Visual Recognition Assignment 2 (Part 1)

Mohd. Rizwan Shaikh
IMT2019513

March 3, 2022

Abstract

The aim of this assignment is to perform image stitching using Python and OpenCV. Given a pair of images that share some common region, our goal is to "stitch" them and create a panoramic image scene.

Contents

1 Introduction

In this assignment, we perform image stitching to "stitch" two images together and create a simple panorama. To construct our image panorama, we utilize computer vision and image processing techniques such as keypoint detection and local invariant descriptors, feature matching, RANSAC and perspective warping.

2 Image Data

In order to validate the image stitching method, we test it on three different pairs of images. These are shown below:

Figure 1: First pair

Figure 2: Second pair

Figure 3: Third pair

In all of the images above, we notice that some part should be common to both the input images in order to stitch them successfully.

3 Method

The image stitching method can be broadly divided into four parts. These parts are described in detail below:

3.1 Detect keypoints and extract local invariant descriptors

First of all, we extract the keypoints and descriptors from both the input images. We do it by using OpenCV's *detectAndCompute()* function. We use an instance of SIFT detector for the same. It returns a 128-dimensional feature vector for each keypoint. Before feeding the image to *detectAndCompute()*, we convert the images to grayscale.

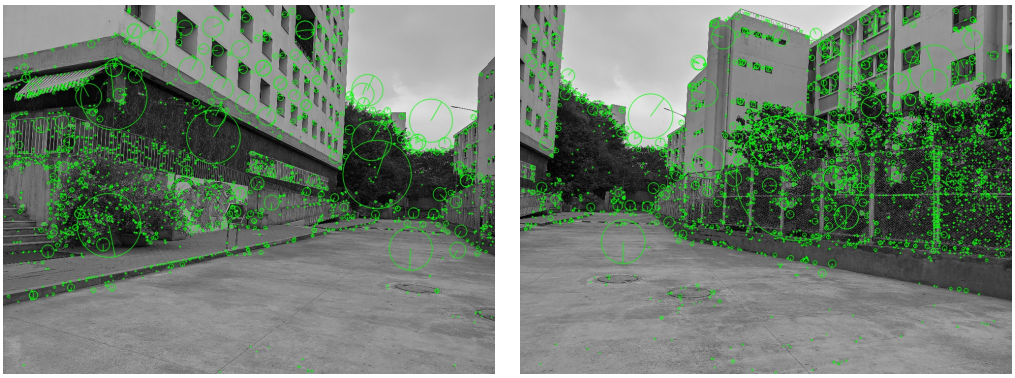

Figure 4: Detection of key points and descriptors using SIFT in first pair of images

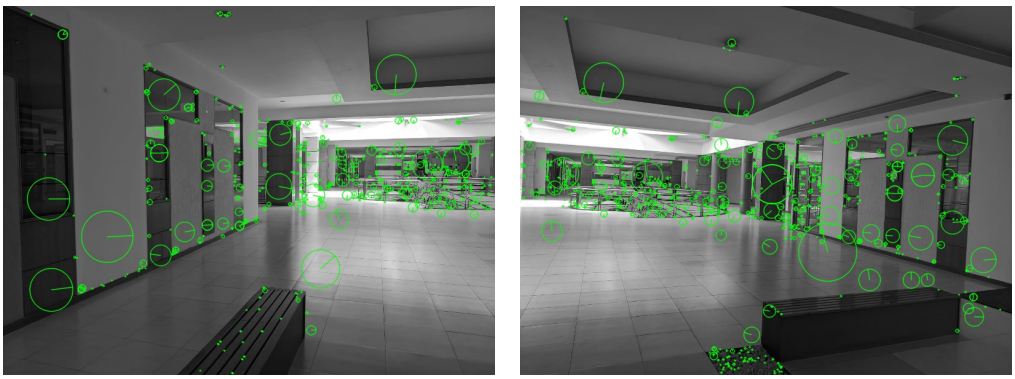

Figure 5: Detection of key points and descriptors using SIFT in second pair of images

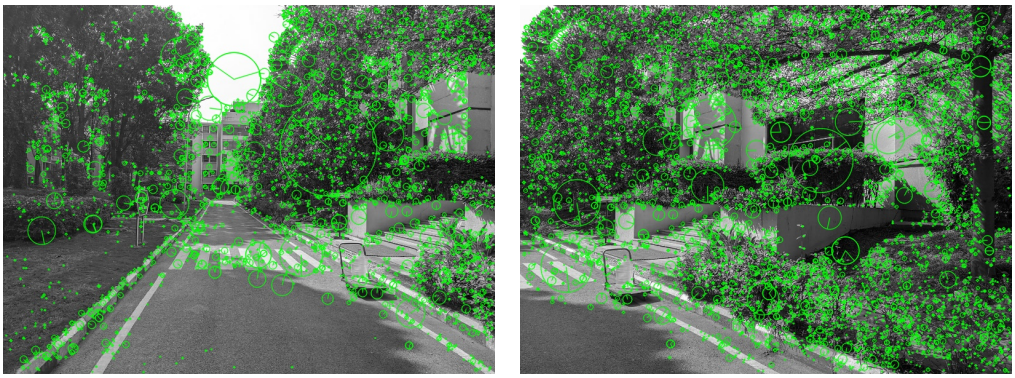

Figure 6: Detection of key points and descriptors using SIFT in third pair of images

3.2 Match the descriptors between the two images

After getting the features, we use kNN matching between the two feature vectors using k = 2 (indicating the top two matches for each feature vector are returned). We choose top two matches to apply David Lowe's ratio test for false-positive match pruning. This is done to ensure that the features returned by KNN are well comparable.

The results of KNN matcher on SIFT features are shown below:

Figure 7: Feature matching using KNN and Ratio Testing on SIFT features

Figure 8: Feature matching using KNN and Ratio Testing on SIFT features

Figure 9: Feature matching using KNN and Ratio Testing on SIFT features

Note that even after applying ratio testing in KNN, some of the features do not match properly. Nevertheless, the Matcher algorithm will give us the best(more similar) set of features from both images. Now, we take these points and find the transformation matrix that will stitch the 2 images together based on their matching points. Such a transformation is called the Homography matrix.

3.3 Estimate Homography using RANSAC

RANdom SAmple Consensus or RANSAC is an iterative algorithm to fit linear models. Different from other linear regressors, RANSAC is designed to be robust to outliers.

Here, we use RANSAC to estimate the Homography matrix. Once we obtain Homography matrix, we warp one of the images to a common plane.

3.4 Apply warping transformation using the Homography matrix

We apply perspective transformation to one of the images. The transformation combines one or more operations like rotation, scale, translation, or shear to transform one of the images so that both the images merge as one. To do this, we use OpenCV's *warpPerspectiv()* function. It takes an image and the homography as the input. Then, it warps the source image to the destination based on the homography.

The result of the operation is shown below:

4 Results

The panoramas created from the three pairs of images are shown below:

Figure 10: Panorama 1

Figure 11: Panorama 2

Figure 12: Panorama 3

5 How is SURF is different from SIFT?

There are four fundamental steps in the SIFT algorith: scale-space extrema detection, keypoint localization, orientation assignment and keypoint descriptor. The first stage is to use difference of Gaussian function to identify potential interest points, which are invariant to scale and orientation. In the keypoint localization step, low contrast points are rejected and edge response is eliminated. After this step, only strong interest points remains. In orientation assignment, an orientation is assigned to each keypoint to achieve invariance to image rotation. The final output is keypoints with same location and scale, but different directions. In the last step, keypoint descriptors are created.

SURF uses box filters to approximate the difference of Gaussian. Instead of using Gaussian averaging, squares are used since convolution with square is considerably faster than using the integral image. To find the points of interest, the SURF uses a BLOB detector which is based on the Hessian matrix. It uses wavelet responses in both horizontal and vertical directions, with appropriate Gaussian weights, for orientation assignment. The wavelet responses are also used for feature description. A neighborhood around the key point is selected and divided into subregions and then for each subregion the wavelet responses are taken and represented to get SURF feature descriptor.

6 Main Principles of FLANN Matching and RANSAC

FLANN (Fast Library for Approximate Nearest Neighbors) is an image matching algorithm for fast approximate nearest neighbor searches in high dimensional spaces. These methods produce compact binary codes by projecting high-dimensional characteristics to a lower-dimensional space. Benefiting from the produced binary codes, fast image search can be carried out using binary pattern matching or Hamming distance measurement, which drastically decreases the computing cost and improves the search efficiency.

RANSAC stands for Random Sample Consensus. We utilise RANSAC parameter estimation to fit a model to data with outliers so that the model does not account for the outliers. RANSAC uses repeated random sub-sampling for this purpose. A model that has been trained on both inliers and outliers is not likely to be correct. RANSAC divides the data into sub-samples and trains a model on each of them. The final model is the one that achieves the greatest results within its subset of data. The best results will come from a model trained on a random sample that is almost completely made up of inliers. However, the likelihood of picking a random sample nearly completely made up of inliers may be influenced by the proportions of inliers in the data as well as the algorithm parameters chosen.