Project 1: MyAutoPano

Dhrumil Kotadia
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, Massachusetts

Problem Statement

The purpose of the project is to stitch two or more images to create a panorama image. The images should have repeated local features which can be used to match features and blend images together. Two approaches are considered in the project i.e. A traditional approach and a Deep Learning approach.

Phase1: Traditional Approach

This section focuses on the implementation of the algorithm to generate a panorama out of multiple images with repeated local features. Two image sets need to be captured that have about 30 to 50 percent image overlap between them. This approach has the following steps:

- Corner Detection
- Adaptive Non-Maximal Suppression (ANMS)
- Feature Descriptor
- Feature Matching
- RANSAC for outlier rejection and to estimate Robust Homography
- Blending Images

A. Corner Detection:

Corner Detection is done using cv2.cornerHarris. Experimenting with different values for the parameters, different outputs were generated for Image 1 of Set1 and the results were compared. The resulting images can be seen in figures 1, 2 and 3

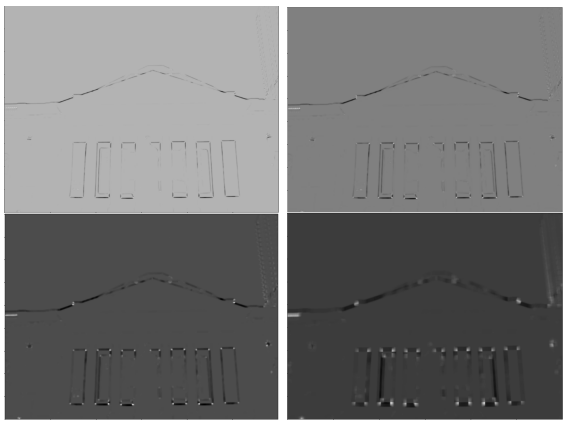

Fig. 1. (From Top Left) Increasing Block Size (2, 3, 5, 10) for cv2.cornerHarris

The parameters are Block Size, K_Size and K. Block Size represents the size of the neighborhood considered for corner detection. It is observed that smaller block size can detect smaller corners but they are sensitive to noise. K_Size is the size of the Sobel Kernel used for computation of the derivative of the image. It is observed that a larger K_Size makes the algorithm less sensitive to noise but it can miss a few details. K is a constant parameter multiplied with the trace of the covariance matrix. It can be seen that a higher value of K makes the algorithm more strict in determining what is considered a corner.

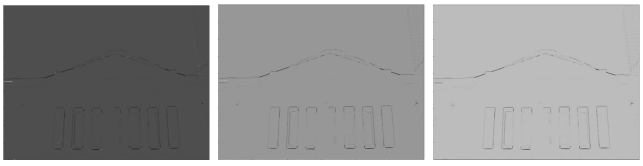

Fig. 2. (From Left) Increasing K (0.03, 0.07, 0.1) for cv2.cornerHarris

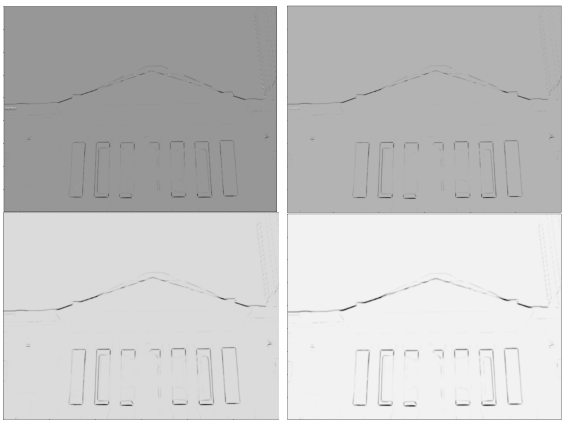

Fig. 3. (From Top Left) Increasing K_Size (3, 5, 9, 15) for cv2.cornerHarris

Based on these observations, the values selected are Block size: 2, K_Size: 1 and K: 0.01 to detect more corners.

Thresholding is done on the output of cv2.cornerHarris to identify the possible corners and the resulting output for the sample sets is shown in the figure 4. Image 1 of each Set is shown here.

B. Adaptive Non Maximal Suppression (ANMS):

After obtaining corners, ANMS is performed to obtain the corners that are equally distributed throughout the image. For that, first the required number of corners are obtained. This number (N_Best) is then provided to ANMS along with the corners of the image to obtain the N_Best corners from the

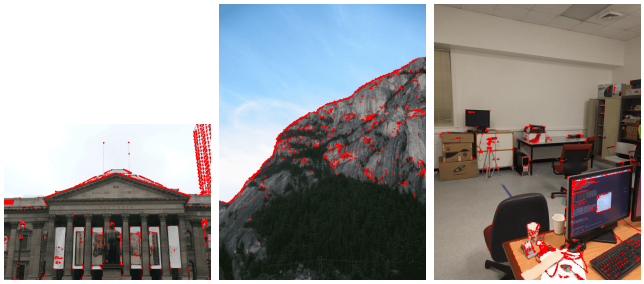

Fig. 4. Corner Detection for Sample Sets using cornerHarris

existing corners. The result of ANMS on sample sets can be seen in figure 5

Fig. 5. Output of ANMS

C. Feature Descriptor:

After obtaining the relevant corners from ANMS, Feature descriptors are created from each corner point. For that, a 41x41 patch centered at the each corner is taken, blurred, reduced dimension and reshaped to get a feature descriptor for each corner. An example of feature descriptors can be seen in figure 6

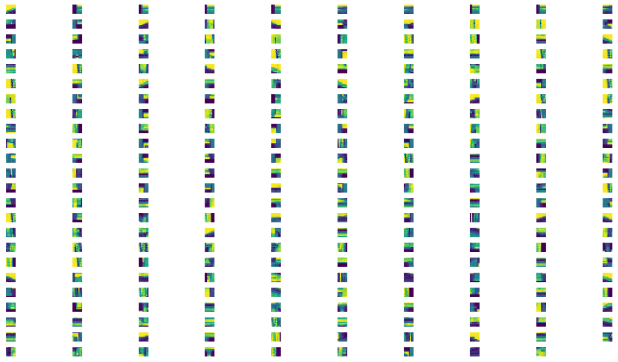

Fig. 6. Feature Descriptors

D. Feature Matching:

Obtaining feature descriptors for each corners, matching of these descriptors is done with the descriptors of another image. Based on this, it is identified which features of the two images match. An exmample of feature mathing can be seen in figure 7. The threshold used to identify if the current match is acceptable is selected as 0.7. After this step, the number of matches is checked. If the number of matches is less than a threshold set, the image is skipped and next image is chosen for blending.

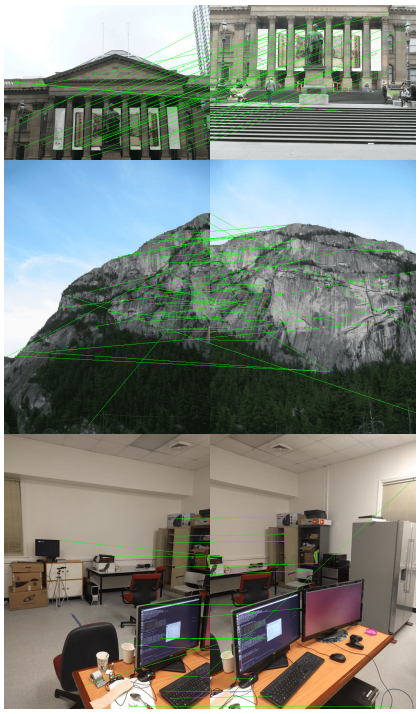

Fig. 7. Feature Matching for Sample Sets

It is evident that in the given images, some of the features did not match properly. For that, RANSAC is used to remove the outlier.

E. RANSAC:

To remove the outliers observed in figure 7, RANSAC is applied. Homography matrices are calculated and used for 4 matches at a time to identify ideal H. 5000 such Iterations are performed and the final homography matrix is used to remove the outliers. The result of RANSAC can be seen in figure 8.

It is evident that only relevant matches are present and the outliers are removed. After RANSAC output has been obtained and all the keypoints are available, we check the number of unique keypoints in both the images. If the number is not same, the algorithm gives out a message ”Improper Matching”. It skips that image and continues to the next image for blending. This is useful in cases where the matching is not proper because the images do not overlap or if many points in one image are matched with a single point on another image.

F. Blending Images:

The images are blended together using a custom function. First the points are transformed using Homography matrix and the new points are used to create a translation of the image. The image is then warped and the background of the image is removed from the image. After that, the warped image is

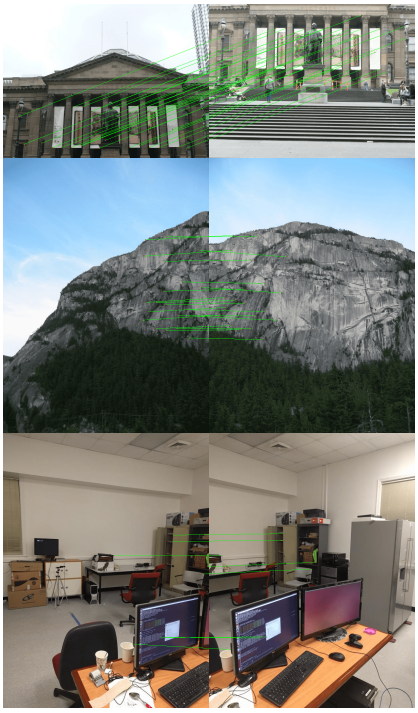

Fig. 8. Using RANSAC to remove outliers

overlayed on the existing image to create a panorama. Because of background removal, the image sometimes develops an very noticeable edge where the two images meet. These points are considered as corners in the next step which causes issues. Weighted sum could have been used here to remove this. The output of the blending for Train Set 1 and Set 2 is shown in figure 9.

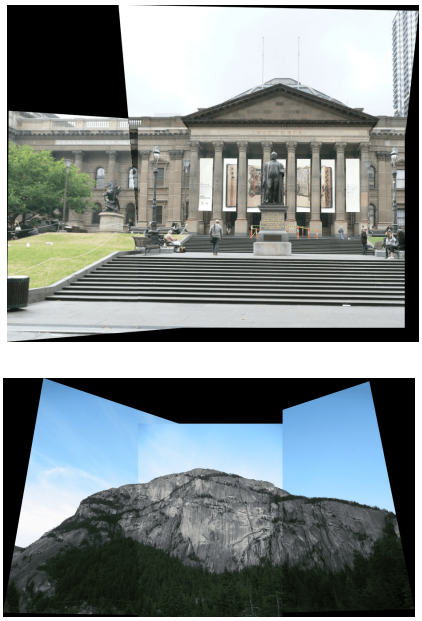

Fig. 9. Blending the images to create a final output for train Set 1 and 2

The algorithm fails to generate the correct panorama from the provided images of Train Set3. Blending work properly for about 4 images but it is not able to find the relevant matches after that. Multiple attempts were made by changing the order of images chosen to generate the panorama (Starting from left, starting from centre, starting from right) which can be seen in figure 10. Attempts were also made to blend the already generated panoramas together but the algorithm could not identify the relevant matches. An example of the algorithm trying to match two panoramas can be observed in figure 11. One of the issues is that once panorama of a few images is generated, there are sharp edges in the image because of previously warped images. The algorithm considers them corners and tries to match them with all corners of the other image. This results in unacceptable results. Also, the images have features that are repeating in later images. For example, there are monitors and desks present in most of the images. While blending a large set, the algorithm finds the matches between them and hence the results become undesirable.

Fig. 10. Attempts to generate panorama for Set 3

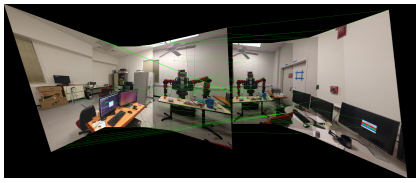

Fig. 11. Improper matching between two panoramas

It should be noted that we have implemented 2 methods for panorama generation. The difference between these two methods is 'in what order the images are taken for panorama generation'. In method 1, all images are taken sequentially from starting till end. In each stage, the panorama generated from previous state is chosen and warped over the new image. Thus, the perspective is as if looking towards the last image of the set. In Method 2, the image that is in the middle of the set is taken and then images on both sides are stitched onto the centre image. Here, the perspective is as if looking towards the centre image of the image set. This difference is evident in figure 12. Both the methods have been used for different test cases in this project depending on the image set.

Fig. 12. Difference in perspective between Method 1(Left) and Method 2(Right)

G. Custom Sets:

In this section, we apply the created algorithm for custom sets of images that have been captured.

1) Custom Set 1: Custom Set1 contains 4 input images which can be observed in figure 13 and the step by step panorama generation can be observed in figure 19

Fig. 13. Input Images for Custom Set 1

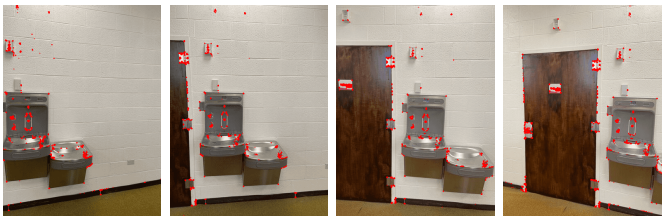

Fig. 14. Corner Detection

Fig. 15. ANMS

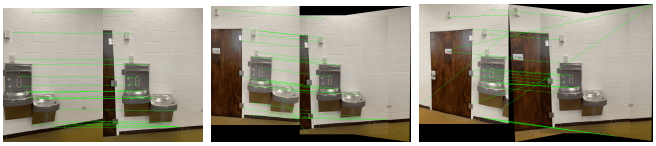

Fig. 16. Feature Matching

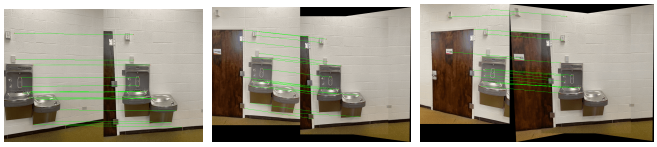

Fig. 17. RANSAC

Fig. 18. Intermediate Generated Panorama for Custom Set 1

Fig. 19. Final Generated Panorama for Custom Set 1

2) Custom Set 2: Custom Set 2 contains 3 images as shown in figure 20 and the output is shown in figure 25.

Fig. 20. Input Images for Custom Set 2

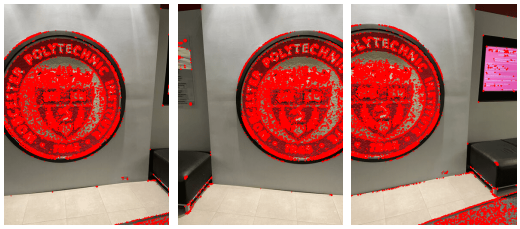

Fig. 21. Corner Detection

Fig. 22. ANMS

Fig. 23. Feature Matching

Fig. 24. RANSAC

Fig. 25. Final Generated Panorama for Custom Set 2

H. Test Sets:

1) Test Set 1: Test Set 1 contains 4 images as shown in figure 34 and the output is shown in figure 25.

Fig. 26. Input Images for Test Set 4

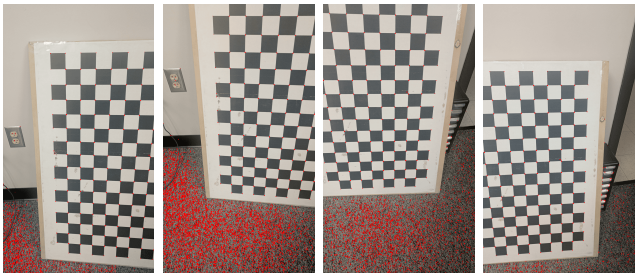

Fig. 27. Corner Detection

Fig. 28. ANMS

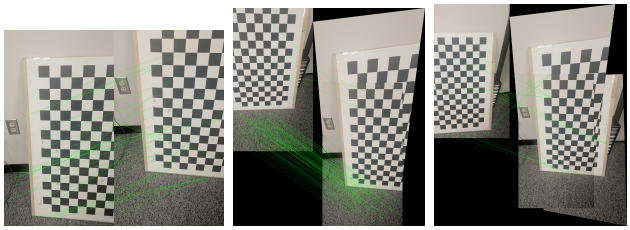

Fig. 29. Feature Matching

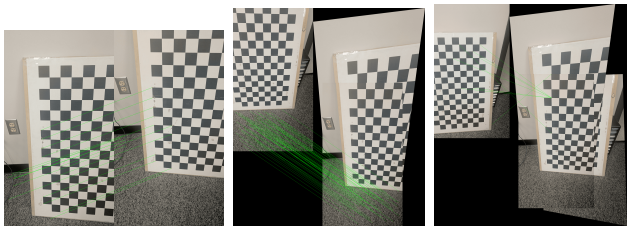

Fig. 30. RANSAC

It is evident that the checkerboard pattern is a difficult pattern to identify as there are a lot of corners that are the same and that can adversely affect the panorama. But the thing to note here is that in the first 3 images, the matching happens very well. The reason behind this is the features outside the checkerboard, especially the carpet. Because of this, these images could be blended together very well. The last image does not have enough extra features that are common with the previous images. Because of this, it does not blend as desired with the rest of the panorama. The matching and RANSAC are both shown in figures 29 and 30 respectively. The final output of the image is shown in figure 31. We tried changing the order of the images that are used for panorama generation. We reversed the order of the images and the resultant output is also shown in figure 31. Thus it is clear that the algorithm does not provide desired results when dealing with repetitive patterns. In this example, when the algorithm has to rely only on the checkerboard pattern, the blending provides undesirable results.

Fig. 31. Final Generated Panorama for Test Set 1 with Normal Order(Left) and Reversed Order(Right)

2) Test Set 2: Test Set 2 contains 9 images. The algorithm is unable to find proper matches and the panorama that is generated consists only of the first three images. The reasons for this is that there aren't enough matching features after that in the two images. This can be seen in figure 32. Thus, it skips the images with the message "Improper Matching". Ideally, the algorithm should skip images 4,5 and 6 from the test set and then blend with images 7 to 9 since they have good overlap. But it considers those also as improper matches. The reason behind that could be changes in illumination. Even though image 3 and image 7 have good overlap, image 7 is noticeably brighter than image 3 which could be the reason behind this. The attempt at blending can be seen in figure 33. Method 1 described in 'Blending Images' section is used to generate this panorama.

Fig. 32. Improper Matching for Test Set 2

Fig. 33. Generated panorama for Test Set 2

3) Test Set 3: Test Set 3 contains 3 images as shown in figure 34 and the output is shown in figure 25.

Fig. 34. Input Images for Test Set 3

Fig. 35. Corner Detection

Fig. 36. ANMS

Fig. 37. Feature Matching

Fig. 38. RANSAC

Fig. 39. Final Generated Panorama for Test Set 3

4) Test Set 4: The Test Set 4 contains 5 images. 3 of them are the first 3 images from Test Set 2 and rest of the images are from Set 1 and Set 3. While generating the panorama, the algorithm identifies and blends the first 3 images and skips the rest 2 images with the message ”Improper Matching”. The output of the blending can be observed in figure 41.

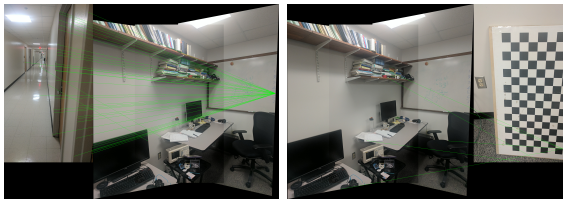

Fig. 40. Improper matching output after RANSAC in Test Set 4

Fig. 41. Final Generated Panorama for Test Set 4

I. Conclusion

In Phase1, We came to learn about to stitch two or more images in order to create one seamless panorama image. Further, we can generate distinct outputs with different value of the parameters used.