lab02-local-features

Joanna Erika Brodbeck, Computer Science, Bsc 7. Semester
`jbrodbec@student.ethz.ch`

Abstract

Here is the report for the lab02-local-features.

For this exercise, Copilot was used to help with the syntax. The written report was checked with Grammarly.

1 Harris Corner Detection

First, the Harris Corner Detection was implemented. This was done in several steps: gradient calculation, blurring, weighted sum, response function and thresholding.

Gradient Calculation The formula to get the gradient is:

$$I_x = \frac{I(x+1, y) - I(x-1, y)}{2} \qquad I_y = \frac{I(x, y+1) - I(x, y-1)}{2}$$

To calculate it faster, the Sobel filter was used to afterwards convolve it to immediately get the gradients www.stackoverflow.com/image-processing-implementing-sobel-filter.

$$\text{Sobel filter for } I_x : \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad \text{Sobel filter for } I_y : \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

After the convolution, the "blocks.jpg" image looks like this:

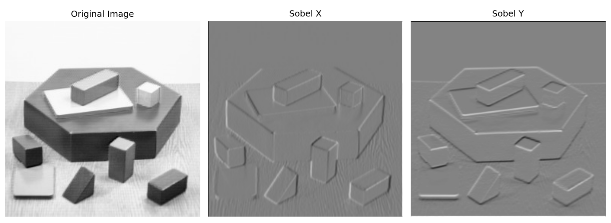

Figure 1: Gradient image of the blocks.jpg image

Blurring The previously convolved images were blurred to get more robust results. This was done with the proposed solution with GaussianBlur where both images, I_x and I_y were blurred, a kernel size of (5, 5) was used, and sigma was set as 2.0.

Weighted Sum After creating the matrix M it was blurred again to get the weighted sum. This was done with the Gaussian blur.

Response Function The determinant of the matrix M and the trace was calculated, and with the formula given for the response, C was calculated. $R = \det(M) - k\,\text{trace}^2(M)$

Thresholding First, all the candidates of a corner were the elements of C bigger than the given threshold. Then, for each $3x3$ window, the maximum of these candidates was calculated. Only the maximum of each window was then used as corners.

Response Function Each key point was checked to ensure it was not too close to the border. This was done by checking if the coordinates were as close as half the patch size to a border. If this was the case, this key point was filtered out.

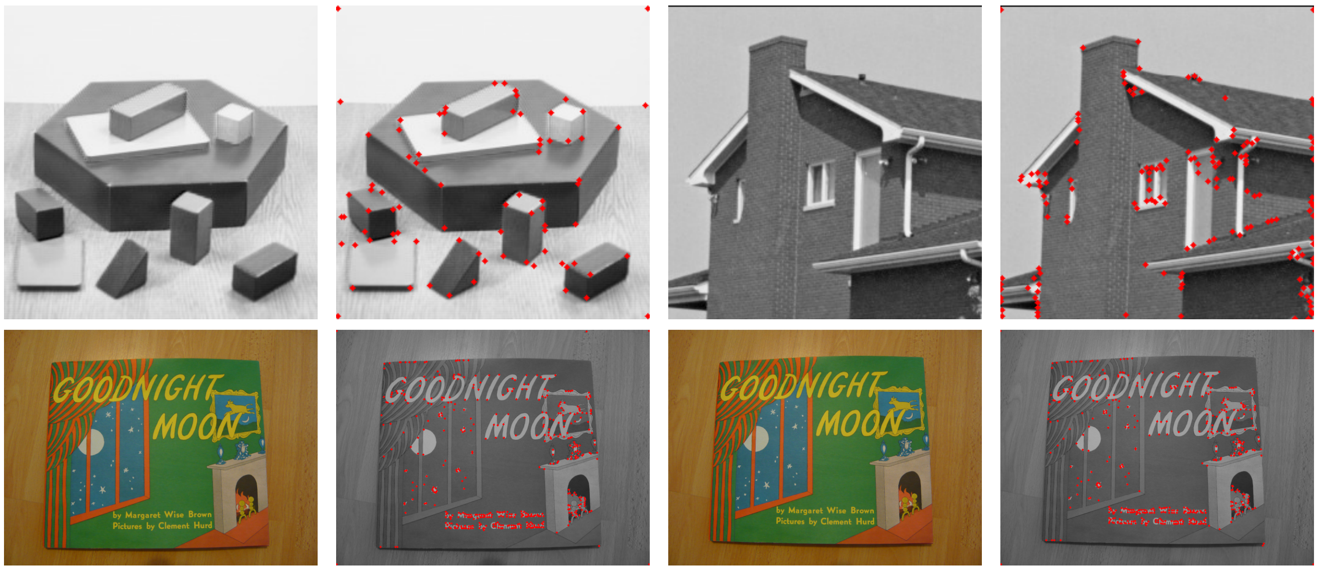

Figure 2: Results of Harris Corner Detection

2 Match Descriptors

SSD The SSD was calculated by extracting the formula from $(p-q)^2$ to $(p^2-2pq+q^2)$. So the library *Numpy* could be easily used to get the sum of the formula.

One Way The nearest_neighbors_indices was calculated with `np.argmin(distances, axis=1)`, which gave each minimum of each line of the matrix. The matches were then stacked such that the indices of the patches were left, and the indices of the closest neighbour were right.
It can be seen that some matches are wrong for the one-way approach. All the matches should appear as a straight line from left to right, but several matches are still crossed. This means that a corner detected on the very bottom left was matched with something on the very top right, which is obviously wrong. A total of 464 matches were identified.

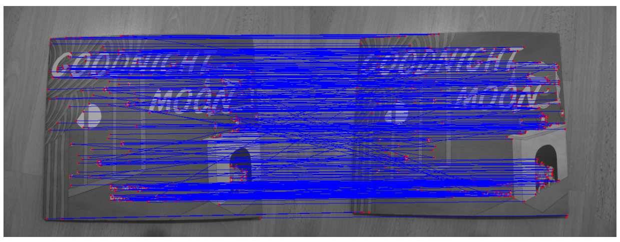

Figure 3: Match one way

Mutual Here again, the nearest neighbour indices were calculated in the direction from image 1 to image 2 and vice versa. Then, the stack option was again used, but only matches were taken where they had the same indices in both directions.
This already gives better results than the last one with no cross matches. This guarantees that each match is mutual and that the smallest distance leads to fewer false positive matches. It gave fewer but more reliable matches. A total of 352 matches were identified.

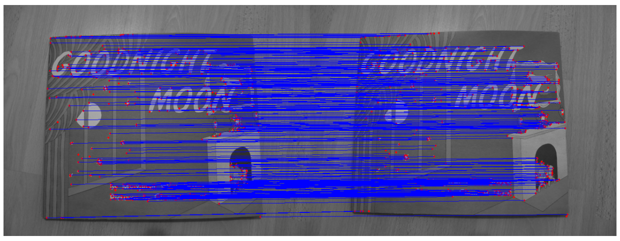

Figure 4: Match Mutual

Ratio The ratio is calculated using both the nearest neighbour and the second nearest neighbour. Then, their ratio is taken. If this ratio is below a given threshold, it is a "good" match; otherwise, it is not. Then, an array is created where only the neighbours are taken, and the ratio fulfils the requirement.
The Ratio approach gives better results than the one-way approach, as it removes matches that are ambiguous and not really sure if they are a match. Depending on the threshold, many or only a handful of the one-way matches are sorted out. With parameter threshold 0.5, 262 matches were identified. So, this approach gave the smallest number of matches but also missed some prominent corners to match. Increasing the threshold leads to more false positive matches.

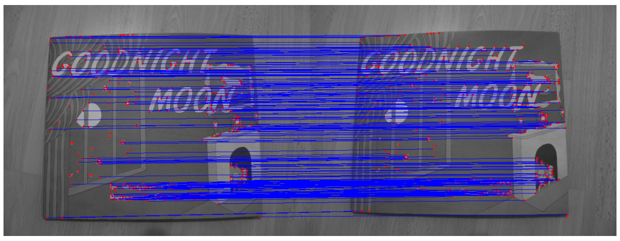

Figure 5: Match Ratio

3 Parameters

```
HARRIS_SIGMA = 2.0
HARRIS_K = 0.04
HARRIS_THRESH = 1E-4
MATCHING_RATIO_TET_THRESHOLD = 0.5
```

It was hard to tweak the parameters correctly because it was a balance between choosing noise or not enough points. In the end, the setting with the slightest noise was selected.