Computer Vision

Features

Jan van Gemert (J.C.vanGemert@uva.nl)

Motivation: Build a Panorama

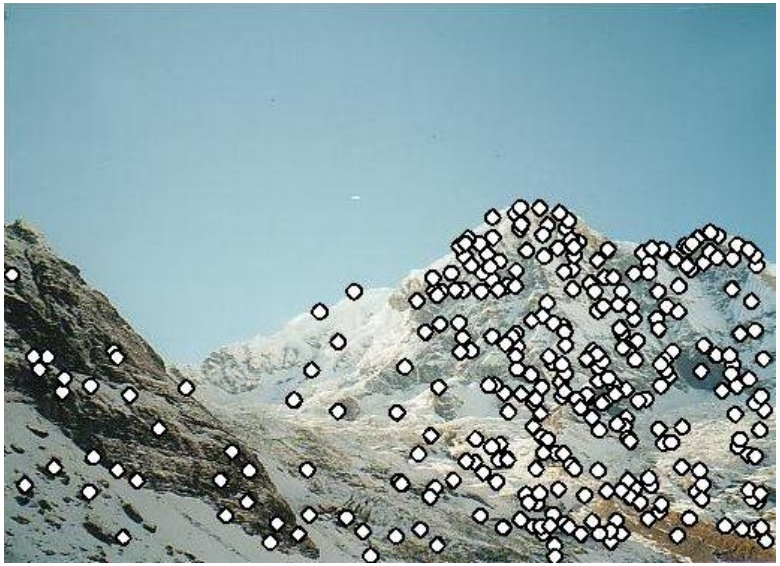
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

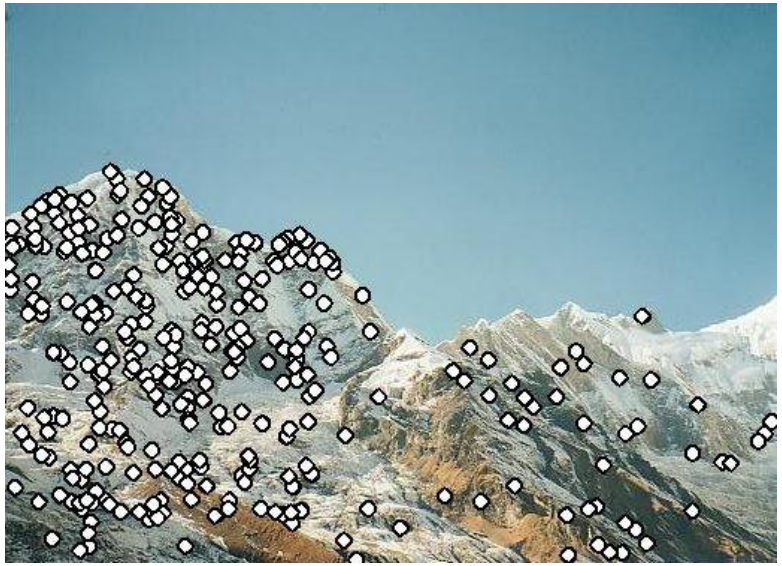
How do we build panorama?

- We need to match (align) images

Matching with Features

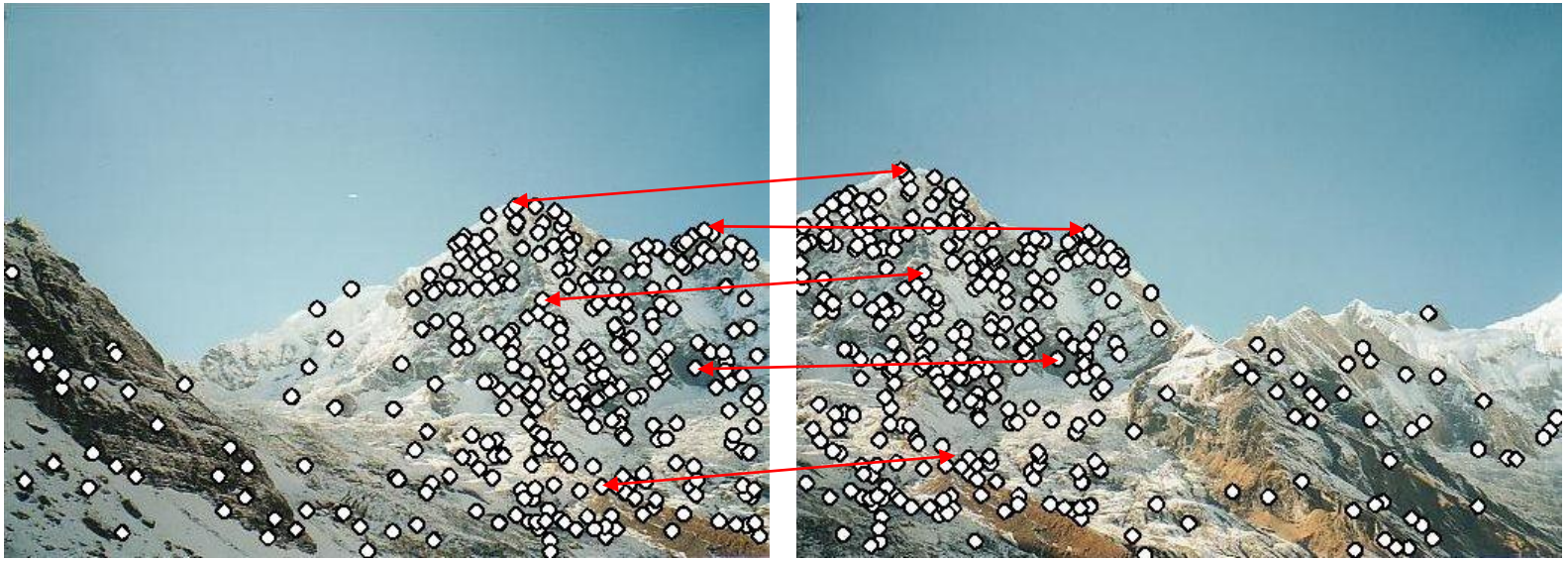
- Detect feature points in both images





Matching with Features

- Detect feature points in both images
- Find corresponding pairs



Matching with Features

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

Matching with Features

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

no chance to match!

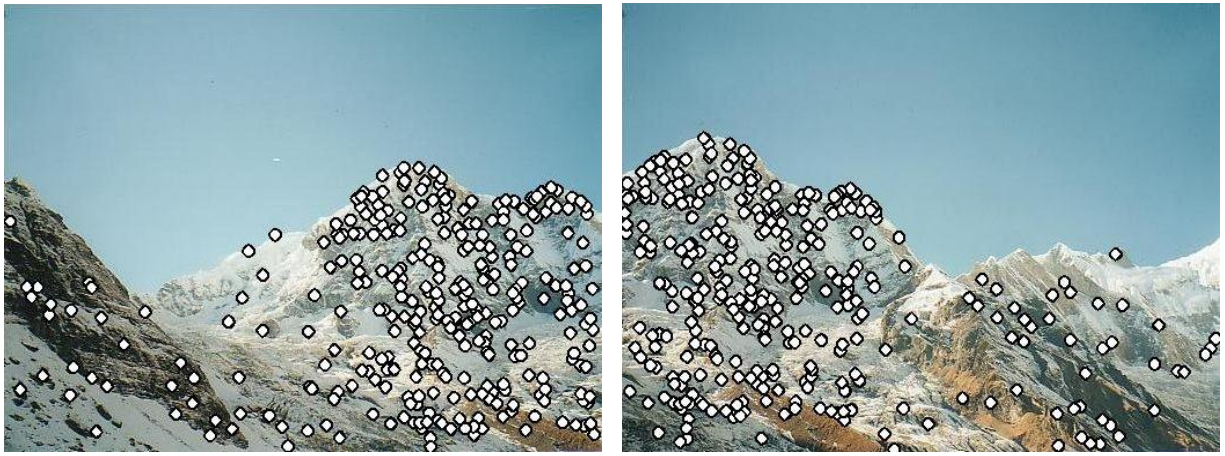
We need a repeatable detector

Matching with Features

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

We need a reliable and distinctive descriptor

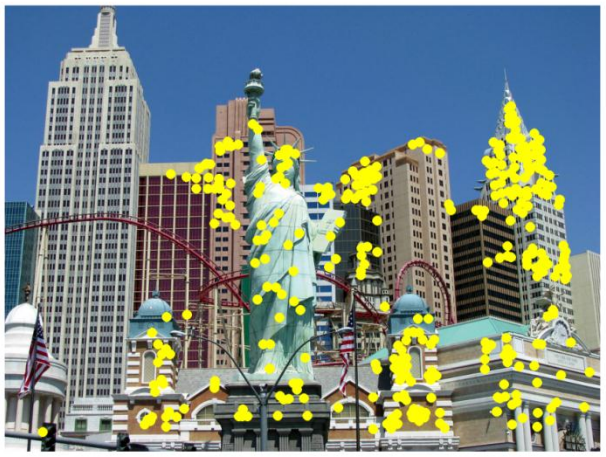
What are good detected features?



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

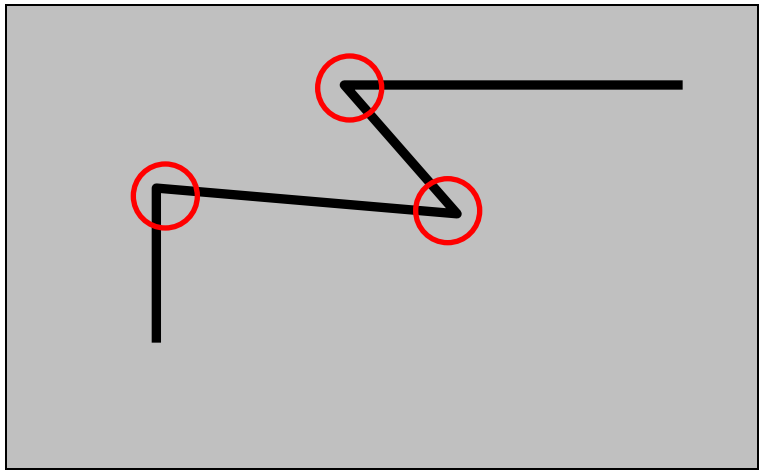
Applications

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



A feature detector example:

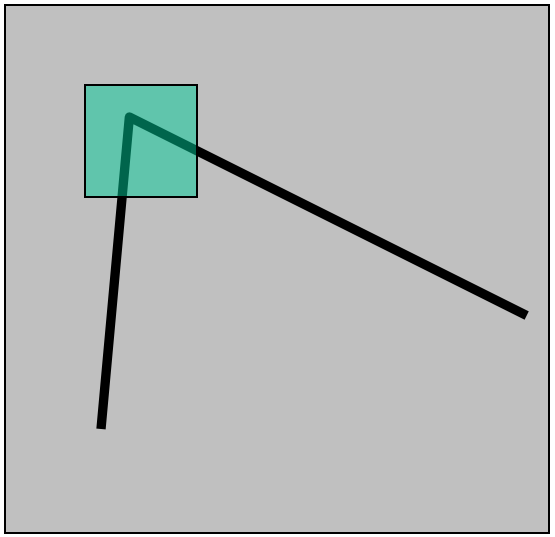
Harris corner detector



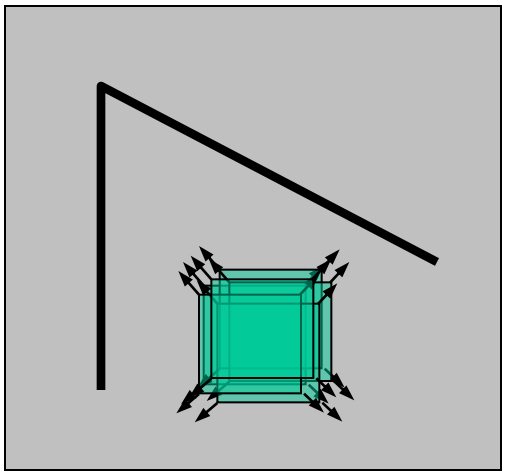
C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988

The Basic Idea

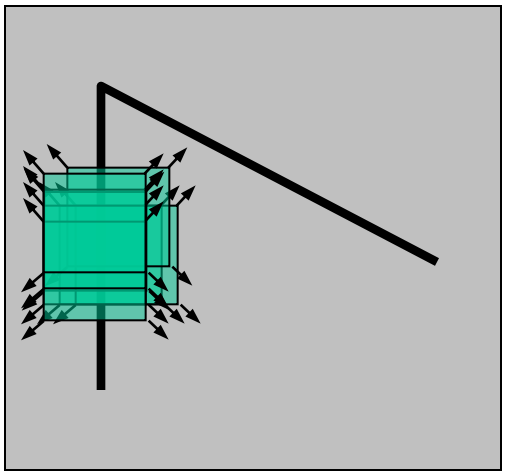
- We should easily localize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



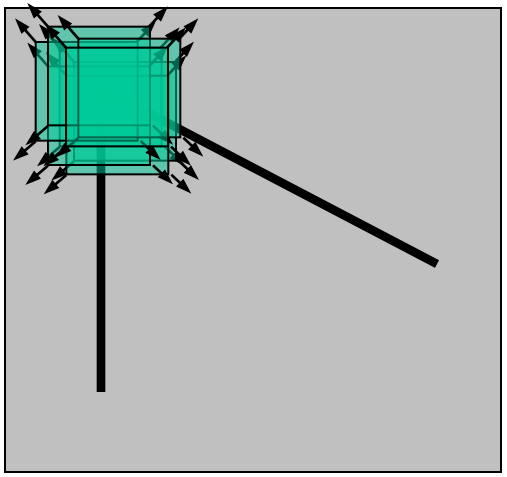
Harris Detector: Basic Idea



"flat" region:
no change as shift window in all directions



"edge":
no change as shift window along the edge direction



"corner":
significant change as shift window in all directions

Harris Detector: Mathematics

Window-averaged change of intensity induced by shifting the image data by [*u,v*]:

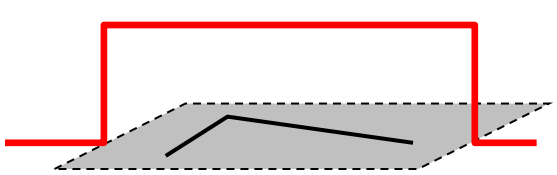
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

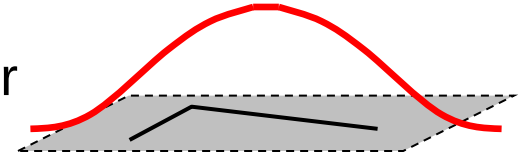
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Taylor series approx to shifted image

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

$$E(u,v) \approx \sum_{x,y} w(x,y)[I(x,y) + uI_x + vI_y - I(x,y)]^2$$

$$= \sum_{x,y} w(x,y)[uI_x + vI_y]^2$$

$$= \sum_{x,y} w(x,y)\begin{pmatrix} u & v \end{pmatrix}\begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}\begin{pmatrix} u \\ v \end{pmatrix}$$

Harris Detector: Mathematics

Expanding I(x,y) in a Taylor series expansion, we have, for small shifts $[u,v]$, a *bilinear* approximation:

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a 2×2 matrix computed from image derivatives:

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

M is also called “structure tensor”
or “second moment matrix”

Harris Detector: Mathematics

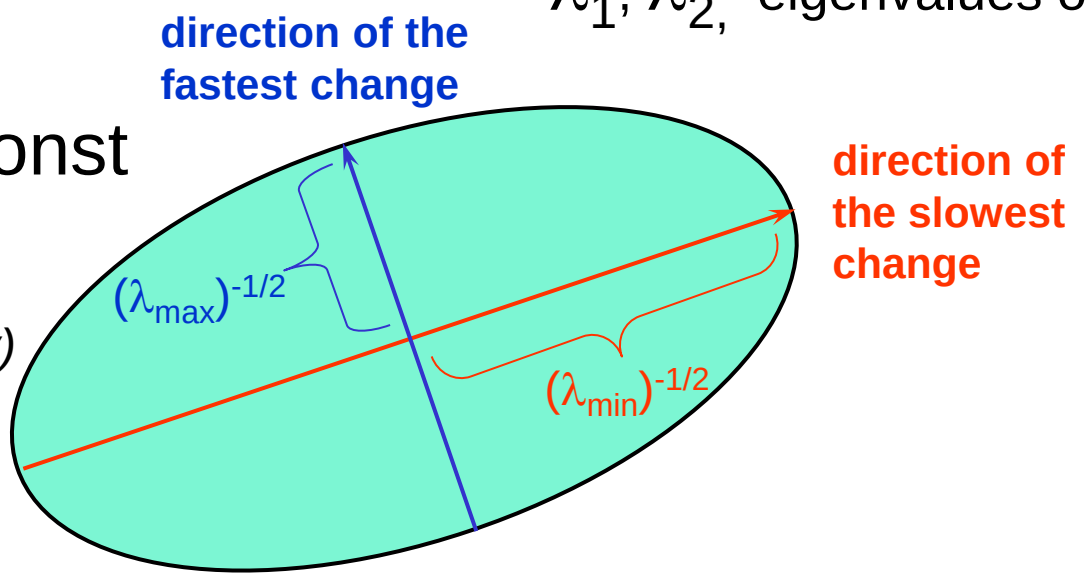
Intensity change in shifting window:

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

Where is the largest change?

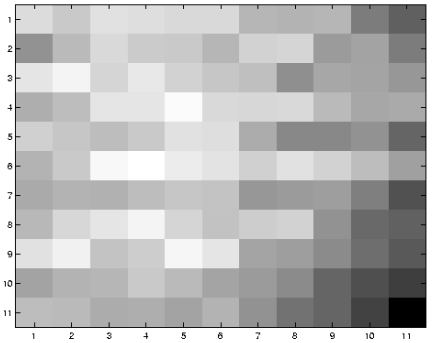
$\lambda_1, \lambda_2,$ eigenvalues of M

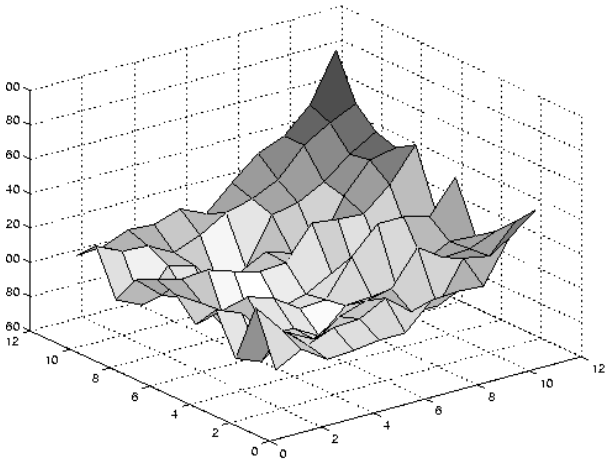
Ellipse $E(u,v)$ = const



Iso-intensity contour of $E(u,v)$

Selecting Good Features

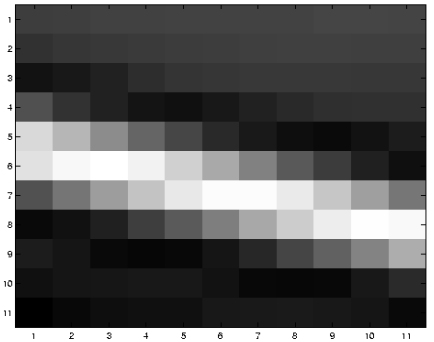


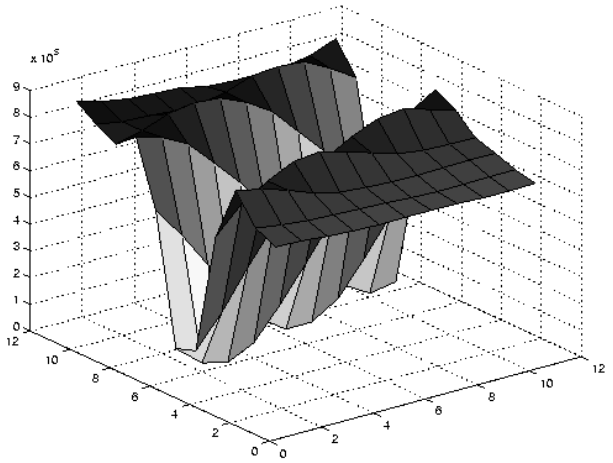


How large is each of the two eigenvalues?

small λ_1, small λ_2

Selecting Good Features

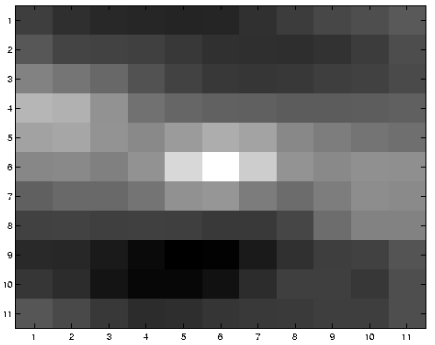


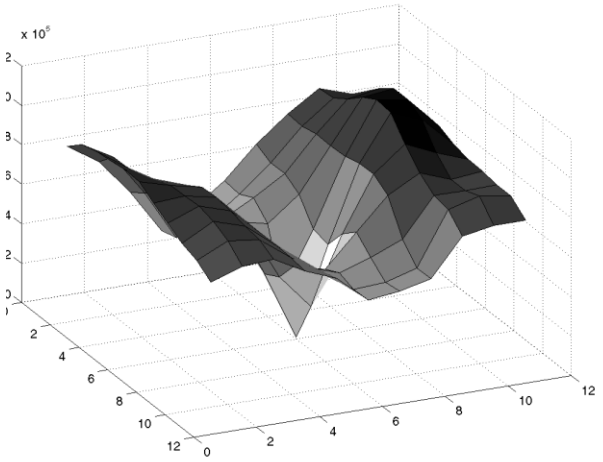


How large is each of the two eigenvalues?

large λ_1, small λ_2

Selecting Good Features



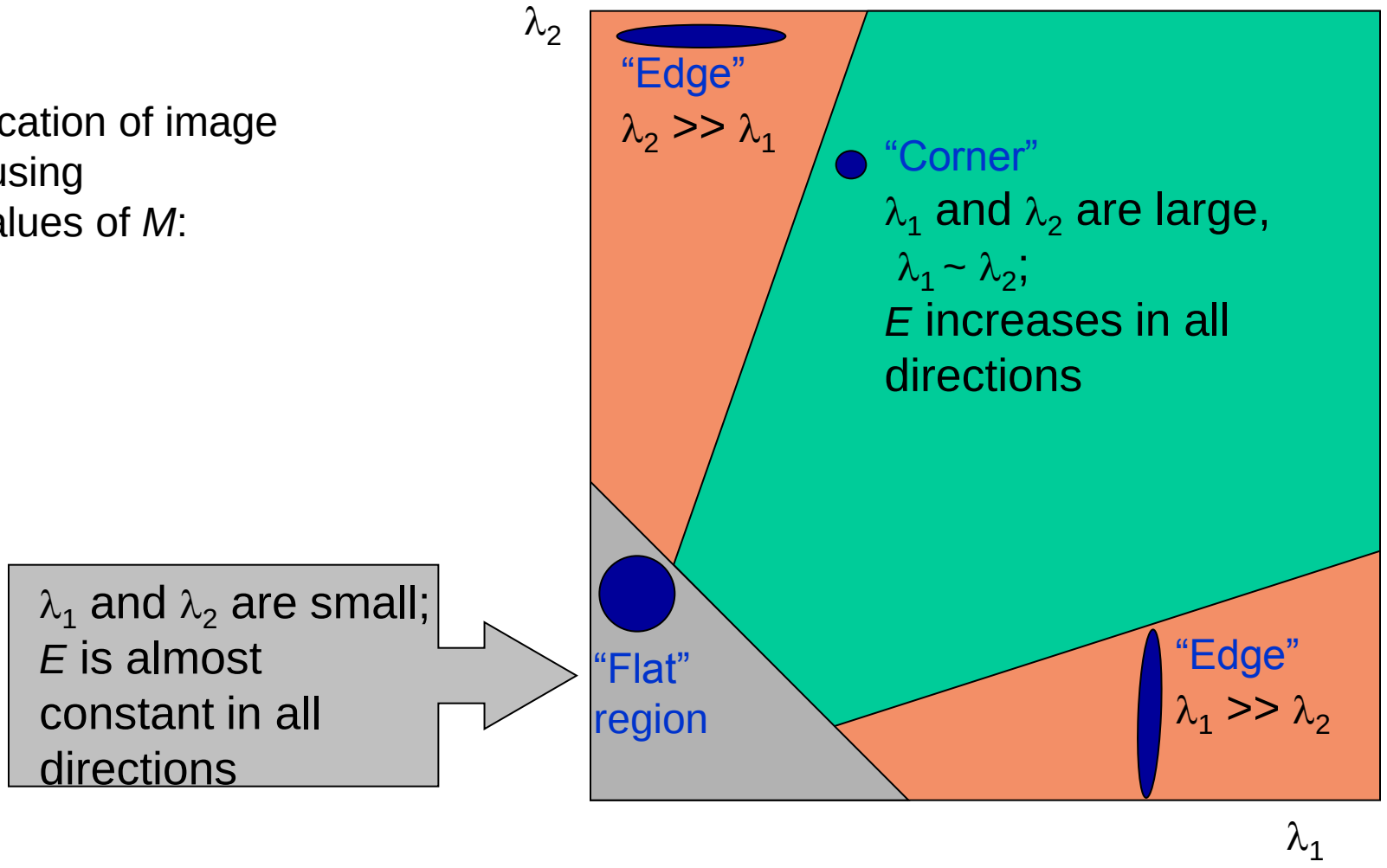


How large is each of the two eigenvalues?

λ_1 and λ_2 are large

Harris Detector: Mathematics

Classification of image points using eigenvalues of *M*:

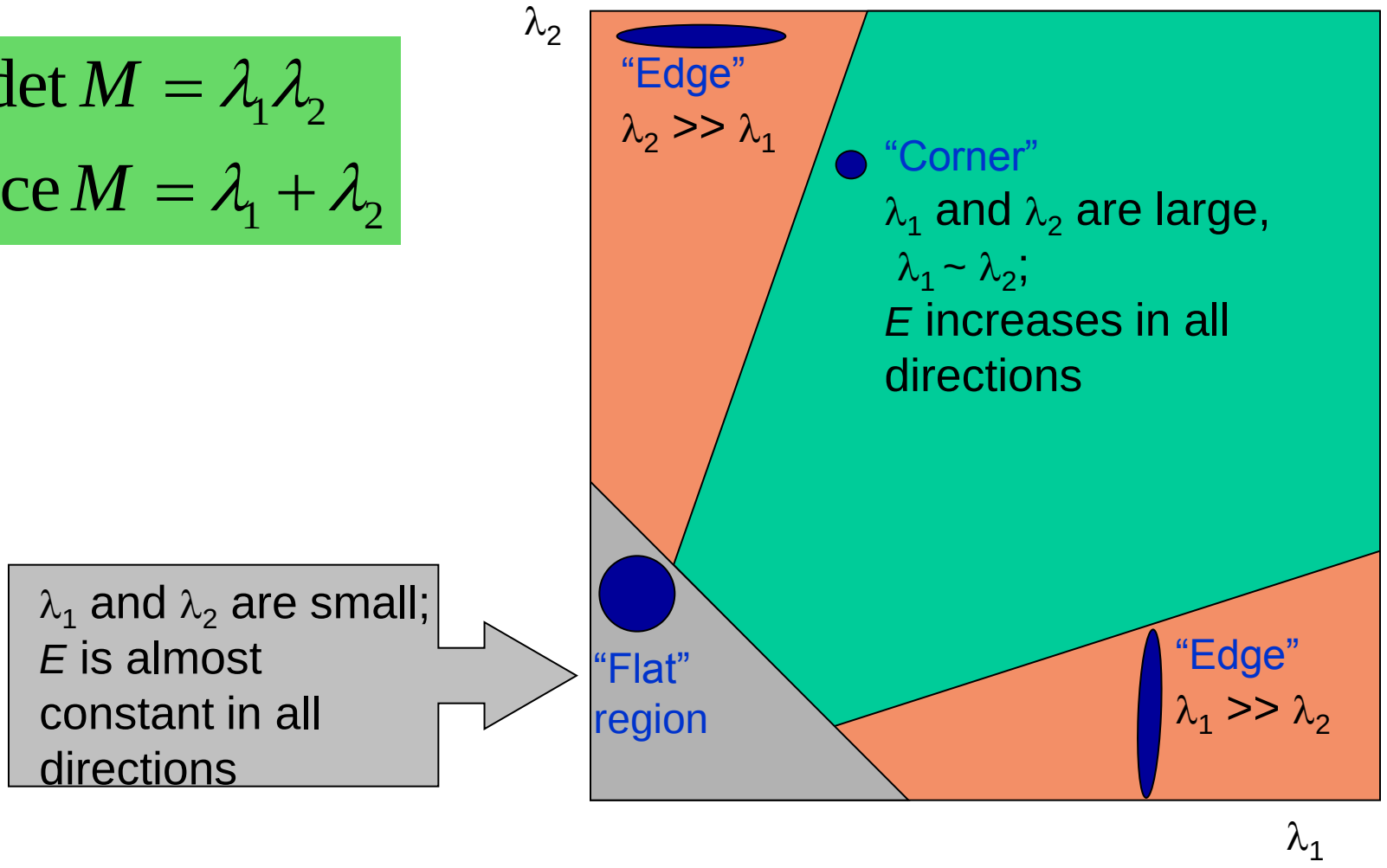


Harris Detector: Mathematics

No need to compute eigenvalues:

$$\det M = \lambda_1 \lambda_2$$

$$\text{trace}\, M = \lambda_1 + \lambda_2$$



Harris Detector: Mathematics

No need to compute eigenvalues:

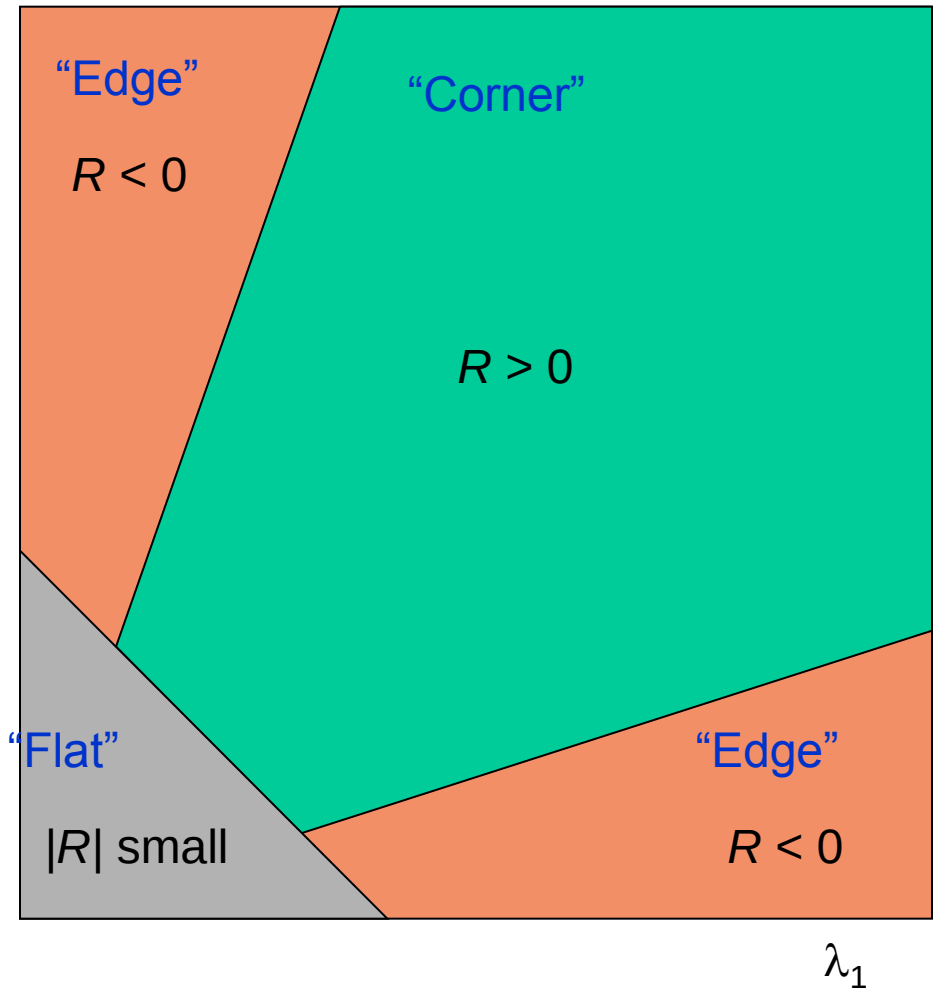
$$\det M = \lambda_1 \lambda_2$$
$$\text{trace}\, M = \lambda_1 + \lambda_2$$

Measure of corner response:

$$R = \det M - k\left(\text{trace}\, M\right)^2$$

(k – empirical constant, k = 0.04-0.06)

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



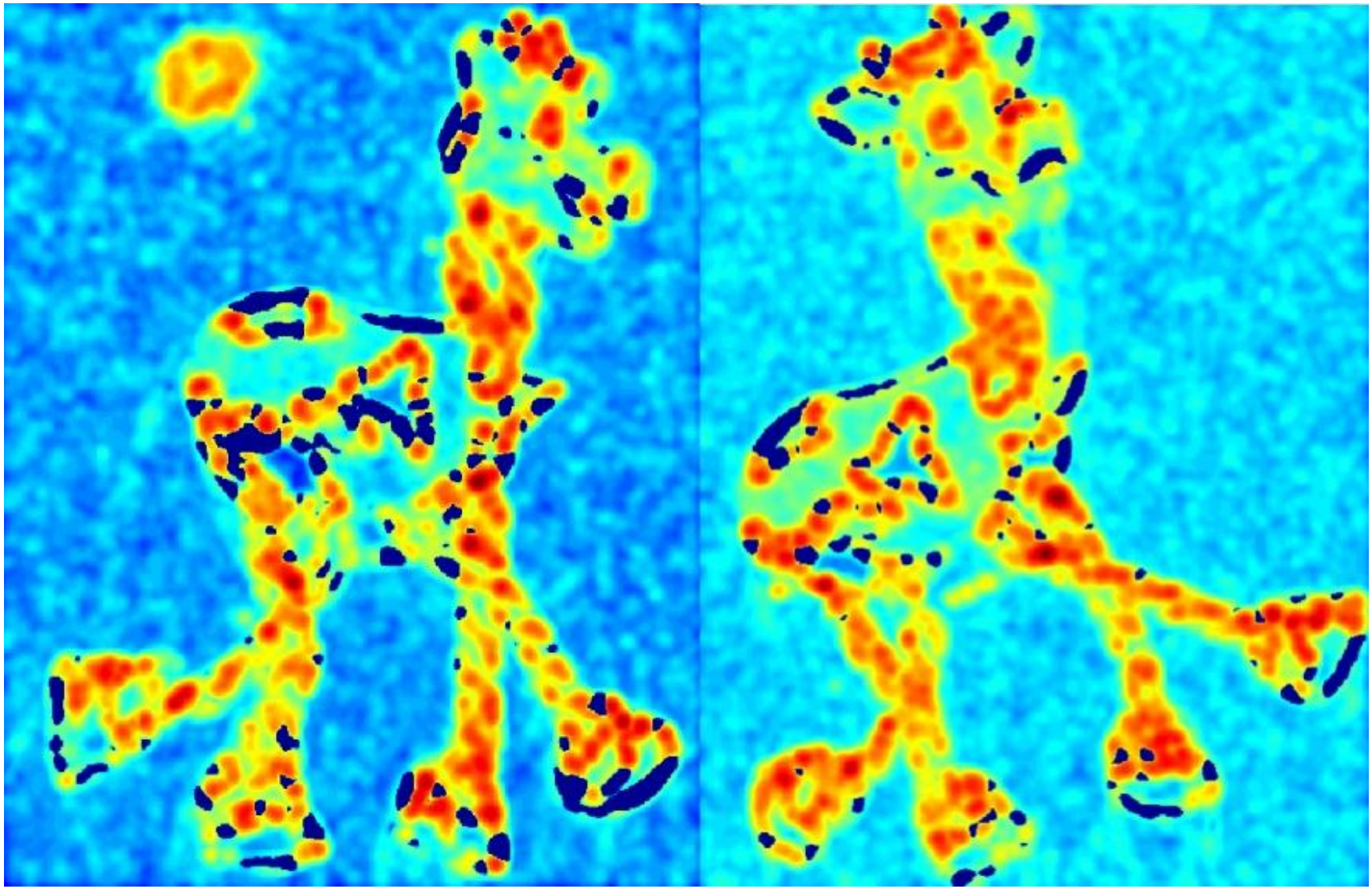
Harris Detector

- The Algorithm:
  - Compute each element of the structure tensor on the image level
  - Find points with large corner response function R (R > threshold)
  - Take the points of local maxima of R

Harris Detector: Steps

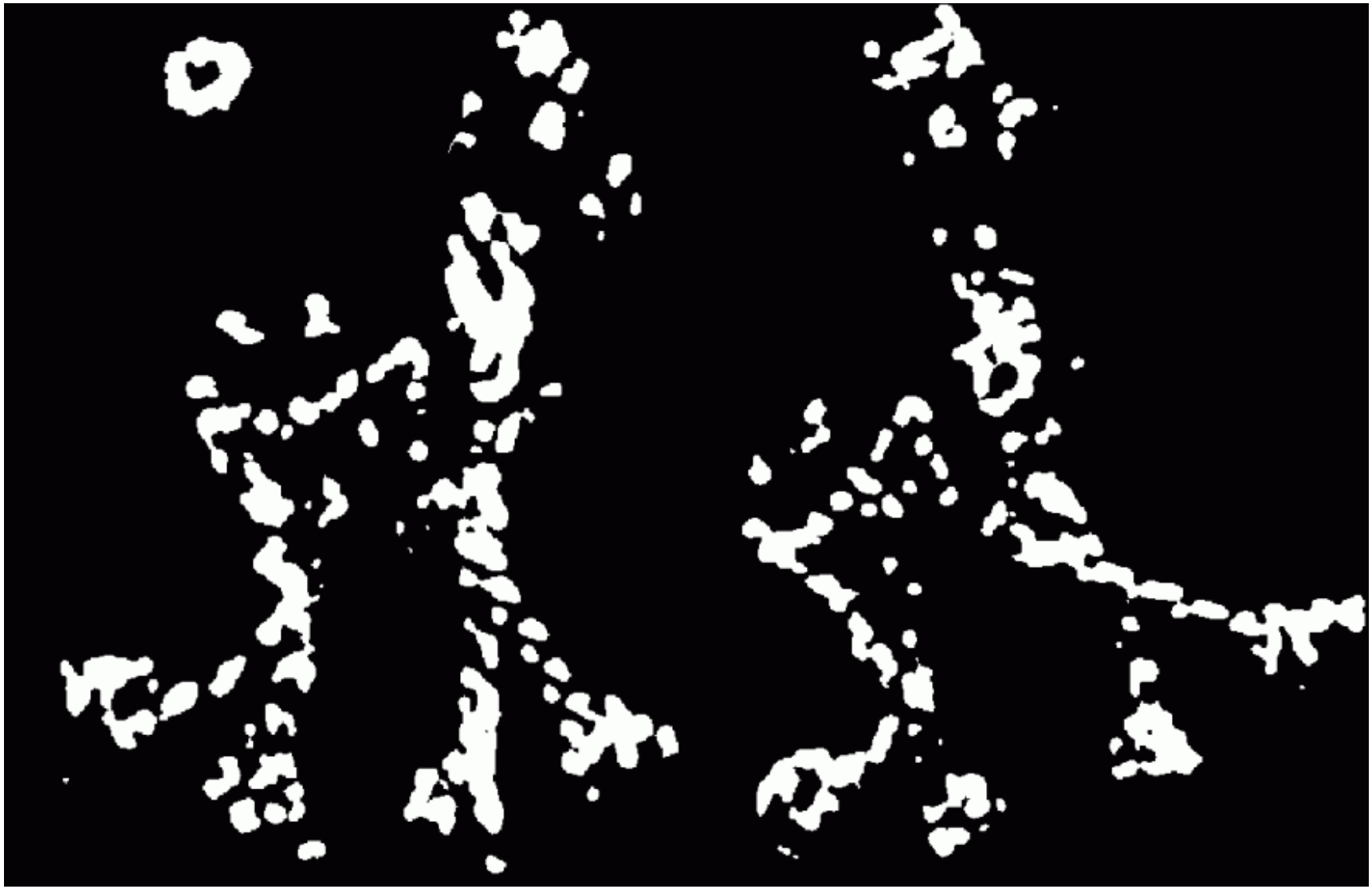
Harris Detector: Steps

Compute corner response *R*



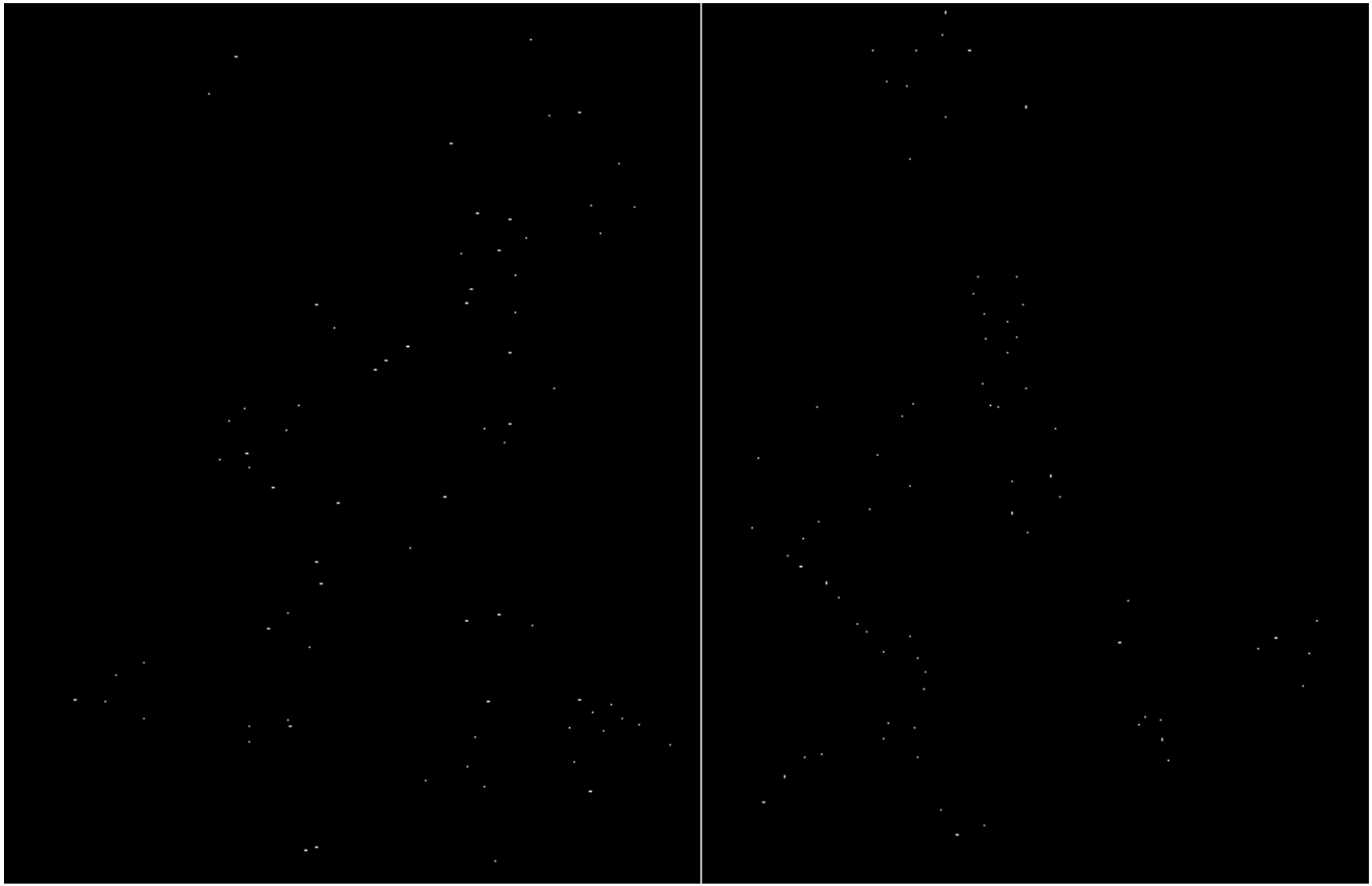
Harris Detector: Steps

Find points with large corner response: R>threshold



Harris Detector: Steps

Take only the points of local maxima of R



Harris Detector: Steps

Invariance and covariance

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

Harris Detector: Summary

- Average intensity change in direction $[u,v]$ can be expressed as a bilinear form:

$$E(u,v) \cong \begin{bmatrix} u,v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

- Describe a point in terms of eigenvalues of M:
  measure of corner response

$$R = \lambda_1 \lambda_2 - k\left(\lambda_1 + \lambda_2\right)^2$$

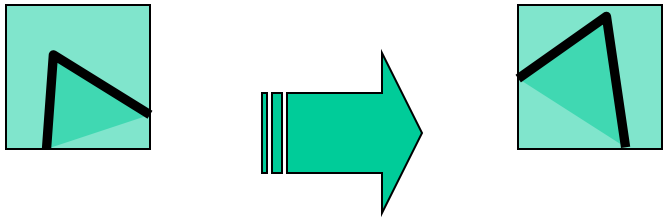
- A good (corner) point should have a *large intensity change* in *all directions*, i.e. R should be large positive

Ideal feature detector

- Would always find the same point on an object, regardless of changes to the image.
- Ie, insensitive to changes in:
  - Scale
  - Lighting
  - Perspective imaging
  - Partial occlusion

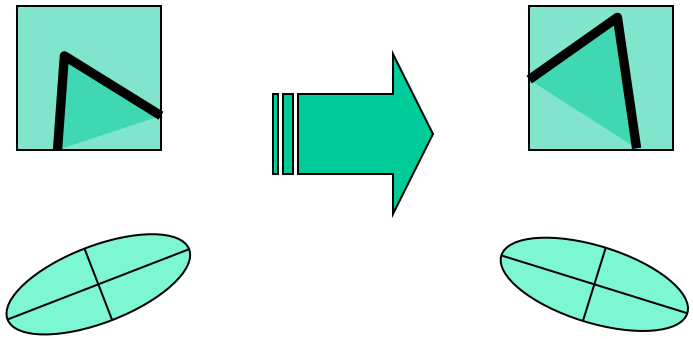
Harris Detector: Some Properties

- Rotation invariance?



Harris Detector: Some Properties

- Rotation invariance?



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

Corner response R is invariant to image rotation

Harris Detector: Some Properties

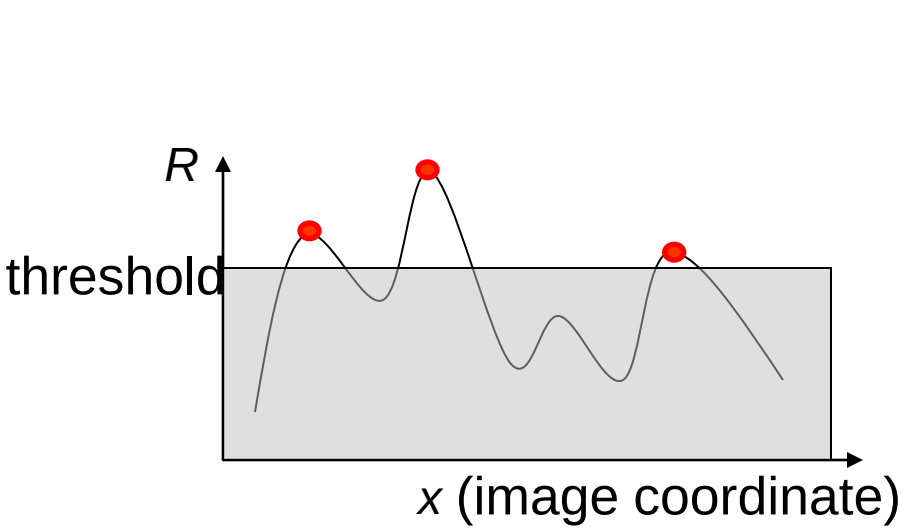
- Invariance to image intensity change?

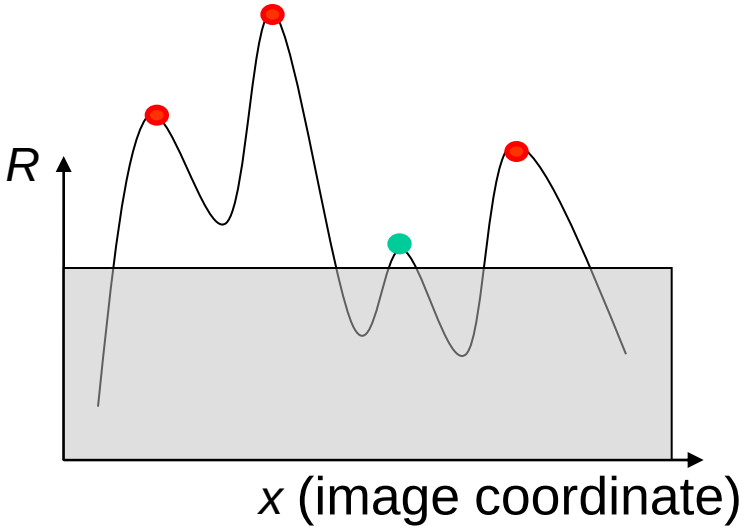
Harris Detector: Some Properties

- Invariance to image intensity change?
- Partial invariance to additive and multiplicative intensity changes

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

  ✓ Intensity scale: $I \rightarrow a\,I$ Because of fixed intensity threshold on local maxima, only partial invariance to multiplicative intensity changes.



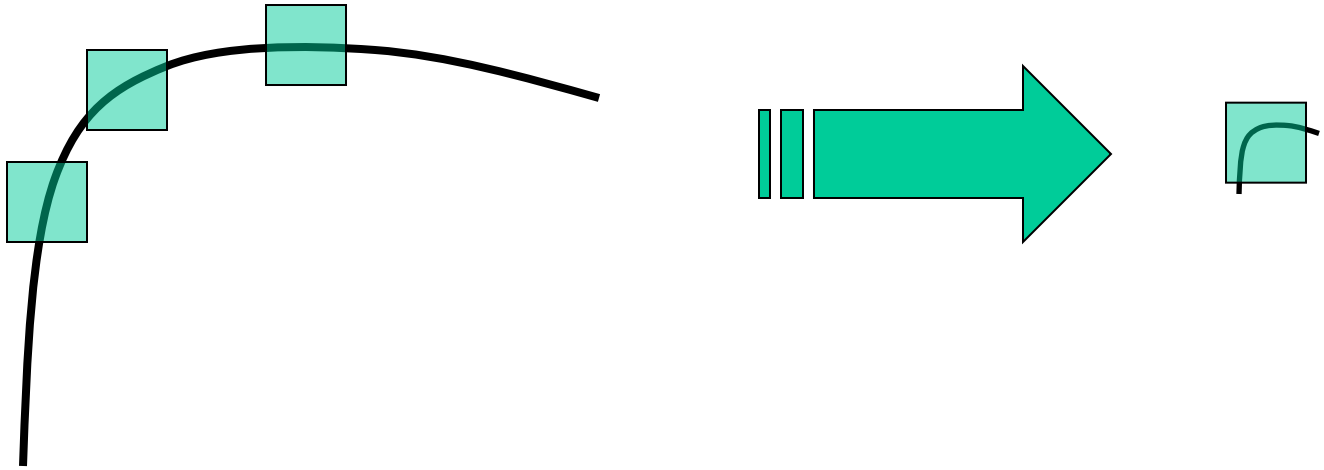


Harris Detector: Some Properties

- Invariant to image scale?

Harris Detector: Some Properties

- Not invariant to *image scale*!

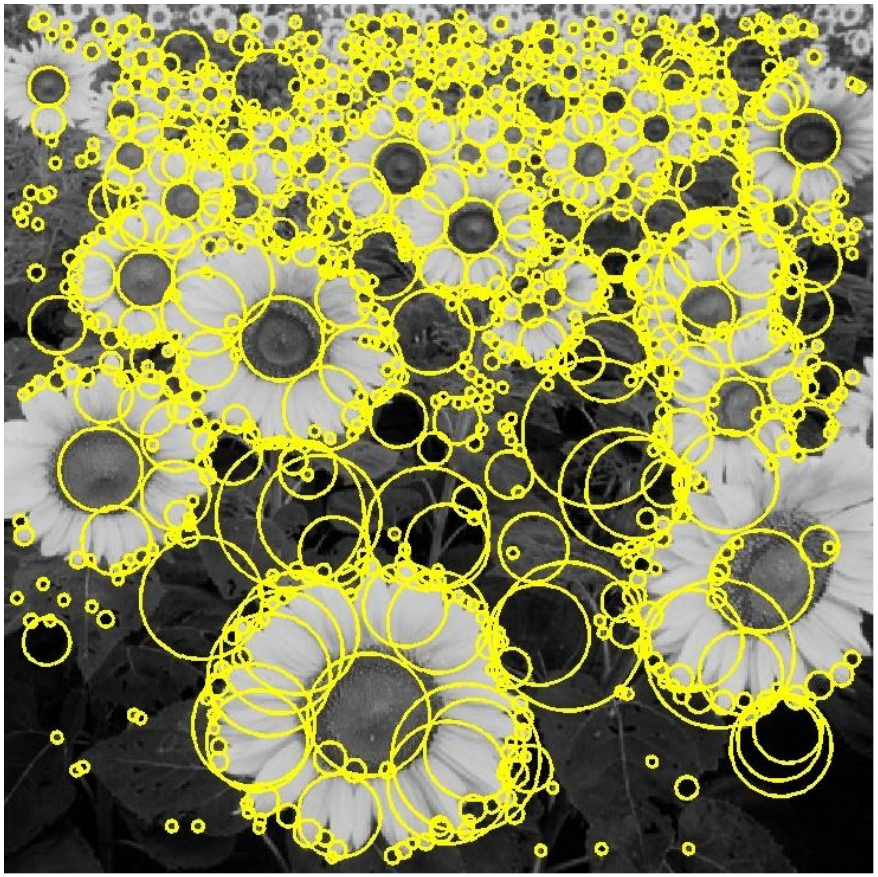


All points will be classified as edges

Corner !

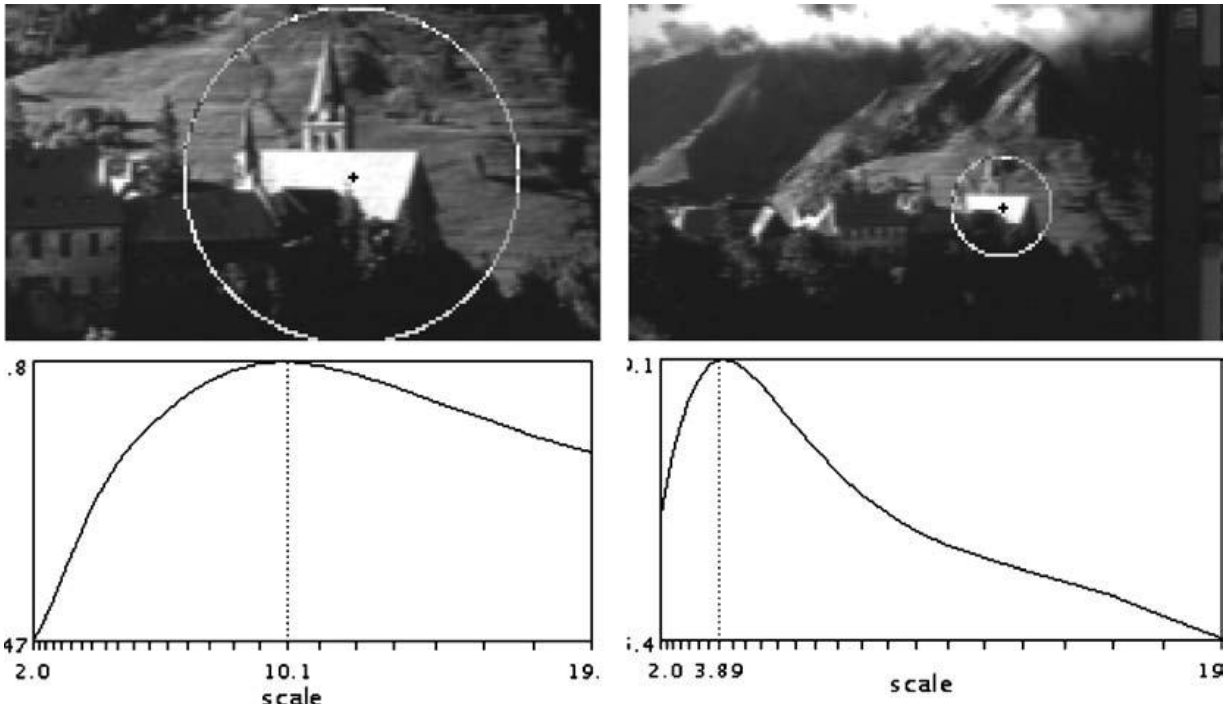
Questions?

Blob detection with scale selection



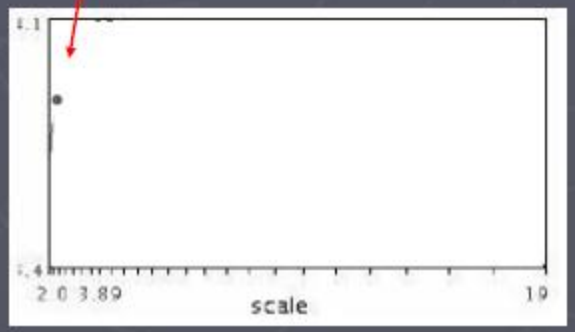
Achieving scale covariance

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



Automatic scale selection

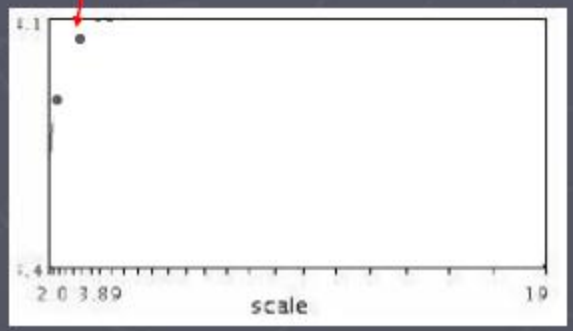
Lindeberg et al., 1996



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

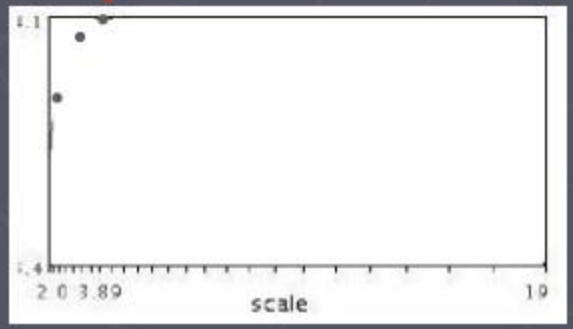
Slide from Tinne Tuytelaars

Automatic scale selection



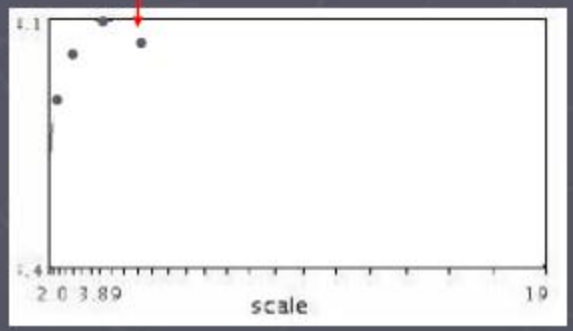
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



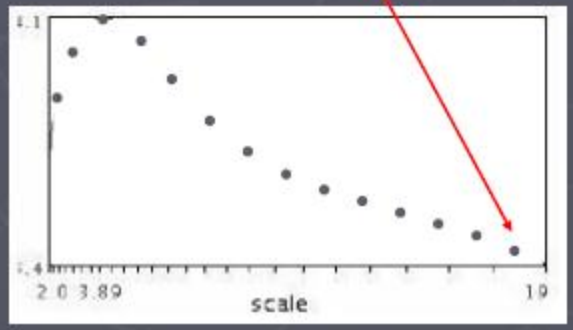
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



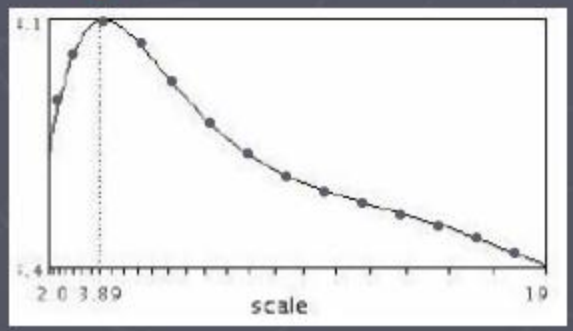
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



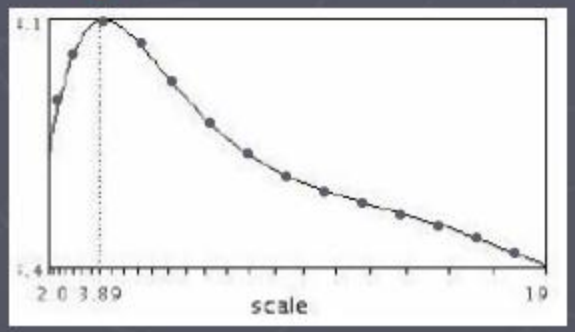
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection

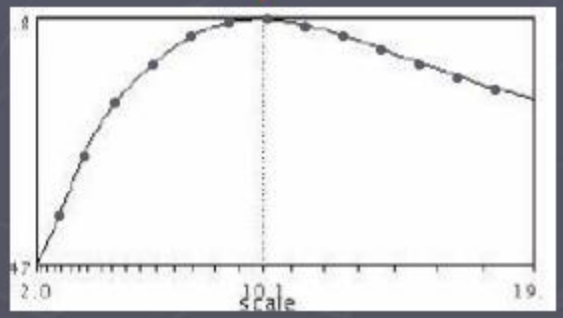


$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

Automatic scale selection



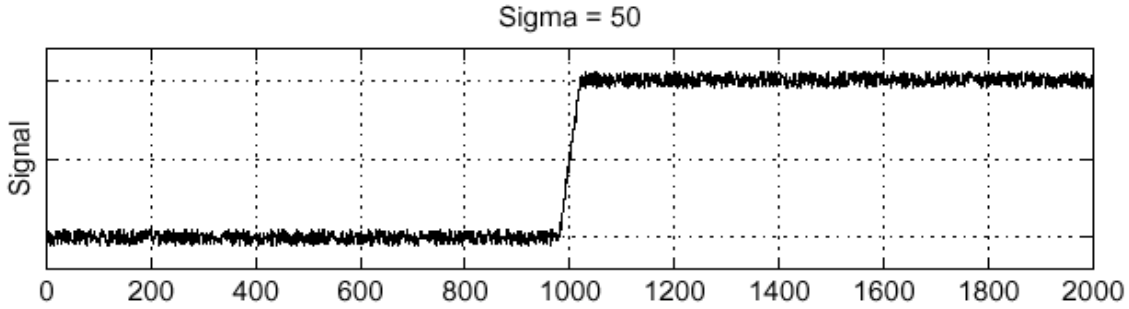
$f(I_{i_1 \ldots i_m}(x,\sigma))$



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

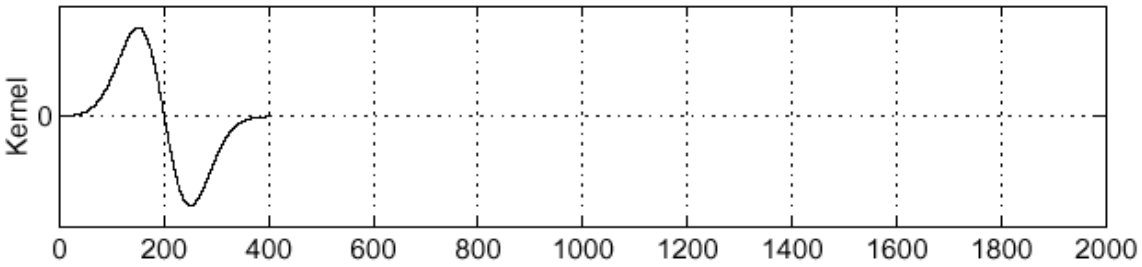
Recall: Edge detection

f



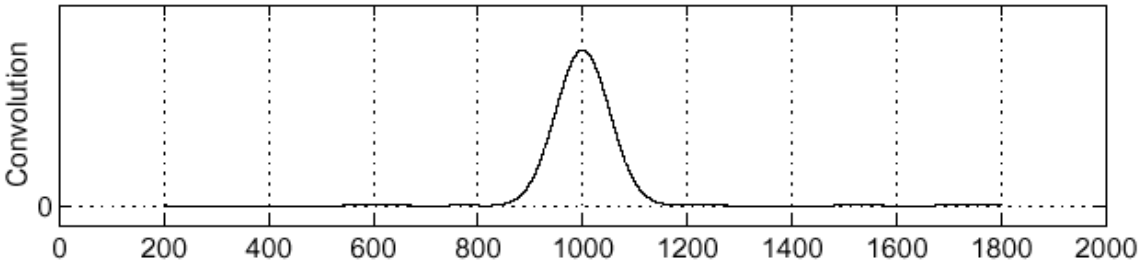
Edge

$\frac{d}{dx} g$



Derivative of Gaussian

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

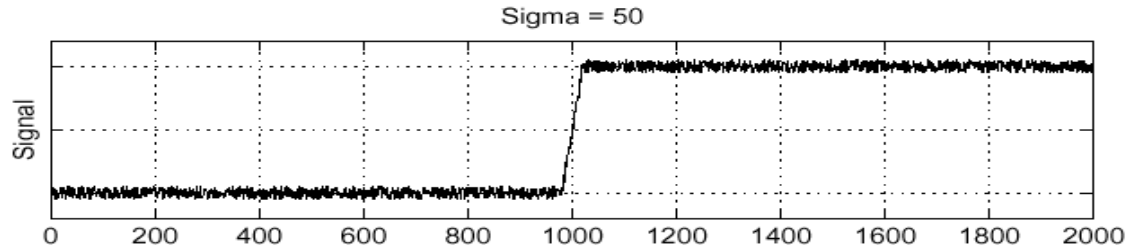


Edge = maximum of derivative

Source: S. Seitz

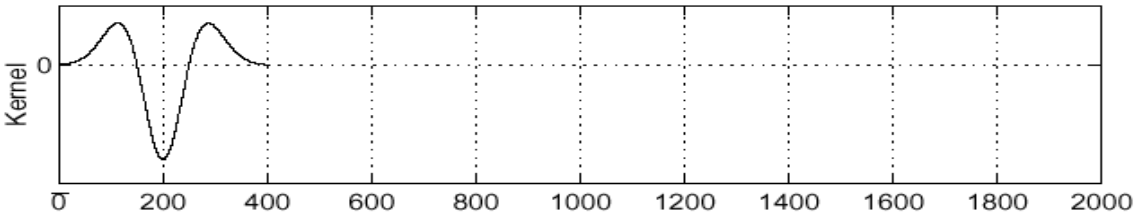
Edge detection, Take 2

f



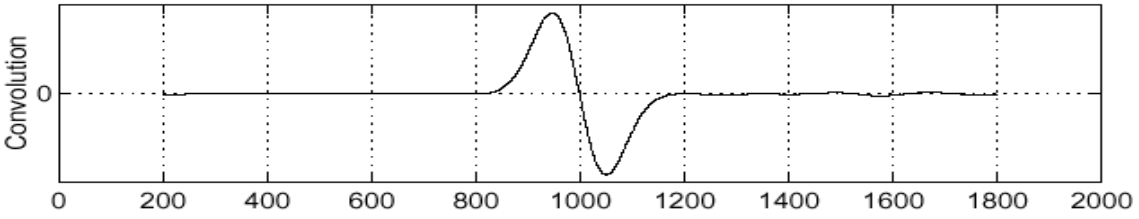
Edge

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



Second derivative of Gaussian (Laplacian)

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

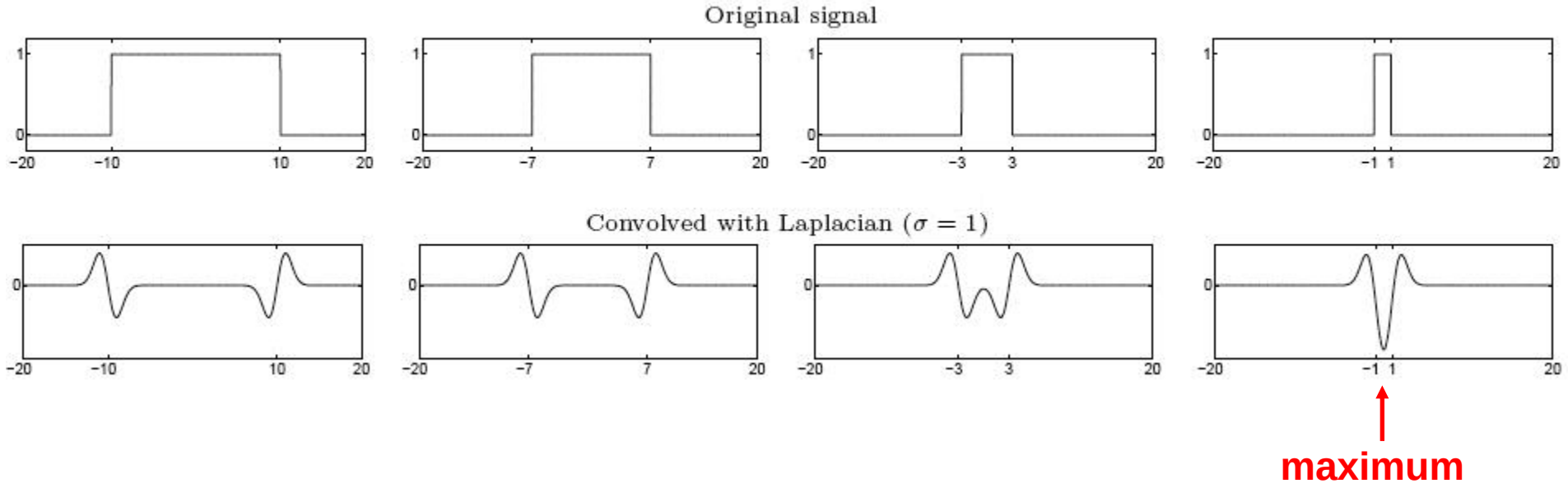


Edge = zero crossing of second derivative

Source: S. Seitz

From edges to blobs

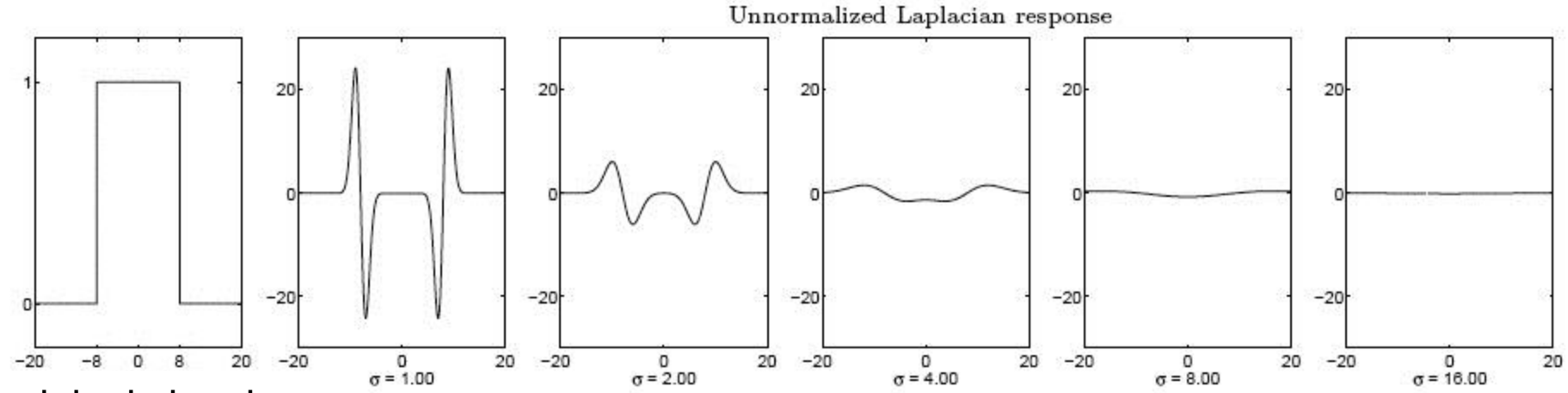
- Edge = ripple
- Blob = superposition of two ripples



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

Scale selection

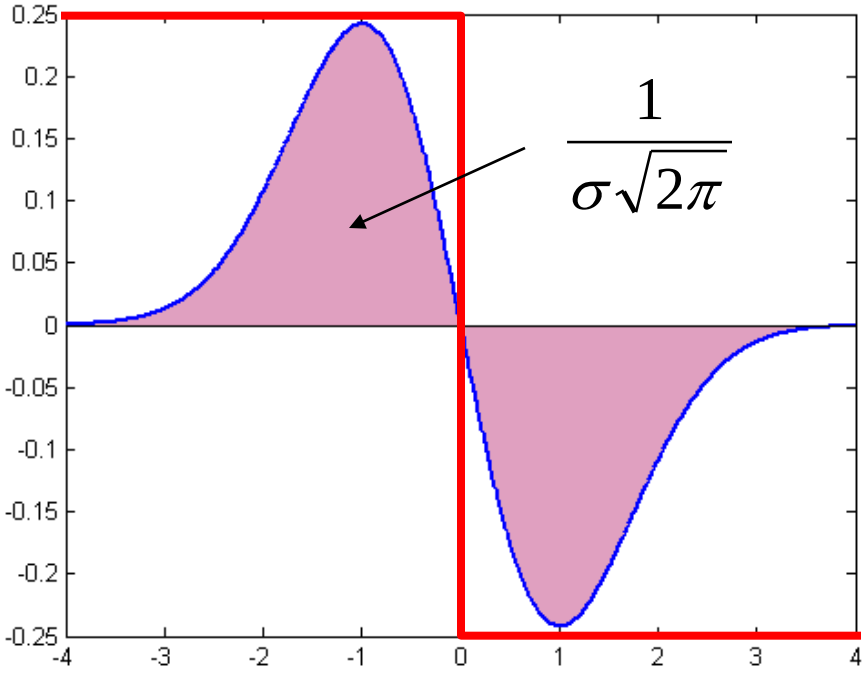
- We want to find the characteristic scale of the blob by convolving it with Laplacians at several scales and looking for the maximum response
- However, Laplacian response decays as scale increases:



Why does this happen?

Scale normalization

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



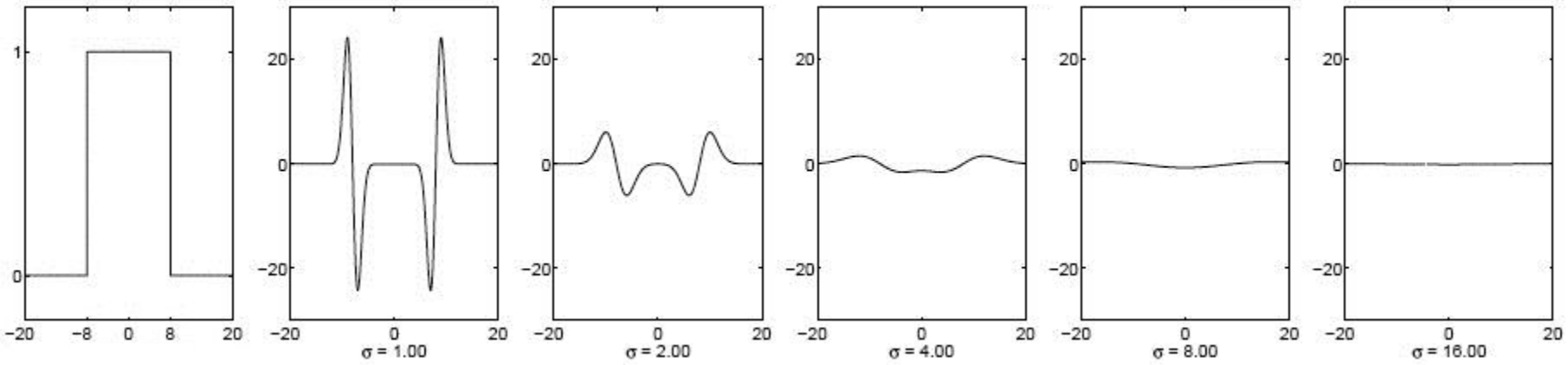
Scale normalization

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

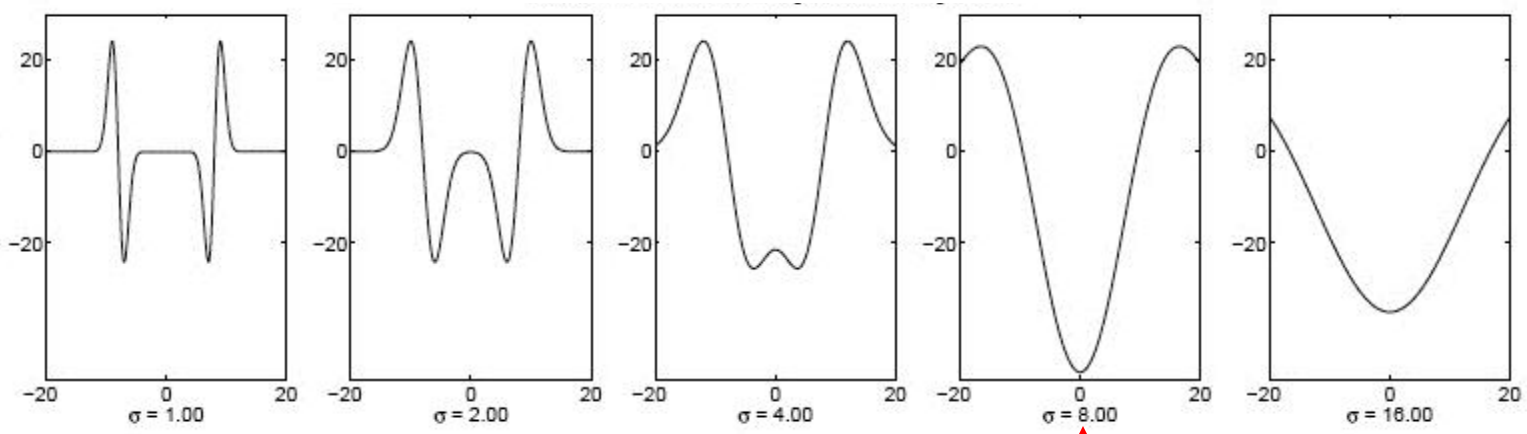
Effect of scale normalization

Original signal

Unnormalized Laplacian response



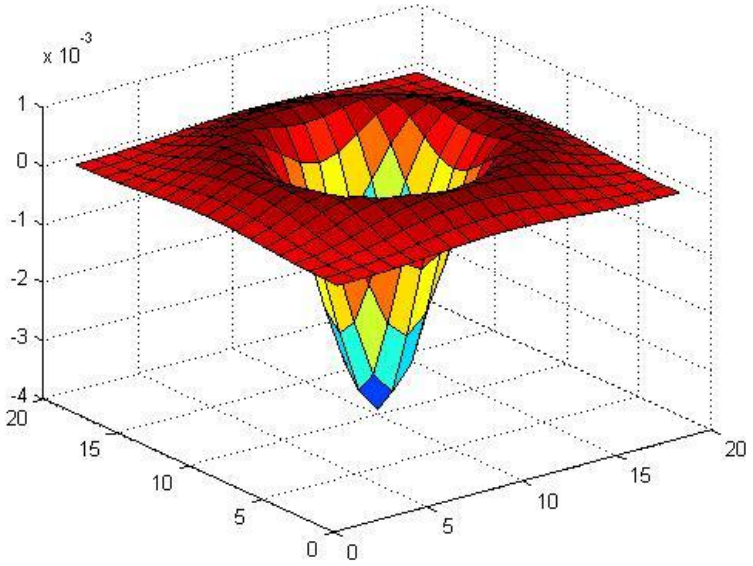
Scale-normalized Laplacian response

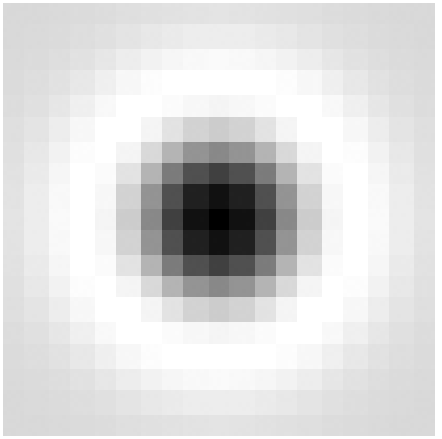


maximum

Blob detection in 2D

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

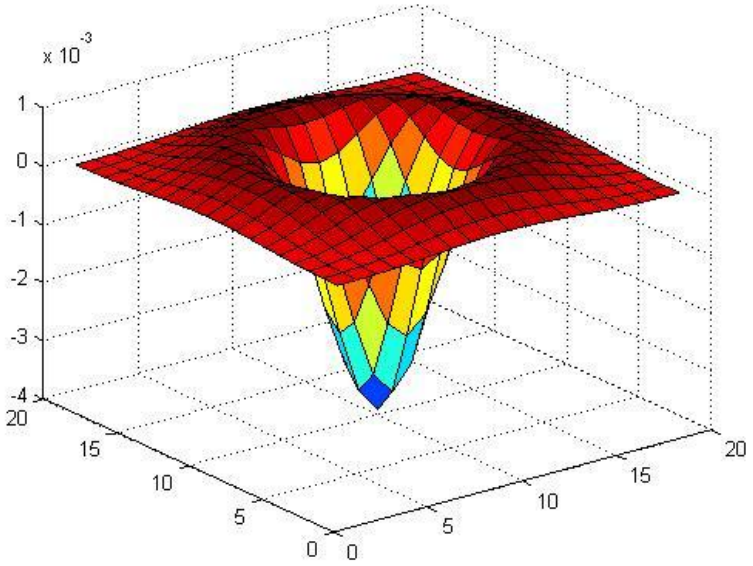


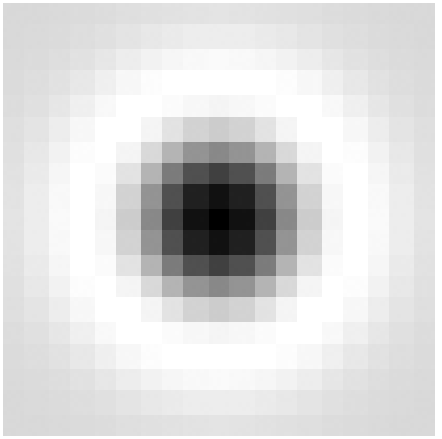


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} = \frac{1}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Blob detection in 2D

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

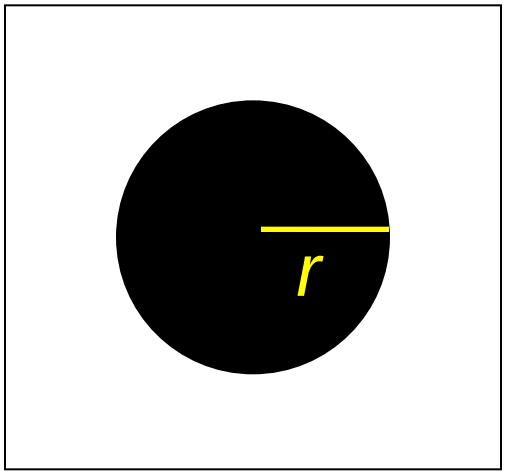




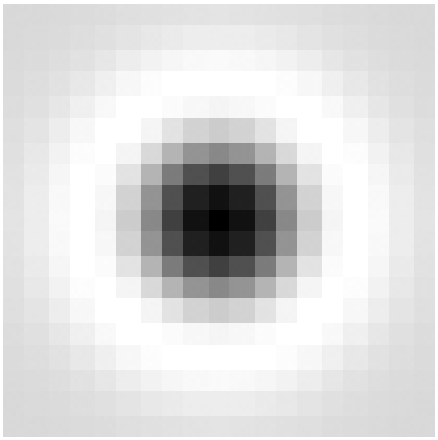
Scale-normalized: $$\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

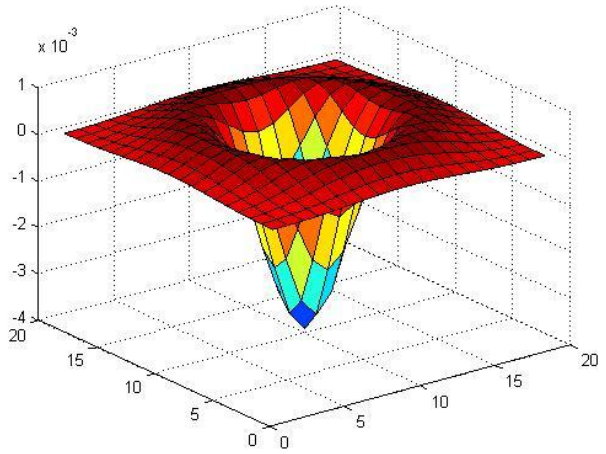
Scale selection

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



image



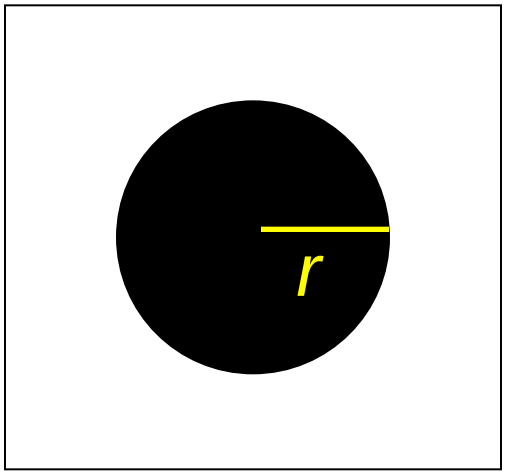


Laplacian

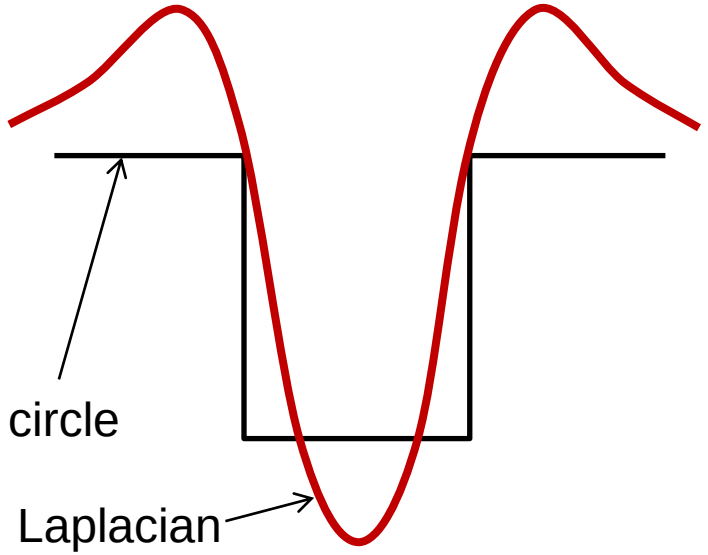
Scale selection

- At what scale is the max response to a binary circle of radius r?
- To get max response, the zeros of the Laplacian have to be aligned with the circle
- What part is the Zeros of Laplacian? $\frac{1}{\pi\sigma^4}\left(1-\frac{x^2+y^2}{2\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$
- Zeros given (up to scale): $\left(1-\frac{x^2+y^2}{2\sigma^2}\right)=0 \quad =\left(\frac{r^2}{2\sigma^2}\right)=1$
- Therefore, the maximum response occurs at

$$\sigma = r/\sqrt{2}.$$

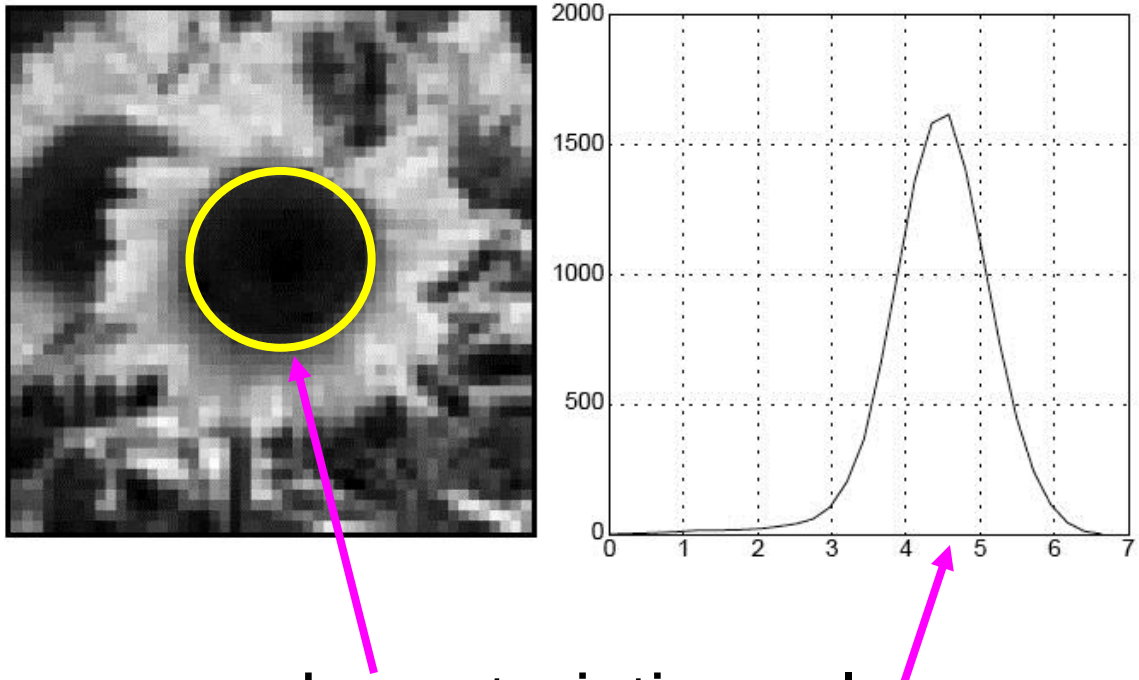


image



Characteristic scale

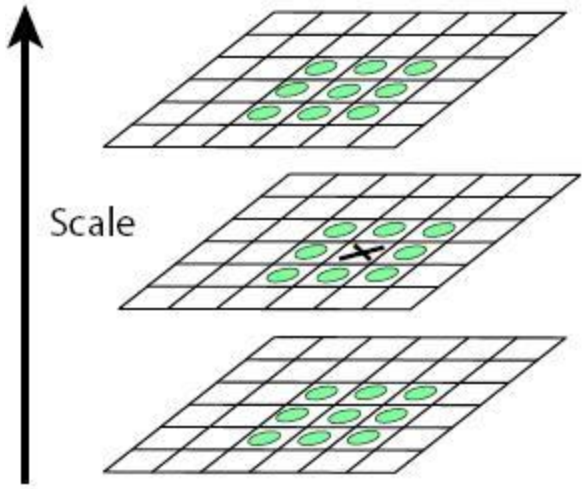
- We define the characteristic scale of a blob as the scale that produces peak of Laplacian response in the blob center



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

Scale-space blob detector

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

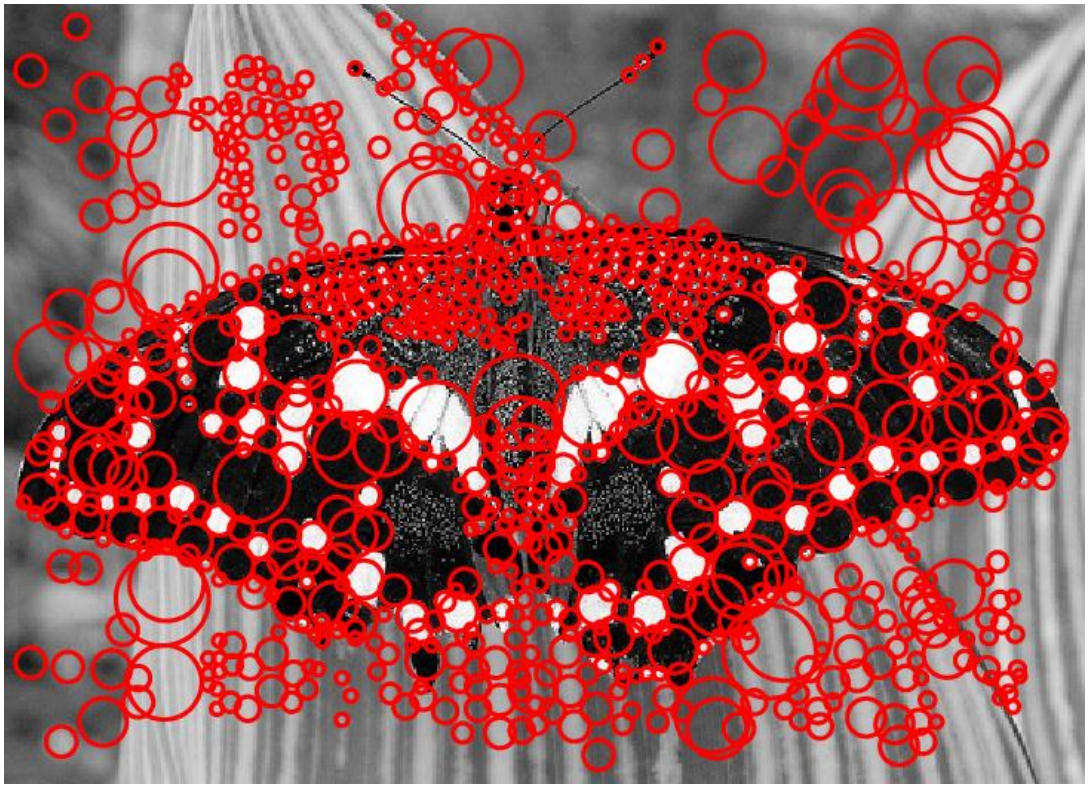


Scale-space blob detector: Example

Scale-space blob detector: Example

sigma = 11.9912

Scale-space blob detector: Example



Efficient implementation

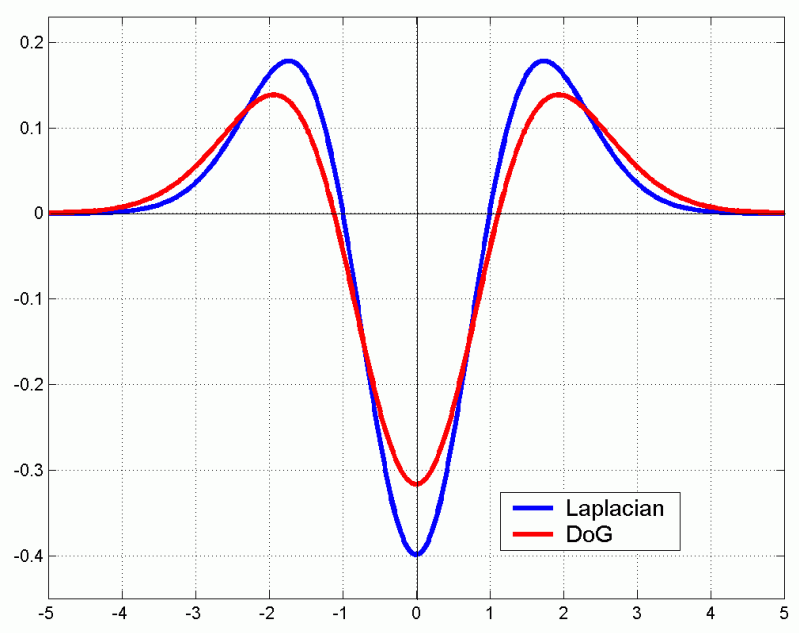
- Approximating the Laplacian with a difference of Gaussians:

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

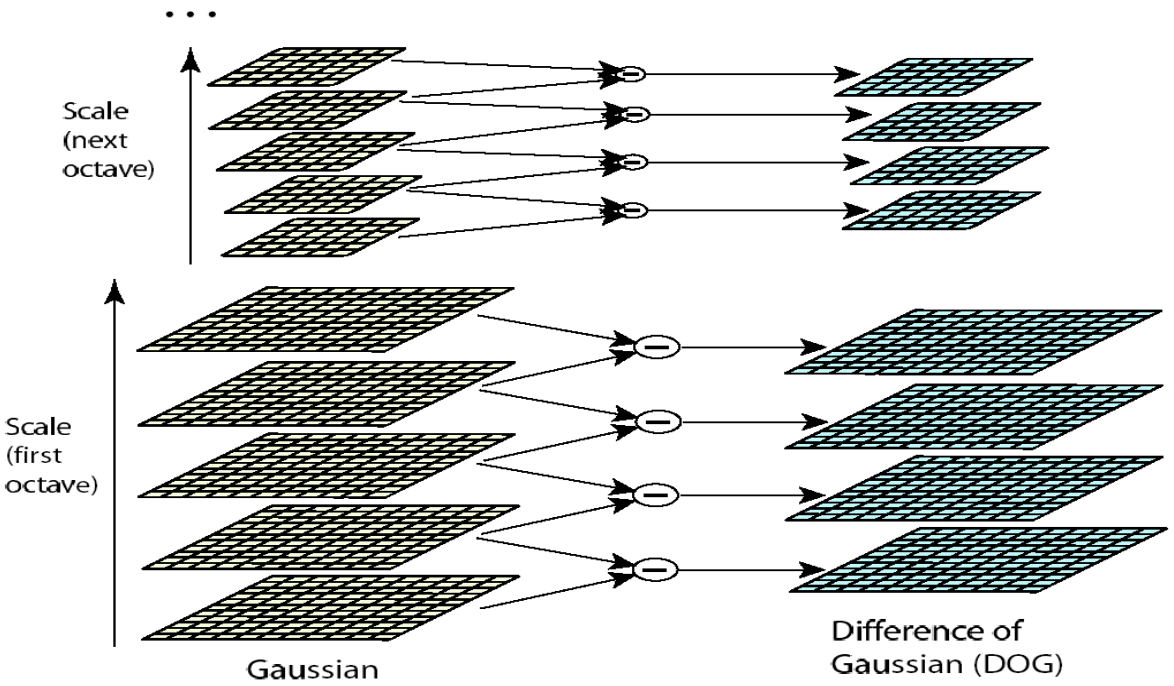
(Laplacian)

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

(Difference of Gaussians)



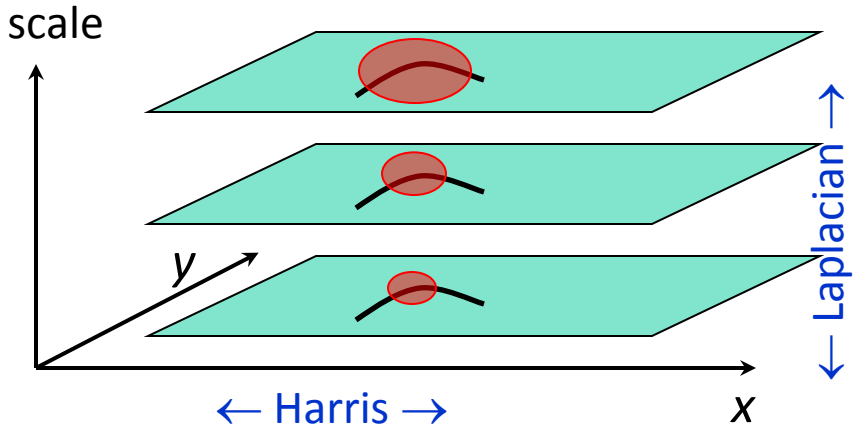
Efficient implementation



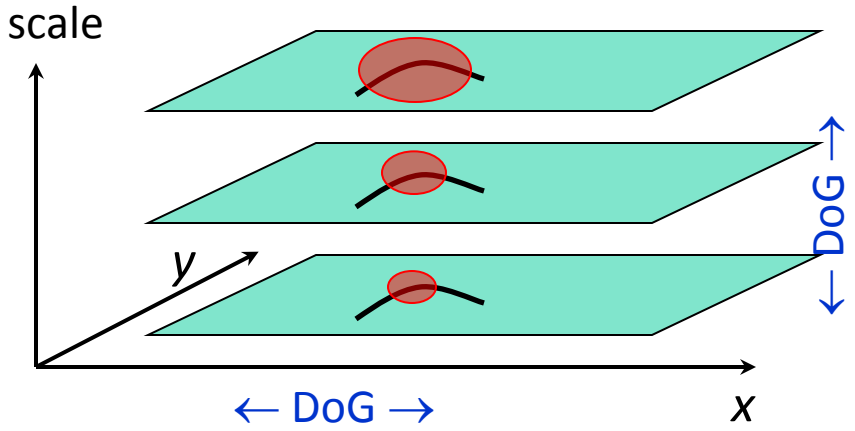
David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Scale Invariant Detectors

- Harris-Laplacian[1]
  Find local maximum of:
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale



- Difference of Gaussians
- a.k.a. SIFT detector[2]
  Find local maximum of:
  - Difference of Gaussians in space and scale



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

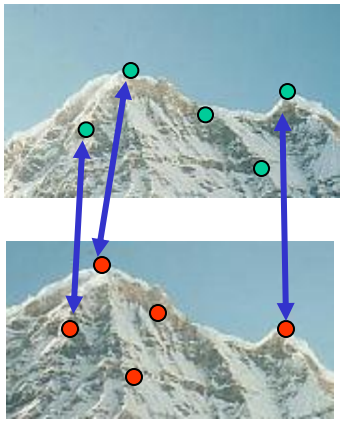
[2] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJCV 2004

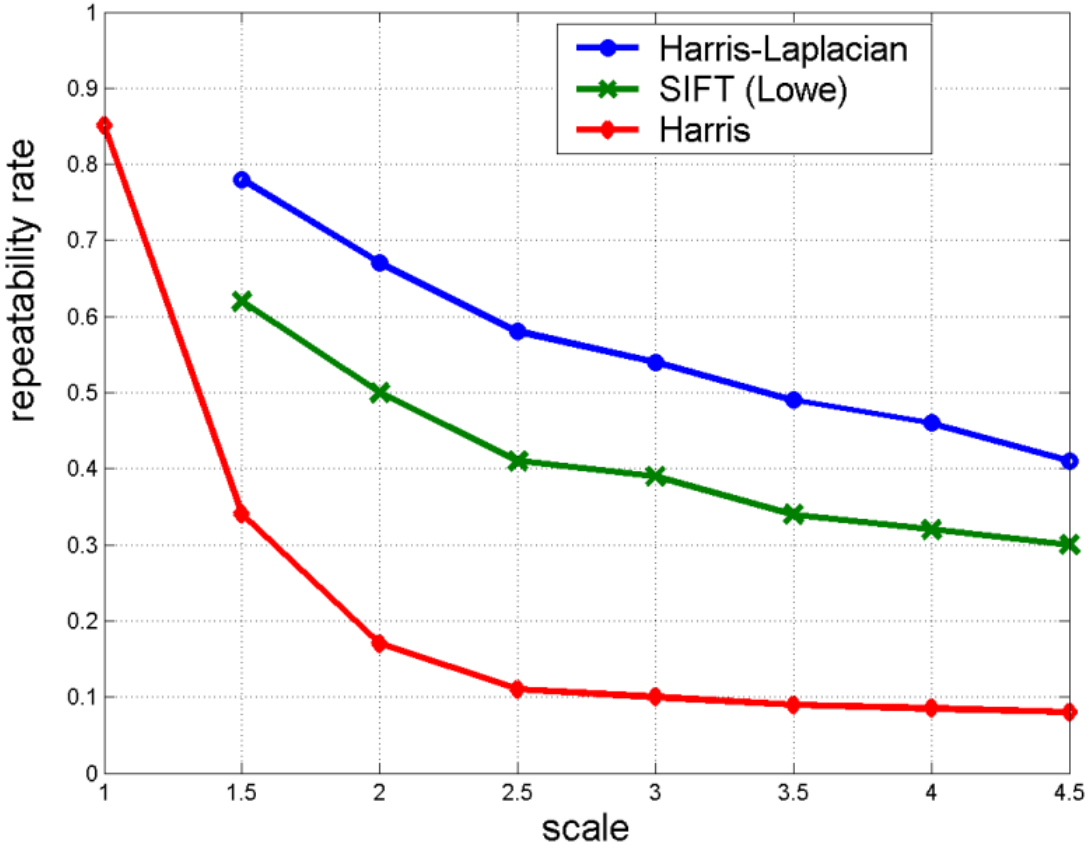
Scale Invariant Detectors

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

Repeatability rate:

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





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

Invariance and covariance properties

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

Questions?

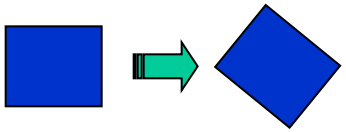
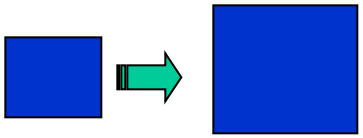
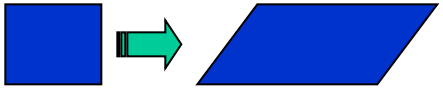
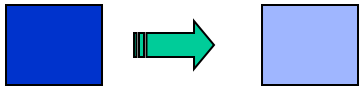
Matching with Features

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

We need a reliable and distinctive descriptor

Regardless of image changes

Models of Image Change

- Geometry
  - Rotation 
  - Similarity (rotation + uniform scale) 
  - Affine 
    valid for: orthographic camera, locally planar object
- Photometry
  - Affine intensity change ($I \rightarrow a\,I + b$) 

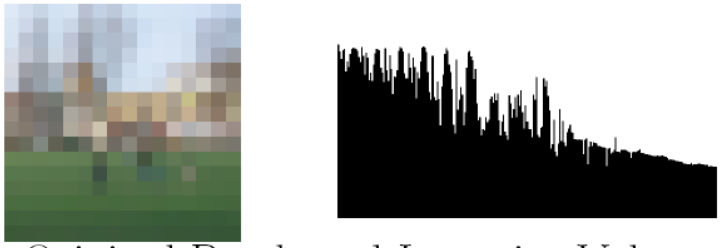
Cross-Correlation

$$CC(P_1, P_2) = \frac{1}{N} \sum_{i}^{N} P_1[i] P_2[i].$$

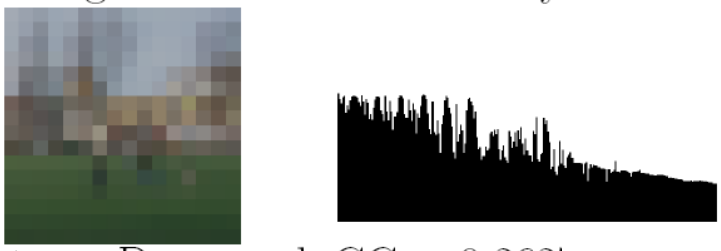
Affine photometric transformation:
$I \rightarrow a\, I + b$

- Output in range
  +1 → -1

- Not invariant
  to changes in a,b



Original Patch and Intensity Values



Brightness Decreased, CC = 0.262

Contrast increased, CC = 0.380

Normalized Cross-Correlation

- Make each patch zero mean:

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

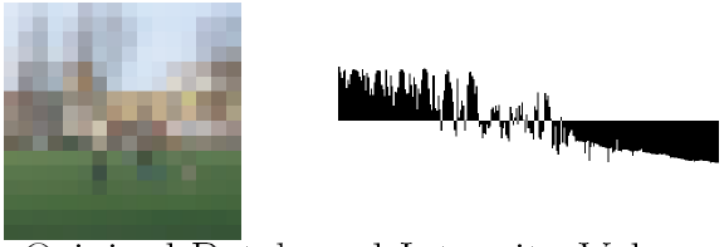
- $Z(x,y) = I(x,y) - \mu$

variance:

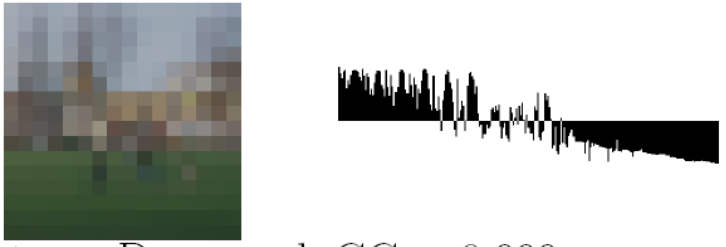
$$\sigma^2 = \frac{1}{N} \sum_{x,y} Z(x,y)^2$$

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

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



Original Patch and Intensity Values



Brightness Decreased, CC = 0.999

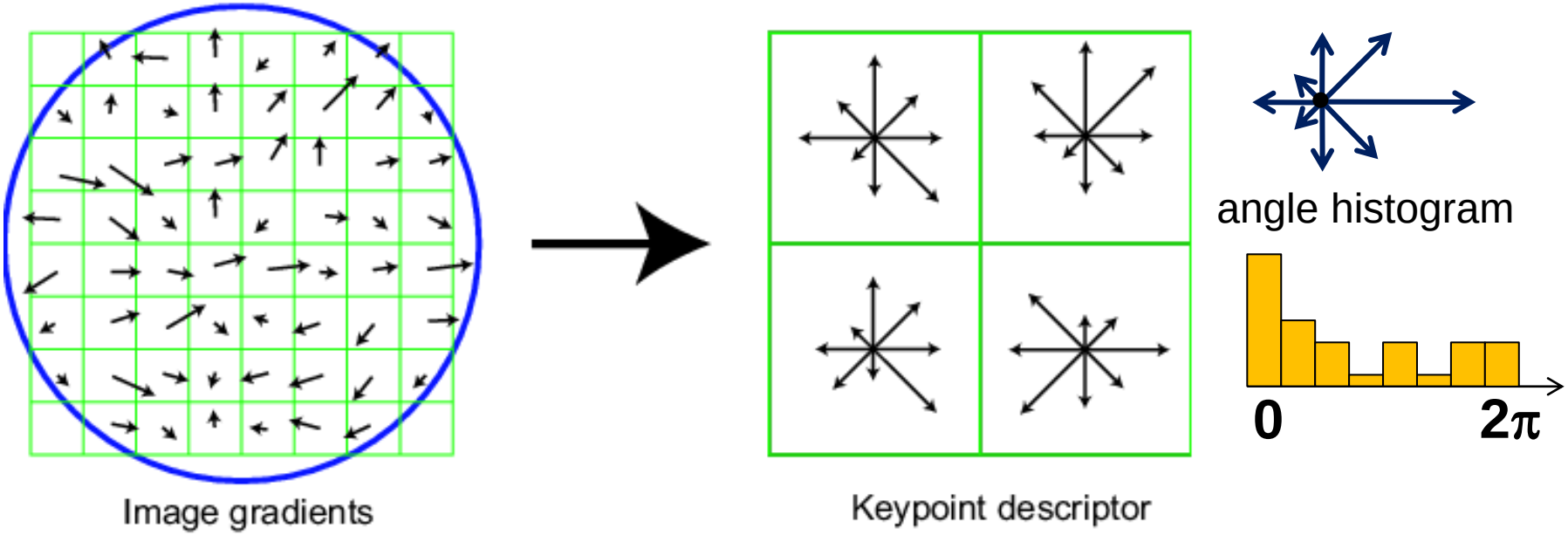
Contrast increased, CC = 0.969

Scale Invariant Feature Transform

David Lowe IJCV 2004

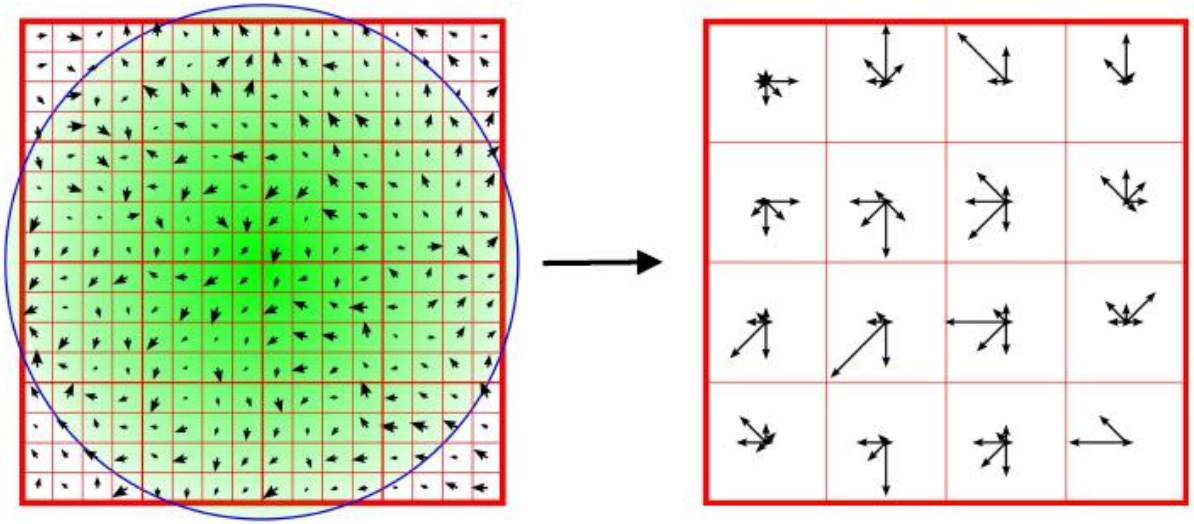
Basic idea:

- Take 16x16 square window around detected feature
- Rotate towards the whole gradient
- Compute edge orientation (angle of the gradient) for each pixel
- Group pixels together in a spatial bins of 4x4
- Create histogram of 8 edge orientations multiplied by the magnitude
- Normalize descriptor to size 1



Orientation Histogram

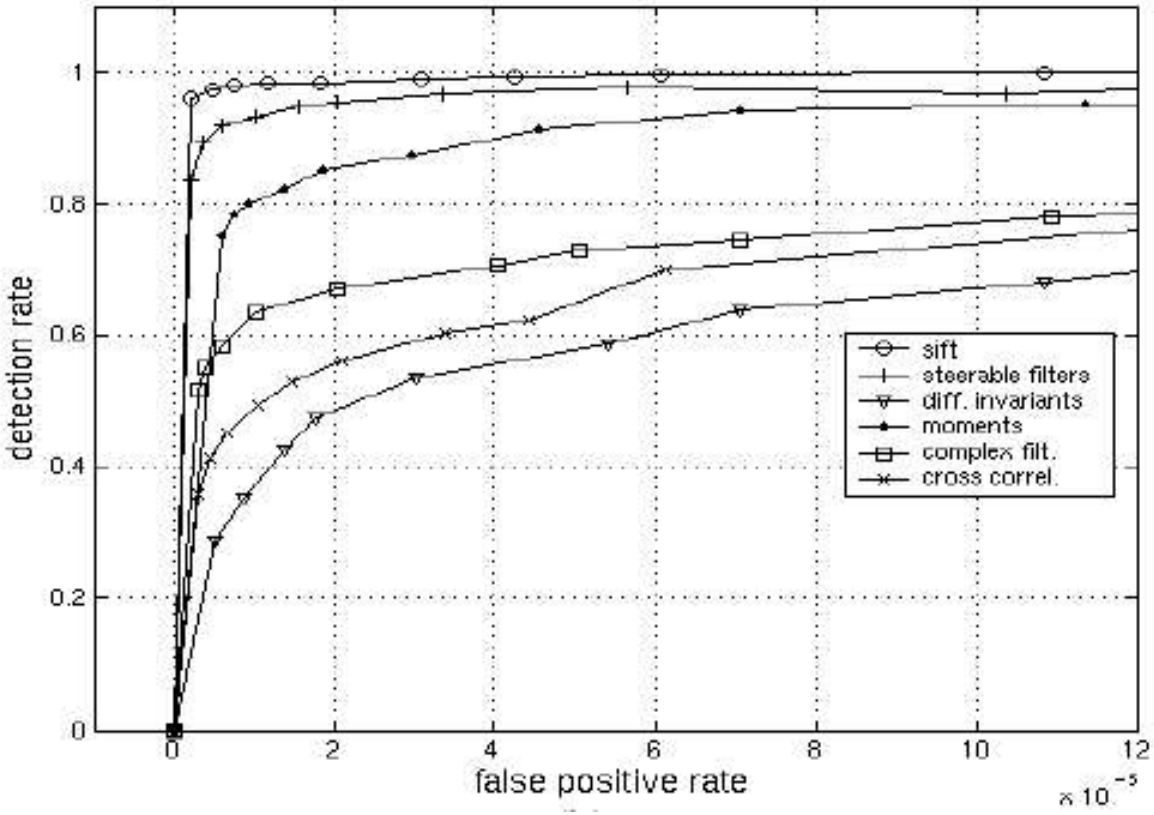
- 4x4 spatial bins (16 bins total)
- Gaussian center-weighting
- 8-bin orientation histogram per bin
- 8 x 16 = 128 dimensions total
- Normalized to unit norm



SIFT – Scale Invariant Feature Transform[1]

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

Scale = 2.5
Rotation = 45^0



[1] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJCV 2004
[2] K.Mikolajczyk, C.Schmid. "A Performance Evaluation of Local Descriptors". CVPR 2003

SIFT invariances

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

Summary of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time

Feature matching

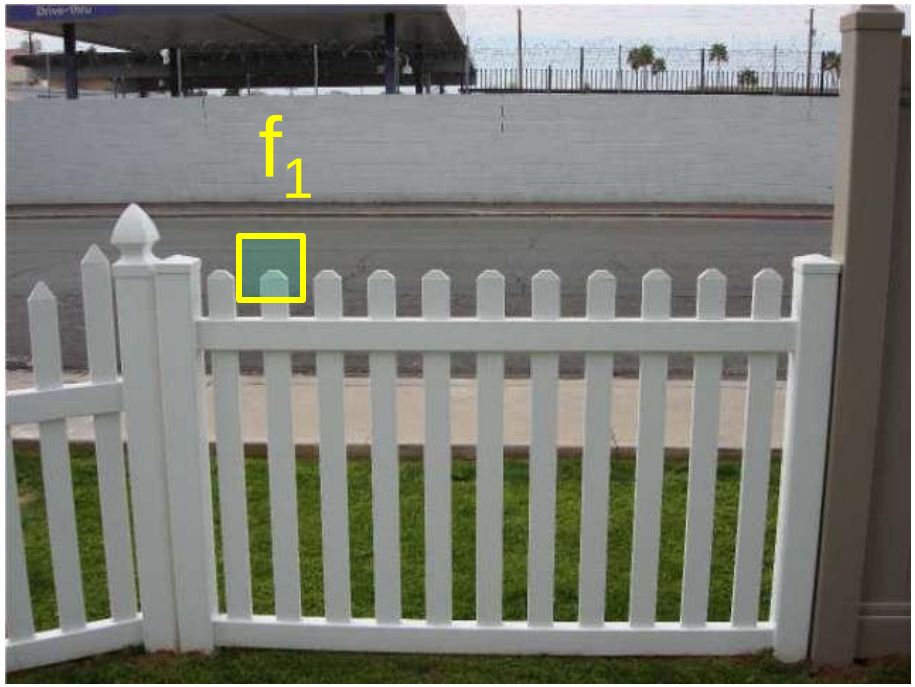
Given a feature in I_1, how to find the best match in I_2?

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

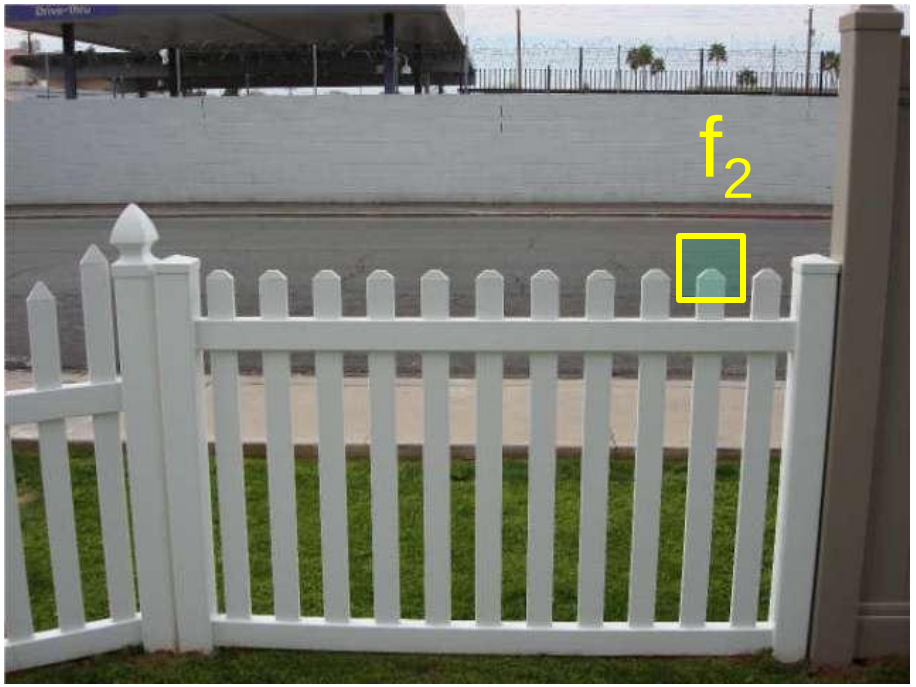
Feature distance

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

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches



I_1



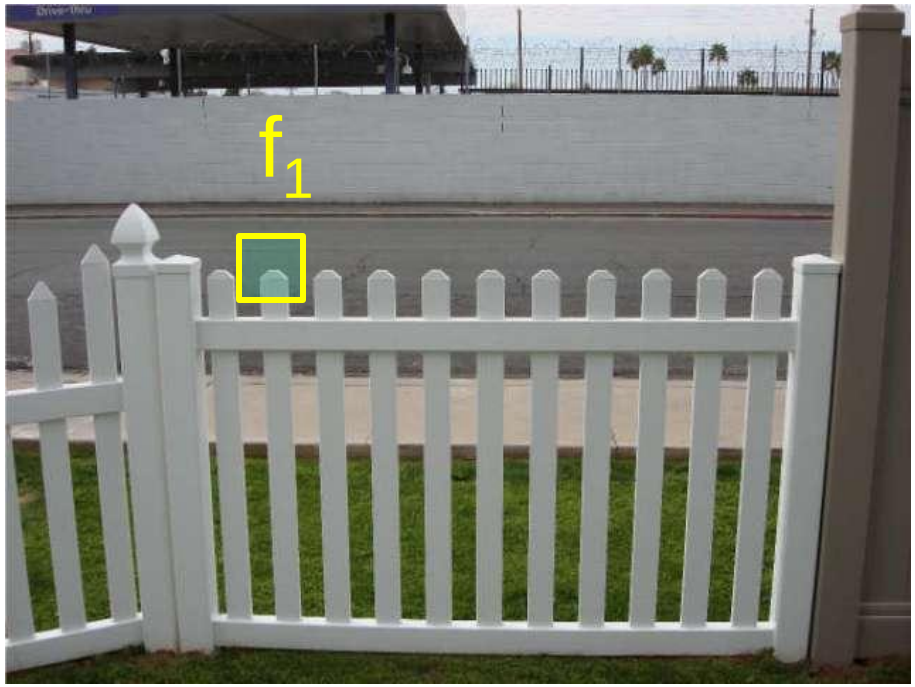
I_2

Slide: S. Seitz

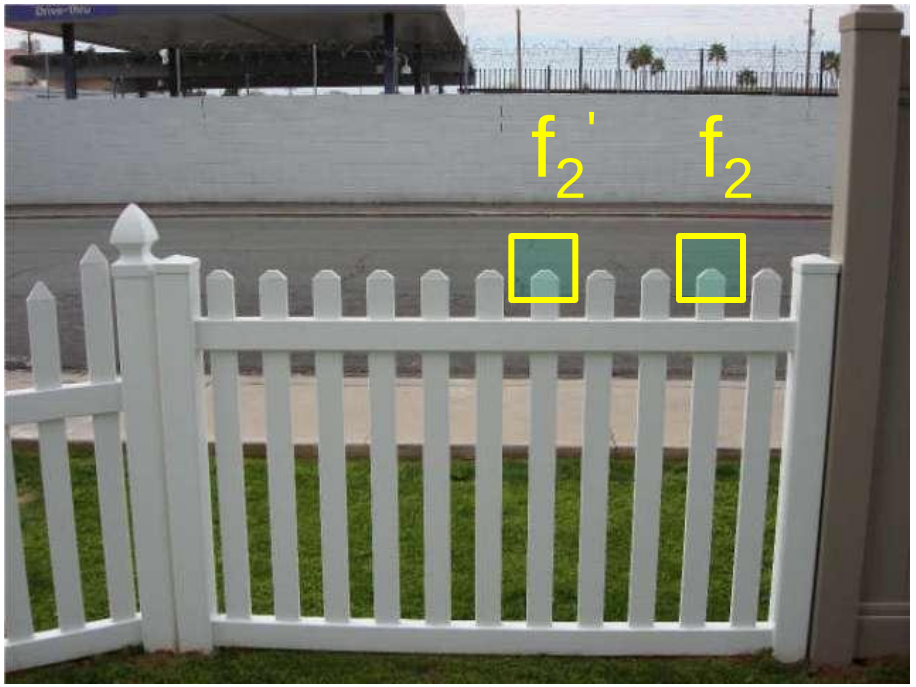
Feature distance

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

- Better approach: ratio distance = SSD(f_1, f_2) / SSD(f_1, f_2')
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2nd best SSD match to f_1 in I_2
  - gives small values for ambiguous matches



I_1

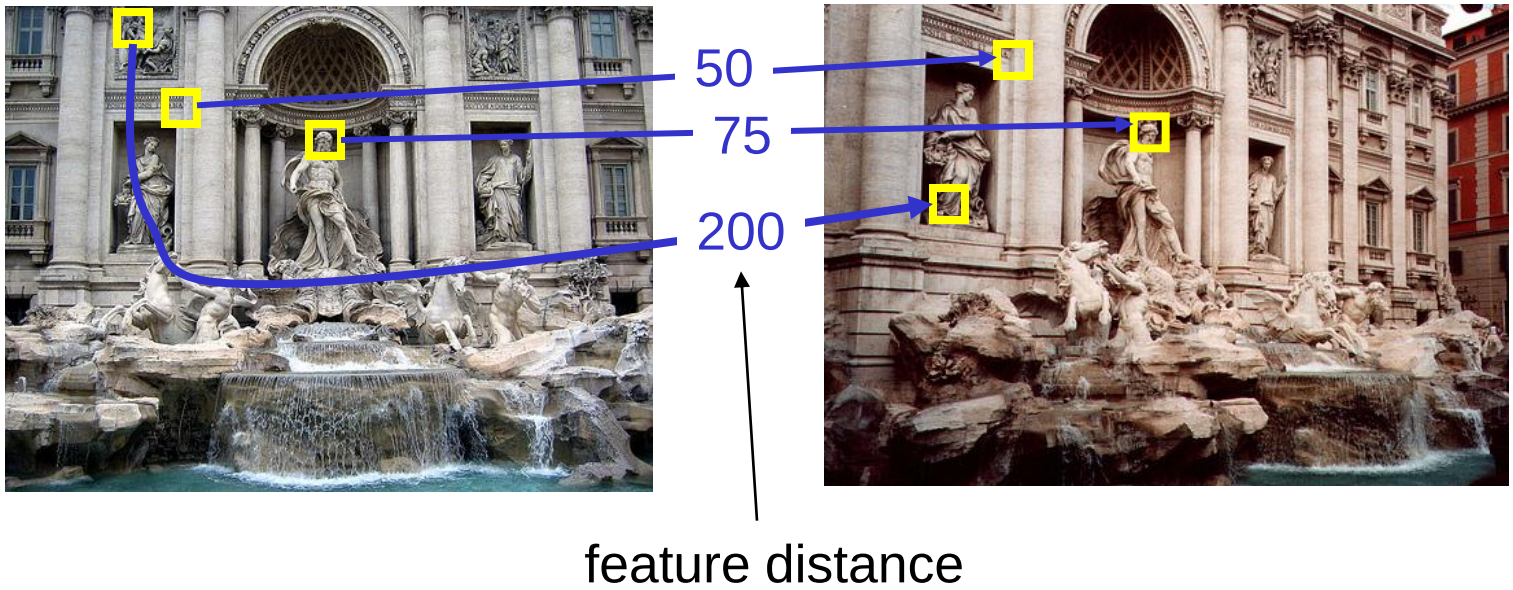


I_2

Slide: S. Seitz

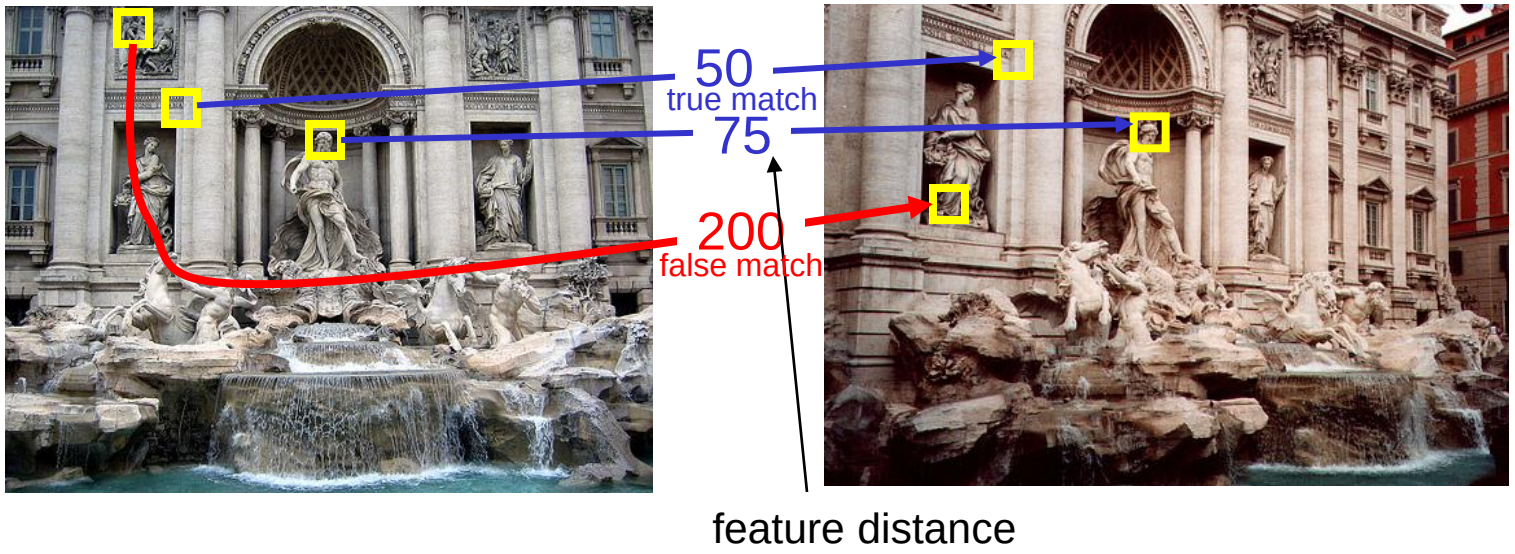
Evaluating the results

How can we measure the performance of a feature matcher?



Slide: S. Seitz

True/false positives



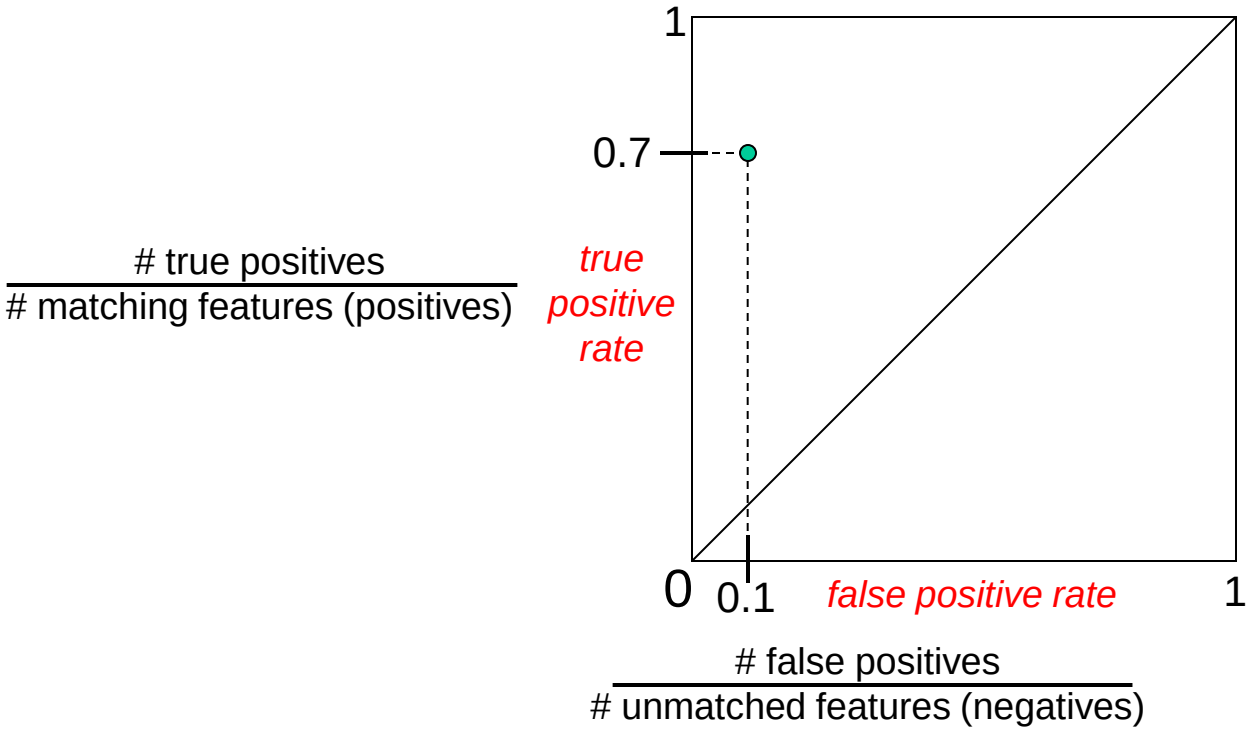
The distance threshold affects performance

- True positives = # of detected matches that are correct
  - Suppose we want to maximize these—how to choose threshold?
- False positives = # of detected matches that are incorrect
  - Suppose we want to minimize these—how to choose threshold?

Slide: S. Seitz

Evaluating the results

How can we measure the performance of a feature matcher?

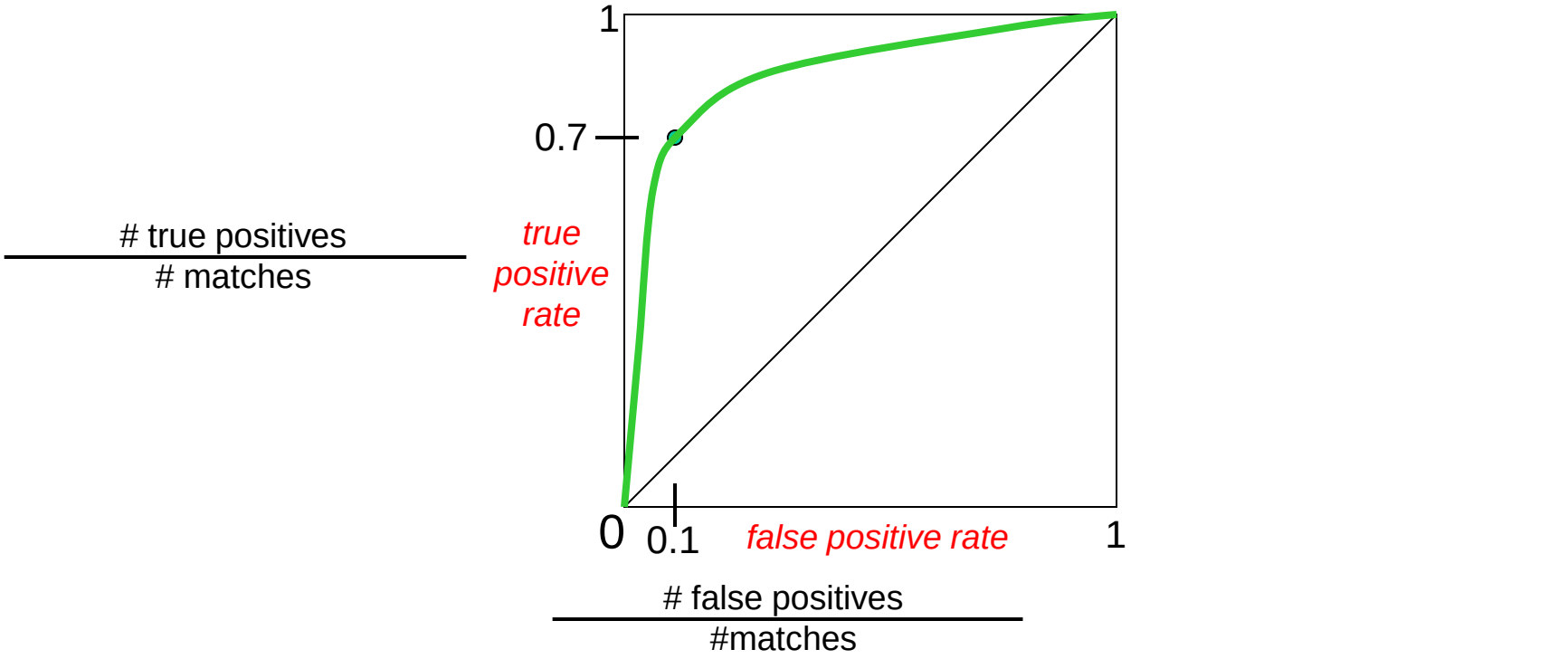


Slide: S. Seitz

Evaluating the results

How can we measure the performance of a feature matcher?

ROC curve (“Receiver Operator Characteristic”)



ROC Curves

- Generated by counting # current/incorrect matches, for different thresholds
- Want to maximize area under the curve (AUC)
- Useful for comparing different feature matching methods

Slide: S. Seitz

Questions?

Edges

http://todayinart.com/files/2009/12/500x388xblind-contour-line-drawing.png.pagespeed.ic.DOli66Ckz1.png

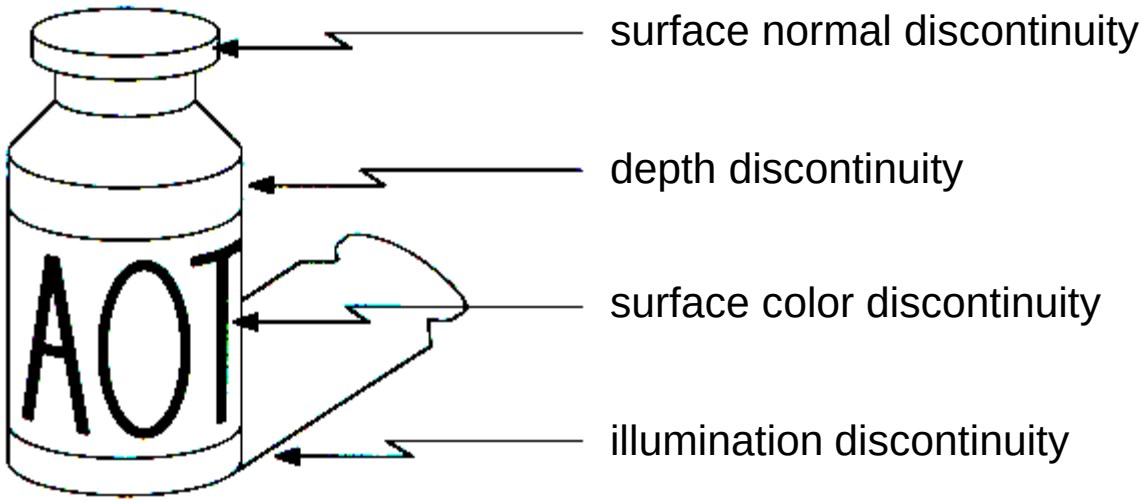
Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

Origin of edges

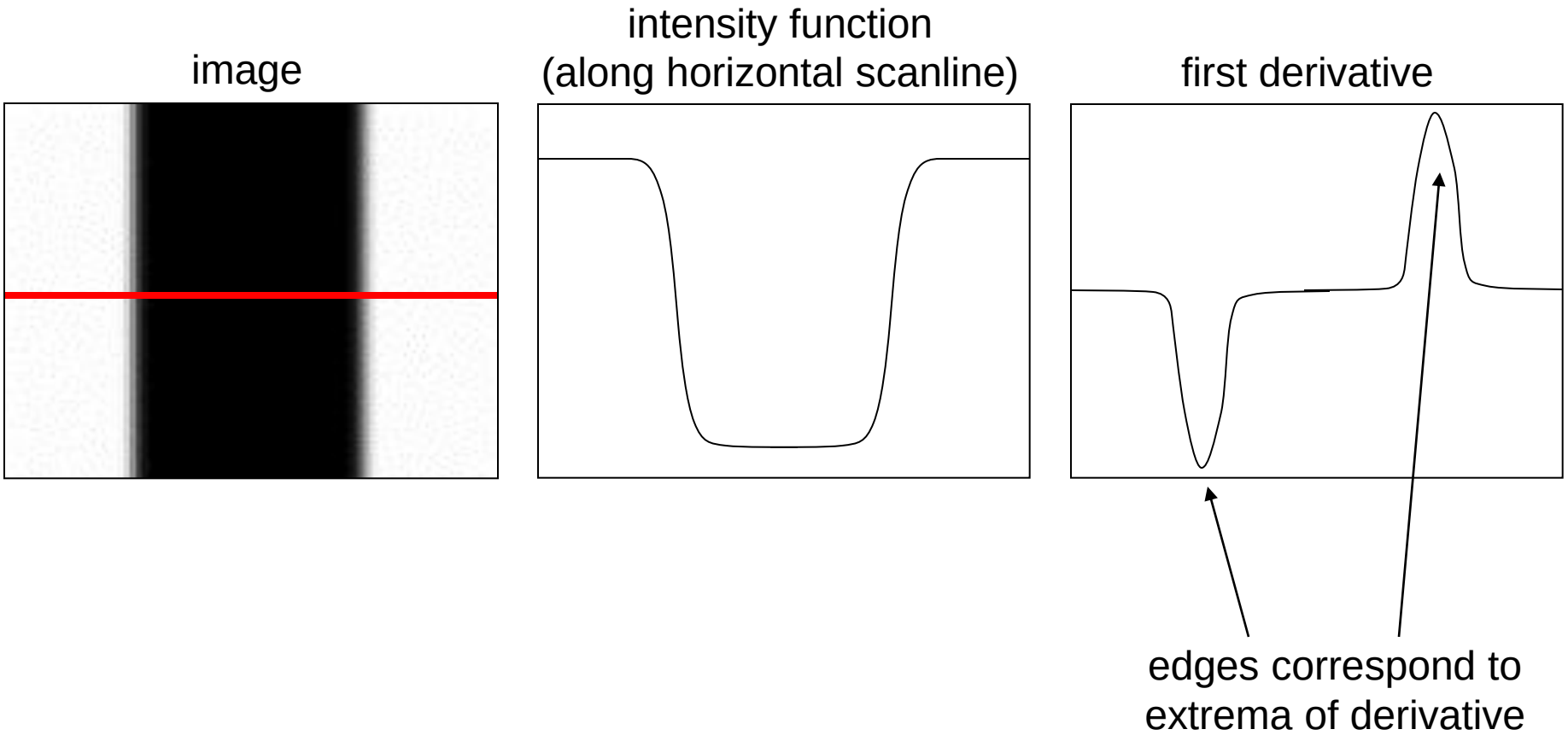
- Edges are caused by a variety of factors:



Source: Steve Seitz

Characterizing edges

- An edge is a place of rapid change in the image intensity function



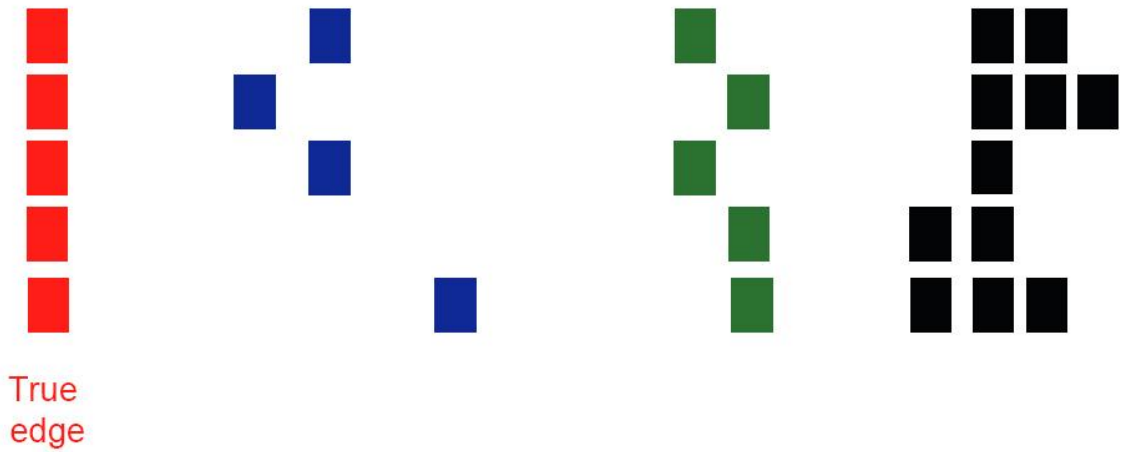
Implementation issues

- The gradient magnitude is large along a thick “trail” or “ridge,” so how do we identify the actual edge points?
- How do we link the edge points to form curves?

Source: D. Forsyth

Designing an edge detector

- Criteria for an "optimal" edge detector:
  - **Good detection:** false positives / false negatives **Good localization:** close to the true edges
  - **Single response:** one point for each true edge point;



Source: L. Fei-Fei

Canny edge detector

- This is probably the most widely used edge detector in computer vision
- MATLAB: edge(image, ‘canny’)

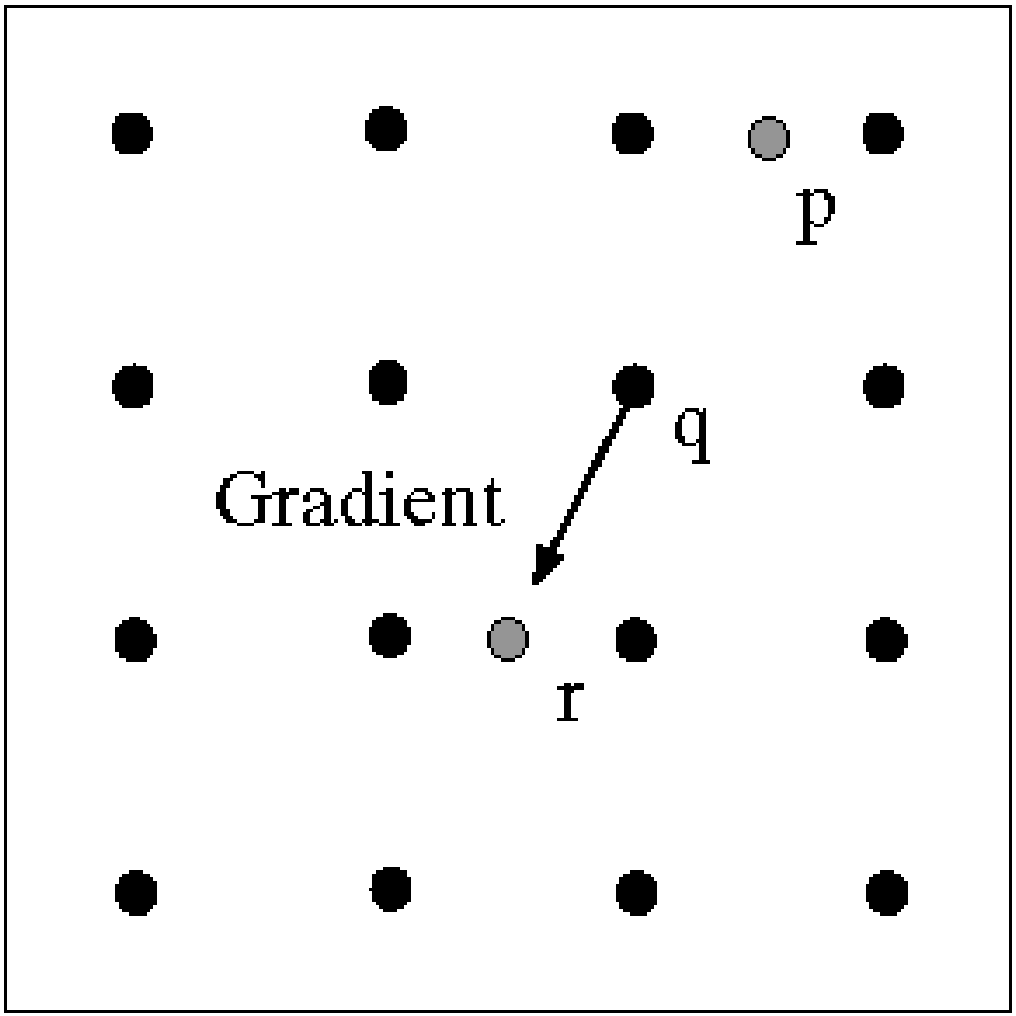
Algorithm:

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Shrink multi-pixel wide “ridges” to single pixel width

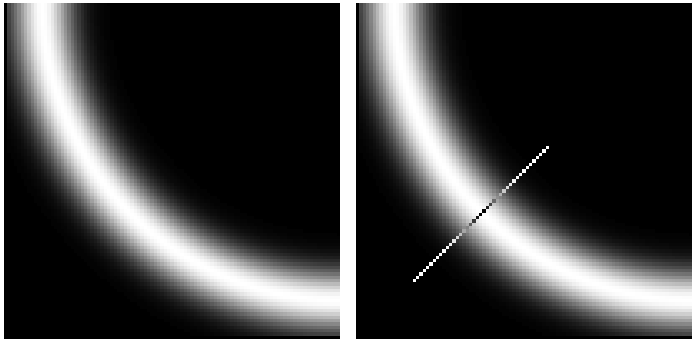
J. Canny, *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Source: D. Lowe, L. Fei-Fei

Non-maximum suppression



At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.

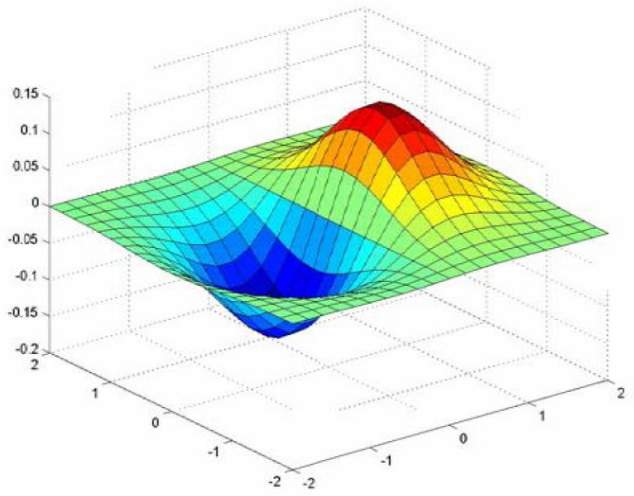


Source: D. Forsyth

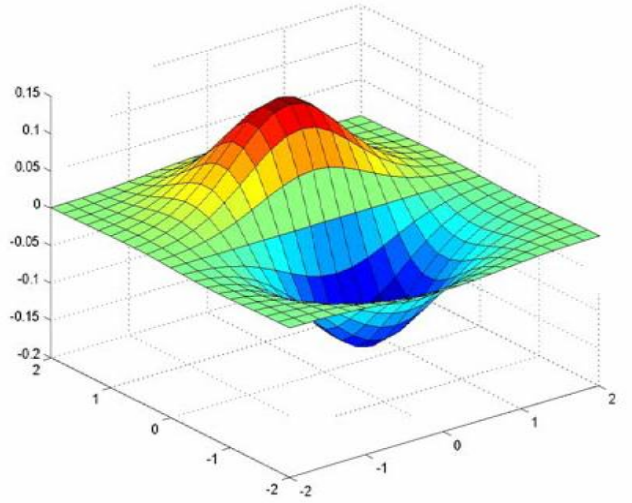
Example

original image (Lena)

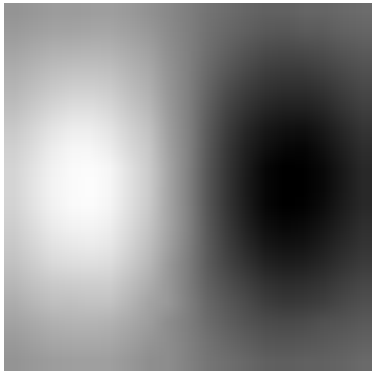
Derivative of Gaussian filter

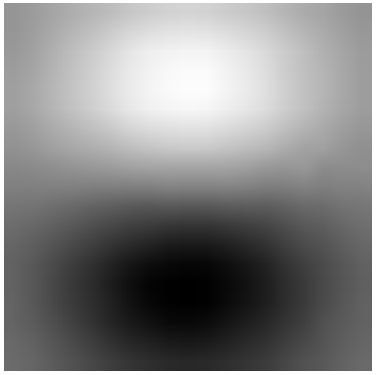


x-direction



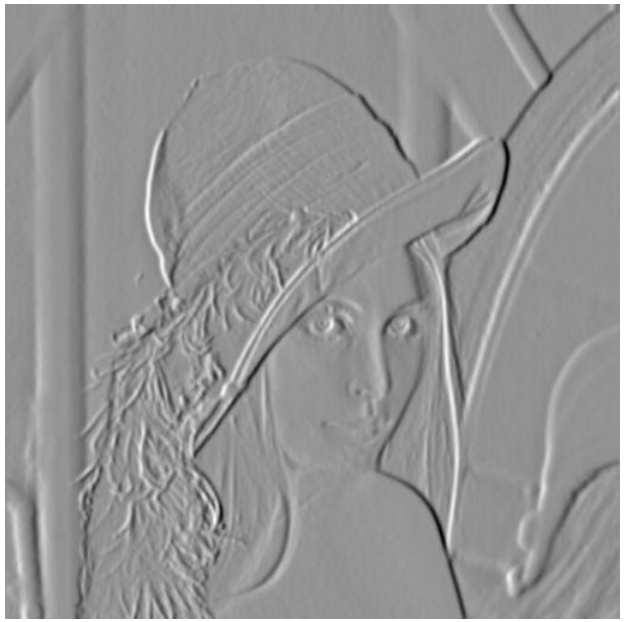
y-direction





Compute Gradients (DoG)

Derivative of Gaussian

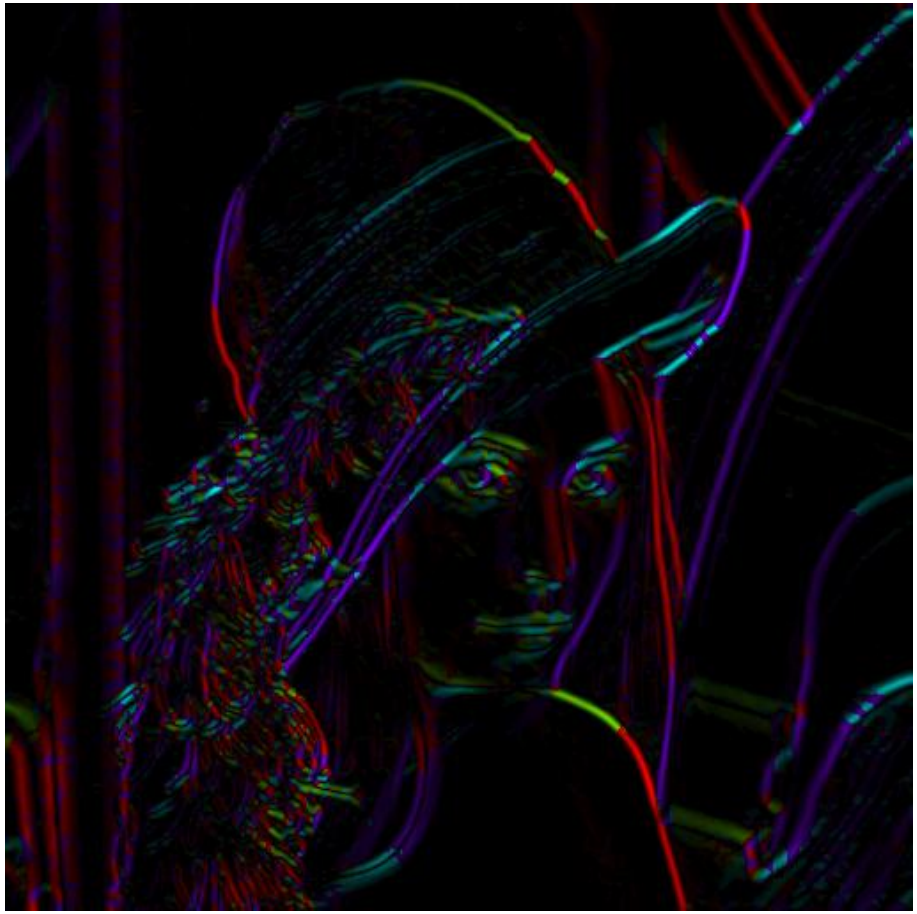


Derivative of Gaussian

Gradient Magnitude

- Which one is the gradient in the x-direction (resp. y-direction)?

Example



Orientation of the gradient

theta = atan2(gy, gx)

Example

norm of the gradient

Example

thresholding

Example

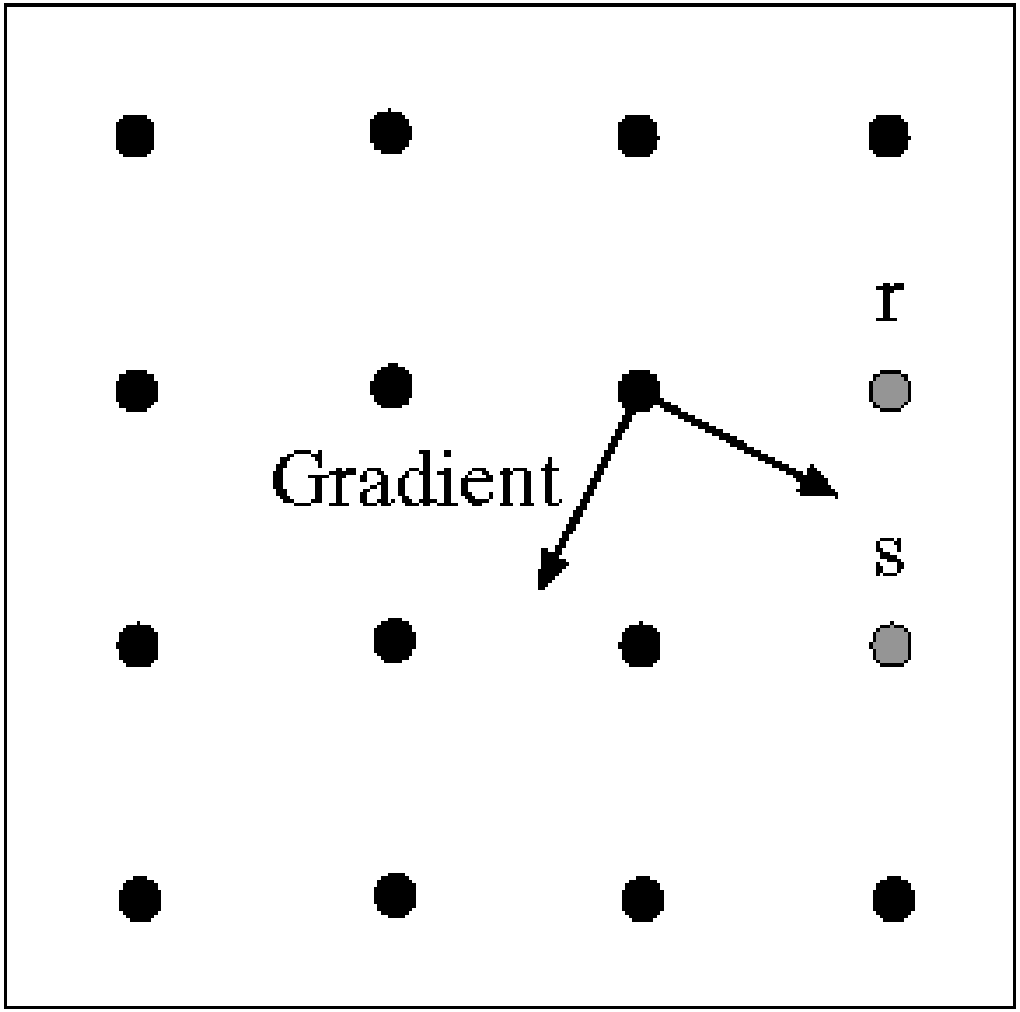
Non-maximum suppression

Canny edge detector

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression
   – Thin multi-pixel wide “ridges” down to single pixel width
4. Linking of edge points

Source: D. Lowe, L. Fei-Fei

Edge linking



Assume the marked point is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (here either r or s).

Source: D. Forsyth

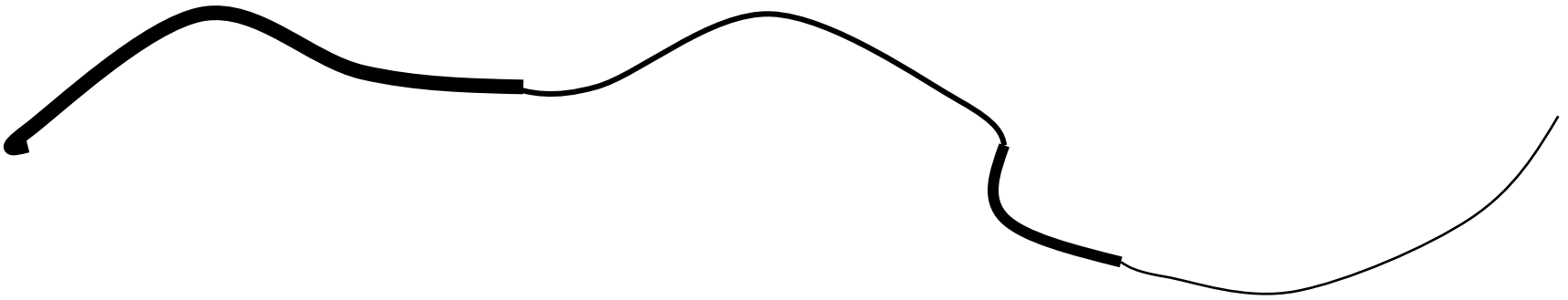
Canny edge detector

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression
   - Thin multi-pixel wide “ridges” down to single pixel width
4. Linking of edge points
   - Hysteresis thresholding: use a higher threshold to start edge curves and a lower threshold to continue them

Source: D. Lowe, L. Fei-Fei

Hysteresis thresholding

- Use a high threshold to start edge curves and a low threshold to continue them
  - Reduces *drop-outs*



Source: S. Seitz

Hysteresis thresholding

original image

high threshold
(strong edges)

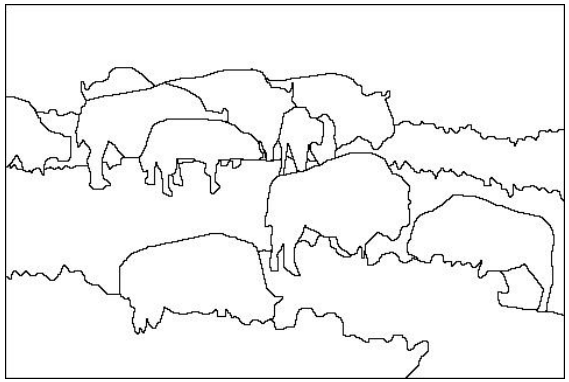
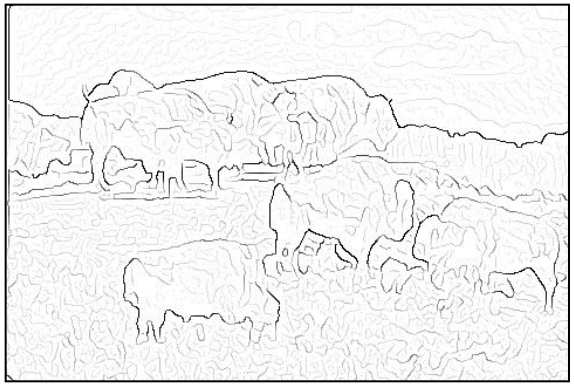
low threshold
(weak edges)

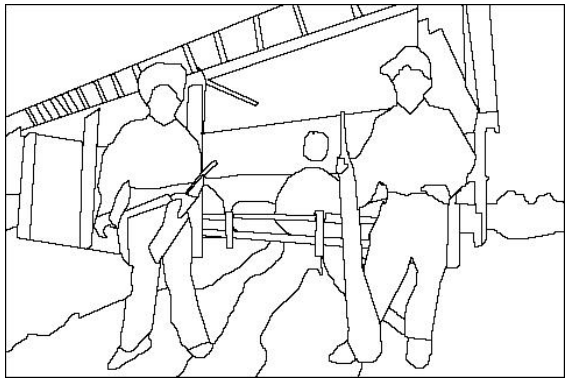
hysteresis threshold

Source: L. Fei-Fei

Edge detection is just the beginning…

image | human segmentation | gradient magnitude


- Berkeley segmentation database:
  http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Questions?

Hough transform

- An early type of voting scheme for line detection
- General outline:
  - Discretize parameter space into bins that represent lines
  - For each feature point in the image, put a vote in every line-bin in the that could have generated this point
  - Find bins that have the most votes

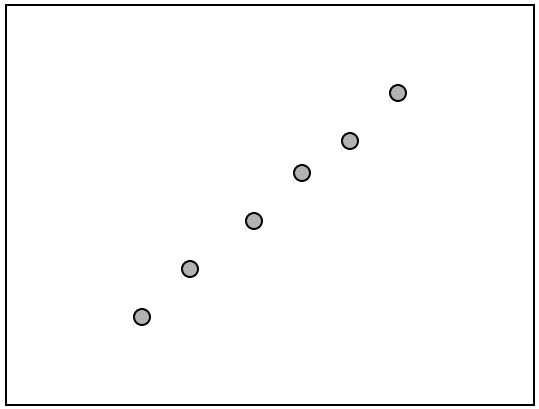
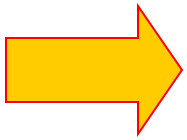
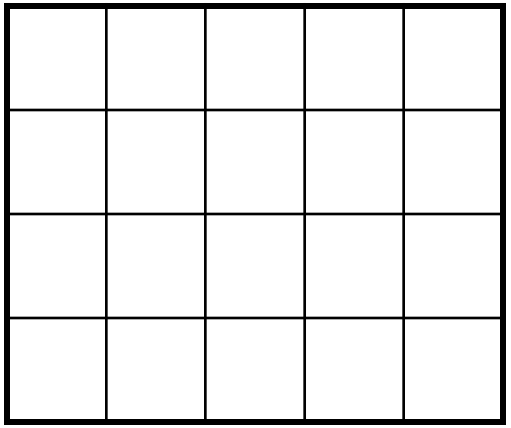


Image space



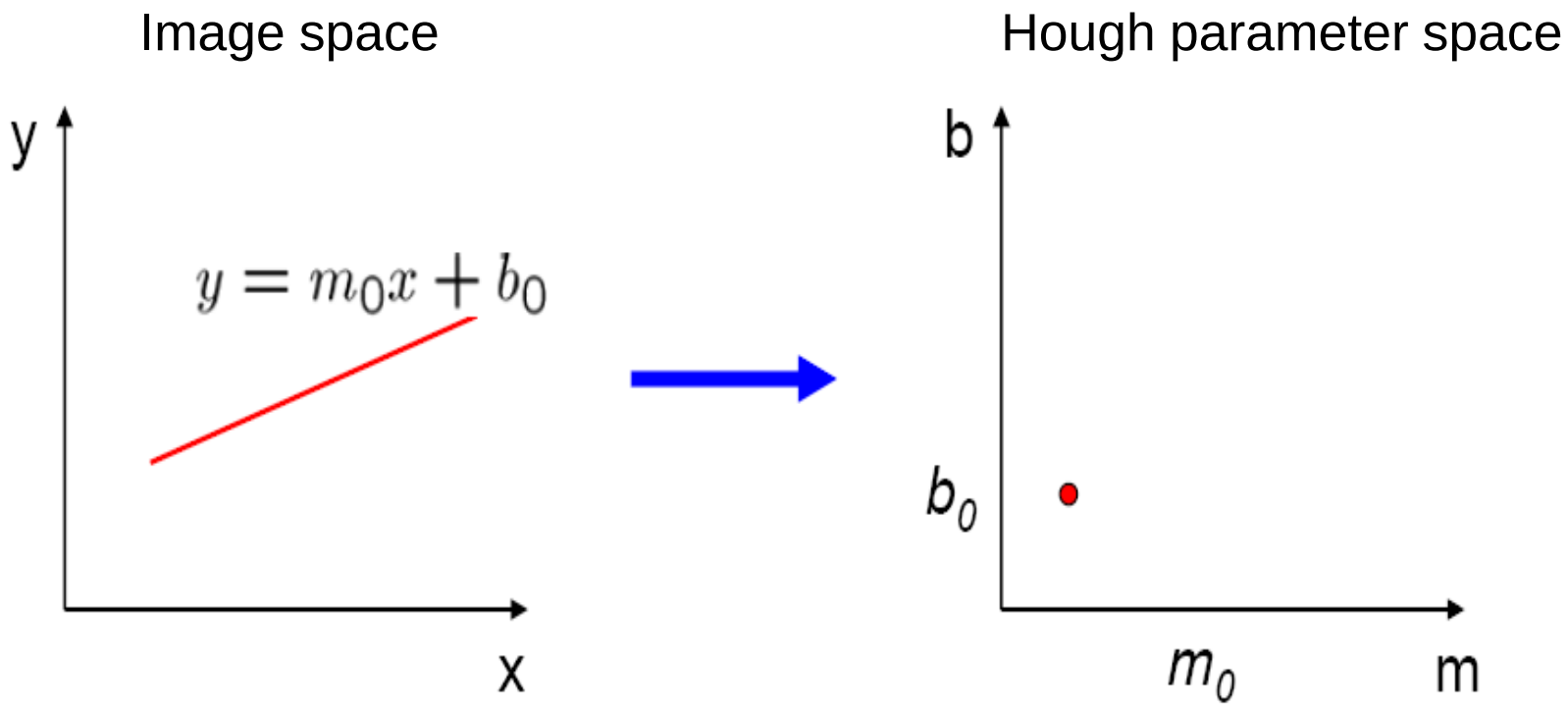


Hough parameter space

P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

Parameter space representation

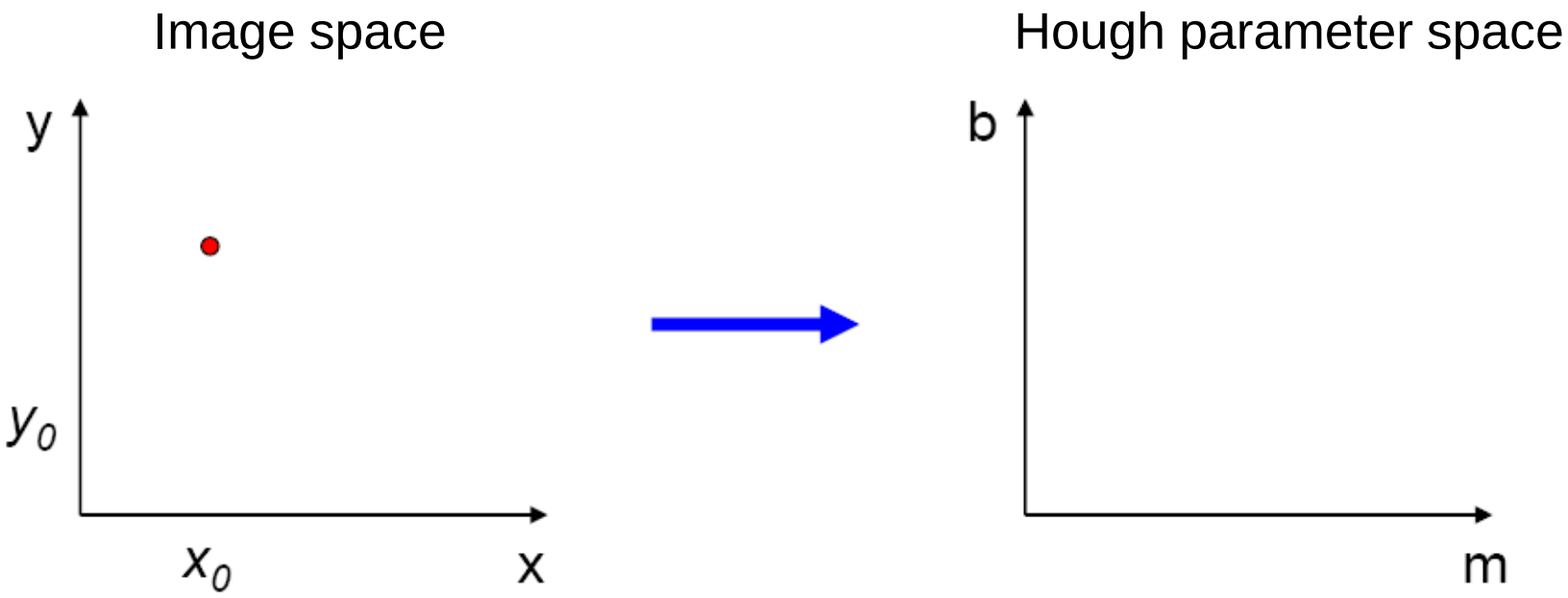
- A line in the image corresponds to a point in Hough space



Source: S. Seitz

Parameter space representation

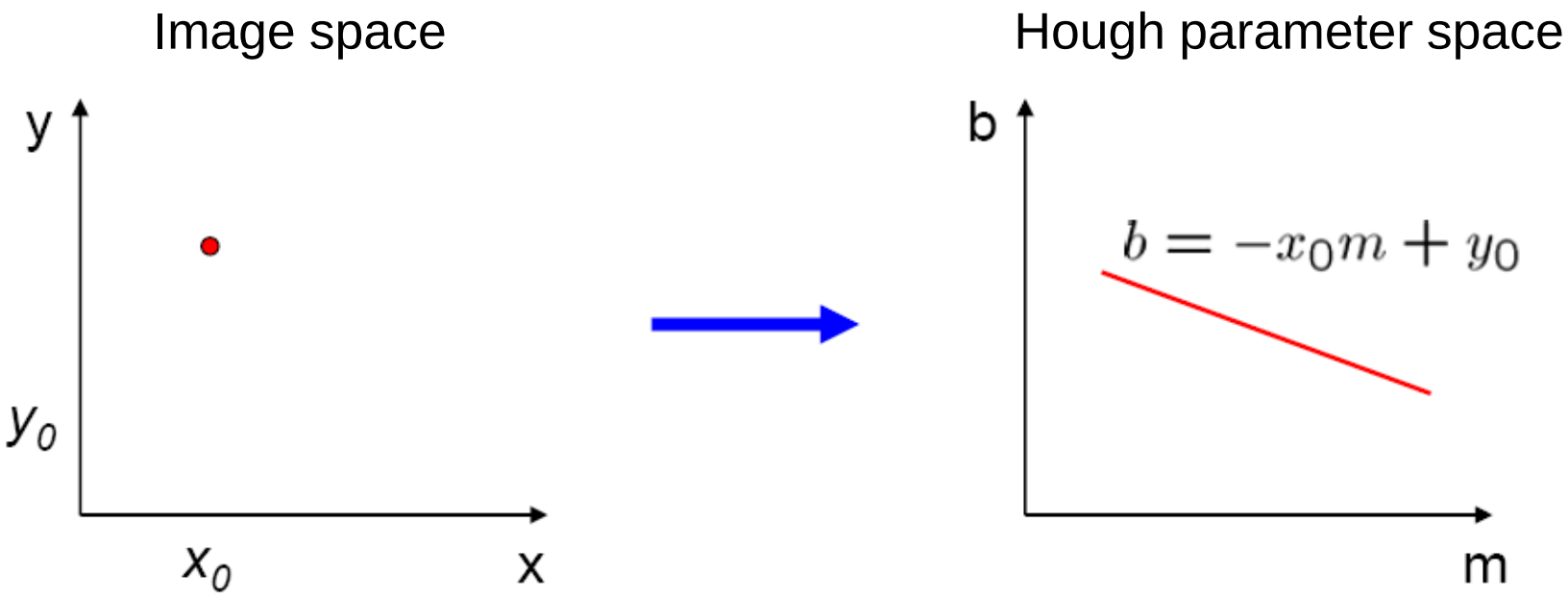
- What does a point (x_0, y_0) in the image space map to in the Hough space?



Source: S. Seitz

Parameter space representation

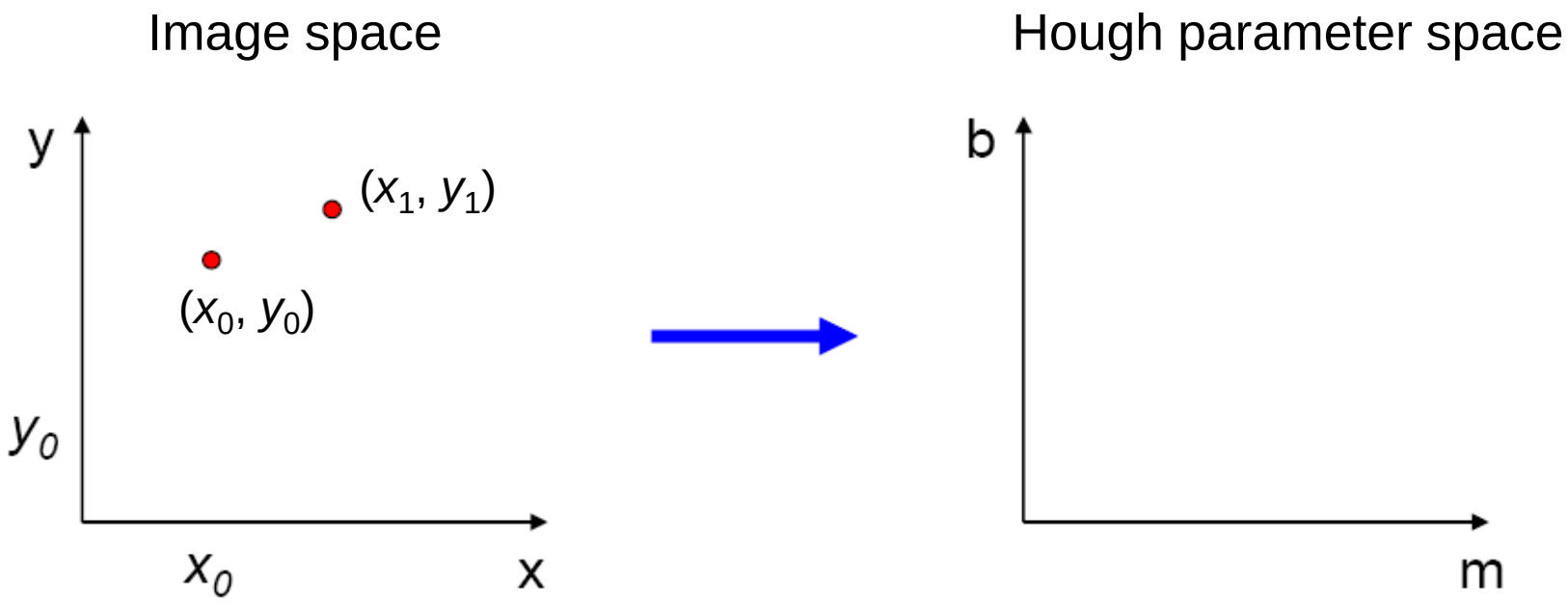
- What does a point (x_0, y_0) in the image space map to in the Hough space?
  - Answer: the solutions of $b = -x_0m + y_0$
  - This is a line in Hough space



Source: S. Seitz

Parameter space representation

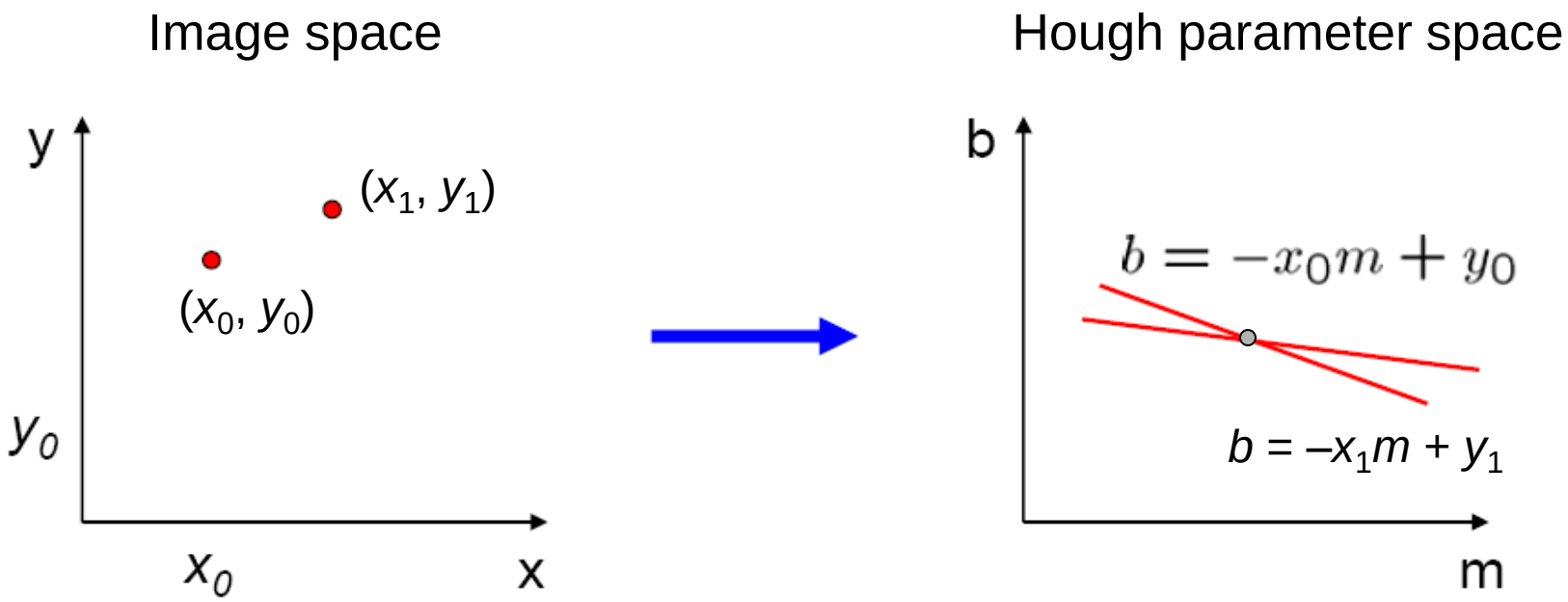
- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?



Source: S. Seitz

Parameter space representation

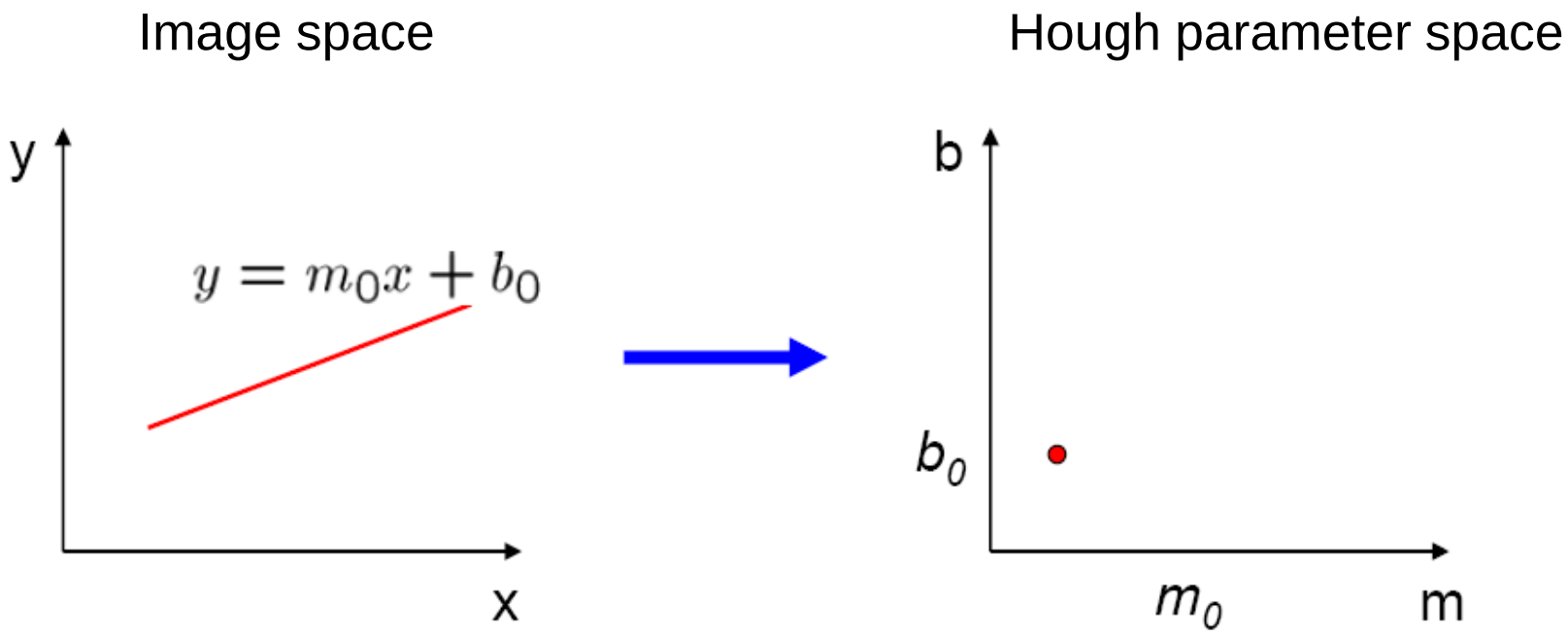
- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?
  - It is the intersection of the lines $b = –x_0m + y_0$ and $b = –x_1m + y_1$



Source: S. Seitz

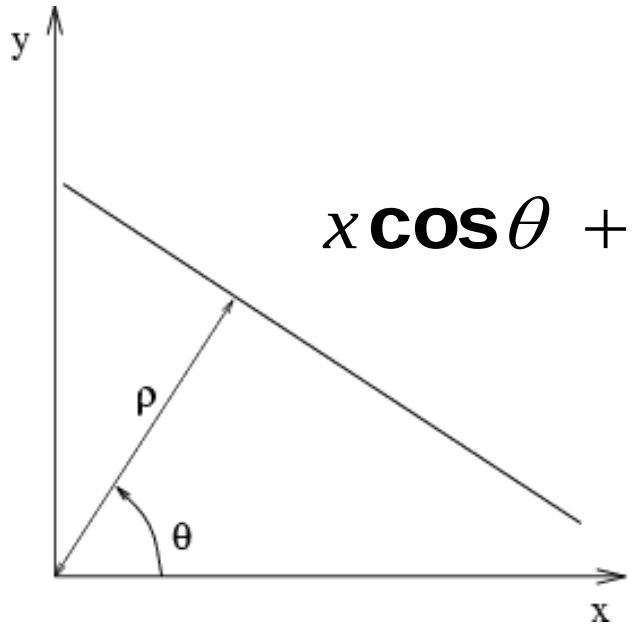
Parameter space representation

- Problems with the (m,b) space?
  - Unbounded parameter domain
  - Vertical lines require infinite m



Parameter space representation

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m
- Alternative: polar representation

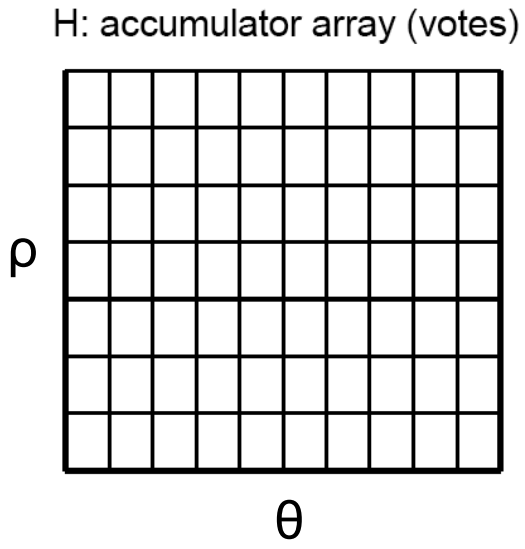


$$x\cos\theta + y\sin\theta = \rho$$

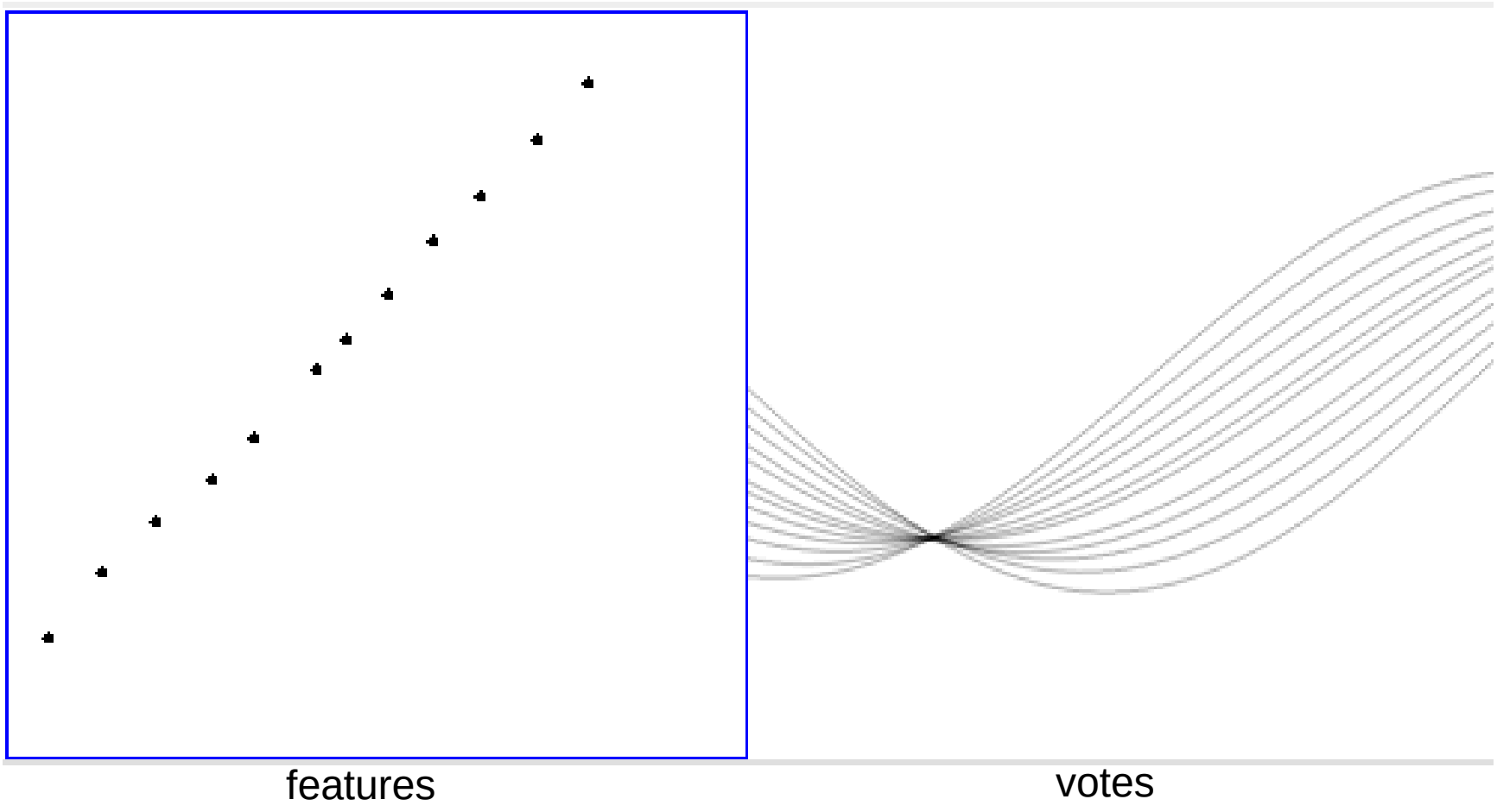
Each point will add a sinusoid in the (θ,ρ) parameter space

Algorithm outline

- Initialize accumulator H to all zeros
- For each edge point (x,y) in the image
  For $\theta = 0$ to 180
    $\rho = x \cos\theta + y \sin\theta$
    $H(\theta, \rho) = H(\theta, \rho) + 1$
  end
end
- Find the value(s) of (θ, ρ) where $H(\theta, \rho)$ is a local maximum
  - The detected line in the image is given by
    $\rho = x \cos\theta + y \sin\theta$



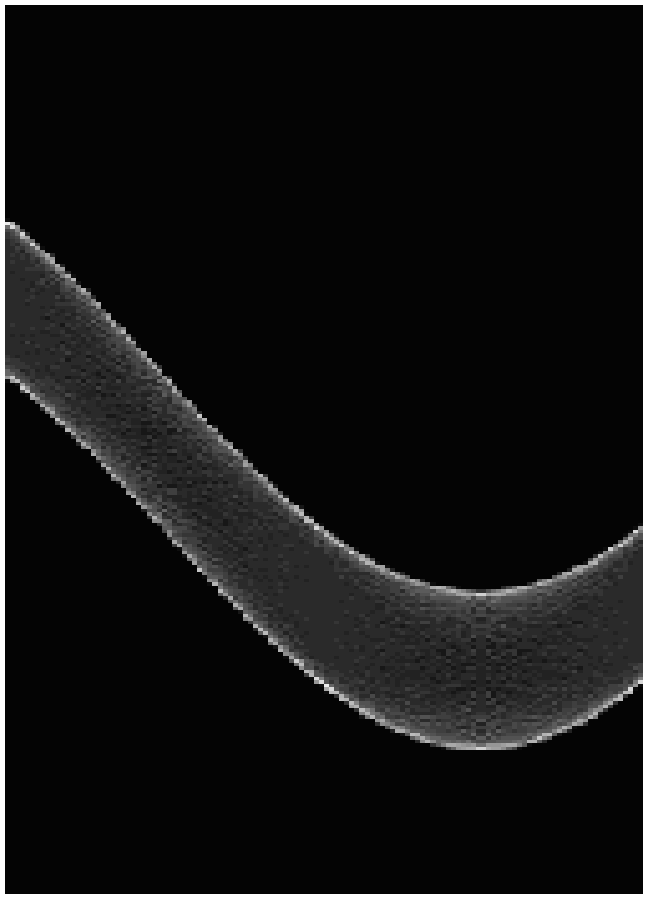
Basic illustration



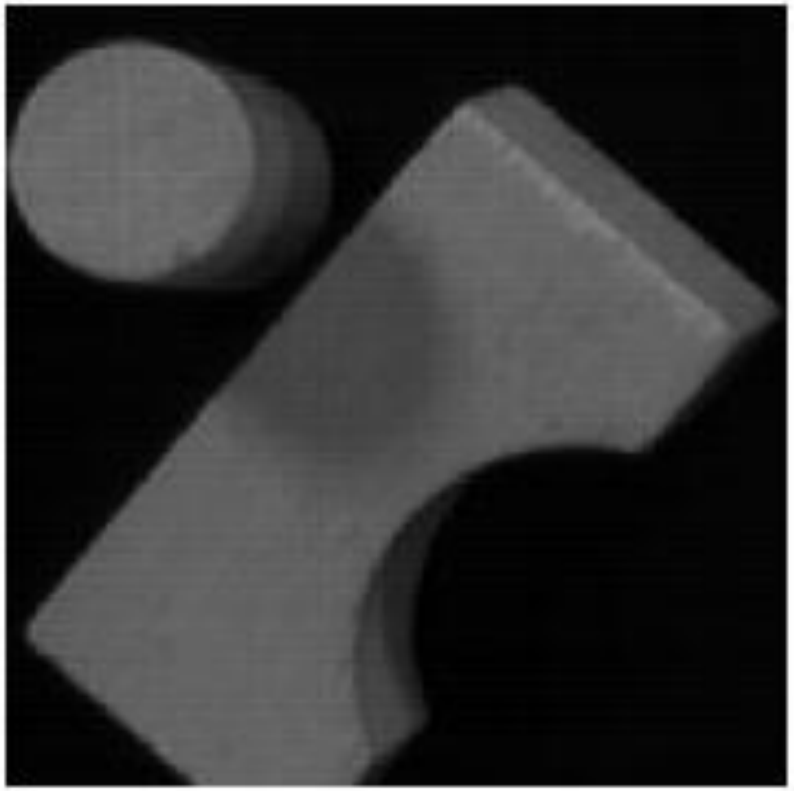
Other shapes

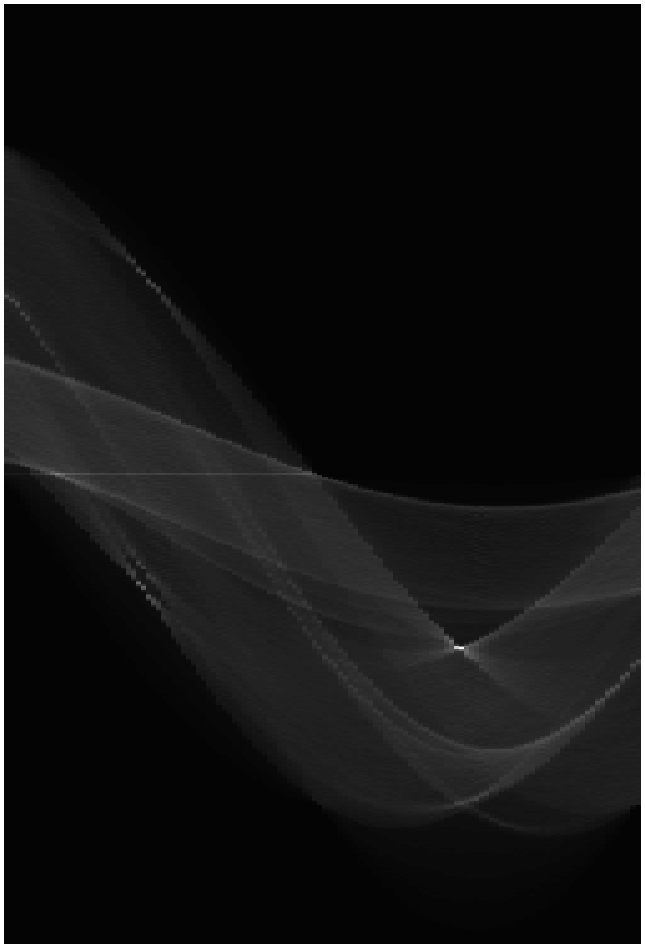
Square

Circle

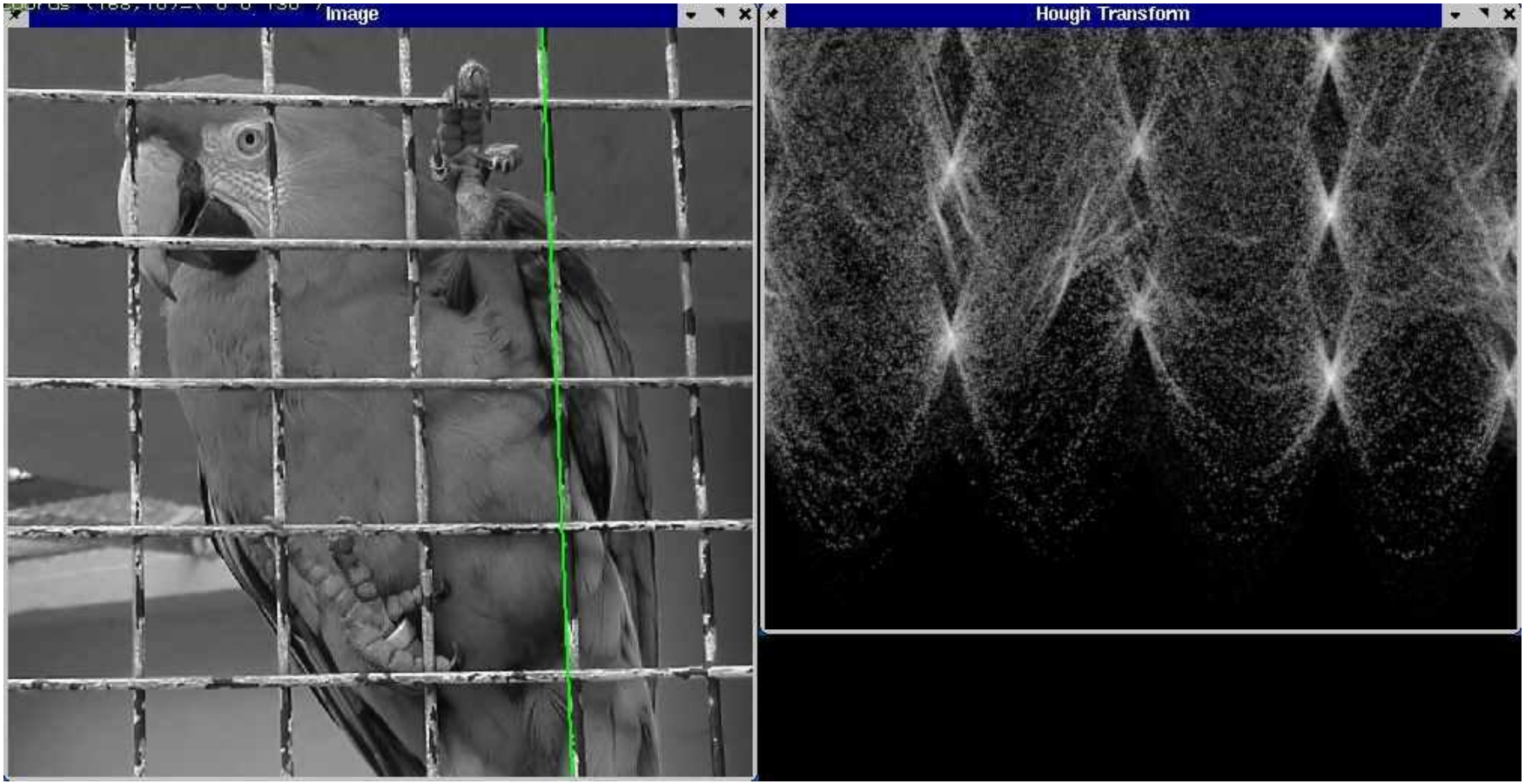


Several lines



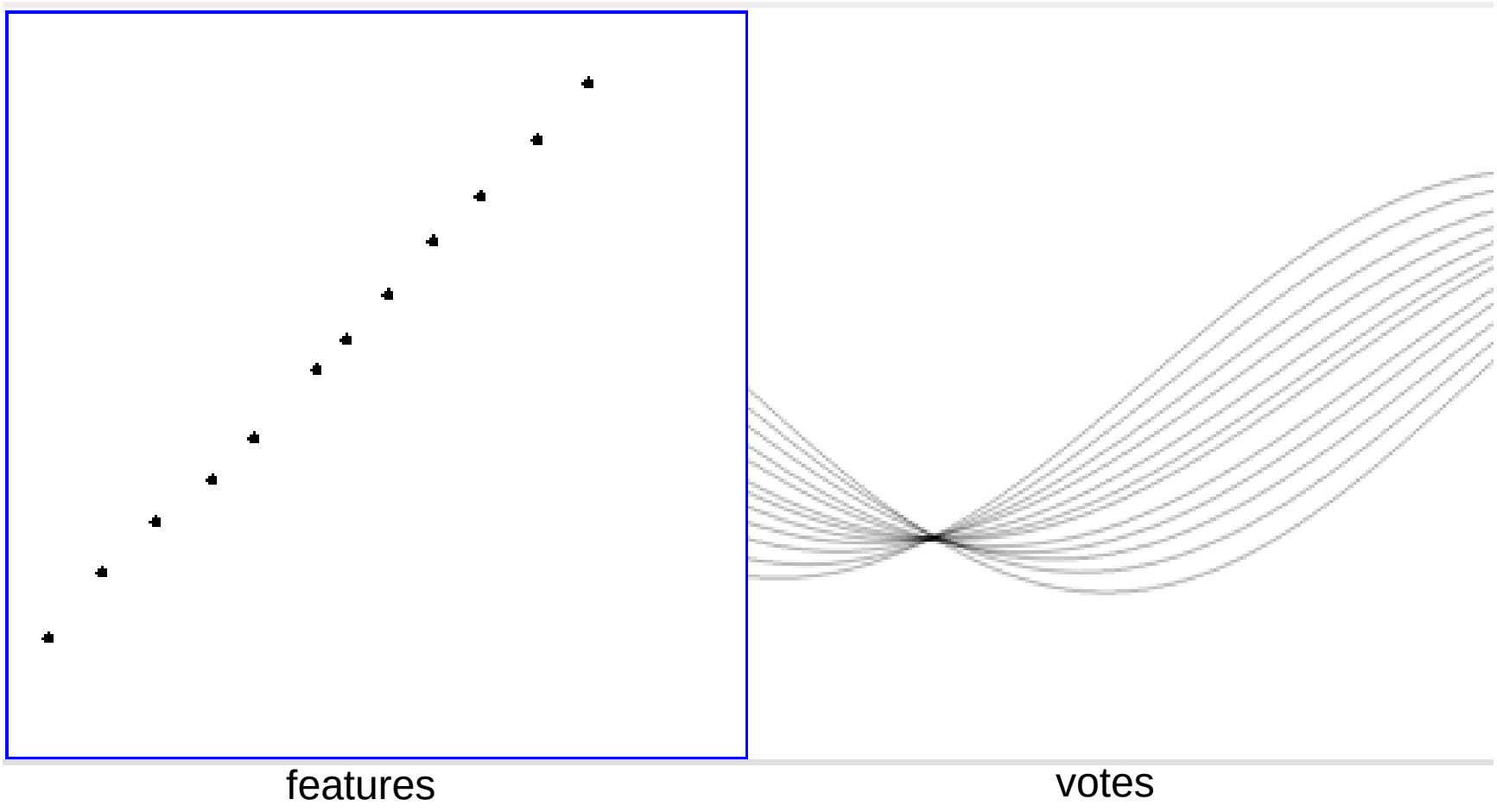


A more complicated image



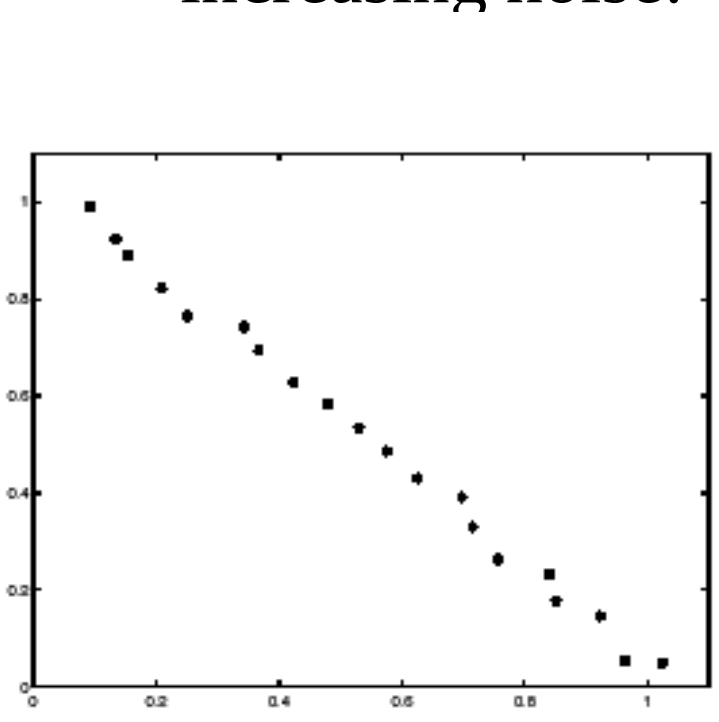
http://ostatic.com/files/images/ss_hough.jpg

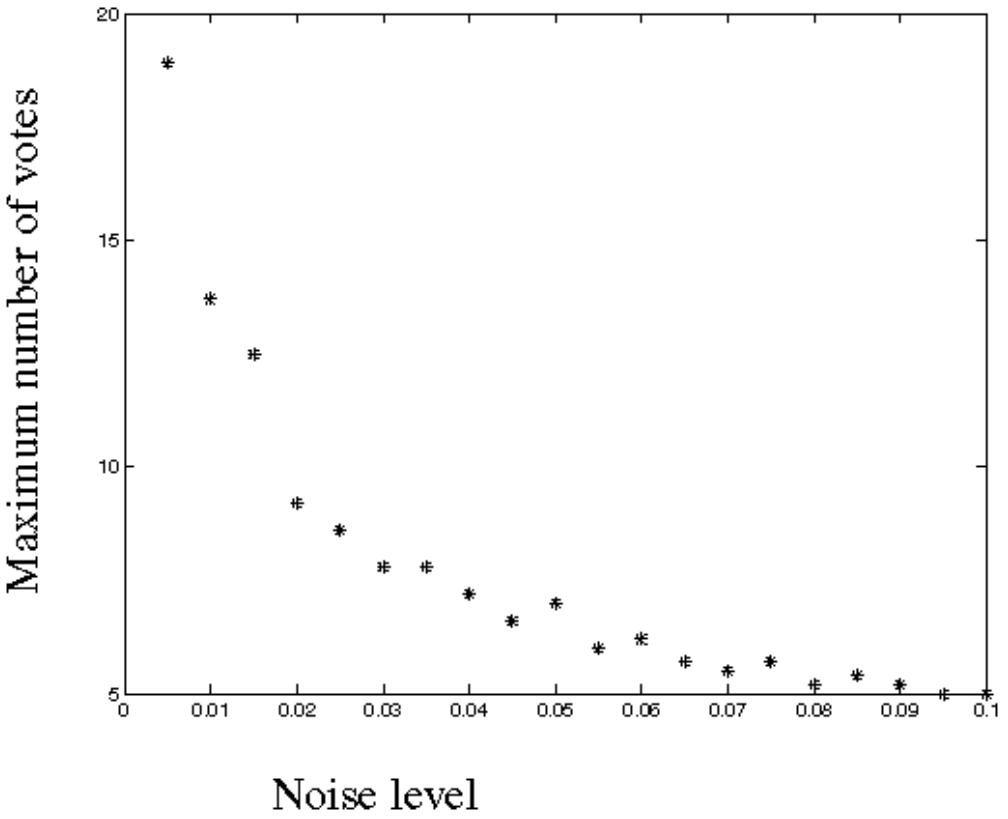
How would this change for noise?



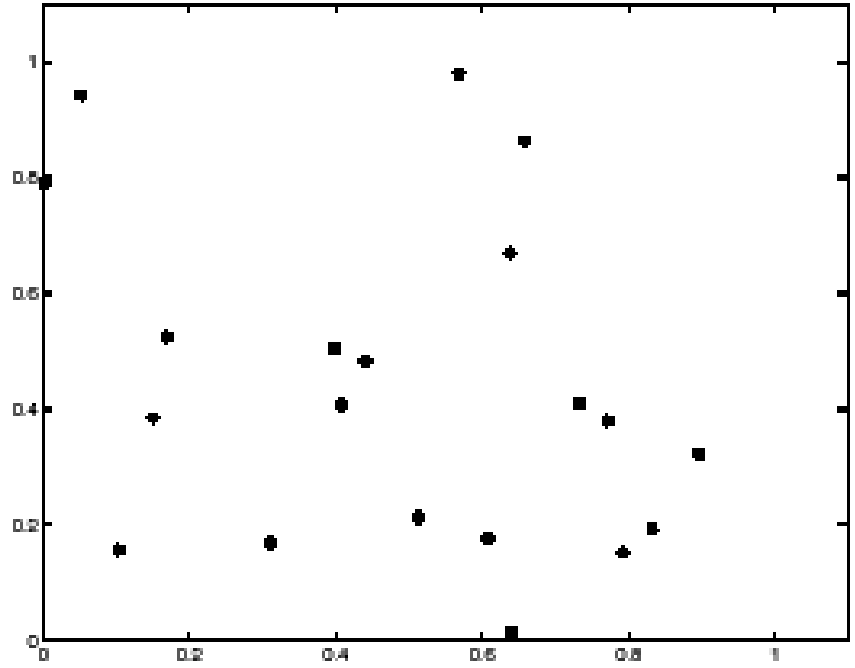
Effect of noise

- Number of votes for a line of 20 points with increasing noise:

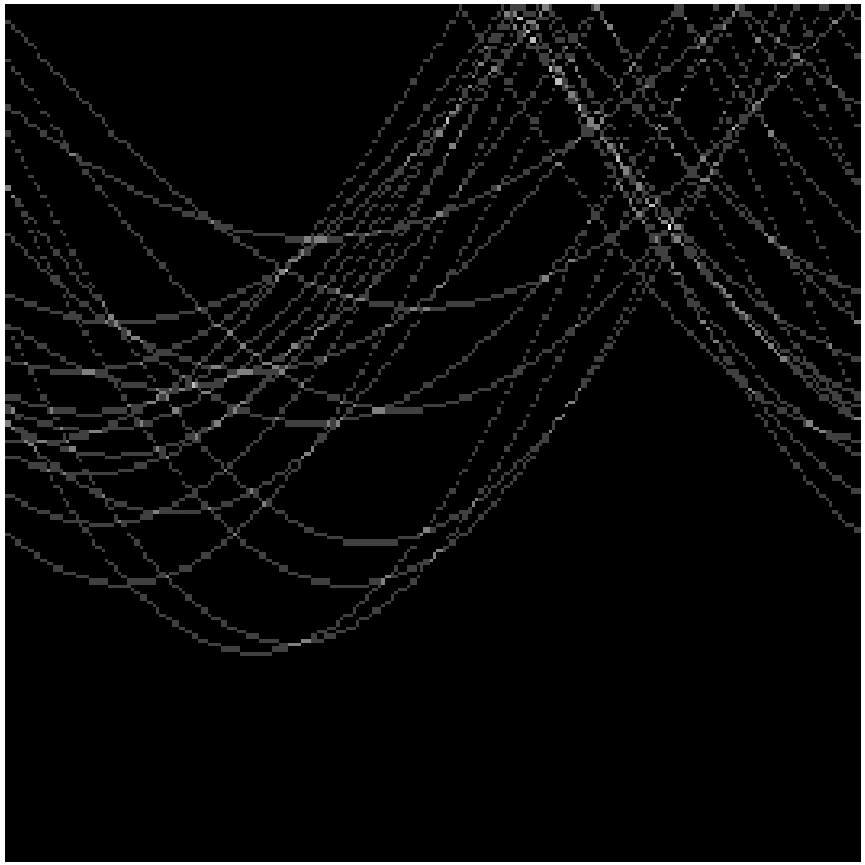




Random points



features

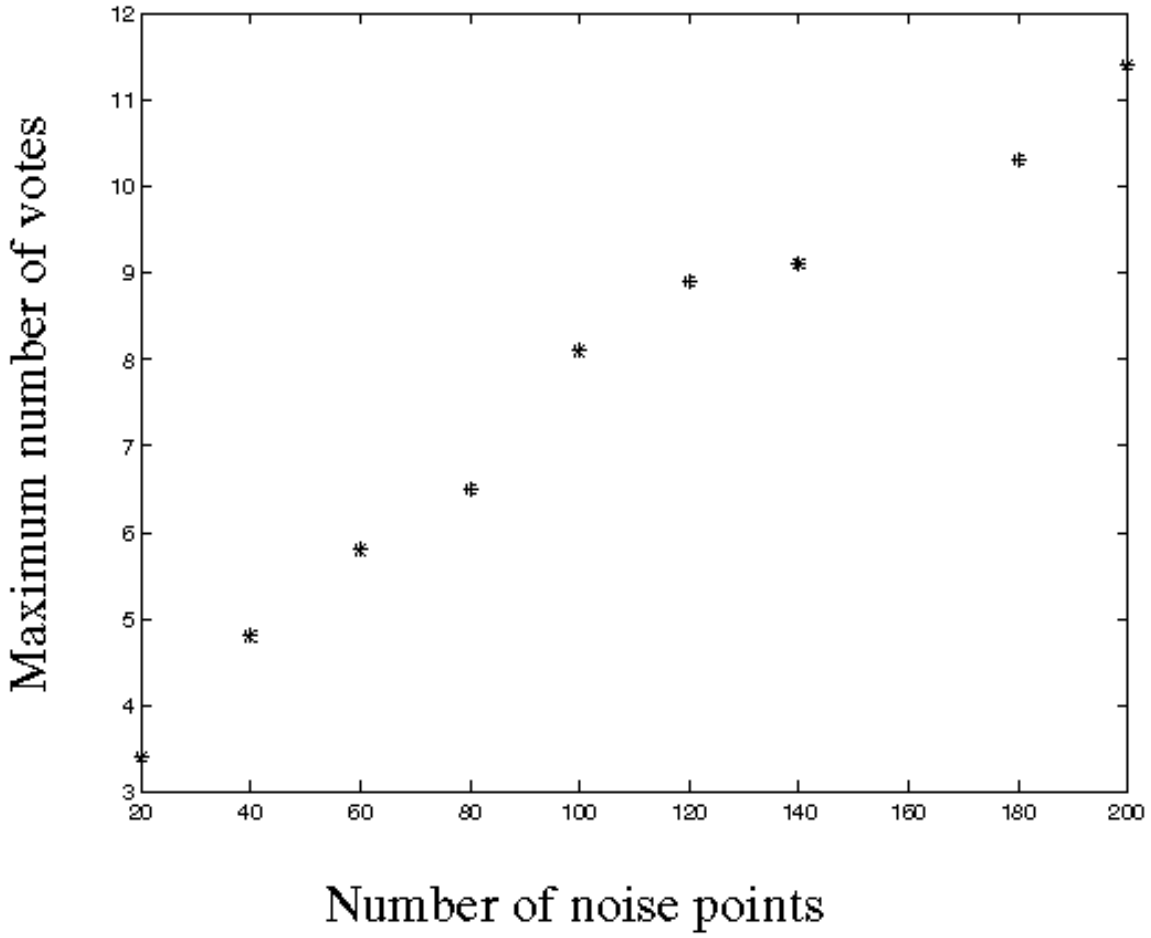


votes

- Uniform noise can lead to spurious peaks in the array

Random points

- As the level of uniform noise increases, the maximum number of votes increases too:



Dealing with noise

- Choose a good grid / discretization
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket
  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets
- Increment neighboring bins (smoothing in accumulator array)
- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude

Questions?

Summary

- Feature point detection
- Harris Corner detection
- Laplacian Blob detection
- Laplacian Scale Selection
- Sift Matching
- Edge detection and Canny
- Hough transform for line finding

Questions?