Feature descriptors and matching

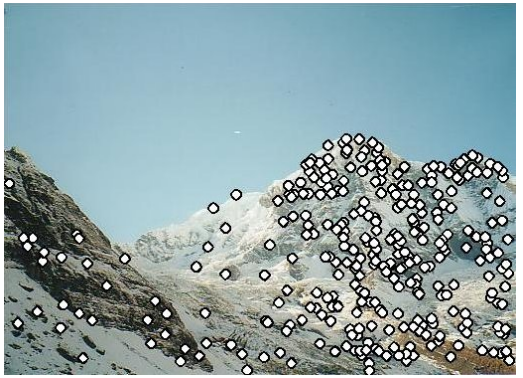
<Vision System>

Department of Robot Engineering

Prof. Younggun Cho

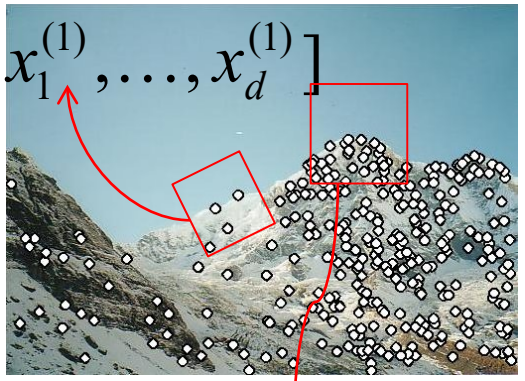
Local features: main components

1) Detection: Identify the interest points



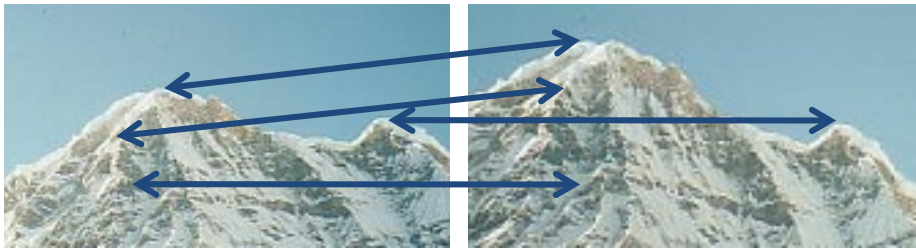
2) **Description: Extract vector feature descriptor surrounding each interest point.**

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



3) Matching: Determine correspondence between descriptors in two views

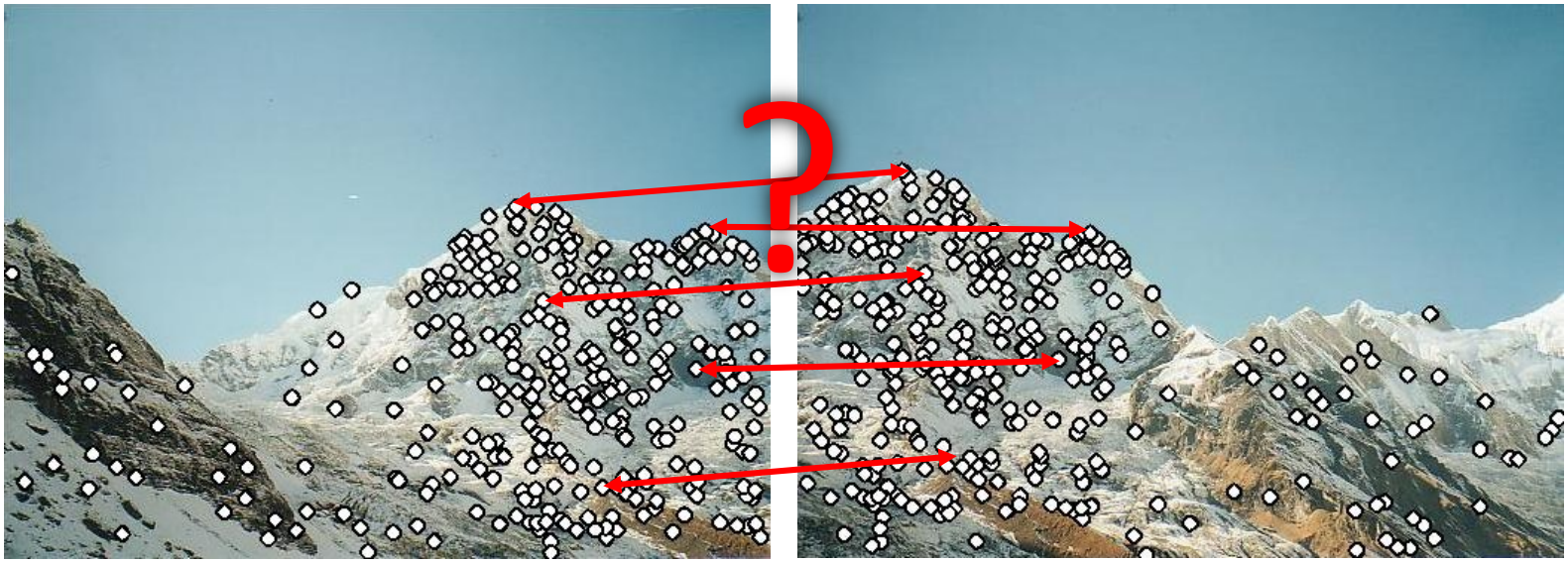
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$



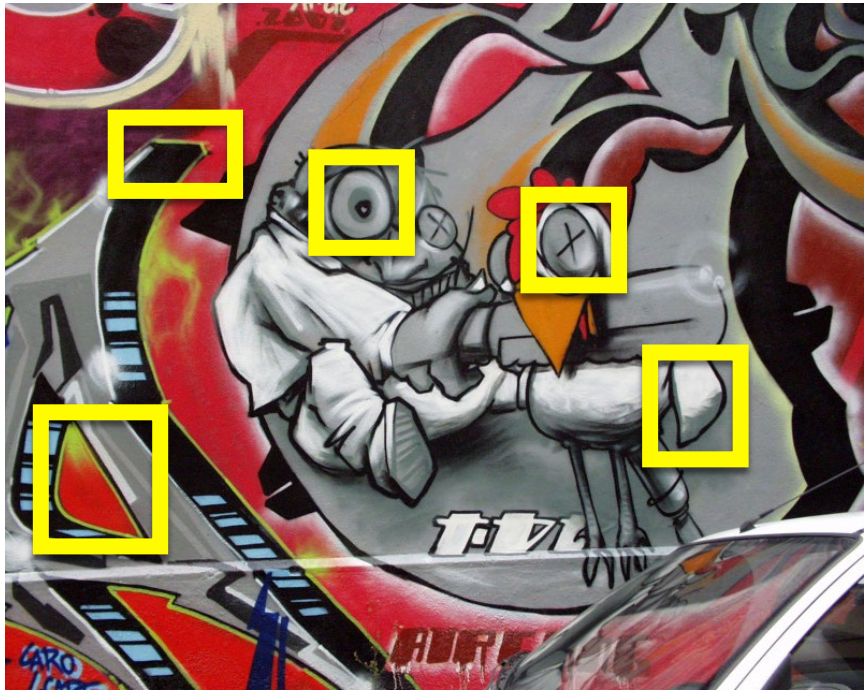
Feature descriptors

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



If we know where the good features are,
how do we match them?

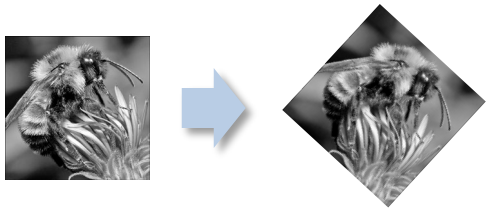
Invariance vs. discriminability

- Invariance:
  - Descriptor shouldn’t change even if image is transformed
- Discriminability:
  - Descriptor should be highly unique for each point

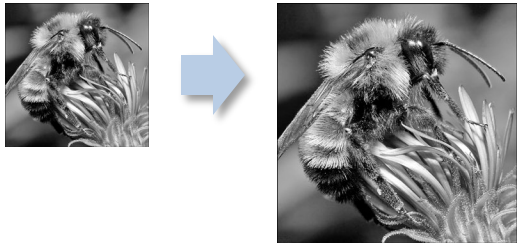
Image transformations revisited

- Geometric

Rotation

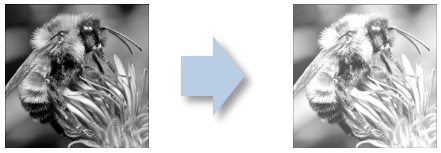


Scale

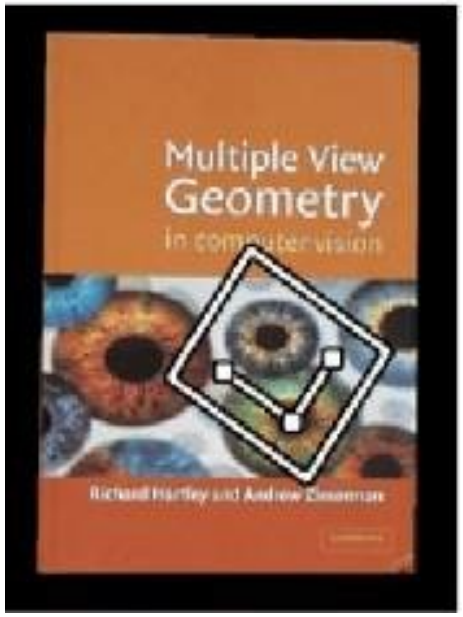


- Photometric

Intensity change



Geometric transformations



objects will appear at different scales,
translation and rotation

Photometric transformations

Invariant descriptors

- We looked at invariant / equivariant **detectors**
- Most feature descriptors are also designed to be invariant to
  - Translation, 2D rotation, scale
- They can usually also handle
  - Limited 3D rotations (SIFT works up to about 60 degrees)
  - Limited affine transforms (some are fully affine invariant)
  - Limited illumination/contrast changes

How to achieve invariance

Need both of the following:

1. Make sure your detector is invariant
2. Design an invariant feature descriptor
   - Simplest descriptor: a single 0
     - What's this invariant to?
   - Next simplest descriptor: a square, axis-aligned 5x5 window of pixels
     - What's this invariant to?
   - Let's look at some better approaches…

Descrptors

Image patch

Just use the pixel values of the patch

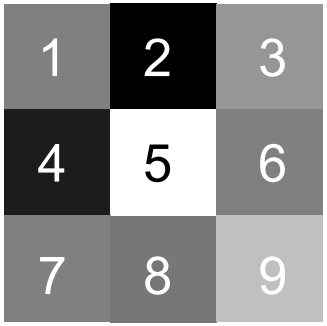
Perfectly fine if geometry and appearance is unchanged (a.k.a. template matching)

Tiny Images

Just down-sample it!
Simple, fast, robust to small affine transforms.

Image patch

Just use the pixel values of the patch



vector of intensity values

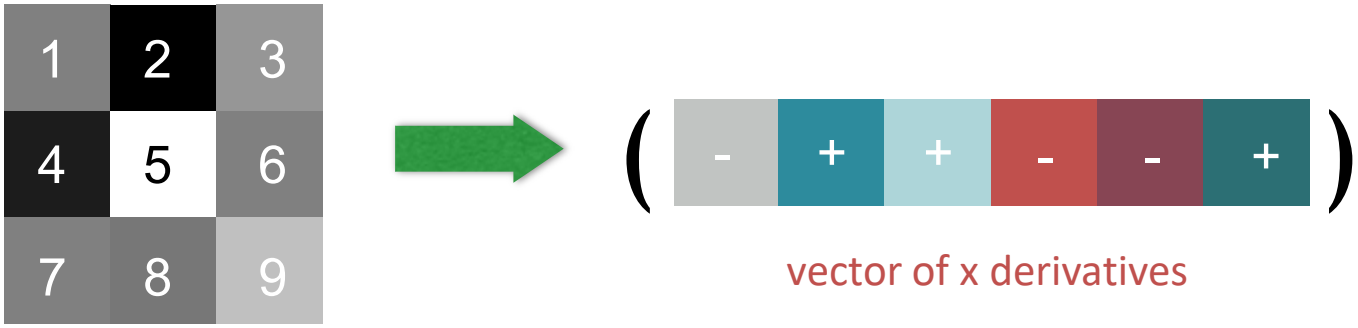
Perfectly fine if geometry and appearance is unchanged (a.k.a. template matching)

What are the problems?

How can you be less sensitive to absolute intensity values?

Image gradients

Use pixel differences



vector of x derivatives

'binary descriptor'

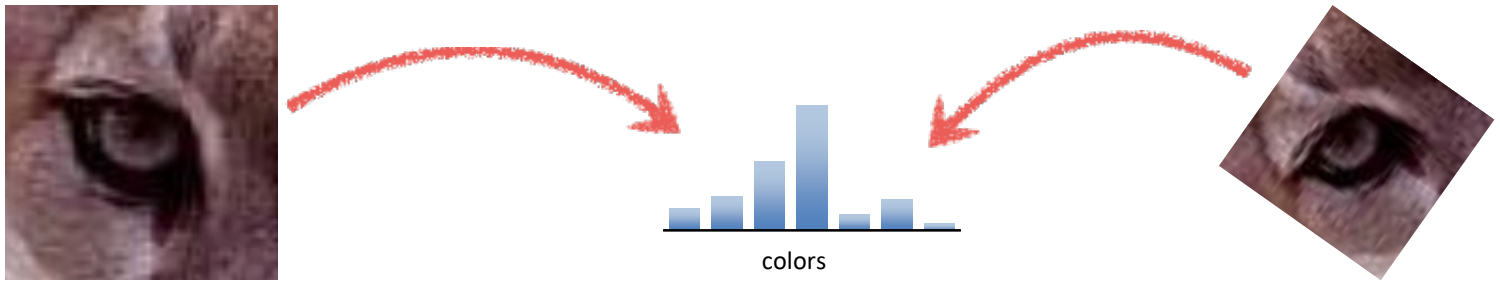
Feature is invariant to absolute intensity values

What are the problems?

How can you be less sensitive to deformations?

Color histogram

Count the colors in the image using a histogram

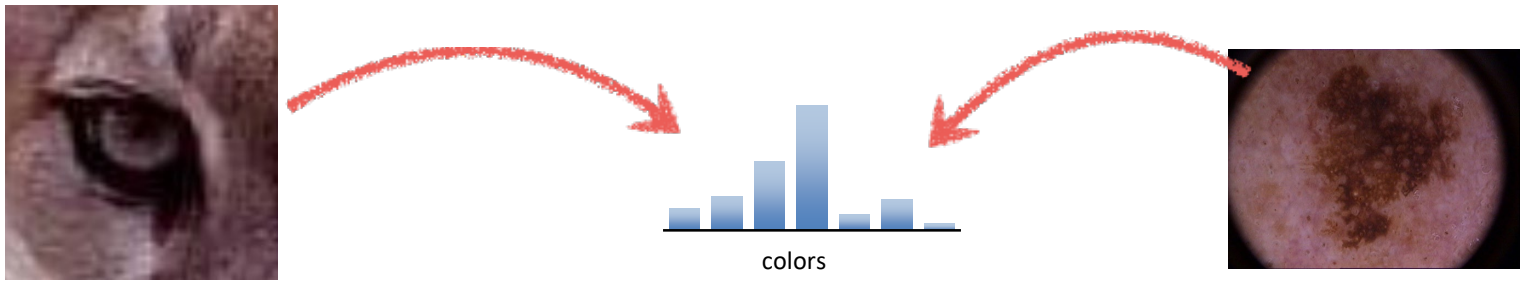


Invariant to changes in scale and rotation

What are the problems?

Color histogram

Count the colors in the image using a histogram

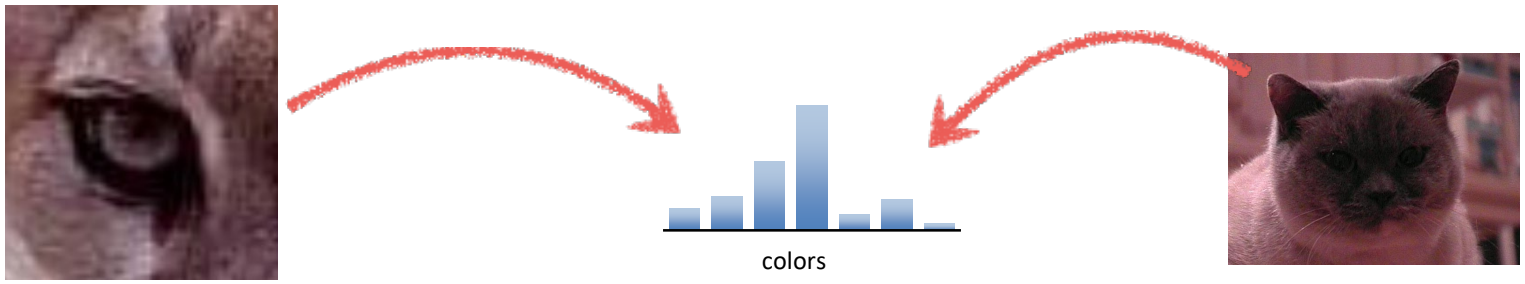


Invariant to changes in scale and rotation

What are the problems?

Color histogram

Count the colors in the image using a histogram



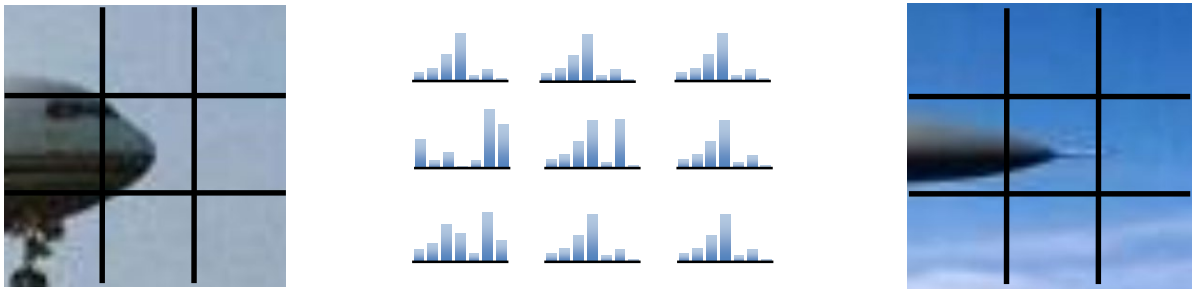
Invariant to changes in scale and rotation

What are the problems?

How can you be more sensitive to spatial layout?

Spatial histograms

Compute histograms over spatial ‘cells’



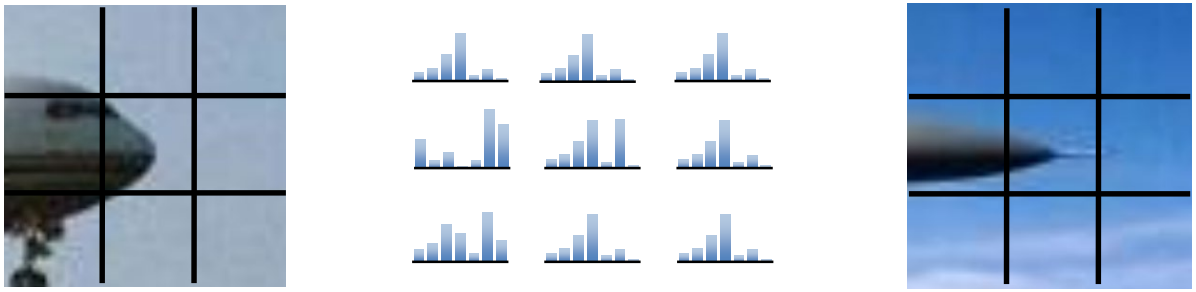
Retains rough spatial layout

Some invariance to deformations

What are the problems?

Spatial histograms

Compute histograms over spatial ‘cells’



Retains rough spatial layout

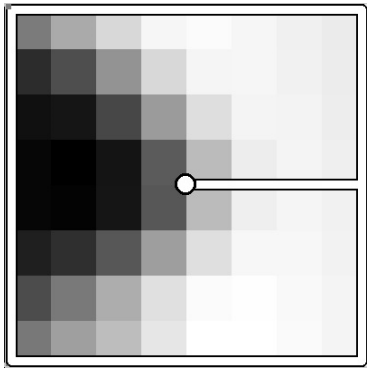
Some invariance to deformations

What are the problems?

How can you be completely invariant to rotation?

Orientation normalization

Use the dominant image gradient direction to normalize the orientation of the patch



save the orientation angle θ along with (x, y, s)

What are the problems?

Rotation invariance for feature descriptors

- Find dominant orientation of the image patch
  - E.g., given by $\mathbf{x}_{max}$, the eigenvector of **H** corresponding to λ_{max} (the *larger* eigenvalue)
  - Or simply the orientation of the (smoothed) gradient
  - Rotate the patch according to this angle

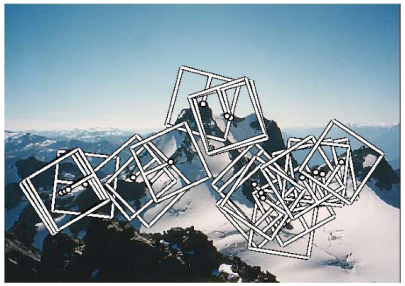
Figure by Matthew Brown

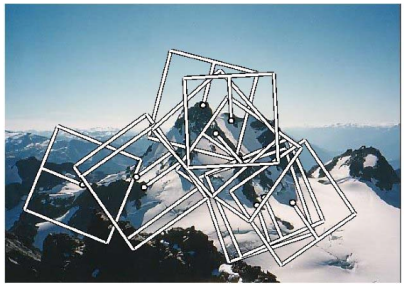
All we got?

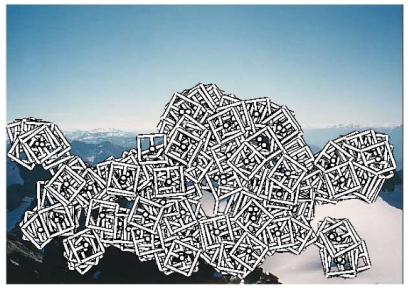
MOPS descriptor

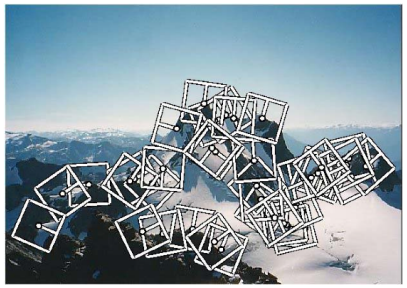
Multi-Scale Oriented Patches (MOPS)

Multi-Image Matching using Multi-Scale Oriented Patches. M. Brown, R. Szeliski and S. Winder. International Conference on Computer Vision and Pattern Recognition (CVPR2005). pages 510-517









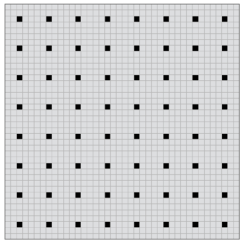
Multi-Scale Oriented Patches (MOPS)

Multi-Image Matching using Multi-Scale Oriented Patches. M. Brown, R. Szeliski and S. Winder. International Conference on Computer Vision and Pattern Recognition (CVPR2005). pages 510-517

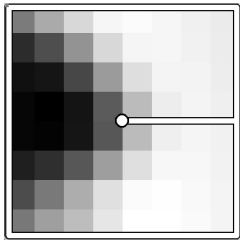
$$(x, y, s, \theta)$$

Given a feature

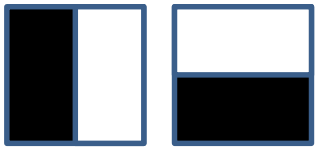
Get 40 x 40 image patch, subsample every 5th pixel
(low frequency filtering, absorbs localization errors)



Subtract the mean, divide by standard deviation
(removes bias and gain)

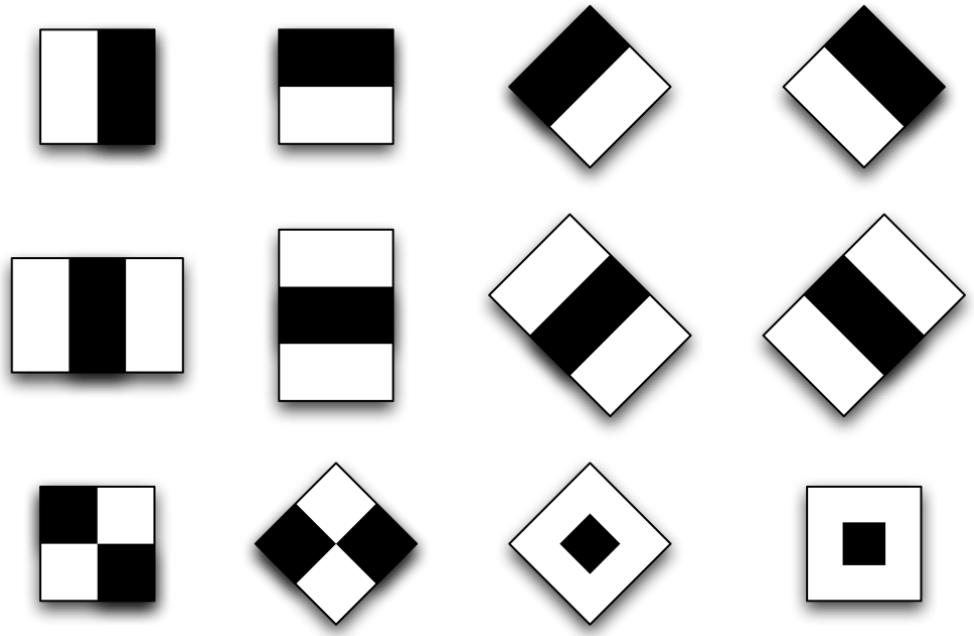


Haar Wavelet Transform
(low frequency projection)



Haar Wavelets
(actually, Haar-like features)

Use responses of a bank of filters as a descriptor



We will see later in class how to compute Haar wavelet responses **efficiently** (in constant time) with integral images

GIST descriptor

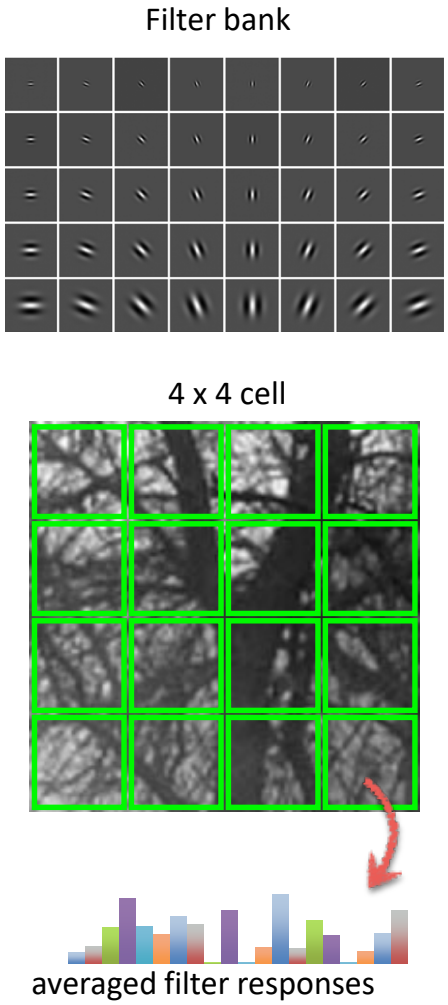
GIST

1. Compute filter responses
   (filter bank of Gabor filters)

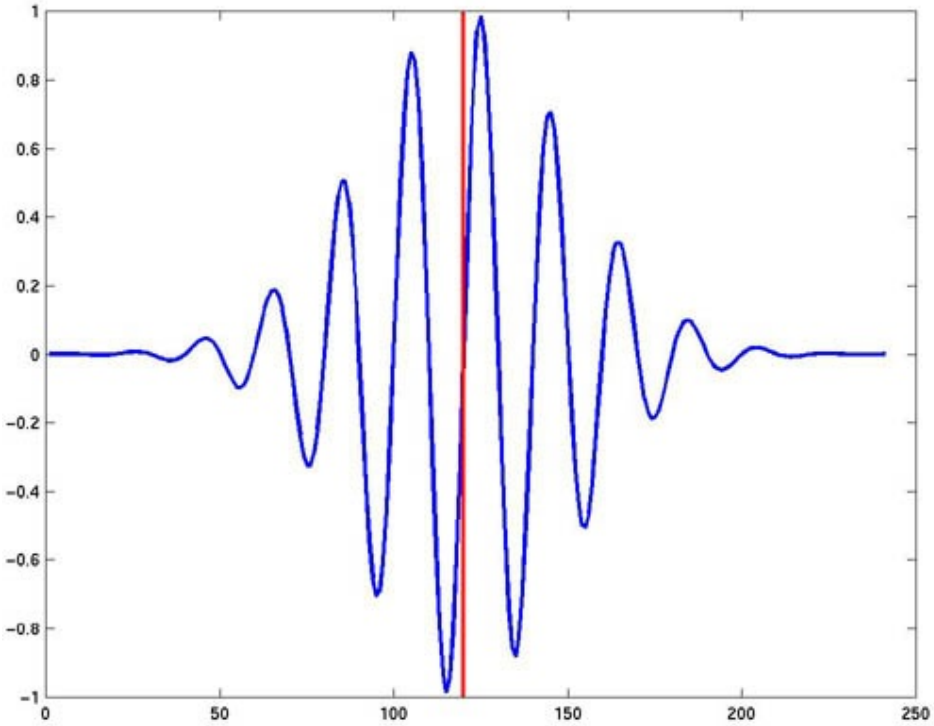
2. Divide image patch into 4 x 4 cells

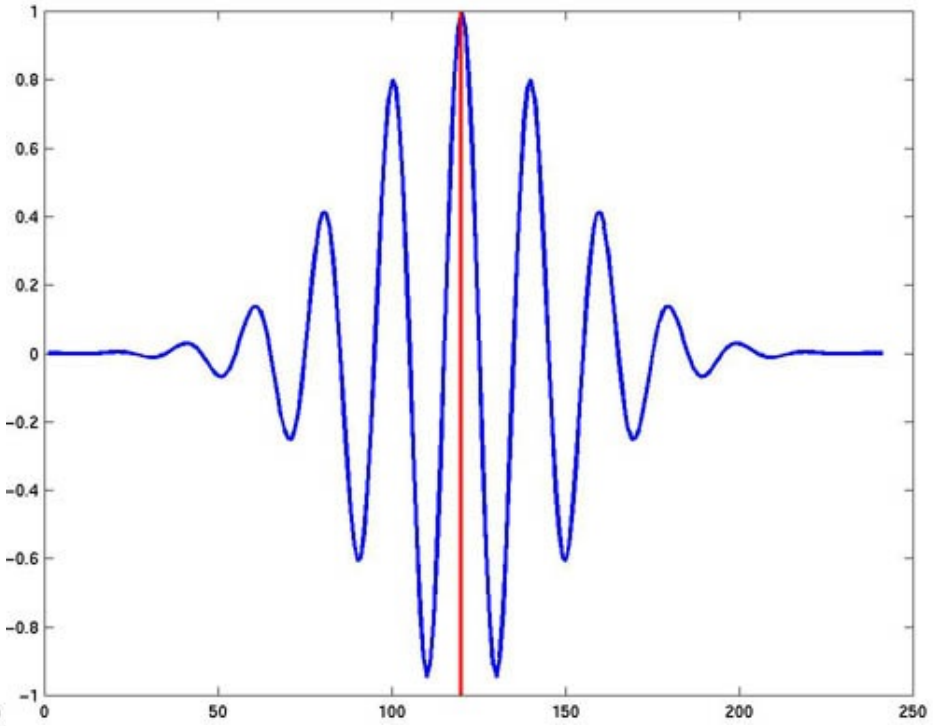
3. Compute filter response averages for each cell

4. Size of descriptor is 4 x 4 x N, where N is the size of the filter bank



Gabor Filters
(1D examples)



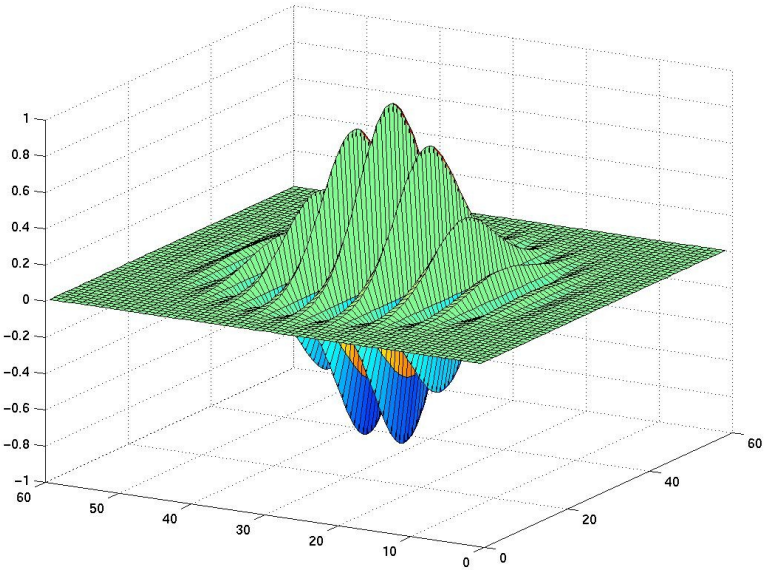


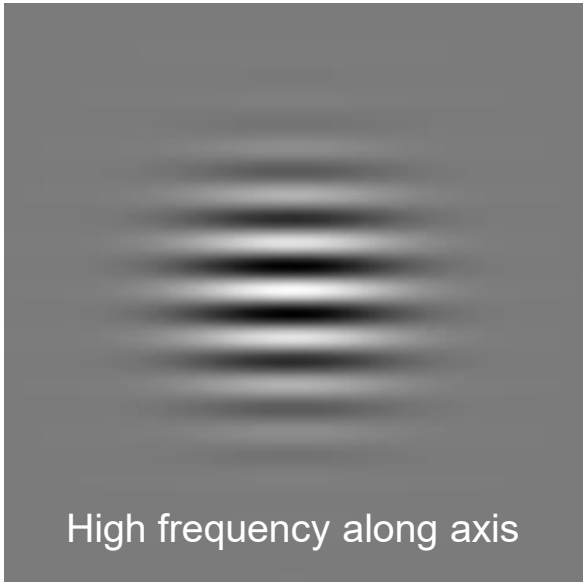
$$e^{-\frac{x^2}{2\sigma^2}} \sin(2\pi\omega x)$$

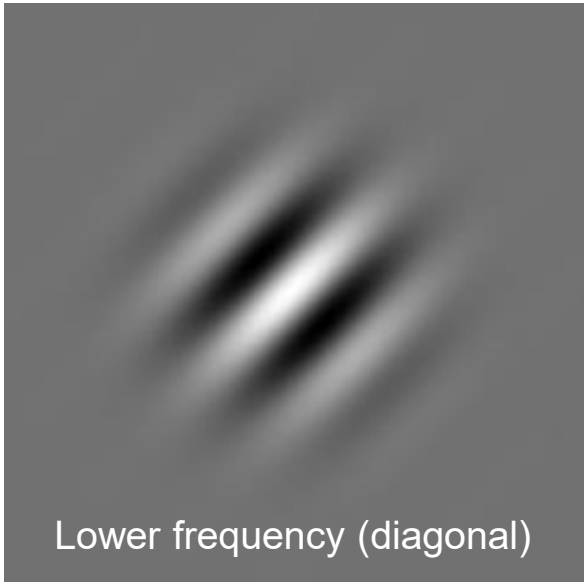
$$e^{-\frac{x^2}{2\sigma^2}} \cos(2\pi\omega x)$$

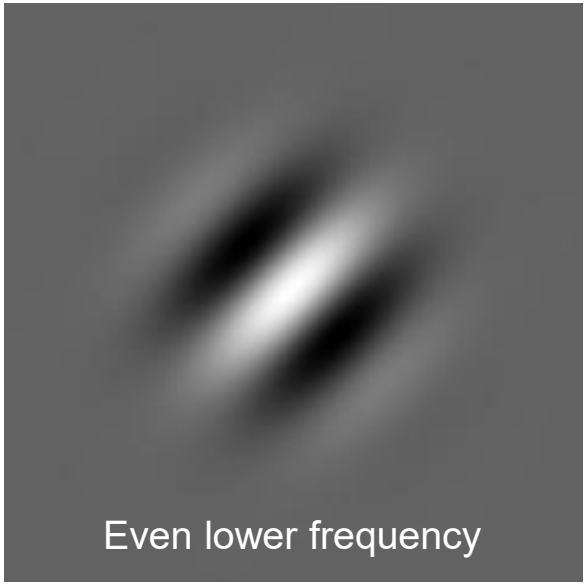
2D Gabor Filters

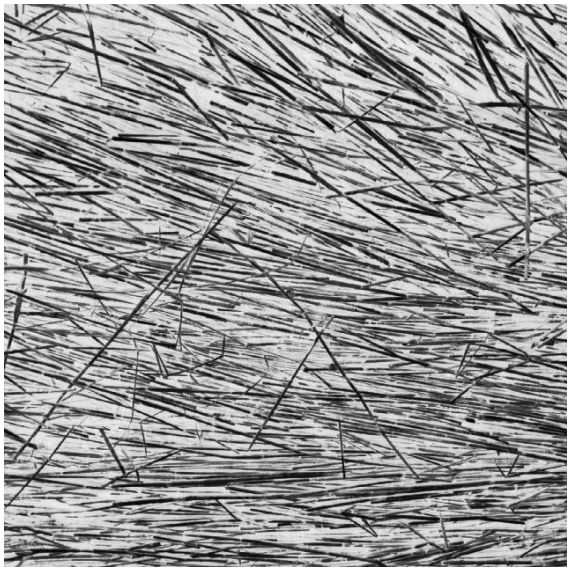
$$e^{-\frac{x^2+y^2}{2\sigma^2}} \cos\left(2\pi\left(k_x x + k_y y\right)\right)$$

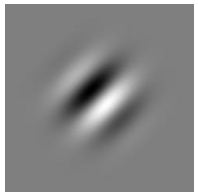


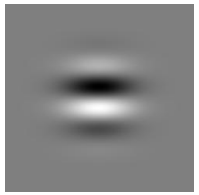


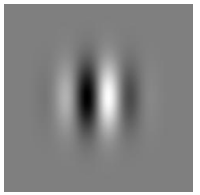


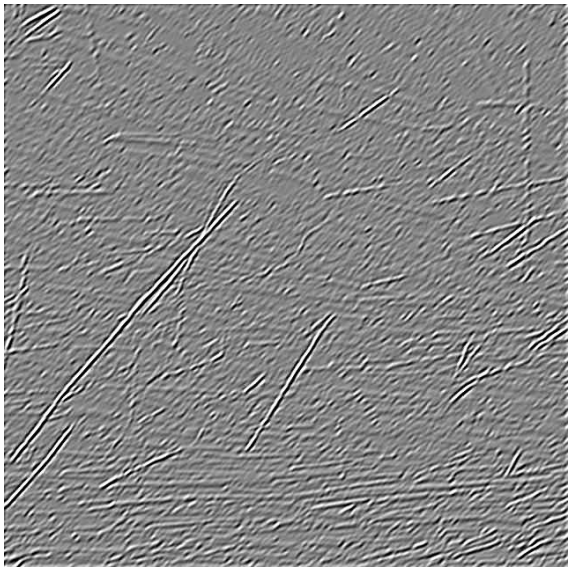


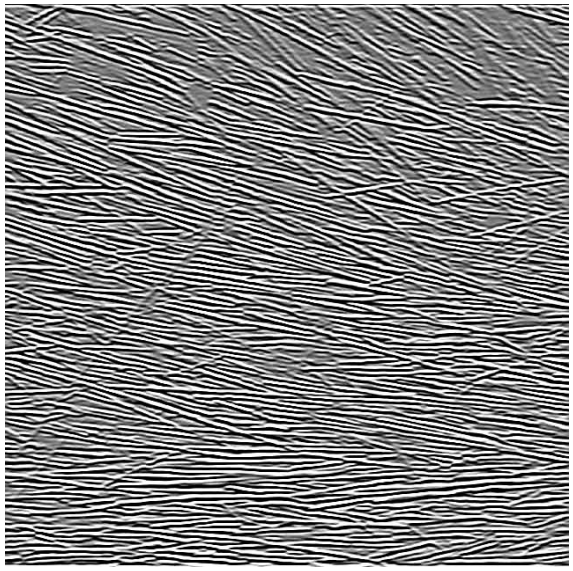


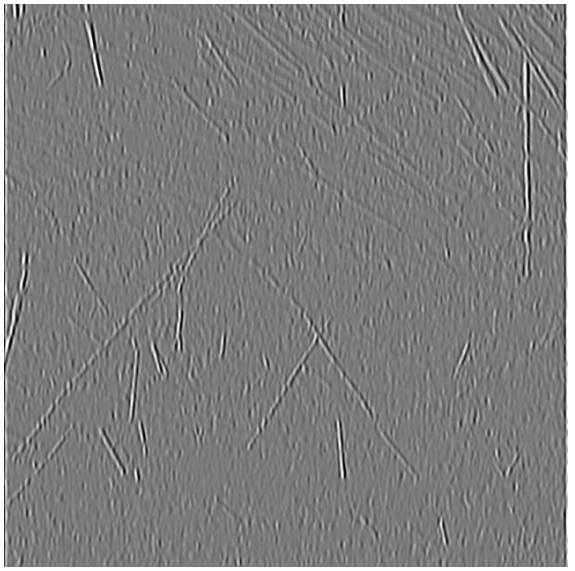


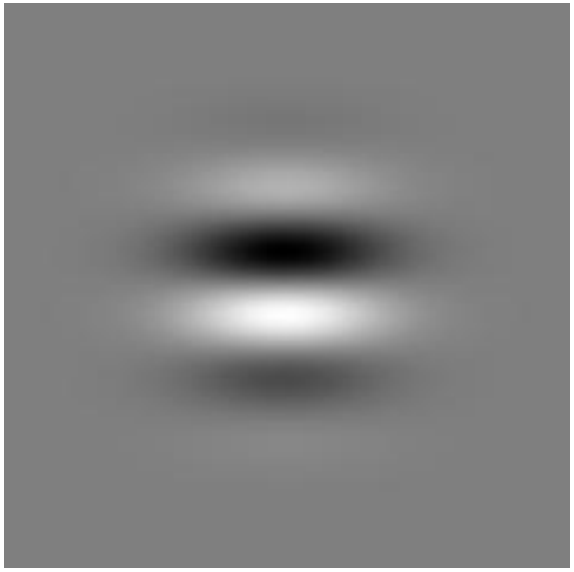




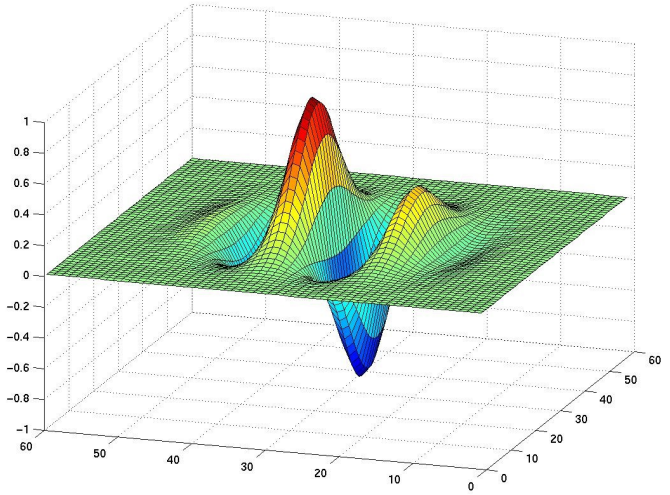




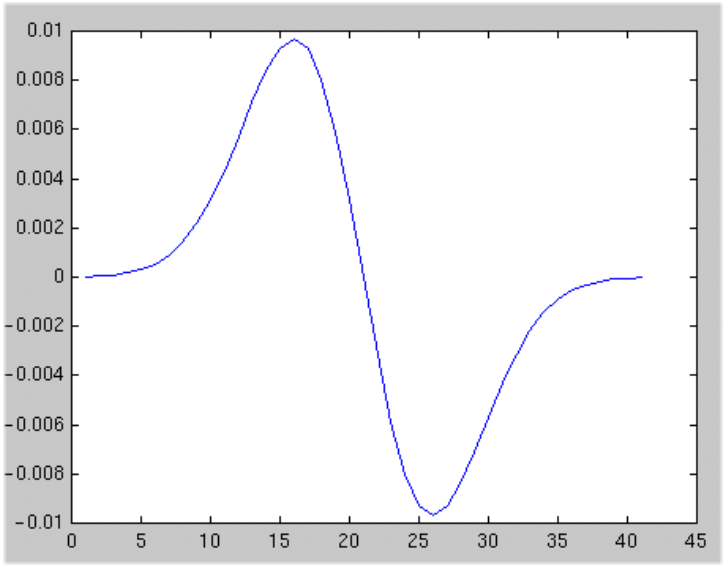




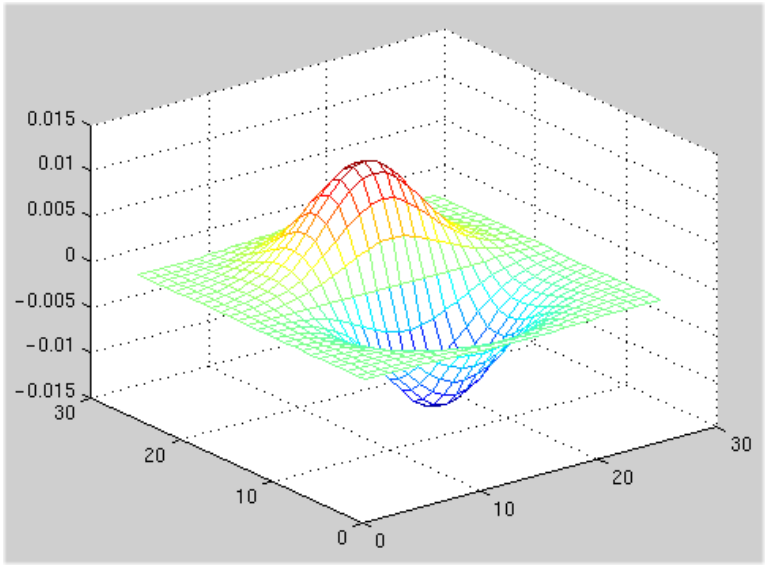
Odd Gabor filter

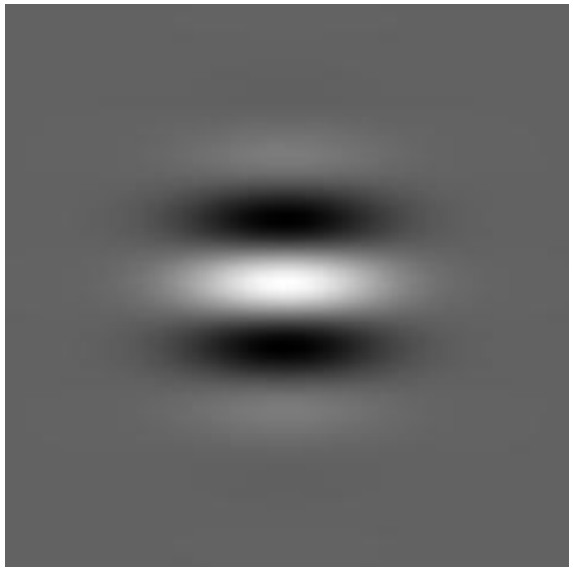


… looks a lot like…

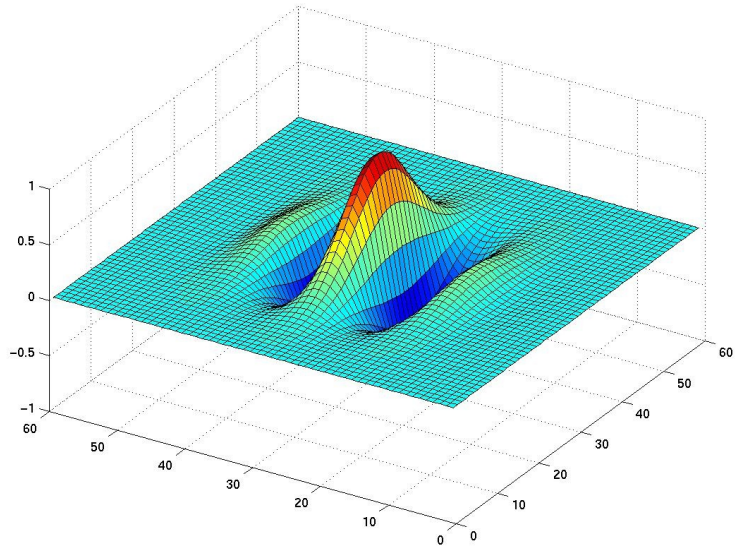


Gaussian Derivative

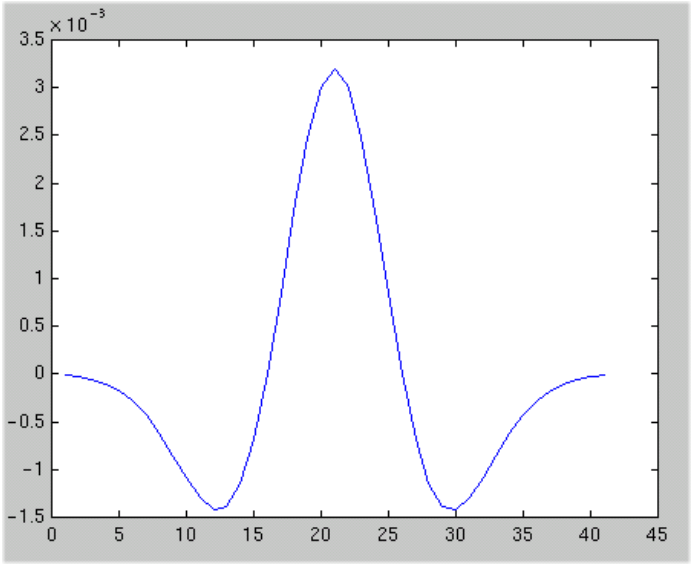




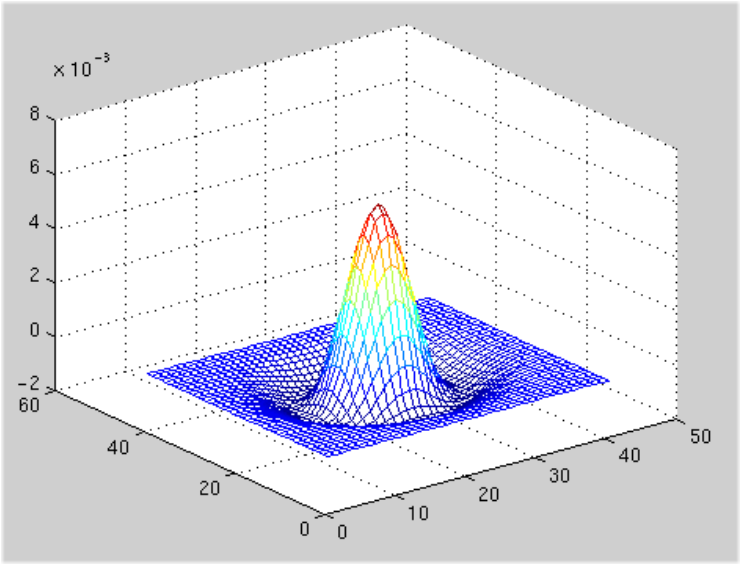
Even Gabor filter



… looks a lot like…



Laplacian



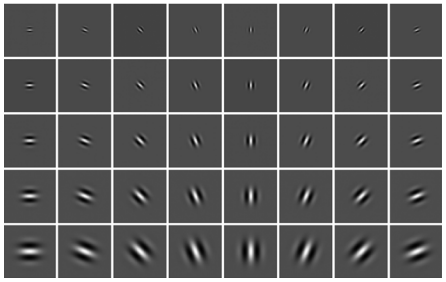
GIST

1. Compute filter responses (filter bank of Gabor filters)
2. Divide image patch into 4 x 4 cells
3. Compute filter response averages for each cell
4. Size of descriptor is 4 x 4 x N, where N is the size of the filter bank

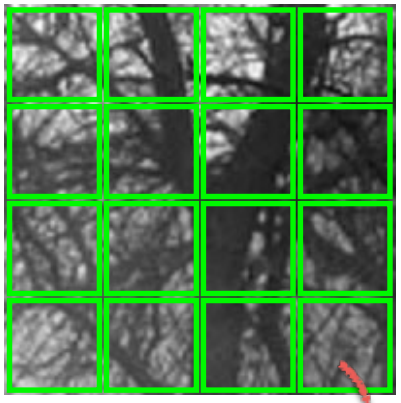
What is the GIST descriptor encoding?

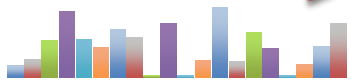
Rough spatial distribution of image gradients

Filter bank



4 x 4 cell





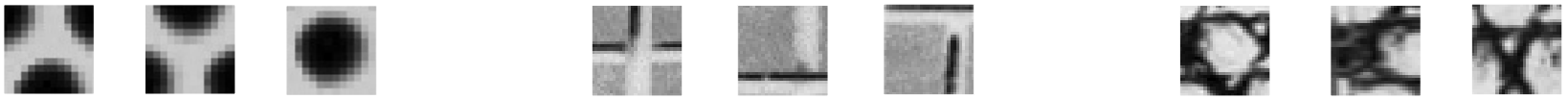
averaged filter responses

Histogram of Textons descriptor

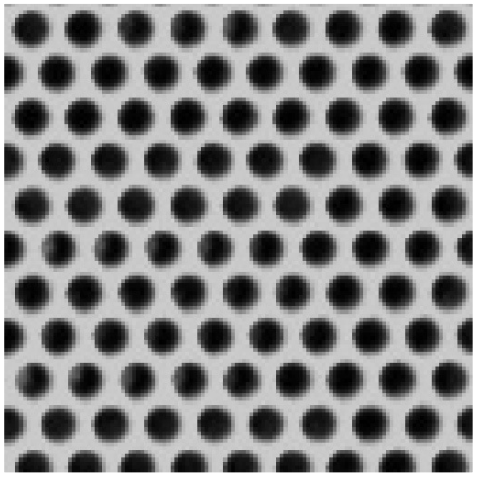
Textons

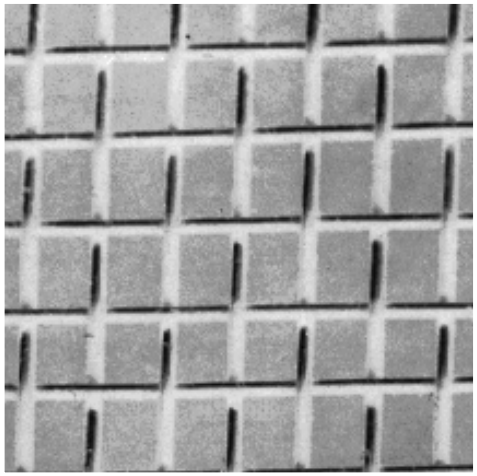
Julesz. Textons, the elements of texture perception, and their interactions. Nature 1981

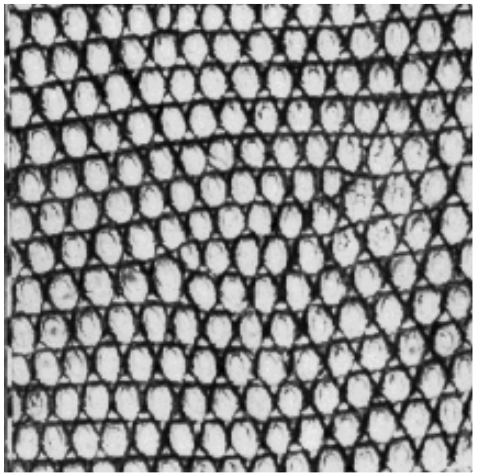
Texture is characterized by the repetition of basic elements or ***textons***



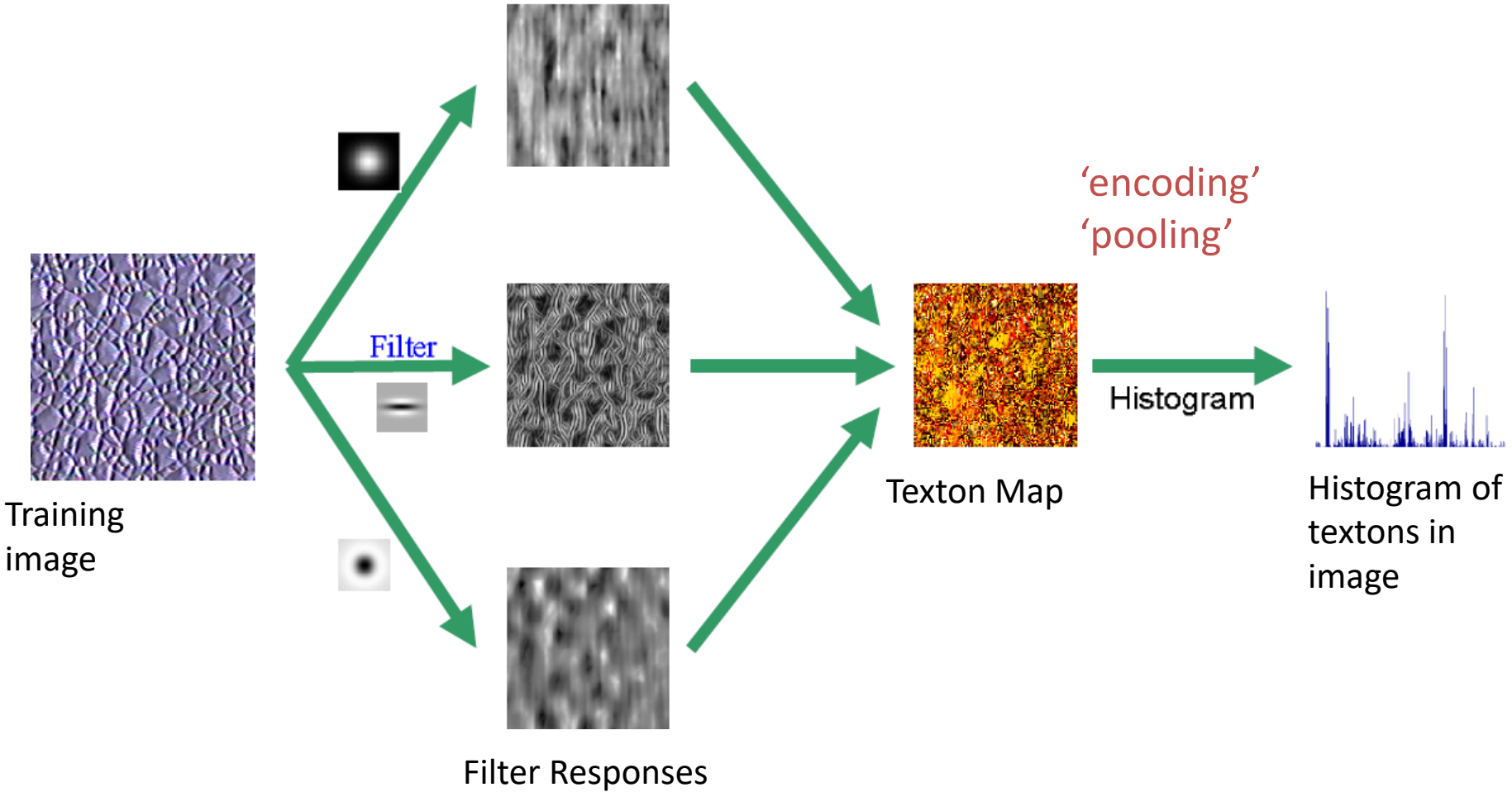
For stochastic textures, it is the identity of the ***textons***, not their spatial arrangement, that matters





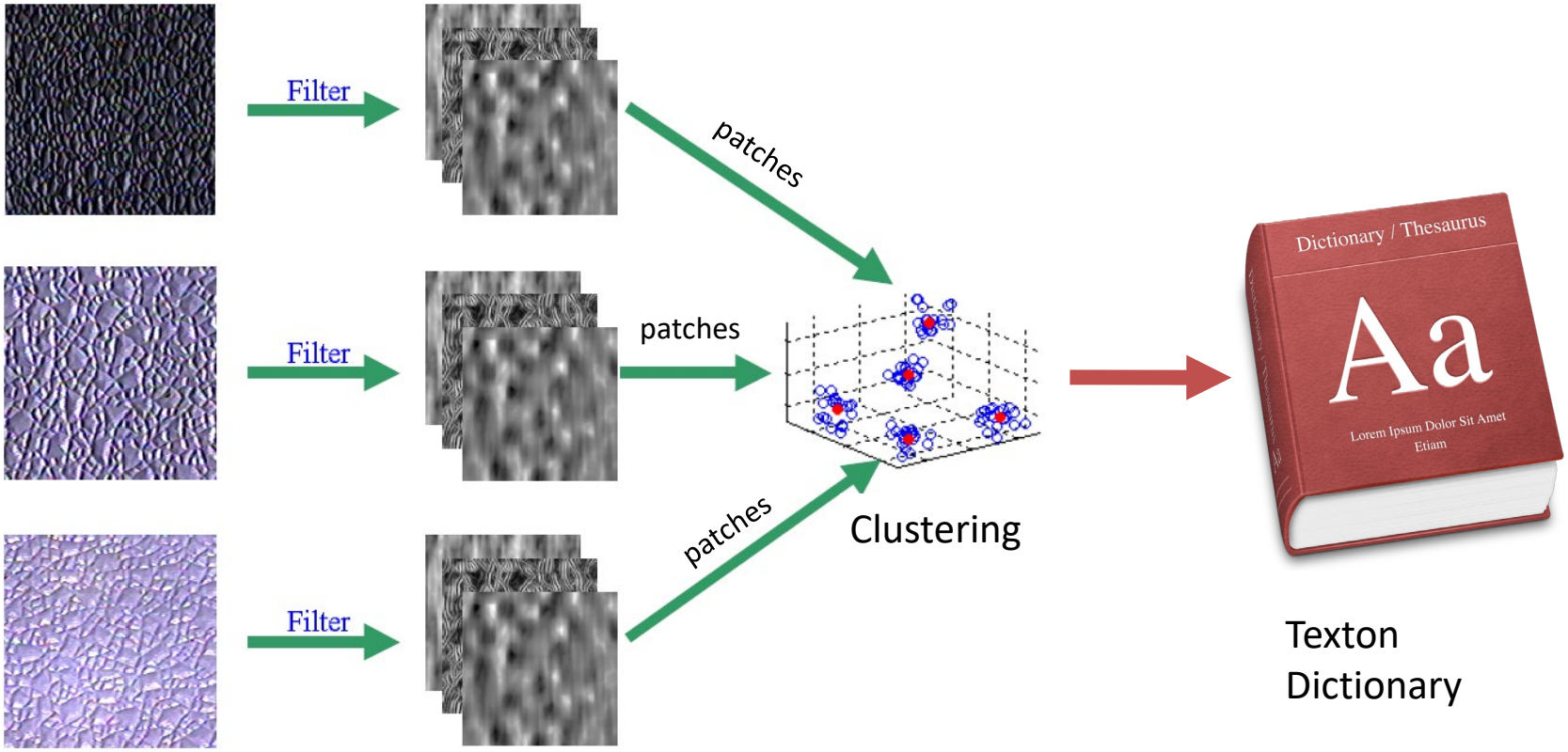


Histogram of Textons descriptor



Malik, Belongie, Shi, Leung. Textons, Contours and Regions: Cue Integration in Image Segmentation. ICCV 1999.

Learning Textons from data

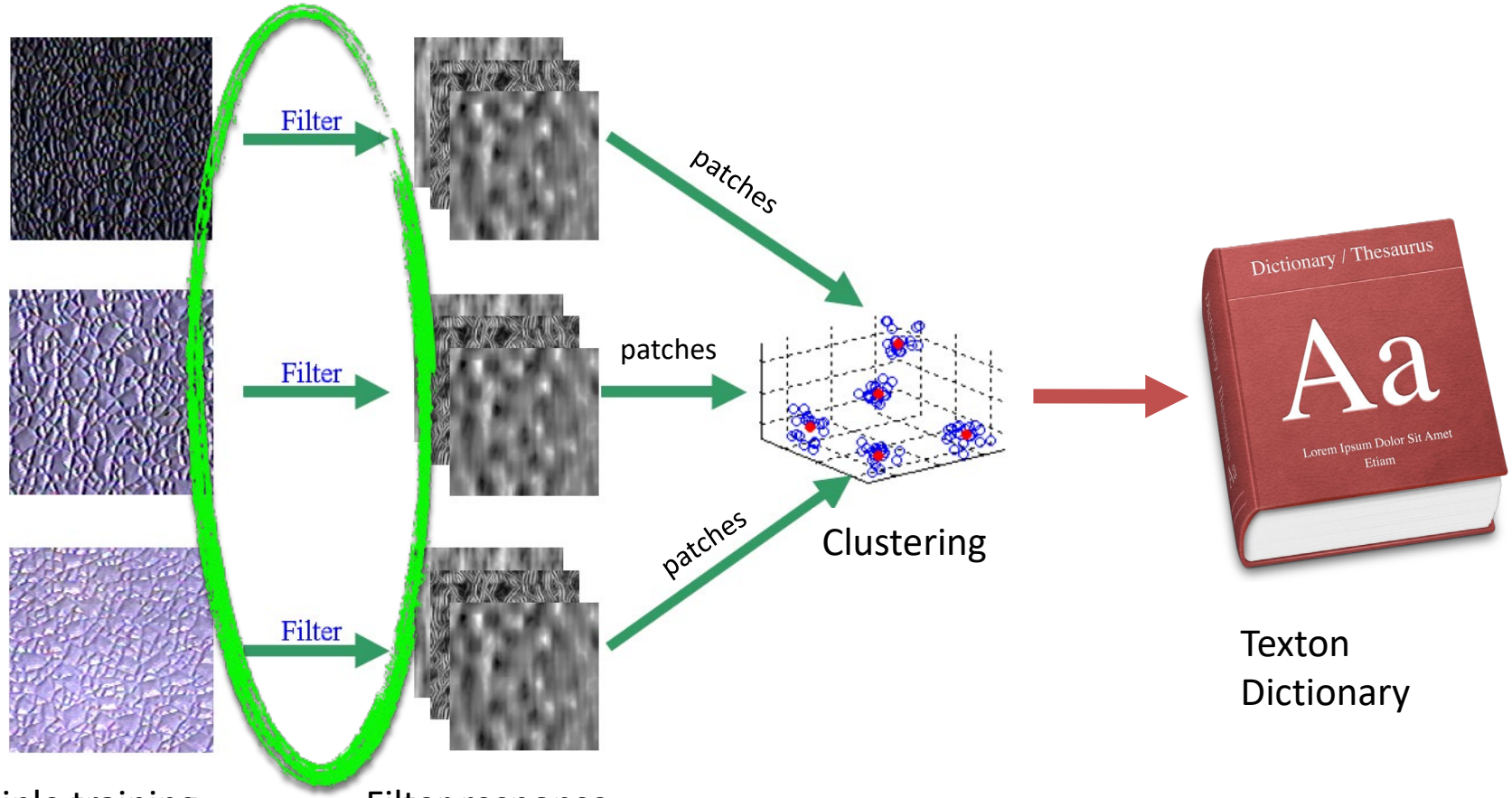


Multiple training images of the **same texture**

Filter response over a bank of filters

Texton Dictionary

Learning Textons from data



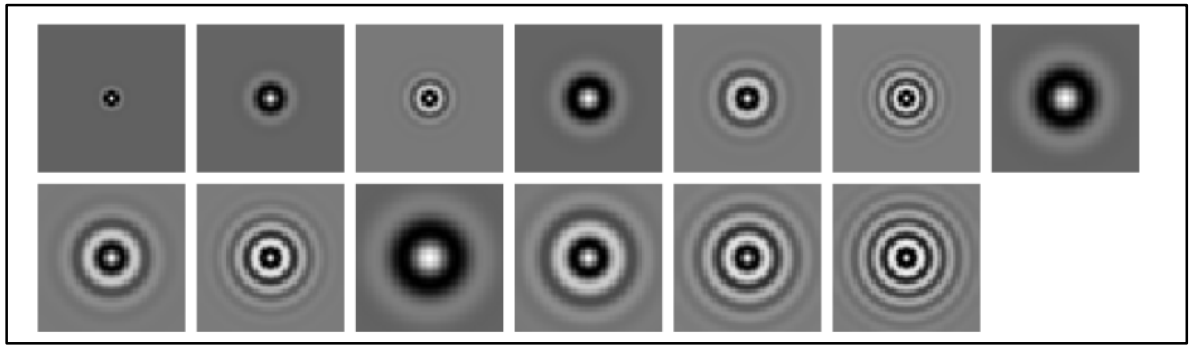
Multiple training images of the **same texture**

Filter response over a bank of filters

Texton Dictionary

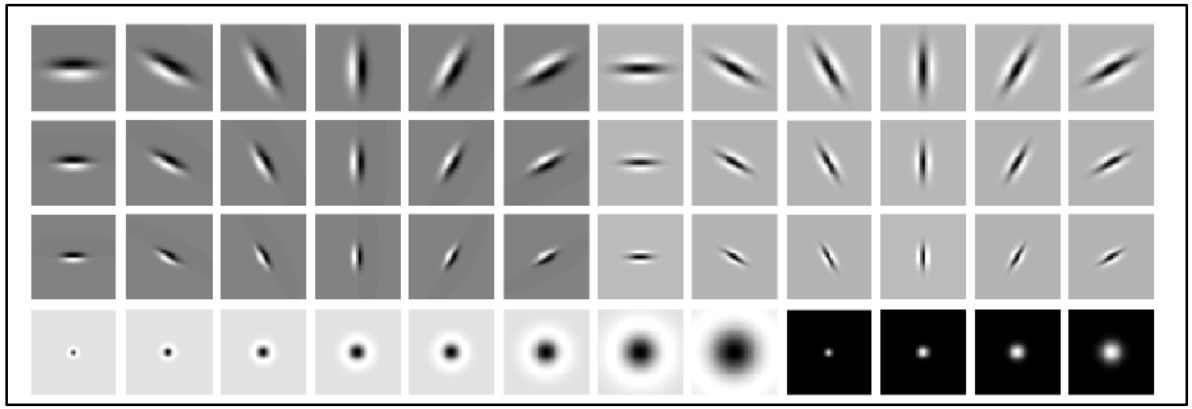
Example of Filter Banks

Isotropic Gabor

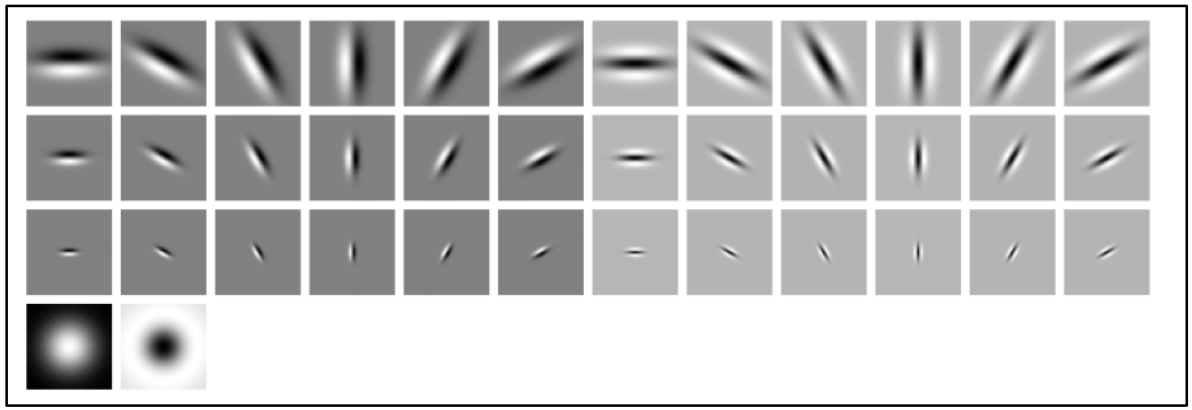


'S'

Gaussian derivatives at different scales and orientations

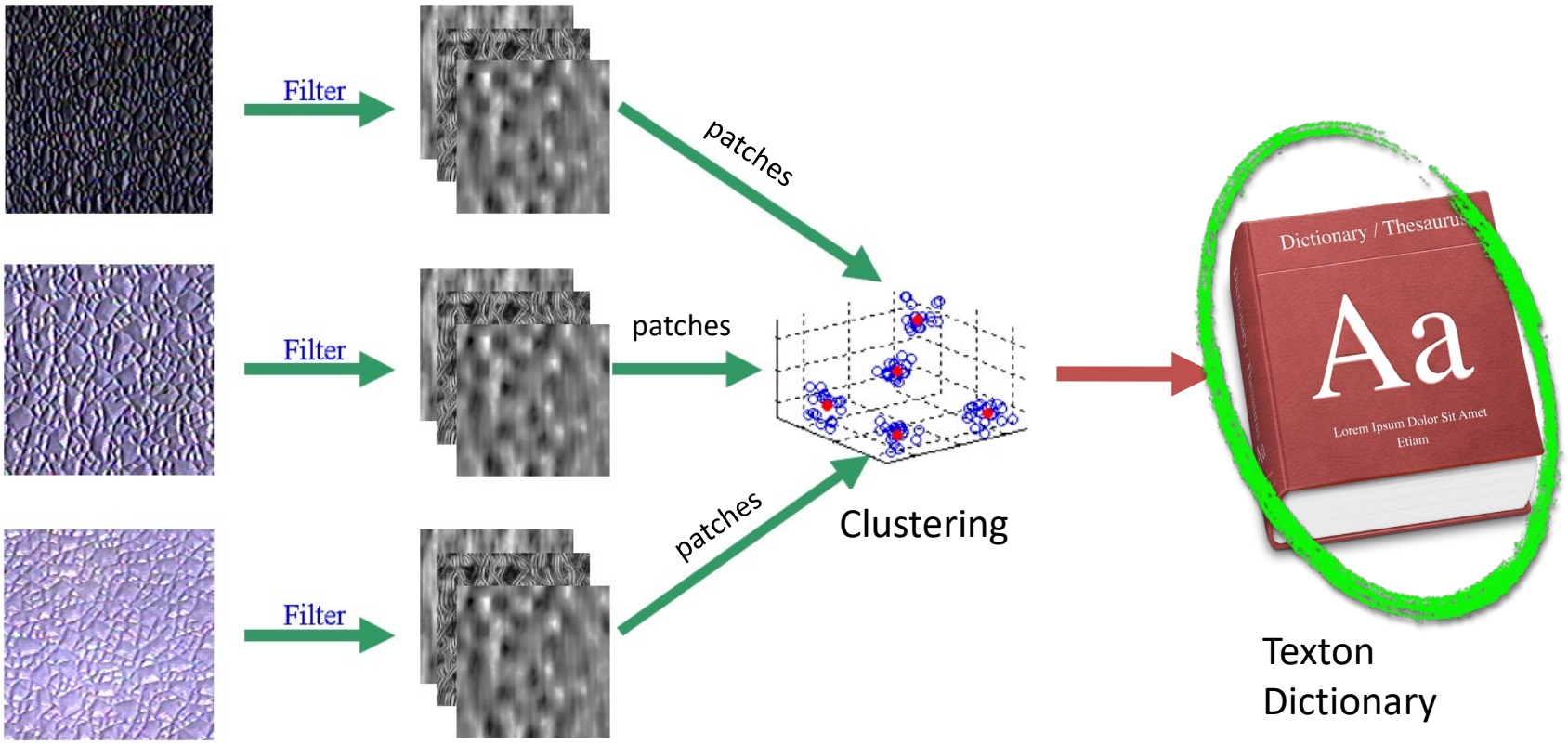


'LM'



'MR8'

Learning Textons from data

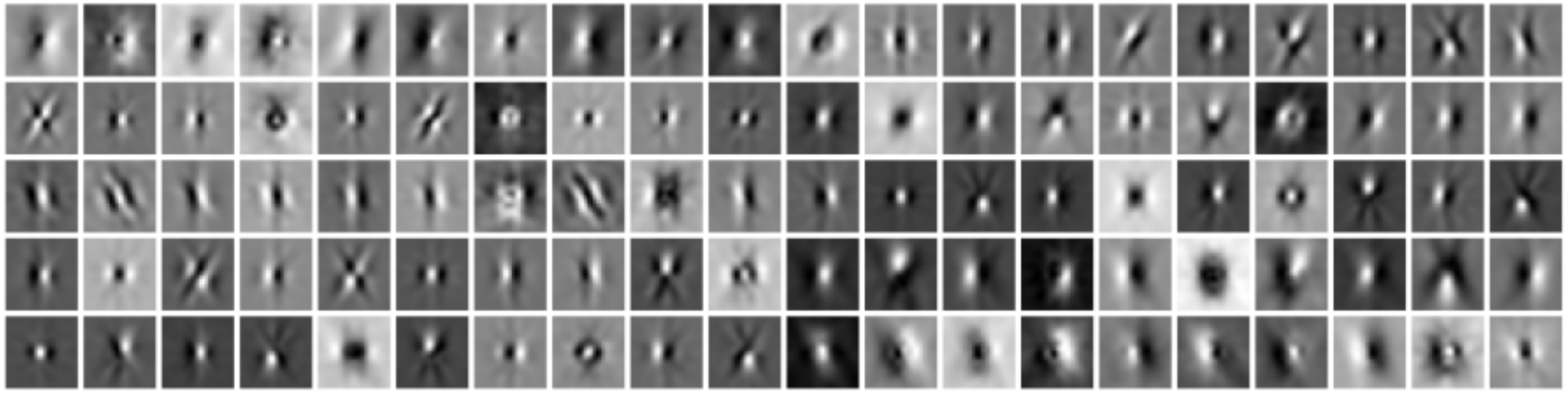


Multiple training images of the same texture

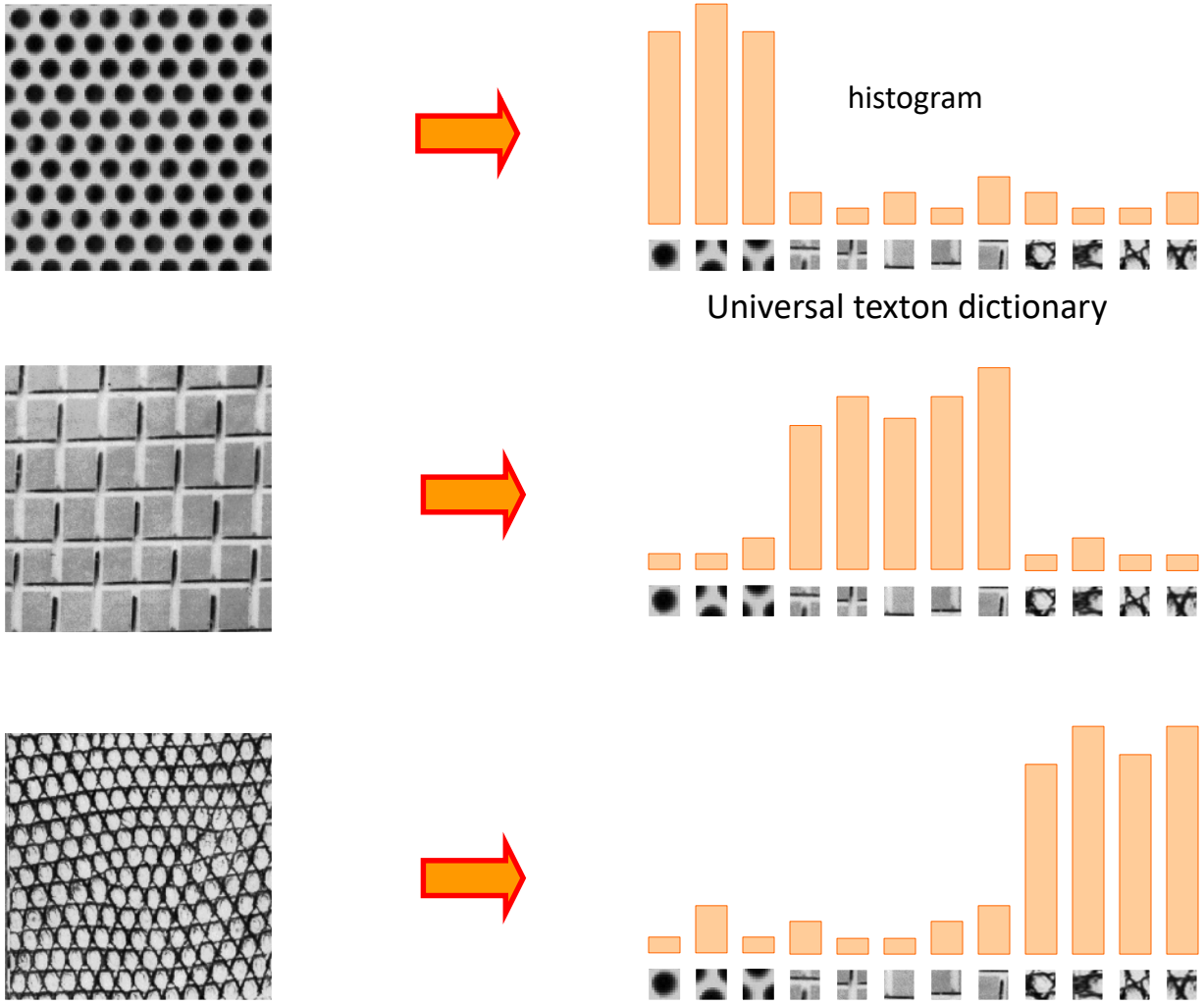
Filter response over a bank of filters

We will learn more about clustering later in class (Bag of Words lecture).

Texton Dictionary

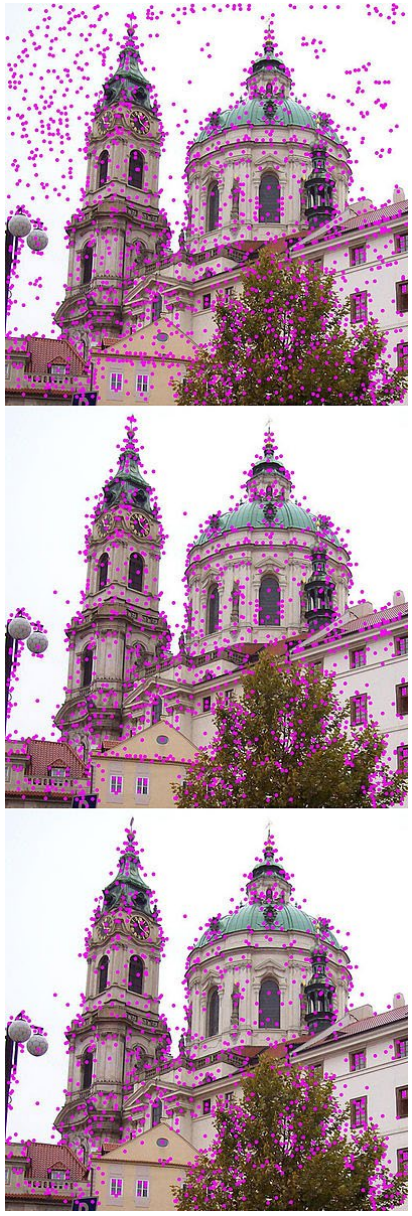


Malik, Belongie, Shi, Leung. Textons, Contours and Regions: Cue Integration in Image Segmentation. ICCV 1999.



Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

SIFT



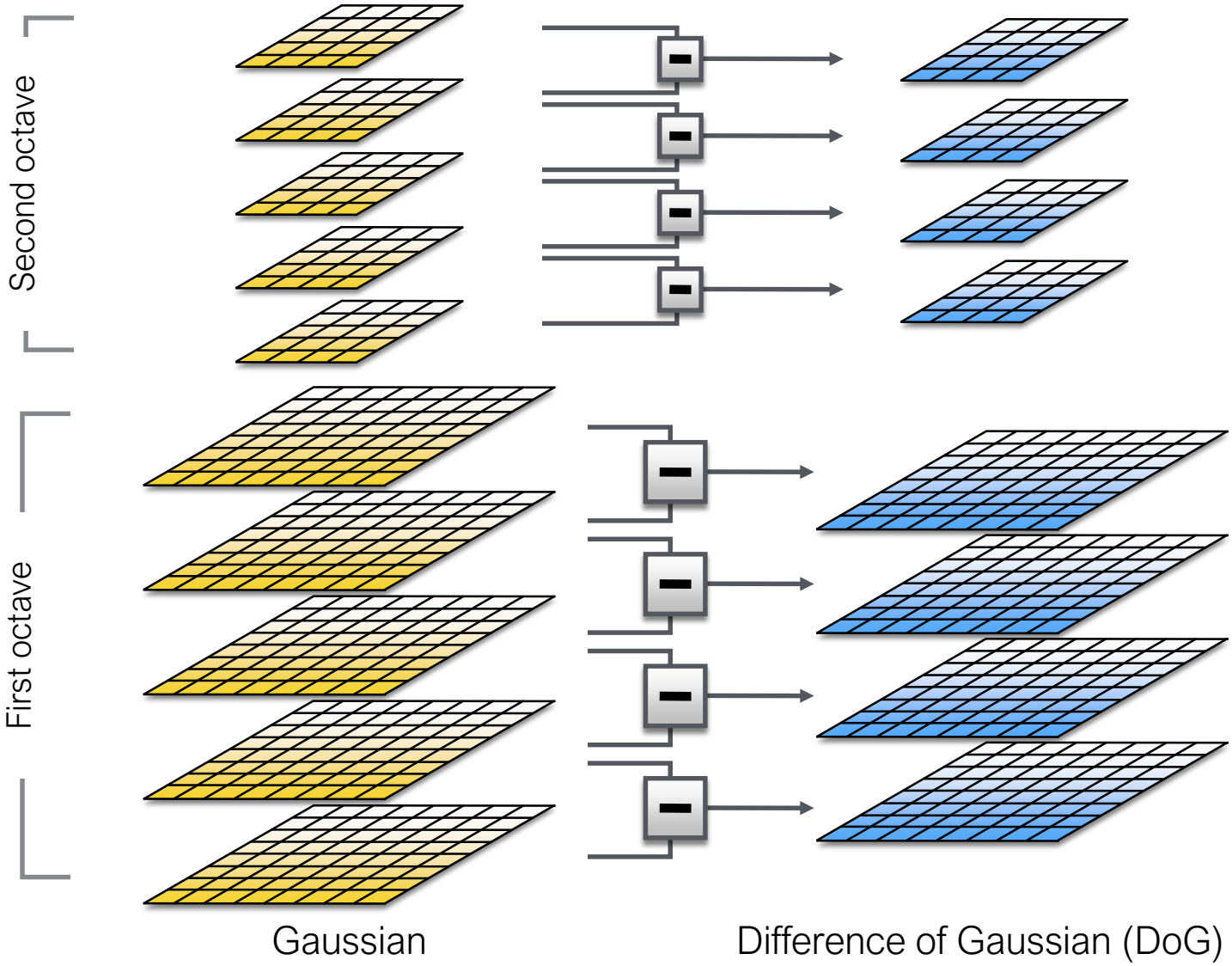
SIFT

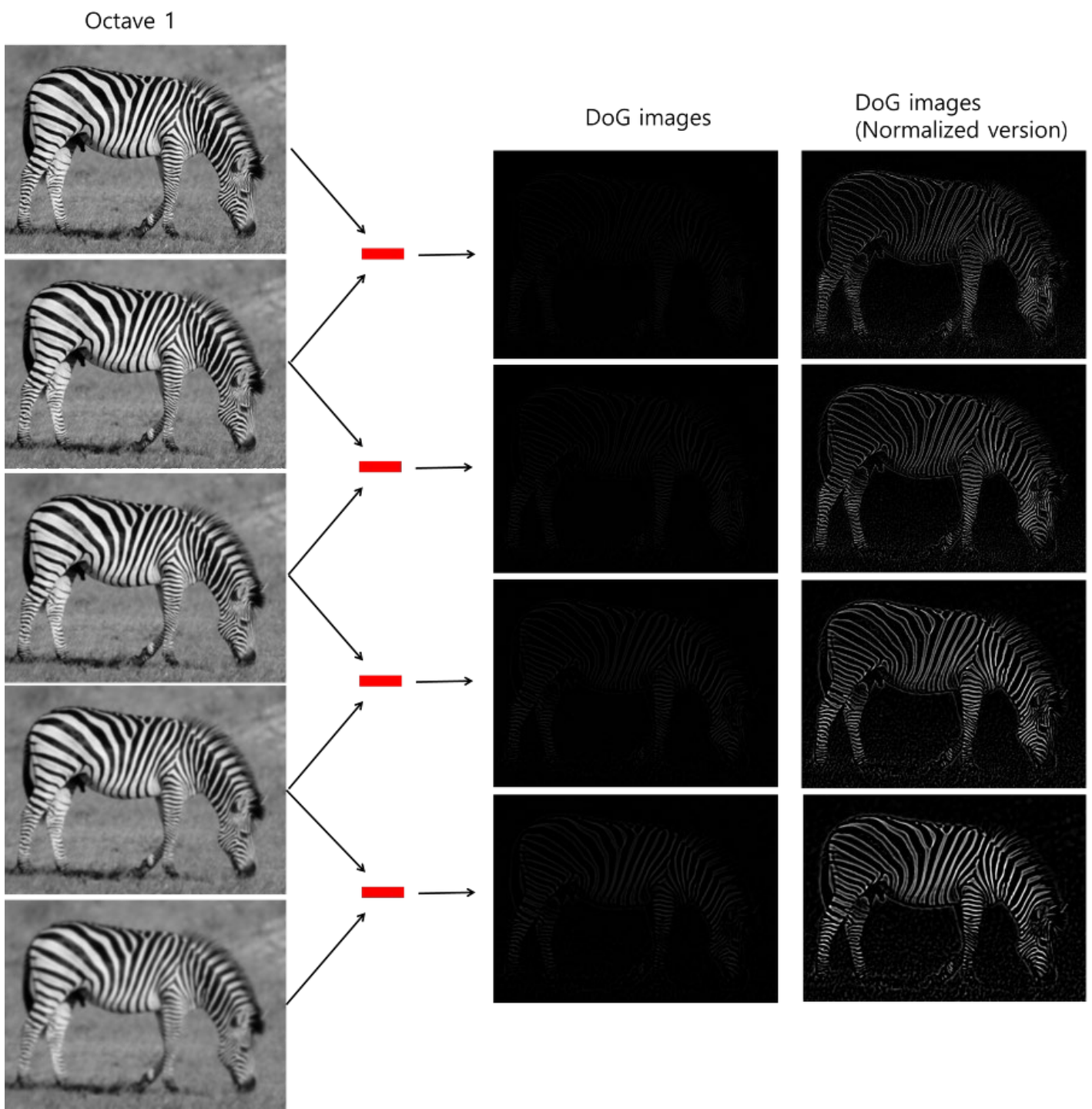
(Scale Invariant Feature Transform)

SIFT describes both a **detector** and **descriptor**

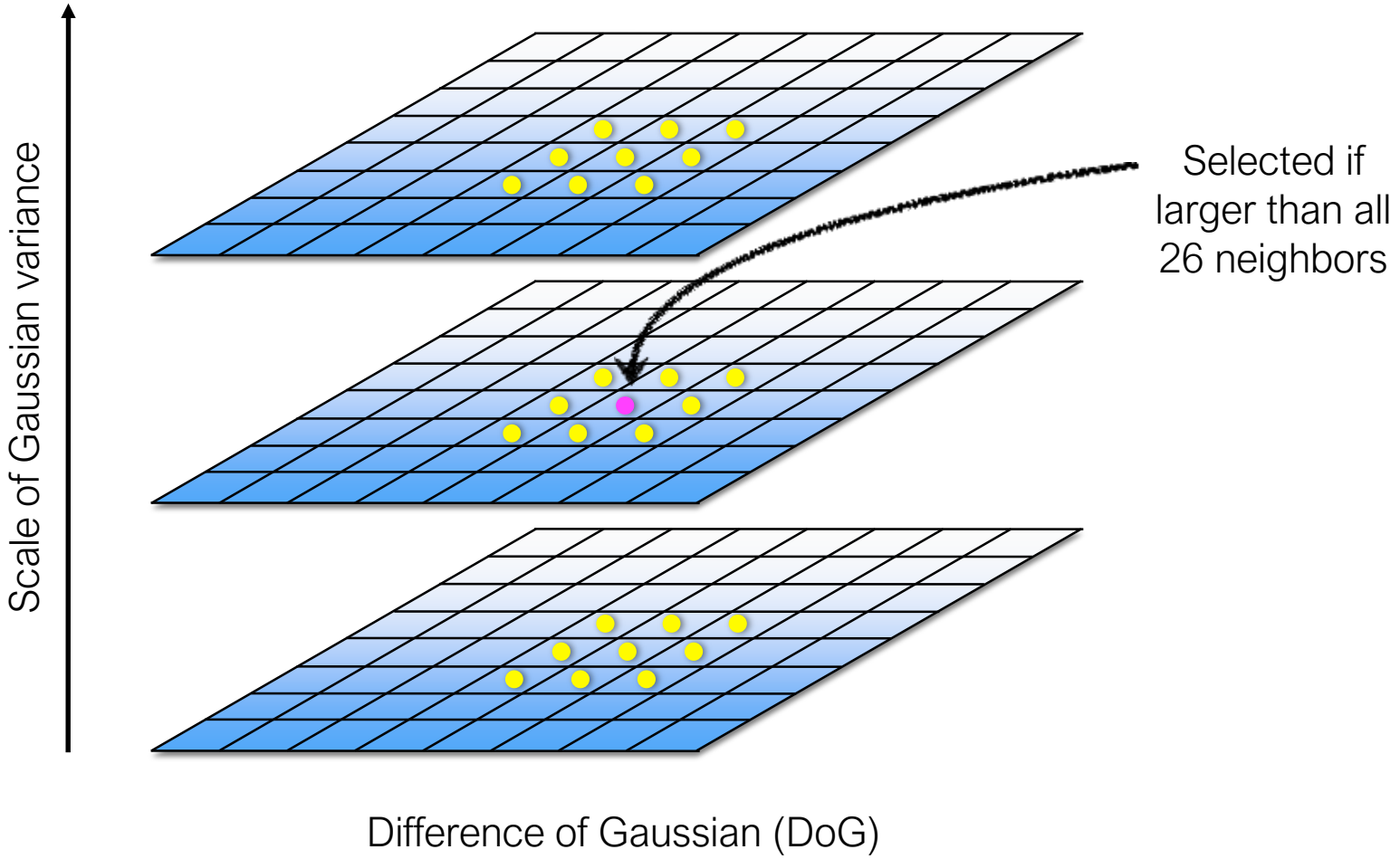
1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

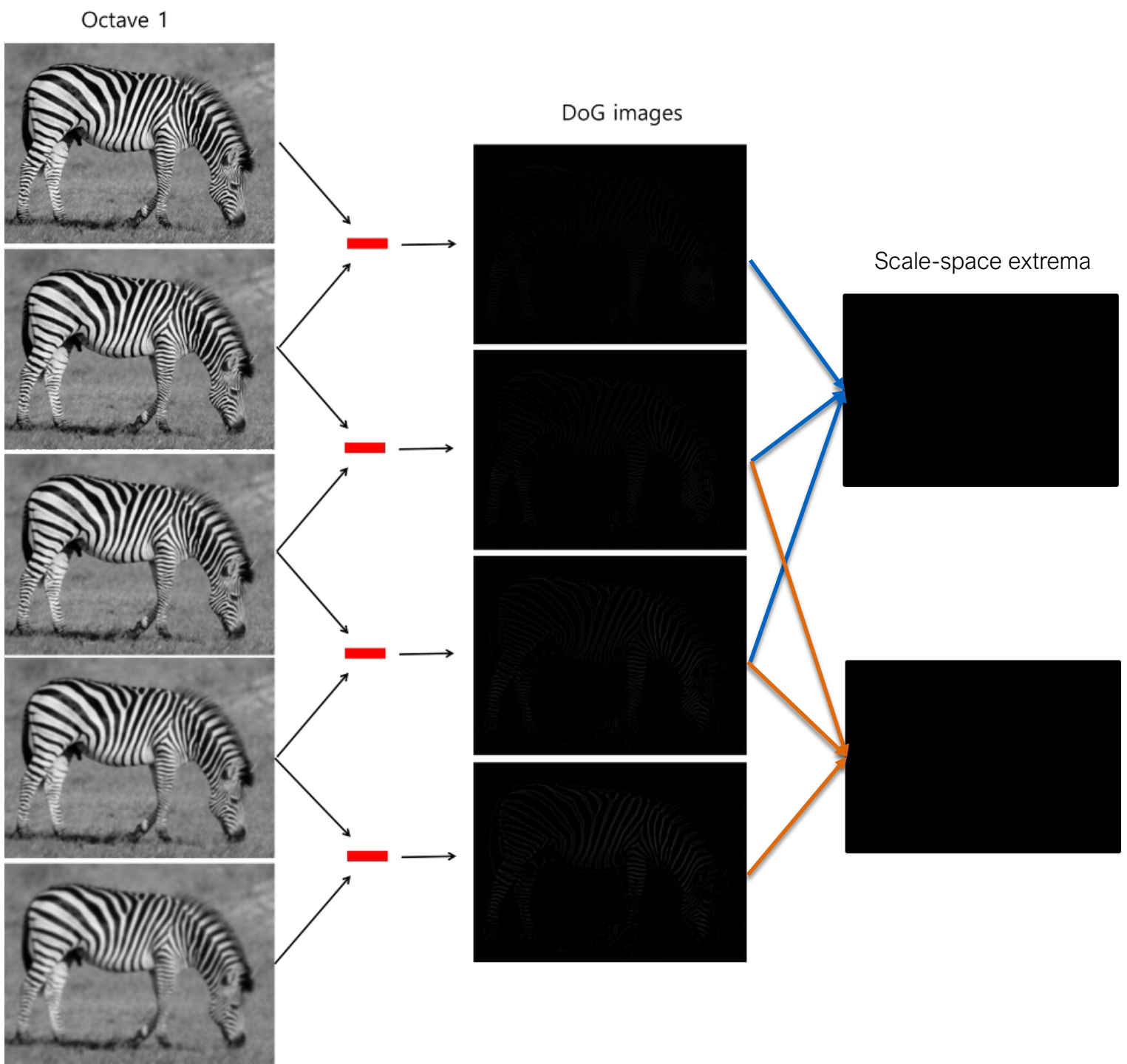
1. Multi-scale extrema detection



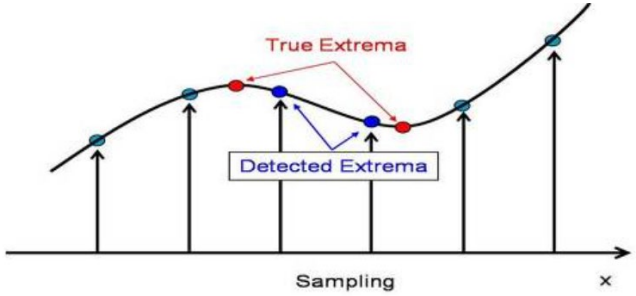

Octave 1
DoG images
DoG images
(Normalized version)

Scale-space extrema




Octave 1
DoG images
Scale-space extrema

2. Keypoint localization



2nd order Taylor series approximation of DoG scale-space

$$f(\mathbf{x}) = f + \frac{\partial f}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 f}{\partial \mathbf{x}^2}\mathbf{x}$$

$$\mathbf{x} = \{x, y, \sigma\}$$

Take the derivative and solve for extrema

$$\mathbf{x}_m = -\frac{\partial^2 f}{\partial \mathbf{x}^2}^{-1} \frac{\partial f}{\partial \mathbf{x}}$$

Additional tests to retain only strong features

3. Orientation assignment

For a keypoint, **L** is the **Gaussian-smoothed** image with the closest scale,

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

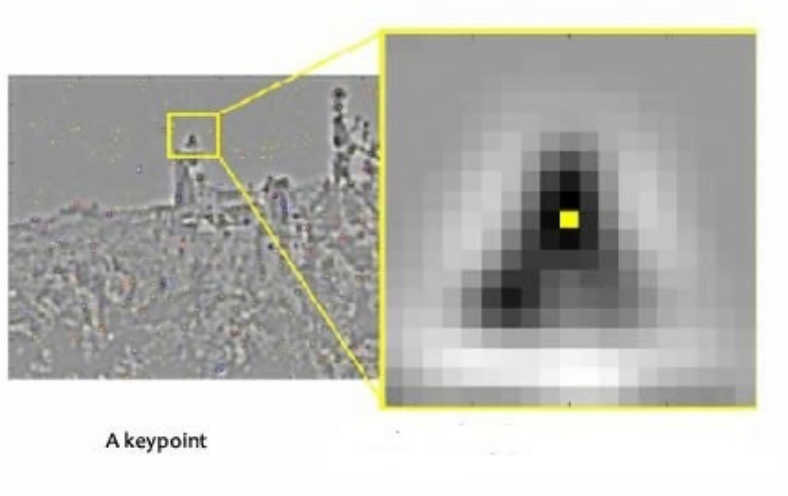
x-derivative y-derivative

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Detection process returns

$$\{x, y, \sigma, \theta\}$$

location scale orientation



3. Orientation assignment

For a keypoint, **L** is the **Gaussian-smoothed** image with the closest scale,

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

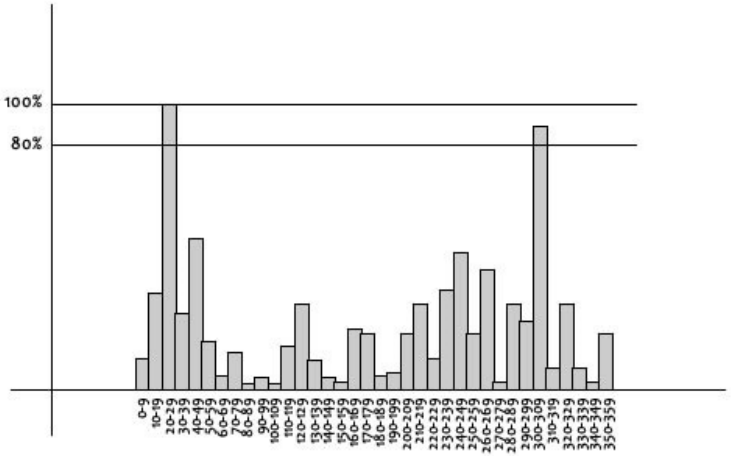
x-derivative y-derivative

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Detection process returns

$$\{x, y, \sigma, \theta\}$$

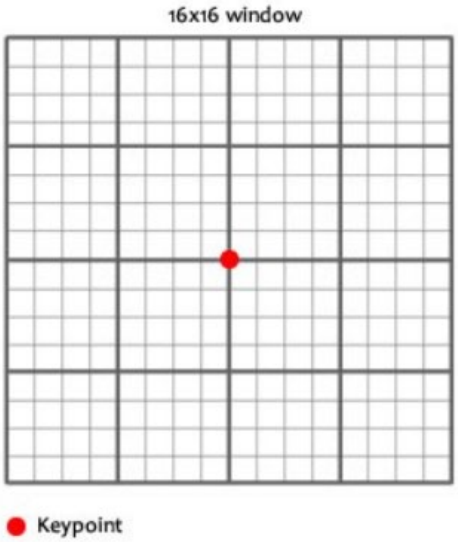
location scale orientation

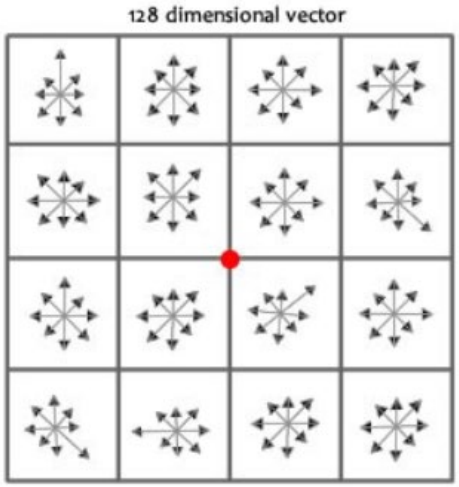


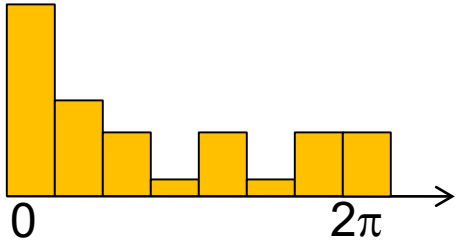
4. Keypoint descriptor

Basic idea:

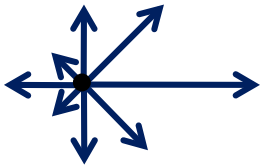
- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations







angle histogram

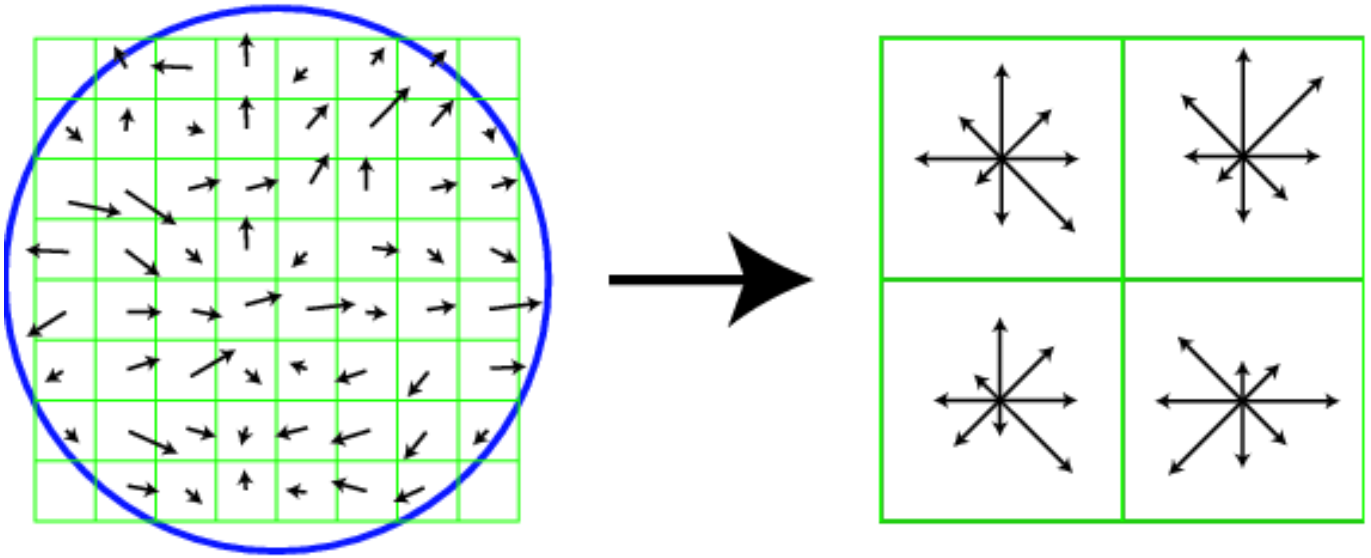


Adapted from slide by David Lowe

4. Keypoint 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

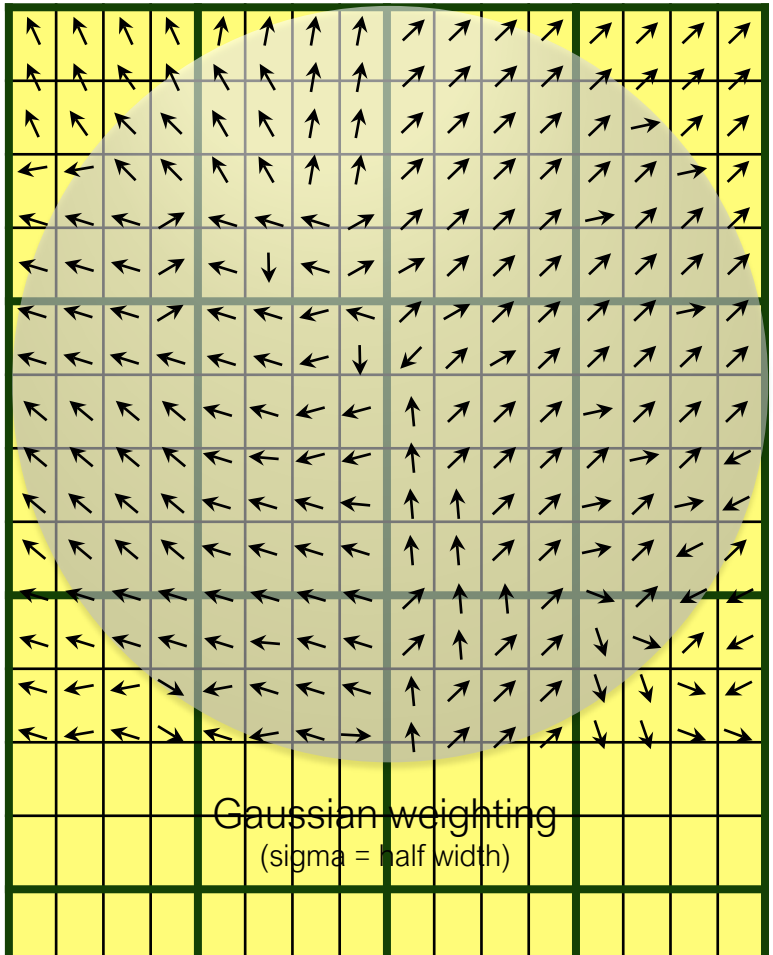


Adapted from slide by David Lowe

4. Keypoint descriptor

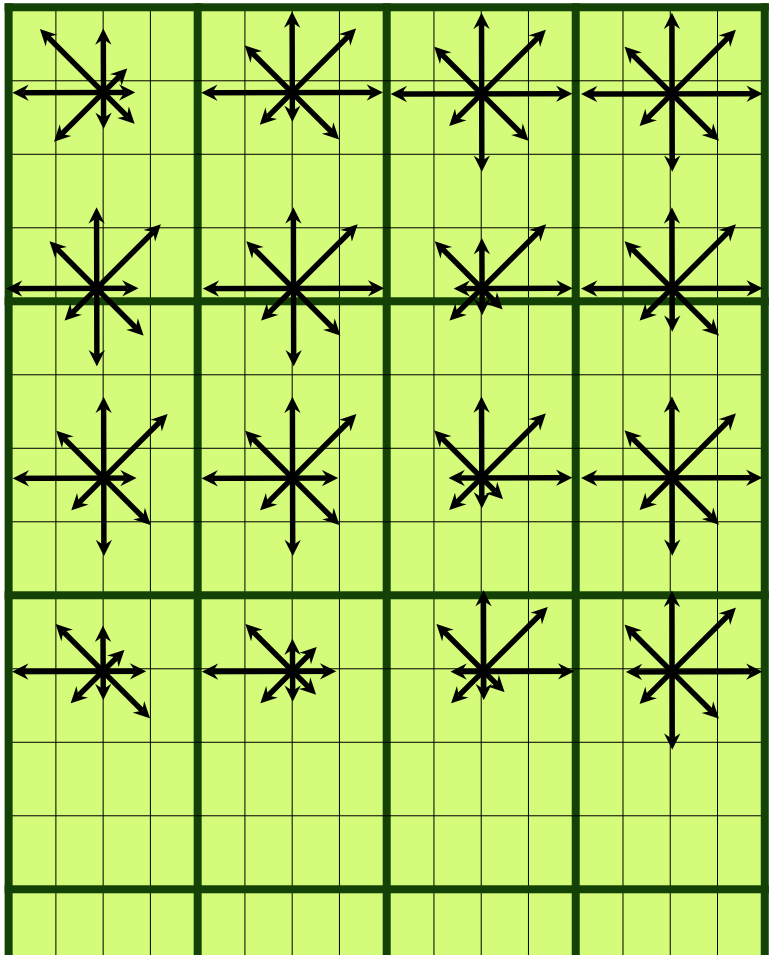
Image Gradients

(4 x 4 pixel per cell, 4 x 4 cells)



SIFT descriptor

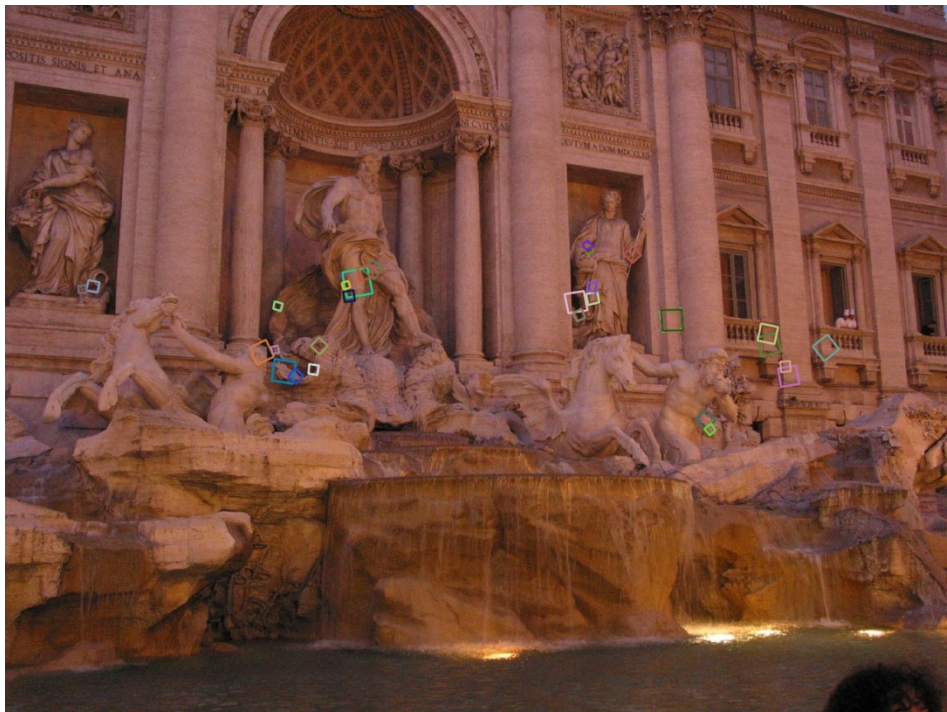
(16 cells x 8 directions = 128 dims)



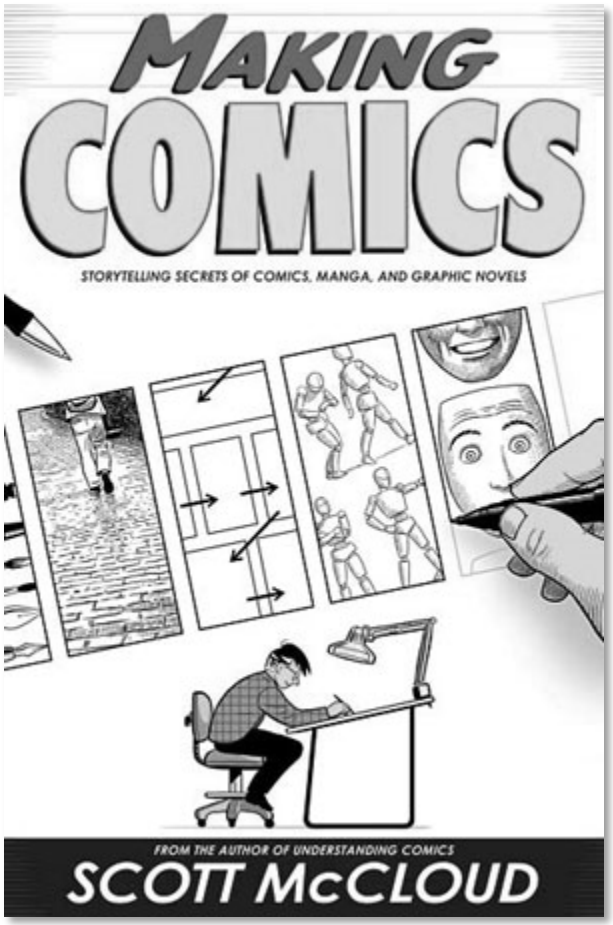
Properties of SIFT

Extraordinarily robust matching technique

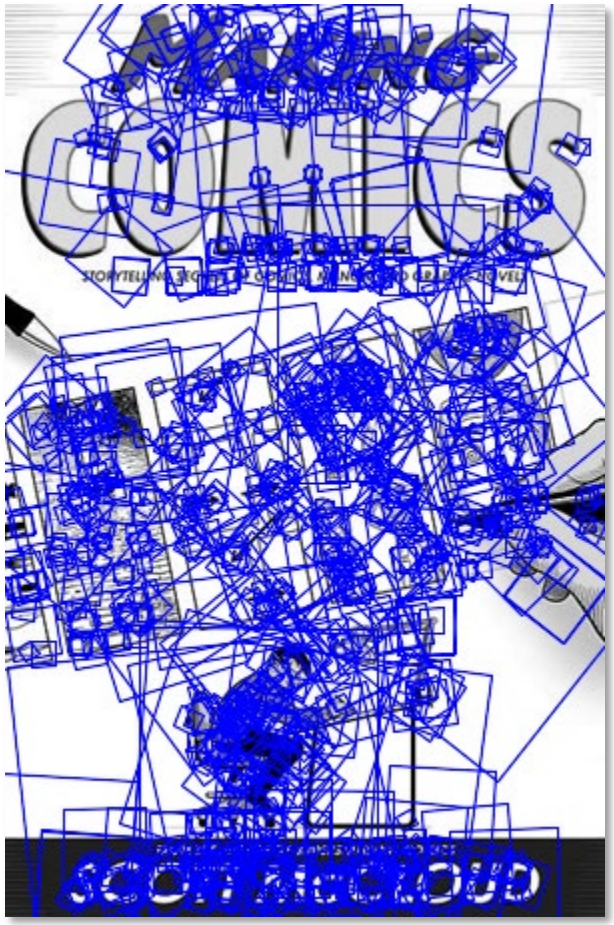
- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT



SIFT Example



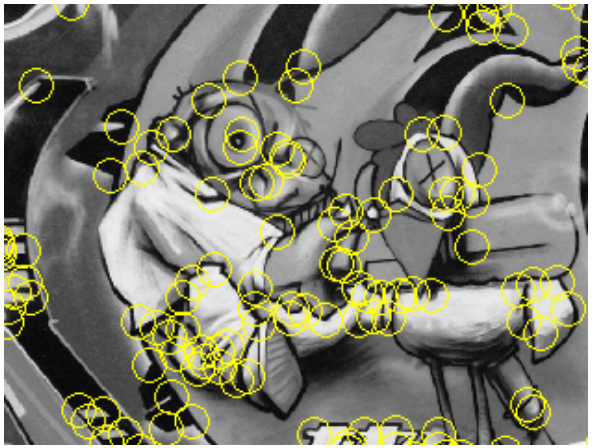
sift



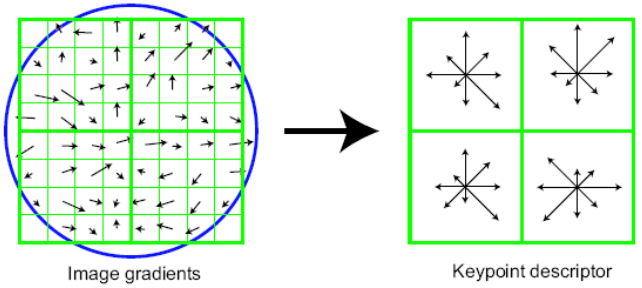
868 SIFT features

Summary

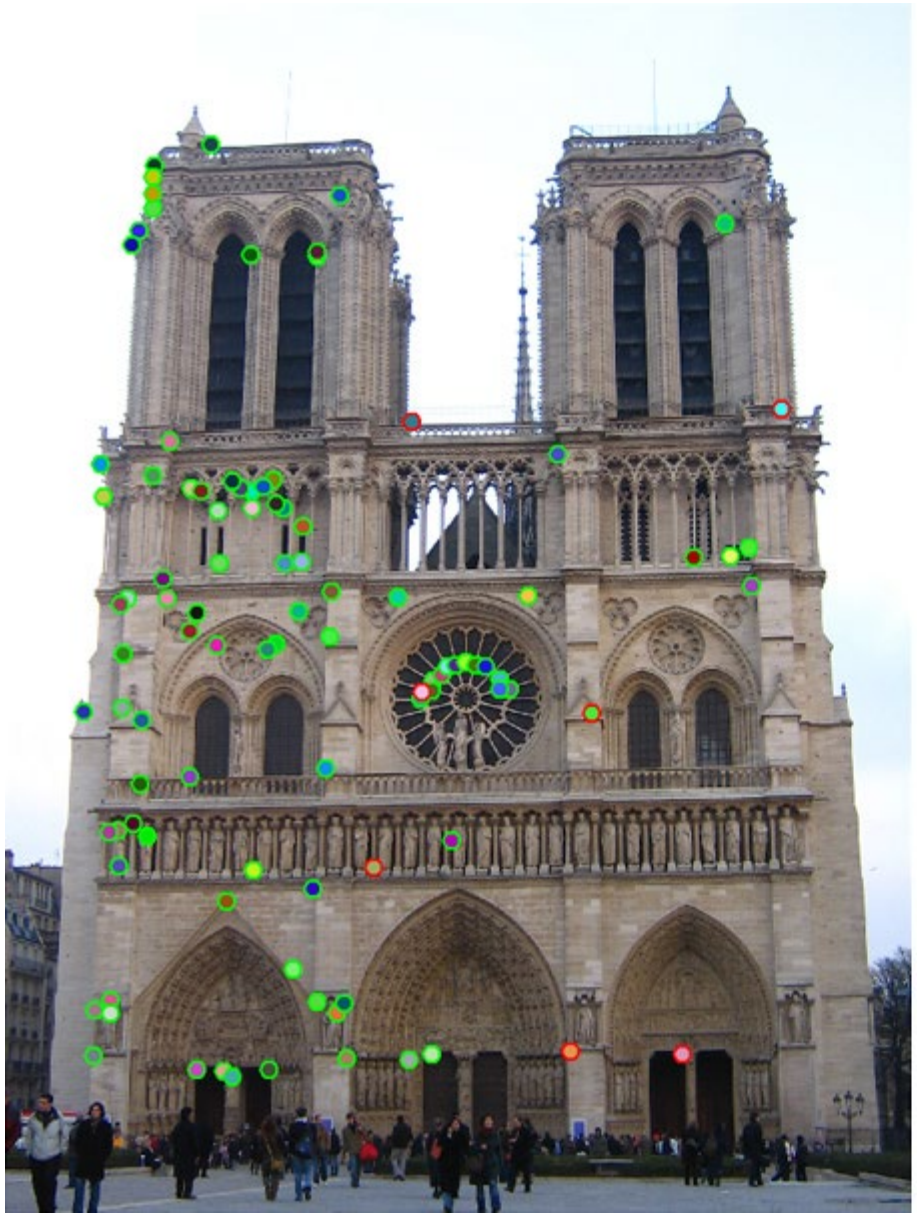
- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG

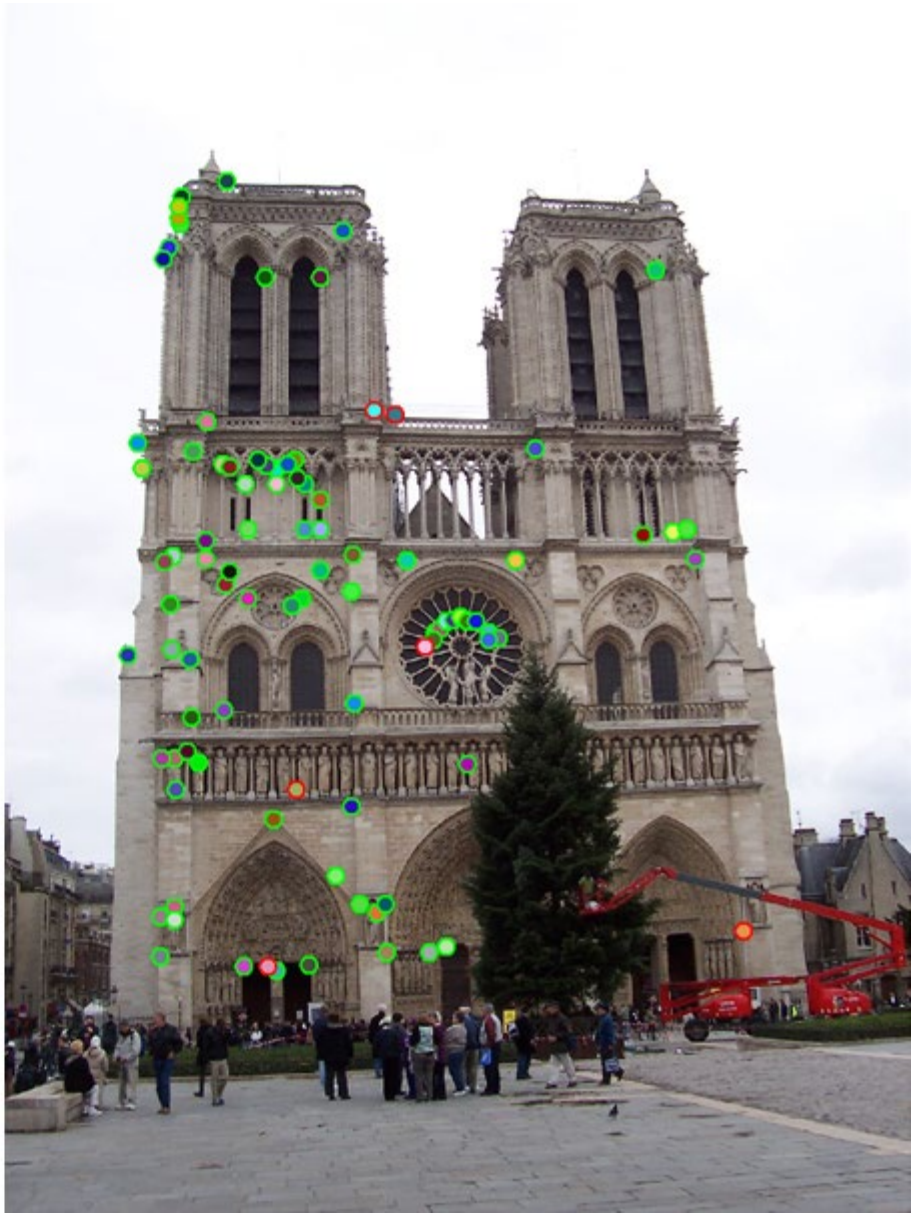


- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT and variants are typically good for stitching and recognition
  - But, need not stick to one



Which features match?





Feature matching

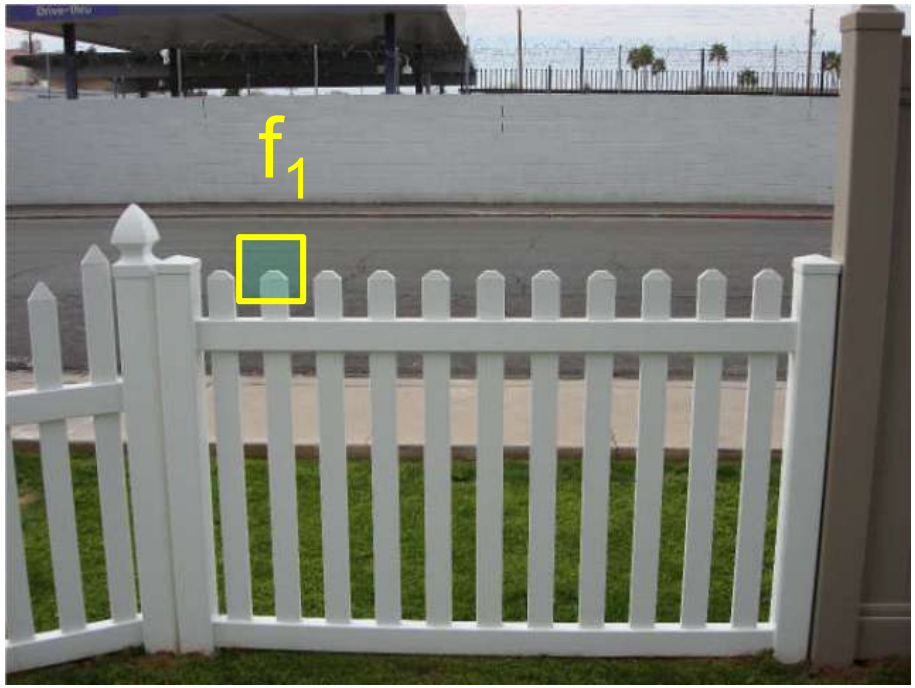
Given a feature in I_1, how to find the best match in I_2?

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

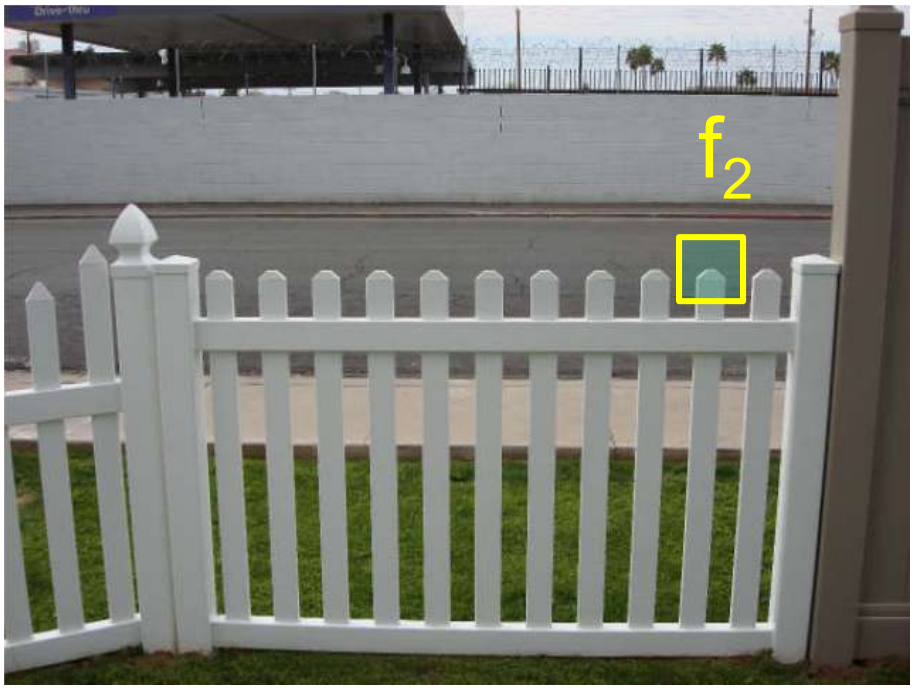
Feature distance

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

- Simple approach: L_2 distance, $||f_1 - f_2||$ Search L2-Norm
- can give small distances for ambiguous (incorrect) matches



I_1

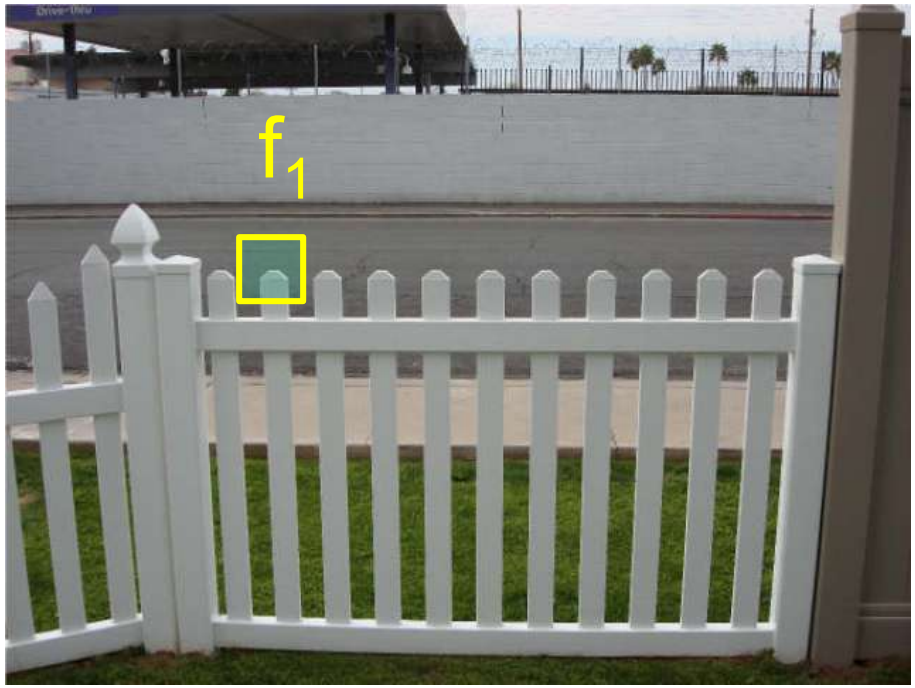


I_2

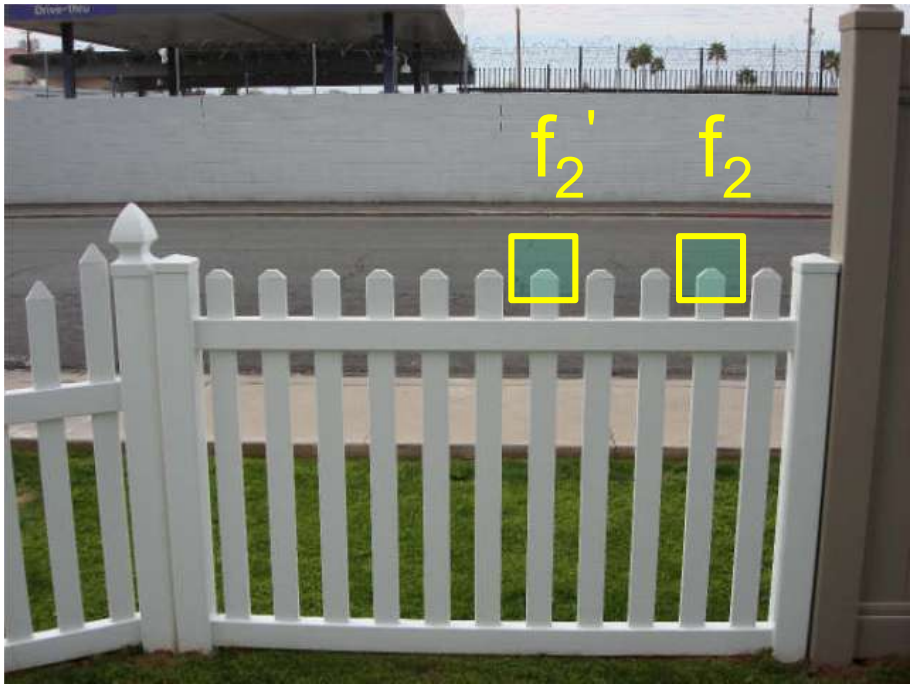
Feature distance

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 best SSD match to f_1 in I_2
  - f_2' is 2nd best SSD match to f_1 in I_2
  - gives large values for ambiguous matches



I_1

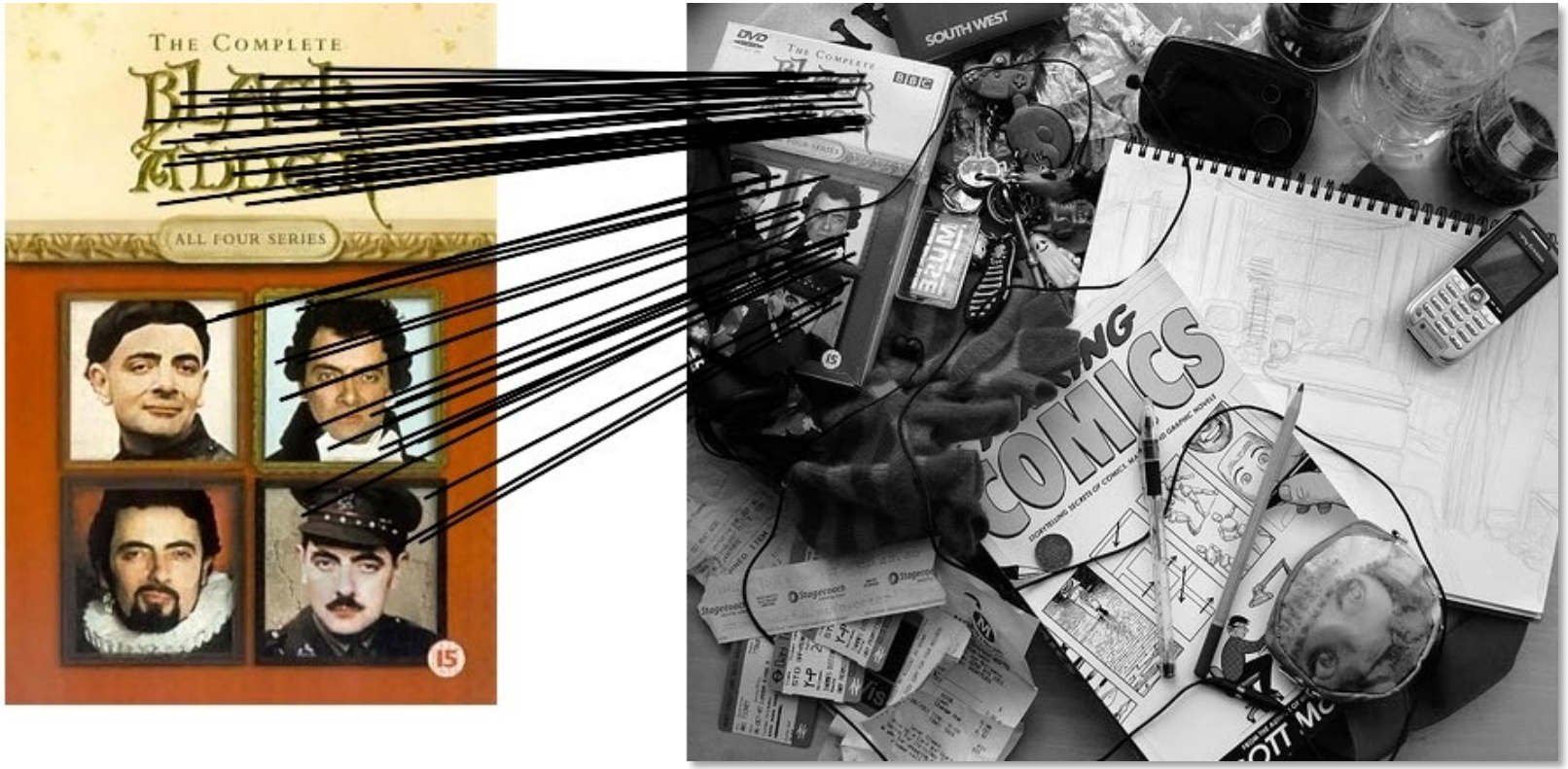


I_2

Feature distance

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

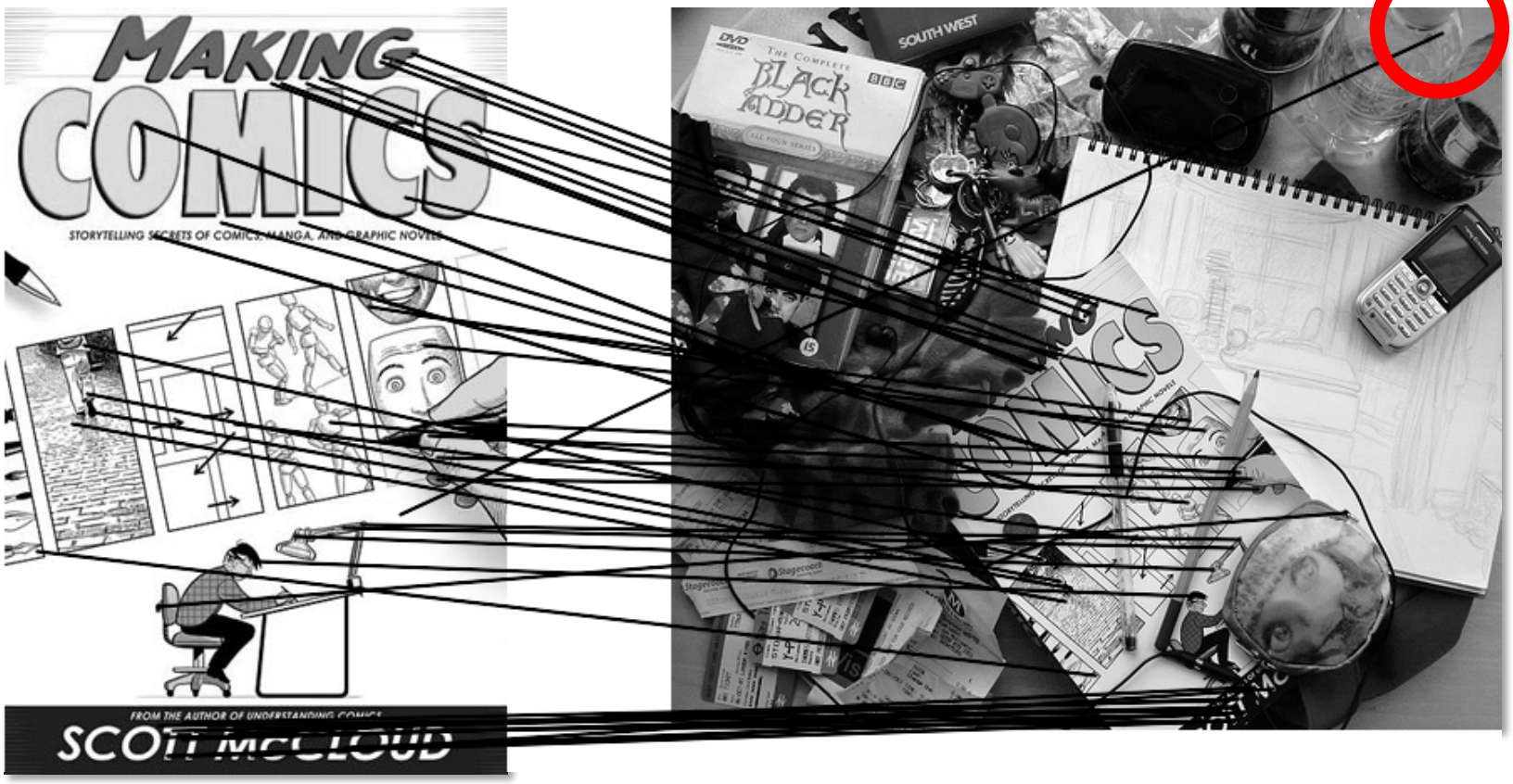
Feature matching example



58 matches (thresholded by ratio score)

Feature matching example

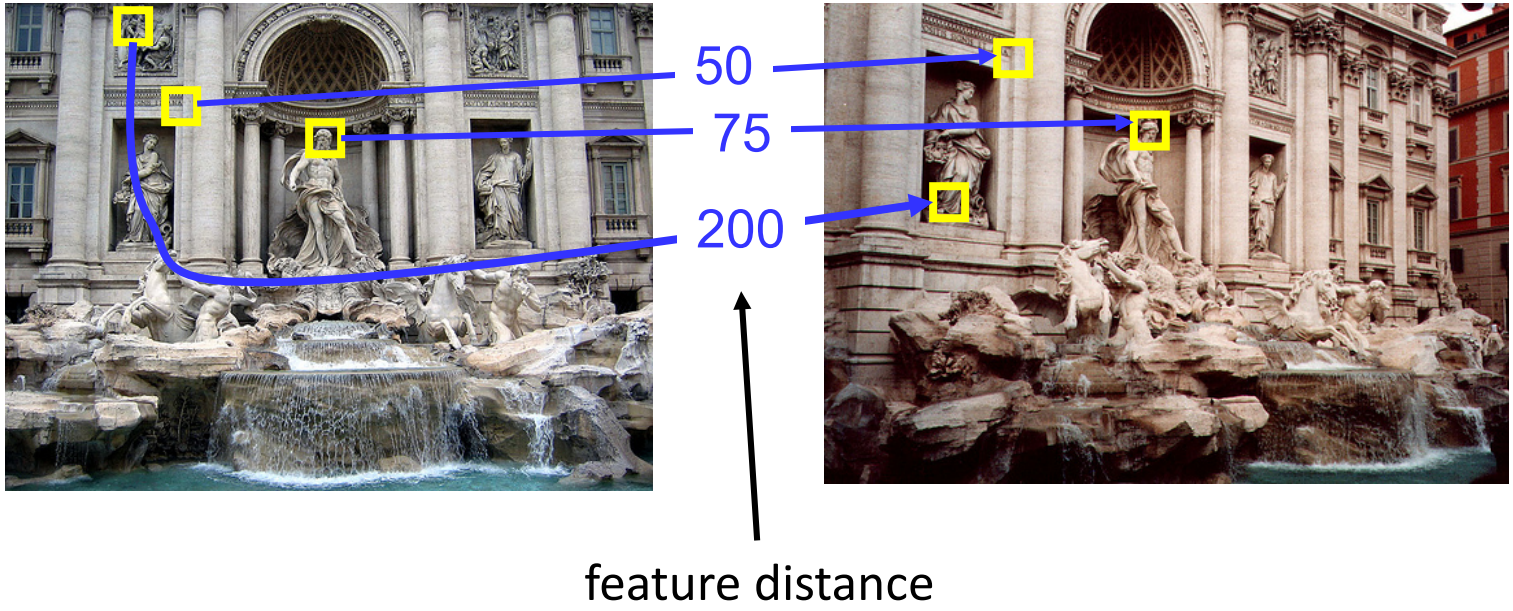
We'll deal with **outliers** later



51 matches (thresholded by ratio score)

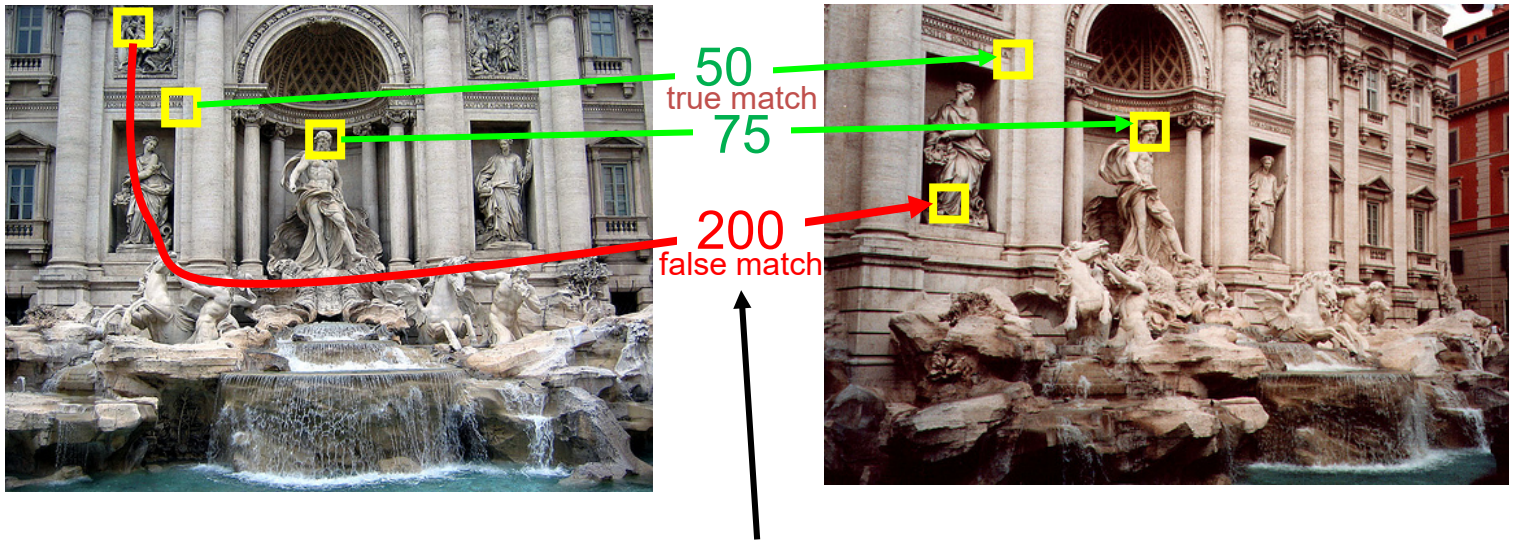
Evaluating the results

How can we measure the performance of a feature matcher?



True/false positives

How can we measure the performance of a feature matcher?

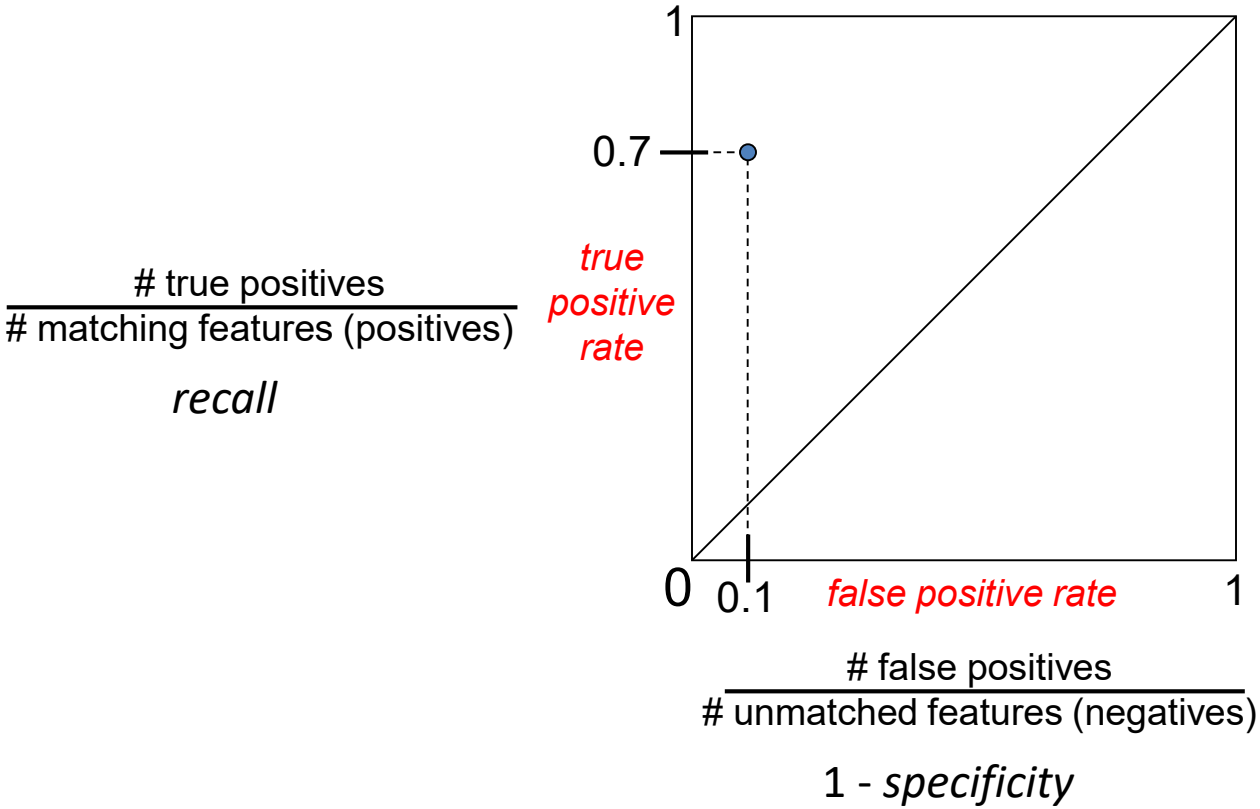


The distance threshold affects performance

- True positives = # of detected matches that 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?

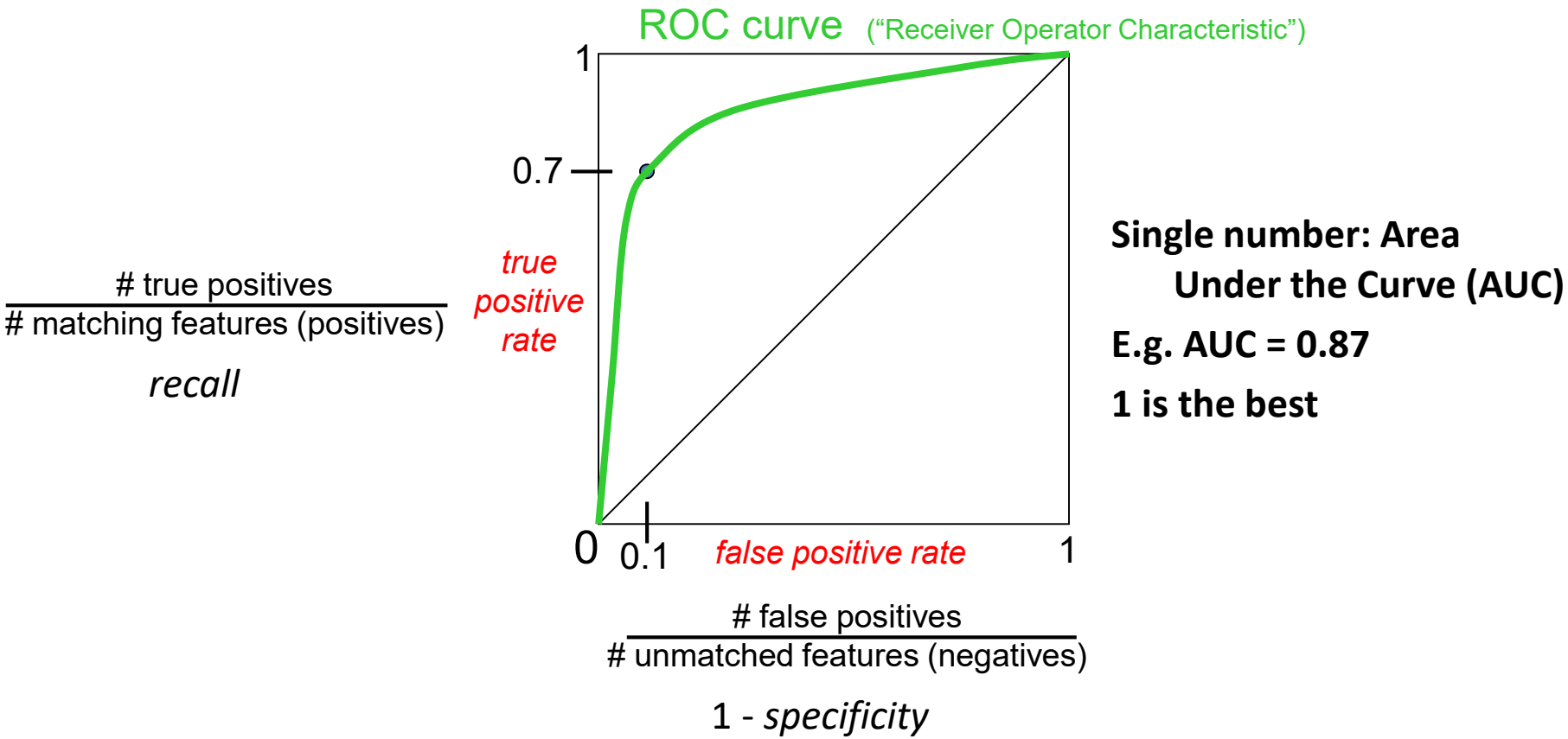
Evaluating the results

How can we measure the performance of a feature matcher?



Evaluating the results

How can we measure the performance of a feature matcher?



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