

计算机视觉与应用

实践

练习2—Tensorflow试用报告

|  |  |  |  |
| --- | --- | --- | --- |
| **姓 名:** | 林雅惠 | **学 号:** | 123106222874 |
| **学 院:** | 计算机科学与工程学院 | | |

**2024年4月30日**

目录

[一、 实验目标 3](#_Toc25805)

[二、 实验说明 3](#_Toc3130)

[2.1实验原理及步骤 3](#_Toc11651)

[2.2实验方法-1 3](#_Toc8429)

[2.2.1特征提取 4](#_Toc10003)

[2.2.2特征匹配 4](#_Toc227)

[2.2.3变换估计 5](#_Toc18012)

[2.2.4图像拼接 6](#_Toc30246)

[2.3实验方法-2 6](#_Toc15197)

[三、 实验结果 6](#_Toc11879)

[四、 实验总结和分析 8](#_Toc3048)

# 实验目标

1. 理解关键点检测算法DOG原理。
2. 理解尺度变化不变特征SIFT。
3. 采集一系列局部图像，自行设计拼接算法。
4. 使用Python实现图像拼接算法。

# 实验说明

## 2.1实验原理及步骤

图像拼接是将多个部分图像合并成一个完整图像的过程，通常用于创建全景图像、广告拼接等。其原理主要包括以下几个步骤：

1. 特征提取：对输入的多个图像进行特征提取，常用的方法包括SIFT（尺度不变特征变换）或SURF（加速稳健特征等）。
2. 特征匹配：将不同图像中相似的特征点进行匹配，以找到它们之间的对应关系。这一步通常会使用一些匹配算法，如基于距离的最近邻匹配。
3. 变换估计：通过特征点的匹配关系，估计出不同图像之间的变换关系，例如平移、旋转、缩放和透视变换。通常是通过计算单应性矩阵来实现。这个矩阵描述了一个图像到另外一个图像的投影变换。
4. 图像拼接：根据估计的几何变换关系，将不同图像进行对齐和融合，以生成拼接后的图像。通常涉及像素值的加权平均或像素级融合技术，以确保拼接后的图像在边缘处平滑过渡。
5. 后处理：对拼接后的图像进行优化处理，包括边缘平滑，色彩校正，去除拼接接缝等，以提高最终拼接质量和视觉效果。这一步骤可以进一步改善拼接结果的外观和质量。

## 2.2实验方法-1

本实验方法中，特征提取阶段，运用的是SIFT算法，对图像分别检测特征点，并计算每个特征点的描述子。特征匹配阶段，运用的是FLANN匹配算法，对图像的特征描述子进行匹配。变换估计阶段，采用RANSAC算法估计两幅图像之间的单应性矩阵，再根据估计的单应性矩阵，对其中一幅图像进行透视变换，使得两幅图像能够对齐。图像拼接阶段，将经过透视变换的图像与另一幅图像进行拼接，创建空白画布，并找到两幅图像重叠区域的最左和最右边界，，将两幅图像的像素值进行加权平均融合。

### 2.2.1特征提取

在图像特征提取之前，为保证拼接的待拼接图像的尺度一致，先读取两张待拼接图像，获取其中一张待拼接图像的长度和宽度，对另外一张待拼接的图像进行resize处理，统一两张待拼接图像的大小。

实验中特征提取阶段采用SIFT算法对图像的特征进行提取，该算法具有尺度不变性和旋转不变性等特点，算法的基本步骤如下：

1. 尺度空间极值检测：首先，使用高斯函数构建图像的高斯金字塔（通过不断降采样原始图像并应用高斯滤波器），然后在不同尺度上检测图像中的极值点（局部最大值和最小值），这些极值点可能对应于关键点.
2. 关键点定位：在尺度空间极值点的基础上，通过对高斯金字塔中的像素进行插值，精确定位关键点的位置，以及通过Hessian矩阵的特征值来判断关键点的稳定性。这些关键点通常被选择为局部区域的极值点，并且通过比较周围像素的梯度方向来确定它们的方向。
3. 方向分配：对于每个关键点，为其分配一个主导方向，通常选择该点周围像素的梯度方向直方图中的主要峰值作为主导方向，以实现旋转不变性。
4. 关键点描述：在关键点周围的邻域内，根据其主导方向构建描述子。描述子是一个向量，用于描述关键点周围的图像局部特征，通常基于梯度方向直方图的统计信息。
5. 关键点匹配：最后，通过比较两幅图像中的关键点描述子，使用诸如最近邻或最近邻筛选等方法来进行关键点匹配，以识别两幅图像中相似的区域。

由于该算法具有尺度不变性的特征，因此对待拼接的图像统一大小再进行特征提取时，对特征的提取不会有影响。在本实验中运用到该算法的前四个步骤，调用cv.SIFT\_create()函数创建SIFT特征点检测，接着调用sift.detectAndCompute()函数检测兴趣点并计算描述子。

### 2.2.2特征匹配

在特征提取阶段中使用到的SIFT算法包括关键点匹配阶段，但是在FLANN匹配算法相较于SIFT算法的关键点匹配更快速和高效，同时具有更低的内存消耗，所以本实验中特征匹配阶段采用OpenCV中的FLANN匹配算法进行特征匹配。FLANN算法的核心思想是通过构建索引结构来加速最近邻搜索，基于以上特征提取阶段获取的特征描述子，先利用FLANN算法的Index类构建索引结构，再对应每个查询点，通过调用索引结构的查询方法，找到与其最近邻和次近邻匹配的结果，最后再过滤匹配点。实验中采用基于距离阈值选择优质匹配点对，如果最近邻的距离小于距离阈值倍的次近邻的距离，则认为该匹配点为优质匹配点。

本实验中先定义 FLANN 匹配器的索引参数和搜索参数，再调用cv.FlannBasedMatcher()函数创建 FLANN 匹配器对象，接着调用flann.knnMatch()函数进行特征匹配，并返回最近邻和次近邻匹配的结果。最后，基于距离阈值选择优质匹配对，如果最近邻m的距离小于距离阈值倍的次近邻n的距离，则认为这个匹配点对是优质的，本实验中设置的距离阈值为0.3。

### 2.2.3变换估计

该实验中采用RANSAC算法计算出两幅图像之间的单应性矩阵，每次都随机选取4个特征点对，并根据计算出的单应性矩阵对两幅图像进行透视变换。

RANSAC算法的基本思想是随机选择一组数据点来拟合模型，然后使用该模型来计算所有数据点与模型的拟合误差。如果某个数据点与模型的误差小于一个阈值，则将其视为内点，否则视为外点。算法不断重复这个过程，直到找到满足一定置信度的最优模型。该算法的基本步骤如下：

1. 从数据集中随机选择一组数据点，根据这些数据点拟合模型。
2. 对于所有的数据点，计算其与模型的误差，并将误差小于一个阈值的数据点视为内点，否则视为外点。
3. 如果内点的数量大于指定的阈值，则使用所有内点重新估计模型，并计算内点的误差，如果内点的数量小于指定的阈值，则返回步骤1。
4. 如果当前模型的内点数量大于之前的模型，则将当前模型作为最优模型，并更新内点的阈值和置信度。
5. 重复步骤1到步骤4，直到达到指定的迭代次数或置信度。

实验中通过调cv.findHomography()函数来估计两幅图像之前的单应性矩阵，并指定阈值为2，接着调用cv.warpPerspective()函数对图像进行透视变换。

### 2.2.4图像拼接

在图像拼接的过程中，首先为变换估计后的图像img2创建副本，保留的原始的变换估计后的图像。接着将待拼接图像img1图像覆盖到图像mg2的坐上角，为在拼接过程中将图像img1和图像img2对齐并放置在正确的位置。紧接着寻找两张图像重叠的最左边界和最右边界，这两个边界确定了重叠区域的水平范围，使得在后续的融合过程中，只需要处理这个范围内的像素，不需要考虑整个图像，提高了效率并确保了拼接效果的准确性。采用循环遍历列来确定重叠区域的编边界。创建一个新的空白图像res，大小与图像img1保持一致。在图像融合阶段使用计算重叠部分之间的加权平均值作为最终的结果。采用三重循环遍历空白图像res的每个像素，如果该像素在img1中不包含任何值，则将该像素从图像img2中复制到图像res中，如果该像素在图像img2中不不包含任何值，则将该像素从img1中复制到图像res中，如果该像素同时在图像jmg1和图像img2中都包含值，则计算两者之间加权平均值作为最终结果，将该结果像素保存在图像res中。

## 2.3实验方法-2

在本实验中基于图像拼接的原理采用OpenCV库中的cv2.Stitcher\_create()函数创建了一个Stritcher对象，其中可以选择参数，设置拼接模式，设置参数为cv2.STITCHER\_PANORAMA时则为全景拼接模式，设置参数为cv2.STITCHER\_SCANS时则为扫描线拼接模式并使用stritcher.stitch(img1,img2)函数将两张图像进行拼接。

# 实验结果

本实验中实验方法1在特征提取和特征匹配阶段结束后，实验结果如图4-1所示。

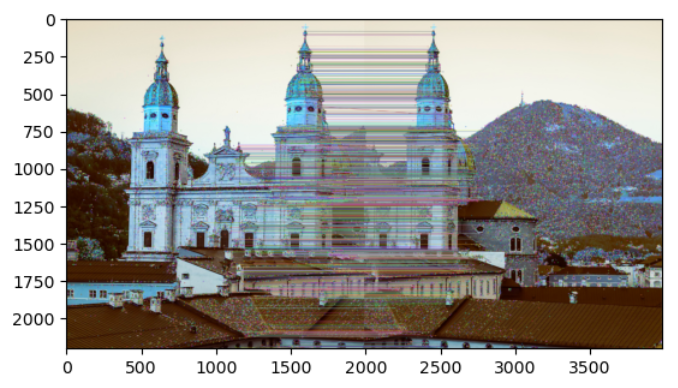


图4-1

在变换估计和图像拼接阶段结束后，实验结果如图4-2所示。

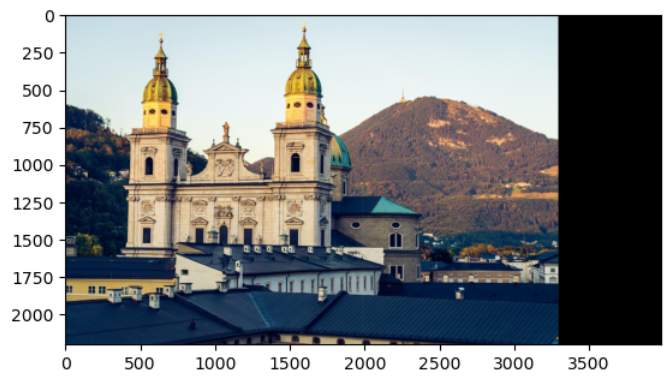


图4-2

本实验中实验方法2图像拼接的拼接结果如图4-3所示。

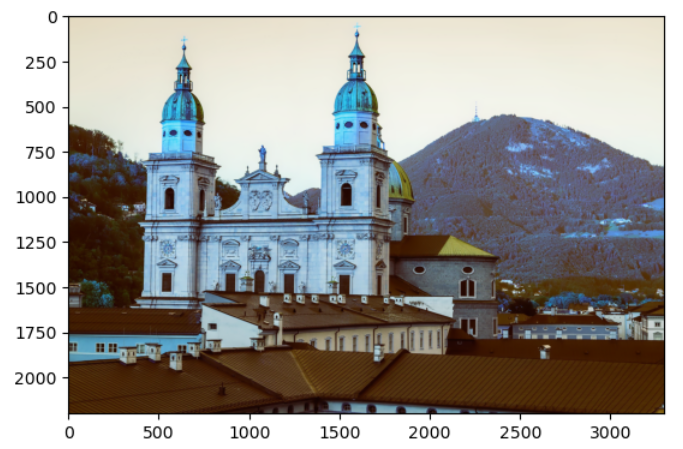


图4-3

# 实验总结和分析

在本实验的整个过程中，特征提取和特征匹配是关键步骤。本实验中采用的SIFT算法对图像特征点的提取很全面，但采用的FLANN匹配算法针对于相似度过高的点匹配的效果不好。针对于本实验，距离阈值设置为0.65时，特征匹配结果如图5-1所示，将实际不匹配的两个特征点匹配上。距离阈值为0.45时，特征匹配结果如图5-2所示，此时匹配准确率较阈值0.65时有所提升，误匹配的点减少。距离阈值设置为0.3时，特征匹配结果如图5-3所示，此时相对于阈值0.45的情况下，特征点匹配的效果更佳。因此针对于相似度过高的点，通过减小阈值大小，可以改变匹配点对数量，阈值越小时，匹配数量越少，匹配的准确率也会上升。

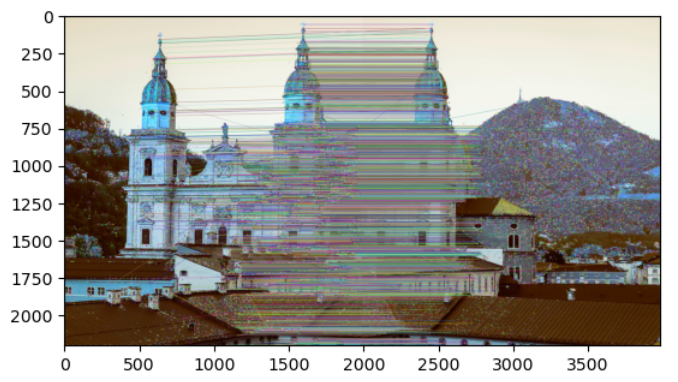


图5-1

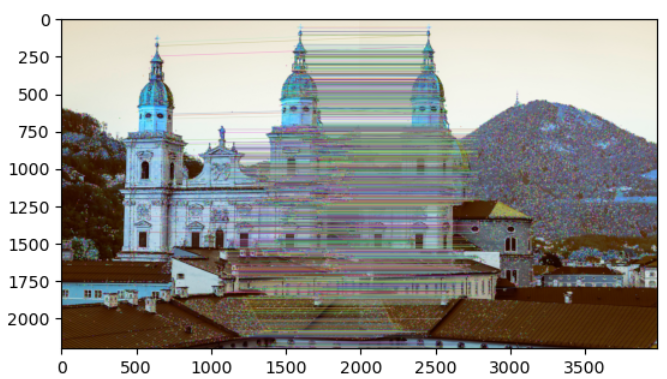


图5-2

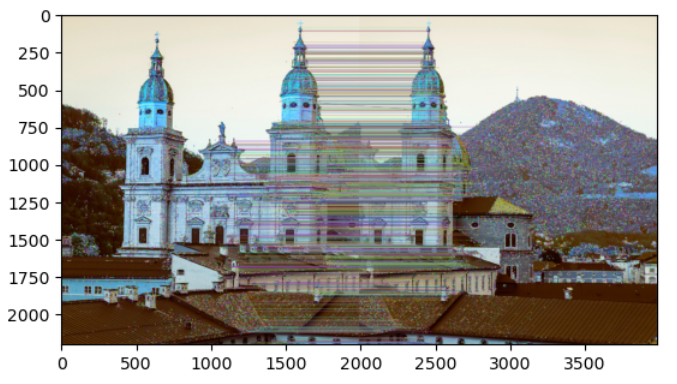


图5-3