实验 1　OpenCV

姓名　学号　班级

日期

1　实验概览

1.1　实验内容

本次实验要求将三幅图像分别以彩色、灰度方式读入，并计算颜色直方图、灰度直方图和梯度直方图。

1.2　实验原理

1. 彩色直方图

   彩色图像的每个像素由 R、G、B 三种颜色分量组成，每种颜色具有 $[0, 255]$ 中的某种强度，记作 $I(x, y, c)$.

   则可以算出某一颜色分量的总能量：

$$E(c) = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} I(x, y, c).$$

   进而得出该颜色分量能量的相对比例：

$$H(c) = \frac{E(c)}{\sum_{i=0}^{2} E(i)}.$$

2. 灰度直方图

   灰度图像的每个像素有灰度值，范围为 $[0, 255]$.

   灰度值为 i 的像素个数为

$$N(i) = \sum_{x=0}^{W_1} \sum_{y=0}^{H-1} I(x, y) == i ? 1 : 0.$$

   进而得到灰度直方图

$$H(i) = \frac{N(i)}{\sum_{j=0}^{255} N(j)}.$$

3. 梯度直方图

$I(x,y)$ 表示一幅灰度图像，x 方向的梯度为

$$I_x(x,y) = \frac{\partial I(x,y)}{\partial x} = I(x+1,y) - I(x-1,y).$$

同理也有 $I_y(x,y)$, 梯度强度定义为

$$M(x,y) = \sqrt{I_x^2(x,y) + I_y^2(x,y)}.$$

易得梯度的范围是

$$[0, 255\sqrt{2}] \approx [0, 361].$$

落在第 i 个梯度区间的像素个数

$$N(i) = \sum_{x=1}^{W-2} \sum_{y=2}^{H-2} \lfloor M(x,y) \rfloor == i ? 1 : 0.$$

进而得到梯度直方图

$$H(i) = \frac{N(i)}{\sum_{j=0}^{360} N(j)}.$$

2 解决思路

1. 彩色直方图

将图片以彩色模式读入，并转换成 RGB 模式。

```
img_color = cv2.imread(IMG_DIR + img_file, cv2.IMREAD_COLOR)
img_rgb = cv2.cvtColor(img_color, cv2.COLOR_BGR2RGB)
```

分别统计各通道的总和，并计算占比。

```
totals = img_rgb.sum(axis=(0,1)).astype('float64')
proportions = totals / totals.sum()
```

2. 灰度直方图

将图片以灰度模式读入。

```
img_gray = cv2.imread(IMG_DIR + img_file, cv2.IMREAD_GRAYSCALE)
```

使用库函数 `cv2.calcHist` 直接生成直方图。

3. 梯度直方图

将图片以灰度模式读入，按公式分别计算 x 和 y 方向上的梯度，并得到梯度强度。

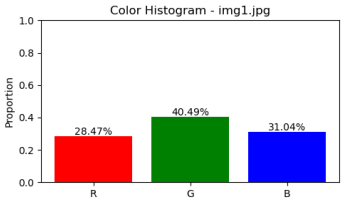
```
for y in range(1, img_gray.shape[0]-1):
    for x in range(1, img_gray.shape[1]-1):
        grad_x[y, x] = img_gray[y, x+1] - img_gray[y, x-1]
        grad_y[y, x] = img_gray[y+1, x] - img_gray[y-1, x]

grad_mag = np.sqrt(grad_x**2 + grad_y**2)
```

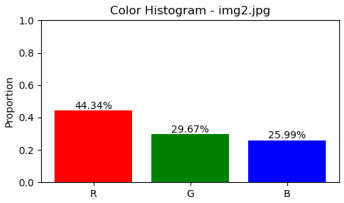
使用库函数 `cv2.calcHist` 生成直方图。

3 运行结果

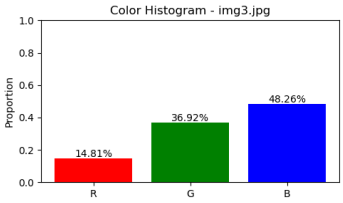
1. 彩色直方图



(a) 图 1



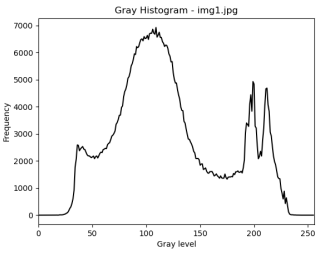
(b) 图 2



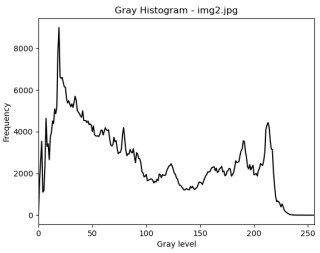
(c) 图 3

图 1: 彩色直方图运行结果

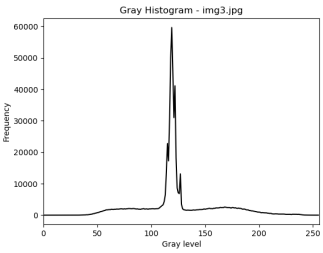
2. 灰度直方图



(a) 图 1



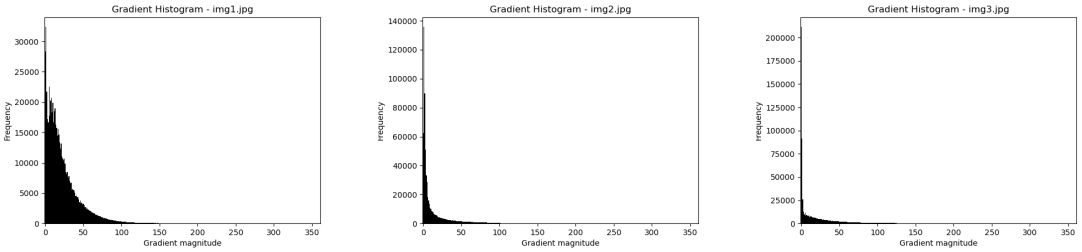
(b) 图 2



(c) 图 3

图 2: 灰度直方图运行结果

3. 梯度直方图



(a) 图 1 (b) 图 2 (c) 图 3

图 3: 梯度直方图运行结果

4 结果分析与思考

4.1 结果分析

1. 对于彩色直方图，三张图中占比最大的颜色通道分别是绿色、红色、蓝色，三张图的整体色调也是绿、红、蓝为主，说明彩色直方图可以反映图像的整体色调。

2. 对于灰度直方图，

   - 图一中灰度集中于中段区域，在较大的区域有两个波峰。说明画面整体比较明亮，同时有部分区域非常亮（即图一中的天空）；
   - 图二中灰度集中于较小的区域，画面也整体较暗；
   - 图三中灰度集中于中段偏小的区域，画面亮度适中。

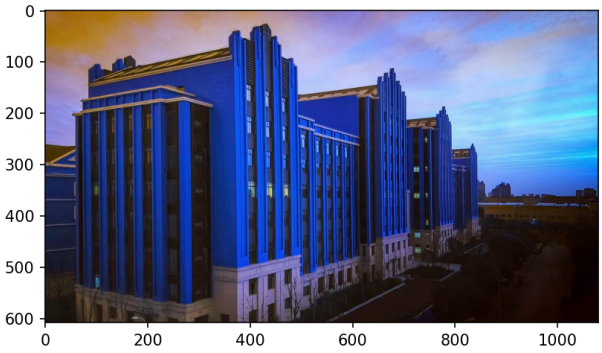
   说明灰度直方图可以反映图像的明暗分布。

3. 对于梯度直方图，由图一到图三越来越集中于靠近 0 的区域。从纹理复杂程度上说，从图一到图三也是越来越简单，说明梯度直方图可以反映图像的纹理复杂程度。

4.2 思考题解答

1. 示例代码中的 `cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)` 的作用是什么？

   答：将读入图像的颜色空间由 BGR 转换成 RGB，防止 pyplot 等外部包显示出错误的颜色（蓝、红颠倒）



   如上图，未加入此行代码，晚霞照射下的理科群楼变成了蓝色。

2. 如何使得 pyplot 正确显示灰度图的颜色？

   答：可以使用 `pyplot.imshow` 的内置参数 `cmap='gray'`。

5 实验感想

一开始在计算梯度时直接使用了 `cv2.Sobel` 函数，并制定 `ksize=3` 参数。后经查询文档，发现 `cv2.Sobel(ksize=3)` 实际上使用了 Sobel 算子，具体为：

$$I_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \boldsymbol{A}, \qquad I_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \boldsymbol{A}.$$

即：

$$\begin{aligned} I_x(x,y) = 2\left[I(x-1,y) - I(x+1,y)\right] \\ + I(x-1,y+1) - I(x+1,y+1) \\ + I(x-1,y-1) - I(x+1,y-1) \end{aligned}$$

与本次实验指定的 $I_x(x,y) = I(x+1,y) - I(x-1,y)$ 有较大区别。

因此，转而使用手动方法计算梯度：

```
for y in range(1, img_gray.shape[0]-1):
    for x in range(1, img_gray.shape[1]-1):
        grad_x[y, x] = img_gray[y, x+1] - img_gray[y, x-1]
        grad_y[y, x] = img_gray[y+1, x] - img_gray[y-1, x]
```

对库的实现不了解时不应贸然使用其内置函数，而是要先弄清函数的具体含义。