ETH Zürich
Computer Science Department

Prof. Marc Pollefeys
Prof. Luc Van Gool
HS 2014

Computer Vision

Exercise 4

Hand-out: 16-10-2014
Hand-in: 23-10-2014 14:00

Objective:
In this exercise you will learn how to establish correct point correspondences in an automatic way.

4.1 Feature extraction and matching (10%)

Capture a pair of images from two different viewpoints from a static scene (You can use the same pair of images that you have captured or the provided image pairs from Exercise 3). The images should have some overlap such that point correspondences can be established. Undistort your images to make straight lines in the scene appear straight in the image. Download the SIFT implementation from Lowe's webpage (`http://www.cs.ubc.ca/~lowe/keypoints/`). Or you can use any other SIFT implementations that can be found online. Proceed on to extract and match the SIFT features from the image pair. Visualize the matches by plotting line segments between the matched points. You can either:

- Display just the first image. For each match, draw a line segment from the point in the first image to the coordinates of the second point, but also in the first image. This gives a sense of the motion of the point from one image to the next.
- Display both images side by side. For each match, draw a line segment from the point in the first image to the point in the second image. This shows exactly which features were matched (this code is given in Lowe's implementation).

4.2 8-Point RANSAC (Total 60%)

You might observe some wrong matches (outliers) from the previous step. In this step, you shall implement the RANSAC algorithm to automatically remove the wrong matches. The eight-point algorithm that you have implemented in Exercise 3 will be used as the model for the RANSAC algorithm.

a) Simple RANSAC (40%)

Implement the eight-point RANSAC algorithm and keep the termination criteria fixed at 1000 trials. The error measure that you should implement to determine if a point correspondence is an outlier or not is given by

$$d(\mathbf{x}', \mathtt{F}\mathbf{x}) + d(\mathbf{x}, \mathtt{F}^\top \mathbf{x}') \tag{1}$$

This computes the sum of the perpendicular distances of the point to the epipolar line in the two respective images. You should specify a threshold, for example 2 pixels, in the RANSAC algorithm for the error measure to determine whether a point correspondence is an outlier or not. Visualize the matches (inliers) from the consensus set with the highest RANSAC score (same visualization as in 4.1). Vary the error threshold and compare the total inlier counts.

b) Adaptive RANSAC (10%)

The best solution from RANSAC is given by running an exhaustive search of all possible sets of 8 points from the correspondences but this is often computationally infeasible. It is more efficient to terminate the RANSAC process after M trials if we know with a probability p that at least one of the random samples of $N = 8$ points from these M trials is free from outliers. For this to happen, Equation 4.2 has to be satisfied.

$$p = 1 - (1 - \mathrm{r}^N)^M \tag{2}$$

r is the largest inlier ratio found after every RANSAC trial. Modify your RANSAC code in Part (a) so that it terminates after M trials following the condition from Equation 4.2. Fix the value of p at 0.99 in your implementation. Report the value of M.

c) Sampson Error (10%)

A first-order approximation to the geometric error (Equation 4.2) is given by the Sampson distance

$$\frac{(\mathbf{x}'^\top \mathtt{F}\mathbf{x})^2}{(\mathtt{F}\mathbf{x})_1^2 + (\mathtt{F}\mathbf{x})_2^2 + (\mathtt{F}^\top \mathbf{x}')_1^2 + (\mathtt{F}^\top \mathbf{x}')_2^2} \tag{3}$$

where $(\mathtt{F}\mathbf{x})_i^2$ is the square of the i-th entry of the vector $\mathtt{F}\mathbf{x}$. Implement a function that computes the Sampson distance of a point correspondence to the corresponding epipolar line of the point correspondence in the other image. Replace the geometric error from your code in Part (b) with the Sampson distance. Run the code and visualize the matches (inliers) from the consensus set with the highest RANSAC score. Vary the error threshold and compare the total inlier counts.

4.3 5-Point RANSAC (30%)

The minimum number of point correspondences needed to compute the essential matrix `E` is 5 since `E` consists of a rotation (3 DOF) and a translation (2 DOF since scale is unknown). In contrast to the 8-point algorithm that computes a linear solution, the 5-point algorithm solves for the roots of a 10-th degree polynomial. Each *real* root corresponds to a possibly correct essential matrix, hence a maximum of 10 solutions. A way to check for the correct solution is to compare the total Sampson distance from all the other point correspondences.

The solution with the lowest total Sampson distance is the correct solution. Implement the 5-point RANSAC algorithm. Report the best essential matrix and visualize the matches (inliers) from the consensus set with the highest RANSAC score. The source code for 5-point algorithm from Nistér and Stewenius is provided. Note that to use the algorithm, you have to input normalized image points ($\hat{\mathbf{x}} = \mathtt{K}^{-1}\mathbf{x}$).

Discuss the difference in the number of trials needed with the 8-point and 5-points algorithms. Is it worth solving a 10^{th} order polynomial ?

4.4 Bonus: (5%)

With the essential matrix computed in the previous section, triangulate and plot all inlier 3D points together with the two camera poses.

4.5 Hand in:

Write a short report explaining the main steps of your implementation and discussing the results of the methods. Be sure to include:

- Images overlaid with extracted SIFT features.
- Plots showing inliers and outliers.
- Plots showing total inlier counts with different error thresholds.

Send the report together with your source code to **fede@inf.ethz.ch**.