Computer Vision and Image Processing (CSE 573)

Shubham Sharma, Person No.: 50290293, UBIT Name: ss628

April 10, 2019

Project 2: Image Stitching

The following approach has been followed to perform the given task:

- Detect Keypoints using *ORB* in *OpenCV*
- Extract Local Invariant Descriptors using *ORB*
- Match the keypoints and descriptors: Filter out the pair of keypoints between any two images using the euclidean distance between the descriptors of those keypoint pairs and choose the pairs of points whose distances are less than a threshold.
- To determine the sequence of images, we see which pair of images has the minimum number of match keypoints. As is apparent, the images at the extreme positions will have the minimum number of match keypoints. Using the average x-coordinate of the keypoints, we detemine which image is to the left and the right of the middle image.
- To find the homography for each image pair, we randomly select four matched keypoints for the image pair and solve the transformation matrix to get a 3x3 Homography matrix as shown in figure 3.
- Use RANSAC (Random Sample Consensus) to estimate the best homography matrix for each image pair using the matched feature vectors
- Apply a warping transformation using the homography matrix obtained in the previous step and the warpPerspective function of OpenCV.

Oriented FAST and rotated BRIEF (ORB)

ORB (Oriented FAST and rotated BRIEF) algorithm is used for keypoint detections and feature description. The reason this algorithm is used is because it is faster that *SIFT* (Scale Invariant Feature Transform) as well as not patended and is available in *OpenCV*. Also, although *SIFT* has proven to be very efficient in object recognition applications, it requires a large computational complexity which is a major drawback especially for real-time applications. On the other hand, *ORB* is an efficient alternative which requires less complexity than *SIFT* with almost similar matching performance.

ORB is a fusion of the FAST key point detector and BRIEF descriptor with some modifications. Initially to determine the key points, it uses FAST. Then a Harris corner measure is applied to find top N points. FAST does not compute the orientation and is rotation variant. It computes the intensity weighted centroid of the patch with located corner at center. The direction of the vector from this corner point to centroid gives the orientation. Moments are computed to improve the rotation invariance. The descriptor BRIEF poorly performs if there is an in-plane rotation. In ORB, a rotation matrix is computed using the orientation of patch and then the BRIEF descriptors are steered according to the orientation.

FAST features are widely used because of their computational properties. However, FAST features do not have an orientation component. FAST takes one parameter, the intensity threshold between the center pixel and those in a circular ring about the center. FAST does not produce a measure of cornerness, and it has large responses along edges. A Harris corner measure is employed to order the FAST keypoints. For a target number N of keypoints, we first set the threshold low enough to get more than N keypoints, then order them according to the Harris measure, and pick the top N points. FAST does not produce

multiscale features. We employ a scale pyramid of the image, and produce FAST features (filtered by Harris) at each level in the pyramid. Basically, it computes the intensity weighted centroid of the patch with located corner at center. The direction of the vector from this corner point to centroid gives the orientation. To improve the rotation invariance, moments are computed with x and y which should be in a circular region of radius r, where r is the size of the patch.

BRIEF (Binary Robust Independent Elementary Features) is a recent feature descriptor that uses simple binary tests between pixels in a smoothed image patch to train a set of classification trees. Once trained on a set of 500 or so typical keypoints, the trees can be used to return a signature for any arbitrary keypoint. In a similar manner, we look for the tests least sensitive to orientation. The classic method for finding uncorrelated tests is Principal Component Analysis; for example, it has been shown that PCA for SIFT can help remove a large amount of redundant information. However, the space of possible binary tests is too big to perform PCA and an exhaustive search is used instead.

BRIEF performs poorly with rotation thus we steer BRIEF according to the orientation of keypoints. For any feature set of n binary tests at location (x_i, y_i), define a $2 \times$ n matrix, S which contains the coordinates of these pixels. Then using the orientation of patch, θ, its rotation matrix is found and rotates the S to get steered(rotated) version S_θ.

ORB discretize the angle to increments of $2\ \pi\ /30$ (12 degrees), and construct a lookup table of precomputed BRIEF patterns. As long as the keypoint orientation θ is consistent across views, the correct set of points S_θ will be used to compute its descriptor.

BRIEF has an important property that each bit feature has a large variance and a mean near 0.5. But once it is oriented along keypoint direction, it loses this property and become more distributed. High variance makes a feature more discriminative, since it responds differentially to inputs. Another desirable property is to have the tests uncorrelated, since then each test will contribute to the result. To resolve all these, ORB runs a greedy search among all possible binary tests to find the ones that have both high variance and means close to 0.5, as well as being uncorrelated. The result is called rBRIEF.

For descriptor matching, multi-probe LSH which improves on the traditional LSH, is used. The paper says ORB is much faster than SURF and SIFT and ORB descriptor works better than SURF. ORB is a good choice in low-power devices for panorama stitching etc.

Homography

A homography is a perspective transformation of a plane, that is, a reprojection of a plane from one camera into a different camera view, subject to change in the translation (position) and rotation (orientation) of the camera.

Perspective transformations map 3-D points onto 2-D image planes using the transformation matrix that incorporates the camera characteristics: focal length, optical centre, and the extrinsic parameters (rotation, translation) .

RANSAC (Random Sample Consensus)

This algorithm is used to avoid the impact of outliers by looking for the inliers and using them only because if we choose an outlier to compute the current fit, then the resulting line won't have much support from the rest of the points.

It is a resampling technique that generates candidate solutions by using the minimum number observations (data points) required to estimate the underlying model parameters. Unlike conventional sampling techniques that use as much of the data as possible to obtain an initial solution and then proceed to prune outliers, RANSAC uses the smallest set possible and proceeds to enlarge this set with consistent data points.

First, we select randomly the minimum number of points required to determine the model parameters. Then, we solve for the parameters of the model and determine how many points from the set of all points fit with a predefined tolerance. If the fraction of the number of inliers over the total number points in the set exceeds a predefined threshold, we reestimate the model parameters using all the identified inliers and terminate otherwise, we repeat the same steps unless we get the fraction of the number of inliers over the total number points in the set more than a predefined threshold.

Results

The stitched image for the given data is shown in figure 1.

The stitched image for the pictures takes of UB is shown in figure 2. The pictures are about REVOLUTION by the office of Inclusive Excellence and the Universities Libraries. The pictures are taken on the first floor of Silverman library.

All the images are available in the folder Results.

Figure 1: Stiched Image for the nevada pictures provided

Figure 2: Stiched Image for the UB pictures taken

$$\underline{A}\,\underline{h} = \underline{b}$$

$$\underline{A} = \begin{bmatrix}
b_{u_1} & b_{v_1} & 1 & 0 & 0 & 0 & -b_{u_1}a_{u_1} & -b_{v_1}a_{u_1} \\
0 & 0 & 0 & b_{u_1} & b_{v_1} & 1 & -b_{u_1}a_{v_1} & -b_{v_1}a_{v_1} \\
b_{u_2} & b_{v_2} & 1 & 0 & 0 & 0 & -b_{u_2}a_{u_2} & -b_{v_2}a_{u_2} \\
0 & 0 & 0 & b_{u_2} & b_{v_2} & 1 & -b_{u_2}a_{v_2} & -b_{v_2}a_{v_2} \\
b_{u_3} & b_{v_3} & 1 & 0 & 0 & 0 & -b_{u_3}a_{u_3} & -b_{v_3}a_{u_3} \\
0 & 0 & 0 & b_{u_3} & b_{v_3} & 1 & -b_{u_3}a_{v_3} & -b_{v_3}a_{v_3} \\
b_{u_4} & b_{v_4} & 1 & 0 & 0 & 0 & -b_{u_4}a_{u_4} & -b_{v_4}-a_{u_4} \\
0 & 0 & 0 & b_{u_4} & b_{v_4} & 1 & -b_{u_4}a_{v_4} & -b_{v_4}a_{v_4}
\end{bmatrix}$$

$\underline{h} = (q, w, e, r, t, y, u, i)$ – Homography matrix values

$\underline{b} = (a_{u_1}\ a_{v_1}\ a_{u_2}\ a_{v_2}\ a_{u_3}\ a_{v_3}\ a_{u_4}\ a_{v_4})$

Figure 3: Homography Computation (a: initial point, b: final point, u: x-coordinate, v: y-coordinate)