数字图像处理实验报告
基于 SIFT 和 RANSAC 算法的图像拼接

万子牛

2015011011

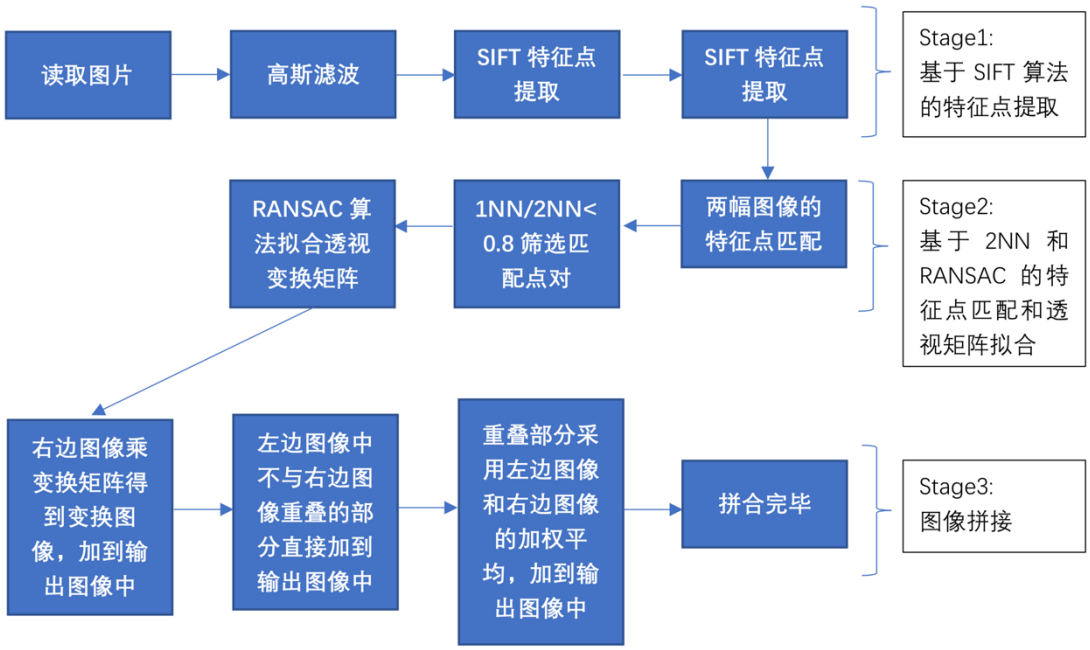
无 52

一、实现环境

操作系统：MacOS

开发环境：Python 2.7.10 + OpenCV 3.4.1 + **OpenCV-Contrib 3.4.1**

二、流程框图



部分流程说明及参数说明：

1. **卷积核**

   大小(5,5)，方差为 1

2. **筛选匹配点对的方法**

1NN/2NN<0.8。即对源图像的每一个特征点，在目标图像中找到距离和它最近的前 2 个特征点。两个特征点的距离即两个特征点的 128 维特征向量的欧式距离，即源图像的某一特征点 P1 和目标图像的某一特征点 P2 的特征向量分别为$\overrightarrow{p_1}$和$\overrightarrow{p_2}$，则其距离为：

$$d = \overrightarrow{p_1}^T \overrightarrow{p_2}$$

假设在目标图像中，离源图像中特征点P_1最近的特征点为P_{21}，第二近的特征点为P_{22}，P_1和它们的距离分别d_1和d_2，则只有当：

$$\frac{d_1}{d_2} < 0.8$$

时，认为P_1和P_{21}是一对“好”的匹配特征点对，将其加入好的匹配特征点对序列中。其中，阈值 0.8 取自 *David G.Lowe* 的论文。

3. **RANSAC 算法**

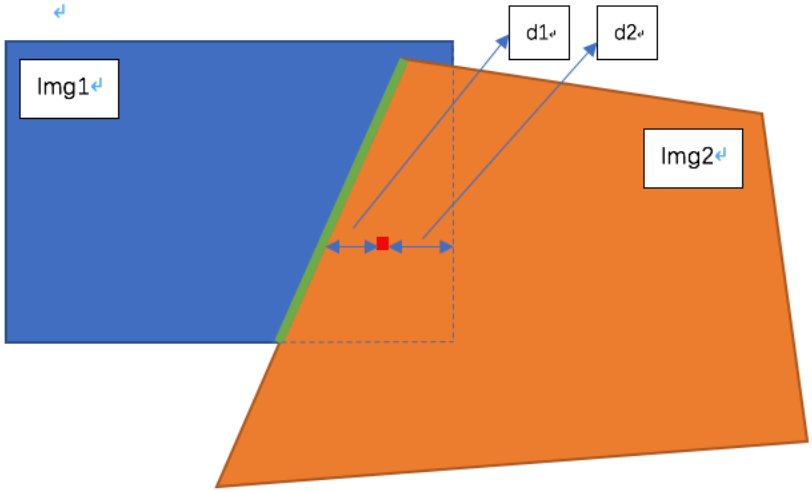
   最高迭代次数为 2000，置信度 0.997。得到一个3×3的透视转换矩阵。

4. **图像拼合重叠部分的加权平均**

   采用下列公式计算重叠部分任意像素值：

$$pixel = \frac{d_1}{d_1 + d_2} \times pixel_{img1} + \frac{d_2}{d_1 + d_2} \times pixel_{img2}$$

   其中，d_1为该待求像素点坐标到两幅图像分界线处的水平距离，d_2为该待求像素点到左边图像右边缘的距离。如下图所示：

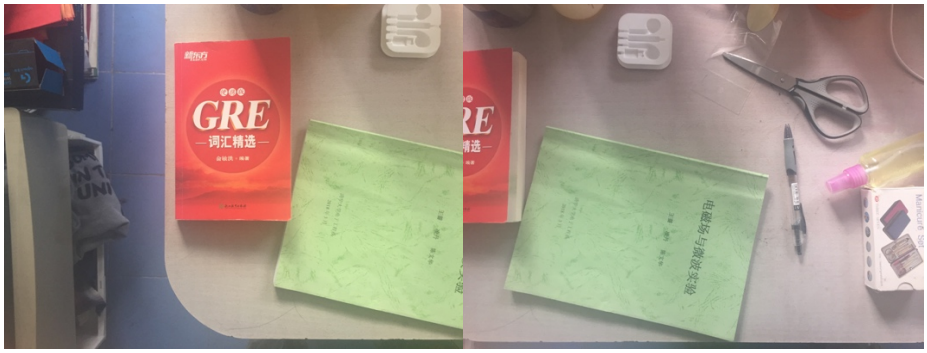


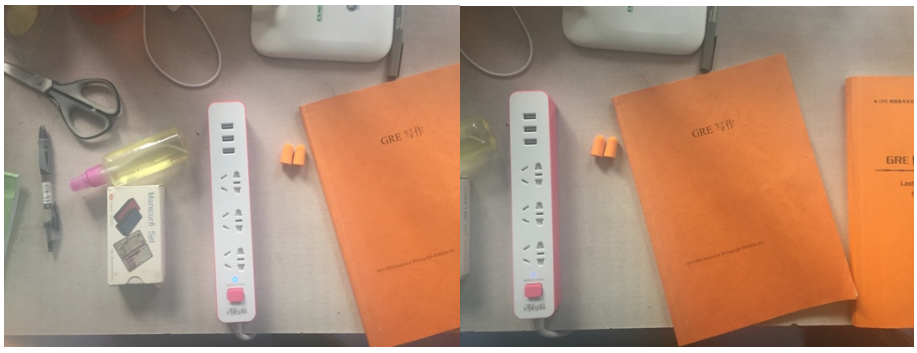
其中绿色实线即两幅图像的分界线。

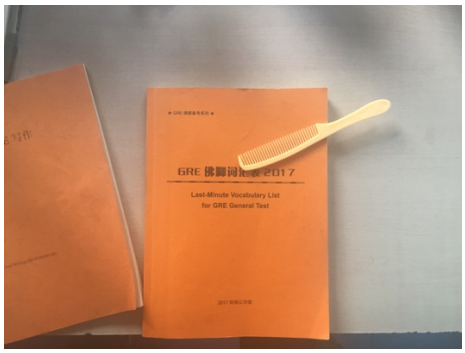
这样加权过后，重叠部分中贴近 img1 的像素更接近 img1，而贴近 img2 的像素更接近 img2，故图像拼合后不会产生明显的分界线。

三、合成图像

1. 原图像



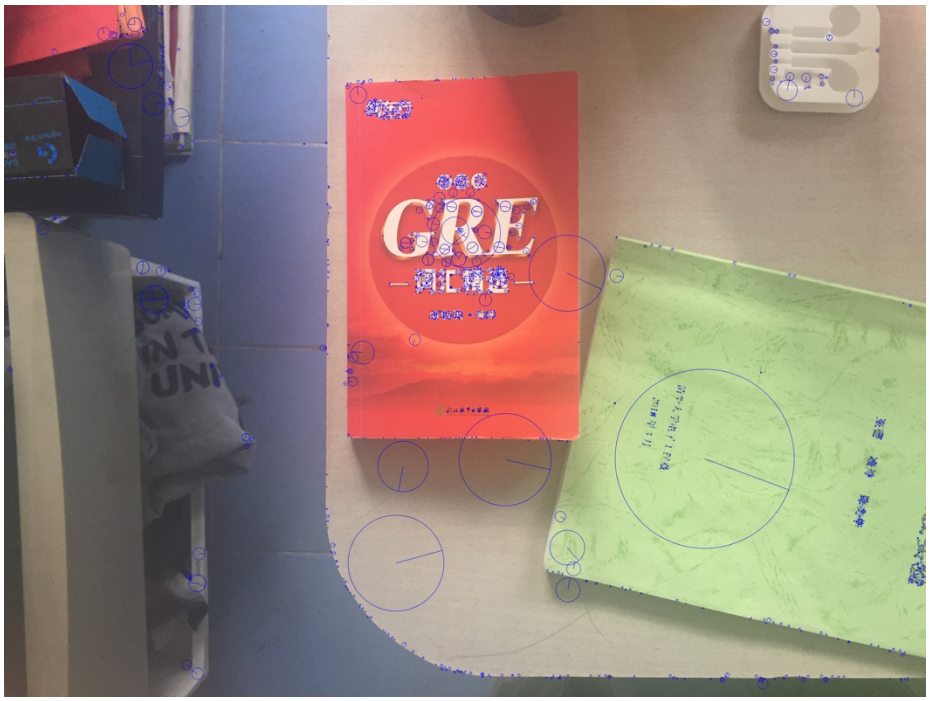




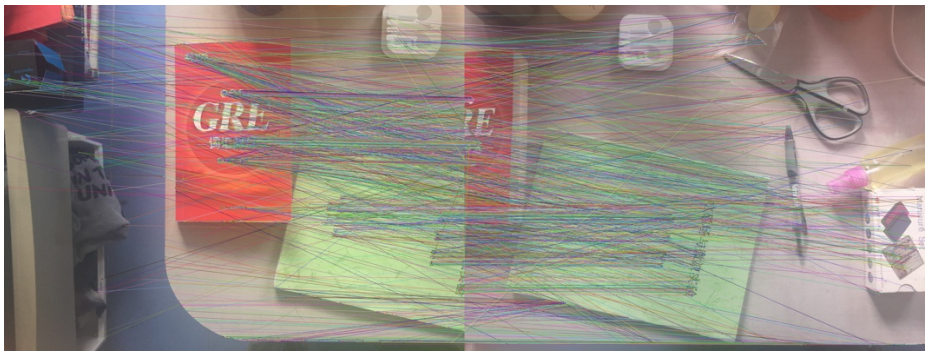
共五张，保存在 Pics 文件夹中。

2. 特征提取及特征匹配结果

1）第一张图片的特征点提取：

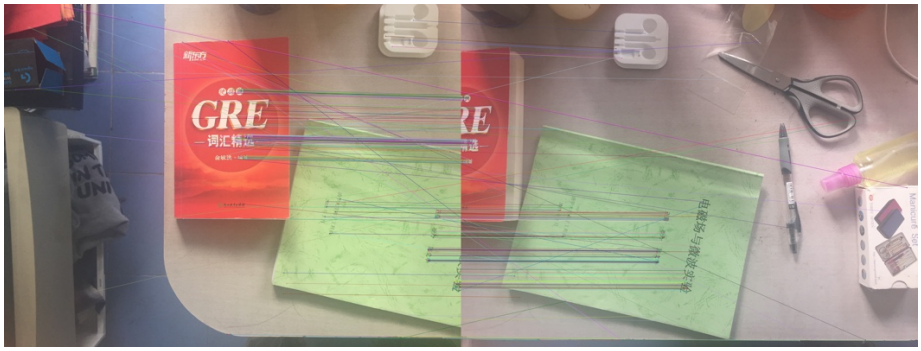


2）第一张图和第二张图的特征点匹配（只考虑 1NN）：



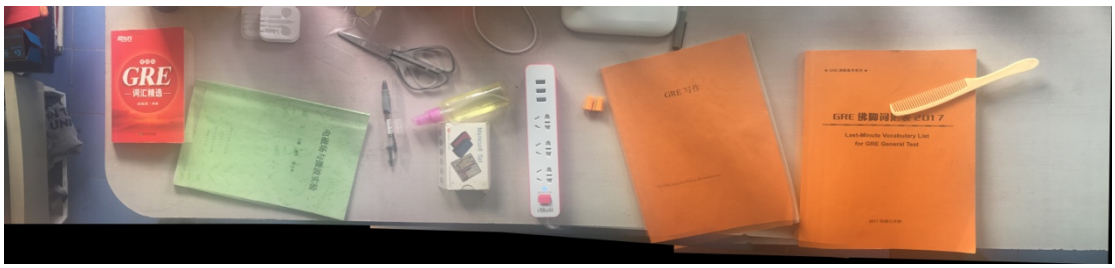
会发现匹配结果很不理想，有很多错配对。

3）第一张图和第二张图的特征点匹配（1NN/2NN<0.8 筛选）：



可以看到筛选后匹配结果清晰了很多。大部分都是正确匹配的特征点对。这样后续的在使用 RANSAC 算法拟合时，迭代次数会大大降低。

3. **合成结果**



可以看到右半部分的合成结果较为理想，基本没有重影。但左半部分的合成结果出现的较多重影，分析原因应该是因为左半部分的物体较多，需要重合的部分太多，而采集的图像又太少，因此出现重影。

合成结果图像保存在 Pics 文件夹中。

四、实验总结

本次实验的代码量比上次实验明显增多，且难度也极大的提高。在网上查阅了许多资料、论文和 OpenCV 官方文档之后才最终完成。虽然最终结果不够完美，但从这个过程中我已经基本熟悉了 numpy 和 python-OpenCV 的操作。收获良多。