16-720 Computer Vision: Homework 2
Feature Descriptor & Homographies & RANSAC

Anbang Hu
anbangh@andrew.cmu.edu

Due: Feb 18, 2016

Student(s) that I worked with: Silun Wang

1 Keypoint Detector

1.1 Gaussian Pyramid

Nothing to be done.

1.2 The DoG Pyramid (5 pts)

Q 1.2: Implemented in `createDoGPyramid.m`.

1.3 Edge Suppression (10 pts)

Q 1.3: Implemented in `computePrincipalCurvature.m`.

1.4 Detecting Extrema (10 pts)

Q 1.4: Implemented in `getLocalExtrema.m`.

1.5 Putting it together (5 pts)

Q 1.5: Implemented in `DoGdetector.m`. The detected keypoints for `model_chickenbroth.jpg` are shown as red dots in Figure 1. Code in `showKeypoints.m` is used to generate the figure.

Figure 1: Detected keypoints using DoG detector

2 BRIEF Descriptor

2.1 Creating a Set of BRIEF Tests (5 pts)

Q 2.1: I chose method (II) in the paper - $(\mathbf{X}, \mathbf{Y}) \sim$ i.i.d. $\mathcal{N}(0, \frac{1}{25}S^2)$. The center is chosen to be $(round(S/2), round(S/2))$ and covariance matrix is chosen to be $[\frac{1}{25}S^2, 0; 0, \frac{1}{25}S^2]$, where S is the `patchWidth`. Values outside the boundary (i.e. < 1 or $> S$) is clamped at the boundary. Implementation is in `makeTestPattern.m`. This routine is run for the given parameters `patchWidth` $= 9$ and $n = 256$ and results are saved in `testPattern.mat`.

2.2 Compute the BRIEF Descriptor (10 pts)

Q 2.2: Implemented in `computeBrief.m`.

2.3 Putting it all Together (5 pts)

Q 2.3: Implemented in `briefLite.m`.

2.4 Check Point: Descriptor Matching (5 pts)

Q 2.4: Implemented in `testMatch.m`.

Results with the two incline images and with the computer vision textbook cover page are shown in Appendix A. We can see from those figures that for pictures with no or little rotation, the matchings are satisfying. On the contrary, for pictures with some extent of rotation, the matchings appear to be messed up. This is because BRIEF descriptor is not rotation invariant.

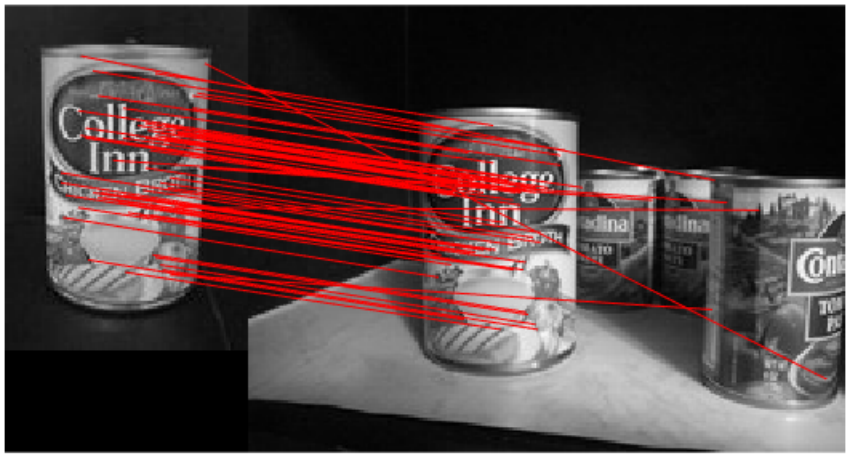

Figure 2: BRIEF matches for `model_chickenbroth.jpg` and `chickenbroth_01.jpg`.

2.5 BRIEF and rotations (10 pts)

Q 2.5: Implemented in `briefRotTest.m`.

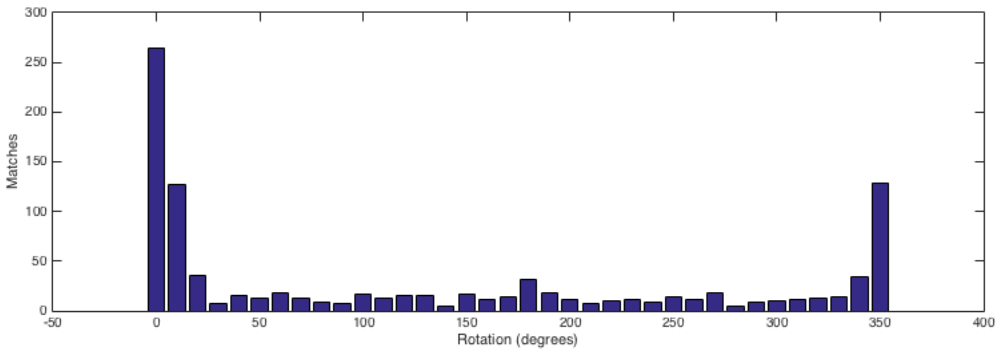

Figure 3: Bar graph for rotation angle vs number of correct matches

Figure 3 plots the bar graph for rotation angles vs number of correct matches (rotation angles are incremented by 10 degrees each time). We can see that the BRIEF descriptor behaves well when there is no or little rotation. This is because the BRIEF descriptor always compares fixed point locations. When rotation happens, the these fixed point locations are rotated as well, resulting in the original fixed locations comparing to other point locations.

3 Planar Homographies: Theory (20 pts)

Q 3.1:

(a) Since $p^i = (x_i, y_i, 1)^T$, $q^i = (u_i, v_i, 1)^T$, we can write $p^i \equiv \mathbf{H}q^i$ as:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \tag{1}$$

We can expand Eq(1) into a linear system:

$$x_i = h_{11}u_i + h_{12}v_i + h_{13} \tag{2}$$

$$y_i = h_{21}u_i + h_{22}v_i + h_{23} \tag{3}$$

$$1 = h_{31}u_i + h_{32}v_i + h_{33} \tag{4}$$

With Eq(4), we can rewrite Eq(2,3) as

$$x_i = \frac{x_i}{1} = \frac{h_{11}u_i + h_{12}v_i + h_{13}}{h_{31}u_i + h_{32}v_i + h_{33}} \tag{5}$$

$$y_i = \frac{y_i}{1} = \frac{h_{21}u_i + h_{22}v_i + h_{23}}{h_{31}u_i + h_{32}v_i + h_{33}} \tag{6}$$

We can further write Eq(5,6) as

$$u_i h_{11} + v_i h_{12} + h_{13} - x_i u_i h_{31} - x_i v_i h_{32} - x_i h_{33} = 0 \tag{7}$$

$$u_i h_{21} + v_i h_{22} + h_{23} - y_i u_i h_{31} - y_i v_i h_{32} - y_i h_{33} = 0 \tag{8}$$

So from a pair of corresponding points p^i and q^i we obtain two linear equations. Imagine that we perform the above procedure for all $i = 1, \ldots, N$, and put the $2N$ linear equations into matrix form, we can get

$$\begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -x_1v_1 & -x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -x_2u_2 & -x_2v_2 & -x_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_N & v_N & 1 & 0 & 0 & 0 & -x_Nu_N & -x_Nv_N & -x_N \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -y_1u_1 & -y_1v_1 & -y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -y_2u_2 & -y_2v_2 & -y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & u_N & v_N & 1 & -y_Nu_N & -y_Nv_N & -y_N \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \mathbf{0} \tag{9}$$

The expression for $\mathbf{A}$ is

$$\begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -x_1v_1 & -x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -x_2u_2 & -x_2v_2 & -x_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_N & v_N & 1 & 0 & 0 & 0 & -x_Nu_N & -x_Nv_N & -x_N \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -y_1u_1 & -y_1v_1 & -y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -y_2u_2 & -y_2v_2 & -y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & u_N & v_N & 1 & -y_Nu_N & -y_Nv_N & -y_N \end{bmatrix} \tag{10}$$

(b) From Eq(9), we can see that there are 9 elements in $\mathbf{h}$.

(c) 4 pairs (correspondences) are required to solve this system. The reason is that there are 8 degrees of freedom in $\mathbf{H}$ and each pair can provide two linear equations.

(d) In order to estimate elements in $\mathbf{h}$, we consider

$$\mathbf{h}^T\mathbf{A}^T\mathbf{A}\mathbf{h} = \mathbf{0} \tag{11}$$

Let $\mathbf{M} = \mathbf{A}^T\mathbf{A}$, we can see that $\mathbf{M}$ is symmetric positive semi-definite (its eigenvalues are non-negative). To estimate $\mathbf{h}$, we need to minimize $\mathbf{h}^T\mathbf{M}\mathbf{h}$ with constraint $\|\mathbf{h}\| = 1$ (Rayleigh Quotient theorem applied here). The minimum value of $\|\mathbf{h}\| = 1$ occurs when $\mathbf{h}$ is the eigenvector of $\mathbf{M}$ that corresponds to the smallest eigenvalue of $\mathbf{M}$. Then we can perform spectral decomposition to find the smallest eigenvalue and its corresponding eigenvector, which estimates $\mathbf{h}$.

4 Planar Homographies: Implementation (10 pts)

Q 4.1 (10pts): Implemented in `computeH.m`.

5 Stitching it together: Panoramas (30 pts)

Q 5.1 (15pts): Implemented in `imageStitching.m` and `testImageStitching.m`. The warped image is saved as `results/q5_1.jpg`. `H2to1` is saved in `results/q5_1.mat`. The warped image is shown in Figure 4.

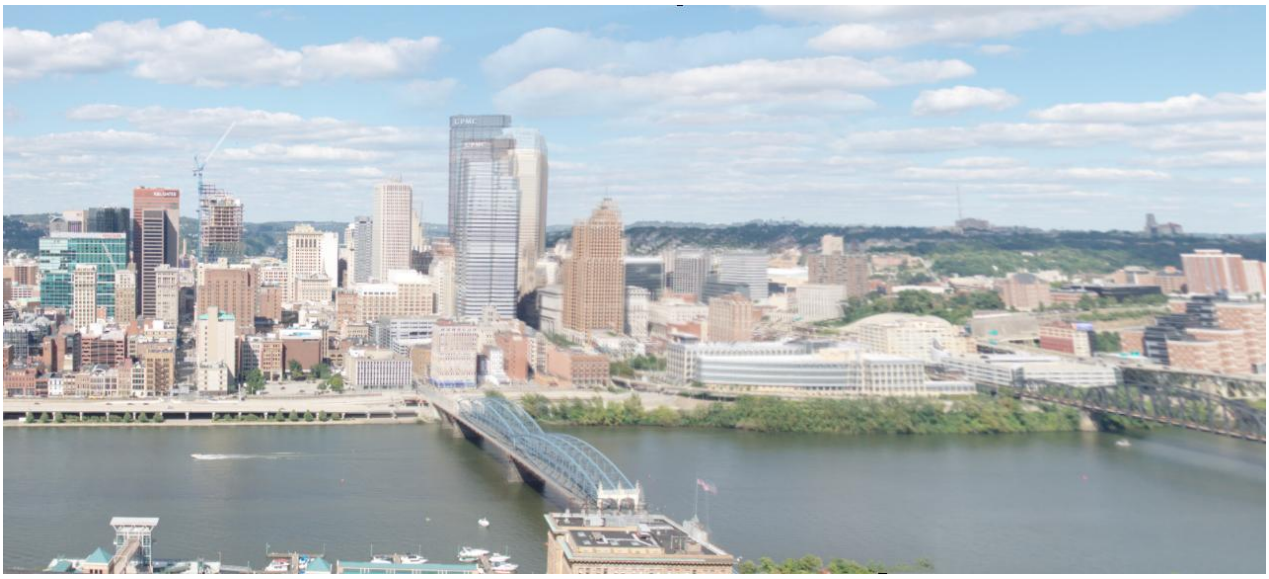

Figure 4: Clipped panorama

Q 5.2 (15pts): Implemented in `imageStitching_noClip.m` and `testImageStitching_noClip.m`. The non-clipped panorama (saved in `../results/q5_2_pan.jpg`) is shown in Figure 5.

6 RANSAC (25 pts)

Q 6.1 (15pts): Implemented in `ransacH.m`.

Q 6.2 (10pts): Implemented in `generatePanorama.m` and `testGeneratePanorama.m`. The final panorama is saved in `results/q6_2.jpg` and is shown in Figure 6.

Figure 5: Non-clipped panorama

Figure 6: Panorama

7 Extra Credit (20 pts)

1. To deal with the non-rotational invariance of BRIEF descriptor, I do the following:
   - Choose a larger patch centered at a valid interested point.
   - Compute gradients and (further) orientations of each pixel in this larger patch.
   - Specify 36 different orientations, 10 degrees apart, from 0 to 359 degrees.
   - Round the orientations of each pixel and count the number of pixels falling into each orientation.
   - Choose the orientation with the largest number of pixels as the orientation of the larger patch. Rotate the patch by the degrees of this orientation.
   - take the patch of required size centered at the center of the rotated larger patch. Then compute BRIEF descriptor for this patch.

Matches vs rotation angles are plotted in bar graph 7. The implementation can be found in `computeBrief RotationInvariant.m`, `briefLiteRotationInvariant` and `ec_1.m`. Comparing Figure 7 and Figure 3, we can see that the number of matches are improved for different amount of degrees of rotation. Observable amount of improvement can be seen in the middle portion of the bar graph. This means that after using oriented patch in computing BRIEF descriptor, the number of matches increase even under rotation. However, this implementation does not give an entirely rotationally invariant descriptor, because the middle portion of bar graph is much lower than two ends, which implies more work to be done. As a visual comparison of how well this implementation can do, Figure 8 shows the matching for rotated book on the floor. Comparing it with Figure 14, we can see that more matchings are found between corresponding locations of the books.

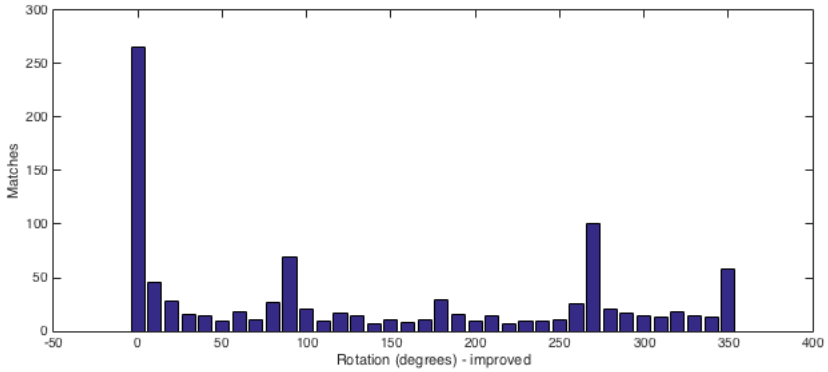

Figure 7: Bar graph for rotation angle vs number of correct matches after using rotation invariant technique

Figure 8: BRIEF matches for `pf_scan_scaled.jpg` and `pf_floor_rot.jpg`. after using orientation invariant technique

2. Figure 9 shows what happens when I match a picture to the same picture at half the size.

   In order to make the detector more robust to changes in scale, I do the following: before computing the descriptor, randomly choose a factor between 0.2 and 0.9 and resize the image according to this factor. Then compute the descriptor. The computation of the descriptor is the same as in extra credit problem 1 (i.e. use rotation invariant technique). The test is done on `model_chickenbroth.jpg` and the same image scaled to different sizes.

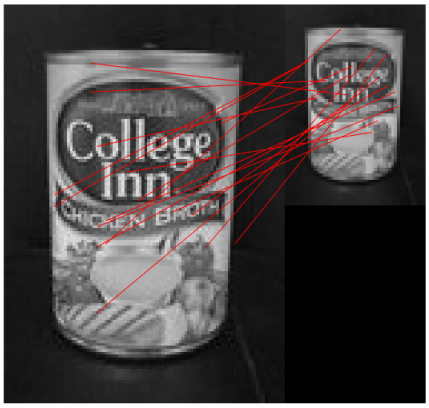

Figure 9: BRIEF matches for `model_chickenbroth.jpg` and itself at half the size

Figure 10 shows the bar graph of matches vs scale of an image with only rotation invariant technique. Figure 11 shows the bar graph of matches vs scale of an image with both rotation and scale invariant techniques. We can see that for different scales of the same image, scale invariant technique increases the relative number of matches in comparison to original image (though the absolute number of matches seems to decrease and at some scales, number of matches are zero - needs more sophisticated work here).

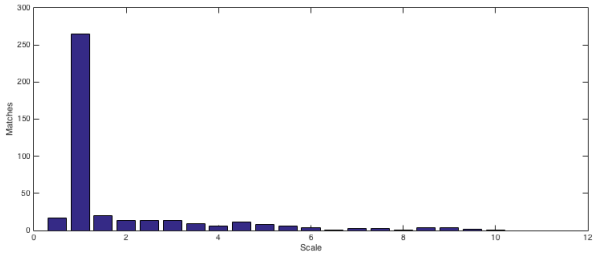

Figure 10: Bar graph for scales vs number of correct matches before using scale invariant technique

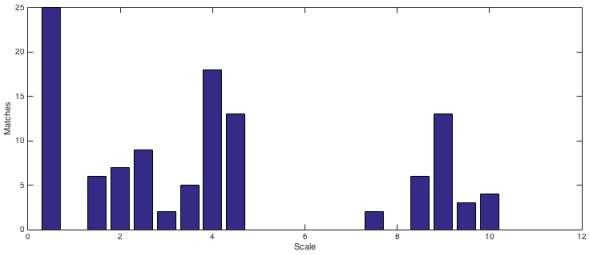

Figure 11: Bar graph for scales vs number of correct matches after using scale invariant technique

The implementation can be found in `computeBriefScaleInvariant.m`, `briefLiteScaleInvariant.m` and `ec_2.m`.

8 Appendix

A Results for Q2.4

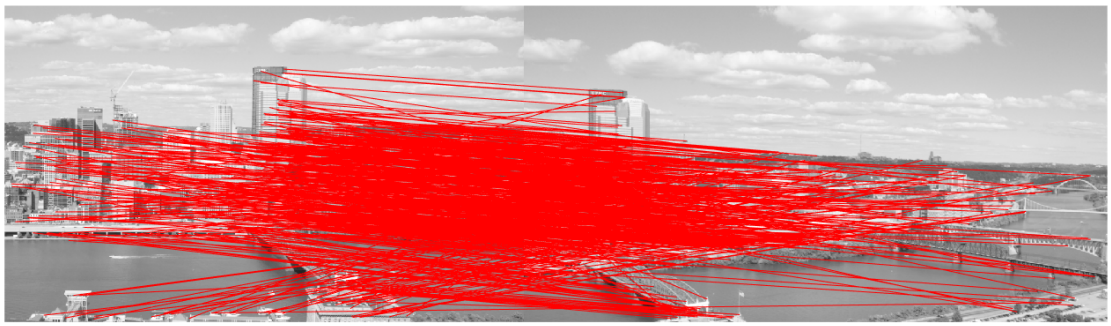

Figure 12: BRIEF matches for `incline_L.jpg` and `incline_R.jpg`.

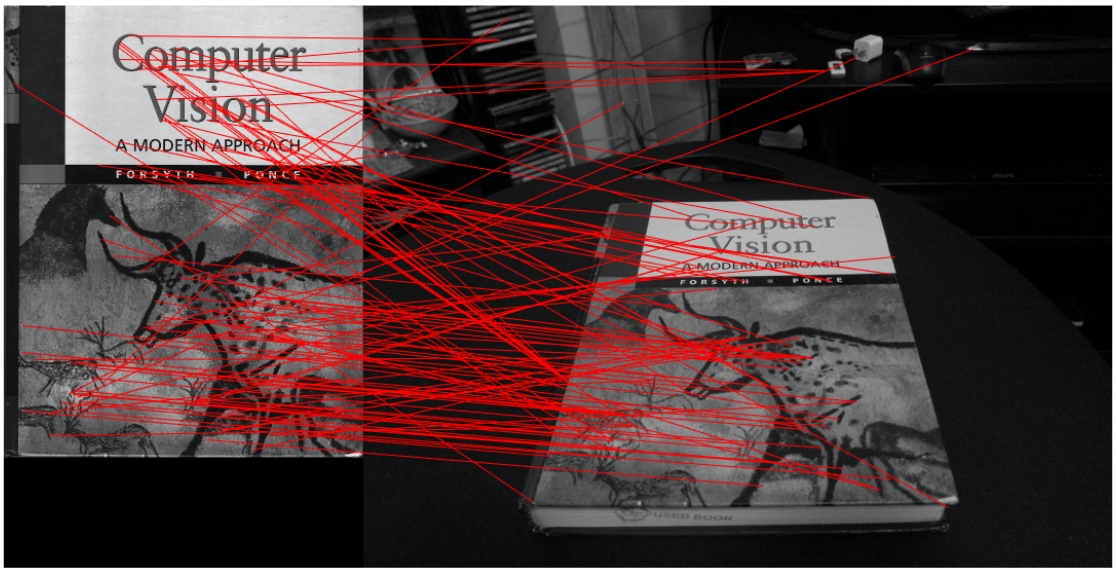

Figure 13: BRIEF matches for `pf_scan_scaled.jpg` and `pf_desk.jpg`.

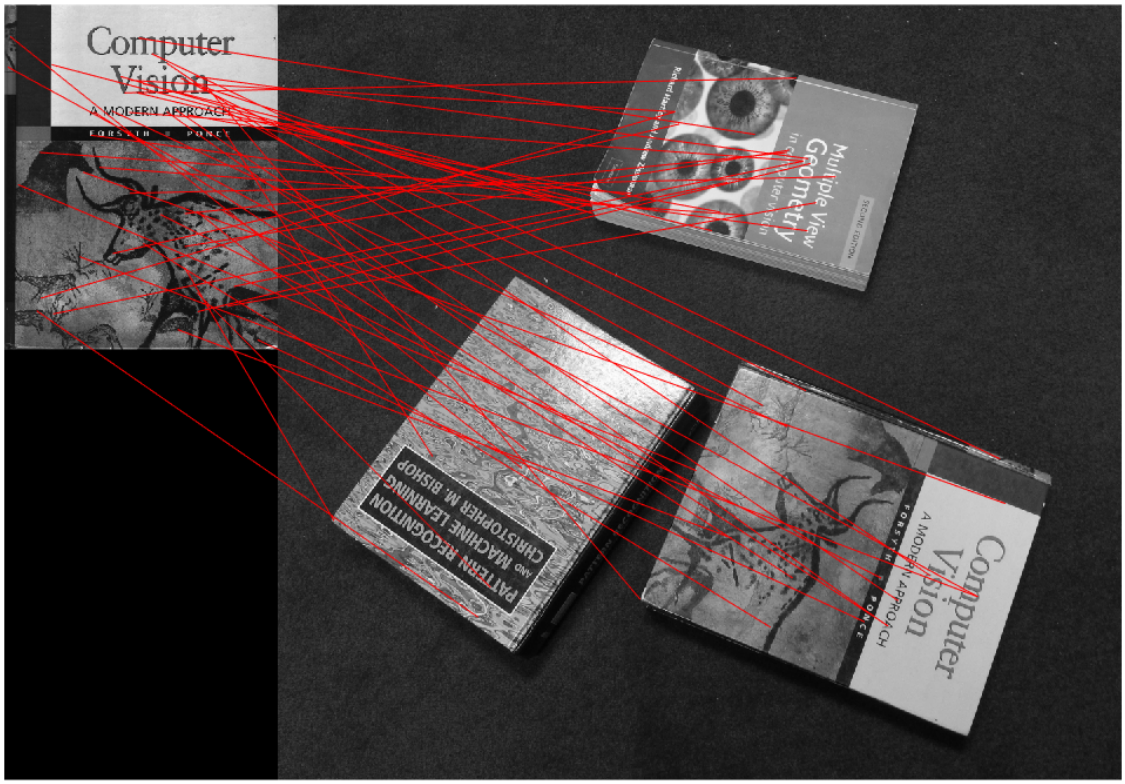

Figure 14: BRIEF matches for `pf_scan_scaled.jpg` and `pf_floor_rot.jpg`.

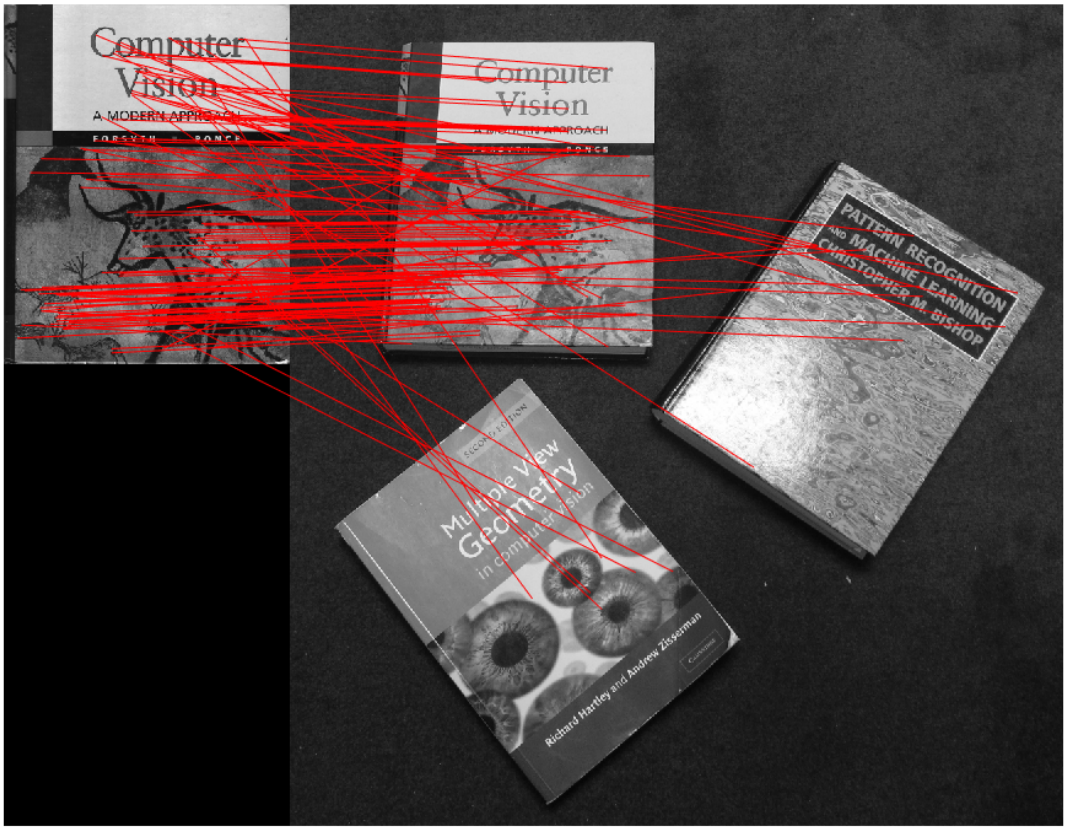

Figure 15: BRIEF matches for `pf_scan_scaled.jpg` and `pf_floor.jpg`.

Figure 16: BRIEF matches for `pf_scan_scaled.jpg` and `pf_pile.jpg`.

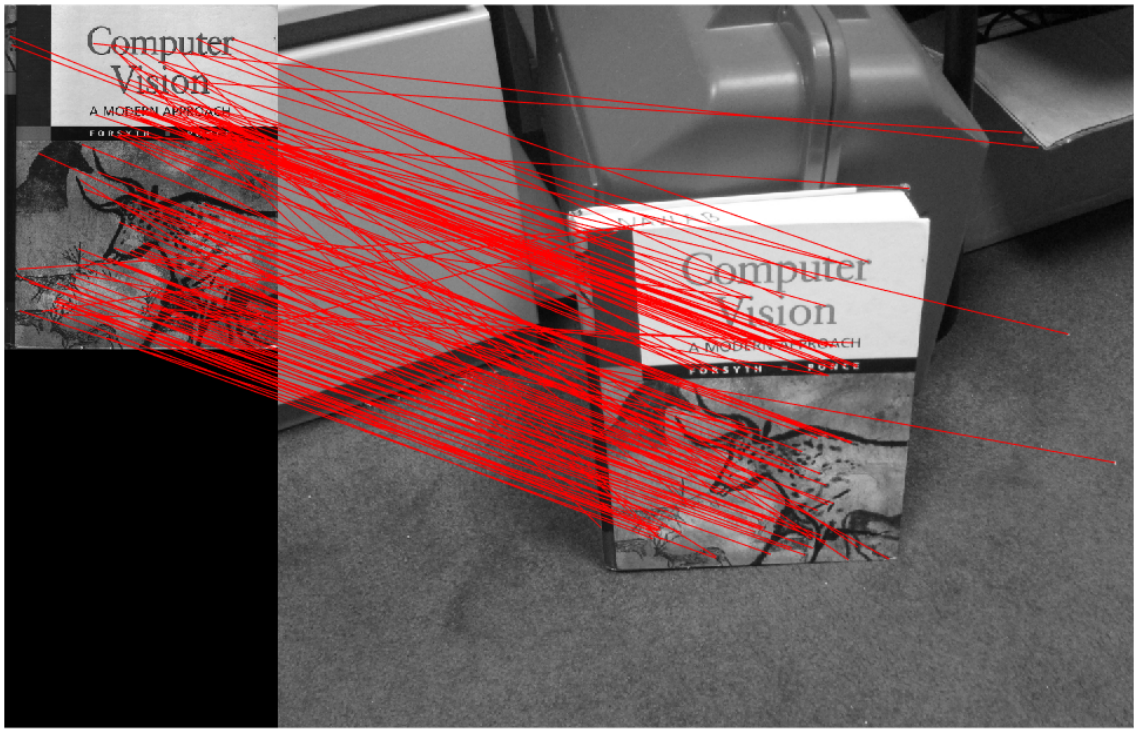

Figure 17: BRIEF matches for `pf_scan_scaled.jpg` and `pf_stand.jpg`.