Computer Vision

Bogdan Alexe

bogdan.alexe@fmi.unibuc.ro

University of Bucharest, 2nd semester, 2023-2024

Course structure

1. **Features and filters: low-level vision**

   Linear filters, color, texture, edge detection, template matching

2. **Grouping and fitting: mid-level vision**

   Fitting curves and lines, robust fitting, RANSAC, Hough transform, segmentation

3. **Multiple views**

   Local invariant feature and description, epipolar geometry and stereo, object instance recognition

4. **Object Recognition: high – level vision**

   Object classification, object detection, part based models, bovw models

5. **Video understanding**

   Object tracking, background subtraction, motion descriptors, optical flow

Local features – lab class 4

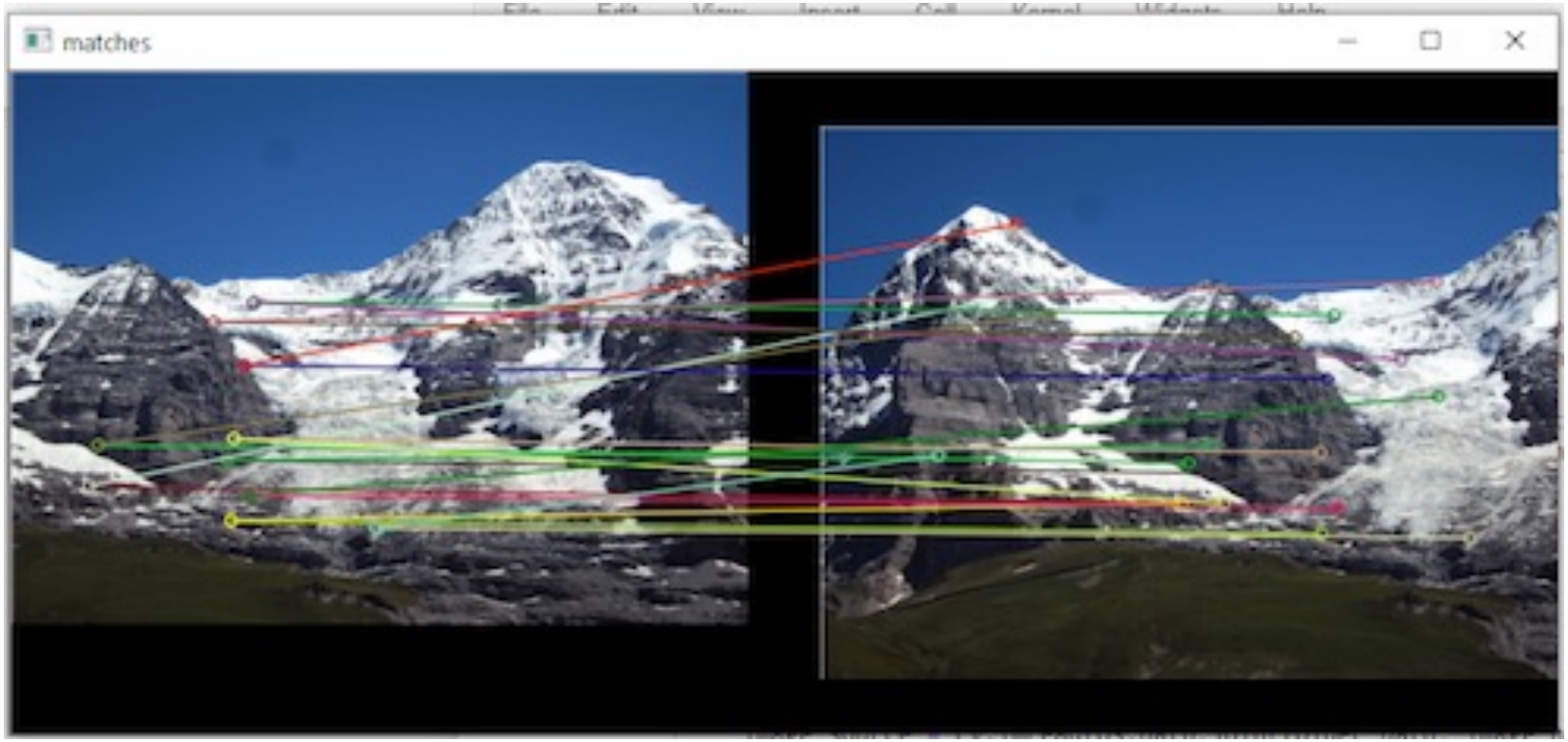
- local image feature = a region in the image that has a rich image content (brightness variation, color variation), has a well defined representation for matching other similar features, has a well-defined position in the image and it is geometrically (translation, rotation, ...) and photometrically (brightness, exposure, ...) invariant.

Local features – lab class 4

- local image feature = a region in the image that has a rich image content (brightness variation, color variation), has a well defined representation for matching other similar features, has a well-defined position in the image and it is geometrically (translation, rotation, ...) and photometrically (brightness, exposure, ...) invariant.

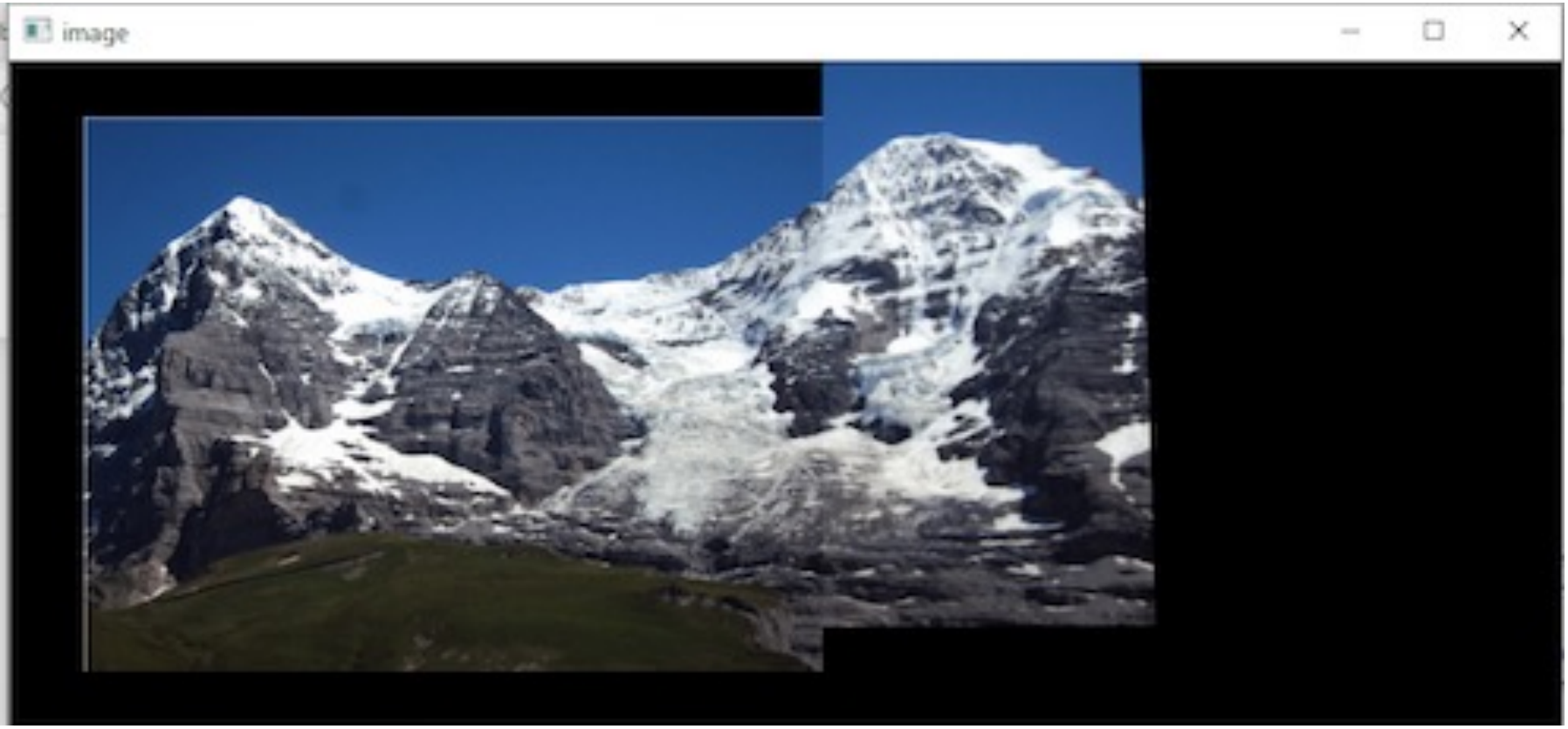
Matching local features – lab class 4

- matching local image feature = find correspondence points between images based on local features that look the same



Panorama stitching – lab class 4

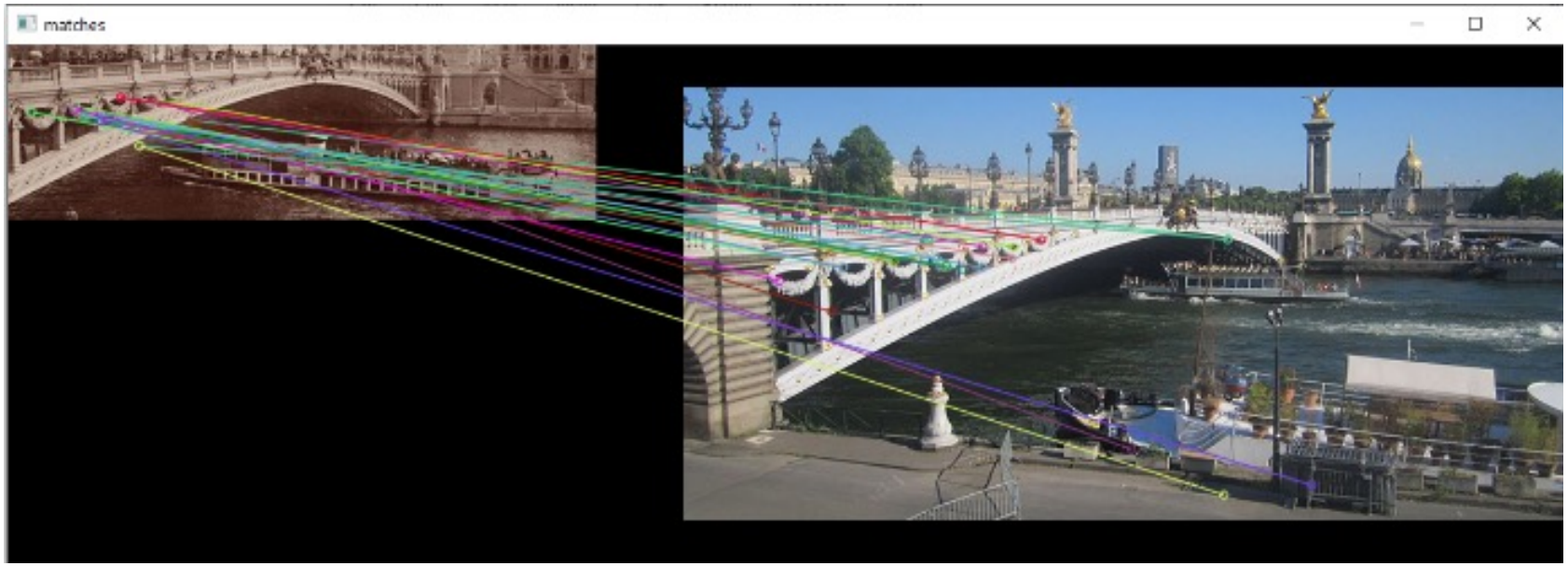
- align images in order to obtain panorama images



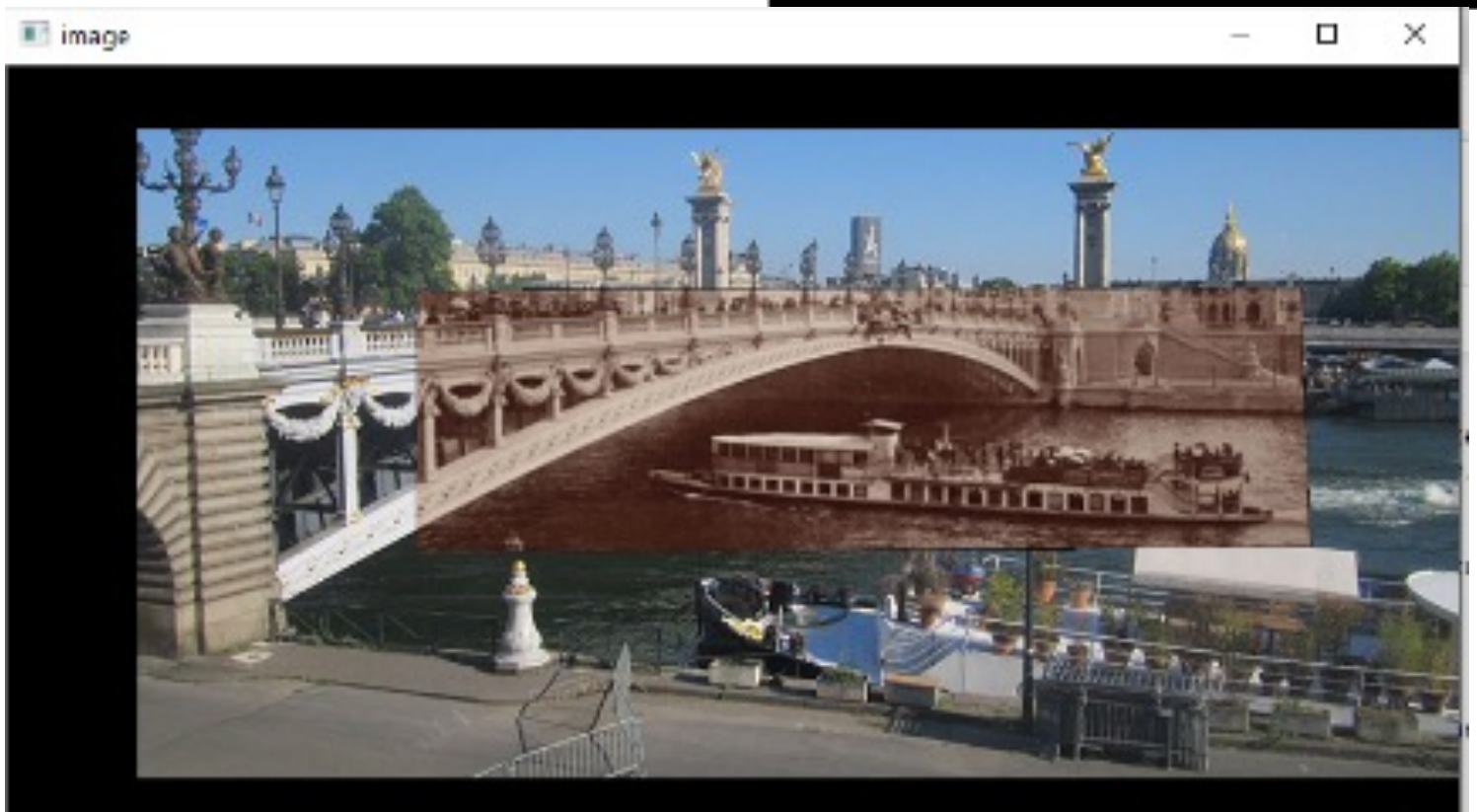
Panorama stitching – lab class 4

- align images in order to obtain panorama images

Then and now – lab class 4



Then and now – lab class 4



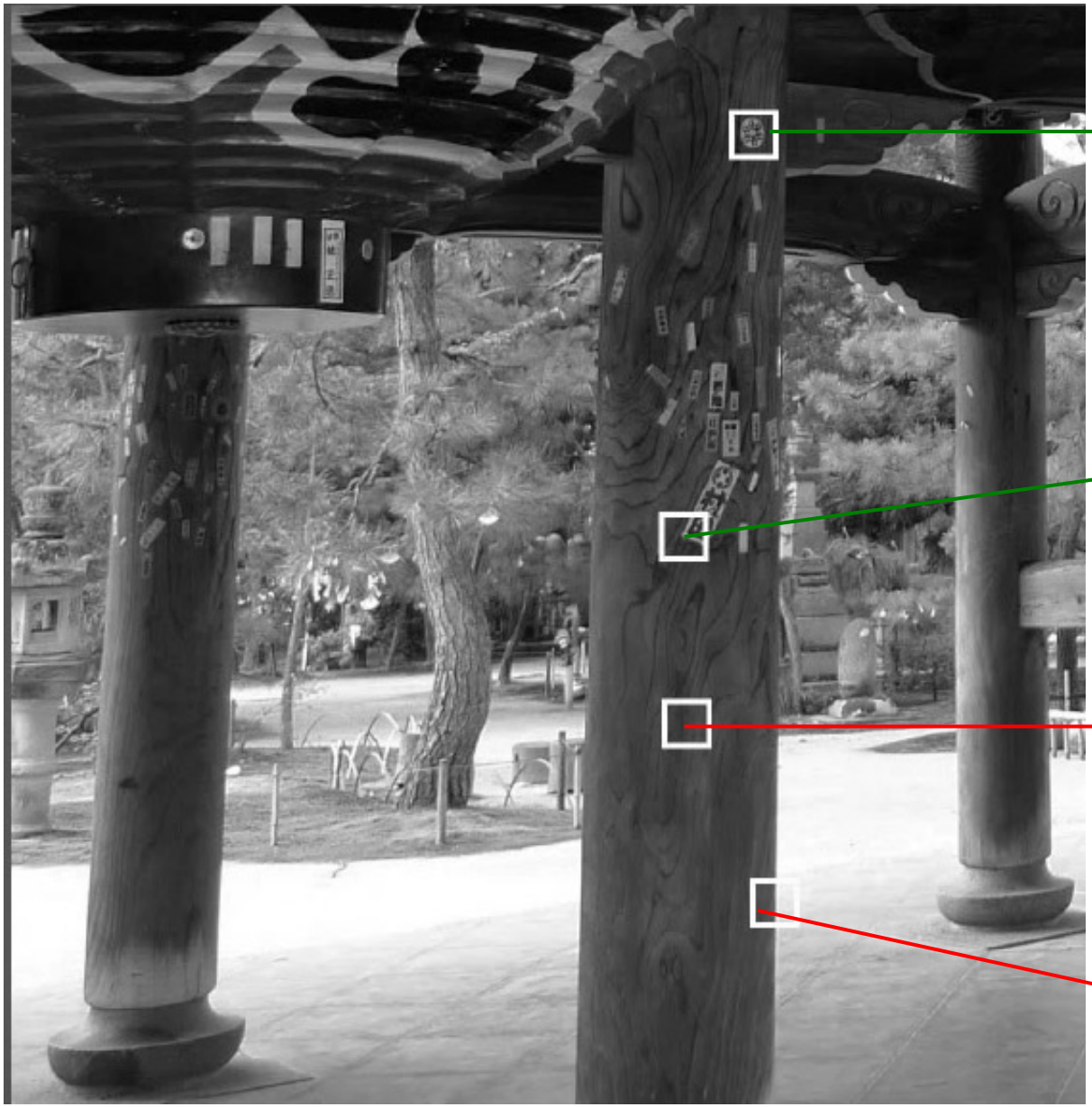
Detecting local invariant features

- Detection of interest points

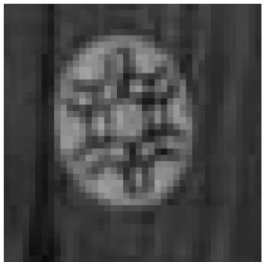
Uniform regions
Ambiguous for matching

Edges
Ambiguous for matching

Corners
Good pentru matching

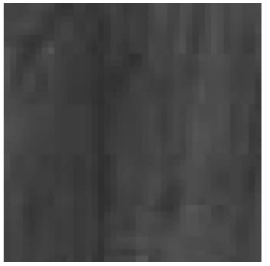


circular region (blob)

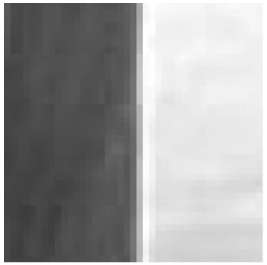


corner

uniform region



edge

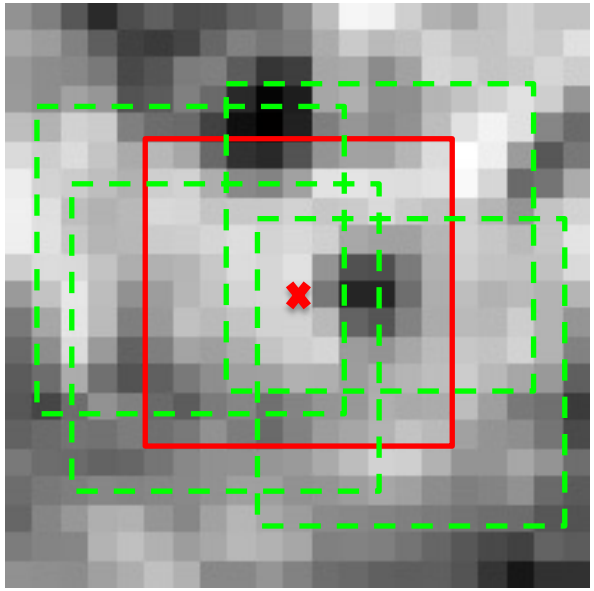


Detecting local invariant features

- Detection of interest points
  - Harris corner detection
  - Scale invariant blob detection: LoG
- Description of local patches

Detecting local invariant features

- How do we mathematically formulate what is a good local feature?

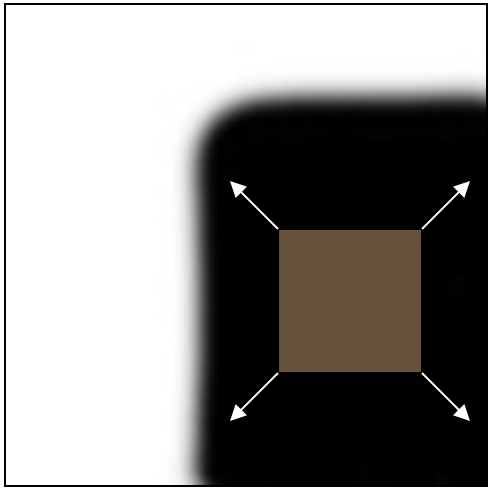


- Characterize the point by a small window centered in it

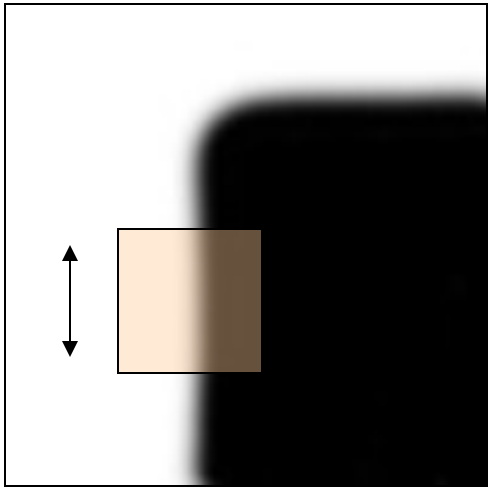
- Good local feature: shifting a window in *any direction* should give *a large change* in intensity (sum of squared distances)

Corner detection: Basic idea

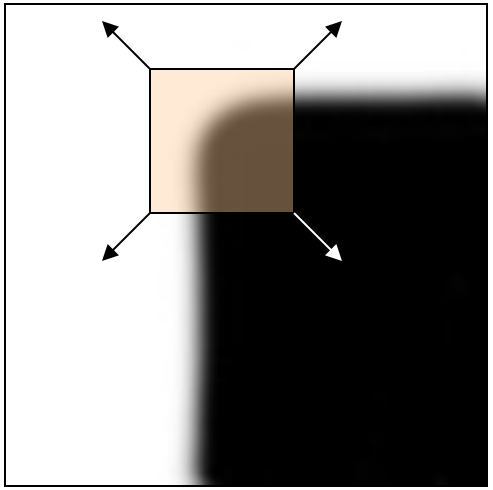
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions



"edge":
no change along the edge direction



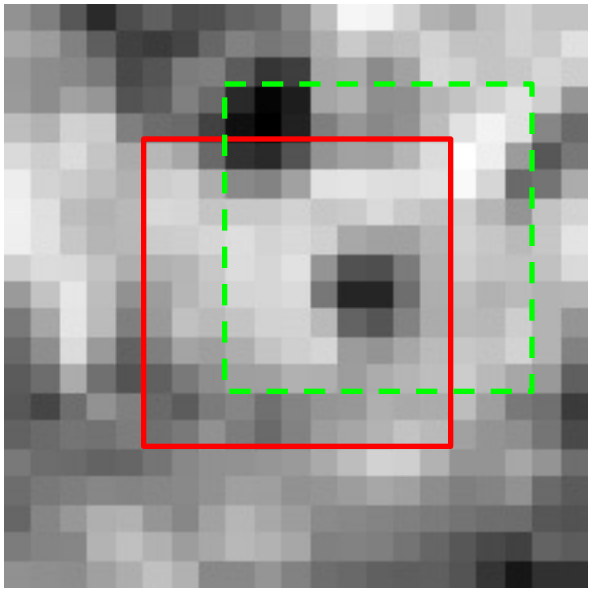
"corner":
significant change in all directions

Corner Detection: Derivation

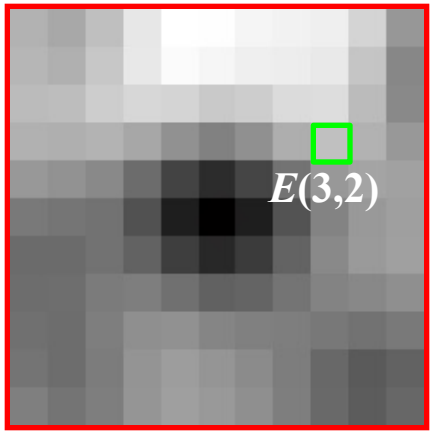
Change in appearance of window W for the shift $[u,v]$:

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

$I(x, y)$



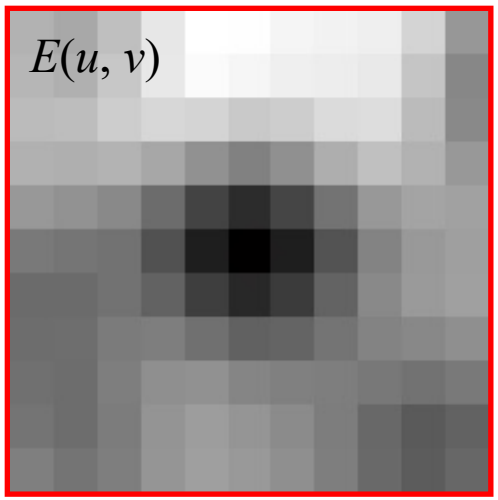
$E(u, v)$

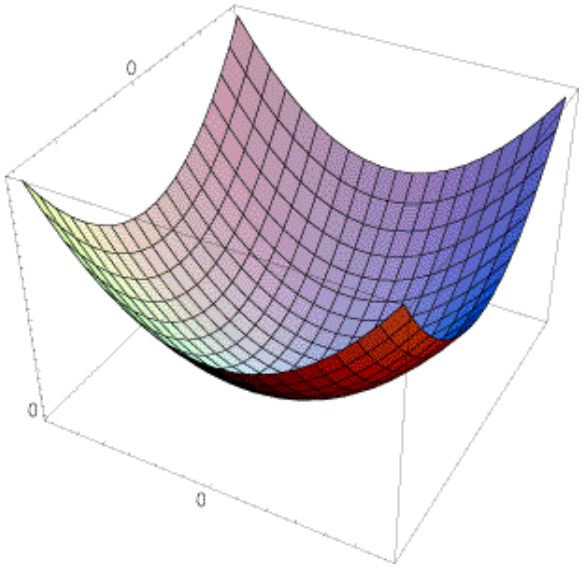


Corner Detection: Derivation

- $E(u,v)$ can be locally approximated by a quadratic surface:

$$E(u,v) \approx u^2 \sum_{x,y} I_x^2 + 2uv \sum_{x,y} I_x I_y + v^2 \sum_{x,y} I_y^2$$



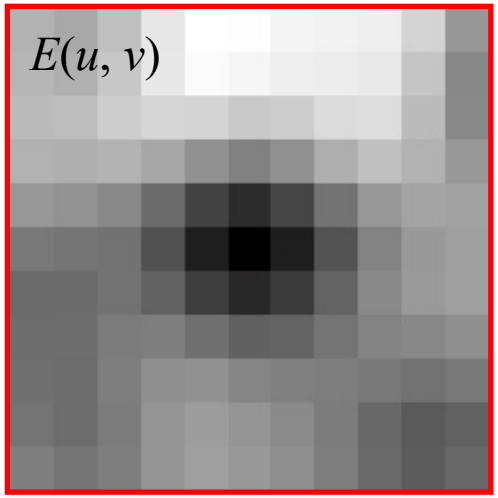


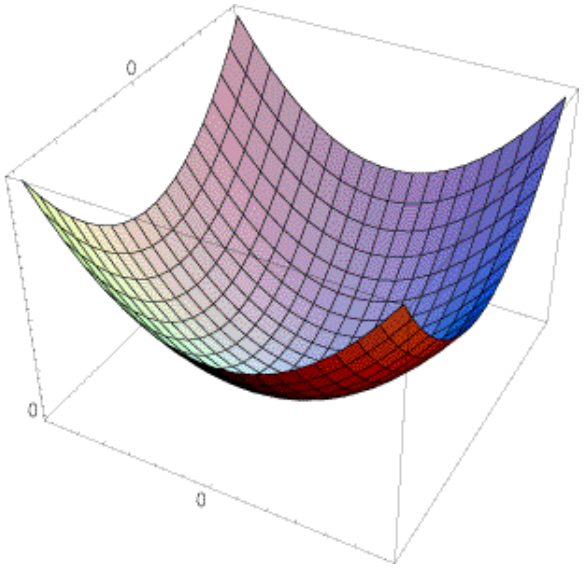
- In which directions does this surface have the fastest/slowest change?

Corner Detection: Derivation

- $E(u,v)$ can be locally approximated by a quadratic surface:

$$E(u,v) \approx u^2 \sum_{x,y} I_x^2 + 2uv \sum_{x,y} I_x I_y + v^2 \sum_{x,y} I_y^2$$





- Eigenvectors given by the largest/smallest eigenvalue

Corner Detection: Derivation

- $E(u,v)$ can be locally approximated by a quadratic surface:

$$E(u,v) \approx u^2 \sum_{x,y} I_x^2 + 2uv \sum_{x,y} I_x I_y + v^2 \sum_{x,y} I_y^2$$

$$= \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} \sum_{x,y} I_x^2 & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

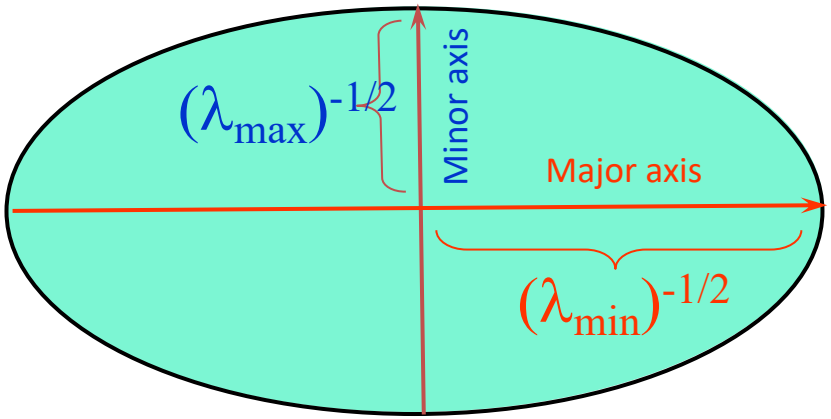
Second moment matrix M

Interpreting the second moment matrix

Consider the axis-aligned case (gradients are either horizontal or vertical):

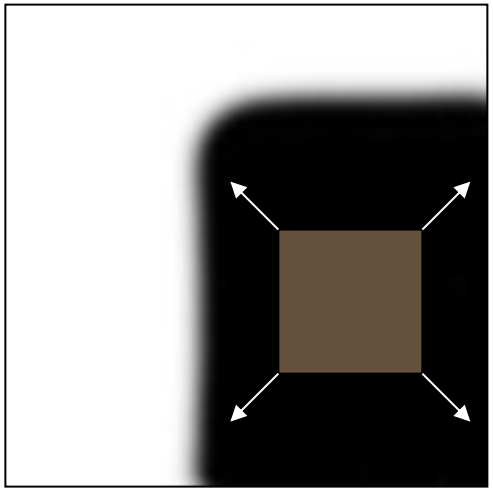
$$M = \begin{bmatrix} \sum_{x,y} I_x^2 & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y^2 \end{bmatrix} =$$

$$[u \;\; v] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = 1$$



Interpreting the second moment matrix

Consider the axis-aligned case (gradients are either horizontal or vertical):



$$M = \begin{bmatrix} \sum_{x,y} I_x I_x & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y I_y \end{bmatrix}$$

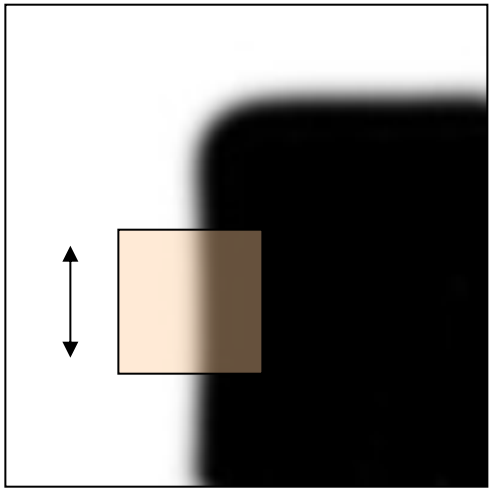
"flat" region:
no change in all directions

$$M \approx \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The eigenvalues of M = solutions of the equation $\det(M - \lambda I_2) = 0$ are $\lambda_1 \approx \lambda_2 \approx 0$

Interpreting the second moment matrix

Consider the axis-aligned case (gradients are either horizontal or vertical):



$$M = \begin{bmatrix} \sum_{x,y} I_x I_x & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y I_y \end{bmatrix}$$

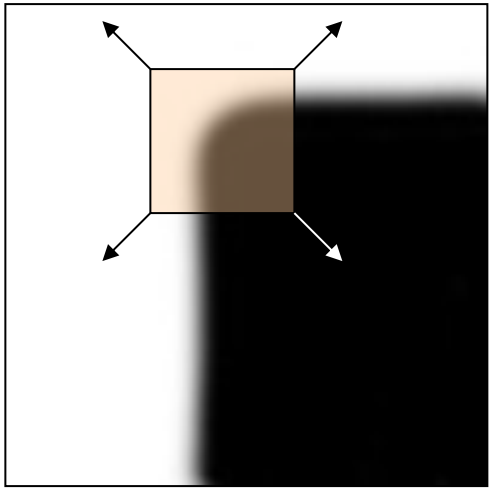
"edge":
no change along
the edge direction

$$M \approx \begin{bmatrix} \lambda & 0 \\ 0 & 0 \end{bmatrix}$$

The eigenvalues of M = solutions of the equation $\det(M - \lambda I_2) = 0$ are $\lambda_1 >> 0$, $\lambda_2 \approx 0$

Interpreting the second moment matrix

Consider the axis-aligned case (gradients are either horizontal or vertical):



$$M = \begin{bmatrix} \sum_{x,y} I_x I_x & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y I_y \end{bmatrix}$$

"corner":
significant change in all directions

$$M \approx \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

The eigenvalues of M = solutions of the equation $\det(M - \lambda I_2) = 0$ are $\lambda_1, \lambda_2 >> 0$

Interpreting the second moment matrix

Consider the axis-aligned case (gradients are either horizontal or vertical):

$$M = \begin{bmatrix} \sum_{x,y} I_x^2 & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means dominant gradient directions align with x or y axis

Look for locations where **both** λ's are large.

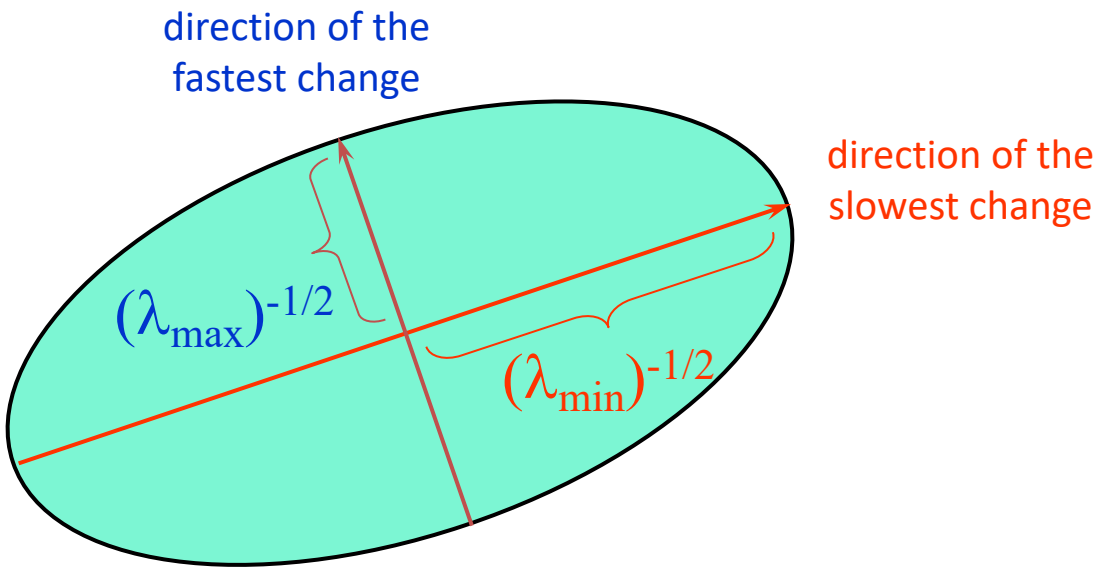
If either λ_1 or λ_2 is close to 0, then this is **not** corner-like.

Interpreting the second moment matrix

In the general case, need to *diagonalize* M:

$$M = R^{-1}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by R:



Interpreting the second moment matrix

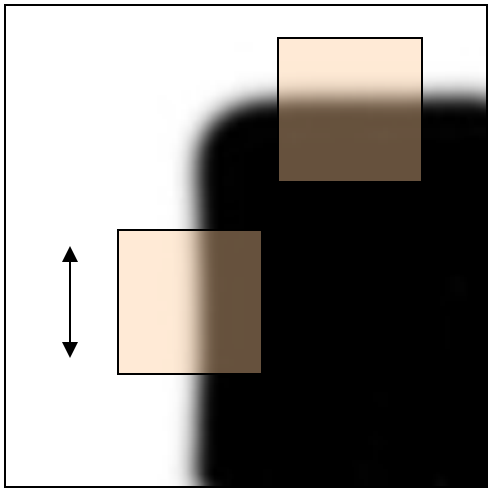
In the general case, need to *diagonalize* M:

$$M = R^{-1}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

$$Mx_i = \lambda_i x_i$$

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

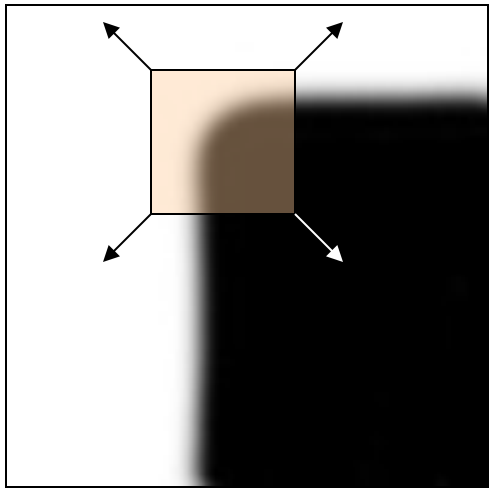
Corner response function



"edge":

$\lambda_1 >> \lambda_2$

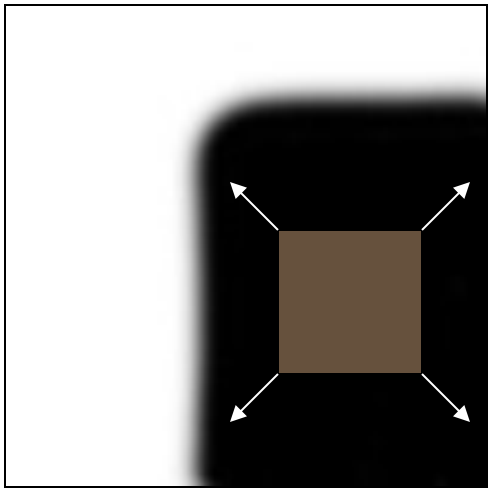
$\lambda_2 >> \lambda_1$



"corner":

λ_1 and λ_2 are large,

$\lambda_1 \sim \lambda_2$;



"flat" region

λ_1 and λ_2 are small

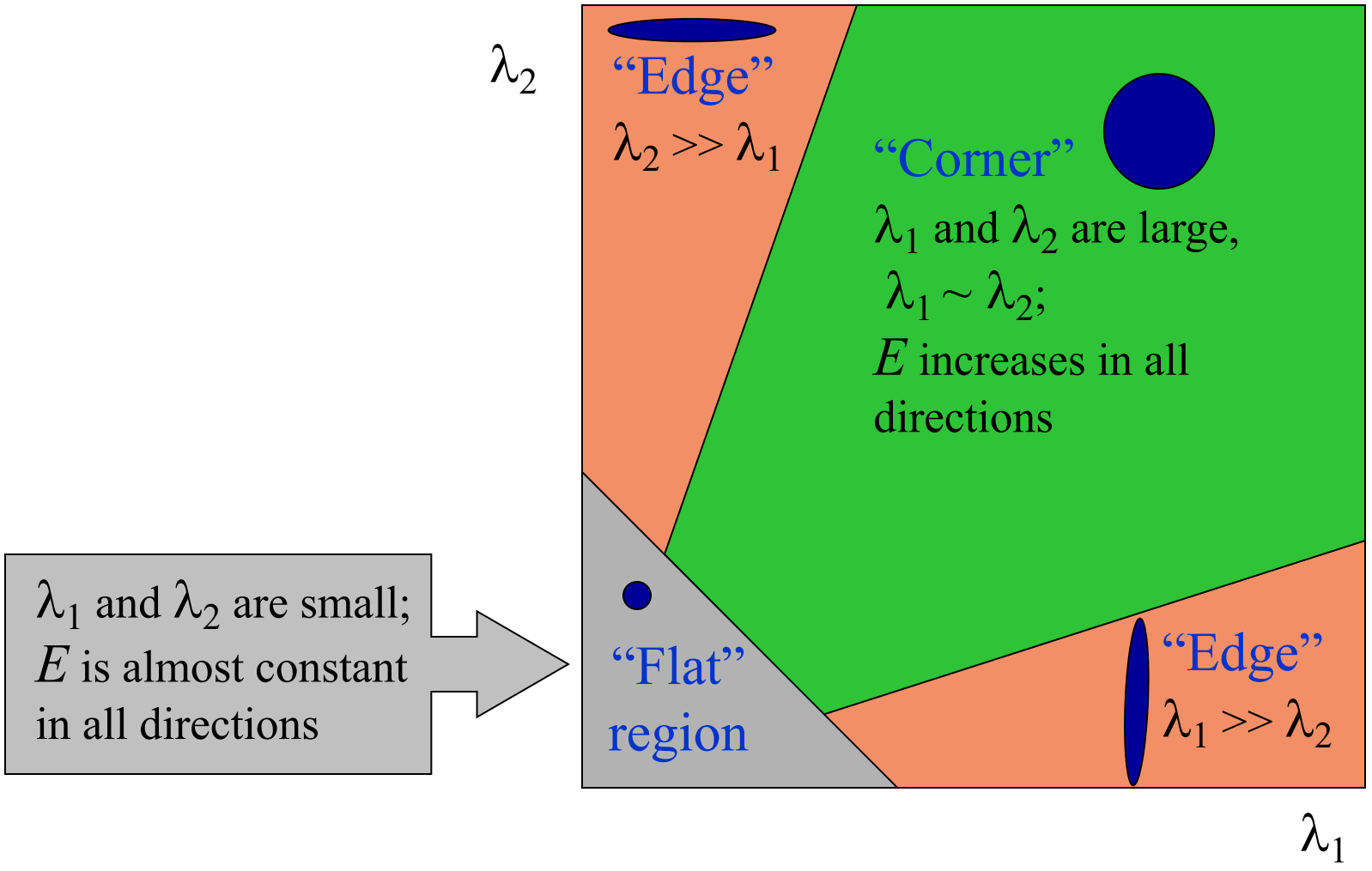
Cornerness score
(other variants possible)

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

$$f = det(M) - \alpha \cdot trace(M)^2 = \lambda_1 \cdot \lambda_2 - \alpha * (\lambda_1 + \lambda_2)^2$$

Interpreting the eigenvalues

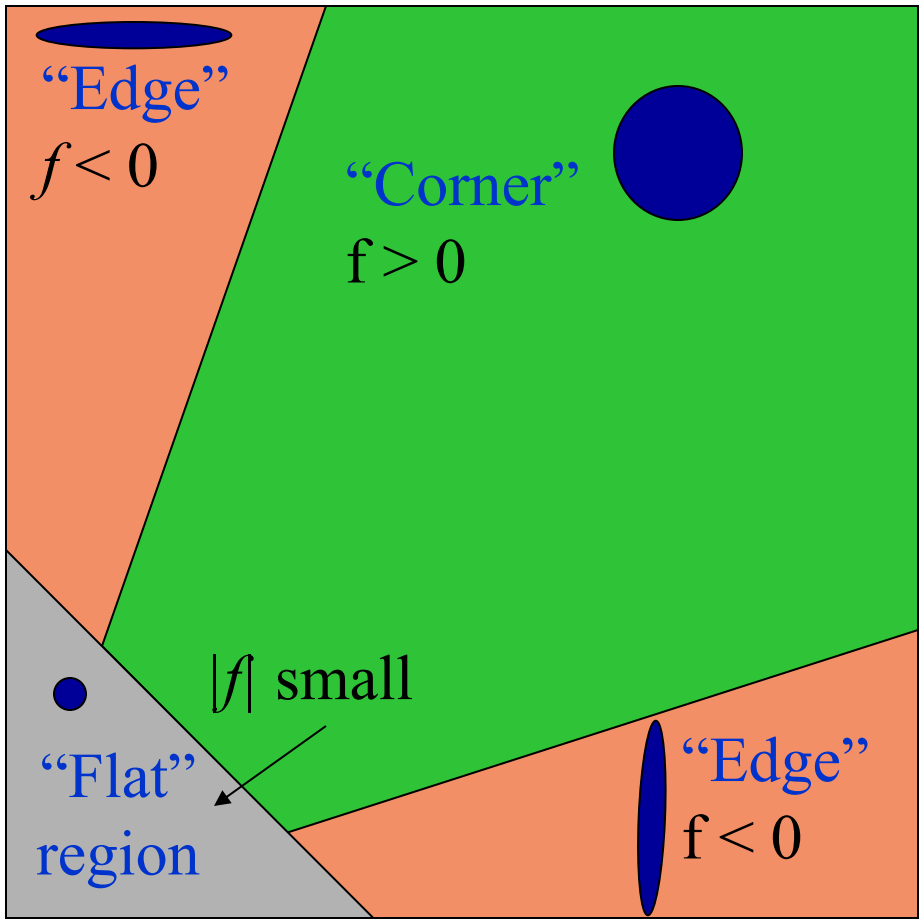
Classification of image points using eigenvalues of M:



Corner response function

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

α: constant (0.04 to 0.06)



Harris corner detector

1) Compute M matrix for each image window to get their *cornerness* scores.

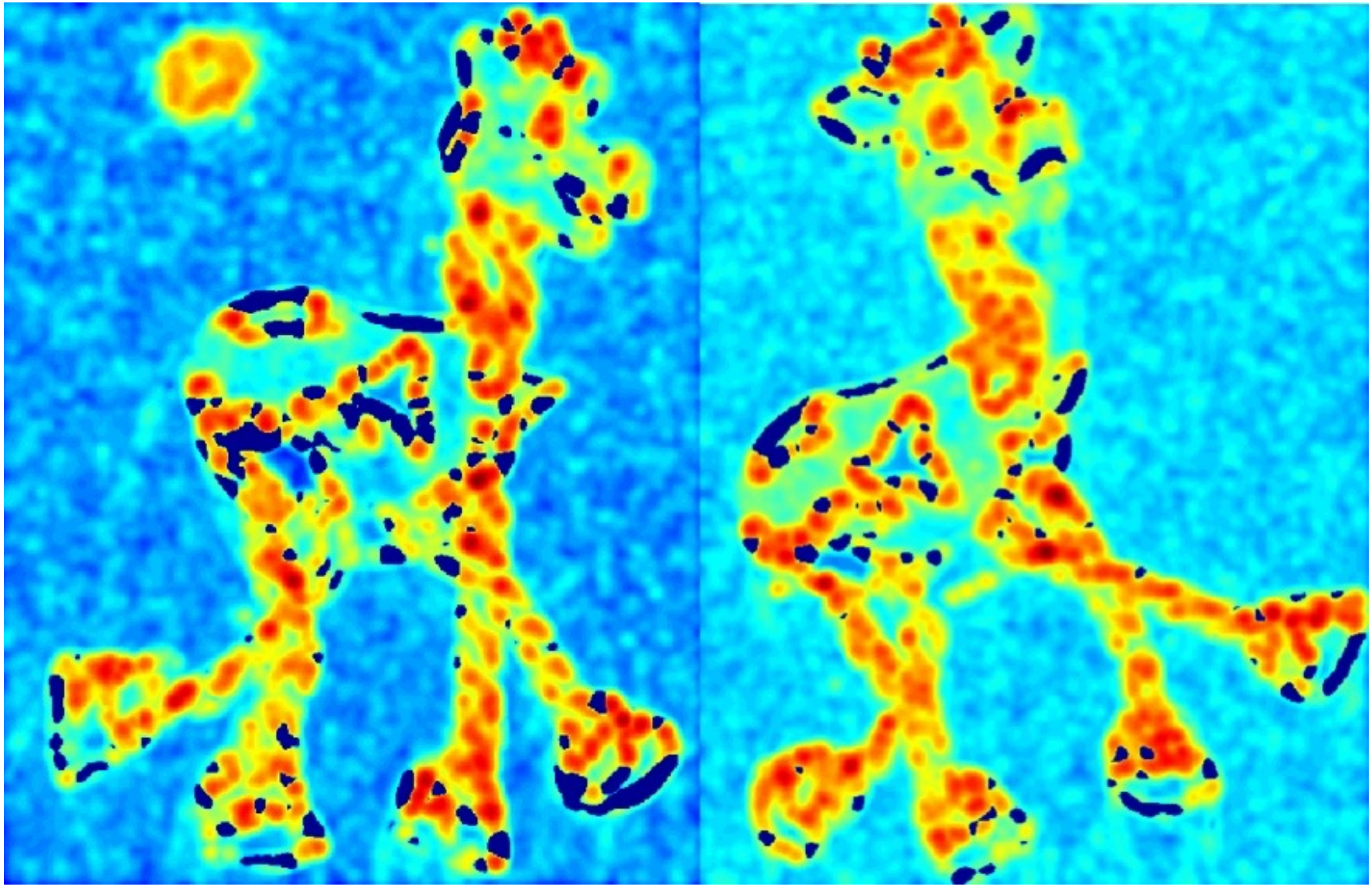
2) Find points whose surrounding window give large corner response (f > threshold)

3) Take the points of local maxima, i.e., perform non-maximum suppression

Harris Detector: Steps

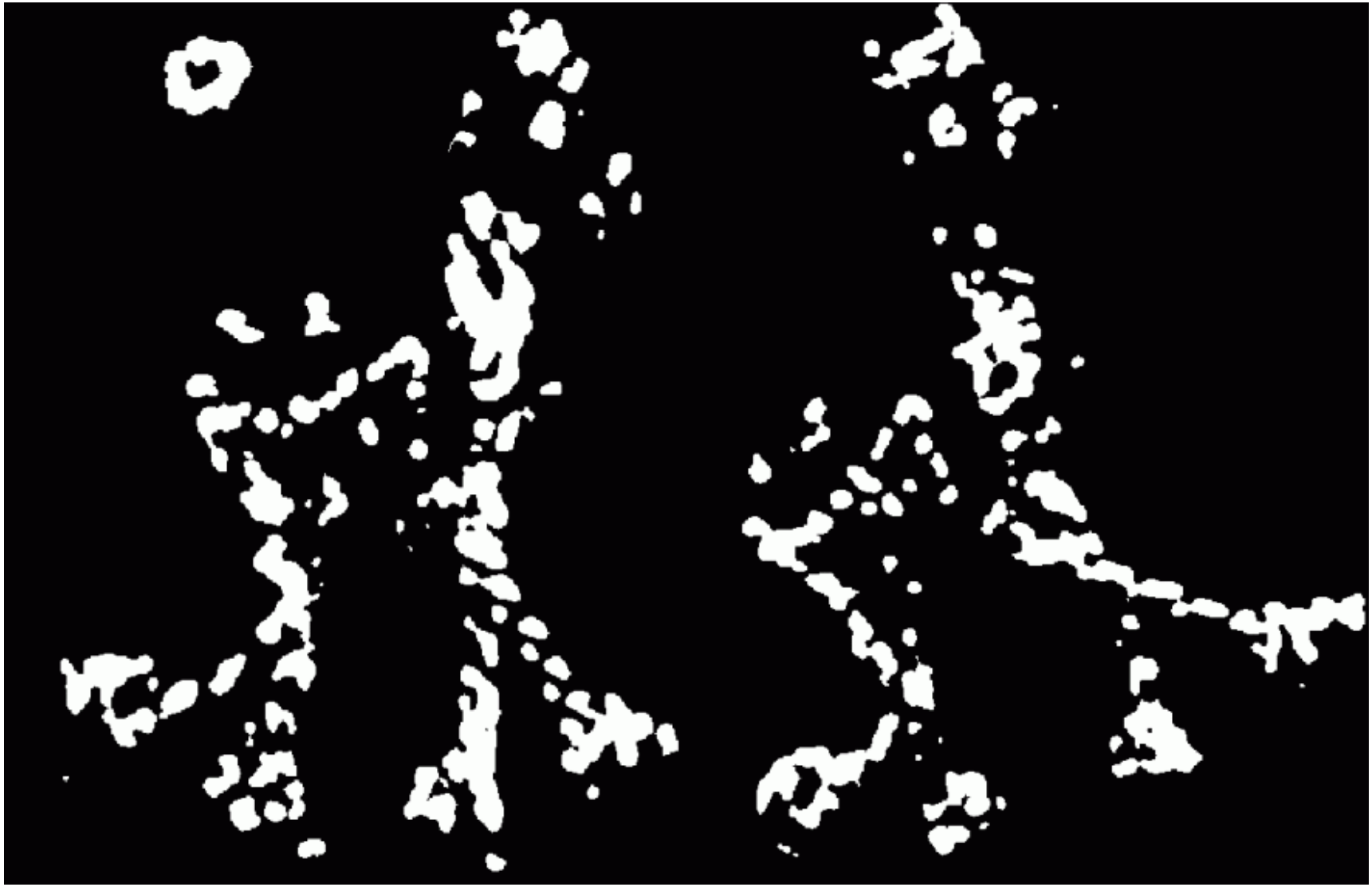
Harris Detector: Steps

Compute corner response f



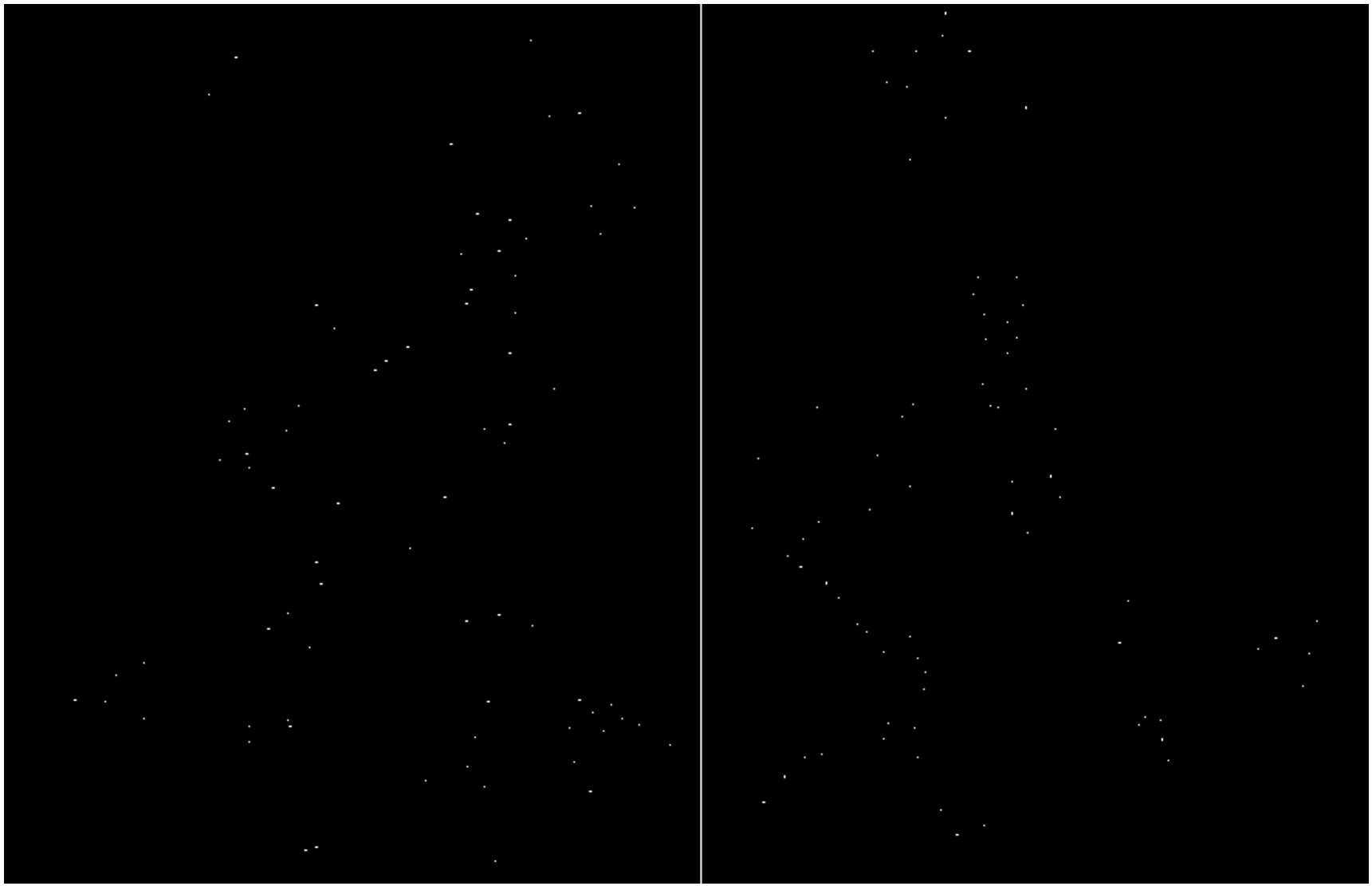
Harris Detector: Steps

Find points with large corner response: $f >$ threshold

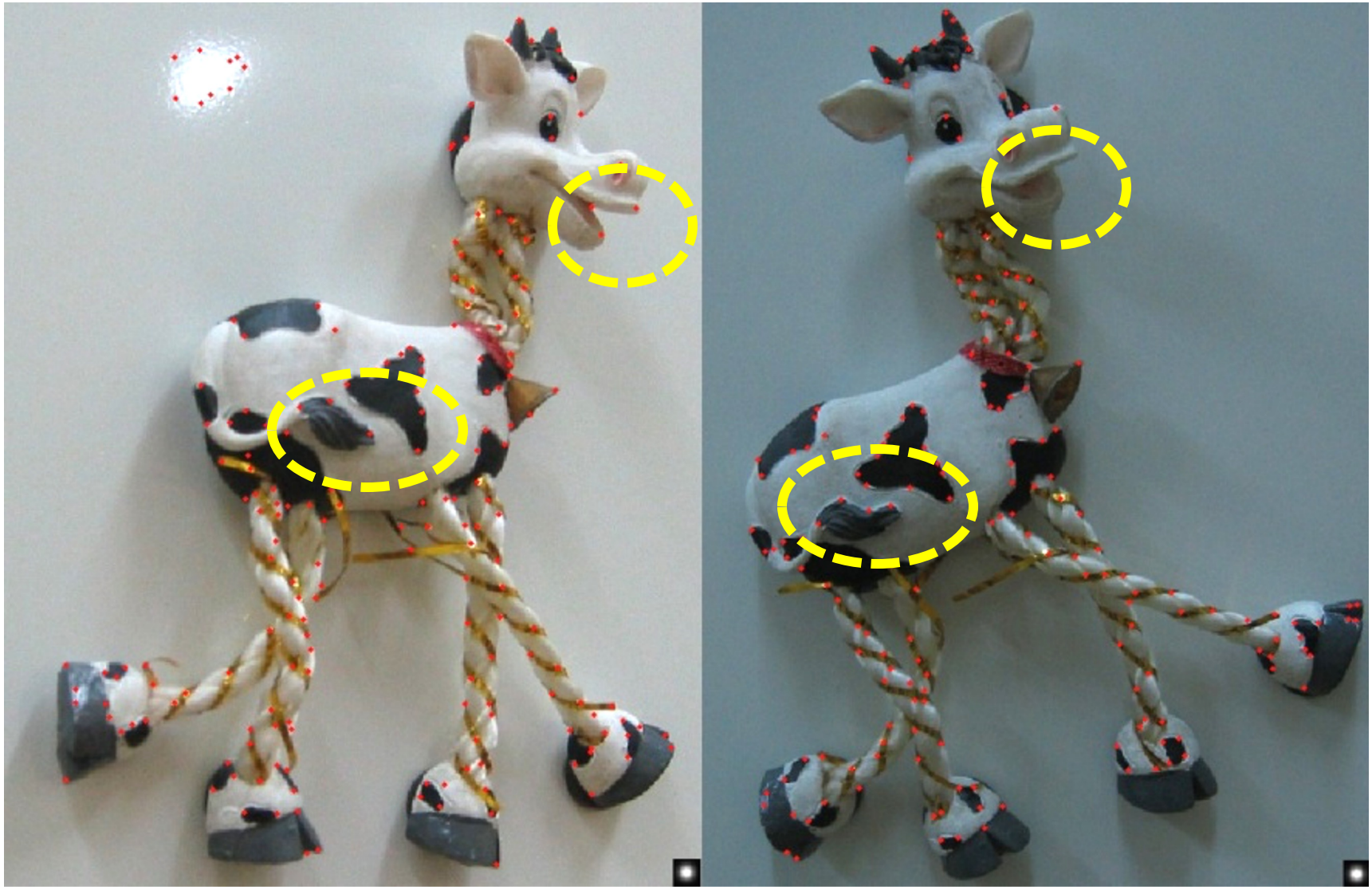


Harris Detector: Steps

Take only the points of local maxima of f



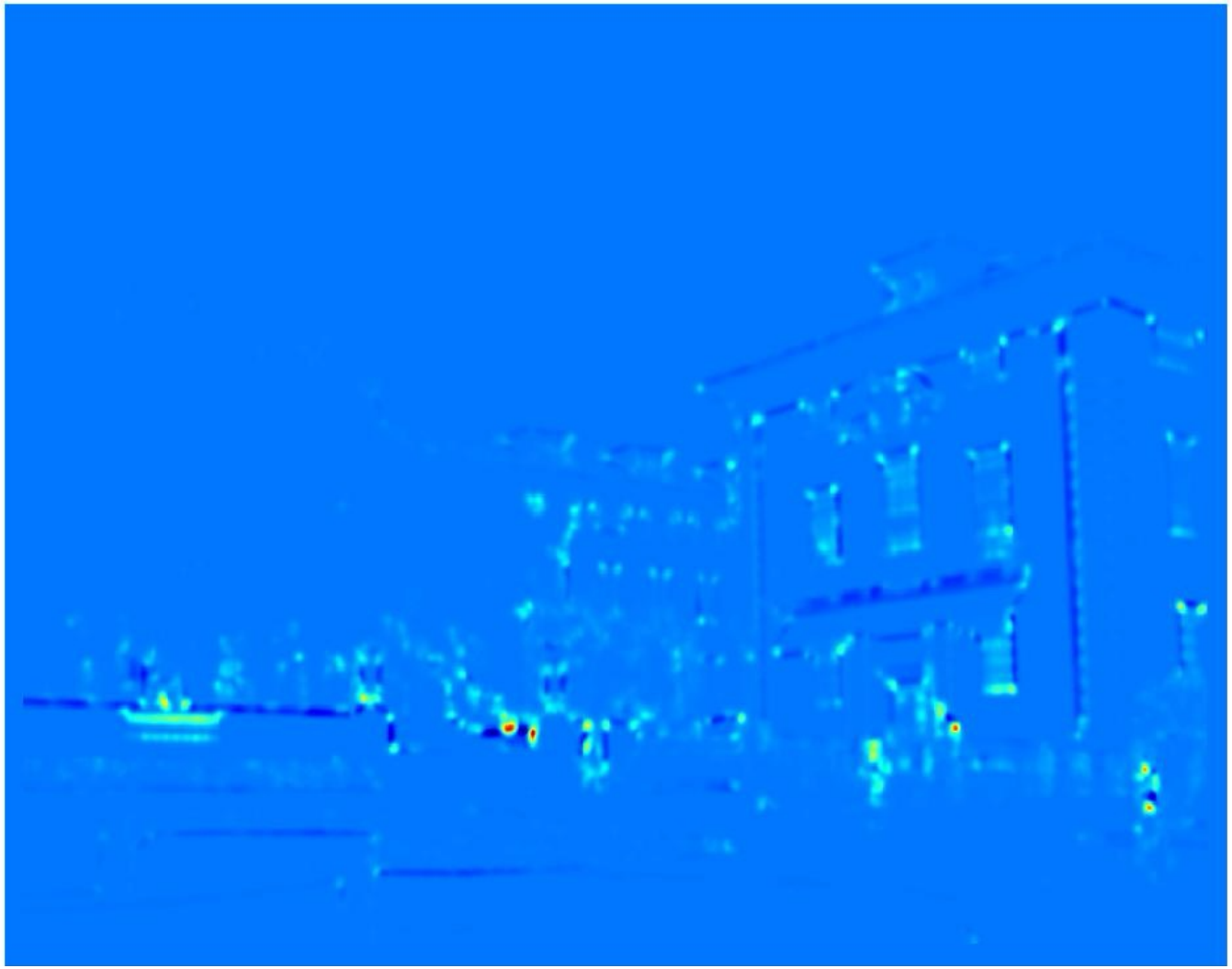
Harris Detector: Steps



Harris Detector: Steps

Harris Detector: Steps

Compute corner response f



Harris Detector: Steps

Take only the points of local maxima of f

Robustness of corner features

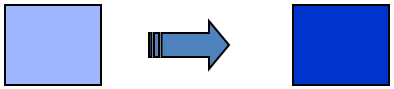
- What happens to corner features when the image undergoes geometric or photometric transformations?

Invariance and covariance

We want to find corners that are:

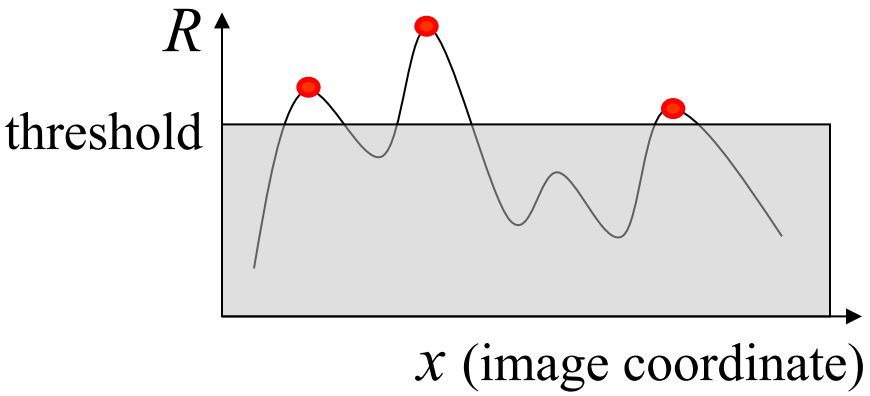
- ***invariant*** to ***photometric transformations***
- ***covariant*** to ***geometric transformation***

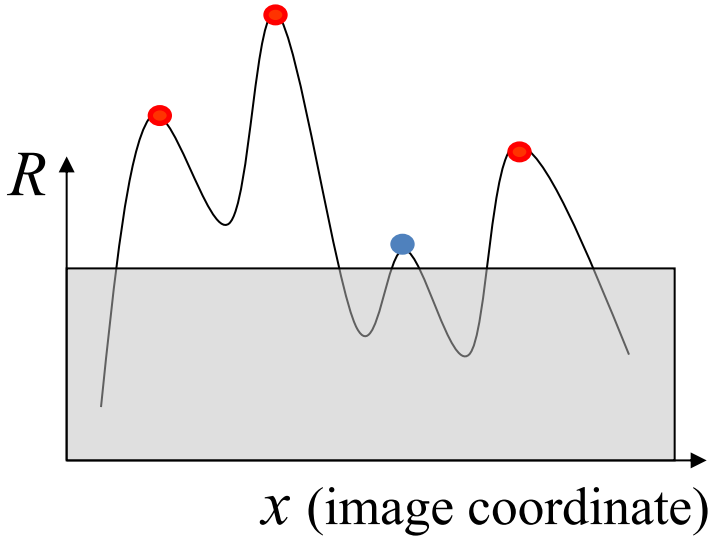
Photometric transformations: affine intensity changes



$I \rightarrow a\,I + b$

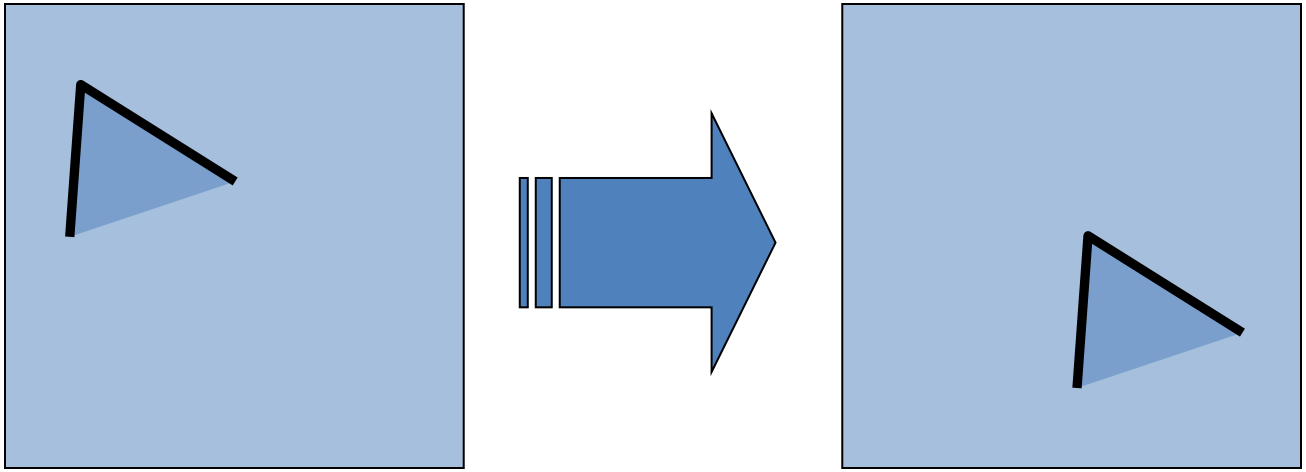
- Only derivatives are used, so invariant to intensity shift $I \rightarrow I + b$
- Intensity scaling: $I \rightarrow a\,I$





Partially invariant to affine intensity change

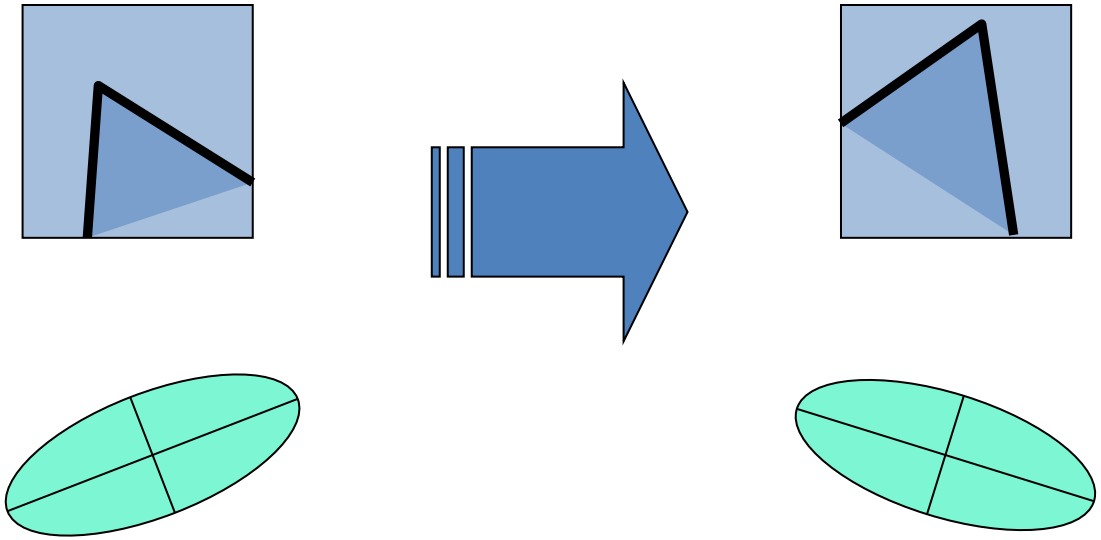
Geometric transformations: translations



- Derivatives and window function are shift-invariant

Corner location is *covariant* w.r.t. translation

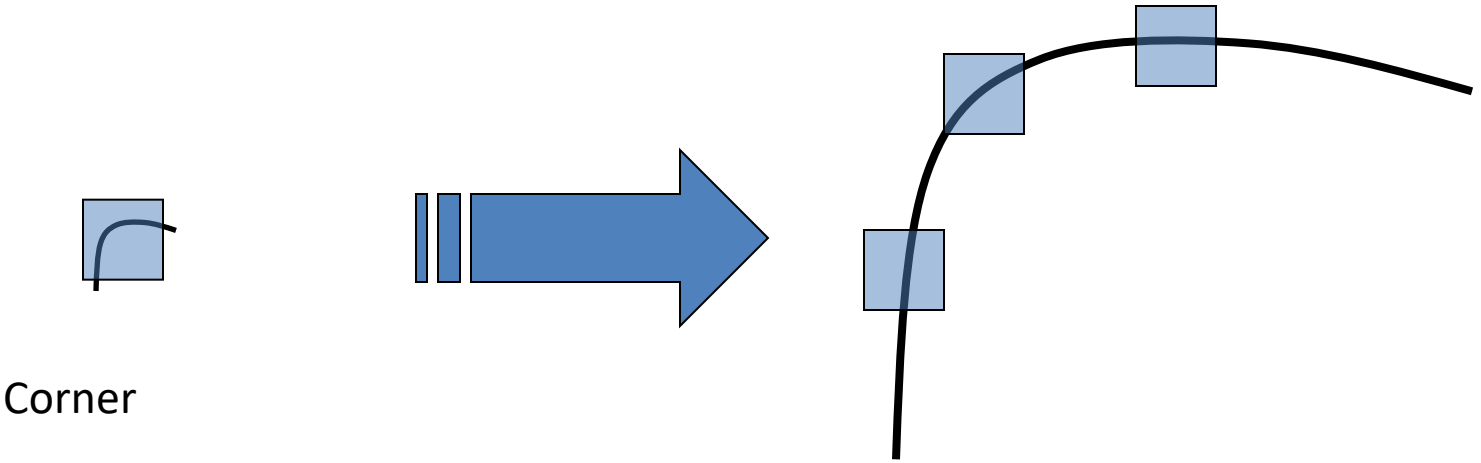
Geometric transformations: rotations



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

Corner location is covariant w.r.t. rotation

Geometric transformations: scaling



Corner

All points will be classified as edges

Corner location is not covariant w.r.t. scaling!

Robustness of corner features

- What happens to corner features when the image undergoes geometric or photometric transformations?
- We want to find corners that are:
  - ***invariant*** to ***photometric transformations***
  - ***covariant*** to ***geometric transformation***

Properties of Harris corner detector

Partially invariant to affine intensity change

Corner location is *covariant* w.r.t. translation

Corner location is covariant w.r.t. rotation

Corner location is not covariant w.r.t. scaling!

Detecting local invariant features

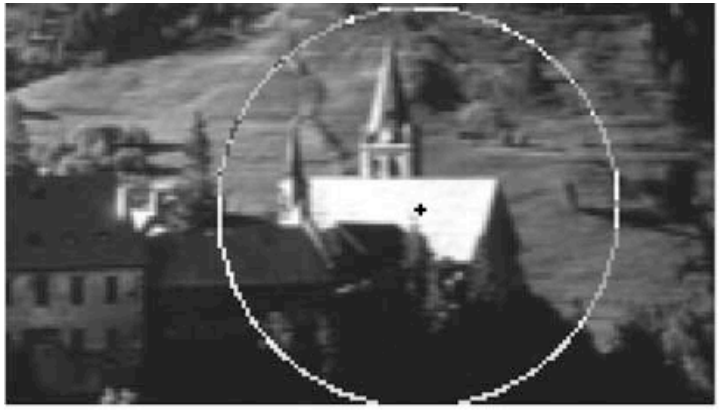
- Detection of interest points
  - Harris corner detection
  - Scale invariant blob detection: LoG
- Description of local patches

Scale invariant interest points

How can we independently select interest points in each image, such that the detections are repeatable across different scales?

Keypoint detection with scale selection

- We want to extract keypoints with *characteristic scales* that are *covariant* w.r.t. the image transformation



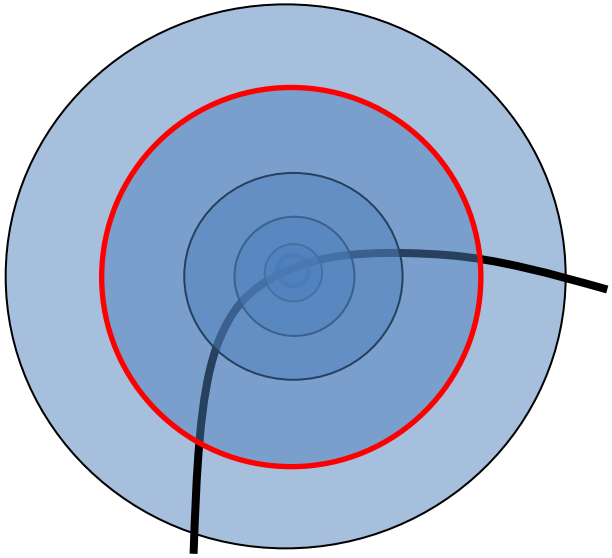
Automatic Scale Selection

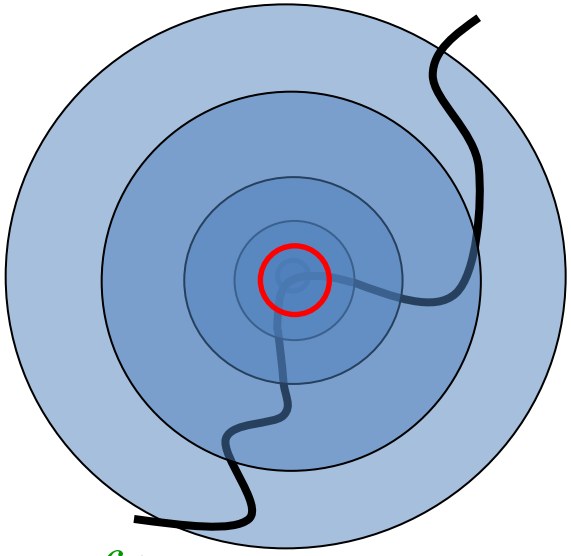
How to find corresponding patch sizes, with only one image in hand?

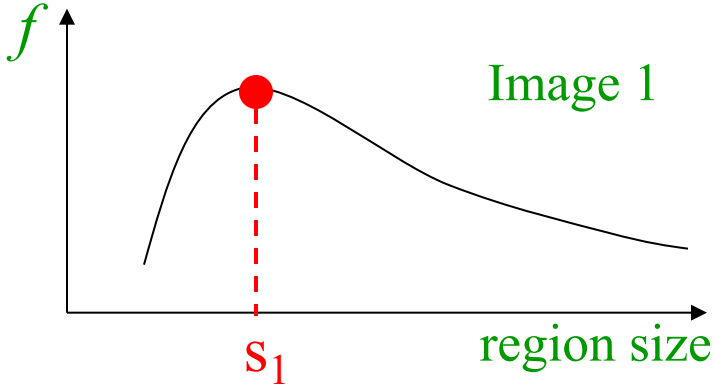
Automatic Scale Selection

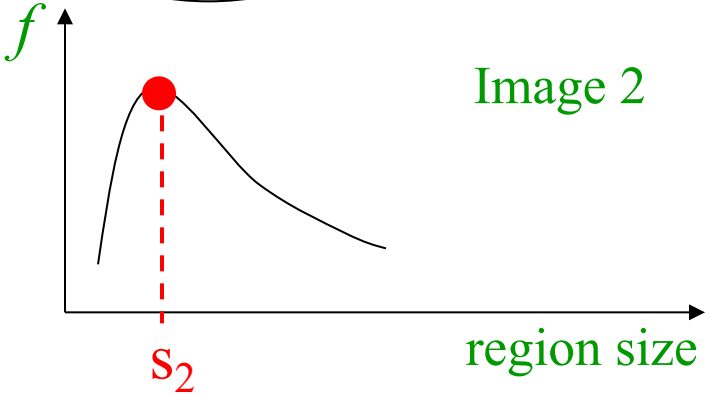
Intuition:

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



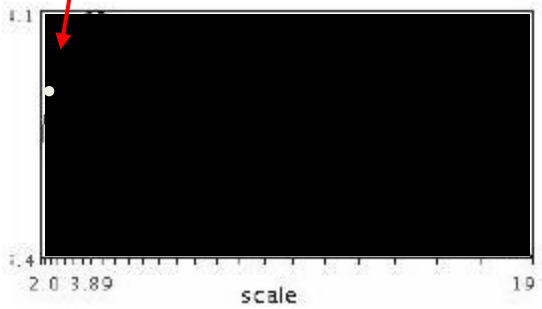




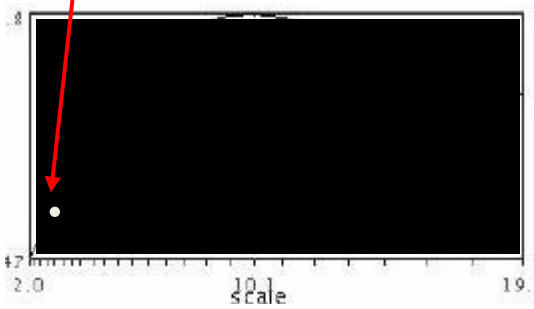


Automatic Scale Selection

- Function responses for increasing scale (scale signature)



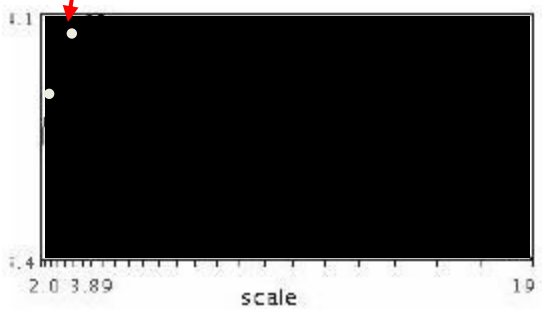
$f(I_{i_1 \ldots i_m}(x,\sigma))$



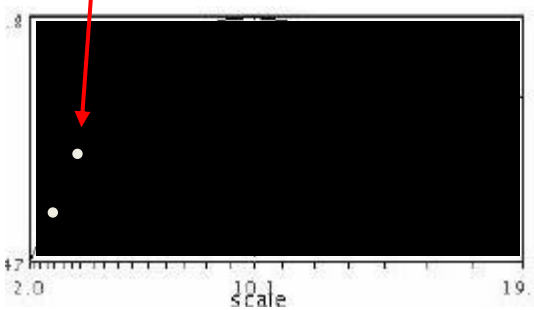
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



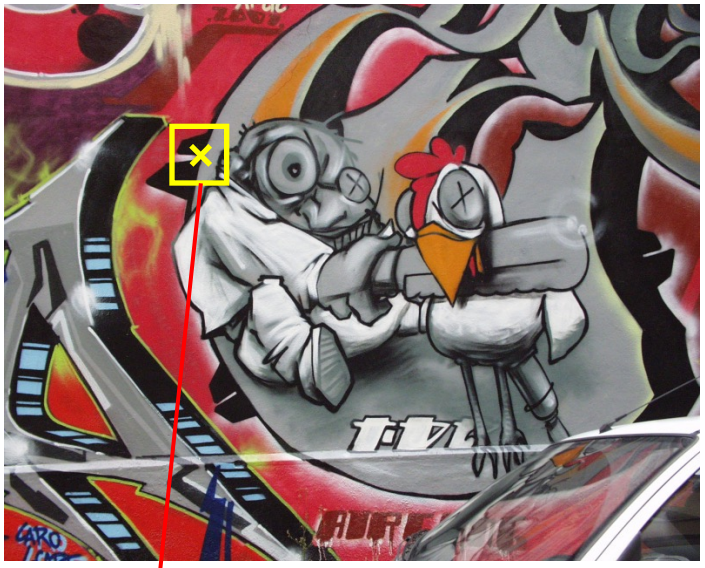
$f(I_{i_1 \ldots i_m}(x,\sigma))$

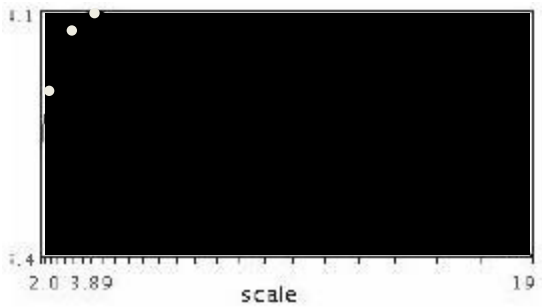


$f(I_{i_1 \ldots i_m}(x',\sigma))$

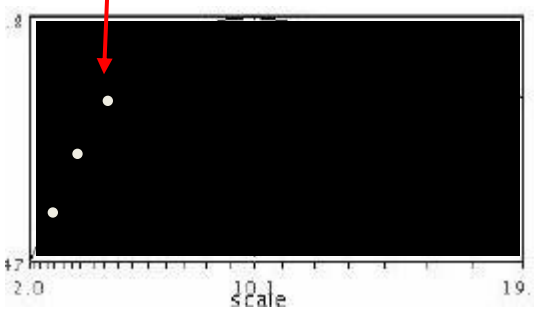
Automatic Scale Selection

- Function responses for increasing scale (scale signature)





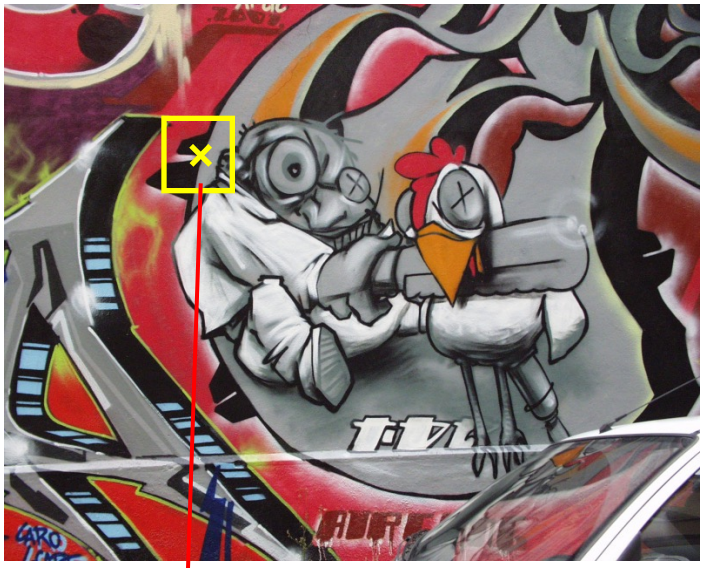
$f(I_{i_1 \ldots i_m}(x,\sigma))$

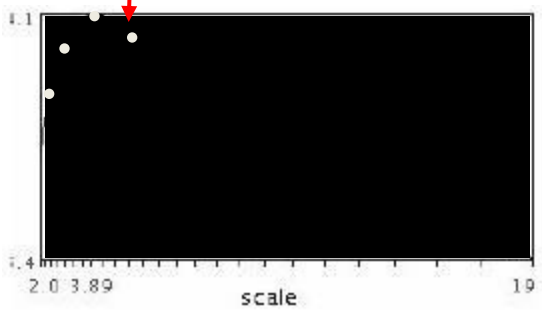


$f(I_{i_1 \ldots i_m}(x',\sigma))$

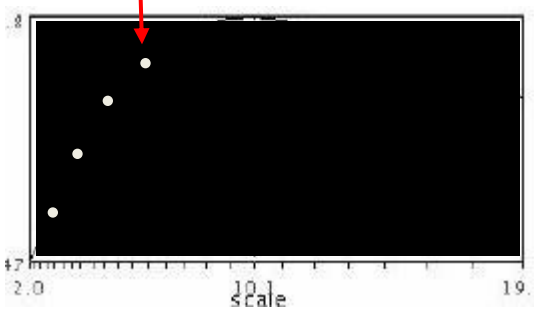
Automatic Scale Selection

- Function responses for increasing scale (scale signature)





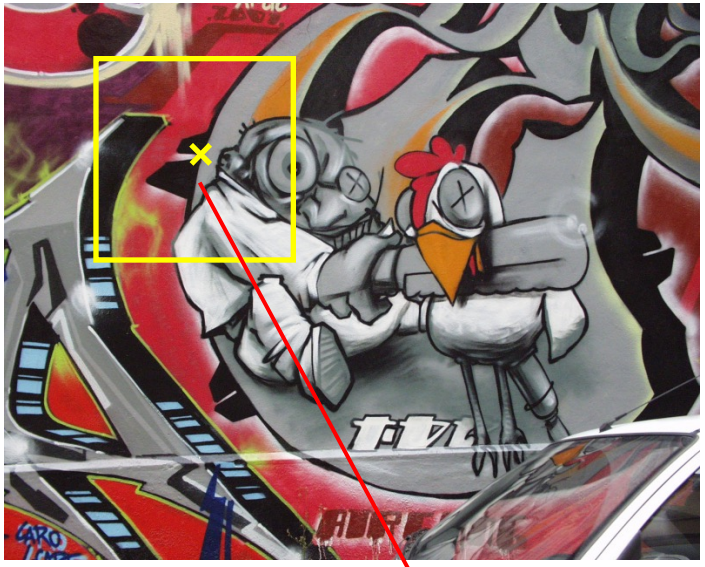
$f(I_{i_1 \ldots i_m}(x,\sigma))$

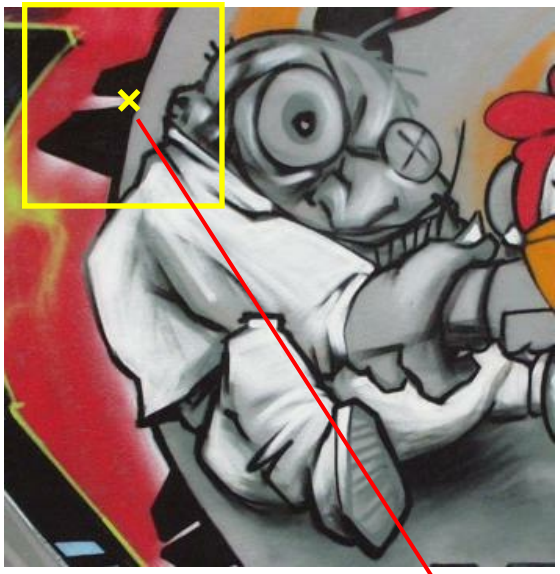


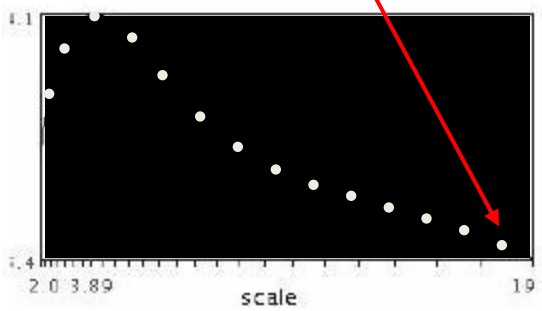
$f(I_{i_1 \ldots i_m}(x',\sigma))$

Automatic Scale Selection

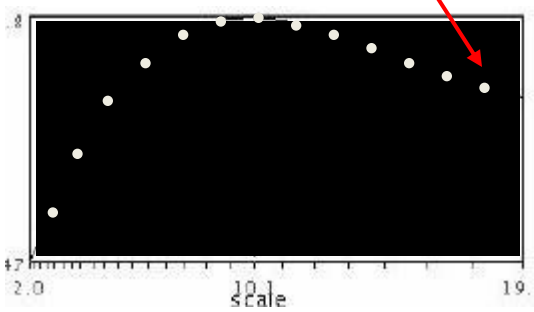
- Function responses for increasing scale (scale signature)







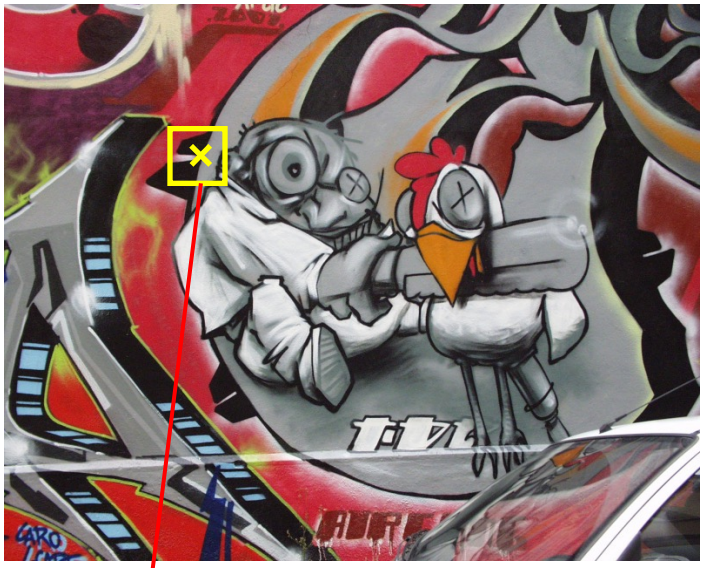
$f(I_{i_1 \ldots i_m}(x,\sigma))$

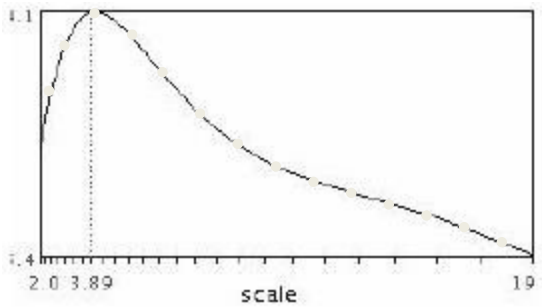


$f(I_{i_1 \ldots i_m}(x',\sigma))$

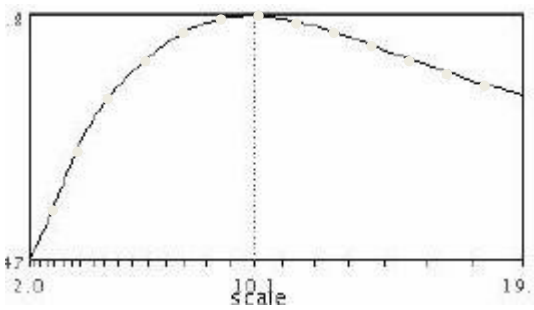
Automatic Scale Selection

- Function responses for increasing scale (scale signature)





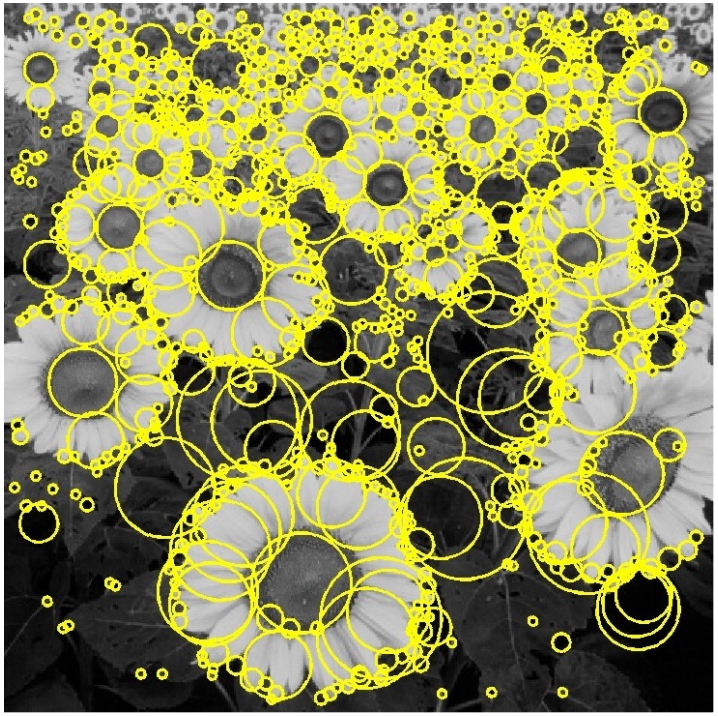
$f(I_{i_1 \ldots i_m}(x,\sigma))$

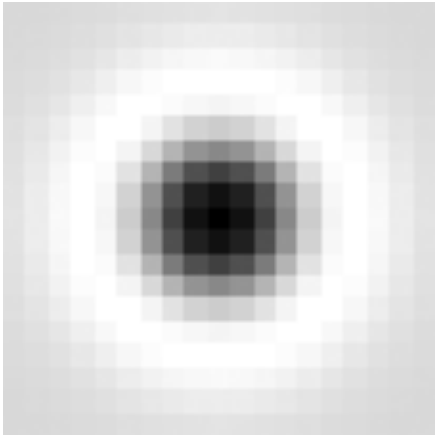


$f(I_{i_1 \ldots i_m}(x',\sigma'))$

Basic idea

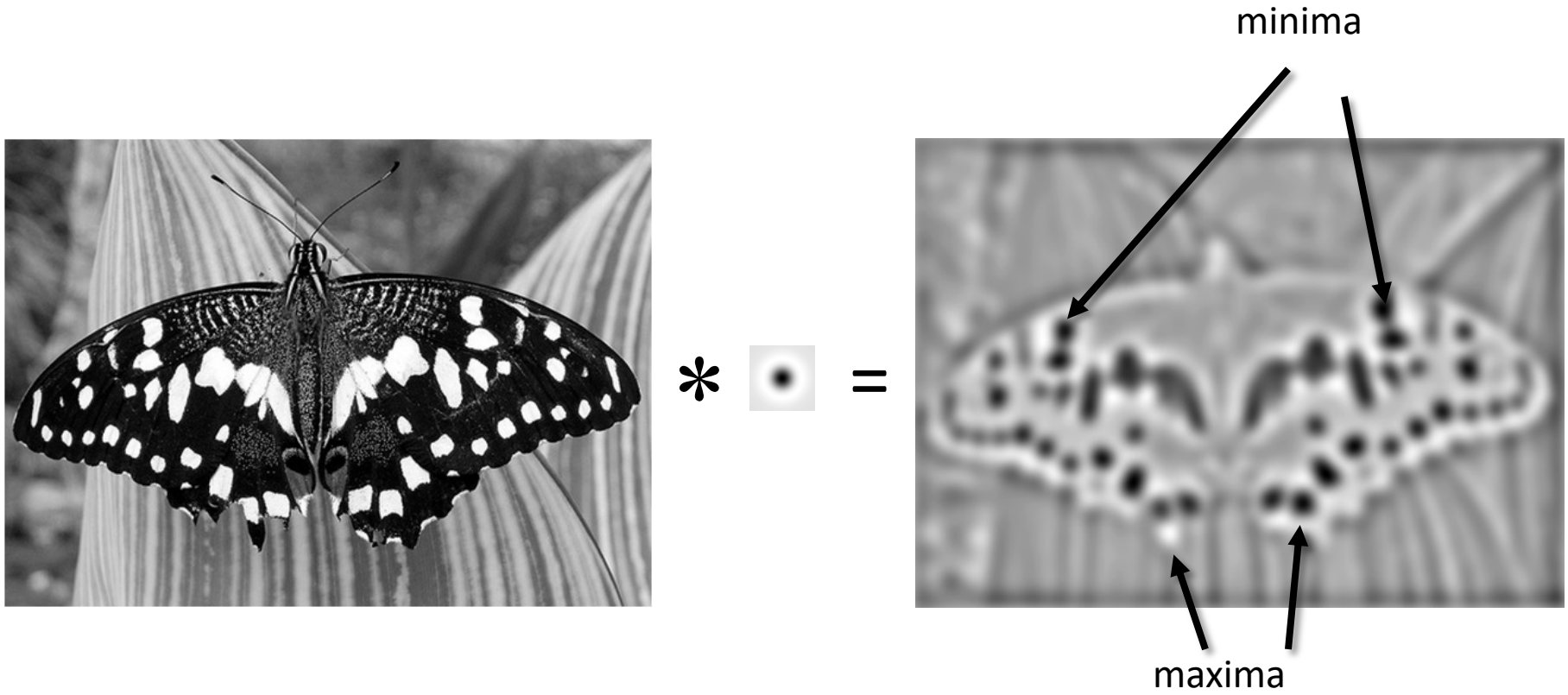
- Convolve the image with a "blob filter" at multiple scales and look for extrema of filter response in the resulting *scale space*





T. Lindeberg, Feature detection with automatic scale selection, *IJCV* 30(2), pp 77-116, 1998

Blob detection

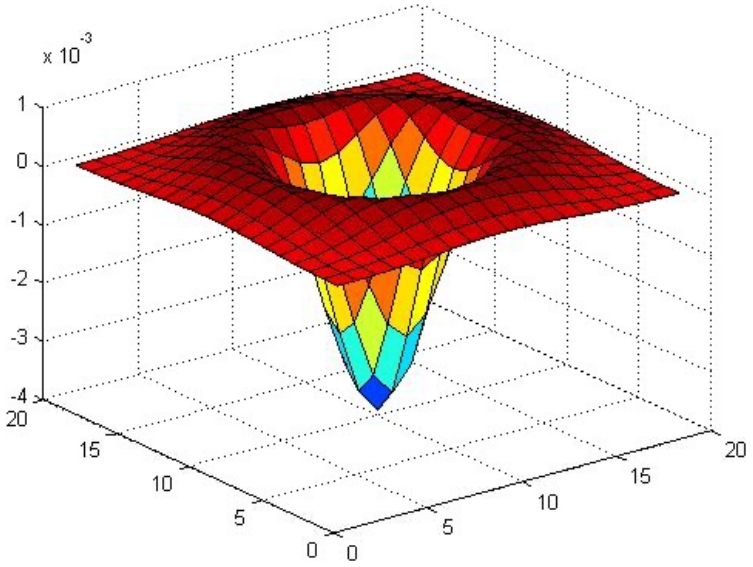


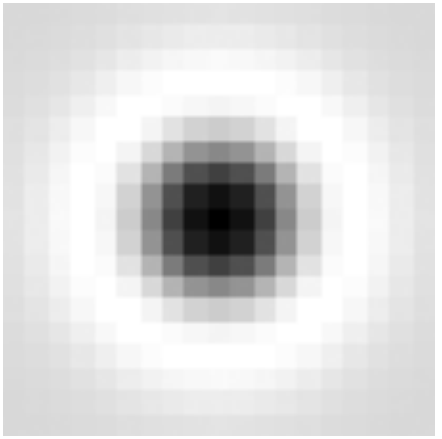
- Find maxima *and minima* of blob filter response in space *and scale*

Slide credit: N. Snavely

Blob detection in 2D

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





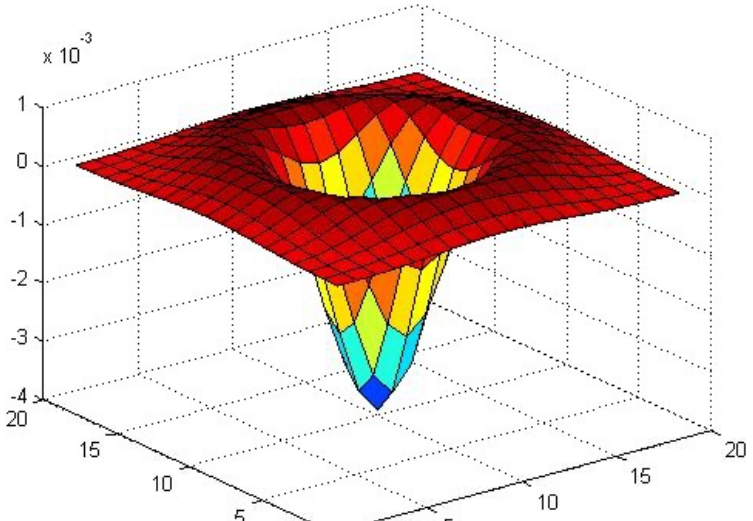
$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

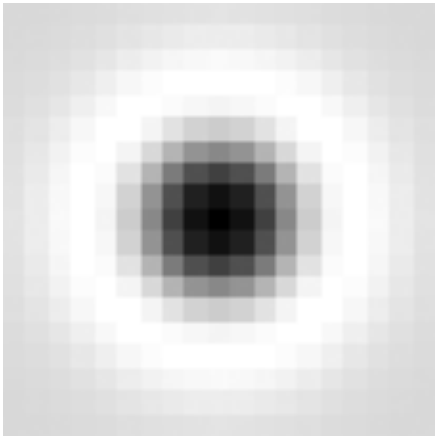
Blob detection in 2D

- Better performance: *Scale-normalized* Laplacian of Gaussian:

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

Lindeberg showed that the normalization of the Laplacian with the factor σ^2 is required for true scale invariance

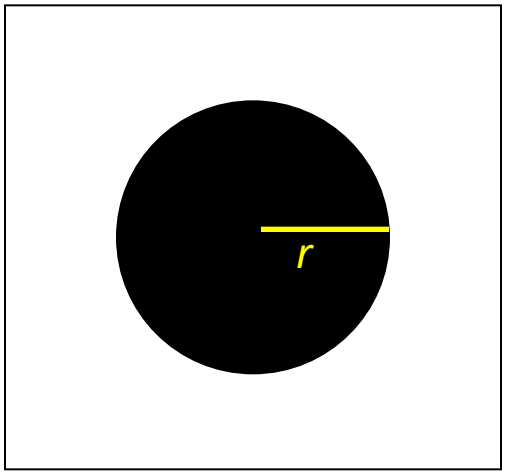




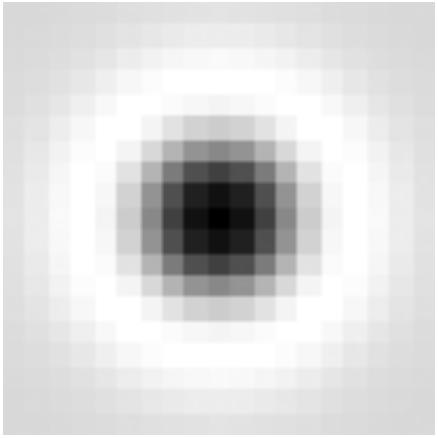
T. Lindeberg, Feature detection with automatic scale selection, *IJCV* 30(2), pp 77-116, 1998

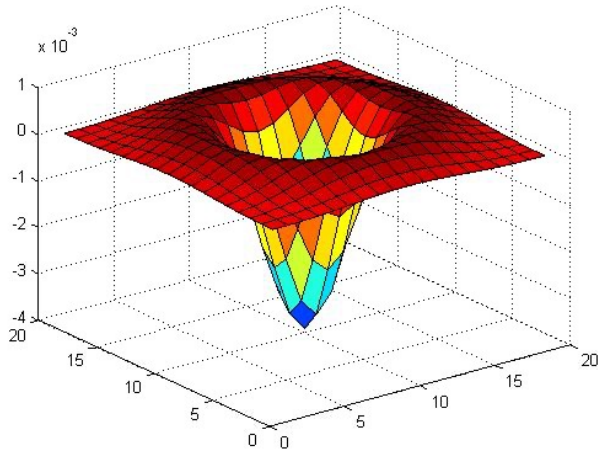
Blob detection in 2D

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



image





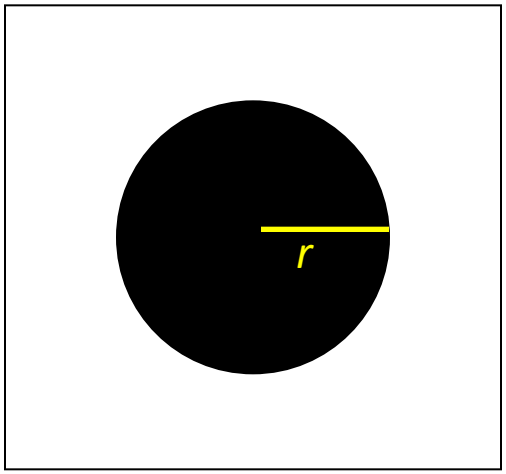
Laplacian

Blob detection in 2D

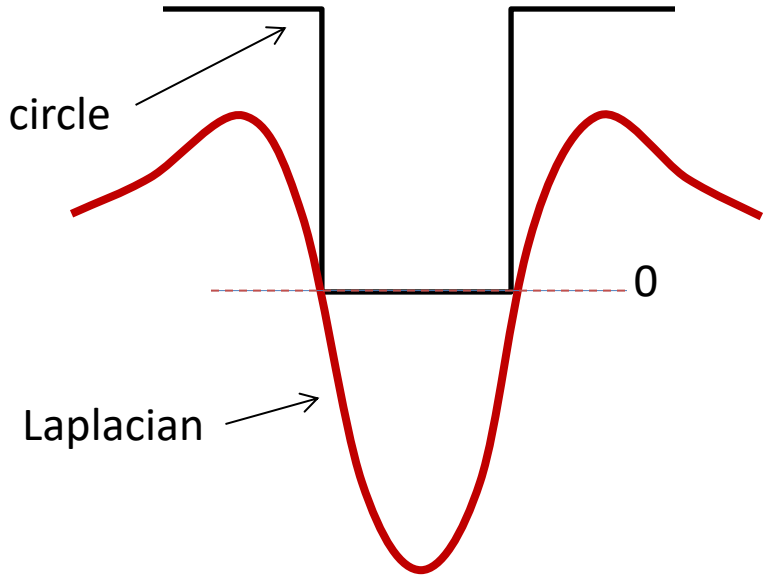
- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- To get maximum response, the zeros of the Laplacian have to be aligned with the circle
- The Laplacian is given by (up to scale):

$$(x^2 + y^2 - 2\sigma^2)\, e^{-(x^2+y^2)/2\sigma^2}$$

- Therefore, the maximum response occurs at $\sigma = r/\sqrt{2}.$

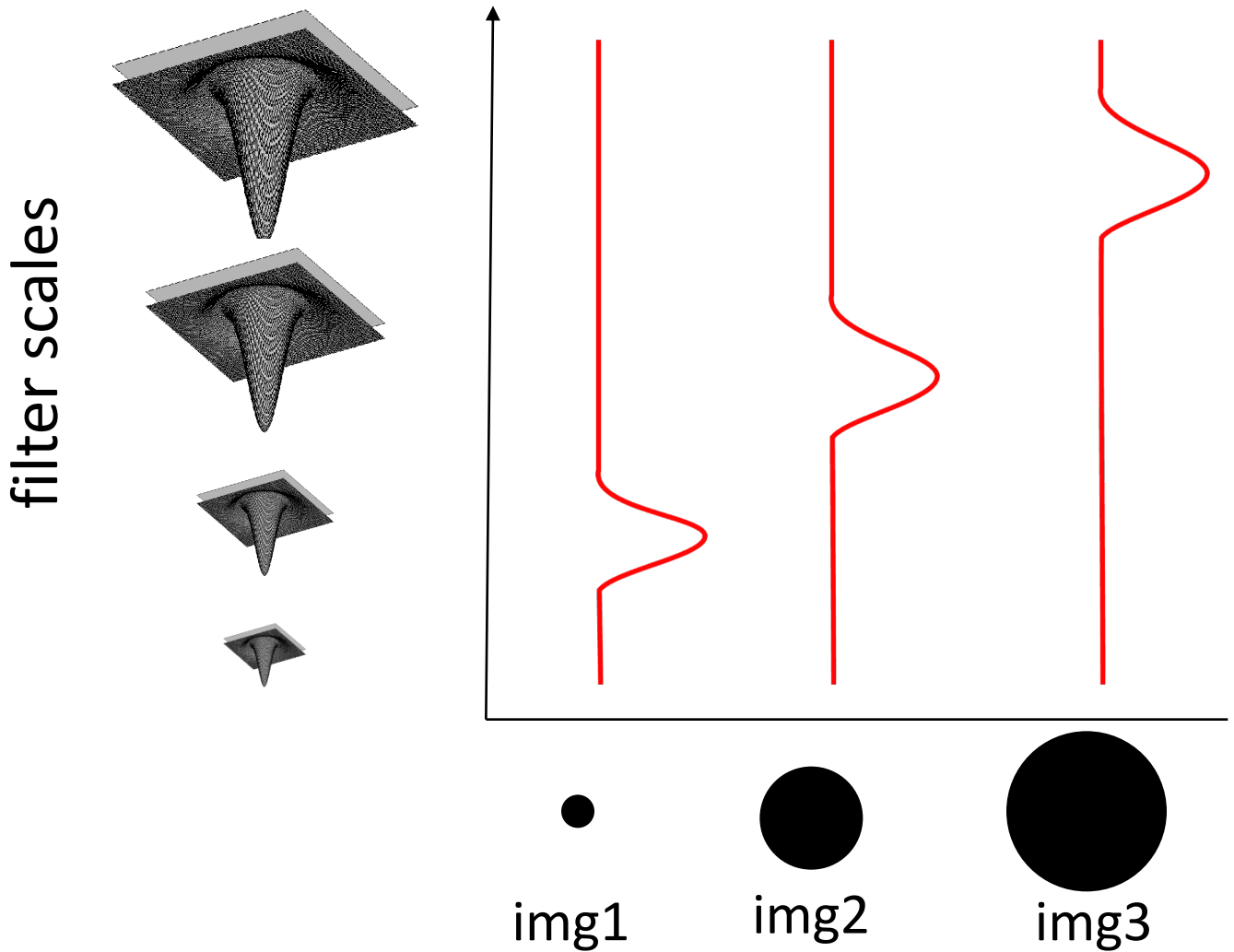


image



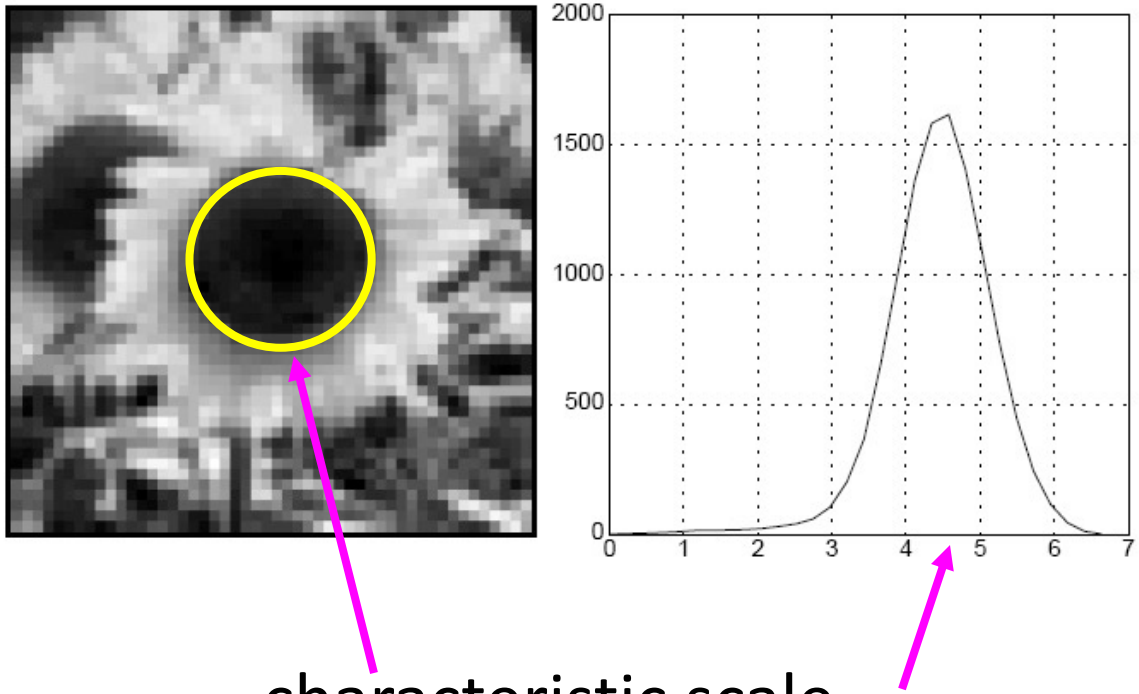
Blob detection in 2D: scale selection

- Laplacian-of-Gaussian = "blob" detector $\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$



Blob detection in 2D

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



Slide credit: Lana Lazebnik

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales

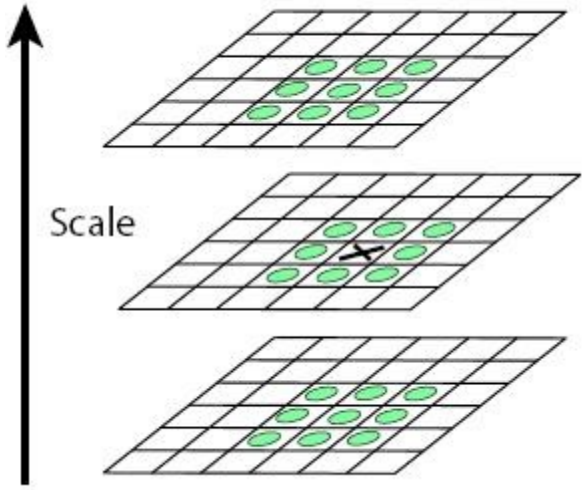
Scale-space blob detector: Example

Scale-space blob detector: Example

sigma = 11.9912

Scale-space blob detector

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



Efficient implementation

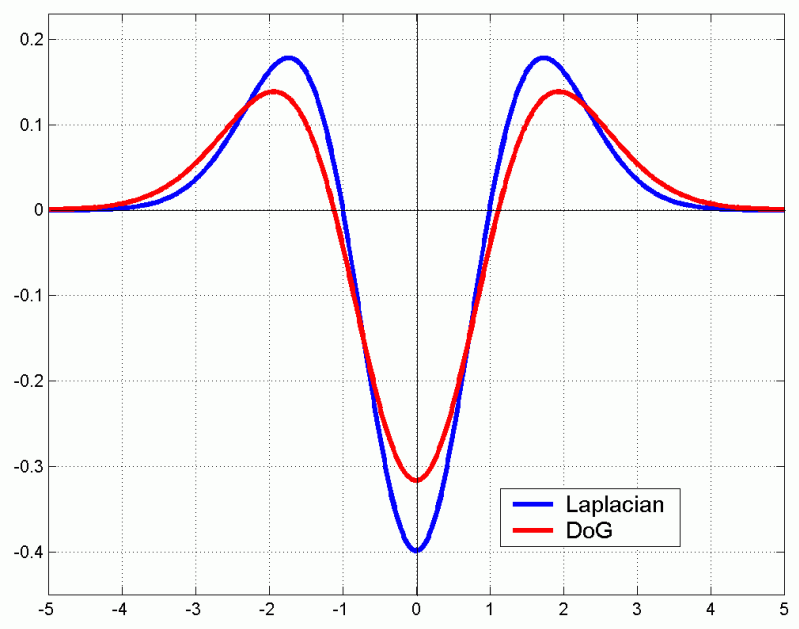
- Approximating the scale-normalized Laplacian with a difference of Gaussians:

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

(Scale normalized Laplacian)

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

(Difference of Gaussians)



Efficient implementation

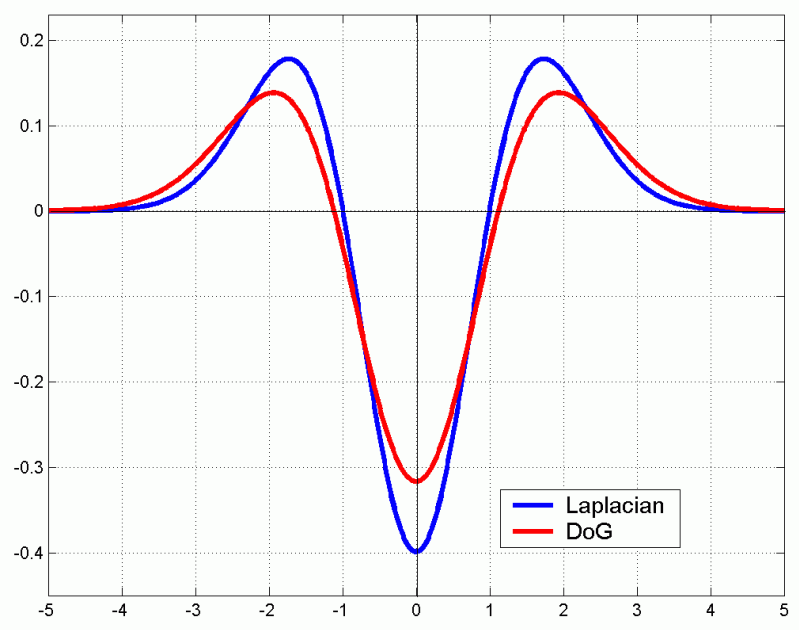
- Approximating the scale-normalized Laplacian with a difference of Gaussians:

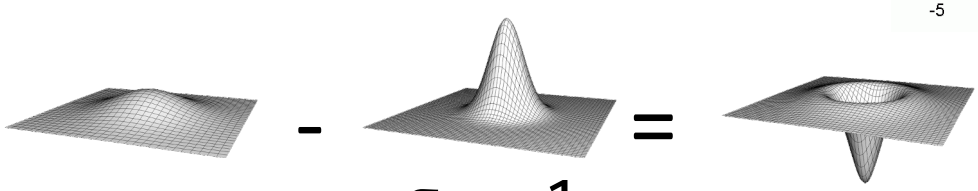
$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

(Scale normalized Laplacian)

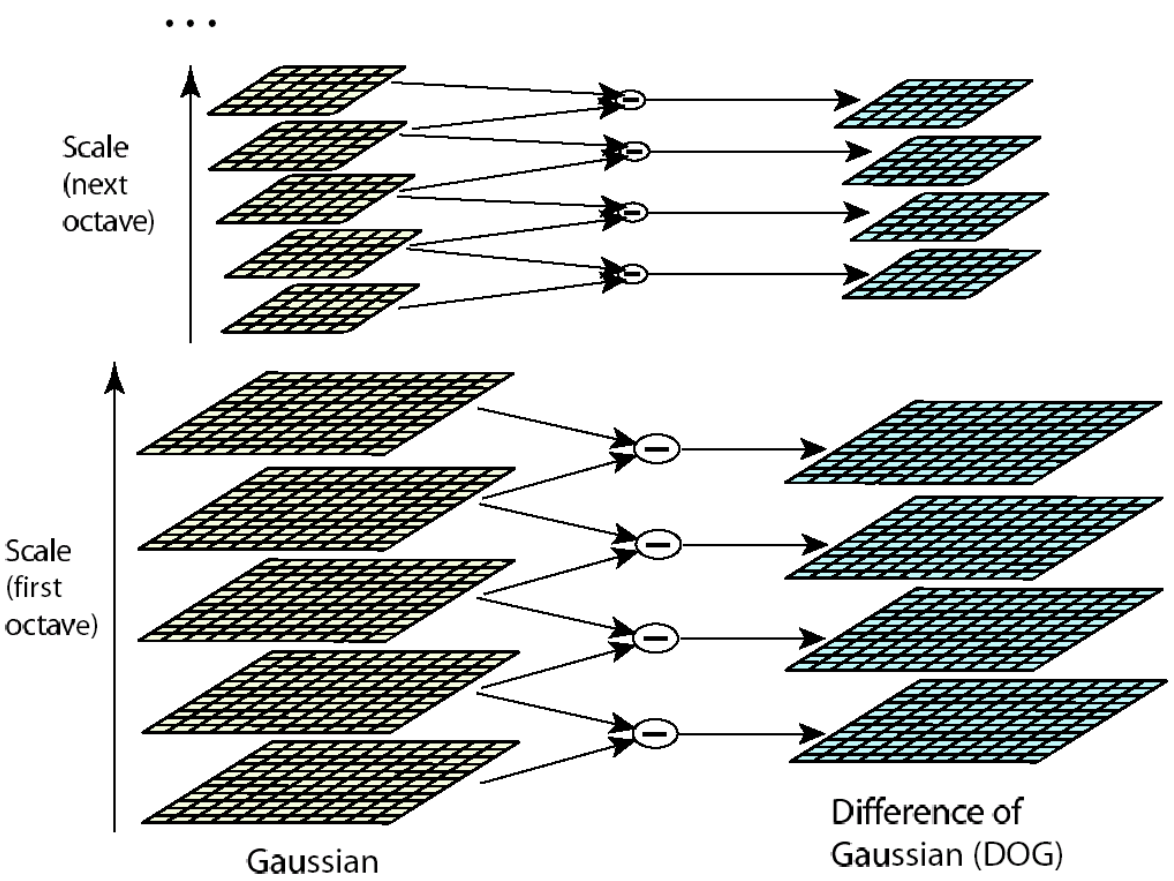
$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(Difference of Gaussians)





Efficient implementation



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

Example

Original image at
¾ the size

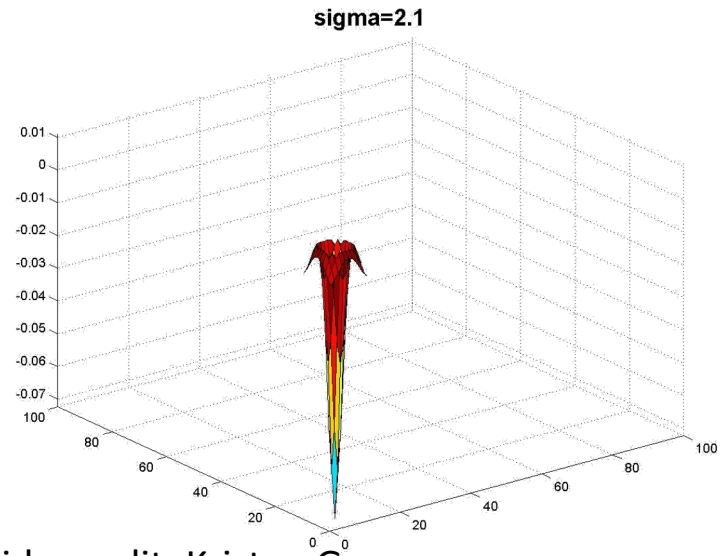
Slide credit: Kristen Grauman

Example

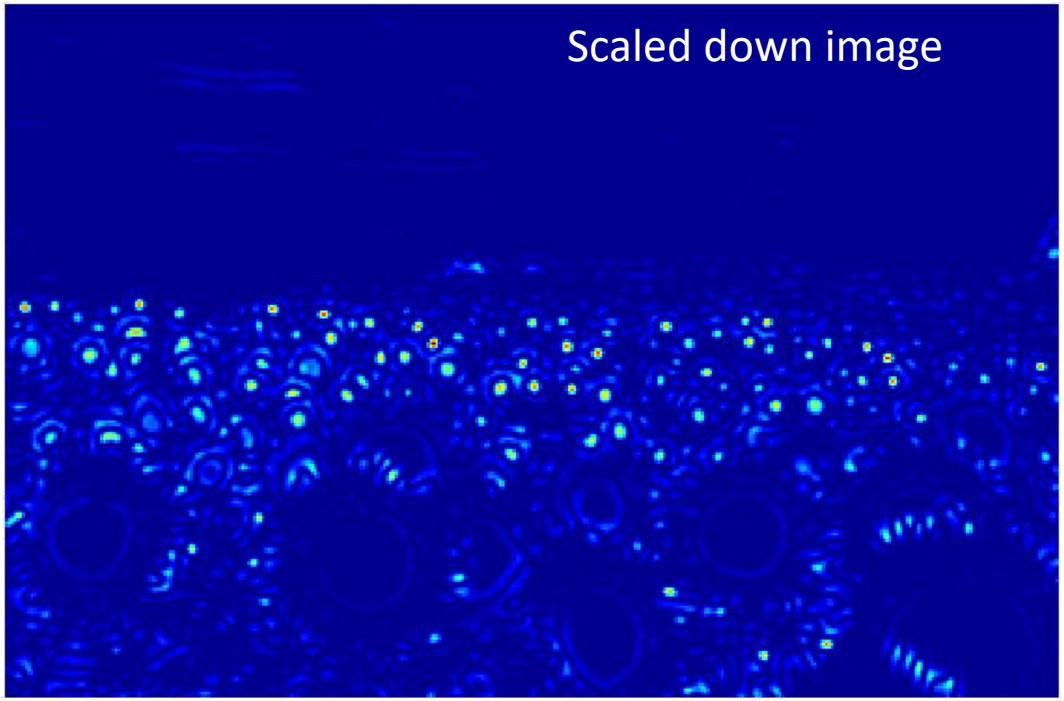
Original image at
¾ the size

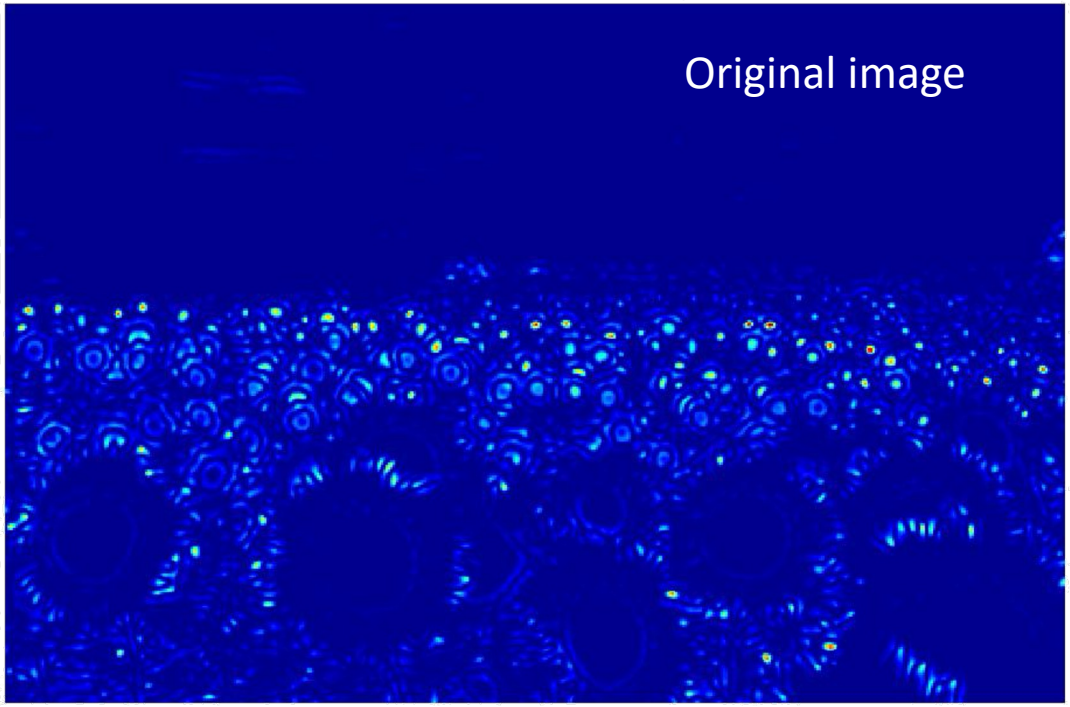
Slide credit: Kristen Grauman

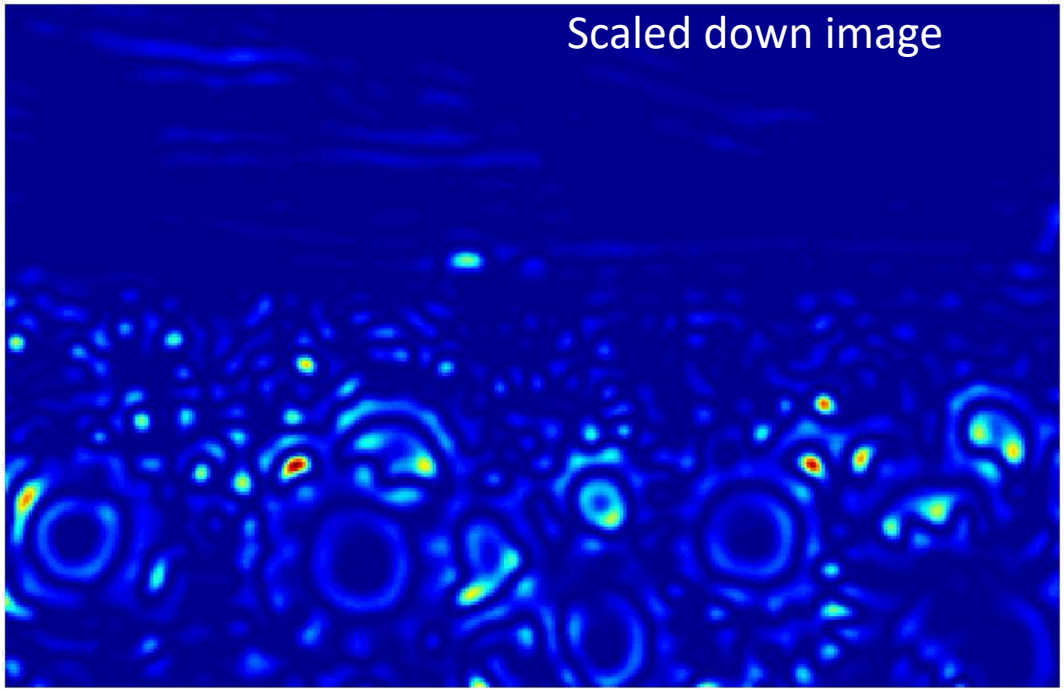
Original image at
¾ the size

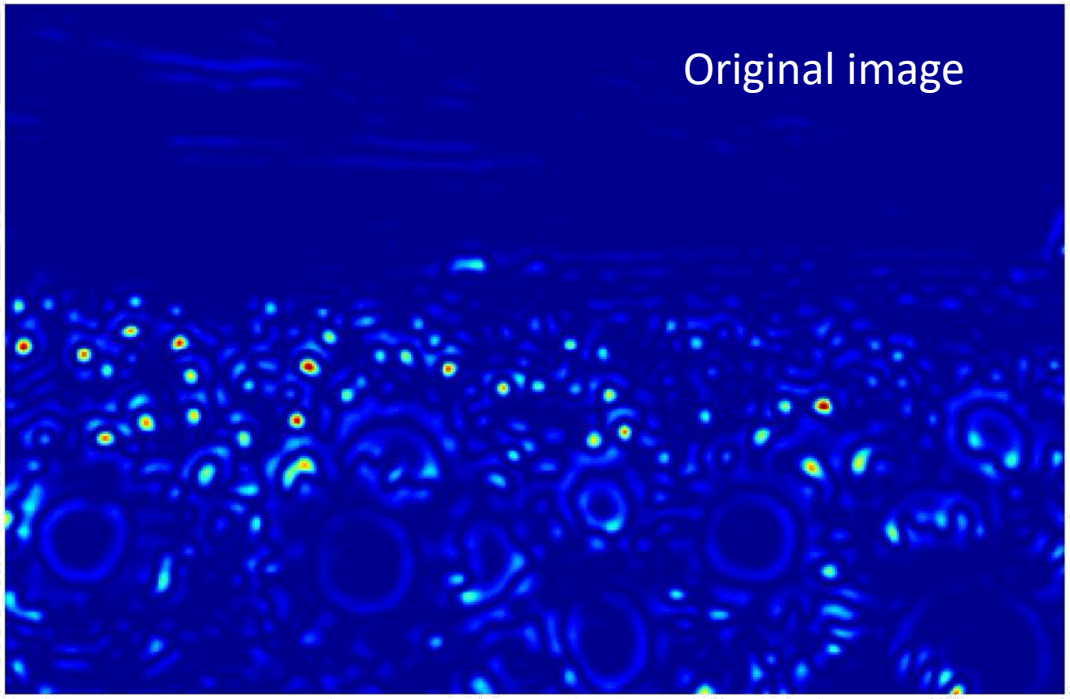


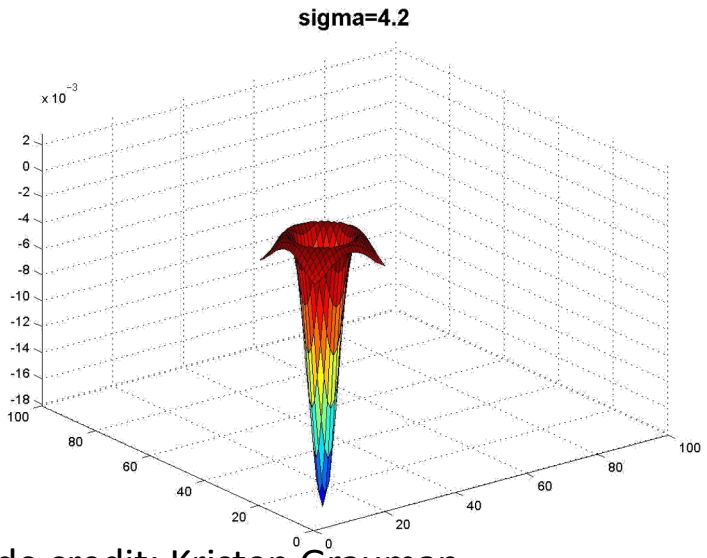
Slide credit: Kristen Grauman



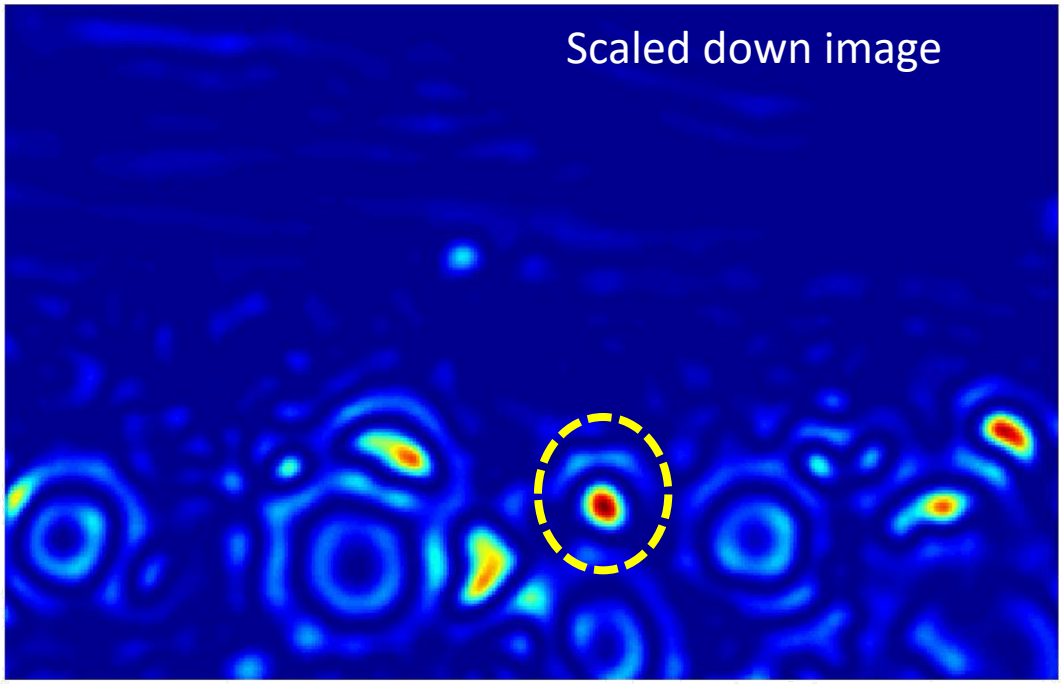


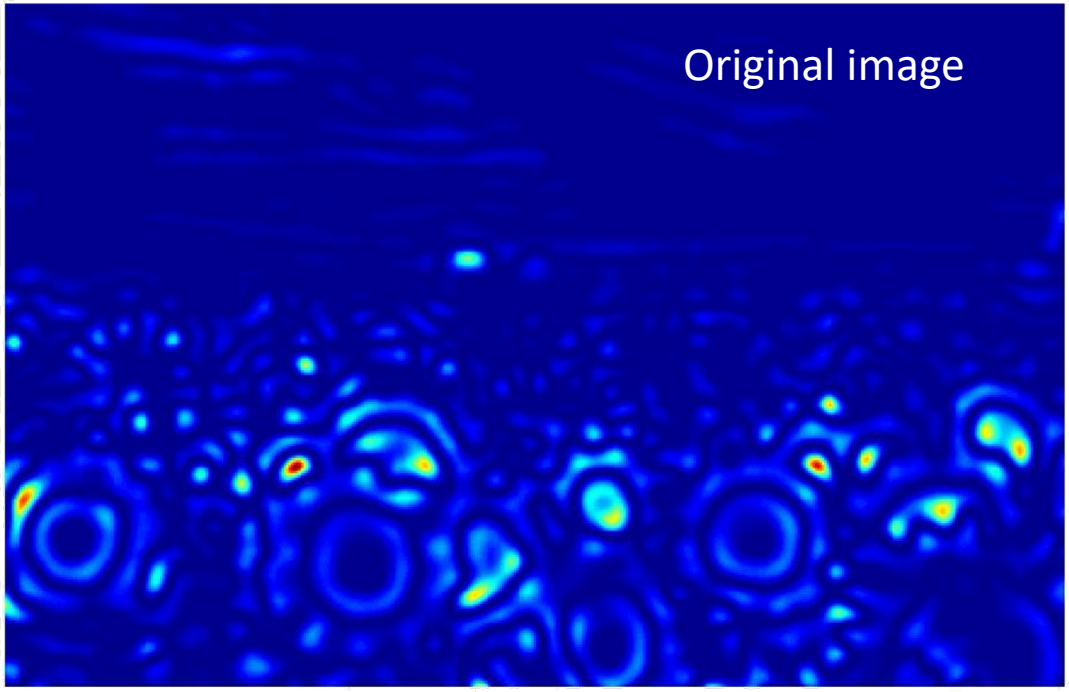


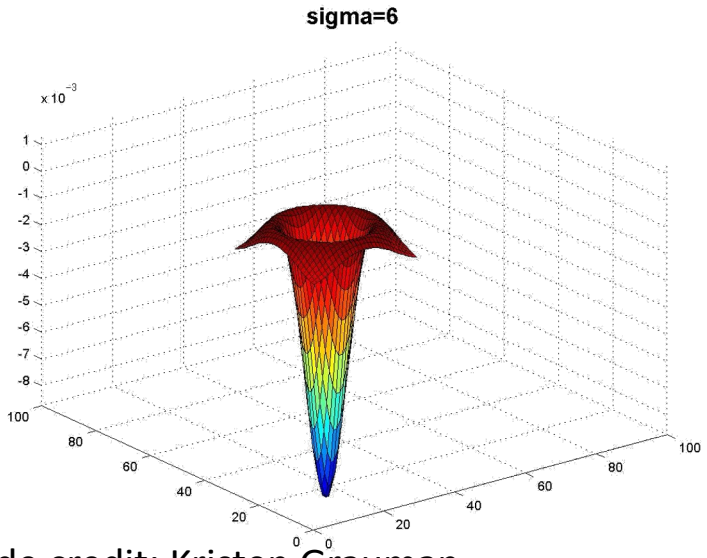




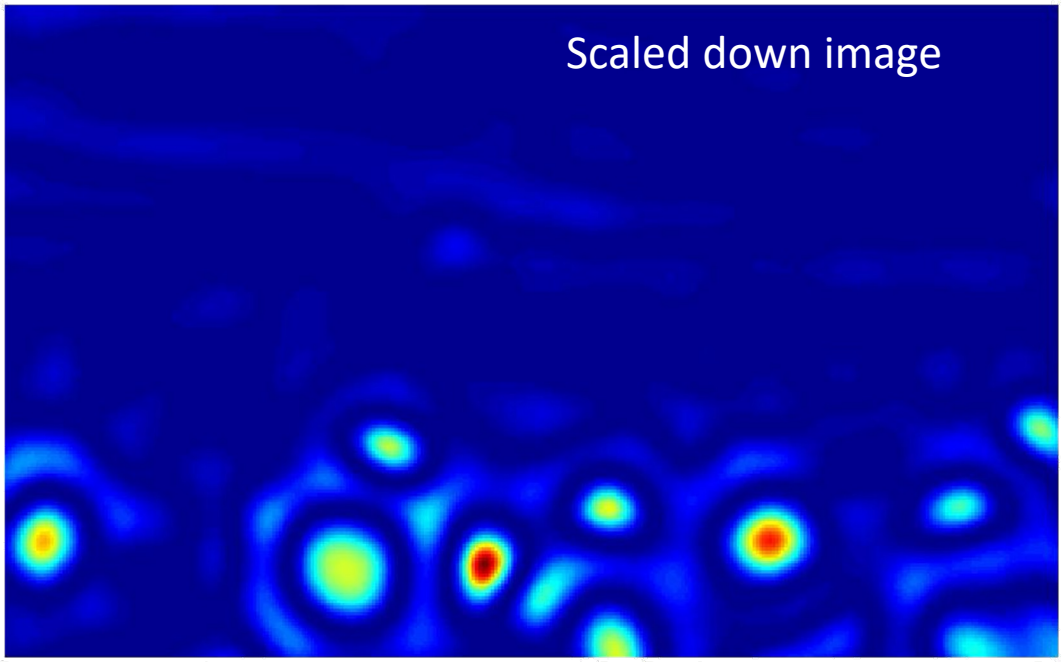
Slide credit: Kristen Grauman

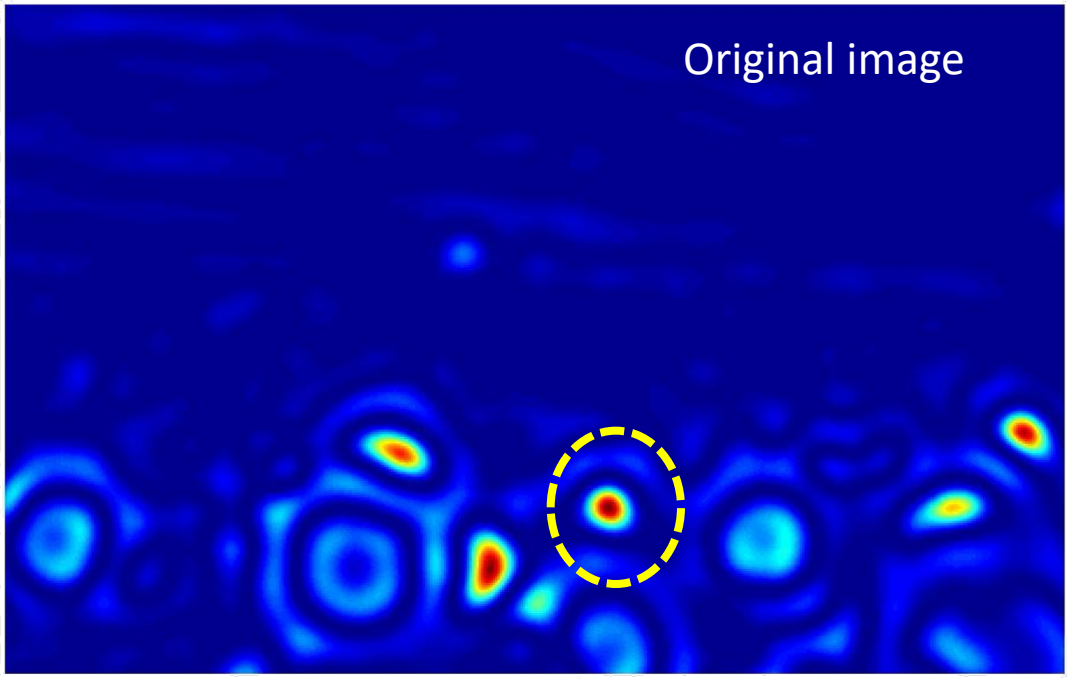


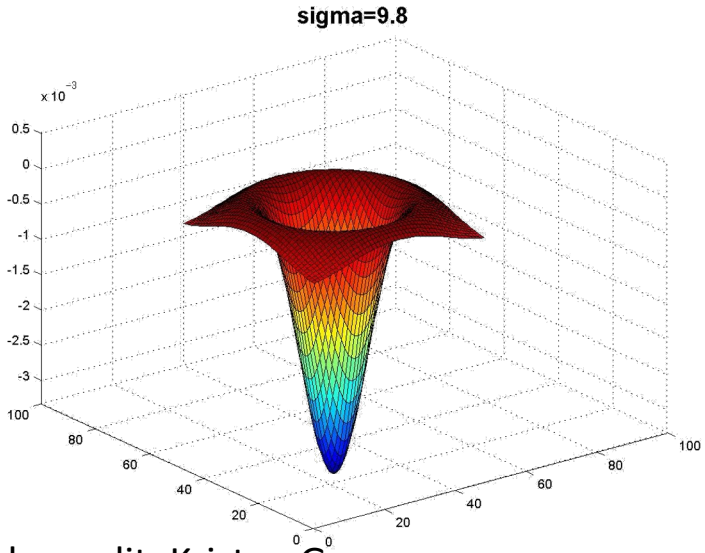




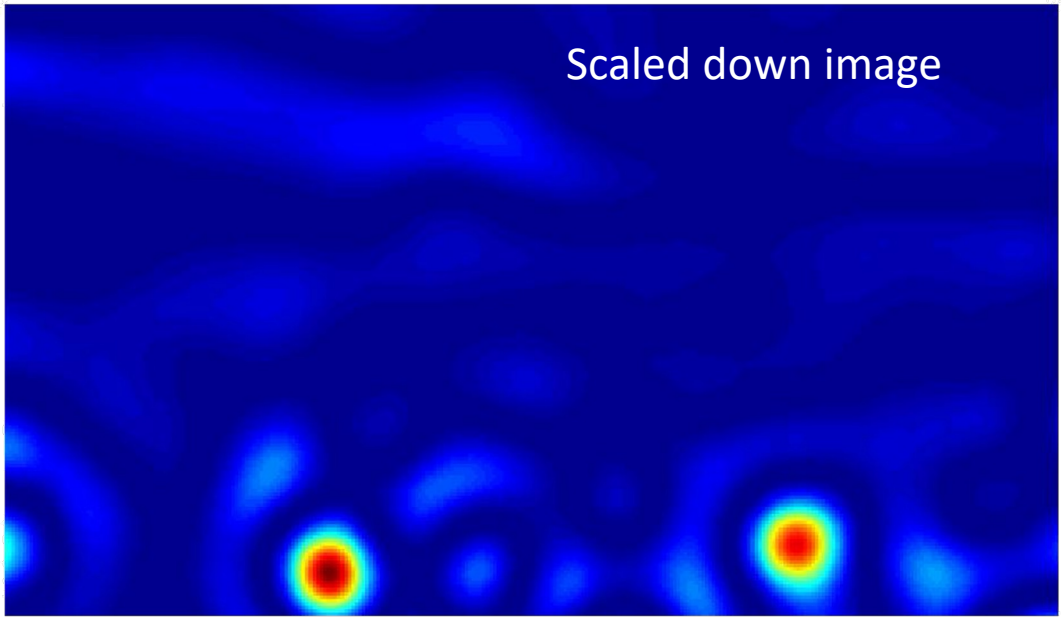
Slide credit: Kristen Grauman

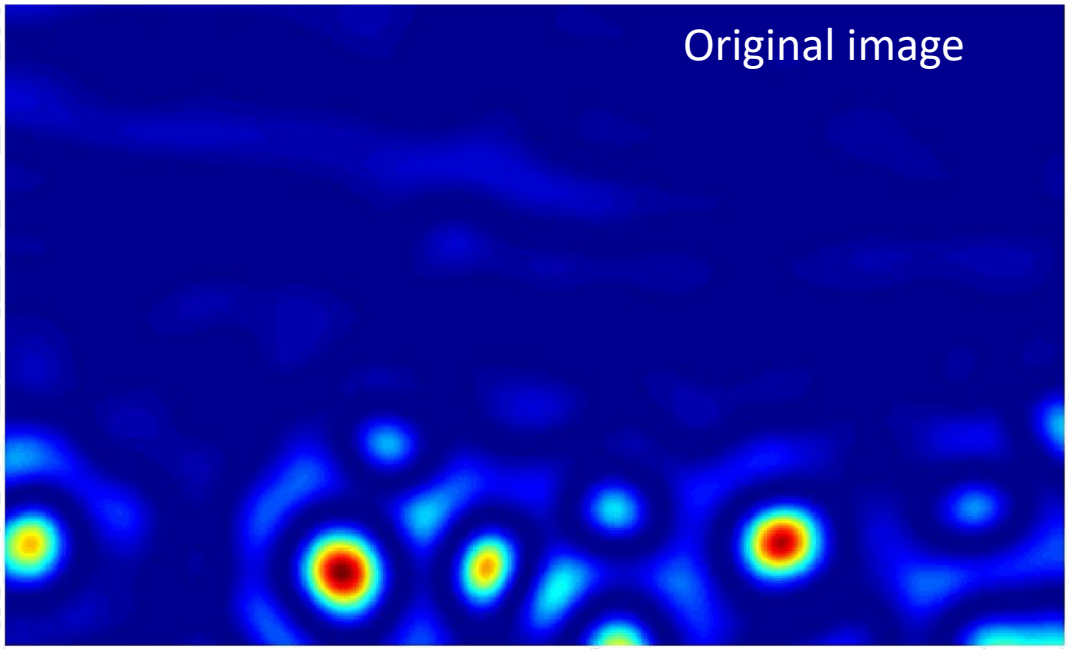


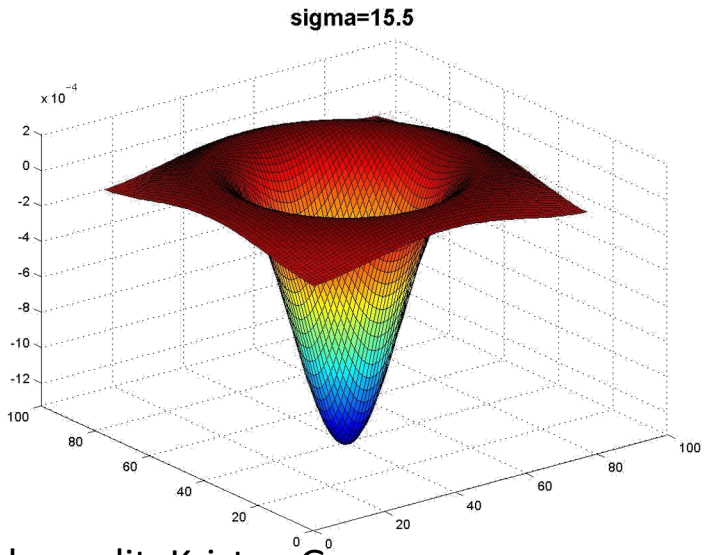




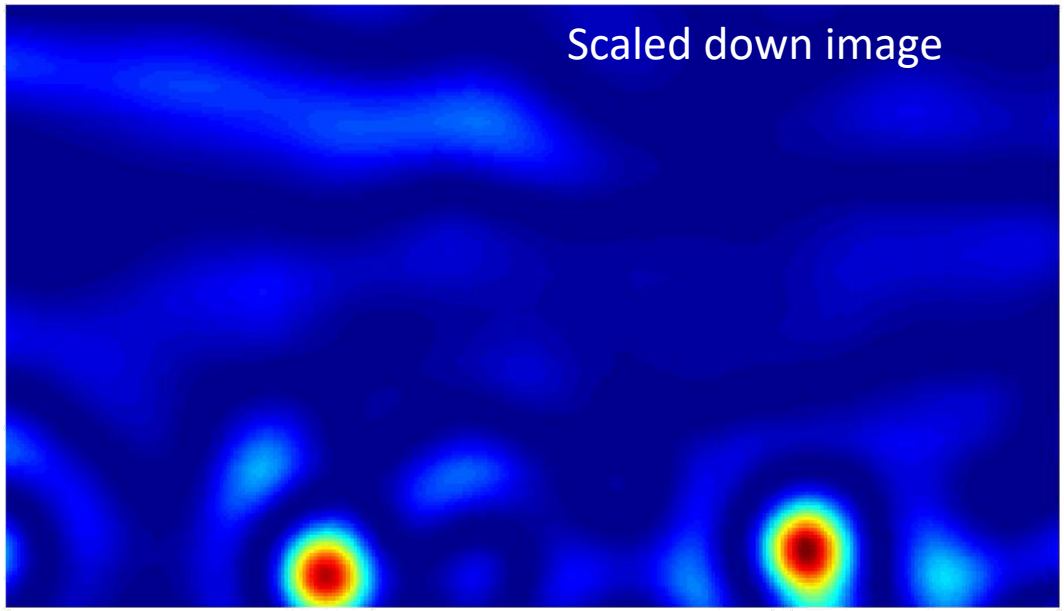
Slide credit: Kristen Grauman

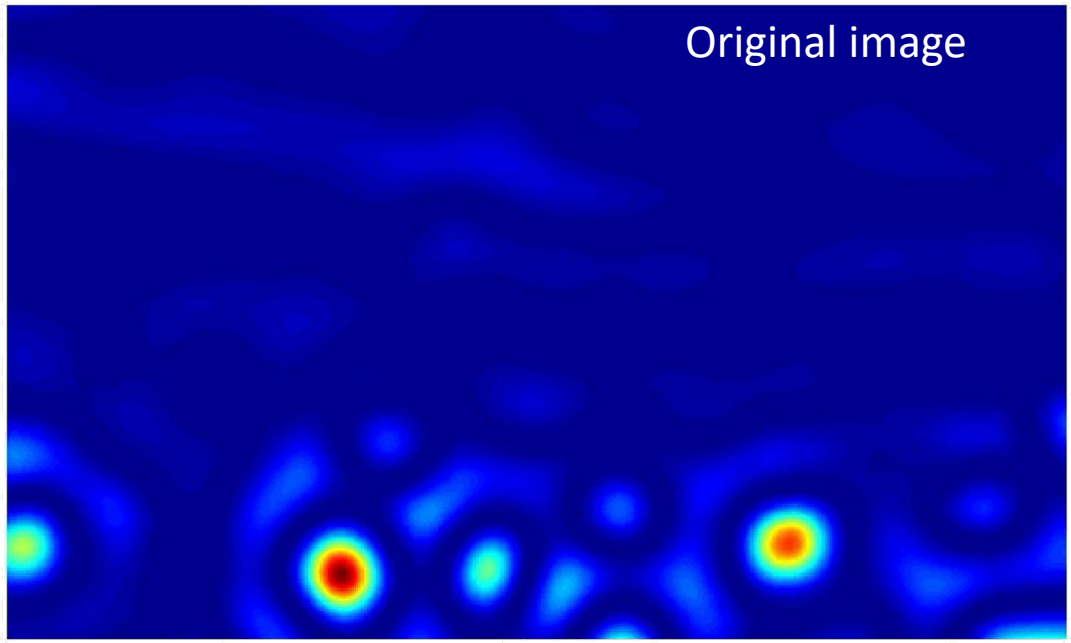


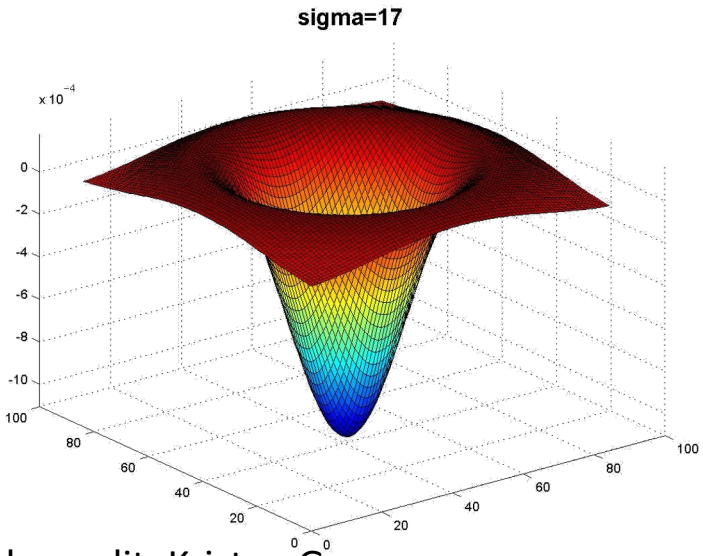




Slide credit: Kristen Grauman





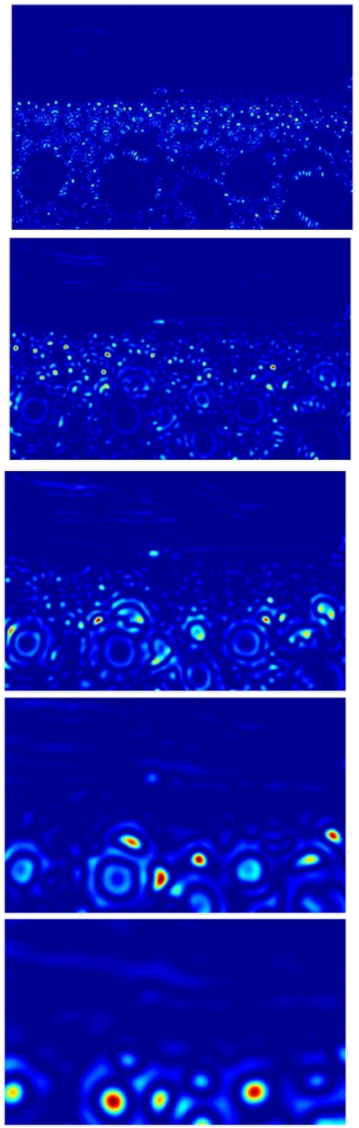


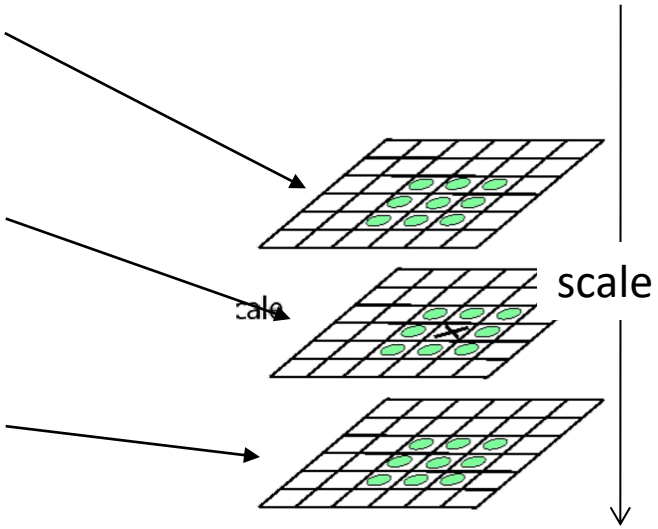
Slide credit: Kristen Grauman

Scale invariant interest points

Interest points are local maxima in both position and scale.

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





⇒ **List of**
(x, y, σ)

Squared filter response maps

Slide credit: Kristen Grauman

Comparison of Feature Detectors

Table 7.1 Overview of feature detectors.

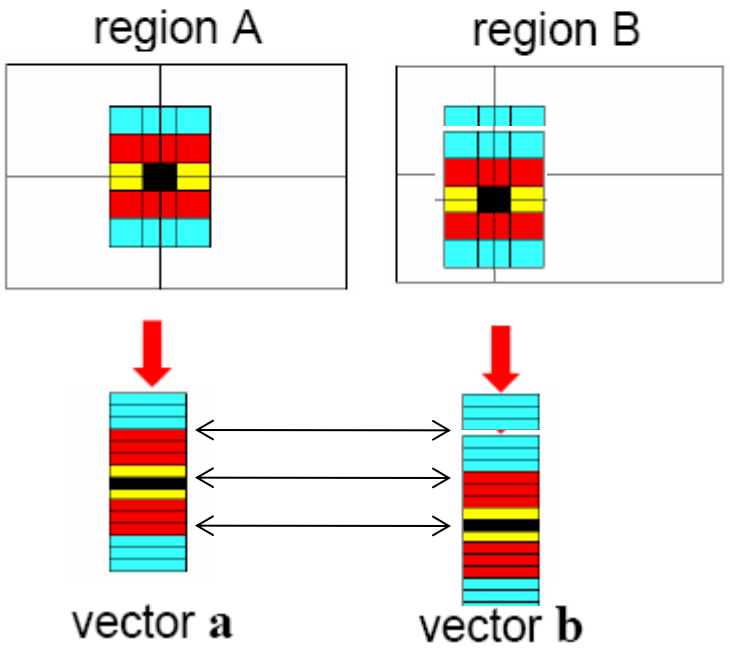
Feature Detector	Corner	Blob	Region	Rotation invariant	Scale invariant	Affine invariant	Repeatability	Localization accuracy	Robustness	Efficiency
Harris	√			√			+++	+++	+++	++
Hessian		√		√			++	++	++	+
SUSAN	√			√			++	++	++	+++
Harris-Laplace	√	(√)		√	√		+++	+++	++	+
Hessian-Laplace	(√)	√		√	√		+++	+++	+++	+
DoG	(√)	√		√	√		++	++	++	++
SURF	(√)	√		√	√		++	++	++	+++
Harris-Affine	√	(√)		√	√	√	+++	+++	++	++
Hessian-Affine	(√)	√		√	√	√	+++	+++	+++	++
Salient Regions	(√)	√		√	√	(√)	+	+	++	+
Edge-based	√			√	√	√	+++	+++	+	+
MSER			√	√	√	√	+++	+++	++	+++
Intensity-based			√	√	√	√	++	++	++	++
Superpixels			√	√	(√)	(√)	+	+	+	+

Tuytelaars Mikolajczyk 2008

Detecting local invariant features

- Detection of interest points
  - Harris corner detection
  - Scale invariant blob detection: LoG
- Description of local patches

Raw patches as local descriptors



The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.

Figure: Andrew Zisserman

Local Descriptors

- The ideal descriptor should be
  - Robust
  - Distinctive
  - Compact
  - Efficient

- Most available descriptors focus on edge/gradient information
  - Capture texture information
  - Color rarely used

David Lowe

FOLLOW

Professor Emeritus, Computer Science Dept., University of British Columbia
Verified email at cs.ubc.ca - Homepage

Computer Vision Object Recognition

TITLE	CITED BY	YEAR
Distinctive image features from scale-invariant keypoints DG Lowe International journal of computer vision 60 (2), 91-110	74793	2004
Object recognition from local scale-invariant features DG Lowe International Conference on Computer Vision, 1999, 1150-1157	25383	1999
Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. M Muja, DG Lowe VISAPP (1) 2, 331-340	4164	2009
Automatic panoramic image stitching using invariant features M Brown, DG Lowe International Journal of Computer Vision 74 (1), 59-73	3554	2007
Unsupervised learning of depth and ego-motion from video T Zhou, M Brown, N Snavely, DG Lowe Proceedings of the IEEE Conference on Computer Vision and Pattern …	2795	2017
Perceptual Organization and Visual Recognition DG Lowe Kluwer Academic Publishers, Boston	2181 *	1985
Three-dimensional object recognition from single two-dimensional images	2108	1987

Distinctive Image Features from Scale-Invariant Keypoints

Abstract

This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from many images. This paper also describes an approach to using these features for object recognition. The recognition proceeds by matching individual features to a database of features from known objects using a fast nearest-neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through least-squares solution for consistent pose parameters. This approach to recognition can robustly identify objects among clutter and occlusion while achieving near real-time performance.

http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

SIFT properties

- Invariant to
  - Scale
  - Rotation

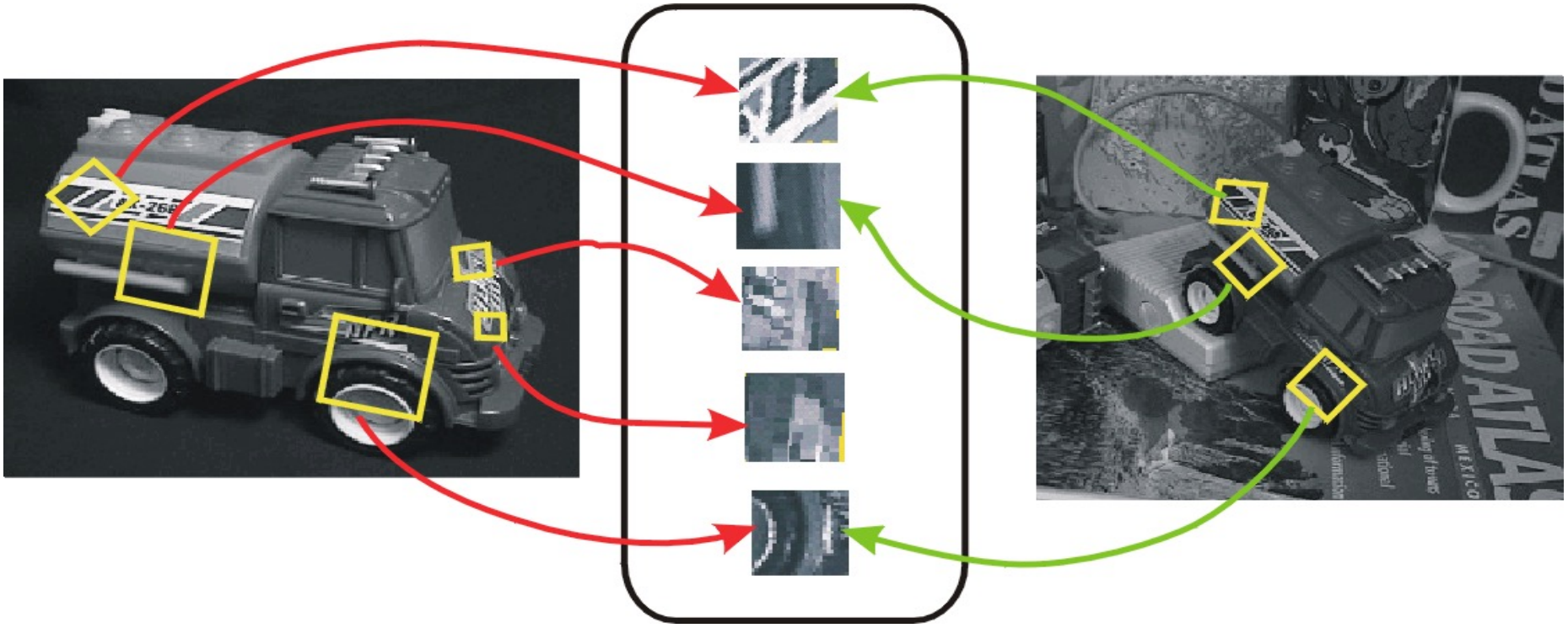
- Partially invariant to
  - Illumination changes
  - Camera viewpoint
  - Occlusion, clutter

Major stages for computing SIFT features

1. **Scale-space extrema detection:** The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.
2. **Keypoint localization:** At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.
3. **Orientation assignment:** One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.
4. **Keypoint descriptor:** The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

This approach has been named the Scale Invariant Feature Transform (SIFT), as it transforms image data into scale-invariant coordinates relative to local features.

From keypoint detection to feature description



- Detection is *covariant*:

  features(transform(image)) = transform(features(image))

- Description is *invariant*:

  features(transform(image)) = features(image)

Details of Lowe's SIFT algorithm

- Run DoG detector
  - Find maxima in location/scale space

Lowe IJCV 2004

Scale space using DoG

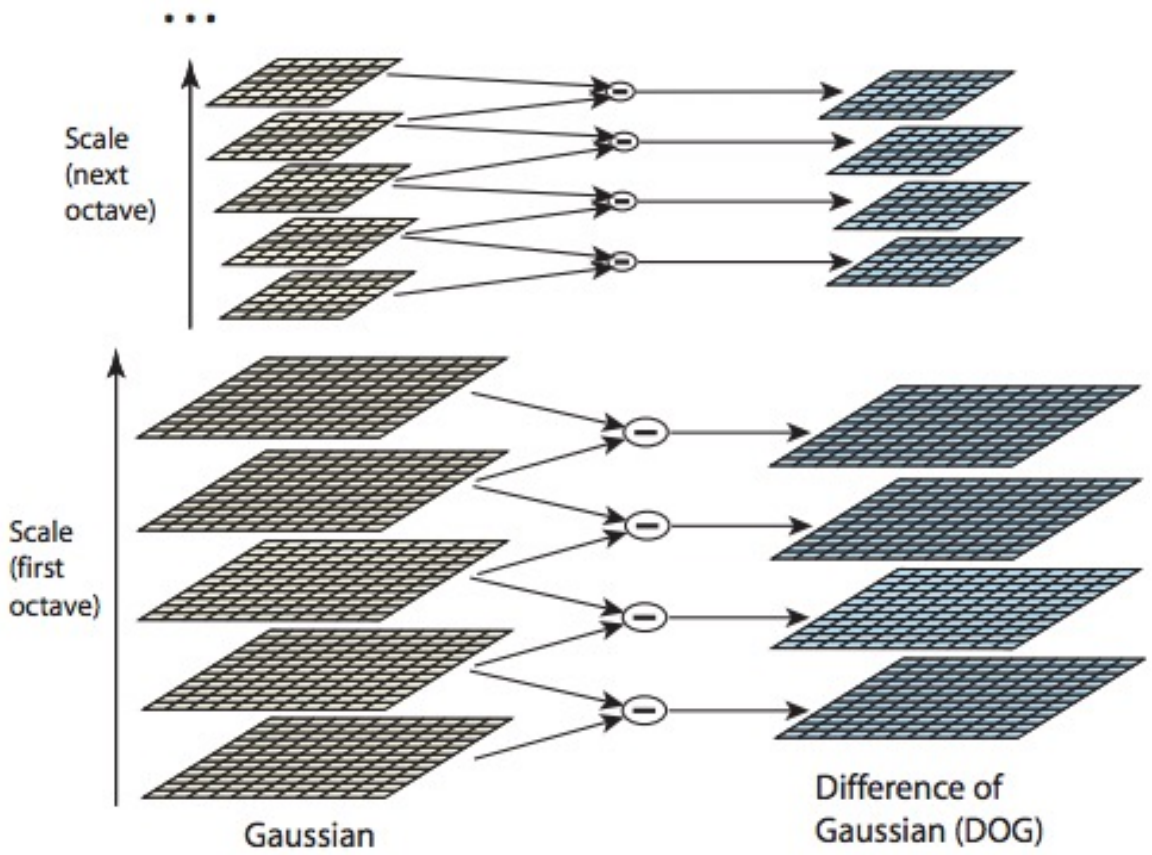
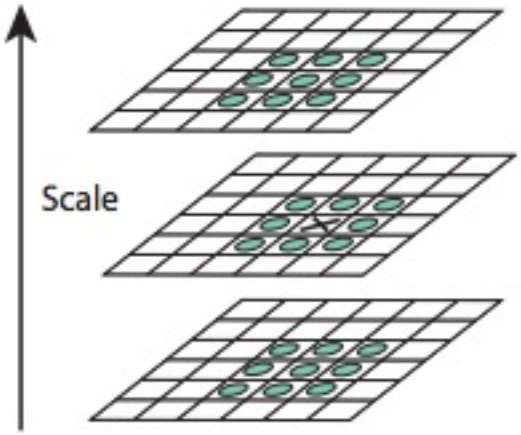


Figure 1: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.



Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in the 3x3 regions at the current and adjacent scales (marked with circles)

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$
$$= L(x, y, k\sigma) - L(x, y, \sigma).$$

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k - 1)\sigma^2 \nabla^2 G.$$

Details of Lowe's SIFT algorithm

- Run DoG detector
  - Find maxima in location/scale space
  - Remove low contrast + edge points

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

4.1 Eliminating edge responses

For stability, it is not sufficient to reject keypoints with low contrast. The difference-of-Gaussian function will have a strong response along edges, even if the location along the edge is poorly determined and therefore unstable to small amounts of noise.

A poorly defined peak in the difference-of-Gaussian function will have a large principal curvature across the edge but a small one in the perpendicular direction. The principal curvatures can be computed from a 2x2 Hessian matrix, **H**, computed at the location and scale of the keypoint:

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \quad (4)$$

The derivatives are estimated by taking differences of neighboring sample points.

The eigenvalues of **H** are proportional to the principal curvatures of D. Borrowing from the approach used by Harris and Stephens (1988), we can avoid explicitly computing the eigenvalues, as we are only concerned with their ratio. Let α be the eigenvalue with the largest magnitude and β be the smaller one. Then, we can compute the sum of the eigenvalues from the trace of **H** and their product from the determinant:

$$\mathrm{Tr}(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$
$$\mathrm{Det}(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

Lowe IJCV 2004

Details of Lowe's SIFT algorithm

- Run DoG detector
  - Find maxima in location/scale space
  - Remove low contrast + edge points

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

In the unlikely event that the determinant is negative, the curvatures have different signs so the point is discarded as not being an extremum. Let r be the ratio between the largest magnitude eigenvalue and the smaller one, so that $\alpha = r\beta$. Then,

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r},$$

which depends only on the ratio of the eigenvalues rather than their individual values. The quantity $(r+1)^2/r$ is at a minimum when the two eigenvalues are equal and it increases with r. Therefore, to check that the ratio of principal curvatures is below some threshold, r, we only need to check

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}.$$

This is very efficient to compute, with less than 20 floating point operations required to test each keypoint. The experiments in this paper use a value of $r = 10$, which eliminates keypoints that have a ratio between the principal curvatures greater than 10. The transition

Lowe IJCV 2004

Detecting keypoints with DoG

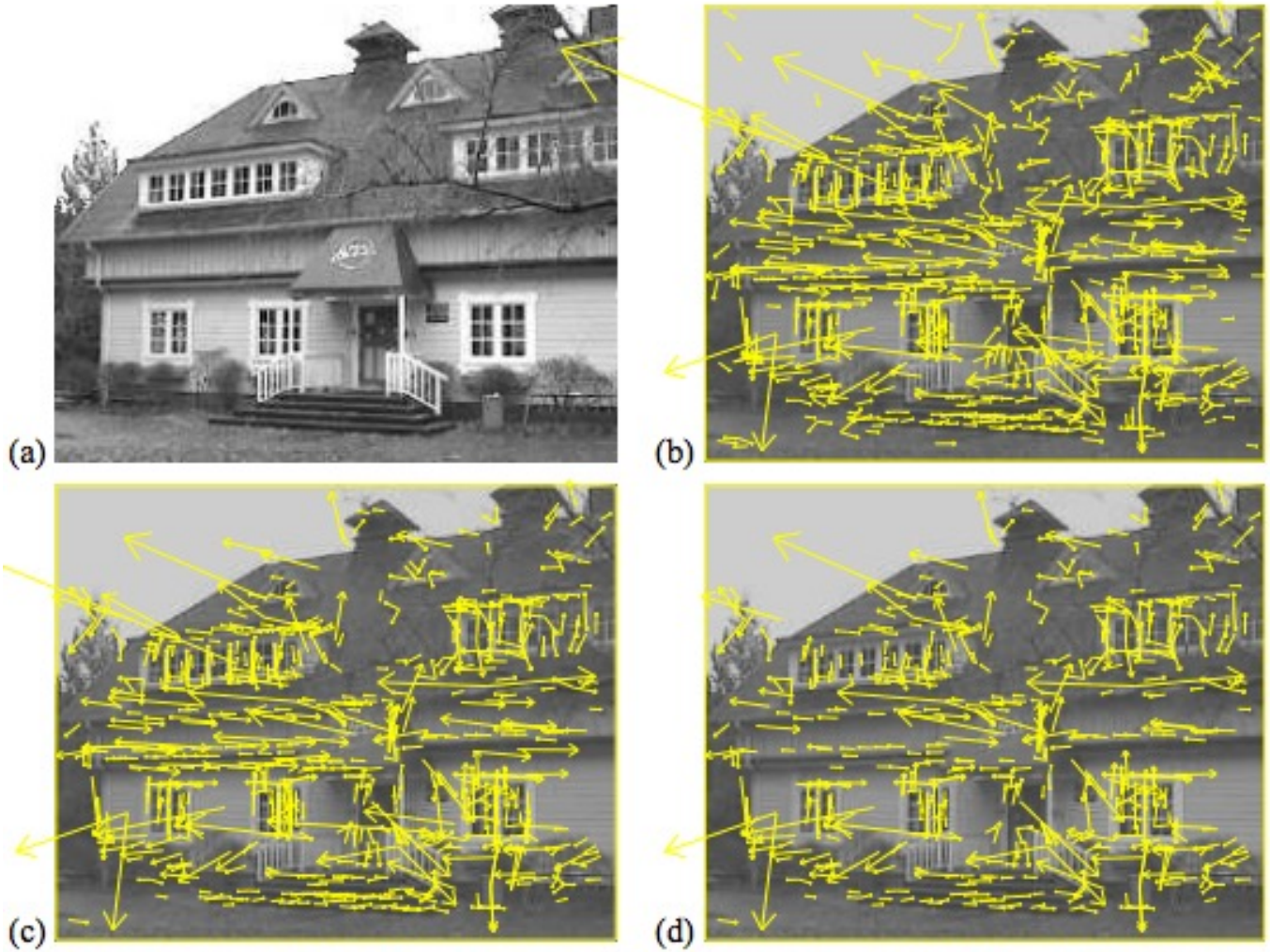


Figure 5: This figure shows the stages of keypoint selection. (a) The 233x189 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain following an additional threshold on ratio of principal curvatures.

Details of Lowe's SIFT algorithm

- Run DoG detector
  - Find maxima in location/scale space
  - Remove low contrast + edge points

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

- Find all major orientations
  - Bin orientations into 36 bin histogram
    - Weight by gradient magnitude
    - Weight by distance to center (Gaussian-weighted mean)
  - Return orientations within 0.8 of peak
    - Use parabola for better orientation fit

Lowe IJCV 2004

Following experimentation with a number of approaches to assigning a local orientation, the following approach was found to give the most stable results. The scale of the keypoint is used to select the Gaussian smoothed image, L, with the closest scale, so that all computations are performed in a scale-invariant manner. For each image sample, $L(x, y)$, at this scale, the gradient magnitude, $m(x, y)$, and orientation, $\theta(x, y)$, is precomputed using pixel differences:

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

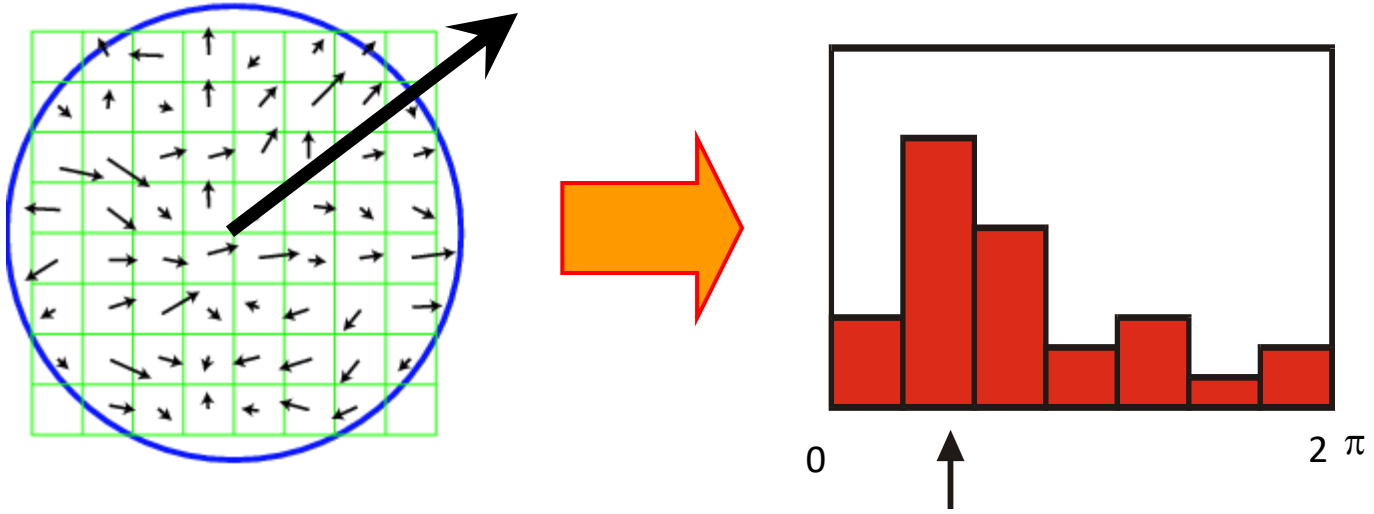
$$\theta(x, y) = \tan^{-1}((L(x, y+1) - L(x, y-1))/(L(x+1, y) - L(x-1, y)))$$

An orientation histogram is formed from the gradient orientations of sample points within a region around the keypoint. The orientation histogram has 36 bins covering the 360 degree range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the keypoint.

Peaks in the orientation histogram correspond to dominant directions of local gradients. The highest peak in the histogram is detected, and then any other local peak that is within 80% of the highest peak is used to also create a keypoint with that orientation. Therefore, for locations with multiple peaks of similar magnitude, there will be multiple keypoints created at the same location and scale but different orientations. Only about 15% of points are assigned multiple orientations, but these contribute significantly to the stability of matching. Finally, a parabola is fit to the 3 histogram values closest to each peak to interpolate the peak position for better accuracy.

Finding a reference orientation

- Create histogram of local gradient directions in the patch
- Assign reference orientation at peak of smoothed histogram



Details of Lowe's SIFT algorithm

- Run DoG detector
  - Find maxima in location/scale space
  - Remove low contrast + edge points

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

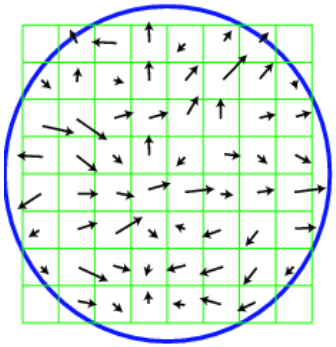
$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

- Find all major orientations
  - Bin orientations into 36 bin histogram
    - Weight by gradient magnitude
    - Weight by distance to center (Gaussian-weighted mean)
  - Return orientations within 0.8 of peak
    - Use parabola for better orientation fit
- For each (x, y, scale, orientation), create descriptor:
  - Sample 16×16 gradient magnitude and relative orientation
  - Bin 4×4 samples into 4×4 histograms
  - Threshold values to max of 0.2, divide by L2 norm
  - Final descriptor: $4 \times 4 \times 8$ normalized histograms = 128d feature vector

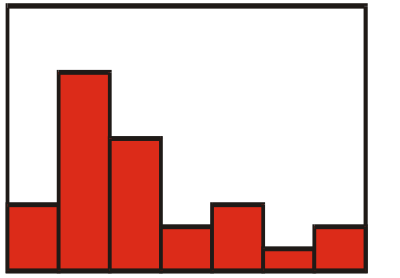
Lowe IJCV 2004

SIFT descriptor

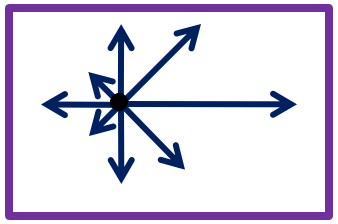
- Use histograms to bin pixels within sub-patches according to their orientation.

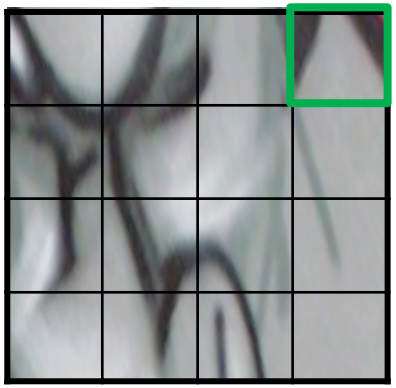


gradients

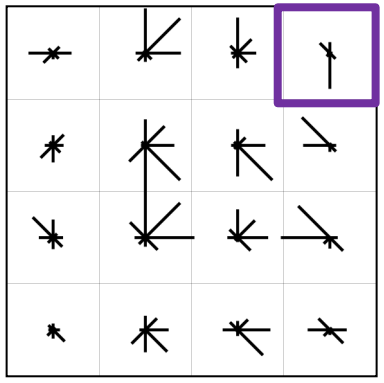


binned by orientation





subdivided local patch



histogram per grid cell

Final descriptor = concatenation of all histograms, normalize

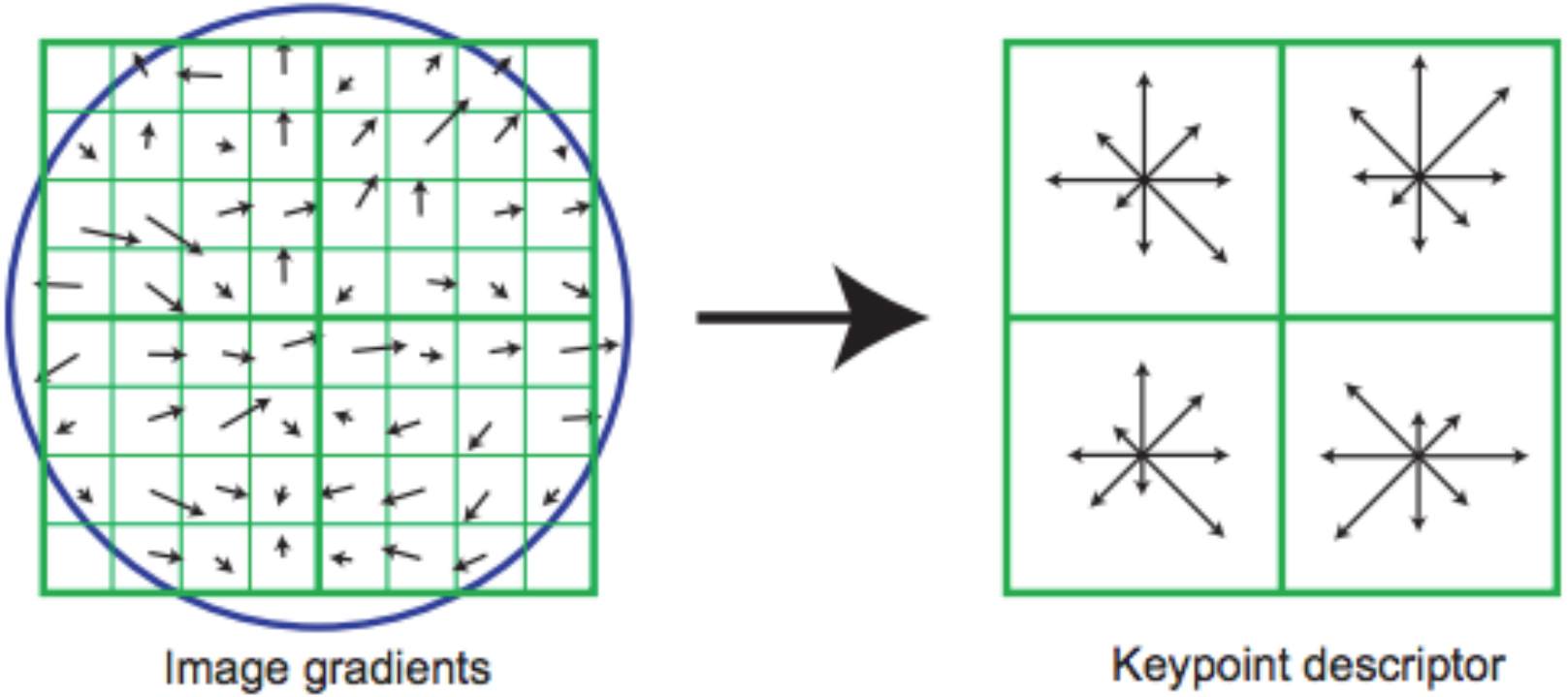


Figure 7: A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2x2 descriptor array computed from an 8x8 set of samples, whereas the experiments in this paper use 4x4 descriptors computed from a 16x16 sample array.

Width and # of orientations

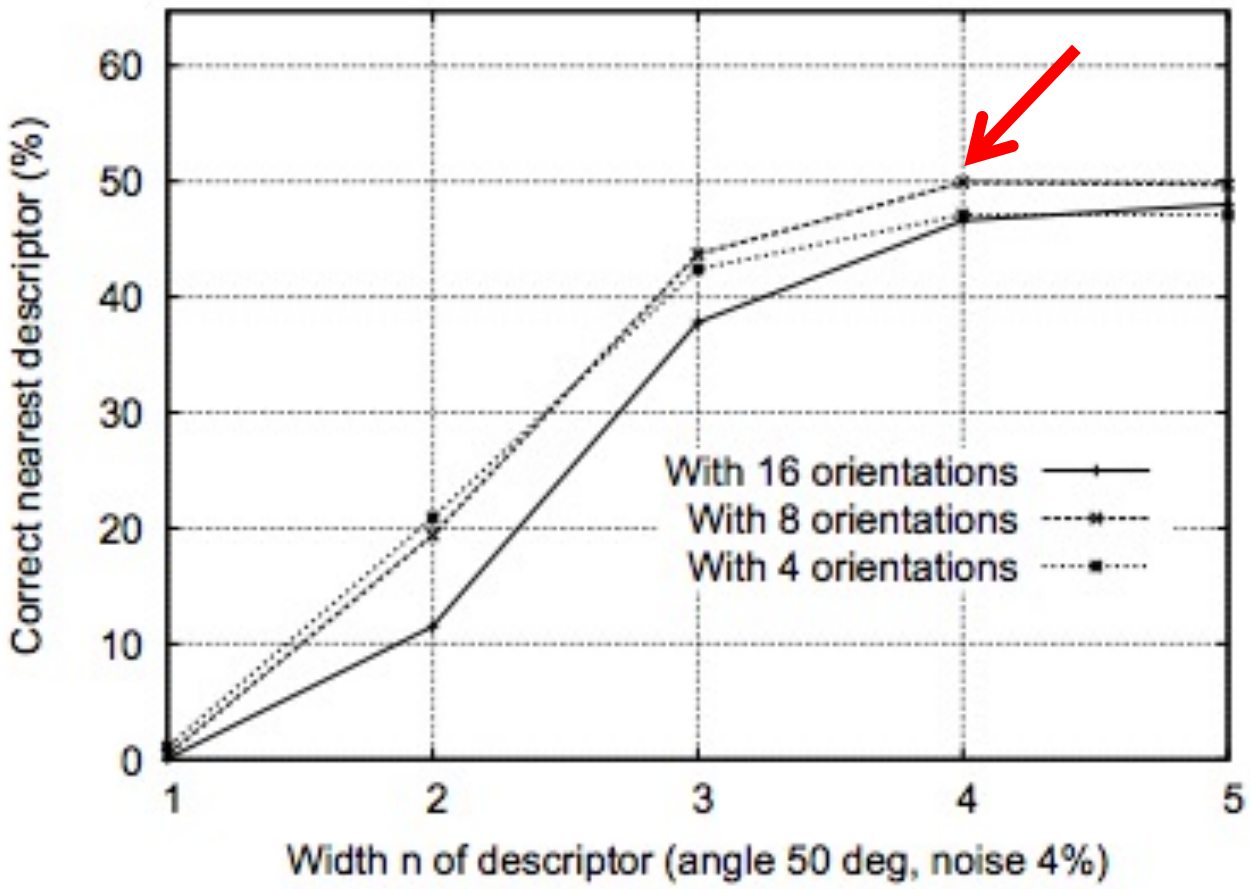
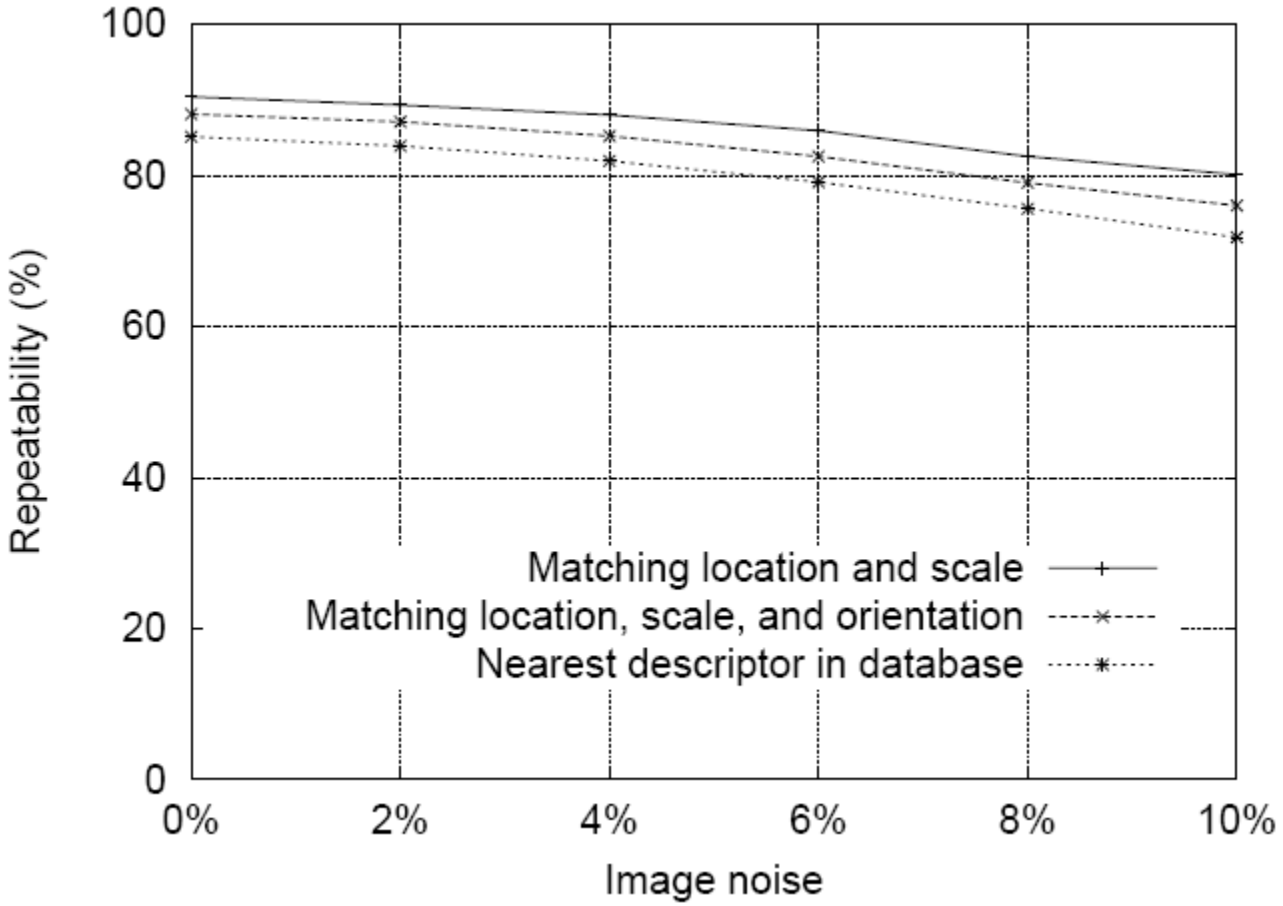


Figure 8: This graph shows the percent of keypoints giving the correct match to a database of 40,000 keypoints as a function of width of the $n \times n$ keypoint descriptor and the number of orientations in each histogram. The graph is computed for images with affine viewpoint change of 50 degrees and addition of 4% noise.

Lowe IJCV 2004

Stability to image noise



Lowe IJCV 2004

Stability to affine distortion

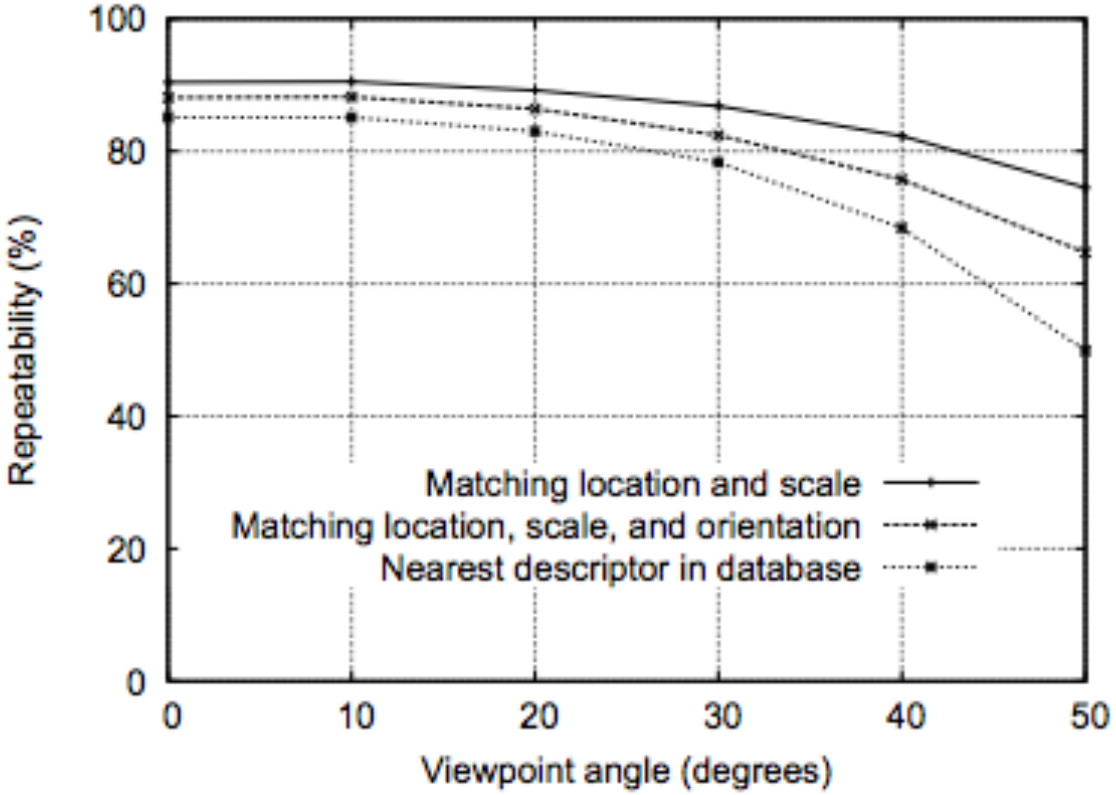


Figure 9: This graph shows the stability of detection for keypoint location, orientation, and final matching to a database as a function of affine distortion. The degree of affine distortion is expressed in terms of the equivalent viewpoint rotation in depth for a planar surface.

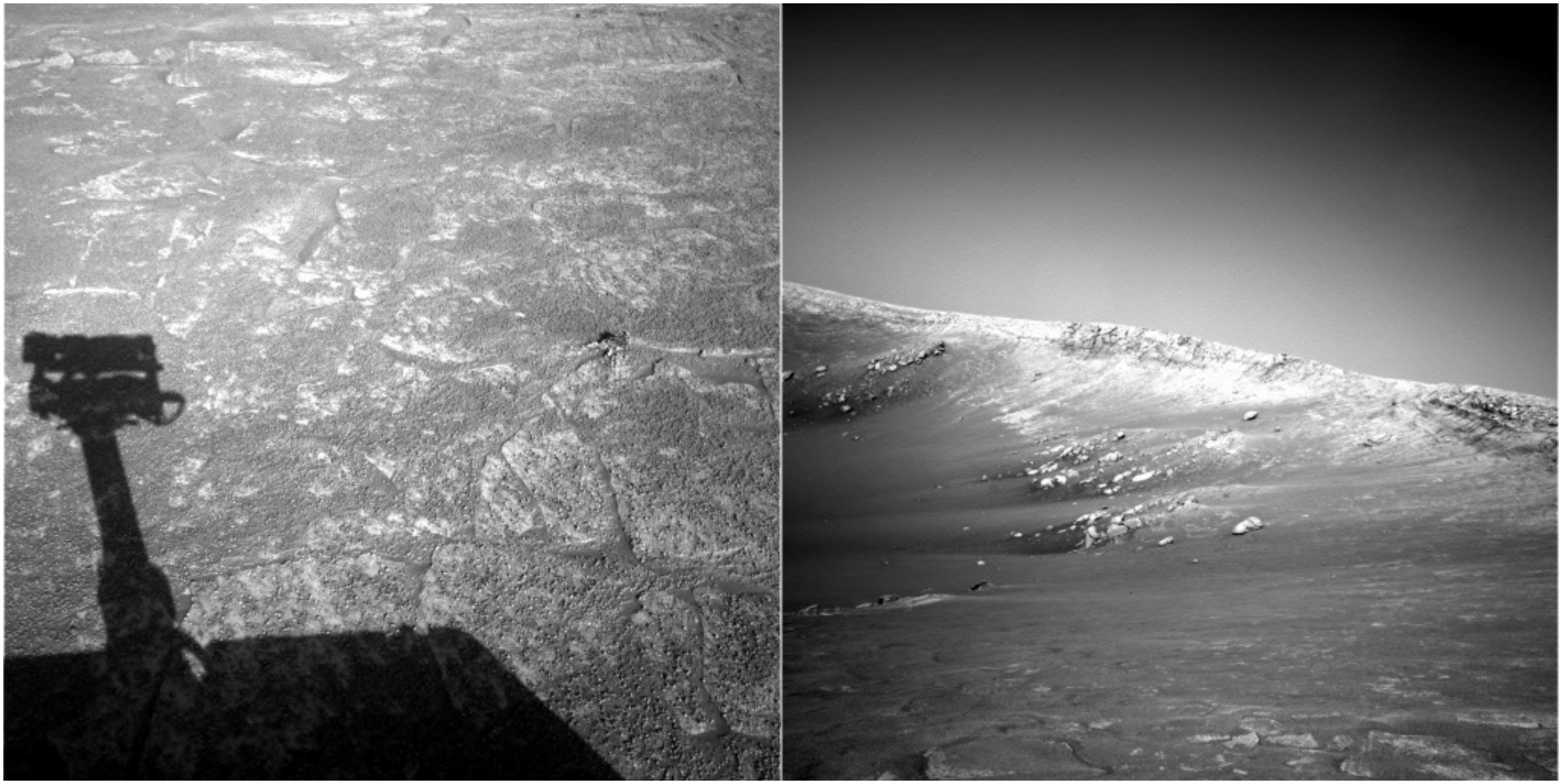
Lowe IJCV 2004

SIFT descriptor

- 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, e.g. http://www.vlfeat.org/overview/sift.html

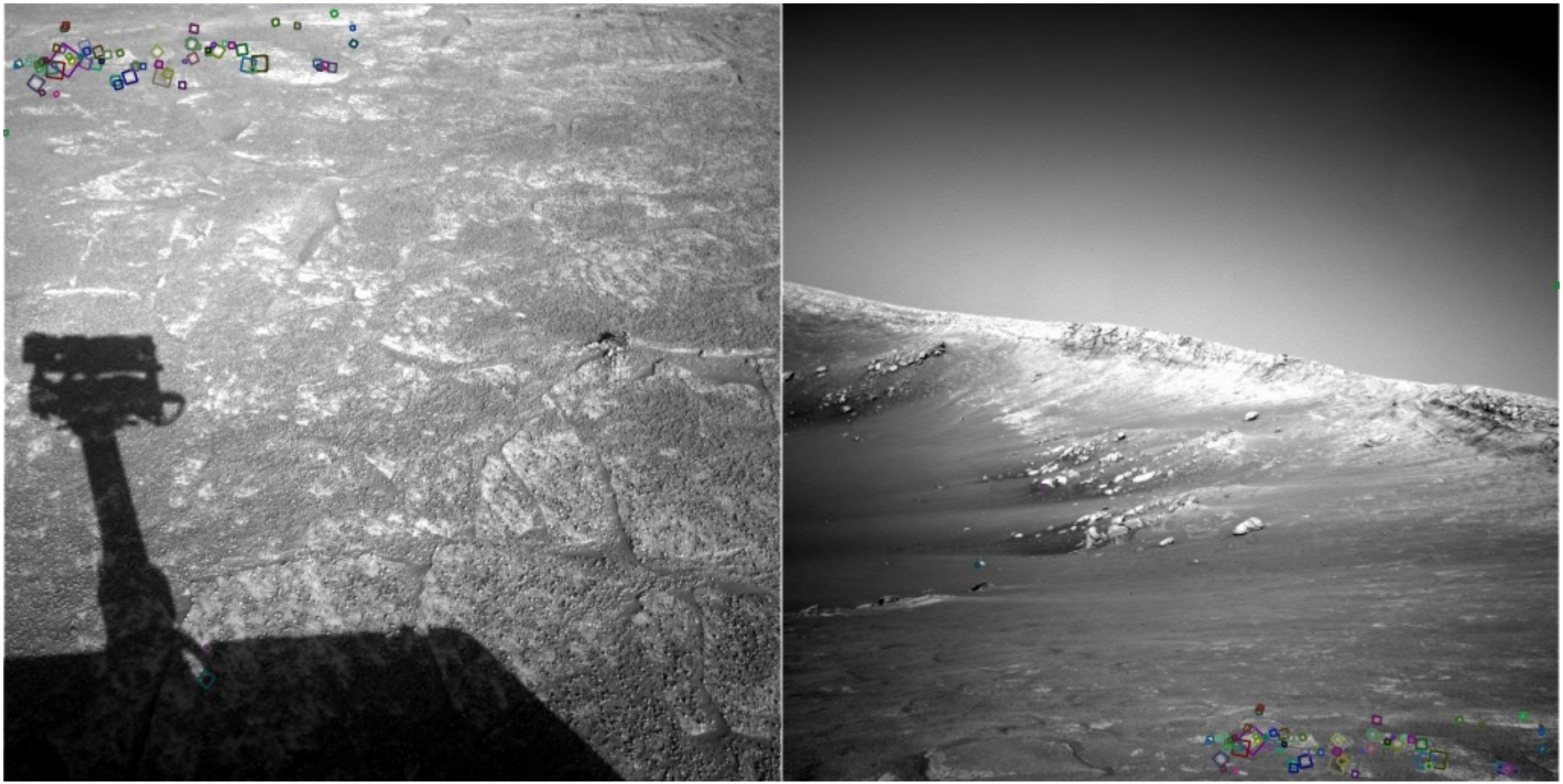
Steve Seitz

Example



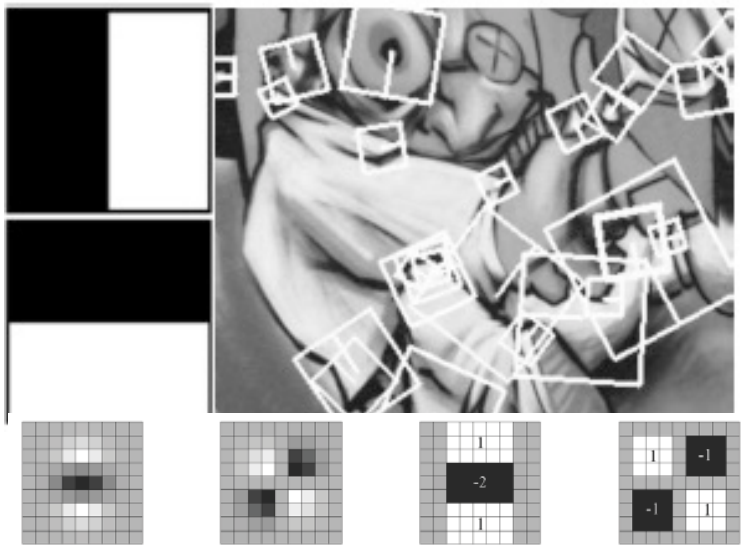
NASA Mars Rover images

Example



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

Local Descriptors: SURF



Fast approximation of SIFT idea

Efficient computation by 2D box filters & integral images

⇒ 6 times faster than SIFT

Equivalent quality for object identification

GPU implementation available

Feature extraction @ 200Hz (detector + descriptor, 640×480 img)

http://www.vision.ee.ethz.ch/~surf

Many other efficient descriptors are also available

[Bay, ECCV'06], [Cornelis, CVGPU'08]

Local Descriptors: ORB

- Many similarities to SIFT/SURF
- Designed for efficiency and robustness to orientation
- Not designed for scale robustness
- Used for tracking and long-range matching in ORB-SLAM

http://www.willowgarage.com/sites/default/files/orb_final.pdf (ICCV 2011)
http://webdiis.unizar.es/~raulmur/orbslam/