Computer Vision Course
Lecture 07

Local Image Descriptors

Ceyhun Burak Akgül, PhD
cba-research.com

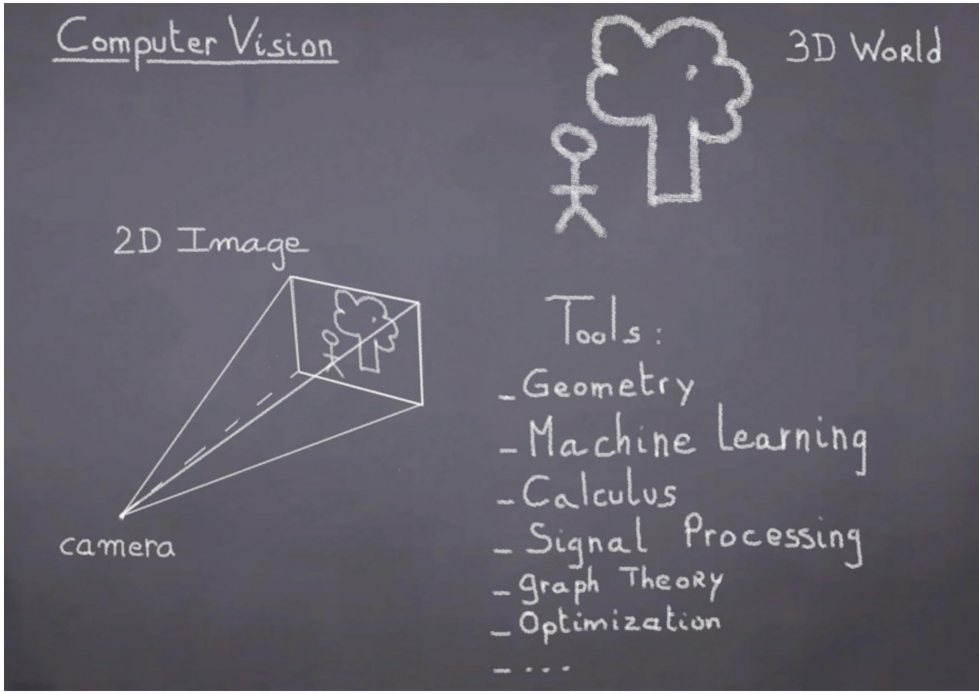
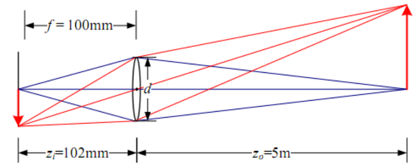


Photo credit: Olivier Teboul
vision.mas.ecp.fr/Personnel/teboul

Spring 2015
Last updated 07/04/2015

These slides have been adapted from James Hays's 2014 Computer Vision course slides at Brown University.

Course Outline



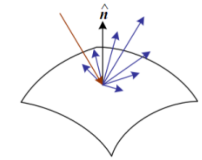
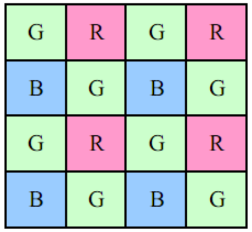


Image Formation and Processing

Light, Shape and Color

The Pin-hole Camera Model, The Digital Camera

Linear filtering, Template Matching, Image Pyramids

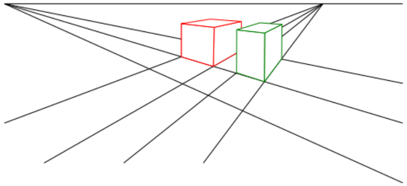


Feature Detection and Matching

Edge Detection, Interest Points: Corners and Blobs

Local Image Descriptors

Feature Matching and Hough Transform



Multiple Views and Motion

Geometric Transformations, Camera Calibration

Feature Tracking , Stereo Vision

Segmentation and Grouping

Segmentation by Clustering, Region Merging and Growing

Advanced Methods Overview: Active Contours, Level-Sets, Graph-Theoretic Methods

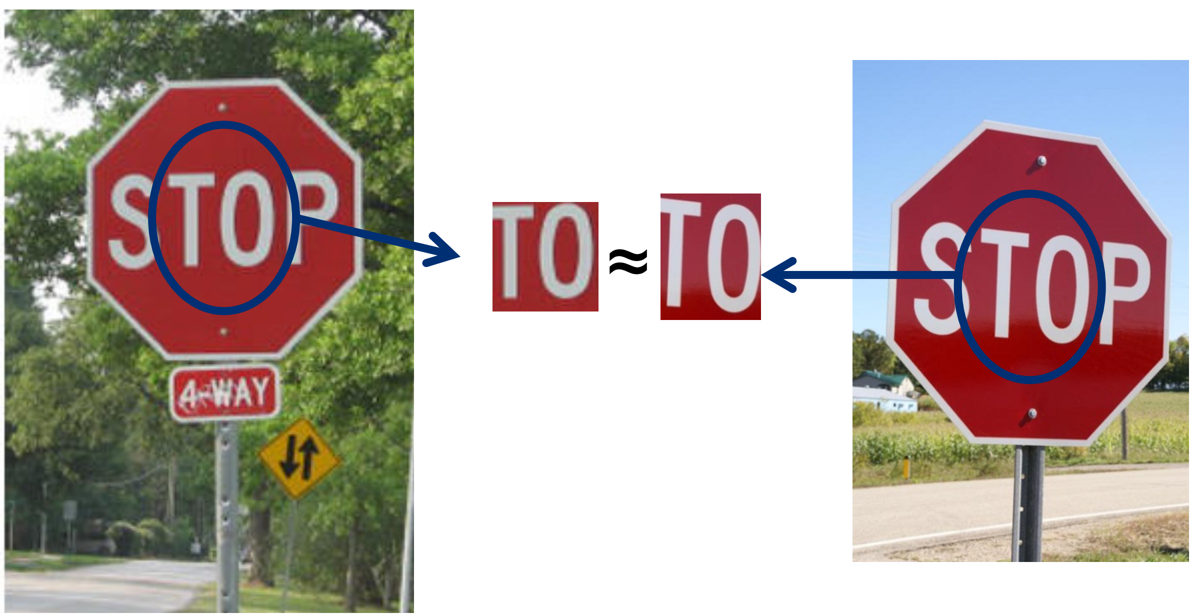
Detection and Recognition

Problems and Architectures Overview

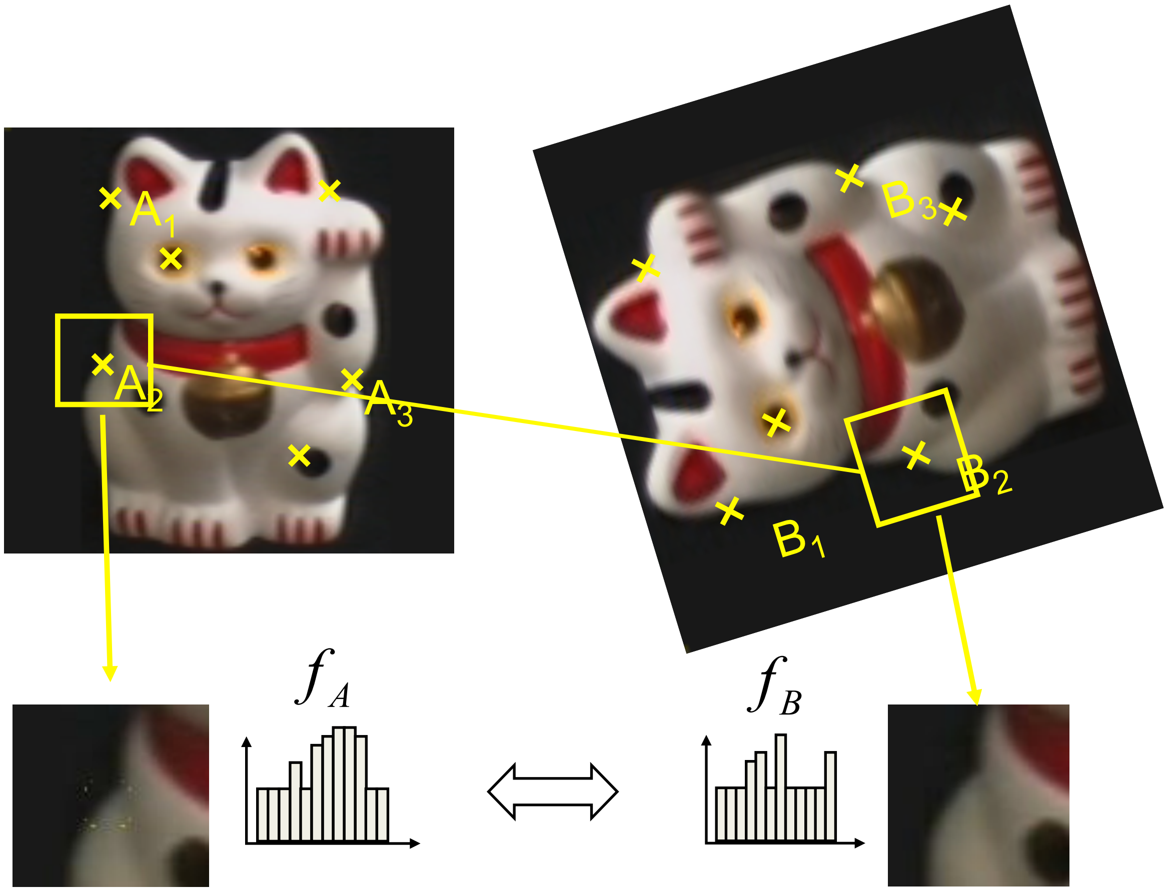
Statistical Classifiers, Bag-of-Words Model, Detection by Sliding Windows

Correspondence and Alignment

- Correspondence: matching points, patches, edges, or regions across images



Overview of Keypoint Matching

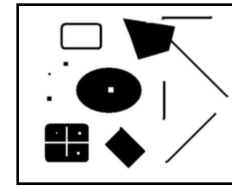


$$d(f_A, f_B) < T$$

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**

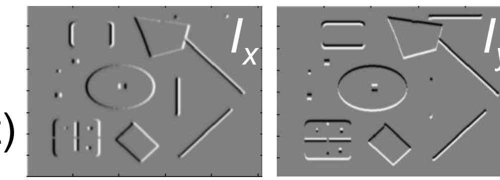
Harris Detector

- Second moment matrix



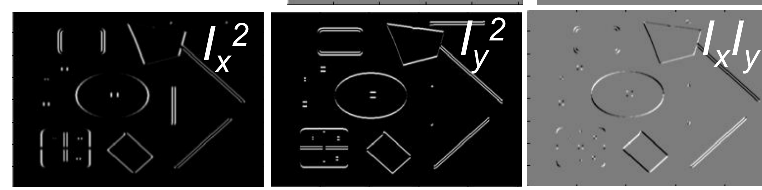
$$\mu(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

1. Image derivatives (optionally, blur first)



$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

2. Square of derivatives



3. Gaussian filter $g(\sigma_I)$

4. Cornerness function – both eigenvalues are strong

$$har = \det[\mu(\sigma_I,\sigma_D)] - \alpha[\operatorname{trace}(\mu(\sigma_I,\sigma_D))^2] =$$
$$g(I_x^2)g(I_y^2) - [g(I_xI_y)]^2 - \alpha[g(I_x^2)+g(I_y^2)]^2$$

5. Non-maxima suppression

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

Automatic Scale Selection

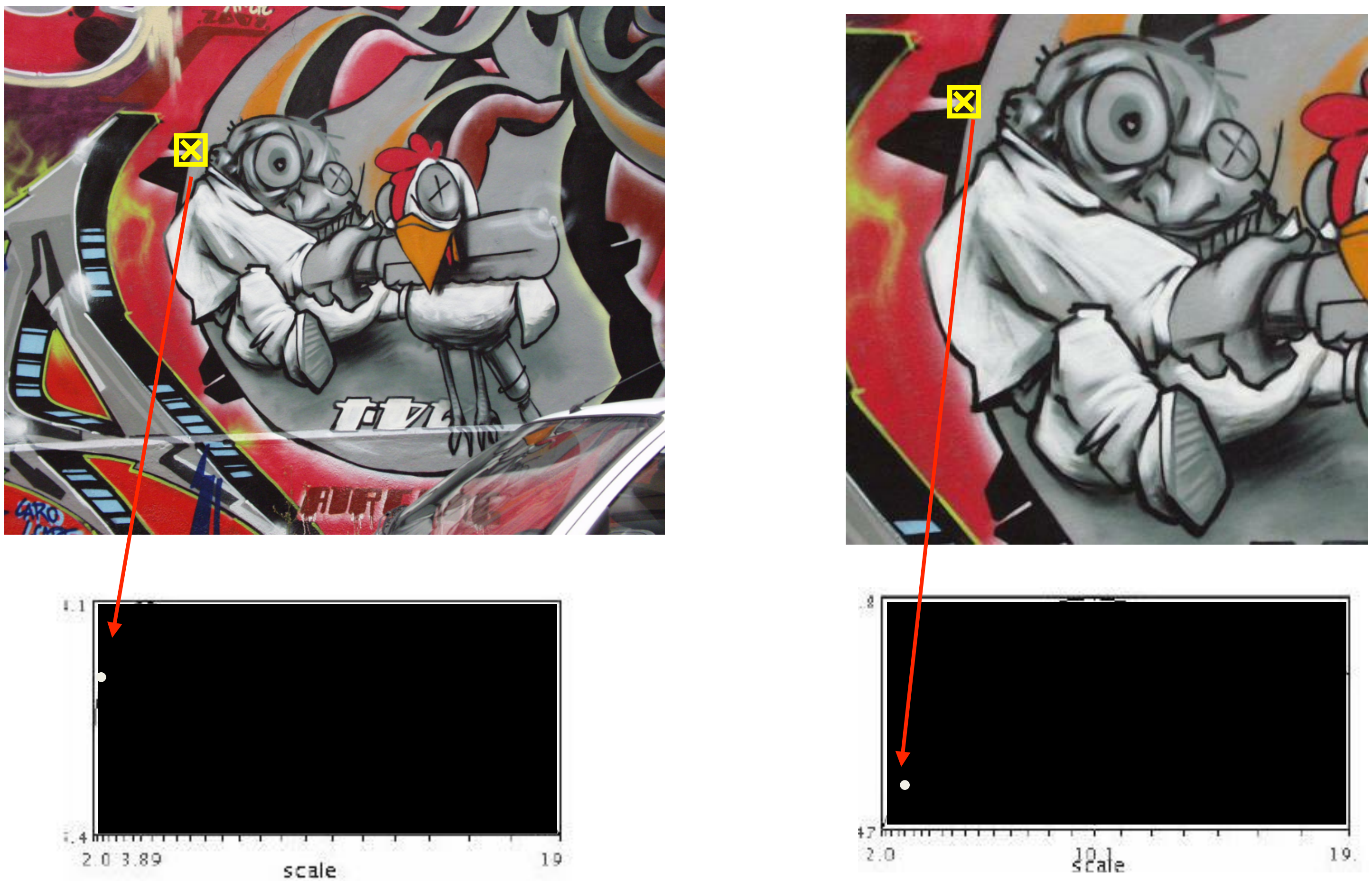
$$f(I_{i_1 \ldots i_m}(x,\sigma)) = f(I_{i_1 \ldots i_m}(x',\sigma'))$$

How to find corresponding patch sizes?

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

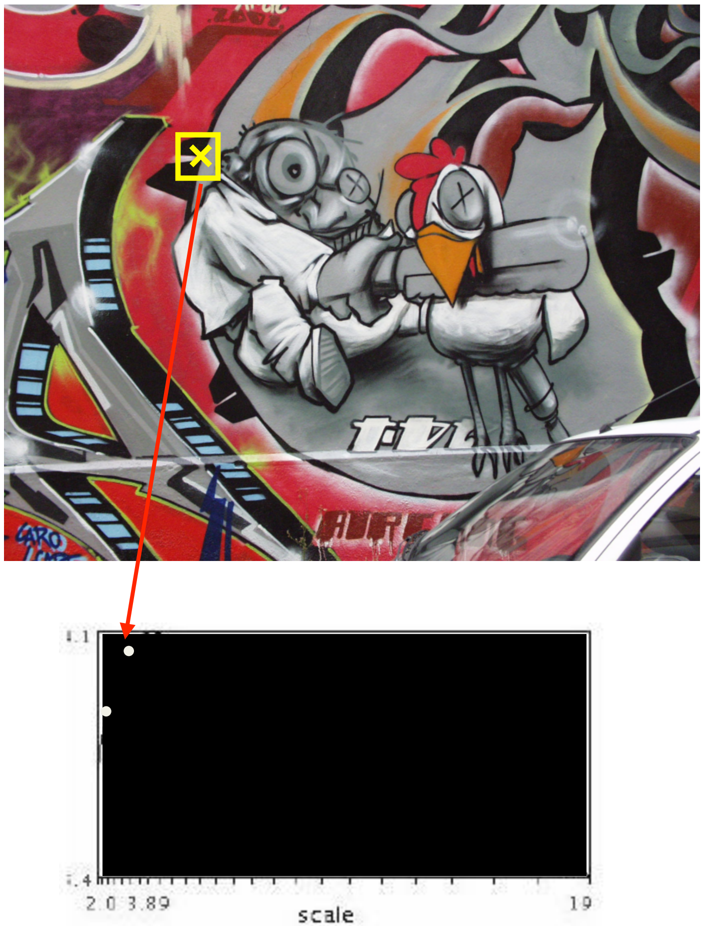


$f(I_{i_1 \ldots i_m}(x,\sigma))$ $f(I_{i_1 \ldots i_m}(x',\sigma))$

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



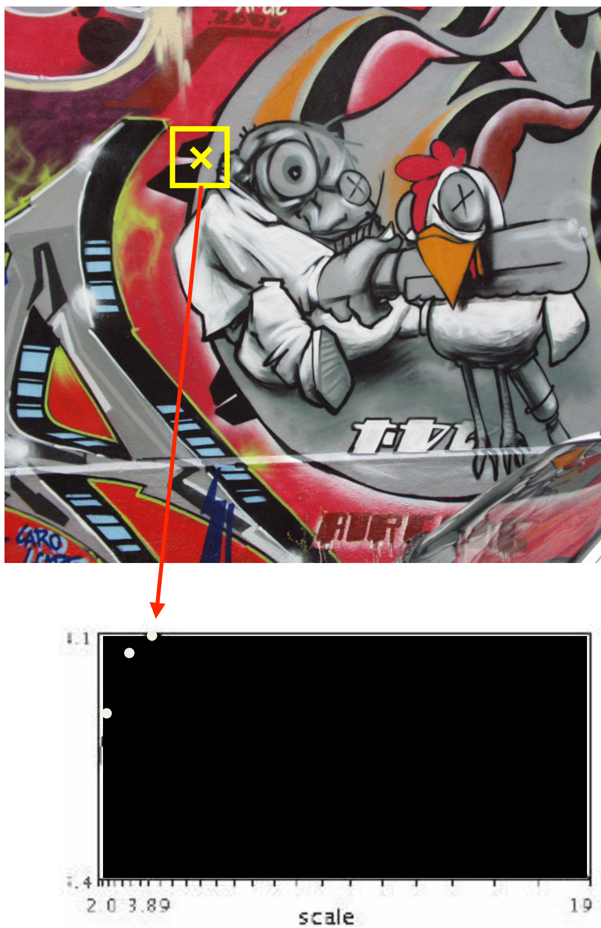
$$f(I_{i_1 \ldots i_m}(x, \sigma))$$

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

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



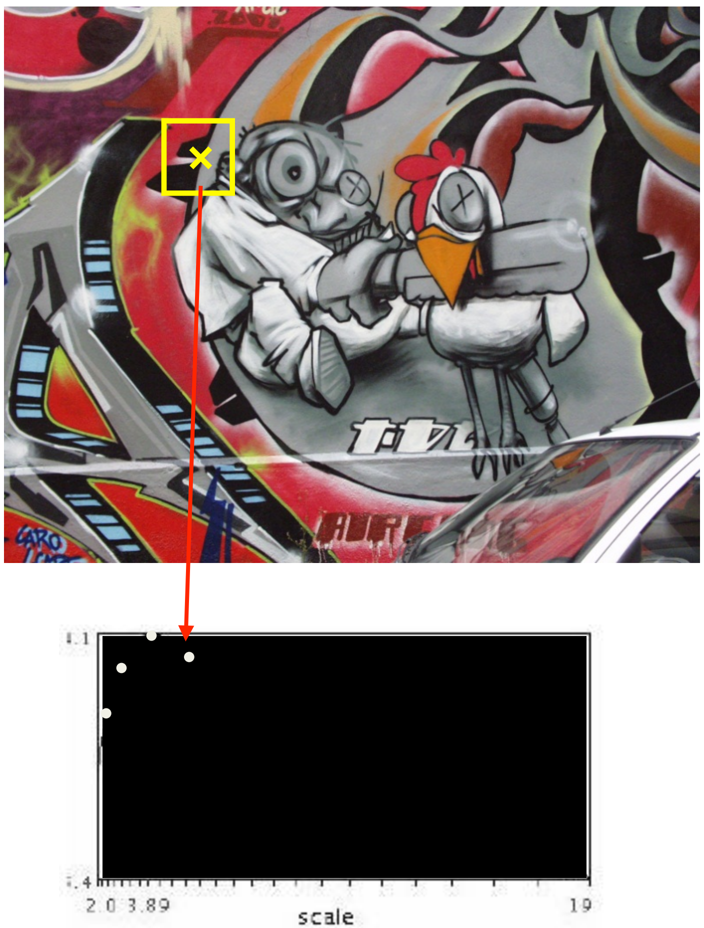
$$f(I_{i_1\ldots i_m}(x,\sigma))$$

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

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



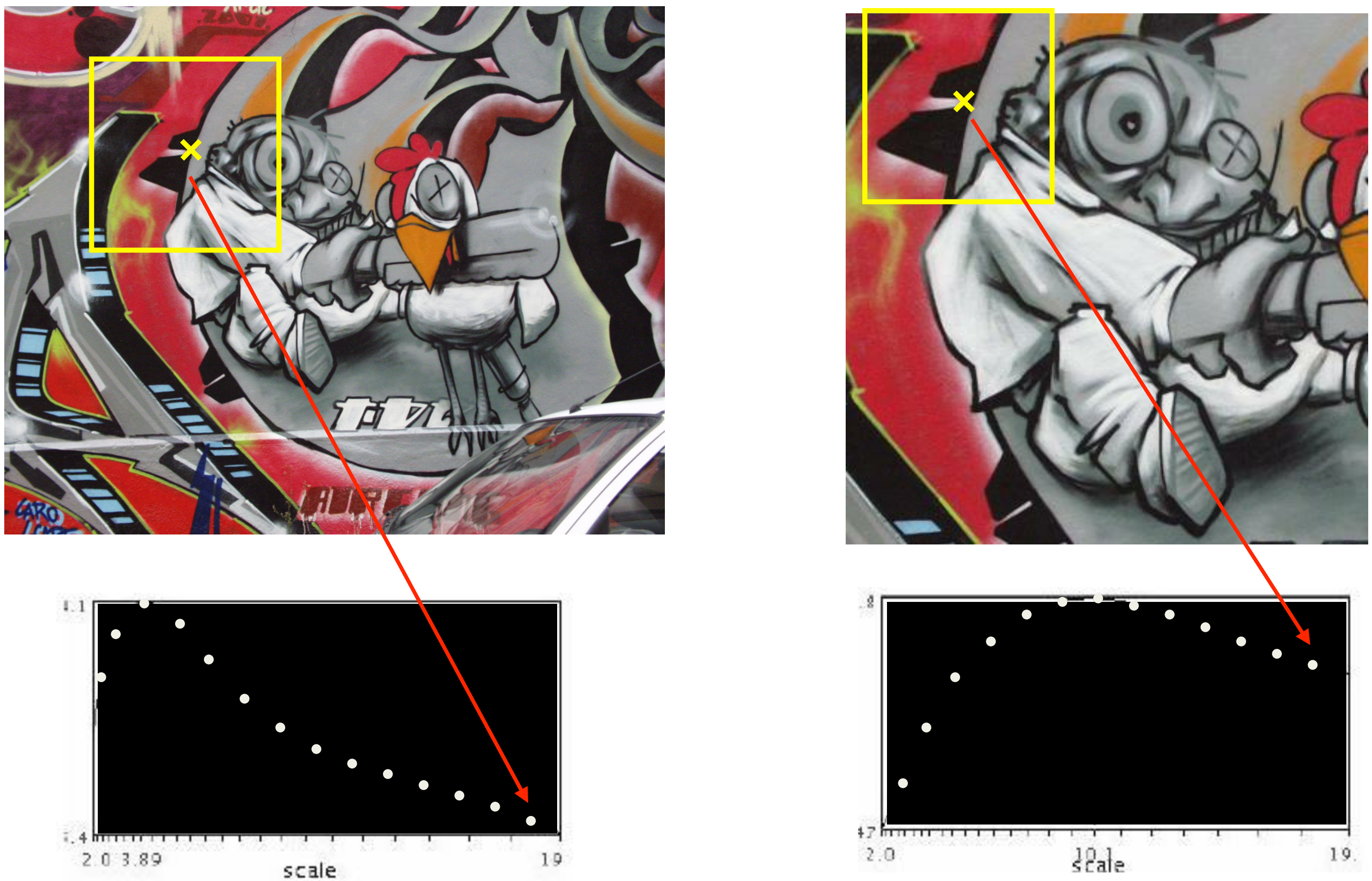
$f(I_{i_1 \ldots i_m}(x,\sigma))$

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

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

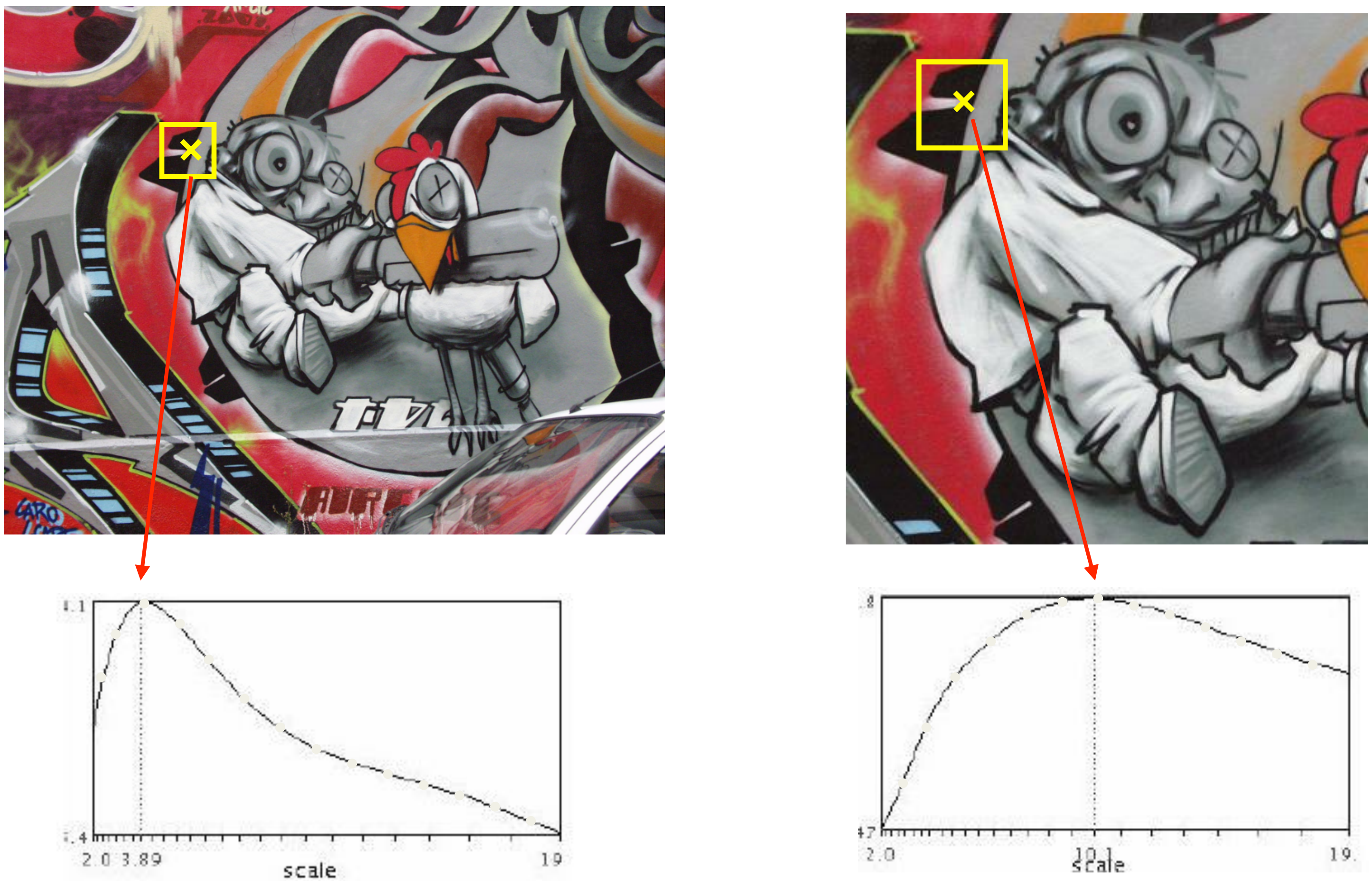


$f(I_{i_1 \ldots i_m}(x,\sigma))$ $\quad$ $f(I_{i_1 \ldots i_m}(x',\sigma))$

K. Grauman, B. Leibe

Automatic Scale Selection

- Function responses for increasing scale (scale signature)

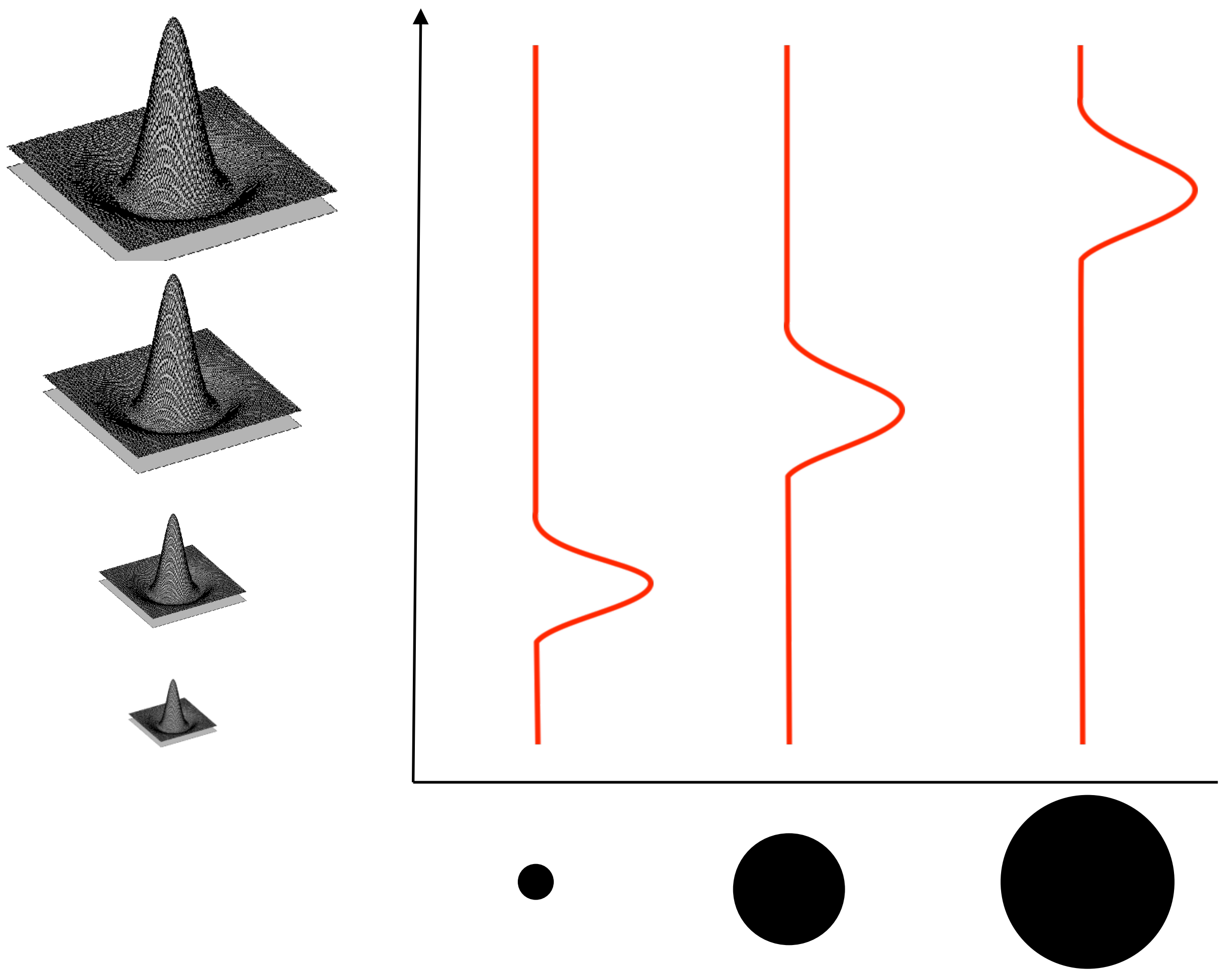


$f(I_{i_1 \dots i_m}(x,\sigma))$ $\qquad$ $f(I_{i_1 \dots i_m}(x',\sigma'))$

K. Grauman, B. Leibe

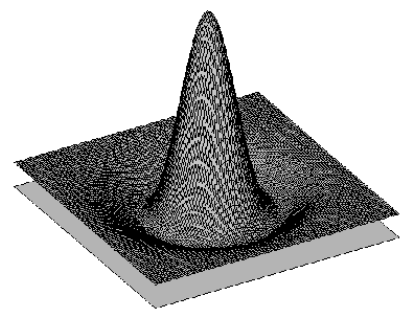
What Is A Useful Signature Function?

- Difference-of-Gaussian = “blob” detector



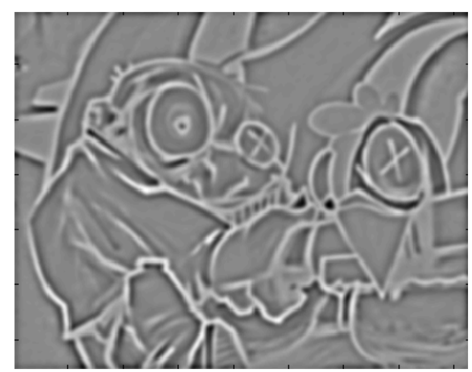
K. Grauman, B. Leibe

Difference-of-Gaussian (DoG)



-

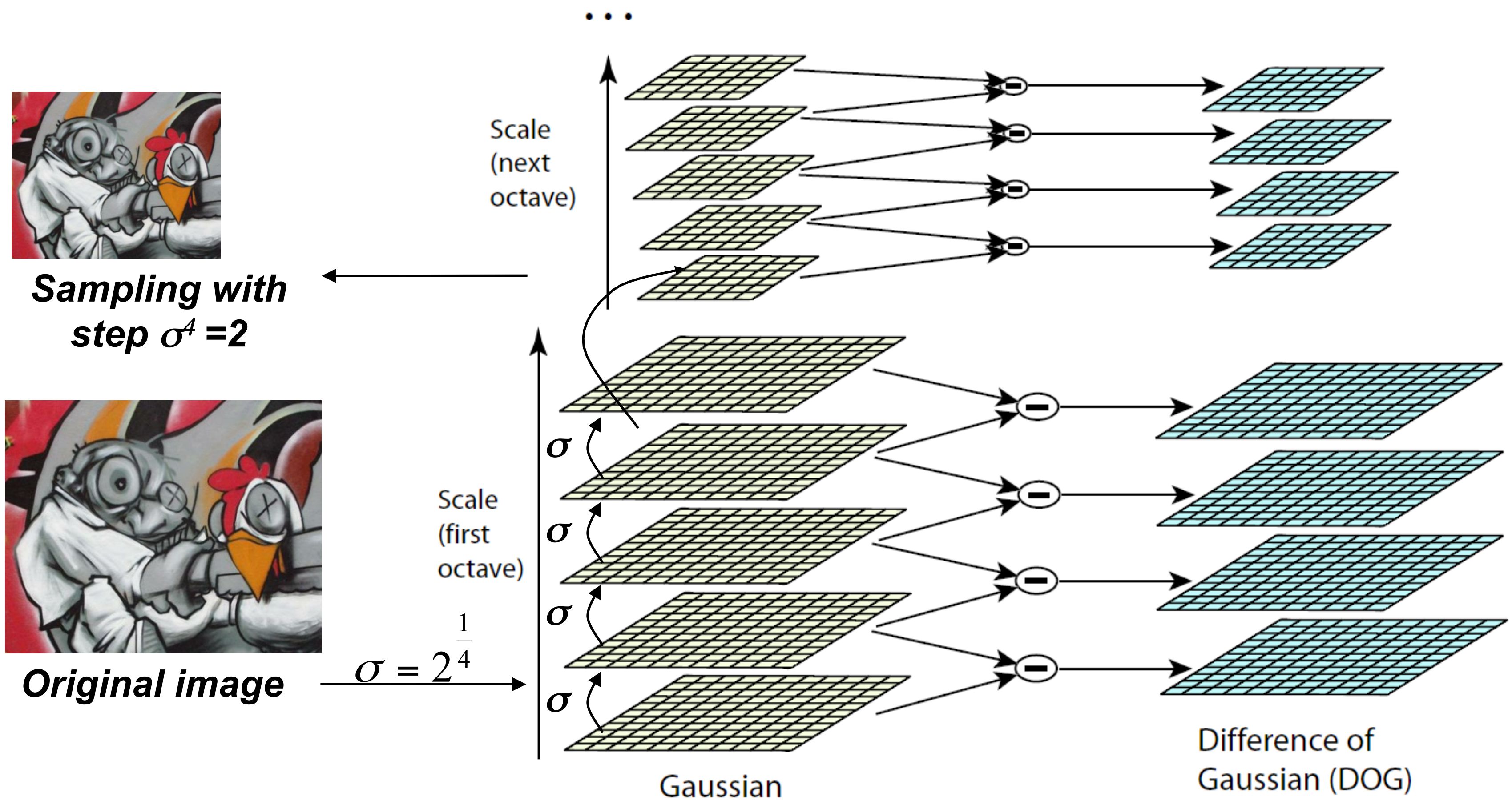
=



K. Grauman, B. Leibe

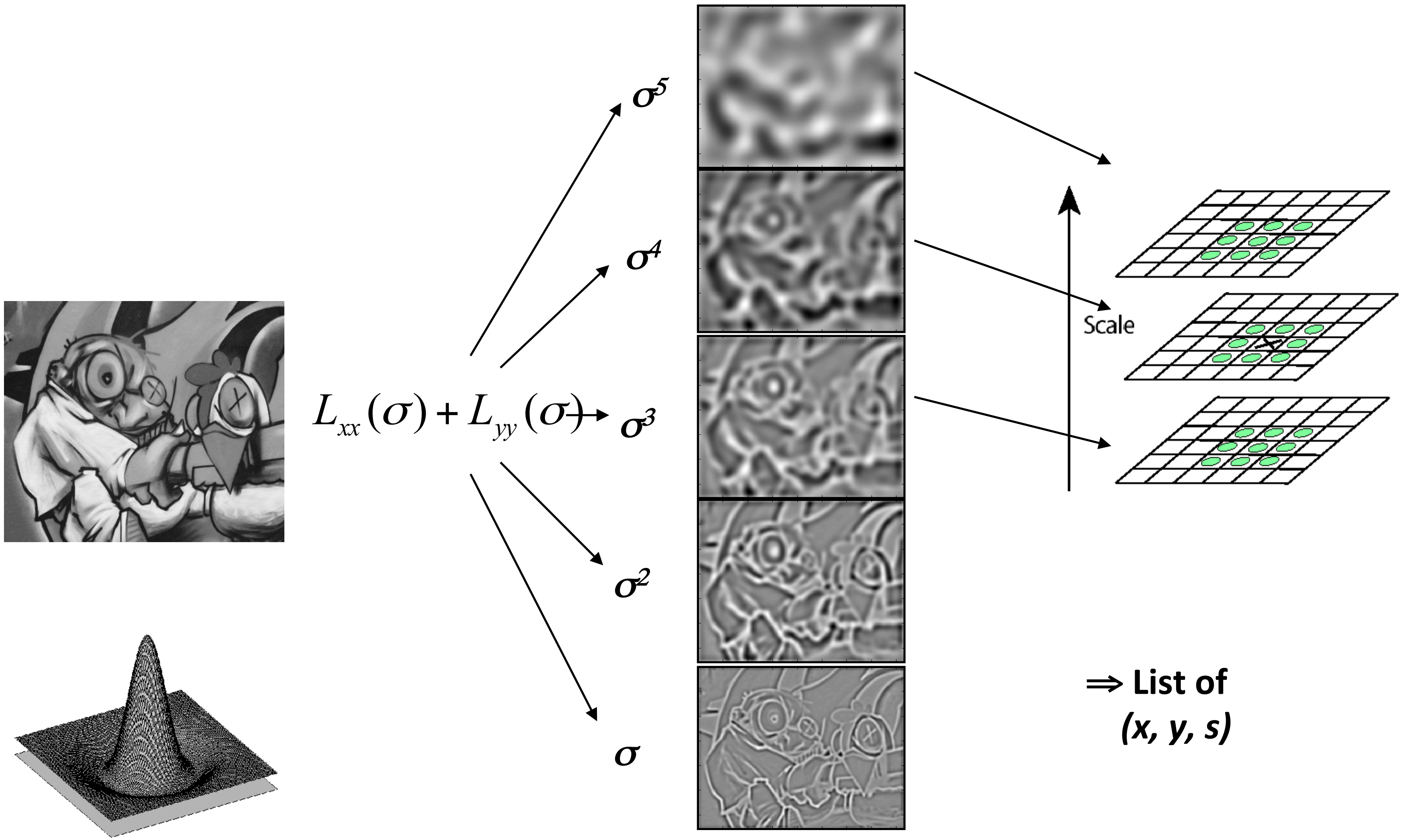
DoG – Efficient Computation

- Computation in Gaussian scale pyramid



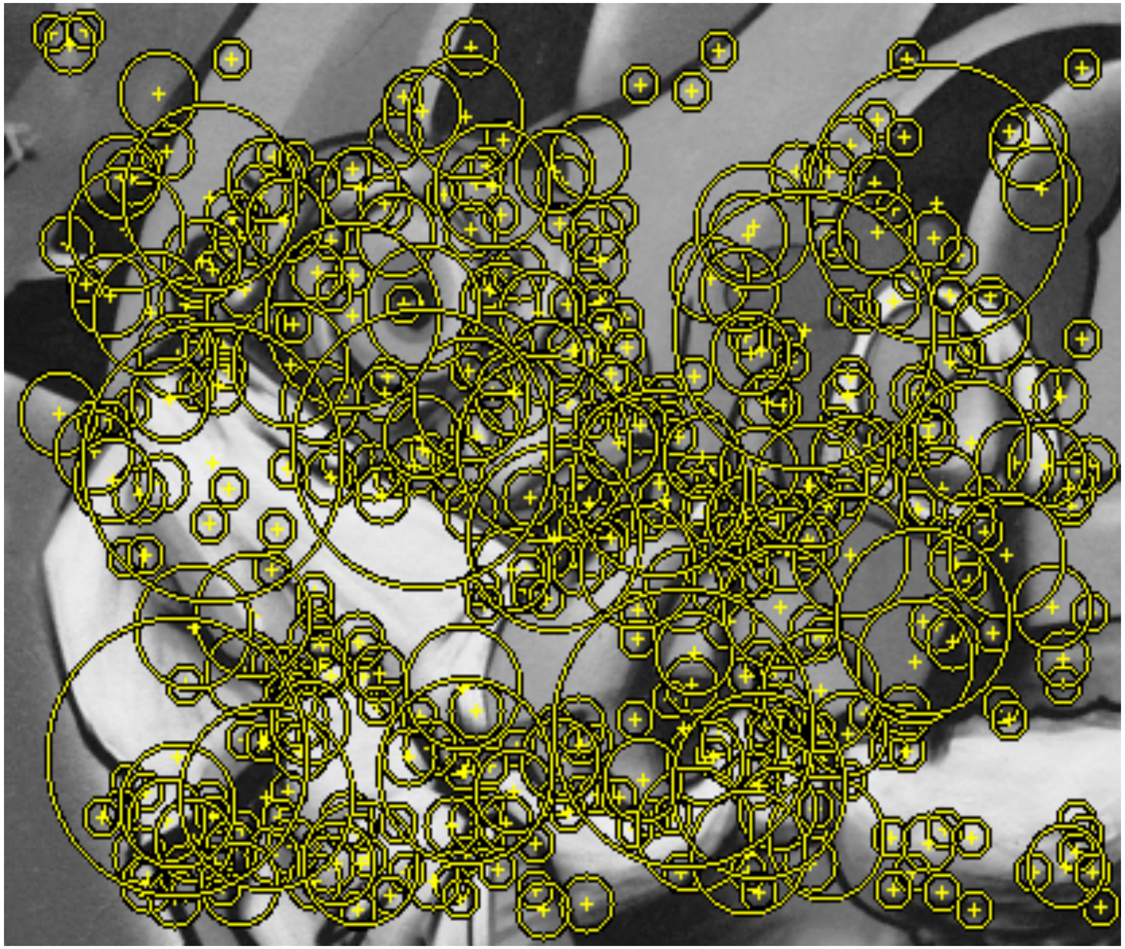
K. Grauman, B. Leibe

Find local maxima in position-scale space of Difference-of-Gaussian



K. Grauman, B. Leibe

Results: Difference-of-Gaussian



K. Grauman, B. Leibe

Choosing a detector

- What do you want it for?
  - Precise localization in x-y: Harris
  - Good localization in scale: Difference of Gaussian
  - Flexible region shape: MSER
- Best choice often application dependent
  - Harris-/Hessian-Laplace/DoG work well for many natural categories
  - MSER works well for buildings and printed things
- Why choose?
  - Get more points with more detectors
- There have been extensive evaluations/comparisons
  - [Mikolajczyk et al., IJCV'05, PAMI'05]
  - All detectors/descriptors shown here work well

Comparison of Keypoint Detectors

Table 7.1 Overview of feature detectors.

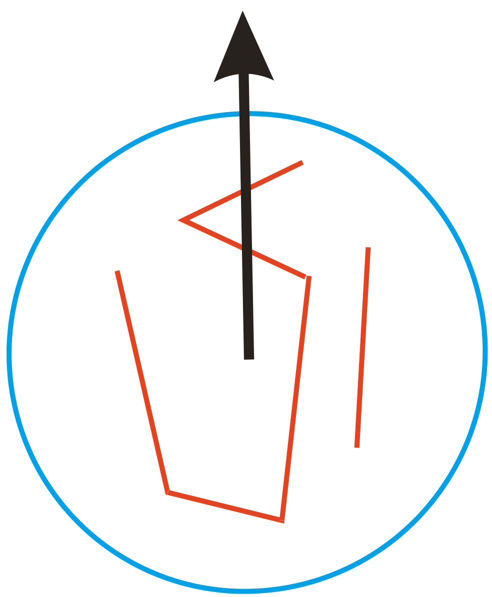
Feature Detector	Corner	Blob	Region	Rotation invariant	Scale invariant	Affine invariant	Repeatability	Localization accuracy	Robustness	Efficiency
Harris	√			√			+++	+++	+++	++
Hessian		√		√			++	++	++	+
SUSAN	√			√			++	++	++	+++
Harris-Laplace	√	(√)		√	√		+++	+++	++	+
Hessian-Laplace	(√)	√		√	√		+++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

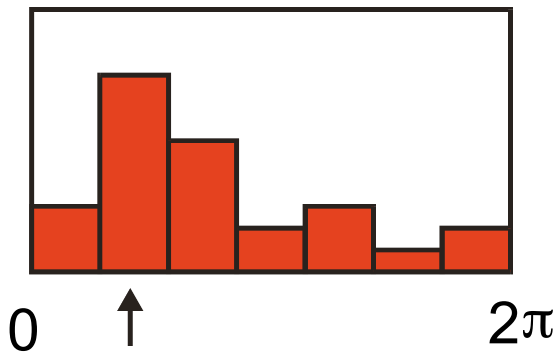
Tuytelaars Mikolajczyk 2008

Orientation Normalization

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

[Lowe, SIFT, 1999]





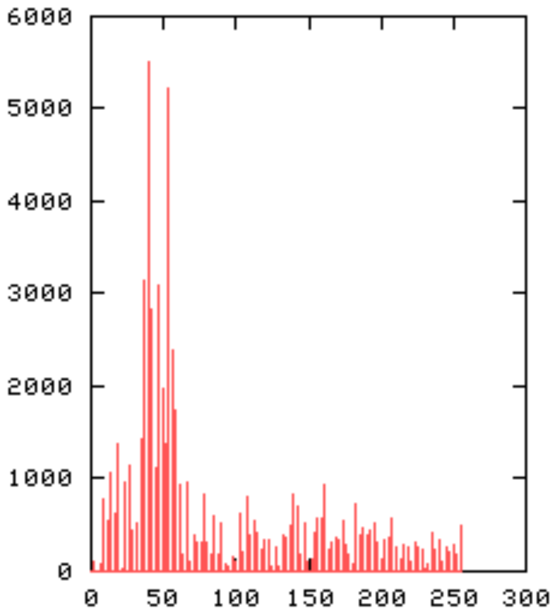
T. Tuytelaars, B. Leibe

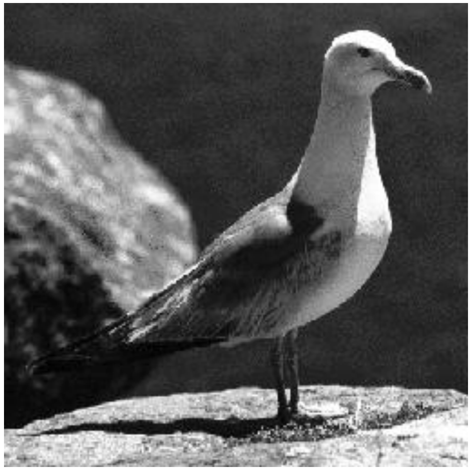
Image representations

- Templates
  - Intensity, gradients, etc.

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

Image Representations: Histograms





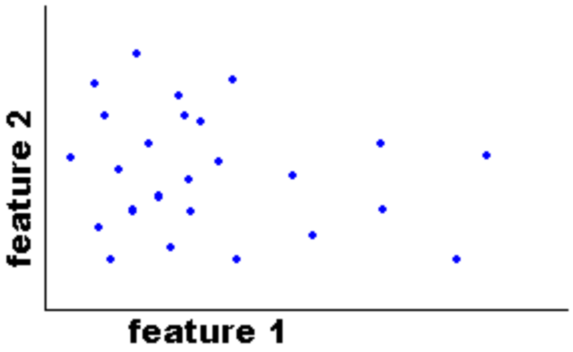
Global histogram

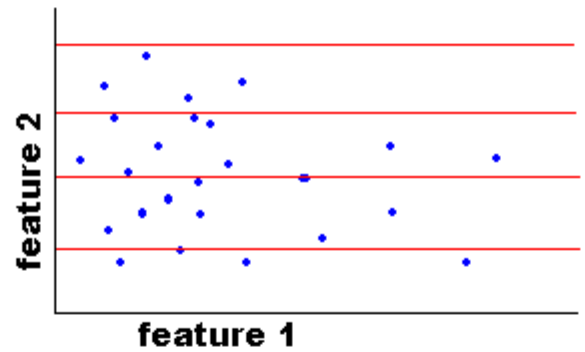
- Represent distribution of features
  - Color, texture, depth, …

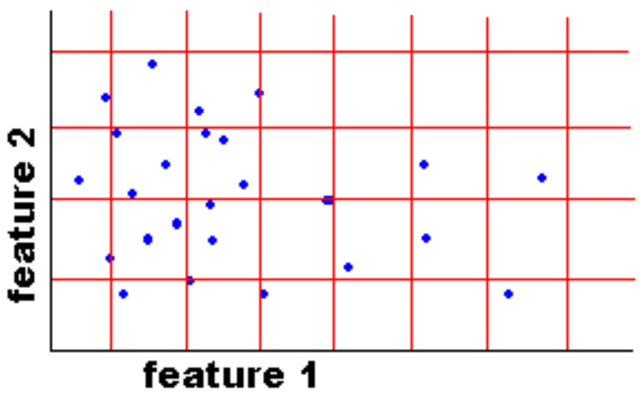
Images from Dave Kauchak

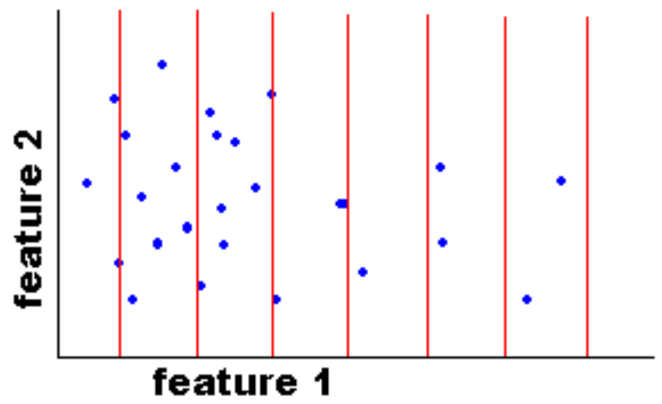
Image Representations: Histograms

Histogram: Probability or count of data in each bin









- Joint histogram
  - Requires lots of data
  - Loss of resolution to avoid empty bins

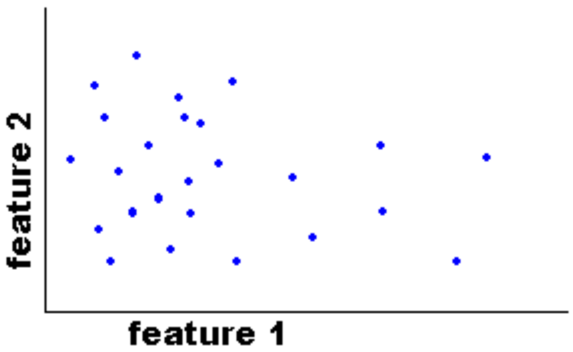
Marginal histogram

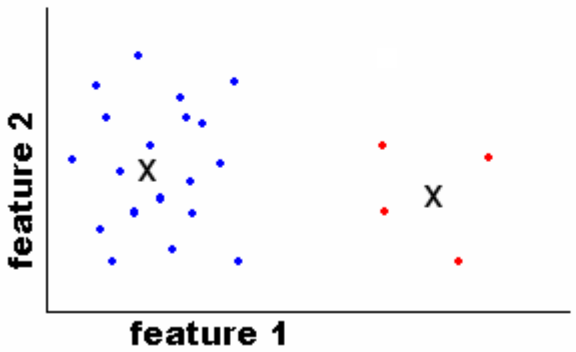
- Requires independent features
- More data/bin than joint histogram

Images from Dave Kauchak

Image Representations: Histograms

Clustering





Use the same cluster centers for all images

Images from Dave Kauchak

Computing histogram distance

$$\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

Histogram intersection (assuming normalized histograms)

$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^{K} \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

Chi-squared Histogram matching distance

Cars found by color histogram matching using chi-squared

Histograms: Implementation issues

- Quantization
  - Grids: fast but applicable only with few dimensions
  - Clustering: slower but can quantize data in higher dimensions

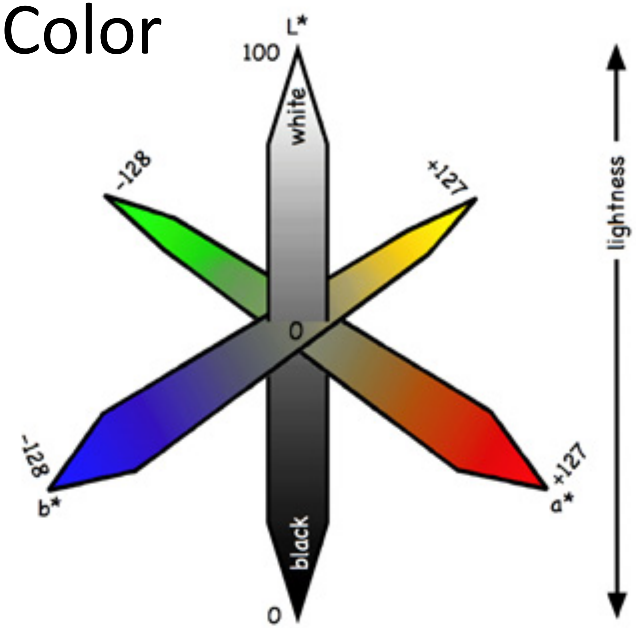
Few Bins
Need less data
Coarser representation

Many Bins
Need more data
Finer representation

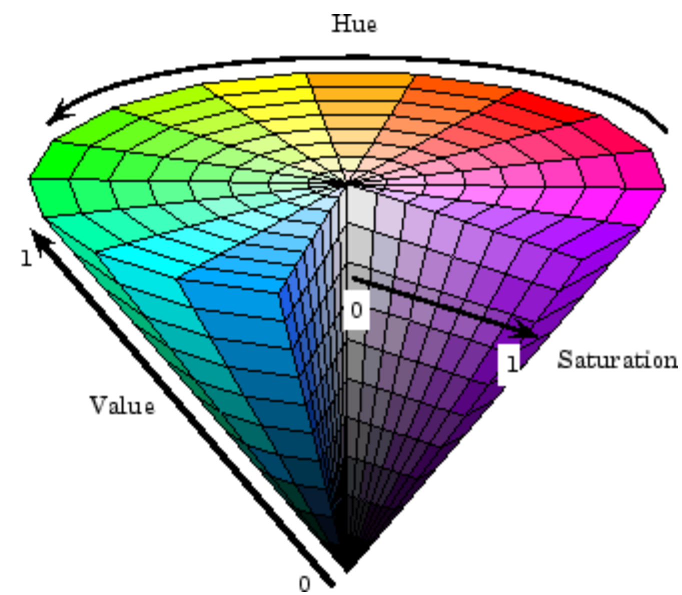
- Matching
  - Histogram intersection or Euclidean may be faster
  - Chi-squared often works better
  - Earth mover's distance is good for when nearby bins represent similar values

What kind of things do we compute histograms of?

- Color



L*a*b* color space

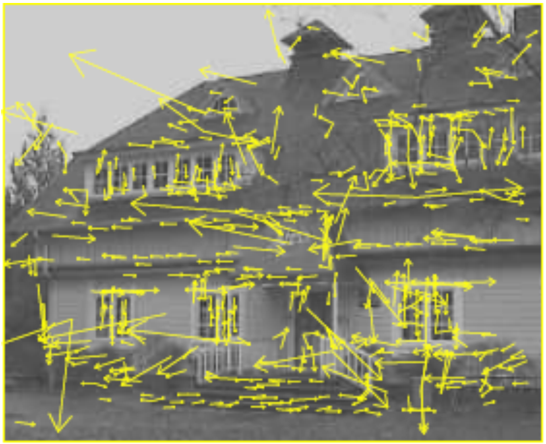


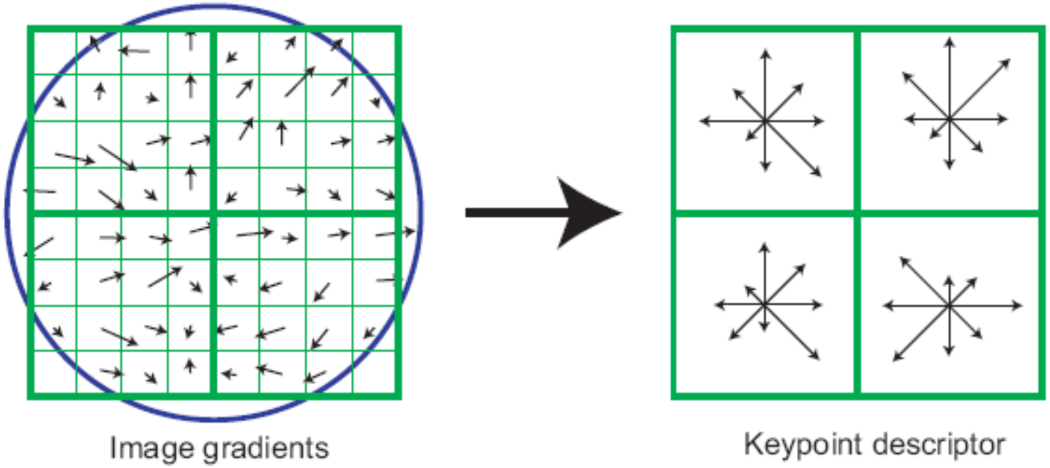
HSV color space

- Texture (filter banks or HOG over regions)

What kind of things do we compute histograms of?

- Histograms of oriented gradients

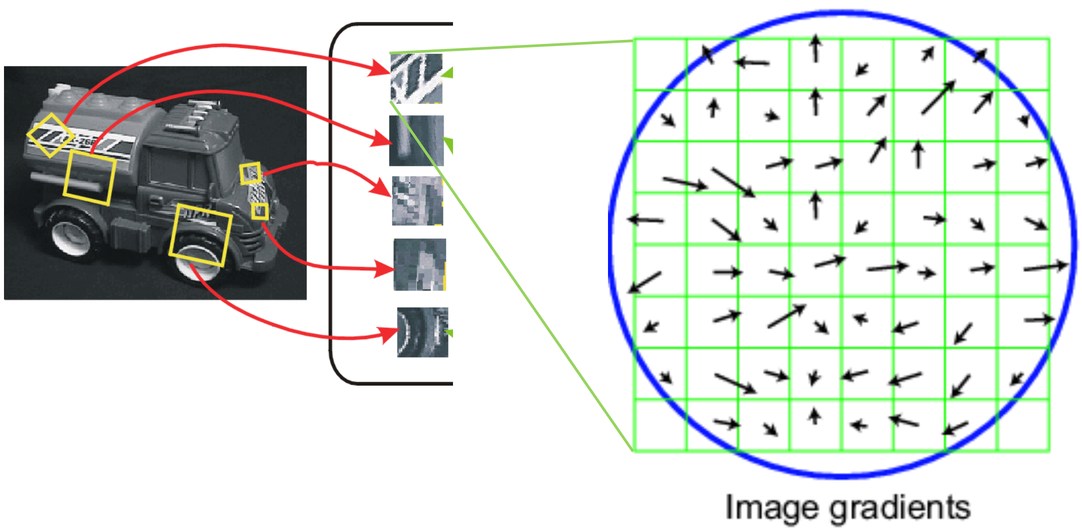




SIFT – Lowe IJCV 2004

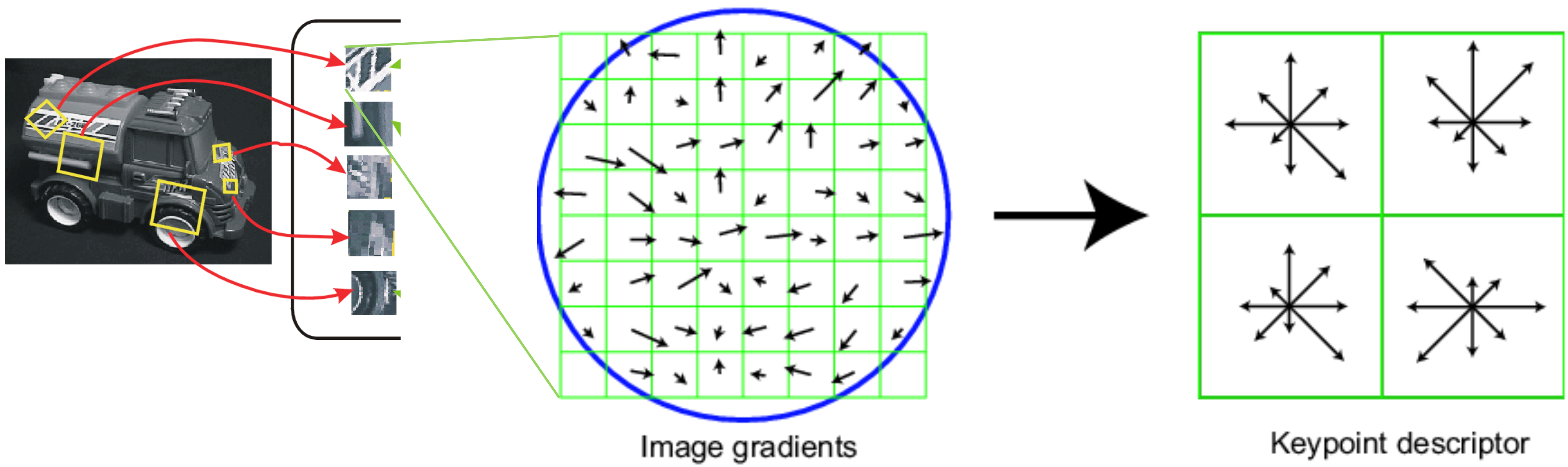
SIFT vector formation

- Computed on rotated and scaled version of window according to computed orientation & scale
  - resample the window
- Based on gradients weighted by a Gaussian of variance half the window (for smooth falloff)



SIFT vector formation

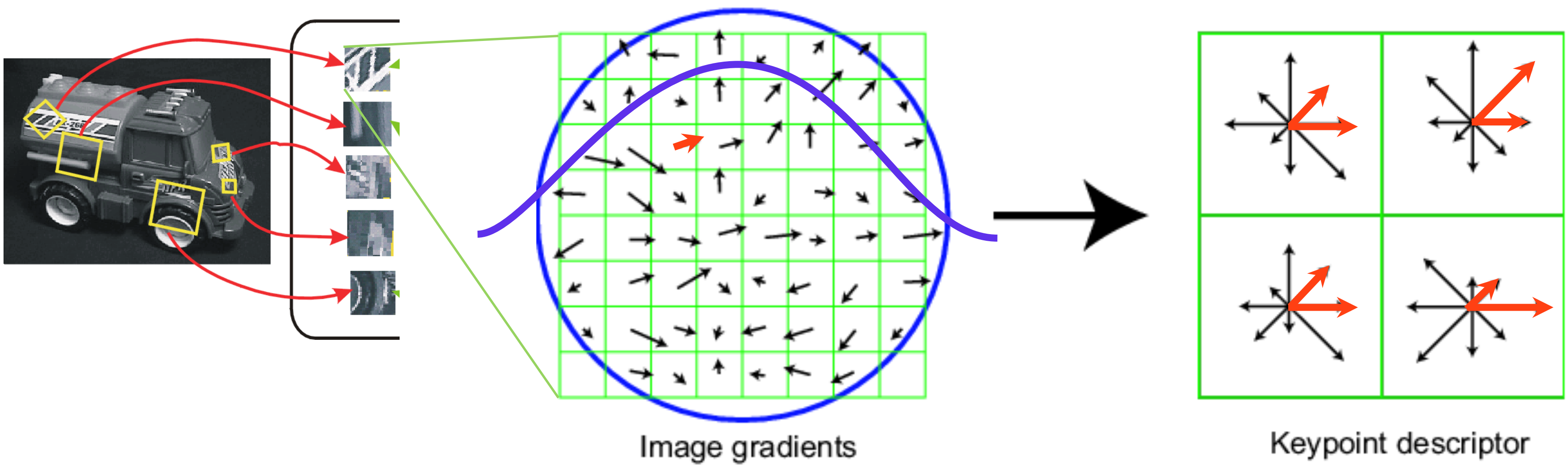
- 4x4 array of gradient orientation histograms weighted by magnitude
- 8 orientations x 4x4 array = 128 dimensions
- Motivation: some sensitivity to spatial layout, but not too much.



showing only 2x2 here but is 4x4

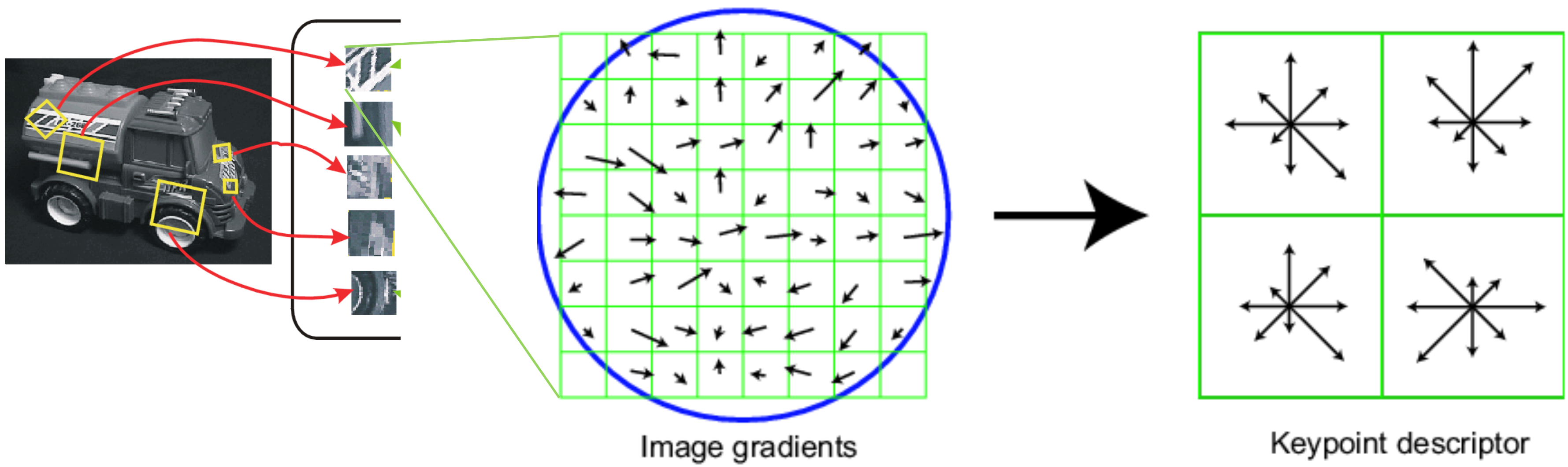
Ensure smoothness

- Gaussian weight
- Trilinear interpolation
  - a given gradient contributes to 8 bins:
    4 in space times 2 in orientation

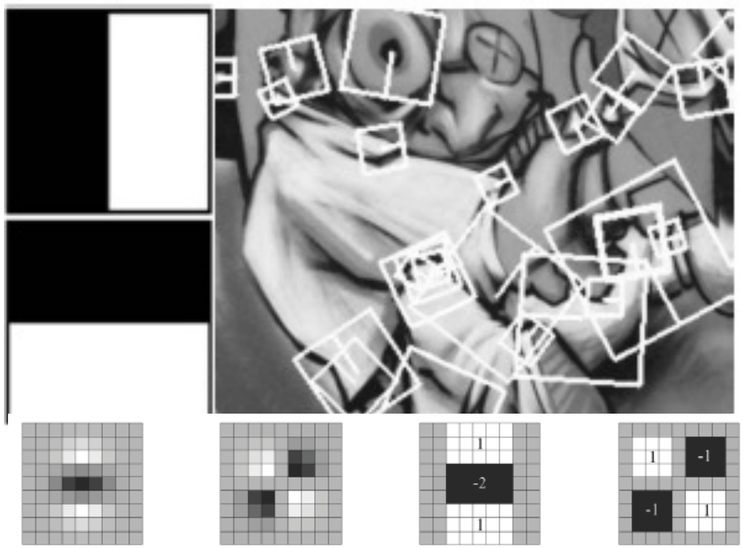


Reduce effect of illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients >0.2
  - renormalize



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

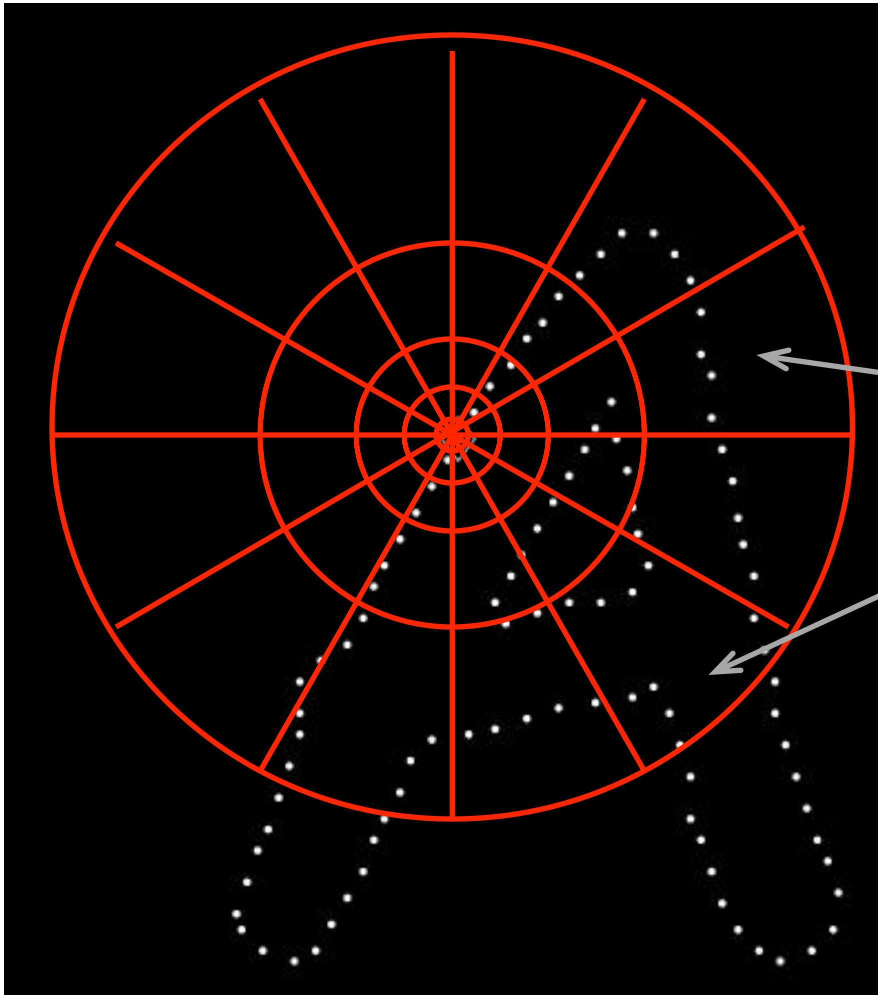
GPU implementation available

- 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

⋮

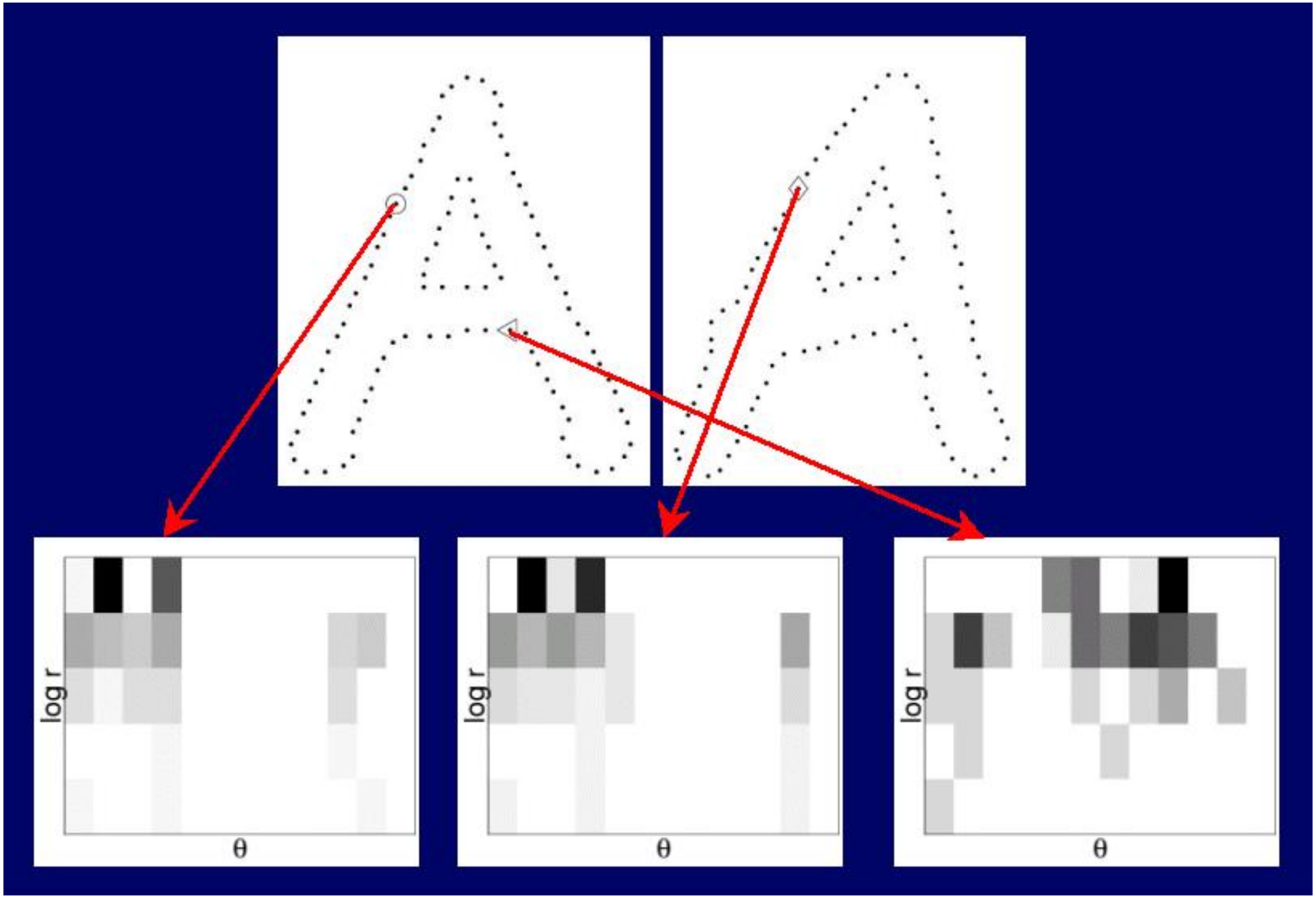
Count = 10

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

Belongie & Malik, ICCV 2001

K. Grauman, B. Leibe

Shape Context Descriptor



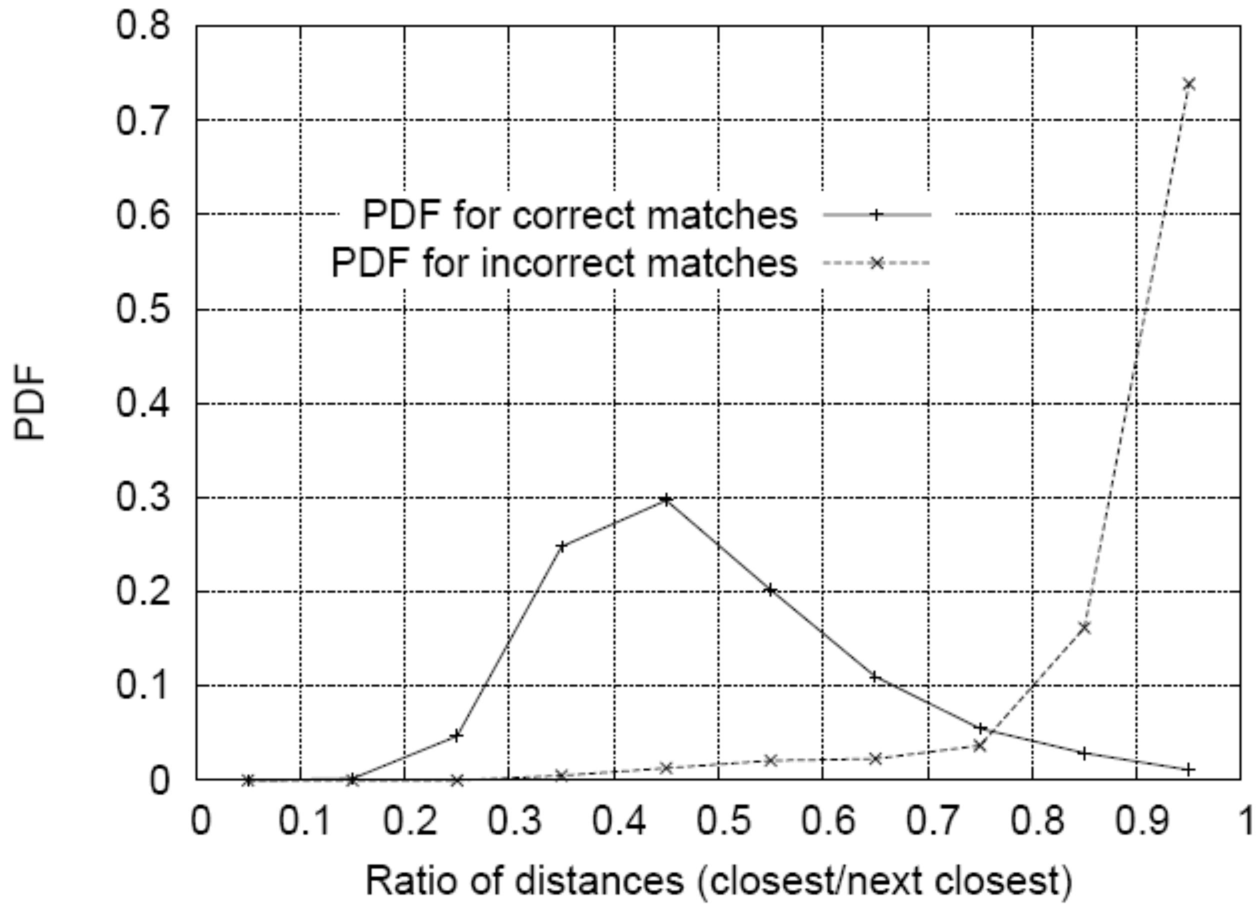
Local Descriptors

- Most features can be thought of as templates, histograms (counts), or combinations
- The ideal descriptor should be
  - Robust
  - Distinctive
  - Compact
  - Efficient
- Most available descriptors focus on edge/gradient information
  - Capture texture information
  - Color rarely used

K. Grauman, B. Leibe

Matching Local Features

- Nearest neighbor (Euclidean distance)
- Threshold ratio of nearest to 2nd nearest descriptor



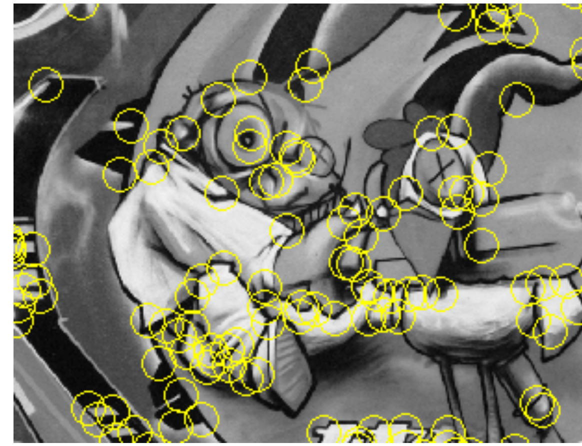
Lowe IJCV 2004

Choosing a descriptor

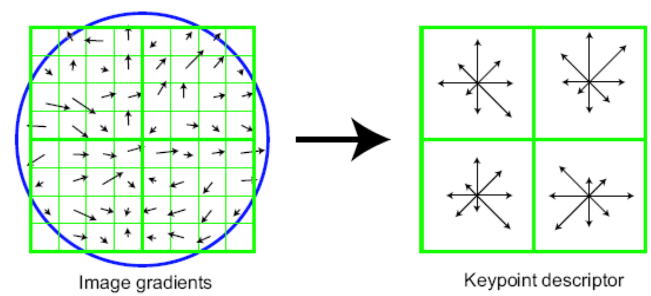
- Again, need not stick to one
- For object instance recognition or stitching, SIFT or variant is a good choice

Things to remember

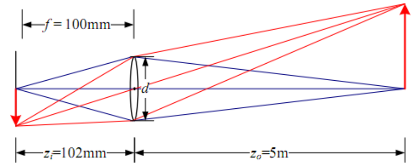
- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG



- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT



Course Outline



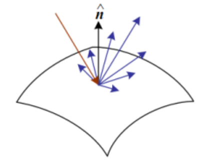
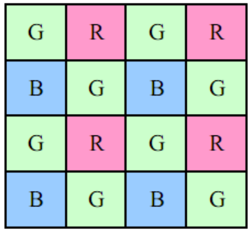


Image Formation and Processing

Light, Shape and Color

The Pin-hole Camera Model, The Digital Camera

Linear filtering, Template Matching, Image Pyramids

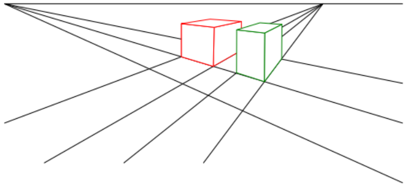


Feature Detection and Matching

Edge Detection, Interest Points: Corners and Blobs

Local Image Descriptors

Feature Matching and Hough Transform



Multiple Views and Motion

Geometric Transformations, Camera Calibration

Feature Tracking , Stereo Vision

Segmentation and Grouping

Segmentation by Clustering, Region Merging and Growing

Advanced Methods Overview: Active Contours, Level-Sets, Graph-Theoretic Methods

Detection and Recognition

Problems and Architectures Overview

Statistical Classifiers, Bag-of-Words Model, Detection by Sliding Windows

Resources

Books

R. Szeliski, Computer Vision: Algorithms and Applications, 2010 – *available online*

D. A. Forsyth and J. Ponce, Computer Vision: A Modern Approach, 2003

L. G. Shapiro and G. C. Stockman, Computer Vision, 2001

Web

CVonline: The Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision

http://homepages.inf.ed.ac.uk/rbf/CVonline/

Dictionary of Computer Vision and Image Processing

http://homepages.inf.ed.ac.uk/rbf/CVDICT/

Computer Vision Online

http://www.computervisiononline.com/

Programming

Development environments/languages: Matlab, Python and C/C++

Toolboxes and APIs: OpenCV, VLFeat Matlab Toolbox, Piotr's Computer Vision Matlab Toolbox, EasyCamCalib Software, FLANN, Point Cloud Library PCL, LibSVM, Camera Calibration Toolbox for Matlab