Local features

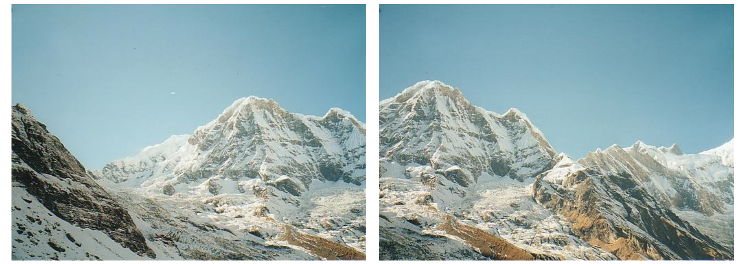
Subhransu Maji

CMPSCI 670: Computer Vision

October 4, 2016

Why extract features?

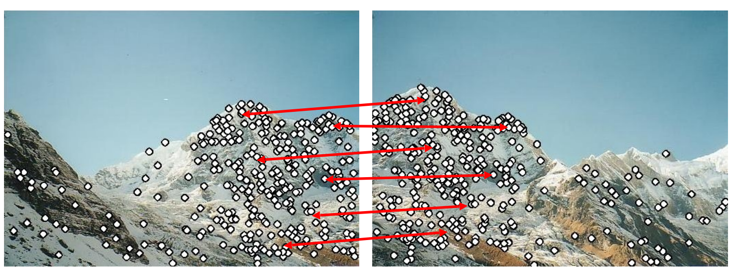
- Motivation: panorama stitching
  - We have two images – how do we combine them?



Slide credit: L. Lazebnik

Why extract features?

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



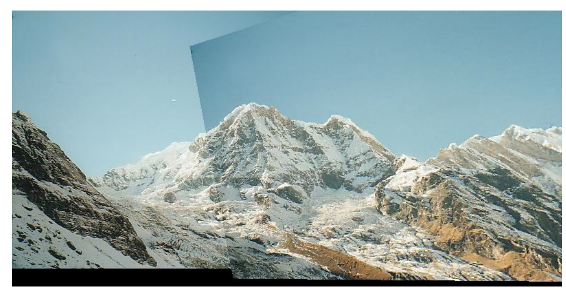
Step 1: extract features

Step 2: match features

Slide credit: L. Lazebnik

Why extract features?

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



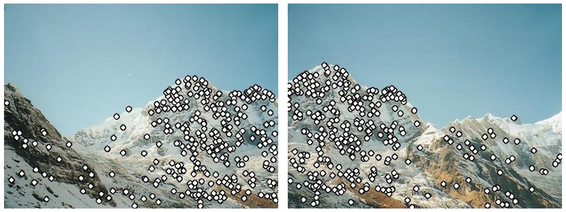
Step 1: extract features

Step 2: match features

Step 3: align images

Slide credit: L. Lazebnik

Characteristics of good features



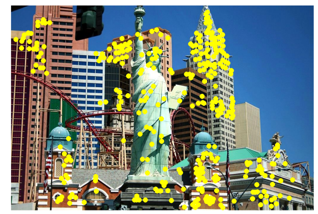
- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature is distinctive
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image;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

Slide credit: L. Lazebnik

Applications

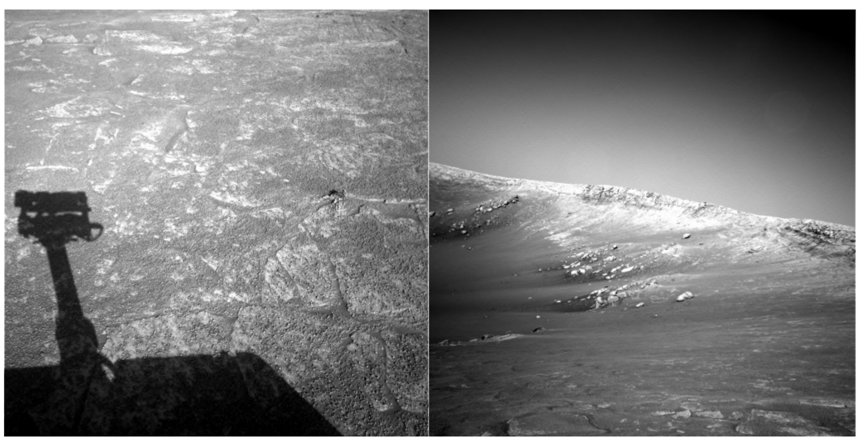
Feature points are used for:

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



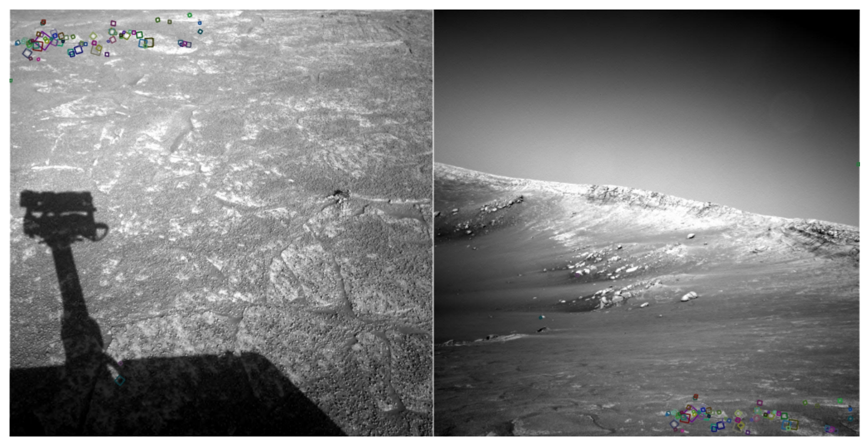
Slide credit: L. Lazebnik

A hard feature matching problem



NASA Mars Rover images

Answer below (look for tiny colored squares…)



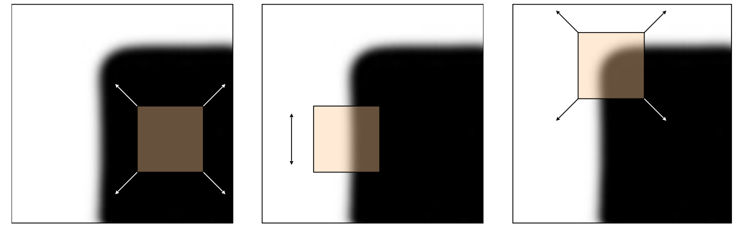
NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

Overview

- Detecting features
  - Corners — translational invariance
  - Blobs — scale and translational invariance
  - Adding rotational invariance

Corner detection: basic idea

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



"flat" region: no change in all directions

"edge": no change along the edge direction

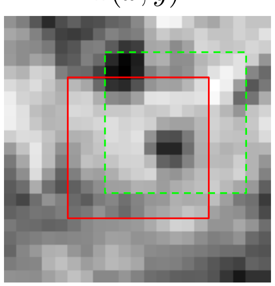
"corner": significant change in all directions

Corner detection: mathematics

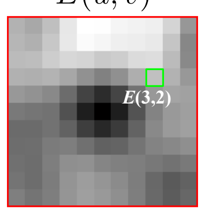
Change in appearance of window *W* for the shift [*u*,*v*]:

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

$I(x,y)$



$E(u,v)$

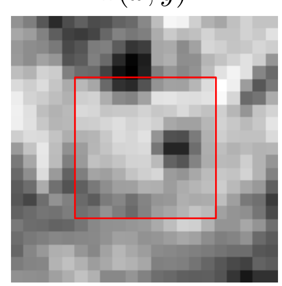


Corner detection: mathematics

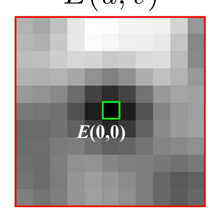
Change in appearance of window *W* for the shift [*u*,*v*]:

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

$I(x,y)$



$E(u,v)$



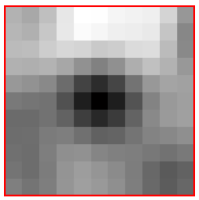
Corner detection: mathematics

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

$E(u, v)$



Corner detection: mathematics

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

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

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

$$\begin{aligned} E(u, v) &= \sum_{(x,y)\in W} [I(x+u, y+v) - I(x,y)]^2 \\ &\simeq \sum_{(x,y)\in W} [I(x,y) + I_x u + I_y v - I(x,y)]^2 \\ &= \sum_{(x,y)\in W} [I_x u + I_y v]^2 \\ &= \sum_{(x,y)\in W} \left[I_x^2 u^2 + I_x I_y uv + I_y I_x uv + I_y^2 v^2\right] \end{aligned}$$

Corner detection: mathematics

The quadratic approximation can be written as

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

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

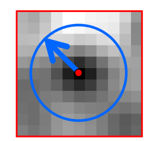
$$M = \begin{bmatrix} \sum_{x,y} I_x^2 & \sum_{x,y} I_x I_y \\ \sum_{x,y} I_x I_y & \sum_{x,y} I_y^2 \end{bmatrix}$$

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

Interpreting the second moment matrix

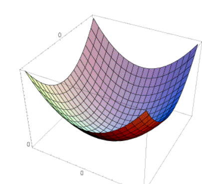
- The surface *E*(*u*,*v*) is locally approximated by a quadratic form. Let's try to understand its shape.
- Specifically, in which directions does it have the smallest/ greatest change?

$E(u, v)$



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

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



Interpreting the second moment matrix

First, consider the axis-aligned case
(gradients are either horizontal or vertical)

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

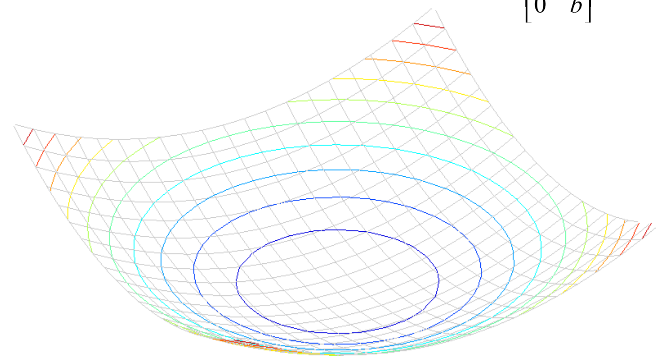
If either *a* or *b* is close to 0, then this is **not** a corner, so look for locations where both are large.

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.

$$M \rightarrow \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$



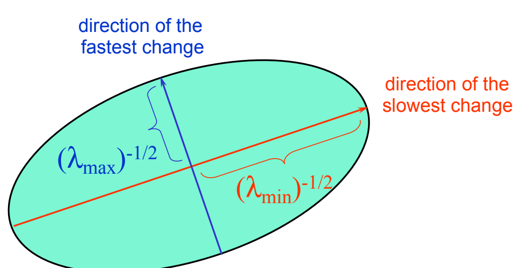
Interpreting the second moment matrix

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

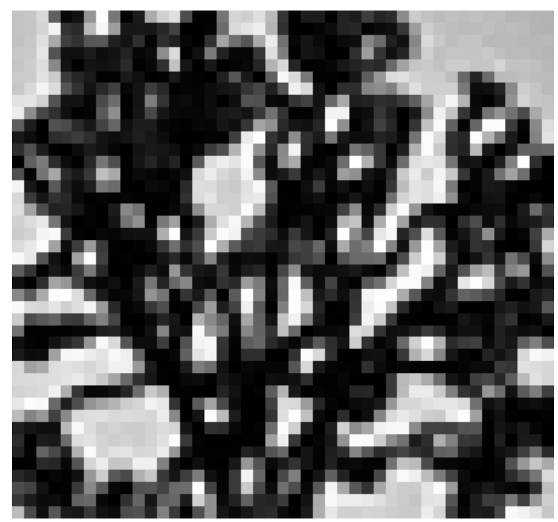
This is the equation of an ellipse.

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

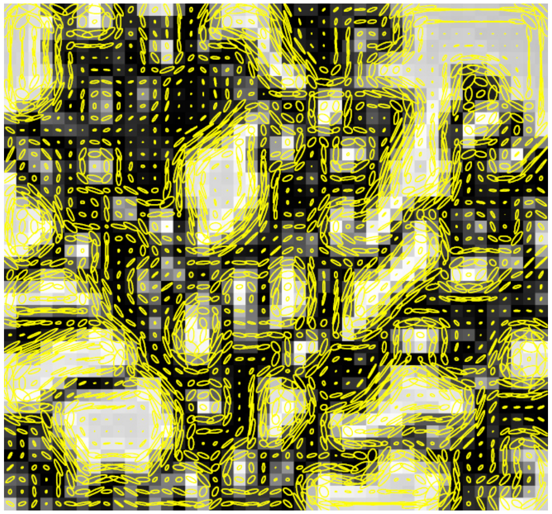
The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by *R*



Visualization of second moment matrices

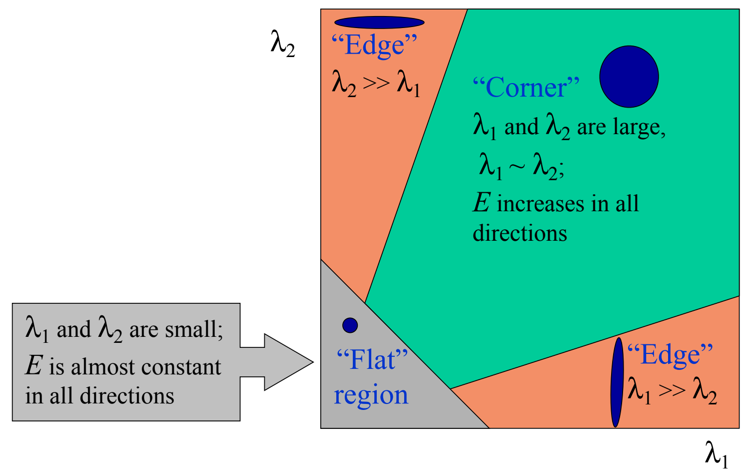


Visualization of second moment matrices



Interpreting the eigenvalues

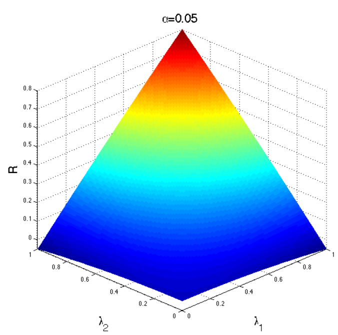
Classification of image points using eigenvalues of *M*:

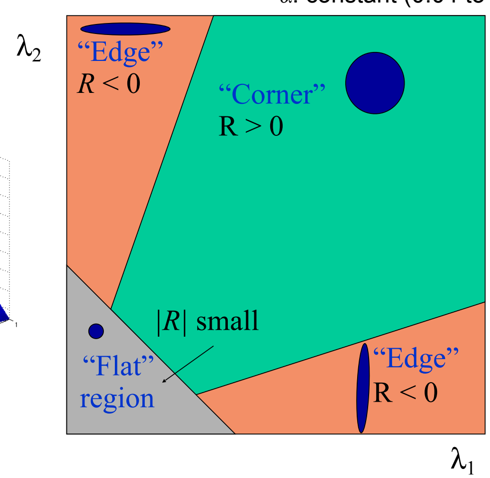


Corner response function

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

α: constant (0.04 to 0.06)





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.

The Harris corner detector

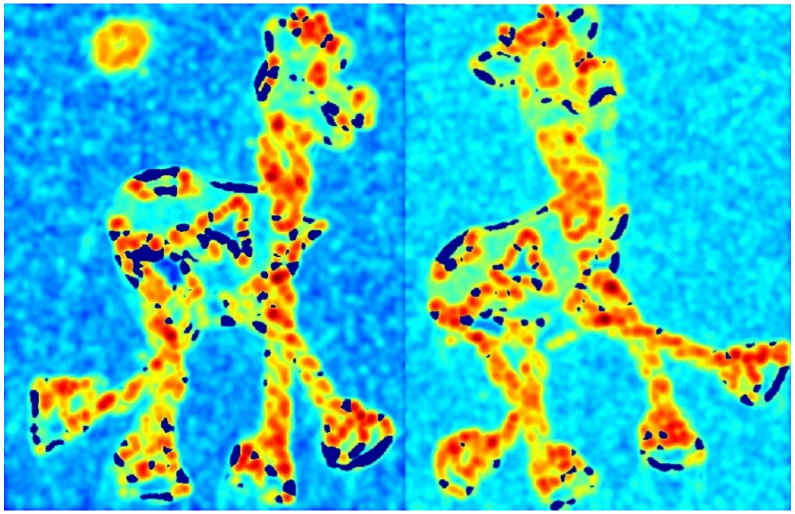
1. Compute partial derivatives at each pixel
2. Compute second moment matrix M in a Gaussian window around each pixel
3. Compute corner response function R

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

Harris detector: steps

Harris detector: steps

Compute corner response R



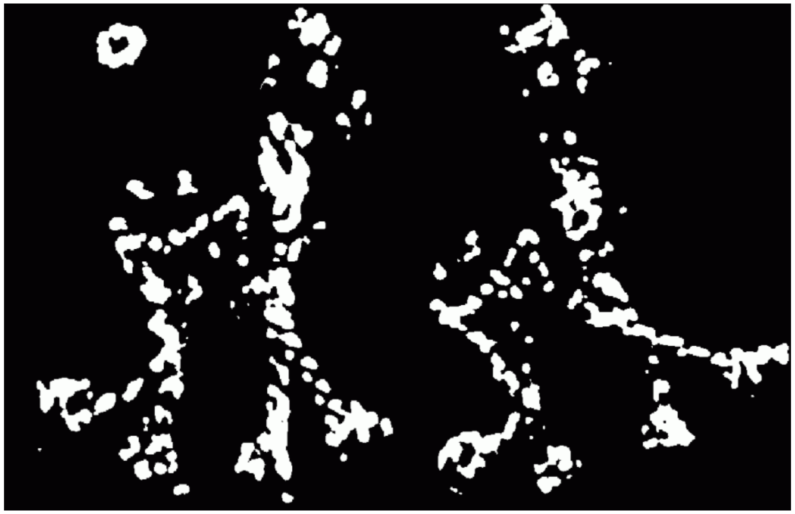
The Harris corner detector

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 (non-maximum suppression)

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

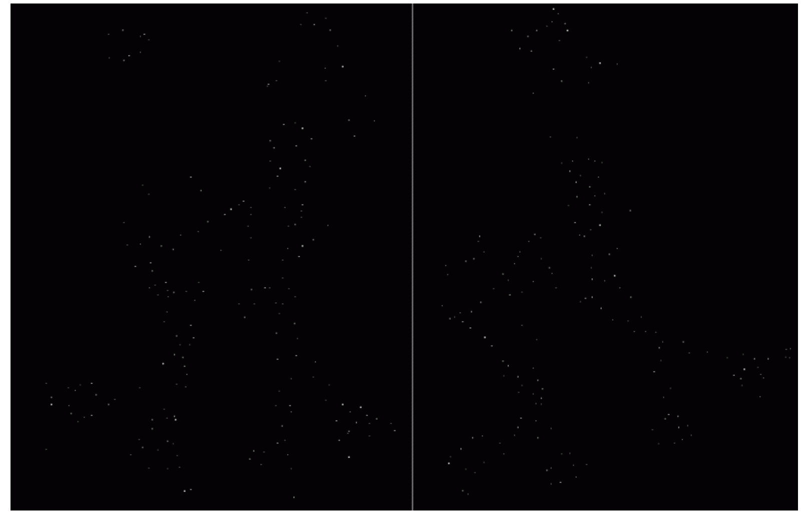
Harris Detector: Steps

Find points with large corner response: R > threshold



Harris Detector: Steps

Take only the points of local maxima of R



Harris Detector: Steps

Further thoughts and readings…

- Original corner detector paper
  - C.Harris and M.Stephens, **"A Combined Corner and Edge Detector."** Proceedings of the 4th Alvey Vision Conference, 1988
- Other corner functions
  - Can you think of other $f(\lambda_1, \lambda_2)$ that work for finding corners?

Local features

Subhransu Maji

CMPSCI 670: Computer Vision

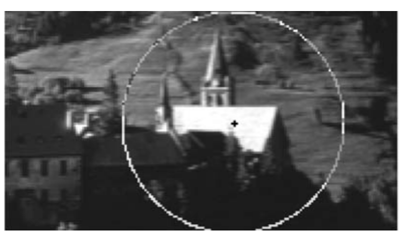
October 6, 2016

Overview

- Detecting features
  - Corners — translational invariance
  - Blobs — scale and translational invariance
  - Adding rotational invariance

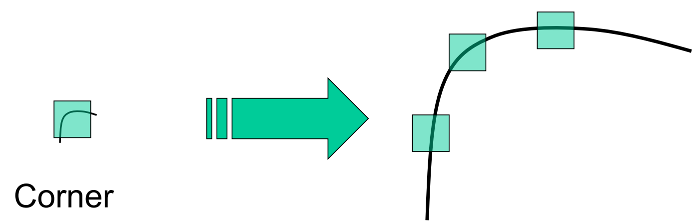
Feature detection with scale selection

- We want to extract features with characteristic scale that matches the image transformation such as scaling and translation



Matching regions across scales

Scaling

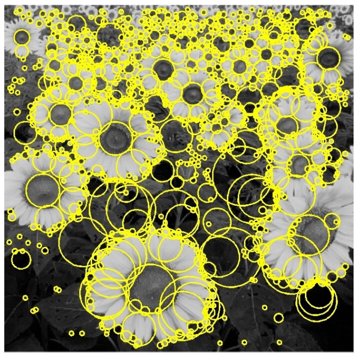


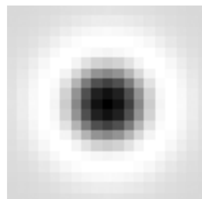
All points will be classified as edges

Corner detection is sensitive to the image scale

Blob detection: basic idea

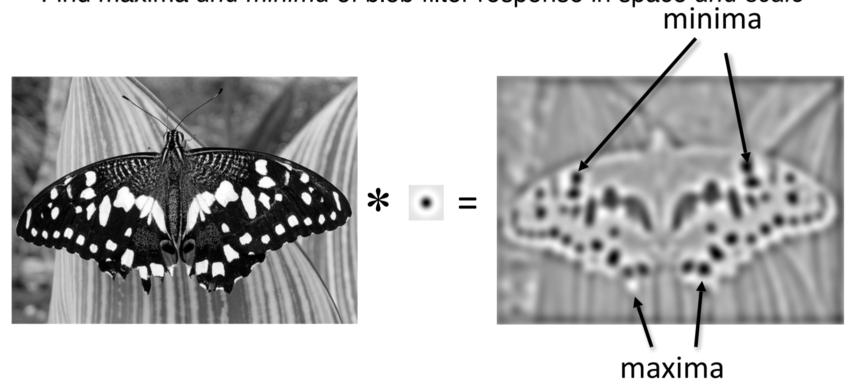
- Convolve the image with a “blob filter” at multiple scales
- Look for extrema (maxima or minima) of filter response in the resulting *scale-space*
- This will give us a scale and location of the detected blob





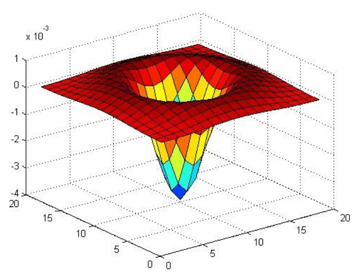
Blob detection: basic idea

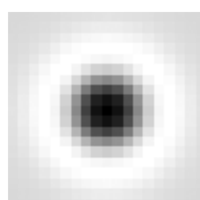
Find maxima *and minima* of blob filter response in space *and scale*



Blob filter

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



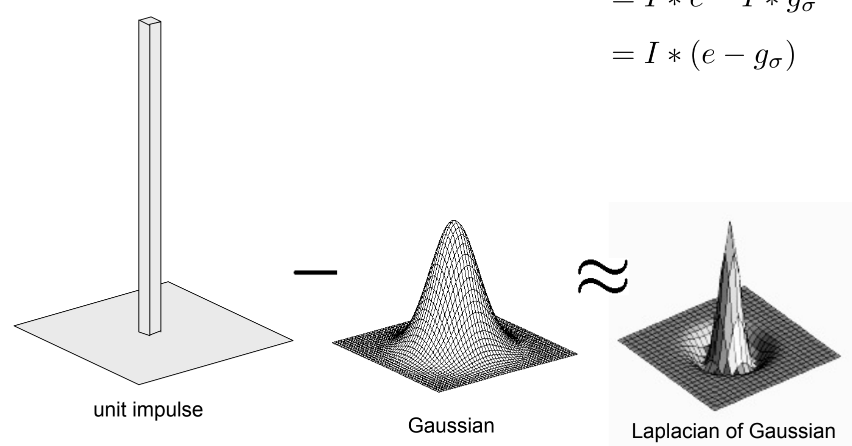


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Recall: sharpening filter

$$I = blurry(I) + sharp(I)$$

$$\begin{aligned} sharp(I) &= I - blurry(I) \\ &= I * e - I * g_\sigma \\ &= I * (e - g_\sigma) \end{aligned}$$

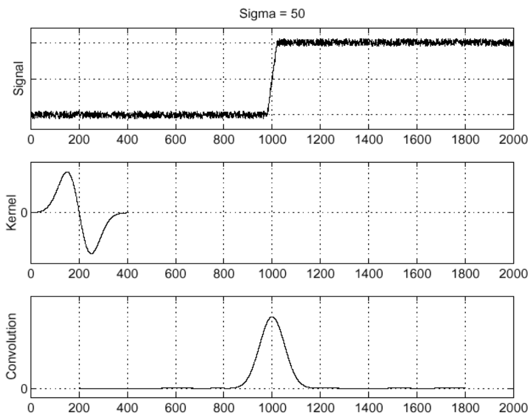


Recall: edge detection

f

g

f*g



signal

edge filter, e.g. Derivative of a Gaussian

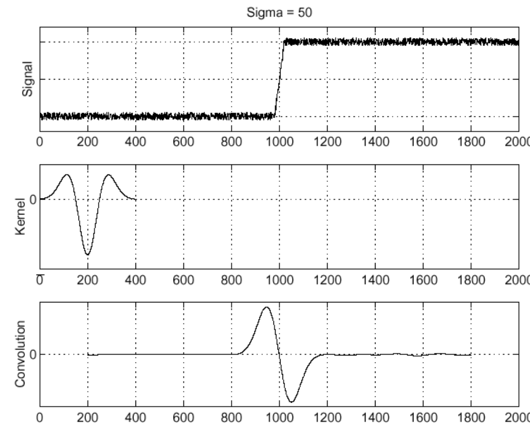
edge response

Edge detection using a Laplacian

f

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

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



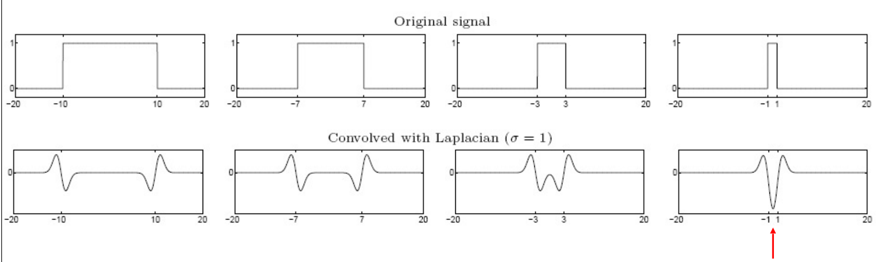
Edge

Second derivative of Gaussian (Laplacian)

Edge = zero crossing of second derivative

From edges to blobs

- edge = ripple
- blob = superposition of two ripples

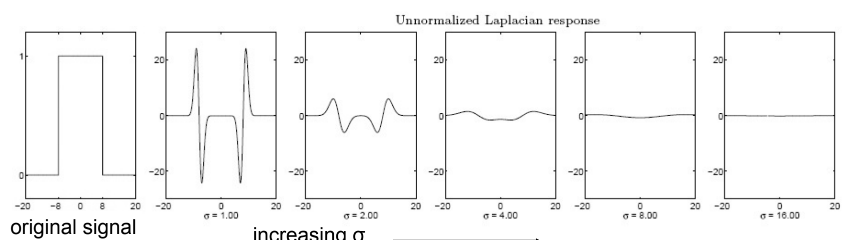


maximum

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

Scale selection

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

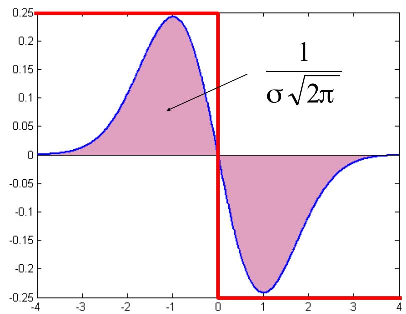


original signal (radius=8)

increasing σ

Scale normalization

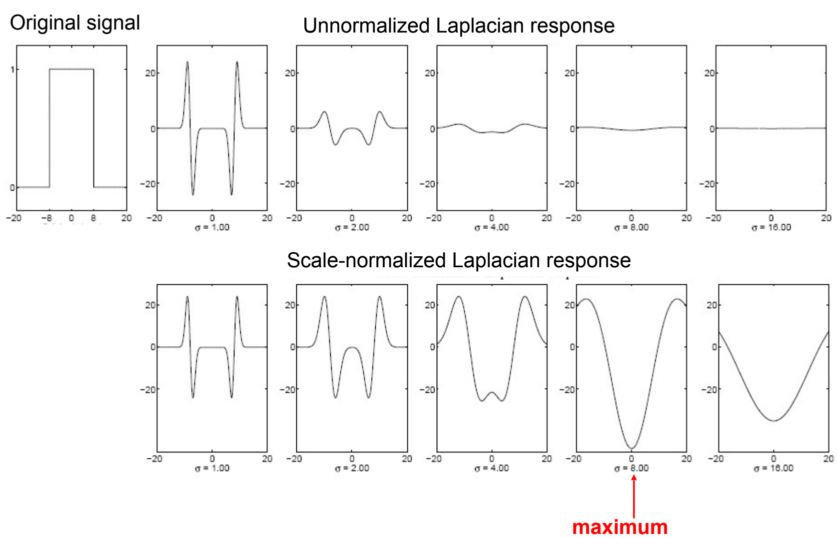
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases



Scale normalization

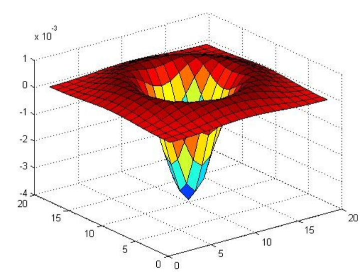
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases
- To keep response the same (scale-invariant), must multiply Gaussian derivative by σ
- Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

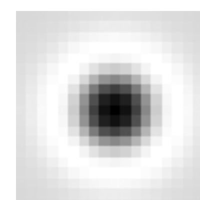
Effect of scale normalization



Blob detection in 2D

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

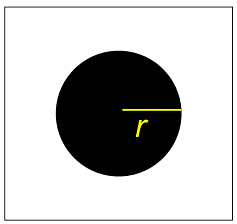




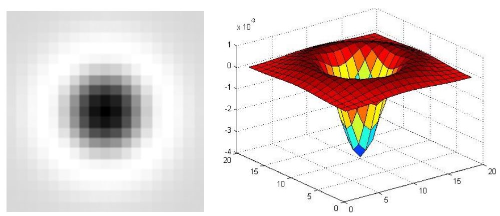
Scale-normalized: $\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$

Scale selection

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



image



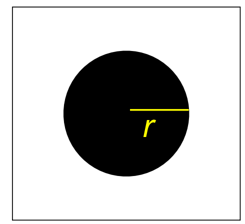
Laplacian

Scale selection

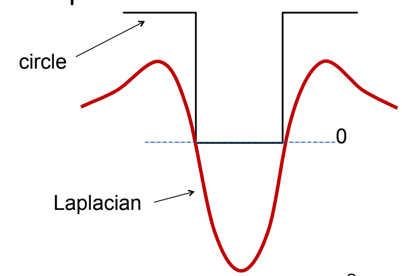
- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- To get maximum response, the zeros of the Laplacian have to be aligned with the circle
- The Laplacian is given by (up to scale):

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

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

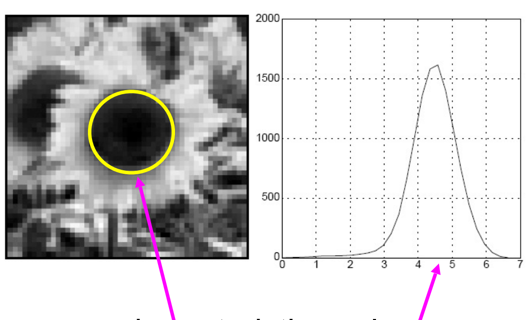


image



Characteristic scale

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



characteristic scale

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

Scale-space blob detector

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

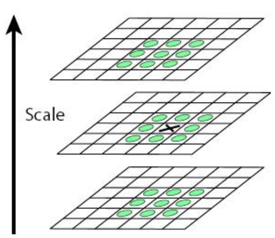
Scale-space blob detector: Example

Scale-space blob detector: Example

sigma = 11.9912

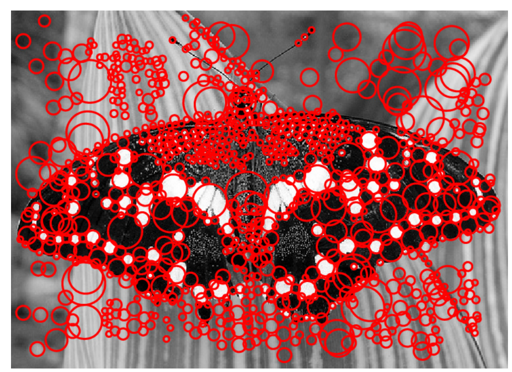
Scale-space blob detector

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



Source: L. Lazebnik

Scale-space blob detector: Example



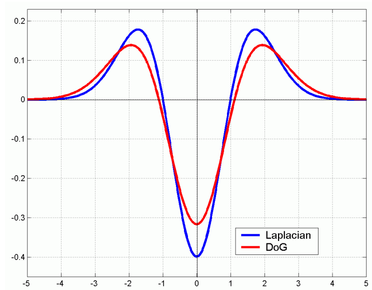
Source: L. Lazebnik

Efficient implementation

- Approximating the Laplacian with a difference of Gaussians:

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

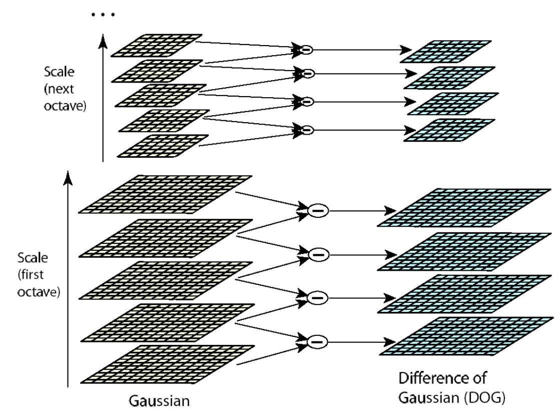
$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$
(Difference of Gaussians)



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

Is the Laplacian separable?

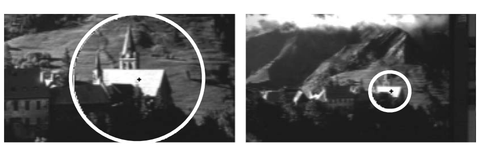
Efficient implementation

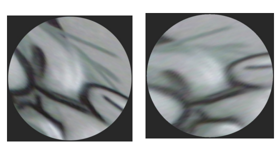


David G. Lowe. **"Distinctive image features from scale-invariant keypoints."** *IJCV* 60 (2), pp. 91-110, 2004.

From feature detection to description

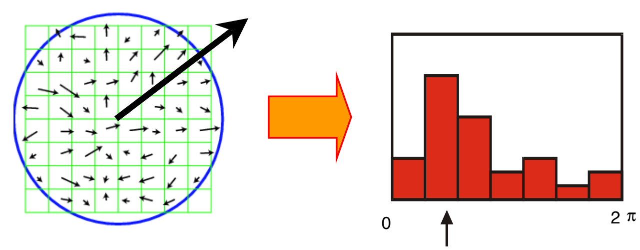
- Scaled and rotated versions of the same neighborhood will give rise to blobs that are related by the same transformation
- What to do if we want to compare the appearance of these image regions?
  - **Normalization**: transform these regions into same-size circles
  - Problem: rotational ambiguity





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

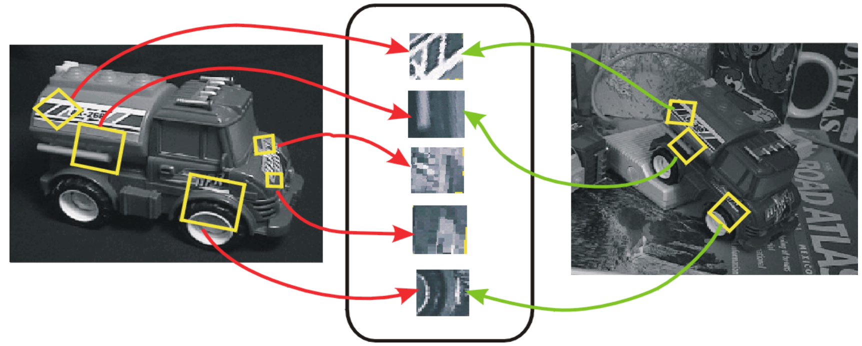


SIFT features

- Detected features with characteristic scales and orientations:

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

From feature detection to description



how should we represent the patches?