Computer Vision

Exercises of Lab 5

Exercise 5.1: Stereo Vision

The goal of this exercise is to understand the principles and experiment with some of the challenges in stereo vision.

Look at the images *left.jpg* and *right.jpg*. Intuitively, how would you extract depth information from these images?

Make sure that you have VLFeat and Peter Kovesis' Computer Vision Matlab functions – direct link here. Open *Exercise5.1.m* and change the variables VLFEATROOT and MATLABFNSROOT to the unpacked paths at your hard drive.

Run the Matlab script and study Figure 1. How has the camera been moved from the left to the right image?

Homography and Fundamental Matrix

Figure 2 illustrates coordinate transformations using a homography estimated from SIFT point correspondences using RANSAC (as usual). Try clicking in one image and look at how the coordinate is transformed in the other. Does the transformation work satisfactorily?

Instead of estimating a homography, we now want to estimate the fundamental matrix, again using point correspondences and RANSAC. Open *fundmatrix.m* and implement the missing code. Use the principle in slide 8 of "Lecture4.1-2 StereoEpipolarAndImageAlignment.pdf".

Run *Exercise5.1.m* again and play with Figure 3. Compare the coordinate transformations in Figure 2 with the epipolar constraint in Figure 3. How can we use the epipolar lines?

Rectification

Figure 4 illustrates the rectified images. We have used the fundamental matrix for transforming the two images such that the epipolar lines are horizontal. In this way, it is easy to implement disparity search using simple scanlines.

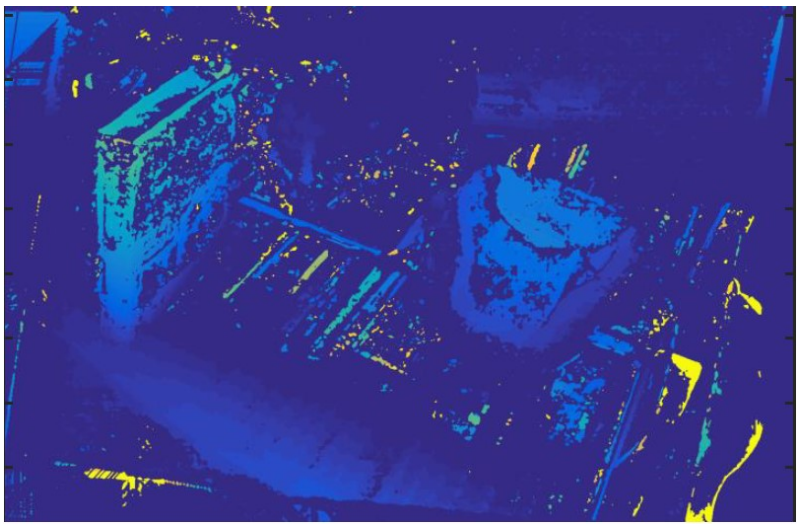
Disparity Map

Figure 5 illustrates a disparity map constructed using Matlab's built-in dense disparity search. Try changing the variables *windowSize* and *disparities* (must be a multiple of 16) and look at the implications.

Given a number of feature points in the left image, implement a disparity search in the right image along the epipolar lines. Your task is (for each feature) to run through all possible positions in the right image and calculate a matching error. Use the function NCC(patch1, patch2) that takes a patch from each image and calculates an error. Keep the errors in the vector costVec.

Figure 6 displays the cost vector for one feature at a time (press a key to process the next). Are all the feature points easy to match uniquely?

Try changing the variables windowSize and maxDisparity and look at the implications.



Depth Estimation

Calculate a rough estimate of the distances for each feature. Use the correspondences fL and fR in the left and right images, respectively. Use the principle in slide 4 of "Lecture4.1-2 StereoEpipolarAndImage Alignment.pdf". Remember that the distances are calculated in an arbitrary scale, since we do not know the intrinsics of the camera. Therefore, to assess the correctness of the found distances, try to look at the relative distances of the features.