Corners, Blobs & Descriptors

Lecture 3 – Prof. Rob Fergus

With slides from S. Lazebnik & S. Seitz, D. Lowe, A. Efros

Motivation: Build a Panorama

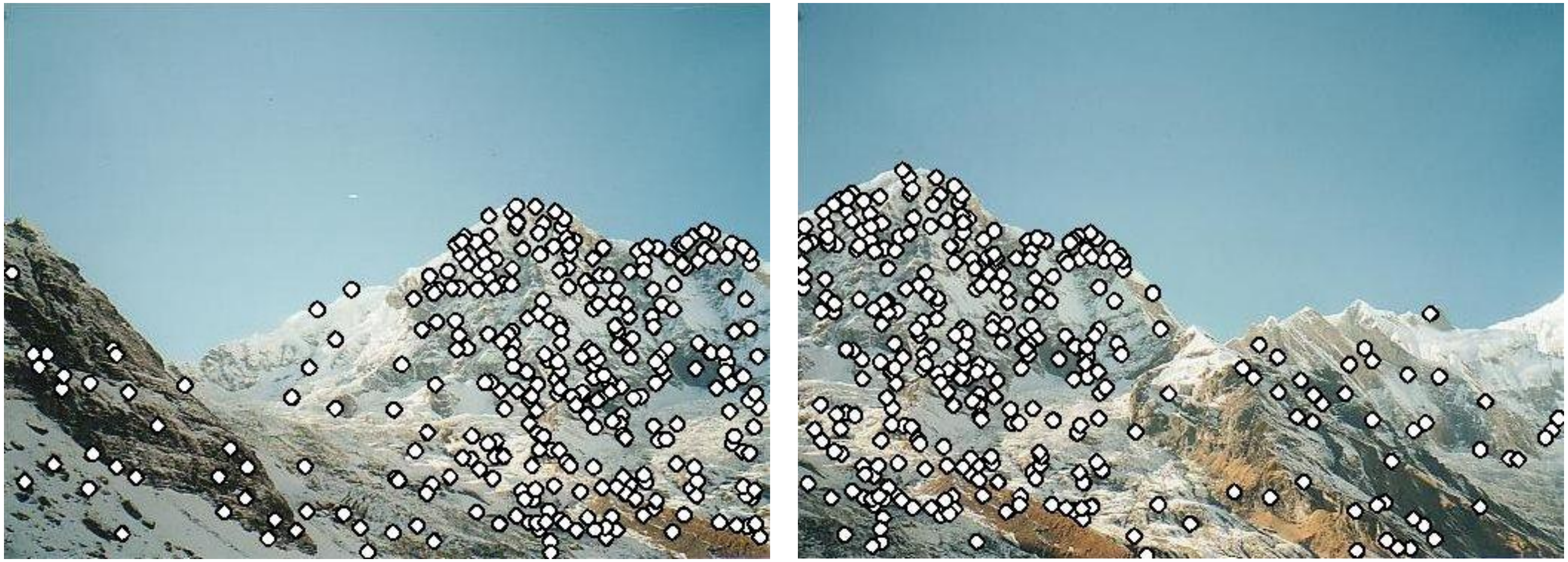
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

How do we build panorama?

- We need to match (align) images

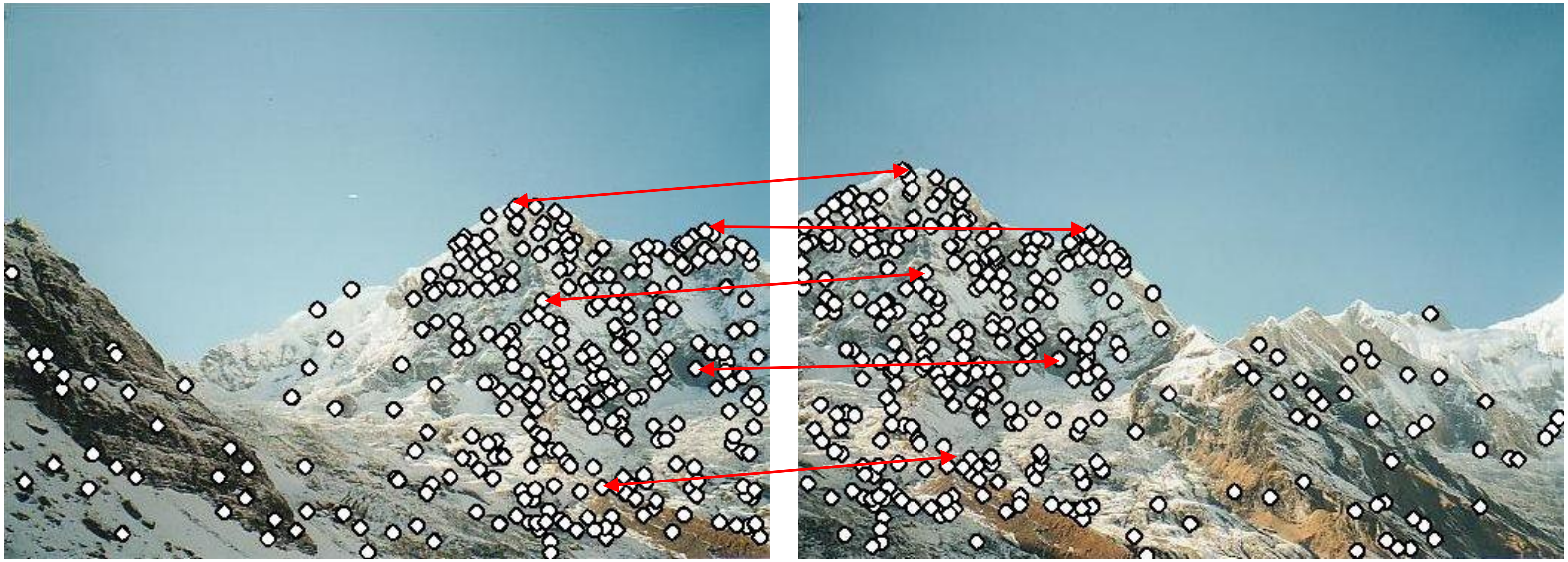
Matching with Features

- Detect feature points in both images



Matching with Features

- Detect feature points in both images
- Find corresponding pairs

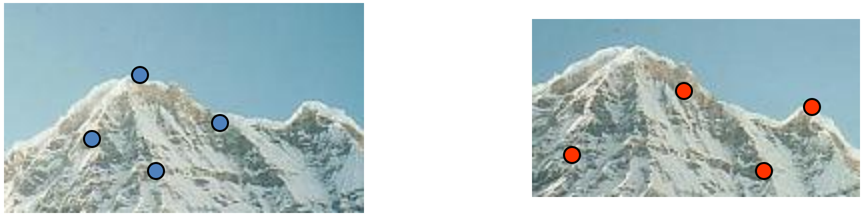


Matching with Features

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

Matching with Features

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



no chance to match!

We need a repeatable detector

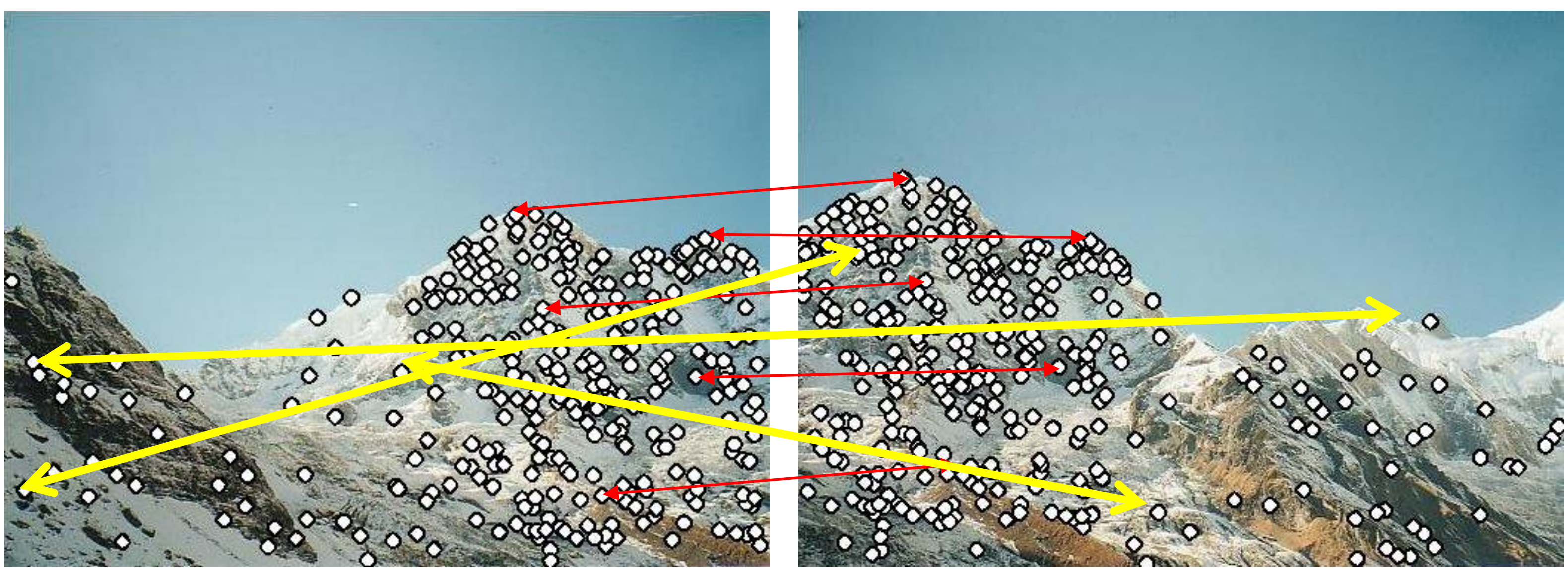
Matching with Features

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

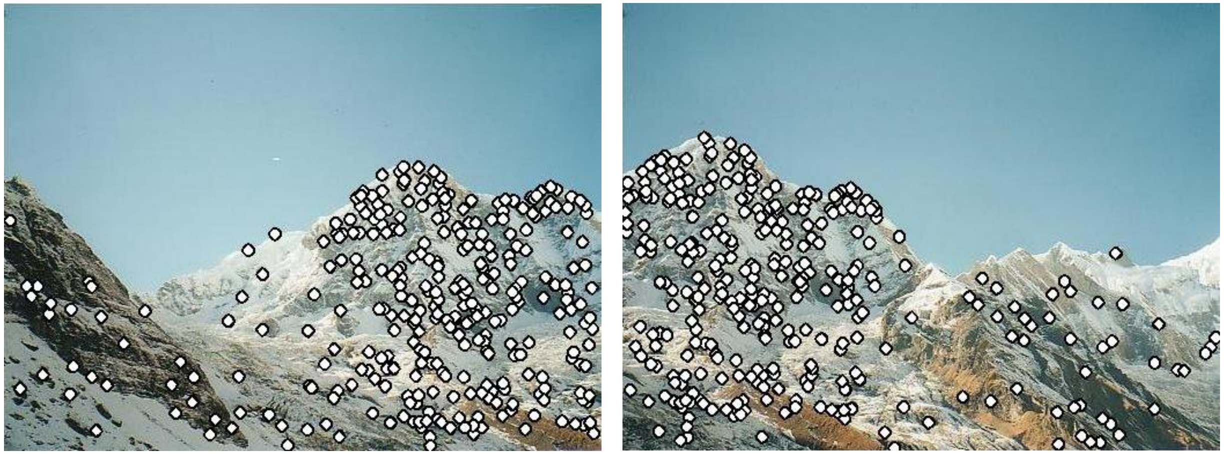
We need a reliable and distinctive descriptor

Matching with Features

- Problem 3:
  - Need to estimate transformation between images, despite erroneous correspondences.



Characteristics of good features



- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Applications

Feature points are used for:

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

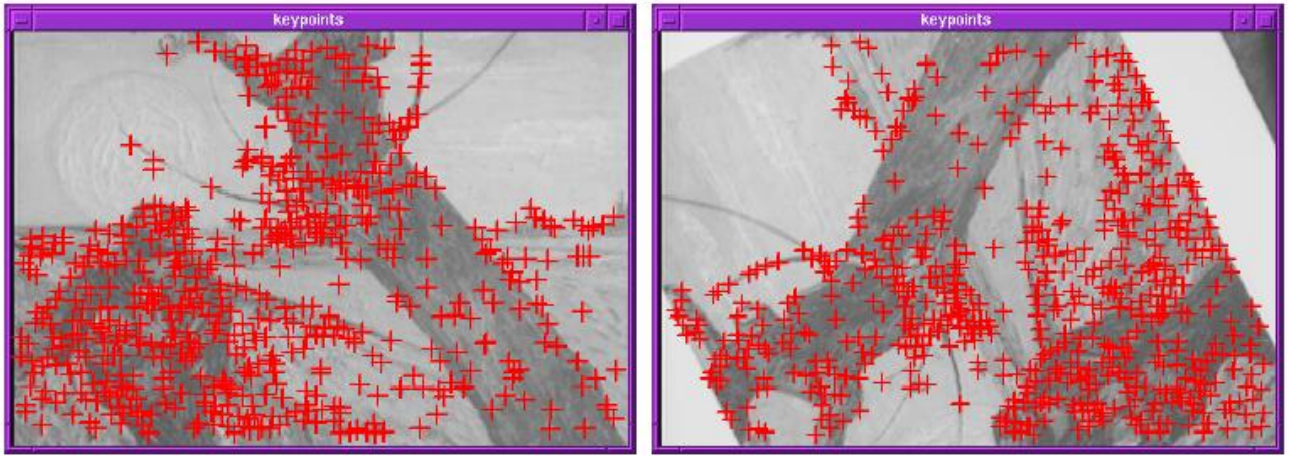
Overview

- Corners (Harris Detector)
- Blobs
- Descriptors

Overview

- Corners (Harris Detector)
- Blobs
- Descriptors

Finding Corners

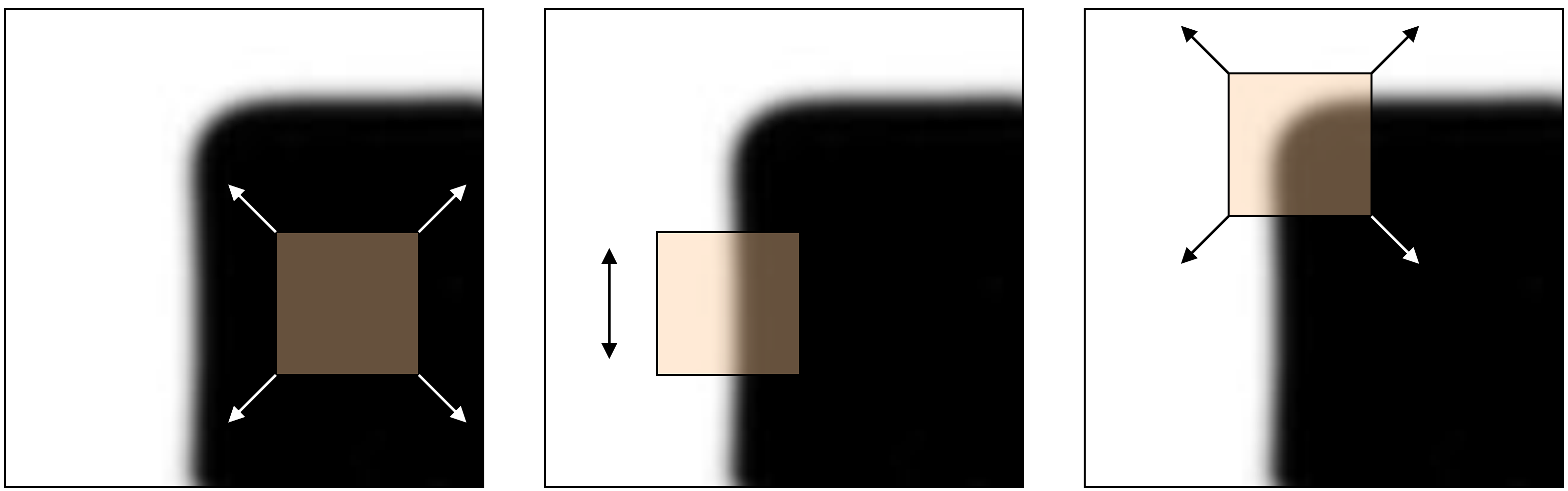


- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

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

Corner Detection: Basic Idea

- 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

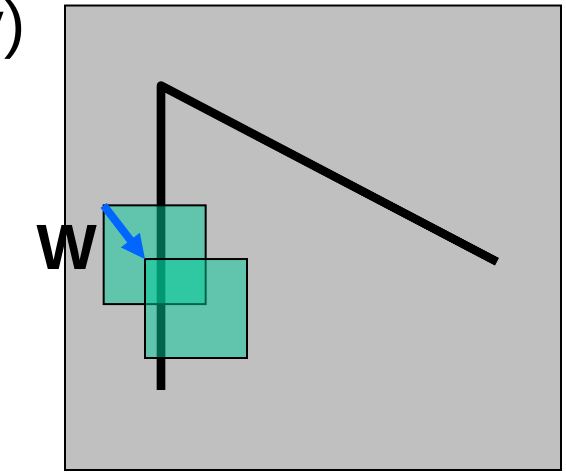
"corner": significant change in all directions

Source: A. Efros

Feature detection: the math

Consider shifting the window **W** by (u,v)

- how do the pixels in **W** change?
- compare each pixel before and after by summing up the squared differences (SSD)
- this defines an SSD "error" of *E(u,v)*:



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

Source: S. Seitz

Small motion assumption

Taylor Series expansion of I:

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

If the motion (u,v) is small, then first order approx is good

$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$$\approx I(x,y) + [I_x \;\; I_y] \begin{bmatrix} u \\ v \end{bmatrix}$$

$$\text{shorthand: } I_x = \frac{\partial I}{\partial x}$$

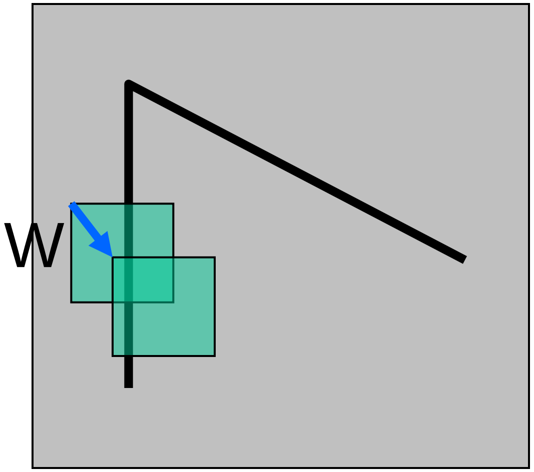
Plugging this into the formula on the previous slide…

Source: S. Seitz

Feature detection: the math

Consider shifting the window W by (u,v)

- how do the pixels in W change?
- compare each pixel before and after by summing up the squared differences
- this defines an "error" of E(u,v):



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

$$\approx \sum_{(x,y)\in W} \left[I(x,y) + [I_x \ I_y]\begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)\right]^2$$

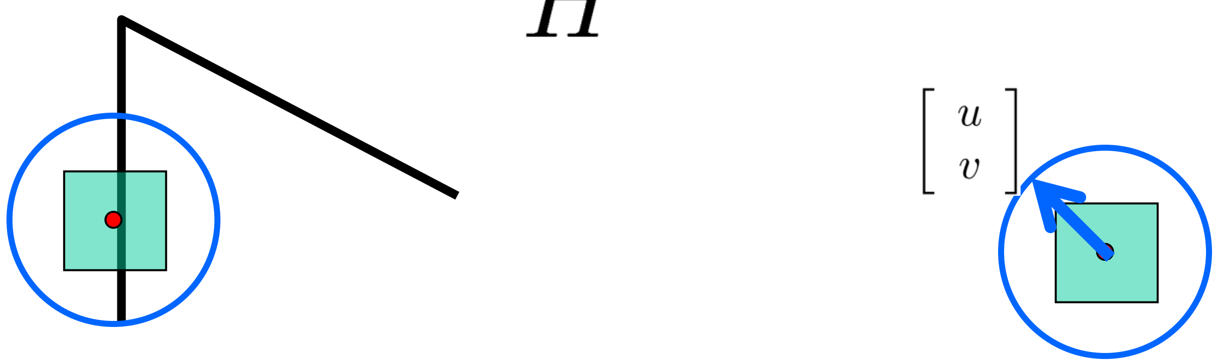
$$\approx \sum_{(x,y)\in W} \left[[I_x \ I_y]\begin{bmatrix} u \\ v \end{bmatrix}\right]^2$$

Source: S. Seitz

Feature detection: the math

This can be rewritten:

$$E(u,v) = \sum_{(x,y)\in W} [u \ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$



For the example above

- You can move the center of the green window to anywhere on the blue unit circle
- Which directions will result in the largest and smallest E values?
- We can find these directions by looking at the eigenvectors of ***H***

Source: S. Seitz

Quick eigenvalue/eigenvector review

The **eigenvectors** of a matrix **A** are the vectors **x** that satisfy:

$$Ax = \lambda x$$

The scalar λ is the **eigenvalue** corresponding to **x**

- The eigenvalues are found by solving:

$$det(A - \lambda I) = 0$$

- In our case, ***A*** = ***H*** is a 2x2 matrix, so we have

$$det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

- The solution:

$$\lambda_{\pm} = \frac{1}{2}\left[(h_{11} + h_{22}) \pm \sqrt{4h_{12}h_{21} + (h_{11} - h_{22})^2}\right]$$

Once you know λ, you find **x** by solving

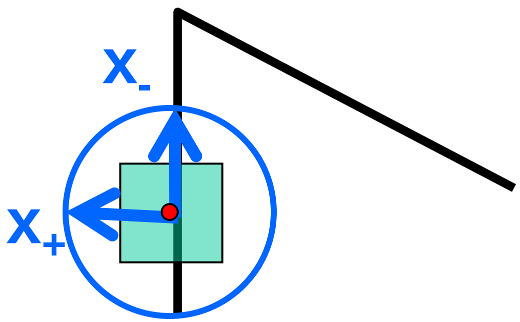
$$\begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

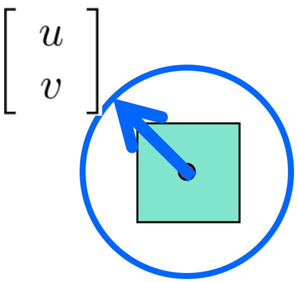
Source: S. Seitz

Feature detection: the math

This can be rewritten:

$$E(u,v) = \sum_{(x,y)\in W} [u \ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$





Eigenvalues and eigenvectors of H

- Define shifts with the smallest and largest change (E value)
- x_+ = direction of **largest** increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of **smallest** increase in E.
- λ- = amount of increase in direction x_+

$$\begin{aligned} Hx_+ &= \lambda_+ x_+ \\ Hx_- &= \lambda_- x_- \end{aligned}$$

Source: S. Seitz

Feature detection: the math

How are λ_+, $\mathbf{x}_+$, λ_-, and $\mathbf{x}_+$ relevant for feature detection?

- What's our feature scoring function?

Source: S. Seitz

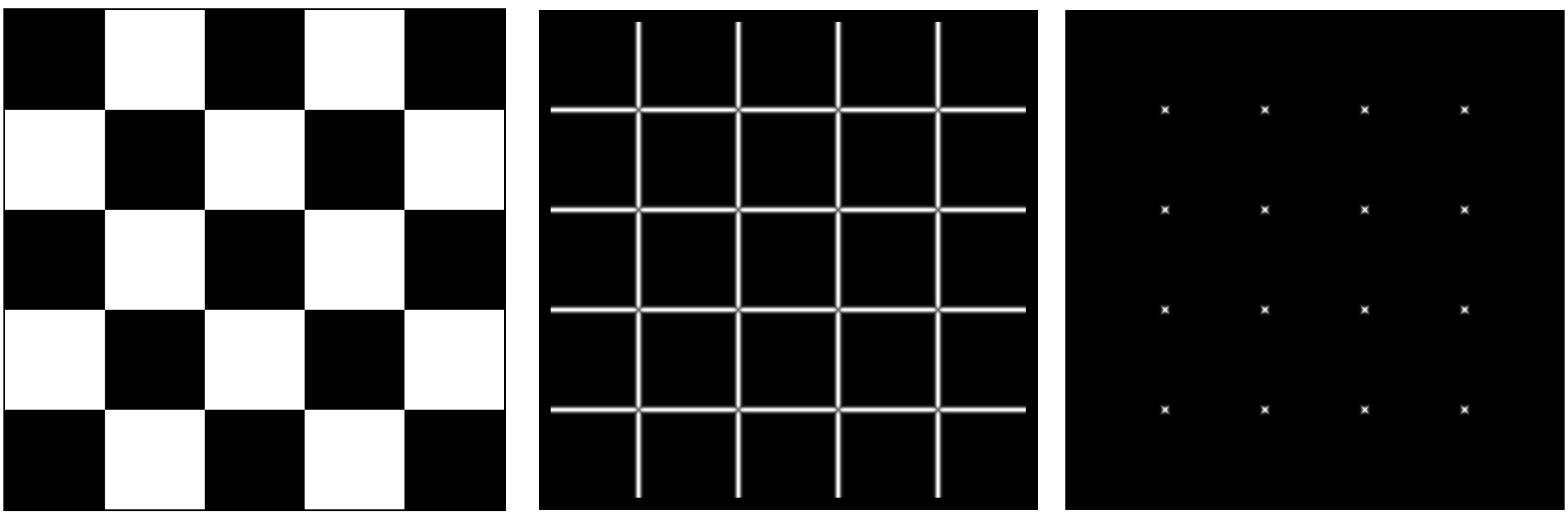
Feature detection: the math

How are λ_+, $\mathbf{x}_+$, λ_-, and $\mathbf{x}_+$ relevant for feature detection?

- What's our feature scoring function?

Want *E(u,v)* to be ***large*** for small shifts in ***all*** directions

- the *minimum* of *E(u,v)* should be large, over all unit vectors [u v]
- this minimum is given by the smaller eigenvalue (λ_-) of ***H***

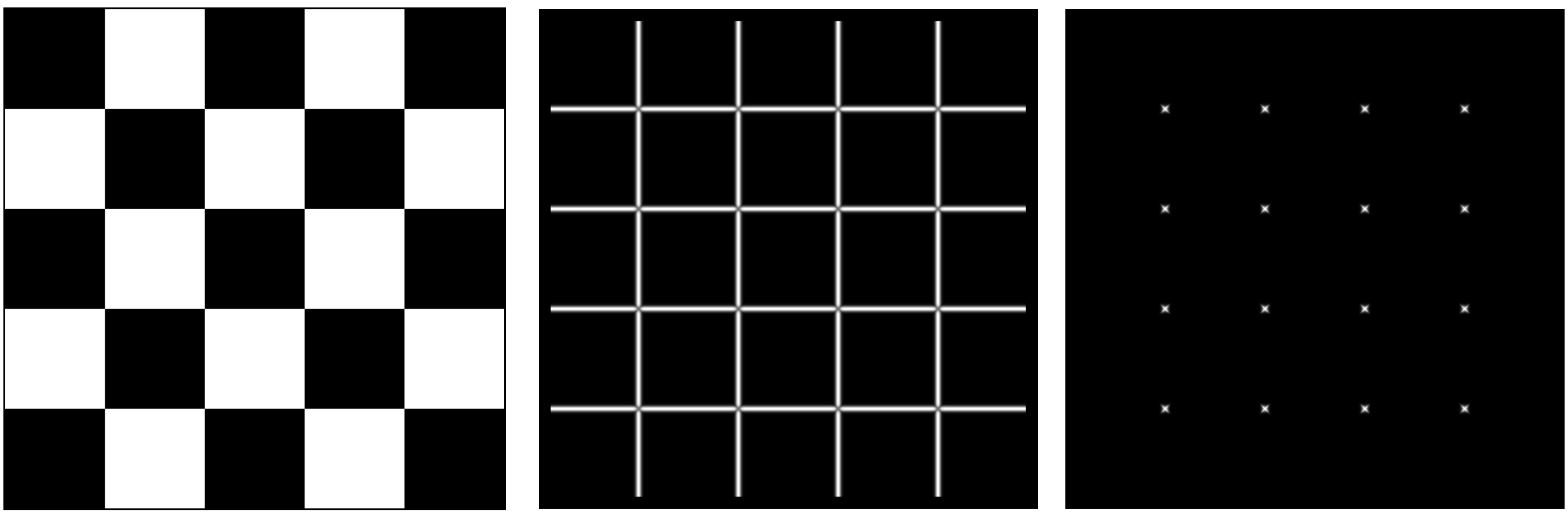


Source: S. Seitz I λ_+ λ_-

Feature detection summary

Here’s what you do

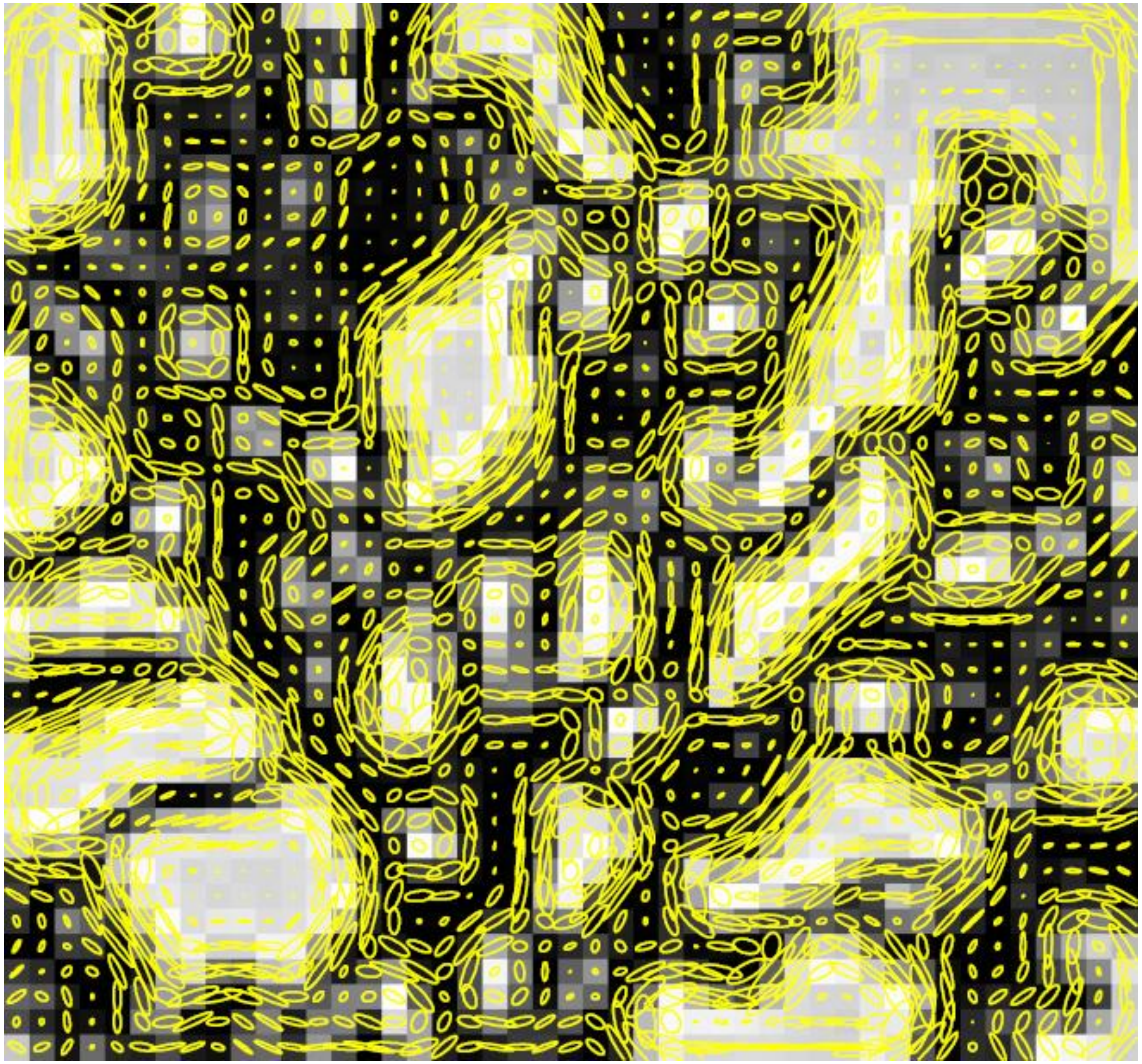
- Compute the gradient at each point in the image
- Create the ***H*** matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features



Source: S. Seitz I λ_+ λ_-

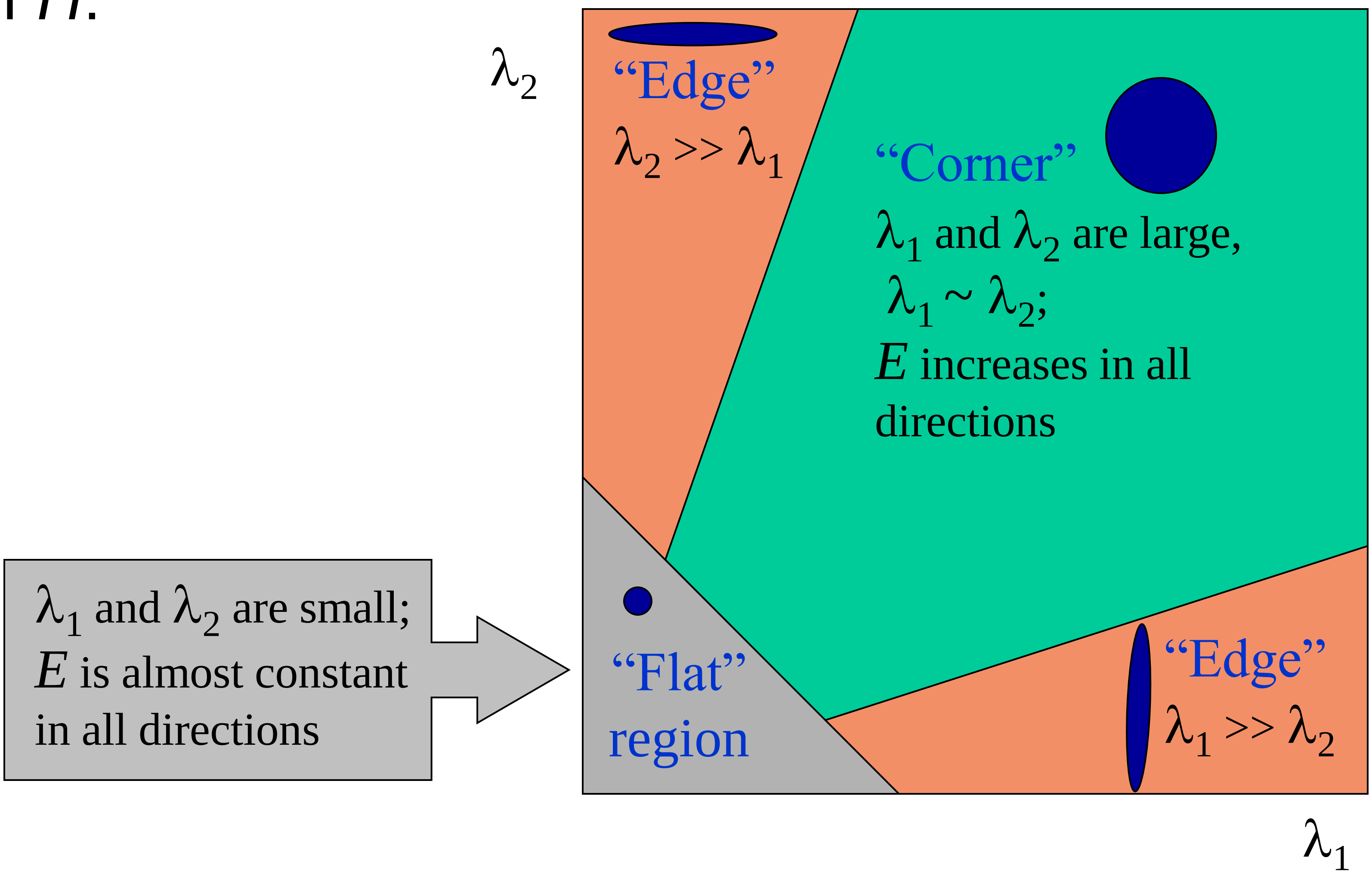
Visualization of second moment matrices

Visualization of second moment matrices



Interpreting the eigenvalues

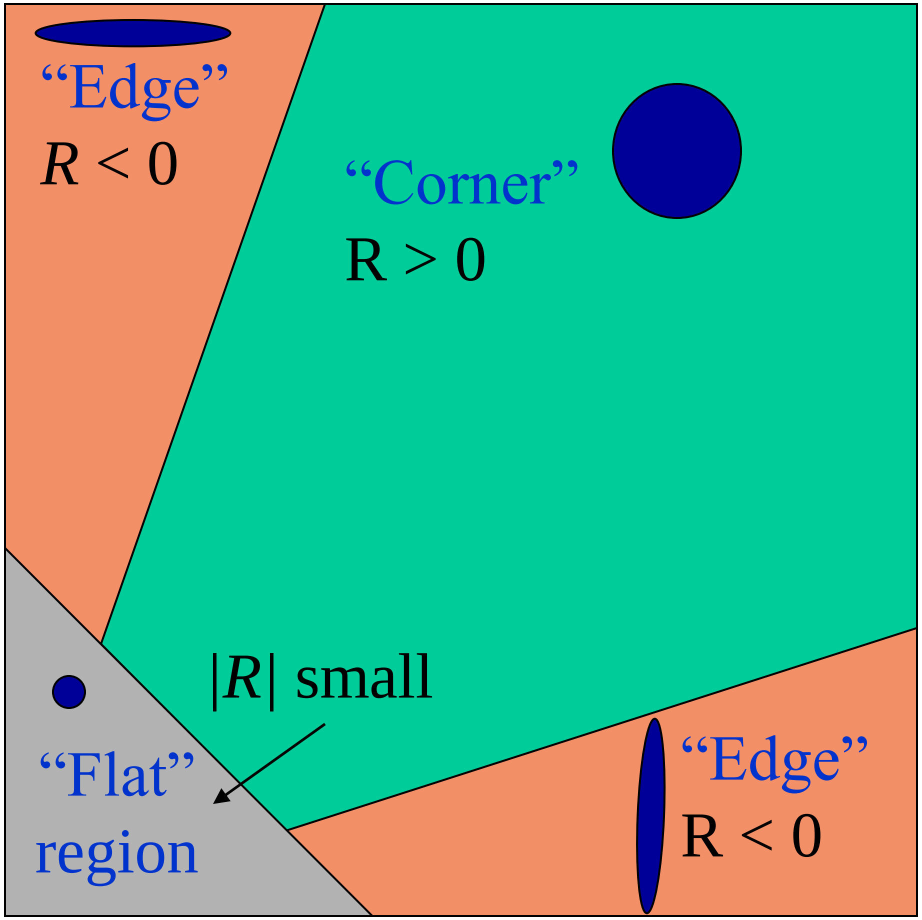
Classification of image points using eigenvalues of *H*:



Corner response function

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

α: constant (0.04 to 0.06)



Harris detector: Steps

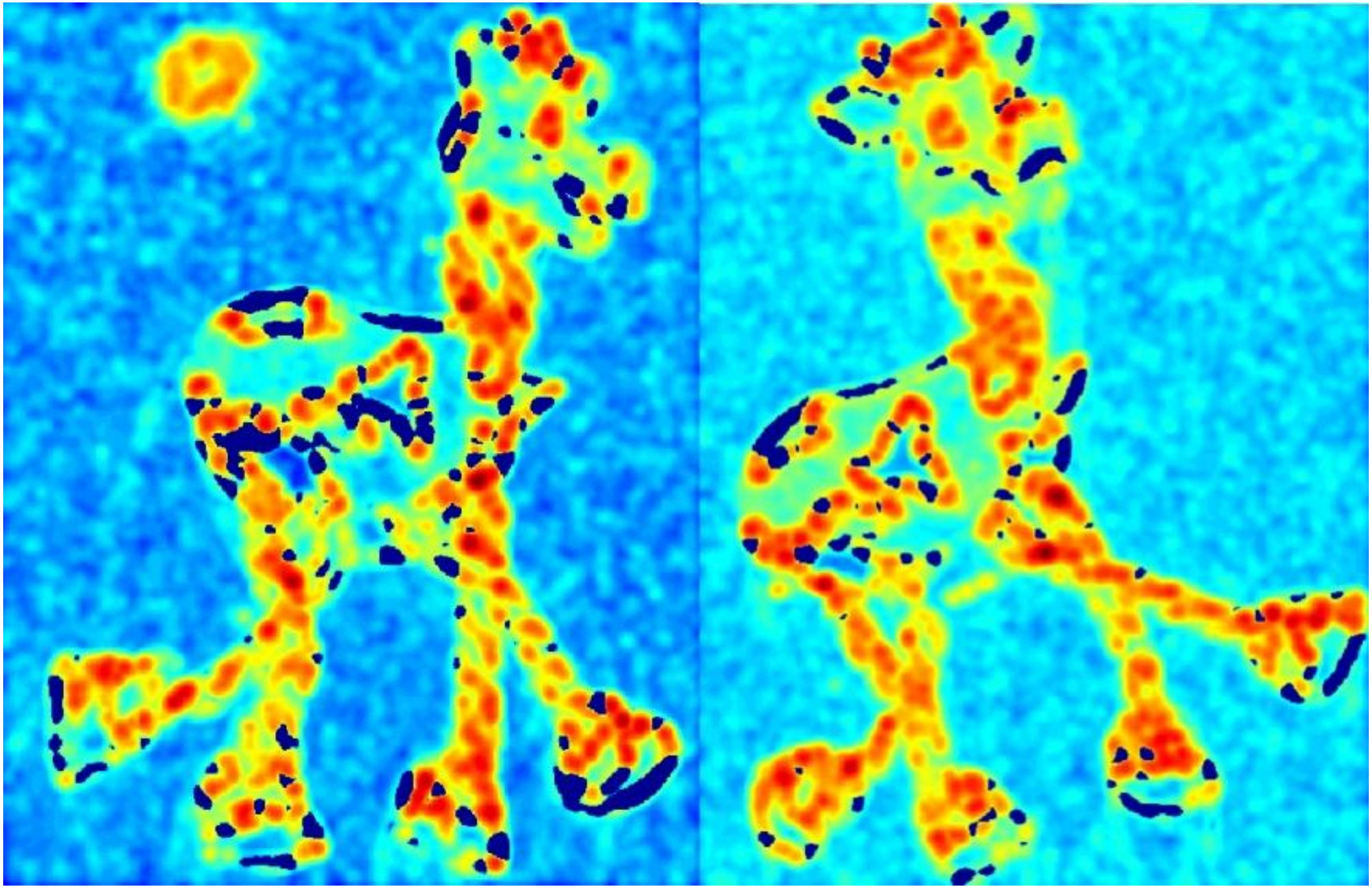
1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix *H* in a Gaussian window around each pixel
3. Compute corner response function *R*
4. Threshold *R*
5. Find local maxima of response function (nonmaximum suppression)

C.Harris and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

Harris Detector: Steps

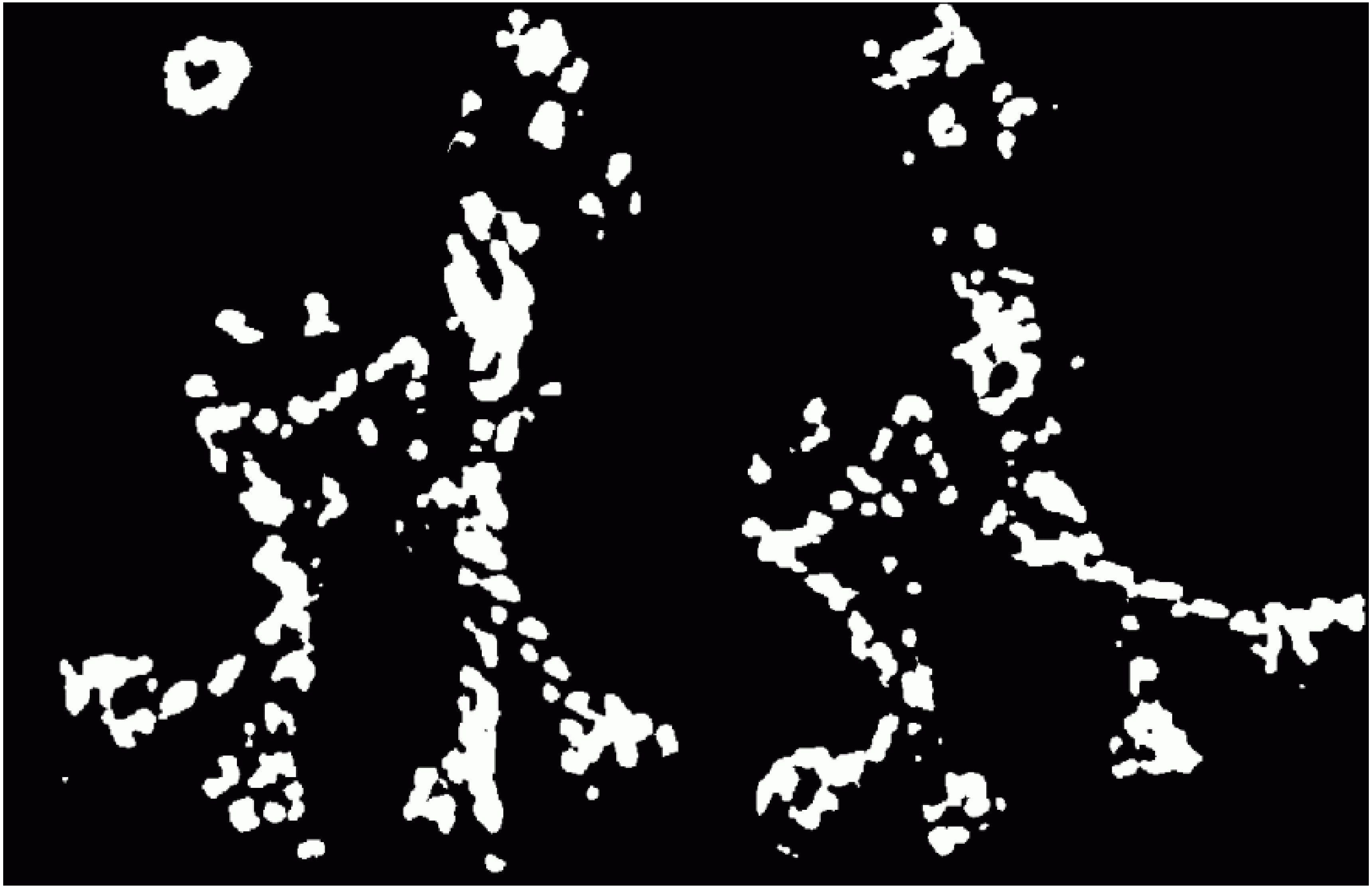
Harris Detector: Steps

Compute corner response R



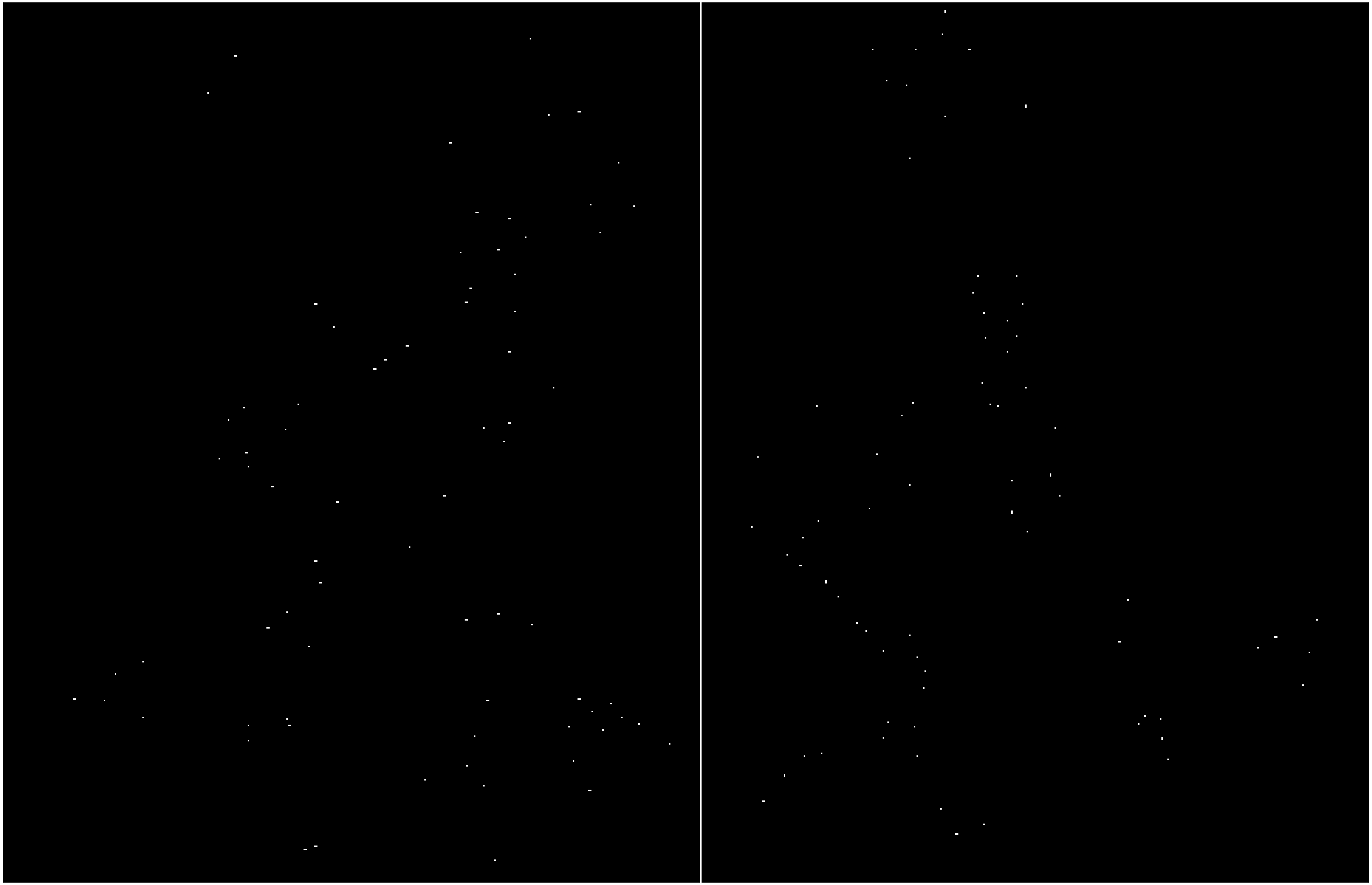
Harris Detector: Steps

Find points with large corner response: R>threshold



Harris Detector: Steps

Take only the points of local maxima of R

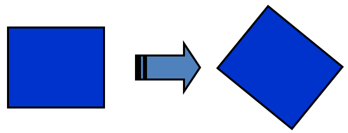


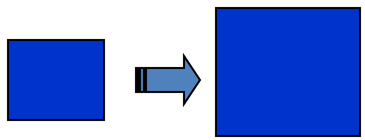
Harris Detector: Steps

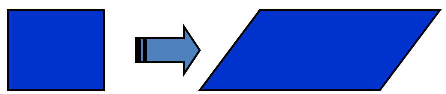
Invariance and covariance

- We want features to be *invariant* to photometric transformations and *covariant* to geometric transformations
  - **Invariance:** image is transformed and features do not change
  - **Covariance:** if we have two transformed versions of the same image, features should be detected in corresponding locations

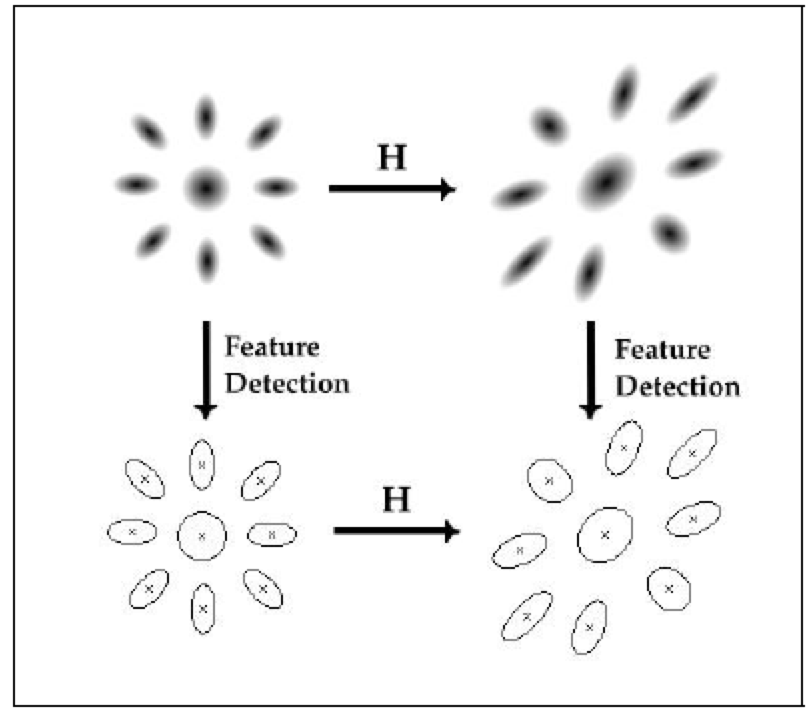
Transformations

- Geometric
  - **Rotation** 

    Scale 

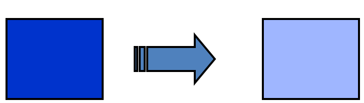
  - **Affine** 

    valid for:
    orthographic camera,
    locally planar object



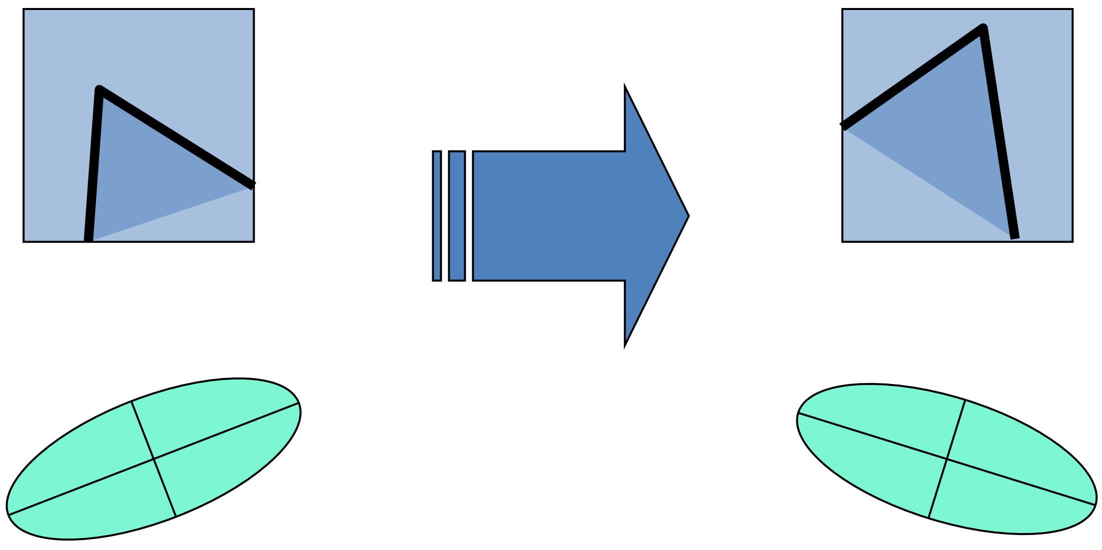
T. Kadir, A. Zisserman and M. Brady, An Affine invariant salient region detector, ECCV 2004

- Photometric
  - **Affine intensity change** $(I \rightarrow a\, I + b)$



Slide credit: S. Lazebnik

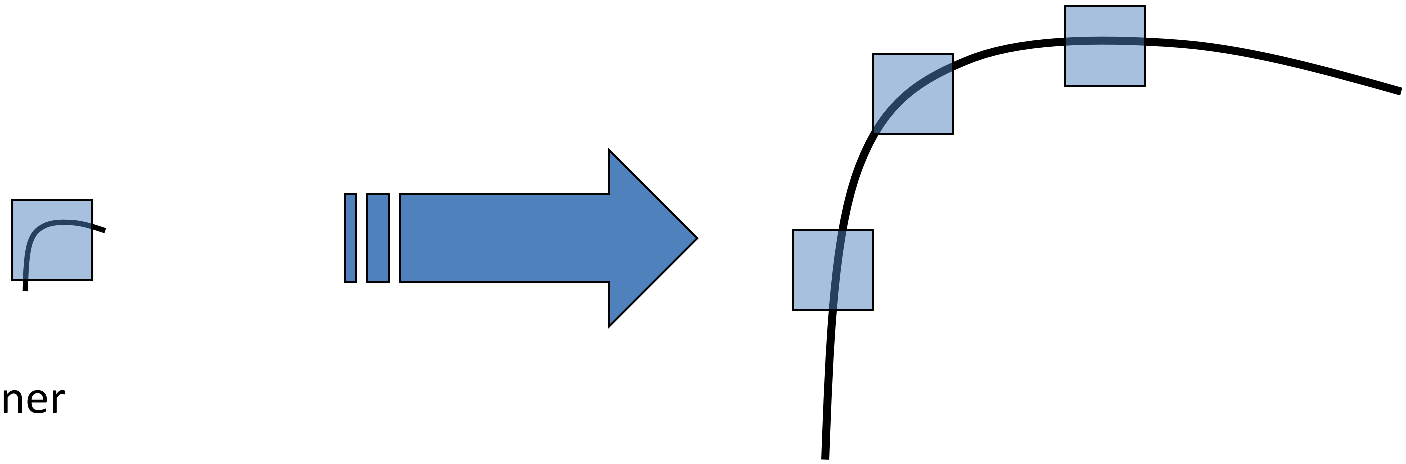
Image rotation



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

Corner response R is invariant w.r.t. rotation and *corner location* is covariant

Scaling



Corner

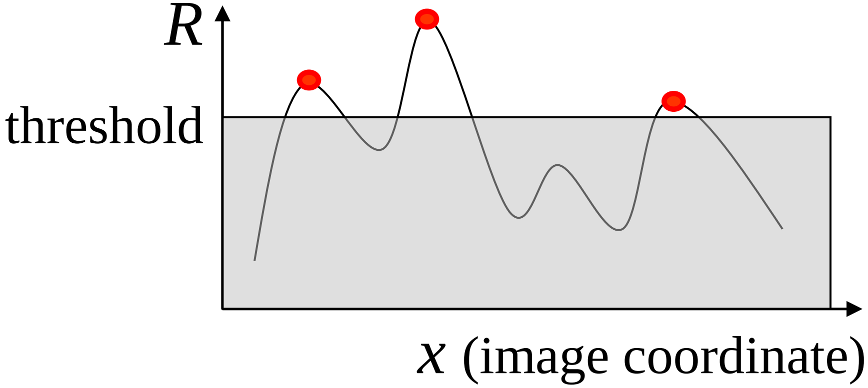
All points will be classified as edges

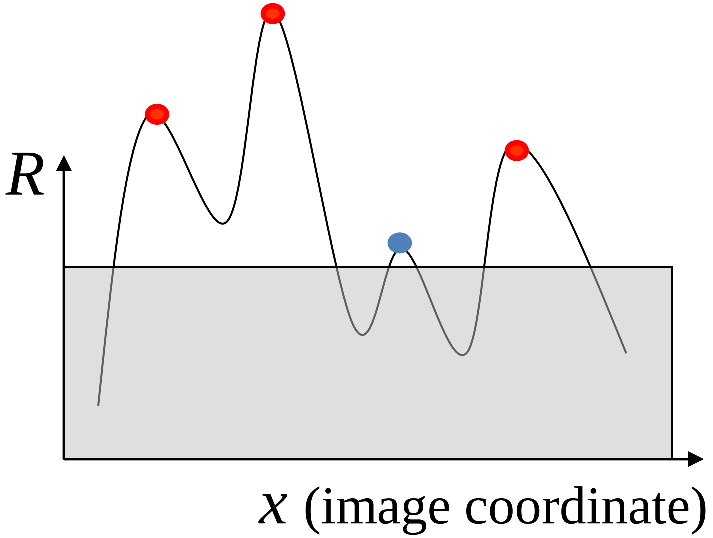
Not invariant to scaling

Affine intensity change

✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

✓ Intensity scale: $I \rightarrow a\,I$



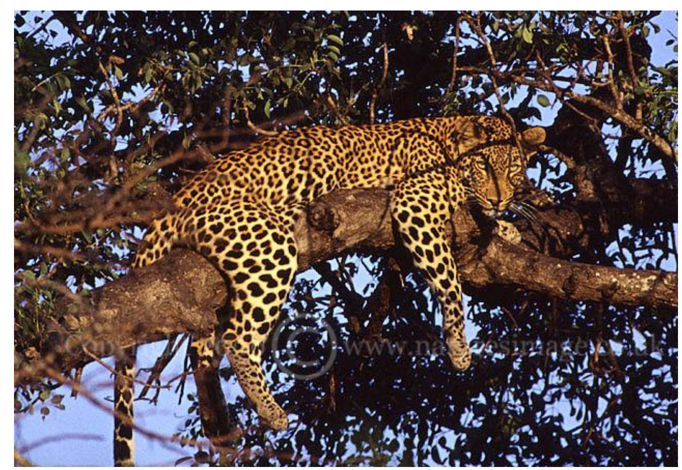


Partially invariant to affine intensity change

What about internal structure?

- Edges & Corners convey boundary information

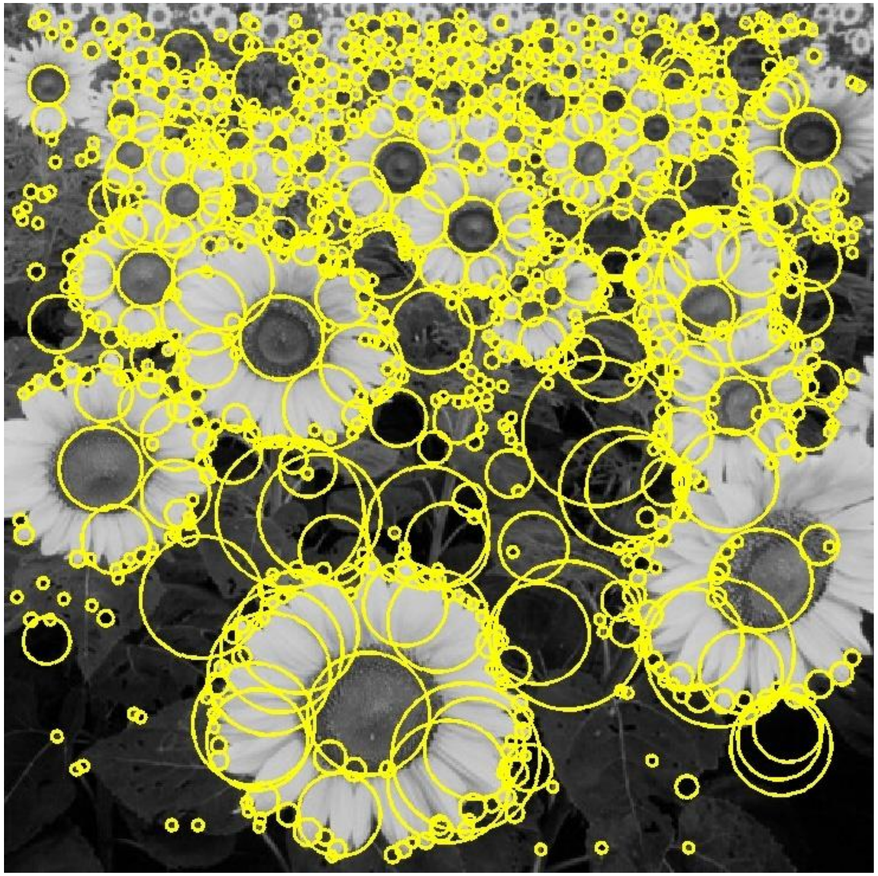
- What about interior texture of the object?



Overview

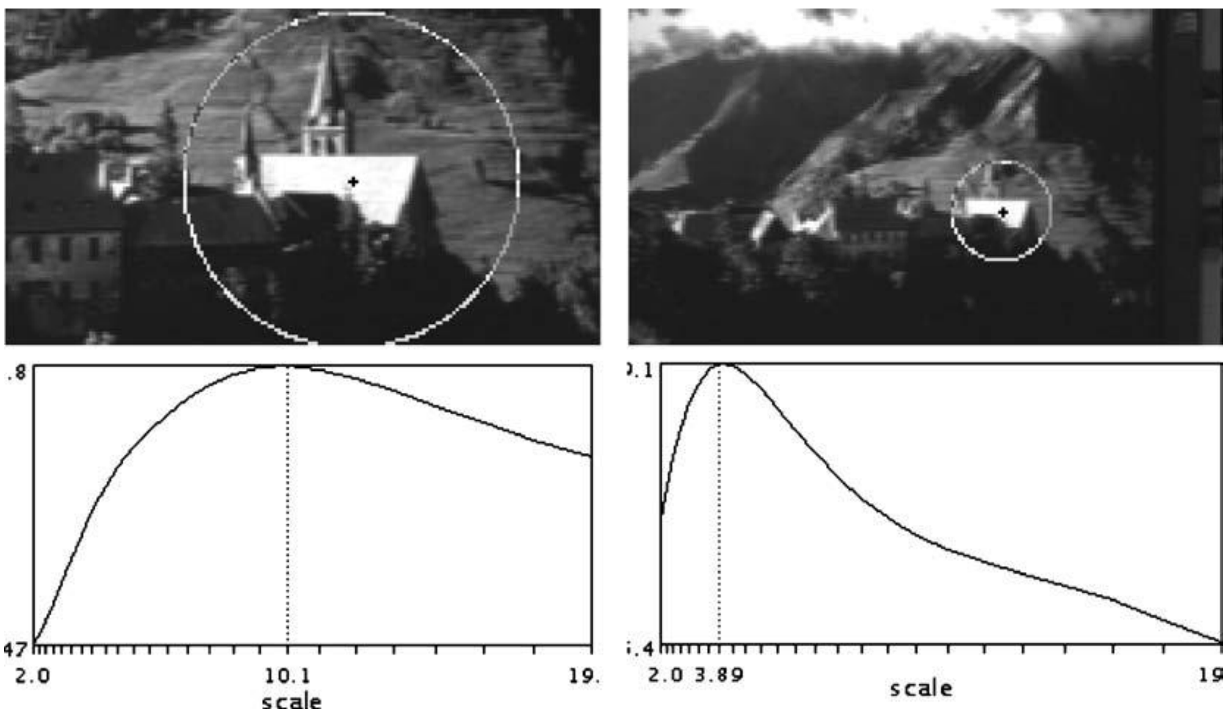
- Corners (Harris Detector)
- Blobs
- Descriptors

Blob detection with scale selection



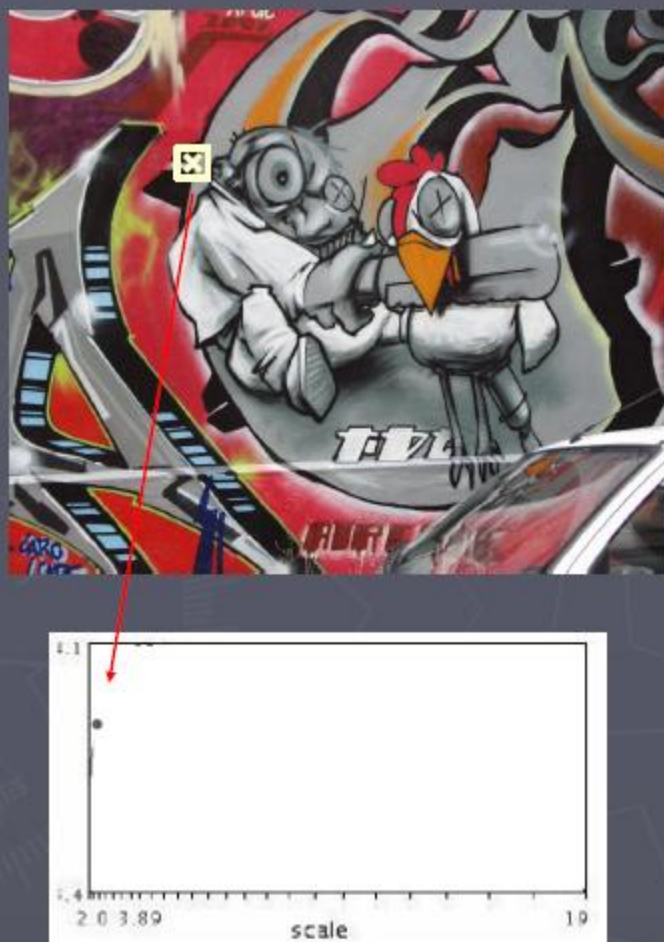
Achieving scale covariance

- Goal: independently detect corresponding regions in scaled versions of the same image
- Need *scale selection* mechanism for finding characteristic region size that is *covariant* with the image transformation



Automatic scale selection

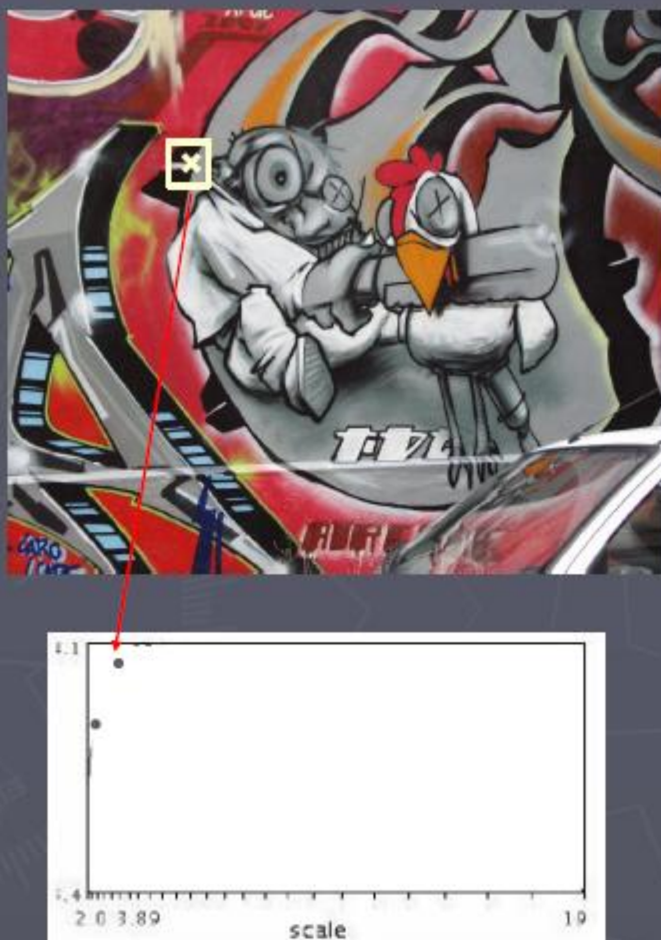
Lindeberg et al., 1996



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

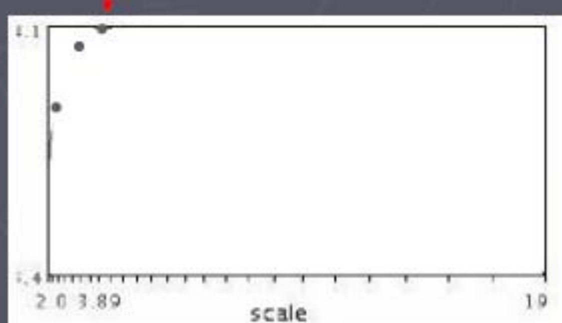
Slide from Tinne Tuytelaars

Automatic scale selection



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

Automatic scale selection

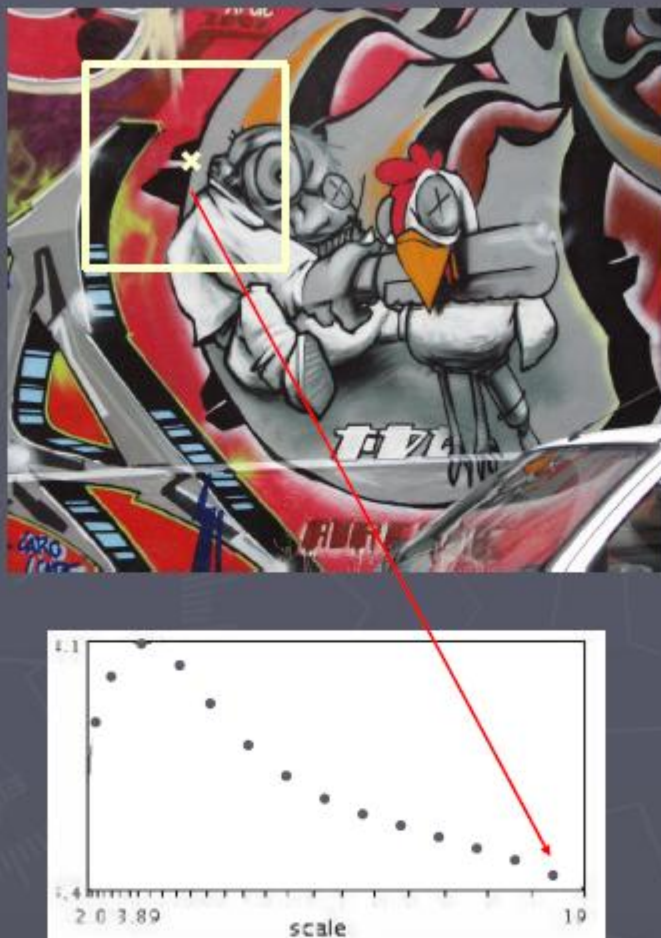


$f(I_{i_1 \ldots i_m}(x, \sigma))$

Automatic scale selection

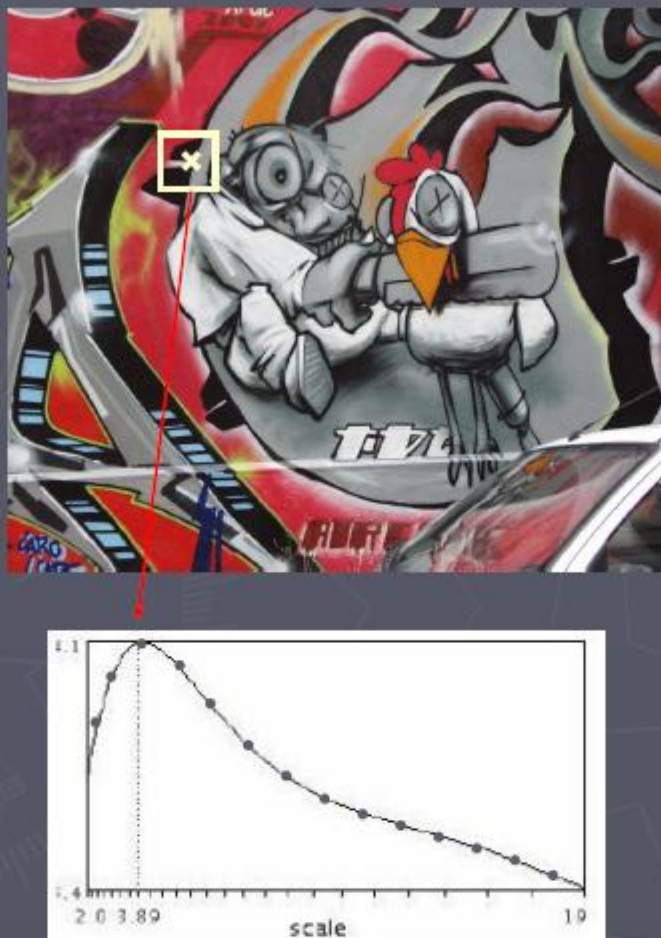
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



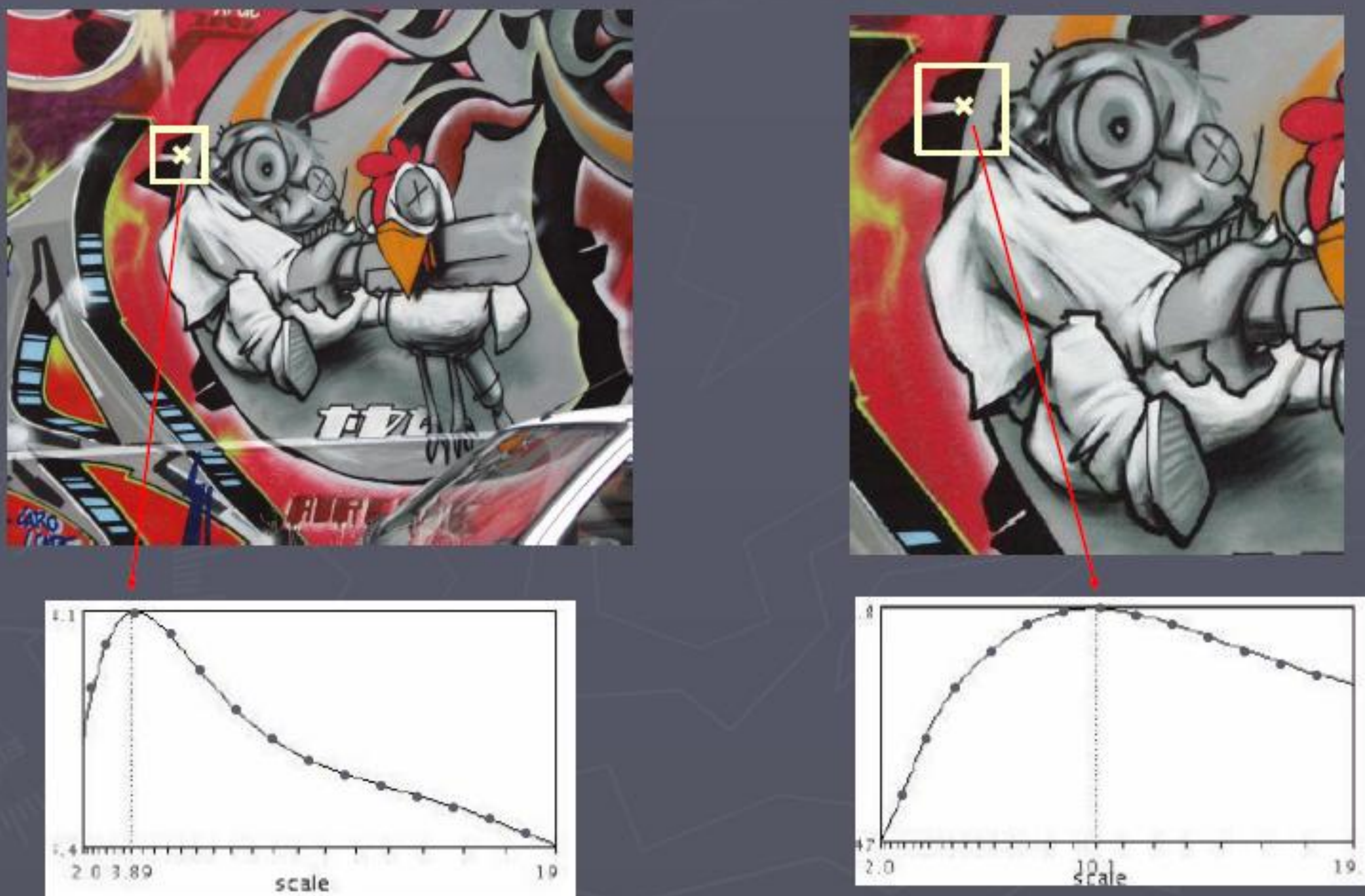
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



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

Automatic scale selection



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

Recall: Edge detection

f

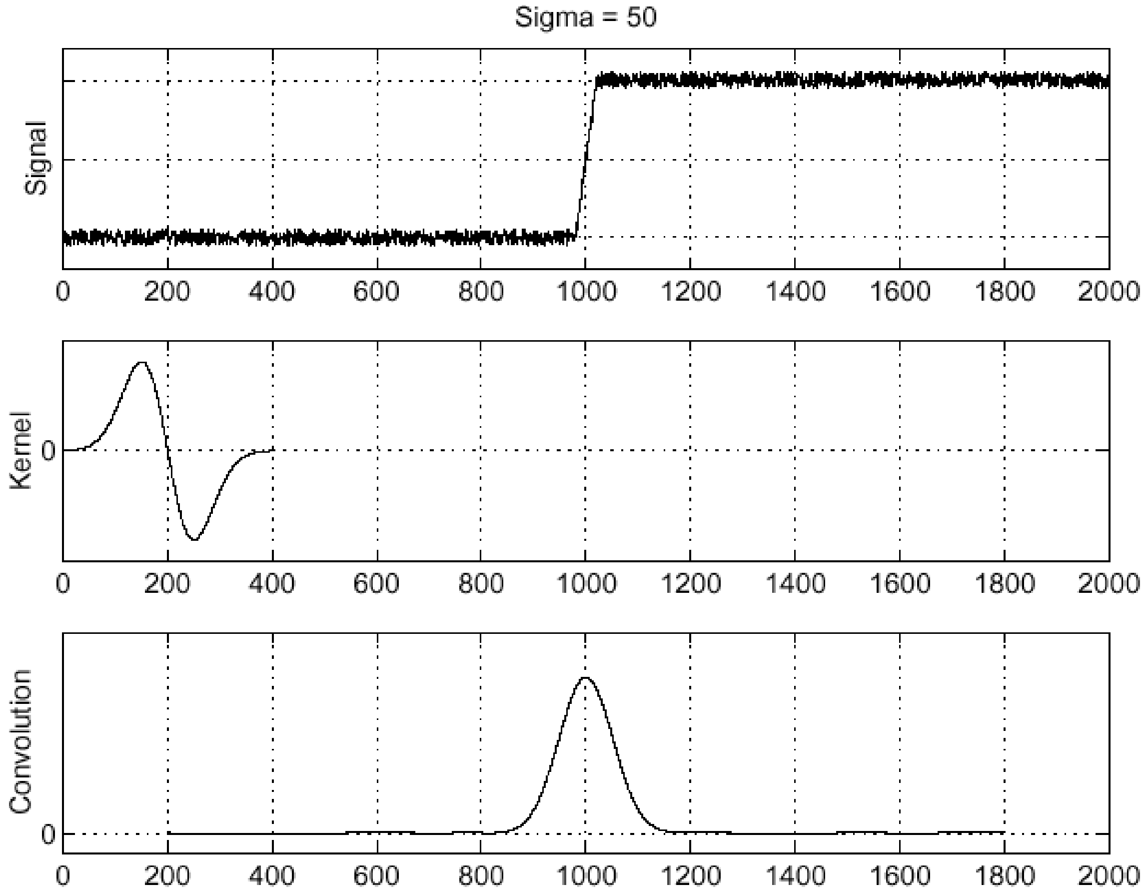
Edge

$\frac{d}{dx} g$

Derivative
of Gaussian

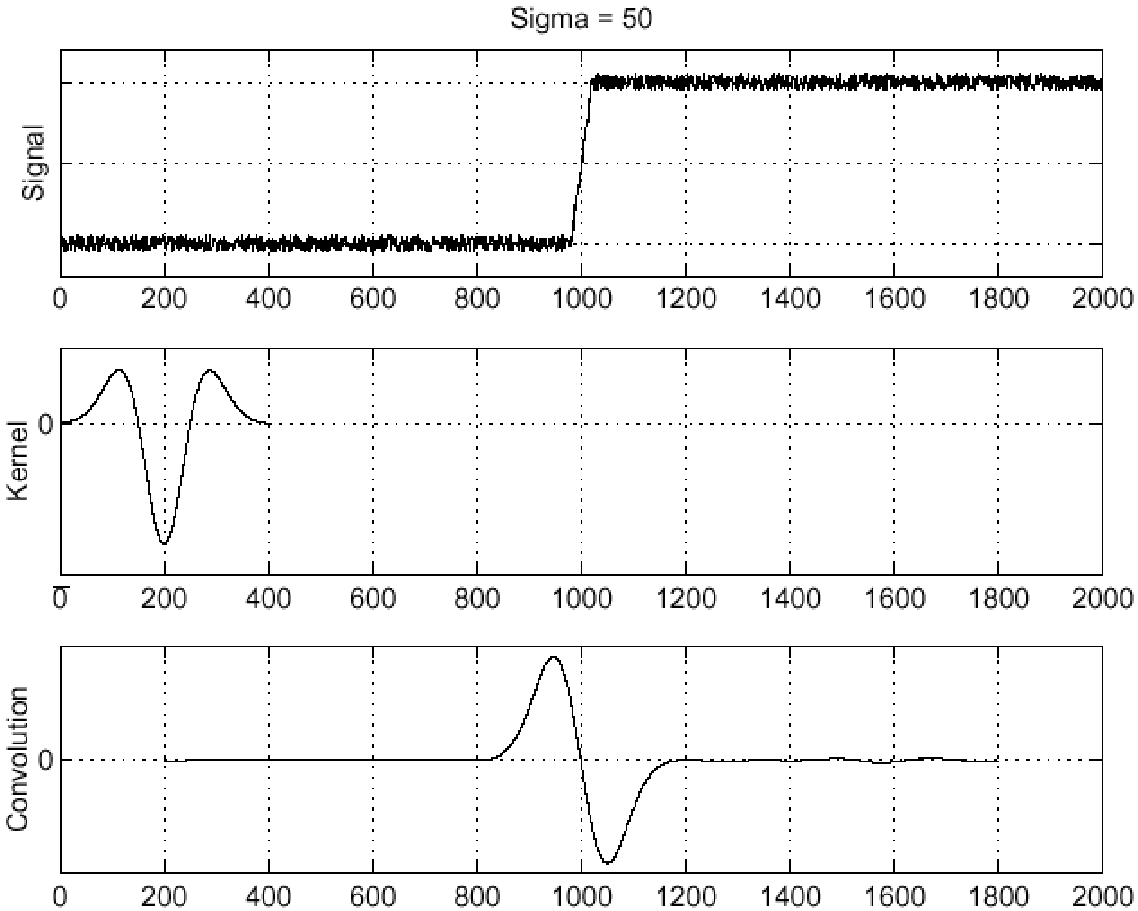
$f * \frac{d}{dx} g$

Edge = maximum
of derivative



Source: S. Seitz

Edge detection, Take 2



f — Edge

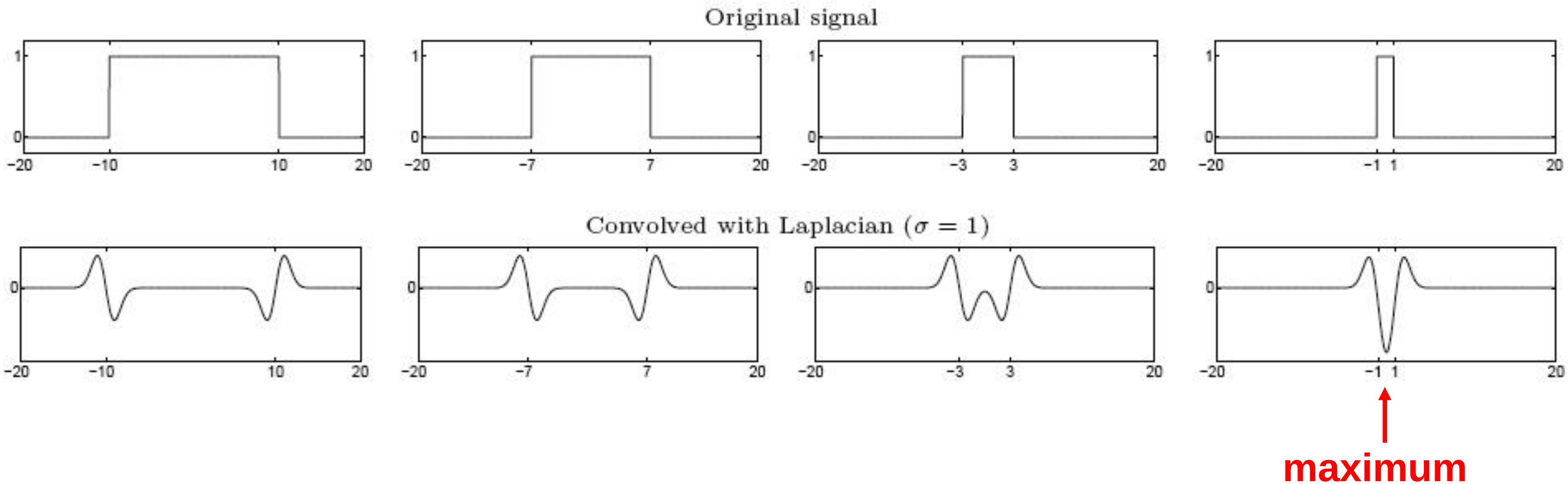
$\frac{d^2}{dx^2} g$ — Second derivative of Gaussian (Laplacian)

$f * \frac{d^2}{dx^2} g$ — Edge = zero crossing of second derivative

Source: S. Seitz

From edges to blobs

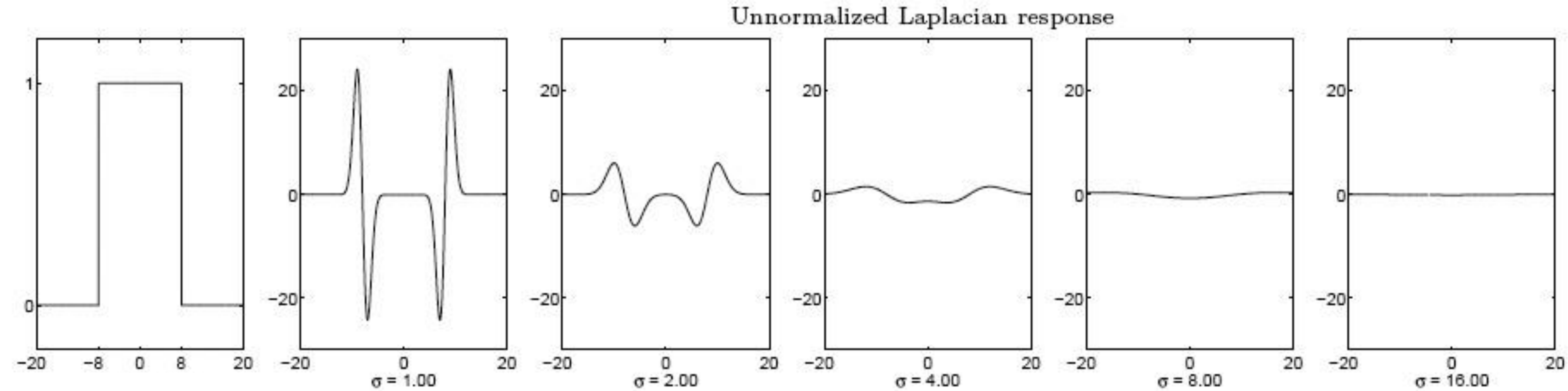
- Edge = ripple
- Blob = superposition of two ripples



Spatial selection: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is "matched" to the scale of the blob

Scale selection

- We want to find the characteristic scale of the blob by convolving it with Laplacians at several scales and looking for the maximum response
- However, Laplacian response decays as scale increases:



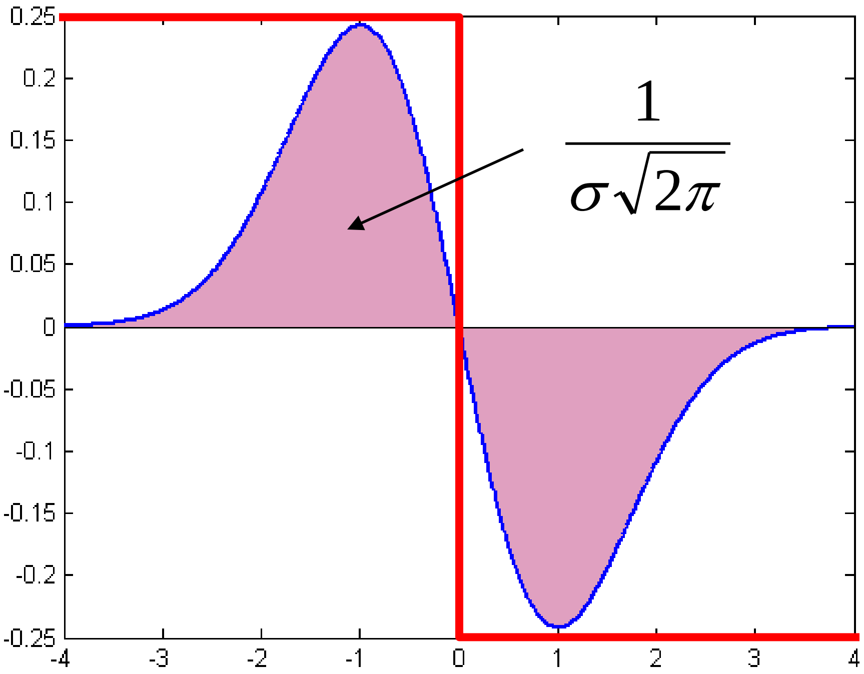
original signal (radius=8)

increasing σ →

Why does this happen?

Scale normalization

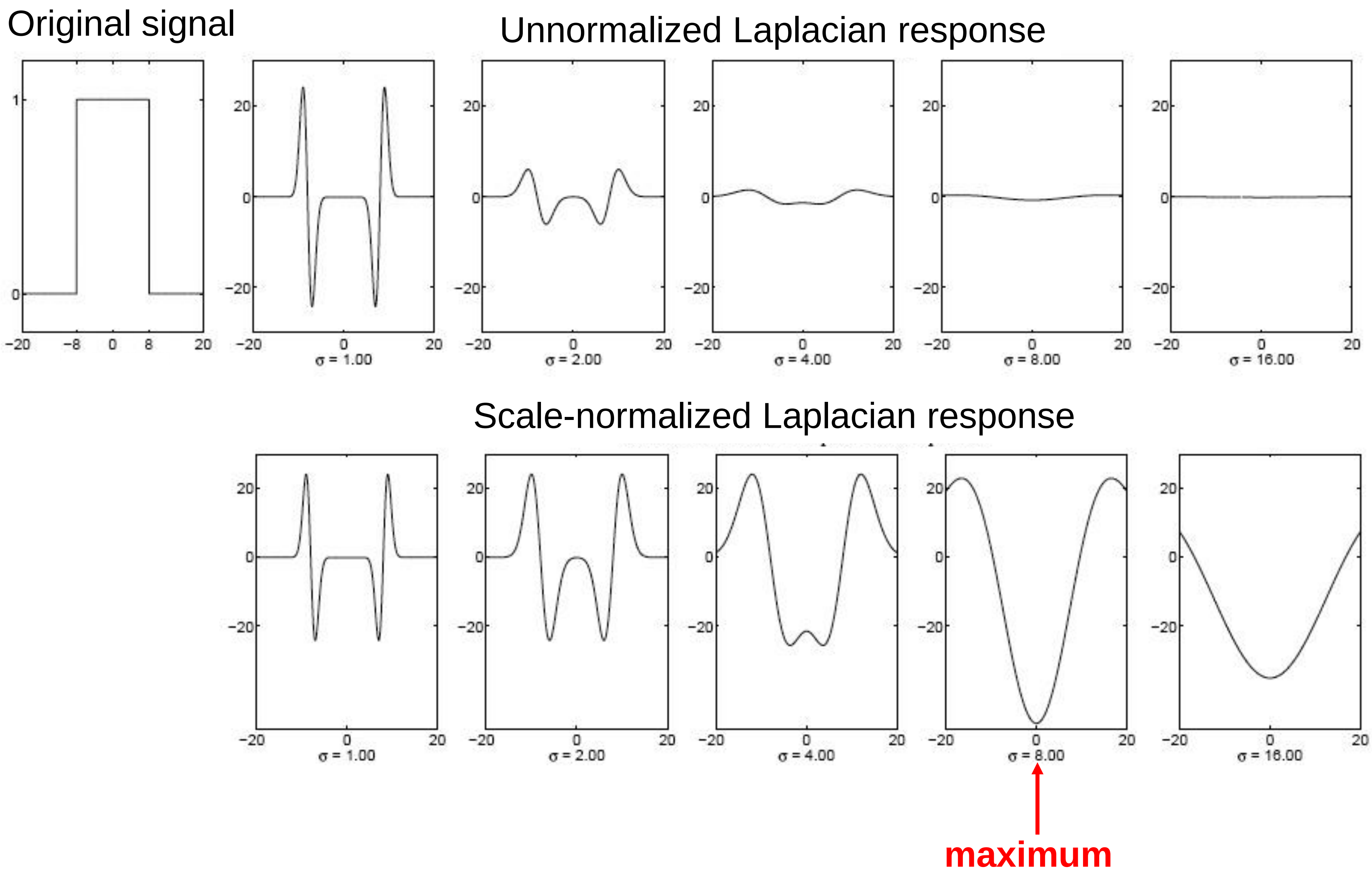
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases



Scale normalization

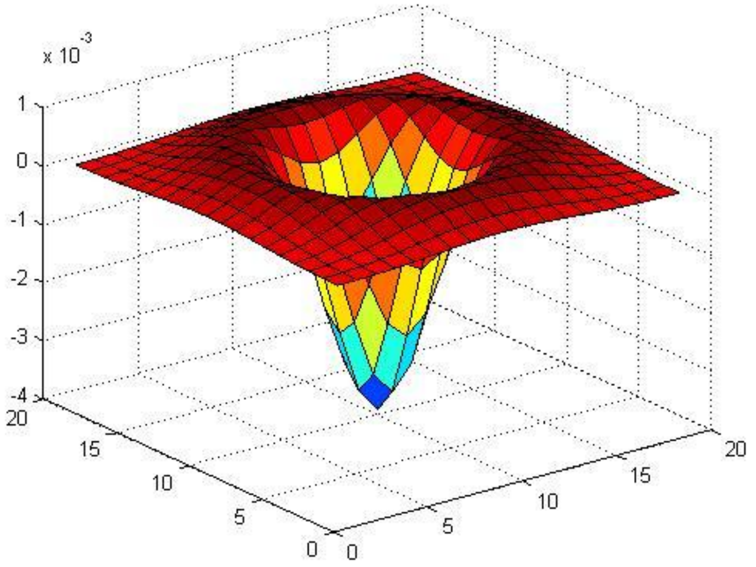
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases
- To keep response the same (scale-invariant), must multiply Gaussian derivative by σ
- Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

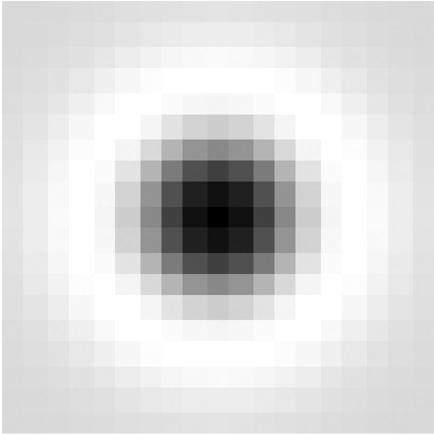
Effect of scale normalization



Blob detection in 2D

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

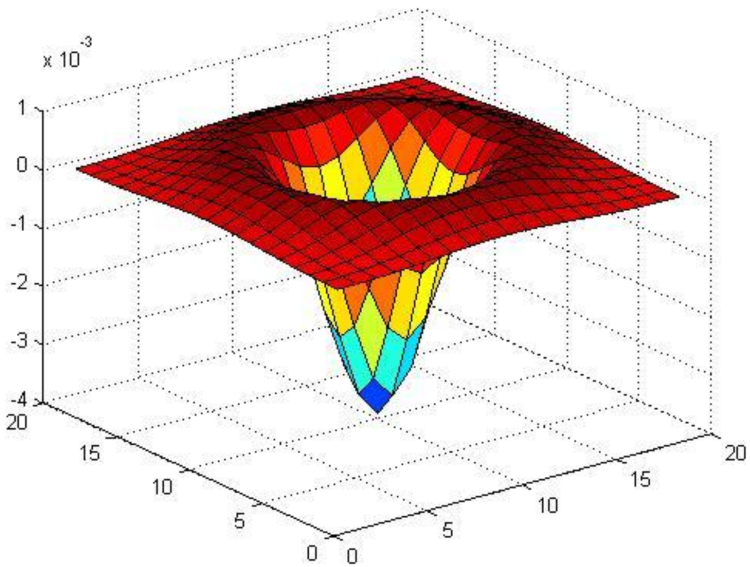


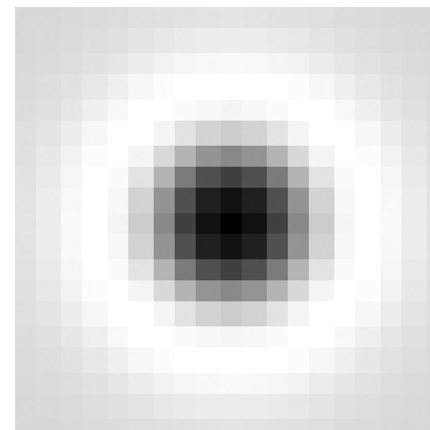


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} = \frac{1}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Blob detection in 2D

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

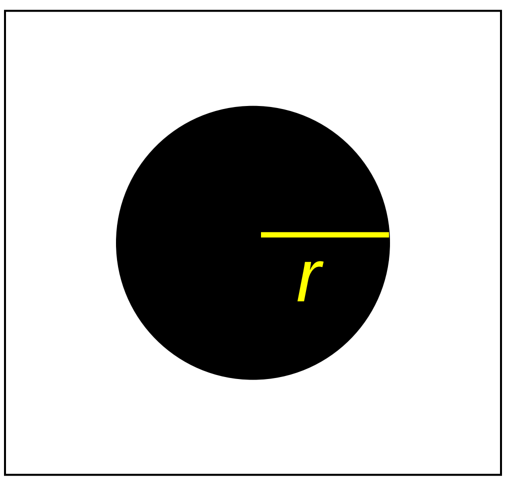




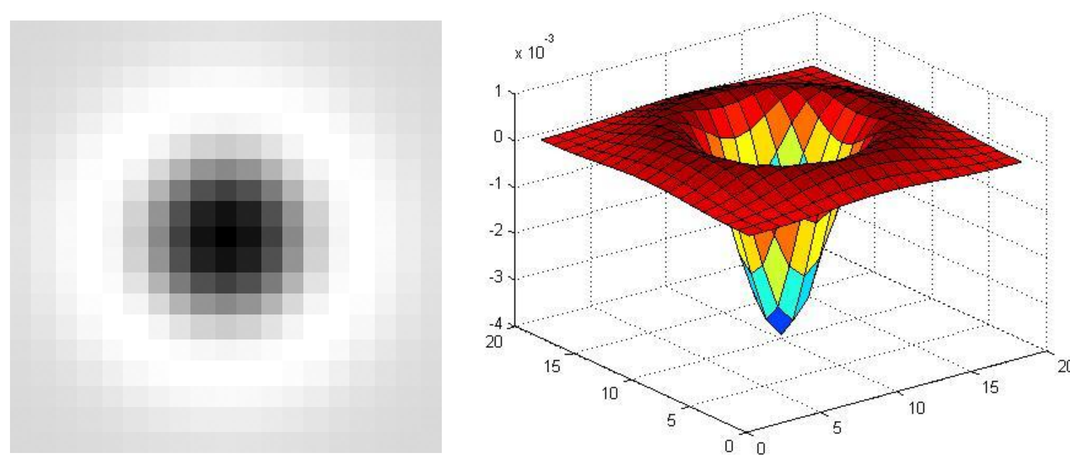
Scale-normalized: $$\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

Scale selection

- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?



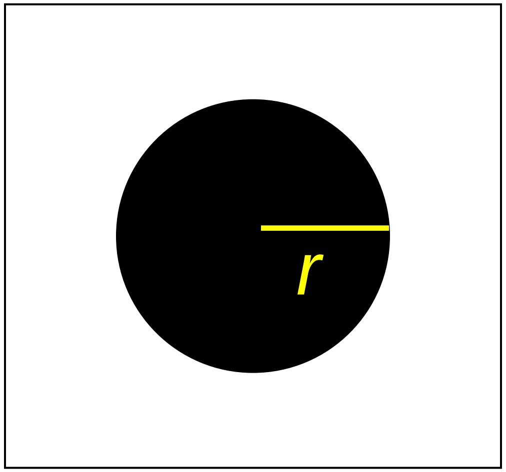
image



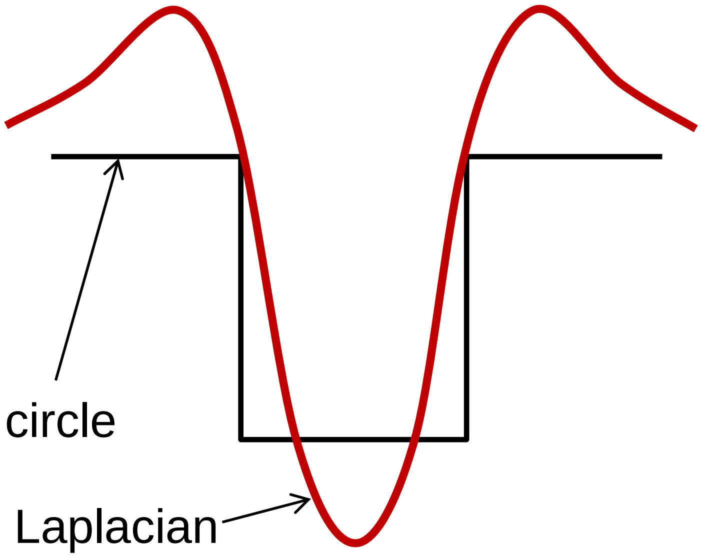
Laplacian

Scale selection

- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- To get maximum response, the zeros of the Laplacian have to be aligned with the circle
- Zeros of Laplacian is given by (up to scale): $\left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) = 0$
- Therefore, the maximum response occurs at $\sigma = r / \sqrt{2}$.

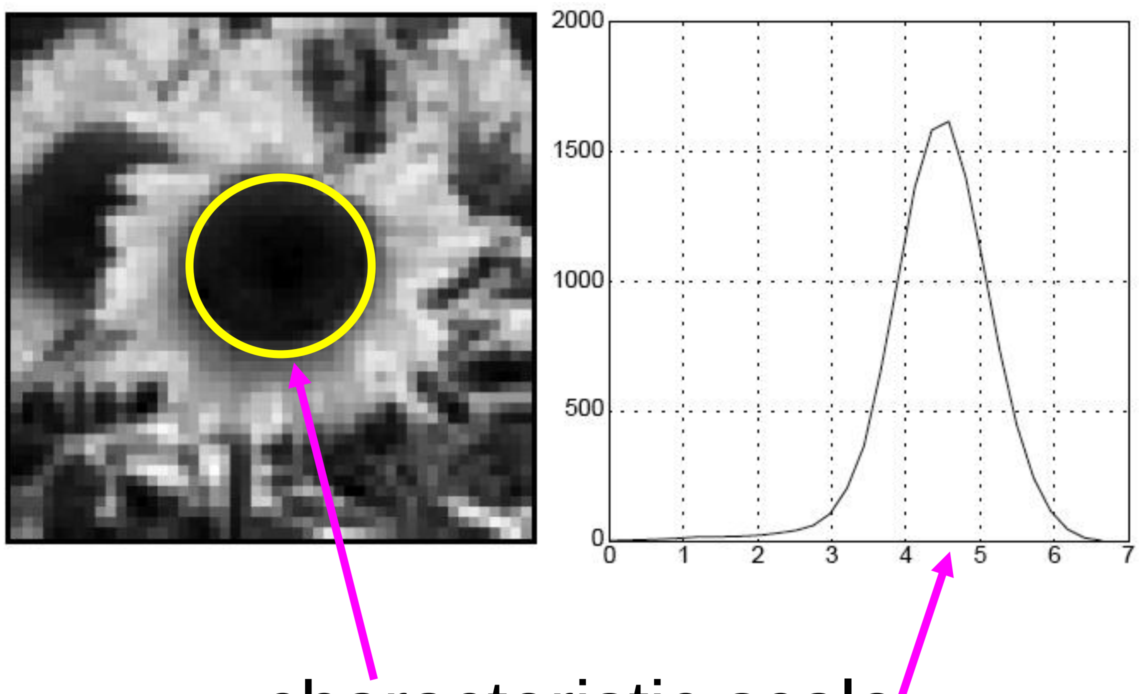


image



Characteristic scale

- We define the characteristic scale of a blob as the scale that produces peak of Laplacian response in the blob center

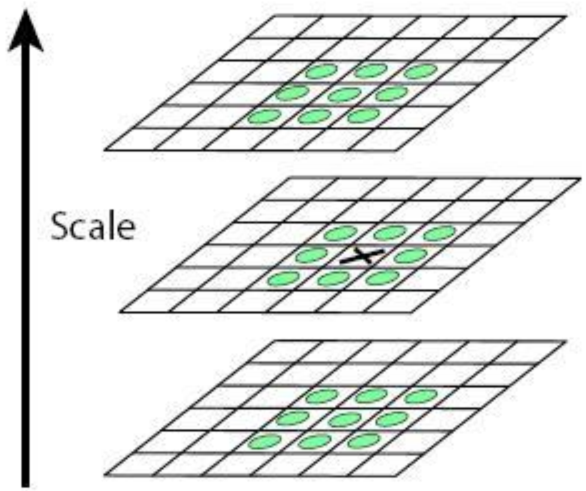


characteristic scale

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

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales
2. Find maxima of squared Laplacian response in scale-space

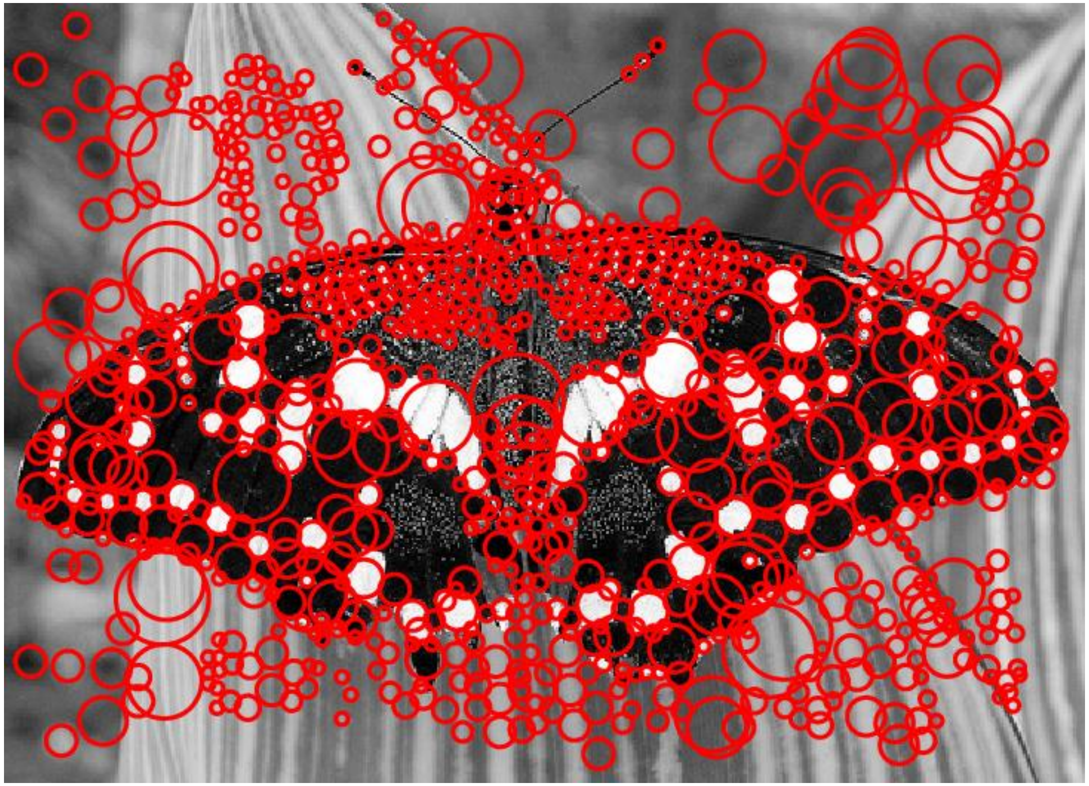


Scale-space blob detector: Example

Scale-space blob detector: Example

sigma = 11.9912

Scale-space blob detector: Example



Efficient implementation

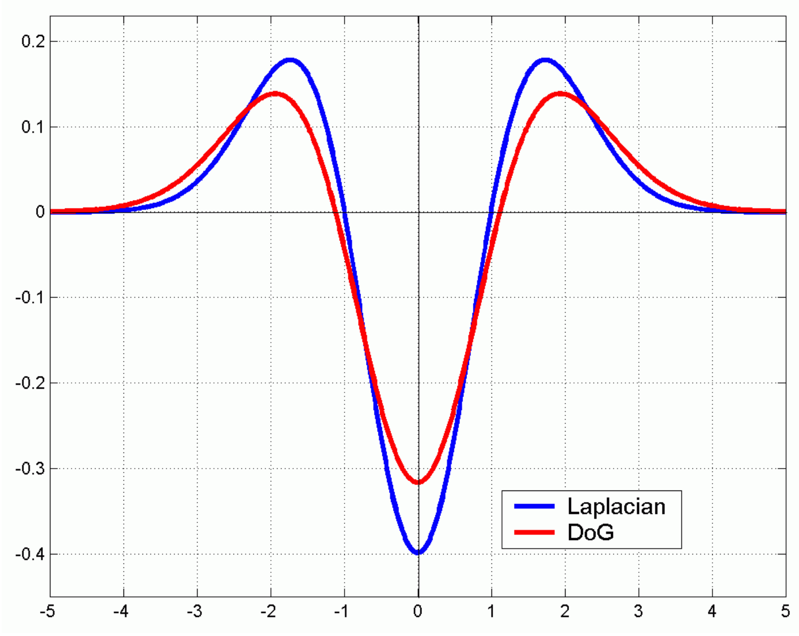
Approximating the Laplacian with a difference of Gaussians:

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

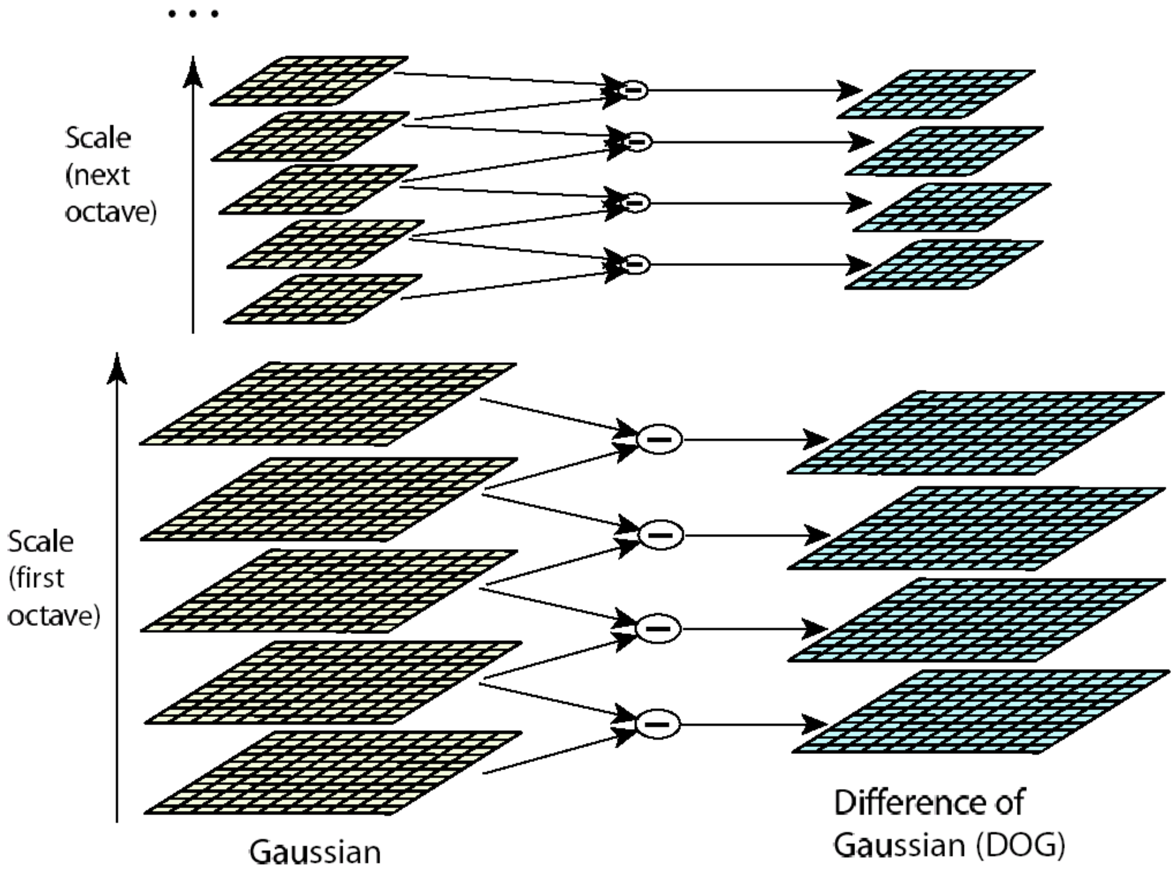
(Laplacian)

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

(Difference of Gaussians)



Efficient implementation



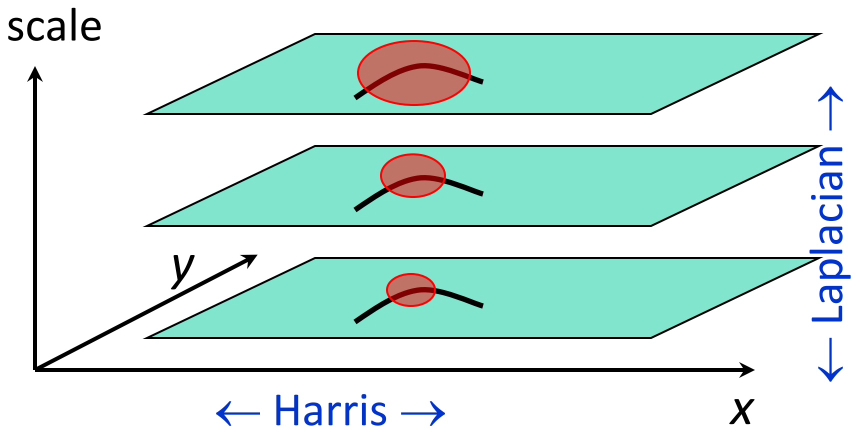
David G. Lowe. **["Distinctive image features from scale-invariant keypoints."]** *IJCV* 60 (2), pp. 91-110, 2004.

Scale Invariant Detectors

Harris-Laplacian[1]

Find local maximum of:

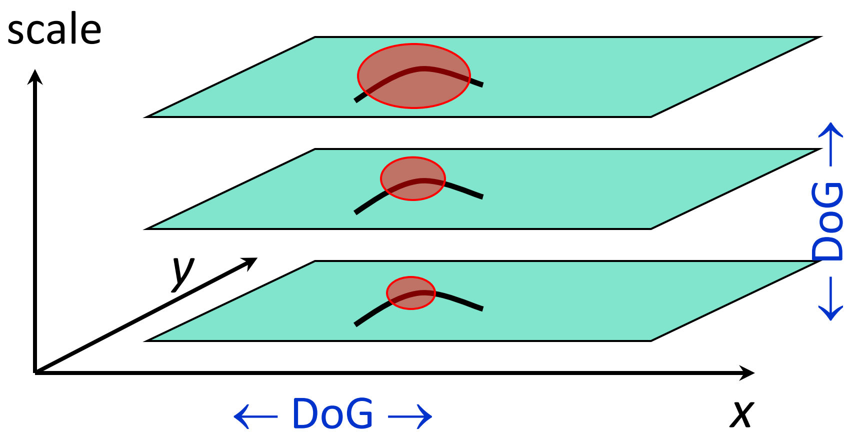
- Harris corner detector in space (image coordinates)
- Laplacian in scale



- Difference of Gaussians
- a.k.a. SIFT (Lowe)[2]

Find local maximum of:

  - Difference of Gaussians in space and scale



[1] K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

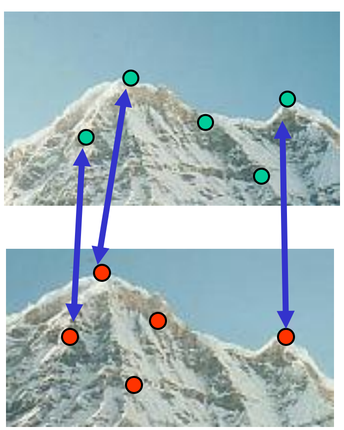
[2] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJCV 2004

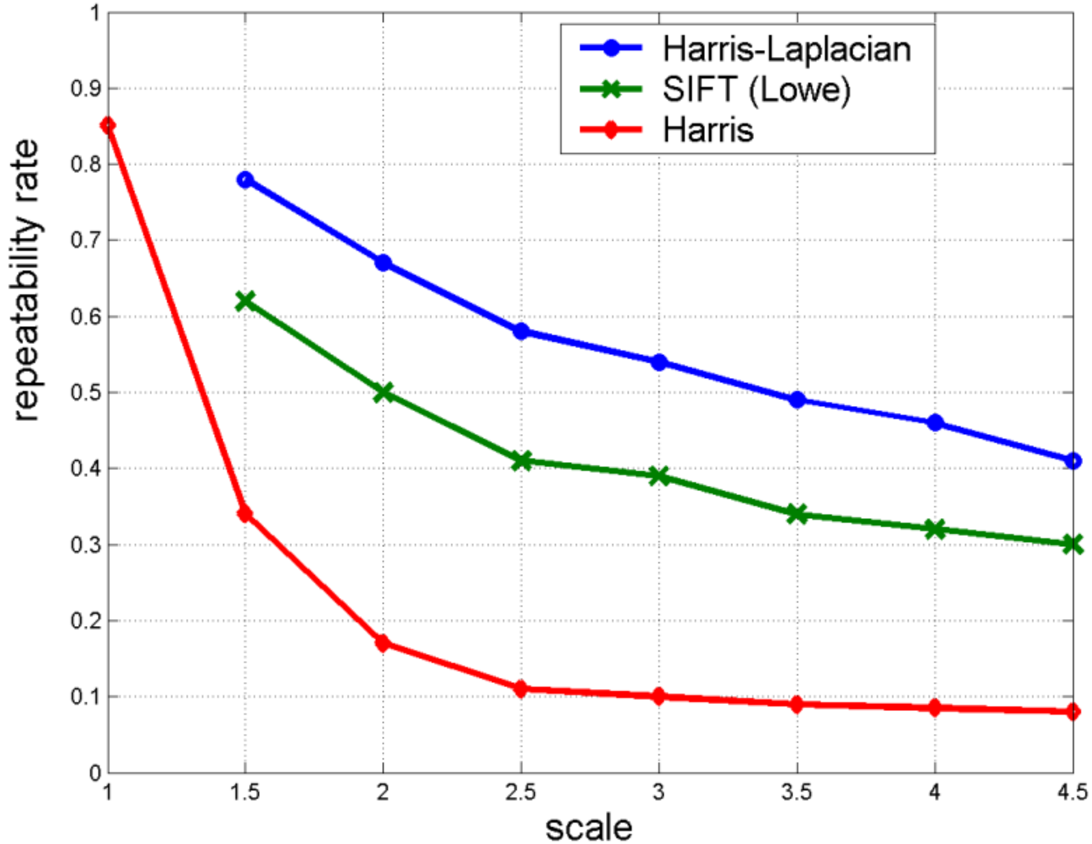
Scale Invariant Detectors

Experimental evaluation of detectors w.r.t. scale change

Repeatability rate:

$$\frac{\#\ \text{correspondences}}{\#\ \text{possible correspondences}}$$





K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

Invariance and covariance properties

- Laplacian (blob) response is *invariant* w.r.t. rotation and scaling
- Blob location is *covariant* w.r.t. rotation and scaling
- What about intensity change?

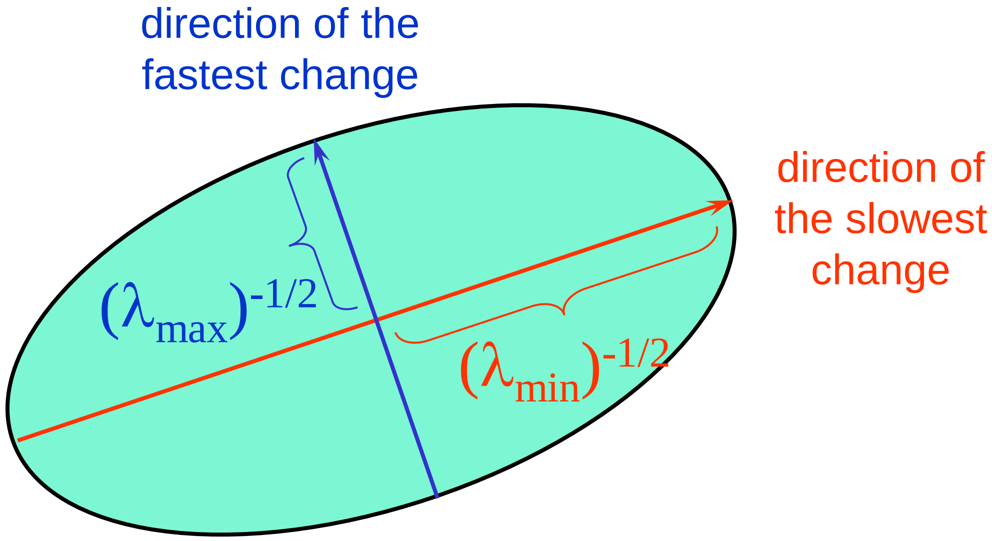
Achieving affine covariance

Consider the second moment matrix of the window containing the blob:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

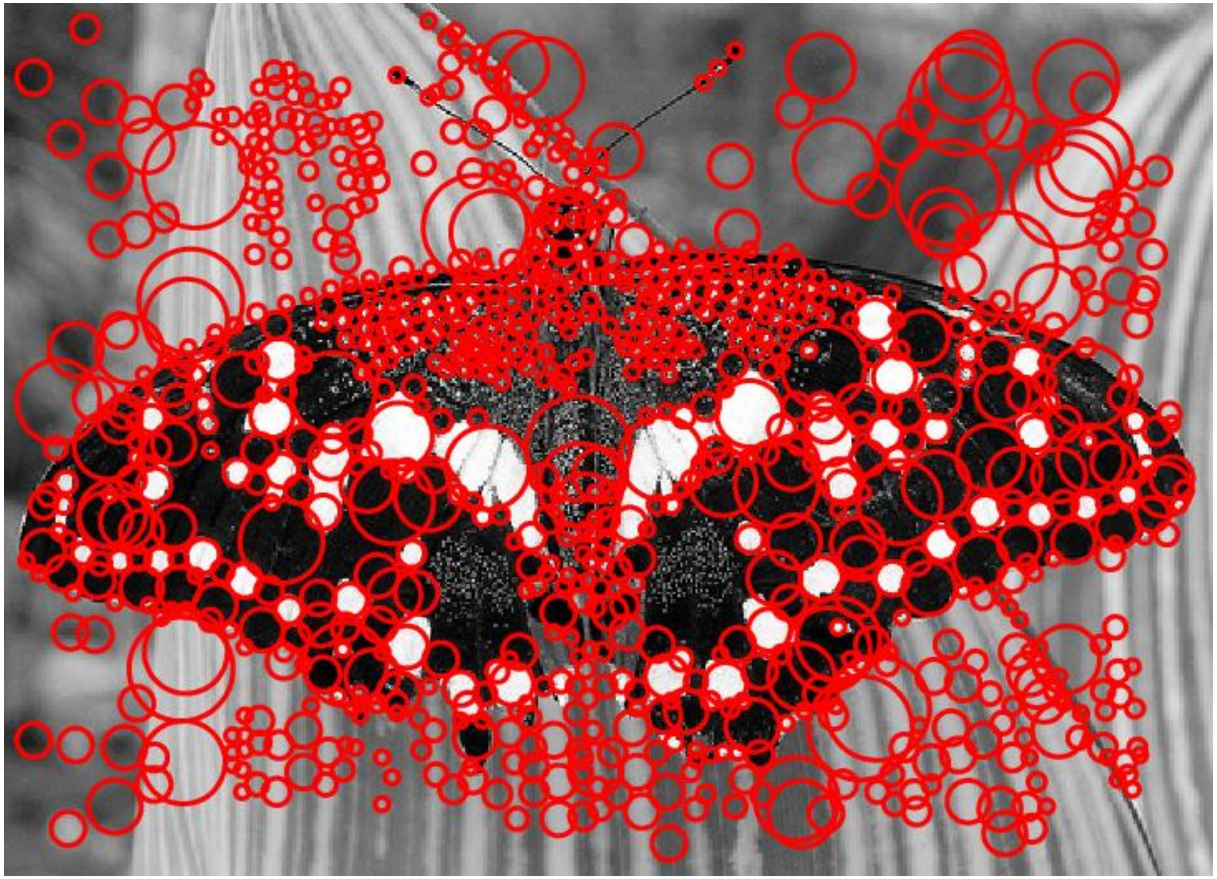
Recall:

$$[u \;\; v] \;\; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$



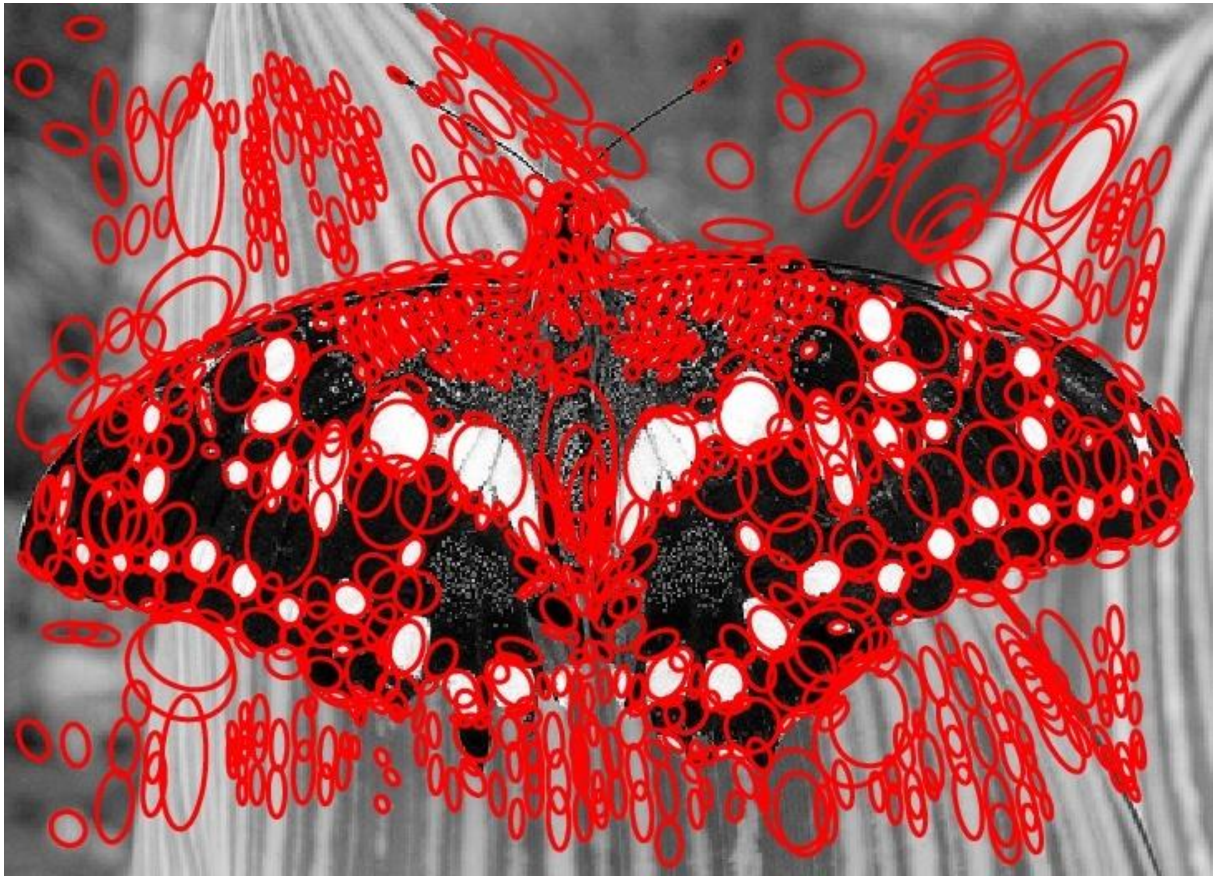
This ellipse visualizes the "characteristic shape" of the window

Affine adaptation example



Scale-invariant regions (blobs)

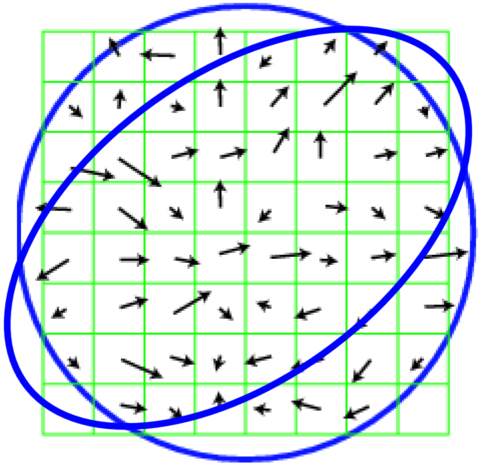
Affine adaptation example



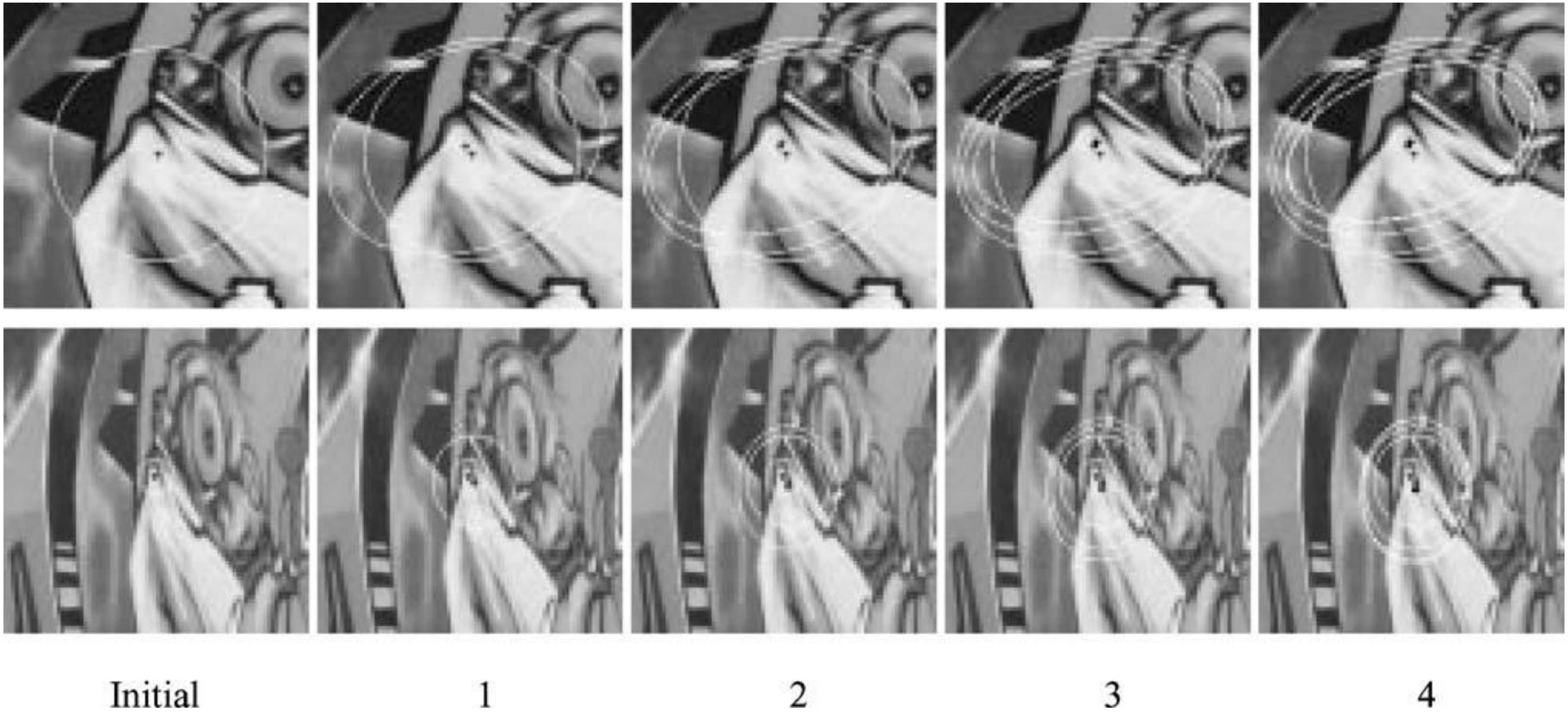
Affine-adapted blobs

Affine adaptation

- Problem: the second moment "window" determined by weights $w(x,y)$ must match the characteristic shape of the region
- Solution: iterative approach
  - Use a circular window to compute second moment matrix
  - Perform affine adaptation to find an ellipse-shaped window
  - Recompute second moment matrix using new window and iterate



Iterative affine adaptation



K. Mikolajczyk and C. Schmid, [Scale and Affine invariant interest point detectors](), IJCV 60(1):63-86, 2004.

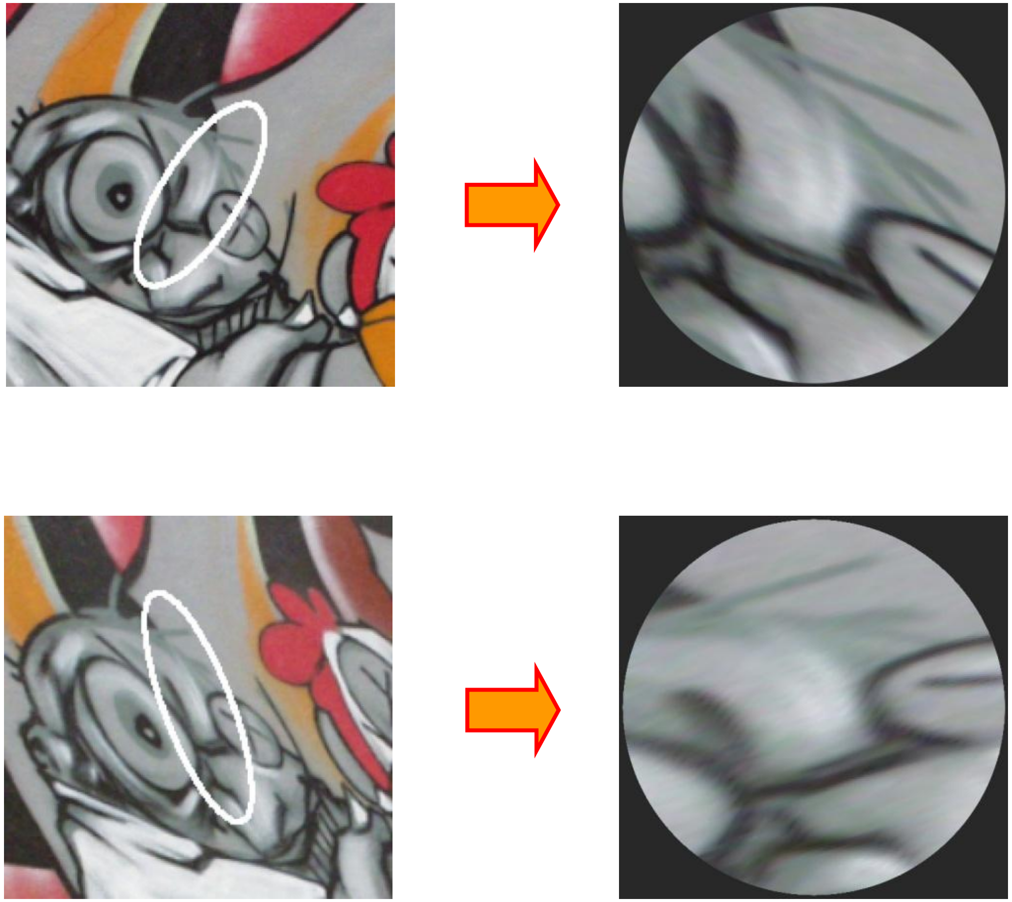
http://www.robots.ox.ac.uk/~vgg/research/affine/

Affine covariance

- Affinely transformed versions of the same neighborhood will give rise to ellipses that are related by the same transformation
- What to do if we want to compare these image regions?
- *Affine normalization*: transform these regions into same-size circles

Affine normalization

- Problem: There is no unique transformation from an ellipse to a unit circle
  - We can rotate or flip a unit circle, and it still stays a unit circle



Maximally Stable Extremal Regions

J.Matas et.al. “Distinguished Regions for Wide-baseline Stereo”. BMVC 2002.

Maximally Stable Extremal Regions

- *Threshold* image intensities: *I > thresh* for several increasing values of thresh
- Extract *connected components* (“Extremal Regions”)
- Find a threshold when region is “Maximally Stable”, i.e. *local minimum* of the relative growth
- Approximate each region with an *ellipse*

Slide: S. Seitz

Overview

- Corners (Harris Detector)
- Blobs
- Descriptors

Matching with Features

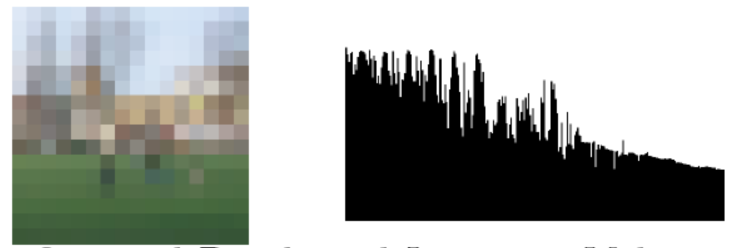
- Problem 2:
  - For each point correctly recognize the corresponding one

We need a reliable and distinctive descriptor

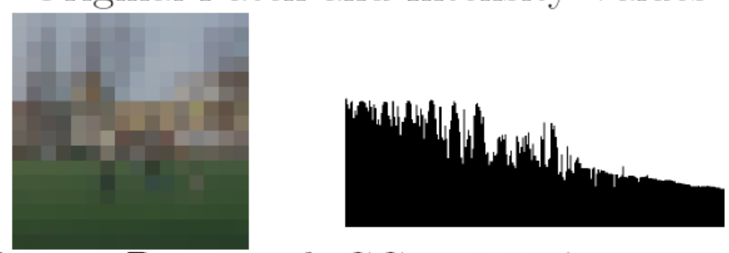
Cross-Correlation

- $$CC(P_1, P_2) = \frac{1}{N}\sum_{i}^{N} P_1[i]P_2[i].$$
- Output in range +1 → -1
- Not invariant to changes in a,b

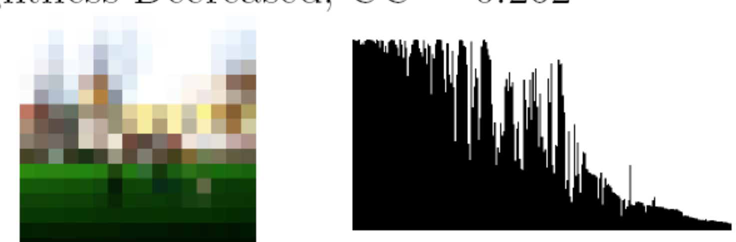
Affine photometric transformation:
$I \rightarrow a\,I + b$



Original Patch and Intensity Values



Brightness Decreased, CC = 0.262



Contrast increased, CC = 0.380

Normalized Cross-Correlation

- Make each patch zero mean:

$$\mu = \frac{1}{N} \sum_{x,y} I(x, y)$$

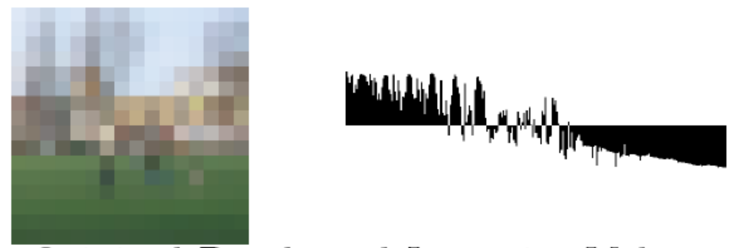
$$Z(x, y) = I(x, y) - \mu$$

- Then make unit variance:

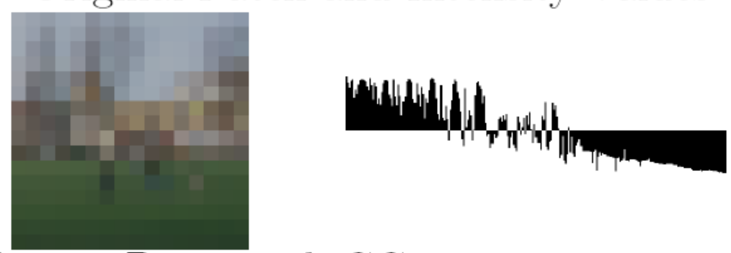
$$\sigma^2 = \frac{1}{N} \sum_{x,y} Z(x, y)^2$$

$$ZN(x, y) = \frac{Z(x, y)}{\sigma}$$

Affine photometric transformation:
$I \rightarrow a\,I + b$



Original Patch and Intensity Values



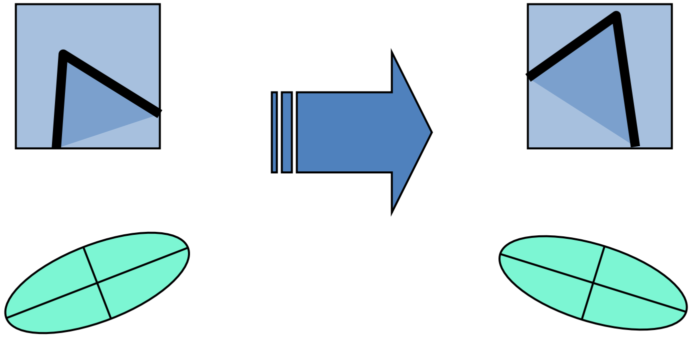
Brightness Decreased, CC = 0.999

Contrast increased, CC = 0.969

Descriptors Invariant to Rotation

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

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



C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988

Descriptors Invariant to Rotation

- Image moments in polar coordinates

$$m_{kl} = \iint r^k e^{-i\theta l} I(r,\theta) dr d\theta$$

Rotation in polar coordinates is translation of the angle:

$$\theta \rightarrow \theta + \theta_0$$

This transformation changes only the phase of the moments, but not its magnitude

Rotation invariant descriptor consists of magnitudes of moments:

$$|m_{kl}|$$

Matching is done by comparing vectors $[|m_{kl}|]_{k,l}$

J.Matas et.al. "Rotational Invariants for Wide-baseline Stereo". Research Report of CMP, 2003

Scale Invariant Feature Transform

David Lowe IJCV 2004

Basic idea:

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

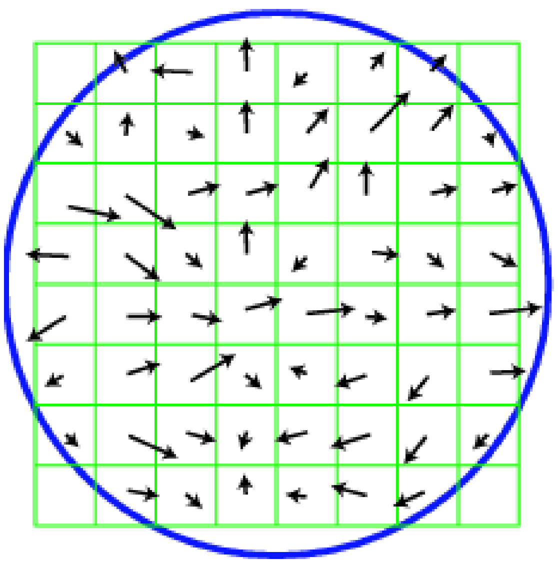
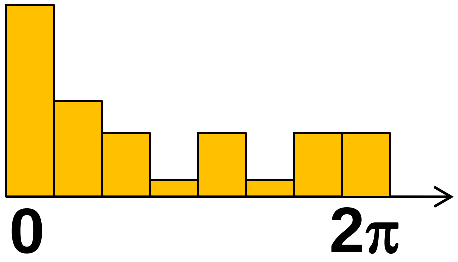
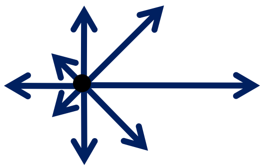


Image gradients



angle histogram

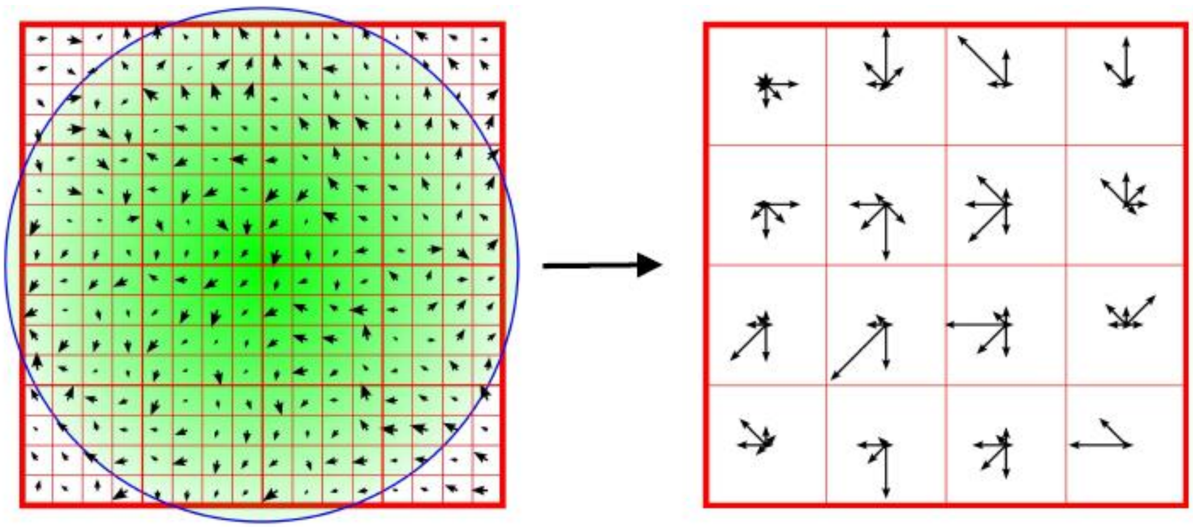


Adapted from slide by David Lowe

Former NYU faculty &
Prof. Ken Perlin's advisor

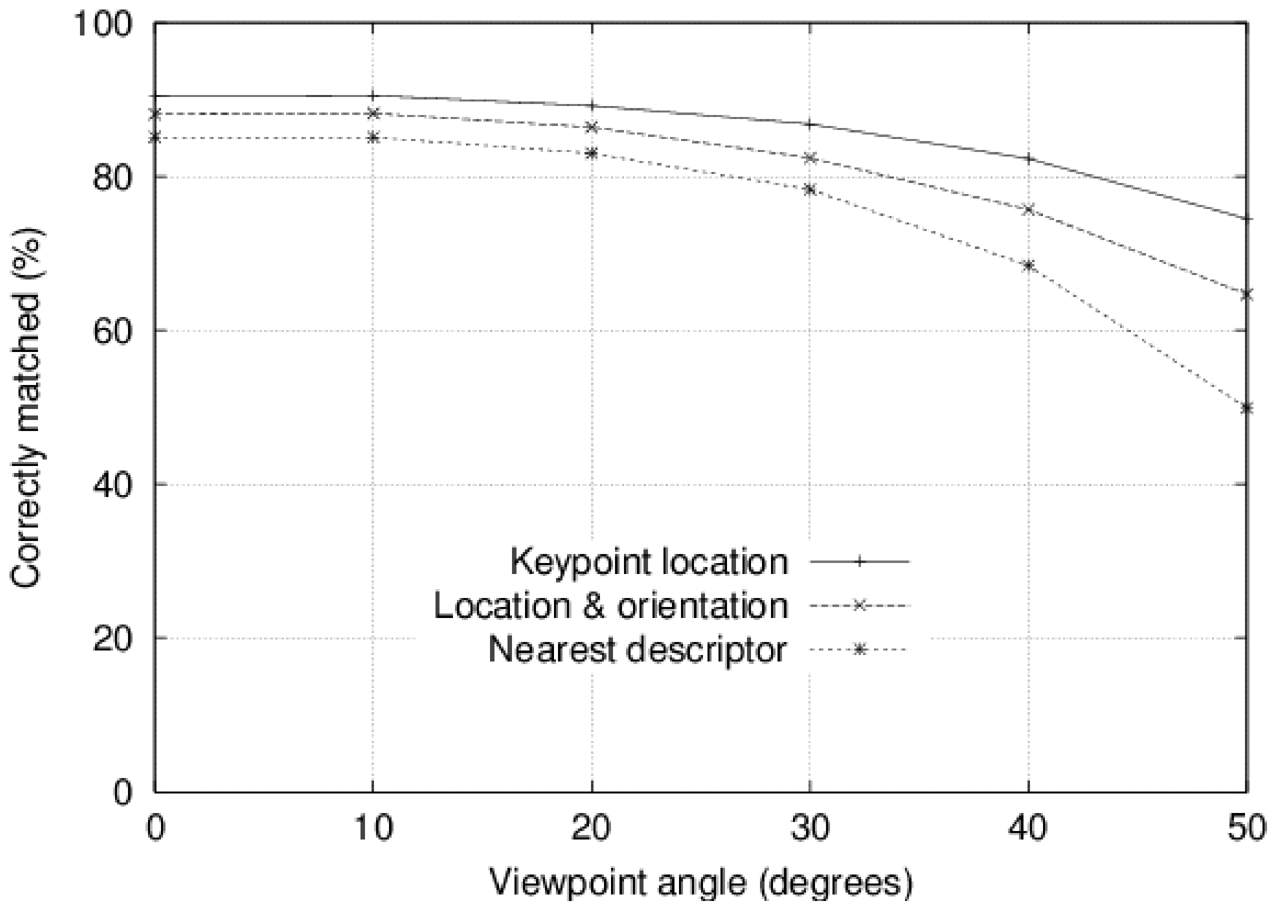
Orientation Histogram

- 4x4 spatial bins (16 bins total)
- Gaussian center-weighting
- 8-bin orientation histogram per bin
- 8 x 16 = 128 dimensions total
- Normalized to unit norm



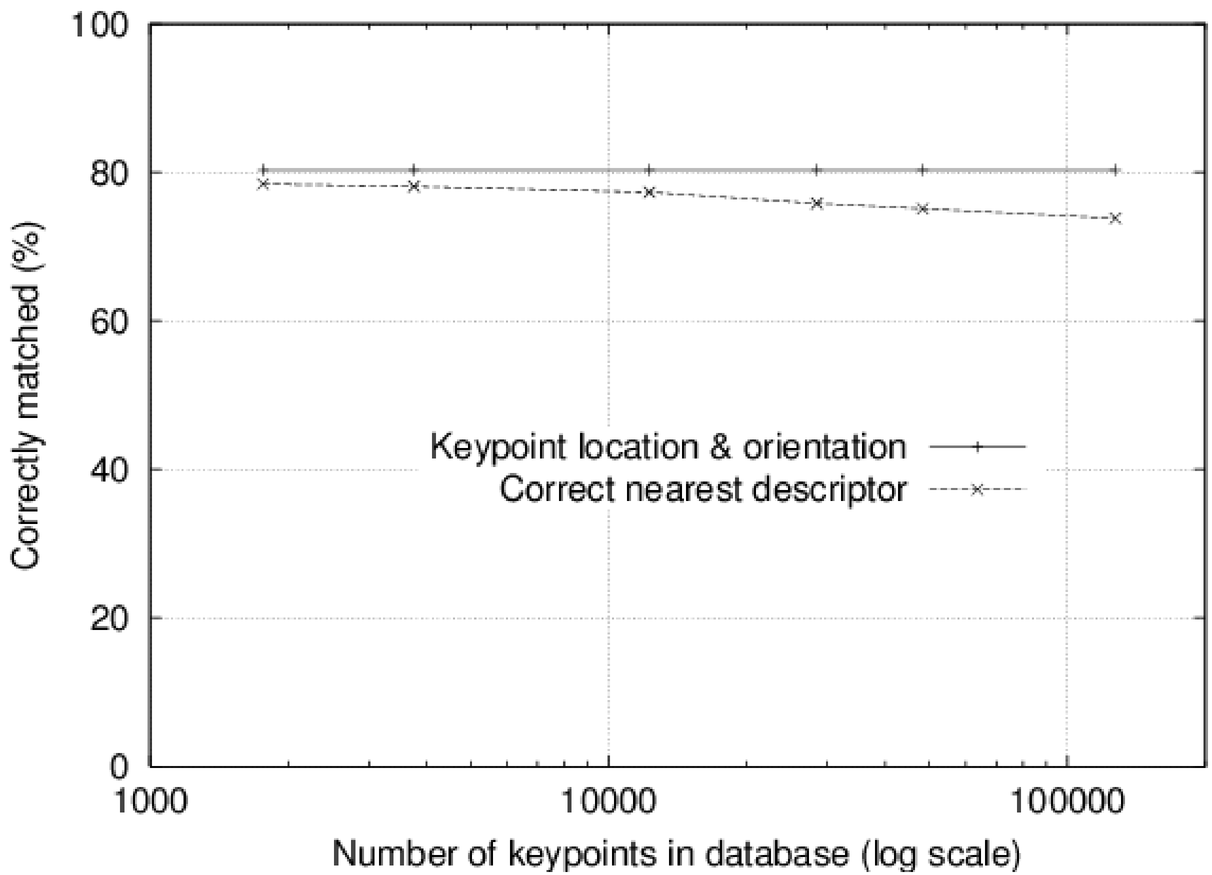
Feature stability to affine change

- Match features after random change in image scale & orientation, with 2% image noise, and affine distortion
- Find nearest neighbor in database of 30,000 features



Distinctiveness of features

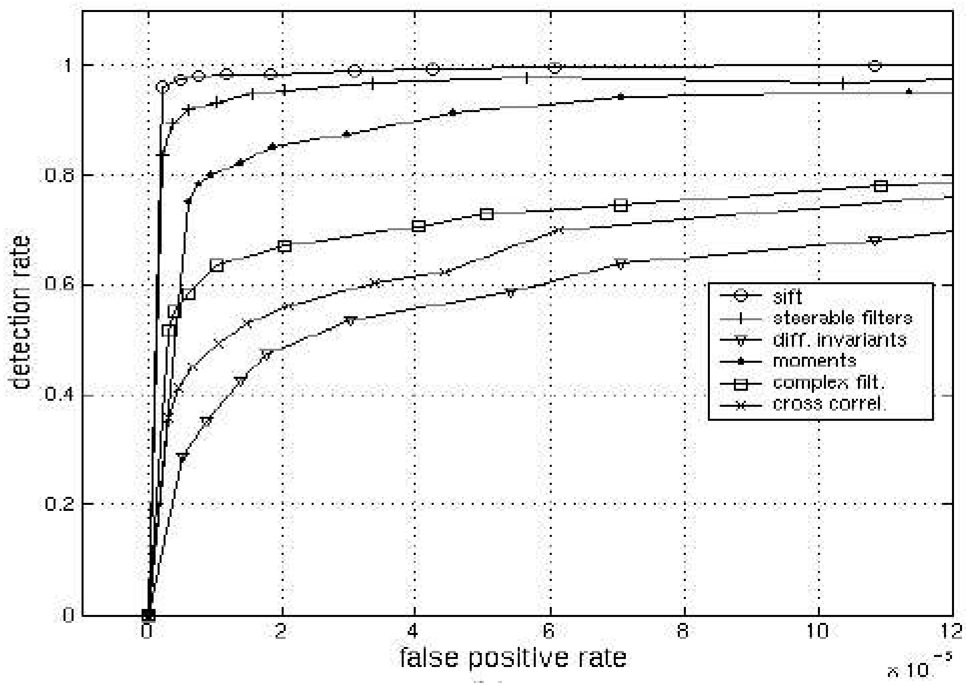
- Vary size of database of features, with 30 degree affine change, 2% image noise
- Measure % correct for single nearest neighbor match



SIFT – Scale Invariant Feature Transform[1]

- Empirically found[2] to show very good performance, invariant to *image rotation*, *scale*, *intensity change*, and to moderate *affine* transformations

Scale = 2.5
Rotation = 45^0



---

[1] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJCV 2004

[2] K.Mikolajczyk, C.Schmid. "A Performance Evaluation of Local Descriptors". CVPR 2003

SIFT invariances

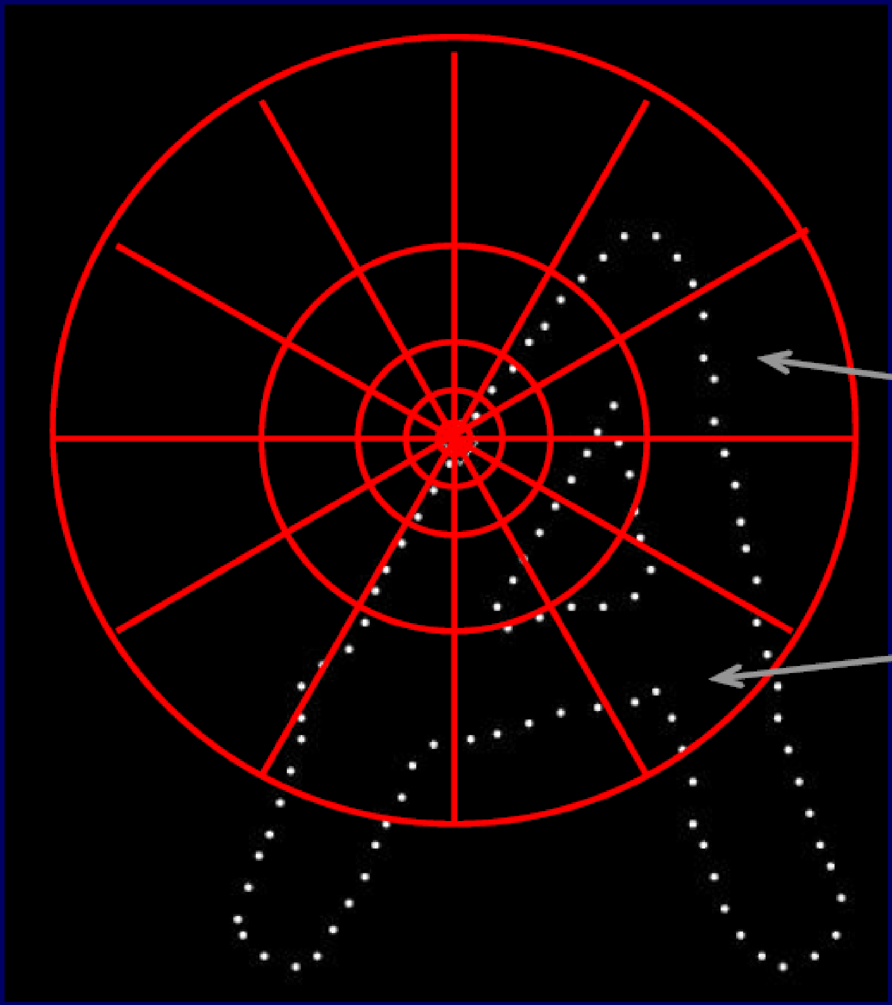
- Spatial binning gives tolerance to small shifts in location and scale
- Explicit orientation normalization
- Photometric normalization by making all vectors unit norm
- Orientation histogram gives robustness to small local deformations

Summary of SIFT

Extraordinarily robust matching technique

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

Shape Context



Count the number of points inside each bin, e.g.:

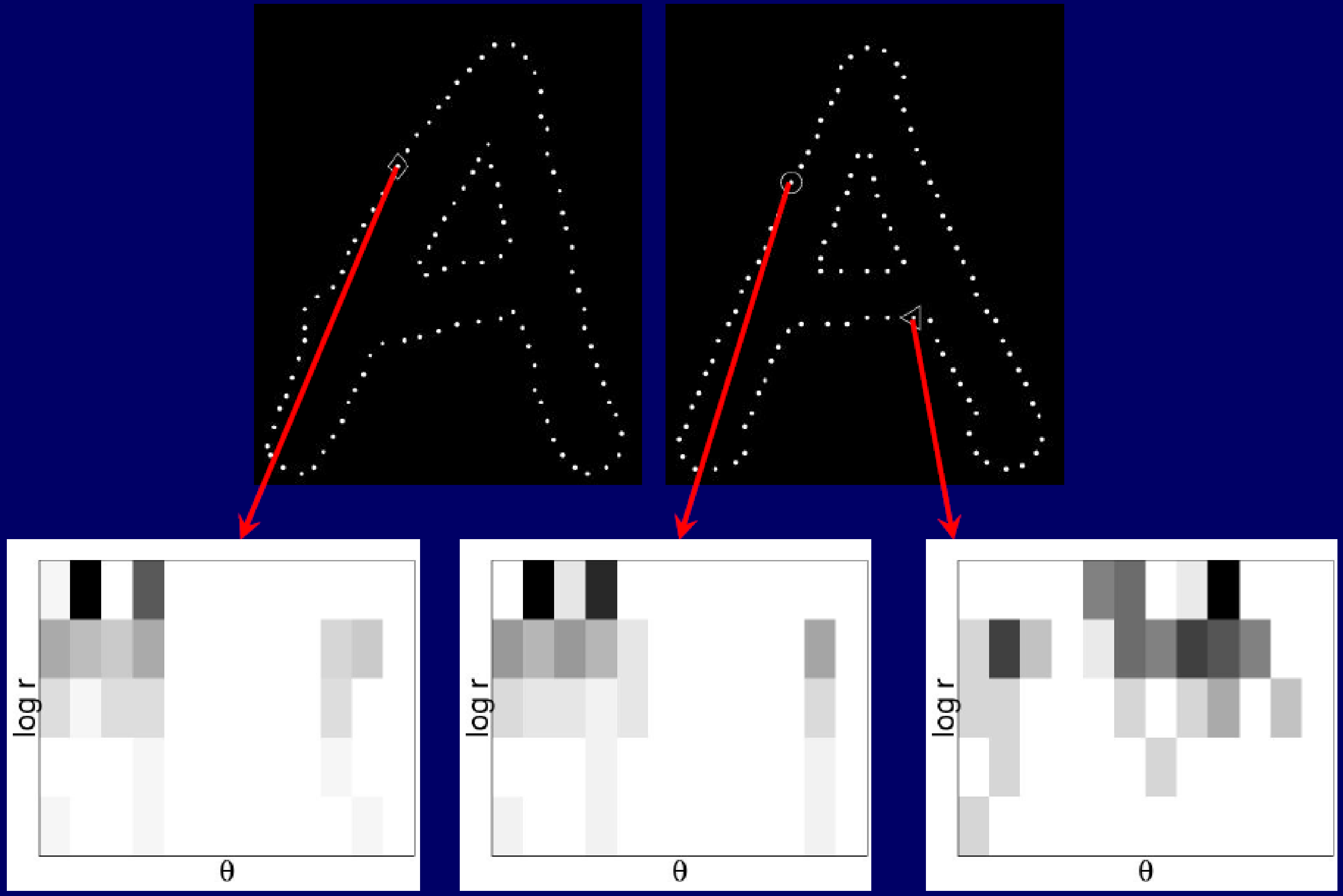
Count = 4

⋮

Count = 10

☞ Compact representation of distribution of points relative to each point

Shape Context



Feature matching

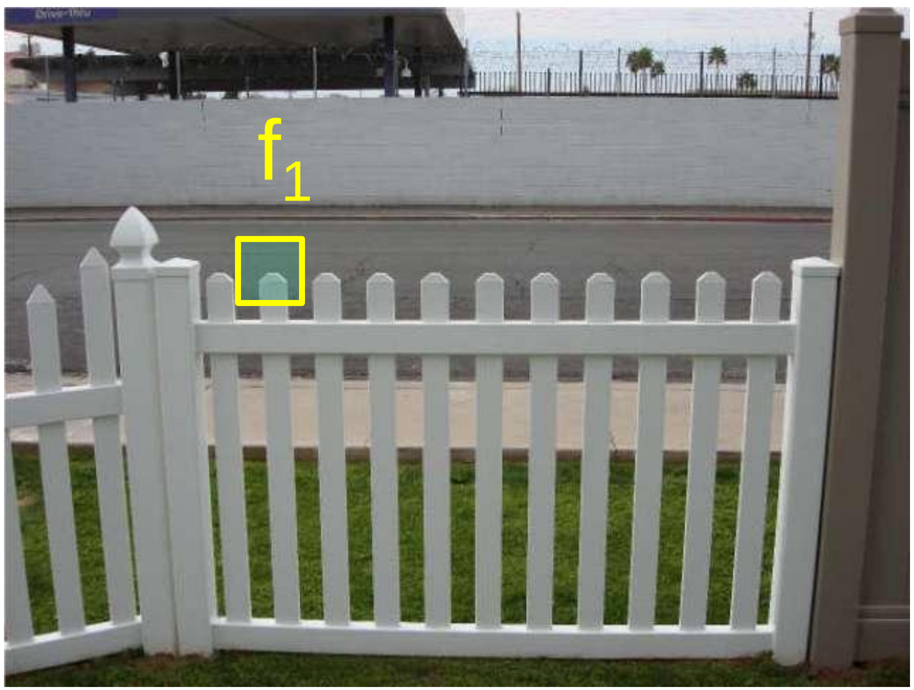
Given a feature in I_1, how to find the best match in I_2?

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

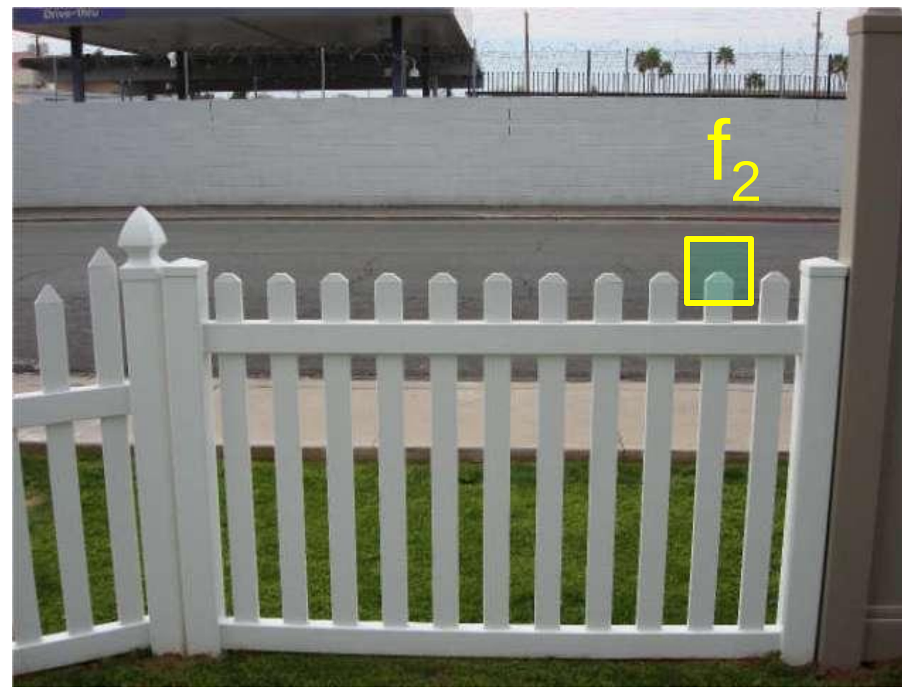
Feature distance

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



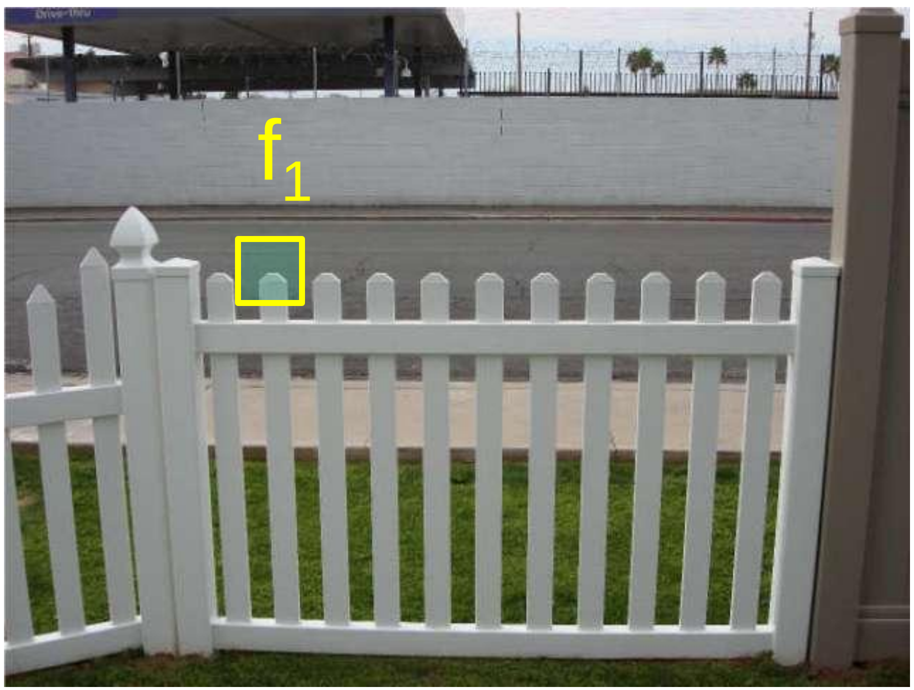
I_2

Slide: S. Seitz

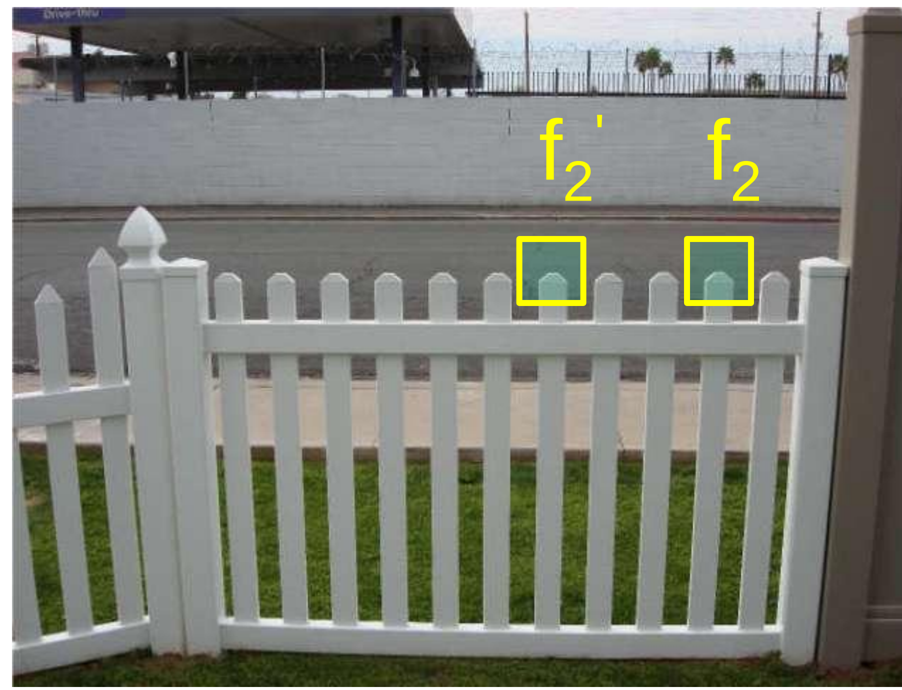
Feature distance

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 2^{nd} best SSD match to f_1 in I_2
  - gives small values for ambiguous matches



I_1

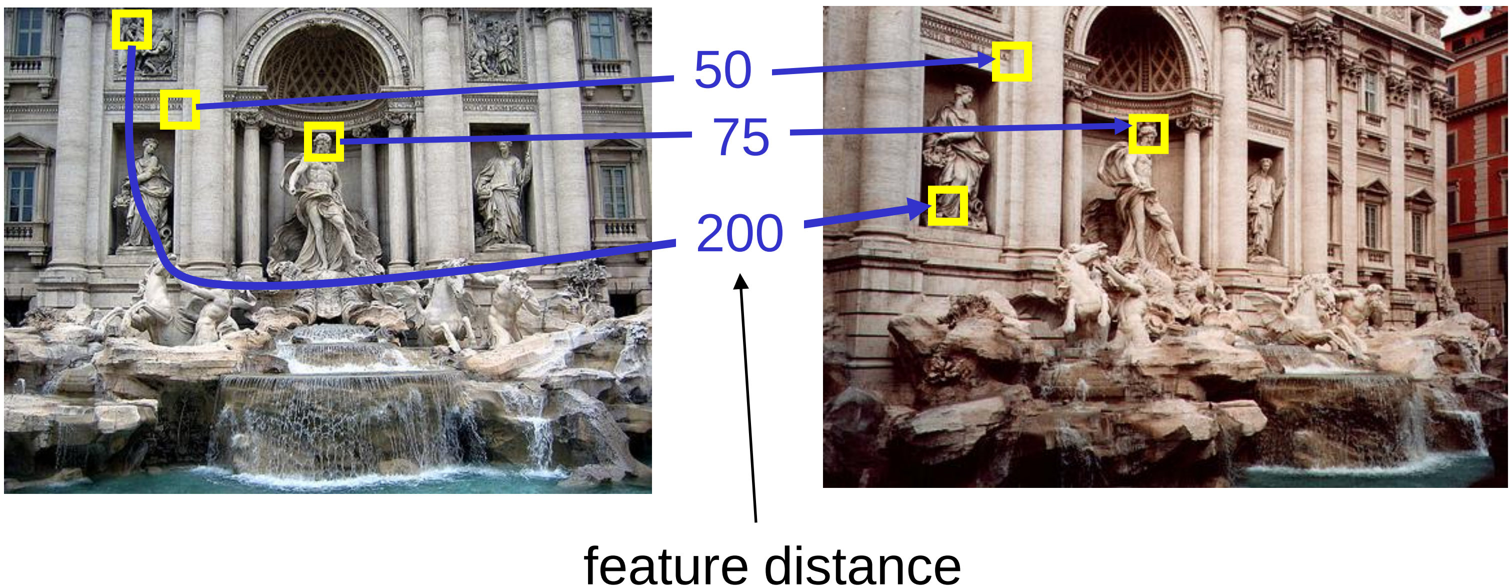


I_2

Slide: S. Seitz

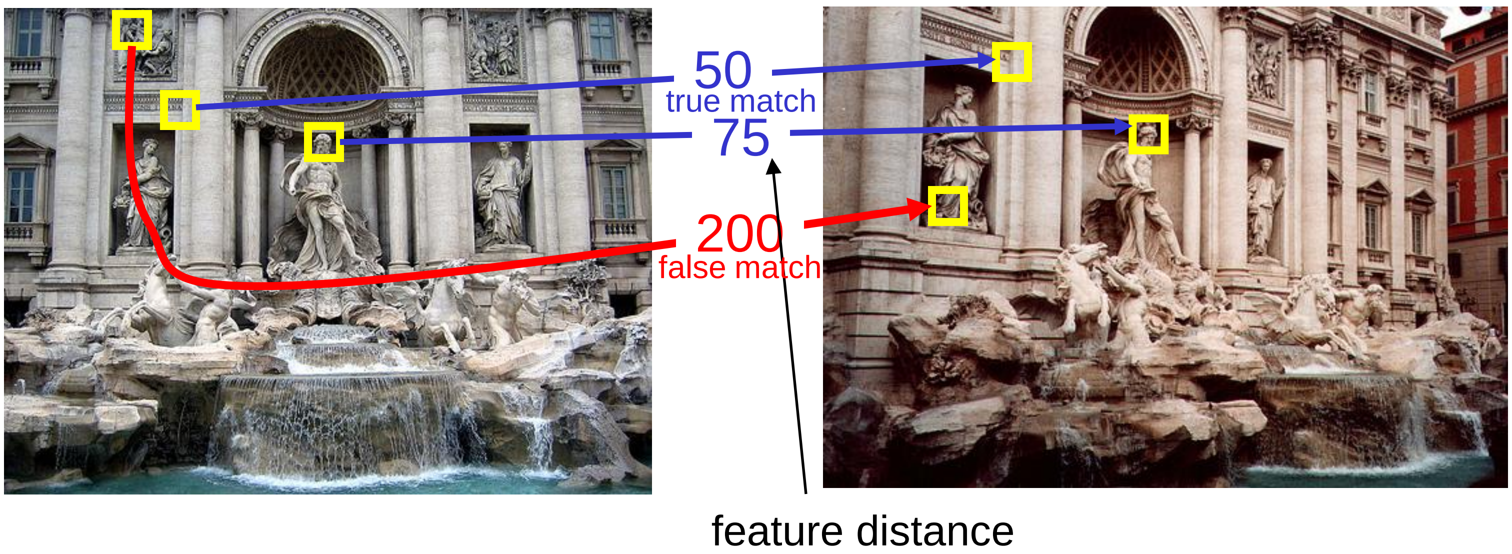
Evaluating the results

How can we measure the performance of a feature matcher?



Slide: S. Seitz

True/false positives



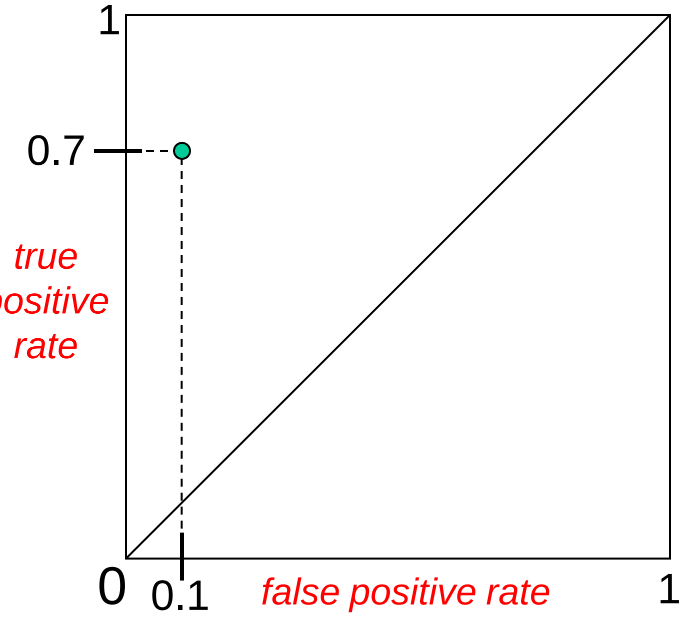
The distance threshold affects performance

- 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?

Slide: S. Seitz

Evaluating the results

How can we measure the performance of a feature matcher?



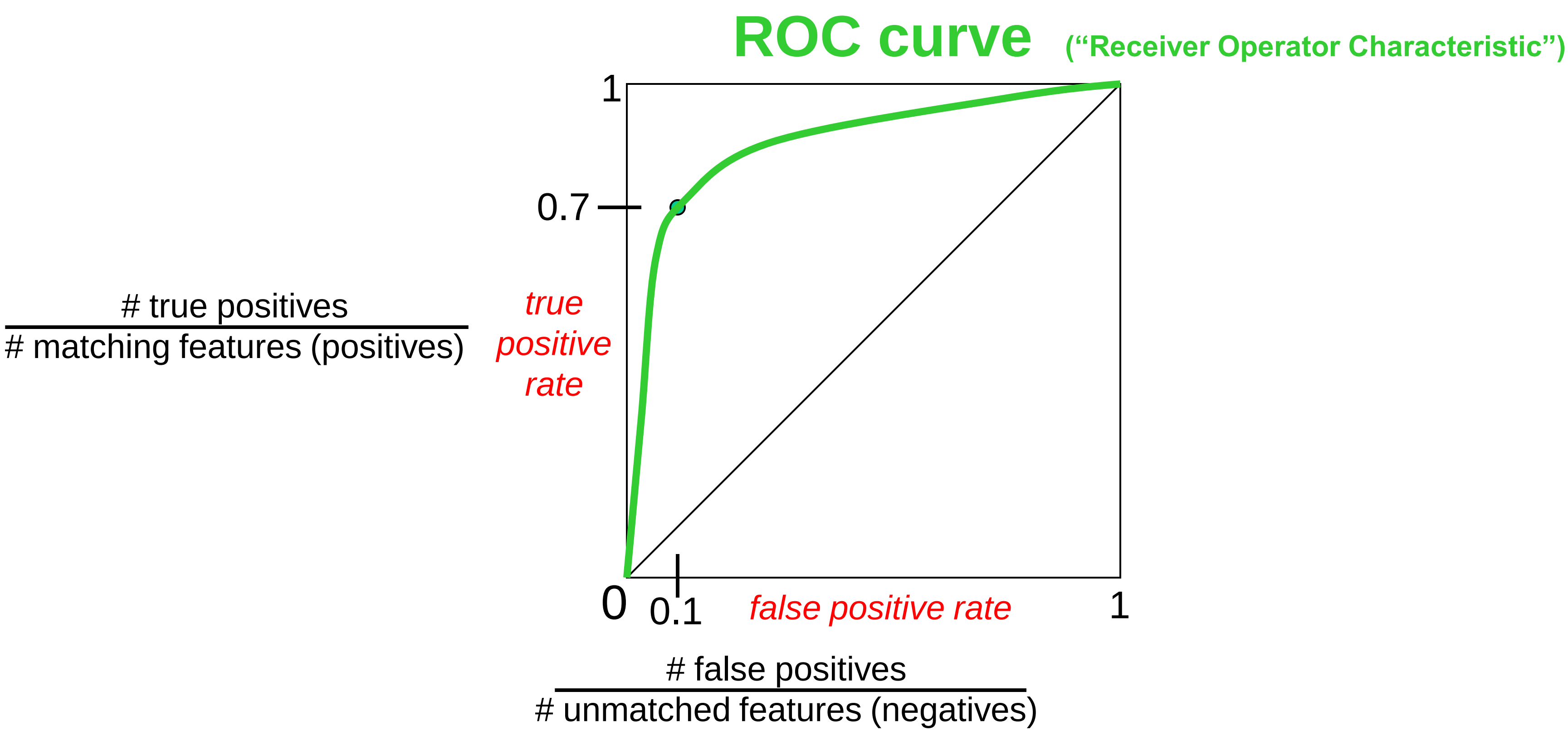
$$\text{true positive rate} = \frac{\text{\# true positives}}{\text{\# matching features (positives)}}$$

$$\text{false positive rate} = \frac{\text{\# false positives}}{\text{\# unmatched features (negatives)}}$$

Slide: S. Seitz

Evaluating the results

How can we measure the performance of a feature matcher?



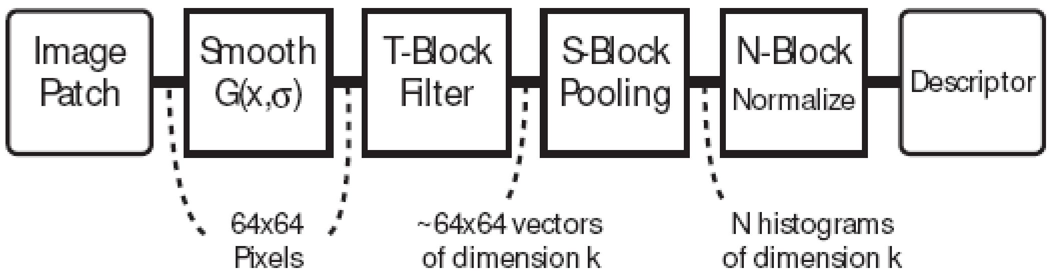
ROC Curves

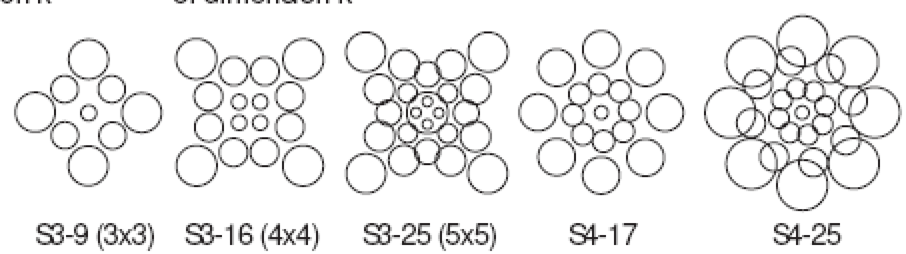
- Generated by counting # current/incorrect matches, for different threholds
- Want to maximize area under the curve (AUC)
- Useful for comparing different feature matching methods
- For more info: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

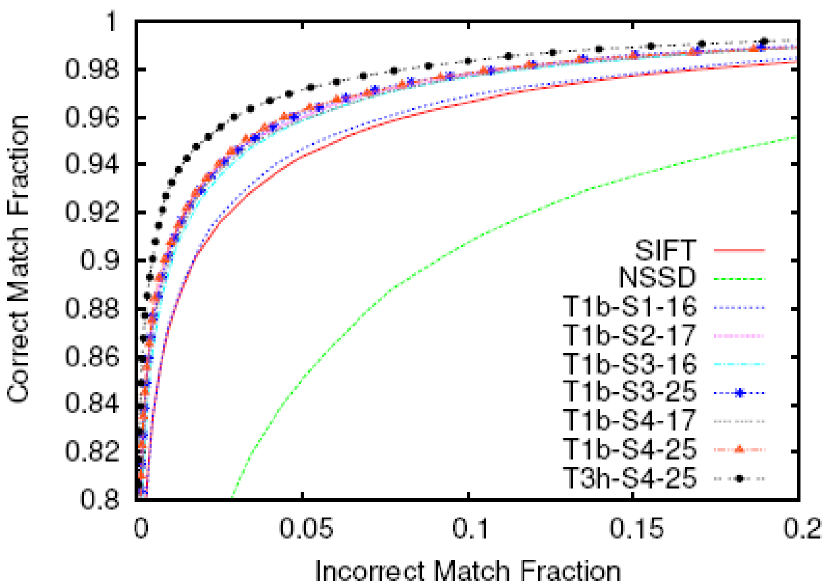
Slide: S. Seitz

Learning Local Image Descriptors

Simon A. J. Winder   Matthew Brown
Microsoft Research
1 Microsoft Way, Redmond, WA 98052, USA







The best result of all was obtained by combining steerable filters with the polar plan of S4 to give T3h-S4-25. At just under a 2% error rate, this is one third of the error rate produced by SIFT at 95% correct matches. The ROC curve for this descriptor is plotted on Figure 11. However the dimensionality is quite high at 400.

Learning Local Image Descriptors

Simon A. J. Winder   Matthew Brown
Microsoft Research
1 Microsoft Way, Redmond, WA 98052, USA

- Want same 3D world point to map to same descriptor

- Build big dataset of patches using ground-truth 3D information

Next Lecture

- 7pm Tuesday
  - Prof. Chris Bregler

- Then back to normal…..