Proceedings of the 5th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/15/20230828

A comparison of feature extraction methods in image stitching

Junxin Zheng

School of Computer and Information Engineering, Shanghai Polytechnic University, 2360 Jin Hai Road, Pudong District, Shanghai 201209, China

20201112625@stu.sspu.edu.cn

Abstract. The usage of feature detectors for image stitching has become a popular research area in computer vision. Various feature extraction algorithms can be used in the process of image stitching process, but they perform varyingly when handling different images and no single algorithm could outperform all others. This paper focuses on the comparison of feature extraction algorithms used for panoramic image stitching. The research utilizes the SIFT, ORB, AKAZE, and BRISK to conduct feature points and match feature points on a group of image sets. The RANSAC algorithm is then used to filter out the outliers and calculate the homograph matrix. Completes the panoramic with image splicing and smoothing through the matrix transformation. Derived from the comparison of the matching and stitching results, the AKAZE detector is found to be the fastest feature point detection and extraction algorithm, while the SIFT detector will provide more feature points to make more accurate matches possible. These findings have implications for the development of efficient and effective computer vision technologies for various applications.

Keywords: feature extraction, computer vision, image stitching, panorama images.

1. Introduction

As computer graphics and technology for image merging have advanced, the demands for computer graphics detection have become increasingly complex and diverse, and greater precision and depth in computer image feature detection is a significant fundament of numerous image processing procedures, such as image stitching, camera calibration, dense reconstruction, scene understanding, as well as face recognition and identity verification for security. For instance, panoramic splicing, which involves stitching several continuous photos together, along with filtering and matching feature points, has become a popular solution for creating complete, detailed, high-quality wide-angle 360-degree panoramas [1]. With inaccurately matched point pairs, the estimated warp between adjacent images can be tremendously different from the actual transformation, which will lead to screwed and unusable result images.

Current techniques and methods for image stitching remain imperfect, which means ample room for growth, as it combines a range of fields, such as optics, computer vision, and computer graphics. The rapid advancement of this technology has the potential to facilitate interdisciplinary collaboration and provide significant technological support for other fields, including virtual reality, object recognition, and aiding devices for individuals with vision impairments. By improving the accuracy and efficiency

DOI: 10.54254/2755-2721/15/20230828

of feature detection, computer systems can analyze and understand input images more precisely and thoroughly.

This paper aims to apply different feature detection methods on the same image resource to compare images for similarities using point features as the main determining criteria. The purpose of this experiment is to evaluate the performance of different feature detection methods in the stitching process.

2. Methods

In this chapter, the algorithms used in the experiment will be introduced, as also the platform and evaluation procedure.

2.1. Feature points extracting methods

In the field of computer vision and image processing, a feature point (also known as a keypoint or interest point) refers to a location in an image that possesses visually distinctive and recognizable local features, such as edges, corners, or unique texture patterns [2]. Two basic requirements for image feature points are difference and repeatability, meaning that differences should be recognizable from visually salient points and the same feature is repeatable and matchable from different perspectives [3].

The selection of a suitable corner/edge detection algorithm typically relies on various factors such as the particular application and the balance between speed and precision required for the task at hand.

The detecting algorithms used in this experiment include Oriented Features from Accelerated Segment Test and Rotated Binary Robust Independent Elementary Features (ORB), Scale-Invariant Feature Transform (SIFT), Binary Robust Invariant Scalable Keypoints (BRISK), and Accelerated-KAZE (AKAZE).

2.1.1. ORB. ORB, or Oriented Features from Accelerated Segment Test (FAST) and Rotated Binary Robust Independent Elementary Features (BRIEF), as the name suggests, is a computer vision algorithm that combines two others widely used algorithms, FAST corner detector and BRIEF descriptor. ORB operates by first using FAST to detect corners in an image, and then computing a brief binary descriptor for each detected key point using BRIEF. However, ORB improves on the original BRIEF algorithm by making its descriptors rotation-invariant. This is achieved by calculating the orientation of each key point using a modified version of FAST, and then rotating the descriptor to align with this orientation [4].

ORB has several advantages over other feature detection and description algorithms. One of its strengths is its computational efficiency, which is superior to other popular algorithms like SIFT and Speeded-Up Robust Features (SURF). Additionally, ORB is robust to changes in scale and rotation, making it well-suited for tasks like object recognition and tracking.

2.1.2. SIFT. SIFT, or Scale-Invariant Feature Transform, is one of the most widely used algorithms for feature detection and description in computer vision. To detect feature points, SIFT will first identify scale-space extrema in an image where the Difference of Gaussian (DoG) function reaches a maximum or minimum. These extrema are filtered to keep only those that are stable under different image transformations, such as scale and rotation while eliminating low-contrast points and edge responses [5].

After identifying key points, SIFT generates a descriptor for each key point that is invariant to scale and rotation. The descriptor is a vector of floating-point values that encodes the image gradient orientations and magnitudes in a local neighbourhood around the key point. SIFT is known for its robustness to scale changes and rotation, making it suitable for tasks such as object recognition and tracking. However, one disadvantage of SIFT is its computational complexity, which can make it slow to compute on large images or in real-time applications [6].

2.1.3. BRISK. BRISK (Binary Robust Invariant Scalable Keypoints) is a keypoints/features detection and description algorithm, it has been optimized for being fast and robust for various use cases. When BRISK processes the images, it detects keypoints first and then computes binary descriptors that are

DOI: 10.54254/2755-2721/15/20230828

resilient to brightness and viewpoint variations, efficiently matched using Hamming distance. In order to find scale-invariant keypoints and generate scale-invariant descriptors that can withstand changes in the size of local picture patches around each keypoint, the algorithm employs a scale-space pyramid for detecting keypoints at multiple scales.

By dividing the local image patch surrounding each keypoint into 4 x 4 square areas and computing binary values for each area based on local image gradient orientations, BRISK computes descriptors using a pattern-based methodology. The final descriptor for each keypoint is created by joining the binary codes together. BRISK has proven to be highly effective on several computer vision tasks, including 3D reconstruction, picture matching, and object identification. Nevertheless, it might not perform as well as more recent algorithms, such as AKAZE, which are created especially to tackle complicated image conditions [7].

2.1.4. AKAZE. AKAZE (Accelerated-KAZE) is a feature detection and description algorithm, designed to operate robustly under various complexities of input images, which is an essential requirement in computer vision applications. AKAZE improves on the KAZE algorithm by effectively detecting features using an accelerated version of nonlinear scale space and by proposing a unique descriptor with insensitivity to noise and blur. The nonlinear scale space method of extracting multiscale-oriented patches around each keypoint is used to calculate the AKAZE descriptor, which is then normalized to account for blur and variations in lightning conditions [8].

Eventually, a special feature vector based on the gradient orientation patterns inside the patch is generated, producing a very distinctive and reliable descriptor.

2.2. Feature points matching

Brute-force matches the descriptors with hamming metrics with 2 descriptors obtained in the previous step using the match_descriptors function provided by scikit-image package. A pair of feature points is considered matched when the distance between their descriptors is below a certain threshold [9].

However, some matched feature points could be mismatched, and it is necessary to eliminate them to ensure the accuracy of the final result. To address this issue, the Random sample consensus (RANSAC) algorithm is often employed, it can effectively reduce the impact of noise in the data by identifying and excluding outliers (Table 1).

Table 1. RANSAC procedure.

RANSAC Procedure
1) Select a subset of n data points randomly from all the keypoints.
2) A transformation matrix is estimated using selected data points.
3) Evaluate remaining data by their distance from the model.
4) Remove the points having a distance exceeding the threshold.

2.3. Image stitching

Apply the homography transformation matrix to all 4 corners of one of the images, to see if the top left corner has a negative **x** value, indicating whether that image is on the left side of the resulting image [10]. Wrap that image with translation and homography transformation according to its relative position to the other image, resize the unmodified image, and stitch the warped image into that.

2.4. Image blending and cropping

2 smoothing windows with a linear decreasing slope from 1-0 and an increasing slope from 0-1 are applied to the borders between the 2 images for merging them. Crop the resulting image along straight lines.

DOI: 10.54254/2755-2721/15/20230828

3. Experiment

3.1. Experiment platform and evaluation procedure

The code interpreter used in the experiment is Python 3.9.0, used libraries include OpenCV 4.7.0, pandas1.5.3, scikit-image 0.19.3, numpy 1.23.5, and matplotlib 3.6.2.

Calculating the average time consumption and match rate of 10 batches of processing with each algorithm on both image sets. Evaluate the performance of the algorithms by their accuracy and efficiency.

3.2. Source images

The data source for the experiment comprises a collection of image sets of various live-action scenes. The images contained in Figure 1 are used for the experiment.

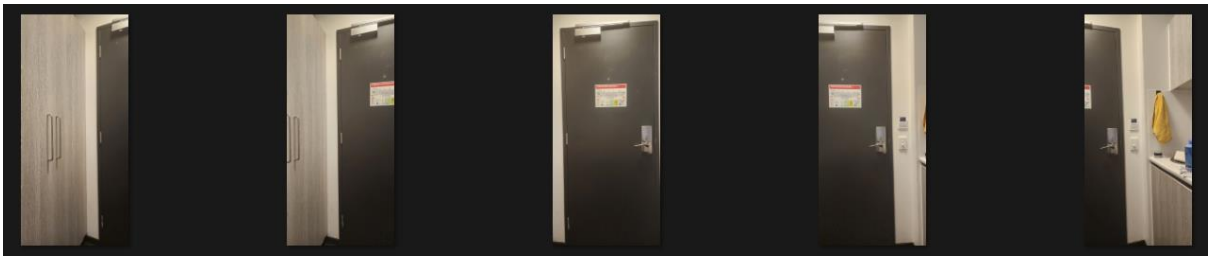

Figure 1. Entryway 001-005 (from left to right).

3.3. Matches plots comparison

Figure 2 and Figure 3 contains result images two steps of the whole matching process with all four different methods. Matches between image2 and image3 (figure 2).

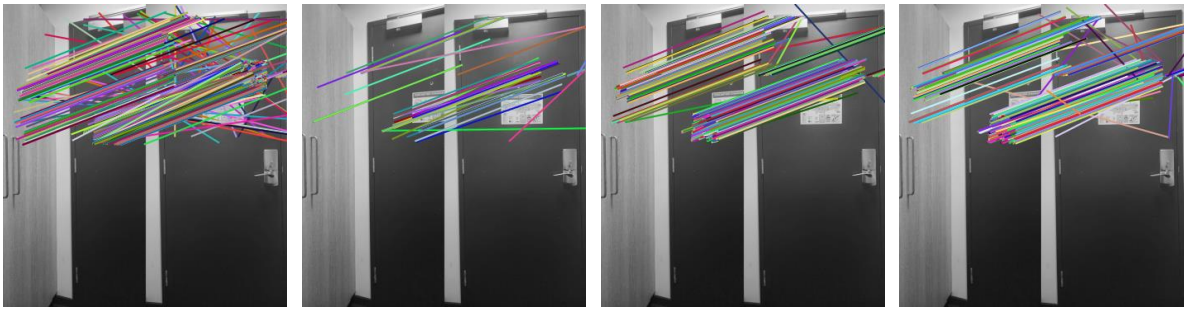

Figure 2. Matches between img2 and img3, from left to right: SIFT, ORB, AKAZE, BRISK.

Matches between image4 and image3 (figure 3)

Figure 3. Matches between img4 and img3, from left to right: SIFT, ORB, AKAZE, BRISK.

DOI: 10.54254/2755-2721/15/20230828

As the figures above show, ORB has a fixed and limited feature points count, which also affected the accuracy of the matching process. AKAZE generally has fewer outliers and fewer feature points than SIFT and BRISK as result. Stitching results (figure 4).

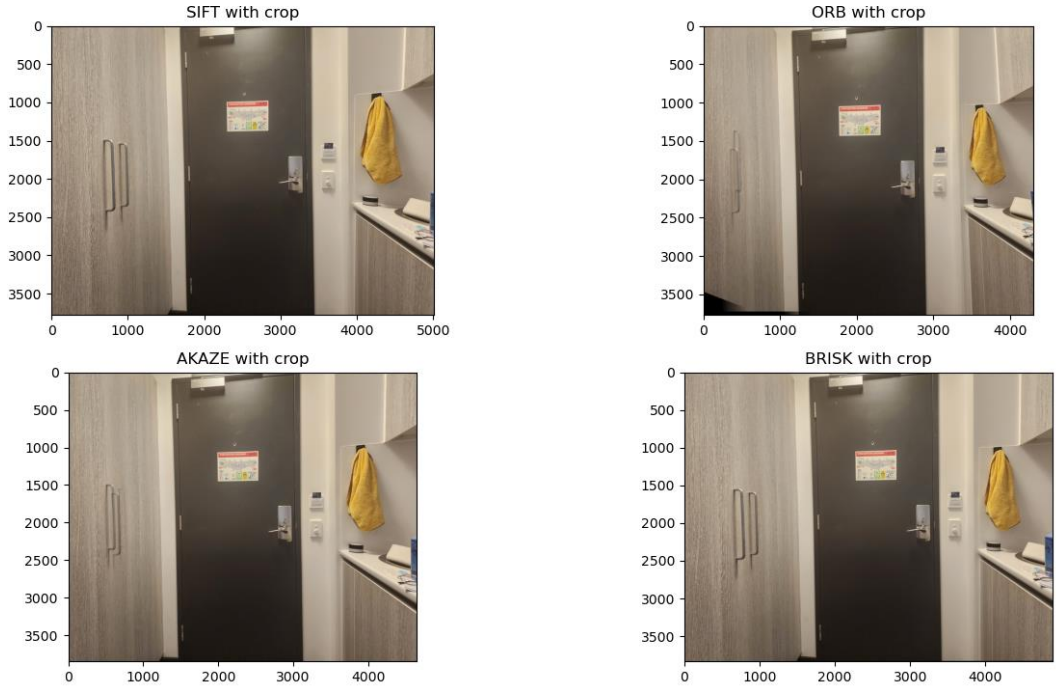

Figure 4. Stitching results, with used algorithms in the title.

As Figure 4 shows, the width of the four result images ranks as follows: SIFT> BRISK>AKAZE>ORB. SIFT preserved the most information from the source images and didn't glitch when dealing with the handlers on the closet.

3.4. Data comparison and analysing

Data collected during stitching img2 and img3 (Table 2).

Table 2. Data collected during stitching img2 and img3.

	Keypoint 1	Keypoint 2	Matches	Match Rate	Time
SIFT	8574	3667	827	0.23	4.26
ORB	500	500	58	0.11	1.55
AKAZE	1540	1847	407	0.26	2.26
BRISK	8916	5575	520	0.09	3.20

Data collected during stitching the combination of img2 and img3 with img1(Table 3).

Table 3. Data collected during stitching img2+3 and img1.

	Keypoint 1	Keypoint 2	Matches	Match Rate	Time
SIFT	23895	13845	585	0.04	23.52
ORB	500	500	22	0.04	1.41
AKAZE	949	1861	119	0.12	2.41
BRISK	19434	7251	188	0.02	6.66

DOI: 10.54254/2755-2721/15/20230828

Data collected during stitching img1+2+3 with img5+4+3(Table 4).

Table 4. Data collected during stitching img1+2+3 with img5+4+3.

	Keypoint 1	Keypoint 2	Matches	Match Rate	Time
SIFT	39784	22680	3333	0.15	75.46
ORB	500	500	209	0.41	1.56
AKAZE	3981	4969	1439	0.36	3.78
BRISK	13028	19620	3021	0.23	11.30

As the histogram below (figure 5) shows, the AKAZE algorithm keeps a relatively high match rate throughout the whole process, only to be surpassed by ORB at the final step.

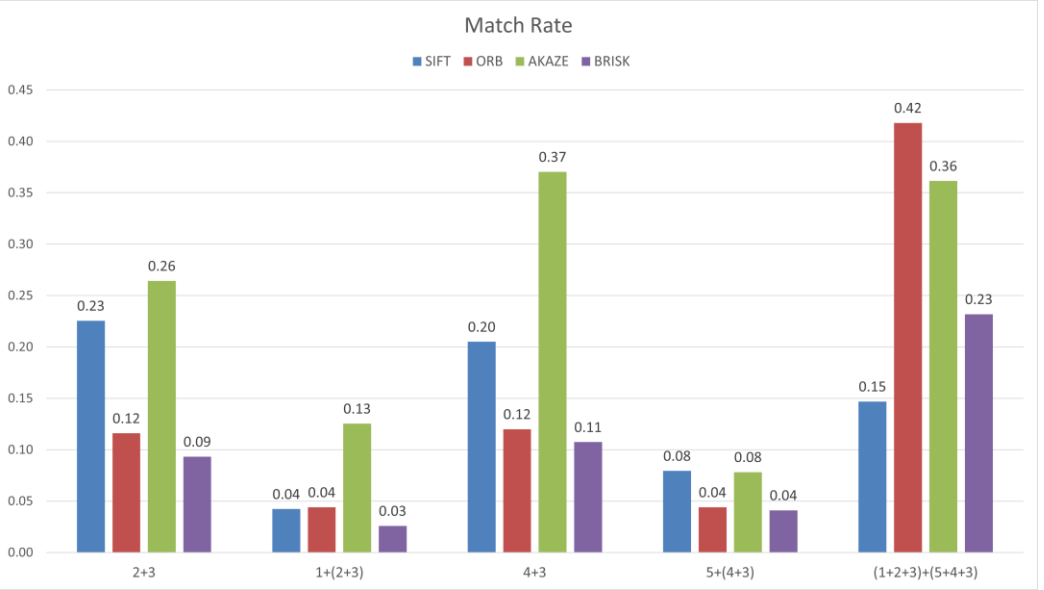

Figure 5. Match rates comparison.

As the histogram below (figure 6) shows, SIFT has generally the highest time costs because of the detecting procedure, and ORB has the lowest and the most stable time consumption due to fixed keypoints counts.

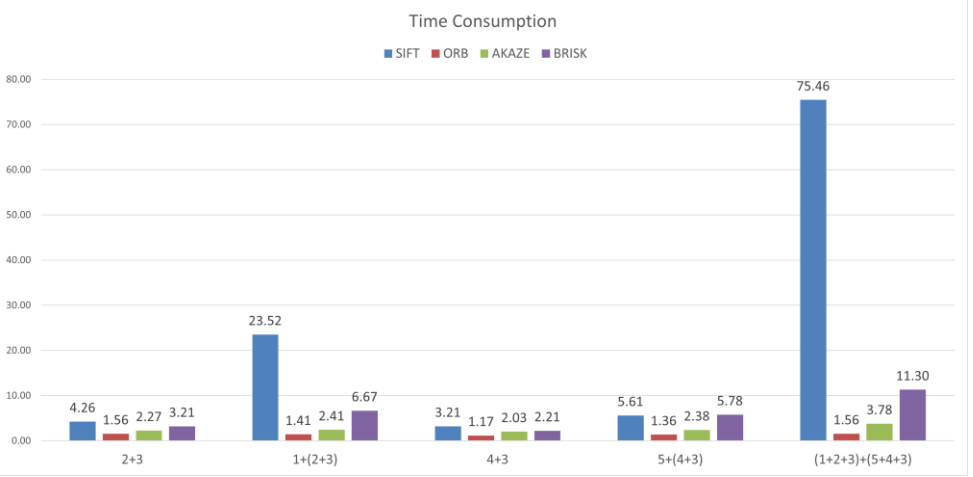

Figure 6. Time consumption comparison.

DOI: 10.54254/2755-2721/15/20230828

Throughout the entire testing process, the ORB detector has been the most time-efficient method in most cases and is currently the most stable feature detection and extraction algorithm. On the other hand, AKAZE usually will have higher match rates. SIFT is the most time-consuming one among all 4 algorithms, which doesn't necessarily lead to higher catch rates or more accurate matches.

4. Conclusion

In conclusion, this study focused on the use of feature detection algorithms for image processing and computer vision applications. Specifically, this work investigated the effectiveness of SIFT, ORB, AKAZE, and BRISK detectors for feature point detection and matching in image sets. Through the use of the RANSAC algorithm to filter out outliers and calculate the homography matrix, this paper was able to compare the results of panoramic image splicing and smoothing using the four different feature detectors.

In the experiment section, target image sets were processed by 4 different detecting algorithms, to measure the match rate and time consumption of different detectors. Our findings indicate that the AKAZE detector is the fastest algorithm for feature point detection and extraction, while the SIFT detector is able to provide a larger number of feature points for more accurate matches, and ORB is the most time-efficient one.

References

[1] Brown, M., & Lowe, D. G. Recognizing panoramas. 2003, Inter. Conf. Com. Vis. 3, 1218.
[2] Szeliski, R. Computer Vision: Algorithms and Applications. 2011 Sci. Bus. Media.387.
[3] Jolhip, M. I., Minoi, J. L., & Lim, T. A comparative analysis of feature detection and matching algorithms for aerial image stitching. 2017, J. Tele., Elec. Com. Eng., 9(2-10), 85-90.
[4] Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. ORB: an efficient alternative to SIFT or SURF. 2011 Inter. Conf. Comp. Vis. 2564-2571.
[5] Lowe, D. G. Distinctive image features from scale-invariant keypoints. 2014 Inter. J. Comp. Vis, 60, 91-110.
[6] Karami, E., Prasad, S., & Shehata, M. Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images. 2017 arXiv preprint arXiv:1710.02726.
[7] Leutenegger, S., Chli, M., & Siegwart, R. Y. BRISK: Binary robust invariant scalable keypoints. 2011 Inter. Conf. Comp. Vis. 2548-2555.
[8] Alcantarilla, P. F., Bartoli, A., & Davison, A. J. KAZE features. 2012 Euro. Conf. Comp. Vis., VI 12, 214-227.
[9] Ou, Y., Cai, Z., Lu, J., Dong, J., & Ling, Y. Evaluation of Image Feature Detection and Matching Algorithms. 2020 Inter. Conf. Comp. Comm. Sys. 220-224.
[10] Nickfarjam, A. M., Ebrahimpour-Komleh, H., & Tehrani, A. A. Binary image matching using scale invariant feature and hough transforms. 2018 Adv. Sci. Eng. Tech. Inter. Conf. 1-5.