Panorama image stitching using SIFT feature detector.

Nguyen Hoang Hiep
HUST
firstauthor@i1.org

Vu Duc Minh
HUST
secondauthor@i2.org

Abstract

Panorama stitching is a key application in computer vision that involves merging multiple images into a cohesive wide-angle view. In this paper, we propose a novel image stitching approach utilizing the Scale-Invariant Feature Transform (SIFT) feature detector to identify stable and distinctive keypoints across overlapping images. The method leverages SIFT's robust keypoint detection and description capabilities to handle significant scale and rotation variations. By matching these keypoints between images, we estimate geometric transformations, including homographies, to accurately align the images. A final blending procedure minimizes seams and artifacts, resulting in a seamless panoramic image. Our approach demonstrates superior performance in handling real-world challenges, such as partial occlusions, variable lighting conditions, and different camera orientations. Experimental evaluations show that the proposed technique achieves state-of-the-art results in terms of alignment precision and visual quality, making it a reliable solution for panoramic image stitching tasks.

1. Introduction

In many computer vision tasks, it is often necessary to combine multiple image segments into a single panoramic image. This process, known as image stitching, is commonly used in applications such as panoramic photography. The challenge lies in identifying common features between overlapping images, aligning them accurately, and seamlessly blending them into a single image.

We will implement a method to process multiple images of the same scene, which may be captured from different angles or have undergone various transformations such as rotation, scaling, translation, or noise. The goal is to generate a single, seamlessly stitched image or panorama that combines these individual images into one cohesive representation of the scene.

To do this, we need to identify key points in images that remain consistent despite transformations, and accurately match these features across different images. Additionally, we must handle any distortions that may occur during the stitching process, particularly when combining multiple images.

2. Related works

2.1. Harris Corner Detection

Harris Corner Detection is a widely used technique to identify corners in images, which are points where there is a significant change in both horizontal and vertical gradients. The method works by analyzing the intensity changes in an image and calculating the eigenvalues of the second moment matrix, which captures how the image's intensity varies in different directions. Corners are regions where the gradient changes significantly in all directions, making them easily identifiable through this matrix. The Harris corner measure is computed, and local maxima in this measure are considered as corners. However, it is sensitive to noise, has poor performance in low-contrast areas, and is not invariant to scale or rotation. Additionally, its computational cost can be high for larger images or real-time applications.

2.2. Shi-Tomasi Corner Detector

The Shi-Tomasi Corner Detector improves upon Harris Corner Detection by modifying the corner response function to use the smaller eigenvalue of the moment matrix directly for corner detection. This adjustment provides more reliable corner detection, especially in low-contrast areas or when the gradients are weaker. The method is less sensitive to noise and variations in gradients, offering better corner detection in challenging conditions. While it still shares limitations with Harris, such as sensitivity to scale and rotation, it performs more consistently in practical applications such as object tracking and 3D reconstruction.

2.3. Scale-Invariant Feature Transform (SIFT)

SIFT (Scale-Invariant Feature Transform) is a powerful feature detection and description method designed to detect keypoints that are invariant to scale, rotation, and partially to affine transformations and noise. It detects potential key points by searching for extrema in the scale-space, identi-

fies stable key points, and assigns orientation to ensure rotation invariance. SIFT then generates descriptors based on local image information around each key point, enabling reliable keypoint matching across images. While SIFT is robust to various transformations, it is computationally expensive, making it less suitable for real-time applications despite its effectiveness in complex tasks like image stitching and object recognition.

2.4. Sped-Up Robust Features (SURF)

SURF (Speeded-Up Robust Features) improves on SIFT by offering similar robustness to scale, rotation, and affine transformations, while being faster and more computationally efficient. SURF uses a fast approximation of the Hessian matrix to detect interest points instead of the computationally expensive difference of Gaussian function. For feature description, SURF uses Haar wavelet responses, computed efficiently via integral images. This makes SURF more suitable for real-time applications, such as video processing and object tracking. While faster than SIFT, SURF is still a patented method, limiting its use in open-source projects.

2.5. Features from Accelerated Segment Test (FAST)

FAST is a corner detection algorithm designed for speed and efficiency. It evaluates the intensity differences between a candidate pixel and its surrounding pixels in a circular region. If a set of contiguous pixels is brighter or darker than the candidate, the point is classified as a corner. The main advantage of FAST is its speed, which makes it suitable for real-time applications. However, it lacks a feature descriptor and is sensitive to noise and changes in illumination, which limits its robustness compared to other methods like SIFT or SURF.

2.6. Binary Robust Independent Elementary Features (BRIEF)

BRIEF is a binary feature descriptor used in combination with corner detection techniques like FAST or Harris. It encodes the local image structure by comparing pixel intensities within a region around the keypoint and generating a binary string. This binary format makes BRIEF highly efficient in terms of computation and memory usage, which is advantageous for real-time applications. However, BRIEF is not invariant to scale or rotation and can struggle with changes in illumination or noise. Despite these limitations, its efficiency makes it a popular choice when speed is prioritized over robustness to transformations.

2.7. Oriented FAST and Rotated BRIEF (ORB)

ORB combines the FAST corner detector with BRIEF for feature detection and description, aiming for speed while ensuring robustness to scale and rotation. By computing the orientation of keypoints, ORB aligns the BRIEF descriptor to this orientation, making it invariant to rotation. ORB is computationally efficient, using binary descriptors for quick matching, and is ideal for real-time applications like robotics and augmented reality. While it is not as scale-invariant as SIFT or SURF, its fast performance and freedom from patent restrictions make it a popular choice for a wide range of tasks.

2.8. Topic of research

In this project, the Scale-Invariant Feature Transform (SIFT) method is chosen for its proven robustness to scale, rotation, and affine transformations, making it ideal for tasks such as image matching, object recognition, and 3D reconstruction.

3. SIFT

SIFT operates by detecting distinctive keypoints in an image that are robust to changes in scale, rotation, and even partially affine transformations. These keypoints are then described with feature vectors, which can be used for image matching, recognition, or stitching.

The SIFT algorithm consists of the following main steps:

1. **Scale-space Extrema Detection**
2. **Keypoint Localization**
3. **Orientation Assignment**
4. **Keypoint Descriptor Generation**
5. **Keypoint Matching**

3.1. Scale-space Extrema Detection

The first step in the SIFT algorithm is detecting keypoints across different scales. This is done by constructing a scale-space using a series of images with different levels of blurring. The scale-space is created by convolving the original image with Gaussian filters at multiple scales. The Difference of Gaussian (DoG) is used to approximate the Laplacian of Gaussian (LoG), which is commonly used for feature detection.

The DoG is defined as:

$$D(x, y, \sigma) = \sigma^2 \left(G(x, y, \sigma) - G(x, y, \kappa\sigma)\right)$$

where:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

is the Gaussian function, and σ is the scale parameter. The parameter κ is typically set to $\sqrt{2}$.

This difference of Gaussian (DoG) is calculated at various scales, and the extrema (local minima and maxima) of the DoG images are identified. These extrema correspond to potential keypoints in the image.

3.2. Keypoint Localization

Once potential keypoints are detected as scale-space extrema, the next step is keypoint localization. This process refines the locations of the keypoints by eliminating those that are unstable or poorly localized. Keypoints that are too close to image borders or are affected by noise are discarded.

Keypoint localization involves fitting a quadratic function to the local extrema to find the precise location and scale of the keypoint. The Taylor expansion of the DoG function is used to refine the location of each keypoint. The keypoint position is refined using the following equations for the spatial position (x, y) and the scale σ:

$$\Delta x = -\left(\frac{\partial^2 D}{\partial x^2}\right)^{-1} \frac{\partial D}{\partial x}$$

$$\Delta y = -\left(\frac{\partial^2 D}{\partial y^2}\right)^{-1} \frac{\partial D}{\partial y}$$

$$\Delta \sigma = -\left(\frac{\partial^2 D}{\partial \sigma^2}\right)^{-1} \frac{\partial D}{\partial \sigma}$$

where the partial derivatives are taken with respect to the spatial position and scale.

3.3. Orientation Assignment

To make the keypoints invariant to rotation, an orientation is assigned to each keypoint based on the local image gradients around the keypoint. The gradient magnitude and orientation at each pixel are computed as follows:

$$m(x, y) = \sqrt{\left(\frac{\partial I(x, y)}{\partial x}\right)^2 + \left(\frac{\partial I(x, y)}{\partial y}\right)^2}$$

$$\theta(x, y) = \arctan\left(\frac{\frac{\partial I(x,y)}{\partial y}}{\frac{\partial I(x,y)}{\partial x}}\right)$$

where $I(x, y)$ is the image intensity at pixel (x, y).

A histogram of gradient orientations is constructed around each keypoint. The orientation of the keypoint is assigned as the peak of this histogram. In cases where multiple peaks are found, the keypoint can have multiple orientations, each representing a distinct keypoint.

3.4. Keypoint Descriptor Generation

Once the keypoints are localized and oriented, the next step is to generate a descriptor for each keypoint. The descriptor is a vector that encodes the local image structure around the keypoint. This is achieved by dividing the region around the keypoint into a grid of cells (e.g., 4x4 cells), computing the gradient orientation and magnitude for each pixel in the cell, and then creating a histogram of gradient orientations for each cell.

The descriptor is typically a 128-dimensional vector, which is constructed by concatenating the histograms from all the cells. The final descriptor is normalized to be robust to changes in illumination. A typical SIFT descriptor is normalized as follows:

$$\mathbf{d} = \frac{\mathbf{d}}{|\mathbf{d}|} \quad \text{if} \quad |\mathbf{d}| > \lambda$$

where λ is a threshold to limit the effect of large gradients, and $|\mathbf{d}|$ is the L2-norm of the descriptor.

3.5. Keypoint Matching

After extracting descriptors from multiple images, the keypoints from different images can be matched based on the similarity of their descriptors. The most common method for matching descriptors is using the Euclidean distance between descriptor vectors. The distance d between two descriptors $\mathbf{d}_1$ and $\mathbf{d}_2$ is given by:

$$d(\mathbf{d}_1, \mathbf{d}_2) = \sqrt{\sum_{i=1}^{n} (d_{1i} - d_{2i})^2}$$

where n is the number of elements in the descriptor vector.

The keypoints with the smallest distances are considered to be the best matches. Often, a ratio test is used to reject poor matches by comparing the distance between the closest match to the distance of the second closest match. This ensures that the matches are sufficiently distinct and reliable.

3.6. Advantages of SIFT

SIFT has several advantages that make it suitable for various computer vision tasks:

- **Scale Invariance:** SIFT is invariant to changes in scale, meaning it can detect the same feature in images of different sizes.
- **Rotation Invariance:** The keypoints are oriented, making them invariant to rotations of the image.
- **Robust to Affine Transformations:** SIFT can handle affine transformations to some extent, which makes it useful for applications where the camera viewpoint changes.
- **Distinctive Descriptors:** The descriptors are highly distinctive, even in challenging conditions such as illumination changes, partial occlusions, and noise.

4. Experiment

In this section, we present the results and observations made during the experiment. The primary focus was on testing the functionality and performance of the image stitching pipeline, which was built using SIFT keypoint detection, feature matching, and image transformations. This experiment evaluated how well the program handled differ-

ent datasets and image transformations (e.g., rotation, scaling, and noise addition) as well as its ability to generate panoramic images.

4.1. Dataset and Input Selection

The first step in the experiment was selecting the dataset and the type of input images to be processed. The program allowed users to choose between two main options:

1. A dataset provided by the professor (for example, SRT1), with images representing typical urban street scenes.
2. A set of images prepared manually, where the user could input images from personal sources, such as Google Maps or other image sources.

In the case of the dataset provided, images were preprocessed and stored in specific directories, while the manually provided images were prepared with a user-friendly interface for easier selection.

4.2. Feature Detection and Matching

After the input images were selected, the next crucial step was detecting keypoints using the SIFT algorithm. Keypoints from the different images were matched using the 'cv::BFMatcher' in OpenCV. This method utilizes the nearest neighbor search to find the most similar keypoints across the images.

For feature matching, the 'knnMatch' method was employed, where the nearest neighbor match was tested for ratio consistency. A ratio test, with a threshold of 0.75, was applied to ensure that the closest match was sufficiently distinct from the second closest match. The program then visualized and saved the matching keypoints on the images using 'cv::drawMatches'.

Example images showing the feature matching results for various datasets are presented below. These images demonstrate how well the SIFT algorithm detected and matched keypoints across multiple images:

As the process continued with additional images, the feature matches were consistent, and the results showed that the system could reliably detect and match features across multiple images in a dataset.

4.3. Stitching the Images

Following the feature detection and matching process, the images were stitched together to create panoramic views. In this experiment, we tested the stitching process for both two and multiple images.

When stitching only two images, the process was relatively straightforward, and the results were satisfactory. However, as we increased the number of images in the panorama (e.g., 4 images, 6 images), we observed distortion, particularly in the images near the edges. This distortion resulted from the cylindrical projection used for stitching.

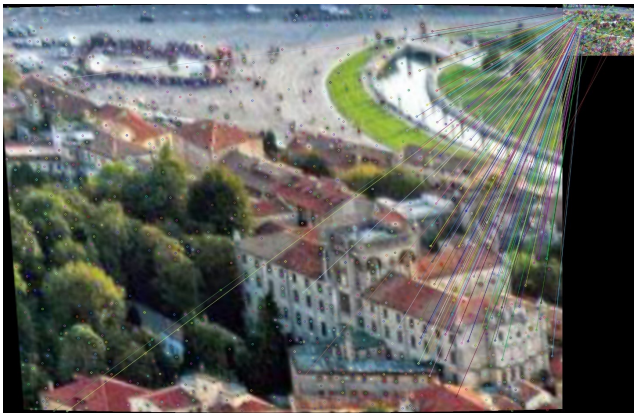

Figure 1. Batches Between the Main Image and 2nd Image. Keypoints from the main image are matched with keypoints from the second image.

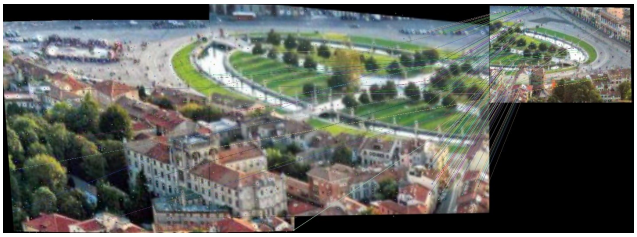

Figure 2. Batches Between the Main Image and 3rd Image. Keypoint matching across the main image and the third image shows reliable feature correspondence.

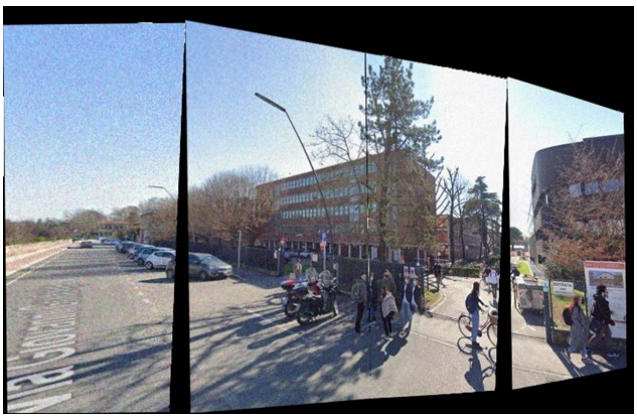

Figure 3. Distortion in the side image during the stitching process. The distortion is visible when stitching multiple images, particularly when the images are projected onto a cylindrical surface.

Despite this issue, the experiment showed that the program could handle the stitching of multiple images, although adjustments may be needed to improve image align-

ment and reduce distortion when dealing with wider panoramas.

4.4. Projection onto a Cylinder

The program used cylindrical projection for stitching the images together. This technique involves unrolling the images onto a cylindrical surface, which helps mitigate some of the distortion seen in wide panoramic images. However, the degree of distortion becomes more apparent when stitching multiple images.

For comparison, alternative projections, such as spherical projection, could be considered to improve results, especially in cases where more images are stitched together. The cylindrical projection worked well for smaller sets of images but showed limitations as the number of images increased.

4.5. Results of Image Stitching and Panoramas

The final results from the experiment were successfully computed using the SIFT keypoint detection and stitching pipeline. Below are the final panoramas generated from the selected datasets:

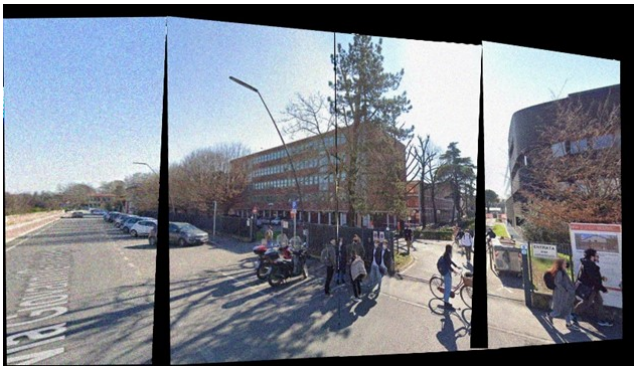

Figure 4. Panorama result of DEI street scene. The stitched image shows a clear representation of the street, combining all individual images seamlessly.

Overall, the results show that the system is capable of generating high-quality panoramas, with some limitations due to distortion when stitching a larger number of images.

5. Conclusion

The experiment demonstrated that the image stitching pipeline, utilizing SIFT feature detection and matching, was able to produce satisfactory panoramic images when stitching small sets of images. However, as the number of images increased, distortion became more prominent, especially in the side images.

The cylindrical projection worked well for smaller image sets but would benefit from further refinement when working with larger numbers of images. Future work could involve experimenting with spherical projections or implementing techniques to better handle edge distortions.

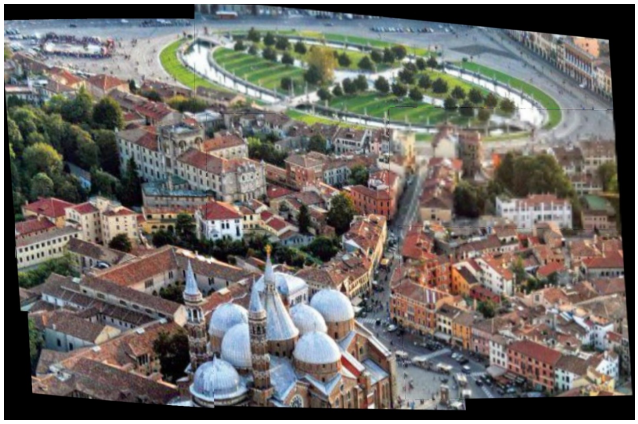

Figure 5. Panorama result of SRT1 dataset. A clear stitching of images from the SRT1 dataset is demonstrated, showing minimal artifacts.

Figure 6. Result from the RT2 dataset. A similar successful stitching process can be seen with minimal distortion.

References

Figure 7. Final result from the LNSRT3 dataset. The panorama demonstrates how well the images from this dataset were stitched together.