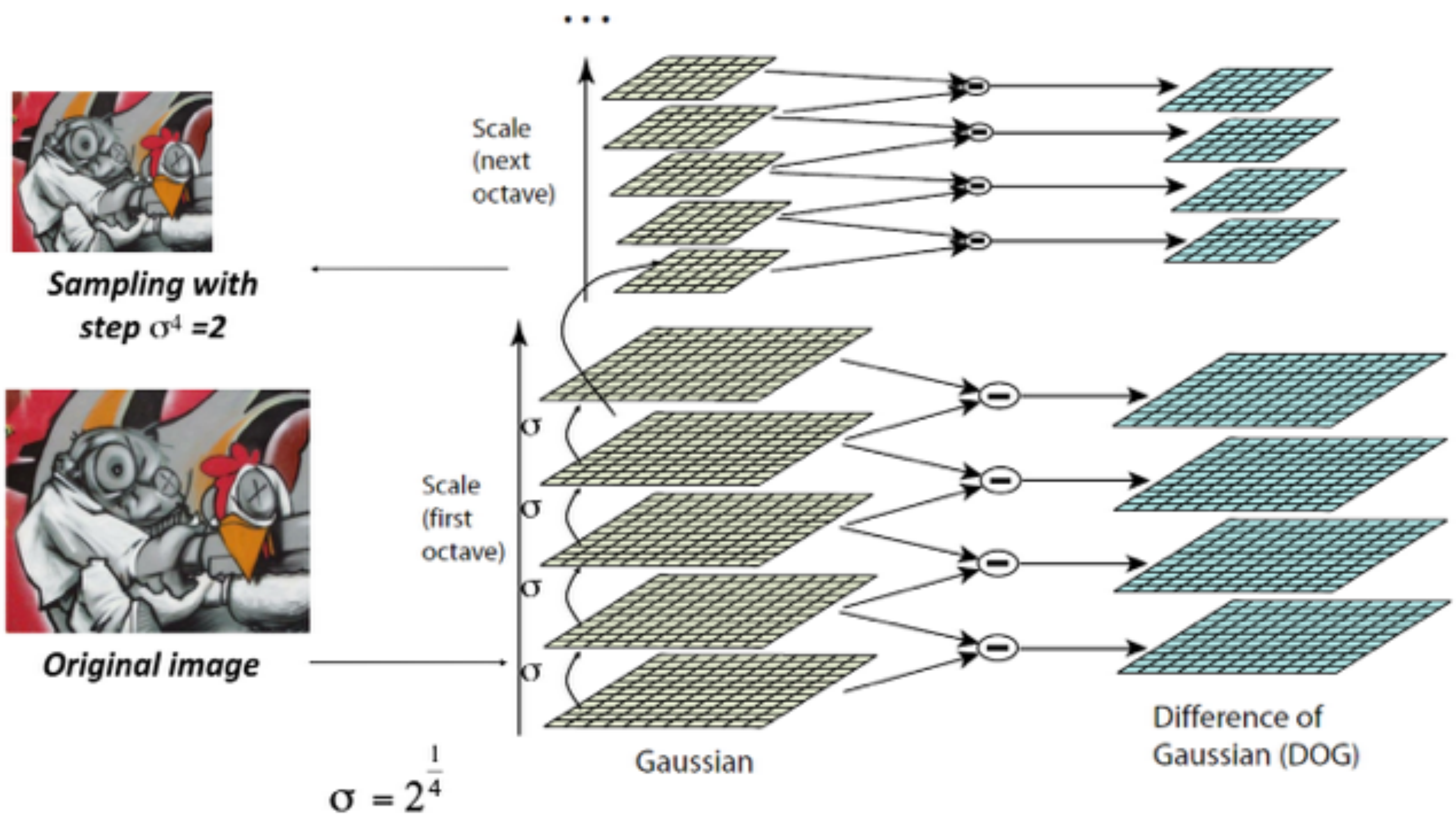
Scale Invariant Feature Transform

1. Scale-space Extrema Detection

Scale-space



Scale Invariant Feature Transform

Scale-space Extrema Detection

Maxima and minima of the DoG Images

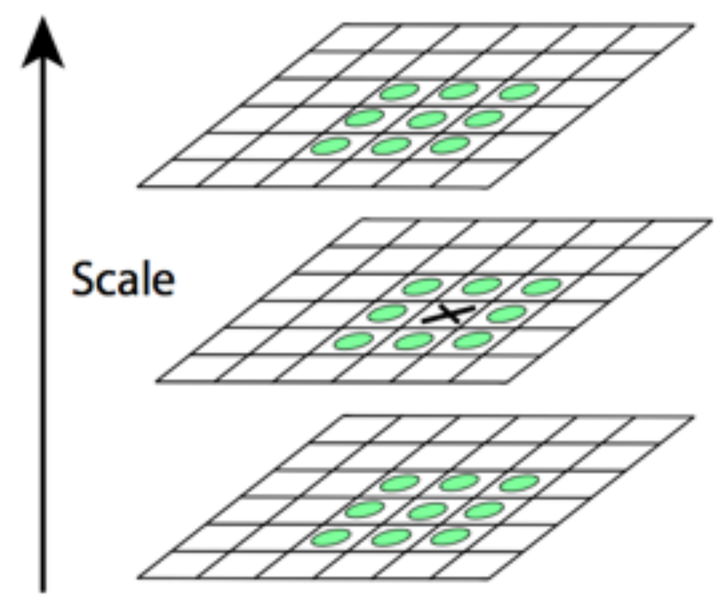
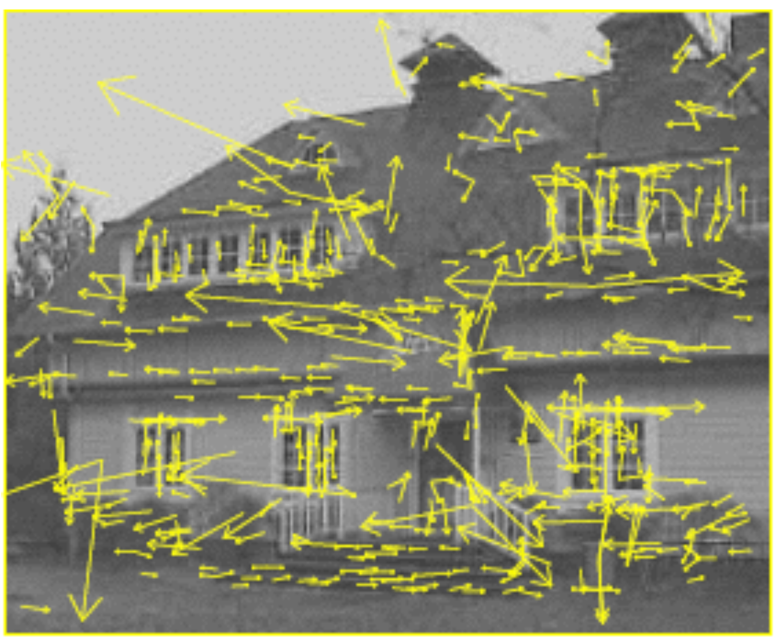


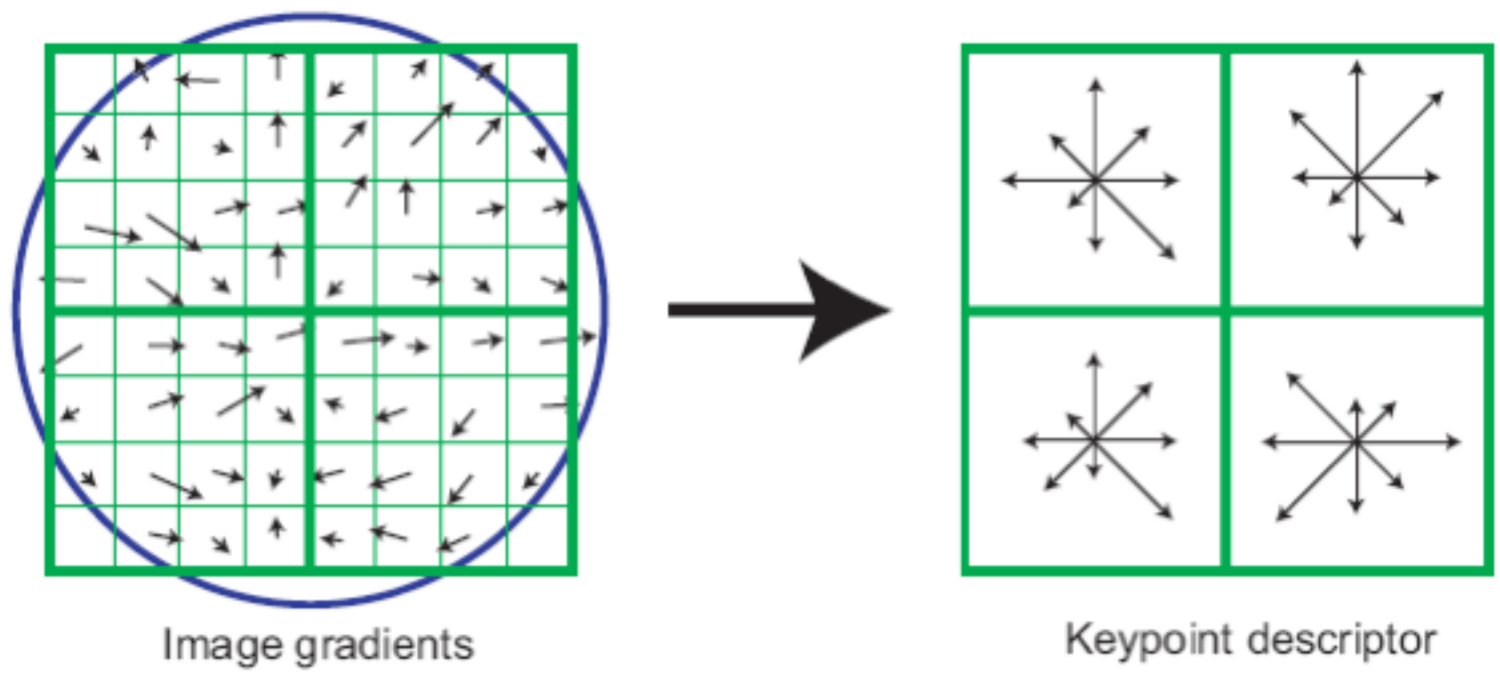
Figure 2: Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles).

http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0405/MURRAY/SIFT.html

What kind of things do we compute histograms of?

- Histograms of oriented gradients



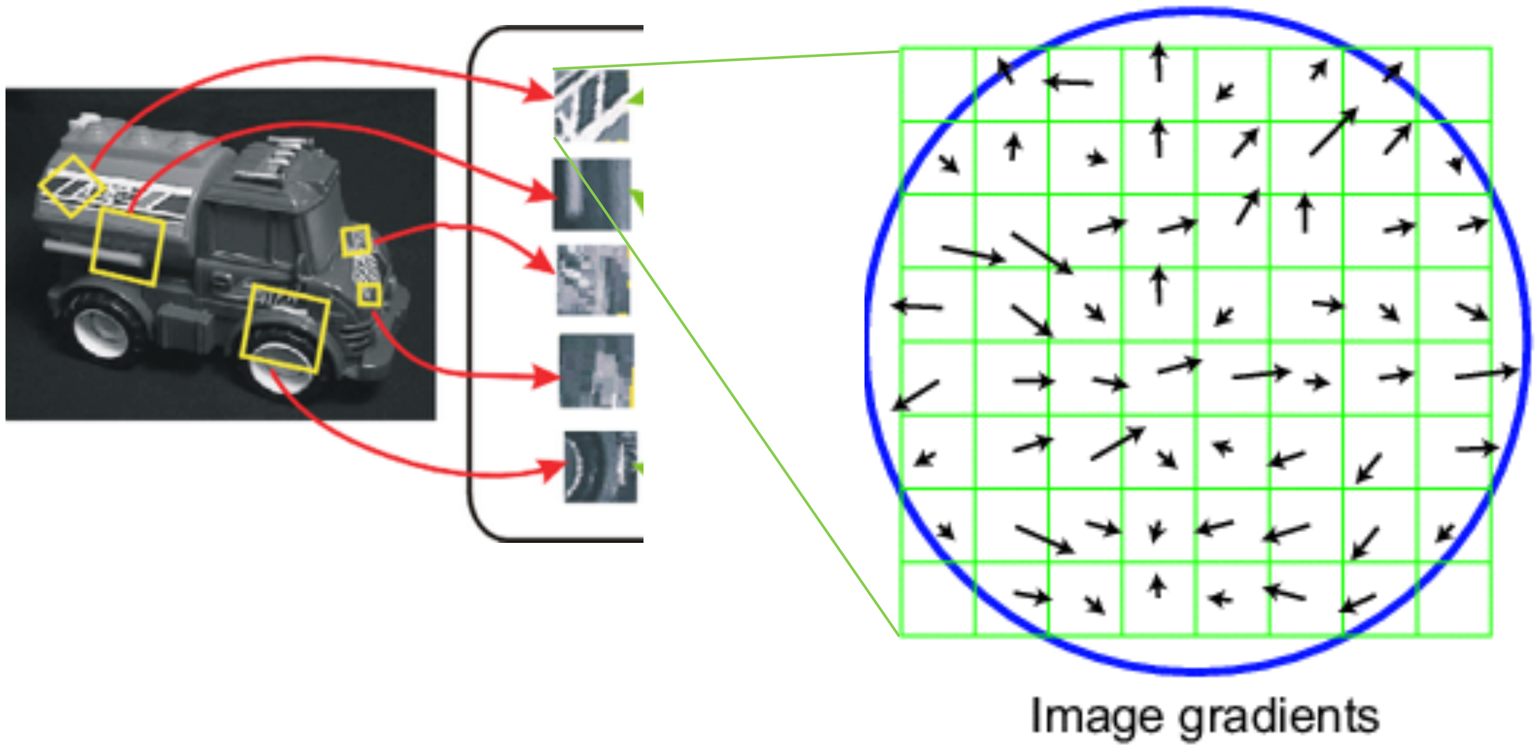


SIFT – Lowe IJCV 2004

Image gradients orientation and magnitudes are sampled around key point location

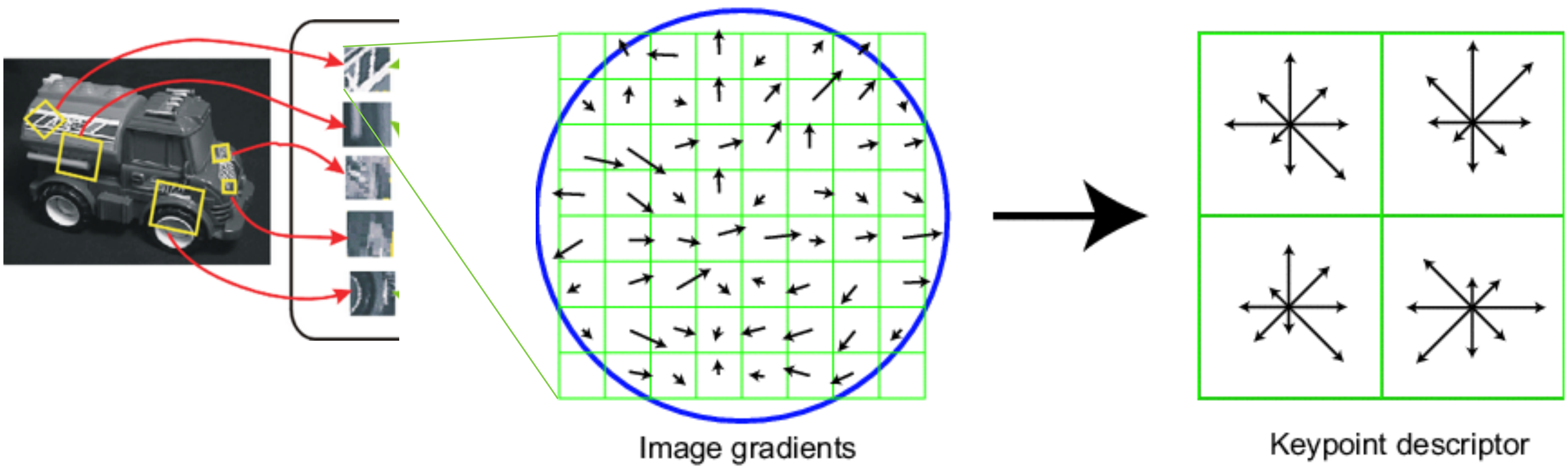
SIFT vector formation

- Computed on rotated and scaled version of window according to computed orientation & scale
  - resample the window
- Based on gradients weighted by a Gaussian of variance half the window (for smooth falloff)



SIFT vector formation

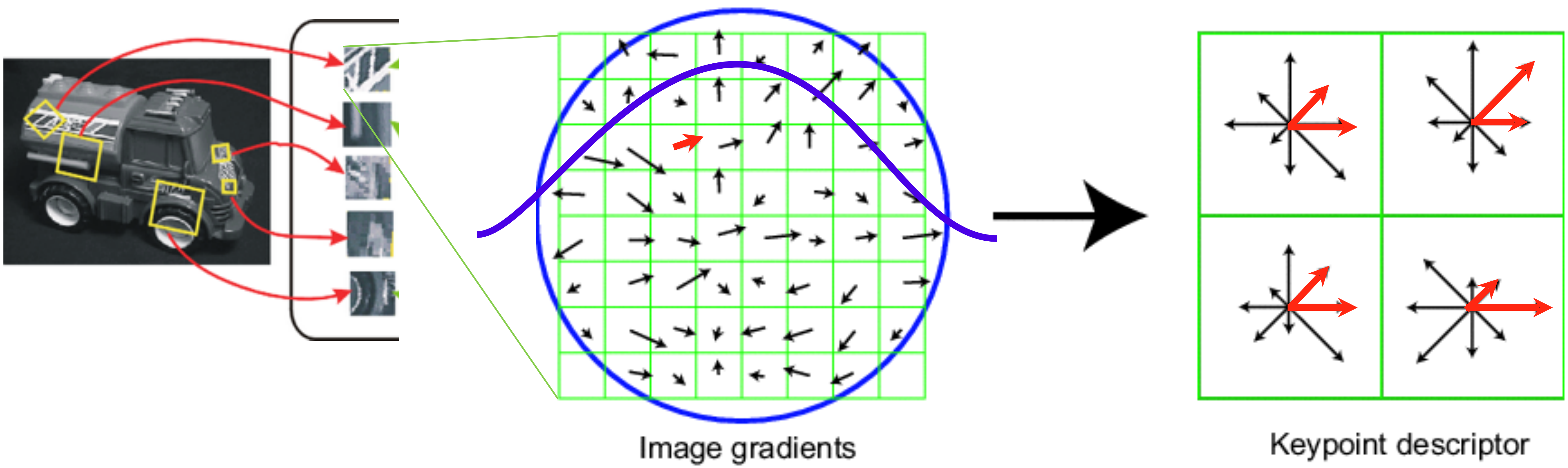
- 4x4 array of gradient orientation histogram weighted by magnitude
- 8 orientations x 4x4 array = 128 dimensions
- Motivation: some sensitivity to spatial layout, but not too much.



showing only 2x2 here but is 4x4

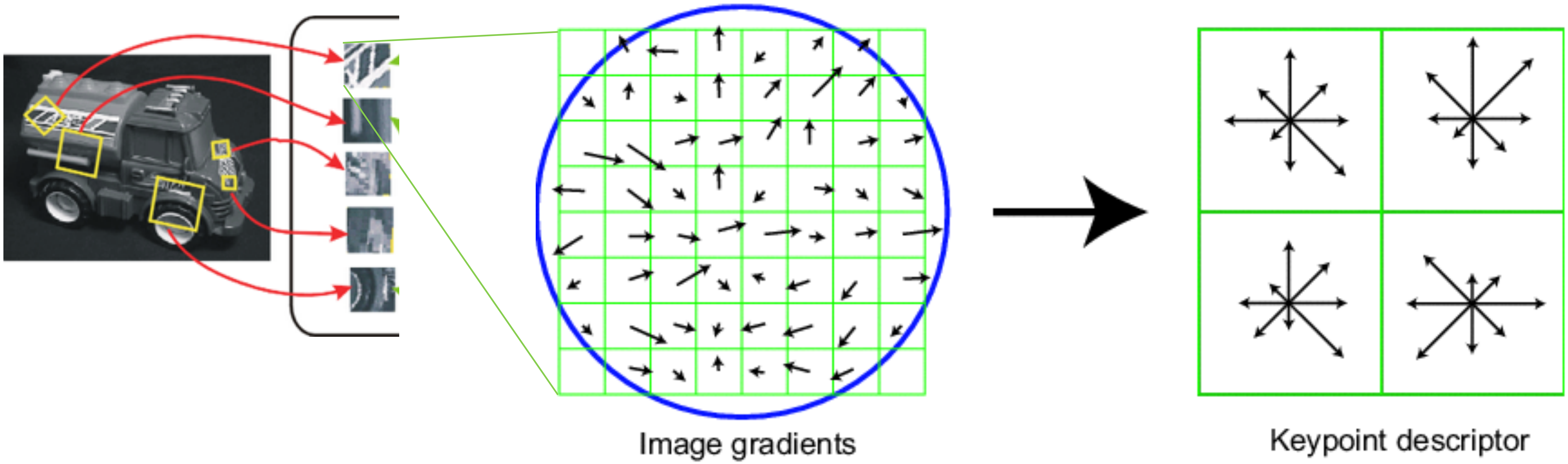
Ensure smoothness

- Gaussian weight
- Trilinear interpolation
  - a given gradient contributes to 8 bins: 4 in space times 2 in orientation



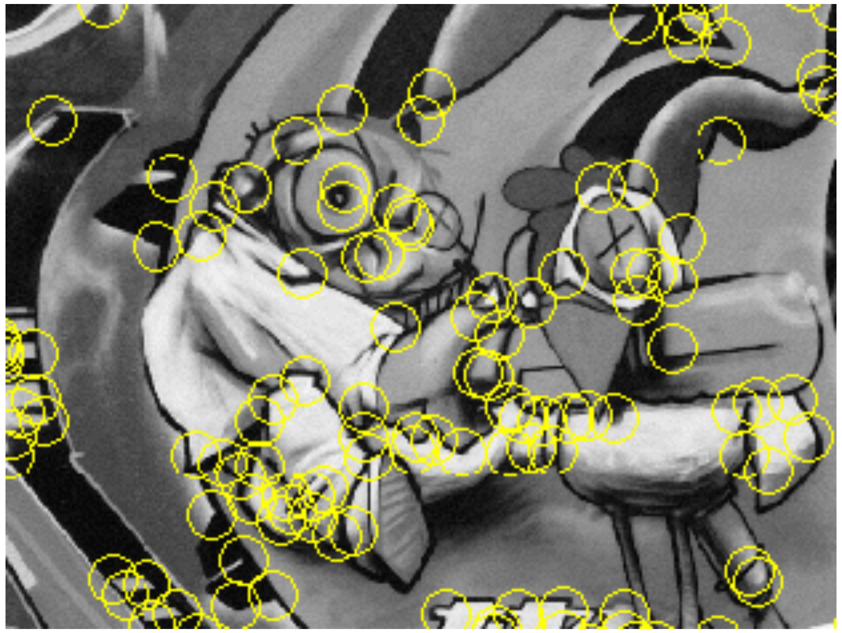
Reduce effect of illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients >0.2
  - renormalize

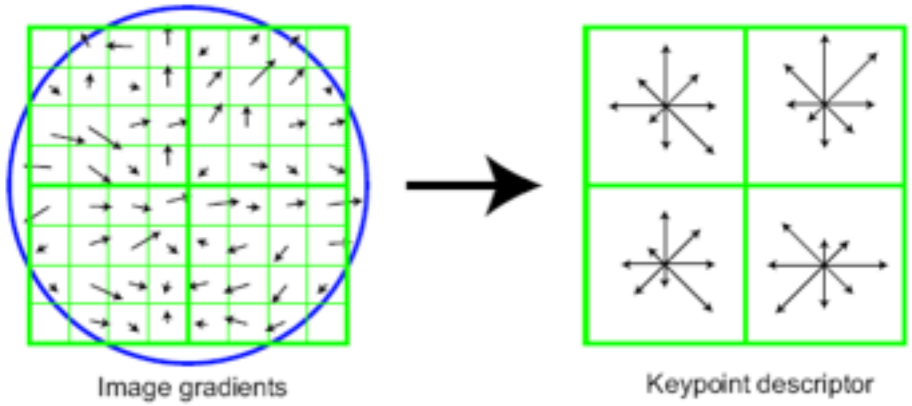


Summary

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



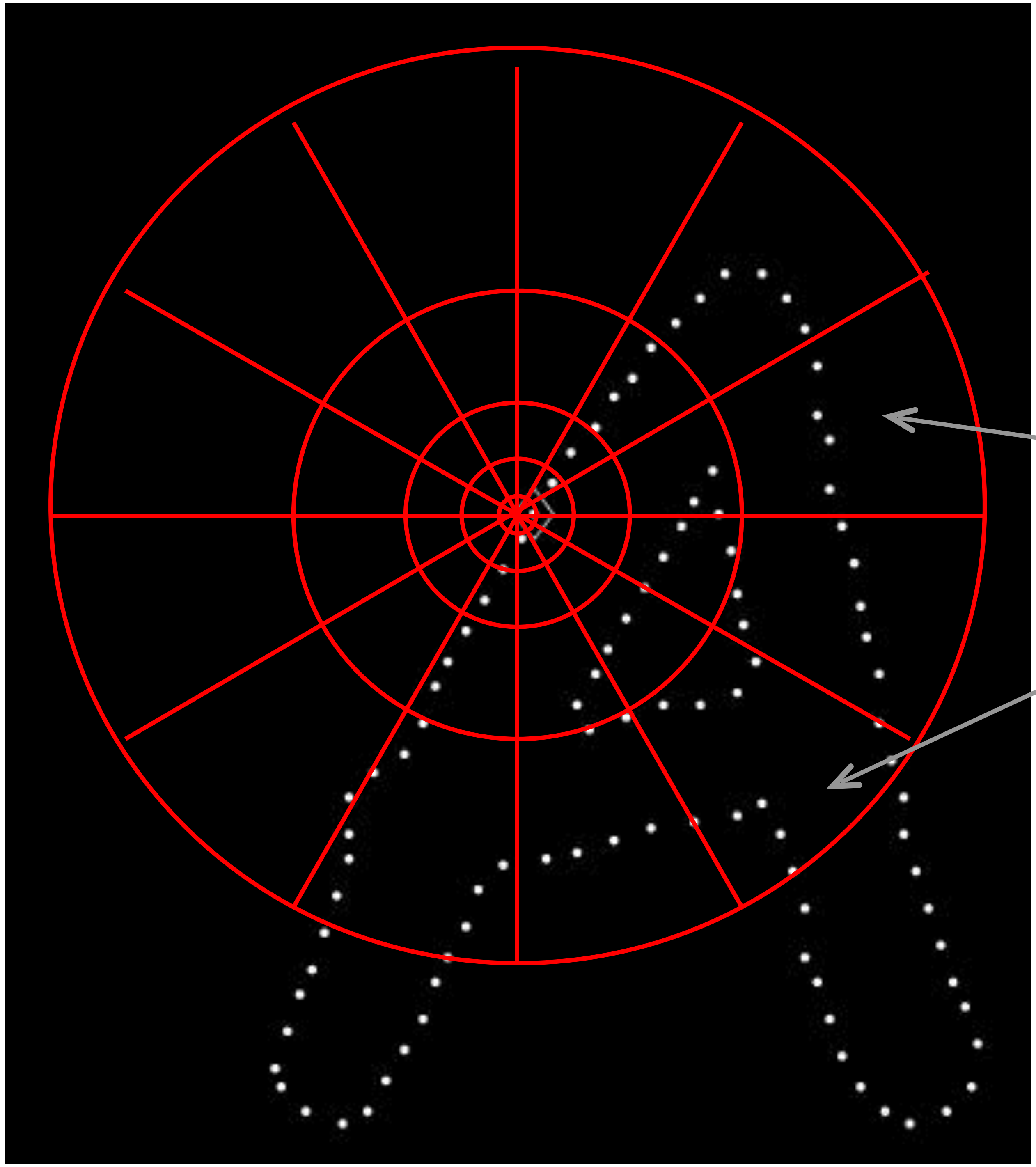
- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT and variants are typically good for stitching and recognition
  - But, need not stick to one



Local Descriptors: Shape Context

Figure 1. Examples of two handwritten digits.

Local Descriptors: Shape Context



Count the number of points inside each bin, e.g.:

Count = 4

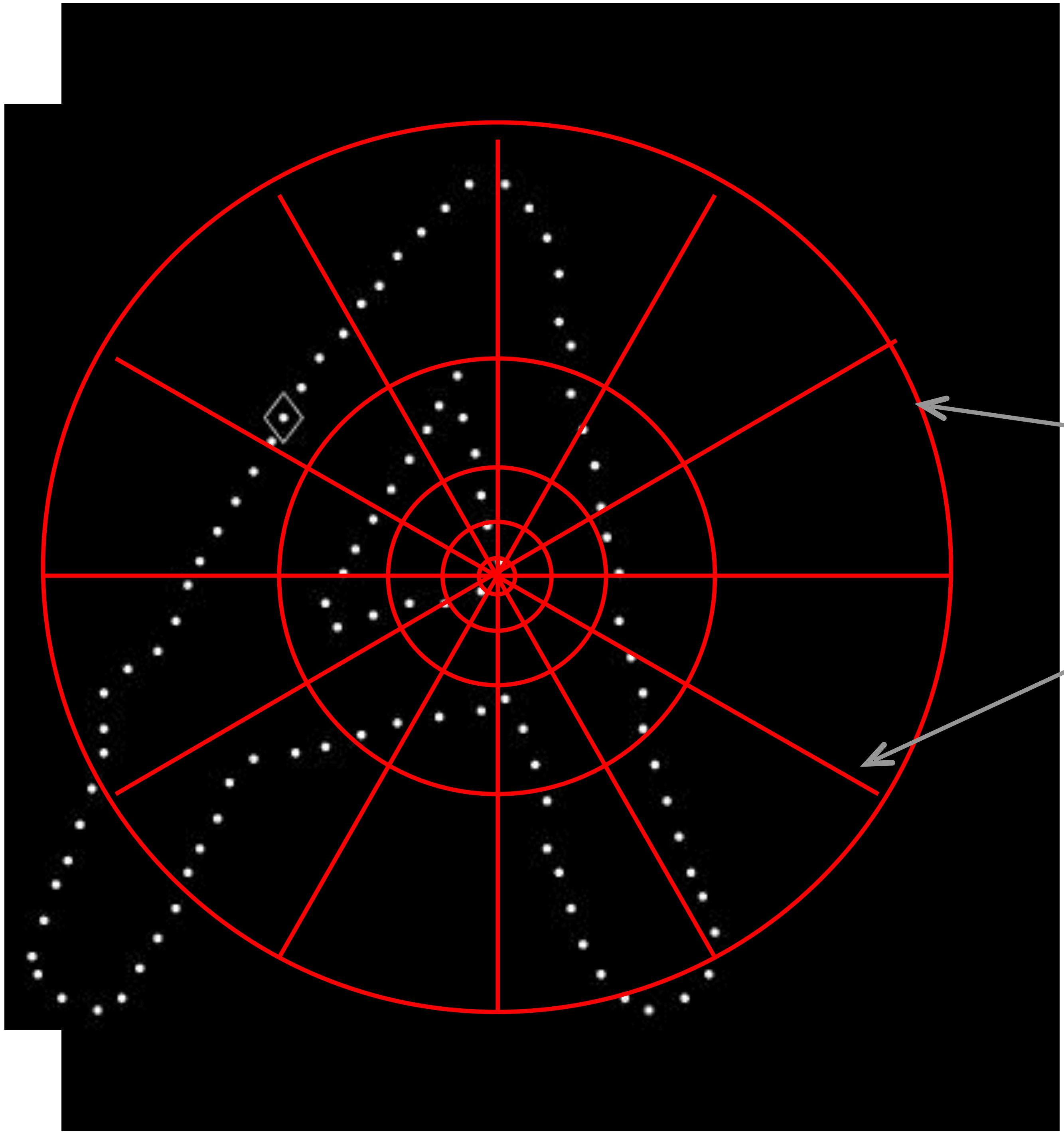
⋮

Count = 10

Log-polar binning: more precision for nearby points, more flexibility for farther points.

Belongie & Malik, ICCV 2001 http://vision.ucsd.edu/sites/default/files/00937552.pdf

Local Descriptors: Shape Context



Count the number of points inside each bin, e.g.:

Count = 0

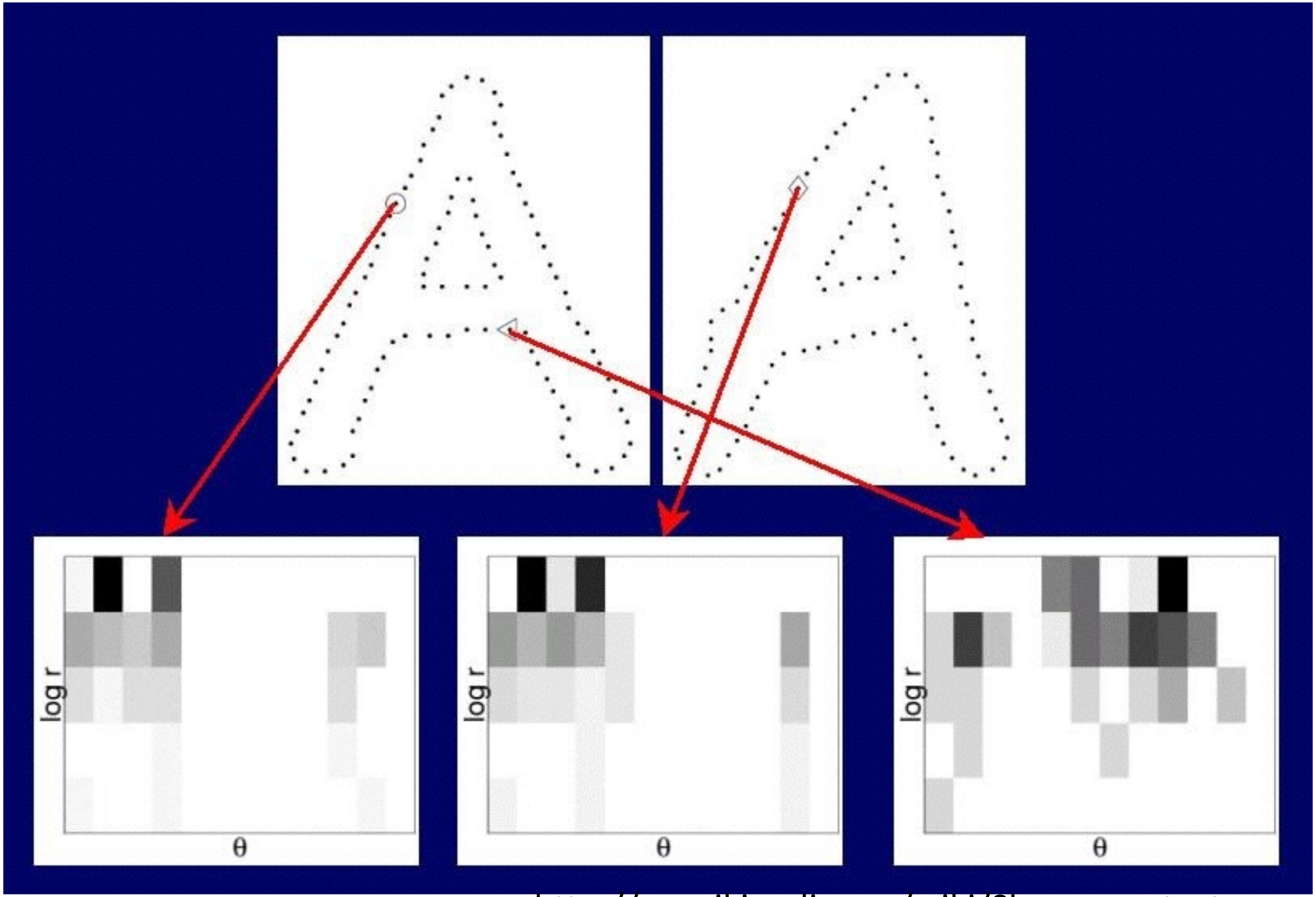
⋮

Count = 6

Log-polar binning: more precision for nearby points, more flexibility for farther points.

Belongie & Malik, ICCV 2001 http://vision.ucsd.edu/sites/default/files/00937552.pdf

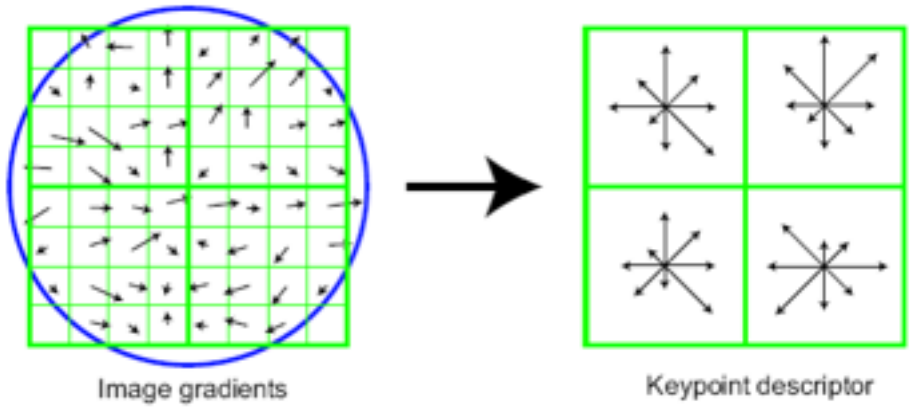
How to find point correspondence?



http://en.wikipedia.org/wiki/Shape_context

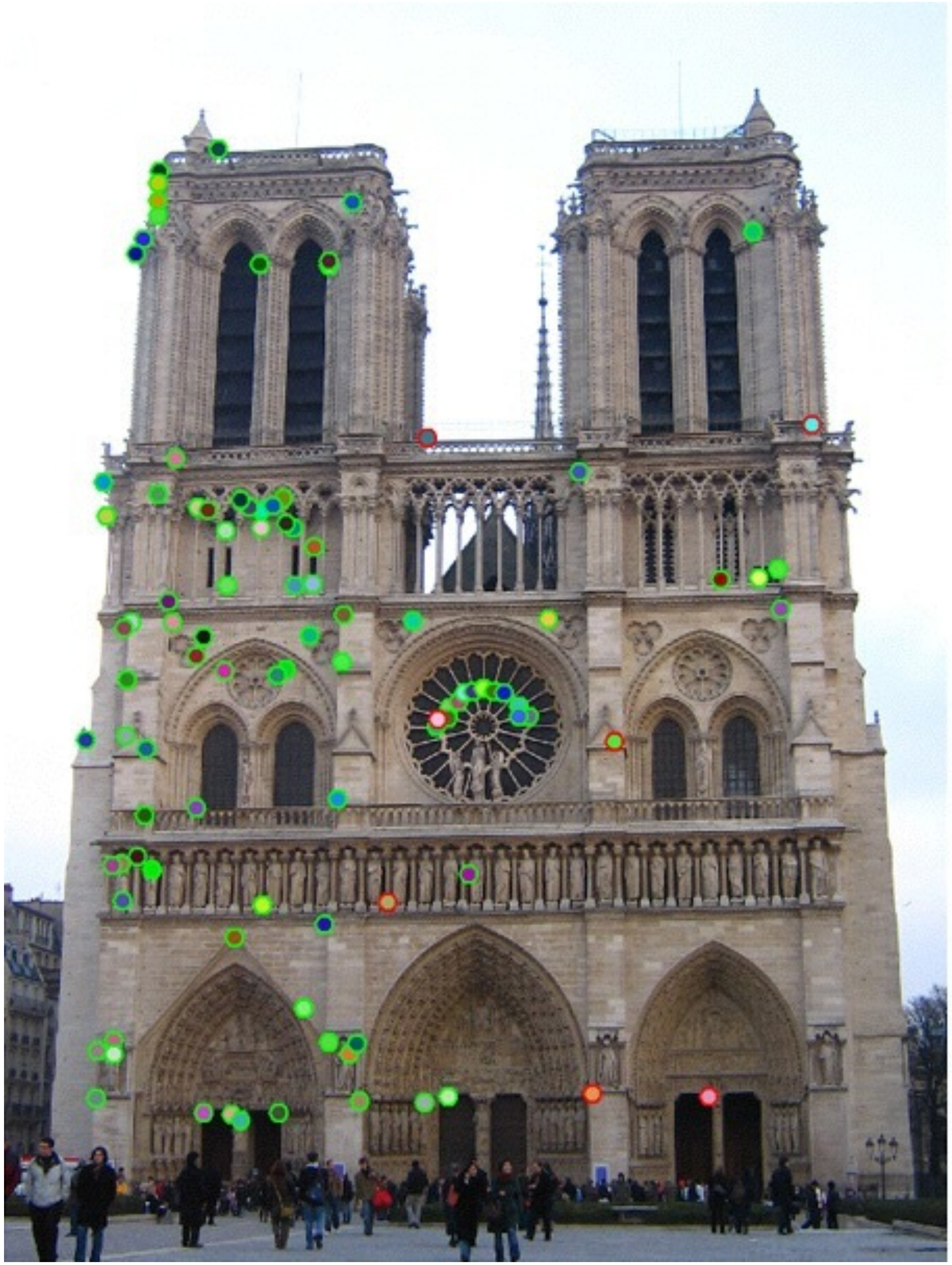
Review: Local Descriptors

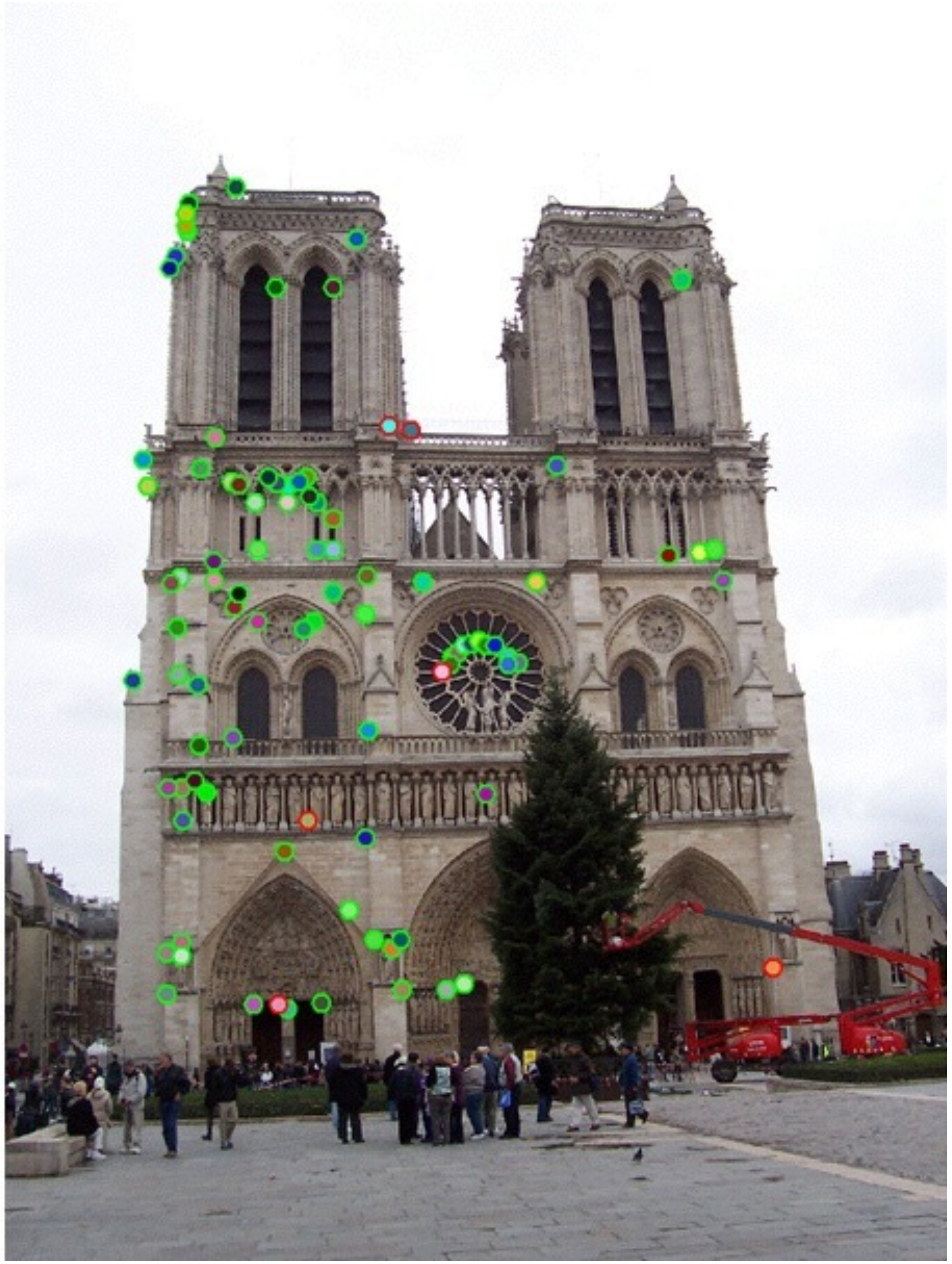
- Most features can be thought of as templates, histograms (counts), or combinations
- The ideal descriptor should be
  - Robust and Distinctive
  - Compact and Efficient



- Most available descriptors focus on edge/ gradient information
  - Capture texture information
  - Color rarely used

How do we decide which features match?





Feature matching

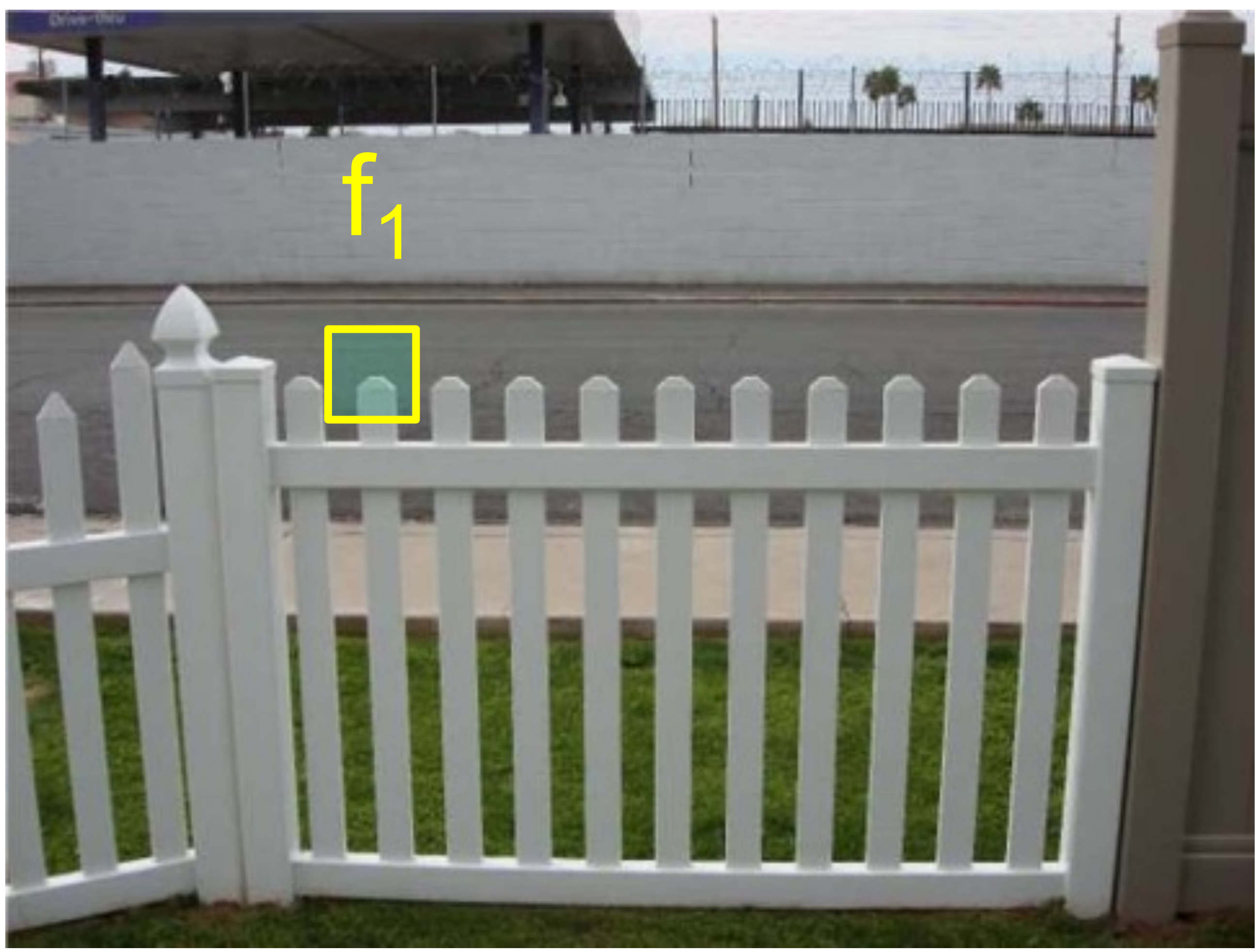
Given a feature in I_1, how to find the best match in I_2?

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

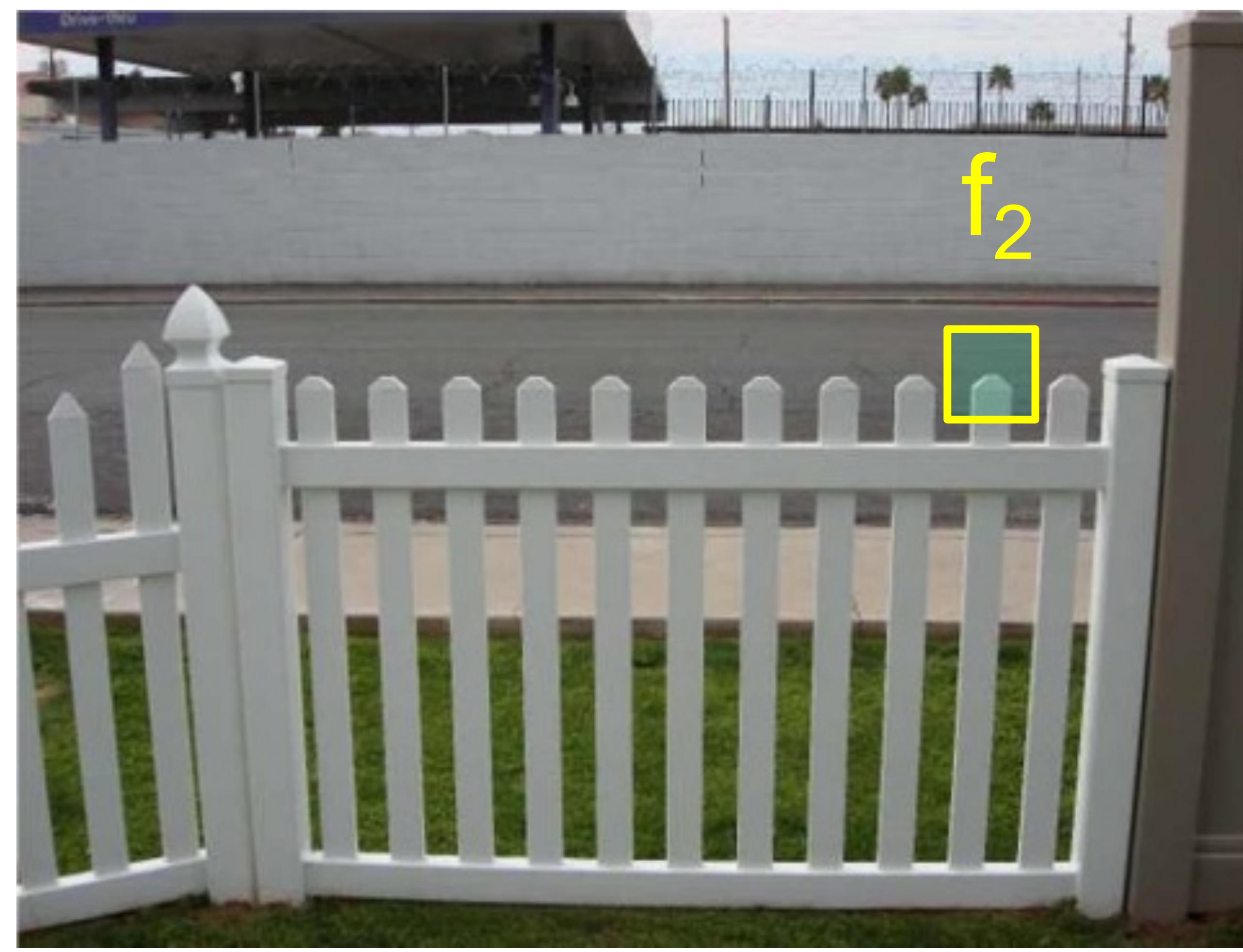
Feature distance

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

- Simple approach: L_2 distance, $||f_1 - f_2||$
- can give good scores to ambiguous (incorrect) matches



I_1

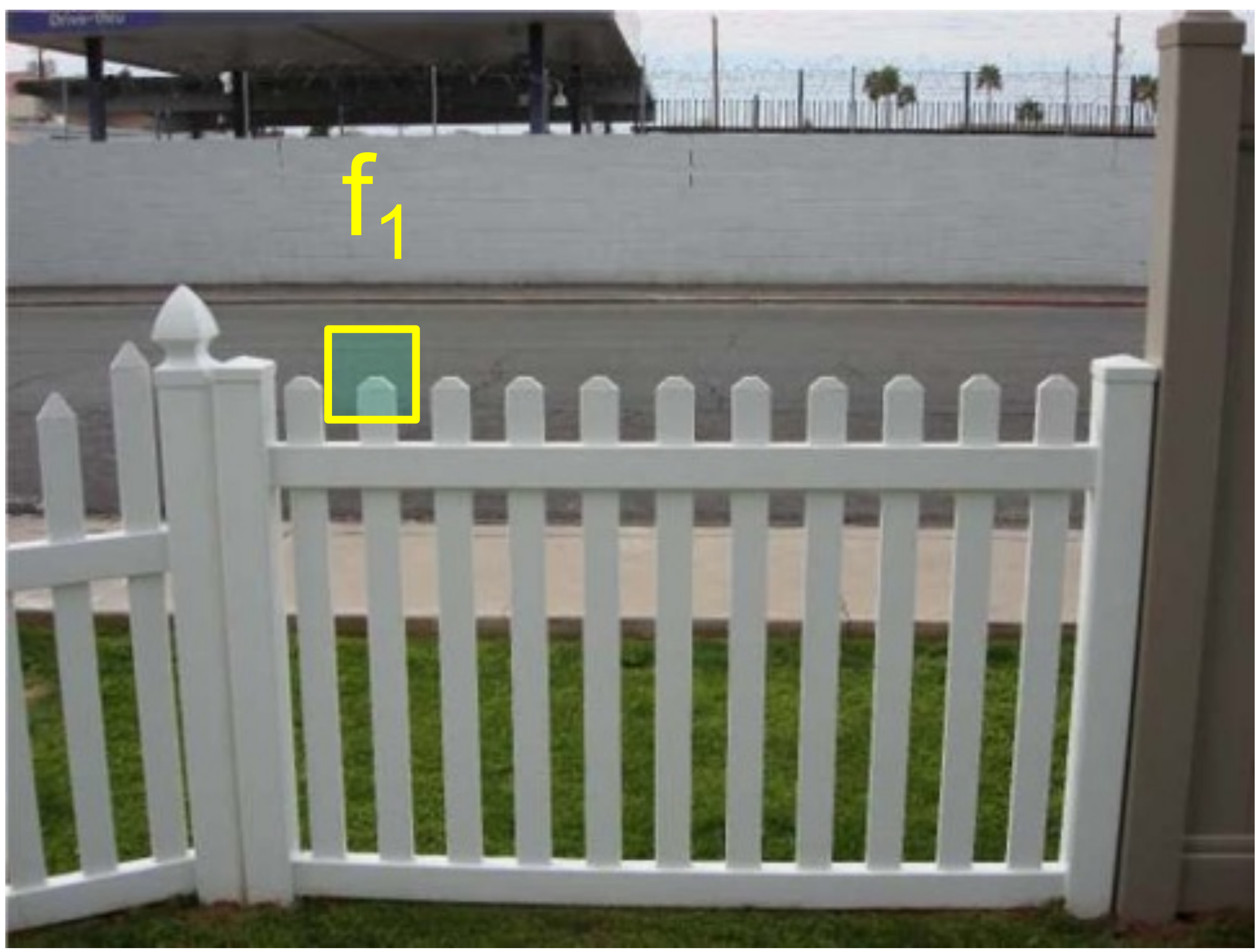


I_2

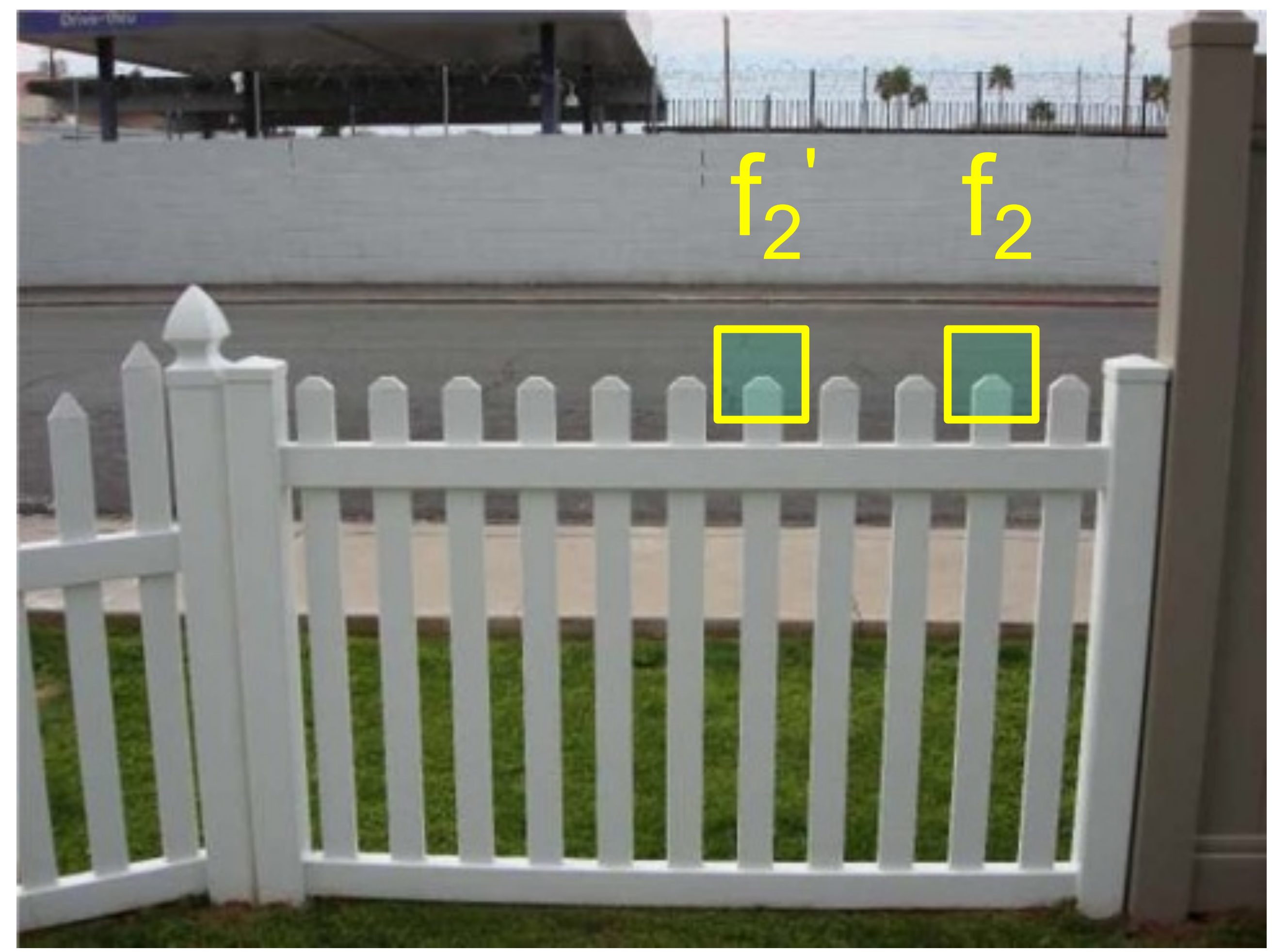
Feature distance

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 best SSD (summed of square distance) match to f_1 in I_2
  - f_2' is 2^{nd} best SSD match to f_1 in I_2
  - gives bad scores for ambiguous matches



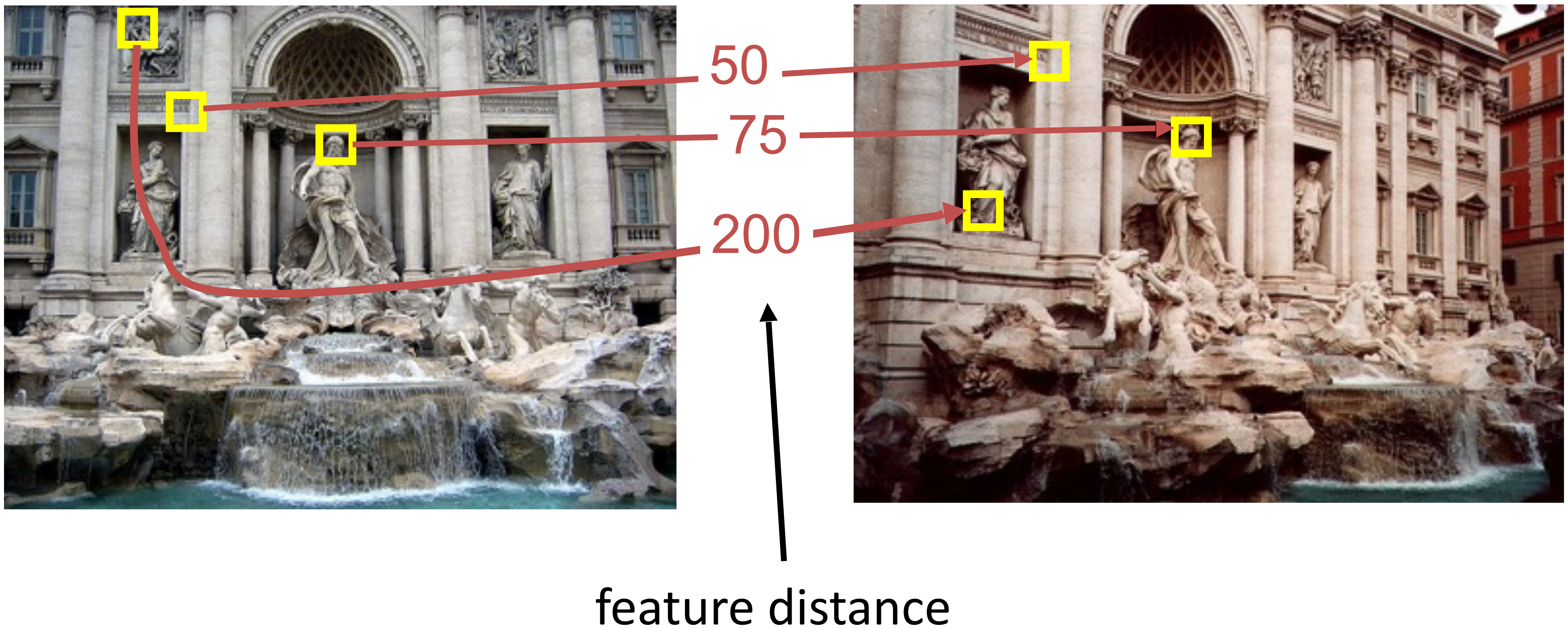
I_1



I_2

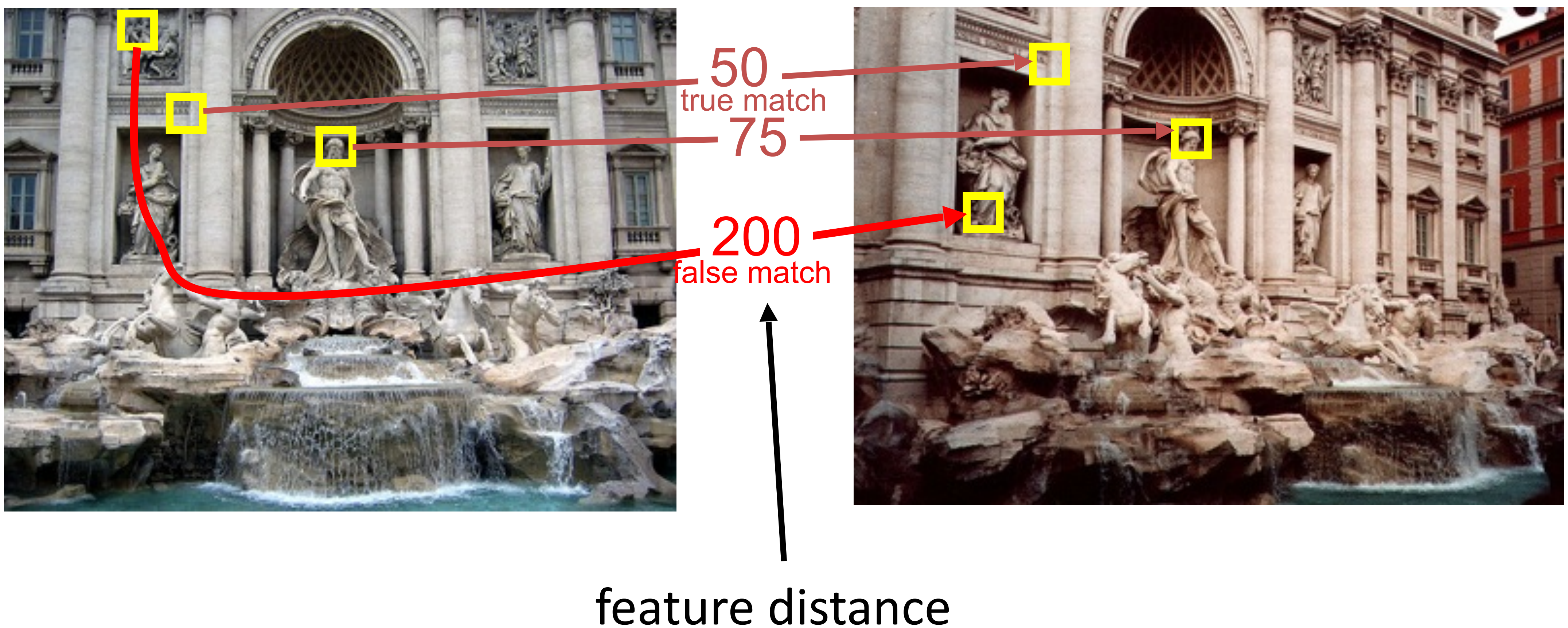
Evaluating the results

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 are correct
  - Suppose we want to maximize these—how to choose threshold?
- **False positives** = # of detected matches that are incorrect
  - Suppose we want to minimize these—how to choose threshold?

A little detour of Pattern Recognition

Precision and Recall

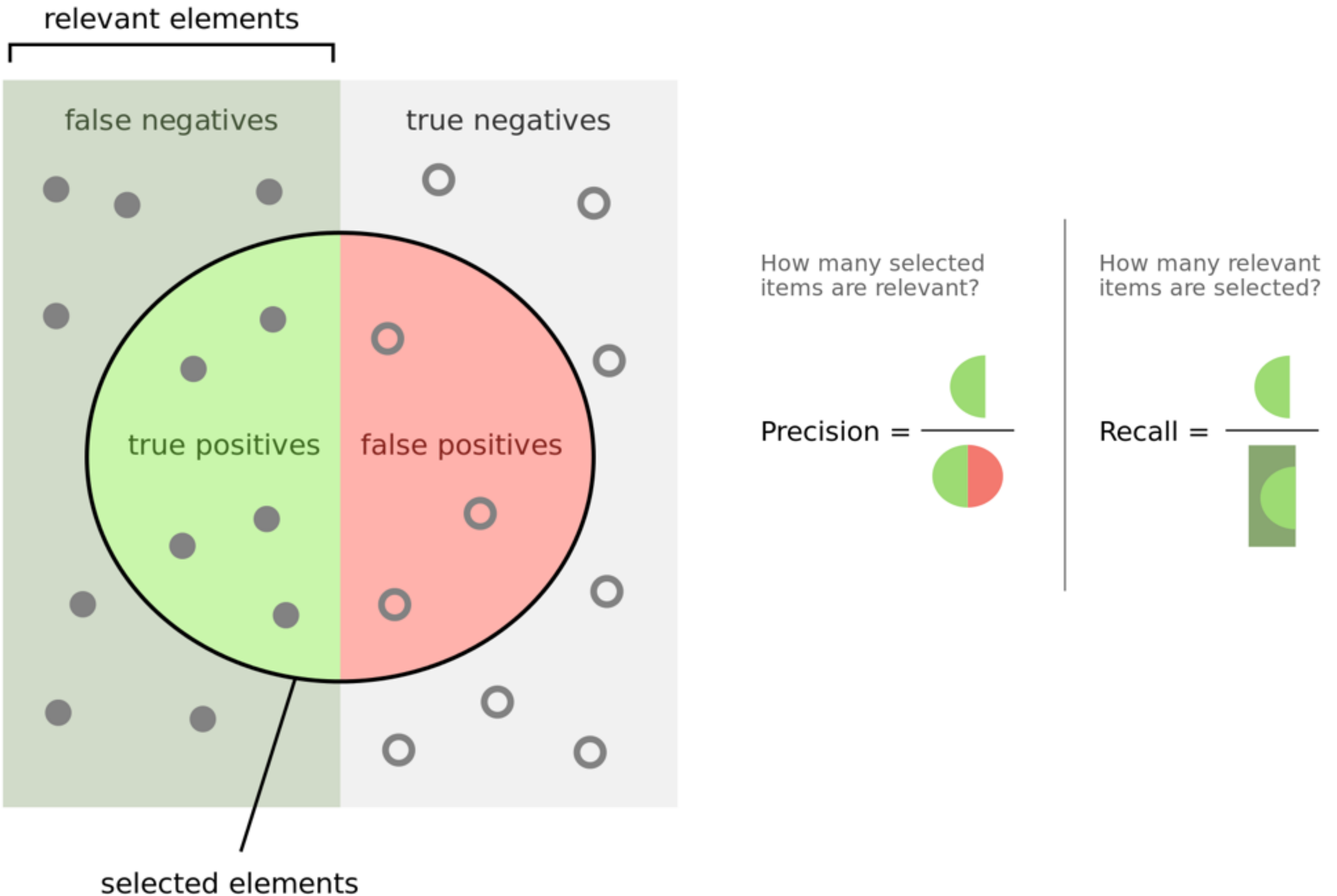
Suppose a video has 9 dogs and some cats

Your algorithm identified 7 dogs

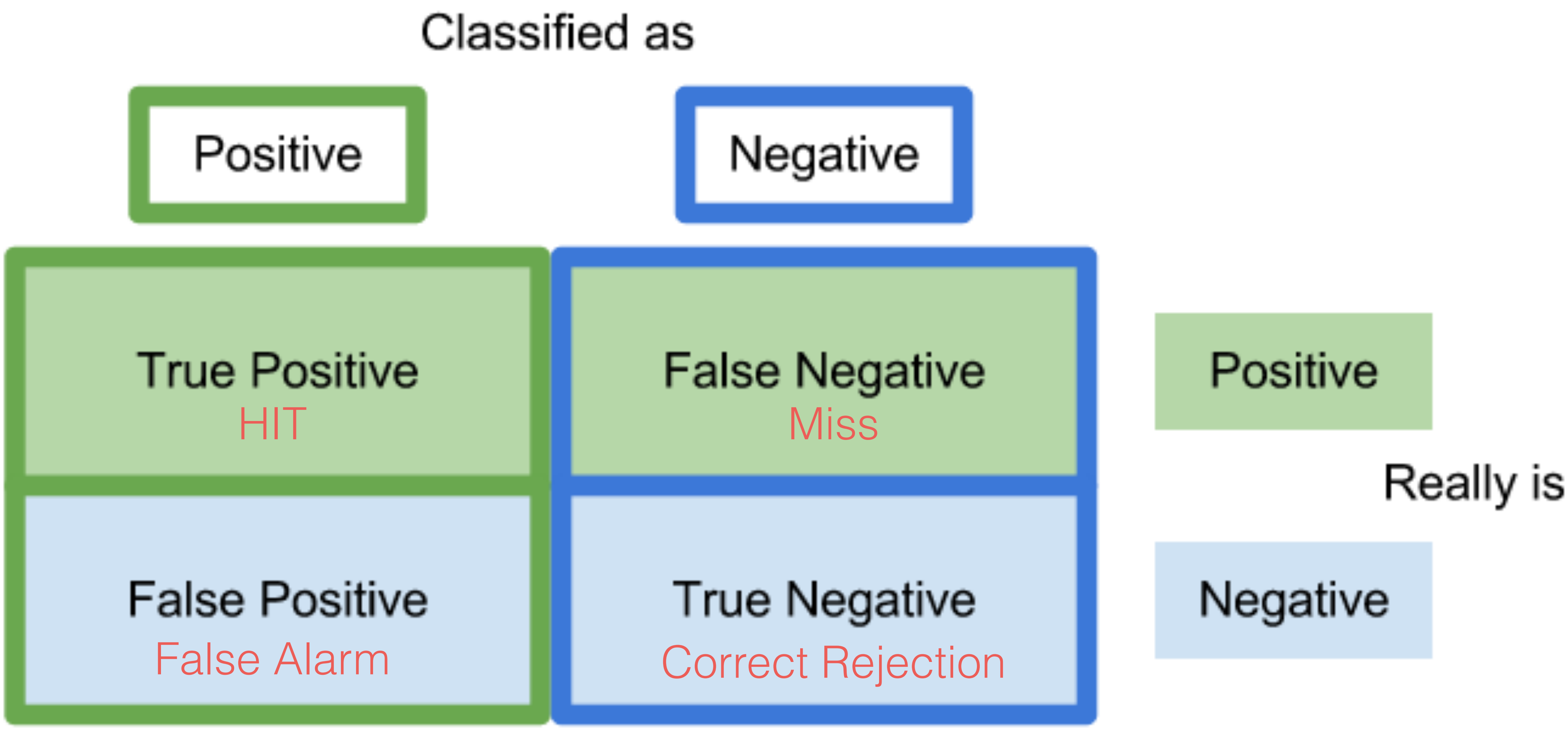
However, only 4 of those are actually dogs

Precision: 4/7, a measure of exactness
Recall: 4/9, a measure of completeness

http://en.wikipedia.org/wiki/Precision_and_recall

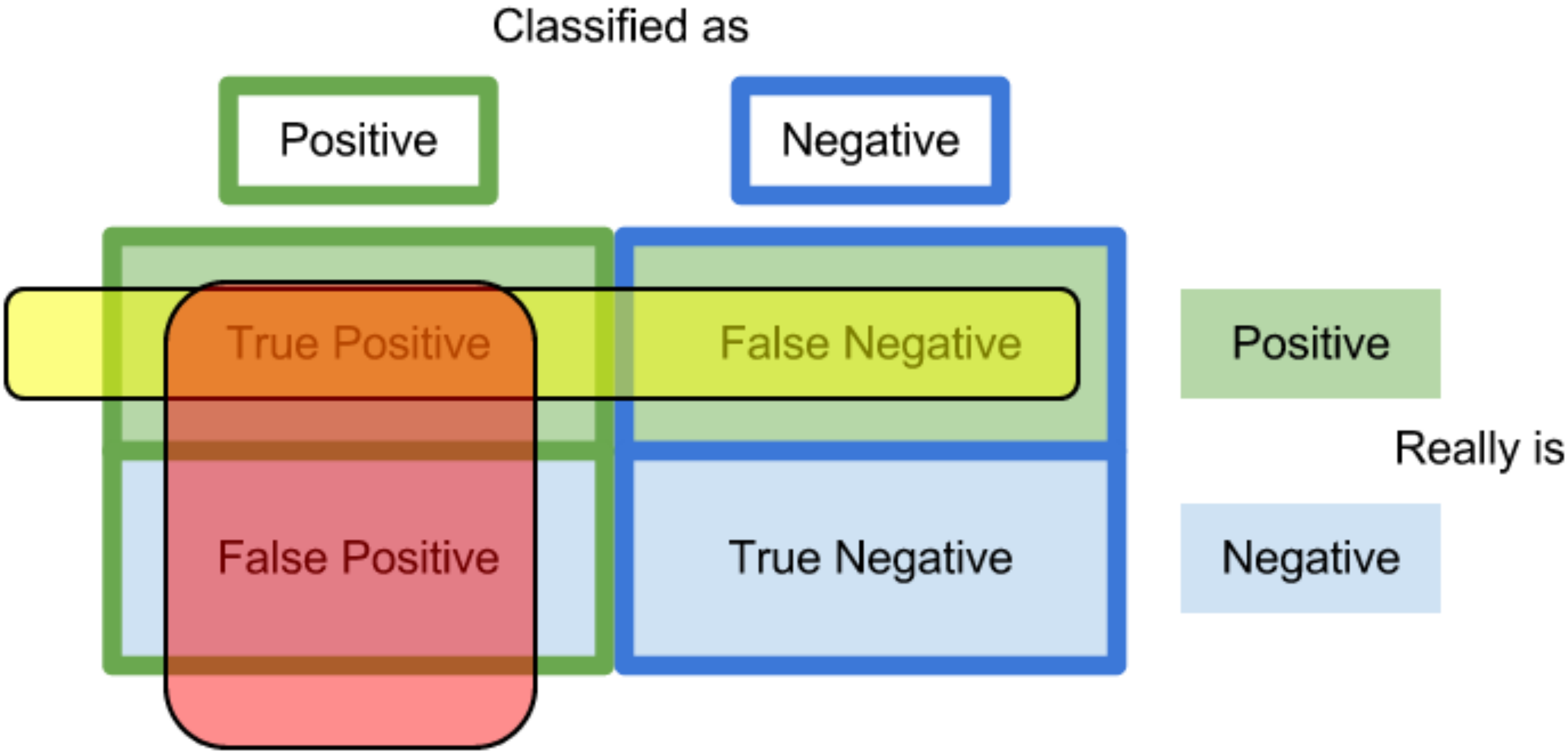

relevant elements
false negatives
true negatives
true positives
false positives
selected elements
How many selected items are relevant?
How many relevant items are selected?
Precision =
Recall =

Confusion Matrix



TP: number of correct matches
FN: matches that were not correctly detected
FP: proposed matches that are incorrect
TN: non matches that were correctly rejected

https://uberpython.wordpress.com/2012/01/01/precision-recall-sensitivity-and-specificity/



— Precision in red, recall in yellow

$$\text{Precision} = (TP)/(TP+FP)$$

$$\text{Recall} = (TP)/(TP+FN)$$

https://uberpython.wordpress.com/2012/01/01/precision-recall-sensitivity-and-specificity/

Precision vs. Recall

1000 animals, 100 dogs

Algorithm finds 50 (of which 40 are dogs, 10 are cats)

TP =	FN =
FP =	TN =

TP True Positive

FP False Positive

FN False Negative

TN True Negative

Precision vs. Recall

1000 animals, 100 dogs

Algorithm finds 50 (of which 40 are dogs, 10 are cats)

Total= 100	TP =40	FN =60	100
Total= 100	FP = 10	TN =890	900
			1000

TP True Positive

FP False Positive

FN False Negative

TN True Negative

Precision = TP/(FP+TP) = 40/50

Recall = TP/(FN+TP) = 40/100

Precision vs. Recall

Examples

1000 animals, 100 dogs

Algorithm finds 50 (of which 40 are dogs, 10 are cats)

Precision =

Recall =

Algorithm finds 10 (of which 10 are dogs)

Precision =

Recall =

Algorithm returns 1000 (of which 100 are dogs)

Precision =

Recall =

Quiz

Assume the following:

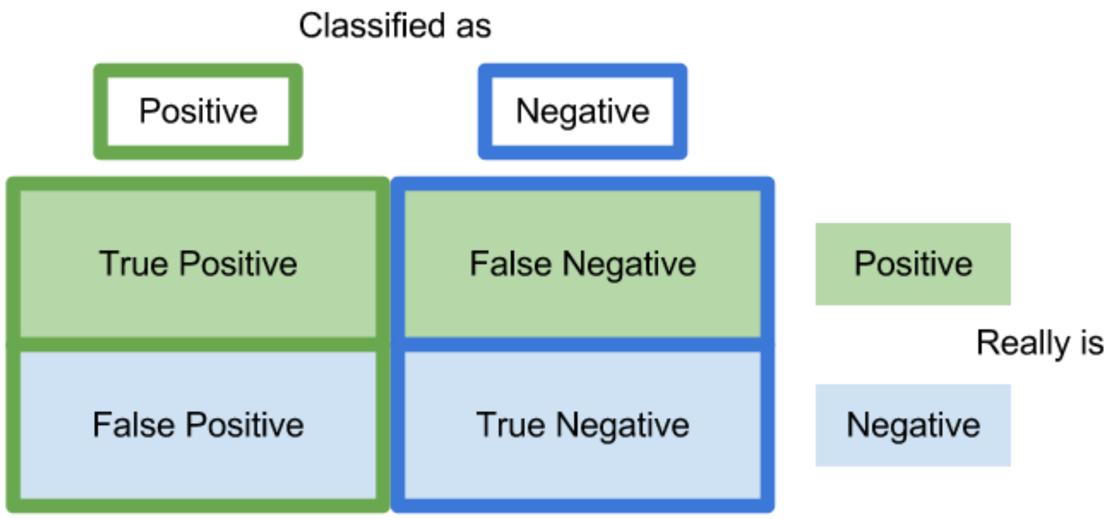
- A database contains **80** records on a particular topic
- A search was conducted on that topic and **60** records were retrieved.
- Of the 60 records retrieved, **45** were relevant.

What is precision and recall?

1. Take a piece of paper out and construct the confusion matrix.
2. Compute precision and recall

Precision = TP/(FP+TP) =

Recall = TP/(FN+TP) =

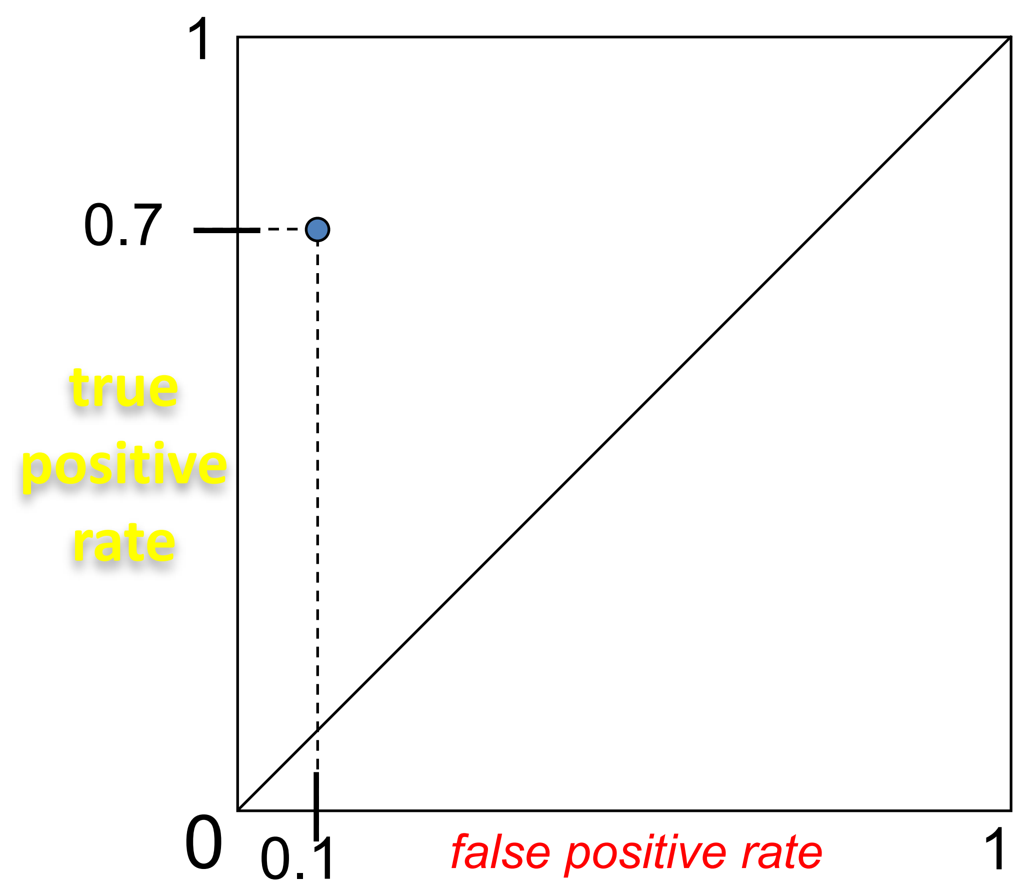


Evaluating the results

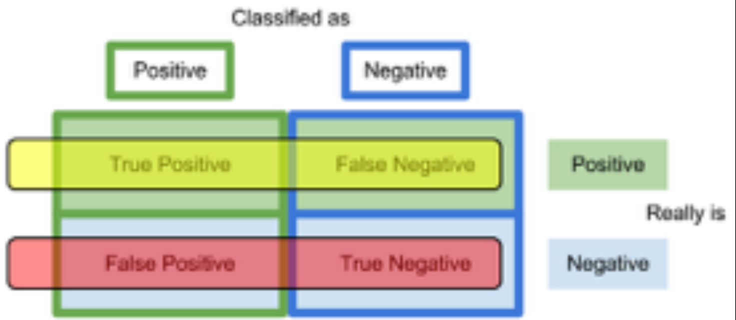
How can we measure the performance of a feature matcher?

$$\text{true positive rate} = \frac{\text{\# true positives}}{\text{\# matching features (positives)}}$$

"recall"



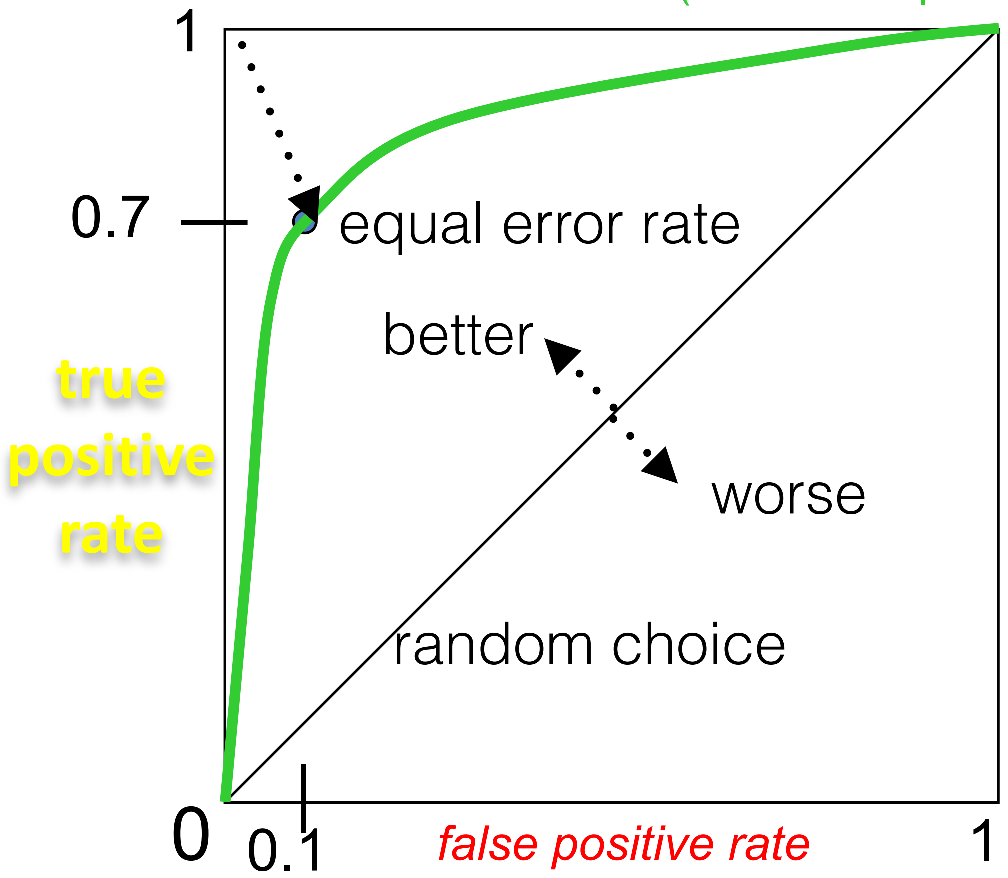
$$\text{false positive rate} = \frac{\text{\# false positives}}{\text{\# unmatched features (negatives)}}$$



Evaluating the results

How can we measure the performance of a feature matcher?

ROC curve ("Receiver Operator Characteristic")



$$\frac{\text{\# true positives}}{\text{\# matching features (positives)}}$$

"recall"

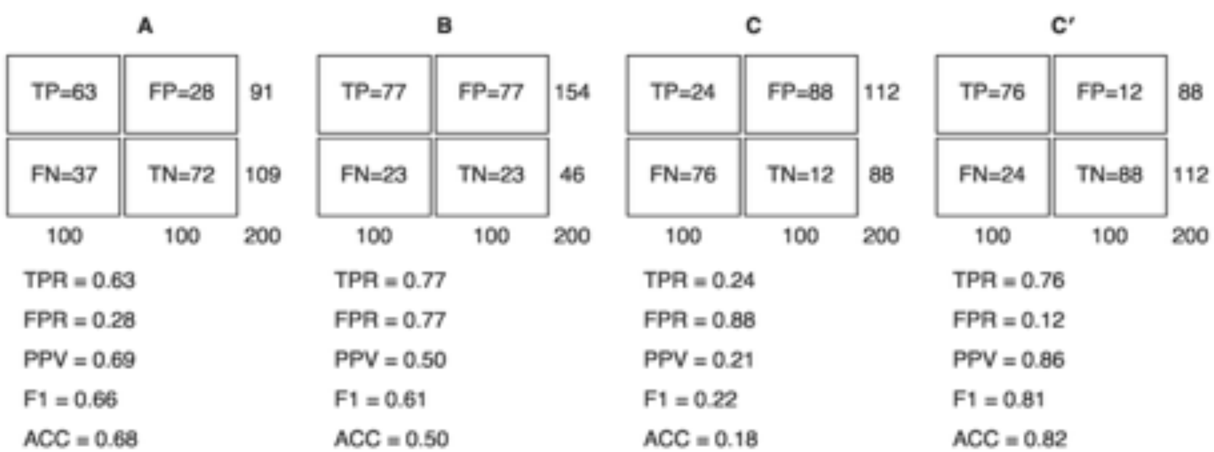
$$\frac{\text{\# false positives}}{\text{\# unmatched features (negatives)}}$$

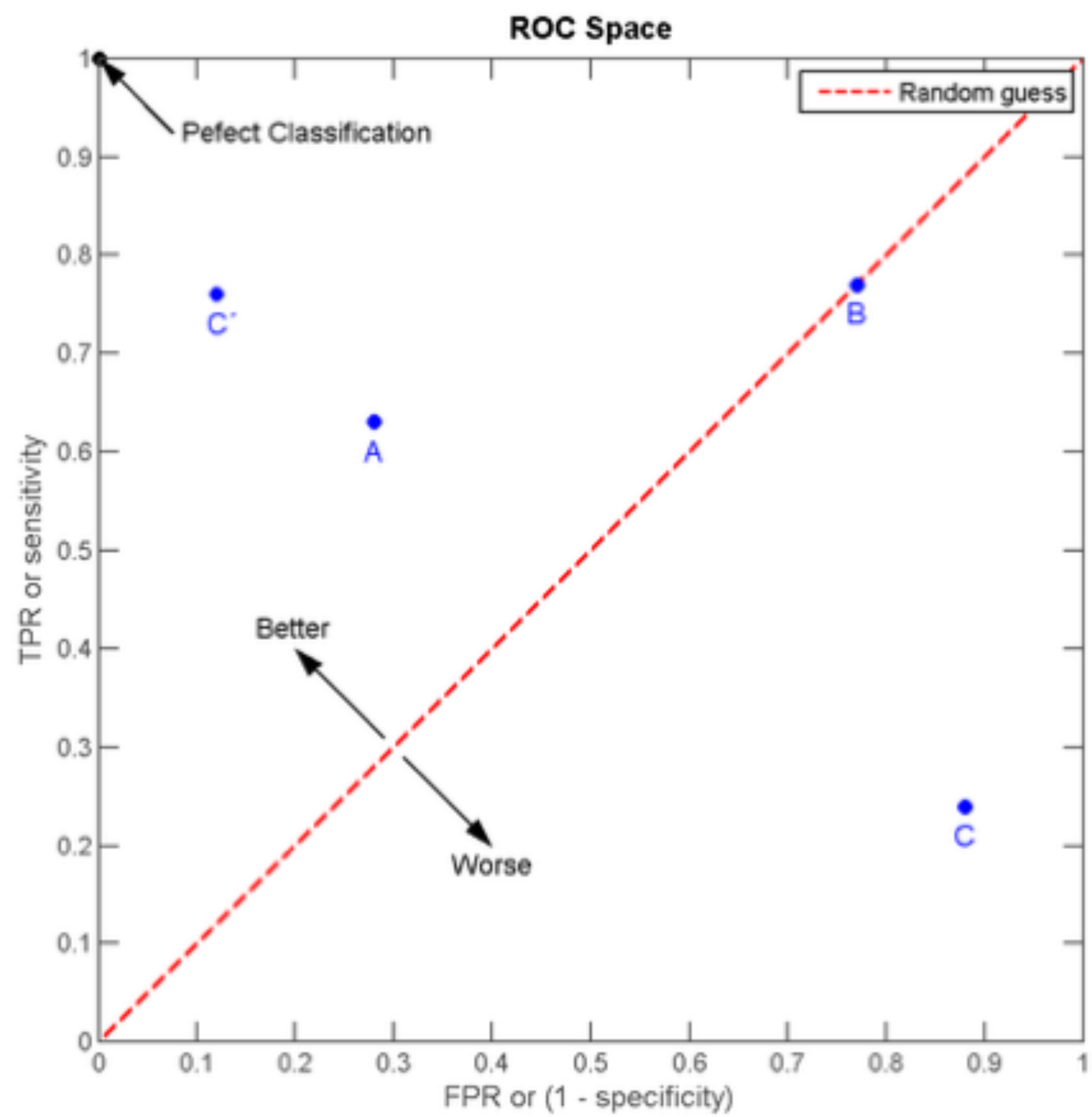
AUC (area under curve)

ROC Curves

- Generated by counting # correct/incorrect matches, for different thresholds
- Want to maximize area under the curve (AUC)
- Useful for comparing different feature matching methods
- For more info: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

Evaluating the results





http://en.wikipedia.org/wiki/Receiver_operating_characteristic

More on feature detection/description

Address http://www.robots.ox.ac.uk/~vgg/research/affine/

Google mikolajczyk Search Web

Affine Covariant Regions

Publications

Region detectors

- *Harris-Affine & Hessian Affine*: K. Mikolajczyk and C. Schmid, Scale and Affine invariant interest point detectors. In IJCV 1(60):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 onaffine invariant regions. In IJCV 1(59):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

Region descriptors

- *SIFT*: D. Lowe, Distinctive image features from scale invariant keypoints. In IJCV 2(60):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. Technical Report, accepted to IJCV. PDF
- K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors. Technical Report, accepted to PAMI. PDF

3D Reconstruction

Internet Photos ("Colosseum")

Reconstructed 3D cameras and points

Object recognition (David Lowe)

Sony Aibo

SIFT usage:

- Recognize charging station
- Communicate with visual cards
- Teach object recognition

Available at a web site near you…

- For most local feature detectors, executables are available online:
  - http://www.robots.ox.ac.uk/~vgg/research/affine
  - http://www.cs.ubc.ca/~lowe/keypoints/
  - http://www.vision.ee.ethz.ch/~surf

SIFT feature implementation

```
import cv2
import numpy as np


img = cv2.imread('home.jpg')
gray= cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)


sift = cv2.SIFT()
kp = sift.detect(gray,None)

img=cv2.drawKeypoints(gray,kp)

cv2.imwrite('sift_keypoints.jpg',img)
```

Questions?

Image alignment