Lecture 4

Feature Detectors: Corners, Blobs and SIFT

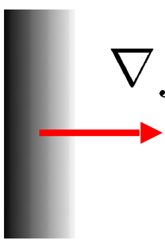
COS 429: Computer Vision

PRINCETON UNIVERSITY

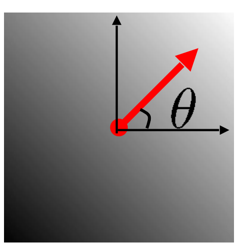
Slides adapted from: Szymon Rusinkiewicz, Jia Deng, Svetlana Lazebnik, Steve Seitz, David Fouhey

Last time: edge detection

90	92	92	93	93	94	94	95	95	96
94	95	96	96	97	98	98	99	99	99
98	99	99	100	101	101	102	102	102	103
103	103	104	104	105	107	106	106	111	121
108	108	109	110	112	111	112	119	123	117
113	113	110	111	113	112	122	120	117	106
118	118	109	96	106	113	112	108	117	114
116	132	120	111	109	106	101	106	117	118
111	142	112	111	101	106	104	109	113	110
114	139	109	108	103	106	107	108	108	108
115	139	117	114	101	104	103	105	114	110
115	129	103	114	101	97	109	116	117	118
120	130	104	111	116	104	107	109	110	99
125	130	103	109	108	98	104	109	119	105
119	128	123	138	140	133	139	120	137	145
164	138	143	163	155	133	145	125	133	155
174	126	123	122	102	106	108	62	62	114
169	134	133	127	92	102	94	47	52	118
125	132	117	122	102	103	98	51	53	120
109	99	113	116	111	98	104	82	99	116



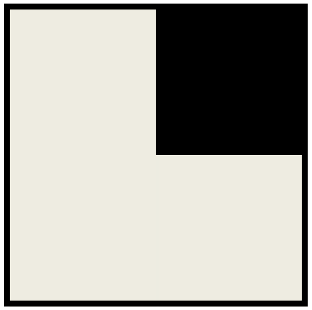
$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



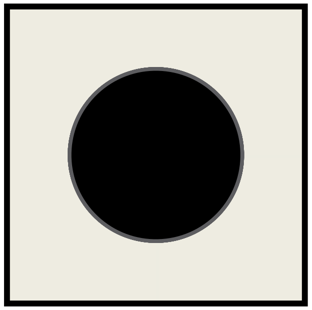
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

This time: keypoints

- Corners



- Blobs

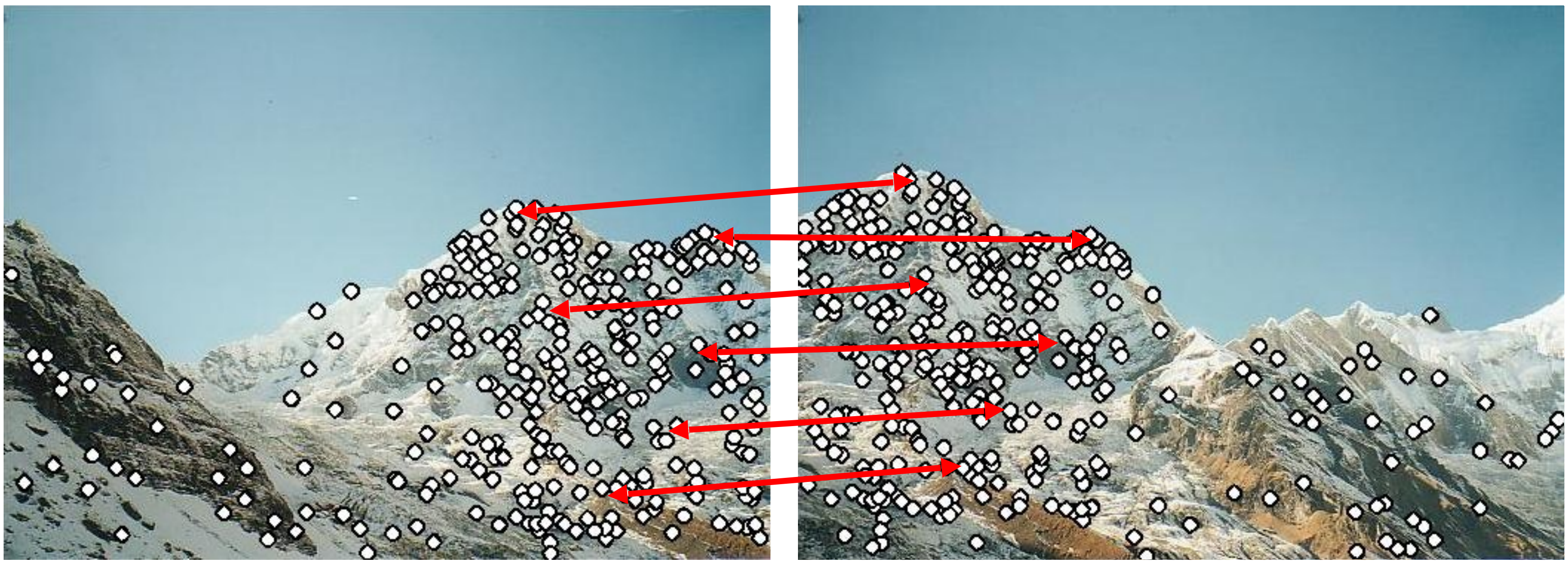


Why Extract Keypoints?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Why Extract Keypoints?

- Motivation: panorama stitching
  - We have two images – how do we combine them?



Step 1: extract keypoints
Step 2: match keypoint features

Why Extract Keypoints?

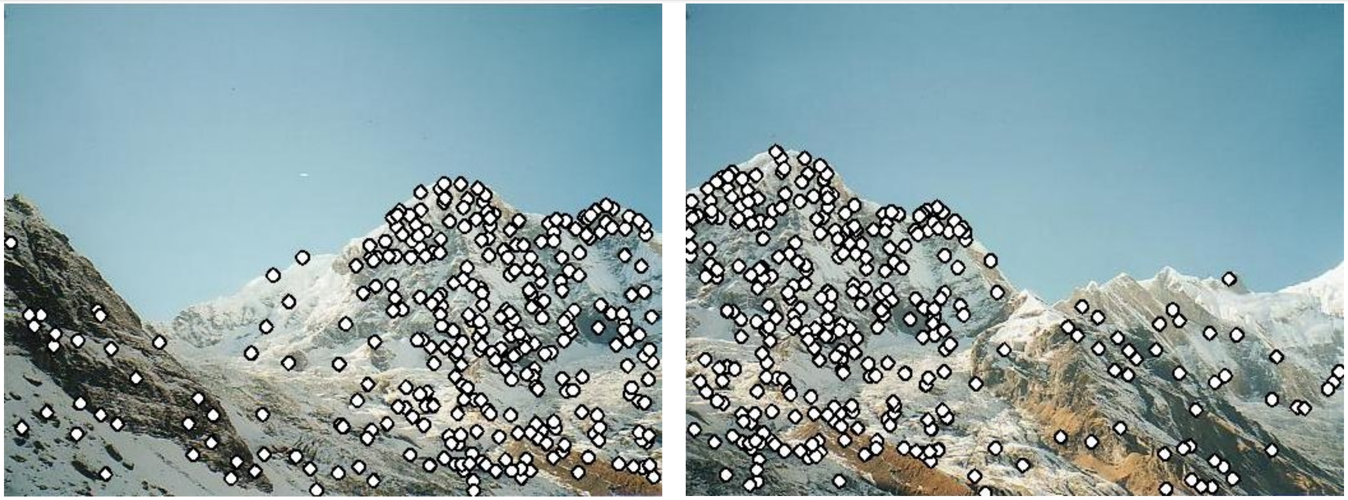
- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 1: extract keypoints
Step 2: match keypoint features
Step 3: align images

Applications

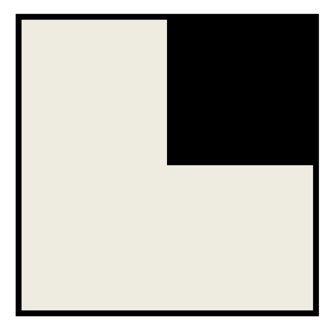
- Keypoints are used for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking
  - Robot navigation
  - Indexing and database retrieval
  - Object recognition

Characteristics of Good Keypoints



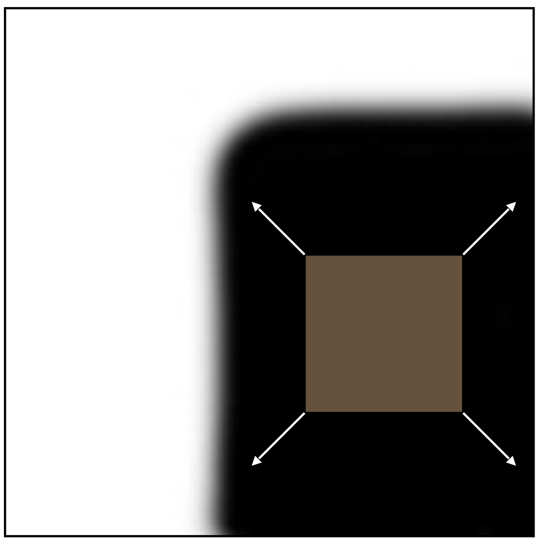
- Repeatability
  - Can be found despite geometric and photometric transformations
- Salience
  - Each keypoint is distinctive
- Compactness and efficiency
  - Many fewer keypoints than image pixels
- Locality
  - Occupies small area of the image; robust to clutter and occlusion

Corners

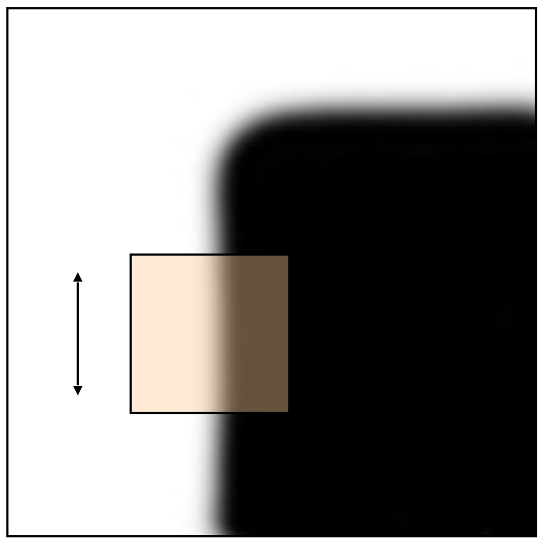


Corner Detection: Basic Idea

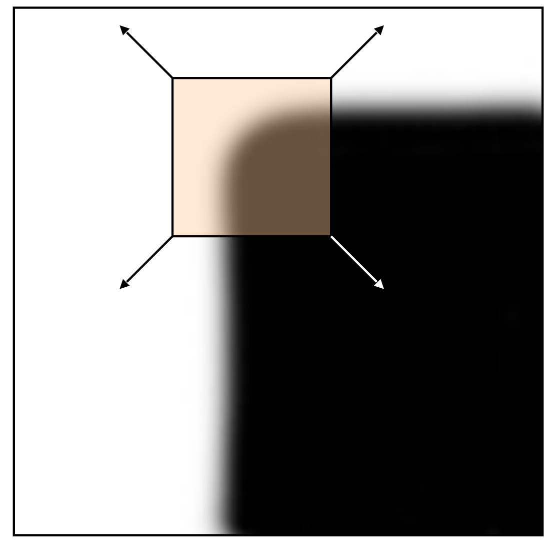
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions



"edge":
no change along the edge direction



"corner":
significant change in all directions

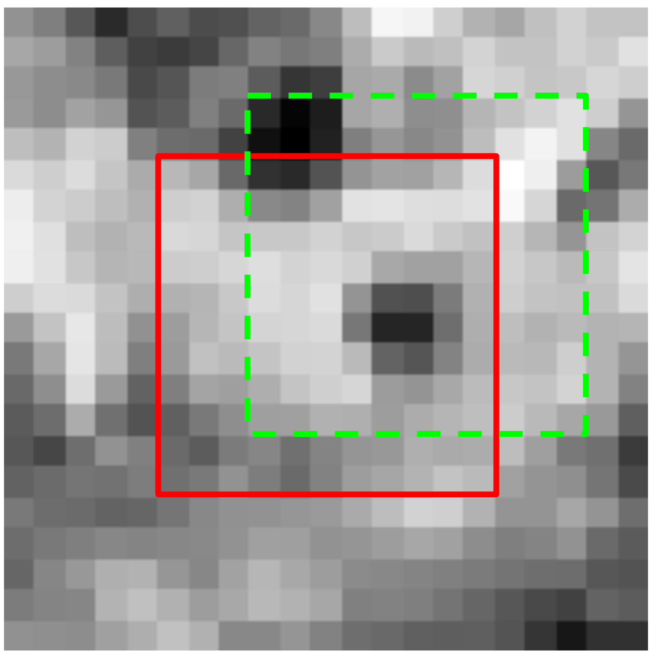
Slide credit: S. Lazebnik

Corner Detection: Mathematics

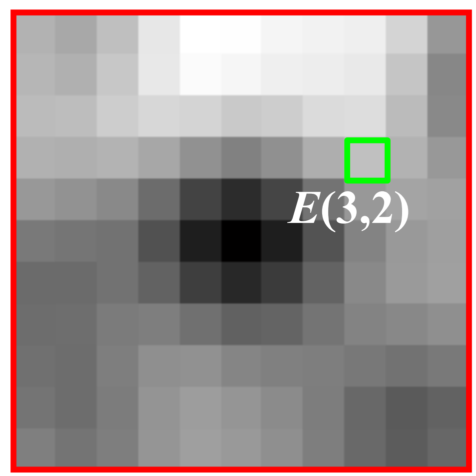
Change in appearance of window W for the shift $[u,v]$:

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

$I(x, y)$



$E(u, v)$



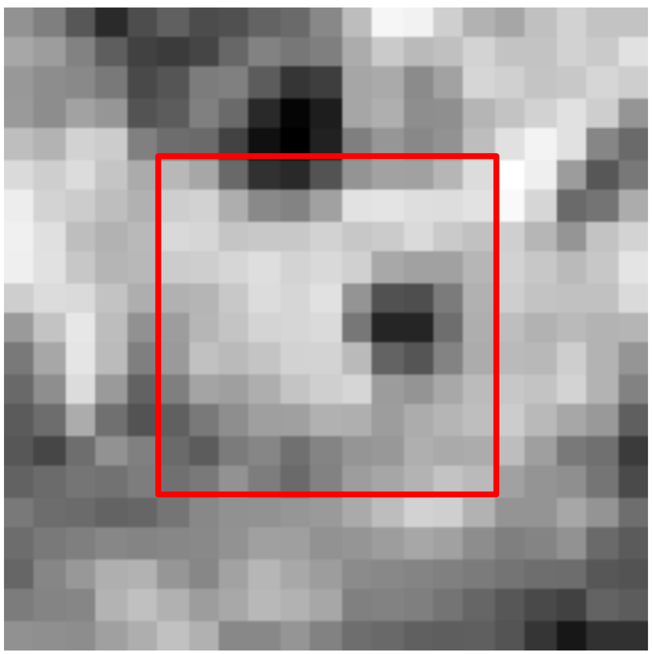
Slide credit: S. Lazebnik

Corner Detection: Mathematics

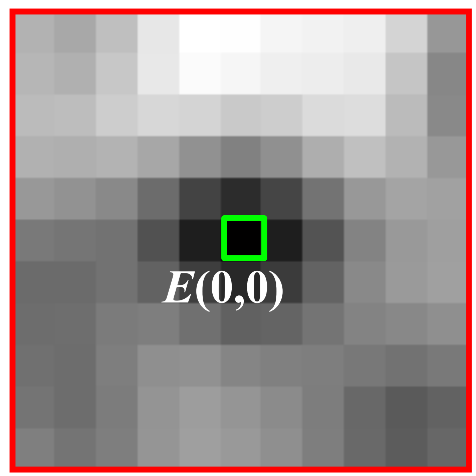
Change in appearance of window *W* for the shift [*u,v*]:

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

$I(x, y)$



$E(u, v)$



Slide credit: S. Lazebnik

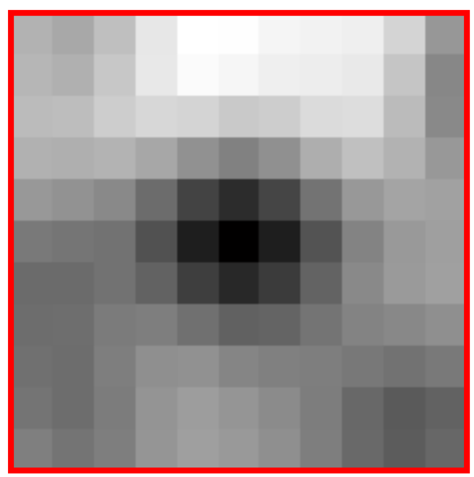
Corner Detection: Mathematics

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

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

We want to find out how this function behaves for small shifts

$E(u, v)$



Slide credit: S. Lazebnik

Corner Detection: Mathematics

- First-order Taylor approximation for small motions [*u*, *v*]:

$$I(x+u, y+v) \approx I(x,y) + I_x u + I_y v$$

- Let's plug this into *E*(*u*,*v*):

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

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

$$= \sum_{(x,y)\in W} [I_x u + I_y v]^2 = \sum_{(x,y)\in W} [I_x^2 u^2 + 2 I_x I_y uv + I_y^2 v^2]$$

Corner Detection: Mathematics

The quadratic approximation can be written as

$$E(u,v) \approx \sum_{(x,y)\in W} I_x^2 u^2 + 2 I_x I_y uv + I_y^2 v^2 = \boxed{[u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix}}$$

where *M* is a *second moment matrix* computed from image derivatives:

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

(the sums are over all the pixels in the window *W*)

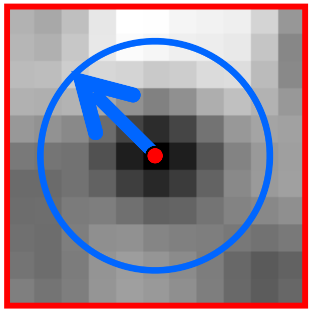
Interpreting the second moment matrix

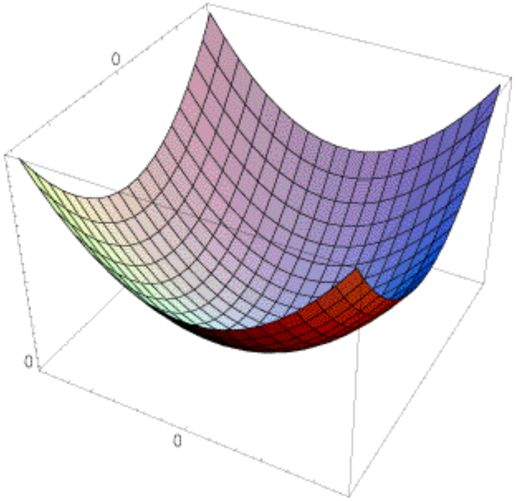
- The surface $E(u,v)$ is locally approximated by a quadratic form. Let's try to understand its shape.

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

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

$E(u, v)$





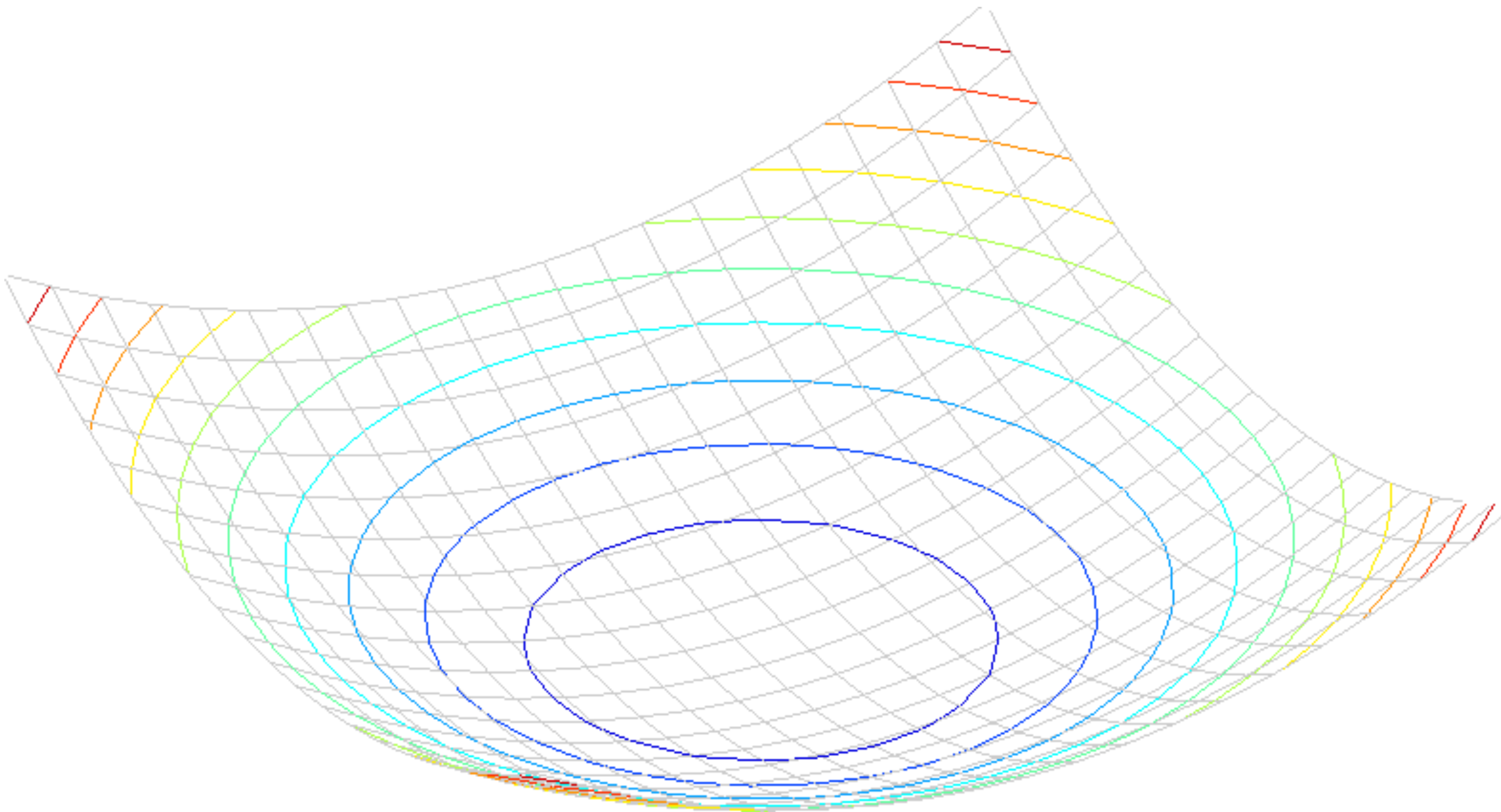
- Specifically, in which directions does it have the smallest/greatest change?

Slide credit: S. Lazebnik

Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$: $$[u \; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$

This is the equation of an ellipse.



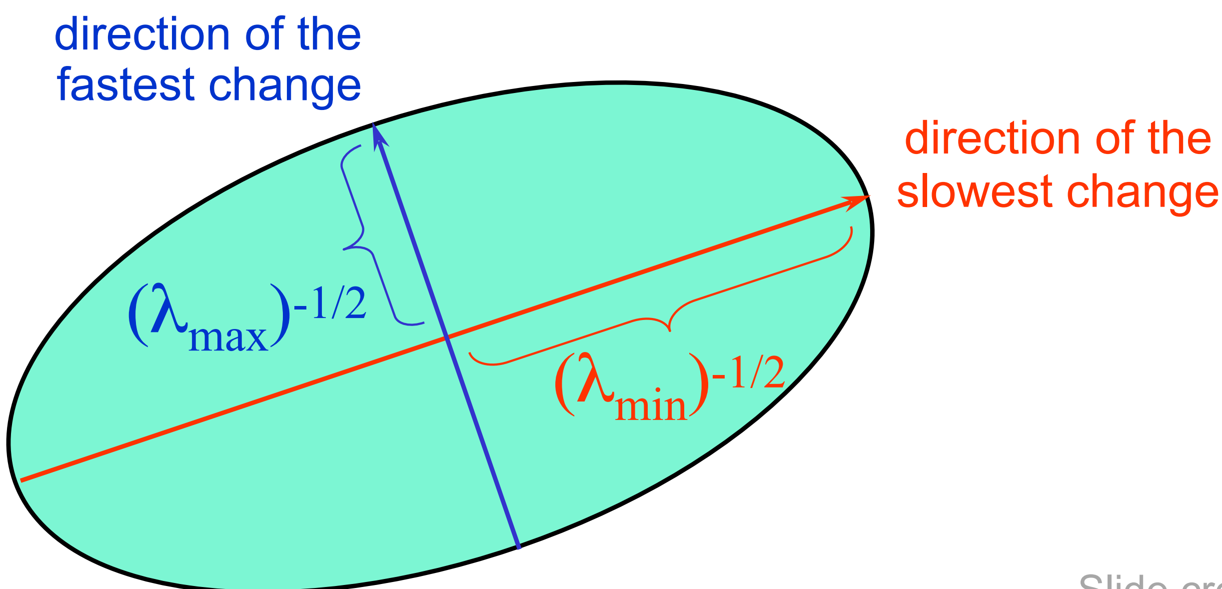
Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$: $\quad [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.

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

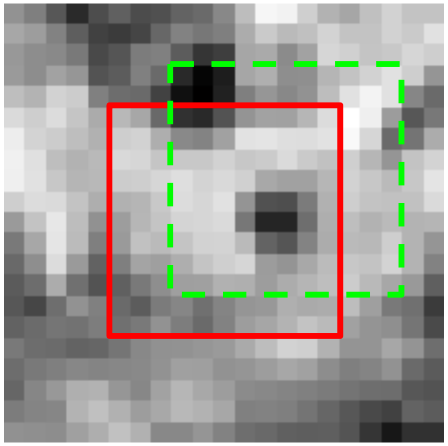
The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by *R*



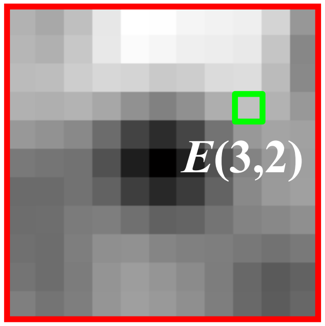
Slide credit: S. Lazebnik

Recap so far

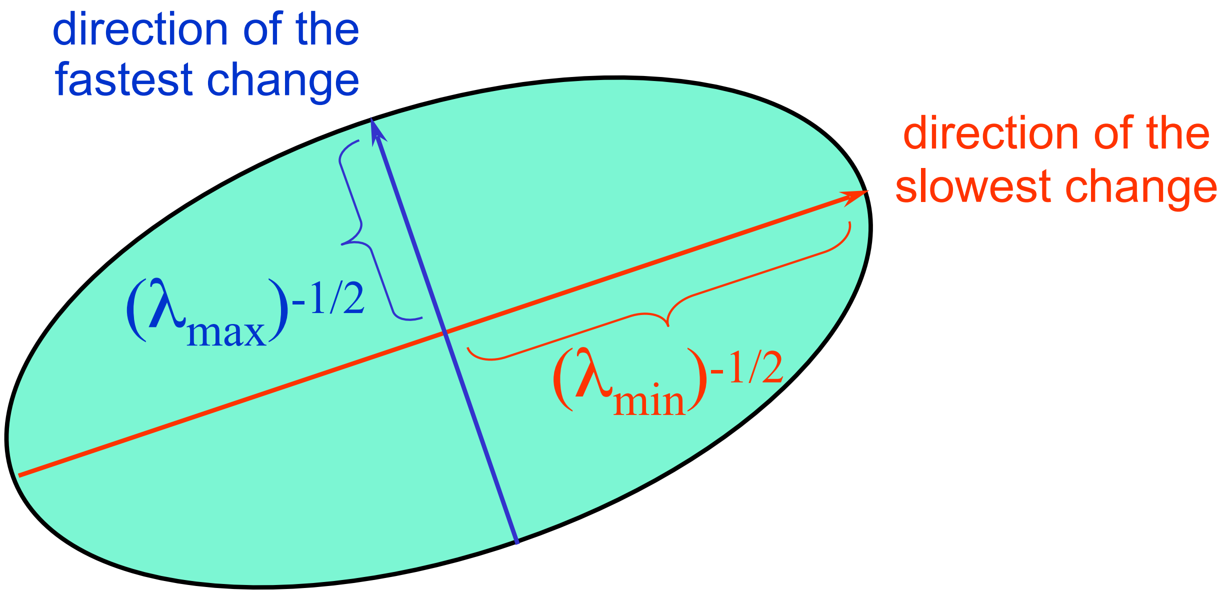
$I(x, y)$



$E(u, v)$

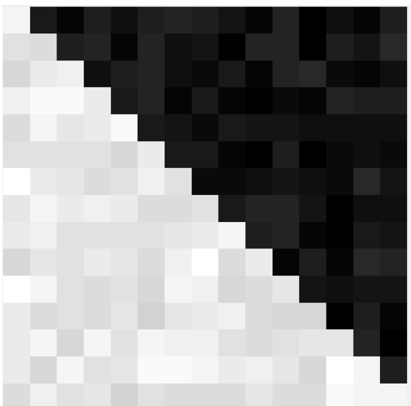


$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

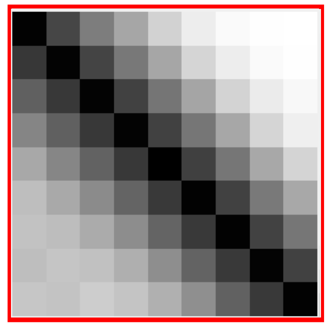


At an edge

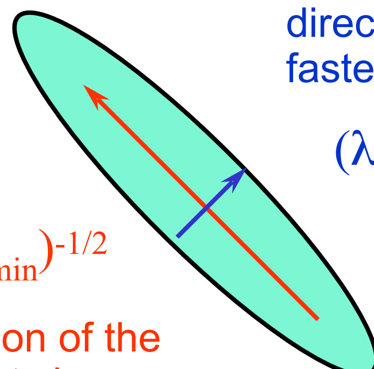
$I(x, y)$



$E(u, v)$



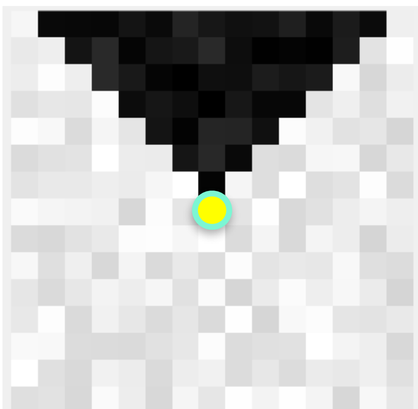
$$E(u,v) \approx [u \ \ v] \ M \begin{bmatrix} u \\ v \end{bmatrix}$$



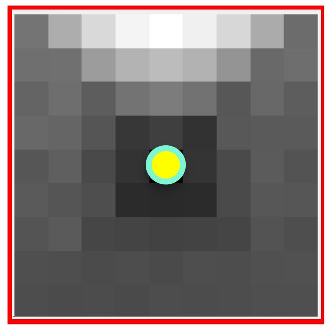
- The direction along the edge results in no change
- λ_{min} is very small

At a corner

$I(x, y)$



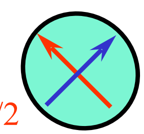
$E(u, v)$



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

direction of the fastest change

$(\lambda_{max})^{-1/2}$



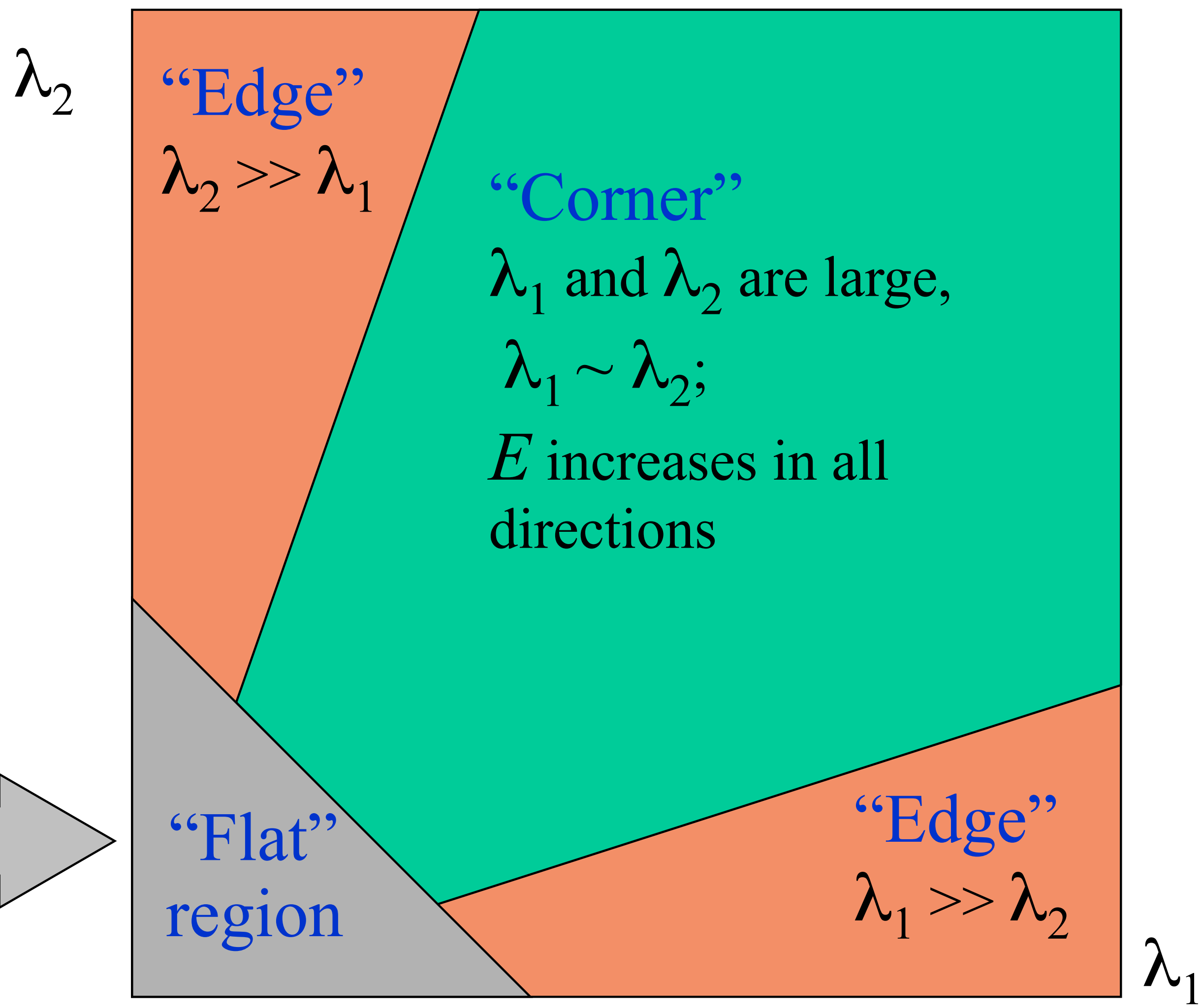
$(\lambda_{min})^{-1/2}$

direction of the slowest change

- All directions result in high change
- λ_{min} is large

Interpreting the eigenvalues

Classification of image points using eigenvalues of *M*:



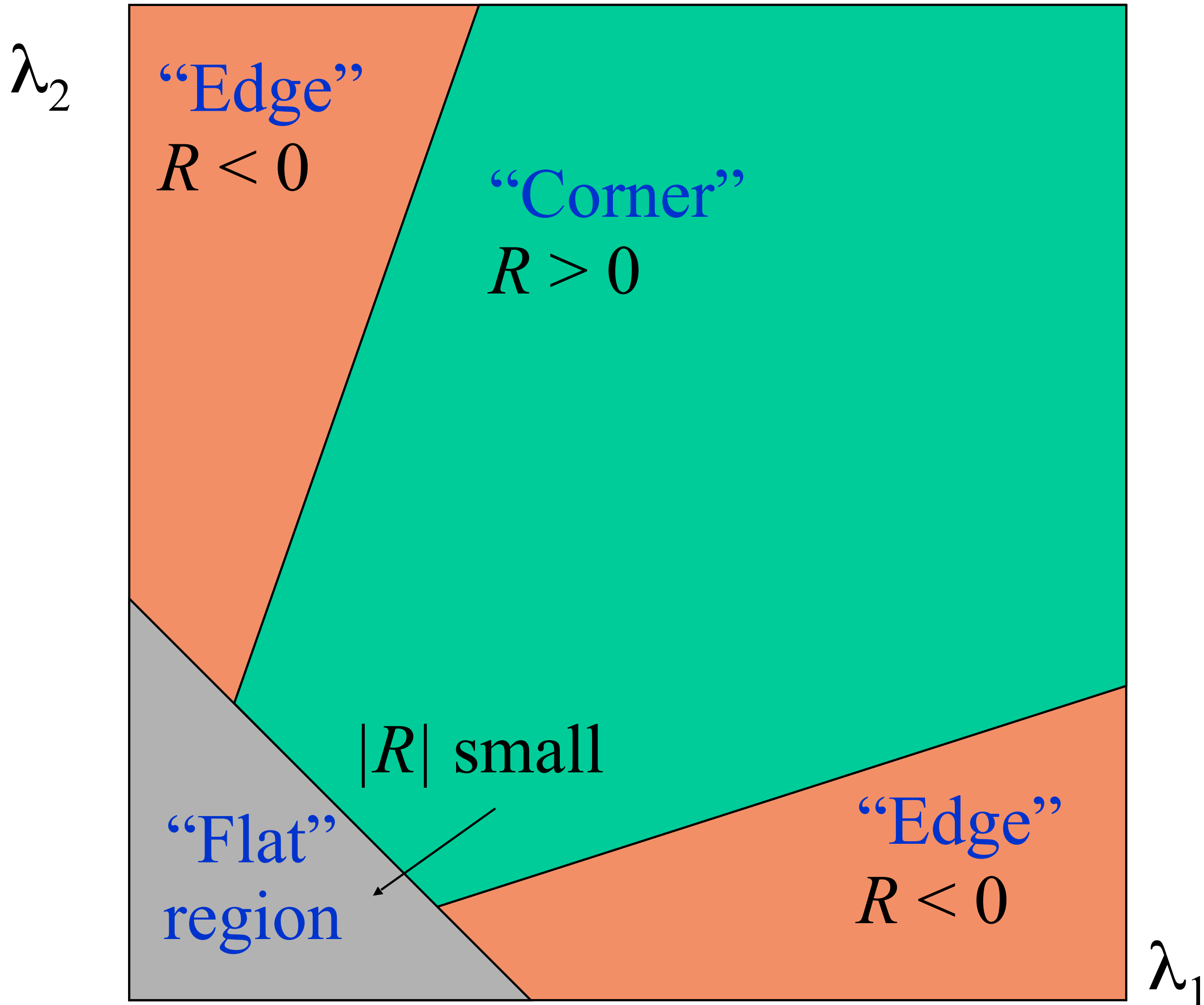
λ_1 and λ_2 are small;
E is almost constant in all directions

Slide credit: S. Lazebnik

Corner response function

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

α: constant



Slide credit: S. Lazebnik

The Harris corner detector

1. Compute partial derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel:

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

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

Slide credit: S. Lazebnik

The Harris corner detector

1. Compute partial derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel
3. Compute corner response function *R*

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

Slide credit: S. Lazebnik

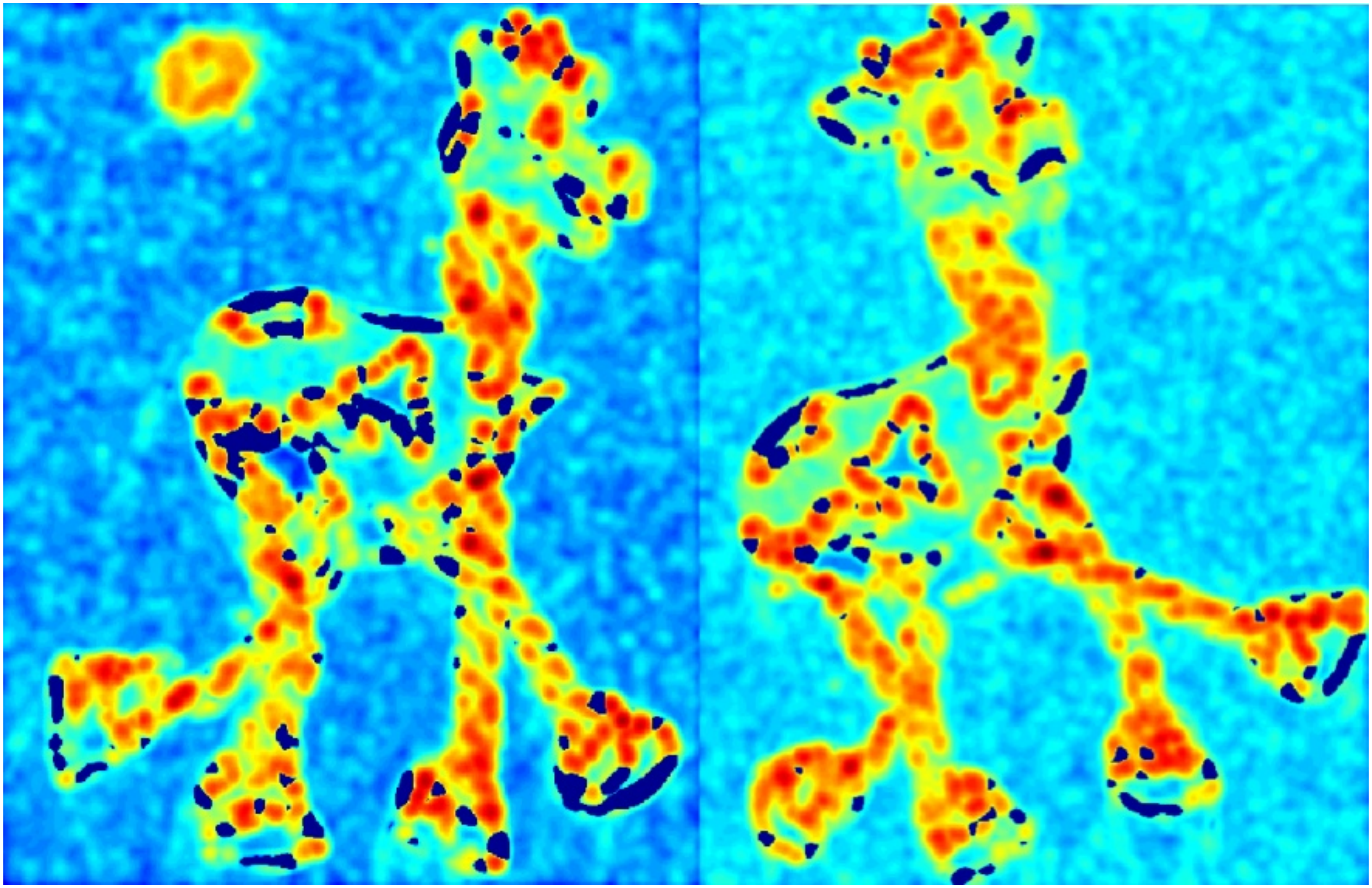
Harris Detector: Steps

Two images of the same object

Slide credit: S. Lazebnik

Harris Detector: Steps

Compute corner response R



Slide credit: S. Lazebnik

The Harris corner detector

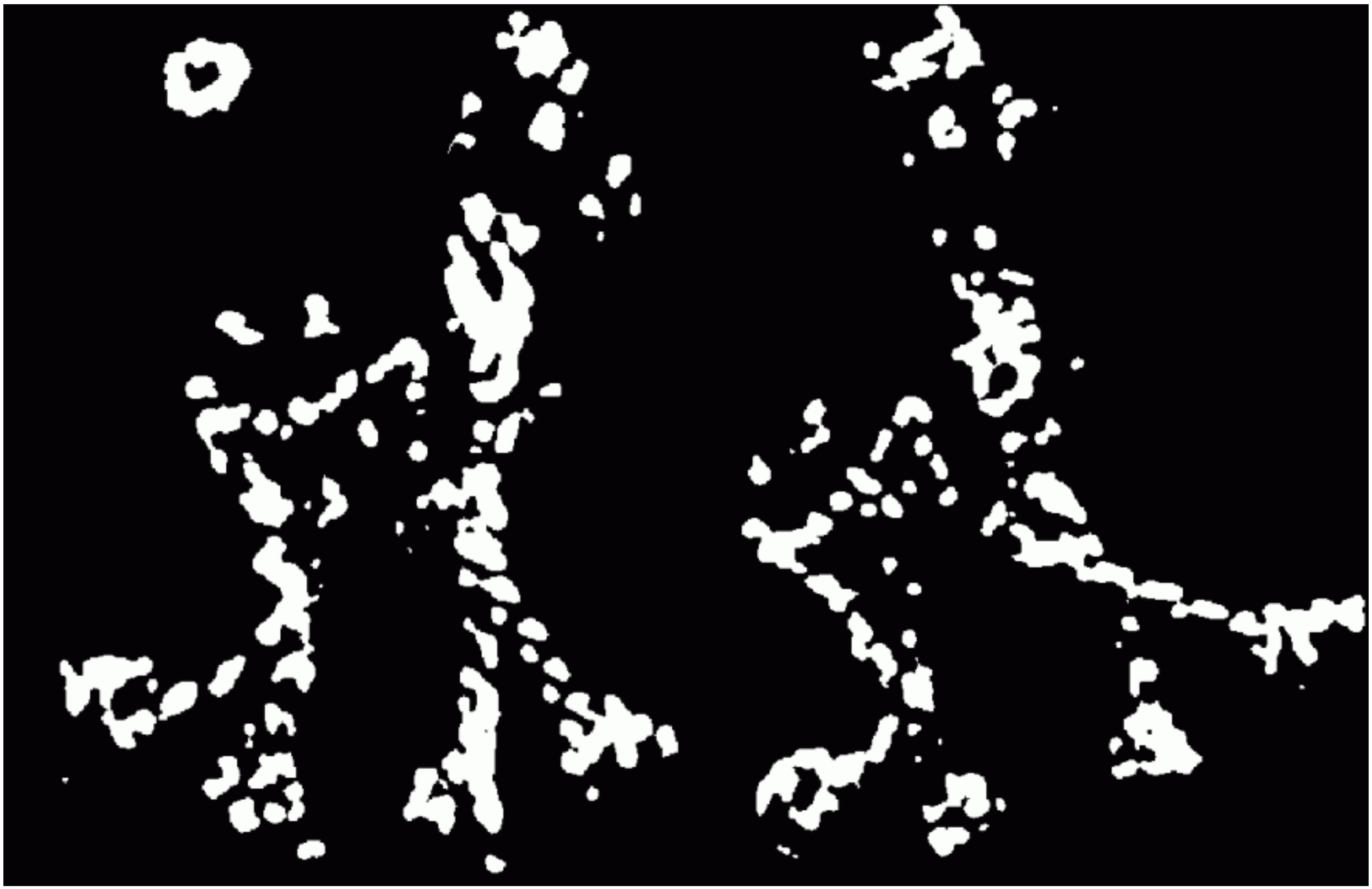
1. Compute partial derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel
3. Compute corner response function *R*
4. Threshold *R*
5. Find local maxima of response function (nonmaximum suppression)

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

Slide credit: S. Lazebnik

Harris Detector: Steps

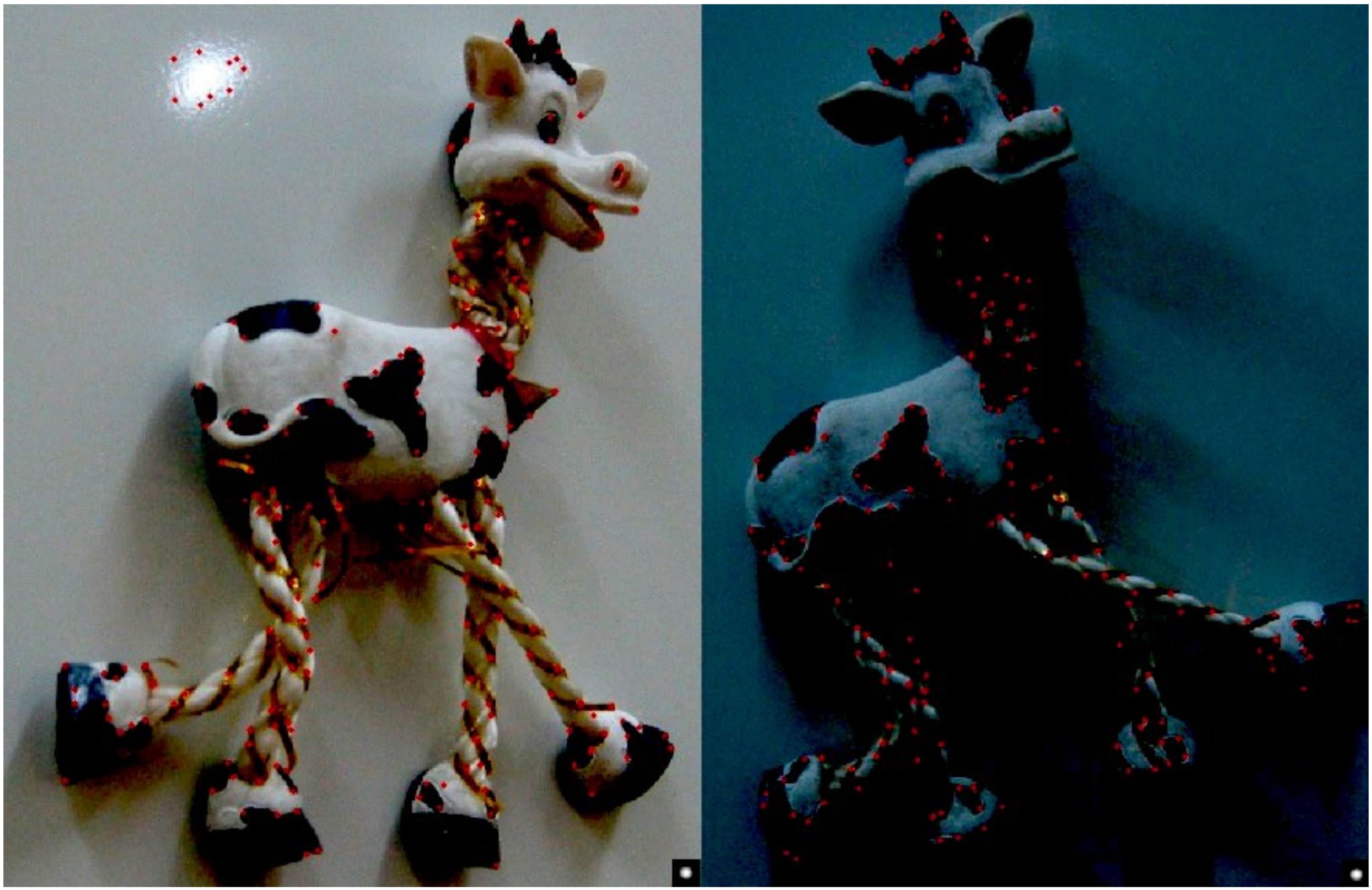
Find points with large corner response: R > threshold



Slide credit: S. Lazebnik

Harris Detector: Steps

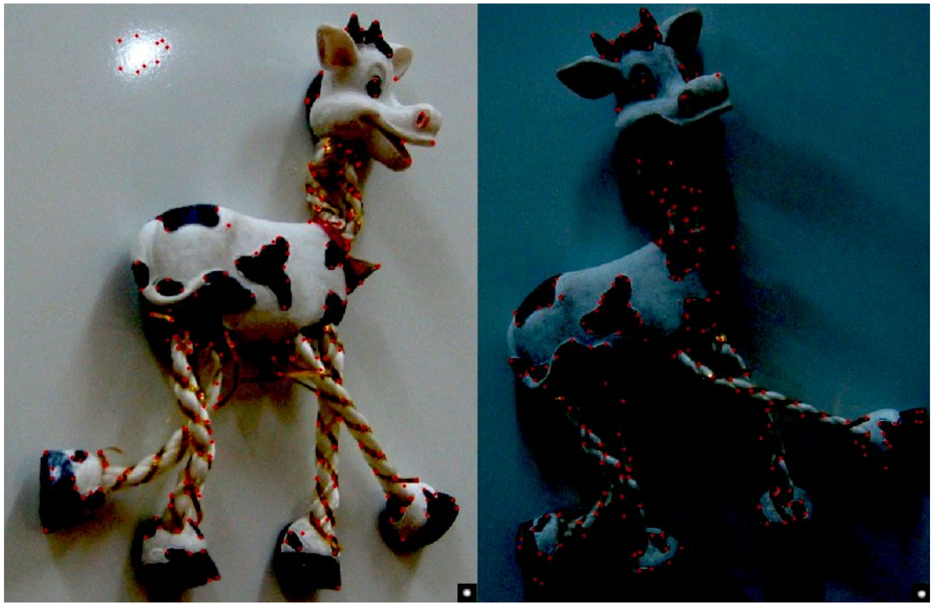
Take only the points of local maxima of *R*



Slide credit: S. Lazebnik

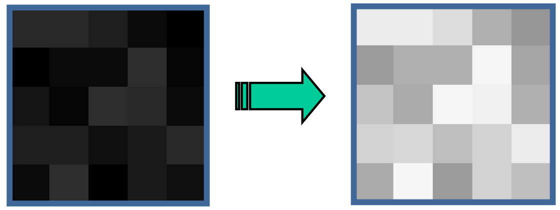
Invariance and covariance

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



Slide credit: S. Lazebnik

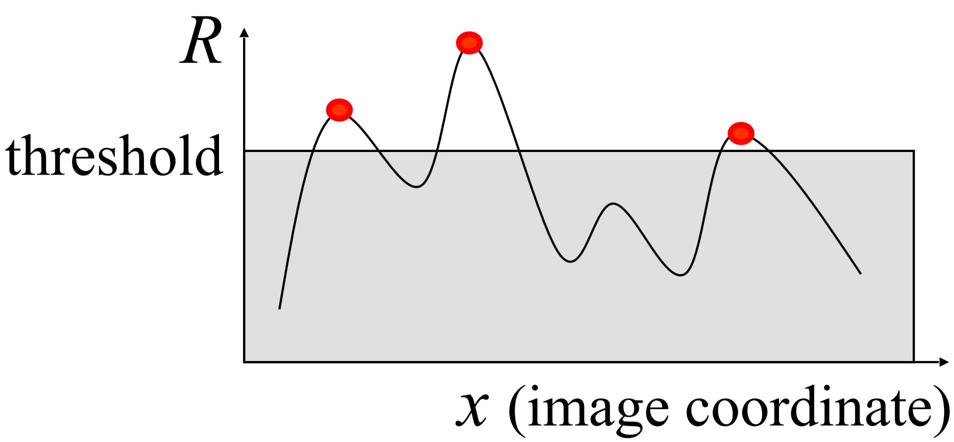
Affine intensity change

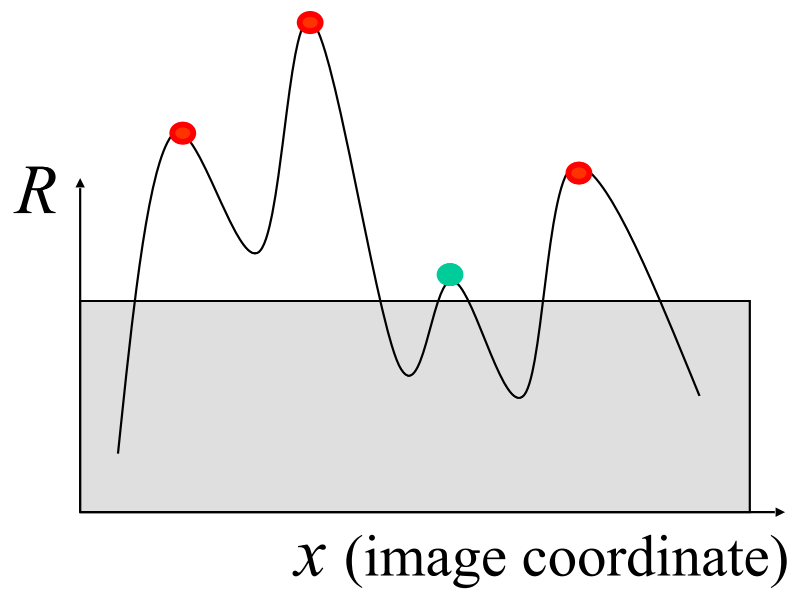


$I \rightarrow a\,I + b$

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

- Intensity scaling: $I \rightarrow a\,I$

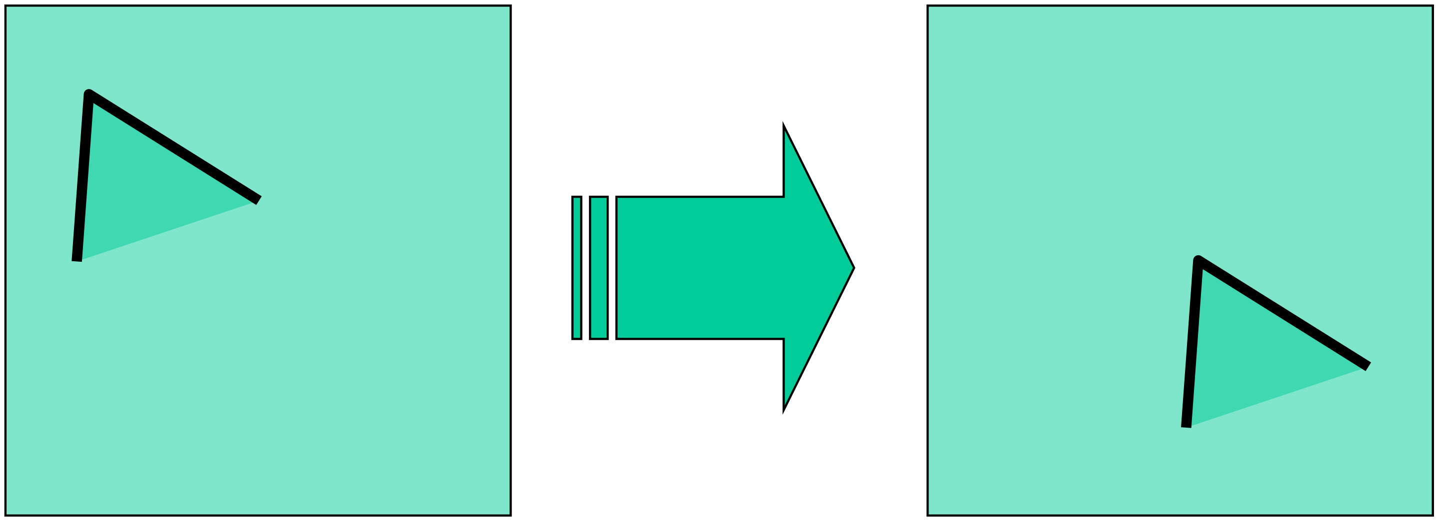




Partially invariant to affine intensity change

Slide: S. Lazebnik

Image translation

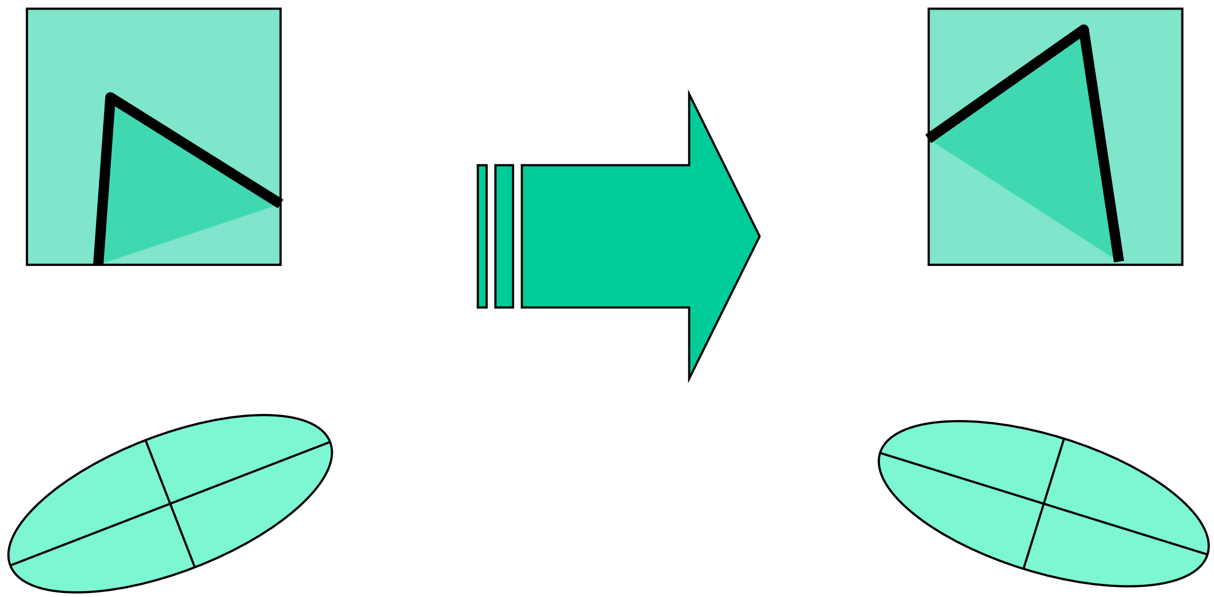


- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation

Slide: S. Lazebnik

Image rotation

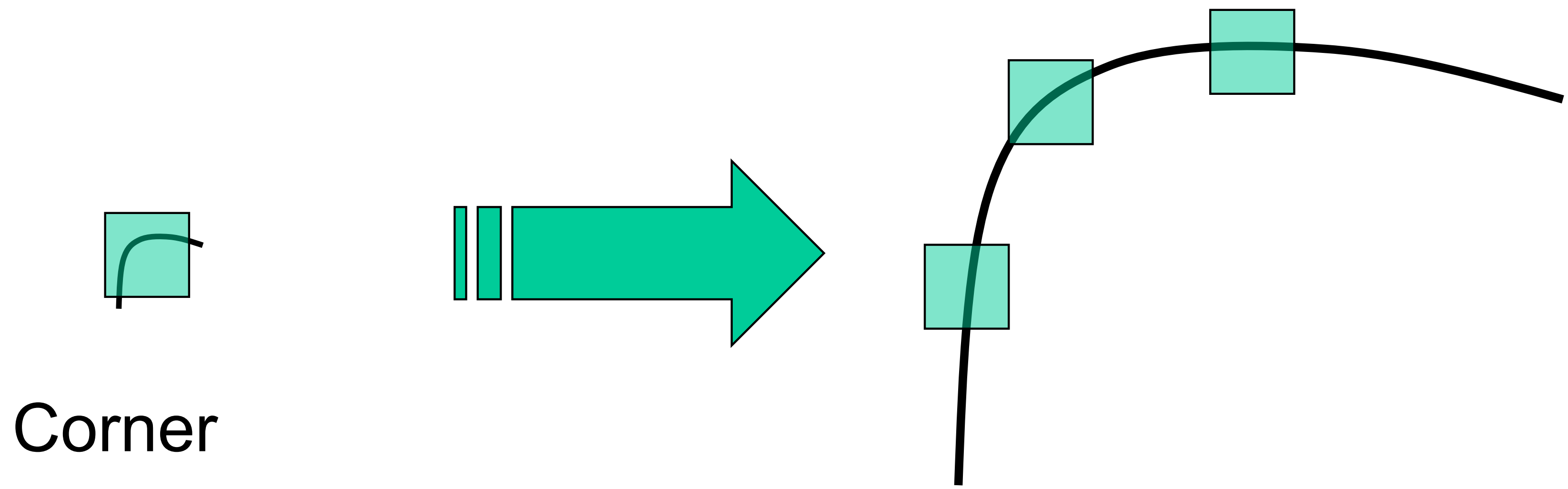


Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation

Slide: S. Lazebnik

Scaling

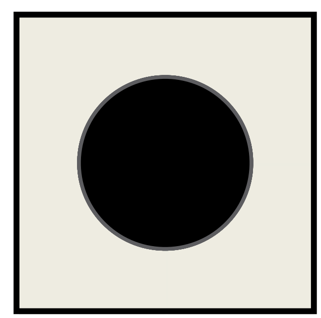


All points will be classified as edges

Corner location is not covariant to scaling!

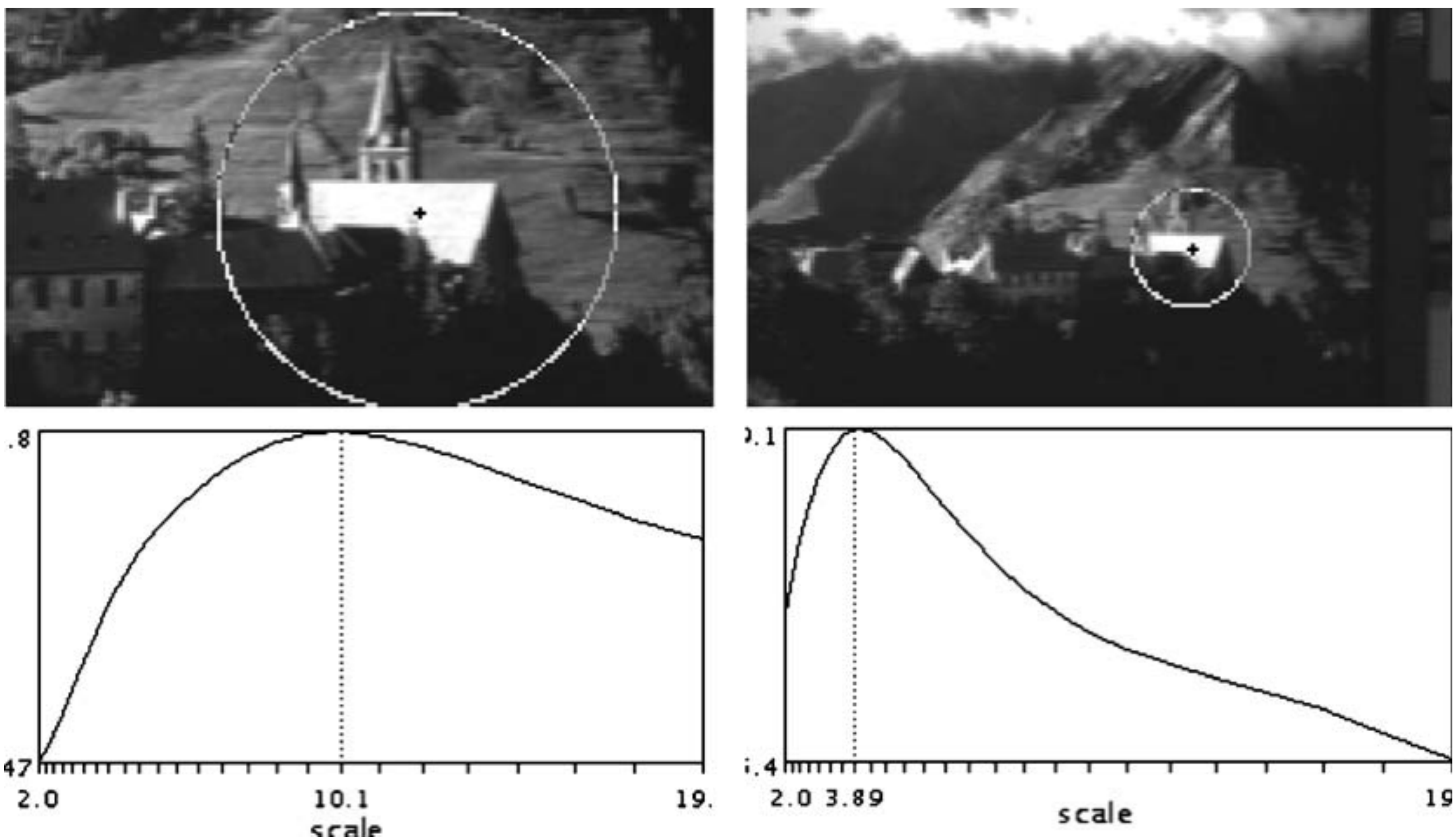
Slide: S. Lazebnik

Blobs



Feature detection with scale selection

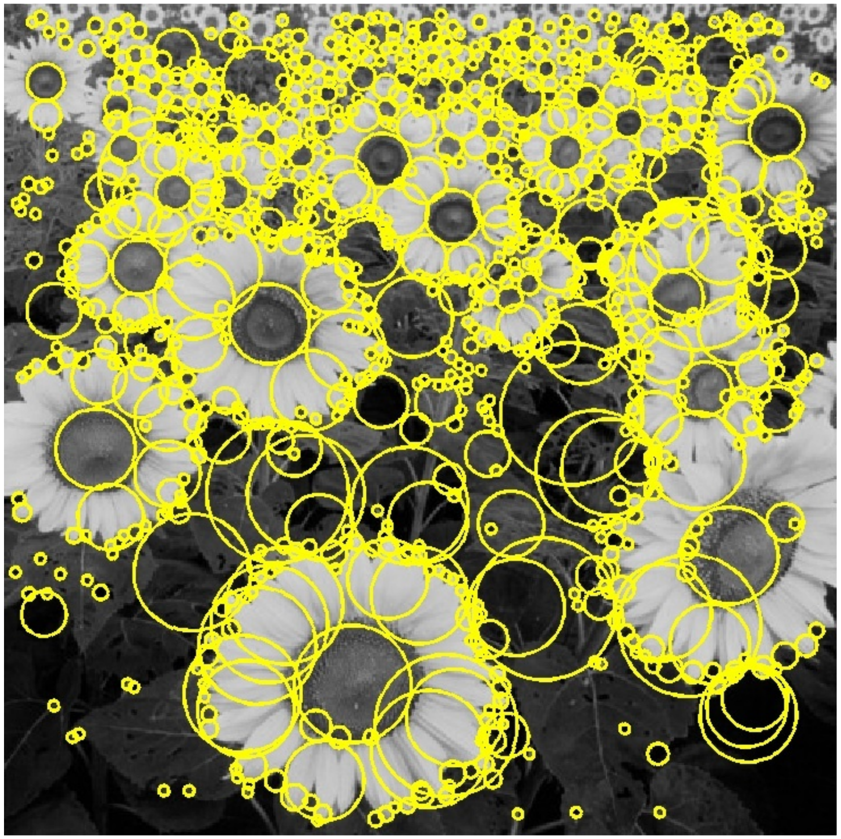
- We want to extract features with characteristic scale that is *covariant* with the image transformation

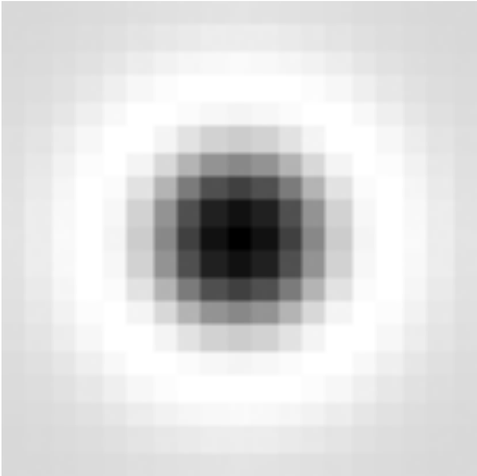


Slide: S. Lazebnik

Blob detection: Basic idea

- To detect blobs, convolve the image with a "blob filter" at multiple scales and look for extrema of filter response in the resulting *scale space*



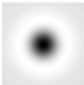


Slide: S. Lazebnik

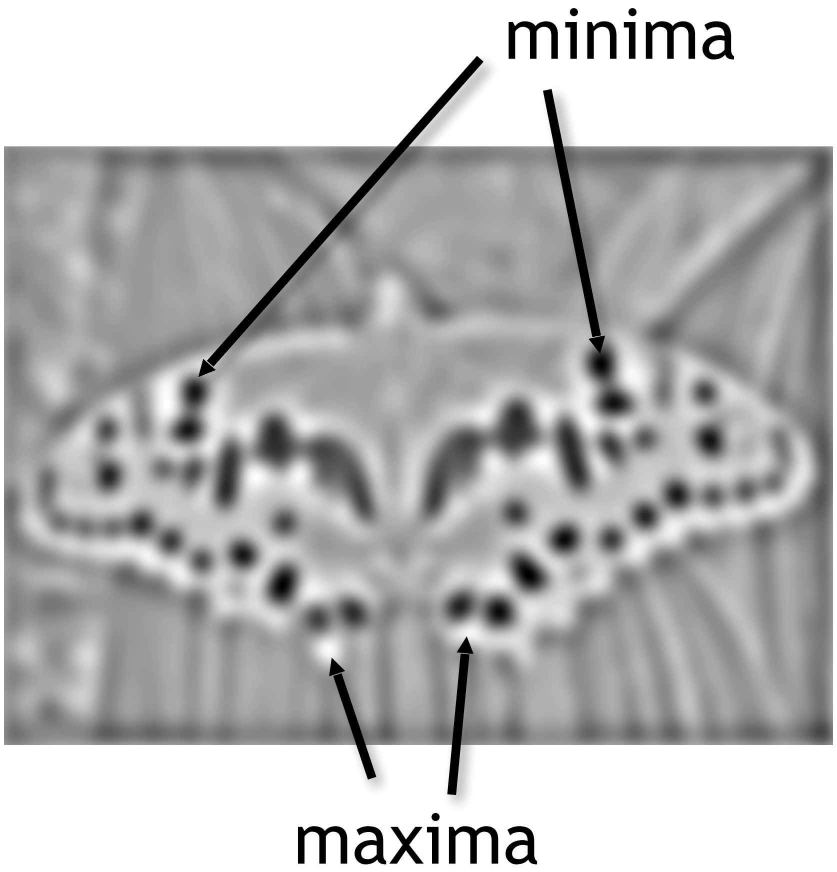
Blob detection: Basic idea

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

$*$



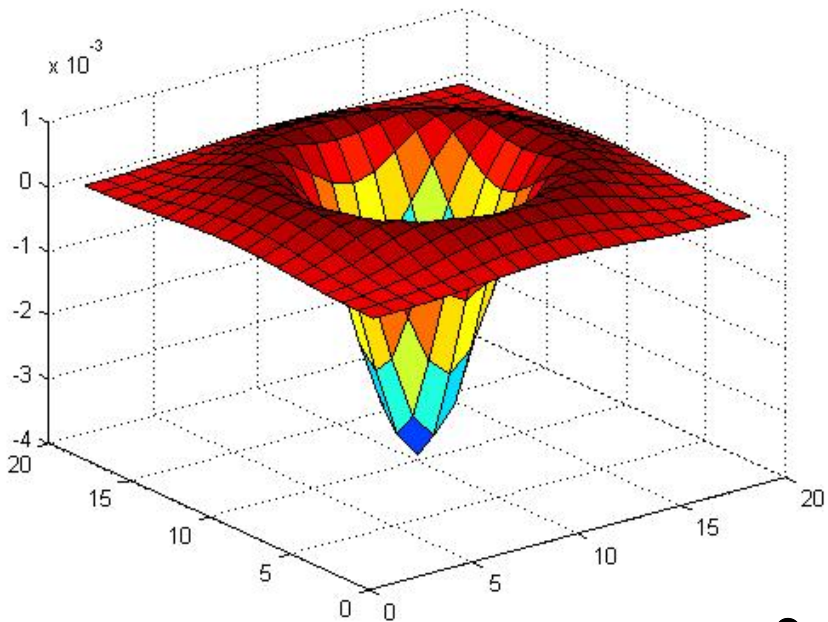
$=$

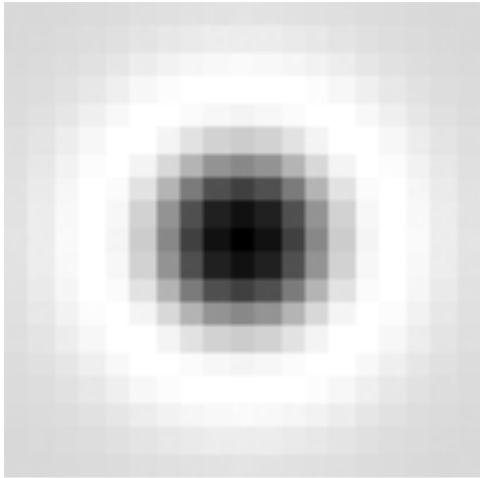


Source: N. Snavely

Blob filter

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





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

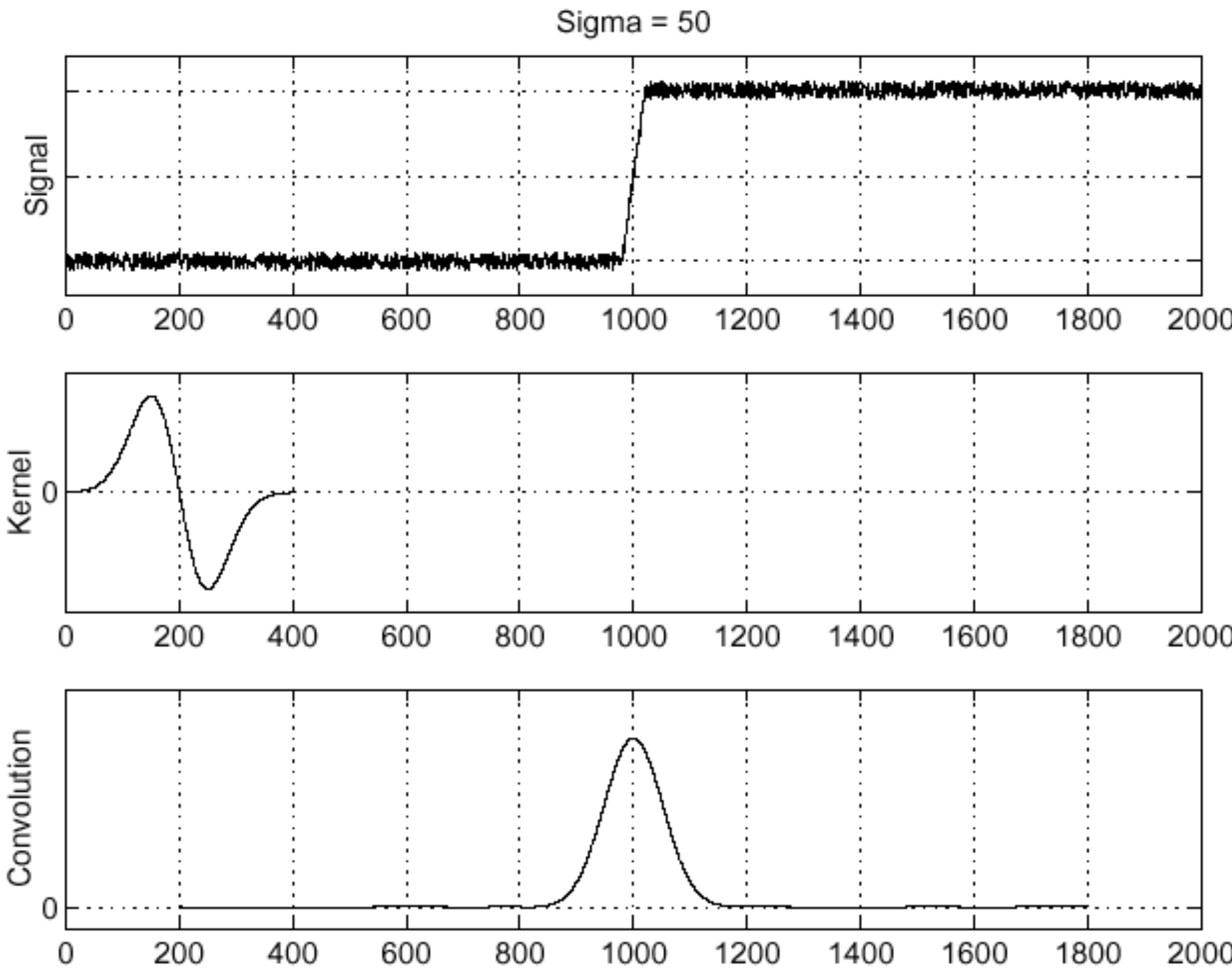
Slide: S. Lazebnik

Recall: Edge detection

f

$\frac{d}{dx} g$

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



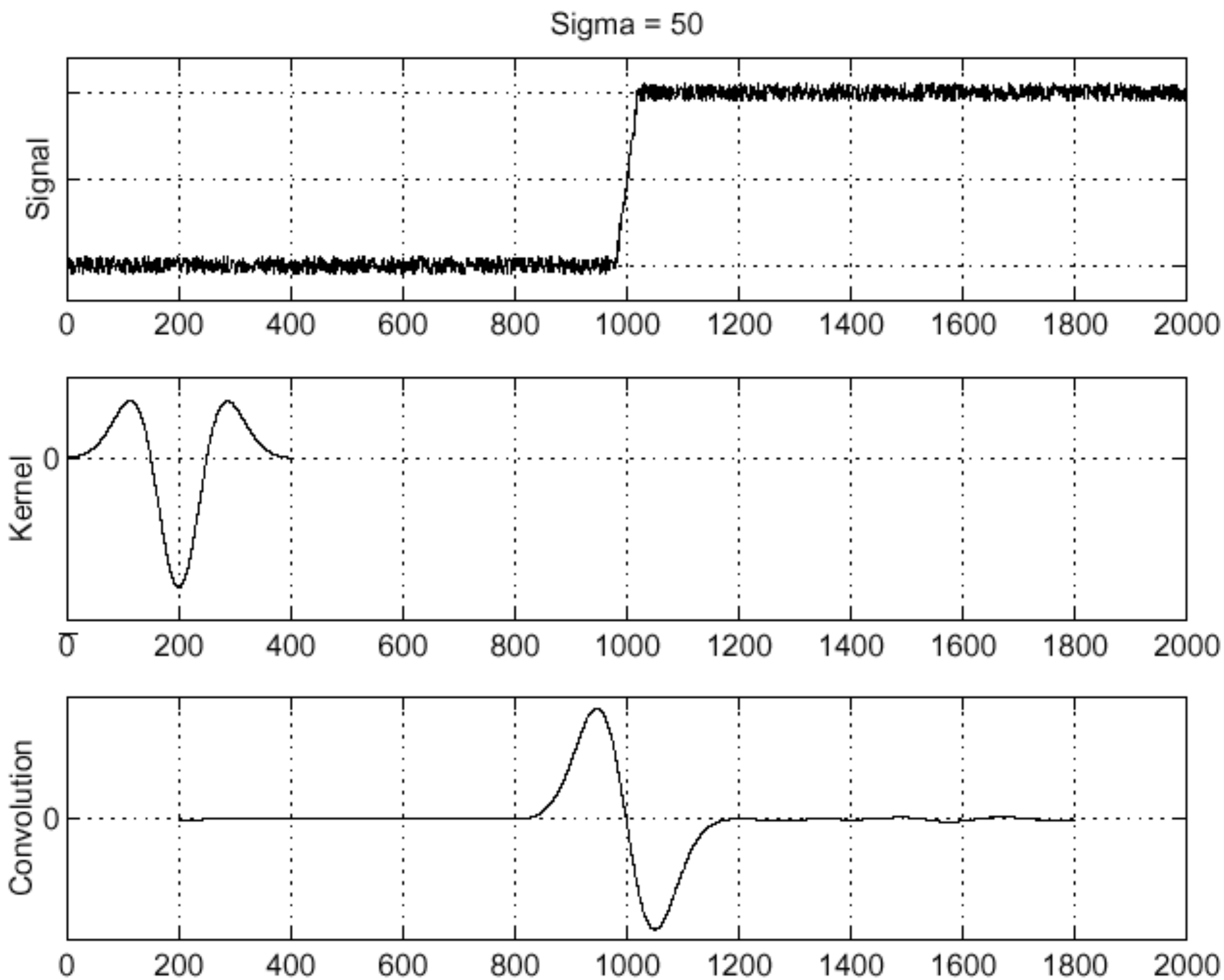
Edge

Derivative of Gaussian

Edge = maximum of derivative

Source: S. Seitz

Edge detection, Take 2



f — Edge

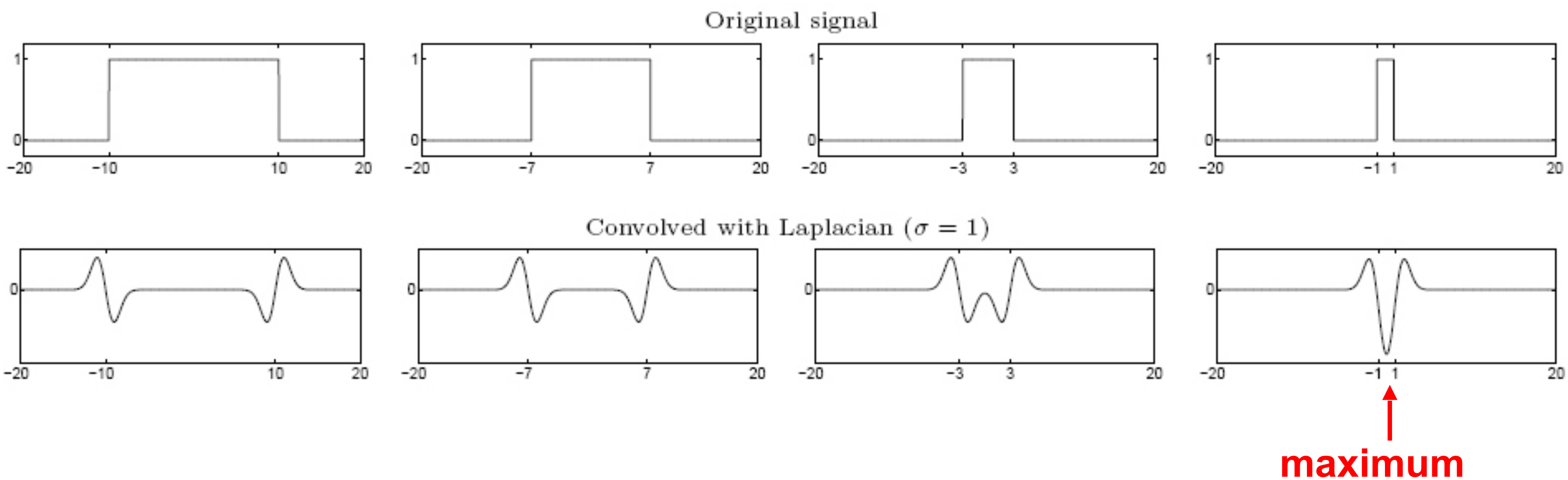
$\frac{d^2}{dx^2} g$ — Second derivative of Gaussian (Laplacian)

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

Source: S. Seitz

From edges to blobs

- Edge = ripple
- Blob = superposition of two ripples



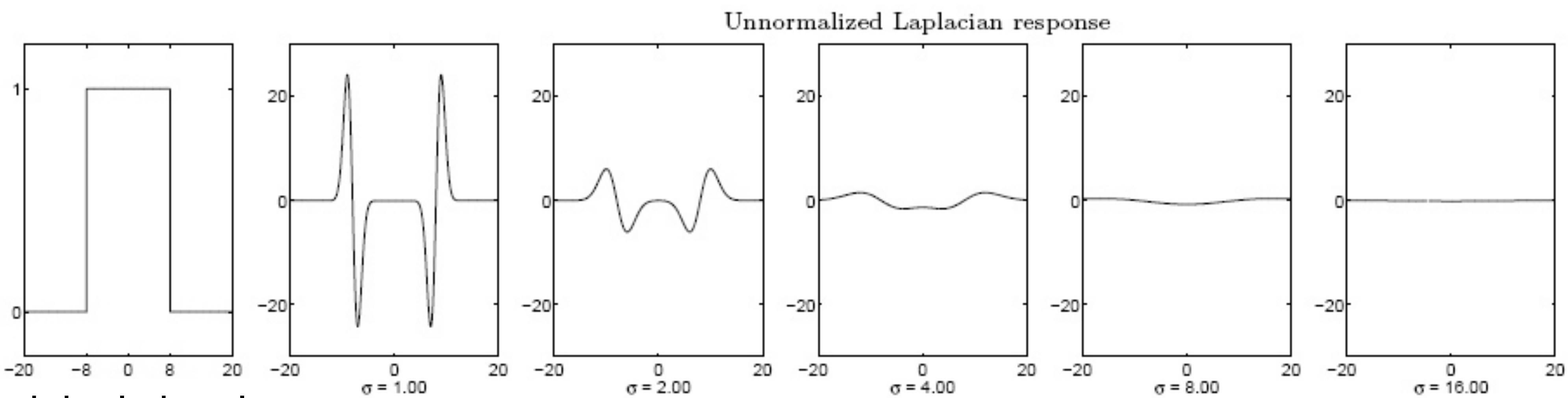
Spatial selection: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is "matched" to the scale of the blob

Slide: S. Lazebnik

Scale selection

- We want to find the characteristic scale of the blob by convolving it with Laplacians at several scales and looking for the maximum response

- However, Laplacian response decays as scale increases:



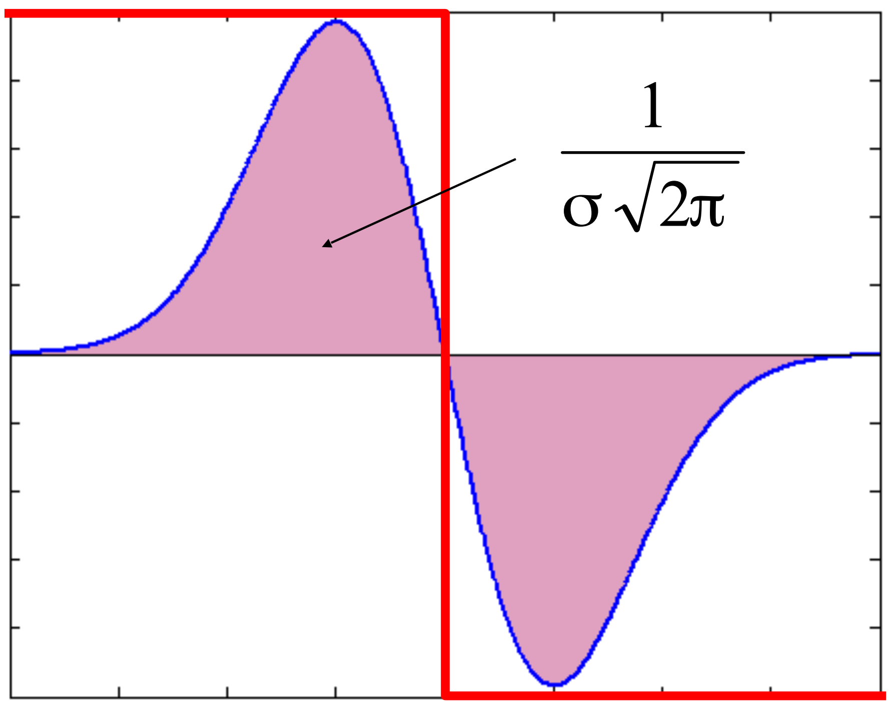
original signal
(radius=8)

increasing σ

Slide: S. Lazebnik

Scale normalization

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



Slide: S. Lazebnik

Scale normalization

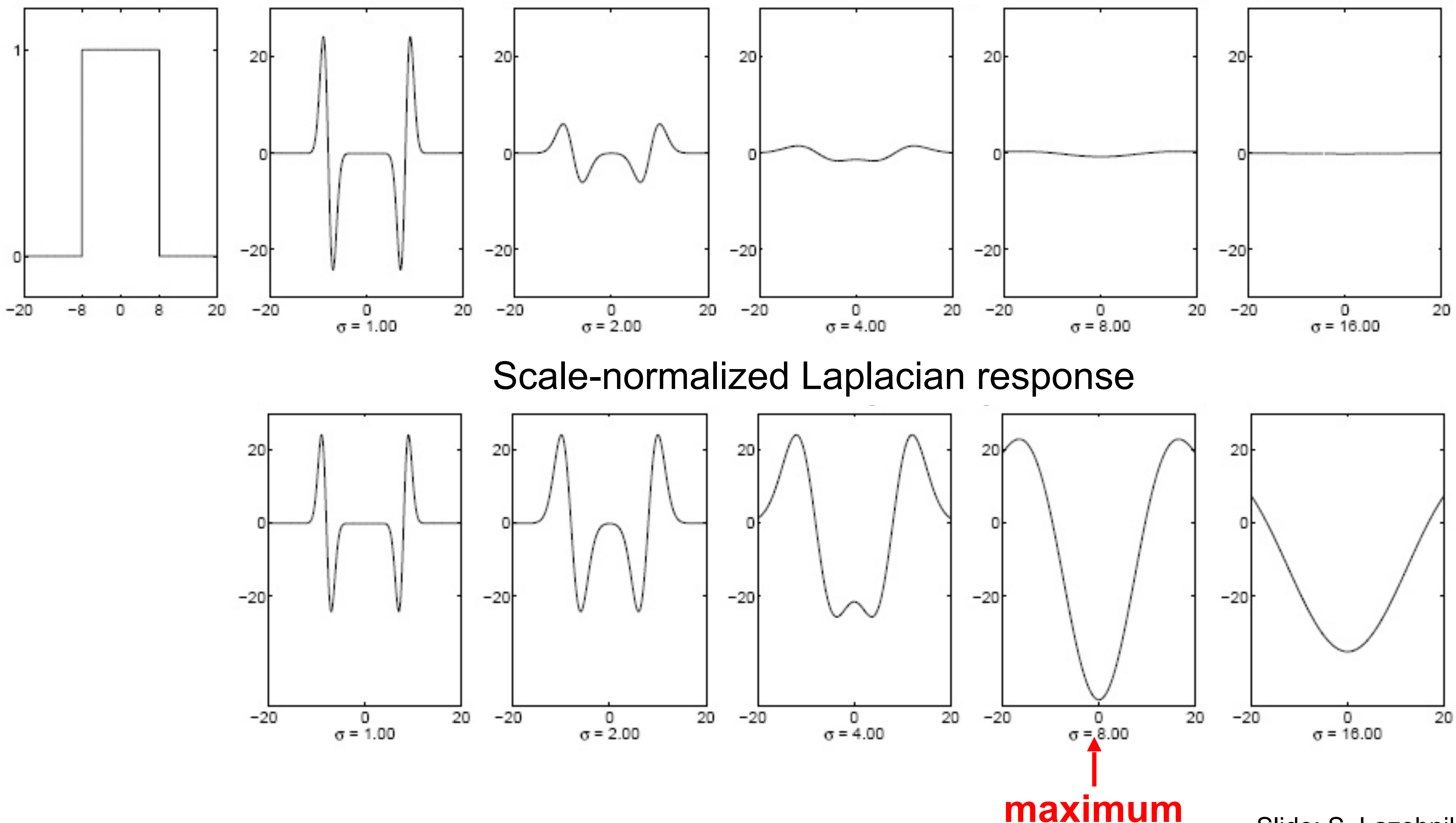
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases
- To keep response the same (scale-invariant), must multiply Gaussian derivative by σ
- Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

Effect of scale normalization

Original signal

Unnormalized Laplacian response

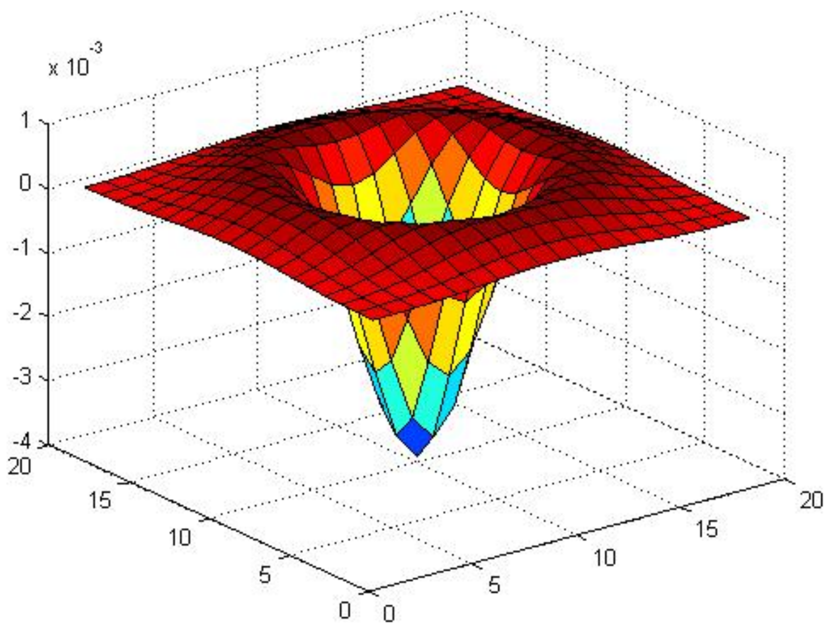
Scale-normalized Laplacian response

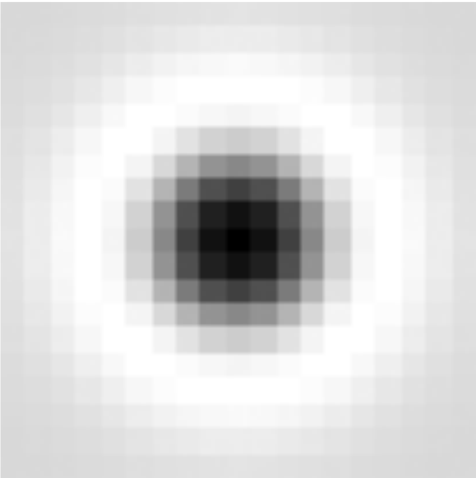


Slide: S. Lazebnik

Blob detection in 2D

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



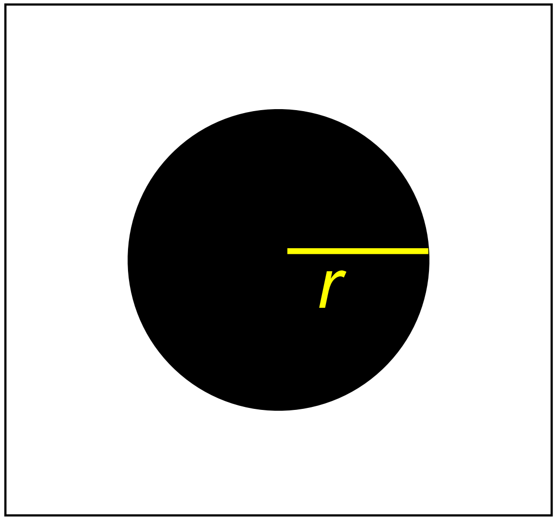


Scale-normalized: $$\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

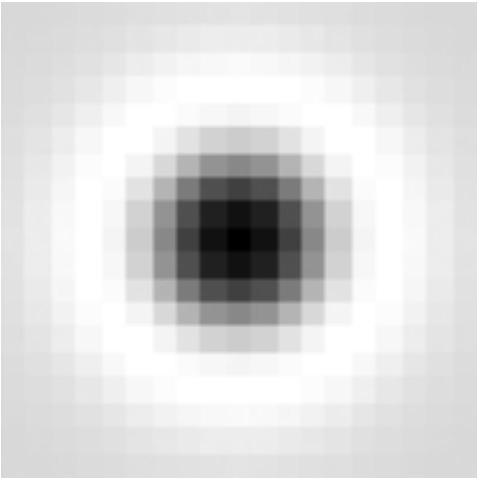
Slide: S. Lazebnik

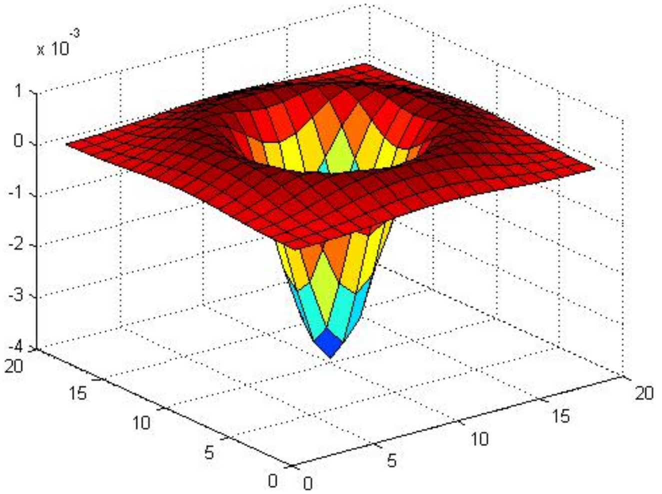
Scale selection

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



image





Laplacian

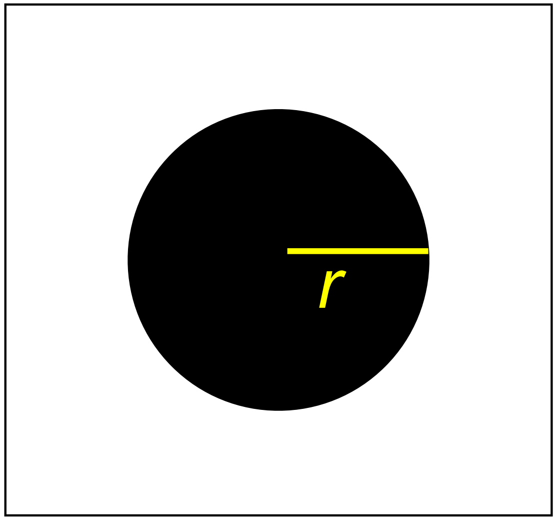
Slide: S. Lazebnik

Scale selection

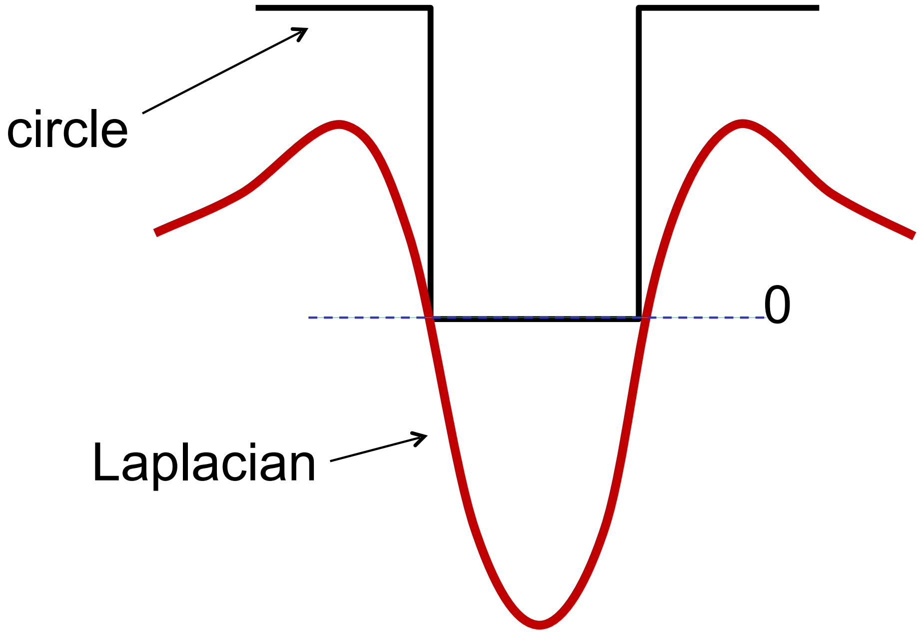
- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- For maximum response: align the zeros of the Laplacian with the circle
- The Laplacian in 2-D is given by (up to scale):

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

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



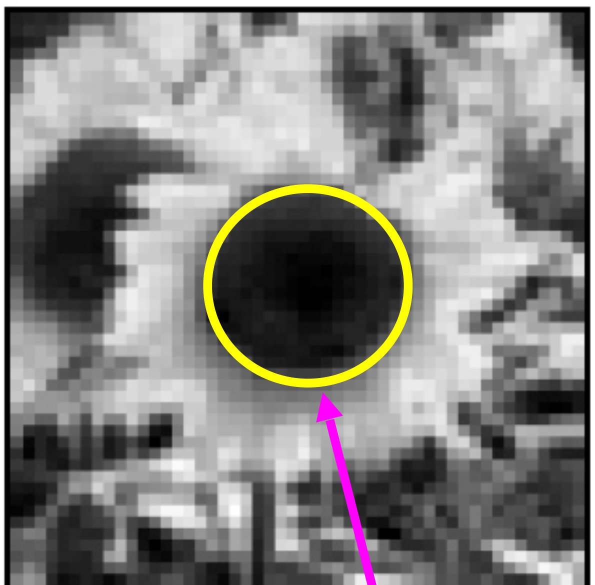
image

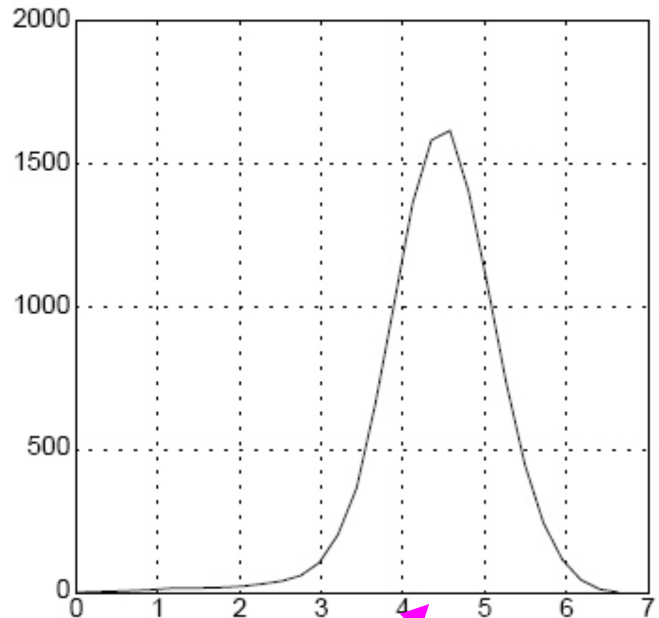


Slide: S. Lazebnik

Characteristic scale

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





characteristic scale

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

Slide: S. Lazebnik

Scale-space blob detector

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

Scale-space blob detector

Slide: S. Lazebnik

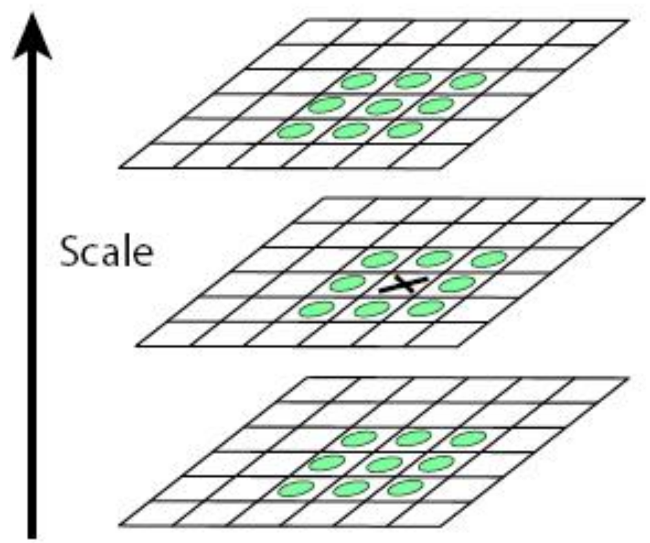
Scale-space blob detector

sigma = 11.9912

Slide: S. Lazebnik

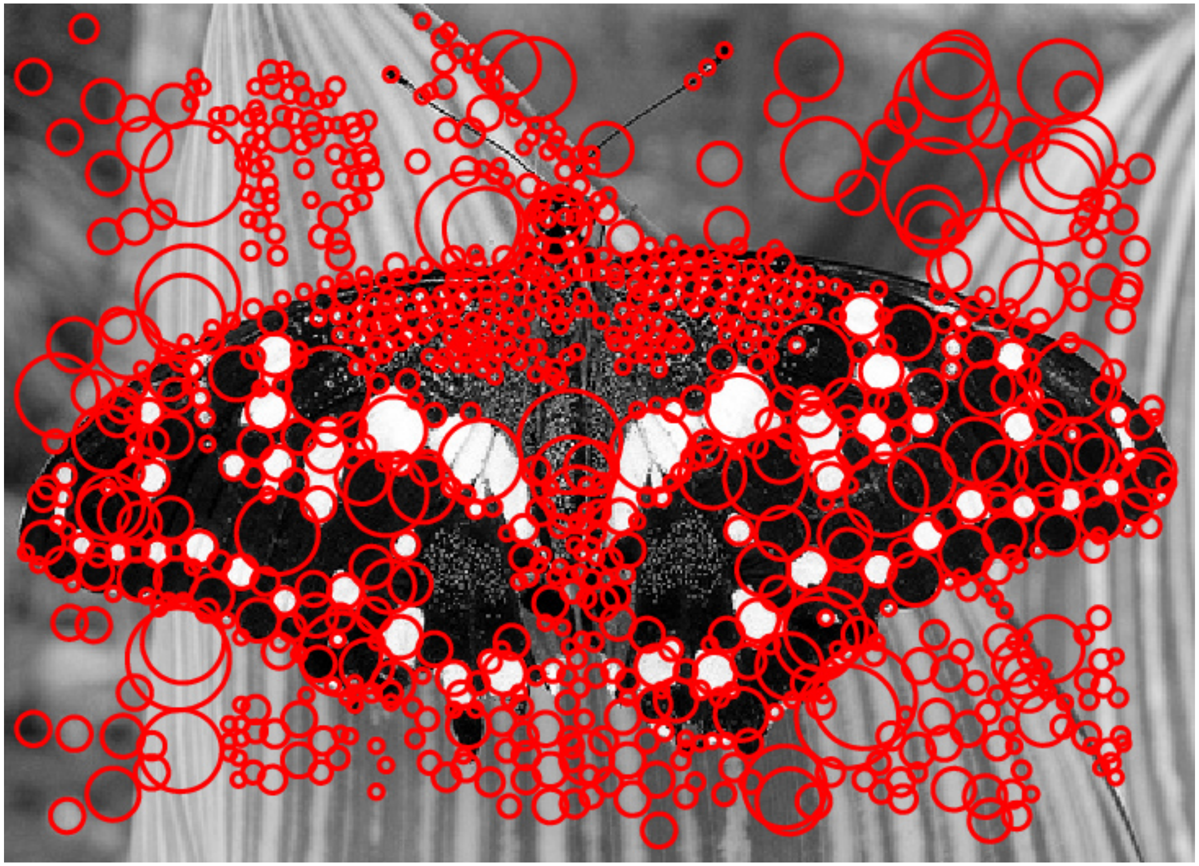
Scale-space blob detector

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



Slide: S. Lazebnik

Scale-space blob detector: Example

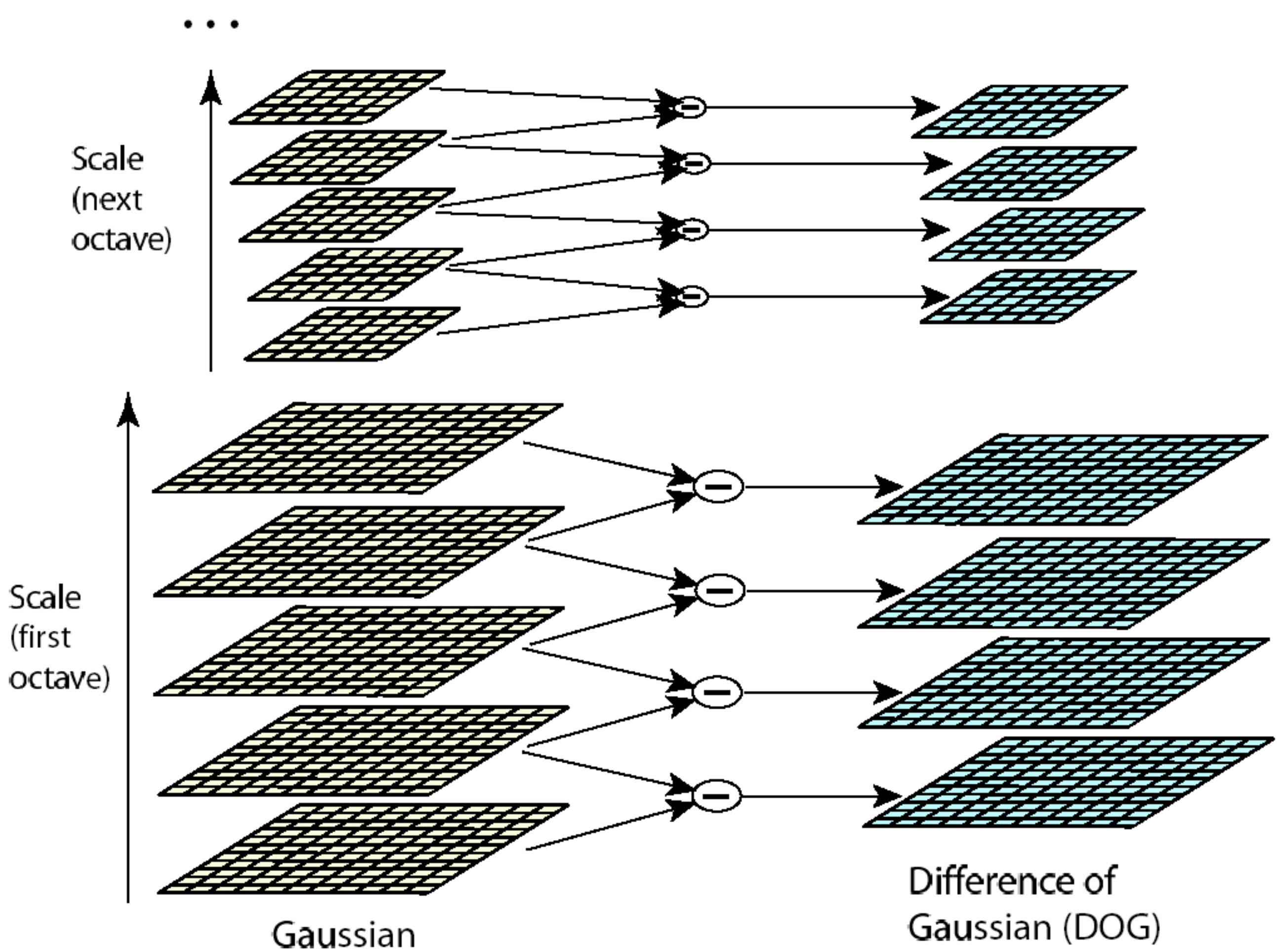


Slide: S. Lazebnik

Efficient implementation

- Laplacian of Gaussian can be approximated by Difference of Gaussians
  - Assignment 1, question 3

Efficient implementation

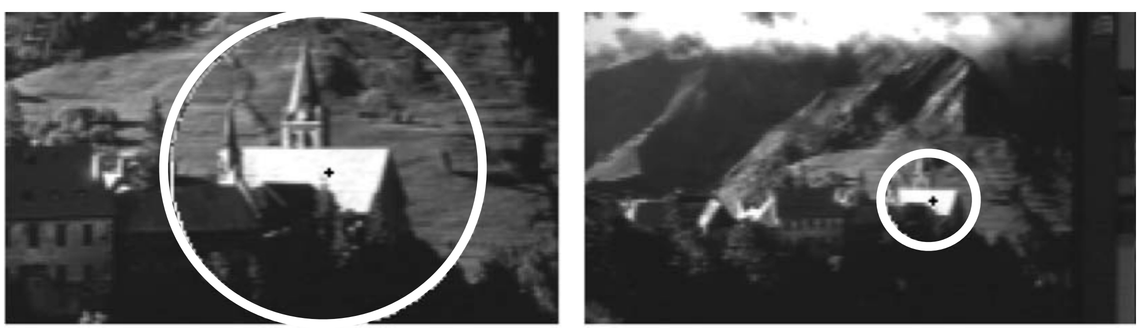


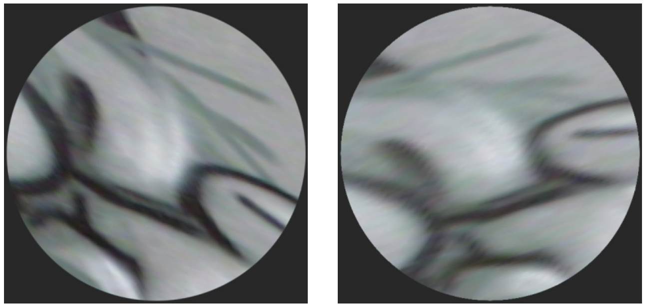
David G. Lowe. ["Distinctive image features from scale-invariant keypoints."]() *IJCV* 60 (2), pp. 91-110, 2004.

Slide: S. Lazebnik

From feature detection to feature description

- Scaled and rotated versions of the same neighborhood will give rise to blobs that are related by the same transformation
- What to do if we want to compare the appearance of these image regions?
  - *Normalization*: transform these regions into same-size circles
  - Problem: rotational ambiguity

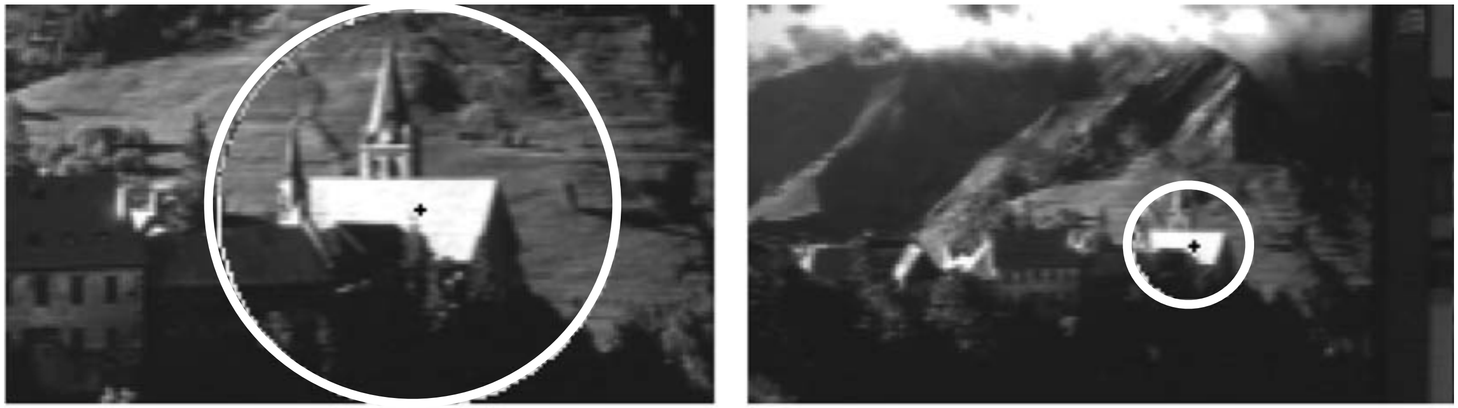


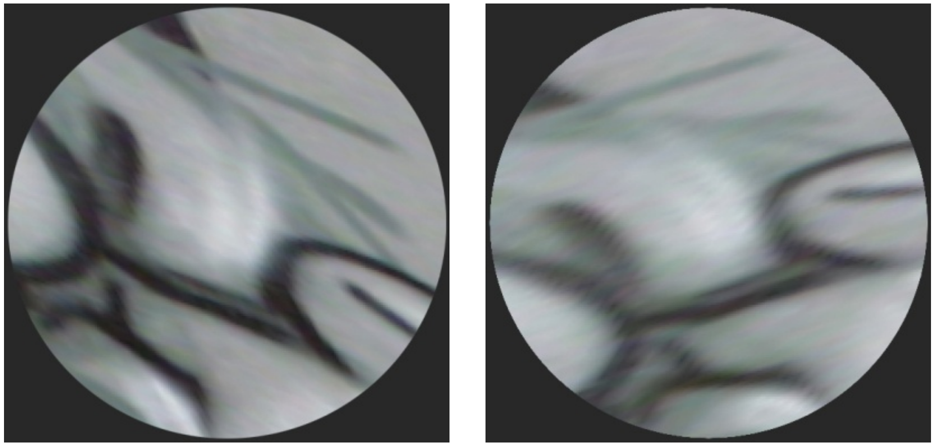


Slide: S. Lazebnik

SIFT descriptors

After blob detection and scale normalization

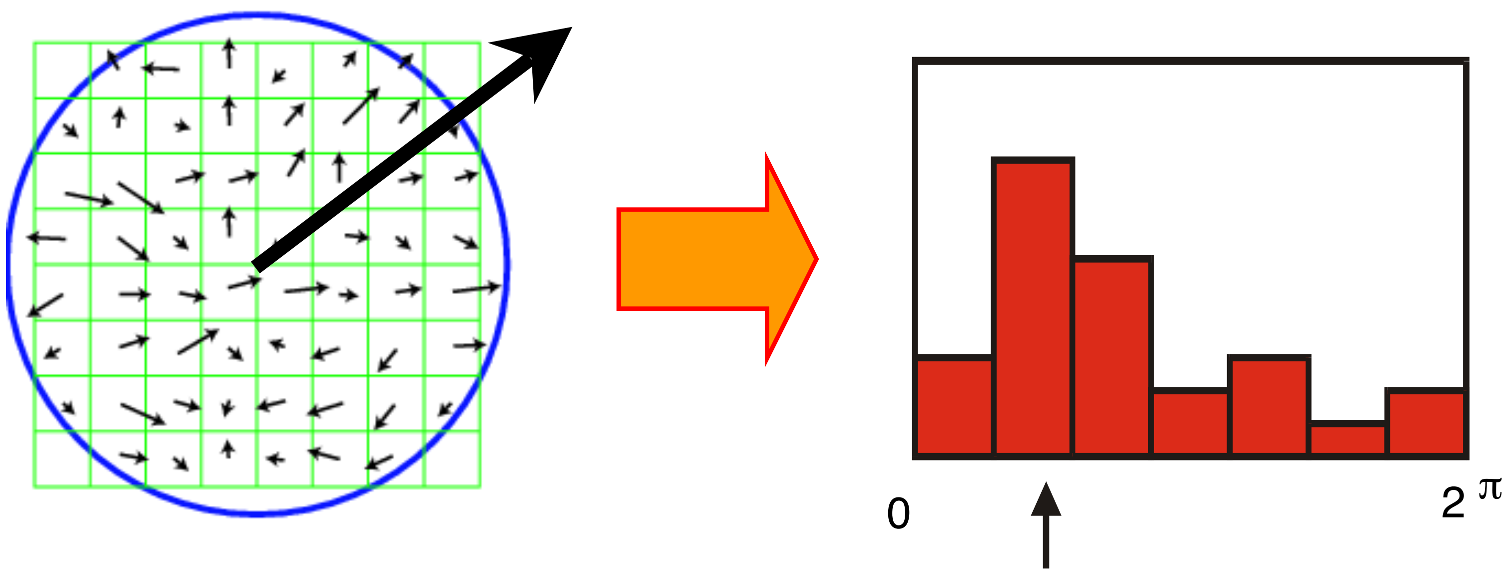




Slide: S. Lazebnik

Eliminating rotation ambiguity

- To assign a unique orientation to circular image windows:
  - Create histogram of local gradient directions in the patch
  - Assign canonical orientation at peak of smoothed histogram



Slide: S. Lazebnik

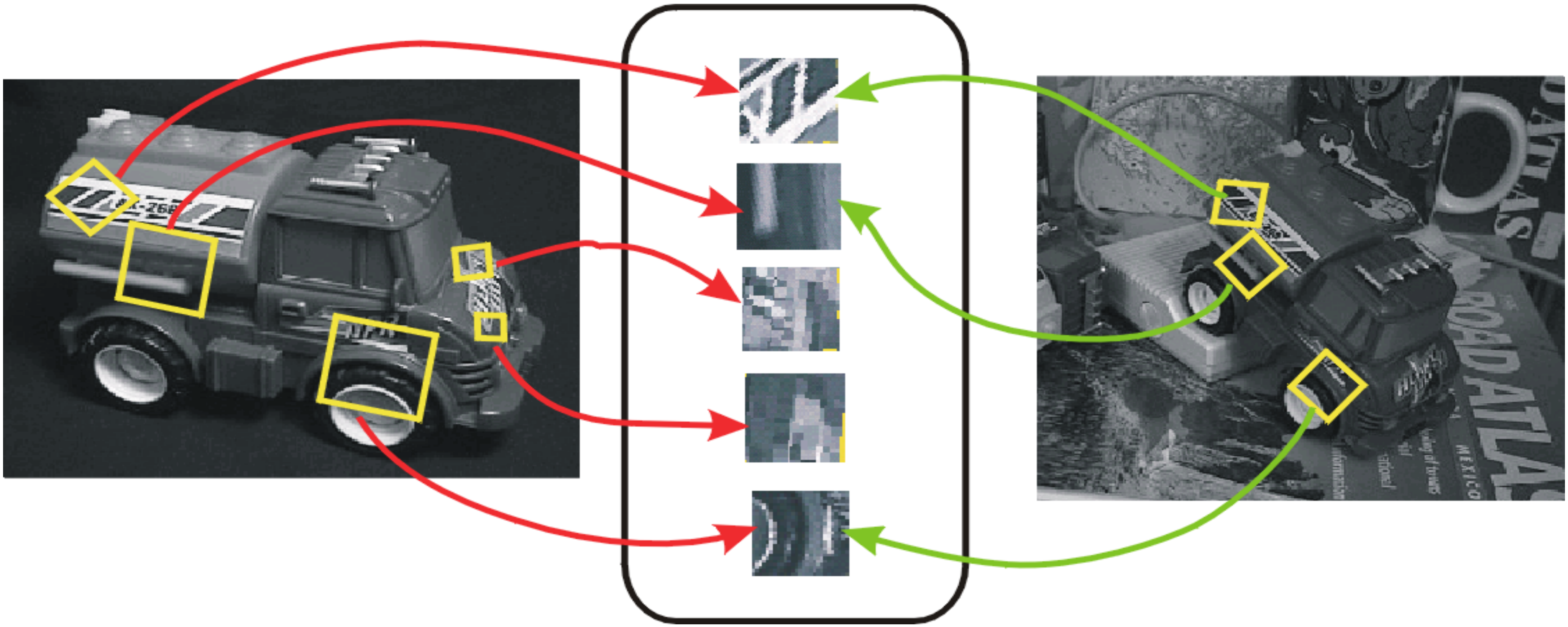
SIFT detected features

- Detected features with characteristic scales and orientations:


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

Slide: S. Lazebnik

From feature detection to feature description



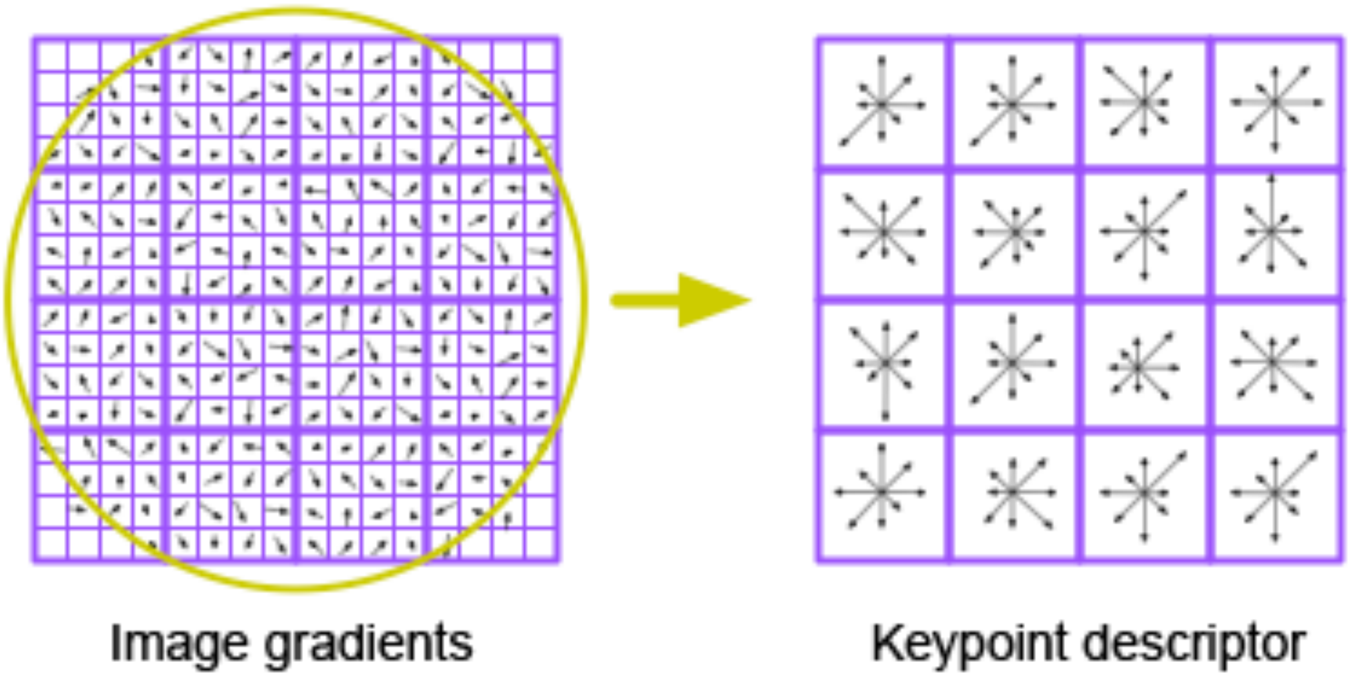
Slide: S. Lazebnik

Properties of Feature Descriptors

- Easily compared (compact, fixed-dimensional)
- Easily computed
- Invariant
  - Translation
  - Rotation
  - Scale
  - Change in image brightness
  - Change in perspective?

SIFT Descriptor

- Divide 16×16 window into 4×4 grid of cells
- Compute an orientation histogram for each cell
  - 16 cells * 8 orientations = 128-dimensional descriptor



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

Properties of SIFT

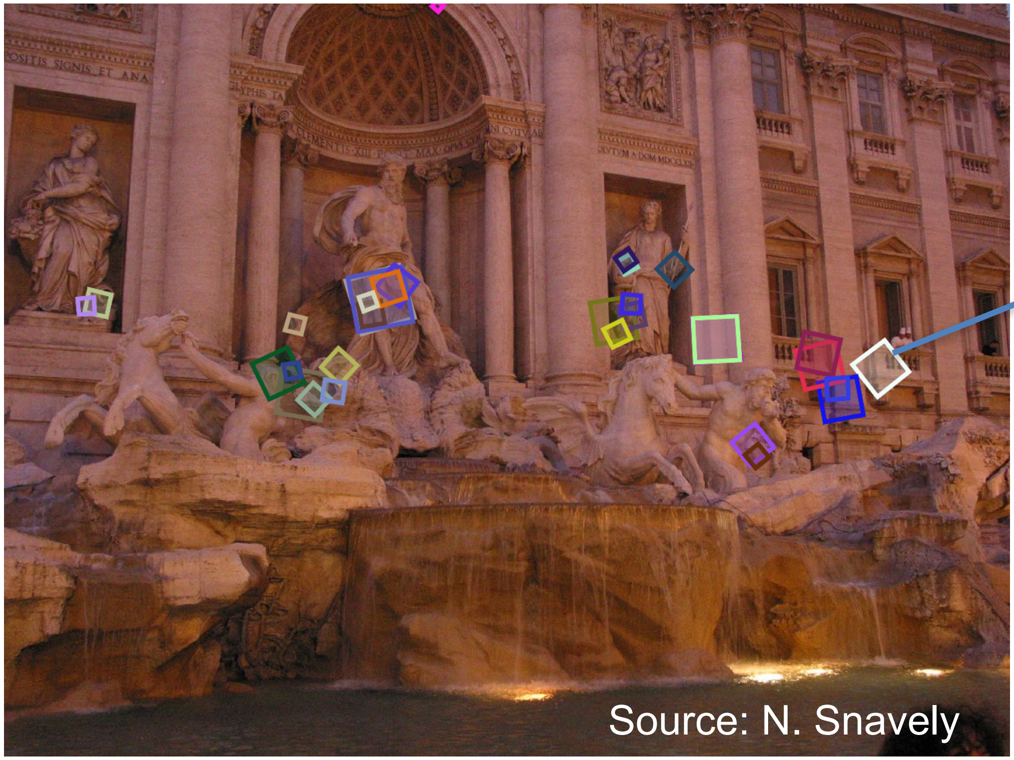
Extraordinarily robust detection and description technique

- Handles changes in viewpoint (~ 60 degree out-of-plane rotation)
- Handles significant changes in illumination (sometimes even day vs night)
- Fast and efficient—can run in real time
- Lots of code available

Source: N. Snavely

Feature descriptors

Think of a feature as some non-linear filter that maps pixels to 128D feature



128-dimensional vector **x**

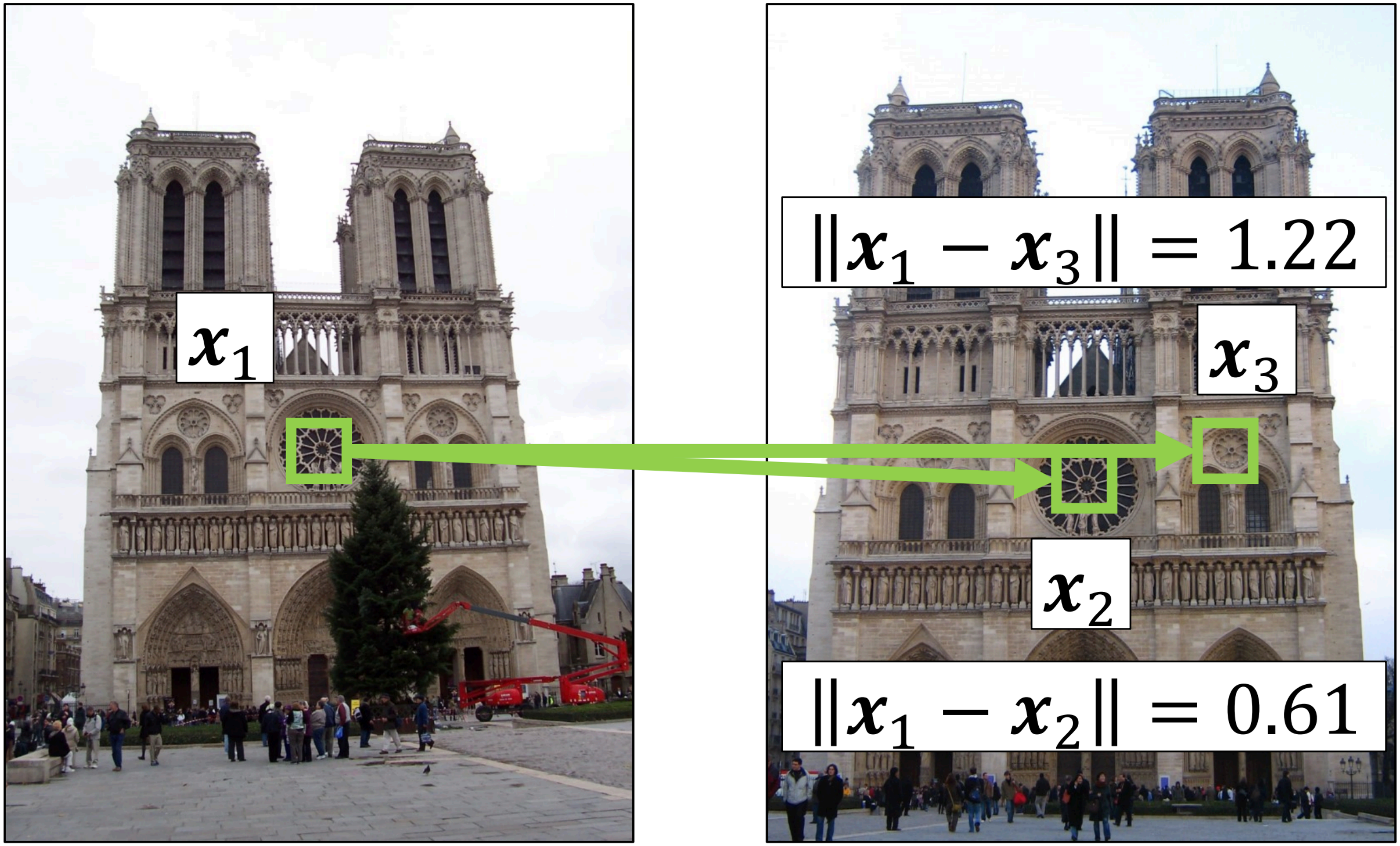
Two use cases:

1. Instance Matching

2. Category recognition

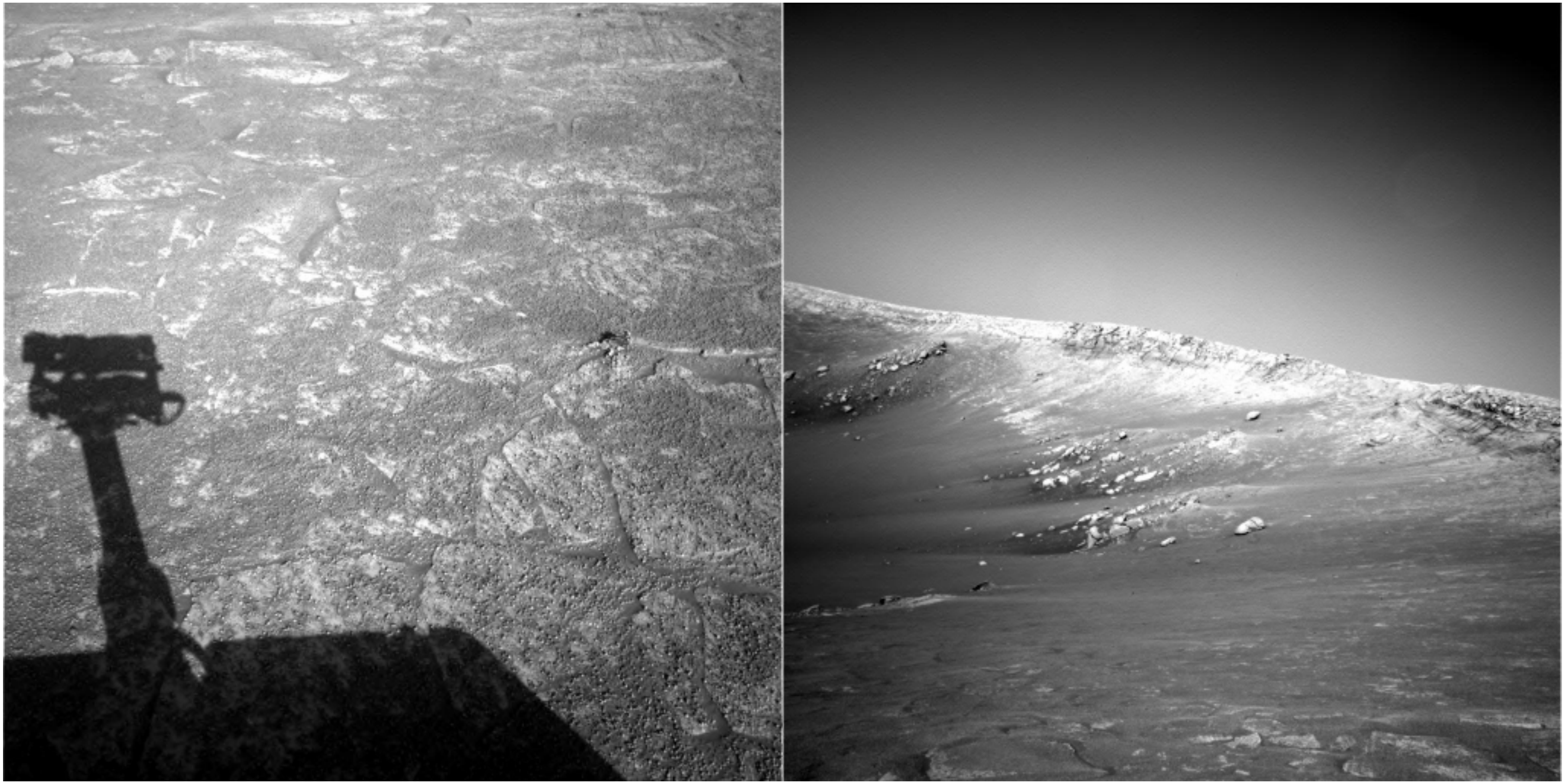
Credit: David Fouhey

Use case 1: Instance matching



Credit: James Hays

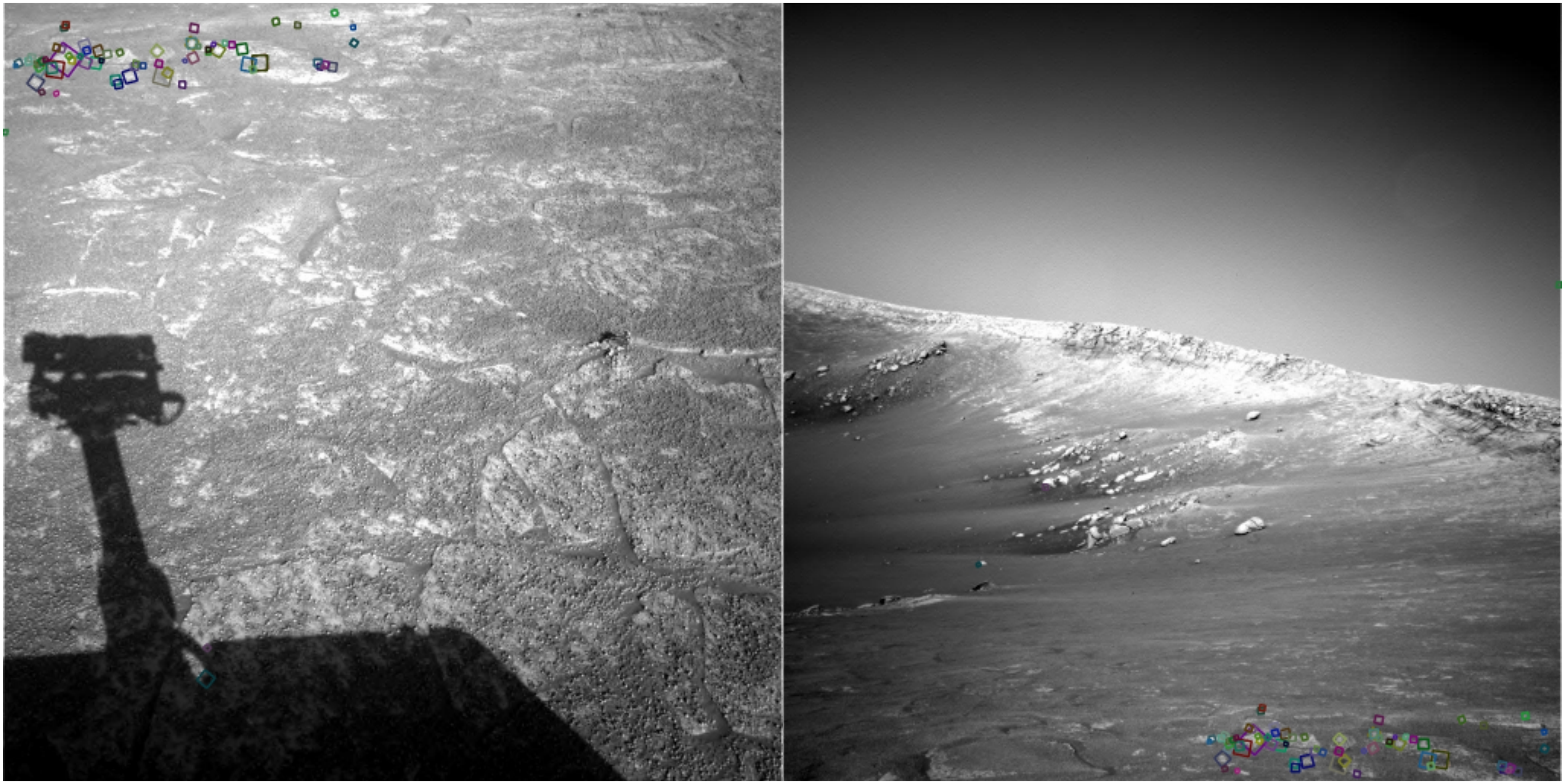
Use case 1: instance matching



NASA Mars Rover images

Slide credit: S. Lazebnik

Use case 1: instance matching (look for tiny colored squares)



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

Slide credit: S. Lazebnik

Use case 2: Category recognition

- Extract features from set of images (either densely or at key points)

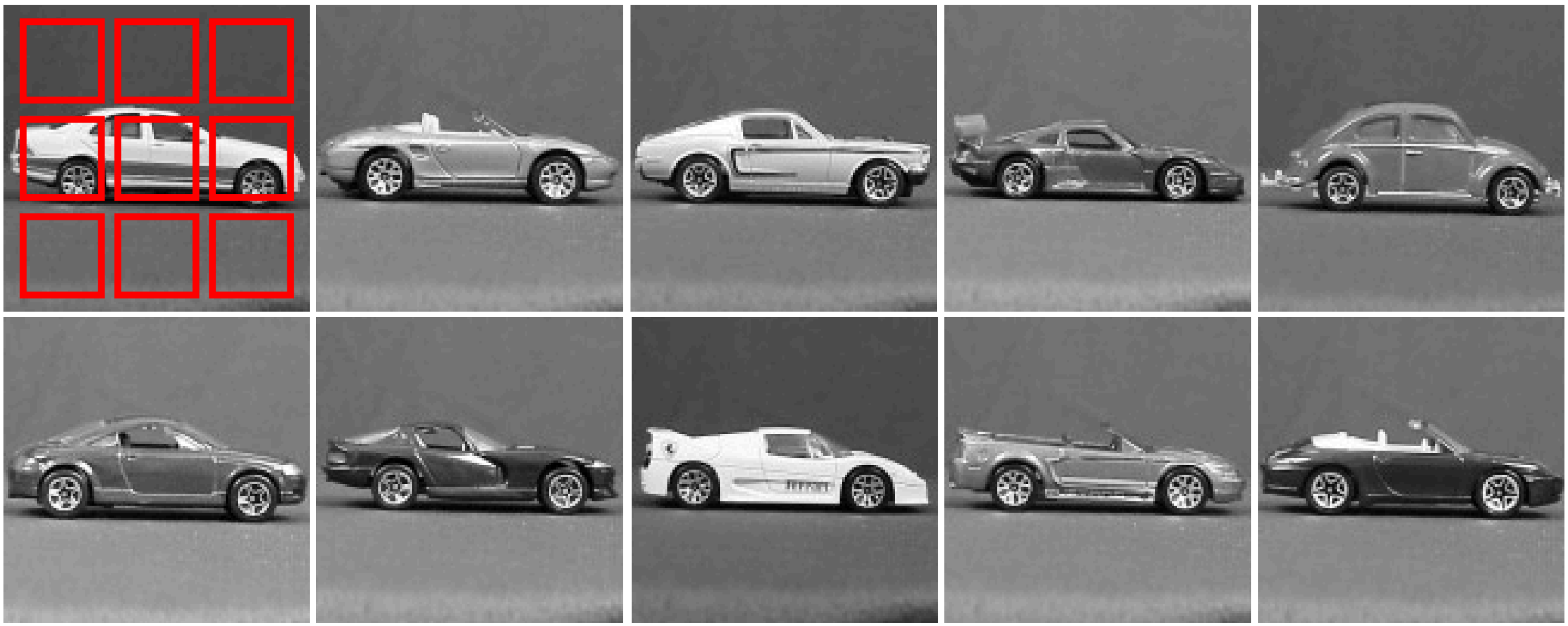


Figure: B. Liebe

Credit: David Fouhey

Use case 2: category recognition

- Build codebook of “concepts”

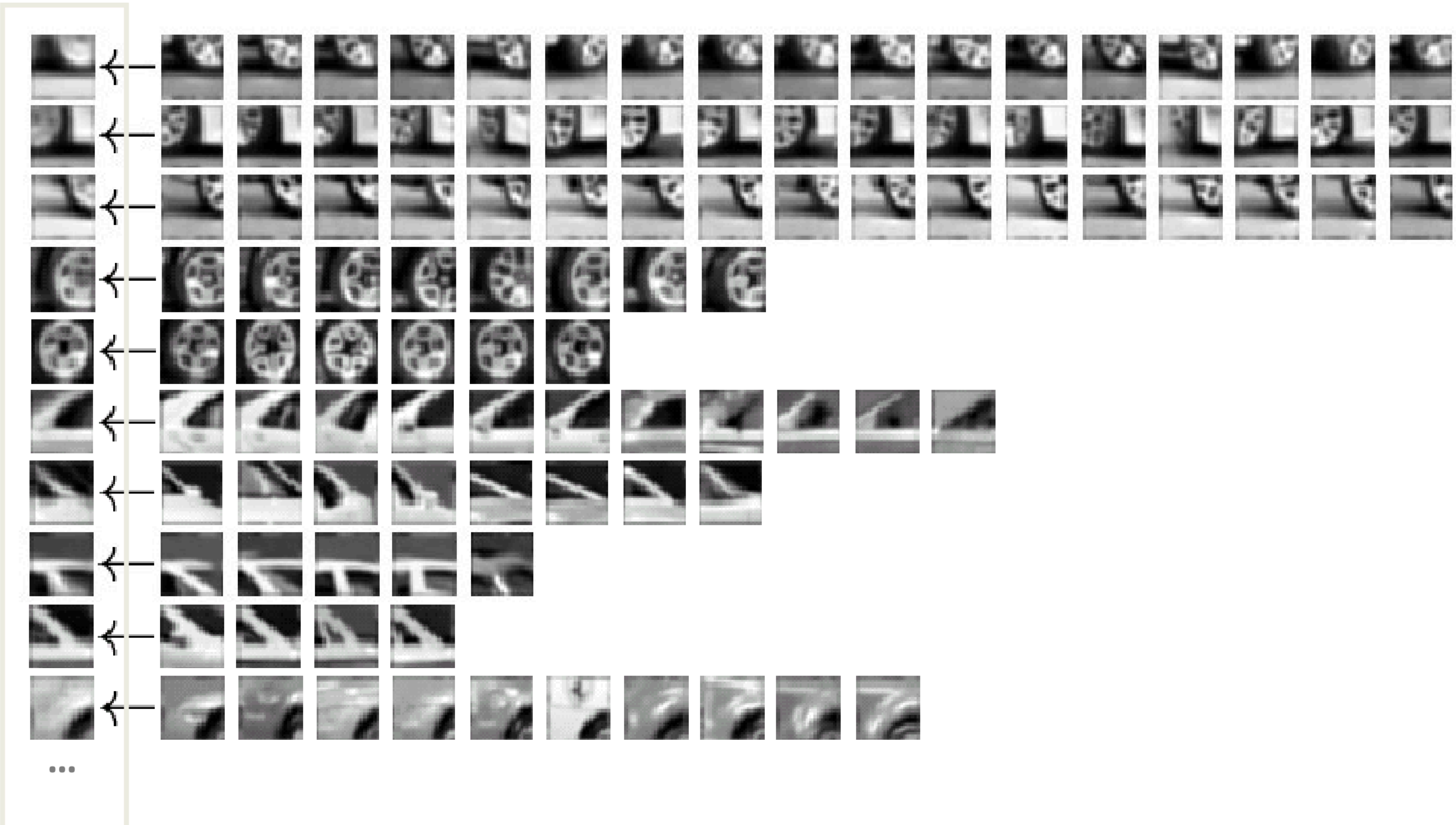
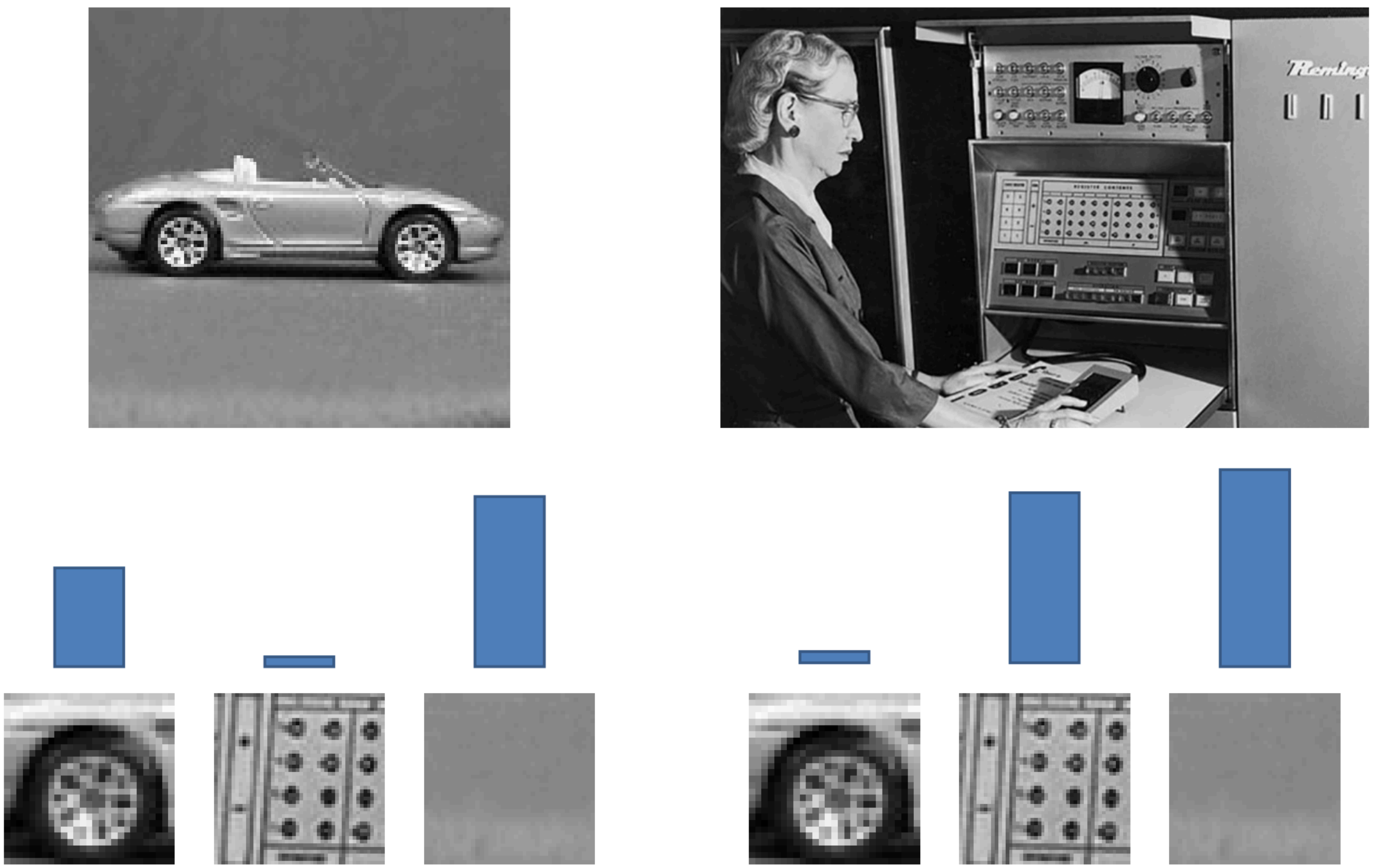


Figure: B. Liebe

Credit: David Fouhey

Use case 2: category recognition

- Represent image as histogram of concepts



Credit: David Fouhey

Next class: fitting, Hough transforms, RANSAC