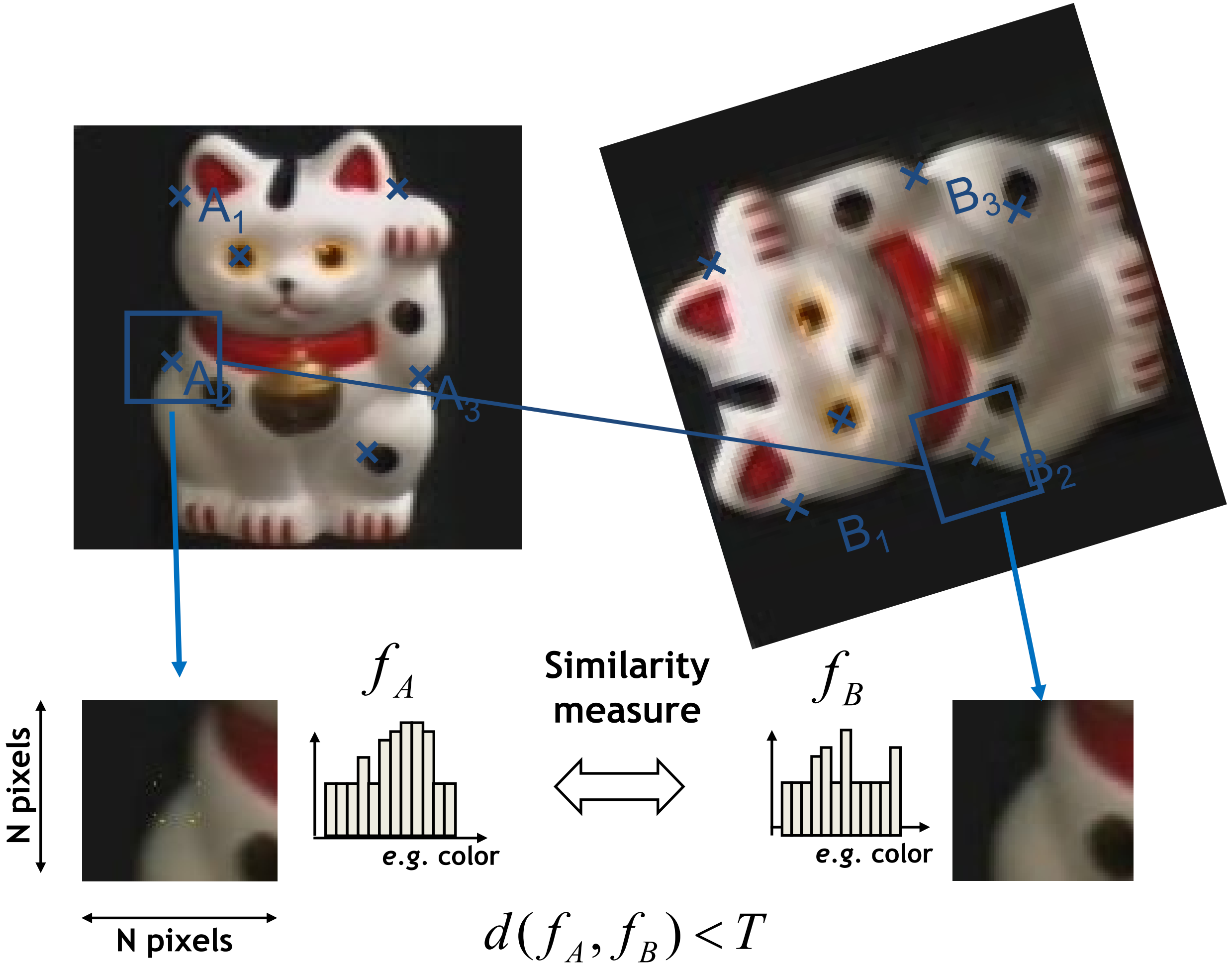
# ECE 4973: Lecture 13
# Local feature extraction

Slide credits: James Tompkin, Juan Carlos Niebles and Ranjay Krishna

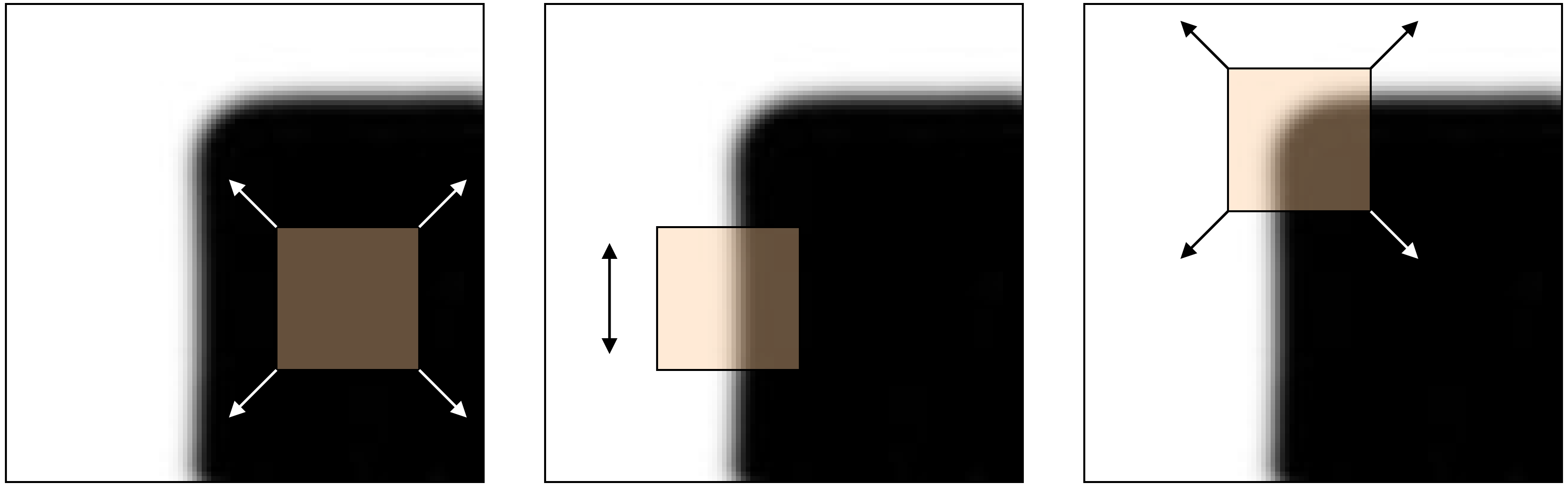
# General Approach




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

# Quick review: Harris Corner Detector



**"flat" region:**
**no change in all directions**

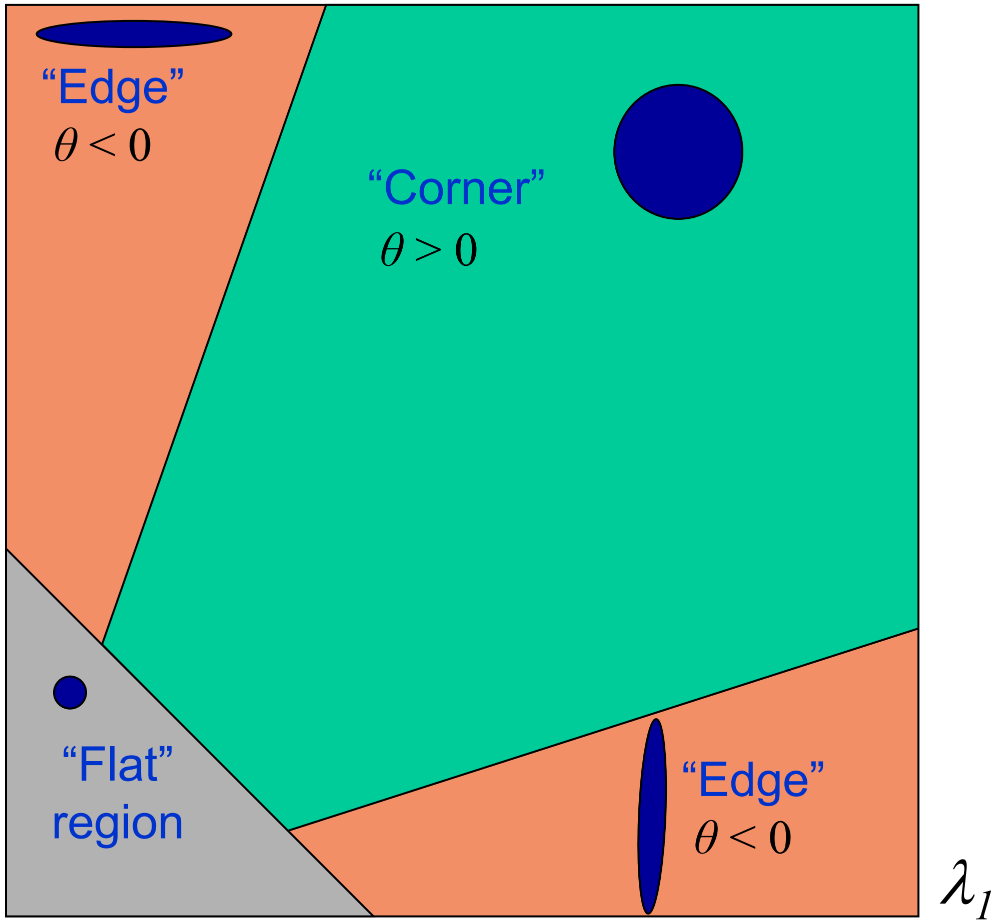
**"edge":**
**no change along the edge direction**

**"corner":**
**significant change in all directions**

Slide credit: Alyosha Efros

# Quick review: Harris Corner Detector

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




- Fast approximation
  - Avoid computing the eigenvalues
  - $\alpha$: constant (0.04 to 0.06)

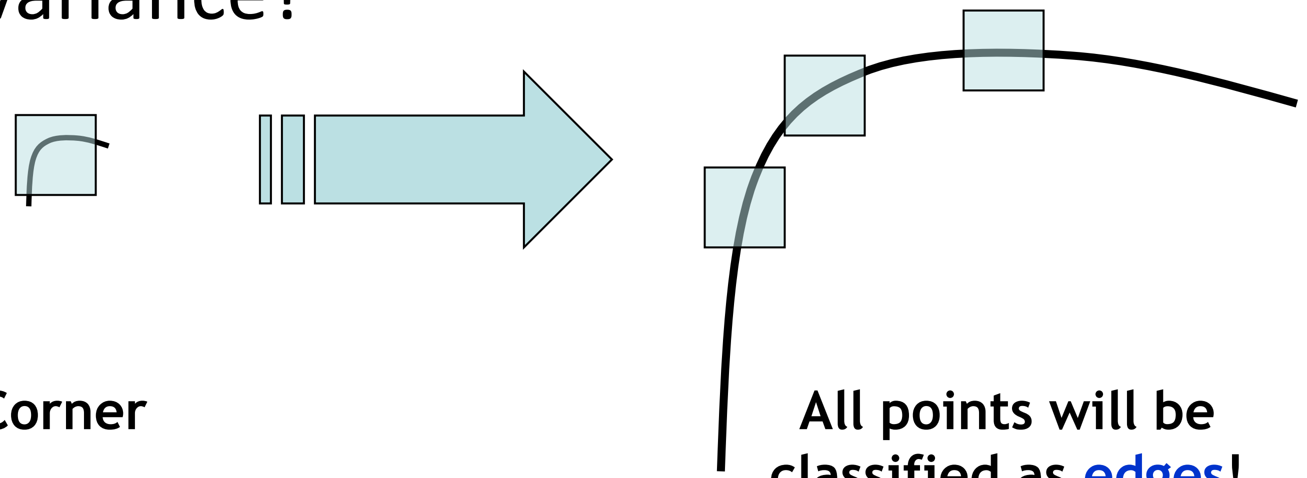
Slide credit: Kristen Grauman

# Quick review: Harris Corner Detector

Slide adapted from Darya Frolova, Denis Simakov

# Quick review: Harris Corner Detector

- Translation invariance
- Rotation invariance
- Scale invariance?




**Not invariant to image scale!**

Slide credit: Kristen Grauman

# WHAT IS THE ‘SCALE’ OF A FEATURE POINT?

# Automatic Scale Selection

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

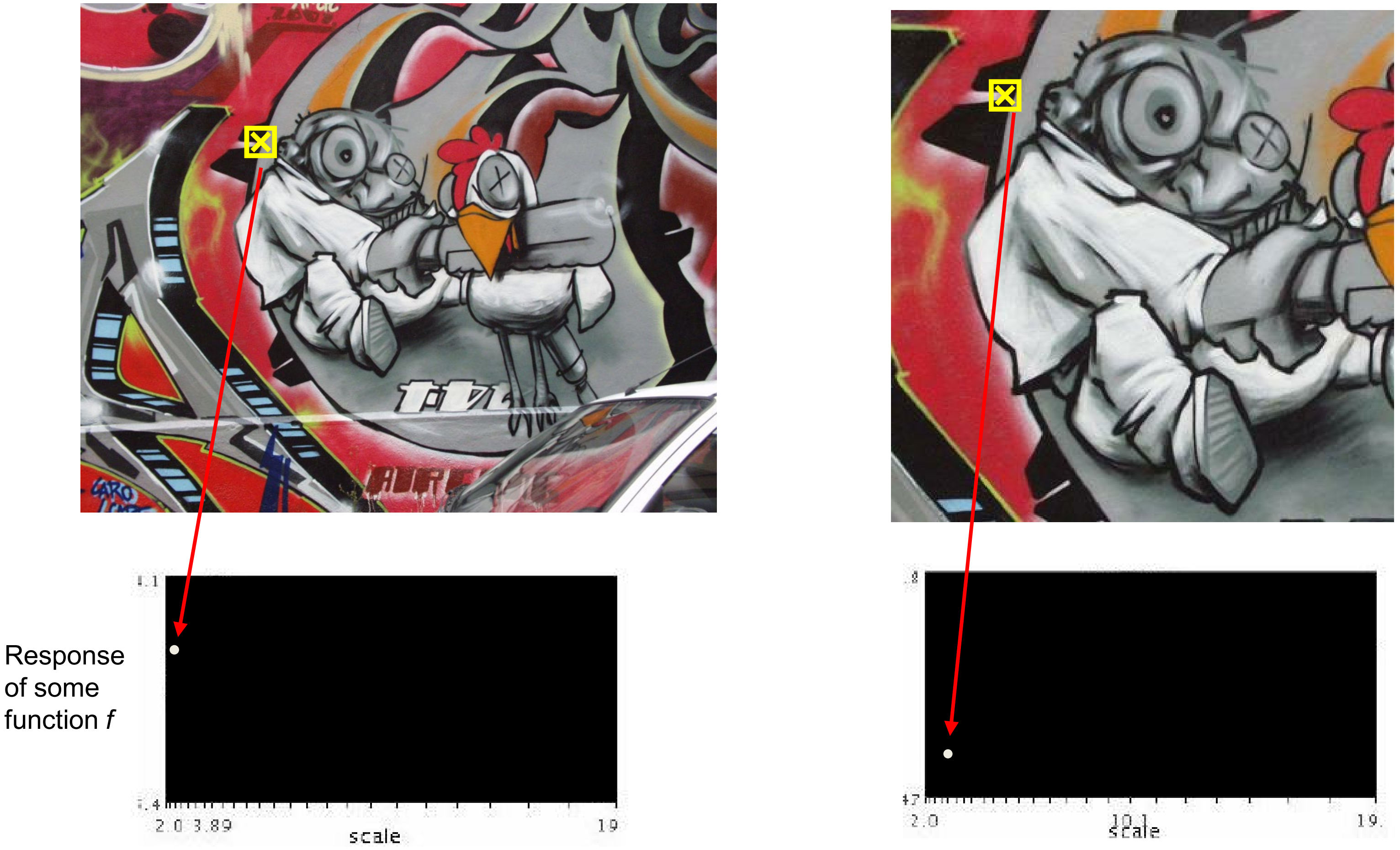
How to find patch sizes at which *f* response is equal?

What is a good *f* ?

K. Grauman, B. Leibe

# Automatic Scale Selection

- Function responses for increasing scale (scale signature)



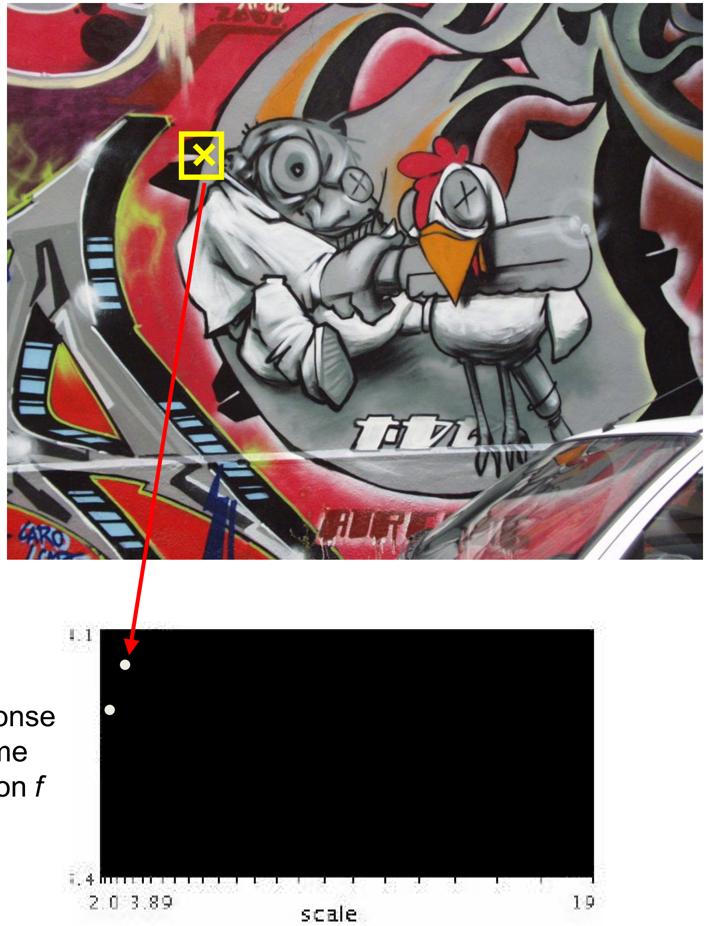

$f(I_{i_1 \ldots i_m}(x,\sigma))$ $\quad$ $f(I_{i_1 \ldots i_m}(x',\sigma'))$

K. Grauman, B. Leibe

# Automatic Scale Selection

- Function responses for increasing scale (scale signature)

Response
of some
function *f*



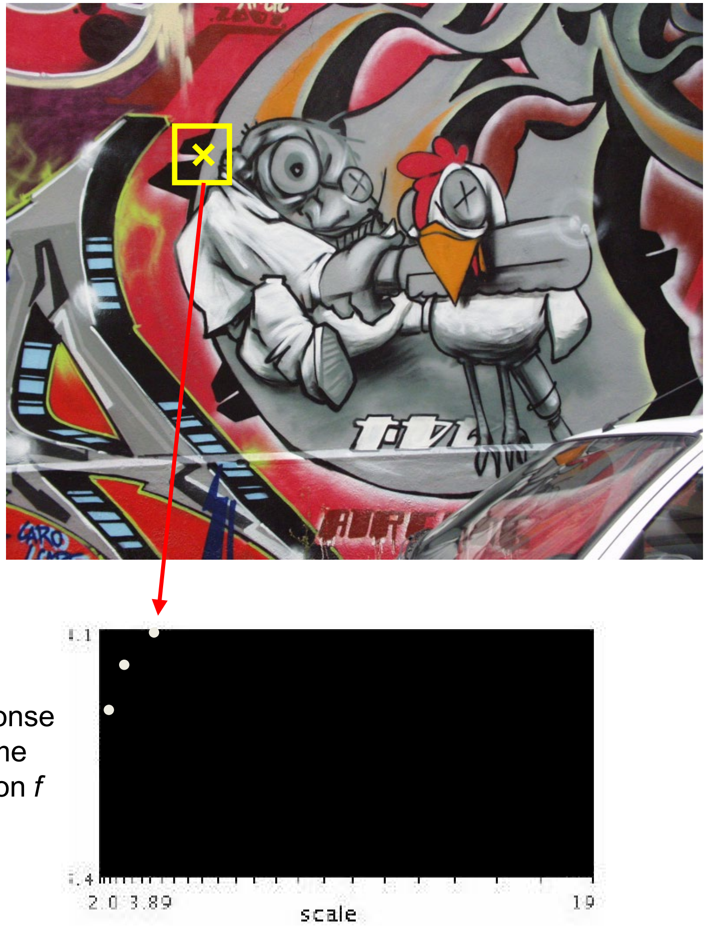

$f(I_{i_1 \ldots i_m}(x, \sigma))$


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

K. Grauman, B. Leibe

# Automatic Scale Selection

- Function responses for increasing scale (scale signature)




Response of some function *f*

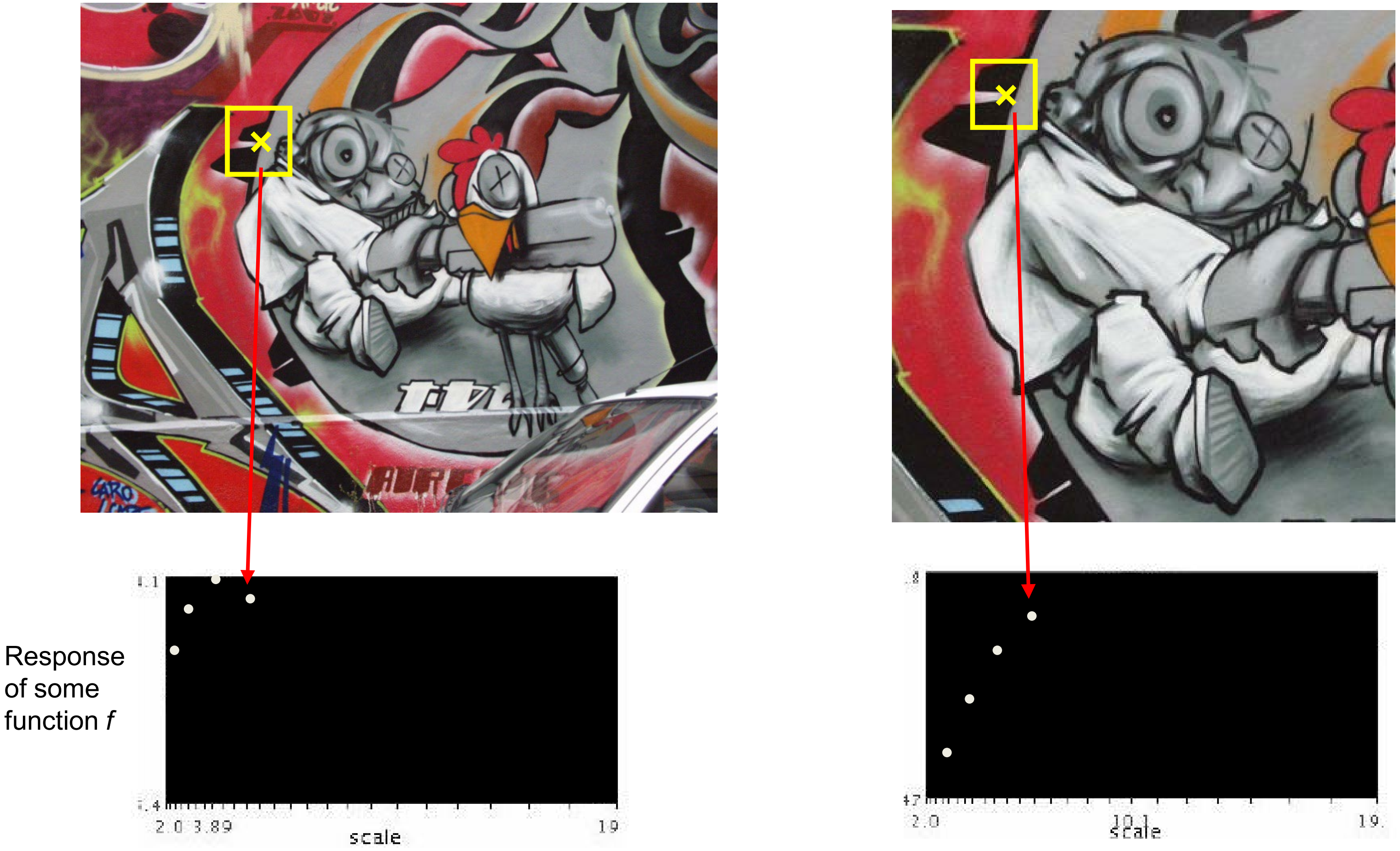
$f(I_{i_1 \ldots i_m}(x,\sigma))$


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

K. Grauman, B. Leibe

# Automatic Scale Selection

- Function responses for increasing scale (scale signature)

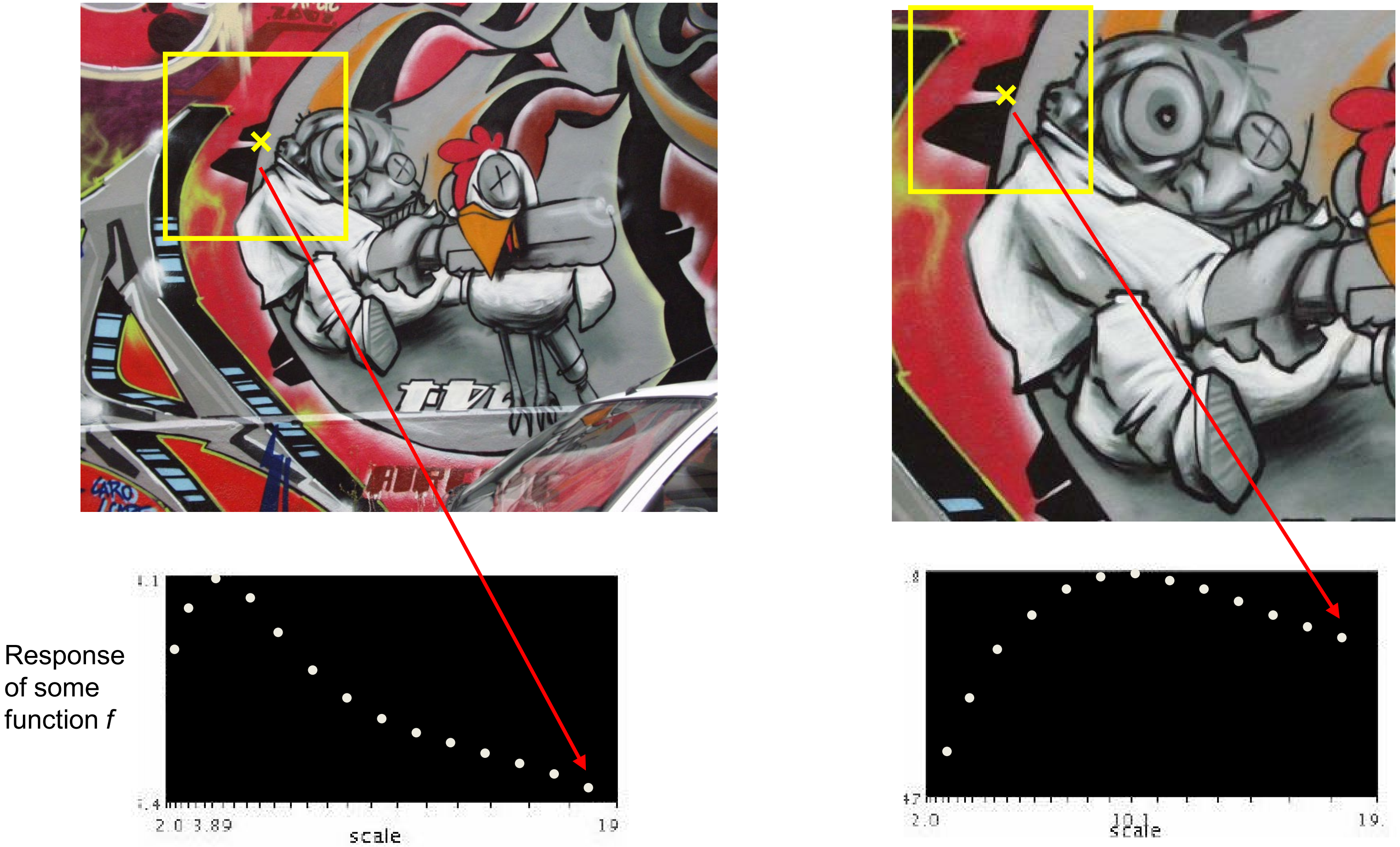



$f(I_{i_1 \ldots i_m}(x,\sigma))$ $\quad$ $f(I_{i_1 \ldots i_m}(x',\sigma'))$

K. Grauman, B. Leibe

# Automatic Scale Selection

- Function responses for increasing scale (scale signature)



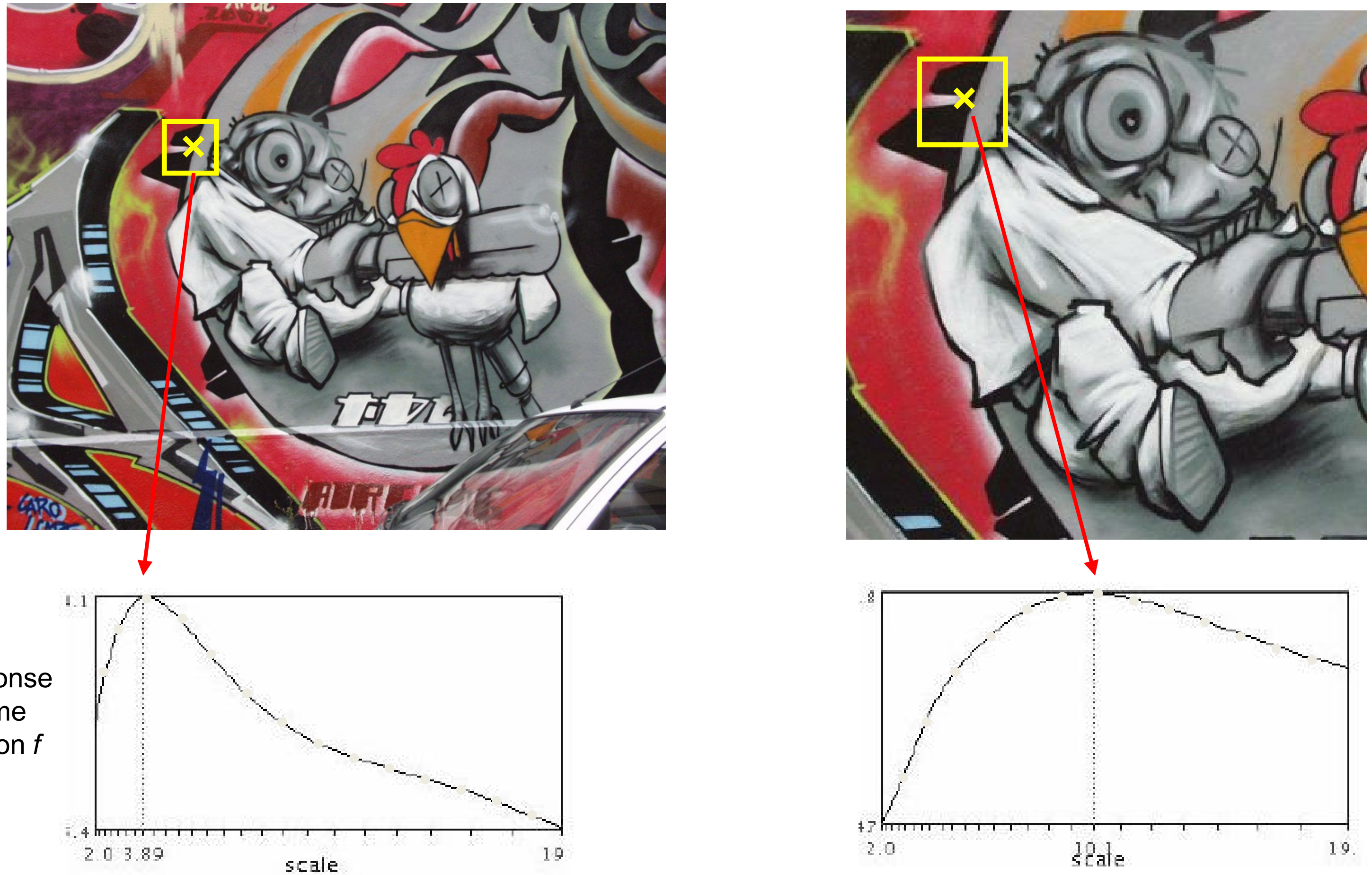

$f(I_{i_1 \ldots i_m}(x, \sigma))$ $f(I_{i_1 \ldots i_m}(x', \sigma'))$

K. Grauman, B. Leibe

# Automatic Scale Selection

- Function responses for increasing scale (scale signature)

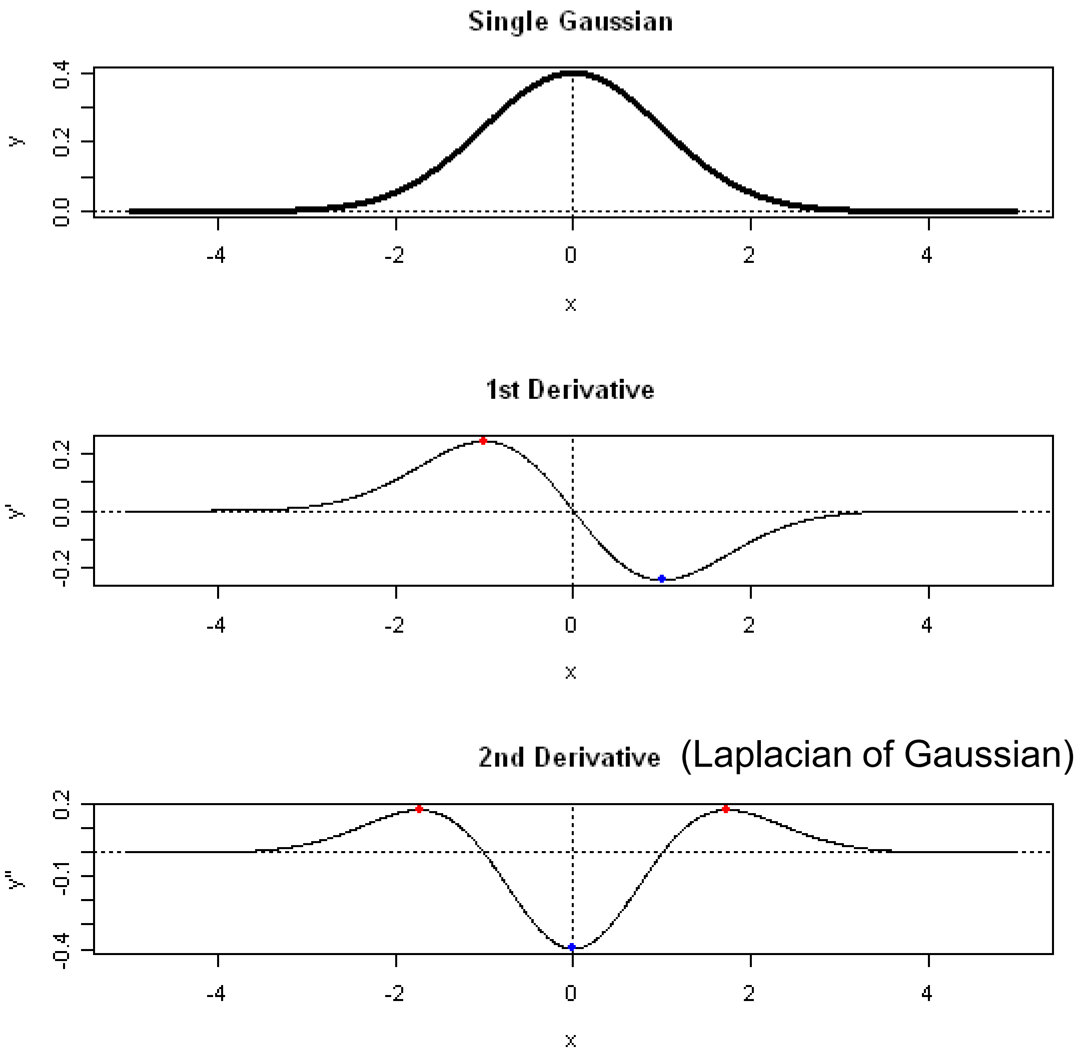
Response of some function *f*



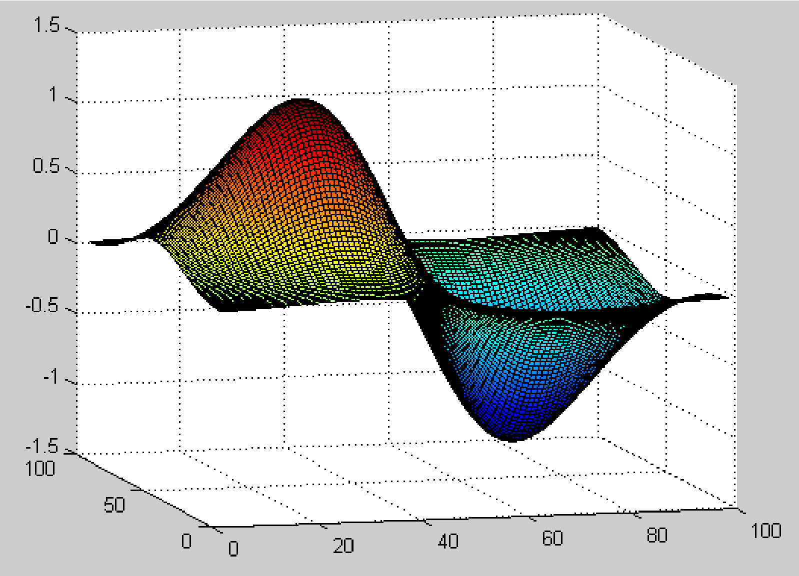

$f(I_{i_1 \ldots i_m}(x, \sigma))$ $f(I_{i_1 \ldots i_m}(x', \sigma'))$

K. Grauman, B. Leibe

# What Is A Useful Signature Function *f* ?



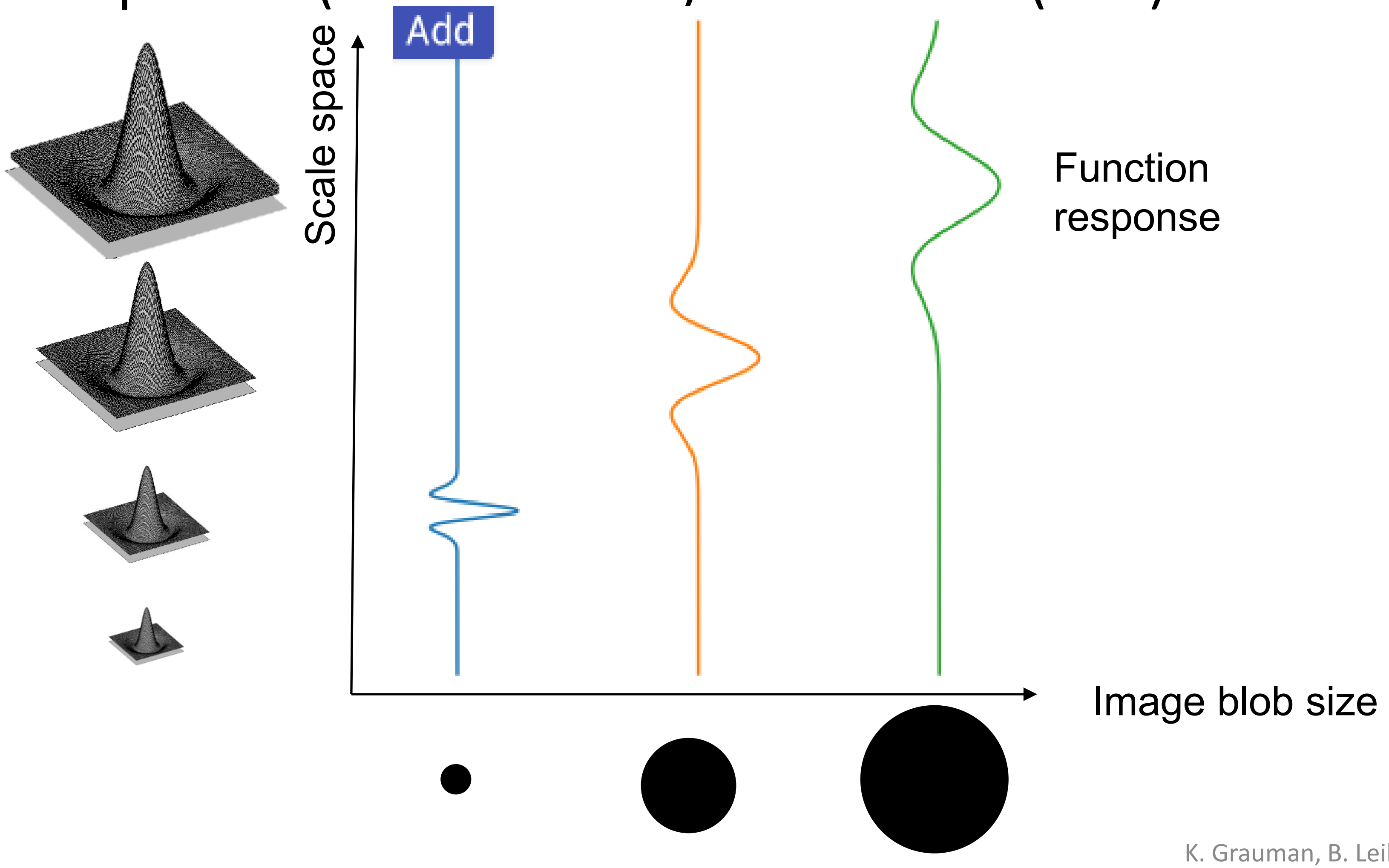

## 1$^{st}$ Derivative of Gaussian




Earl F. Glynn

# What Is A Useful Signature Function *f* ?

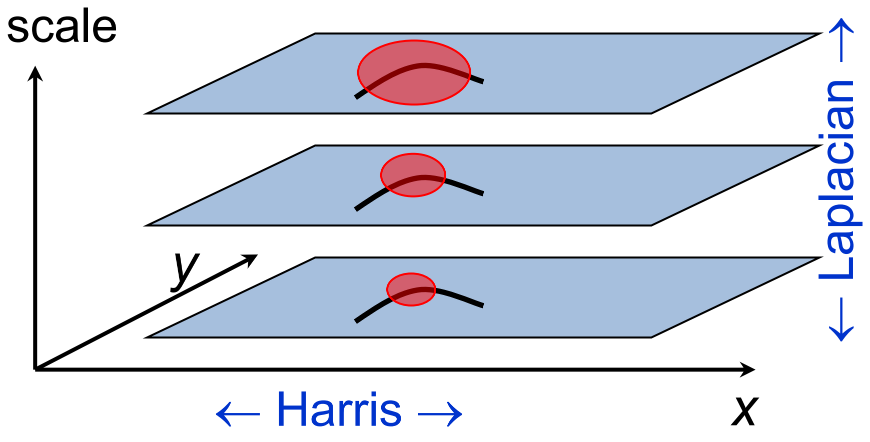
- "Blob" detector is common for corners
  - - Laplacian (2nd derivative) of Gaussian (LoG)




K. Grauman, B. Leibe

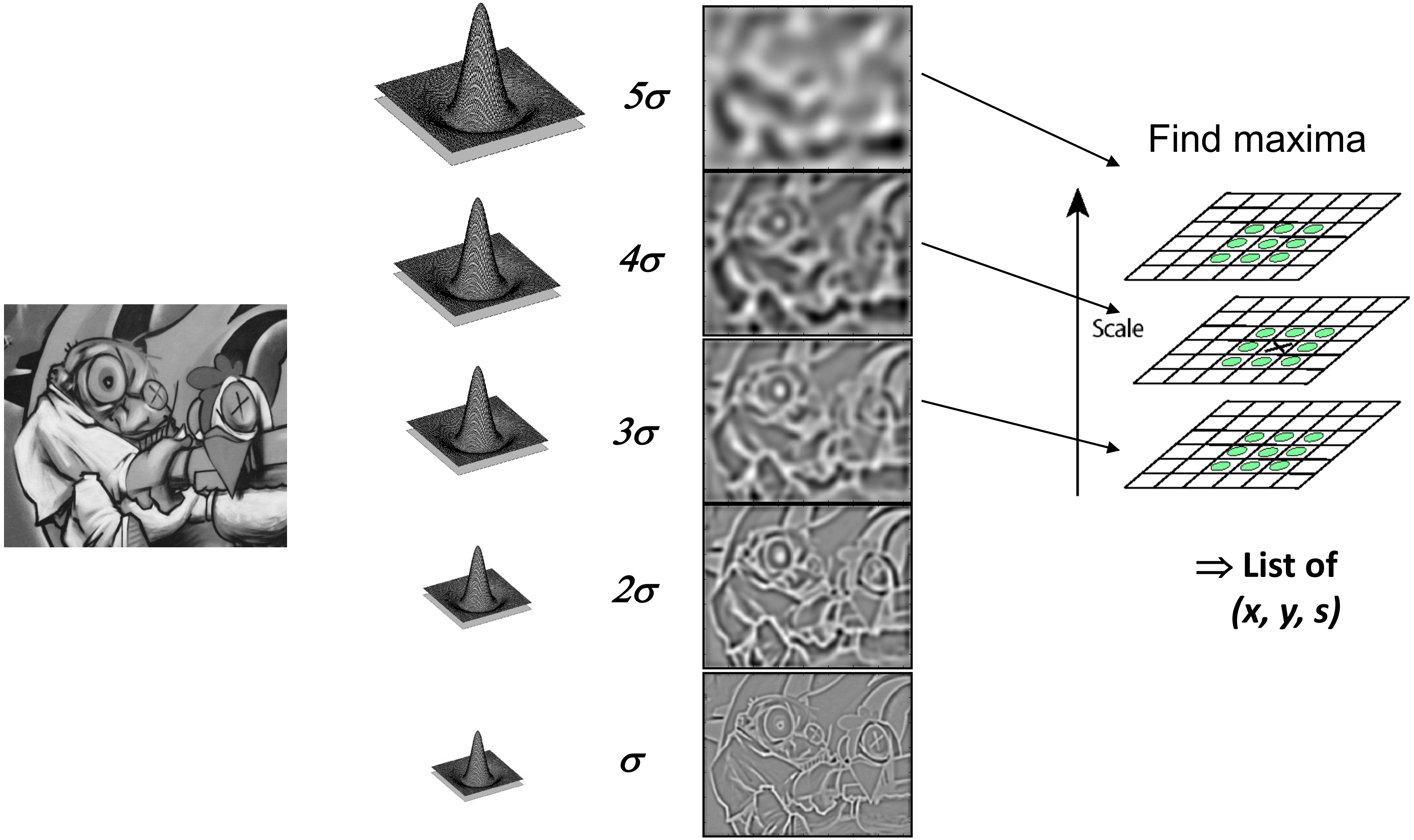
# Scale Invariant Detectors

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

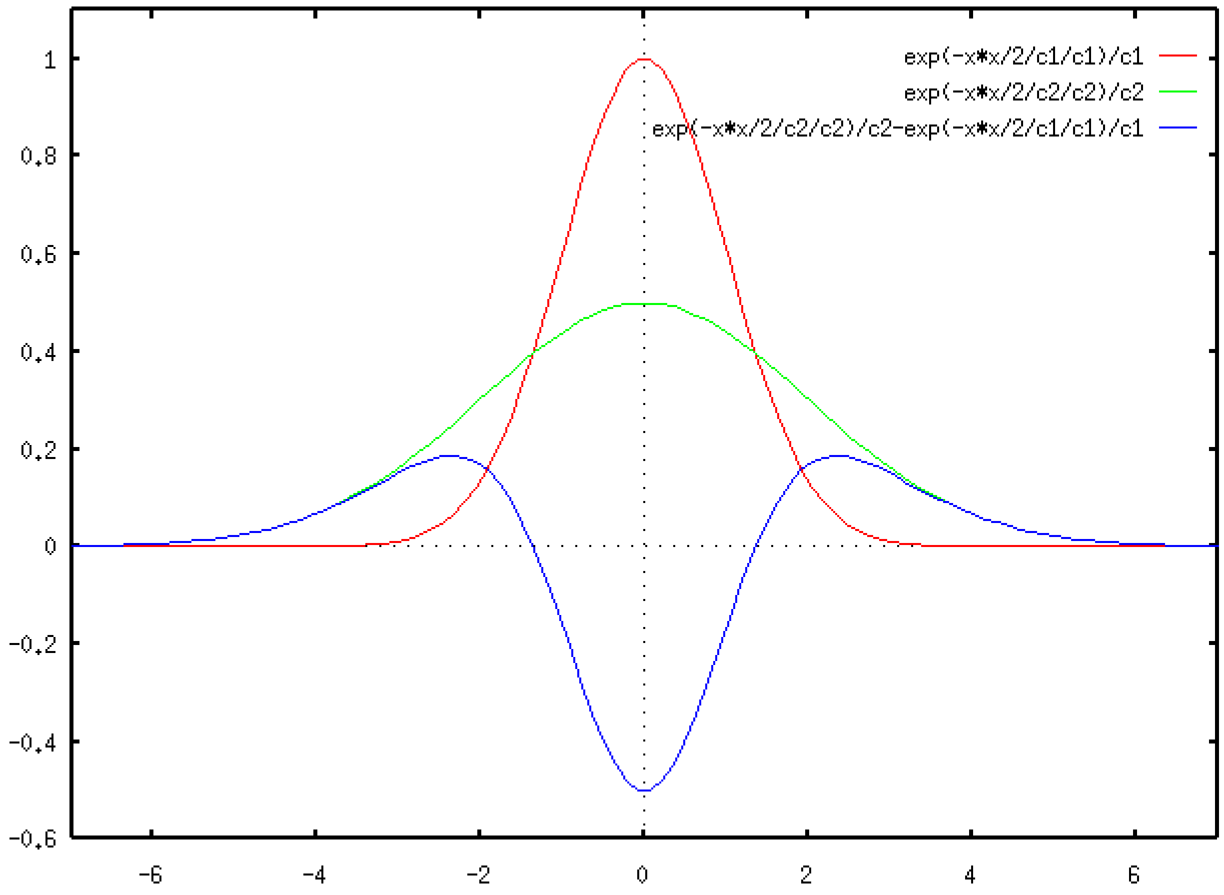
# Find local maxima in position-scale space




K. Grauman, B. Leibe

# Alternative approach

Approximate LoG with Difference-of-Gaussian (DoG).



# Scale Invariant Detection

- Functions for determining scale $f = \text{Kernel} * \text{Image}$

Kernels:

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

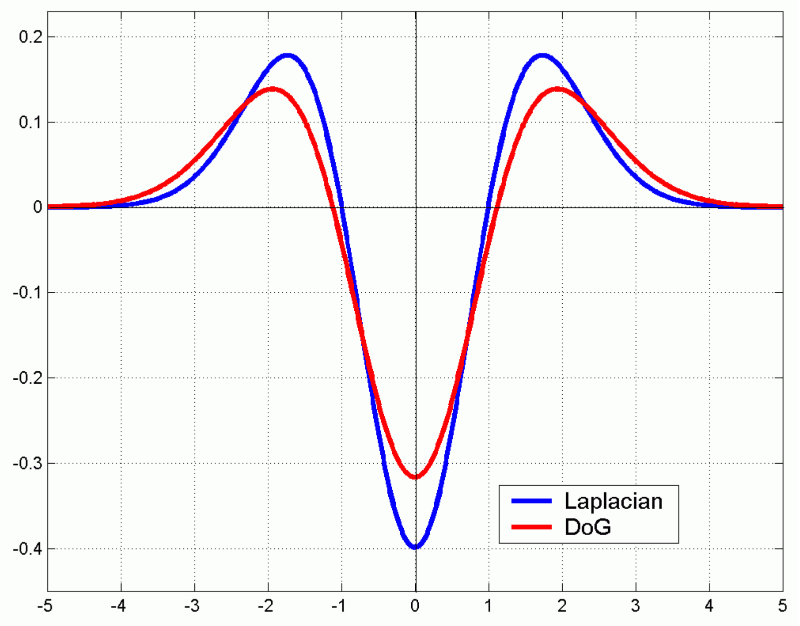
(Laplacian)

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

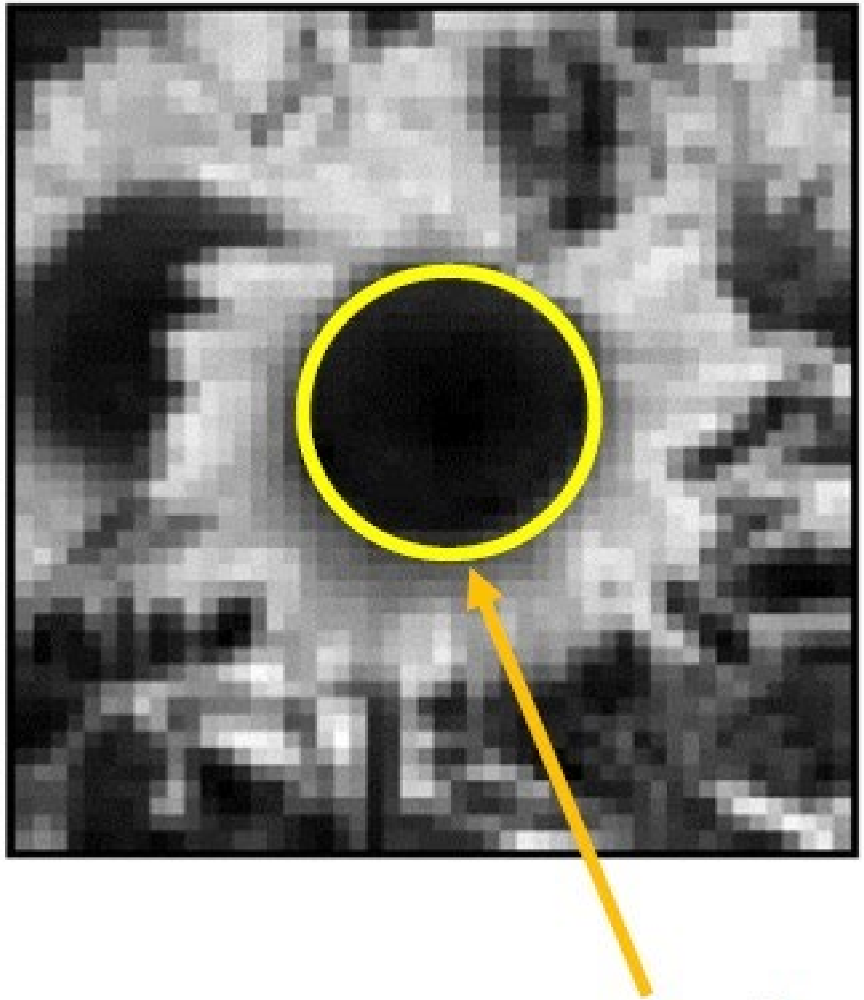
(Difference of Gaussians)

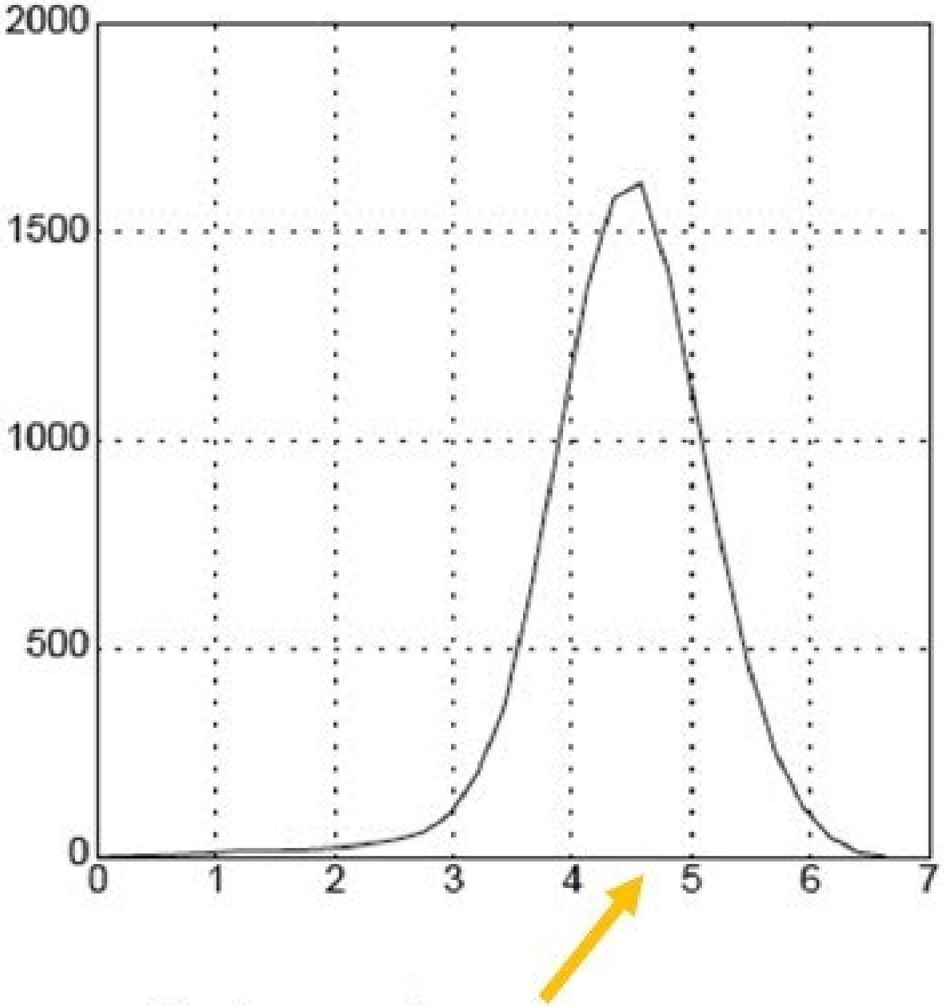
where Gaussian

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



# Laplacian

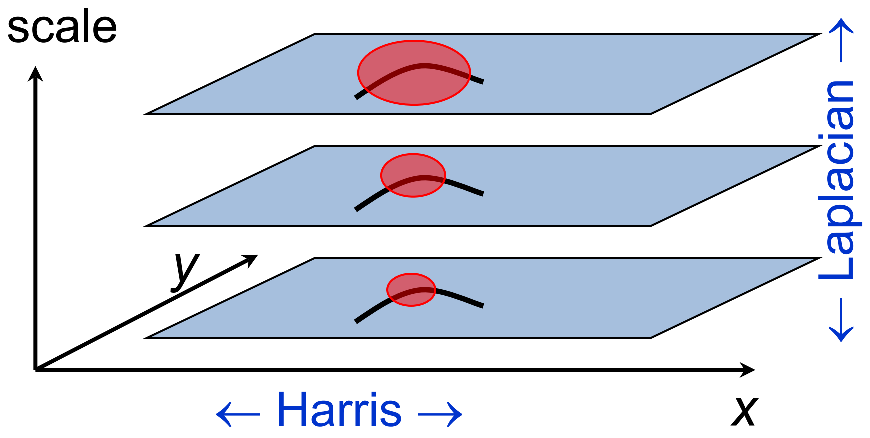




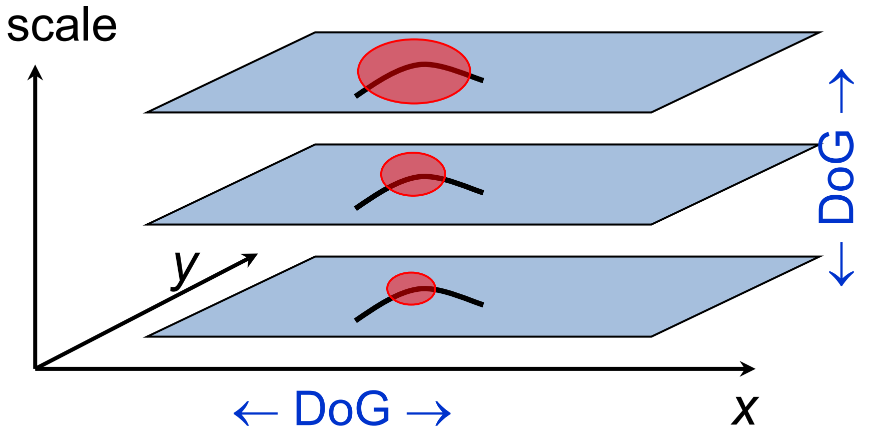

Characteristic scale

# Scale Invariant Detectors

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




- SIFT (Lowe)[2]
  *Find local maximum of:*
  - Difference of Gaussians in space and scale
  - Post-processing to remove "outliers"



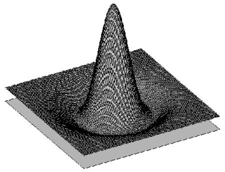

[1] K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001
[2] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004

# Alternative approach

Approximate LoG with Difference-of-Gaussian (DoG).

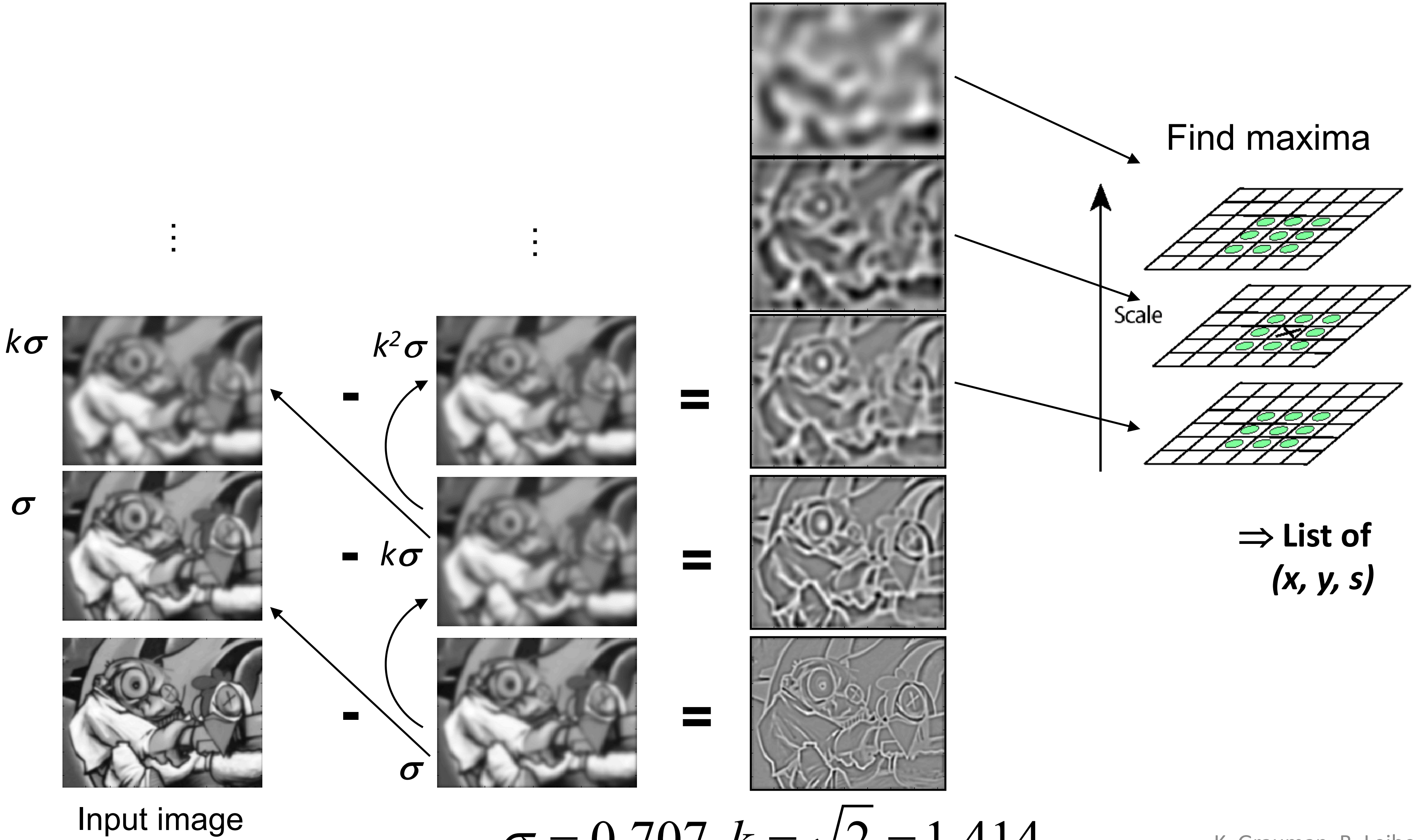
Don’t get confused with Derivative of Gaussian

1. Blur image with $\sigma$ Gaussian kernel
2. Blur image with $k\sigma$ Gaussian kernel
3. Subtract 2. from 1.



K. Grauman, B. Leibe

# Find local maxima in position-scale space of DoG



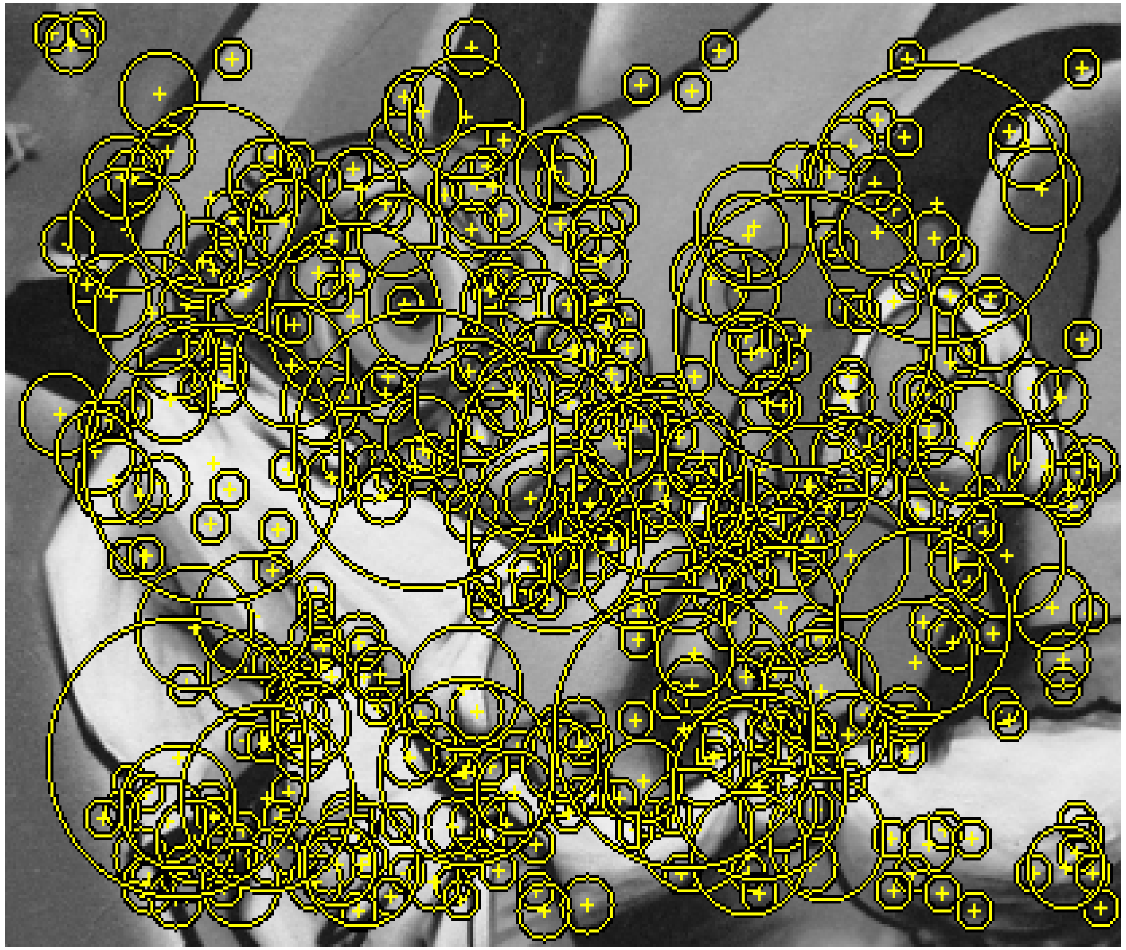

⇒ **List of *(x, y, s)***

$$\sigma = 0.707, k = \sqrt{2} = 1.414$$

K. Grauman, B. Leibe

# Results: Difference-of-Gaussian

- Larger circles = larger scale
- Descriptors with maximal scale response



K. Grauman, B. Leibe

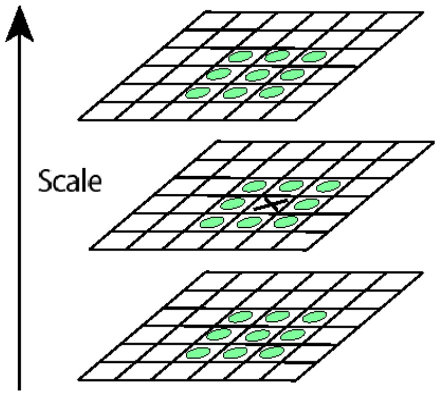
# Outlier Rejection

Avoid low contrast candidates (small magnitude extrema)

- Taylor series expansion of DoG from the center pixel

$$D(\mathbf{x}) = D_0 + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

where $\mathbf{x} = (x, y, \sigma)^T$




- Minima or maxima at $\mathbf{x}^* = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}$
- Iterate $\mathbf{x}^{(k+1)} \leftarrow -\frac{\partial^2 \mathrm{D}}{\partial \mathbf{x}^2}^{-1} \frac{\partial \mathrm{D}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}^{(k)}}$, discard candidates if
  - $X^{(k+1)}$ does not converge
  - $|D(x^*)|$< th(~0.03)

# Further Outlier Rejection

Remove edge-like points

- Use trick similar to Harris corner detector
- Compute Hessian of *D*

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \qquad \begin{aligned} tr(H) &= D_{xx} + D_{yy} = \lambda_1 + \lambda_2 \\ \det(H) &= D_{xx} D_{yy} - D_{xy}^2 = \lambda_1 \lambda_2 \end{aligned}$$

- Let $r = \lambda_1 / \lambda_2$, then

$$\frac{tr(H)^2}{\det(H)} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} = \frac{(r\lambda_2 + \lambda_2)^2}{r\lambda_2^2} = \frac{(r+1)^2}{r}$$

- Reject candidates when *r*>10, i.e., $\frac{tr(H)^2}{\det(H)} > \frac{(10+1)^2}{10}$

$(r+1)^2 / r$ is a monotonic function for $r > 1$

# Second derivative filters

- $D_{xy}$ ?

$$\frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

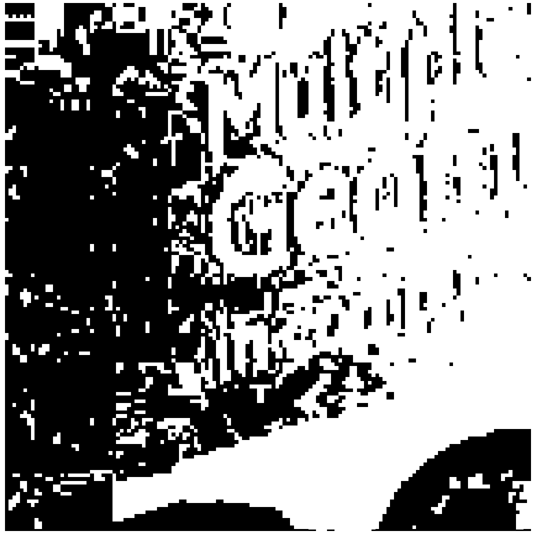
- $D_{xx}$ ?

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ -1 & 1 \\ 0 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 0 \\ -1 & 1 \\ 0 & 0 \end{bmatrix}$$

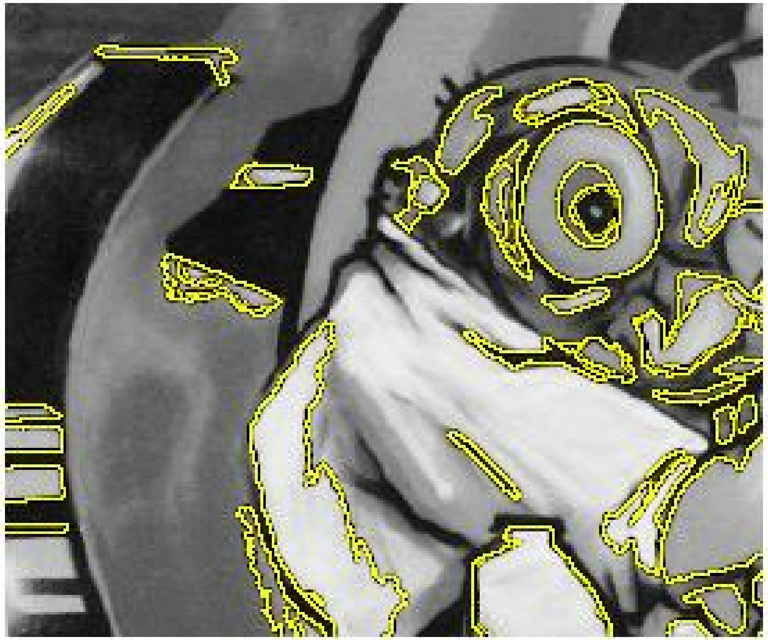
# SOME OTHER “KEYPOINT” EXTRACTORS

# Maximally Stable Extremal Regions [Matas '02]

- Based on Watershed segmentation algorithm
- Select regions that stay stable over a large parameter range

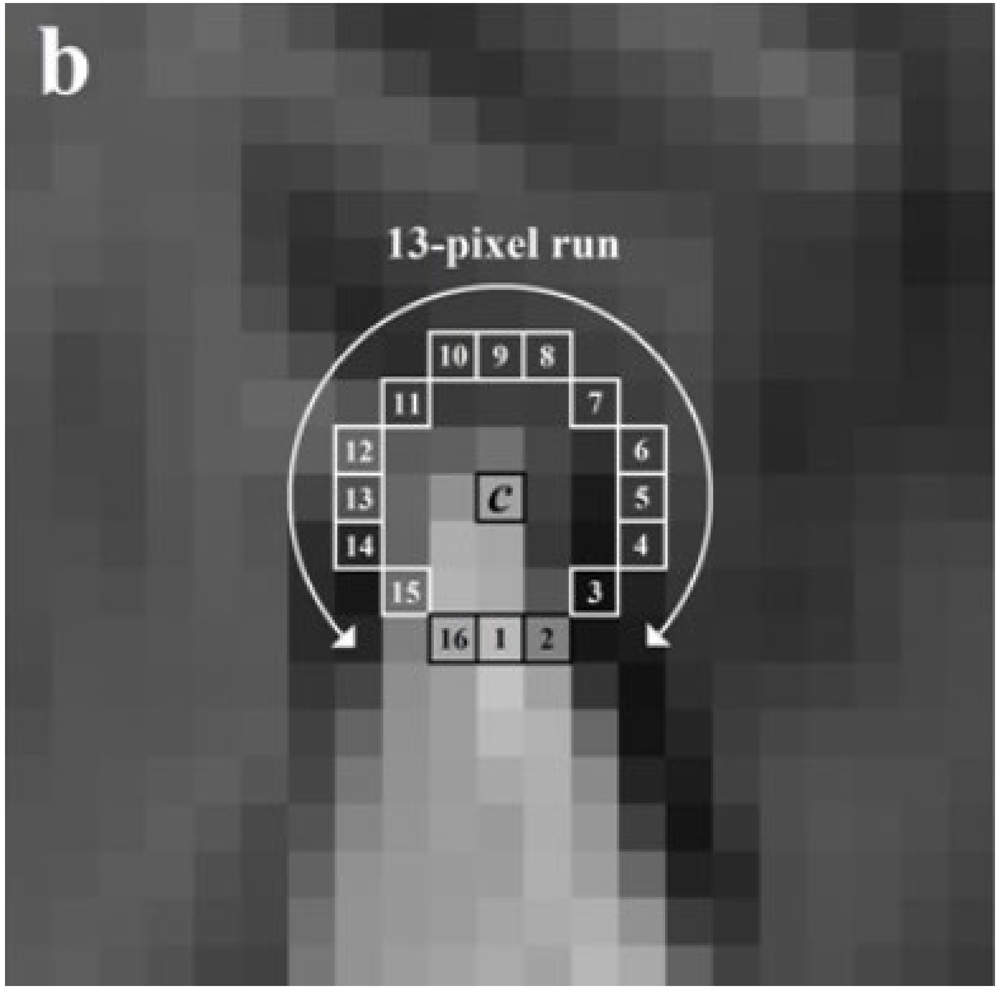
# Example Results: MSER




K. Grauman, B. Leibe

# Features from Accelerated Segment Test (FAST)

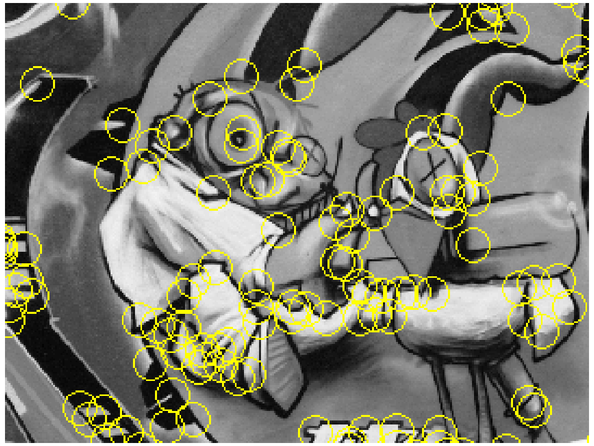
- Darker or lighter than target pixel for continuous 13-pixel run
- Can check only 1, 5, 9, 13 pixels first. Reject non-corner quickly
- Very fast
- Use in ORB



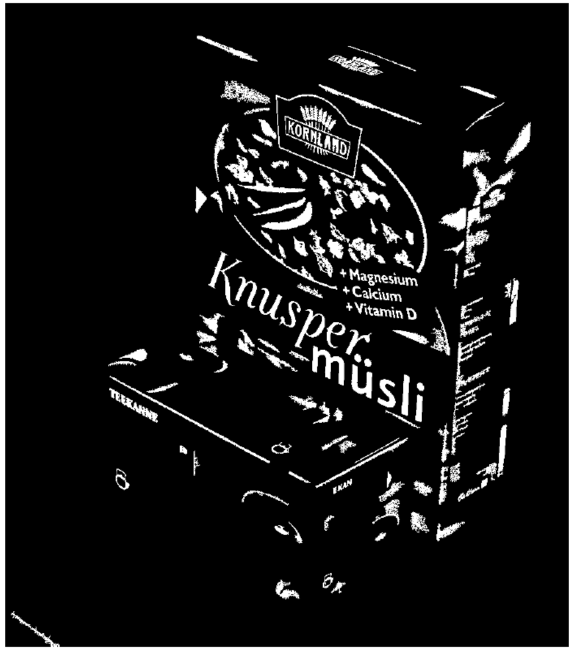

Slide credit: Allyson Auger

# Review: Interest points

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG, MSER, pixel difference



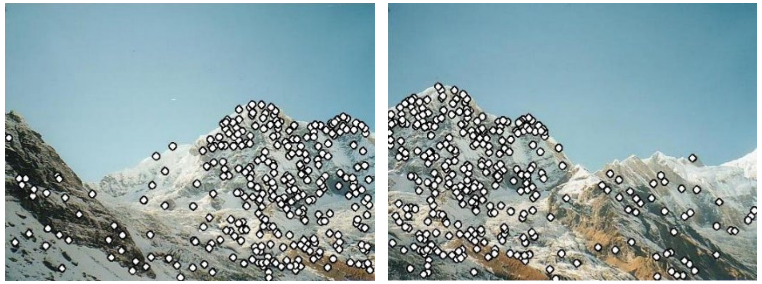

(a) Gray scale input image



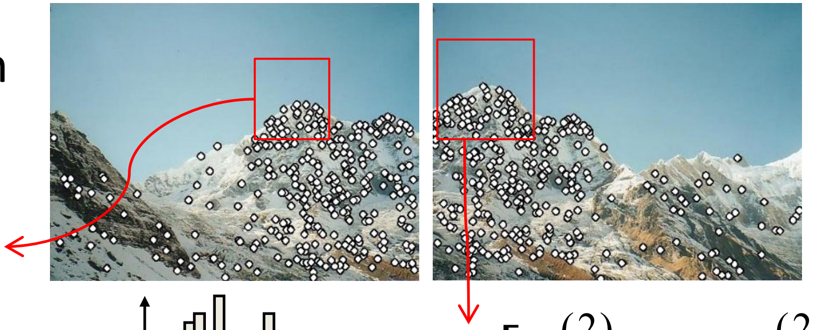

(b) Detected MSERs

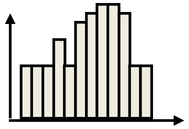
# Local features: main components

1) Detection:
   Find a set of distinctive key points.

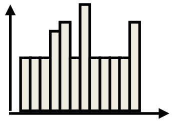


2) Description:
   Extract feature descriptor around each interest point as vector.



$x_1$ 

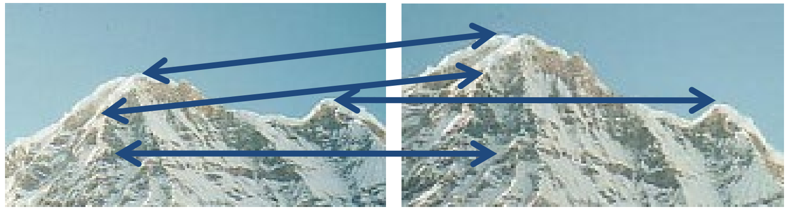
$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

$x_2$ 

$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) Matching:
   Compute distance between feature vectors to find correspondence.

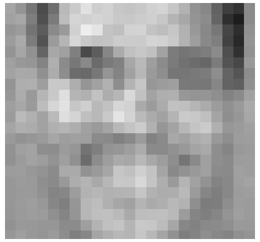
$$d(\mathrm{x}_1, \mathrm{x}_2) < T$$



K. Grauman, B. Leibe

# Image representations

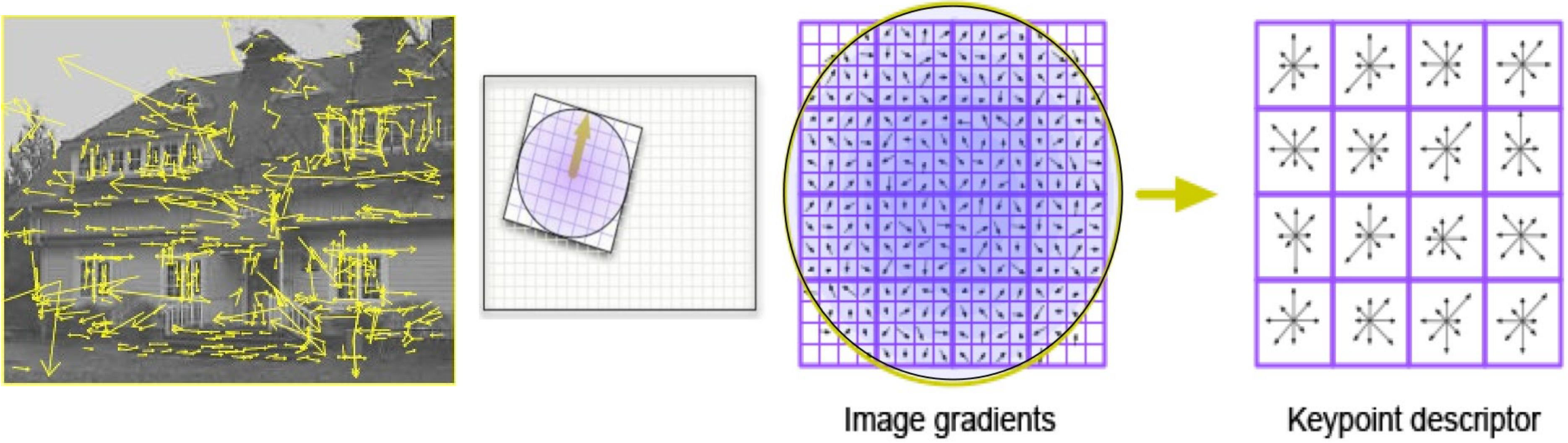
- Templates
  - Intensity, gradients, etc.



- Histograms
  - Color, texture, SIFT descriptors, etc.

# For what things do we compute histograms?

- Texture
- Local histograms of oriented gradients
- SIFT: Scale Invariant Feature Transform




SIFT – Lowe IJCV 2004

James Hays

# SIFT

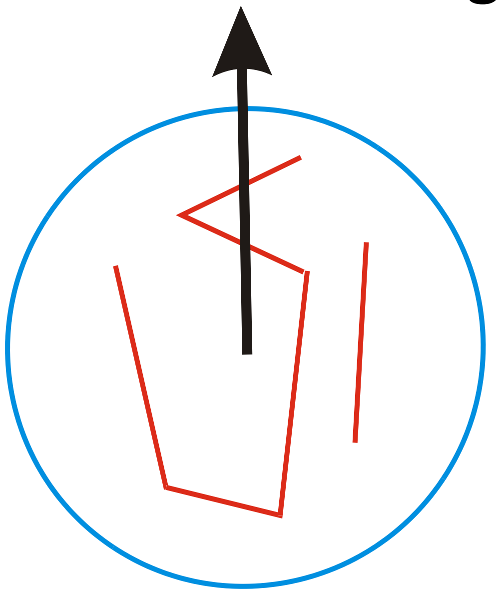
- Find Difference of Gaussian scale-space extrema
- Post-processing
  - Position interpolation
  - Discard low-contrast points
  - Eliminate points along edges

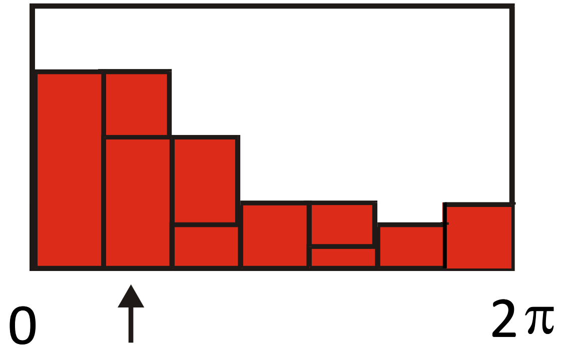
# SIFT

- Find Difference of Gaussian scale-space extrema
- Post-processing
  - Position interpolation
  - Discard low-contrast points
  - Eliminate points along edges
- Orientation estimation

# SIFT Orientation Normalization

- Compute orientation histogram
- Select dominant orientation $\Theta$
- Normalize: rotate to fixed orientation
  - In practice, use a local reference frame aligned with the orientation before computing orientation histogram





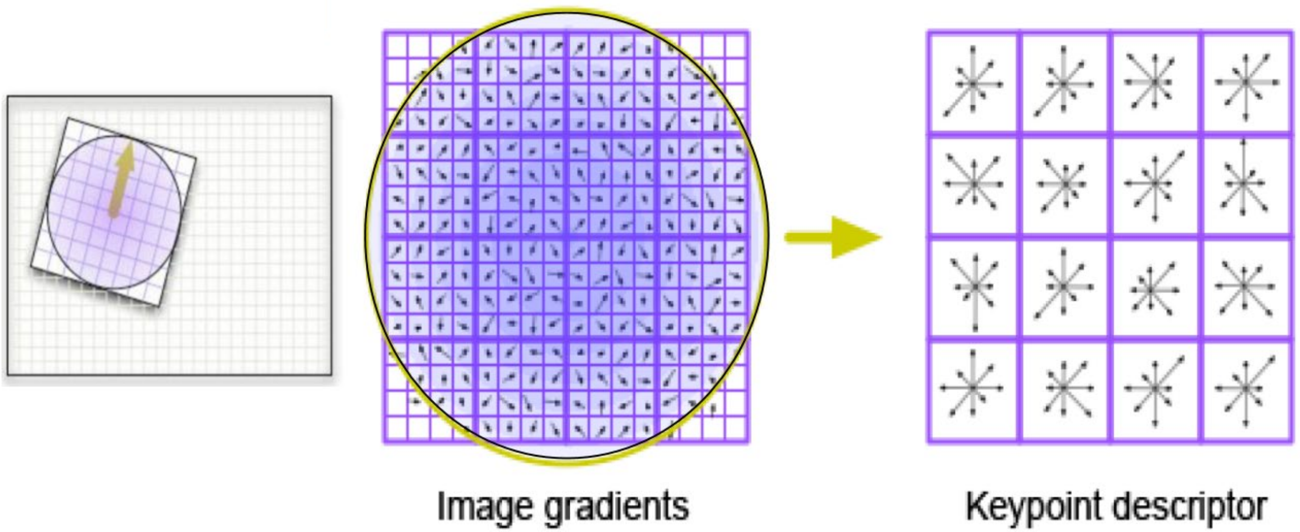

[Lowe, SIFT, 1999]

# SIFT

- Find Difference of Gaussian scale-space extrema
- Post-processing
  - Position interpolation
  - Discard low-contrast points
  - Eliminate points along edges
- Orientation estimation
- Descriptor extraction
  - Motivation: We want some sensitivity to spatial layout, but not too much, so blocks of histograms give us that.

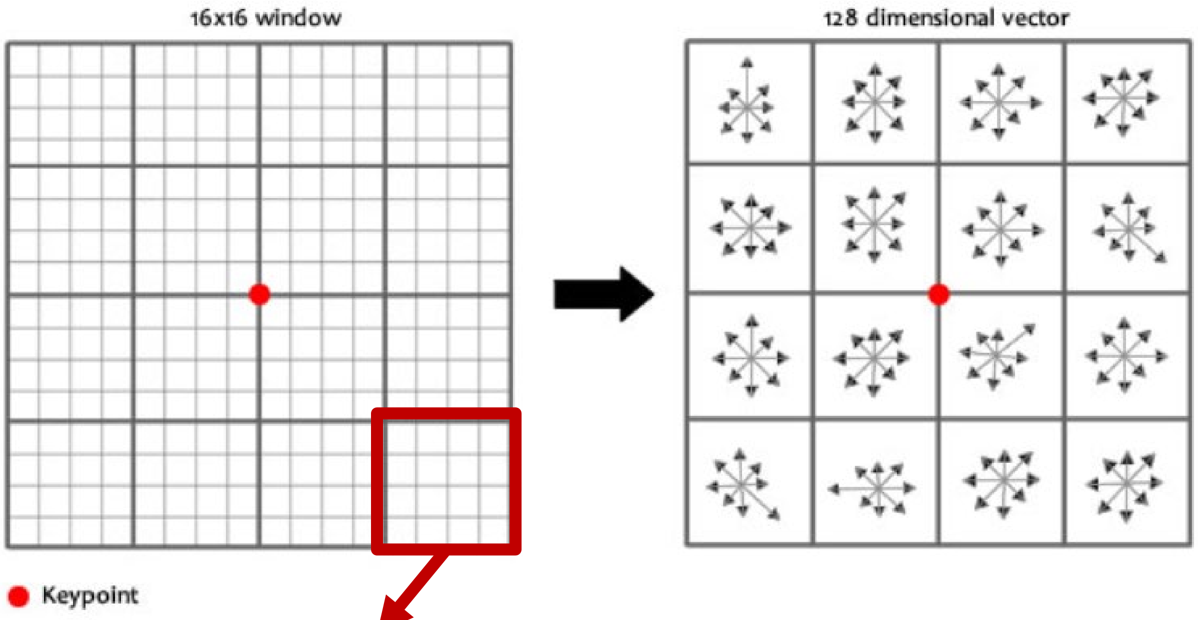
# SIFT Descriptor Extraction

- Given a keypoint with scale and orientation:
  - Pick scale-space image which most closely matches estimated scale
  - Resample image to match orientation OR
  - Normalize orientation by shifting histogram.

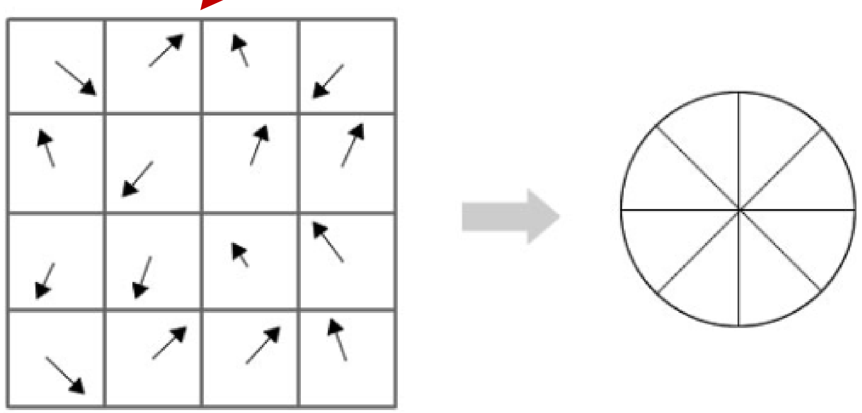


# SIFT Descriptor Extraction

- Given a keypoint with scale and orientation




Gradient magnitude and orientation



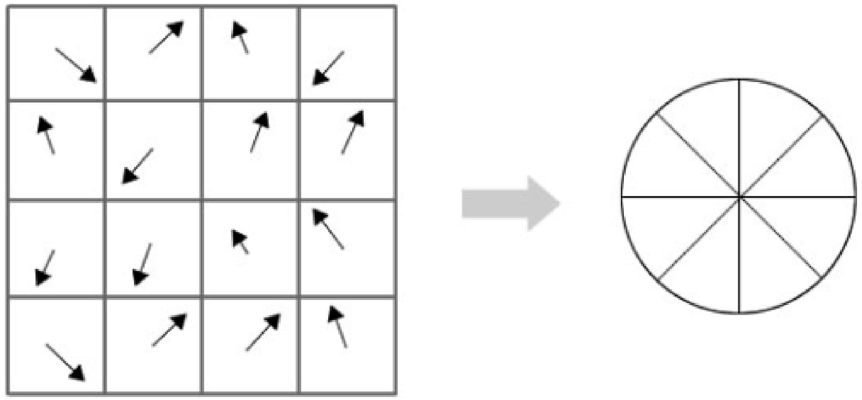
8 bin 'histogram' - add magnitude amounts!

Utkarsh Sinha

# SIFT Descriptor Extraction

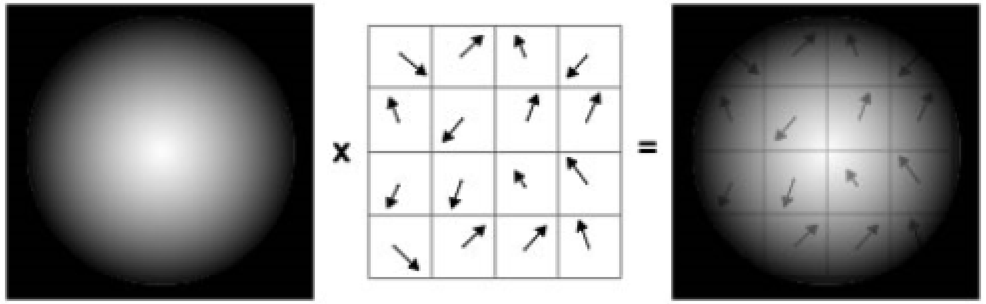
- Within each 4x4 window

Gradient magnitude and orientation



8 bin 'histogram' - add magnitude amounts!

Weight magnitude that is added to 'histogram' by Gaussian

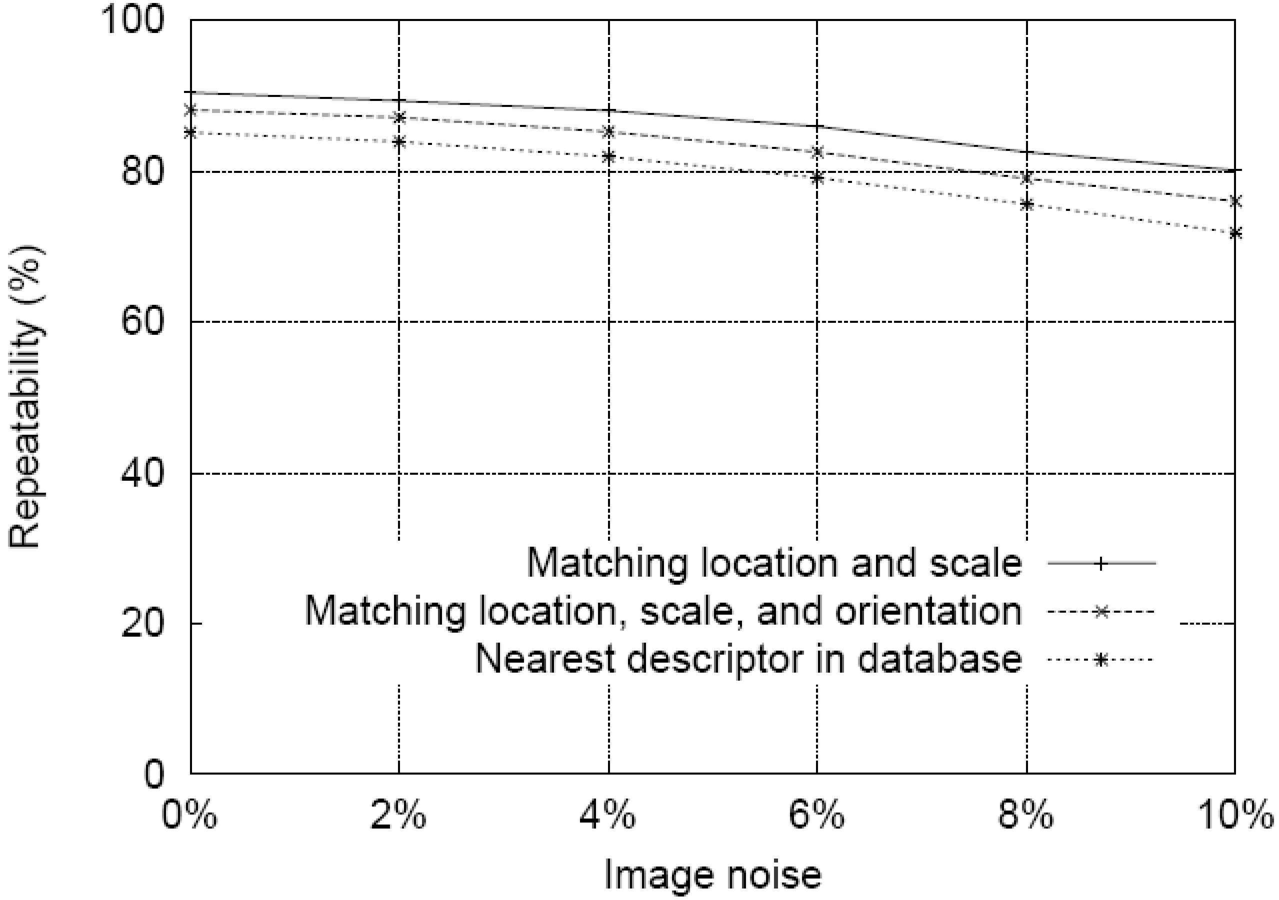


# SIFT Descriptor Extraction

- 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 $I = \alpha I$ brightness changes
  - Clamp all vector values > 0.2 to 0.2.
    - Robust to “non-linear illumination effects”
    - Image value saturation / specular highlights
  - Renormalize

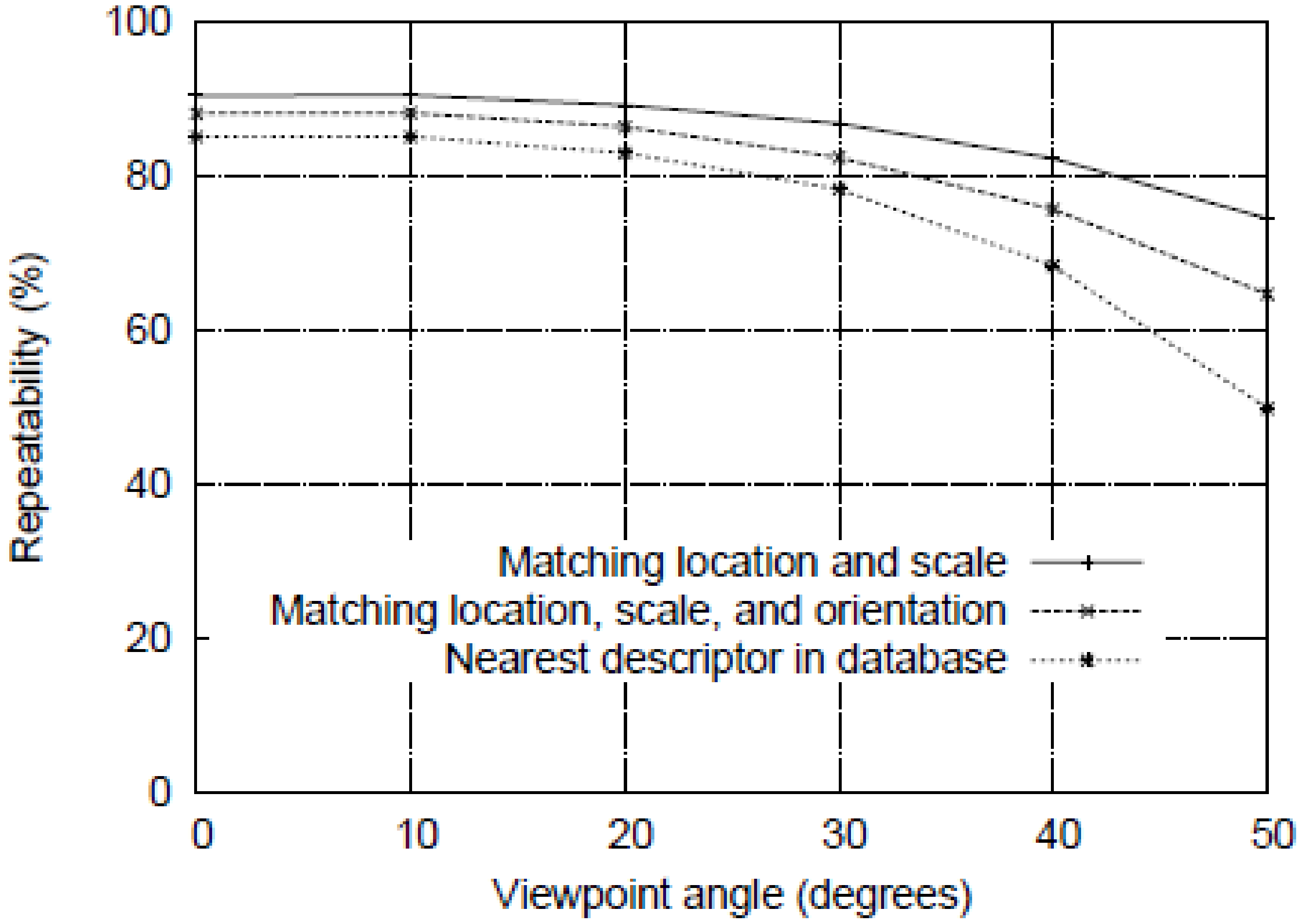
# HOW GOOD IS SIFT?

# SIFT Repeatability



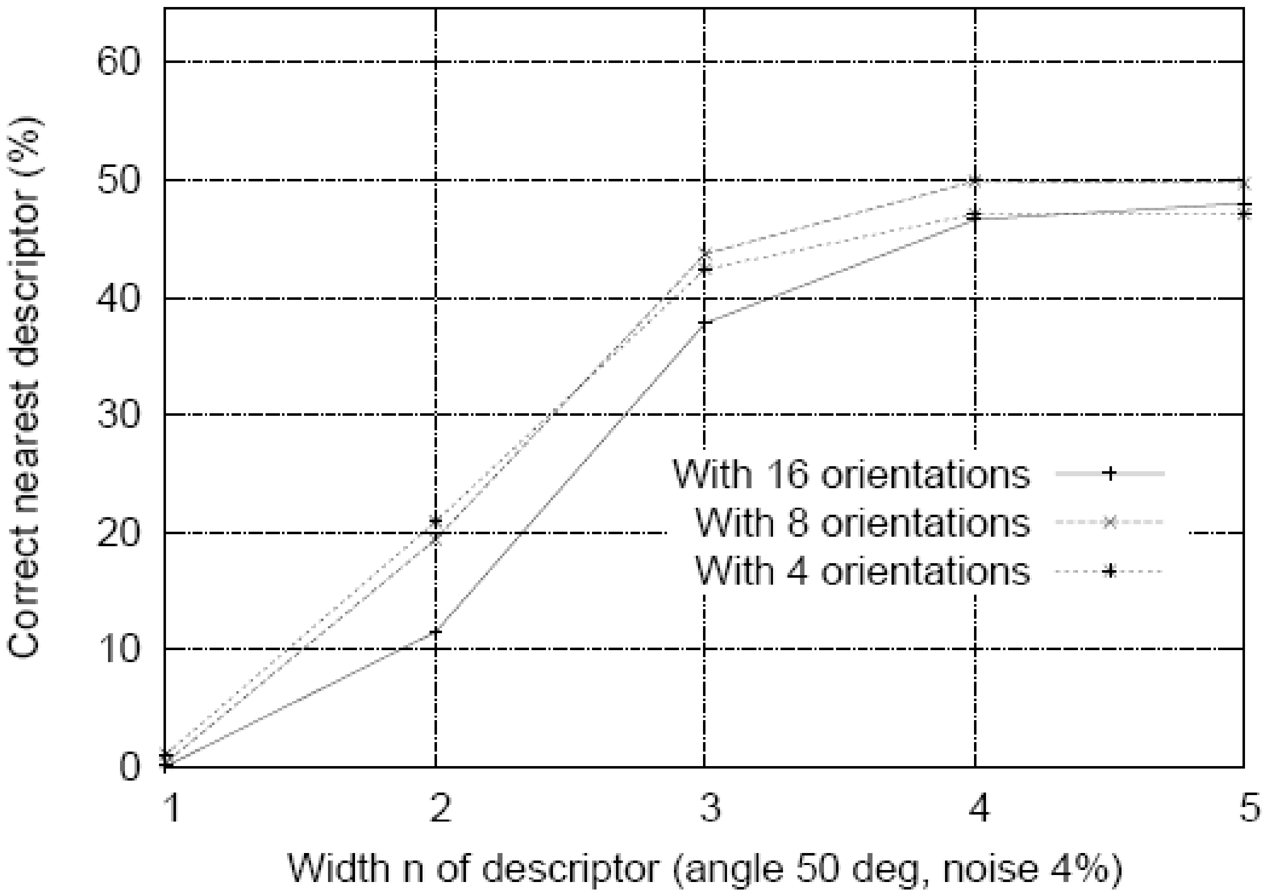

Lowe IJCV 2004

# SIFT Repeatability



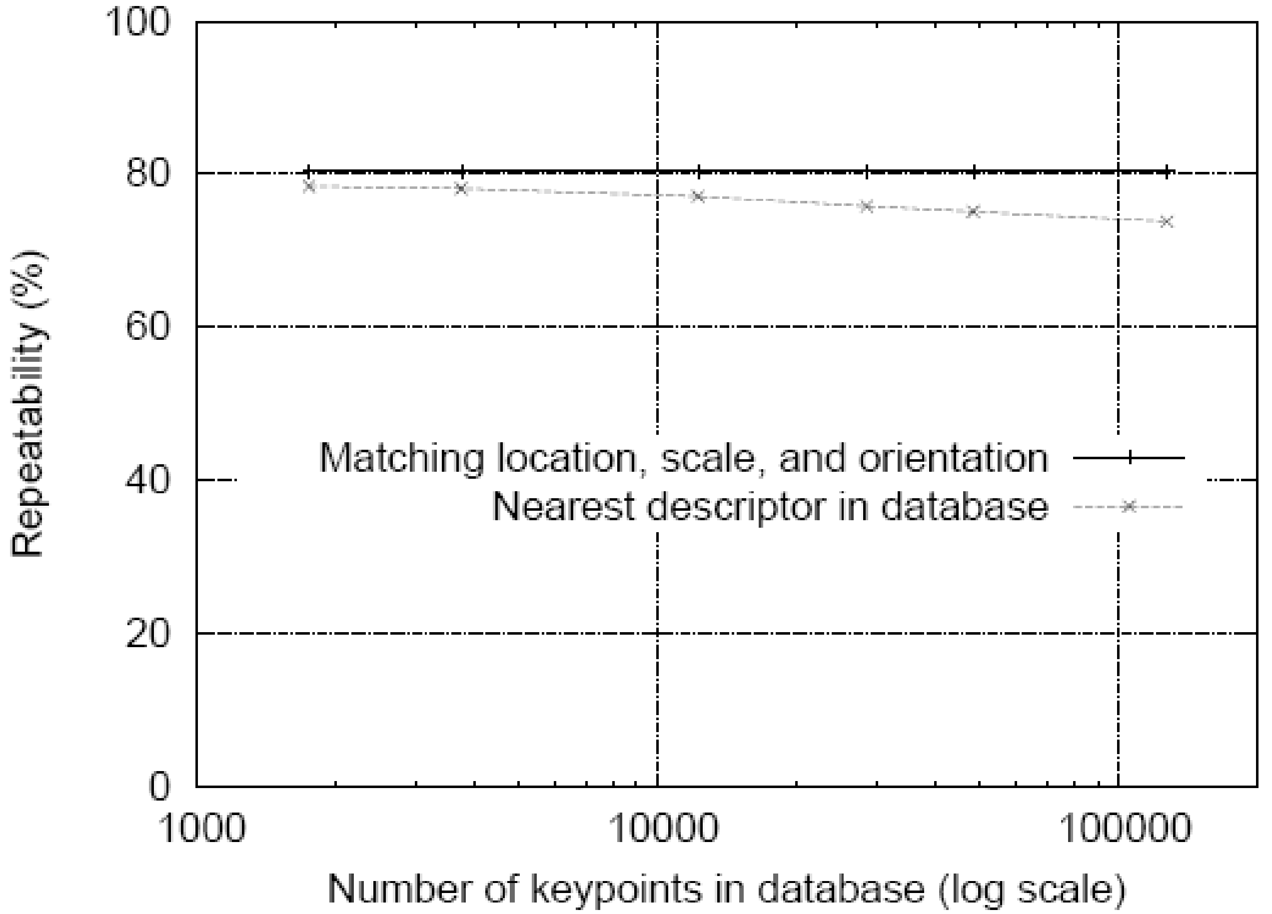

Lowe IJCV 2004

# SIFT Repeatability



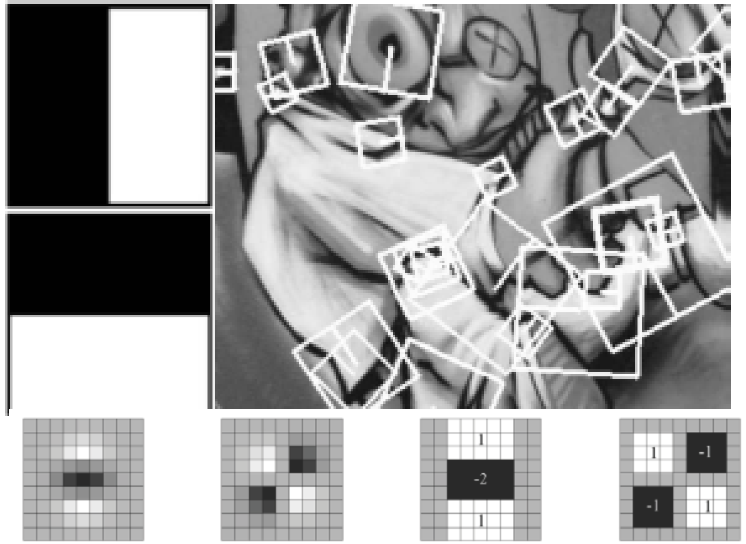

Lowe IJCV 2004

# SIFT Repeatability




Lowe IJCV 2004

# Local Descriptors: SURF




**Fast approximation of SIFT idea**

- **Efficient computation by 2D box filters & integral images**
  **⇒ 6 times faster than SIFT**
- **Equivalent quality for object identification**

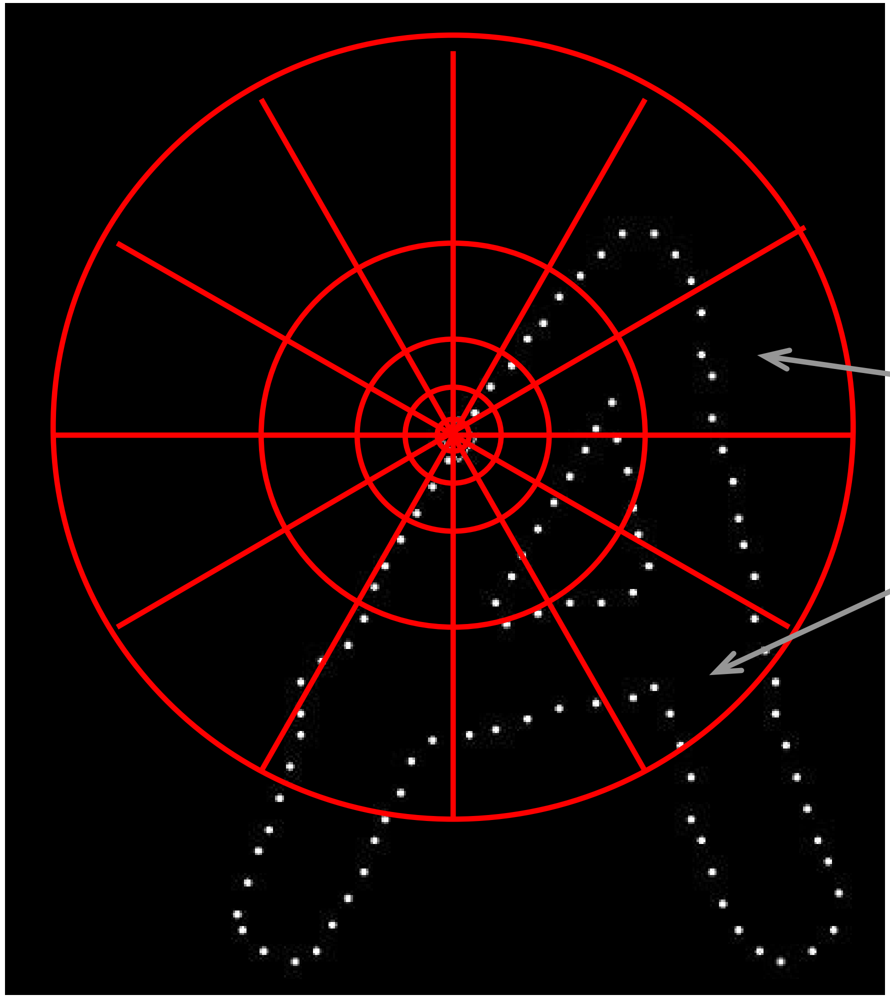
**GPU implementation available**

- **Feature extraction @ 200Hz (detector + descriptor, 640×480 img)**
- http://www.vision.ee.ethz.ch/~surf

[Bay, ECCV'06], [Cornelis, CVGPU'08]

K. Grauman, B. Leibe

# Local Descriptors: Shape Context



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

**Count = 4**

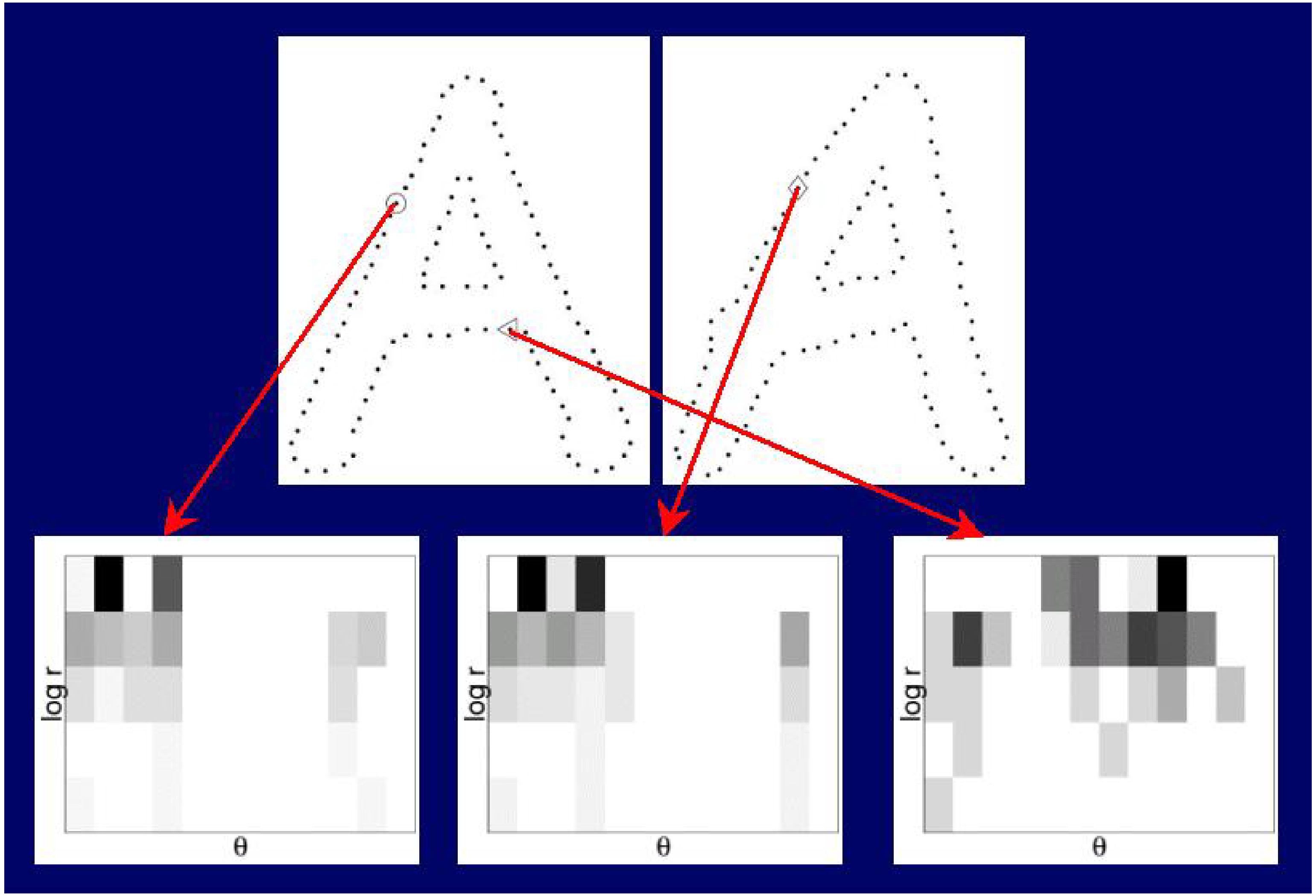
**⋮**

**Count = 10**

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

Belongie & Malik, ICCV 2001

# Shape Context Descriptor

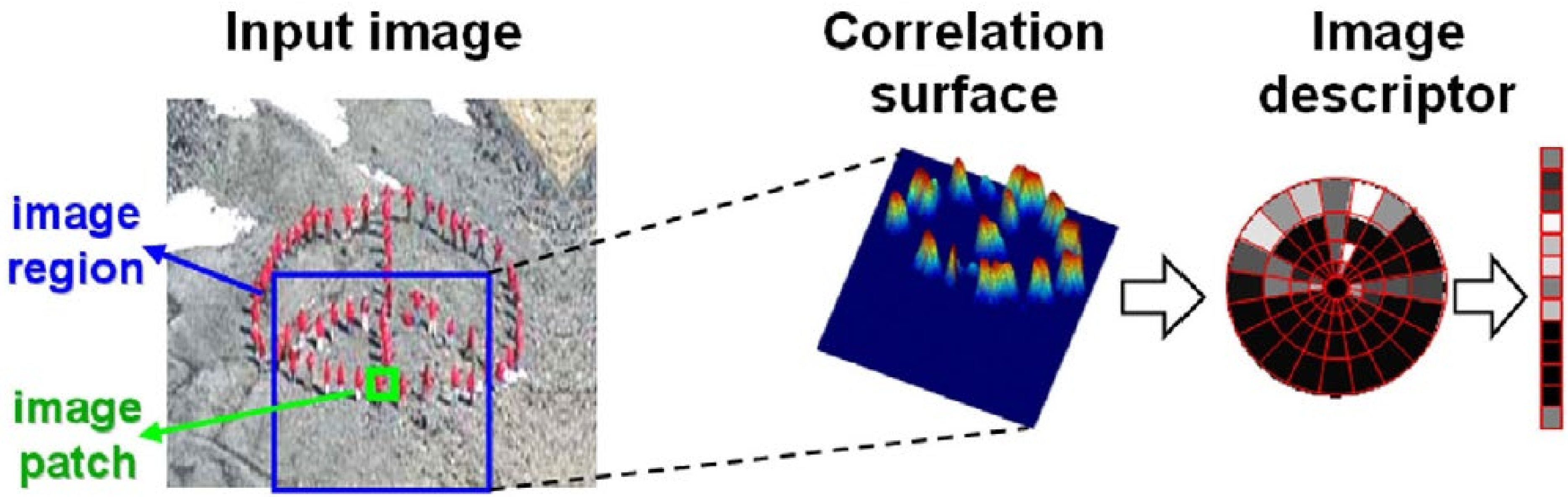


# Self-similarity Descriptor

Figure 1. *These images of the same object (a heart) do NOT share common image properties (colors, textures, edges), but DO share a similar geometric layout of local internal self-similarities.*

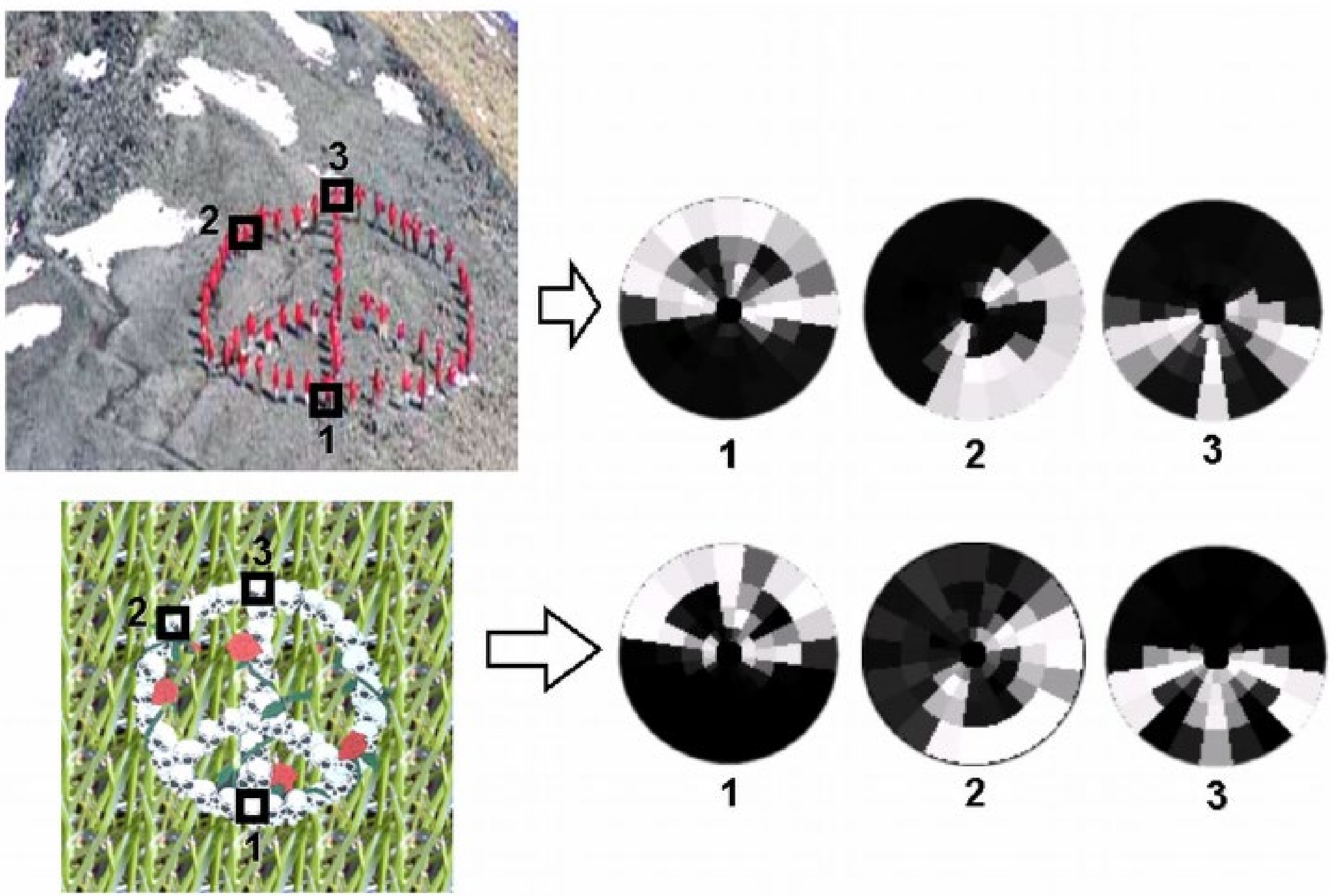
Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

# Self-similarity Descriptor




Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

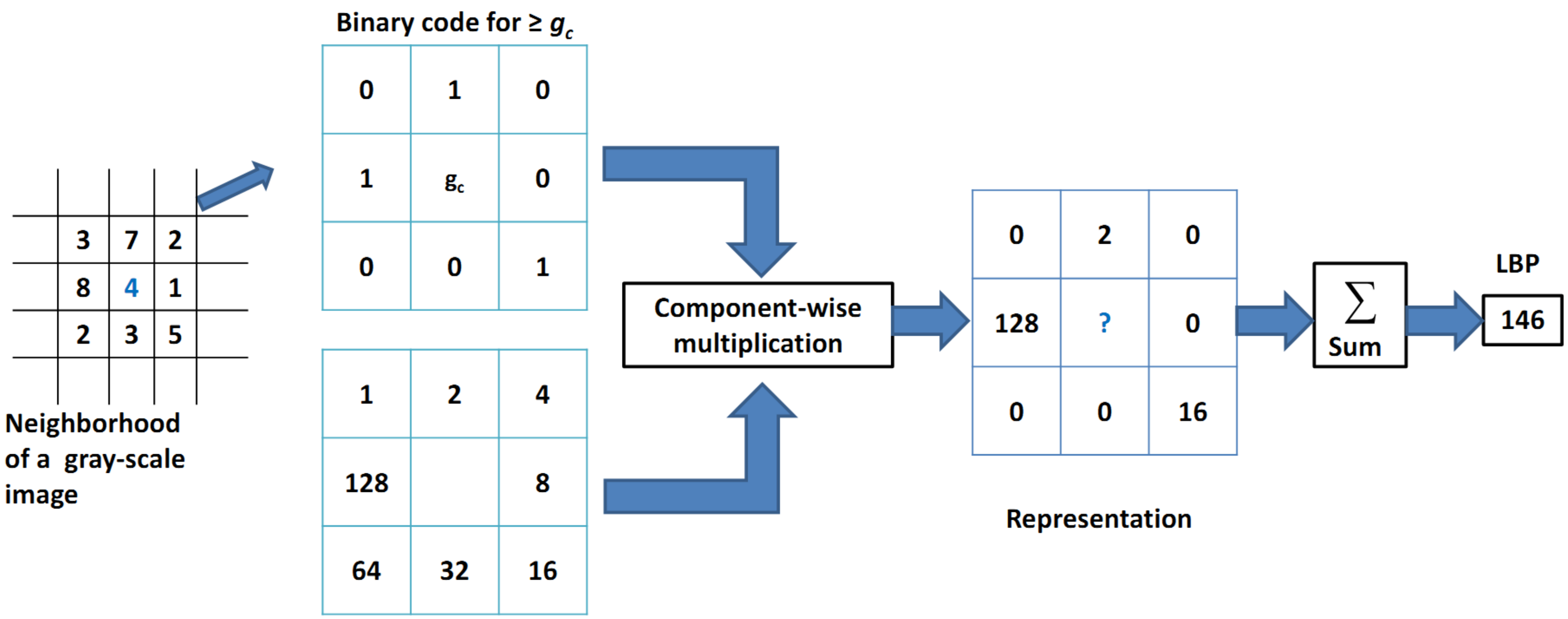
# Self-similarity Descriptor



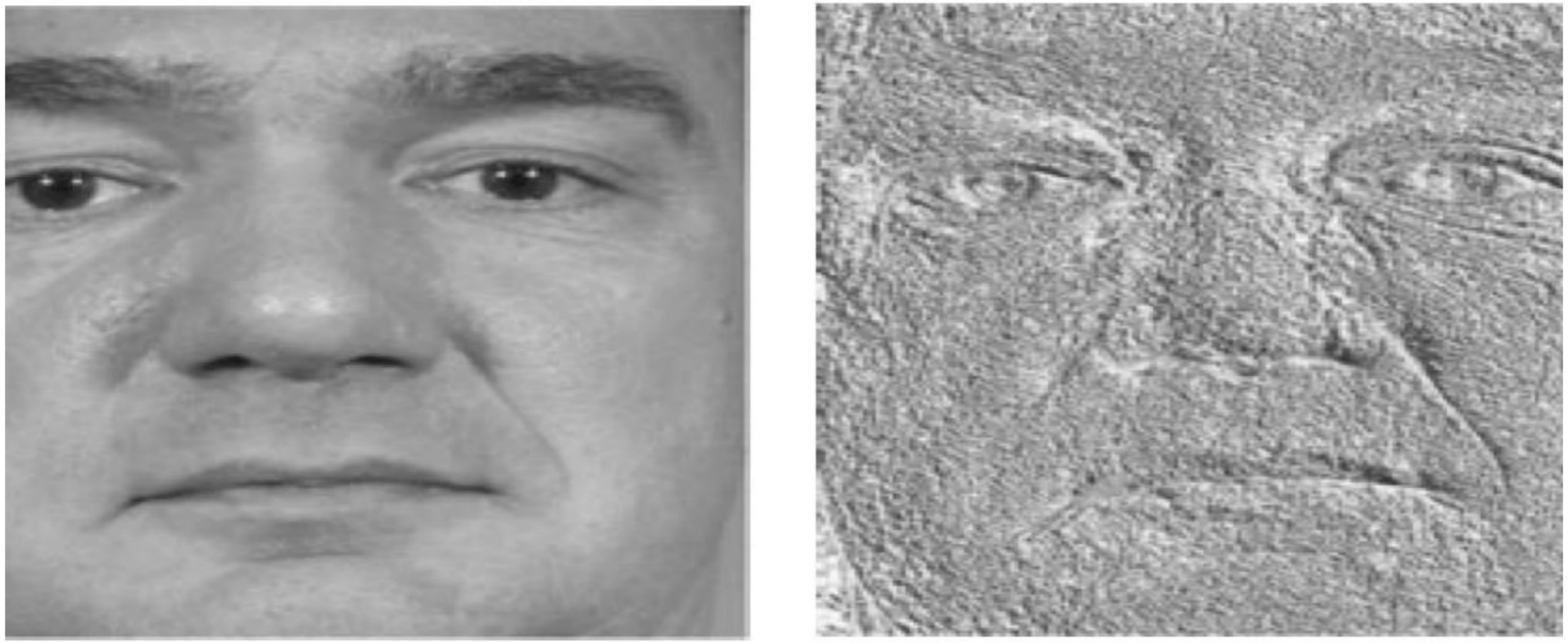

Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

# Local binary pattern (LBP)

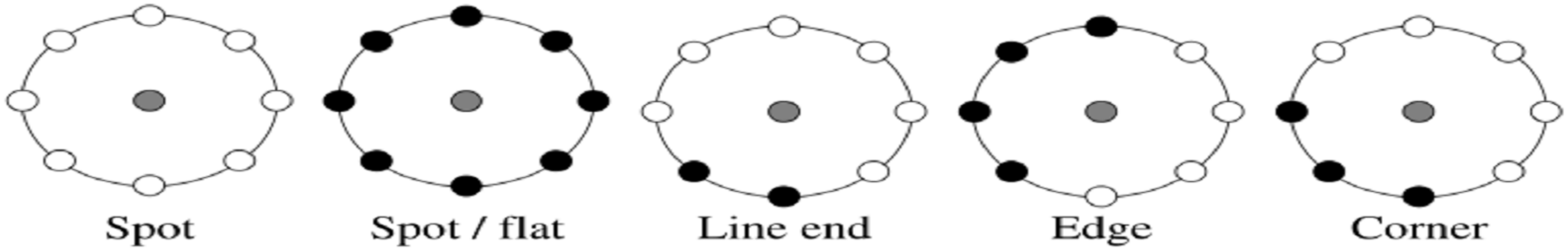
- Introduced by Ojala *et al.* in 1996
- Popular in late 2000



# LBP

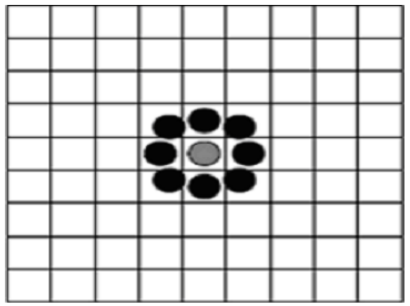


# Different detectable textures by LBP

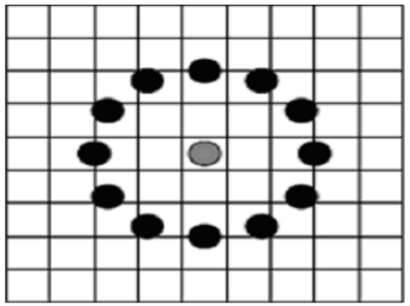


# "Advanced" LBP(P,R)

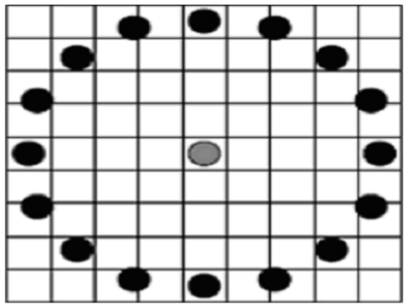
P = Pixels
R = Radius



LBP(8,1)



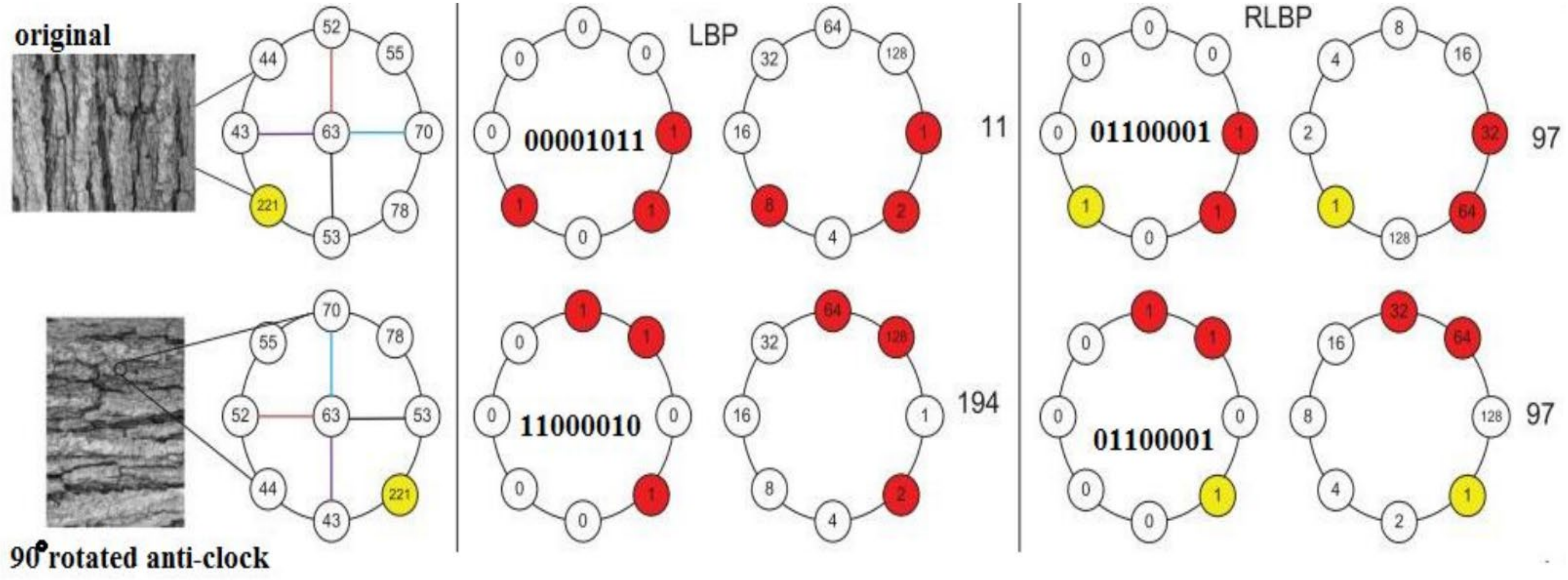
LBP(16,2)



LBP(20,4)

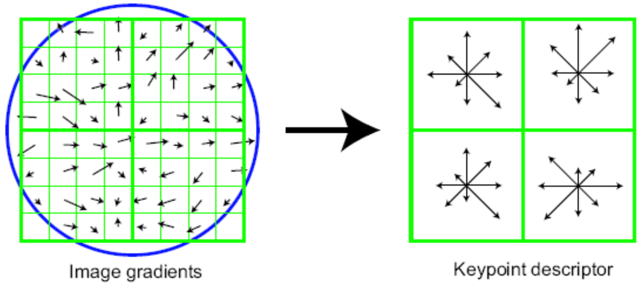
# Rotated LBP (RLBP)

- LBP is not rotational invariance by default
- But can easily modified it to be so



# 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



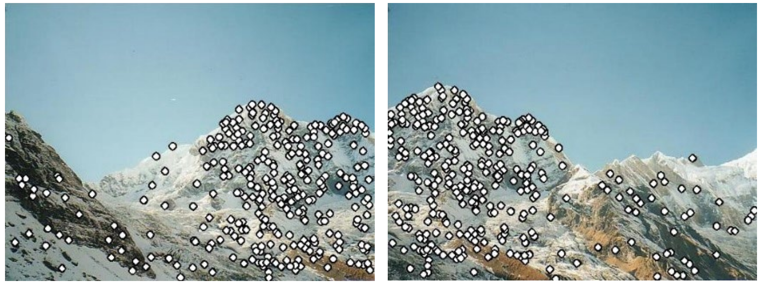

- Most available descriptors focus on edge/gradient information
  - Capture texture information
  - Color rarely used

# Binary Robust Independent Elementary Features (BRIEF)

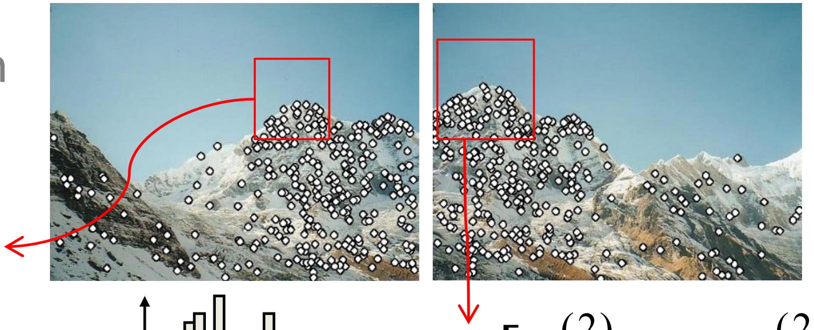
- Very similar to LBP but the pattern is more arbitrary
- Random pattern is usually used
  - Choose 256 pairs from 35x35 pixel area
  - Input is first smooth with a 9x9 Gaussian filter with $\sigma = 7$
- Resulting in 256 bit string (32 bytes)
- Usually better in pattern matching than LBP, LBP is better in texture analysis
- Use in ORB

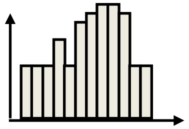
# Local features: main components

1) **Detection:**
Find a set of distinctive key points.

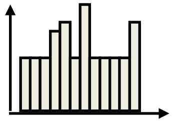


2) **Description:**
Extract feature descriptor around each interest point as vector.



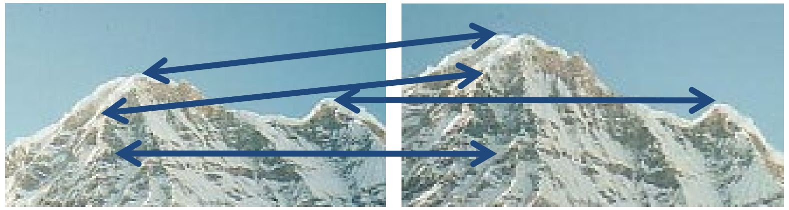
$x_1$ 

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

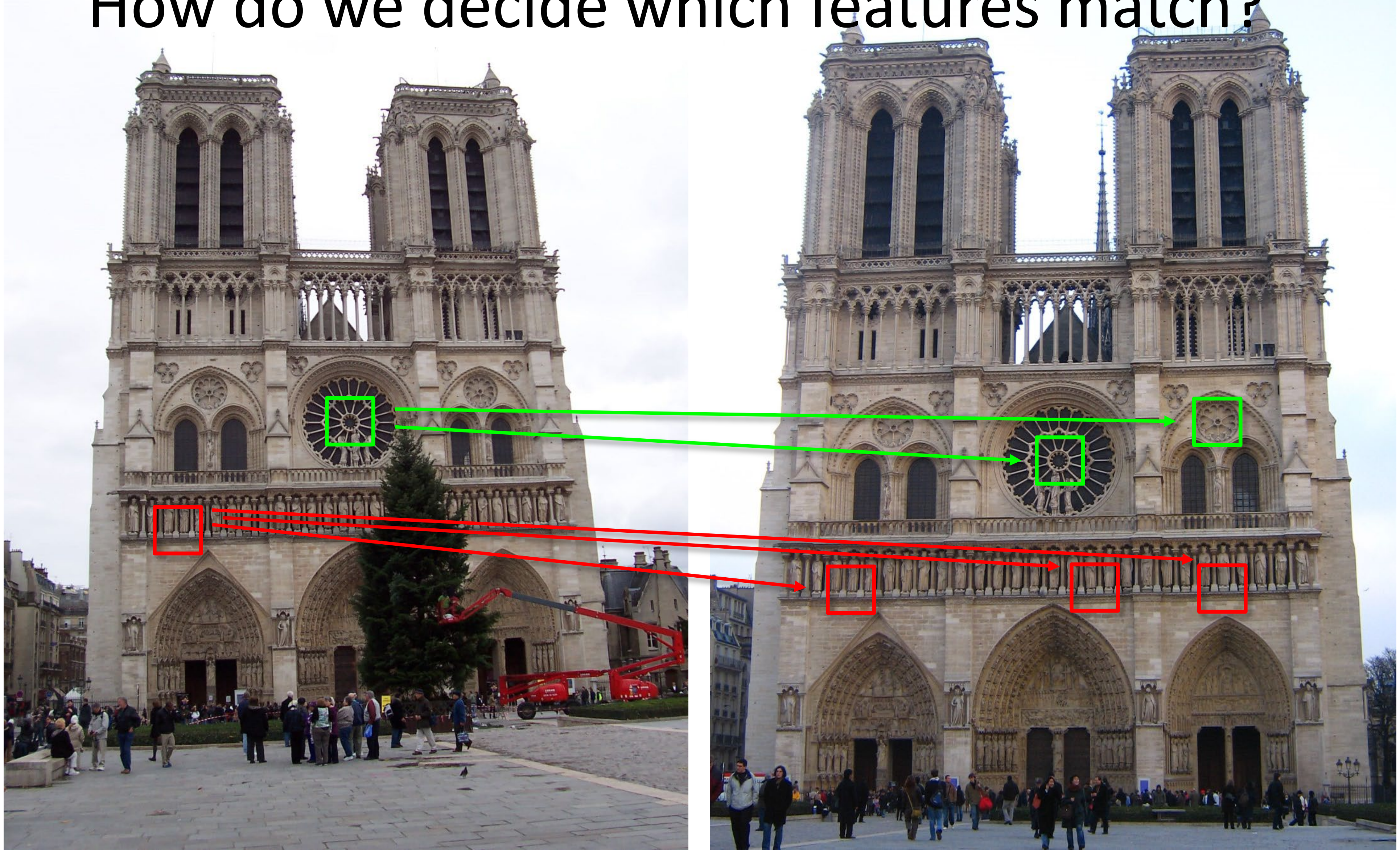
$x_2$ 

$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) **Matching:**
Compute distance between feature vectors to find correspondence.



# How do we decide which features match?



Distance: 0.34, 0.30, 0.40

Distance: 0.61, 1.22

# Matching for SIFT-like features

- Euclidean distance:

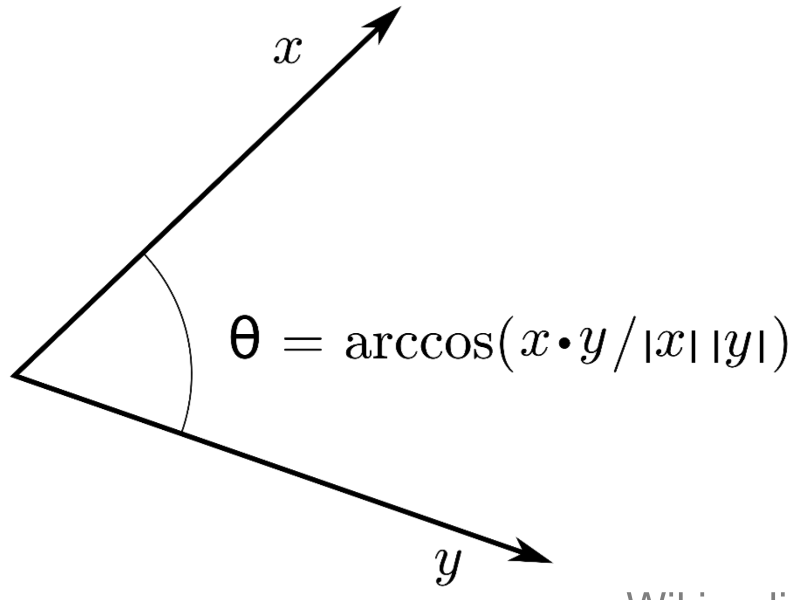
$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

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

- 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.*

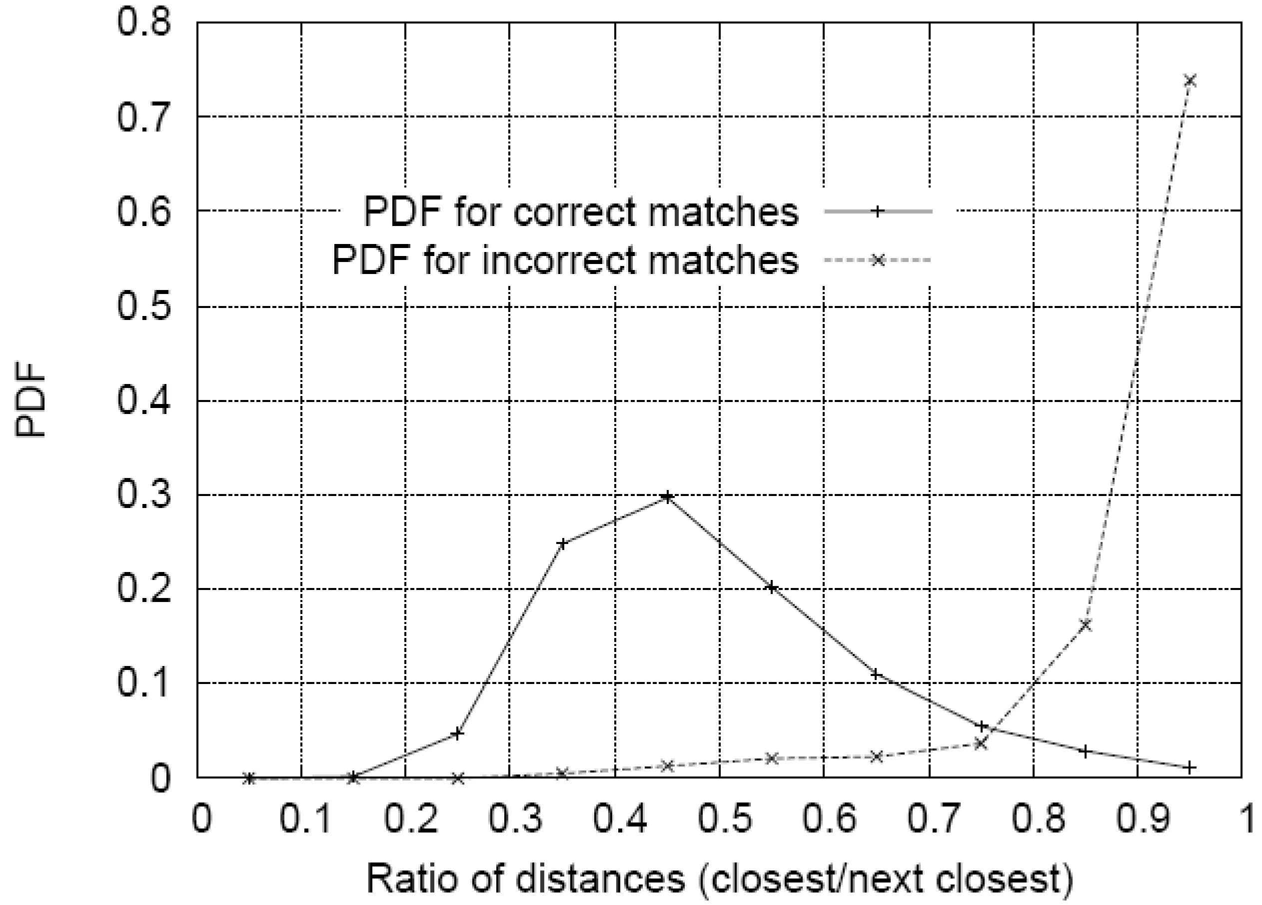
- If NN1 ≈ NN2, ratio $\frac{NN1}{NN2}$ will be ≈ 1 -> matches too close.
- As NN1 << NN2, ratio $\frac{NN1}{NN2}$ tends to 0.

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

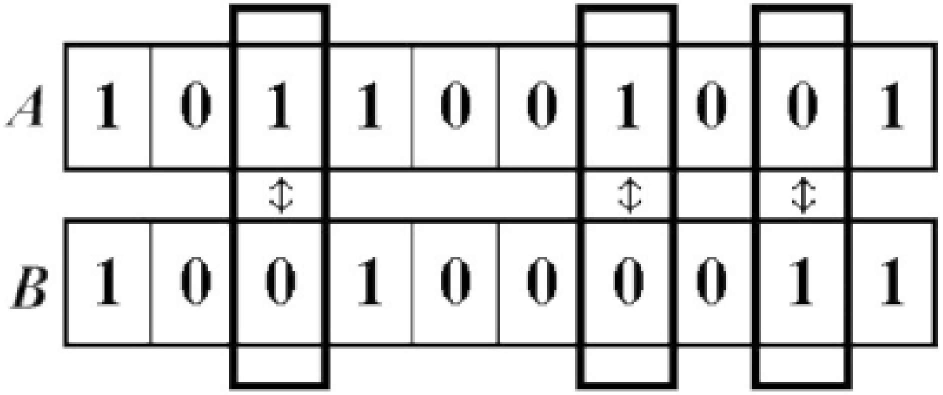
- Operate on matrices of descriptors
- E.g., for row vectors,

```
features_image1 * features_image2^T
```

produces matrix of dot product results
for all pairs of features

# Matching for binary feature

- We focus on SIFT-like (floating point) features earlier
- For binary features such as BRIEF, Hamming distance is more reasonable (i.e., counting number of bit differences)
- What is the Hamming distance between A and B below?



# Summary

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG, pixel difference
- Descriptors: robust and selective
  - Spatial histograms of orientation
  - SIFT, LBP, BRIEF
- Matching:
  - SIFT-like: Euclidean, cosine similarity (usually better)
  - LBP-like (binary): Hamming distance