



数字图像处理与机器视觉 实验报告

作业名称 HW1 Problem2

姓 名 杨逍宇

学 号 3220105453

电子邮箱 3220105453@zju.edu.cn

联系电话 13518290755

导 师 蔡声泽/曹雨齐/姜伟



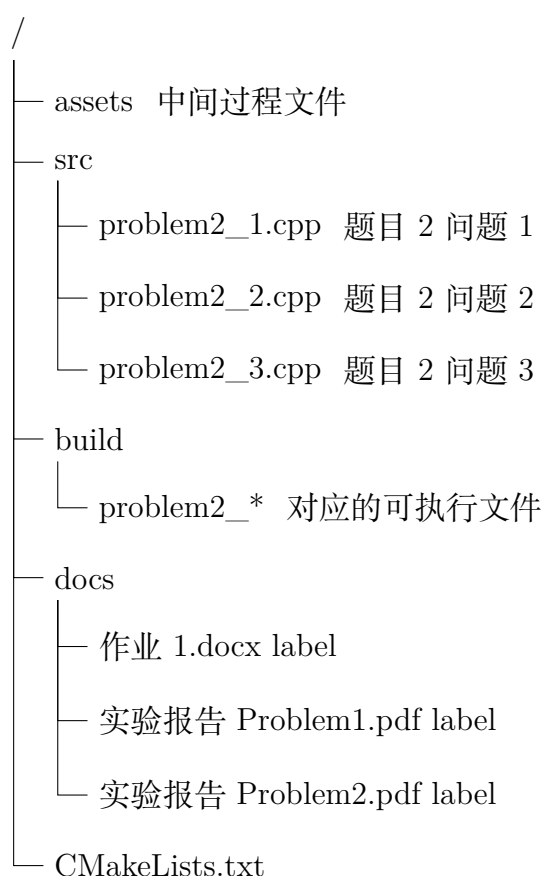
2025 年 3 月 7 日

1 已实现的功能简述及运行简要说明

1.1 已实现的功能简述:

- (1). `problem2_1.cpp`, `problem2_2.cpp`, `problem2_3.cpp` 分别实现了题目二的三个问题。
- (2). 运行相关的可执行文件, 会显示中间运行过程中的图像, 并将中间过程文件保存在 `assets` 文件夹中。

项目目录树如下:



2 开发与运行环境

本实验使用的软件和工具如下:

- 开发环境: Visual Studio Code on Ubuntu22.04
- 编程语言: C++
- 库: OpenCV 4.7.0
- 构建工具: CMake

3 算法基本思路

使用 opencv 库进行相关的图像操作。

本作业中涉及以下关键步骤：

- (1). **直方图均衡化**：使用 `cv::equalizeHist` 函数来进行直方图均衡化。我是用 HSV 颜色空间，为了不让图像失真，仅对亮度（V 通道）进行均衡化，保留色调（H）和饱和度（S）。编写 `GetHistogram` 函数，计算直方图并进行归一化实现直方图的绘制。

```
1  Mat GetHistogram(const Mat& image)
2  {
3      // 计算直方图
4      Mat hist;
5      int channels[] = {0};
6      int histSize[] = {256};
7      float range[] = {0, 256};
8      const float* ranges[] = {range};
9      calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
10
11     // 归一化直方图以便显示
12     normalize(hist, hist, 0, 400, NORM_MINMAX); // 400为显示高度
13
14     // 创建直方图画布
15     Mat histImage(400, 512, CV_8UC3, Scalar(255, 255, 255));
16
17     // 绘制直方图
18     for (int i = 0; i < 256; i++) {
19         rectangle(histImage, Point(i * 2, 400 - cvRound(hist.at<float>
20             >(i))),
21             Point((i + 1) * 2, 400), Scalar(0, 0, 255), -1);
22     }
23     return histImage;
24 }
```

- (2). **直方图匹配**：编写 `MatchHistogram` 函数，分别计算图像直方图并进行归一化，然后计算累积分布函数 CDF，构建映射表，最后应用映射表，实现直方图匹配。

```
1  void MatchHistogram(const Mat& src, const Mat& target, Mat& matched)
```

```
2  {
3      // 计算源图像和目标图像的直方图
4      Mat src_hist, target_hist;
5      const float range[] = {0, 256};
6      const float* ranges[] = {range};
7
8      int channels[] = {0};
9      int histSize[] = {256};
10
11     calcHist(&src, 1, channels, Mat(), src_hist, 1, histSize, ranges)
12         ;
13     calcHist(&target, 1, channels, Mat(), target_hist, 1, histSize,
14         ranges);
15
16     // 归一化直方图
17     normalize(src_hist, src_hist, 0, 1, NORM_MINMAX);
18     normalize(target_hist, target_hist, 0, 1, NORM_MINMAX);
19
20     // 计算累积分布函数 (CDF)
21     Mat src_cdf = src_hist.clone(), target_cdf = target_hist.clone();
22     for (int i = 1; i < 256; i++) {
23         src_cdf.at<float>(i) += src_cdf.at<float>(i - 1);
24         target_cdf.at<float>(i) += target_cdf.at<float>(i - 1);
25     }
26
27     // 构建映射表
28     Mat lut(1, 256, CV_8U);
29     for (int i = 0; i < 256; i++) {
30         float min_diff = 1.0;
31         for (int j = 0; j < 256; j++) {
32             float diff = fabs(src_cdf.at<float>(i) - target_cdf.at<
33                 float>(j));
34             if (diff < min_diff) {
35                 min_diff = diff;
36                 lut.at<uchar>(i) = j;
37             }
38         }
39     }
40 }
```

```
37  
38     // 应用映射表  
39     LUT(src, lut, matched);  
40 }
```

通过上述步骤，程序实现了对图像直方图均衡化和直方图匹配操作。

4 实验结果及分析

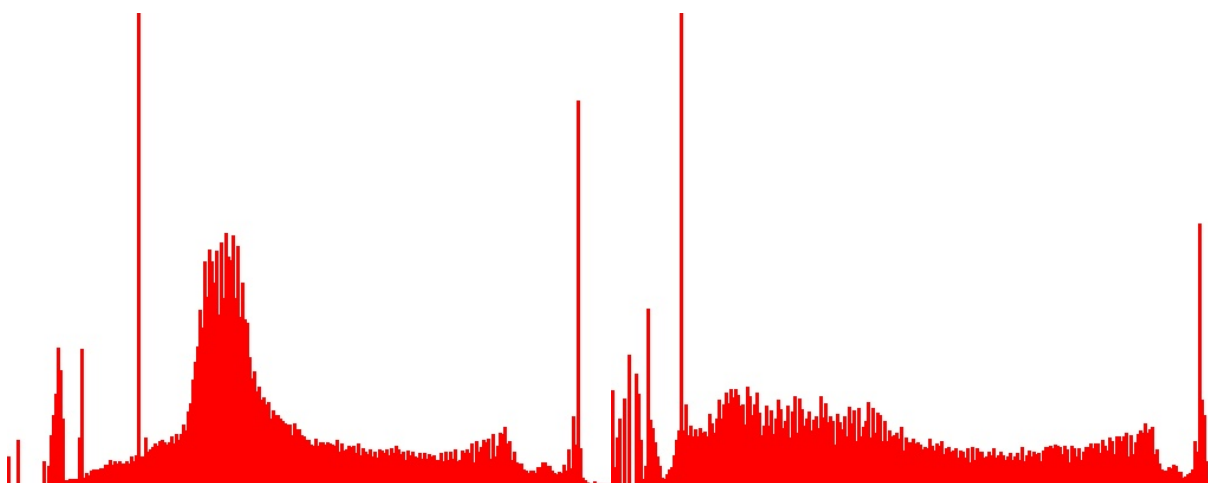
读取一幅彩色图像，然后对其直方图均衡化，得到对比效果如下：



origin image

equalized image

均衡化前后的直方图对比如下：



origin histogram1

equalized histogram1

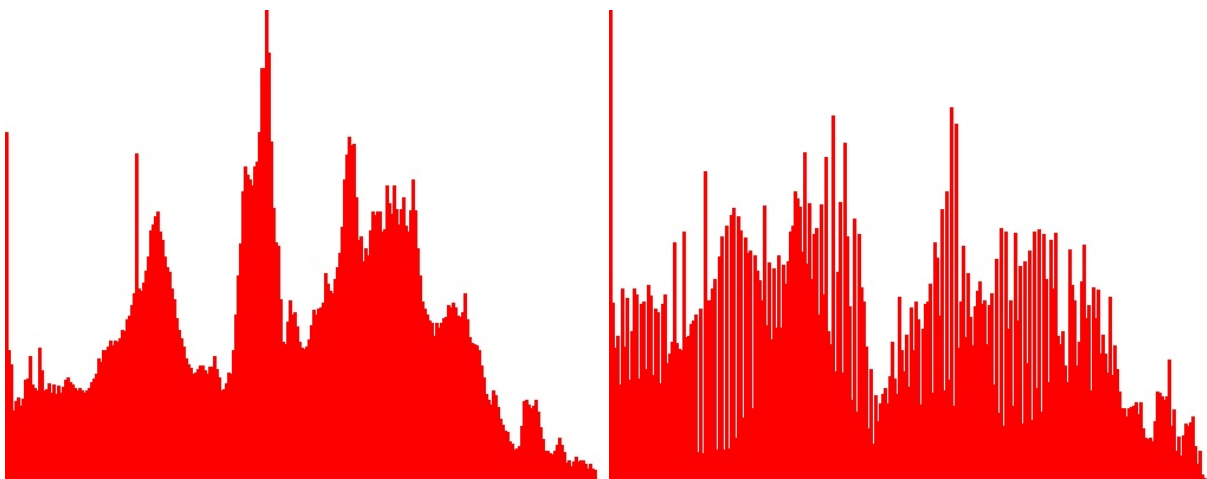
从中可以看到直方图相比之前更加均衡，分布更加均匀。然后我们提供的图像素材进行直方图均衡化，得到效果如下：



origin image

equalized image

均衡化前后的直方图对比如下：



origin histogram2

equalized histogram2

然后我们进行两张图的直方图匹配操作，为了效果显示更加明显，我选择使用灰度图来进行匹配操作，图像 src 和 target 如下：

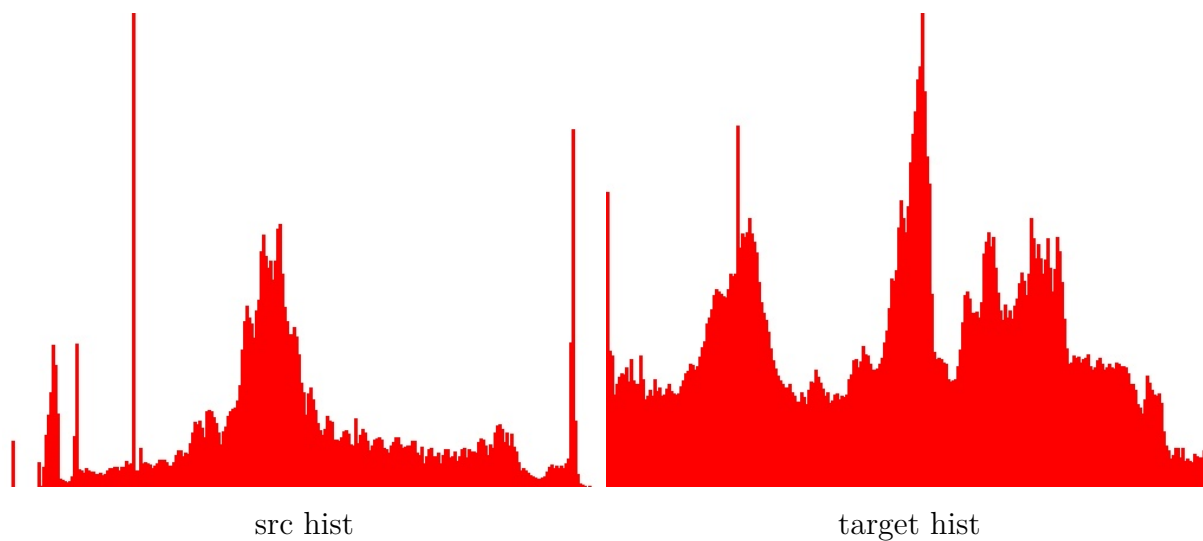


得到的图像如下：



Figure 1: matched image

可见匹配后的图像相较原始图像更加暗，更加偏向 target 的灰度分布。图像 src 和 target 对应的直方图如下：



得到匹配后的直方图如下：

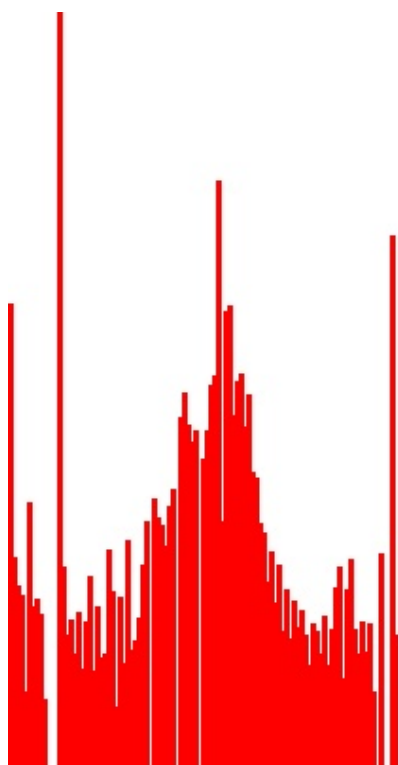


Figure 2: matched hist

5 结论与心得体会

在本次实验中，我实现了对图像进行直方图均衡和两张图的直方图匹配，受益匪浅。