实验报告

16340282 袁之浩

实验目标

将多张图片拼接成一张全景图。

实验步骤

1. SIFT 算法检测特征点

尺度不变特征转换（Scale-invariant feature transform）是在不同的尺度空间上查找特征点，并计算出方向，实质上就是用不同尺度（标准差）的高斯函数对图像进行平滑，然后比较图像的差别。

第一步，建立尺度空间。一个图像的尺度空间定义为原始图像 I(x, y)与一个可变尺度的二维高斯函数 G(x, y, σ)进行卷积。σ 是尺度坐标，σ 越大分辨率越低，对应粗糙尺度。

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

为了更有效的在尺度空间检测到稳定的特征点，利用不同尺度的高斯函数与图像卷积，得到一组尺度空间。然后对第一幅图像进行二分之一采样，用同样的方法创建下一组尺度空间，最后就形成了一个图像金字塔。将组内的相邻两幅图像相减，就得到了高斯差分尺度空间（DoG scale-space）。高斯拉普拉斯算子 LoG(Laplacian of Gaussian)，即图像的二阶导数，能够在不同的尺度下检测到图像的斑点特征，从而检测到图像中尺度变化下的位置不动点，但是 LoG 的运算效率不高。而 DoG 是 LoG 的近似。

第二步，检测特征点。将每一个采样点和它同尺度的 8 个相邻点和上下相邻尺度对应的

9*2 个点共 26 个点比较，以确保在尺度空间和二维图像空间都检测到极值点。当然这样产生的极值点并不都是稳定的特征点，因为某些极值点响应较弱，而且 DOG 算子会产生较强的边缘响应。由于图像中的物体的边缘位置的点的主曲率一般会比较高，因此我们可以通过主曲率来判断该点是否在物体的边缘位置。

第三步，计算特征点方向。计算特征点的邻域区域内所有像素的梯度幅角和梯度幅值。这里邻域区域定义为在该图像中以特征点为圆心，以 3*1.5σ 为半径的圆形区域。

$$m(x,y)=\sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2}$$

$$\theta(x,y)=arctan\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right)$$

还需要对邻域范围内的像素点进行加权，然后建立直方图来统计邻域内各个像素点的幅角。直方图分为 36 个柱，每个柱表示 10 度。把直方图建立好后，为了防止噪声的干扰，需要对直方图进行平滑。

第四步，生成特征点描述符。把特征点的邻域区域划分为 4*4 个正方形区域，每个正方形的边长为 3σ，为了保证特征点的方向不变性，还要对特征点及邻域进行旋转。

$$\begin{bmatrix}x'\\y'\end{bmatrix}=\begin{bmatrix}cos\theta & -sin\theta\\sin\theta & cos\theta\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix}$$

再次计算梯度幅值和梯度幅角，用于生成直方图，最终形成的是一个 128 柱的直方图（d*d*8），这个直方图的结果就是特征描述符。为了去除光照的影响，进行长度归一化。

$$q_i=\frac{p_i}{\sqrt{p_1^2+p_2^2+\ldots+p_{128}^2}},i=1,2,3,\ldots,128$$

2. RANSAC 算法特征点匹配

由上面的步骤得到的只是一张图片的特征点，先采用欧式距离最短的方法计算 128 维

特征描述符来大致匹配特征点，再使用 RANSAC 算法排除 outliers。

(1) 从样本集中随机抽选一个 RANSAC 样本，即 4 个匹配点对

(2) 根据这 4 个匹配点对计算变换矩阵 M

(3) 根据样本集，变换矩阵 M，和误差度量函数计算满足当前变换矩阵的一致集 consensus，并返回一致集中元素个数

(4) 根据当前一致集中元素个数判断是否最优(最大)一致集，若是则更新当前最优一致集

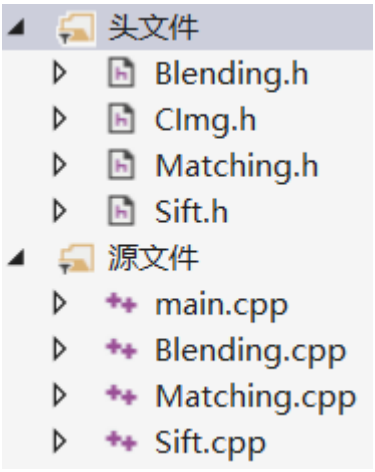
(5) 更新当前错误概率 p，若 p 大于允许的最小错误概率则重复(1)至(4)继续迭代，直到当前错误概率 p 小于最小错误概率

3. 利用关键点进行图像拼接

首先将 img2 经过变换矩阵 H 变换到一个新图像中，然后将图像融合的过程分为三部分，最左边完全取自 img1 的数据，中间的重合部分使用泊松融合，最右边的部分取自 img2 经变换后的图像。

泊松融合的思想是在目标图像的边缘不变的情况下，求出融合部分的图像，使得融合部分的梯度与源图像在融合部分的梯度最为接近。具体步骤：首先计算目标图像和原图像的梯度场，然后计算融合图像的梯度场，再对融合图像的梯度求偏导，得到散度，然后构建泊松方程，就可以解出每个像素的 RGB 值。

代码结构



Sift.h Sift.cpp 用于提取特征点

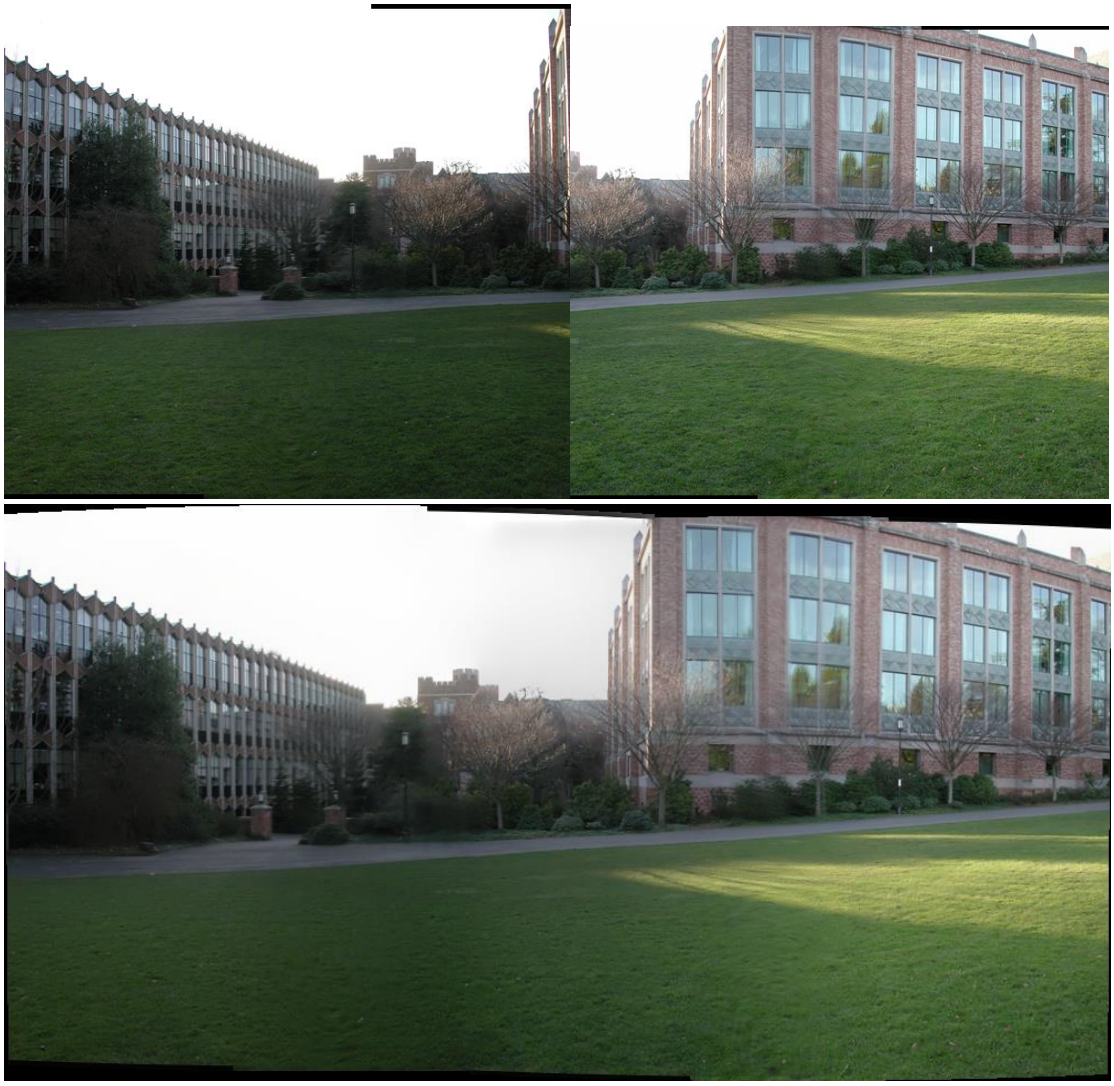
Match.h Math.cpp 用于特征点匹配

Blend.h Blend.cpp 用于图像融合

实验结果

测试集 1

先两两合成，再合成最终图片，中间结果再 Output/1-1 和 Output/1-2 文件夹中



自己拍摄的照片在 myData 文件夹中 ，拍摄的地方是南实验楼，拍摄了 4 张照片，将前两张照片合成为一张，后两者照片合成为一张，最后将两张中间图片合成为一张。