CS 2770: Computer Vision

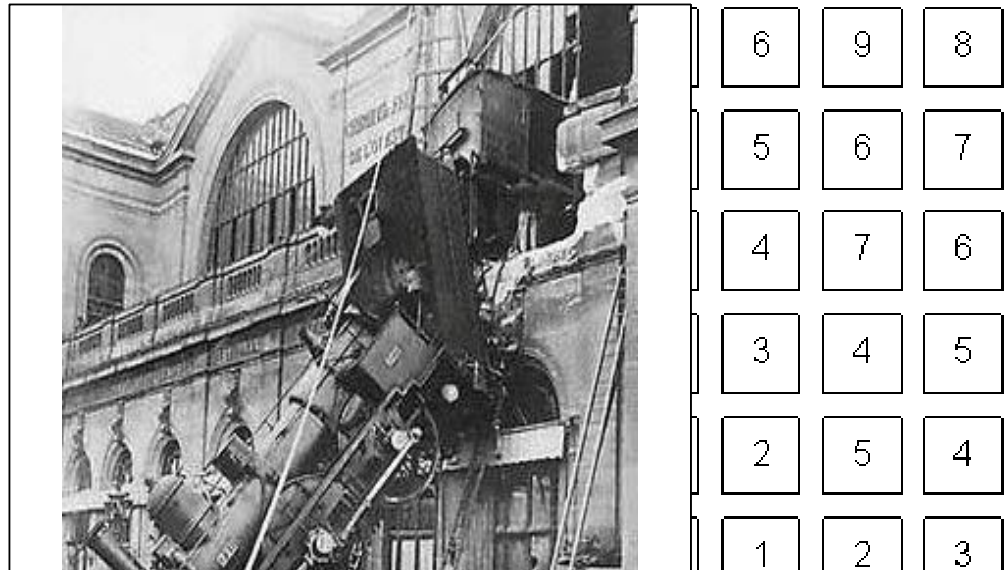
Local Feature Detection and Description

Prof. Adriana Kovashka
University of Pittsburgh
February 9, 2017

Plan for today

- Feature detection / keypoint extraction
  - Corner detection
  - Blob detection
- Feature description (of detected features)

An image is a set of pixels…

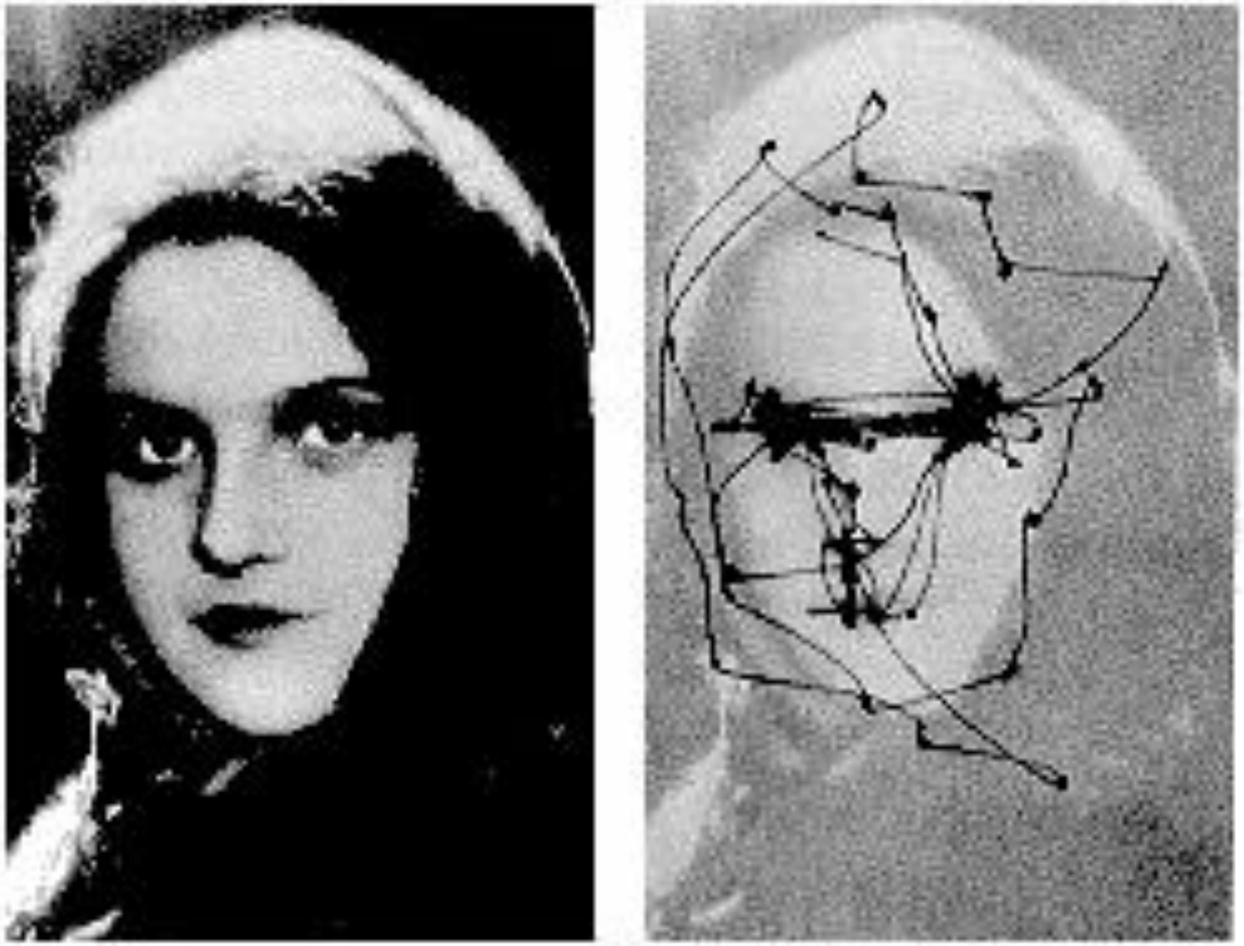


Adapted from S. Narasimhan

Problems with pixel representation

- Not invariant to small changes
  - Translation
  - Illumination
  - etc.
- Some parts of an image are more important than others
- What do we want to represent?

Human eye movements



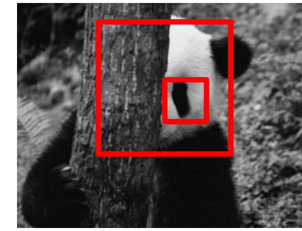
Yarbus eye tracking

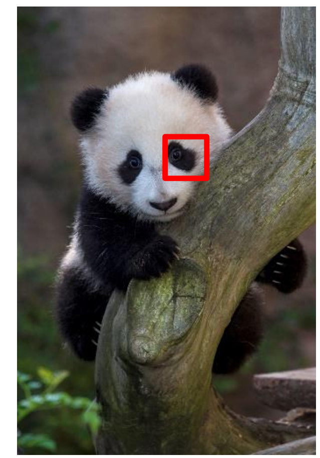
Local features

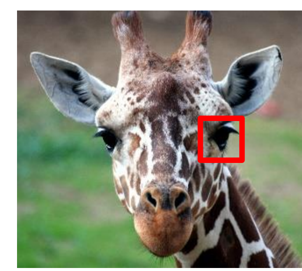
- *Local* means that they only cover a small part of the image
- There will be many local features detected in an image
- Later we'll talk about how to use those to compute a representation of the whole image
- Local features usually exploit image gradients, ignore color

Local features: desired properties

- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion
- Repeatability and flexibility
  - The same feature can be found in several images despite geometric, photometric transformations
  - Robustness to expected variations
  - Maximize correct matches
- Distinctiveness
  - Each feature has a distinctive description
  - Minimize wrong matches
- Compactness and efficiency
  - Many fewer features than image pixels







Adapted from K. Grauman and D. Hoiem

Interest(ing) points

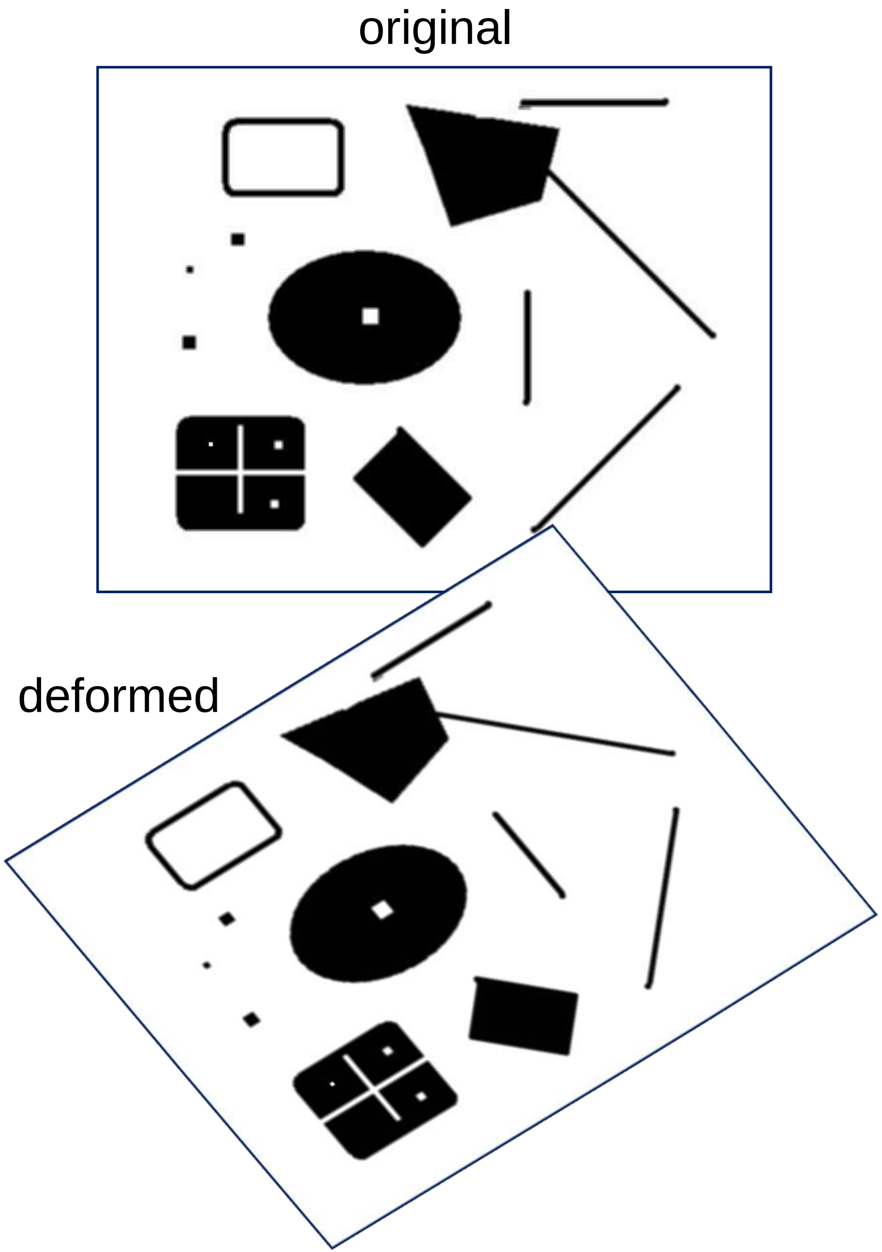
- Note: “interest points” = “keypoints”, also sometimes called “features”

- Many applications
  - Image search: which points would allow us to match images between query and database?
  - Recognition: which patches are likely to tell us something about the object category?
  - 3D reconstruction: how to find correspondences across different views?
  - Tracking: which points are good to track?

Adapted from D. Hoiem

Interest points

- Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
  - Which points would you choose?



D. Hoiem

Choosing interest points

Where would you tell your friend to meet you?

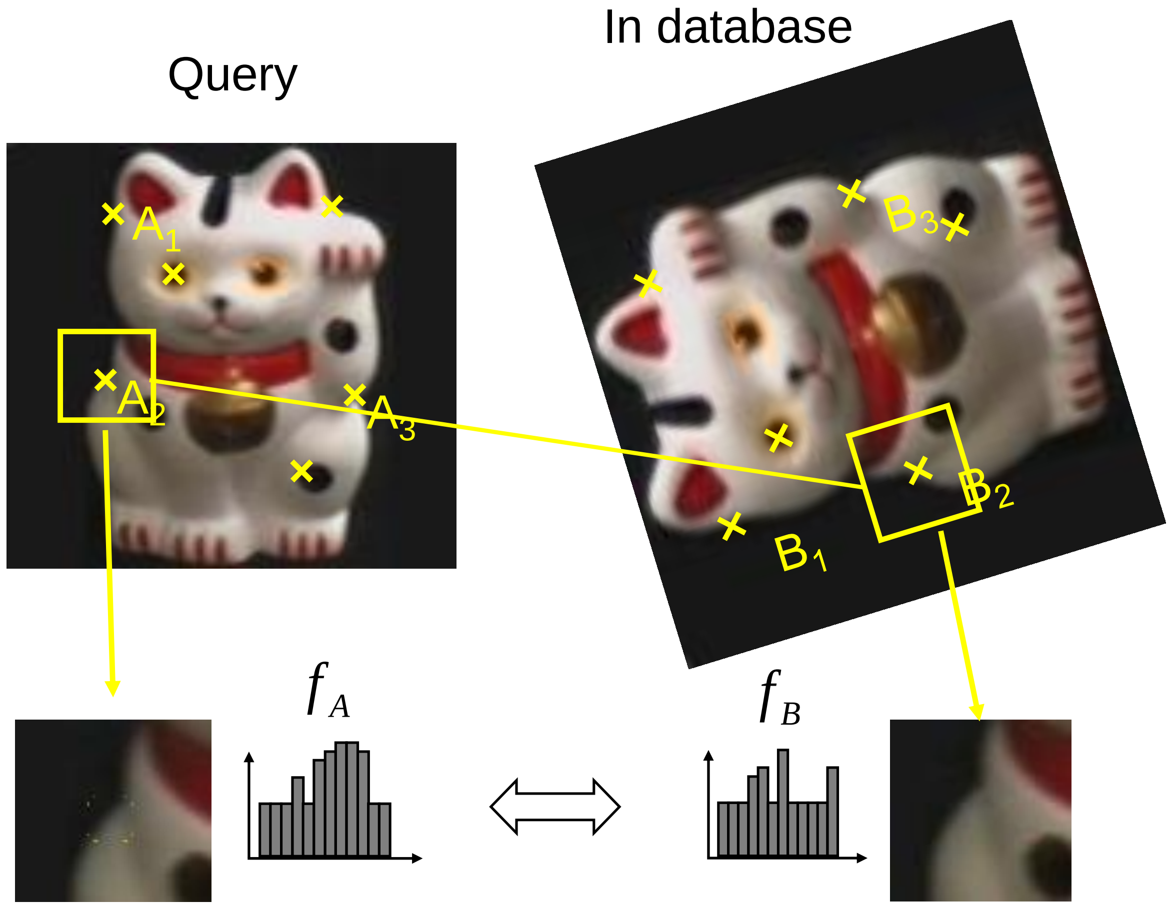
→ Corner detection

Choosing interest points

Where would you tell your friend to meet you?

→ Blob detection

Application 1: Keypoint Matching for Search



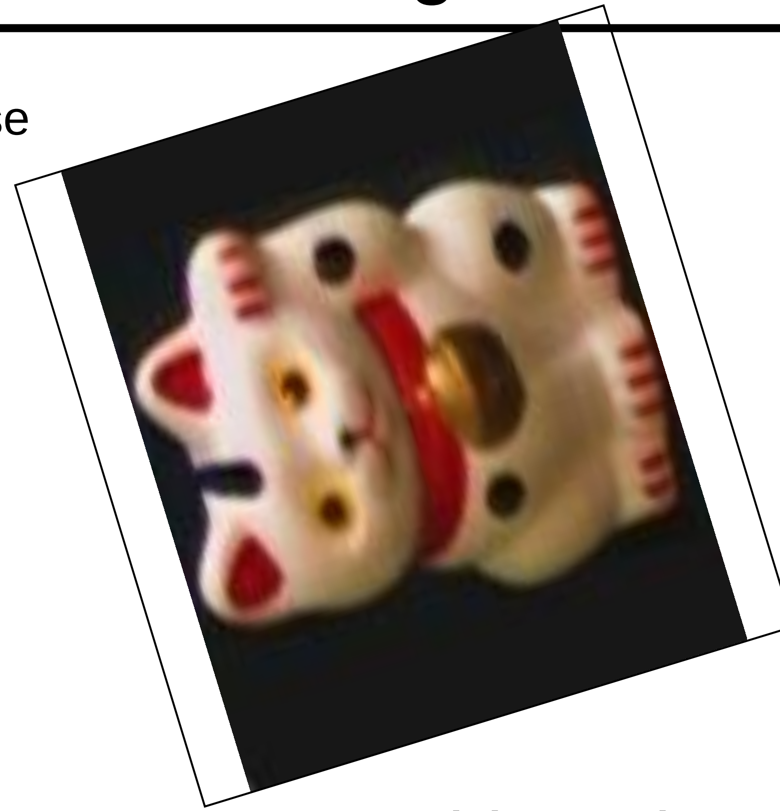
$$d(f_A, f_B) < T$$

1. Find a set of distinctive key-points
2. Define a region around each keypoint (window)
3. Compute a local descriptor from the region
4. Match descriptors

Adapted from K. Grauman, B. Leibe

Application 1: Keypoint Matching For Search

Query



In database

Goal:

We want to detect points that are *repeatable* and *distinctive*

- *Repeatable:* so that if images are slightly different, we can still retrieve them
- *Distinctive:* so we don't retrieve irrelevant content

Adapted from D. Hoiem

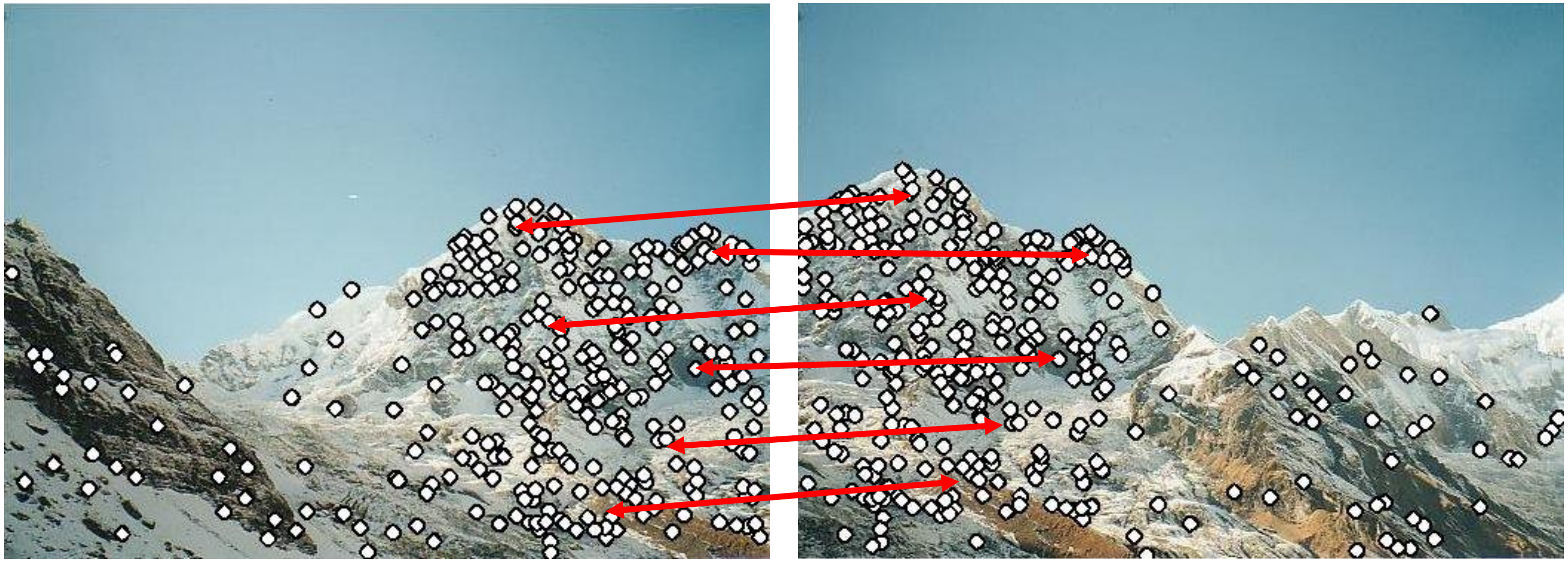
Application 2: Panorama stitching

We have two images – how do we combine them?

L. Lazebnik

Application 2: Panorama stitching

We have two images – how do we combine them?



Step 1: extract features

Step 2: match features

Application 2: Panorama stitching

We have two images – how do we combine them?

Step 1: extract features

Step 2: match features

Step 3: align images

L. Lazebnik

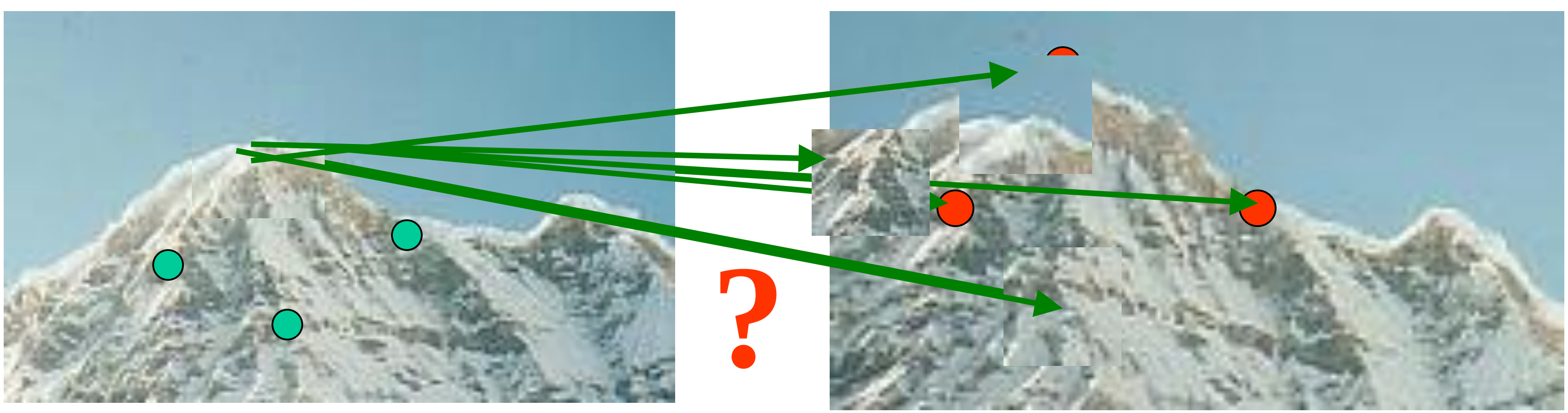
Application 2: Panorama stitching

- No chance to find true matches, yet we have to be able to run the detection procedure *independently* per image.
- We want to detect (at least some of) the same points in both images → want *repeatability* of the interest operator

Adapted from K. Grauman

Application 2: Panorama stitching

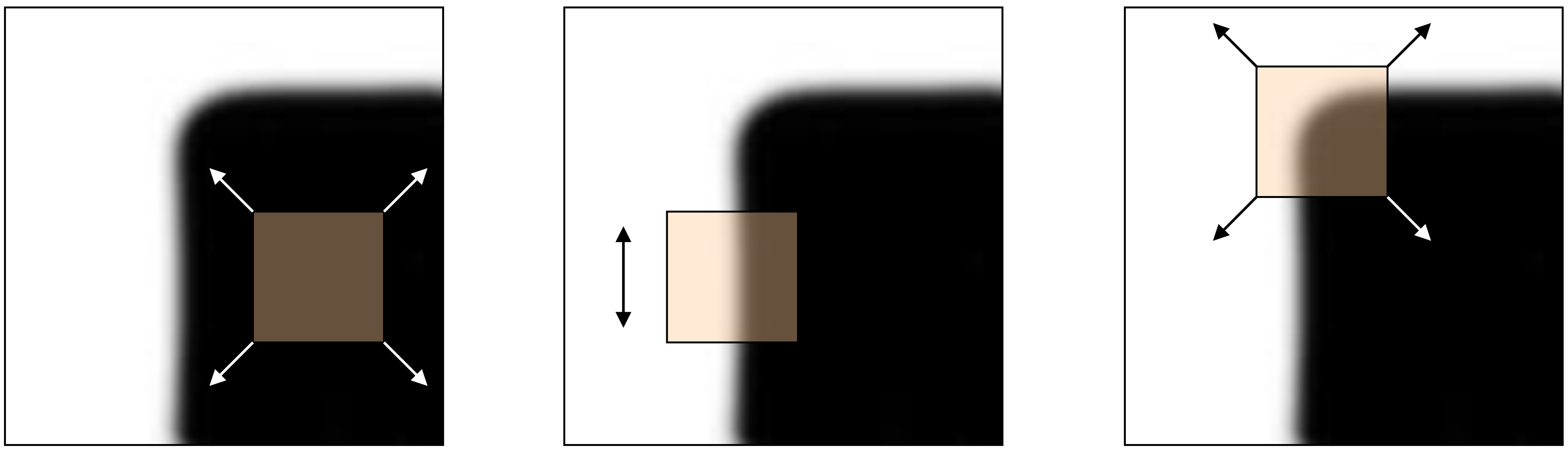
- We want to be able to reliably determine which point goes with which → want operator *distinctiveness*
- Must provide some invariance to geometric and photometric differences between the two views, without finding many false matches



Adapted from K. Grauman

Corners as distinctive interest points

- We should easily recognize the keypoint 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

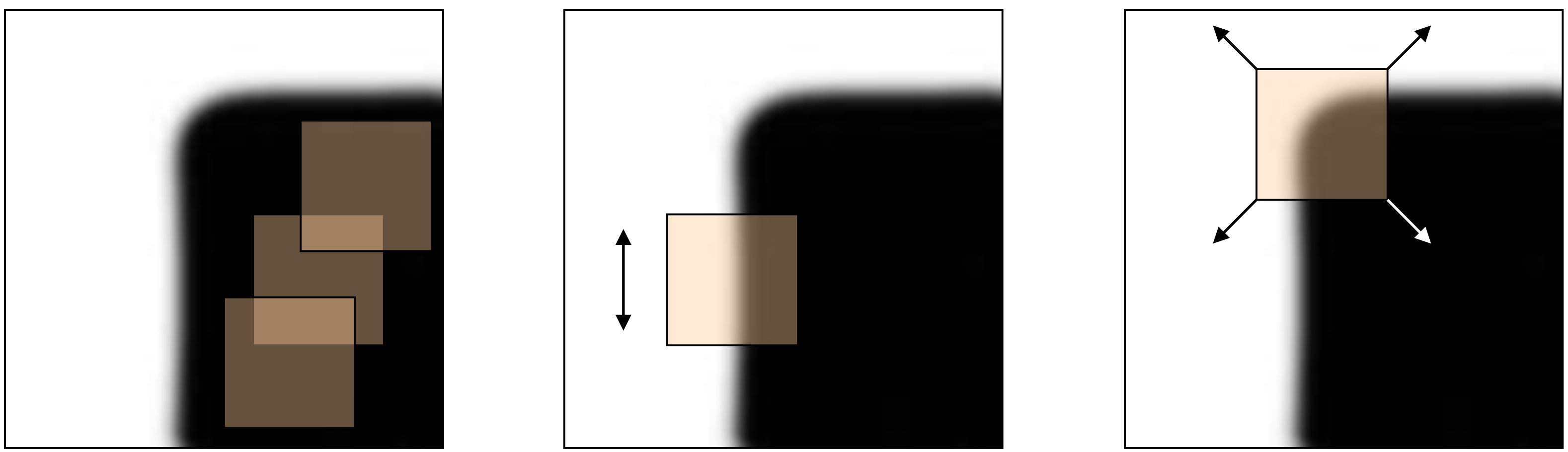
"edge":
no change along
the edge direction

"corner":
significant change
in all directions

A. Efros, D. Frolova, D. Simakov

Corners as distinctive interest points

- We should easily recognize the keypoint 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

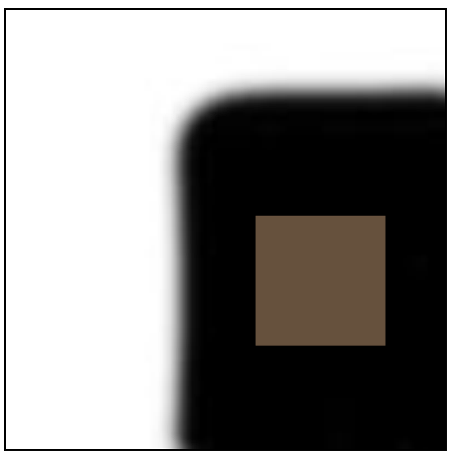
"edge":
no change along
the edge direction

"corner":
significant change
in all directions

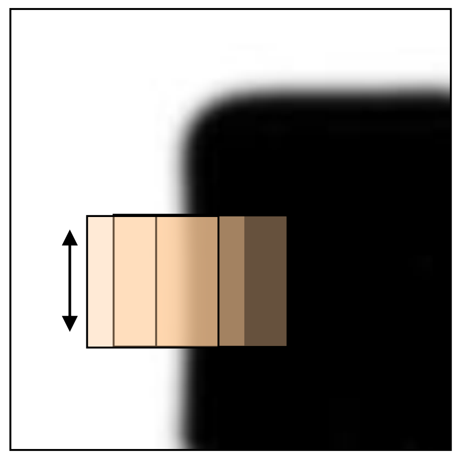
Adapted from A. Efros, D. Frolova, D. Simakov

Corners as distinctive interest points

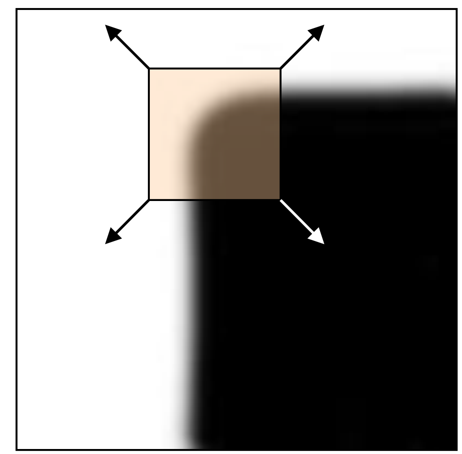
- We should easily recognize the keypoint by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in
all directions



"edge":
no change along
the edge direction

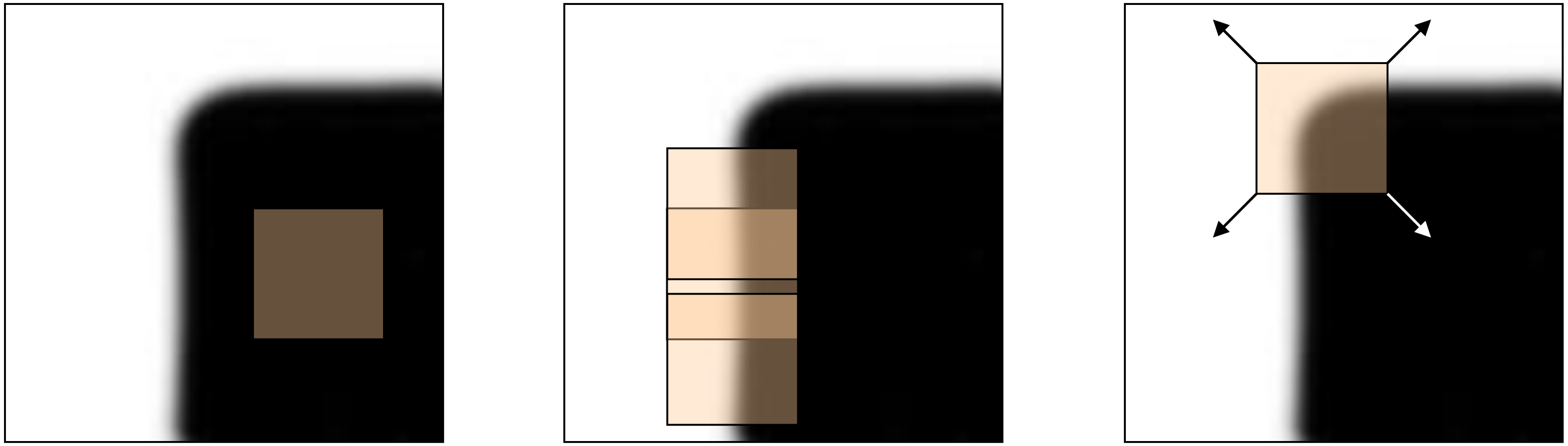


"corner":
significant change
in all directions

Adapted from A. Efros, D. Frolova, D. Simakov

Corners as distinctive interest points

- We should easily recognize the keypoint 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

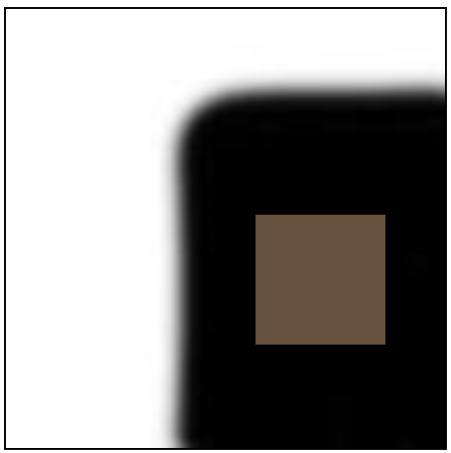
"edge":
no change along the edge direction

"corner":
significant change in all directions

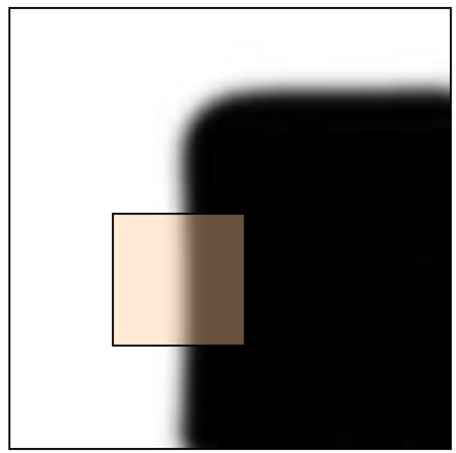
Adapted from A. Efros, D. Frolova, D. Simakov

Corners as distinctive interest points

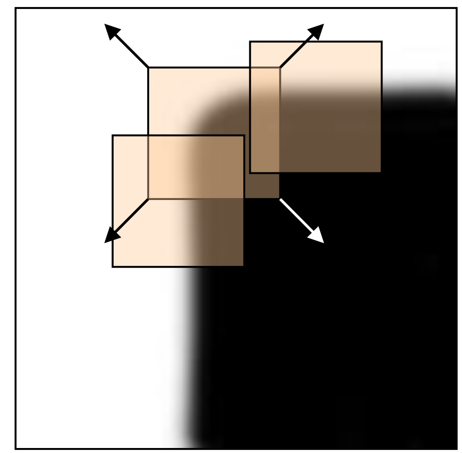
- We should easily recognize the keypoint by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in
all directions



"edge":
no change along
the edge direction



"corner":
significant change
in all directions

Adapted from A. Efros, D. Frolova, D. Simakov

What points would you choose?

K. Grauman

Harris Detector: Mathematics

Window-averaged squared change of intensity induced by shifting the image data by $[u,v]$:

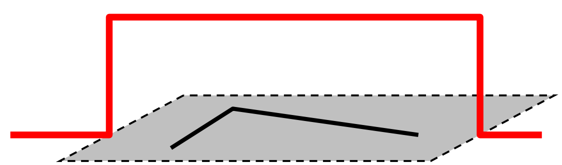
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function → $w(x,y)$

Shifted intensity → $I(x+u,y+v)$

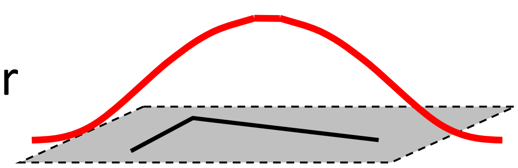
Intensity → $I(x,y)$

Window function $w(x,y)$ =



1 in window, 0 outside

or



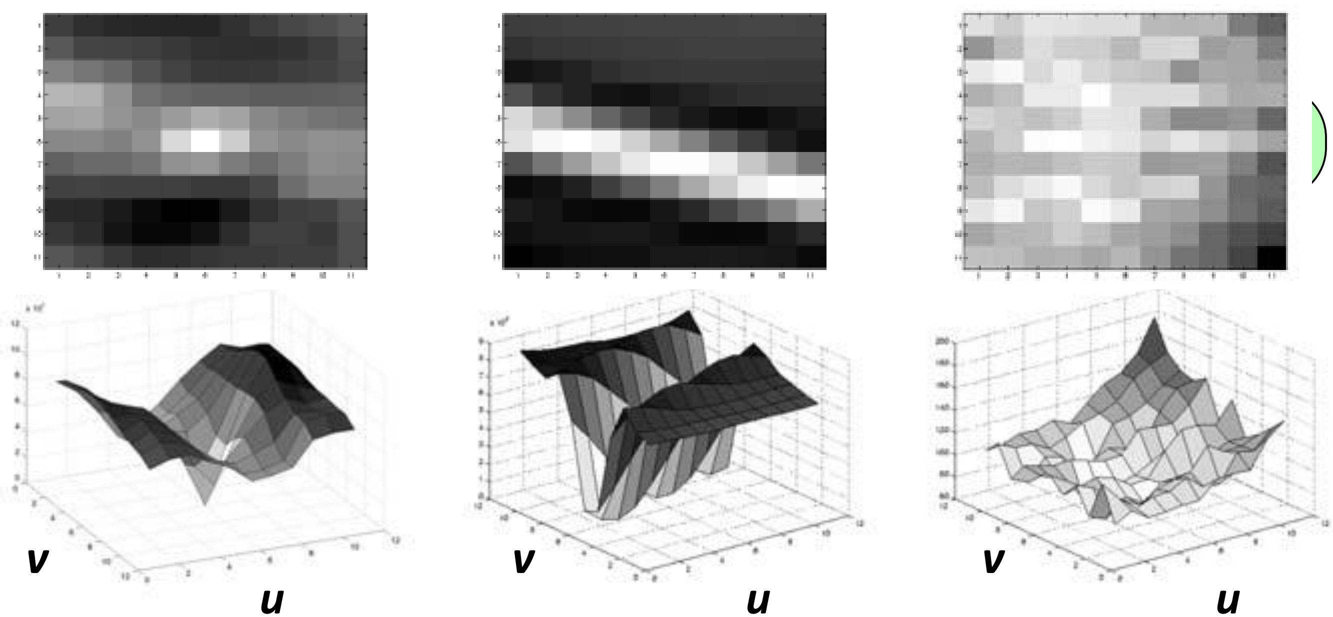
Gaussian

Harris Detector: Mathematics

Window-averaged squared change of intensity induced by shifting the image data by $[u,v]$:

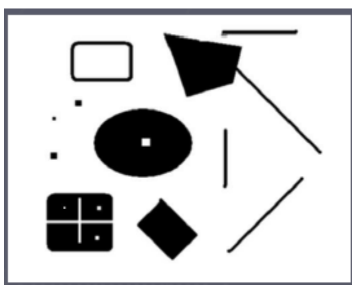
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

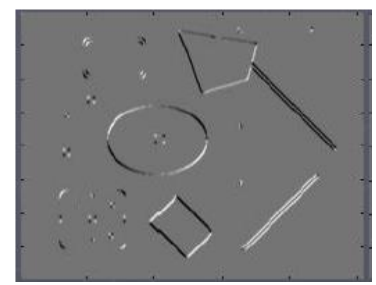
$E(u, v)$



Harris Detector: Mathematics

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





Notation:

$$I_x \Leftrightarrow \frac{\partial I}{\partial x} \qquad I_y \Leftrightarrow \frac{\partial I}{\partial y} \qquad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$$

K. Grauman

Harris Detector: Mathematics

Expanding I(x,y) in a Taylor series expansion, we have, for small shifts [u,v], a quadratic approximation to the error surface between a patch and itself, shifted by [u,v]:

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

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

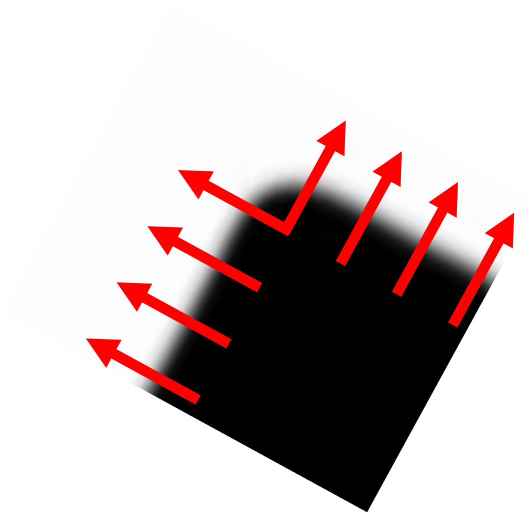
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

D. Frolova, D. Simakov

What does the matrix M reveal?

Since *M* is symmetric, we have

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

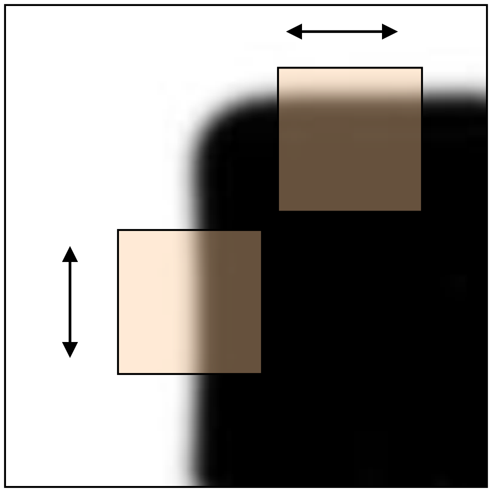


$$Mx_i = \lambda_i x_i$$

The *eigenvalues* of *M* reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.

K. Grauman

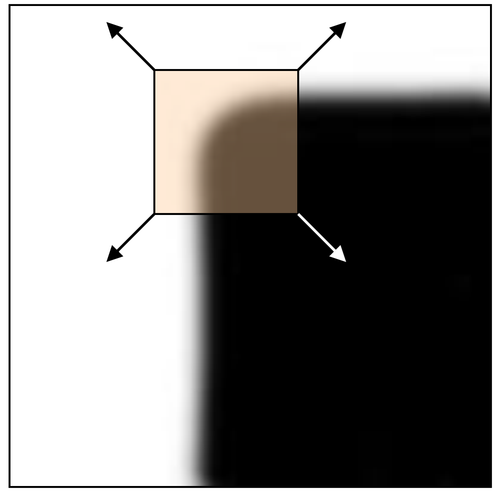
Corner response function



"edge":

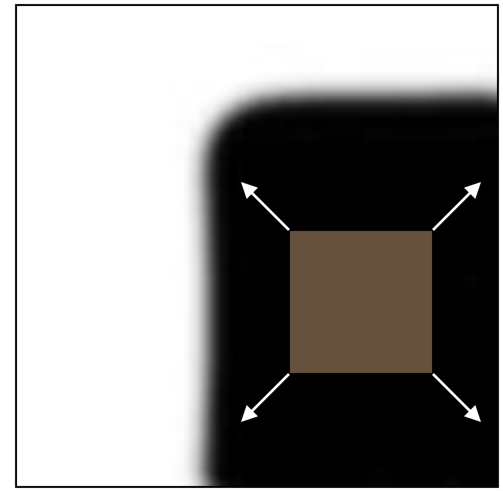
$\lambda_1 >> \lambda_2$

$\lambda_2 >> \lambda_1$



"corner":

λ_1 and λ_2 are large,
$\lambda_1 \sim \lambda_2$



"flat" region:

λ_1 and λ_2 are small

Adapted from A. Efros, D. Frolova, D. Simakov, K. Grauman

Harris Detector: Mathematics

Measure of corner response:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

D. Frolova, D. Simakov

Harris Detector: Algorithm

- Compute image gradients I_x and I_y for all pixels
- For each pixel
  - Compute $$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

    by looping over neighbors x, y
  - compute $$R = \det M - k\left(\operatorname{trace} M\right)^2$$

How can I write this without using w(x, y), if window function is just 1 inside, 0 outside?

(k :empirical constant, k = 0.04-0.06)

- Find points with large corner response function R (R > threshold)
- Non-max suppression: Take the points of locally maximum R as the detected feature points (i.e., pixels where R is bigger than for all the 4 or 8 neighbors)

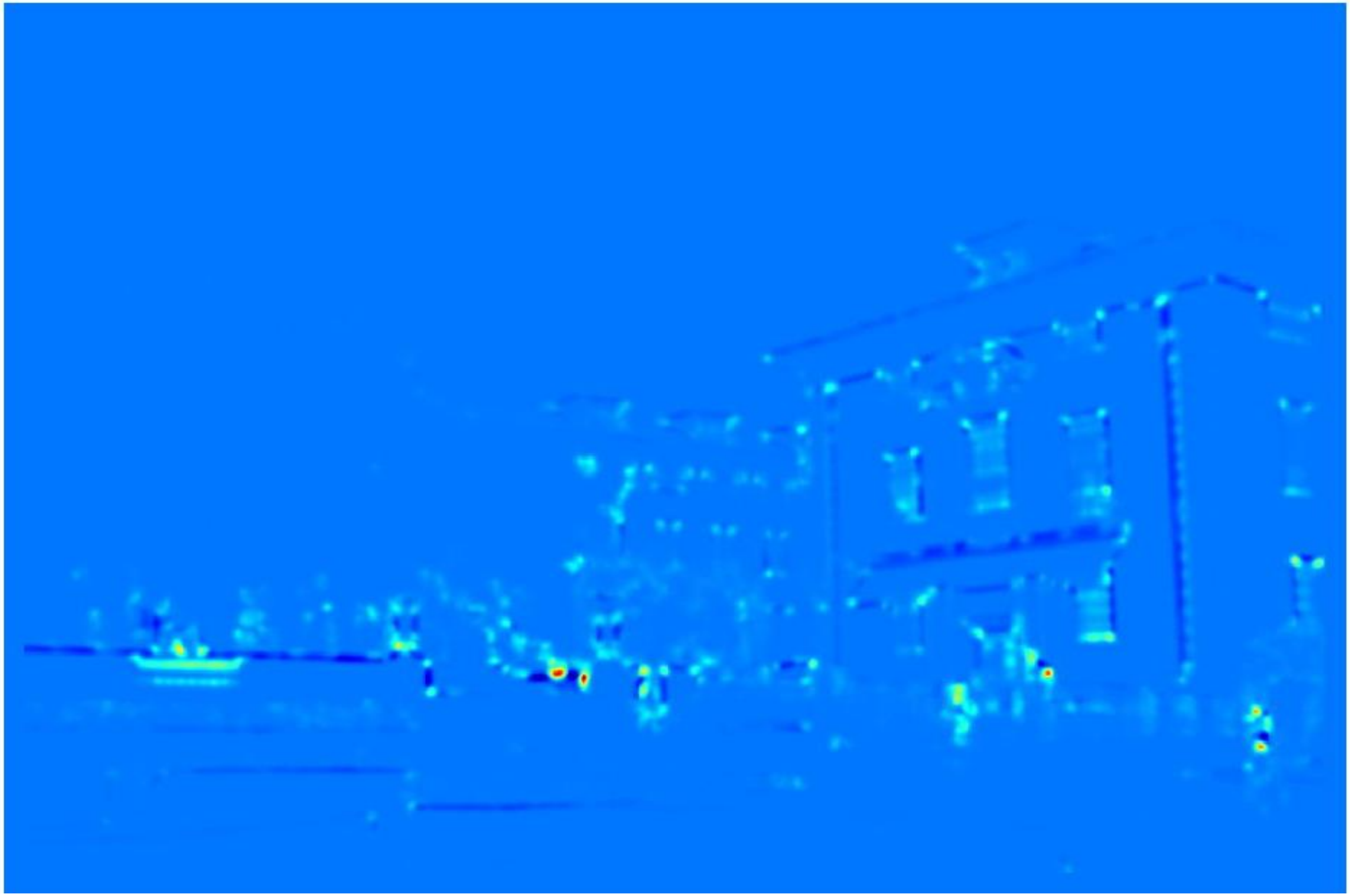
D. Frolova, D. Simakov

Example of Harris application

K. Grauman

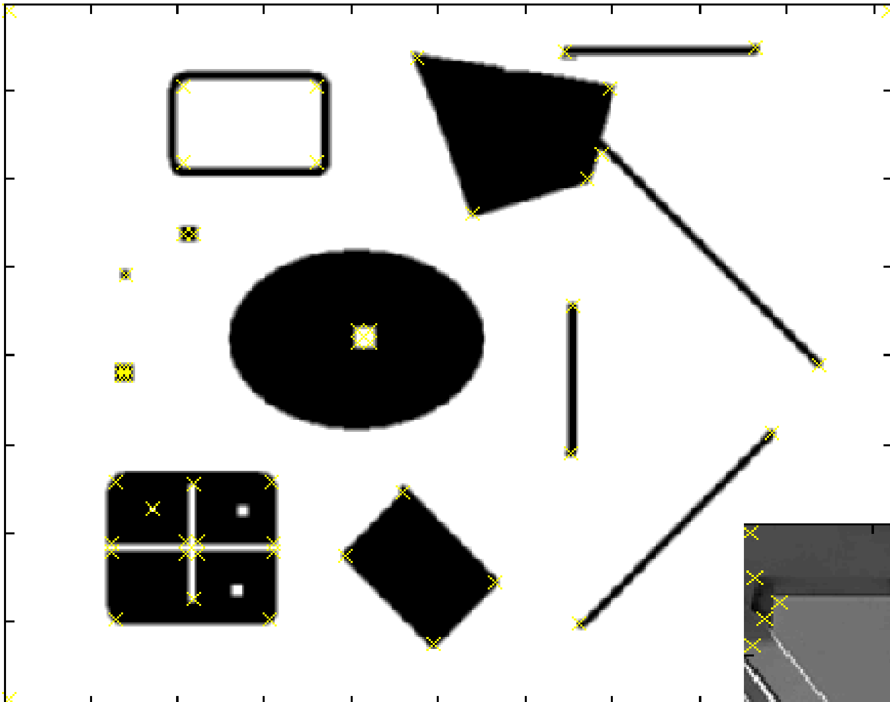
Example of Harris application

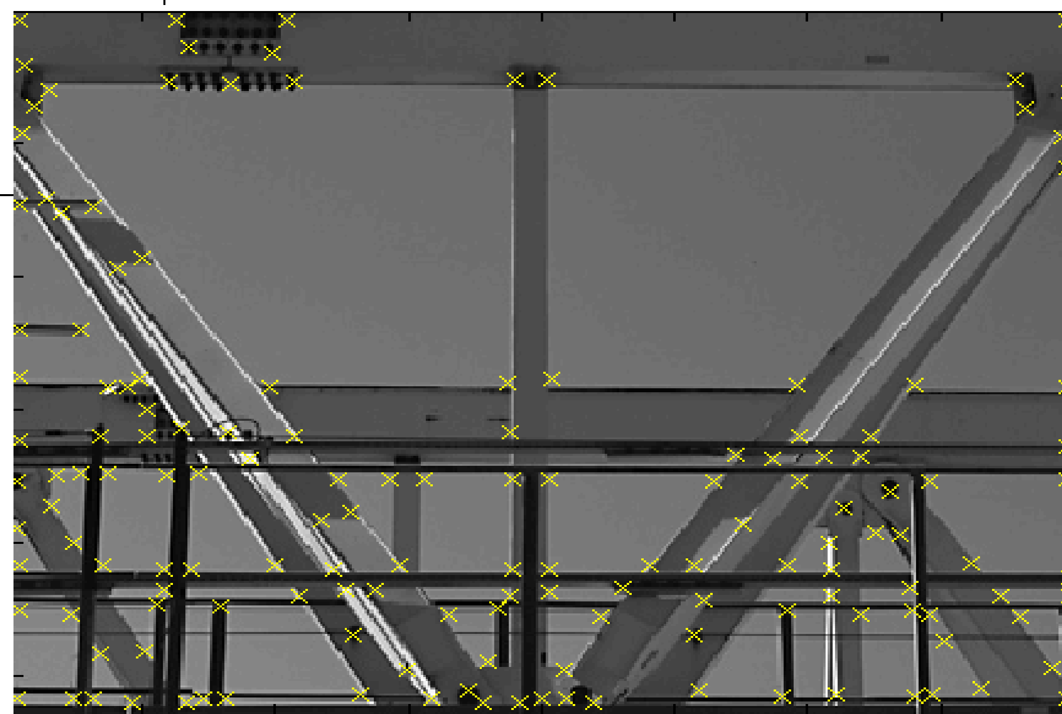
- Corner response at every pixel



K. Grauman

More Harris responses





***Effect:* A very precise corner detector.**

D. Hoiem

More Harris responses

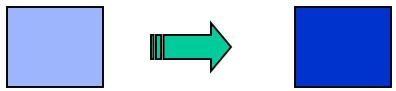
D. Hoiem

Properties: Invariance vs covariance

"A function is *invariant* under a certain family of transformations if its value does not change when a transformation from this family is applied to its argument. A function is *covariant* when it commutes with the transformation, i.e., applying the transformation to the argument of the function has the same effect as applying the transformation to the output of the function. […] [For example,] the area of a 2D surface is invariant under 2D rotations, since rotating a 2D surface does not make it any smaller or bigger.
But the orientation of the major axis of inertia of the surface is covariant under the same family of transformations, since rotating a 2D surface will affect the orientation of its major axis in exactly the same way."

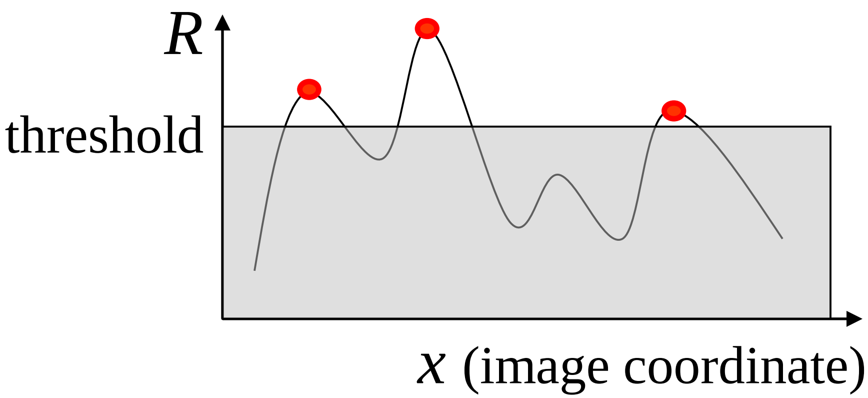
"Local Invariant Feature Detectors: A Survey" by Tinne Tuytelaars and Krystian Mikolajczyk,
in *Foundations and Trends in Computer Graphics and Vision* Vol. 3, No. 3 (2007) 177–280
Chapter 1, 3.2, 7 http://homes.esat.kuleuven.be/%7Etuytelaa/FT_survey_interestpoints08.pdf

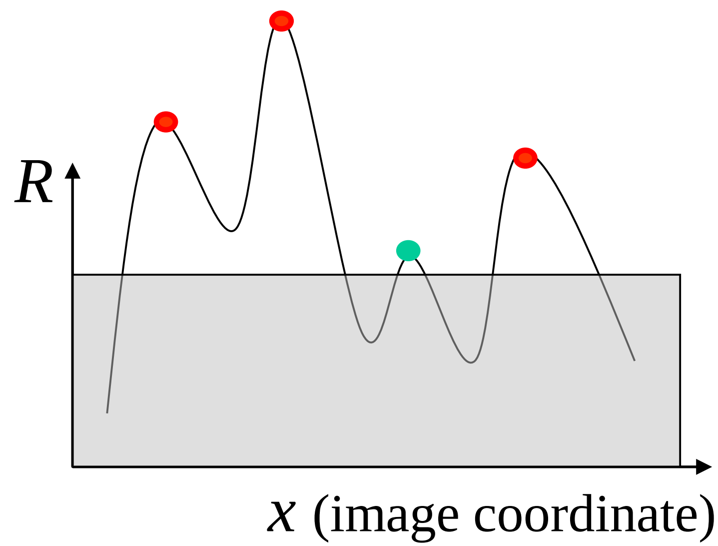
What happens if: Affine intensity change



$$I \to a\,I + b$$

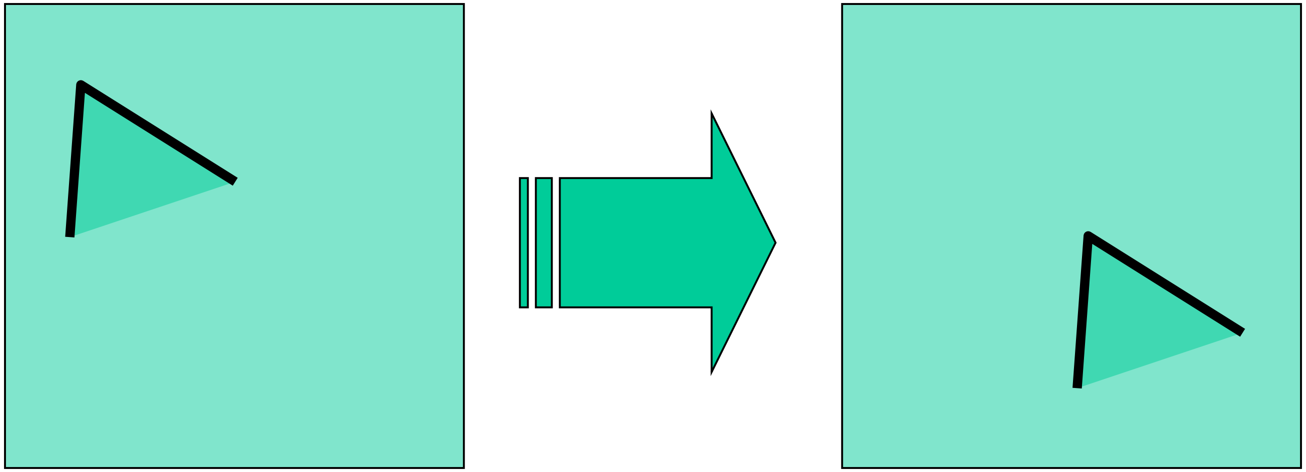
- Only derivatives are used => invariance to intensity shift $I \to I + b$
- Intensity scaling: $I \to a\,I$





Partially invariant to affine intensity change

What happens if: Image translation

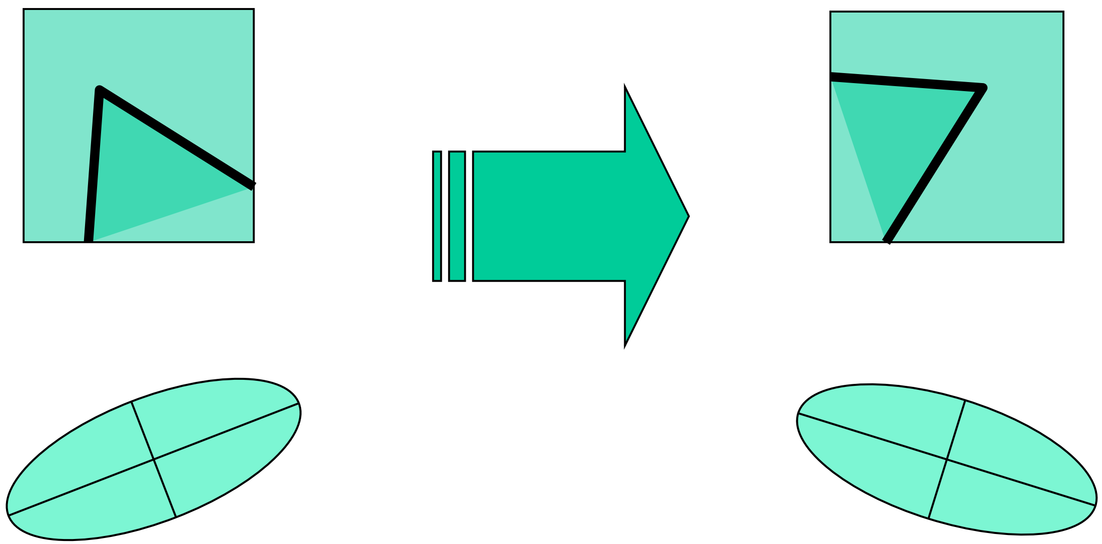


- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation (on image level), corner response is invariant (on patch level)

Adapted from L. Lazebnik

What happens if: Image rotation



Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation (on image level), corner response is invariant (on patch level)

Adapted from L. Lazebnik

What happens if: Scaling

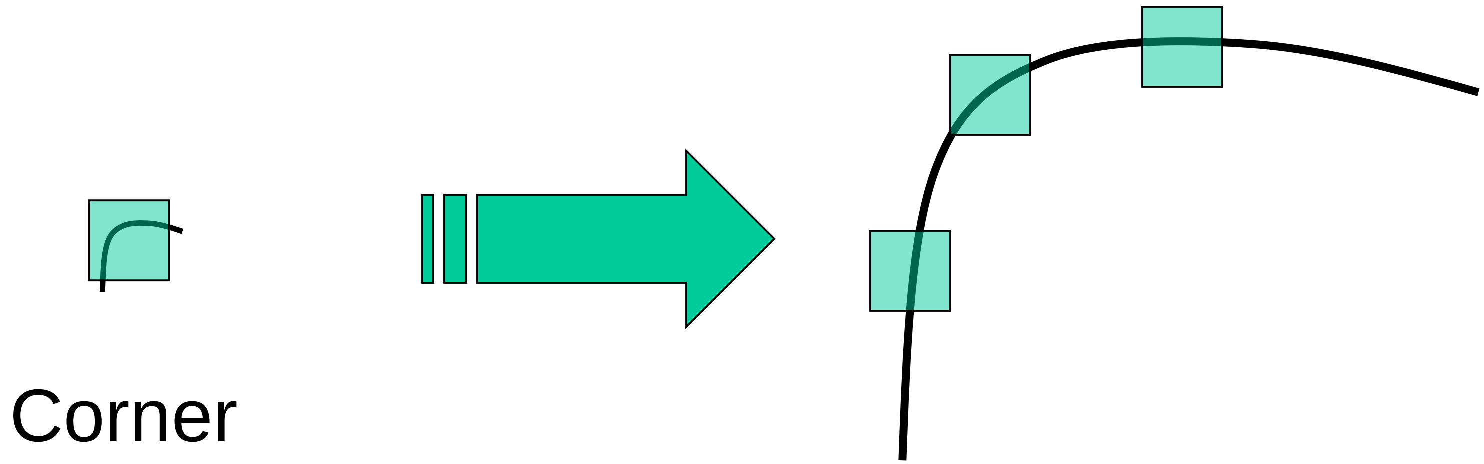
Invariant to image scale?

image

zoomed image

A. Torralba

What happens if: Scaling



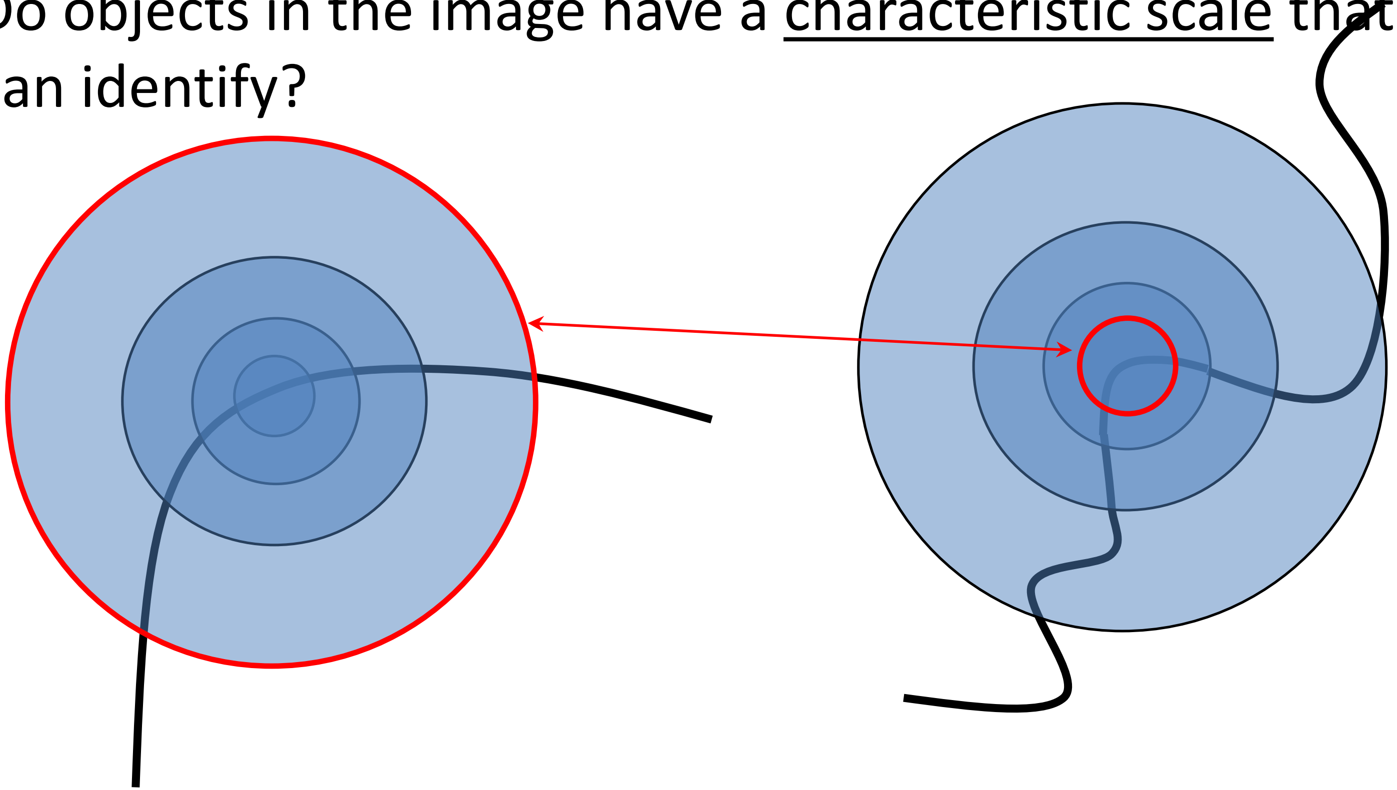
All points will be classified as edges

Corner location is not covariant to scaling!

L. Lazebnik

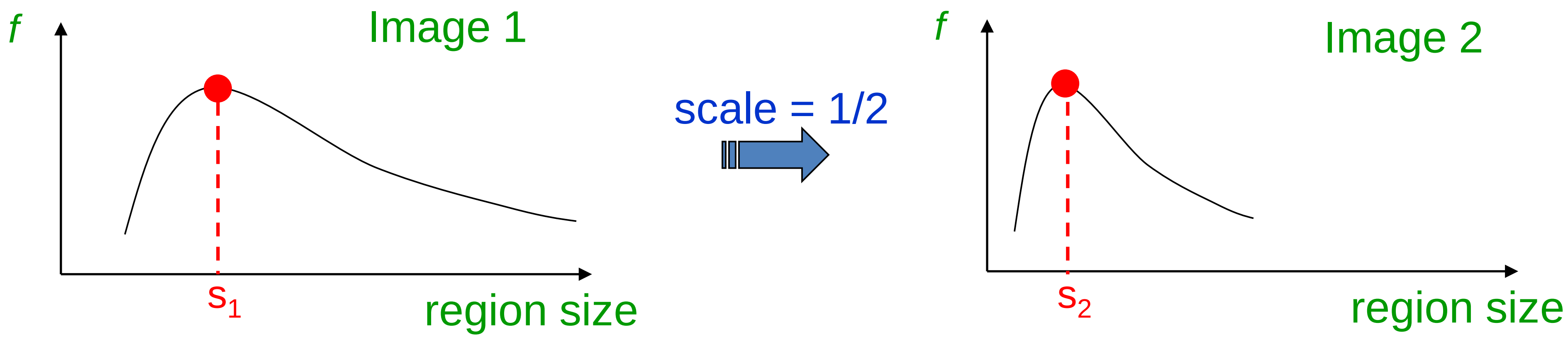
Scale invariant detection

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



Scale invariant detection

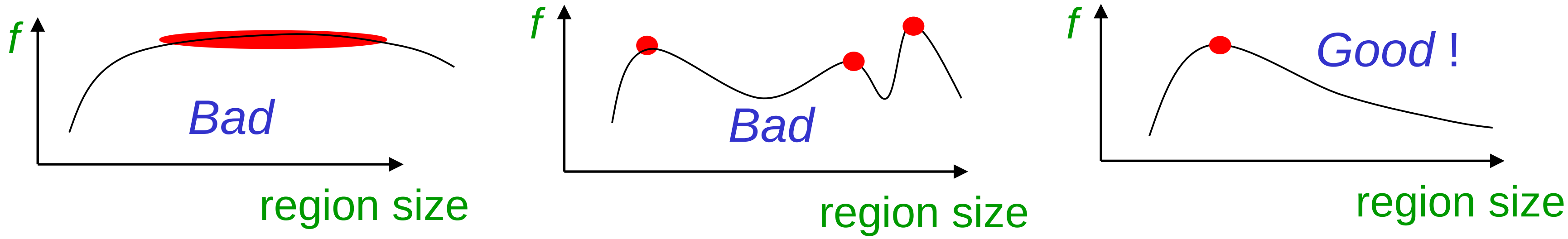
- Solution:
  - Design a function on the region which is “scale invariant” (has the same shape even if the image is resized)
  - Take a local maximum of this function



Adapted from A. Torralba

Scale invariant detection

- A “good” function for scale detection:
  has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

A. Torralba

Automatic scale selection

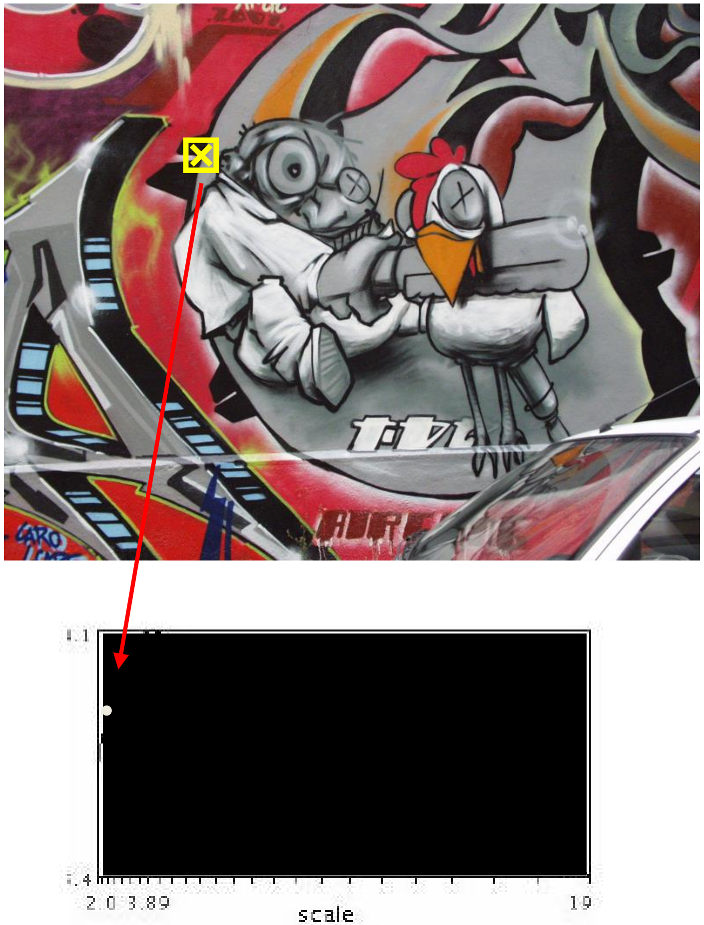
$$f(I_{i_1 \ldots i_m}(x,\sigma)) = f(I_{i_1 \ldots i_m}(x',\sigma'))$$

How to find corresponding patch sizes?

K. Grauman, B. Leibe

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

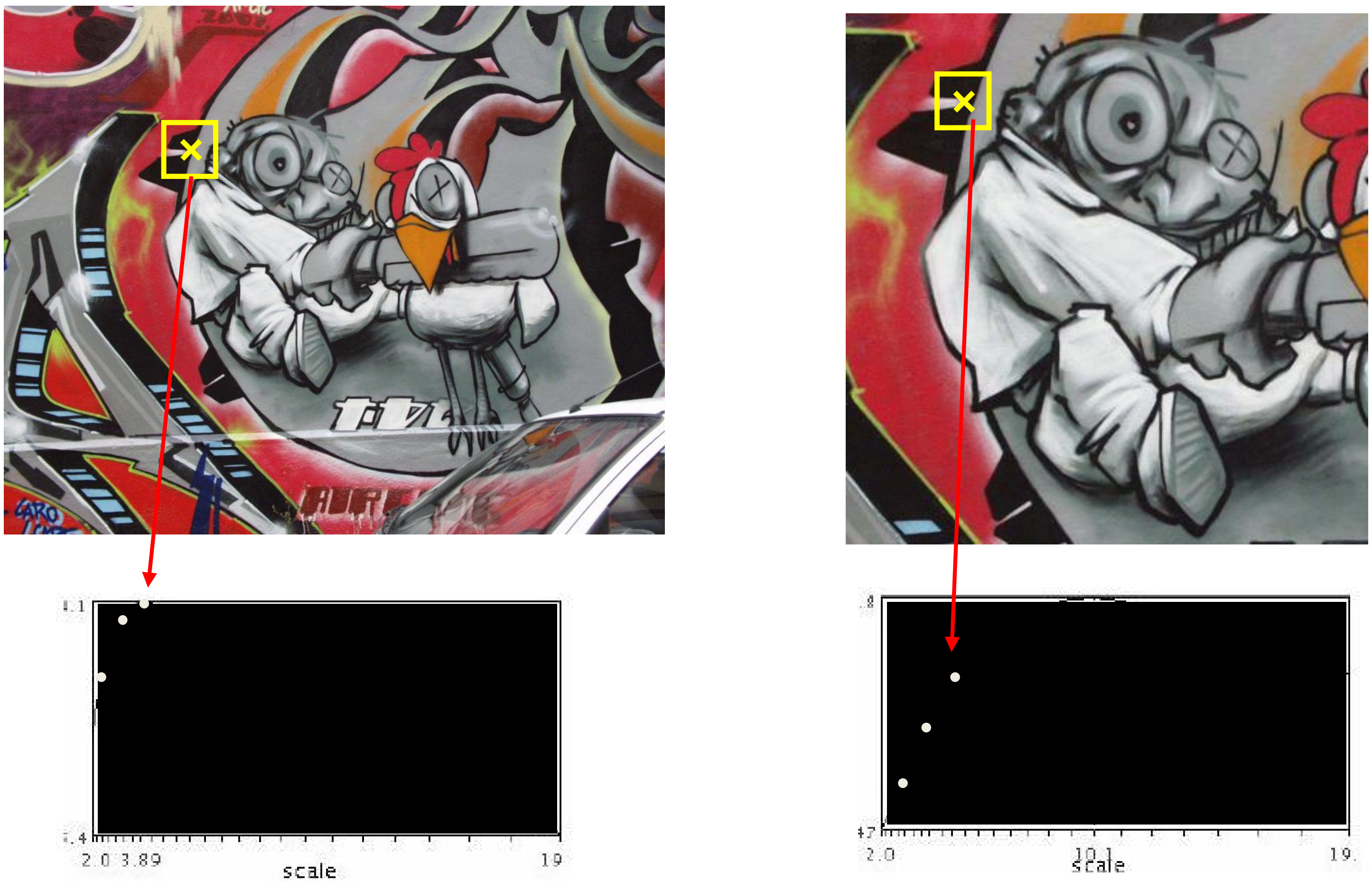
- Function responses for increasing scale (scale signature)

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

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

Automatic scale selection

- Function responses for increasing scale (scale signature)

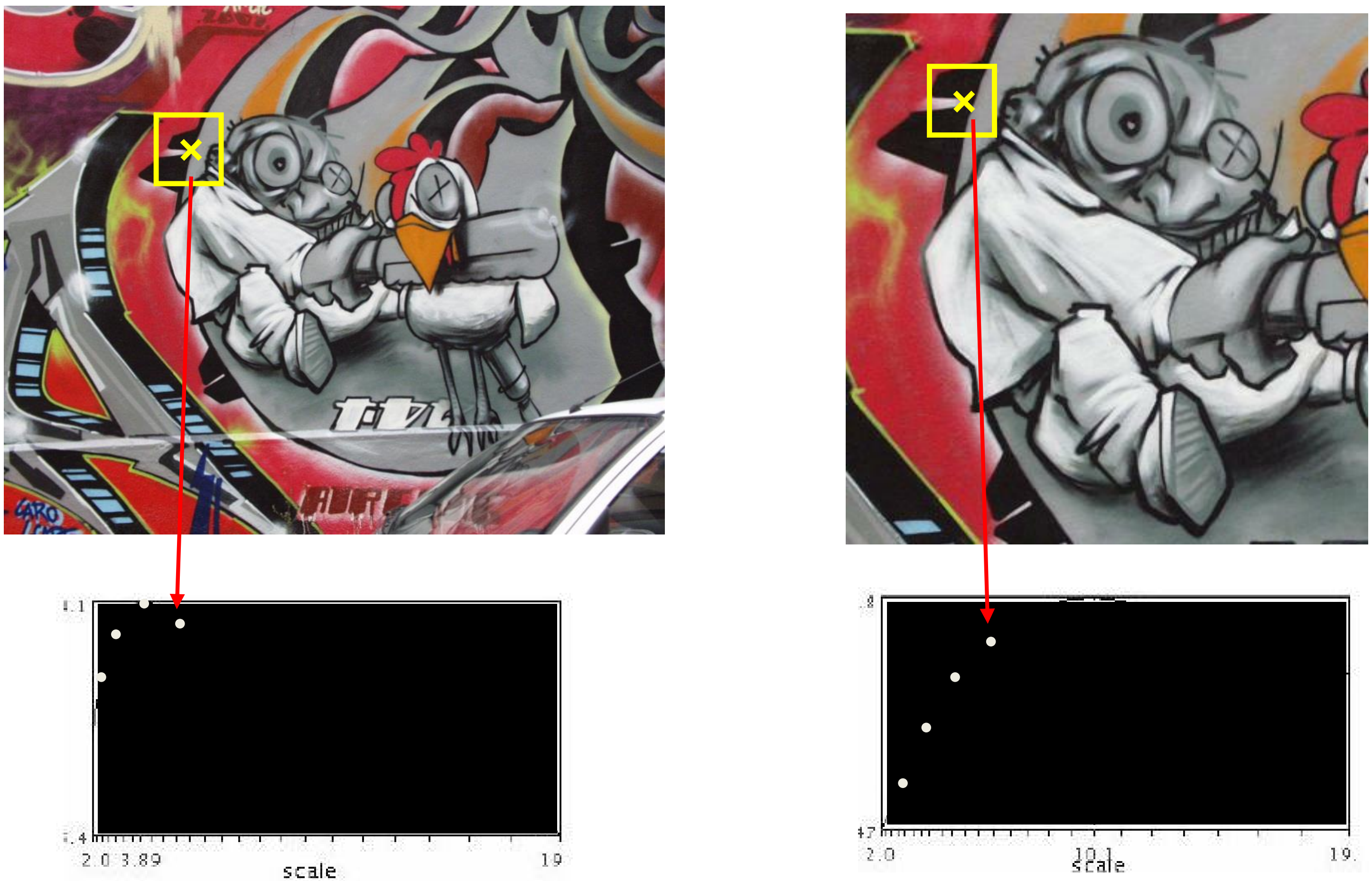


$f(I_{i_1 \ldots i_m}(x,\sigma))$ $f(I_{i_1 \ldots i_m}(x',\sigma))$

K. Grauman, B. Leibe

Automatic scale selection

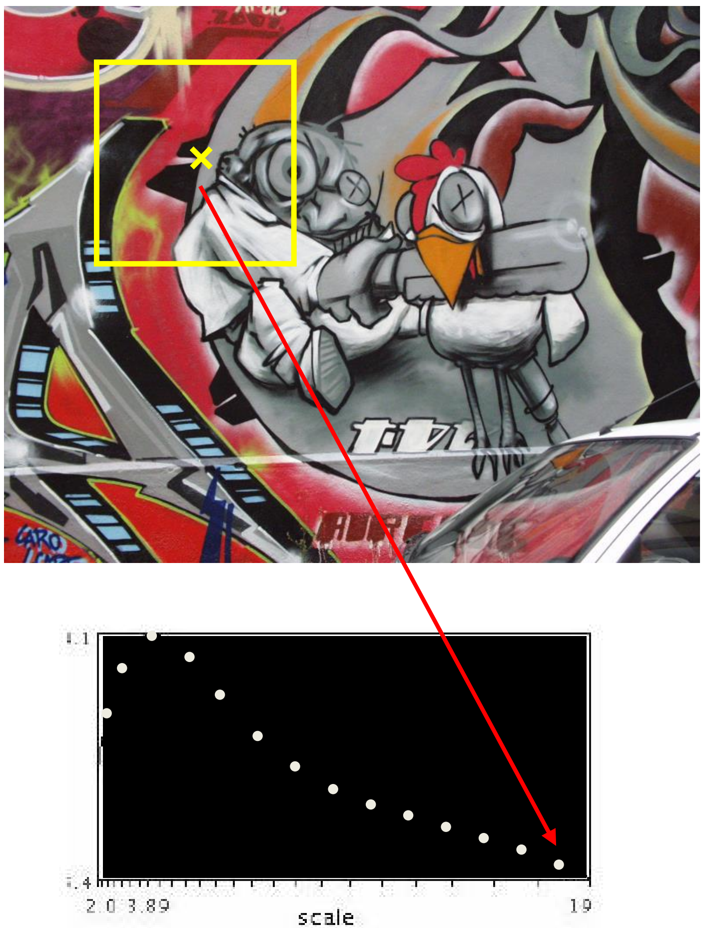
- Function responses for increasing scale (scale signature)



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

Automatic scale selection

- Function responses for increasing scale (scale signature)

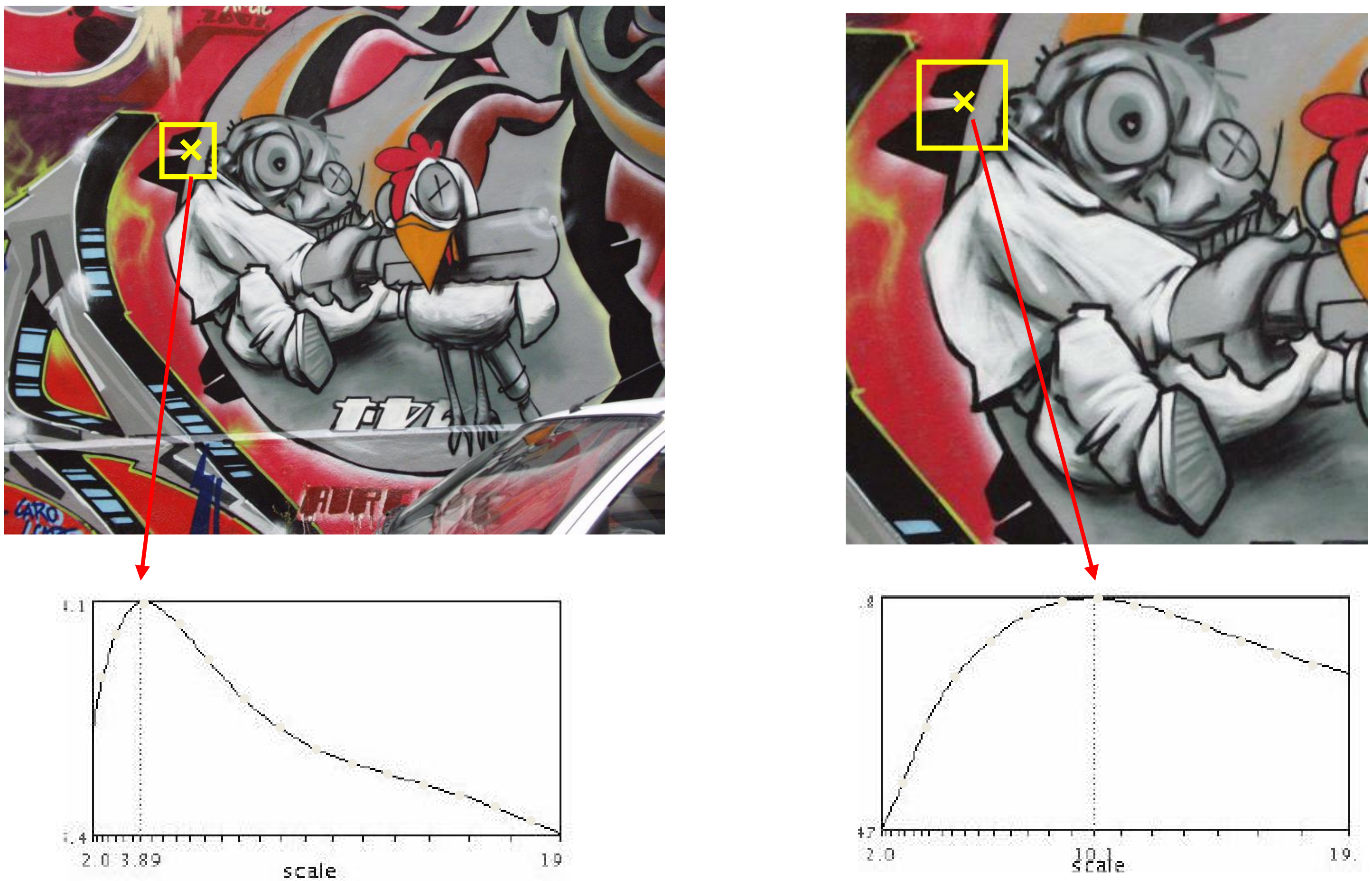


$f(I_{i_1 \ldots i_m}(x, \sigma))$

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

Automatic scale selection

- Function responses for increasing scale (scale signature)

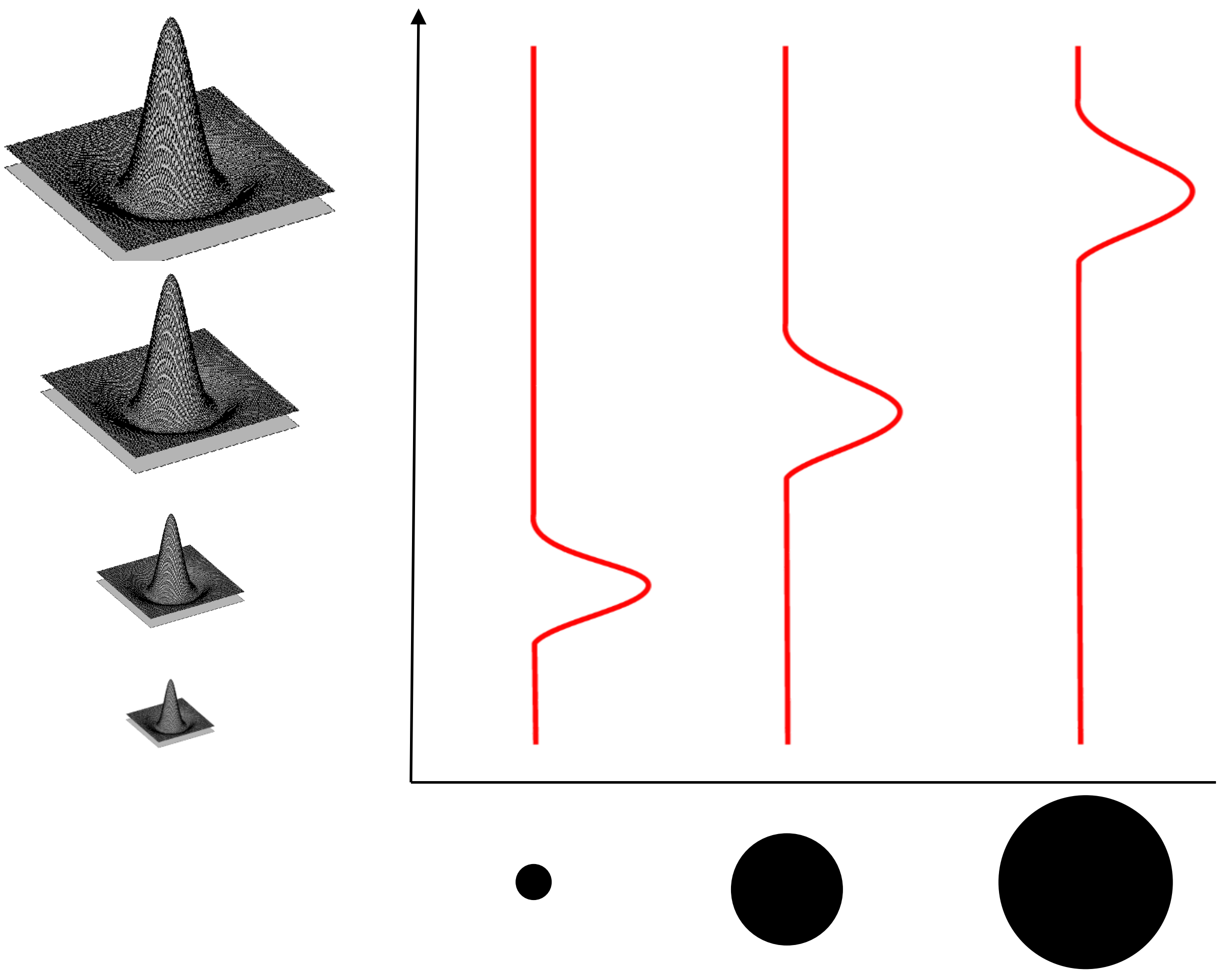


$f(I_{i_1 \ldots i_m}(x, \sigma))$ $\quad$ $f(I_{i_1 \ldots i_m}(x', \sigma'))$

K. Grauman, B. Leibe

What is a useful signature function?

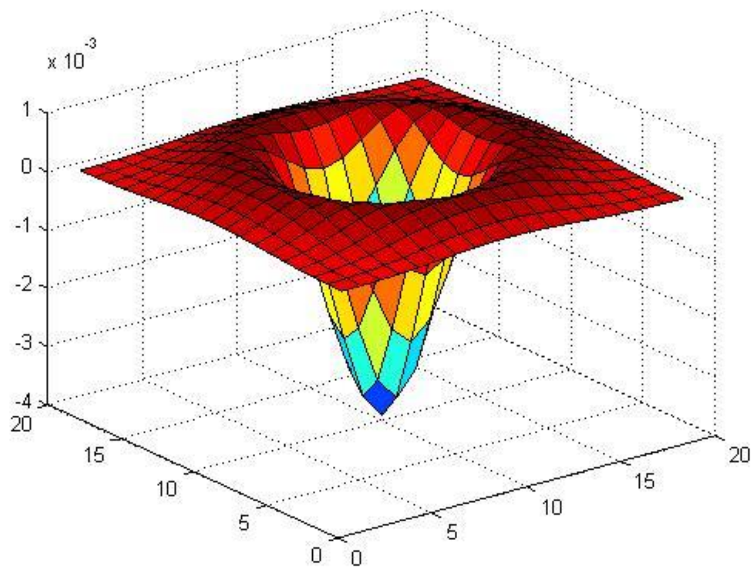
- Laplacian of Gaussian = “blob” detector

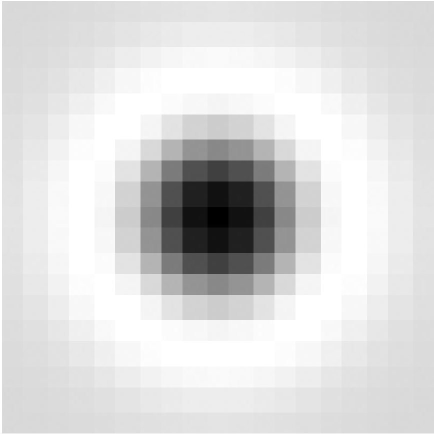


K. Grauman, B. Leibe

Blob detection in 2D

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D





$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

K. Grauman

Difference of Gaussian ≈ Laplacian

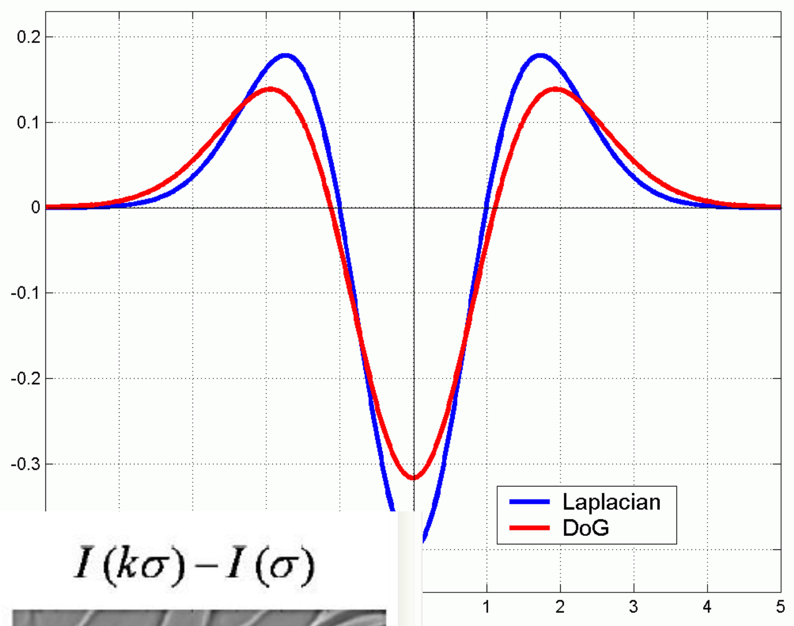
- We can approximate the Laplacian with a difference of Gaussians; more efficient to implement.

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

(Laplacian)

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

(Difference of Gaussians)



$I(k\sigma)$ $\qquad$ $I(\sigma)$ $\qquad$ $I(k\sigma) - I(\sigma)$

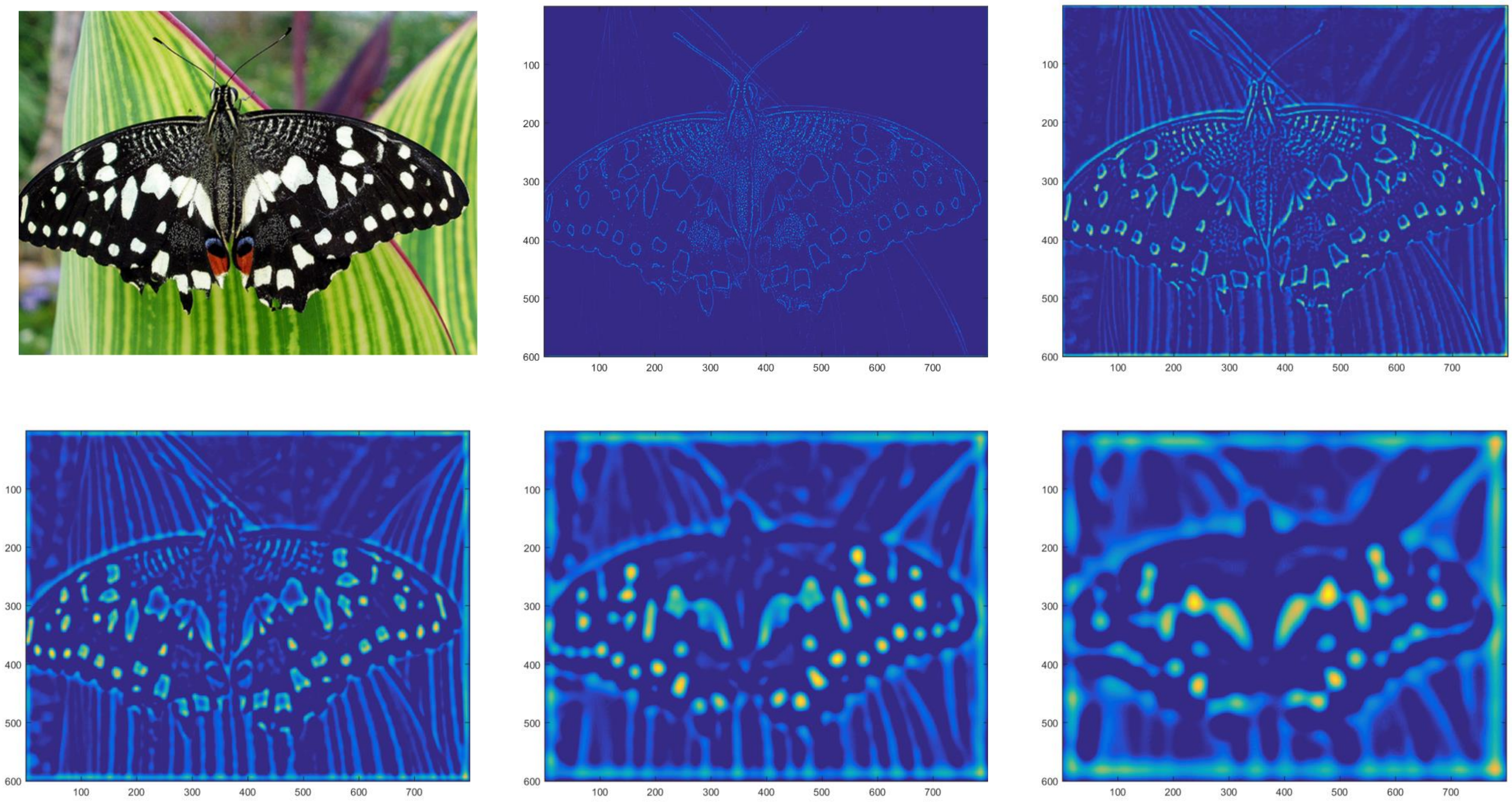
-

=

K. Grauman

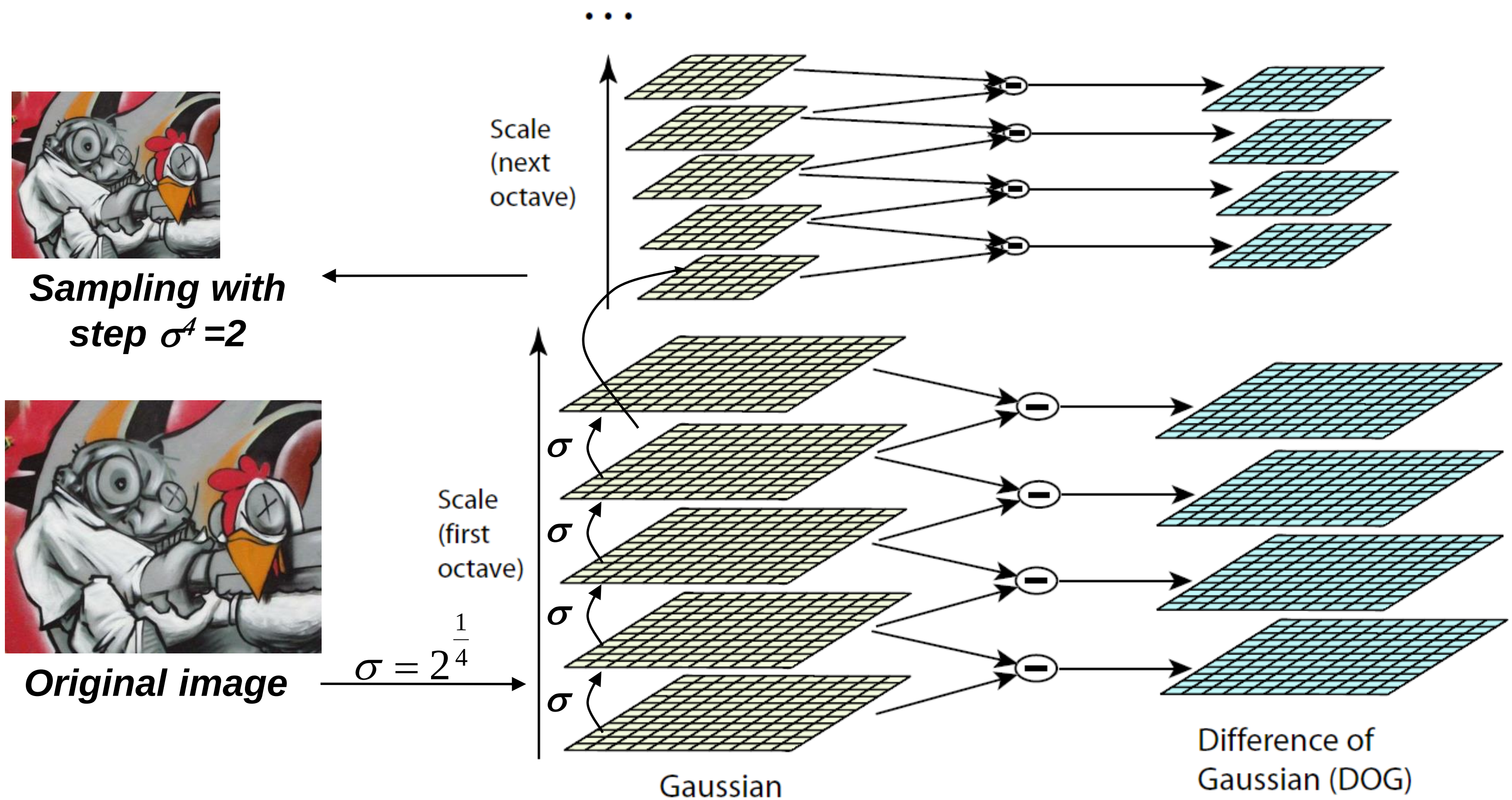
Laplacian pyramid example

- Allows detection of increasingly coarse detail



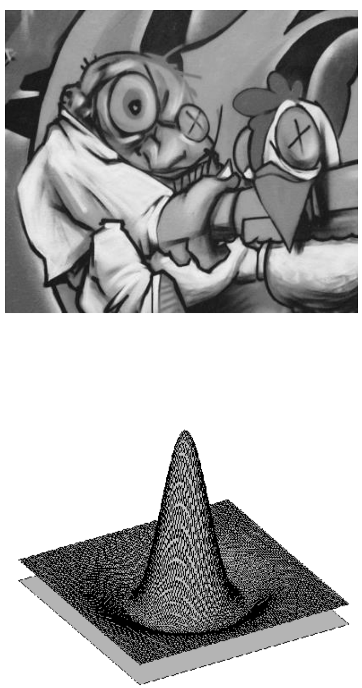
Difference of Gaussian: Efficient computation

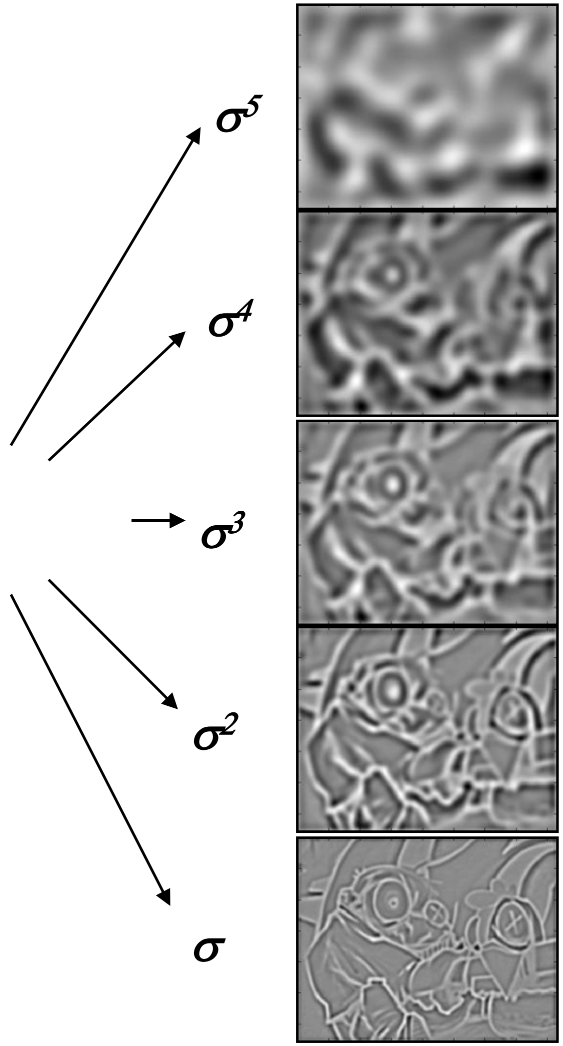
- Computation in Gaussian scale pyramid



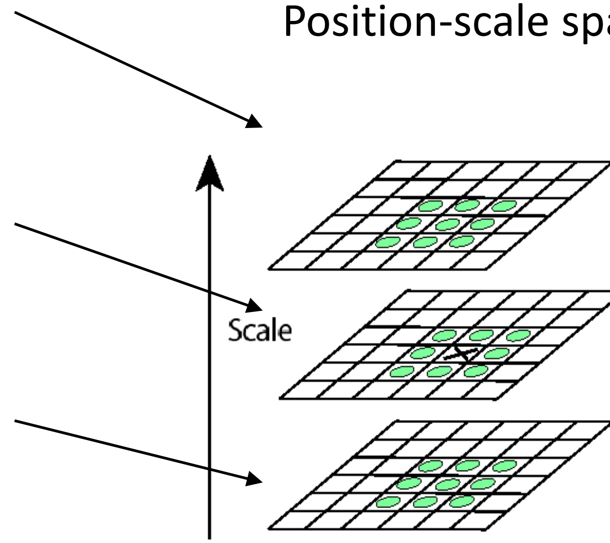
K. Grauman, B. Leibe

Find ***local maxima*** in position-scale space of Difference-of-Gaussian





Position-scale space:

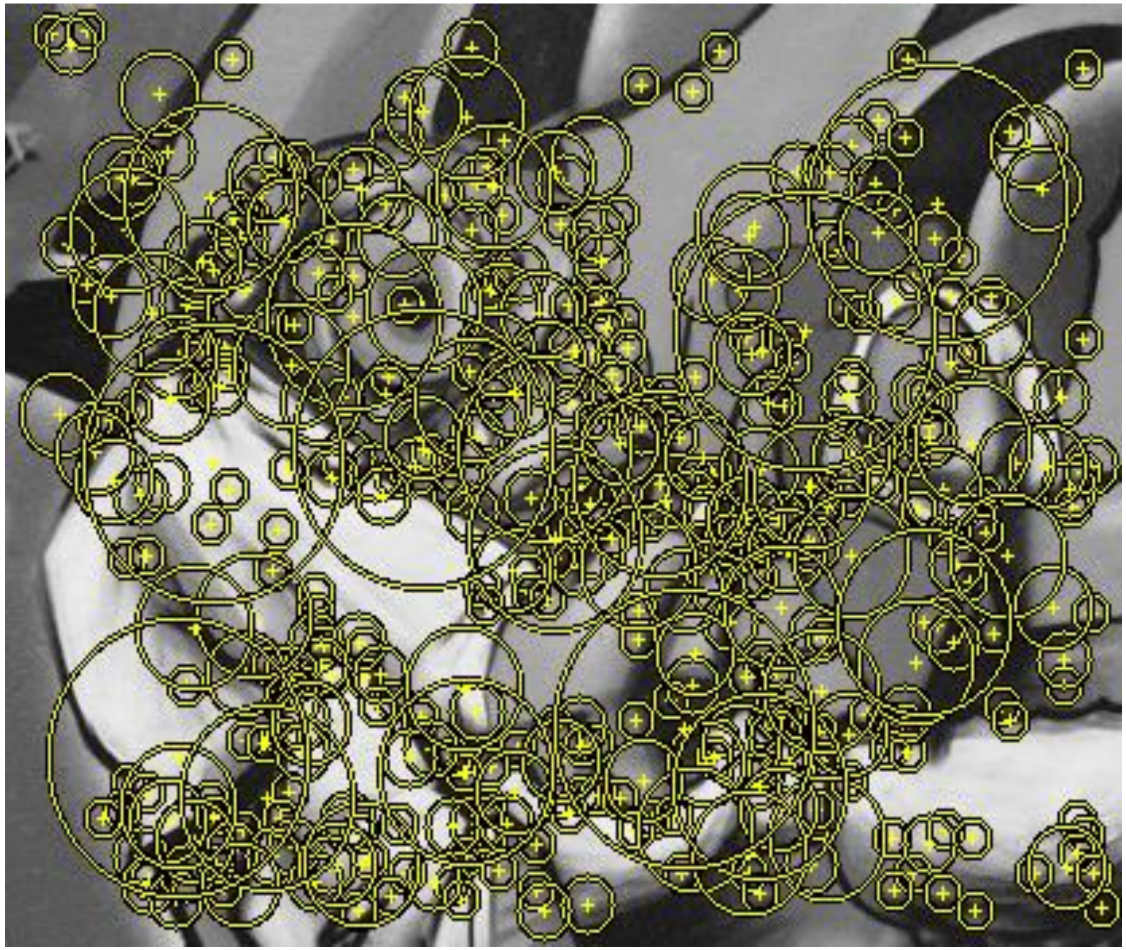


Find places where X greater than all of its neighbors (in green)

⇒ List of *(x, y, s)*

Adapted from K. Grauman, B. Leibe

Results: Difference-of-Gaussian

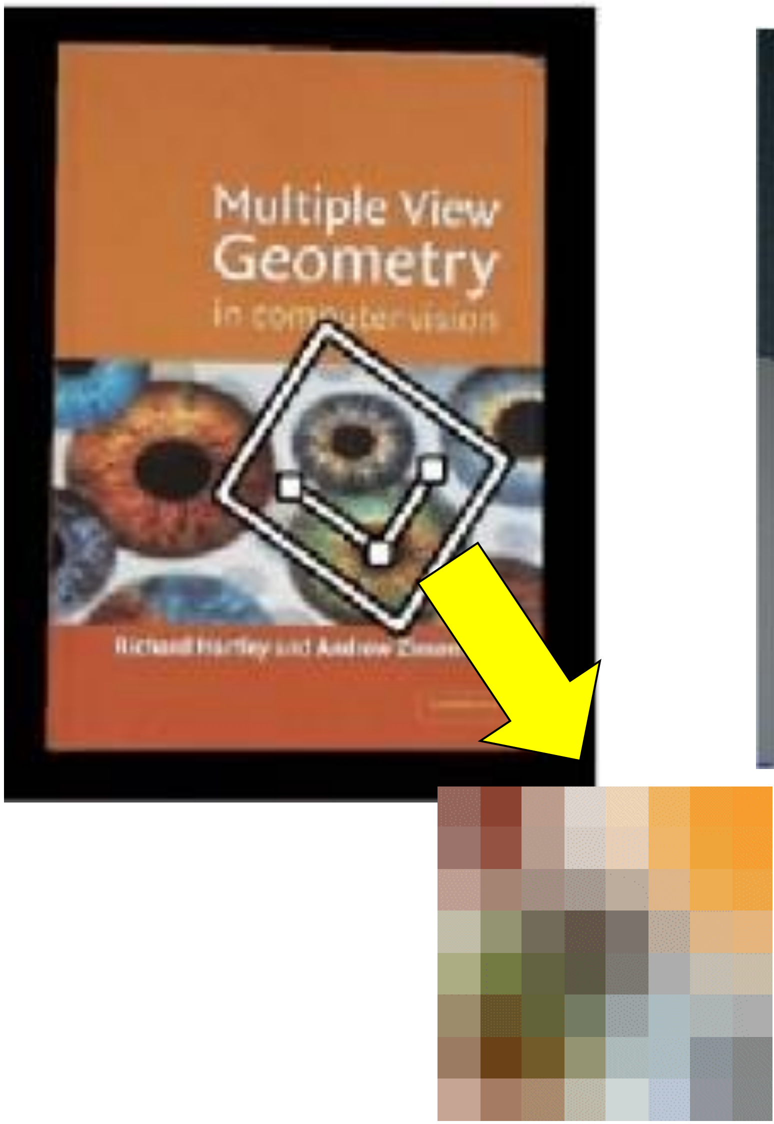


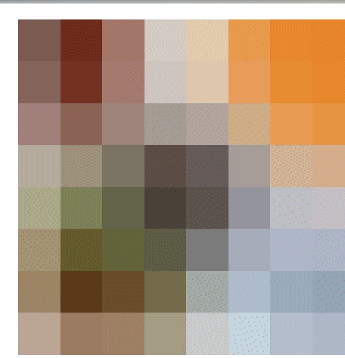
K. Grauman, B. Leibe

Plan for today

- Feature detection / keypoint extraction
  - Corner detection
  - Blob detection
- Feature description (of detected features)

Geometric transformations





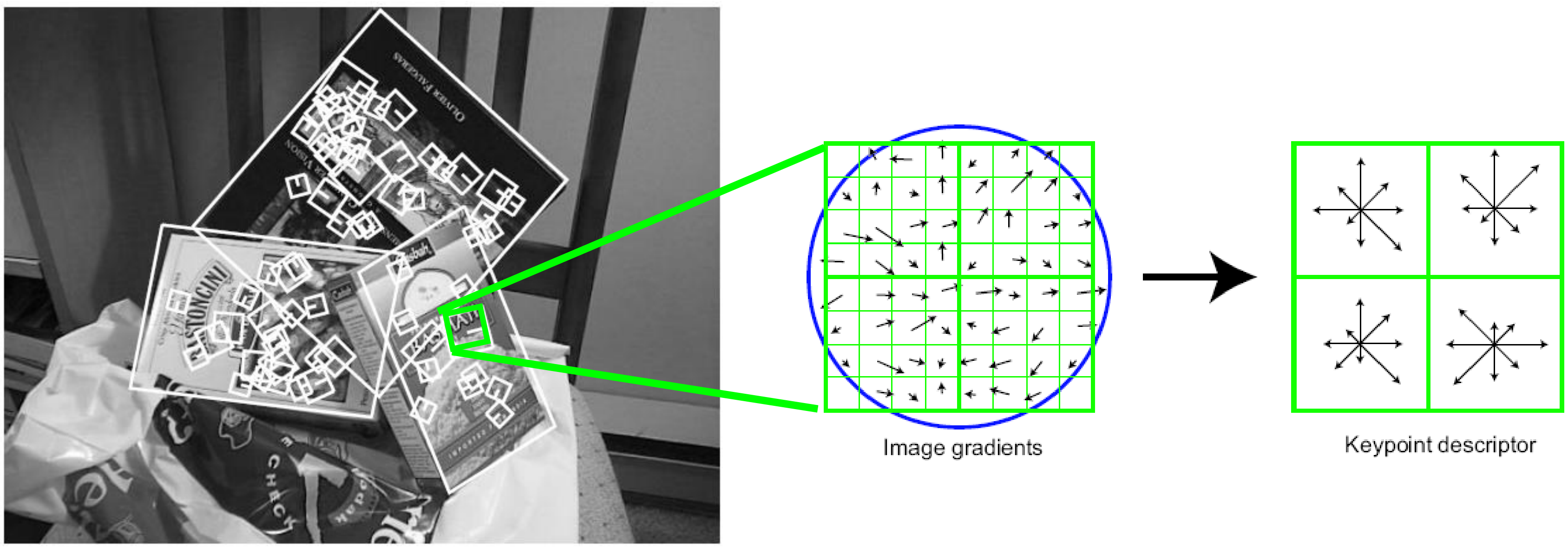
e.g. scale, translation, rotation

K. Grauman

Photometric transformations

T. Tuytelaars

Scale-Invariant Feature Transform (SIFT) descriptor



Histogram of oriented gradients

- **Captures important texture information**
- **Robust to small translations / affine deformations**

[Lowe, ICCV 1999]

K. Grauman, B. Leibe

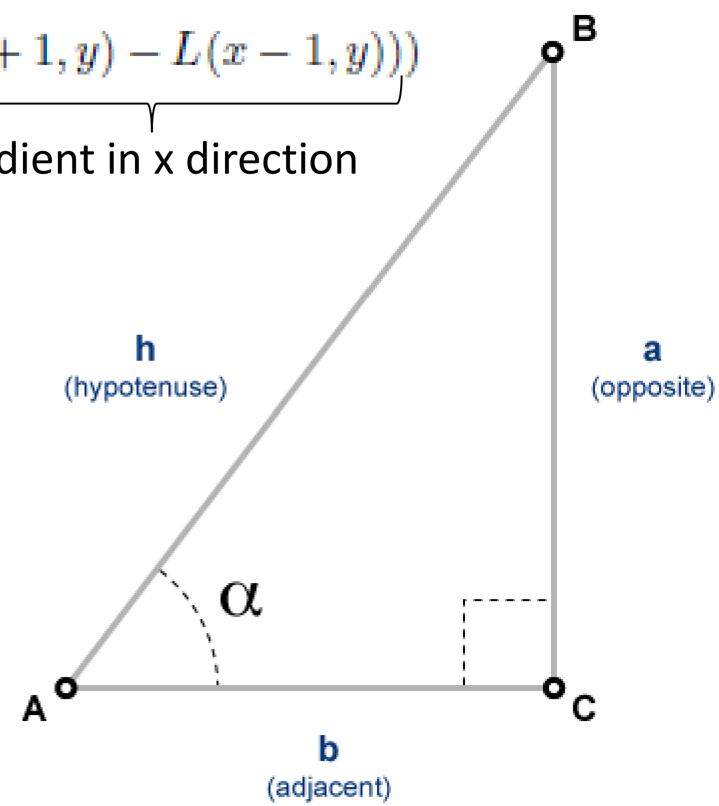
Computing gradients

L = the image intensity

$$m(x,y) = \sqrt{\underbrace{(L(x+1,y) - L(x-1,y))}_{\text{gradient in x direction}}^2 + \underbrace{(L(x,y+1) - L(x,y-1))}_{\text{gradient in y direction}}^2}$$

$$\theta(x,y) = \tan^{-1}(\underbrace{(L(x,y+1) - L(x,y-1))}_{\text{gradient in y direction}}/\underbrace{(L(x+1,y) - L(x-1,y))}_{\text{gradient in x direction}})$$

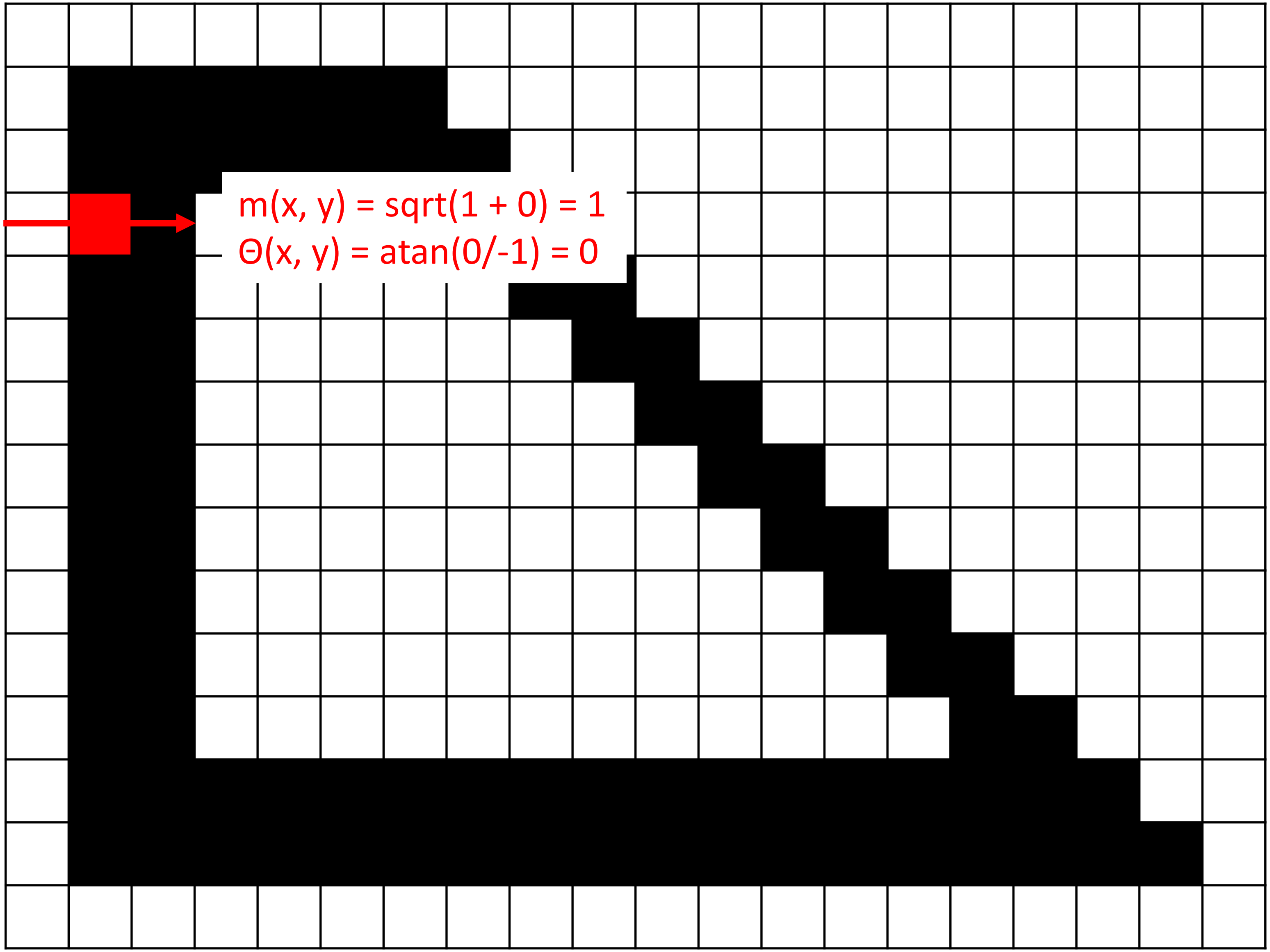
- $\tan(\alpha) = \dfrac{opposite\ side}{adjacent\ side}$



Gradients

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

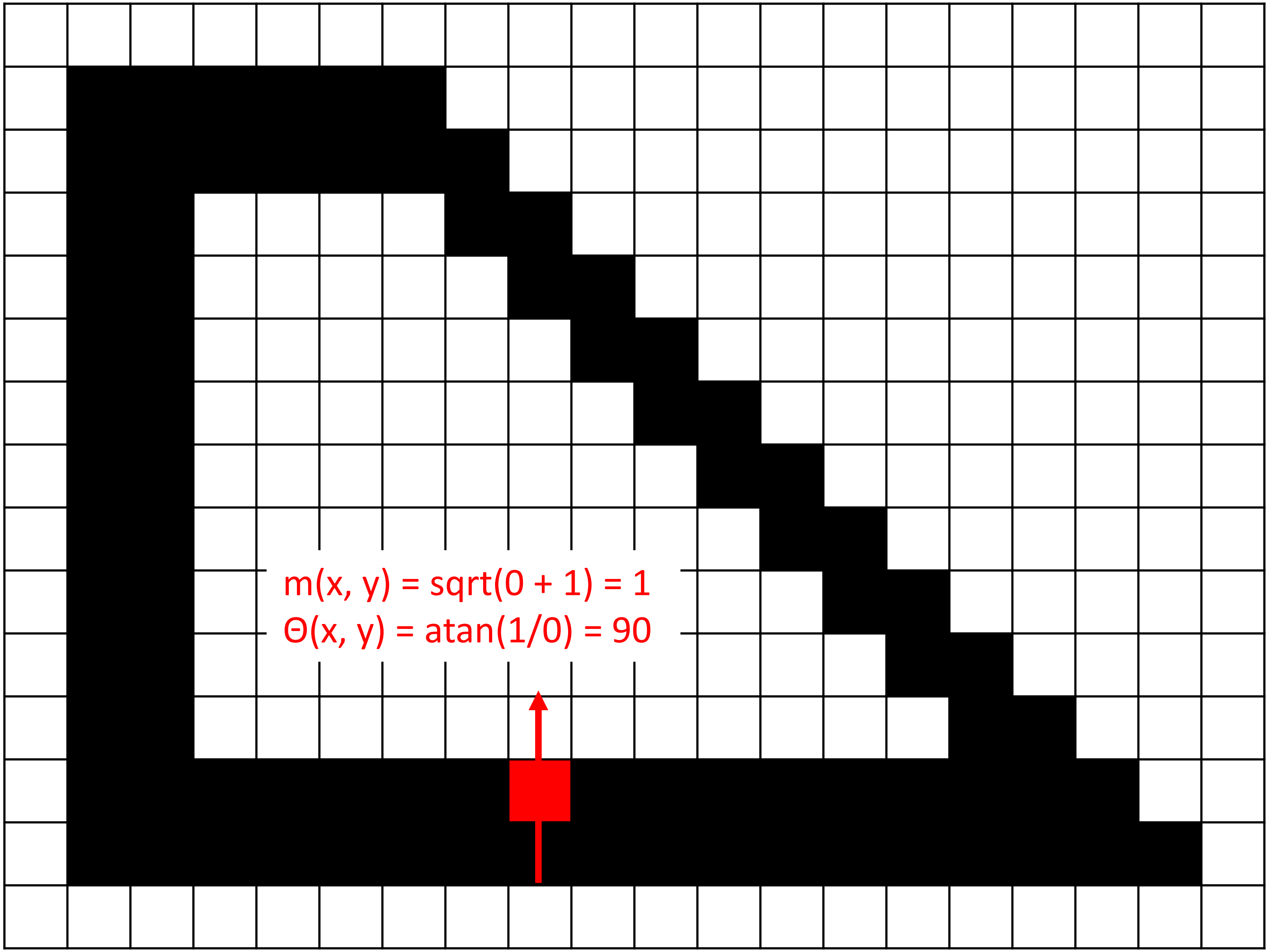
$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$



Gradients

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

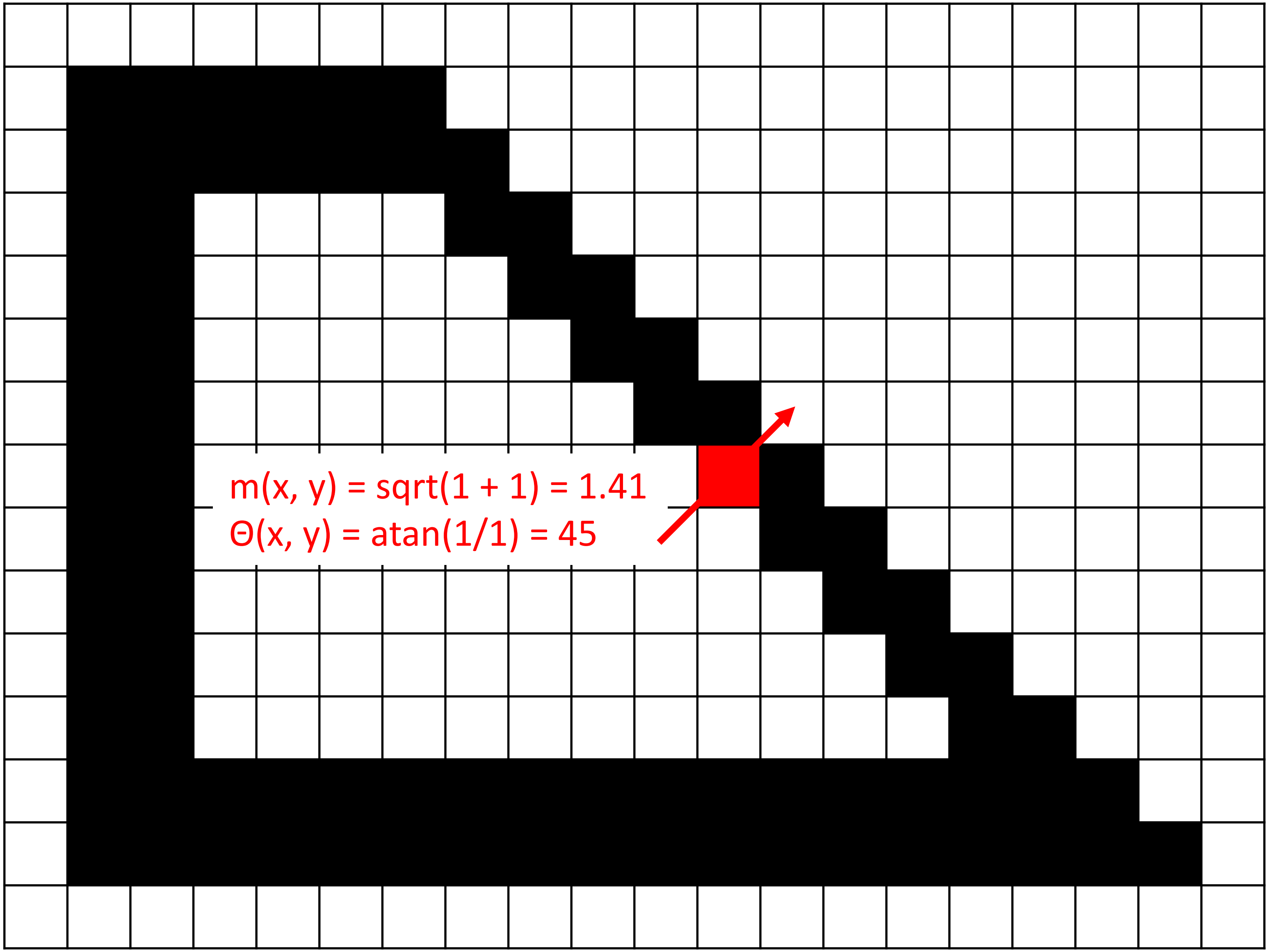
$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$



Gradients

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

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



Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature
- Compute gradient orientation for each pixel
- Create histogram over edge orientations weighted by magnitude
- That’s your feature descriptor!

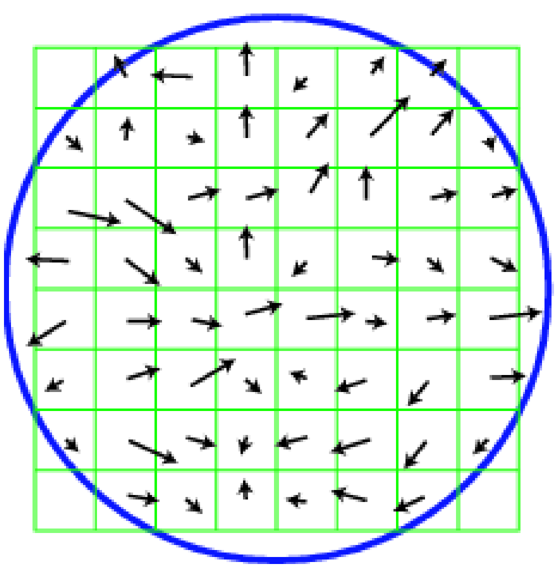
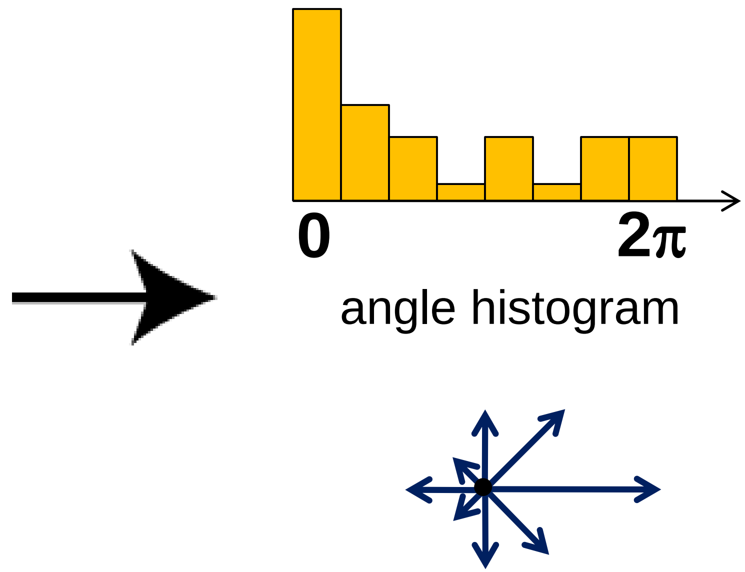


Image gradients

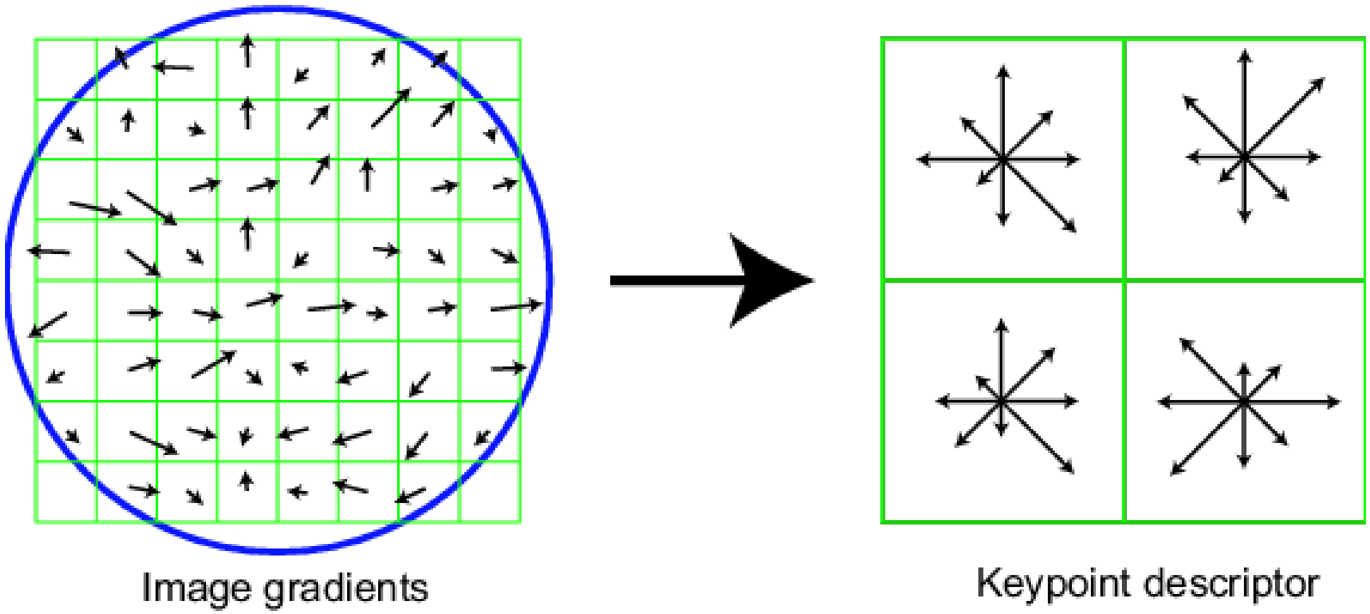


Adapted from L. Zitnick, D. Lowe

Scale Invariant Feature Transform

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Quantize the gradient orientations i.e. snap each gradient to one of 8 angles
- Each gradient contributes not just 1, but magnitude(gradient) to the histogram, i.e. stronger gradients contribute more
- 16 cells * 8 orientations = 128 dimensional descriptor for each detected feature



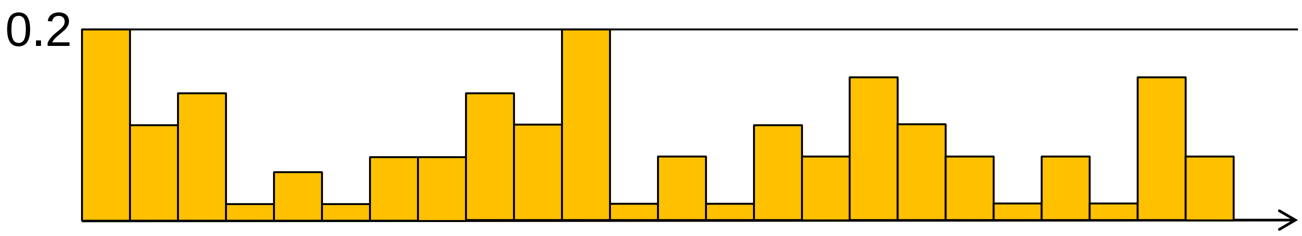
Adapted from L. Zitnick, D. Lowe

Scale Invariant Feature Transform

Full version

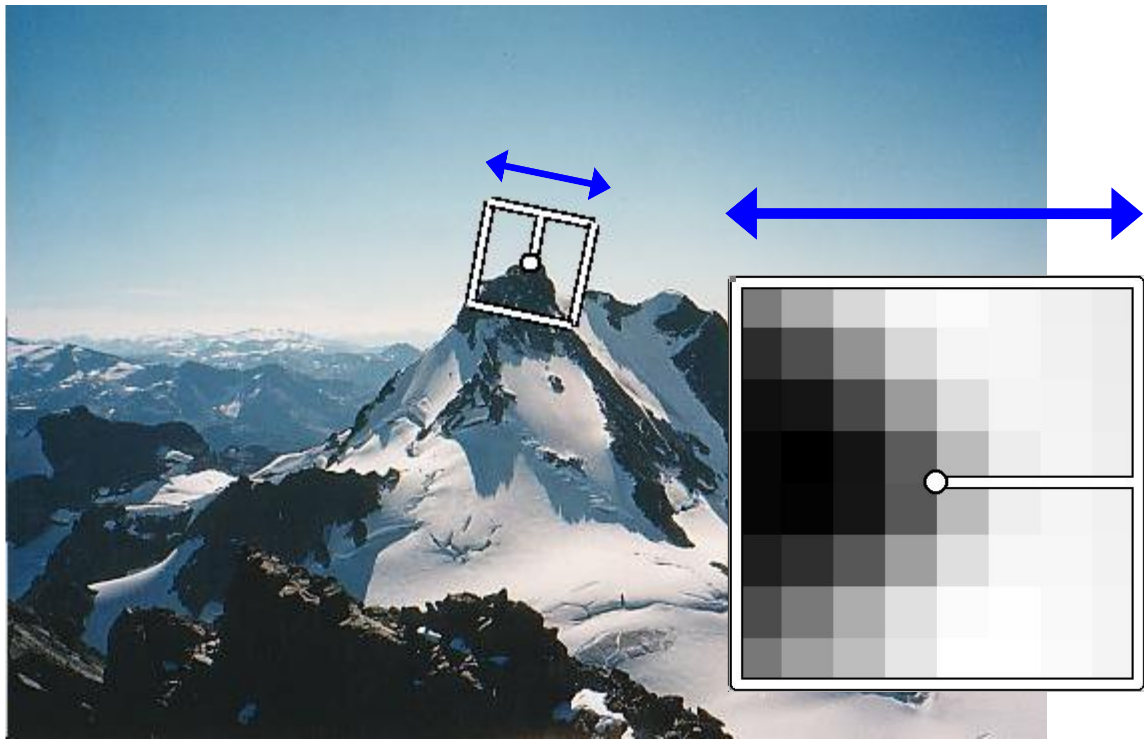
- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Quantize the gradient orientations i.e. snap each gradient to one of 8 angles
- Each gradient contributes not just 1, but magnitude(gradient) to the histogram, i.e. stronger gradients contribute more
- 16 cells * 8 orientations = 128 dimensional descriptor for each detected feature
- Normalize + clip (threshold normalize to 0.2) + normalize the descriptor
- After normalizing, we have:

$$\sum_i d_i = 1 \quad \text{such that: } d_i < 0.2$$



Adapted from L. Zitnick, D. Lowe

Making descriptor rotation invariant



- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation

K. Grauman

Image from Matthew Brown

SIFT is robust

- 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

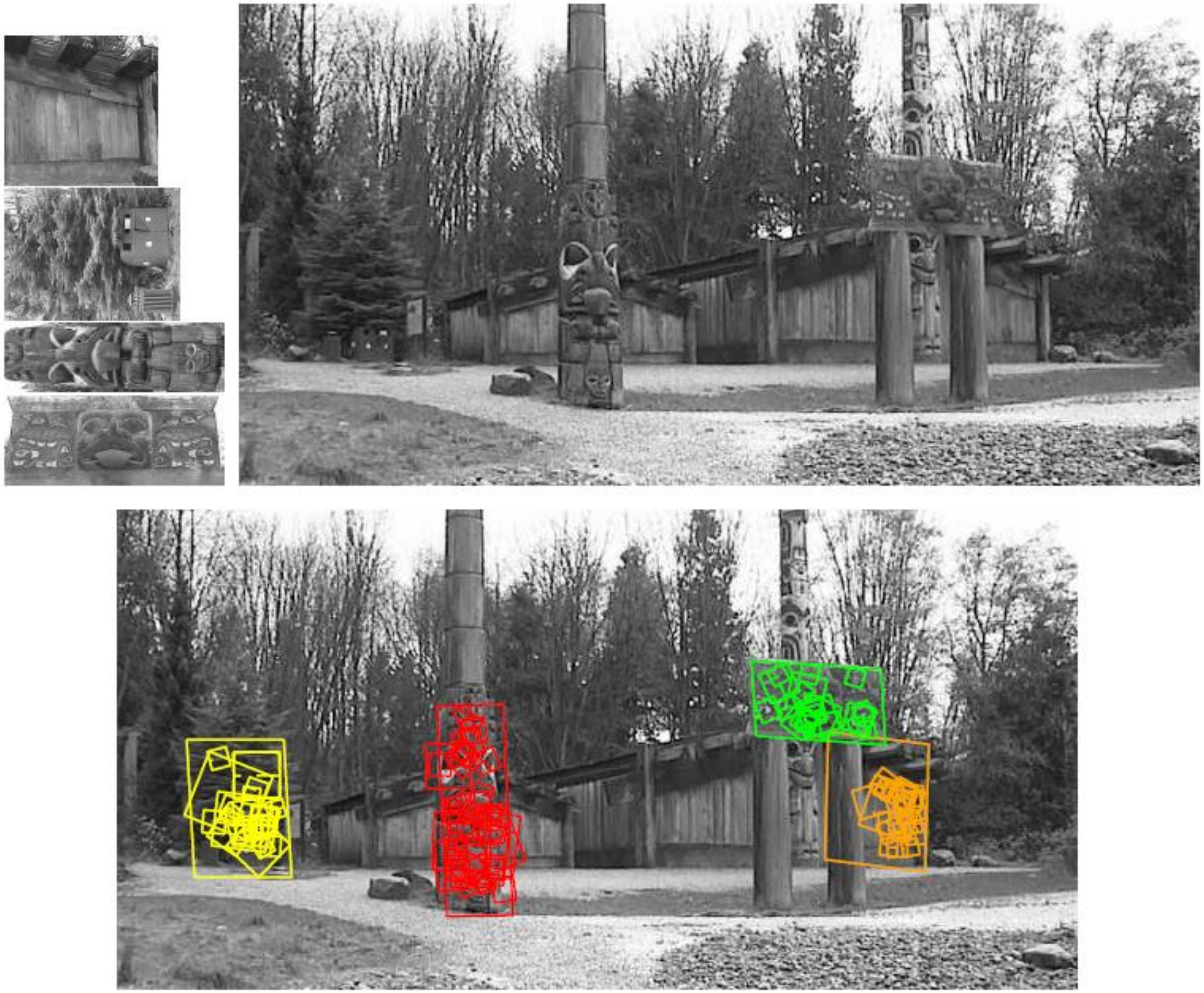
- Can be made to work without feature detection, resulting in "dense SIFT" (more points means robustness to occlusion)

- One commonly used implementation
  - http://www.vlfeat.org/overview/sift.html

Adapted from S. Seitz

Examples of using SIFT

Examples of using SIFT

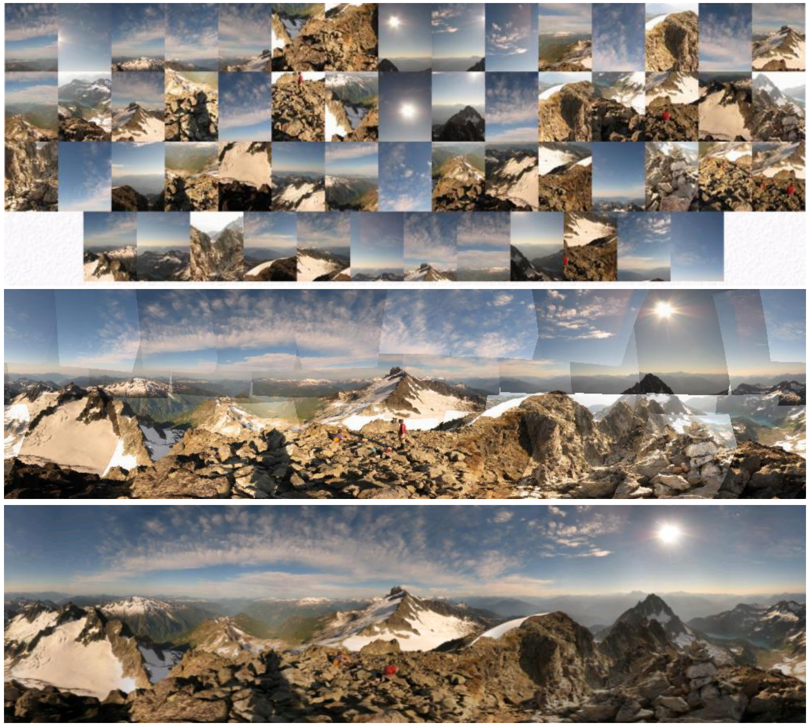


Examples of using SIFT

Images from S. Seitz

Applications of local invariant features

- Object recognition
- Indexing and retrieval
- Robot navigation
- 3D reconstruction
- Motion tracking
- Image alignment
- Panoramas and mosaics
- …



http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Adapted from K. Grauman and L. Lazebnik

Additional references

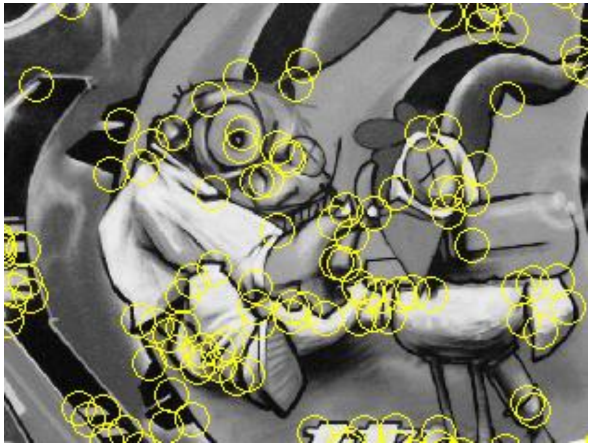
- Survey paper on local features
  - "Local Invariant Feature Detectors: A Survey" by Tinne Tuytelaars and Krystian Mikolajczyk, in *Foundations and Trends in Computer Graphics and Vision* Vol. 3, No. 3 (2007) 177–280 (mostly Chapters 1, 3.2, 7) http://homes.esat.kuleuven.be/%7Etuytelaa/FT_survey_interestpoints08.pdf

- Making Harris detection scale-invariant
  - "Indexing based on scale invariant interest points" by Krystian Mikolajczyk and Cordelia Schmid, in ICCV 2001 https://hal.archives-ouvertes.fr/file/index/docid/548276/filename/mikolajcICCV2001.pdf

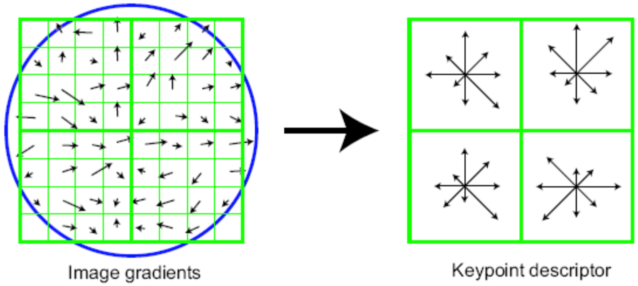
- SIFT paper by David Lowe
  - "Distinctive Image Features from Scale-Invariant Keypoints" by David G. Lowe, in IJCV 2004 http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

Summary

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Laplacian of Gaussian, automatic scale selection



- Descriptors: robust and selective
  - Histograms for robustness to small shifts and translations (SIFT descriptor)



Adapted from D. Hoiem, K. Grauman