RWTH AACHEN UNIVERSITY

Computer Vision - Lecture 11

Local Features II

30.11.2016

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

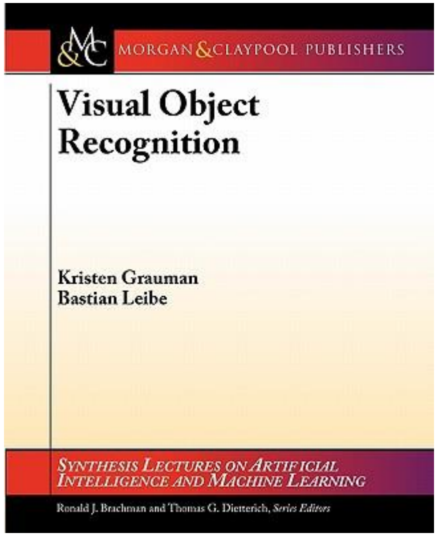
leibe@vision.rwth-aachen.de

Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Object Categorization I
  - Sliding Window based Object Detection
- Local Features & Matching
  - Local Features - Detection and Description
  - Recognition with Local Features
- Object Categorization II
  - Part based Approaches
  - Deep Learning Approaches
- 3D Reconstruction
- Motion and Tracking

A Script…

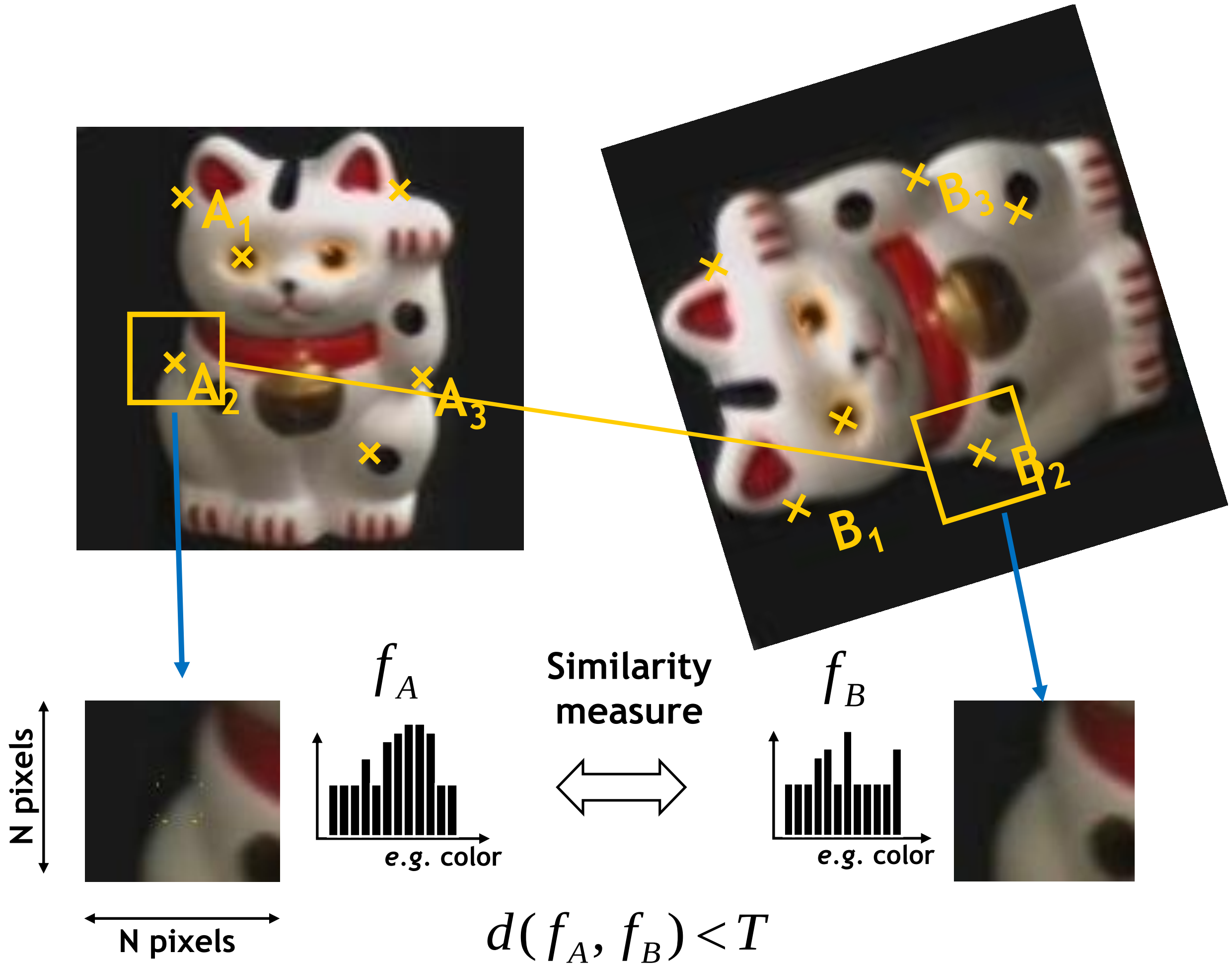
- We've created a script… for the part of the lecture on object recognition & categorization
  - K. Grauman, B. Leibe
    Visual Object Recognition
    Morgan & Claypool publishers, 2011



- Chapter 3: Local Feature Extraction (Last+this lecture)
- Chapter 4: Matching (Monday's topic)
- Chapter 5: Geometric Verification (Wednesday's topic)

- Available on the L2P -

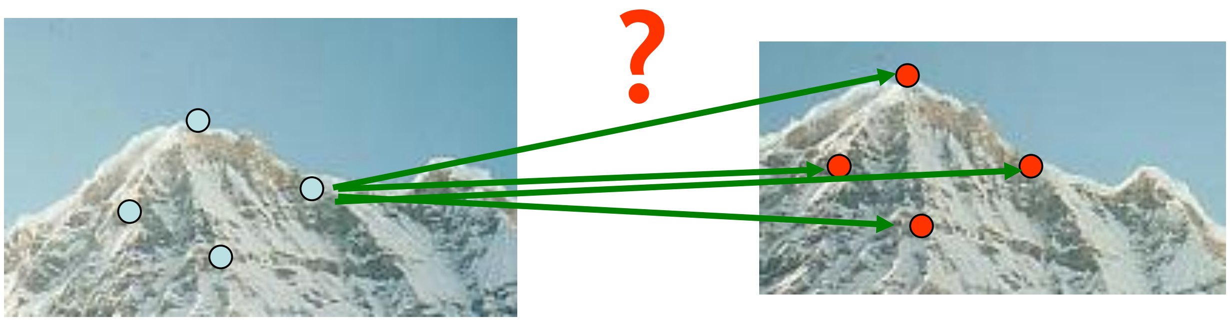
Recap: Local Feature Matching Outline



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

Recap: Requirements for Local Features

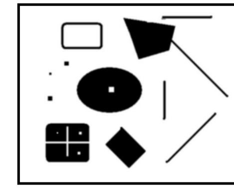
- **Problem 1:**
  - Detect the same point *independently* in both images
- **Problem 2:**
  - For each point correctly recognize the corresponding one



We need a repeatable detector!

We need a reliable and distinctive descriptor!

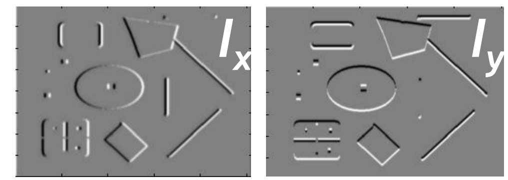
Recap: Harris Detector [Harris88]



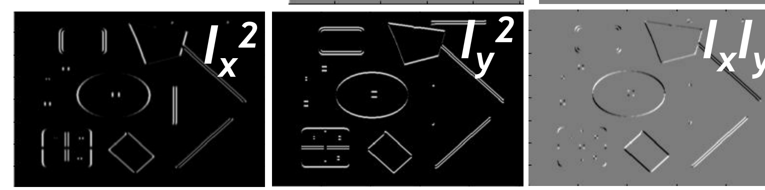
- Compute second moment matrix (autocorrelation matrix)

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

1. Image derivatives



2. Square of derivatives



3. Gaussian filter $g(\sigma_I)$

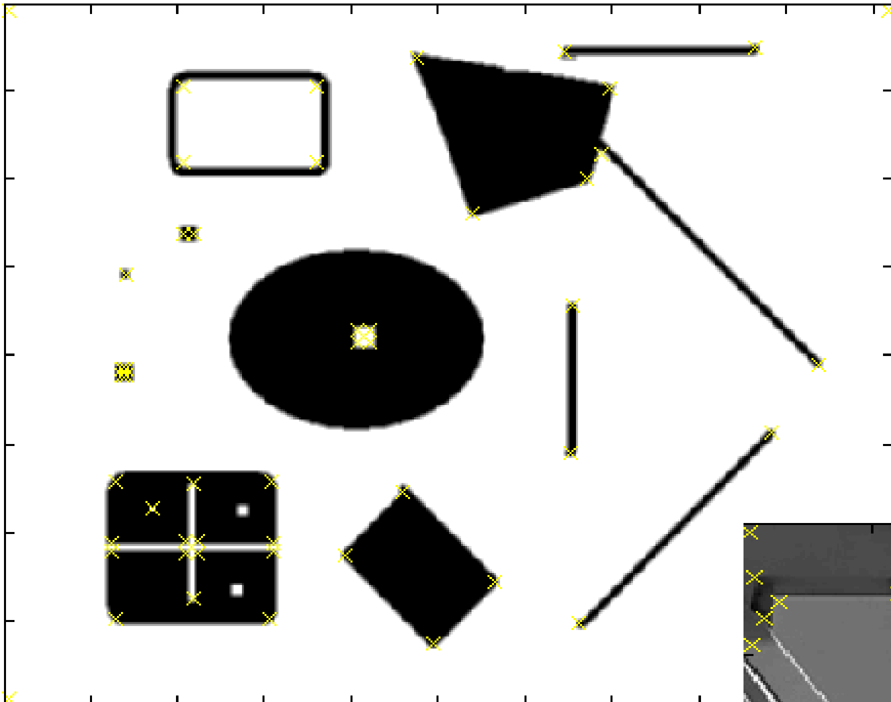
4. Cornerness function - two strong eigenvalues

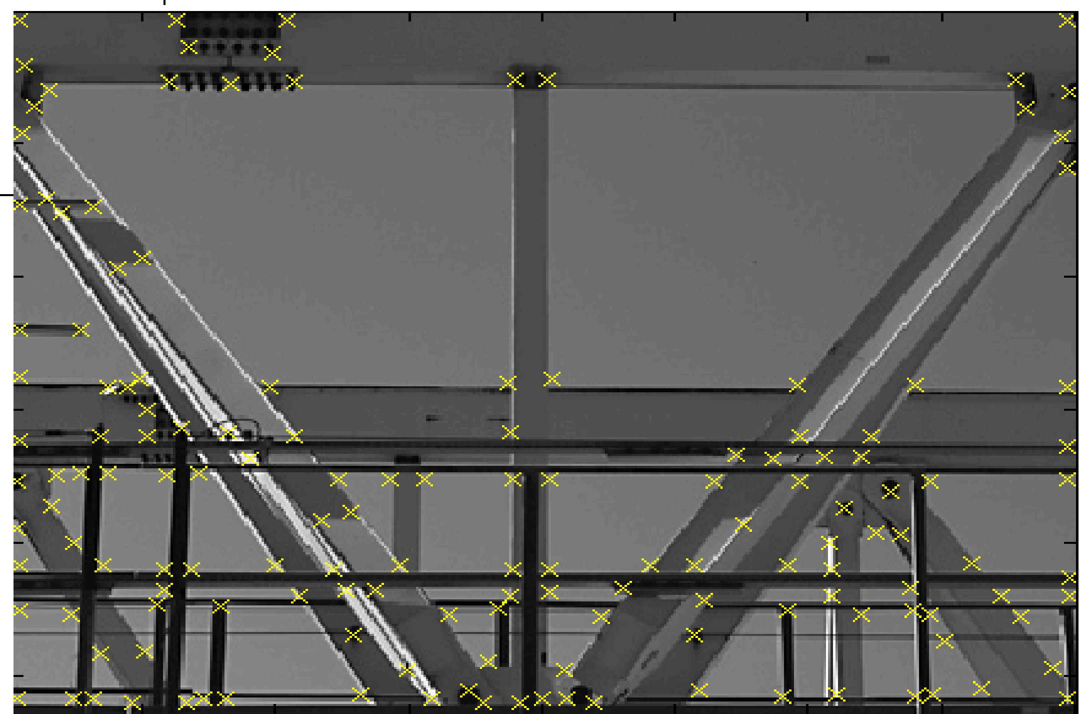
$$R=\det[M(\sigma_I,\sigma_D)]-\alpha[\operatorname{trace}(M(\sigma_I,\sigma_D))]^2$$
$$=g(I_x^2)g(I_y^2)-[g(I_xI_y)]^2-\alpha[g(I_x^2)+g(I_y^2)]^2$$

5. Perform non-maximum suppression

Slide credit: Krystian Mikolajczyk

Recap: Harris Detector Responses [Harris88]





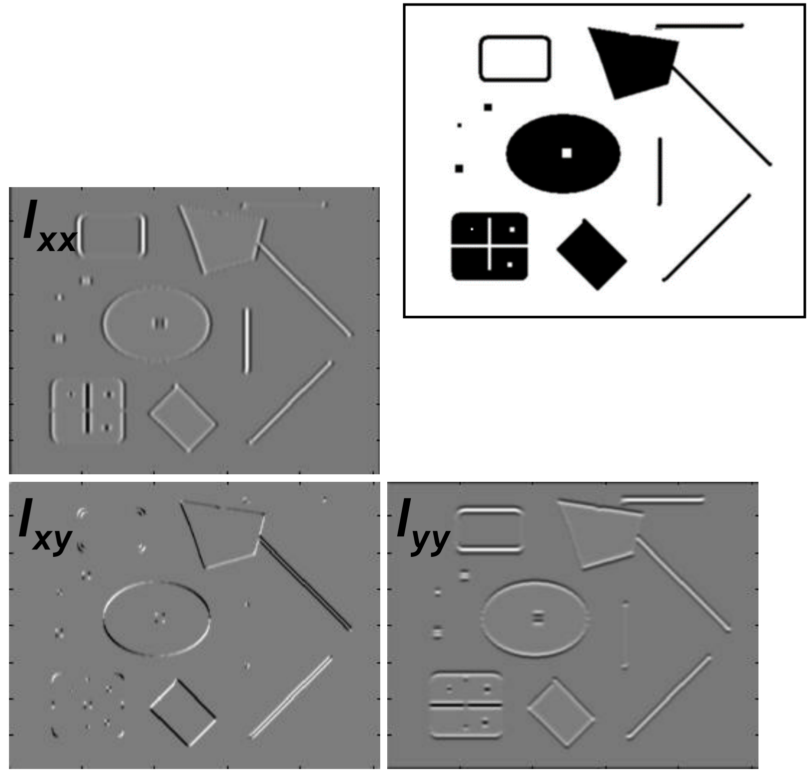
Effect: A very precise corner detector.

Hessian Detector [Beaudet78]

- **Hessian determinant**

$$Hessian(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

Note: these are 2nd derivatives!

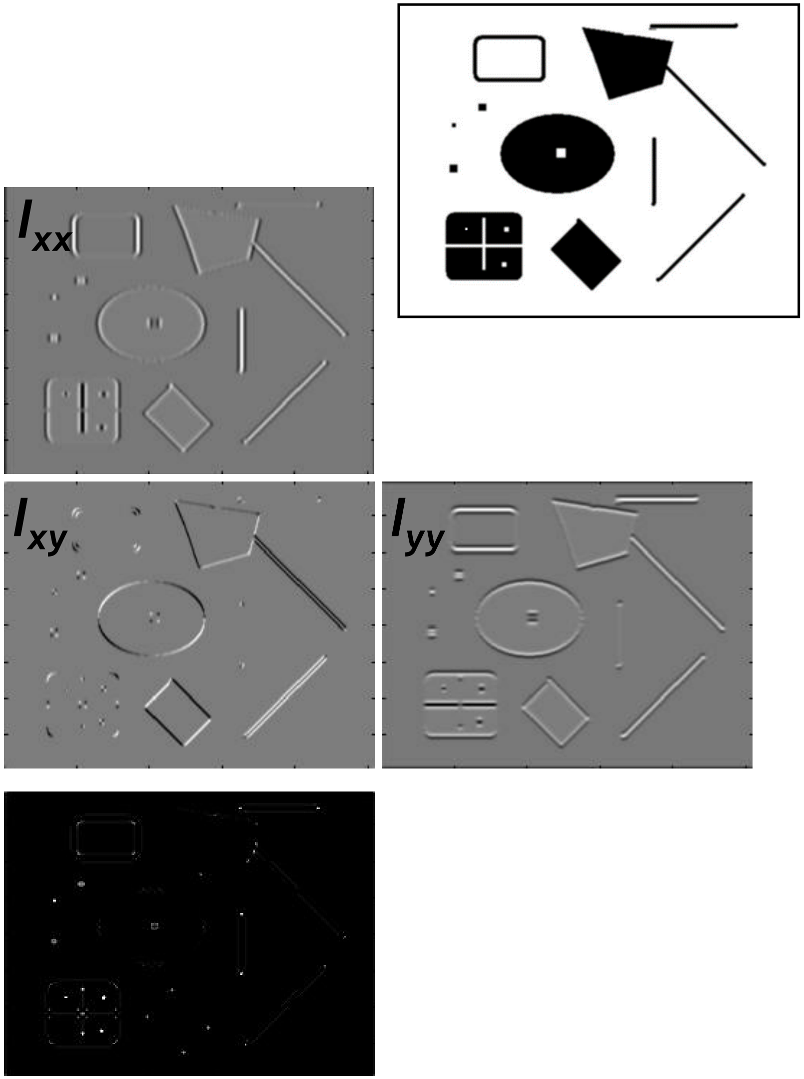


Intuition: **Search for strong derivatives in two orthogonal directions**

Hessian Detector [Beaudet78]

- **Hessian determinant**

$$Hessian(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

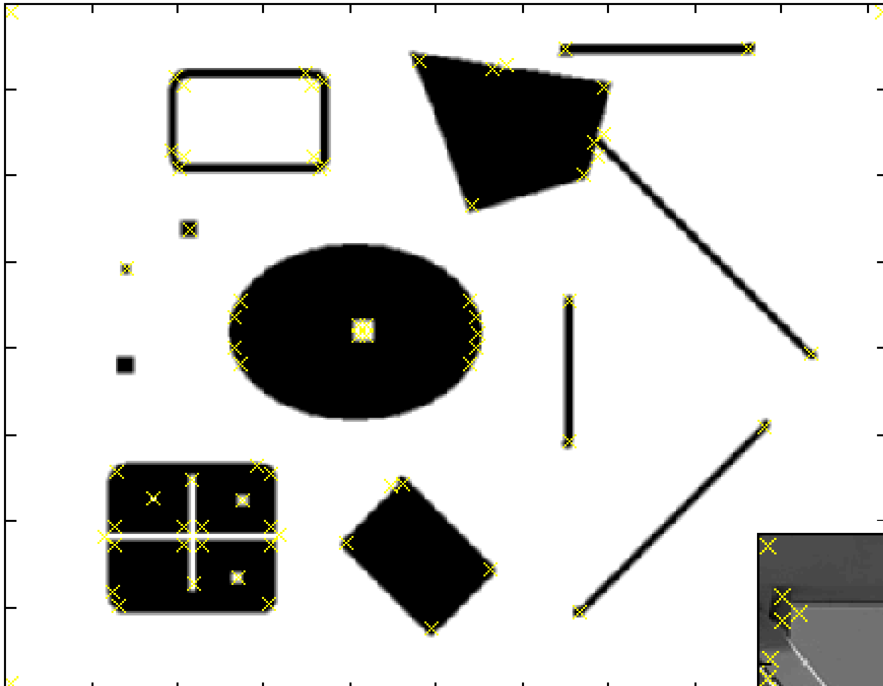


$$\det(Hessian(I)) = I_{xx} I_{yy} - I_{xy}^2$$

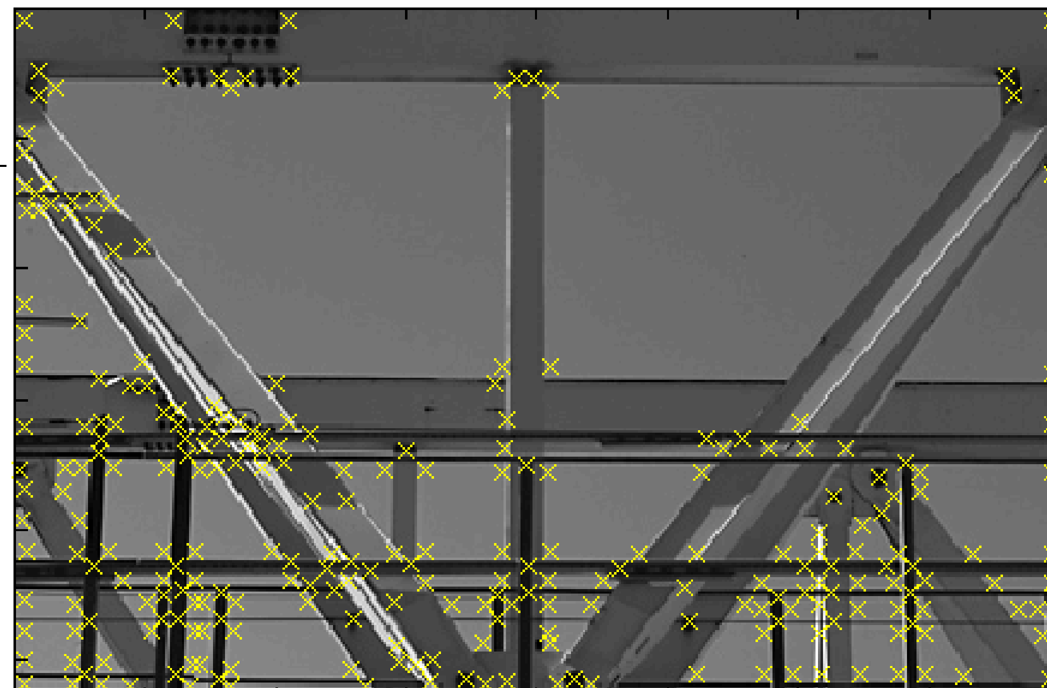
In Matlab:

$$I_{xx} .* I_{yy} - (I_{xy})\wedge 2$$

Hessian Detector – Responses [Beaudet78]



Effect: Responses mainly on corners and strongly textured areas.



Slide credit: Krystian Mikolajczyk

Hessian Detector – Responses [Beaudet78]

Slide credit: Krystian Mikolajczyk

Topics of This Lecture

- Local Feature Extraction (cont'd)
  - Scale Invariant Region Selection
  - Orientation normalization
  - Affine Invariant Feature Extraction
- Local Descriptors
  - SIFT
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

From Points to Regions…

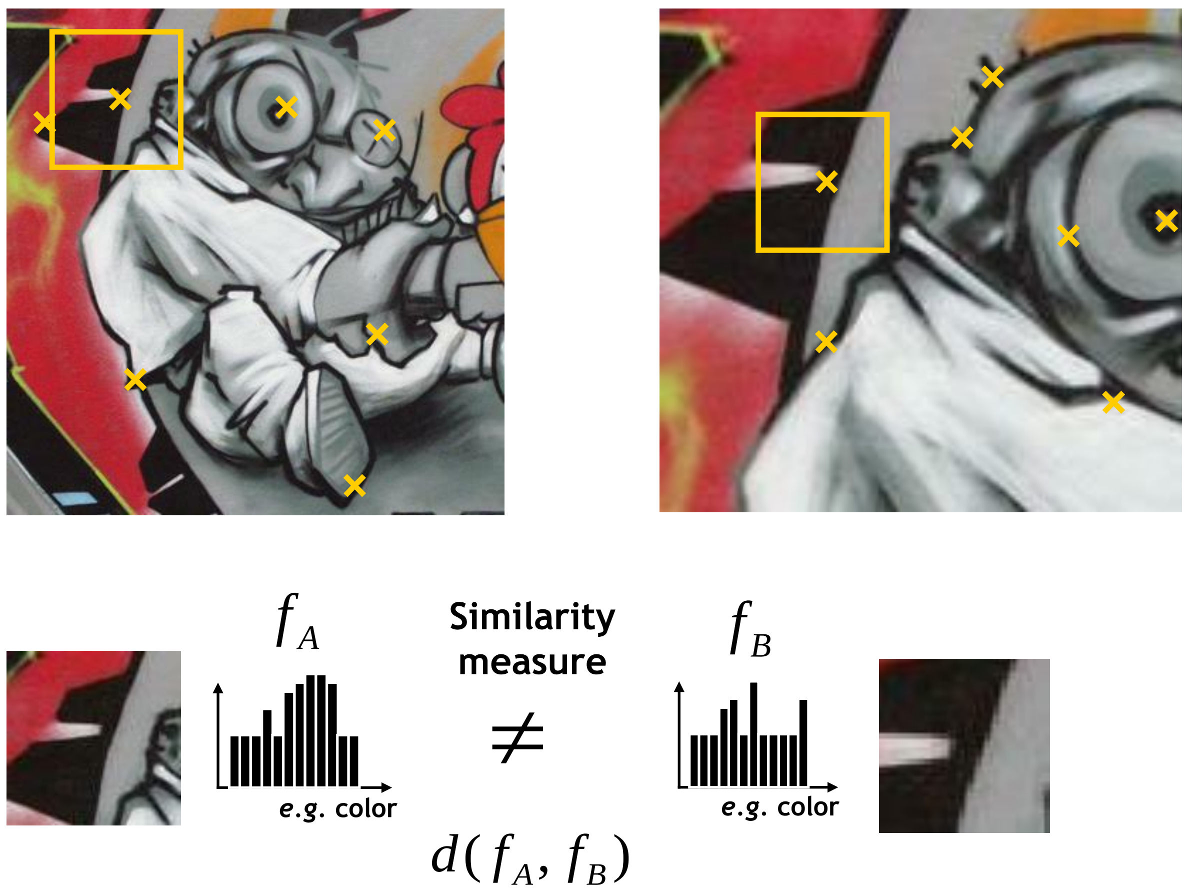
- The Harris and Hessian operators define interest points.
  - Precise localization
  - High repeatability

- In order to compare those points, we need to compute a descriptor over a region.
  - How can we define such a region in a scale invariant manner?

- *I.e. how can we detect scale invariant interest regions?*

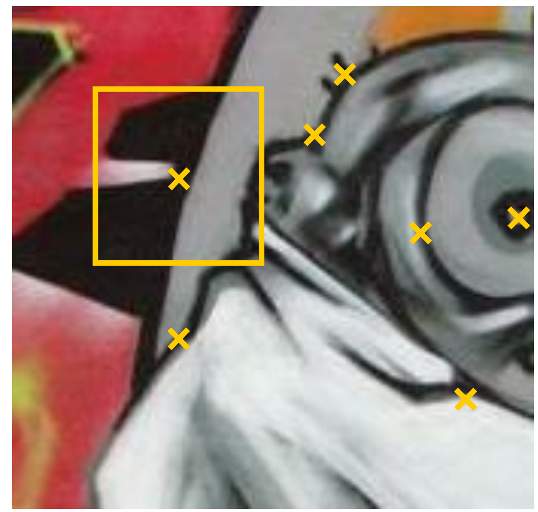
Naïve Approach: Exhaustive Search

- **Multi-scale procedure**
  - Compare descriptors while varying the patch size

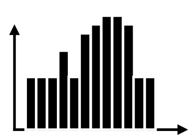


Naïve Approach: Exhaustive Search

- **Multi-scale procedure**
  - Compare descriptors while varying the patch size



f_A

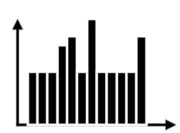


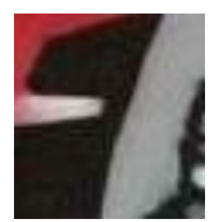
Similarity measure

$\neq$

$d(f_A, f_B)$

f_B

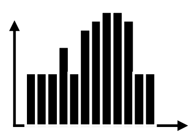




Naïve Approach: Exhaustive Search

- **Multi-scale procedure**
  - Compare descriptors while varying the patch size

f_A

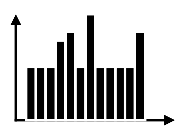


Similarity measure

$$\neq$$

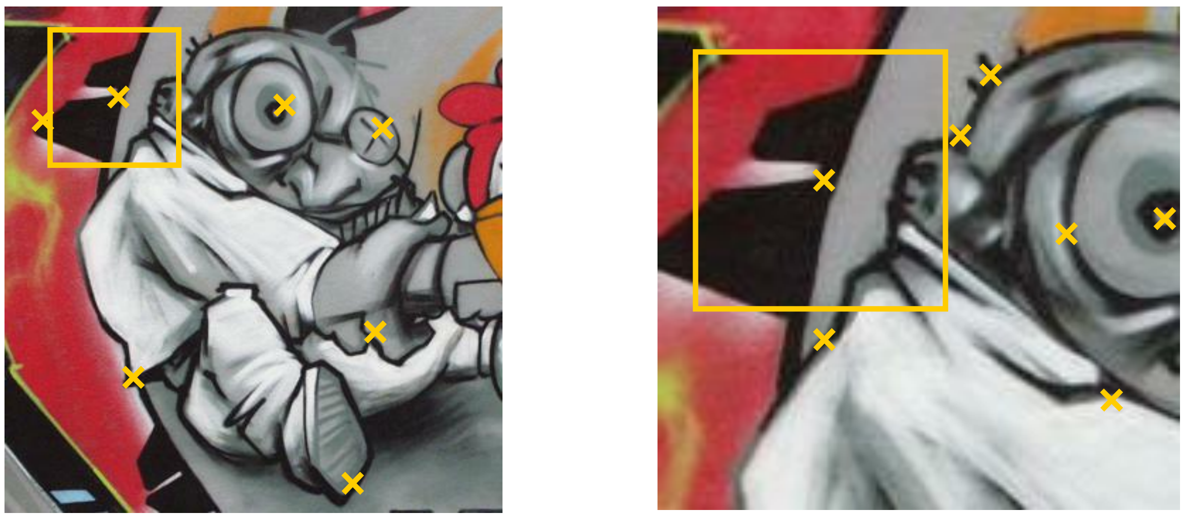
$$d(f_A, f_B)$$

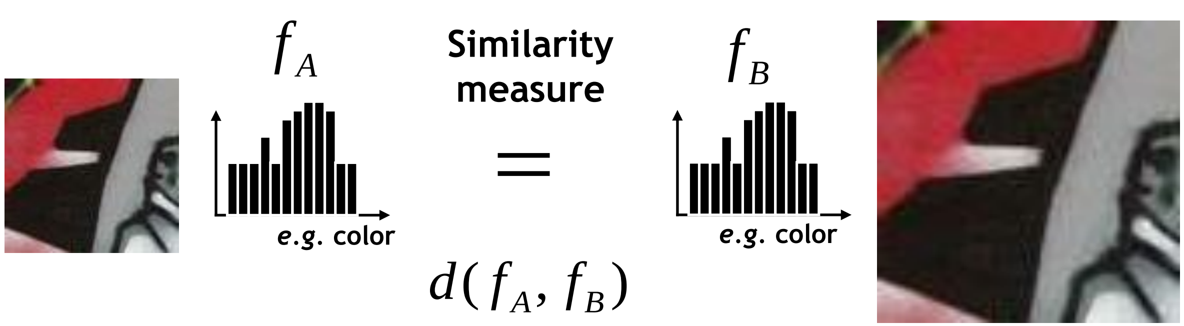
f_B



Naïve Approach: Exhaustive Search

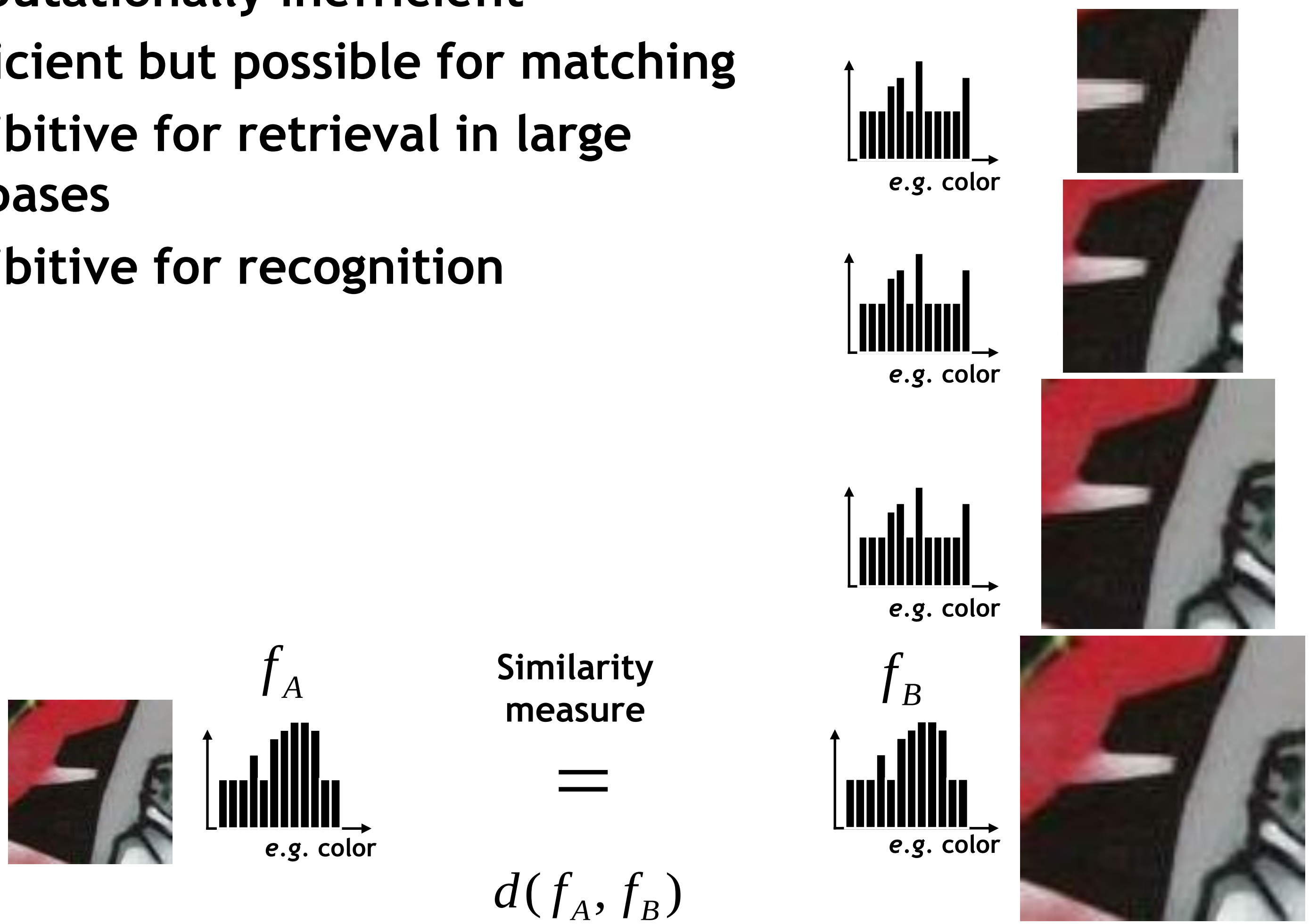
- **Multi-scale procedure**
  - Compare descriptors while varying the patch size





Naïve Approach: Exhaustive Search

- Comparing descriptors while varying the patch size
  - Computationally inefficient
  - Inefficient but possible for matching
  - Prohibitive for retrieval in large databases
  - Prohibitive for recognition

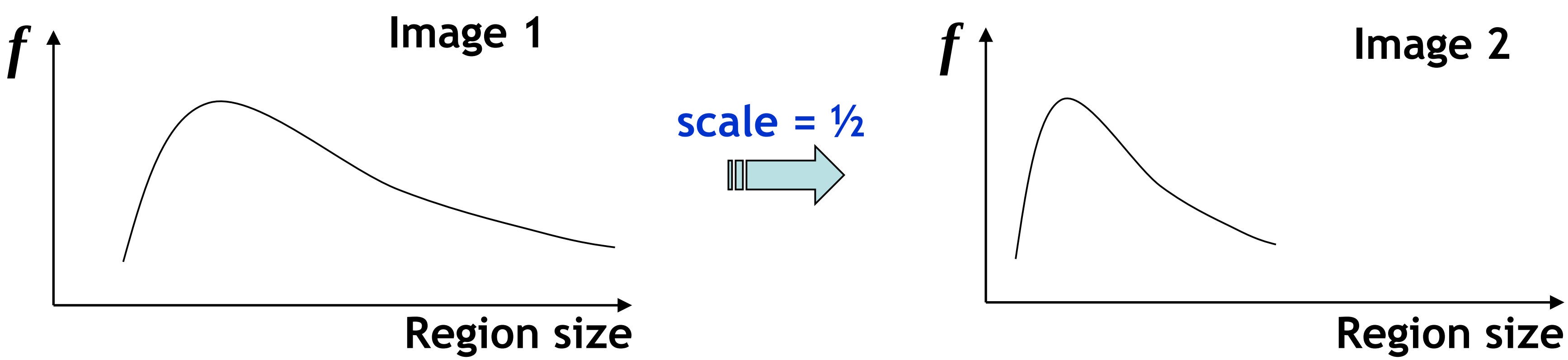


Automatic Scale Selection

- **Solution:**
  - Design a function on the region, which is "scale invariant" (*the same for corresponding regions, even if they are at different scales*)

    Example: average intensity. For corresponding regions (even of different sizes) it will be the same.

  - For a point in one image, we can consider it as a function of region size (patch width)

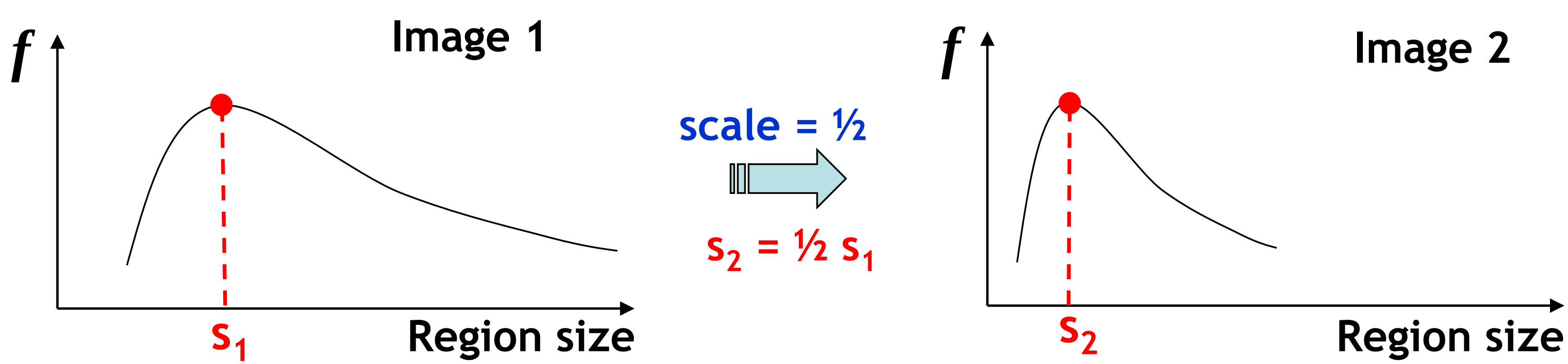


Slide credit: Kristen Grauman

Automatic Scale Selection

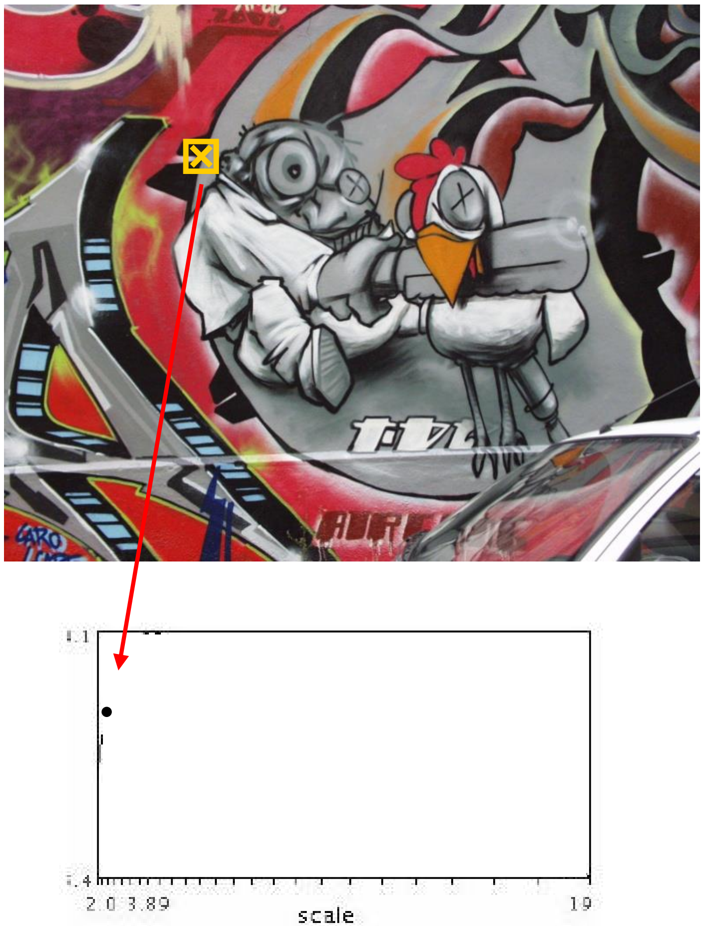
- Common approach:
  - Take a local maximum of this function.
  - Observation: region size for which the maximum is achieved should be *invariant* to image scale.

Important: this scale invariant region size is found in each image independently!



Automatic Scale Selection

- Function responses for increasing scale (scale signature)



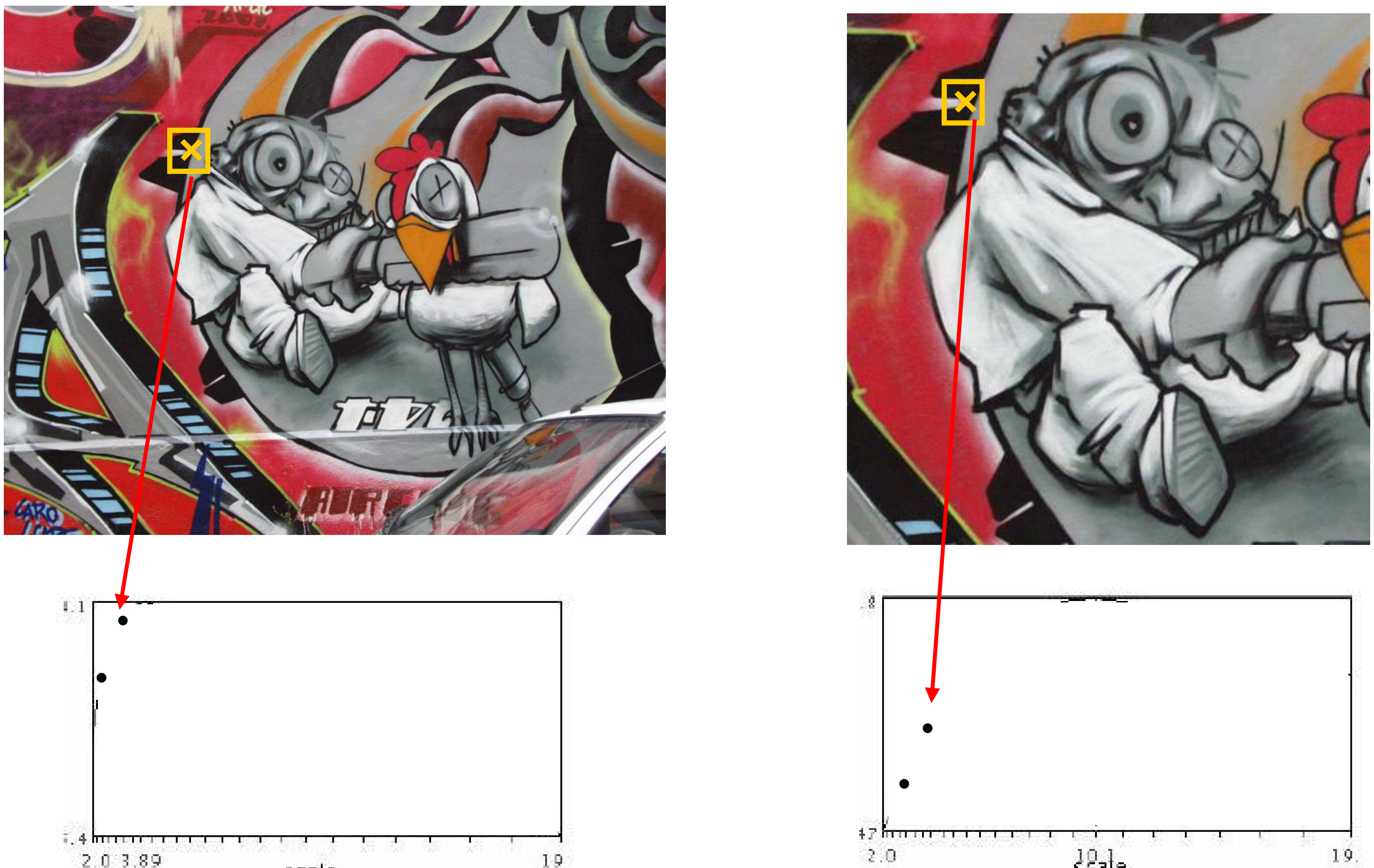
$f(I_{i_1 \ldots i_m}(x, \sigma))$

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

Slide credit: Krystian Mikolajczyk

Automatic Scale Selection

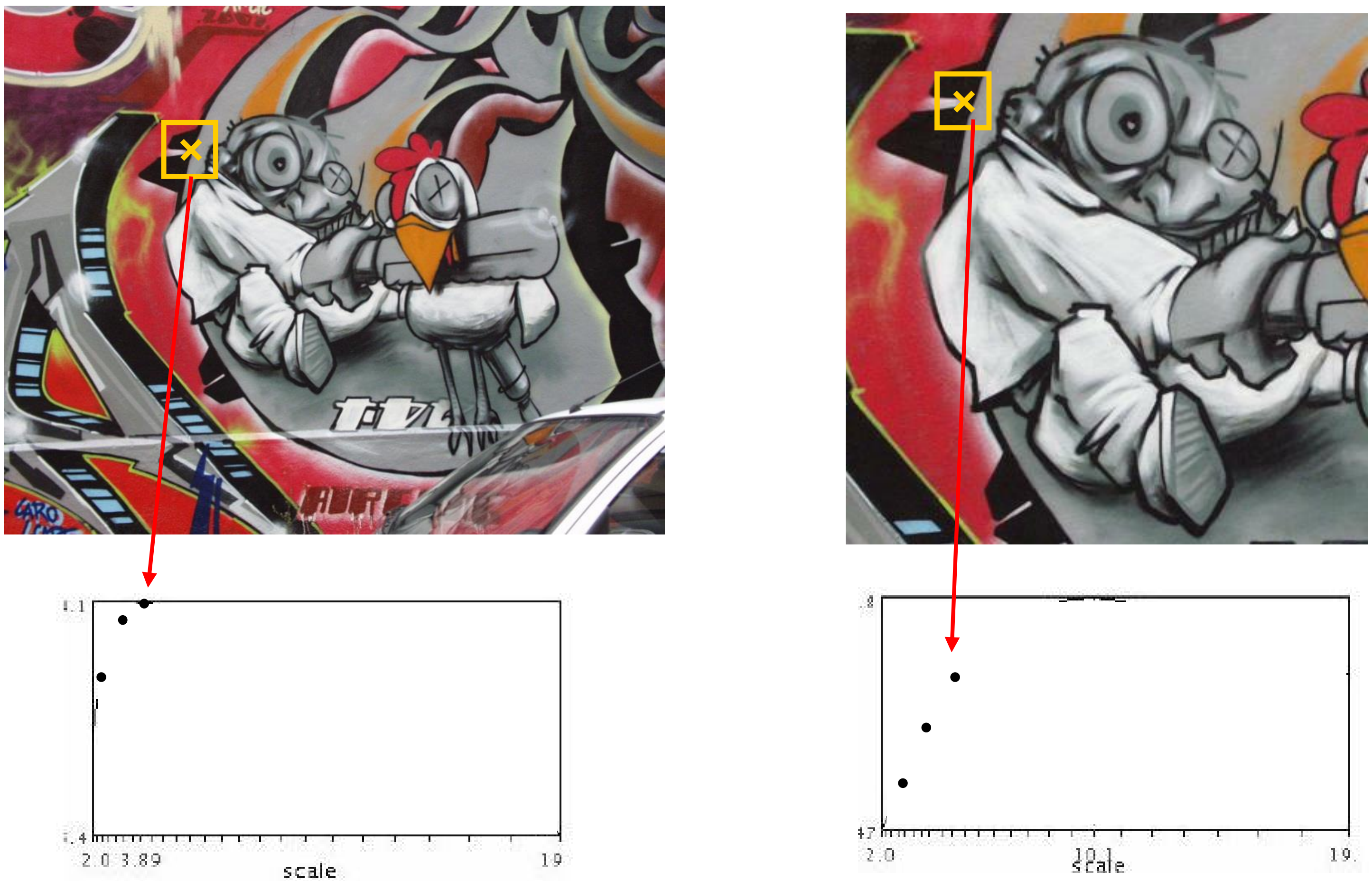
- Function responses for increasing scale (scale signature)



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

Automatic Scale Selection

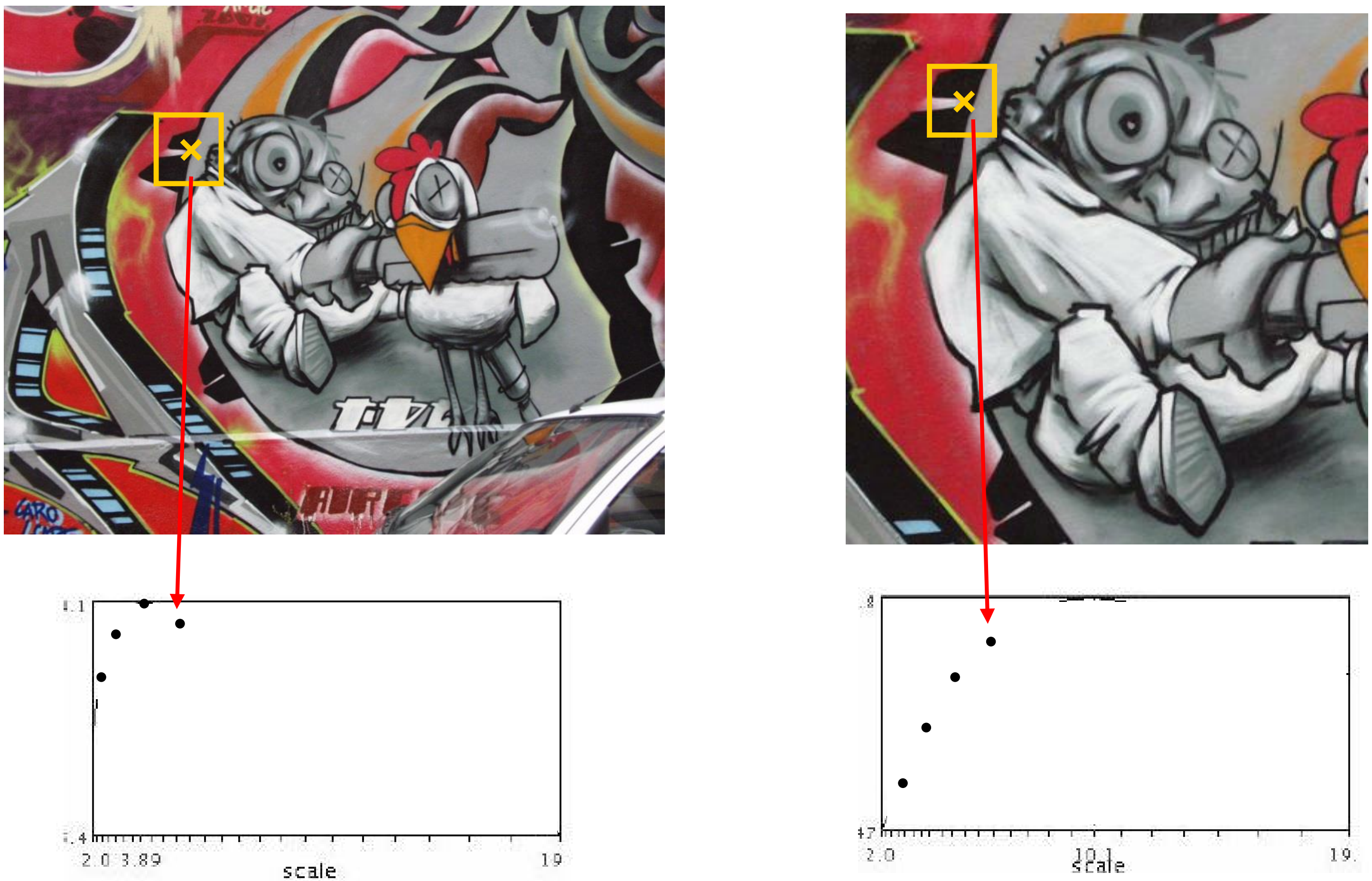
- Function responses for increasing scale (scale signature)



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

Automatic Scale Selection

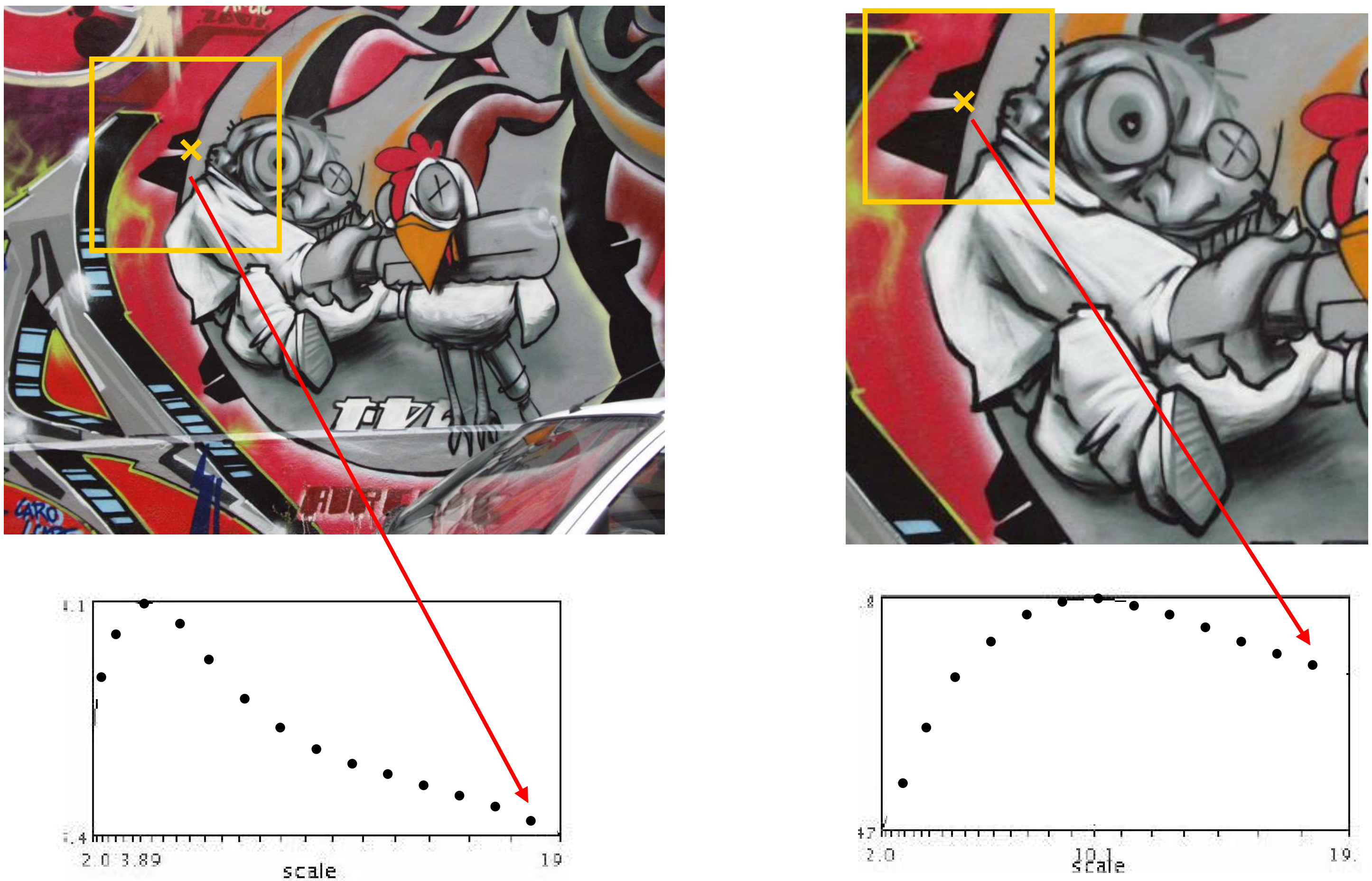
- Function responses for increasing scale (scale signature)



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

Automatic Scale Selection

- **Function responses for increasing scale (scale signature)**

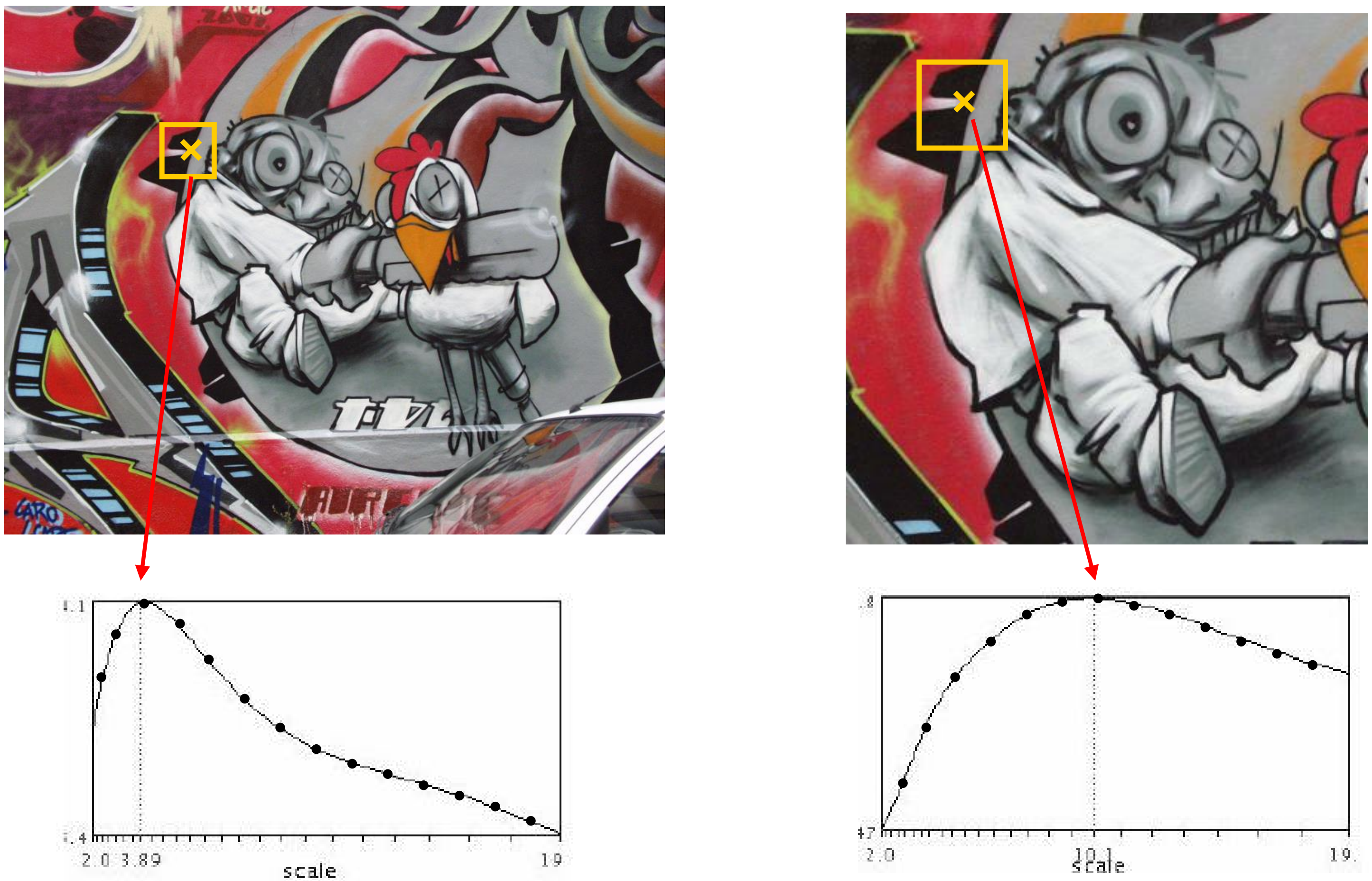


$f(I_{i_1 \dots i_m}(x,\sigma))$ $\quad$ $f(I_{i_1 \dots i_m}(x',\sigma))$

Slide credit: Krystian Mikolajczyk

Automatic Scale Selection

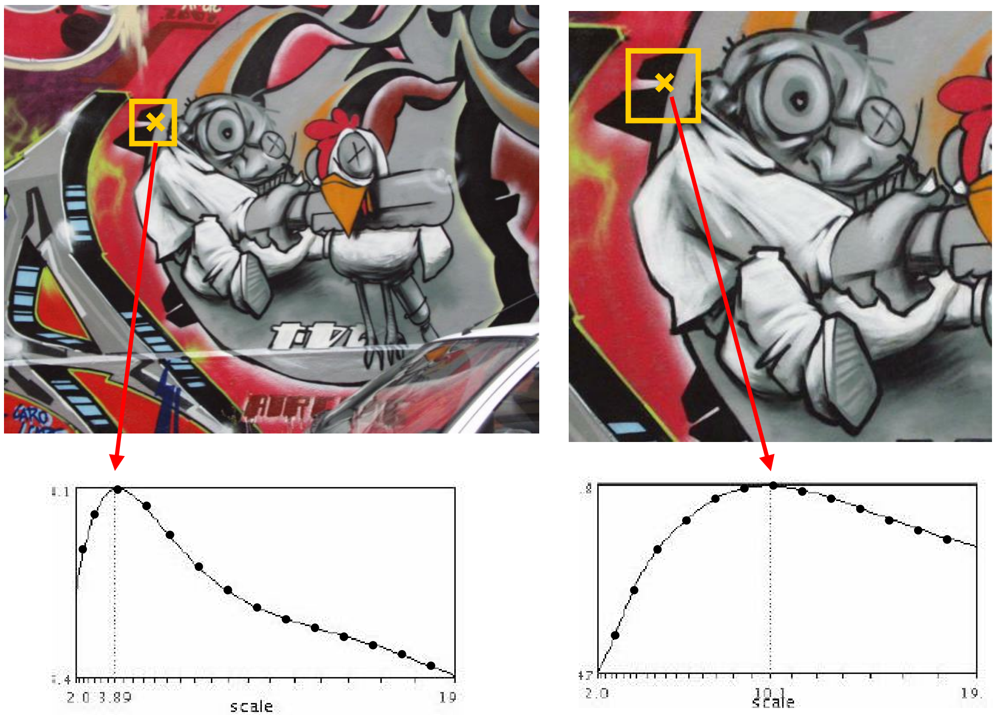
- **Function responses for increasing scale (scale signature)**



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

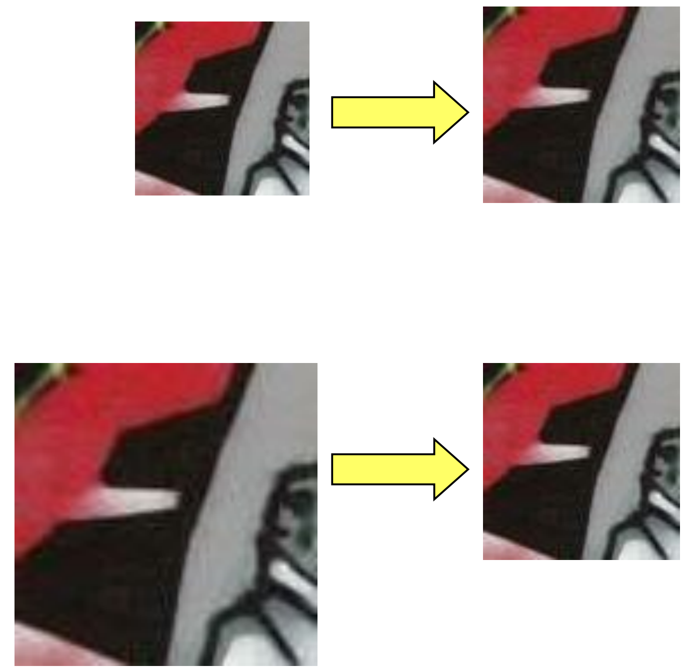
Automatic Scale Selection

- **Normalize: Rescale to fixed size**



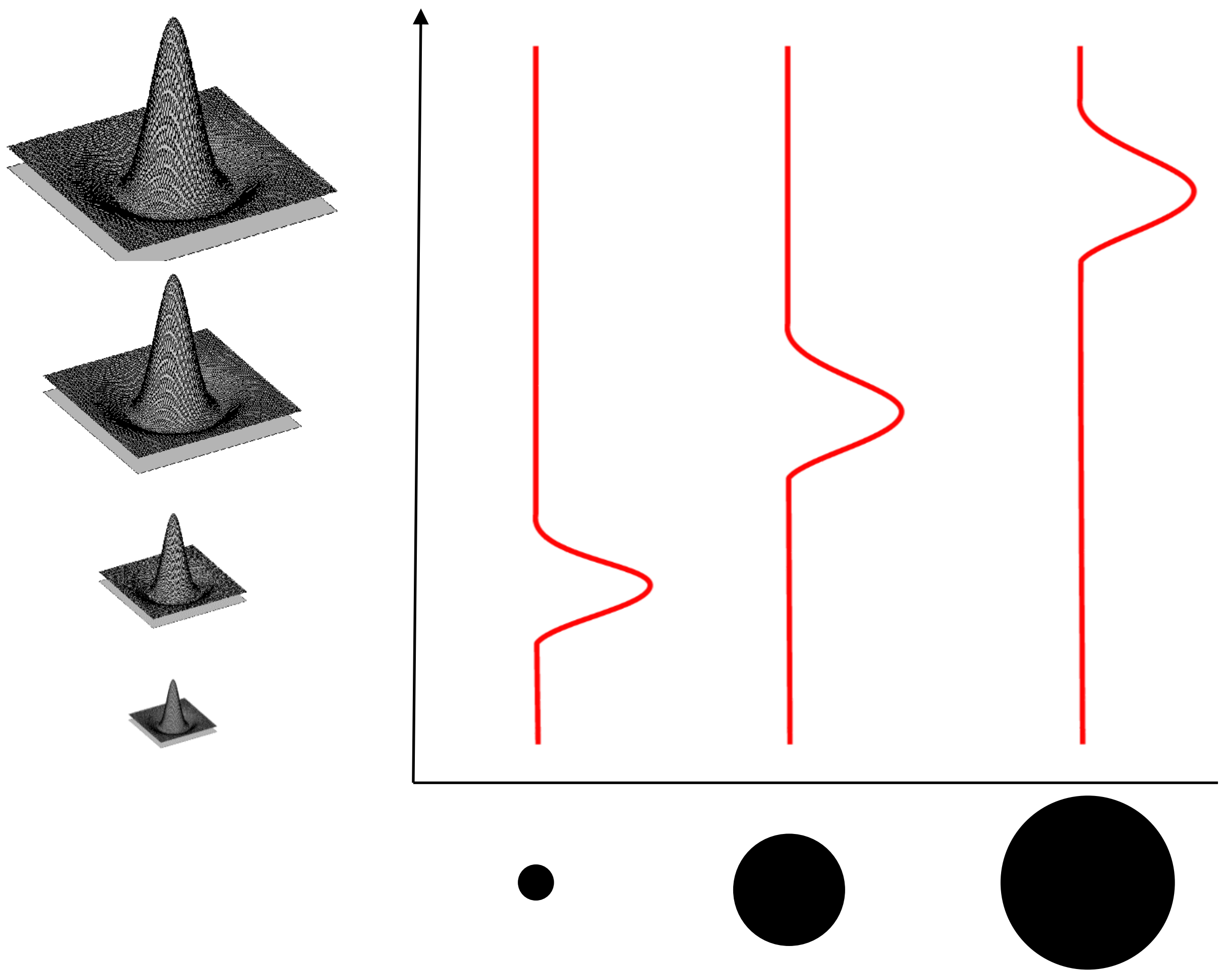
$f(I_{i_1 \ldots i_m}(x, \sigma))$

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



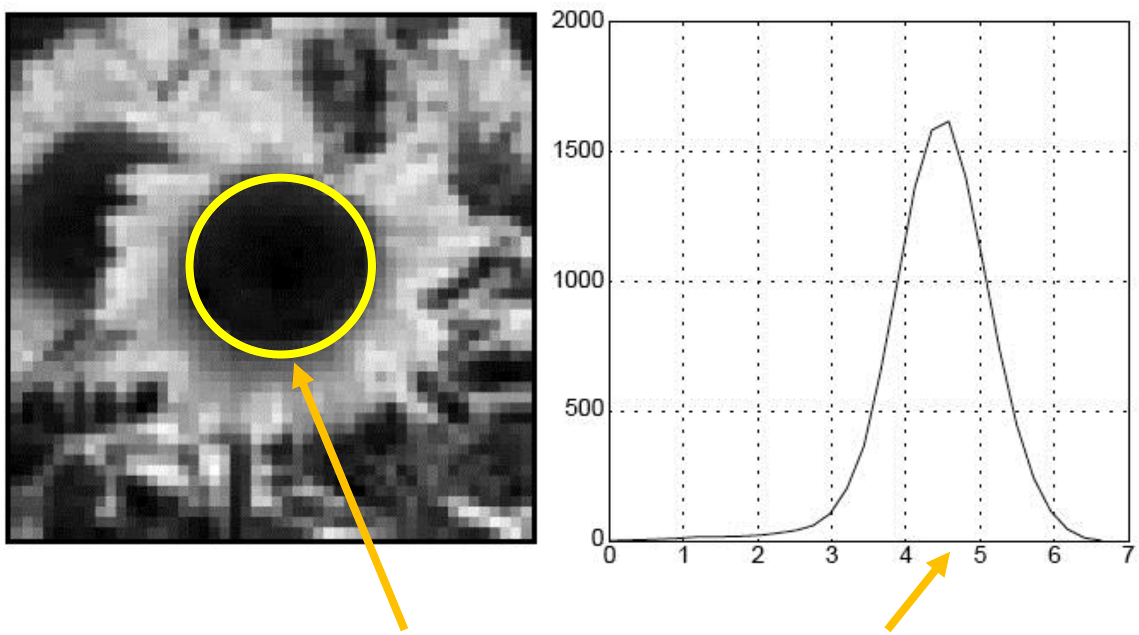
What Is A Useful Signature Function?

- Laplacian-of-Gaussian = “blob” detector



Characteristic Scale

- We define the *characteristic scale* as the scale that produces peak of Laplacian response

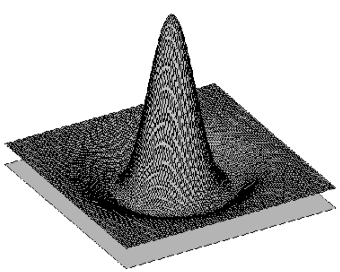


Characteristic scale

T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* 30 (2): pp 77--116.

Laplacian-of-Gaussian (LoG)

- **Interest points:**
  - **Local maxima in scale space of Laplacian-of-Gaussian**



$$L_{xx}(\sigma) + L_{yy}(\sigma)$$

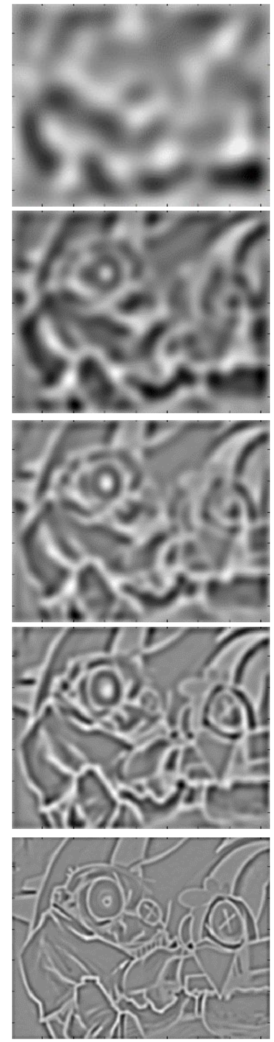
σ^5

σ^4

σ^3

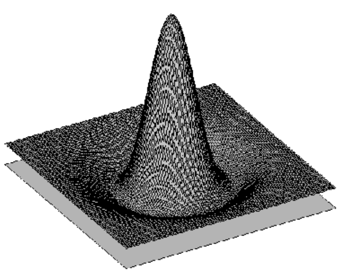
σ^2

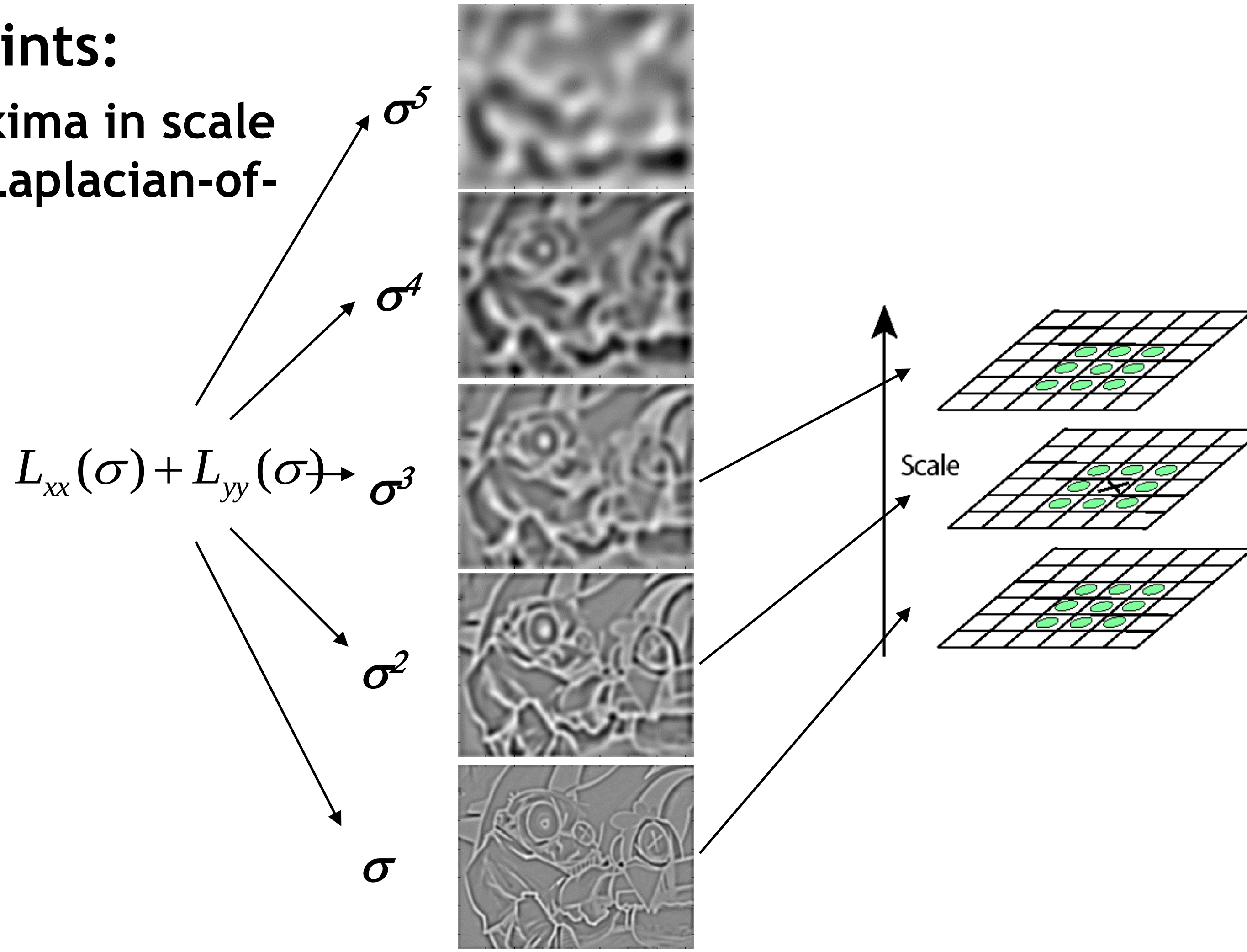
σ



Laplacian-of-Gaussian (LoG)

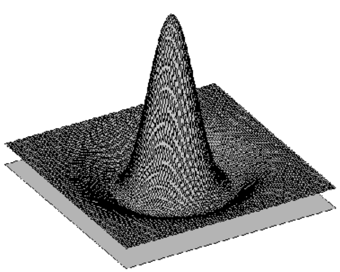
- **Interest points:**
  - Local maxima in scale space of Laplacian-of-Gaussian



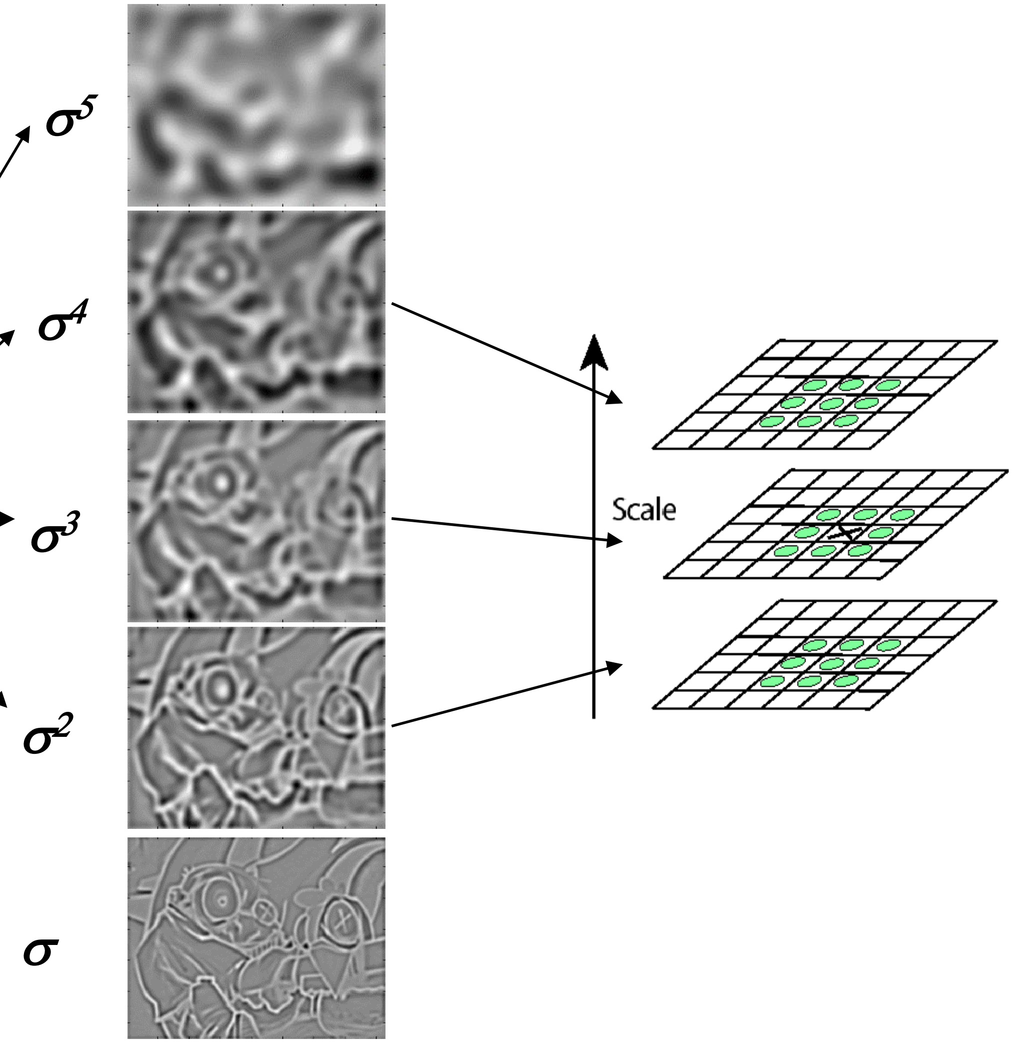


Laplacian-of-Gaussian (LoG)

- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian

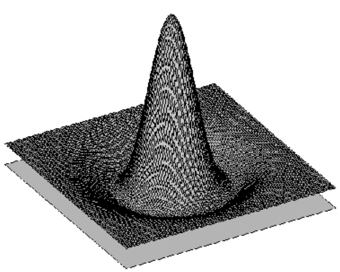


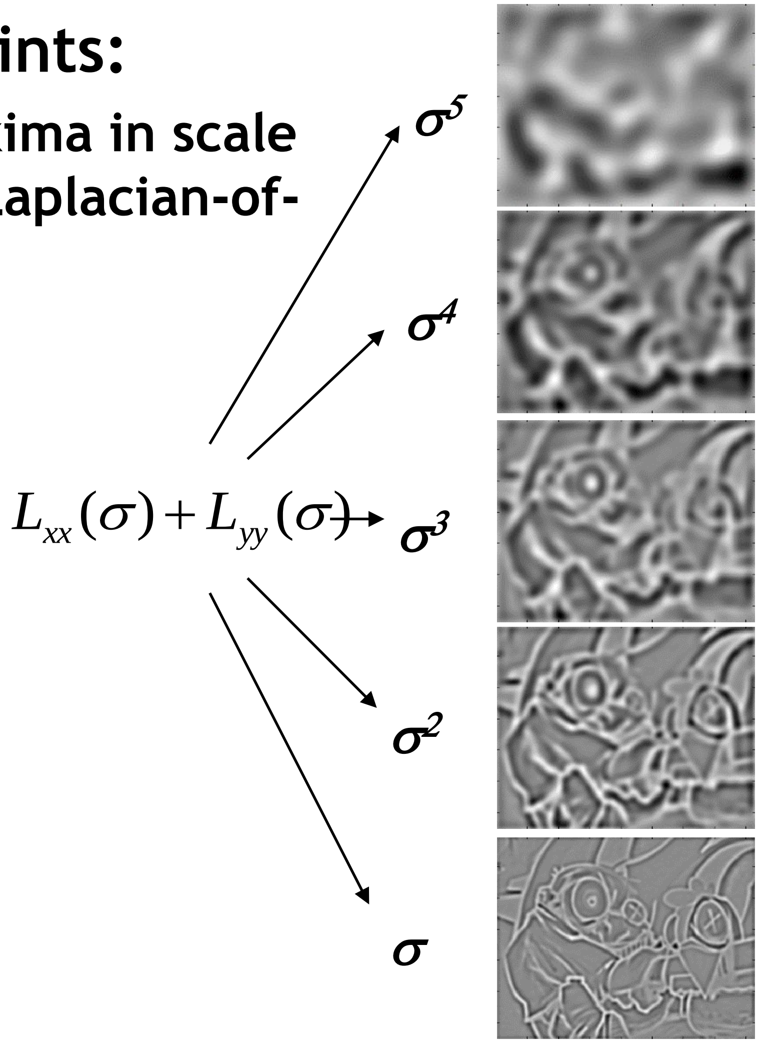
$$L_{xx}(\sigma) + L_{yy}(\sigma)$$

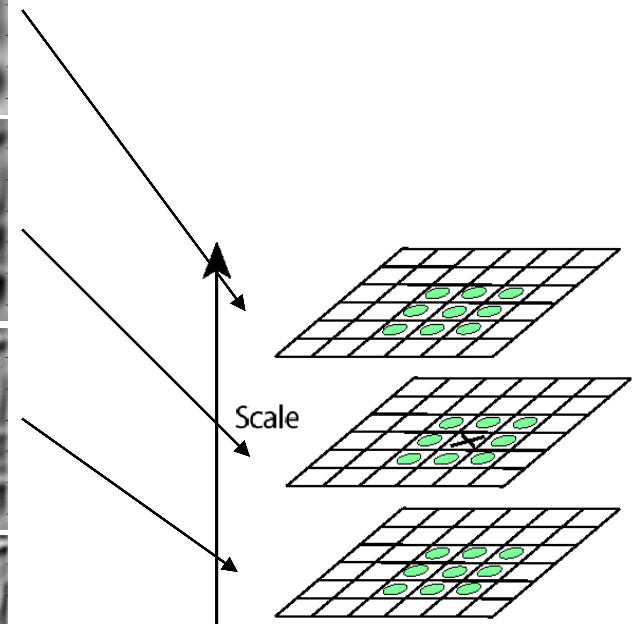


Laplacian-of-Gaussian (LoG)

- **Interest points:**
  - Local maxima in scale space of Laplacian-of-Gaussian







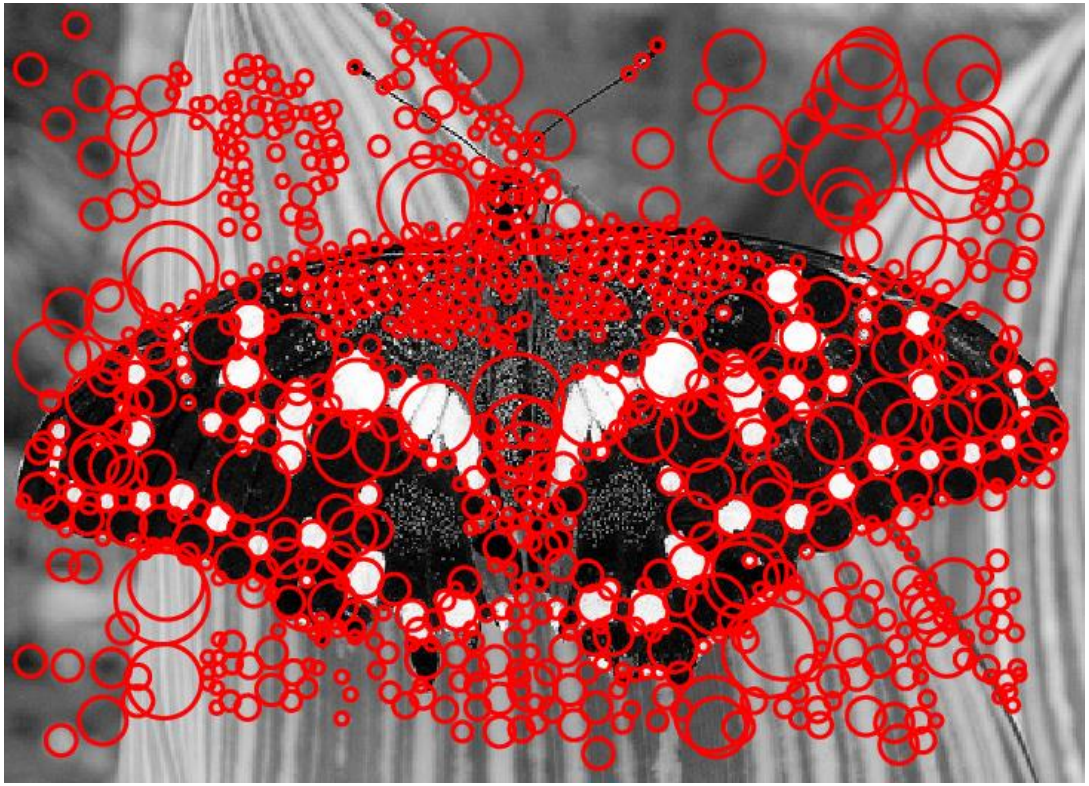
$\Rightarrow$ **List of** (x, y, σ)

LoG Detector: Workflow

LoG Detector: Workflow

sigma = 11.9912

LoG Detector: Workflow



Difference-of-Gaussian (DoG)

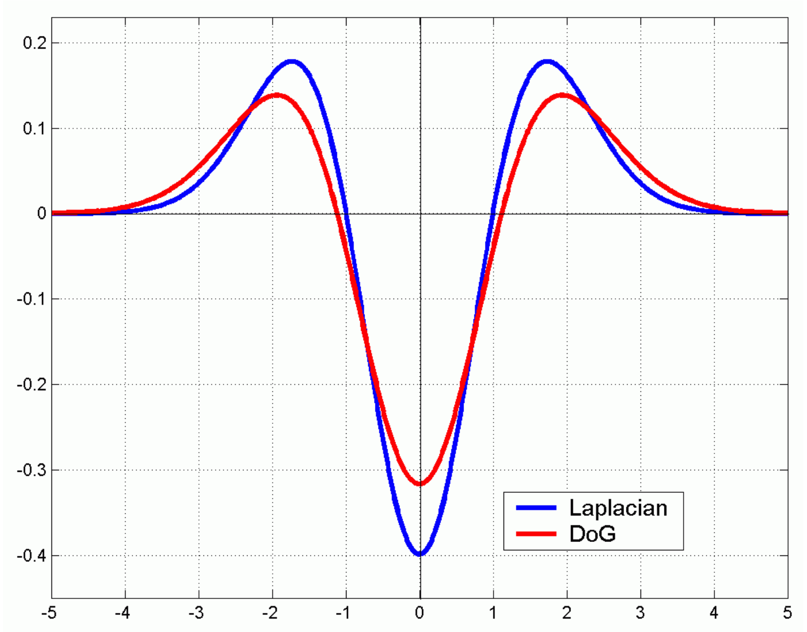
- We can efficiently approximate the Laplacian with a difference of Gaussians:

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

(Laplacian)

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

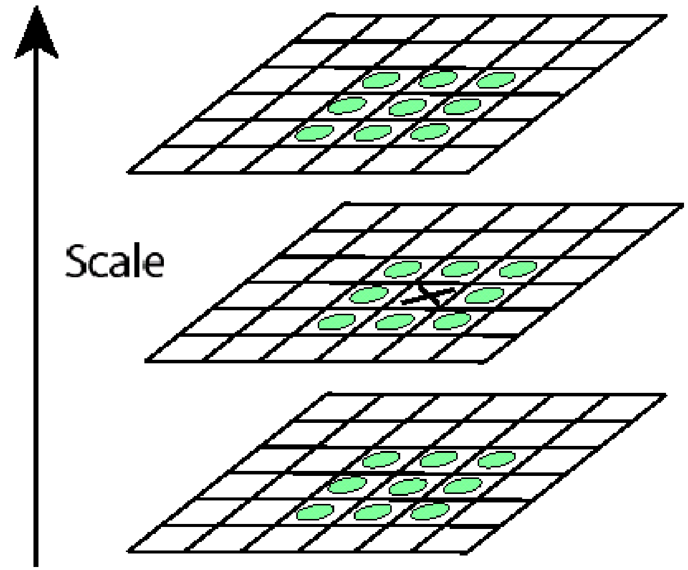
(Difference of Gaussians)



- Advantages?
  - No need to compute 2nd derivatives.
  - Gaussians are computed anyway, e.g. in a Gaussian pyramid.

Key point localization with DoG

- Detect maxima of difference-of-Gaussian (DoG) in scale space
- Then reject points with low contrast (threshold)
- Eliminate edge responses



Candidate keypoints:
list of (x,y,σ)

Slide credit: David Lowe

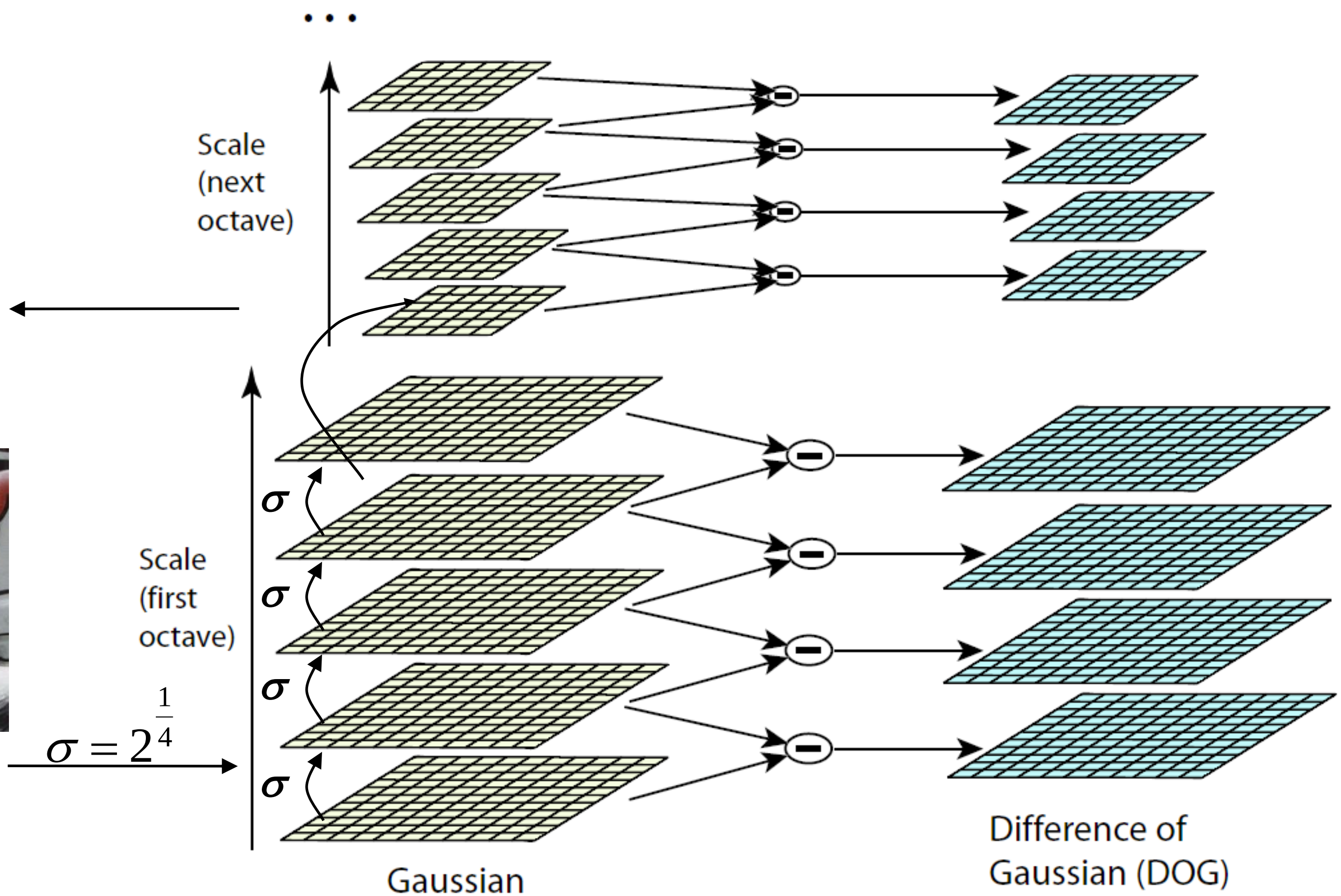
DoG - Efficient Computation

- **Computation in Gaussian scale pyramid**

Sampling with step $\sigma^4 = 2$

Original image

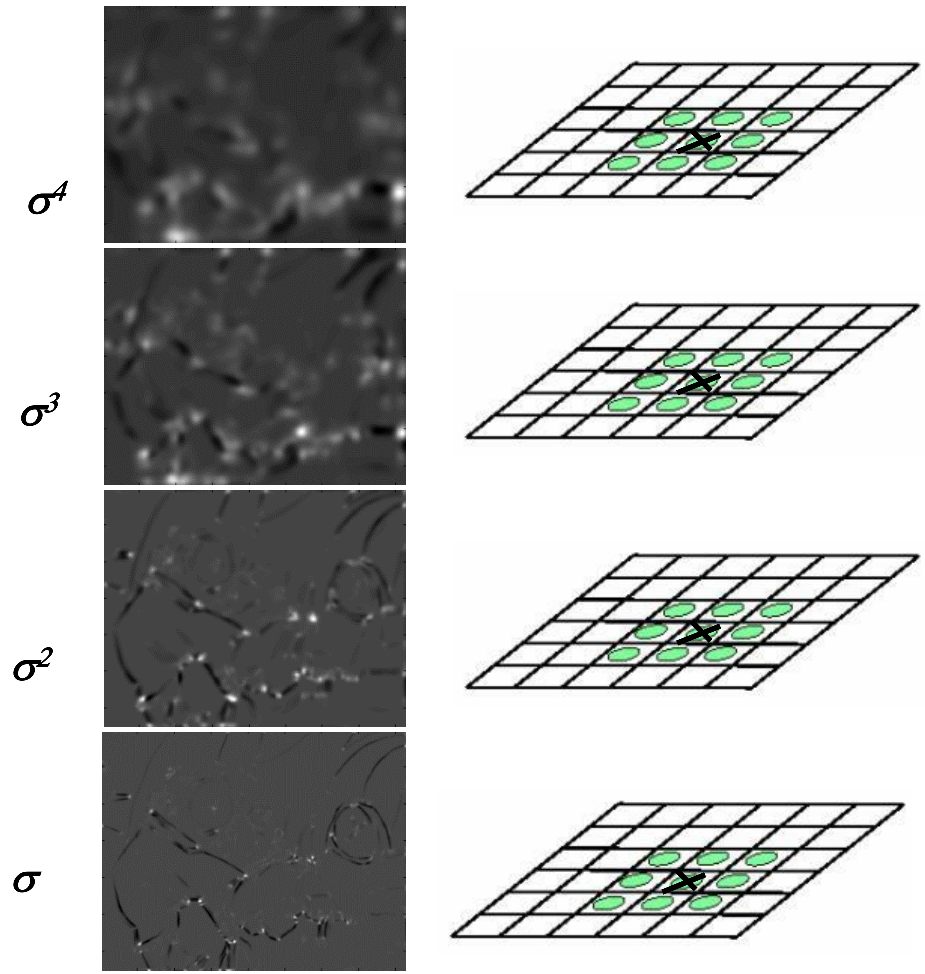
$\sigma = 2^{\frac{1}{4}}$



Results: Lowe's DoG

Harris-Laplace [Mikolajczyk '01]

1. **Initialization: Multiscale Harris corner detection**

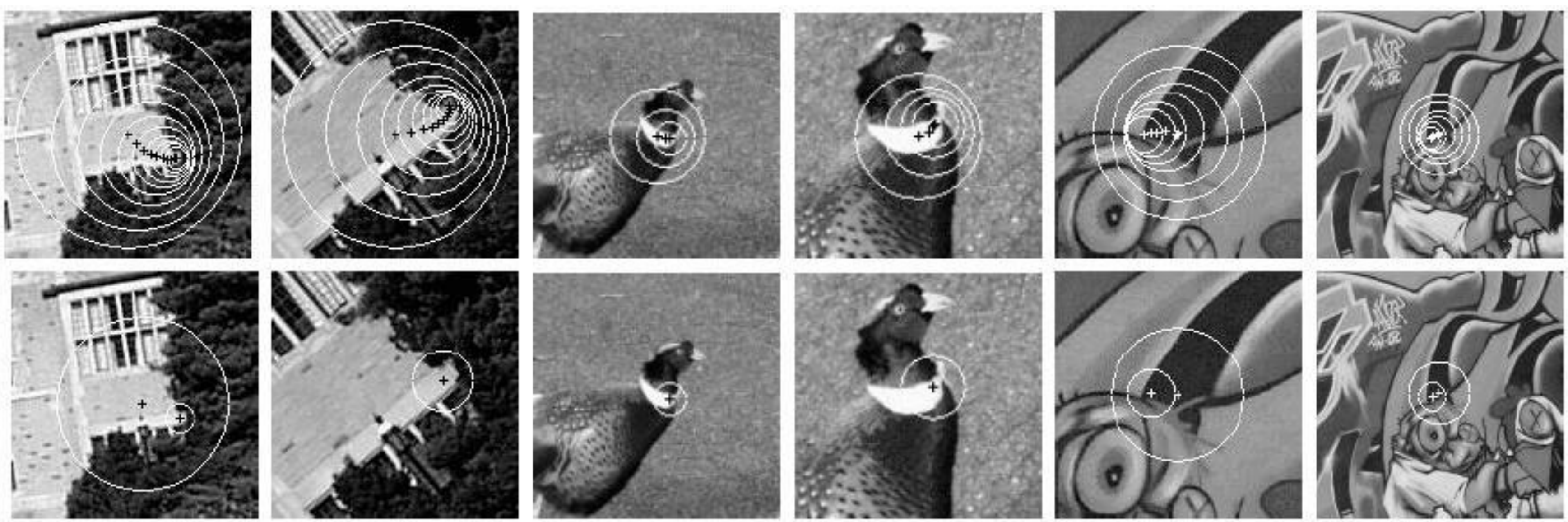


Computing Harris function **Detecting local maxima**

Harris-Laplace [Mikolajczyk '01]

1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian
(same procedure with Hessian ⇒ Hessian-Laplace)

Harris points



Harris-Laplace points

Summary: Scale Invariant Detection

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

- Two strategies
  - Laplacian-of-Gaussian (LoG)
  - Difference-of-Gaussian (DoG) as a fast approximation

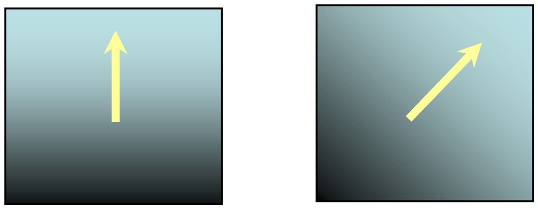
  - *These can be used either on their own, or in combinations with single-scale keypoint detectors (Harris, Hessian).*

Topics of This Lecture

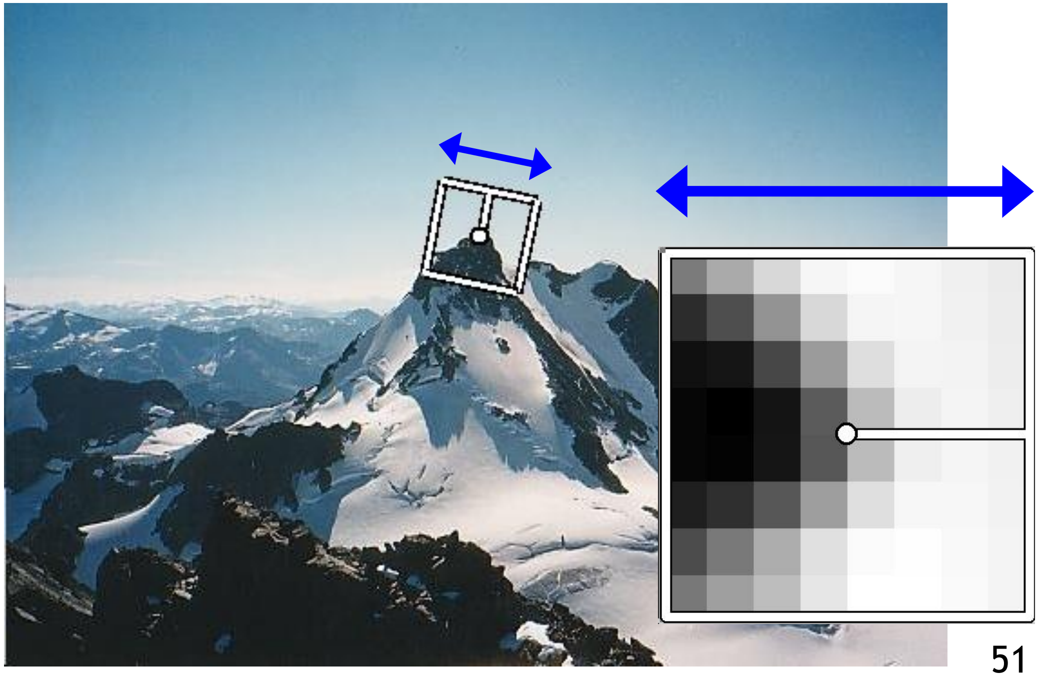
- Local Feature Extraction (cont'd)
  - Scale Invariant Region Selection
  - Orientation normalization
  - Affine Invariant Feature Extraction
- Local Descriptors
  - SIFT
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

Rotation Invariant Descriptors

- Find local orientation
  - Dominant direction of gradient for the image patch



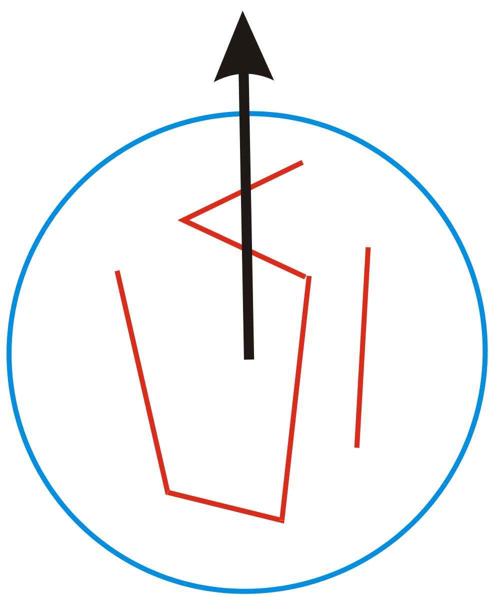
- Rotate patch according to this angle
  - This puts the patches into a canonical orientation.

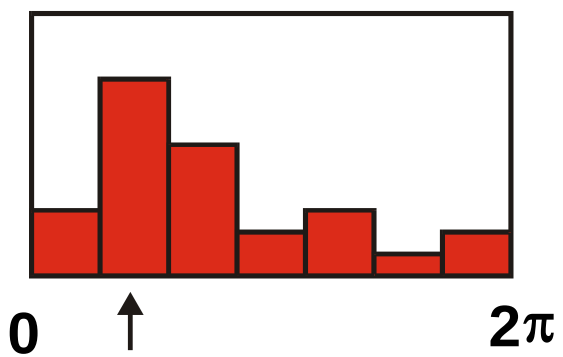


Slide credit: Svetlana Lazebnik, Matthew Brown

Orientation Normalization: Computation

- Compute orientation histogram [Lowe, SIFT, 1999]
- Select dominant orientation
- Normalize: rotate to fixed orientation



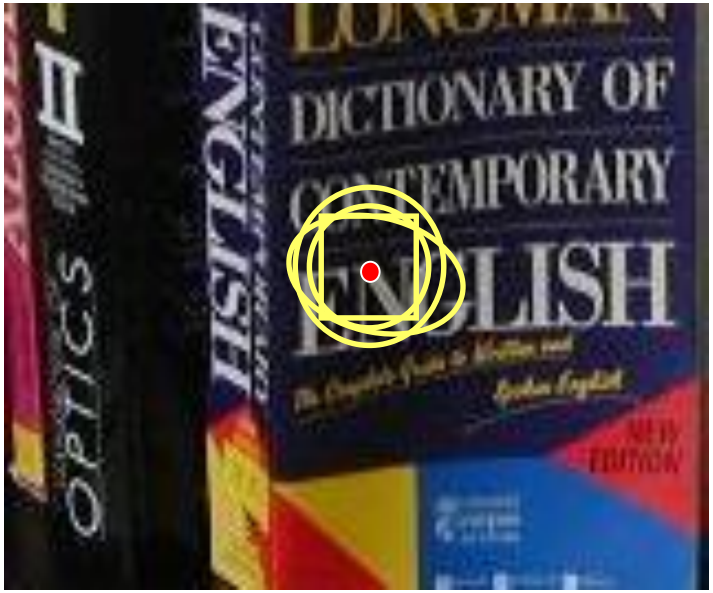


Slide adapted from David Lowe

Topics of This Lecture

- **Local Feature Extraction (cont'd)**
  - Scale Invariant Region Selection
  - Orientation normalization
  - Affine Invariant Feature Extraction
- Local Descriptors
  - SIFT
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

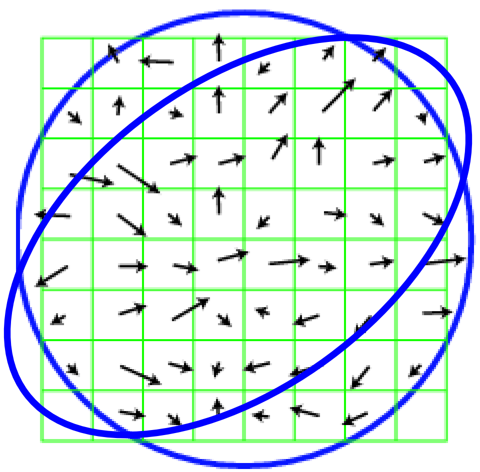
The Need for Invariance



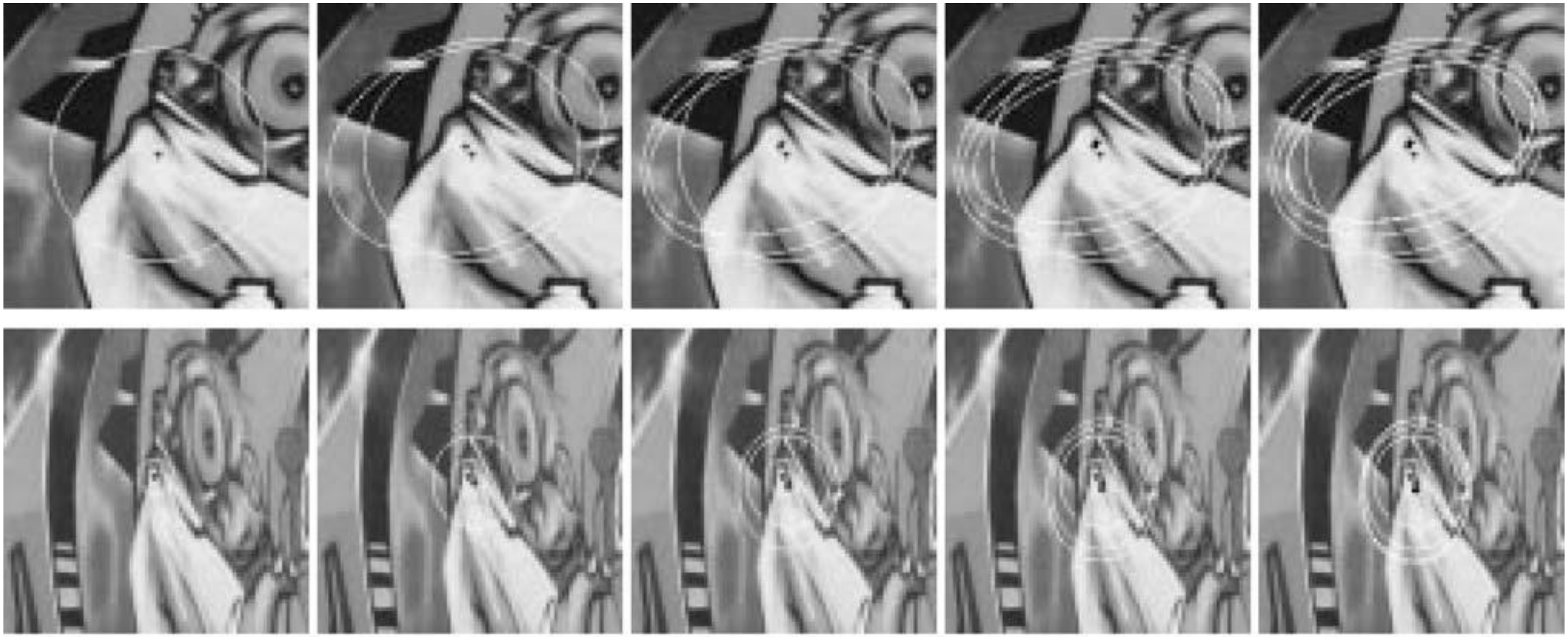
- Up to now, we had invariance to
  - Translation
  - Scale
  - Rotation
- Not sufficient to match regions under viewpoint changes
  - For this, we need also affine adaptation

Affine Adaptation

- **Problem:**
  - Determine the characteristic shape of the region.
  - Assumption: shape can be described by "local affine frame".
- **Solution: iterative approach**
  - Use a circular window to compute second moment matrix.
  - Compute eigenvectors to adapt the circle to an ellipse.
  - Recompute second moment matrix using new window and iterate…



Iterative Affine Adaptation



1. Detect keypoints, e.g. multi-scale Harris
2. Automatically select the scales
3. Adapt affine shape based on second order moment matrix
4. Refine point location

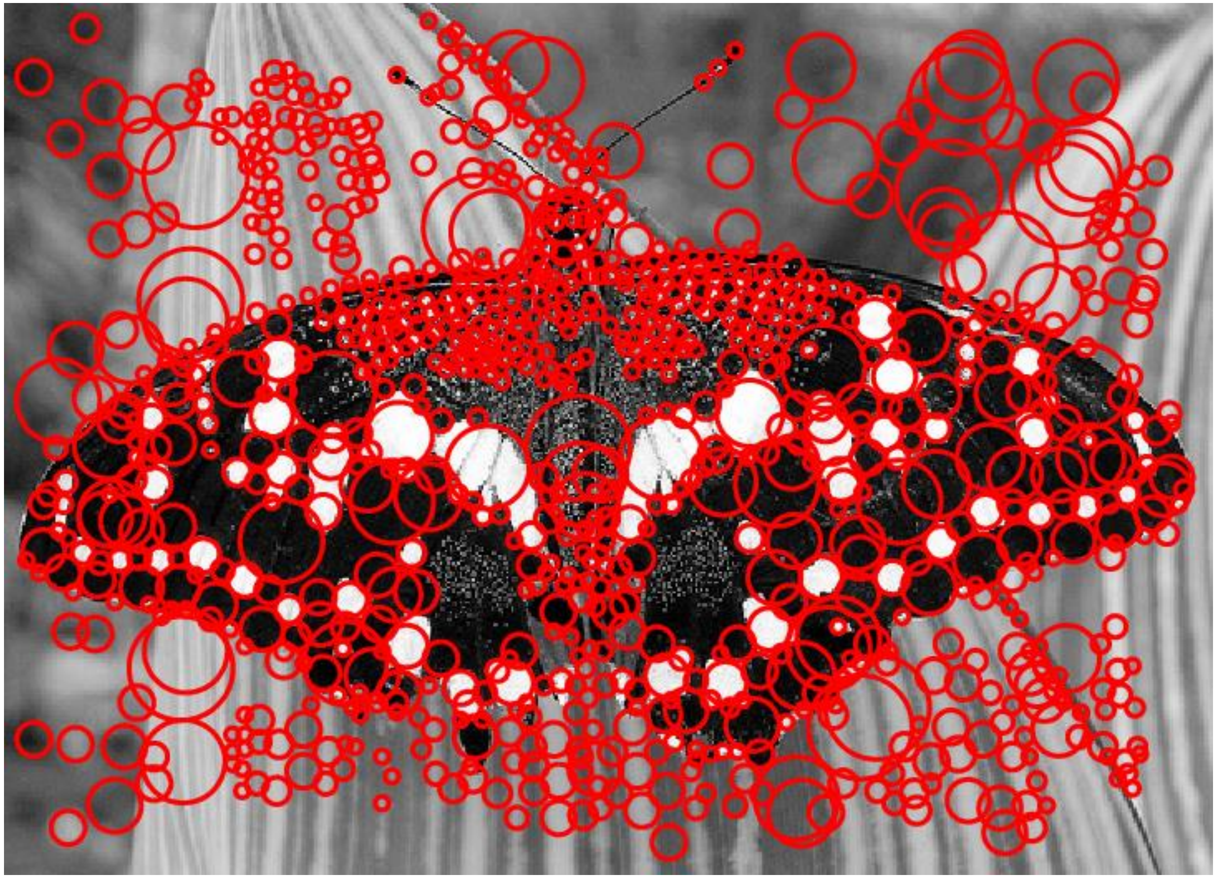
K. Mikolajczyk and C. Schmid, Scale and affine invariant interest point detectors, IJCV 60(1):63-86, 2004.

Slide credit: Tinne Tuytelaars

Affine Normalization/Deskewing

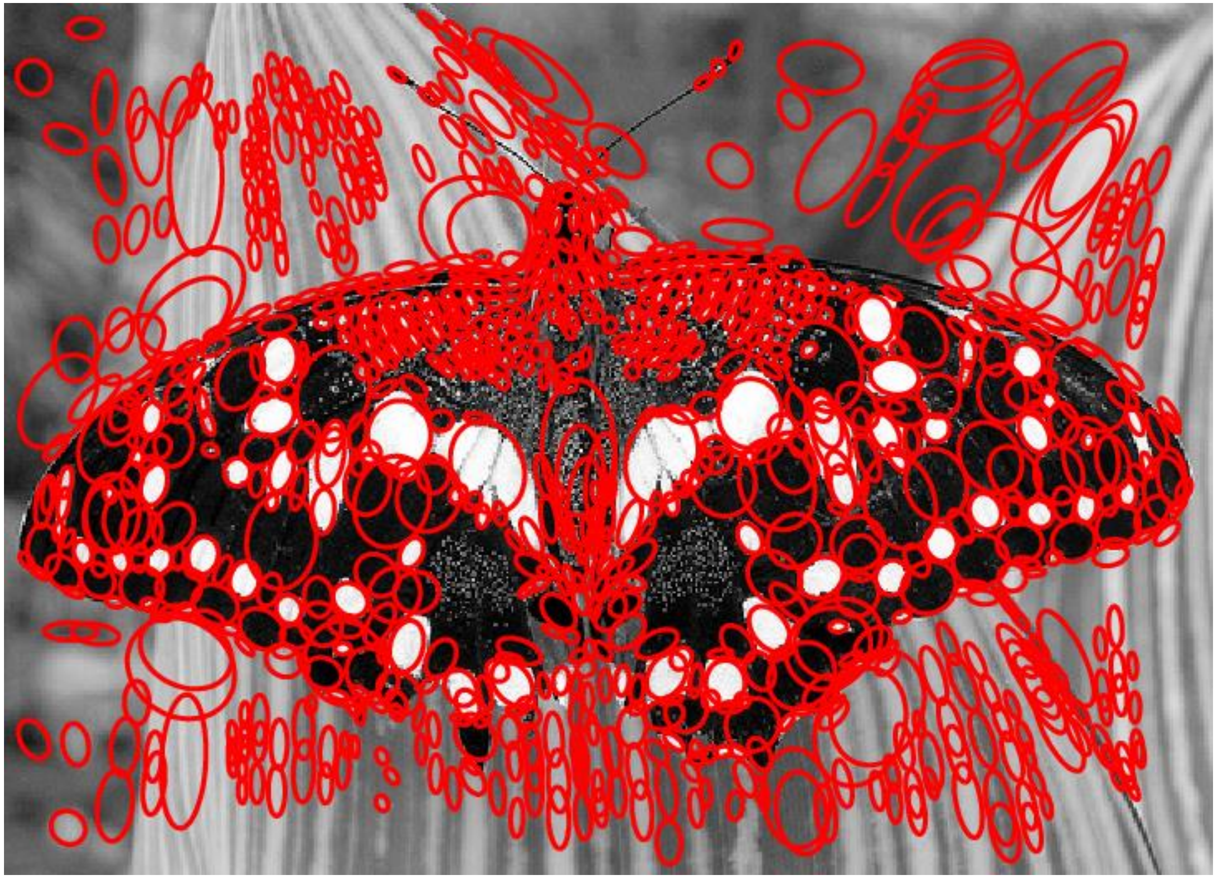
- **Steps**
  - Rotate the ellipse's main axis to horizontal
  - Scale the x axis, such that it forms a circle

Affine Adaptation Example



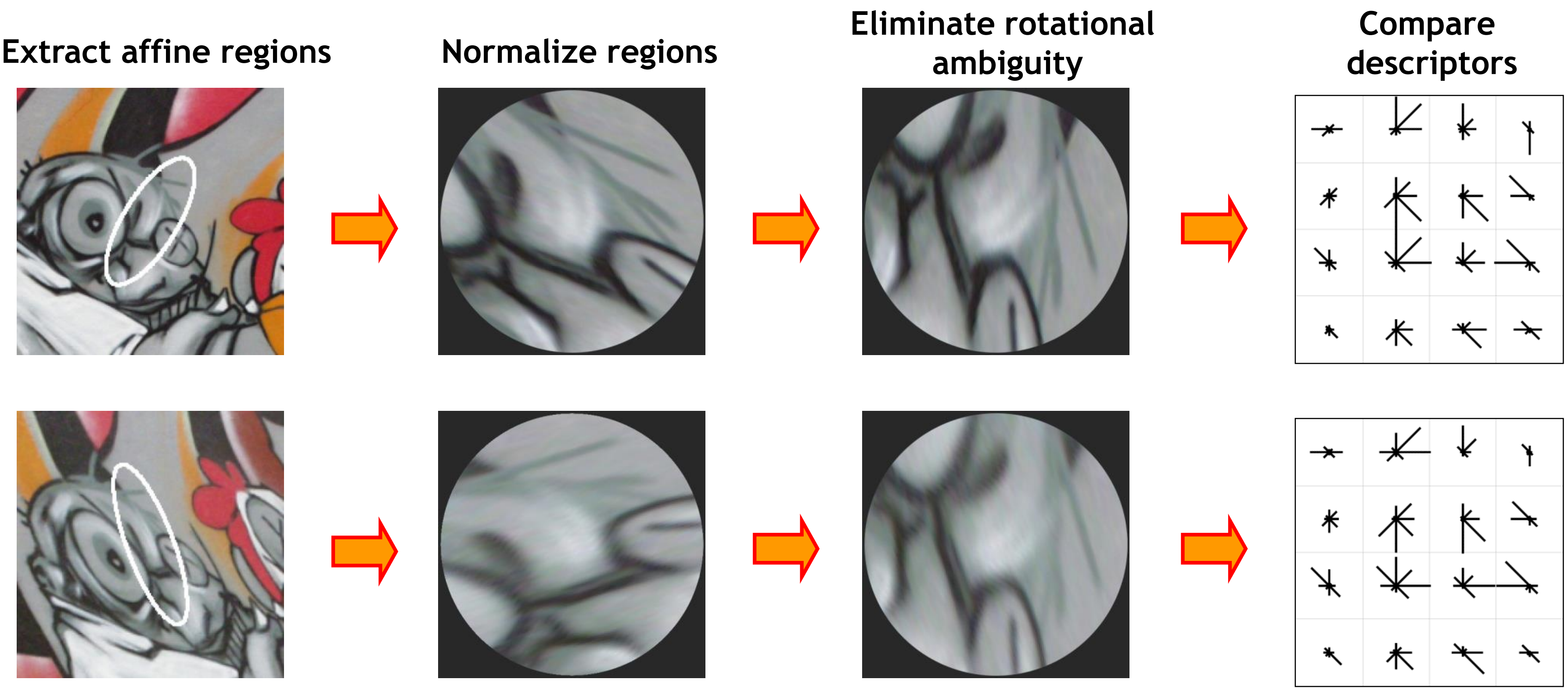
Scale-invariant regions (blobs)

Affine Adaptation Example



Affine-adapted blobs

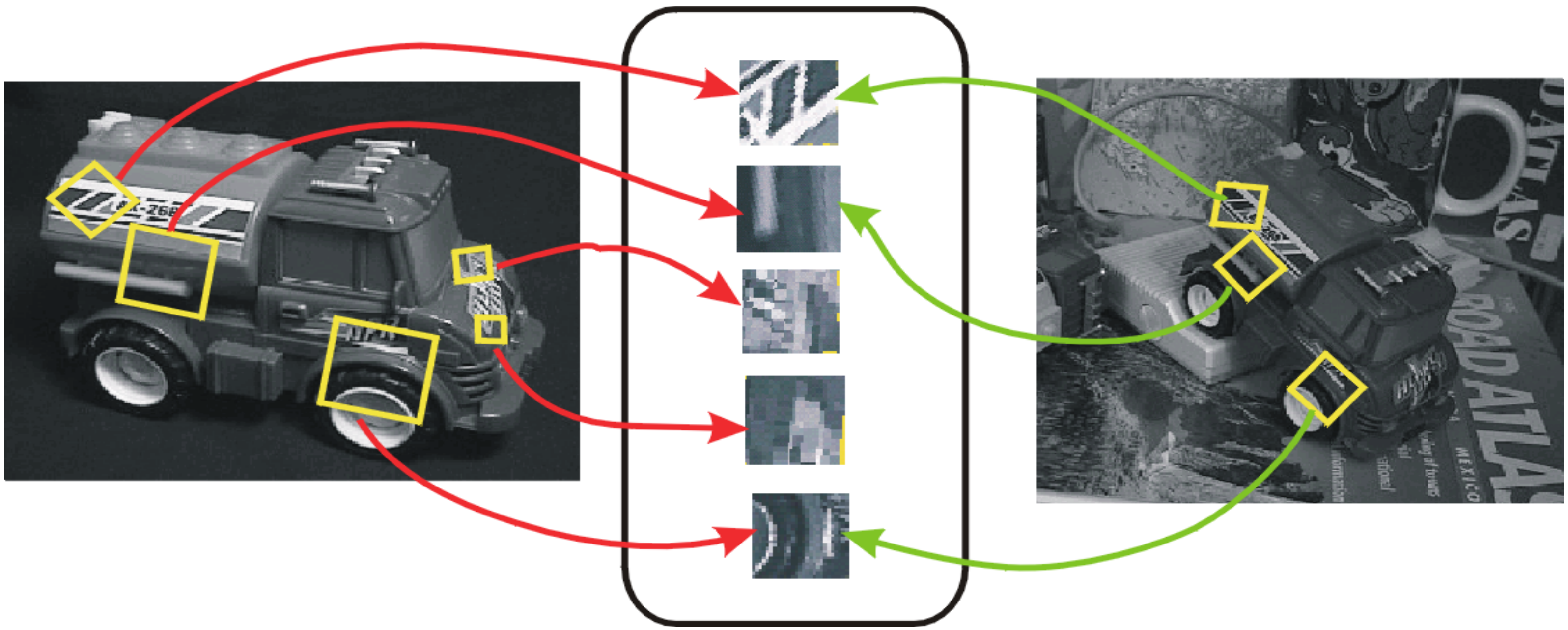
Summary: Affine-Inv. Feature Extraction



Slide credit: Svetlana Lazebnik

Invariance vs. Covariance

- **Invariance:**
  - features(transform(image)) = features(image)
- **Covariance:**
  - features(transform(image)) = transform(features(image))



Covariant detection ⇒ invariant description

Slide credit: Svetlana Lazebnik, David Lowe

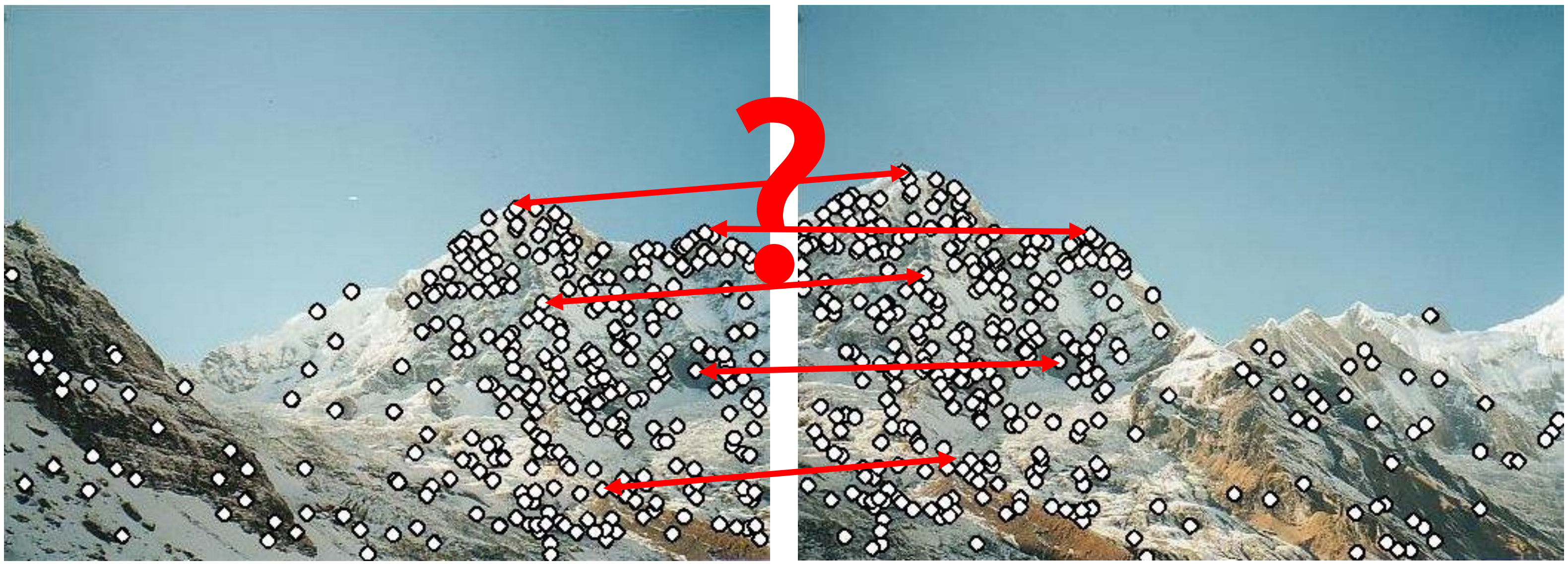
Topics of This Lecture

- Local Feature Extraction (cont'd)
  - Orientation normalization
  - Affine Invariant Feature Extraction
- Local Descriptors
  - SIFT
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

Local Descriptors

- We know how to detect points
- Next question:

How to *describe* them for matching?



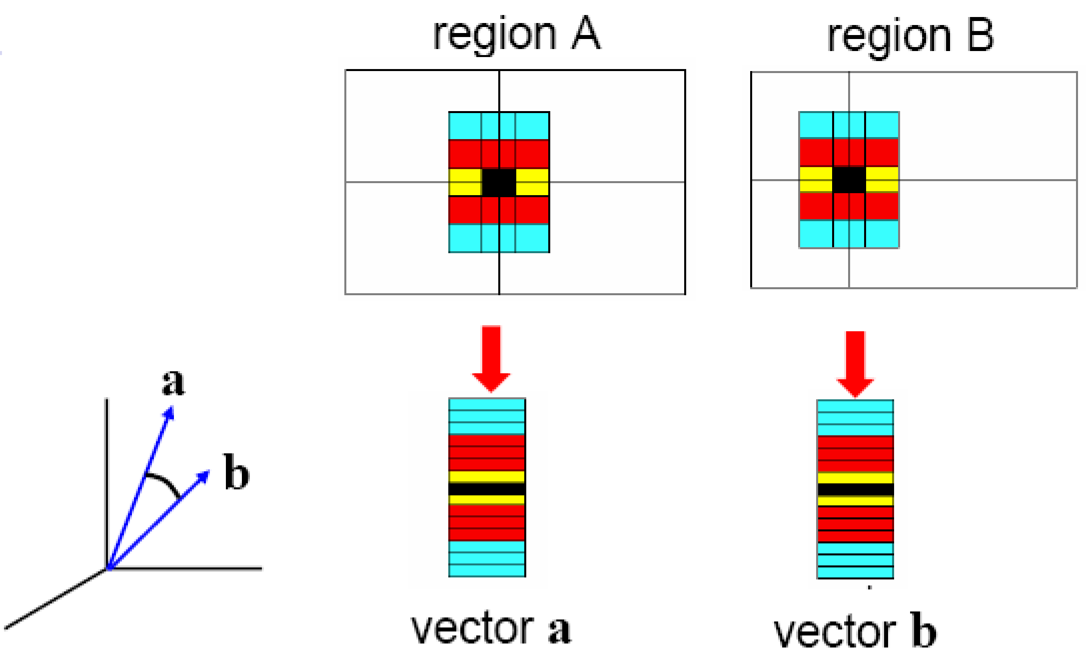
Point descriptor should be:
1. Invariant
2. Distinctive

Local Descriptors

- **Simplest descriptor: list of intensities within a patch.**
- **What is this going to be invariant to?**

Write regions as vectors

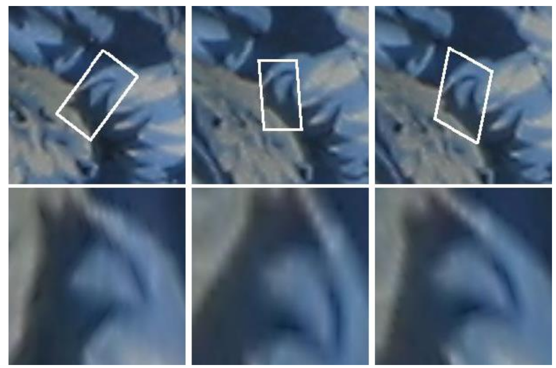
$$A \rightarrow \mathbf{a}, \quad B \rightarrow \mathbf{b}$$



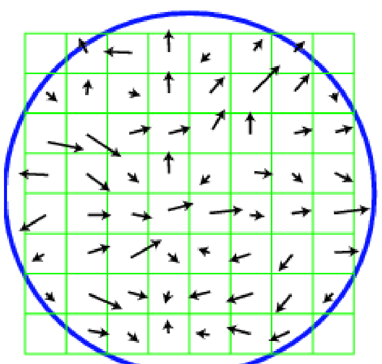
Slide credit: Kristen Grauman

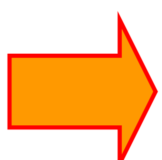
Feature Descriptors

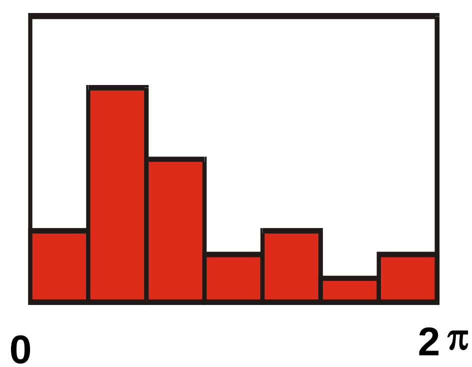
- **Disadvantage of patches as descriptors:**
  - Small shifts can affect matching score a lot

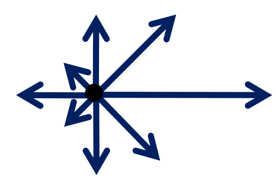


- **Solution: histograms**



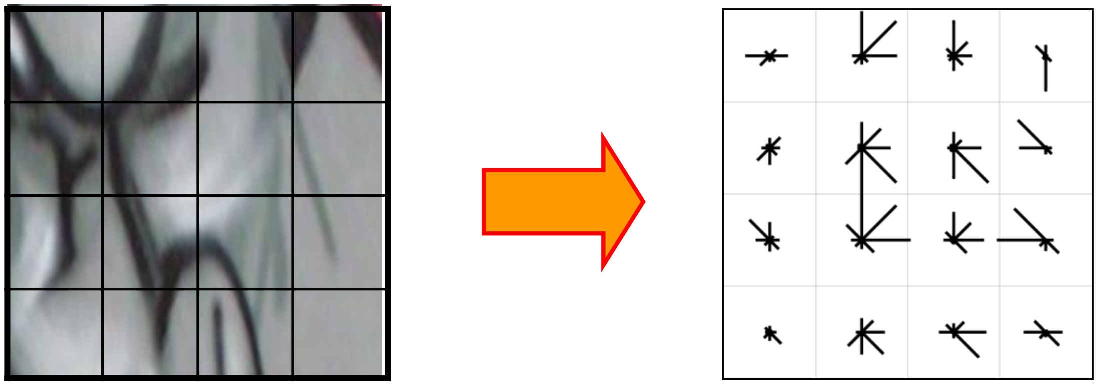






Feature Descriptors: SIFT

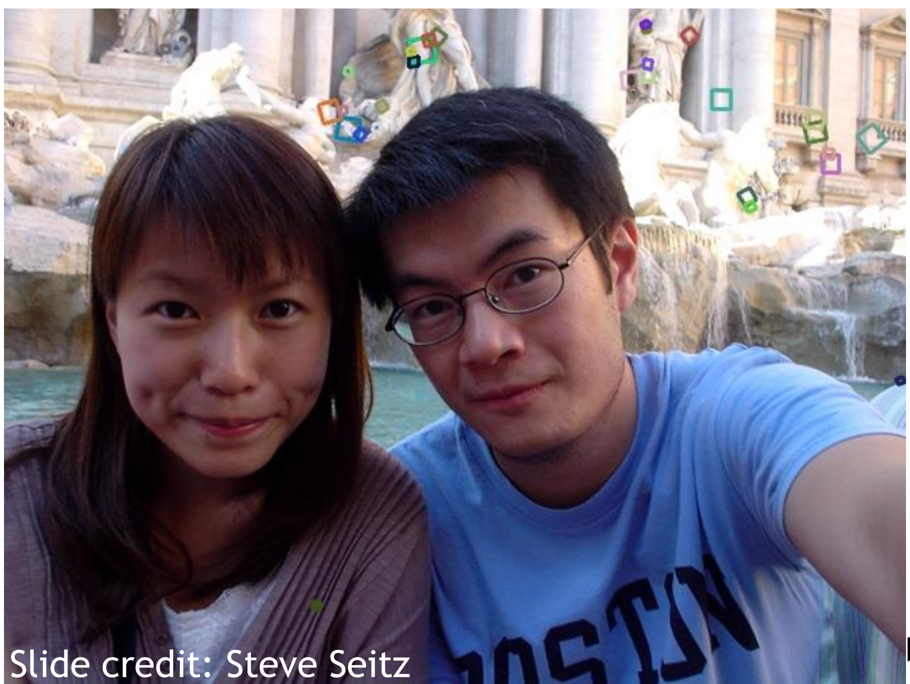
- **Scale Invariant Feature Transform**
- **Descriptor computation:**
  - Divide patch into 4x4 sub-patches: 16 cells
  - Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions

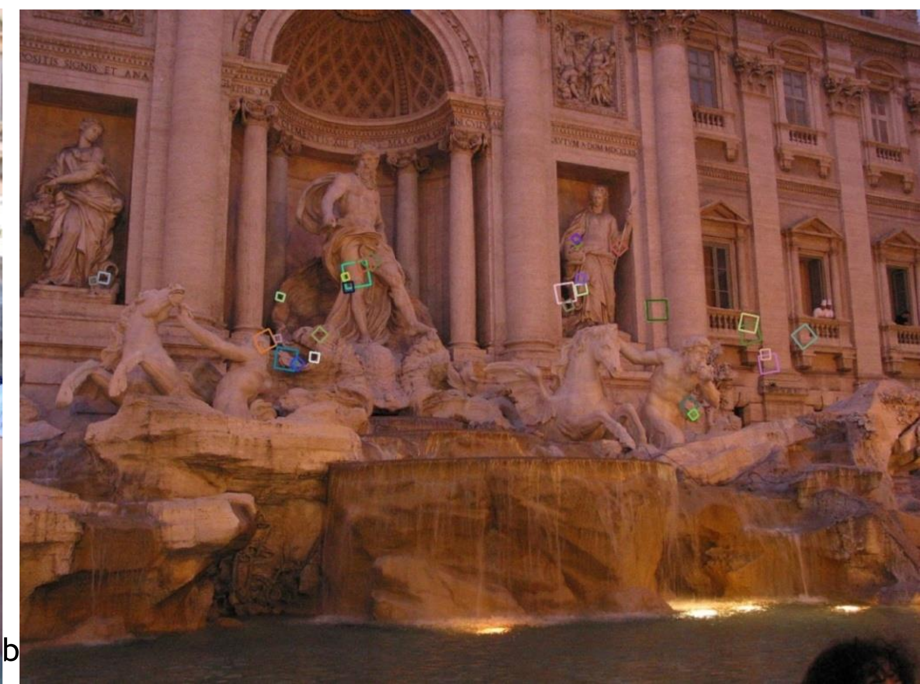


David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Overview: SIFT

- Extraordinarily robust matching technique
  - Can handle changes in viewpoint up to ~60 deg. out-of-plane rotation
  - Can handle significant changes in illumination
    - Sometimes even day vs. night (below)
  - Fast and efficient—can run in real time
  - Lots of code available
    - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT





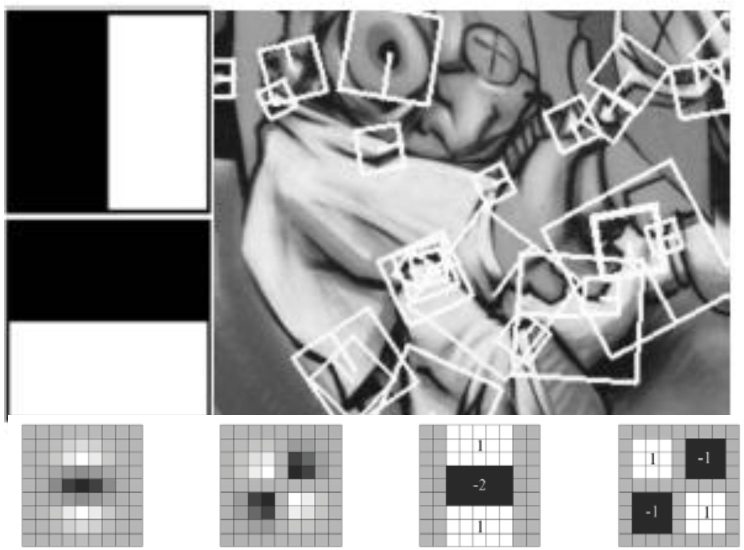
Slide credit: Steve Seitz

b

Working with SIFT Descriptors

- **One image yields:**
  - n 2D points giving positions of the patches
    - [n x 2 matrix]
  - n scale parameters specifying the size of each patch
    - [n x 1 vector]
  - n orientation parameters specifying the angle of the patch
    - [n x 1 vector]
  - n 128-dimensional descriptors: each one is a histogram of the gradient orientations within a patch
    - [n x 128 matrix]

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
  - http://www.vision.ee.ethz.ch/~surf

- GPU implementation available
  - Feature extraction @ 100Hz (detector + descriptor, 640×480 img)
  - http://homes.esat.kuleuven.be/~ncorneli/gpusurf/

[Bay, ECCV’06], [Cornelis, CVGPU’08]

You Can Try It At Home…

- For most local feature detectors, executables are available online:
- http://robots.ox.ac.uk/~vgg/research/affine
- http://www.cs.ubc.ca/~lowe/keypoints/
- http://www.vision.ee.ethz.ch/~surf
- http://homes.esat.kuleuven.be/~ncorneli/gpusurf/

Affine Covariant Features

Collaborative work between: the Visual Geometry Group, Katholieke Universiteit Leuven, Inria Rhone-Alpes and the Center for Machine Perception.

Affine Covariant Region Detectors

Input image

→ Region detector

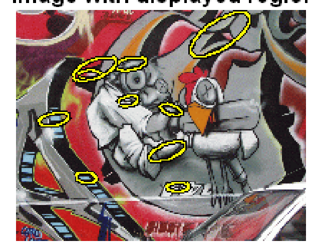
Detector output

format:
1.0
m
$u_1\ v_1\ a_1\ b_1\ c_1$
:
:
$u_m\ v_m\ a_m\ b_m\ c_m$

output example:
img1.haraff

→ display_features.m

Image with displayed regions



Parameters defining an affine region

`u,v,a,b,c` in `a(x-u)(x-u)+2b(x-u)(y-v)+c(y-v)(y-v)=1`
with `(0,0)` at image top left corner

Code

- provided by the authors, see publications for details and links to authors web sites.

Linux binaries	Example of use	Displaying r
Harris-Affine & Hessian-Affine	`prompt>./h_affine.ln -haraff -i img1.ppm -o img1.haraff -thres 1000`	`matlab>> d`
	`prompt>./h_affine.ln -hesaff -i img1.ppm -o img1.hesaff -thres 500`	`matlab>> d`
MSER - Maximaly stable extremal regions (also Windows)	`prompt>./mser.ln -t 2 -es 2 -i img1.ppm -o img1.mser`	`matlab>> d`
IBR - Intensity extrema based detector	`prompt>./ibr.ln img1.ppm img1.ibr -scalefactor 1.0`	`matlab>> d`
EBR - Edge based detector	`prompt> ./ebr.ln img1.ppm img1.ebr`	`matlab>> d`
Salient region detector	`prompt>./salient.ln img1.ppm img1.sal`	`matlab>> d`

http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries

Topics of This Lecture

- Local Feature Extraction (cont'd)
  - Orientation normalization
  - Affine Invariant Feature Extraction
- **Local Descriptors**
  - **SIFT**
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

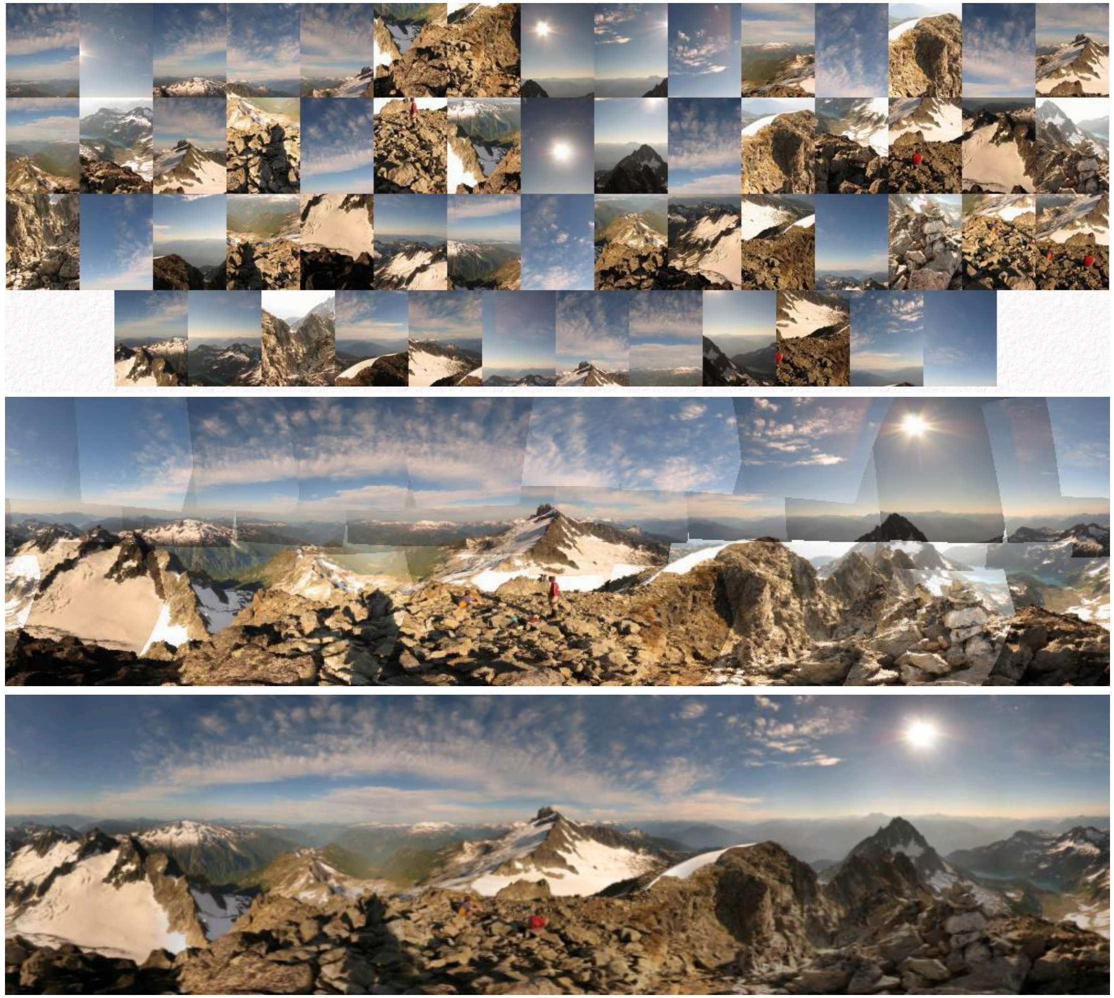
Applications of Local Invariant Features

- Wide baseline stereo
- Motion tracking
- Panoramas
- Mobile robot navigation
- 3D reconstruction
- Recognition
  - Specific objects
  - Textures
  - Categories
- …

Wide-Baseline Stereo

Image from T. Tuytelaars ECCV 2006 tutorial

Automatic Mosaicing



[Brown & Lowe, ICCV'03]

Panorama Stitching

(a) Matier data set (7 images)

(b) Matier final stitch

http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

iPhone version available

[Brown, Szeliski, and Winder, 2005]

Recognition of Specific Objects, Scenes

Schmid and Mohr 1997

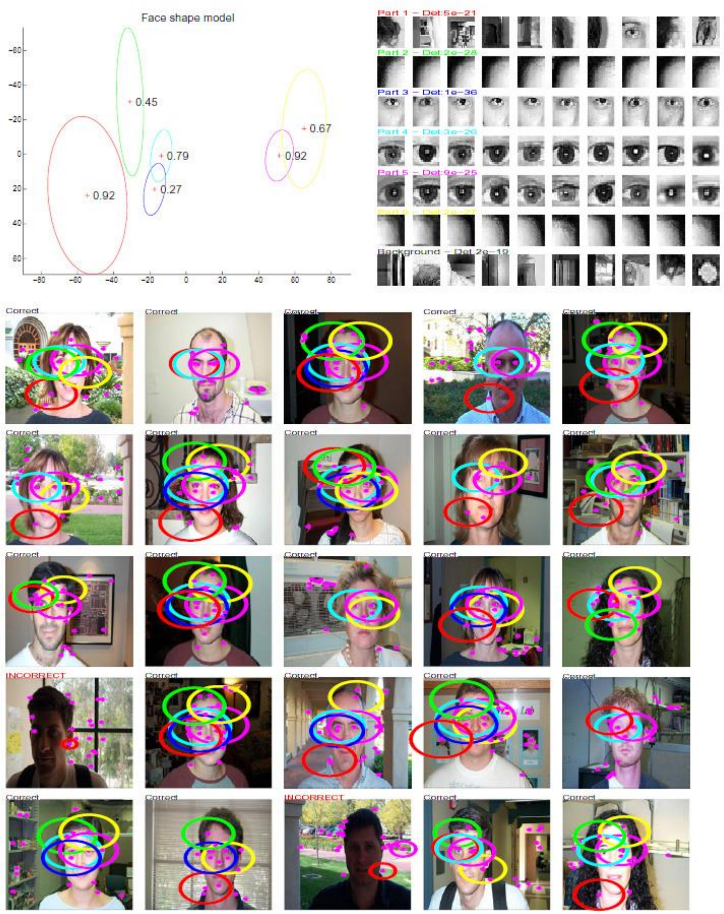
Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

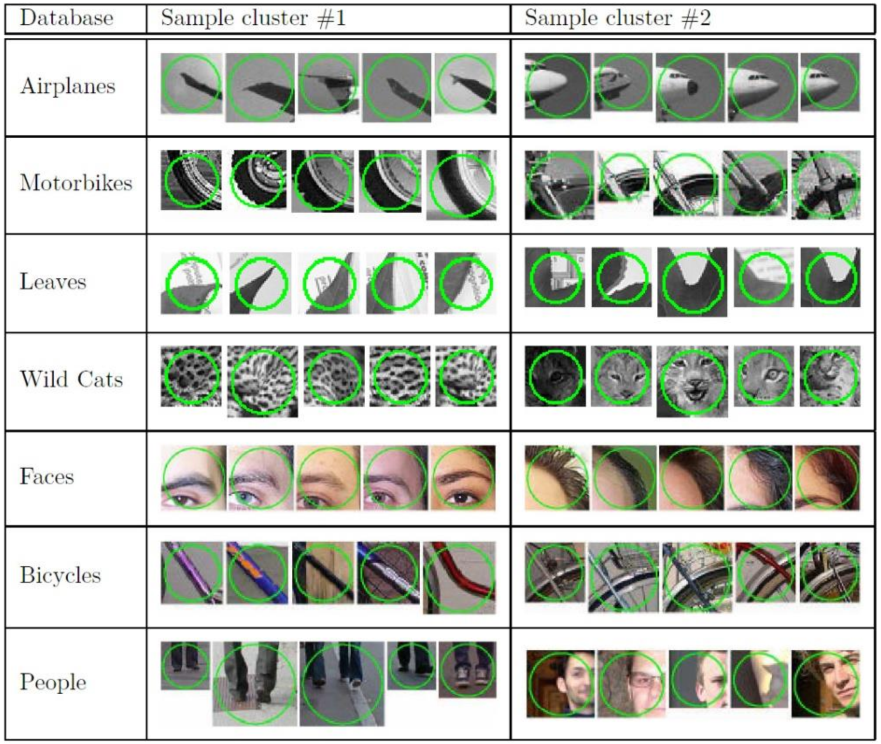
Recognition of Categories

Constellation model



Weber et al. (2000)
Fergus et al. (2003)

Bags of words



Csurka et al. (2004)
Dorko & Schmid (2005)
Sivic et al. (2005)
Lazebnik et al. (2006), …

Value of Local Features

- Advantages
  - Critical to find distinctive and repeatable local regions for multi-view matching.
  - Complexity reduction via selection of distinctive points.
  - Describe images, objects, parts without requiring segmentation; robustness to clutter & occlusion.
  - Robustness: similar descriptors in spite of moderate view changes, noise, blur, etc.

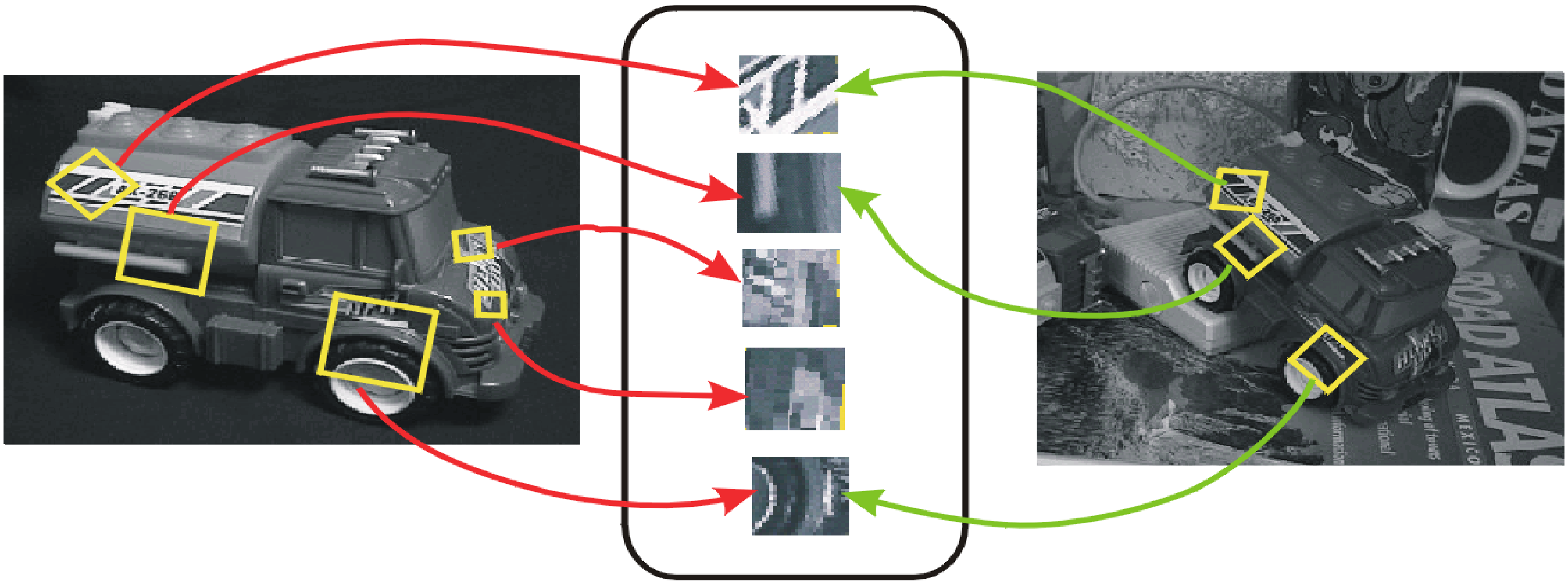
- How can we use local features for such applications?
  - Next: matching and recognition

Topics of This Lecture

- Local Feature Extraction (cont'd)
  - Orientation normalization
  - Affine Invariant Feature Extraction
- Local Descriptors
  - SIFT
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

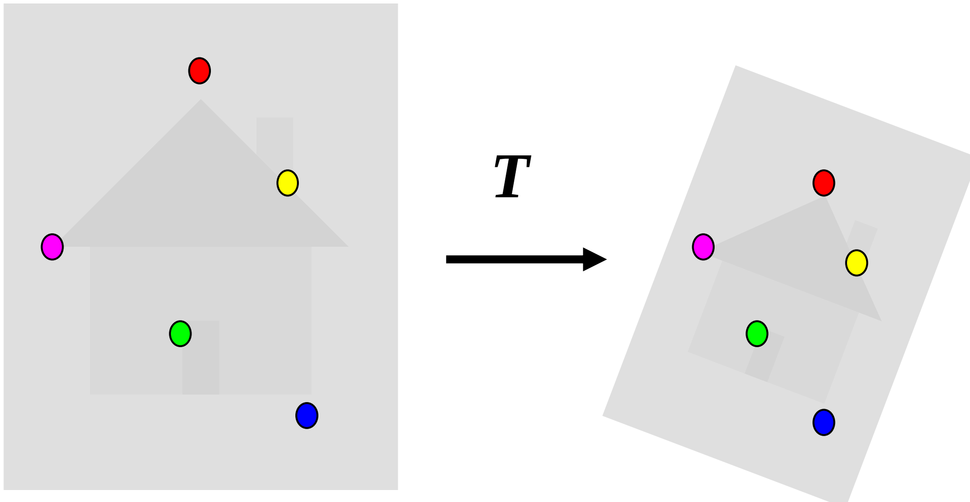
Recognition with Local Features

- Image content is transformed into local features that are invariant to translation, rotation, and scale
- Goal: Verify if they belong to a consistent configuration

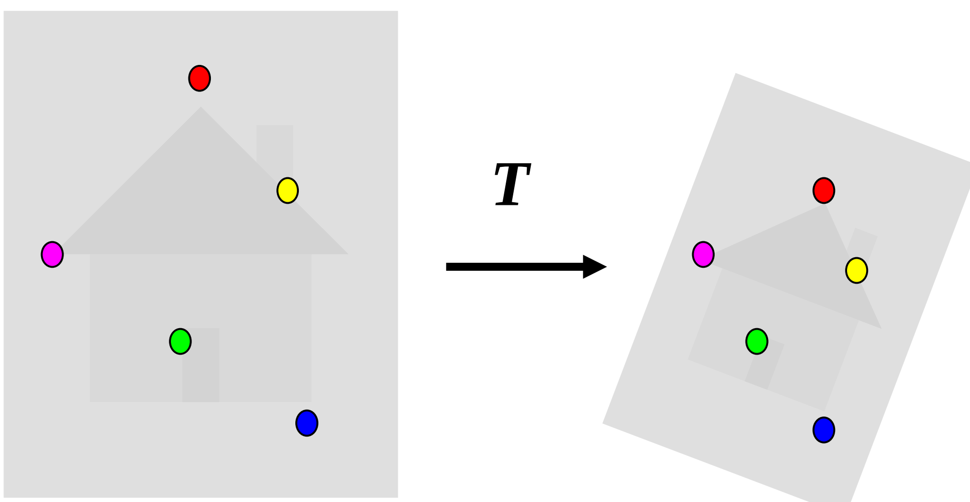


Local Features,
e.g. SIFT

Warping vs. Alignment

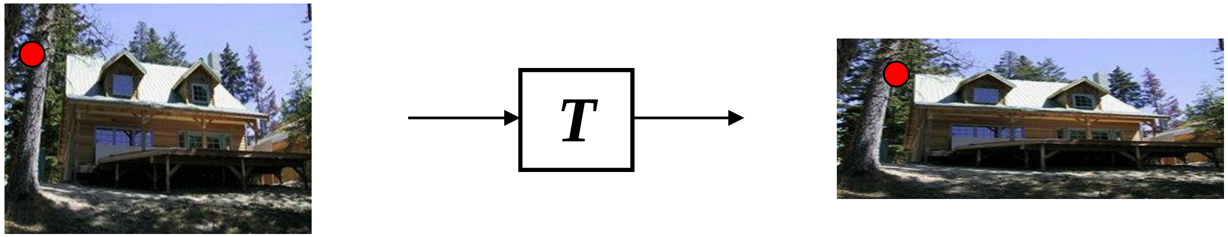


Warping: Given a source image and a transformation, what does the transformed output look like?



Alignment: Given two images with corresponding features, what is the transformation between them?

Parametric (Global) Warping



p = (x,y)   p’ = (x’,y’)

- **Transformation T is a coordinate-changing machine:**

$$p' = T(p)$$

- **What does it mean that T is global?**
  - It’s the same for any point p
  - It can be described by just a few numbers (parameters)
- **Let’s represent T as a matrix:**

$$p' = \mathbf{M}p \,, \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \end{bmatrix}$$

What Can be Represented by a 2×2 Matrix?

- **2D Scaling?**

$$x' = s_x * x$$

$$y' = s_y * y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **2D Rotation around (0,0)?**

$$x' = \cos\theta * x - \sin\theta * y$$

$$y' = \sin\theta * x + \cos\theta * y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **2D Shearing?**

$$x' = x + sh_x * y$$

$$y' = sh_y * x + y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Slide credit: Alexej Efros

What Can be Represented by a 2×2 Matrix?

- **2D Mirror about y axis?**

$$x' = -x$$

$$y' = y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **2D Mirror over (0,0)?**

$$x' = -x$$

$$y' = -y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **2D Translation?**

$$x' = x + t_x$$

$$y' = y + t_y$$

NO!

2D Linear Transforms

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **Only linear 2D transformations can be represented with a 2x2 matrix.**
- **Linear transformations are combinations of …**
  - Scale,
  - Rotation,
  - Shear, and
  - Mirror

Homogeneous Coordinates

- Q: How can we represent translation as a 3x3 matrix using homogeneous coordinates?

$$x' = x + t_x$$

$$y' = y + t_y$$

- A: Using the rightmost column:

$$\text{Translation} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Basic 2D Transformations

- **Basic 2D transformations as 3x3 matrices**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translation

$$\begin{bmatrix} \mathbf{x'} \\ \mathbf{y'} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{s_x} & 0 & 0 \\ 0 & \mathbf{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ 1 \end{bmatrix}$$

Scaling

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shearing

2D Affine Transformations

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

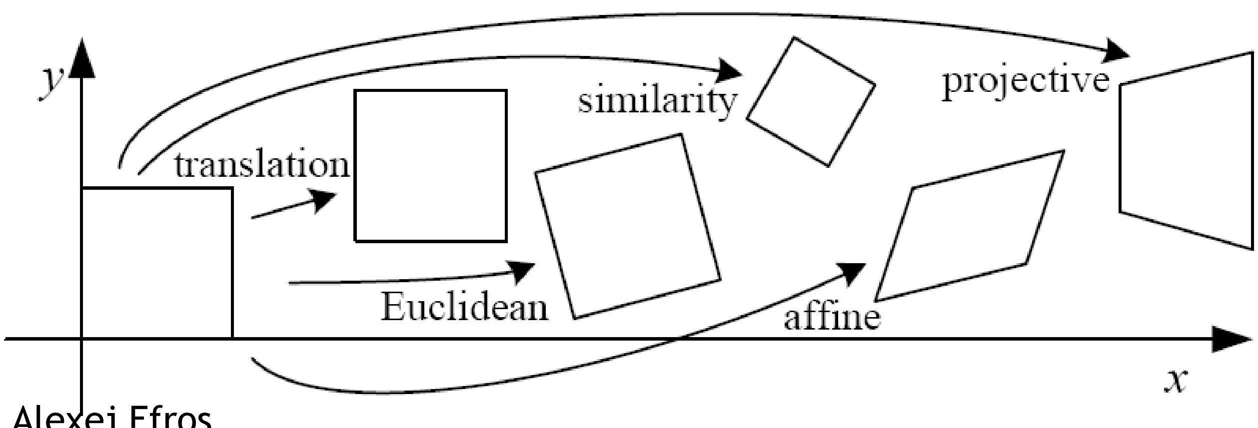
- Affine transformations are combinations of …
  - Linear transformations, and
  - Translations
- Parallel lines remain parallel

Slide credit: Alexej Efros

Projective Transformations

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

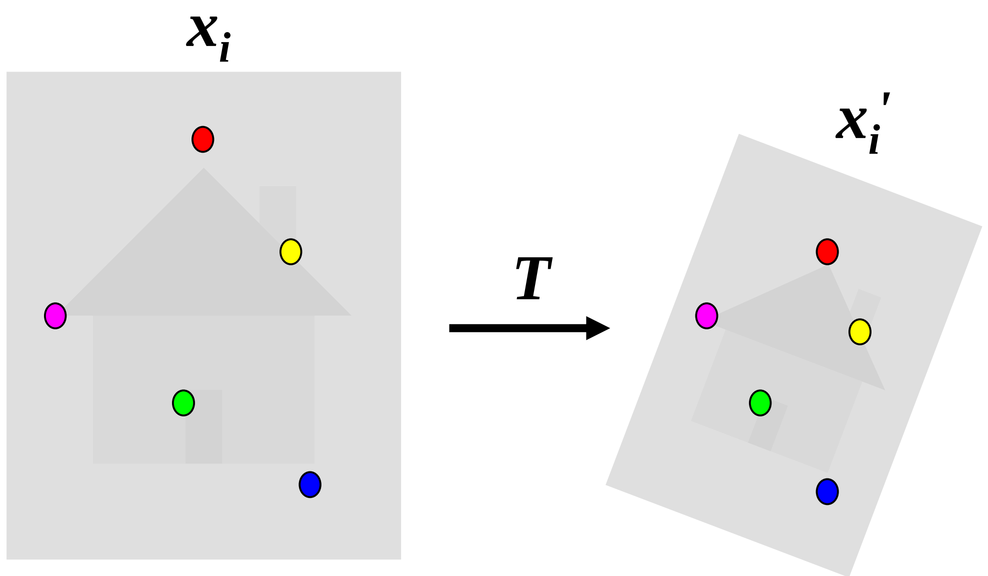
- **Projective transformations:**
  - **Affine transformations, and**
  - **Projective warps**
- **Parallel lines do not necessarily remain parallel**



Slide credit: Alexej Efros

Alignment Problem

- We have previously considered how to fit a model to image evidence
  - e.g., a line to edge points
- In alignment, we will fit the parameters of some transformation according to a set of matching feature pairs (“correspondences”).



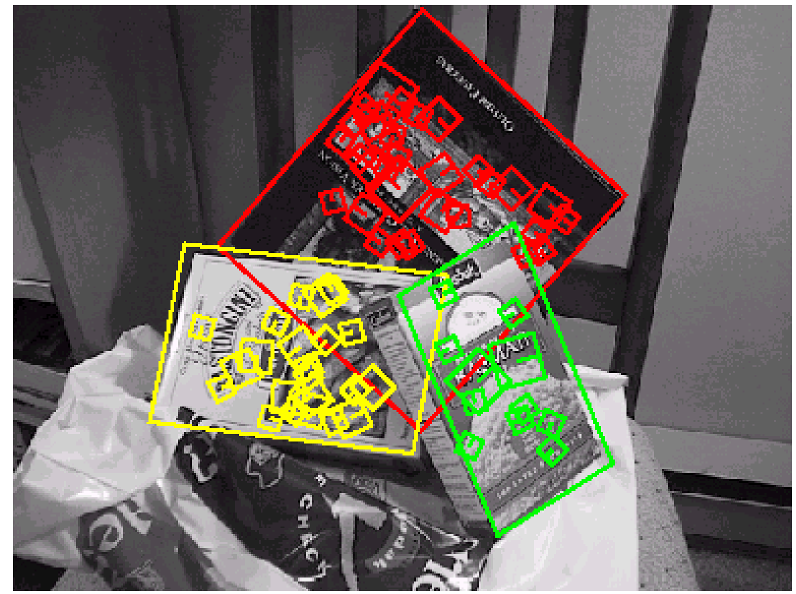
Slide credit: Kristen Grauman

Let’s Start with Affine Transformations

- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models

Slide credit: Svetlana Lazebnik

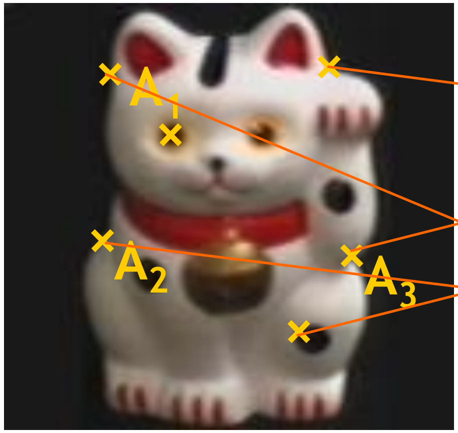
Fitting an Affine Transformation



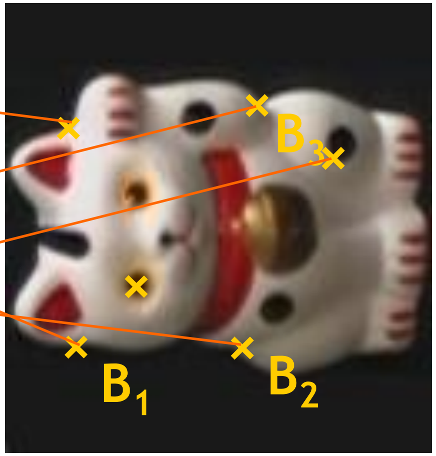
- **Affine model approximates perspective projection of planar objects**

Fitting an Affine Transformation

- Assuming we know the correspondences, how do we get the transformation?



(x_i, y_i)



(x_i', y_i')

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

Recall: Least Squares Estimation

- Set of data points: $(X_1, X_1'), (X_2, X_2'), (X_3, X_3')$
- Goal: a linear function to predict X's from Xs:

$$Xa + b = X'$$

- We want to find a and b.
- How many (X, X') pairs do we need?

$$X_1 a + b = X_1'$$
$$X_2 a + b = X_2'$$

$$\begin{bmatrix} X_1 & 1 \\ X_2 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} X_1' \\ X_2' \end{bmatrix} \qquad Ax = B$$

- What if the data is noisy?

$$\begin{bmatrix} X_1 & 1 \\ X_2 & 1 \\ X_3 & 1 \\ \dots & \dots \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} X_1' \\ X_2' \\ X_3' \\ \dots \end{bmatrix}$$

Overconstrained problem

$$\min \|Ax - B\|^2$$

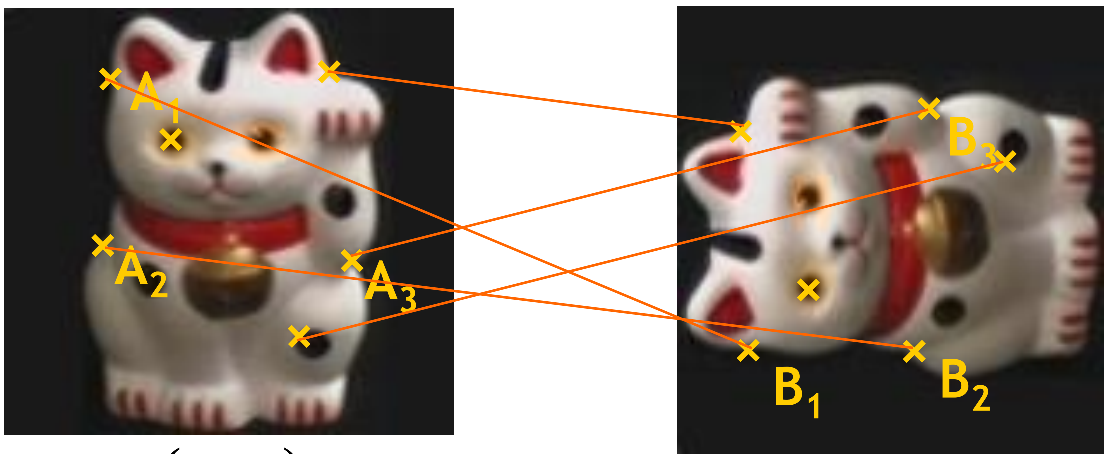
$\Rightarrow$ Least-squares minimization

Matlab:

$$x = A \backslash B$$

Fitting an Affine Transformation

- **Assuming we know the correspondences, how do we get the transformation?**



(x_i, y_i) $\qquad$ (x_i', y_i')

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \qquad \begin{bmatrix} & \\ & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

Fitting an Affine Transformation

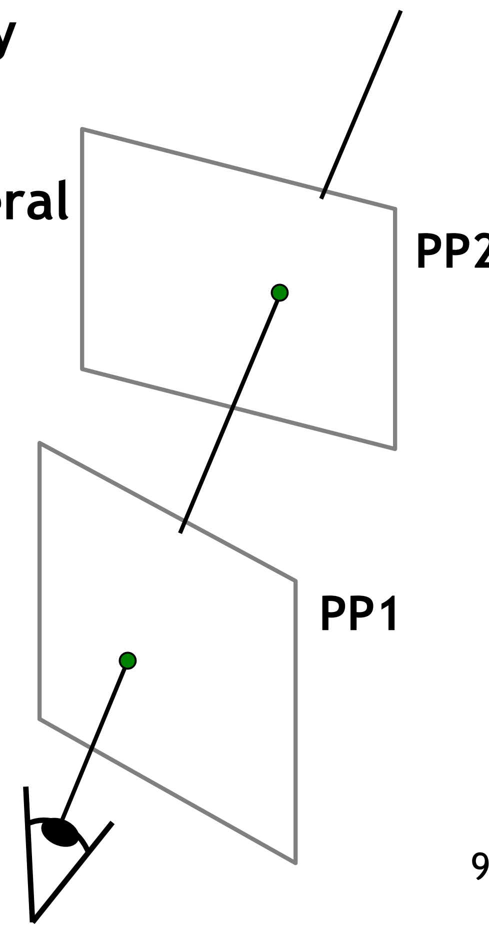
$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

- **How many matches (correspondence pairs) do we need to solve for the transformation parameters?**
- **Once we have solved for the parameters, how do we compute the coordinates of the corresponding point for (x_{new}, y_{new})?**

Homography

- A projective transform is a mapping between any two perspective projections with the same center of projection.
  - I.e. two planes in 3D along the same sight ray
- Properties
  - Rectangle should map to arbitrary quadrilateral
  - Parallel lines aren't
  - but must preserve straight lines
- This is called a homography

$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{p'} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{p}$$



Homography

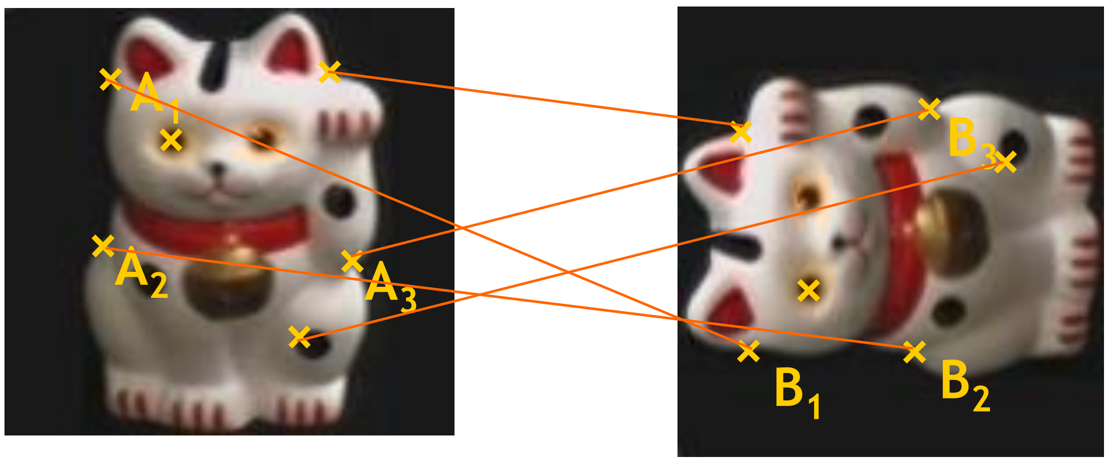
- A projective transform is a mapping between any two perspective projections with the same center of projection.
  - I.e. two planes in 3D along the same sight ray
- Properties
  - Rectangle should map to arbitrary quadrilateral
  - Parallel lines aren't
  - but must preserve straight lines
- This is called a **homography**

$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{p'} = \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{p}$$

Set scale factor to 1
$\Rightarrow$ 8 parameters left.

Fitting a Homography

- **Estimating the transformation**



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

Homogenous coordinates

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Image coordinates

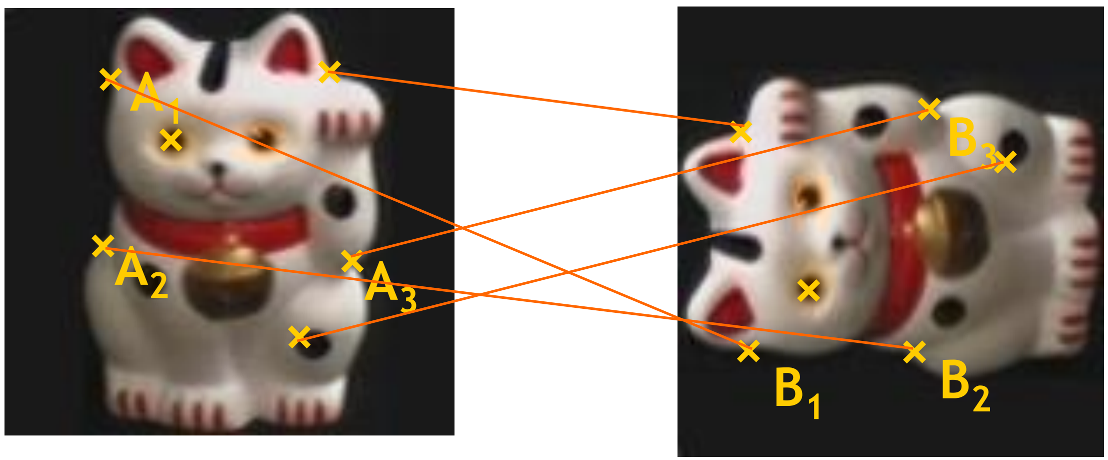
$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

Matrix notation

$$x' = Hx$$
$$x'' = \frac{1}{z'} x'$$

Fitting a Homography

- Estimating the transformation



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

Homogenous coordinates

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Image coordinates

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

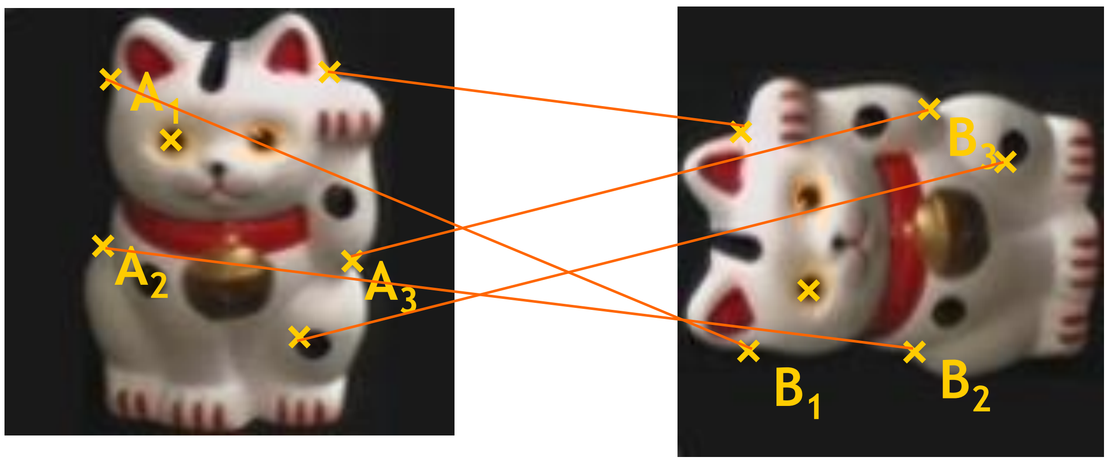
Matrix notation

$$x' = Hx$$

$$x'' = \frac{1}{z'} x'$$

Fitting a Homography

- **Estimating the transformation**



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

Homogenous coordinates

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Image coordinates

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

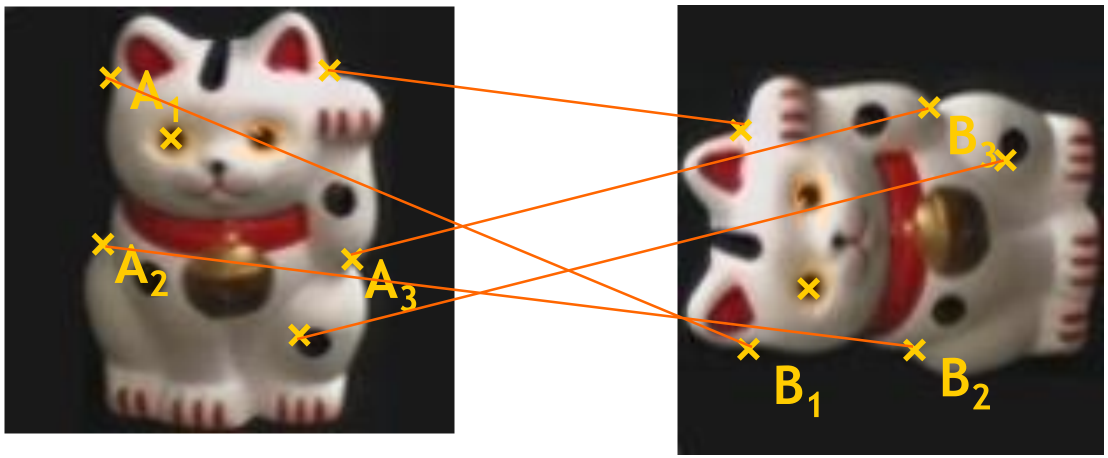
Matrix notation

$$x' = Hx$$
$$x'' = \tfrac{1}{z'} x'$$

$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

Fitting a Homography

- Estimating the transformation



Homogenous coordinates

Image coordinates

Matrix notation

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \boxed{\frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}}$$

$$x' = Hx$$

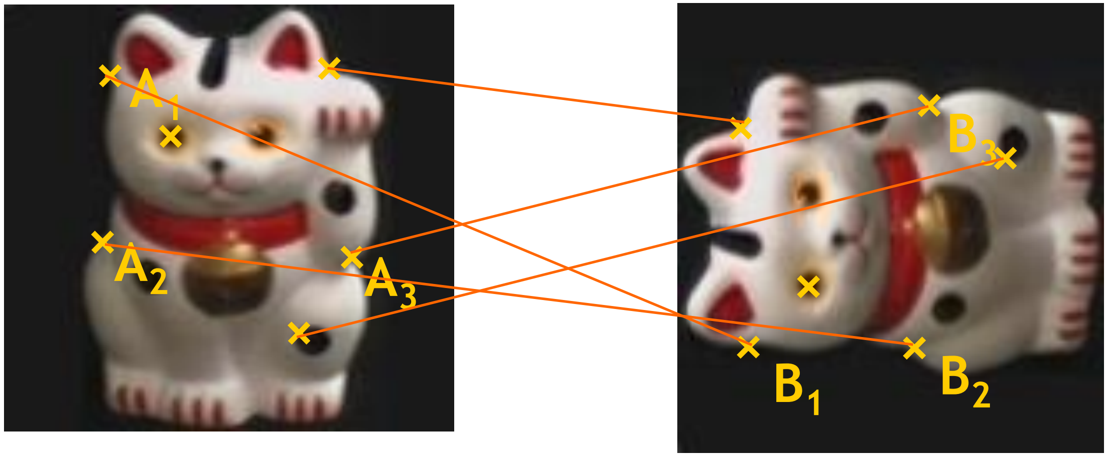
$$x'' = \frac{1}{z'} x'$$

$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1} \qquad y_{A_1} = \frac{h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

Slide credit: Krystian Mikolajczyk

Fitting a Homography

- Estimating the transformation



Homogenous coordinates **Image coordinates**

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

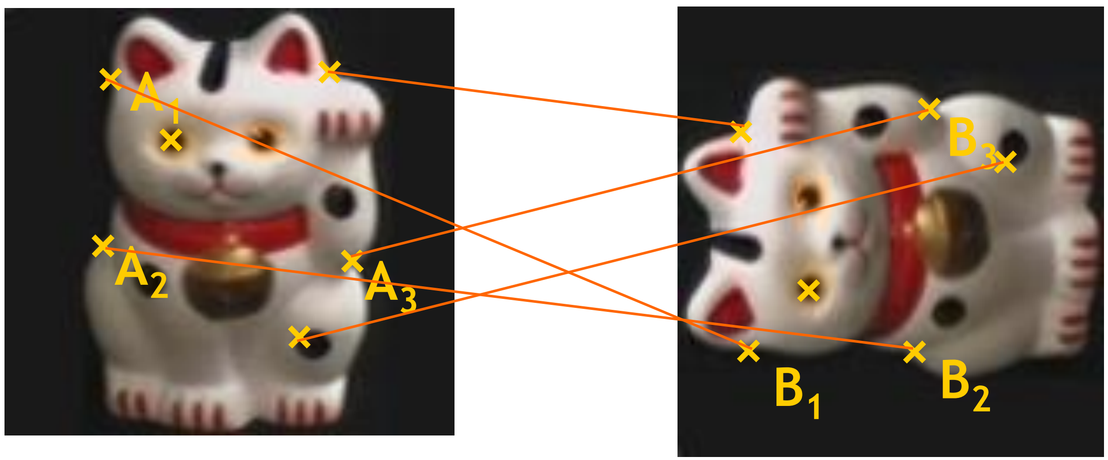
$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1} \qquad y_{A_1} = \frac{h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

$$x_{A_1} h_{31}\, x_{B_1} + x_{A_1} h_{32} y_{B_1} + x_{A_1} = h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}$$

Slide credit: Krystian Mikolajczyk

Fitting a Homography

- **Estimating the transformation**



Homogenous coordinates **Image coordinates**

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

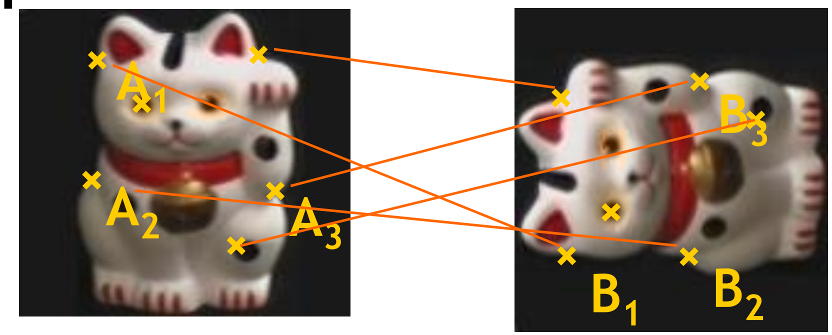
$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1} \qquad y_{A_1} = \frac{h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

$$x_{A_1} h_{31}\, x_{B_1} + x_{A_1} h_{32} y_{B_1} + x_{A_1} = h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}$$

$$\boxed{\begin{aligned} h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13} - x_{A_1} h_{31}\, x_{B_1} - x_{A_1} h_{32} y_{B_1} - x_{A_1} &= 0 \\ h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23} - y_{A_1} h_{31}\, x_{B_1} - y_{A_1} h_{32} y_{B_1} - y_{A_1} &= 0 \end{aligned}}$$

Fitting a Homography

- **Estimating the transformation**



$$\boxed{\begin{aligned} h_{11}\,x_{B_1} + h_{12}y_{B_1} + h_{13} - x_{A_1}h_{31}\,x_{B_1} - x_{A_1}h_{32}y_{B_1} - x_{A_1} &= 0 \\ h_{21}\,x_{B_1} + h_{22}y_{B_1} + h_{23} - y_{A_1}h_{31}\,x_{B_1} - y_{A_1}h_{32}y_{B_1} - y_{A_1} &= 0 \end{aligned}}$$

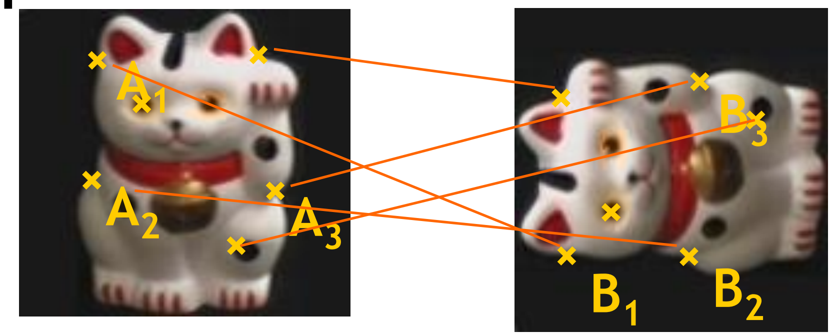
$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

$$\begin{bmatrix} x_{B_1} & y_{B_1} & 1 & 0 & 0 & 0 & -x_{A_1}x_{B_1} & -x_{A_1}y_{B_1} & -x_{A_1} \\ 0 & 0 & 0 & x_{B_1} & y_{B_1} & 1 & -y_{A_1}x_{B_1} & -y_{A_1}y_{B_1} & -y_{A_1} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \end{bmatrix} \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ . \end{bmatrix}$$

$$Ah = 0$$

Fitting a Homography

- **Estimating the transformation**
- **Solution:**
  - Null-space vector of A
  - Corresponds to smallest eigenvector



$$Ah = 0$$

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

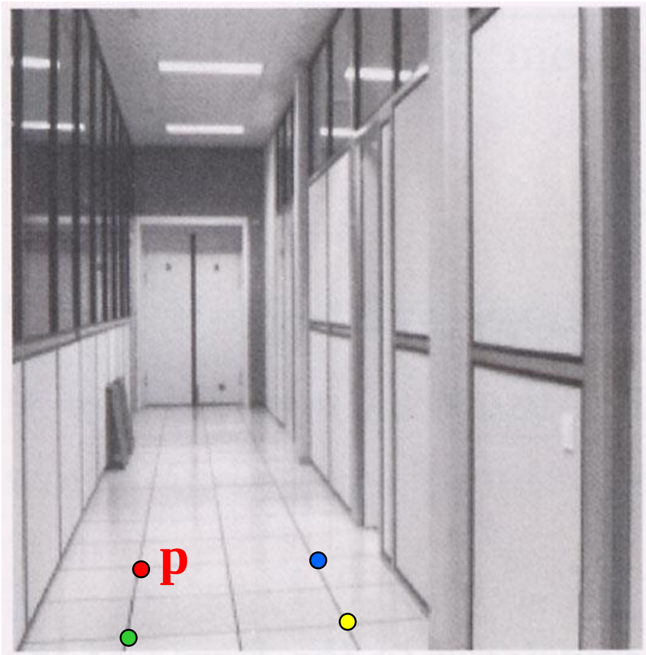
SVD

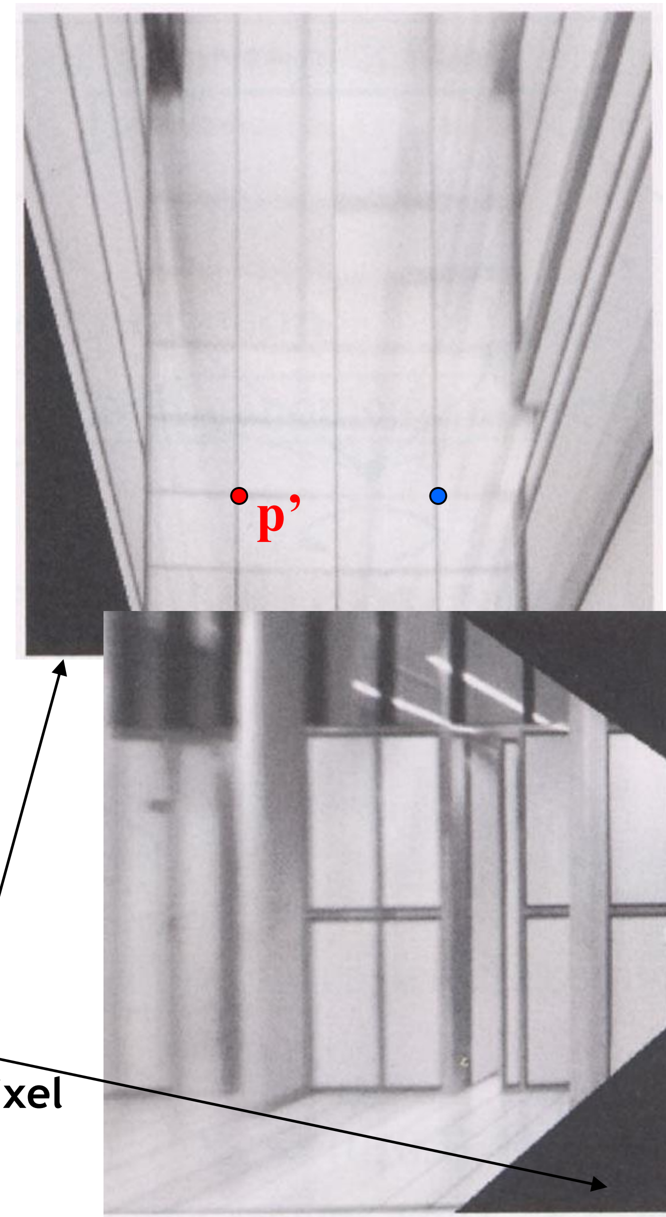
$$\mathbf{A} = \mathbf{UDV}^T = \mathbf{U} \begin{bmatrix} d_{11} & \cdots & d_{19} \\ \vdots & \ddots & \vdots \\ d_{91} & \cdots & d_{99} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & \boxed{v_{19}} \\ \vdots & \ddots & \boxed{\vdots} \\ v_{91} & \cdots & \boxed{v_{99}} \end{bmatrix}^T$$

$$\boxed{\mathbf{h} = \frac{[v_{19}, \cdots, v_{99}]}{v_{99}}}$$

Minimizes least square error

Image Warping with Homographies





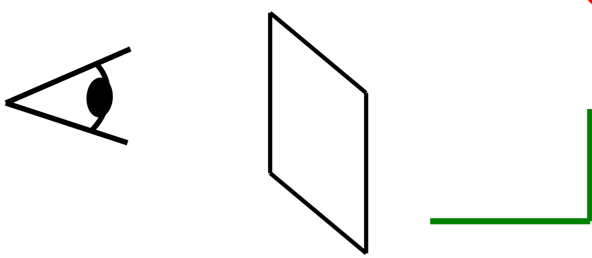


Image plane in front

Black area where no pixel maps to

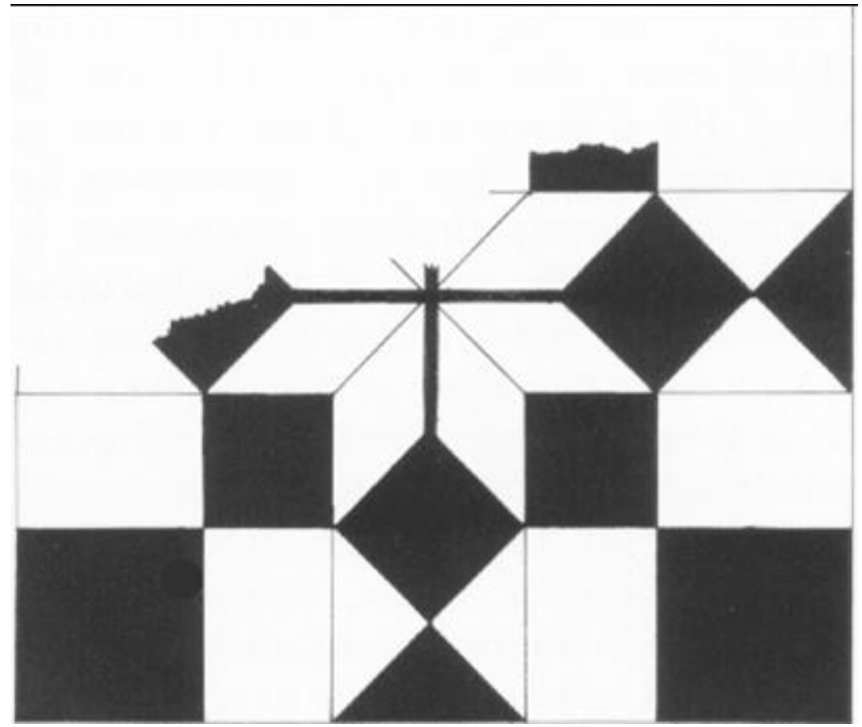
Uses: Analyzing Patterns and Shapes

- What is the shape of the b/w floor pattern?

The floor (enlarged)

Analyzing Patterns and Shapes

Automatic rectification



From Martin Kemp *The Science of Art* *(manual reconstruction)*

Slide credit: Antonio Criminisi

Summary: Recognition by Alignment

- Basic matching algorithm
  1. Detect interest points in two images.
  2. Extract patches and compute a descriptor for each one.
  3. Compare one feature from image 1 to every feature in image 2 and select the nearest-neighbor pair.
  4. Repeat the above for each feature from image 1.
  5. Use the list of best pairs to estimate the transformation between images.

- Transformation estimation
  - Affine
  - Homography

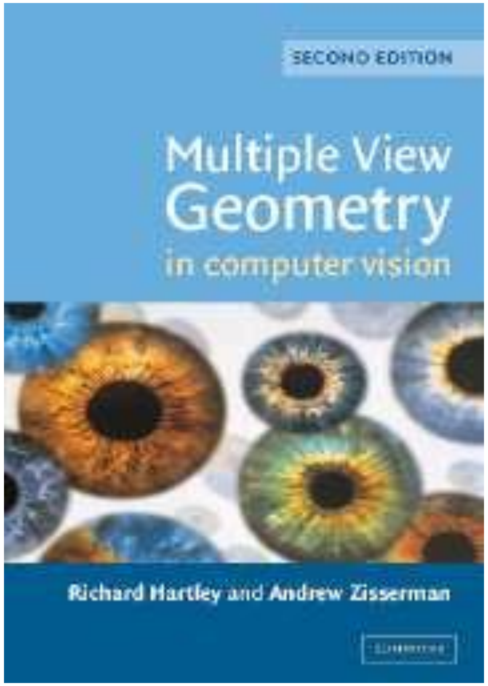
RWTH AACHEN UNIVERSITY

Time for a Demo…

Automatic panorama stitching

References and Further Reading

- More details on homography estimation can be found in Chapter 4.7 of
  - R. Hartley, A. Zisserman
    Multiple View Geometry in Computer Vision
    2nd Ed., Cambridge Univ. Press, 2004



- Details about the DoG detector and the SIFT descriptor can be found in
  - D. Lowe, Distinctive image features from scale-invariant keypoints,
    IJCV 60(2), pp. 91-110, 2004

- Try the available local feature detectors and descriptors
  - http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries