C280, Computer Vision

Prof. Trevor Darrell

trevor@eecs.berkeley.edu

Lecture 6: Local Features

Last Time: Image Pyramids

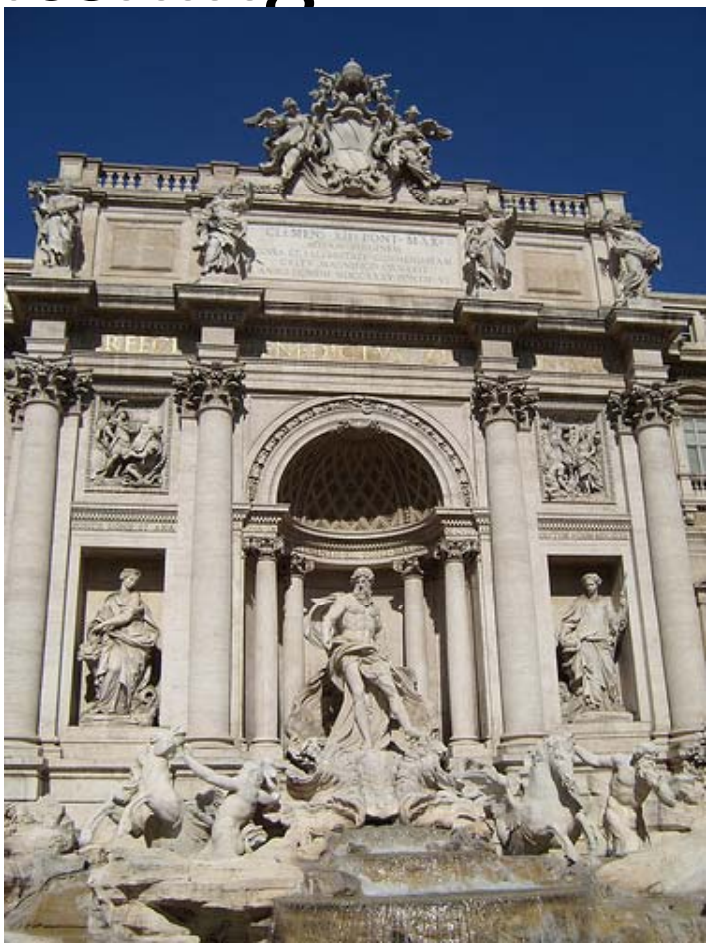
- Review of Fourier Transform
- Sampling and Aliasing
- Image Pyramids
- Applications: Blending and noise removal

Today: Feature Detection and Matching

- Local features
- Pyramids for invariant feature detection
- Invariant descriptors
- Matching

Image matching

by Diva Sian



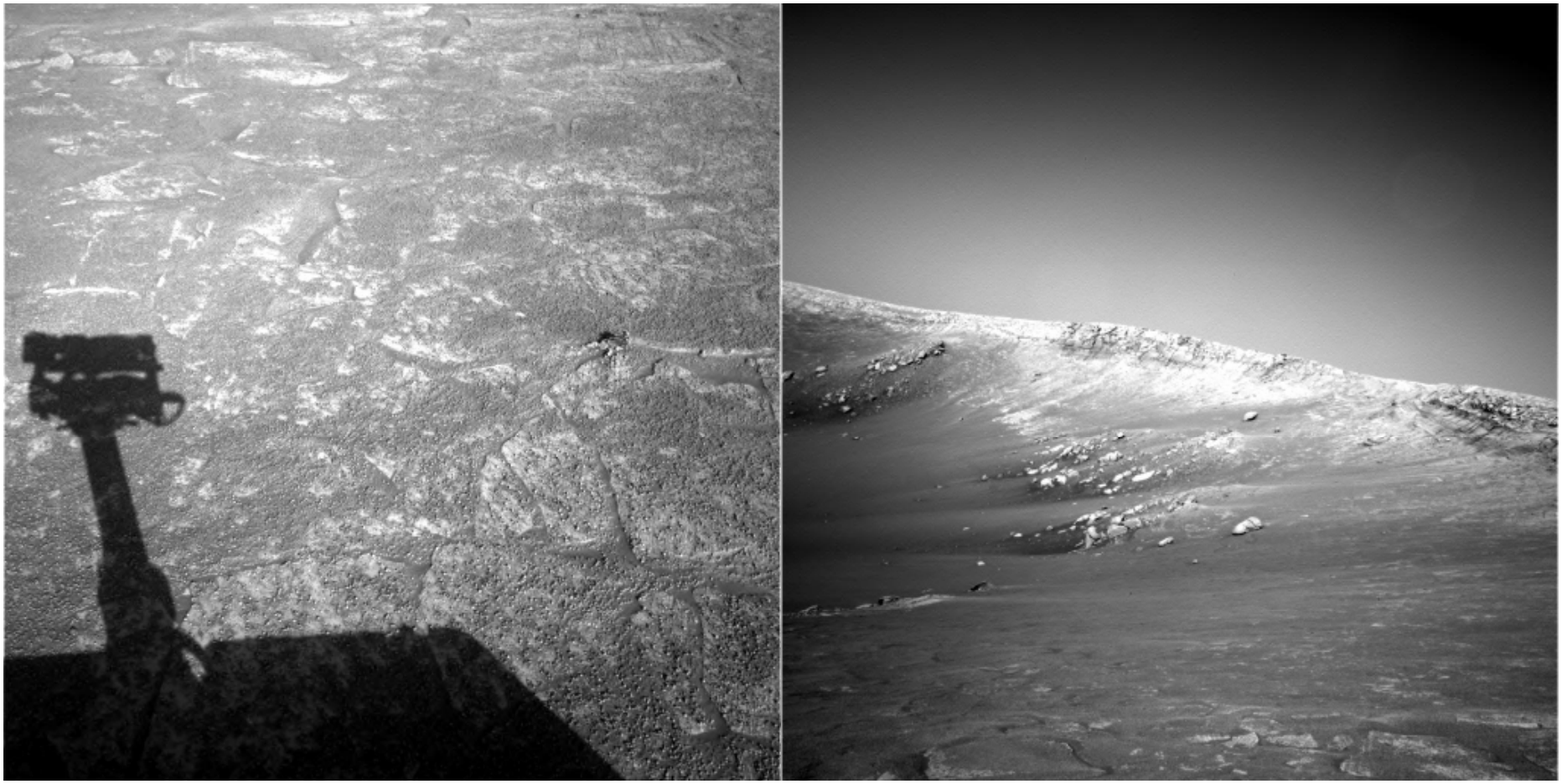
by swashford

Harder case

by Diva Sian

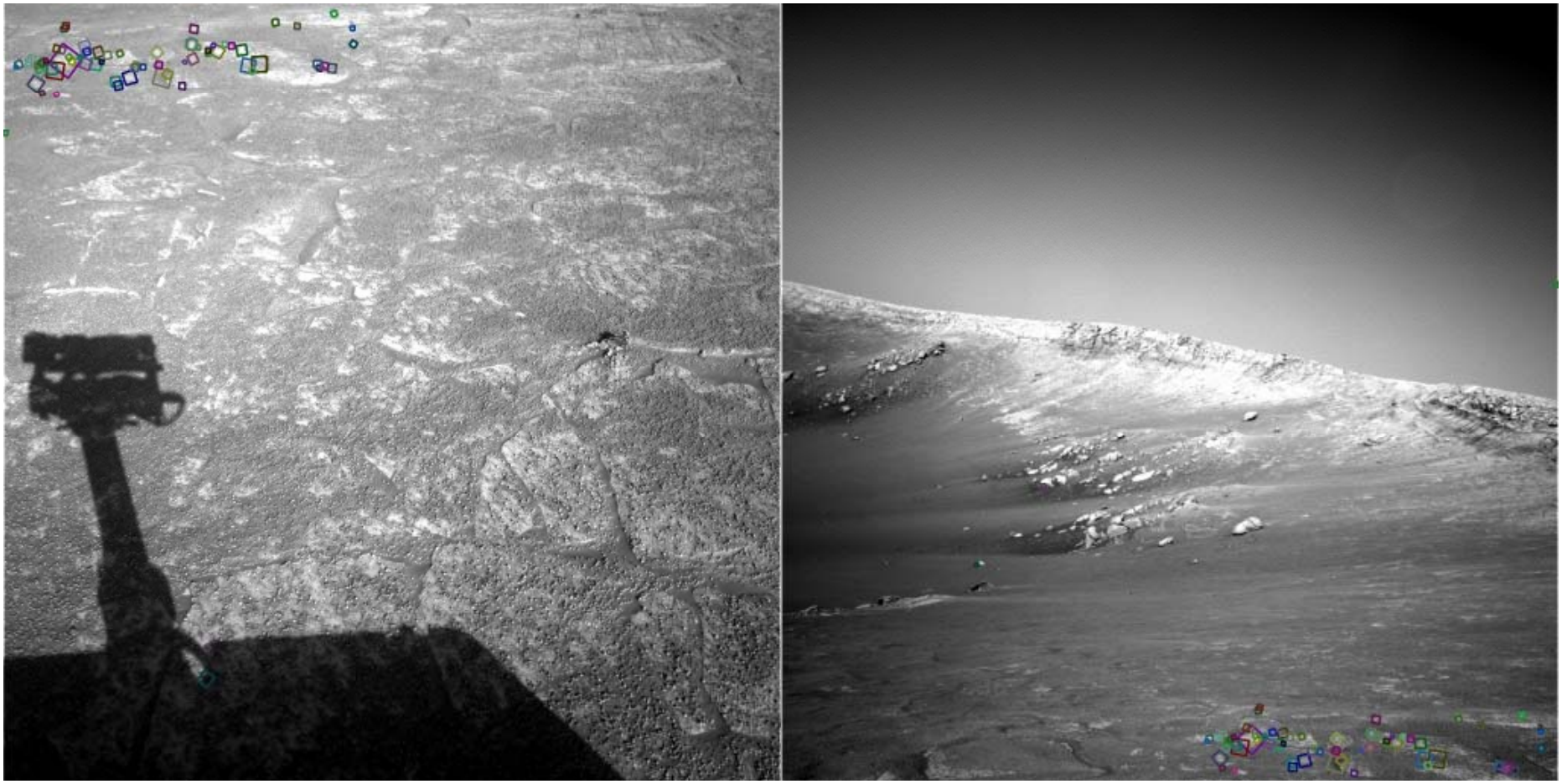
by scgbt

Harder still?



NASA Mars Rover images

Answer below (look for tiny colored squares…)



NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

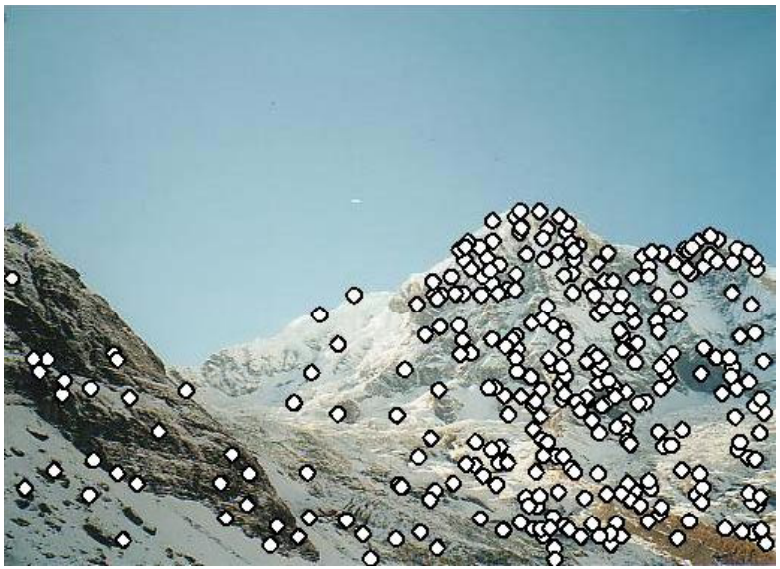
Local features and alignment

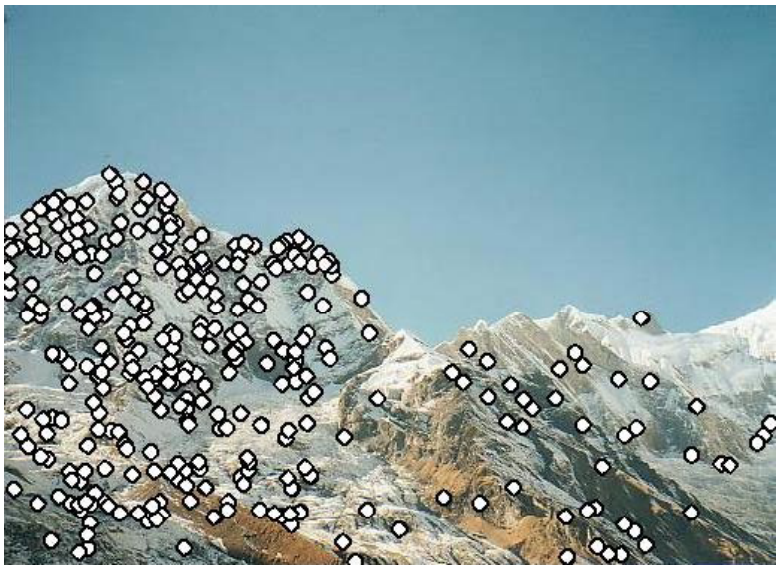
- We need to match (align) images
- Global methods sensitive to occlusion, lighting, parallax effects. So look for local features that match well.
- How would you do it by eye?

[Darya Frolova and Denis Simakov]

Local features and alignment

- Detect feature points in both images

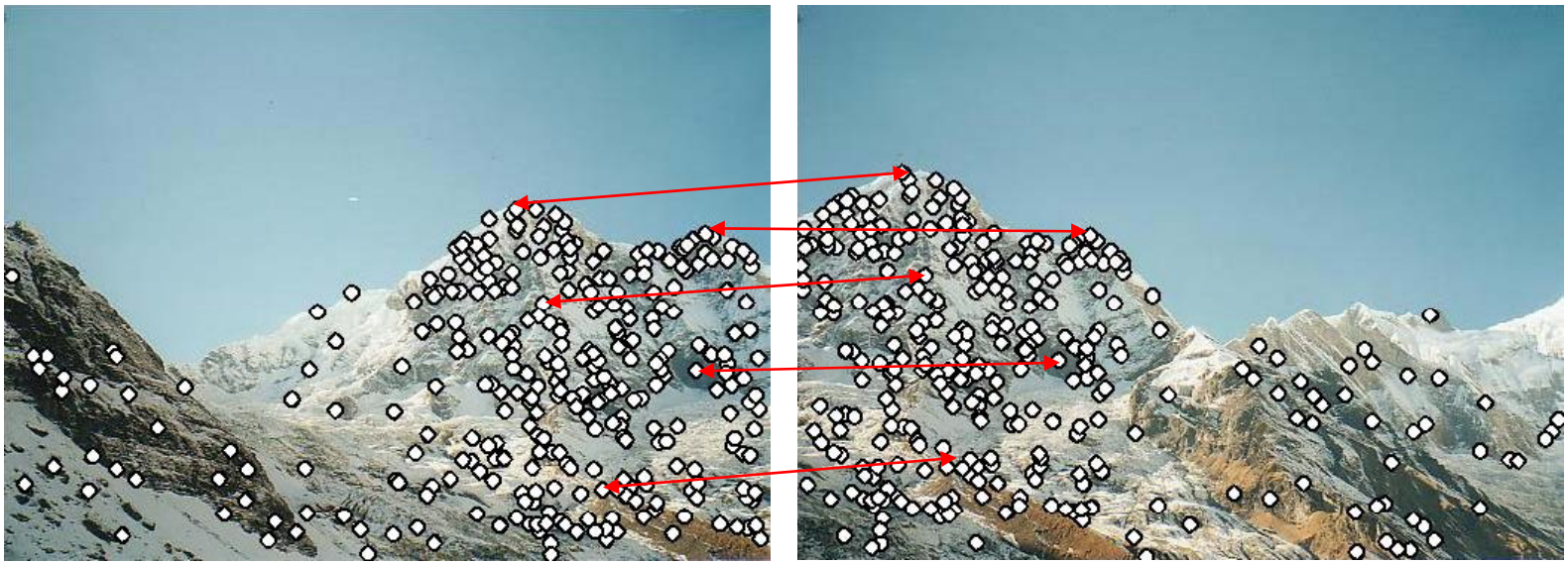




[Darya Frolova and Denis Simakov]

Local features and alignment

- Detect feature points in both images
- Find corresponding pairs



[Darya Frolova and Denis Simakov]

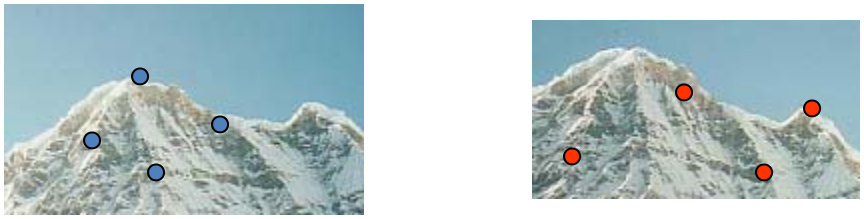
Local features and alignment

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

[Darya Frolova and Denis Simakov]

Local features and alignment

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



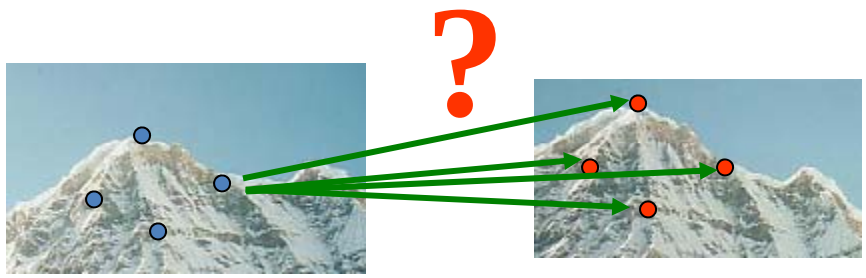
no chance to match!

We need a repeatable detector

[Darya Frolova and Denis Simakov]

Local features and alignment

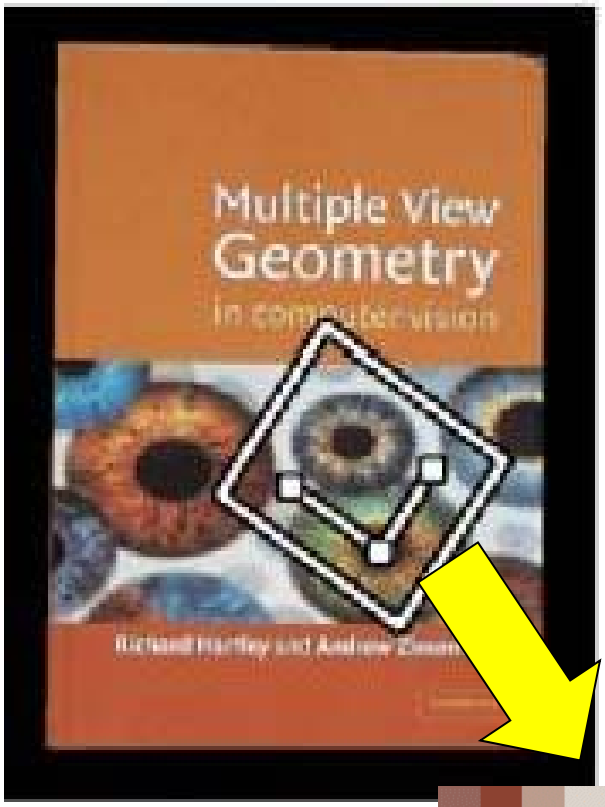
- Problem 2:
  - For each point correctly recognize the corresponding one

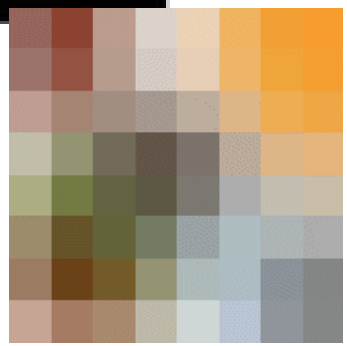


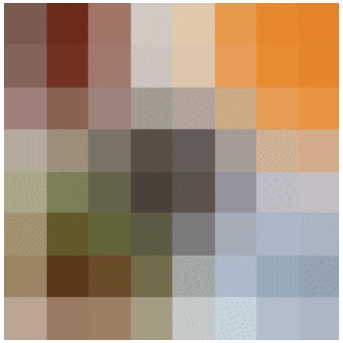
We need a reliable and distinctive descriptor

[Darya Frolova and Denis Simakov]

Geometric transformations







Photometric transformations

Figure from T. Tuytelaars ECCV 2006 tutorial

And other nuisances…

- Noise
- Blur
- Compression artifacts
- …

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

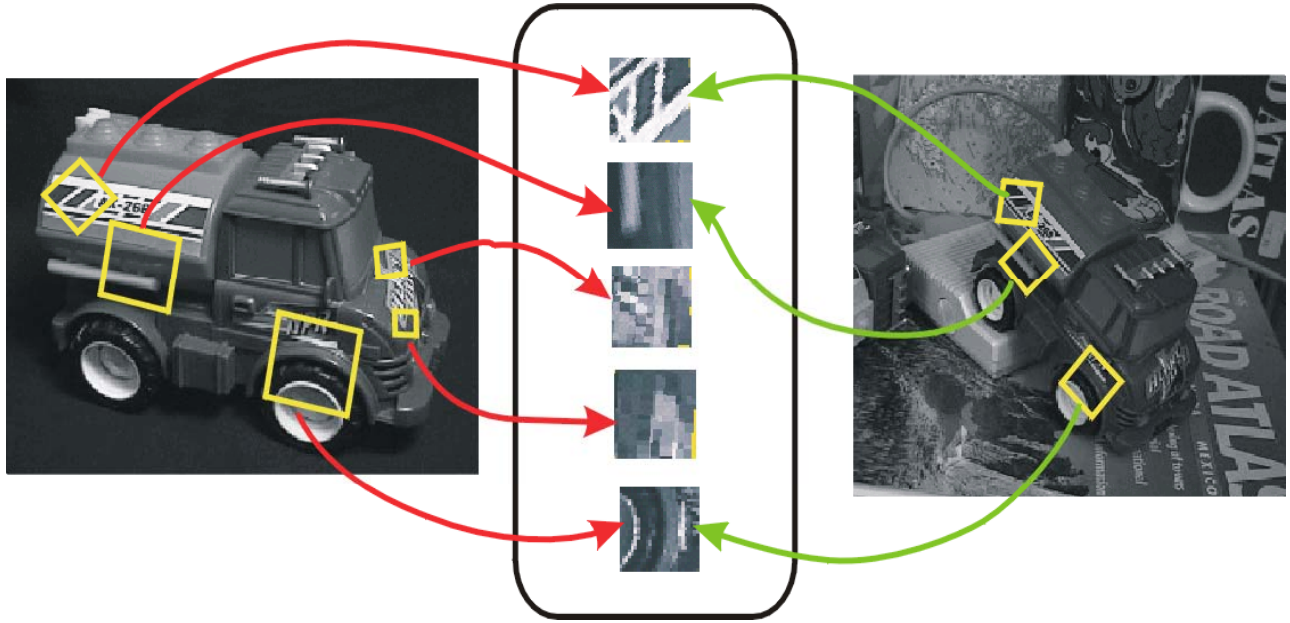


Figure: David Lowe

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?

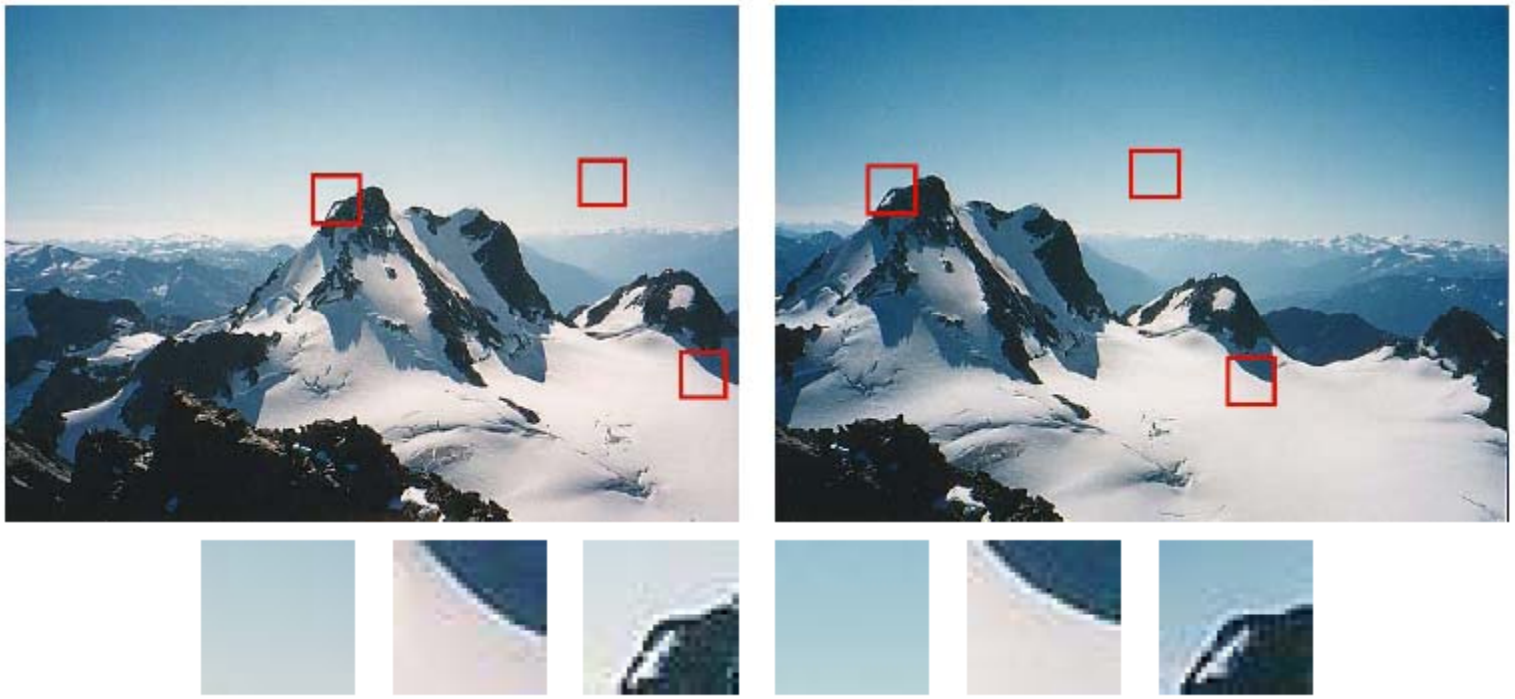
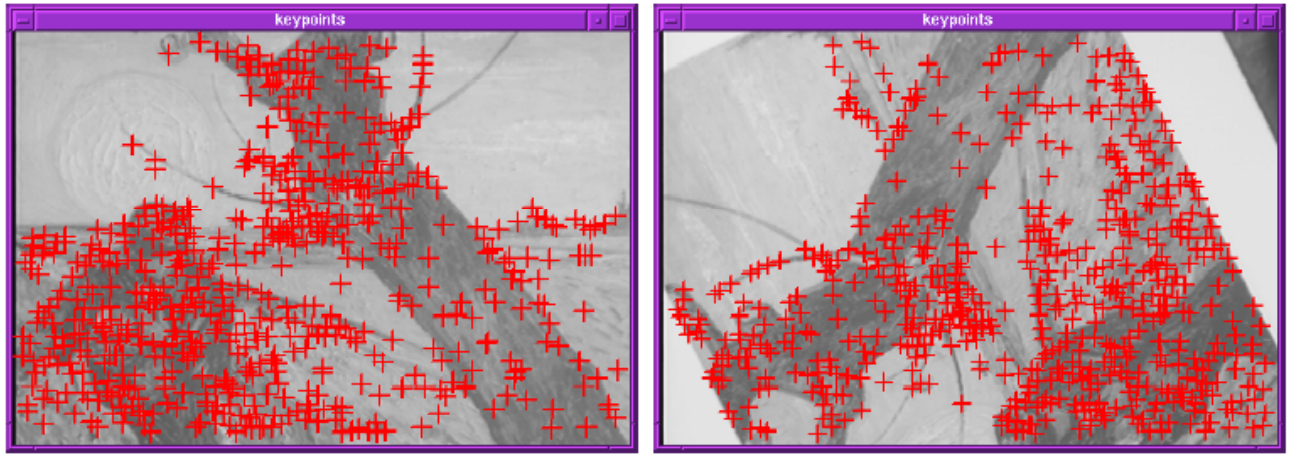


Figure 4.3: *Image pairs with extracted patches below. Notice how some patches can be localized or matched with higher accuracy than others.*

Finding Corners



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

Corners are repeatable and **distinctive**

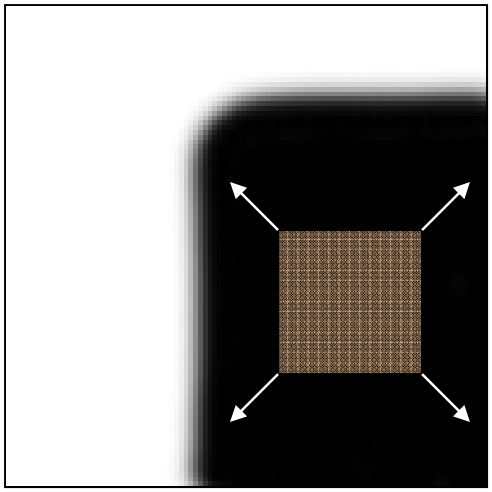
C.Harris and M.Stephens. "A Combined Corner and Edge Detector."
Proceedings of the 4th Alvey Vision Conference: pages 147--151.

Source: Lana Lazebnik

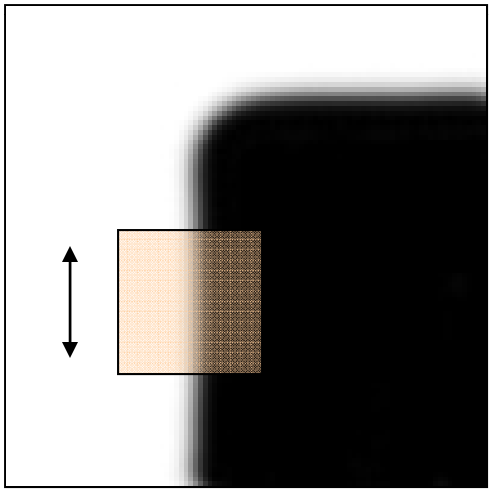
Corners as distinctive interest points

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

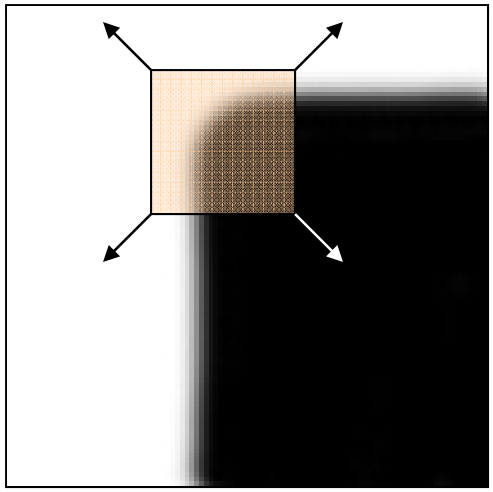
Shifting a window in *any direction* should give *a large change* in intensity



"flat" region: no change in all directions



"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*]:

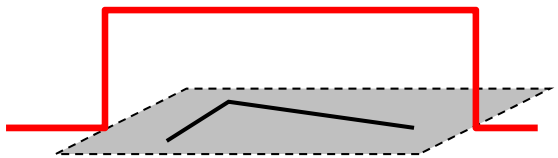
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

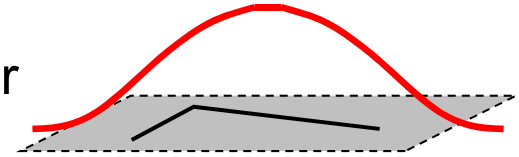
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

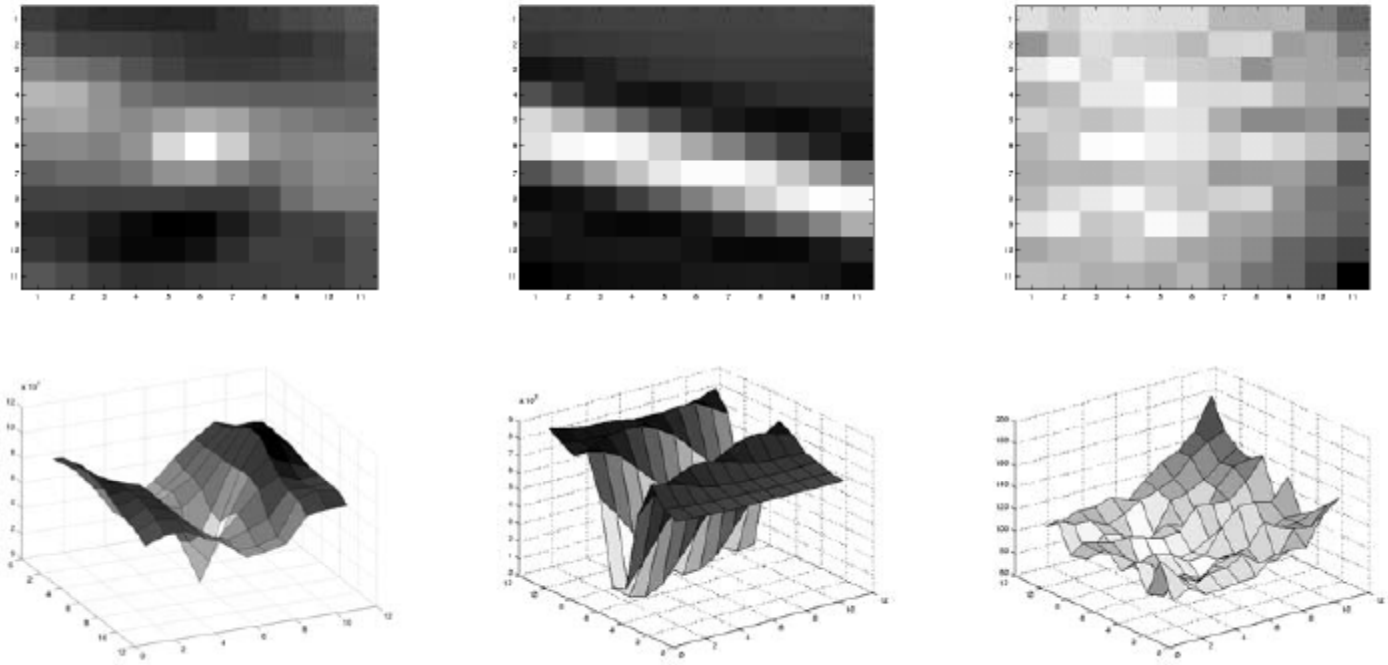
or



Gaussian

Source: R. Szeliski

(a)



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:

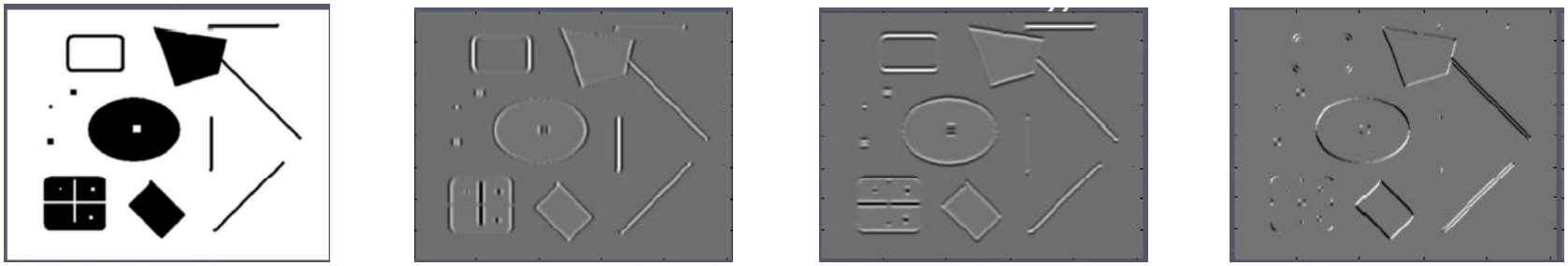
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

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:

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

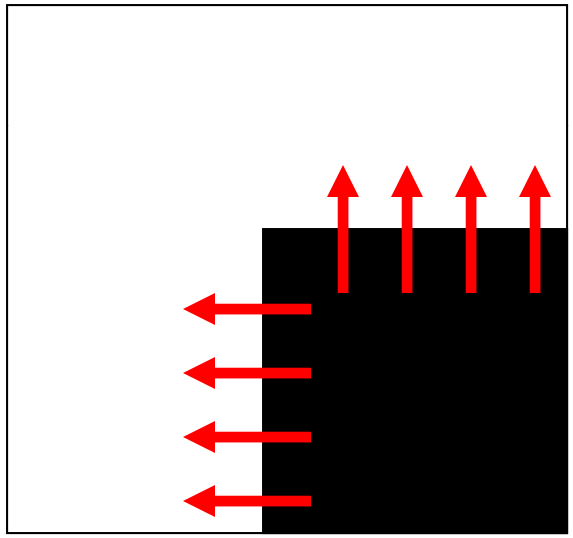
Gradient with respect to x, times gradient with respect to y

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.

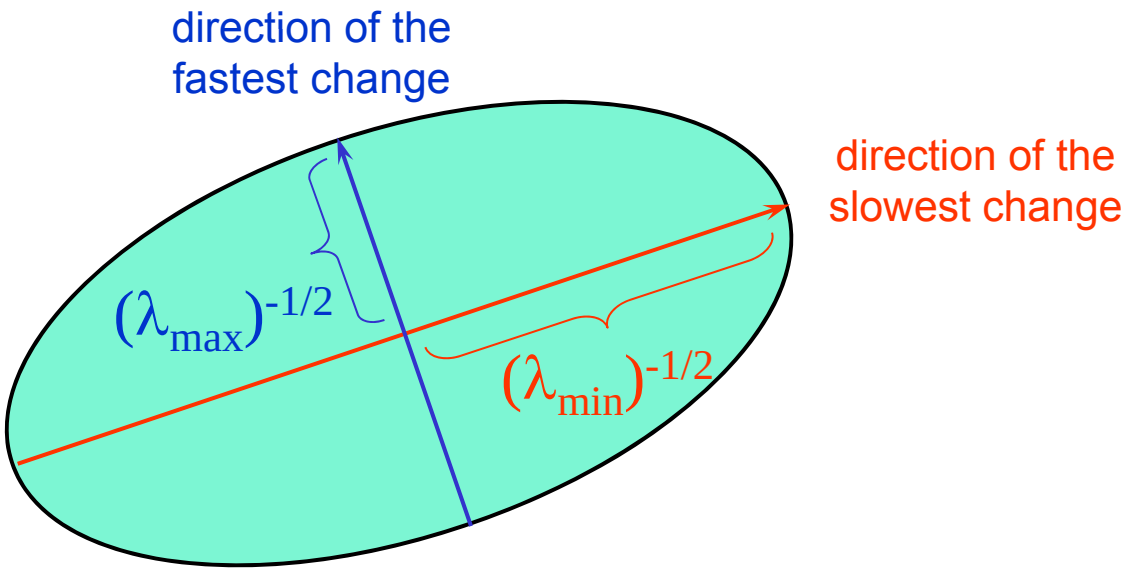
What if we have a corner that is not aligned with the image axes?

Slide credit: David Jacobs

General Case

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

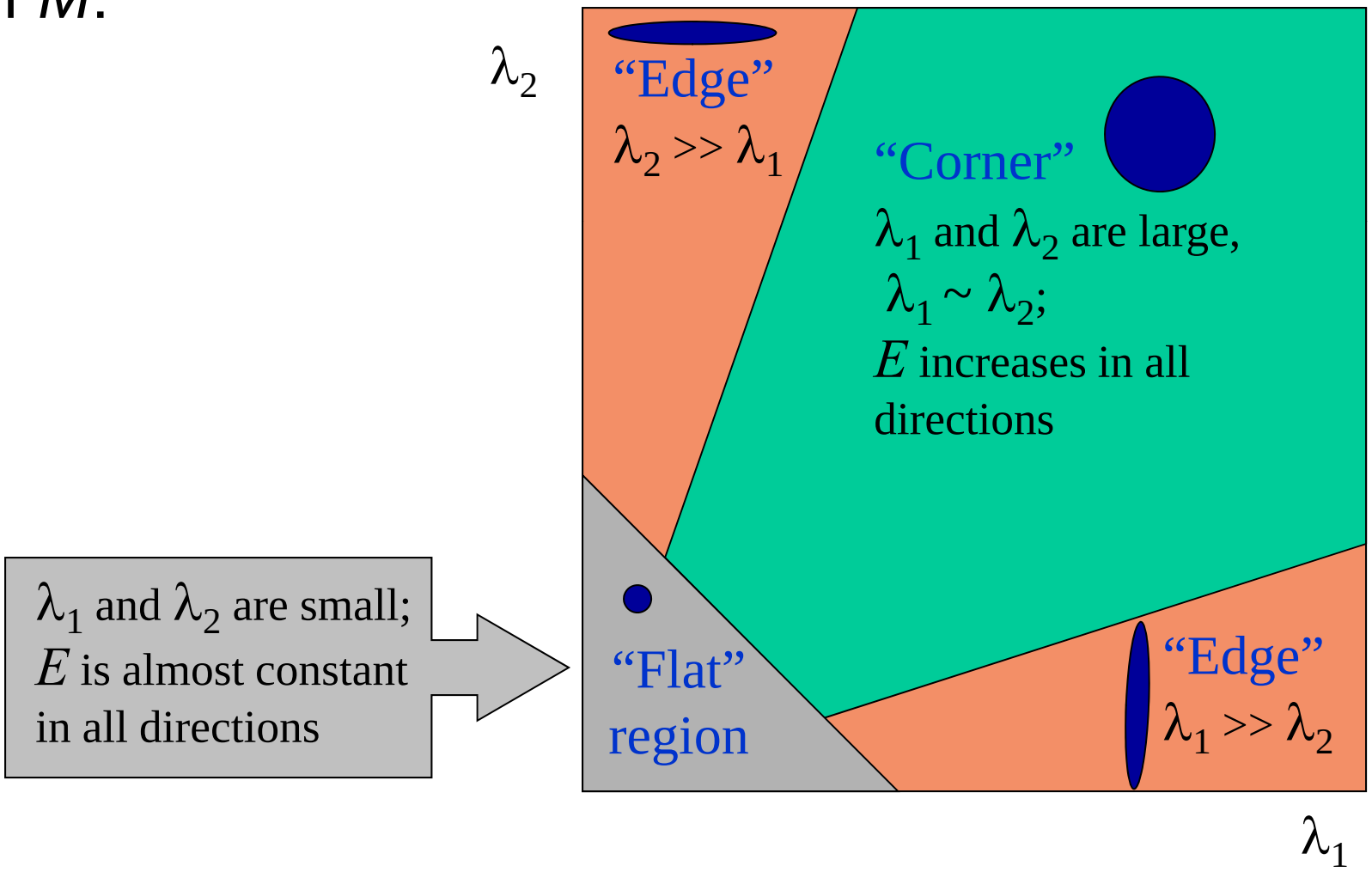
We can visualize *M* as an ellipse with axis lengths determined by the eigenvalues and orientation determined by *R*



Slide adapted form Darya Frolova, Denis Simakov.

Interpreting the eigenvalues

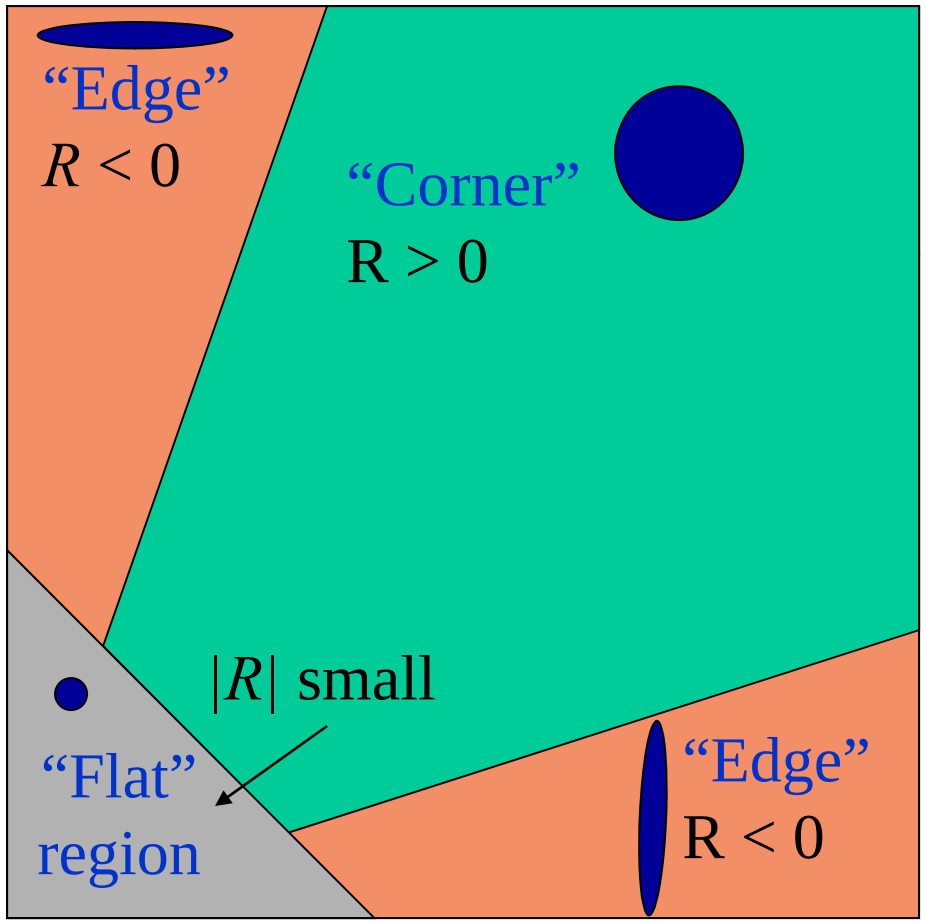
Classification of image points using eigenvalues of *M*:



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

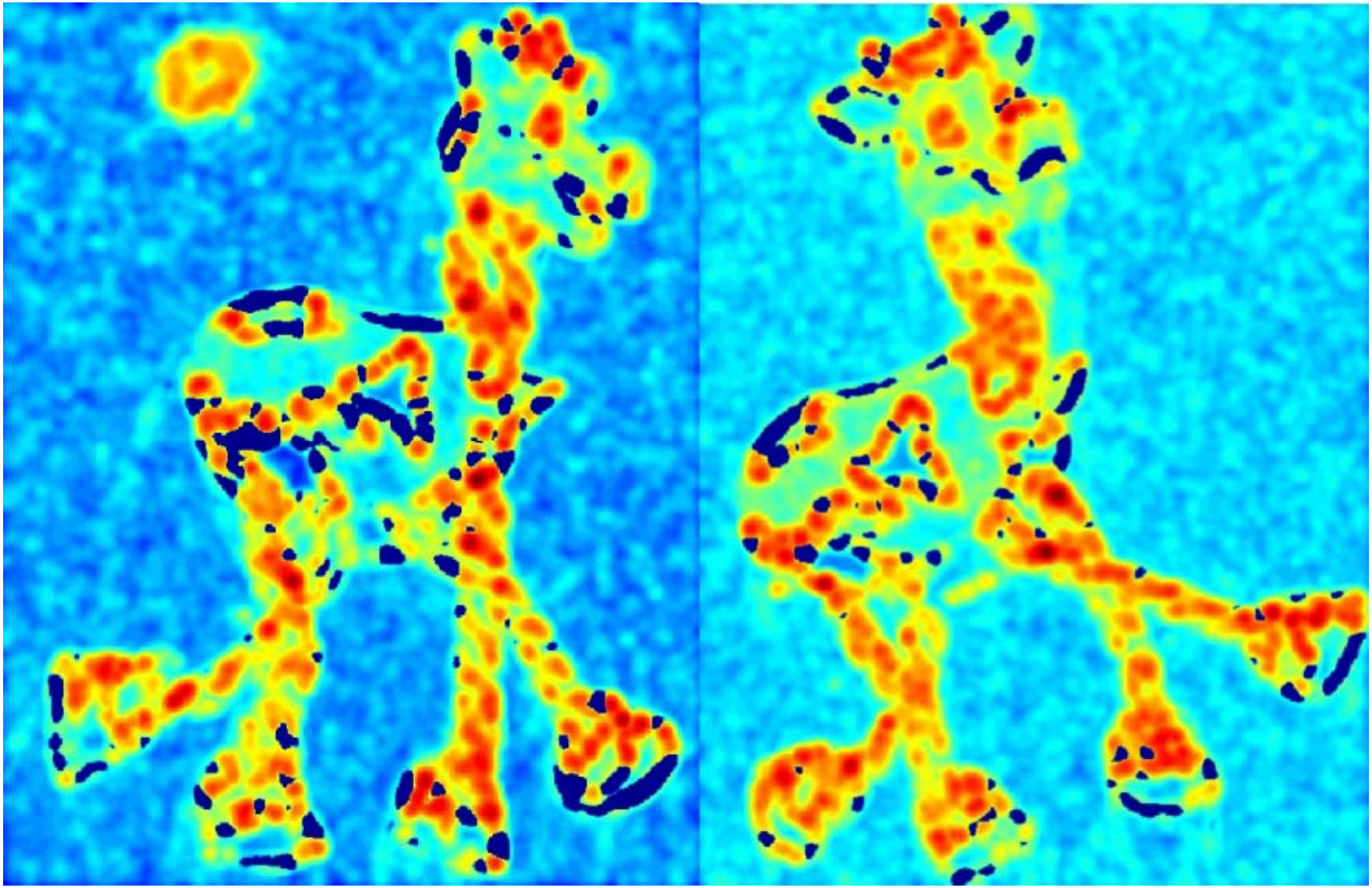
    (R > threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

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

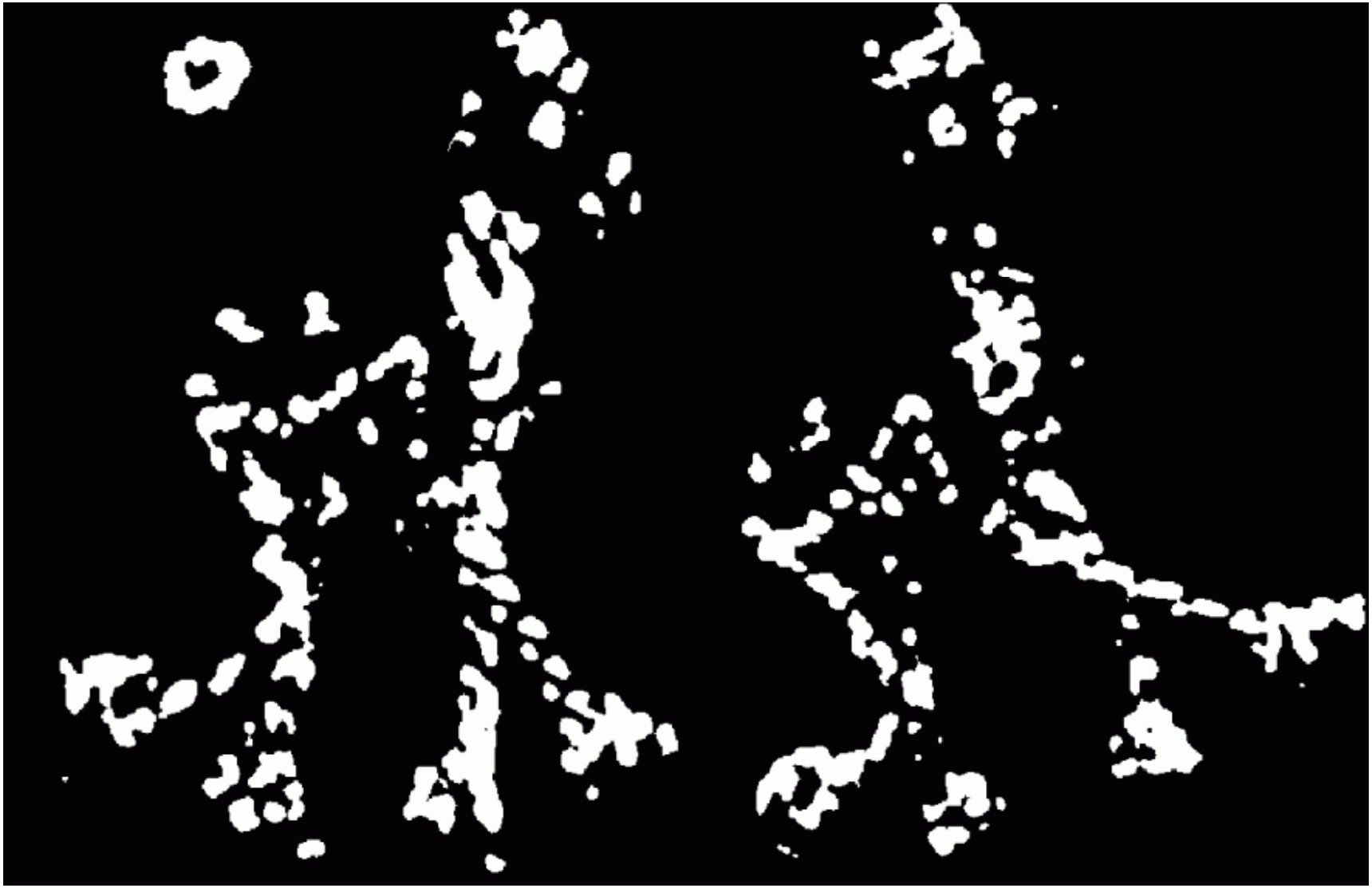
Harris Detector: Workflow

Compute corner response R



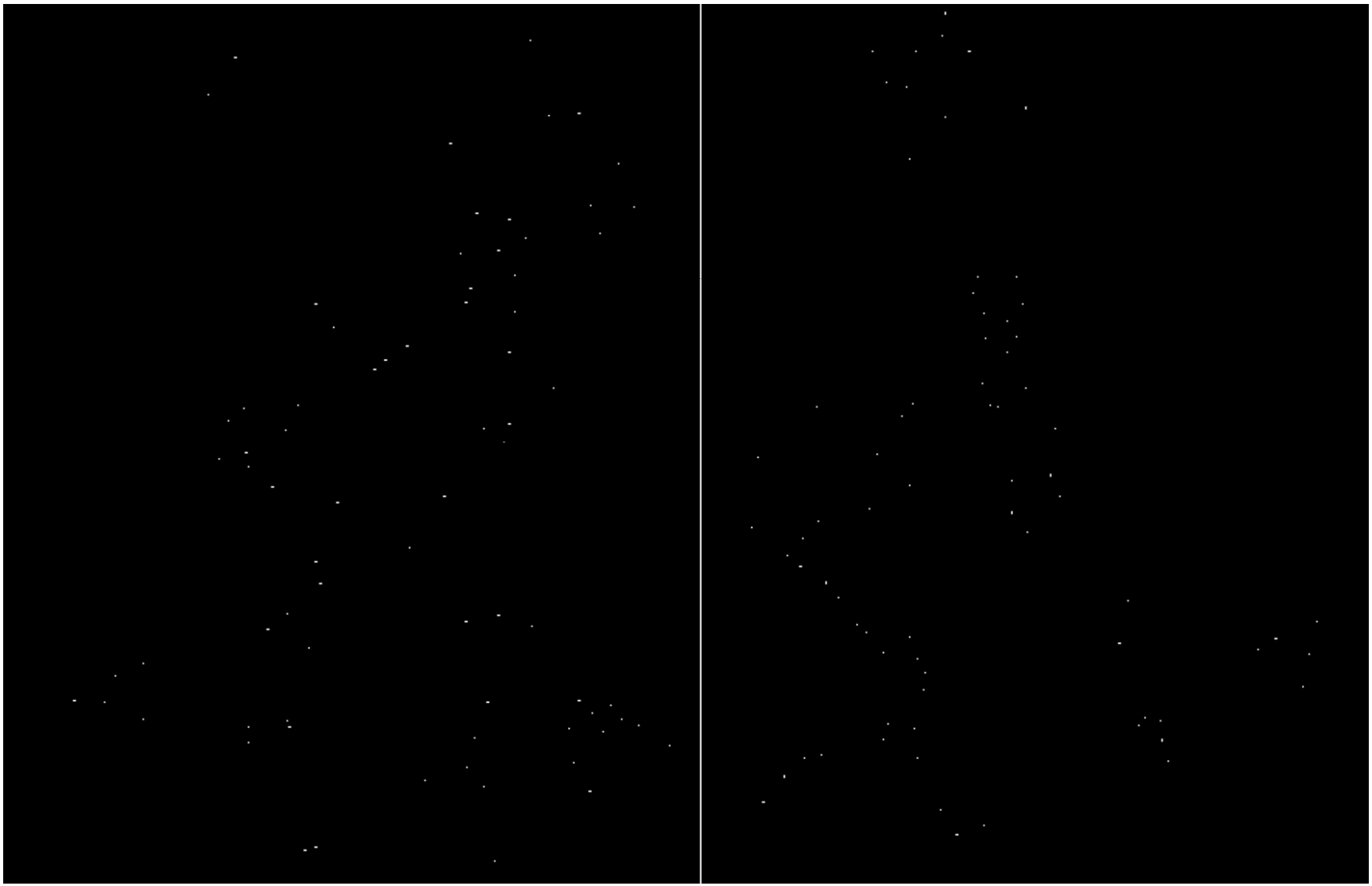
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

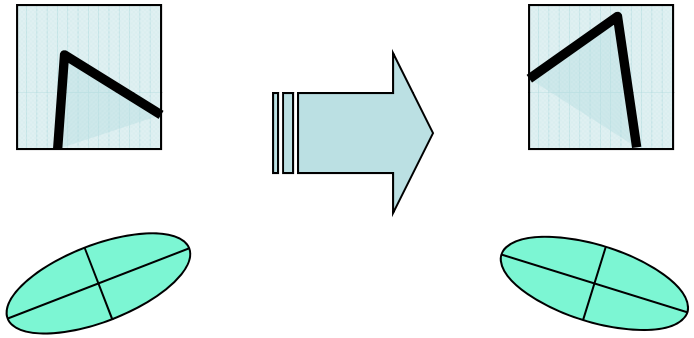
Take only the points of local maxima of R



Harris Detector: Workflow

Harris Detector: Properties

- Rotation invariance

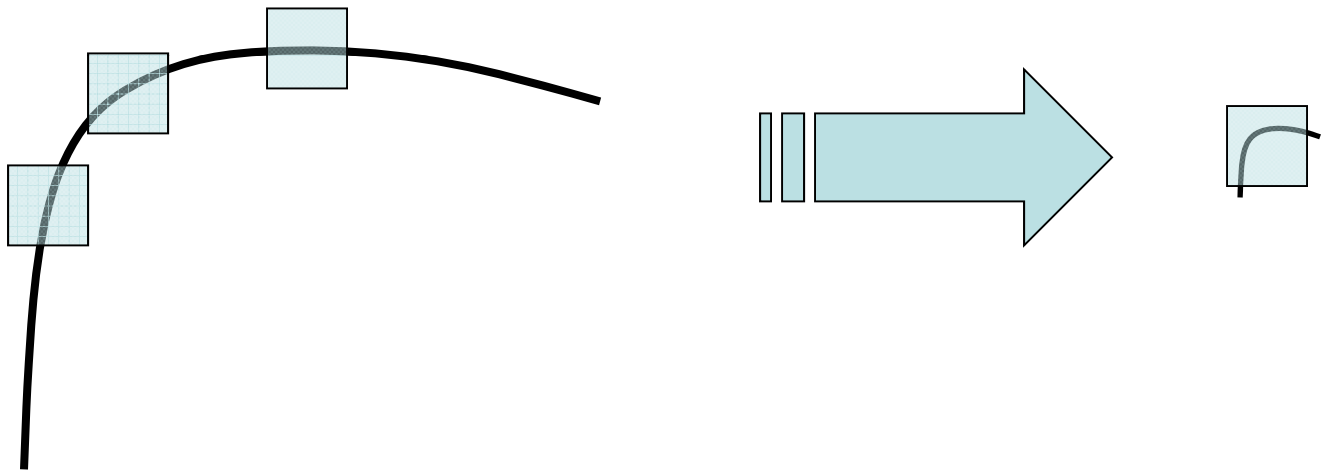


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

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?

How to cope with transformations?

- Exhaustive search
- Invariance
- Robustness

Exhaustive search

- Multi-scale approach

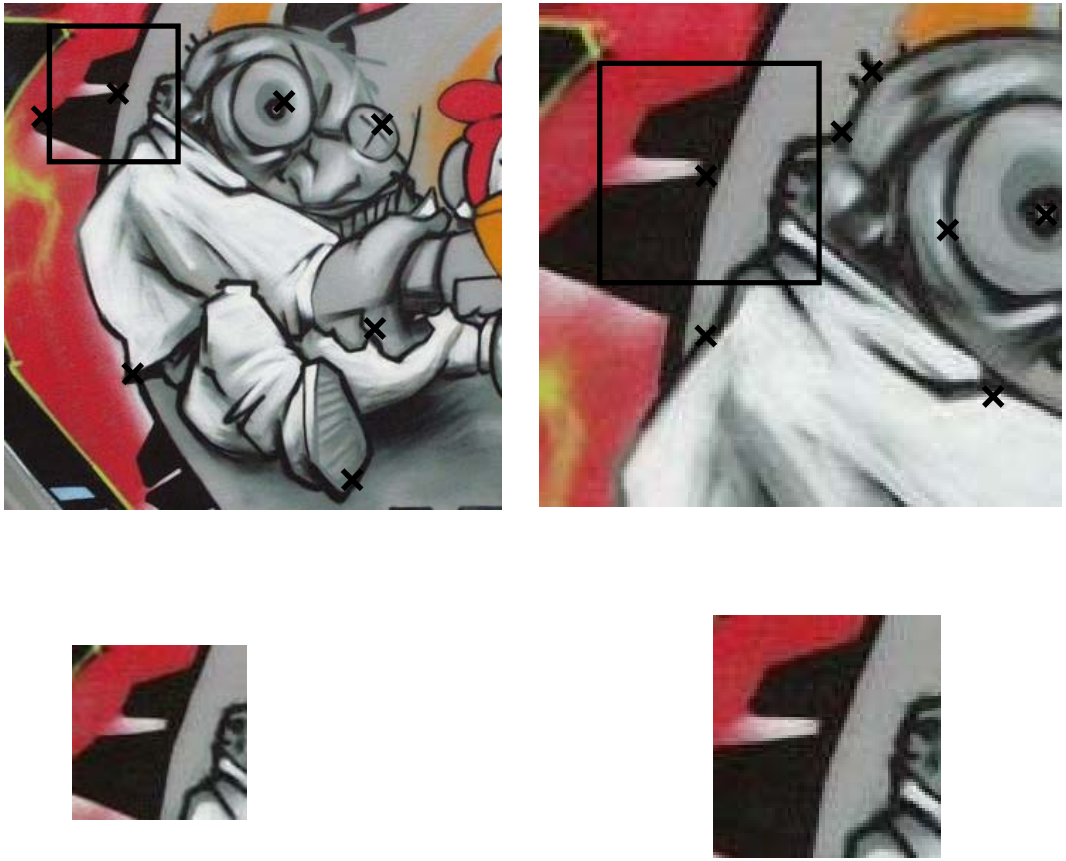
Slide from T. Tuytelaars ECCV 2006 tutorial

Exhaustive search

- Multi-scale approach

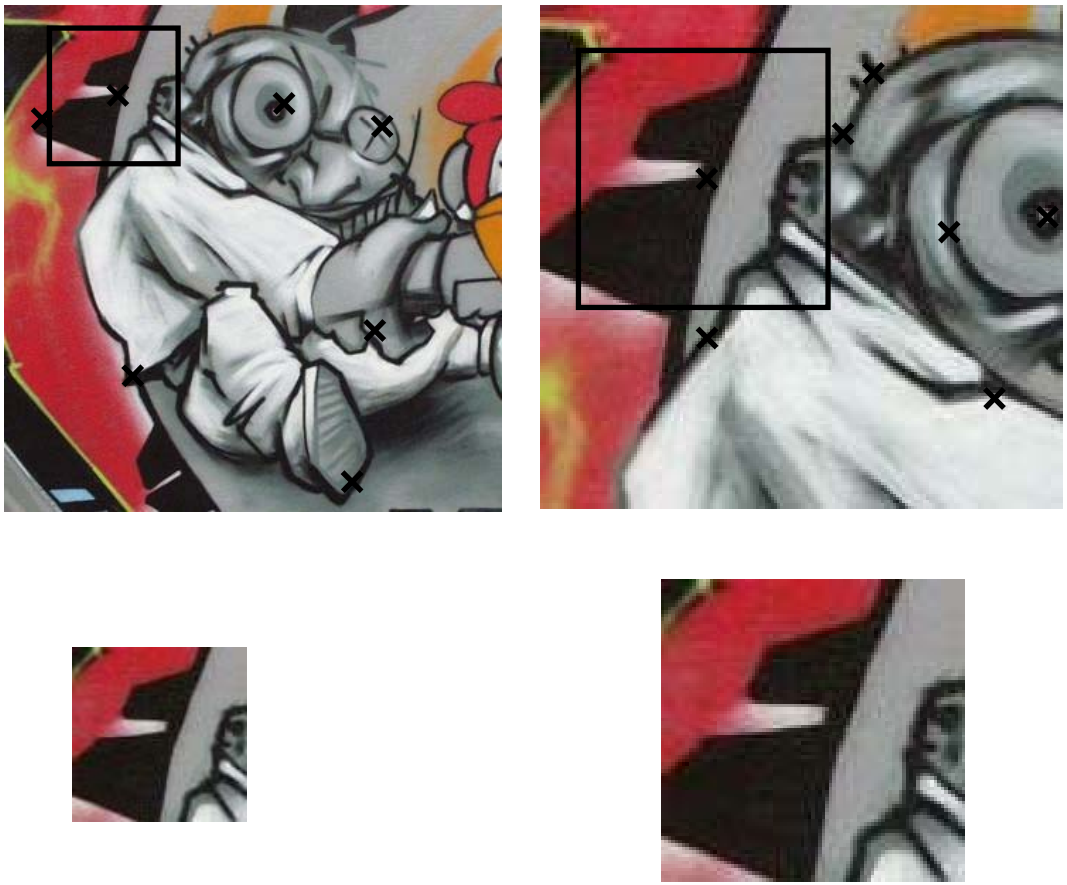
Exhaustive search

- Multi-scale approach



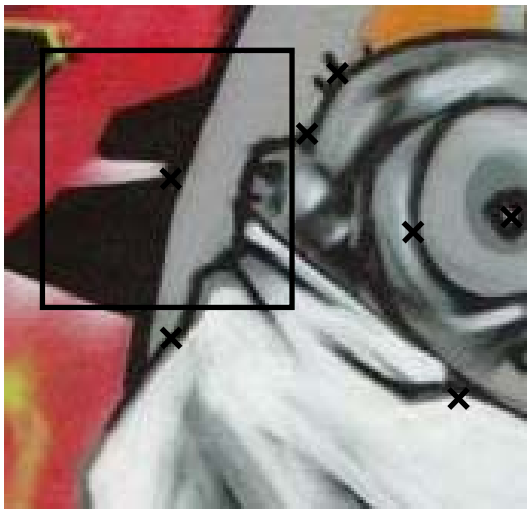
Exhaustive search

- Multi-scale approach



Invariance

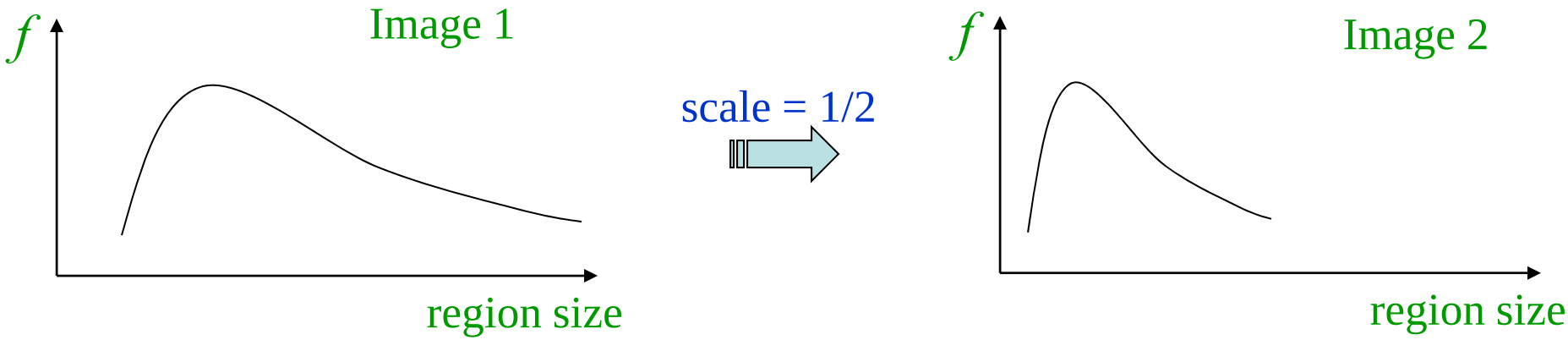
- Extract patch from each image individually



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)



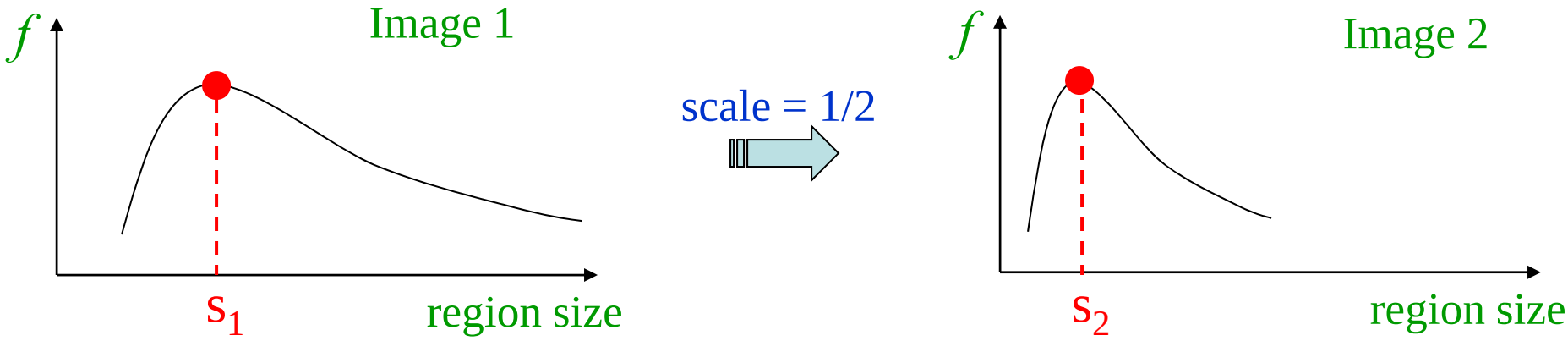
Automatic scale selection

- 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!



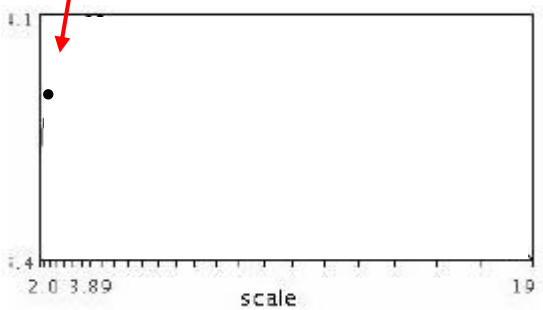
Automatic Scale Selection

$$f(I_{i_1 \ldots i_m}(x,\sigma)) \;=\; f(I_{i_1 \ldots i_m}(x',\sigma'))$$

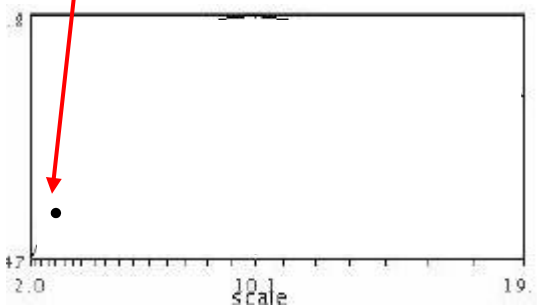
Same operator responses if the patch contains the same image up to scale factor.

Automatic Scale Selection

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



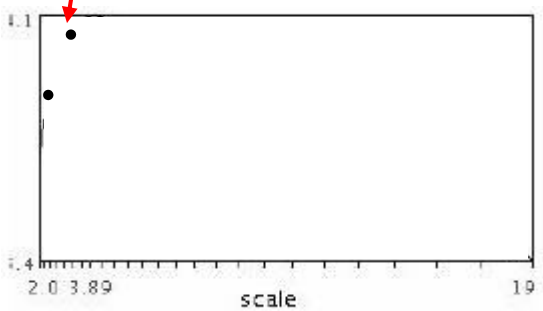
$f(I_{i_1 \ldots i_m}(x, \sigma))$



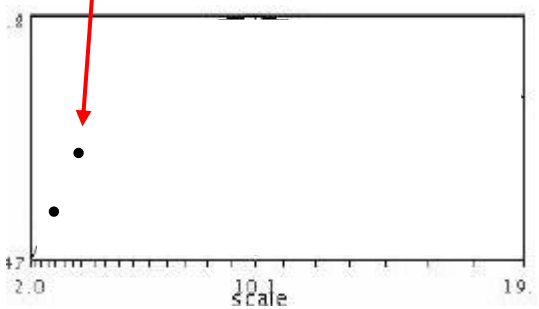
$f(I_{i_1 \ldots i_m}(x', \sigma))$

Automatic Scale Selection

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



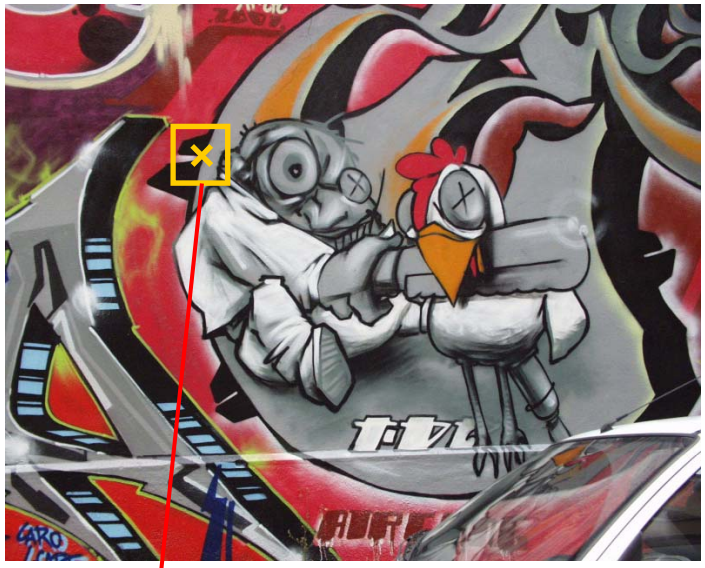
$f(I_{i_1 \ldots i_m}(x,\sigma))$

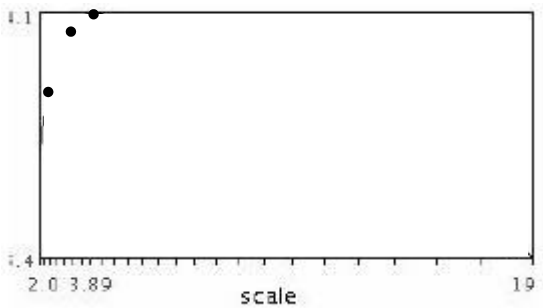


$f(I_{i_1 \ldots i_m}(x',\sigma))$

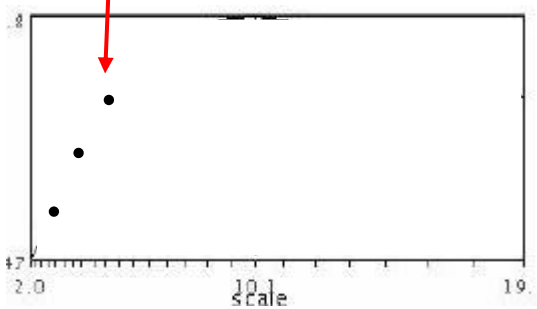
Automatic Scale Selection

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





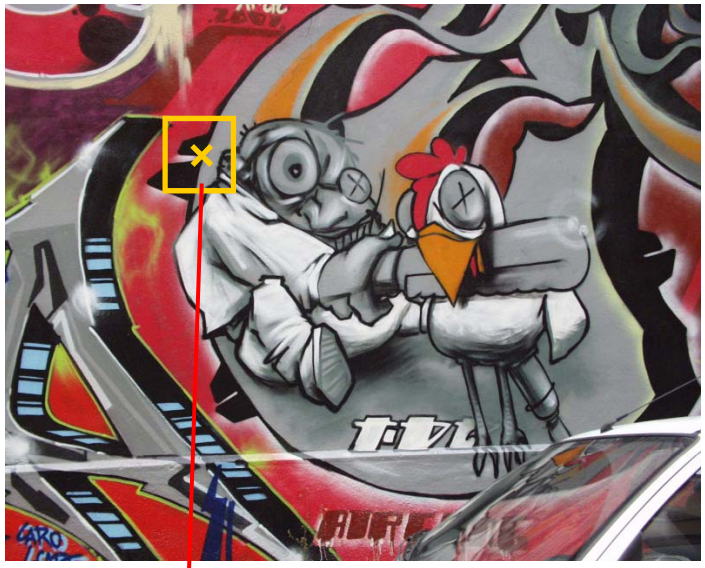
$f(I_{i_1 \ldots i_m}(x, \sigma))$

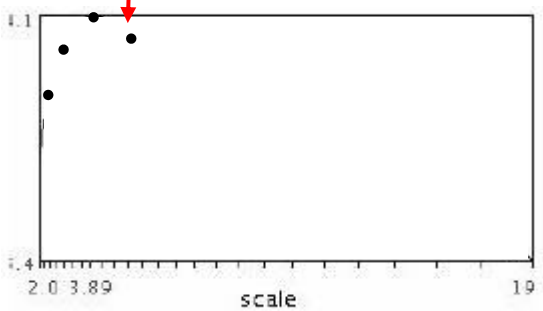


$f(I_{i_1 \ldots i_m}(x', \sigma))$

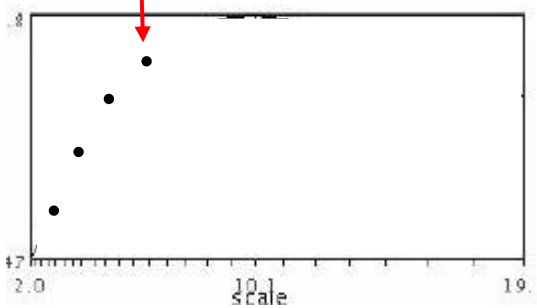
Automatic Scale Selection

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





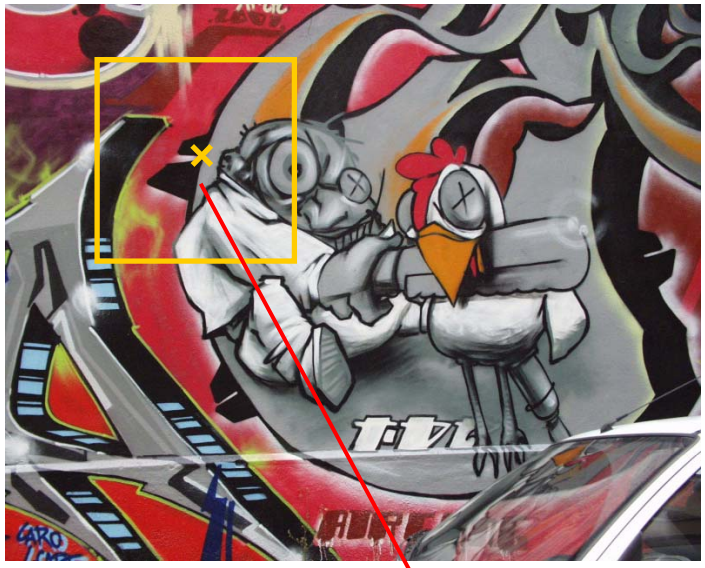
$f(I_{i_1 \ldots i_m}(x,\sigma))$

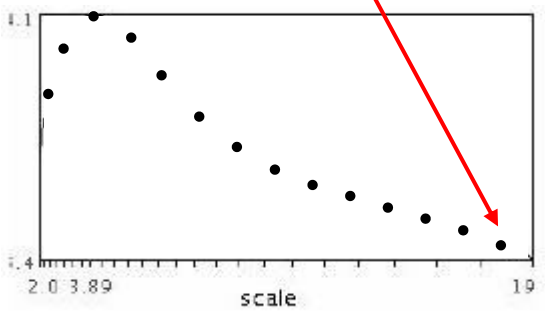


$f(I_{i_1 \ldots i_m}(x',\sigma))$

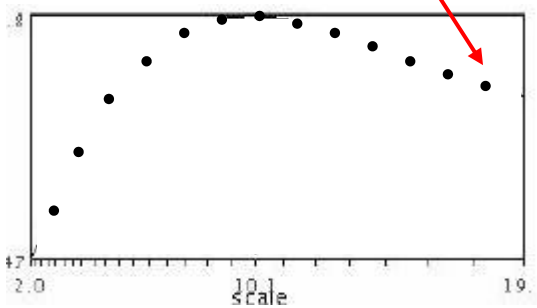
Automatic Scale Selection

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





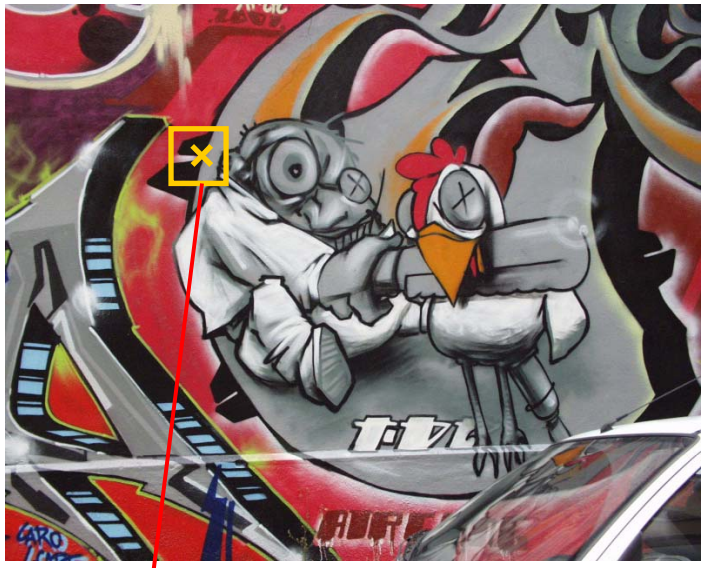
$f(I_{i_1 \ldots i_m}(x,\sigma))$

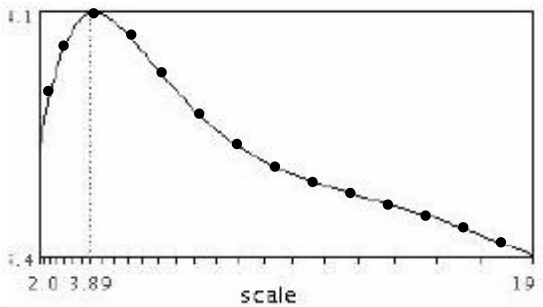


$f(I_{i_1 \ldots i_m}(x',\sigma))$

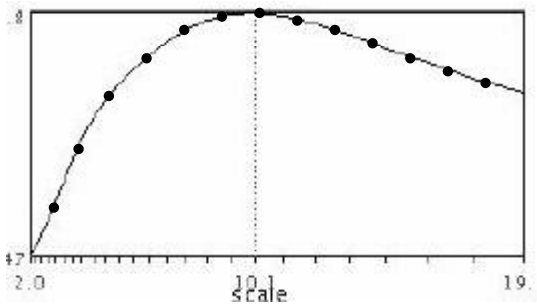
Automatic Scale Selection

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





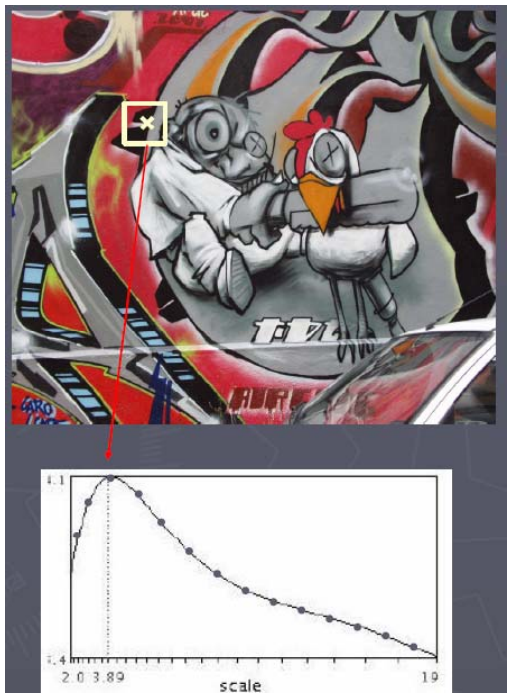
$f(I_{i_1 \ldots i_m}(x, \sigma))$

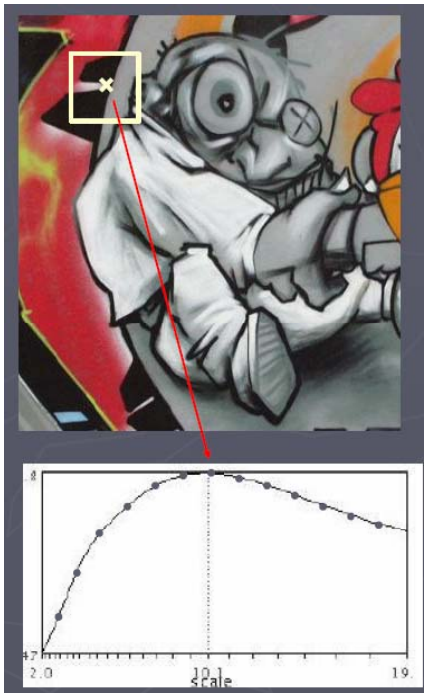


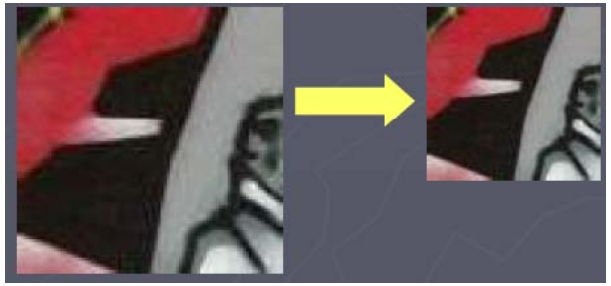
$f(I_{i_1 \ldots i_m}(x', \sigma'))$

Scale selection

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



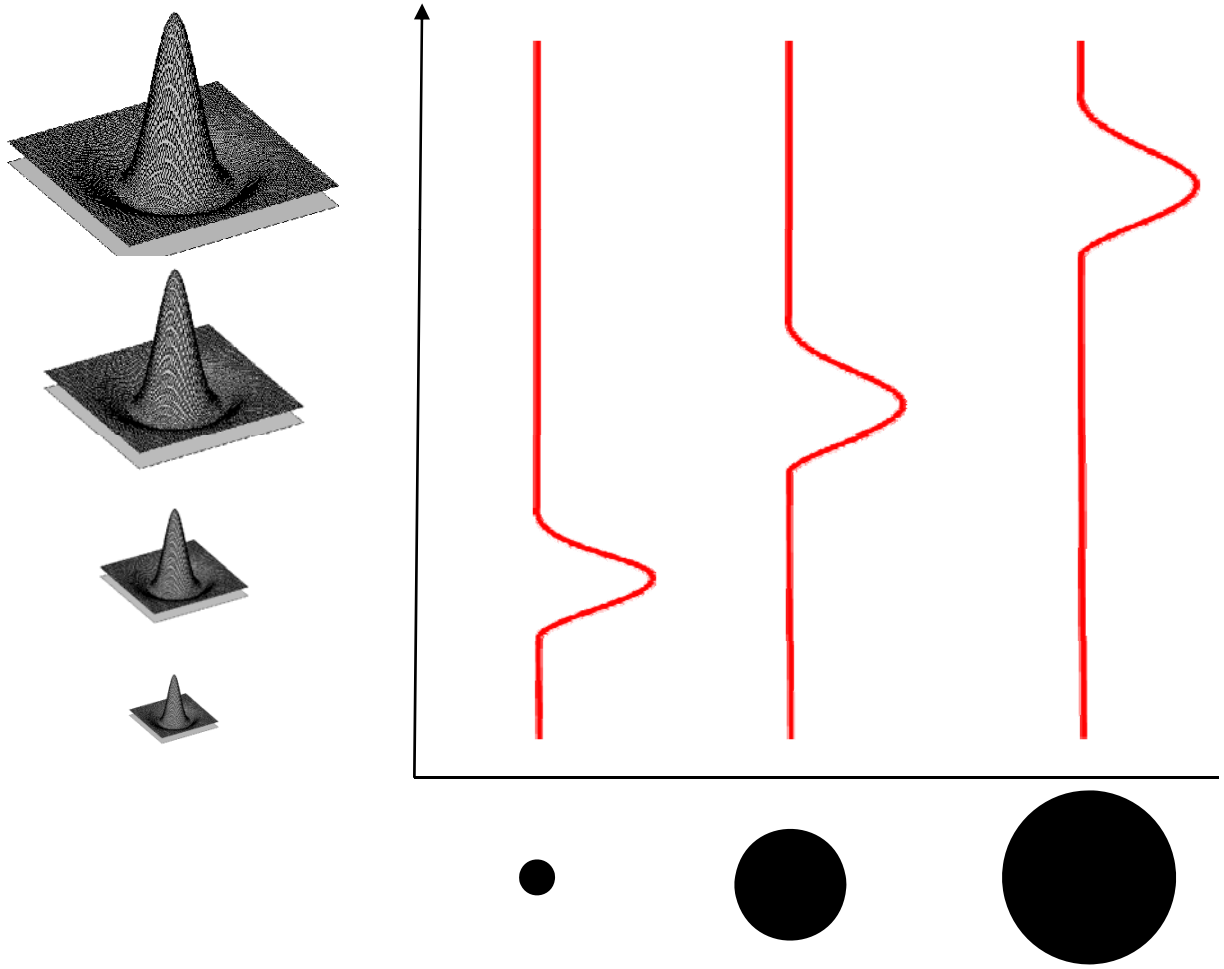




[Images from T. Tuytelaars]

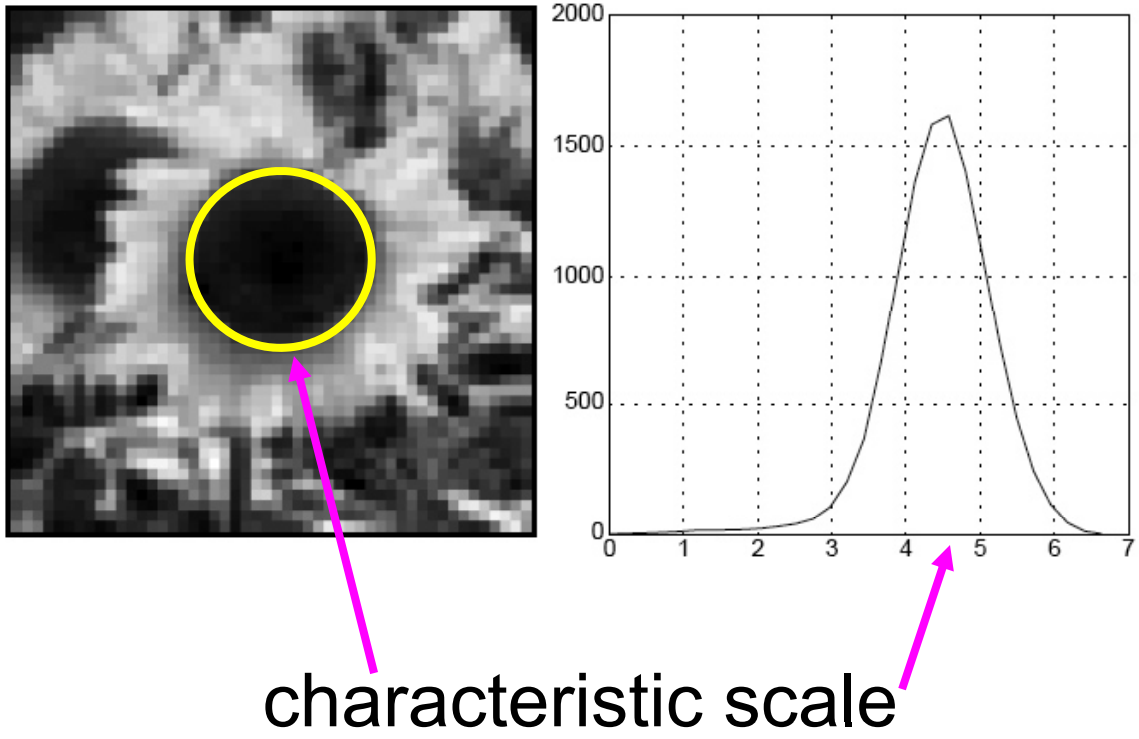
What Is A Useful Signature Function?

- **Laplacian-of-Gaussian = "blob" detector**



Characteristic scale

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



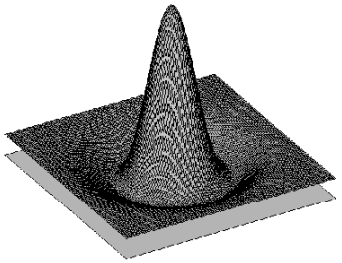
T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* **30** (2): pp 77--116.

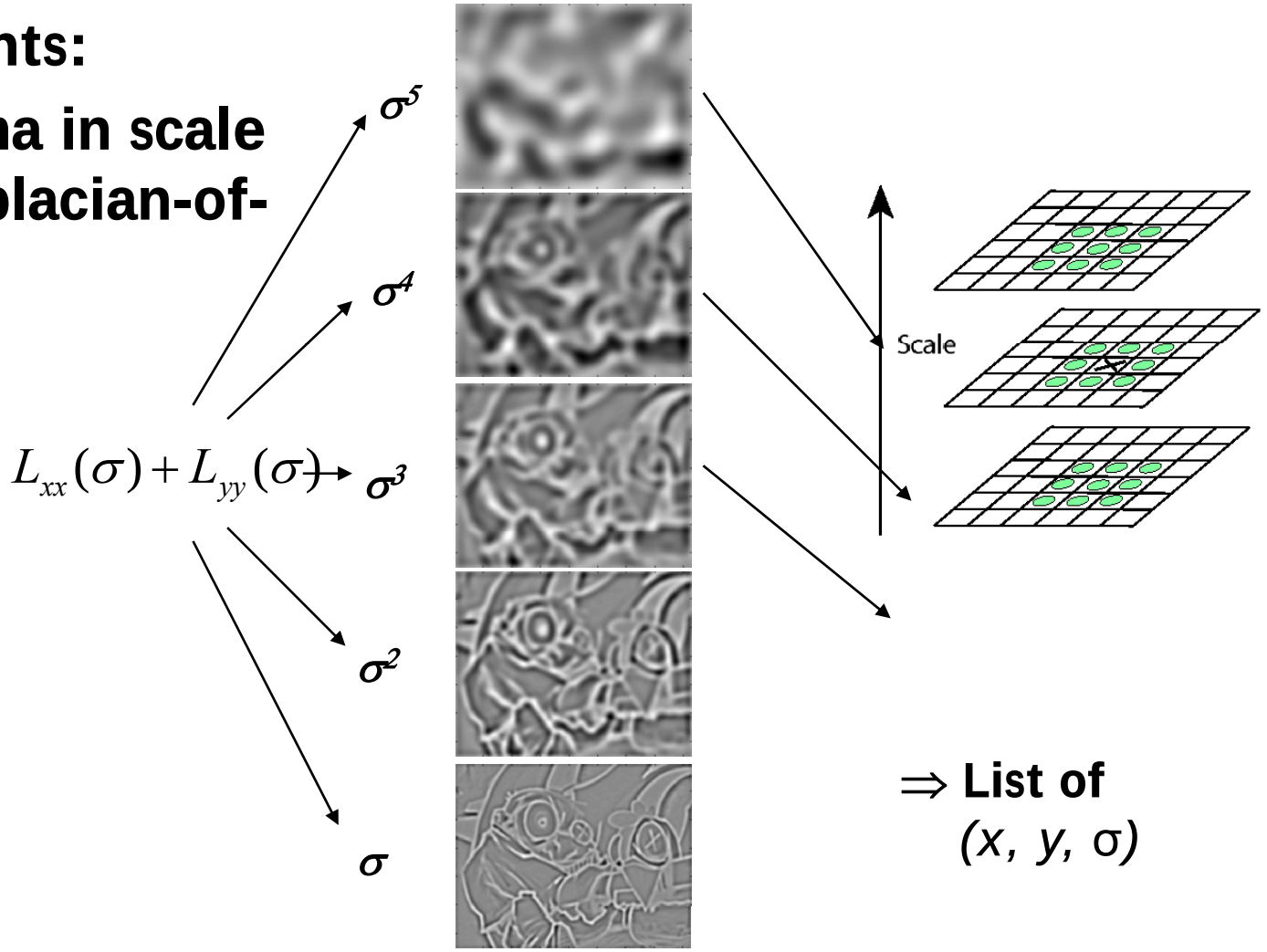
Source: Lana Lazebnik

Laplacian-of-Gaussian (LoG)

- **Interest points:**

  Local maxima in scale space of Laplacian-of-Gaussian





⇒ **List of** (x, y, σ)

Scale-space blob detector: Example

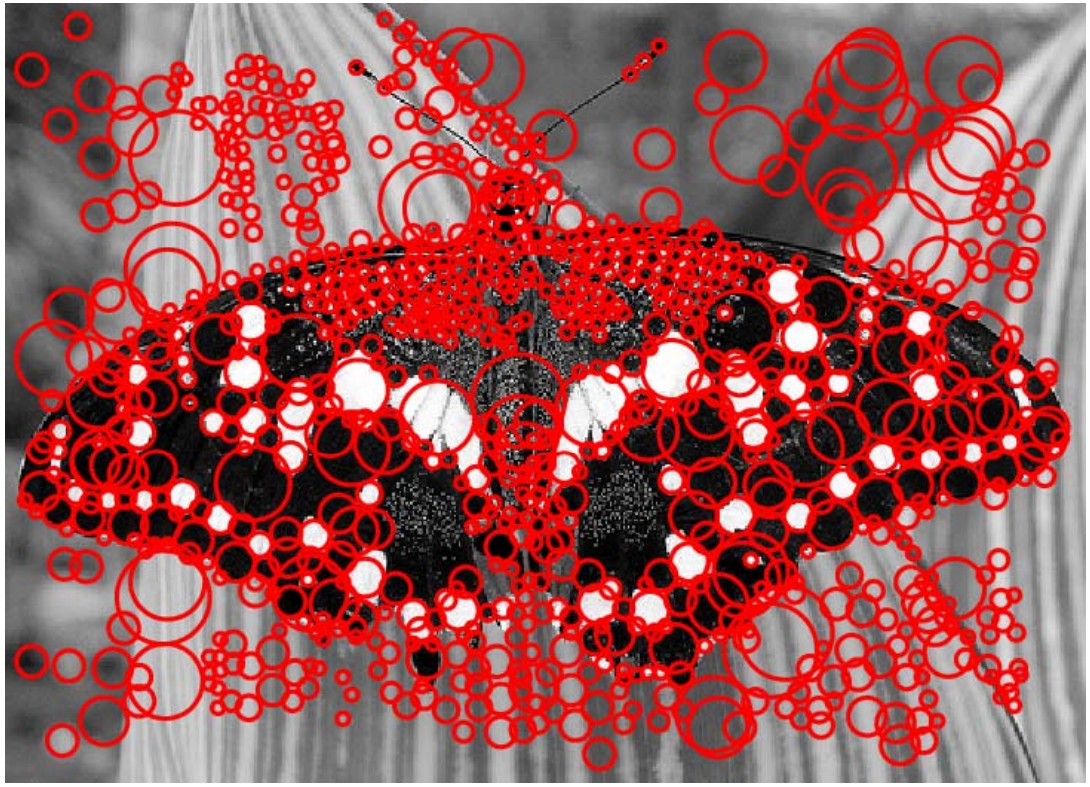
Source: Lana Lazebnik

Scale-space blob detector: Example

sigma = 11.9912

Source: Lana Lazebnik

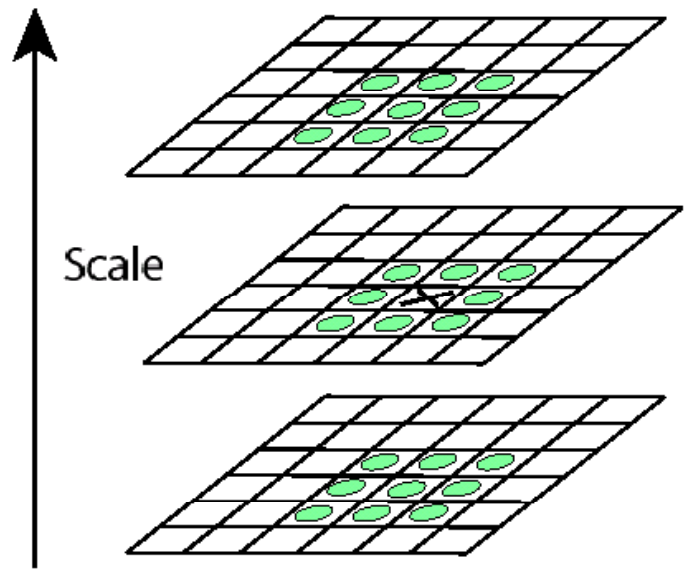
Scale-space blob detector: Example



Source: Lana Lazebnik

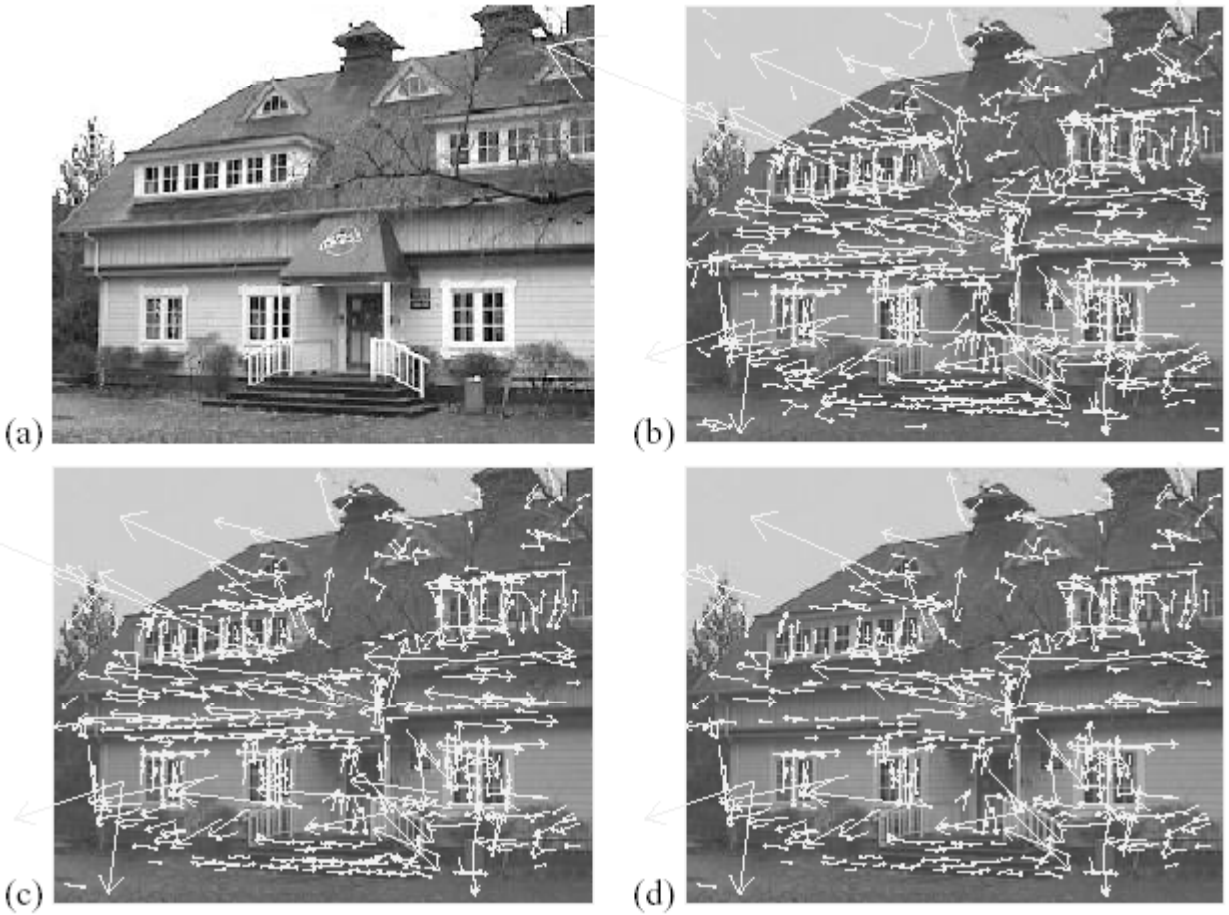
Key point localization with DoG

- 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)

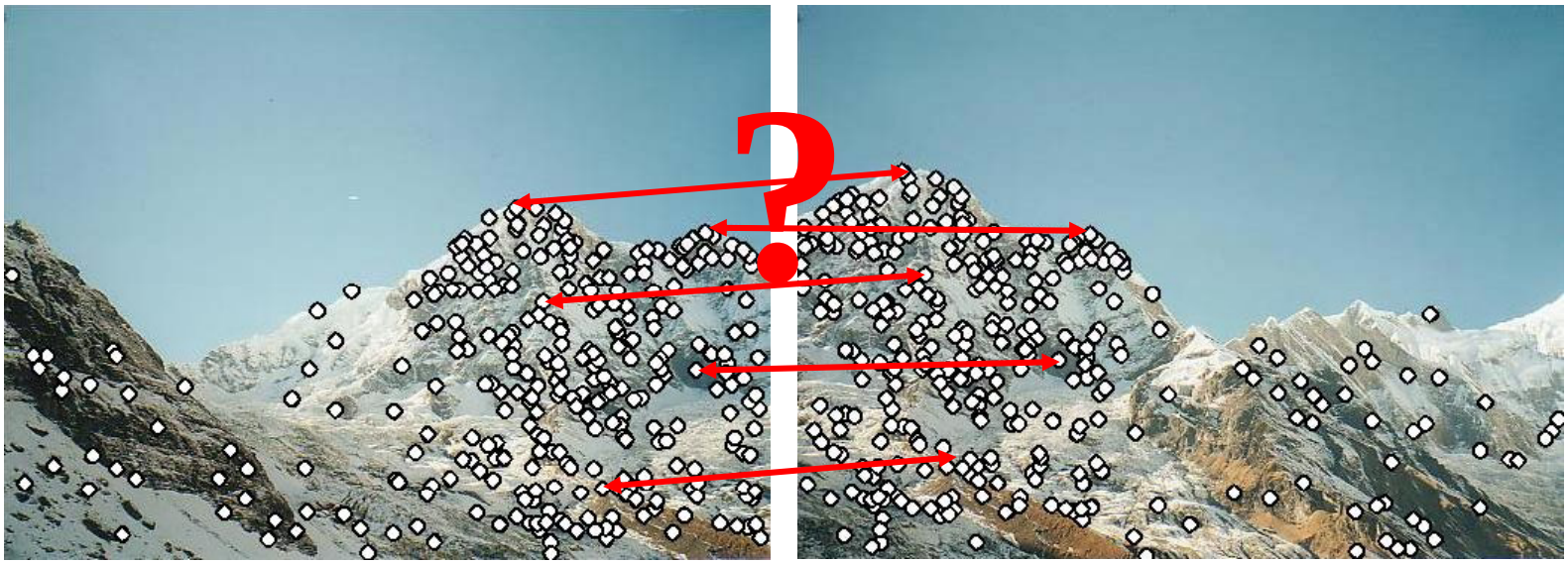
Main questions

- 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:

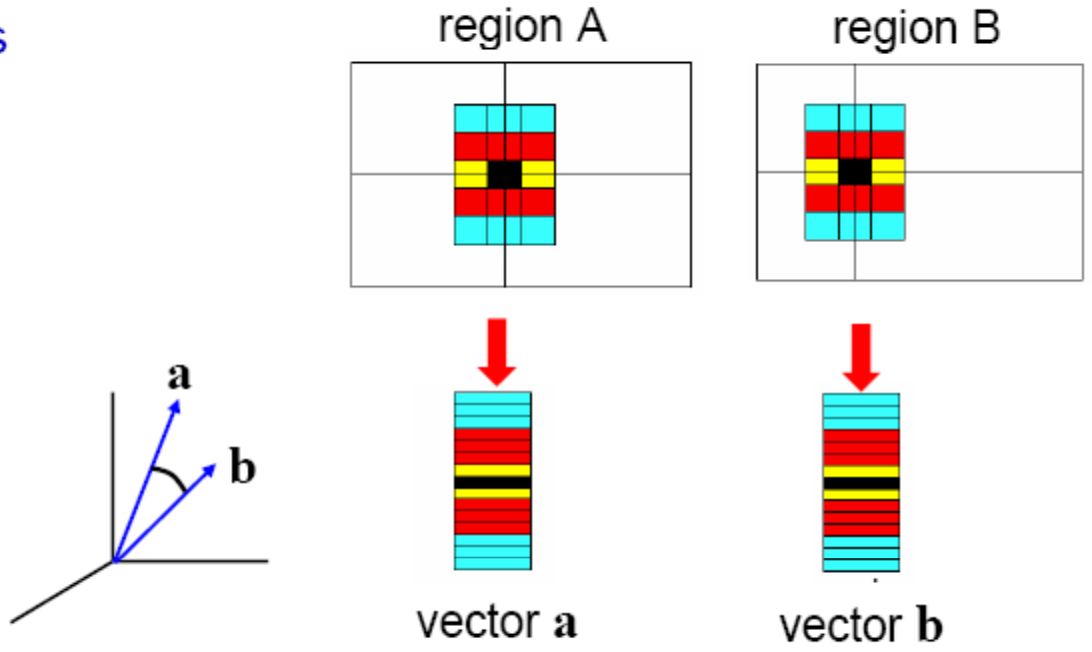
1. Invariant
2. Distinctive

Local descriptors

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

Write regions as vectors

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

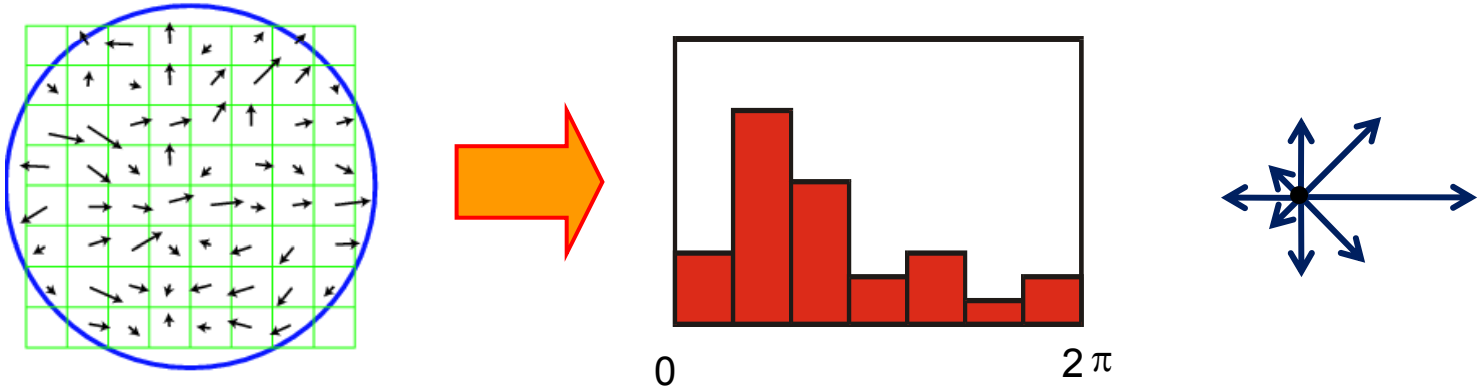


Feature descriptors

Disadvantage of patches as descriptors:

- Small shifts can affect matching score a lot

Solution: histograms



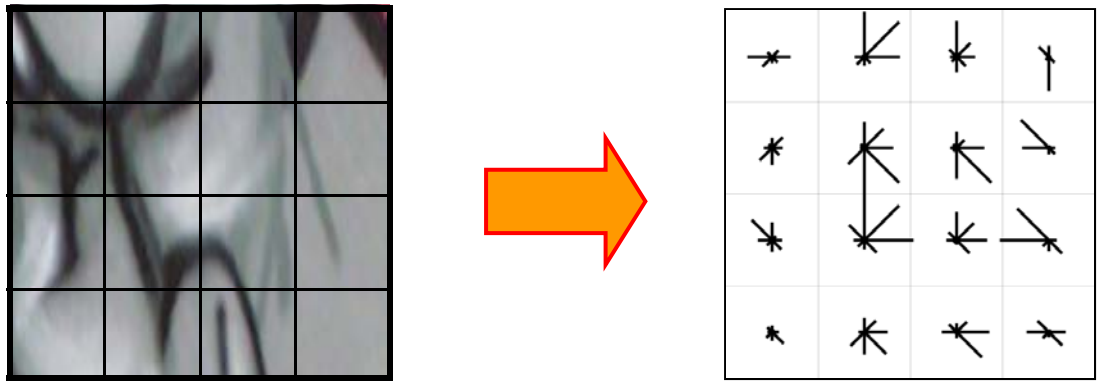
Source: Lana Lazebnik

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

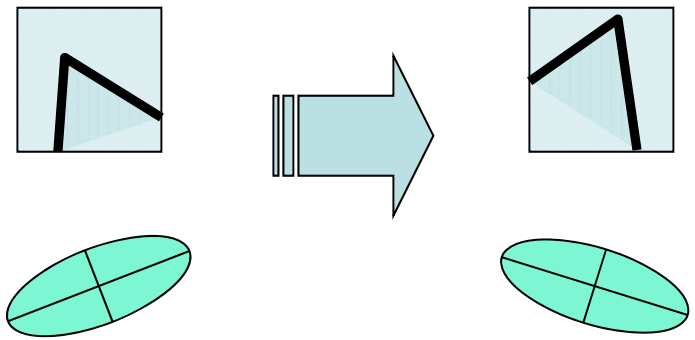


David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Source: Lana Lazebnik

Rotation Invariant Descriptors

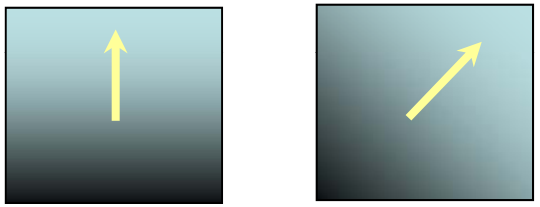
- Harris corner response measure:
  depends only on the eigenvalues of the matrix *M*



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

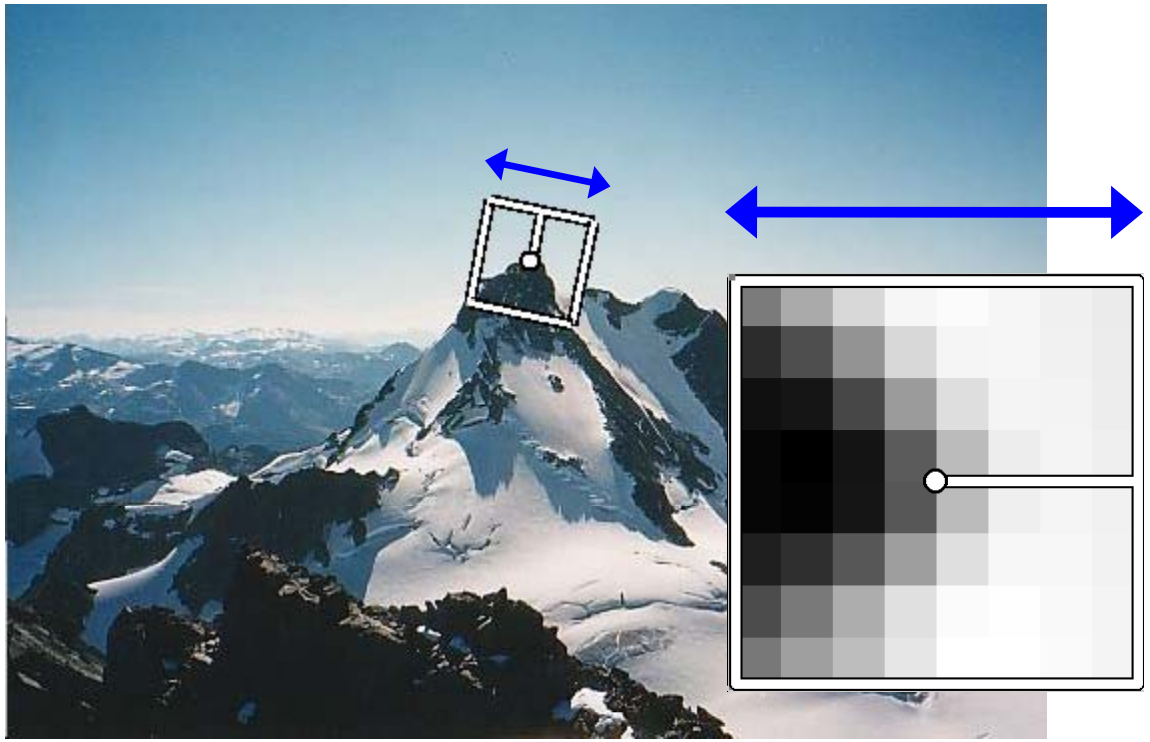
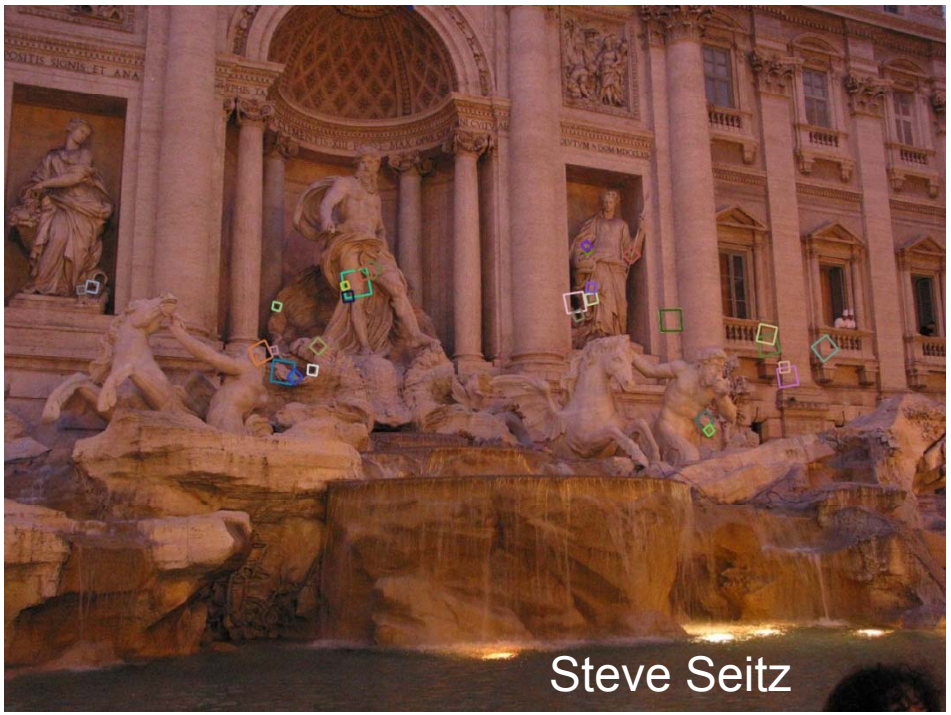


Image from Matthew Brown

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



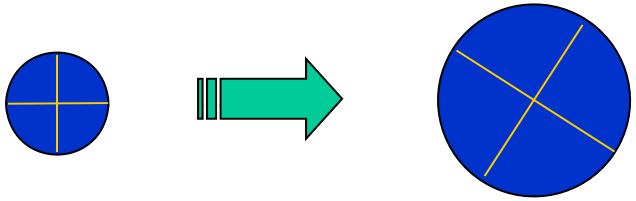
Working with SIFT descriptors

- 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]

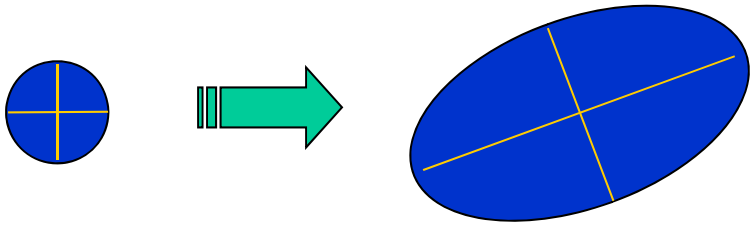
Affine Invariant Detection

(a proxy for invariance to perspective transformations)

- Above we considered:
  Similarity transform (rotation + uniform scale)

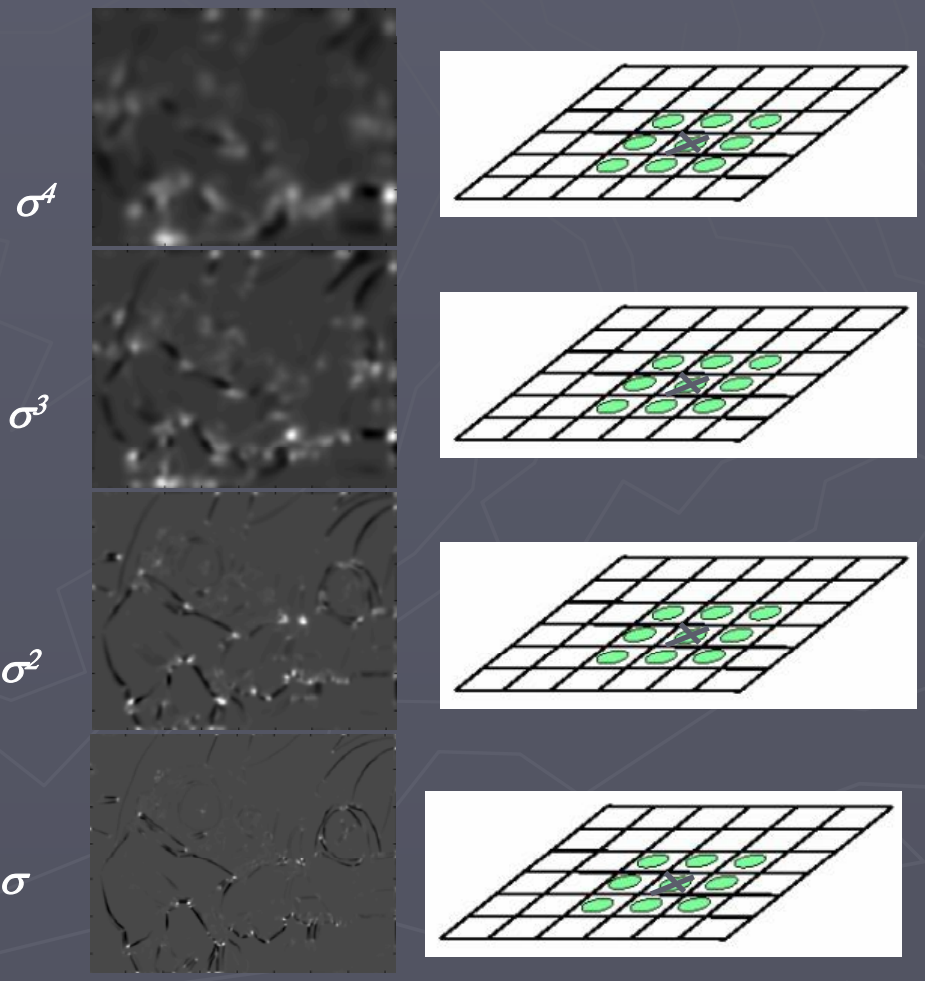


- Now we go on to:
  Affine transform (rotation + non-uniform scale)



Mikolajczyk: Harris Laplace

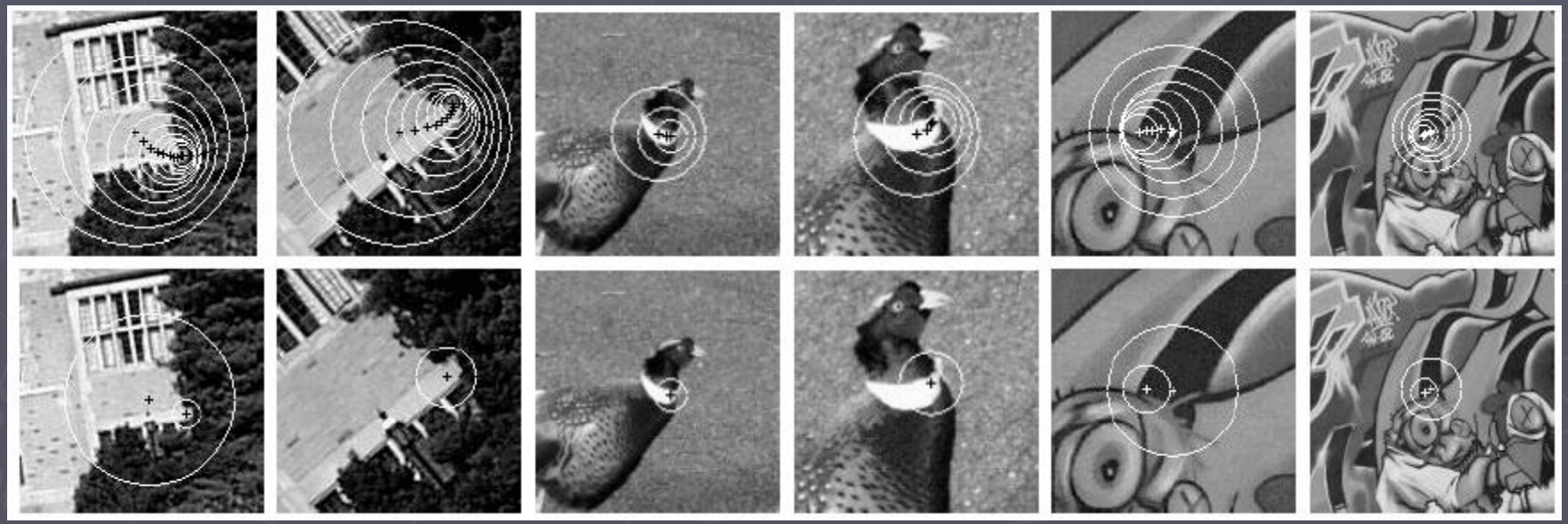
1. Initialization: Multiscale Harris corner detection



Mikolajczyk: Harris Laplace

1. ***Initialization: Multiscale Harris corner detection***
2. ***Scale selection based on Laplacian***

Harris points



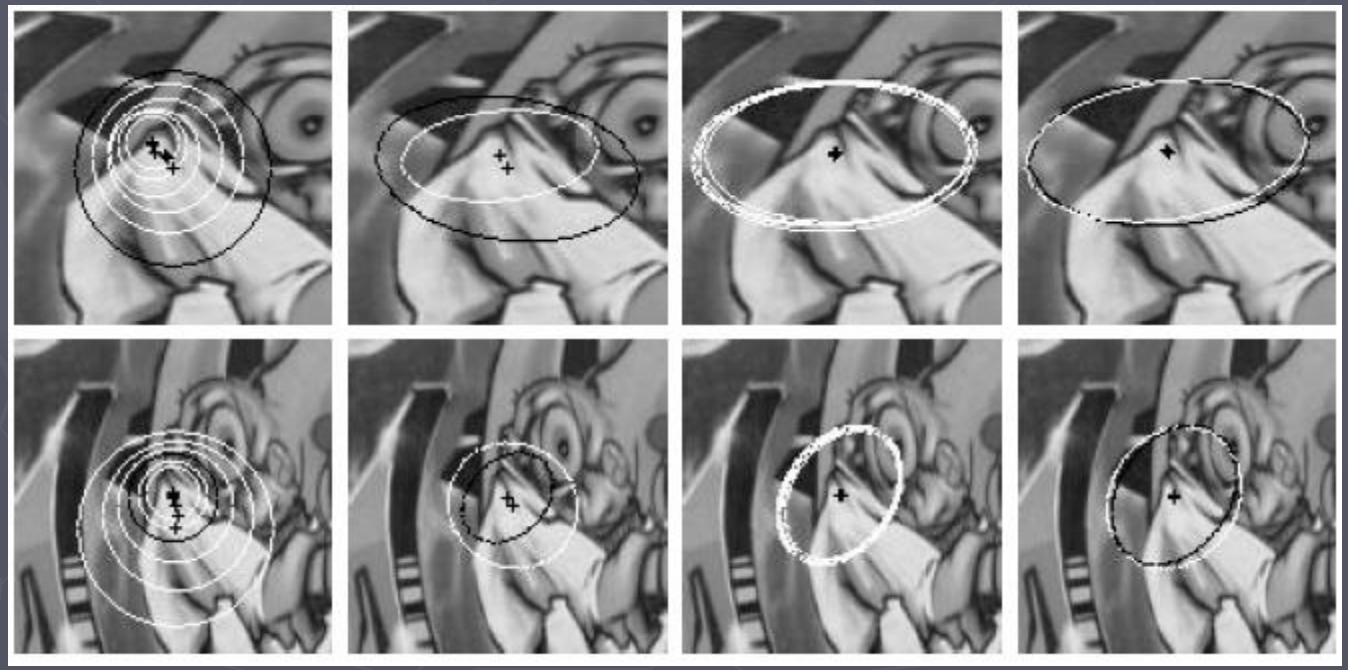
Harris-Laplace points

Mikolajczyk: Harris Affine

- Based on Harris Laplace
- Using normalization / deskewing

Mikolajczyk: Harris Affine

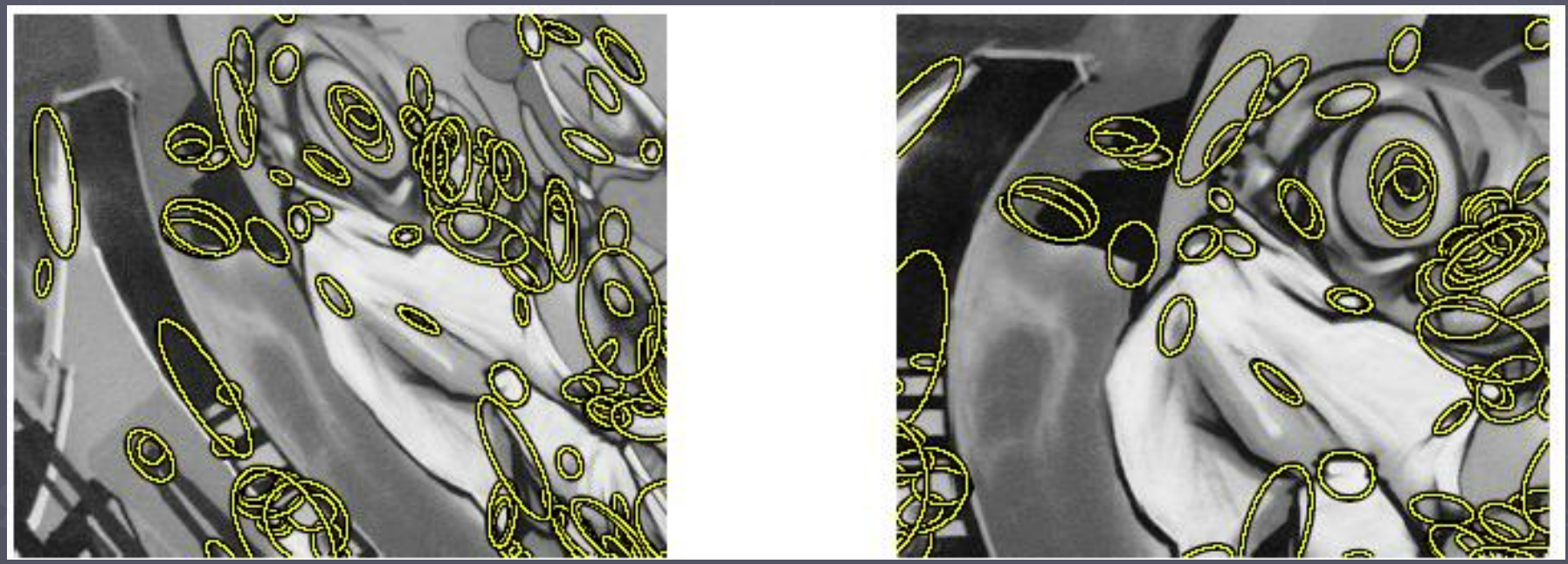
1. Detect multi-scale Harris points
2. Automatically select the scales
3. Adapt affine shape based on second order moment matrix
4. Refine point location



Mikolajczyk: affine invariant interest points

1. Initialization: Multiscale Harris corner detection
2. Iterative algorithm
   1. Normalize window (deskewing)
   2. Select integration scale (max. of LoG)
   3. Select differentiation scale (max. λ_{min} / λ_{max})
   4. Detect spatial localization (Harris)
   5. Compute new affine transformation (μ)
   6. Go to step 2. (unless stop criterion)

Harris Affine



Affine Invariant Detection : Summary

- Under affine transformation, we do not know in advance shapes of the corresponding regions
- Ellipse given by geometric covariance matrix of a region robustly approximates this region
- For corresponding regions ellipses also correspond

Other Methods:

1. Search for extremum along rays [Tuytelaars, Van Gool]:
2. Maximally Stable Extremal Regions [Matas et.al.]

Feature detector and descriptor summary

- Stable (repeatable) feature points can be detected regardless of image changes
  - Scale: search for correct scale as *maximum* of appropriate function
  - Affine: approximate regions with *ellipses* (this operation is affine invariant)
- Invariant and distinctive descriptors can be computed
  - Invariant *moments*
  - *Normalizing* with respect to scale and affine transformation

More on feature detection/description

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

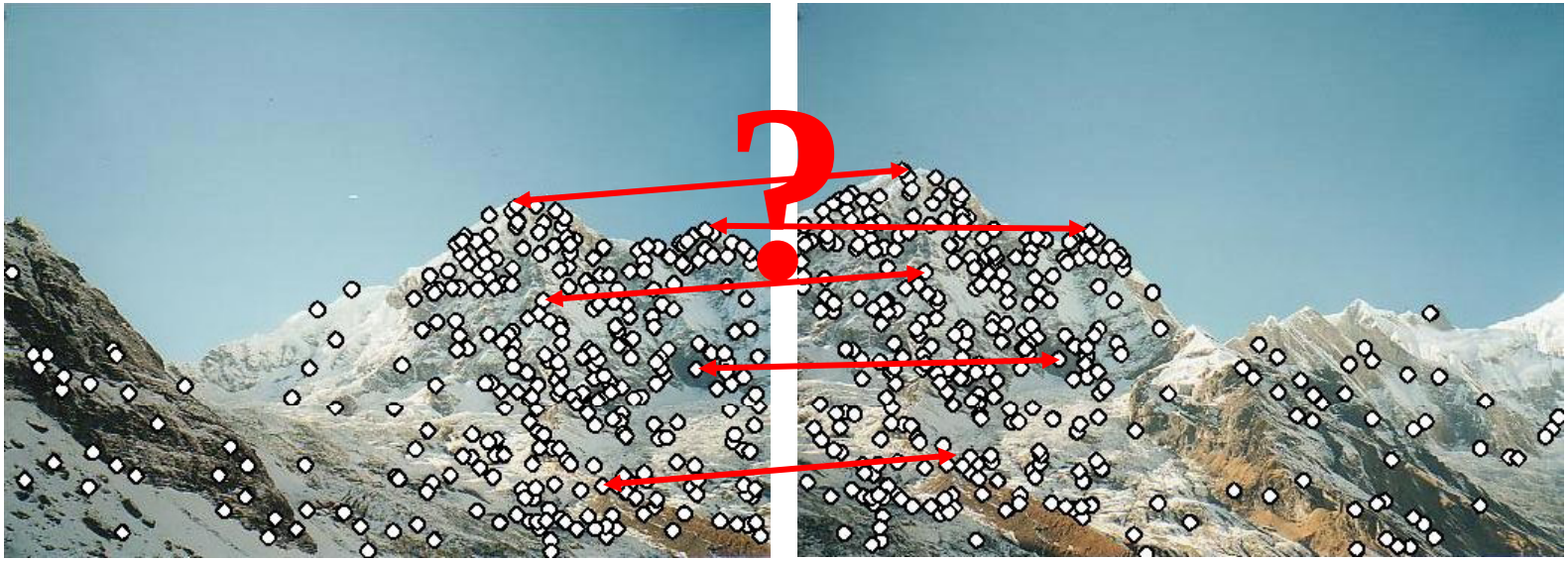
- K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. Van Gool, A comparison of affine region detectors. Technical Report, accepted to IJCV. PDF
- K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors. Technical Report, accepted to PAMI. PDF

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?

Feature descriptors

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



Feature matching

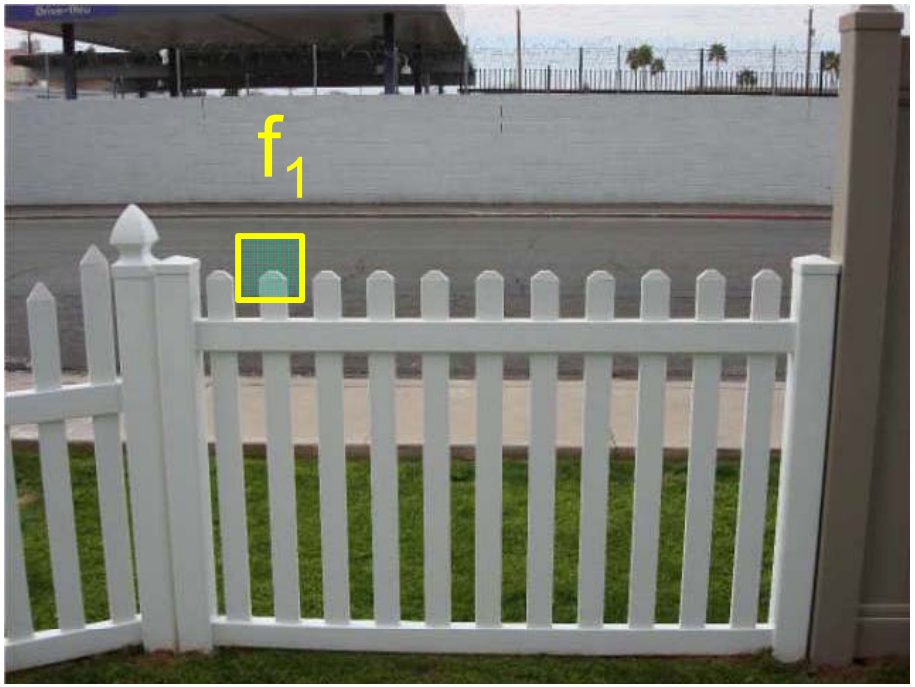
Given a feature in I_1, how to find the best match in I_2?

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

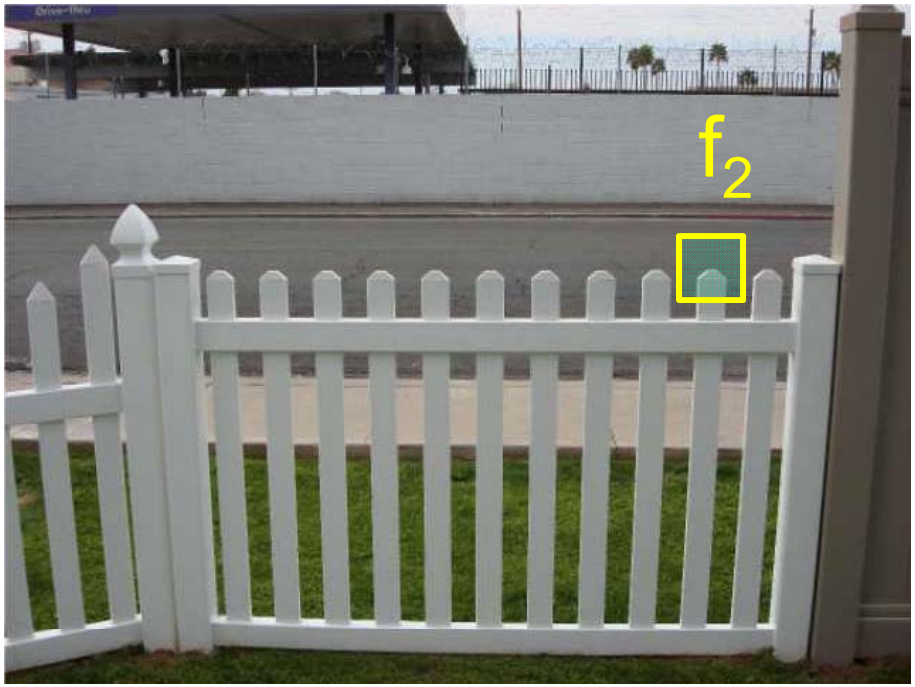
Feature distance

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

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches



I_1

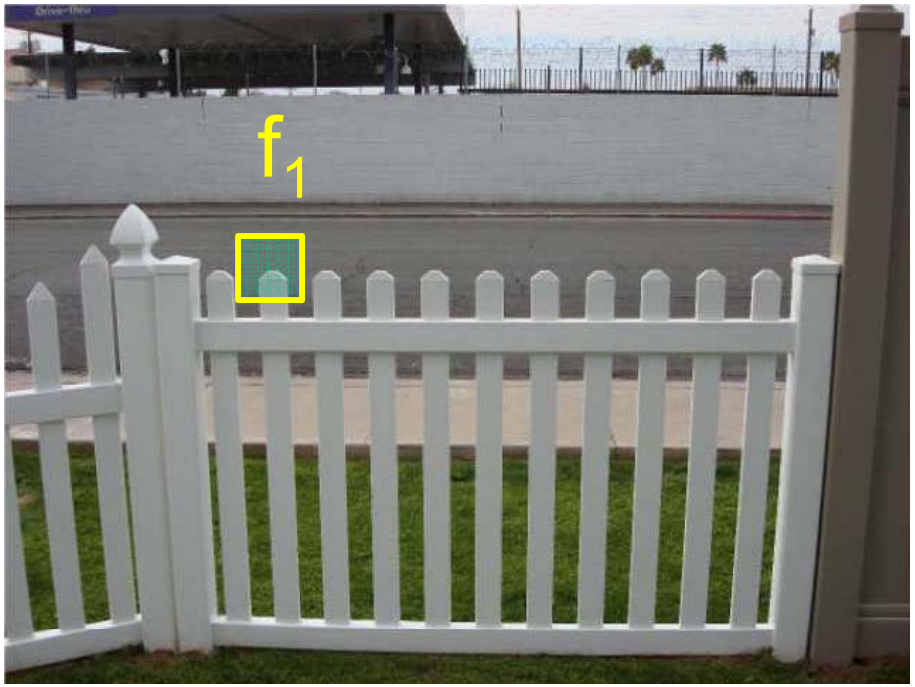


I_2

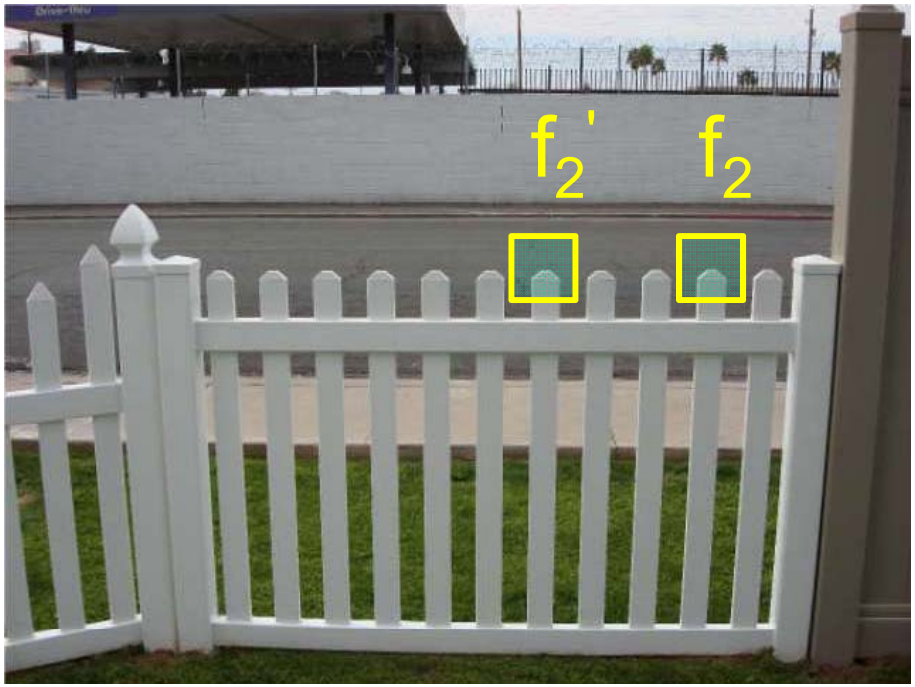
Feature distance

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

- Better approach: ratio distance = SSD(f_1, f_2) / SSD(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 small values for ambiguous matches



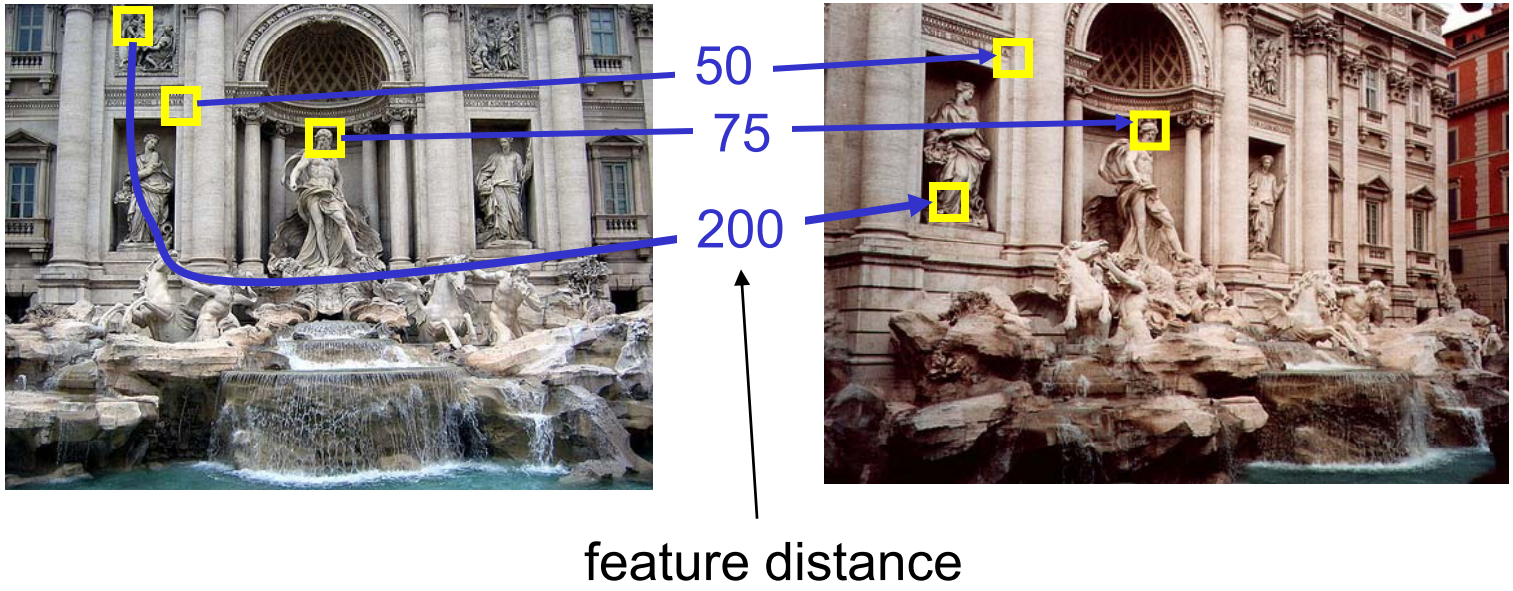
I_1



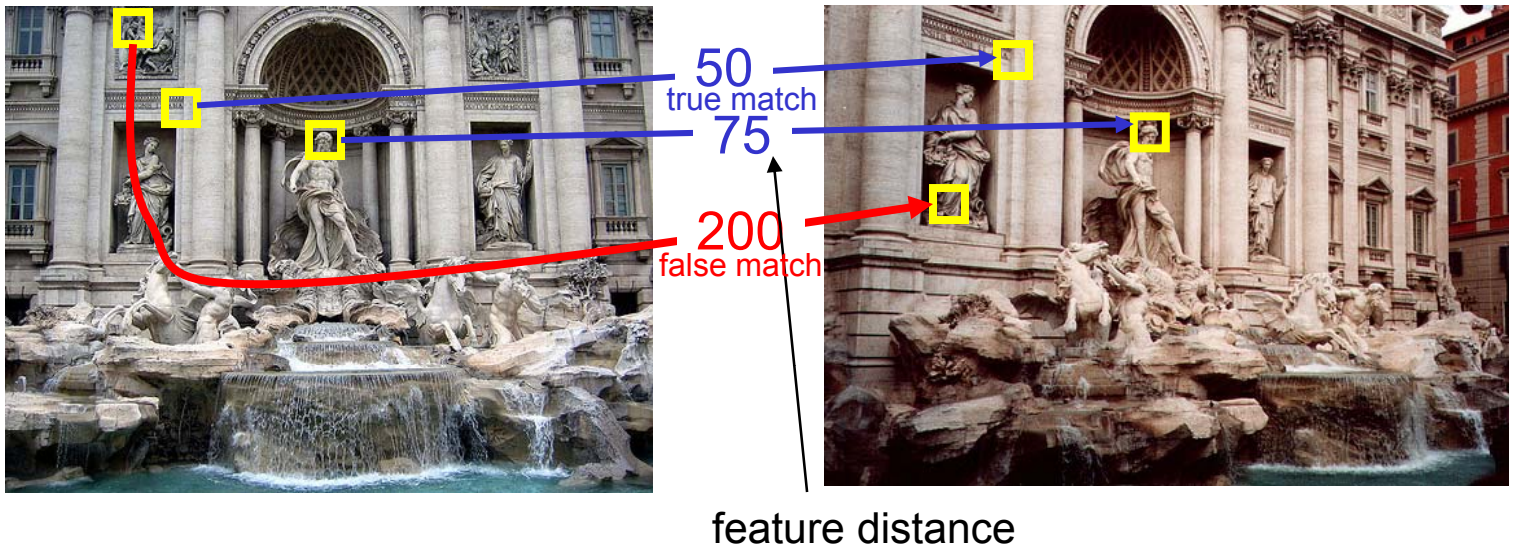
I_2

Evaluating the results

How can we measure the performance of a feature matcher?



True/false positives

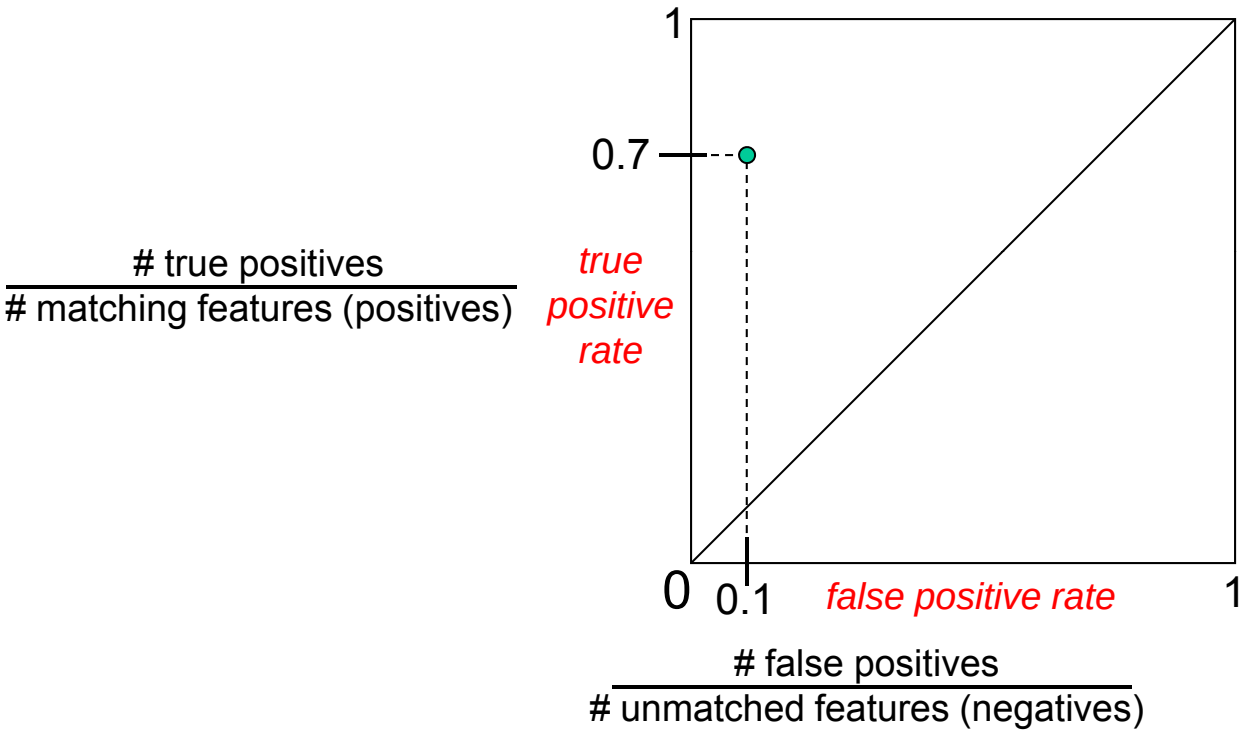


The distance threshold affects performance

- 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?

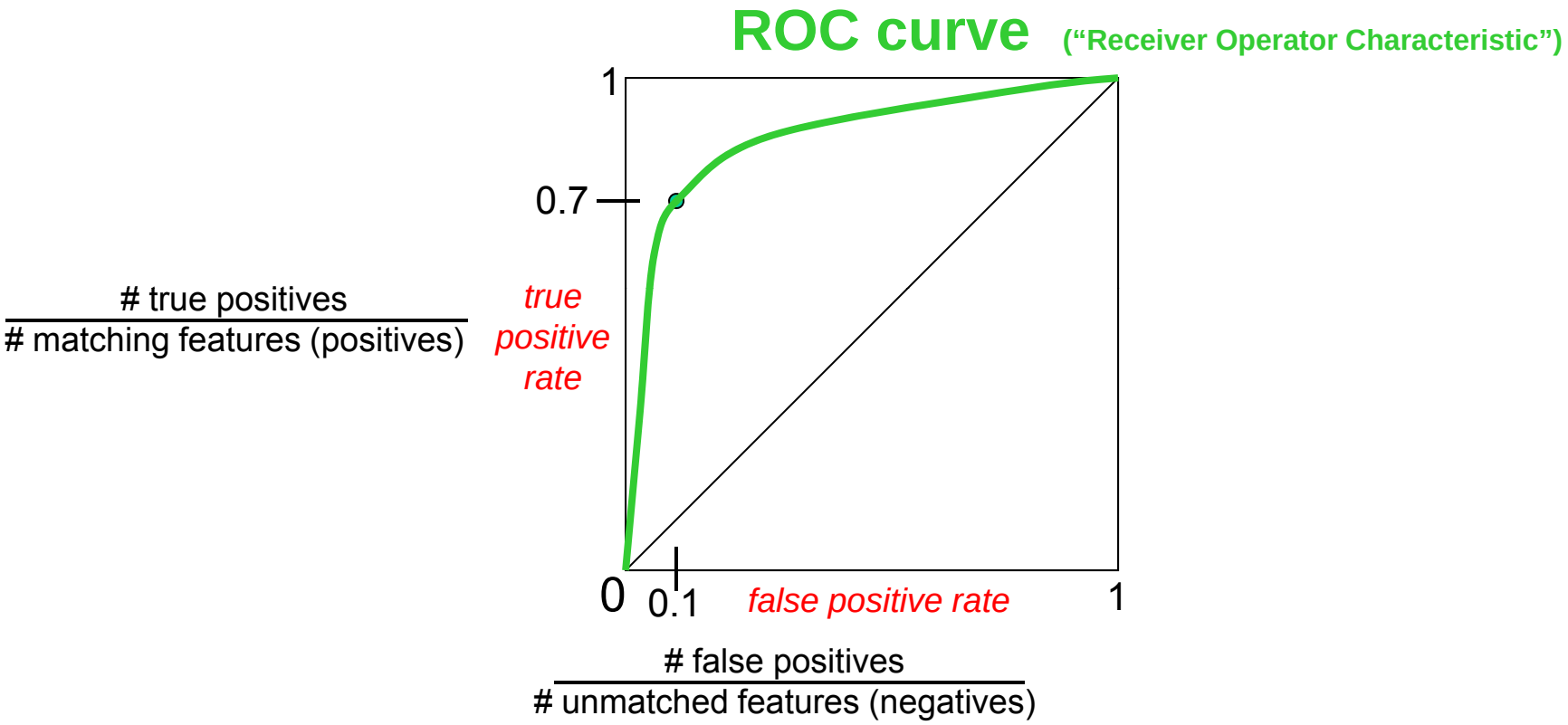
Evaluating the results

How can we measure the performance of a feature matcher?



Evaluating the results

How can we measure the performance of a feature matcher?



ROC Curves

- Generated by counting # current/incorrect matches, for different threholds
- 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

Advanced local features topics

- Self-Similarity
- Space-Time

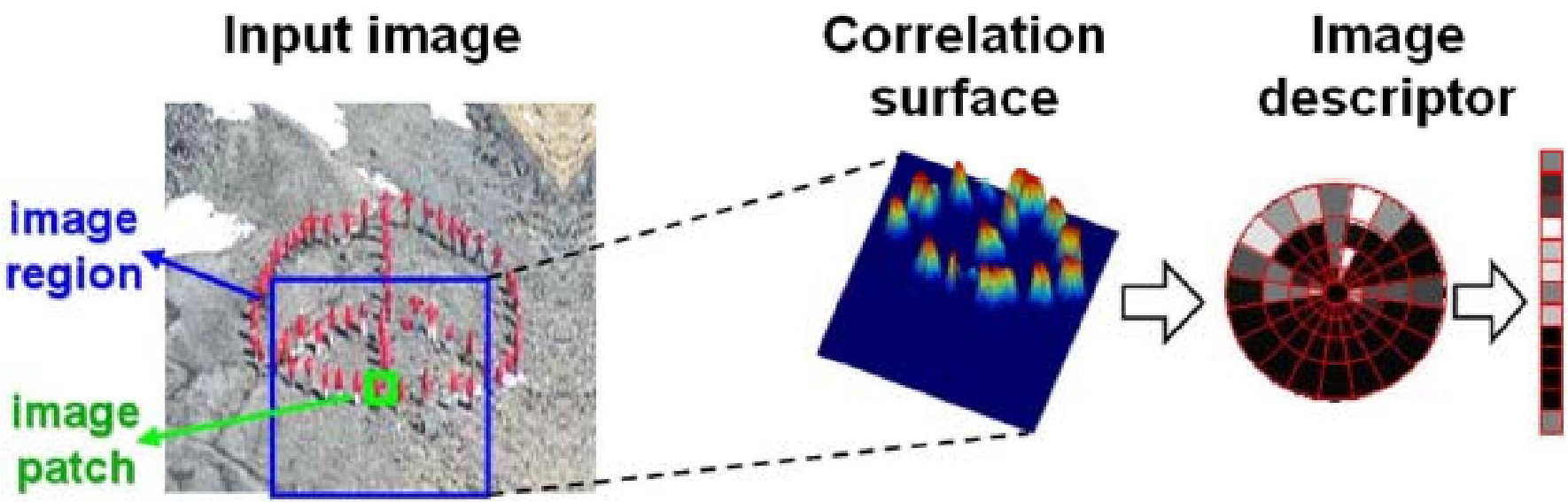
Matching Local Self-Similarities across Images and Videos

Eli Shechtman Michal Irani
Dept. of Computer Science and Applied Math
The Weizmann Institute of Science
76100 Rehovot, Israel

Abstract

We present an approach for measuring similarity between visual entities (images or videos) based on matching internal self-similarities. What is correlated across images (or across video sequences) is the internal layout of local self-similarities (up to some distortions), even though the patterns generating those local self-similarities are quite different in each of the images/videos. These internal self-similarities are efficiently captured by a compact local "self-similarity descriptor", measured densely throughout the image/video, at multiple scales, while accounting for local and global geometric distortions. This gives rise to matching capabilities of complex visual data, including detection of objects in real cluttered images using only rough hand-sketches, handling textured objects with no clear boundaries, and detecting complex actions in cluttered video data with no prior learning. We compare our measure to commonly used image-based and video-based similarity measures, and demonstrate its applicability to object detection, retrieval, and action detection.

Figure 1. *These images of the same object (a heart) do NOT share common image properties (colors, textures, edges), but DO share a similar geometric layout of local internal self-similarities.*


Input image
Correlation surface
Image descriptor
image region
image patch

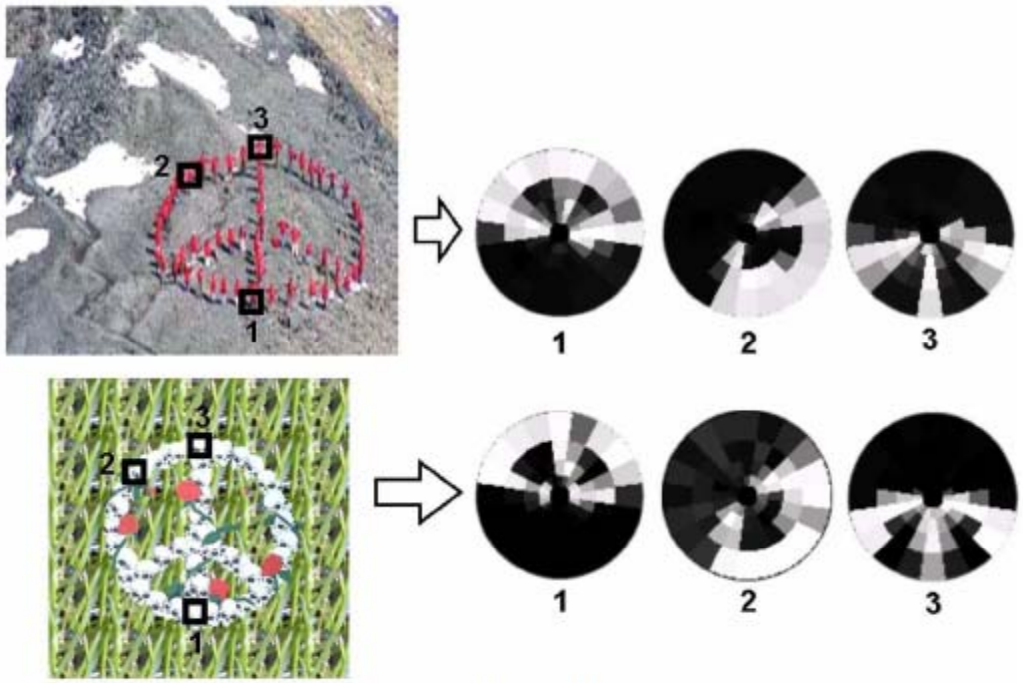


Figure 3. **Corresponding "Self-similarity descriptors".** *We show a few corresponding points (1,2,3) across two images of the same object, with their "self-similarity"descriptors. Despite the large difference in photometric properties between the two images, their corresponding "self-similarity" descriptors are quite similar.*

Figure 4. **Object detection.** *(a) A single template image (a flower). (b) The images against which it was compared with the corresponding detections. The continuous likelihood values above a threshold (same threshold for all images) are shown superimposed on the gray-scale images, displaying detections of the template at correct locations (red corresponds to the highest values).*

(a)

(b)

Figure 6. Detection using a sketch. *(a) A hand-sketched template. (b) The images against which it was compared with the corresponding detections.*

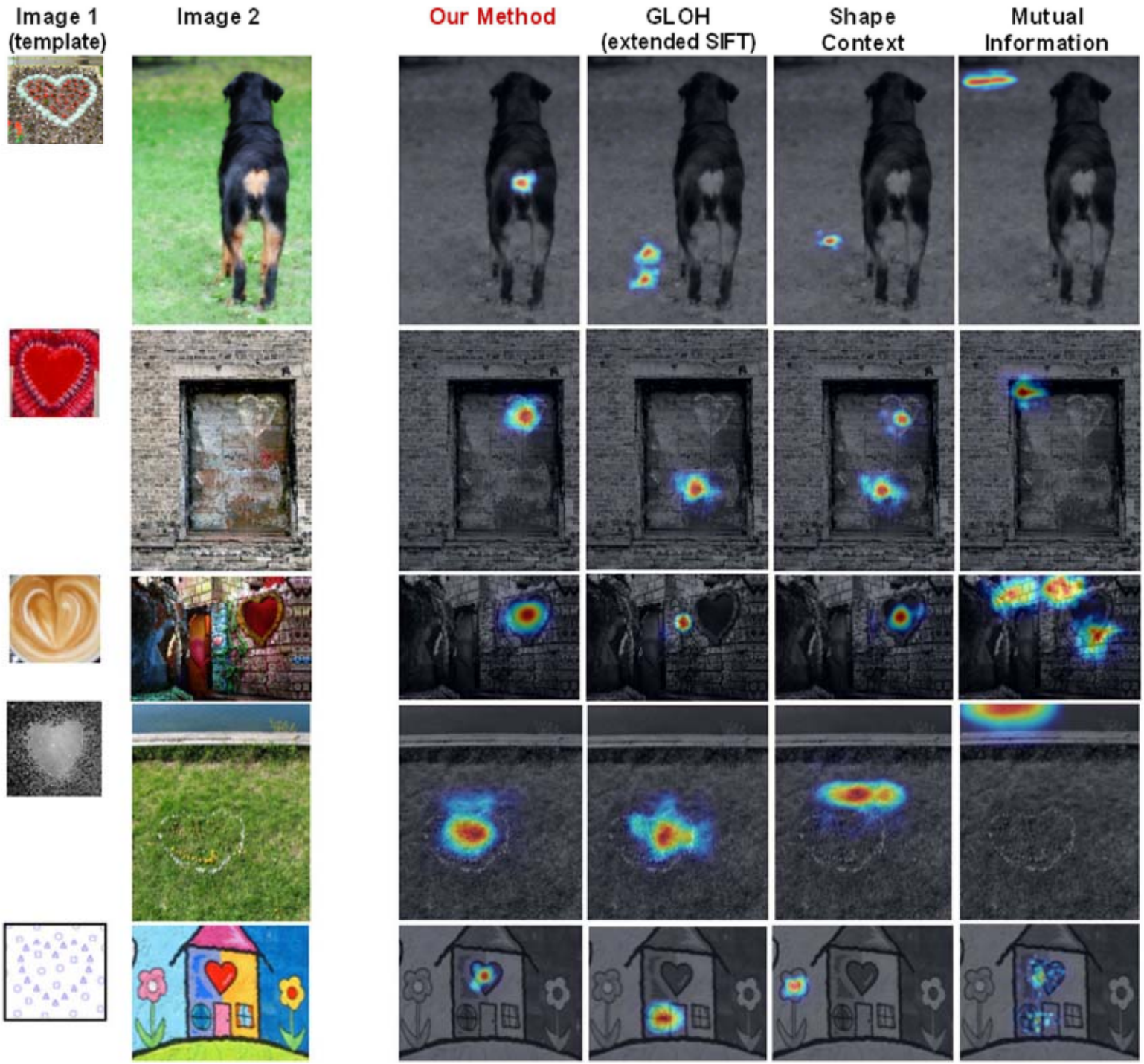
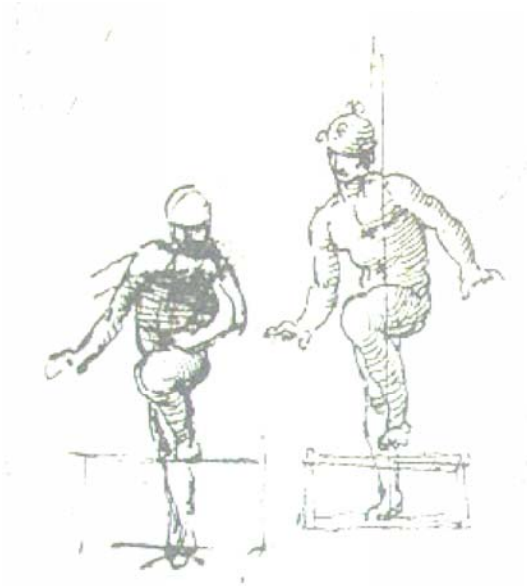


Image 1 (template)
Image 2
Our Method
GLOH (extended SIFT)
Shape Context
Mutual Information



Human actions
in computer vision

Ivan Laptev
INRIA Rennes, France
ivan.laptev@inria.fr

Summer school, June 30 - July 11, 2008, Lotus Hill, China

Motivation

Goal: Interpretation of dynamic scenes

... non-rigid object motion ... camera motion ... complex background motion

Common methods:

- **Camera stabilization**
- **Segmentation** ?
- **Tracking** ?

Common problems:

- **Complex BG motion**
- **Changes in appearance**

⇒ ***No global assumptions about the scene***

Space-time

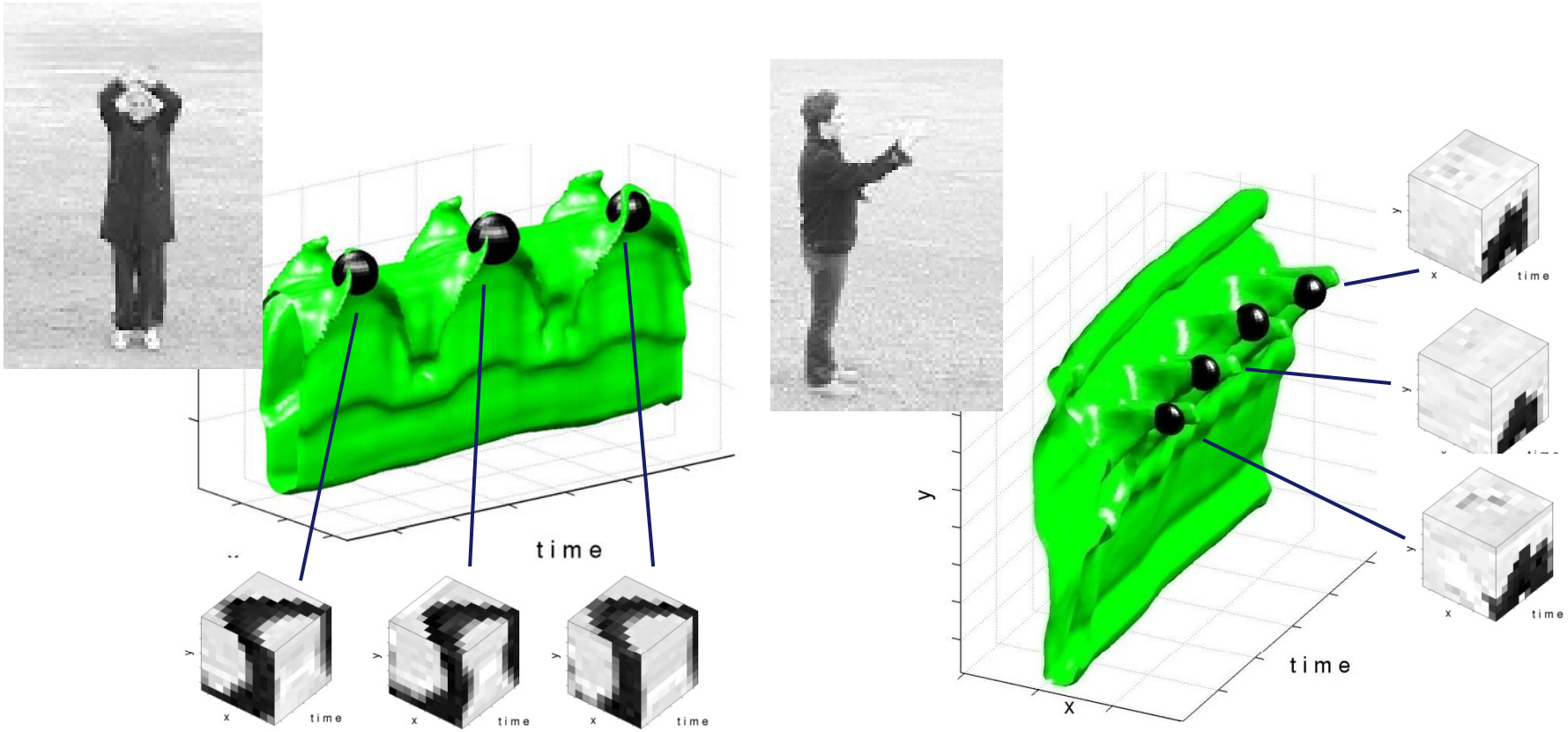
No global assumptions $\Rightarrow$

Consider local spatio-temporal neighborhoods

Space-time

No global assumptions ⇒

Consider local spatio-temporal neighborhoods



Space-Time interest points

What neighborhoods to consider?

Distinctive **neighborhoods** $\Rightarrow$ **High image variation in *space* and *time*** $\Rightarrow$ **Look at the distribution of the gradient**

Definitions:

$f : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$ Original image sequence

$g(x, y, t; \Sigma)$ Space-time Gaussian with covariance $\Sigma \in \mathrm{SPSD}(3)$

$L_\xi(\cdot; \Sigma) = f(\cdot) * g_\xi(\cdot; \Sigma)$ Gaussian derivative of f

$\nabla L = (L_x, L_y, L_t)^T$ Space-time gradient

$$\mu(\cdot; \Sigma) = \nabla L(\cdot; \Sigma)(\nabla L(\cdot; \Sigma))^T * g(\cdot; s\Sigma) = \begin{pmatrix} \mu_{xx} & \mu_{xy} & \mu_{xt} \\ \mu_{xy} & \mu_{yy} & \mu_{yt} \\ \mu_{xt} & \mu_{yt} & \mu_{tt} \end{pmatrix}$$

Second-moment matrix

Space-Time interest points

Properties of $\mu(\cdot;\Sigma)$:

$\mu(\cdot;\Sigma)$ defines second order approximation for the local distribution of ∇L within neighborhood Σ

$\text{rank}(\mu) = 1 \Rightarrow$ 1D space-time variation of f, e.g. *moving bar*

$\text{rank}(\mu) = 2 \Rightarrow$ 2D space-time variation of f, e.g. *moving ball*

$\text{rank}(\mu) = 3 \Rightarrow$ 3D space-time variation of f, e.g. *jumping ball*

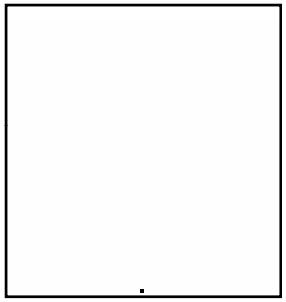
Large eigenvalues of μ can be detected by the local maxima of *H* over *(x,y,t)*:

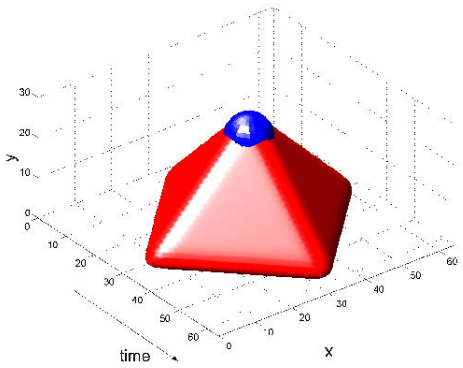
$$H(p;\Sigma) = \det(\mu(p;\Sigma)) + k\,\text{trace}^3(\mu(p;\Sigma))$$
$$= \lambda_1\lambda_2\lambda_3 - k(\lambda_1+\lambda_2+\lambda_3)^3$$

(similar to Harris operator [Harris and Stephens, 1988])

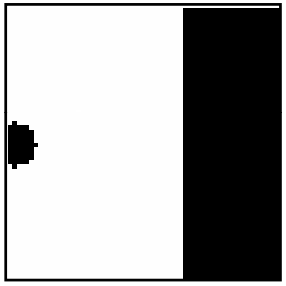
Space-Time interest points

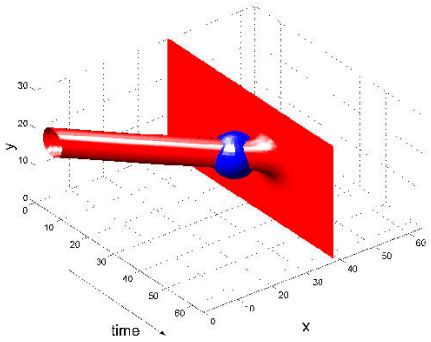
accelerations



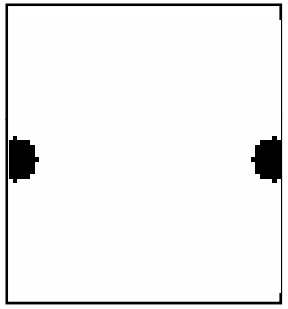


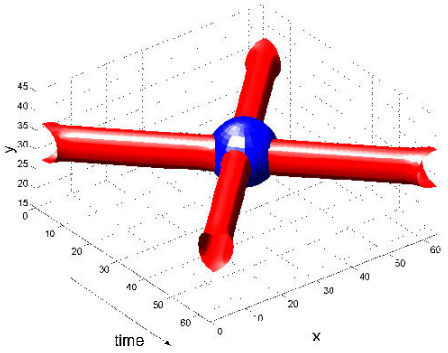
appearance/
disappearance





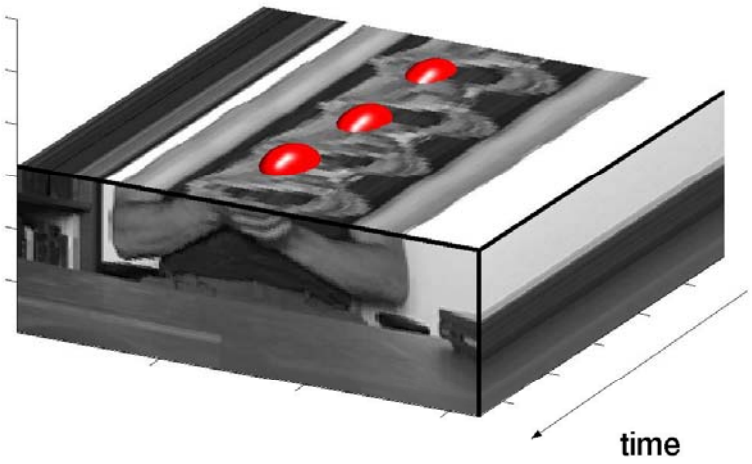
split/merge

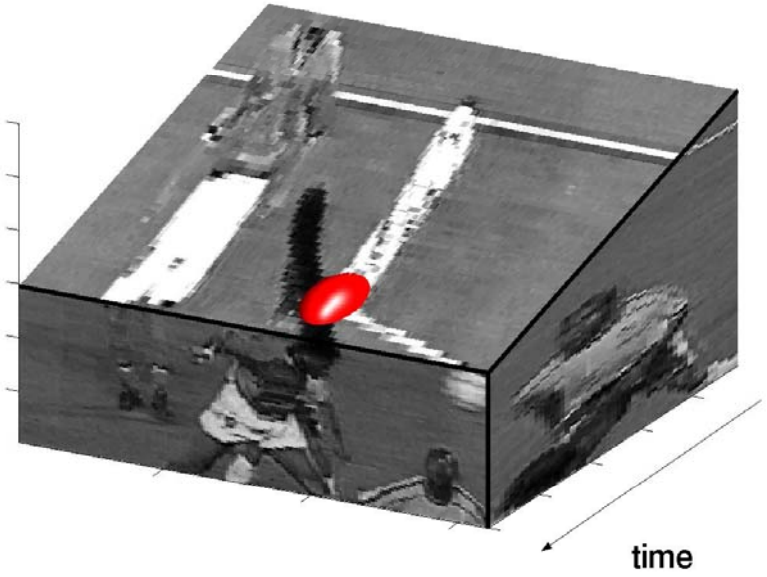




Space-Time interest points

Motion event detection



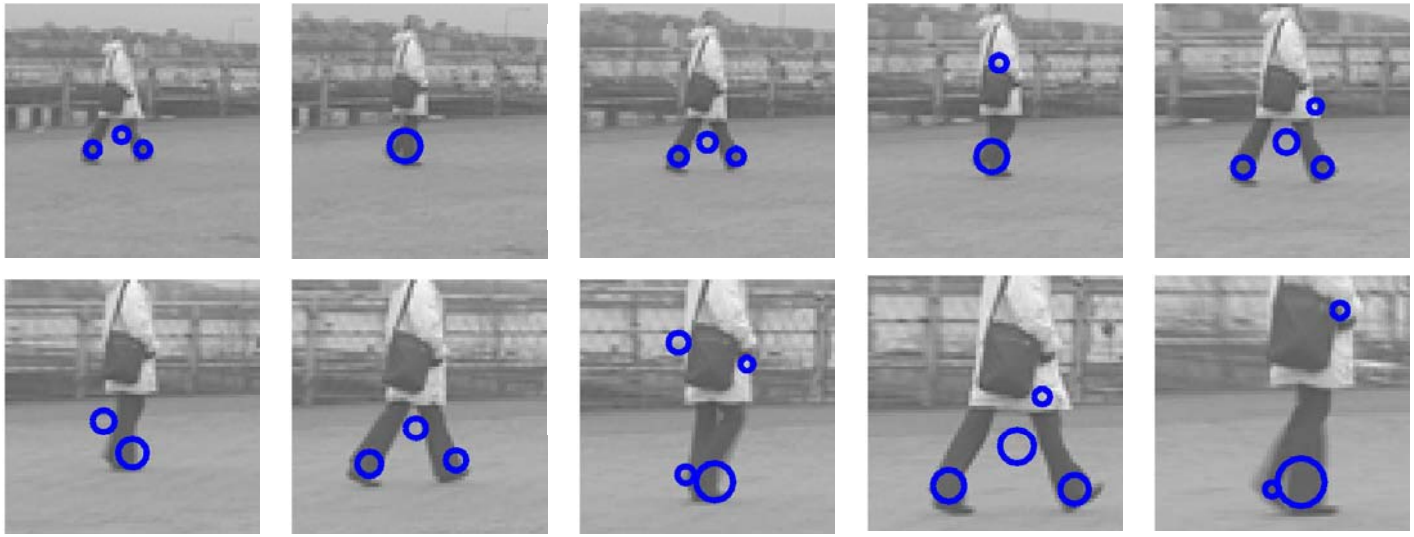


Space-Time interest points

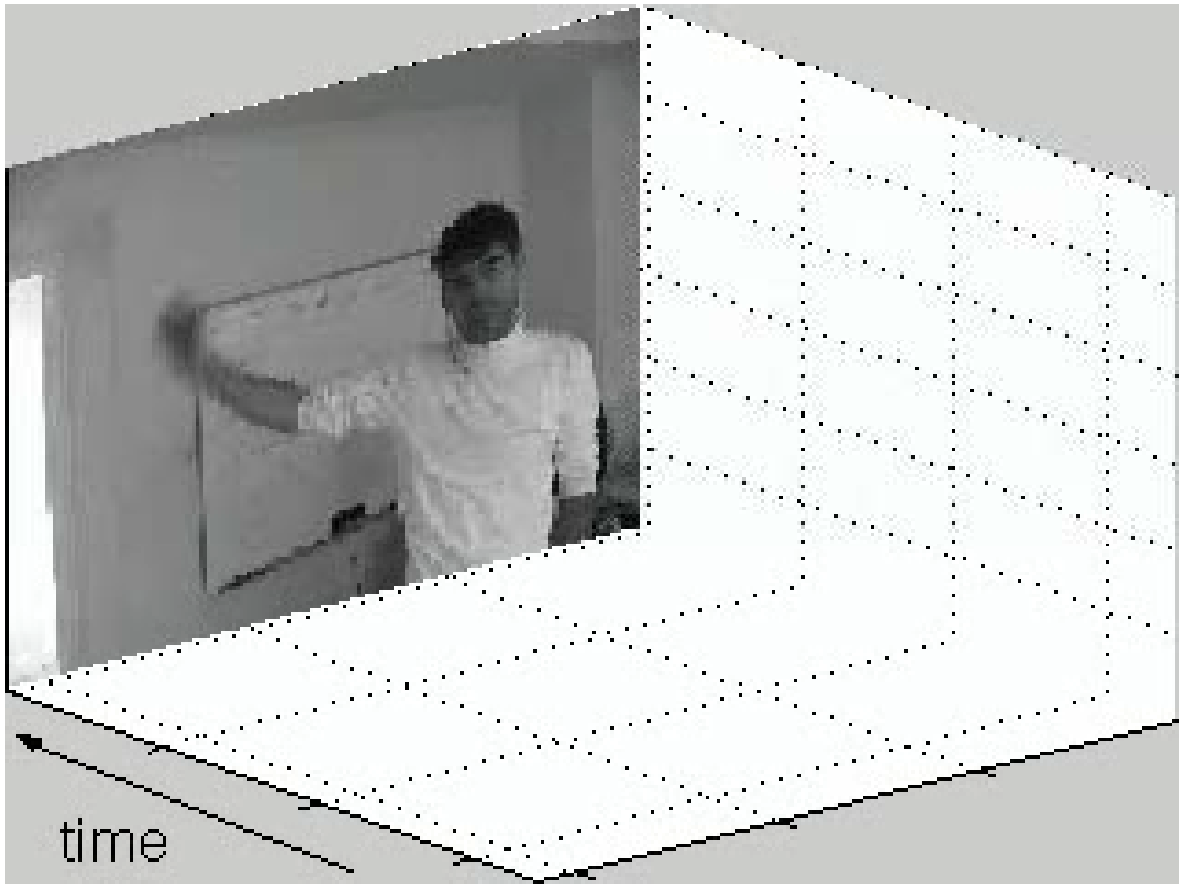
Motion event detection: complex background

Spatio-temporal scale selection

Stability to size changes, e.g. camera zoom



Spatio-temporal scale selection



Selection of temporal scales captures the *frequency* of events

Sequence alignment

- Given a data sequence with the current moment t_0, detect and classify interest points in the time window of length t_w: (t_0, t_0-t_w)

- Transform model features according to X and for each model feature $f_{m,i}=(x_{m,i}, y_{m,i}, t_{m,i}, \sigma_{m,i}, \tau_{m,i}, c_{m,i})$ compute its distance d_i to the most close data feature $f_{d,j}$, $c_{d,j}=c_{m,i}$:

$$d_i = \sqrt{\frac{a}{\sigma_{m,i}^2}((x_{m,i} - x_{d,j})^2 + (y_{m,i} - y_{d,j})^2) + \frac{b}{\tau_{m,i}^2}(t_{m,i} - t_{d,j})^2}$$

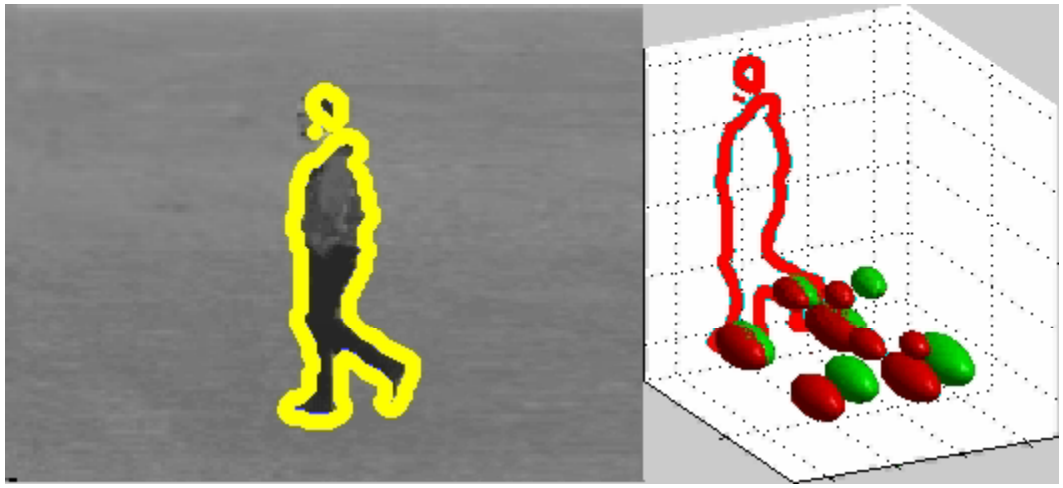
- Define the "fit function" D of model configuration X *as a sum* of distances of all features weighted w.r.t. their "age" (t_0-t_m) such that recent features get more influence on the matching

$$D(X) = \sum_i^N d_i \exp(-\frac{t_0 - t_{m,i}}{\rho^2})$$

Sequence alignment

At each moment t_0 minimize D with respect to X using standard Gauss-Newton minimization method

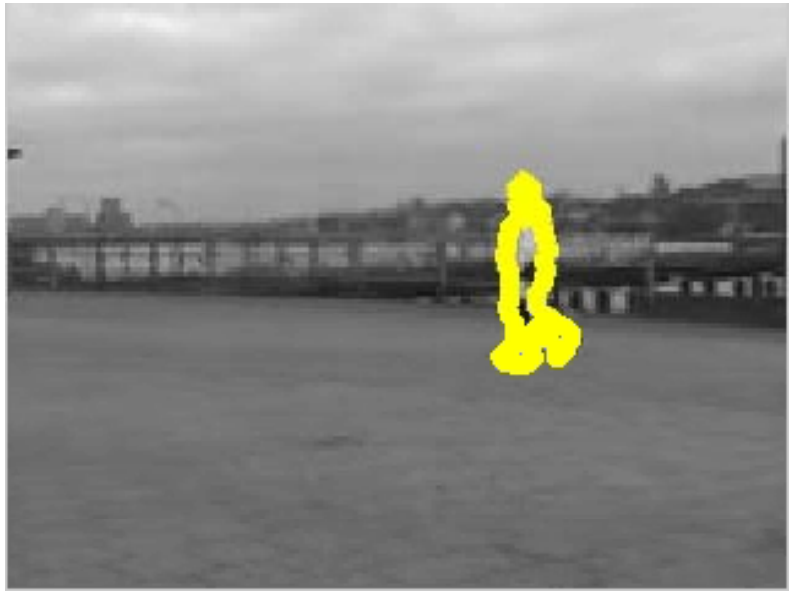
$$\tilde{X} = \text{argmin}_X \, D(X, t_0)$$



data features

model features

Experiments



Experiments

Today: Feature Detection and Matching

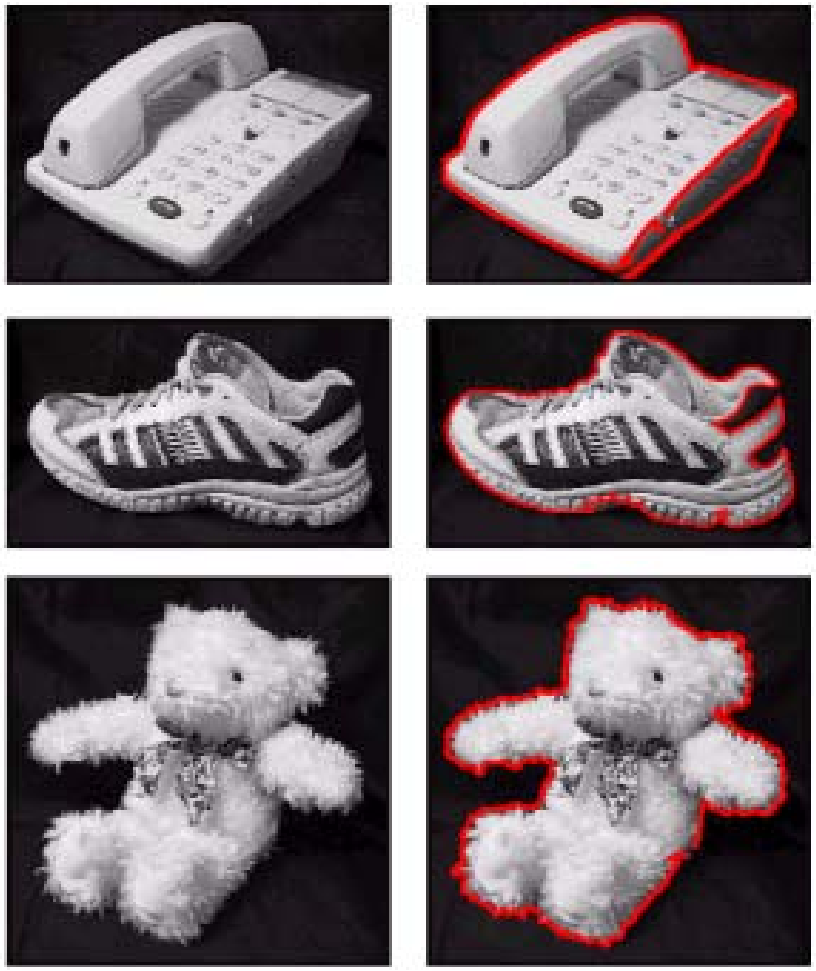
- Local features
- Pyramids for invariant feature detection
- Local descriptors
- Matching

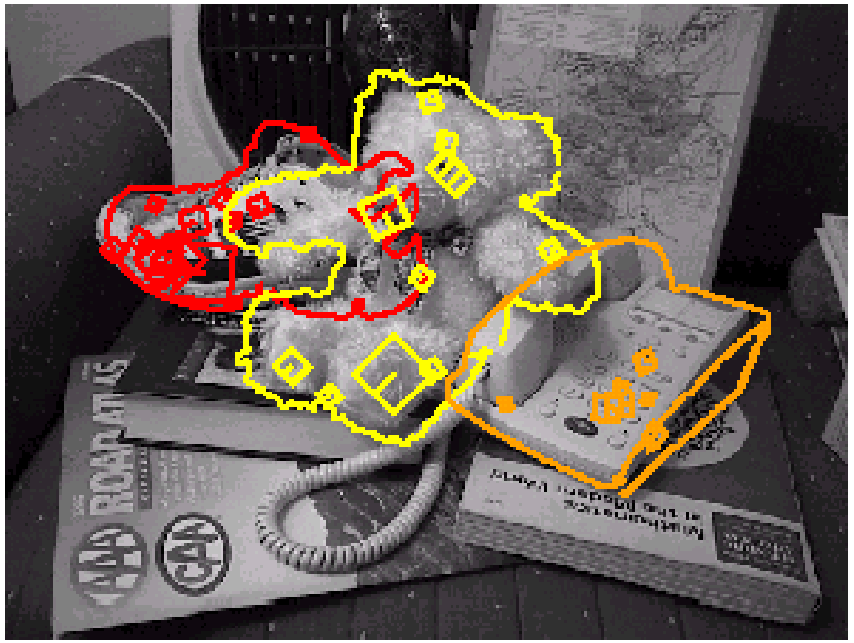
Lots of applications

Features are used for:

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

Object recognition (David Lowe)





Snaptell

http://snaptell.com/demos/DemoLarge.htm

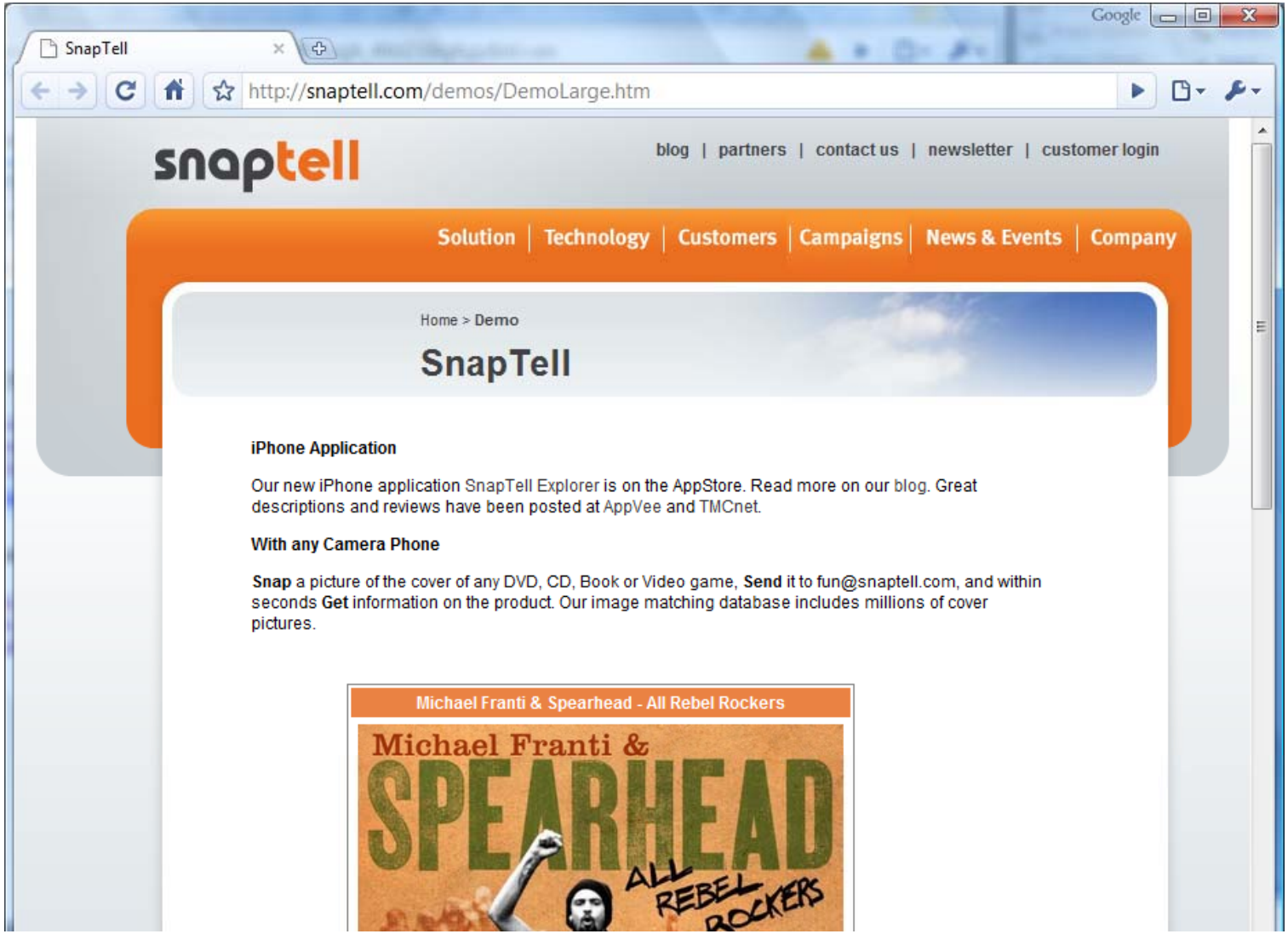
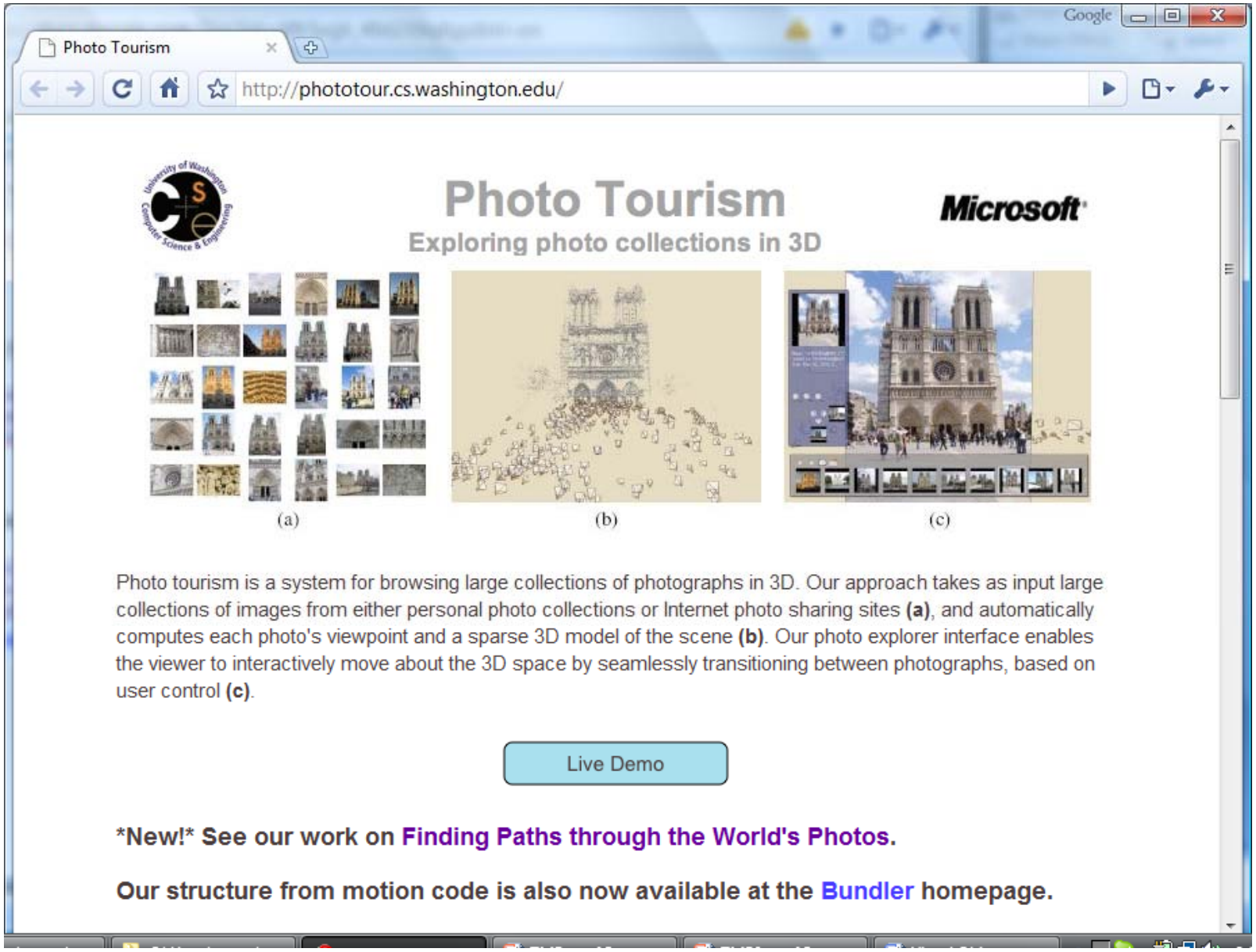


Photo Tourism



Slide Credits

- Bill Freeman
- Kristen Grauman
- Steve Sietz
- Ivan Laptev
- Tinne Tuytelaars

Next time: Texture