

# CS 223b: Introduction to Computer Vision

## Assignment 3: Image Features

Due date: **Monday, February 2nd 23:59 PST**

You may work in **teams of up to 3 persons**

Submission via email with subject “Assignment 3: [NAMES]” to cs223b09@gmail.com, where [NAMES] is a comma separated list of the full names of everyone in your group.

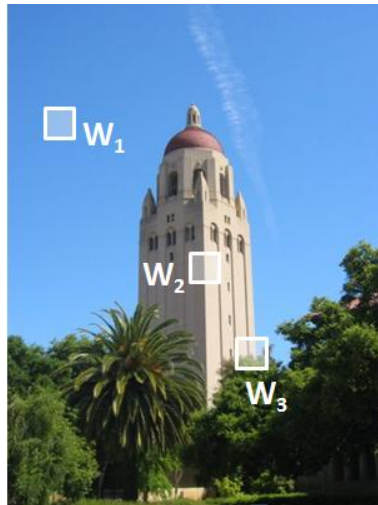
### 1 Harris Corners(10 pts)

- The method of Harris corners was derived from a corner detection algorithm that compared a small window with many neighboring overlapping windows. Sum-of-squared differences (*SSD*) was used to compare these windows:

$$SSD(\mathcal{W}, \Delta x, \Delta y) = \sum_{x,y \in \mathcal{W}} (I(x, y) - I(x + \Delta x, y + \Delta y))^2 \quad (1)$$

$\mathcal{W}$  is the window,  $x$  and  $y$  are pixel coordinates within the window, and  $\Delta x$  and  $\Delta y$  are a shift that defines a second window  $\mathcal{W}'$ . For example, if  $\mathcal{W}$  is `img(10:20, 110:120)` and  $\Delta x = 2, \Delta y = 2$ , then  $\mathcal{W}'$  is `img(12:22, 112:122)`.  $I(x, y)$  is the pixel intensity and we assume color images are converted to grayscale.

Say we evaluate the *SSD* for each of  $\{\Delta x, \Delta y\} = \{(0, 2), (2, 2), (0, -2), (-2, 2), (2, 0), (2, -2), (-2, 0), (-2, -2)\}$  on each of the following windows:



How do the  $\min_{\{\Delta x, \Delta y\}} SSD(\mathcal{W}, \Delta x, \Delta y)$  for each of the three windows compare? How can you find corners in an image with this? Use words and intuition - no numbers or math.

- Show how taking a Taylor expansion of  $I(x + \Delta x, y + \Delta y)$  in eq.( 1) leads to an expression for  $SSD$  that uses the Harris corner matrix

$$\mathbf{C} = \begin{pmatrix} \sum_{x,y \in \mathcal{W}} I_x(x, y)^2 & \sum_{x,y \in \mathcal{W}} I_x(x, y)I_y(x, y) \\ \sum_{x,y \in \mathcal{W}} I_x(x, y)I_y(x, y) & \sum_{x,y \in \mathcal{W}} I_y(x, y)^2 \end{pmatrix} \quad (2)$$

$I_x$  and  $I_y$  are the gradient images in the  $x$  and  $y$  direction (see page 82 of the book). Hint:  $\vec{v}^T \mathbf{A} \vec{v} = \sum_{i,j} v_i A_{ij} v_j$  maybe useful to you.

- Show that the eigenvalues of  $\mathbf{C}$  are both non-negative. Use math, not intuition. Why does it make sense that this must be the case?

Note: Neither of these last two questions should take more than half a page.

## 2 Image Features(20 pts)

In this problem, you will explore the properties of different features used to match objects between pairs of images. The keypoints will be provided, so your task is to write the feature computation and matching part.

Download the image pairs and Matlab code from <http://cs223b.stanford.edu/assign3>. `testFeatures.m` is a function that will run a feature function on all pairs of images and display the results. For example, try

```
testFeature(@random)
```

The @ indicates a function handle and `random.m` is a function that randomly assigns matches.

### Feature Function Specification

Your feature will take the following inputs and produce the following outputs:

```
[ C ]=random(im1, im2, keypts1, keypts2)
```

`im1` and `im2` are intensity images (not color). `keypts#` have the form

$$\begin{bmatrix} x_1 & x_2 & \cdots \\ y_1 & y_2 & \cdots \end{bmatrix} \quad (3)$$

where  $(x_i, y_i)$  is the location of the  $i$ -th keypoint of the image#.

$$C(i, j) = \begin{cases} 1 & : \text{ if keypt } i \text{ in im1 matches keypt } j \text{ in im2} \\ 0 & : \text{ otherwise} \end{cases} \quad (4)$$

Note that your feature functions must do both the feature computation (producing a vector of numbers that describe the point) and the feature matching (deciding which features are actually the same in the two images). The matching can become quite complex in practice, but here use a simple  $\mathcal{L}_2$  distance. If [the smallest distance between a given feature in image 1 and features in image 2] is smaller than  $[0.8 \times \text{the second smallest distance between a given}$

feature in image 1 and features in image 2], than we assume the given feature in image 1 and the closet feature in image 2 matches.

Write a feature for each of the following methods:

- **Template Matching** Write a simple template matching algorithm. The feature vector should be made of the  $16 \times 16$  pixels in a window around the keypoint.
- **Template Matching with Lighting Invariance** For each feature vector, subtract the mean of the elements, than scale them so that they have a variance of 1. What is the scaling factor that makes the zero-mean elements have a variance of 1?
- **SIFT-like Histogram of Gradients** Find the gradient strength and direction at each pixel in a  $16 \times 16$  pixel window centered on the keypoint. Build a histogram of gradient strength vs. direction (i.e., each bin contains the sum of the gradient strenghts for all the gradients whose directions fell within that bin) and apply the lighting invariance method from the previous part. See [Lowe 04] for more details, but note that their method is far more complex than what you need to implement.
- **SIFT-like Histogram of Gradients with Orientation Invariance** Building on the previous feature, find the peak in each histogram of gradients and use this in some intelligent way to give your descriptor a degree of orientation invariance.

This assignment is quite open-ended. Grading will be based both on whether your features work where they whould (and don't worry- there are many places they won't, by construction) as well as your understanding of *why* they succeed or fail.

### 3 Submitting

- Responses for the Harris corner questions (Sec. 1).
- `correspondences_[function name]_[pair_id].jpg` for each of your features.
- For each feature, write a sentence or two describing why it works or does not work for each image pair.
- Scaling factor for lighting invariance.
- Matlab code for your features.

When finished, submit any sourcecode and a .doc or .pdf file containing all output images. Submission should be via an e-mail to `cs223b09@gmail.com` with the subject "Assignment 3: [NAMES]".

### References

[Lowe 04] D. Lowe: *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, 60, 2 (2004). pp. 91-110.