Due: 28th April 2021 2:00 PM Total points: **50**

In this project you will form a panoramic image from a set of photos captured at different perspectives.

1 Restricted Functions and Installation

You may use numpy, cv2, os, matplotlib, and scipy to help complete the assignment.

2 Intro to homographies

15 points

The assignment contains three images; 1.jpg, 2.jpg, and 3.jpg. Using `cv2.warpPerspective` and three different homography matrices, rotate 1.jpg clockwise 10 degrees, translate 2.jpg 100 pixels right, and shrink 3.jpg by half. Use the last argument to warpPerspective to specify that the output image should be 1000 pixels wide and 800 pixels tall.

3 Panoramic Stitching

40 points total

In this section you will use SIFT features and the RANSAC algorithm to determine two homographies that align images 1.jpg and 3.jpg with image 2.jpg. You will then apply these homographies and fuse the images.

3.1 Compute SIFT features

5 points

Compute the SIFT features for each of the images.

3.2 Match features

15 pts

For each SIFT feature in image 2.jpg find the SIFT features in images 1.jpg and 3.jpg that are the most similar, in terms of ℓ_2 distance. The `distance_matrix` command in `scipy.spatial` may prove useful here. You will now have two sets of matched features, but these sets will contain many false positives.

Get rid of all but the 100 best matches (again in an ℓ_2 sense) between image 2.jpg's features and image 1.jpg's features. Similarly, get rid of all but the 100 best matches between image

2.jpg's features and image 3.jpg's. You will be left with 2 sets of match features; the first cleary show up in images 1 and 2 and the second clearly show up in images 2 and 3.

3.3 Estimate the homographies

10 pts

Use the `cv2.findHomography` command to apply RANSAC to find a homography mapping image 1.jpg's matched features to image 2.jpg's. Similarly find a homography mapping image 3.jpg's matched features to image 2.jpg's. When applying RANSAC, declare a set of matched features inliers if the reprojection error (the difference between where the homography places a feature and its true location) is less than 2 pixels.

3.4 Warp and translate images

10 pts

Using a single homography matrix (made of the product of two other matrices), use the `cv2.warpPerspective` command to align image 1.jpg with image 2.jpg and then translate it 350 pixels to the right and 300 pixels down.
Using a single homography matrix (made of the product of two other matrices), use the `cv2.warpPerspective` command to align image 3.jpg with image 2.jpg and then translate it 350 pixels to the right and 300 pixels down.
Using a single homography matrix, use the `cv2.warpPerspective` command to translate image 2.jpg 350 pixels to the right and 300 pixels down.
Fuse the three images using `np.maximum`. While some minor ghosting artifacts (e.g., people walking) are expected, the details around the edges of the three images should align quite well.

Submission Instructions

Your canvas submission should consist of a zip file named **YourDirectoryID_Project4.zip**, for example xyz123_Project4.zip. The file must contain the following:

- Project4.py, *not .ipynb*
- report.pdf
- set1/ (directory containing the three images)

Collaboration Policy

You are encouraged to discuss ideas with your peers. However, the code should be your own and should represent your understanding of the assignment. **Code should not be shared or copied.** If you reference anyone else's code in writing your project, you must properly cite it in your code (in comments) and in your report.

Please list any individuals you collaborated with at the end of your report.

Plagiarism

Plagiarism of any form will not be tolerated. You are expected to credit all sources explicitly. If you have any doubts regarding what is and is not plagiarism, talk to me.

Credit

Thanks to Ashok Veeraraghavan and Mohammad Teli for sharing their course resources.