Image Processing - Exercise 4

Amir Kelman, amir.kelman, 208288811

Introduction

The goal of this exercise is to implement a stereo mosaicing algorithm that creates a panoramic video from a sequence of frames, assuming significant overlap and left-to-right camera motion. The algorithm stitches frames into multiple panoramas, handling dynamic scenes (e.g., trees) or changing viewpoints (e.g., boat). Key techniques include:

- **Feature Detection and Description:** Identifying robust keypoints using SIFT.
- **Feature Matching:** Matching keypoints across frames using FLANN with Lowe's ratio test.
- **Geometric Transformation Estimation:** Computing homographies via RANSAC to align frames.
- **Image Warping and Stitching:** Warping frames to a common coordinate system and blending them into panoramas.

While feature detection and matching were successful, challenges in homography estimation and stitching limited the final mosaic quality.

Algorithm

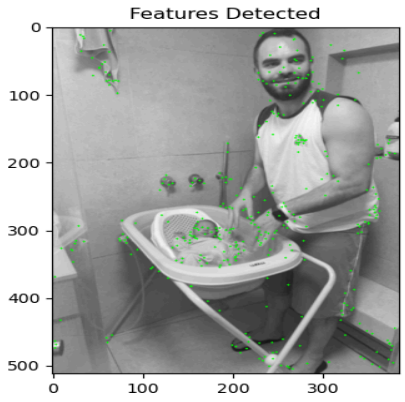
The stereo mosaicing pipeline processes a video to produce panoramic videos:

1. **Frame Extraction:** Extract frames from the input video (get_frames).
2. **Pairwise Homography Estimation:** Compute homographies between consecutive frames:
    - Build a Gaussian pyramid (build_gaussian_pyramid).
    - Detect features using SIFT on pyramid level 2 (find_features_SIFT).
    - Match features using FLANN (match_features_flann).
    - Estimate homography using RANSAC (ransac_homography).
3. **Homography Accumulation:** Compute global homographies relative to the middle frame (accumulate_homographies).

4. **Image Warping:** Warp frames to the reference frame's coordinate system (warp_image).
5. **Panorama Stitching:** Combine warped frames using a strip-based approach (build_panoramas) or weighted averaging (warp_all_images).

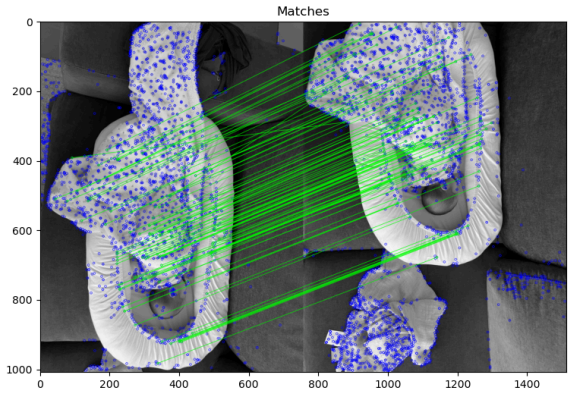
Sub-Algorithms & Visualization:

1. **Feature Detection (SIFT)**
    - **Input:** Grayscale image from pyramid level 2.
    - **Output:** Keypoint coordinates (N, 2) and descriptors (N, 7, 7).
    - **Visualization:** Figure 1 shows keypoints plotted on an image I took (e.g., from "First_shower.jpg").
    - **Explanation:** Green circles indicate detected keypoints, showing robust feature identification.



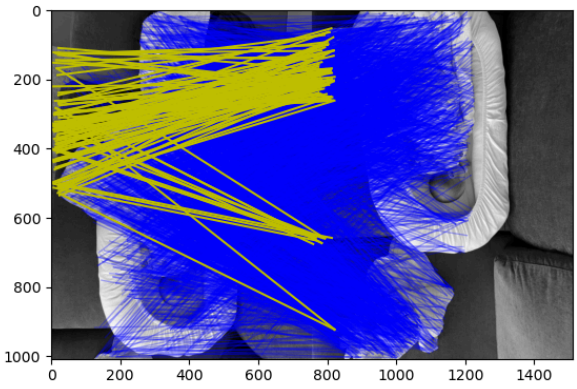
(Figure 1)

2. **Feature Matching (FLANN)**
    - **Input:** Descriptors from two consecutive images I took of my baby sleeping.
    - **Output:** Indices of matched keypoints.
    - **Visualization:** Figure 2 shows matches between images, with green lines for inliers (RANSAC) and blue dots for outliers.
    - **Explanation:** Matches demonstrate correspondence quality, with inliers indicating reliable pairs.



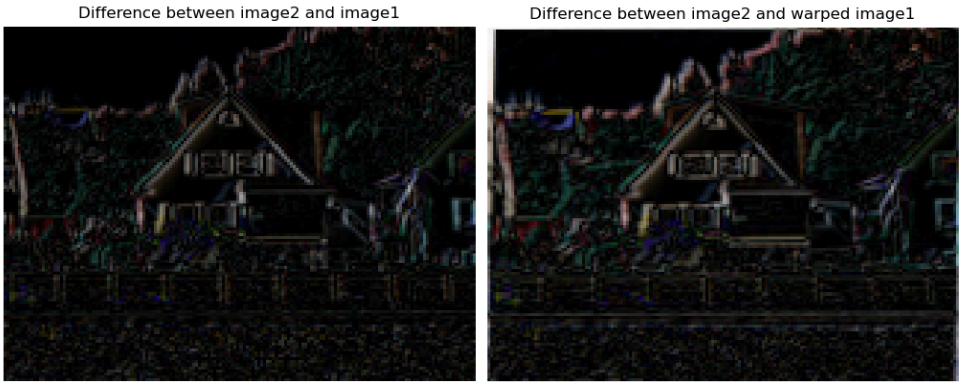
(Figure 2)

3. **Homography Estimation (RANSAC)**
    - **Input:** Matched keypoint coordinates.
    - **Output:** 3x3 homography matrix and inlier indices.
    - **Visualization:** Figure 3 shows inliers (yellow) vs. outliers (blue) on stacked frames.
    - **Explanation:** Inliers represent points consistent with the estimated transformation, but errors in homography estimation caused misalignment.



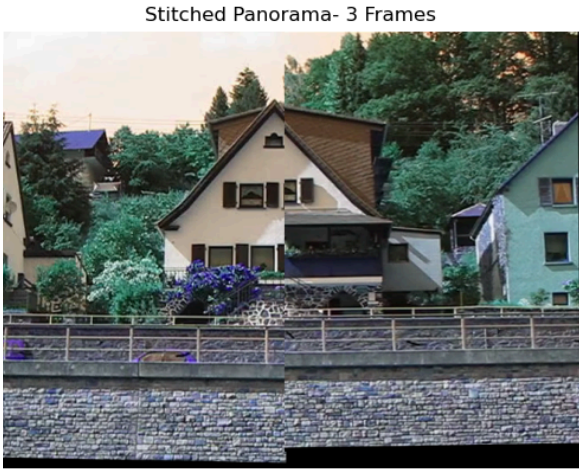
(Figure 3)

4. **Warping**
    - **Input:** Frame and homography matrix.
    - **Output:** Warped frame.
    - **Visualization:** Figure 4 shows the difference between the warped frame and target frame vs. original frames.
    - **Explanation:** Larger differences in the warped case indicate homography inaccuracies.



(Figure 4)

5. **Stitching**
    - **Input:** Warped frames and homographies.
    - **Output:** Panorama.
    - **Visualization:** Figure 5 shows a 3-frame panorama attempt.
    - **Explanation:** Misalignments highlight stitching challenges due to accumulated homography errors.



(Figure 5)

Implementation Details

Approach

The implementation follows the described pipeline, with key components:

- **SIFT**: Applied on Gaussian pyramid level 2 for efficiency.
- **FLANN**: Used with Lowe's ratio test (min_score=0.5) for robust matching.
- **RANSAC**: Configured with 5000 iterations and 2px inlier tolerance.
- **Stitching**: Attempted both strip-based and weighted-average methods.

Libraries

- **OpenCV**: Used for SIFT (cv2.SIFT_create), FLANN (cv2.FlannBasedMatcher), RANSAC (cv2.findHomography), and warping (cv2.warpPerspective).
- **Custom**: Implemented homography accumulation, strip boundary computation, and panorama assembly.

Hyperparameters

- **SIFT**: Default parameters; pyramid level 2 balances speed and detail.
- **FLANN**: min_score=0.5 filters weak matches; k=2 enables ratio test.
- **RANSAC**: 5000 iterations ensure robust estimation; 2px tolerance suits image resolution.
- **Reference Frame**: Middle frame minimizes error accumulation.

Challenges

1. **Homography Errors**: Inaccurate homographies led to misalignment (see Figure 4). Debugged by visualizing differences between warped and target frames.
2. **Panorama Drift**: Accumulated errors over many frames caused visible distortions in 100-frame attempts.
3. **Strip-Based Stitching**: Incorrect strip boundary calculations resulted in incomplete panoramas. Debug prints helped identify coordinate issues.

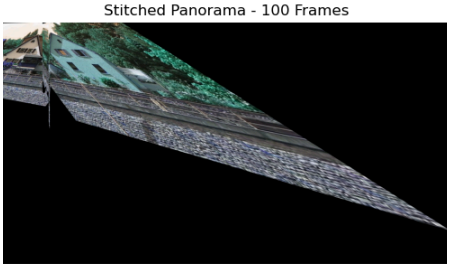
Visual Results

Provided Videos

1. **Dynamic Mosaic (Trees)**
    - **Input**: "trees.mp4" (Moodle).
    - **Results**: Figure 6 shows the first, middle, and last frames of the attempted mosaic.
    - **Explanation**: Misalignments are evident due to homography errors, but feature matching was robust.
    - **Flaws**: Seams and distortions indicate stitching failures.

(Figure 6)

2. **Changing Viewpoint Mosaic (Boat)**
    - **Input**: "boat.mp4" (Moodle).
    - **Results**: Figure 7 shows the first, middle, and last frames of a 3-frame panorama.
    - **Explanation**: Partial alignment shows warping functionality, but inaccuracies persist.
    - **Flaws**: Visible seams and incorrect blending.



(Figure 7)

Conclusion

This exercise provided practical experience with stereo mosaicing, reinforcing key concepts:

- **Feature Matching**: SIFT and FLANN effectively identified correspondences, validated by visualizations.
- **Homography Estimation**: RANSAC's limitations were evident in error-prone cases, requiring careful tuning.
- **Stitching Challenges**: Accumulated errors highlighted the need for robust error correction.

Future improvements could include:

- Multi-frame homography optimization to reduce drift.
- Advanced blending techniques to minimize seams.
- Adaptive thresholding for feature matching in dynamic scenes.

While the final results showed limitations, the debugging process revealed critical insights about the algorithms we learned come into practice and how they work on real images. The exercise highlighted that even theoretically sound algorithms require careful parameter tuning and validation to work reliably in practice.