Panorama Stitching

Doriya Spielman - 313466625
Kineret Ruth Nahary - 206903684
Levi Dworkin - 204807044
Github : https://github.com/Kineruth/Panorama_Stitching

Introduction

What is the problem?

Image stitching is the process of combining multiple photographic images with overlapping fields of view to produce a segmented *panorama* or high-resolution image. The most common approaches to image stitching require exact overlaps between images and identical exposures to produce seamless results.
The human visual system has a field of view of around 135 x 200 degrees, but a typical camera has a field of view of only 35 x 50 degrees. Therefore, panoramic image mosaicing works by taking lots of pictures from an ordinary camera and stitching them together to form a composite image with a much larger field of view.
The quality of image stitching is measured by the similarity of the stitched image to each of the input images. It also can be measured by the visibility of the seam between the stitched images.

Approach and Method

1. **Registration**
    - ***Feature Extraction (SIFT):***
      For each pair of consecutive frames in our image sequence, we use scale-invariant feature transform (SIFT) to detect and describe local features between them. The SIFT enables reliable image matching with various orientation and zoom. The basic steps of the extraction algorithm are Scale-space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor.

    - ***Feature matching (BF):***
      After finding all the feature points in both images, we use Brute-Force Matcher to match those feature points to each other.
      The algorithm runs over each keypoint K_1 from the first set (image) and takes every keypoint in the second set and calculates the distance. The keypoint K_2 with the smallest distance will be considered its pair. The algorithm returns a set of all matched keypoints.

    - ***Registering the transformation****:*
      When matching the SIFT feature points using BF, there will be lots of mismatches. The RANSAC algorithm can be used to remove the mismatches by finding the transformation matrix of these feature points.
      RANSAC (RANdom SAmple Consensus) is a non-deterministic algorithm

because it doesn't ensure to return acceptable results. It is used to estimate parameters for the Homography of a mathematical model from a set of observed data that contains outliers iteratively.
RANSAC loop involves:
***** selects four random feature pairspoint.
***** computes their homography H using least square (SVD composition).
***** computes estimated points of the second frame using the homography that was calculated.
***** computes inliers - if $E_j = \| P'_{1,j} - P_{2,j} \| < inlierTol$ where $j = 1..N \ and \ inlierTol = some\ constant\ threshold$, then it is added as an inlier.
***** keeps the largest set of inliers and its homography.
When finished, RANSAC will return the maximum set of inliers found between each pair of frames and the transformation matrix of its feature points (H).
We chose $inlierTol = 10$ because we saw it gives a good amount of inliers sets - quite a bit and not much either.

2. Panorama Stitching

- ***Transforming into a common coordinate system:***
  We need to choose a coordinate system in which we would like the panorama to be rendered. We chose it to be the coordinate system of the middle frame I_m in our image sequence and transformed all other frames to that coordinate system.
  The resulting panorama image will have a frame I_m coordinate system and be composed of all the transformed frames back-warped so that they properly align with it.

- ***Rendering the panorama:***
  So far, we have H_{tot} that is a set of M homographies $\tilde{H}_{i,m}$ where $i = 1..M \ and \ M \ is \ the \ amount \ of \ frames,$ that transforms pixel coordinates in frame I_i to the panorama coordinate system.
  First, we need to define where we want the panorama image I_{pano} to be rendered. We would like this region to be large enough to include all pixels from all frames. For that, we define which parts of I_{pano} should be obtained from each frame I_i.
  We divide the panorama to M vertical strips and finally, we Back-Warp the images on the strips of the panorama using H_{tot}.
  We did not succeed implementing the way was instructed, so we used opencv function *wrapPerspective*.

Results

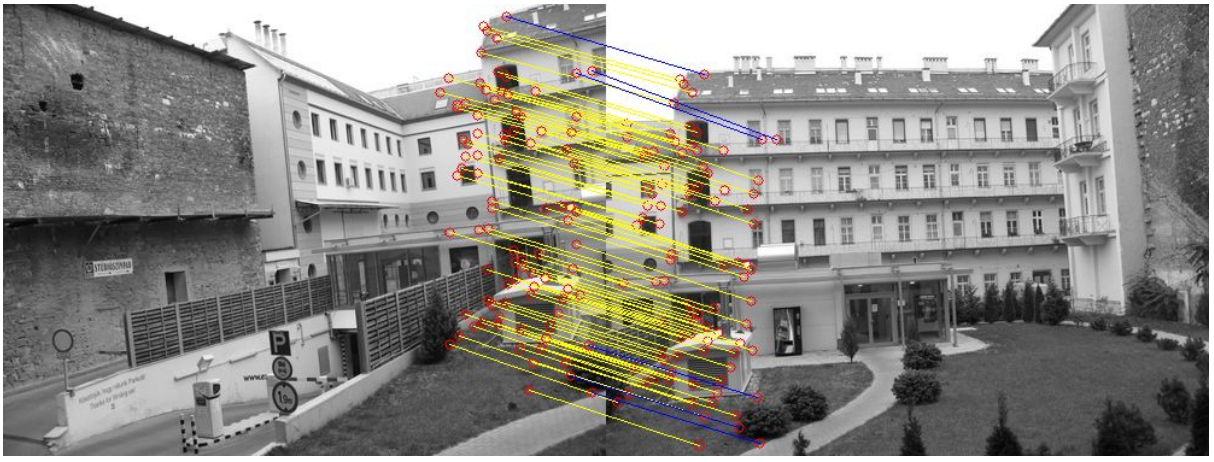
The experiments we have done:

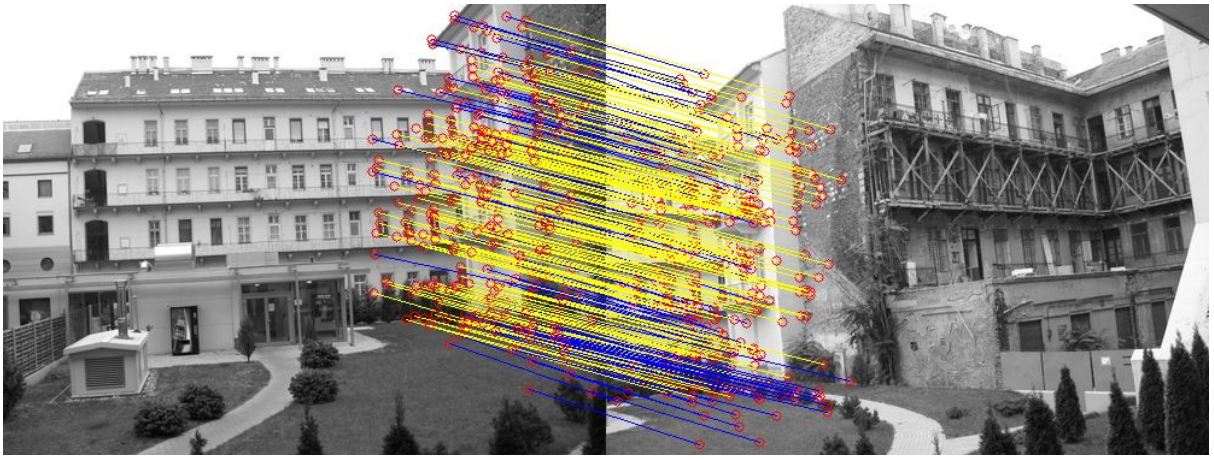
- ***findFeatures -*** we debated between SURF and SIFT algorithms and decided to go with SIFT because people say it is more used for implementing finding features in image stitching.
- ***matchFeatures -*** at first we matched descriptor vectors with a Flann based Matcher, but saw that it does not return accurate matched values, so we switched to Brute-Force Matcher and it gave much more accurate results.
- ***Choosing threshold (inlierTol) -*** we ran our implementation on our input pictures:
  * for threshold = 5 → got around 115 inliers
  * for threshold = 10 → got around 110~120 inliers
  * for threshold = 20 → got around 120~130 inliers

  And so we used threshold=10 because it looked the most accurate for the eye.

Our final results:

- **DisplayMatches images:**





- **Panorama Examples folder output:**

- **Panorama Mine folder output:**