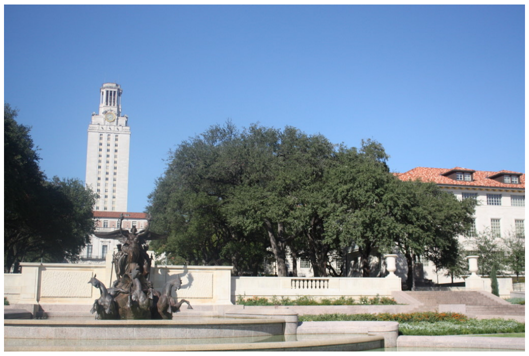
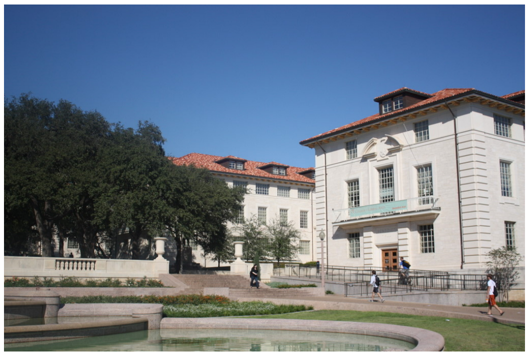
ECS 174, Spring 2022

Homework 2: Image stitching

Due date: 05/13/2022, 2PM

The goal of this assignment is to implement robust homography estimation to stitch pairs of images together. We provide some helper code in MATLAB in the text. You may find similar code for Python in utils.py

You will be working with the following pair (click on the images to download the high-resolution versions):

1. [5 points] Load both images, convert to double and to grayscale.

2. [5 points] Detect feature points in both images. You can use harris.m for Harris corner detection.

3. [10 points] Extract local neighborhoods around every keypoint in both images, and form descriptors simply by "flattening" the pixel values in each neighborhood to one-dimensional vectors. Experiment with different neighborhood sizes to see which one works the best.

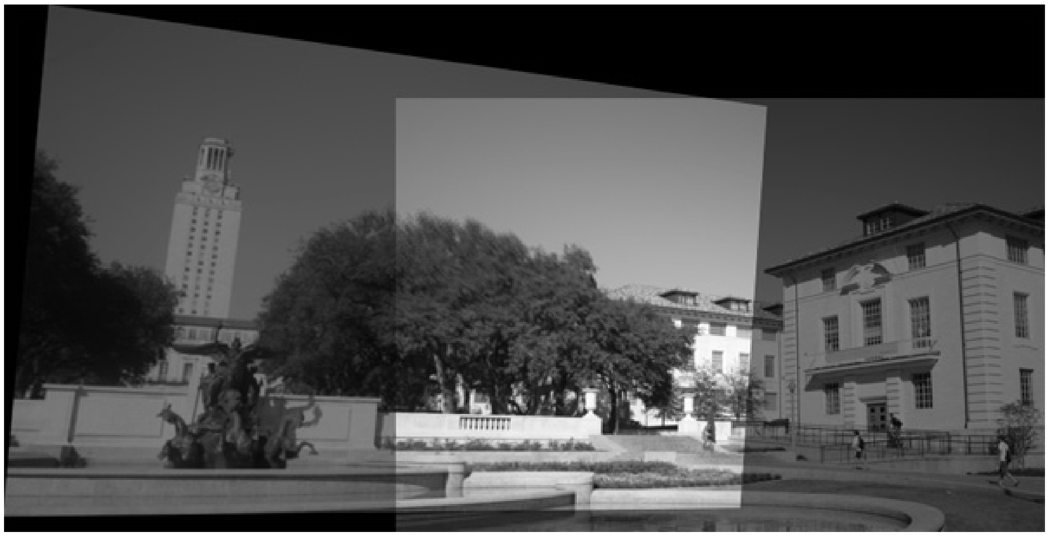
4. [10 points] Compute distances between every descriptor in one image and every descriptor in the other image. You can use dist2.m for fast computation of Euclidean distance. Alternatively, experiment with computing normalized correlation, or Euclidean distance after normalizing all descriptors to have zero mean and unit standard deviation. Optionally, feel free to experiment with SIFT descriptors. Here is some find_sift.m for computing SIFT descriptors of circular regions.

5. [10 points] Select putative matches based on the matrix of pairwise descriptor distances obtained above. You can select all pairs whose descriptor distances are below a specified threshold, or select the top few hundred descriptor pairs with the smallest pairwise distances.

6. [40 points] Run RANSAC to estimate a homography mapping one image onto the other. You need to weite the code for RANSAC by yourself. Report the number of inliers and the average residual for the inliers (squared distance between the point coordinates in one image and the transformed coordinates of the matching point in the other image). Also, display the locations of inlier matches in both images.

7. [10 points] Warp one image onto the other using the estimated transformation. To do this, you will need to learn about `maketform` and `imtransform` functions in MATLAB or similar functions in Python. You can use off-the-shelf code for this transformation (warping).

8. [10 points] Create a new image big enough to hold the panorama and composite the two images into it. You can composite by simply averaging the pixel values where the two images overlap. Your result should look something like this (but hopefully with a more precise alignment):



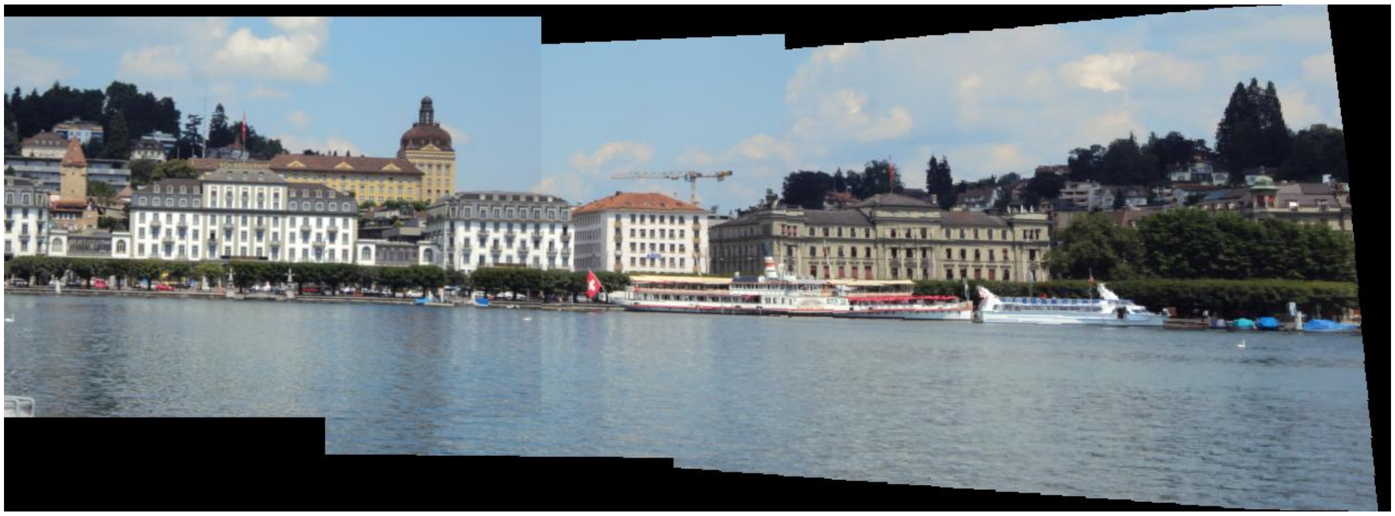
You should create color panoramas by applying the same compositing step to each of the color channels separately (for estimating the transformation, it is sufficient to use grayscale images).

Tips and Details

- For RANSAC, a very simple implementation is sufficient. Use four matches to initialize the homography in each iteration. You should output a single transformation that gets the most inliers in the course of all the iterations. For the various RANSAC parameters (number of iterations, inlier threshold), play around with a few "reasonable" values and pick the ones that work best. For randomly sampling matches, you can use the `randperm` or `randsample` functions in MATLAB.
- In MATLAB, the solution to a nonhomogeneous linear least squares system `AX=B` is given by `X = A\B, so you do not need to code pseudo-inverse by yourself.`
- Homography fitting calls for homogeneous least squares (AX=0). The solution to the homogeneous least squares system AX=0 is obtained from the SVD of A by the singular vector corresponding to the smallest singular value. In MATLAB: `[U,S,V]=svd(A); X = V(:,end);`

For extra credit

- [10 points] Extend your homography estimation to work on multiple images. You can use **this data**, consisting of three sequences consisting of three images each. For the "pier" sequence, sample output can look as follows (although yours may be different):



Source of data and results: Arun Mallya

Alternatively, feel free to acquire your own images and stitch them.

- [10 points] Experiment with registering very "difficult" image pairs or sequences -- for instance, try to find a modern and a historical view of the same location to mimic the kinds of composites found here. Or try to find two views of the same location taken at different times of day, different times of year, etc. Another idea is to try to register images with a lot of repetition, or images separated by an extreme transformation (large rotation, scaling, etc.). To make stitching work for such challenging situations, you may need to experiment with alternative feature detectors and/or descriptors, as well as feature space outlier rejection techniques such as Lowe's ratio test.

Instructions for turning in the assignment

Please submit your PDF report and zipped code archive, named **lastname_firstname_hw2.pdf** and **lastname_firstname_hw2.zip**.

Acknowledgements

This homework was developed by Lana Lazebnik at UIUC.