Computer Vision - Lecture 10

Local Features

28.11.2016

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

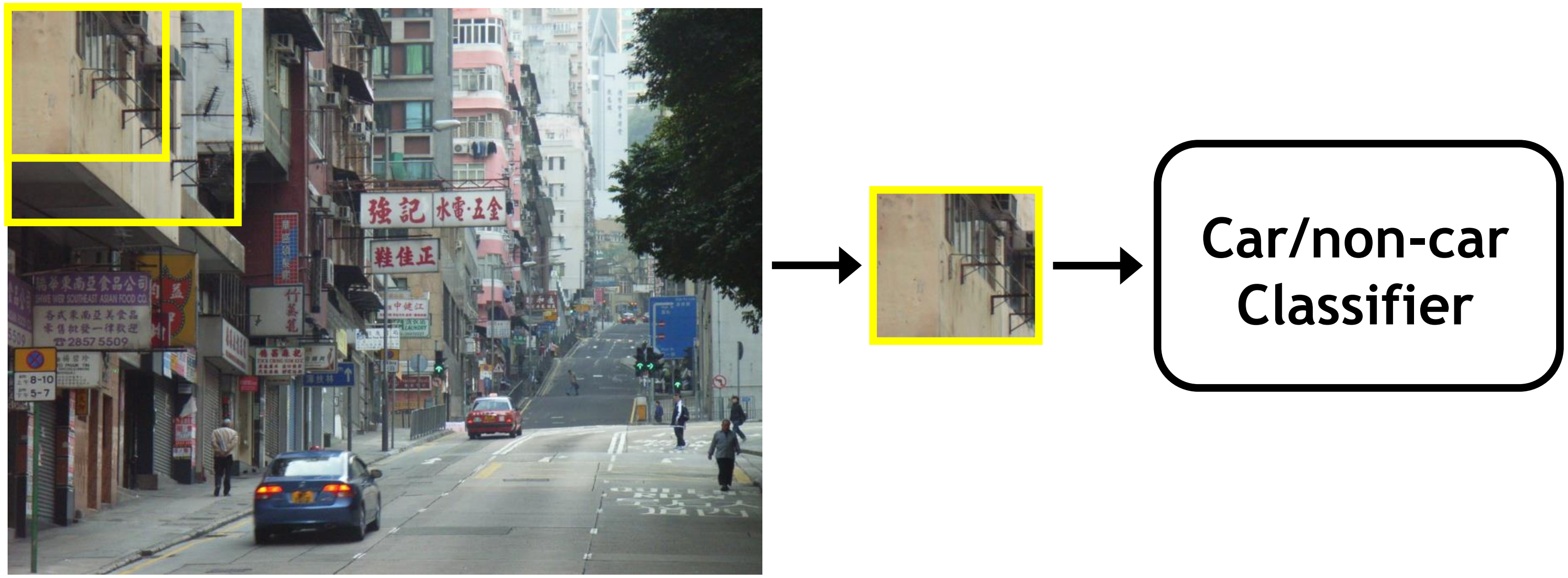
leibe@vision.rwth-aachen.de

Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition & 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

Recap: Sliding-Window Object Detection

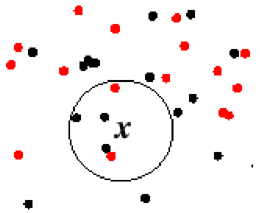
- If object may be in a cluttered scene, slide a window around looking for it.



- Essentially, this is a brute-force approach with many local decisions.

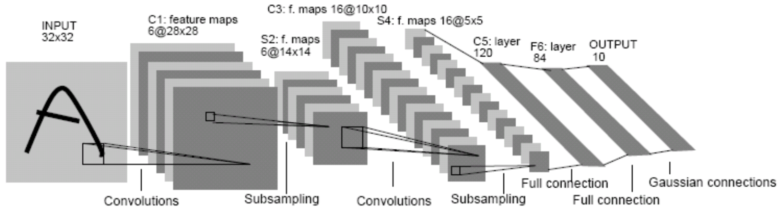
Classifier Construction: Many Choices…

Nearest Neighbor



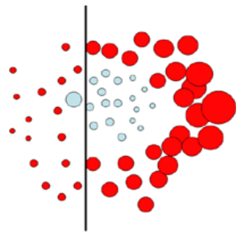
Berg, Berg, Malik 2005,
Chum, Zisserman 2007,
Boiman, Shechtman, Irani 2008, …

Neural networks



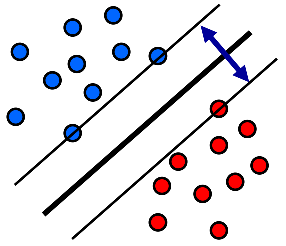
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Boosting



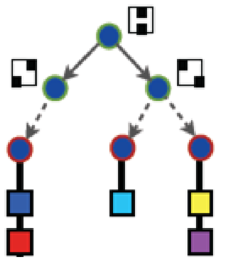
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,
Benenson 2012, …

Support Vector Machines



Vapnik, Schölkopf 1995,
Papageorgiou, Poggio '01,
Dalal, Triggs 2005,
Vedaldi, Zisserman 2012

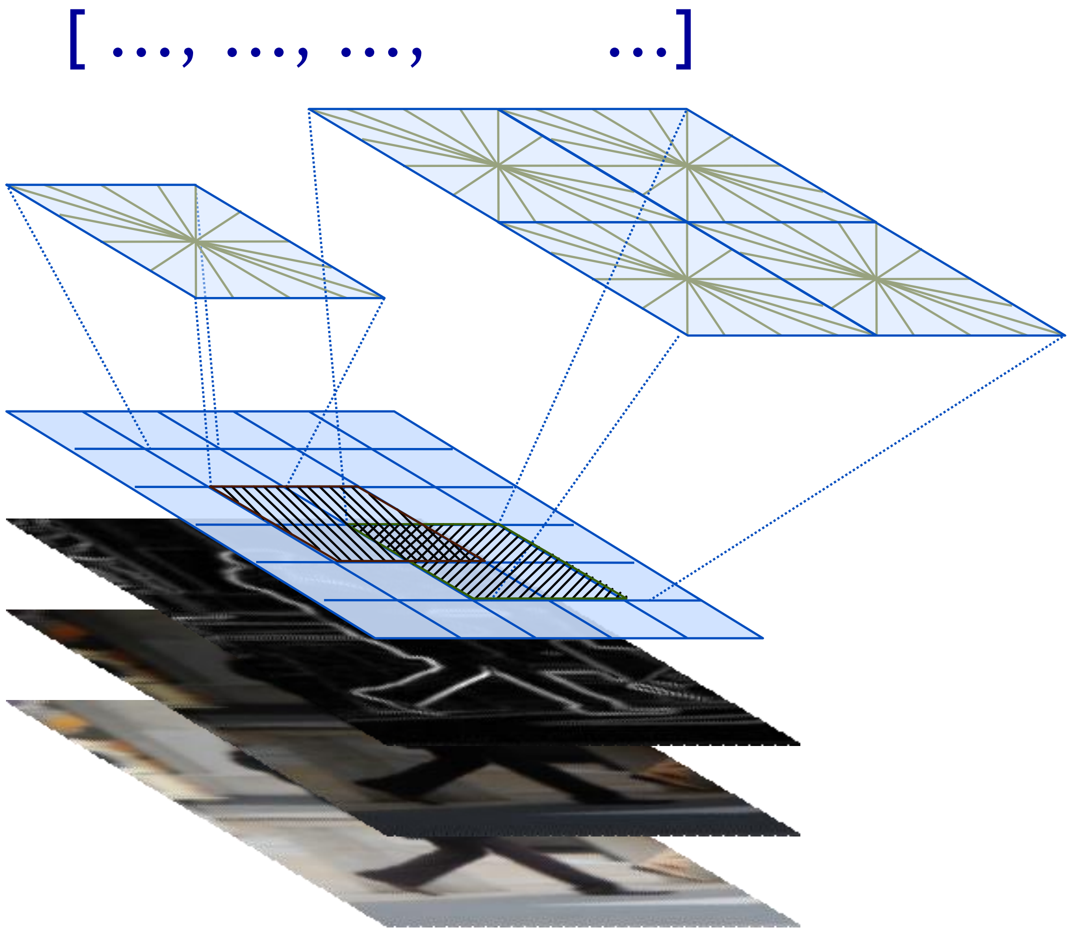
Randomized Forests



Amit, Geman 1997,
Breiman 2001,
Lepetit, Fua 2006,
Gall, Lempitsky 2009,…

Slide adapted from Kristen Grauman

Recap: HOG Descriptor Processing Chain



Object/Non-object

↑

Linear SVM

↑

Collect HOGs over detection window

↑

Contrast normalize over overlapping spatial cells

↑

Weighted vote in spatial & orientation cells

↑

Compute gradients

↑

Gamma compression

↑

Image Window

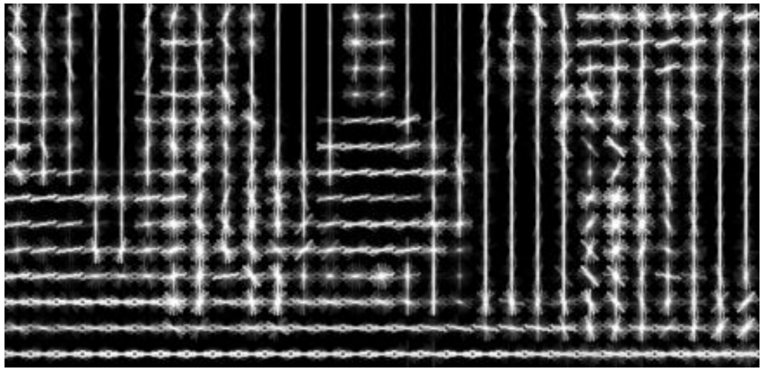
Slide credit: Navneet Dalal

Recap: Pedestrian Detection with HOG

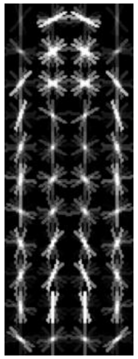
- Train a pedestrian template using a linear SVM
- At test time, convolve feature map with template
  - Linear SVM classification function

$$y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

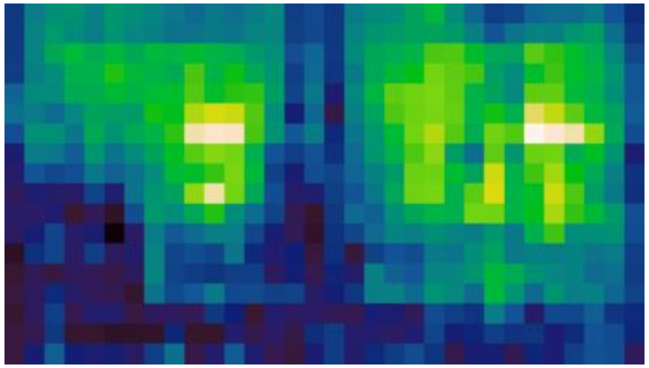
HOG feature map



Template



Detector response map

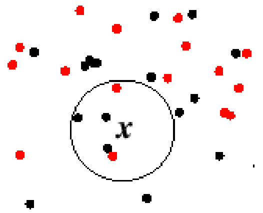


N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005

Slide credit: Svetlana Lazebnik

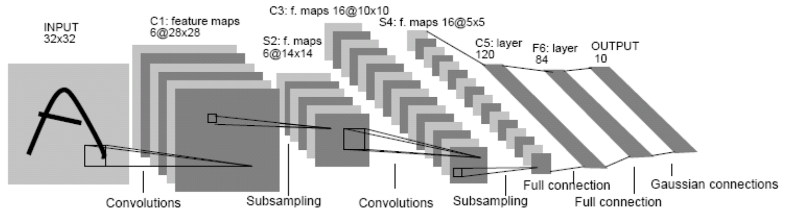
Classifier Construction: Many Choices…

Nearest Neighbor



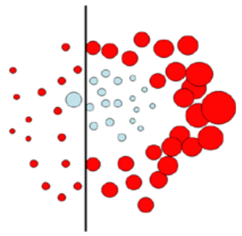
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005,
Boiman, Shechtman, Irani 2008, …

Neural networks



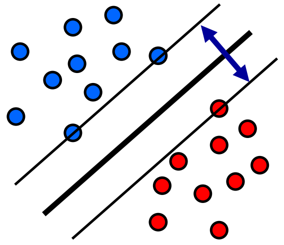
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Boosting



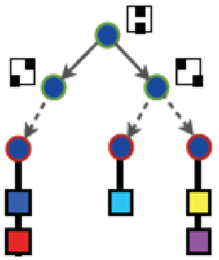
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,
Benenson 2012, …

Support Vector Machines



Vapnik, Schölkopf 1995,
Papageorgiou, Poggio '01,
Dalal, Triggs 2005,
Vedaldi, Zisserman 2012

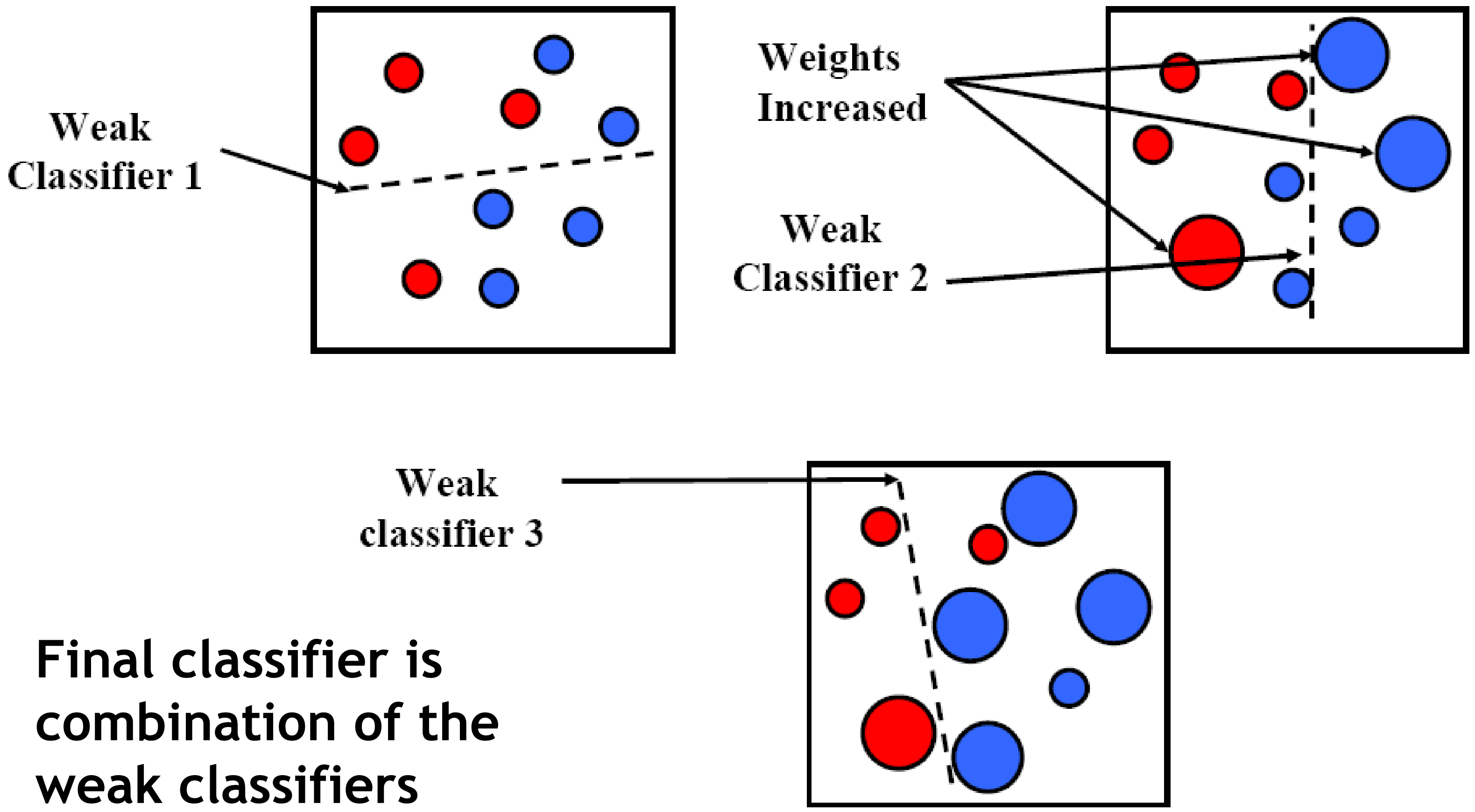
Randomized Forests



Amit, Geman 1997,
Breiman 2001,
Lepetit, Fua 2006,
Gall, Lempitsky 2009,…

Slide adapted from Kristen Grauman

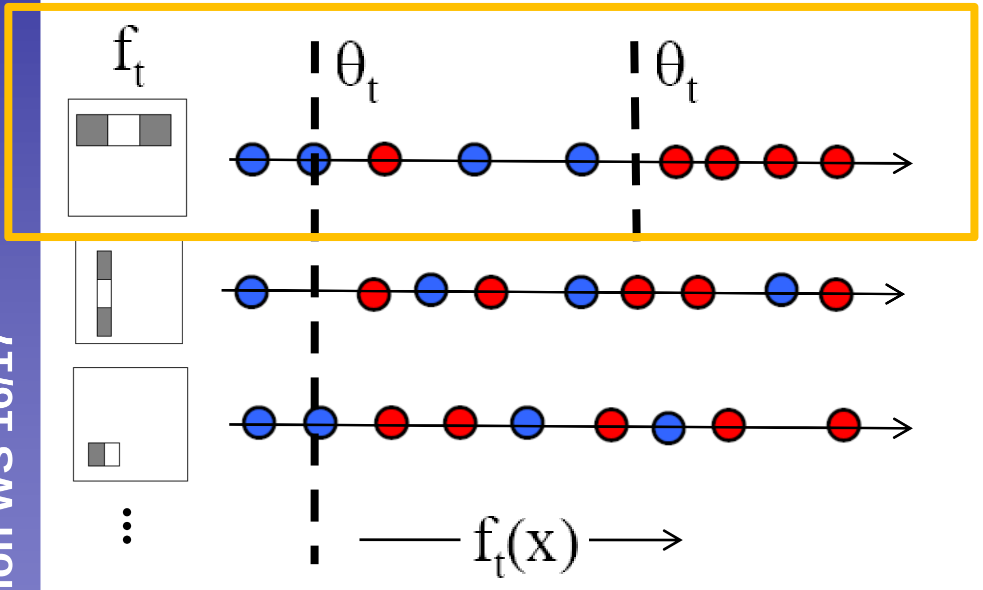
Recap: AdaBoost



Final classifier is combination of the weak classifiers

Recap: AdaBoost Feature+Classifier Selection

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of *weighted* error.



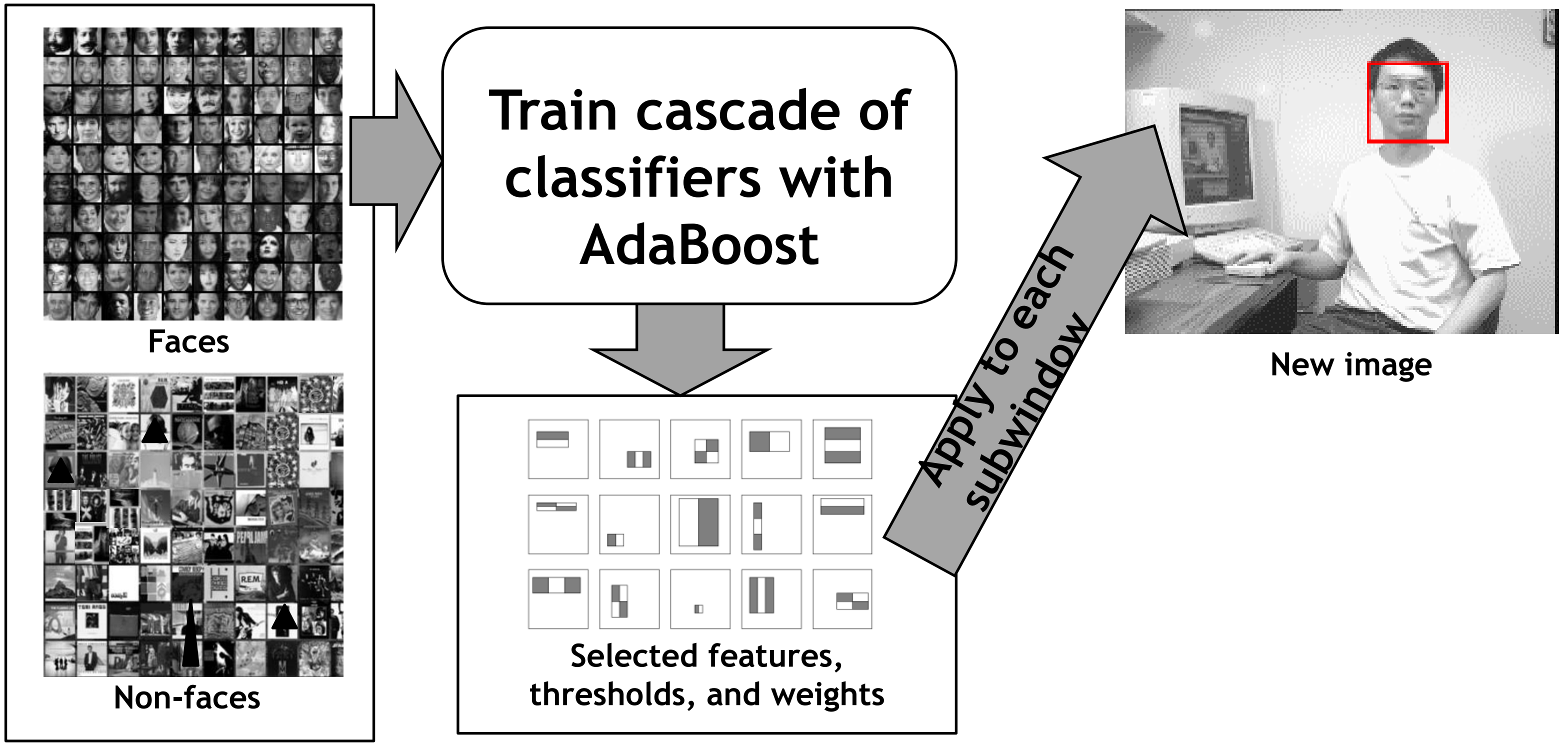
Outputs of a possible rectangle feature on faces and non-faces.

Resulting weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

Recap: Viola-Jones Face Detector


Faces
Non-faces
Train cascade of classifiers with AdaBoost
Selected features, thresholds, and weights
Apply to each subwindow
New image

- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- [Implementation available in OpenCV: http://sourceforge.net/projects/opencvlibrary/]

Slide credit: Kristen Grauman

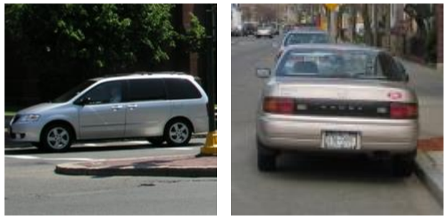
Limitations: Low Training Resolutions

- Many (older) S/W detectors operate on tiny images
  - Viola&Jones: 24×24 pixels
  - Torralba et al.: 32×32 pixels
  - Dalal&Triggs: 64×96 pixels (notable exception)
- Main reasons
  - Training efficiency (exhaustive feature selection in AdaBoost)
  - Evaluation speed
  - Want to recognize objects at small scales
- But…
  - Limited information content available at those resolutions
  - Not enough support to compensate for occlusions!

Limitations: Changing Aspect Ratios

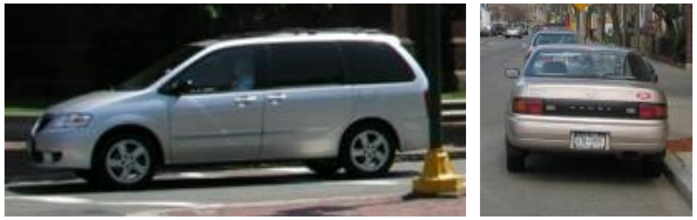
- Sliding window requires fixed window size
  - Basis for learning efficient cascade classifier
- How to deal with changing aspect ratios?
  - Fixed window size

    ⇒ Wastes training dimensions



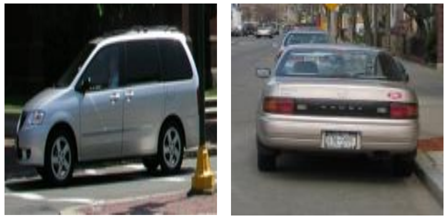
  - Adapted window size

    ⇒ Difficult to share features



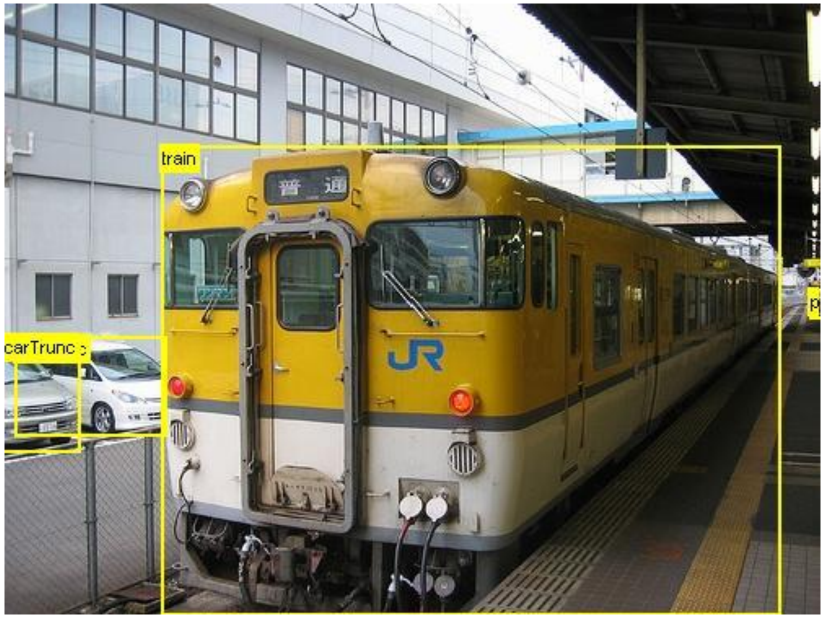
  - "Squashed" views [Dalal&Triggs]

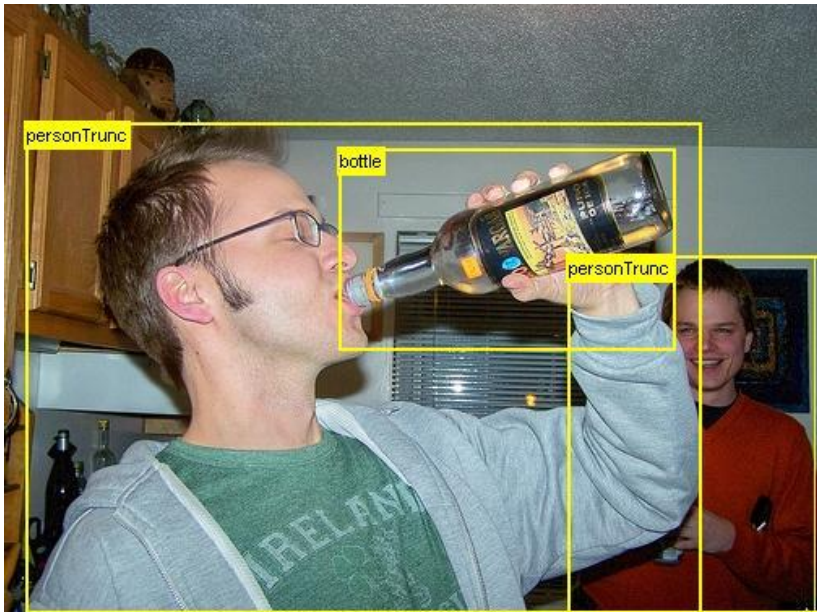
    ⇒ Need to squash test image, too



Limitations (continued)

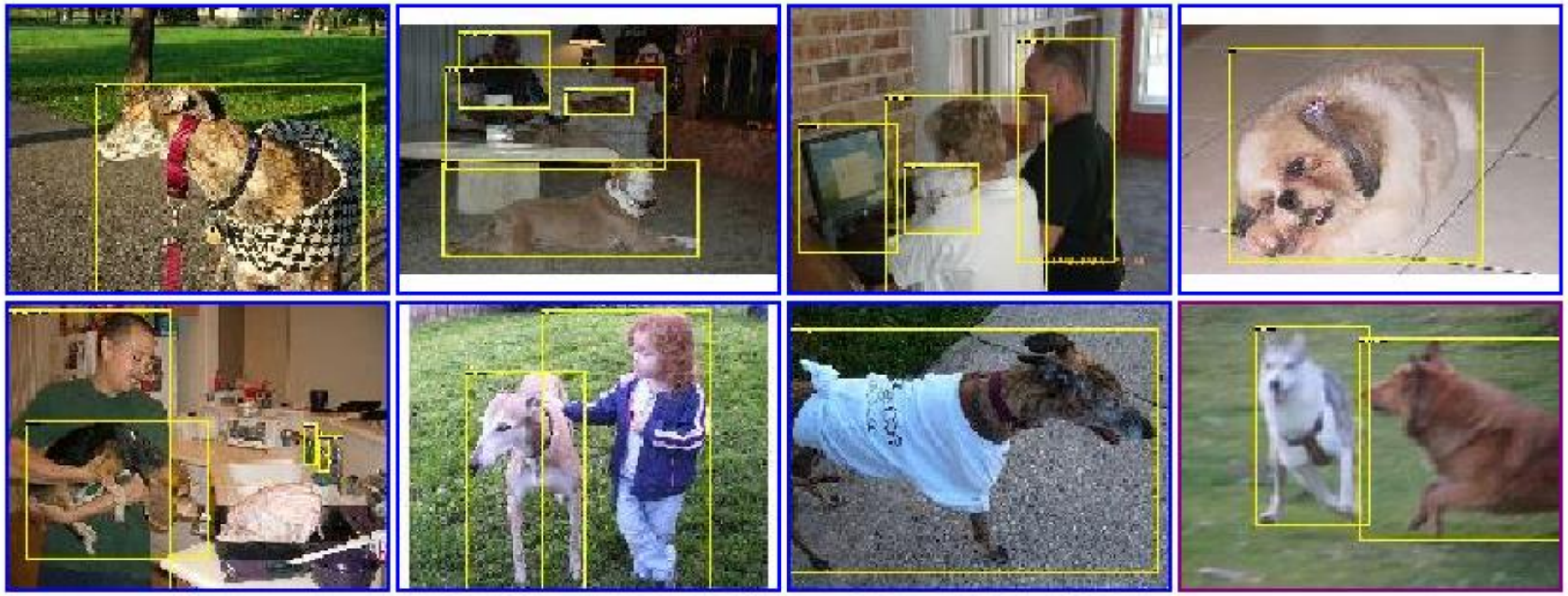
- Not all objects are "box" shaped





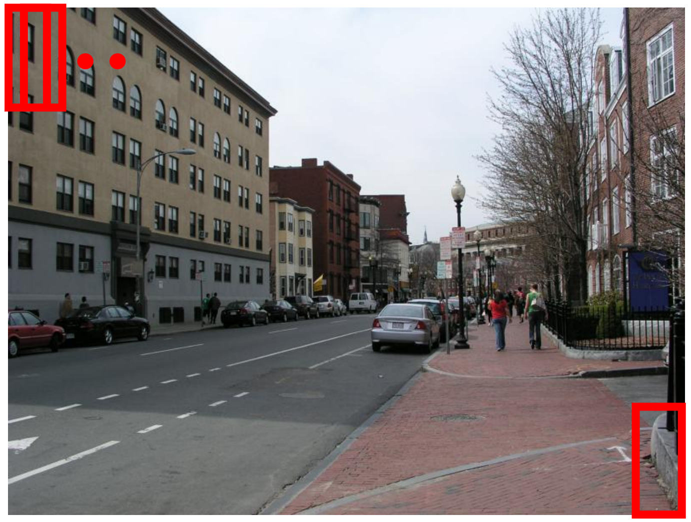
Limitations (continued)

- Non-rigid, deformable objects not captured well with representations assuming a fixed 2D structure; or must assume fixed viewpoint
- Objects with less-regular textures not captured well with holistic appearance-based descriptions

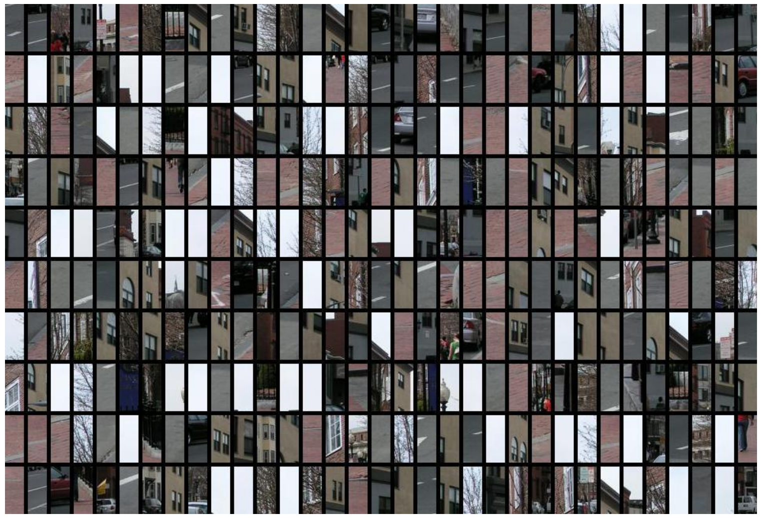


Limitations (continued)

- If considering windows in isolation, context is lost



Sliding window



Detector’s view

Figure credit: Derek Hoiem

Limitations (continued)

- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions

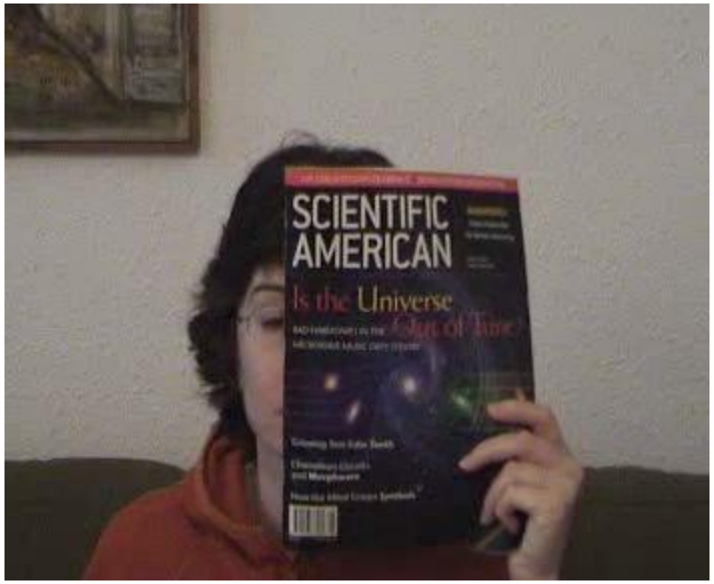


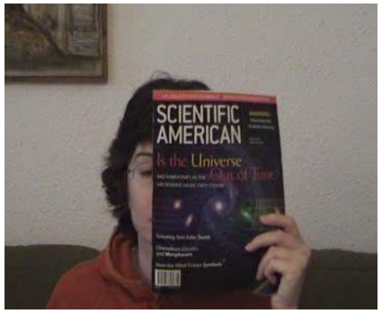
Image credit: Adam, Rivlin, & Shimshoni

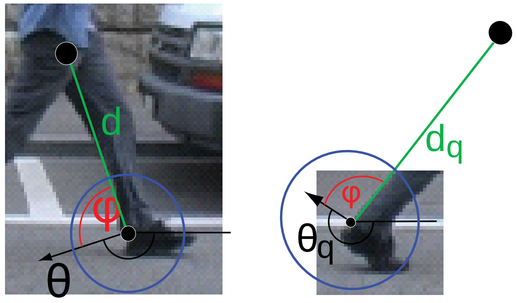
Topics of This Lecture

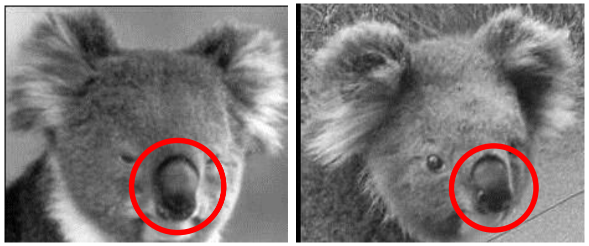
- Local Invariant Features
  - Motivation
  - Requirements, Invariances
- Keypoint Localization
  - Harris detector
  - Hessian detector
- Scale Invariant Region Selection
  - Automatic scale selection
  - Laplacian-of-Gaussian detector
  - Difference-of-Gaussian detector
  - Combinations
- Local Descriptors
  - Orientation normalization
  - SIFT

Motivation

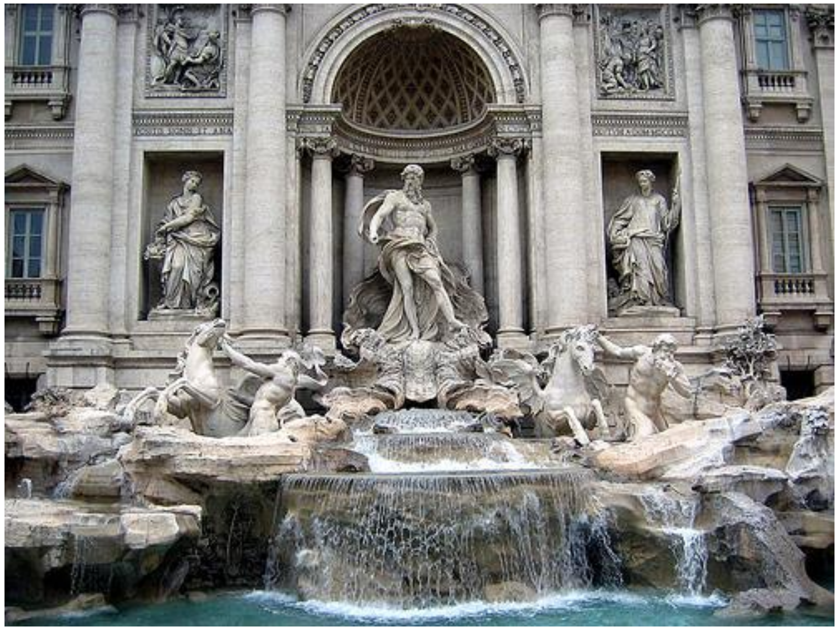
- Global representations have major limitations
- Instead, describe and match only local regions
- Increased robustness to
  - Occlusions
  - Articulation
  - Intra-category variations

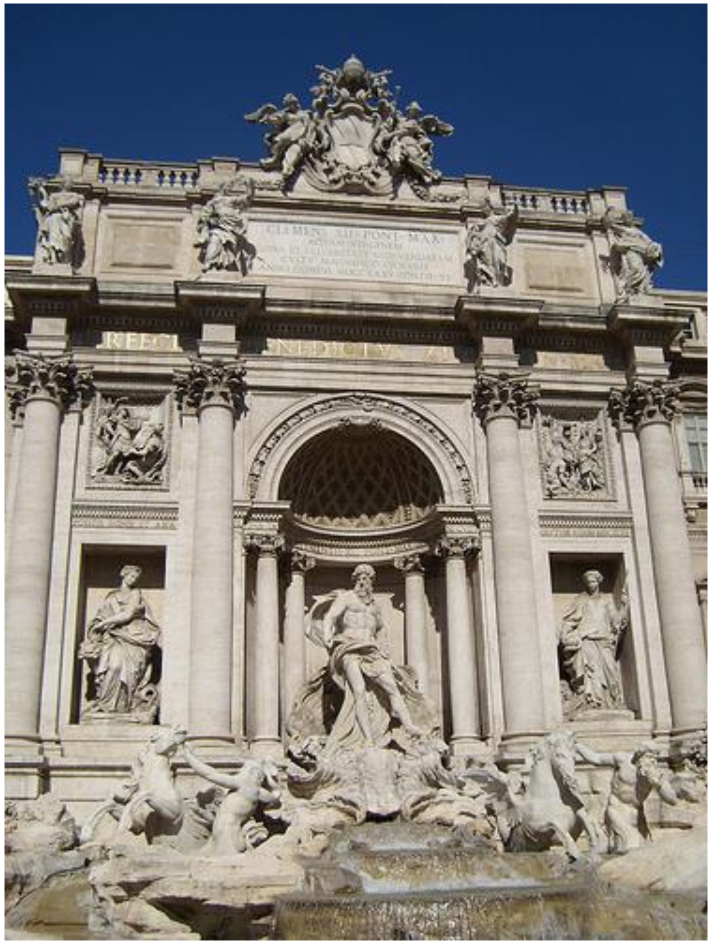




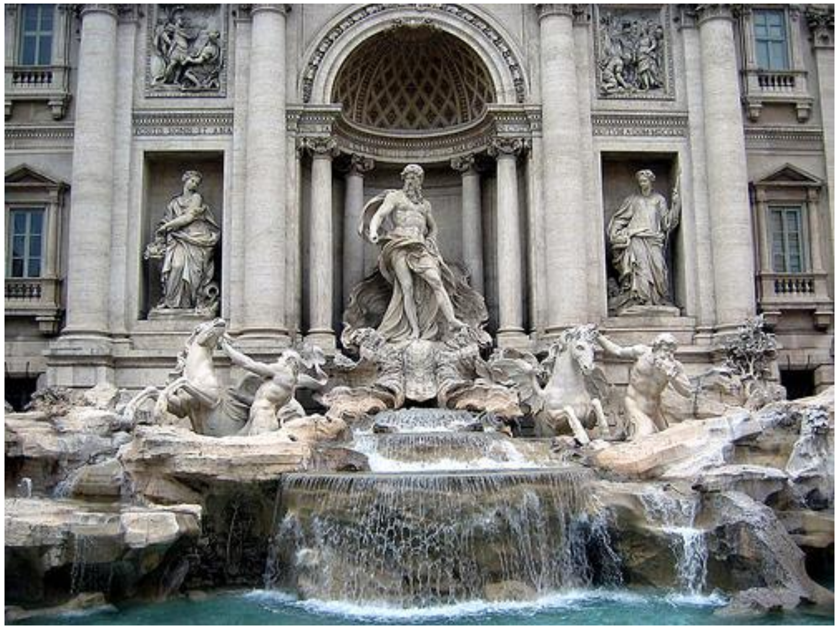


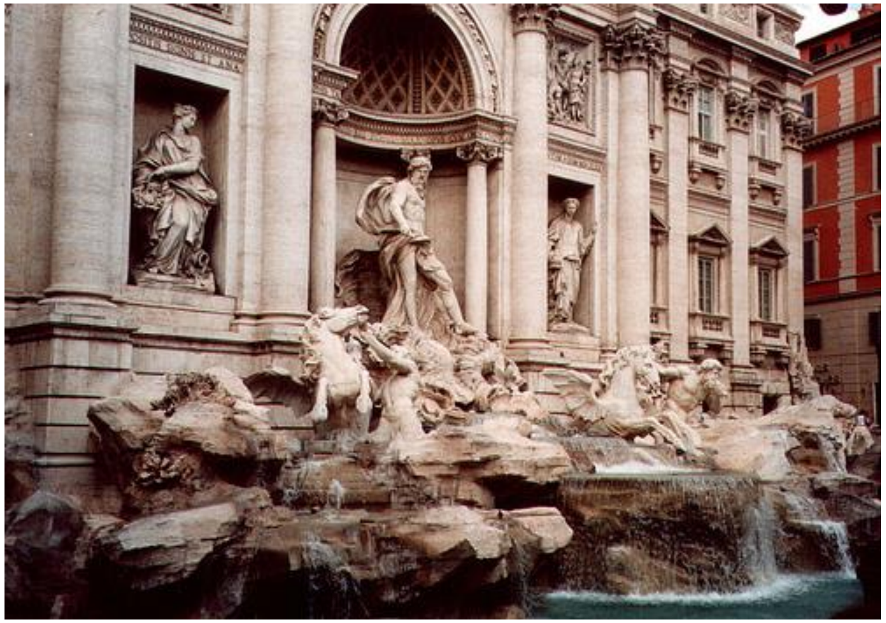
Application: Image Matching


by Diva Sian

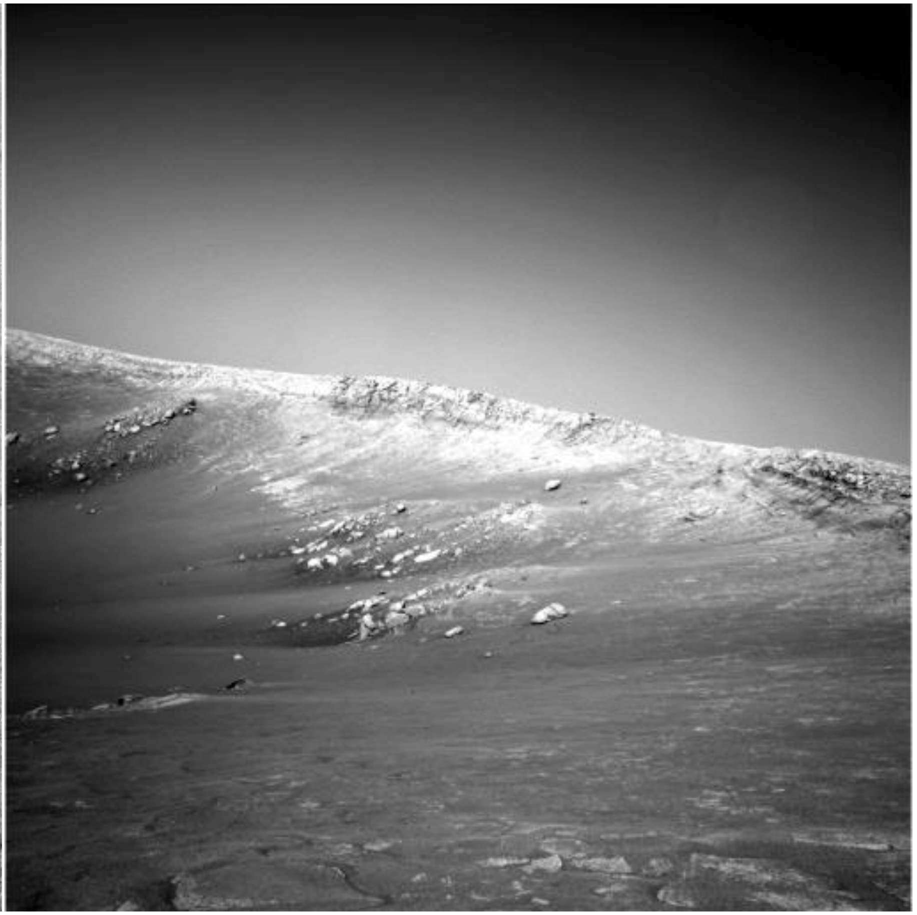

by swashford

Harder Case


by Diva Sian

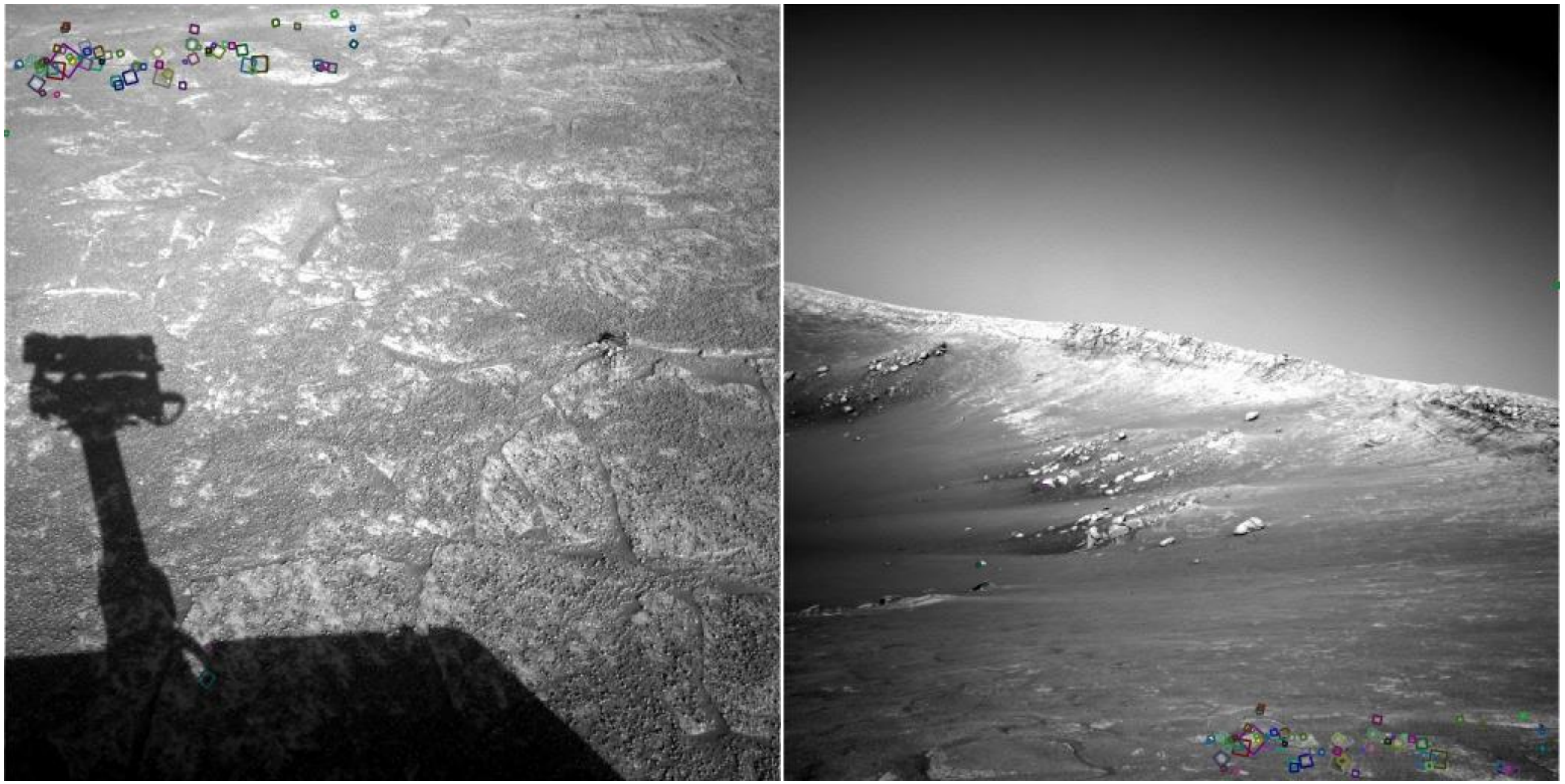

by scgbt

Harder Still?



NASA Mars Rover images

Answer Below (Look for tiny colored squares)

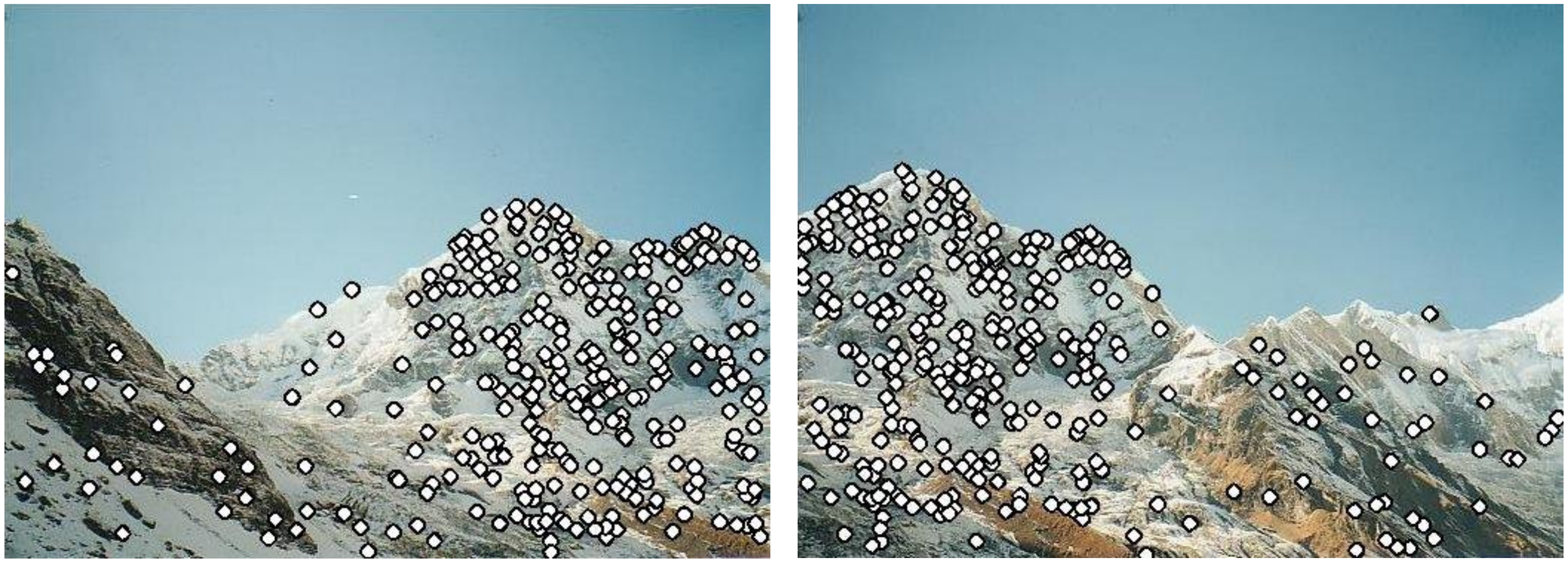


NASA Mars Rover images
with SIFT feature matches
(Figure by Noah Snavely)

Application: Image Stitching

Slide credit: Darya Frolova, Denis Simakov

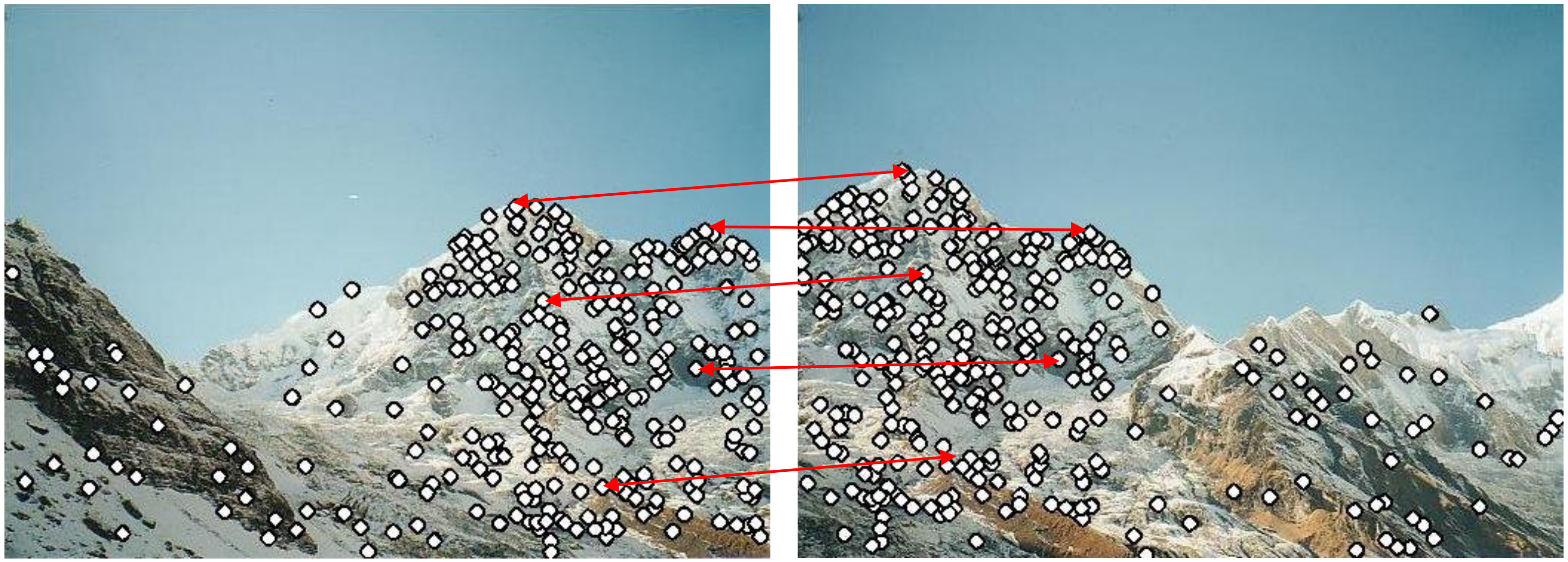
Application: Image Stitching



- **Procedure:**
  - **Detect feature points in both images**

RWTHAACHEN UNIVERSITY

Application: Image Stitching

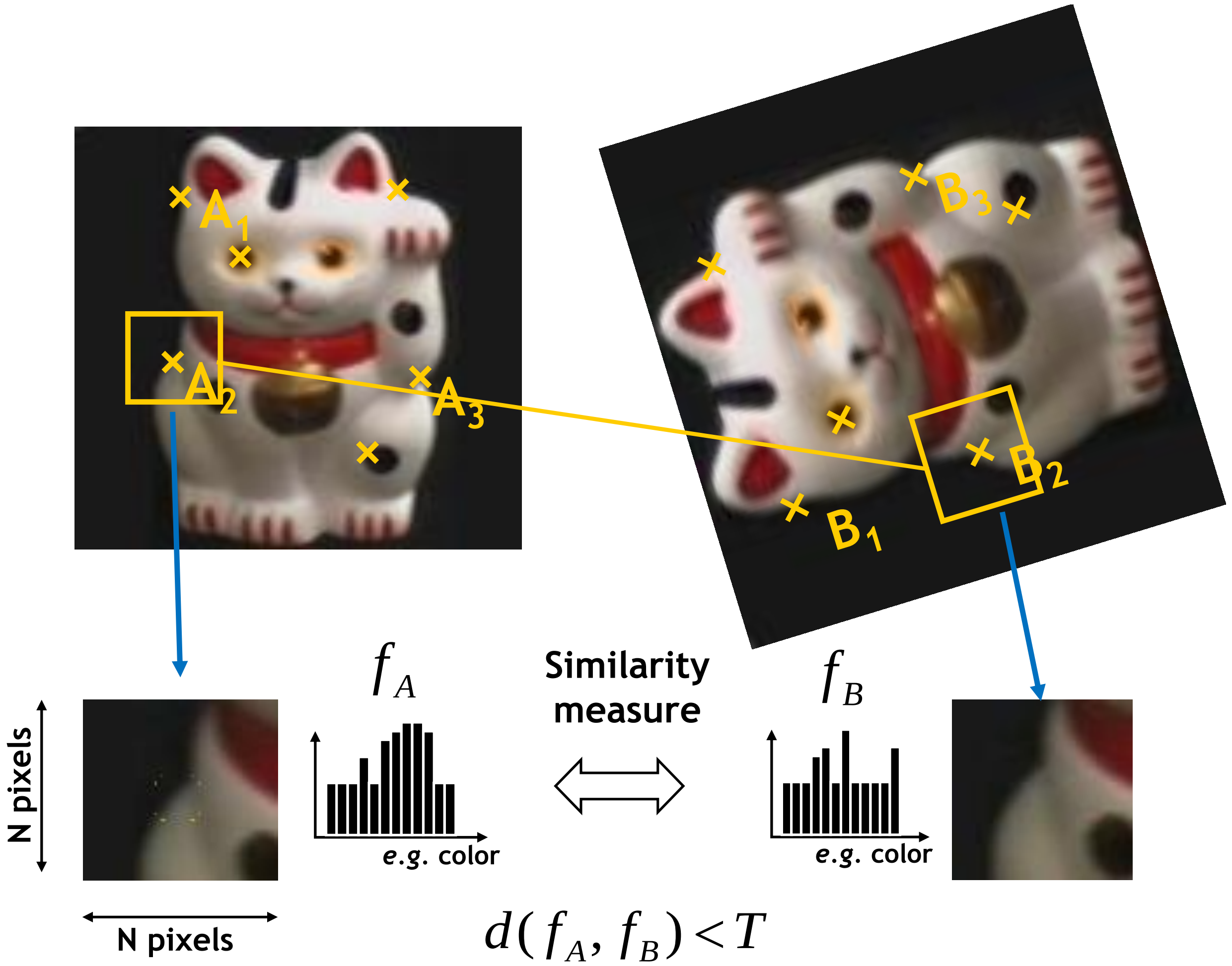


- **Procedure:**
  - **Detect feature points in both images**
  - **Find corresponding pairs**

Application: Image Stitching

- **Procedure:**
  - Detect feature points in both images
  - Find corresponding pairs
  - Use these pairs to align the images

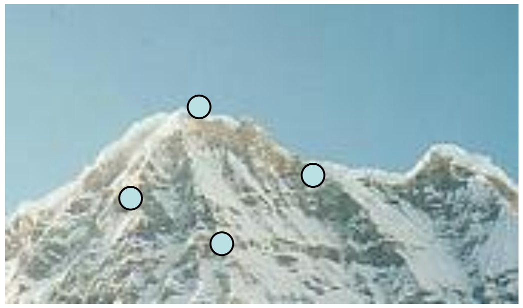
General Approach

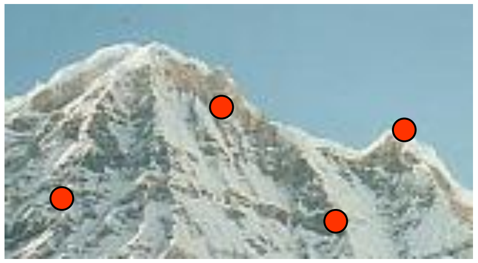


1. Find a set of distinctive key-points
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

Common Requirements

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



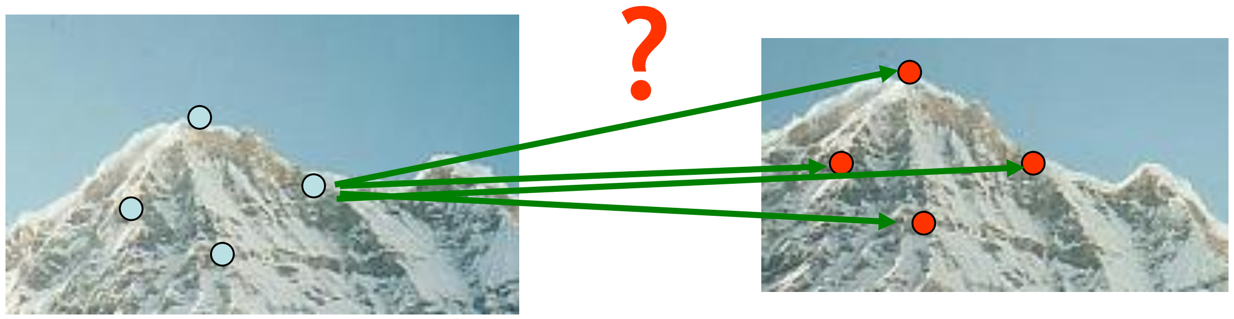


No chance to match!

We need a repeatable detector!

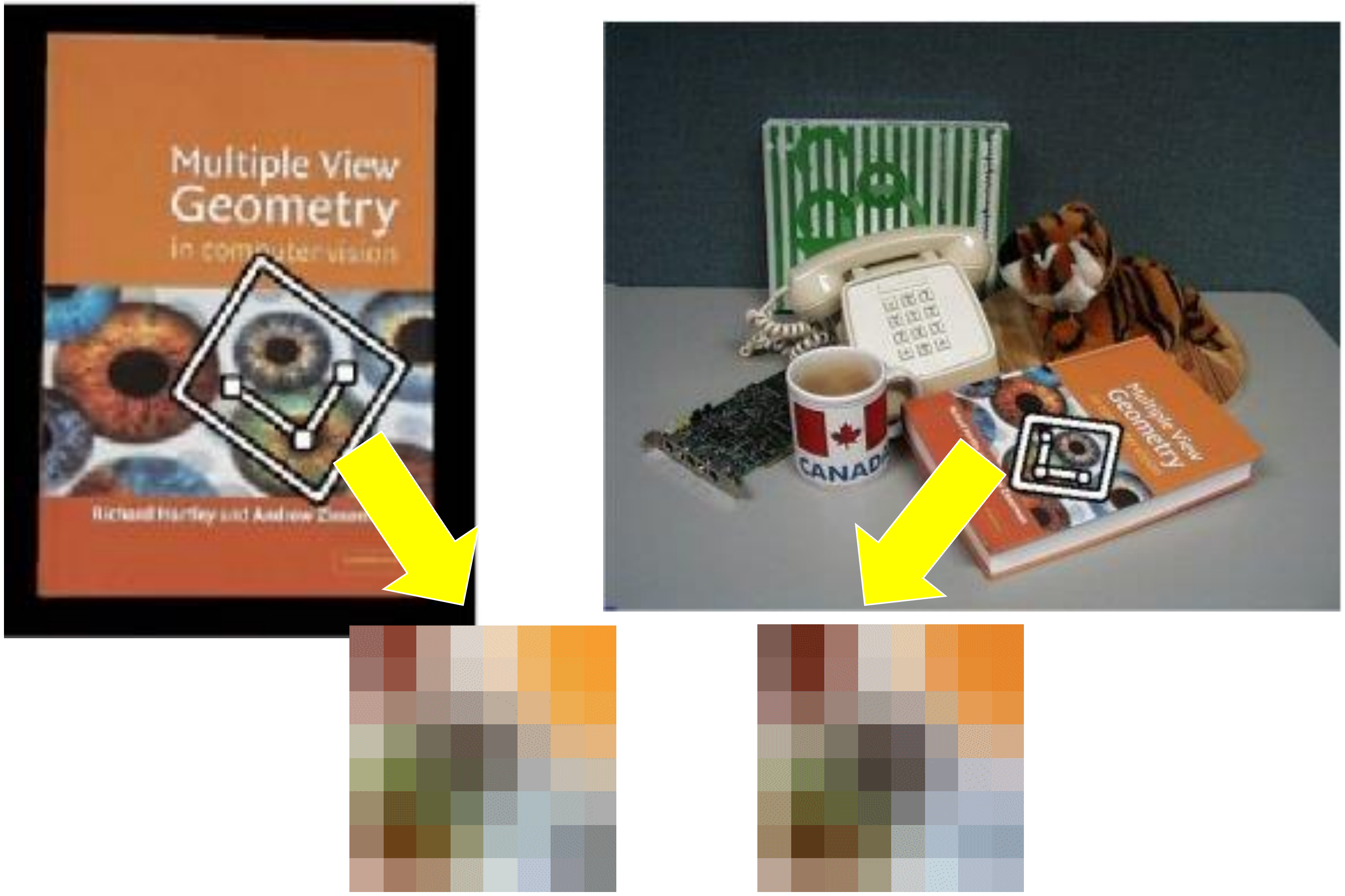
Common Requirements

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



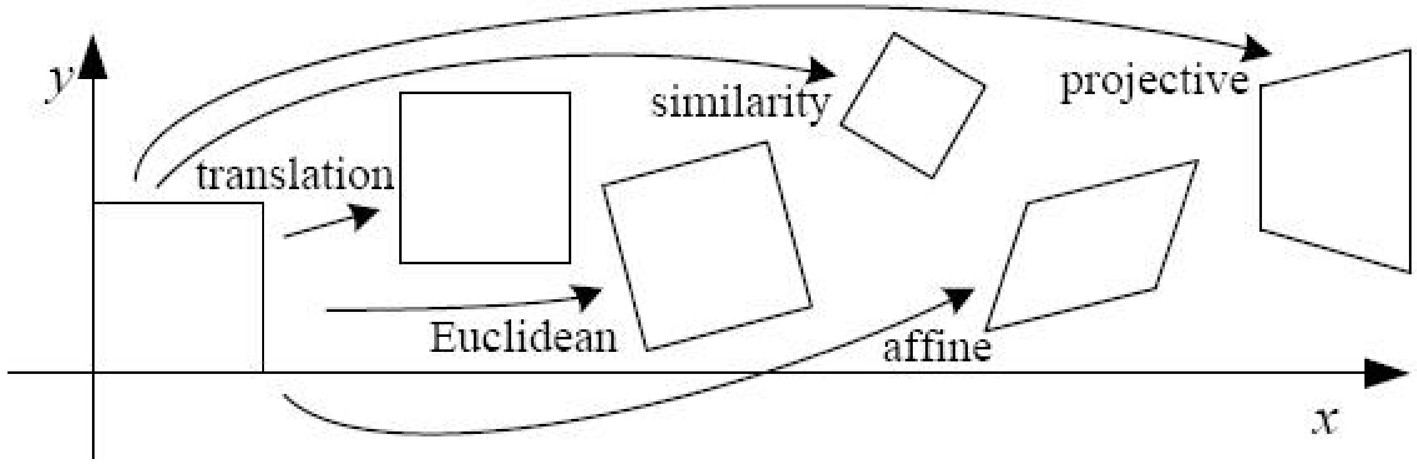
We need a reliable and distinctive descriptor!

Invariance: Geometric Transformations


Multiple View Geometry in computer vision
Richard Hartley and Andrew Zisserman
CANADA

Slide credit: Steve Seitz

Levels of Geometric Invariance



Requirements

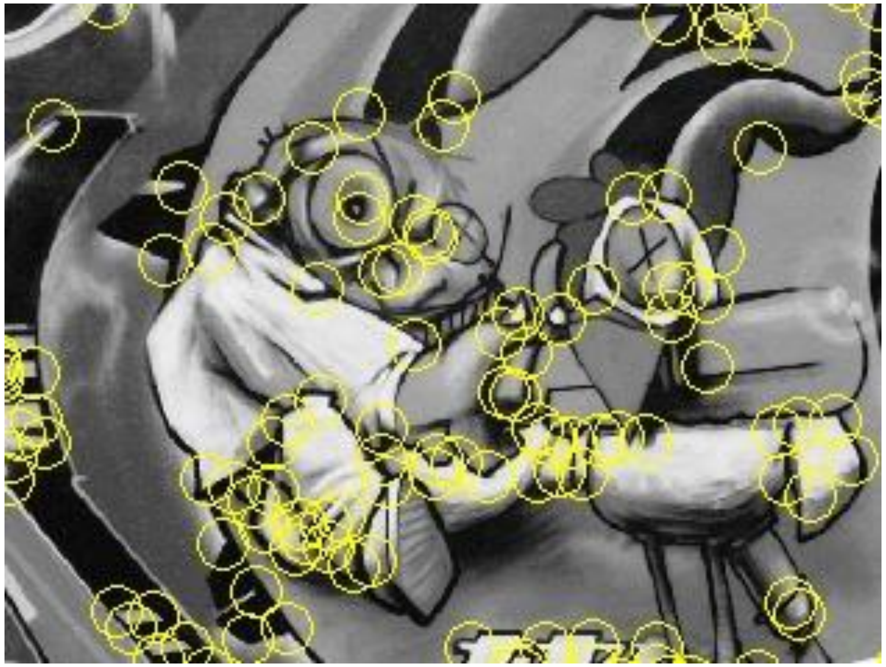
- Region extraction needs to be repeatable and accurate
  - Invariant to translation, rotation, scale changes
  - Robust or covariant to out-of-plane (≈affine) transformations
  - Robust to lighting variations, noise, blur, quantization
- Locality: Features are local, therefore robust to occlusion and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

Many Existing Detectors Available

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

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

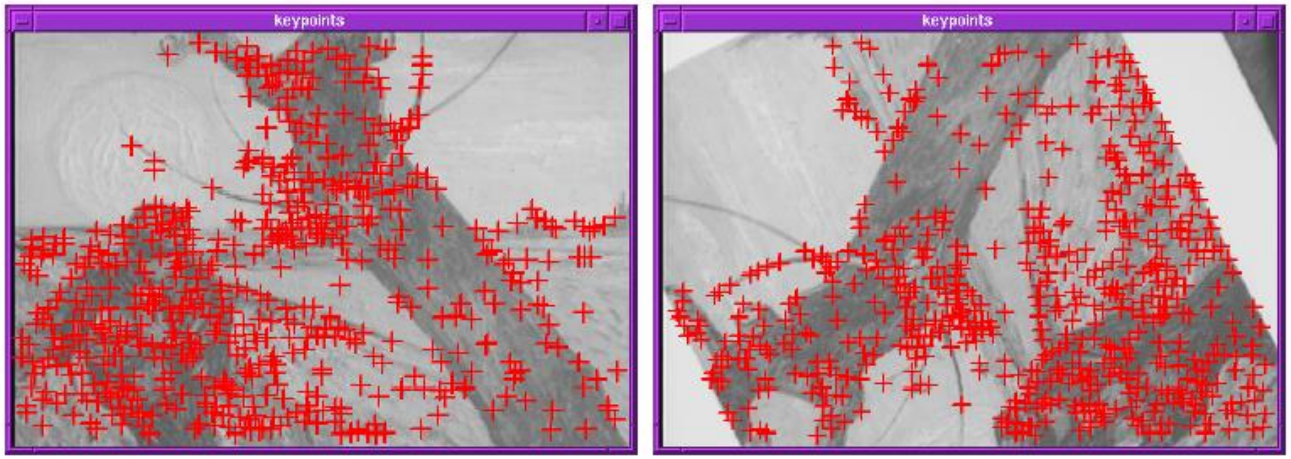
Keypoint Localization



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

$\Rightarrow$ ***Look for two-dimensional signal changes***

Finding Corners

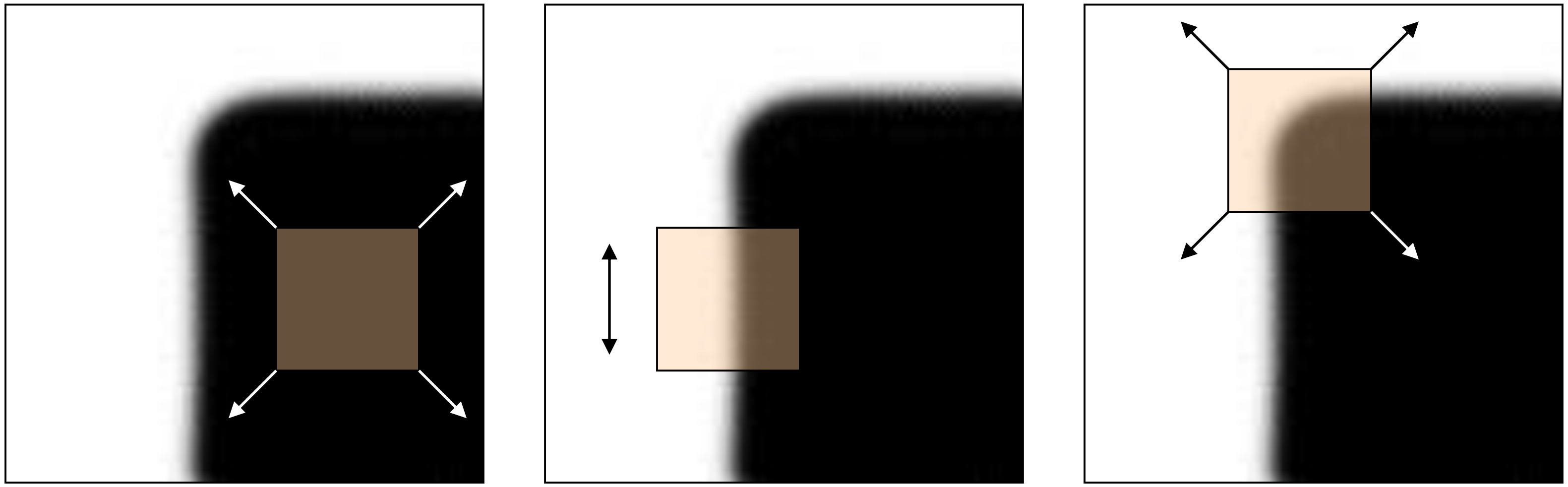


- Key property:
  - In the region around a corner, image gradient has two or more dominant directions
- Corners are *repeatable* and *distinctive*

C.Harris and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*, 1988.

Corners as Distinctive Interest Points

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



"flat" region: no change in all directions

"edge": no change along the edge direction

"corner": significant change in all directions

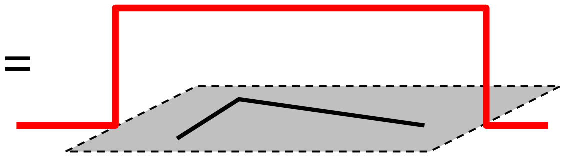
Slide credit: Alexej Efros

Harris Detector Formulation

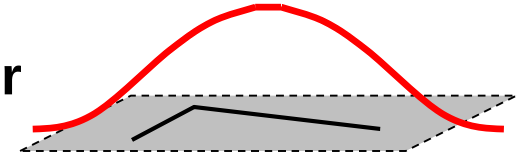
- Change of intensity for the shift [u,v]:

$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{Window function}} \left[\underbrace{I(x+u, y+v)}_{\text{Shifted intensity}} - \underbrace{I(x,y)}_{\text{Intensity}} \right]^2$$

Window function $w(x,y)$ =



or



1 in window, 0 outside

Gaussian

Harris Detector Formulation

- This measure of change can be approximated by:

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

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

$$M = \underbrace{\sum_{x,y}}_{\text{Sum over image region - the area we are checking for corner}} w(x,y) \begin{bmatrix} I_x^2 & \underbrace{I_x I_y}_{\text{Gradient with respect to } x \text{, times gradient with respect to } y} \\ I_x I_y & I_y^2 \end{bmatrix}$$

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

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

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

Harris Detector Formulation

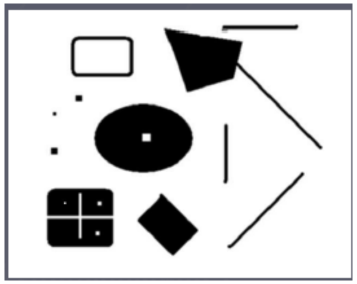
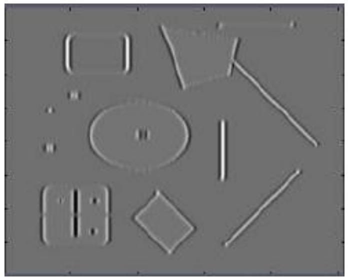
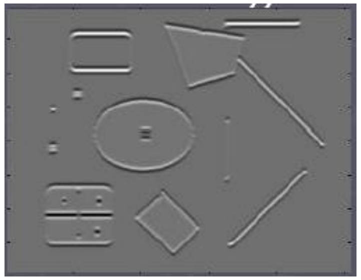
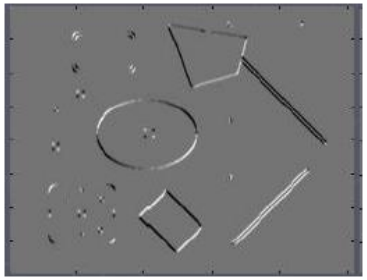


Image I


I_x


I_y


$I_x I_y$

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

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

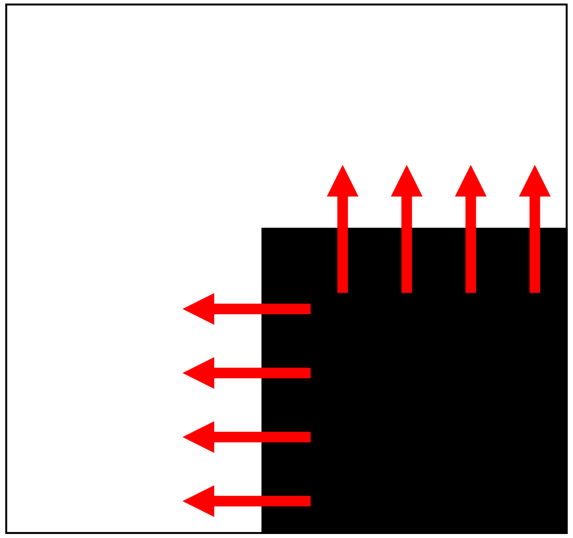
Gradient with respect to x, times gradient with respect to y

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

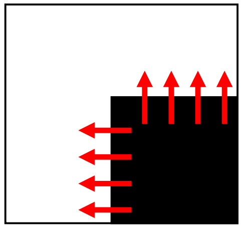
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y]$$

What Does This Matrix Reveal?

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



What Does This Matrix Reveal?



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

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

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

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

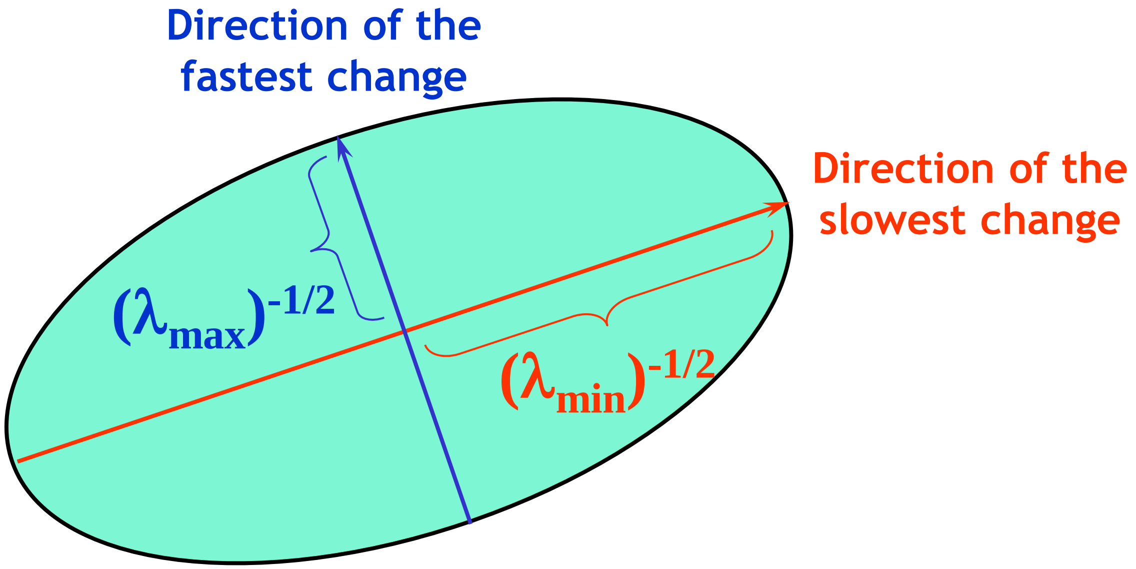
Slide credit: David Jacobs

General Case

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

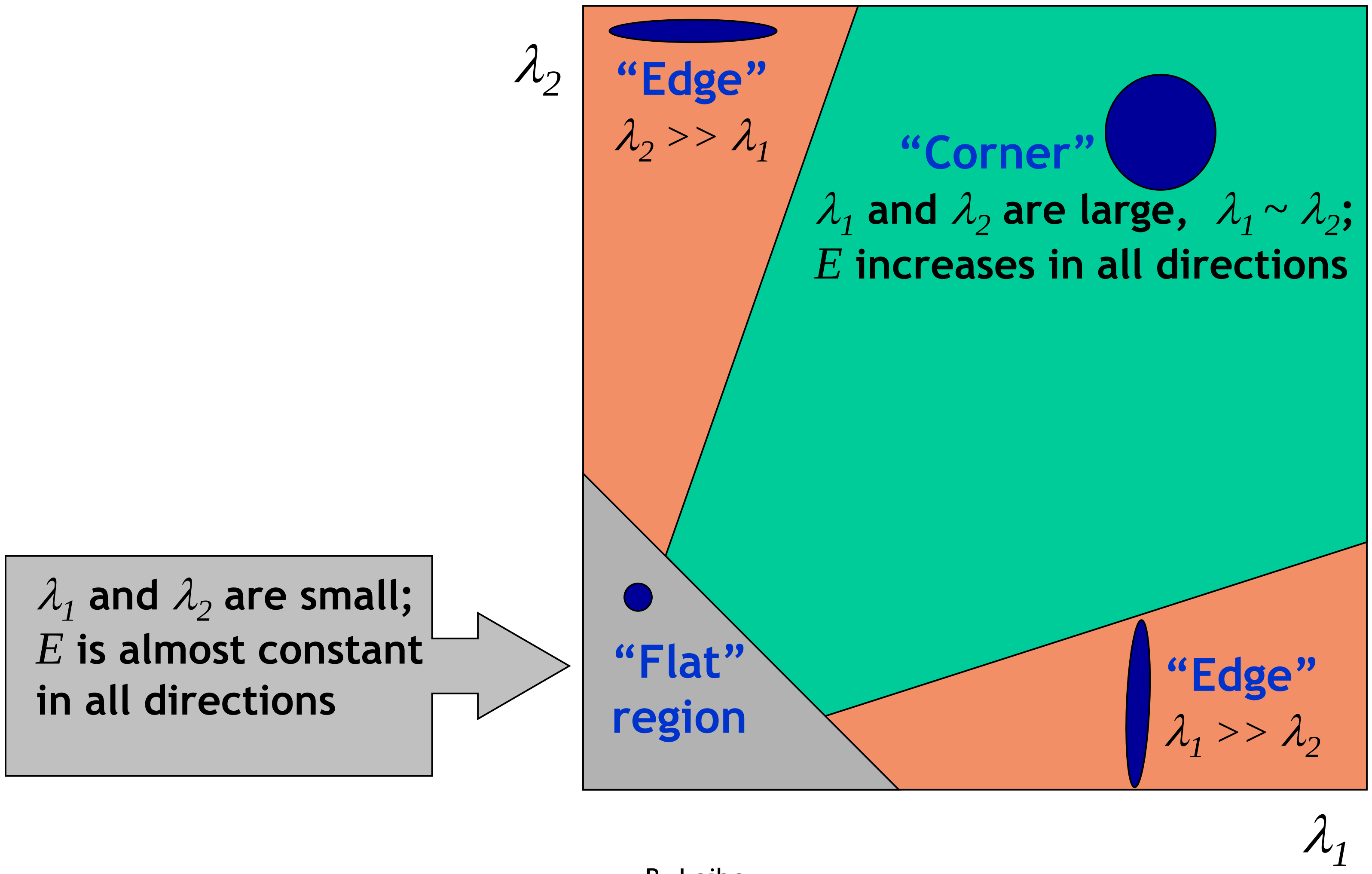
(Eigenvalue decomposition)

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


Direction of the fastest change
Direction of the slowest change
$(\lambda_{max})^{-1/2}$
$(\lambda_{min})^{-1/2}$

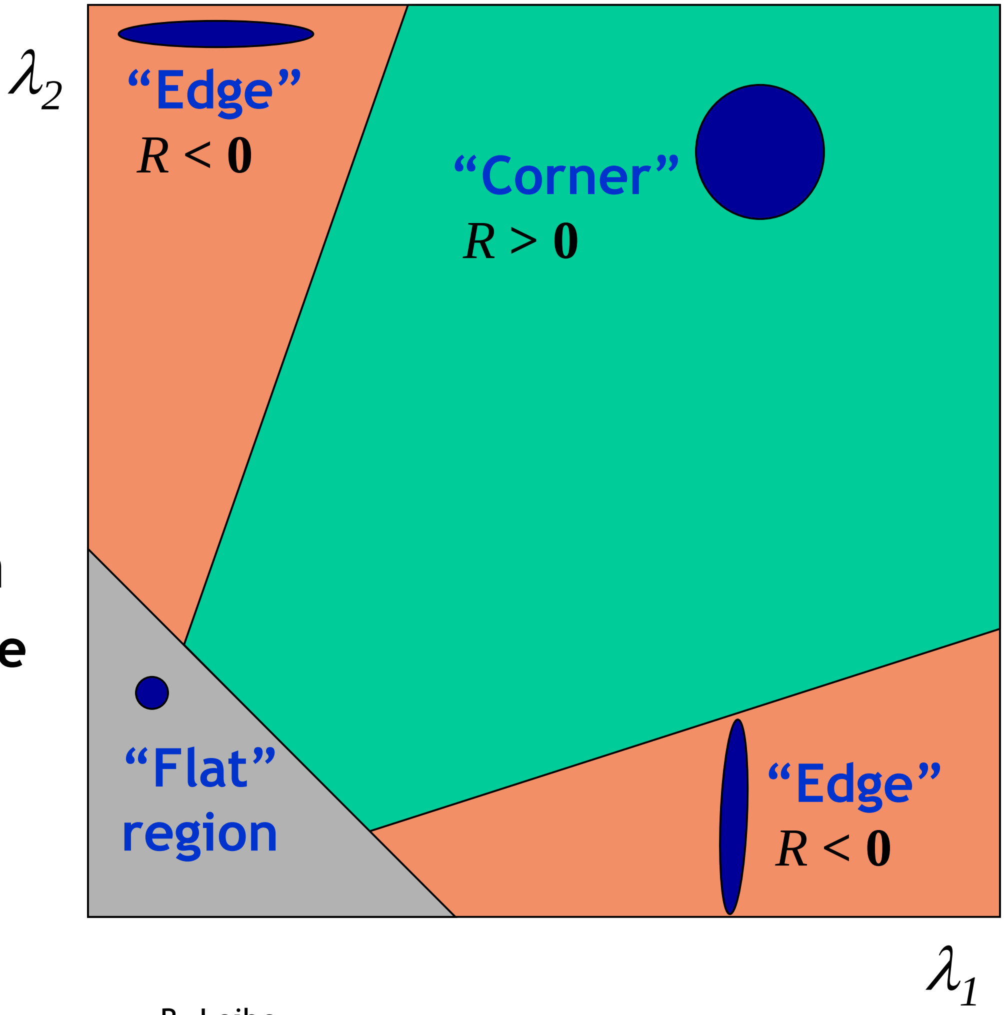
Interpreting the Eigenvalues

- Classification of image points using eigenvalues of M:


λ_2
"Edge"
$\lambda_2 >> \lambda_1$
"Corner"
λ_1 and λ_2 are large, $\lambda_1 \sim \lambda_2$;
E increases in all directions
λ_1 and λ_2 are small;
E is almost constant in all directions
"Flat" region
"Edge"
$\lambda_1 >> \lambda_2$
λ_1

Corner Response Function

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



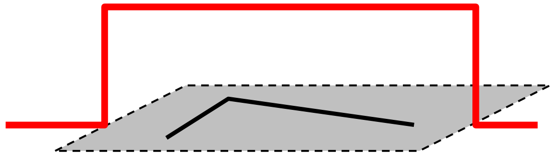
- **Fast approximation**
  - Avoid computing the eigenvalues
  - α: constant (0.04 to 0.06)

Window Function $w(x,y)$

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

- Option 1: uniform window
  - Sum over square window

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

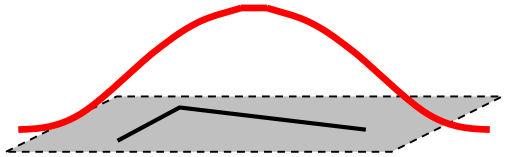


1 in window, 0 outside

  - Problem: not rotation invariant

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

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

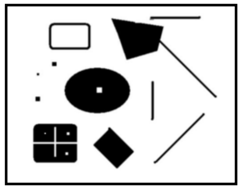


Gaussian

  - Result is rotation invariant

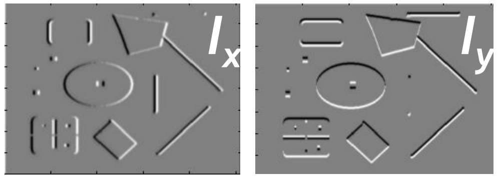
Summary: Harris Detector [Harris88]

- Compute second moment matrix (autocorrelation matrix)

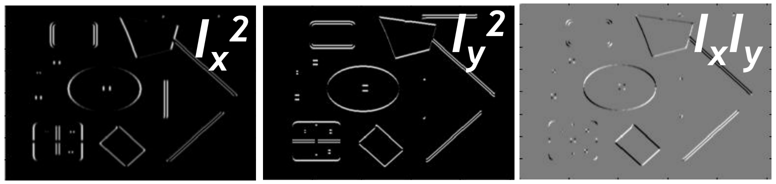


$$M(\sigma_I,\sigma_D)=g(\sigma_I)*\begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

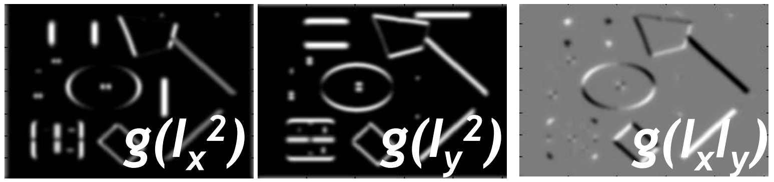
1. Image derivatives



2. Square of derivatives

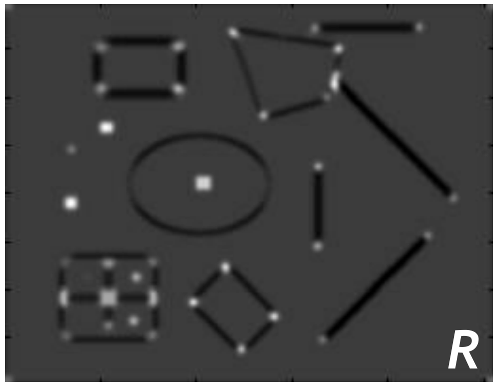


3. Gaussian filter $g(\sigma_I)$



4. Cornerness function - two strong eigenvalues

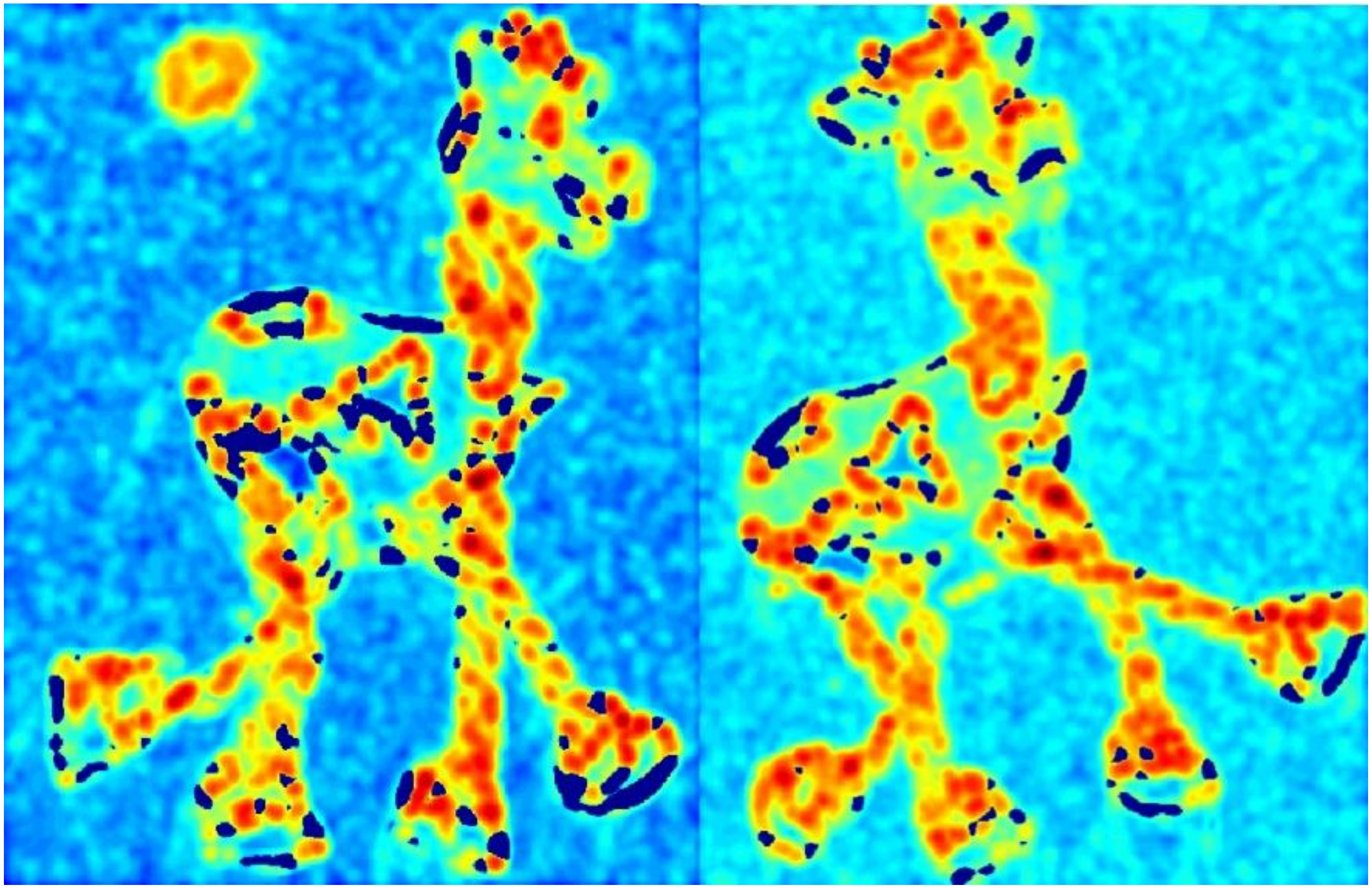
$$R=\det[M(\sigma_I,\sigma_D)]-\alpha[\operatorname{trace}(M(\sigma_I,\sigma_D))]^2$$
$$=g(I_x^2)g(I_y^2)-[g(I_xI_y)]^2-\alpha[g(I_x^2)+g(I_y^2)]^2$$



5. Perform non-maximum suppression

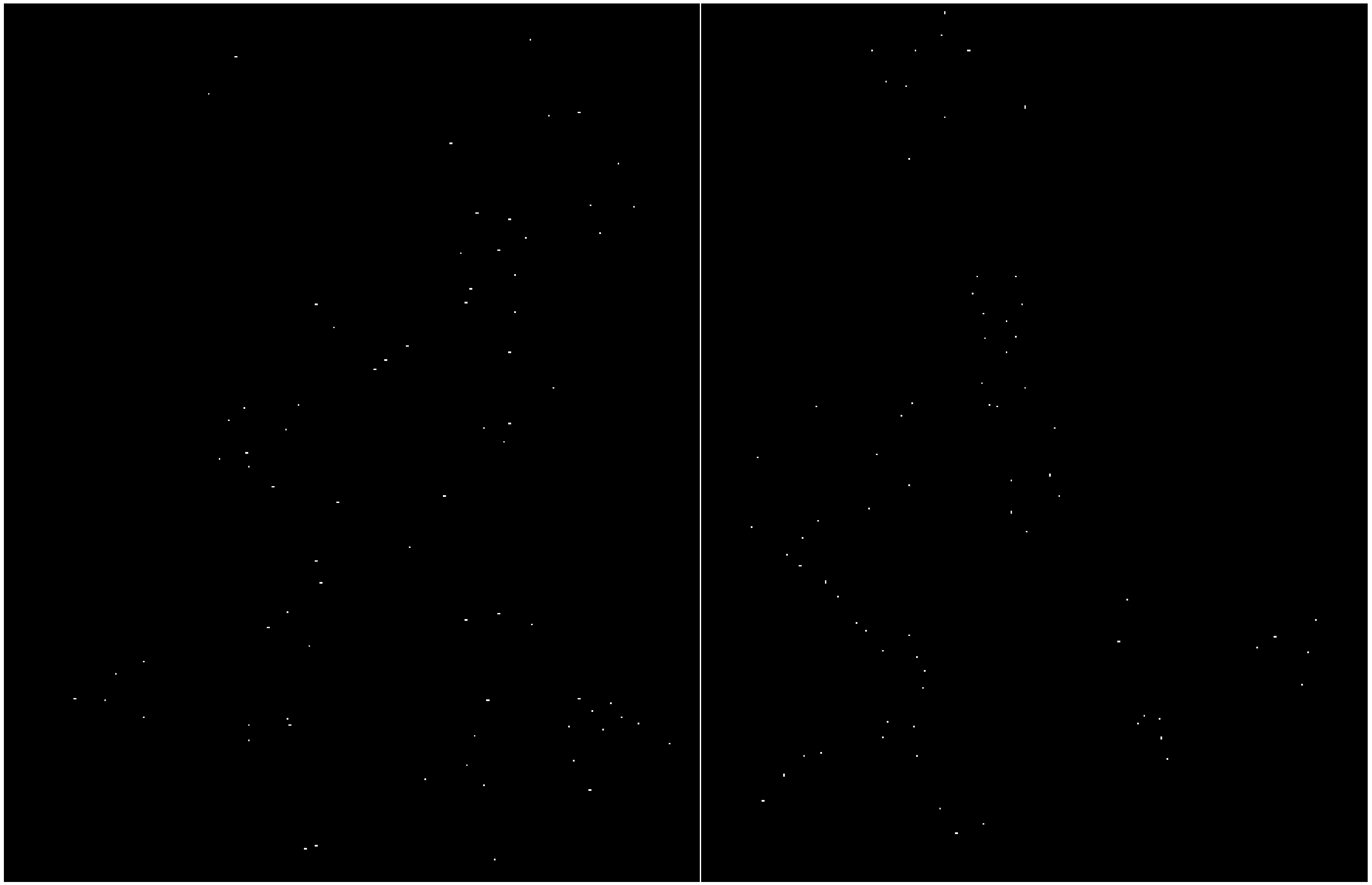
Harris Detector: Workflow

Harris Detector: Workflow



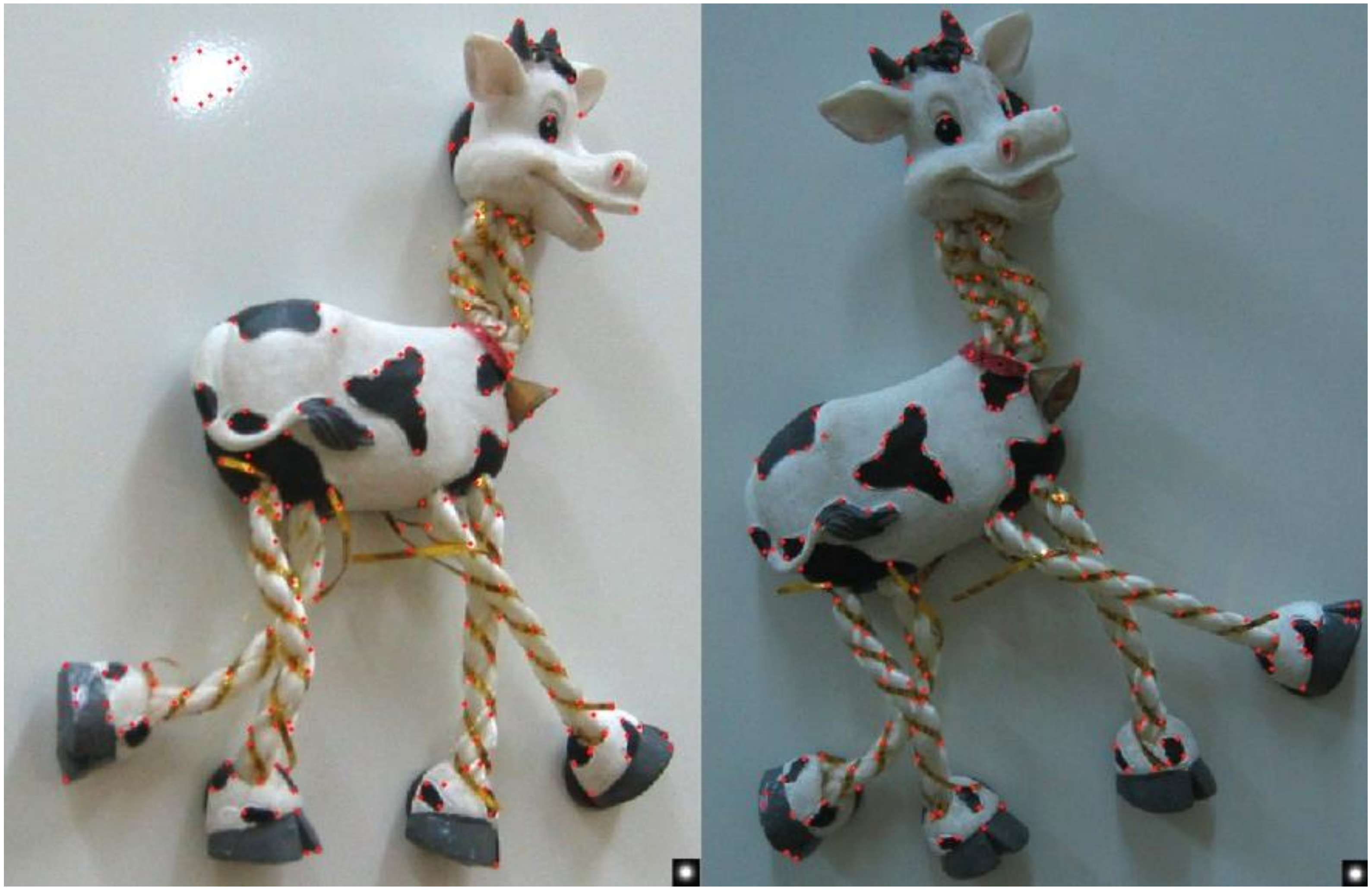
- **Compute corner responses *R***

Harris Detector: Workflow



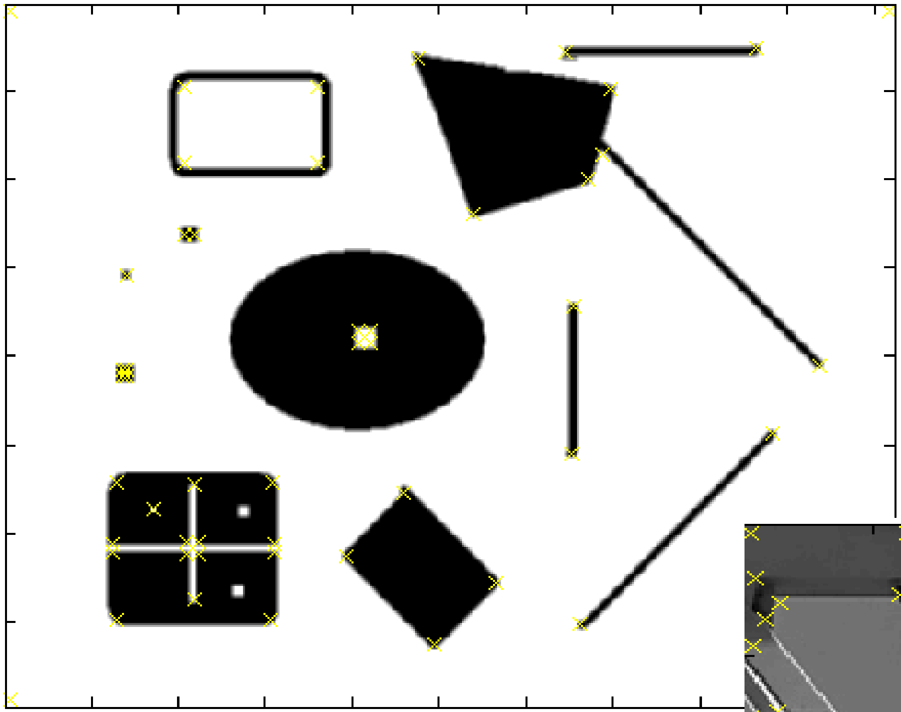
- Take only the local maxima of *R*, where *R* > threshold.

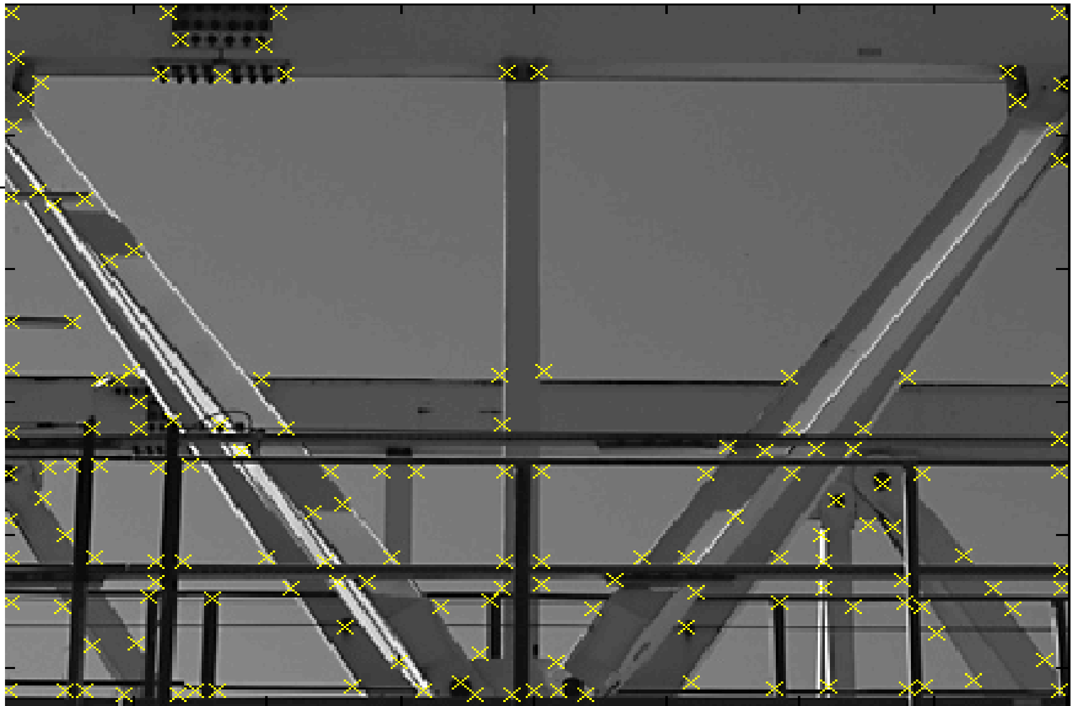
Harris Detector: Workflow



- **Resulting Harris points**

Harris Detector – Responses [Harris88]

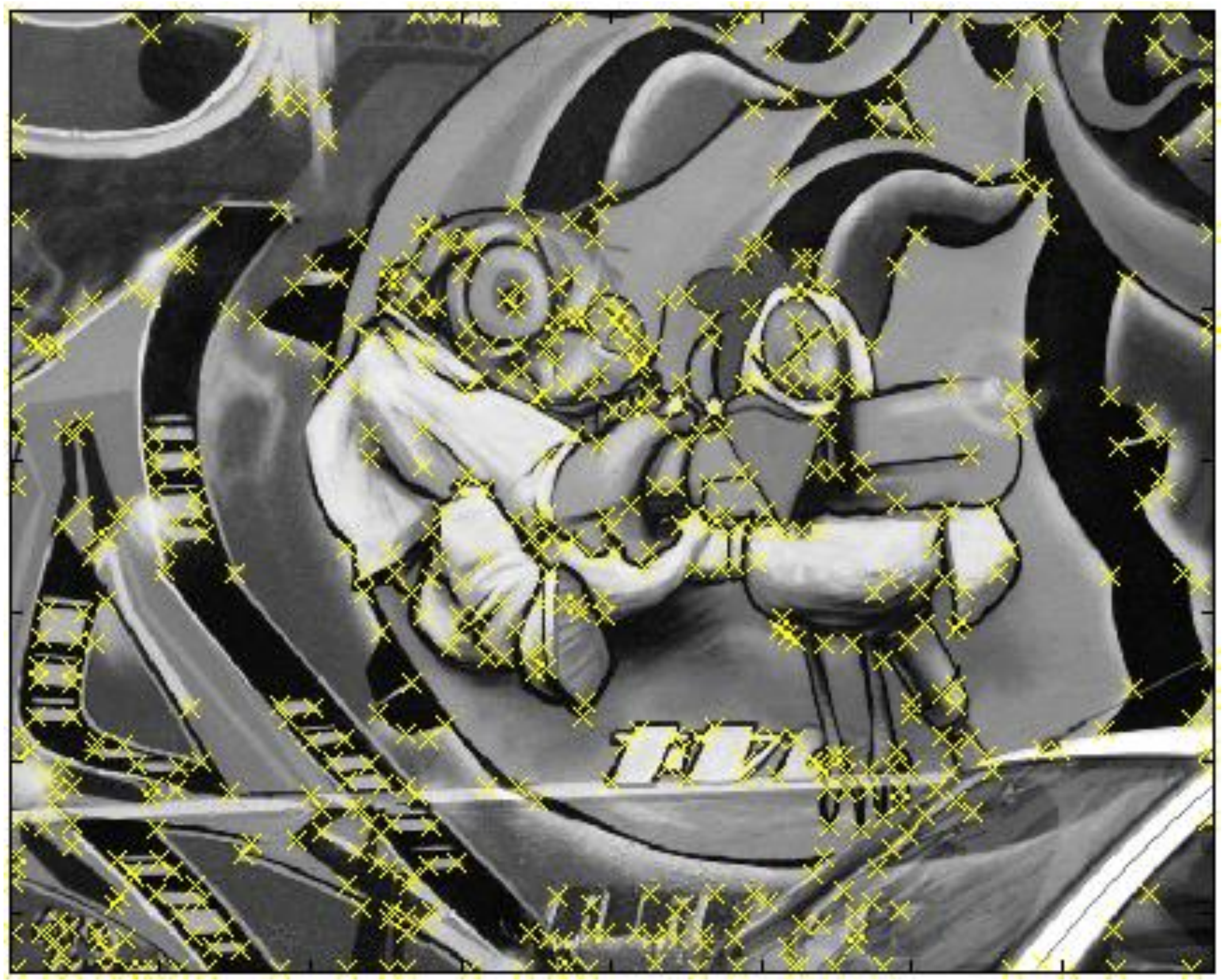




Effect: A very precise corner detector.

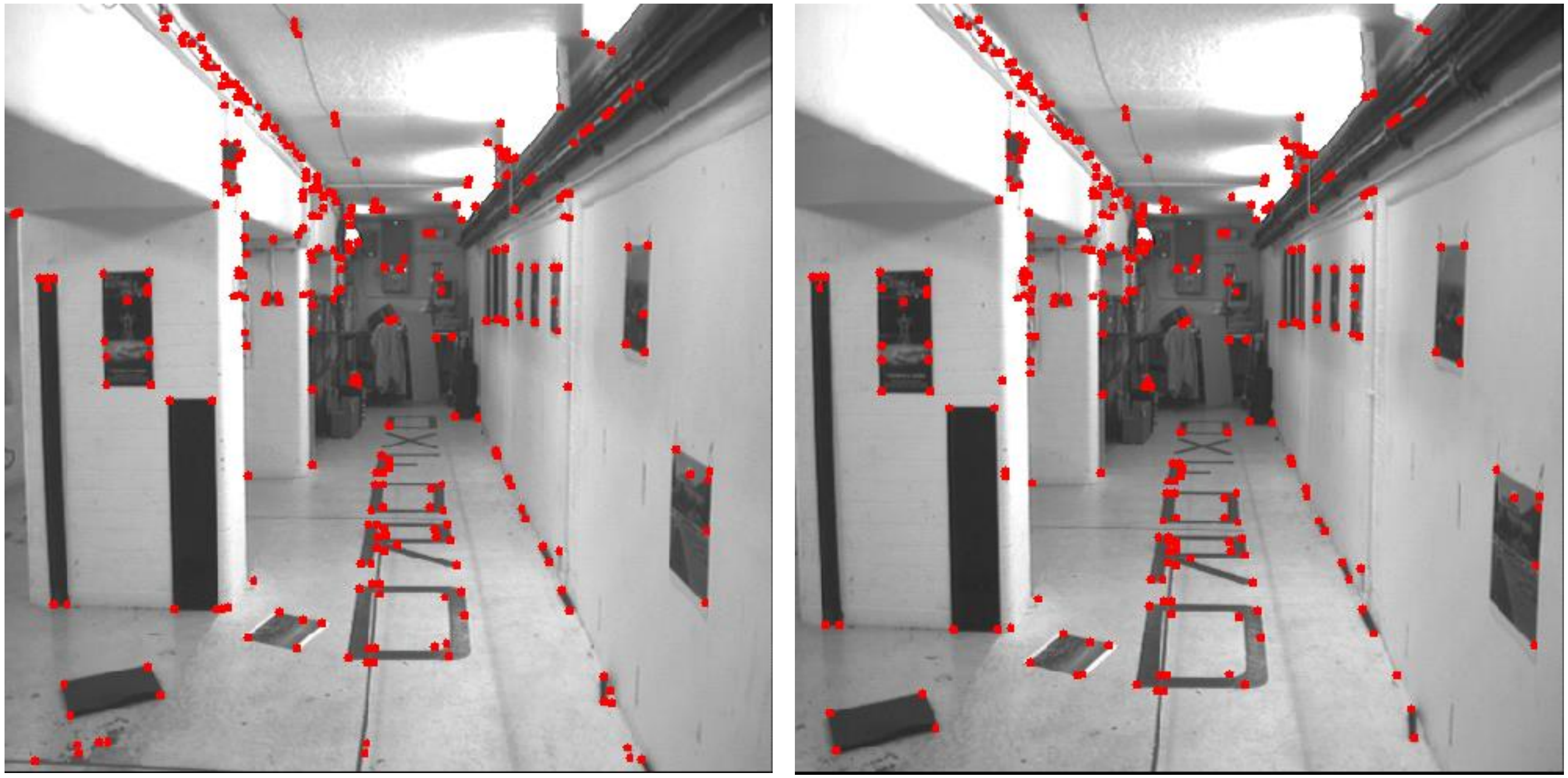
Slide credit: Krystian Mikolajczyk

Harris Detector - Responses [Harris88]



Slide credit: Krystian Mikolajczyk

Harris Detector – Responses [Harris88]

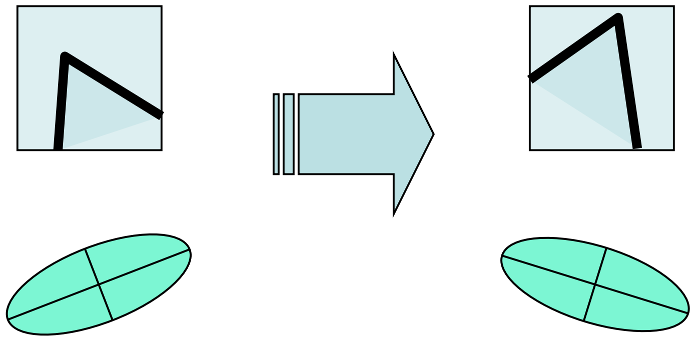


- **Results are well suited for finding stereo correspondences**

Slide credit: Kristen Grauman

Harris Detector: Properties

- Rotation invariance?

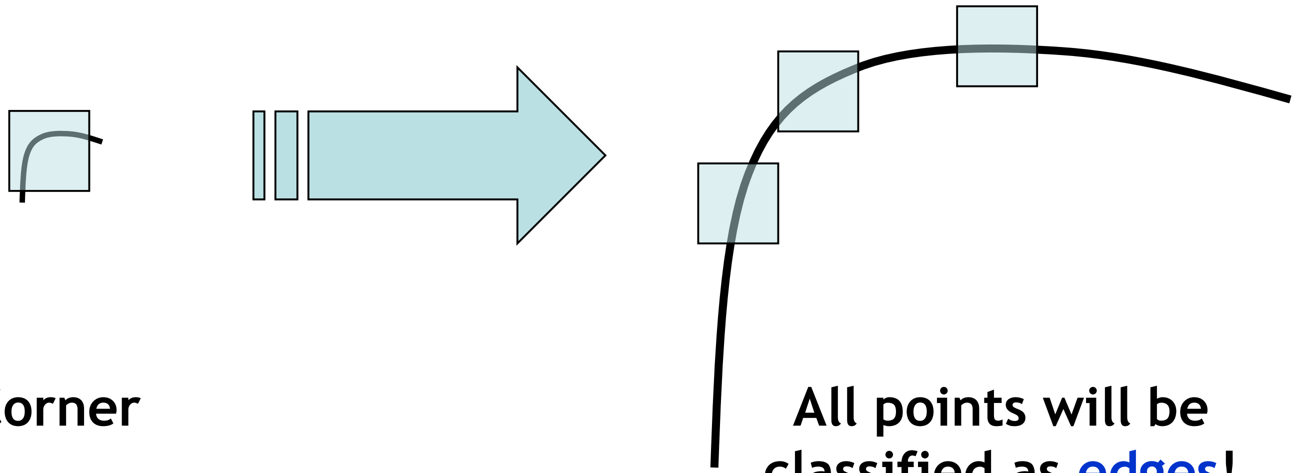


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R is invariant to image rotation

Harris Detector: Properties

- Rotation invariance
- Scale invariance?



Not invariant to image scale!

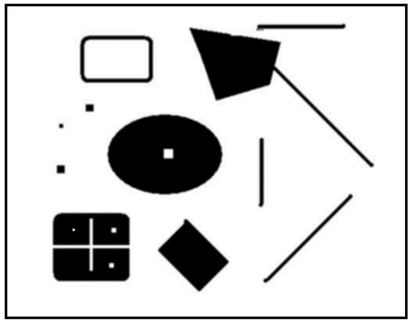
RWTH AACHEN UNIVERSITY

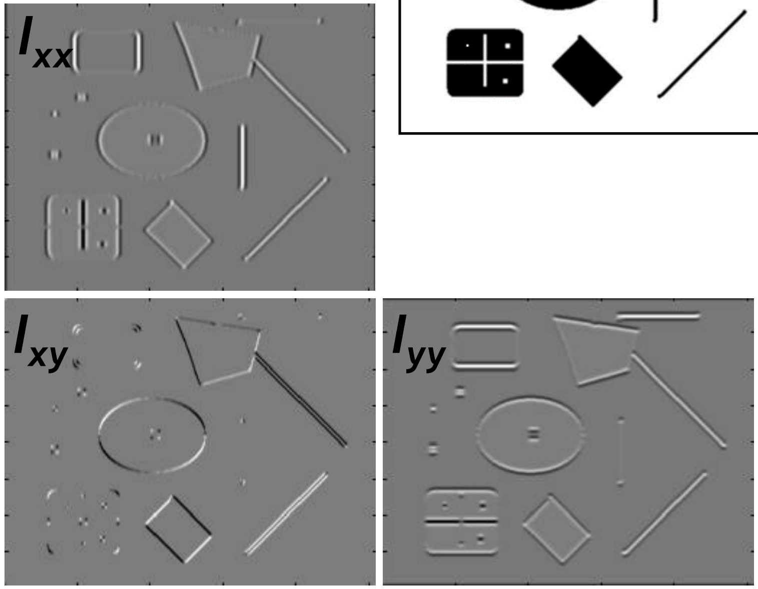
Hessian Detector [Beaudet78]

- **Hessian determinant**

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

Note: these are 2nd derivatives!



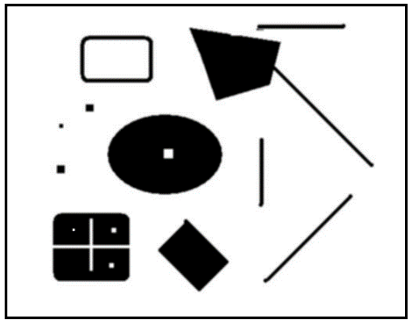


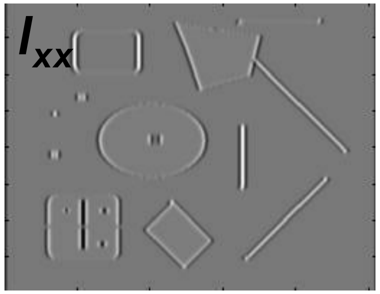
Intuition: Search for strong derivatives in two orthogonal directions

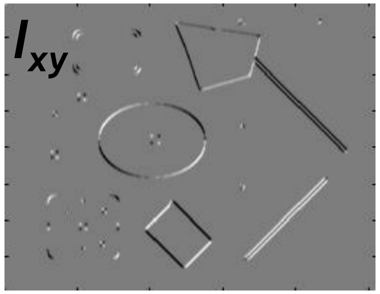
Hessian Detector [Beaudet78]

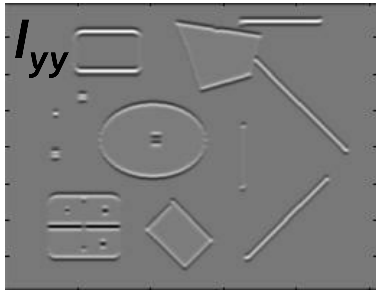
- **Hessian determinant**

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





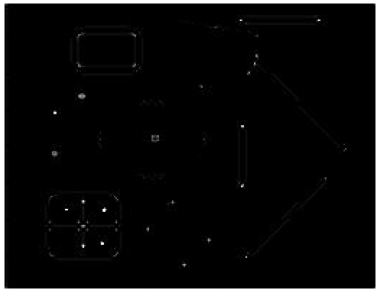




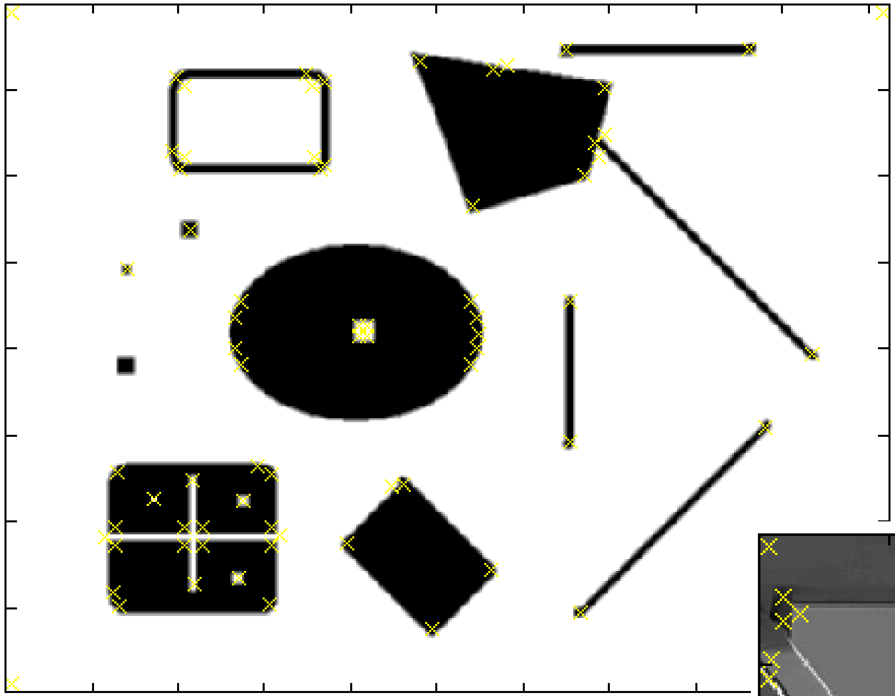
$$\det(Hessian(I)) = I_{xx} I_{yy} - I_{xy}^2$$

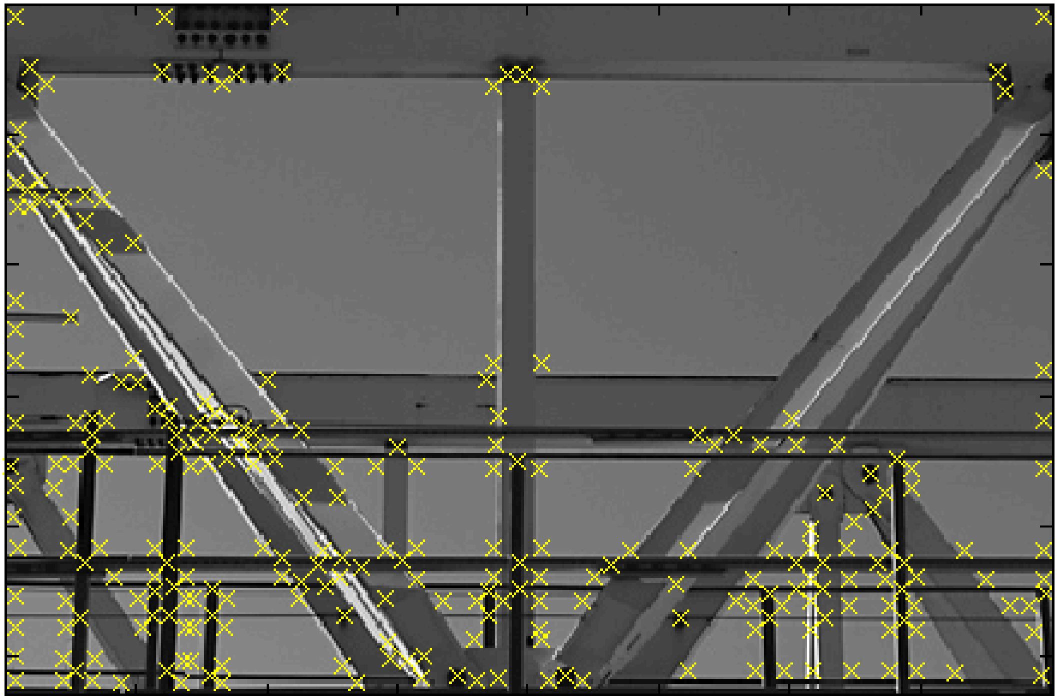
In Matlab:

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



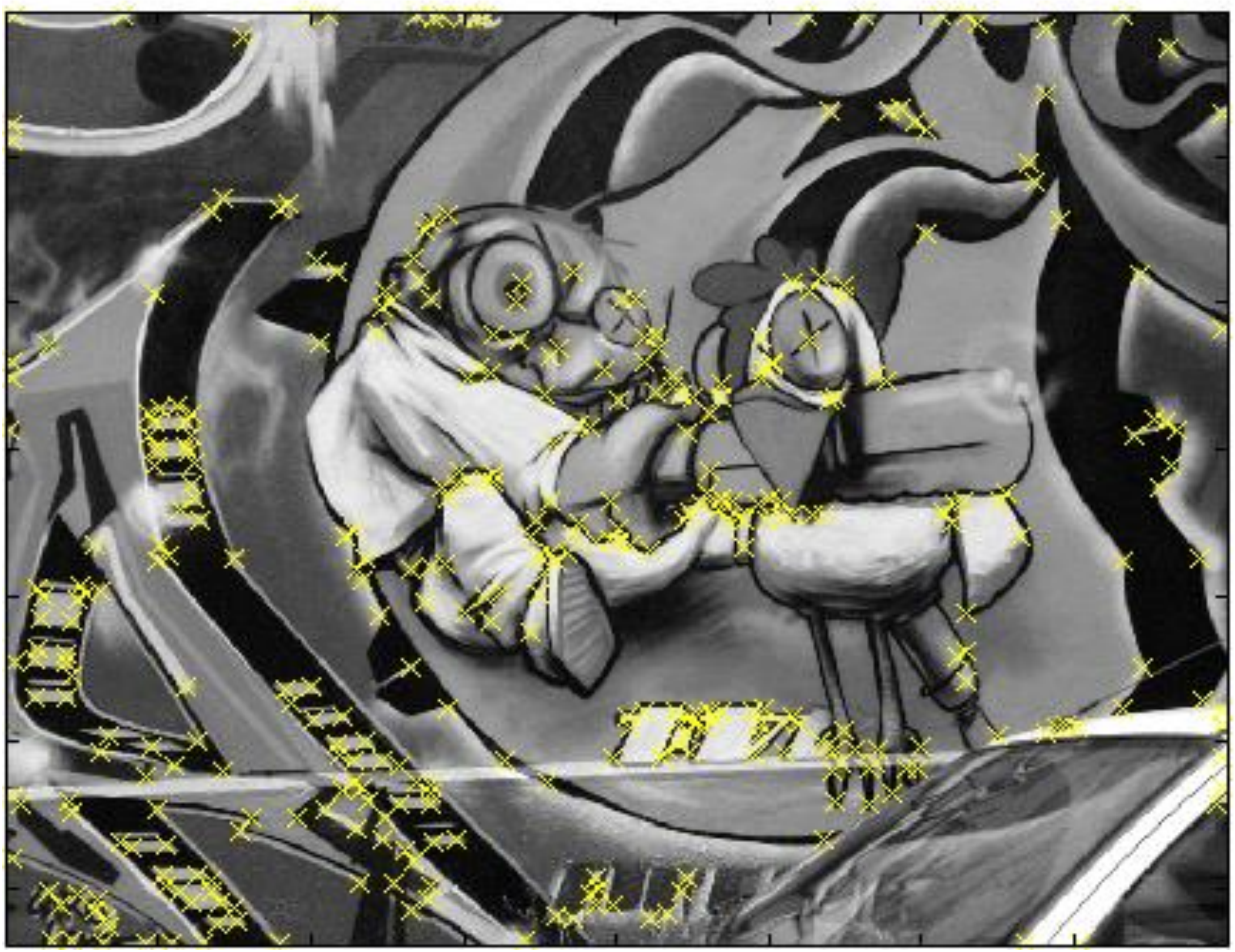
Hessian Detector – Responses [Beaudet78]





Effect: Responses mainly on corners and strongly textured areas.

Hessian Detector – Responses [Beaudet78]



Slide credit: Krystian Mikolajczyk

Topics of This Lecture

- Local Invariant Features
  - Motivation
  - Requirements, Invariances
- Keypoint Localization
  - Harris detector
  - Hessian detector
- Scale Invariant Region Selection
  - Automatic scale selection
  - Laplacian-of-Gaussian detector
  - Difference-of-Gaussian detector
  - Combinations
- Local Descriptors
  - Orientation normalization
  - SIFT

From Points to Regions…

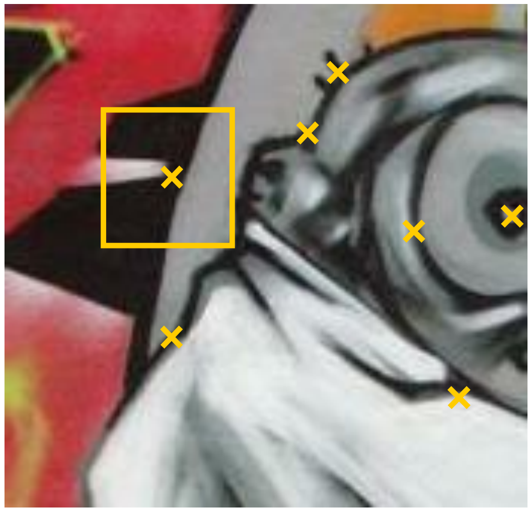
- The Harris and Hessian operators define interest points.
  - Precise localization
  - High repeatability

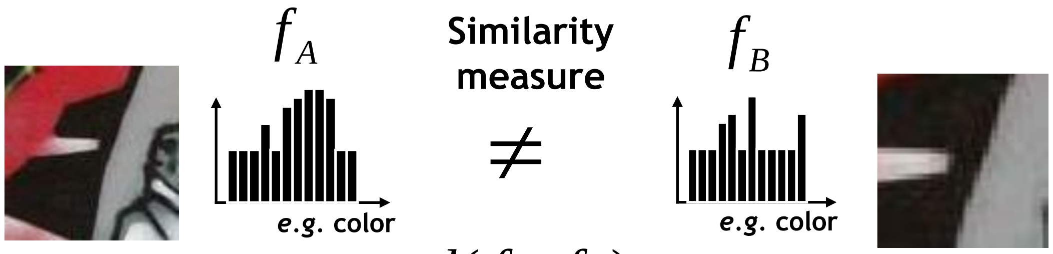
- In order to compare those points, we need to compute a descriptor over a region.
  - How can we define such a region in a scale invariant manner?

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

Naïve Approach: Exhaustive Search

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

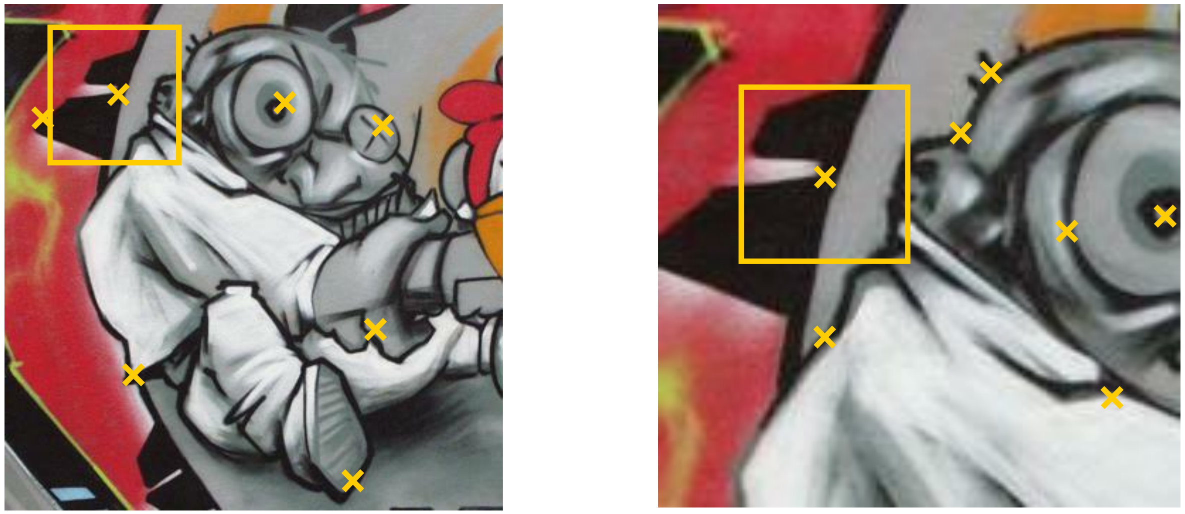


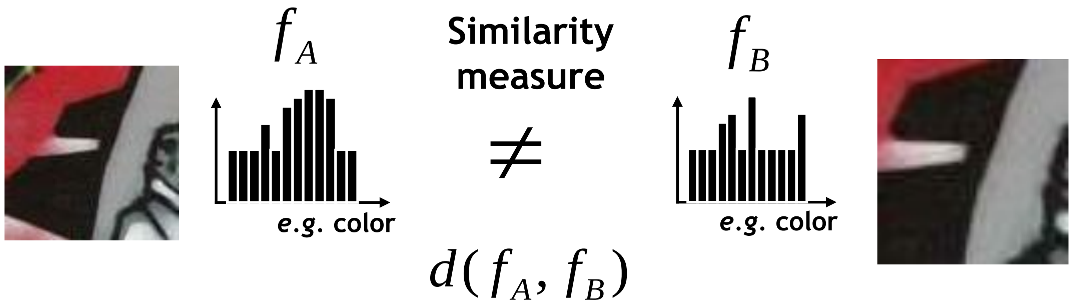


$d(f_A, f_B)$

Naïve Approach: Exhaustive Search

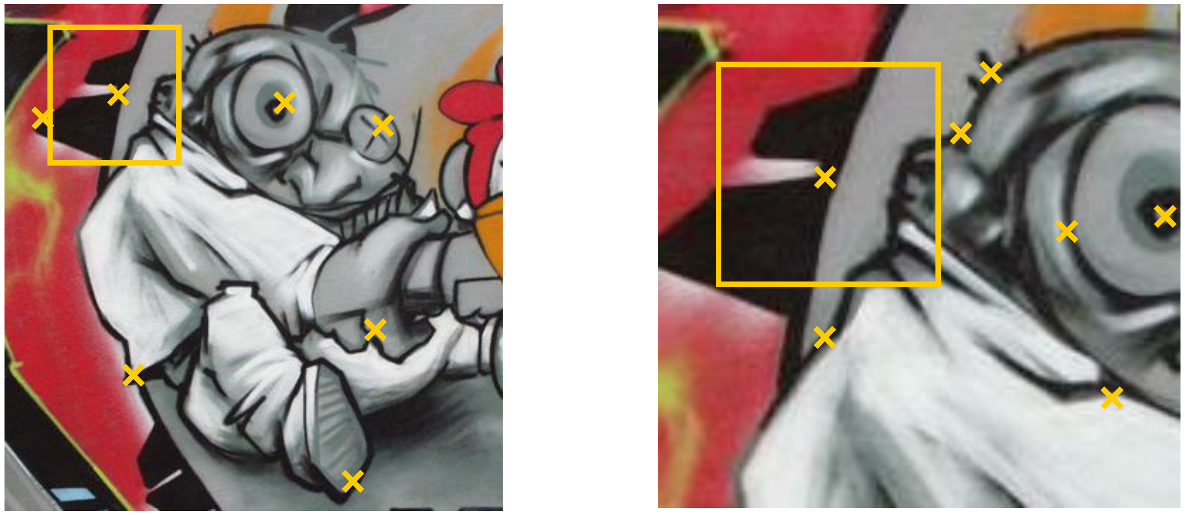
- **Multi-scale procedure**
  - Compare descriptors while varying the patch size

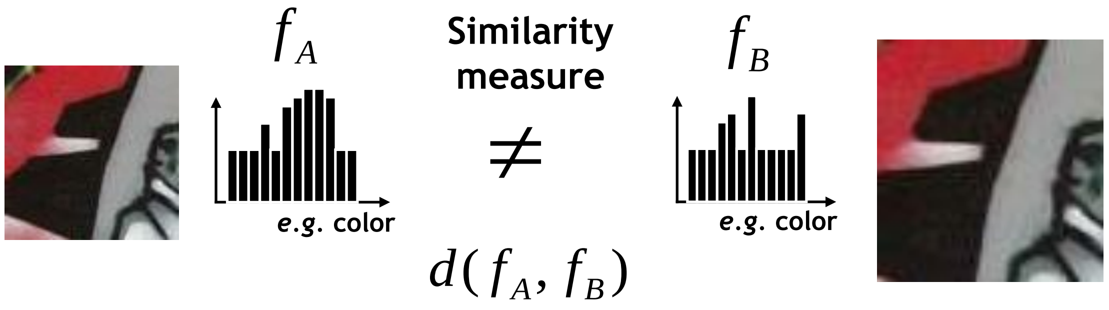




Naïve Approach: Exhaustive Search

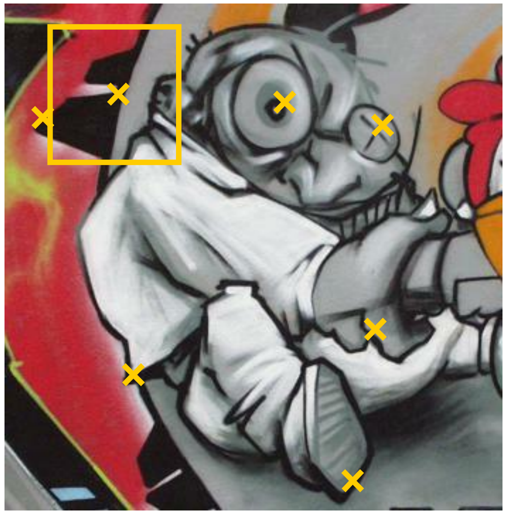
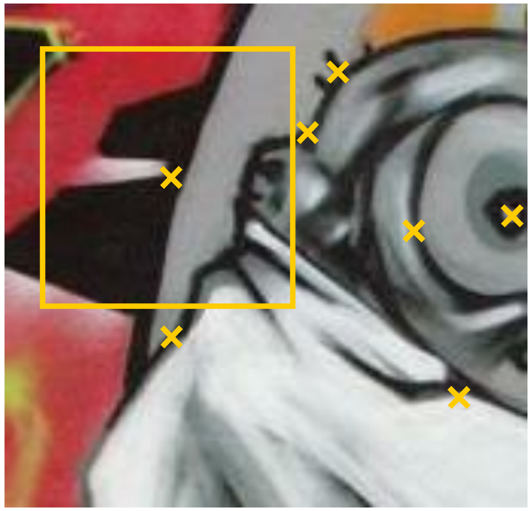
- Multi-scale procedure
  - Compare descriptors while varying the patch size

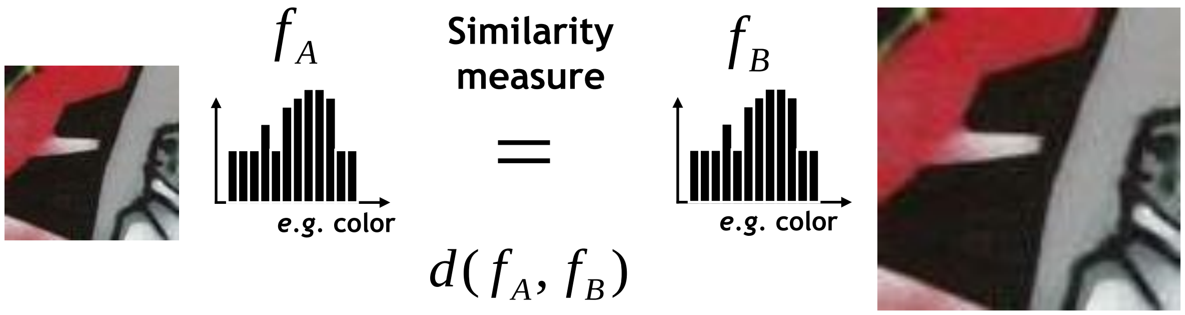




Naïve Approach: Exhaustive Search

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

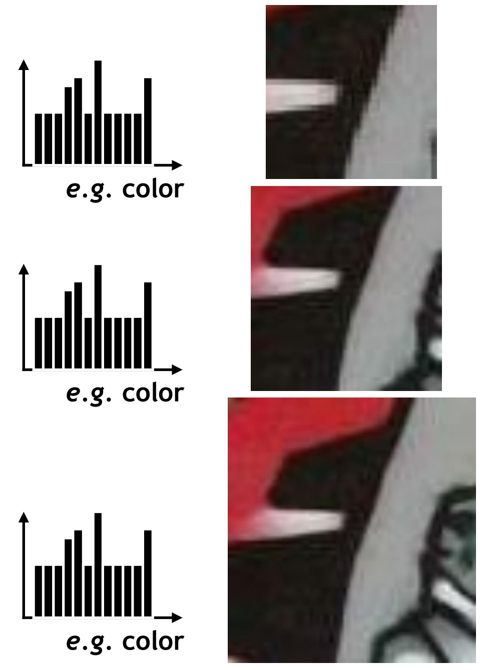



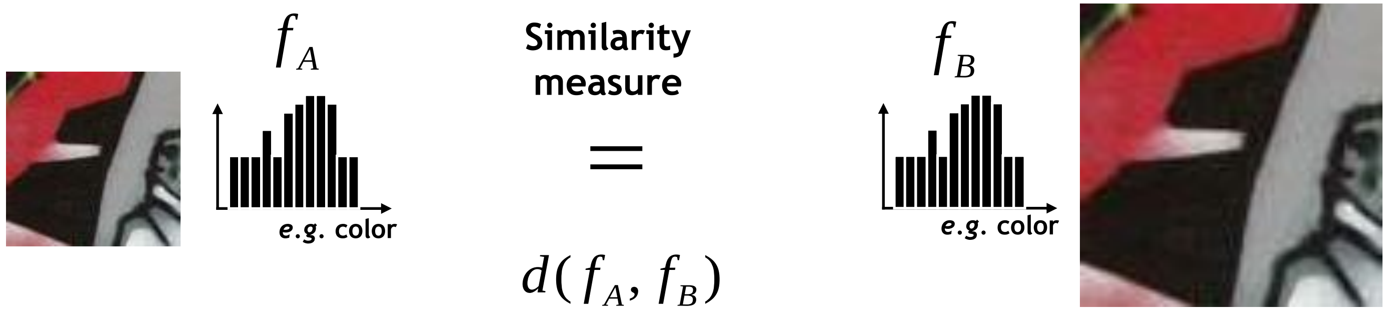


$d(f_A, f_B)$

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



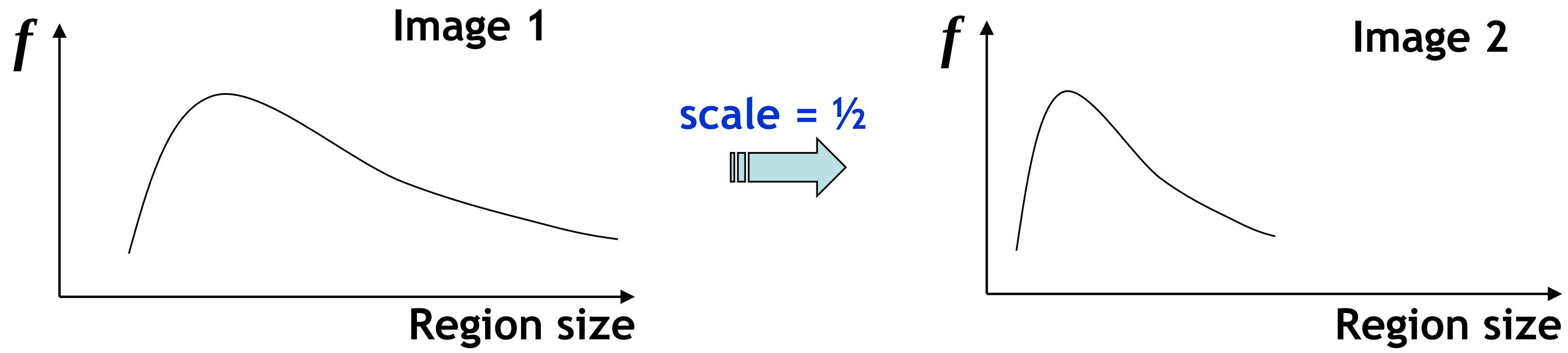


Automatic Scale Selection

- 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.

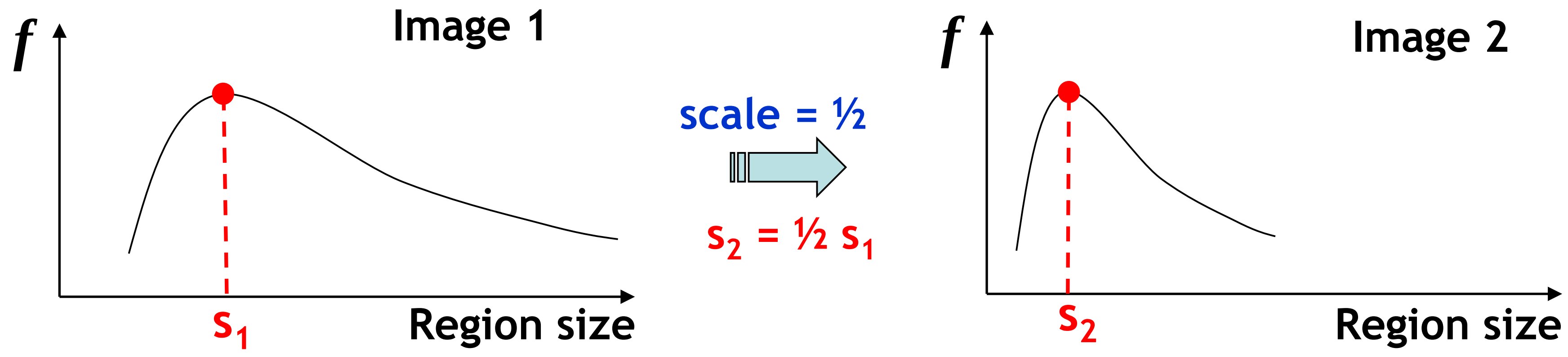
  - For a point in one image, we can consider it as a function of region size (patch width)



Automatic Scale Selection

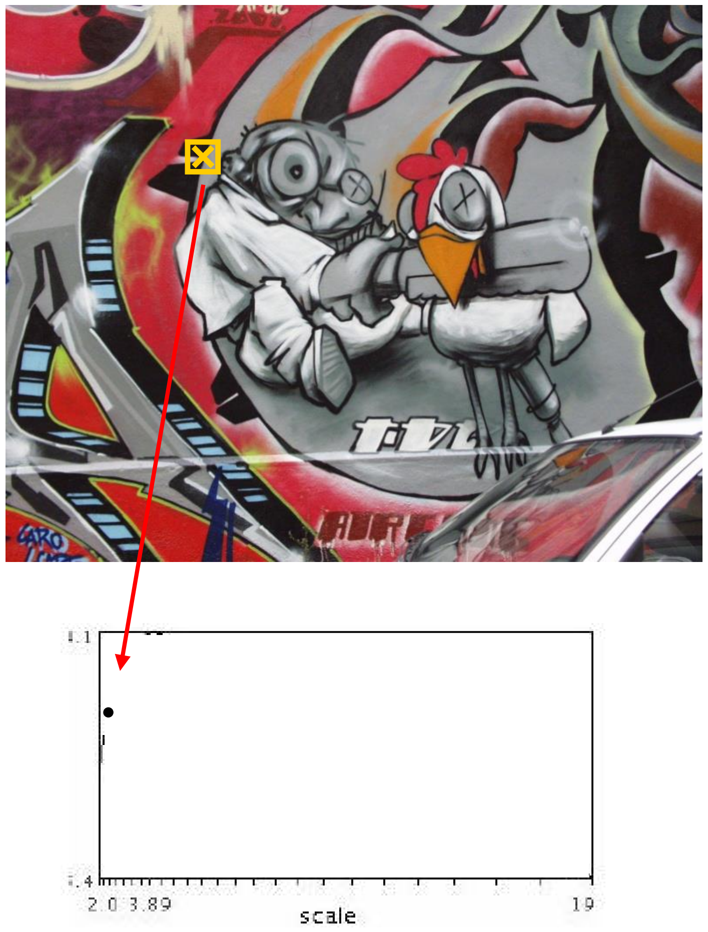
- Common approach:
  - Take a local maximum of this function.
  - Observation: region size for which the maximum is achieved should be *invariant* to image scale.

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

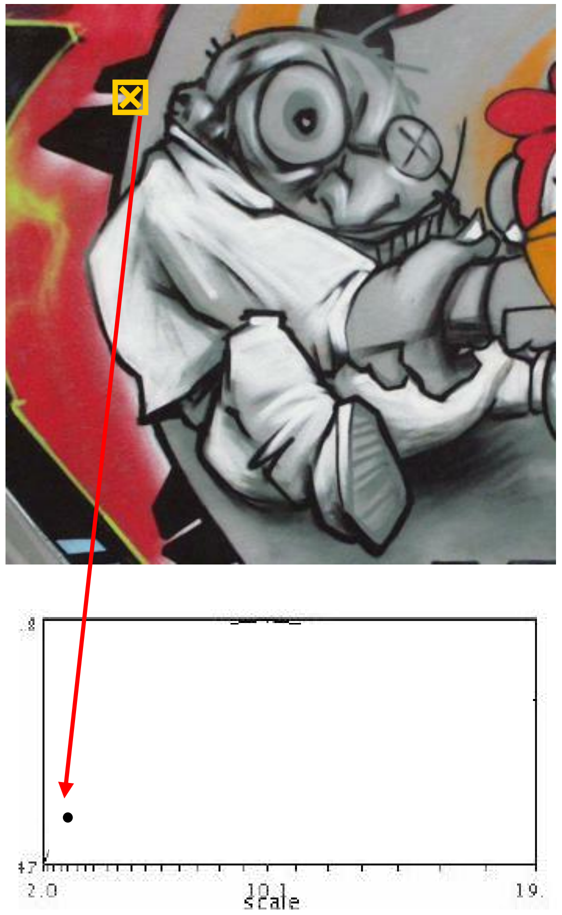


Automatic Scale Selection

- Function responses for increasing scale (scale signature)



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

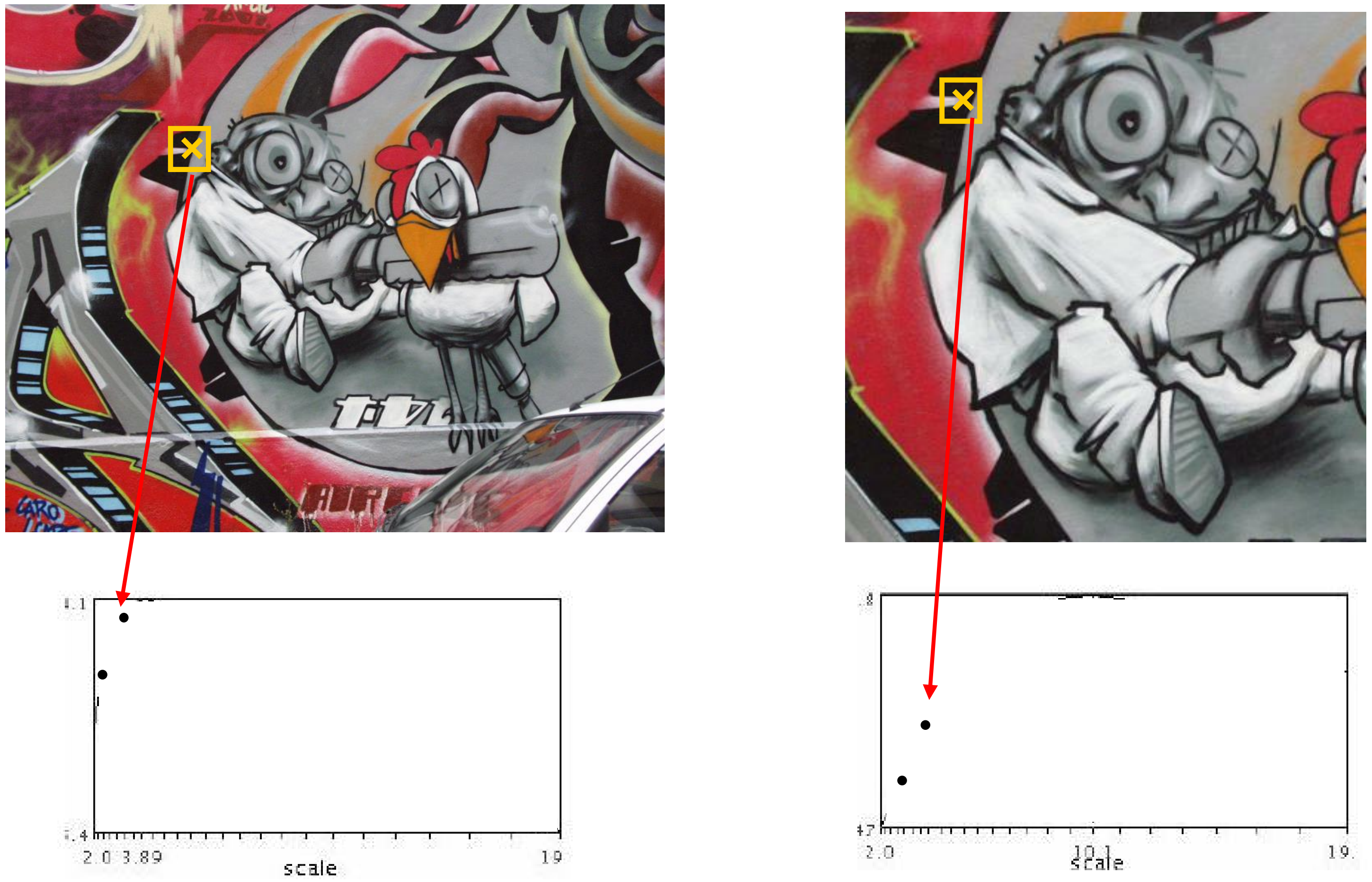


$f(I_{i_1 \ldots i_m}(x',\sigma))$

Slide credit: Krystian Mikolajczyk

Automatic Scale Selection

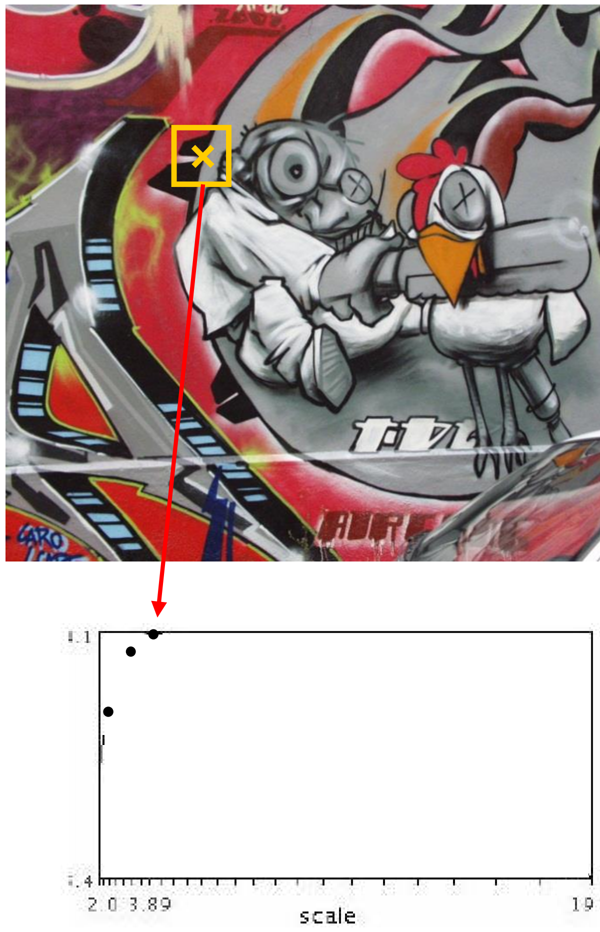
- Function responses for increasing scale (scale signature)



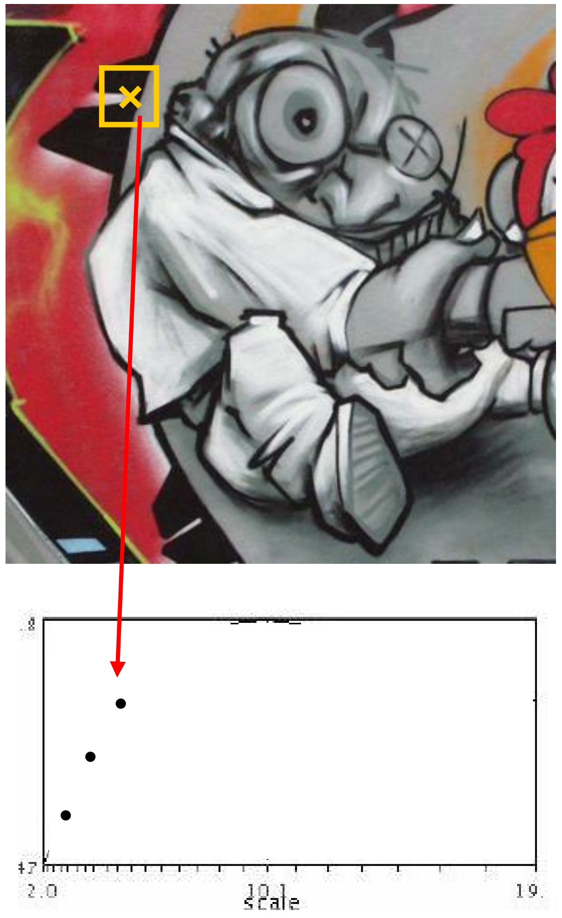
$f(I_{i_1 \ldots i_m}(x, \sigma))$ $f(I_{i_1 \ldots i_m}(x', \sigma))$

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



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

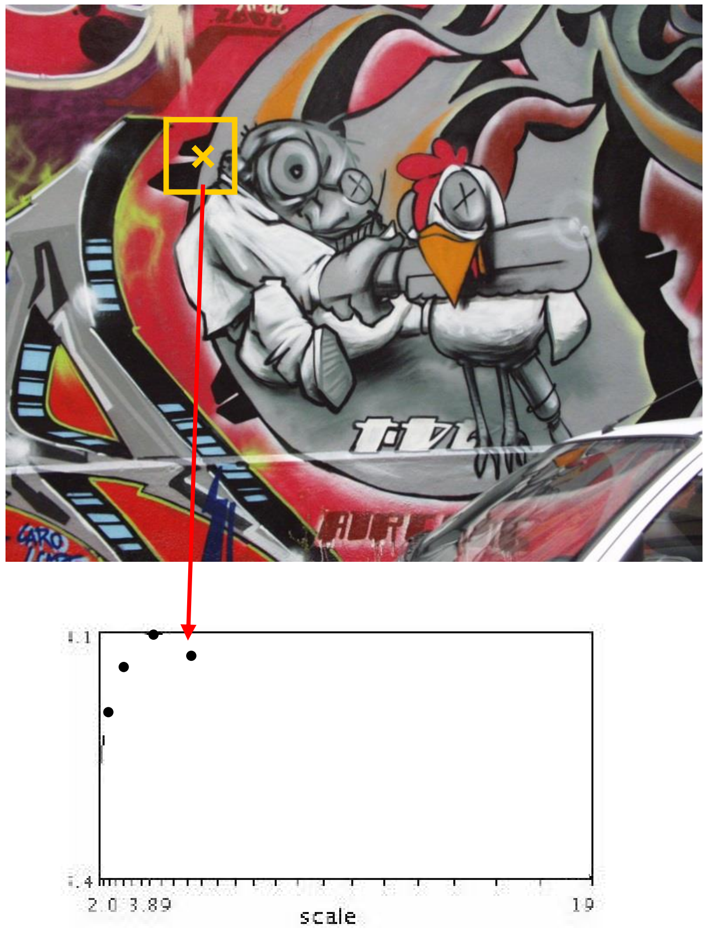


$f(I_{i_1 \dots i_m}(x', \sigma))$

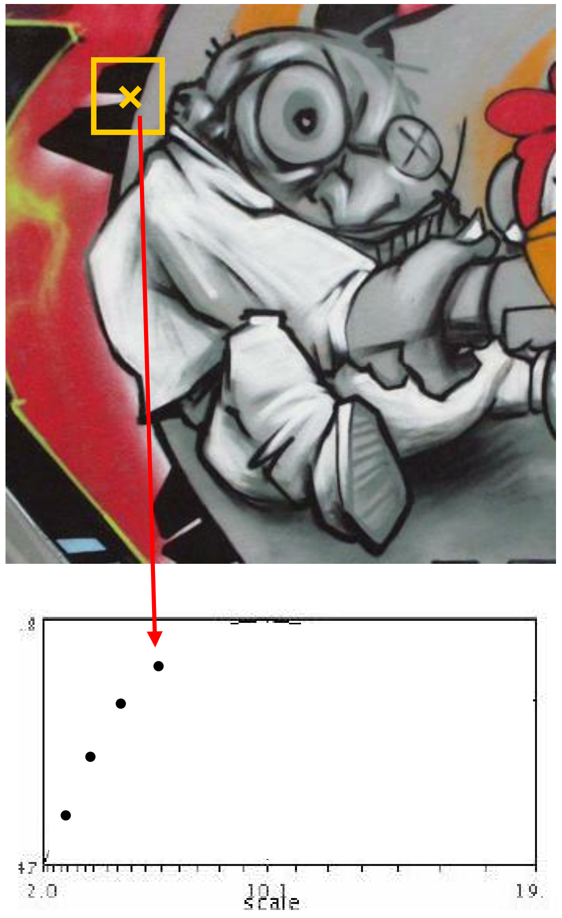
Slide credit: Krystian Mikolajczyk

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



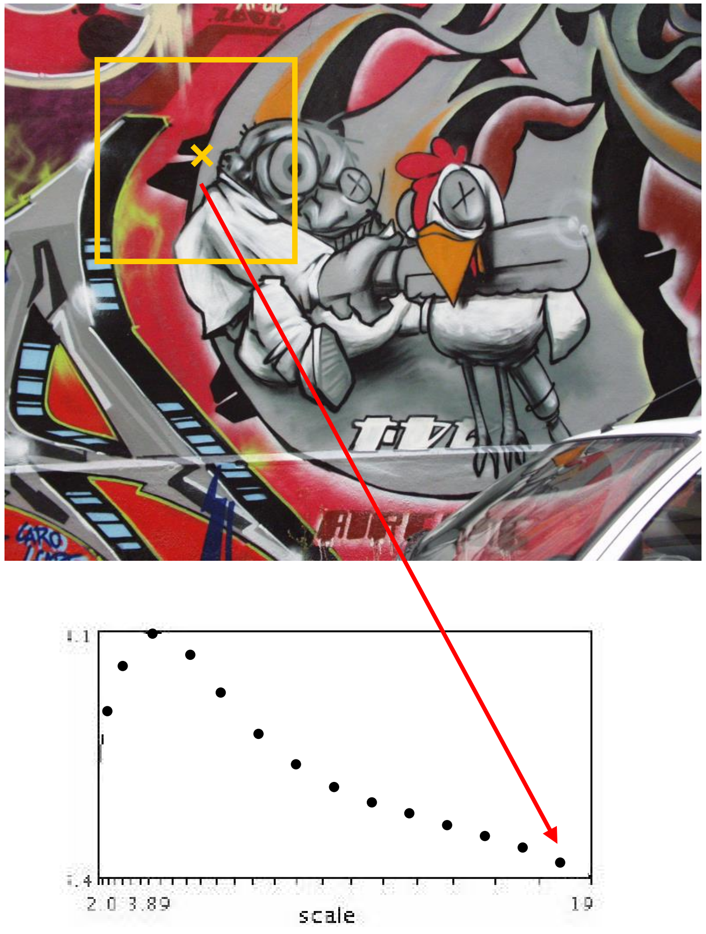
$f(I_{i_1 \dots i_m}(x, \sigma))$



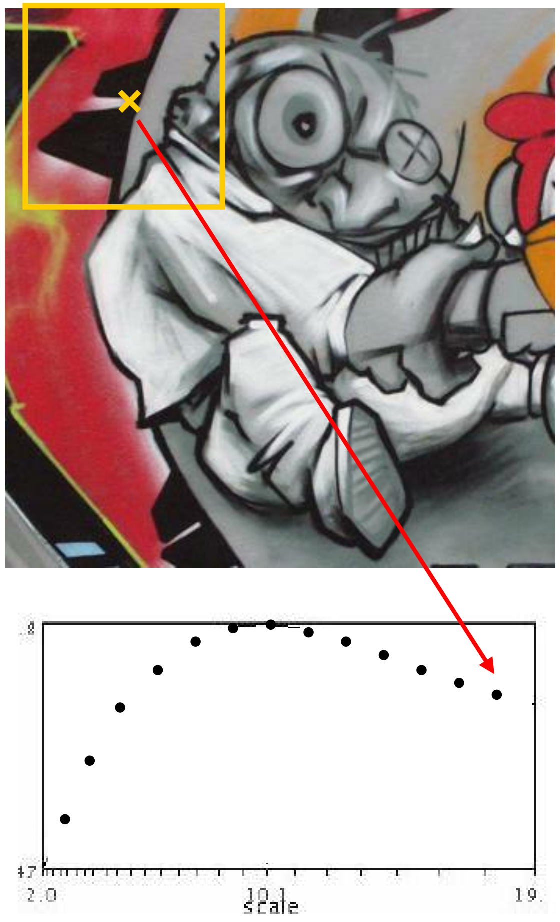
$f(I_{i_1 \dots i_m}(x', \sigma))$

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



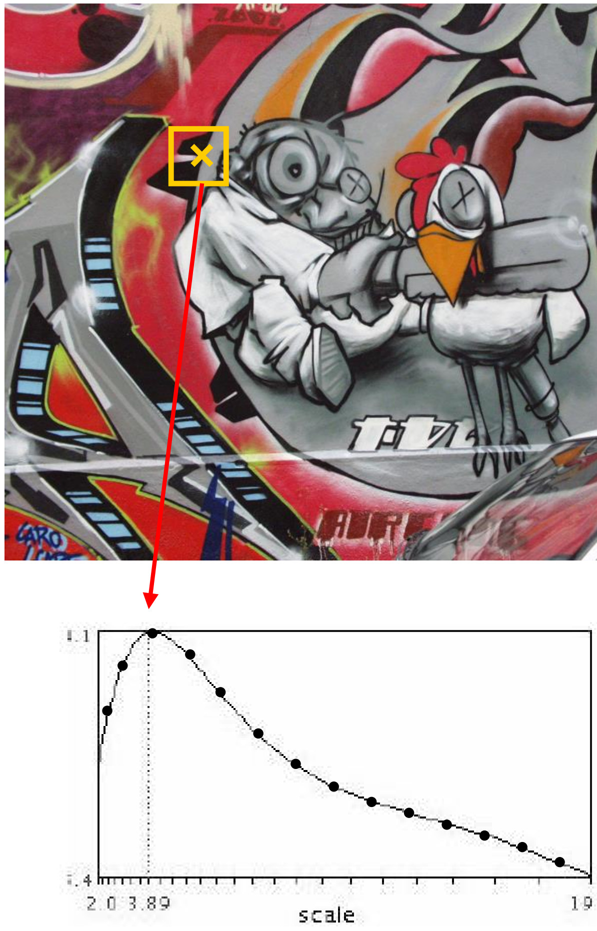
$f(I_{i_1 \ldots i_m}(x, \sigma))$



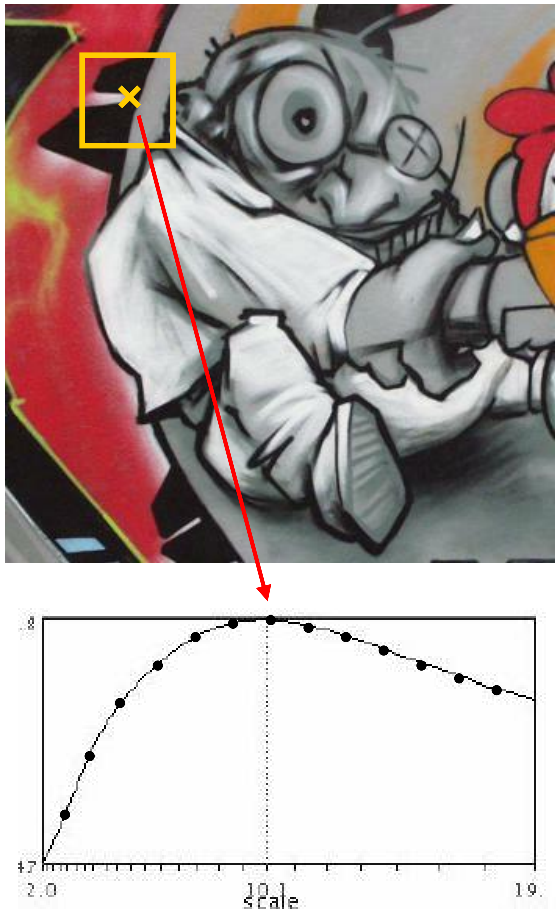
$f(I_{i_1 \ldots i_m}(x', \sigma))$

Automatic Scale Selection

- Function responses for increasing scale (scale signature)



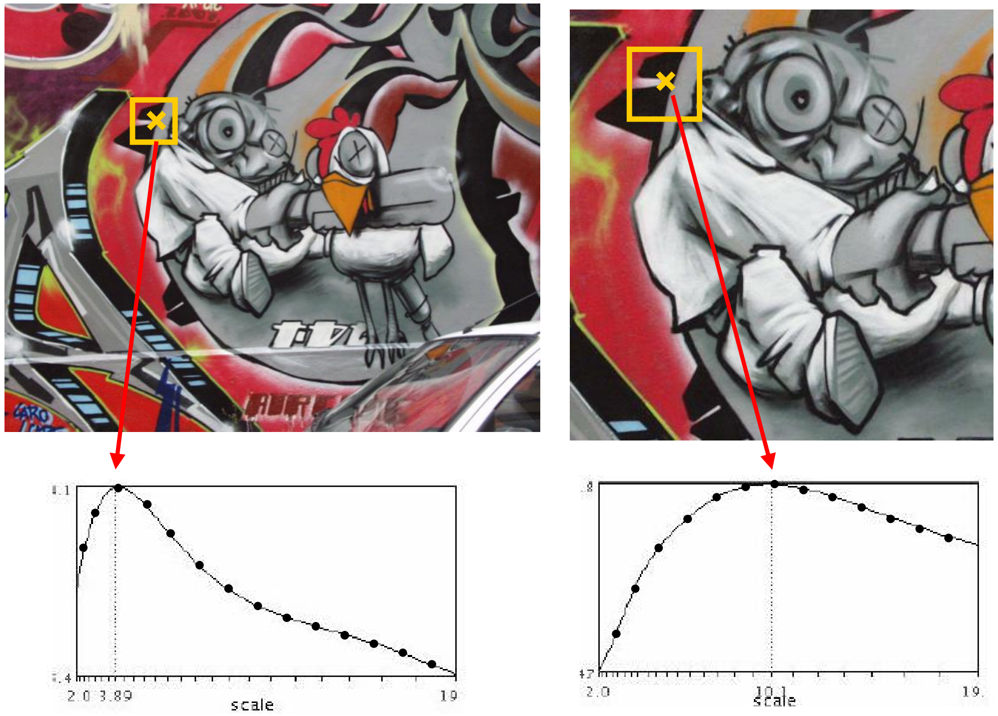
$f(I_{i_1\ldots i_m}(x,\sigma))$



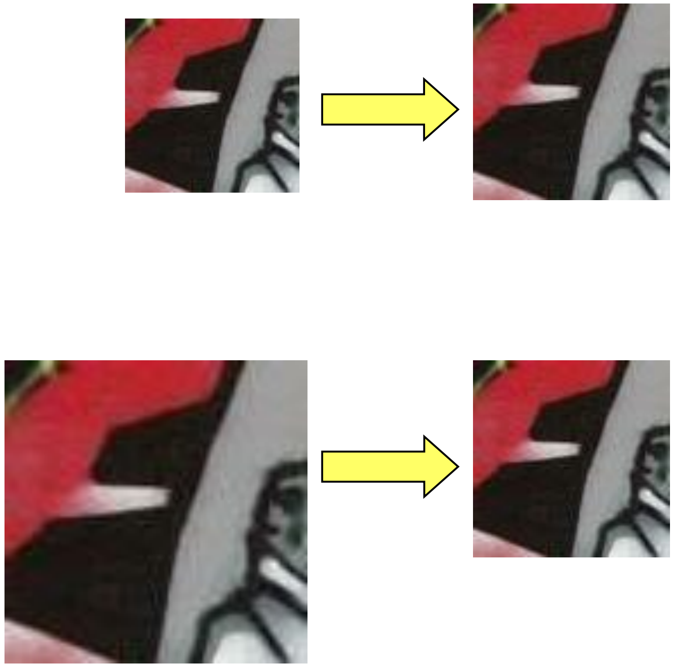
$f(I_{i_1\ldots i_m}(x',\sigma'))$

Automatic Scale Selection

- Normalize: Rescale to fixed size



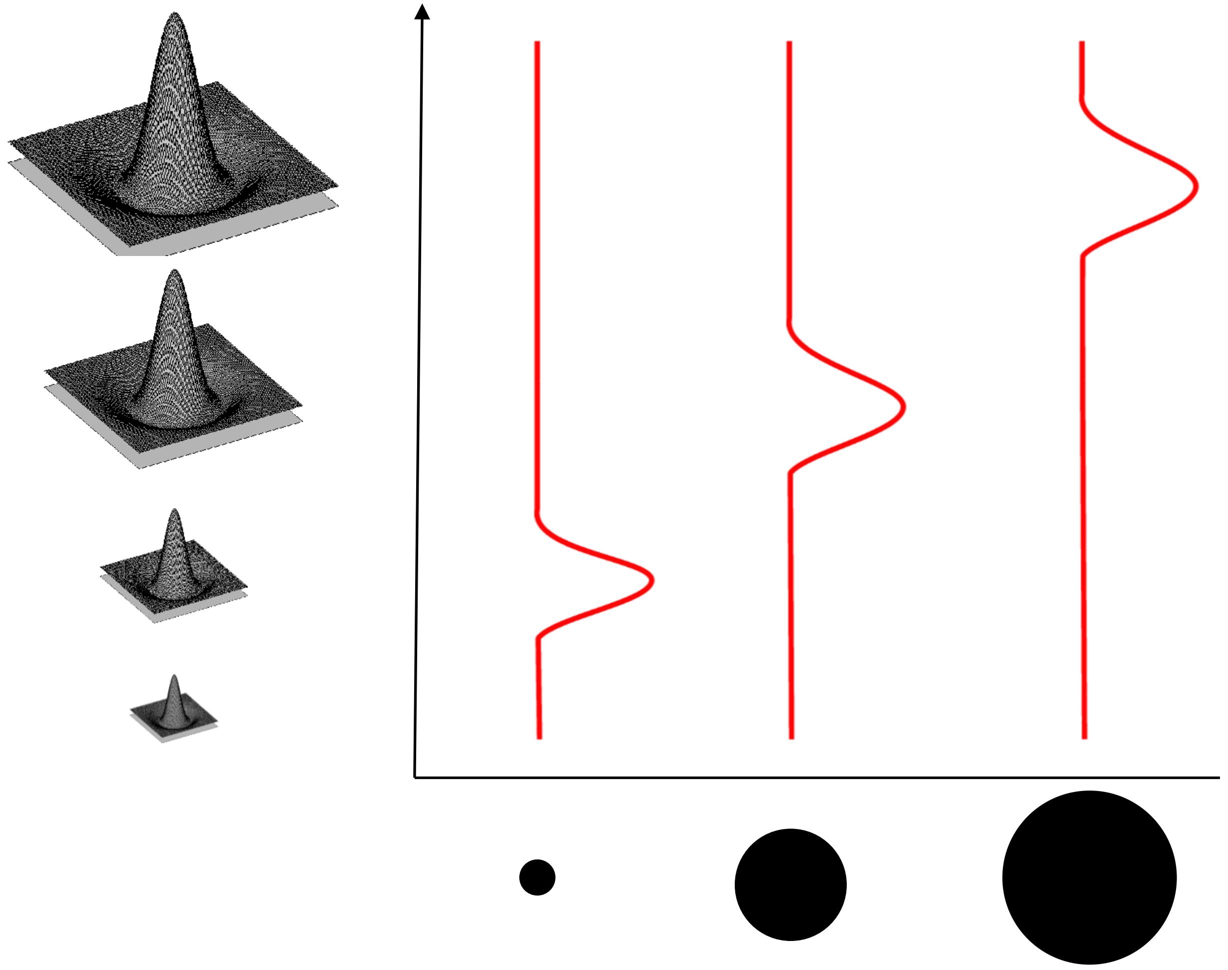
$f(I_{i_1 \ldots i_m}(x,\sigma))$ $f(I_{i_1 \ldots i_m}(x',\sigma'))$



Slide credit: Tinne Tuytelaars

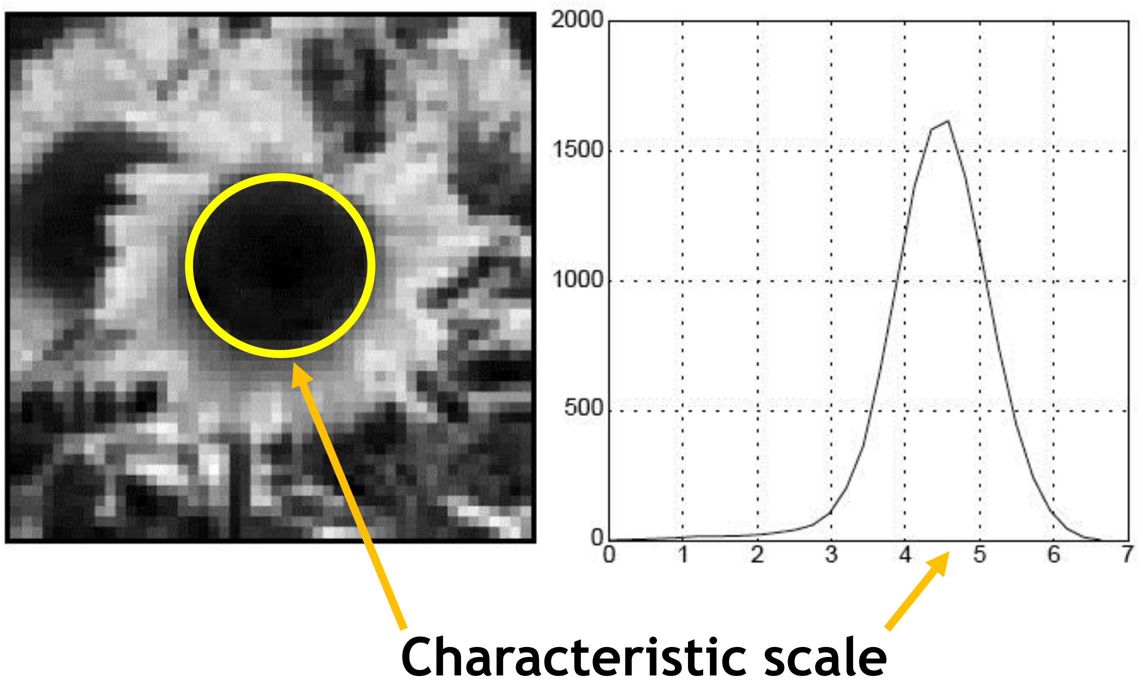
What Is A Useful Signature Function?

- Laplacian-of-Gaussian = “blob” detector



Characteristic Scale

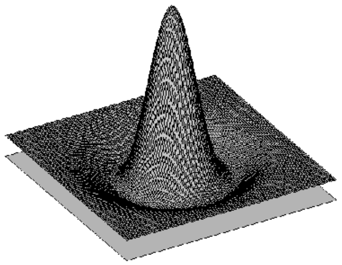
- We define the *characteristic scale* as the scale that produces peak of Laplacian response

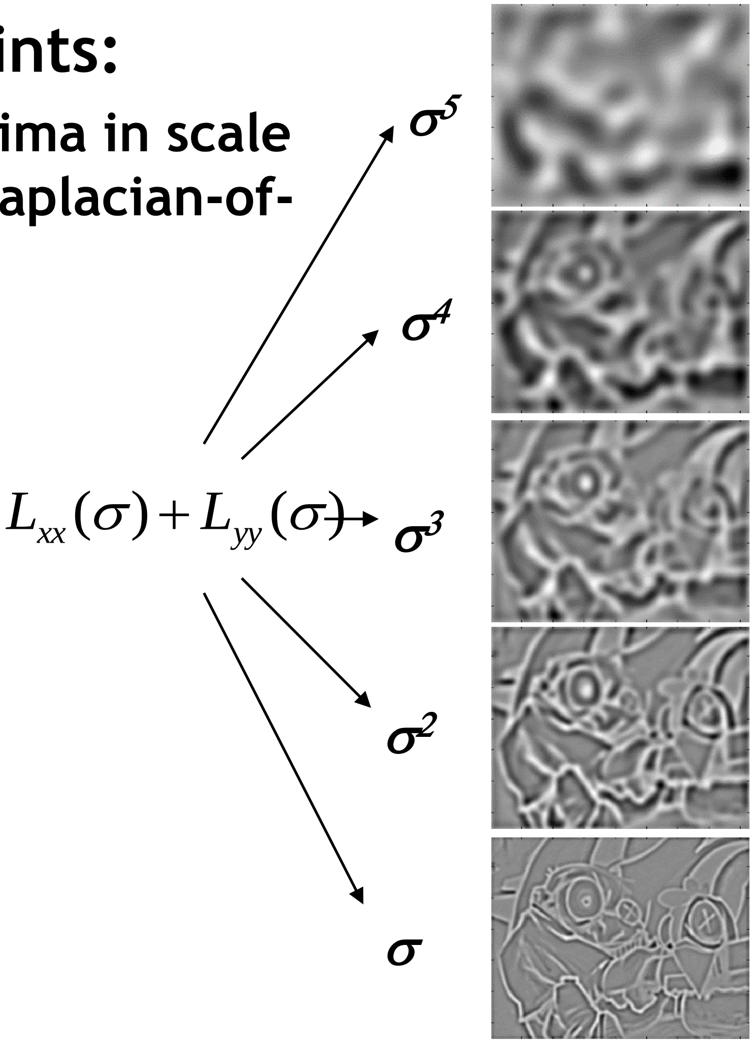


T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* 30 (2): pp 77--116.

Laplacian-of-Gaussian (LoG)

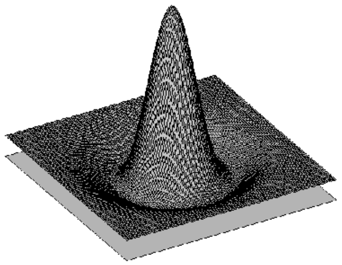
- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian

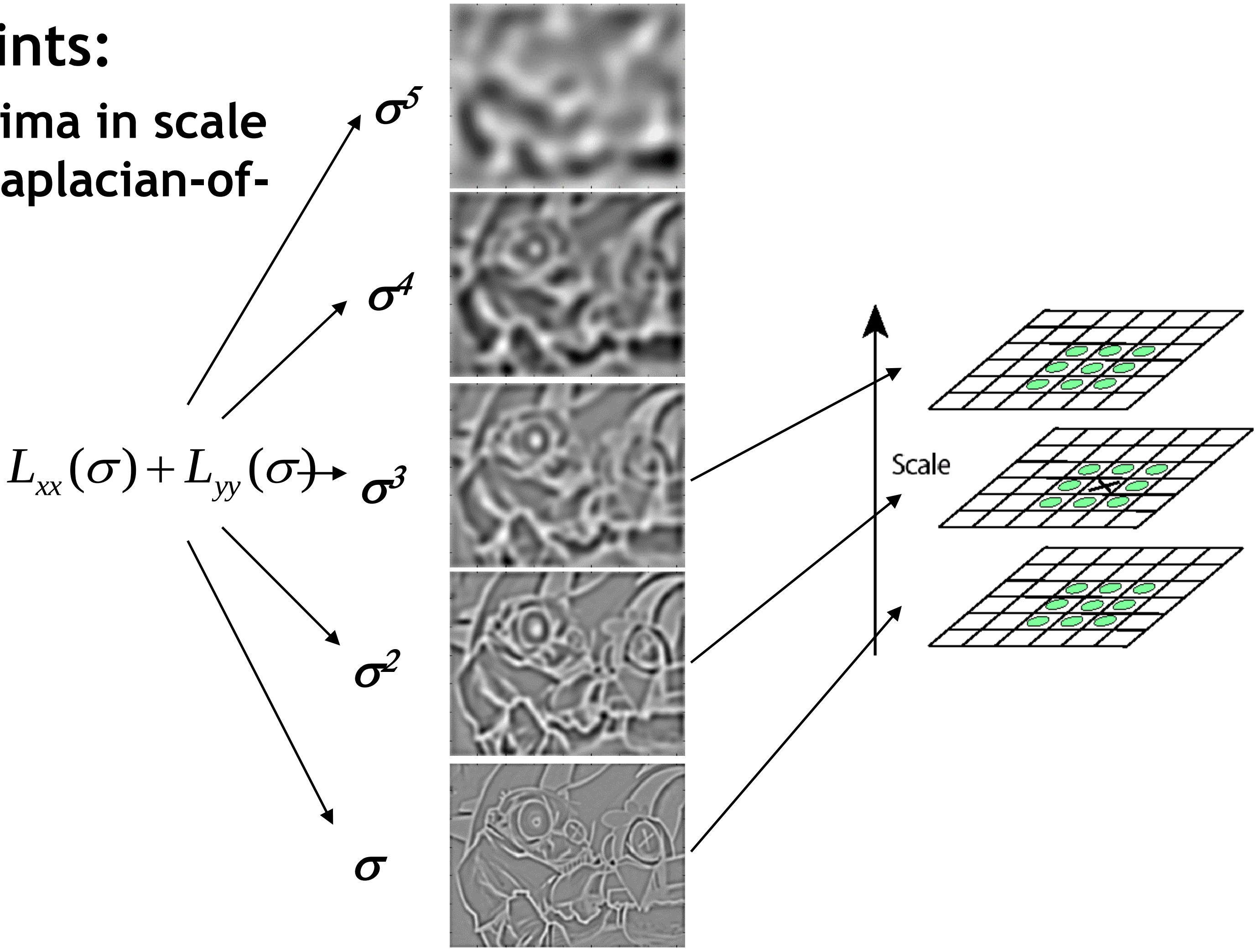




Laplacian-of-Gaussian (LoG)

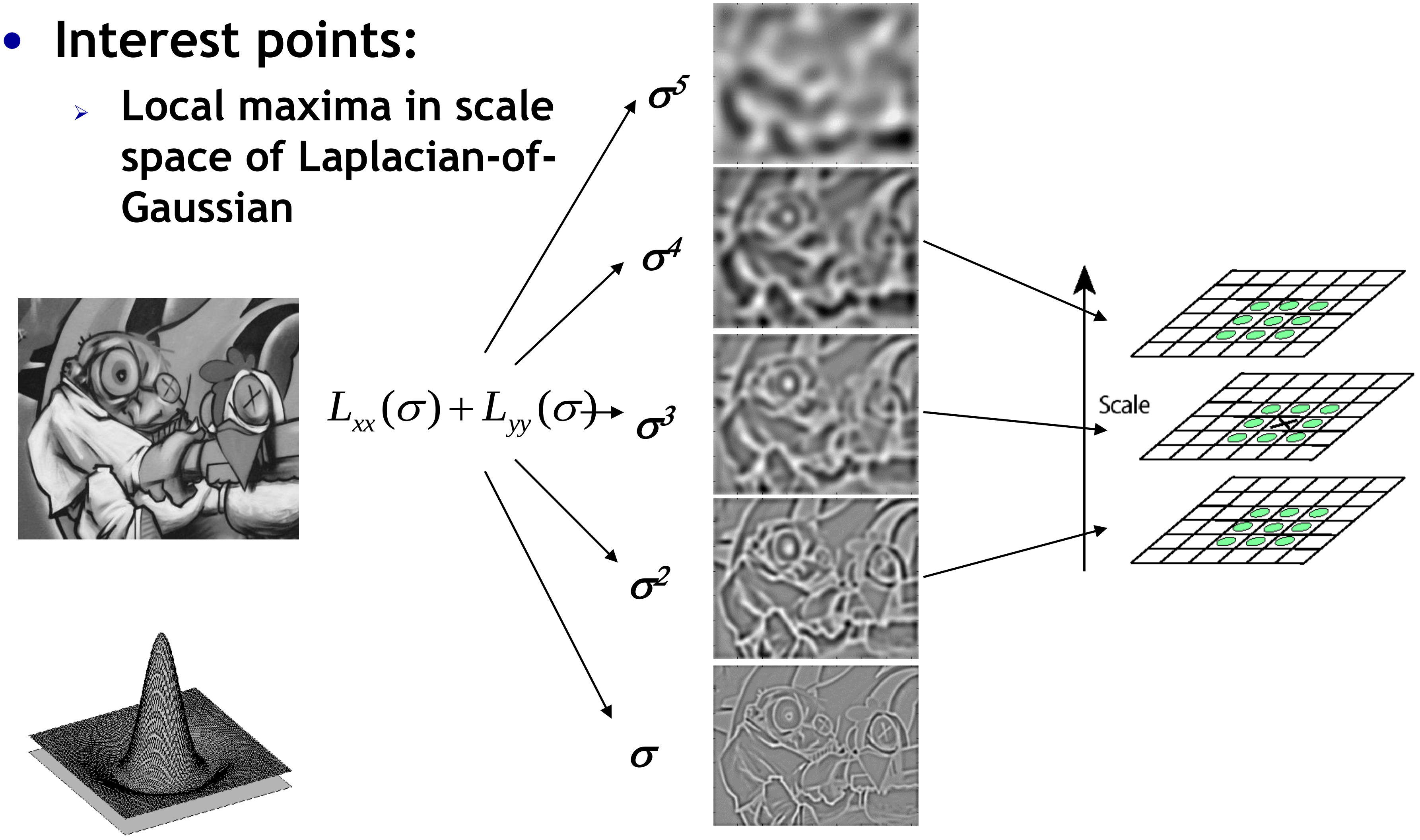
- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian





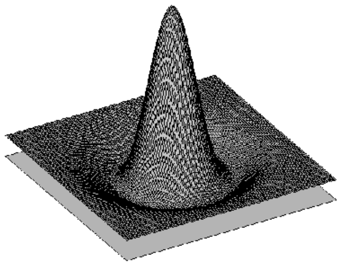
Laplacian-of-Gaussian (LoG)

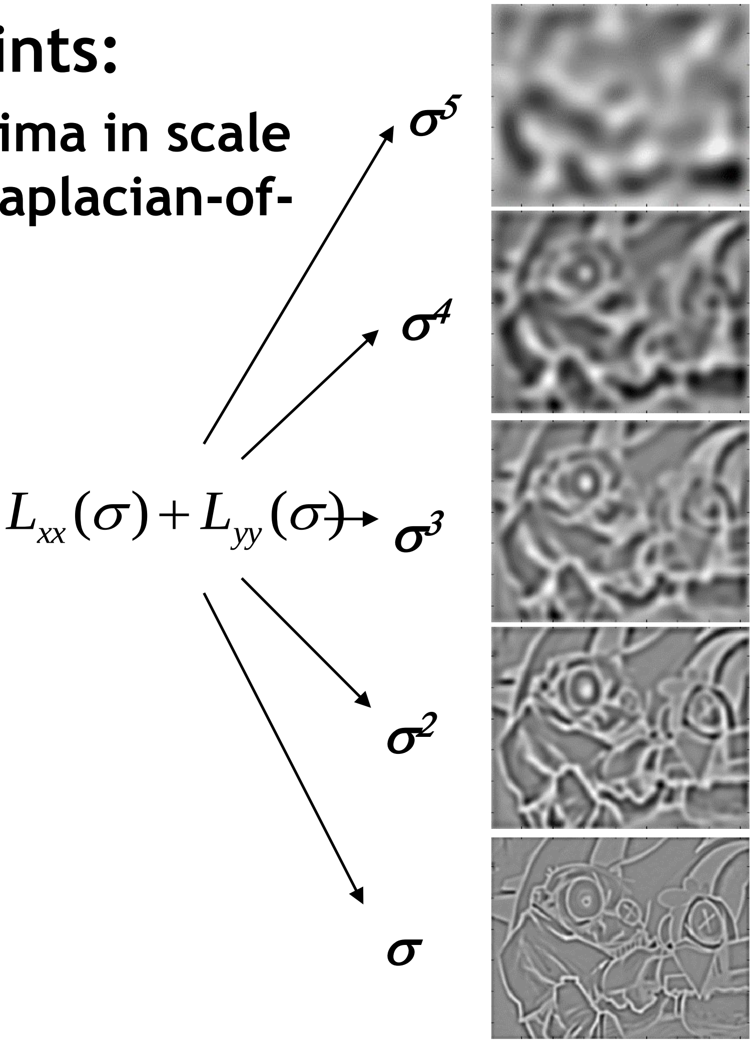
- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian

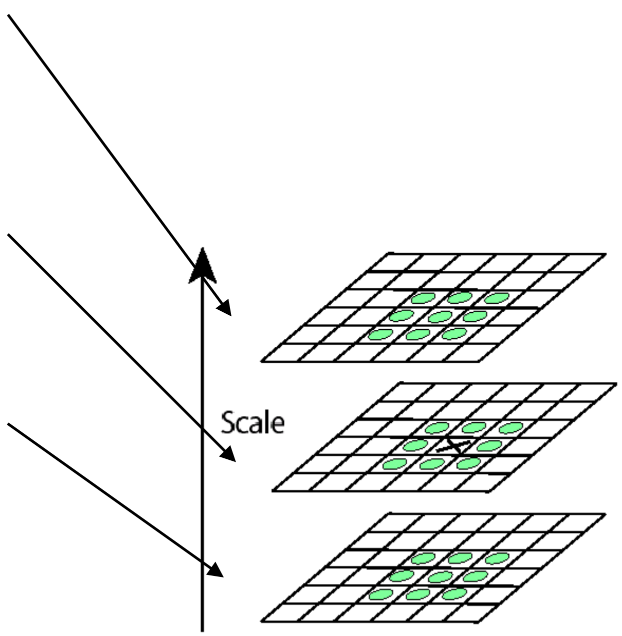


Laplacian-of-Gaussian (LoG)

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







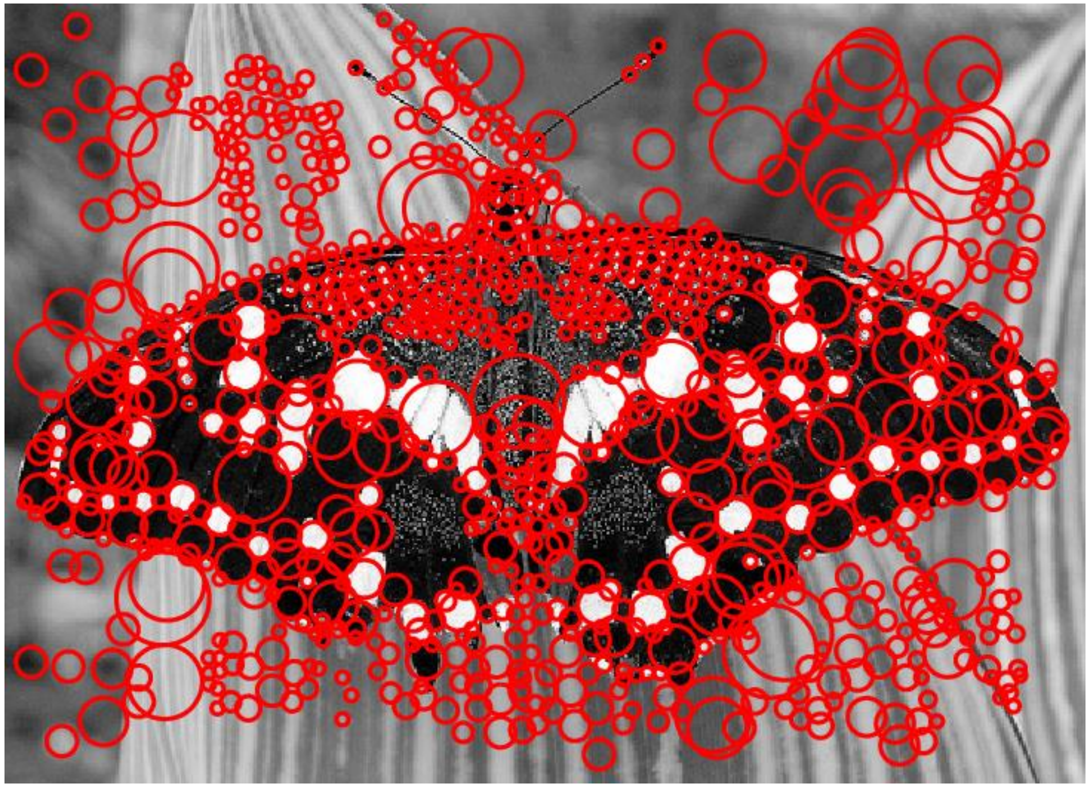
$\Rightarrow$ **List of (x, y, σ)**

LoG Detector: Workflow

LoG Detector: Workflow

sigma = 11.9912

LoG Detector: Workflow



Slide credit: Svetlana Lazebnik

Technical Detail

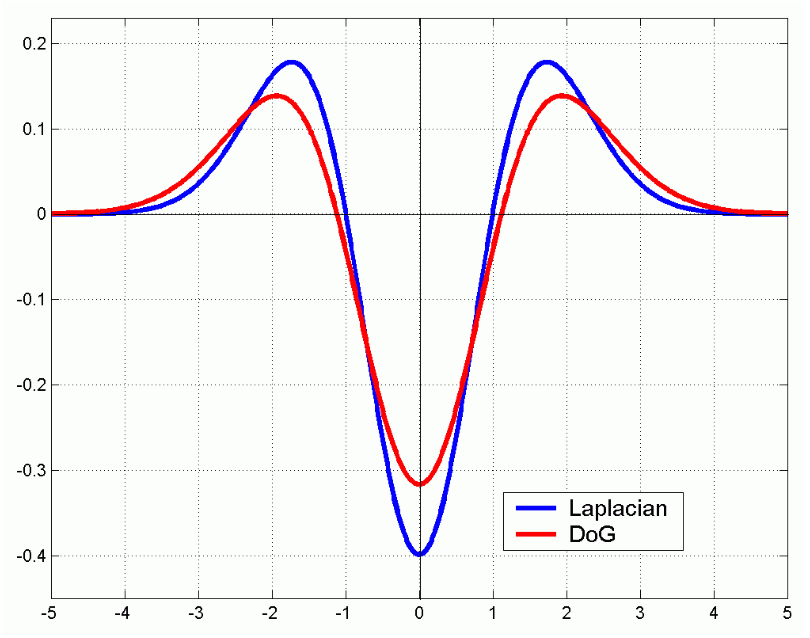
- We can efficiently approximate the Laplacian with a difference of Gaussians:

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

(Laplacian)

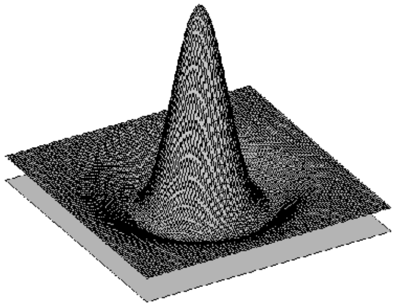
$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

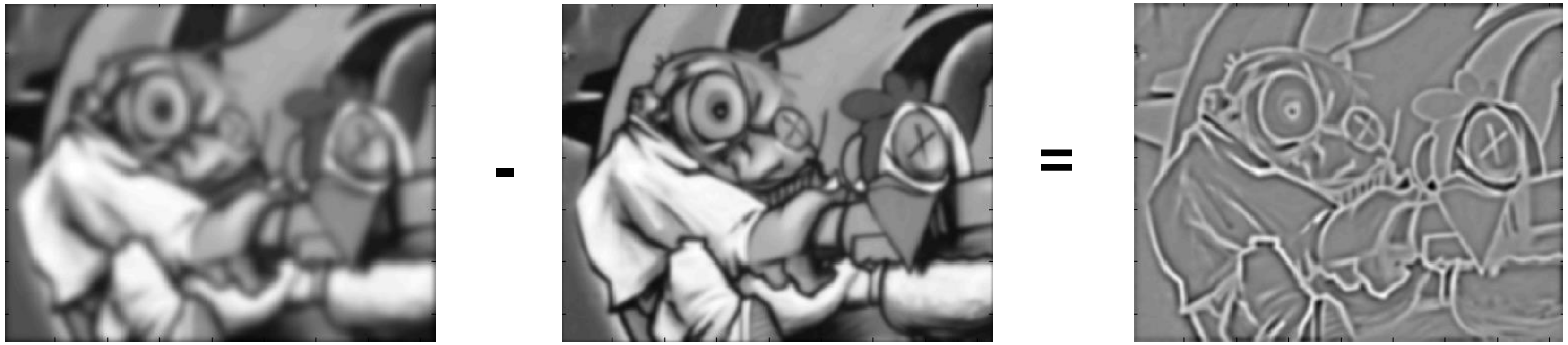
(Difference of Gaussians)



Difference-of-Gaussian (DoG)

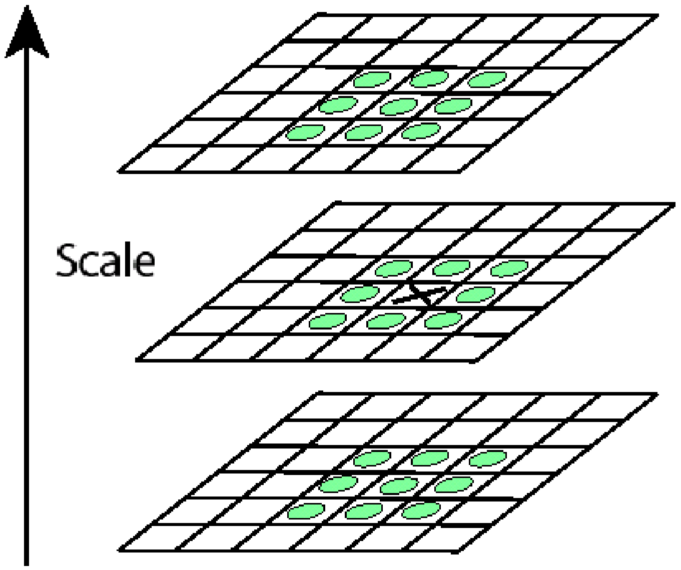
- Difference of Gaussians as approximation of the LoG
  - This is used e.g. in Lowe's SIFT pipeline for feature detection.
- Advantages
  - No need to compute 2nd derivatives
  - Gaussians are computed anyway, e.g. in a Gaussian pyramid.





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

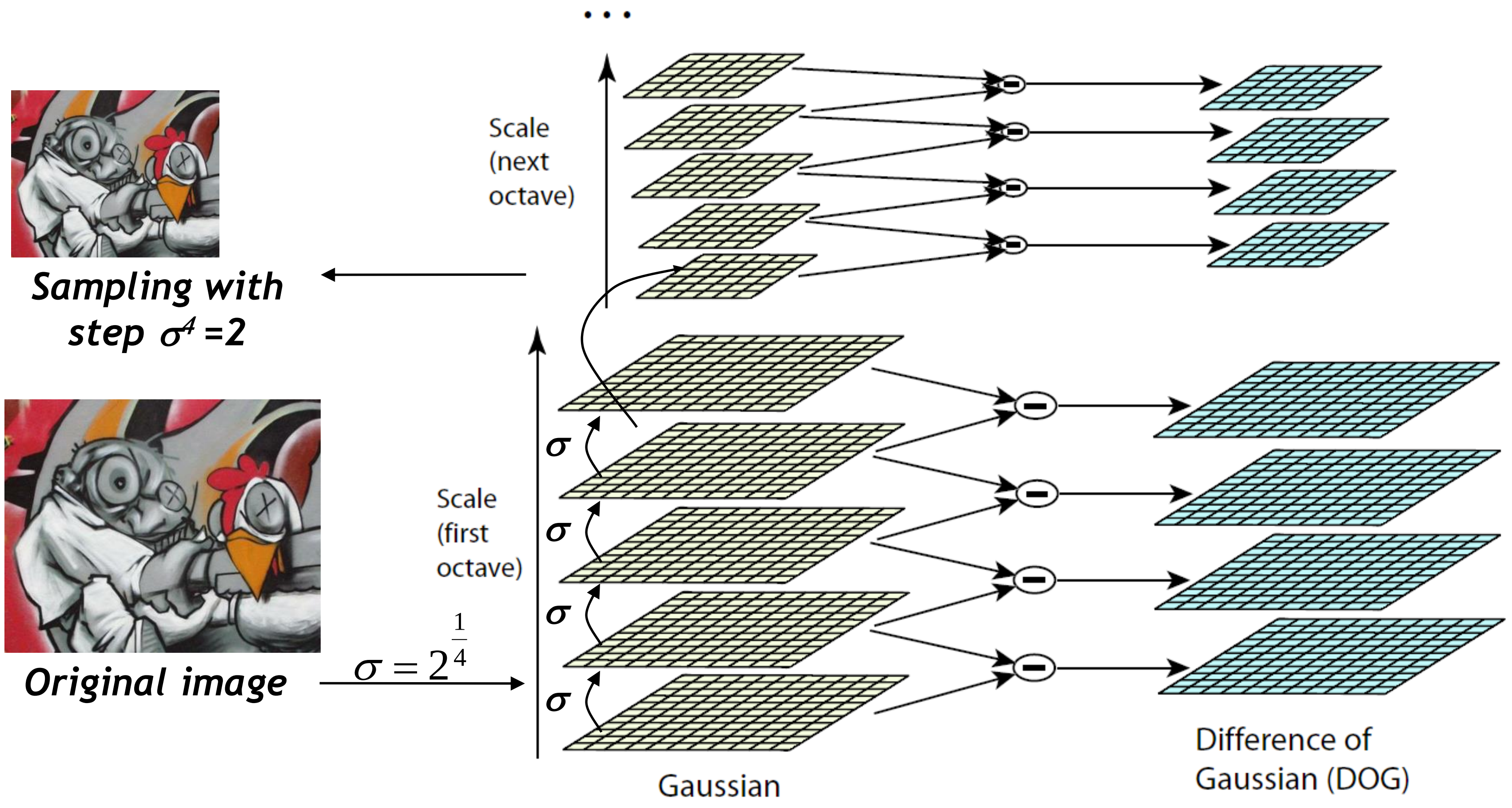


Candidate keypoints: list of (x,y,σ)

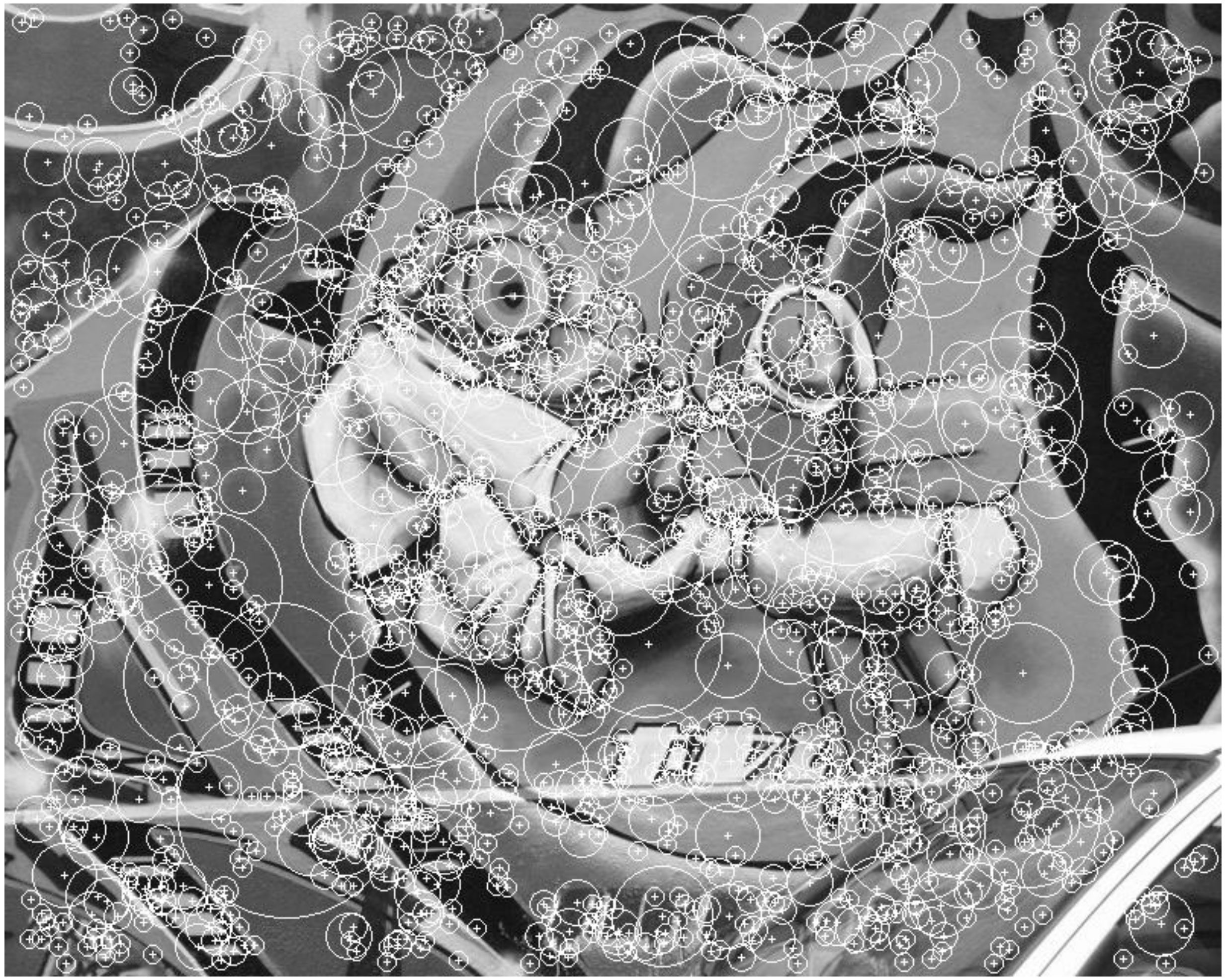
Slide credit: David Lowe

DoG - Efficient Computation

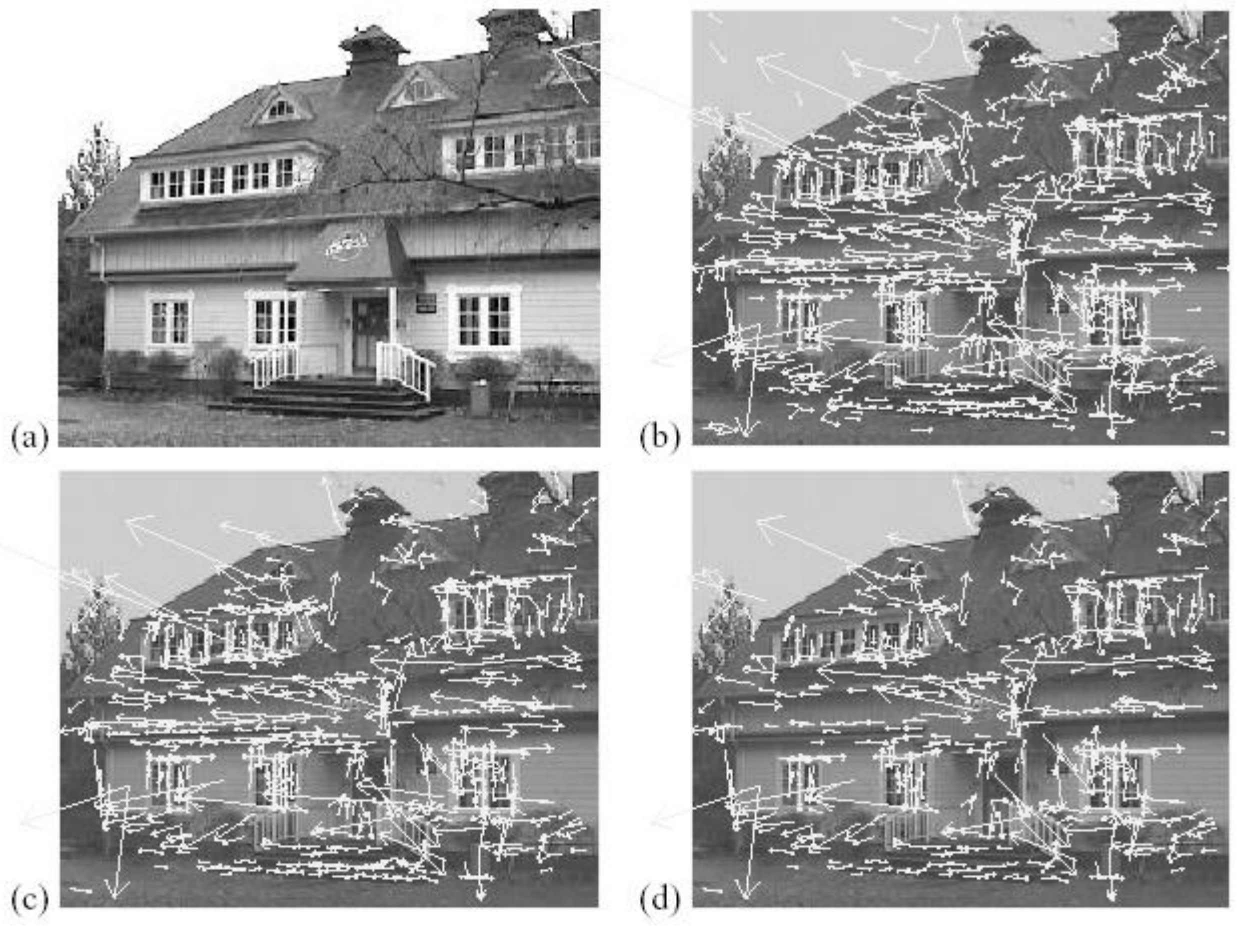
- Computation in Gaussian scale pyramid



Results: Lowe's DoG



Example of Keypoint Detection



(a) 233x189 image

(b) 832 DoG extrema

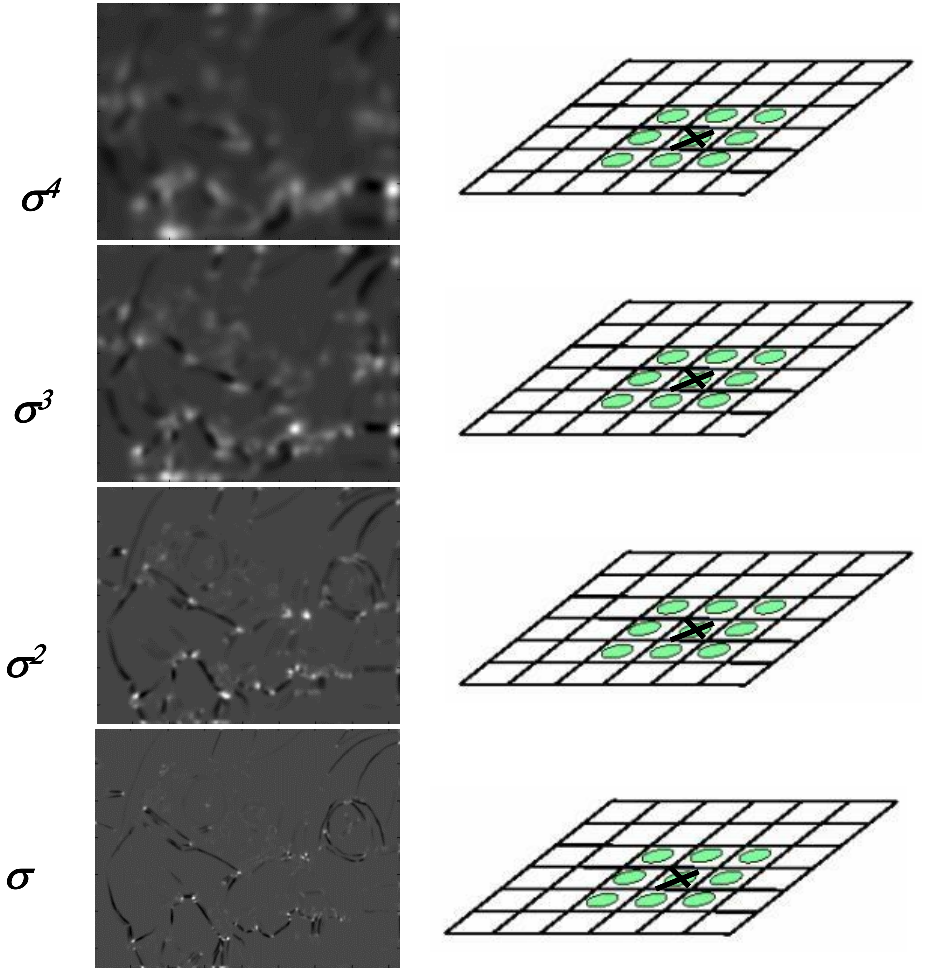
(c) 729 left after peak value threshold

(d) 536 left after testing ratio of principle curvatures (removing edge responses)

Slide credit: David Lowe

Harris-Laplace [Mikolajczyk '01]

1. Initialization: Multiscale Harris corner detection



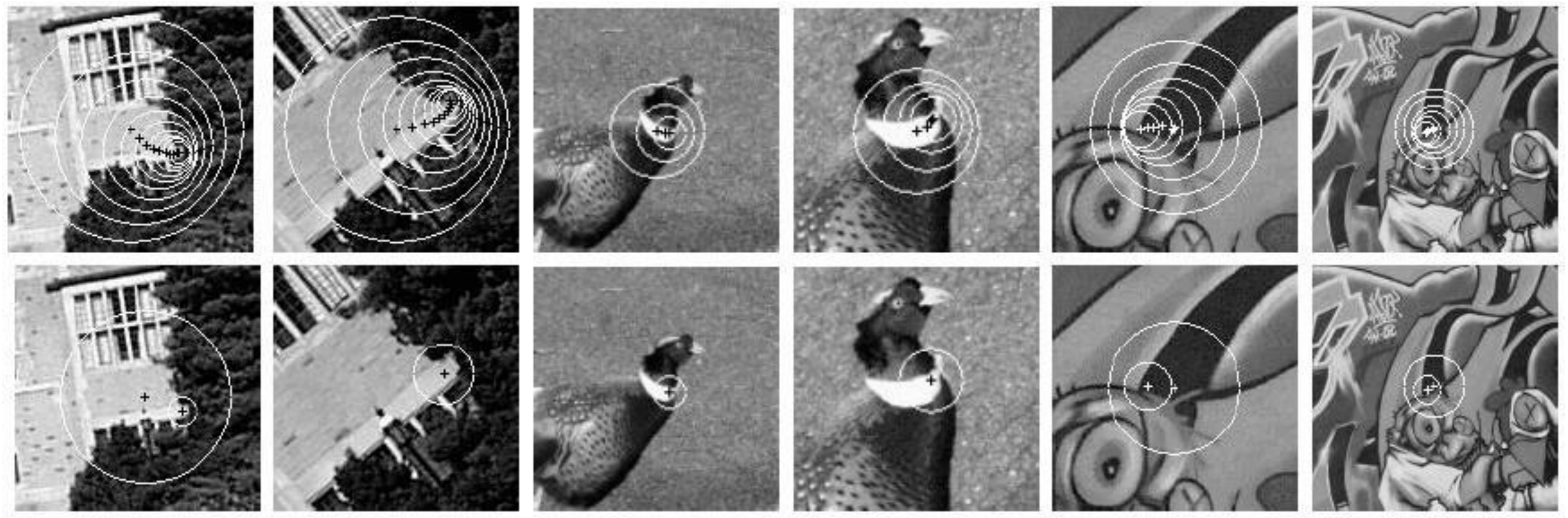
Computing Harris function Detecting local maxima

Slide adapted from Krystian Mikolajczyk

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

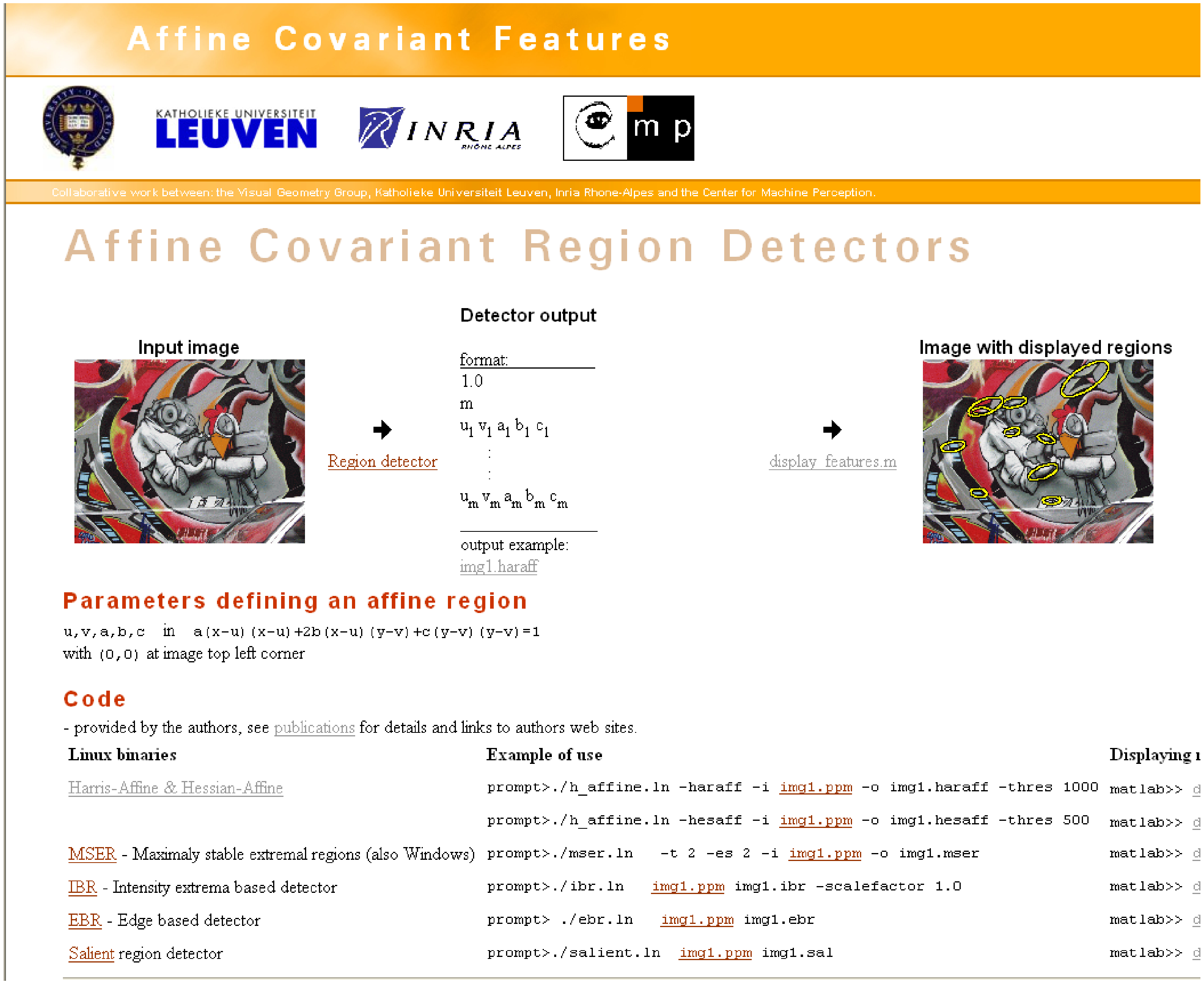
- Given: Two images of the same scene with a large *scale difference* between them.
- Goal: Find *the same* interest points *independently* in each image.
- Solution: Search for *maxima* of suitable functions in *scale* and in *space* (over the image).

- 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).*

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

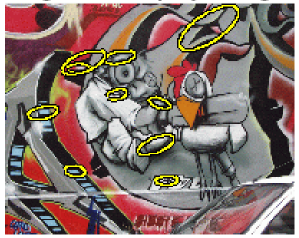
Detector output

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

References and Further Reading

- Read David Lowe’s SIFT paper
  - D. Lowe,
    Distinctive image features from scale-invariant keypoints,
    IJCV 60(2), pp. 91-110, 2004

- Good survey paper on Int. Pt. detectors and descriptors
  - T. Tuytelaars, K. Mikolajczyk, *Local Invariant Feature Detectors: A Survey*, Foundations and Trends in Computer Graphics and Vision, Vol. 3, No. 3, pp 177-280, 2008.

- Try the example code, binaries, and Matlab wrappers
  - Good starting point: Oxford interest point page
    http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries