FISB: Feature-based Image Stitching Benchmark

Abhishek Rajora
Indian Institute of Technology, Jodhpur
rajora.1@iitj.ac.in

Abu Shahid
Indian Institute of Technology, Jodhpur
shahid.3@iitj.ac.in

Abstract

Image stitching is a broad and active research area in image processing and computer vision. This paper presents a detailed study of feature-based image stitching algorithms with different evaluation metrics. We present an image stitching pipeline for creating panoramic images from multiple input images. The proposed method involves several stages, including feature extraction, matching, and blending. For evaluation, the Google Landmarks database was used. In addition, a custom data set of 49 scenes is made with images taken from different viewpoints and varying illumination. This data set is used to evaluate the effectiveness of the proposed pipeline and to fine-tune the pipeline's parameters. The performance of the proposed pipeline is evaluated using both objective and subjective measures, including accuracy, speed, and visual quality. Experimental results show that the proposed pipeline can effectively stitch images and produce seamless panoramas. The pipeline is scalable and can be applied to a wide range of applications, such as surveillance, virtual reality, and cartography.

1. INTRODUCTION

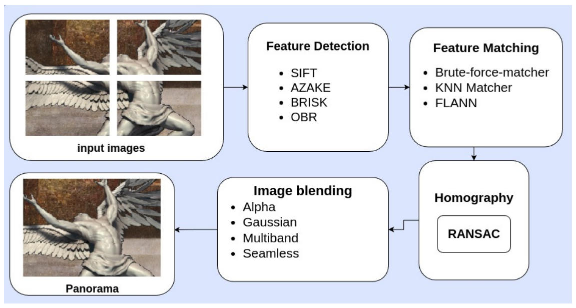

Figure 1. FISB pipeline

One of the major limitations of current image stitching algorithms is their inability to handle images taken from different perspectives, with different illuminations, or when images are rotated. Additionally, many of the basic and simple implementations of these algorithms can only stitch images in one orientation, such as horizontal or vertical. This restricts their usefulness for more complex applications, such as creating panoramas or stitching images from multiple viewpoints.

In this paper, we propose a pipeline for image stitching that incorporates several popular feature-detecting algorithms, including SIFT, AKAZE, BRISK, and ORB. We employ brute force matching and FLANN with K-nearest neighbor (KNN) algorithm to remove false positives. Following feature matching, we utilize homography to combine overlapping features of images using the RANSAC algorithm. The pipeline determines whether the images are to be stitched based on a probabilistic overlap threshold. The effectiveness of our proposed pipeline is demonstrated through experiments on a variety of images.

There is currently no dataset available for the purpose of image stitching benchmarking. As a result, we created our own dataset consisting of 49 scenes, which include a mix of real and digital, indoor and outdoor scenes. The sub-images that need to be stitched vary in terms of rotation, perspective, viewpoint, zoom, and illumination. The purpose of creating this dataset was to test the robustness of a method and determine its performance. We took into consideration the noise that is commonly associated with real-world use cases.

2. IMAGE STITCHING TECHNIQUES

In image stitching algorithms, the choice of feature extraction algorithm is critical for accurate and efficient image alignment.

2.1. Feature Detection Algorithm

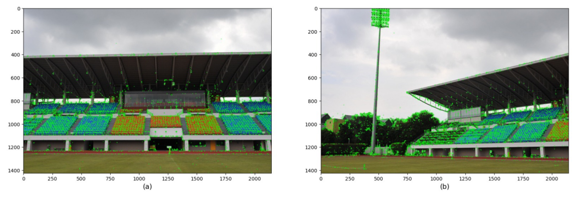

Figure 2. Feature Detection using SIFT

Feature detection algorithms use descriptors to detect and describe the keypoints in an image. The keypoints are then used to match the images. The most popular feature detection algorithms are SIFT, AKAZE, ORB, BRISK, AKAZE, and KAZE.

1. **SIFT:** SIFT (Scale-Invariant Feature Transform) is a feature detection algorithm that uses a scale-space representation of the image to detect and describe keypoints. SIFT uses a Gaussian pyramid to generate a scale-space representation of the image and then uses the difference of Gaussian (DoG) to detect keypoints. SIFT uses a local neighborhood of each keypoint to generate a feature descriptor.
    - BFMatcher object is created with parameters normType=L2_Norm and crossCheck kept True
    - KNN matcher is used with k=2 to draw two match-lines for each keypoint and crossCheck is kept False
    - FLANN_INDEX_KDTREE = 1 (passed in algorithm) and trees = 5 is set while using FLANN based Matcher
2. **AKAZE:** AKAZE (Accelerated-KAZE) is a feature detection and description algorithm used in computer vision and image processing. It is an extension of the KAZE algorithm and is designed to be computationally efficient and robust to changes in scale and viewpoint. AKAZE uses a nonlinear scale space to detect and describe keypoints in an image and generates feature descriptors using a novel combination of binary and floating-point values.
    - In BFMatcher object normType = NORM_HAMMING and crossCheck = True are set
    - Same parameters are kept for KNN Matcher and FLANN-based Matcher as SIFT
3. **BRISK:** BRISK (Binary Robust Invariant Scalable Keypoints) is a binary descriptor that uses a scale-space pyramid to extract keypoints and produces a compact binary descriptor that is efficient to compute and compare. BRISK uses a sampling pattern to determine the location and scale of keypoints, and a binary descriptor to describe the local image features.
    - For BF Matcher, normType = NORM_HAMMING is preferred for Brisk in BF Matcher, cross-check is kept False
    - Same parameters are kept for KNN Matcher and FLANN-based Matcher as SIFT and AKAZE
4. **ORB:** ORB (Oriented FAST and Rotated BRIEF) is another binary descriptor that uses a combination of FAST keypoint detector and BRIEF (Binary Robust Independent Elementary Features) descriptor to extract and match keypoints. ORB uses a rotated patch to estimate the orientation of the keypoints, making it more robust to rotation than other feature extraction algorithms. It also uses a multi-scale pyramid approach for keypoint detection and descriptor computation. Generally, it detects more features than required, therefore though being fast in speed its accuracy certainly drops.
    - normType = NORM_HAMMING is preferred with crossCheck = True
    - KNN parameters are kept the same for all algorithms
    - For Flann Matcher,
        - FLANN_INDEX_LSH = 6 is passed into the algorithm
        - table_number = 6, key_size = 12, multi_probe_level = 1 are set according to the docs

2.2. Feature Matching

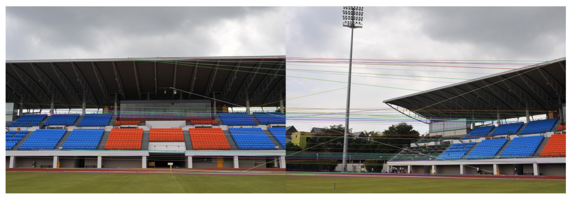

Figure 3. Brute Force feature matching with KNN

After feature detection, feature matching compares these features between the images to find corresponding points.

- **Brute force feature (BF) matching** is a simple but effective technique for finding matching feature points between two images. The basic idea behind brute force matching is to compare each feature in one image to every feature in the other image and find the closest

match based on some distance metric.
Brute force matching can be computationally expensive but is guaranteed to find the exact match.

- **K-nearest neighbor (KNN) matching** is a more efficient alternative to brute force matching. KNN matching is a two-step process. First, the algorithm finds the K nearest neighbors of each feature descriptor. Then, it filters out the matches that are not unique. For example, if the nearest neighbor of a feature descriptor in one image is also the nearest neighbor of another feature descriptor in the other image, then the match is considered unique, and kept rest are discarded.

- **FLANN (Fast Library for Approximate Nearest Neighbors) matching** is a fast and efficient algorithm for finding nearest neighbors in large datasets. FLANN is a library that implements a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features.
It contains a collection of algorithms we refer to as approximate nearest neighbors (ANN) algorithms. These algorithms can be used to search for nearest neighbors with a query point in sets of points with arbitrary sizes. The algorithms in FLANN can also be used to find a good approximation of the nearest neighbors with a query point in sets of points with arbitrary size.

2.3. Geometric Transformation

We utilize the matched key points to estimate the homography matrix that aligns the images. Homography estimation is a crucial step in image stitching, as it determines the transformation required to warp the images into a common coordinate system. Moreover, in real-world use cases, such as cartography from drone images, a change of viewpoint and perspective is a surety, which makes this step extremely necessary. Given the matched feature points, the geometric transformation between the two images is estimated using a robust estimation algorithm such as Random Sample Consensus (RANSAC). The transformation can be a homography or affine transformation, depending on the number of matching points and their distribution.

2.4. Image Blending

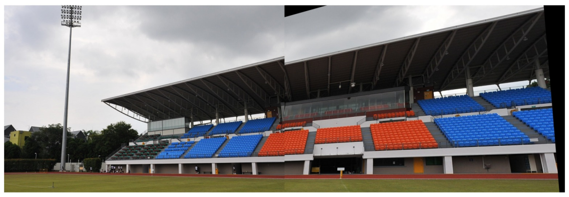

Figure 4. Image Blending with warp perspective

Once the homography matrix is estimated, we can use it to warp one of the images onto the plane of the other image, so that the two images are aligned. However, the resulting image may contain visible seams or artifacts due to differences in brightness, color, or texture between the two images.
We can use image blending techniques to smooth out the seams and create a more seamless transition between the images to address this issue. One common approach is to use a weighted average of the pixel values in the overlapping region, where the weights depend on the distance of each pixel from the boundary of the overlapping region.
Finally, Image blending with warp perspective is utilized in the pipeline, which allows us to create seamless panoramas from multiple overlapping images. It requires accurate estimation of the homography matrix and careful selection of the blending technique to ensure a smooth and seamless transition between the images. On our end, we implemented alpha and gaussian blending techniques to achieve this.

1. **Alpha blending** Alpha blending is a process that takes the transparency of each pixel into account. The degree of transparency, or alpha value, is used to blend the colors of the foreground and background images or objects in a way that creates a smooth transition between them.

```
i. Choose a masking value M( transparency level)
ii. Final image= M*img1 + (1-M)*img2
```

---

2. **Gaussian blending** Gaussian blending, on the other hand, is a process of blending images or objects using a Gaussian distribution. In this method, a Gaussian filter is applied to each pixel, which smooths out the sharp edges and creates a softer transition between the images or objects being blended. The degree of smoothing is determined by the size of the filter kernel.

```
i. Compute the Gaussian pyramid for each image
ii. Compute the Laplacian pyramid for each image
iii. Combine the left and right halves of each level of
the Laplacian pyramid
iv. Reconstruct the blended image from the Laplacian
pyramid
```

---

3. **Seamless blending** This method uses an image processing operator that enables the merging of two images without creating any unsightly seams. This method also ensures that the color of the inserted image is likewise altered so that the inserted object appears to be a natural part of the target image's surroundings.

---

i. Create a mask for the center of the image
ii. Create a rough mask around the center of the image
iii. Use the rough mask around the center to find the center of the image
iv. Use the center to create a seamless cloning mask

4. **Multiband blending** In this process, the photos that will be blended are first divided into a collection of component images with band-pass filters. After that, a comparable bandpass mosaic is constructed from the individual component images. Using a weighted average within a transition zone whose size is proportional to the wave durations represented in the band, component images are merged in this step. In order to create the desired image mosaic, these band-pass mosaic images are finally added together. When coarse features are present close to boundaries, they are blended gradually across a sizable distance without distorting or otherwise impairing the finer aspects of the surrounding image.

i. Create a rough mask around the center of the image
ii. Convert the rough mask to the required type
iii. Use the rough mask to find the center of the image
iv. Create a multiband blender blend the images

3. COMPARATIVE STUDY AND ANALYSIS

Section 2 discussed about the literature of various image-stitching techniques with the integration of various featured detection, feature matching and image blending algorithms. The compatibility of parameters for the above algorithms are well-known and listed in the documentation[2]. The end-to-end pipeline for all the techniques is constructed and tested upon.

3.1. Quantitative Analysis

For the performance evaluation of different feature-based image stitching techniques, we executed our pipeline with various modalities.

TABLE I. Comparison of Feature Detection Techniques

Algorithm	Matches	MSE	SSIM	PSNR	NMI
SIFT	3707	4.89	0.96	22.26	0.94
AKAZE	4210	6.73	0.89	20.62	0.91
BRISK	3798	5.32	0.92	18.54	0.87
ORB	11876	10.34	0.52	8.87	0.65

It can be seen from the table that ORB is performing relatively poorly compared to the other three algorithms indicating that its high speed comes with a tradeoff. The orb descriptor produces redundant features which shadow the good ones while finding good matches.

We now simulate the pipeline for all the matching algorithms keeping SIFT as a constant feature detection algorithm.

TABLE II. Comparison of Feature Matching Techniques

Matcher	Matches	MSE	SSIM	PSNR	NMI
BF	3707	4.89	0.96	22.26	0.94
KNN	4765	5.1	0.89	19.67	0.91
FLANN	7320	6.2	0.92	14.35	0.87

From Table II, we see the FLANN though being relatively fast produces sub-optimal results. Feature matcher can therefore be chosen based on context, need, and size of the dataset. As our dataset is small, the overhead of using a complex algorithm is negligible.

TABLE III. Comparison of Image Blending Techniques

Blender	MSE	SSIM	PSNR	NMI
Alpha	8.43	0.55	7.98	0.76
Gaussian	9.72	0.49	12.23	0.63
MultiBand	6.18	0.92	18.49	0.87
Seamless	4.89	0.96	22.26	0.94

On analyzing the above table (Table III), we can conclude that the Seamless Clone blending is able to merge images with smoother transitions and better results on all fronts. MultiBand blender performed well due to its decomposition of images into multiple frequency bands thus preserving the high-frequency details while still producing smooth transitions between the images.

Alpha and Gaussian blenders performed relatively poorly since the quantitative approach of alpha blending involves computing a weighted sum of the two input images based on the alpha mask which is highly sensitive to the mask value. Gaussian Blended is rather a noisy image and thus it's blurry when comparing the resulting panorama with the original image.

3.2. Qualitative Analysis

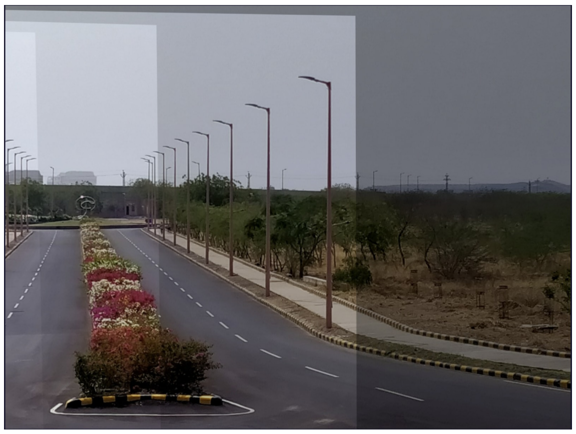

Figure 5. Alpha Blending

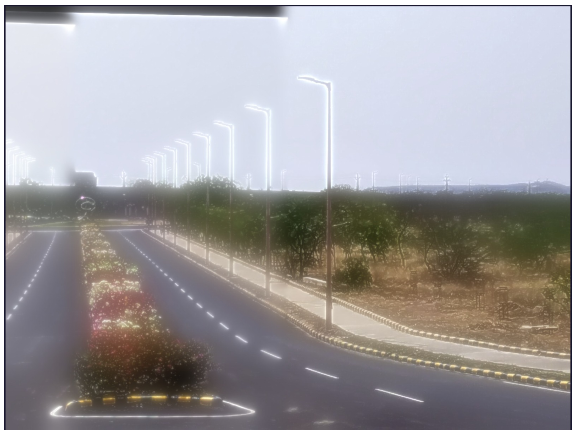

Figure 6. Gaussian Blending

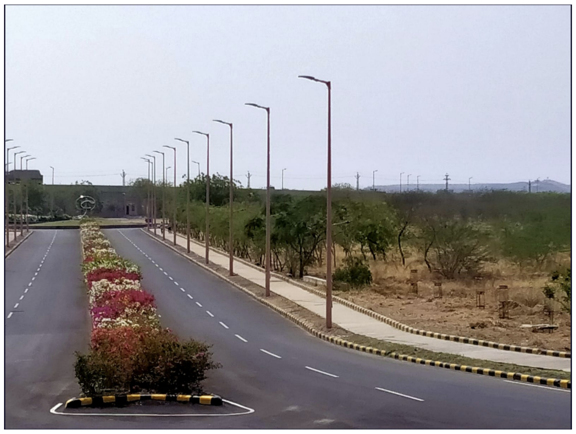

Figure 7. MultiBand Blending

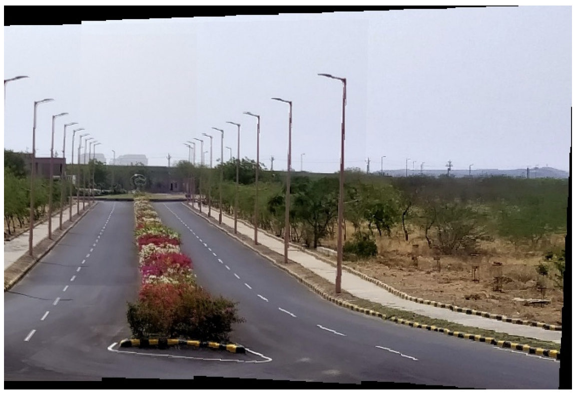

Figure 8. Seamless Clone Blending

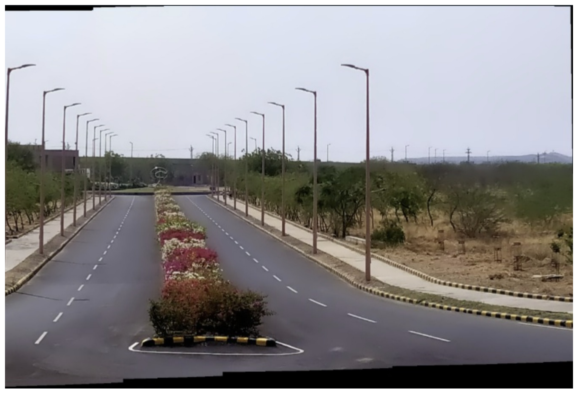

Figure 9. OpenCV

From the above observation, we can deduce that seamless clone is the best choice for image blending. It is a technique that uses gradient-domain compositing to merge images. It creates a smooth transition between images without introducing any visible seams. The main advantage of seamless blending is that it produces very high-quality results with no visible artifacts or blending seams. In general, seamless blending tends to produce higher-quality results with no visible seams, while alpha Gaussian Multi-Band blending may preserve high-frequency details better.

4. Conclusion

To evaluate the performance of each method, we conducted experiments using the Google Landmark dataset. Images were taken and cropped into sub-images and then fed into our pipeline. The Google Landmark dataset however have will have sub-images in a single perspective and have global illumination. To test the robustness of the proposed method, we also created our own dataset (currently have 49 scenes). Each set is made up of multiple sub-images having varying illumination, orientation, and perspective and a super image (to be used for validation). We measured each method's accuracy, computational complexity, and robustness regarding the number of correctly matched feature points, the time required for feature extraction and matching, and the sensitivity to image noise and illumination changes. However, to show the mean trends, the exercise needs to be carried with more scene images with the end-to-end pipeline. We, however, indulged extensively in experimenting with popular image quality metrics and the same has been reported in the previous sections. Given our use-case of a small dataset, more than sufficient compute, and a non-real-time setting; SIFT as feature descriptor, BF matcher as feature matcher, and Seamless Clone as image blender gave the best results.

References

[1] Zhe Huang Luyu Yang, Zhigang Tan and Gene Cheung. A content-aware metric for stitched panoramic image quality assessment. *ICCV Workshop*, 2017.

[2] Moushumu Zaman Bonny and Mohammad Shorif Uddin. Feature-based image stitching algorithms. *International Workshop on Computational Intelligence (ICWI)*, 2016.

[3] PETER J. BURT and EDWARD H. ADELSON. A multiresolution spline with application to image mosaics. *RCA David Sarnoff Research Center*.

[4] Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. *International Conference on Computing, Mathematics and Engineering Technologies – iCoMET*, 2018.

[5] Bin Song Yanning Guo Yue Sun, Yueyong Lv and Lifan Zhou. Image stitching method of aerial image based on feature matching and iterative optimization. *Proceedings of the 40th Chinese Control Conference*, 2021.

[6] Yao Li and Lizhuang Ma. A fast and robust image stitching algorithm. *Proceedings of the 6th World Congress on Intelligent Control and Automation*, 2006.

[7] Moushumu Zaman Bonny and Mohammad Shorif Uddin. Feature-based image stitching algorithms. *International Workshop on Computational Intelligence (ICWI)*, 2016.

[8] Zhongke Zhang. Image stitching algorithm based on combined feature detection. *IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, 2020.

[9] Michael S. Brown Julio Zaragoza, Tat-Jun Chin and David Suter. As-projective-as-possible image stitching with moving dlt. *CVPR*, 2013.

[1] [2] [3] [4] [5] [6] [7] [8] [9]