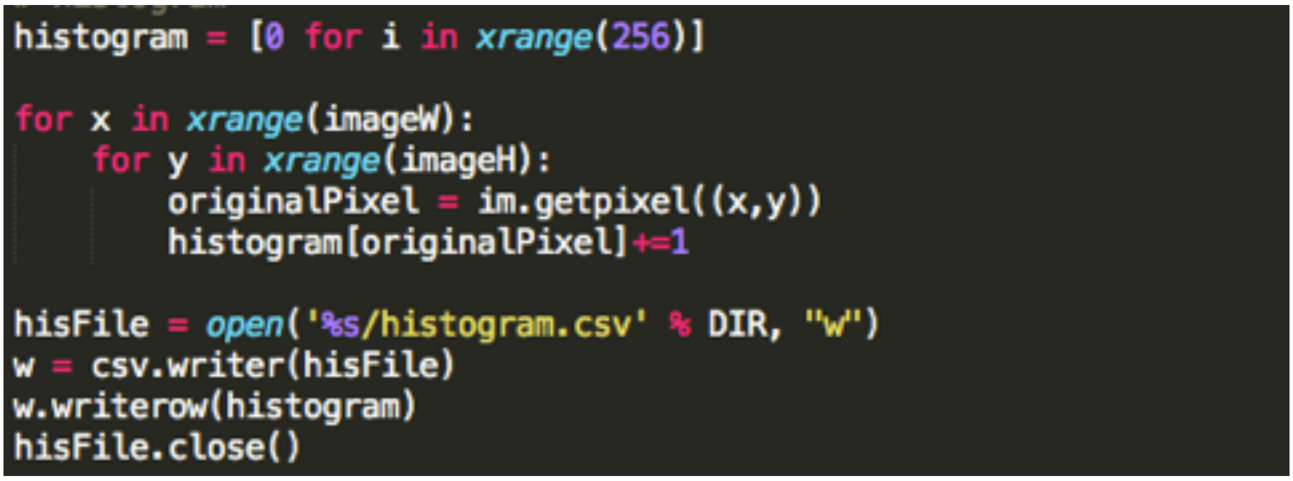
```
for x in xrange(imageW):
    for y in xrange(imageH):
        originalPixel = im.getpixel((x,y))
        if originalPixel < binaryThreshold:
            binaryPixels[x, y] = 0
            conectPixels[x, y] = (0,0,0)
        elif originalPixel >= binaryThreshold and originalPixel <= 255:
            binaryPixels[x, y] = 255
            conectPixels[x, y] = (255,255,255)
```

## Histogram

### Description

Count every pixel's value and generate csv file which store the times of pixel value from 0 ~ 255

### Principal code fragment

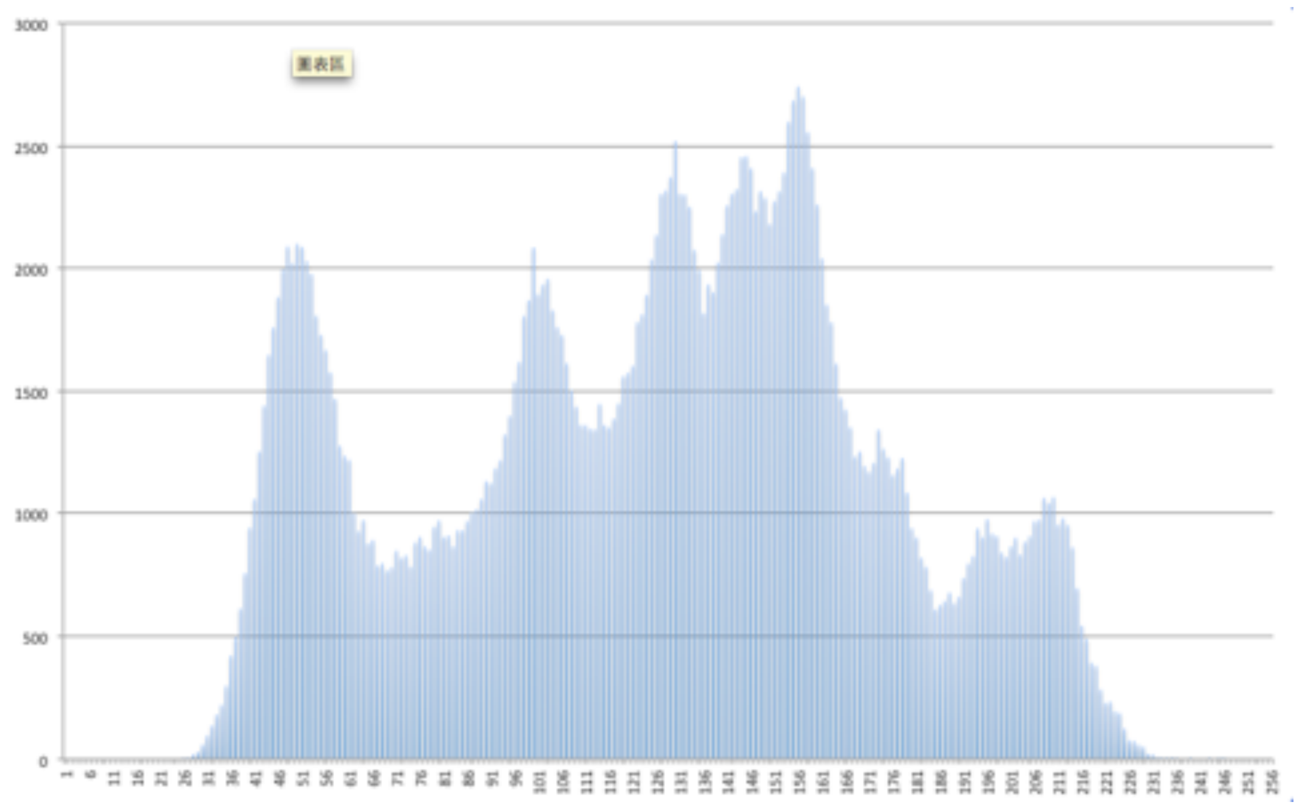


```
histogram = [0 for i in xrange(256)]

for x in xrange(imageW):
    for y in xrange(imageH):
        originalPixel = im.getpixel((x,y))
        histogram[originalPixel]+=1

hisFile = open('%s/histogram.csv' % DIR, "w")
w = csv.writer(hisFile)
w.writerow(histogram)
hisFile.close()
```

## Result

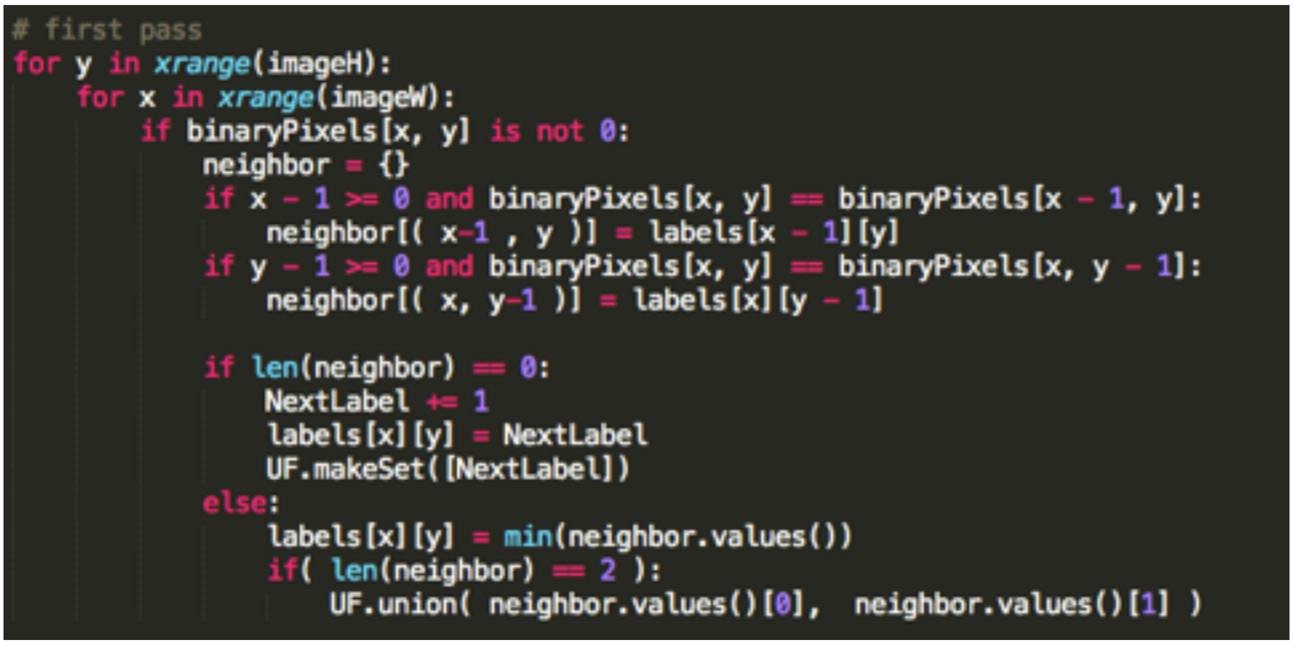


## Connected components

### Description

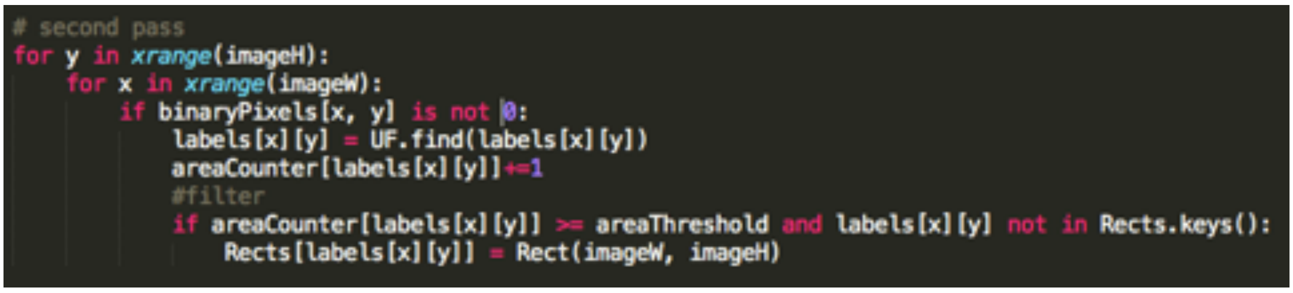
I Use Union Find Data Structure to record label equivalent and two-pass ( 4-direction scanning )to label connected components.
First, scan all pixels to connect left and top neighbor of every pixel. If both top and left are neighbors, set left and top neighbors' label equal.
Second, Scan all pixel and change label as the smallest one in label equivalent.

### Principal code fragment



```
# first pass
for y in xrange(imageH):
    for x in xrange(imageW):
        if binaryPixels[x, y] is not 0:
            neighbor = {}
            if x - 1 >= 0 and binaryPixels[x, y] == binaryPixels[x - 1, y]:
                neighbor[( x-1 , y )] = labels[x - 1][y]
            if y - 1 >= 0 and binaryPixels[x, y] == binaryPixels[x, y - 1]:
                neighbor[( x, y-1 )] = labels[x][y - 1]

            if len(neighbor) == 0:
                NextLabel += 1
                labels[x][y] = NextLabel
                UF.makeSet([NextLabel])
            else:
                labels[x][y] = min(neighbor.values())
                if( len(neighbor) == 2 ):
                    UF.union( neighbor.values()[0],  neighbor.values()[1] )
```



```
# second pass
for y in xrange(imageH):
    for x in xrange(imageW):
        if binaryPixels[x, y] is not 0:
            labels[x][y] = UF.find(labels[x][y])
            areaCounter[labels[x][y]]+=1
            #filter
            if areaCounter[labels[x][y]] >= areaThreshold and labels[x][y] not in Rects.keys():
                Rects[labels[x][y]] = Rect(imageW, imageH)
```

## Result