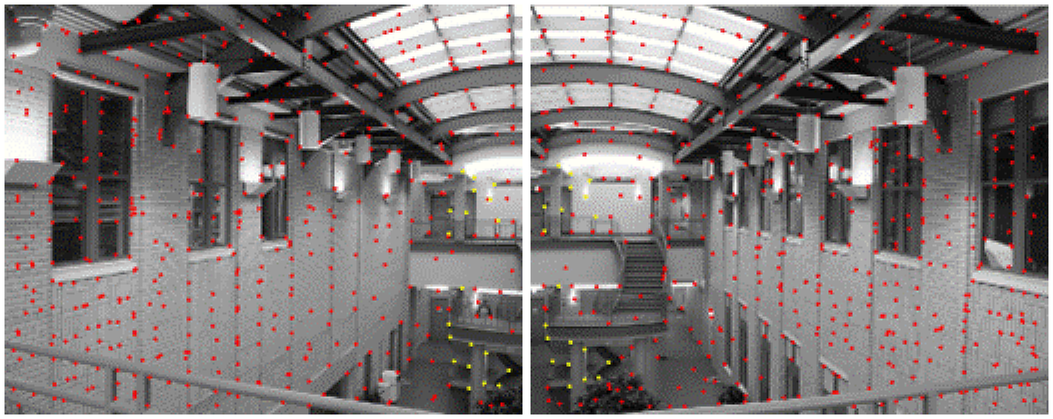
Programming Project #3 (`proj3B`) (second part)
CS180/280A: Intro to Computer Vision and Computational Photography



FEATURE MATCHING for AUTOSTITCHING (second part of a larger project)

The goal of this project is to create a system for automatically stitching images into a mosaic. A secondary goal is to learn how to read and implement a research paper. The project will consist of the following steps:

- B.1: Detecting corner features in an image (20 pts)
- B.2: Extracting a Feature Descriptor for each feature point (20 pts)
- B.3: Matching these feature descriptors between two images (20 pts)
- B.4: Use a robust method (RANSAC) to compute and apply homographies (40 pts)
- B.5: Bells & Whistles

For the following sections, we will follow the paper “Multi-Image Matching using Multi-Scale Oriented Patches” by Brown et al. but with several simplifications. Read the paper first and make sure you understand it, then implement the algorithm.

B.1: Harris Corner Detection

- Start with Harris Interest Point Detector (Section 2). We won’t worry about muti-scale – just do a single scale. Also, don’t worry about sub-pixel accuracy. Re-implementing Harris is a thankless task – so you can use my sample code: harris.py . Include on your webpage a figure of the Harris corners overlaid on the image.
- Implement Adaptive Non-Maximal Suppression (ANMS, Section 3). Include on your webpage a figure of the chosen corners overlaid on the image. This section has multiple moving parts; you may need to read it a few times. You may want to skip this step and come back to it; just choose a random set of corners instead in the meantime.

Deliverables: Show detected corners overlaid on image, with and without ANMS.

B.2: Feature Descriptor Extraction

Implement Feature Descriptor extraction (Section 4 of the paper). Don't worry about rotation-invariance – just extract axis-aligned 8x8 patches. Note that it's extremely important to sample these patches from the larger 40x40 window to have a nice big blurred descriptor. Don't forget to bias/gain-normalize the descriptors. Ignore the wavelet transform section.

Deliverables: Extract normalized 8x8 feature descriptors. Show several extracted features.

B.3: Feature Matching

Implement Feature Matching (Section 5 of the paper). That is, you will need to find pairs of features that look similar and are thus likely to be good matches. For thresholding, use the simpler approach due to Lowe of thresholding on the ratio between the first and the second nearest neighbors. Consult Figure 6b in the paper for picking the threshold. Ignore Section 6 of the paper.

Deliverables: Show matched features between image pairs.

B.4: RANSAC for Robust Homography

For step 4, use 4-point RANSAC as described in class to compute robust homography estimates. Then, produce mosaics by adapting your code from Part A. You may use the same images from part A, but show both manually and automatically stitched results side by side. Produce at least **three** mosaics.

Deliverables: Implement 4-point RANSAC from scratch. Show comparison of stitching manually and automatically. Create >=3 automatic mosaics.

B.5: Bells & Whistles (Extra for CS180, Mandatory for CS280A)

Choose one:

- **Multiscale processing:** Add multiscale processing for corner detection and feature description.
- **Rotation invariance:** Add rotation invariance to the feature descriptors.
- **Panorama recognition:** Implement panorama recognition from an unordered set of images.