ENGN1610 & 2605 Image Understanding

Lab06: Panorama / Image Stitching

The goal of this lab is to

- Learn how to estimate a homography matrix from four feature correspondences.
- Learn how to find a reliable homography matrix from noisy feature correspondences using RANSAC, and apply it to an image stitching application.

Problem 1. Image Stitching From Two Views Implement an image stitching algorithm that can piece a pair of images together based on feature correspondences. The view changes can be either a change in translation, Figure 1(a), a change in rotation, Figure 1(b), or even a change in time like the Providence's City Hall, Figure 1(c). The necessary steps to follow in this question are:

1. **Feature Detection and Descriptor Extraction:** After loading images in Figure 1, detect SIFT features and extract its corresponding descriptors from each image using VLFeat as you did in the last lab.
2. **Feature Matching:** Choose one image as the reference frame and find the feature correspondences based on the similarities between feature descriptors the same way you did in the last lab. Make sure to apply Lowe's ratio test to discard ambiguous correspondences.
3. **RANSAC for Homography Matrix Esimtation:** Create the following function that performs a RANSAC algorithm to estimate the homography matrix that transforms features from the reference image to another:

   ```
   function finalH = Ransac4Homography(matches1, matches2)
   ```

   The inputs `matches1` and `matches2` are the lists of matched feature pixel coordinates of the two images, while the output `finalH` is the final homography matrix used for image stitching. Specifically, what you should do in this function is:

   (a) Pick 4 random candidate correspondences from the set `matches1` and `matches2` and compute the homography matrix.

   (b) Using this homography matrix, transform all features from the reference image to another image as you did in the previous lab.

   (c) Count the number of inliers after homography transformation, *i.e.*, a candidate correspondence is considered as an *inlier* if the Euclidean distance between the transformed coordinate and the matched coordinate is less than some small threshold, *e.g.*, 1 pixel.

(d) Repeat steps (a) to (c) under an iterative loop with a predefined number of iterations. Set the number of iterations at least 1000.

(e) Finally, after the iterative loop ends, pick the homography matrix estimation with the maximum number of inliers as your output `finalH` (final homography). Report your inlier ratio, *i.e.*, the number of inliers over the number of all correspondences.

(f) It is optional but preferable to visualize your feature matching result with inlier and outlier correspondences.

4. **Stitch Two Images:** Using the final homography matrix having the maximum number of inliers, stitch two images using the provided MATLAB function file `getNewImg.m`. Follow the instructions in that file to determine what you should give as the inputs.

5. **Blend Stitched Image:** Often times, there are exposure differences between the images which result in visible seams when they are stitched. One way to mitigate these seams is to apply a blending algorithm. Use the provided MATLAB function file `blendImgs.m` to make a seamless stitched image. Follow the instructions in that file to determine what you should give as the inputs. Try different blending algorithms (specified by the input as a string argument) and see how the resultant stitched image looks like.

6. **Visualization:** Finally, show the stitched image. A few examples are given in Figure 2

Figure 1: (a) Golden Gate. (b) LEMS lab. (c) Providence City Hall (1900 v.s. 2016).

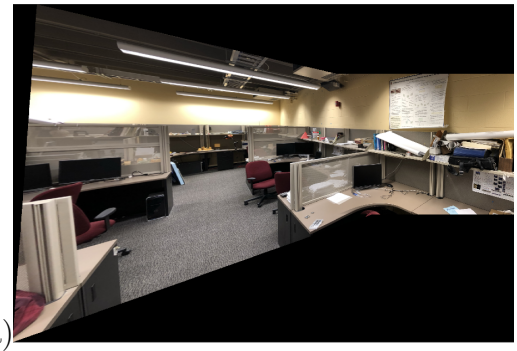
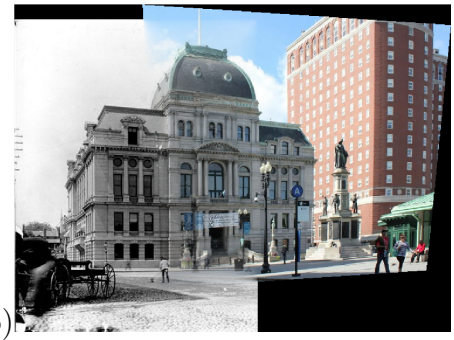
(a)  (b) 

Figure 2: Stitched image examples. (a) LEMS Lab taking the right image as the reference image. (b) Providence City Hall taking the left image as the reference image.

Problem 2. Image Stitching From Triplet Views Extend your implementation in problem 1, stitch triplets of images in Figure 3 together. One way to do is to stitch a pair of images first and then treat the resultant panaroma as the reference image to stitch with the third image.

(a) 

(b) 

Figure 3: (a) Half Dome. (b) Hotel.

Problem 3. Answer the Following Questions (This part is mandatory for ENGN2605 students but is optional for ENGN1610 students who will get 5 extra points per question if answered correctly.)

Question 1: In the lecture, we say that a pair of feature correspondence can be used to estimate a translation between two images. For N pairs of feature correspondences, *e.g.*, $N = 100$, the translation can be estimated by a standard least-square solution which minimizes the sum of N squared translation errors. Likewise, four pairs of feature correspondences can be used to estimate a homography matrix, and for N pairs of feature correspondences

we can also use a least-square solution to find the homography matrix that minimizes the sum of squared transformed errors. However, instead of using a least-square solution, we are using RANSAC to do homography estimation in problem 1 and 2. Why is that?

Question 2: Four pairs of feature correspondences give you an estimate of a homography matrix. Are *any* arbitrary four correspondences that are co-planar in 3D enabling us to successfully find a homography matrix? (Hint: Think about what scenario makes the matrix A in $Ah = 0$ becomes non-full rank. h is the vector representation of the homography matrix. Notations are borrowed from the lecture.)

Problem 4. Try Other Features (This part is optional for all students who will get 5 extra points if it is finished correctly.)

Experiment with different features and descriptors other than SIFT, *e.g.*, corners and SSD/NCC, and perform the image stitching algorithm you have implemented in problem 1. How well do they perform in terms of image stitching? Do they have any advantage over SIFT features in terms of homography matrix estimation? Discuss/Explain your results.