作业 2：图像配准和拼接

学号：19049100002　　姓名：张泽群　　任课老师：李宇楠　　课程号：CS205201

实现步骤

（1）基于图像特征点提取的相关内容，选择相关特征点匹配算法（SIFT、SURF、HOG）完成特征点提取

（2）利用 RANSAC 算法实现特征点匹配

（3）利用图像几何变换算法实现图像变换和拼接

1. 特征点提取(*SURF*)

1.1 简介

Speeded Up Robust Features（*SURF*，加速稳健特征）是一种稳健的局部特征点检测和描述算法。最初由*Herbert Bay*发表在2006年的欧洲计算机视觉国际会议上，并在2008年正式发表在《*Computer Vision and Image Understanding*》期刊上。

*SURF*是对*David Lowe*在1999年提出的*SIFT*算法的改进，提升了算法的执行效率，为算法在实时计算机视觉系统中应用提供了可能。

1.2 原理与计算过程

1. 构建 Hessian 矩阵，生成所有的兴趣点，用于特征的提取

黑塞矩阵（*Hessian Matrix*）是一个多元函数的二阶偏导数构成的方阵，描述了函数的局部曲率。由德国数学家*Ludwin Otto Hessian*于19世纪提出。surf构造的金字塔图像与sift有很大不同，sift采用的是DOG图像，而surf采用的是Hessian矩阵行列式近似值图像。

Hessian矩阵是Surf算法的核心，构建Hessian矩阵的目的是为了生成图像稳定的边缘点（突变点），为下文的特征提取做好基础。

每一个像素点都可以求出一个Hessian矩阵。

$$H(f(x,y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x^2} \end{bmatrix}$$

Hessian矩阵的判别式为：

$$det(H) = \frac{\partial^2}{\partial x^2}\frac{\partial^2 f}{\partial y^2} - (\frac{\partial^2 f}{\partial x \partial y})$$

当Hessian矩阵的判别式取得局部极大值时，判定当前点是比周围邻域内其他点更亮或更暗的点，由此来定位关键点的位置。

在SURF算法中，图像像素l(x，y)即为函数值f(x，y)。但是由于我们的特征点需要具备尺度无关性，所以在进行Hessian矩阵构造前，需要对其进行高斯滤波，选用二阶标准高斯函数作为滤波器。

$$L(x,t) = G(t) \cdot I(x,t)$$

通过特定核间的卷积计算二阶偏导数。通过特定核间的卷积计算二阶偏导数，这样便能计算出H矩阵的三个矩阵元素*L_xx, L_xy, L_yy*从而计算出H矩阵：

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix}$$

由于高斯核是服从正态分布的，从中心点往外，系数越来越低，为了提高运算速度，Surf使用了盒式滤波器来近似替代高斯滤波器，提高运算速度。盒式滤波器（Boxfilter）对图像的滤波转化成计算图像上不同区域间像素和的加减运算问题，只需要简单几次查找积分图就可以完成。

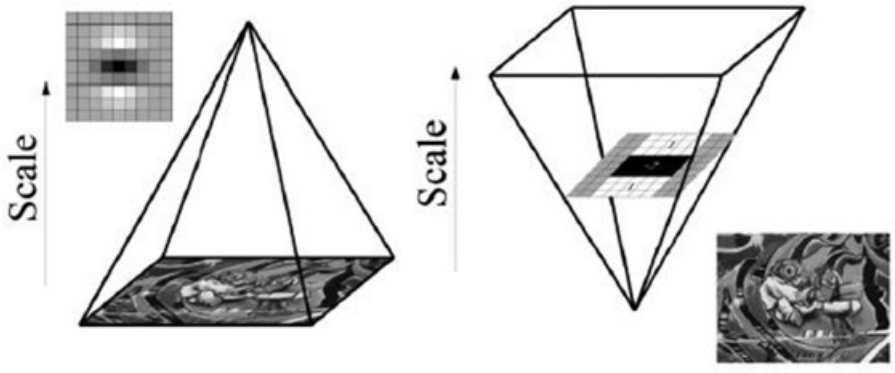
每个像素的Hessian矩阵行列式的近似值：

$$det(H) = Dxx * Dyy - (0.9 * Dxy)^2$$

在*Dxy*上乘了一个加权系数0.9，目的是为了平衡因使用盒式滤波器近似所带来的误差。

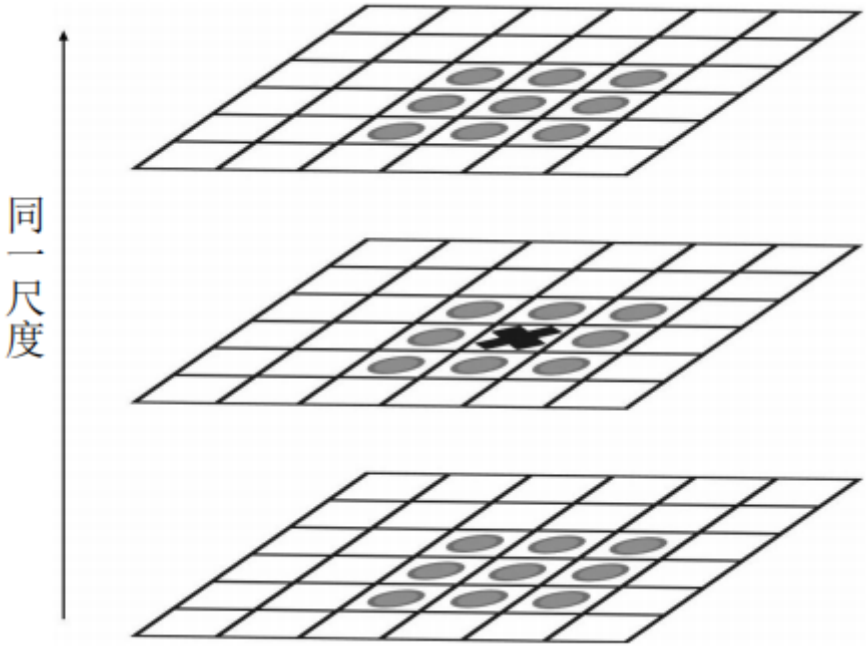
2. 构建尺度空间

同Sift一样，Surf的尺度空间也是由O组L层组成，不同的是，Sift中下一组图像的尺寸是上一组的一半，同一组间图像尺寸一样，但是所使用的高斯模糊系数逐渐增大；而在Surf中，不同组间图像的尺寸都是一致的，但不同组间使用的盒式滤波器的模板尺寸逐渐增大，同一组间不同层间使用相同尺寸的滤波器，但是滤波器的模糊系数逐渐增大。



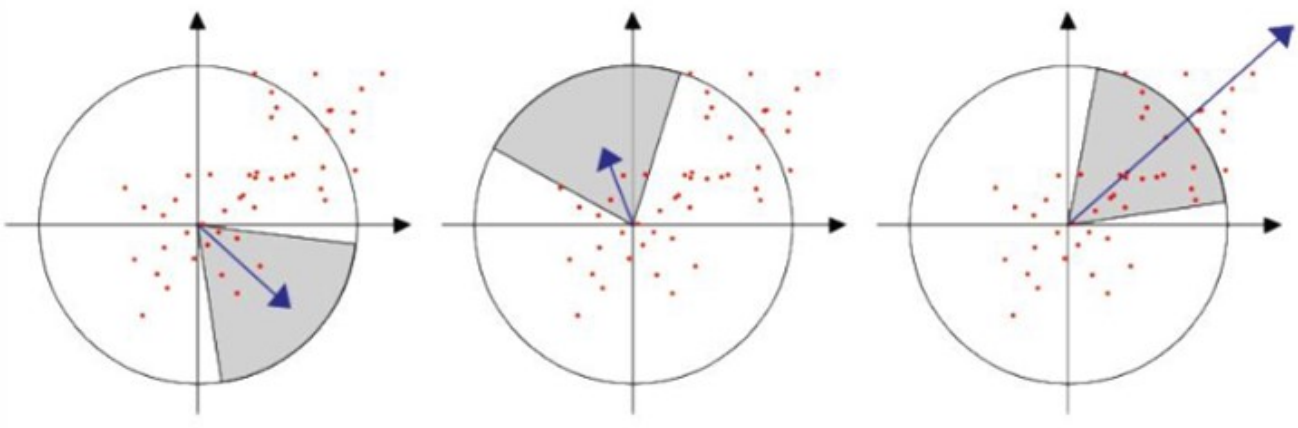
3. 特征点定位

特征点的定位过程Surf和Sift保持一致，将经过Hessian矩阵处理的每个像素点与二维图像空间和尺度空间邻域内的26个点进行比较，初步定位出关键点，再经过滤除能量比较弱的关键点以及错误定位的关键点，筛选出最终的稳定的特征点。



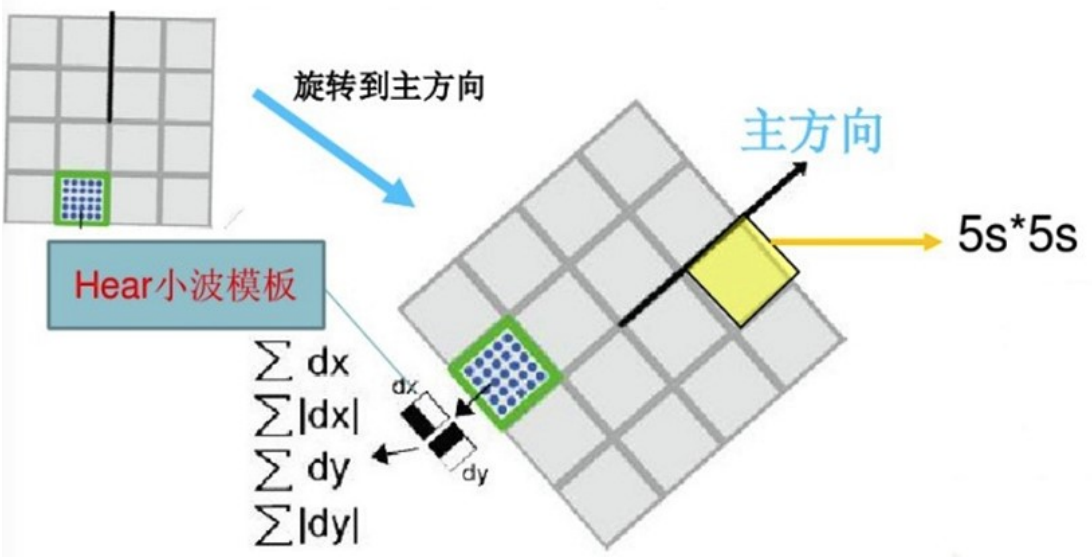
4. 特征点主方向分配

Sift特征点方向分配是采用在特征点邻域内统计其梯度直方图，而在Surf中，采用的是统计特征点圆形邻域内的harr小波特征。
在特征点的圆形邻域内，统计60度扇形内所有点的水平、垂直harr小波特征总和，然后扇形以一定间隔进行旋转并再次统计该区域内harr小波特征值之后，最后将值最大的那个扇形的方向作为该特征点的主方向。



5. 生成特征点描述子

在Sift中，是取特征点周围4×4个区域块，统计每小块内8个梯度方向，用着4×4×8=128维向量作为Sift特征的描述子。
Surf算法中，也是在特征点周围取一个4×4的矩形区域块，但是所取得矩形区域方向是沿着特征点的主方向。每个子区域统计25个像素的水平方向和垂直方向的haar小波特征，这里的水平和垂直方向都是相对主方向而言的。该haar小波特征为水平方向值之后、垂直方向值之后、水平方向绝对值之后以及垂直方向绝对值之和4个方向。



把这4个值作为每个子块区域的特征向量，所以一共有4^3=64维向量作为Surf特征的描述子，比Sift特征的描述子减少了一半。

6. 特征点匹配

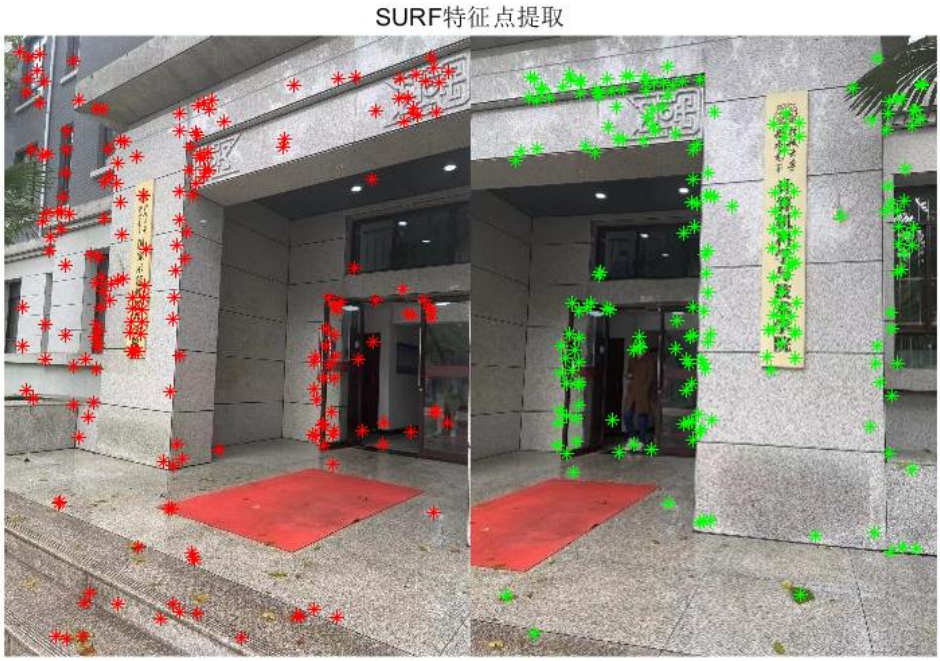
与Sift特征点匹配类似，Surf也是通过计算两个特征点间的欧式距离来确定匹配度，欧氏距离越短，代表两个特征点的匹配度越好。
不同的是Surf还加入了Hessian矩阵迹的判断，如果两个特征点的矩阵迹正负号相同，代表这两个特征具有相同方向上的对比度变化，如果不同，说明这两个特征点的对比度变化方向是相反的，即使欧氏距离为0，也直接予以排除。

1.3 实现以及结果

注：利用Matlab Computer Vision Toolbox的函数detectSURFFeatures()实现特征点提取。

```
% SURF算法读取特征点
points1 = detectSURFFeatures(img1);
points2 = detectSURFFeatures(img2);
% 特征点图像
drawPoint(I1,points1.Location,I2,points2.Location);
```

结果如下所示：



2. *RANSAC* 算法实现特征点匹配

2.1 简介

随机抽样一致算法（RANdom SAmple Consensus,RANSAC）,采用迭代的方式从一组包含离群的被观测数据中估算出数学模型的参数。RANSAC算法假设数据中包含正确数据和异常数据（或称为噪声）。正确数据记为内点（inliers），异常数据记为外点（outliers）。同时RANSAC也假设，给定一组正确的数据，存在可以计算出符合这些数据的模型参数的方法。该算法核心思想就是随机性和假设性，被广泛应用在计算机视觉领域和数学领域。

2.2 原理与计算过程

RANSAC 算法是一种通过随机抽样观察数据来估计模型参数的学习技术。给定一个数据元素同时包含内部值和异常值的数据集，RANSAC 使用投票方案来寻找最佳拟合结果。数据集中的数据元素用于为一个或多个模型投票。该投票方案的实现基于两个假设：嘈杂的特征不会一致地投票给任何单个模型（很少有异常值），并且有足够的特征来就一个好的模型达成一致(很少有缺失的数据)。RANSAC 算法本质上由迭代重复的两个步骤组成：

1. 第一步，从输入数据集中随机选择包含最少数据项的样本子集。仅使用该样本子集的元素计算拟合模型和相应的模型参数。样本子集的基数是最小的，足以确定模型参数。
2. 第二步，算法检查整个数据集的哪些元素与第一步获得的估计模型参数实例化的模型一致。如果某个数据元素在定义噪声影响的最大偏差的某个误差阈值内不符合由一组估计模型参数实例化的拟合模型，则该数据元素将被视为异常值。

为拟合模型获得的内点集称为共识集。RANSAC 算法会反复重复以上两个步骤，直到在某一次迭代中获得的共识集有足够的内点。

RANSAC 算法的输入是一组观测数据值、一种将某种模型拟合到观测值的方法以及一些置信度参数。RANSAC 通过重复以下步骤来实现其目标：

1. 假定模型（如直线方程），并随机抽取N个（以2个为例）样本点，对模型进行拟合
2. 由于不是严格线性，数据点都有一定波动，假设容差范围为：sigma，找出距离拟合曲线容差范围内的点，并统计点的个数。
3. 重新随机选取N个点，重复第一步~第二步的操作，直到结束迭代。
4. 每一次拟合后，容差范围内都有对应的数据点数，找出数据点个数最多的情况，就是最终的拟合结果。

2.3 实现以及结果

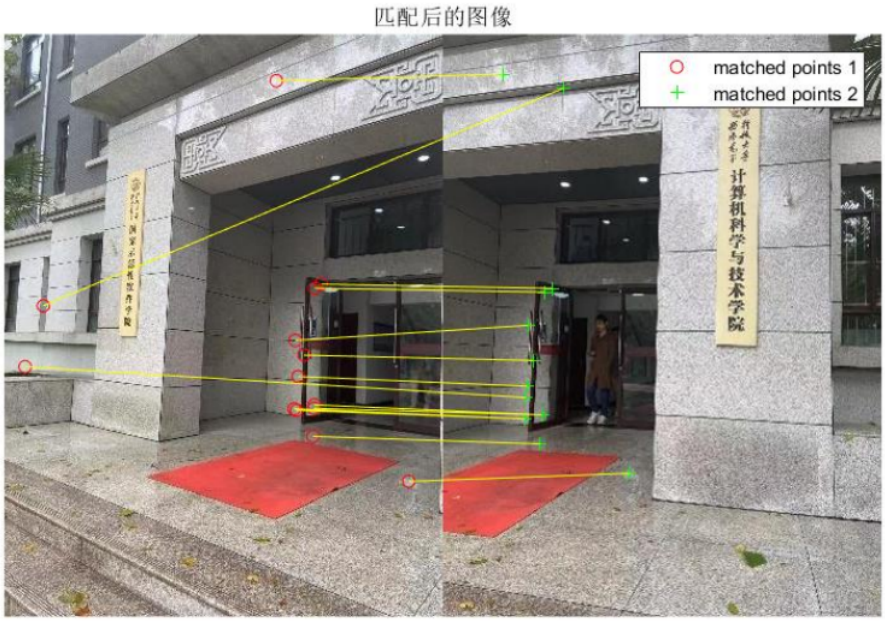
注：利用Matlab Computer Vision Toolbox的函数extractFeatures()，matchFeatures()实现初步的特征点匹配。

```
% 提取特征向量
[f1, vpts1] = extractFeatures(img1, points1,'Method','SURF');
[f2, vpts2] = extractFeatures(img2, points2,'Method','SURF');

% 进行匹配，检索匹配点的位置
indexPairs = matchFeatures(f1, f2, 'Prenormalized', true) ;
matched_pts1 = vpts1(indexPairs(:, 1));
matched_pts2 = vpts2(indexPairs(:, 2));

% 显示匹配,但仍存在离群点的影响
figure('name','匹配后的图像');
showMatchedFeatures(I1,I2,matched_pts1,matched_pts2,'montage');
title('匹配后的图像');
legend('matched points 1','matched points 2');
```

结果如下所示：



注：利用Matlab Computer Vision Toolbox的函数estimateGeometricTransform2D(),该函数使用随机样本一致性（RANSAC）算法去除错误匹配点。

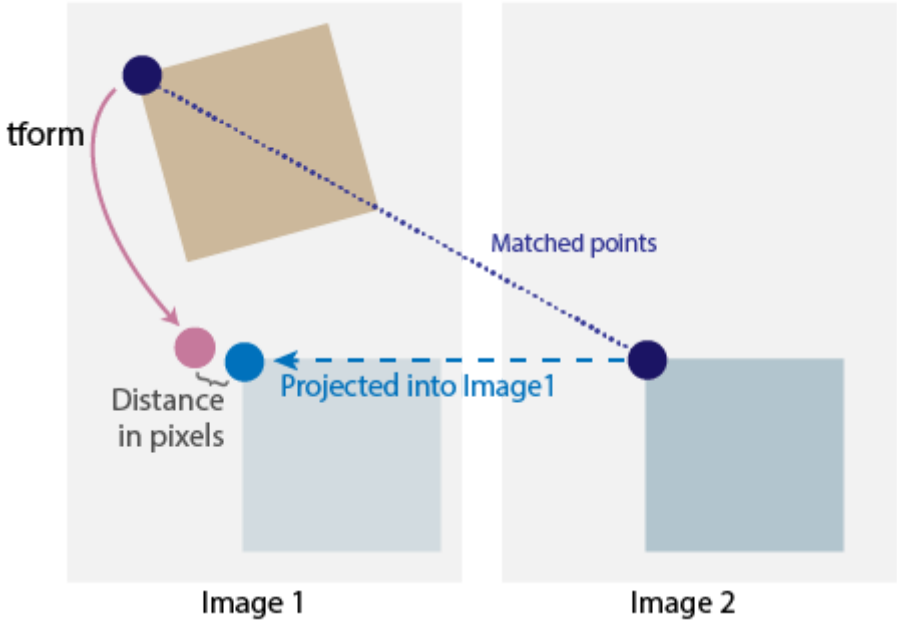
```
% 从匹配点对估计几何变换
% 该函数使用随机样本一致性（RANSAC）算法的变体MSAC算法实现，去除误匹配点
[tform,inlierIdx] =
estimateGeometricTransform2D(matched_pts2,matched_pts1,'similarity');
inlier_pts1 = matched_pts1(inlierIdx,:);
inlier_pts2 = matched_pts2(inlierIdx,:);
figure
showMatchedFeatures(I1,I2,inlier_pts1,inlier_pts2,'montage');
title('RANSAC算法消除离群点影响后的图像')
```

结果如下所示：

3. 图像变换和拼接

3.1 原理与计算过程

即利用之前所得的tform矩阵，对Image 2进行仿射变换，并在相应位置进行拼接。

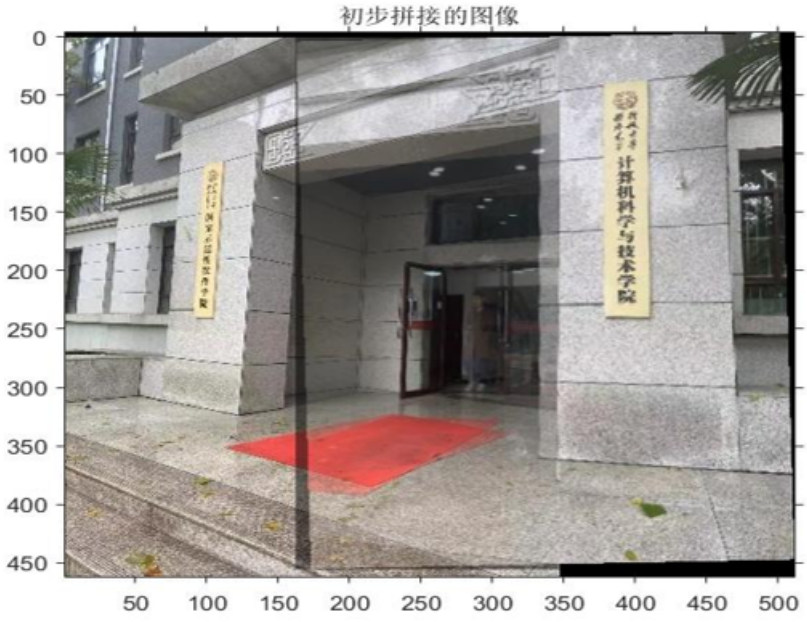


3.2 实现以及结果

注：利用imwarp(),根据此前获得的变换矩阵tform可将两图片初步配准与拼接。

```
% 利用图像几何变换算法实现图像变换和拼接
Rfixed = imref2d(size(I1)); %获取图像的世界坐标
[B, RB] = imwarp(I2, tform);%根据几何变换 tform 来变换I2
figure
imshowpair(I1,Rfixed,B,RB,'blend');
title('初步拼接的图像')
```

结果如下所示：



对重叠区域进行亮度归一化处理即可得到后处理图像：

后处理图像

4. 实验结果及最终分析

4.1 实验结果

实验最终结果如下图所示：

后处理图像

4.2 最终分析

分析：各个步骤(SURF，RANSAC，几何变换)基本实现，图像的配准与拼接功能基本实现。从最终获得的图样来看，图样的配准度还没有做到特别完善，当然可能是因为图像本身的原因以及噪声处理的不到位。

5. 源程序附录

1. Main.m

```
clear all;
clc;

% 读取图像
I1=imread('I1.png');
I2=imread('I2.png');

ISize=size(I1);
I2 = imresize(I2,[ISize(1),ISize(2)]);

% 并排显示两幅待拼接图像
Imgs=[I1,I2];
figure,imshow(Imgs);
title('待拼接图像');

% 转化为灰度图
img1=rgb2gray(I1);
img2=rgb2gray(I2);

% 对图像进行滤波
img1=medfilt2(img1);
img2=medfilt2(img2);

% SURF算法读取特征点
points1 = detectSURFFeatures(img1);
points2 = detectSURFFeatures(img2);
% 特征点图像
drawPoint(I1,points1.Location,I2,points2.Location);

% 提取特征向量
[f1, vpts1] = extractFeatures(img1, points1,'Method','SURF');
[f2, vpts2] = extractFeatures(img2, points2,'Method','SURF');

% 进行匹配，检索匹配点的位置
indexPairs = matchFeatures(f1, f2, 'Prenormalized', true) ;
matched_pts1 = vpts1(indexPairs(:, 1));
matched_pts2 = vpts2(indexPairs(:, 2));
```

```
% 显示匹配,但仍存在离群点的影响
figure('name','匹配后的图像');
showMatchedFeatures(I1,I2,matched_pts1,matched_pts2,'montage');
title('匹配后的图像');
legend('matched points 1','matched points 2');

% 从匹配点对估计几何变换
% 该函数使用随机样本一致性（RANSAC）算法的变体MSAC算法实现，去除误匹配点
[tform,inlierIdx] =
estimateGeometricTransform2D(matched_pts2,matched_pts1,'similarity');
inlier_pts1 = matched_pts1(inlierIdx,:);
inlier_pts2 = matched_pts2(inlierIdx,:);
figure
showMatchedFeatures(I1,I2,inlier_pts1,inlier_pts2,'montage');
title('RANSAC算法消除离群点影响后的图像')

% 利用图像几何变换算法实现图像变换和拼接
Rfixed = imref2d(size(I1)); %获取图像的世界坐标
[B, RB] = imwarp(I2, tform);%根据几何变换 tform 来变换I2
figure
imshowpair(I1,Rfixed,B,RB,'blend');
title('初步拼接的图像')

[xlim, ylim] = outputLimits(tform, [1 ISize(2)], [1 ISize(1)]);
% 找到输出空间限制的最大最小值
xMin = min([1; xlim(:)]);%1
xMax = max([ISize(2); xlim(:)]);
yMin = min([1; ylim(:)]);
yMax = max([ISize(1); ylim(:)]);

% 全景图的宽高
width  = round(xMax - xMin);
height = round(yMax - yMin);

% 创建2D空间参考对象定义全景图尺寸
xLimits = [xMin xMax];
yLimits = [yMin yMax];
% 创建全景图
panoramaView = imref2d([height width], xLimits, yLimits);

% 变换图像I2到全景图
warped_img = imwarp(I2, tform, 'OutputView', panoramaView);

%亮度归一化
mask2 = (warped_img(:,:,1)>0 |warped_img(:,:,2)>0 |
warped_img(:,:,3)>0);% 获取变换图像掩膜
Idone = zeros(size(warped_img));
Idone(1:size(I1,1), 1: size(I1,2),:) = I1; % I1不变
```

```
mask1 = (Idone(:,:,1)>0 | Idone(:,:,2)>0 | Idone(:,:,3)>0); % 获取非变换图像掩膜
mask = and(mask2, mask1);% 重叠区图像掩膜

% 获得重叠区透明度(i2)
[~,col] = find(mask);
left = min(col);
right = max(col);
mask = ones(size(mask));
mask(:,left:right) = repmat(linspace(0,1,right-left+1),size(mask,1),1);%复制平铺矩阵

% 融合RGB通道(I2)
warped_img=double(warped_img);
warped_img(:,:,1) = warped_img(:,:,1).*mask;
warped_img(:,:,2) = warped_img(:,:,2).*mask;
warped_img(:,:,3) = warped_img(:,:,3).*mask;

% 反转透明度值(I1)
mask(:,left:right) = repmat(linspace(1,0,right-left+1),size(mask,1),1);
% 融合RGB通道(I1)
Idone(:,:,1) = Idone(:,:,1).*mask;
Idone(:,:,2) = Idone(:,:,2).*mask;
Idone(:,:,3) = Idone(:,:,3).*mask;

% 相加得出后处理图像
Idone(:,:,1) = warped_img(:,:,1) + Idone(:,:,1);
Idone(:,:,2) = warped_img(:,:,2) + Idone(:,:,2);
Idone(:,:,3) = warped_img(:,:,3) + Idone(:,:,3);
Idone=uint8(Idone);
figure
imshow(Idone);
title('后处理图像');
```

2. drawPoint.m

```
function drawPoint(I1,pos1,I2,pos2)
diff=size(I1,2);
figure('name','SURF特征点提取')
imshowpair(I1,I2,'montage');
hold on
title('SURF特征点提取');
plot(pos1(:,1),pos1(:,2),'r*');
plot(pos2(:,1)+diff,pos2(:,2),'g*');
hold off
```