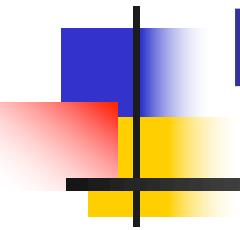


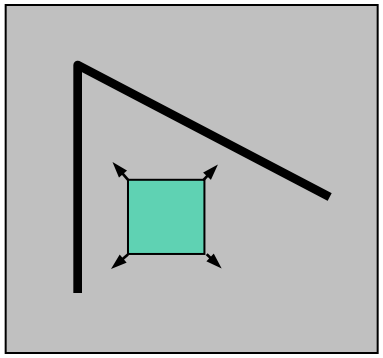
Feature detection and Model Fitting

Jayanta Mukhopadhyay
Dept. of Computer Science and Engg.

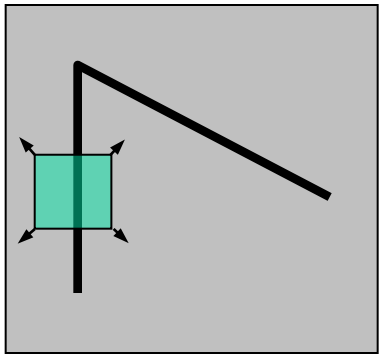
Feature detection (Chapter 4 - Szeleski)

Local measure of feature uniqueness

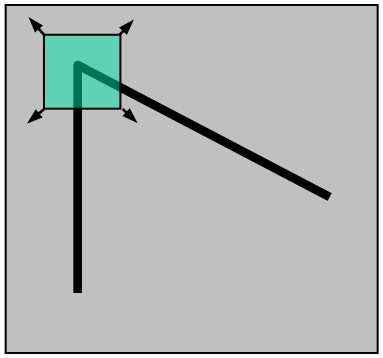
- How does the window change when you shift it?
- Shifting the window in *any direction* may cause a *big change.*



"flat" region: no change in all directions



"edge": no change along the edge direction



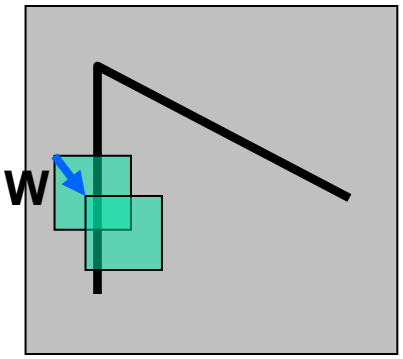
"corner": significant change in all directions

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature detection

Consider shifting the window **W** by (u,v)

- how do the pixels in **W** change?
- compare each pixel before and after by summing up the squared differences (SSD)
- this defines an SSD "error" of *E(u,v)*:



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

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Small motion assumption

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

For small u and v

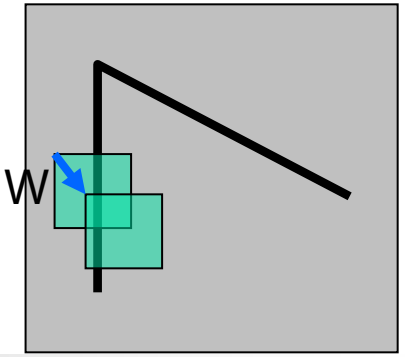
$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$$\approx I(x,y) + \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

shorthand: $I_x = \frac{\partial I}{\partial x}$

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature detection



$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v \approx I(x,y) + [I_x \; I_y]\begin{bmatrix} u \\ v \end{bmatrix}$$

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

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

$$\approx \sum_{(x,y)\in W} \left[[I_x \; I_y]\begin{bmatrix} u \\ v \end{bmatrix}\right]^2$$

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

$$\left(\begin{bmatrix} I_x & I_y \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}\right)^T \left(\begin{bmatrix} I_x & I_y \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}\right)$$

$$= \begin{bmatrix} u & v \end{bmatrix}\begin{bmatrix} I_x \\ I_y \end{bmatrix}\begin{bmatrix} I_x & I_y \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

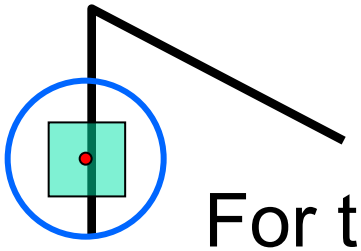
$$= \begin{bmatrix} u & v \end{bmatrix}\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

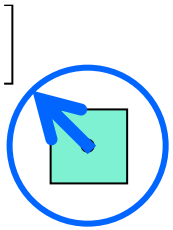
Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Feature detection

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$





For the example above

- You can move the center of the green window to anywhere on the blue unit circle
- Which directions will result in the largest and smallest E values?
- We can find these directions by looking at the eigenvectors of ***H.***

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Quick eigenvalue/eigenvector review

$$Ax = \lambda x$$

The **eigenvectors** of a matrix **A** are the vectors **x** that satisfy:

$$det(A - \lambda I) = 0 \quad det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

The scalar λ is the **eigenvalue** corresponding to **x**

- The eigenvalues are found by solving:
- In our case, ***A*** = ***H*** is a 2x2 matrix, so we have

- The solution:

$$\begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

$$\lambda_{\pm} = \frac{1}{2} \left[(h_{11} + h_{22}) \pm \sqrt{4h_{12}h_{21} + (h_{11} - h_{22})^2} \right]$$

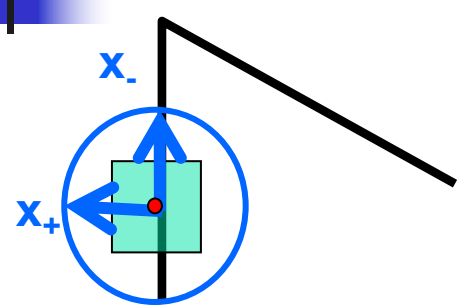
Once you know λ, you find **x** by solving

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

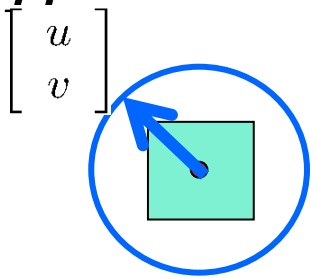
Feature detection

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

H



$$Hx_+ = \lambda_+ x_+$$
$$Hx_- = \lambda_- x_-$$



Eigenvalues and eigenvectors of H

- Define shifts with the smallest and largest change (E value)
- x_+ = direction of **largest** increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of **smallest** increase in E.
- λ_- = amount of increase in direction x_+

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature detection

How are λ_+, $\mathbf{x}_+$, λ_-, and $\mathbf{x}_-$ relevant for feature detection?

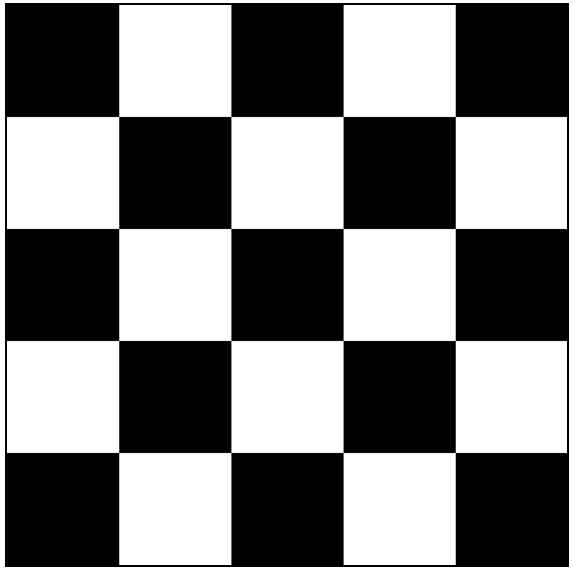
- What's our feature scoring function?

Want $E(u,v)$ to be ***large*** for small shifts in **all** directions

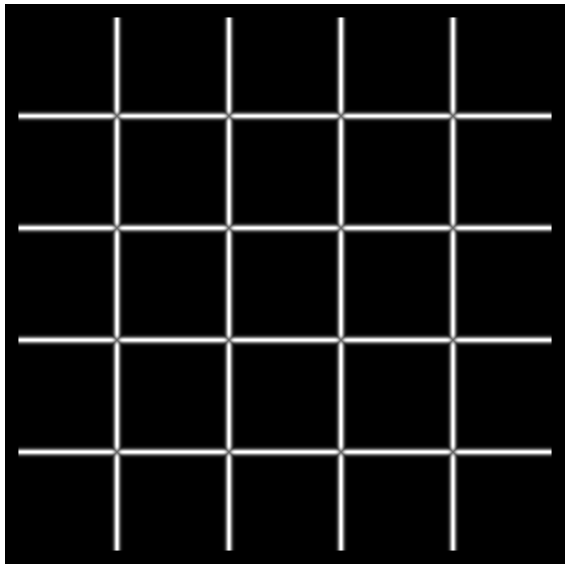
- the *minimum* of $E(u,v)$ should be large, over all unit vectors [u v].
- this minimum is given by the smaller eigenvalue (λ_-) of $\boldsymbol{H}$

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

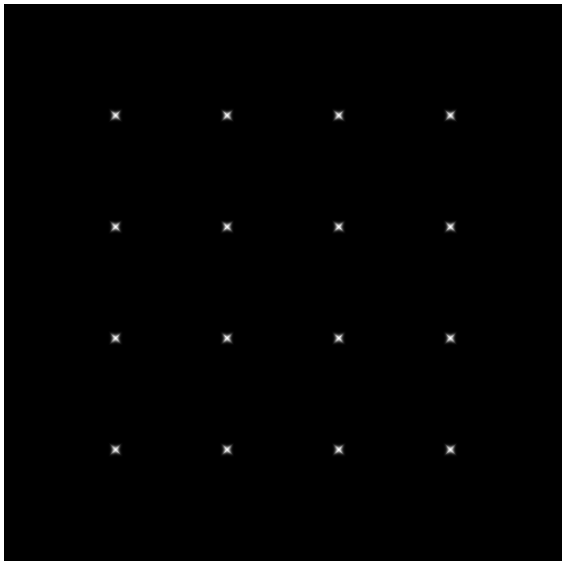
Feature detection



I



λ_+



λ_-

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature detection: summary

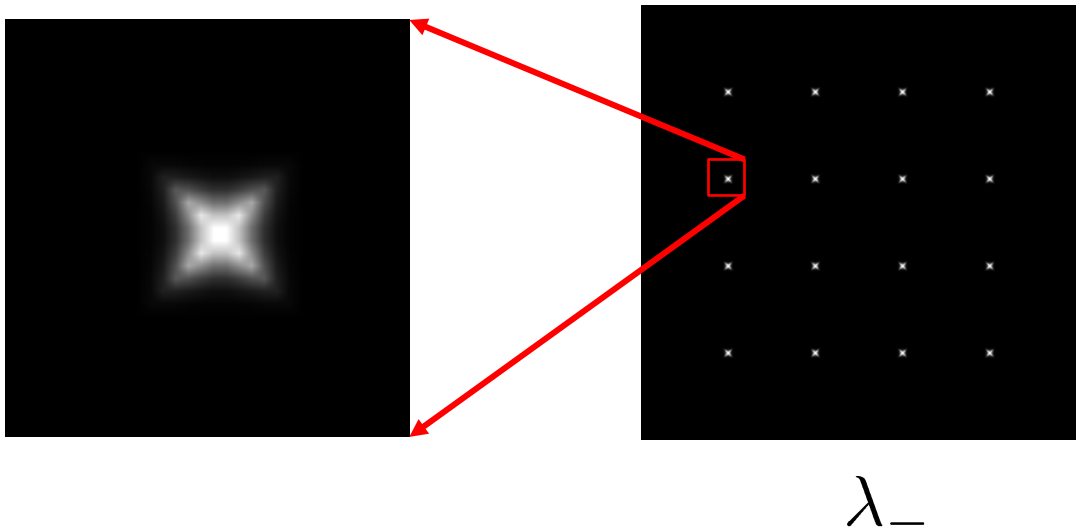
Here's what we do

- Compute the gradient at each point in the image
- Create the ***H*** matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature detection summary

- Choose those points where λ_- is a local maximum as features



Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

The Harris operator

λ_- is a variant of the "Harris operator" for feature detection.

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

$$= \frac{determinant(H)}{trace(H)}$$

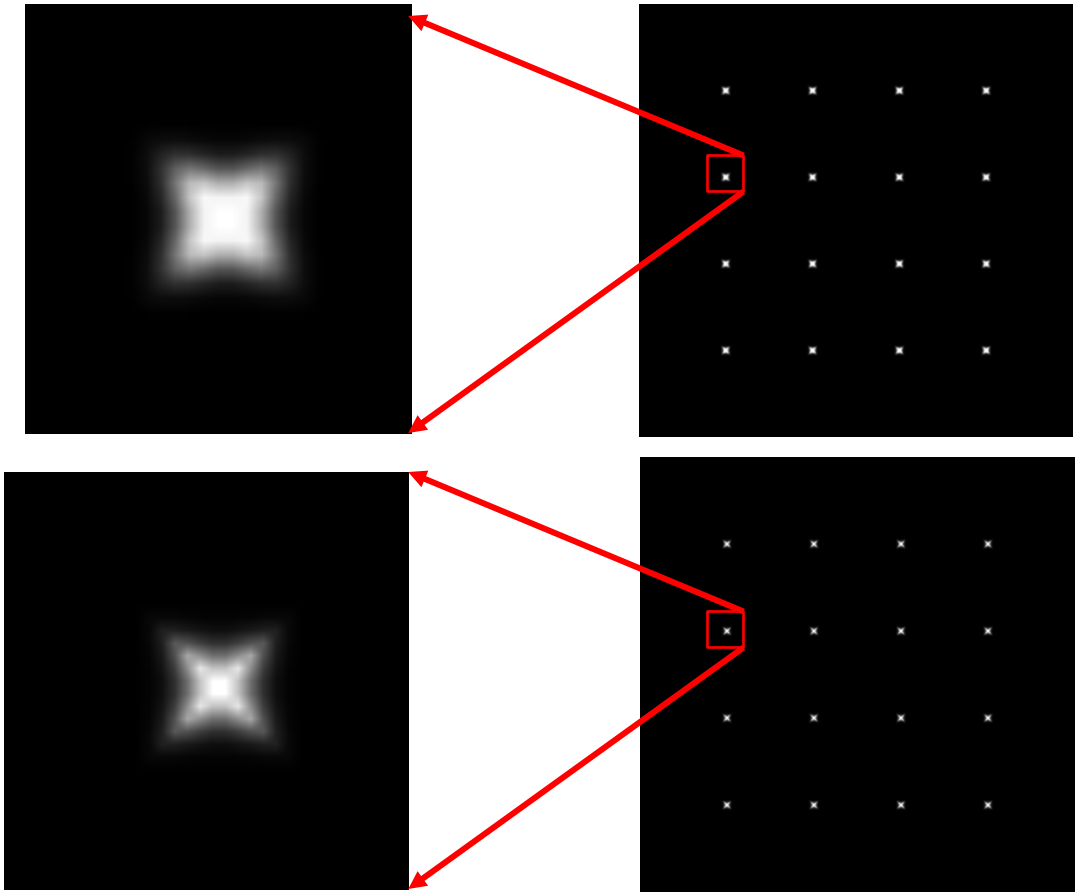
- The *trace* is the sum of the diagonals, i.e.,

  $trace(H) = h_{11} + h_{22}$

- Very similar to λ_- but less expensive (no square root)
- Called the "Harris Corner Detector" or "Harris Operator"
- Lots of other detectors, this is one of the most popular.

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

The Harris operator



Harris operator

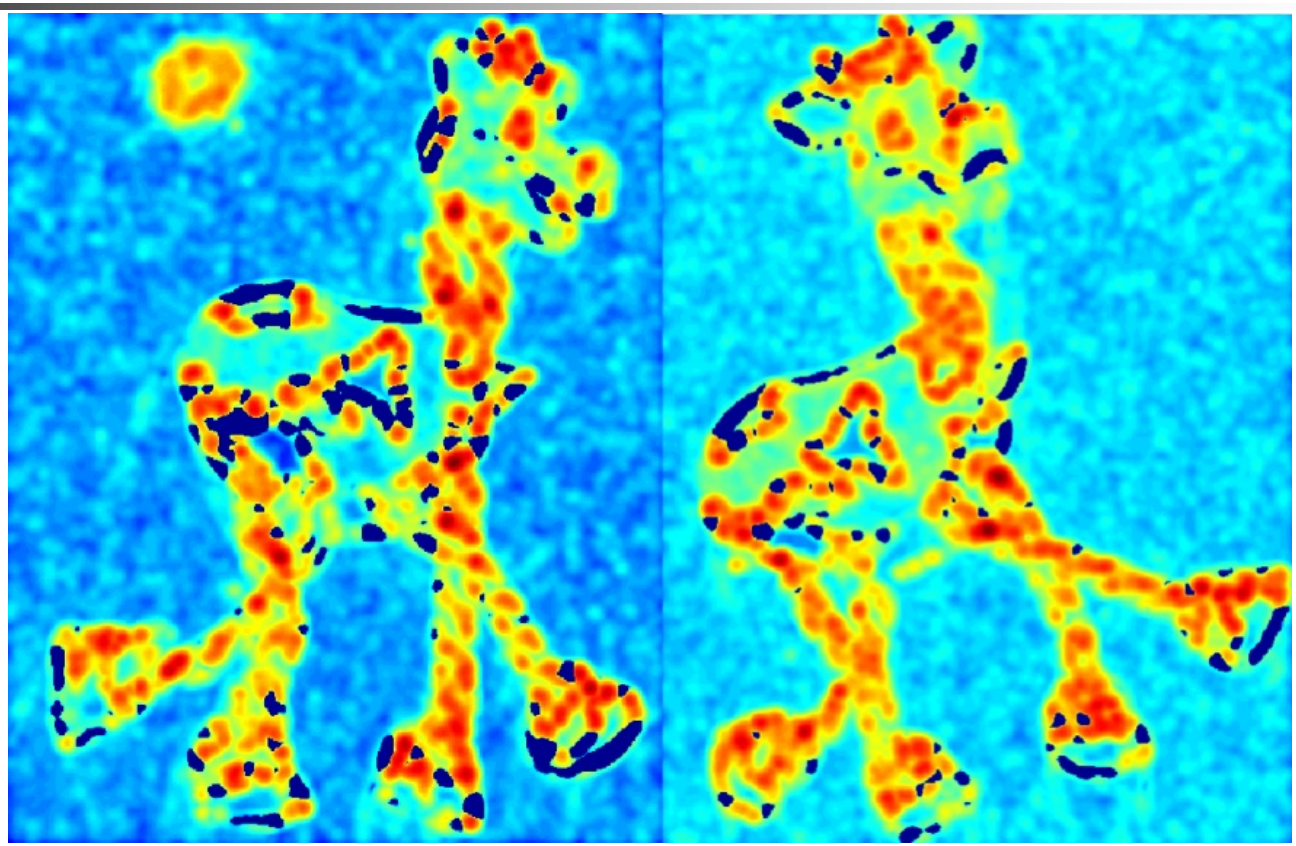
λ_-

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Harris detector example

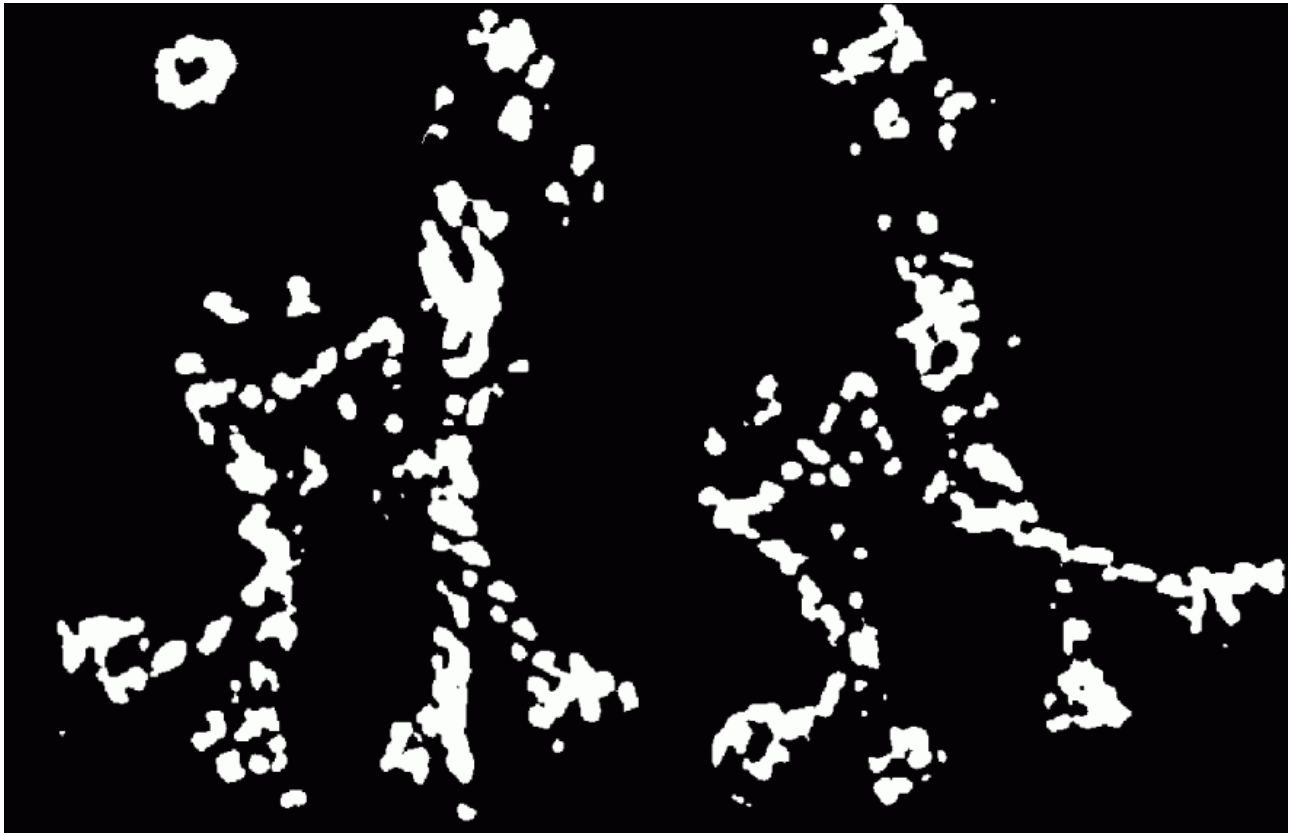
Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

f value (red high, blue low)



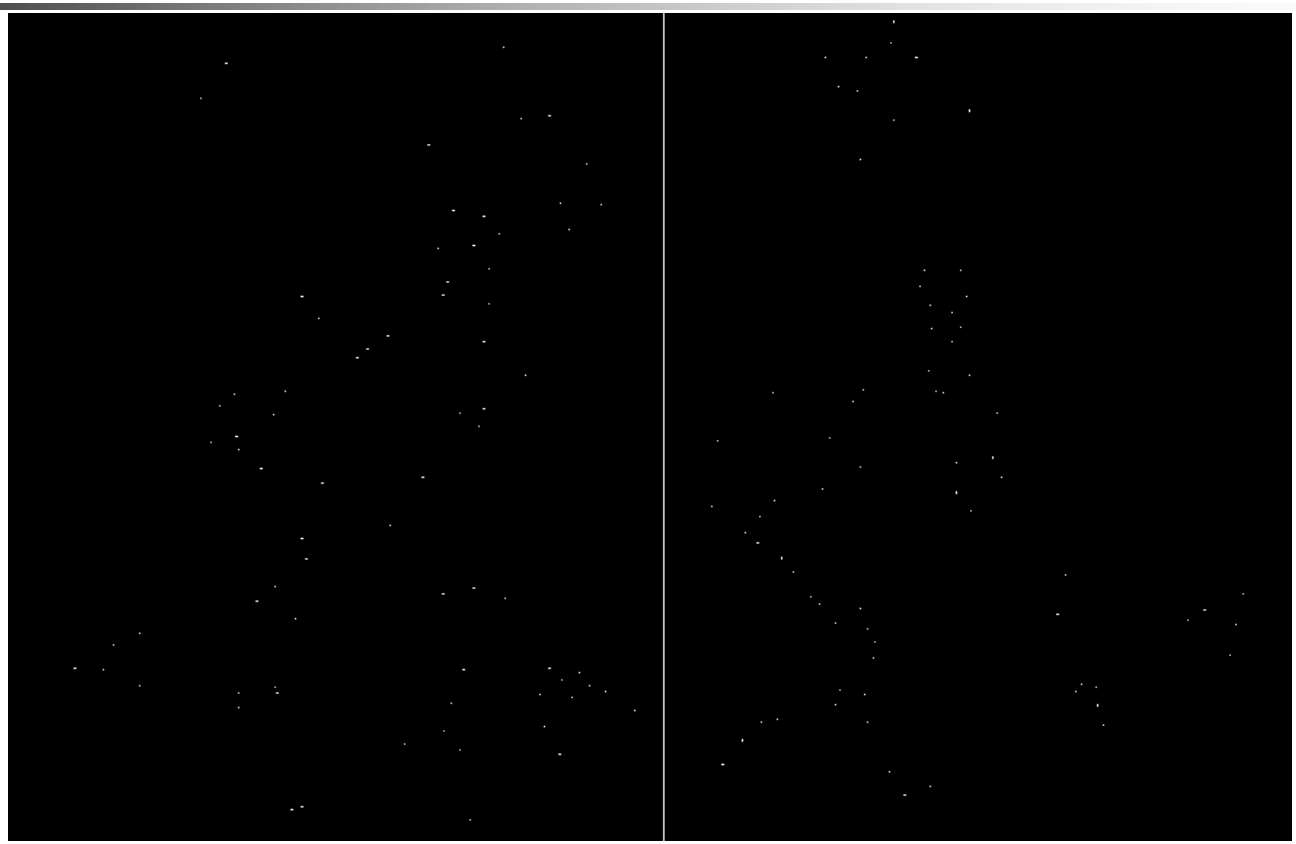
Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Threshold (f > value)



Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Find local maxima of f



Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

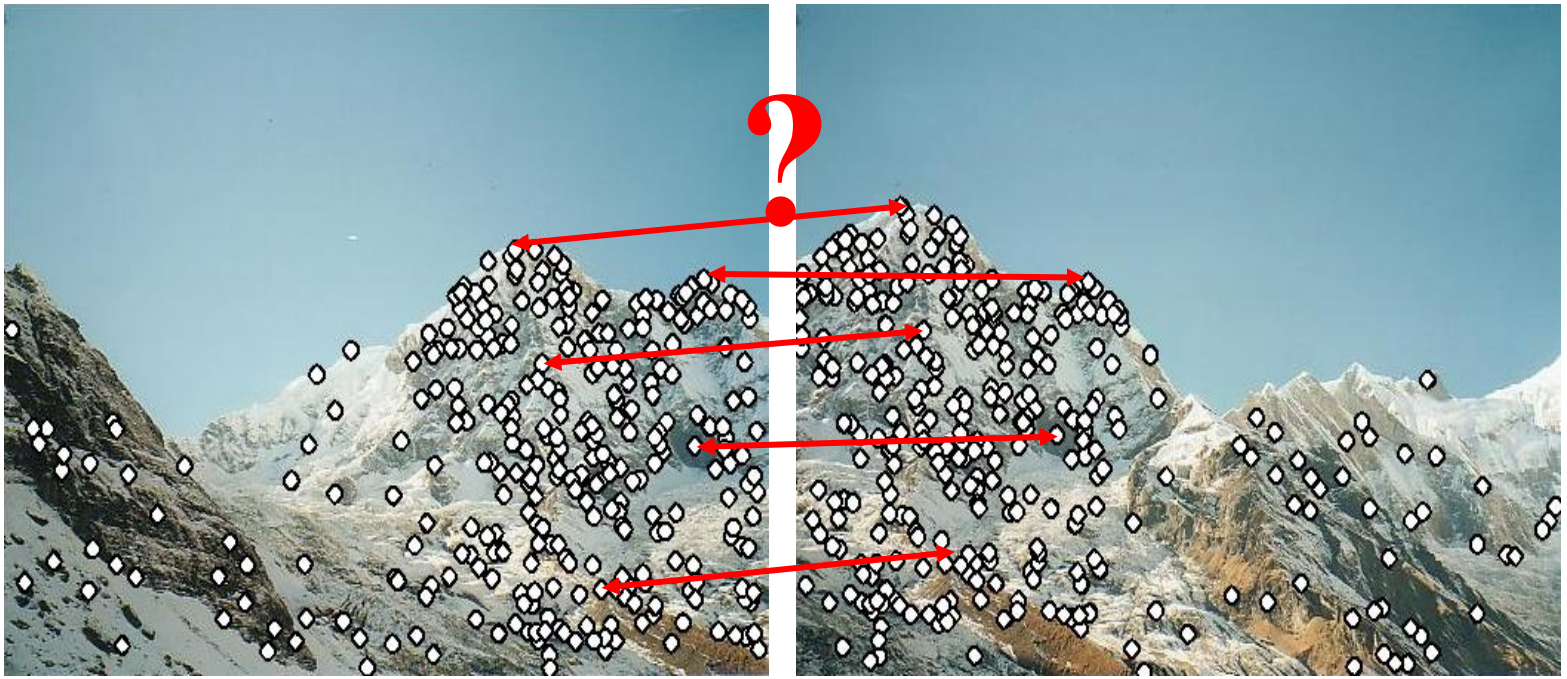
Harris features (in red)

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature Matching

We know how to detect good points

Next question: **How to match them?**

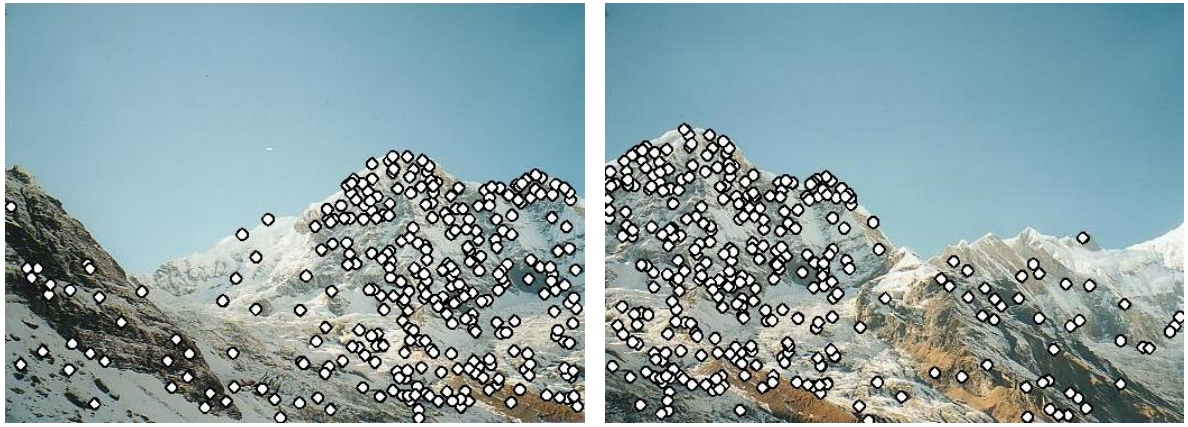


Describe feature points: Feature descriptor.

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Matching with Features

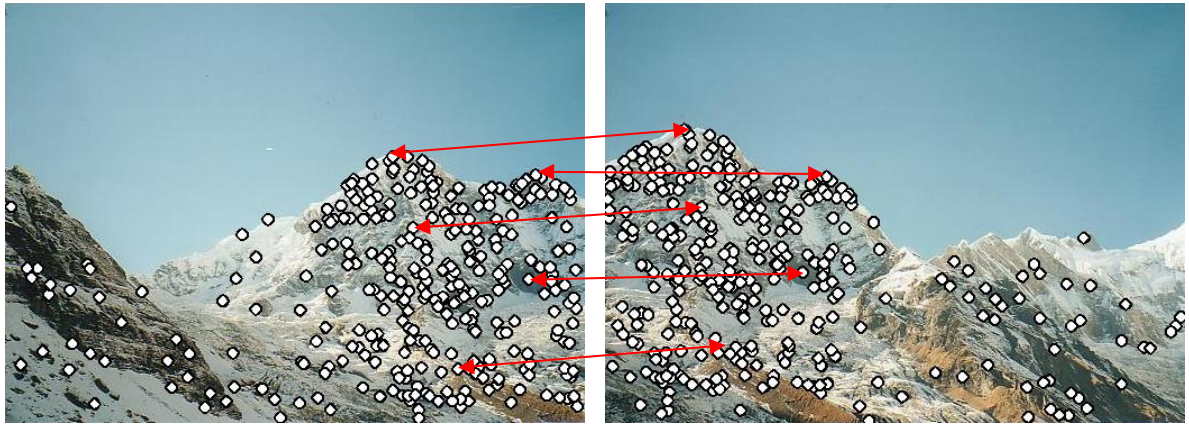
- Detect feature points in both images



Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Matching with Features

- Detect feature points in both images
- Find corresponding pairs

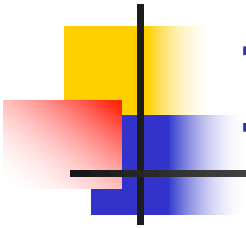


Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Matching with Features

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

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.



Invariance

Suppose you **rotate** the image by some angle

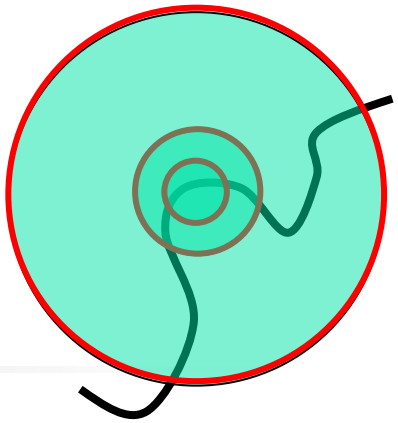
- Will you still pick up the same features?

What if you change the brightness?

Scale?

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Scale invariant detection

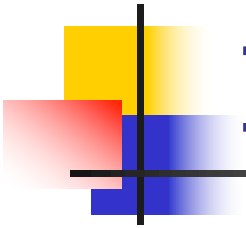


Suppose you're looking for corners

Key idea: find scale that gives local maximum of f

- f is a local maximum in both position and scale
- Common definition of f: Laplacian (or difference between two Gaussian filtered images with different s.d.'s.).

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

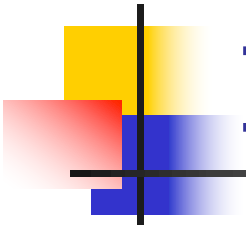


Invariance

Suppose we are comparing two images I_1 and I_2

- I_2 may be a transformed version of I_1
- What kinds of transformations are we likely to encounter in practice?

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.



Invariance

We'd like to find the same features regardless of the transformation

- This is called transformational ***invariance***
- Most feature methods are designed to be invariant to
  - Translation, 2D rotation, scale
- They can usually also handle
  - Limited 3D rotations (SIFT works up to about 60 degrees)
  - Limited affine transformations (some are fully affine invariant)
  - Limited illumination/contrast changes

How to achieve invariance

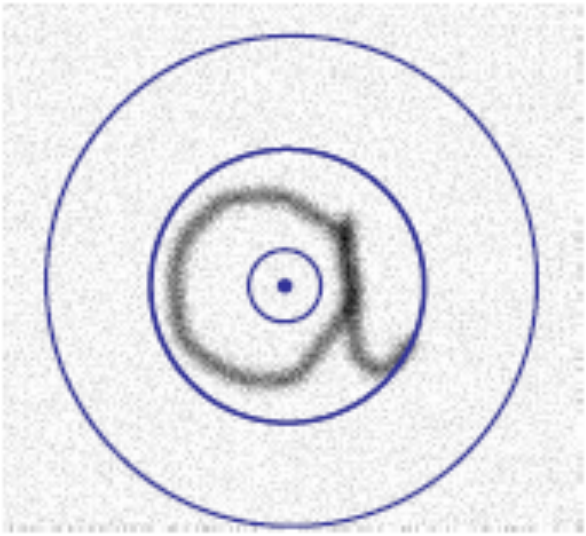
Need both of the following:

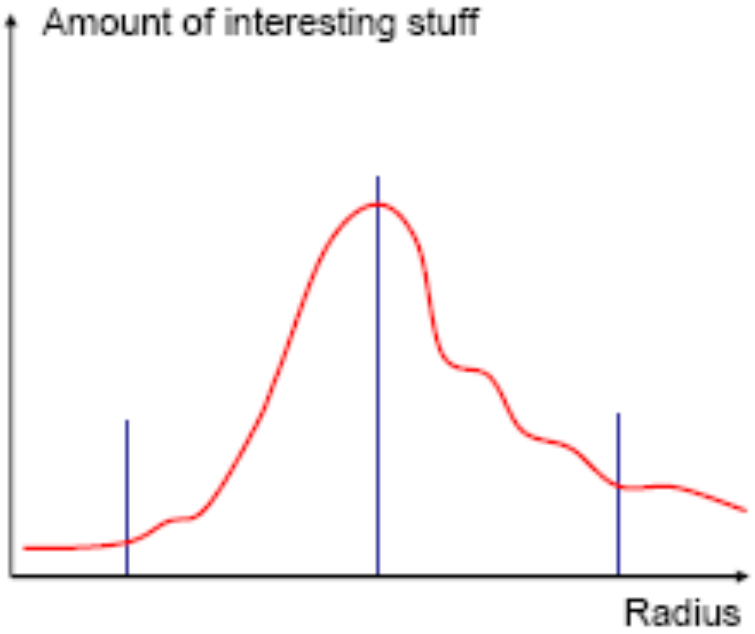
1. Make sure your detector is invariant
   - Harris is invariant to translation and rotation
   - Scale is trickier
     - common approach is to detect features at many scales using a Gaussian pyramid.
     - More sophisticated methods find "the best scale" to represent each feature (e.g., SIFT)
2. Design an invariant feature *descriptor*
   - A descriptor captures the information in a region around the detected feature point
   - The simplest descriptor: a square window of pixels

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Finding Keypoints – Scale, Location

- How do we choose scale?





Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Scale Invariant Detection

- Functions for determining scale

Kernels:

$$f = \text{Kernel} * \text{Image}$$

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

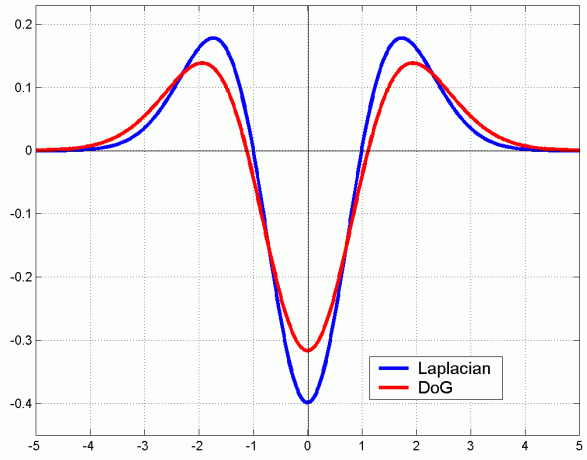
(Laplacian)

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

(Difference of Gaussians)

where Gaussian

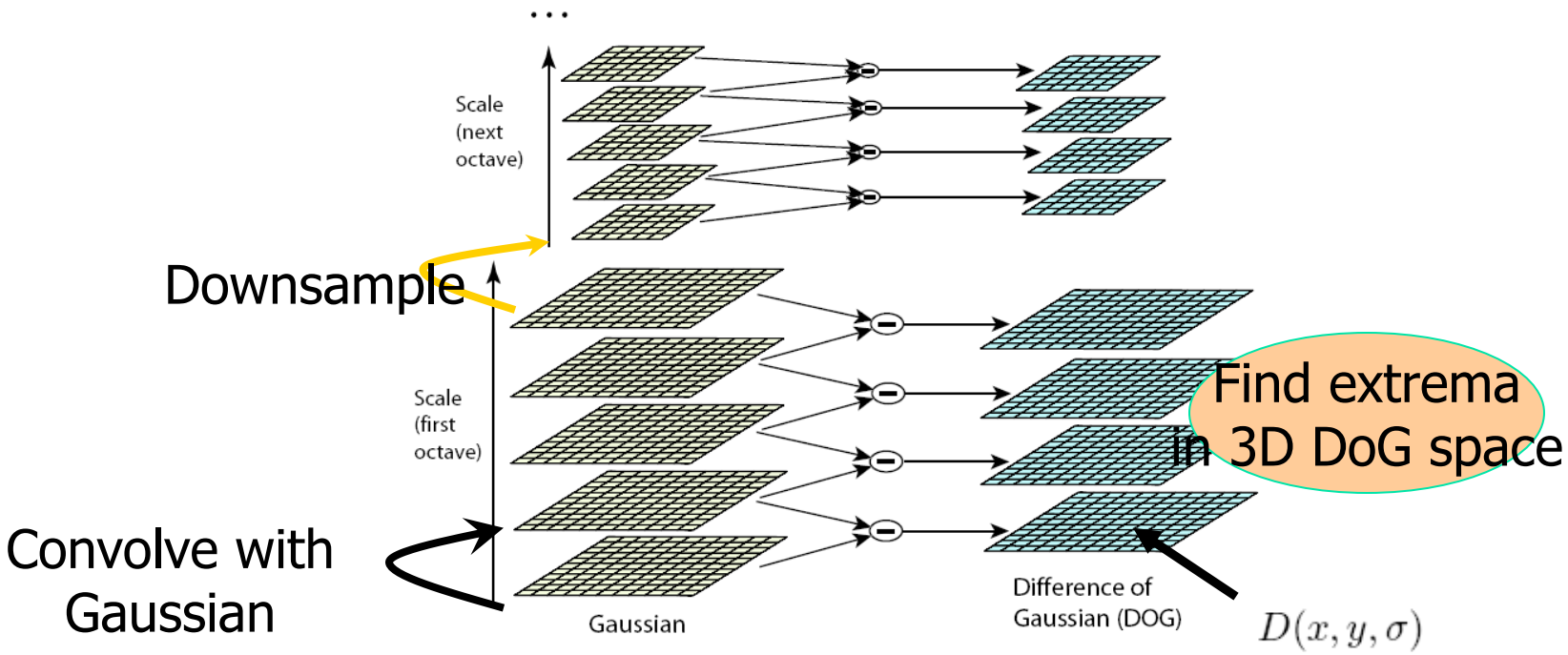
$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2 + y^2}{2\sigma^2}}$$



Note: both kernels are invariant to *scale* and *rotation*

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Finding Keypoints – Scale, Location



Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Relationship between LoG and DoG operator

$$\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G$$

$$\sigma \nabla^2 G = \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma}$$

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

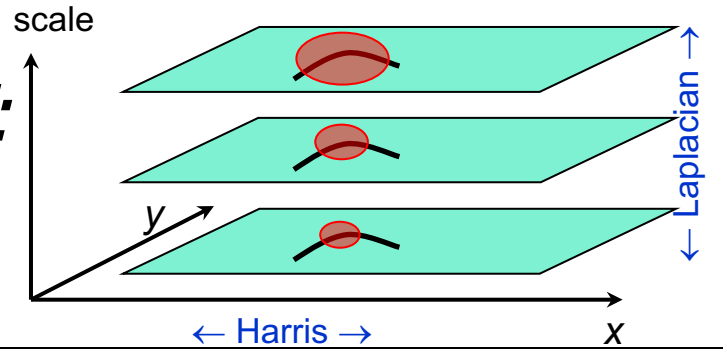
Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Scale Invariant Detectors

- Harris-Laplacian[1]

  Find local maximum of:

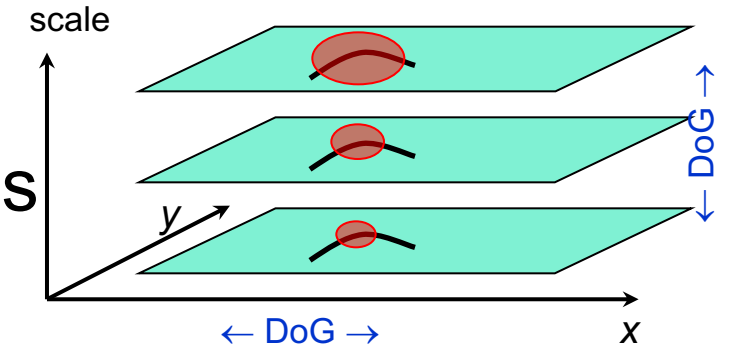
  - Harris corner
  - Laplacian in scale



- SIFT (Lowe)[2]

  Find local maximum of:

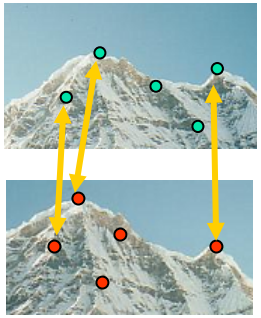
  – Difference of Gaussians in space and scale



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

[2] D.Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. IJCV 2004

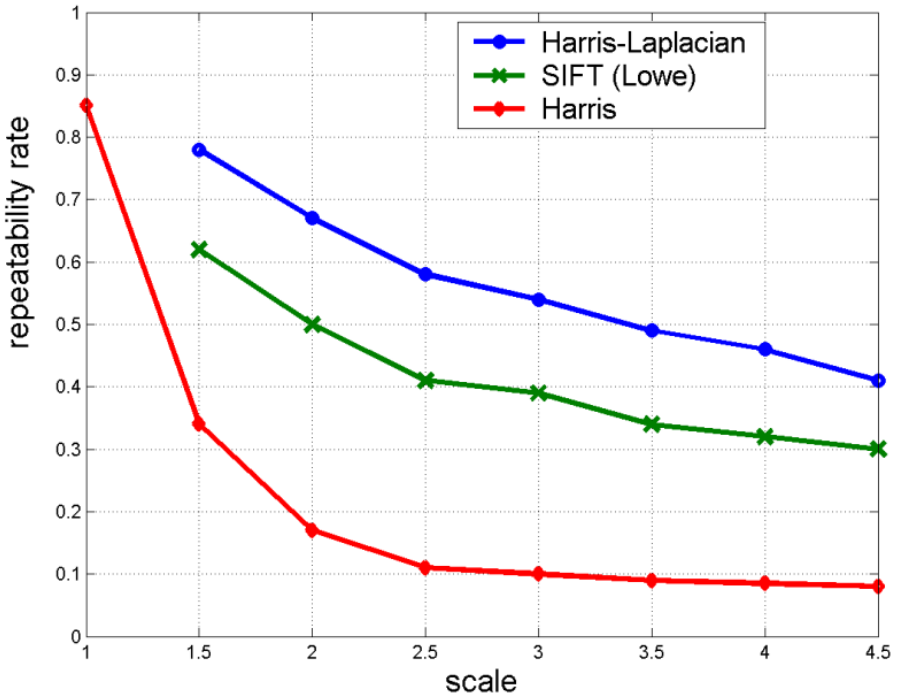
Scale Invariant Detectors



- 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

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Scale Invariant Detection: Summary

- Given: two images of the same scene with a large *scale difference* between them.
- Goal: find *the same* interest points *independently* in each image.
- Solution: search for *maxima* of suitable functions in *scale* and in *space* (over the image).

Methods:

1. Harris-Laplacian [Mikolajczyk, Schmid]: maximize Laplacian over scale, Harris' measure of corner response over the image.
2. SIFT [Lowe]: maximize Difference of Gaussians over scale and space.

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Keypoint localization

- There are still a lot of points, some of them are not good enough.
- The locations of keypoints may be not accurate.
- Eliminating edge points.

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Eliminating edge points

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

$D(.)$=DoG of the function.

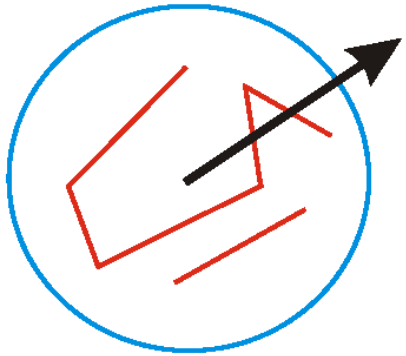
- Edge point: large principal curvature across the edge but a small one in the perpendicular direction.
- The eigenvalues of H are proportional to the principal curvatures, so two eigenvalues shouldn't differ too much.

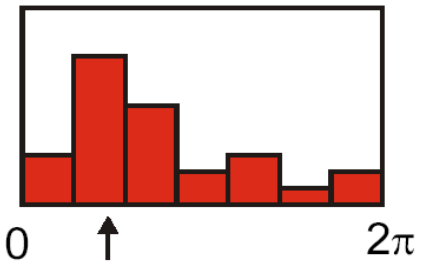
Eliminate keypoint if the ratio greater than the threshold.

$$\frac{Tr(H)^2}{Det(H)} < \frac{(r+1)^2}{r}$$ for high value of r (say 10)

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Orientation assignment





- Create histogram of local gradient directions at selected scale.
- Assign canonical orientation at peak of smoothed histogram.
- Each key specifies stable 2D coordinates (x, y, scale, orientation).

If 2 major orientations, use both.

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Keypoint localization with orientation

233x189

initial keypoints

832

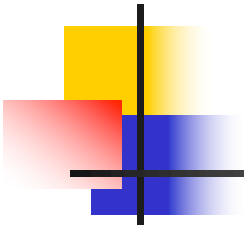
729

keypoints after gradient threshold

536

keypoints after ratio threshold

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.



Keypoint Descriptors

- At this point, each keypoint has
  - location
  - scale
  - orientation
- Next is to compute a descriptor for the local image region about each keypoint that is
  - highly distinctive
  - invariant as possible to variations such as changes in viewpoint and illumination

Scale Invariant Feature Transform (Lowe'99, ICCV)

- Take 16x16 square window (orientation corrected) around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations

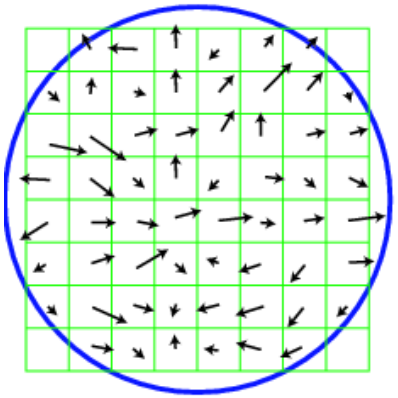
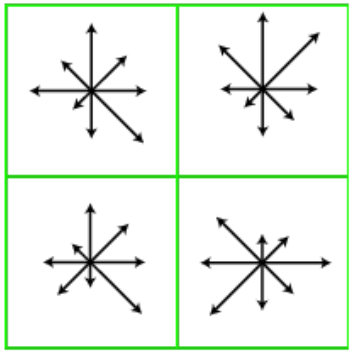
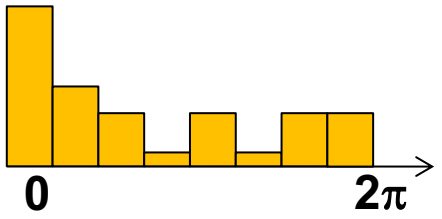


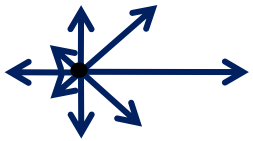
Image gradients



Keypoint descriptor



angle histogram

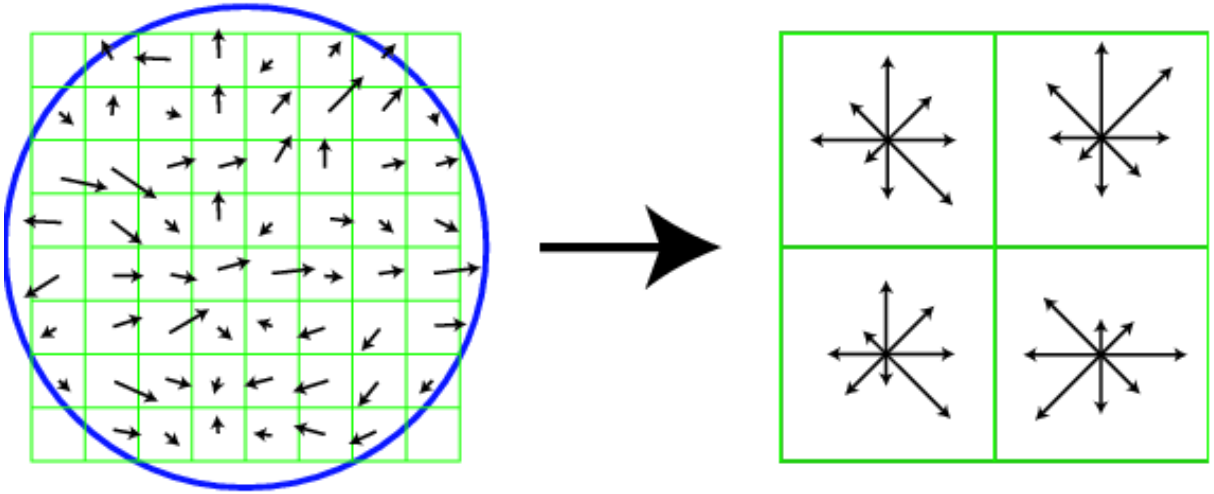


Adapted from slide by David Lowe

SIFT descriptor

Full version

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



Adapted from slide by David Lowe

Properties 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
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT

Speeded-Up Robust Features (SURF): Another descriptor

- Speeded-Up Robust Features (SURF)
  - (Bay et al. ECCV, 2006)
  - Box-type convolution filters and use of integral images to speed up the computation.
  - Use of Hessian operator for key point detection→ Local maxima of det(H).
  - Accumulate orientation corrected Haar wavelet responses.

Hessian Operator

Convolution with the Gaussian second order derivative with image.

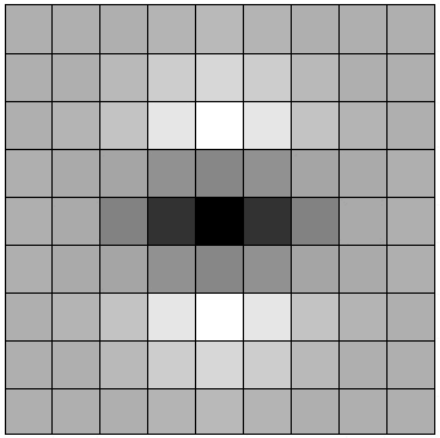
$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix},$$

Keypoint: Maximum of det(H(.)) over space and scale.

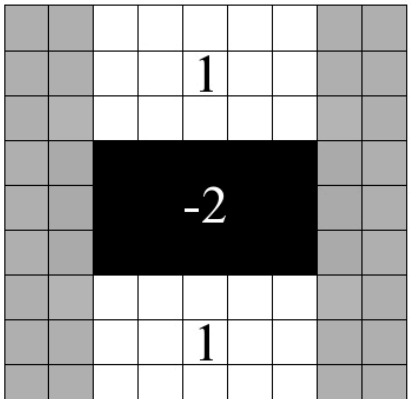
Bay et al, Speeded up robust features (SURF), CVIU, 2008

Approximation of Gaussian by Box filters

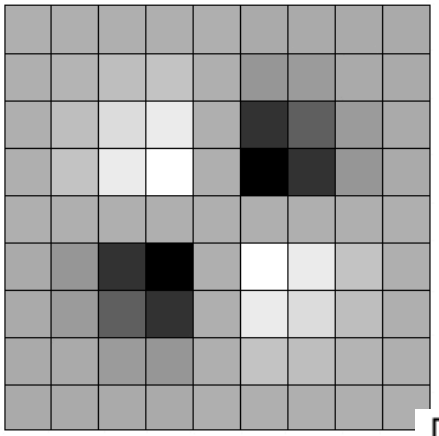
Bay et al, Speeded up robust features (SURF), CVIU, 2008



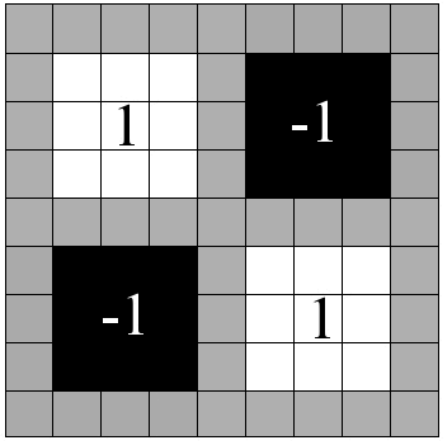
L_{yy}



9x9 Box-filters are approximation of Gaussian width 1.2.



L_{xy}

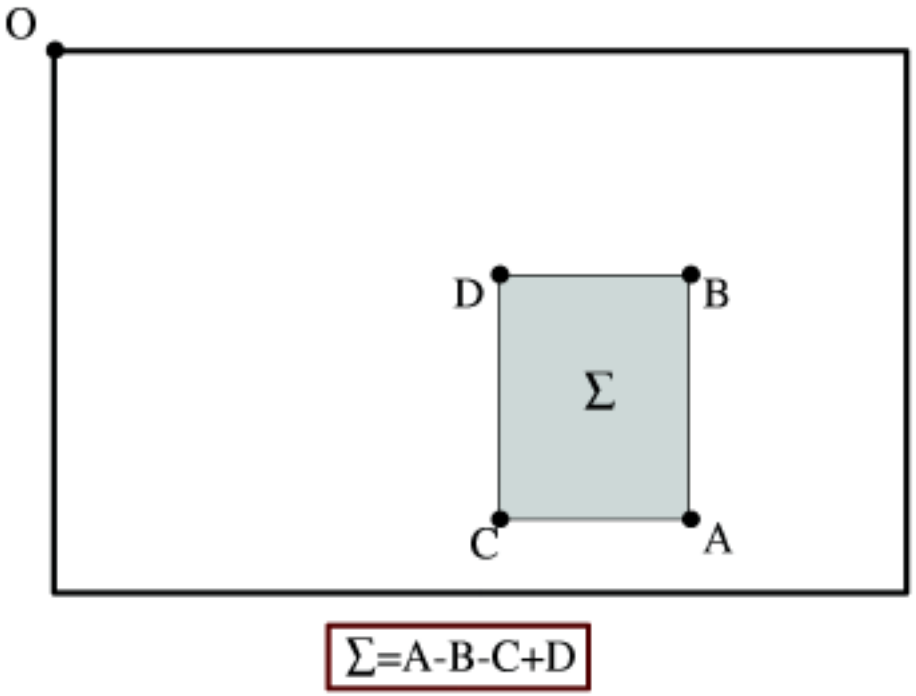


$$\det(\mathcal{H}_{\text{approx}}) = D_{xx}D_{yy} - (wD_{xy})^2.$$

w~0.9

Fast computation using the integral image

$$I_{\Sigma}(\mathbf{x}) = \sum_{i=0}^{i \le x} \sum_{j=0}^{j \le y} I(i, j)$$



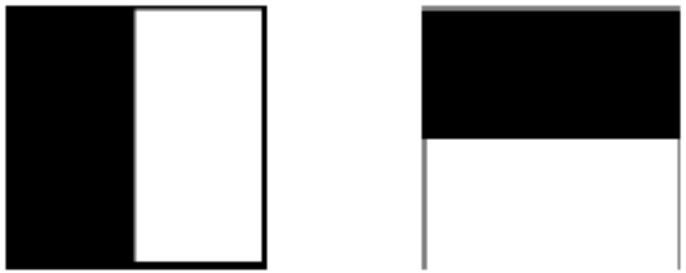
Only 3 additions and four memory access.

Bay et al, Speeded up robust features (SURF), CVIU, 2008

Bay et al, Speeded up robust features (SURF), CVIU, 2008

Haar Filter Responses

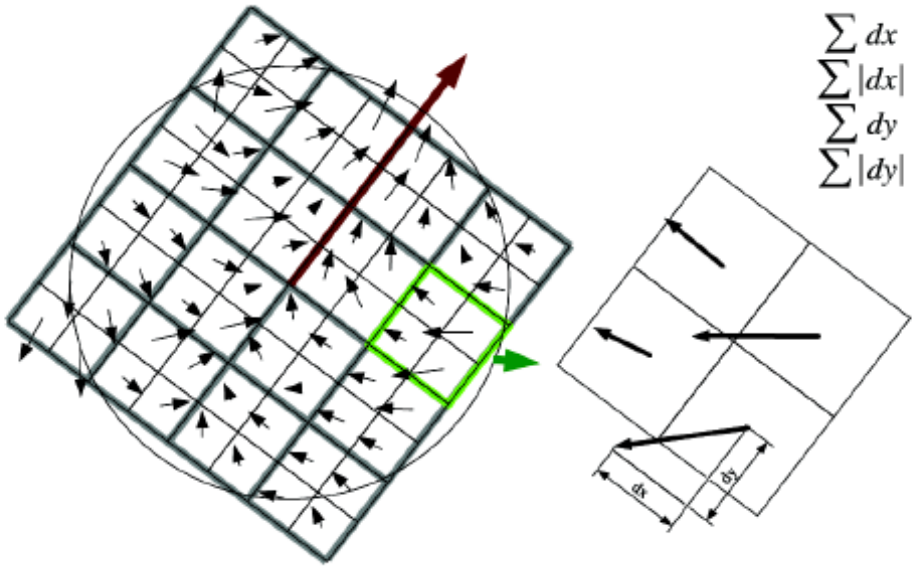
- Dominant orientation by accumulating Haar horizontal and vertical responses in a sliding window (of width 60^0) at the scale of key point.
- The longest vector provides the dominant direction.



Haar Filters
Box filter implementation.

SURF: Sum of Haar Wavelet responses

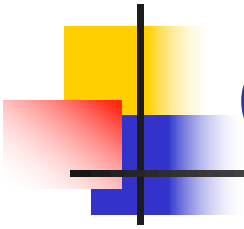
- Partitioned into 4x4 square sub-regions.
- Haar wavelet responses at regularly spaces 5x5 sample patches.
- Each sub-region has 4D vector.
- Concatenate them to 64 D vector.



Size of the window=20 x scale

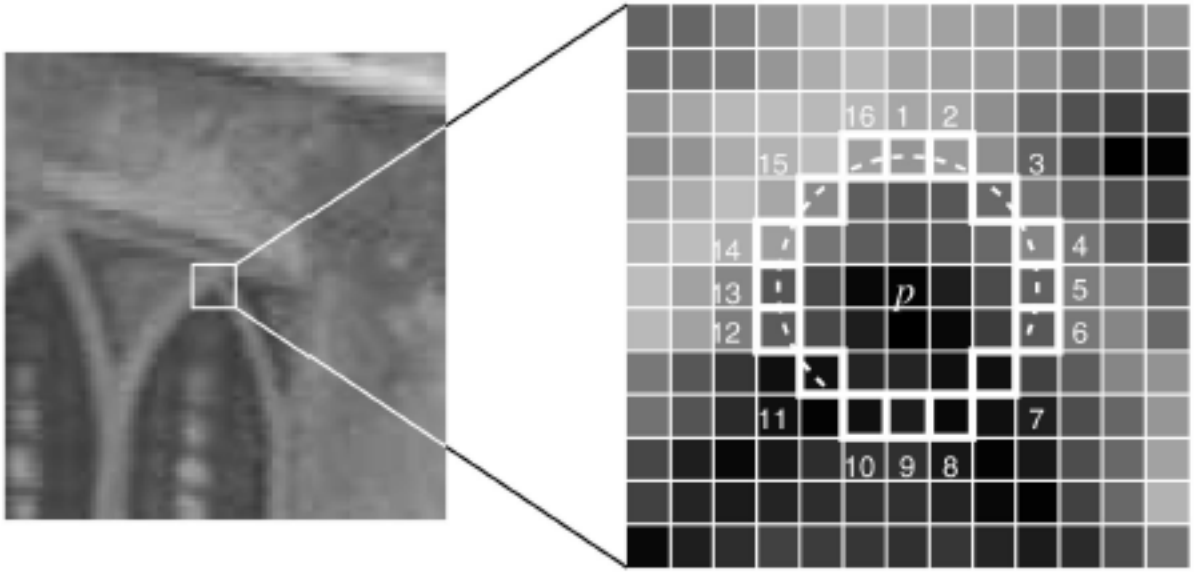
Bay et al, Speeded up robust features (SURF), CVIU, 2008

Other types of detectors and descriptors



- FAST: Features from Accelerated Segment Test (Rosten et al, PAMI, 2010).
- BRIEF: Binary Robust Independent Elementary Features (Colonder et al, ECCV, 2010).
- ORB: Oriented FAST and Rotated BRIEF (Rublee et al, ICCV, 2011)

FAST: Principle



- 12 point test: If there exists 12 consecutive points in the set of 16 points brighter than the central pixel (Rosten et al ICCV'05).
- Modified strategy: Train decision tree on the boolean conditions given labelled data.

Rosten et al, Faster and Better: A Machine Learning Approach to Corner Detection, PAMI, 2010

FAST: Partitioning of points on the circle and Training a DT

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \leq I_p - t & \text{(darker)} \\ s, & I_p - t < I_{p\to x} < I_p + t & \text{(similar)} \\ b, & I_p + t \leq I_{p\to x} & \text{(brighter)} \end{cases}$$

- Classification of points in three classes and create three partitions for a point.
- Train a decision tree.

Rosten et al, Faster and Better: A Machine Learning Approach to Corner Detection, PAMI, 2010

BRIEF: Principle

- Generate randomly a set of n_d pairs of locations $(\mathbf{x_i}, \mathbf{y_i})$ in a patch centered around a point.
- Perform the following boolean test.

$$\tau(\mathbf{p}; \mathbf{x}, \mathbf{y}) := \begin{cases} 1 & \text{if } \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & \text{otherwise} \end{cases} ,$$

BRIEF descriptor: n_d *Dimensional binary string*

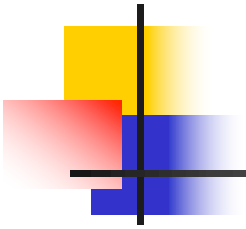
128, 256, 512,

$$f_{n_d}(\mathbf{p}) := \sum_{1 \le i \le n_d} 2^{i-1} \tau(\mathbf{p}; \mathbf{x}_i, \mathbf{y}_i) .$$

Colonder et al, BRIEF: Binary Robust Independent Elementary Features, ECCV, 2010

ORB: Principle

- FAST does not operate across scale.
- Apply FAST detector on pyramid of smoothened images.
- Orientation by intensity centroid: The vector from the center of the patch and to a centroid considering the intensity distribution (intensity weighted center of patch).
- Rotate the patch by the angle and compute BRIEF called steered BRIEF.

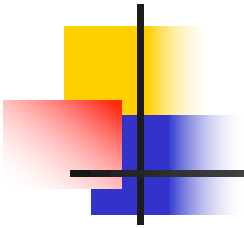


Region descriptors

- Patch descriptors (HOG)
- Image / Sub-Image descriptors (BoVW)

Patch Descriptor: Histogram of Gradients (HoG) (N. Dalal and B. Triggs CVPR 2005)

- Compute centered horizontal and vertical gradients with no smoothing.
- Compute gradient orientation and magnitudes,
- For color image, pick the color channel with the highest gradient magnitude for each pixel.
- For a 64x128 image, divide the image into 16x16 blocks of 50% overlap. →7x15=105 blocks in total



Histogram of Gradients (HoG)

- Each block: 2x2 cells with size 8x8.
- Quantize the gradient orientation into 9 bins.
- The vote is the gradient magnitude.
- Interpolate votes between neighboring bin center.
- The vote can also be weighted with Gaussian to down-weight the pixels near the edges of the block.
- Concatenate histograms→Feature dimension: 105x4x9 = 3,780

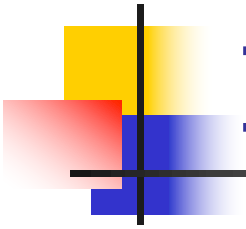
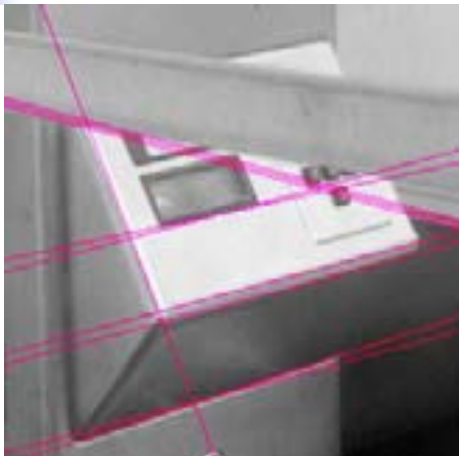


Image / Object Descriptor

- Bag of visual words (Sivic et. al., ICCV'05)
  - Compute key-point based feature descriptors.
  - Quantize them (clustering) to form a finite set of representative descriptors (visual words).
  - Represent by a histogram of visual words.

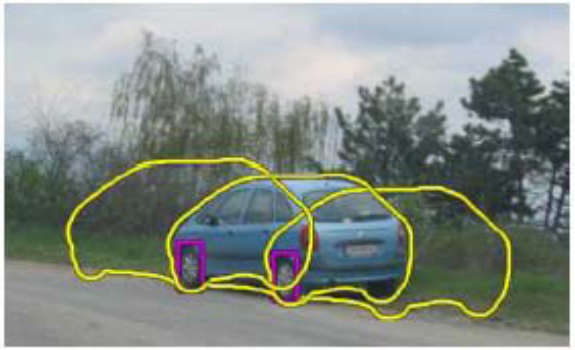
Model Fitting

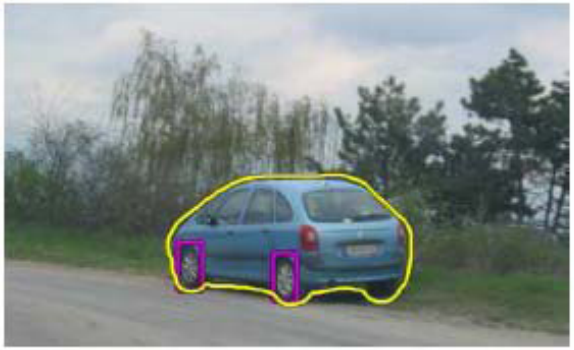
- Choose a parametric model to represent a set of features



simple model: lines

simple model: circles





complicated model: car

Source: K. Grauman

Model Fitting: Issues

Case study: Line detection

- **Noise** in the measured feature locations
- **Extraneous data:** clutter (outliers), multiple lines
- **Missing data:** occlusions

Slide: S. Lazebnik

Fitting: Issues

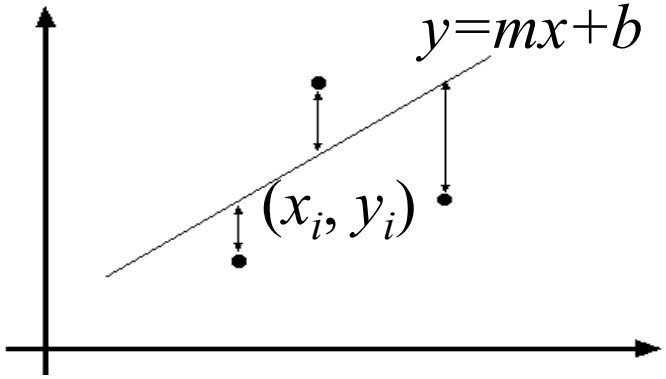
- If we know which points belong to the line, how do we find the "optimal" line parameters?
  - Least squares
- What if there are outliers?
  - Robust fitting, RANSAC
- What if there are many lines?
  - Voting methods: RANSAC, Hough transform
- What if we're not even sure it's a line?
  - Model selection

Slide: S. Lazebnik

Least squares line fitting

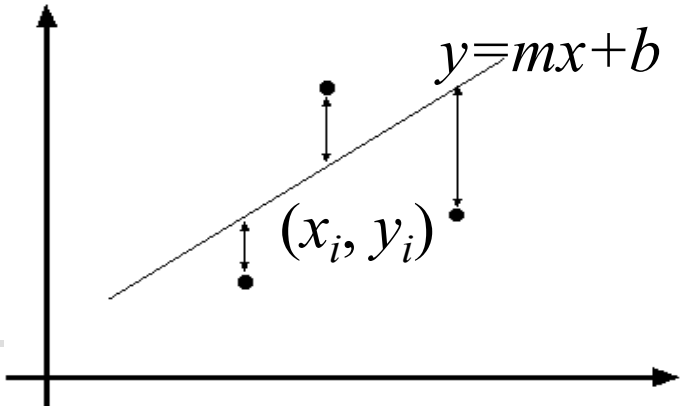
$$E = \sum_{i=1}^{n} (y_i - mx_i - b)^2$$

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m\, x_i + b$
- Find (m, b) to minimize



Slide: S. Lazebnik

Least squares line fitting



- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m x_i + b$
- Find (m, b) to minimize

$$E = \sum_{i=1}^{n} (y_i - m x_i - b)^2$$

$$E = \sum_{i=1}^{n} \left(y_i - \begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} \right)^2 = \left\| \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} \right\|^2 = \|Y - XB\|^2$$

$$= (Y - XB)^T (Y - XB) = Y^T Y - 2(XB)^T Y + (XB)^T (XB)$$

$$\frac{dE}{dB} = 2X^T XB - 2X^T Y = 0$$

$$X^T XB = X^T Y$$

Normal equations: least squares solution to $XB = Y$

Slide: S. Lazebnik

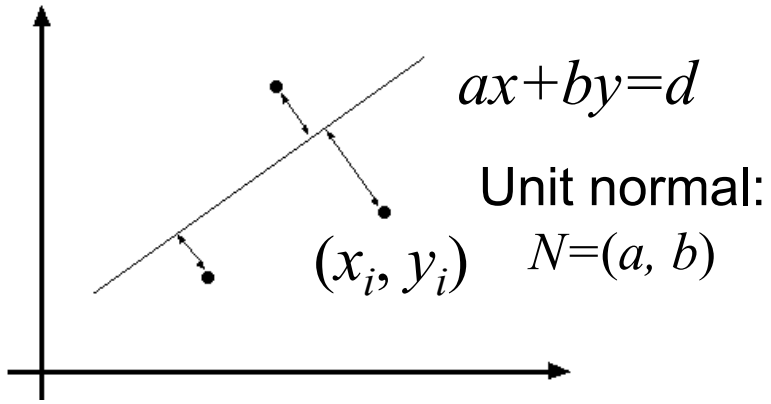
Problem with “vertical” least squares

- Not rotation-invariant
- Fails completely for vertical lines

Slide: S. Lazebnik

Total least squares

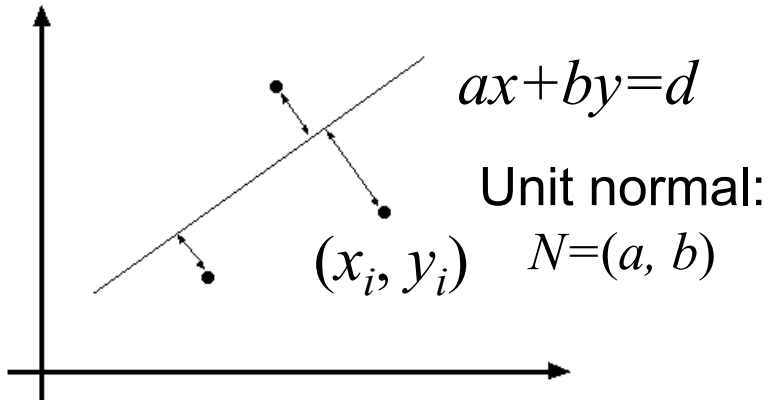
- Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$



Slide: S. Lazebnik

Total least squares

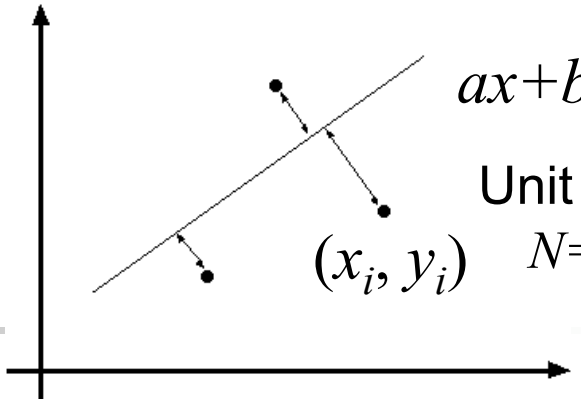
- Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$



$$E = \sum_{i=1}^{n} (ax_i + by_i - d)^2$$

Slide: S. Lazebnik

Total least squares



$$E = \sum_{i=1}^{n} (ax_i + by_i - d)^2$$

$$\frac{\partial E}{\partial d} = \sum_{i=1}^{n} -2(ax_i + by_i - d) = 0$$

$$d = \frac{a}{n}\sum_{i=1}^{n} x_i + \frac{b}{n}\sum_{i=1}^{n} x_i = a\bar{x} + b\bar{y}$$

$$E = \sum_{i=1}^{n} (a(x_i - \bar{x}) + b(y_i - \bar{y}))^2 = \left\| \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 = (UN)^T (UN)$$

$$\frac{dE}{dN} = 2(U^T U)N = 0$$

Solution to $(U^T U)N = 0$, subject to $\|N\|^2 = 1$: eigenvector of $U^T U$ associated with the smallest eigenvalue (least squares solution to *homogeneous linear system* $UN = 0$)

Slide: S. Lazebnik

Total least squares

$$U = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \quad U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \bar{x})^2 & \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \\ \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) & \sum_{i=1}^{n} (y_i - \bar{y})^2 \end{bmatrix}$$

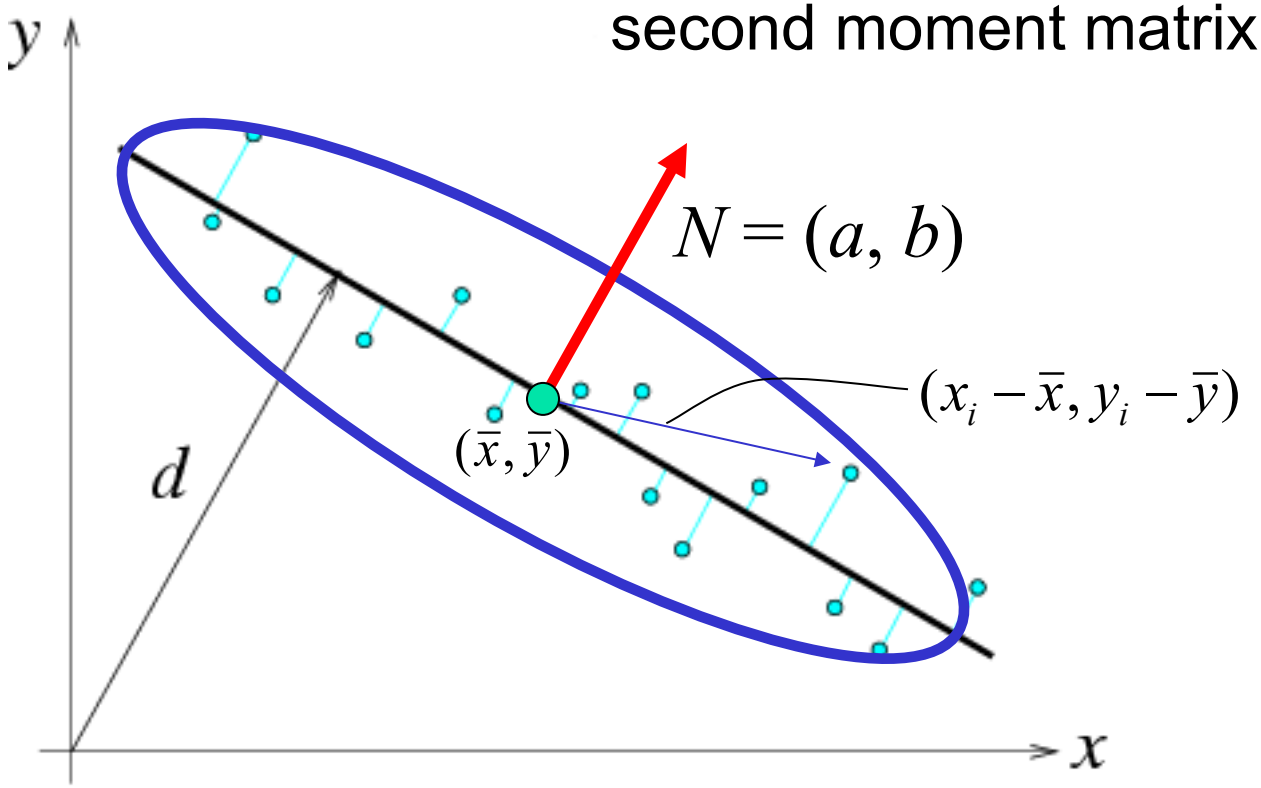
second moment matrix

Slide: S. Lazebnik

Total least squares

$$U = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \qquad U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \bar{x})^2 & \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \\ \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) & \sum_{i=1}^{n} (y_i - \bar{y})^2 \end{bmatrix}$$

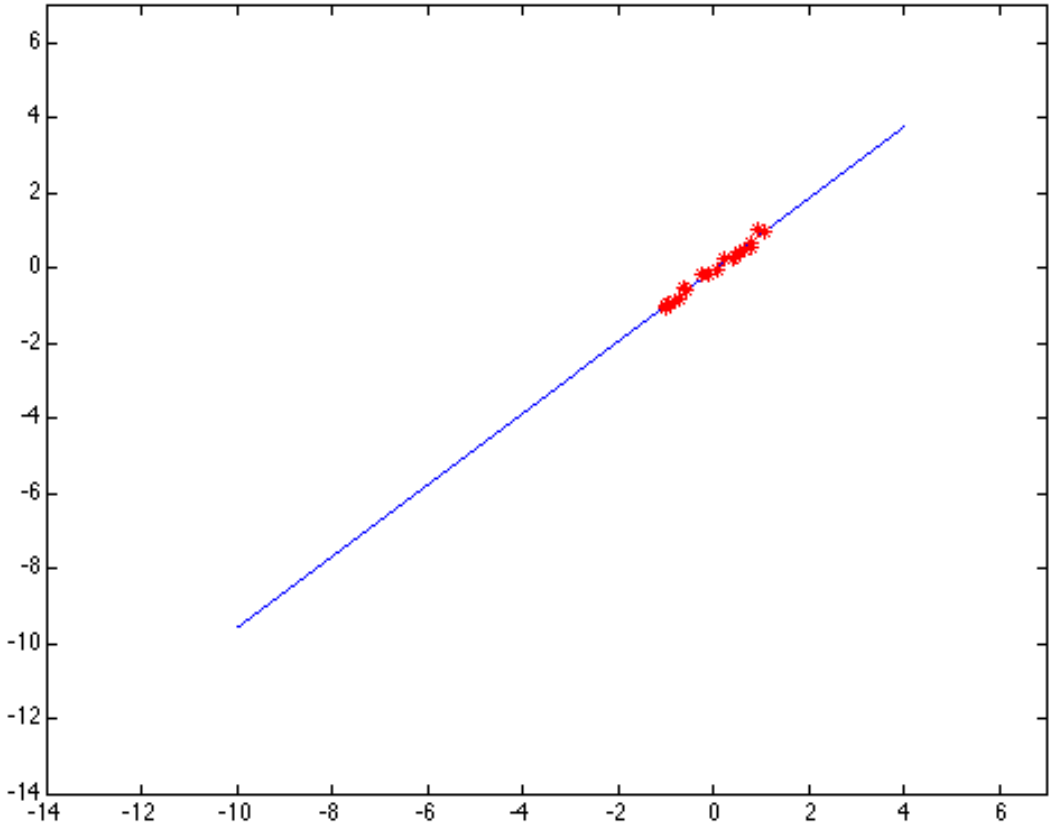
second moment matrix



Slide: S. Lazebnik

Least squares: Robustness to noise

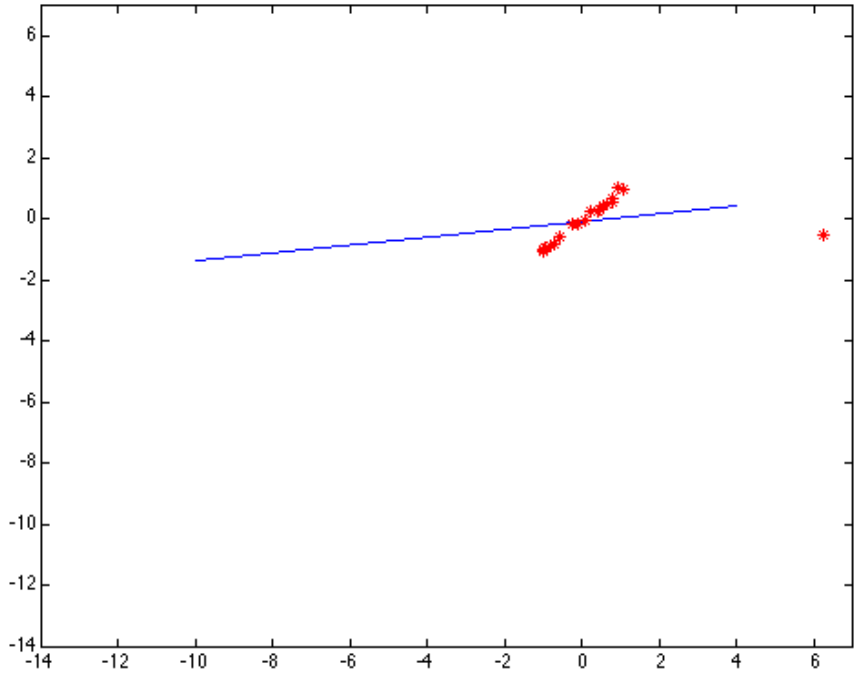
- Least squares fit to the red points:



Slide: S. Lazebnik

Least squares: Robustness to noise

- Least squares fit with an outlier:



Problem: squared error heavily penalizes outliers

Slide: S. Lazebnik

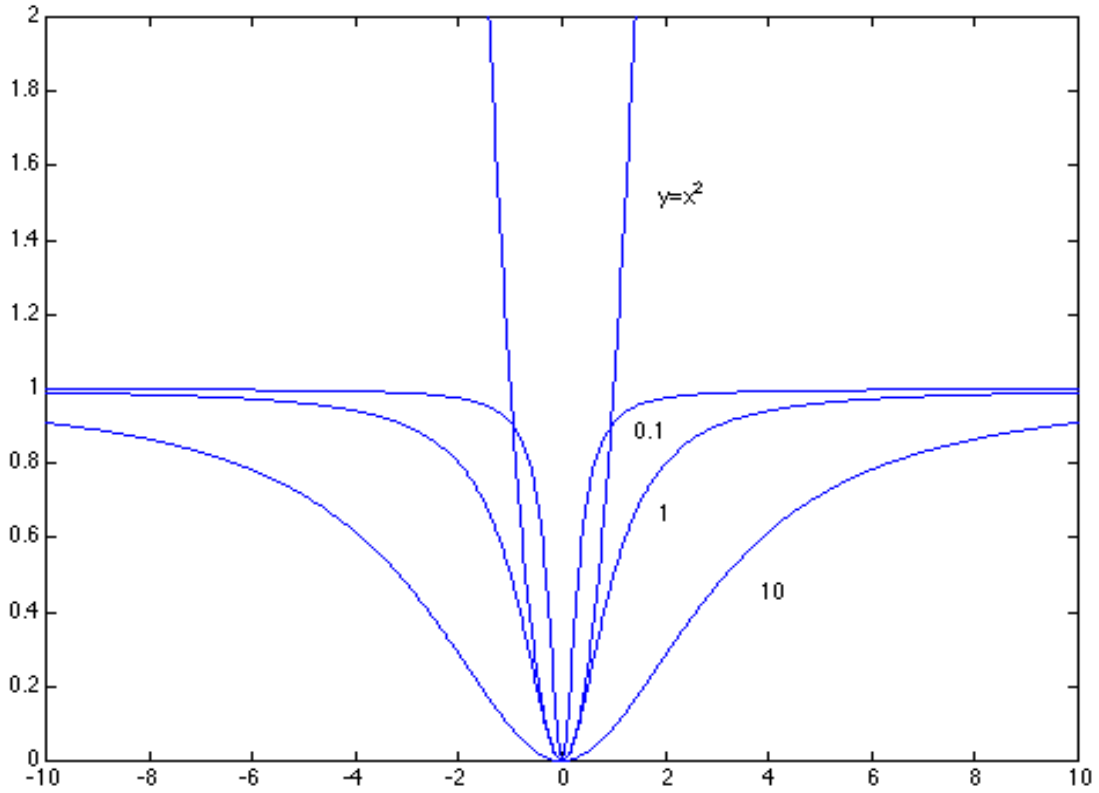
Robust estimators

$$\rho(u;\sigma) = \frac{u^2}{\sigma^2 + u^2}$$

- General approach: minimize $\sum_i \rho\left(r_i\left(x_i, \theta\right); \sigma\right)$

$r_i(x_i, \theta)$ – residual of ith point w.r.t. model parameters θ

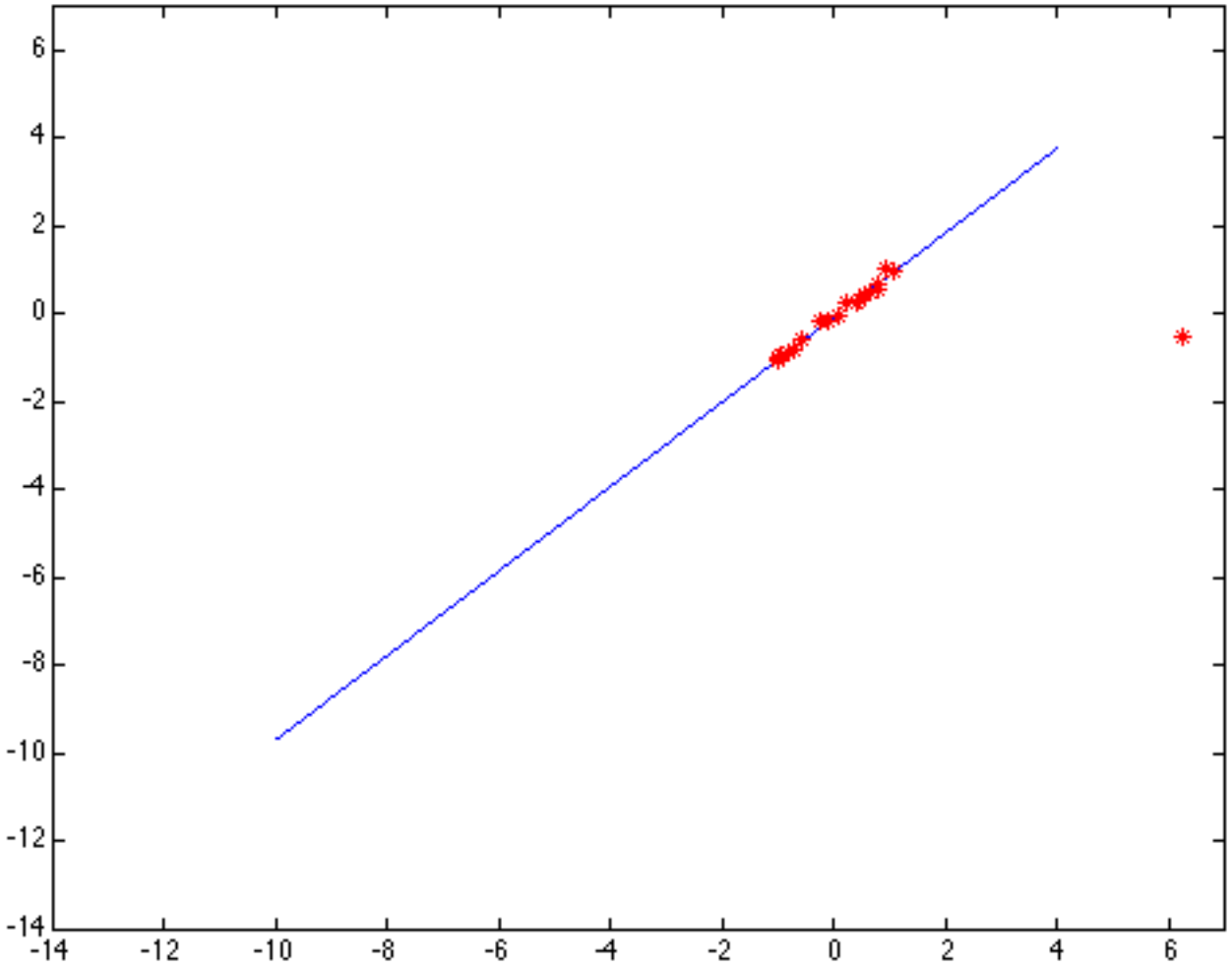
ρ – robust function with scale parameter σ



The robust function ρ behaves like squared distance for small values of the residual *u* but saturates for larger values of *u*

Slide: S. Lazebnik

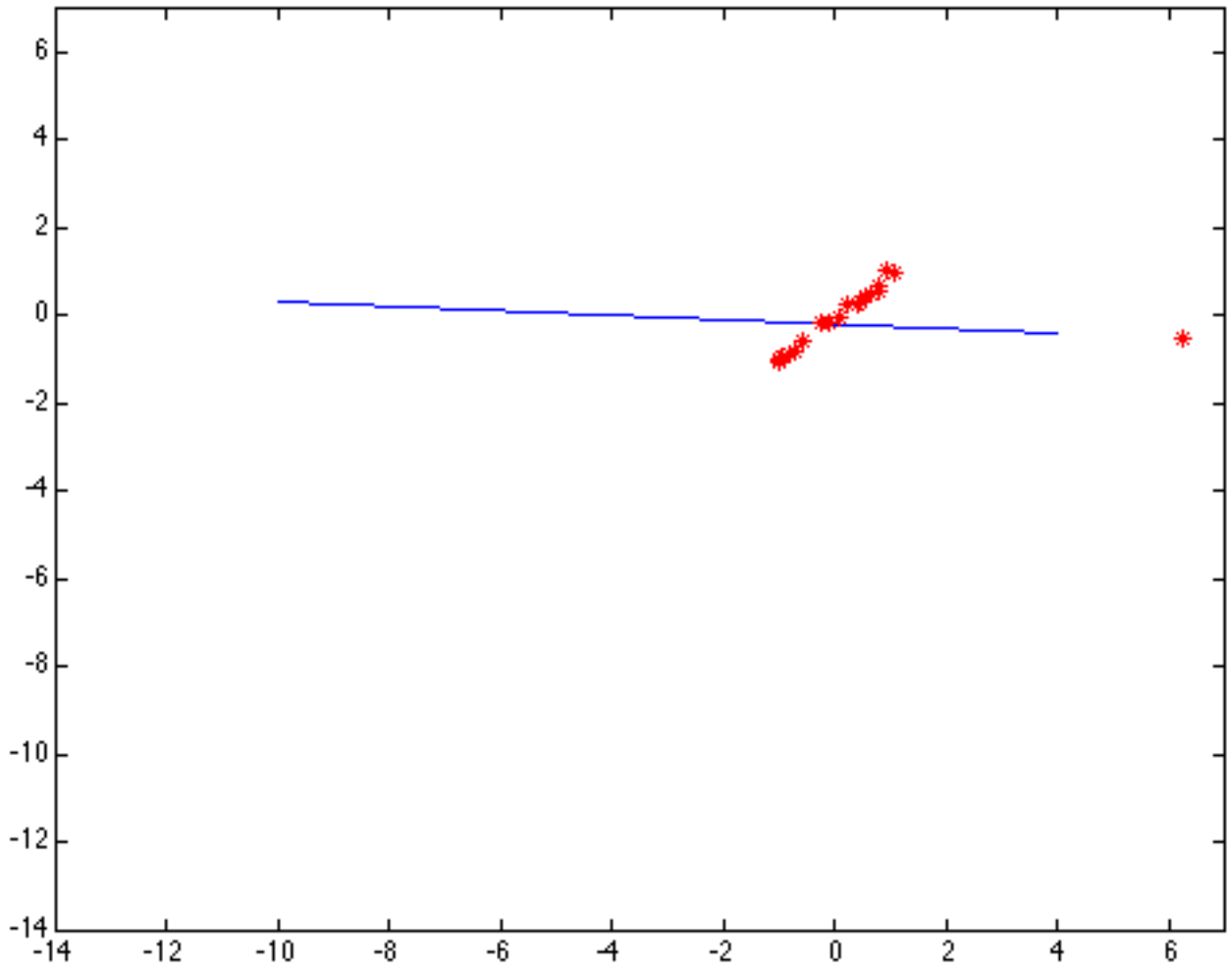
Choosing the scale: Just right



The effect of the outlier is minimized

Slide: S. Lazebnik

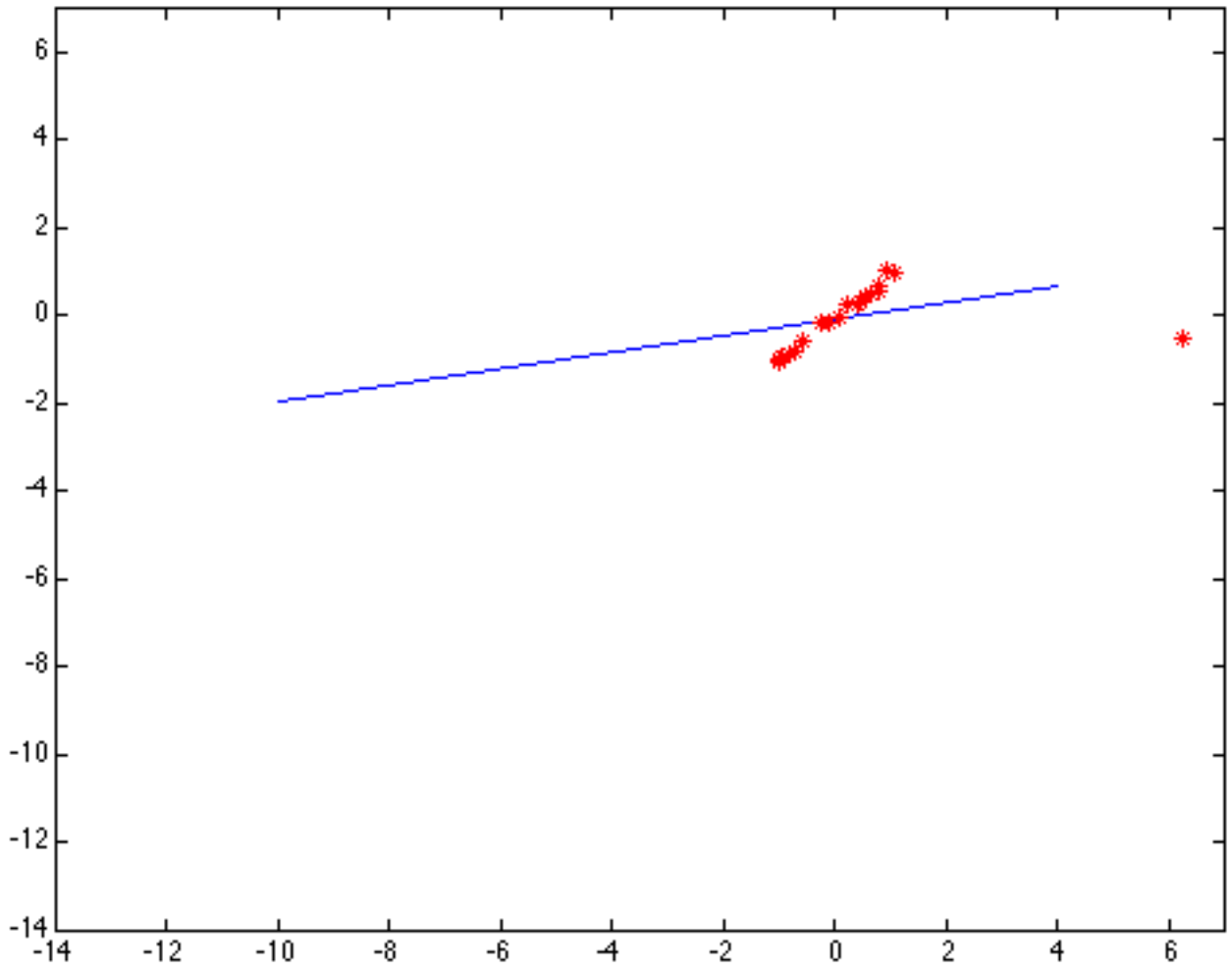
Choosing the scale: Too small



The error value is almost the same for every point and the fit is very poor

Slide: S. Lazebnik

Choosing the scale: Too large



Behaves much the same as least squares

Slide: S. Lazebnik

RANSAC

- Robust fitting can deal with a few outliers – what if we have very many?
- Random sample consensus (RANSAC): Very general framework for model fitting in the presence of outliers
- Outline
  - Choose a small subset of points uniformly at random
  - Fit a model to that subset
  - Find all remaining points that are "close" to the model and reject the rest as outliers
  - Do this many times and choose the best model

M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.

Slide: S. Lazebnik

RANSAC for line fitting

- Repeat ***N*** times:
- Draw ***s*** points uniformly at random
- Fit line to these ***s*** points
- Find inliers to this line among the remaining points (i.e., points whose distance from the line is less than ***t***)
- If there are ***d*** or more inliers, accept the line and refit using all inliers

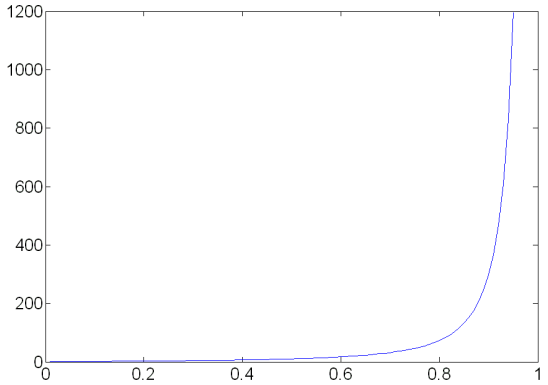
Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

Source: M. Pollefeys

Choosing the parameters



$$\left(1-\left(1-e\right)^{s}\right)^{N}=1-p$$

	proportion of outliers e						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

$$N=\log\left(1-p\right)/\log\left(1-\left(1-e\right)^{s}\right)$$

Probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

Source: M. Pollefeys

Choosing the parameters

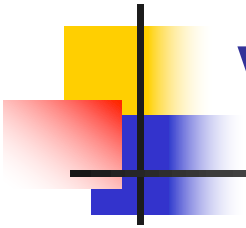
- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)
- Consensus set size *d*
  - Should match expected inlier ratio

Source: M. Pollefeys

RANSAC pros and cons

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Lots of parameters to tune
  - Can’t always get a good initialization of the model based on the minimum number of samples
  - Sometimes too many iterations are required
  - Can fail for extremely low inlier ratios
  - We can often do better than brute-force sampling

Source: M. Pollefeys



Voting schemes

- Let each feature vote for all the models that are compatible with it
- Hopefully the noise features will not vote consistently for any single model
- Missing data doesn't matter as long as there are enough features remaining to agree on a good model

Source: S. Seitz

Hough transform

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

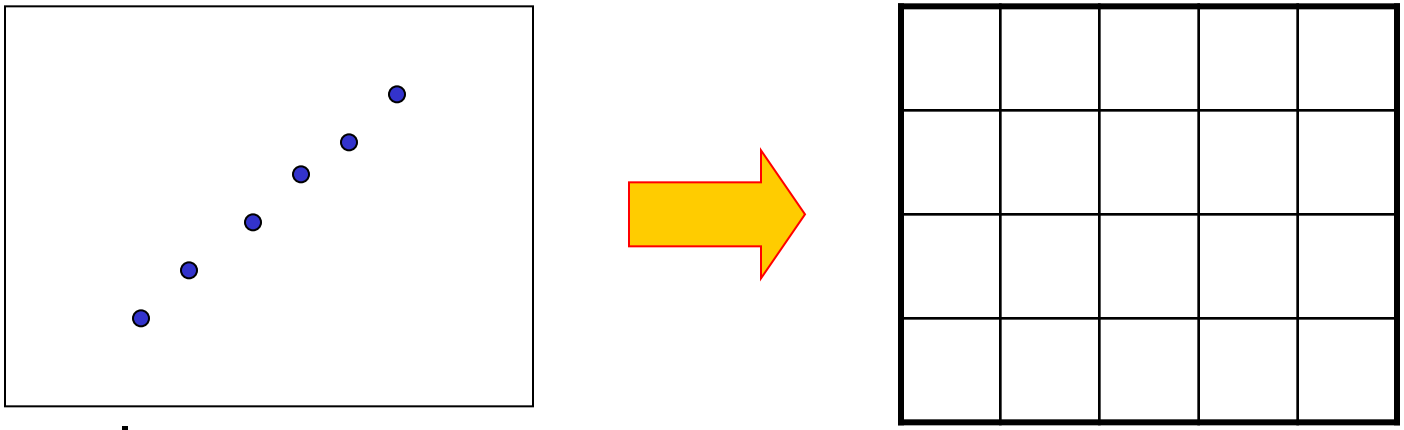


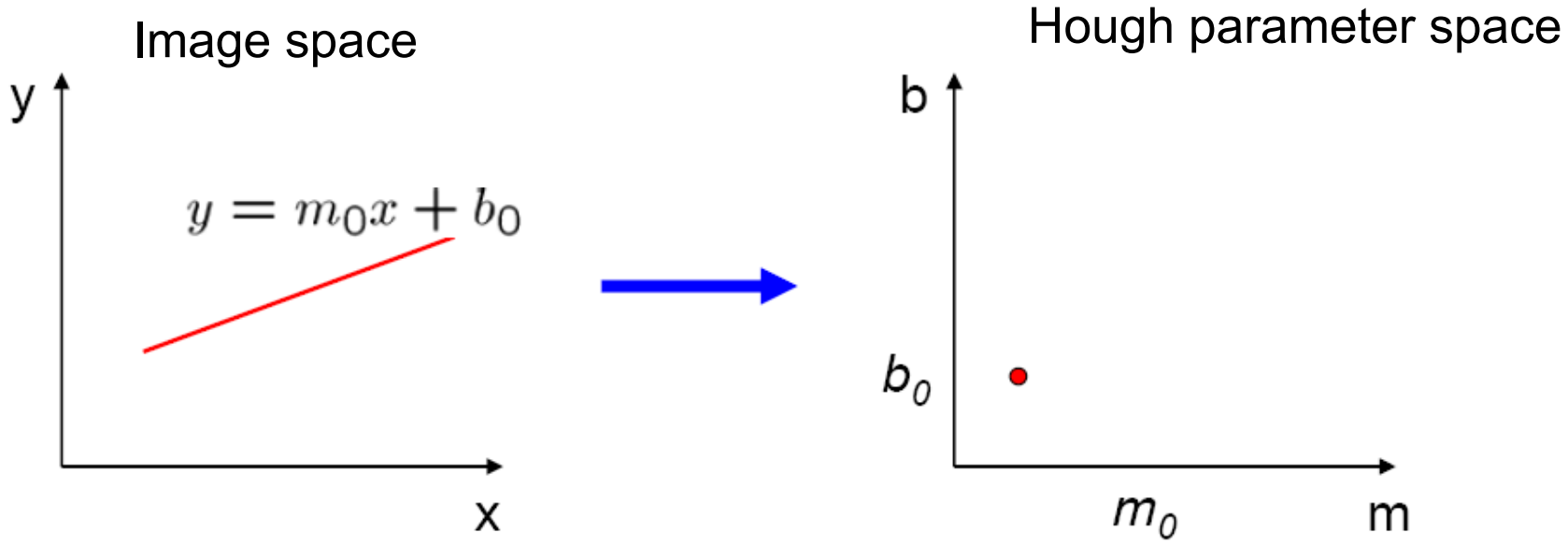
Image space

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

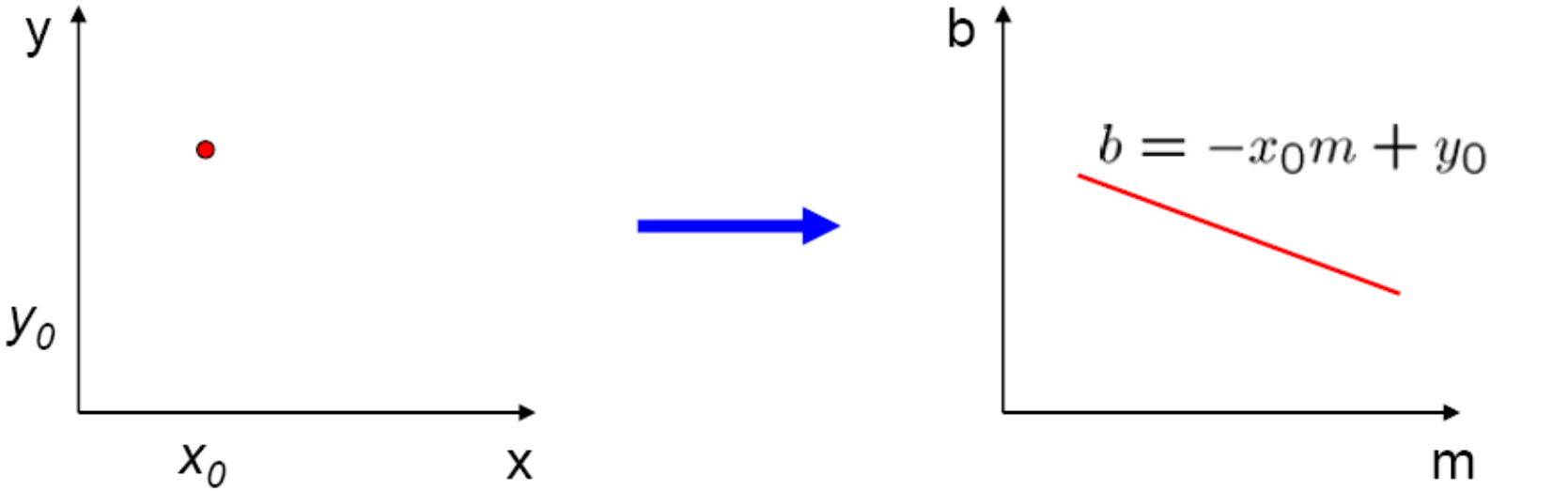
- A line in the image corresponds to a point in Hough space



Source: S. Seitz

Parameter space representation

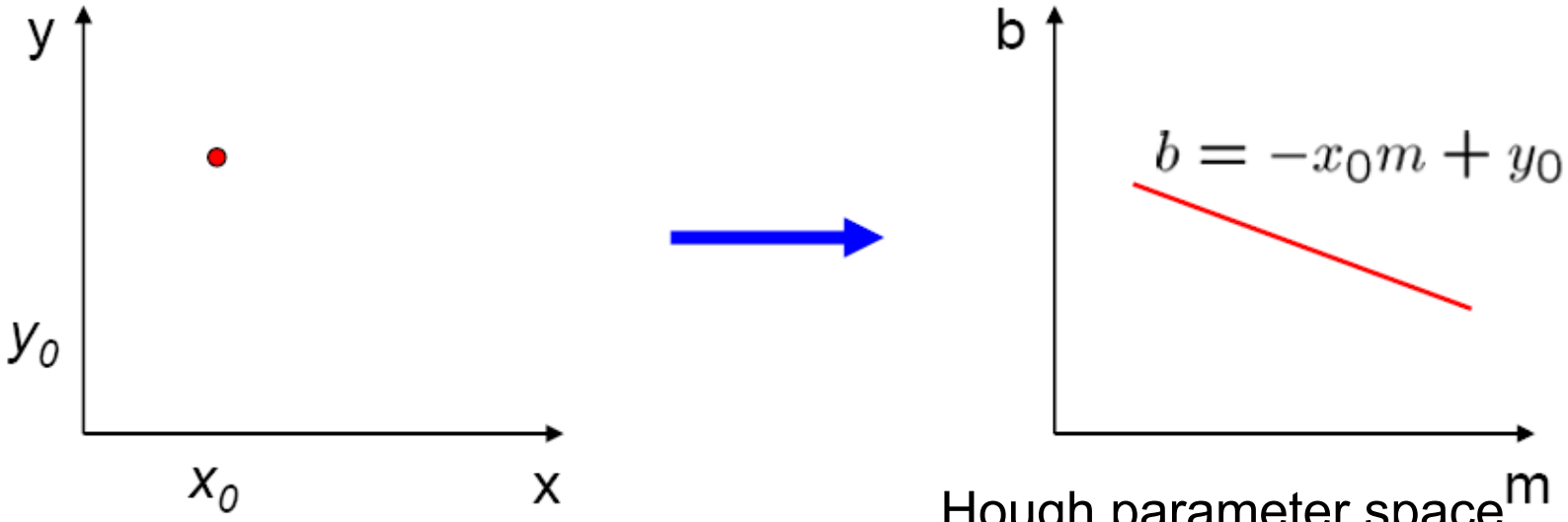
- What does a point (x_0, y_0) in the image space map to in the Hough space?



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

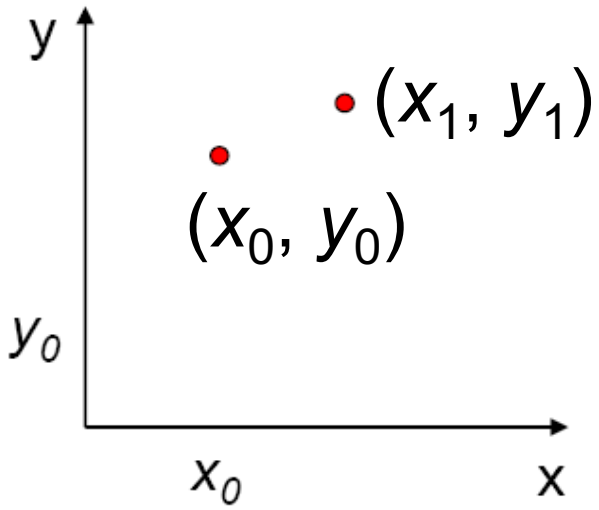


Source: S. Seitz

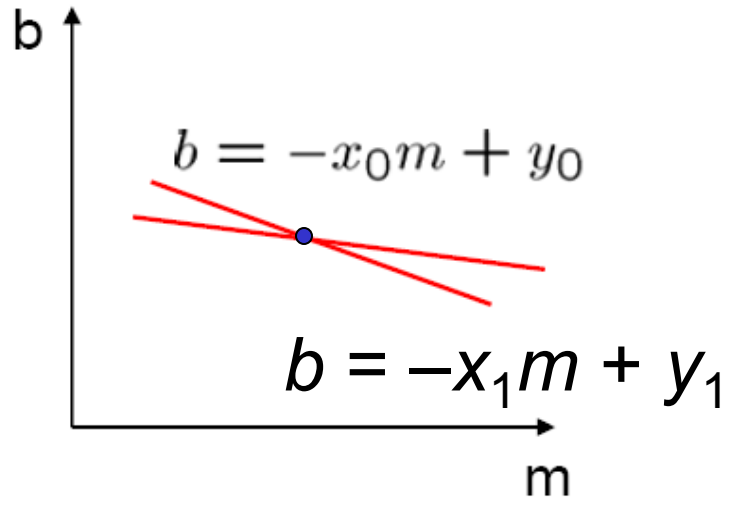
Parameter space representation

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

Image space



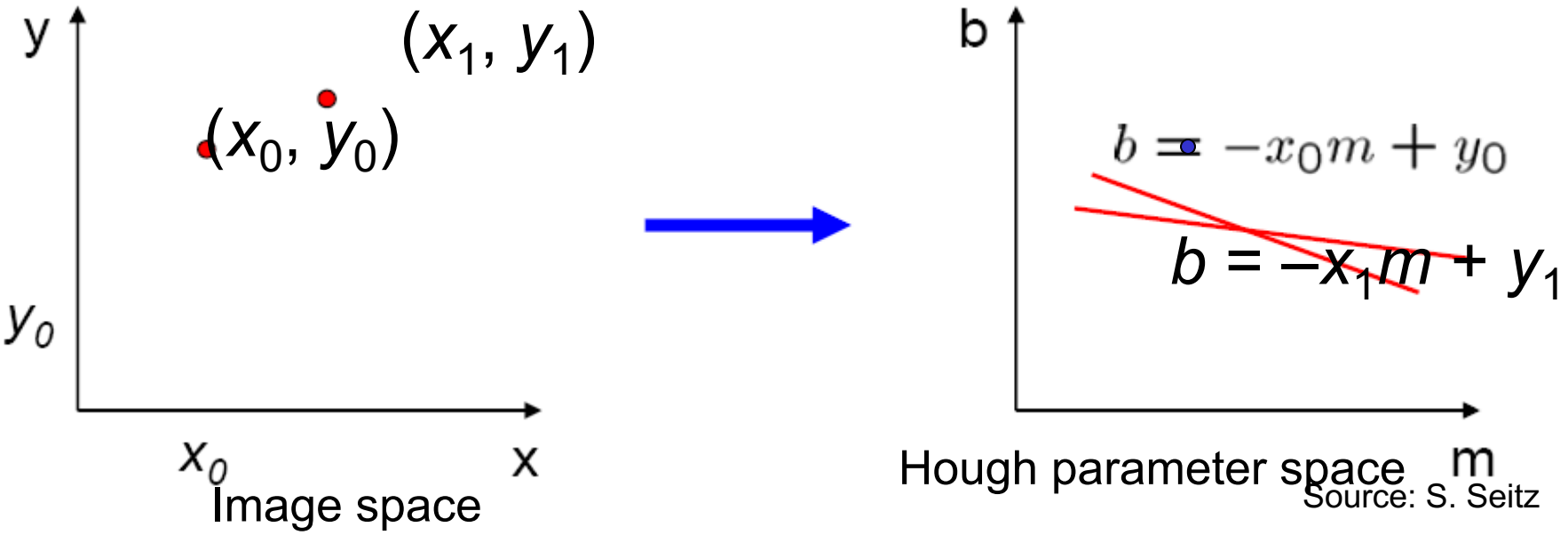
Hough parameter space



Source: S. Seitz

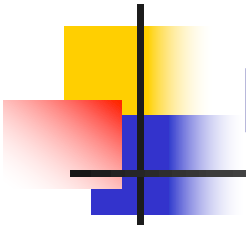
Parameter space representation

- 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



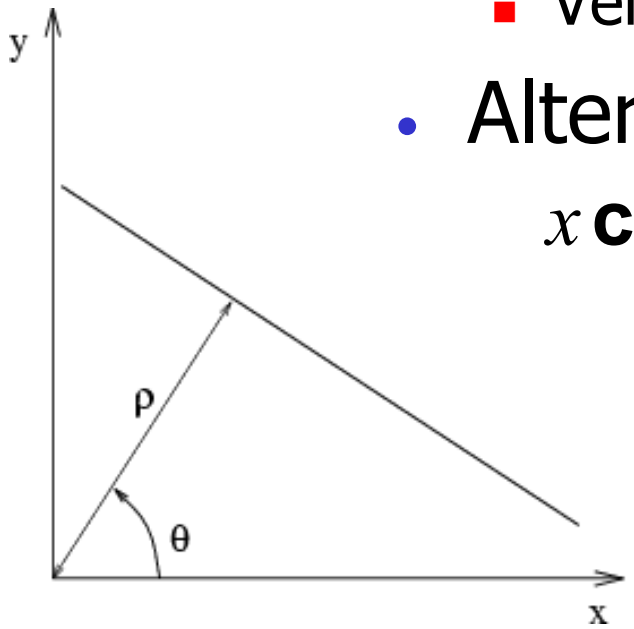
- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m

Source: S. Seitz

Parameter space representation

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

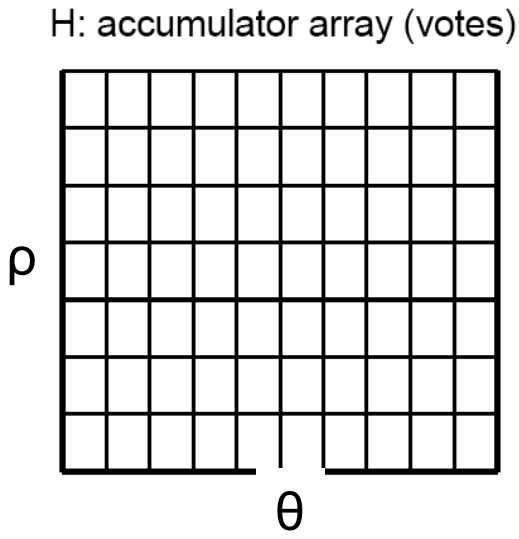
$$x\cos\theta + y\sin\theta = \rho$$



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

Source: S. Seitz

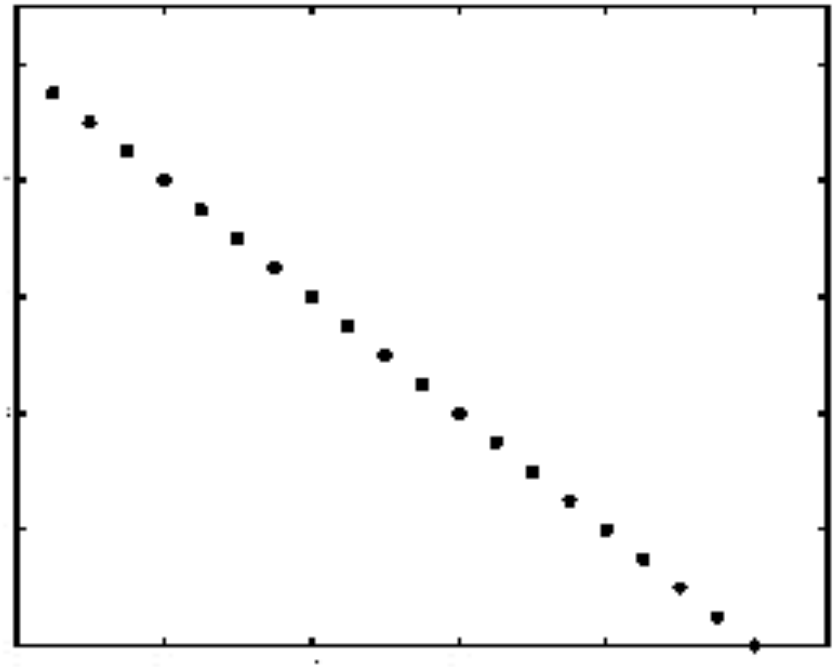
Algorithm outline



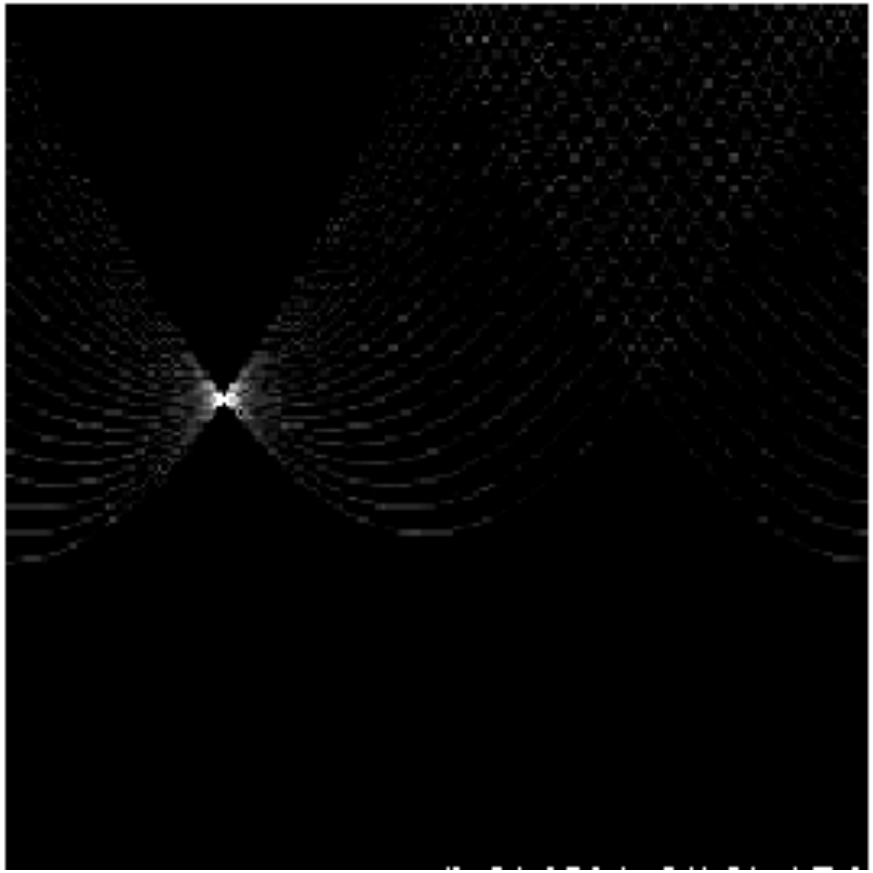
- Initialize accumulator H to all zeros
- For each edge point (x,y) in the image
  For θ = 0 to 180
    ρ = x cos θ + y sin θ
    H(θ, ρ) = H(θ, ρ) + 1
  end
end
- Find the value(s) of (θ, ρ) where H(θ, ρ) is a local maximum
  - The detected line in the image is given by
    ρ = x cos θ + y sin θ

Source: S. Seitz

Basic illustration



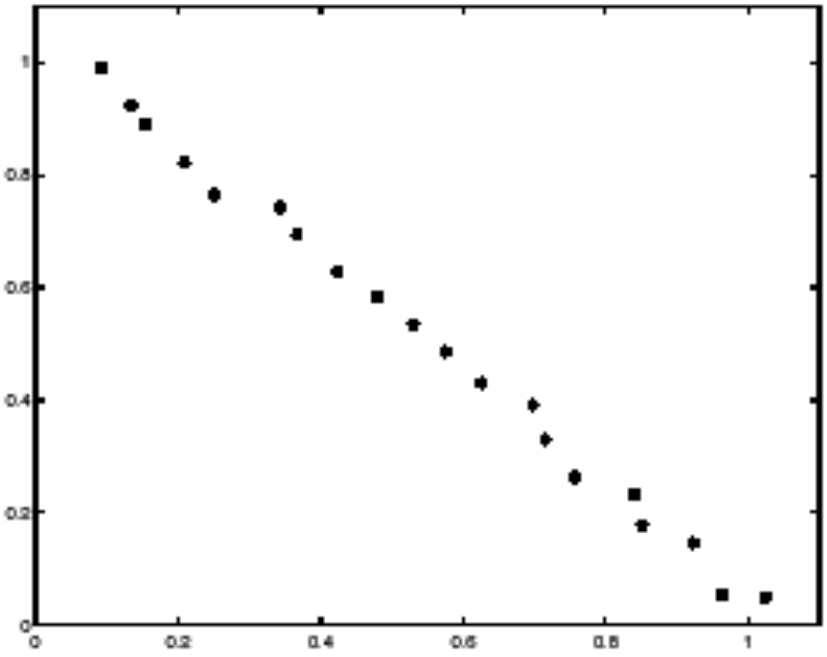
features



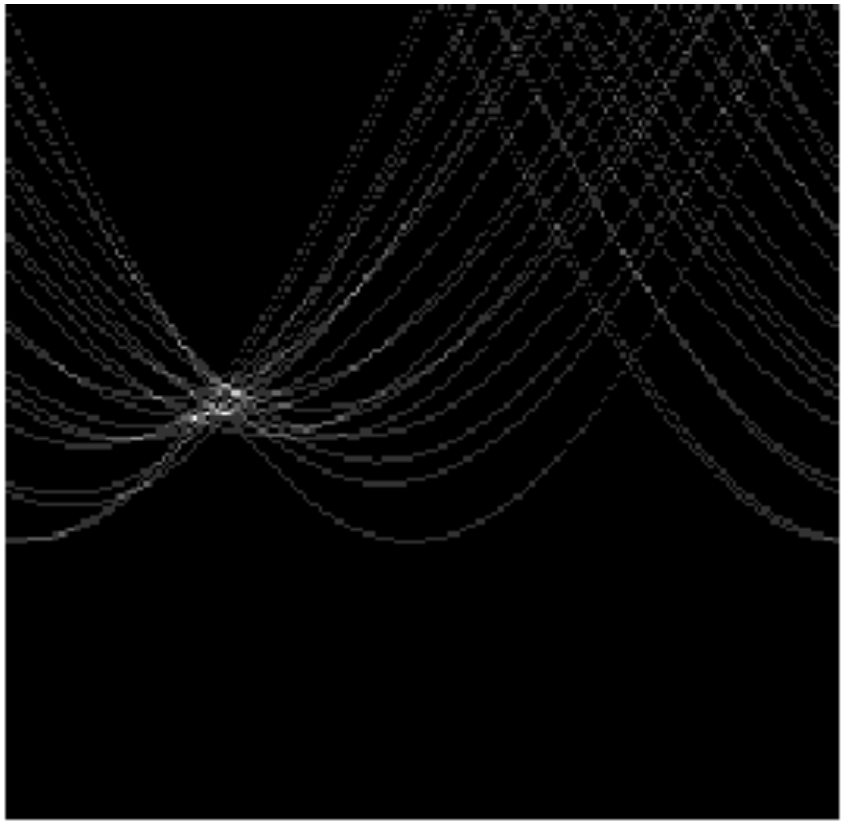
votes

Source: S. Seitz

Effect of noise



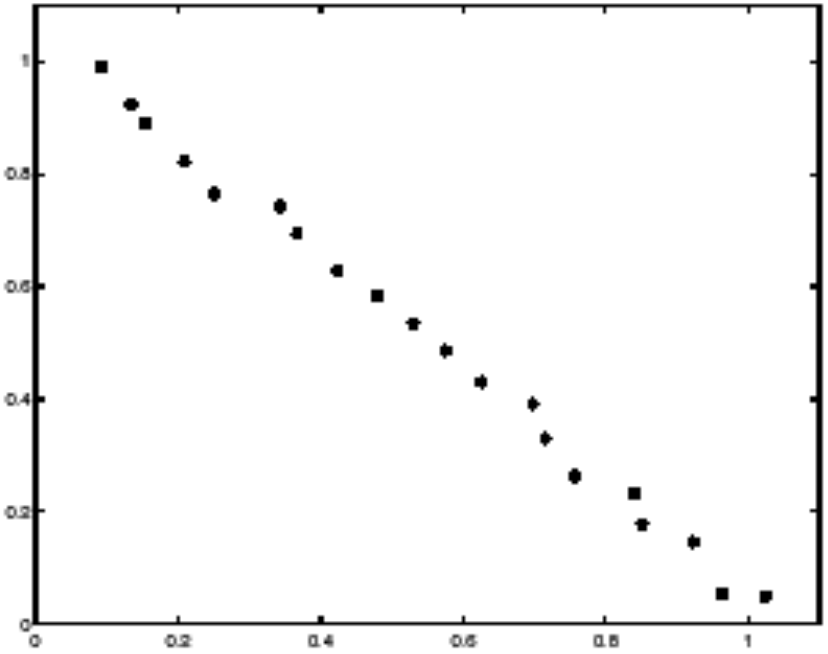
features



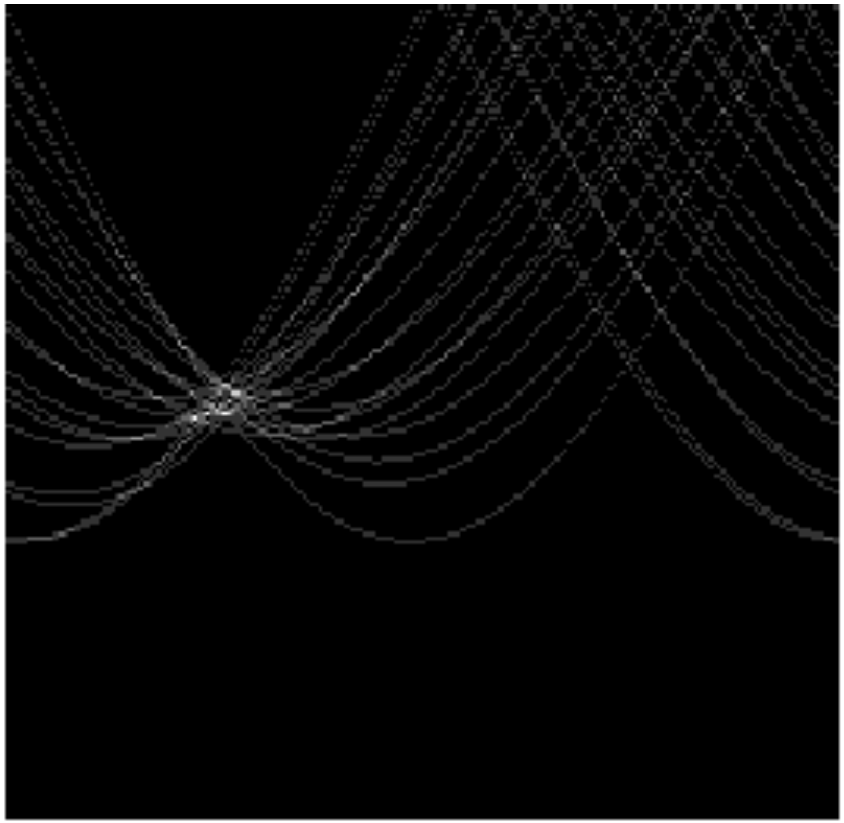
votes

Source: S. Seitz

Effect of noise



features

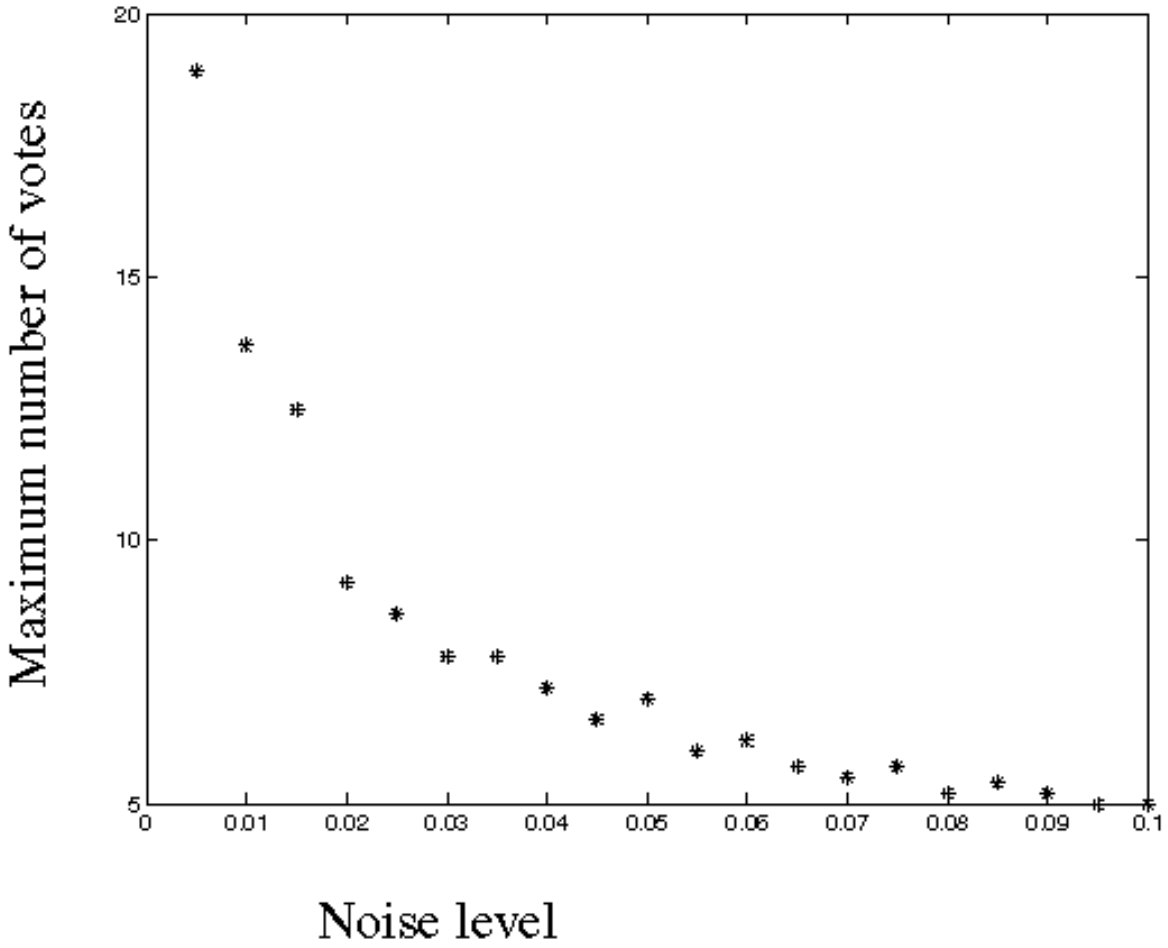


votes

- Peak gets fuzzy and hard to locate

Source: S. Seitz

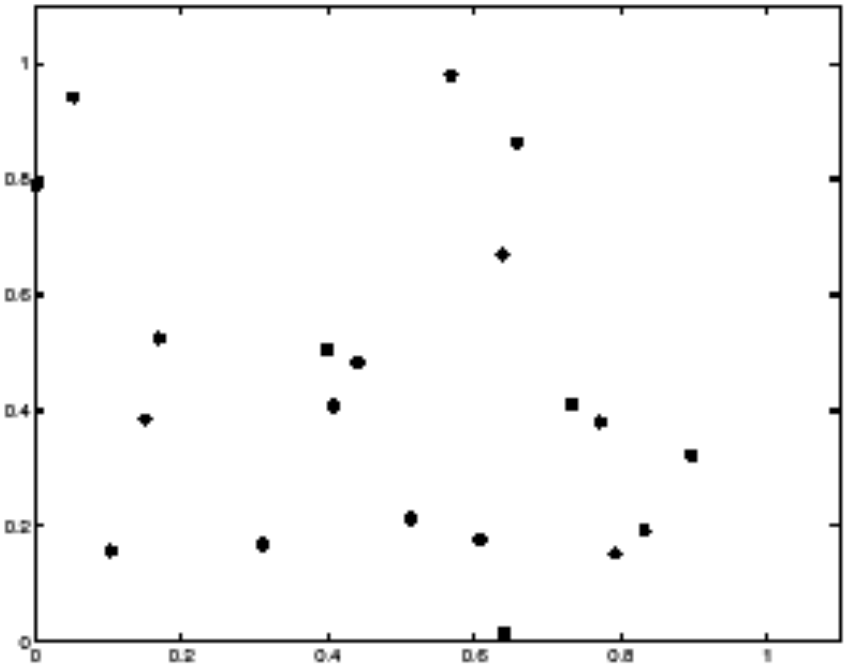
Effect of noise



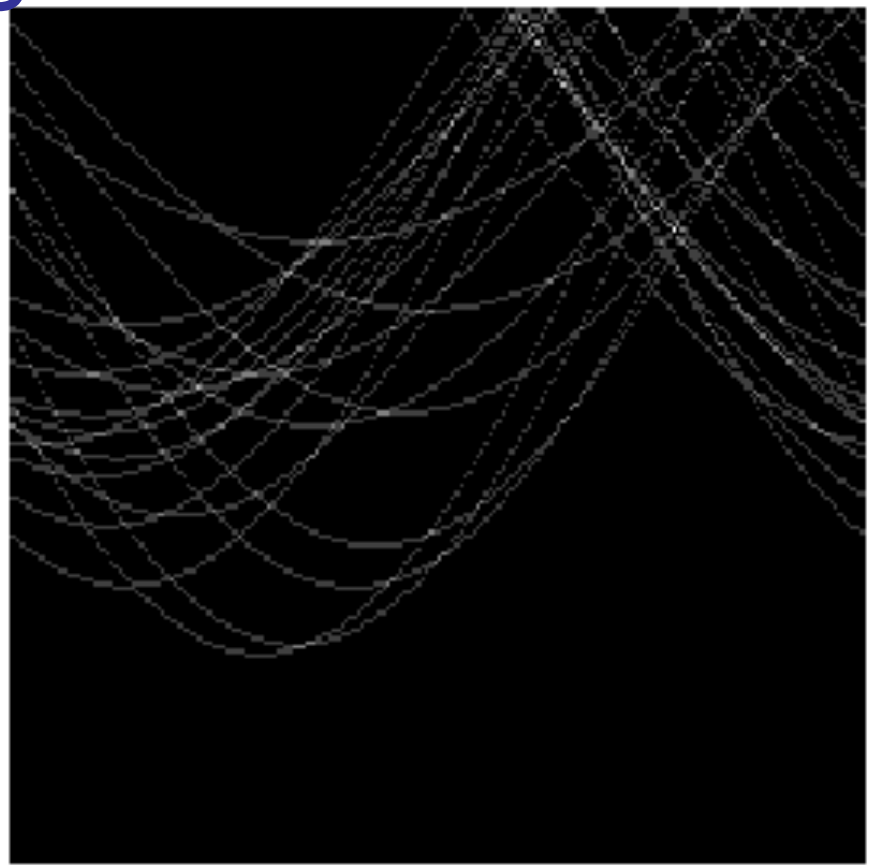
- Number of votes for a line of 20 points with increasing noise.

Source: S. Seitz

Random points



features

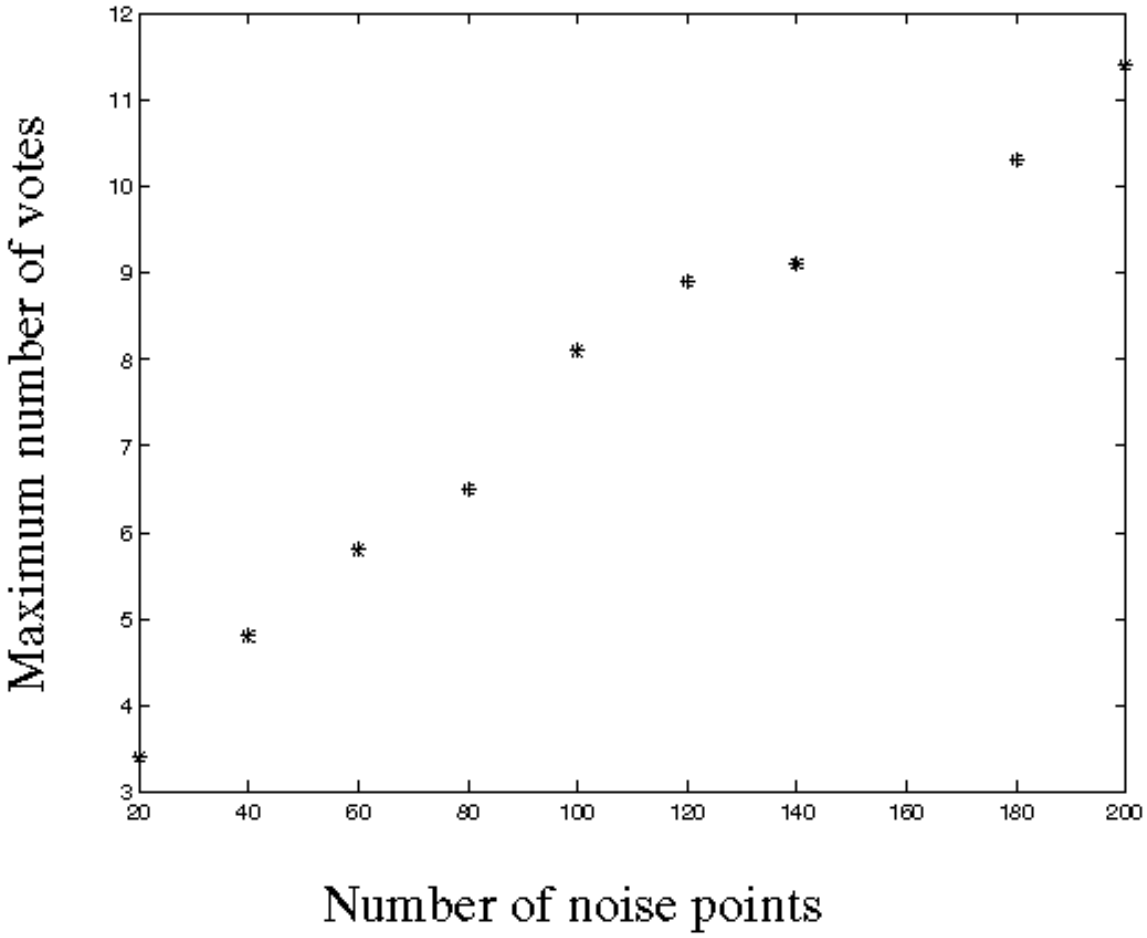


votes

- Uniform noise can lead to spurious peaks in the array

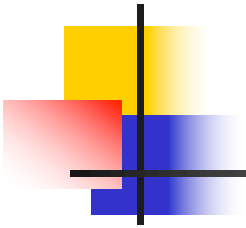
Source: S. Seitz

Random points



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

Source: S. Seitz



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

Source: S. Seitz

Summary of Techniques

- Scale and transformation invariant feature detection techniques:
  - Harris corner-Laplacian Maximum.
  - DOG Maximum.
  - Intensity weighted FAST.
- Scale and transformation invariant Feature descriptor techniques:
  - SIFT, SURF, ORB
- Model fitting techniques:
  - LSE, RANSAC, Hough Transform


Thank you!