Yue Pan
yuepan@student.ethz.ch

Assignment 2

Computer Vision
October 10, 2019

Task 1: Detection

For this task, please refer to *extractHarris.m*.

Firstly, the image gradients are calculated. The $\begin{bmatrix}-0.5 & 0 & 0.5\end{bmatrix}^T$ and $\begin{bmatrix}-0.5 & 0 & 0.5\end{bmatrix}$ convolution kernels are used respectively for calculating the x and y gradient of the image, namely I_x and I_y. The *conv* function is used to do the convolution with the parameter *'same'* so that the image size would not change after the convolution.

Secondly, the local auto-correlation matrices are constructed. I_x^2, I_y^2 and I_xI_y are calculated from I_x and I_y. Then I apply the gaussian filter using *imgaussfilt* function on I_x^2, I_y^2 and I_xI_y respectively.

Then, for each pixel, the Harris response $R = \det(M) - kTr^2(M)$ is calculated from its auto-correlation matrix. Then the Harris corner points are extracted under strength threshold and local-maximum conditions.

There are three key parameters in the process: the Harris response threshold T, the constant k in Harris response function and the By varying the threshold T, the standard deviation σ when calculating the auto-correlation matrix. By varying these parameters , corner points are extracted using Harris operator for both the *block* and *house* images, as shown in Fig.1 and Fig.2.

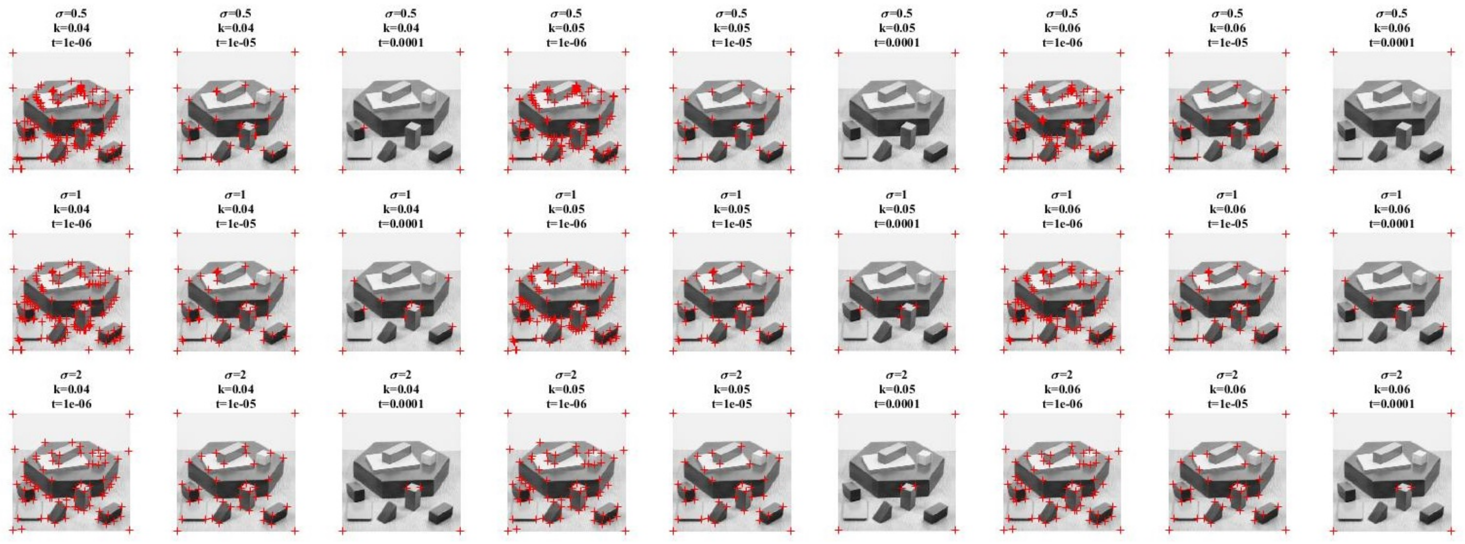

Figure 1: Corner detection result for 'block' image under different parameter settings

From Fig.1 and 2, we can draw the following conclusion:

1. The larger t is, the less corner points would be detected because more pixels would be rejected by the threshold. But when t is too small, there would be a lot of false positive detection.

2. The larger k is, the less corner points would be detected because the Harris response would be smaller.

3. When $sigma = 1$, there are some adjacent false positive detection. More corner points would be detected when $sigma = 2$ than $sigma = 0.5$.

Finally, the parameters are set as Table 1 because relatively more true positive corner points would be detected

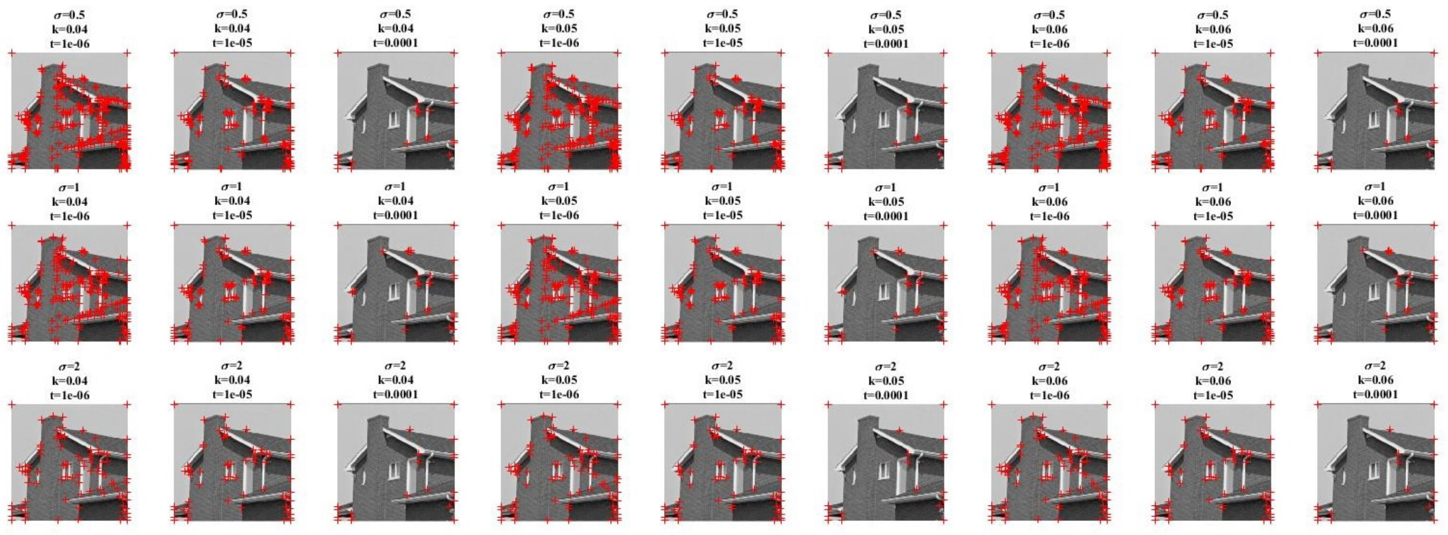

Figure 2: Corner detection result for 'house' image under different parameter settings

under certain circumstance. The parameter settings are adopted in the following tasks.

Table 1: Parameter settings

parameter	value
σ	2.0
k	0.04
threshold	1e-05

Task 2: Description & Matching

For this task, please refer to *extractPatches.m* , *extractDescriptors.m* and *matchDescriptors.m*.

Corner points extraction result for the image pair is shown in Fig.3. The detected corner keypoints are then filtered. Those false positive corners locating at the image boundary are filtered. The boundary size is set as $9/2-1=4$ pixels. The filtered result is shown in Fig.4. After that, the 9×9 neighborhood patch is extracted for each corner pixel. The corner pixel's descriptor vector is assigned as the intensity value of each neighbor pixel in the patch. Then the $m\times n$ sum-of-squared-differences (**SSD**) matrix between m corner points in image 1 and n corner points in image 2 can be determined from these corner points' description vectors.

As for the keypoints matching task, several matching strategies are adopted in this task.

One-way Nearest Neighbor Matcing:

The one-way nearest neighbor matching result is shown in Fig.5. There are 152 pairs of correspondence points,

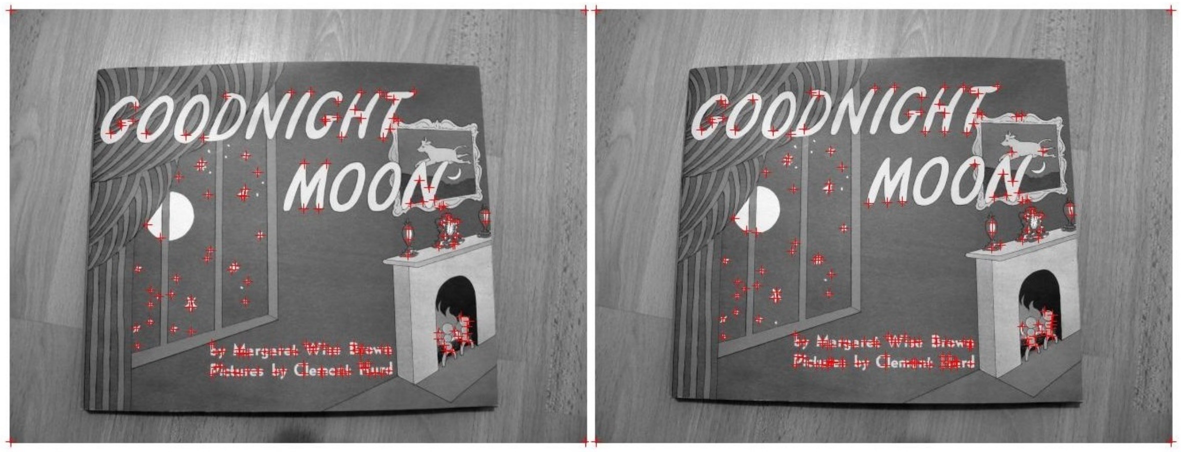

Figure 3: 156 corner keypoints extracted from image 1 and 172 corner keypoints extracted from image 2.

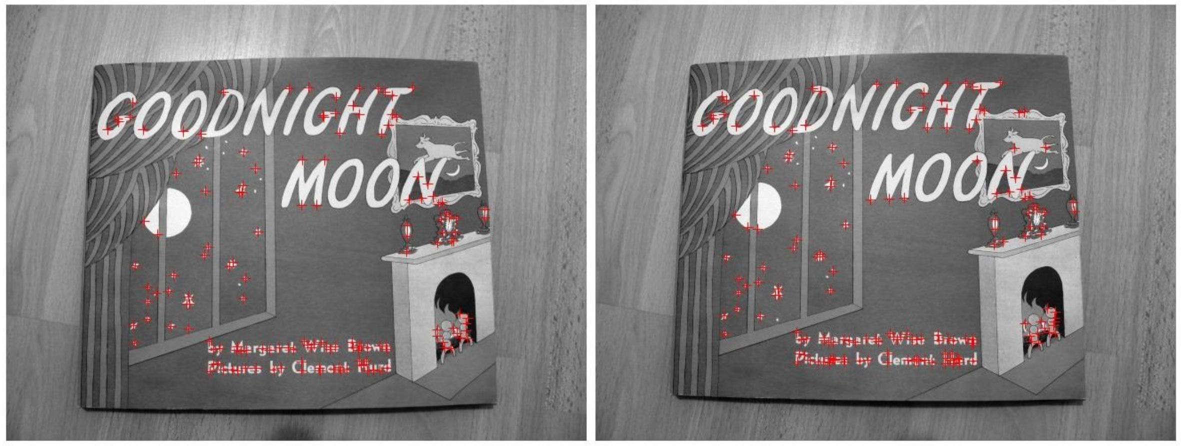

Figure 4: 152 corner keypoints left in image 1 and 168 corner keypoints left in image 2 after boundary corner filtering.

exactly the number of filtered corner points in the first image. It's clear that some of the matching pairs are wrong. The implement of this matching strategy is as following: For each row in the **SSD** matrix, the minimum value and its column index is determined using *[dis,index]=min(SSDMatrix,[ ],2)* function. Then the row index and the column index corresponding to the minimum value of the row make up a matching pair.

Mutual Nearest Neighbor Matcing:

The mutual nearest neighbor matching result is shown in Fig.6. There are 106 matching pairs. 46 pairs are rejected from the 152 pairs of one-way matching because the corner point in image 1 is not the nearest neighbor of the corresponding point in image 2. It can be noticed that there are still some false positive match. The implement of this matching strategy is as following: For each column i of the **SSD** matrix, the minimum value and the corresponding row index j can be drawn. Only if the minimum column index of row

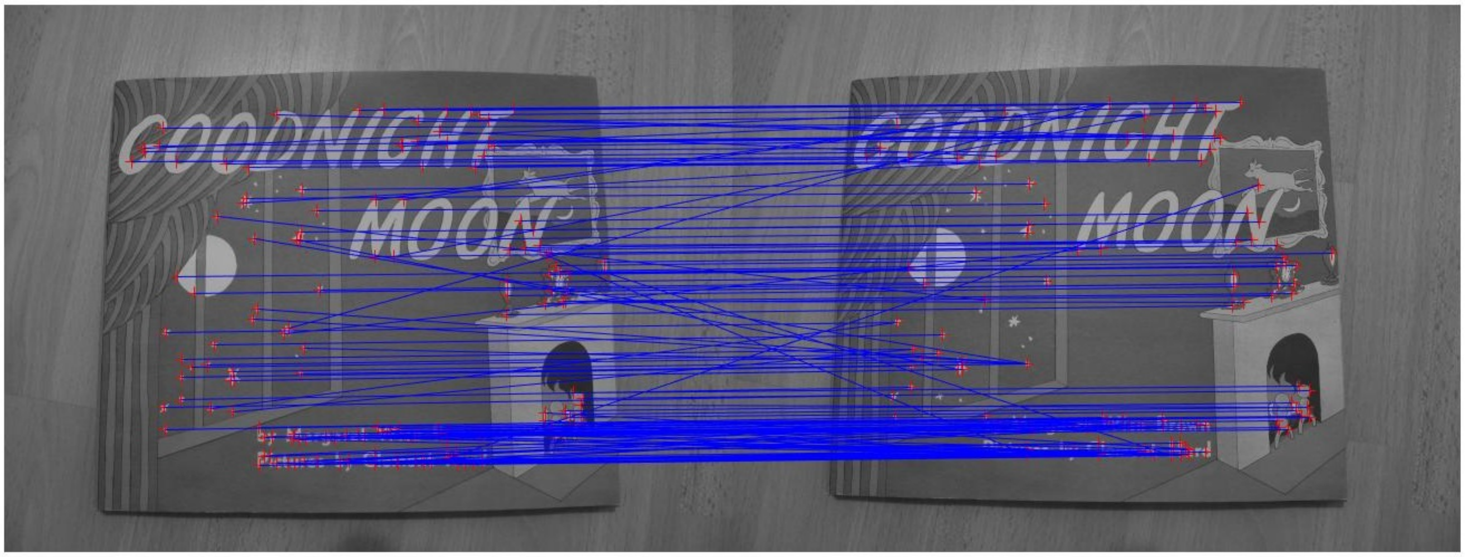

Figure 5: One-way Nearest Neighbor Match: 156 matching pairs in total.

j is equal to i, i and j would be regarded as a pair of matching corner points.

When swapping image 1 and 2, the correspondences found by mutual nearest neighbor matching keeps the same, as shown in Fig.7.

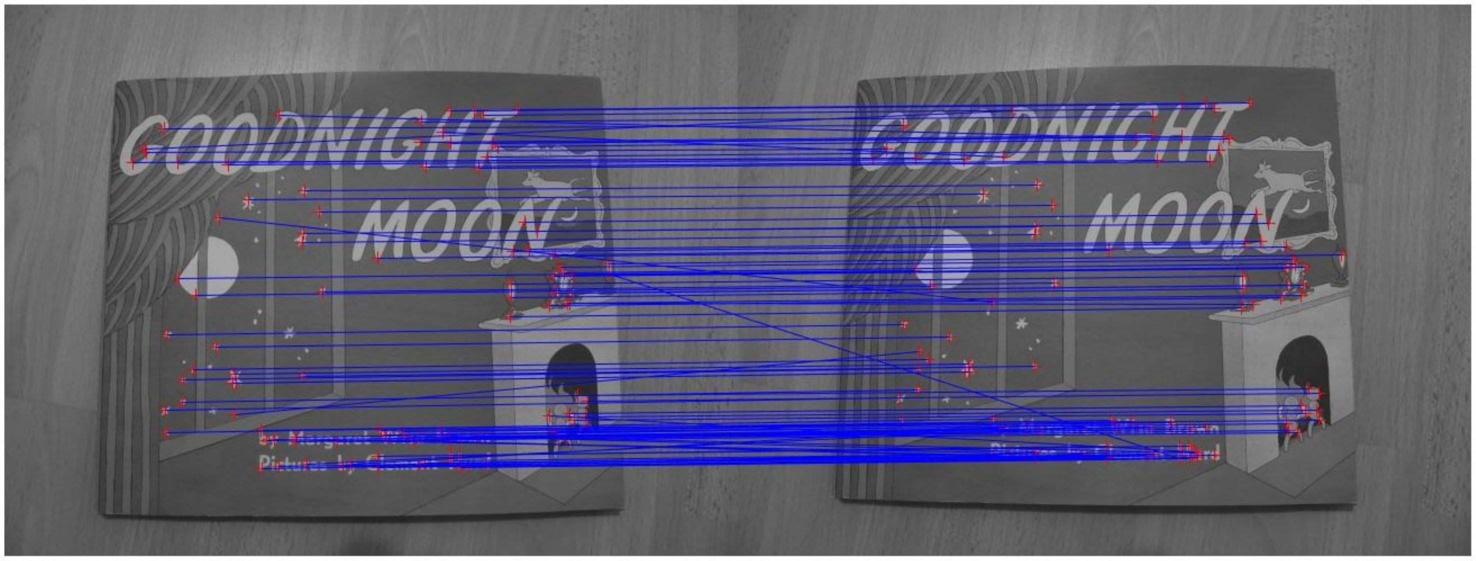

Figure 6: Mutual Nearest Neighbor Match: 106 match pairs in total.

Nearest Neighbor and Second Nearest Neighbor Ratio Test Matching:

For the FNN/SNN ratio test, for each row in the **SSD** matrix, both the first and the second nearest neighbor

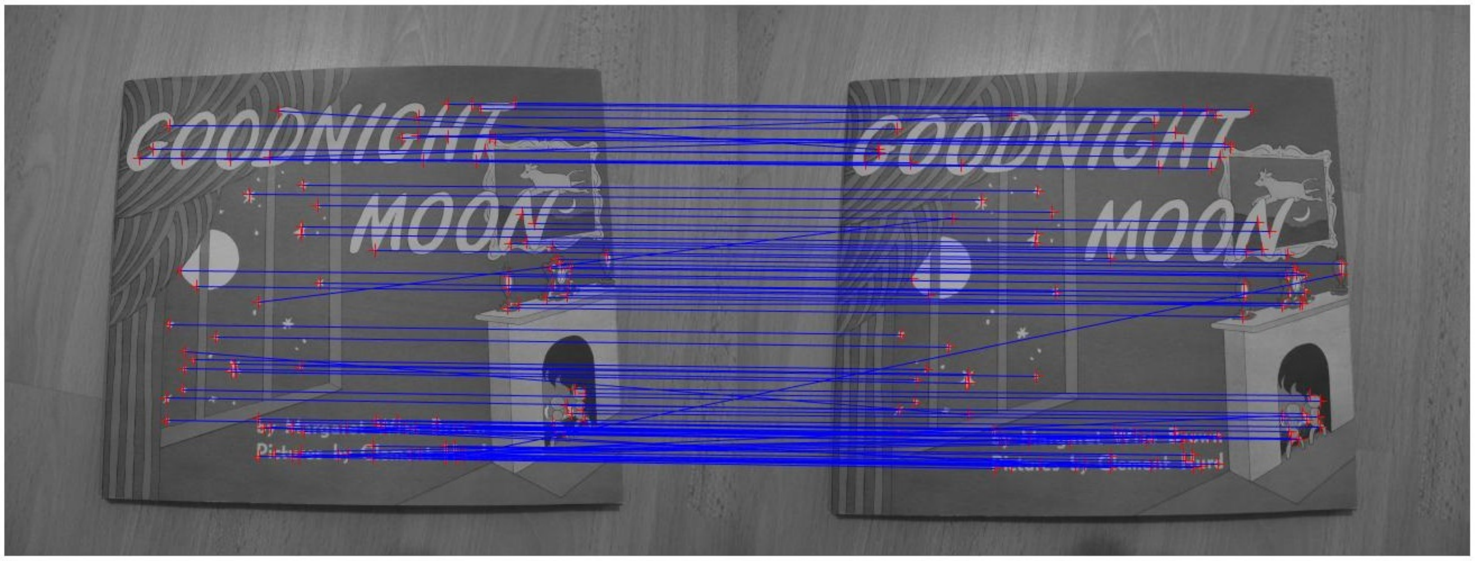

Figure 7: Mutual Nearest Neighbor Match (swap the images): 106 match pairs in total.

point are required to be found. Since my Matlab's version is 2016a, there's no *mink* function for it. The workaround is to assign the first nearest neighbor a very big value (for example, *realmax* in Matlab) and then find the nearesst neighbor for the target row again to get the second nearest neighbor.

I test three ratio threshold values (0.2, 0.5 and 0.8) here. Those one-way matching pairs whose FNN/SNN distance ratio is larger than this threshold would be rejected. The result is shown respectively in Fig.8-10. The smaller the threshold is, the less matching pairs would be kept while the precision increases.

The number of correspondences for different keypoints matching strategies are list in Table 2.

To sum up, the correspondence grouping strategy should consider the balance of precision and recall (calculated as Equ.1) of the correspondences. Generally speaking, precision and recall are a pair of trade-off. The feasible way is to label the ground truth matching and then draw the Precision-Recall curve. Those matching strategy resulting in a P-R curve on the most right-upper side would be regarded as the best method.

$$\begin{cases} \text{precision} = \frac{\text{true positive}}{\text{true positive+false positive}} \\ \text{recall} = \frac{\text{true positive}}{\text{true positive+false negative}} \end{cases} \tag{1}$$

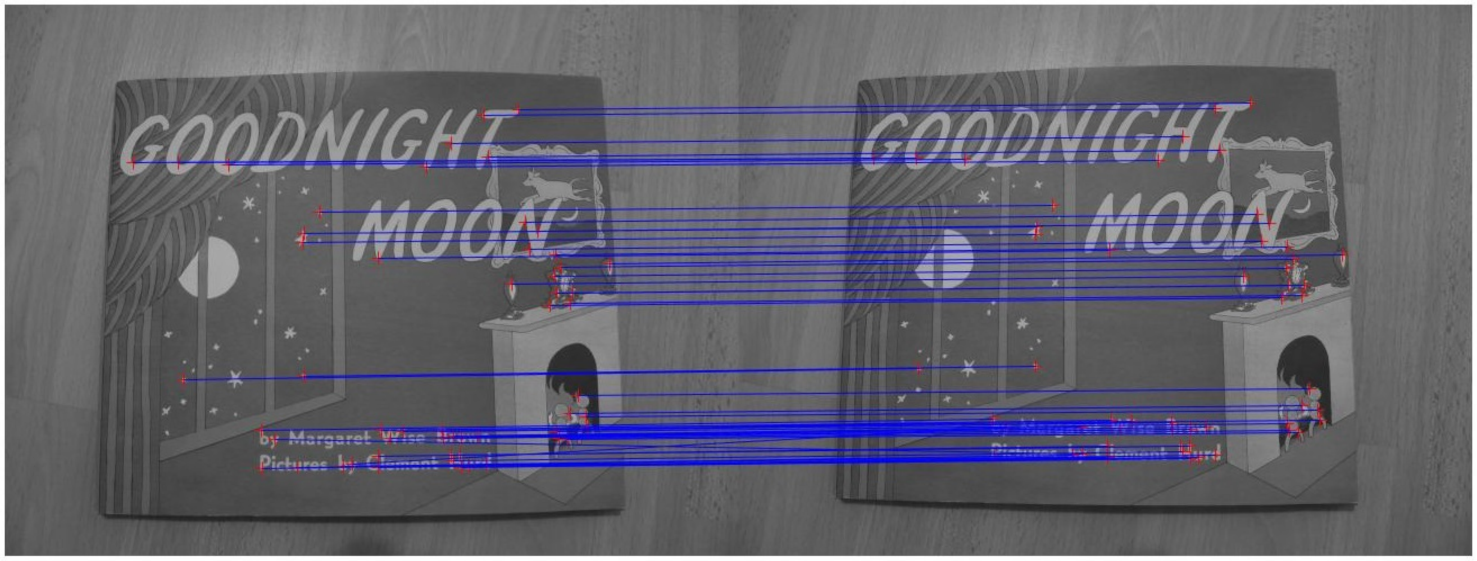

Figure 8: Matching result when the first nearest neighbor and second nearest neighbor distance ratio set to be 0.2 : 55 match pairs in total.

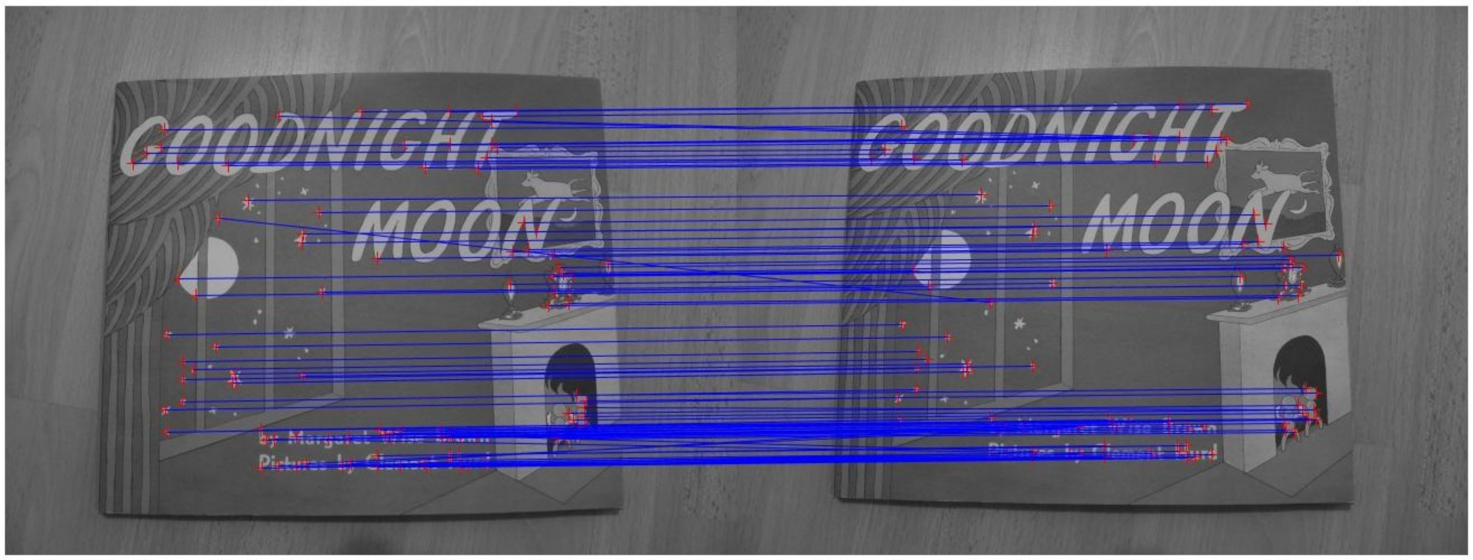

Figure 9: Matching result when the first nearest neighbor and second nearest neighbor distance ratio set to be 0.5 : 100 match pairs in total.

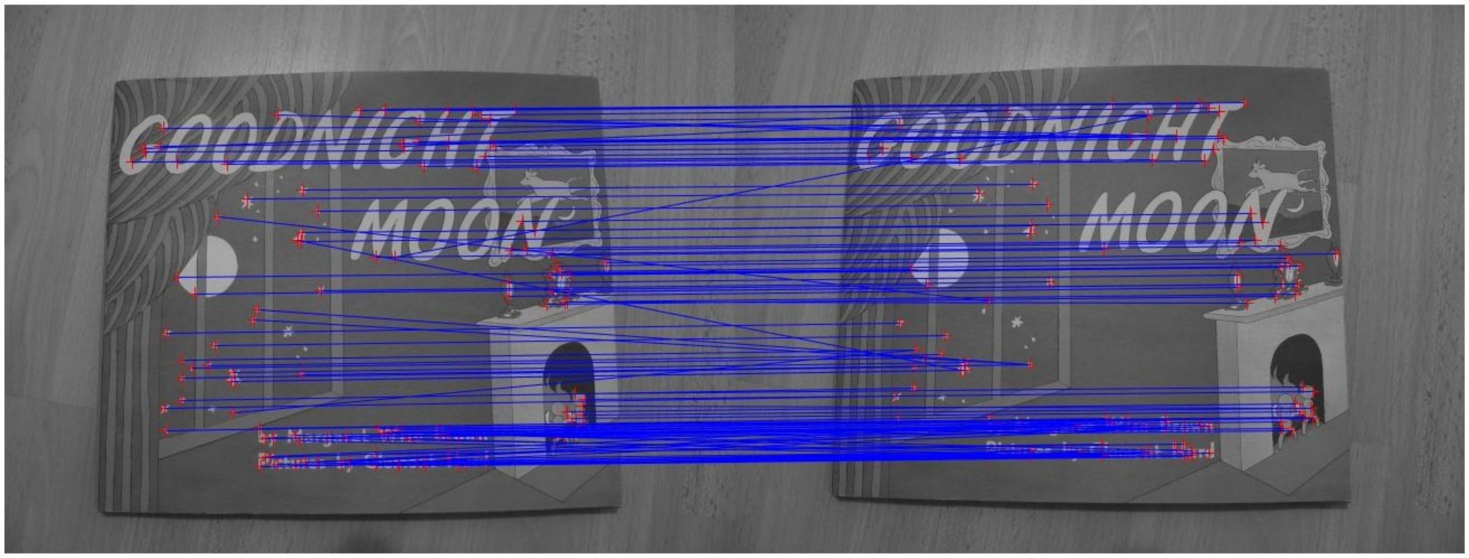

Figure 10: Matching result when the first nearest neighbor and second nearest neighbor distance ratio set to be 0.8 : 125 match pairs in total.

Table 2: Number of matching pairs for different matching strategies

matching strategy	# pairs
one way	156
mutual	106
FNN/SNN ratio = 0.5	100
FNN/SNN ratio = 0.2	55
FNN/SNN ratio = 0.8	125