ECE 661: Homework 3
Due October 5th

Problem:

The goal of this homework is to become familiar with low-level image feature matching. This will be a first step towards automatically finding the homography between two images of a planar scene which differ by only a small rotation and translation (cf. Figure 4.9 (a) and (b) in the textbook).

Feature matching algorithms generally involve two steps:

1. Select features in both images which should be "matchable" in the other image and are well localized (meaning they match only at a particular pixel location, not over a whole section of the image).

2. Iteratively match features from one image to the other by choosing the best match for each selected feature which exceeds some threshold.

For this homework, you should use the Harris corner detector to select the features (step 1). To match the features (step 2), you should compare two different methods: normalized cross correlation (NCC) and sum of squared difference (SSD). Note that for both methods you should play around with the window size and the matching threshold to obtain the best result. However, perfect matching isn't necessary.

As always, you are encouraged to use the OpenCV library to do your homework but you may not use the function *cornerHarris*.

Solution:

You should turn in a report in pdf format of your homework solution using electronic turn-in. The report should include:

1. A brief outline of the feature selection algorithm including the relevant equations.

2. A brief description of how you did the feature matching for both NCC and SSD. Include some discussion on how the two feature matching algorithms compare on real images (parts 3 and 4 below) and how you set the matching thresholds.

3. The result of applying each matching method to the pair of images at
   http://engineering.purdue.edu/ece661/homework/ECE661_hw3_images.zip
   You should show the matching results by generating a combined image similar to the ones shown below. You can use the OpenCV function *circle* to mark the selected features and the function *line* to show the matches.

4. The results when you apply each matching method to two pairs of images taken with your own camera.

5. Your source code.

As always, you are permitted to look at sample solutions from previous semesters. However, the work you turn in must be your own!

Example Results: