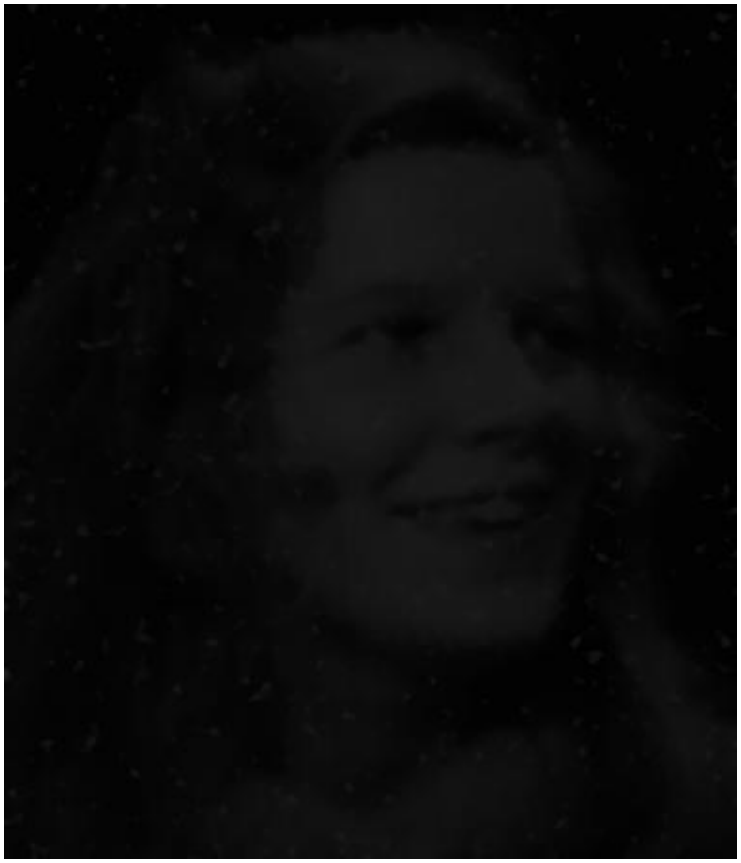
cv2016.mp1.104356034 - Histogram Equalization

104356034　資管碩一 傅品甄

(1) Apply histogram equalization to mp1.jpg.

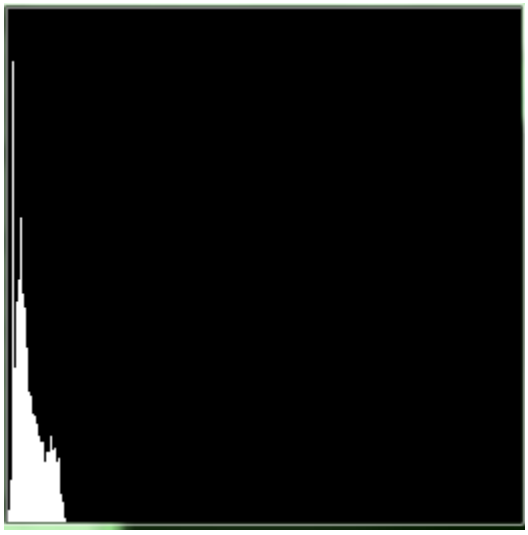
把 mp1 的灰階圖片透過 equalization 讓圖片變得清楚



我先利用第一個程式 gray 把此灰階圖片以直方圖顯示出來

```python
import cv2;
import numpy as np

  #顯示出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;

  #讀取圖片  第二個參數若為0  代表為灰階
img = cv2.imread('D:\\Freddy\\vision\\mp1.jpg',0)
  #利用上面的函式去畫出直方圖
histgray = calcAndDrawHist(img,[255,255,255])
  #顯示出直方圖與原始圖
cv2.imshow("histgray", histgray)
cv2.imshow("Image", img);
cv2.waitKey(0);
```

mp1 原圖跑出來的直方圖為下



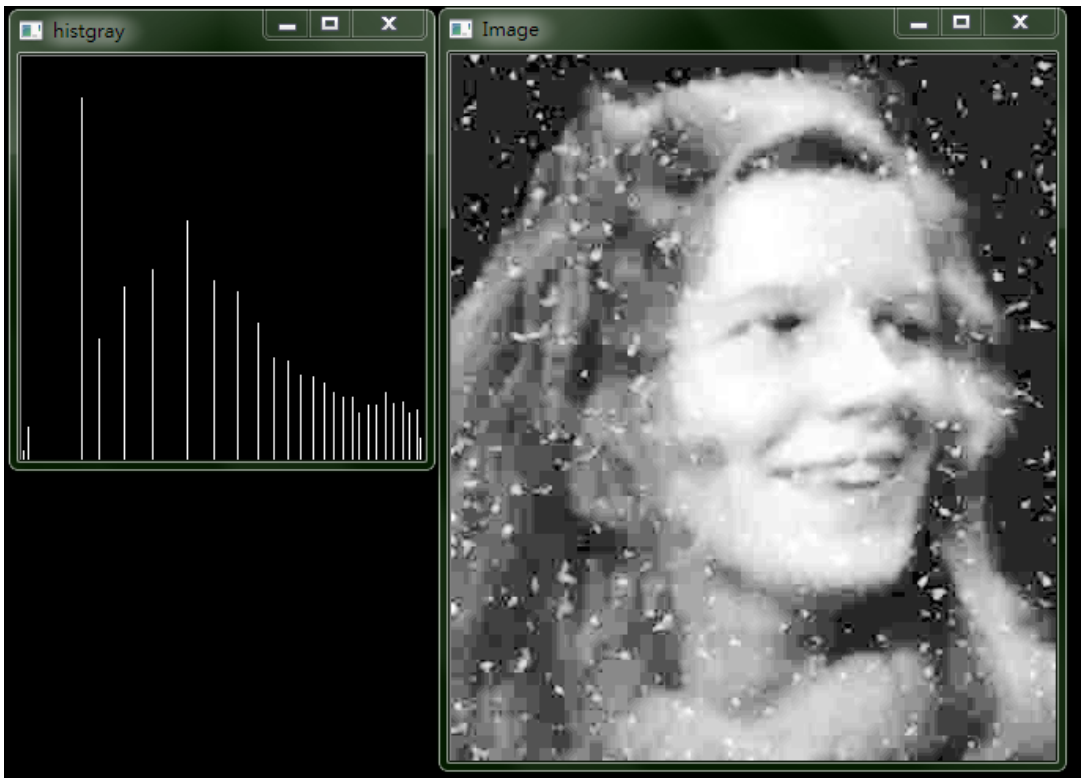
可看出分布的很不平均

故我們利用老師所教的 equalization 去等化這個圖的顏色分布

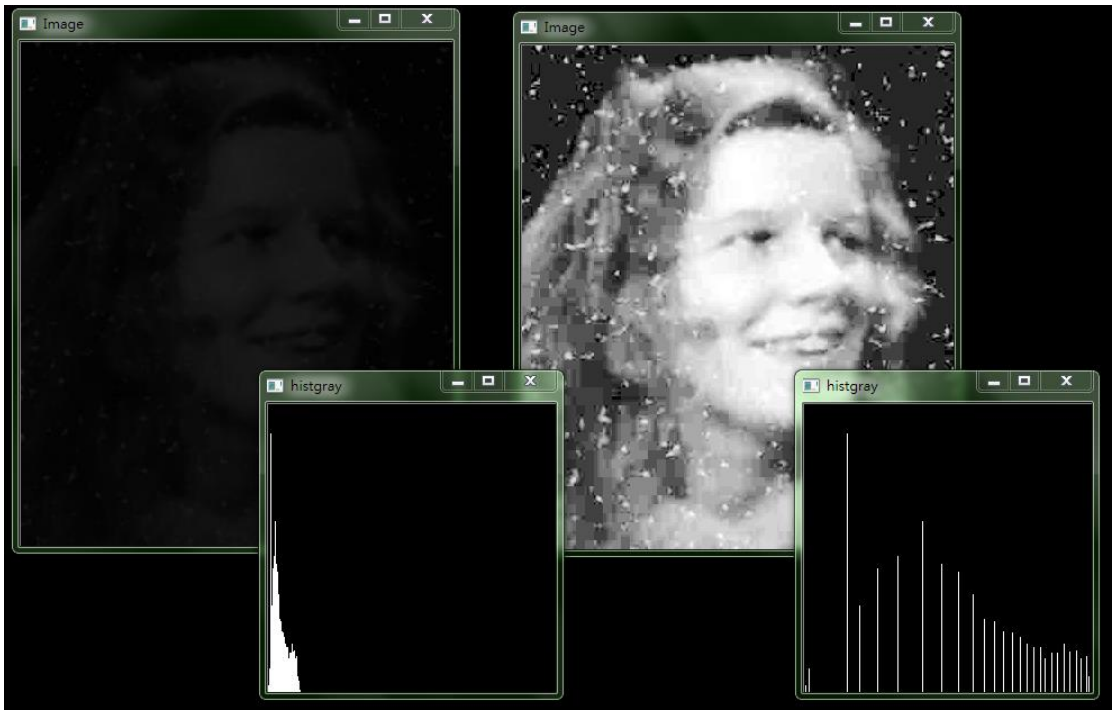
```python
import cv2;
import numpy as np
#畫出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;
#讀取圖片  第二個參數若為0  代表為灰階
imggray = cv2.imread('D:\\Freddy\\vision\\mp1.jpg',0)
#初始各項變數
bit = 256
high = len(imggray)
width = len(imggray[0])
sum = high*width
oldimggraydis = [0] * bit
oldimggrayrat = [0] * bit
newimggraydis = [0] * bit
```

```
26  #先算出每個色階的數量
27  for i in range(0,high,1):
28      for j in range(0,width,1):
29          oldimggraydis[imggray[i][j]]+=1
30  #計算每個色階所佔的比例  看出分布
31  for i in range(0,bit,1):
32      oldimggrayrat[i] = float(oldimggraydis[i])/sum
33  #算出每個色階在經過equalization後所轉換成的色階
34  for i in range(0,bit,1):
35      tempgray = 0
36
37      for j in range(0,i+1,1):
38          tempgray = tempgray + oldimggrayrat[j]
39
40      newimggraydis[i] = round((bit-1) * tempgray)
41
42  #把轉換好的色階覆蓋回去原圖的色階
43  for i in range(0,high,1):
44      for j in range(0,width,1):
45          imggray[i][j] = newimggraydis[imggray[i][j]]
46
47  #顯示出經過equalization的圖片和直方圖
48  histgray = calcAndDrawHist(imggray,[255,255,255])
49  cv2.imshow("histgray", histgray)
50  cv2.imshow("Image", imggray);
51  cv2.waitKey(0);
52  cv2.destroyAllWindows();
```

經過 equalization 之後的結果如下



讓我們來看一下原圖和等化後的圖片，發現從原本幾乎看不清楚輪廓的灰階圖變得很清晰可見，雖然有一些小雜訊，但卻能清楚看出是什麼圖案



(2) Process mp1a.jpg by applying histogram equalization to
這題要把 mp1a 分別以以下三種方法去 equalization

(a) R,G,B channels separately

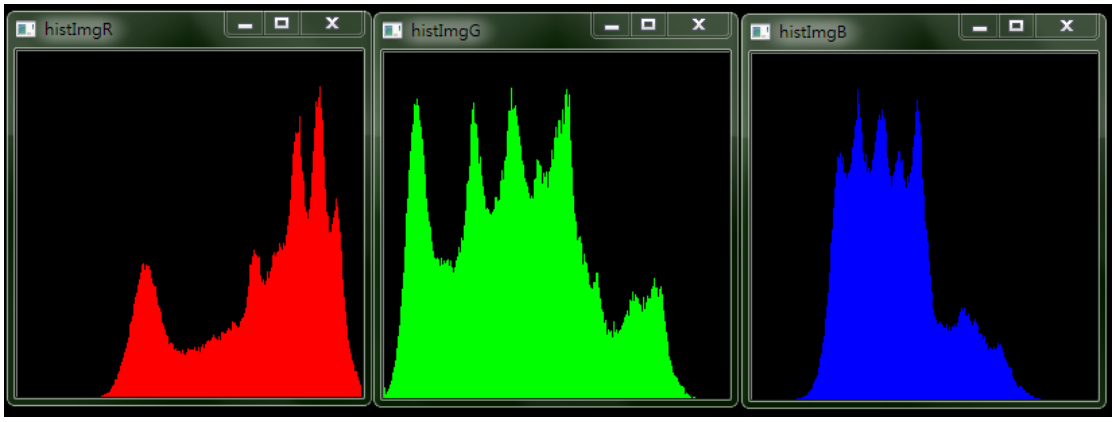
在這邊我們要將彩色圖片先截取出三種不同的顏色元素 RGB，再將各色階分別去做 equalization 之後，再把三個元素的圖片合在一起

首先，我想先看看原圖的 RGB 的直方圖是如何分布的

```python
import cv2;
import numpy as np
#畫出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;


#讀取圖片進來
img = cv2.imread('D:\\Freddy\\vision\\mp1a.jpg')
#把RGB都分開出來
b, g, r= cv2.split(img)
histImgB = calcAndDrawHist(b, [255, 0, 0])
histImgG = calcAndDrawHist(g, [0, 255, 0])
histImgR = calcAndDrawHist(r, [0, 0, 255])

#分別顯示RGB的直方圖和原圖
cv2.imshow("histImgB", histImgB)
cv2.imshow("histImgG", histImgG)
cv2.imshow("histImgR", histImgR)
cv2.imshow("Image", img);
cv2.waitKey(0);
```

執行程式後的結果(我只截取直方圖的部分)

左到右分別是 R G B 的直方圖



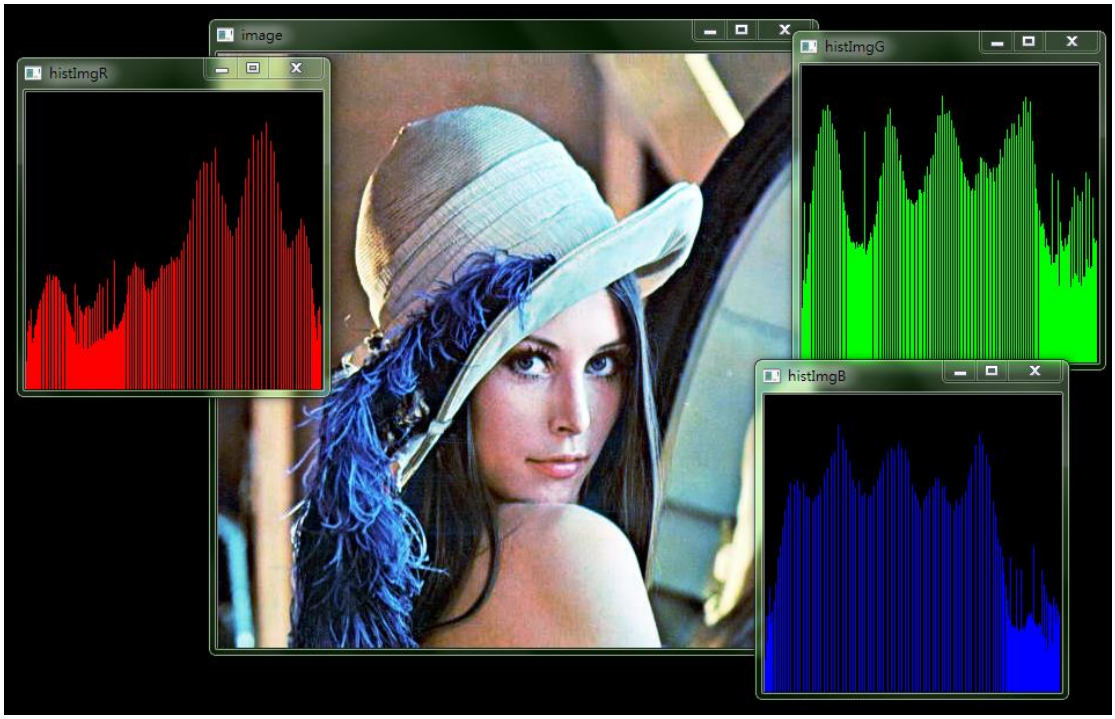
接著我們分別把 RGB 分開去做 equalization

```python
import cv2;
import numpy as np
#顯示出直方圖的函式
def calcAndDrawHist(img, color):
    hist= cv2.calcHist([img], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)
    return histImg;


#讀取圖片進來
img = cv2.imread('D:\\Freddy\\vision\\mp1a.jpg')
#把RGB個別分開
b,g,r= cv2.split(img)
#初始各項變數
bit = 256
high = len(b)
width = len(b[0])
sum = high*width
oldBdis = [0] * bit
oldGdis = [0] * bit
oldRdis = [0] * bit
oldBrat = [0] * bit
oldGrat = [0] * bit
oldRrat = [0] * bit
newBdis = [0] * bit
newGdis = [0] * bit
newRdis = [0] * bit
```

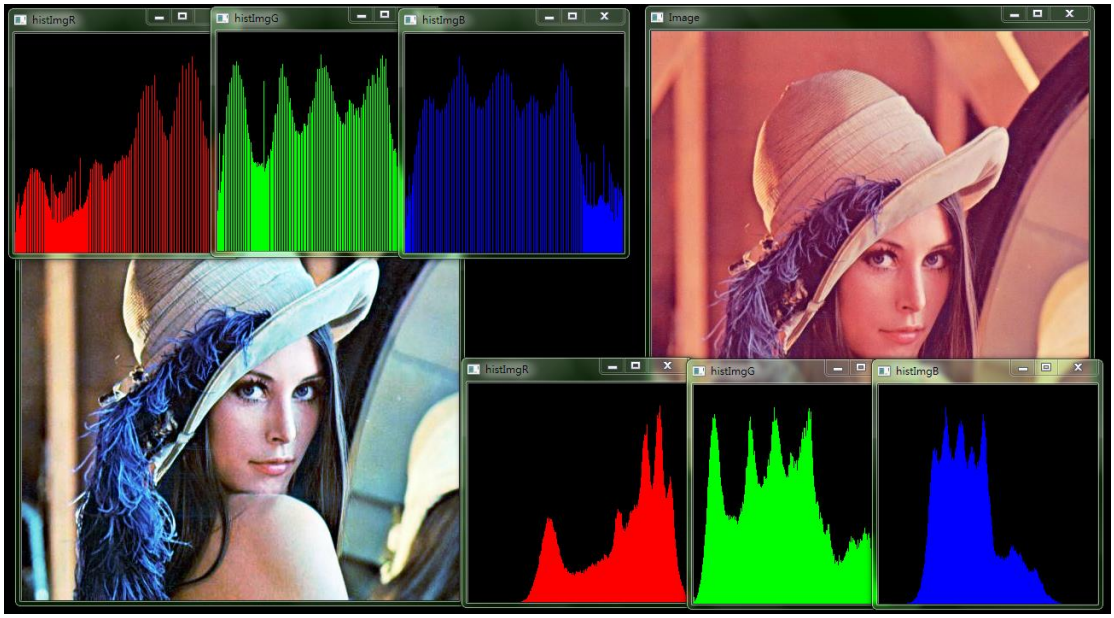
```python
#個別算出RGB各色階的數量
for i in range(0,high,1):
    for j in range(0,width,1):
        oldBdis[b[i][j]]+=1
        oldGdis[g[i][j]]+=1
        oldRdis[r[i][j]]+=1

#算出RGB的各色階比例  分布
for i in range(0,bit,1):
    oldBrat[i] = float(oldBdis[i])/sum
    oldGrat[i] = float(oldGdis[i])/sum
    oldRrat[i] = float(oldRdis[i])/sum


#利用老師教的euqalization公式去算初等化後的色階
for i in range(0,bit,1):
    tempB = 0
    tempG = 0
    tempR = 0
    for j in range(0,i+1,1):
        tempB = tempB + oldBrat[j]
        tempG = tempG + oldGrat[j]
        tempR = tempR + oldRrat[j]

    newBdis[i] = round((bit-1) * tempB)
    newGdis[i] = round((bit-1) * tempG)
    newRdis[i] = round((bit-1) * tempR)

```

```
63    #把算好的色階覆蓋回去原本的圖片色階
64    for i in range(0,high,1):
65        for j in range(0,width,1):
66            b[i][j] = newBdis[b[i][j]]
67            g[i][j] = newGdis[g[i][j]]
68            r[i][j] = newRdis[r[i][j]]
69
70
71    histImgB = calcAndDrawHist(b, [255, 0, 0])
72    histImgG = calcAndDrawHist(g, [0, 255, 0])
73    histImgR = calcAndDrawHist(r, [0, 0, 255])
74    cv2.imshow("histImgB", histImgB)
75    cv2.imshow("histImgG", histImgG)
76    cv2.imshow("histImgR", histImgR)
77    #再把RGB合併在一起
78    img = cv2.merge((b,g,r))
79    cv2.imshow('image',img);
80    #另存一個名為new的圖片
81    cv2.imwrite('new.jpg',img);
82    cv2.waitKey(0);
83    cv2.destroyAllWindows();
```

經過了 RGB 的等化之後，結果如下



再將原圖和等化後的圖作比較，可以發現顏色鮮明了不少，不會特別偏向某一種色系



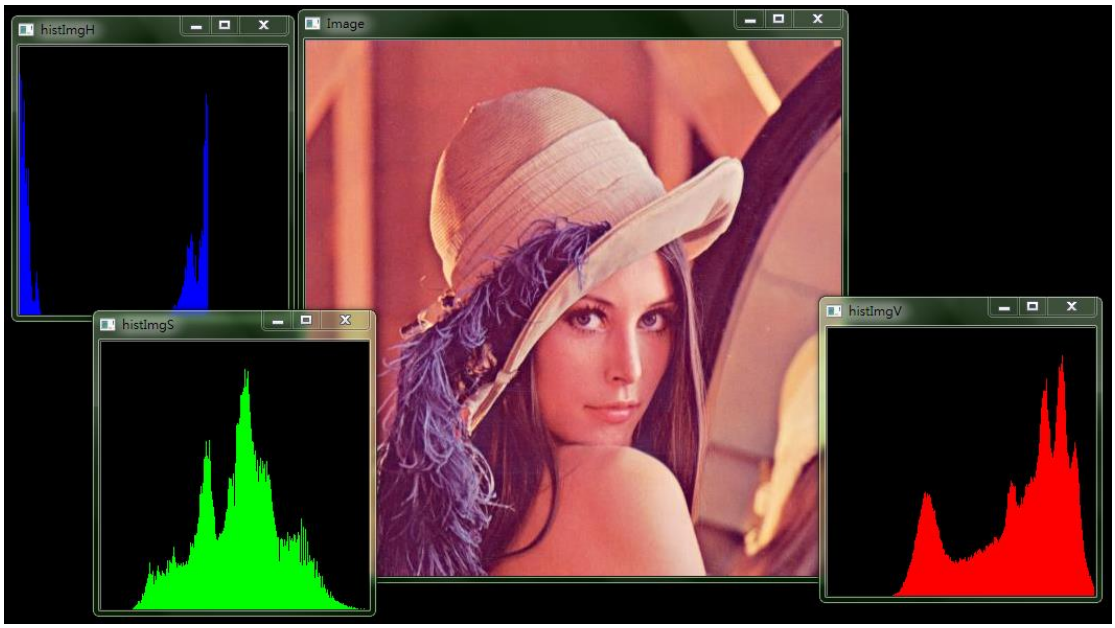
(b) V channel of HSV representation

第 b 小題和第 a 小題很類似，只是現在是利用 HSV 的 V 去作等化，我一樣先去看看原本的 HSV 個別是怎麼分布的

```python
import cv2;
import numpy as np
#顯示出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;


#把圖片讀取進來
img = cv2.imread('D:\\Freddy\\vision\\mp1a.jpg')
#先將圖片轉換成HSV  再把HSV分開來
imghsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
h, s, v = cv2.split(imghsv)

histImgH = calcAndDrawHist(h, [255,0,0])
histImgS = calcAndDrawHist(s, [0,255,0])
histImgV = calcAndDrawHist(v, [0,0,255])
#分開後最後再把HSV合併在一起
out = cv2.cvtColor(imghsv, cv2.COLOR_HSV2BGR)
#印出HSV個別直方圖
cv2.imshow("histImgH", histImgH)
cv2.imshow("histImgS", histImgS)
cv2.imshow("histImgV", histImgV)
cv2.imshow("Image", out);
cv2.waitKey(0);
```

而在這裡，我把 H 以藍色表示，S 以綠色表示，V 以紅色表示

H 是控管色相，S 是控管飽和度，V 是控管明度

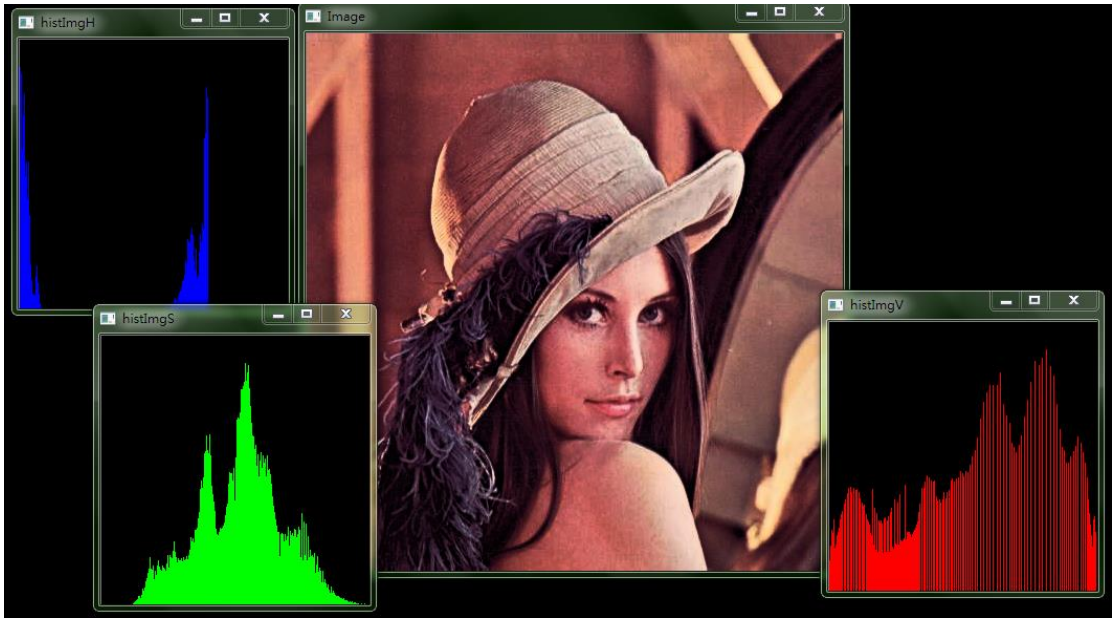


接著進行 V 的 equalization，也就是對明度進行等化

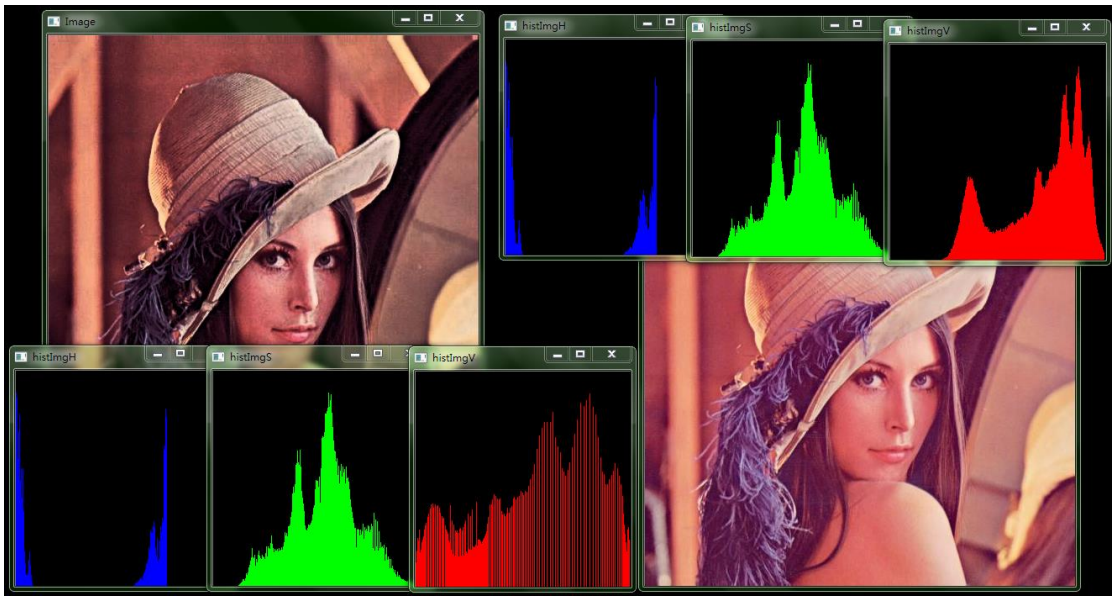
```
import cv2;
import numpy as np
#顯示出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;


#把圖片讀取進來
img = cv2.imread('D:\\Freddy\\vision\\mp1a.jpg')
#先把圖片轉換成HSV  再把HSV分開來
imghsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
h, s, v = cv2.split(imghsv)

#初始各項變數
bit = 256
high = len(v)
width = len(v[0])
sum = high*width
oldVdis = [0] * bit
oldVrat = [0] * bit
newVdis = [0] * bit
```

```python
#先計算V的各色階數量
for i in range(0,high,1):
    for j in range(0,width,1):
        oldVdis[v[i][j]]+=1
#計算各色階的比例  分布
for i in range(0,bit,1):
    oldVrat[i] = float(oldVdis[i])/sum

#透過老師教的equalization公式進行轉換
for i in range(0,bit,1):
    tempV = 0
    for j in range(0,i+1,1):
        tempV = tempV + oldVrat[j]

    newVdis[i] = round((bit-1) * tempV)

#把轉換好的色階覆蓋回去原本的色階
for i in range(0,high,1):
    for j in range(0,width,1):
        v[i][j] = newVdis[v[i][j]]

histImgH = calcAndDrawHist(h, [255,0,0])
histImgS = calcAndDrawHist(s, [0,255,0])
histImgV = calcAndDrawHist(v, [0,0,255])
#要再把HSV合併在一起  並從HSV轉回RGB
imghsv = cv2.merge([h, s, v])
out = cv2.cvtColor(imghsv, cv2.COLOR_HSV2BGR)

cv2.imshow("histImgH", histImgH)
cv2.imshow("histImgS", histImgS)
cv2.imshow("histImgV", histImgV)
cv2.imshow("Image", out);
#另存一個hsvnew的新圖片
cv2.imwrite('hsvnew.jpg',out);
cv2.waitKey(0);
cv2.destroyAllWindows();
```

等化 V 之後的結果如下，V 變得比原本更平均了



再來看一下原圖和等化後的圖，明暗度變的明顯清楚



(c) Y channel of YCbCr representation.

第 c 小題和前面的 a b 沒有太大的差別，只是改成把 YCrCb 的 Y 作等化

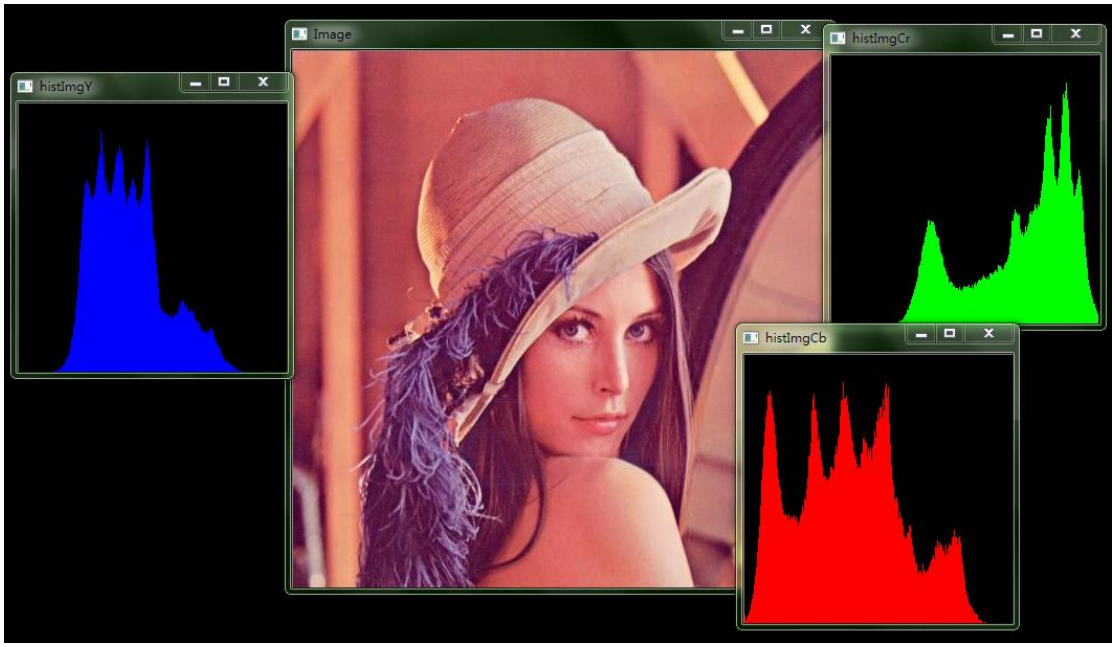
我還是一樣先把原圖的 YCrCb 直方圖印出來

```python
import cv2;
import numpy as np
#顯示出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;


#讀取圖片進來
img = cv2.imread('D:\\Freddy\\vision\\mp1a.jpg')
#先將圖片轉成YCrCb  再個別分開
imghsv = cv2.cvtColor(img, cv2.COLOR_BGR2YCR_CB)
Y, Cb, Cr= cv2.split(img)

histImgY = calcAndDrawHist(Y, [255, 0, 0])
histImgCb = calcAndDrawHist(Cb, [0, 255, 0])
histImgCr = calcAndDrawHist(Cr, [0, 0, 255])
#最後要將Y Cr Cb合併在一起
out = cv2.cvtColor(imghsv, cv2.COLOR_YCR_CB2BGR)

cv2.imshow("histImgY", histImgY)
cv2.imshow("histImgCb", histImgCb)
cv2.imshow("histImgCr", histImgCr)
cv2.imshow("Image", out);
cv2.waitKey(0);
```

顯示出的 Y Cr Cb 直方圖如下

在這 Y 以藍色表示，Cr 以綠色表示，Cb 以紅色表示

而 Y 主要是控管流明，Cr 是控管紅色色度，Cb 是控管藍色色度



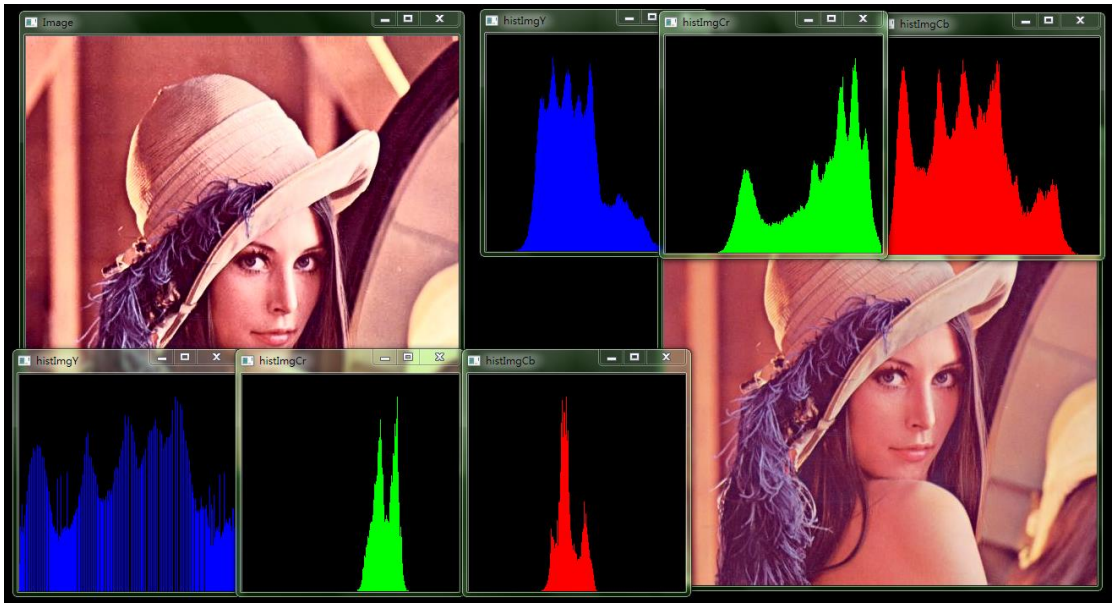
接著我們對 Y 進行 equalization

```python
import cv2;
import numpy as np
#顯示出直方圖的函式
def calcAndDrawHist(image, color):
    hist= cv2.calcHist([image], [0], None, [256], [0.0,255.0])
    minVal, maxVal, minLoc, maxLoc = cv2.minMaxLoc(hist)
    histImg = np.zeros([256,256,3], np.uint8)
    hpt = int(0.9* 256);

    for h in range(256):
        intensity = int(hist[h]*hpt/maxVal)
        cv2.line(histImg,(h,256), (h,256-intensity), color)

    return histImg;


#讀取圖片進來
img = cv2.imread('D:\\Freddy\\vision\\mp1a.jpg')
#先將圖片轉換成YCrCb  在個別分開
imgYCrCb = cv2.cvtColor(img, cv2.COLOR_BGR2YCR_CB)
Y, Cr, Cb = cv2.split(imgYCrCb)

#初始各項變數
bit = 256
high = len(Y)
width = len(Y[0])
sum = high*width
oldYdis = [0] * bit
oldYrat = [0] * bit
newYdis = [0] * bit
```

```
31
32  #先計算Y裡面各色階的數量
33  for i in range(0,high,1):
34      for j in range(0,width,1):
35          oldYdis[Y[i][j]]+=1
36  #計算各色階所佔比例  分布
37  for i in range(0,bit,1):
38      oldYrat[i] = float(oldYdis[i])/sum
39
40  #利用老師教的等化公式把原本色階做轉換
41  for i in range(0,bit,1):
42      tempY = 0
43      for j in range(0,i+1,1):
44          tempY = tempY + oldYrat[j]
45
46      newYdis[i] = round((bit-1) * tempY)
47
48  #把轉換好的色階覆蓋回去原圖的色階
49  for i in range(0,high,1):
50      for j in range(0,width,1):
51          Y[i][j] = newYdis[Y[i][j]]
52
53  histImgY = calcAndDrawHist(Y, [255,0,0])
54  histImgCr = calcAndDrawHist(Cr, [0,255,0])
55  histImgCb = calcAndDrawHist(Cb, [0,0,255])
56  #把Y Cr Cb合併起來  再轉換成RGB
57  imghsv = cv2.merge([Y, Cr, Cb])
58  out = cv2.cvtColor(imghsv, cv2.COLOR_YCR_CB2BGR)
59
60  cv2.imshow("histImgY", histImgY)
61  cv2.imshow("histImgCr", histImgCr)
62  cv2.imshow("histImgCb", histImgCb)
63  cv2.imshow("Image", out);
64  #另存一個名為YCrCbnew的圖片
65  cv2.imwrite('YCrCbnew.jpg',out);
66  cv2.waitKey(0);
67  cv2.destroyAllWindows();
```

將 Y 進行等化後的結果

在 Y 的部分比原圖平均了不少，再看一下原圖和等化後的結果，五官變得更立體明顯



Equalization 把圖片經過處理之後，讓各色階的分布平均化，使圖片變得清楚鮮明，不會偏向某一種色調而有模糊感