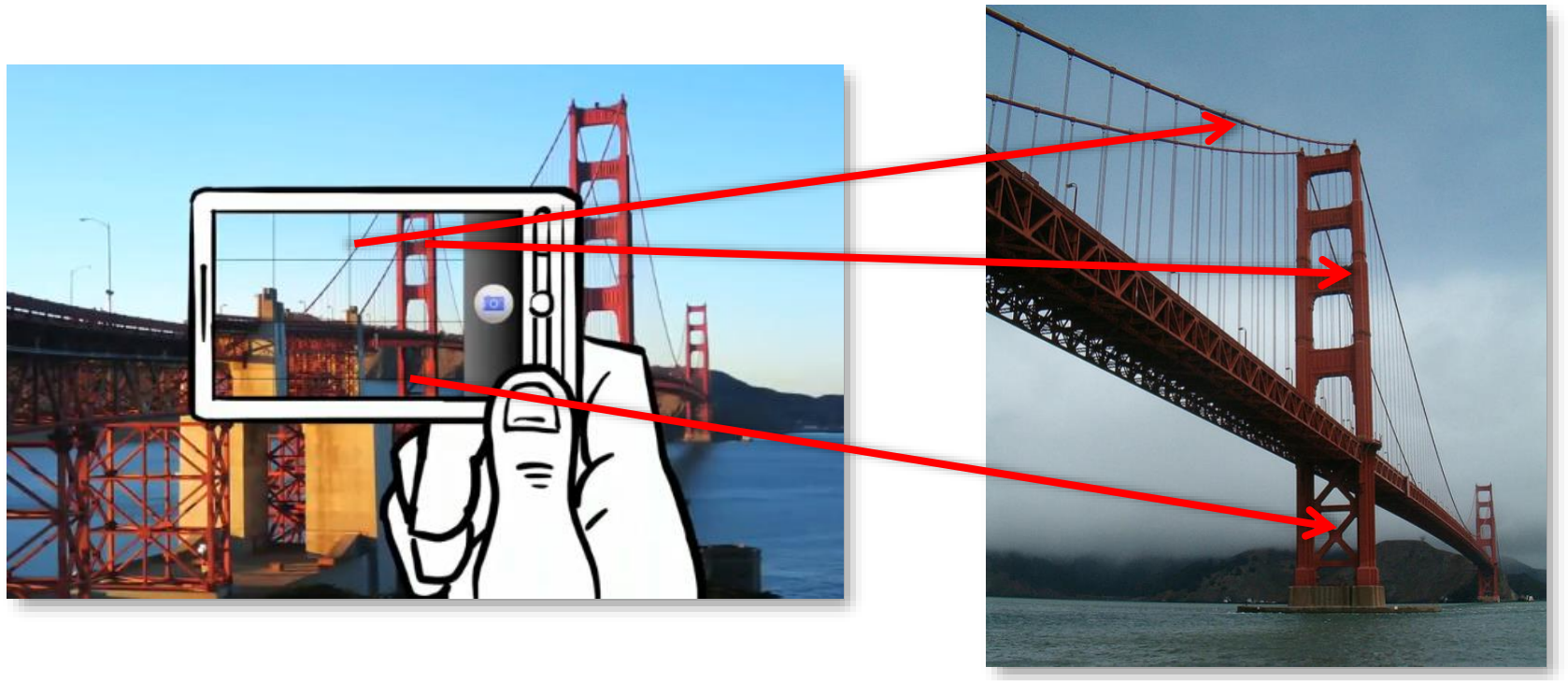
CS5670: Computer Vision

Feature matching



Reading

- Szeliski (1st edition): 4.1

Announcements

- Project 1 artifact due tonight at 11:59pm on CMSX
- Project 2 out today, due Friday, March 12 at 7pm
  - To be done in groups of two – will host breakout sessions at the end of class (last 10 minutes)
- A (slightly shorter) quiz this Wednesday, due 7 minutes after the start of class

Project 2 Demo

Alternate Harris score

- For Project 2, you will use an alternate definition of the Harris score:

$$R = \lambda_1 \lambda_2 - k \cdot (\lambda_1 + \lambda_2)^2 = \det(M) - k \cdot \mathrm{tr}(M)^2$$

Quiz

Question 1

1 / 1 pts

Below, we have a 3x3 image filter D_y that we would like to compute the approximate **first-order partial derivative of the image with respect to y** (i.e., the partial derivative in the vertical direction) when convolved with an image. There are four values missing from this filter: A, B, C, and D. What numbers would we assign to A, B, C, and D in order to form the desired filter?

0	[A]	0
[B]	0	[C]
0	[D]	0

- (●) A = -1/2, B = 0, C = 0, D = 1/2
- ( ) A = 0, B = 1/2, C = -1/2, D = 0
- ( ) A = 0, B = -1/2, C = 1/2, D = 0
- ( ) A = 1/4, B = 1/4, C = 1/4, D = 1/4

Quiz

Question 2

1 / 1 pts

Consider the filter D_y in the previous question that computes a first-order partial image derivative with respect to y. Imagine convolving this filter with itself, resulting in a new filter kernel X.

How would you describe the operation that X performs when convolved with an image?

- Image blur
- Image sharpening
- Second-order partial derivative with respect to y
- Second-order partial derivative with respect to x

Question 3

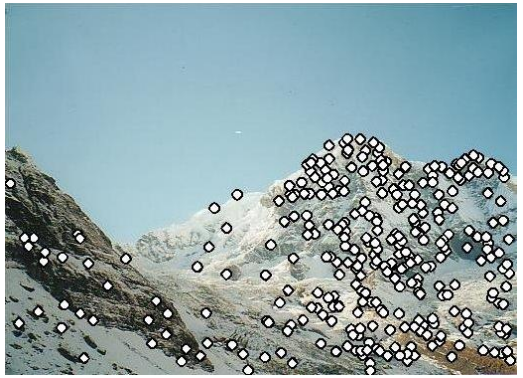
1 / 1 pts

Which of the following image regions would make good locations for detecting an image feature?

- Flat, textureless image area
- Image edge
- Corner-like point
- All of the above

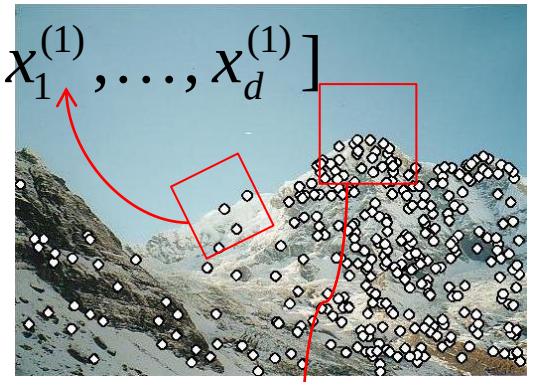
Local features: main components

1) Detection: Identify the interest points



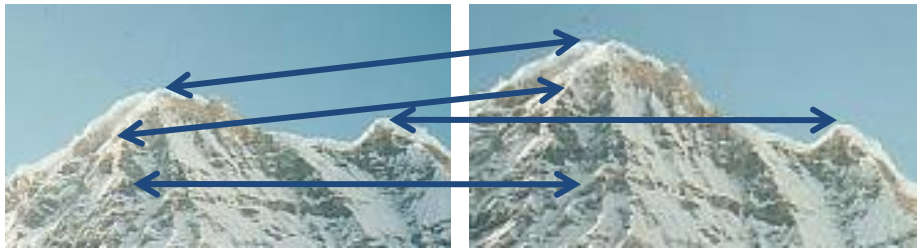
2) **Description: Extract vector feature descriptor surrounding each interest point.**

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

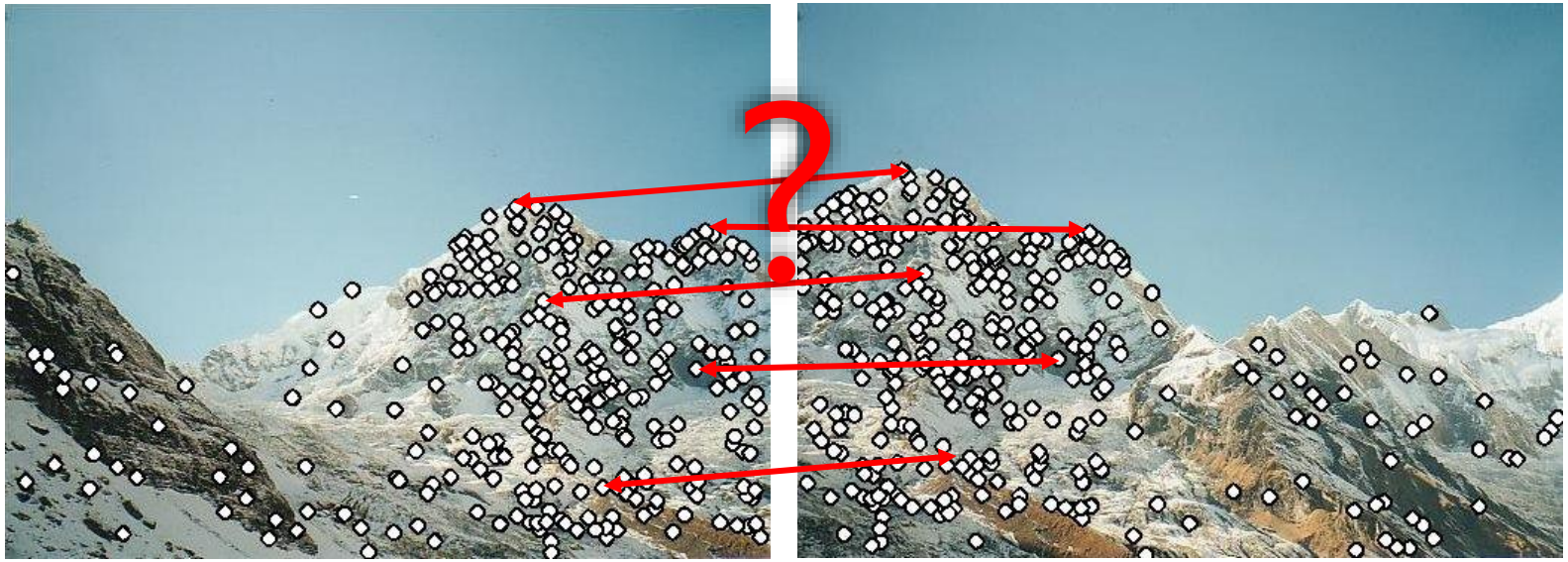
3) Matching: Determine correspondence between descriptors in two views



Kristen Grauman

Feature descriptors

We know how to detect good points
Next question: **How to match them?**

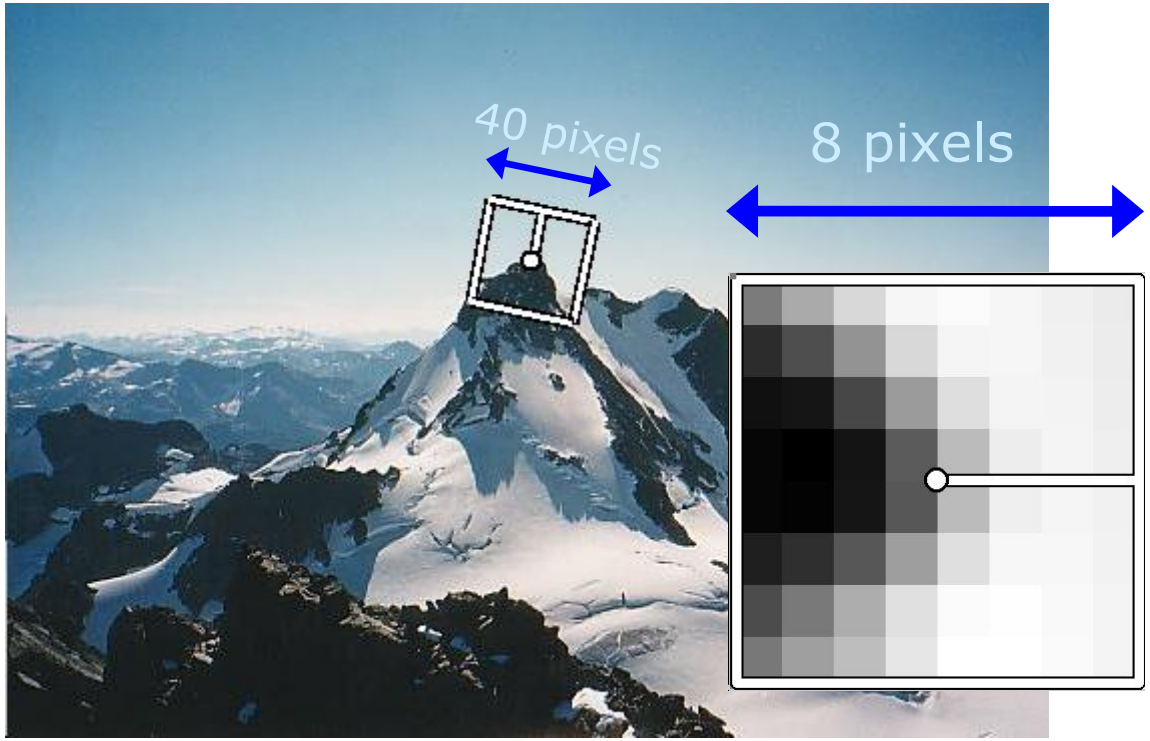


Answer: Come up with a *descriptor* for each point, find similar descriptors between the two images

Multiscale Oriented PatcheS descriptor

Take 40x40 square window around detected feature

- Scale to 1/5 size (using prefiltering)
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window (why?)



Adapted from slide by Matthew Brown

Detections at multiple scales

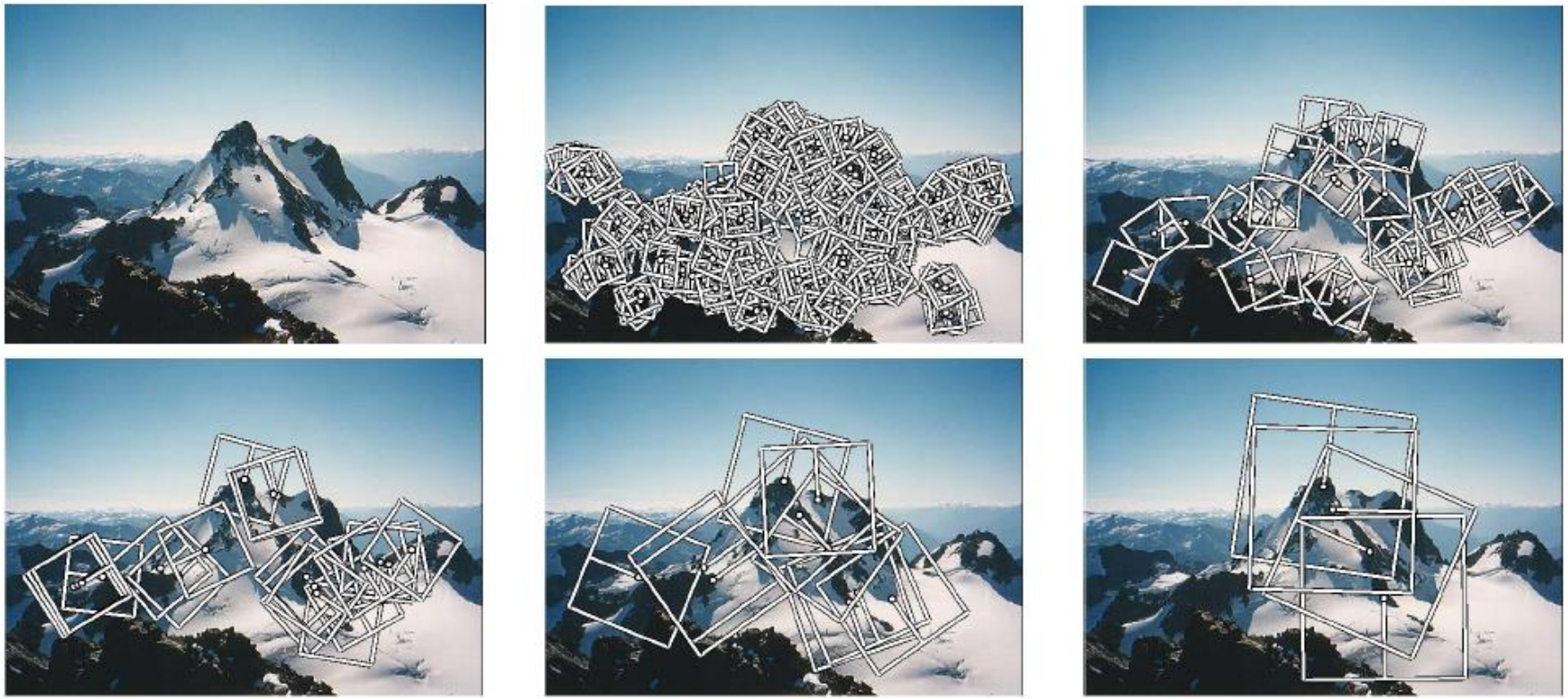
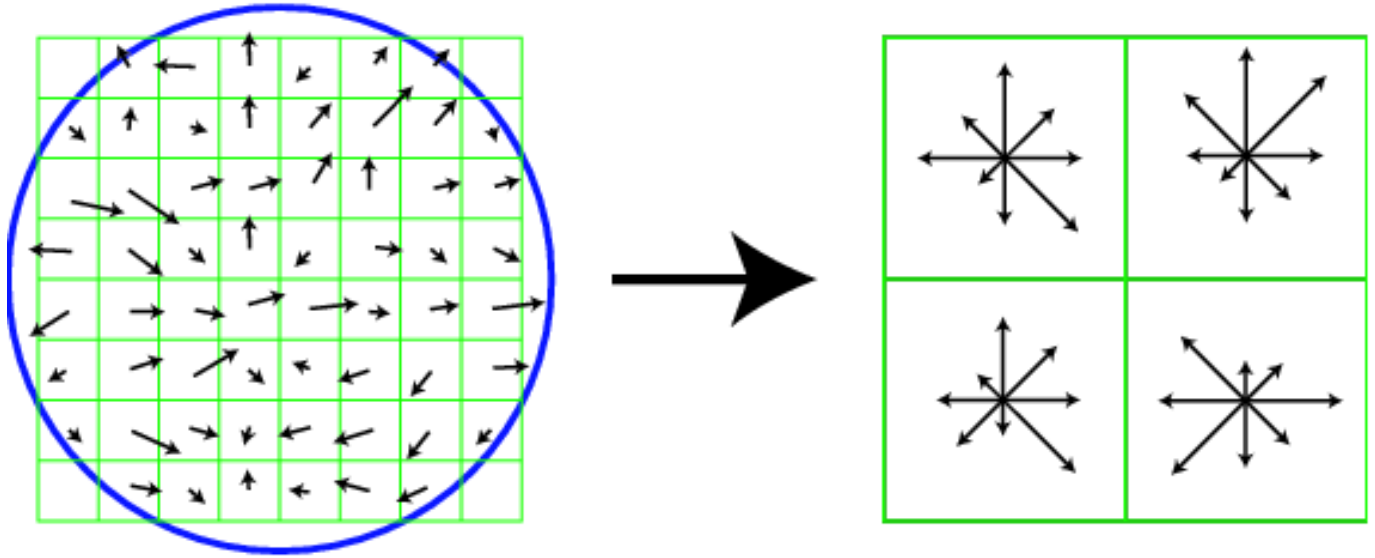


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor

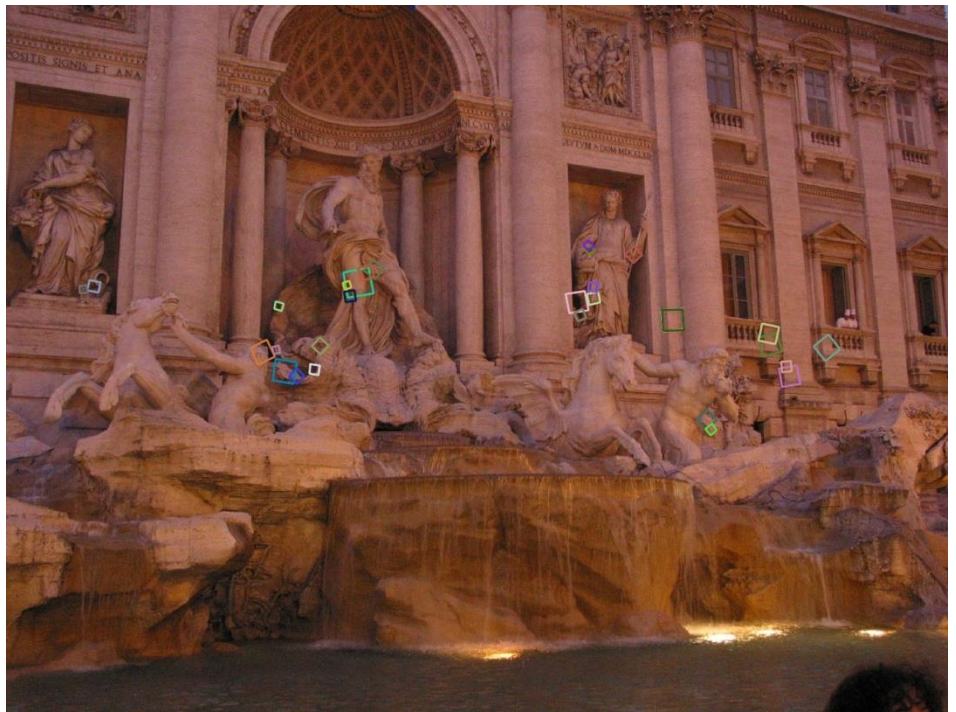


Adapted from slide by David Lowe

Properties of SIFT

Extraordinarily robust matching technique

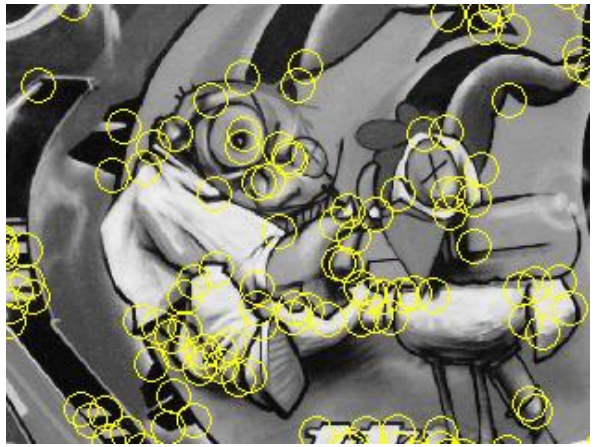
- Can handle changes in viewpoint (up to about 60 degree out of plane rotation)
- Can handle significant changes in illumination (sometimes even day vs. night (below))
- Pretty fast—hard to make real-time, but can run in <1s for moderate image sizes
- Lots of code available



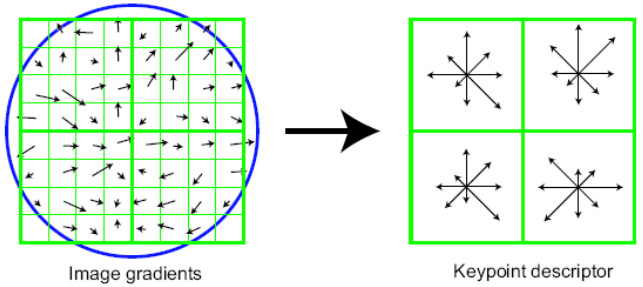
Questions?

Summary

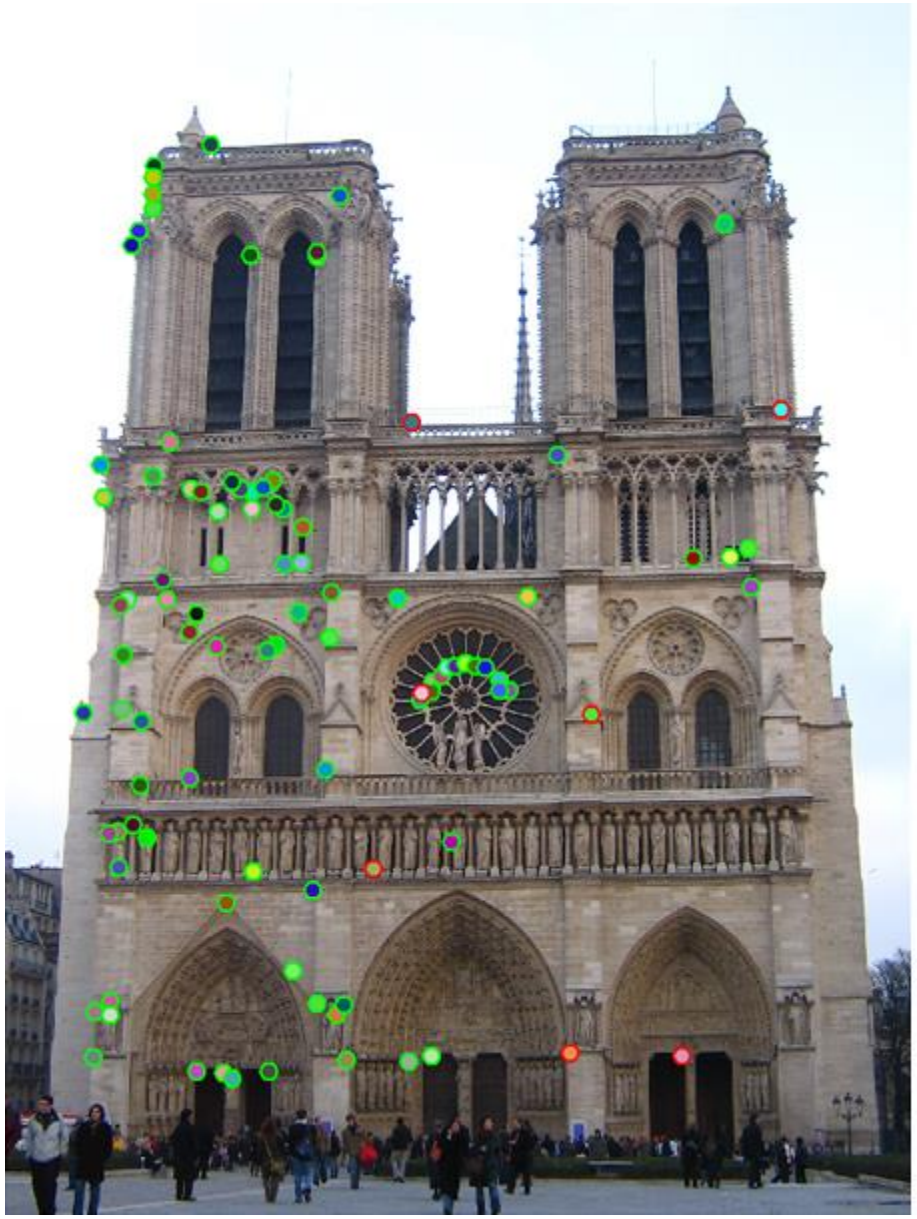
- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG

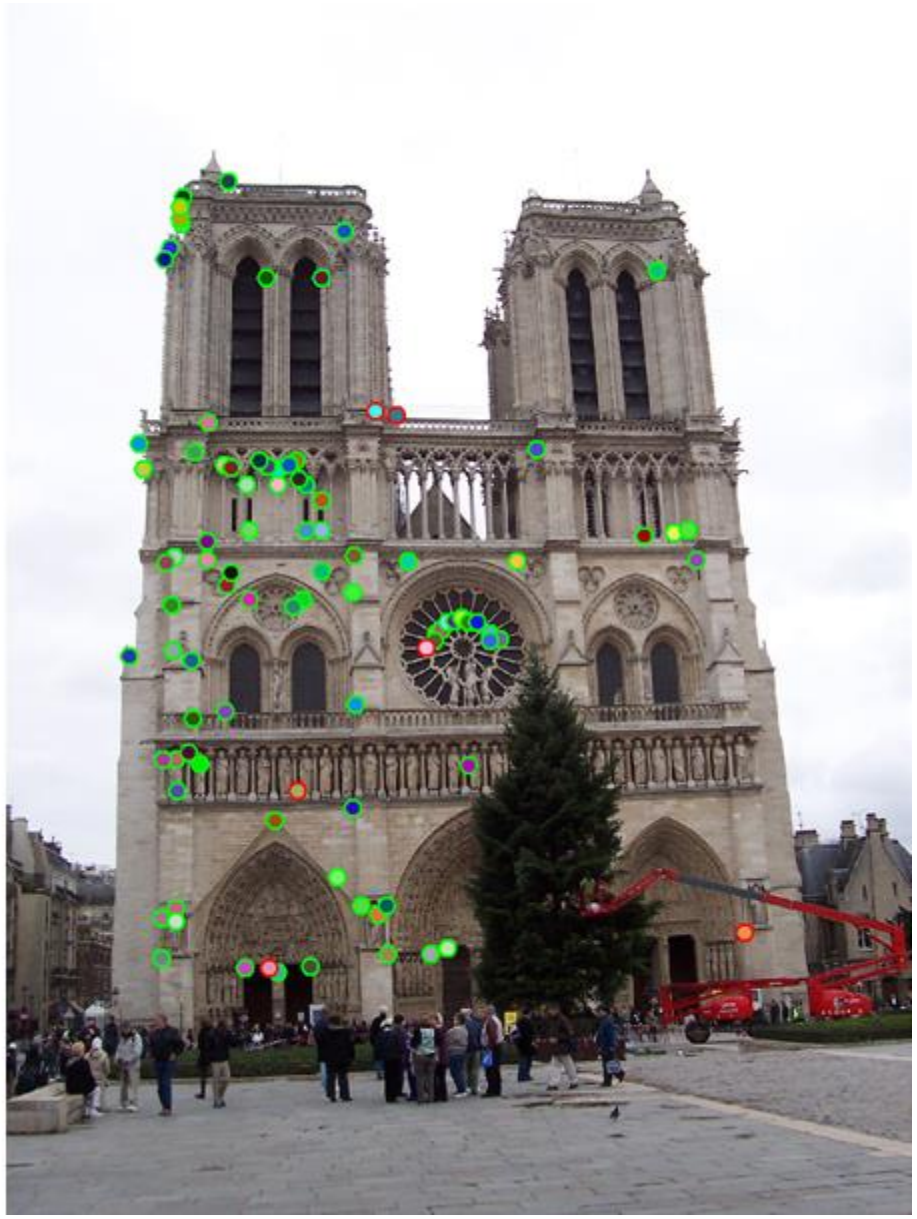


- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT and variants are typically good for stitching and recognition
  - Can learn descriptors (and detectors)



Which features match?





Feature matching

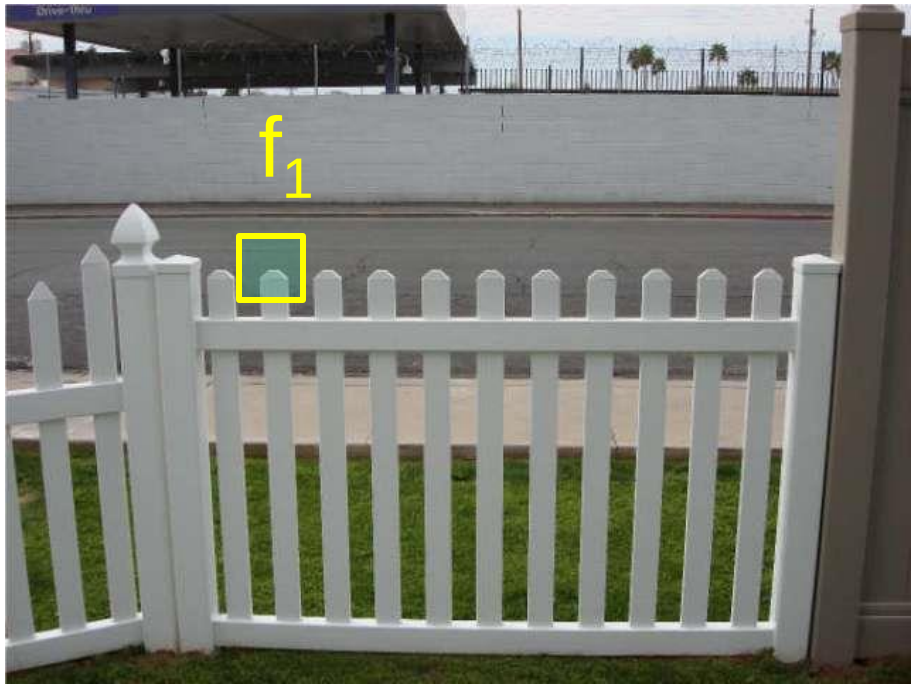
Given a feature in I_1, how to find the best match in I_2?

1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min distance

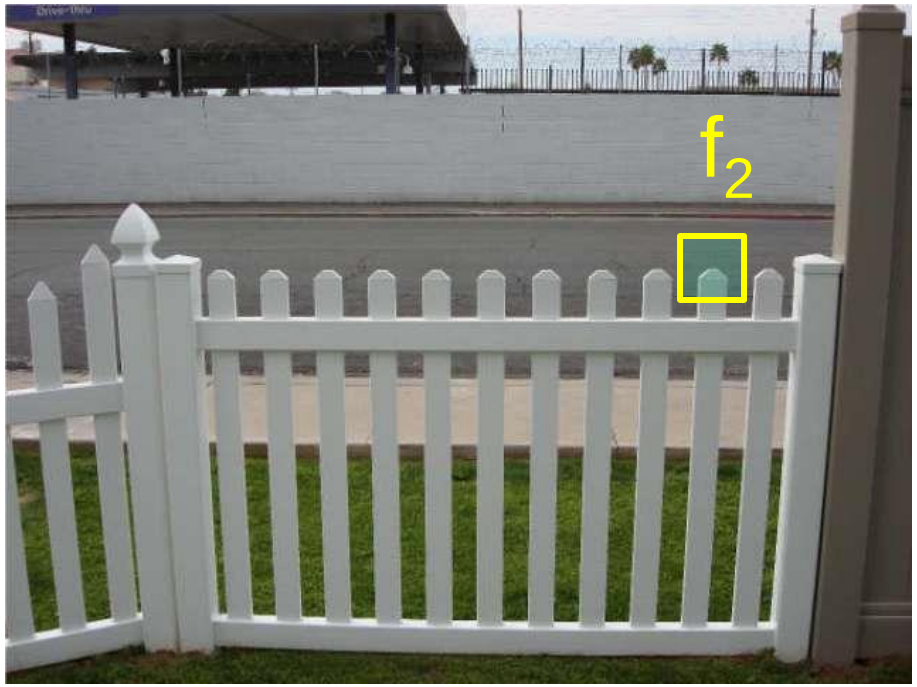
Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach: L_2 distance, $\| f_1 - f_2 \|$
- can give small distances for ambiguous (incorrect) matches



I_1

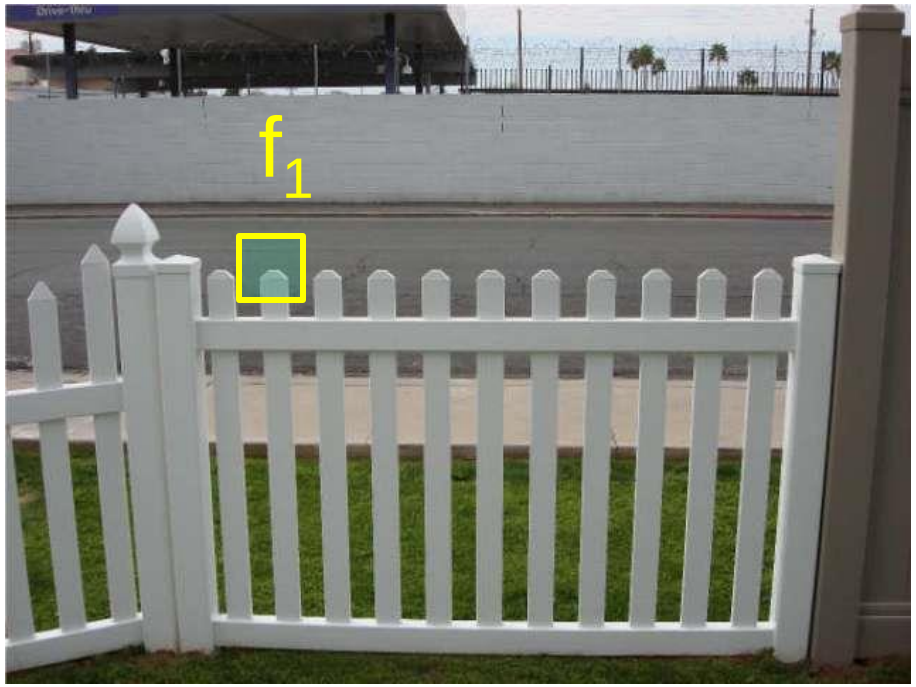


I_2

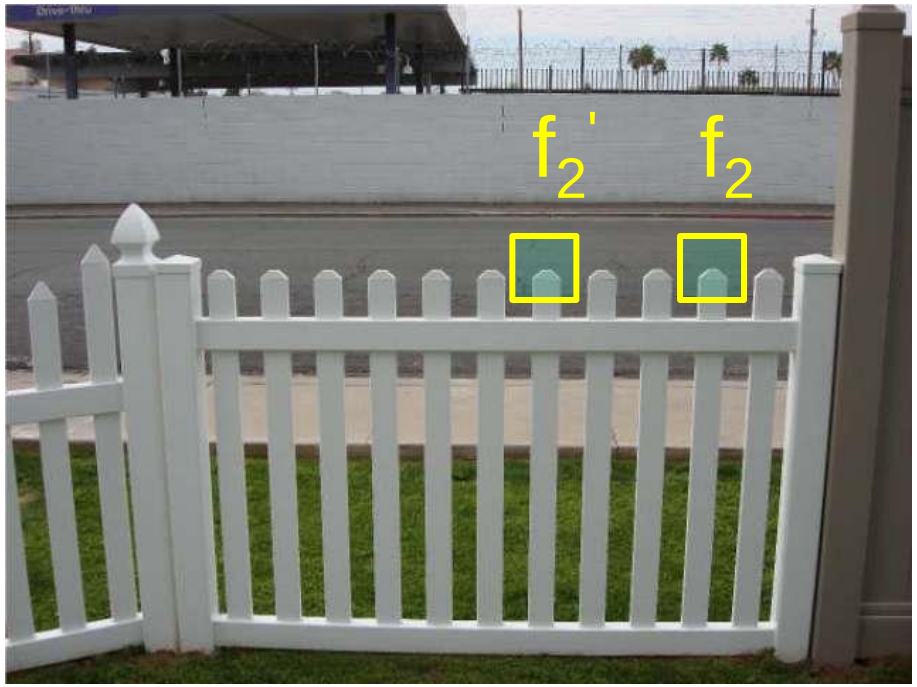
Feature distance

How to define the difference between two features f_1, f_2?

- Better approach: ratio distance = $\| f_1 - f_2 \| / \| f_1 - f_2' \|$
  - f_2 is the best SSD match to f_1 in I_2
  - f_2' is the 2nd best SSD match to f_1 in I_2
  - gives large values for ambiguous matches



I_1

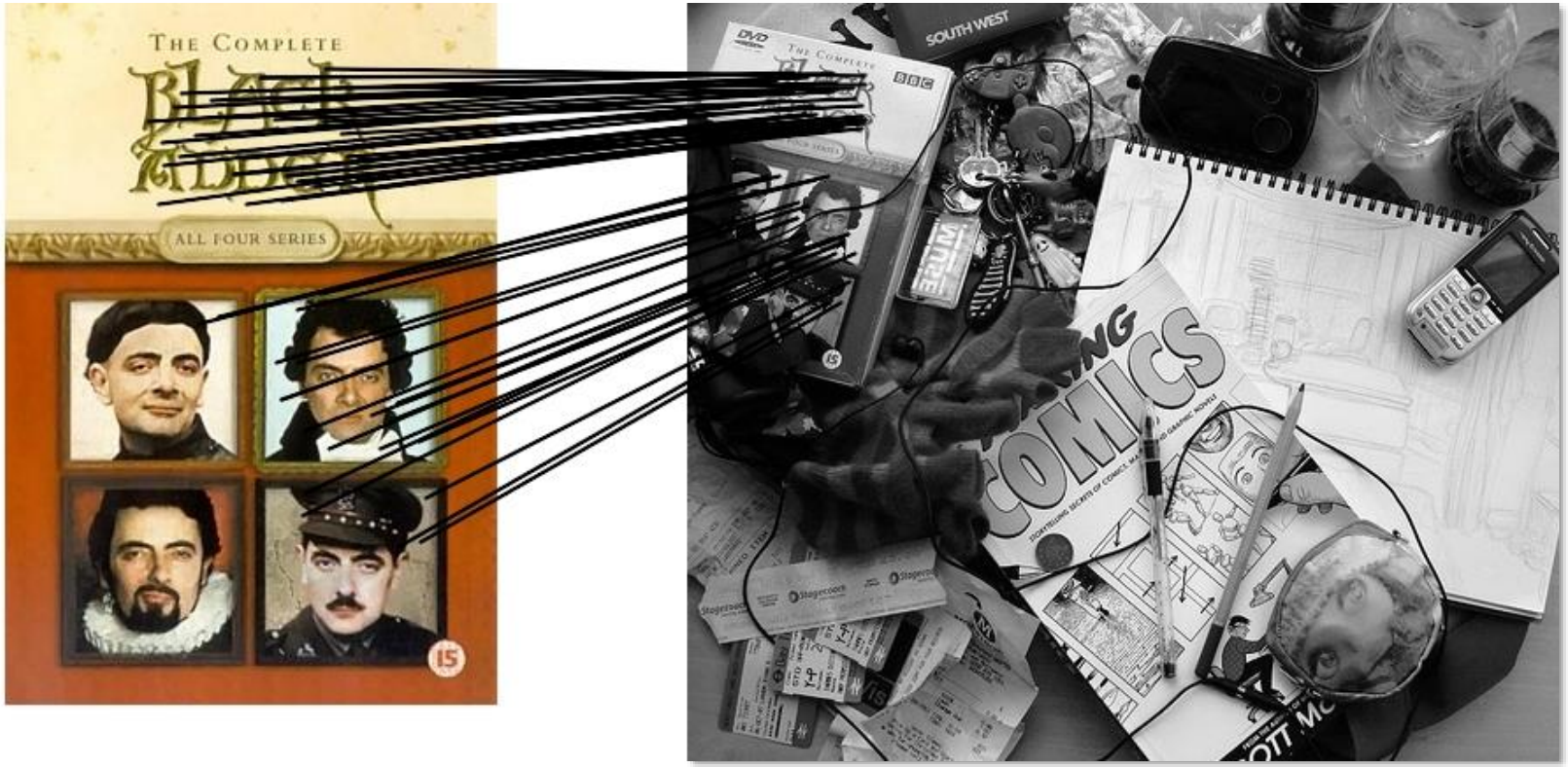


I_2

Feature distance

- Does the SSD vs “ratio distance” change the best match to a given feature in image 1?

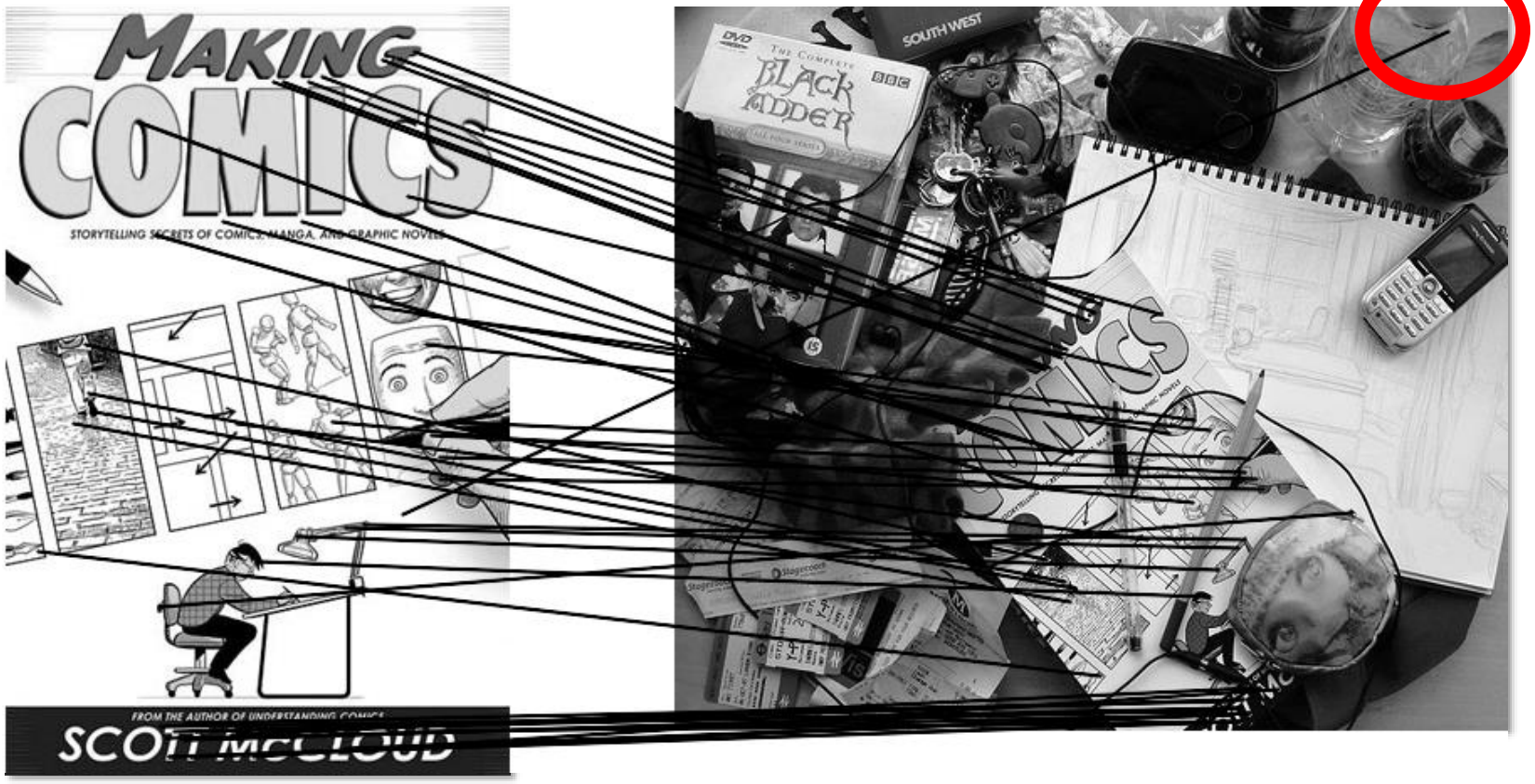
Feature matching example



58 matches (thresholded by ratio score)

Feature matching example

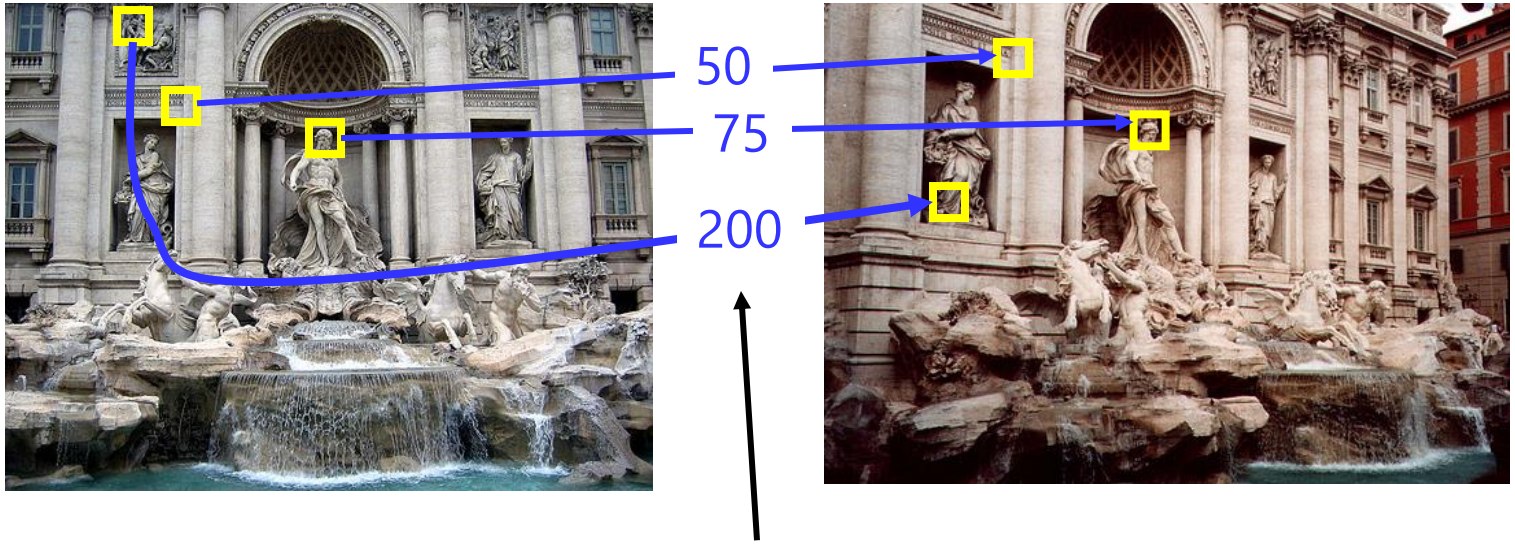
We'll deal with **outliers** later



51 matches (thresholded by ratio score)

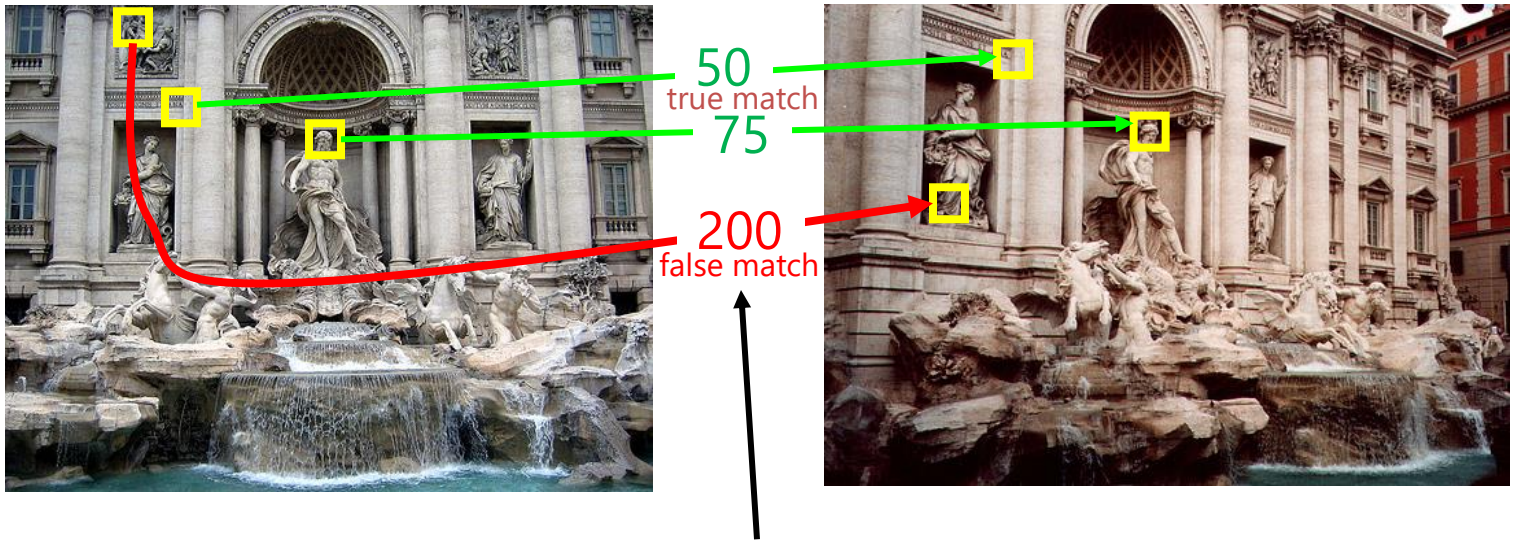
Evaluating the results

How can we measure the performance of a feature matcher?



True/false positives

How can we measure the performance of a feature matcher?



The distance threshold affects performance

- True positives = # of detected matches that survive the threshold that are correct
- False positives = # of detected matches that survive the threshold that are incorrect

Example

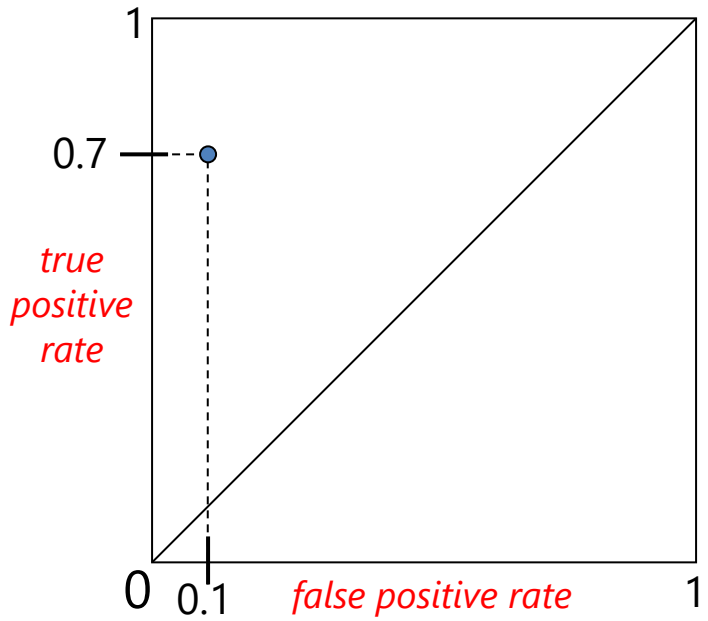
- Suppose our matcher computes 1,000 matches between two images
  - 800 are correct matches, 200 are incorrect (according to an oracle that gives us ground truth matches)
  - A given threshold (e.g., ratio distance = 0.6) gives us 600 correct matches and 100 incorrect matches that survive the threshold
  - True positive rate = 600 / 800 = ¾
  - False positive rate = 100 / 200 = ½

Evaluating the results

How can we measure the performance of a feature matcher?

$$\frac{\text{\# true positives surviving threshold}}{\text{\# total correct matches (positives)}}$$

recall



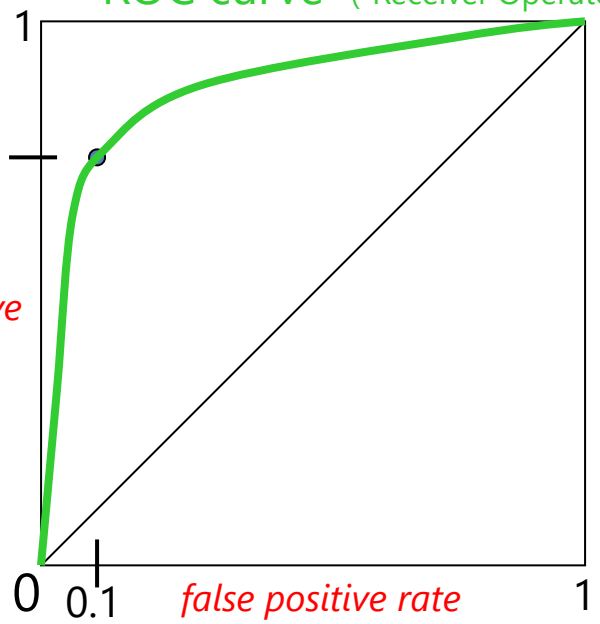
$$\frac{\text{\# false positives surviving threshold}}{\text{\# total incorrect matches (negatives)}}$$

1 - *specificity*

Evaluating the results

How can we measure the performance of a feature matcher?

ROC curve ("Receiver Operator Characteristic")



$$\frac{\text{\# true positives surviving threshold}}{\text{\# total correct matches (positives)}}$$

recall

$$\frac{\text{\# false positives surviving threshold}}{\text{\# total incorrect matches (negatives)}}$$

1 - *specificity*

Single number: Area Under the Curve (AUC)

E.g. AUC = 0.87

1 is the best

ROC curves – summary

- By thresholding the score at different thresholds, we can generate sets of matches with different true/false positive rates
- ROC curve is generated by computing rates at a set of threshold values swept through the full range of possible threshold
- Area under the ROC curve (AUC) summarizes the performance of a feature pipeline (higher AUC is better)

More on feature detection/description

http://www.robots.ox.ac.uk/~vgg/research/affine/
http://www.cs.ubc.ca/~lowe/keypoints/
http://www.vision.ee.ethz.ch/~surf/

Publications

Region detectors

- *Harris-Affine & Hessian Affine*: K. Mikolajczyk and C. Schmid, Scale and Affine invariant interest point detectors. In IJC V 60(1):63-86, 2004. PDF
- *MSER*: J.Matas, O. Chum, M. Urban, and T. Pajdla, Robust wide baseline stereo from maximally stable extremal regions. In BMVC p. 384-393, 2002. PDF
- *IBR & EBR*: T.Tuytelaars and L. Van Gool, Matching widely separated views based on affine invariant regions . In IJCV 59(1):61-85, 2004. PDF
- *Salient regions*: T. Kadir, A. Zisserman, and M. Brady, An affine invariant salient region detector. In ECCV p. 404-416, 2004. PDF
- *All Detectors - Survey*: T. Tuytelaars and K. Mikolajczyk , Local Invariant Feature Detectors - Survey. In CVG, 3(1):1-110, 2008. PDF

Region descriptors

- *SIFT*: D. Lowe, Distinctive image features from scale invariant keypoints. In IJCV 60(2):91-110, 2004. PDF

Performance evaluation

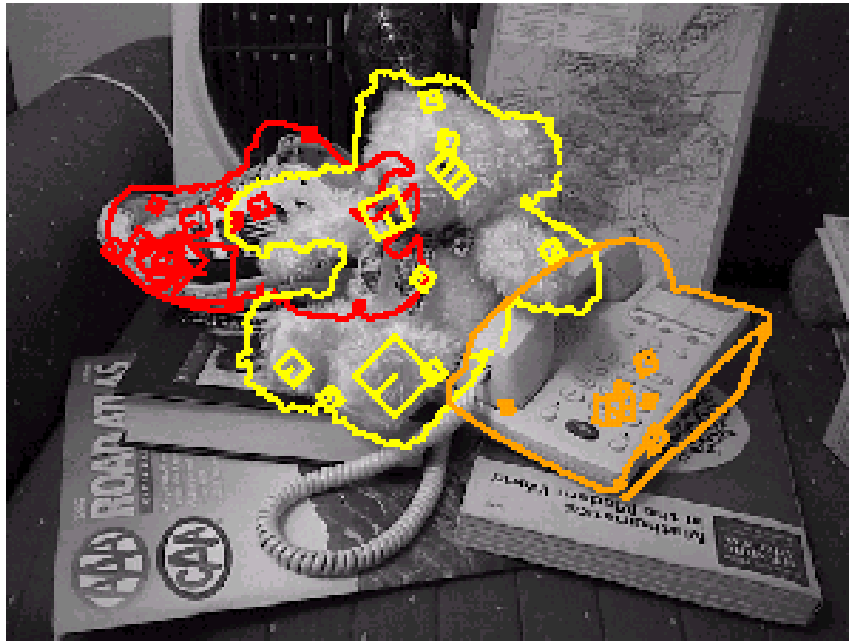
- K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. Van Gool, A comparison of affine region detectors. In IJCV 65(1/2):43-72, 2005. PDF
- K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors. In PAMI 27(10):1615-1630 . PDF

Lots of applications

Features are used for:

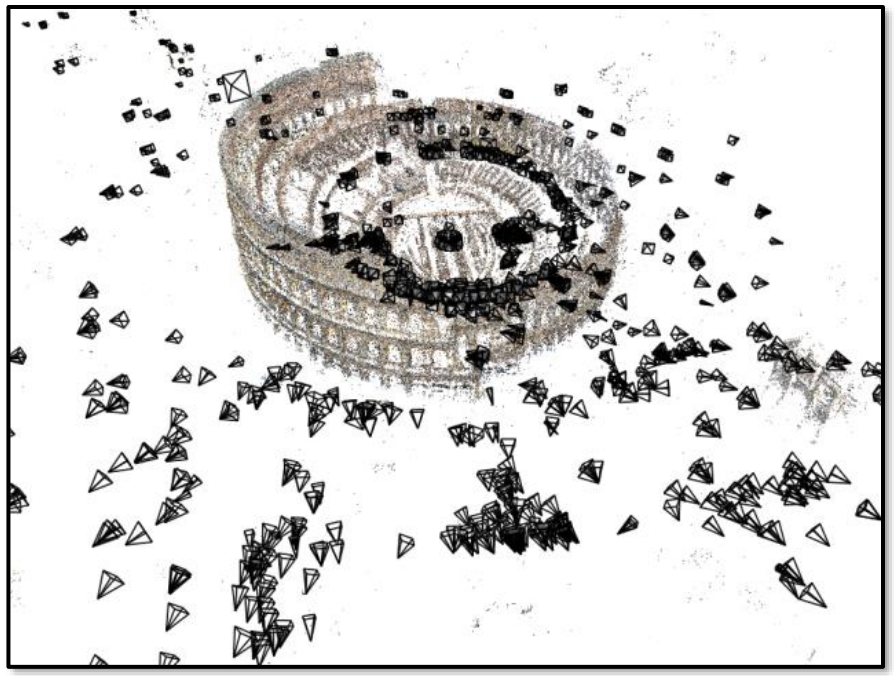
- Image alignment (e.g., mosaics)
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
- … other

Object recognition (David Lowe)



3D Reconstruction

Internet Photos (“Colosseum”)



Reconstructed 3D cameras and points

Augmented Reality

Live demo?

Questions?