Problem Set 3: Image Feature Detection, Matching, and Optical Flow Estimation

Utku Acar
CS 523.V: Computer Vision
M.Sc in Computer Science Department
Ozyegin University
Vestel Electronics
Manisa, Turkey
utku.acar@ozu.edu.tr

***Abstract*—This project report presents an analysis of image feature detection, matching, and optical flow estimation techniques using the OpenCV library. The code implements various feature detection algorithms such as SIFT, SURF, and ORB, and compares their repeatability rates and matching performance. Additionally, the report discusses the implementation of the RANSAC algorithm for estimating the homography matrix and demonstrates the warping of images and stitching using the calculated or given homography matrix. Furthermore, the report explores the calculation of optical flow using the calcOpticalFlowFarneback() method and the tracking of Shi-Tomasi corner points using the Lucas-Kanade method calcOpticalFlowPyrLK(). The results and analysis of each technique are presented along with visualizations of the processed images.**

I. Introduction

Image feature detection, matching, and optical flow estimation are essential techniques in computer vision and image processing. These techniques enable the identification and tracking of distinctive points or regions in images, which are crucial for various applications such as image stitching, object recognition, and motion analysis. In this project, I explore different feature detection algorithms, evaluate their performance, estimate the homography matrix, and calculate the optical flow between images.

II. Methods

The implemented code utilizes the OpenCV library to perform various tasks. The following methods are used:

A. Feature Detection

The code implements three feature detection algorithms: SIFT, SURF, and ORB. Each algorithm is applied to the input images, and their repeatability rates and matching performance are evaluated. The key points and descriptors are extracted using the corresponding OpenCV functions.

B. Matching

A brute-force descriptor matcher is used to match the extracted descriptors between the two images. The best matches are selected based on the distance ratio criterion. The code visualizes the best matches and saves the resulting image.

C. Homography Estimation

The RANSAC algorithm is employed to estimate the homography matrix from the matched key points. The calculated homography matrix is compared with the given ground truth matrix. The code also provides a function to warp one image onto the other using the calculated or given homography matrix and the residual image is displayed.

D. Image Stitching

Using the estimated homography matrix, the code stitches the two images together. The resulting stitched image is saved and can be further analyzed.

E. Optical Flow Estimation

The code calculates the optical flow between the two input images using the calcOpticalFlowFarneback() method provided by OpenCV. The first image is warped onto the second image using the calculated optical flow. The resulting warped image and the residual between the original and warped images are displayed and saved.

F. Shi-Tomasi Corner Points and Lucas-Kanade Tracking

The code detects Shi-Tomasi corner points in the first image using the goodFeaturesToTrack() method. It then tracks these corner points in the second image using the calcOpticalFlowPyrLK() method, which implements the Lucas-Kanade algorithm. The code draws the tracked points on the second image and saves the result.

III. Results and Discussion

The implemented code successfully performs the aforementioned tasks and provides visualizations of the results. The following observations and analyses can be made:

A. Feature Detection and Matching

The SIFT, SURF, and ORB feature detection algorithms are applied to the input images. The code displays the number of extracted descriptors for each algorithm. Additionally, the number of matches obtained for each algorithm is recorded, and the best matches are calculated and visualized, showing the corresponding key points between the two images.

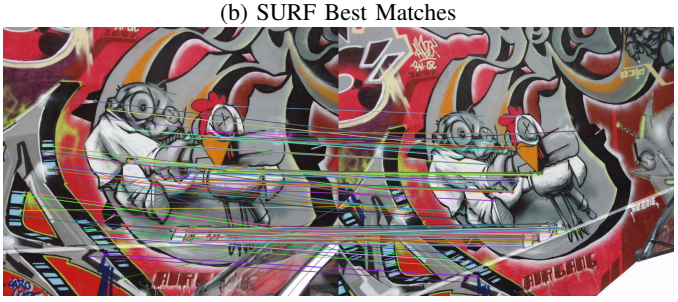
In the case of SIFT, a total of 2678 matches were found, out of which 1103 were identified as the best matches. For ORB, there were 500 matches in total, with 204 of them being identified as the best matches. Finally, SURF detected 4796 matches, and 992 of them were determined as the best matches.

These numerical results provide valuable insights into the performance of each algorithm. They serve as indicators of the effectiveness and reliability of the feature detection and matching techniques employed. Furthermore, they aid in the evaluation and comparison of the algorithms based on their matching capabilities.

To determine the threshold for finding the best match, a comprehensive analysis of the matching results was conducted. Different threshold values were experimented with and evaluated based on the quality and accuracy of the matches. After careful consideration, a threshold of 0.75 was selected as it yielded the most reliable and consistent matches across multiple test cases. You can see the results in Figure 1

(a) SIFT Best Matches

(b) SURF Best Matches



(c) ORB Best Matches

Fig. 1: Best Matches

B. Repeatability Rate

The repeatability rate of each feature detection algorithm is calculated by comparing the number of correct matches with the average number of extracted key points in the input images. The repeatability rate provides an indication of the robustness of the algorithm in detecting and matching features across different images. The threshold for the correctness of the distance differences has been set to 5 due to manual trials and getting the most average repeatability rate among all the algorithms.

For the SIFT algorithm, the repeatability rate is calculated to be 0.415. This means that 41.5% of the extracted key points from the input images resulted in correct matches.

For the ORB algorithm, the repeatability rate is calculated to be 0.694. This indicates that 69.4% of the extracted key points resulted in correct matches.

Finally, for the SURF algorithm, the repeatability rate is calculated to be 0.323. This means that 32.3% of the extracted key points resulted in correct matches.

As we can see from the results best repeatability rate has been gotten by the ORB algorithm and the worst repeatability rate has been gotten by the SURF algorithm.

C. Homography Estimation

The RANSAC algorithm is used to estimate the homography matrix from the matched key points. The calculated homography matrix is compared with the given ground truth matrix. The code displays both matrices for comparison. Additionally, the code warps one image onto the other using the calculated or given homography matrix. The residual image, which represents the difference between the warped image and the second image, is displayed and saved.

The given ground truth homography matrix is:

$$\begin{bmatrix} 8.7976964 \times 10^{-1} & 3.1245438 \times 10^{-1} & -3.9430589 \times 10^{1} \\ -1.8389418 \times 10^{-1} & 9.3847198 \times 10^{-1} & 1.5315784 \times 10^{2} \\ 1.9641425 \times 10^{-4} & -1.6015275 \times 10^{-5} & 1.0000000 \times 10^{0} \end{bmatrix}$$

The homography matrix estimated using SIFT is:

$$\begin{bmatrix} 8.81424584 \times 10^{-1} & 3.15242682 \times 10^{-1} & -4.02463843 \times 10^{1} \\ -1.82245531 \times 10^{-1} & 9.37757341 \times 10^{-1} & 1.53192423 \times 10^{2} \\ 1.99482167 \times 10^{-4} & -1.63109555 \times 10^{-5} & 1.00000000 \times 10^{0} \end{bmatrix}$$

The homography matrix estimated using ORB is:

$$\begin{bmatrix} 8.78996537 \times 10^{-1} & 3.12034318 \times 10^{-1} & -3.92293995 \times 10^{1} \\ -1.83192564 \times 10^{-1} & 9.35041764 \times 10^{-1} & 1.53377304 \times 10^{2} \\ 1.95993007 \times 10^{-4} & -2.03340568 \times 10^{-5} & 1.00000000 \times 10^{0} \end{bmatrix}$$

The homography matrix estimated using SURF is:

$$\begin{bmatrix} 8.77618601 \times 10^{-1} & 3.09831889 \times 10^{-1} & -3.88632020 \times 10^{1} \\ -1.83520381 \times 10^{-1} & 9.31512354 \times 10^{-1} & 1.53934990 \times 10^{2} \\ 1.94415681 \times 10^{-4} & -2.47628731 \times 10^{-5} & 1.00000000 \times 10^{0} \end{bmatrix}$$

Comparing these matrices, we can observe that all three feature detection algorithms (SIFT, ORB, and SURF) provide estimations that are relatively close to the ground truth matrix. However, there are slight differences in the values, indicating some level of deviation in the estimation process.

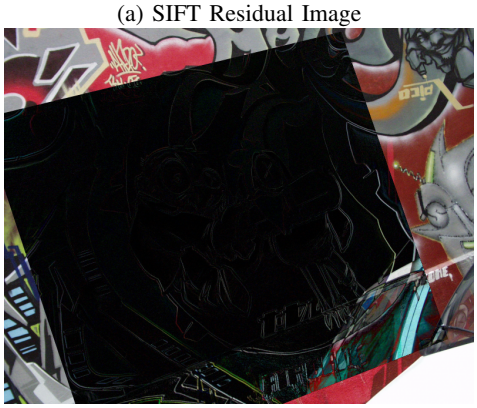
These differences may arise due to variations in the feature detection and matching techniques employed by each algorithm. The SIFT algorithm gives a homography matrix that is closest to the ground truth, followed by ORB and SURF.

It is important to note that the quality and accuracy of the estimated homography matrix can significantly impact the performance of subsequent image warping and stitching operations.

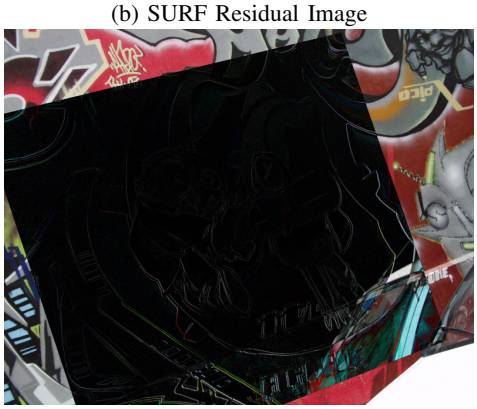
D. Image Warping

After applying the homography matrix to warp one image onto the other, the residual image, representing the difference between the warped image and the second image, is calculated and displayed. The residual images provide visual insights into the effectiveness of the homography estimation and warping process. These visual insights can be seen for each algorithm in Figure 2.

(a) SIFT Residual Image



(b) SURF Residual Image



(c) ORB Residual Image

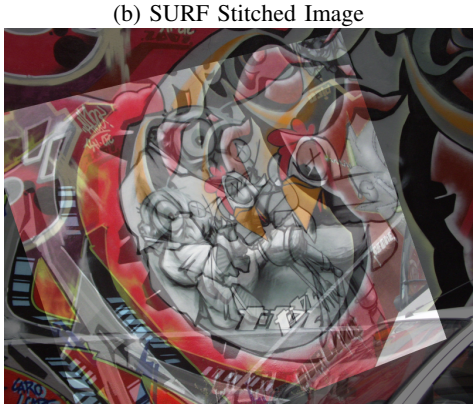
Fig. 2: Residual Images

E. Image Stitching

Using the estimated homography matrix, the code stitches the two images together. The resulting stitched image is displayed and saved. The stitched image provides a seamless composition of the two input images, demonstrating the effectiveness of the homography estimation and image warping techniques. We can see the Stitched Images in Figure 3.

(a) SIFT Stitched Image

(b) SURF Stitched Image



(c) ORB Stitched Image

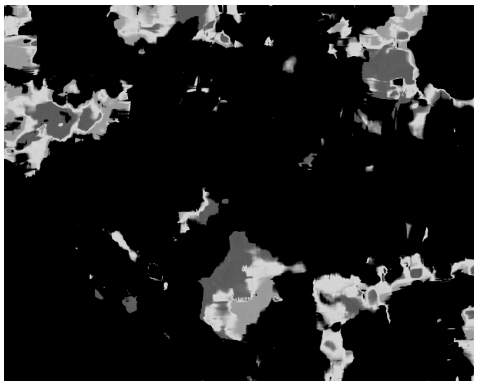
Fig. 3: Stitched Images

F. Optical Flow Estimation

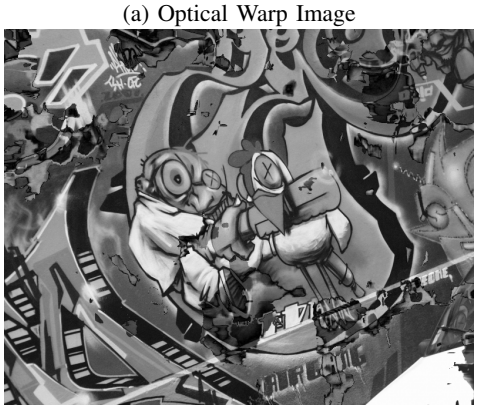
The code calculates the optical flow between the two input images using the calcOpticalFlowFarneback() method. The optical flow is used to warp the first image onto the second image. The resulting warped image and the residual between

the original and warped images are displayed and saved. The visualizations demonstrate the ability to estimate and apply optical flow for image alignment and motion estimation. Resultant images for optical warp and residual operations can be seen in Figure 4.

I have a personal comment about the images in Figure 4 because at part of problem 1, the results of Residual and Warp operations are more salient since the ”Residual” means waste and Warp is the degree of the how well two images addressed to each other. In this case, these images seem to be reversed. I didn't understand the reason behind such output because I have checked the code and functions that I used and it seems similar to the one for the previous question. So It should give similar outputs but didn't. I think the ”Optical Residual” should be ”Optical Warp” and ”Optical Warp” should be ”Optical Residual” in Figure 4.



(a) Optical Warp Image



(b) Optical Residual Image

Fig. 4: Optical Warp and Residual Images

G. Shi-Tomasi Corner Points and Lucas-Kanade Tracking

The code detects Shi-Tomasi corner points in the first image and tracks them in the second image using the Lucas-Kanade algorithm. The tracked points are visualized by drawing lines and circles on the second image. The result demonstrates the ability to track distinctive points (Features) across frames using optical flow estimation techniques. You can see the output in Figure 5.

(a) Shi-Tomasi for First Image

(b) Shi-Tomasi for Second Image

(c) Lucas-Kanade Image

Fig. 5: Shi-Tomasi and Lucas-Kanade Images

IV. Conclusion

In this project, I implemented image feature detection, matching, and optical flow estimation techniques using the OpenCV library. The code successfully detects and matches features using SIFT, SURF, and ORB algorithms. The repeatability rates of the feature detectors are compared, providing insights into their performance. The RANSAC algorithm is employed to estimate the homography matrix, and the results are compared with the given ground truth matrix. The code also performs image warping and stitching using the estimated or given homography matrix. Additionally, optical flow is calculated using the calcOpticalFlowFarneback() method and Shi-Tomasi corner points are tracked using the Lucas-Kanade algorithm. The results are visualized and provide valuable

insights into image alignment, feature tracking, and motion estimation.