《数字图像处理》实验报告

姓名： 汪雨卿 学号：19120191

实验四

一. 任务 1

自己实现一个 Gamma 变换（考虑是否使用查找表），和程序库自带的版本（Matlab、Pillow、OpenCV 或者其他图像库）进行变换结果和计算效率的比较。至少在“light.tif”“ dark.png”上进行测试。

a) 核心代码：

方法一：建立自己的查找表，并且利用查找表进行变换。*实现对 light.tif 进行转换输出 gamma_table_light.jpg*

1. 利用公式，建立自己的查找表

```
# 方法一：建立查找表，替换
# 建立自己的查找表
def build_table(table, gamma):
    for i in range(255):
        temp = double(i) / 255
        temp = pow(temp, gamma)
        table[0, i] = temp * 255
    return
```

2. 对于读取的照片三个分度的每一个像素，找到查找表中对应的值进行替换。

```
def produce_img_gamma(gamma, img):
    chart = np.zeros((1, 256))  # 利用numpy一个256个元素的数组
    build_table(chart, gamma)  # 构造gamma查找表
    # img = to_grey(img)
    rows, cols, counts = img.shape
    for i in range(rows):
        for j in range(cols):
            for k in range(counts):
                img[i, j, k] = chart[0, img[i, j, k]]
    return img
```

方法二：直接进行数据替换。*实现对 dark.png 进行转换输出 gamma_direct_dark.jpg*

1. 对于读入的 Img 的每个像素点直接进行 gamma 变换

```python
# 方法二：直接替换
def gamma_direct(gamma, img):
    direct = img.copy()
    direct /= 255.
    direct = pow(direct, gamma)
    direct *= 255
    direct = direct.astype(np.uint8)
    return direct
```

2. 读入照片，并设置数据的类型

```python
def produce_dark():
    img = cv.imread("dark.png").astype(np.float64)
    result = gamma_direct(0.5, img)
    # cv.imshow("gamma_direct", result)
    # cv.waitKey(0)
    cv.imwrite("gamma_direct_dark.jpg", result)
```

方法三：利用外部库 skimage 实现 gamma 变换

```python
    result = exposure.adjust_gamma(img, 2)
```

b) 实验结果截图

原图：

自己写的 gamma 变换

调库做的 gamma 变换

原图：

自己写的 gamma 变化

调库做的 gamma 变化

效率比较：

```
查找表light花费的时间：0.9543466567993164
直接dark花费的时间：0.023082733154296875
sk库light花费的时间：0.019915103912353516
sk库dark花费的时间：0.018949270248413086
```

二． 任务 2

自己实现直方图均衡化，并和所选图像库中函数进行结果对比，并绘出变换前后的直方图。至少在如下图像上测试：school.png, baby.png, hill.jpg

a) 核心代码：

方法一：建立自己的查找表，并且利用查找表进行变换。*实现对 school.png, baby.png, hill.jpg 进行转换输出 my_school.jpg, my_baby.jpg, my_hill.jpg*

1. 统计并建立灰度表

```python
def origin_grey(img):
    gram = {}
    row, col = img.shape[:2]
    # 统计灰度表
    for i in range(row):
        for j in range(col):
            k = img[i][j]
            if k in gram:
                gram[k] += 1
            else:
                gram[k] = 1
    # 建立排序后的灰度表
    sort_gram = {}
    sort_list = sorted(gram)
    for j in range(len(sort_list)):
        sort_gram[sort_list[j]] = gram[sort_list[j]]
    return sort_gram
```

2. 建立概率的分布表，并且进行转换

```python
def equ_grey(gram, img):
    rols, cols = img.shape[:2]
    # 建立概率分布的表
    pcharm = {}
    for i in gram.keys():
        pcharm[i] = gram[i] / (rols * cols)

    temp = 0
    for j in pcharm.keys():
        temp += pcharm[j]
        pcharm[j] = max(gram) * temp

    out_img = np.zeros(shape=(rols, cols), dtype=np.uint8)

    for m in range(rols):
        for n in range(cols):
            out_img[m][n] = pcharm[img[m][n]]
    return out_img
```

方法二：调库直方图均衡化。*实现对 school.png, baby.png, hill.jpg 进行转换输出。*

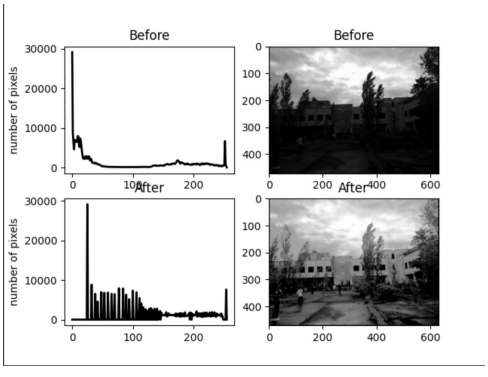
```
def mat_change(img):
    return np.hstack((img, cv.equalizeHist(img)))
```

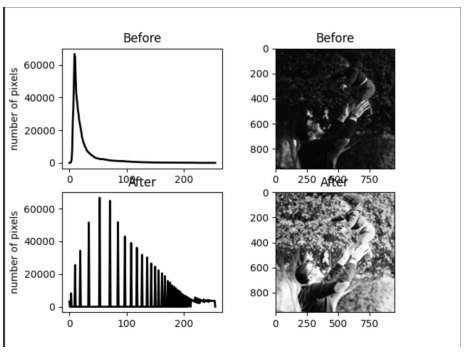
3. 绘出变换前后的直方图

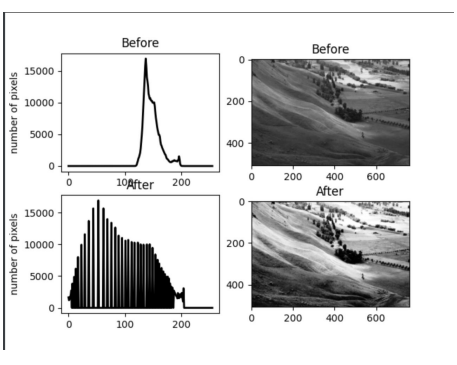
```
46
47  def myHist(img):
48      rols, cols = img.shape[:2]
49      myHist = np.zeros([256], np.uint64)
50      for i in range(rols):
51          for j in range(cols):
52              myHist[img[i][j]] += 1
53      return myHist
```

```
def draw_gram(originHist, newHist, img, after):
    x = np.arange(256)
    # 绘制灰度直方图
    plt.figure(num=1)
    plt.subplot(2, 2, 1)
    plt.plot(x, originHist, 'r', linewidth=2, c='black')
    plt.title("Before")
    plt.ylabel("number of pixels")
    plt.subplot(2, 2, 3)
    plt.plot(x, newHist, 'r', linewidth=2, c='black')
    plt.title("After")
    plt.ylabel("number of pixels")
    plt.subplot(2, 2, 2)
    plt.imshow(img, cmap=plt.cm.gray)
    plt.title('Before')
    plt.subplot(2, 2, 4)
    plt.imshow(after, cmap=plt.cm.gray)
    plt.title('After')
    plt.show()
```

b) 实验结果截图







利用库实现

效率：

```
my_school花费的时间：0.2559645175933838
my_baby花费的时间：0.8074929714202881
my_hill花费的时间：0.33734607696533203
my_mat_school花费的时间：0.012963294982910156
my_mat_baby花费的时间：0.041925668716430664
my_mat_hill花费的时间：0.016948461532592773
```

c) 实验小结

本次实验通过编写程序实现 gamma 变换以及直方图均衡变化，不仅让我理解了这两种变化背后数学方法的应用和处理，也通过实际操作让我学习了如何利用代码实现平时 p 图工具中的一些图像处理功能。此外，我也学习了如何利用 python 的库绘制想要的图标。