Computer Vision - Exercise 8
Structure from Motion

Xingze Tian 17-941-527

December 3, 2017

1 Feature extraction and initialization with epipolar geometry

- **SIFT Extraction**
  This is done via `vl_sift` and the feature matching result is as shown in Figure 1:

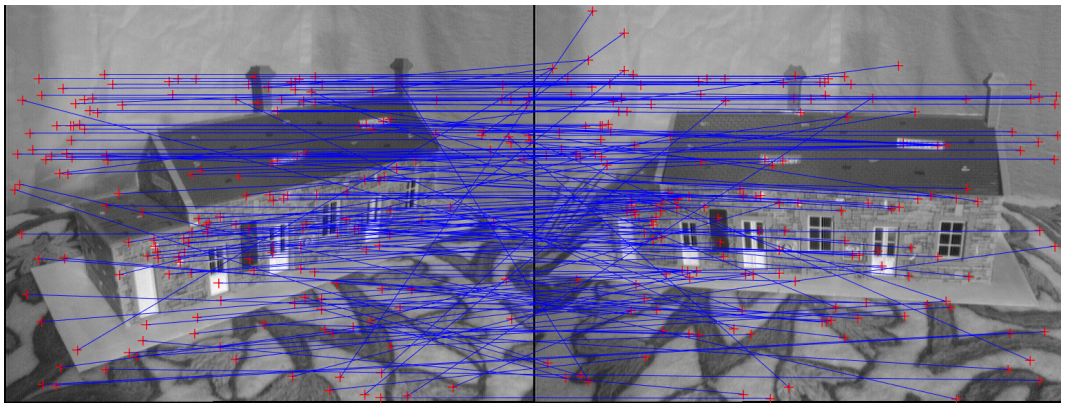
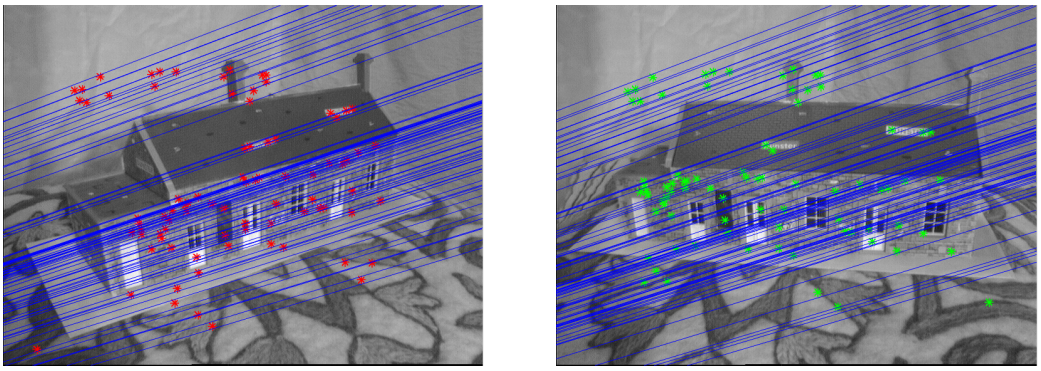


Figure 1: Feature matching using SIFT: img0 and 4

- **8-Point RANSAC**
  I used the method `ransacfitfundmatrix()` to first find the fundamental matrix $\mathbf{F}$ (which can also be optimized using the `funddist()` method) and the inlier-set. The essential matrix $\mathbf{E}$ is then computed as $\mathbf{E} = \mathbf{K}^{-1}\mathbf{F}\mathbf{K}$. The projection matrix of the second view can thus be calculated by the `decomposeE()` method.

  The results are as the following:



(a)img000 (b)img004

Figure 2: Epipolar geometry for initialization

$$\mathbf{F} = \begin{bmatrix} 0.0000 & 0.0000 & -0.0034 \\ -0.0000 & 0.0000 & -0.0066 \\ 0.0012 & 0.0025 & 0.6156 \end{bmatrix} \quad \mathbf{E} = \begin{bmatrix} 1.1245 & 3.3824 & -0.2116 \\ -0.1748 & 0.9202 & -4.1623 \\ 1.3754 & 4.0669 & -0.0551 \end{bmatrix}$$

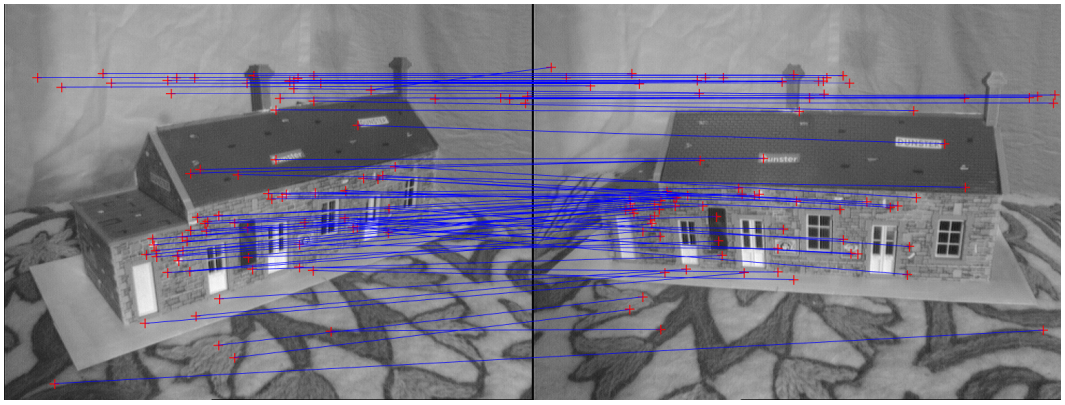


Figure 3: Feature matching inliers: img 0 and 4

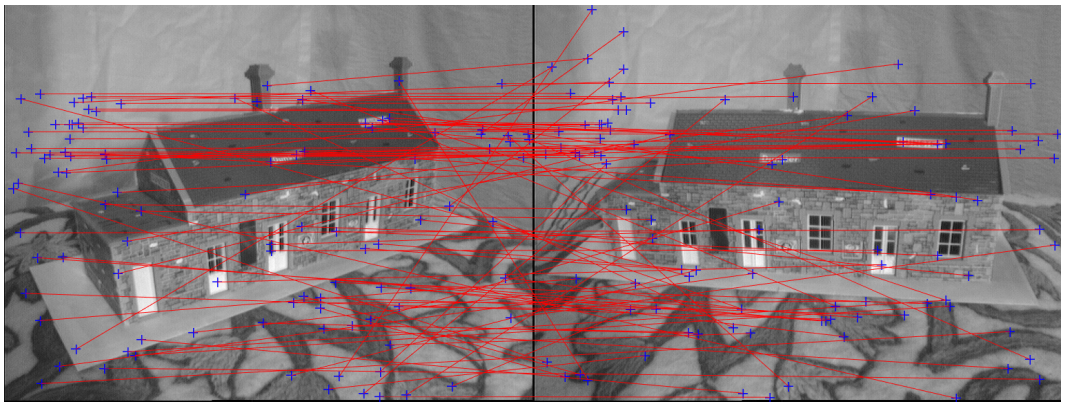


Figure 4: Feature matching outliers: img 0 and 4

- **Triangulation**
  This is done using the `linearTriangulation()` method.

2 Triangulation and adding new views

- **6-Point RANSAC**
  The feature matching process is similar as the previous section using the `vl_ubcmatch()` method. Here I used image 0 as the existing view and match the new views with the calculated inliers in image 0. The projection matrix is computed using the `ransacfitprojmatrix()` method. The results are as the following:

3 Plotting

I used the matlab `plot3()` method and the provided `drawCameras()` to plot all the 3D points calculated during the triangulation process and the cameras. The results are shown as below:

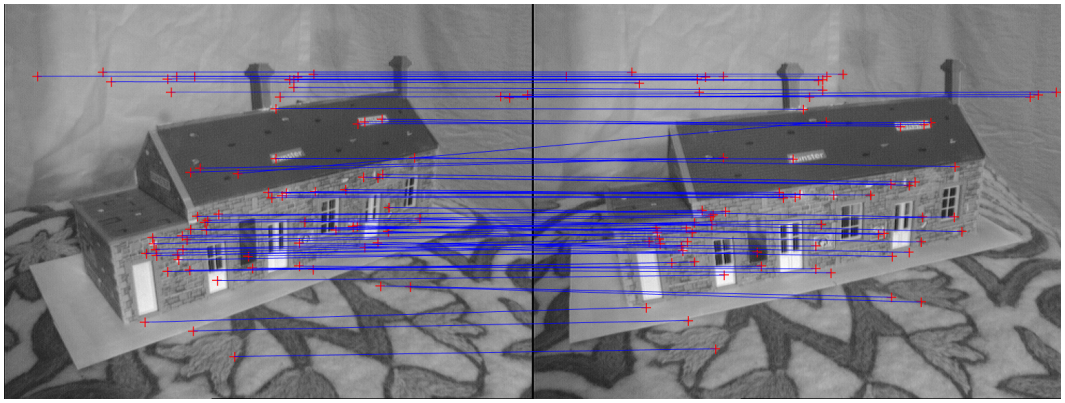


Figure 5: Feature matching inliers: img 0 and 1

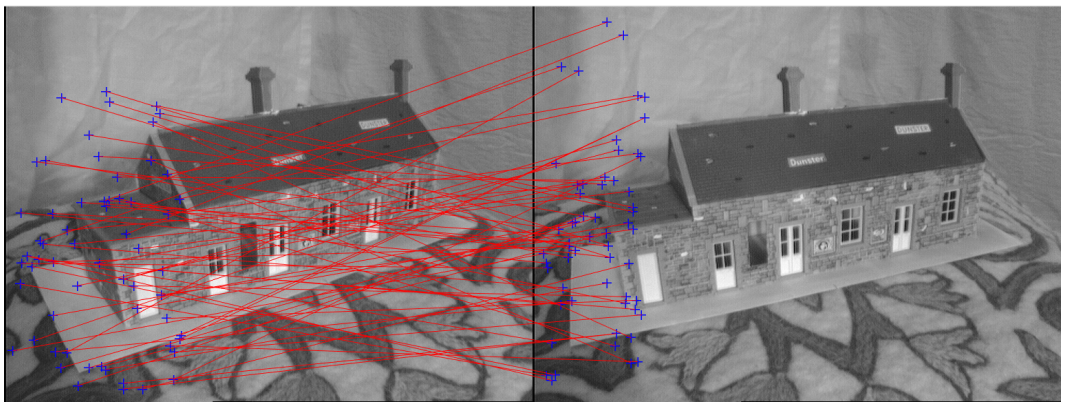


Figure 6: Feature matching outliers: img 0 and 1

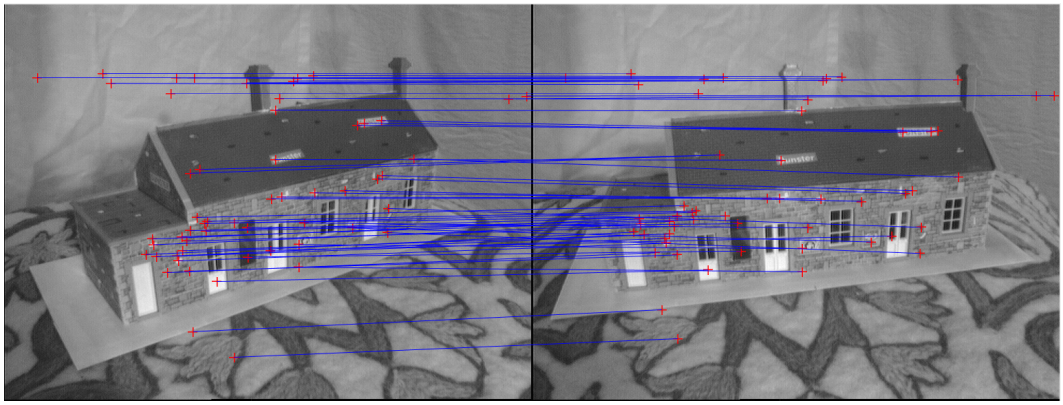


Figure 7: Feature matching inliers: img 0 and 2

4 Dense reconstruction

I did some experiments on this section using the matlab computer vision toolbox. First I rectified the two images (img 0 and 4) using the matlab method `estimateUncalibratedRectification()` with the fundamental matrix calculated as input. After that I computed used the disparity map using the `disparity()` method with the rectified images. The depth map can be computed with `reconstructScene()` method using the disparity map. Finally I used the `create3DModel()` method to reconstruct the scene.

During the experiments I got some weird results as shown in Figure 12 and 13. When converting disparity map to depth maps I also encountered some problems, however, due to time limit, this task is not completely debugged.

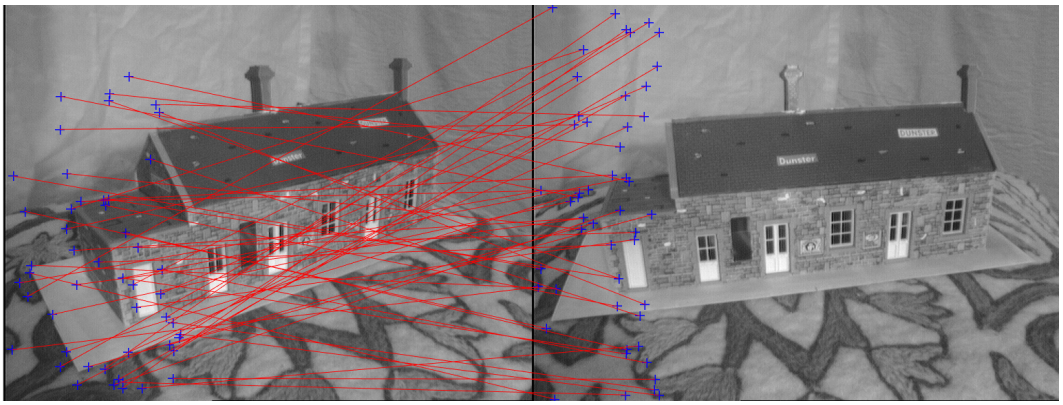


Figure 8: Feature matching outliers: img 0 and 2

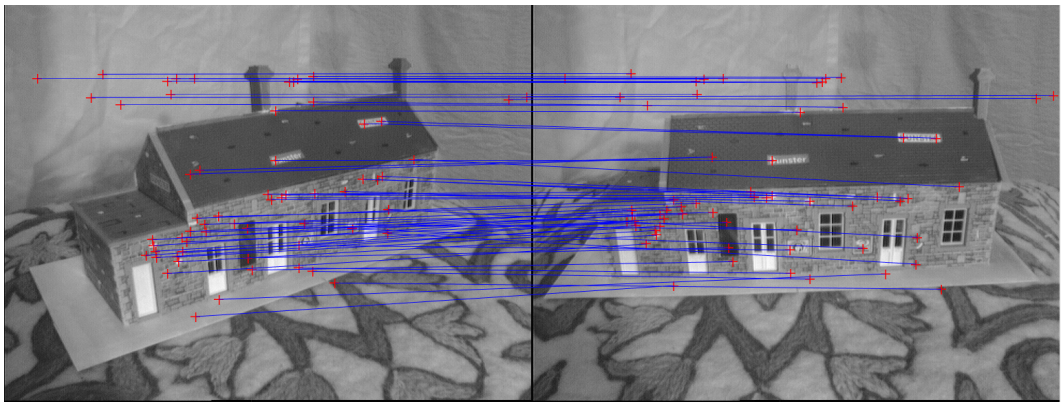


Figure 9: Feature matching inliers: img 0 and 3

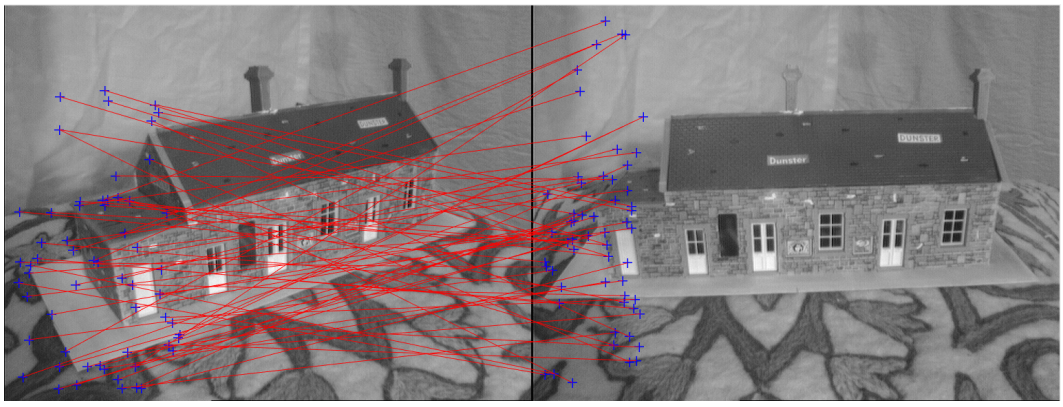


Figure 10: Feature matching outliers: img 0 and 3

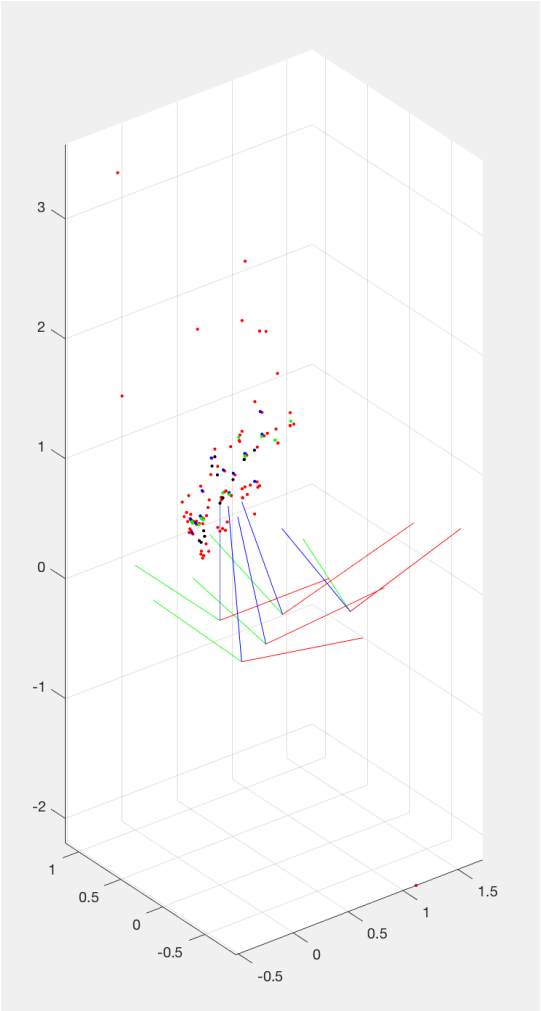


Figure 11: Triangulated inlier matches and cameras

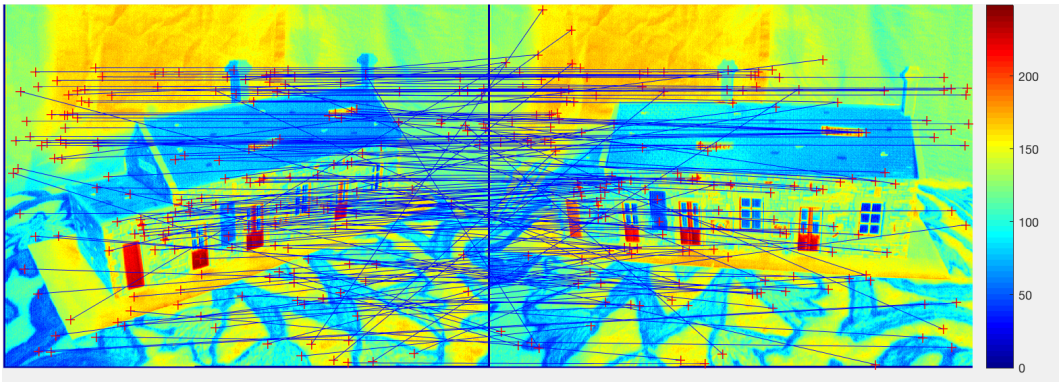


Figure 12: Two images used in spectrum display

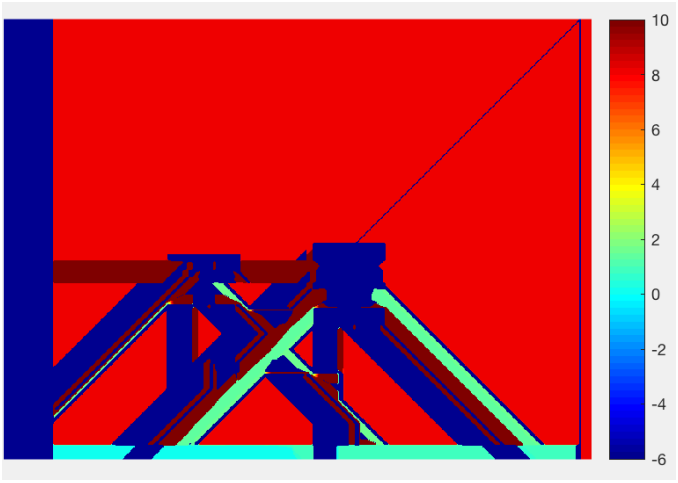


Figure 13: One disparity map constructed

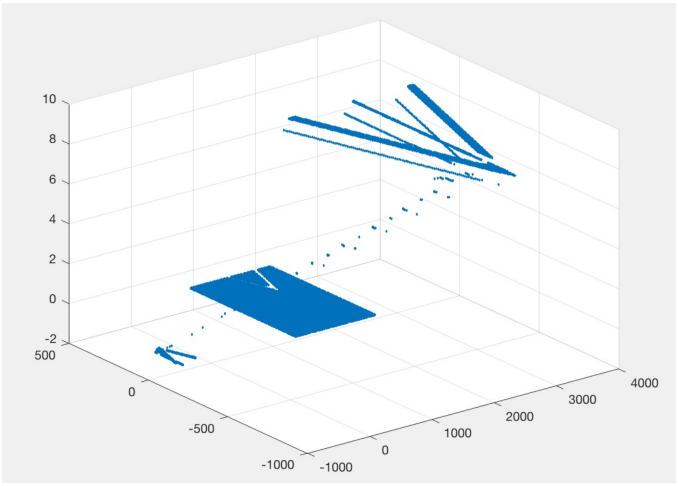


Figure 14: A weird reconstructed scene using just the disparity map