Homework 2
Panorama Stitching

Due date: 23:59 Sunday December 29th (2024)

If the camera center is fixed, or if the scene is planar, different images of the same scene are related by a homography. In this project, you will implement an algorithm to calculate the homography between two images. Several images of each scene will be provided. Your program should generate image mosaics according to these estimated homographies.

1. The basic algorithm

[20 marks]

Feature detection, descriptor, and matching: To estimate the homography between two images, we need to first identify corresponding points in these images. We can use a feature detector to identify feature points in both images, and find corresponding points by comparing their descriptors. Popular detectors include Harris, FAST and SIFT detector. For feature descriptors, we might use either the SIFT descriptor or the concatenated pixel values within a local window. At last, the feature matching should be conducted to find the correspondence between images. In order to avoid ambiguous matches, the ratio between the distance to the closest neighbor and the distance to the second closest neighbor as mentioned in the class should be adopted. (*You are allowed to use any existing library for feature detector and descriptor, but you should implement the feature matching by yourself.*)

[40 marks]

Estimation of homography: Note that we should normalize the pixel coordinates of these feature points, such that the lower-left corner of the image has coordinates of (-1,-1) and the upper-right corner of the image has coordinates of (1,1). We can then apply the RANSAC and DLT algorithms to obtain the largest set of inliers. Basically, the program will randomly select 4 corresponding pairs at a time to use the DLT algorithm to compute a homography. Then the other points are classified as inliers or outliers according to how good they fit to this homography. We need to decide the number of iterations according to our estimated ratio of outliers. Among all the results during random sampling, choose the largest set of inliers. We can then apply the M estimator algorithm to all these inliers.

[20 marks]

Image stitching: Once the homography is estimated, we can use it to transfer images to align with a reference image to create mosaics. We can simply use linear blending to decide pixel values at overlapped regions. In other words, if a pixel is covered by

two images, we can retrieve its values from both images and set it to the average value. More advanced blending methods such as multi-band blending can be also applied.

2. Comparison of different feature descriptors:

[20 marks]

The Different feature descriptors have different performance. They would generate matches with different percentage of outliers. This ratio will affect the number of iterations we need for the RANSAC algorithm. We can use the smallest number of RANSAC iterations that ensures a correct result to compare these feature detectors/descriptors. Please compare the SIFT descriptor vs. descriptor formed by concatenated pixel values. To decide the number of iterations needed, we can choose a smallest number that ensures the RANSAC algorithm succeed in 50% percent chance of many independent tests, say 100 tests.

3. Grading

You are required to submit both your report and source code (Matlab or C/C++ or Python) to Learning in ZJU (学在浙大). Your report should be in the **pdf** format with no more than 6 pages.

For the basic algorithm, your report should include **at least one** stitching image for each example data. For the comparison of different feature descriptors, your report should include **at least two figures:** (a) the matching result for any two input images of a certain example data using descriptors formed by concatenated pixel values; (b) the matching result for the same images in (a) using SIFT descriptor. A table summarizing the feature matching results is also expected.

In the report, you are expected to discuss your findings through the experiment. For example, what brings troubles to homography fitting? What kind of data works best/worst? How the implemented algorithm can be improved?

Please zip everything together in **a single file**, and **name it with your student id** before uploading.