实验四　直方图图像处理技术

一、 实验目的

（1）学习图像直方图和累积直方图的计算方法

•使用 OpenCV 或 Numpy 函数计算直方图

•使用 Opencv 或者 Matplotlib 函数绘制直方图

•将要学习的函数有：cv2.calcHist()，np.histogram()

（2）掌握直方图均衡化与规定化的方法

（3）能够通过直方图均衡化对彩色图像进行处理

二、 实验原理

通过直方图可以对整幅图像的灰度分布有一个整体的了解。直方图的 x 轴是灰度值（例如，8bit 图像灰度级为 0 到 255 的整数），y 轴是图像中具有同一个灰度值的像素点的数目。

直方图均衡化是通过拉开密集出现的相邻的灰度级的距离（灰度级差值），压缩出现次数较少的灰度级的数量，来增强对比度。直方图均衡化可以视为,将已知图像的灰度直方图的灰度级的分布映射变换为均匀分布。

同理，直方图规定化，是将已知图像的直方图的灰度级分布映射为任意给出分布，达到改善图像对比对的效果。

三、 实验步骤

1.直方图的计算，绘制与分析

1.1 统计直方图

使用 OpenCV 统计直方图函数 cv2.calcHist 统计一幅图像的直方图。该函数和它的参数如下：

cv2:calcHist(images; channels; mask; histSize; ranges[; hist[; accumulate]])

images: 原图像（图像格式为 uint8 或 float32）。当传入函数时应该

用中括号[] 括起来，例如：[img]。

channels：同样需要用中括号括起来，它会告诉函数我们要统计那幅图像的直方图。如果输入图像是灰度图，它的值就是[0]；如果是彩色图像的话，传入的参数可以是[0]，[1]，[2]，分别对应着通道 B，G，R。

mask：掩模图像。要统计整幅图像的直方图就把它设为 None。但是如果只需统计图像某一部分的直方图，就需要制作一个掩模图像。

histSize:BIN 的数目。也应该用中括号括起来，例如：[32],[128],[256]。

ranges：像素值范围，通常为[0，256]

以灰度格式加载一幅图像并统计图像的直方图，步骤如下：

```
img = cv2.imread('imagefile.jpg',0)
# 别忘了中括号[img],[0],None,[256],[0,256]，只有 mask 没有中括号
hist = cv2.calcHist([img],[0],None,[256],[0,256])
```

使用**Numpy** 统计直方图Numpy 中的函数**np.histogram()** 也可以帮我们统计直方图。可以尝试如下代码：

```
#img.ravel() 将图像转成一维数组，这里没有中括号。
 hist,bins = np.histogram(img.ravel(),256,[0,256])
```

1.2 绘制直方图

使用 Matplotlib 中的绘图函数绘制灰度直方图，函数如下：

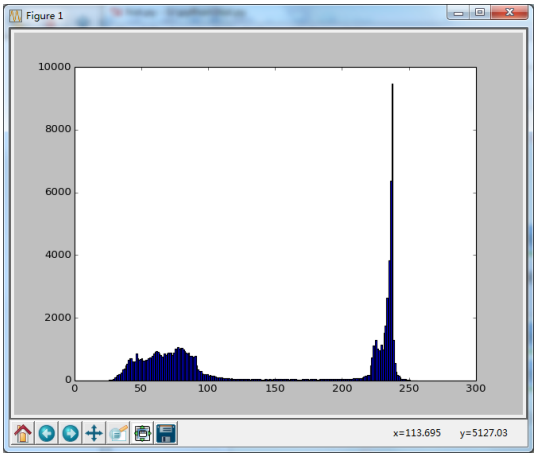
matplotlib.pyplot.hist()

使用函数 **calcHist()** 或 **np.histogram()**统计直方图后，可以使用 **matplotlib.pyplot.hist()**函数直接统计并绘制直方图。代码如下：

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

```
img = cv2.imread('flower3.jpg',0)
plt.hist(img.ravel(),256,[0,256]);
plt.show()
```

原图



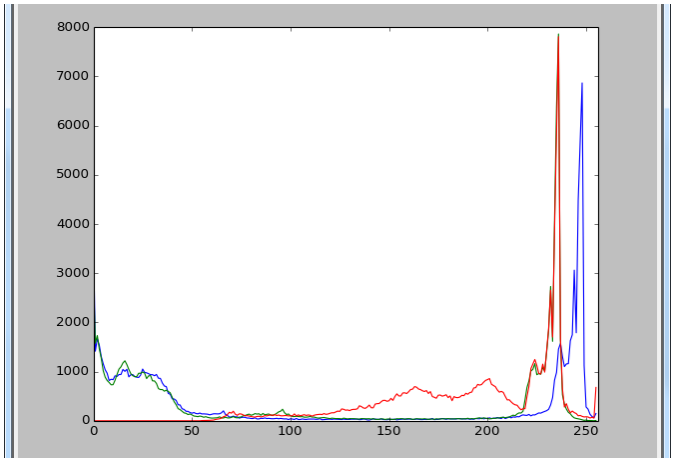
直方图

或者只使用 matplotlib 的绘图功能，这在同时绘制多通道（BGR）的直方图。下面的代码：

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread( 'flower3.jpg')
color = ('b','g','r')
# 对一个列表或数组既要遍历索引又要遍历元素时
# 使用内置 enumerrate 函数会有更加直接，优美的做法
#enumerate 会将数组或列表组成一个索引序列。
# 使我们再获取索引和索引内容的时候更加方便
for i,col in enumerate(color):
    histr = cv2.calcHist([img],[i],None,[256],[0,256])
    plt.plot(histr,color = col)
```

```
        plt.xlim([0,256])
    plt.show()
```

结果：



使用 OpenCV 使用 OpenCV 自带函数绘制直方图，可以参考 OpenCV-Python2 的官方示例。

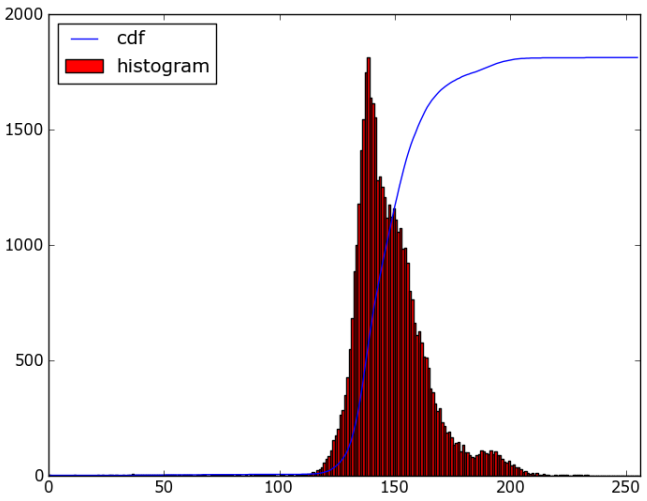
1.3 累积直方图计算与绘制

累积直方图值是小于等于某个灰度级的像素点在图像中出现的总次数或比例。

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('wiki.tif',0)
#flatten() 将数组变成一维
hist,bins = np.histogram(img.flatten(),256,[0,256])
# 计算累积分布图
cdf = hist.cumsum()
cdf_normalized = cdf * hist.max()/cdf.max()
```

```
plt.plot(cdf_normalized, color = 'b')
plt.hist(img.flatten(),256,[0,256], color = 'r')
plt.xlim([0,256])
plt.legend(('cdf','histogram'), loc = 'upper left')
plt.show()
```



实验提示：

可以通过单步调试方法在使用 Spyder 中变量观察器查看 cdf_normalized 变量的值。如何显示一个以百分比表示的累积直方图和直方图？

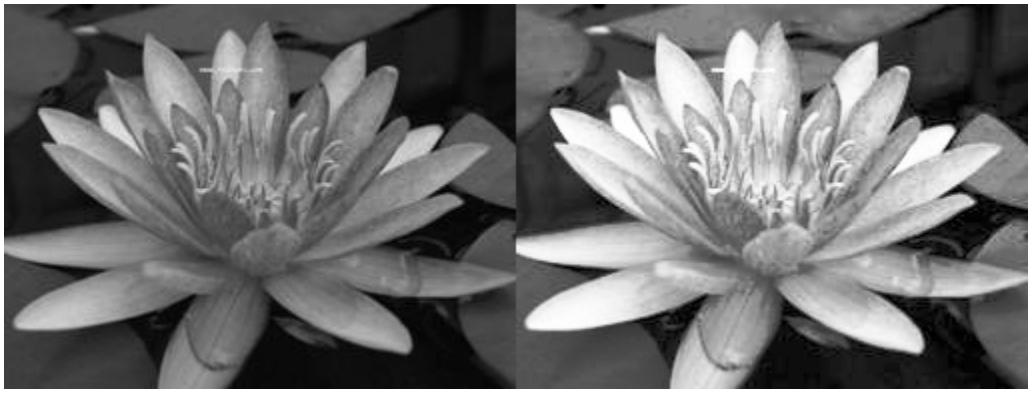
2. 直方图均衡化

2.1 灰度图像均衡化

OpenCV 中的直方图均衡化函数为 cv2.equalizeHist()。这个函数的输入图片仅仅是一副灰度图像，输出结果是直方图均衡化之后的图像。下边的代码是对图像进行直方图均衡化处理：

```
import cv2
import numpy as np
img = cv2.imread('flower2.jpg',0)
equ = cv2.equalizeHist(img)
```

```
res = np.hstack((img,equ))
cv2.imwrite('hist.jpg',res)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



(请使用 Fig5 试一下！)

四、 实验报告要求

1. 描述实验的基本步骤，用数据和图片给出各个步骤中取得的实验结果并进行必要的讨论。

2. 必须包括原始图像及其计算处理后的图像以及相应的解释。

五、 思考题

1.给出彩色图像 Fig6，请在 HSI 空间对图像的亮度分量进行均衡化，观察彩色图像效果。

2.对同一幅图像 Fig6 在 B、G、R 空间逐一做均衡化处理，观察最终的图像与思考题 1 中的结果异同。

3.编程实现直方图规定化的处理程序；给定图像 Fig7A 和图像 Fig7B，把 Fig7A 图像直方图规范化为接近 Fig7B 图像直方图的分布。

六、图像示例：

Fig1

Fig2

Fig3

Fig4

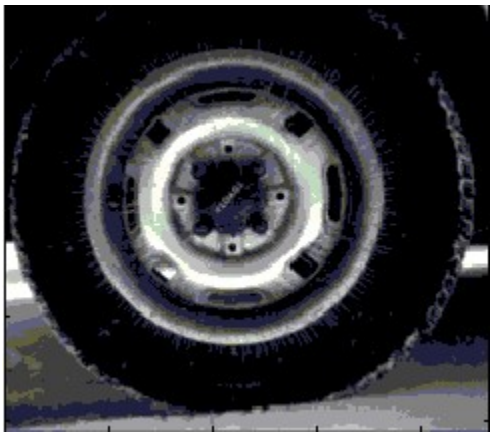


Fig5

Fig6

Fig7A

Fig7B