Local Features

What we will learn today?

- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector
- Local features
  - SIFT
- Feature Matching

Some background reading:
Rick Szeliski, Chapter 4.1.1; David Lowe, IJCV 2004

Image matching: a challenging problem

Image matching: a challenging problem

by Diva Sian

by swashford

Slide credit: Steve Seitz

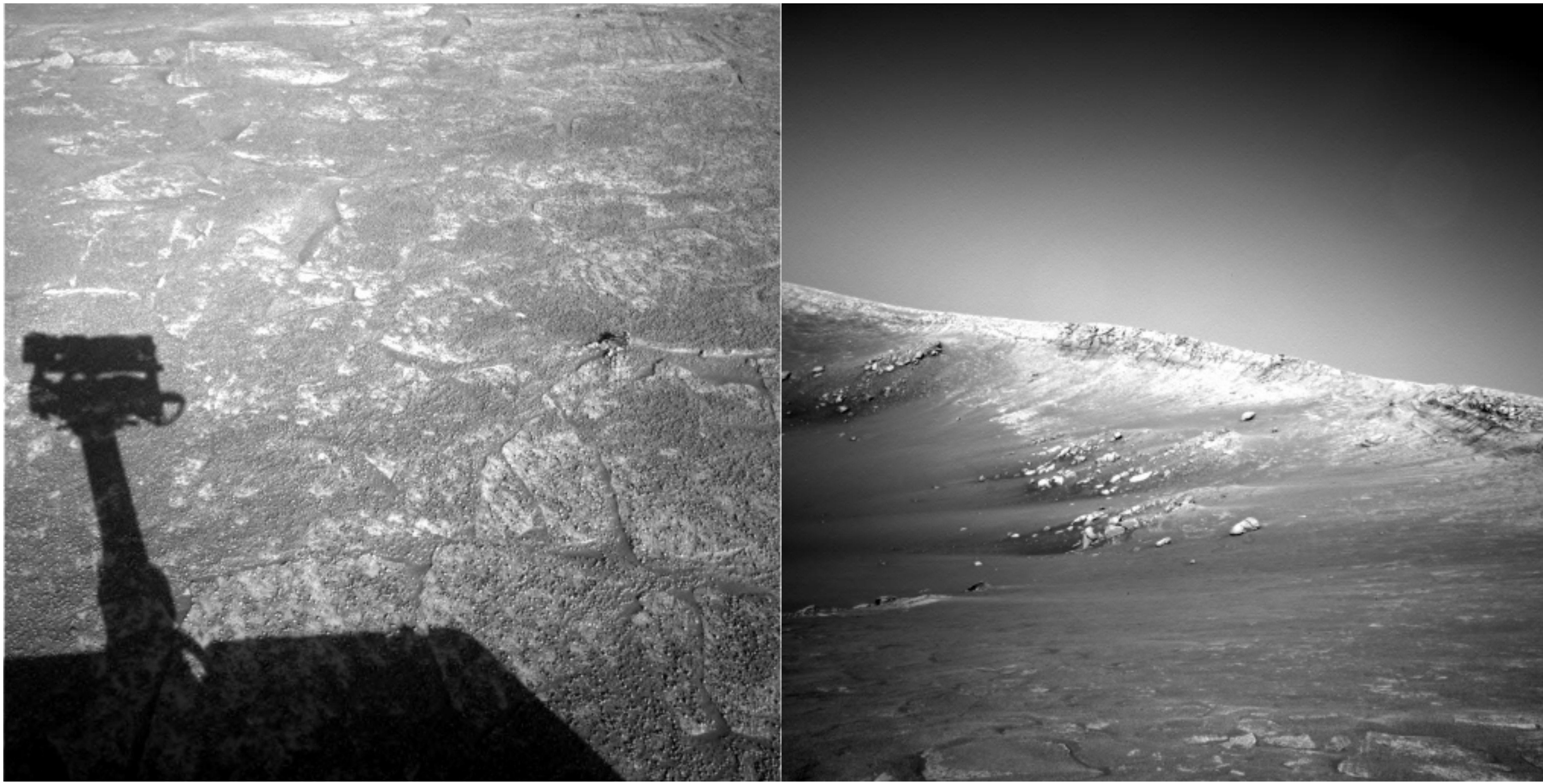
Harder Case

by Diva Sian

by scgbt

Slide credit: Steve Seitz

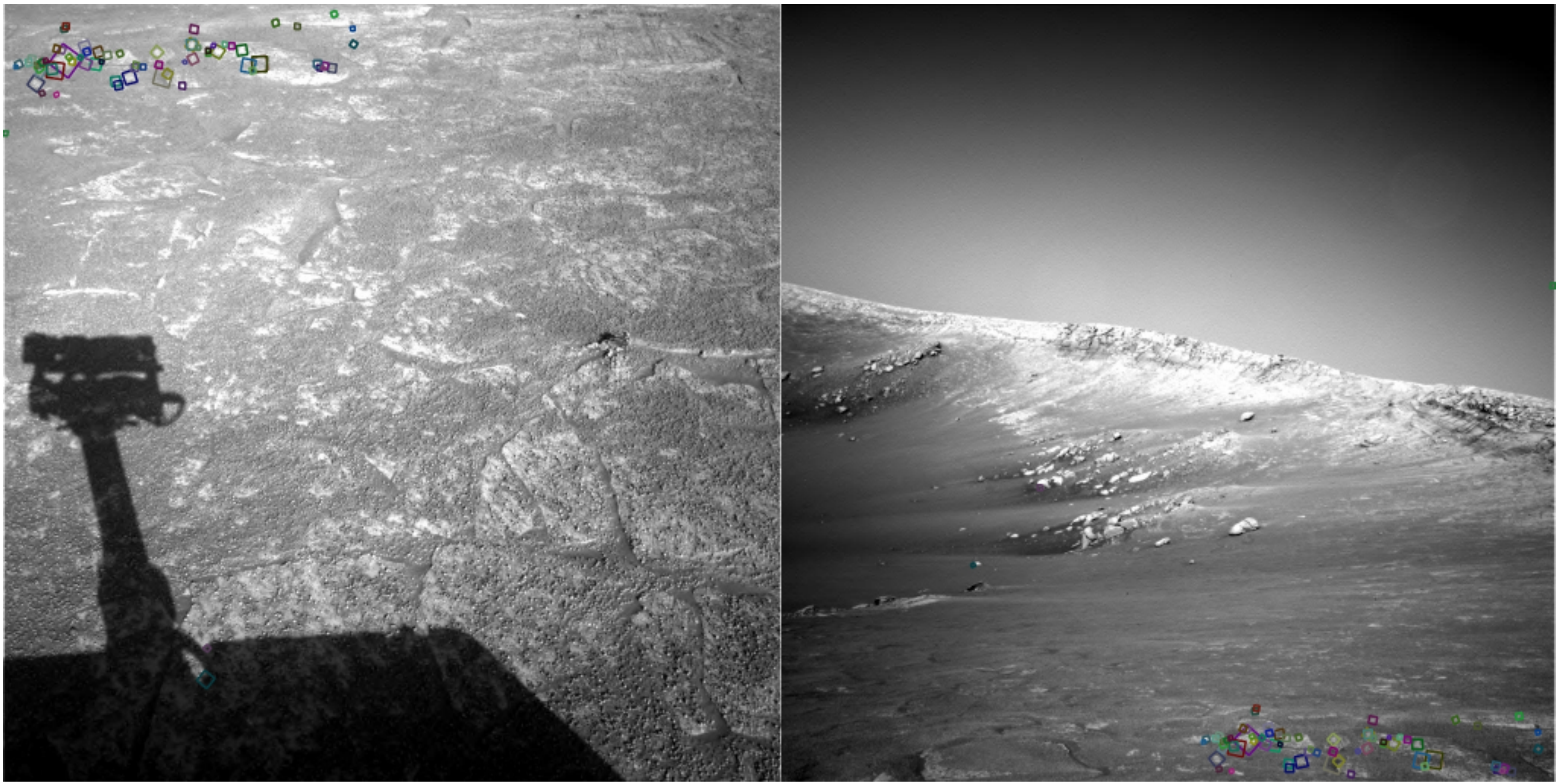
Harder Still?



NASA Mars Rover images

Slide credit: Steve Seitz

Answer Below (Look for tiny colored squares)



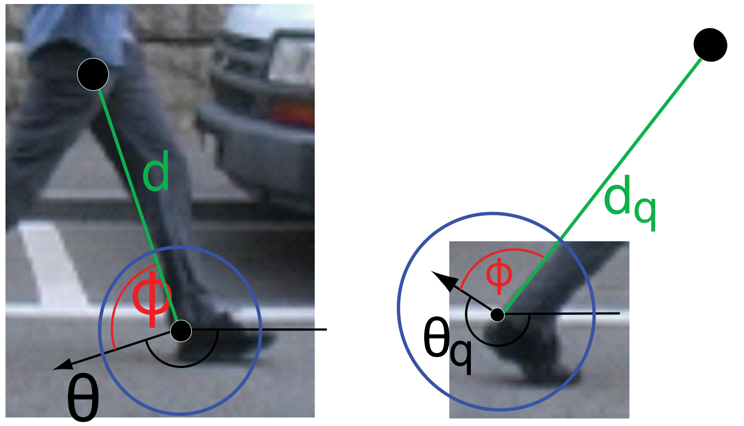
NASA Mars Rover images with SIFT feature matches
(Figure by Noah Snavely)

Slide credit: Steve Seitz

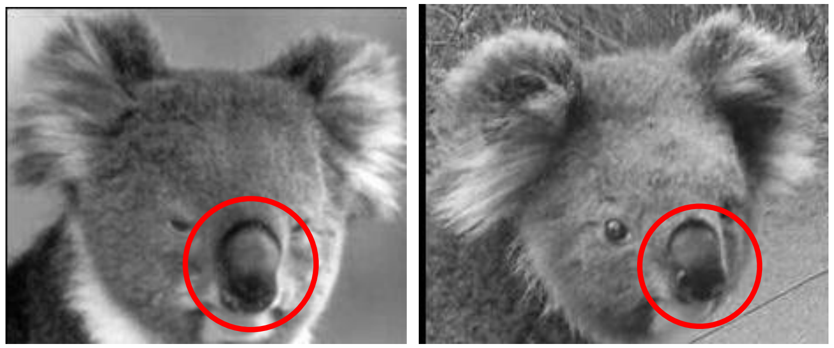
Motivation for using local features

- Global representations have major limitations
- Instead, describe and match only local regions
- Increased robustness to
  - Occlusions

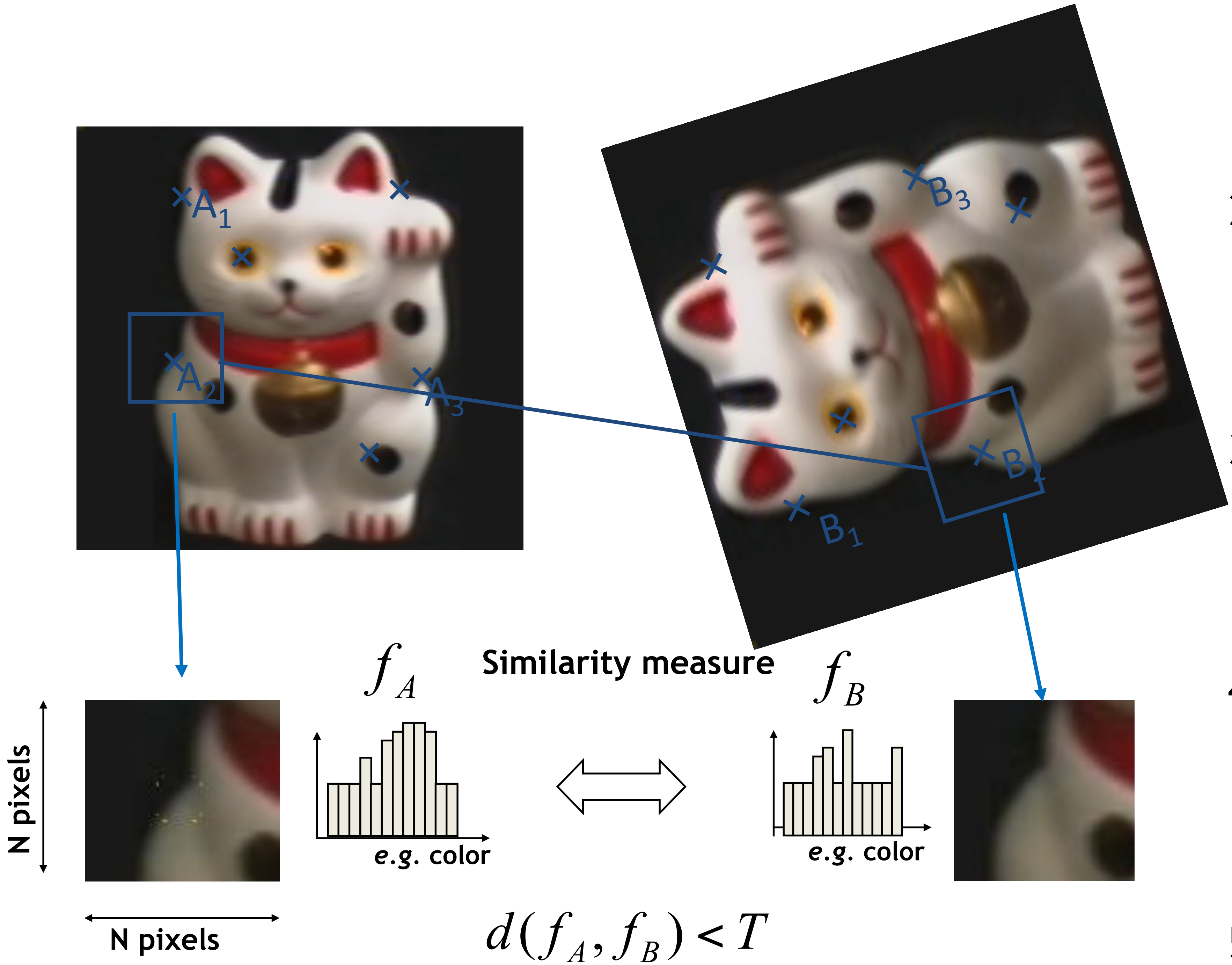
  - Articulation



  - Intra-category variations



General Approach



f_A f_B

$$d(f_A, f_B) < T$$

1. Find a set of distinctive key-points
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

Slide credit: Bastian Leibe

Common Requirements

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

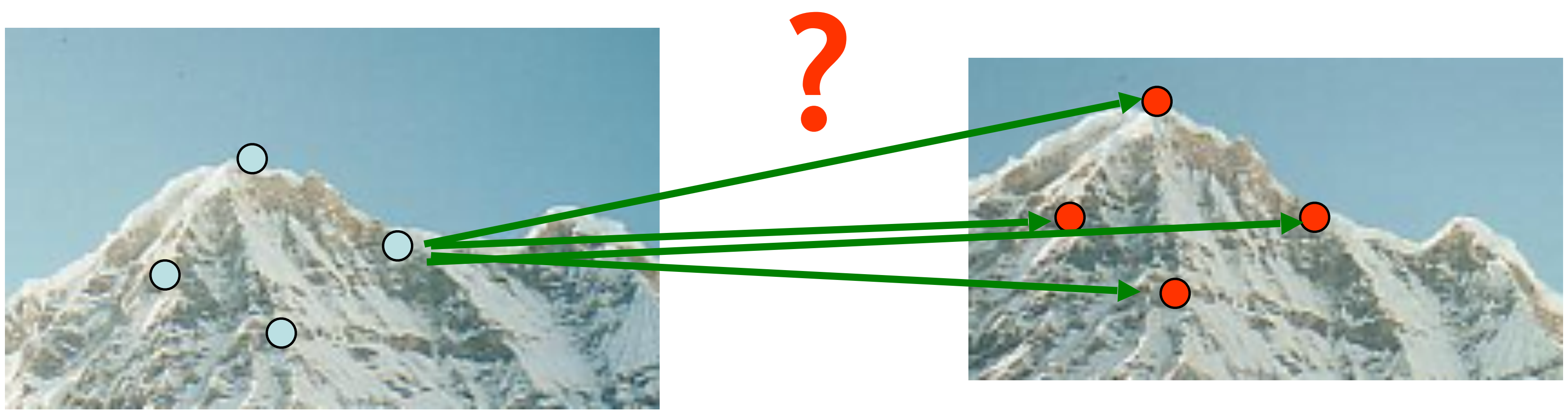
No chance to match!

We need a repeatable detector!

Slide credit: Darya Frolova, Denis Simakov

Common Requirements

- Problem 1:
  - Detect the same point independently in both images
- Problem 2:
  - For each point correctly recognize the corresponding one



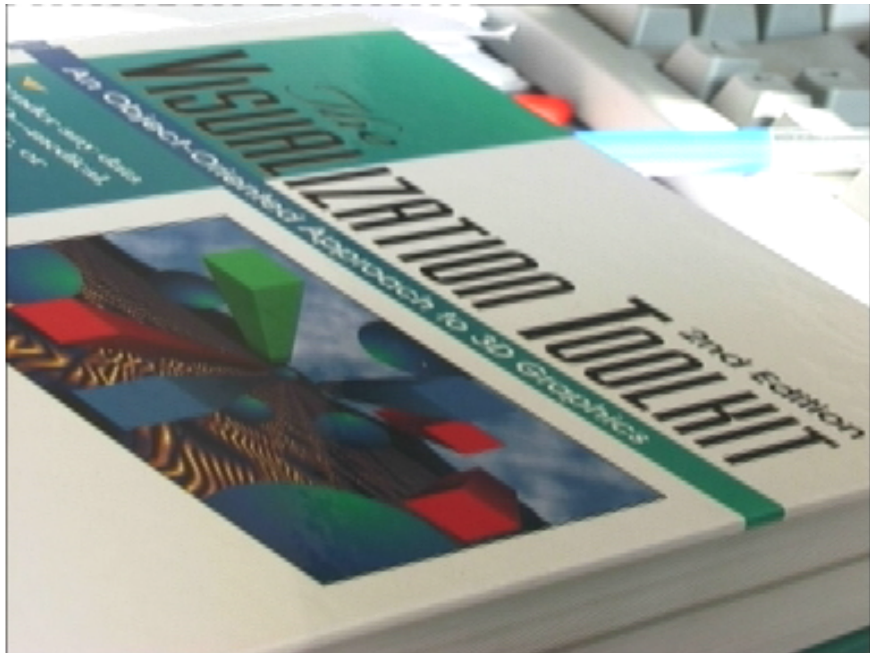
We need a reliable and distinctive descriptor!

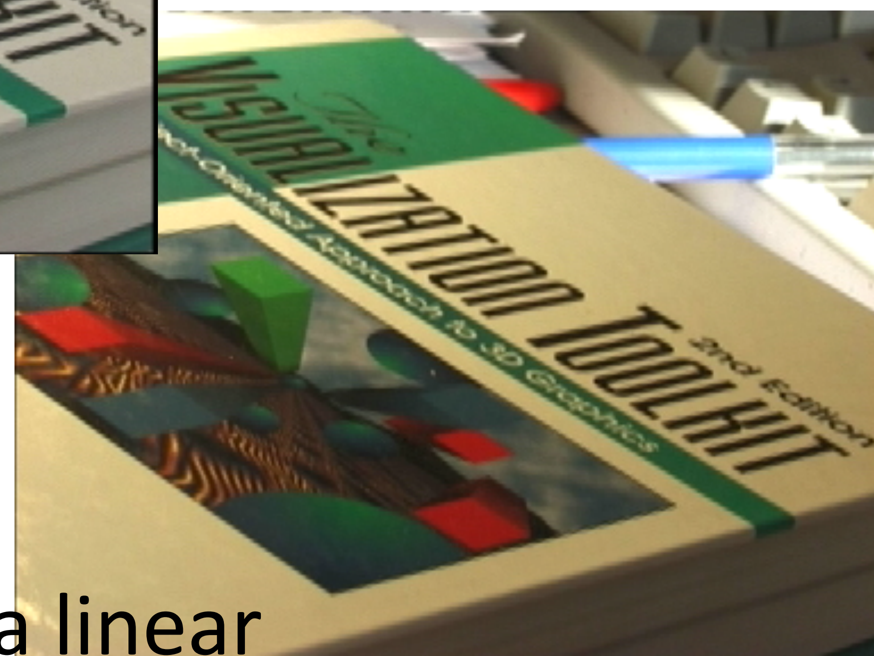
Slide credit: Darya Frolova, Denis Simakov

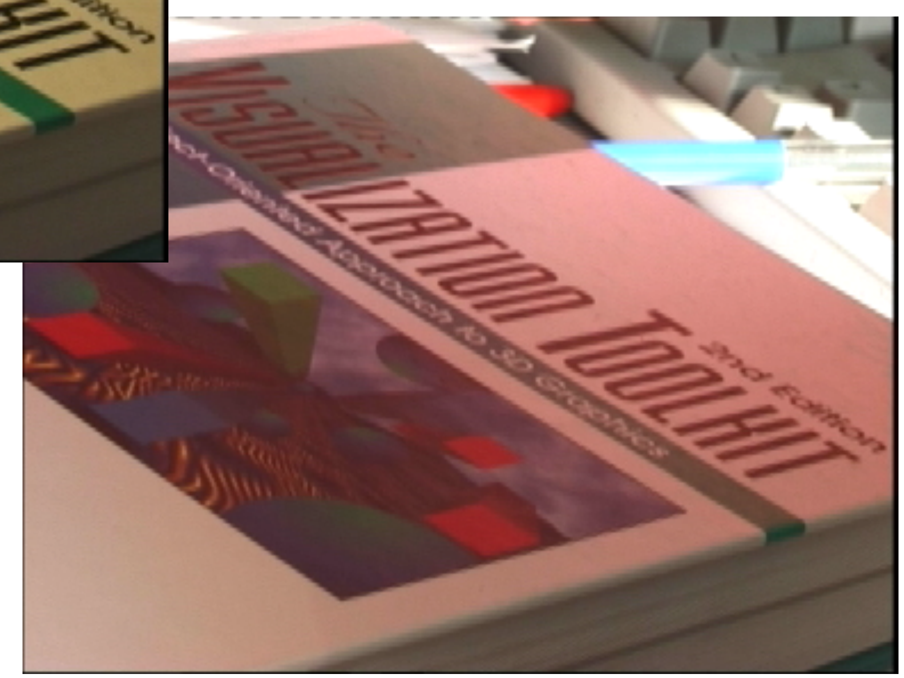
Invariance: Geometric Transformations

Slide credit: Steve Seitz

Invariance: Photometric Transformations







- Often modeled as a linear transformation:
  - Scaling + Offset

Slide credit: Tinne Tuytelaars

Requirements

- Region extraction needs to be repeatable and accurate
  - Invariant to translation, rotation, scale changes
  - Robust or covariant to out-of-plane (≈affine) transformations
  - Robust to lighting variations, noise, blur, quantization
- Locality: Features are local, therefore robust to occlusion and clutter.
- Quantity: We need a sufficient number of regions to cover the object.
- Distinctivenes: The regions should contain “interesting” structure.
- Efficiency: Close to real-time performance.

Slide credit: Bastian Leibe

Many Existing Detectors Available

- Hessian & Harris [Beaudet '78], [Harris '88]
- Laplacian, DoG [Lindeberg '98], [Lowe '99]
- Harris-/Hessian-Laplace [Mikolajczyk & Schmid '01]
- Harris-/Hessian-Affine [Mikolajczyk & Schmid '04]
- EBR and IBR [Tuytelaars & Van Gool '04]
- MSER [Matas '02]
- Salient Regions [Kadir & Brady '01]
- Others…

- Those detectors have become a basic building block for many recent applications in Computer Vision.

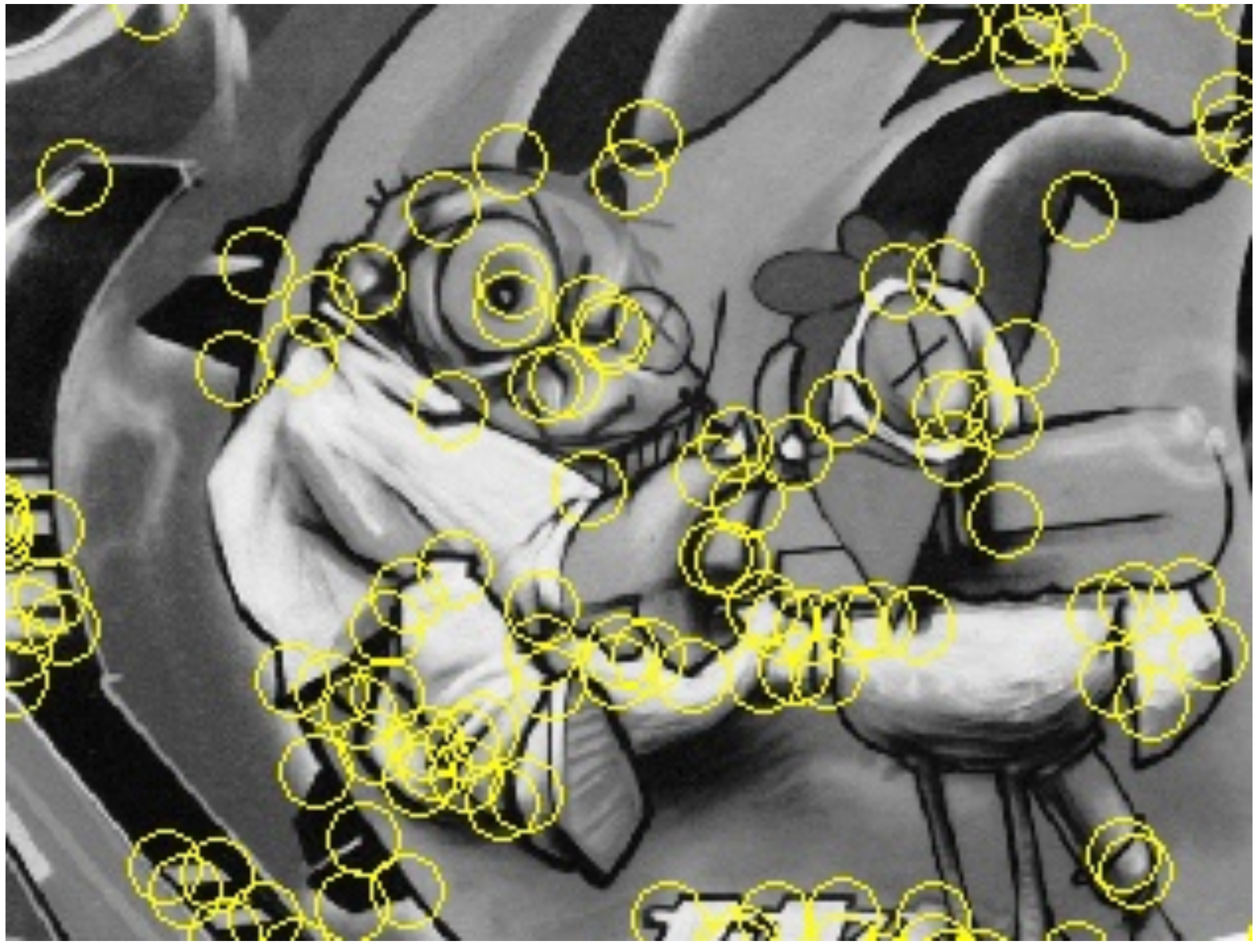
Slide credit: Bastian Leibe

What we will learn today?

- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector
- Local features
  - SIFT
- Feature Matching

Some background reading:
Rick Szeliski, Chapter 4.1.1; David Lowe, IJCV 2004

Keypoint Localization

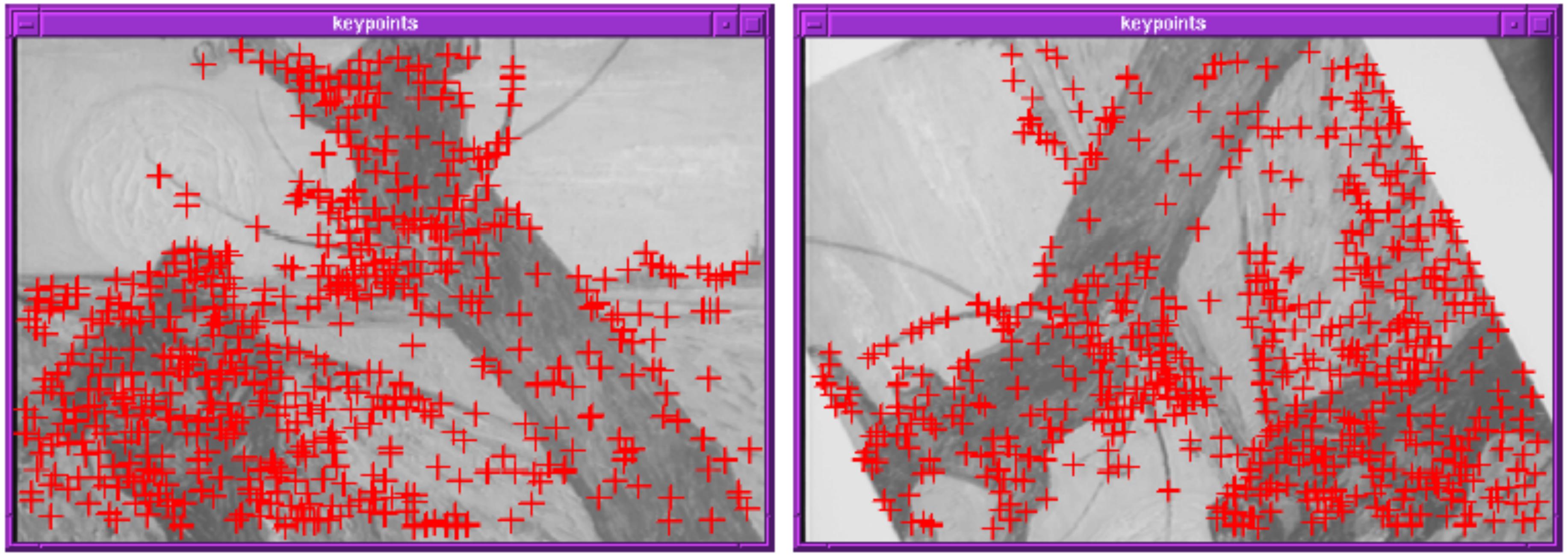


- Goals:
  - Repeatable detection
  - Precise localization
  - Interesting content

  $\Rightarrow$ Look for two-dimensional signal changes

Slide credit: Bastian Leibe

Finding Corners



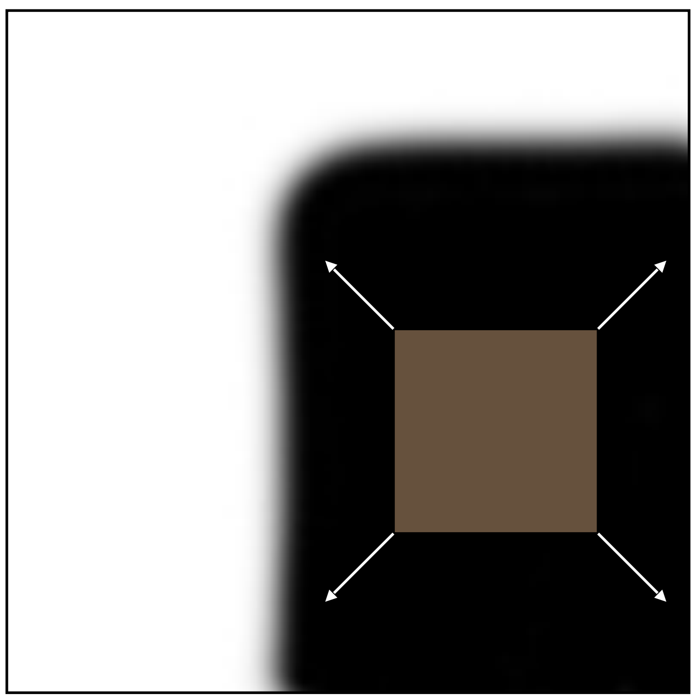
- Key property:
  - In the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

C.Harris and M.Stephens. ["A Combined Corner and Edge Detector."](#) *Proceedings of the 4th Alvey Vision Conference*, 1988.

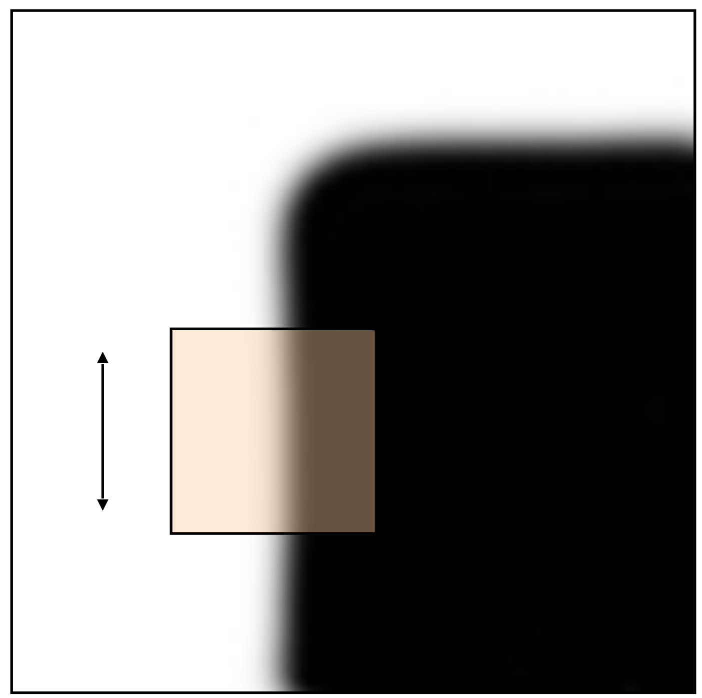
Slide credit: Svetlana Lazebnik

Corners as Distinctive Interest Points

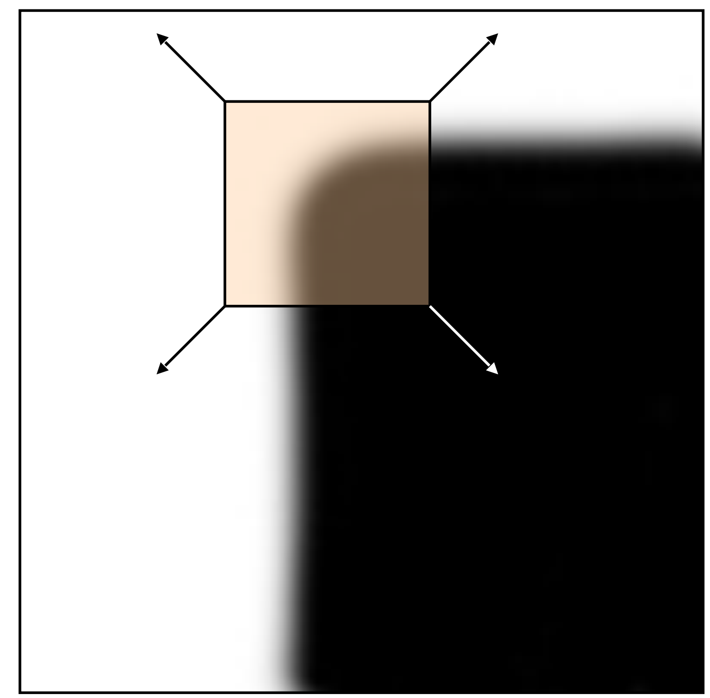
- Design criteria
  - We should easily recognize the point by looking through a small window (locality)
  - Shifting the window in any direction should give a large change in intensity (good localization)



“flat” region:
no change in all directions



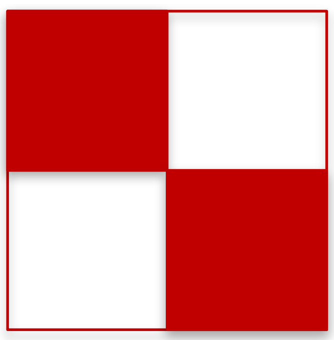
“edge”:
no change along the edge direction



“corner”:
significant change in all directions

Slide credit: Alyosha Efros

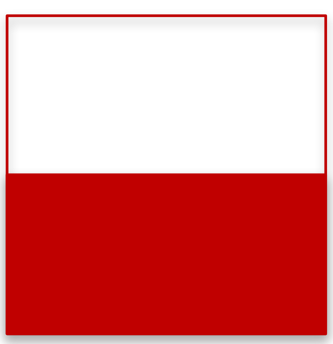
Corners versus edges



$\sum I_x^2$ → Large

$\sum I_y^2$ → Large

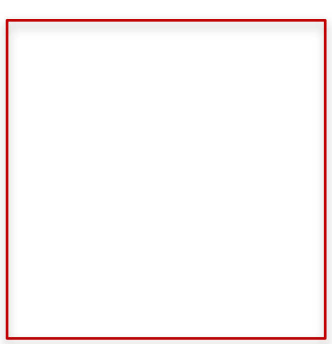
Corner



$\sum I_x^2$ → Small

$\sum I_y^2$ → Large

Edge

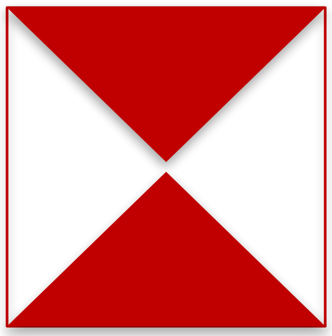


$\sum I_x^2$ → Small

$\sum I_y^2$ → Small

Nothing

Corners versus edges



$$\sum I_x^2 \longrightarrow ??$$

$$\sum I_y^2 \longrightarrow ??$$

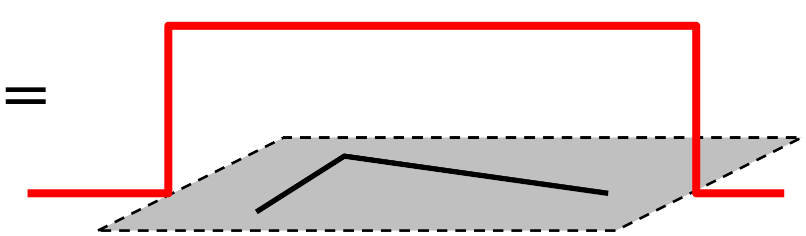
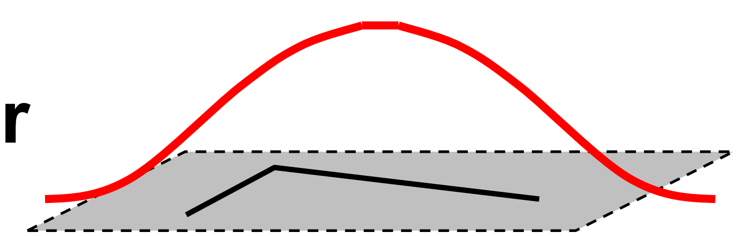
Corner

Harris Detector Formulation

- Change of intensity for the shift [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 $w(x,y) =$  **or** 

1 in window, 0 outside **Gaussian**

Slide credit: Rick Szeliski

Harris Detector Formulation

- This measure of change can be approximated by:

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

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

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

Gradient with respect to x, times gradient with respect to y

Sum over image region – the area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y]$$

Slide credit: Rick Szeliski

Harris Detector Formulation

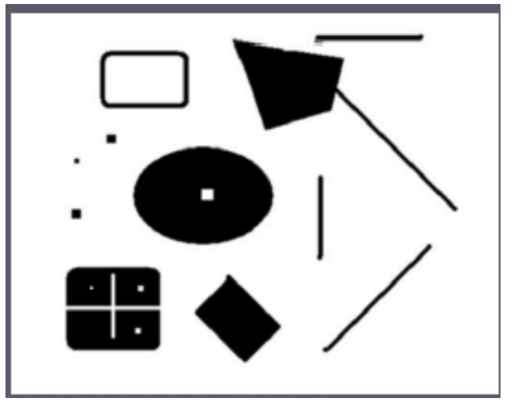
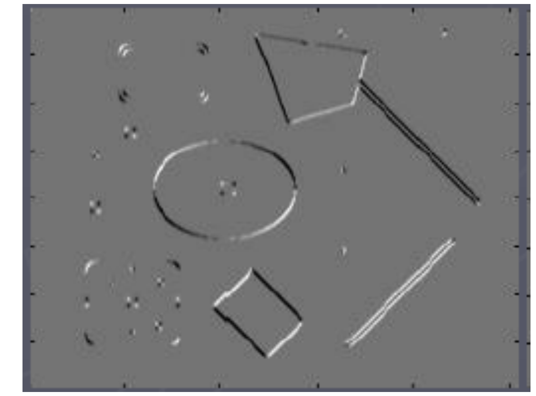


Image I

I_x

I_y



$I_x I_y$

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

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

Gradient with respect to x, times gradient with respect to y

Sum over image region – the area we are checking for corner

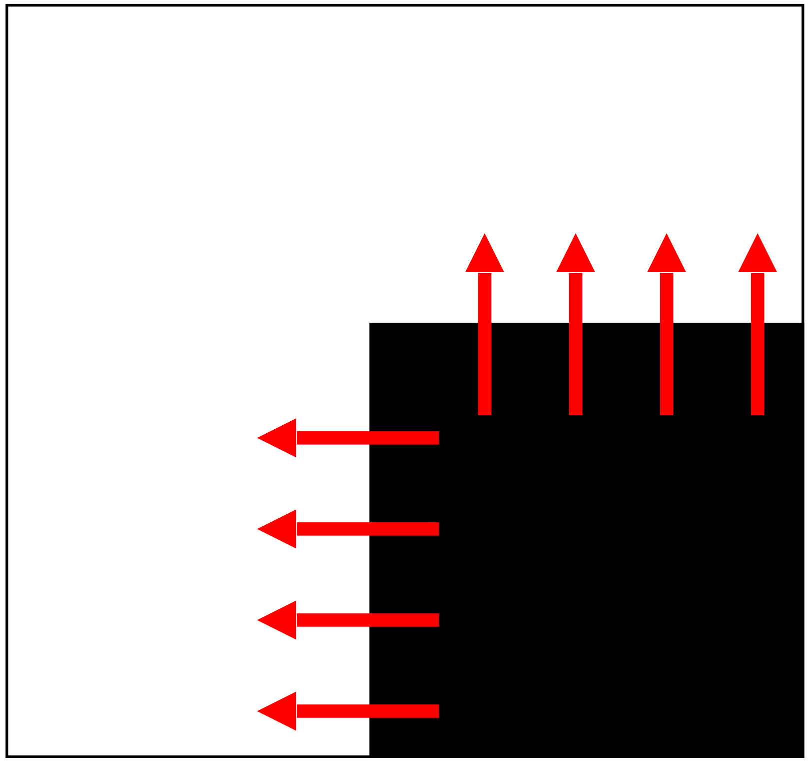
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y]$$

Slide credit: Rick Szeliski

What Does This Matrix Reveal?

- First, let's consider an axis-aligned corner:

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

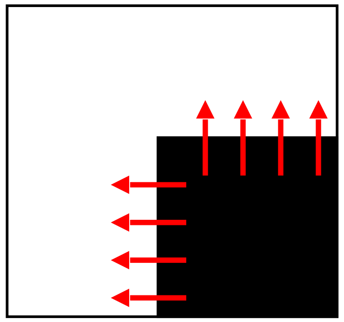


Slide credit: David Jacobs

What Does This Matrix Reveal?

- First, let's consider an axis-aligned corner:

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



- This means:
  - Dominant gradient directions align with x or y axis
  - If either λ is close to 0, then this is not a corner, so look for locations where both are large.

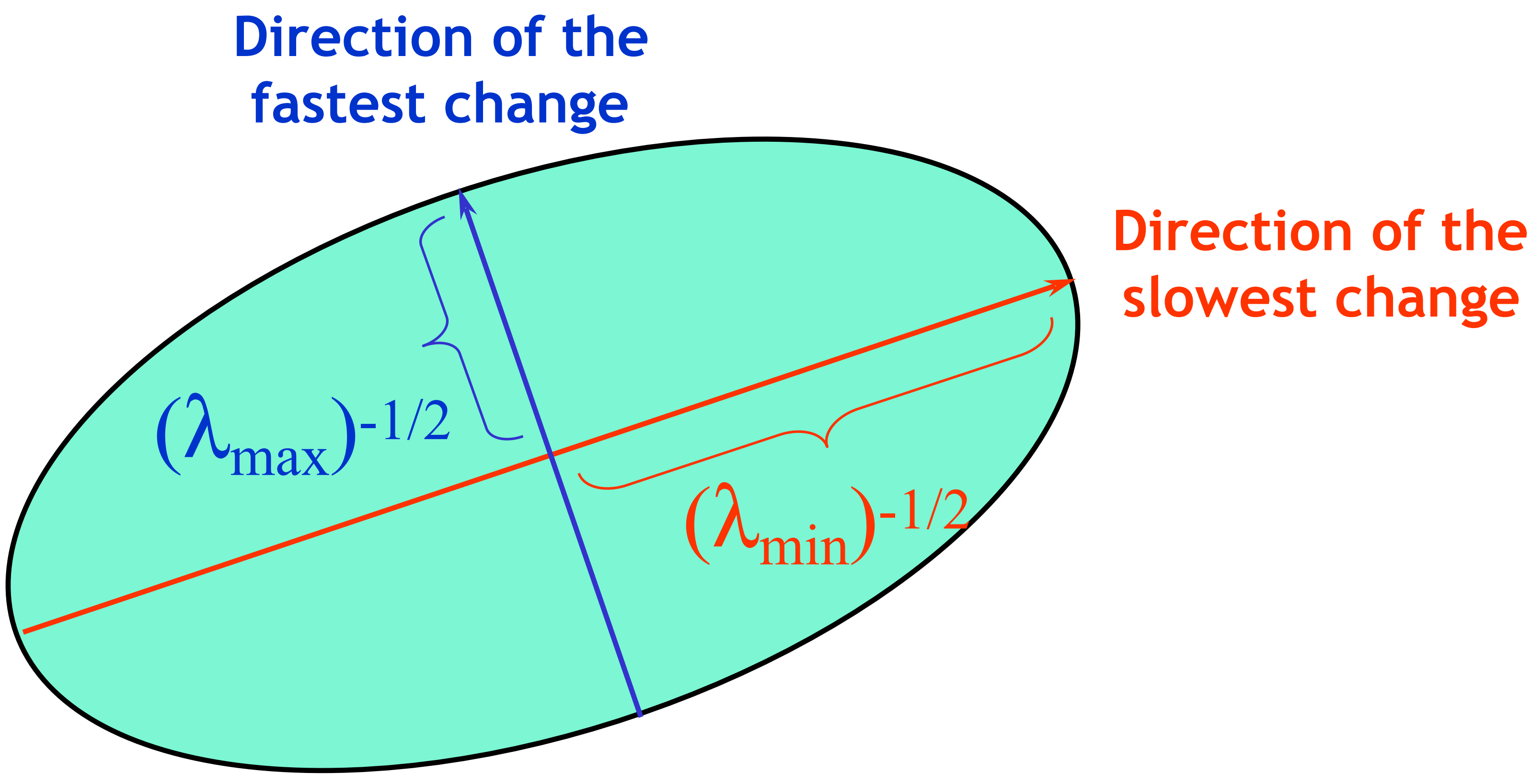
- What if we have a corner that is not aligned with the image axes?

General Case

- Since M is symmetric, we have $$M = R^{-1}\begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix}R$$

  (Eigenvalue decomposition)

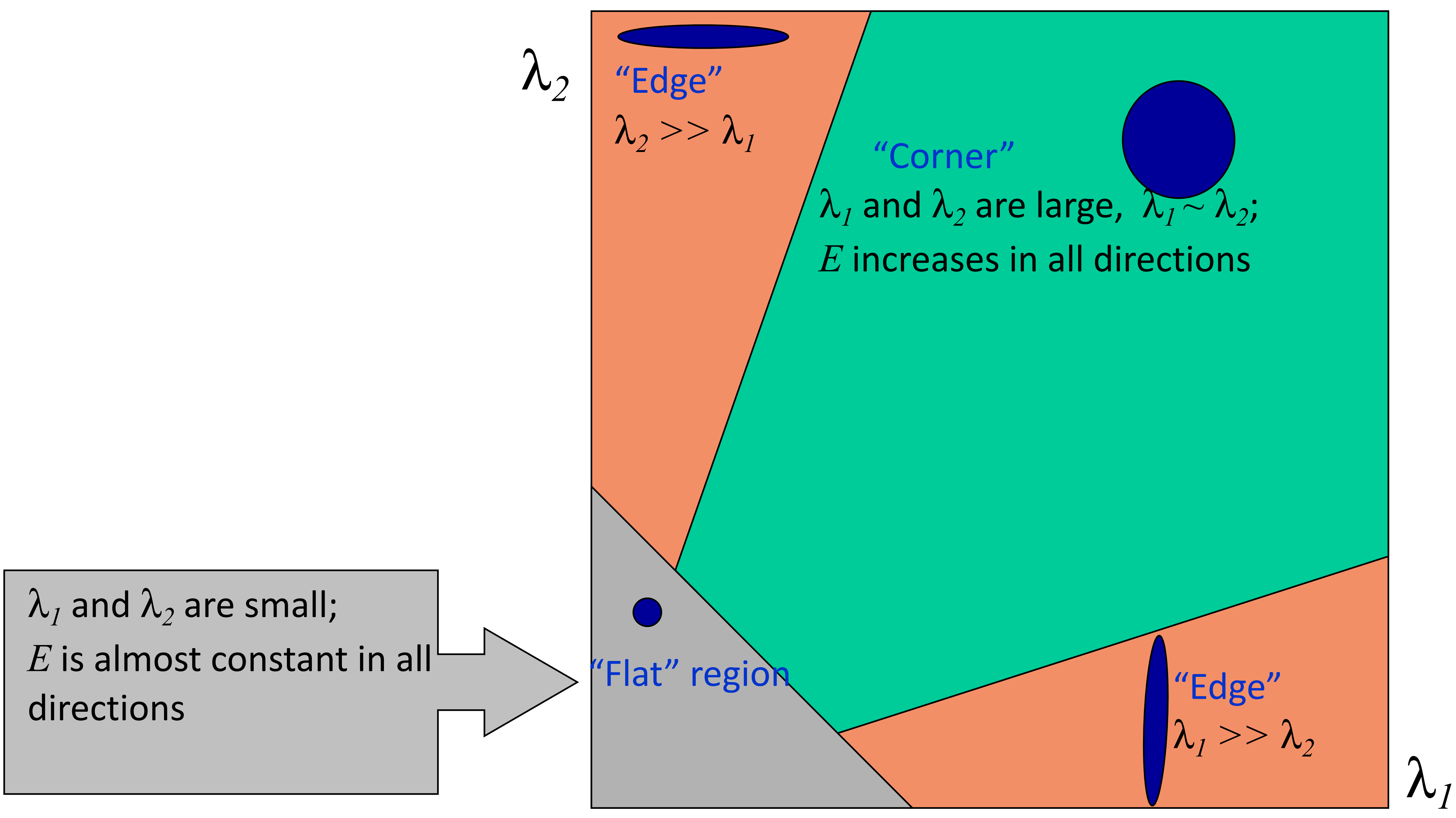
- We can visualize M as an ellipse with axis lengths determined by the eigenvalues and orientation determined by R



adapted from Darya Frolova, Denis Simakov

Interpreting the Eigenvalues

- Classification of image points using eigenvalues of M:

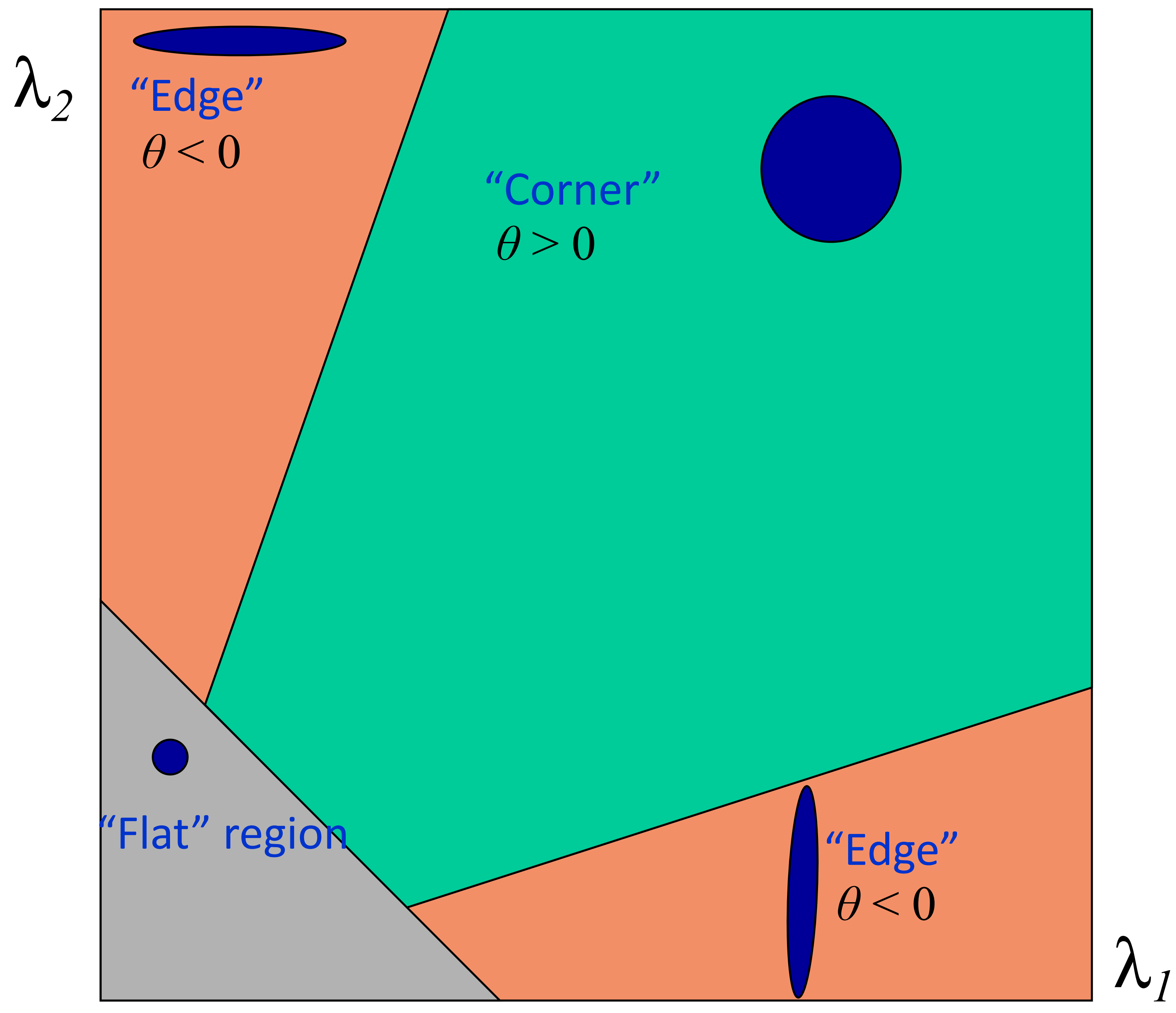


Slide credit: Kristen Grauman

Corner Response Function

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

- Fast approximation
  - Avoid computing the eigenvalues
  - α: constant (0.04 to 0.06)



Slide credit: Kristen Grauman

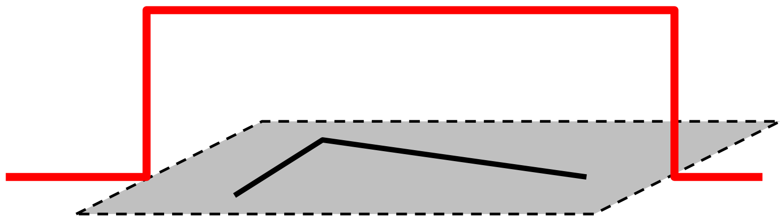
Window Function $w(x,y)$

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

- Option 1: uniform window
  - Sum over square window

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

  - Problem: not rotation invariant

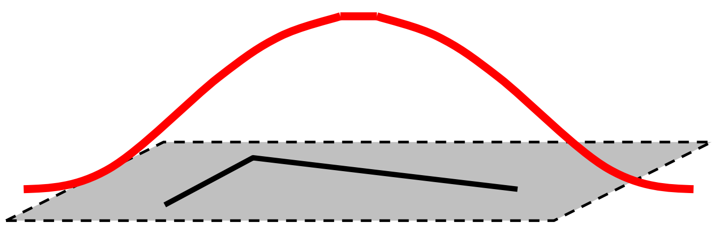


1 in window, 0 outside

- Option 2: Smooth with Gaussian
  - Gaussian already performs weighted sum

$$M = g(\sigma) * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

  - Result is rotation invariant



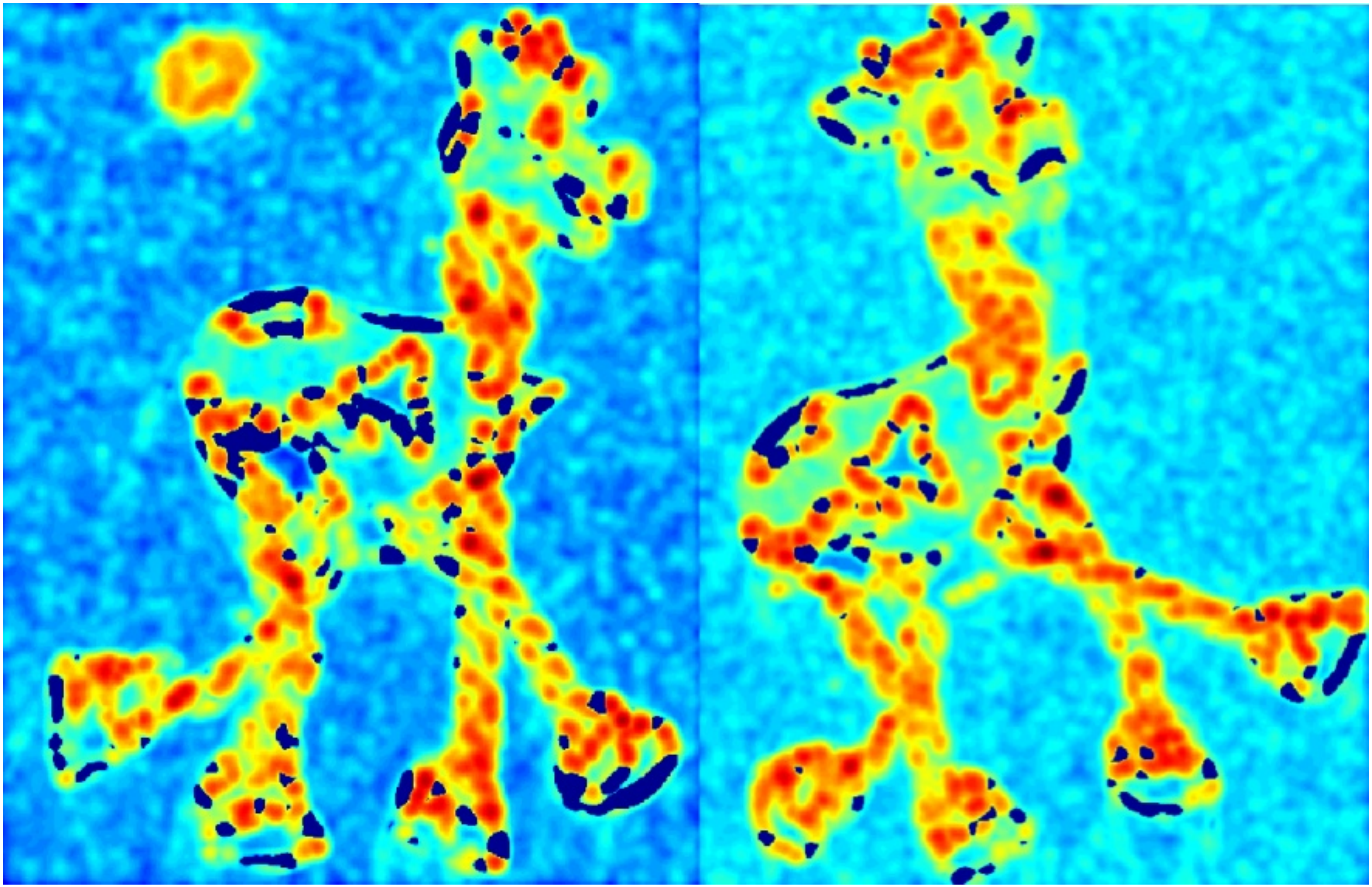
Gaussian

Slide credit: Bastian Leibe

Harris Detector: Workflow

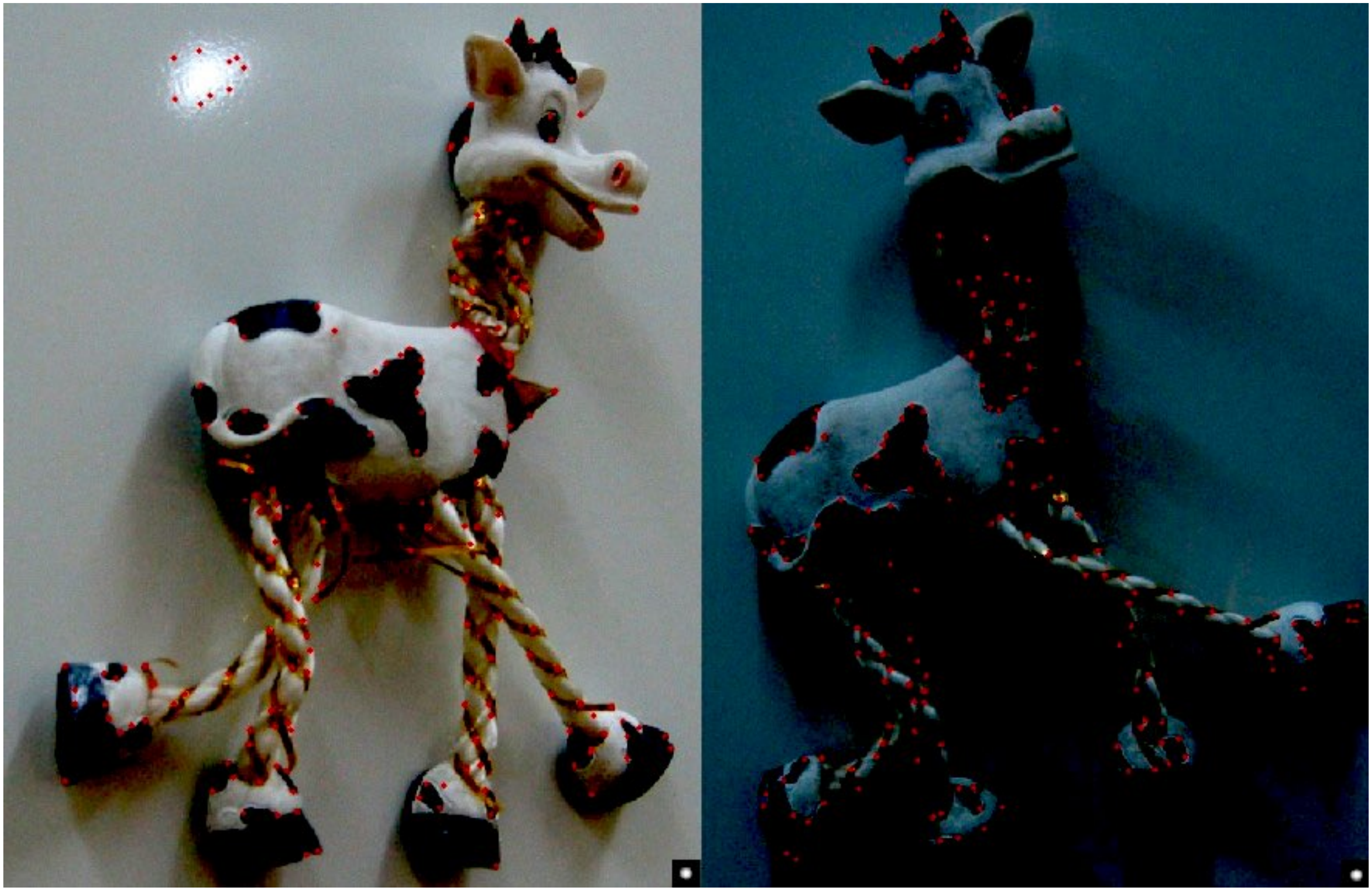
Slide adapted from Darya Frolova, Denis Simakov

Harris Detector: Workflow
- computer corner responses θ



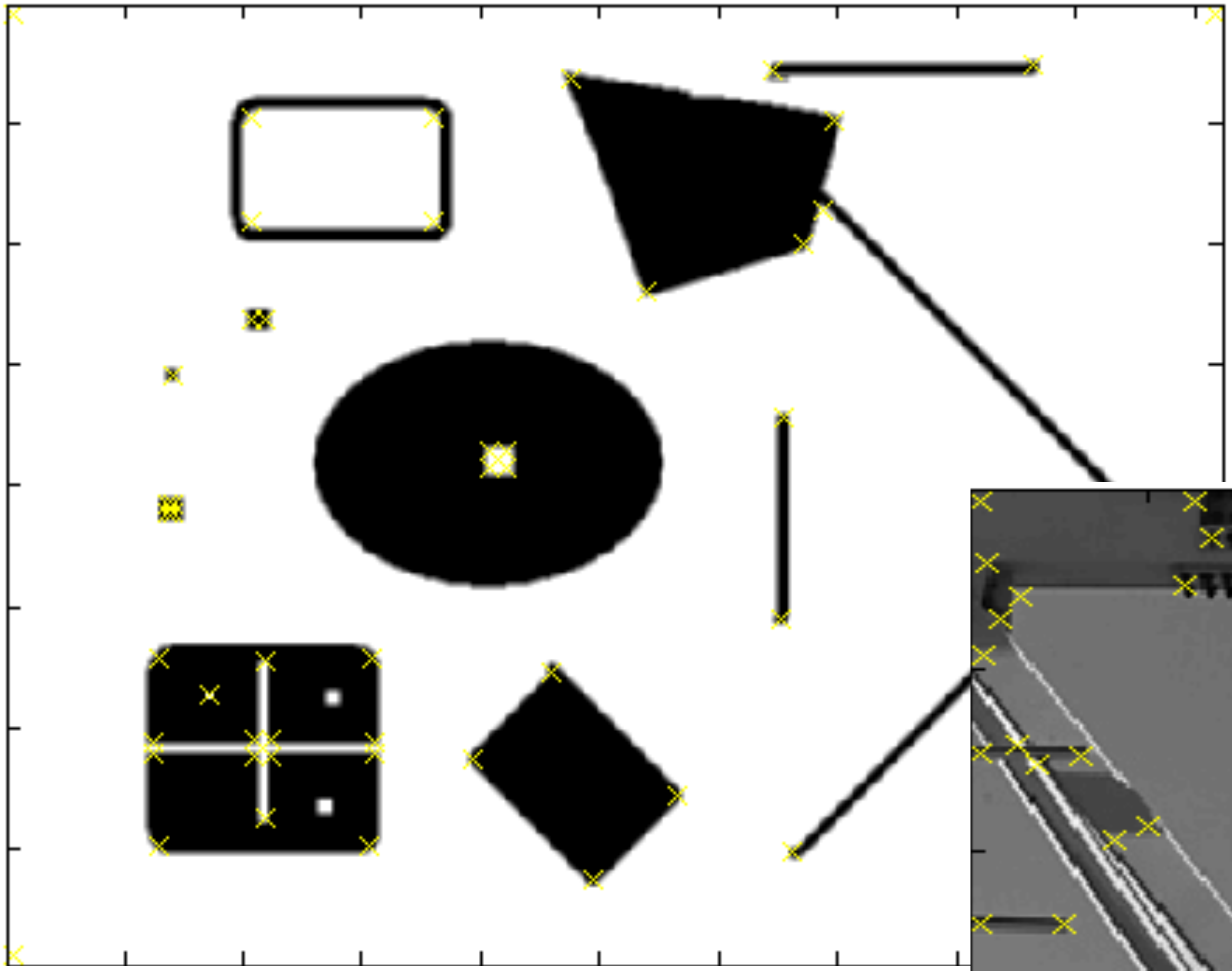
Slide adapted from Darya Frolova, Denis Simakov

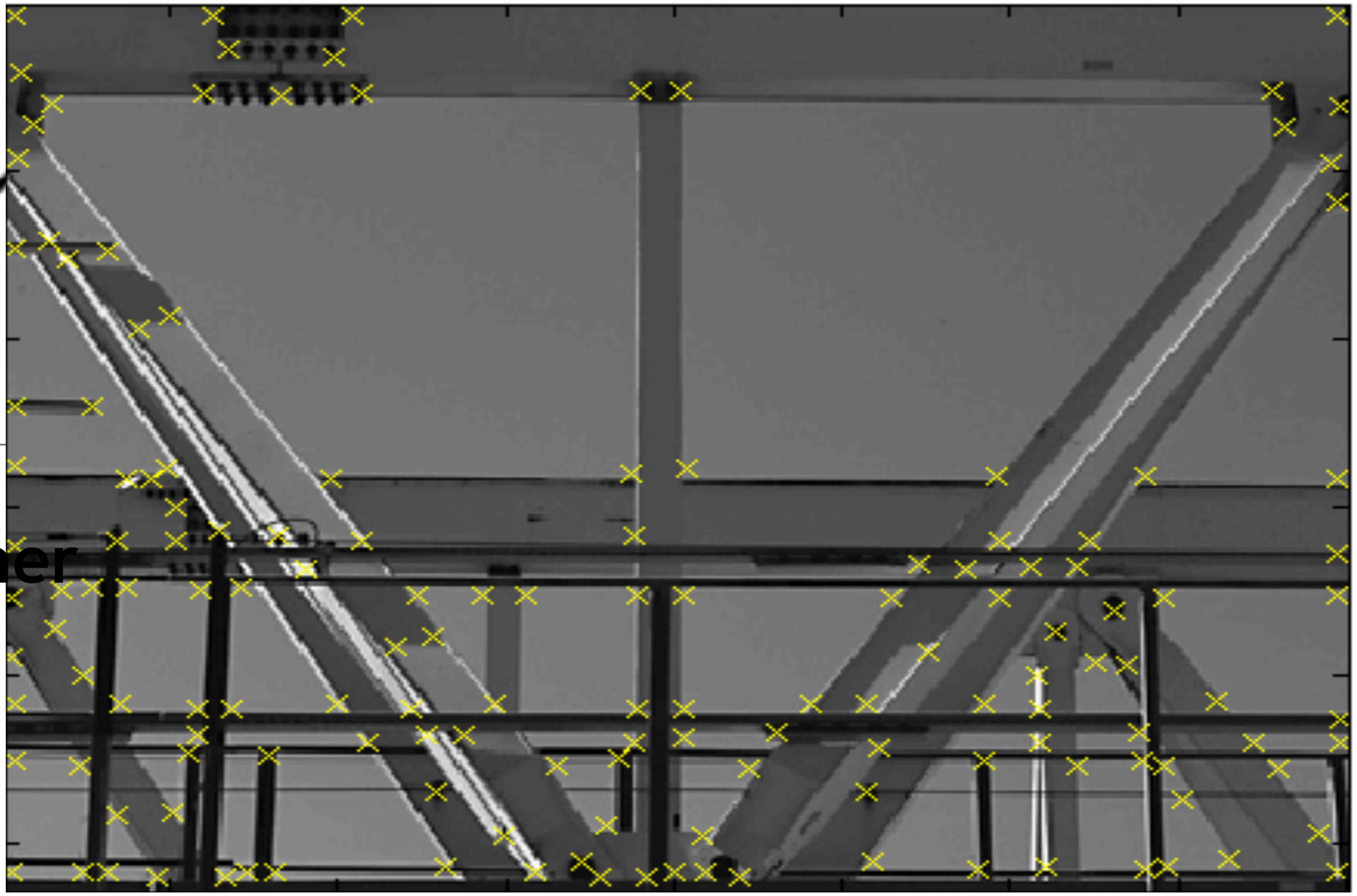
Harris Detector: Workflow
- Resulting Harris points



Slide adapted from Darya Frolova, Denis Simakov

Harris Detector – Responses [Harris88]





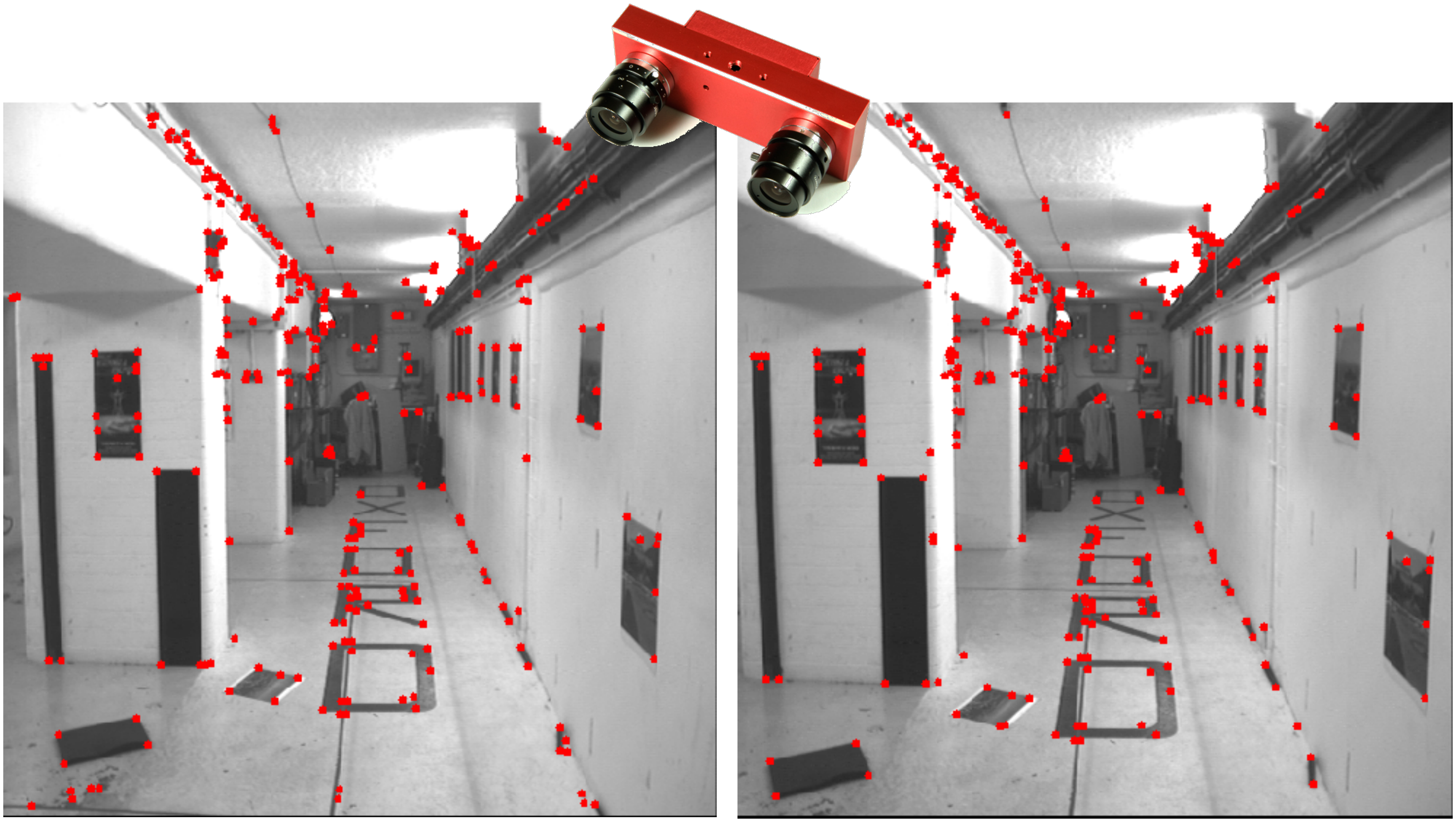
Effect: **A very precise corner detector.**

Slide credit: Krystian Mikolajczyk

Harris Detector – Responses [Harris88]

Slide credit: Krystian Mikolajczyk

Harris Detector – Responses [Harris88]



- Results are well suited for finding stereo correspondences

Slide credit: Kristen Grauman

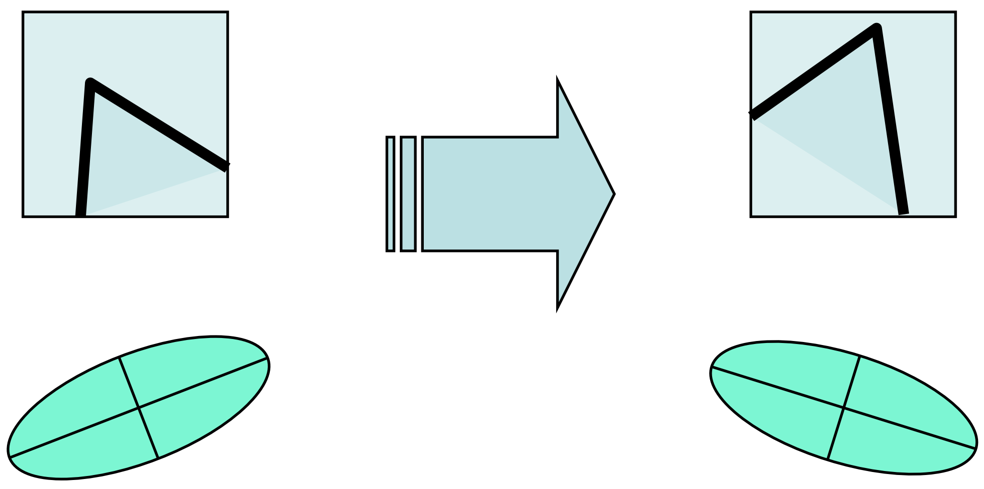
Harris Detector: Properties

- Translation invariance?

Slide credit: Kristen Grauman

Harris Detector: Properties

- Translation invariance
- Rotation invariance?



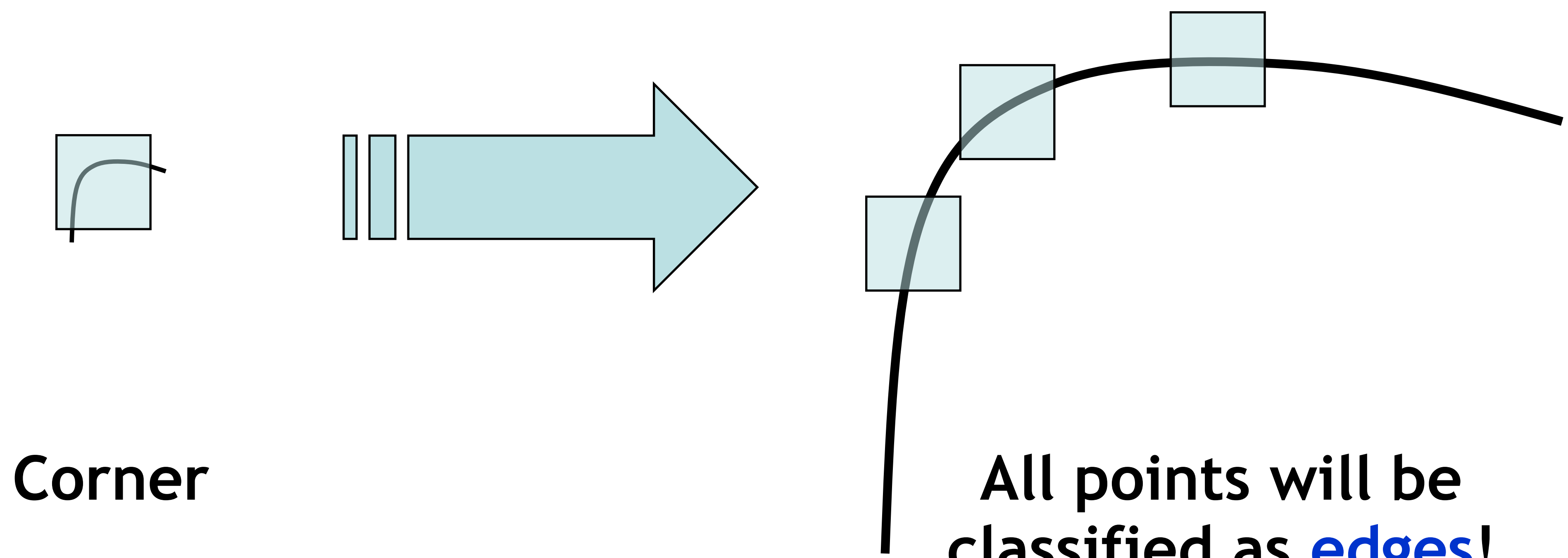
Ellipse rotates but its shape (i.e. eigenvalues) remains the same

***Corner response θ* is invariant to image rotation**

Slide credit: Kristen Grauman

Harris Detector: Properties

- Translation invariance
- Rotation invariance
- Scale invariance?



Not **invariant to image scale!**

Slide credit: Kristen Grauman

So far: can localize in x-y, but not scale

Automatic Scale Selection

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

How to find patch sizes at which *f* response is equal?

What is a good *f* ?

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

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

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

Response of some function *f*

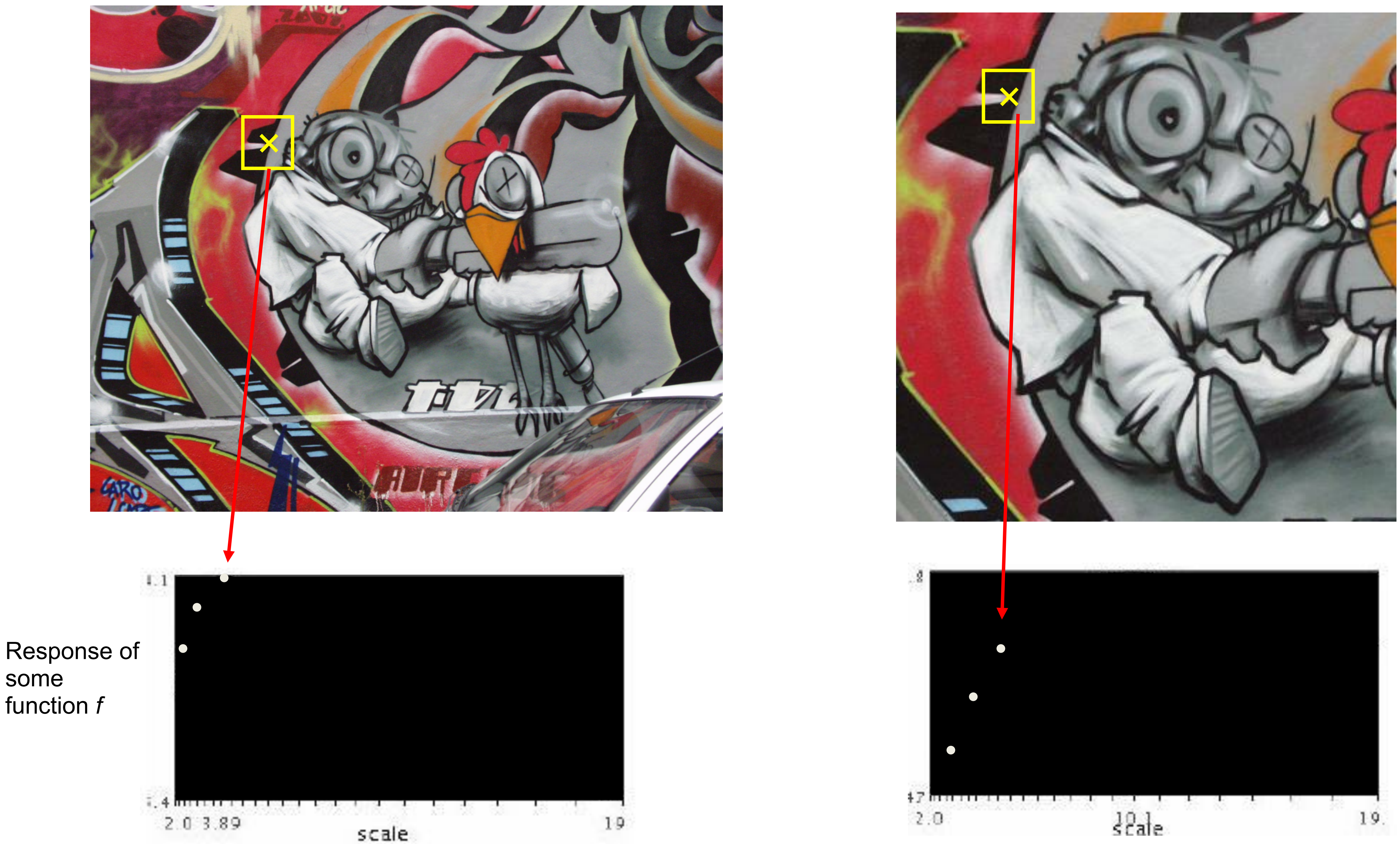
$$f(I_{i_1 \ldots i_m}(x,\sigma))$$

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

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



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

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

Response of some function *f*

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

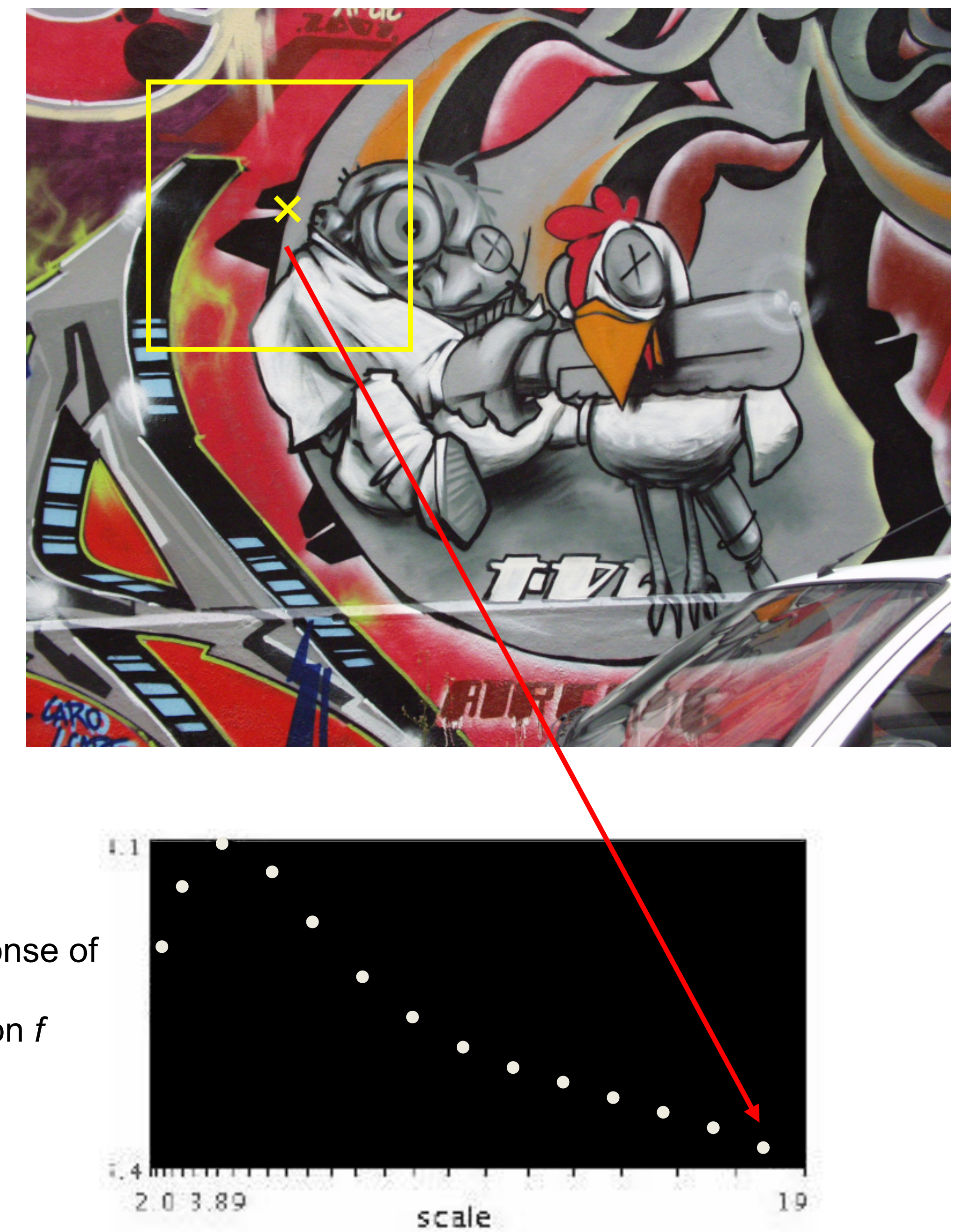
$$f(I_{i_1 \dots i_m}(x',\sigma'))$$

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

Response of some function *f*



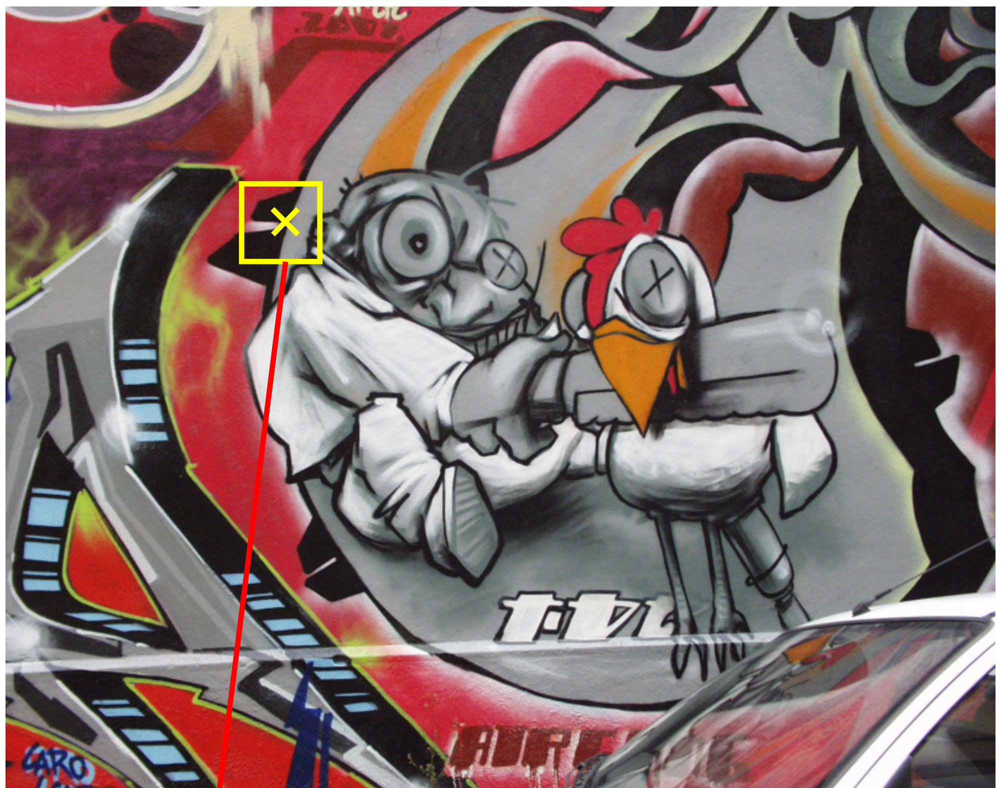
$$f(I_{i_1\ldots i_m}(x,\sigma))$$

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

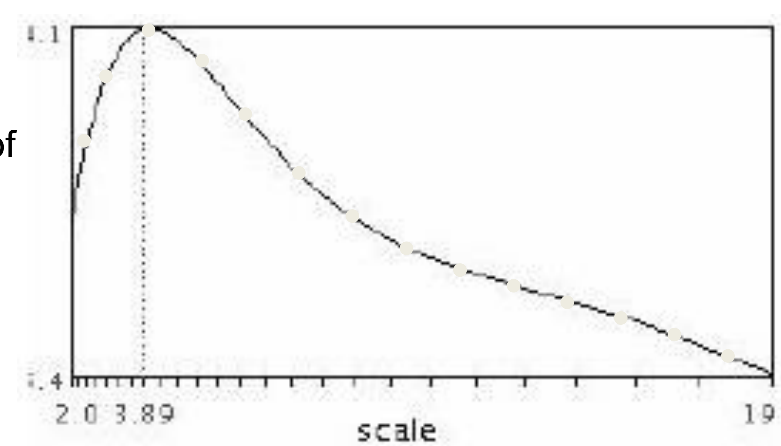
K. Grauman, B. Leibe

Automatic Scale Selection

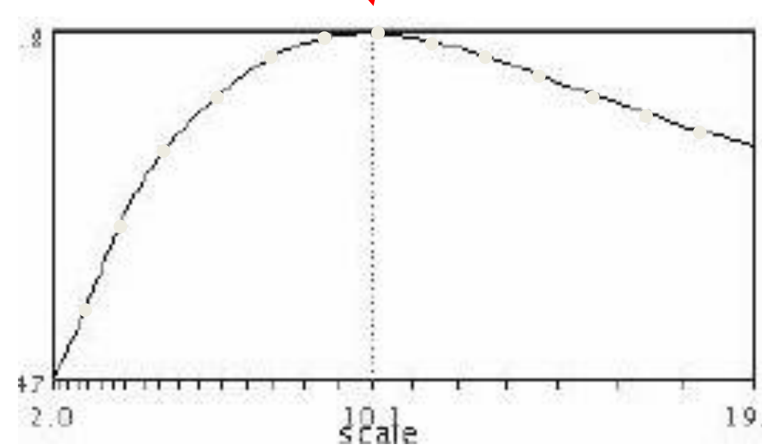
- Function responses for increasing scale (scale signature)



Response of some function *f*



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



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

Comparison of Keypoint Detectors

Table 7.1 Overview of feature detectors.

Feature Detector	Corner	Blob	Region	Rotation invariant	Scale invariant	Affine invariant	Repeatability	Localization accuracy	Robustness	Efficiency
Harris	√			√			+++	+++	+++	++
Hessian		√		√			++	++	++	+
SUSAN	√			√			++	++	++	+++
Harris-Laplace	√	(√)		√	√		+++	+++	++	+
Hessian-Laplace	(√)	√		√	√		+++	+++	+++	+
DoG	(√)	√		√	√		++	++	++	++
SURF	(√)	√		√	√		++	++	++	+++
Harris-Affine	√	(√)		√	√	√	+++	+++	++	++
Hessian-Affine	(√)	√		√	√	√	+++	+++	+++	++
Salient Regions	(√)	√		√	√	(√)	+	+	++	+
Edge-based	√			√	√	√	+++	+++	+	+
MSER			√	√	√	√	+++	+++	++	+++
Intensity-based			√	√	√	√	++	++	++	++
Superpixels			√	√	(√)	(√)	+	+	+	+

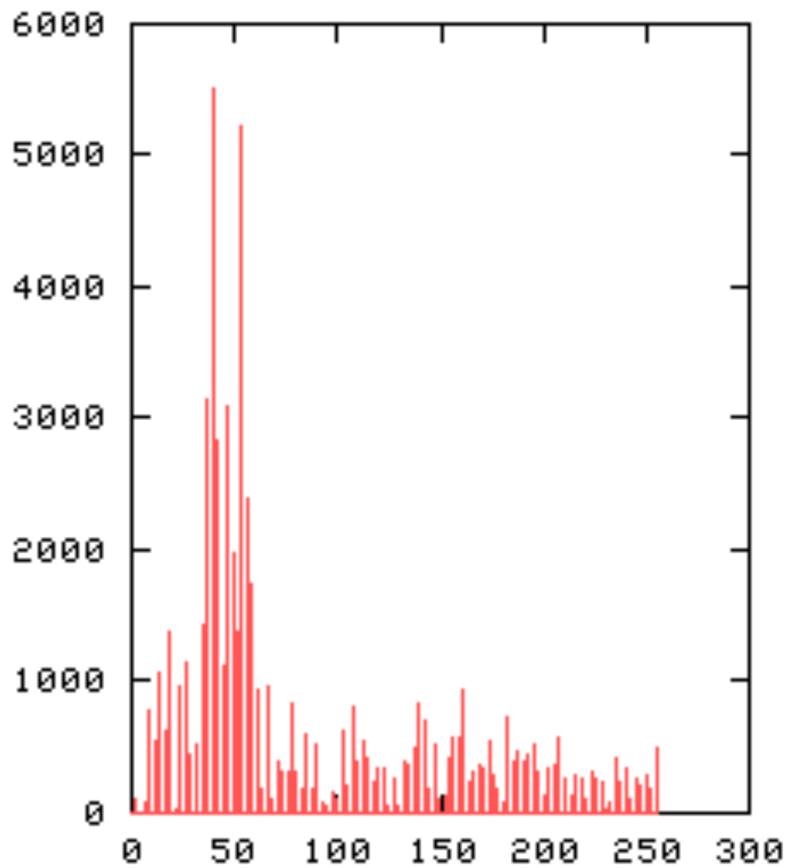
Tuytelaars Mikolajczyk 2008

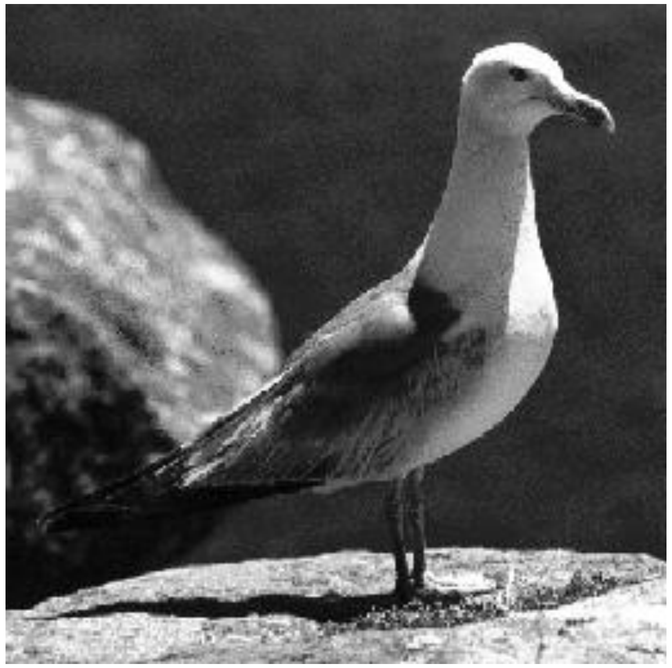
What we will learn today?

- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector
- Local features
  - SIFT
- Feature Matching

Some background reading:
Rick Szeliski, Chapter 4.1.1; David Lowe, IJCV 2004

Image Representations: Histograms





Global histogram to represent distribution of features

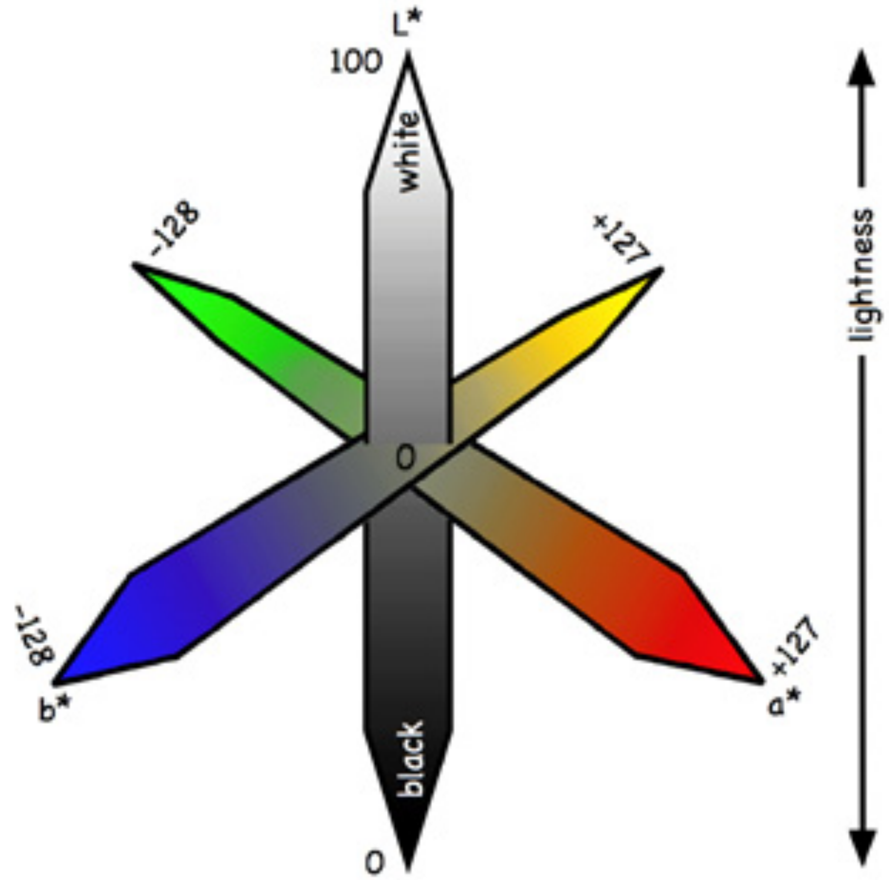
– Color, texture, depth, …

Local histogram per detected point

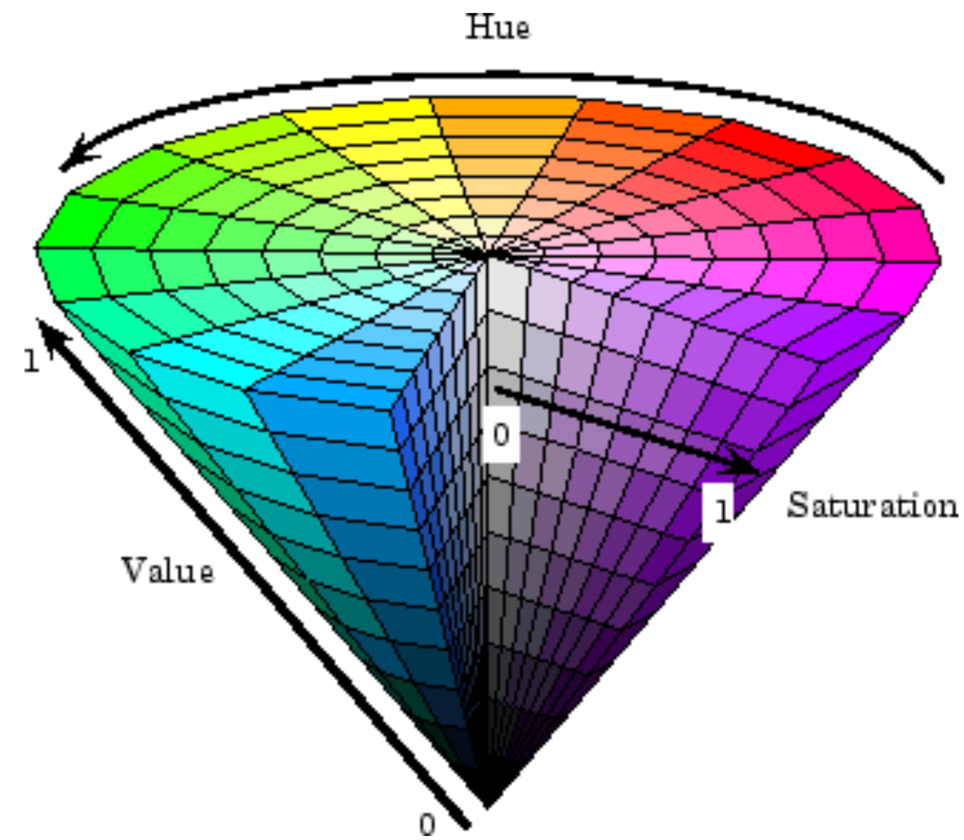
Images from Dave Kauchak

For what things do we compute histograms?

- Color



L*a*b* color space



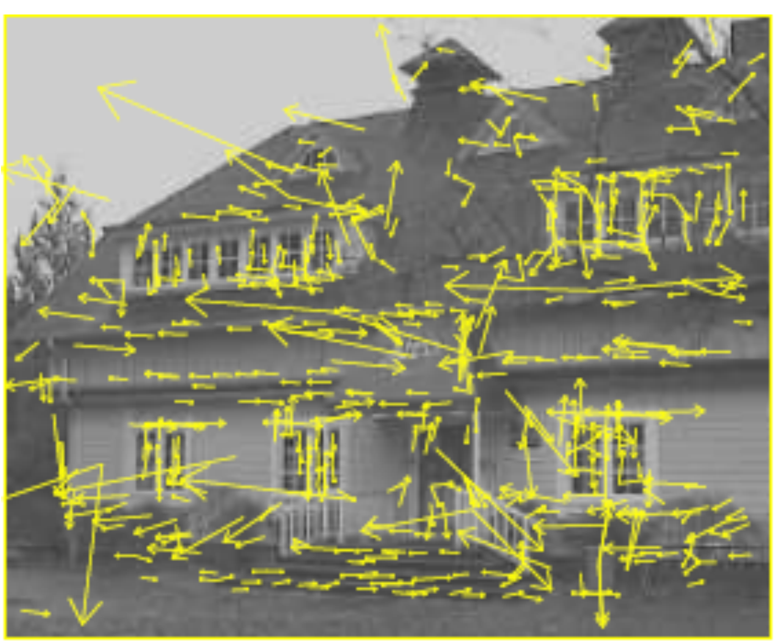
HSV color space

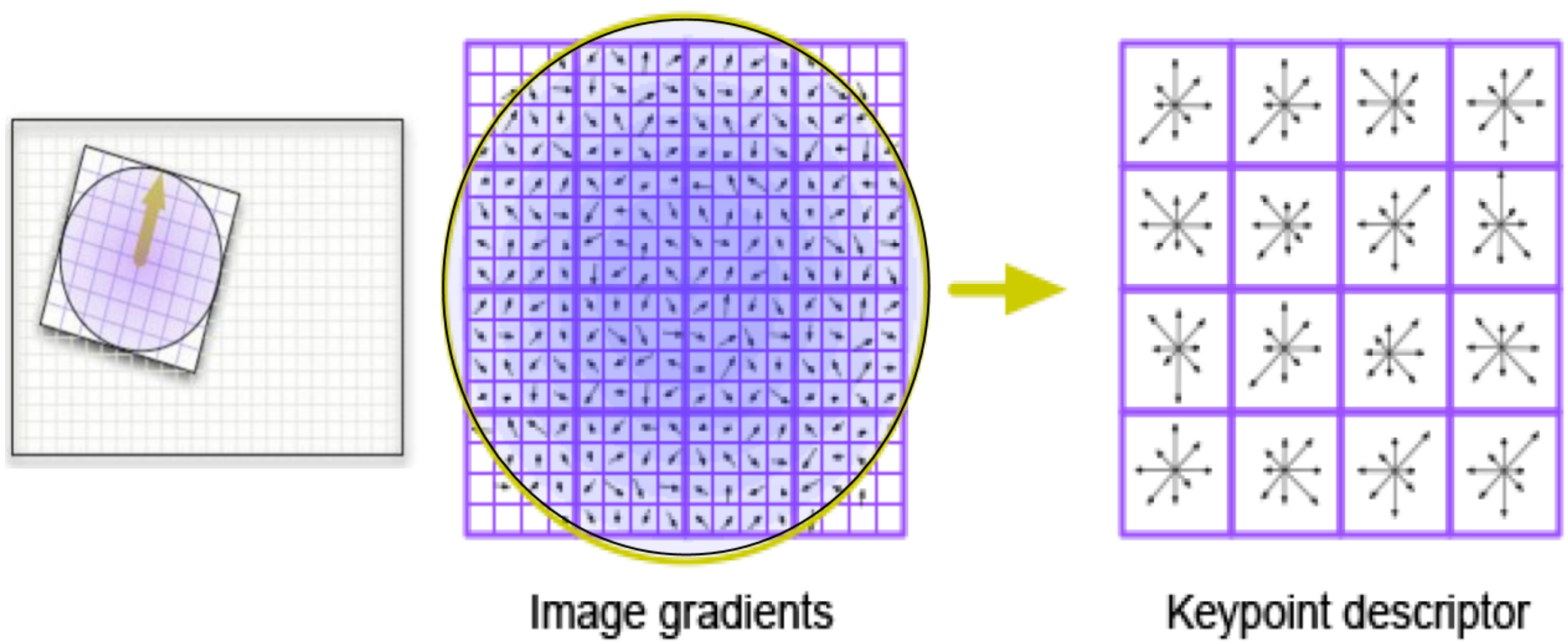
- Model local appearance

James Hays

For what things do we compute histograms?

- Texture
- Local histograms of oriented gradients
- SIFT: Scale Invariant Feature Transform
  - Extremely popular (40k citations)



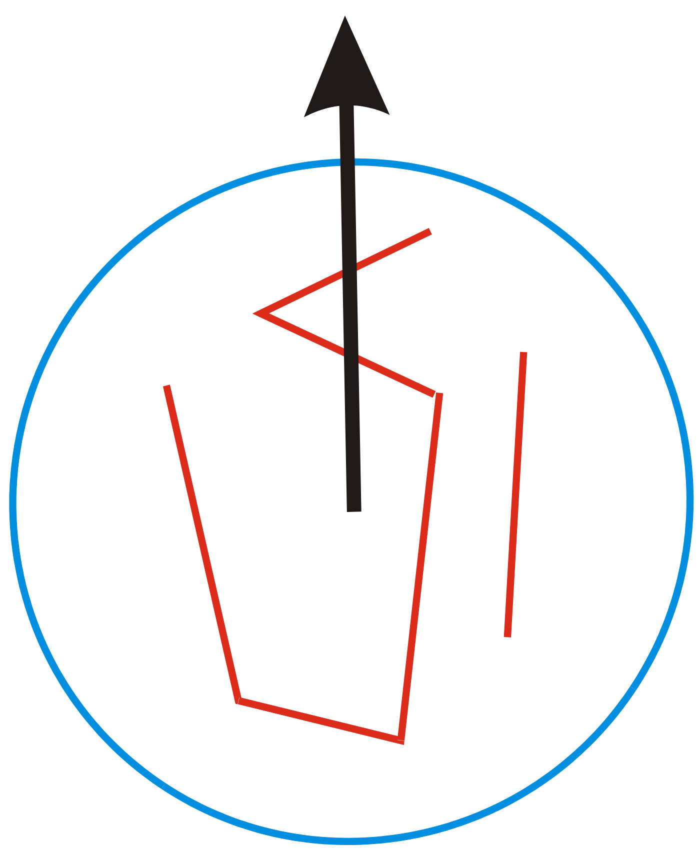


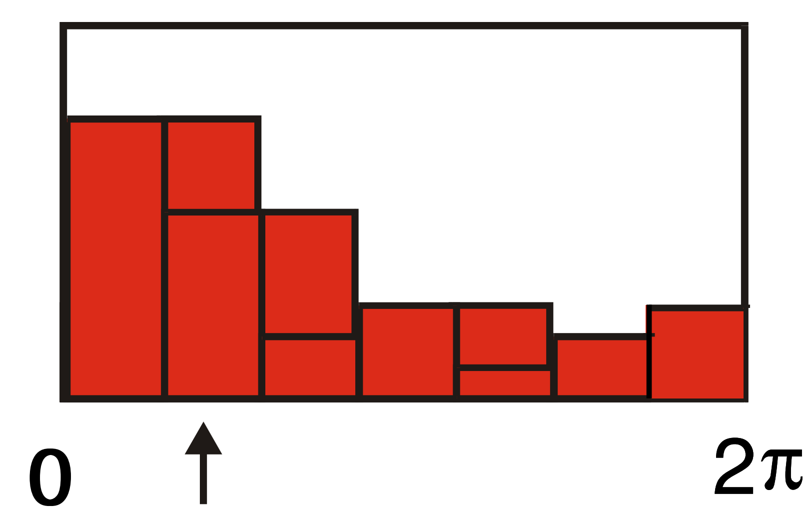
SIFT – Lowe IJCV 2004

James Hays

SIFT Orientation Normalization

- Compute orientation histogram
- Select dominant orientation Θ
- Normalize: rotate to fixed orientation

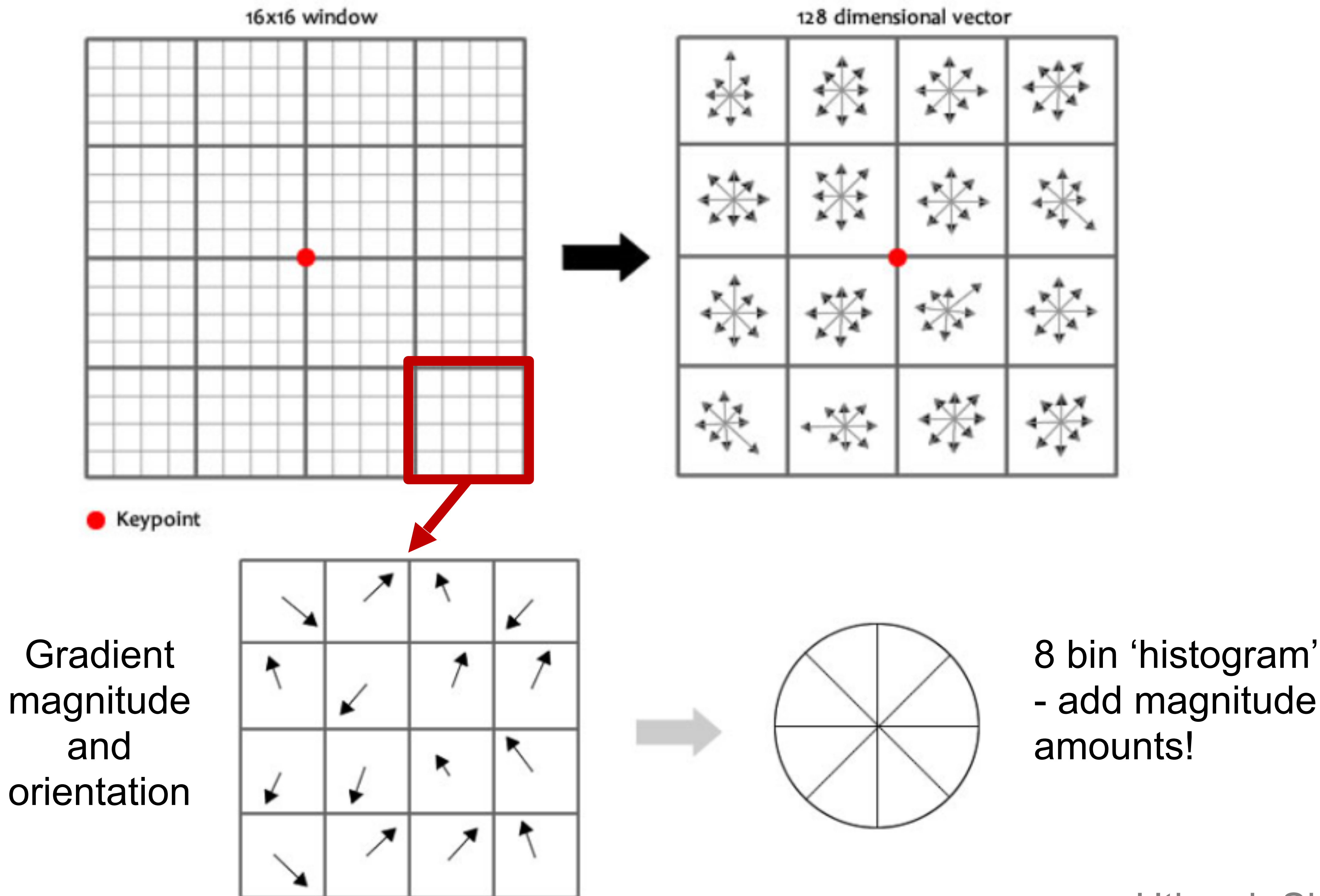




[Lowe, SIFT, 1999]

SIFT Descriptor Extraction

- Given a keypoint with scale and orientation

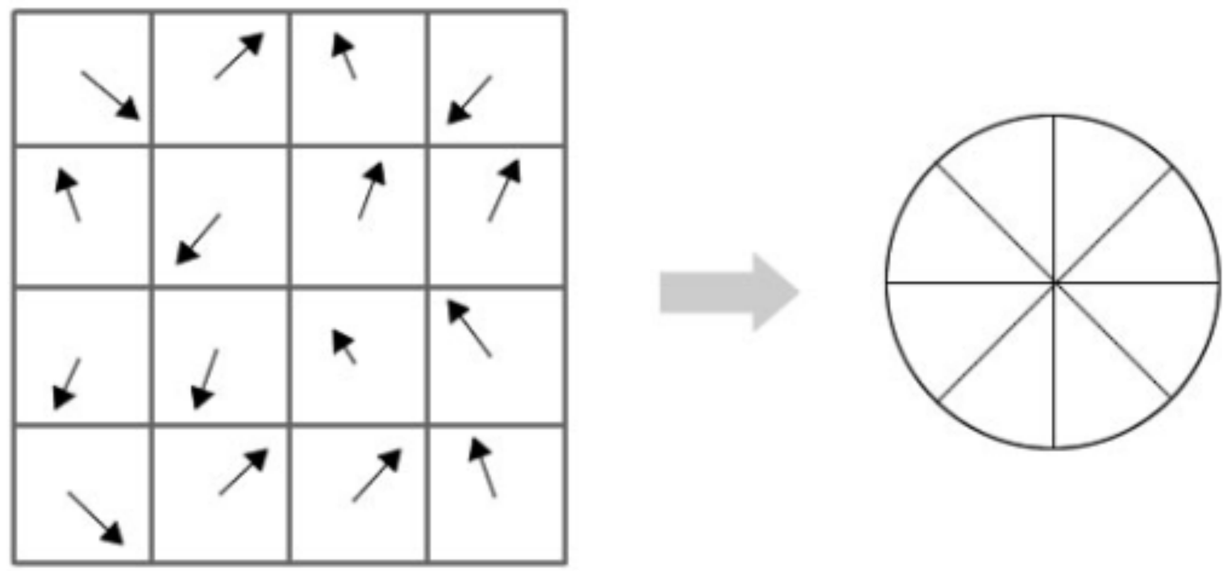


Utkarsh Sinha

SIFT Descriptor Extraction

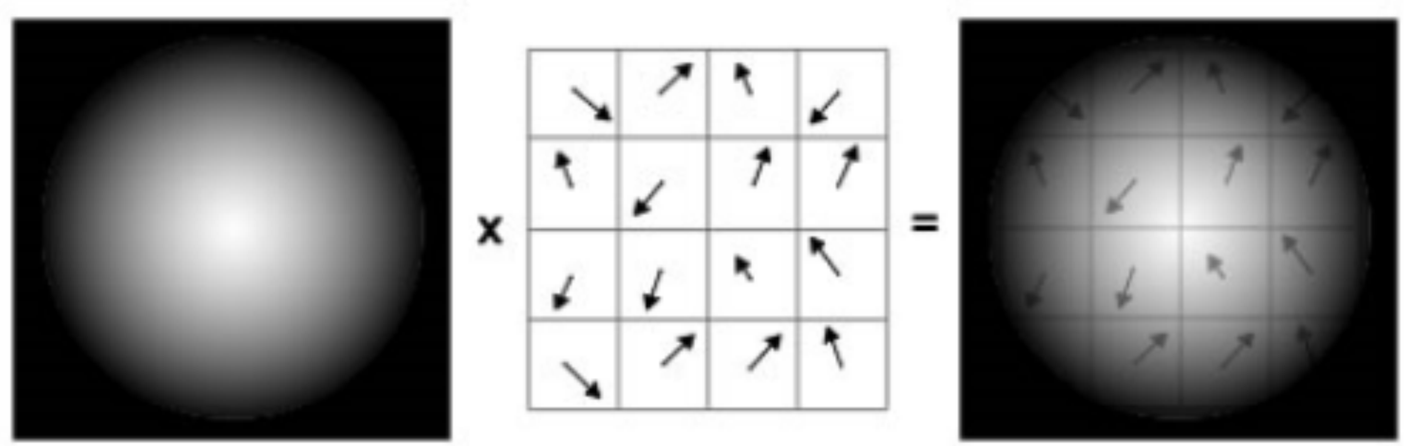
- Within each 4x4 window

Gradient magnitude and orientation



8 bin 'histogram' - add magnitude amounts!

Weight magnitude that is added to 'histogram' by Gaussian

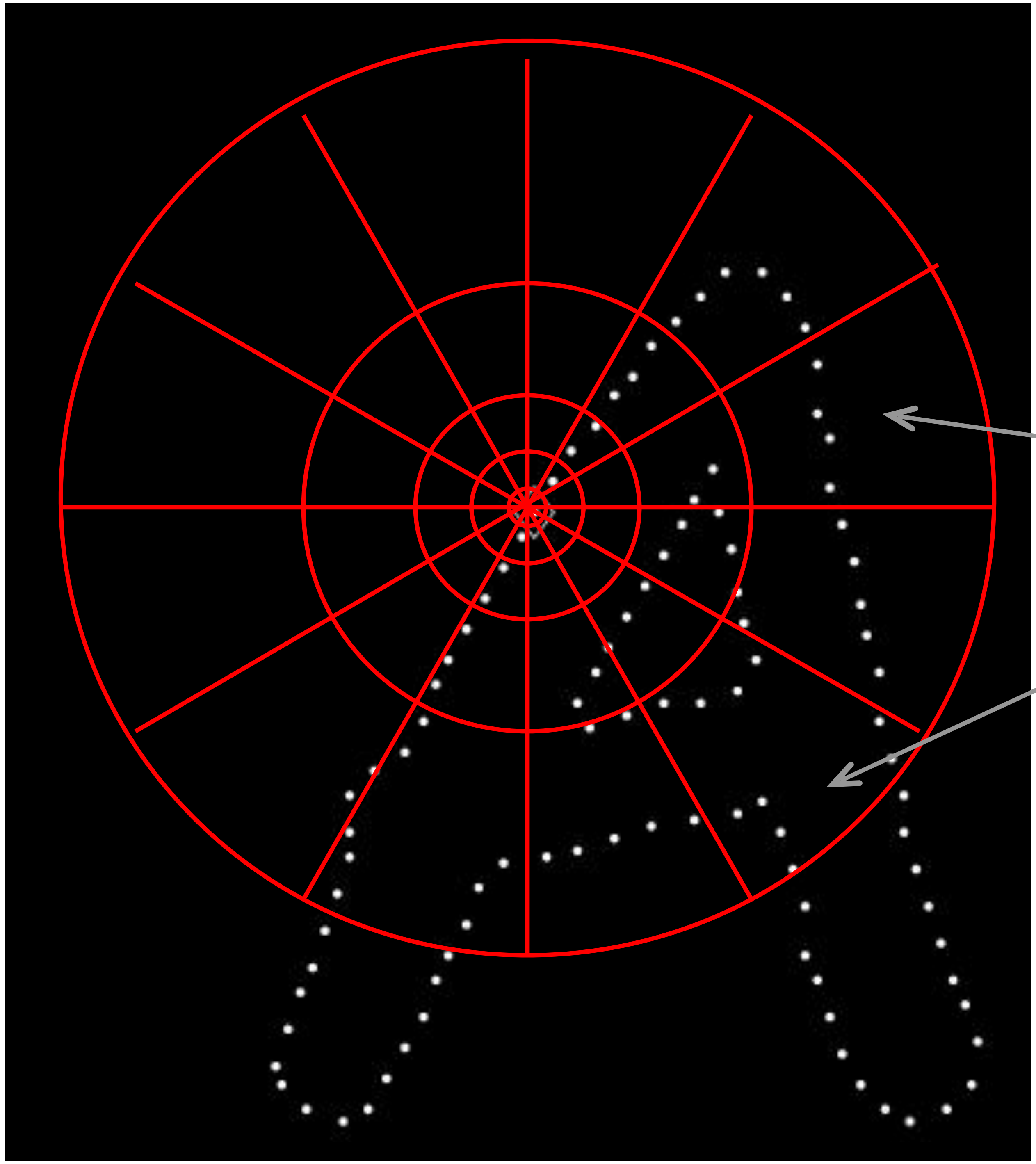


Utkarsh Sinha

SIFT Descriptor Extraction

- Extract 8 x 16 values into 128-dim vector
- Illumination invariance:
  - Working in gradient space, so robust to I = I + b
  - Normalize vector to [0…1]
    - Robust to I = αI brightness changes
  - Clamp all vector values > 0.2 to 0.2.
    - Robust to “non-linear illumination effects”
    - Image value saturation / specular highlights
  - Renormalize

Local Descriptors: Shape Context



Count the number of points inside each bin, e.g.:

Count = 4

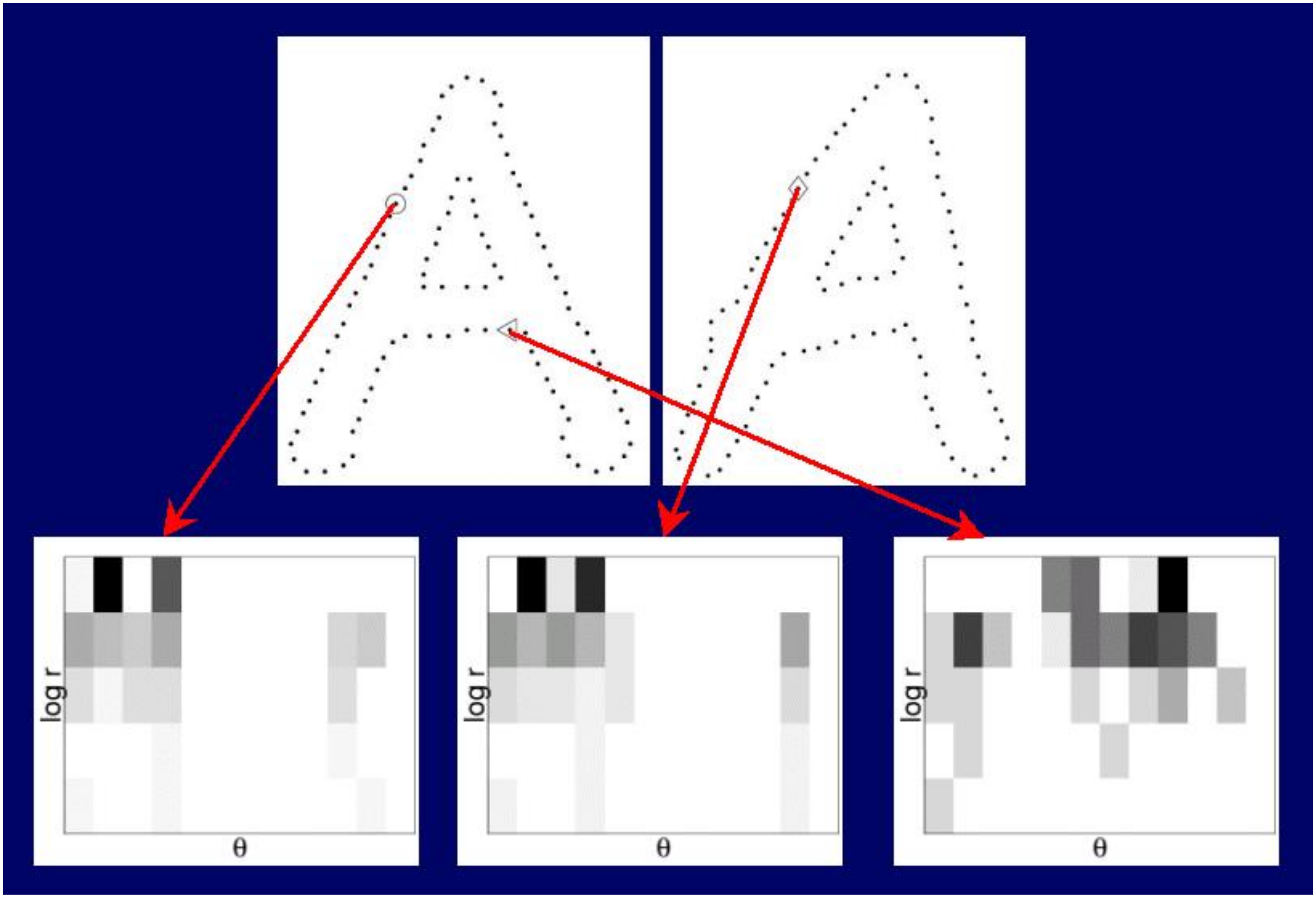
⋮

Count = 10

Log-polar binning:
More precision for nearby points, more flexibility for farther points.

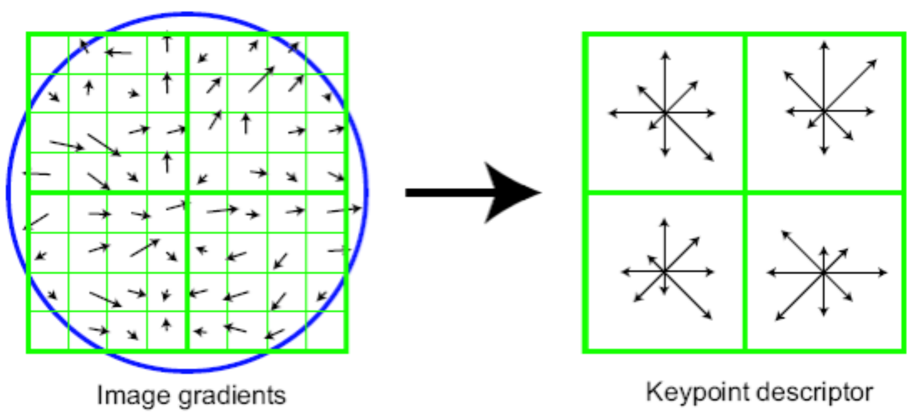
Belongie & Malik, ICCV 2001

Shape Context Descriptor



Review: Local Descriptors

- Most features can be thought of as templates, histograms (counts), or combinations
- The ideal descriptor should be
  - Robust and Distinctive
  - Compact and Efficient



- Most available descriptors focus on edge/ gradient information
  - Capture texture information
  - Color rarely used

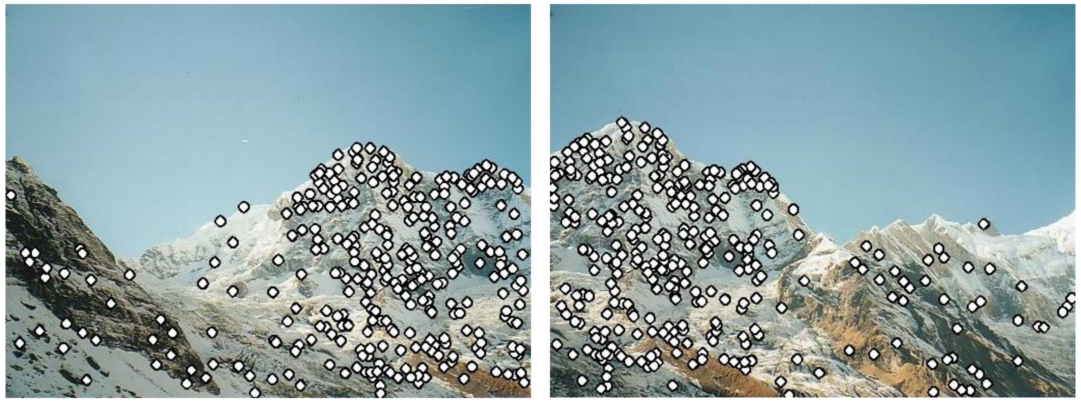
What we will learn today?

- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector
- Local features
  - SIFT
- Feature Matching

Some background reading:
Rick Szeliski, Chapter 4.1.1; David Lowe, IJCV 2004

Think-Pair-Share

- Design a feature point matching scheme.
- Two images, I_1 and I_2



- Two sets X_1 and X_2 of feature points
  - Each feature point x_1 has a descriptor

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

- Distance, bijective/injective/surjective, noise, confidence, computational complexity, generality…

Euclidean distance vs. Cosine Similarity

- Euclidean distance:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

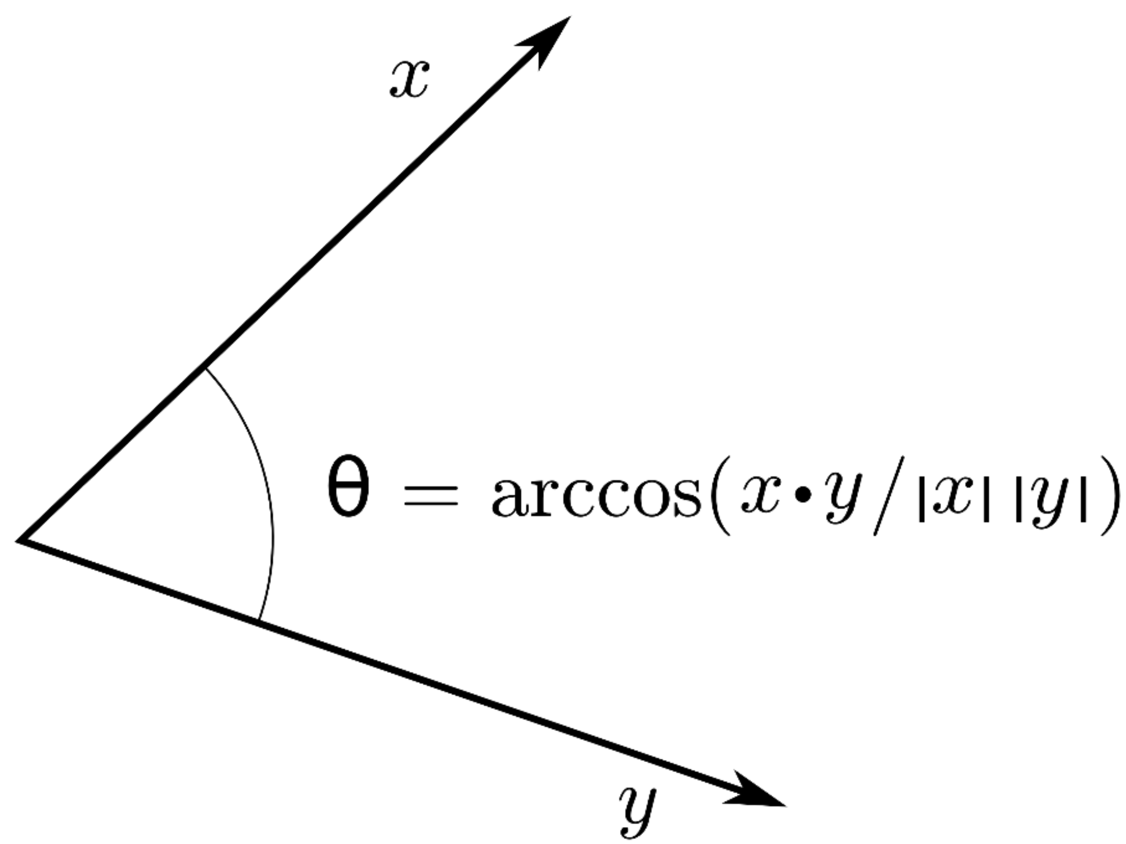
$$= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

$$\|\mathbf{q}-\mathbf{p}\| = \sqrt{(\mathbf{q}-\mathbf{p})\cdot(\mathbf{q}-\mathbf{p})}.$$

- Cosine similarity:

$$\mathbf{a}\cdot\mathbf{b} = \|\mathbf{a}\|_2\|\mathbf{b}\|_2\cos\theta$$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|_2\|\mathbf{B}\|_2}$$



Wikipedia

Feature Matching

- Criteria 1:
  - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)

- Problems:
  - Does everything have a match?

Feature Matching

- Criteria 2:
  - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)
  - Ignore anything higher than threshold (no match!)

- Problems:
  - Threshold is hard to pick
  - Non-distinctive features could have lots of close matches, only one of which is correct

Nearest Neighbor Distance Ratio

Compare distance of closest (NN1) and second-closest (NN2) feature vector neighbor.

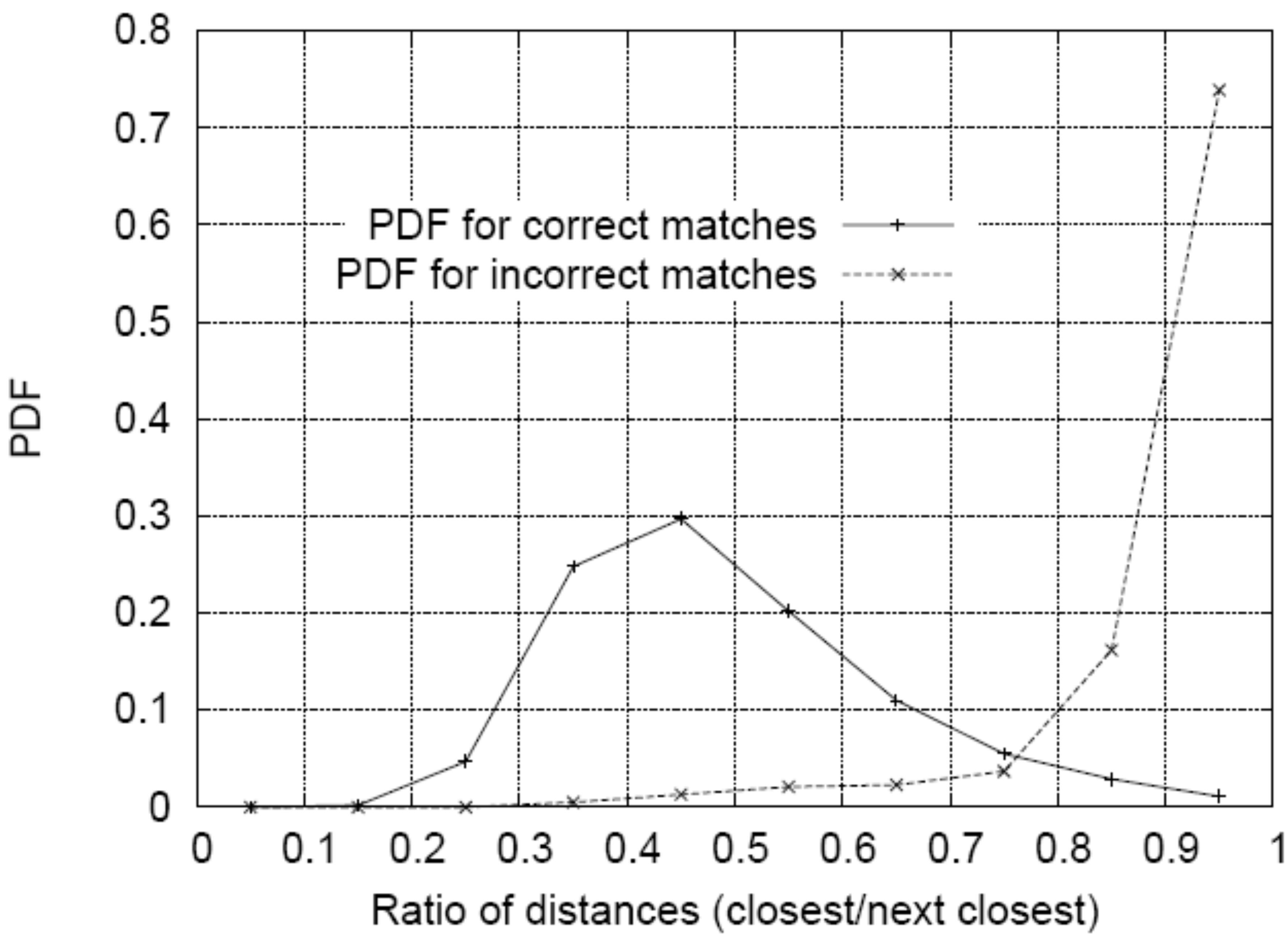
- If NN1 ≈ NN2, ratio will be ≈ 1 -> matches too close.
- As NN1 << NN2, ratio tends to 0.

Sorting by this ratio puts matches in order of confidence.

Threshold ratio – but how to choose?

Nearest Neighbor Distance Ratio

- Lowe computed a probability distribution functions of ratios
- 40,000 keypoints with hand-labeled ground truth



Ratio threshold depends on your application's view on the trade-off between the number of false positives and true positives!

Lowe IJCV 2004

Efficient compute cost

- Naïve looping: Expensive

- Operate on matrices of descriptors
- E.g., for row vectors,

  `features_image1 * features_image2`^{T}

  produces matrix of dot product results for all pairs of features