实验八报告

唐灵
519030910052
F1903002

2020 年 11 月 19 日

摘要

这是电工导 c 课程的第七次实验

1 实验概览

本次实验主要应用 cv2 对于原始图片进读取，利用 numpy 对于我们希望得到的图片的基本特征进行处理以及抽取，最终利用 plt 对于我们提取到的特征进行可视化。

2 实验环境

本次实验的所有代码在电类工程导论 c 课程中在课程方统一给定的“ee208” *Docker* 容器中运行并实现。

3 练习解决思路

3.1 练习一的解决思路

练习一要求作出 RGB 直方图，选择读取图像过后，将先将图片转化为 RGB 通道的图片。直接统计三个通道的总和，计算频率，可以简单通过 numpy 函数的 sum 函数得到：

```python
img_bgr = cv2.imread('images/img{}.jpg'.format(i))
img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)
#得到数据
feature_rgb_total = np.sum(img_rgb,axis=(0,1))
feature_rgb = np.array([i / np.sum(feature_rgb_total) for i in feature_rgb_total])
```

图 1: 得到 RGB 频数信息

之后再使用 plt 作直方图即可，值得注意的是应该考虑设定相当的颜色。

3.2 练习二的解决思路

练习二首先要求，读取灰度图，这个过程可以直接通过 cv2 实现，为方便作出频率直方图（plt.hist 函数只能处理一维的 narray 数据），所以无论是在作梯度图还是在作灰度直方图的时候，都应该考虑将二维的图片“压”为一维的数组。

对于灰度图直接作图即可，对于 hist 参数的选定，多考虑一下，比如归一化，将区间设定为整数等等。

```python
plt.hist(feature_gray,bins=range(256),density = True)
```

图 2: 作灰度直方图

而对于梯度直方图而言，必须对于图片本身进行处理，我的处理方式分三个步骤：

1. 首先必须先对于图片本身的数据类型进行强制转化，因为 cv2 默认读取进来的是非负整型，没有办法进行接下来的运算
2. 通过两个循环体，每个循环体的每一次循环将计算一行或者一列元素的梯度
3. 将计算出的梯度放到列表中，最后再次转化为 narray 对象，
4. 将对象的边缘进行切割，去除我们不需要的部分
5. 其中列方向显然需要进行转置，才能进行下一步的处理
6. 将两个方向分别得到的梯度图像进行平方和并开方的处理，得到最终的梯度图像

最终的实际的代码相对来说还是比较简洁：

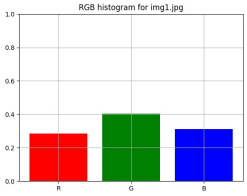
```
#将数组类型转化为浮点，方便处理
img_gray = img_gray_int.astype(np.float)
#得到高度
hight = img_gray.shape[0]
width = img_gray.shape[1]
#得到水平梯度梯度和竖直梯度图，并切割掉边缘不用的元素
gy = np.array([img_gray[i+1,:]-img_gray[i-1,:] for i in range(1,hight-1)])[:,1:width-1]
gx = np.array([img_gray[:,j+1]-img_gray[:,j-1] for j in range(1,width-1)])[:,1:hight-1].transpose()
#对于水平梯度和竖直梯度进行处理，并压平
feature_grad = np.floor((gy**2 + gx**2)**0.5).flatten()
```

图 3: 提取梯度信息

4 代码运行结果

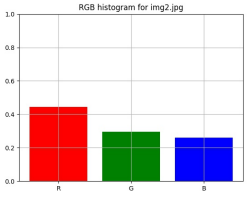
4.1 练习一的运行结果

(a) img1-原图

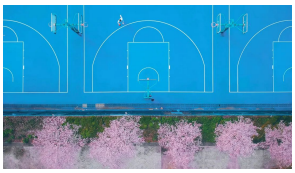


(b) img1-RGB 直方图

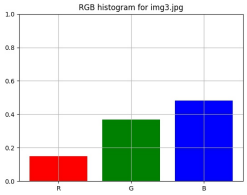
(c) img2-原图



(d) img2-RGB 直方图



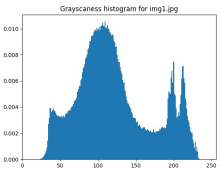
(e) img3-原图



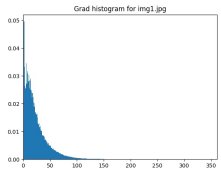
(f) img3-RGB 直方图

4.2 练习二的运行结果

(g) img1-原图

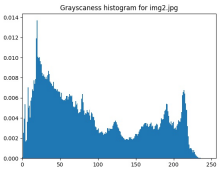


(h) img1-灰度直方图

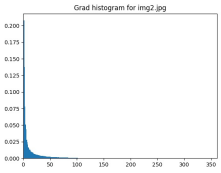


(i) img1-灰度梯度直方图

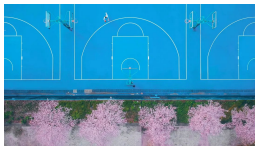
(j) img2-原图



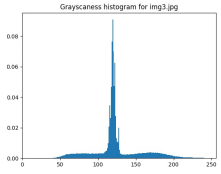
(k) img2-灰度直方图



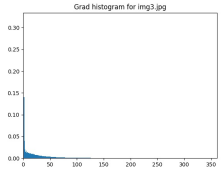
(l) img2-灰度梯度直方图



(m) img3-原图



(n) img3-灰度直方图



(o) img3-灰度梯度直方图

4.3 思考题的运行结果

对于思考题，对 plt.imshow 的函数采用不同的参数可以得到我们希望看到的灰度图的图像

```
plt.imshow(img_gray,cmap="gray")
```

图 4: 添加参数

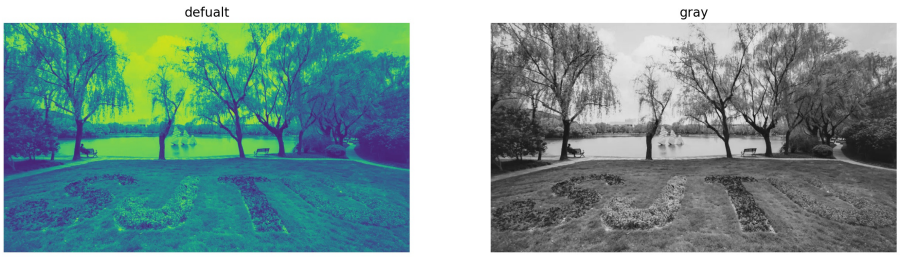


图 5: 左图为直接显示，右图为添加了参数的结果

5 思考题

1. 由于 cv2 默认读取图片的方式是 BGR，但 plt 的 imshow 函数又是默认进行 RGB 展示，所以这个函数是将图片从 BGR 模式转化为 RBG 式，当然，也可以通过 numpy 自身的切片操作来实现，具体不再赘述。本质就是将外层三个数组的顺序颠倒而已。

2. 通过简单设定 plt 的显示参数可以进行实现，具体效果在上一个“代码运行板块中已经进行了可视化说明”