# 图像之间单应性变化

## 实验目的：

通过计算两幅图像之间的单应性矩阵，实现图像之间的单应性变化，展示图像在几何上的转换效果。

## 实验步骤：

1. 读取两幅输入图像：image1.jpg 和 image2.jpg。
2. 使用SIFT算法提取并匹配图像的特征点。
3. 根据特征点的对应关系，利用RANSAC算法估计两幅图像之间的单应性矩阵。
4. 利用计算得到的单应性矩阵，对image1进行透视变换，将其与image2拼接在一起形成新的图像。
5. 展示实验结果，并分析图像之间的单应性变化效果。

## 实验代码：

import cv2  
import numpy as np  
  
# 读取两幅图像  
img1 = cv2.imread('1.png', cv2.IMREAD\_GRAYSCALE)  
img2 = cv2.imread('2.png', cv2.IMREAD\_GRAYSCALE)  
  
# 使用SIFT算法提取并匹配特征点  
sift = cv2.SIFT\_create()  
keypoints1, descriptors1 = sift.detectAndCompute(img1, None)  
keypoints2, descriptors2 = sift.detectAndCompute(img2, None)  
  
bf = cv2.BFMatcher(cv2.NORM\_L2, crossCheck=True)  
matches = bf.match(descriptors1, descriptors2)  
matches = sorted(matches, key = lambda x:x.distance)  
  
# 提取匹配的特征点  
src\_pts = np.float32([keypoints1[m.queryIdx].pt for m in matches]).reshape(-1, 1, 2)  
dst\_pts = np.float32([keypoints2[m.trainIdx].pt for m in matches]).reshape(-1, 1, 2)  
  
# 计算单应性矩阵  
H, \_ = cv2.findHomography(src\_pts, dst\_pts, cv2.RANSAC, 5.0)  
  
# 将img2图像进行透视变换  
warped\_img2 = cv2.warpPerspective(img2, H, (img1.shape[1] + img2.shape[1], img1.shape[0]))  
  
# 拼接图像  
result = np.zeros((max(img1.shape[0], warped\_img2.shape[0]), img1.shape[1] + warped\_img2.shape[1]), dtype=np.uint8)  
result[:img1.shape[0], :img1.shape[1]] = img1  
result[:warped\_img2.shape[0], img1.shape[1]:] = warped\_img2  
  
# 显示结果  
cv2.imshow('Result', result)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

## 实验结果：

实验中我们成功计算了两幅图像之间的单应性矩阵，并利用该矩阵实现了图像之间的几何变换。通过实验，我们展示了图像在几何上的转换效果，呈现出图像之间的单应性变化。实验结果如下所示（插入展示结果的图片）：





## 结论与分析：

通过本实验，我们深入了解了图像之间的单应性变化原理和实现方法。单应性变换可用于图像配准、图像融合等领域，具有重要的应用价值。在实际应用中，我们可以根据需求选择合适的特征提取算法和匹配算法，进一步优化单应性变换的效果。