Project #4: Feature Matching

Yang Xu (yxu71@vols.utk.edu)

Problem Description

In this project, this task is to create a simple stitcher that can perform image alignment. To address the problem of homograph matrix for alignment, we address this problem through feature matching

Code Design

Function	Description
find_points(image,block_size=5, max_block=9,k=0.05)	This function will use Harris corner detector to find key points. It refines key points by first thresholding and second finding local maximum.
ori_points(image,points)	Use 3X3 Sobel filter to convolve image and calculate orientation of pixel. It only returns orientations given the list of points as input.
points_intensity(image,points)	This descriptor function simply gets 5X5 grayscale pixel values around the key point.
rotation(image,angle)	Clockwise rotate image in a certain angle
points_descriptor(image,points, orientation)	Different from points_intensity(), points_descriptor() will take image, points and orientations of points to calculated rotated patch descriptor. It first cuts a broader patch (larger than 40X40) and then rotates the patch by the orientation. Next, it cuts a 40X40 patch and resizes it to 5X5. Flatten the resized patch and then normalize the vector.
feature_match(des1,des2,ratio= 0.5)	It takes two lists of descriptors and finds the matches in the two lists. Parameter ratio sets the threshold of good match. This ratio test is in both ways.

Approach

1. Detecting key points by Harris corner detector
2. Pixel intensity descriptor or rotated patch descriptor
3. Finding good matches via ratio test
4. Finding homograph matrix via RANSAC
5. Stitches images

Running Code

All functions used in this project are written in feature_matching.py. The implementation of the 5-step approach is written in implementation.py with brief comments.

You can also run all steps in the terminal by the commend line below

python Project4_Final.py images/translation/yosemite1.jpg images/translation/yosemite2.jpg 1 0.15

The first two arguments are the paths of two images. 0 or 1 indicates using pixel intensity descriptor or ratoted patch descriptor. The fourth argument is the threshold in ratio test.

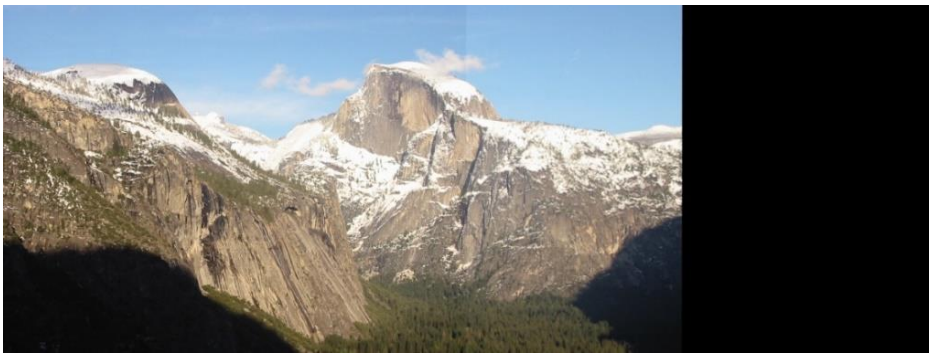
Example Results

Illumination

Translation

Perspective

Stitching



Result by rotated patch descriptor

Image alignment

From left to right: original image, aligned image by pixel intensity descriptor, aligned image by rotated patch descriptor, true aligned image

In this project, I experimented feature matching by using two different descriptors mentioned above. To compare the performance of feature matching from the two descriptors, I tested in three conditions, illumination, translation, and perspective respectively. Experimental results were listed above (Left: pixel intensity descriptor; Right: rotated patch descriptor). Results from illumination and translation showed both descriptors can reasonably match key points in images and have comparable performance. The difference appears when images present a strong different on perspective. Rotated patch descriptor can still find good matches in two images, but pixel intensity descriptor not. Image alignment also showed using matches found by rotated patch descriptor gets a good homograph matrix and returns a image with a better resemblance to the image on the right.

Conclusion

A good designed descriptor can improve much better feature matching task, than simply using pixel intensity descriptor. However, facing a more complicated situation, a sophisticated design of key point detection and descriptor are required, say SIFT feature for feature matching.