EQ2425 Analysis and Search of Visual Data

EQ2425, Project 1

Hanqi Yang hanqi@kth.se

Qian Zhou qianzho@kth.se

September 18, 2022

Summary

In this project, our first goal is to analyze two different algorithms for extracting feature keypoints. These two are SIFT (Scale Invariant Feature Transform) and SURF (Speeded-Up Robust Features). We test the detectors' robustness against some simple transformations such as rotation and scaling. We find that SIFT works better on average.

Our second goal is to implement image feature matching between two images using three different algorithms: "fixed threshold" , "nearest neighbor" and "nearest neighbor distance ratio" matching algorithm. After finding "nearest neighbor distance ratio" performs the best, we use it to compare the performance of SIFT and SURF, which again proves more accurate matching in SIFT than in SURF. Therefore, under the conditions we set, SIFT works better on average than SURF although SURF uses fewer resources and less computational time.

1 Introduction

This project mainly includes two parts, one is to implement the SIFT and SURF algorithms[1] and test the robustness of the two keypoint detectors against rotation and scale changes. The two algorithms can extract feature keypoints from images, that describes the characteristics of the visualized scenario. The other is to implement three feature matching algorithms, "fixed threshold", "nearest neighbor", and "nearest neighbor distance ratio". For each algorithm, based on the extracted SIFT features, we plot side-by-side views of the query image obj1_5.JPG and the database image obj1_t5.JPG with matched feature points connected by lines. And we visually examine the matched features and comment on the performance of the three matching algorithms. In addition, we compare SIFT and SURF's performances when using "nearest neighbor distance ratio" algorithm.

In this project, we use the open source library VLFeat for SIFT feature extraction[2], and the Matlab function "detectSURFFeatures" for SURF feature extraction.

2 Problem Description

The project can be divided by two parts: Robustness of Keypoint Detector and Image Feature Matching.

2.1 Robustness of Keypoint Detector

To test the robustness of keypoint detectors and descriptors, first, we need to apply SIFT and SURF keypoint detectors on the image "obj1_5.JPG". To visualize only a few hundred features, we need to choose appropriate peak threshold and edge threshold of the SIFT keypoint detector so that we can visually examine the performances. First, we choose peak and edge threshold both as 10, and we find that there exist 996 keypoints, which should be reduced. Since the peak threshold removes the part of the extreme point in the DoG scale space that is too small in absolute value, the number of keypoints obtained decreases when the value of peak threshold increases. Similarly, for the edge threshold, its function is to remove the extreme points of the DoG scale space where the curvature is too small (such extreme points are difficult to find keyframes), so more features will be obtained when edge threshold increases. Therefore, we change peak threshold to 13 and edge threshold to 5, then we obtain 359 keypoints. For SURF detector, when the value of strongest feature threshold decrease, more blobs will be returned. Hence, we select 6000 as the threshold and 445 keypoints are obtained.

Then, two ways are implemented to modify the original image and we evaluate the "repeatability" measure to determine which one performs better. The repeatability measure is defined as the number of matching features between the original image and the modified image, divided by the number of detected features in the original image. First, we plot repeatability versus rotation angle in increments of 15 degrees, from 0 degrees to 360 degrees for the two keypoint detectors. To compute repeatability, we find the keypoints $[x_0, y_0]$ of the rotated image and the rotated keypoints $[x', y']$ of the original image. We compare them to check how many of them are matched, which satisfies $|x_0 - x'| \leq 2$ and $|y_0 - y'| \leq 2$. Then, we comment on the robustness of the two keypoint detectors against rotation. Second, we plot repeatability versus scaling factor with the scaling factors $(m^0, m^1, m^2, ..., m^8, where\ m = 1.2)$ for the two keypoint detectors. Likewise, we comment on the robustness of the two keypoint detectors against scale changes.

2.2 Image Feature Matching

In this part, we compare two images representing the same scenario from different perspectives by implementing three feature matching algorithms, "fixed threshold", "nearest neighbor", and "nearest neighbor distance ratio". Then, we comment on the performance of the three matching algorithms. Before using the matching algorithms, we extract a few hundred SIFT features from the query image obj1_5.JPG and the database image obj1_t5.JPG. When implementing the "fixed threshold" matching algorithm, we calculate the distances between the two images feature points and compare them with an appropriate distance threshold. When the distance between two points is less than the threshold, we can think of it as a match. When implementing the "nearest neighbor" matching algorithm, for the feature point of the original image, we find the feature point with the smallest distance in the target image to match. When implementing the "nearest neighbor distance ratio" matching algorithm, for the feature point of the original image, we find the point with the smallest distance and the point with the second smallest distance in the target image. Then, we divide the smallest distance by the second smallest distance and compare it with an appropriate threshold. When the ratio is less than the threshold, we can think of it as a match. Finally, we extract a few hundred SURF features from the two images and use the "nearest neighbor distance ratio" matching algorithm, comparing the result with the SIFT implementation. To show the

result for each algorithm, we plot side-by-side views of the two images with matched feature points connected by lines.

3 Results

3.1 Robustness of Keypoint Detector

(a) In this part, we plot images showing the detected SIFT keypoints and SURF keypoints, superimposed on the original image for each detector separately. As mentioned in 2.1, for the SIFT algorithm the peak threshold and edge threshold are set to 13 and 5 respectively. As shown in Figure 1(a), we can find 359 key points, ensuring visually examining the comparison. Similarly for the SURF algorithm, we select strongest feature threshold as 6000 and 445 keypoints are generated, which is shown in Figure 1(b). Comparing the two images, we can see the roof, the windows on the left side and the KTH logo seem to generate large numbers of SIFT and SURF keypoints, which further demonstrates how the detections are sensible on the edges.

(a) SIFT feature

(b) SURF feature

Figure 1: Detected SIFT and SURF keypoints

(b) To test the robustness of the two keypoint detectors against rotation, we plot repeatability versus rotation angle in increments of 15 degrees, from 0 degrees to 360 degrees for the two keypoint detectors, as shown in Figure 2. As the picture rotates, every 90 degrees is roughly a cycle, with a valley in the middle of each cycle and a peak at both ends. The performance of the valley in SIFT is better than that of SURF, while the peaks perform better in SURF, with a repeatability of 1 at angles 90°, 180°, 270°. In general, SIFT algorithm performs better overall angles.
(c) To test the robustness of the two keypoint detectors against scale changes, we plot repeatability versus scaling factor with the scaling factors for the two keypoint detectors, as shown in Figure 3. For both algorithms, the repeatability shows a decreasing trend as the scale factor increases. For SIFT, we observe the repeatability decreases linearly when the scaling factor is larger than 2, with a maximum degradation of 50% when the scaling factor is 4.3. For SURF, however, we observe a behavior more sensible to scaling, with a maximum degradation of 75% when the scaling factor is 4.3. In general, SIFT algorithm has better performance when the scaling factor is changing.

3.2 Image Feature Matching

(a) We extract a few hundred SIFT features from the query image obj1_5.JPG and the database image obj1_t5.JPG and show the feature keypoints superim-

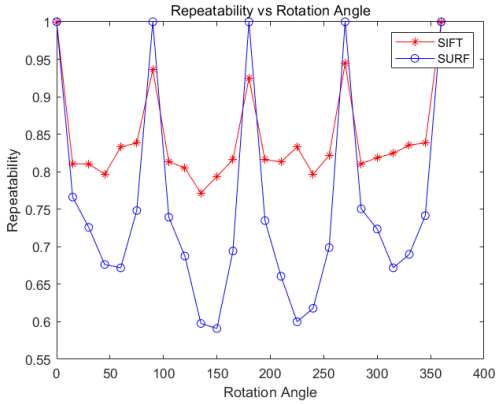

Figure 2: Repeatability versus Rotation Angle

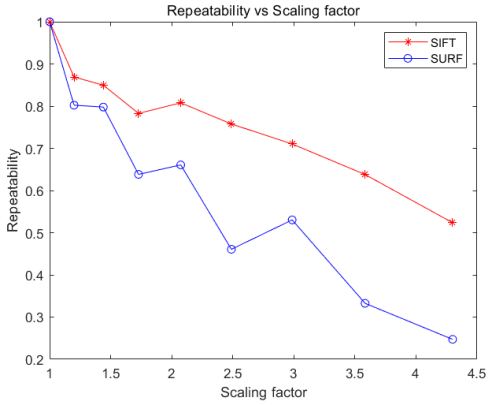

Figure 3: Repeatability versus Scaling Factor

posed on top of the two images. As shown in Figure 4, we get 359 and 285 keypoints respectively. For the following parts, we will apply different matching algorithms based on these SIFT feature keypoints.

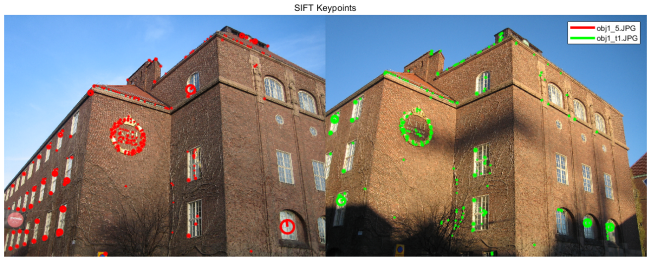

Figure 4: SIFT features of query image and database image

(b) We implement the "fixed threshold" matching algorithm. To obtain a satisfying matching result, we adjust the distance threshold to 55. As shown in Figure 5, we can observe there exist 15 matching lines with 5 mismatches. In our suboptimal result in Figure 6 where the distance threshold is set to 60, however, a lot of mismatches have been added with the number of matching connections basically unchanged. If we continue to increase the threshold, the larger thresh-

old will provide more matches, but probably include more mismatches with one-to-many relationships.

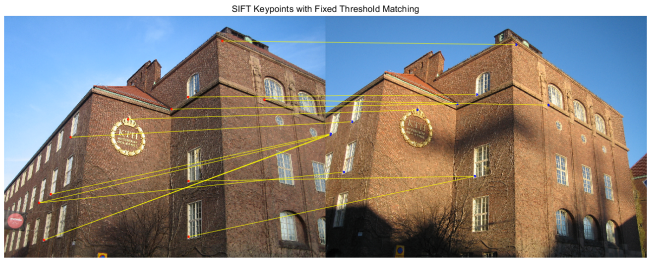

Figure 5: Fixed Threshold Matching Algorithm on SIFT with $threshold = 55$

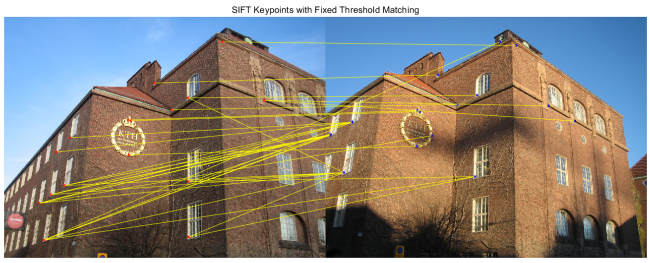

Figure 6: Fixed Threshold Matching Algorithm on SIFT with $threshold = 60$

(c) We implement the "nearest neighbor" matching algorithm and the result is shown in Figure 7. Comparing to the "fixed threshold" matching algorithm, it is obvious that the number of matches and mismatches is greatly increased. Since the "nearest neighbor" algorithm finds a match for each point of the original image, there will be a large number of mismatches, especially for those points that only appear once.

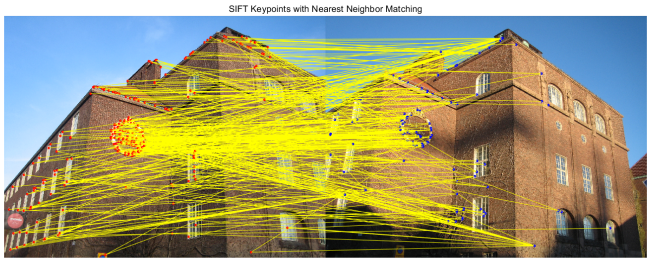

Figure 7: Nearest Neighbor Matching Algorithm on SIFT

(d) We implement the "nearest neighbor distance ratio" matching algorithm. To obtain a satisfying matching result, we adjust the ratio threshold to 0.78. As shown in Figure 8, we can observe there exist 20 matching lines without mismatches. In our suboptimal result in Figure 9 where the ratio threshold is set to 0.8, one mismatch appears within 27 matches. Therefore, it can be said that this algorithm is the best so far.
(e) We extract a few hundred SURF features from the two images and use the "nearest neighbor distance ratio" matching algorithm. Again, we set the

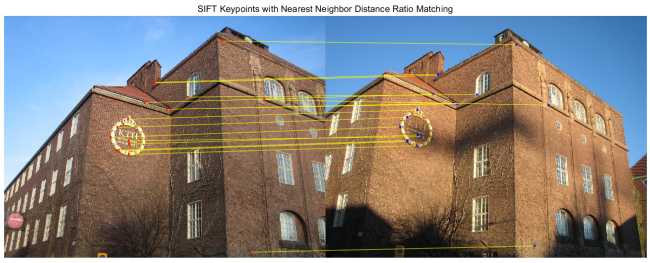

Figure 8: Nearest Neighbor Distance Ratio Matching Algorithm on SIFT with $threshold = 0.78$

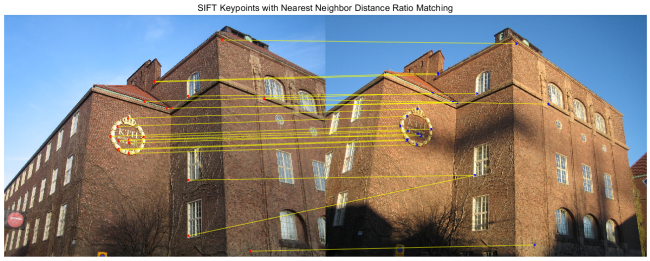

Figure 9: Nearest Neighbor Distance Ratio Matching Algorithm on SIFT with $threshold = 0.80$

ratio threshold to 0.63 and plot side-by-side views with matched feature points connected by lines in Figure 10. It can be observed that there are 30 matches with 3 mismatches. Hence, SURF is not as effective as SIFT when using the "nearest neighbor distance ratio" algorithm.

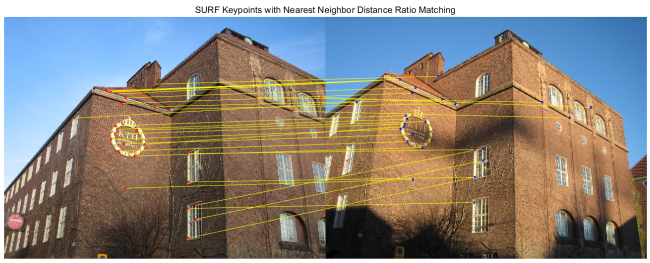

Figure 10: Nearest Neighbor Distance Ratio Matching Algorithm on SURF with $threshold = 0.63$

4 Conclusions

In conclusion, the SIFT algorithm performs better than SURF for both parts. For the robustness against rotation, SIFT performs much better than SURF in the valley part. For the robustness against scale changes, the repeatability of SIFT decreases more slowly than SURF when the scaling factor increases. When both applying the "nearest neighbor distance ratio" matching algorithm, SURF is not as effective as SIFT with more mismatches. Although SURF performs

faster and is more adaptable for real-time and low computational capabilities, SIFT performs better within the scope of this project.

Appendix

Who Did What

Both of us have the same contribution to the completion of this project.

References

[1] Markus Flierl, EQ2425 Analysis and Search of Visual Data, Lecture Slides, 2022

[2] ICT business. https://www.vlfeat.org/install-matlab.html. ultimo accesso 16/09/2021.