Robotic 3D Vision

Lecture 7: Keypoint Detection, Description and Matching

WS 2020/21
Dr. Niclas Zeller
Artisense GmbH

What We Will Cover Today

- Uncertainty in pose estimation (leftover from last lecture)
- Keypoint detection
  - Corner detection
  - Blob detection
- Keypoint description
  - Scale-Invariant Feature Transform (SIFT)
- Keypoint matching
- RANSAC

Recap: Uncertainty Propagation

- Given a non-linear function in a Gaussian variable

$$\mathbf{y} = f(\mathbf{x})$$

- Apply first-order Taylor approximation

$$\mathbf{y} \approx f(\mathbf{x}_0) + \nabla_{\mathbf{x}} f(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0)$$

- Note: Linear transformation $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ of Gaussian variable remains Gaussian: $\mathbf{y} \sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu}_{\mathbf{x}} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{A}^\top)$

- Gaussian approximation of the non-linearly transformed variable

$$\mathbf{y} \sim \mathcal{N}(f(\boldsymbol{\mu}_{\mathbf{x}}), \nabla_{\mathbf{x}} f(\boldsymbol{\mu}_{\mathbf{x}})\boldsymbol{\Sigma}_{\mathbf{x}}\nabla_{\mathbf{x}} f(\boldsymbol{\mu}_{\mathbf{x}})^\top)$$

Recap: Uncertainty of Depth Estimates

- Given Gaussian uncertainty in the disparity σ_d^2
- Inverse depth z^{-1} will also be Gaussian

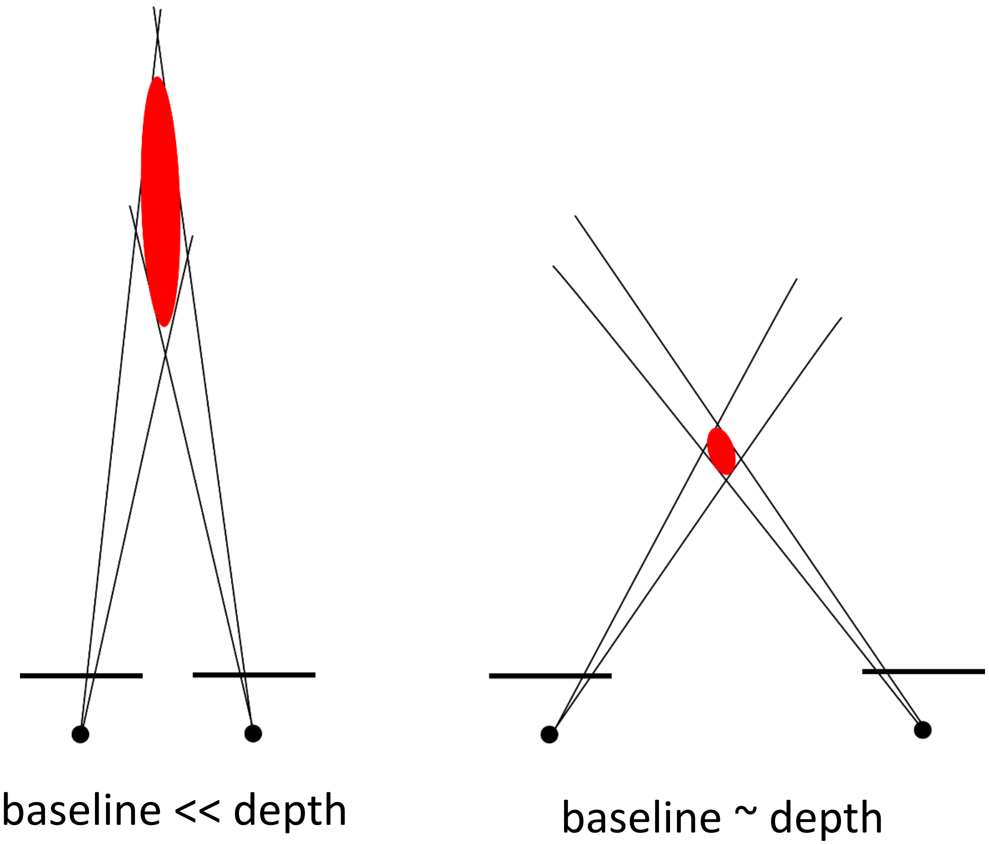
$$z^{-1} = \frac{d}{bf}$$

$$\sigma_{z^{-1}}^2 = \frac{1}{(bf)^2}\sigma_d^2$$

- Uncertainty in depth z can be approximate by a Gaussian with

$$\sigma_z^2 = \left|\frac{\partial z}{\partial d}\right|^2 \sigma_d^2 = \frac{(bf)^2}{d^4}\sigma_d^2$$

$$= \frac{z^4}{(bf)^2}\sigma_d^2$$

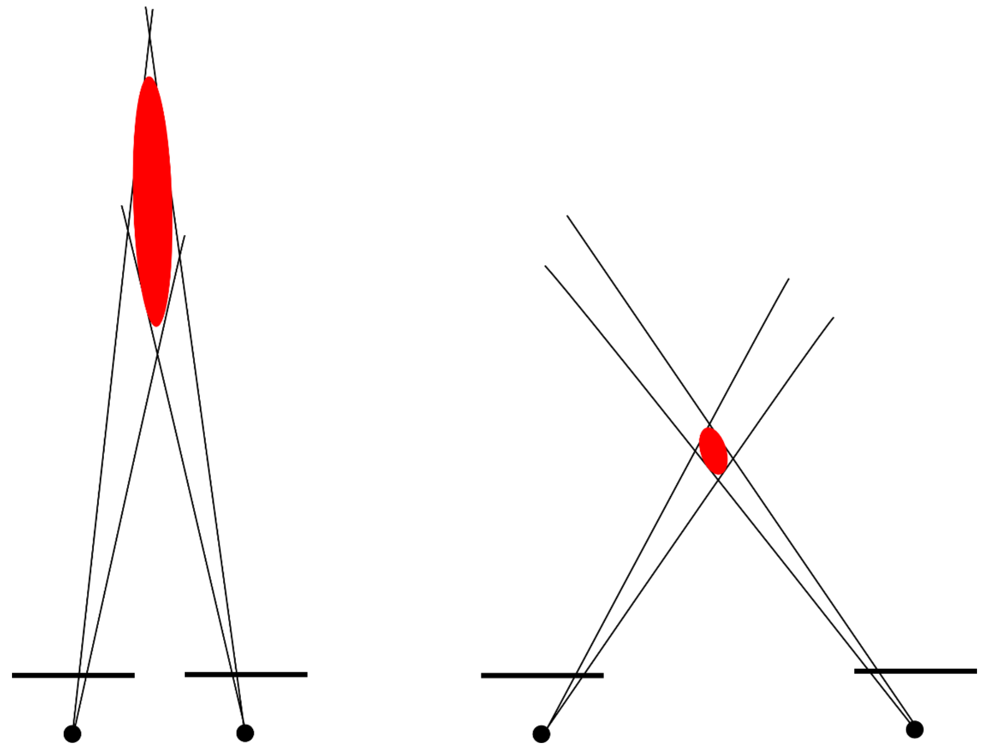


Drift in Motion Estimates

- Since we aggregate pose estimates from relative pose estimates, estimation errors in relative poses accumulate: Drift

- Noisy observations of 2D image point location

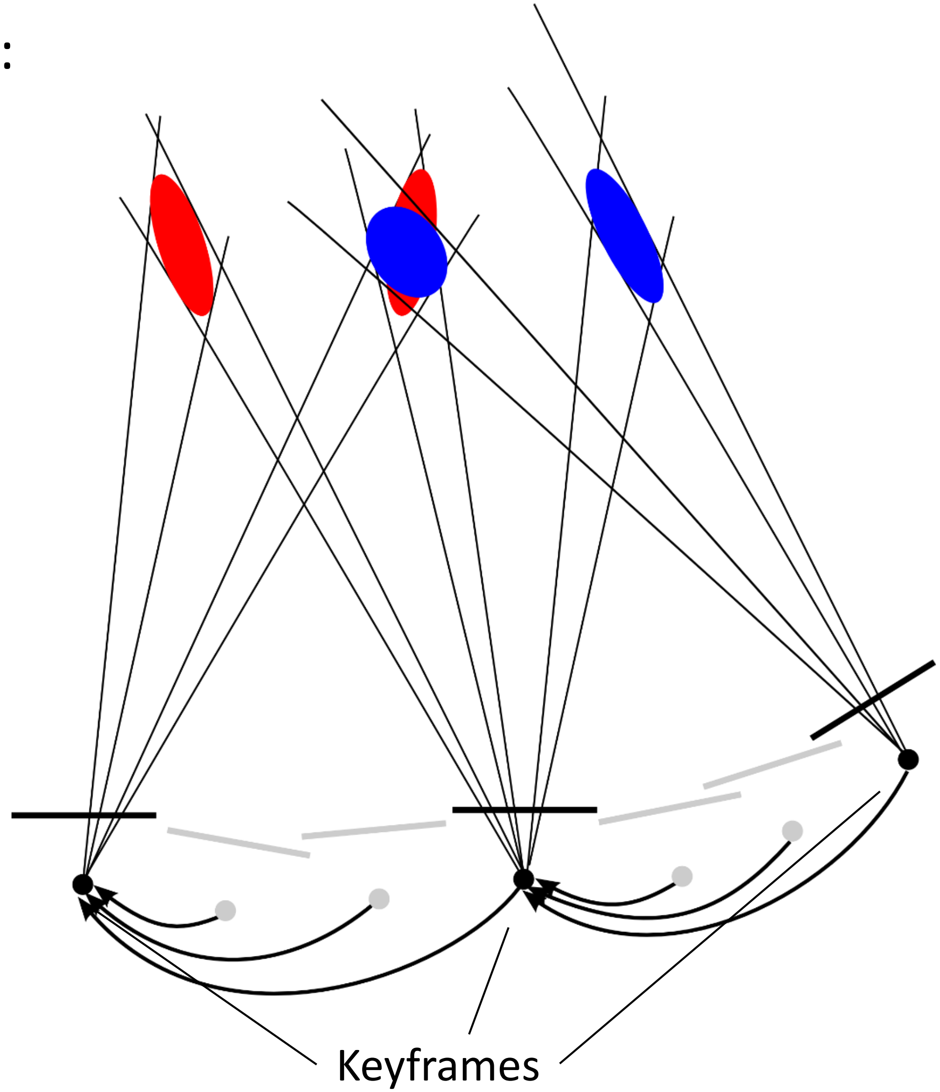
- How does uncertainty in motion estimate depend on observation noise?



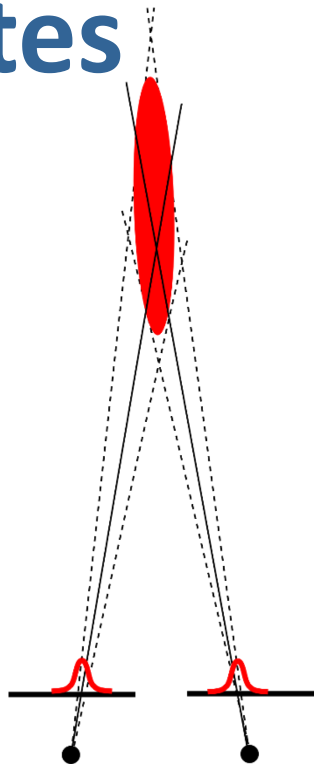
baseline << depth baseline ~ depth

Keyframes

- Popular approach to reduce drift: Keyframes
- Carefully select reference images for motion estimation / triangulation
- Incrementally estimate motion towards keyframe
- Select keyframes which have large baseline but still sufficient overlap



Uncertainty in Pose Estimates



- Model image point observation likelihood $p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi})$

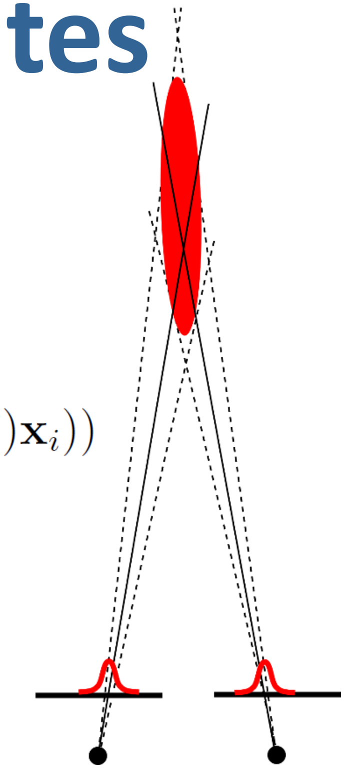
  f.e. Gaussian: $p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi}) = \mathcal{N}(\mathbf{y}_i; \pi(\mathbf{T}(\boldsymbol{\xi})\overline{\mathbf{x}}_i), \boldsymbol{\Sigma}_{\mathbf{y}_i})$

- Optimize maximum a-posteriori likelihood of estimates

$$p(\mathcal{X}, \boldsymbol{\xi} \mid \mathcal{Y}) \propto p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\xi})\, p(\mathcal{X}, \boldsymbol{\xi}) = p(\mathcal{X}, \boldsymbol{\xi}) \prod_{i=1}^{N} p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi})$$

Neg. log-likelihood: $E(\mathcal{X}, \boldsymbol{\xi}) = -\log(p(\mathcal{X}, \boldsymbol{\xi})) - \sum_{i=1}^{N} \log(p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi}))$

Uncertainty in Pose Estimates



- Gaussian prior and observation likelihood:

$$E(\mathcal{X}, \boldsymbol{\xi}) = \text{const.} + \left(\boldsymbol{\xi} - \boldsymbol{\mu}_{\boldsymbol{\xi},0}\right)^\top \boldsymbol{\Sigma}_{\boldsymbol{\xi},0}^{-1} \left(\boldsymbol{\xi} - \boldsymbol{\mu}_{\boldsymbol{\xi},0}\right) +$$

$$\sum_{i=1}^{N} \left(\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{x}_i,0}\right)^\top \boldsymbol{\Sigma}_{\mathbf{x}_i,0}^{-1} \left(\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{x}_i,0}\right) + \left(\mathbf{y}_i - \pi\left(\mathbf{T}(\boldsymbol{\xi})\mathbf{x}_i\right)\right)^\top \boldsymbol{\Sigma}_{\mathbf{y}_i}^{-1} \left(\mathbf{y}_i - \pi\left(\mathbf{T}(\boldsymbol{\xi})\mathbf{x}_i\right)\right)$$

- We use Gauss-Newton to find

$$\arg\min_{\mathbf{x}} E(\mathbf{x}) = \tfrac{1}{2}\mathbf{r}(\mathbf{x})^\top \mathbf{W} \mathbf{r}(\mathbf{x})$$

  - $\mathbf{W}$ models inverse covariances of observations and priors
  - The inverse Hessian of the Gauss-Newton approximation

$$\boldsymbol{\Sigma} \approx \left(\nabla_{\mathbf{x}} \mathbf{r}(\boldsymbol{\mu})^\top \mathbf{W} \nabla_{\mathbf{x}} \mathbf{r}(\boldsymbol{\mu})\right)^{-1}$$

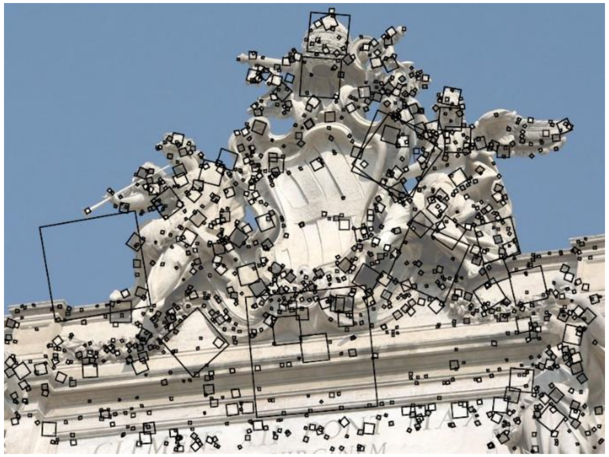
yields an approximate covariance of the estimates

Recap: Keypoint Detection

- Desirable properties of keypoint detectors for visual odometry:
  - high repeatability,
  - localization accuracy,
  - robustness,
  - invariance,
  - computational efficiency

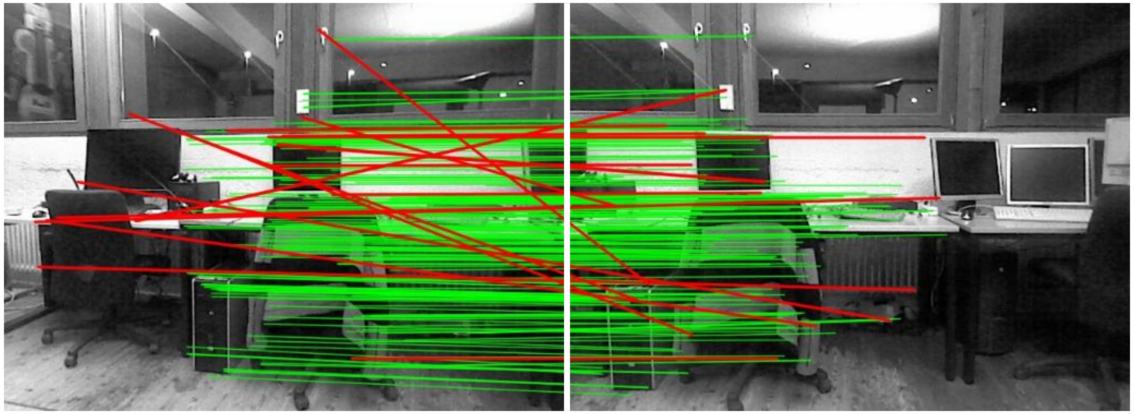
Harris Corners

Image source: Svetlana Lazebnik



DoG (SIFT) Blobs

Recap: Keypoint Matching



- Desirable properties for VO:
  - High recall
  - Precision
  - Robustness
  - Computational efficiency
- One possible approach to keypoint matching: by descriptor

Image Matching

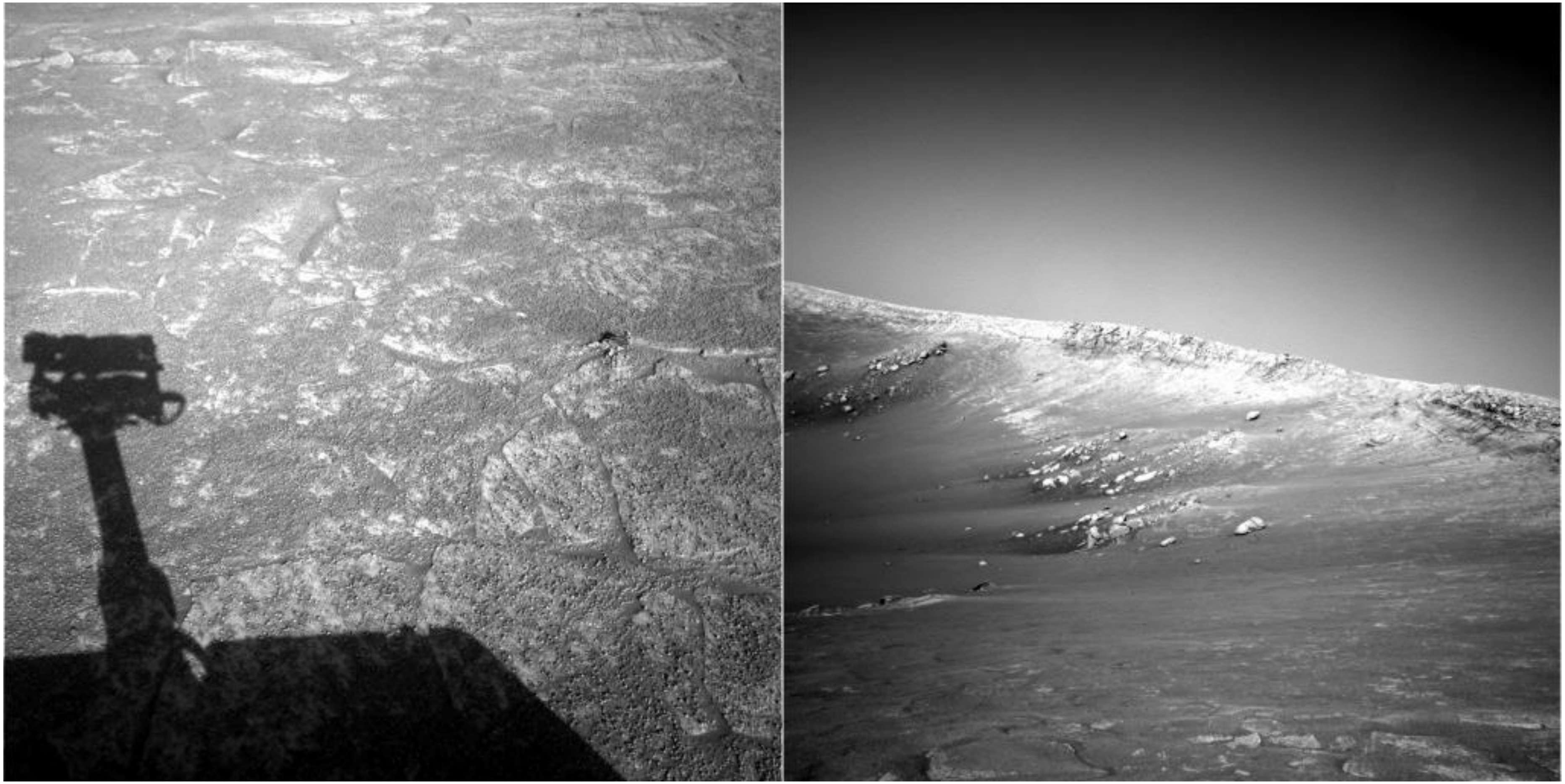


Figure by Noah Snavely

NASA Mars Rover images

Slide adapted from Steve Seitz

Image Matching

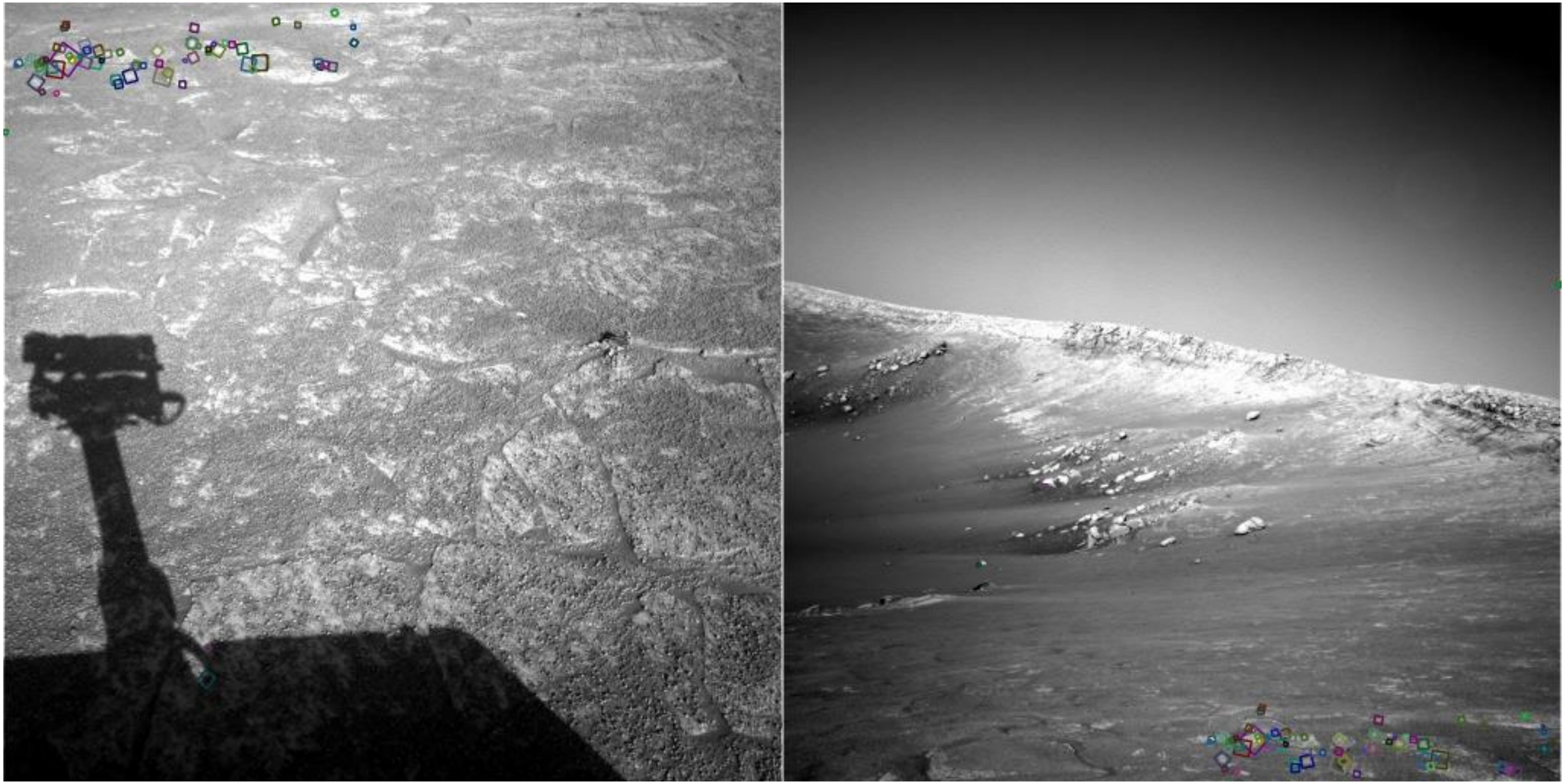


Figure by Noah Snavely

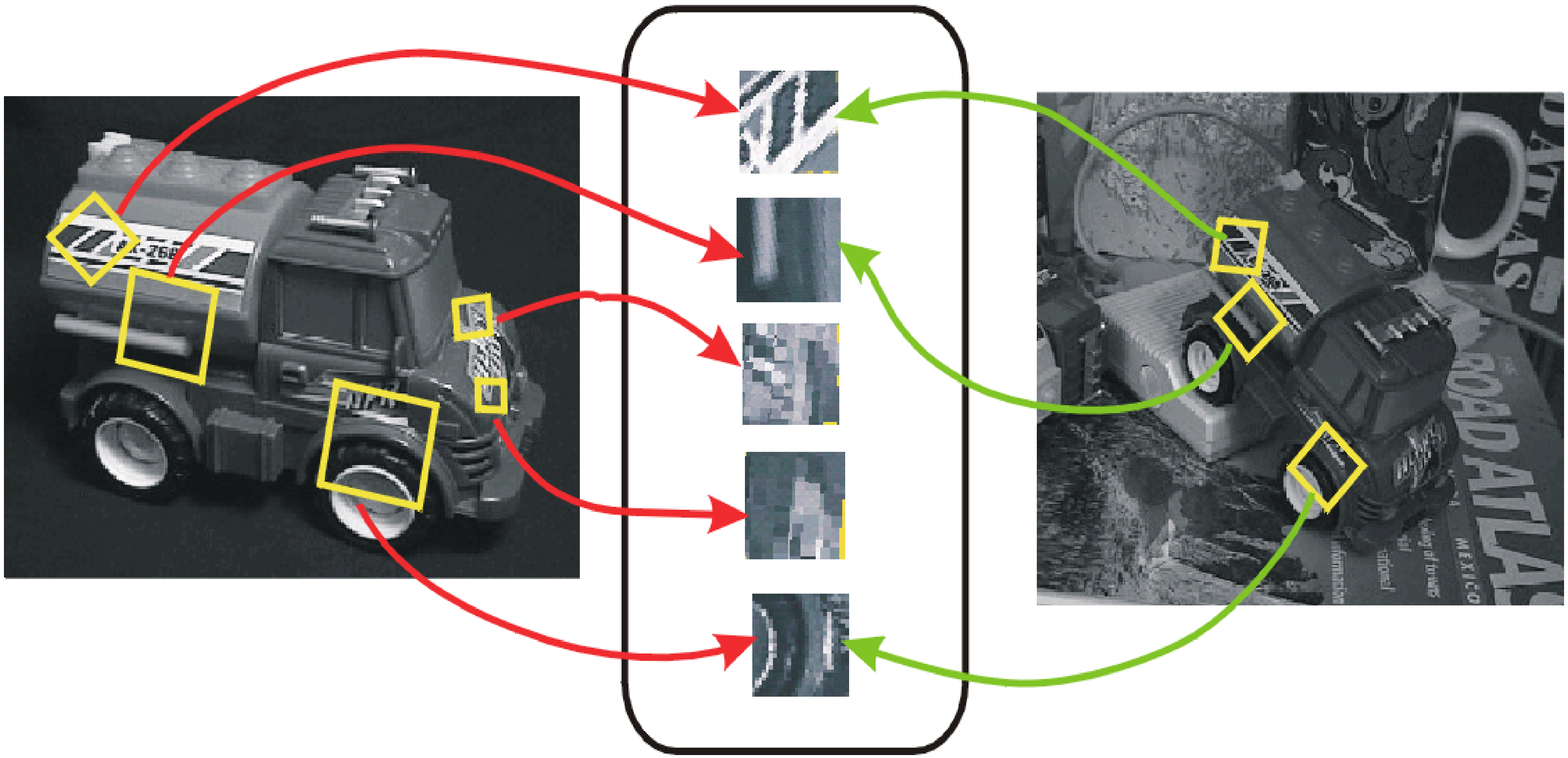
NASA Mars Rover images
with SIFT feature matches

Slide adapted from Steve Seitz

Invariant Local Features

Find features that are invariant to transformations

- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, …



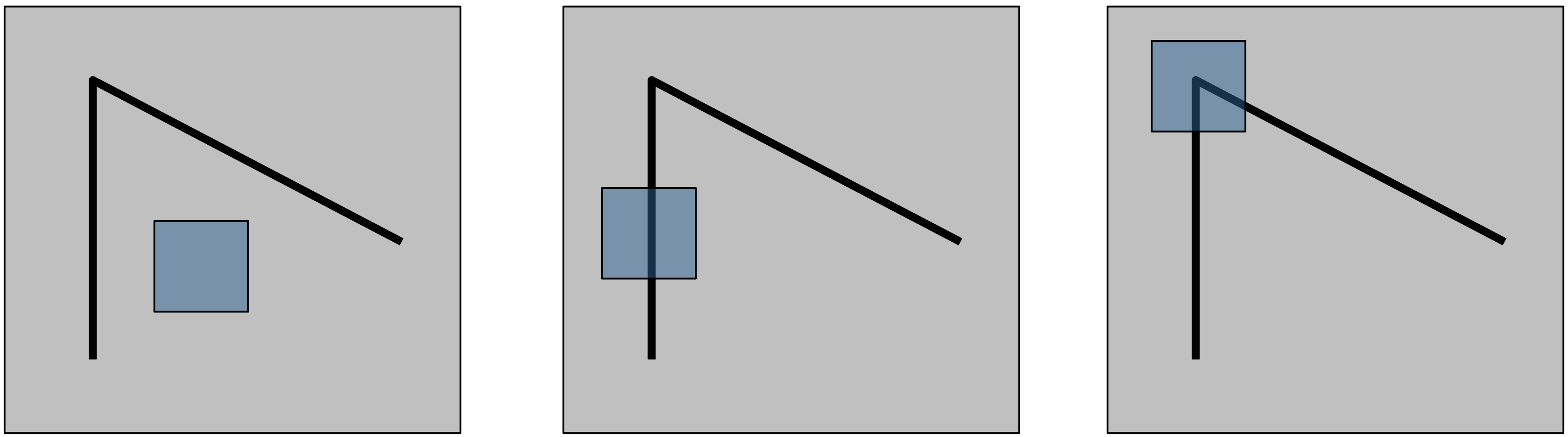
Feature Descriptors

Slide adapted from Steve Seitz

Local Measures of Uniqueness

Suppose we only consider a small window of pixels

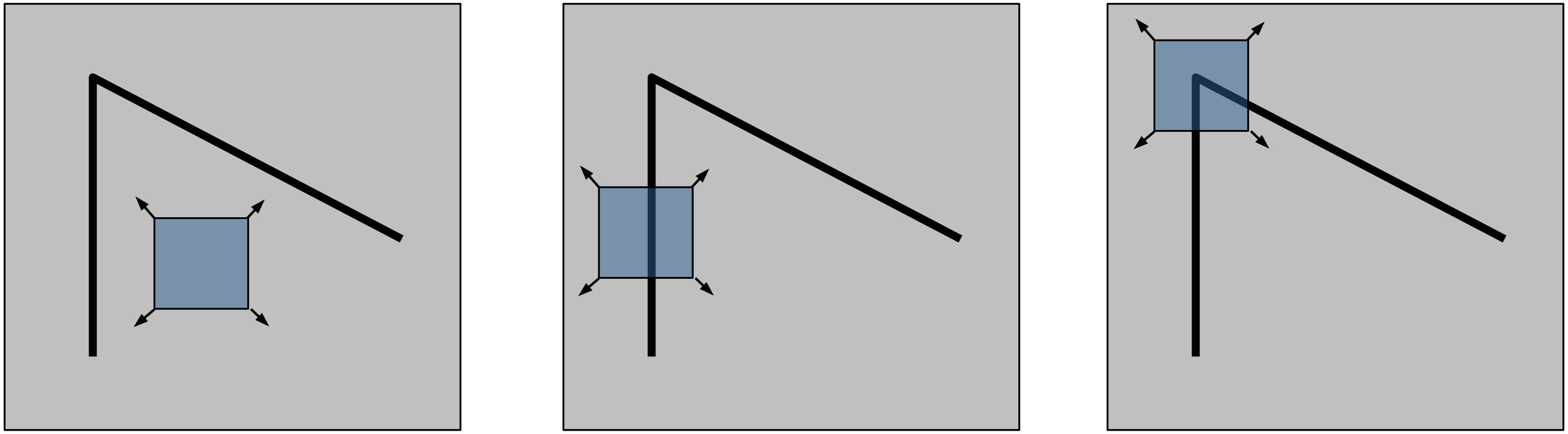
- What defines whether a feature is well localized and unique?



Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute

Local Measure of Uniqueness

- How does the window change when you shift by a small amount?



"flat" region:
no change in all directions

"edge":
no change along the edge direction

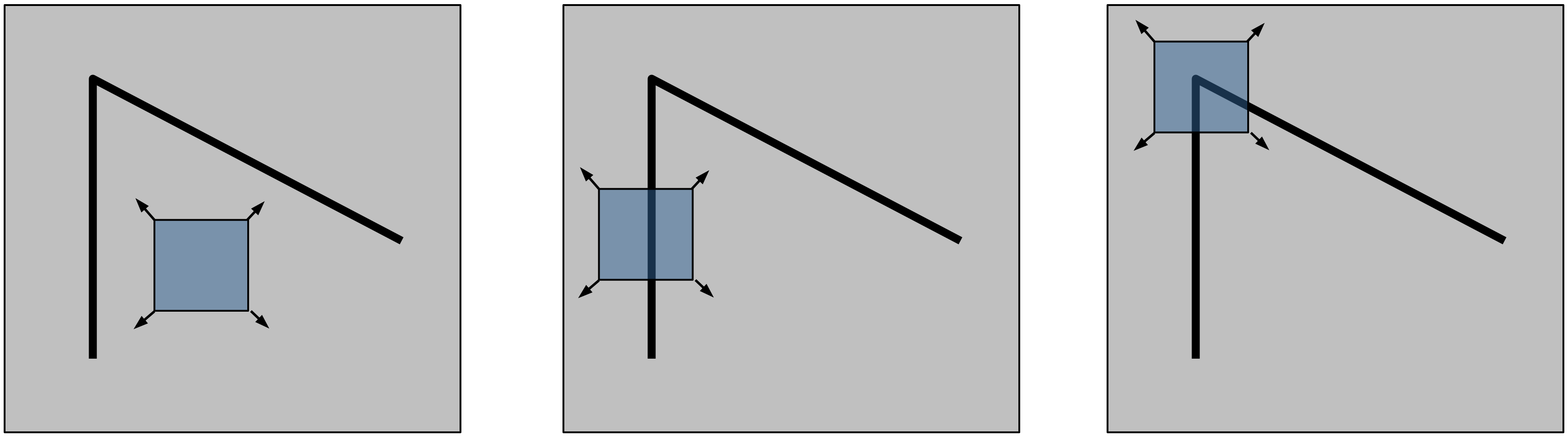
"corner":
significant change in all directions

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute

Locally Unique Features (Corners)

Define

E(u,v) = amount of change when you shift the window by (u,v)



E(u,v) is small
for all shifts

E(u,v) is small
for some shifts

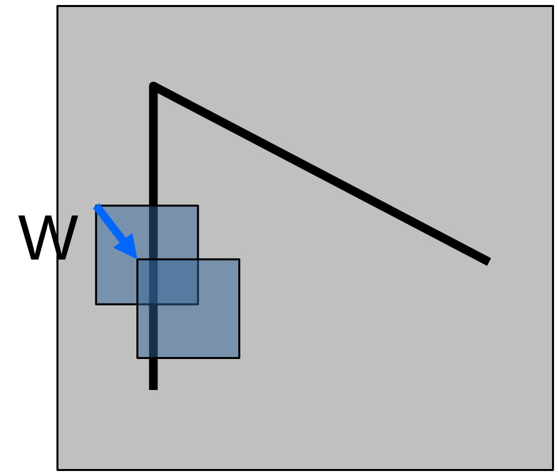
E(u,v) is small
for no shifts

Slide adapted from Steve Seitz

Corner Detection

Consider shifting the window W by (u,v)

- how do the pixels in W change?
- compare each pixel before and after by Sum of the Squared Differences (SSD)
- this defines an SSD "error" E(u,v):



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

Sum of Squared Differences (SSD)

Slide adapted from Steve Seitz

Small Motion Assumption

Taylor Series expansion of I:

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

If the motion (u,v) is small, then first order approx. is good

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

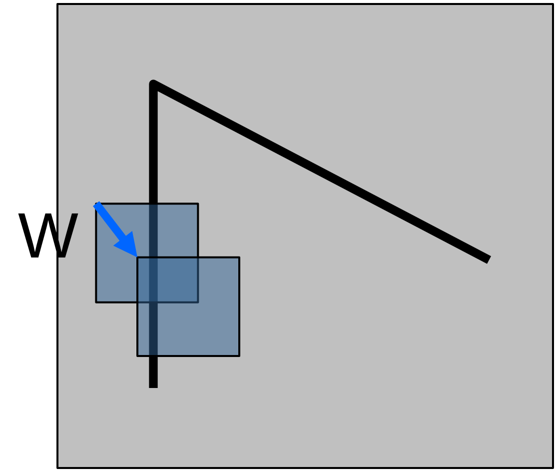
$$\approx I(x,y) + [I_x \;\; I_y] \begin{bmatrix} u \\ v \end{bmatrix}$$

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

Plugging this into the formula on the previous slide…

Slide adapted from Steve Seitz

Corner 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
- this defines an "error" of E(u,v):

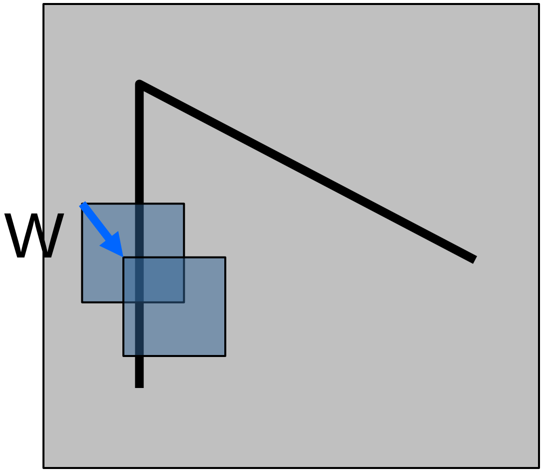
$$E(u,v) = \sum_{(x,y)\in W} \left[I(x+u, y+v) - I(x,y)\right]^2$$

Slide adapted from Steve Seitz

Corner 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
- this defines an "error" of E(u,v):



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

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

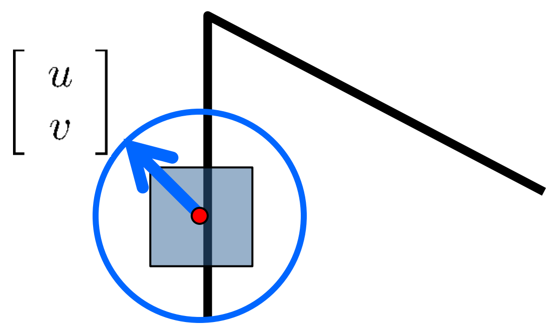
$$\approx \sum_{(x,y)\in W} \left[[I_x \; I_y] \begin{bmatrix} u \\ v \end{bmatrix}\right]^2$$

Slide adapted from Steve Seitz

Corner Detection

This can be rewritten:

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



For the example above

- You can move the center of the blue window to anywhere on the blue unit circle
- Which directions will result in the largest and smallest E values?

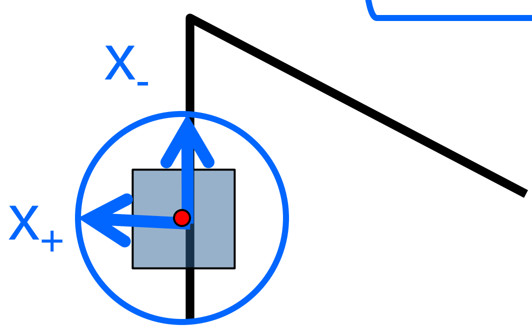
Slide adapted from Steve Seitz

Corner Detection

This can be rewritten:

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

H „structure tensor“



Eigenvalues and eigenvectors of H

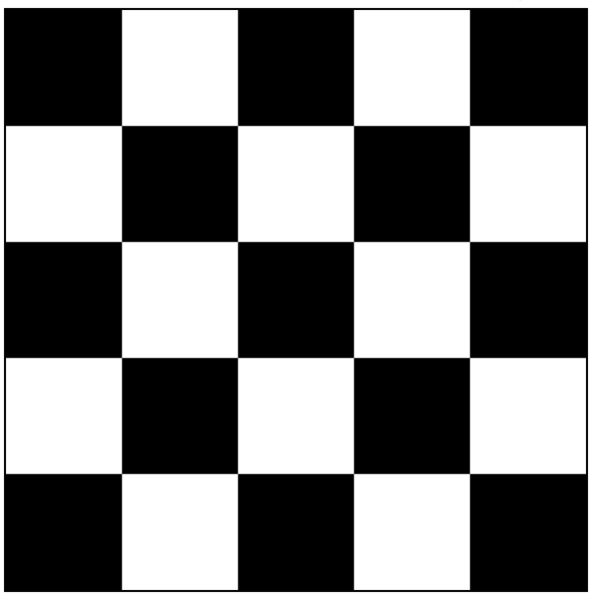
- Define shifts with the smallest and largest change (E value)
- x_+ = direction of largest increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of smallest increase in E.
- λ_- = amount of increase in direction x_-

$$\begin{aligned} Hx_+ &= \lambda_+ x_+ \\ Hx_- &= \lambda_- x_- \end{aligned}$$

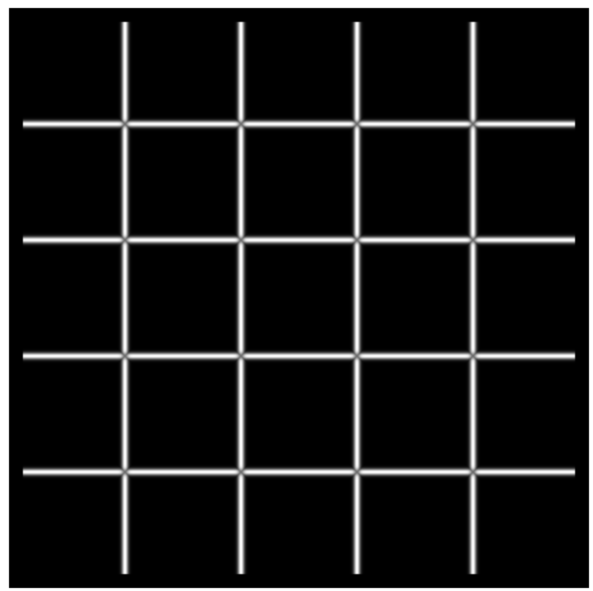
Slide adapted from Steve Seitz

Corner Detection Recipe

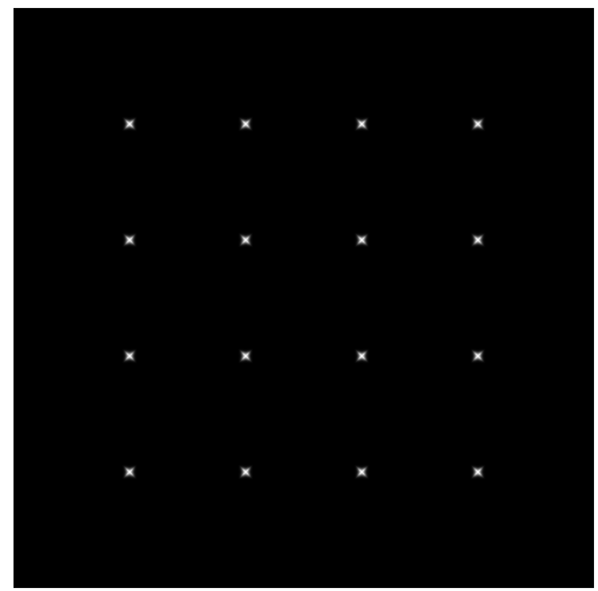
- Compute the gradient at each point in the image
- Create the H matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features



I



λ_+

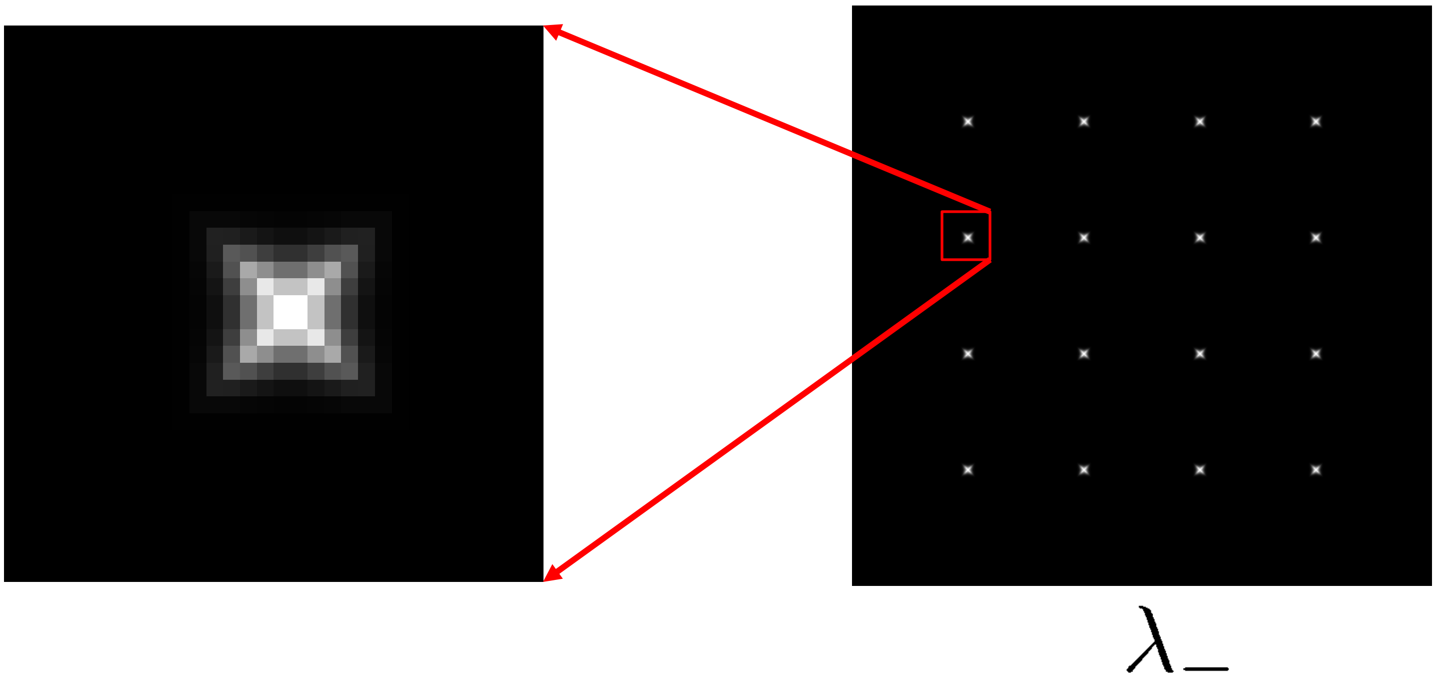


λ_-

Slide adapted from Steve Seitz

Corner Detection Recipe

- 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 Steve Seitz

Harris Operator

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

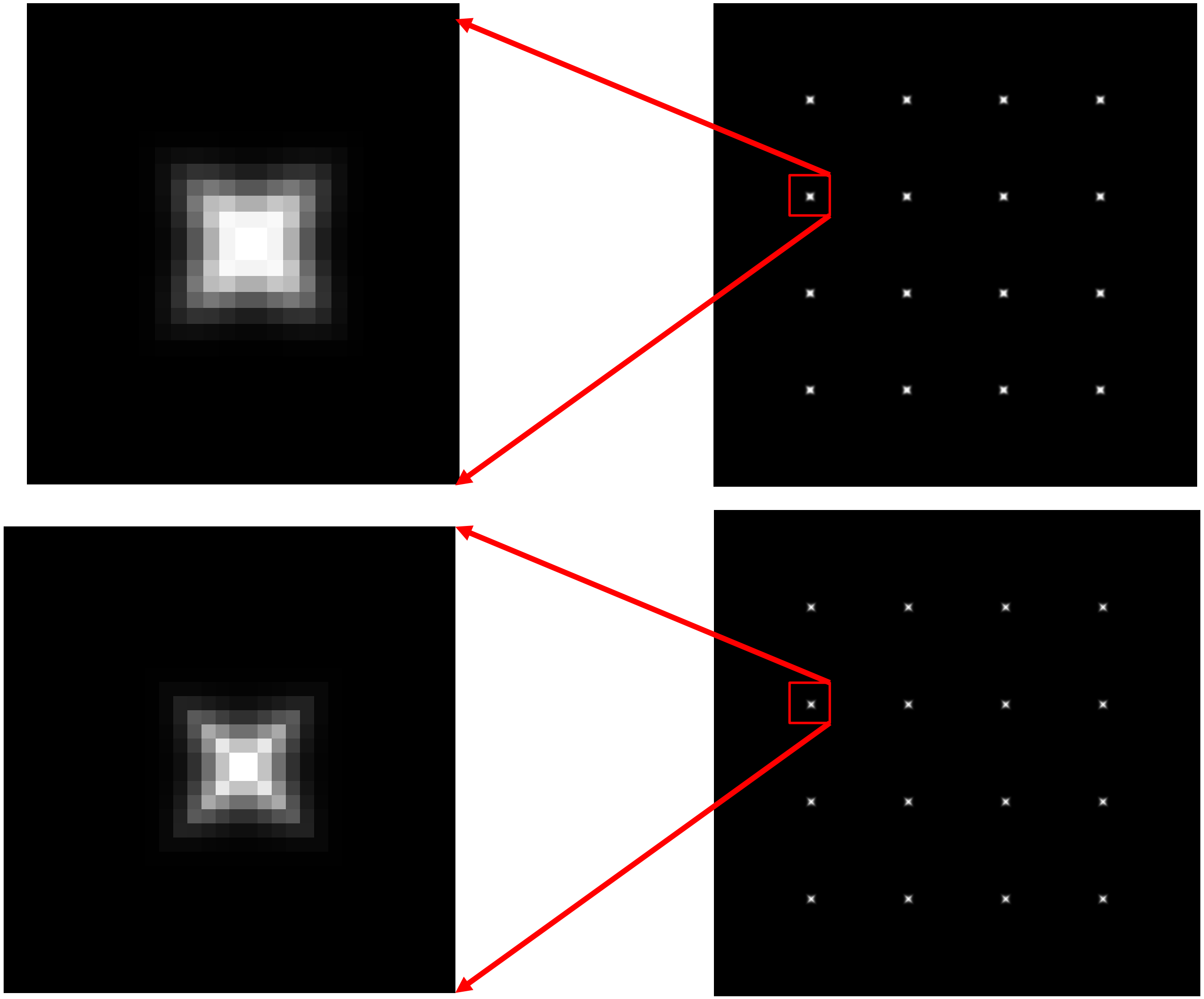
$$f = \frac{\lambda_- \lambda_+}{\lambda_- + \lambda_+}$$

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

  - 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 Steve Seitz

Harris Operator



Harris operator

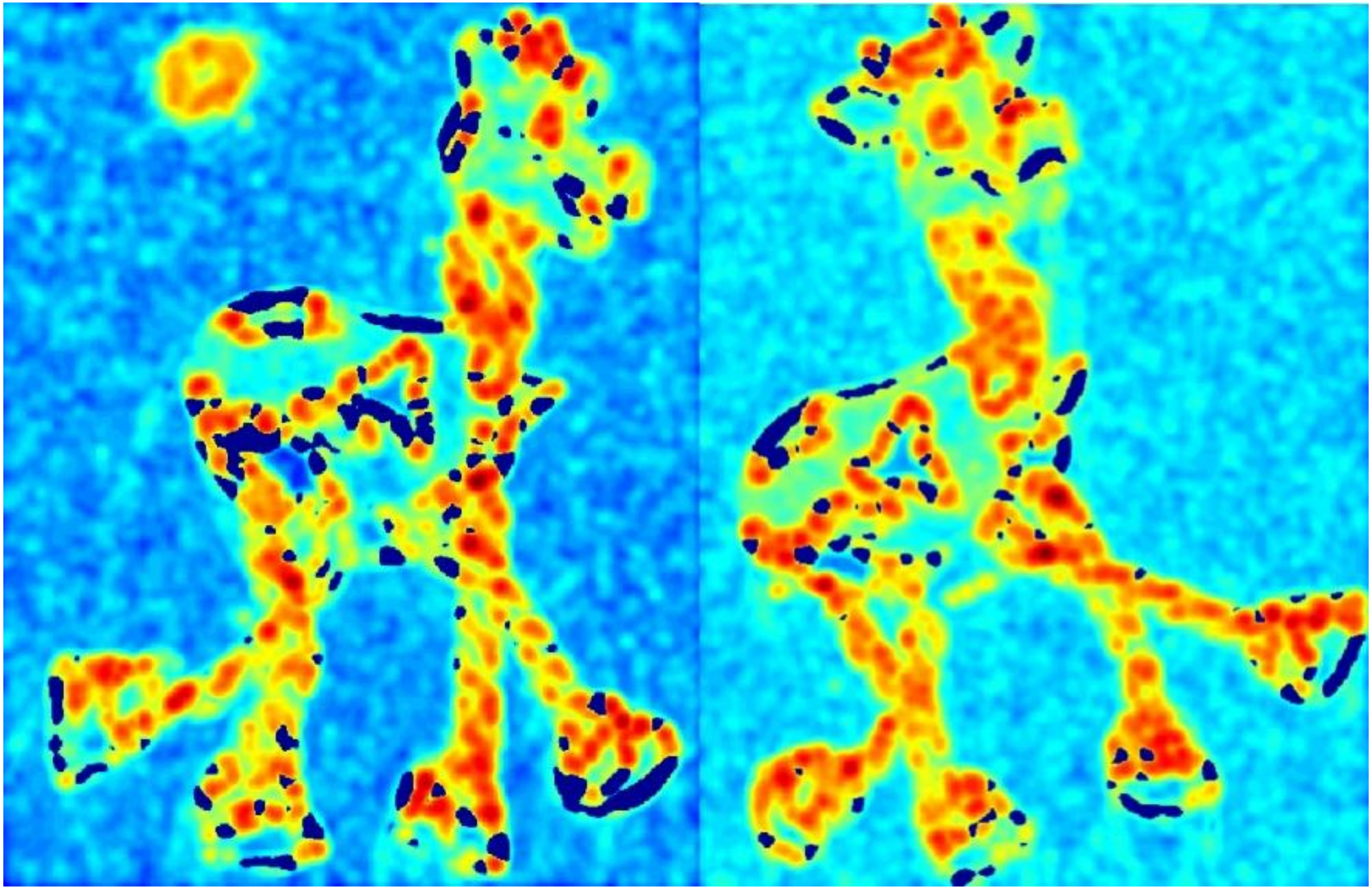
λ_-

Slide adapted from Steve Seitz

Harris Detector Example

Slide adapted from Steve Seitz

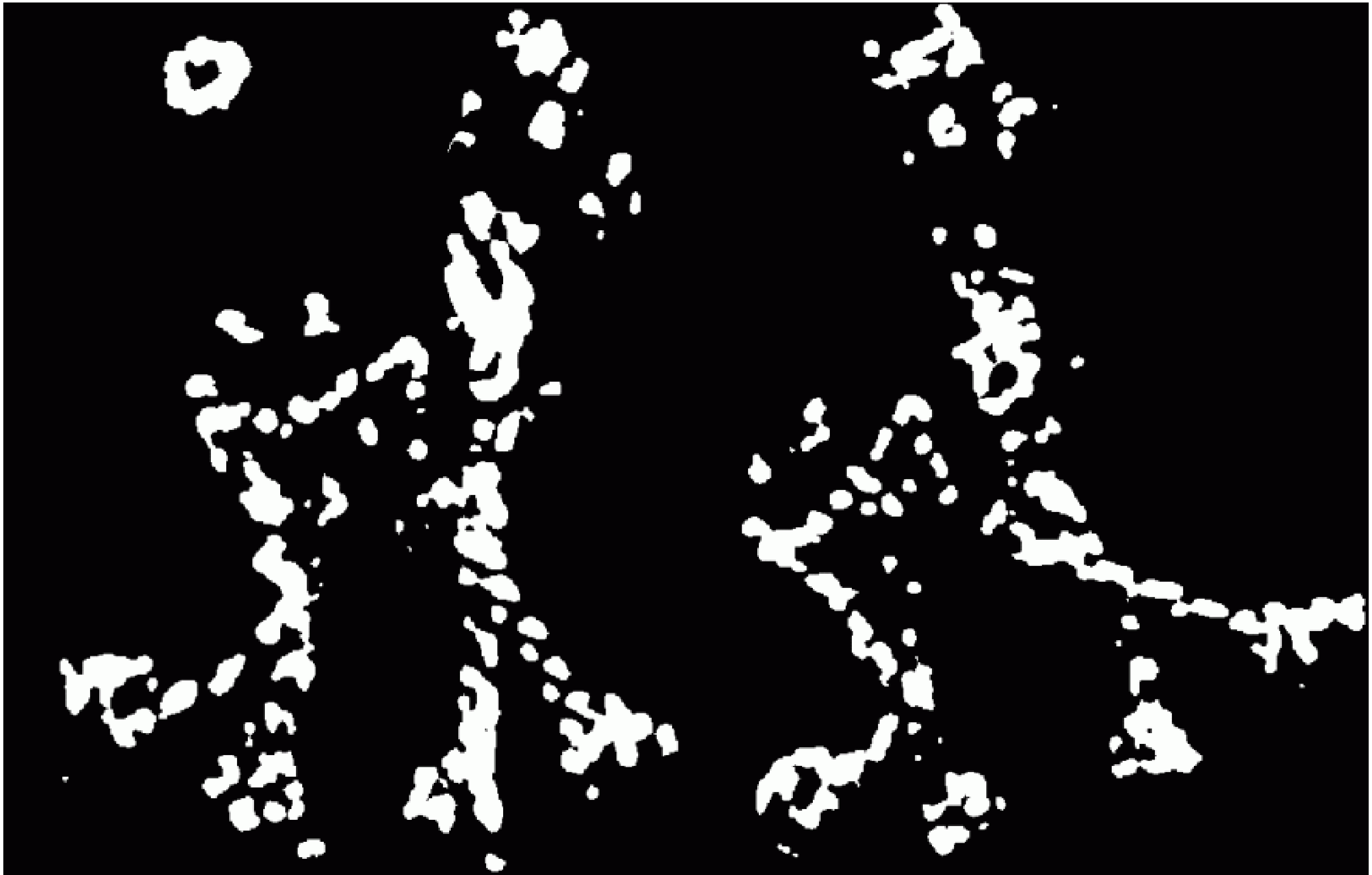
Harris Corner Response



(red high, blue low)

Slide adapted from Steve Seitz

Thresholded Harris Corner Response



Slide adapted from Steve Seitz

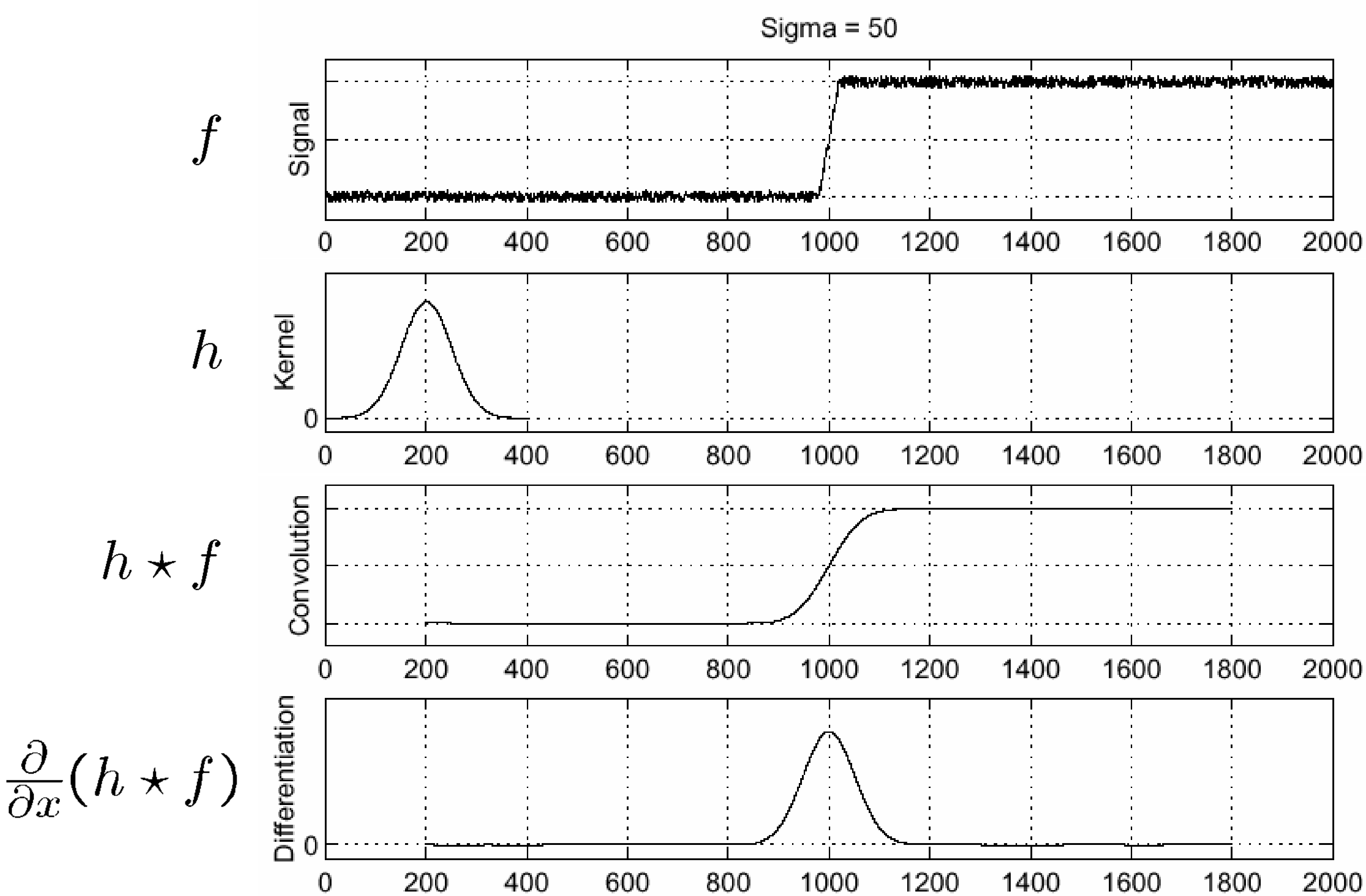
Local Maxima of Harris Corner Response

Slide adapted from Steve Seitz

Harris Corners

Slide adapted from Steve Seitz

Edge Detection



Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Slide adapted from Silvio Savarese

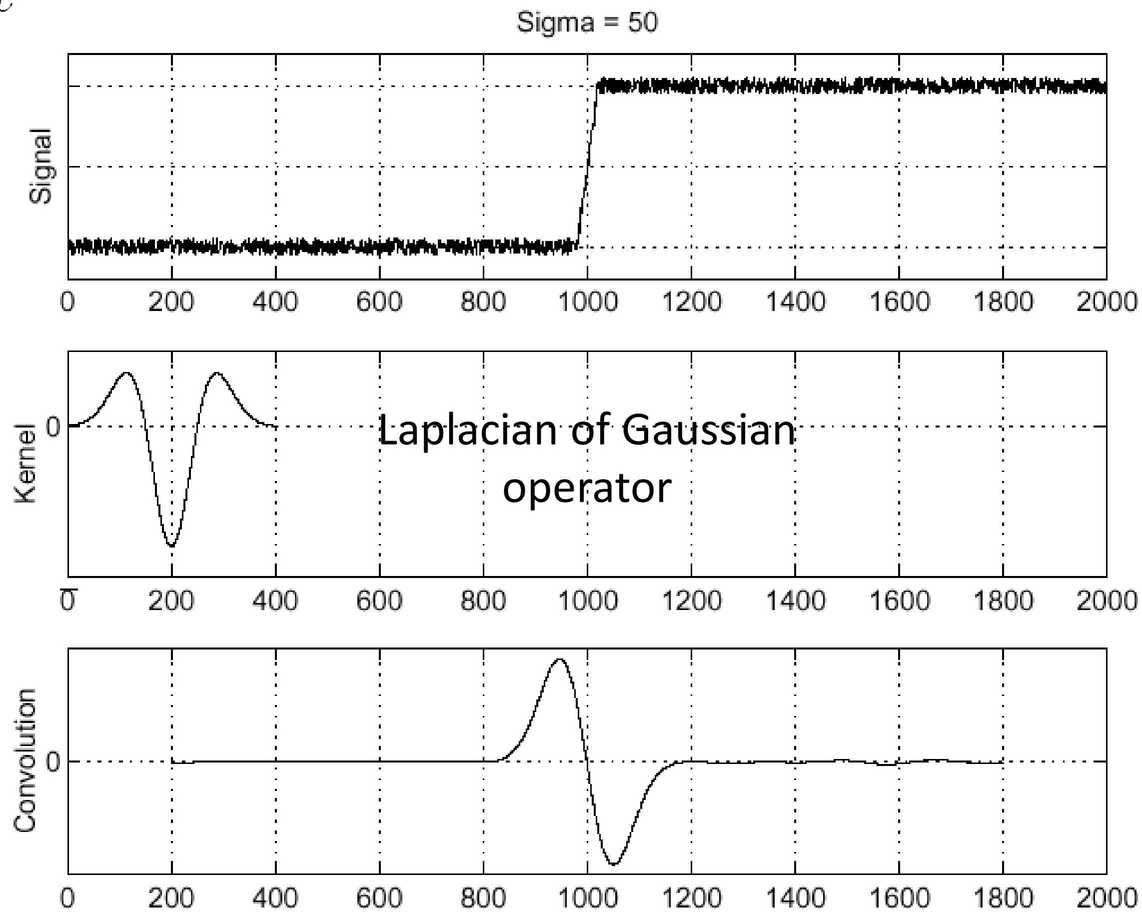
Laplacian of Gaussian

- Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

f

$\frac{\partial^2}{\partial x^2}h$

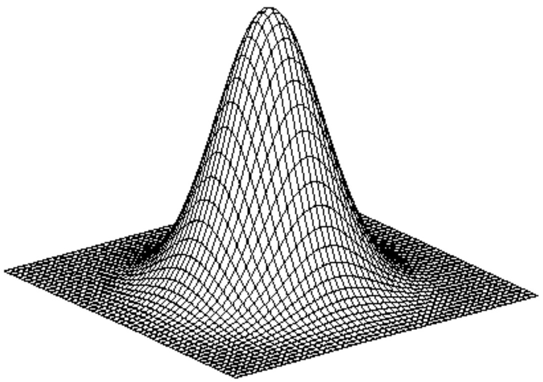
$(\frac{\partial^2}{\partial x^2}h) \star f$

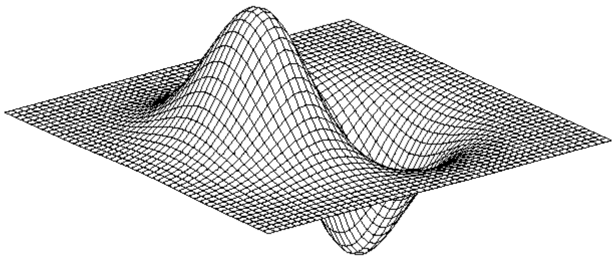


Where is the edge? Zero-crossings of bottom graph

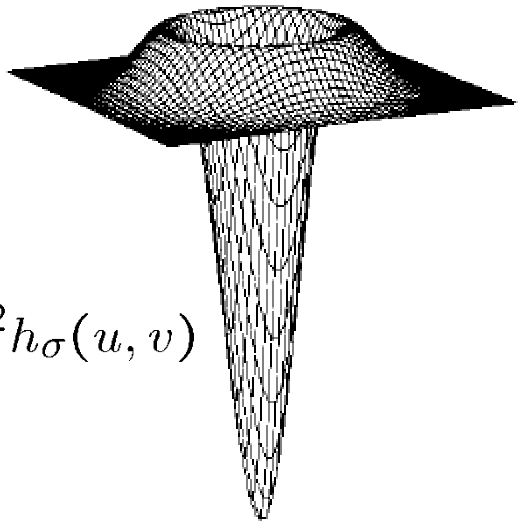
Slide adapted from Silvio Savarese

Laplacian of Gaussian in 2D





Laplacian of Gaussian



Gaussian

derivative of Gaussian

$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

$$\nabla^2 h_\sigma(u,v)$$

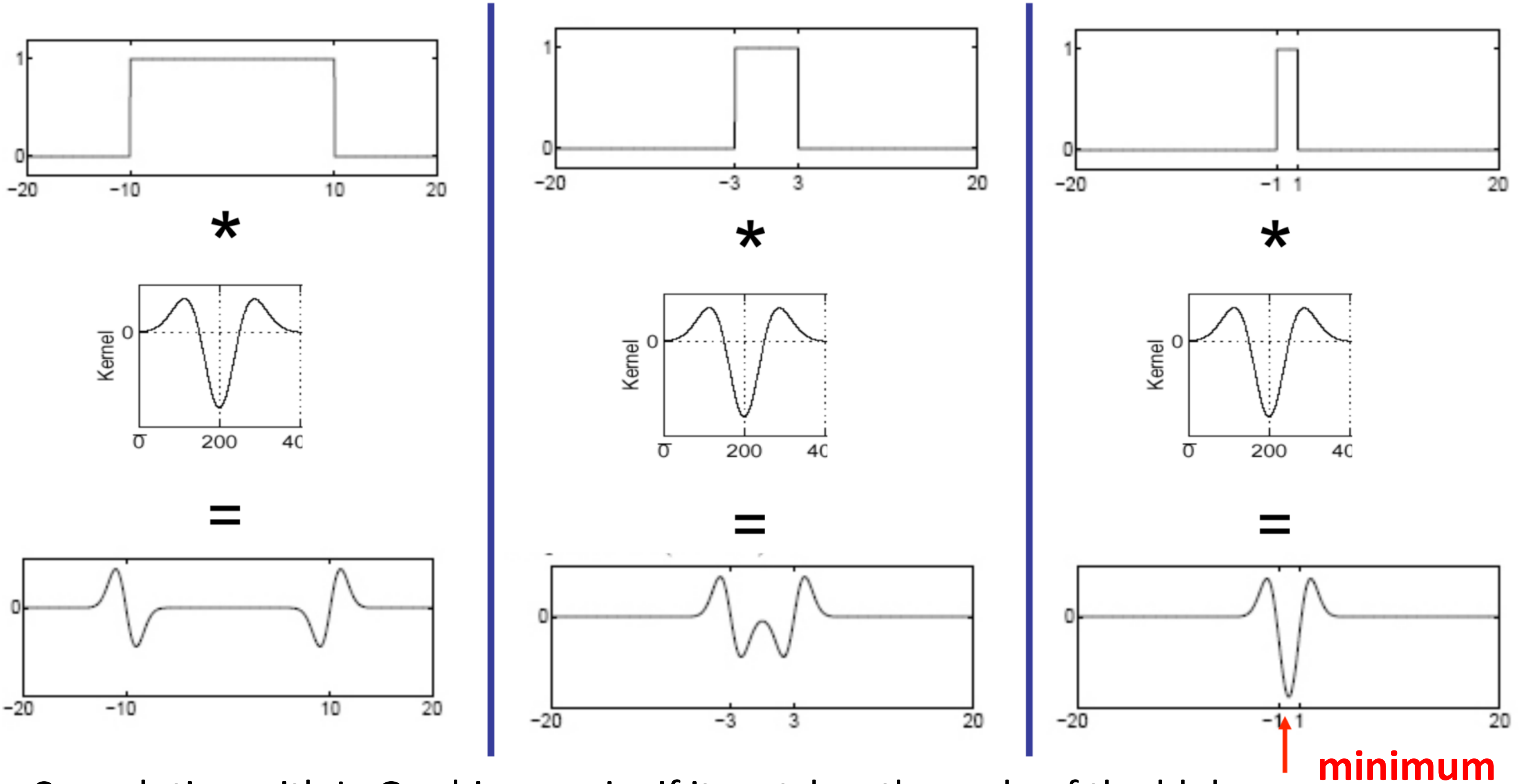
∇^2 is the Laplacian operator:

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

Slide adapted from Steve Seitz

Blob Detection in 1D

- Can we use the Laplacian of Gaussian (LoG) to find blobs?

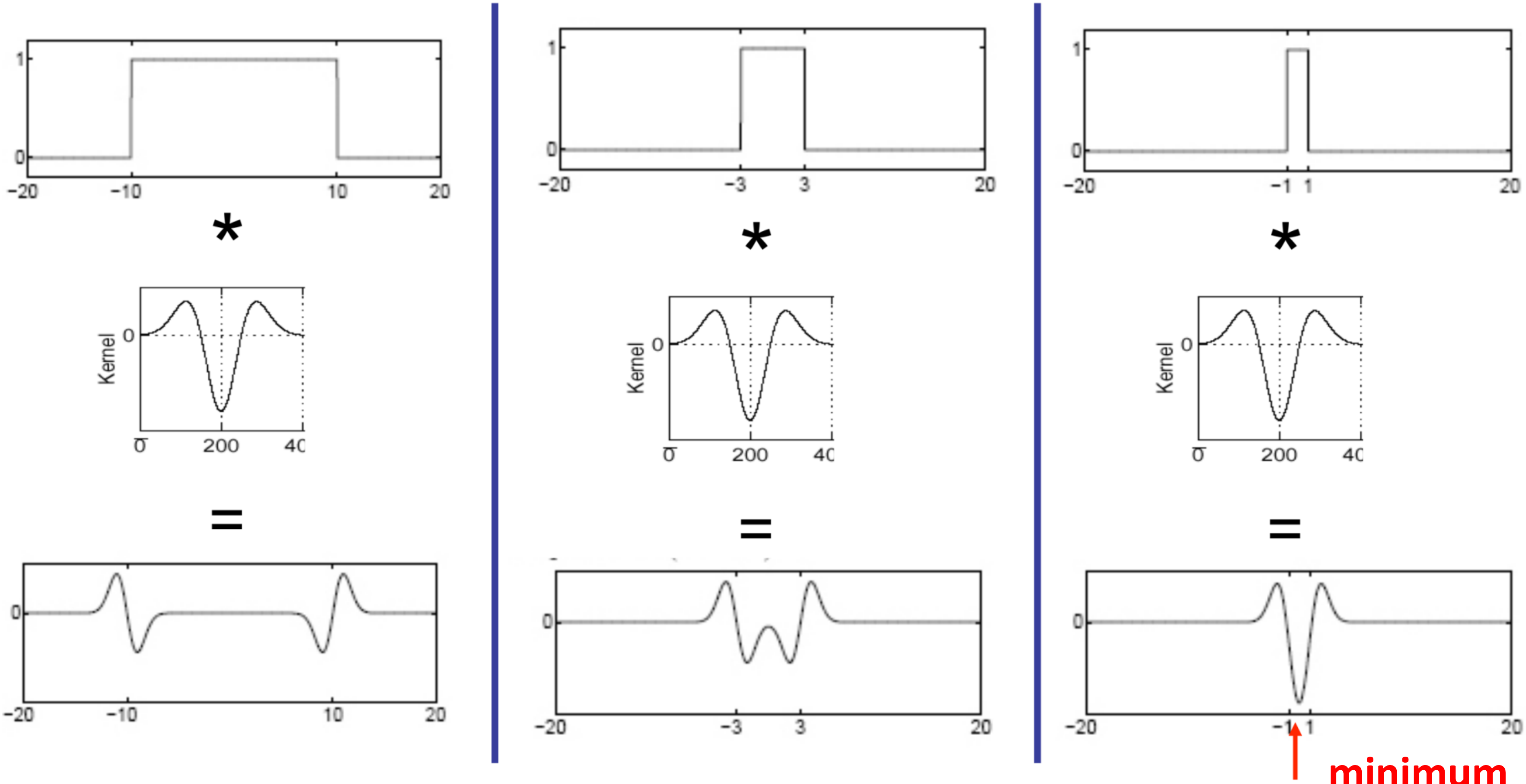


Convolution with LoG achieves min. if it matches the scale of the blob

Slide adapted from Silvio Savarese

Blob Detection in 1D

- How can we detect blobs at many different scales?

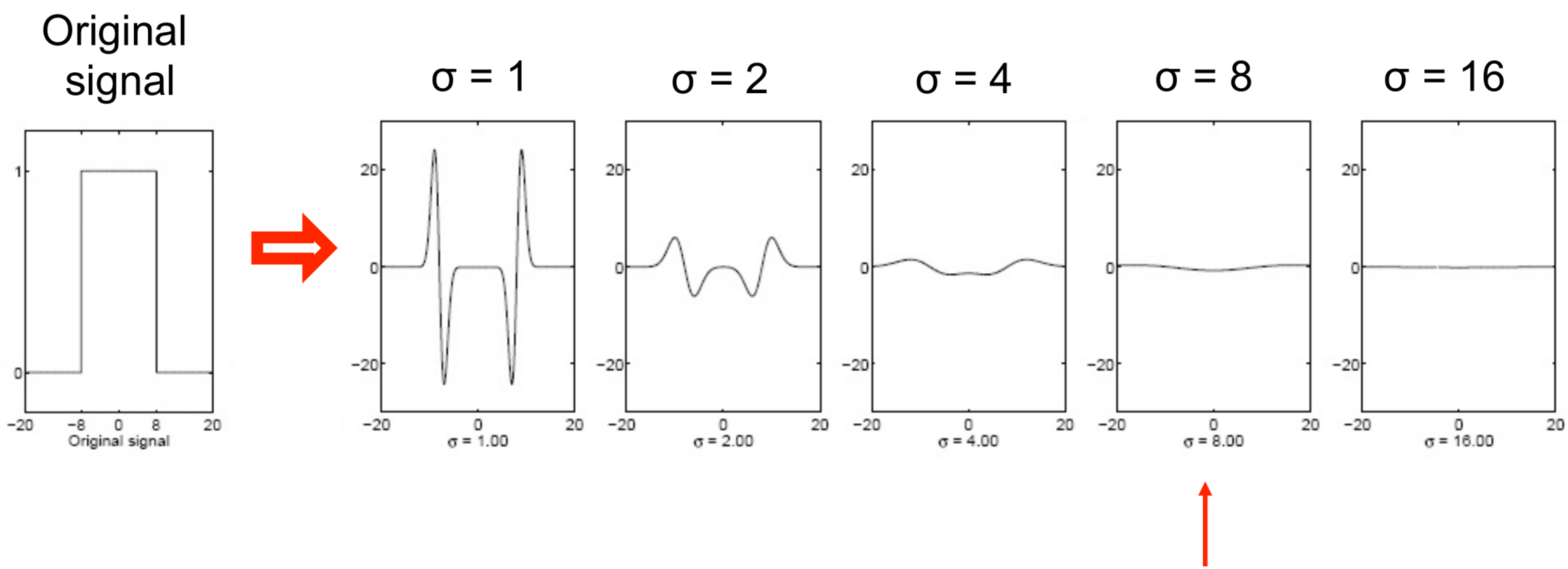


Idea: convolve the image with LoGs of different scales

Slide adapted from Silvio Savarese

Scale-Space Blob Detection in 1D

- How can we detect blobs at many different scales?



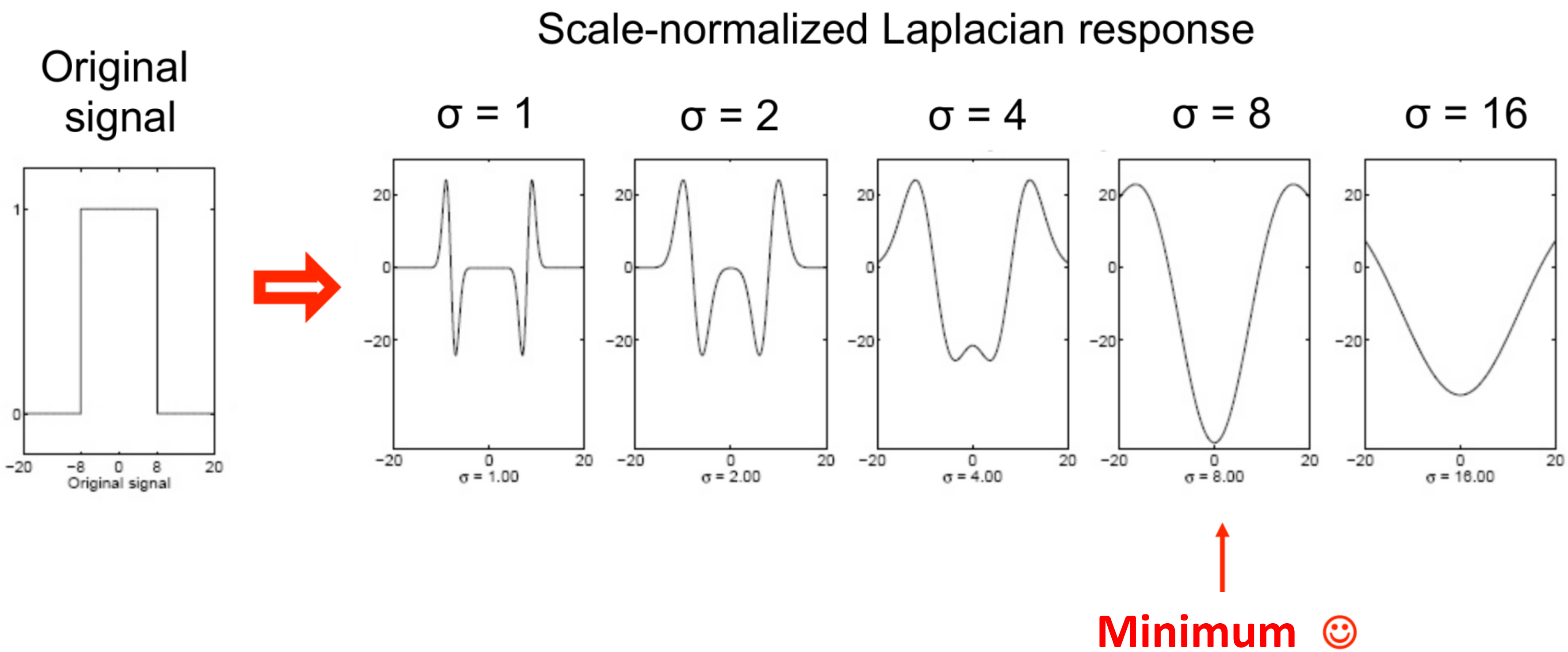
This should give the max response ☹

What is wrong here?

Slide adapted from Silvio Savarese

Characteristic Scale

- We need to scale-normalize the LoG operator so that the energy of the convolved signal remains the same
- Multiply LoG operator with σ^2

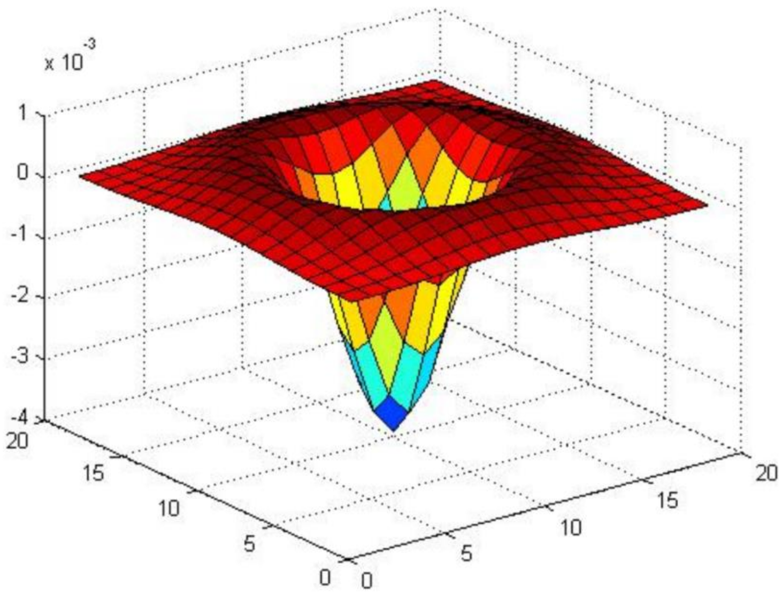


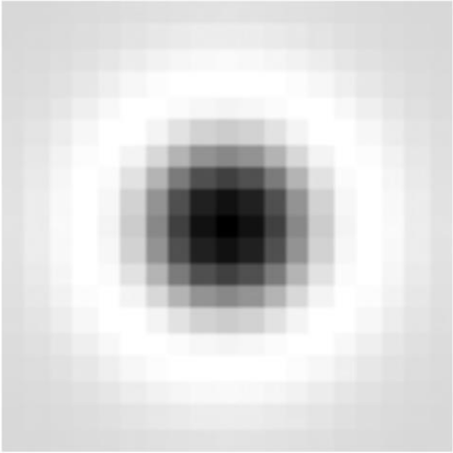
The convolved signal attains a minimum at its characteristic scale

Slide adapted from Silvio Savarese

Scale-Normalized LoG in 2D

- Laplacian of Gaussian in 2D
- Circular symmetric operator



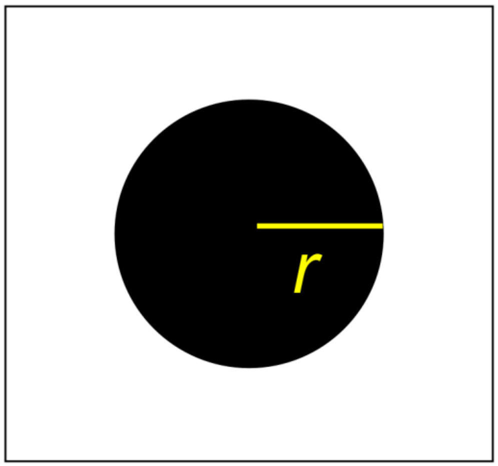


$$\mathrm{LoG}_{norm}(u,v) = \sigma^2 \nabla^2 h_\sigma(u,v)$$

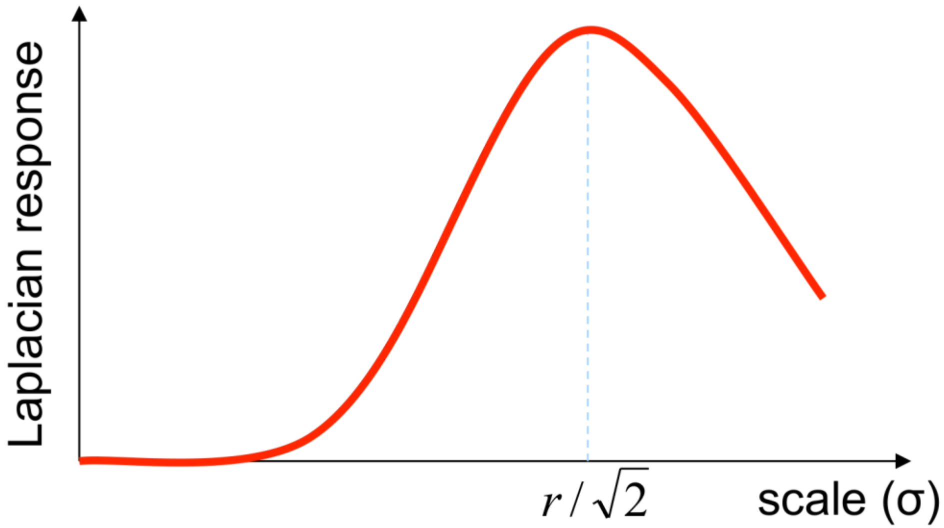
Slide adapted from Silvio Savarese

Scale Selection

- For a circle of radius r, convolution with scale-normalized LoG attains a maximum response at $\sigma = \frac{r}{\sqrt{2}}$



image



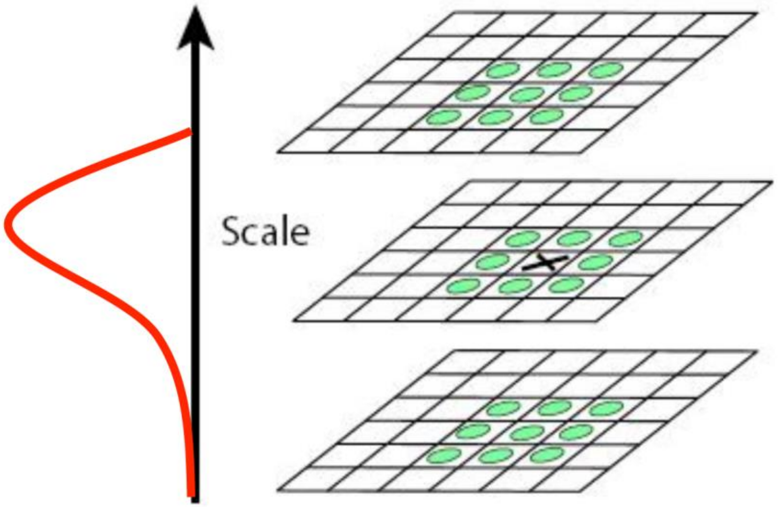
Slide adapted from Silvio Savarese

Scale-Space Blob Detection

- Convolve image with scale-normalized LoG of different neighboring scales
- Find maxima of squared Laplacian response along the spatial and the scale dimension

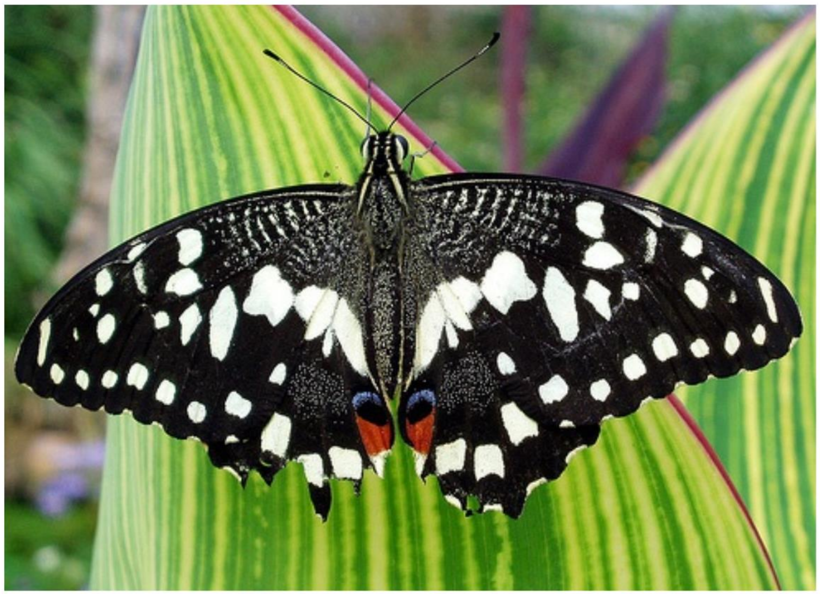
- SIFT: approximate LoG with Difference of Gaussians

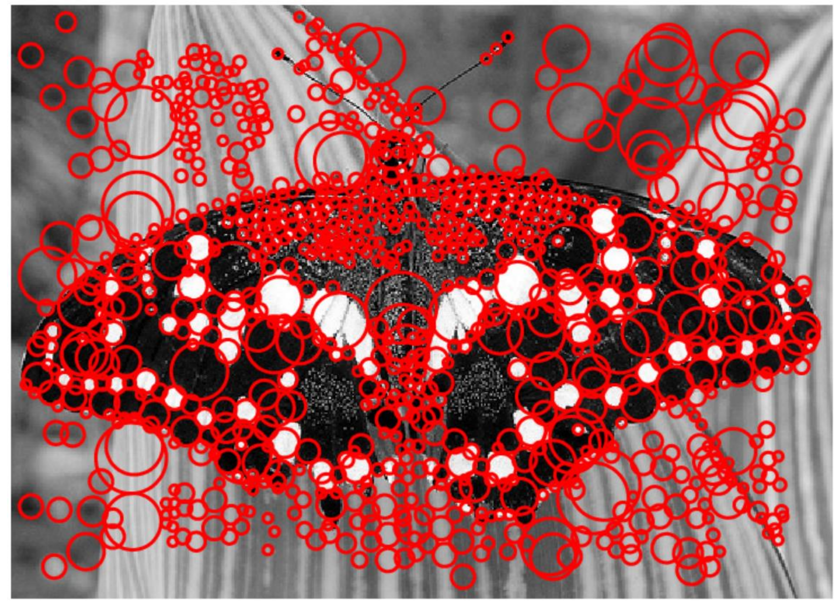
- SURF: approximate LoG with Haar features (box filters / rect. filters)



Slide adapted from Silvio Savarese

Blob Detection Example

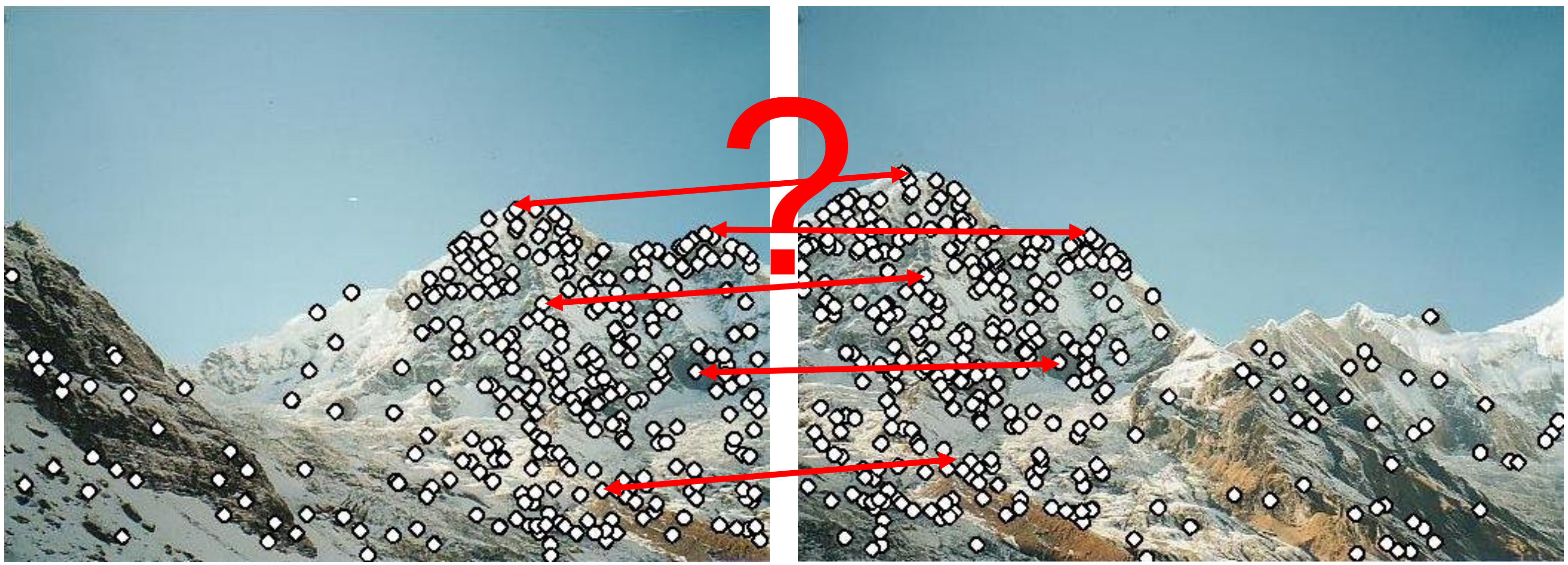




Slide adapted from Silvio Savarese

Keypoint Descriptors

- We know how to detect good points
- Next question: How to match them?

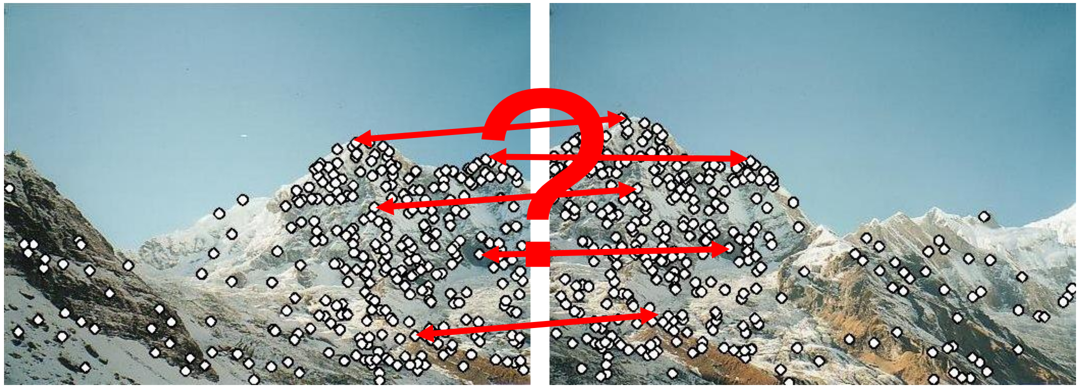


- Idea: extract distinctive descriptor vector from a local patch around the keypoint

Slide adapted from Steve Seitz

Invariance

- Goal: match keypoints regardless of image transformation
  - This is called transformational invariance
- Most keypoint detection and description 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



Slide adapted from Steve Seitz

Invariant Detection and Description

- Make sure your detector is invariant
  - Harris and LoG/DoG are invariant to translation and rotation
  - Scale is trickier
    - Scale selection for blobs (f.e. SIFT)
    - Keypoints at multiple scales for same location
- 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
    - What’s this invariant to?
  - Let’s look at some better approaches…

Slide adapted from Steve Seitz

2D Rotation Invariance

- Idea: align the descriptor with a dominant 2D orientation
- Example approach: Use the eigenvector of H corresponding to larger eigenvalue

Figure by Matthew Brown

Slide adapted from Steve Seitz

Scale Invariant Feature Transform (SIFT)

- Take 16x16 square window around detected feature at the optimal scale
- Compute gradient (orientation/angle and magnitude)
- Create histogram of gradient angle (weighted by magnitude)
  - Using 36 bins
- Any peak within 80% of the highest peak creates a feature

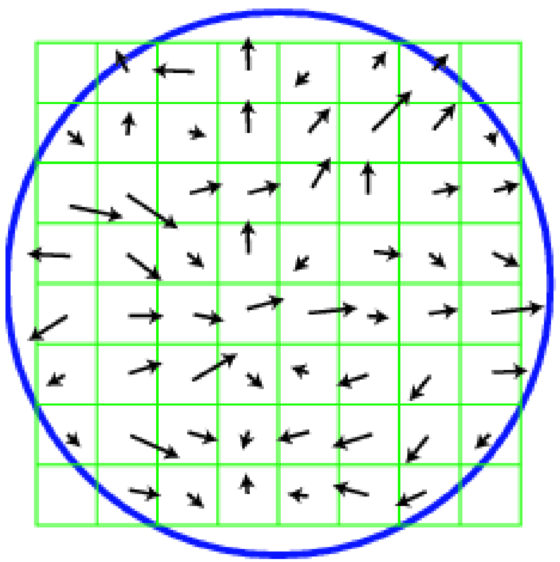
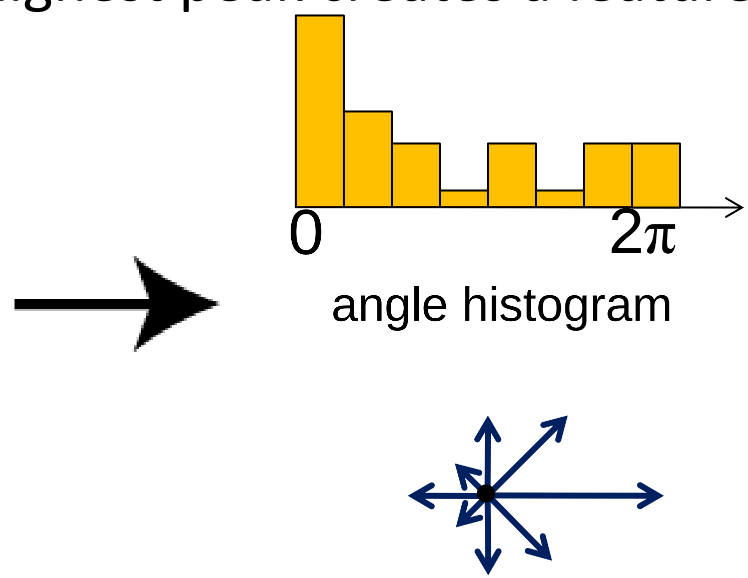


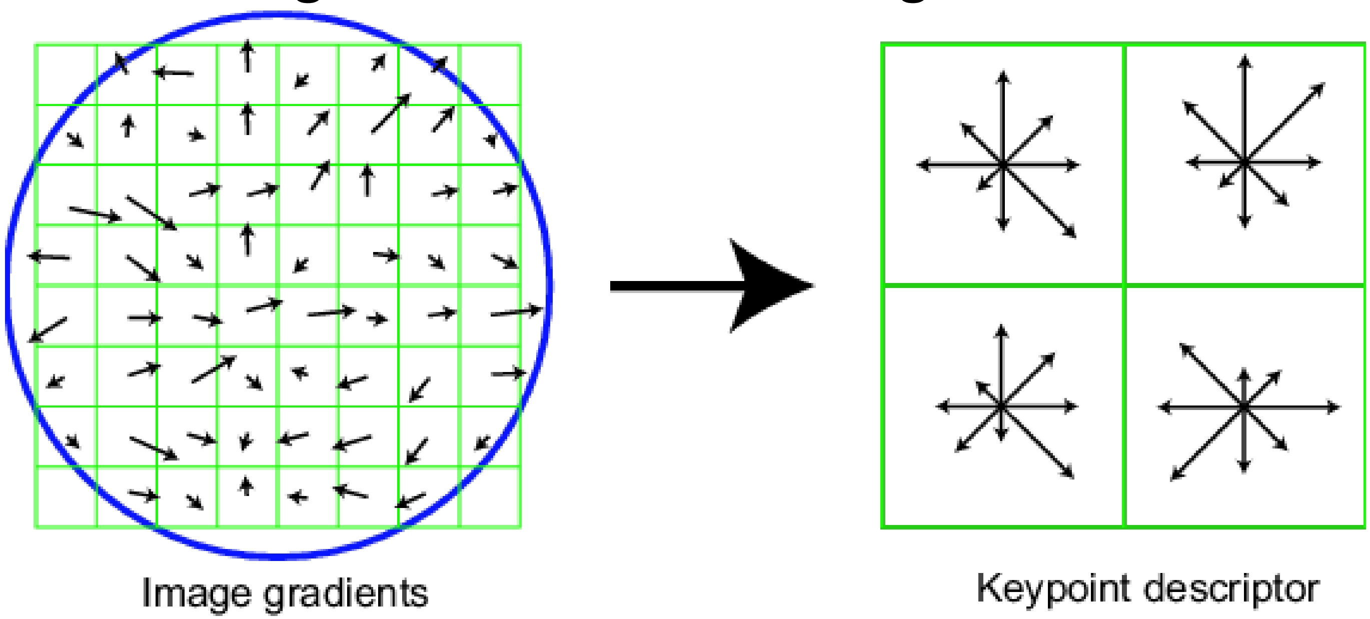
Image gradients



Slide adapted from David Lowe

SIFT Descriptor

- 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
- Normalize histogram and threshold magnitude



Slide adapted from David Lowe

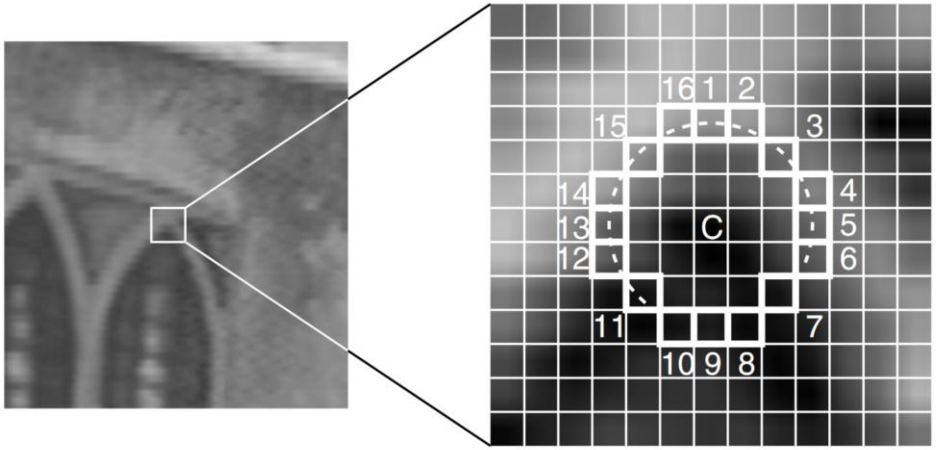
SURF

- Speeded Up Robust Features
- Approximates LoG and descriptor calculation in SIFT using Haar wavelets
  - Faster computation
  - Similar performance like SIFT

Bay, Tuytelaars, Van Gool, Speeded Up Robust Features, ECCV 2006

FAST Detector

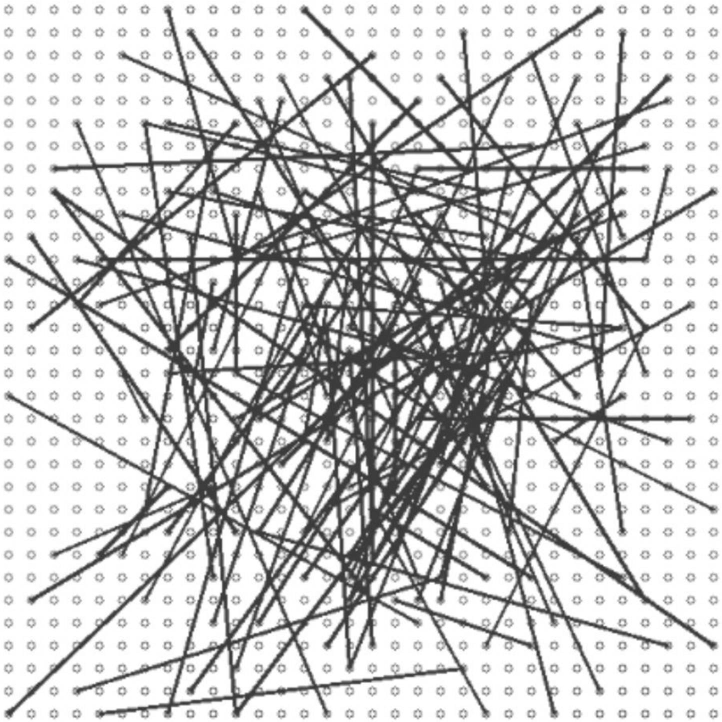
- Features from
  Accelerated Segment Test
- Check relation of brightness values to center pixel along circle
- Specific number of contiguous pixels brighter or darker than center
- Very fast corner detection



Rosten, Drummond, Fusing Points and Lines for High Performance Tracking, ICCV 2005

BRIEF Descriptor

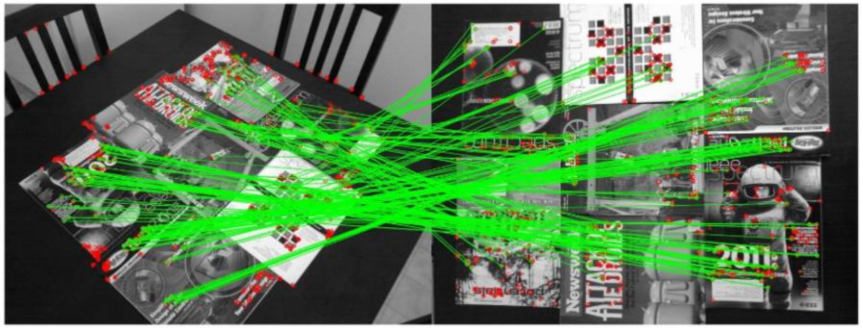
- Binary Robust Independent Elementary Features
- Binary descriptor from intensity comparisons at sample positions
- Very efficient to compute
- Fast matching distance through Hamming distance

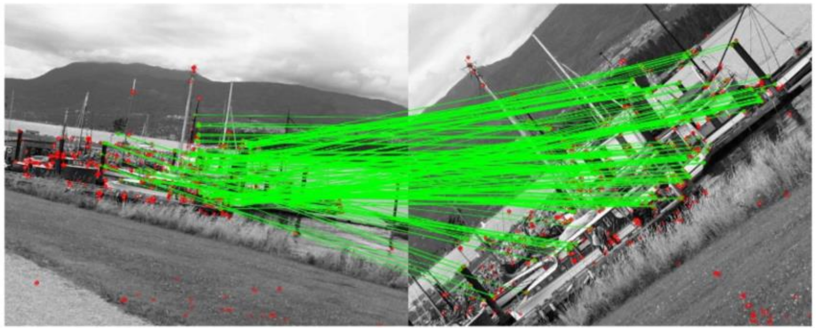


Calonder, Lepetit, Strecha, Fua, BRIEF: Binary Robust Independent Elementary Features, ECCV’10

ORB Descriptor

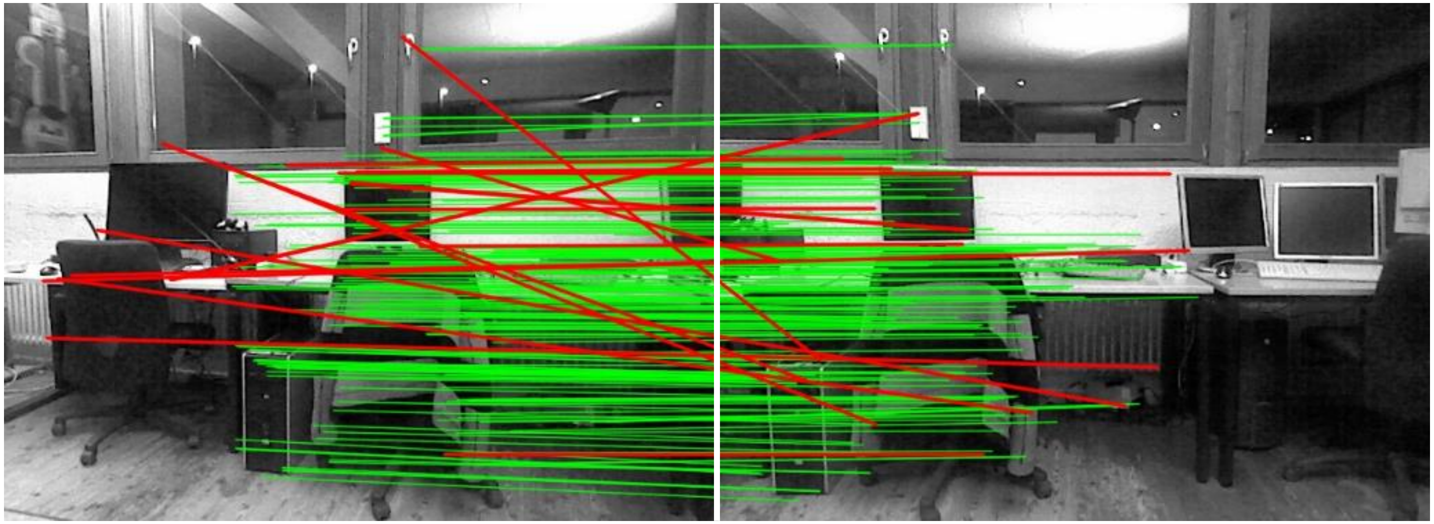
- Oriented Fast and Rotated BRIEF
  - Combination of FAST detector and BRIEF descriptor
  - Rotation-invariant BRIEF: Estimate dominant orientation from patch moments
- Very popular for VO





Rublee, Rabaud, Konolige, Bradski, ORB: an efficient alternative to SIFT or SURF, ICCV 2011

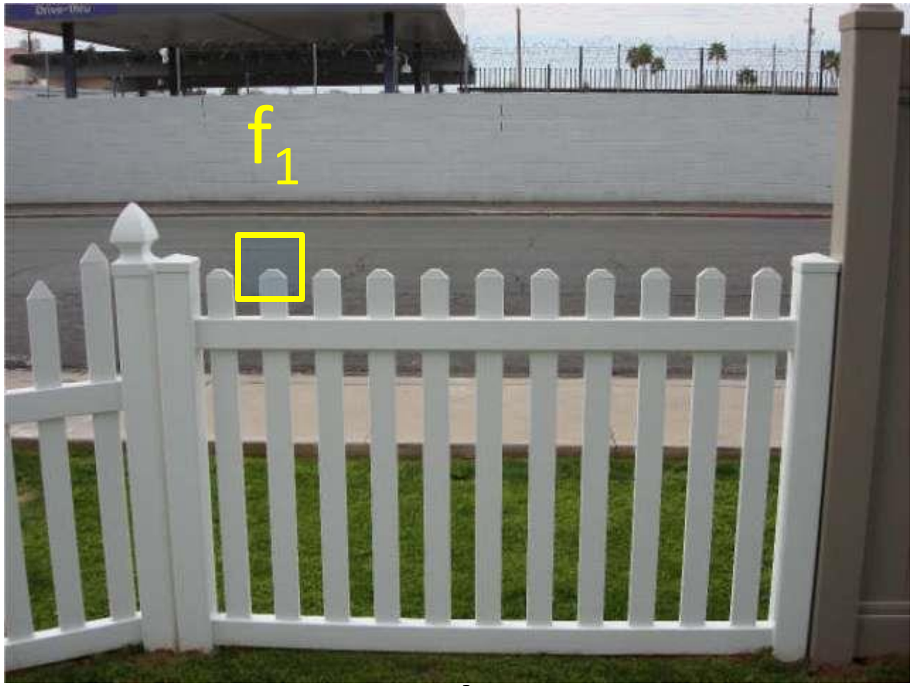
Keypoint Matching



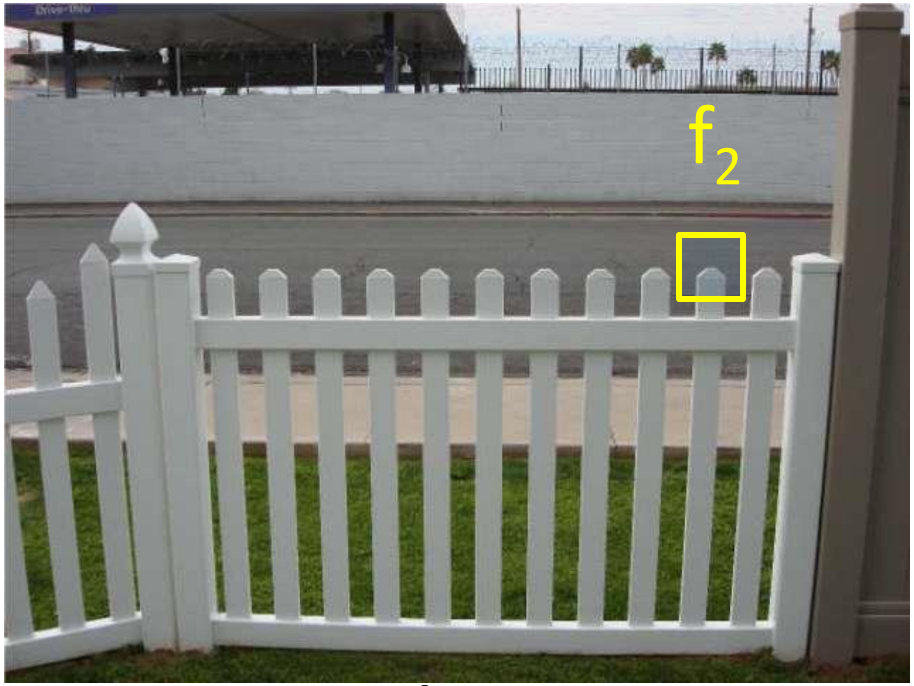
- Match keypoints with similar descriptors

Matching Distance

- How to define the difference between two descriptors f1, f2?
- Simple approach is to assign keypoints with minimal sum of square differences SSD(f1, f2) between entries of the two descriptors



I_1

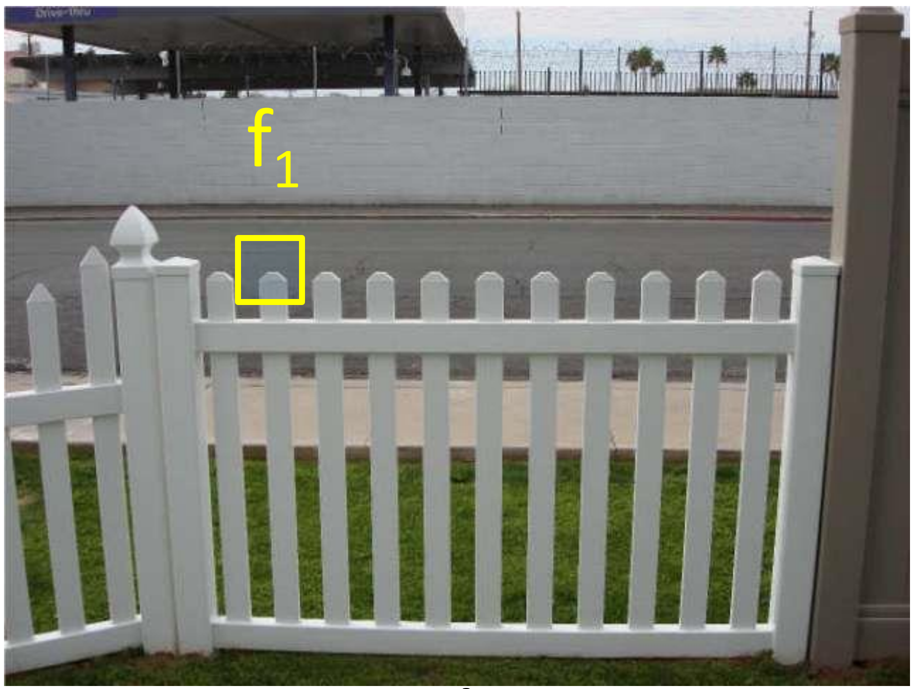


I_2

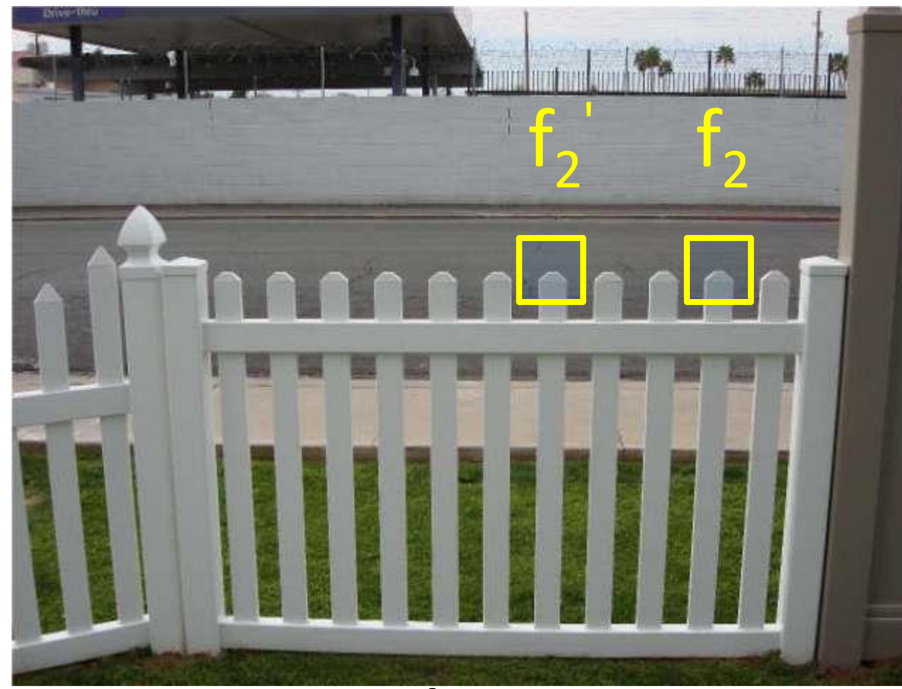
Slide adapted from Steve Seitz

Matching Distance

- Better approach (Lowe's distance ratio):
  best to second best ratio distance = SSD(f1, f2) / SSD(f1, f2')
  - f2 is best SSD match to f1 in I2
  - f2' is 2nd best SSD match to f1 in I2



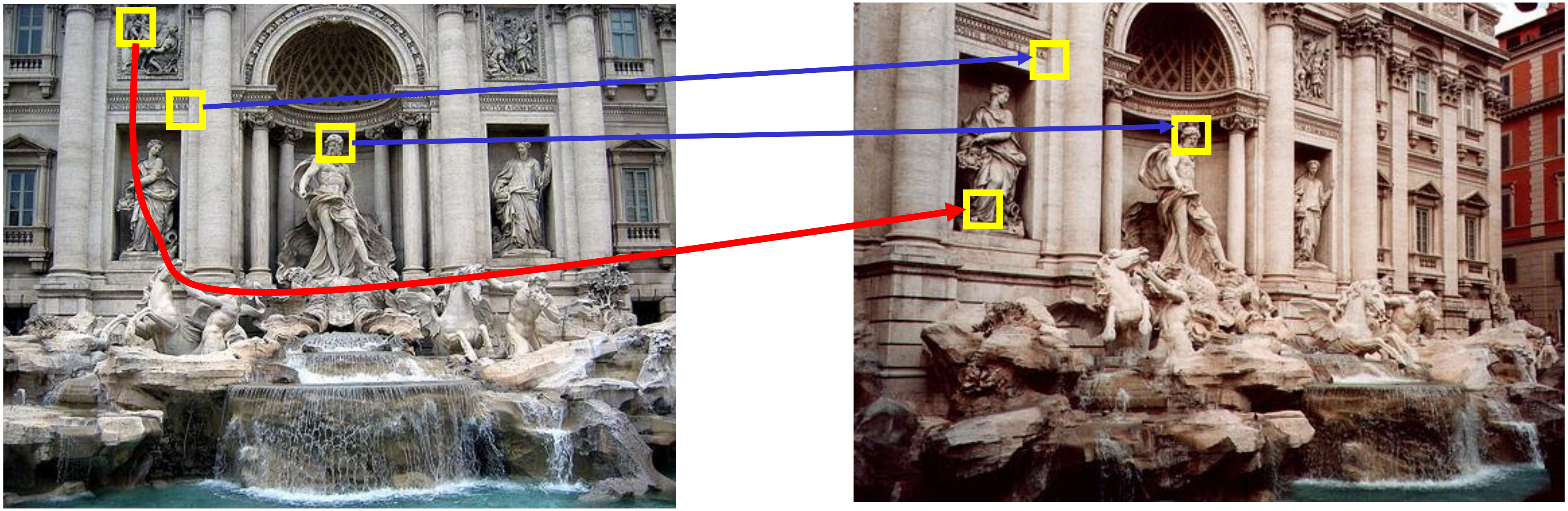
I_1



I_2

Slide adapted from Steve Seitz

Eliminating Bad Matches

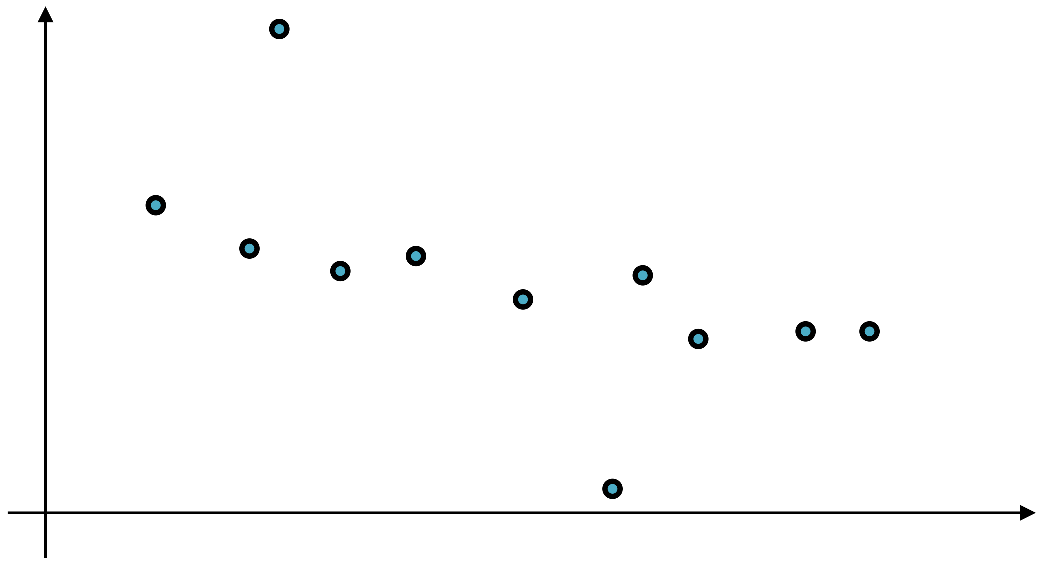


- Only accept matches with distance smaller a threshold
- Choice of threshold affects performance
  - Too restrictive: less false positives (#false matches) but also less true positives (#true matches)
  - Too lax: more true positives but also more false positives
- What else can we do?

Slide adapted from Steve Seitz

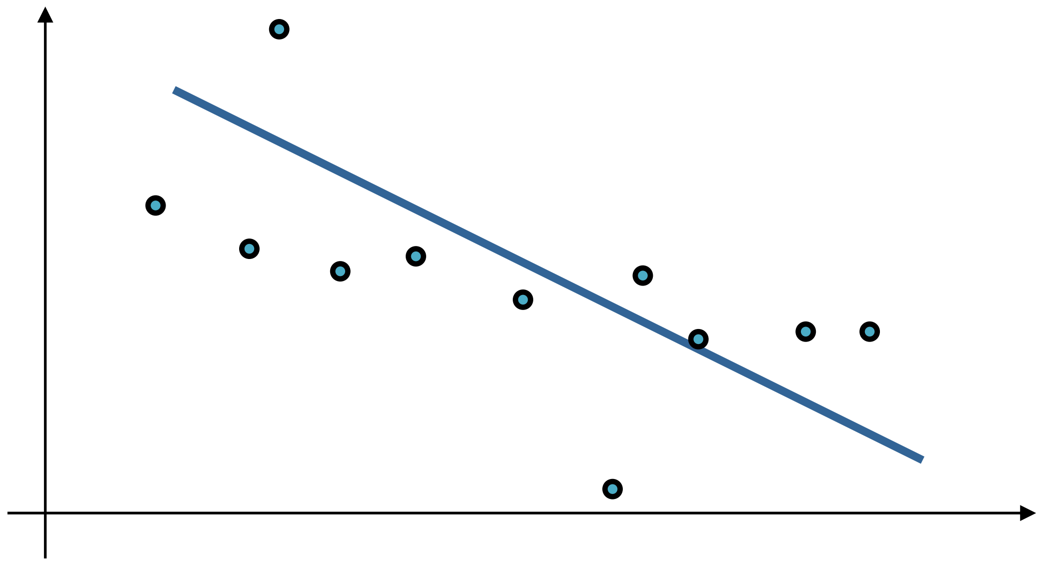
Random Sample Consensus (RANSAC)

- Model fitting in presence of noise and outliers
- Example: fitting a line through 2D points



RANSAC

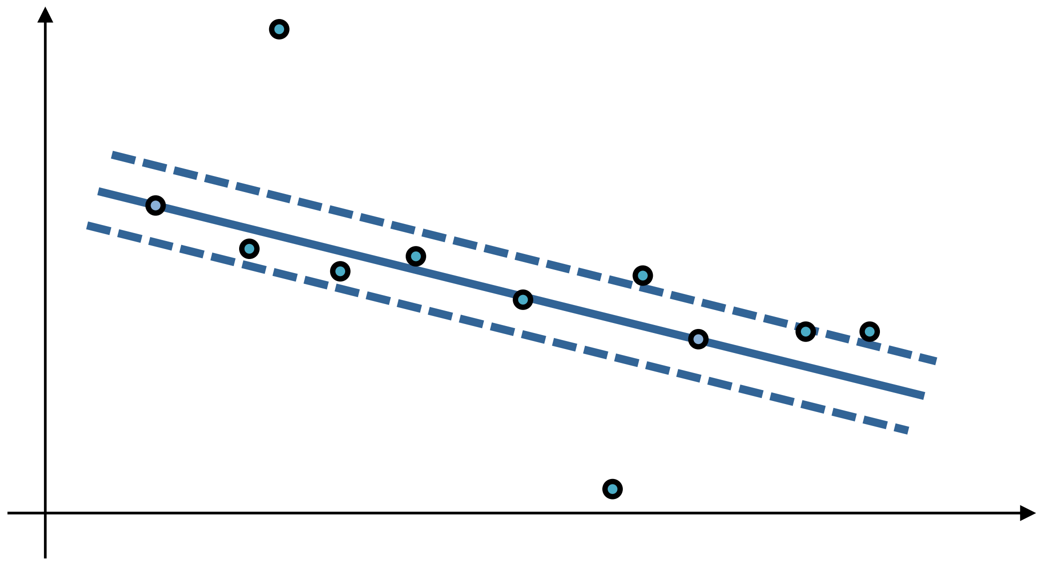
- Least-squares solution, assuming constant noise for all points



Bad!

RANSAC

- We only need 2 points to fit a line. Let's try 2 random points

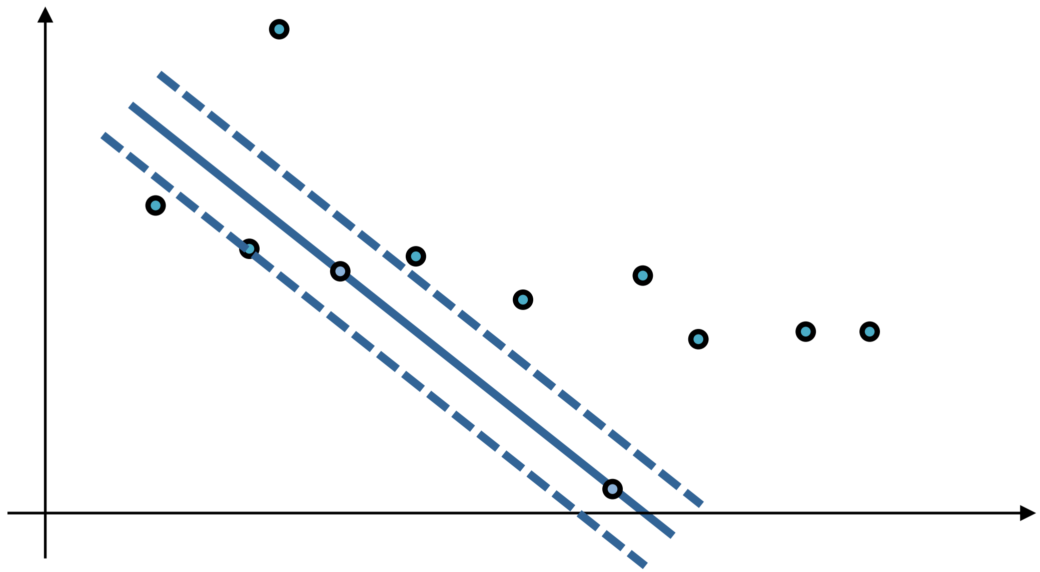


Quite ok

7 inliers
4 outliers

RANSAC

- Let’s try 2 other random points

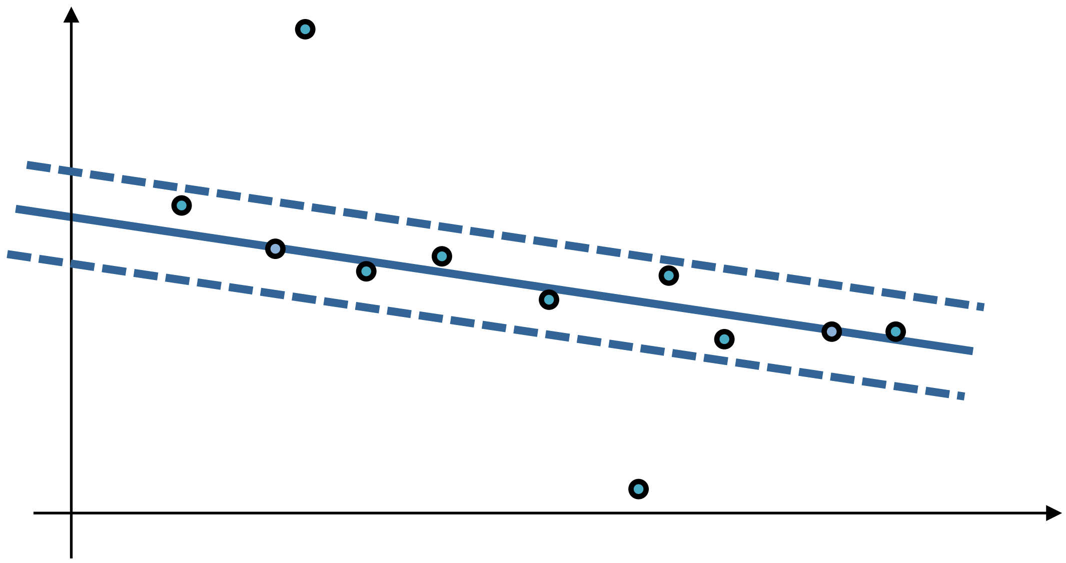


Quite bad

3 inliers
8 outliers

RANSAC

- Let's try yet another 2 random points

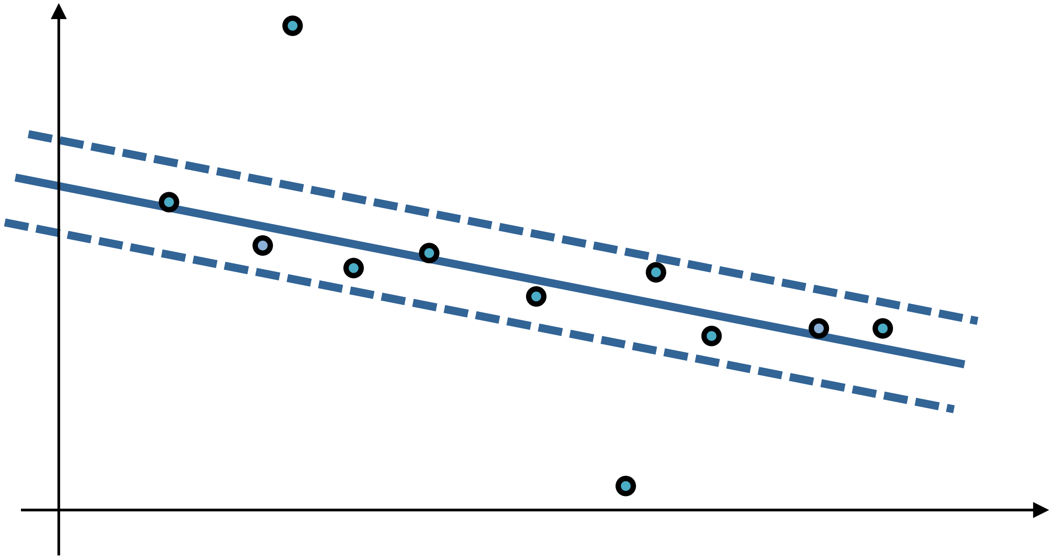


Quite good!

9 inliers
2 outliers

RANSAC

- Let's use the inliers of the best trial so far to perform least squares fitting



Even better!

RANSAC

- How many iteration do we need to find the optimal solution
  - p - probability of finding the correct solution
  - ϵ - outlier ration → $w = 1 - \epsilon$
  - s - number of data points required to calculate solution
  - N - number of iterations

Probability of picking at least one outlier

$$\underbrace{1 - p}_{\text{Probability of not a single correct solution}} \geq (\overbrace{1 - w^s}^{\text{Probability of picking at least one outlier}})^N = (1 - (1 - \epsilon)^s)^N$$

Probability of not a single correct solution

Probability of picking s good samples (→ w^s)

$$N \geq \frac{\log(1 - p)}{\log(1 - (1 - \epsilon)^s)}$$

RANSAC Algorithm

- RANdom SAmple Consensus algorithm formalizes this idea
- Algorithm:

Input: data D, s required #data points for fitting, success probability p, outlier ratio ϵ

Output: inlier set

1. Compute required number of iterations $N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)}$
2. For N iterations do:
   1. Randomly select a subset of s data points
   2. Fit model on the subset
   3. Count inliers and keep model/subset with largest number of inliers
3. Refit model using found inlier set

RANSAC

N for $p = 0.99$

	Required points s	Outlier ratio ϵ 10%	20%	30%	40%	50%	60%	70%
Line	2	3	5	7	11	17	27	49
Plane	3	4	7	11	19	35	70	169
Essential matrix	8	9	26	78	272	1177	7025	70188

Lessons Learned Today

- Keypoint detection, description and matching is a well researched topic
- Highly performant corner and blob detectors exist
- Corners are optimized for localization accuracy
- Blobs have a natural notion of scale through the scale-normalized LoG
- ORB is currently most popular detector/descriptor combination for visual motion estimation
- Keypoint matching by descriptor distance
- Robust matching based on model fitting using RANSAC

Thanks for your attention!

Slides Information

- These slides have been initially created by Jörg Stückler as part of the lecture “Robotic 3D Vision” in winter term 2017/18 at Technical University of Munich.
- The slides have been revised by myself (Niclas Zeller) for the same lecture held in winter term 2020/21
- Acknowledgement of all people that contributed images or video material has been tried (please kindly inform me if such an acknowledgement is missing so it can be added).