Computer Vision

Local Features

Image matching

by Diva Sian

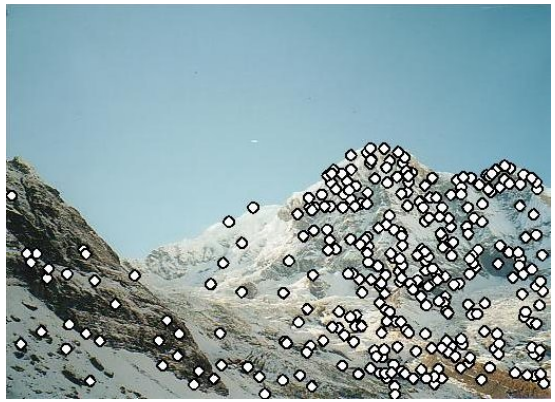
by swashford

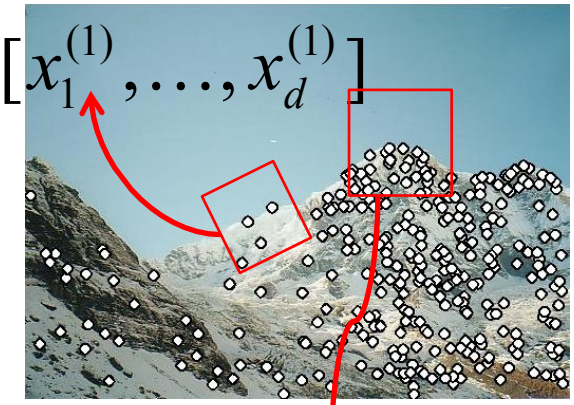
Local features: main components

1) **Detection:** Identify the interest points

2) **Description**: Extract vector feature descriptor surrounding each interest point.

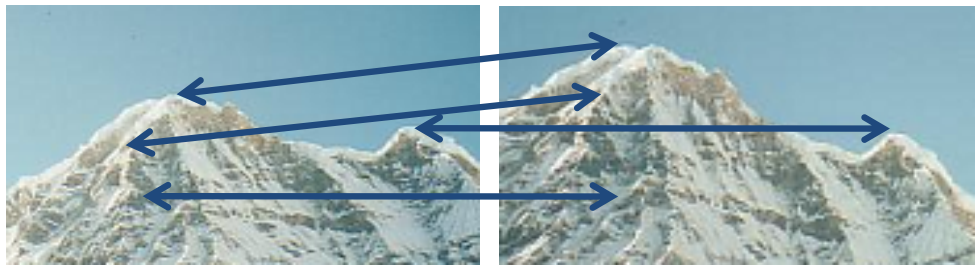
3) **Matching:** Determine correspondence between descriptors in two views





$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$



Repeatability

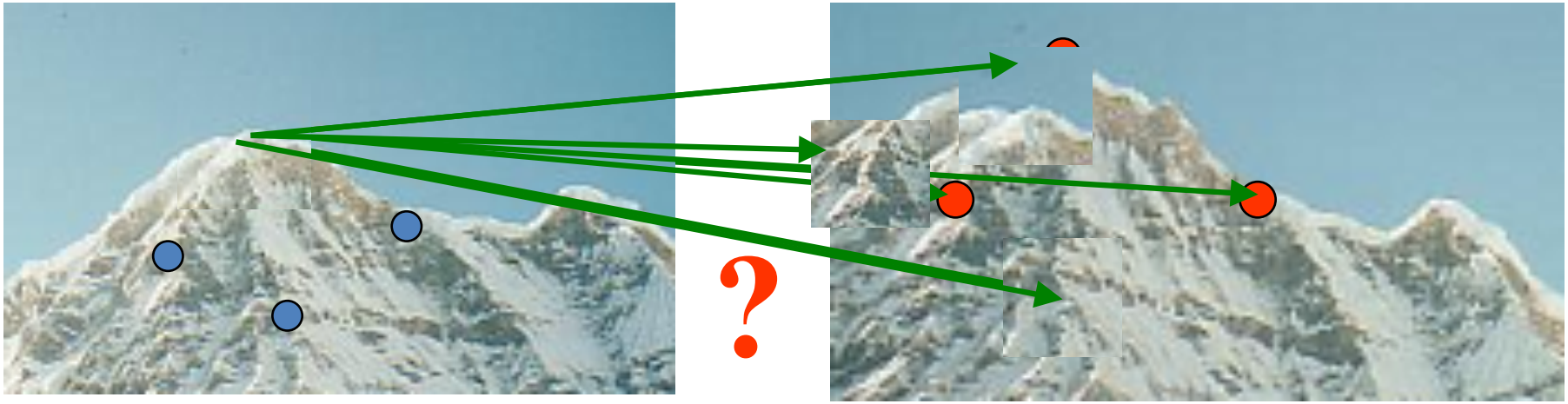
- We want to detect (at least some of) the same points in both images.

No chance to find true matches!

- Yet we have to be able to run the detection procedure *independently* per image.

Distinctiveness

- We want to be able to reliably determine which point goes with which.



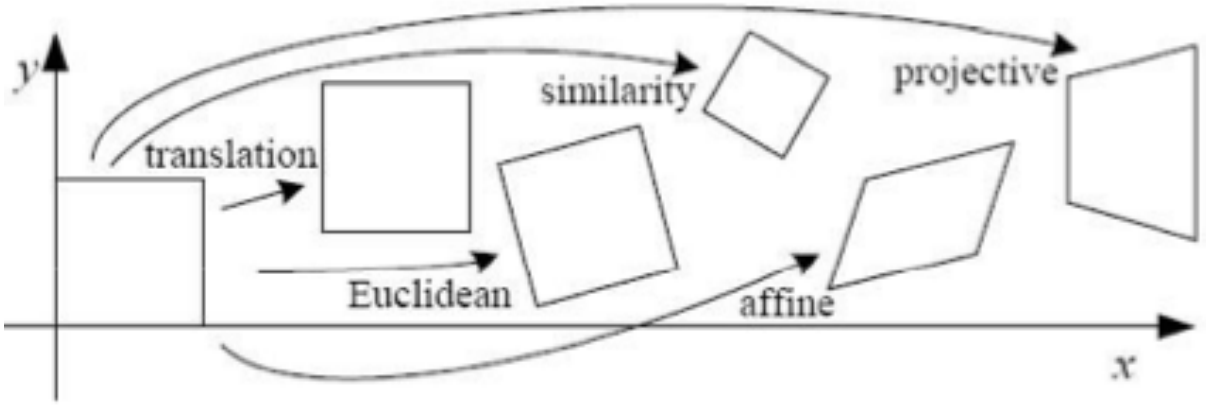
- Must provide some invariance to geometric and photometric differences between the two views.

Invariant local features

Geometric transformations

- Translation
- Euclidean (translation + rotation)
- Similarity (translation + rotation + scale)
- Affine transformations
- Projective transformations

Only holds for planar patches



Invariant local features

Photometric transformations

- Often modeled as a linear transformation:
  - Scaling + Offset

Requirements

- Region extraction needs to be **repeatable** and **accurate**
  - Invariant to translation, rotation, scale changes
  - Robust or covariant to out-of-plane (affine) transformations
  - Robust to lighting variations, noise, blur, quantization
- **Locality**: Features are local, therefore robust to occlusion and clutter.
- **Quantity**: We need a sufficient number of regions to cover the object.
- **Distinctiveness**: The regions should contain "interesting" structure.
- **Efficiency**: Close to real-time performance.

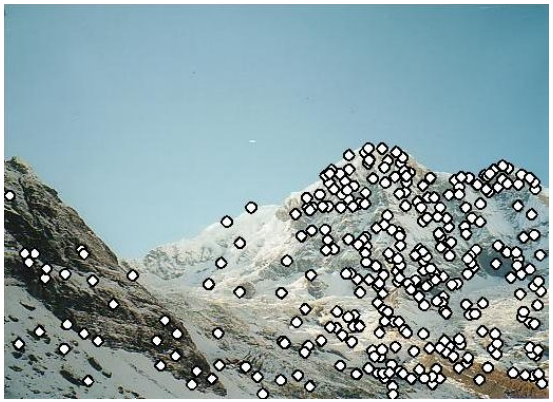
Many Existing Detectors Available

- Hessian & Harris [Beaudet ‘78], [Harris ‘88]
- Laplacian, DoG [Lindeberg ‘98], [Lowe ‘99]
- Harris-/Hessian-Laplace [Mikolajczyk & Schmid ‘01]
- Harris-/Hessian-Affine [Mikolajczyk & Schmid ‘04]
- EBR and IBR [Tuytelaars & Van Gool ‘04]
- MSER [Matas ‘02]
- Salient Regions [Kadir & Brady ‘01]
- Others…

Those detectors have become a basic building block for many recent applications in Computer Vision.

Local features: main components

1) Detection: Identify the interest points

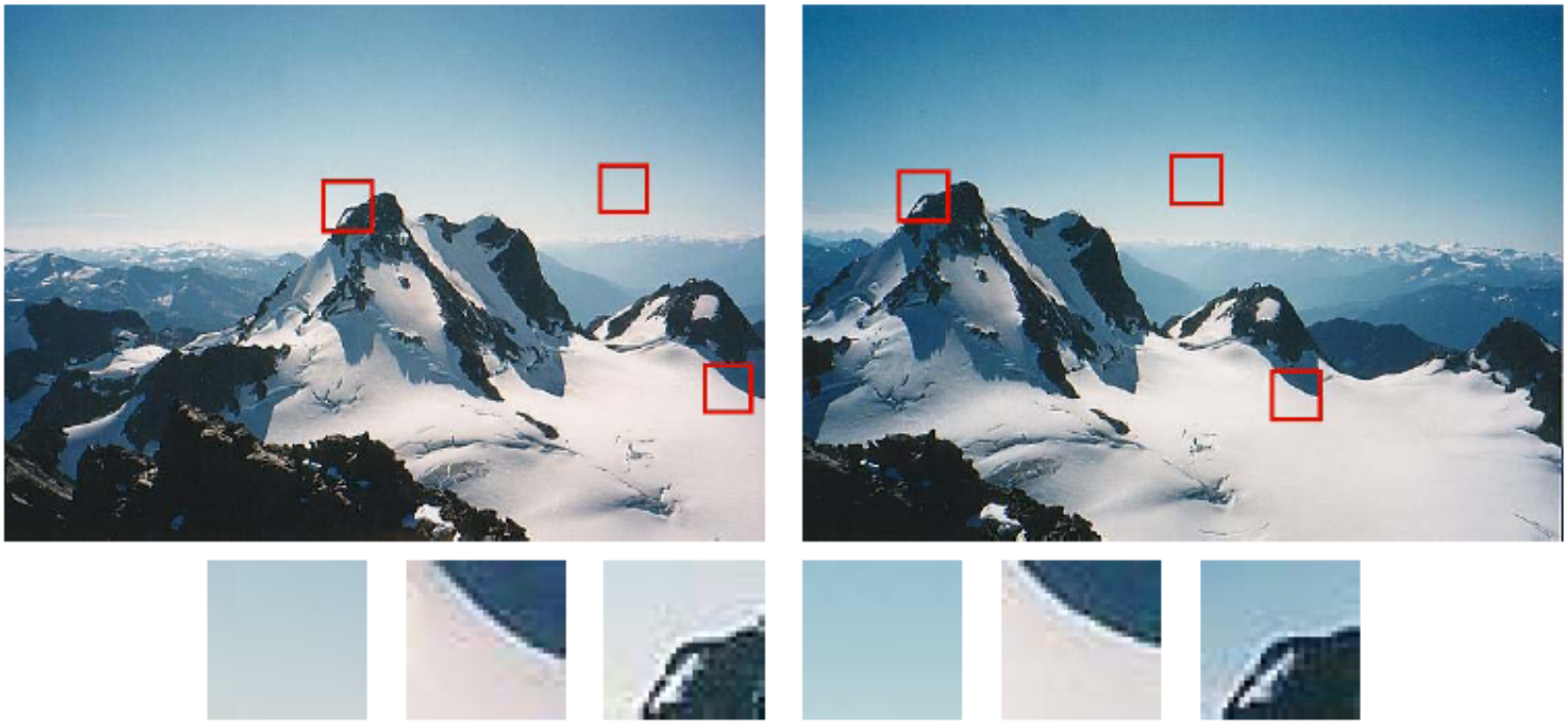


2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

Local features detectors

- What are salient features that can be *detected* in multiple views?

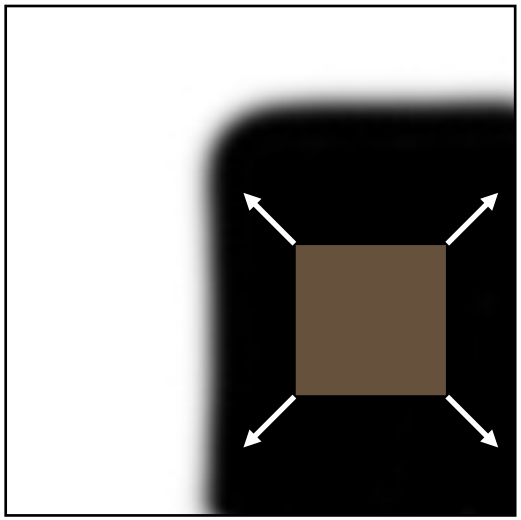


Local features detectors

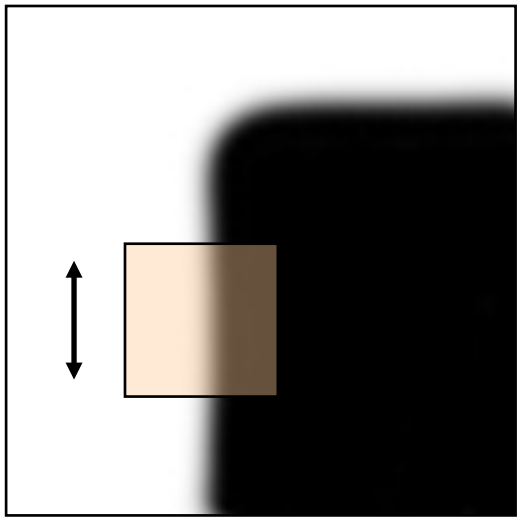
- What points would be better?

Corners

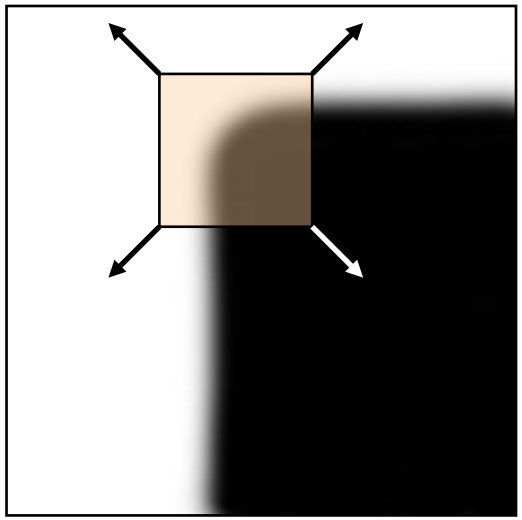
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"**flat**" region:
no change in
all directions



"**edge**":
no change
along the edge
direction



"**corner**":
significant
change in all
directions

Harris Detector Formulation

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

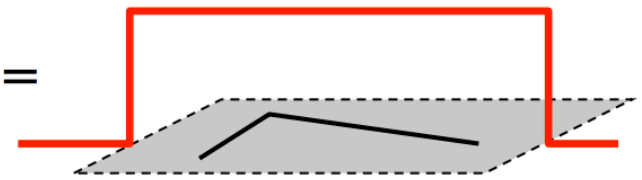
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

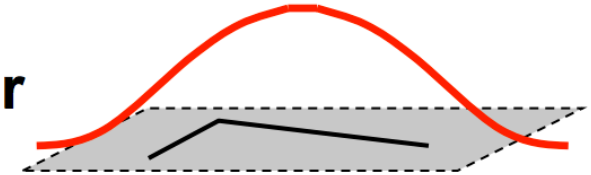
Intensity

Window function $w(x,y) =$



1 in window, 0 outside

or



Gaussian

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

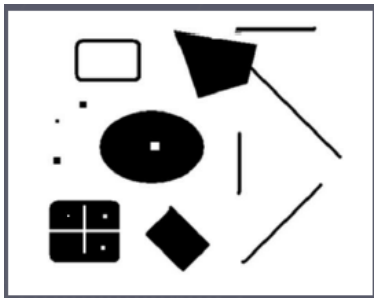
Sum over image region – area we are checking for corner

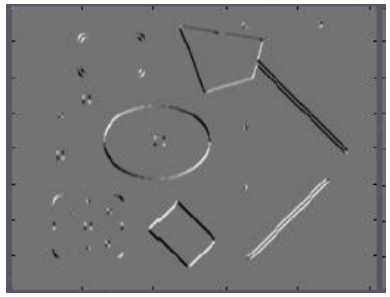
$$\boldsymbol{M} = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y]$$

Harris Detector Formulation

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

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).



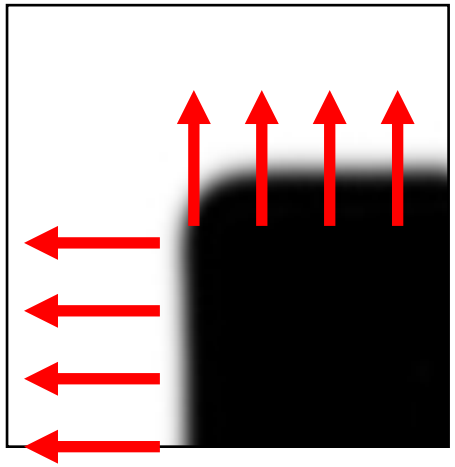


Notation:

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

What does this matrix reveal?

First, consider an axis-aligned corner:



What does this matrix reveal?

First, consider an axis-aligned corner:

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

This means dominant gradient directions align with x or y axis

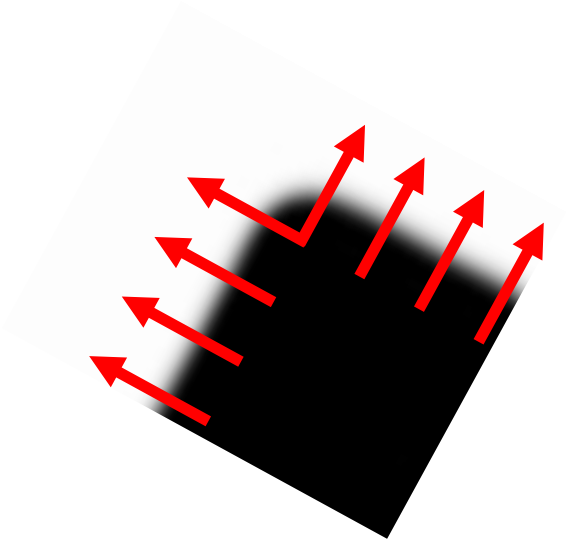
Look for locations where both λ's are large.

If either λ is close to 0, then this is not corner-like.

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

What does this matrix reveal?

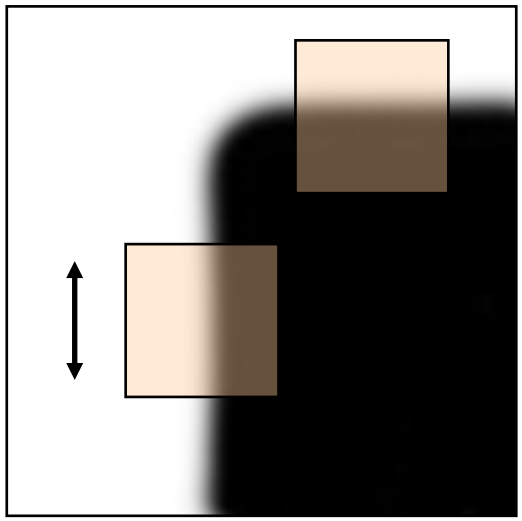
Since M is symmetric, we have $M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$



$$Mx_i = \lambda_i x_i$$

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

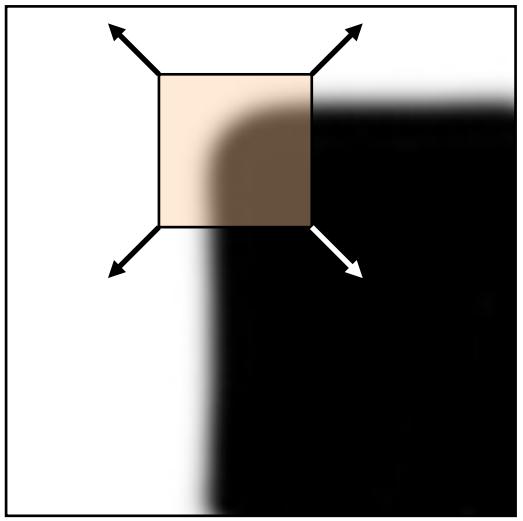
Corner response function



"edge":

$\lambda_1 >> \lambda_2$

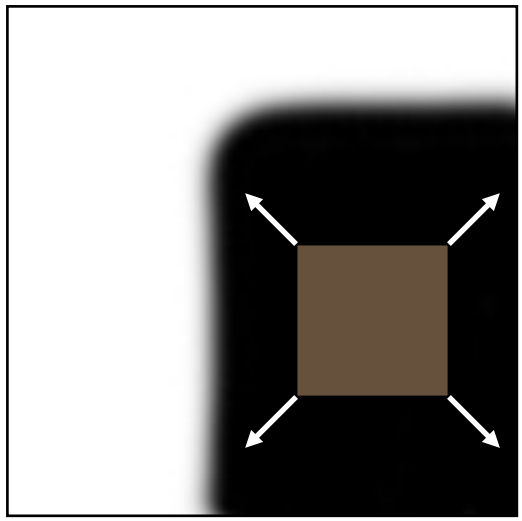
$\lambda_2 >> \lambda_1$



"corner":

λ_1 and λ_2 are large,

$\lambda_1 \sim \lambda_2$;



"flat" region

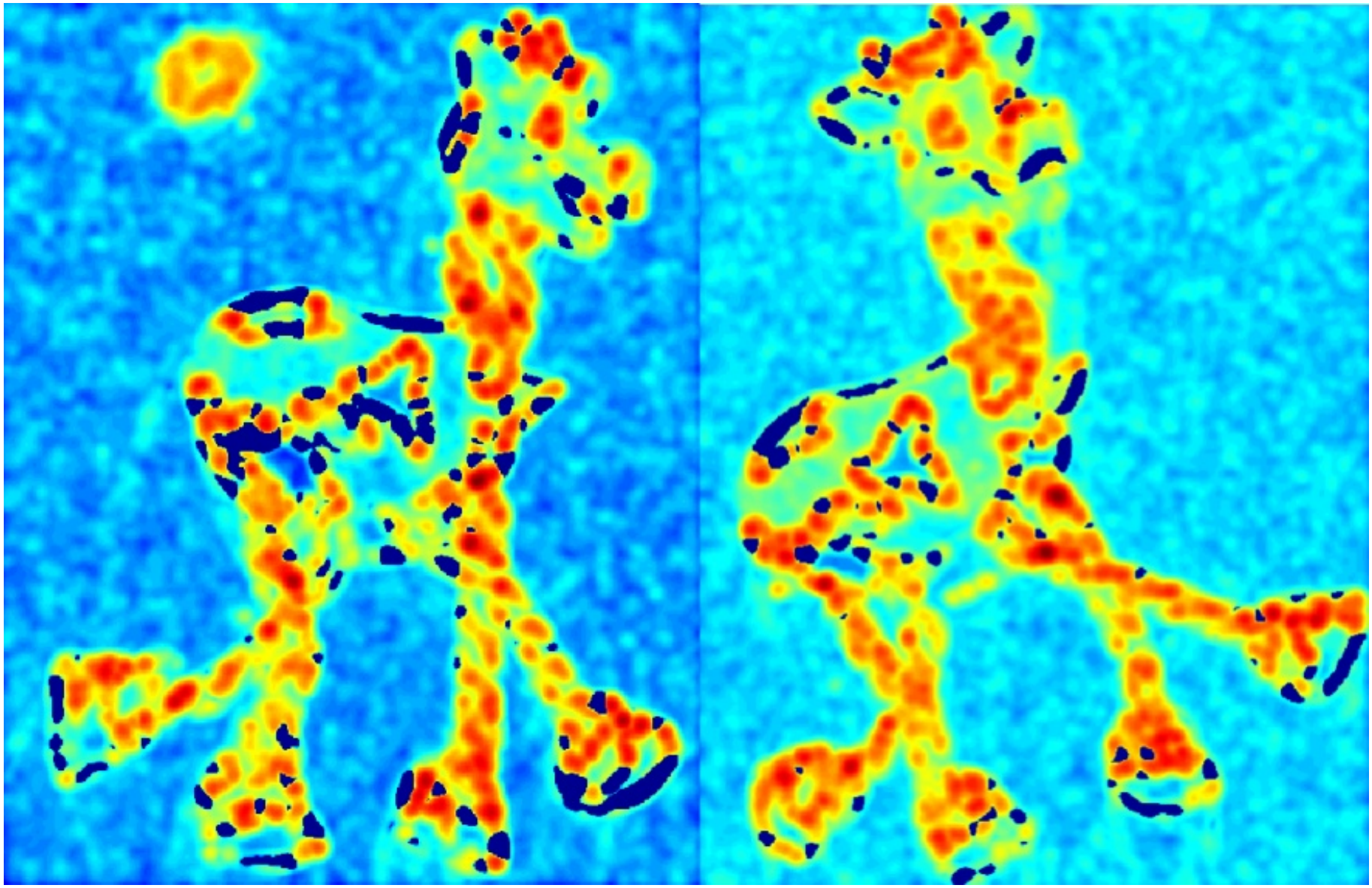
λ_1 and λ_2 are small;

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

Harris corner detector

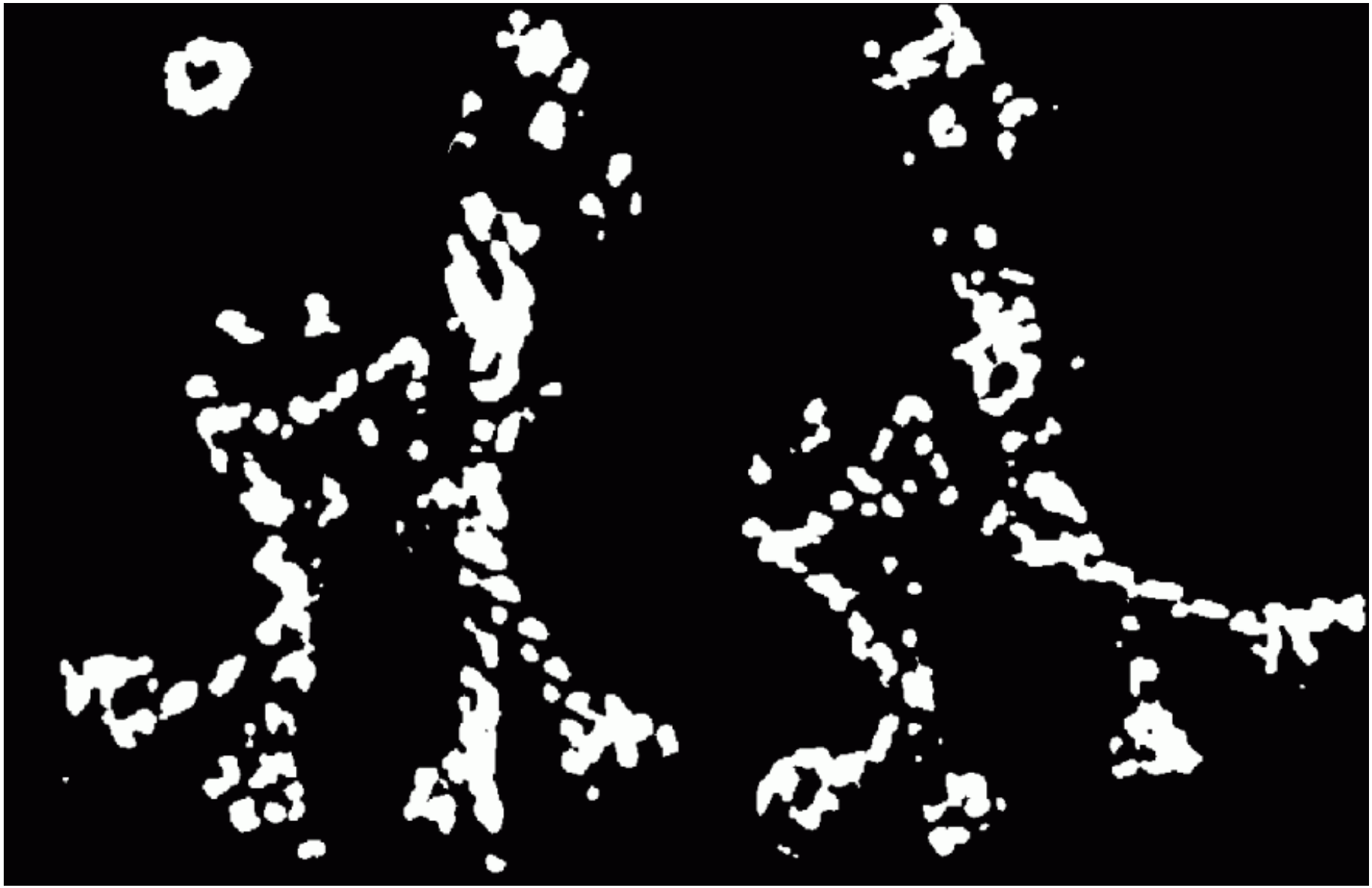
Harris corner detector

Compute corner response *f*



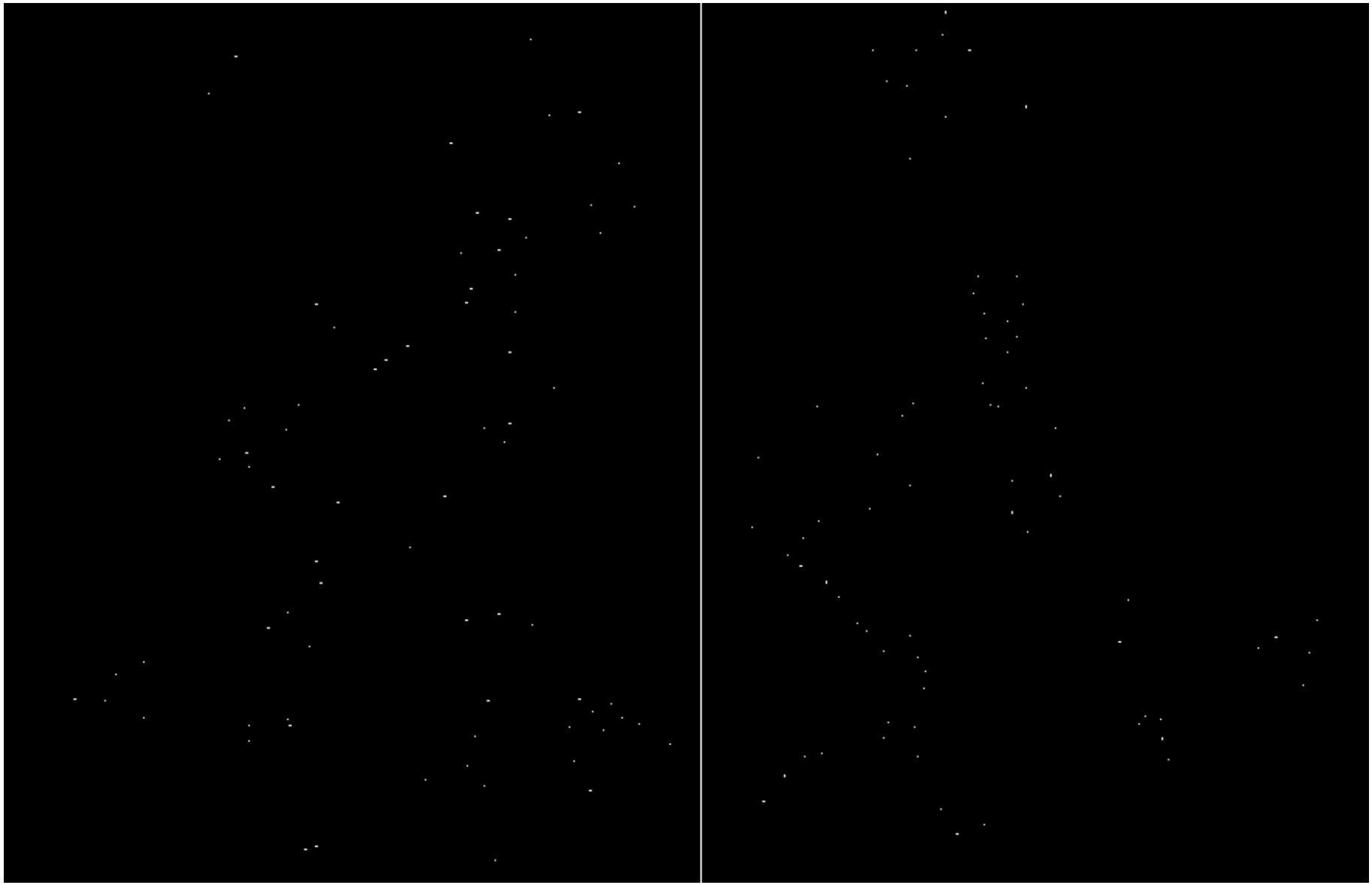
Harris corner detector

Find points with large corner response: f > threshold

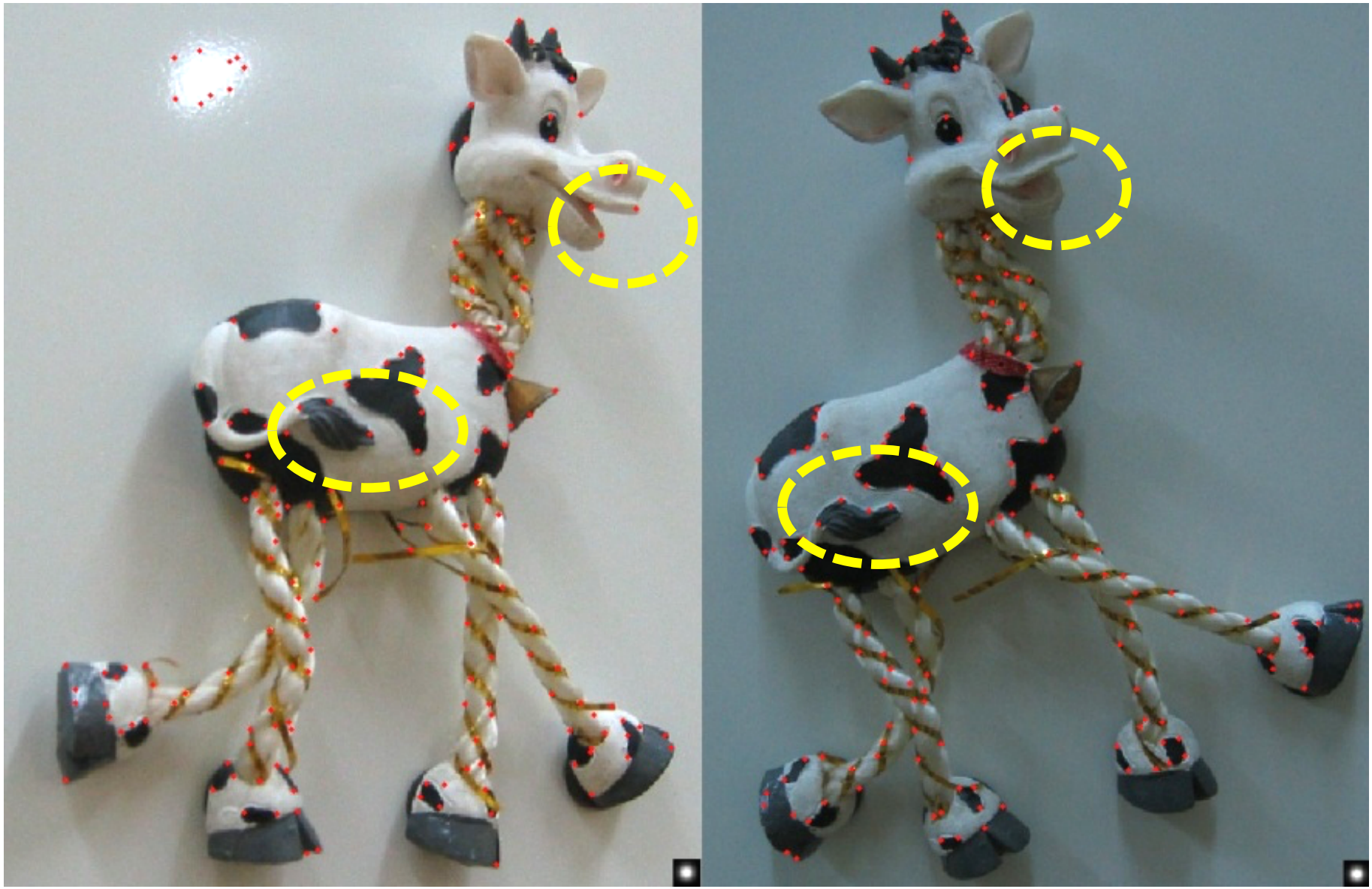


Harris corner detector

Take only the points of local maxima of *f*



Harris corner detector



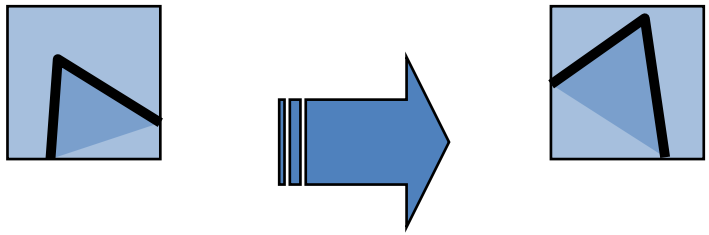
Harris corner detector

Harris corner detector

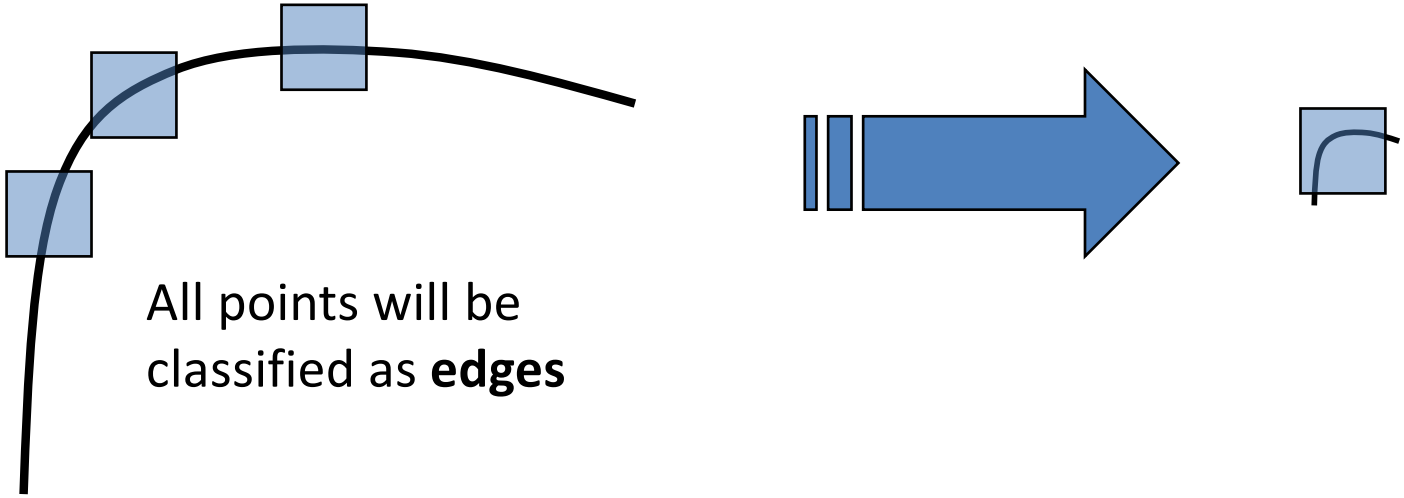
Harris corner detector

- Properties of the Harris corner detector
  - Rotation invariance

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



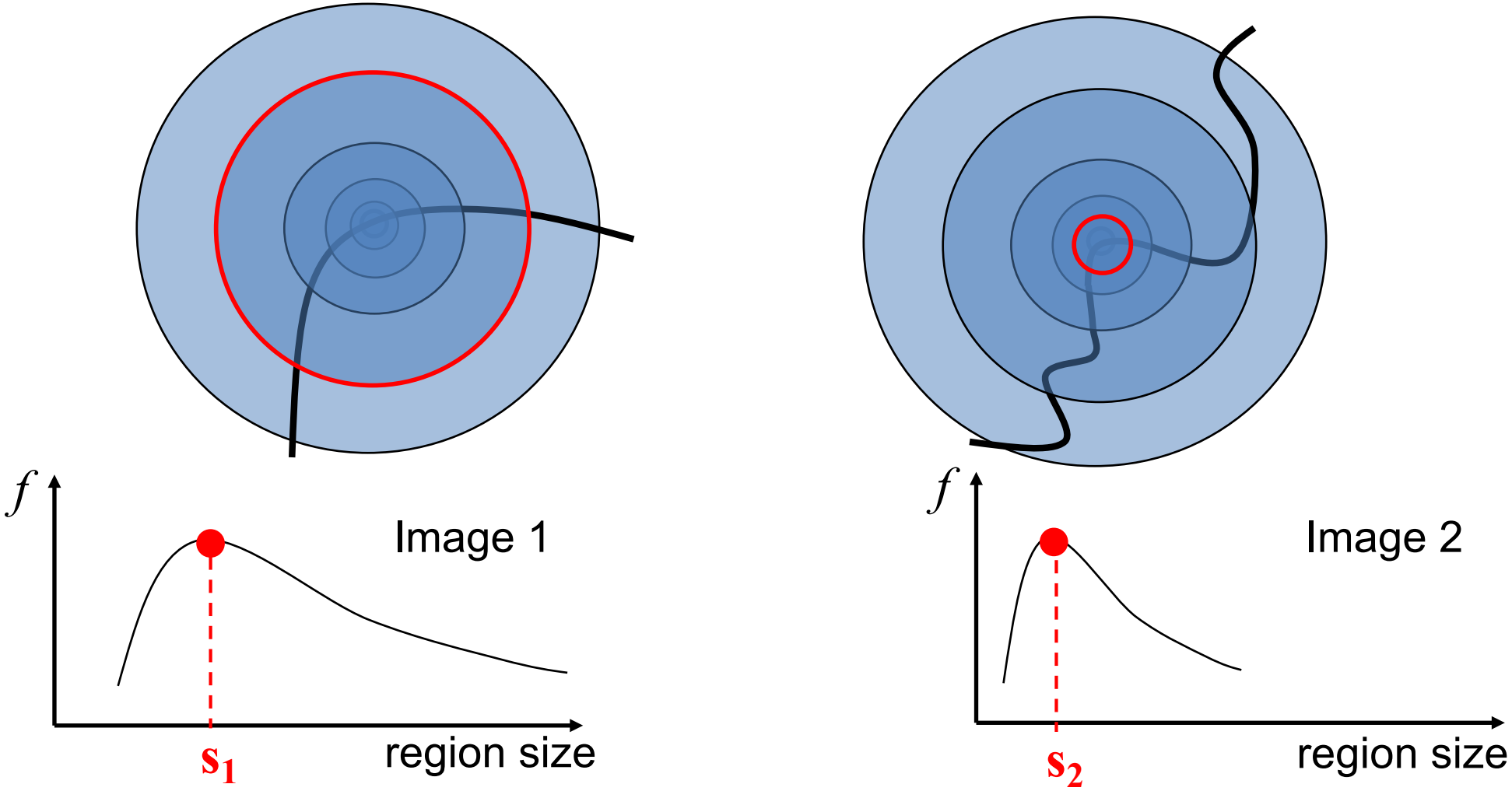
  - Not invariant to image scale



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

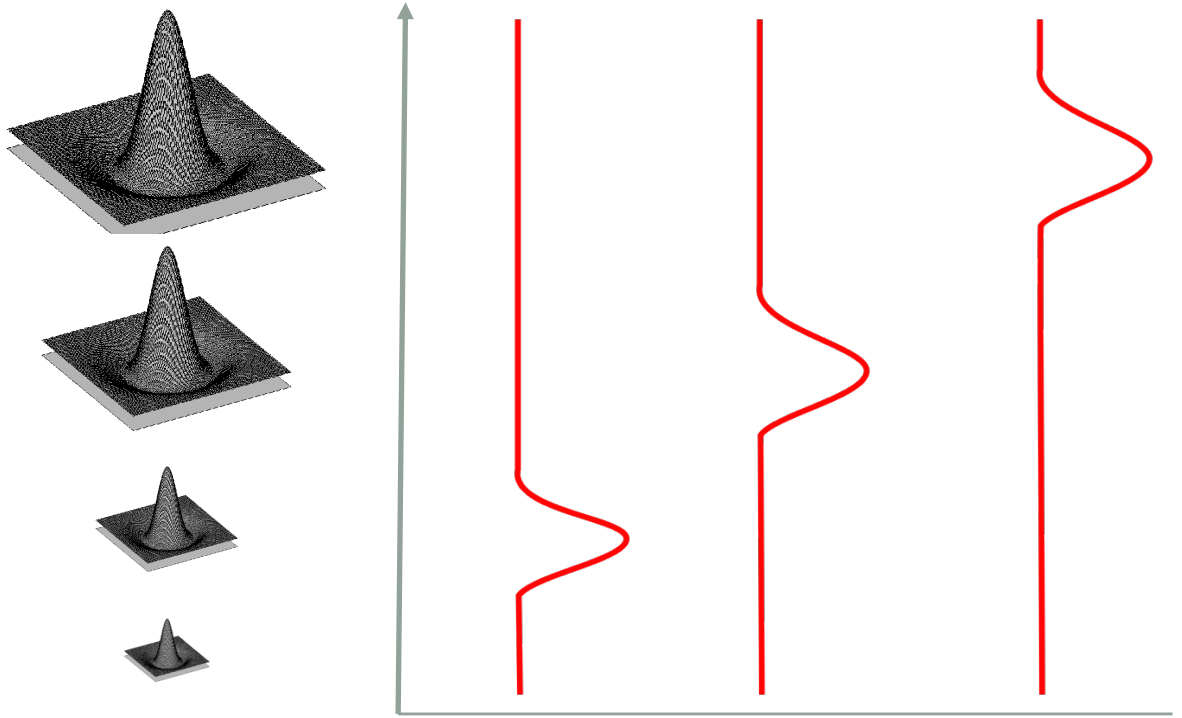
Automatic scale selection

Intuition - Find scale that gives local maxima of some signature function f in both position and scale.



Scale invariance detection

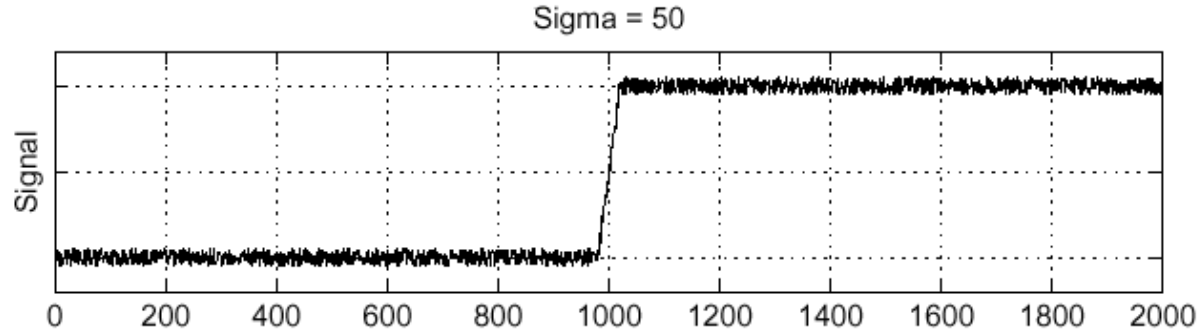
- Useful signature function
  - Laplacian-of-Gaussian = “blob” detector



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

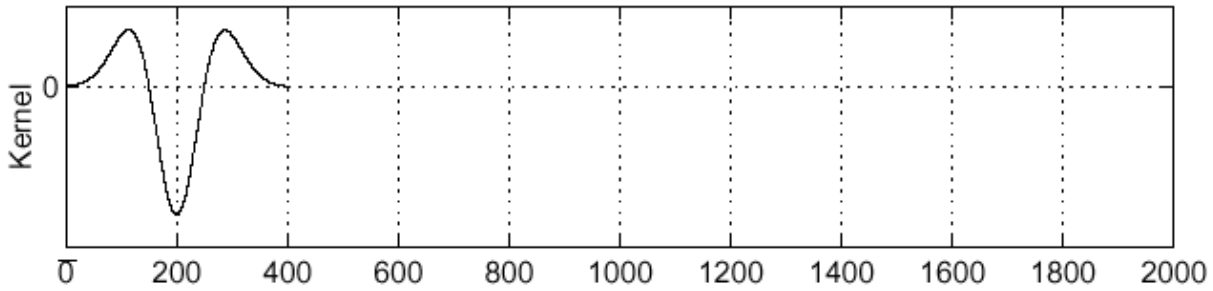
Scale invariance detection

f



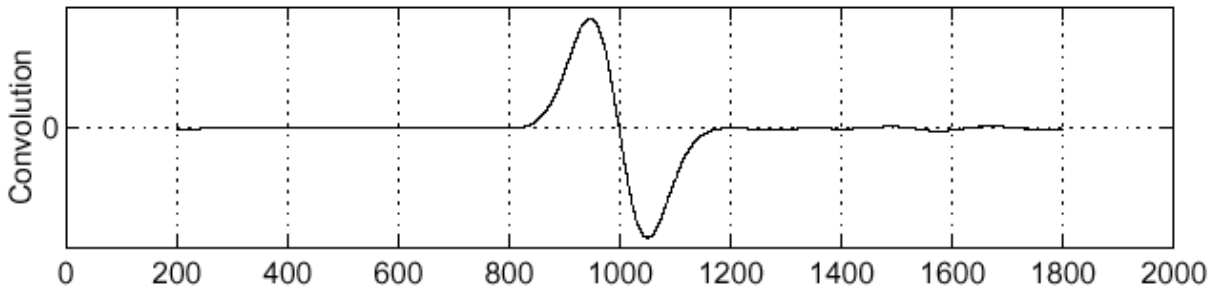
Edge

$\frac{d^2}{dx^2} g$



Second derivative of Gaussian (Laplacian)

$f * \frac{d^2}{dx^2} g$

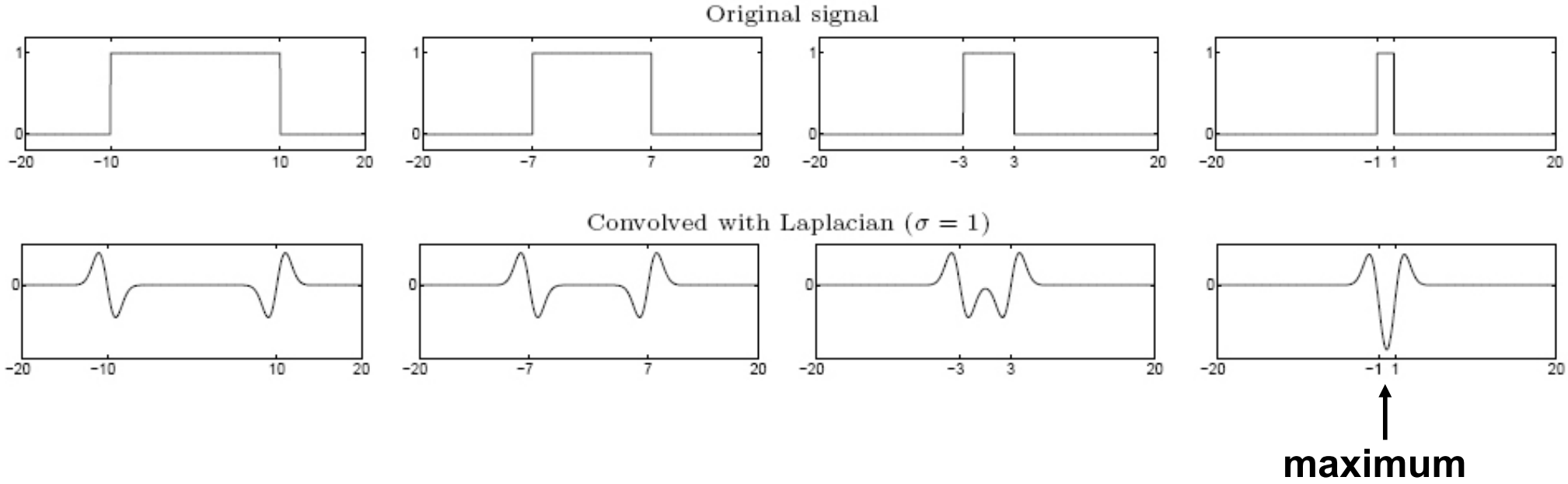


Edge = zero crossing of second derivative

Source: S. Seitz

Scale invariance detection

- Edge = ripple
- Blob = superposition of two ripples

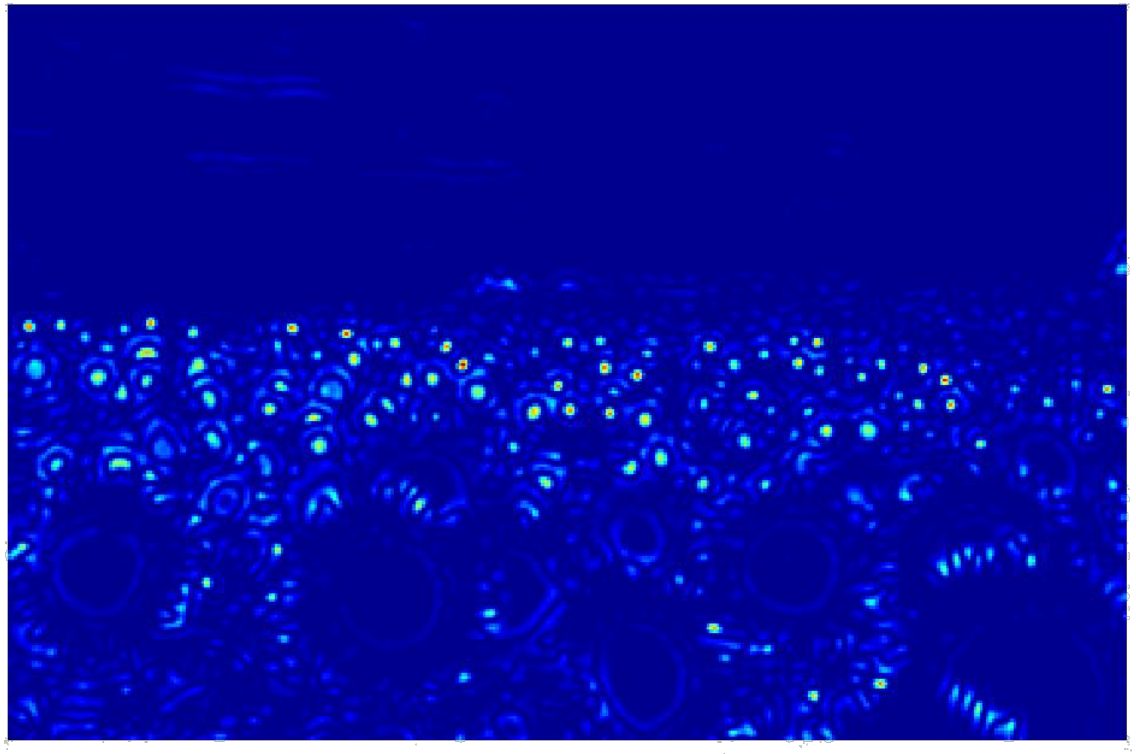


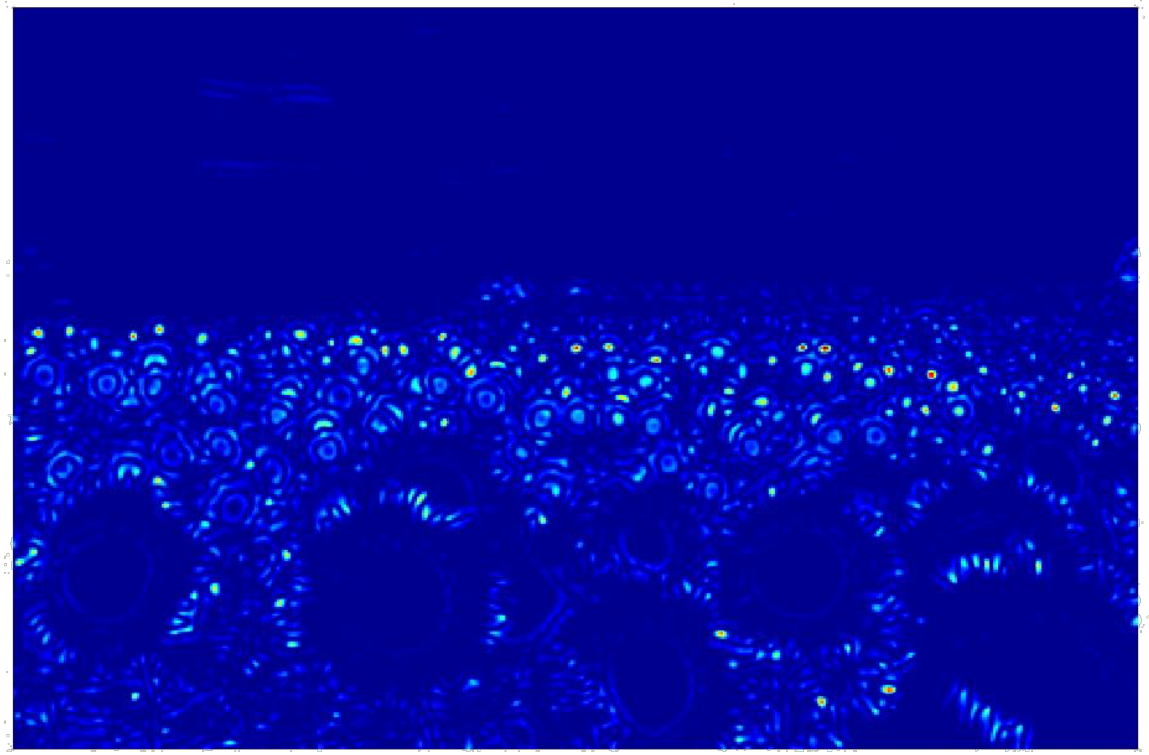
Spatial selection: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is "matched" to the scale of the blob

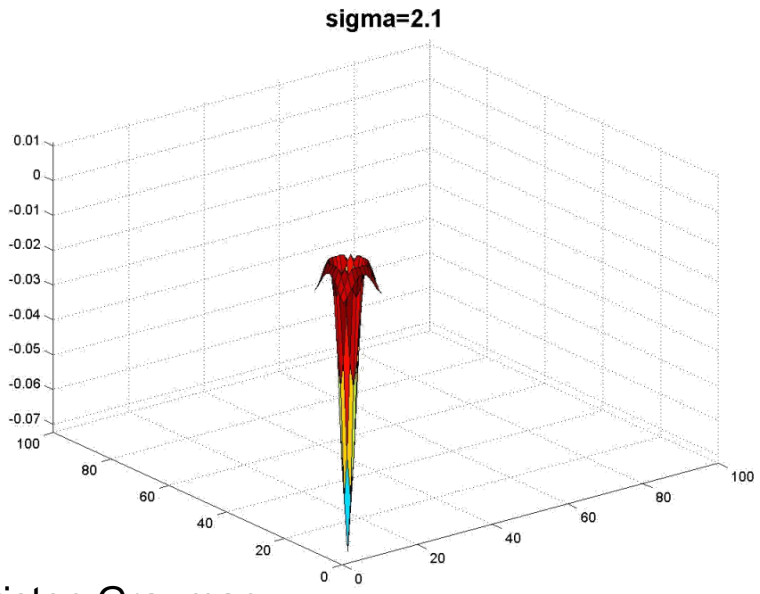
Lana Lazebnik

Scaled image
at ¾ the size

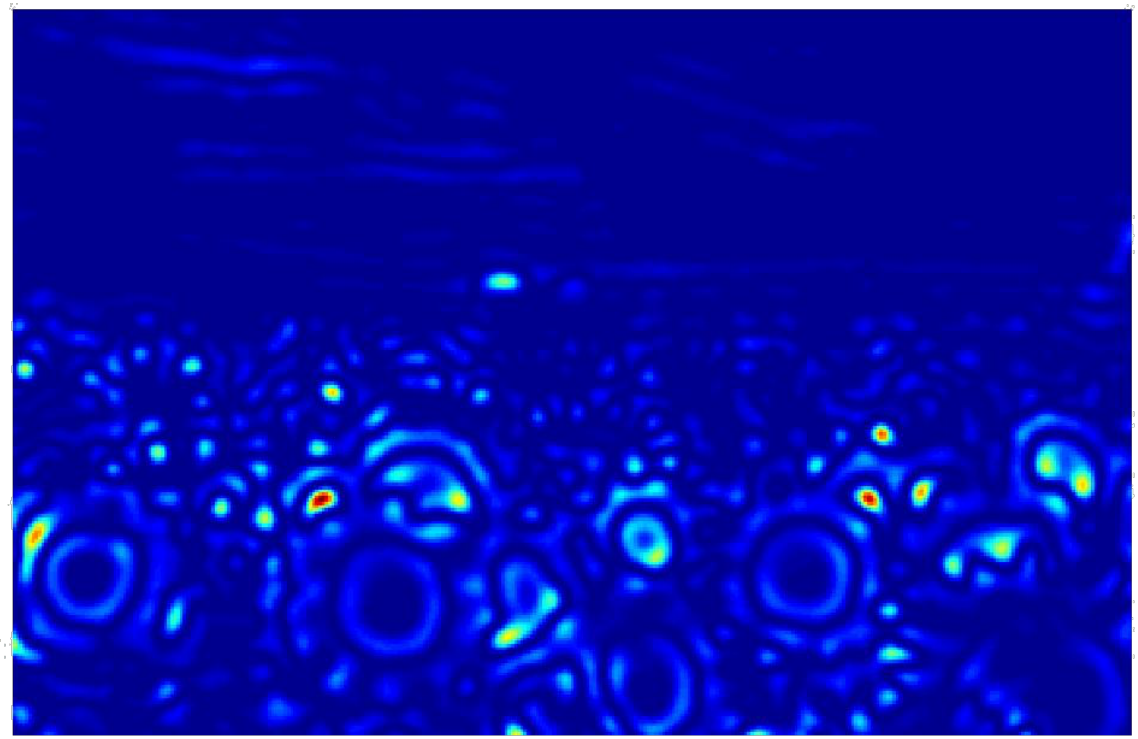
Original image

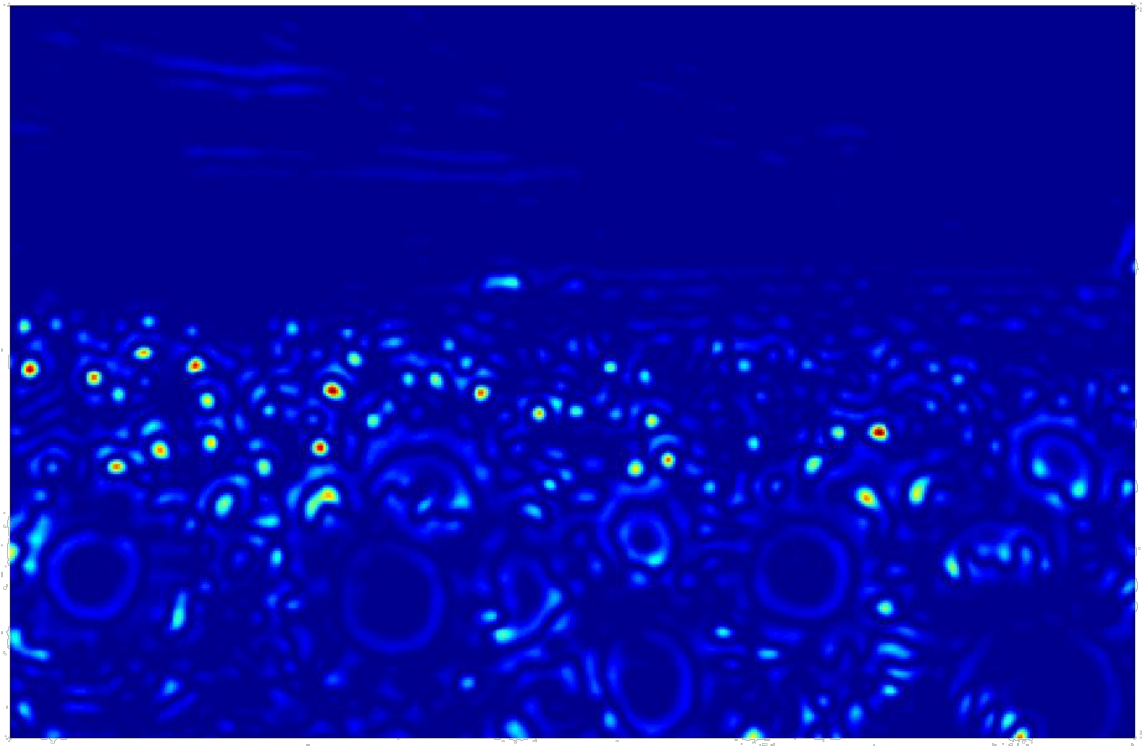


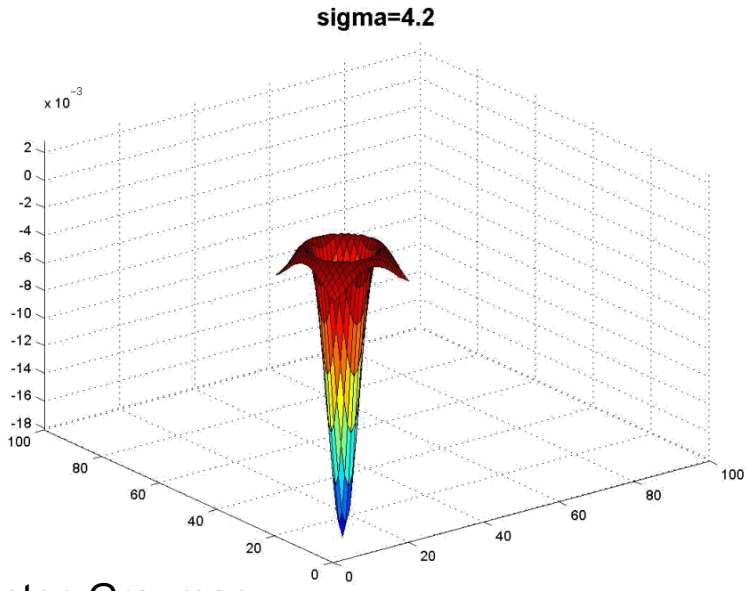




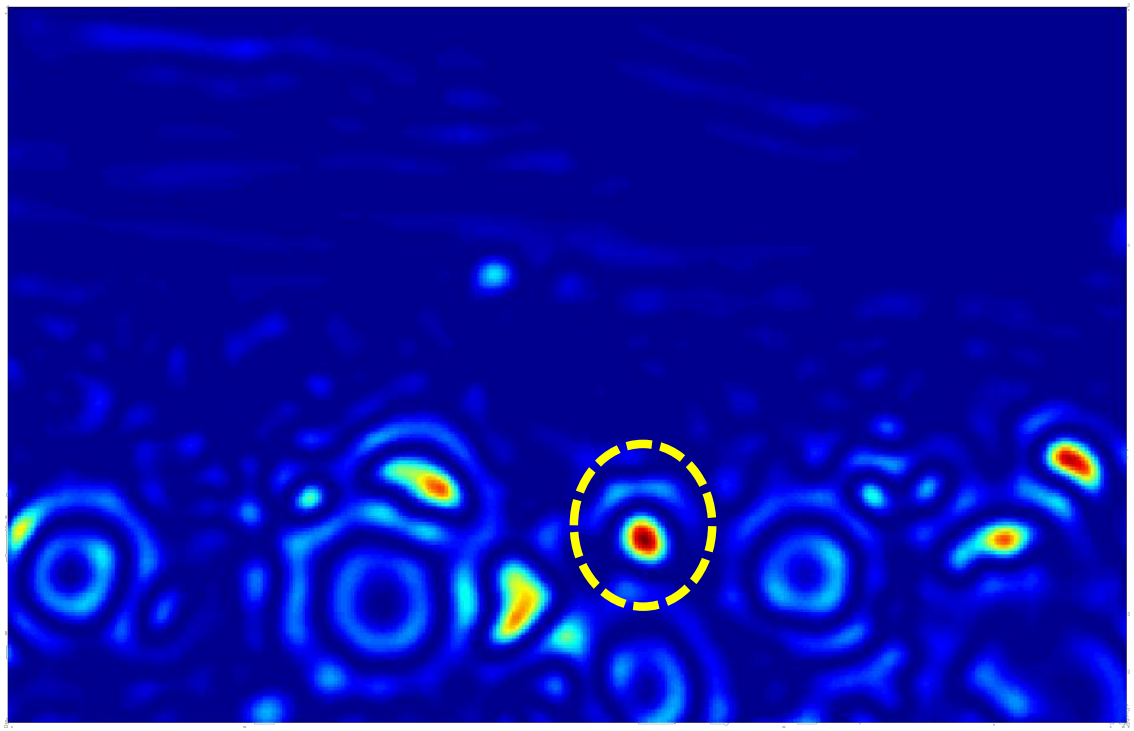
Kristen Grauman

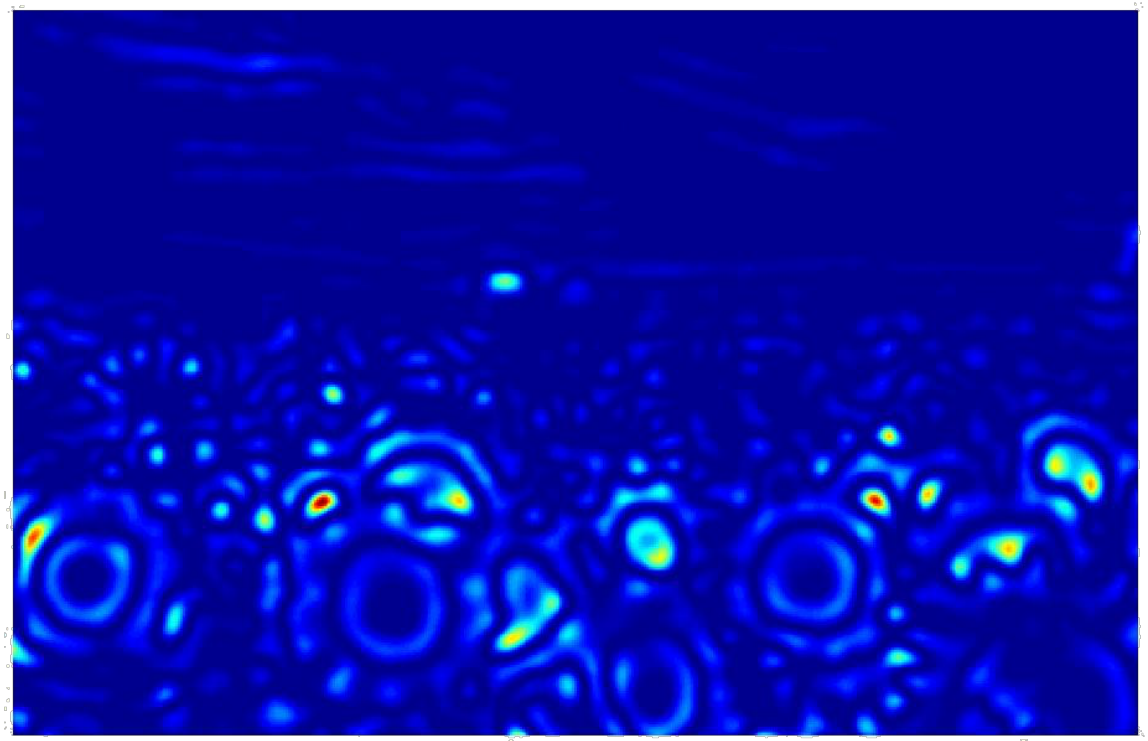


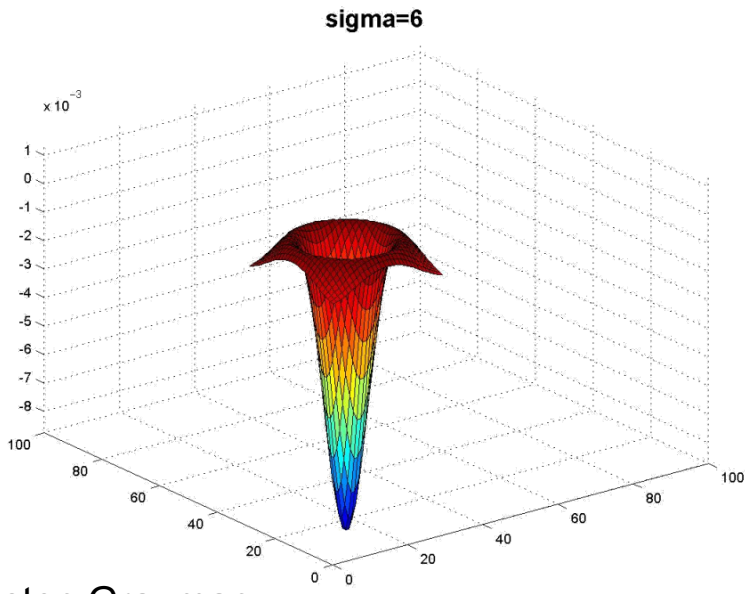




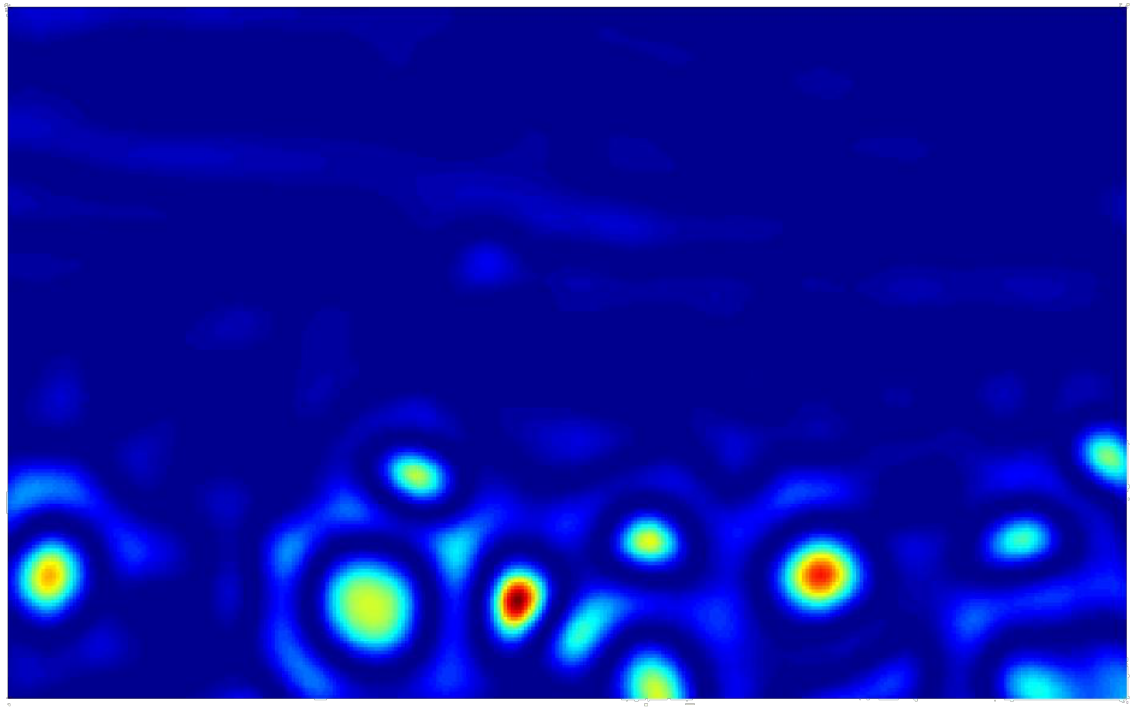
Kristen Grauman

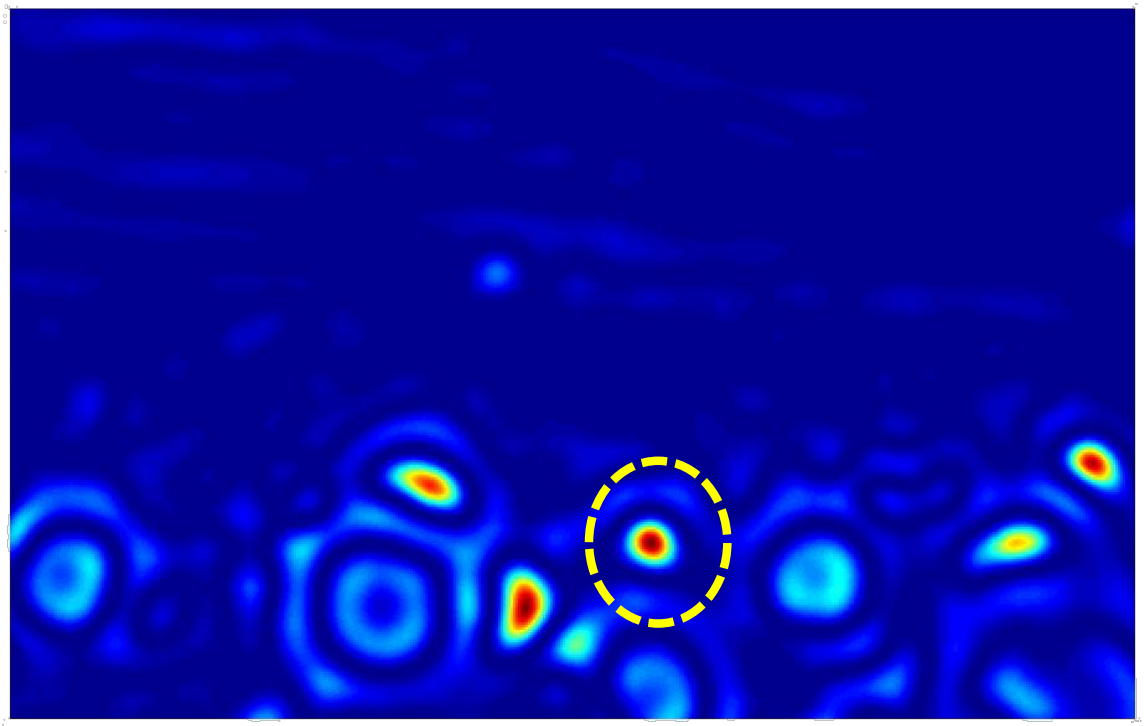


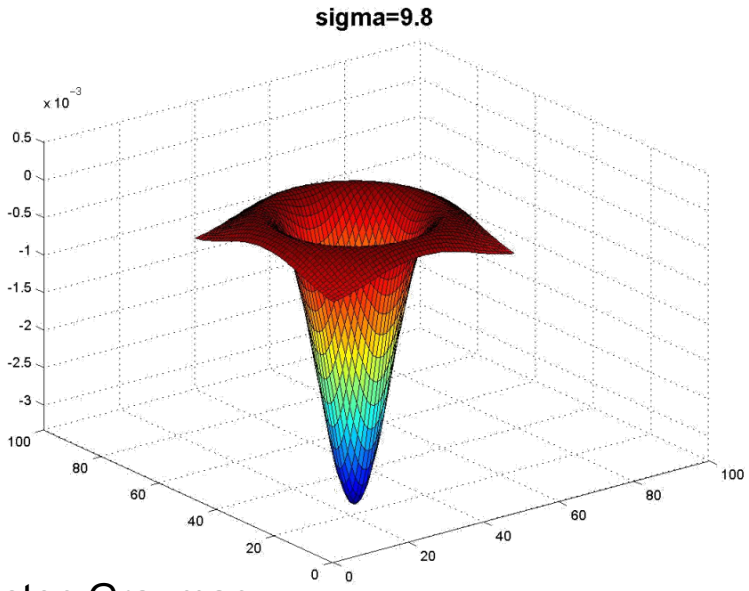




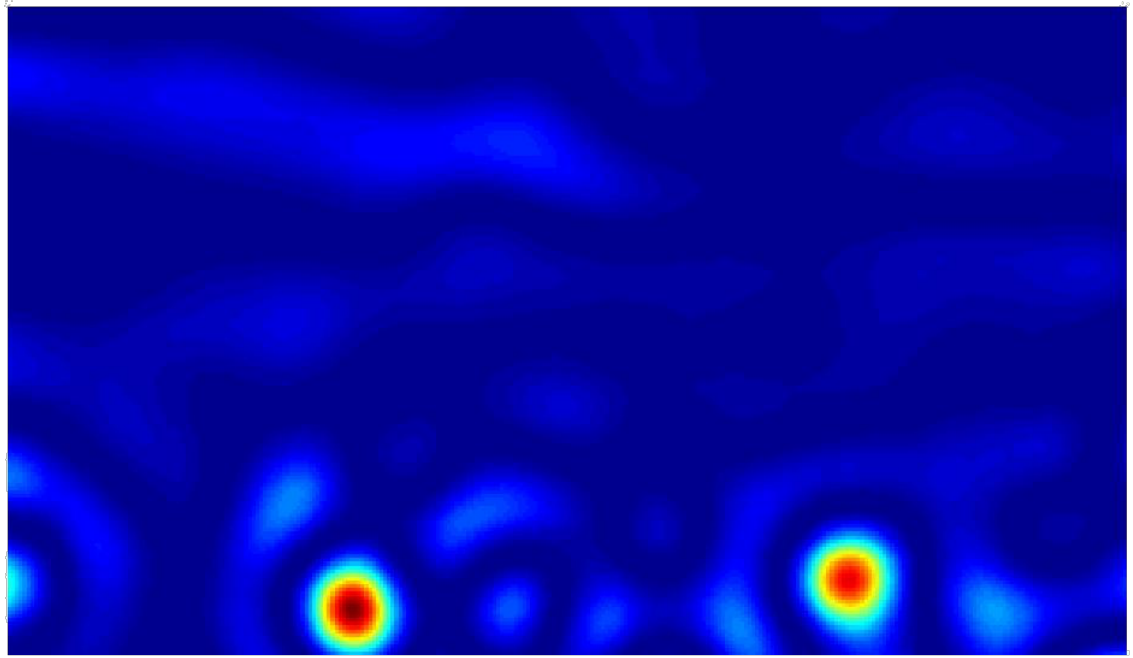
Kristen Grauman

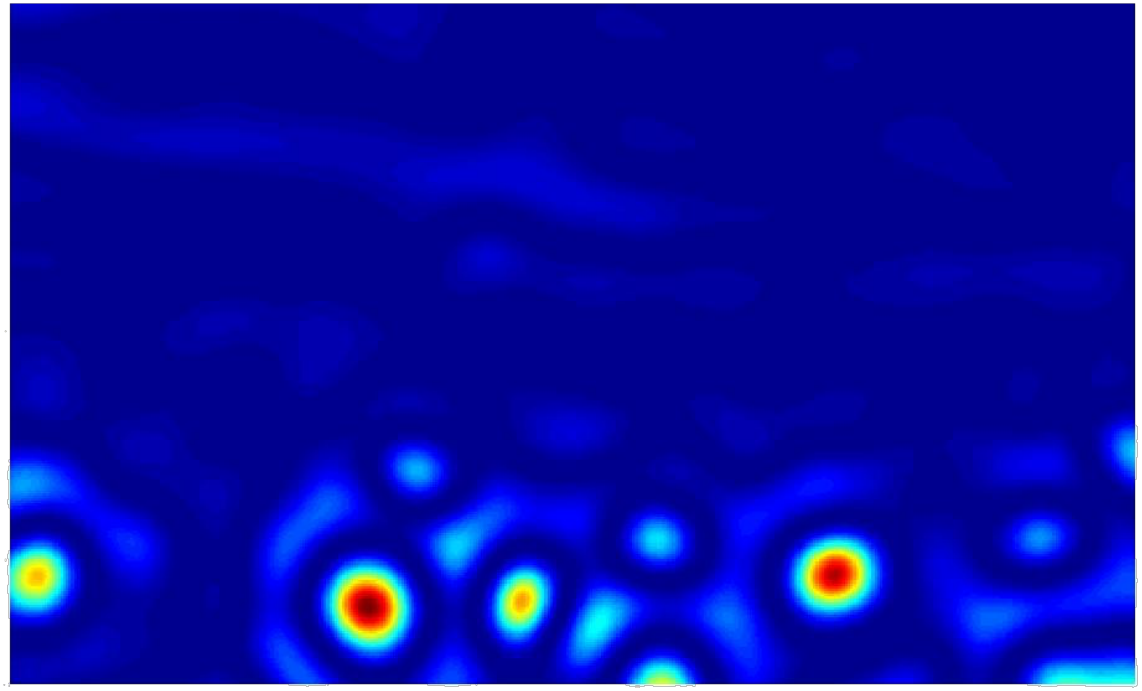


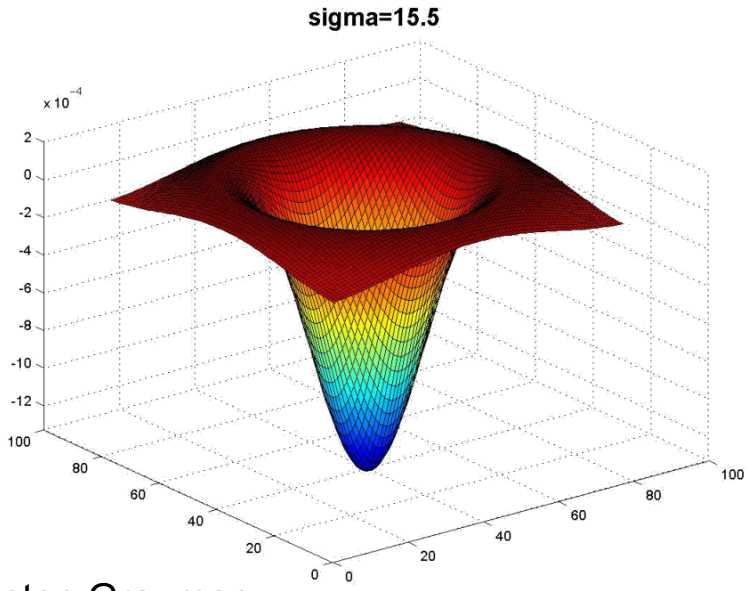


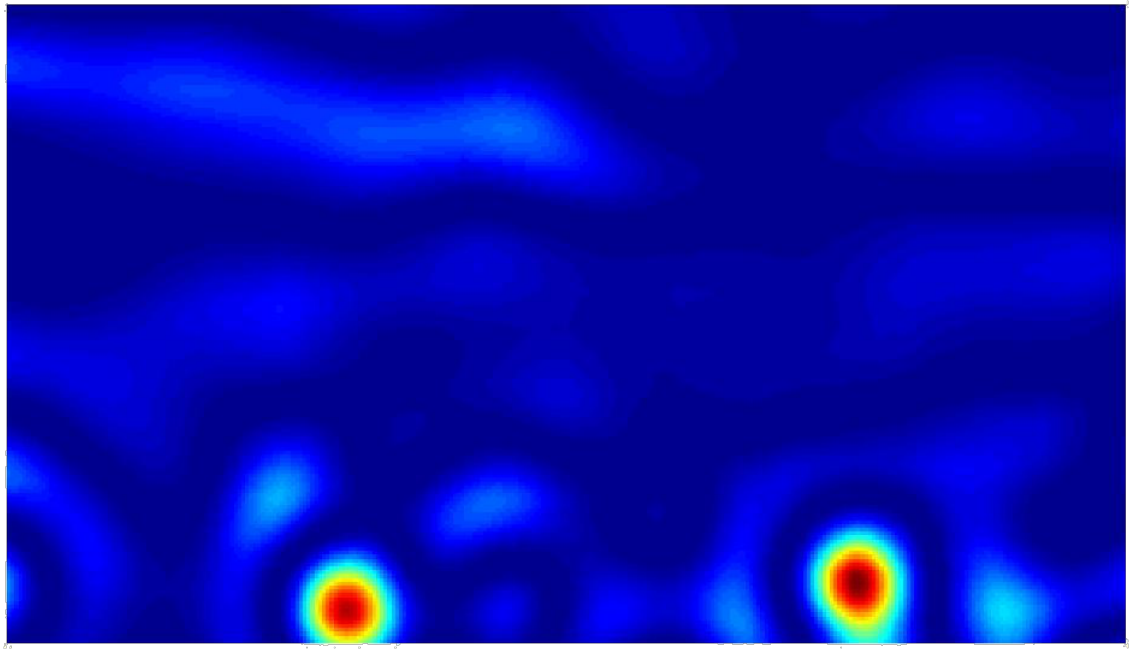


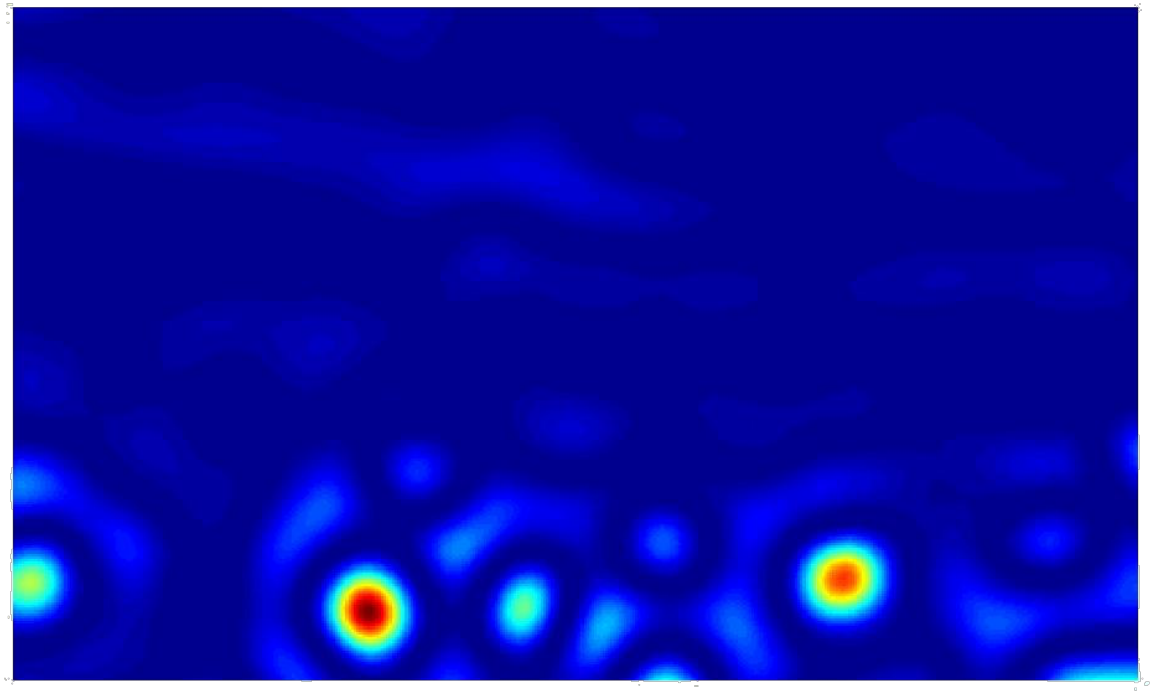
Kristen Grauman

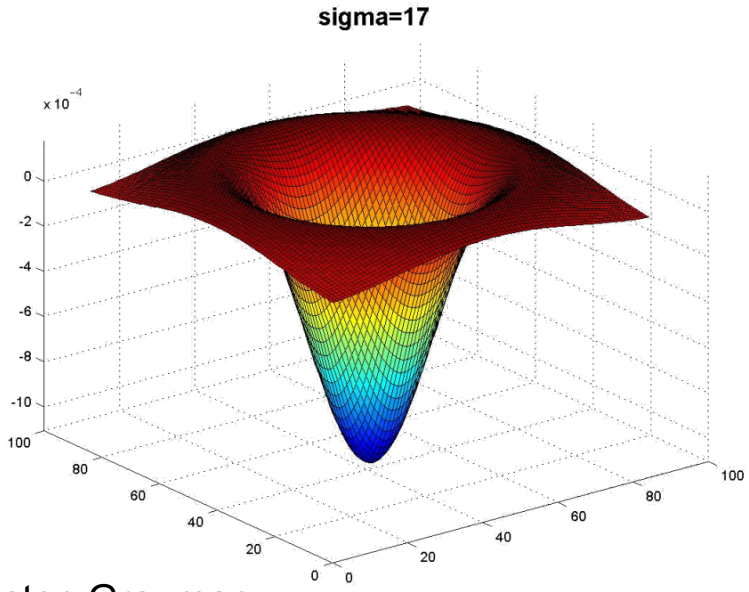





sigma=15.5
x 10^-4
2
0
-2
-4
-6
-8
-10
-12
100
80
60
40
20
0
0
20
40
60
80
100







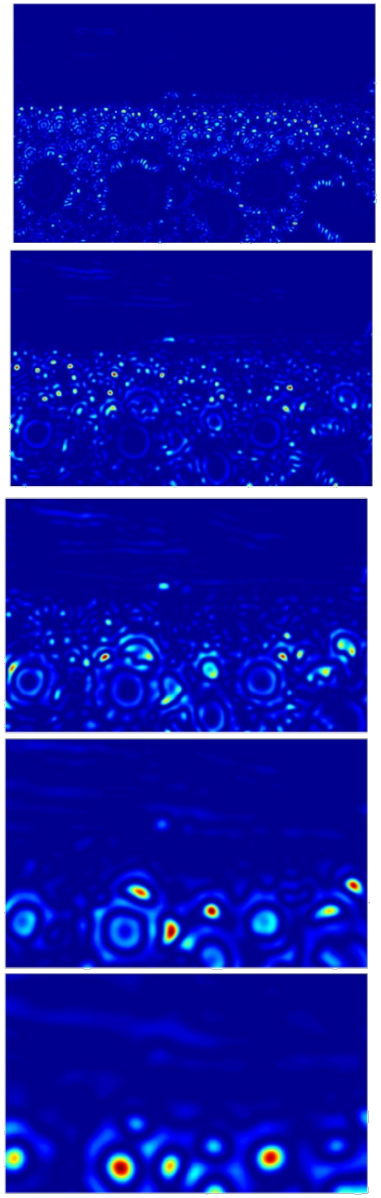
Kristen Grauman

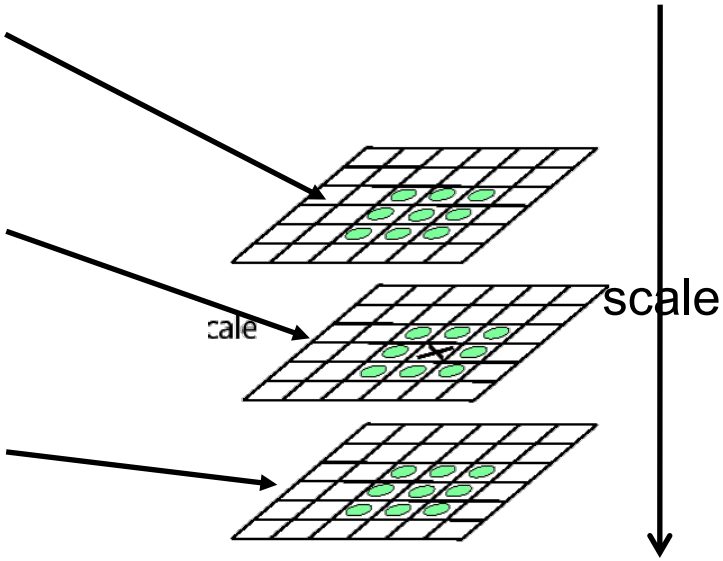
Scale invariant interest points

Interest points are local maxima in both position and scale.

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

σ5
σ4
σ3
σ2
σ1





⇒ **List of**
(x, y, σ)

Squared filter response maps

Scale invariant interest points

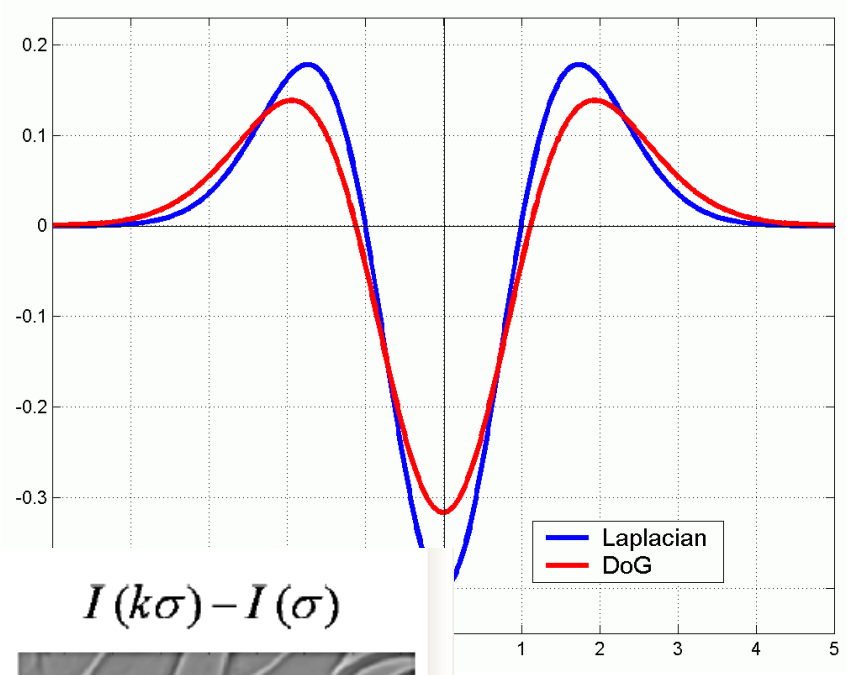
- We can approximate the Laplacian with a difference of Gaussians, which is more efficient to implement.

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

(Laplacian)

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

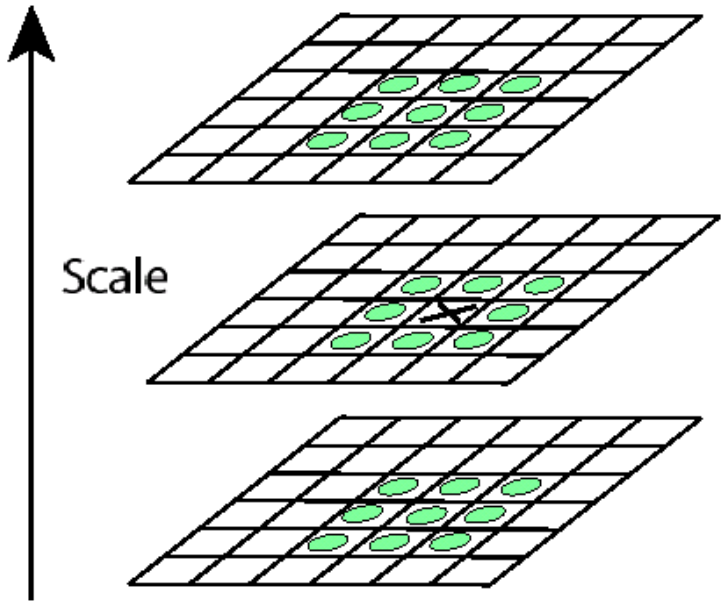
(Difference of Gaussians)



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

Scale invariance detection

- LoG can be approximated by a Difference of two Gaussians (DoG) at different scales
- Detect maxima of DoG in the scale space volume
- Reject points with low contrast (threshold)
- Reject points that are localized along an edge



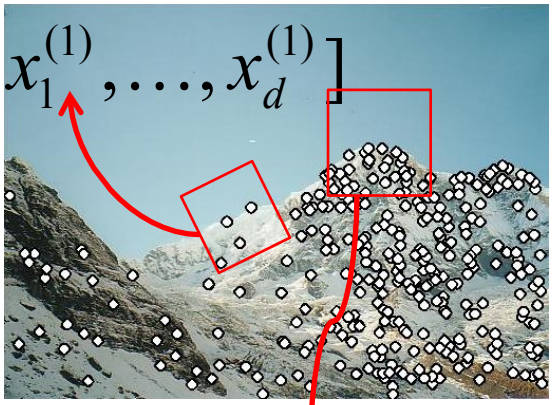
Candidate keypoints:
list of (x,y,σ)

Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

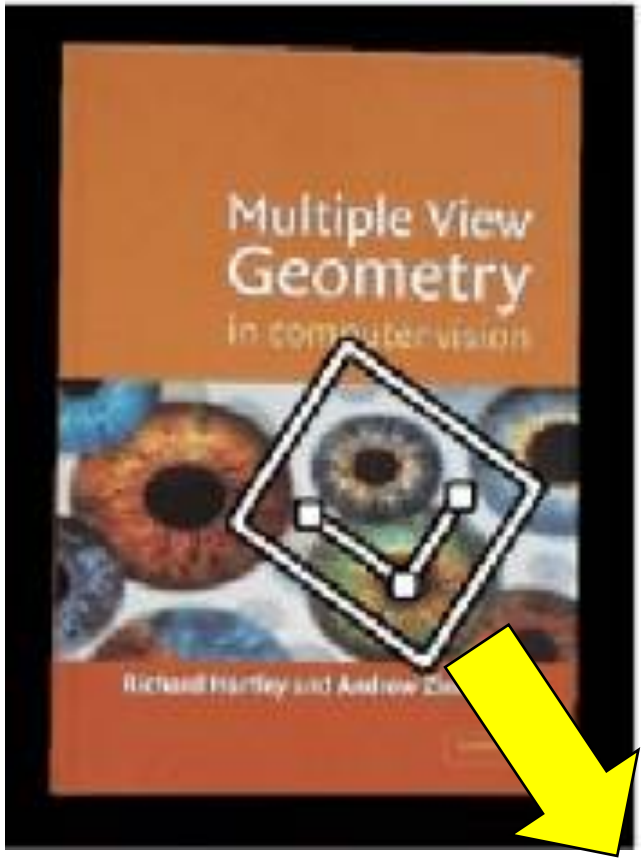
$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

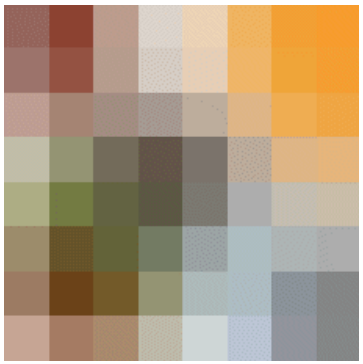


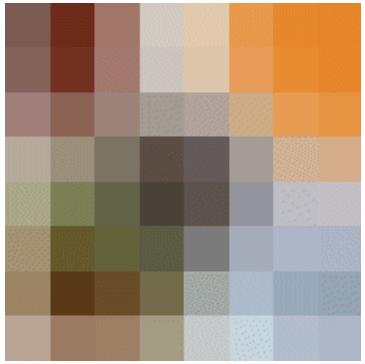
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) Matching: Determine correspondence between descriptors in two views

Geometric transformation



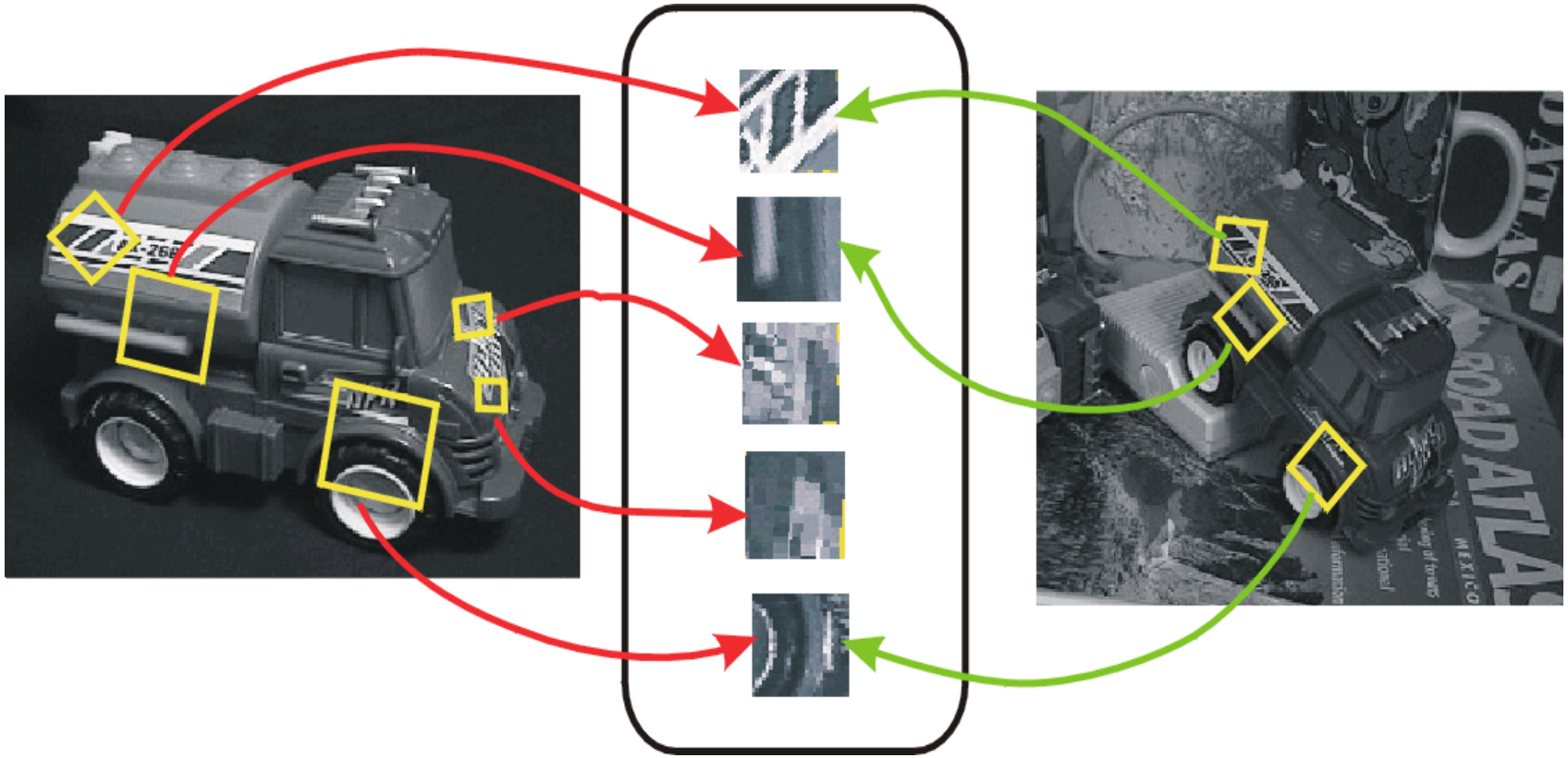




Invariant local features

Find features that are invariant to transformations

- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, …



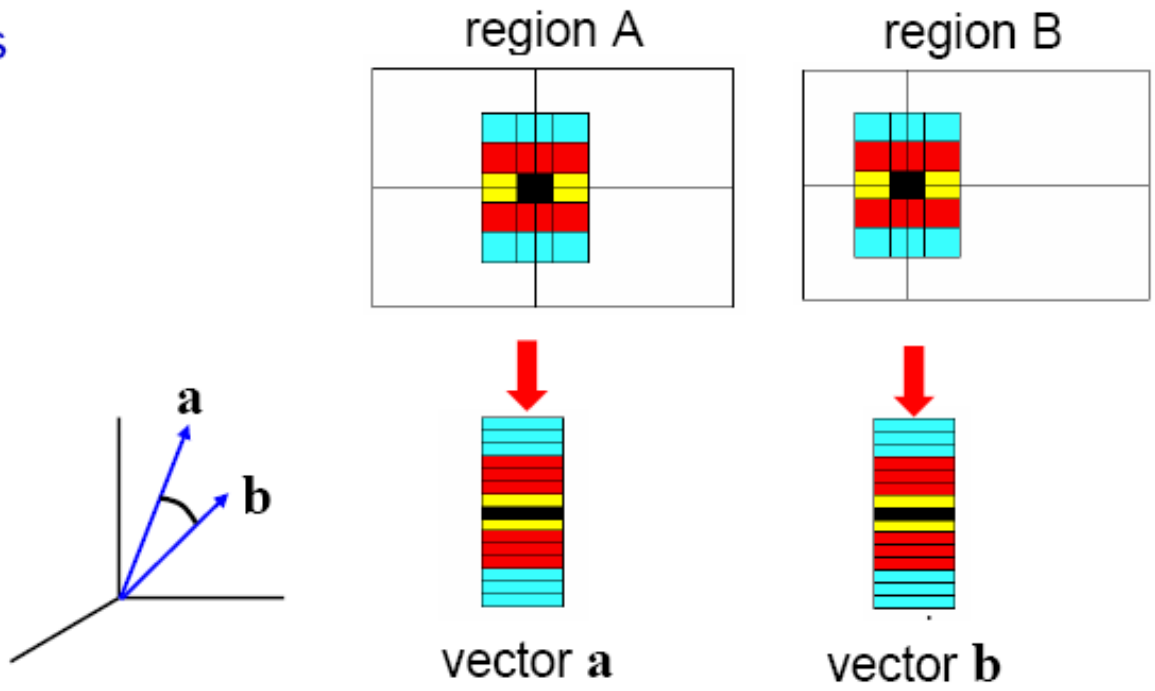
Feature Descriptors

Local descriptors

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

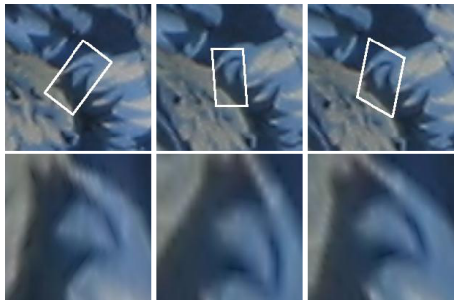
Write regions as vectors

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

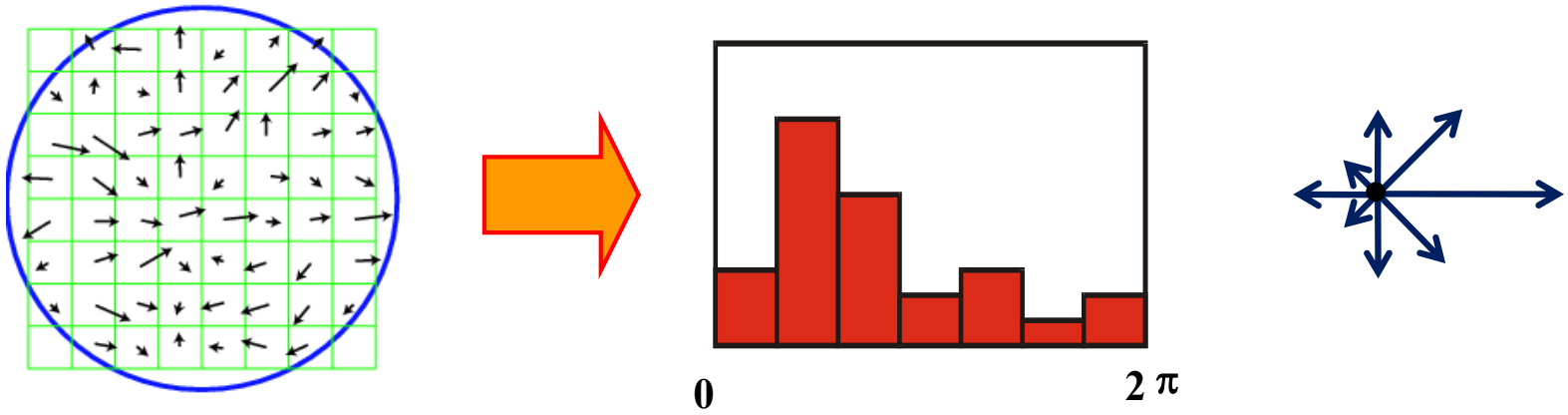


Local descriptors

- Disadvantage of patches as descriptors:
  - Small shifts can affect matching score a lot



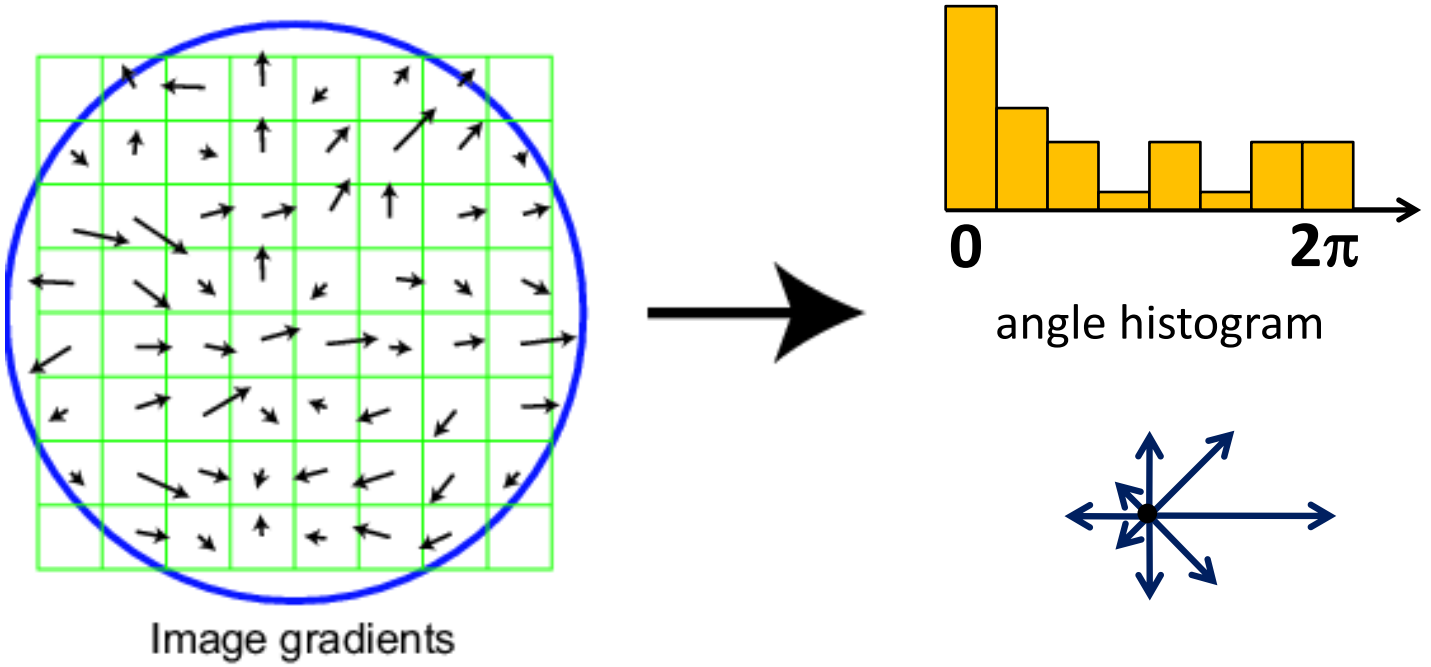
- Solution: histograms



SIFT descriptor

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



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

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor

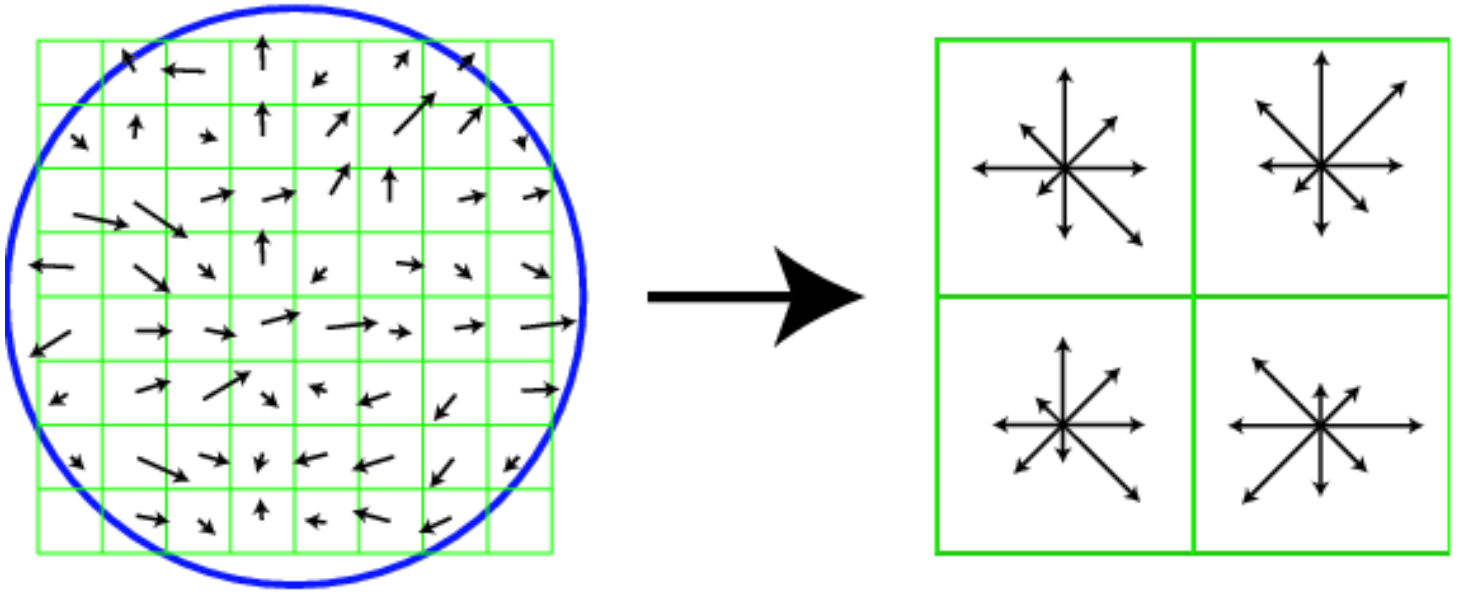


Image gradients

Keypoint descriptor

SIFT summary

- Sparse, distinctive point features
- Sometimes unclear what features correspond to in image
- **Translation** independent by using **local histogram**
- **Rotation** independent by **orientation adjustment**
- **Scale** independent by **extremal scale estimation**
- **Illumination** independent by **descriptor normalisation**
- Widely used
- Real-time implementation possible

Local descriptors

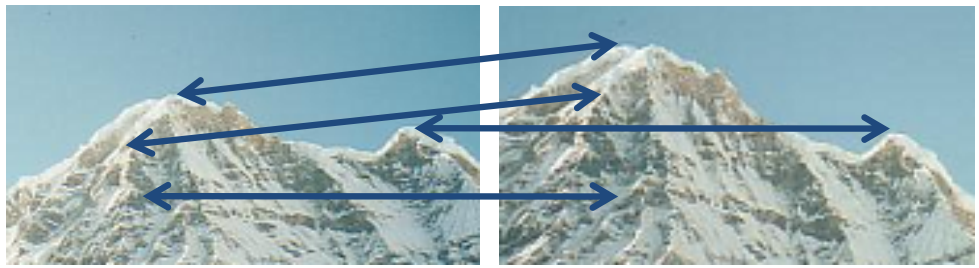
- Histogram-based descriptors
  - Based on the histogram of oriented gradient
  - SIFT, SURF, GLOH and HOG
- Compact descriptors
  - Based on binary strings obtained comparing pairs of image intensities
  - BRIEF, ORB, BRISK and FREAK

Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views



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

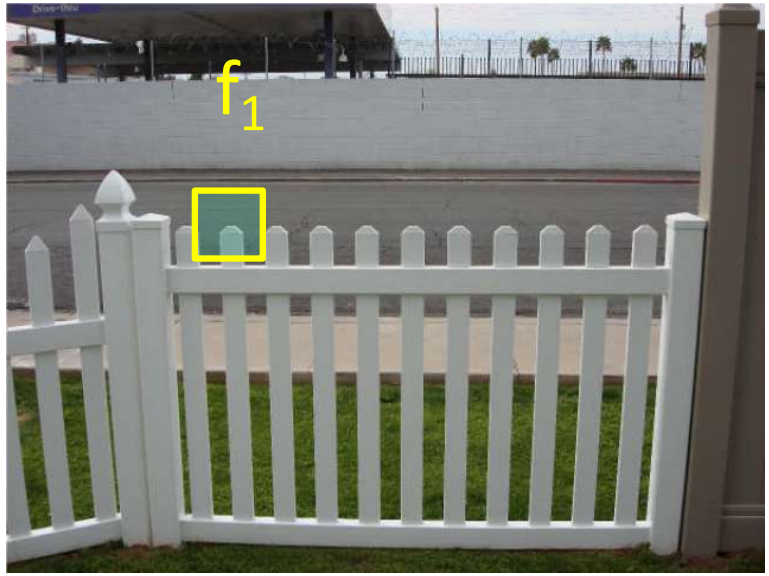
I_1

I_2

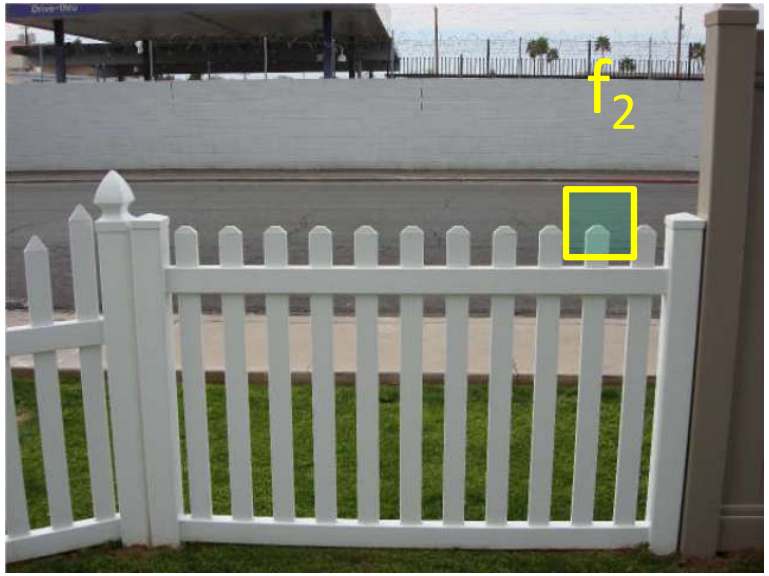
Feature matching

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

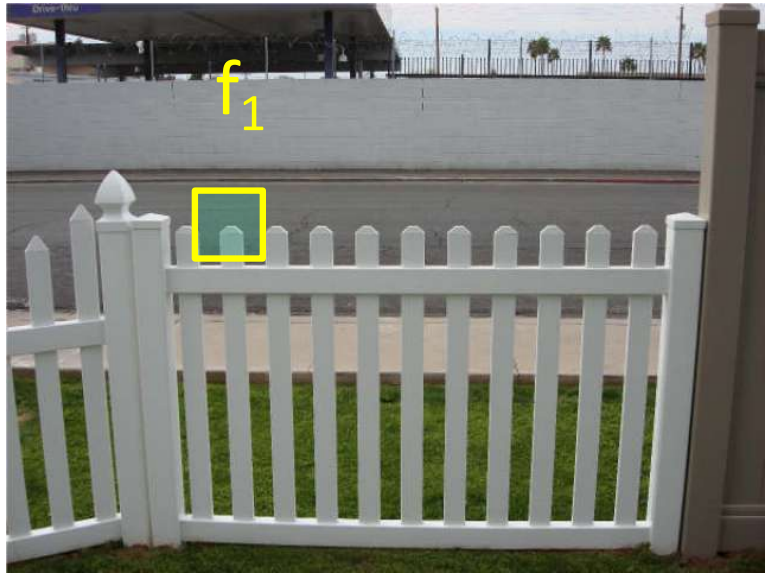


I_2

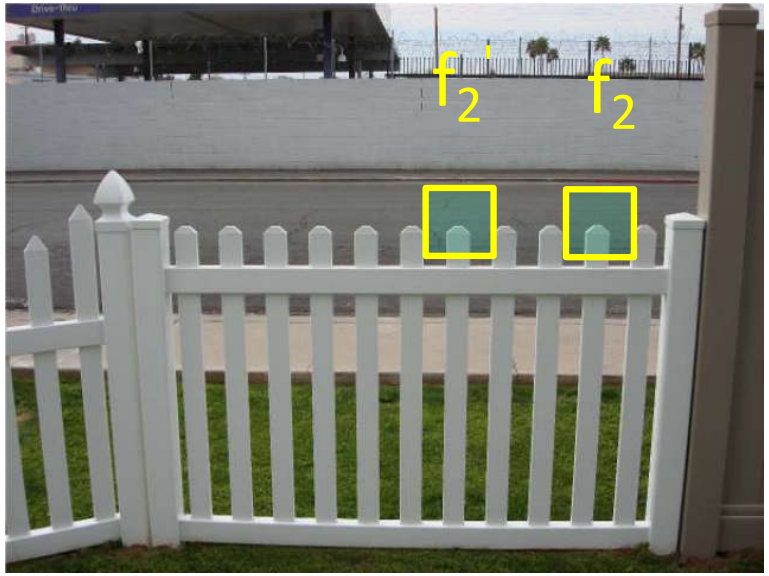
Feature matching

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

- 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 large values (~1) for ambiguous matches

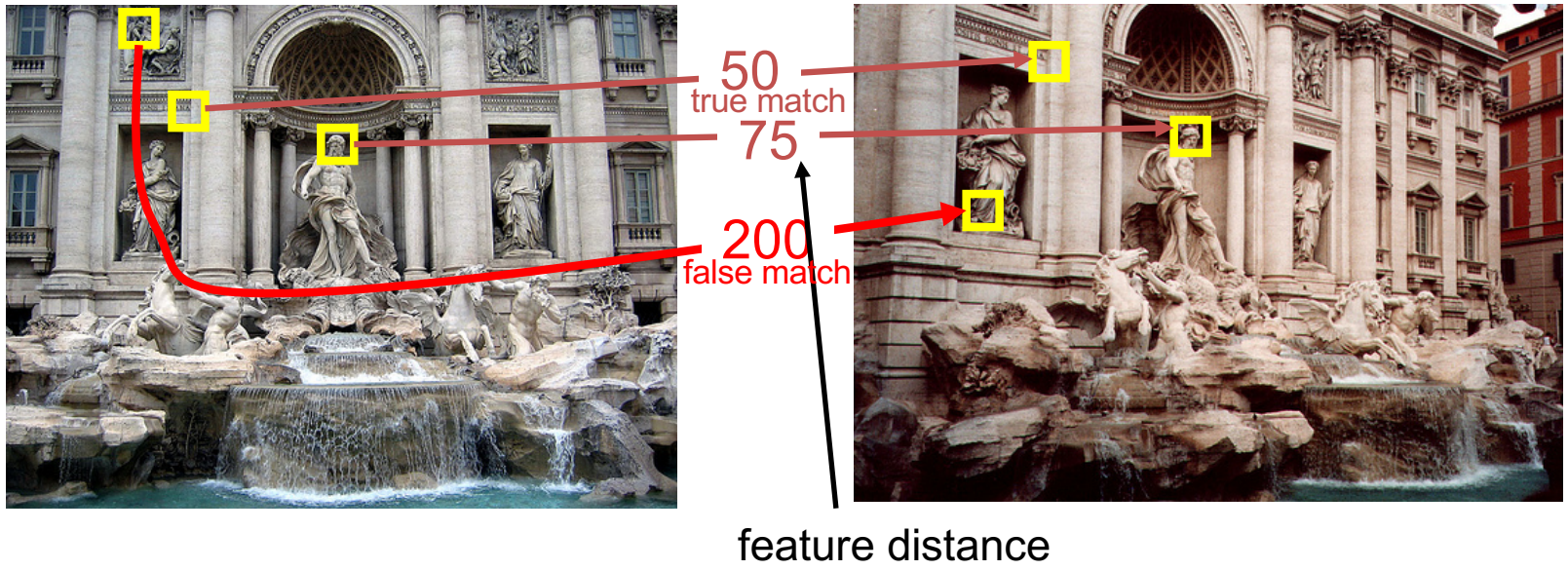


I_1



I_2

Feature matching



- Eliminate **bad matches**: throw out features with distance > threshold

Applications

Features are used for:

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

Applications

Automatic mosaicing

http://matthewalunbrown.com/autostitch/autostitch.html

Applications

Wide baseline stereo

[Image from T. Tuytelaars ECCV 2006 tutorial]