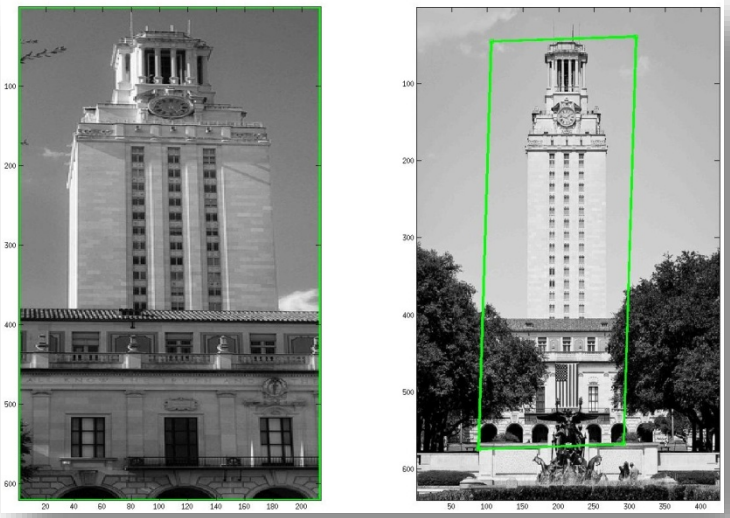
CS 381V Visual Recognition, Spring 2016
Assignment 1: Recognizing specific objects with local feature matching

Out: Wednesday Feb 3
Due: Friday Feb 19, 11:59 pm



Goal

For this assignment you will implement a local feature matching approach to recognize specific objects or scenes in images. Suppose we are given as input 1) a "template" or model image of the object and 2) a "scene" image within which we will search for that object.

What to implement

The basic pipeline to implement is as follows:

- **Descriptors**: For both images, extract local features at interest points. We'll use DoG detections and SIFT descriptors.
- **Initial candidate matches**: Identify the candidate SIFT matches using nearest neighbor search. For every feature in the template, find the nearest feature in the scene.
- **Reduce candidate matches**: Filter the initial matches based on Lowe's ratio test. That is, eliminate matches for which the distance to the first neighbor divided by the distance to the second neighbor is above a threshold of your choosing (e.g., ratio threshold = 0.6). Note, this

entails finding both the first and second nearest feature in the scene, for each feature in the template.

- **RANSAC**: Use RANSAC to find an "inlier" set of matches according to an affine transformation. First, set parameters for the number of iterations and the inlier distance threshold. Then, for each iteration, choose 3 random candidate matches and solve for the affine transformation. Transform all template features according to that transformation. Then, count how many matches among *all* candidates are inliers, based on the distance between the transformed template features' positions and the original scene feature positions. If they are below the threshold, they are an inlier match. Maintain the best affine fit parameters (and the associated inlier match count) found over all iterations. Note that you may want to experiment informally to identify a reasonable value for the inlier distance threshold.
- **Best detection window**: Display the resulting best affine warp of the template boundaries on the scene image (like above).
- **Detection decision**: Decide whether the template object is present in the scene based on those matches (e.g., by a threshold on the number of inliers)

Provided code and data

Download the provided Matlab scripts and test images from the course homepage.

Also download the VLFeats library here: http://www.vlfeat.org/. Follow their instructions for setup. For Matlab, this entails calling `>>> run('VLFEATROOT/toolbox/vl_setup');` where VLFEATROOT is the directory where you downloaded the code. See http://www.vlfeat.org/install-matlab.html .

The provided scripts use functions from VLFeats for SIFT feature extraction and displaying matched features. Run the provided script `hw1script.m` to get started. It will demonstrate the provided display code as well as a simple single feature match.

I recommend debugging your implementation using the `object-template.jpg` as the template and `object-template-rotated.jpg` as the scene image.

What to include in your submission

Your commented source code (NOT including the library code).

A pdf with answers to the following questions, with the relevant figures embedded:

1. Write a script called `matchComparison.m` that uses `showLinesBetweenMatches.m` to display your matching feature results for three scenarios:

   i) **Thresholded nearest neighbors**: Show all matched features according to nearest neighbors, with a threshold on the raw Euclidean distance. That is, eliminate any matches whose distance exceeds a threshold of your choosing (e.g., threshold = 0.8*mean distance should be reasonable).

   ii) **Thresholded ratio test**: Show all matched features that survive the first-second neighbor ratio test (e.g., threshold = 0.6 on the ratio should be reasonable).

   iii) **Inliers**: Show all matched features that survive the affine verification with RANSAC, i.e., all "inlier" matches.

   Do this using each of the three provided scene images in turn as the "scene", and `object-template.jpg` as the "template object". Thus, you will have a total of 9 figures for this part. Please title each one clearly.

   Briefly explain the outcome – what should we notice about these figures?

2. Show your system used for detection. Write a script called `detectObject.m` that cycles over the three provided scene images, and decides whether or not `object-template.jpg` is present in each one. If it is found to be present, the script should draw a rectangle around its boundaries. To do this, transform the four corners of the template image (i.e., [1, 1], [w, 1], [1,h], and [w, h], where w is the template's width and h is the template's height) into the scene image using the affine parameters obtained from RANSAC. Then you can use the provided `drawRectangle.m` function to draw lines connecting those corners. (See the example rectangle in `hw1script.m`.)

   Briefly describe your approach to make the detection decision, and show figures to illustrate the results.

Submit your work as a single zip file via Canvas. Compress the code and pdf report and zip together.

Extra credit (optional, 10 points each)

Test your code on images of your choosing, in order to examine the limits of SIFT matching and affine spatial verification. Explain your results and show figures to illustrate in the writeup.

1. Choose example(s) with stronger scale/rotation/illumination variation that you can still successfully match, and then an example where it fails.

2. Show an example where the number of ratio test neighbors is high, yet the object is not present. Show that the spatial verification stage can eliminate this false match.

Thanks to Flickr users RHensley, vxla, and faster panda kill kill for sharing their images under the Creative Commons license.

Relevant references

- Feb 3 lecture
- Object Recognition from Local Scale-Invariant Features, Lowe, ICCV 1999.
- Object Retrieval with Large Vocabularies and Fast Spatial Matching. J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, CVPR 2007.
- Visual Object Recognition, K. Grauman and B. Leibe. Synthesis Lectures on AI and ML, April 2011, Vol 5, No 2, pp 1-181.