CV Assignment2 Report

Siyuan Yin 2100017768

October 2023

1 Code Implementation

1.1 Harris Corner Detection

The Harris Corner Detection is a commonly used algorithm.The overall philosophy of the implementation is to detect corners in an image by calculating the Harris response, which is a measure of corner strength. The *corner_selection* function then selects the corners that exceed a certain threshold and are separated by a minimum distance. This approach allows the identification of distinctive features in an image. Figure 1 are the result on the example images.

Figure 1: Harris Corner Detection

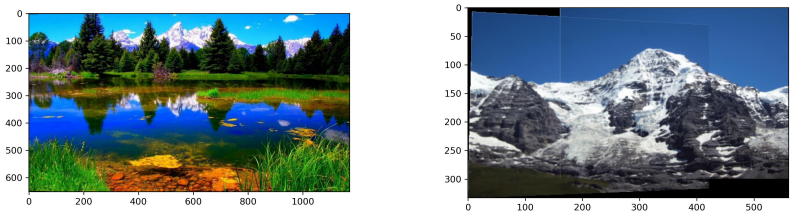
1.2 RANSAC for Homography

The function *align_pair(pixels_1, pixels_2)* is used to align two sets of matched pixel coordinates *(pixels_1 and pixels_2)* using the RANSAC (Random Sample Consensus) algorithm to compute the optimal homography transformation matrix.

- *num_iteration, inlier_threshold num_sample* are parameters for controlling the RANSAC algorithm, Number of RANSAC iterations to perform, maximum distance threshold for considering a point as an inlier and number of points randomly selected to estimate the homography matrix in each RANSAC iteration, respectively.
- In each iteration, it ramdomly select some points to estimate a homography matrix. The homography then is used to transform the pixel coordinates. Then the Euclidean distances between the transformed pixel1 and pixel2. Inliers are determined bt counting the number of distances less than the threshold.
- After all RANSAC iterations, the function identifies the best set of inliers and uses them to compute a final homography matrix.

1.3 Pair-image Stitching

The *stitch_blend* stitches the pair-image into one. It calculates the boundary of the new image based on the coordinates of the projected corners and the corners of the other image. Then the function use the inverse homography to map pixels back to the originale image's coordinates. Then the function perform the interpolation to blend the pixels from both images.Further more, vectorizing the operations using NumPy can avoid the nested for-loops. Figure 2 are the result of the example images.

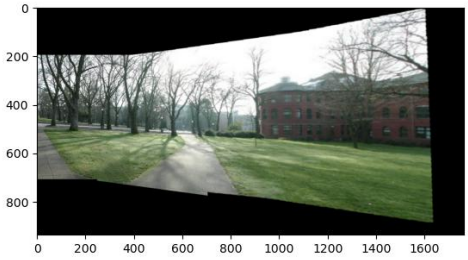


(a) (b)

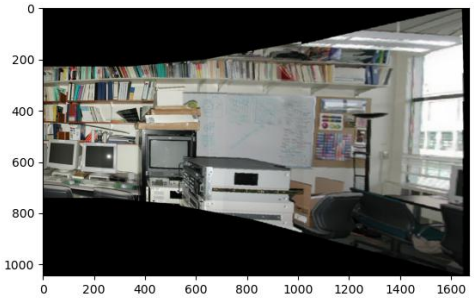
Figure 2: Stich blend

1.4 Panorama Generatin

The function *generate_panorama* generates a panorama by stiching together a sequence of ordered images. This function iteratively aligns and blends each image into the existing panorama, extending it gradually to create a complete panoramic image. Figure 4 show the result on example images.



(a)



(b)

Figure 3: panoramas

2 Analyze

5.Explore how to use more powerful and robust feature detectors and descriptors to improve the panorama, make comparisons to show the improvements, and analyze.

Using Stitcher class in cv2 packages, we can get panorama with higher quility. Here are some reasons:

- **Feature Detection and Matching**: First, Stitcher performs feature detection on each input image, typically using keypoint detection algorithms like SIFT, ORB, or SURF. It then calculates feature matches between pairs of images, representing corresponding points or features in different images.

- **Image Stitching:** The stitching process involves overlaying all the images to create a panoramic image. To maintain image continuity, Stitcher performs pixel-wise blending and fusion based on the calculated transformations, eliminating potential discontinuities or stitching artifacts.
- **Optimization and Correction:** Additional steps may include optimization, distortion correction, color correction, and other adjustments to ensure that the stitched panoramic image has good visual quality.

However, in the above implementation, there are several shortcomings:

- Regarding feature detection, we've only implemented one step of SIFT, neglecting complex processes like Gaussian pyramids, which significantly impacts the effectiveness of our feature detection.
- During image stitching, we've used too many for-loops, making the panorama generation process very slow. Additionally, we haven't resized the new image, resulting in numerous black regions in our results.

(a)

(b)

Figure 4: panoramas after improving