Local invariant features

Tuesday, Oct 28

Kristen Grauman
UT-Austin

Today

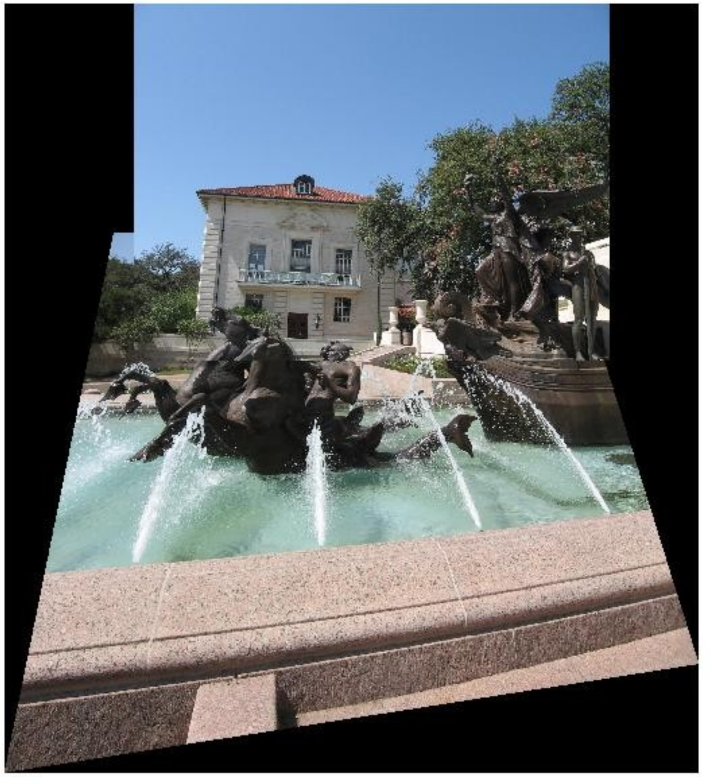
- Some more Pset 2 results
- Pset 2 returned, pick up solutions
- Pset 3 is posted, due 11/11

- Local invariant features
  - Detection of interest points
    - Harris corner detection
    - Scale invariant detection: LoG / DoG
  - Description of local patches
    - SIFT : Histograms of oriented gradients

Local features and alignment

im1

im2



mosaic

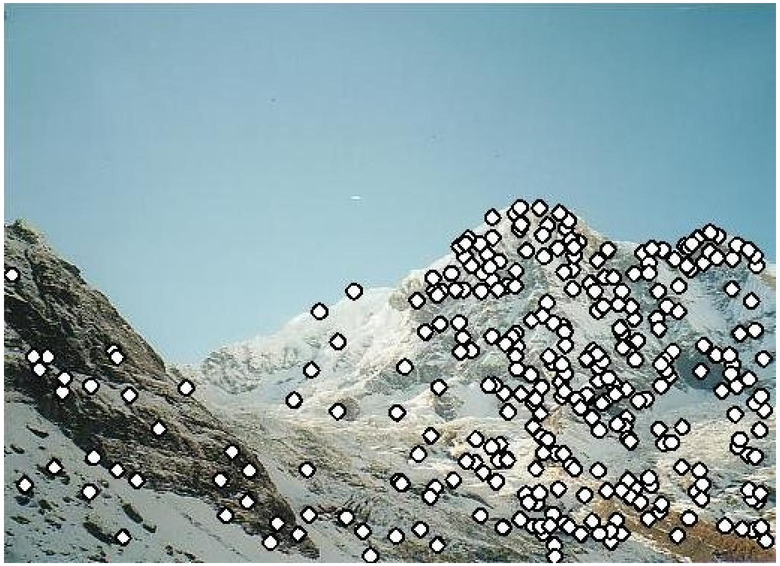
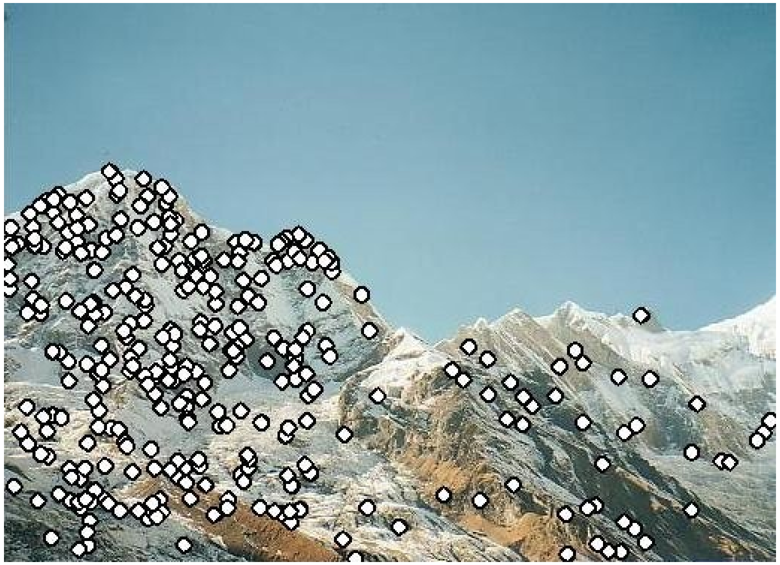
To compute the homography, we needed pairs of **corresponding points** in the images.

Local features and alignment


[Slide credit: Darya Frolova and Denis Simakov]

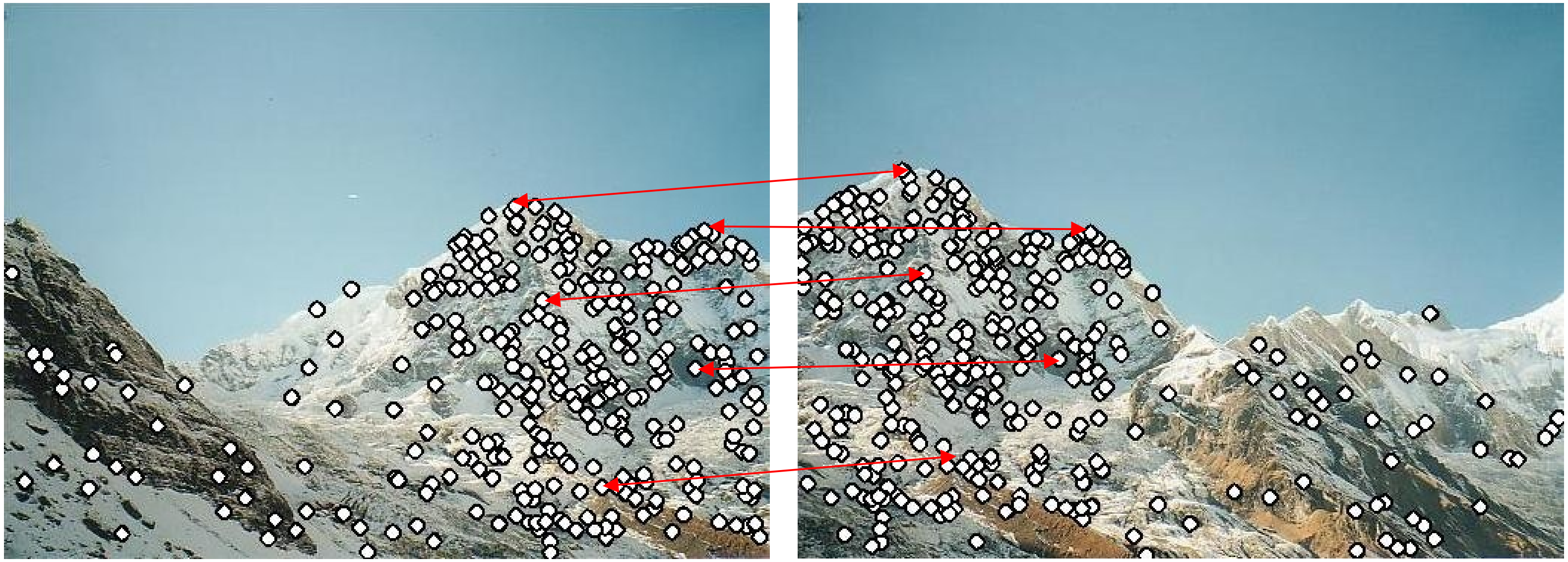
Local features and alignment

- Detect feature points in both images

Local features and alignment

- Detect feature points in both images
- Find corresponding pairs

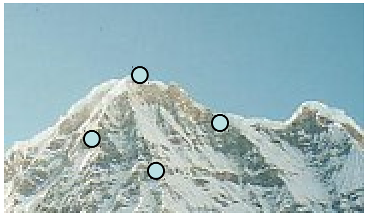


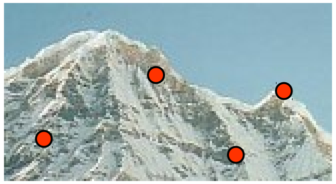
Local features and alignment

- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

Local features and alignment

- Problem 1:
  – Detect the *same* point *independently* in both images



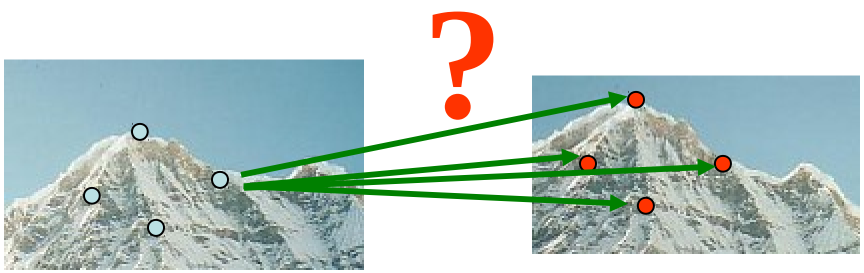


no chance to match!

We need a repeatable detector

Local features and alignment

- Problem 2:
  - For each point correctly recognize the corresponding one



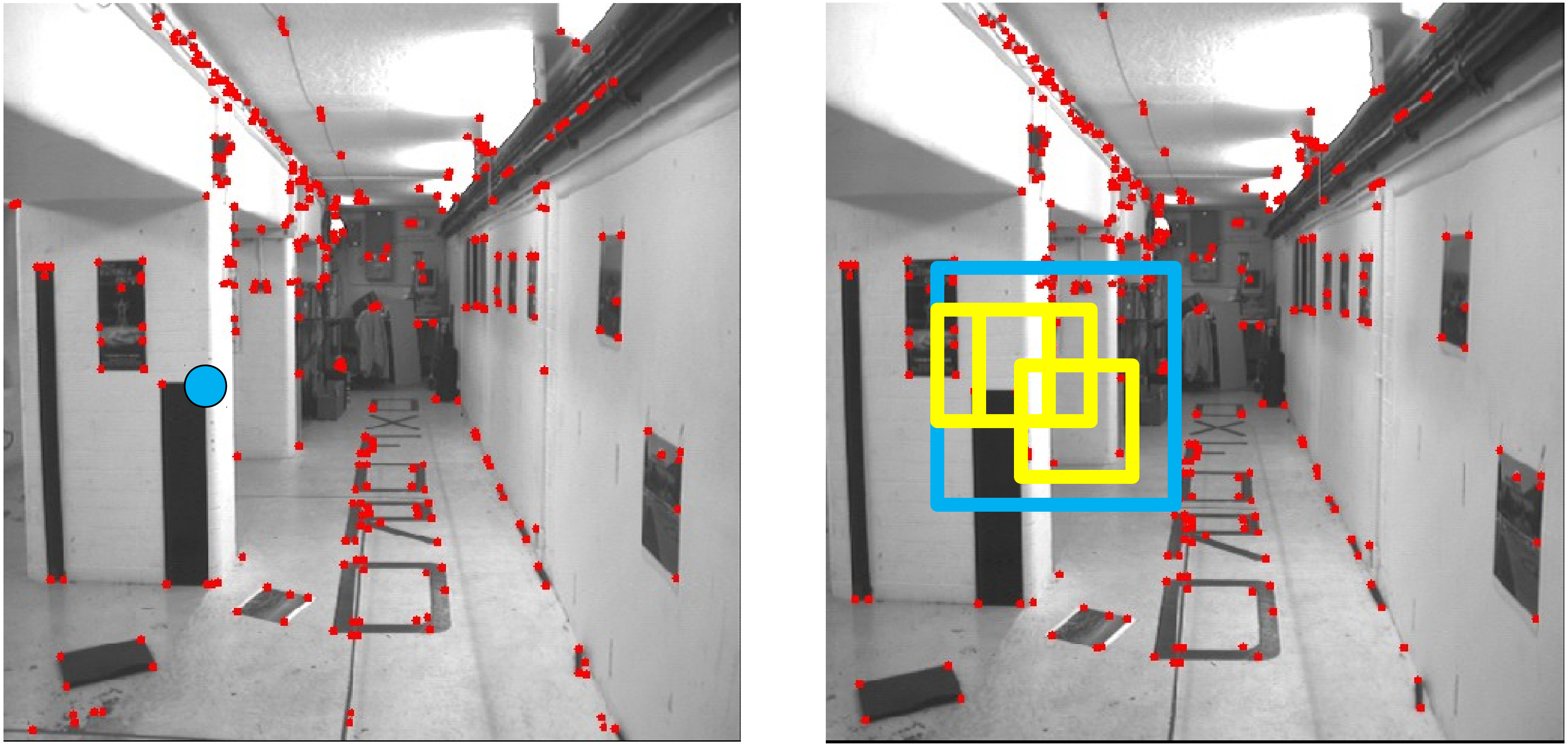
We need a reliable and distinctive descriptor

Local features and stereo matching

Similarly, the first step in our stereo pipeline using weak calibration was to find interest points,…

Local features and stereo matching

… and we let the surrounding pixels in a neighborhood patch serve as the local descriptor, which we can compare with correlation



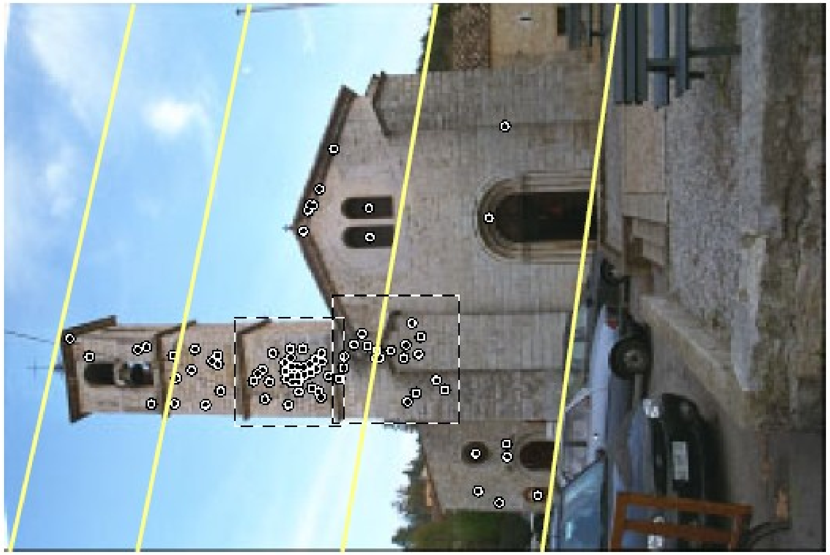
We want a sparse set of reliably detectable interest points.

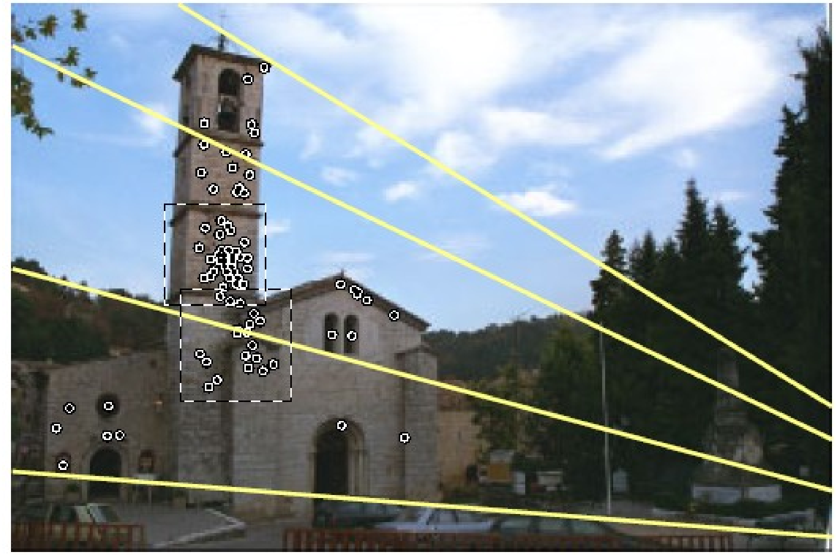
Local features and stereo matching

… and we let the surrounding pixels in a neighborhood patch serve as the local descriptor, which we can compare with correlation

Putative matches

Local features and stereo matching

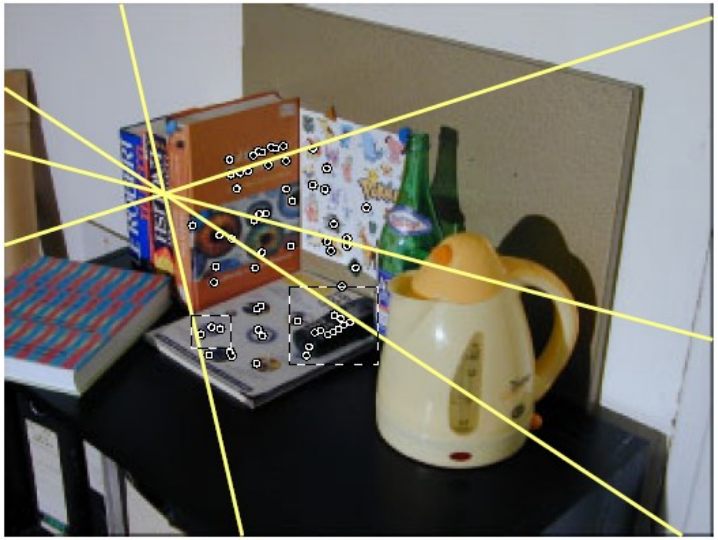


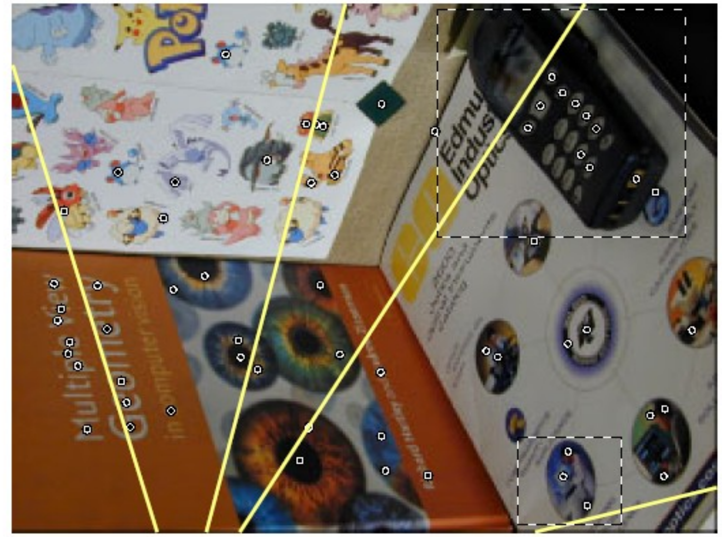


- Patches of intensity have limited robustness for matching across different views
- Consider the case where we have a **wide baseline** separating the two views

J. Matas, O. Chum, M. Urban, T. Pajdla. Robust Wide Baseline Stereo From Maximally Stable Extremal Regions, BMVC 2002.

Local features and stereo matching





- Patches of intensity have limited robustness for matching across different views
- Consider the case where we have a **wide baseline** separating the two views

J. Matas, O. Chum, M. Urban, T. Pajdla. Robust Wide Baseline Stereo From Maximally Stable Extremal Regions, BMVC 2002.

- What would we like our local features to be invariant to?

Geometric transformations

Photometric transformations

Figure from T. Tuytelaars ECCV 2006 tutorial

And other nuisances…

- Noise
- Blur
- Compression artifacts
- Appearance variation for a category

Invariant local features

Subset of local feature types designed to be invariant to common geometric and photometric transformations.

Basic steps:

1) Detect distinctive interest points
2) Extract invariant descriptors

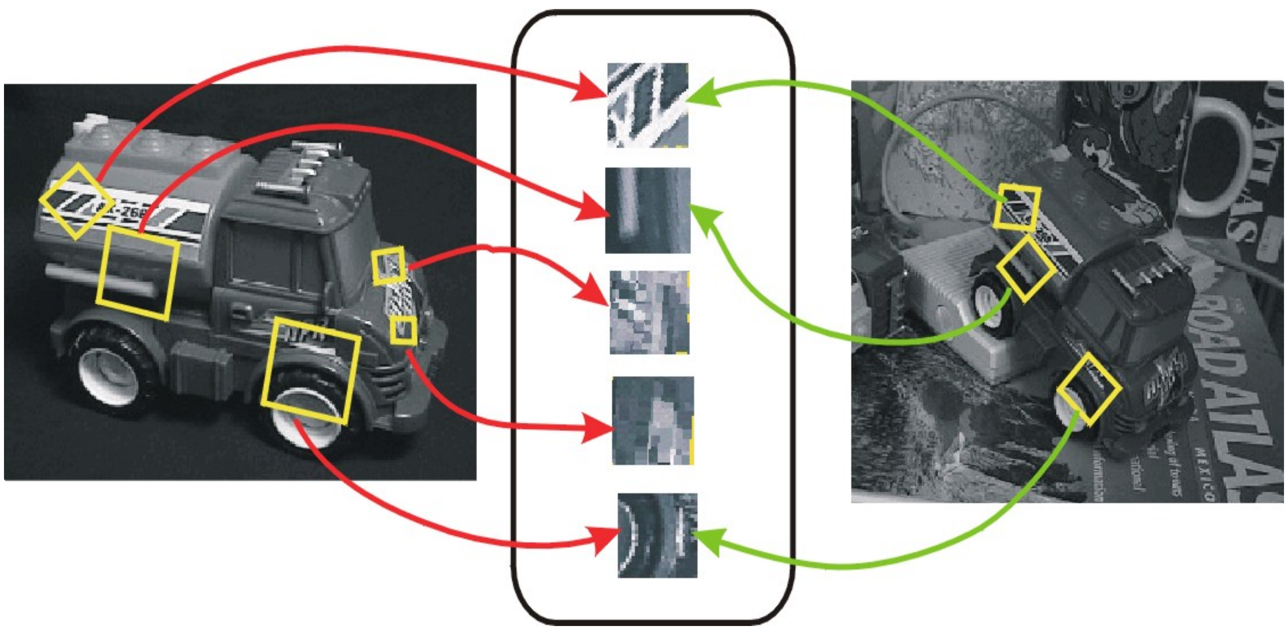
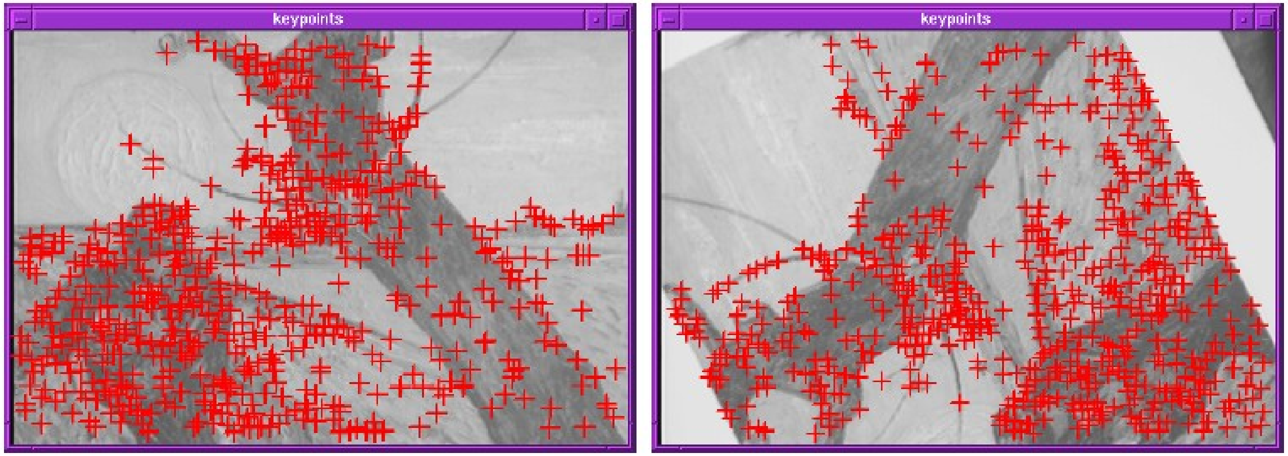


Figure: David Lowe

Main questions

- Where will the interest points come from?
  - What are salient features that we'll *detect* in multiple views?
- How to *describe* a local region?
- How to establish *correspondences*, i.e., compute matches?

Finding Corners



Key property: in the region around a corner, image gradient has two or more dominant directions

Corners are repeatable and **distinctive**

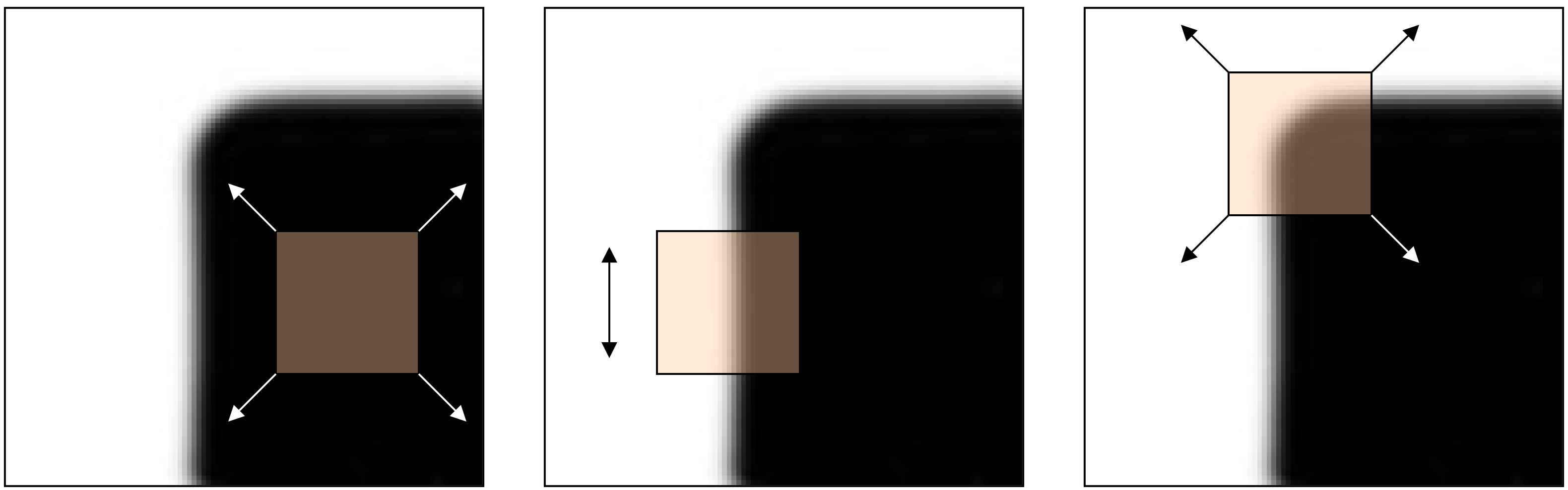
C.Harris and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*: pages 147--151.

Source: Lana Lazebnik

Corners as distinctive interest points

We should easily recognize the point by looking through a small window

Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change
in all

"edge":
no change
along the
edge direction

"corner":
significant
change in all
directions

Source: A. Efros

Harris Detector formulation

Change of intensity for the shift [*u,v*]:

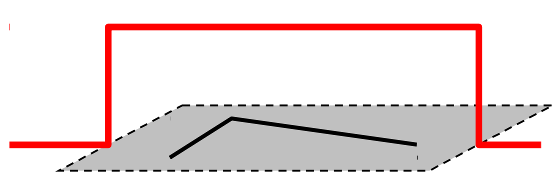
$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

Window function

Shifted intensity

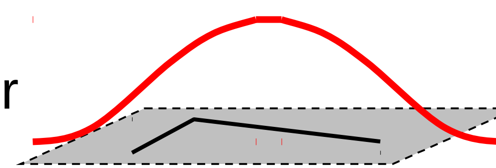
Intensity

Window function $w(x,y) =$



1 in window, 0 outside

or



Gaussian

Source: R. Szeliski

Harris Detector formulation

This measure of change can be approximated by:

$$E(u,v) \approx [u\ v]\ M \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a 2×2 matrix computed from image derivatives:

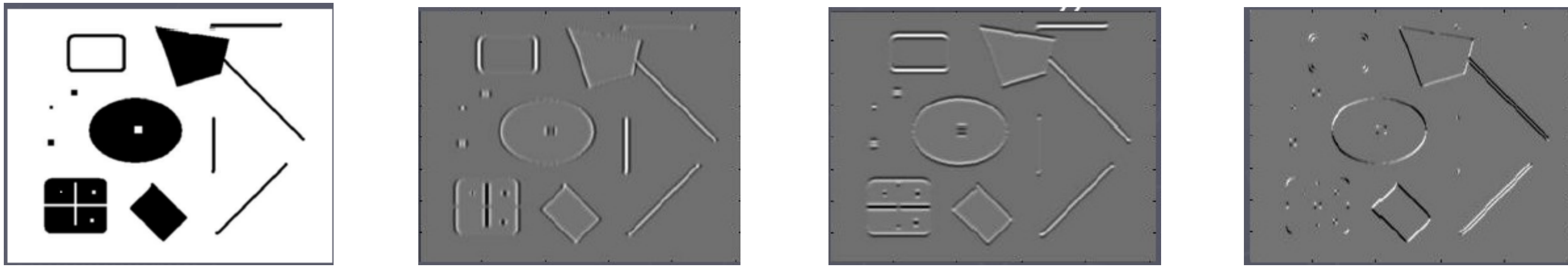
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Gradient with respect to x, times gradient with respect to y

Sum over image region – area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x\ I_y]$$

Harris Detector formulation



where *M* is a 2×2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

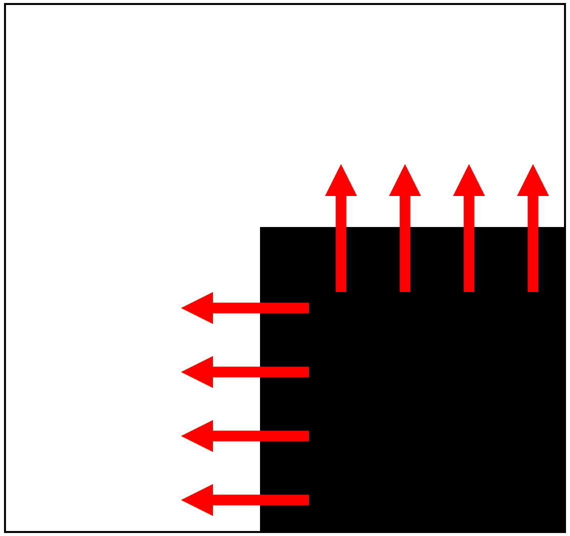
Gradient with respect to x, times gradient with respect to y

Sum over image region – area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y]$$

What does this matrix reveal?

First, consider an axis-aligned corner:



What does this matrix reveal?

First, consider an axis-aligned corner:

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means dominant gradient directions align with x or y axis

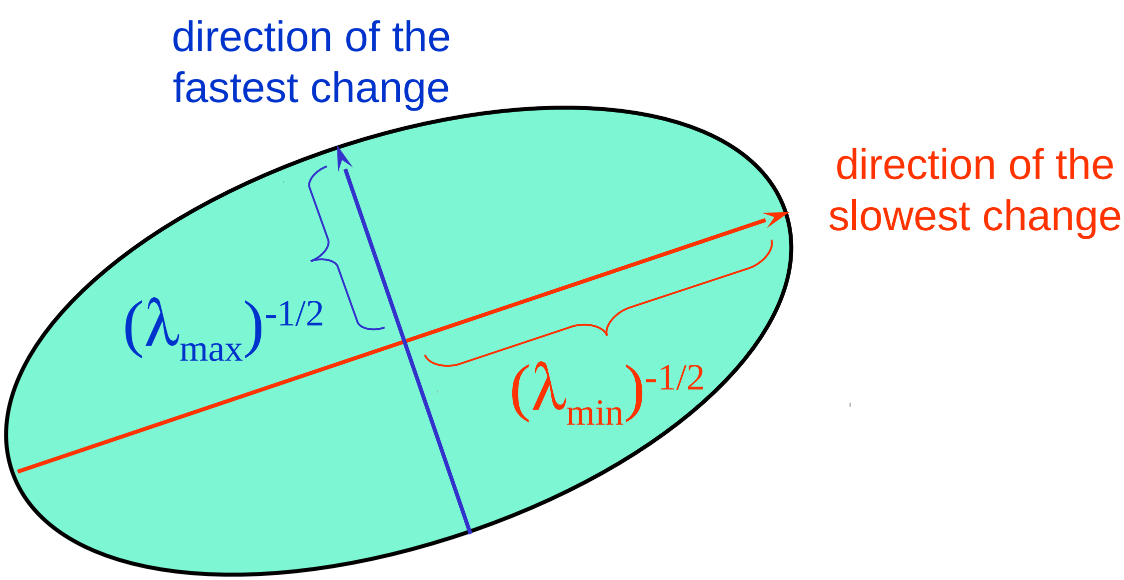
If either λ is close to 0, then this is **not** a corner, so look for locations where both are large.

What if we have a corner that is not aligned with the image axes?

General Case

Since M is symmetric, we have $M = R^{-1}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$

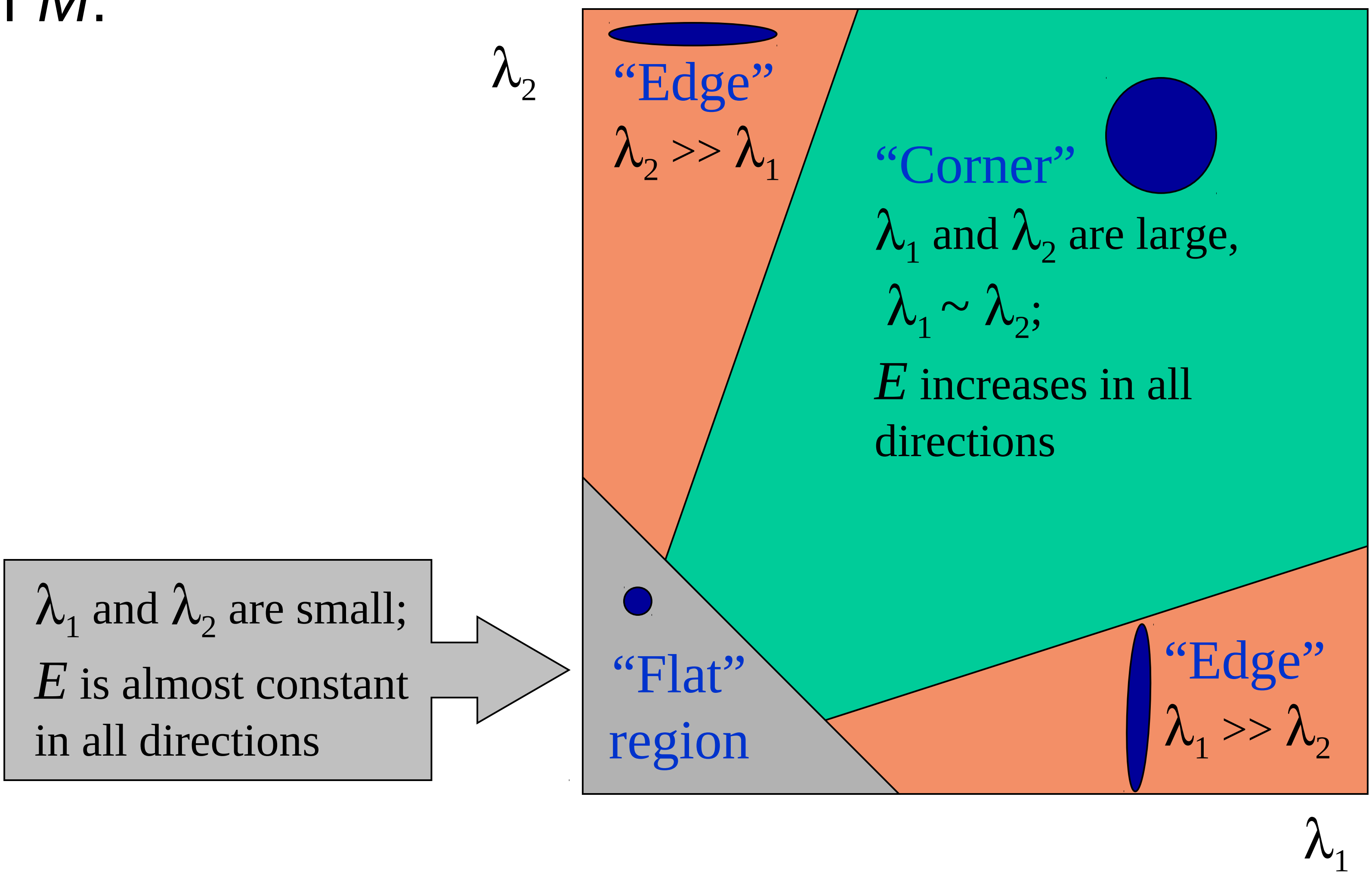
We can visualize *M* as an ellipse with axis lengths determined by the eigenvalues and orientation determined by *R*



Slide adapted form Darya Frolova, Denis Simakov.

Interpreting the eigenvalues

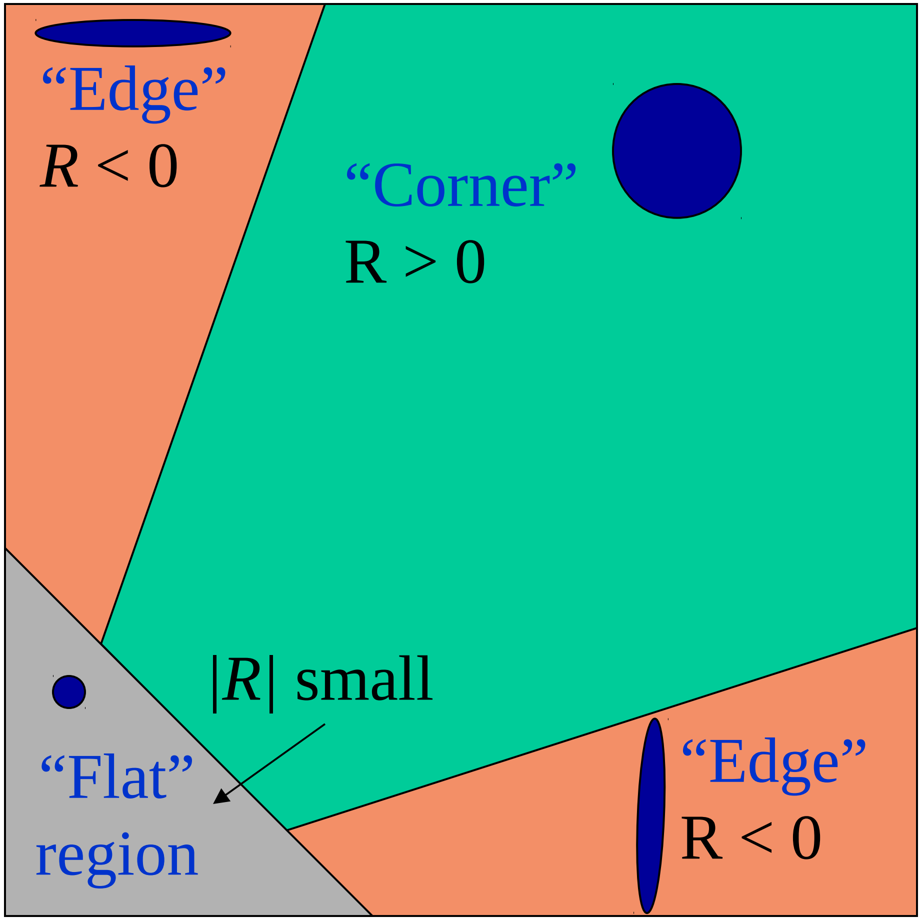
Classification of image points using eigenvalues of *M*:



Corner response function

$$R = \det(M) - \alpha \ \mathrm{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α: constant (0.04 to 0.06)



Harris Corner Detector

- Algorithm steps:
  - Compute M matrix within all image windows to get their R scores
  - Find points with large corner response

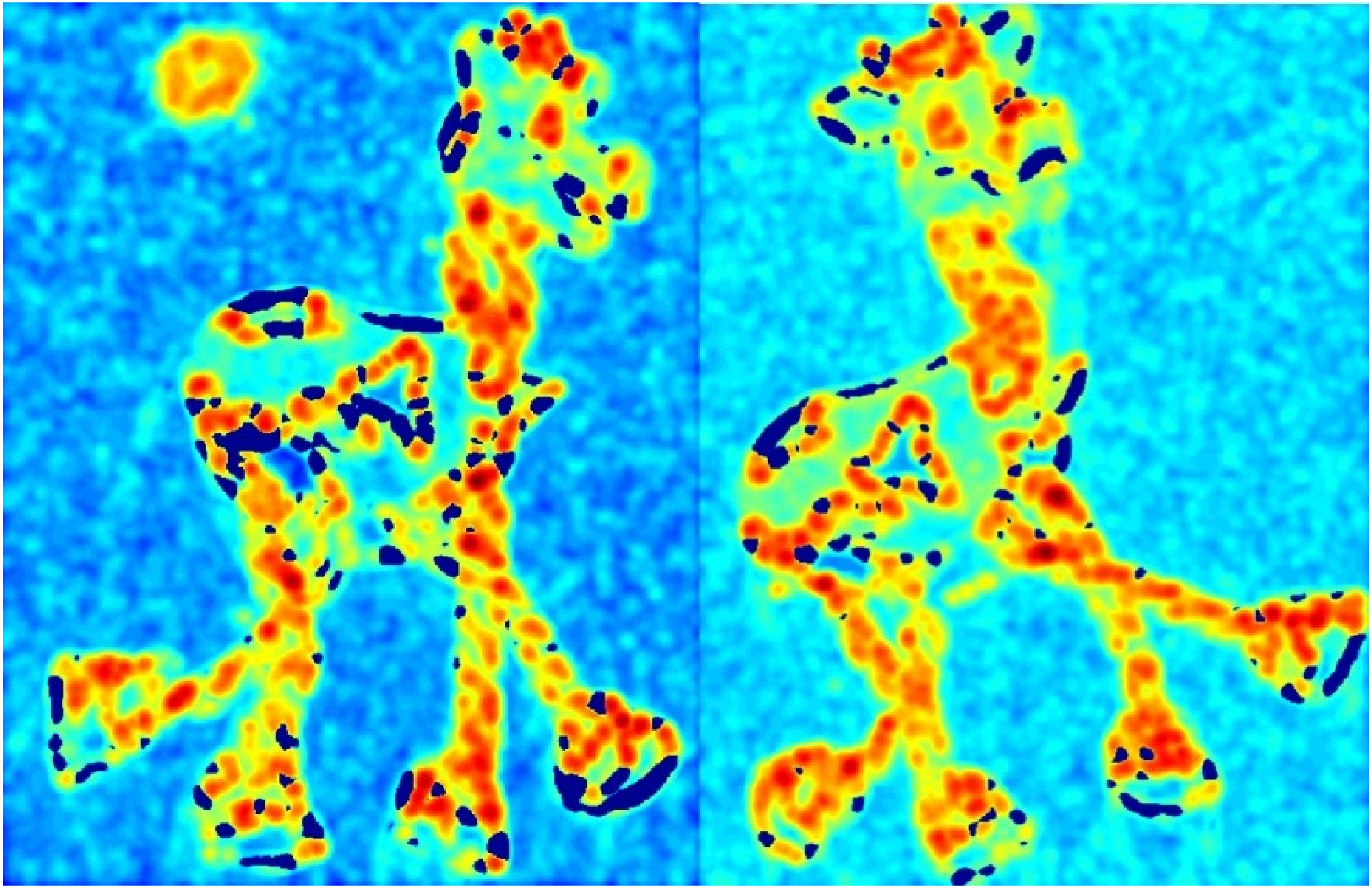
    (R > threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

Slide adapted form Darya Frolova, Denis Simakov, Weizmann Institute.

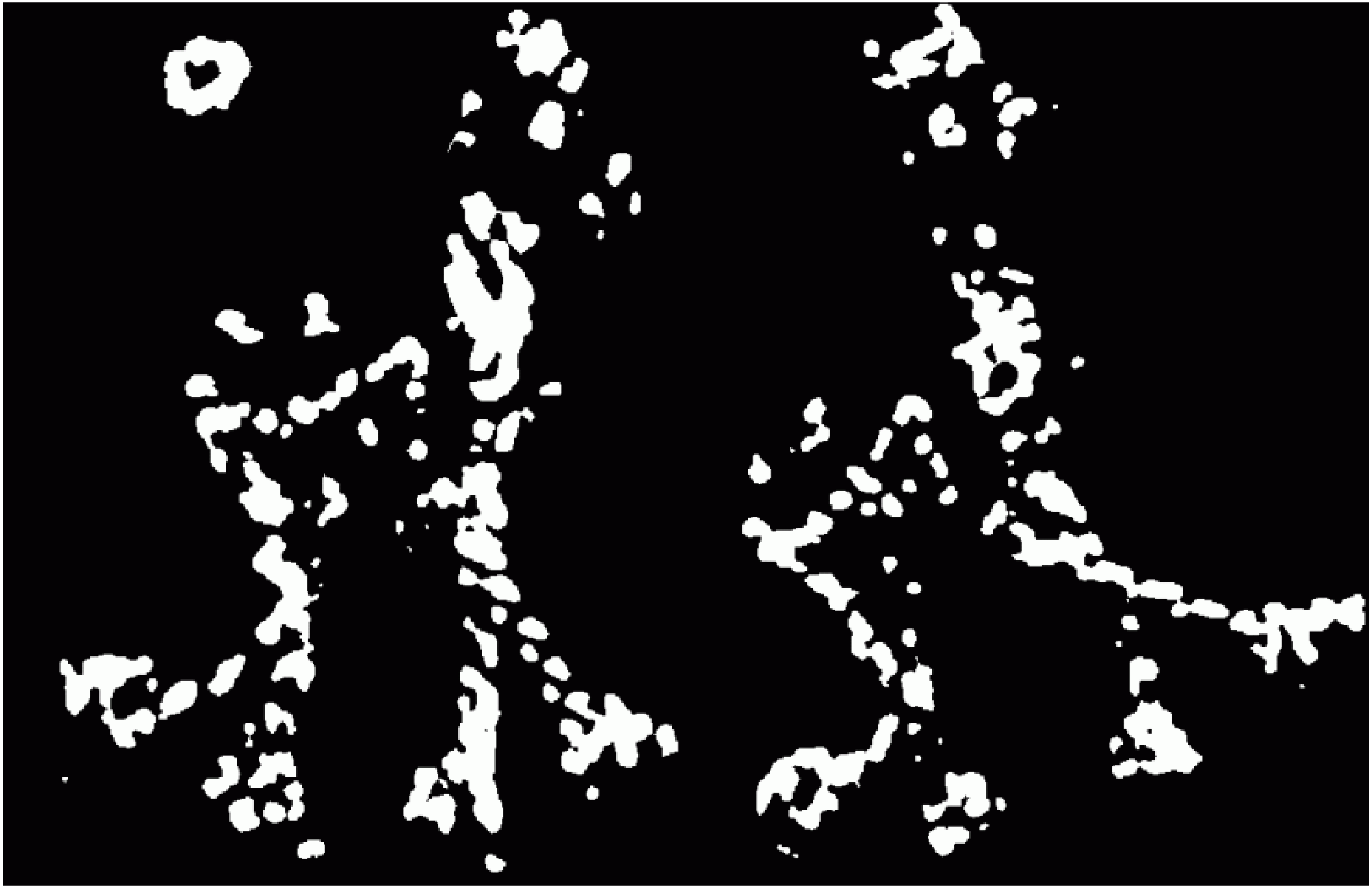
Harris Detector: Workflow

Compute corner response R



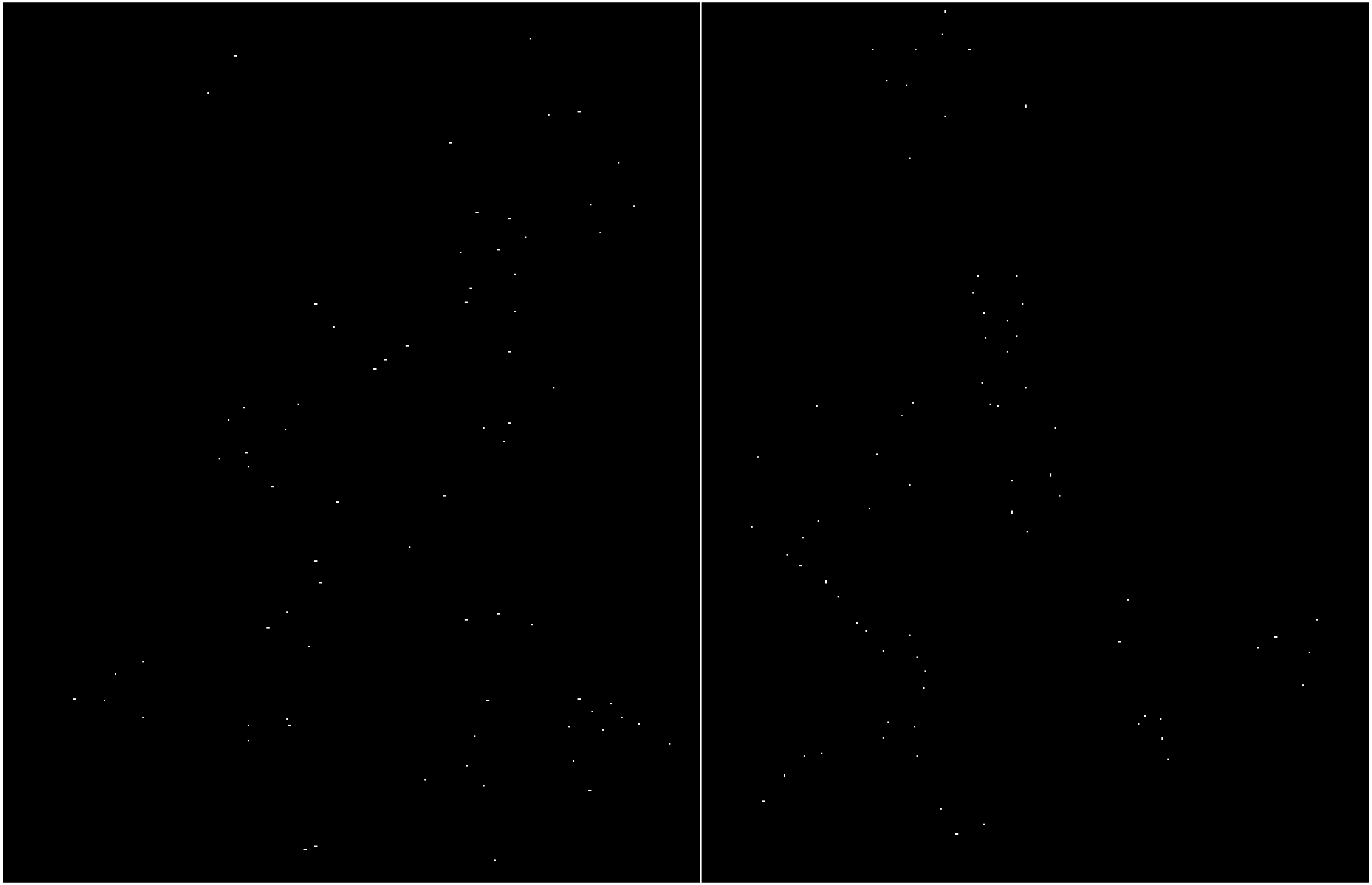
Harris Detector: Workflow

Find points with large corner response: R>threshold

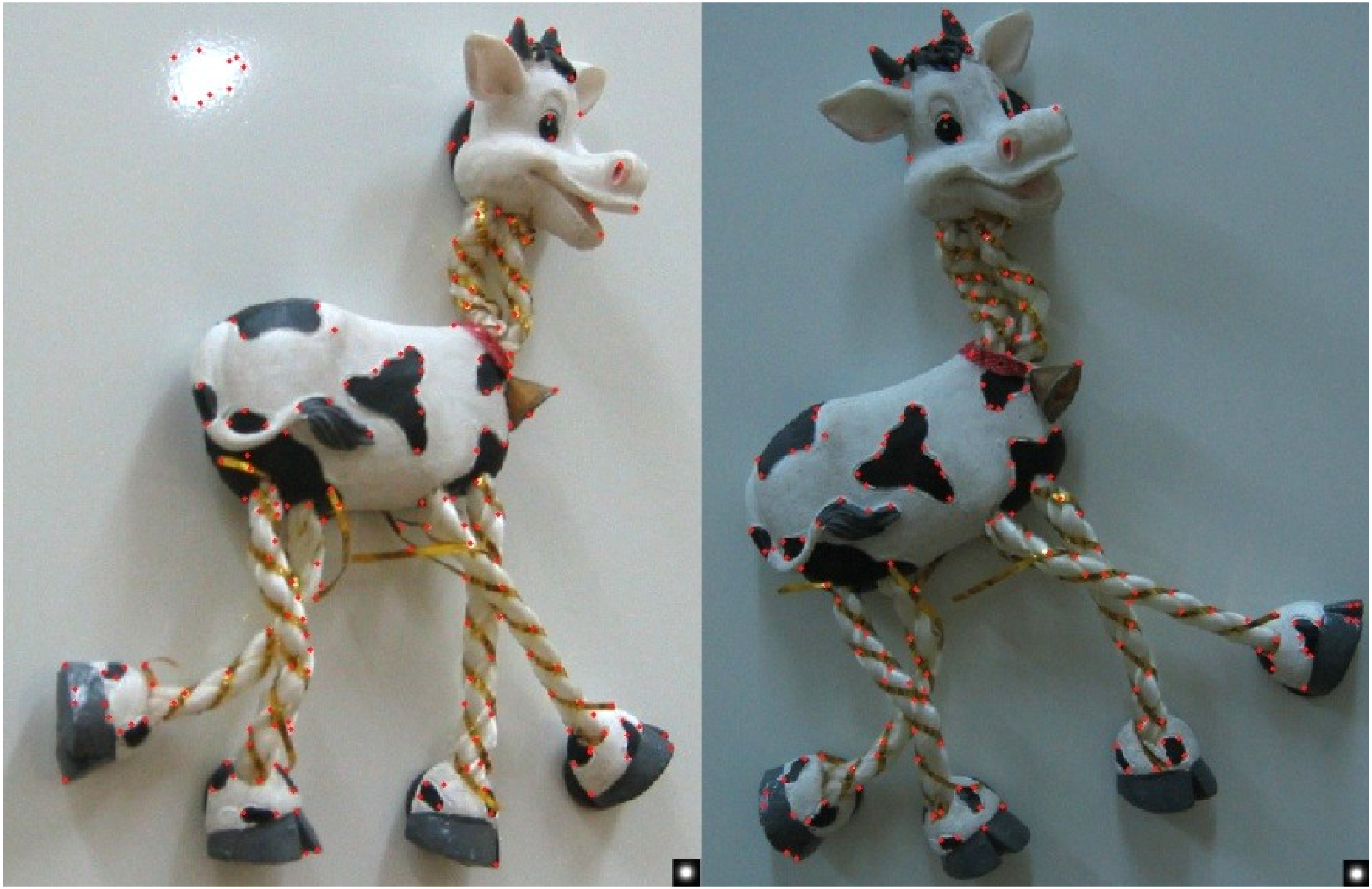


Harris Detector: Workflow

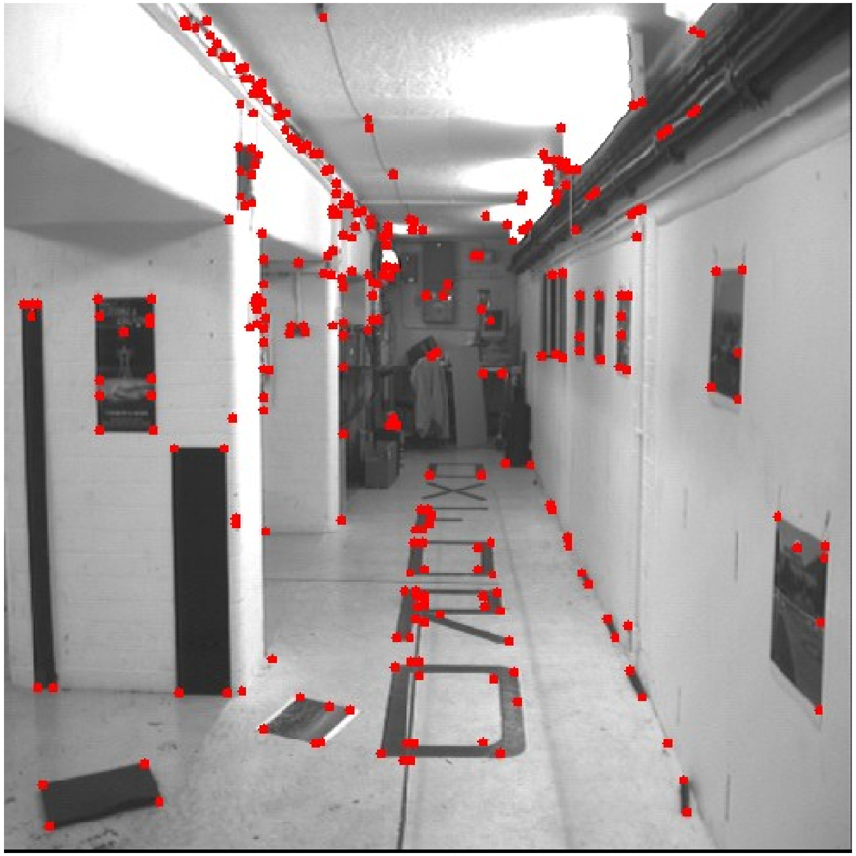
Take only the points of local maxima of R



Harris Detector: Workflow

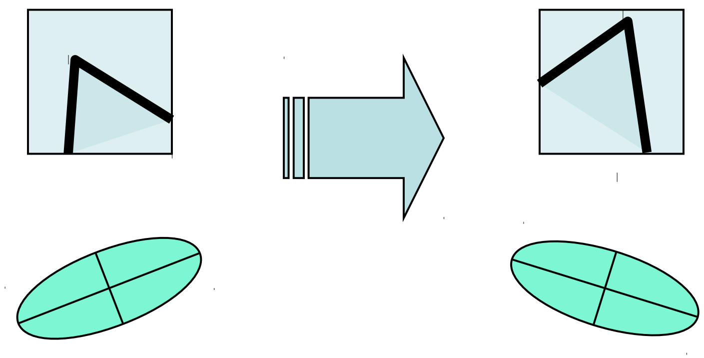


1) Find interest points

Harris Detector: Properties

- Rotation invariance

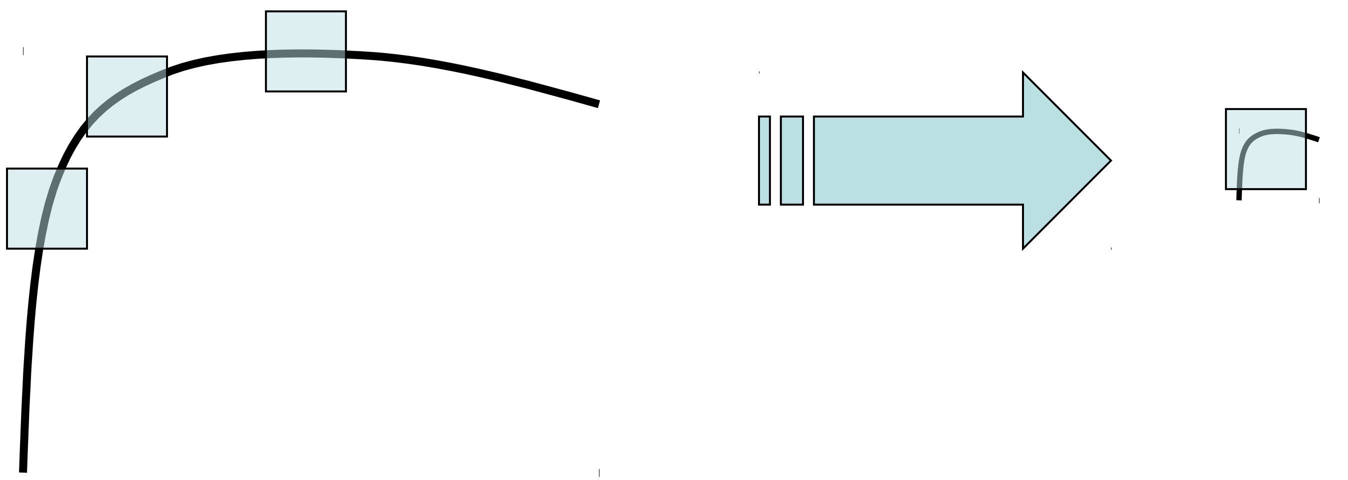


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R is invariant to image rotation

Harris Detector: Properties

- Not invariant to image scale



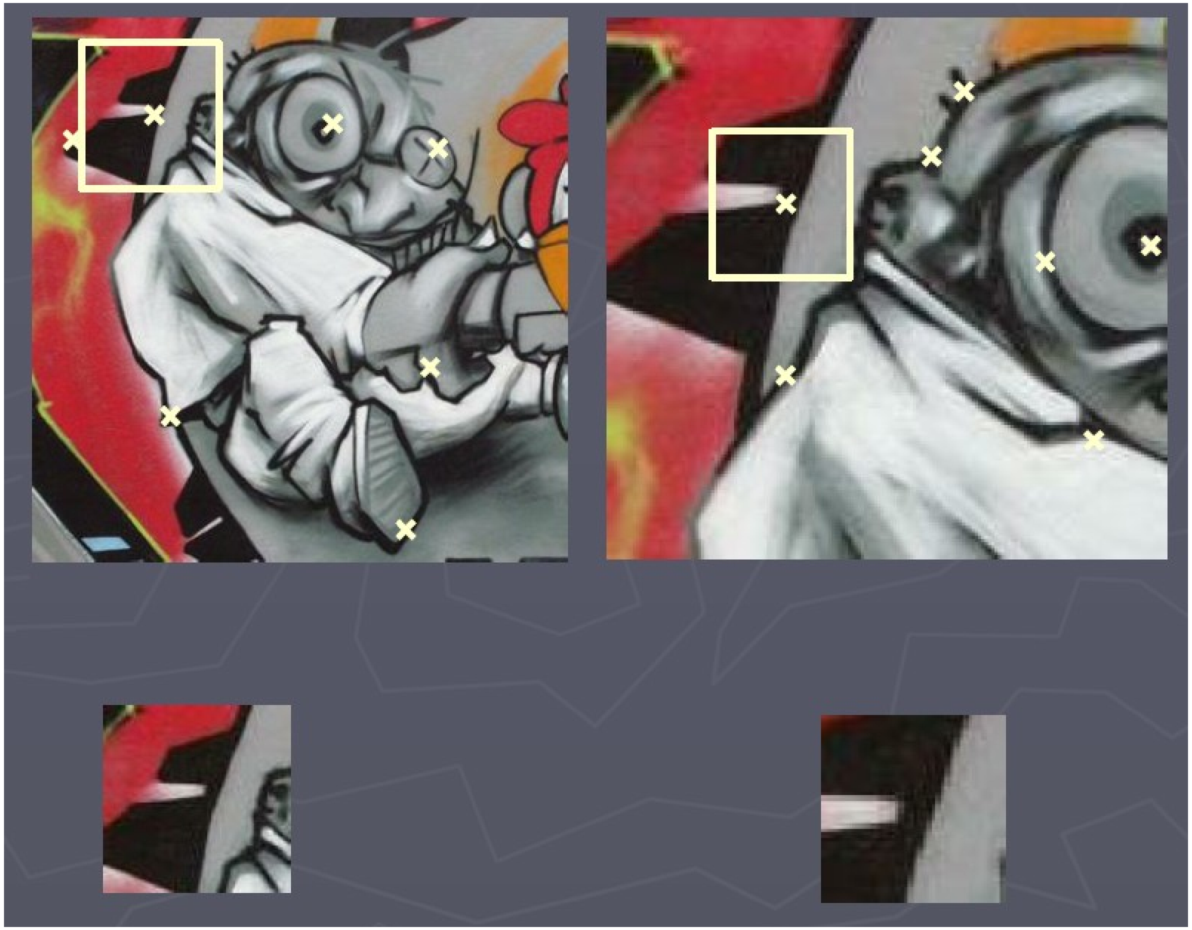
All points will be classified as edges

Corner !

- How can we detect **scale invariant** interest points?

Exhaustive search

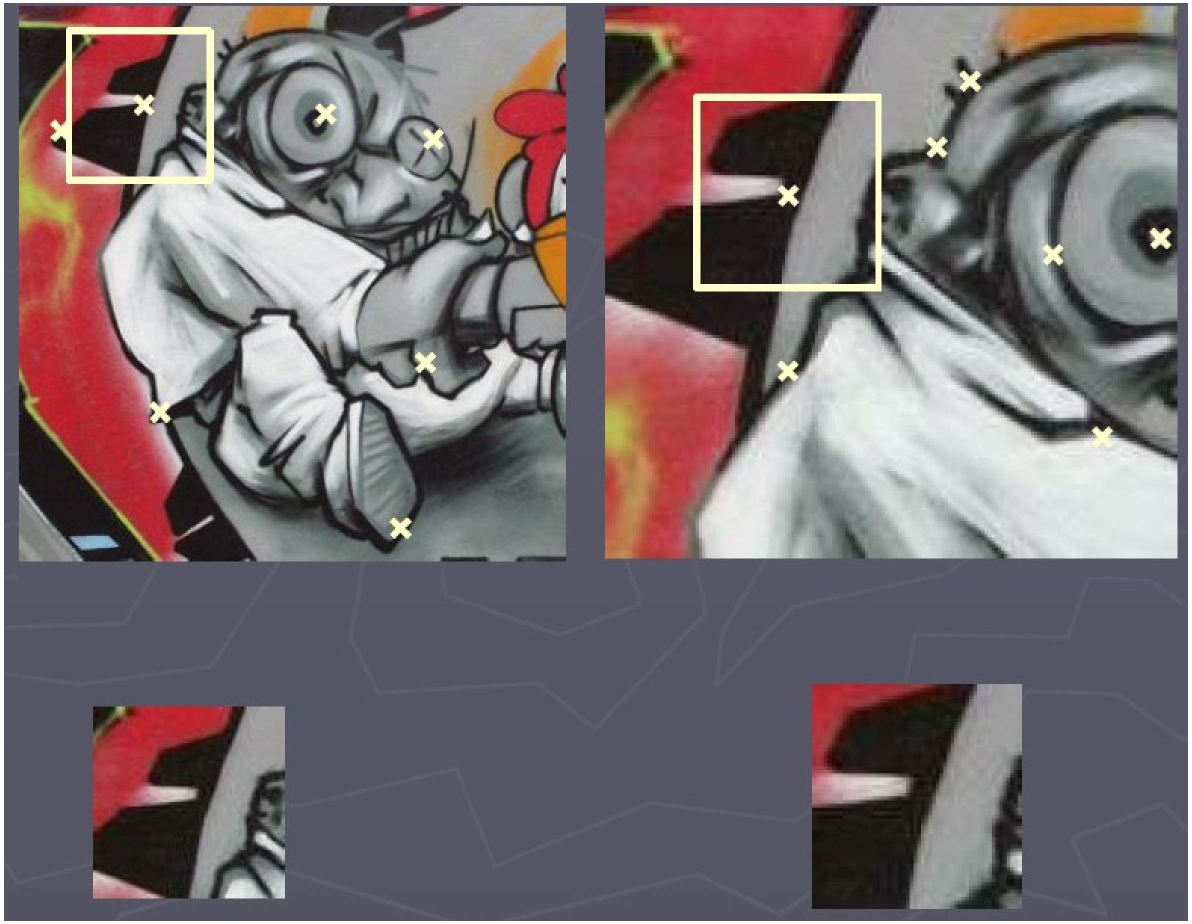
A multi-scale approach



Slide from T. Tuytelaars ECCV 2006 tutorial

Exhaustive search

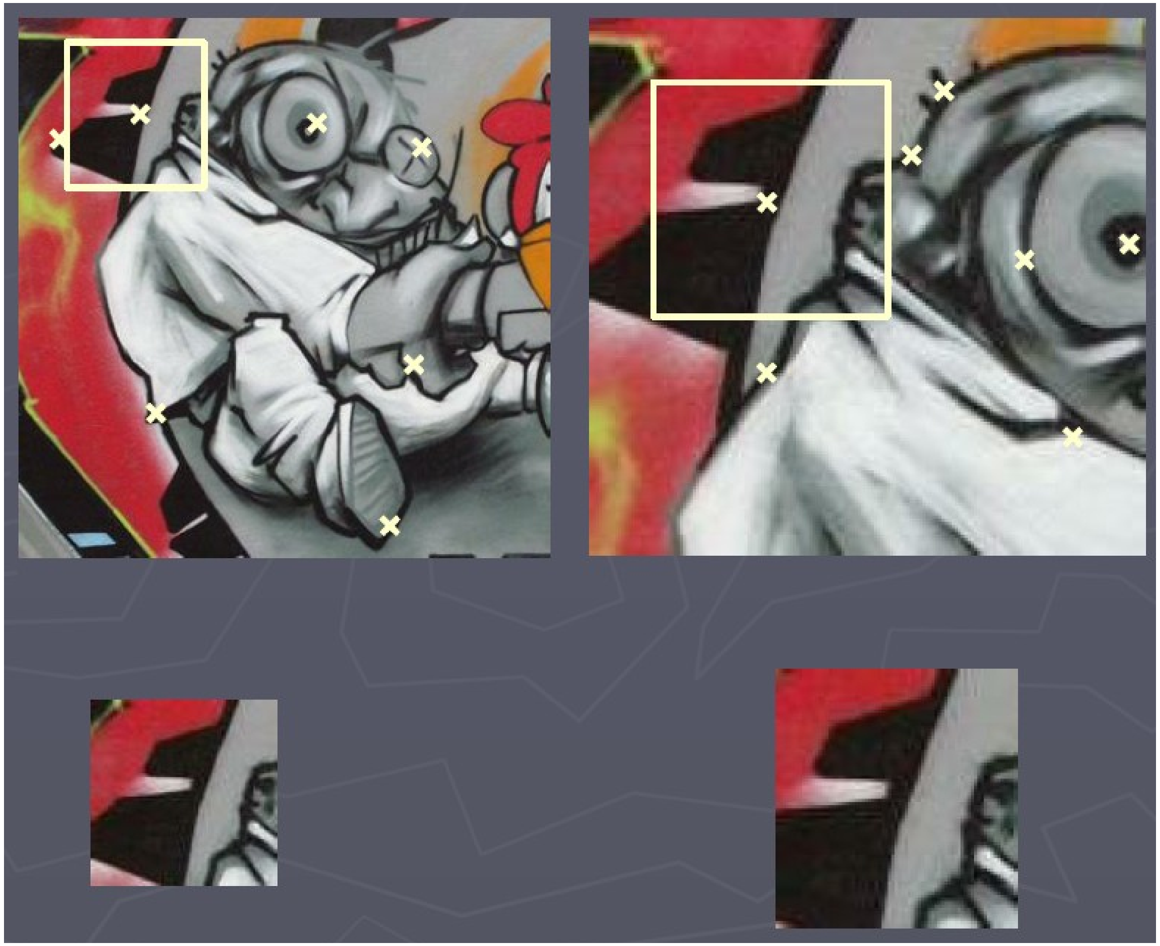
A multi-scale approach



Slide from T. Tuytelaars ECCV 2006 tutorial

Exhaustive search

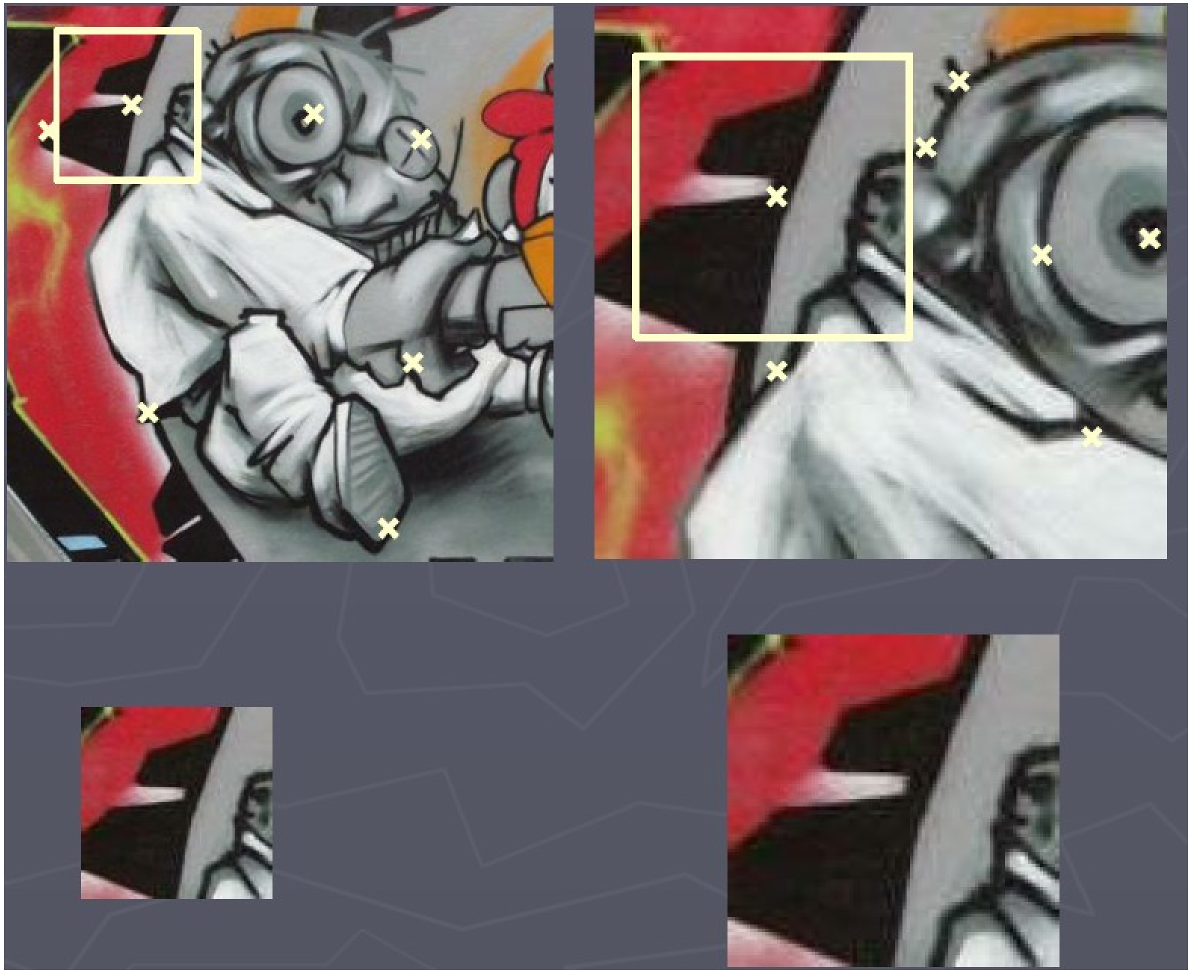
A multi-scale approach



Slide from T. Tuytelaars ECCV 2006 tutorial

Exhaustive search

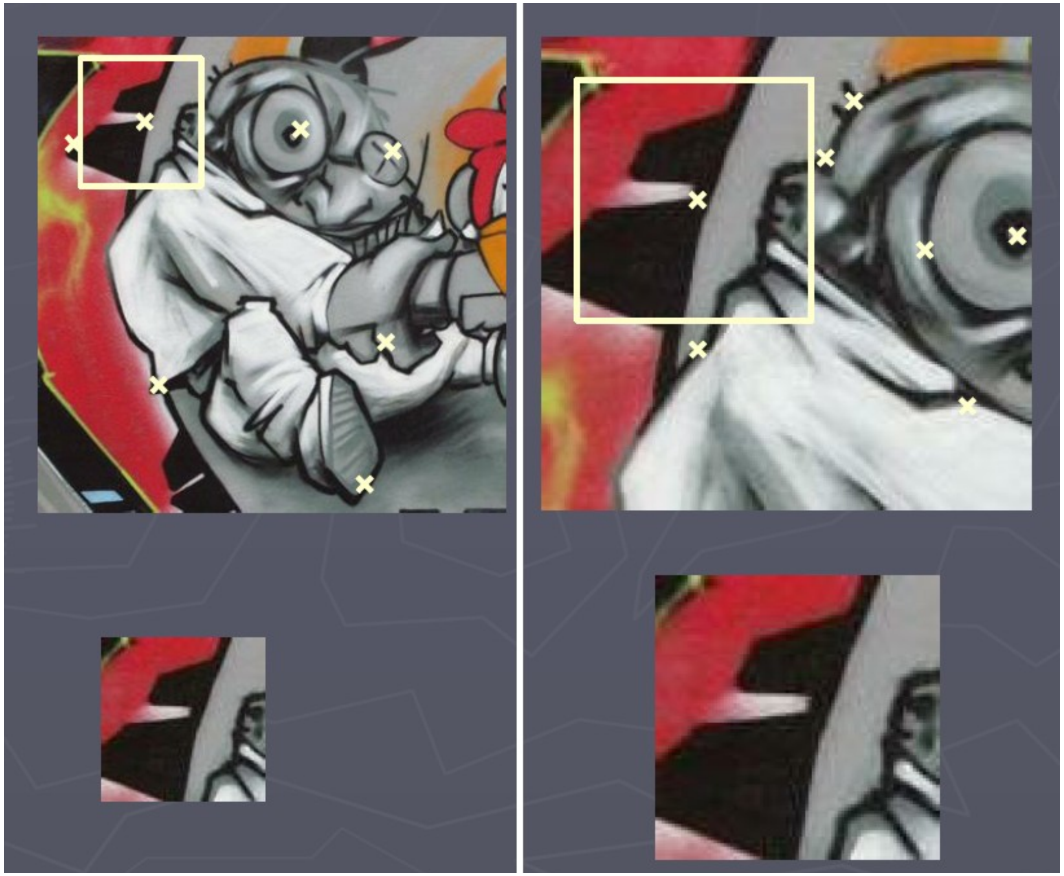
A multi-scale approach



Slide from T. Tuytelaars ECCV 2006 tutorial

Automatic scale selection

We want to extract the patches from each image *independently*.



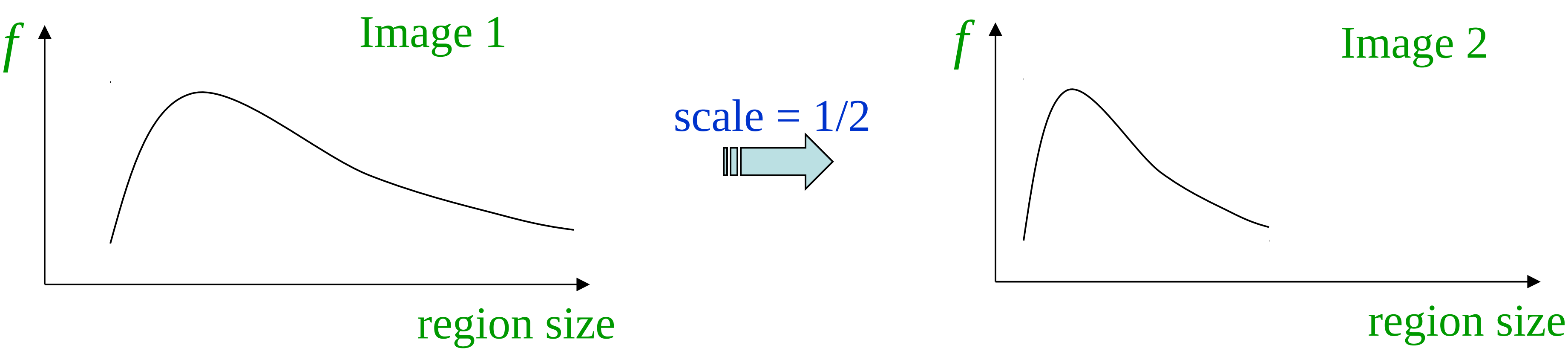
Slide adapted from T. Tuytelaars ECCV 2006 tutorial

Automatic scale selection

- Solution:
  - Design a function on the region, which is “scale invariant” (*the same for corresponding regions, even if they are at different scales*)

Example: average intensity. For corresponding regions (even of different sizes) it will be the same.

  - For a point in one image, we can consider it as a function of region size (patch width)



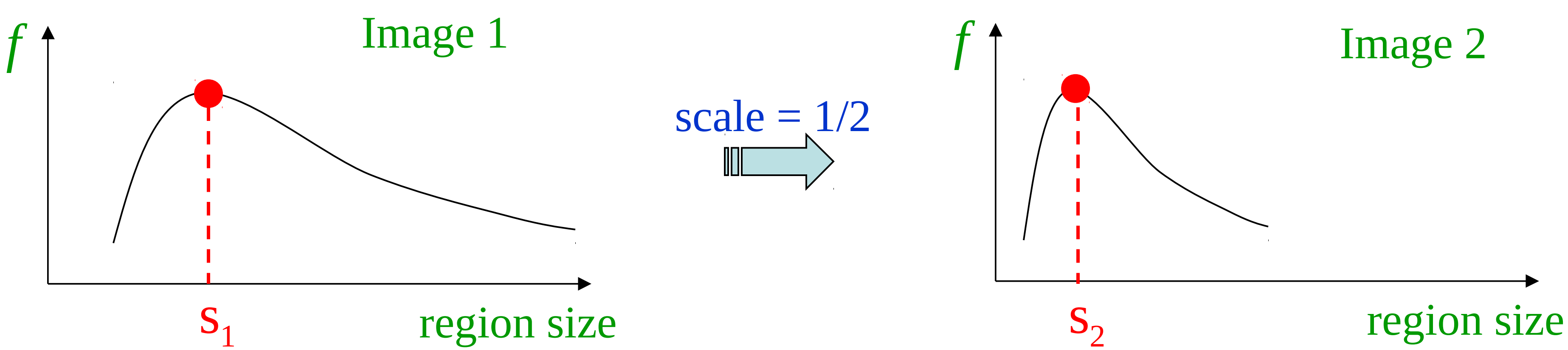
Automatic scale selection

- Common approach:

  Take a local maximum of this function

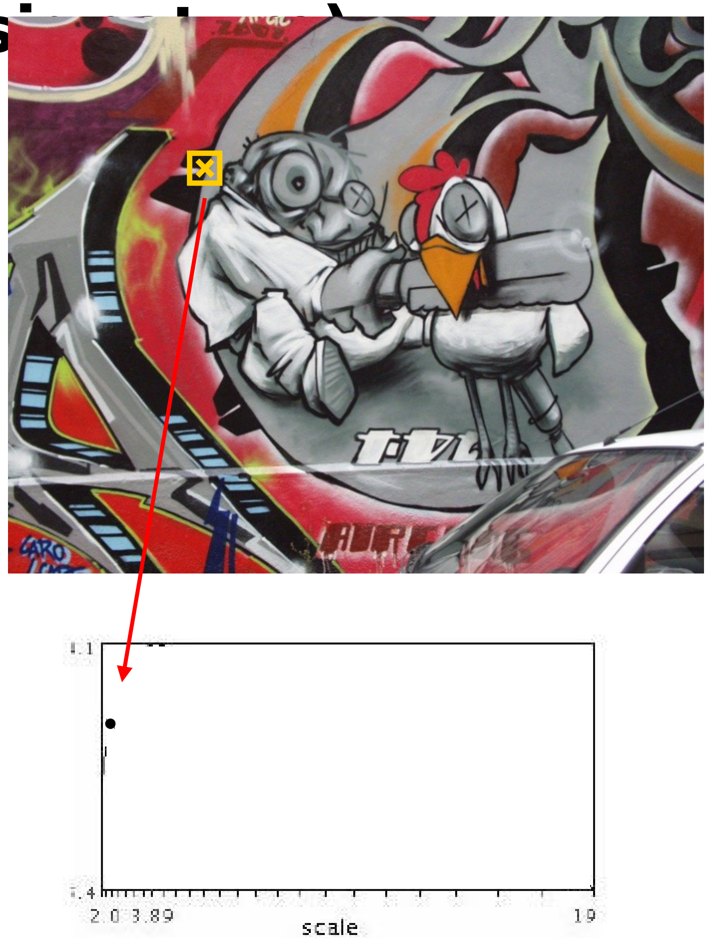
  Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

Important: this scale invariant region size is found in each image independently!



Automatic Scale Selection

- **Function responses for increasing scale (scale signature)**

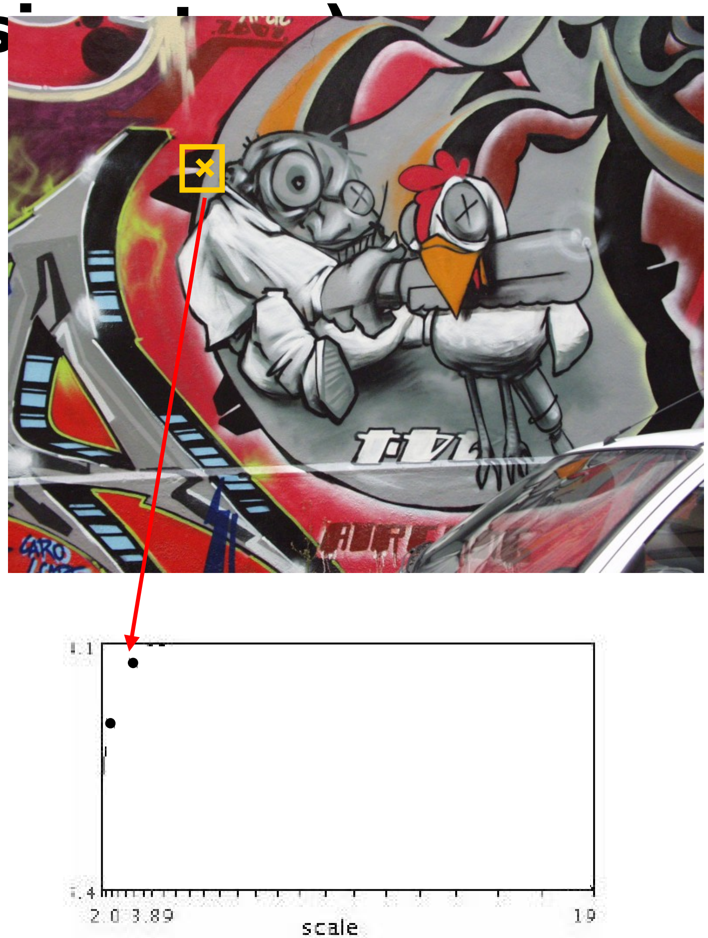


$f\left(I_{i_1 \ldots i_m}(x, \sigma)\right)$

$f\left(I_{i_1 \ldots i_m}(x', \sigma)\right)$

Automatic Scale Selection

- **Function responses for increasing scale (scale signature)**

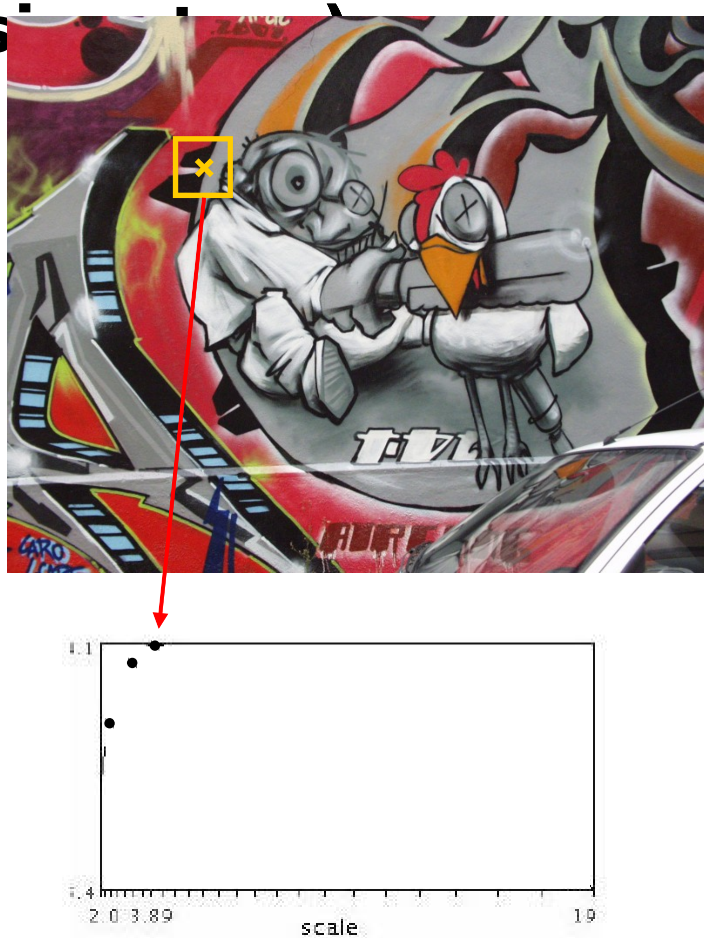


$f\left(I_{i_1 \ldots i_m}(x, \sigma)\right)$

$f\left(I_{i_1 \ldots i_m}(x', \sigma)\right)$

Automatic Scale Selection

- **Function responses for increasing scale (scale s**

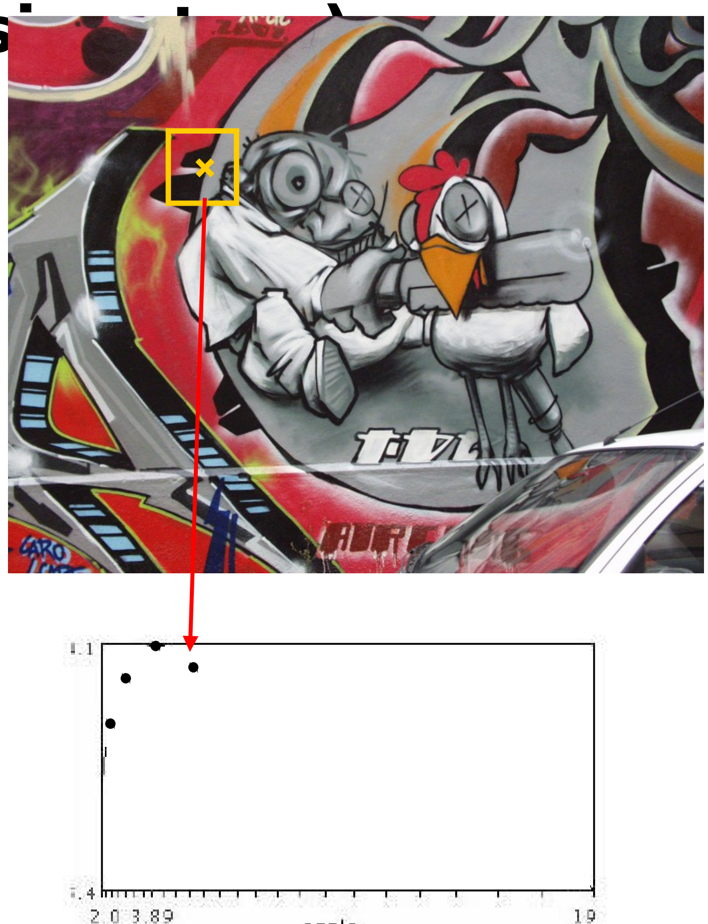


$f\left(I_{i_1 \ldots i_m}(x, \sigma)\right)$

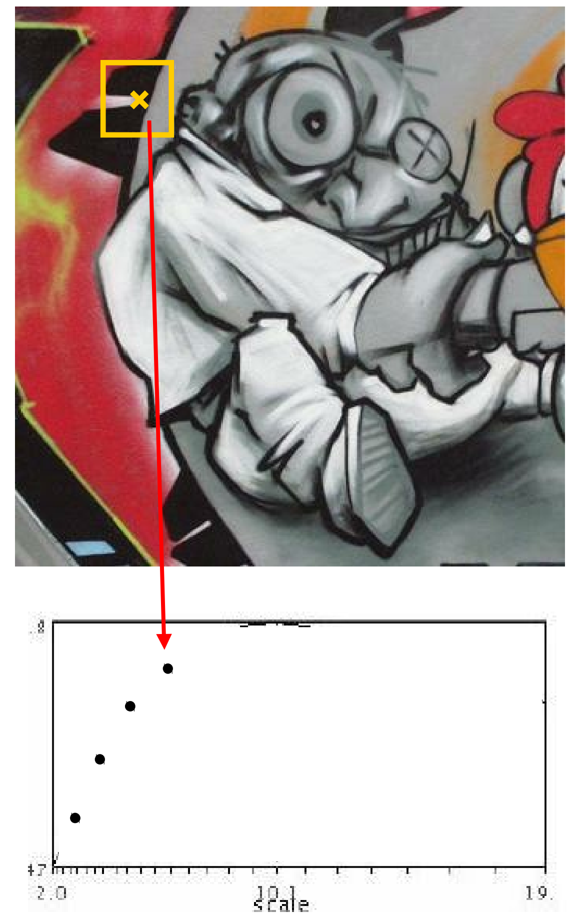
$f\left(I_{i_1 \ldots i_m}(x', \sigma)\right)$

Automatic Scale Selection

- **Function responses for increasing scale (scale signature)**



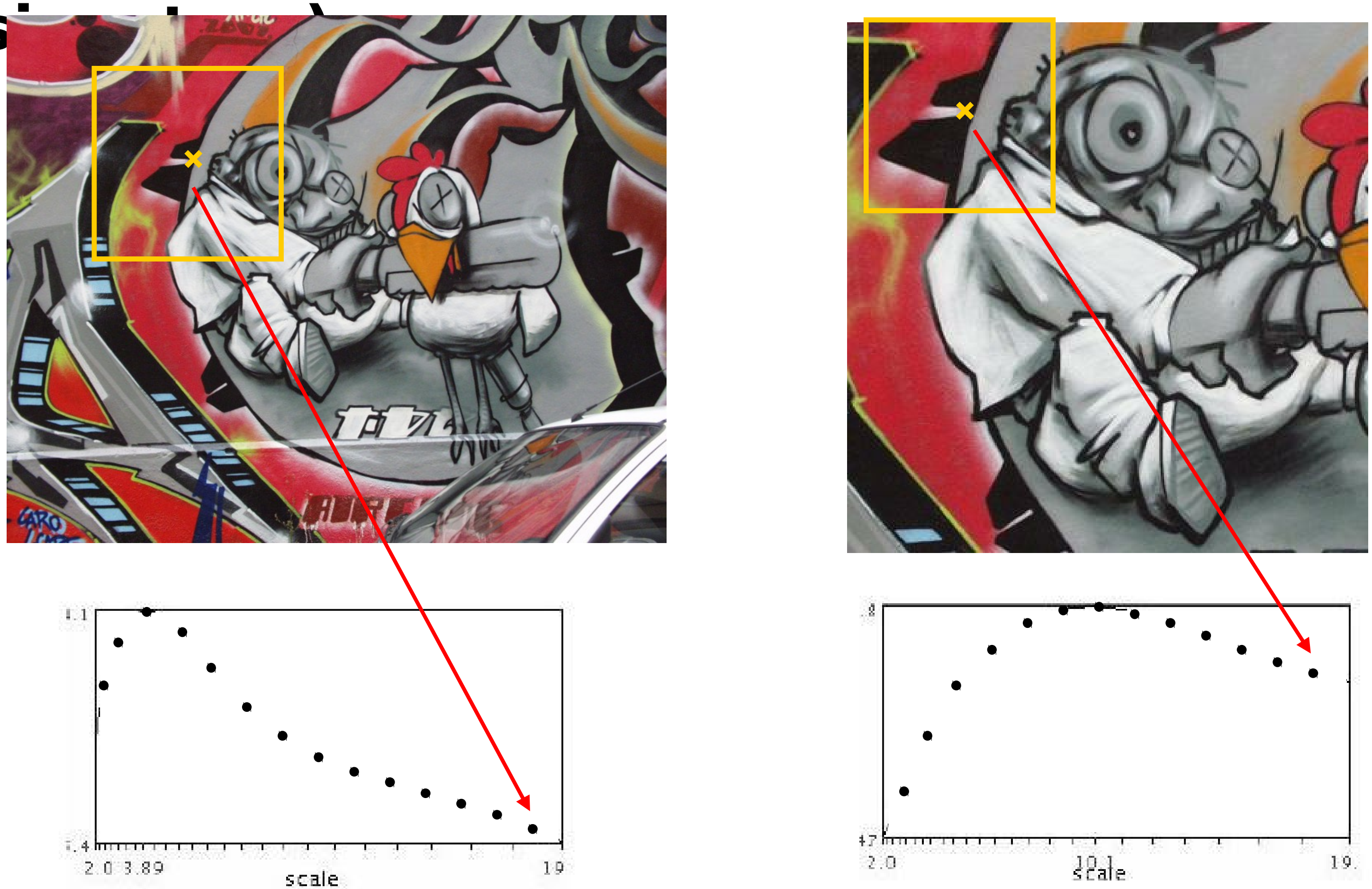
$f\left(I_{i_1 \dots i_m}(x, \sigma)\right)$



$f\left(I_{i_1 \dots i_m}(x', \sigma)\right)$

Automatic Scale Selection

- **Function responses for increasing scale (scale s**

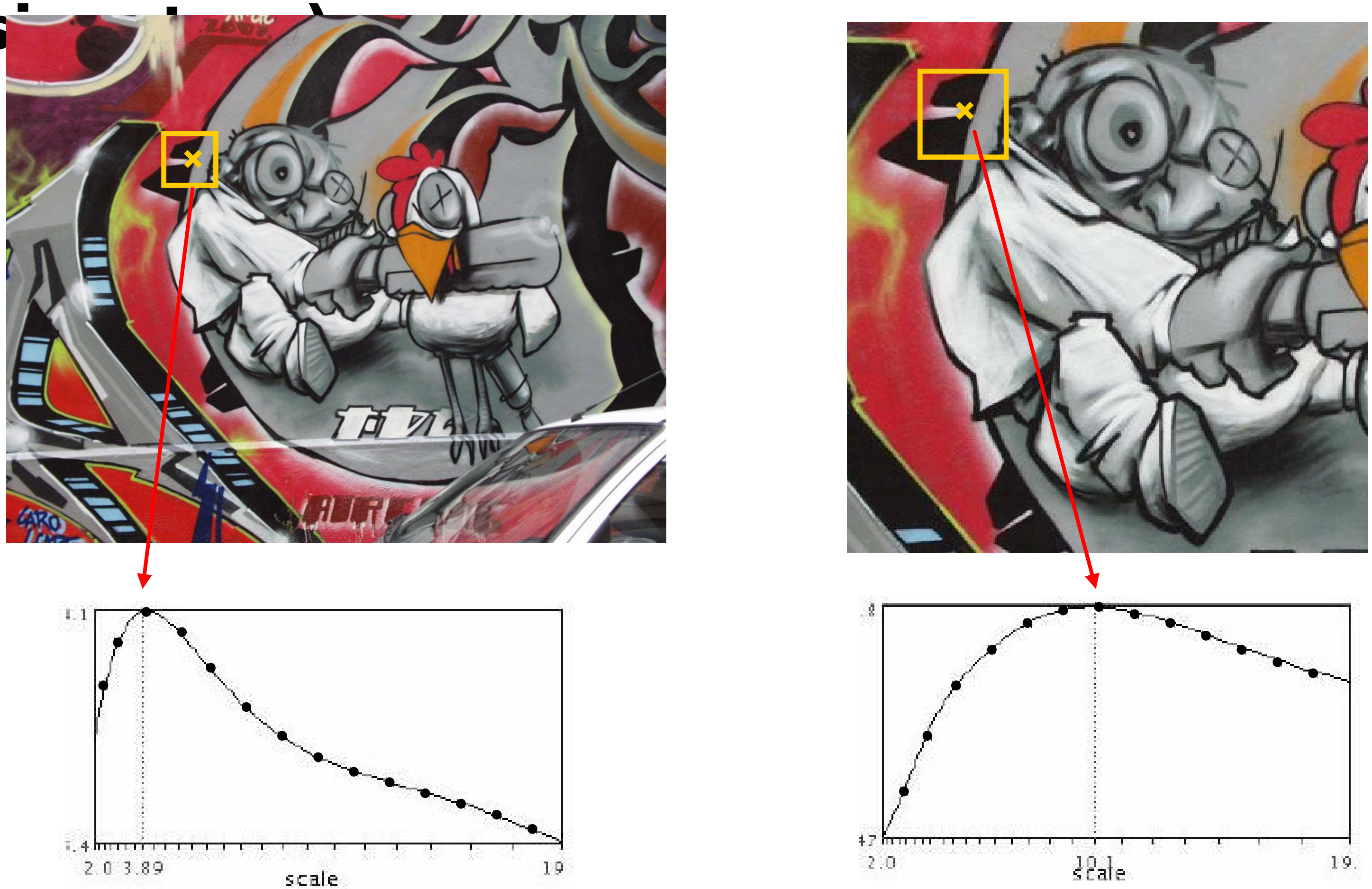


$f\left(I_{i_1 \dots i_m}(x, \sigma)\right)$

$f\left(I_{i_1 \dots i_m}(x', \sigma)\right)$

Automatic Scale Selection

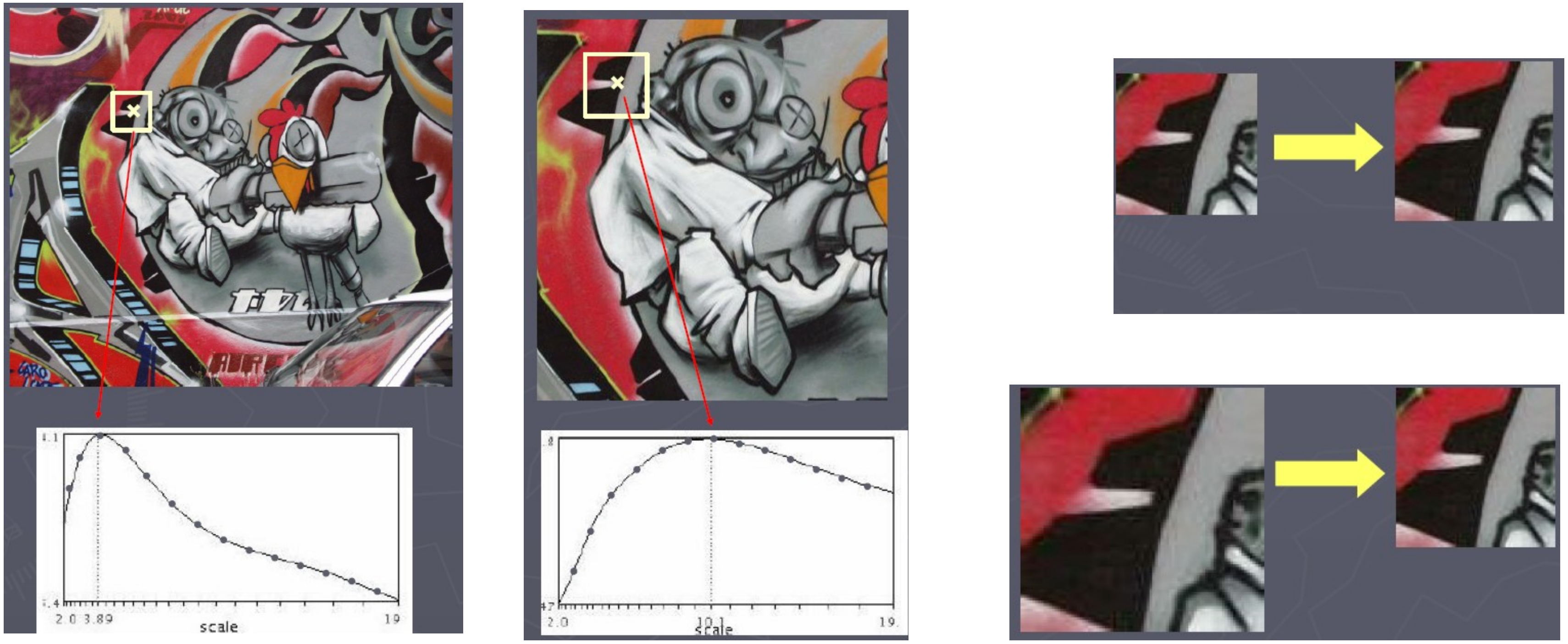
- **Function responses for increasing scale (scale signature)**



$f\left(I_{i_1 \ldots i_m}(x, \sigma)\right)$ $\quad$ $f\left(I_{i_1 \ldots i_m}(x', \sigma')\right)$

Scale selection

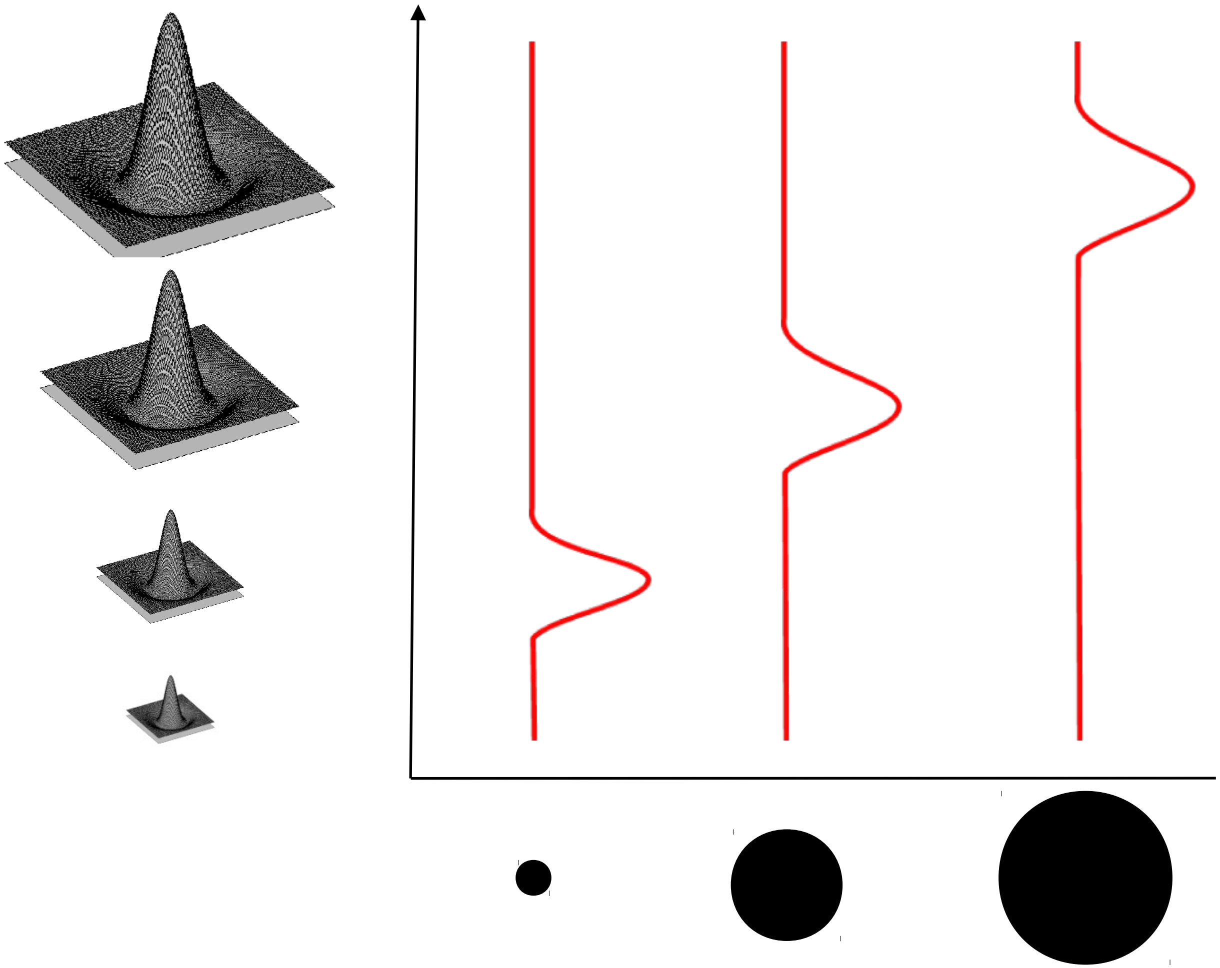
- Use the scale determined by detector to compute descriptor in a normalized frame



[Images from T. Tuytelaars]

What Is A Useful Signature Function?

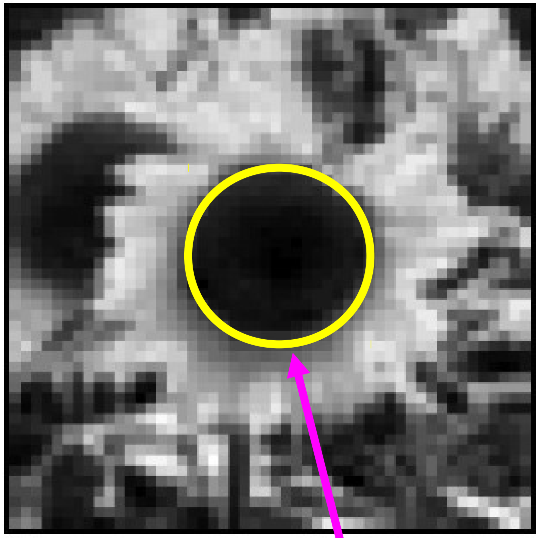
- **Laplacian-of-Gaussian = “blob” detector**

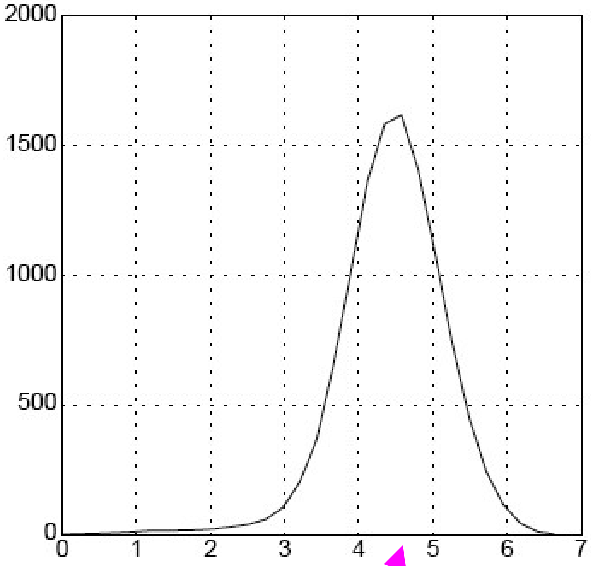


K. Grauman, B. Leibe

Characteristic scale

We define the *characteristic scale* as the scale that produces peak of Laplacian response





characteristic scale

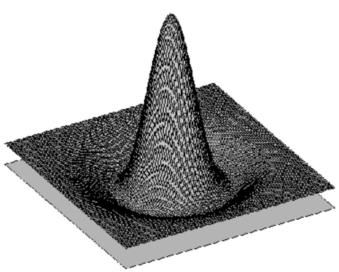
T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* **30** (2): pp 77--116.

Source: Lana Lazebnik

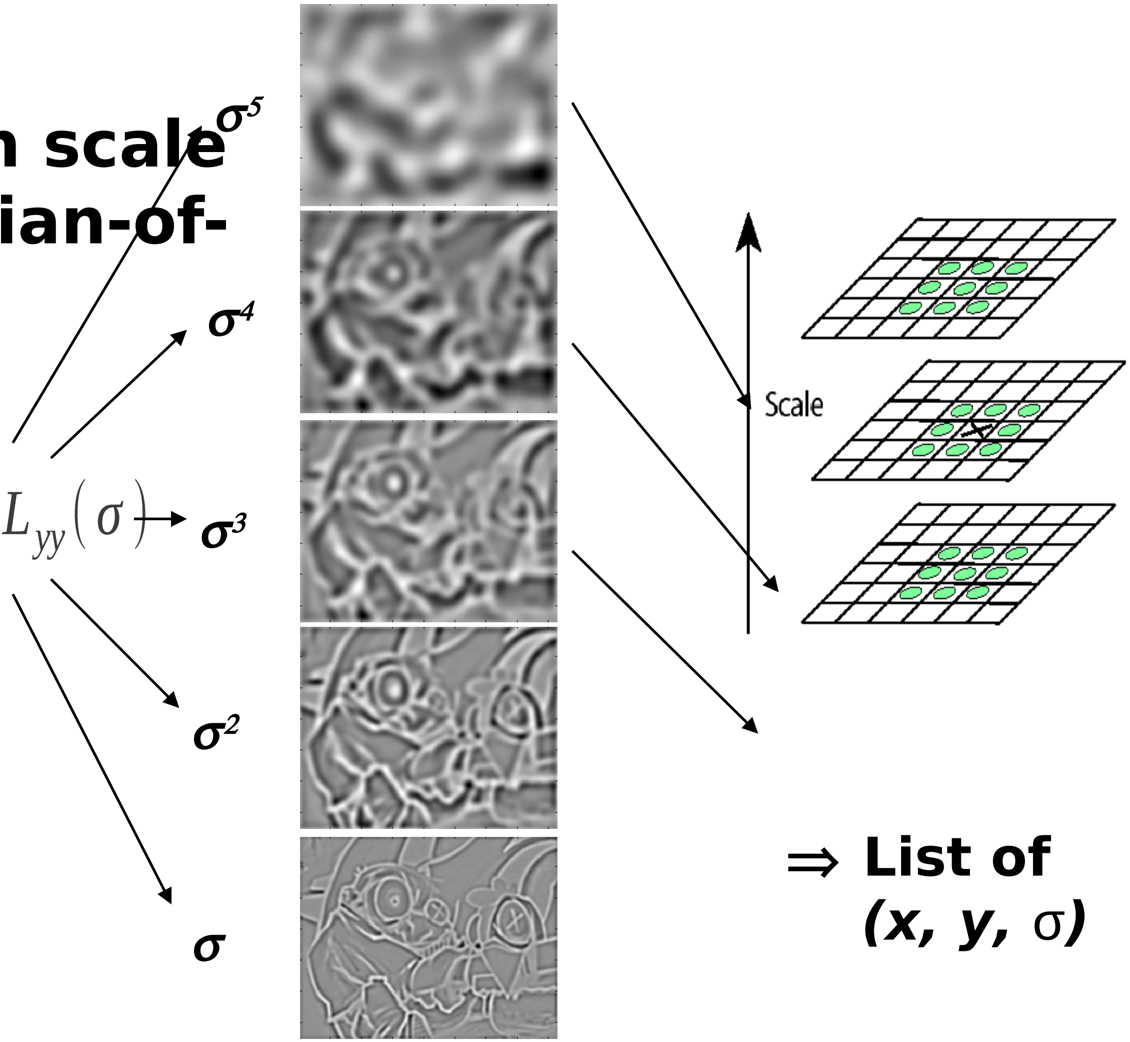
Laplacian-of-Gaussian (LoG)

- **Interest points:**

  Local maxima in scale space of Laplacian-of-Gaussian



$$L_{xx}(\sigma) + L_{yy}(\sigma)$$



$\Rightarrow$ **List of (x, y, σ)**

Scale-space blob detector: Example

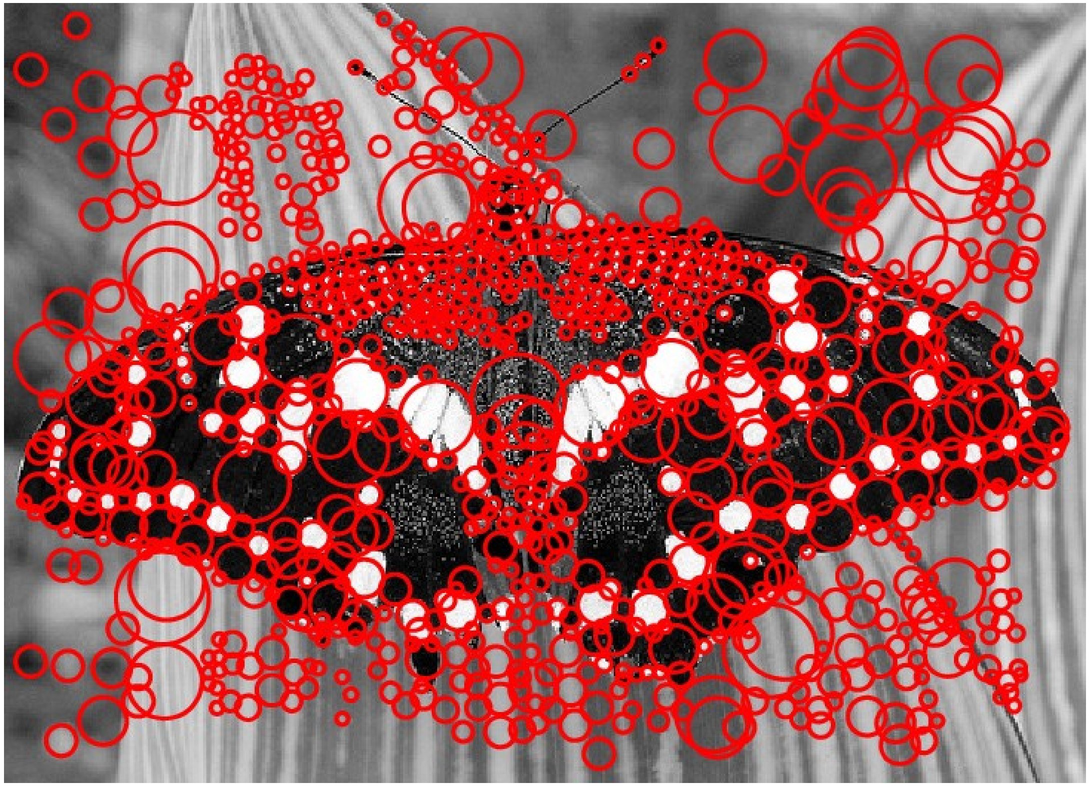
Source: Lana Lazebnik

Scale-space blob detector: Example

sigma = 11.9912

Source: Lana Lazebnik

Scale-space blob detector: Example



Source: Lana Lazebnik

Technical detail

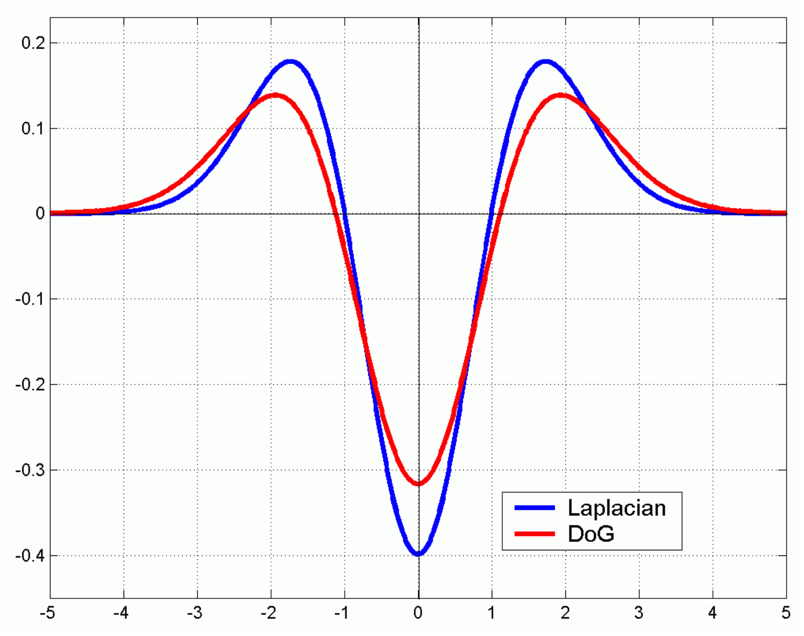
We can efficiently approximate the Laplacian with a difference of Gaussians:

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

(Laplacian)

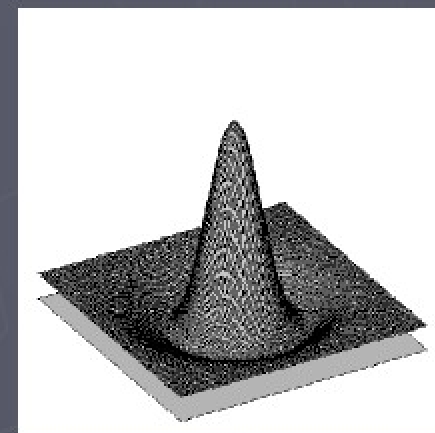
$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

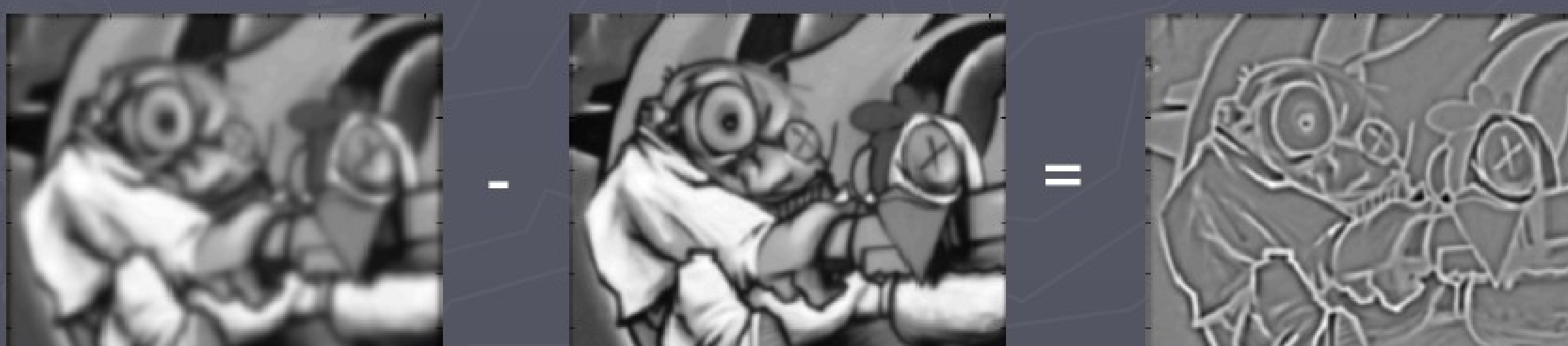
(Difference of Gaussians)



This is used in Lowe's SIFT (**S**cale **I**nvariant **F**eature **T**ransform) pipeline for keypoint detection

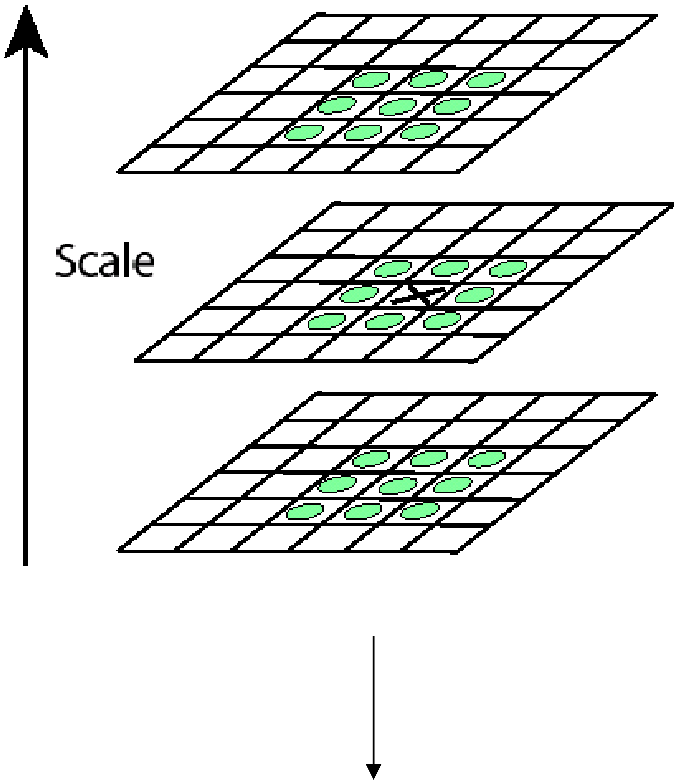
Difference of Gaussians as approximation of the Laplacian of Gaussian





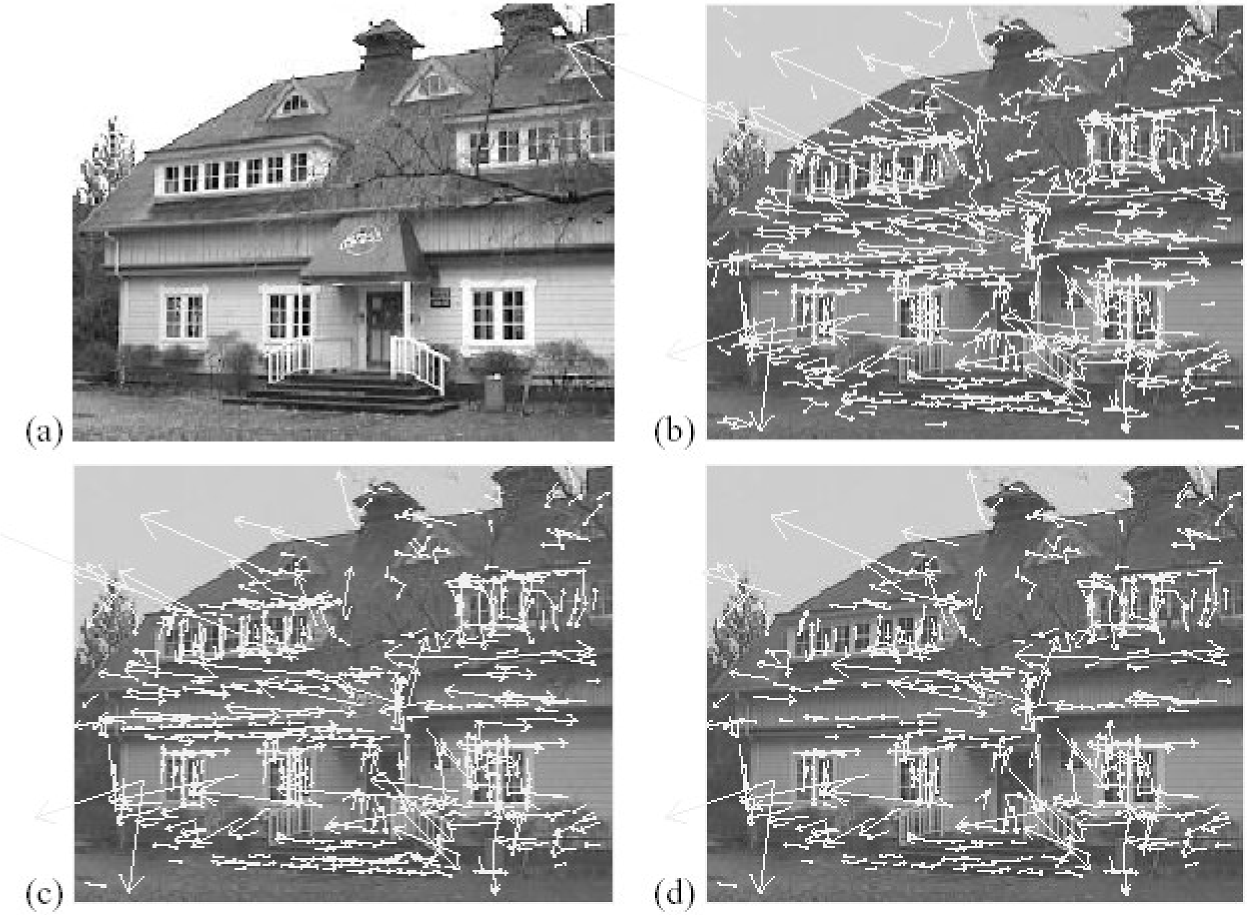
Key point localization with DoG

- Detect maxima of difference-of-Gaussian (DoG) in scale space
- Then reject points with low contrast (threshold)
- Eliminate edge responses



Candidate keypoints: list of (x,y,σ)

Example of keypoint detection



(a) 233x189 image
(b) 832 DOG extrema
(c) 729 left after peak value threshold
(d) 536 left after testing ratio of principle curvatures (removing edge responses)

Scale Invariant Detection: Summary

- Given: two images of the same scene with a large *scale difference* between them
- Goal: find *the same* interest points *independently* in each image
- Solution: search for *maxima* of suitable functions in *scale* and in *space* (over the image)

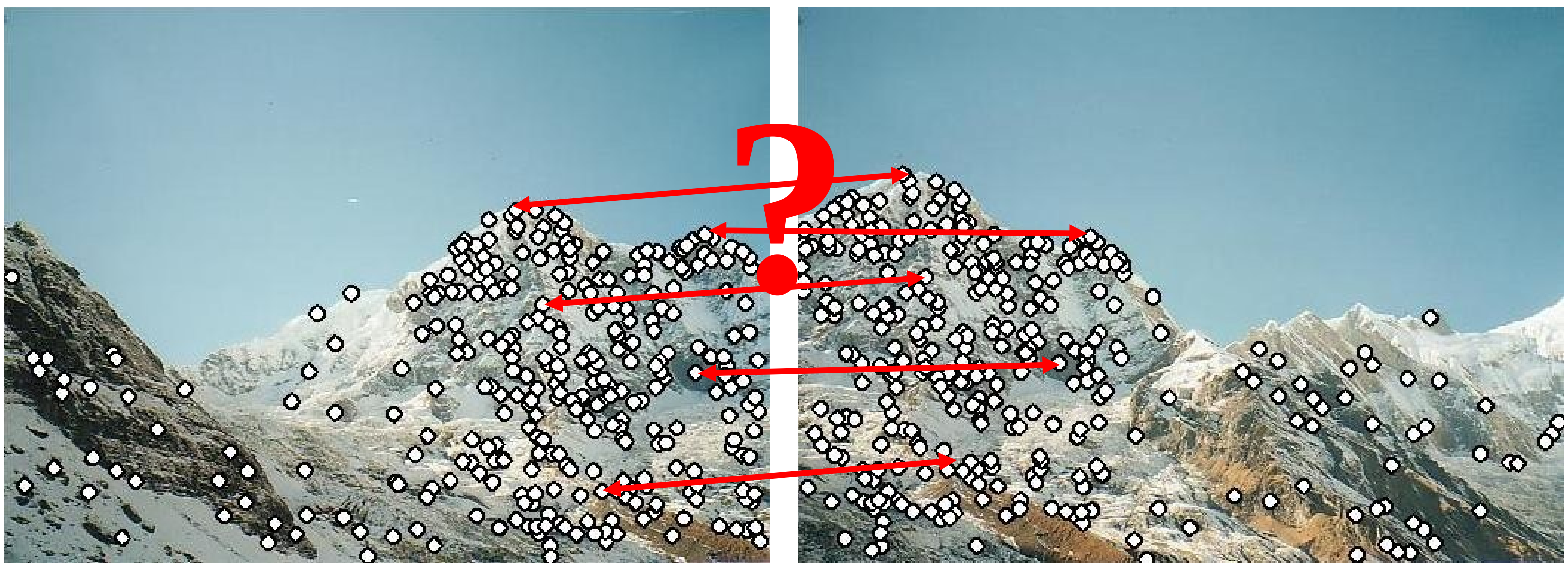
Main questions

- Where will the interest points come from?
  - What are salient features that we'll *detect* in multiple views?
- How to *describe* a local region?
- How to establish *correspondences*, i.e., compute matches?

Local descriptors

- We know how to detect points
- Next question:

How to *describe* them for matching?



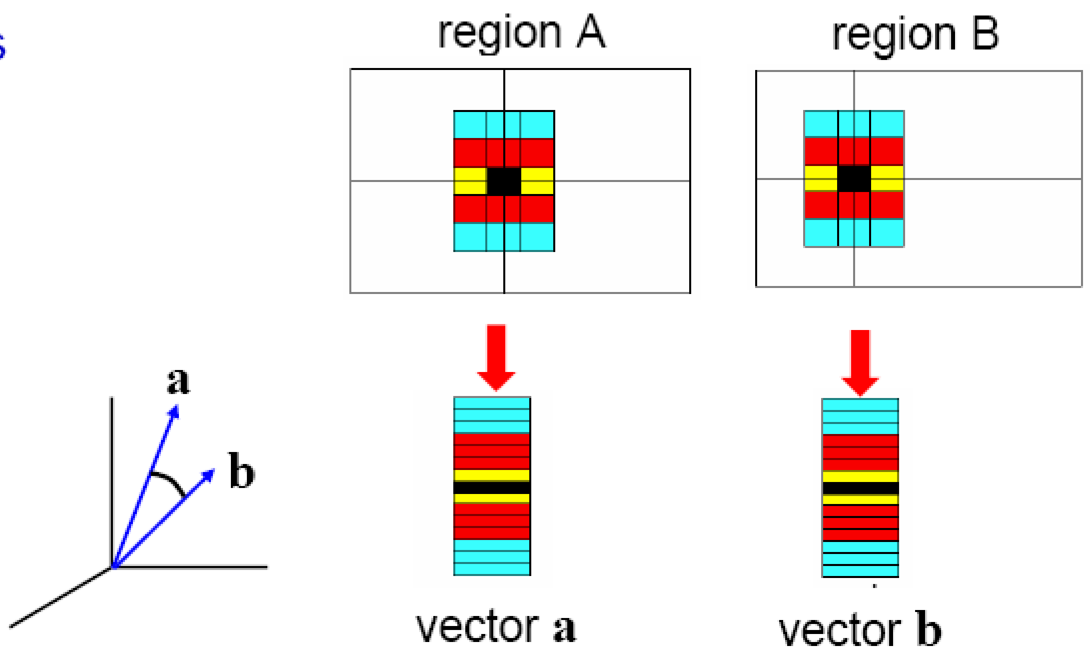
Point descriptor should be:
1. Invariant
2. Distinctive

Local descriptors

- Simplest descriptor: list of intensities within a patch.
- What is this going to be invariant to?

Write regions as vectors

$$A \rightarrow \mathbf{a}, \;\; B \rightarrow \mathbf{b}$$

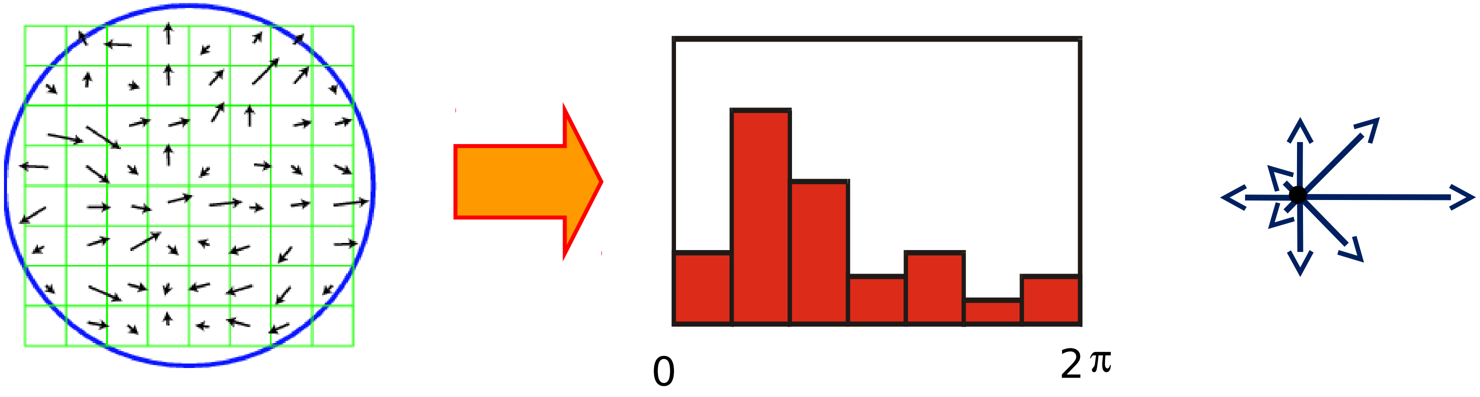


Feature descriptors

Disadvantage of patches as descriptors:

- Small shifts can affect matching score a lot

Solution: histograms



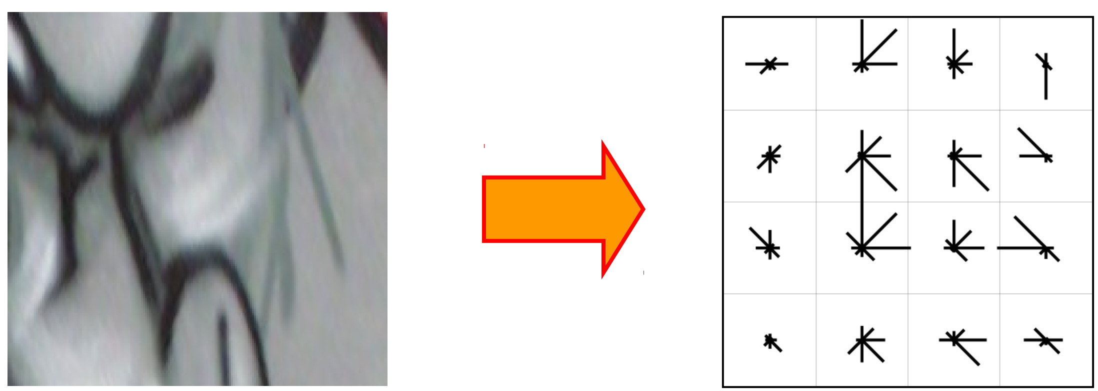
Source: Lana Lazebnik

Feature descriptors: SIFT

Scale **I**nvariant **F**eature **T**ransform

Descriptor computation:

- Divide patch into 4x4 sub-patches: 16 cells
- Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
- Resulting descriptor: 4x4x8 = 128 dimensions



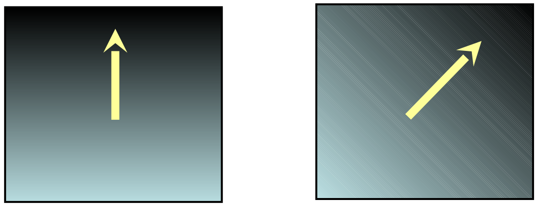
David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Source: Lana Lazebnik

Rotation Invariant Descriptors

- Find local orientation

  Dominant direction of gradient for the image patch



- Rotate patch according to this angle

  This puts the patches into a canonical orientation.

Rotation Invariant Descriptors

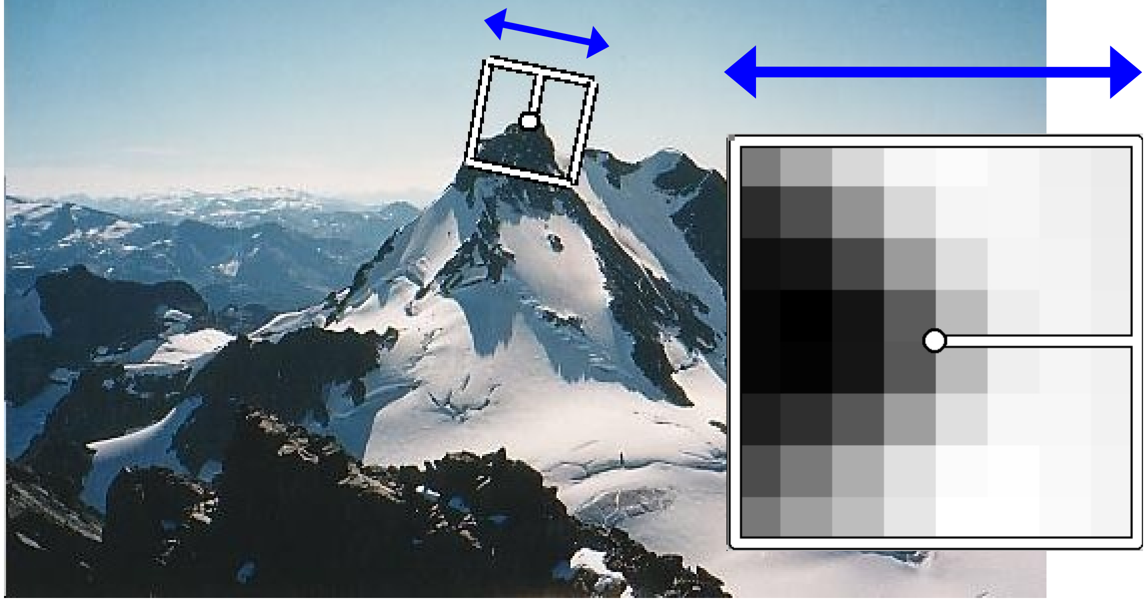


Image from Matthew Brown

Feature descriptors: SIFT

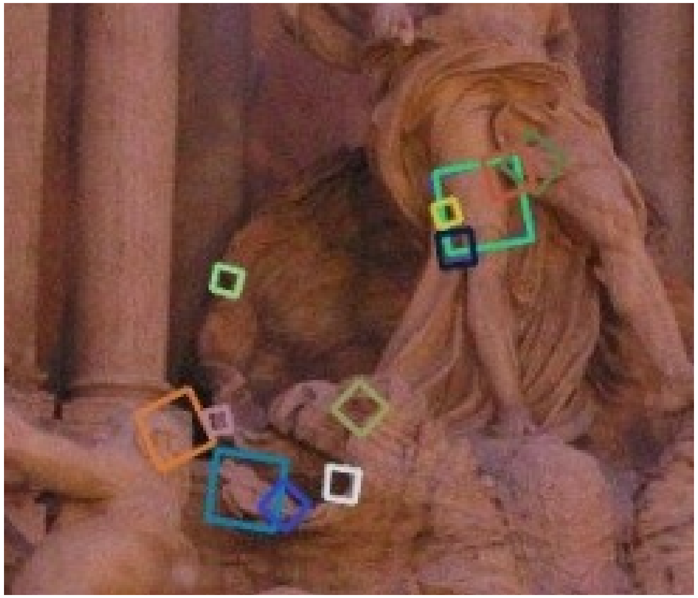
Extraordinarily robust matching technique

- 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

Steve Seitz

Working with SIFT descriptors

- One image yields:
  - n 128-dimensional descriptors: each one is a histogram of the gradient orientations within a patch
    - [n x 128 matrix]
  - n scale parameters specifying the size of each patch
    - [n x 1 vector]
  - n orientation parameters specifying the angle of the patch
    - [n x 1 vector]
  - n 2d points giving positions of the patches
    - [n x 2 matrix]



Main questions

- Where will the interest points come from?
  – What are salient features that we'll *detect* in multiple views?
- How to *describe* a local region?
- How to establish *correspondences*, i.e., compute matches?

We stopped here on Tuesday, to be continued Thursday