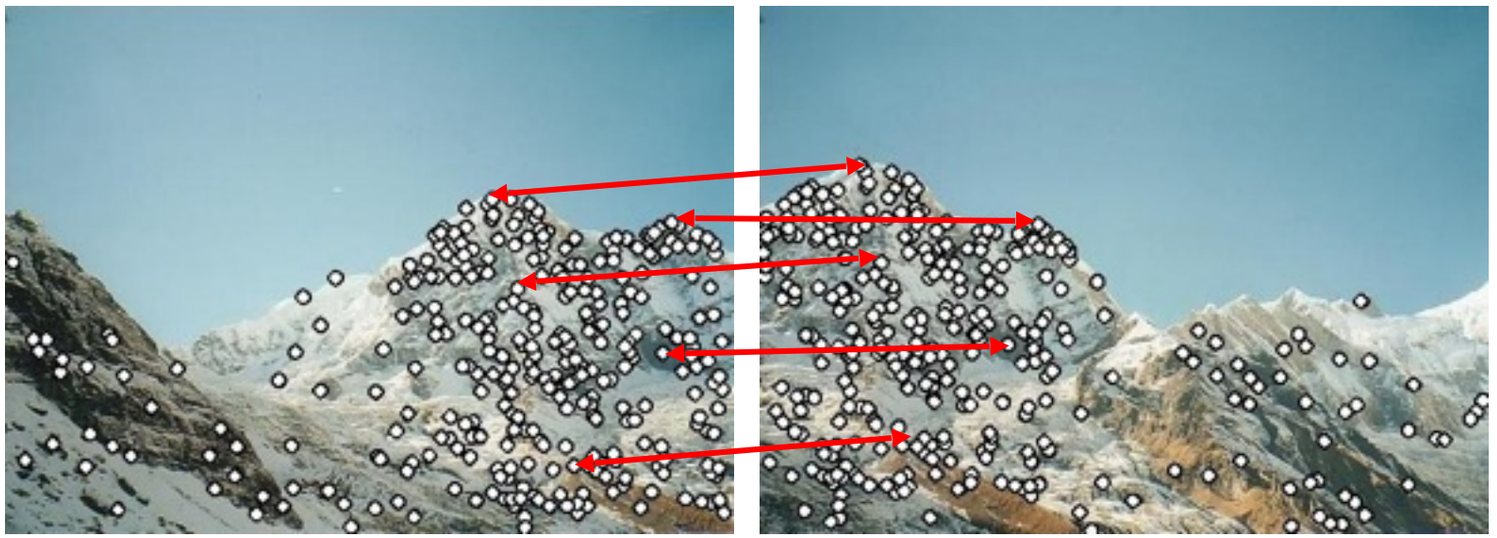
CSC 589 Introduction to Computer Vision

Lecture 18
Feature description, SIFT feature



Read Szeliski 4.1
Solem 2

Bei Xiao

Spring, 2014

American University

Finding the “same” thing across images

Categories Find a bottle:

Can’t do
unless you do not
care about few errors…

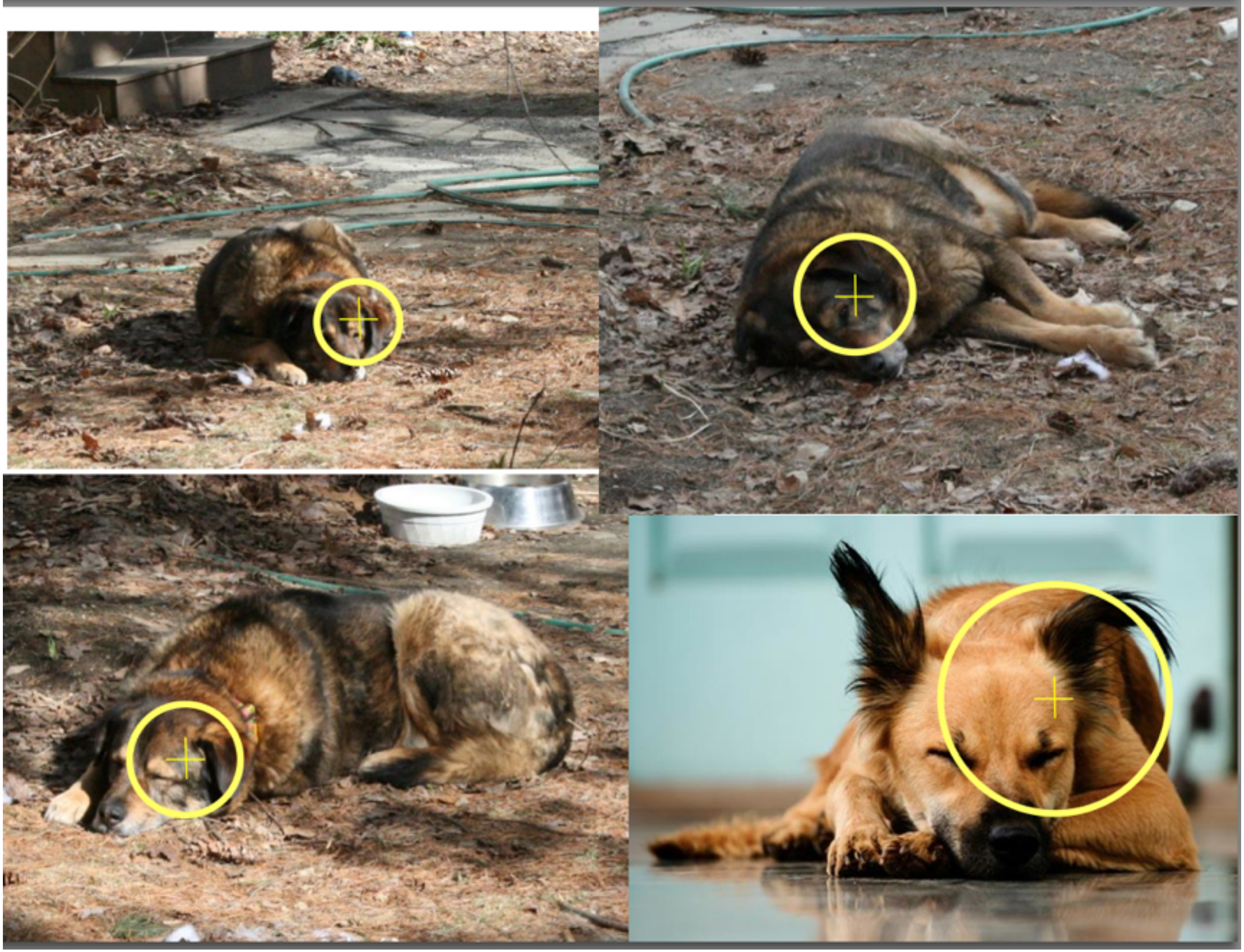
Instances Find these two objects

Can nail it

Figure 12: The training images for two objects are shown on the left. These can be recognized in a cluttered image with extensive occlusion, shown in the middle. The results of recognition are shown on the right. A parallelogram is drawn around each recognized object showing the boundaries of the original training image under the affi ne transformation solved for during recognition. Smaller squares indicate the keypoints that were used for recognition.

But where is that point?

M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

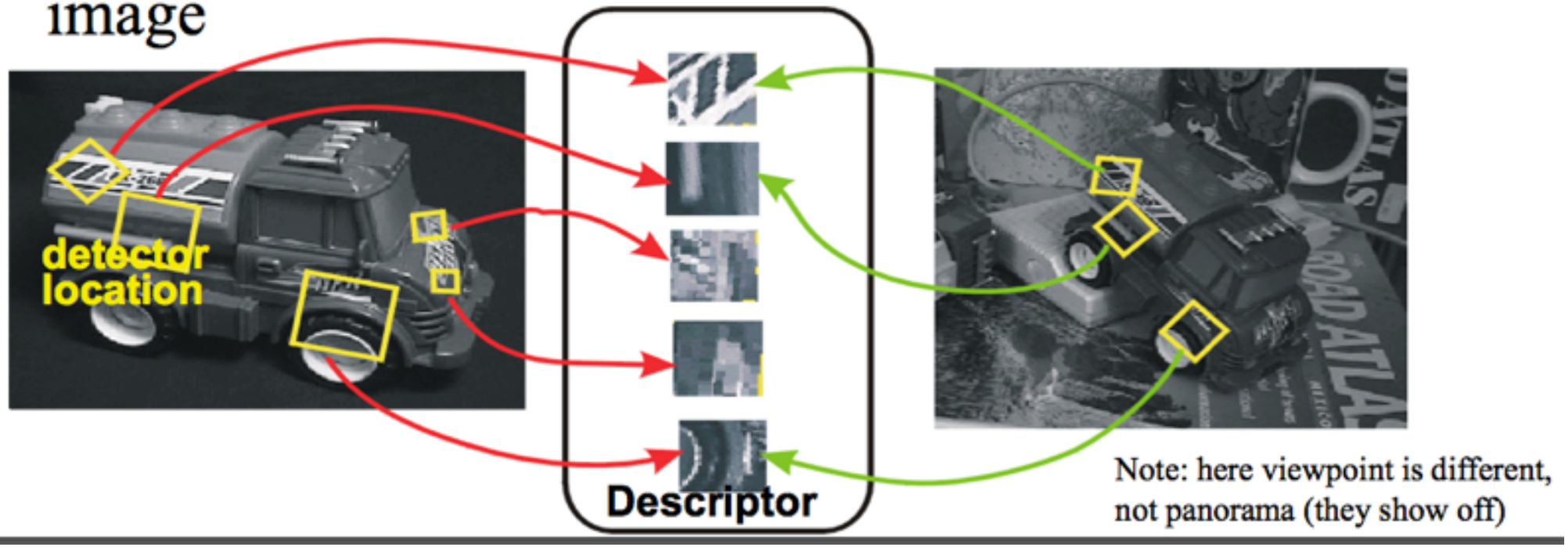


Uses for feature point detectors and descriptors in computer vision and graphics.

- Image alignment and building panoramas
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
- … other

Preview

- **Detector**: detect same scene points independently in both images
- **Descriptor**: encode local neighboring window
  - Note how scale & rotation of window are the same in both image (but computed independently)
- **Correspondence**: find most similar descriptor in other image

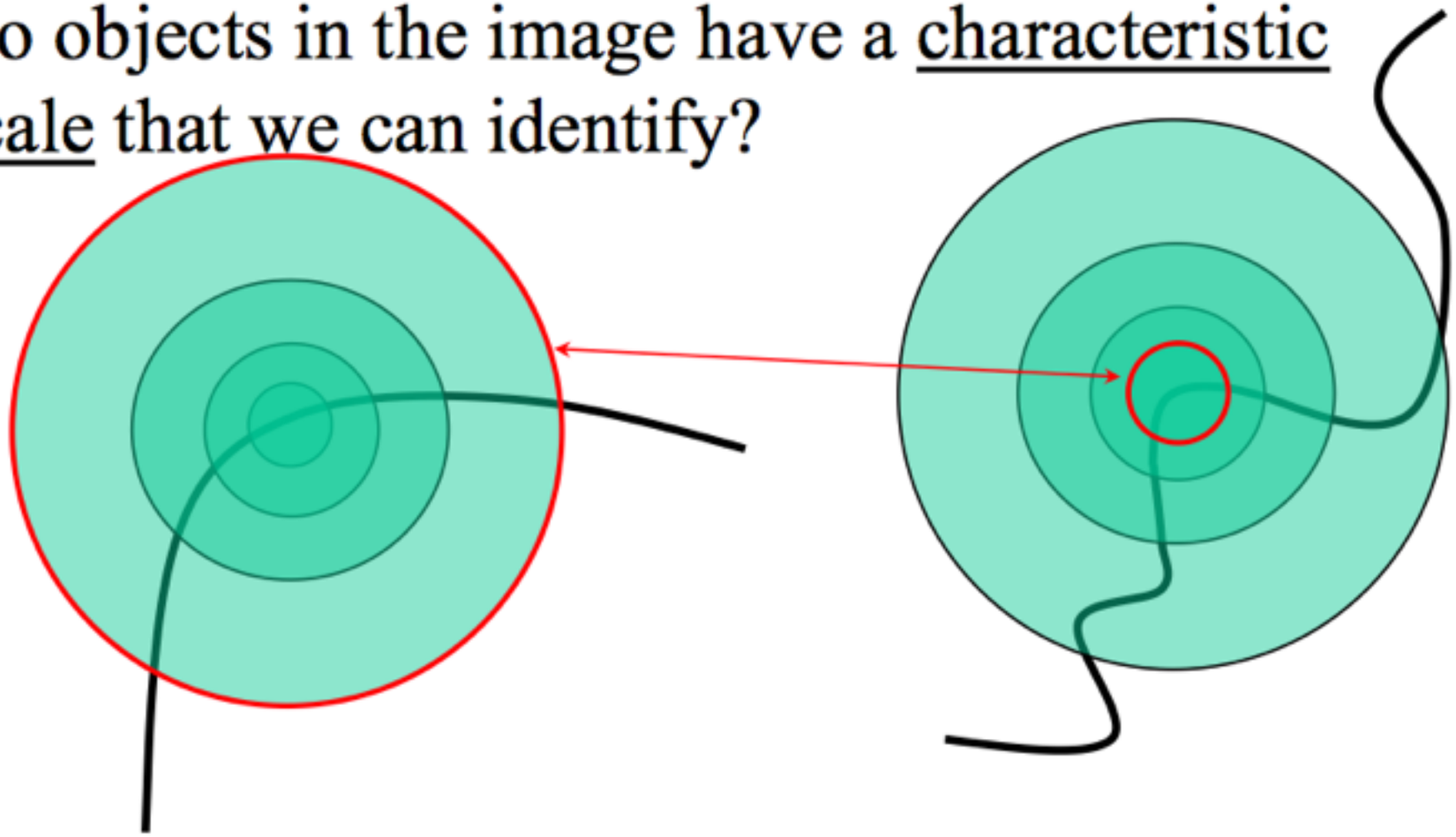


Outline

- Feature point detection
  - Harris corner detector
  - finding a characteristic scale: DoG or Laplacian of Gaussian
- Local image description
  - SIFT features

Last lecture: Scale invariance

- The problem: how do we choose corresponding circles ***independently*** in each image?
- Do objects in the image have a characteristic scale that we can identify?



Laplacian of Gaussian for selection of characteristic scale

http://www.robots.ox.ac.uk/~vgg/research/affine/det_eval_files/mikolajczyk_ijcv2004.pdf

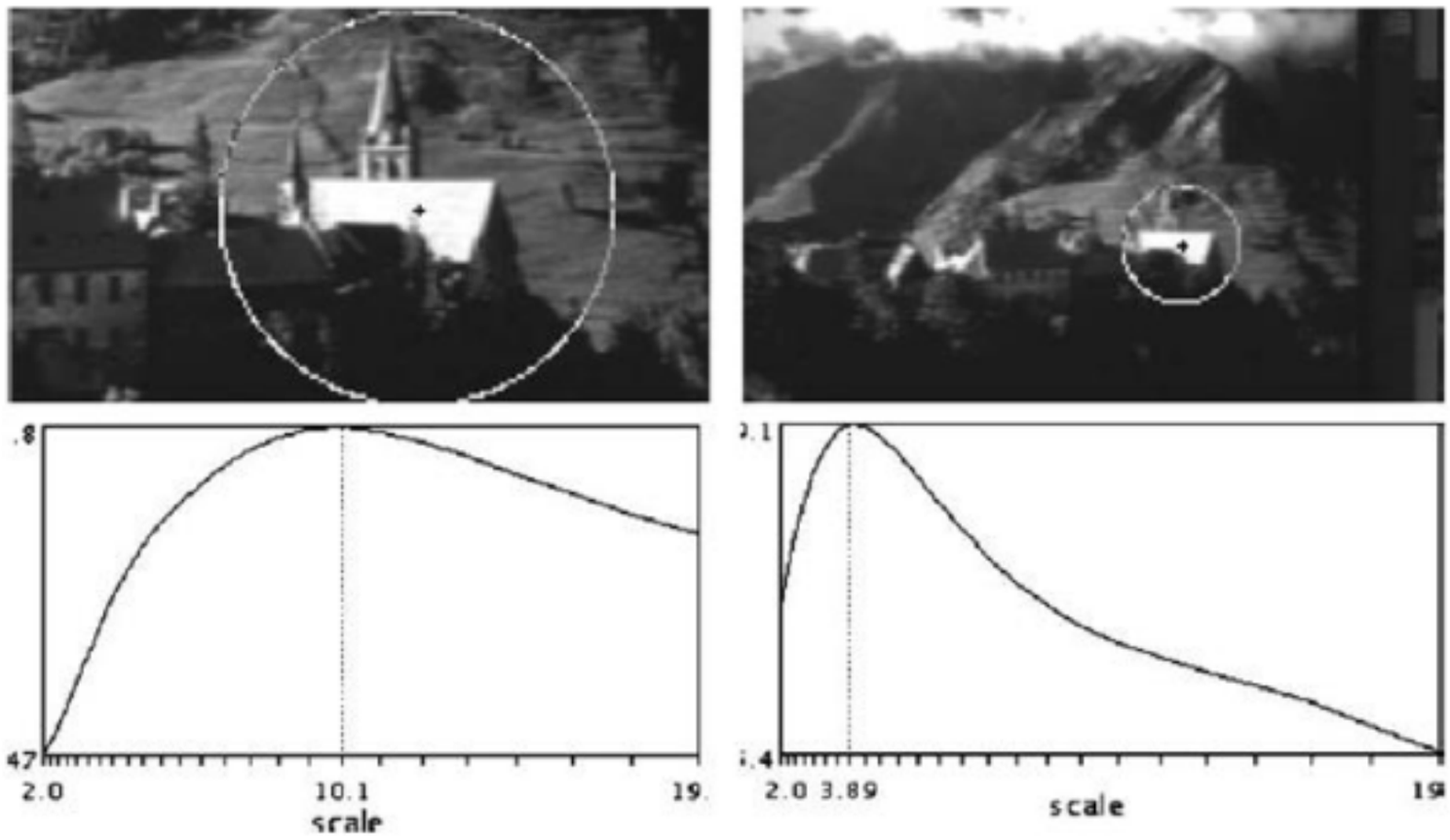


Figure 1. Example of characteristic scales. The top row shows two images taken with different focal lengths. The bottom row shows the response $F_{\text{norm}}(\mathbf{x}, \sigma_n)$ over scales where F_{norm} is the normalized LoG (cf. Eq. (3)). The characteristic scales are 10.1 and 3.89 for the left and right image, respectively. The ratio of scales corresponds to the scale factor (2.5) between the two images. The radius of displayed regions in the top row is equal to 3 times the characteristic scale.

Scale Invariant Detection

- Functions for determining scale $f = \text{Kernel} * \text{Image}$

Kernels:

$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

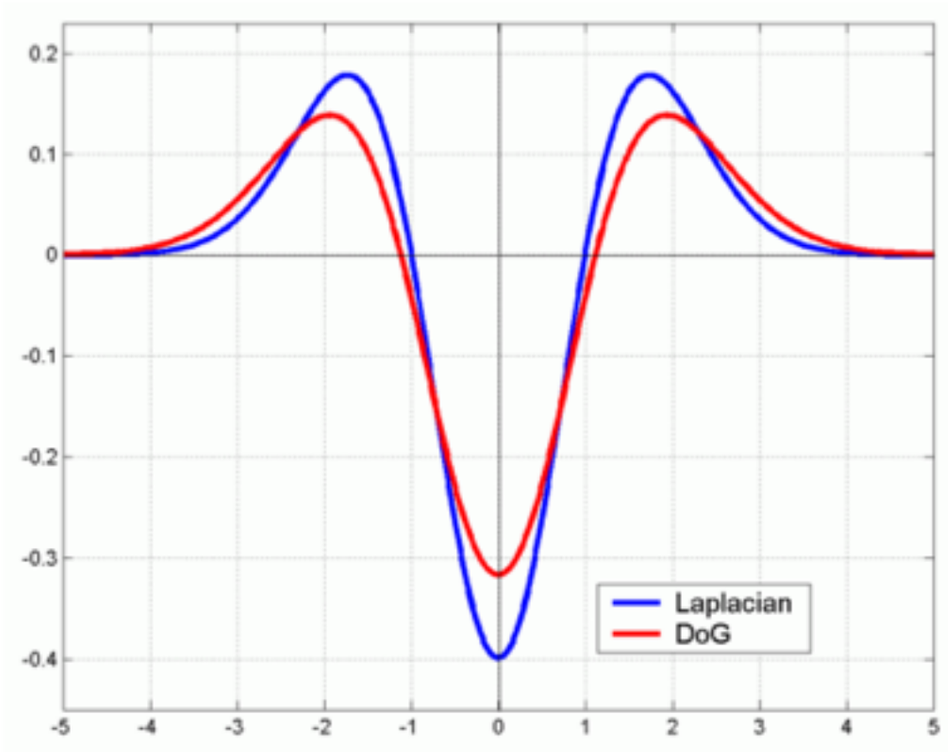
(Laplacian: 2nd derivative of Gaussian)

$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(Difference of Gaussians)

where Gaussian

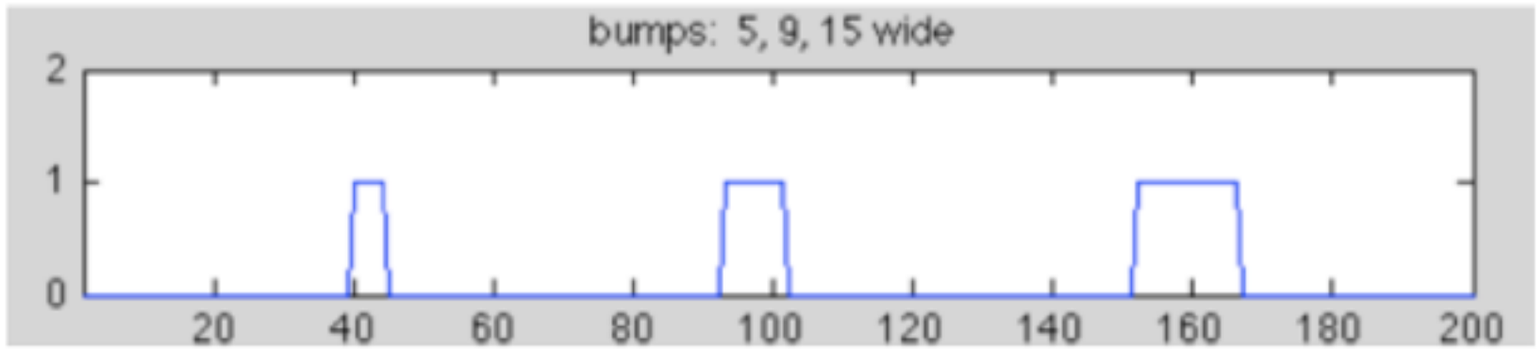
$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2 + y^2}{2\sigma^2}}$$



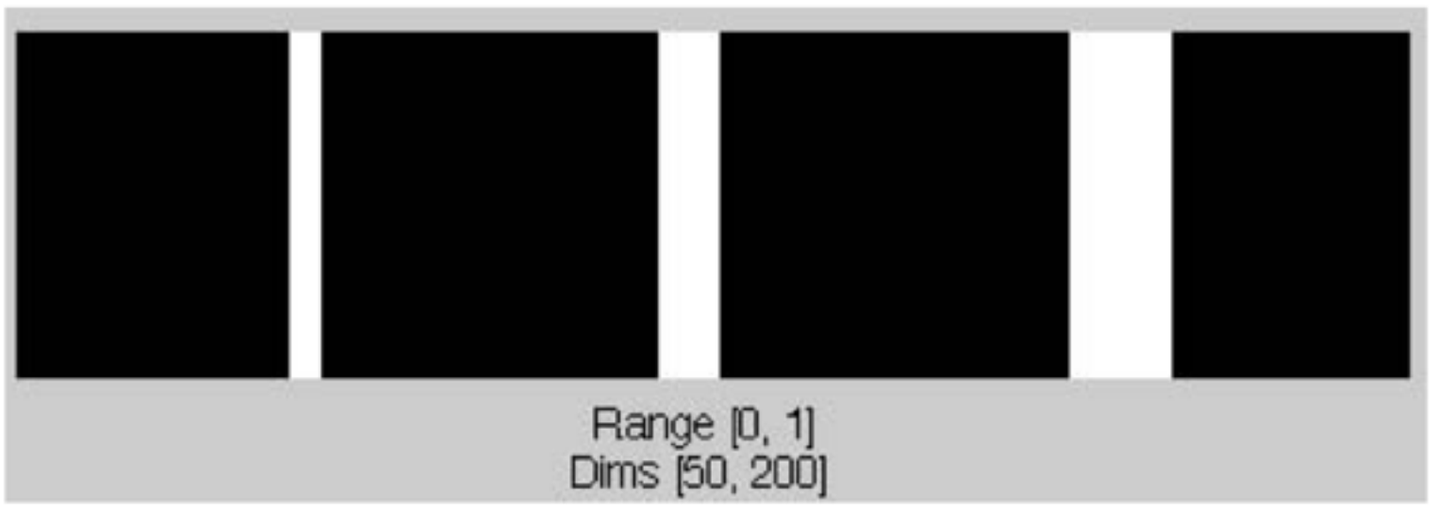
Note: both kernels are invariant to *scale* and *rotation*

Scale-space example: 3 bumps of different widths.

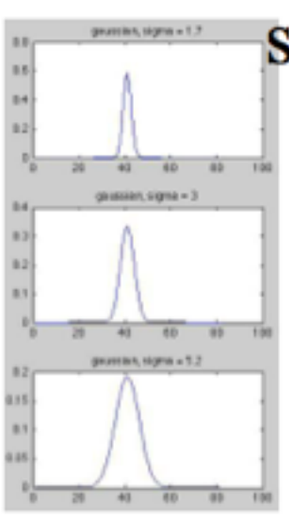
1-d bumps



display as an image

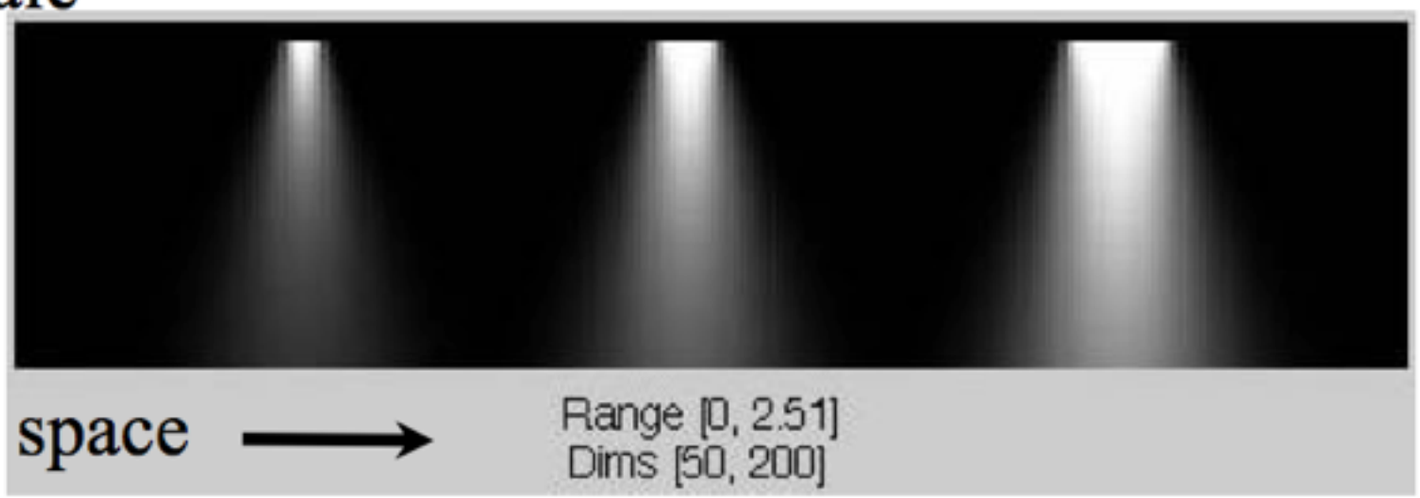


blur with Gaussians of increasing width

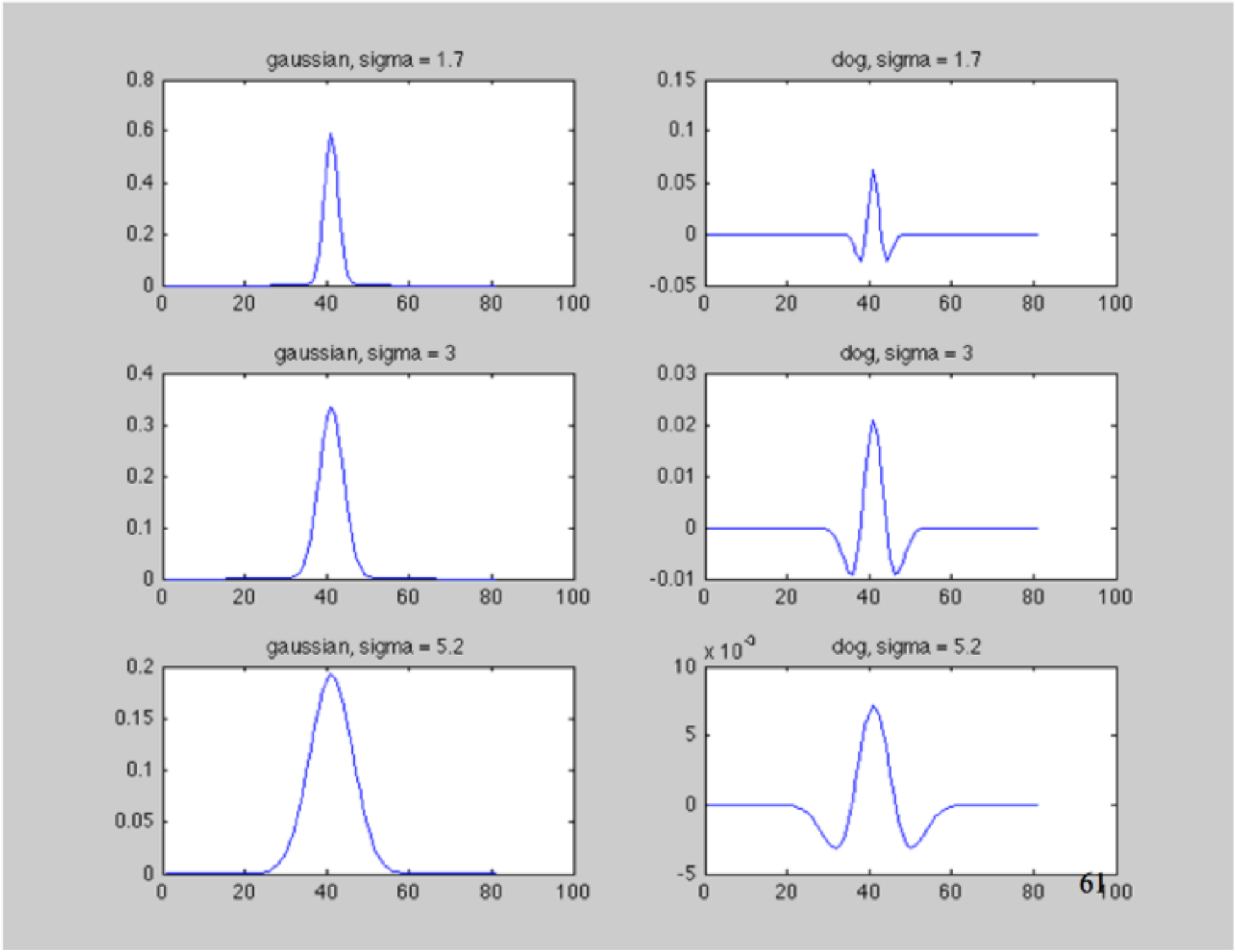


scale

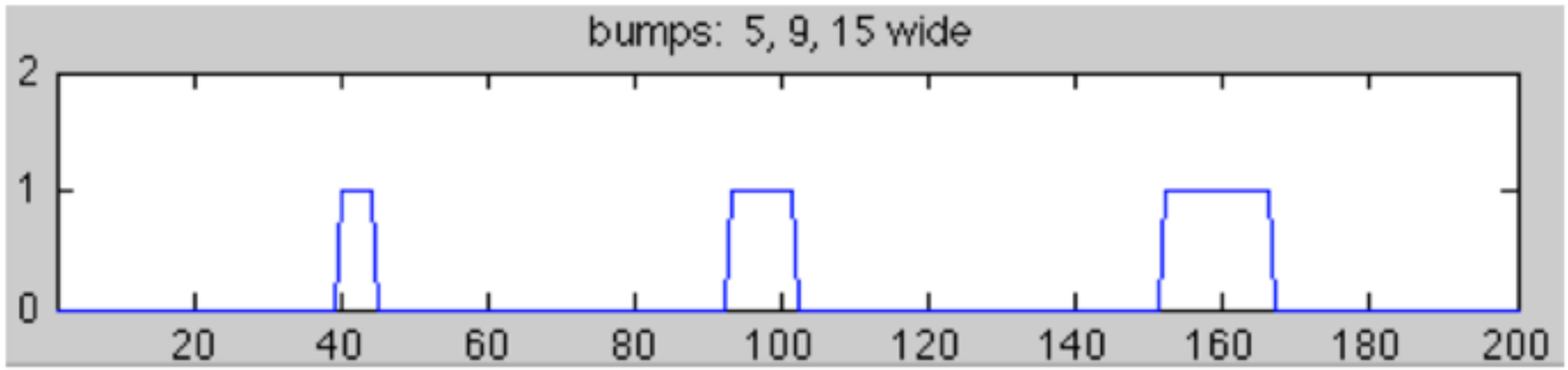
space



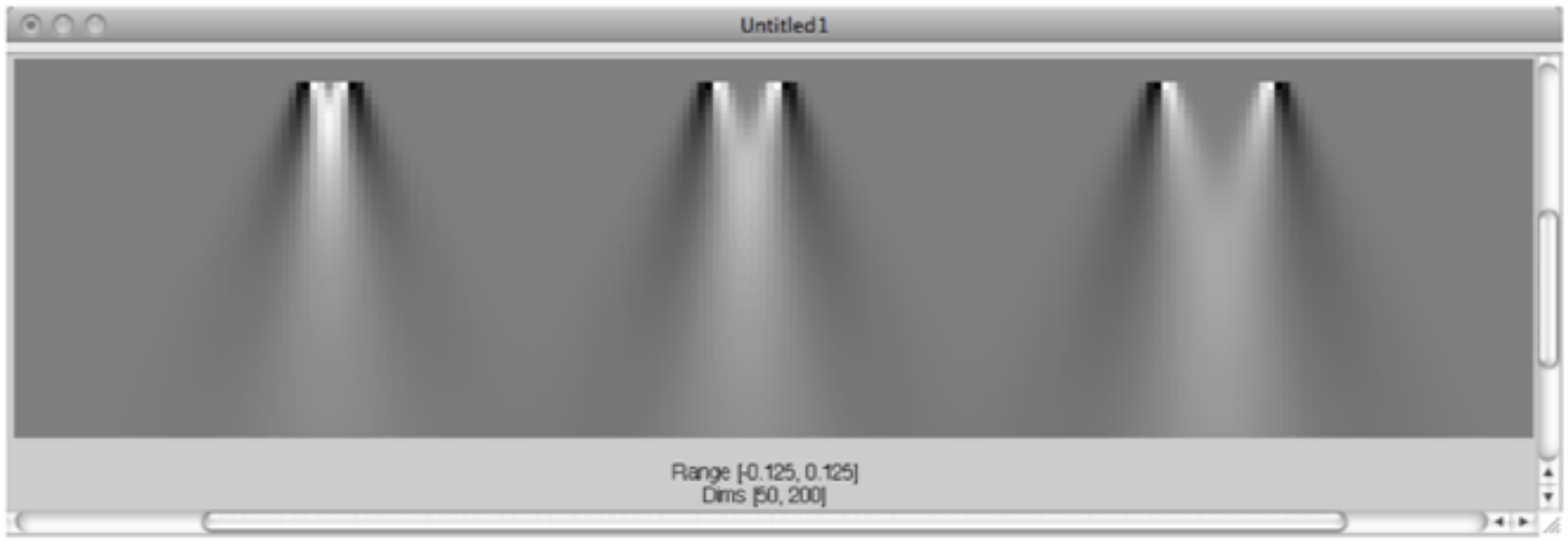
Gaussian and difference-of-Gaussian filters



The bumps, filtered by difference-of-Gaussian filters

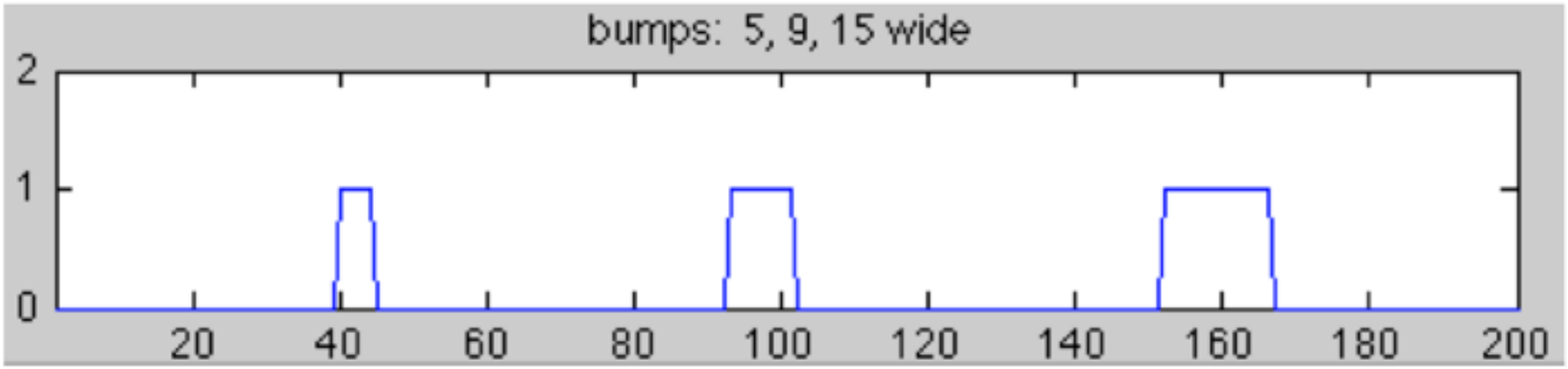


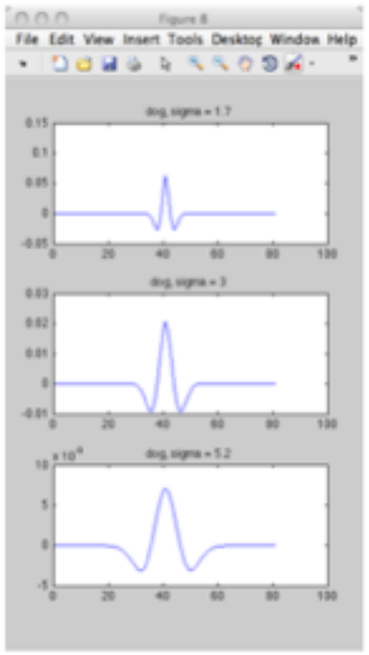
scale

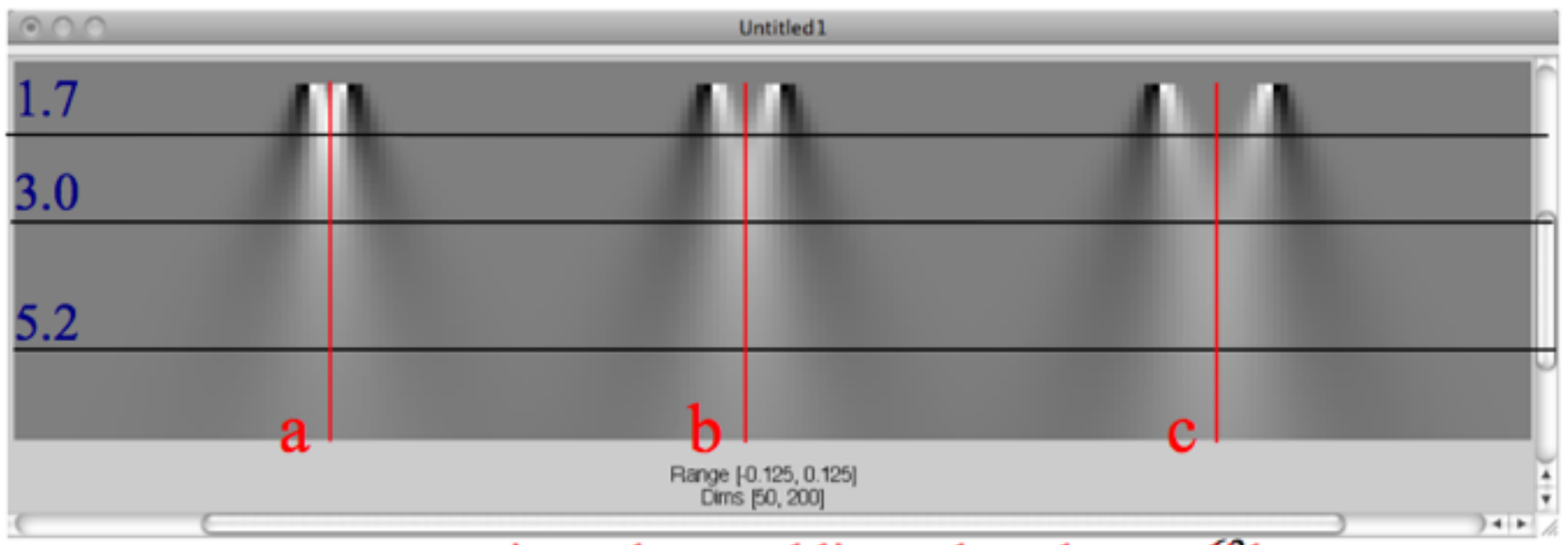


space

The bumps, filtered by difference-of-Gaussian filters



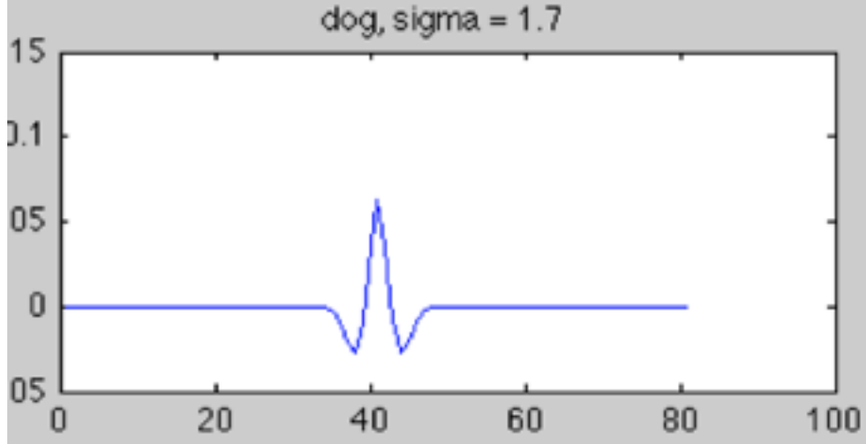


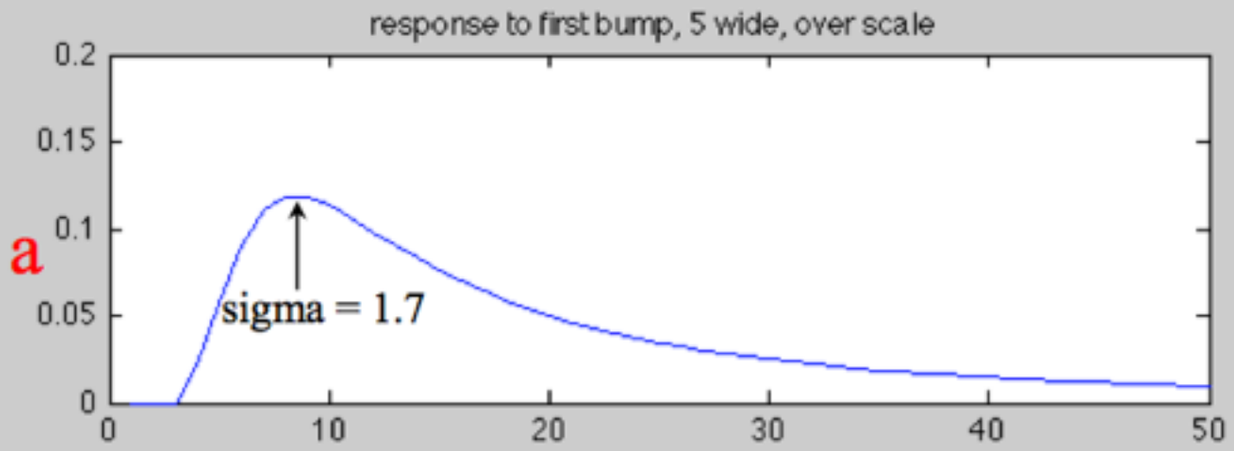


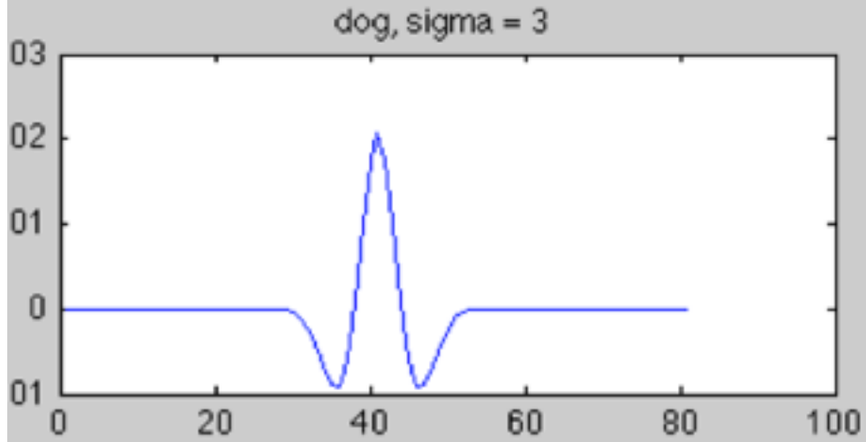
cross-sections along red lines plotted next slide

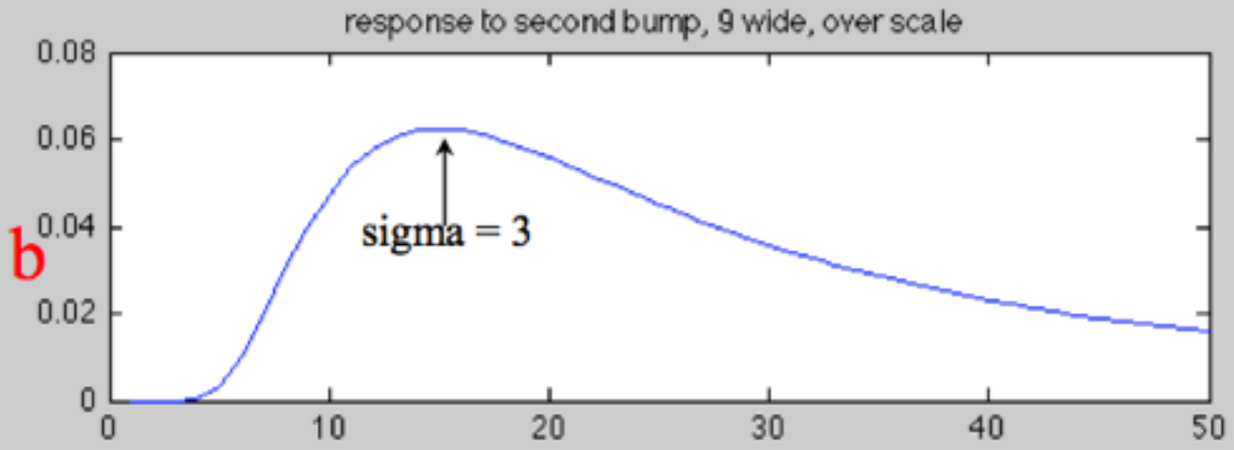
Scales of peak responses are proportional to bump width (the characteristic scale of each bump):

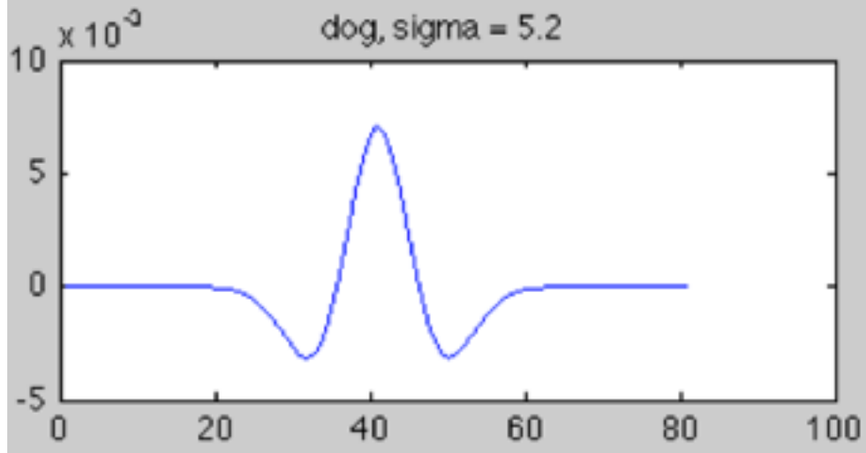
[1.7, 3, 5.2] ./ [5, 9, 15] = 0.3400 0.3333 0.3467

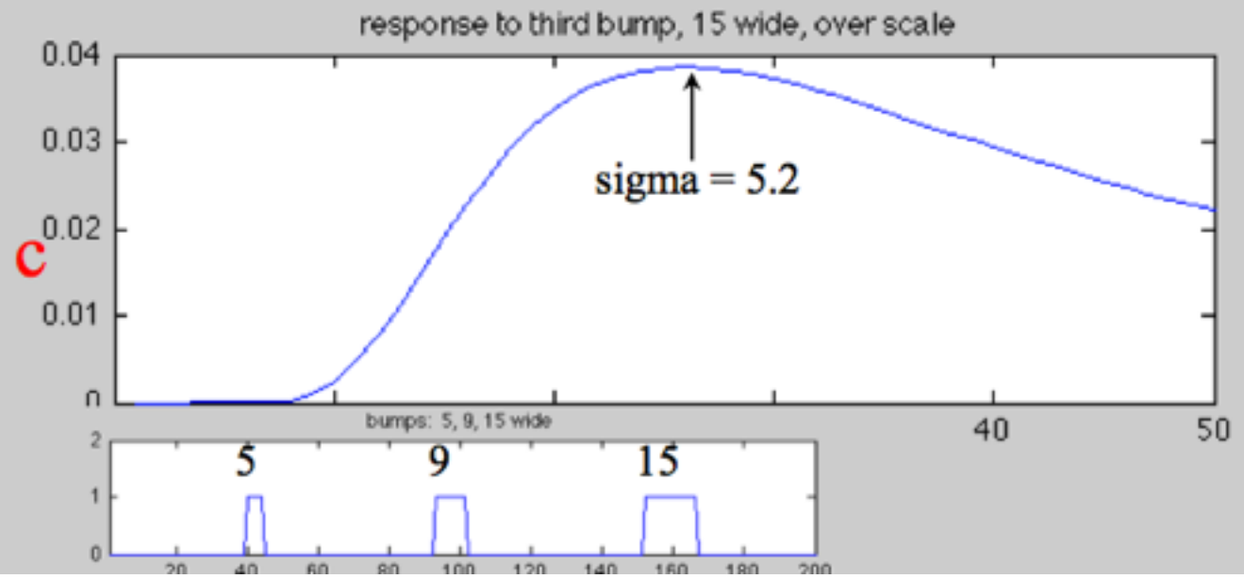




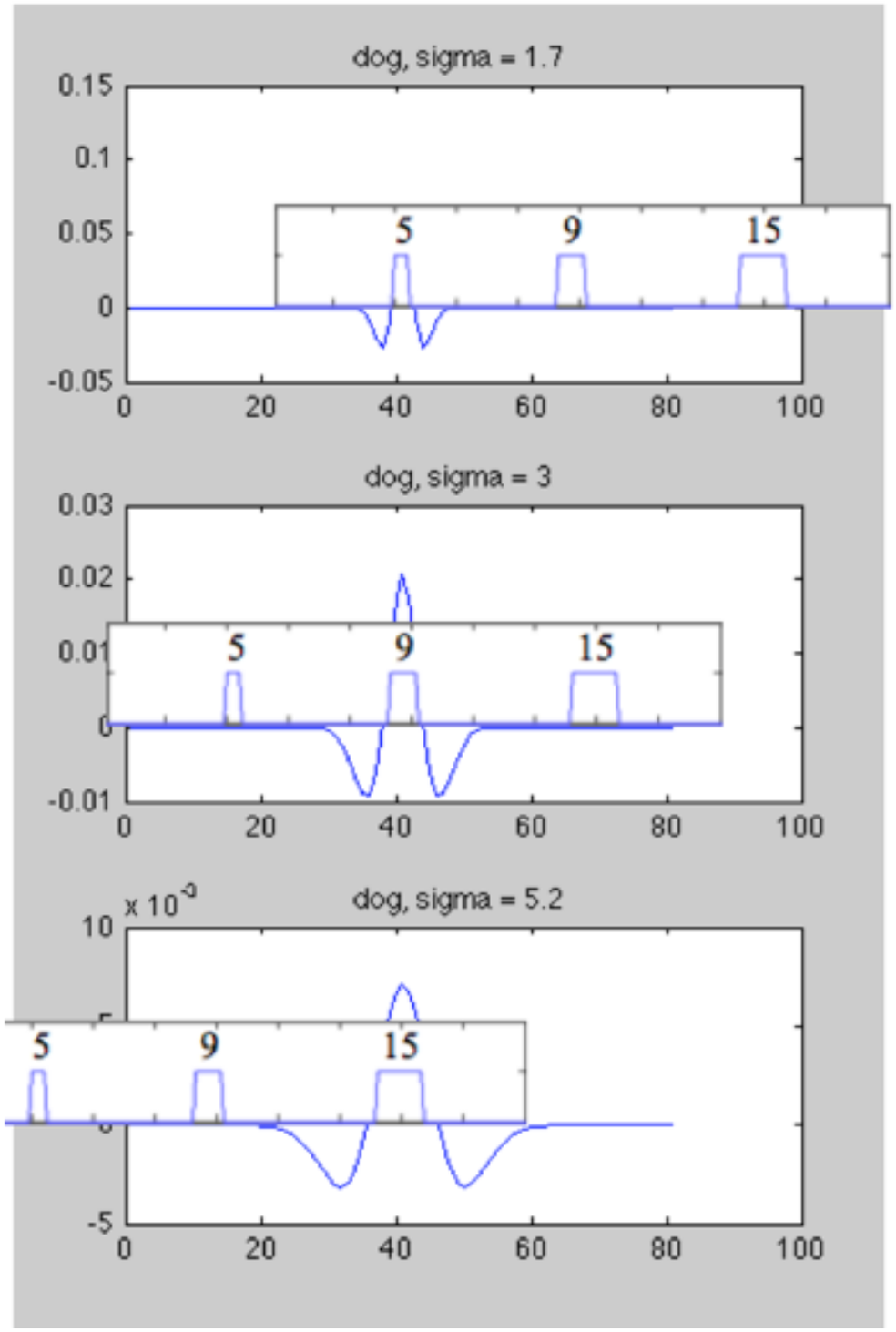








Diff of Gauss filter giving peak response



Scales of peak responses are proportional to bump width (the characteristic scale of each bump):

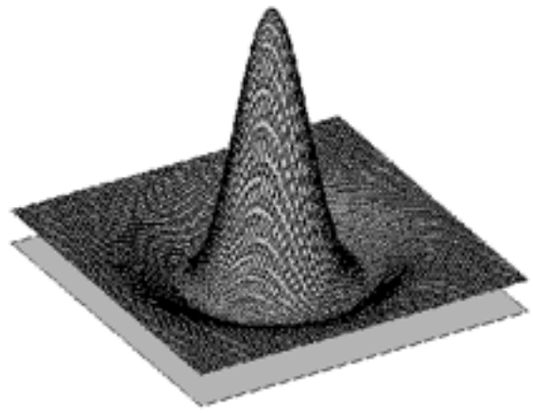
[1.7, 3, 5.2] ./ [5, 9, 15] = 0.3400

0.3333 0.3467

Note that the max response filters each has the same relationship to the bump that it favors (the zero crossings of the filter are about at the bump edges). So the scale space analysis correctly picks out the "characteristic scale" for each of the bumps.

More generally, this happens for the features of the images we analyze.

Difference-of-Gaussian (DoG)



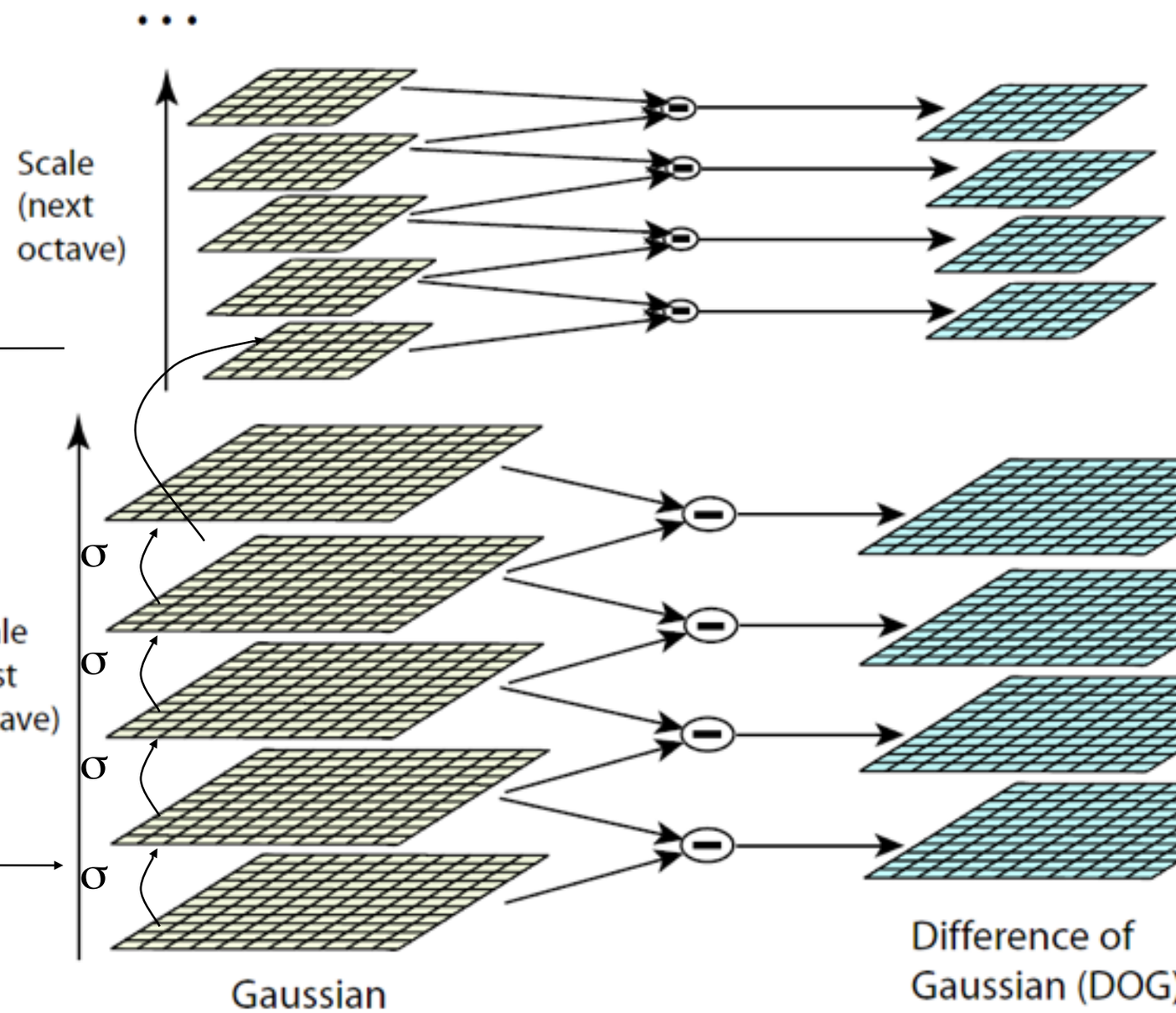
K. Grauman, B. Leibe

DoG – Efficient Computation

- Computation in Gaussian scale pyramid

Sampling with step $\sigma^4 = 2$

Original image

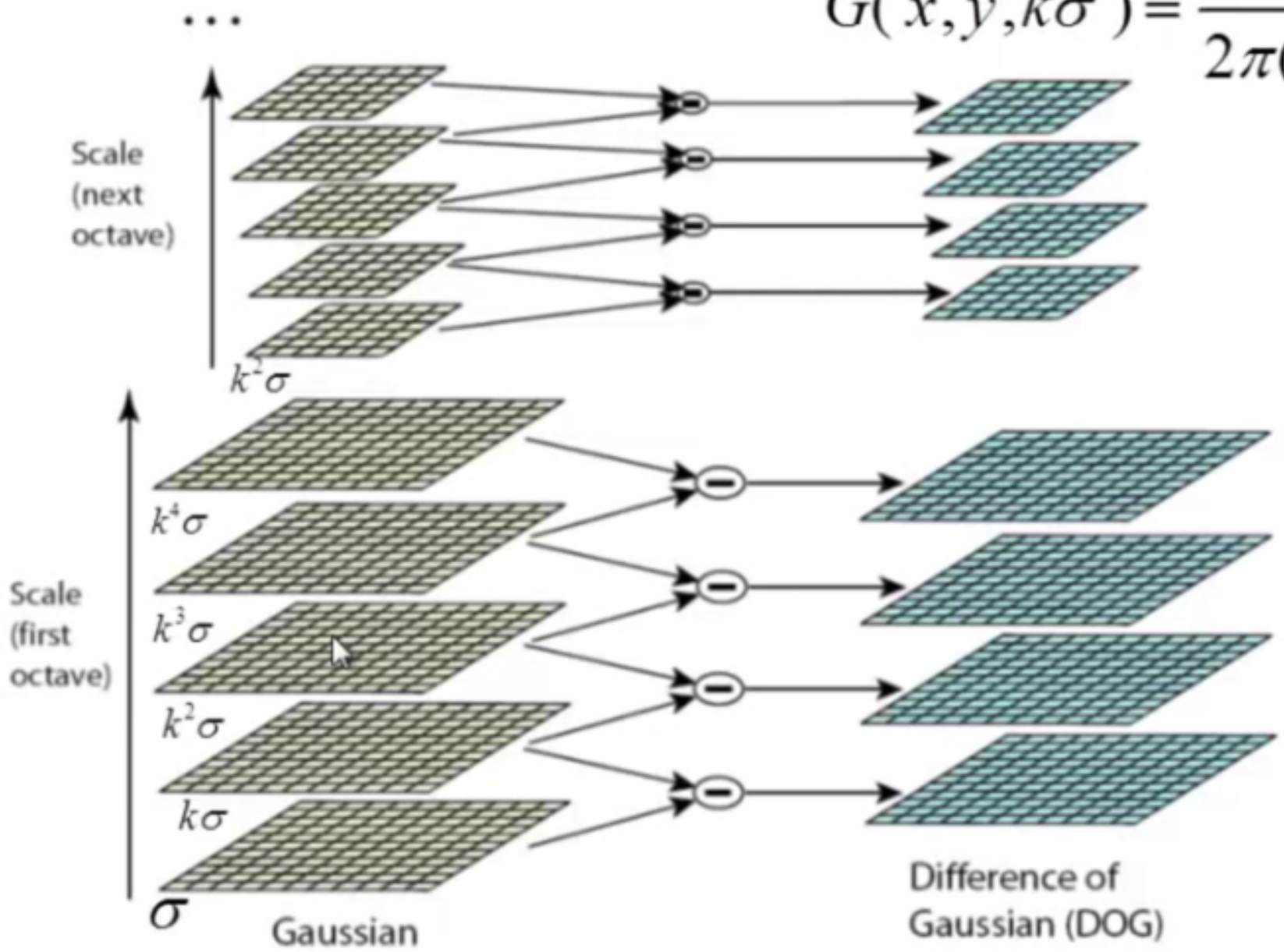


$$\sigma = 2^{\frac{1}{4}}$$

K. Grauman, B. Leibe

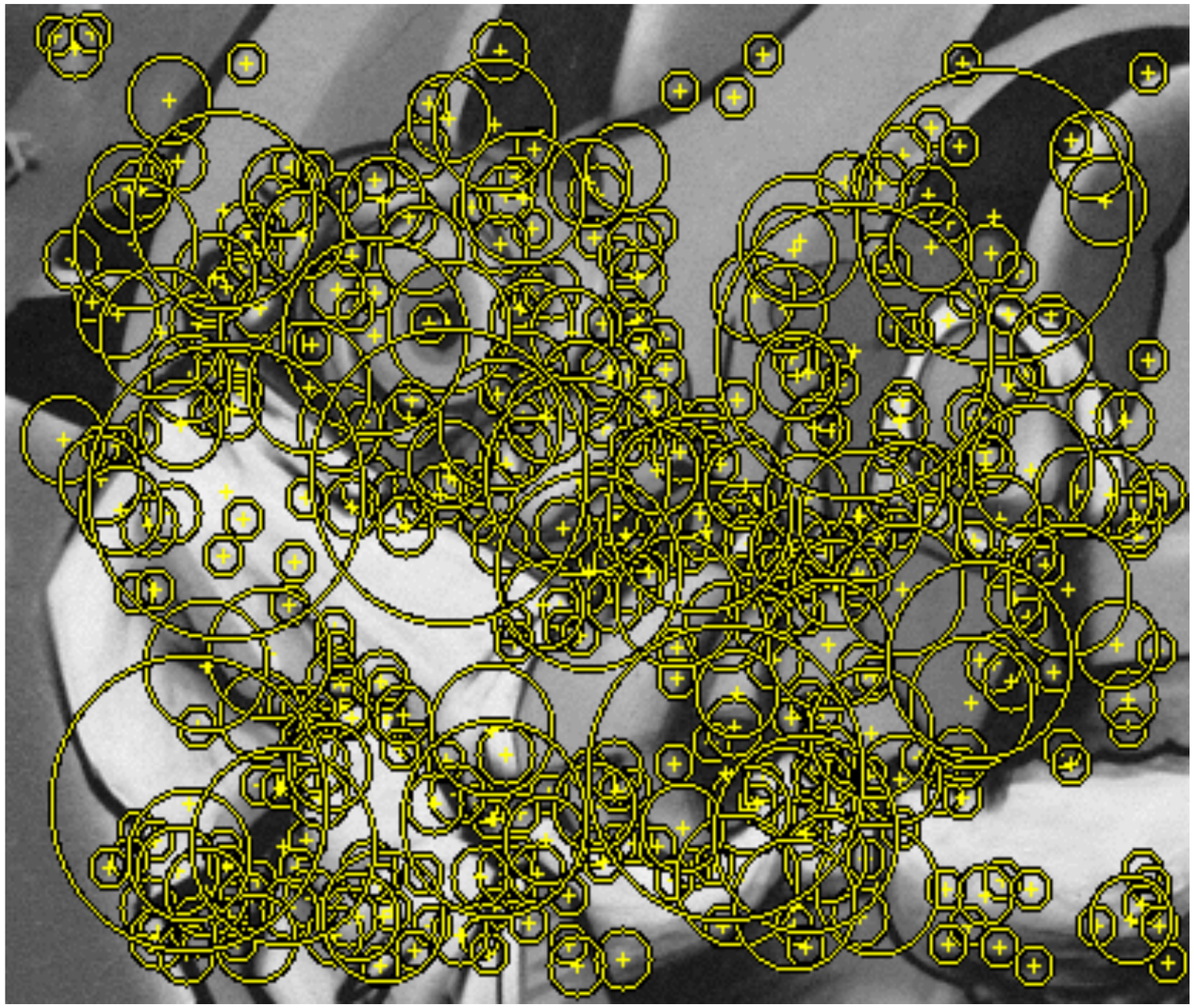
Building a Scale Space

$$G(x, y, k\sigma) = \frac{1}{2\pi(k\sigma)^2} e^{-(x^2+y^2)/2k^2\sigma^2}$$



$$k = \sqrt{2}$$

Results: Difference-of-Gaussian



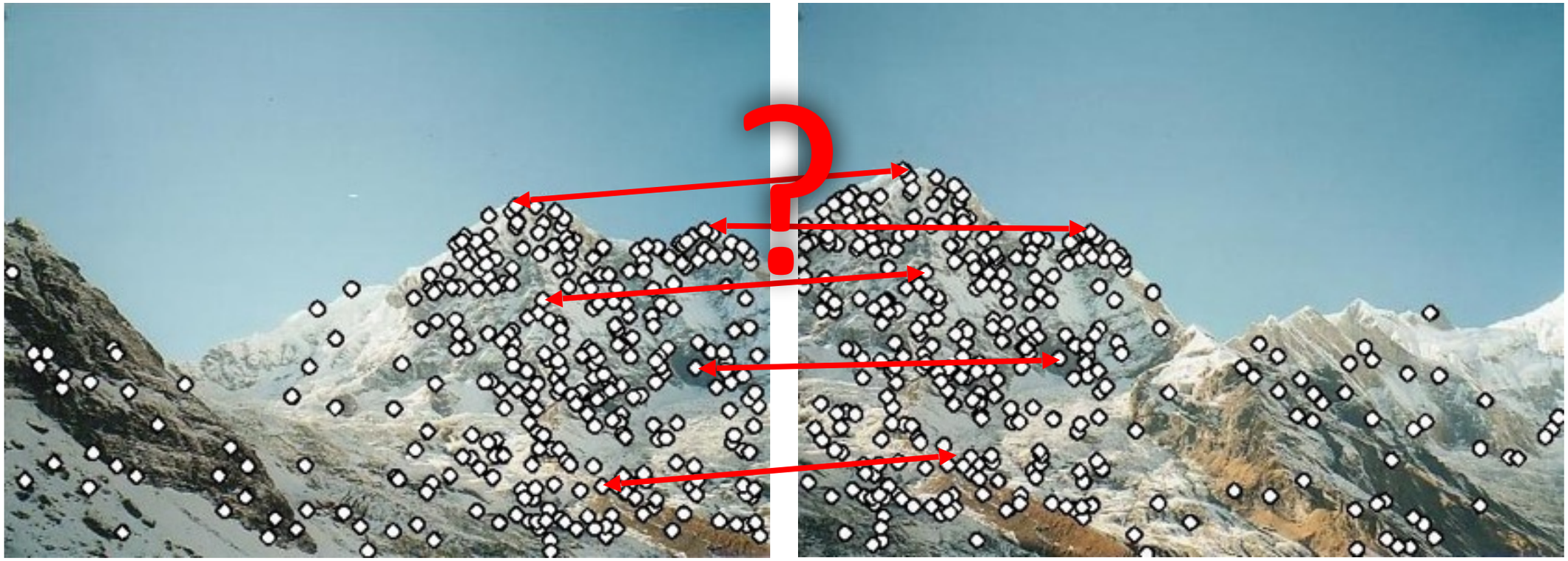
Outline

- Feature point detection
  - Harris corner detector
  - finding a characteristic scale
- Local image description
  - SIFT features

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

Invariance vs. discriminability

- Invariance:
  - Descriptor shouldn't change even if image is transformed

- Discriminability:
  - Descriptor should be highly unique for each point

Invariance

- Most feature descriptors are 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 transformations (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 window of pixels
     - What's this invariant to?
   - Let's look at some better approaches…

Rotation invariance for feature descriptors

- Find dominant orientation of the image patch
  - This is given by $\mathbf{x}_{max}$, the eigenvector of $\mathbf{M}$ corresponding to λ_{max} (the *larger* eigenvalue)
  - Rotate the patch according to this angle

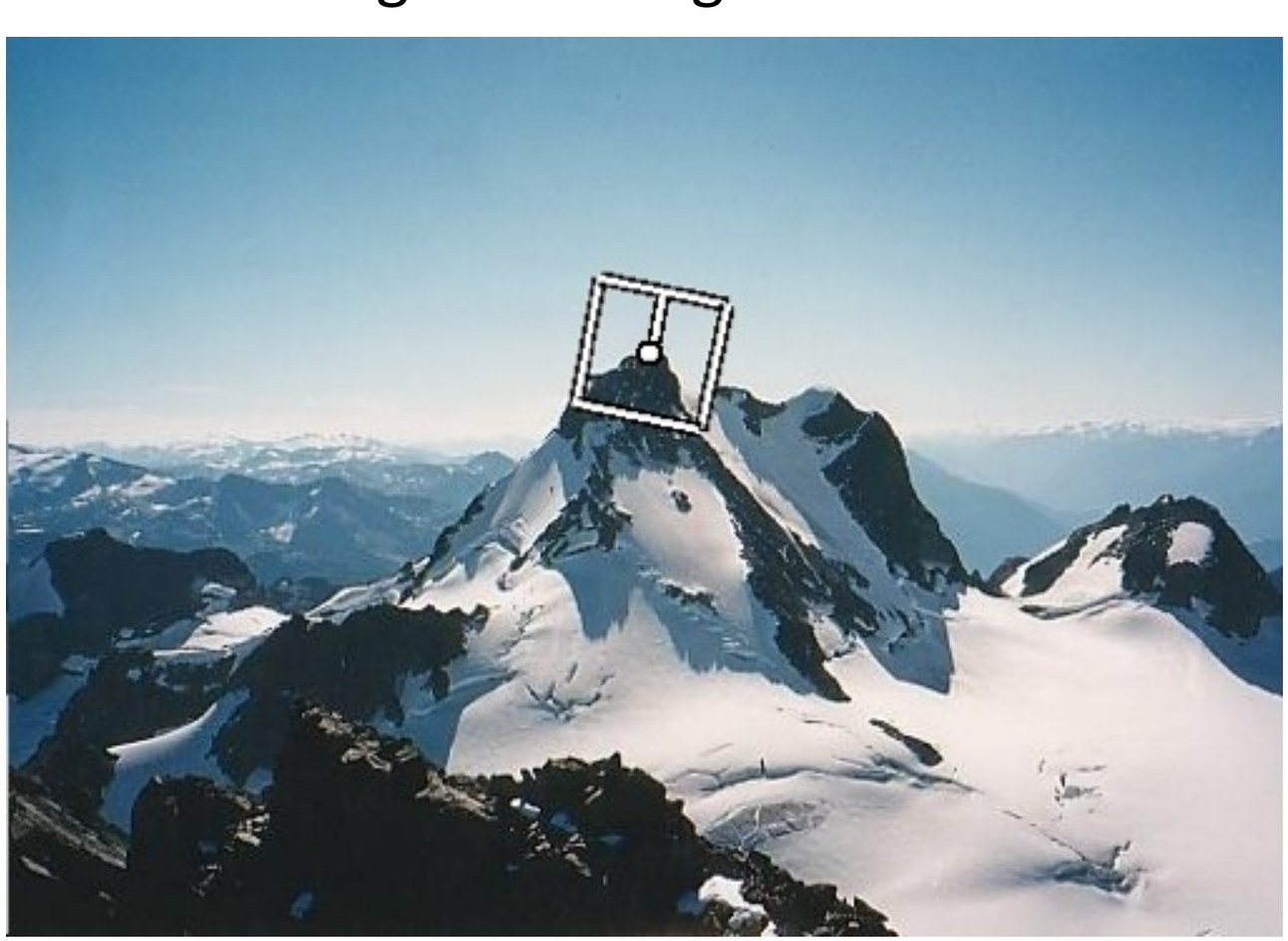
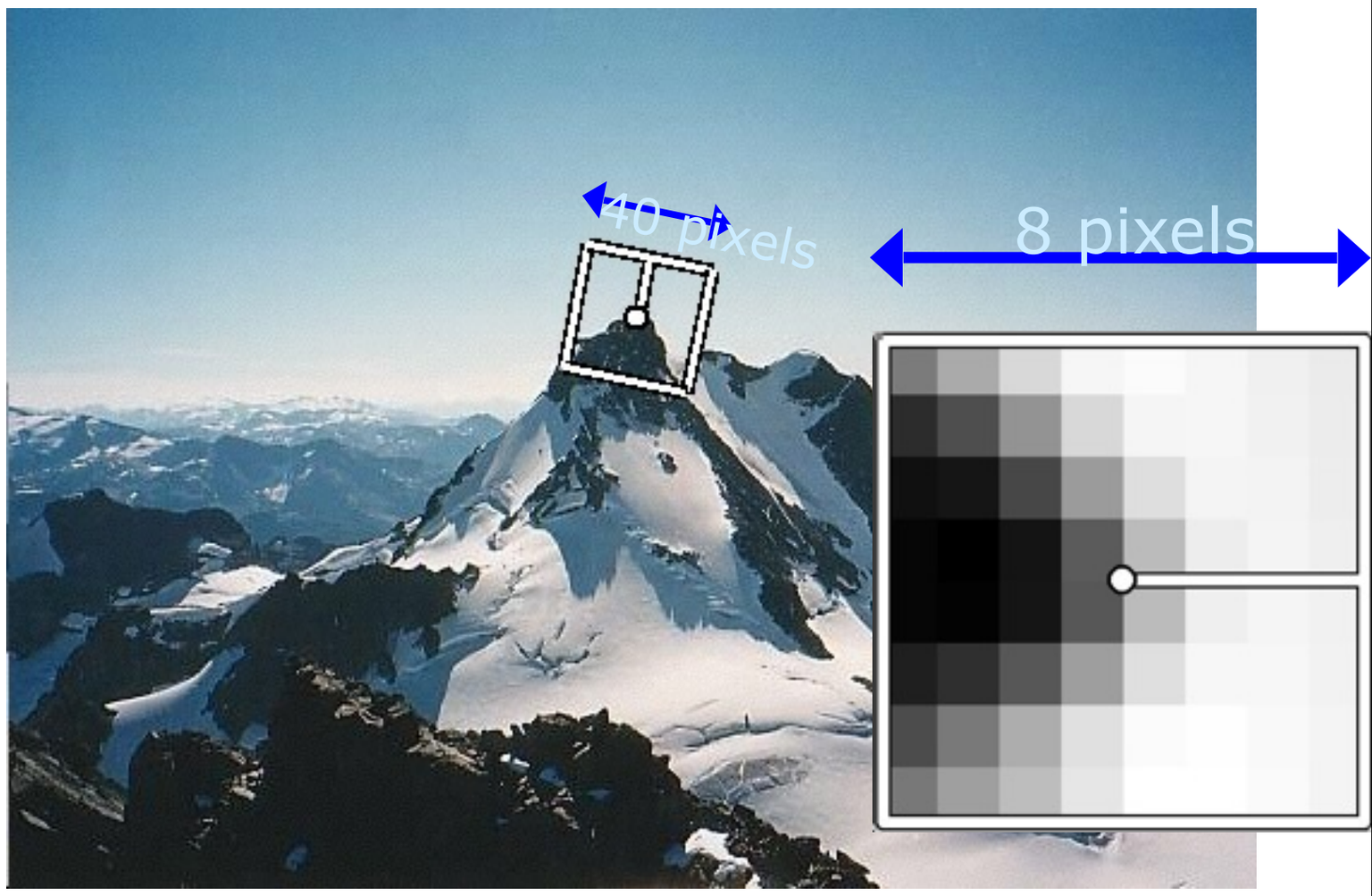


Figure by Matthew Brown

Multiscale Oriented PatcheS descriptor

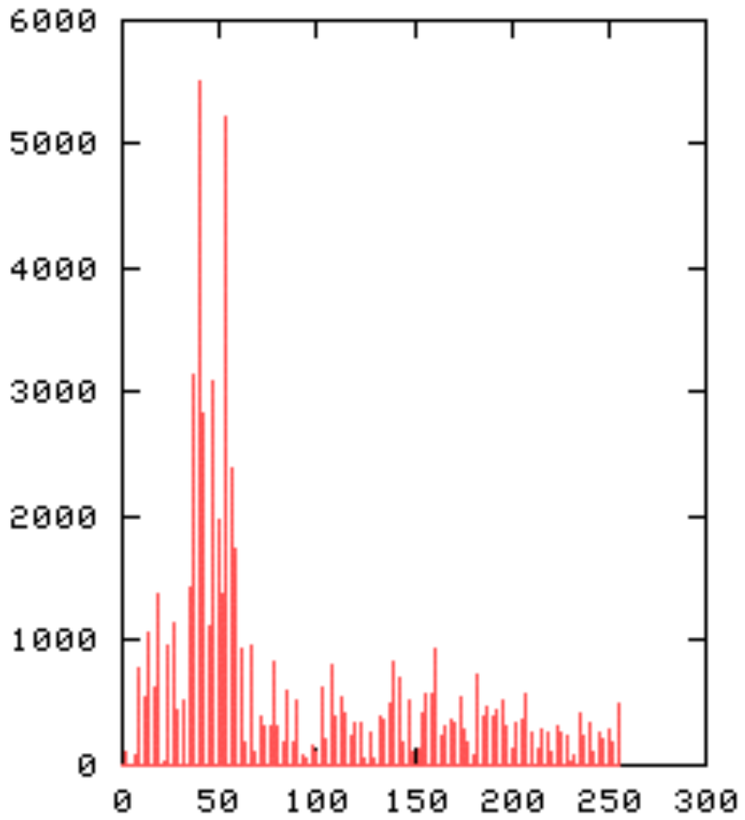
Take 40x40 square window around detected feature

- 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



You might use MOPS in your project

Image Representations: Histograms

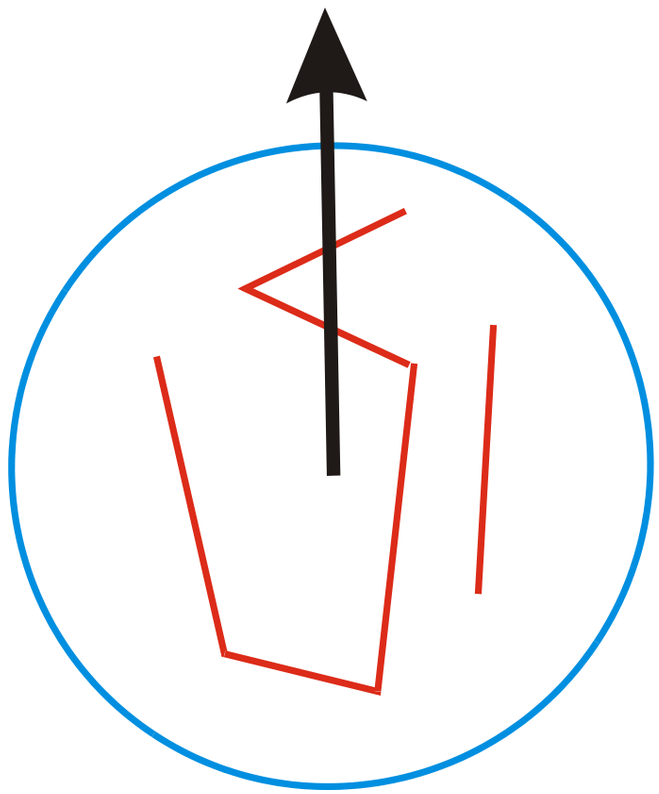


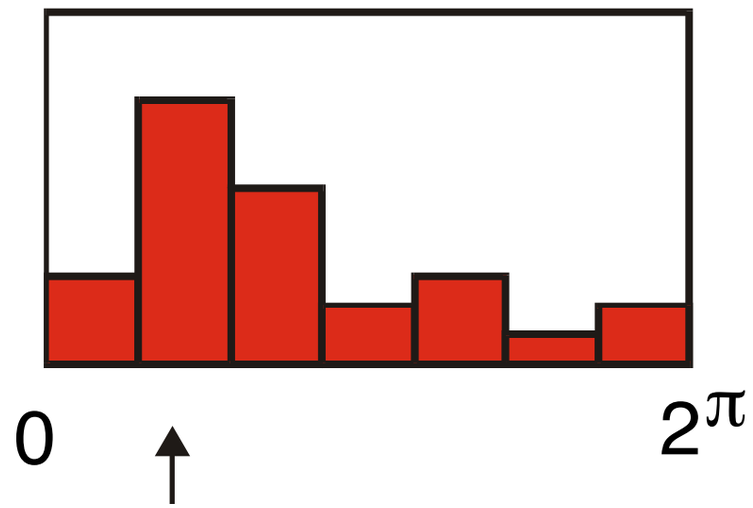
Global histogram

- Represent distribution of features
  - Color, texture, depth, …

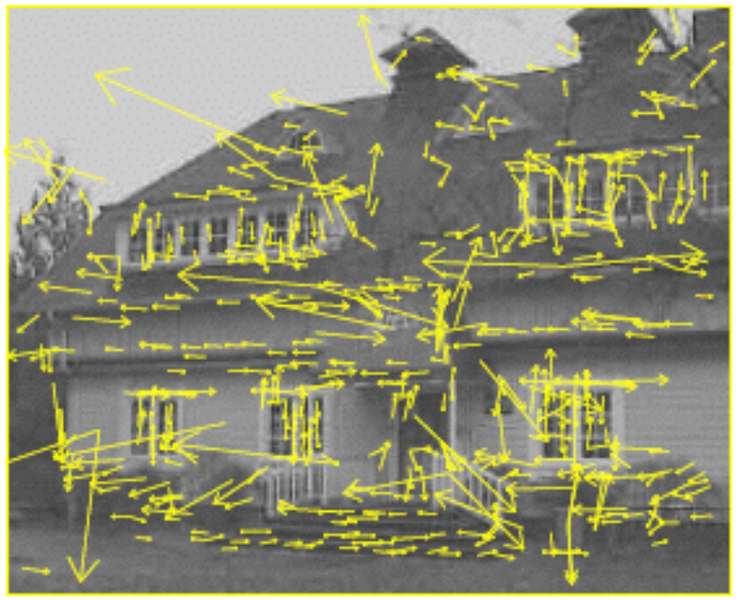
Orientation Normalization

- Compute orientation histogram [Lowe, SIFT, 1999]
- Select dominant orientation
- Normalize: rotate to fixed orientation





What kind of things do we compute histograms of?



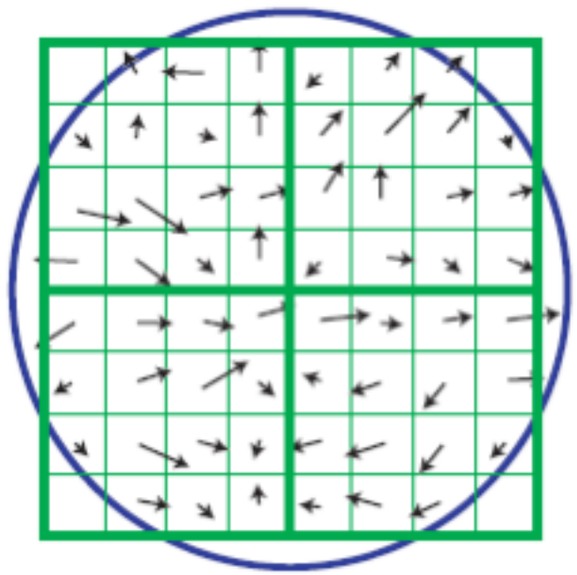
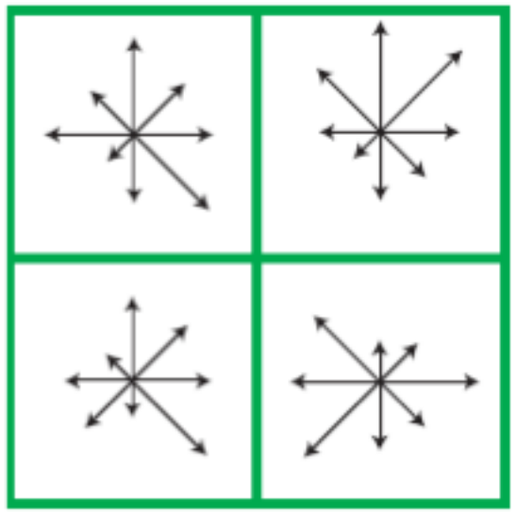


Image gradients



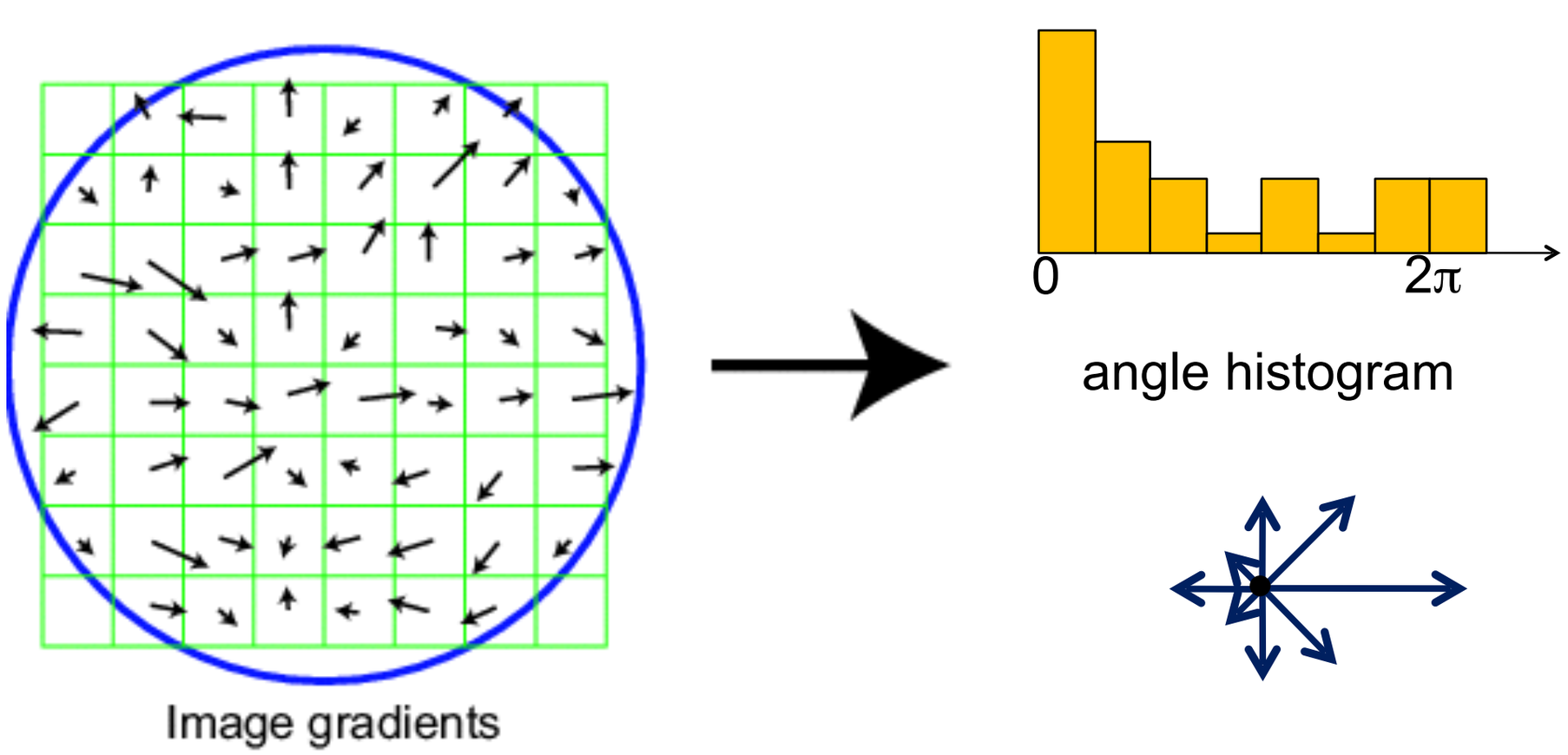
Keypoint descriptor

SIFT – Lowe IJCV 2004

Scale Invariant Feature Transform

Basic idea:

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



Adapted from slide by David Lowe

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 1.5 times the window (for smooth falloff)

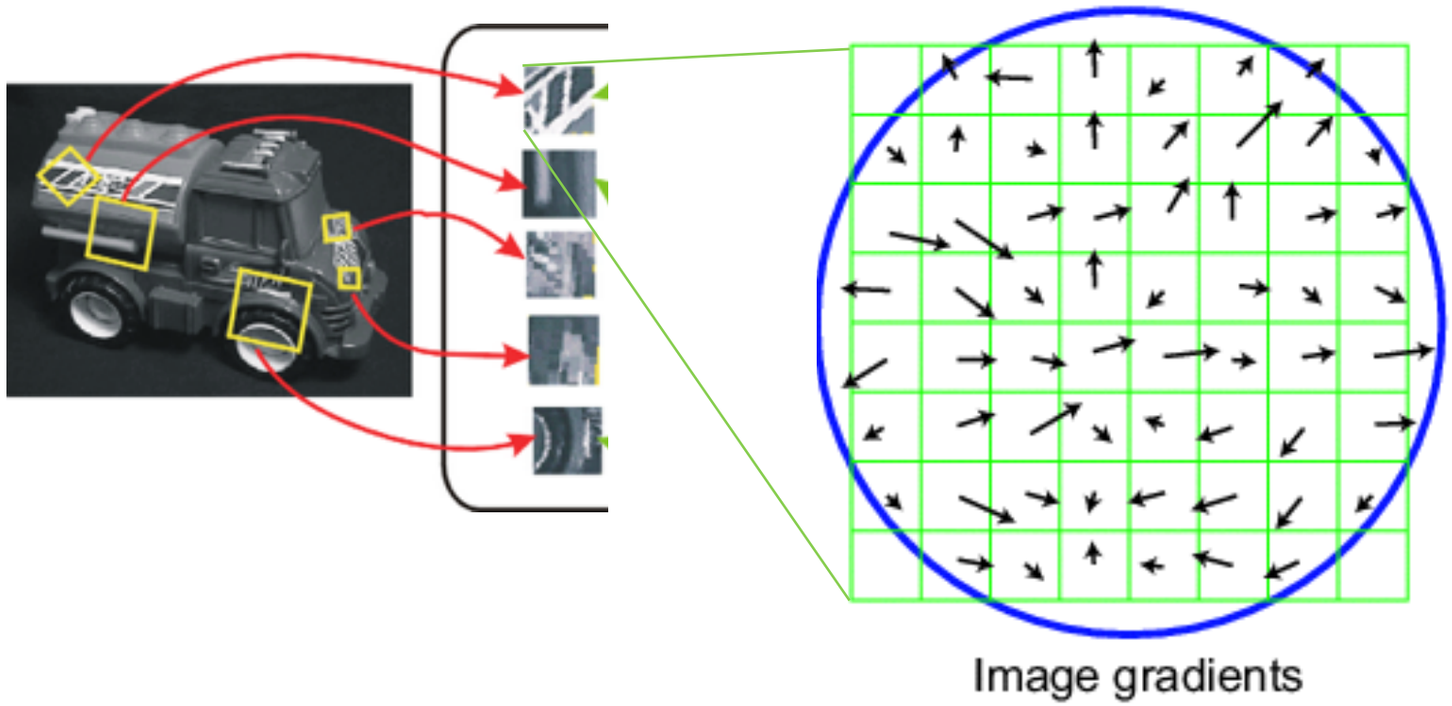
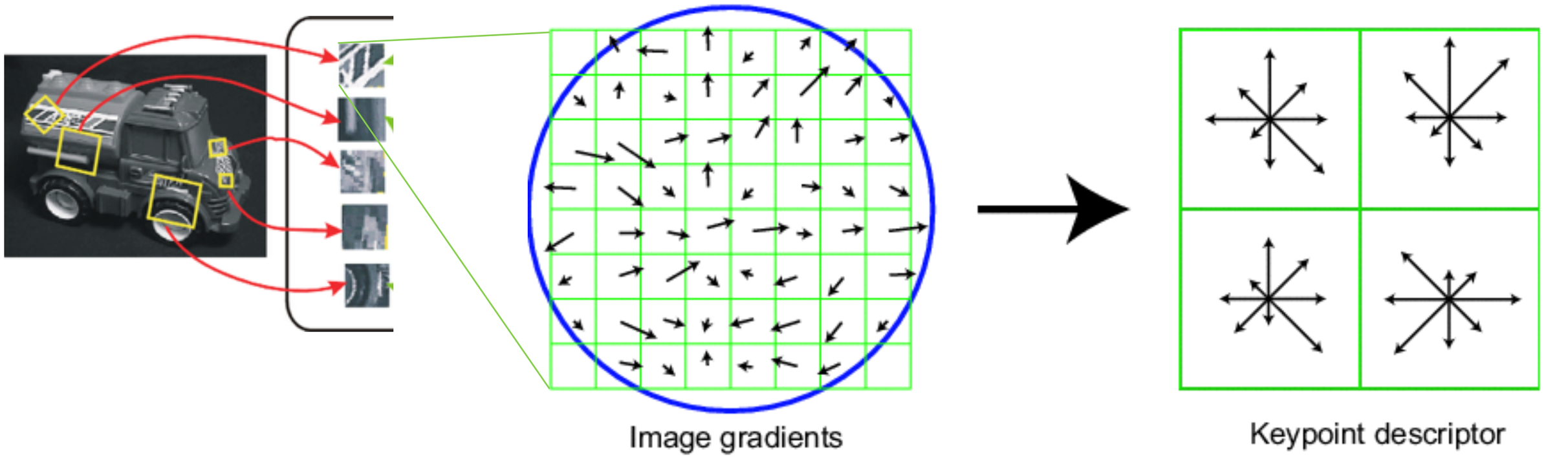


Image gradients

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

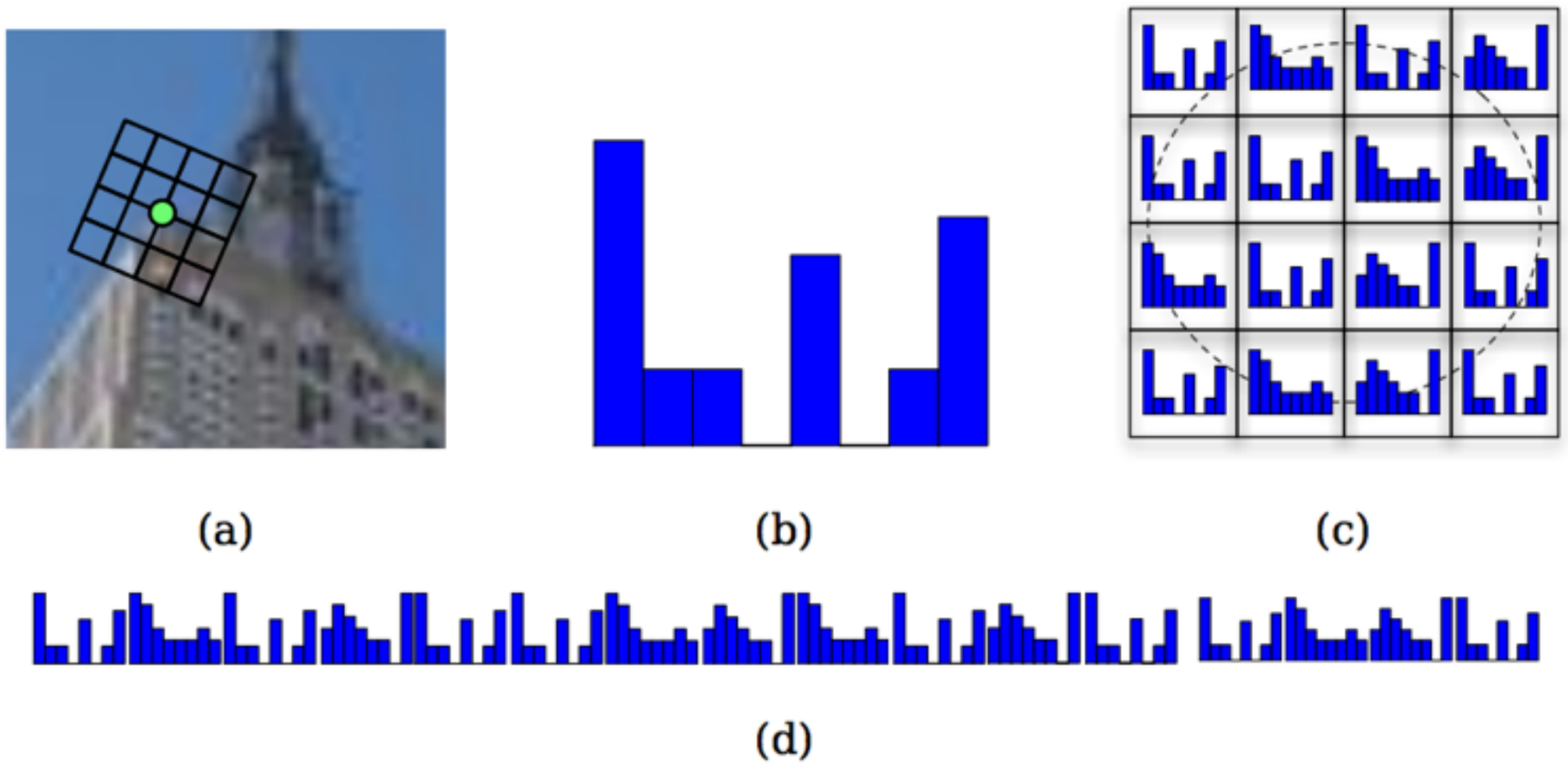
Orientation Assignment

- To achieve rotation invariance
- Compute central derivatives, gradient magnitude and direction of *L* (smooth image) at the scale of key point (x,y)

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

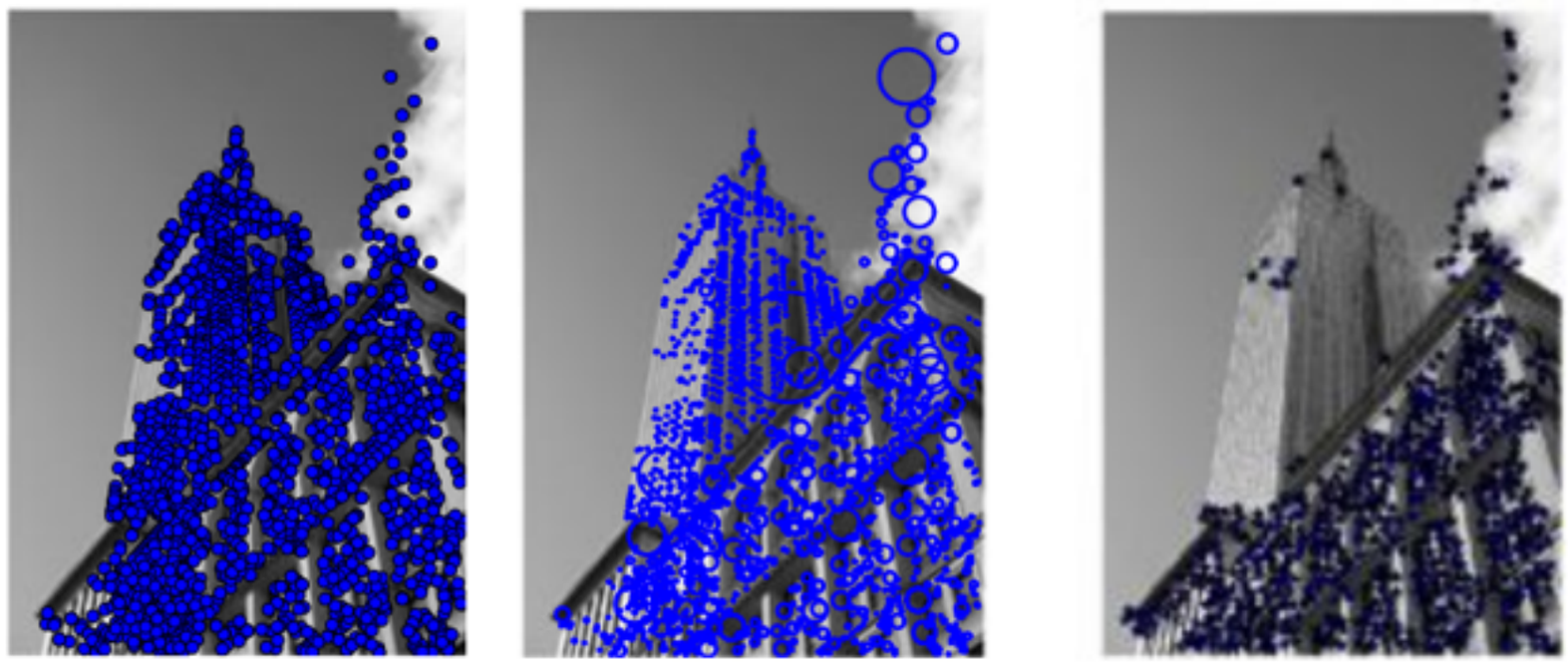
$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Construction of SIFT vector



http://programmingcomputervision.com/downloads/ProgrammingComputerVision_CCdraft.pdf

SIFT features extracted



(a) (b) (c)

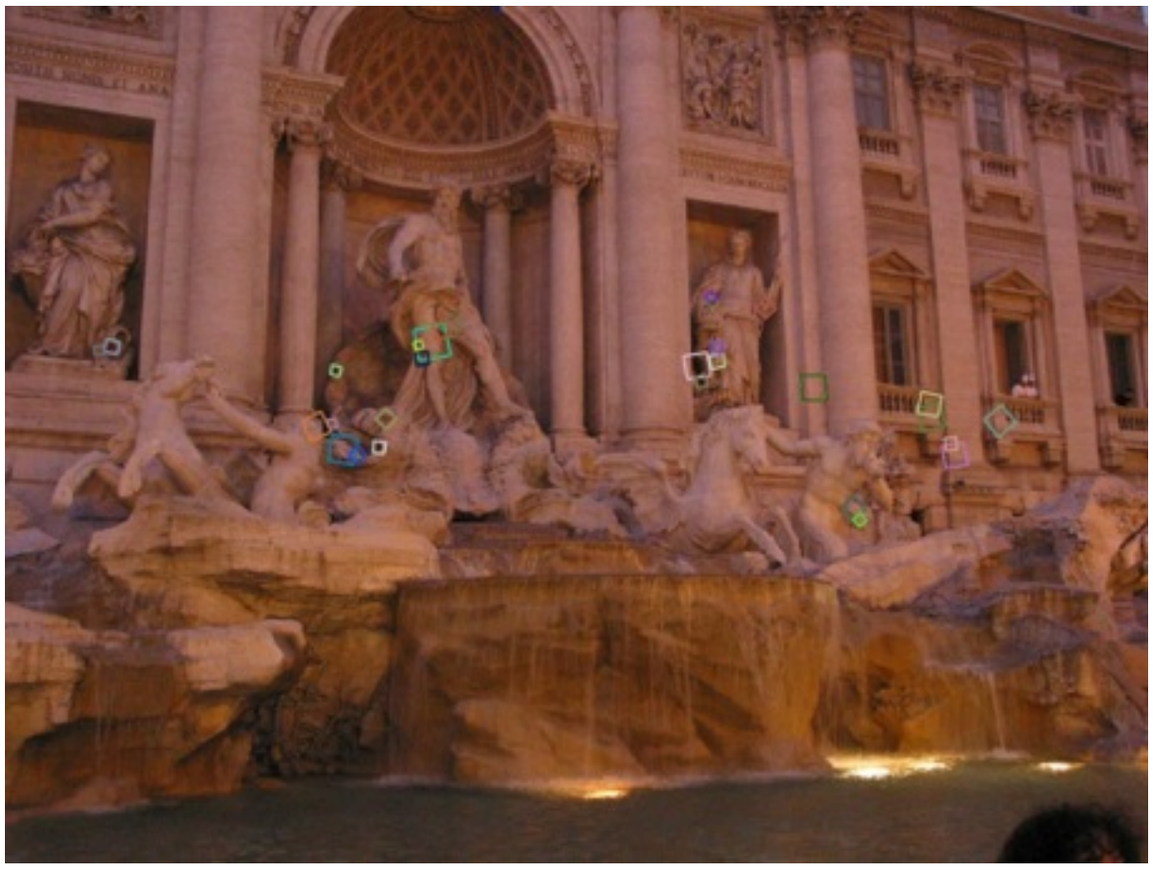
Figure 2.4: An example of extracting SIFT features for an image. (a) SIFT features (b) SIFT features shown with circle indicating the scale of the feature (c) Harris points for the same image for comparison.

http://programmingcomputervision.com/downloads/ProgrammingComputerVision_CCdraft.pdf

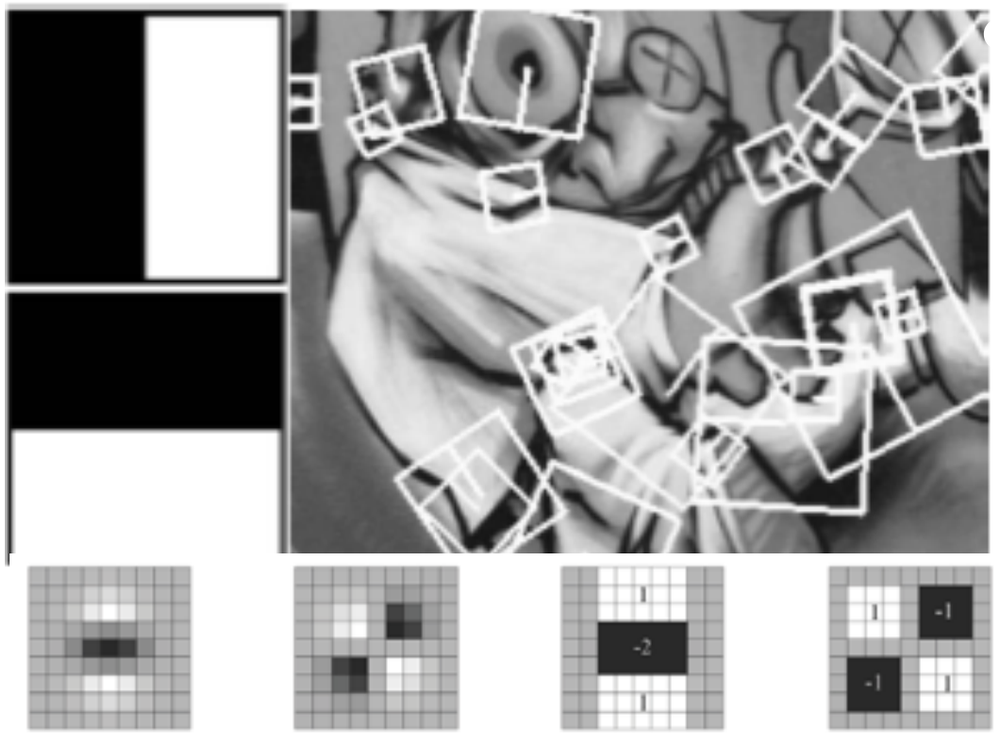
Properties of SIFT

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
- Python Implementation is here: http://www.maths.lth.se/matematiklth/personal/solem/downloads/sift.py
- MATLAB implementation: http://www.robots.ox.ac.uk/~vedaldi/code/sift.html



Local Descriptors: SURF



Fast approximation of SIFT idea

Efficient computation (Haar wavelets)

⇒ 6 times faster than SIFT

Equivalent quality for object identification

GPU implementation available

Feature extraction @ 200Hz
(detector + descriptor, 640×480 img)

http://www.vision.ee.ethz.ch/~surf

[Bay, ECCV'06], [Cornelis, CVGPU'08]

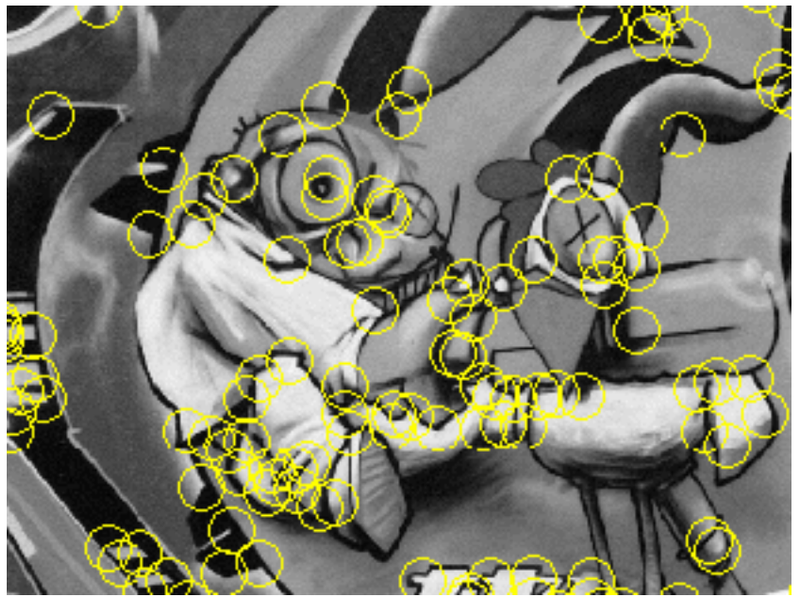
Other descriptors

- HOG: Histogram of Gradients (HOG)
  - Dalal/Triggs
  - Sliding window, pedestrian detection

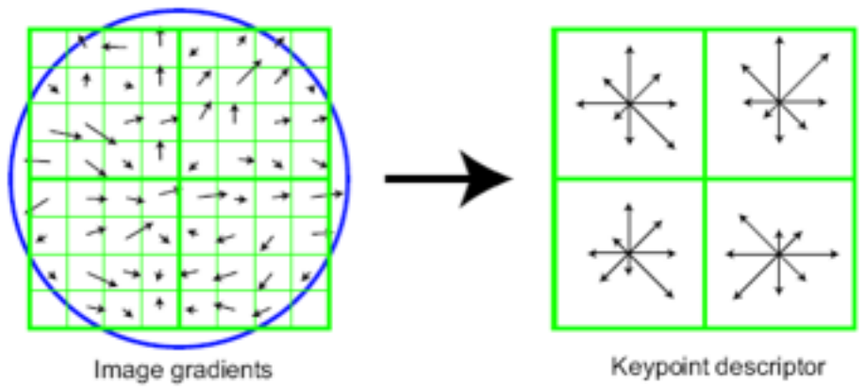
- FREAK: Fast Retina Keypoint
  - Perceptually motivated

Summary

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

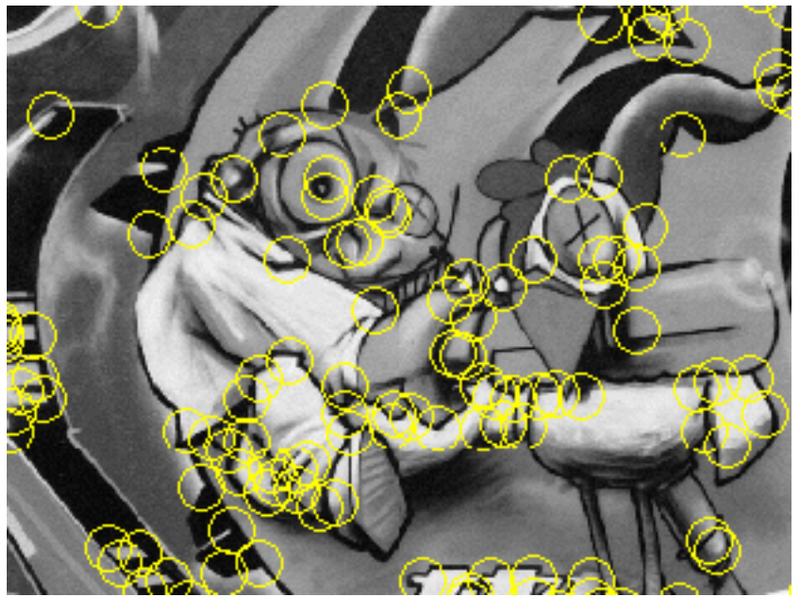


- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT and variants are typically good for stitching and recognition
  - But, need not stick to one

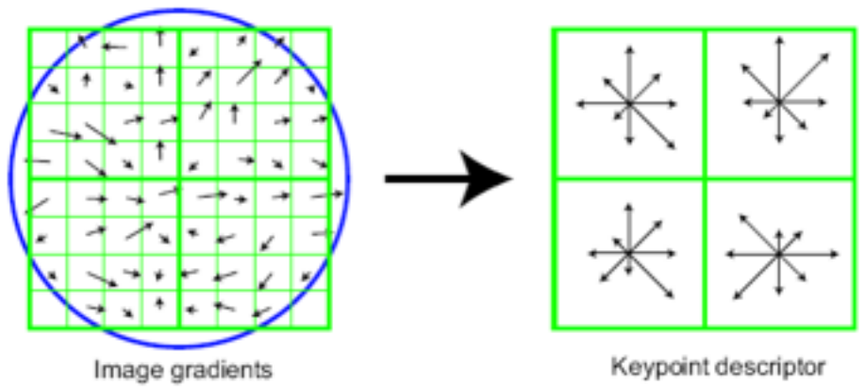


Things to remember

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



- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT



Learn more about SIFT

- http://en.wikipedia.org/wiki/Scale-invariant_feature_transform

- Original paper (cited 29089 times):

  https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

- Chapter 2. Solem, Python implementation

- Point Feature Data set:

  http://roboimagedata.compute.dtu.dk/?page_id=24