



# 计算机视觉表征与识别

## Chapter 8: Interest Points: Descriptor

王利民

媒体计算课题组

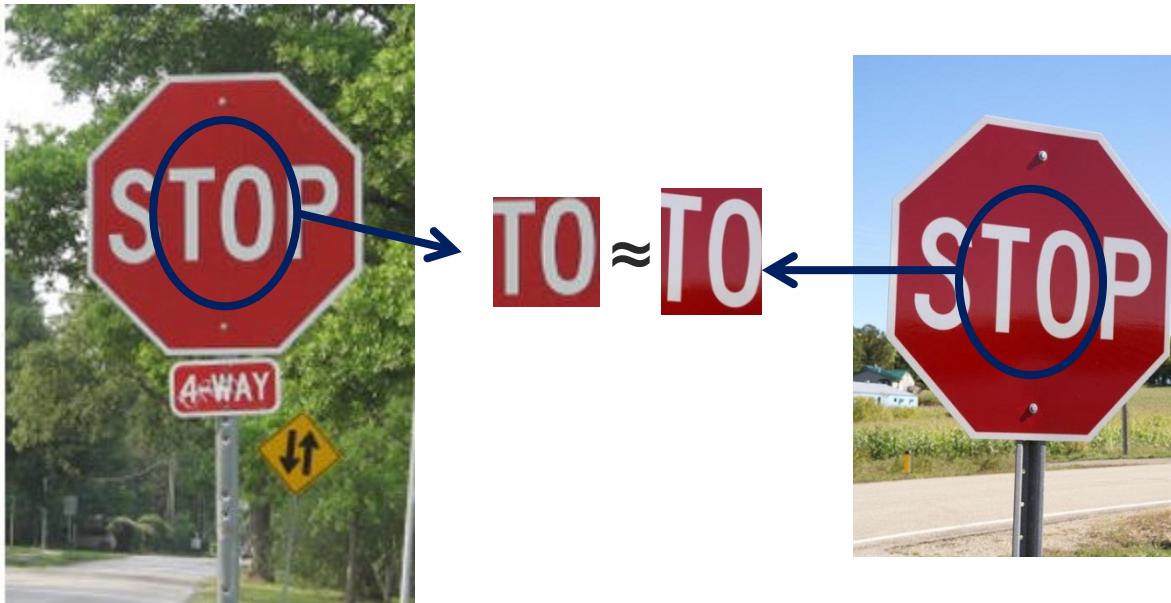
<http://mcg.nju.edu.cn/>



# Correspondence and alignment

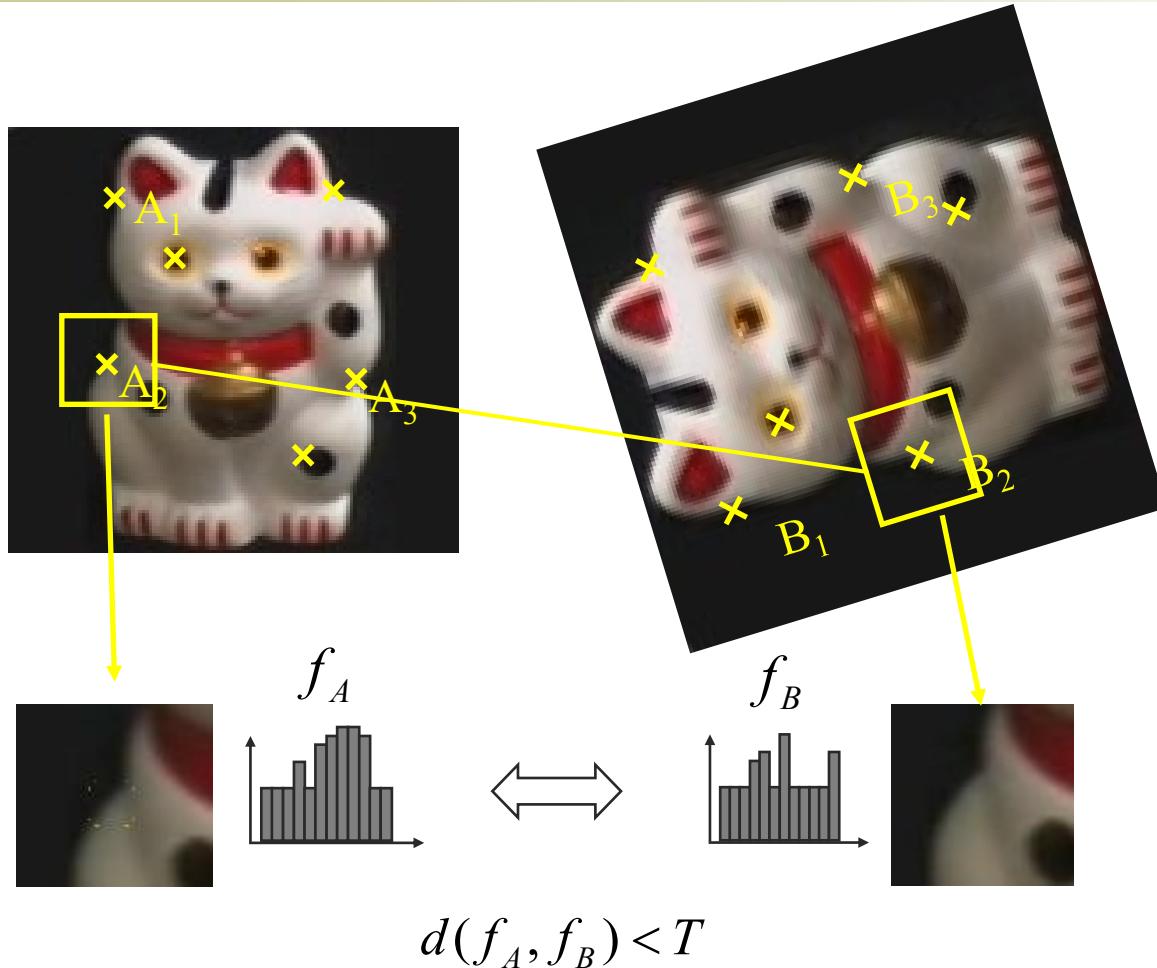


**Correspondence:** matching points, patches, edges, or regions across images





# Recap: Keypoint Matching

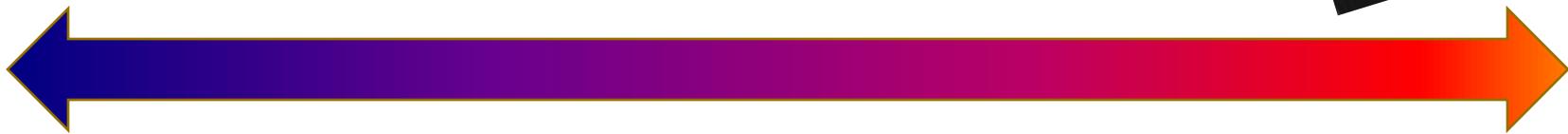


- 1. Find a set of distinctive keypoints**
- 2. Define a region around each keypoint**
- 3. Extract and normalize the region content**
- 4. Compute a local descriptor from the normalized region**
- 5. Match local descriptors**



# Recap: Key trade-offs

## Detection



More Repeatable

Robust detection  
Precise localization

More Points

Robust to occlusion  
Works with less texture

## Description



More Distinctive

More wrong matches

More Flexible

Robust to expected variations  
Maximize correct matches

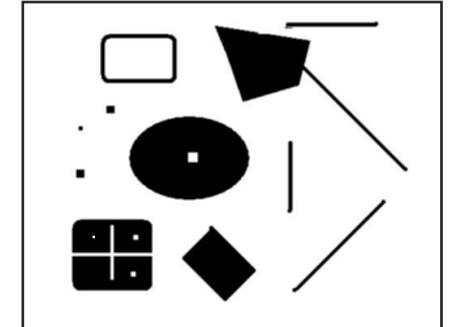


# Harris Detector [Harris88]



## ■ Second moment matrix

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



*Intuition:* Search for local neighborhoods where the image content has two main directions (eigenvectors).

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



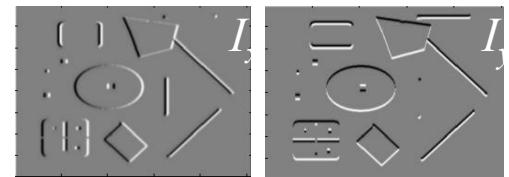
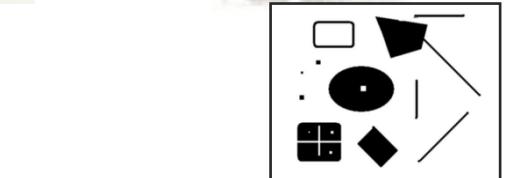
# Harris Detector [Harris88]



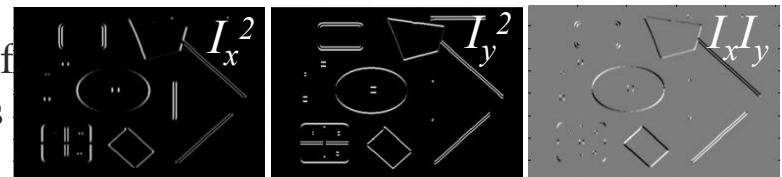
## ■ Second moment matrix

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

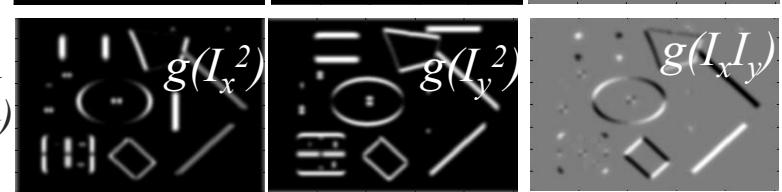
1. Image derivatives  
(optionally, blur first)



2. Square of derivatives



3. Gaussian filter  $g(\sigma_D)$



$$\det M = \lambda_1 \lambda_2$$

$$\text{trace } M = \lambda_1 + \lambda_2$$

4. Cornerness function – both eigenvalues are strong

$$har = \det[\mu(\sigma_I, \sigma_D)] - \alpha [\text{trace}(\mu(\sigma_I, \sigma_D))^2] =$$

$$g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2$$

5. Non-maxima suppression

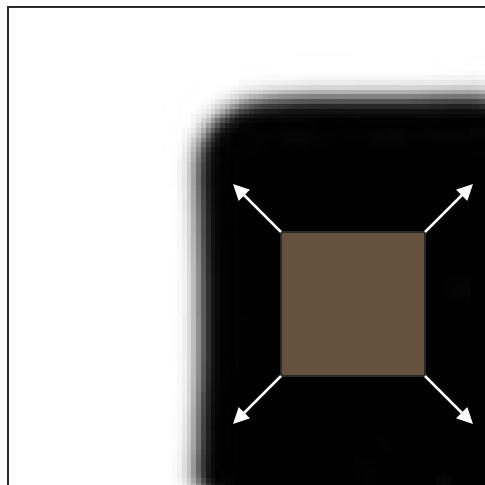




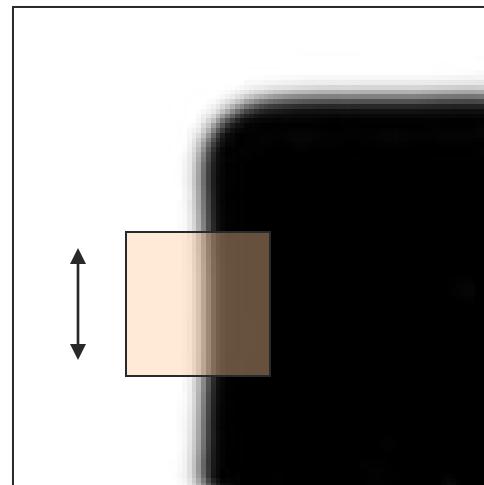
# Corners as distinctive interest points



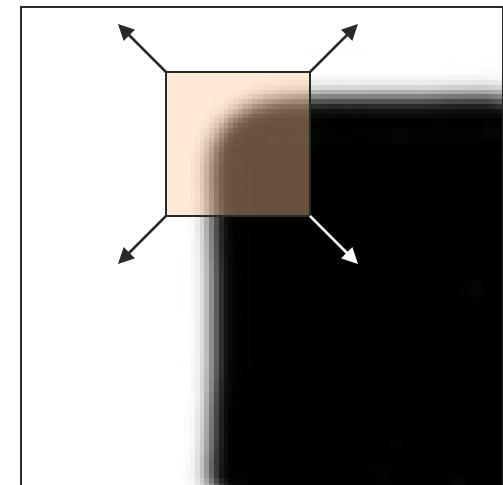
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:  
no change in  
all directions



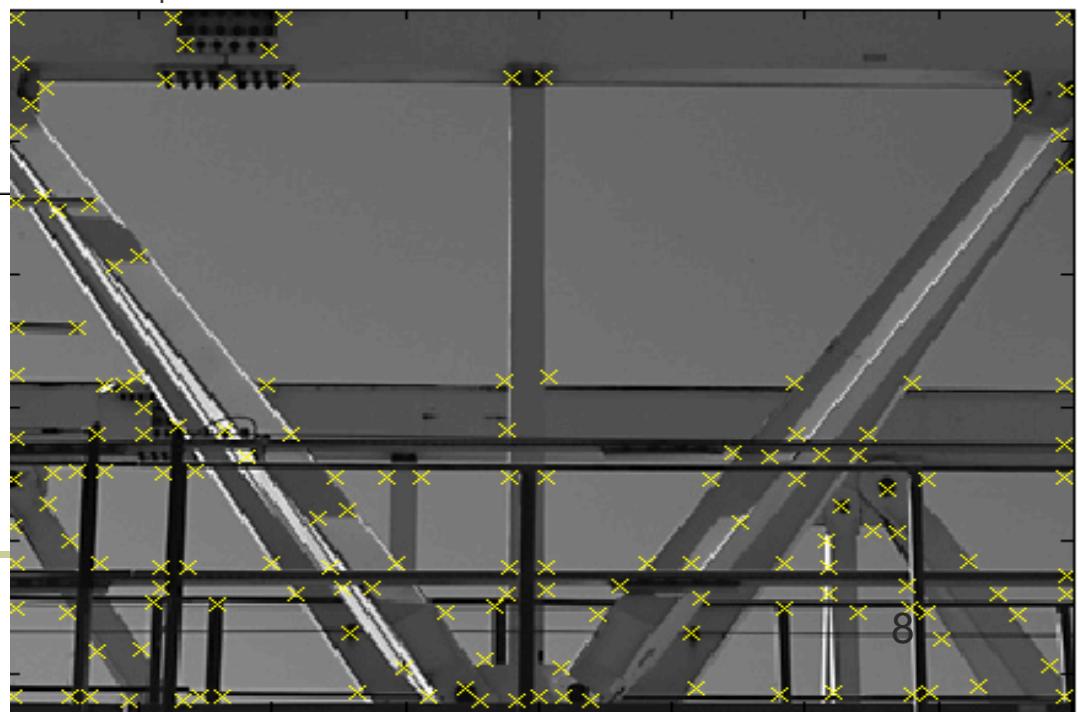
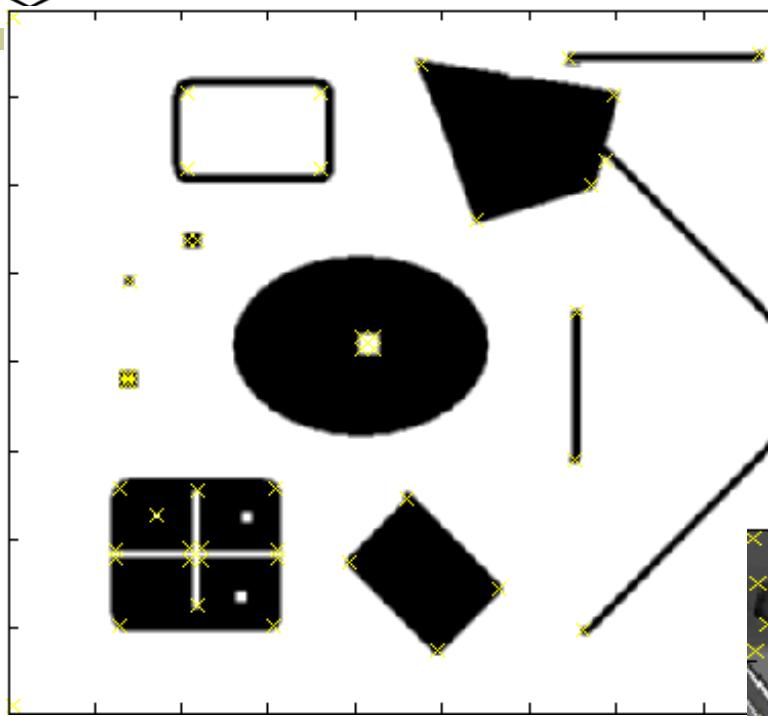
“edge”:  
no change  
along the edge  
direction



“corner”:  
significant  
change in all  
directions <sup>7</sup>



# Harris Detector – Responses [Harris88]



*Effect:* A very precise corner detector.

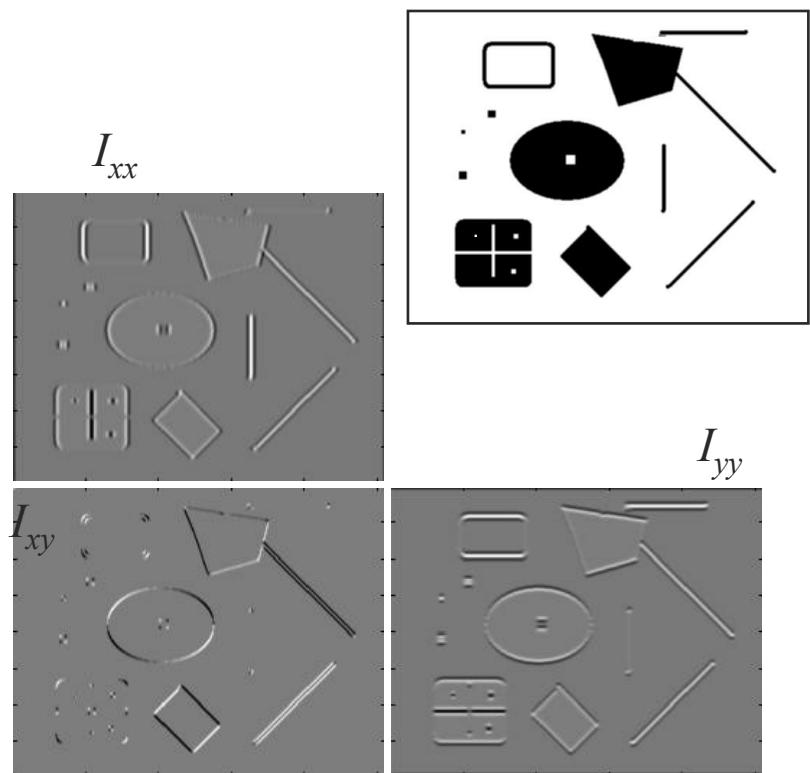


# Hessian Detector [Beaudet78]



## ■ Hessian determinant

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



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



# Hessian Detector [Beaudet78]



## ■ Hessian determinant

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

$$\det M = \lambda_1 \lambda_2$$

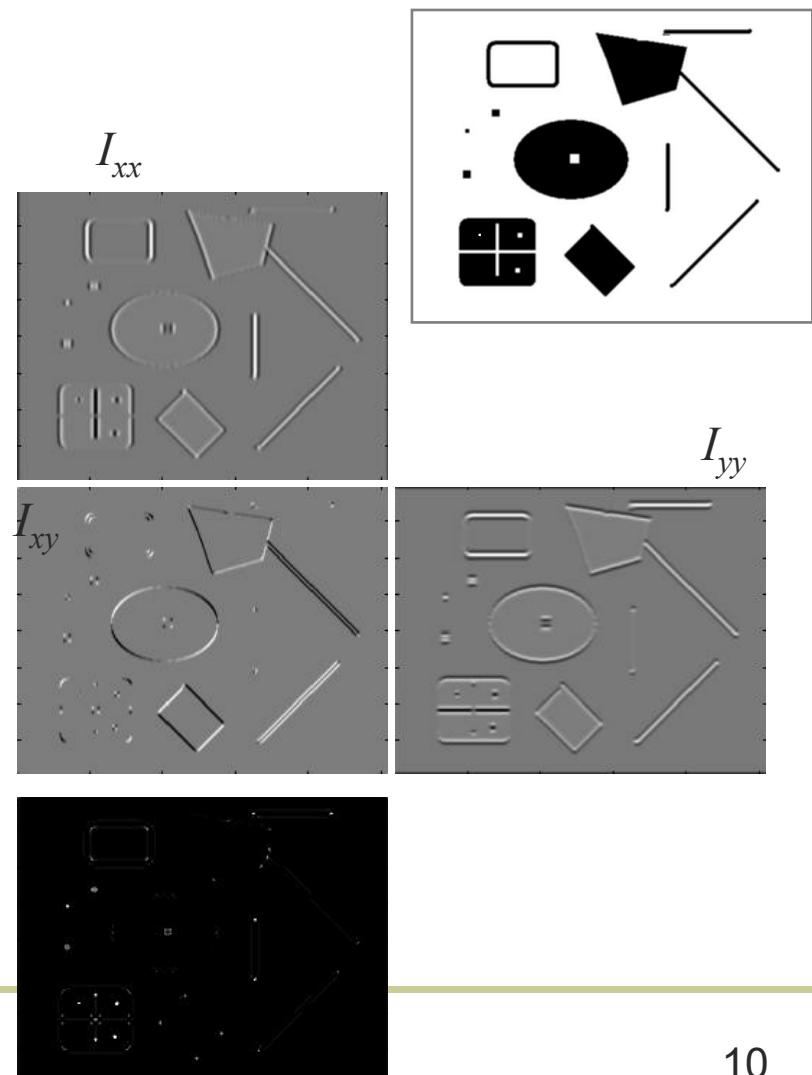
$$\text{trace } M = \lambda_1 + \lambda_2$$

Find maxima of determinant

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

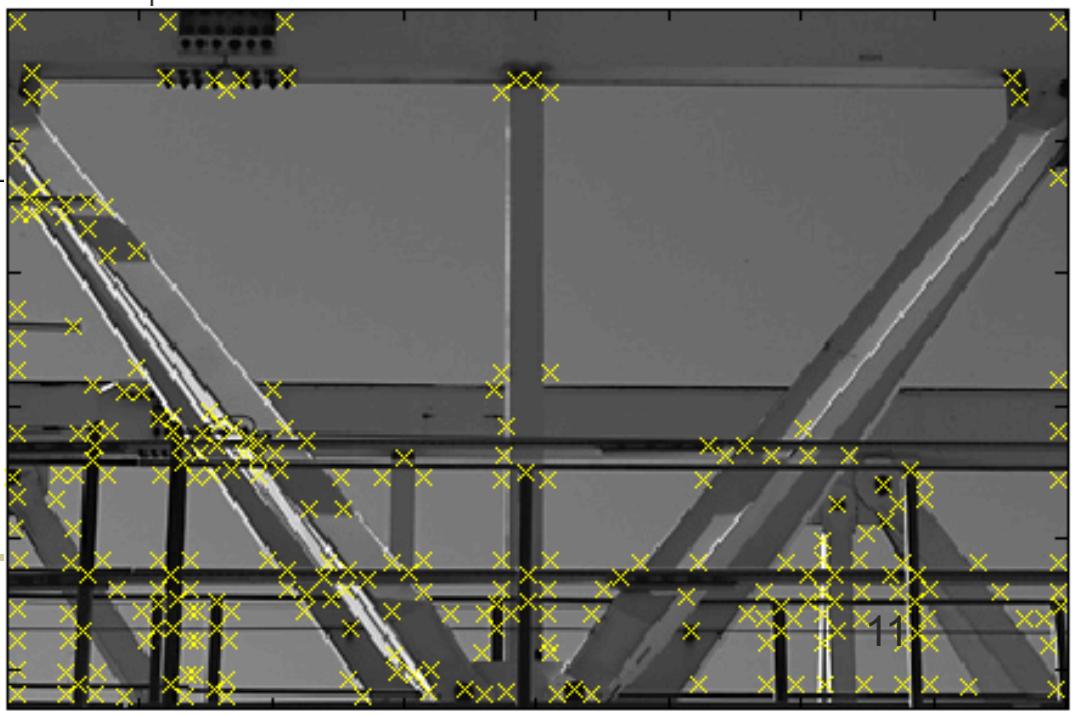
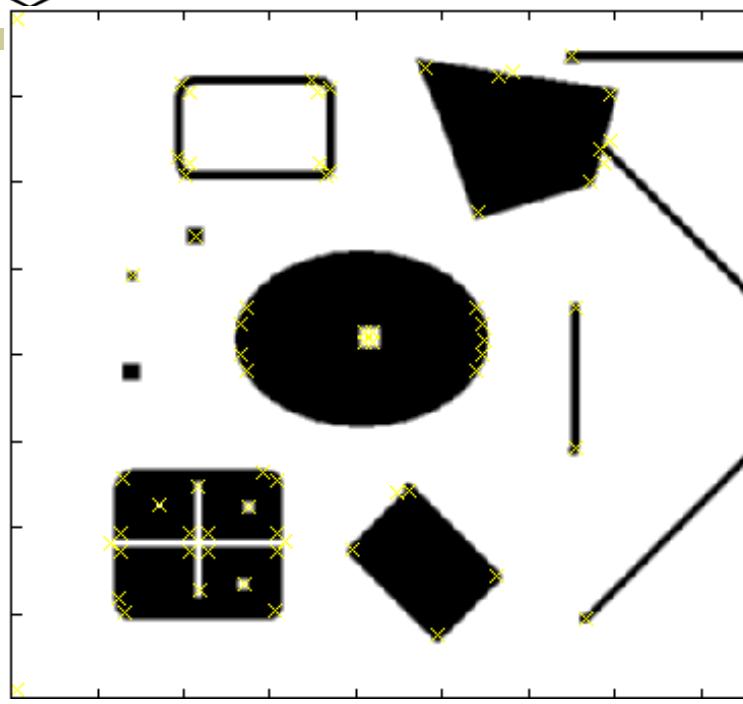
In Matlab:

$$2022/5/25 I_{xx}.*I_{yy} - (I_{xy})^2$$





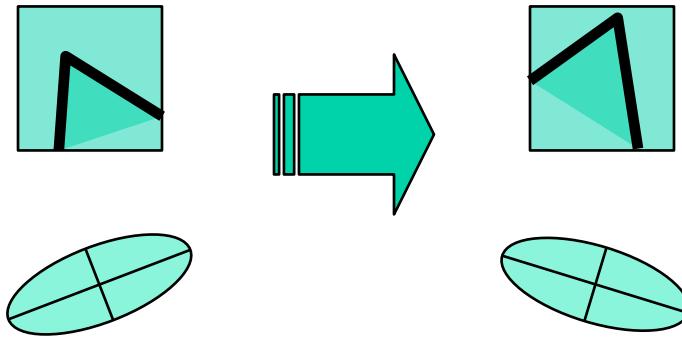
# Hessian Detector – Responses [Beaudet78]



*Effect:* Responses mainly on corners and strongly textured areas.



# Harris corner response is invariant to rotation



Ellipse rotates but its shape  
**(eigenvalues)** remains the same

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



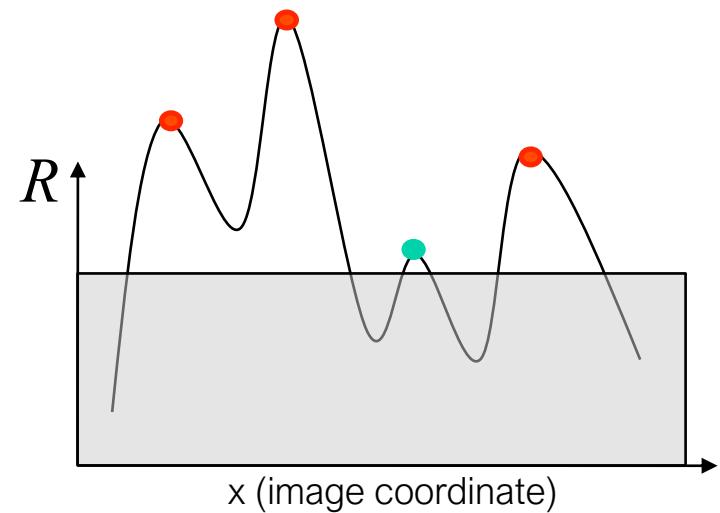
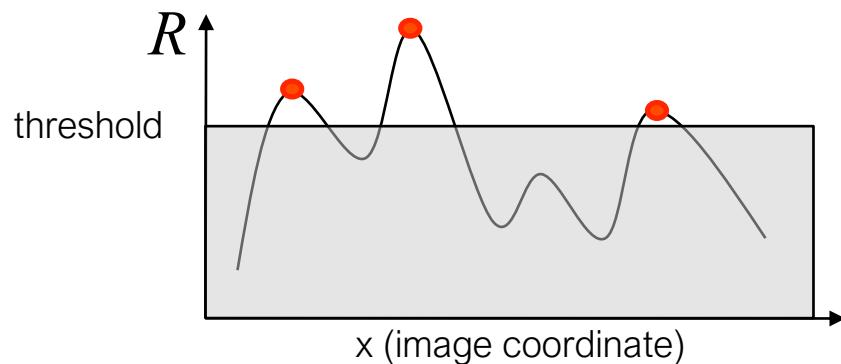
# Harris corner response is invariant to intensity changes



Partial invariance to *affine intensity* change

Only derivatives are used => invariance to intensity shift  $I \rightarrow I + b$

Intensity scale:  $I \rightarrow a I$

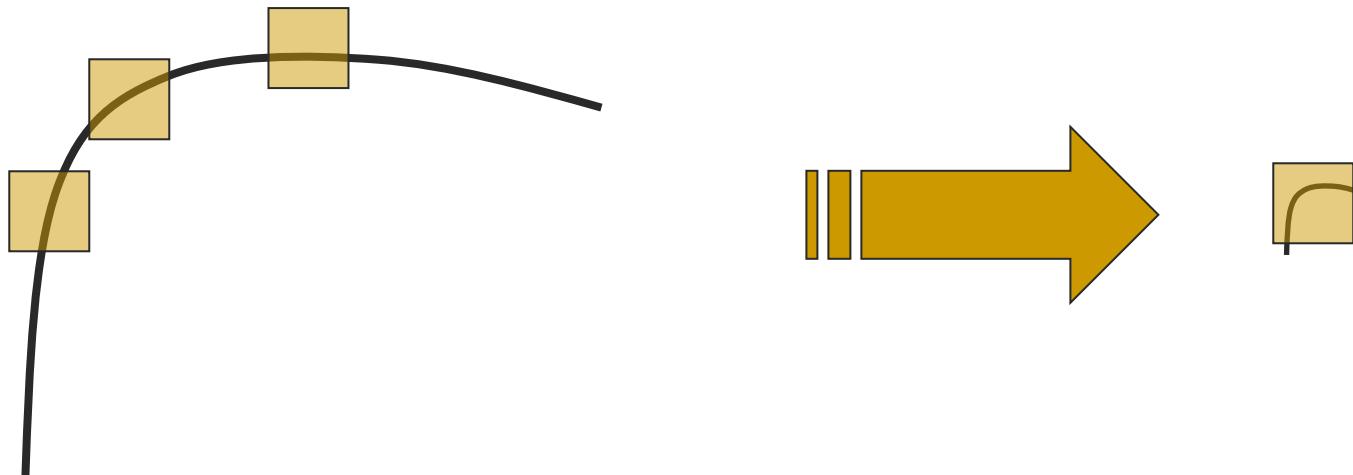




# Scale invariance?



■ Scale invariant? No



All points will be  
classified as edges

Corner !



# From points to regions



- The Harris and Hessian operators define interest points.

- Precise localization
  - High repeatability



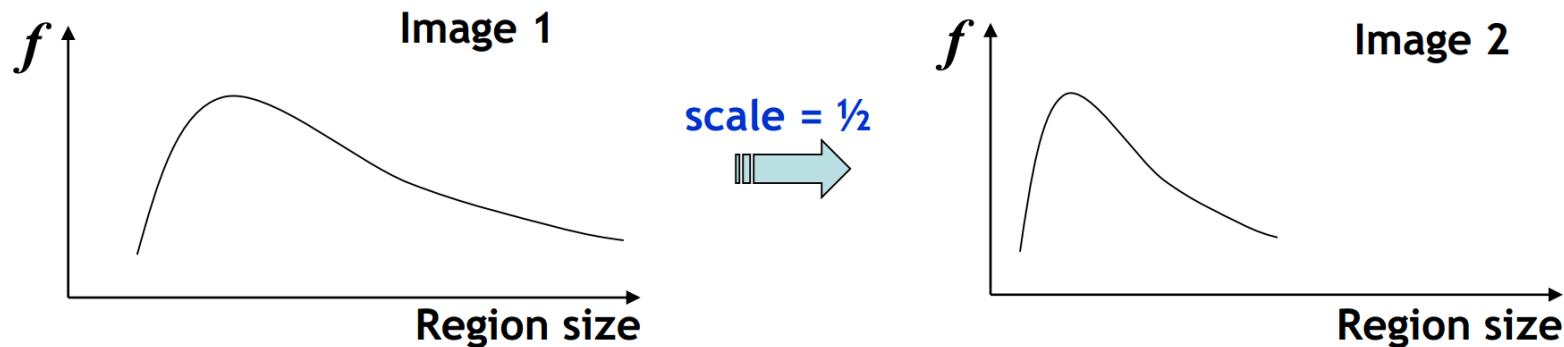
- In order to compare those points, we need to compute a descriptor over a region.
  - How can we define such a region in a scale invariant manner?
- *I.e. how can we detect scale invariant interest regions?*



# 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)



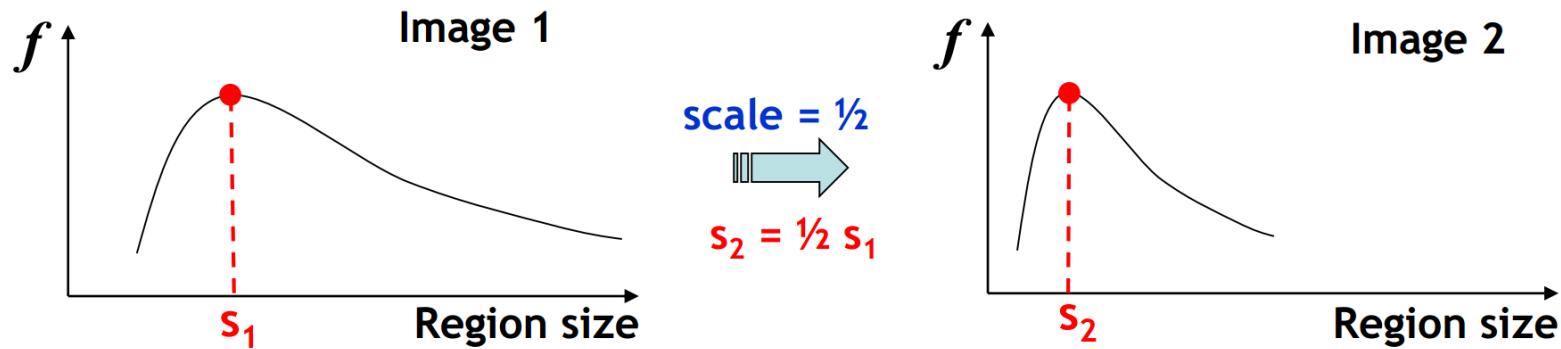


# Automatic scale selection



- 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!**

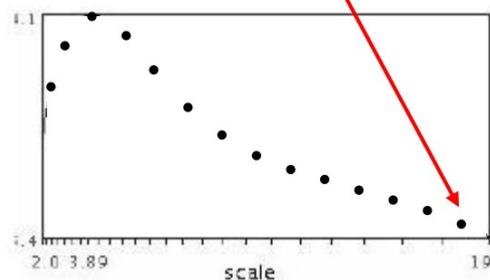




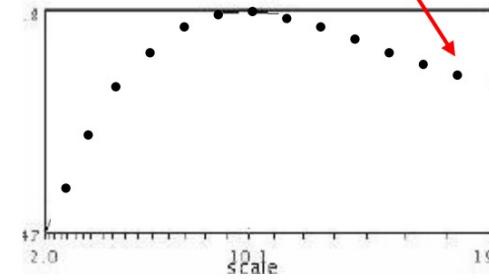
# Automatic scale selection



- Function responses for increasing scale (scale signature)



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



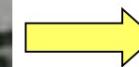
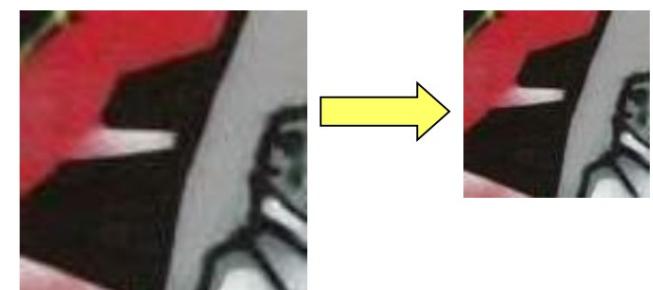
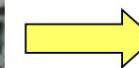
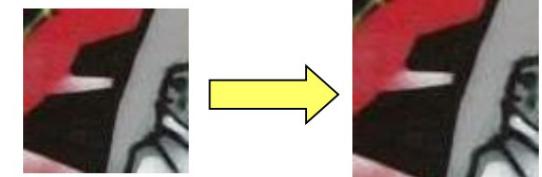
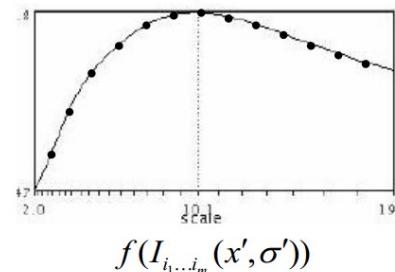
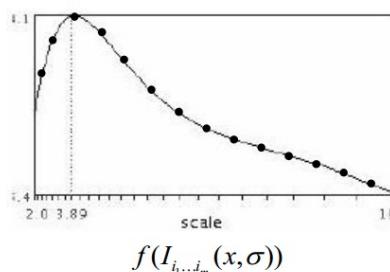
$f(I_{i_1 \dots i_m}(x', \sigma))$



# Automatic scale selection



- Normalize: Rescale to fixed size





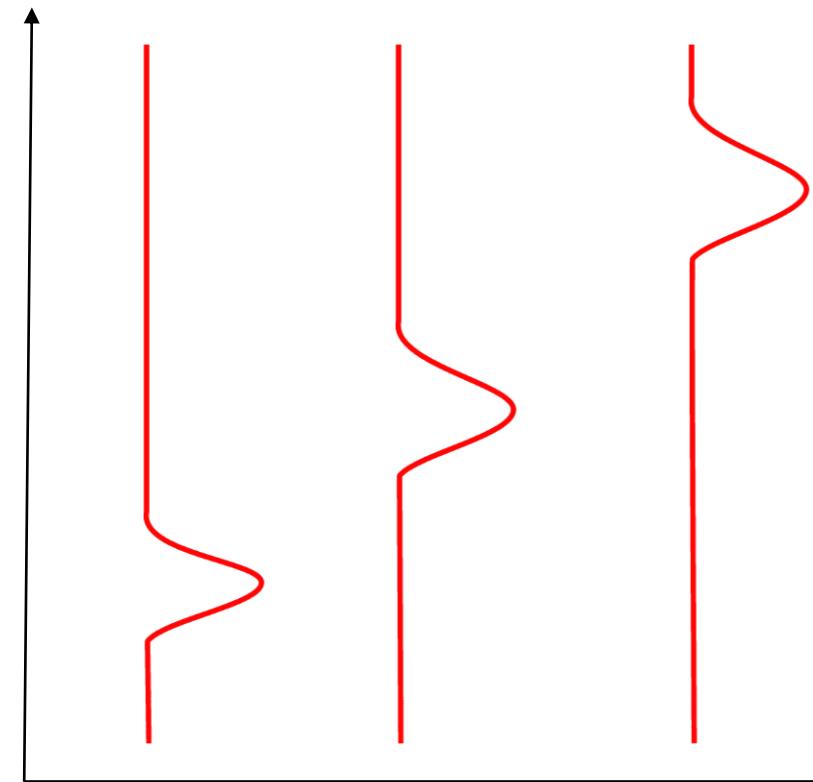
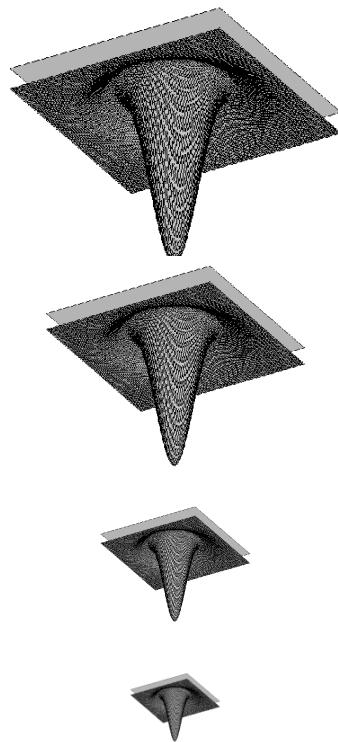
# Blob detection in 2D



- Laplacian-of-Gaussian = “blob” detector

$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

filter scales

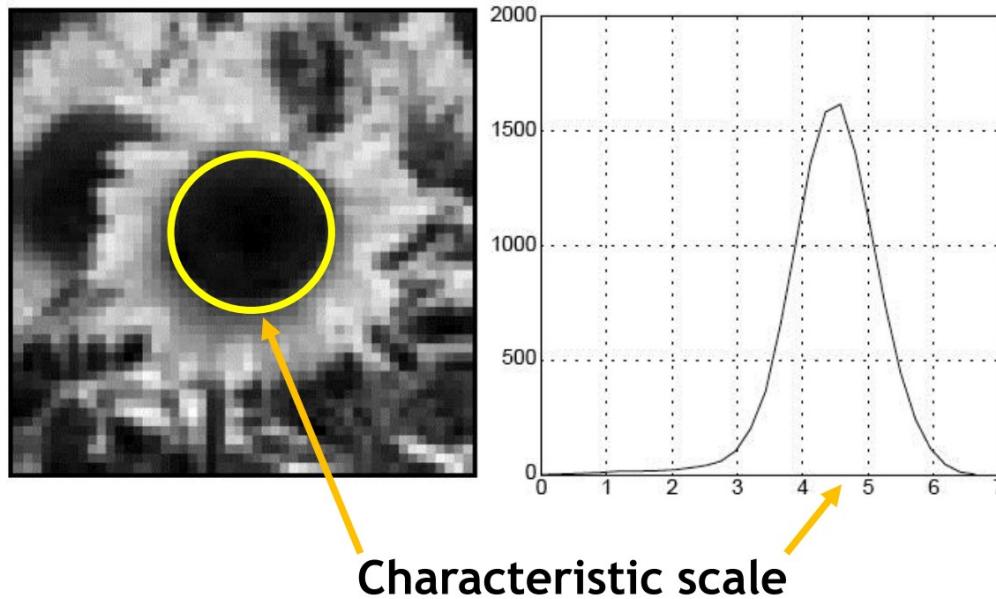




# Characteristic scale



- 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.

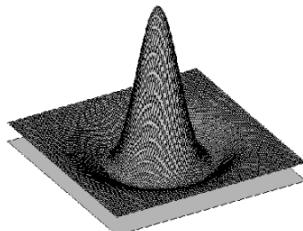


# Laplacian-of-Gaussian (LoG)



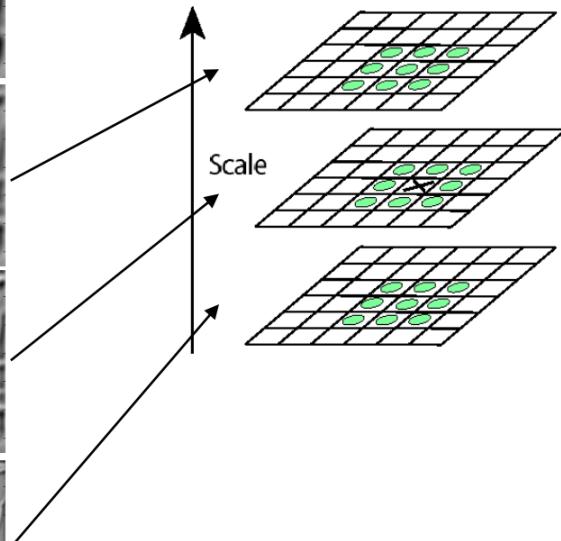
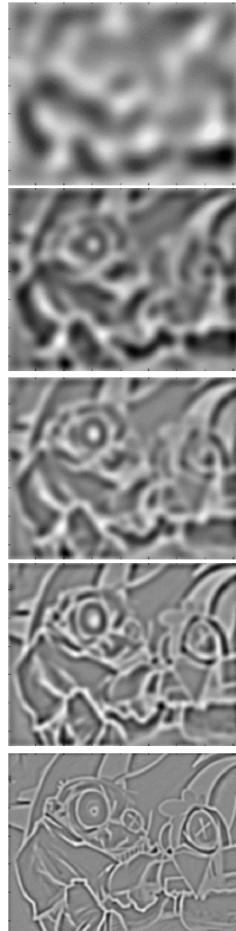
- Interest points:

- Local maxima in scale space of Laplacian-of-Gaussian



$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^3$$
$$\sigma^2$$
$$\sigma$$
$$\sigma^4$$
$$\sigma^5$$

A diagram illustrating the computation of the Laplacian of a Gaussian (LoG). It shows a flow from a 3D surface plot of a Gaussian peak down to a series of five grayscale images. The first image is the original input. Subsequent images show increasing levels of blurring and the resulting Laplacian response. Arrows indicate the progression: from the peak to  $\sigma^3$ , then to  $\sigma^2$ , then to  $\sigma$ , then to  $\sigma^4$ , and finally to  $\sigma^5$ .



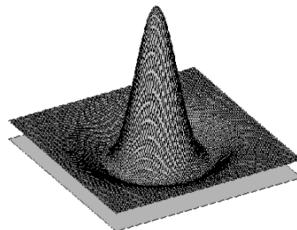


# Laplacian-of-Gaussian (LoG)



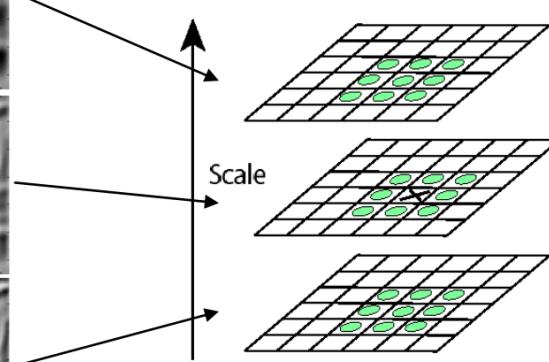
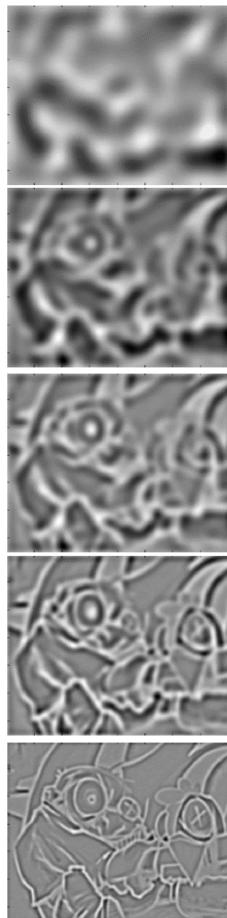
- **Interest points:**

- Local maxima in scale space of Laplacian-of-Gaussian



$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^3$$
$$\sigma^2$$
$$\sigma$$
$$\sigma^4$$
$$\sigma^5$$

A diagram illustrating the computation of the Laplacian of a Gaussian (LoG). It shows a sequence of five grayscale images representing the LoG at increasing scales ( $\sigma^5$  to  $\sigma$ ). Below the images, a mathematical equation indicates the sum of the second-order spatial derivatives ( $L_{xx}$  and  $L_{yy}$ ) results in a third-order polynomial relationship with scale ( $\sigma^3$ ). Arrows point from the images to the corresponding scale levels and the resulting polynomial equation.



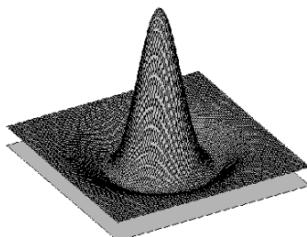


# Laplacian-of-Gaussian (LoG)



- Interest points:

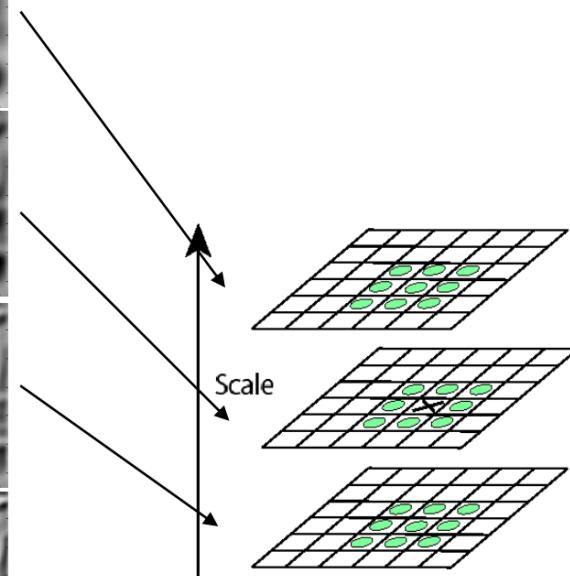
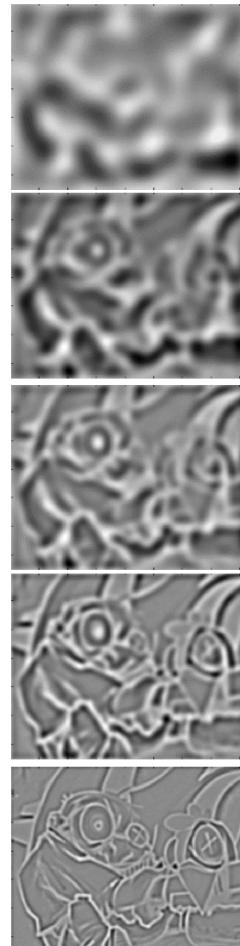
- Local maxima in scale space of Laplacian-of-Gaussian



$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma^3$$

$\sigma^5$   
 $\sigma^4$   
 $\sigma^3$   
 $\sigma^2$   
 $\sigma$

A diagram showing the relationship between the second derivatives of the image ( $L_{xx}$  and  $L_{yy}$ ) and the resulting scale space. Arrows point from the sum of the second derivatives to successive powers of  $\sigma$ , starting from  $\sigma^3$  and increasing by one at each step up to  $\sigma^5$ .



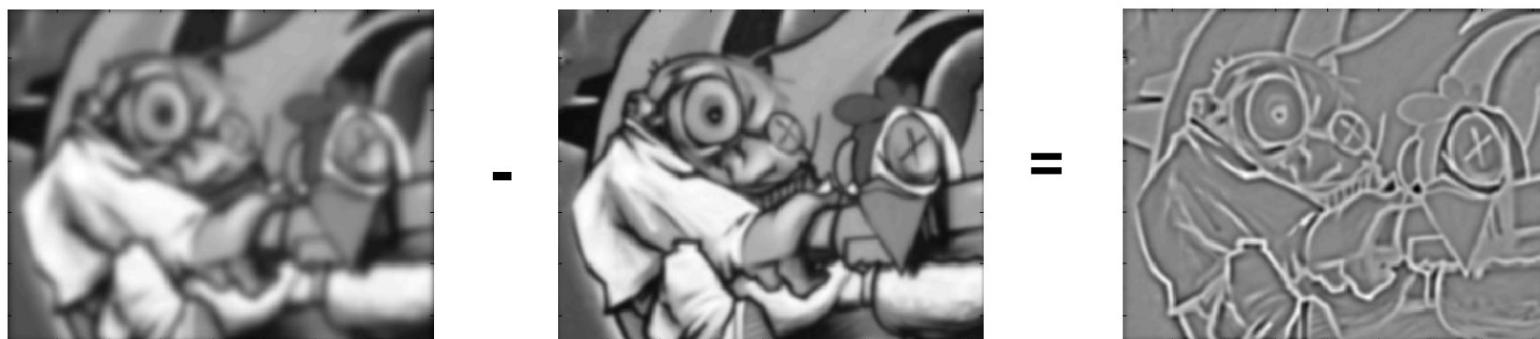
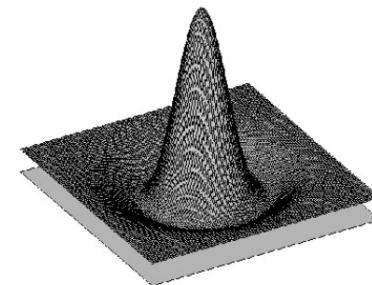
⇒ List of (x, y, σ)



# 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 2<sup>nd</sup> derivatives
  - Gaussians are computed anyway, e.g. in a Gaussian pyramid.





# DoG: Efficient implementation



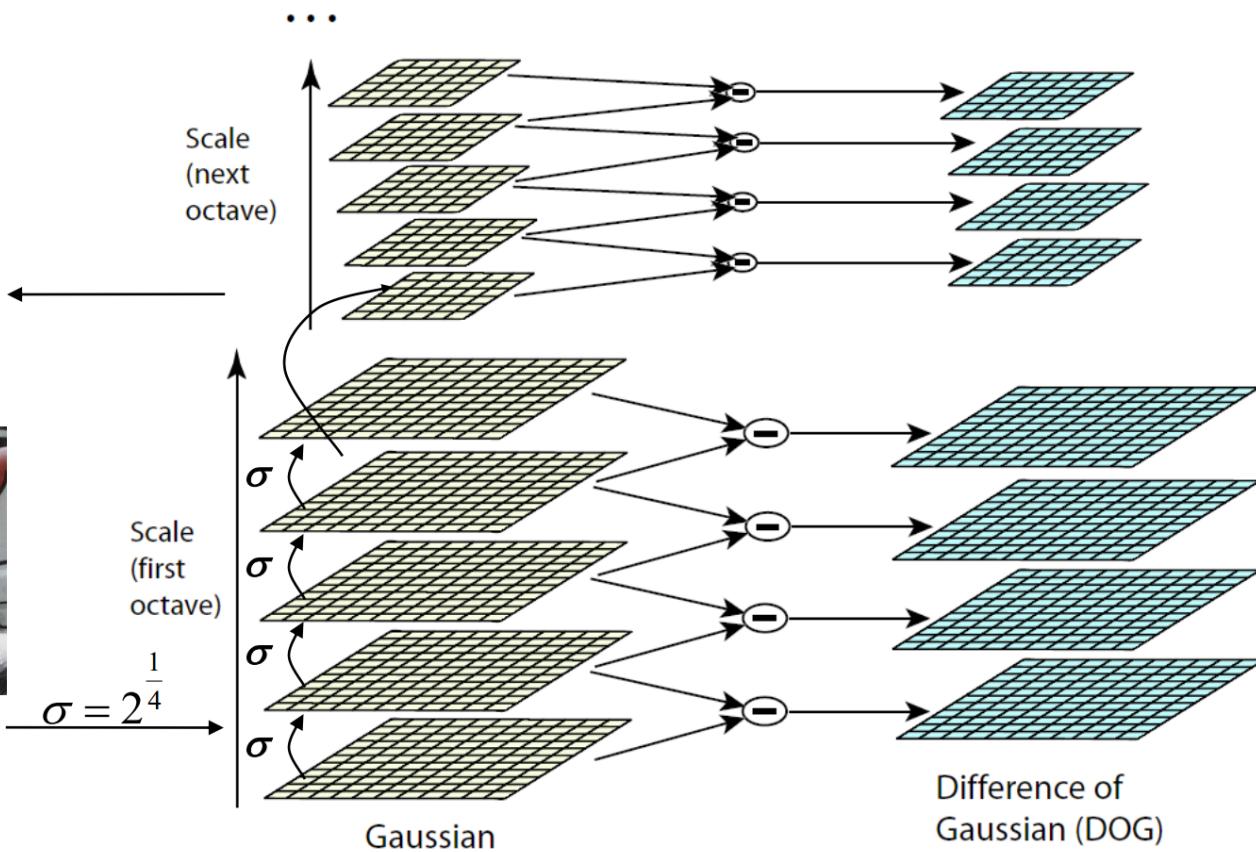
- Computation in Gaussian scale pyramid



*Sampling with  
step  $\sigma^4 = 2$*



*Original image*





# Results: Lowe's DoG

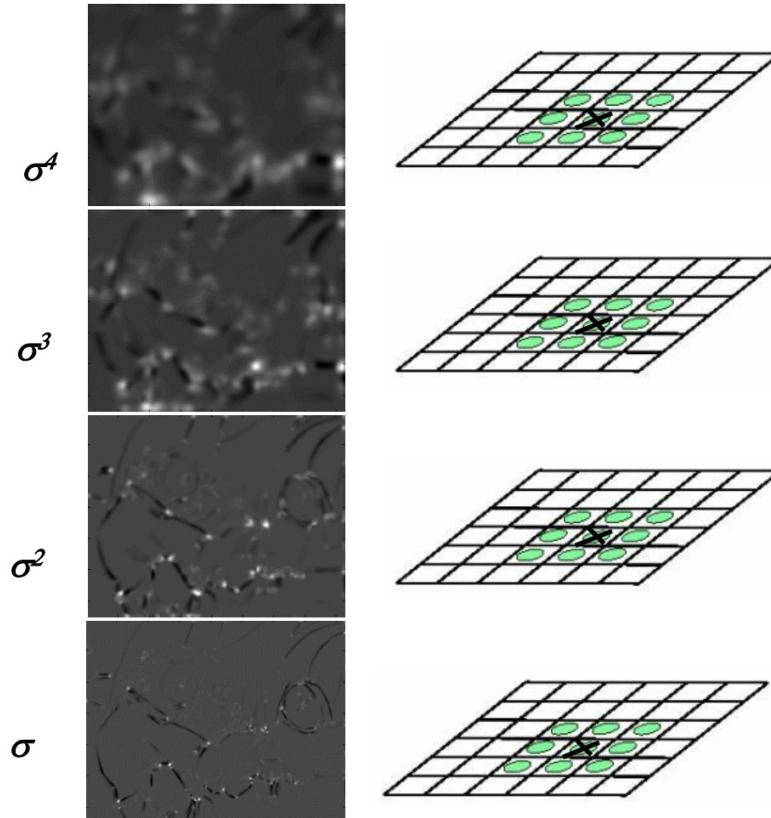




# Harris-Laplace [Mikolajczyk '01]



## 1. Initialization: Multiscale Harris corner detection



Computing Harris function

Detecting local maxima

Slide adapted from Krystian Mikolajczyk

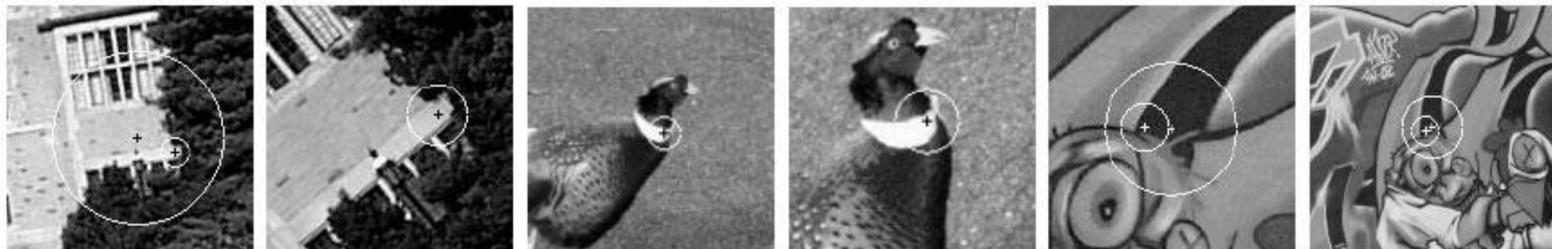
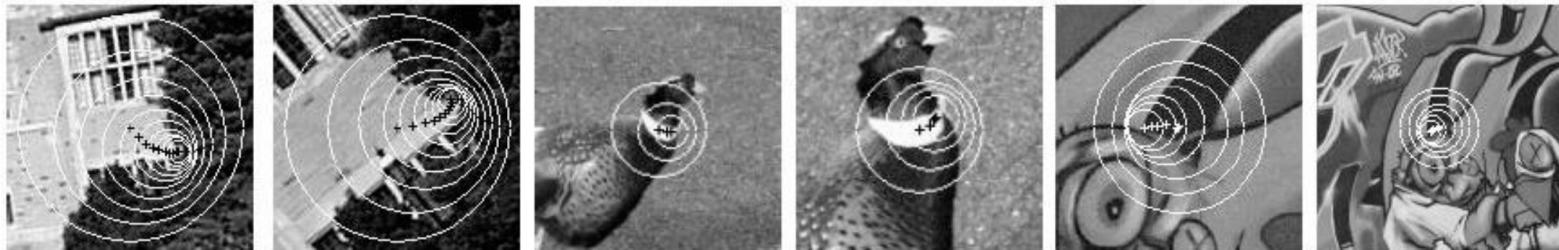


# Harris-Laplace [Mikolajczyk '01]



1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian  
(same procedure with Hessian  $\Rightarrow$  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).*



# Today's class



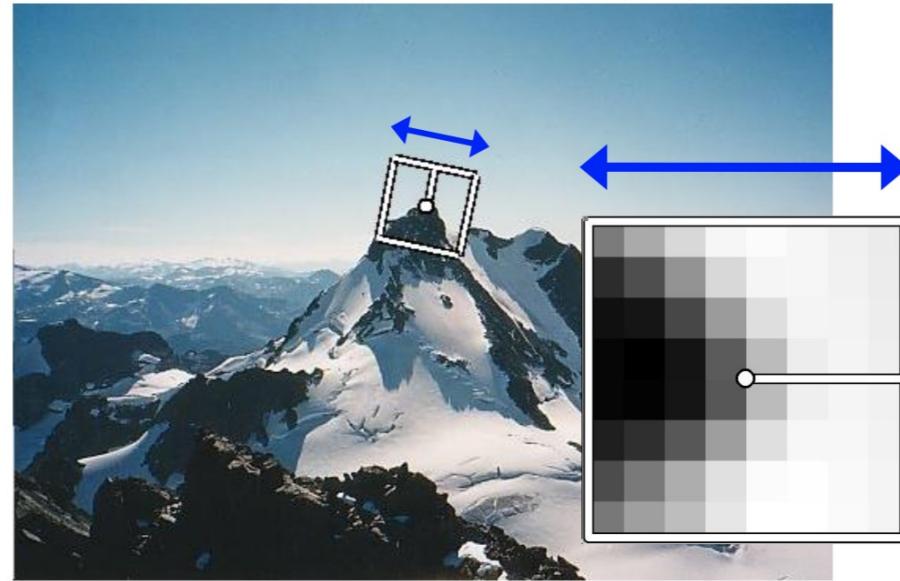
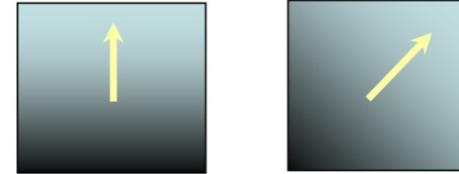
- Normalization
  - Orientation normalization
  - Affine invariant feature extraction
- Local descriptor
  - SIFT, SURF, GIST
- Binary descriptor
  - LBP, BRIEF
- CNN based descriptor
  - MatchNet, DeepCompare, DeepDesc, LIFT



# 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.

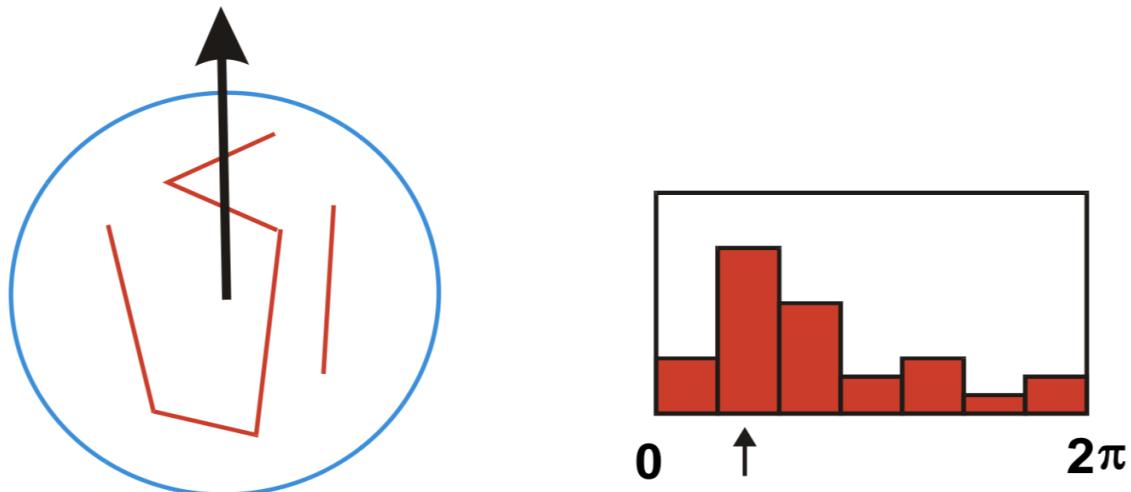




# Orientation Normalization



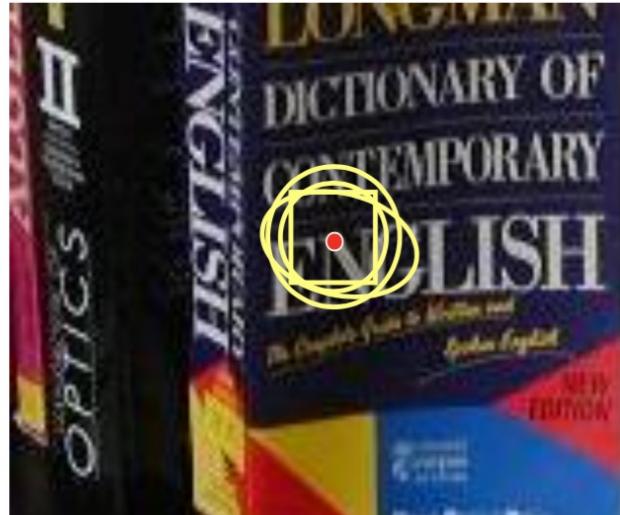
- Compute orientation histogram [Lowe, SIFT, 1999]
- Select dominant orientation
- Normalize: rotate to fixed orientation



Slide adapted from David Lowe



# The Need for Invariance



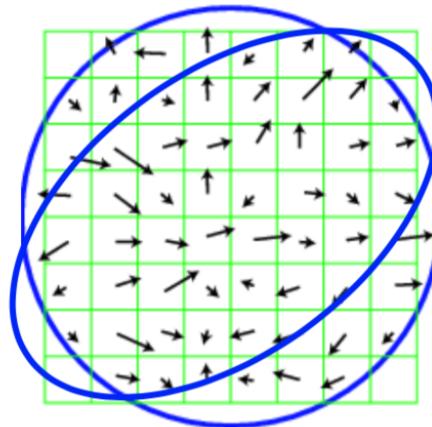
- Up to now, we had invariance to
  - Translation
  - Scale
  - Rotation
- Not sufficient to match regions under viewpoint changes
  - For this, we need also affine adaptation



# Affine Adaption

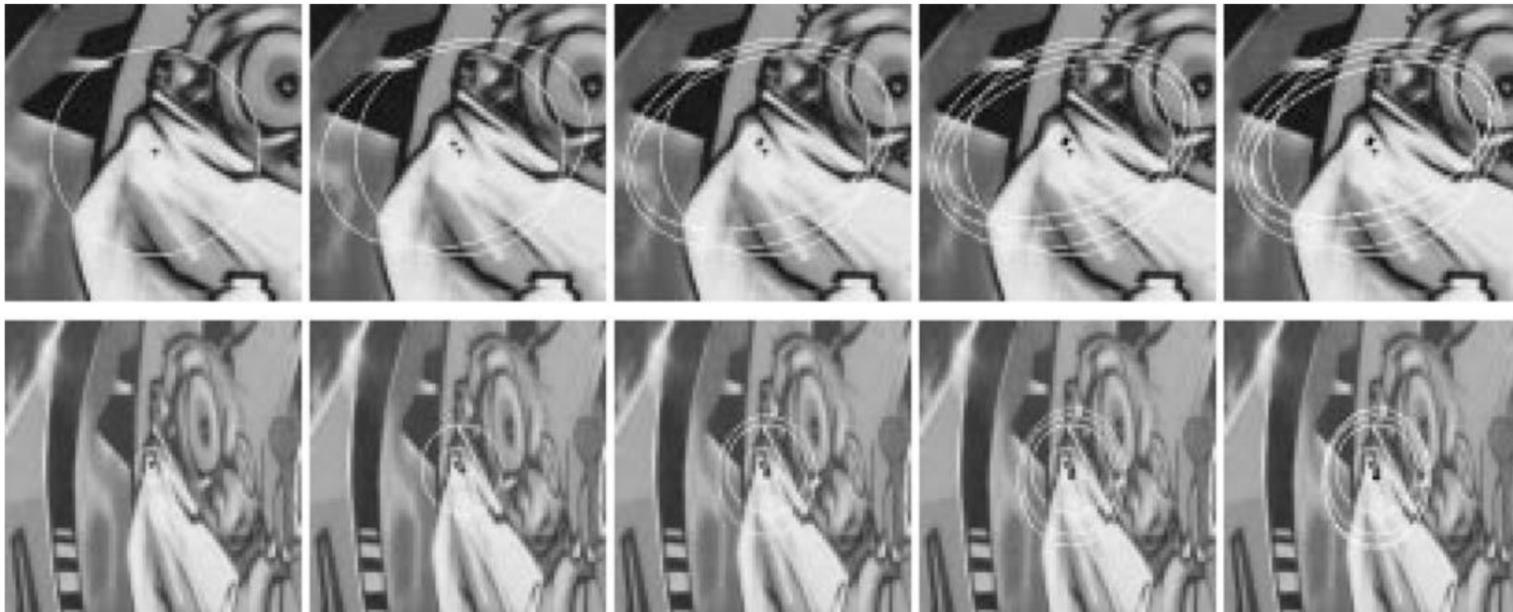


- **Problem:**
  - Determine the characteristic shape of the region.
  - Assumption: shape can be described by “local affine frame”.
- **Solution: iterative approach**
  - Use a circular window to compute second moment matrix.
  - Compute eigenvectors to adapt the circle to an ellipse.
  - Recompute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





# Iterative Adaption

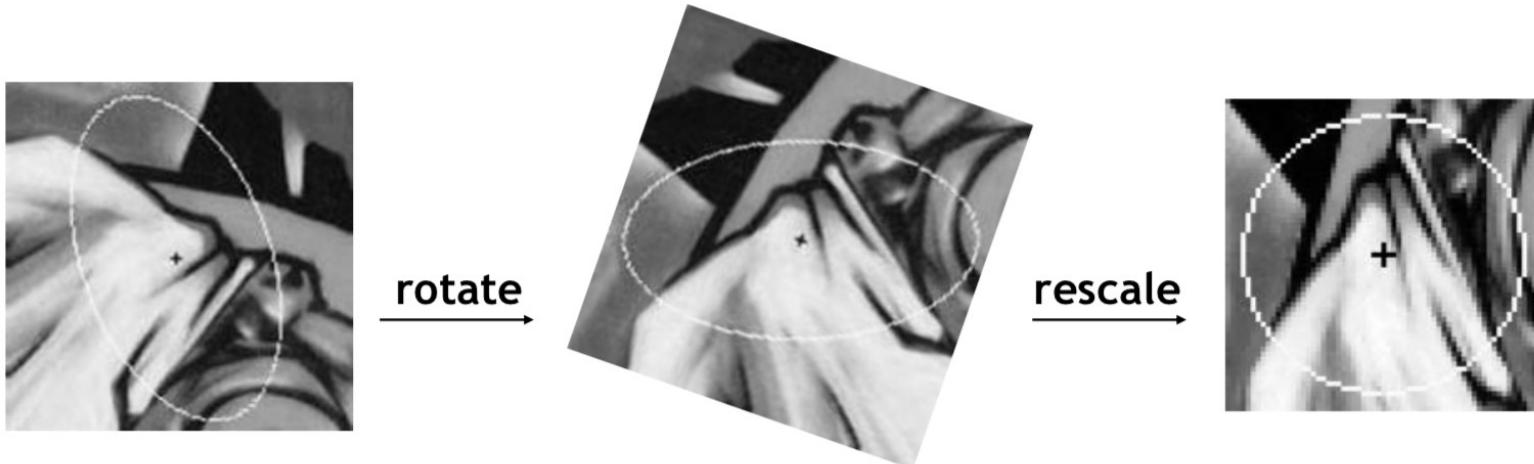


1. Detect keypoints, e.g. multi-scale Harris
2. Automatically select the scales
3. Adapt affine shape based on second order moment matrix
4. Refine point location

K. Mikolajczyk and C. Schmid, [Scale and affine invariant interest point detectors](#), 56  
IJCV 60(1):63-86, 2004. Slide credit: Tinne Tuytelaars



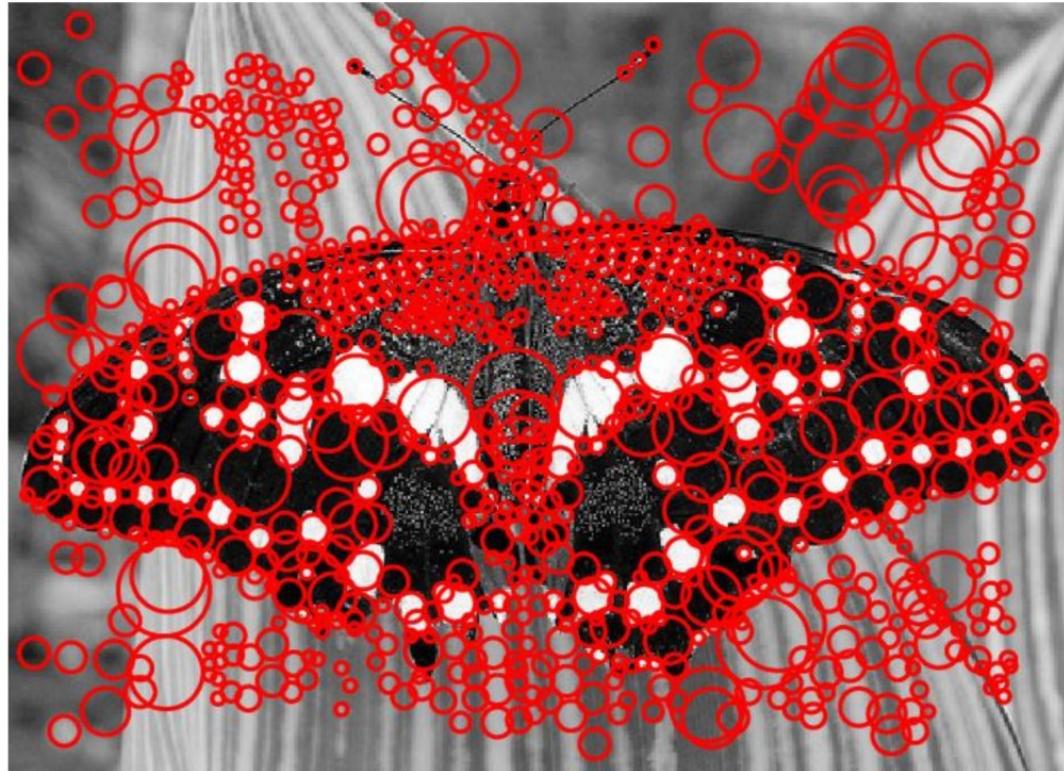
# Affine Normalization



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



# Affine Adaption Example



Scale-invariant regions (blobs)



# Affine Adaption Example



Affine-adapted blobs



# Summary: Affine Invariance



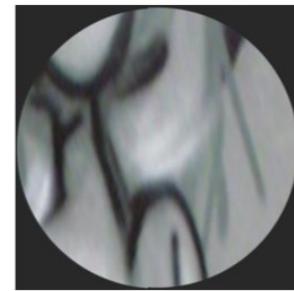
Extract affine regions



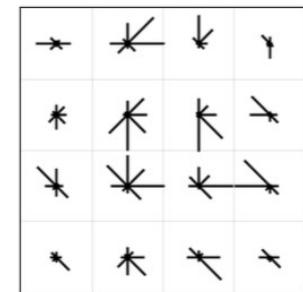
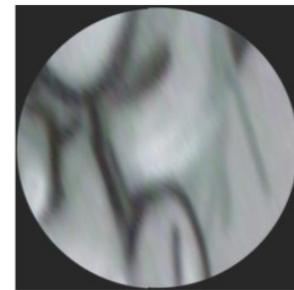
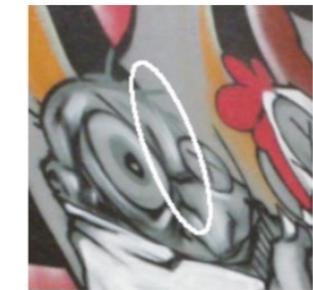
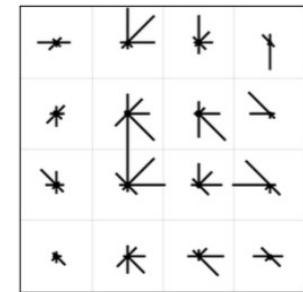
Normalize regions



Eliminate rotational ambiguity



Compare descriptors

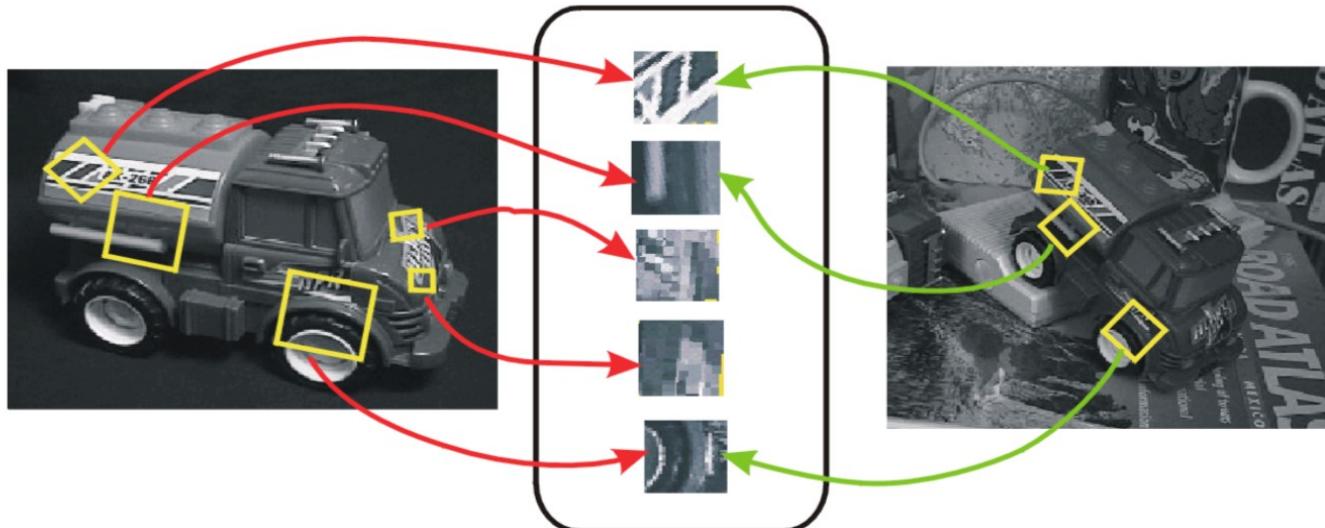




# Invariance vs. Covariance



- **Invariance:**
  - $\text{features}(\text{transform}(\text{image})) = \text{features}(\text{image})$
- **Covariance:**
  - $\text{features}(\text{transform}(\text{image})) = \text{transform}(\text{features}(\text{image}))$



**Covariant detection  $\Rightarrow$  invariant description**



# Today's class



- Normalization
  - Orientation normalization
  - Affine invariant feature extraction
- Local descriptor
  - SIFT, SURF, GIST
- Binary descriptor
  - LBP, BRIEF
- CNN based descriptor
  - MatchNet, DeepCompare, DeepDesc, LIFT

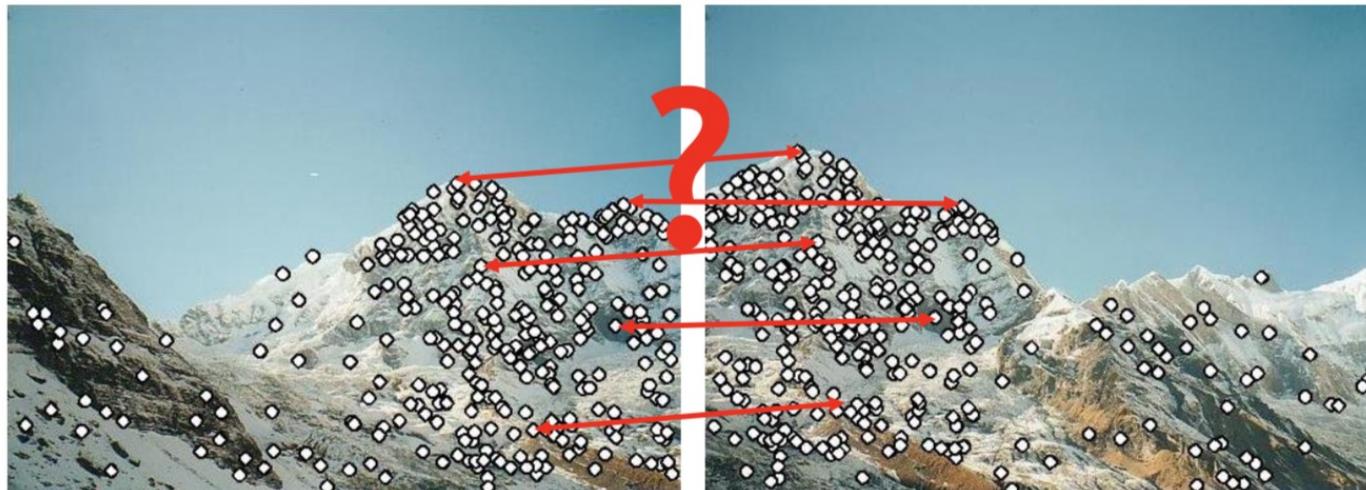


# Local Descriptor



- We know how to detect points
- Next question:

**How to *describe* them for matching?**

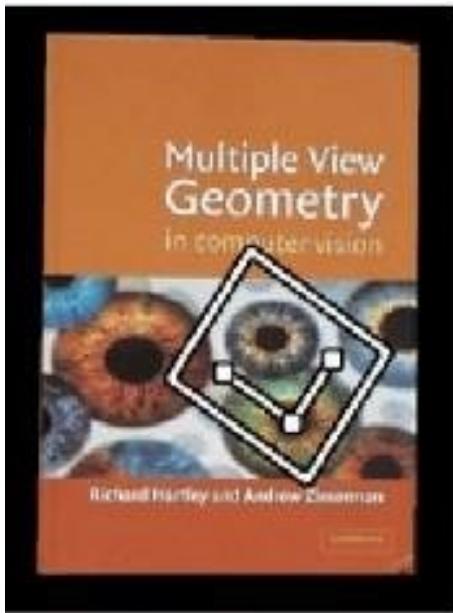


**Point descriptor should be:**

1. Invariant
2. Distinctive

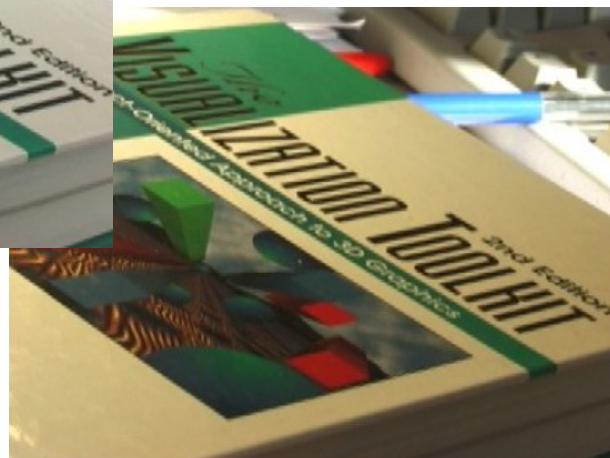
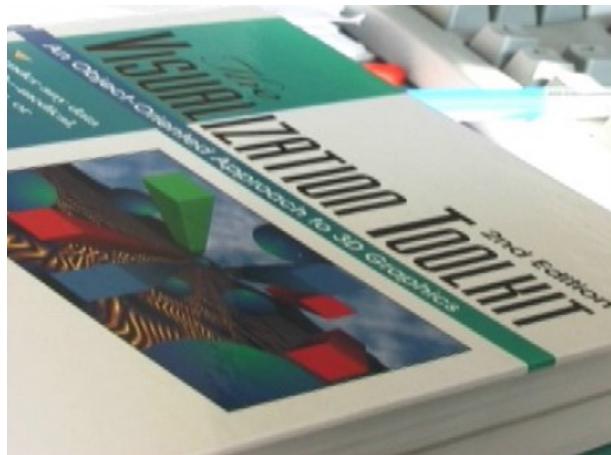


# Geometric transformations



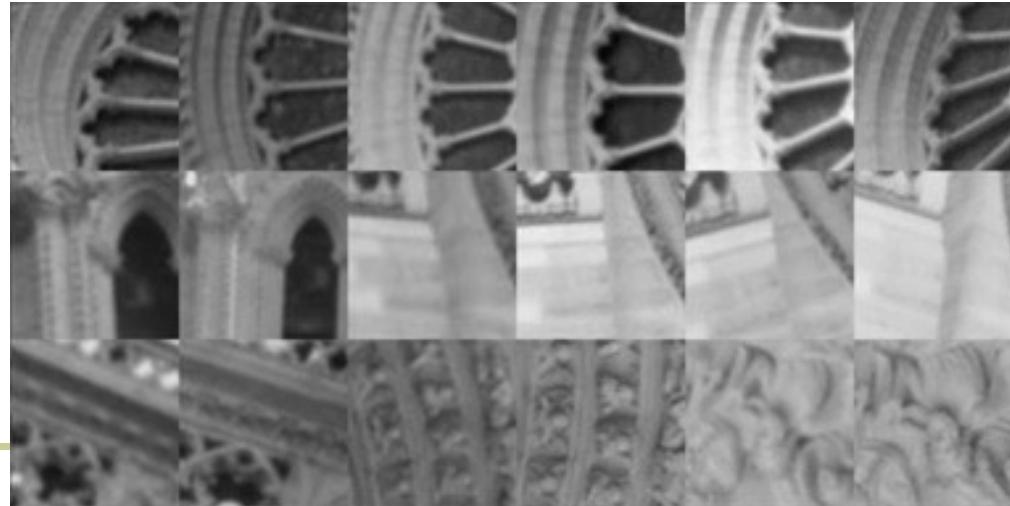


# Photometric transformations





*What is the best descriptor for an image feature?*





# Local Descriptor

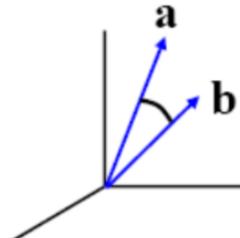


- Simplest descriptor: list of intensities within a patch.
- What is this going to be invariant to?

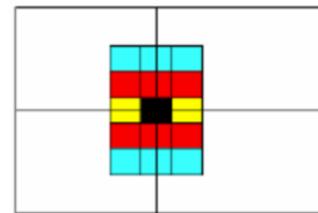
Perfectly fine if geometry and appearance is unchanged  
(a.k.a. template matching)

Write regions as vectors

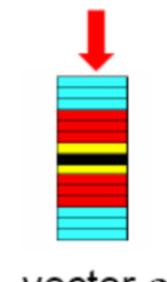
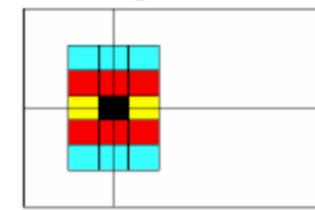
$$A \rightarrow \mathbf{a}, \quad B \rightarrow \mathbf{b}$$



region A



region B



vector  $\mathbf{a}$



vector  $\mathbf{b}$



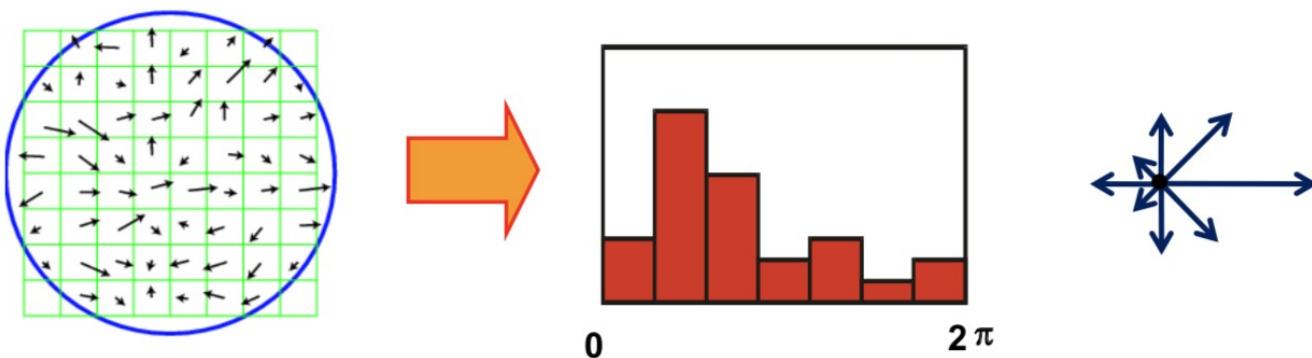
# Feature Descriptor



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



- Solution: histograms

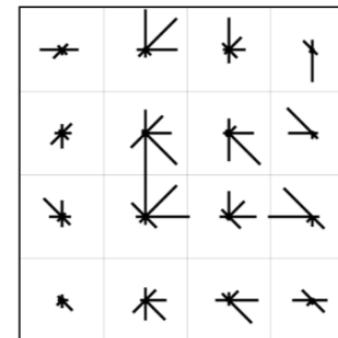
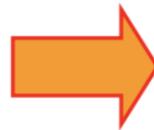
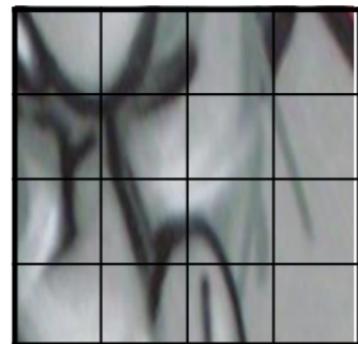




# Feature Descriptor: SIFT



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



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



# SIFT Properties



- 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](http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT)



Slide credit: Steve Seitz





# Summary: SIFT

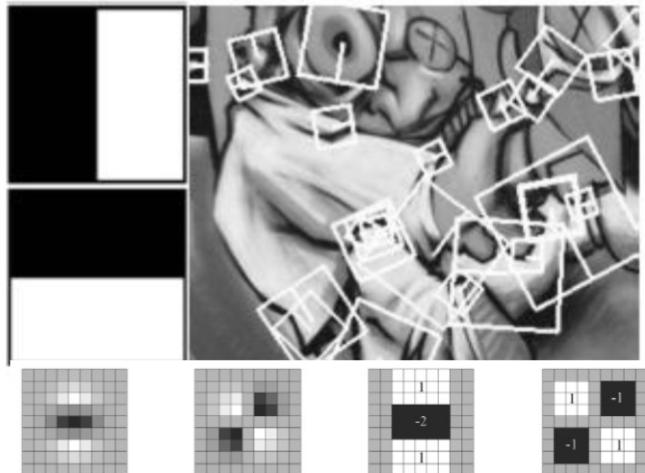


- One image yields:
  - $n$  2D points giving positions of the patches
    - $[n \times 2 \text{ matrix}]$
  - $n$  scale parameters specifying the size of each patch
    - $[n \times 1 \text{ vector}]$
  - $n$  orientation parameters specifying the angle of the patch
    - $[n \times 1 \text{ vector}]$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





# Feature Descriptor: 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>

Well received! 20745

- More than 8000 citations.
- CVIU Most Cited Paper
- Koenderink Prize of ECCV'16

## GPU implementation available

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

Herbert Bay et al., SURF: Speeded Up Robust Feature, in ECCV 2006



# SURF: Keypoint Detection



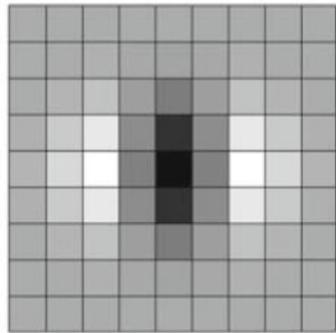
- Uses determinant of Hessian matrix
- Approximate 2nd derivatives in Hessian matrix with box filters

$$H(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \longrightarrow \hat{H}(x, \sigma) = \begin{bmatrix} D_{xx}(x, \sigma) & D_{xy}(x, \sigma) \\ D_{xy}(x, \sigma) & D_{yy}(x, \sigma) \end{bmatrix}$$

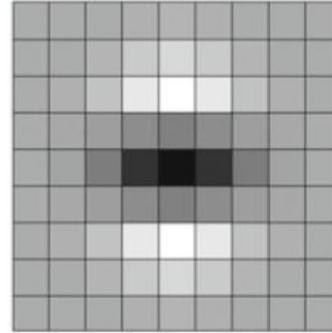
$$R(x, \sigma) = L_{xx}L_{yy} - L_{xy}^2 \approx D_{xx}D_{yy} - (0.9D_{xy}^2)$$



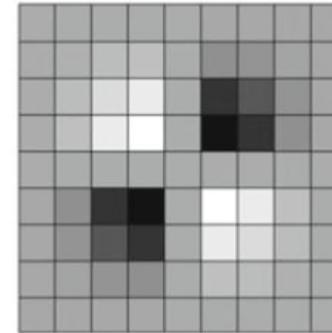
# SURF: Keypoint Detection



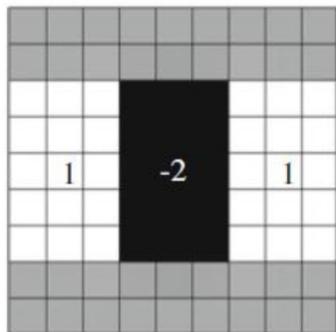
$L_{xx}$



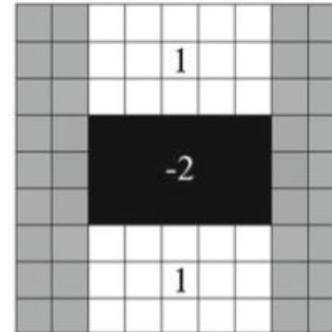
$L_{yy}$



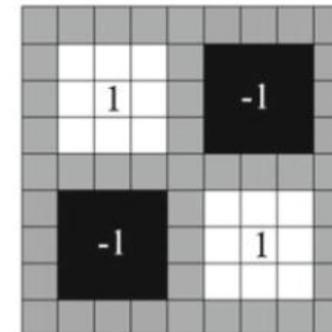
$L_{xy}$



$D_{xx}$



$D_{yy}$



$D_{xy}$



# Integral Image



$I(x, y)$			$A(x, y)$		
original image			integral image		
1	5	2	1	6	8
2	4	1	3	12	15
2	1	1	5	15	19

$$A(x, y) = \sum_{x' \leq x, y' \leq y} I(x', y')$$



# Integral Image



$I(x, y)$			$A(x, y)$		
original image			integral image		
1	5	2	1	6	8
2	4	1	3	12	15
2	1	1	5	15	19

$$A(x, y) = \sum_{x' \leq x, y' \leq y} I(x', y')$$

Can find the **sum** of any block using **3** operations

$$A(x_1, y_1, x_2, y_2) = A(x_2, y_2) - A(x_1, y_2) - A(x_2, y_1) + A(x_1, y_1)$$



What is the sum of the bottom right 2x2 square?

$$A(x_1, y_1, x_2, y_2) = A(x_2, y_2) - A(x_1, y_2) - A(x_2, y_1) + A(x_1, y_1)$$

$I(x, y)$

1	5	2
2	4	1
2	1	1

image

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

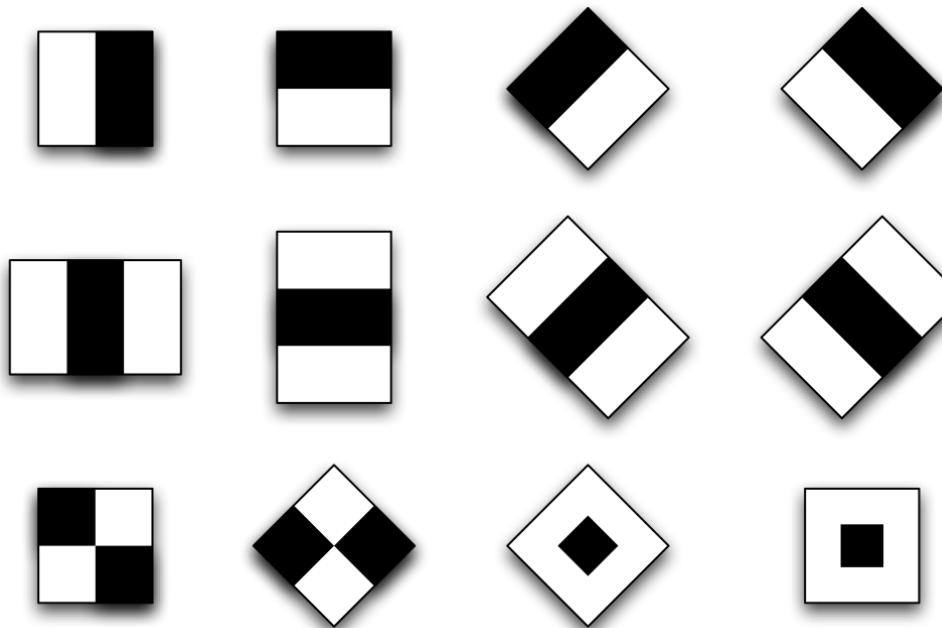
$$\begin{aligned} A(1, 1, 3, 3) &= A(3, 3) - A(1, 3) - A(3, 1) + A(1, 1) \\ &= 19 - 8 - 5 + 1 \\ &= 7 \end{aligned}$$



# Haar Wavelets (actually, Haar-like features)



Use responses of a bank of filters as a descriptor



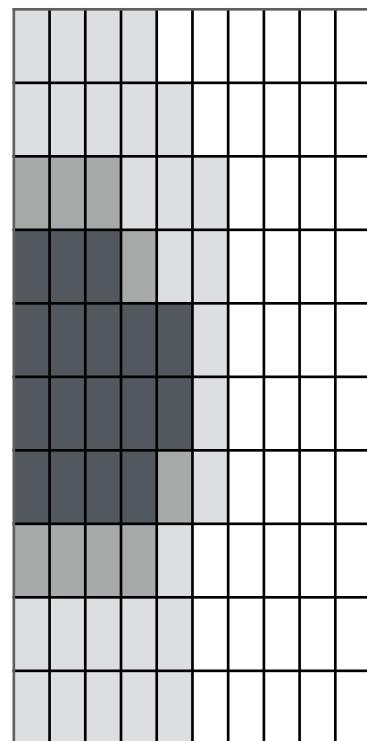
How to compute Haar wavelet responses **efficiently** (in constant time) with integral images



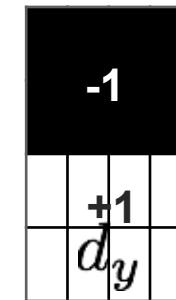
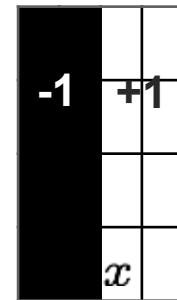
# Haar wavelet responses can be computed with filtering



image patch



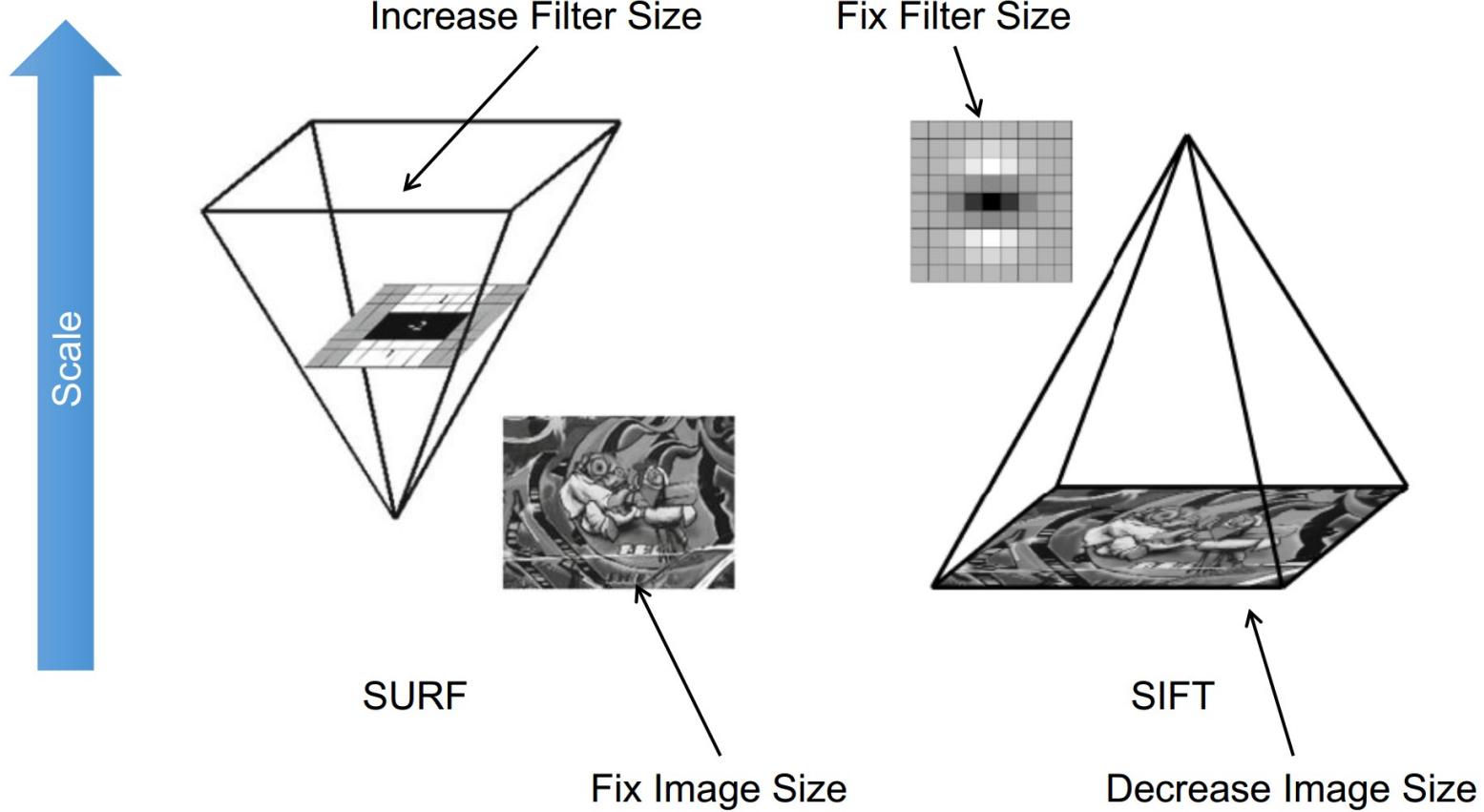
Haar wavelets filters



Haar wavelet responses can be computed  
**efficiently** (in constant time) with  
integral images

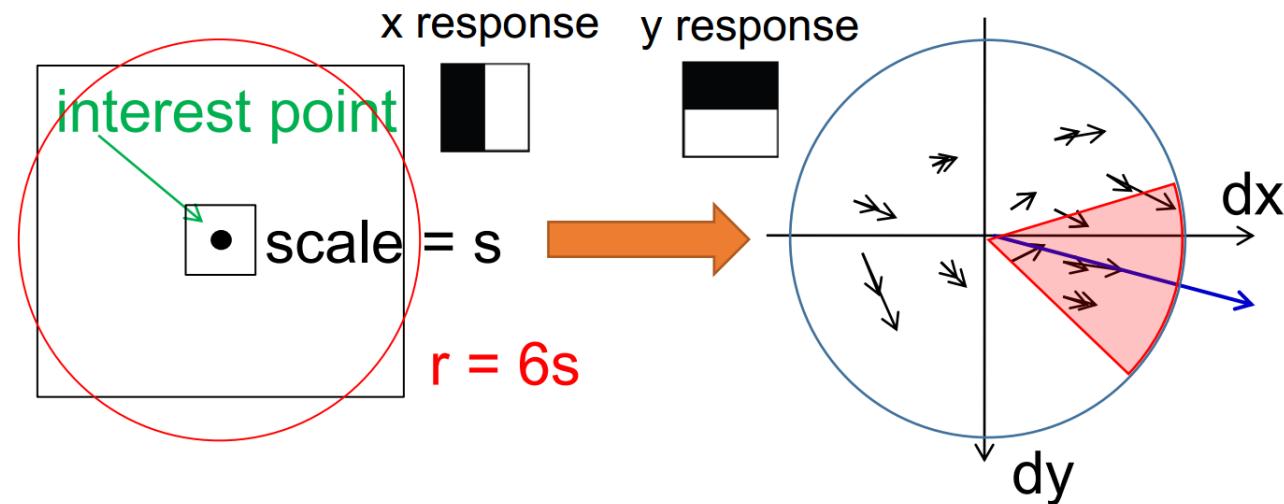


# SURF vs. SIFT: Scale Space





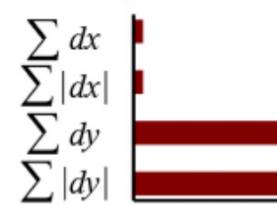
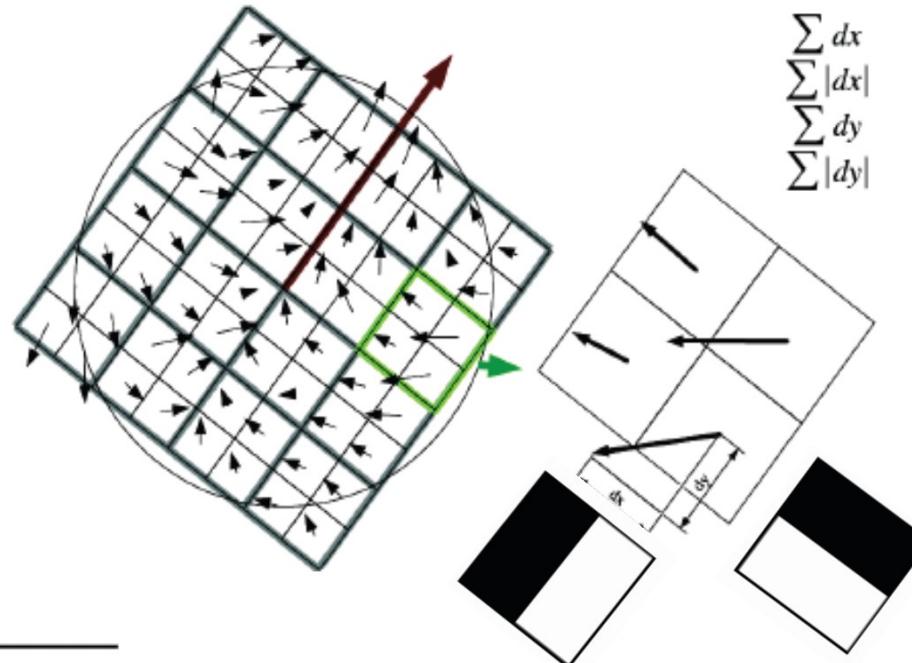
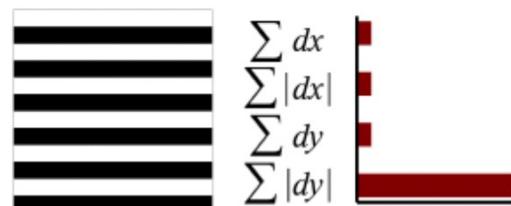
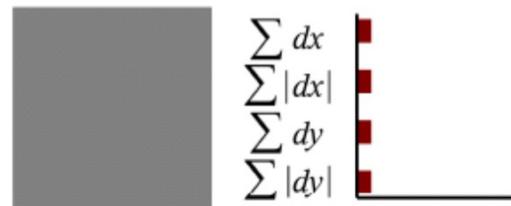
# Dominant Orientation Estimation



- The Haar wavelet responses (x and y) are represented as vectors.
- Sum all responses within a sliding orientation window covering an angle of 60 degree.
- The longest vector is the dominant orientation



# SURF: Descriptor Extraction

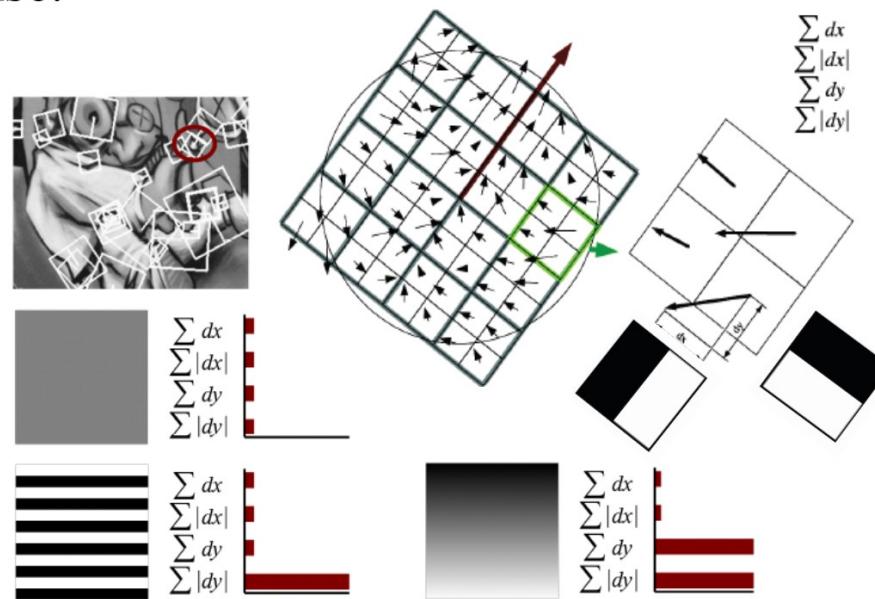




# SURF: Descriptor Extraction



1. Split the interest region ( $20s \times 20s$ ) into  $4 \times 4$  square sub-regions.
2. Calculate Haar wavelet responses  $dx$  and  $dy$ , and weight the responses with a Gaussian kernel.
3. Sum the response over each sub-region for  $dx$  and  $dy$ , then sum the absolute value of response.
4. Concatenate summation results in all sub-regions, forming a 64D SURF descriptor.





# Summary: SURF

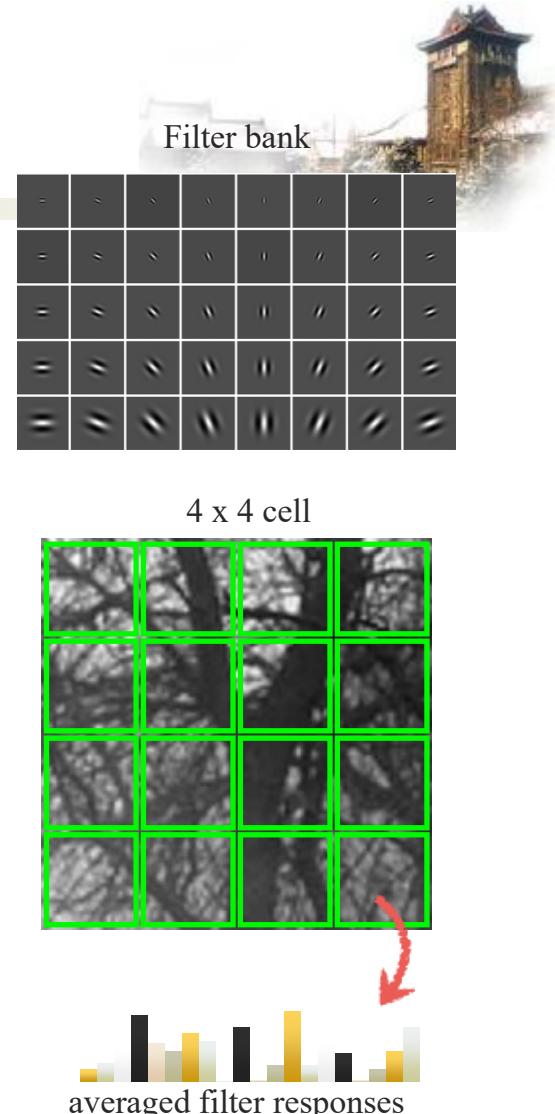


- Approximation yet can be computed much faster.
  - relying on **integral images** for image convolutions
  - building on the strengths of the **leading existing detectors and descriptors**
  - **simplifying** these methods to the essential
- Combination of novel detection, description, and matching steps.



# GIST

1. Compute filter responses (filter bank of Gabor filters)
2. Divide image patch into  $4 \times 4$  cells
3. Compute filter response averages for each cell
4. Size of descriptor is  $4 \times 4 \times N$ , where  $N$  is the size of the filter bank



Oliva, A., & Torralba, A., Modeling the Shape of the Scene: a Holistic Representation of the Spatial Envelope. IJCV 2001.

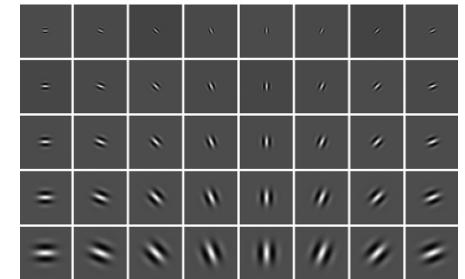


# Summary: GIST

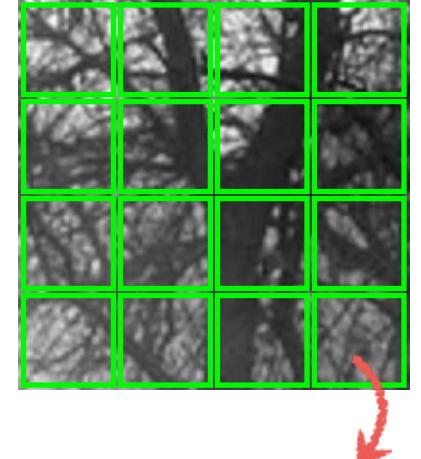


1. Compute filter responses (filter bank of Gabor filters)
2. Divide image patch into  $4 \times 4$  cells
3. Compute filter response averages for each cell
4. Size of descriptor is  $4 \times 4 \times N$ , where  $N$  is the size of the filter bank

Filter bank



$4 \times 4$  cell



*What is the GIST descriptor encoding?*

Rough spatial distribution of image gradients



averaged filter responses



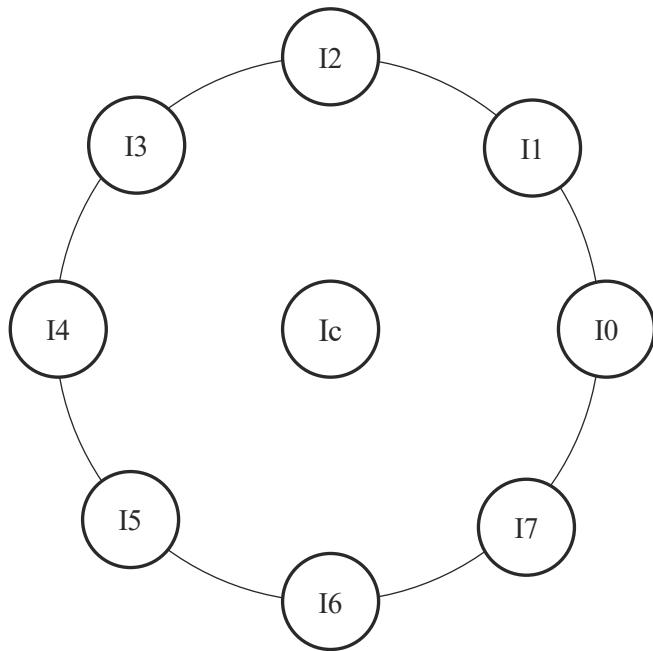
# Today's class



- Normalization
  - Orientation normalization
  - Affine invariant feature extraction
- Local Descriptor
  - SIFT, SURF, GIST
- Binary descriptor
  - LBP, BRIEF
- CNN Based descriptor
  - MatchNet, DeepCompare, DeepDesc, LIFT



# Center Symmetric Local Binary Patterns



$$\text{CS-LBP} = \text{sign}(I_0 - I_c)2^0 + \text{sign}(I_1 - I_c)2^1 + \text{sign}(I_2 - I_c)2^2 + \text{sign}(I_3 - I_c)2^3$$

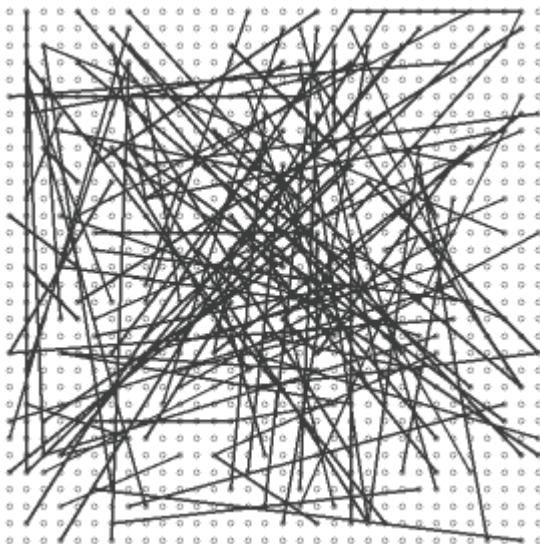
$$\text{LBP} = \text{sign}(I_0 - I_c)2^0 + \text{sign}(I_1 - I_c)2^1 + \text{sign}(I_2 - I_c)2^2 + \text{sign}(I_3 - I_c)2^3 + \text{sign}(I_4 - I_c)2^4 + \text{sign}(I_5 - I_c)2^5 + \text{sign}(I_6 - I_c)2^6 + \text{sign}(I_7 - I_c)2^7$$

Robustness to illumination.

CSLBP [Heikkilä '09]



# Binary Robust Independent Elementary Features



Descriptor construction: intensity test  
between given point pairs:

$$f_n(P) = [\tau(P; x_1, y_1), \dots, \tau(P; x_n, y_n)] \in \{0, 1\}^n$$

$$\tau(P; x, y) = \begin{cases} 1, & P(x) > P(y) \\ 0, & P(x) \leq P(y) \end{cases}$$

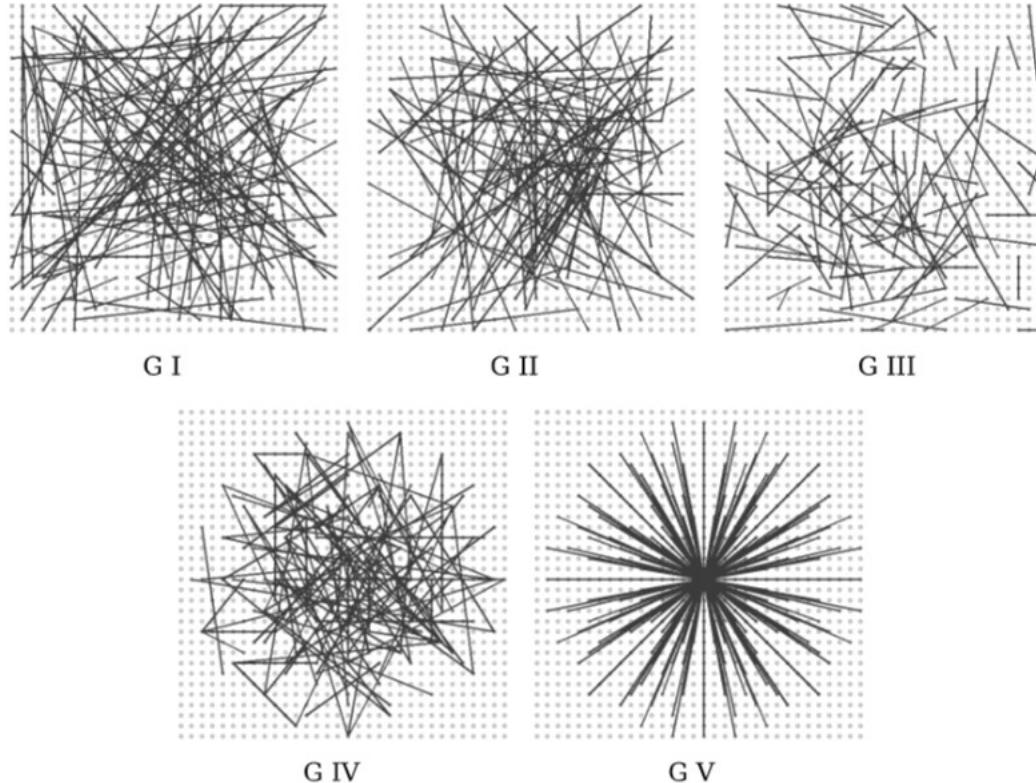
Point pairs: given by random sampling

$$x \square G(0, \frac{S^2}{25}), y \square G(0, \frac{S^2}{25})$$

- Simple, fast, moderate performance
- First binary descriptor for patch matching

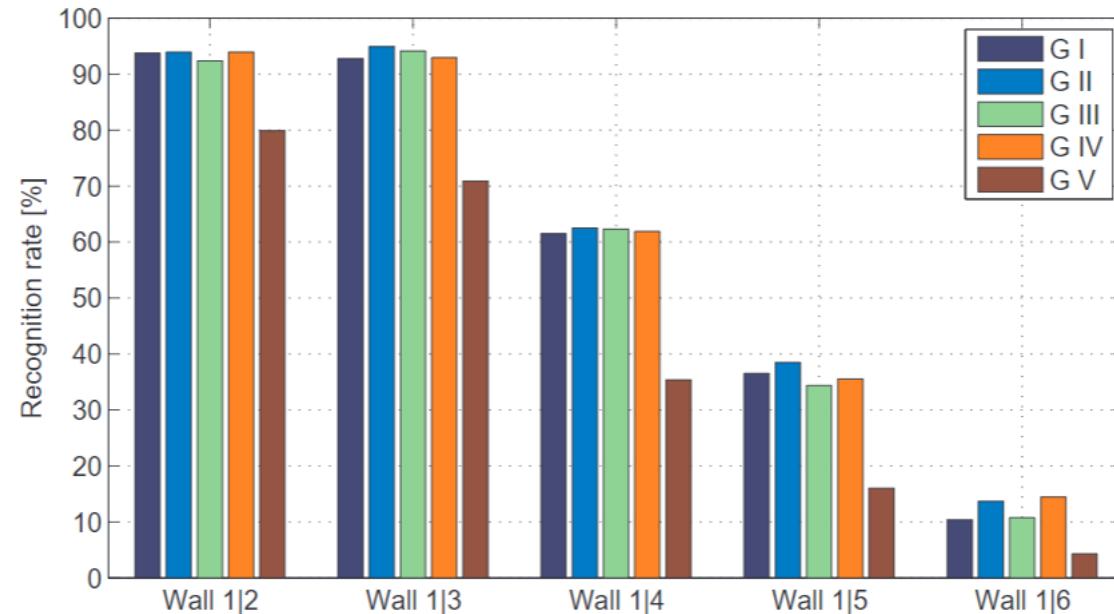


# Binary Robust Independent Elementary Features





# Binary Robust Independent Elementary Features



**Fig. 3.** Recognition rate for the five different test geometries introduced in section 3.2.



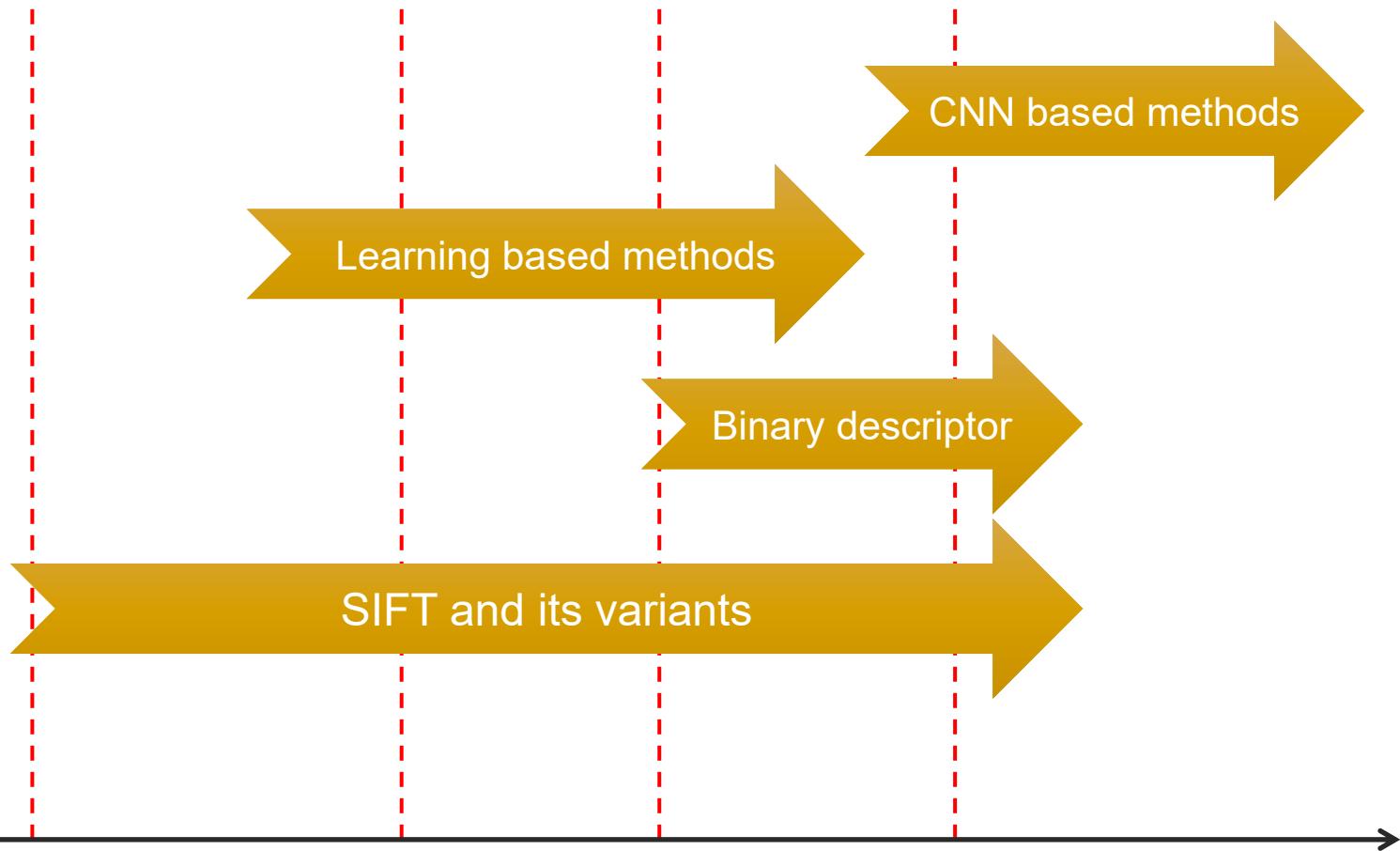
# Today's class



- Normalization
  - Orientation normalization
  - Affine invariant feature extraction
- Local descriptor
  - SIFT, SURF, GIST
- Binary descriptor
  - LBP, BRIEF
- **CNN Based descriptor**
  - MatchNet, DeepCompare, DeepDesc, LIFT



# Local Descriptors: Trend





# A Deep Casualty?



Distinctive image features from scale-invariant keypoints

[PDF] from psu.edu

Authors David G Lowe

Publication date 2004/11/1

Journal International journal of computer vision

Volume 60

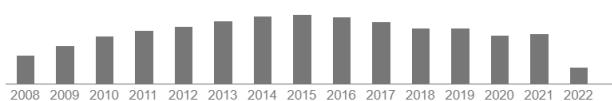
Issue 2

Pages 91-110

Publisher Springer Netherlands

Description This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from many images. This paper also describes an approach to using these features for object recognition. The recognition proceeds by matching individual features to a database of features from known objects using a fast nearest-neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through ...

Total citations Cited by 67096



- The SIFT paper is the most cited computer vision paper **ever**.
- But it's not as dominant as it once was.

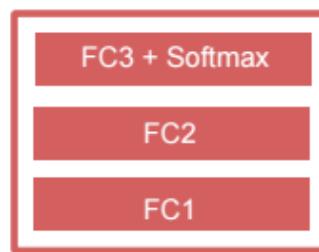


# MatchNet

A: Feature network



B: Metric network



C: MatchNet in training



- Simultaneously learn the descriptor and the metric
- Siamese Feature descriptor network
- Metric network on top
- Cross-entropy loss, transfer matching problem to classification problem
- Train time: 1 day – 1 week



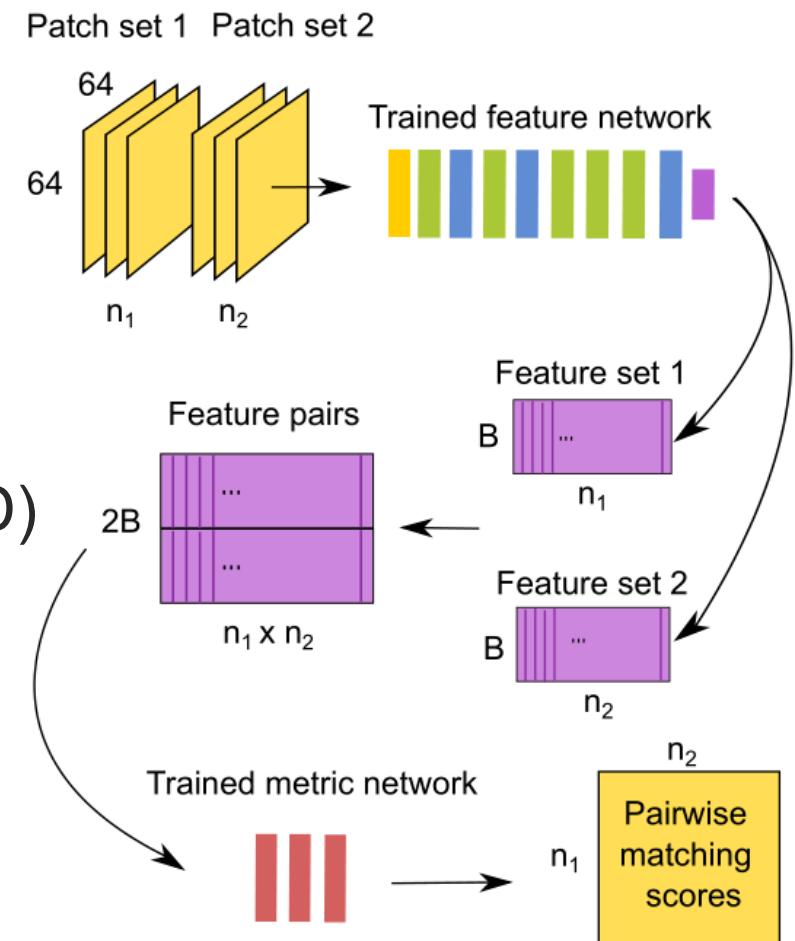
# Training MatchNet

- Cross-entropy error

$$E = -\frac{1}{n} \sum_{i=1}^n [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

- Stochastic gradient descent (SGD)

- A special reservoir sampler for negative sampling



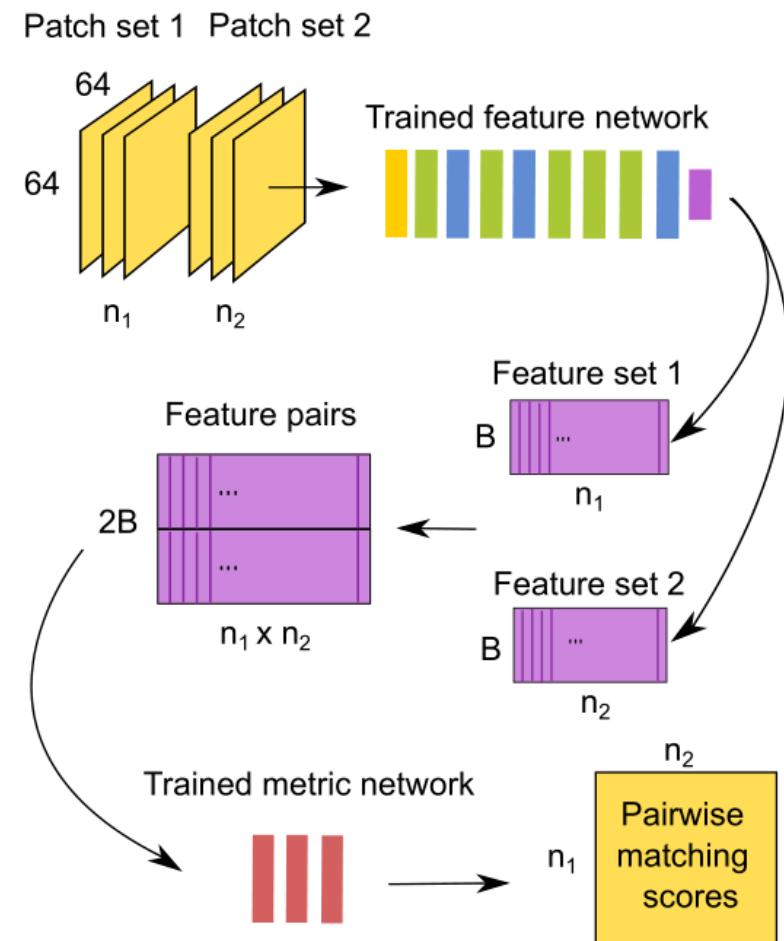


# Testing MatchNet

A two-stage prediction pipeline:

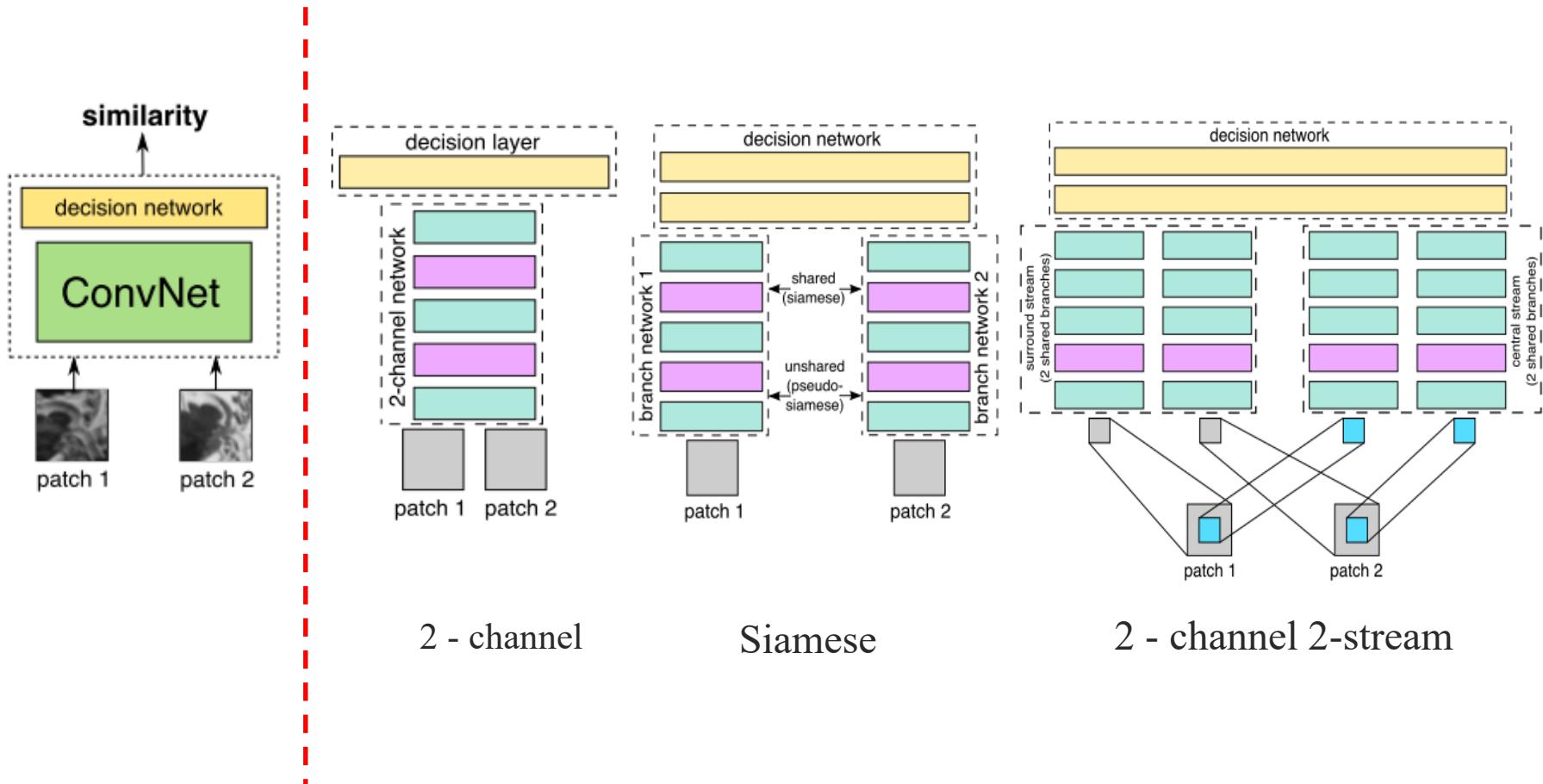
1. Generate feature descriptors for all patches.

2. Pair the features and push them through the metric network to get the scores.





# DeepCompare





# DeepCompare

## ■ Architecture

- **2-channel structure**

- No direct notion of descriptor in the 2-channel architecture. It simply considers the two patches of an input pair as a 2-channel image, which is directly fed to the first convolutional layer of the network.

- **Central-surround two-stream network**

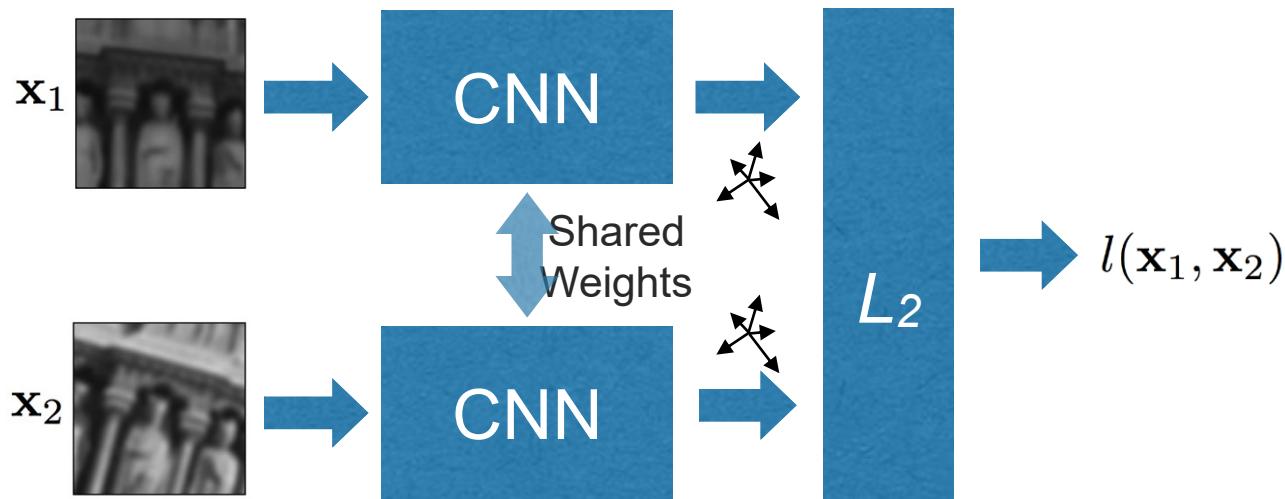
- Consists of two separate streams, central and surround, allowing the network to process at two different resolutions.

## ■ Drawback

- Pair-wise operation, can not re-use descriptor of each patch



# DeepDesc

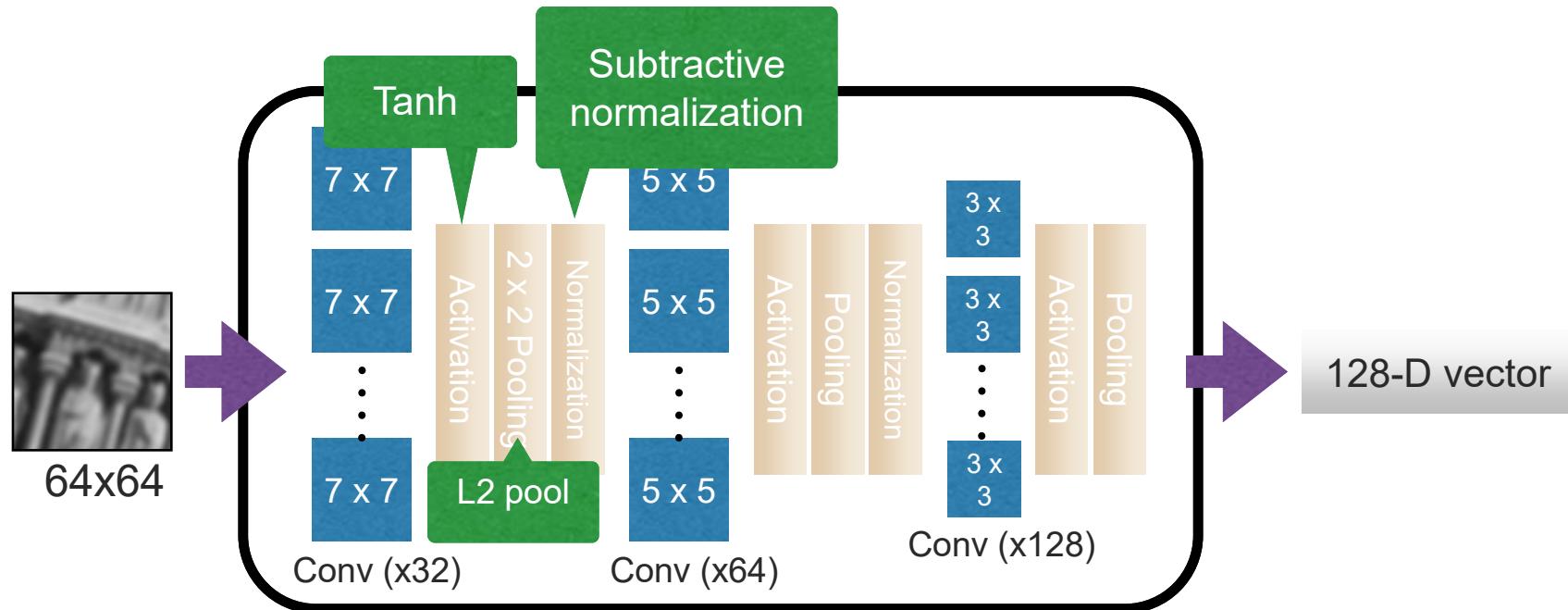


- Use Euclidean distance, direct substitution of SIFT
- loss: minimize pairwise hinge loss

$$l(\mathbf{x}_1, \mathbf{x}_2) = \begin{cases} \|\mathbf{D}(\mathbf{x}_1) - \mathbf{D}(\mathbf{x}_2)\|_2, & p_1 = p_2 \\ \max(0, C - \|\mathbf{D}(\mathbf{x}_1) - \mathbf{D}(\mathbf{x}_2)\|_2), & p_1 \neq p_2 \end{cases}$$



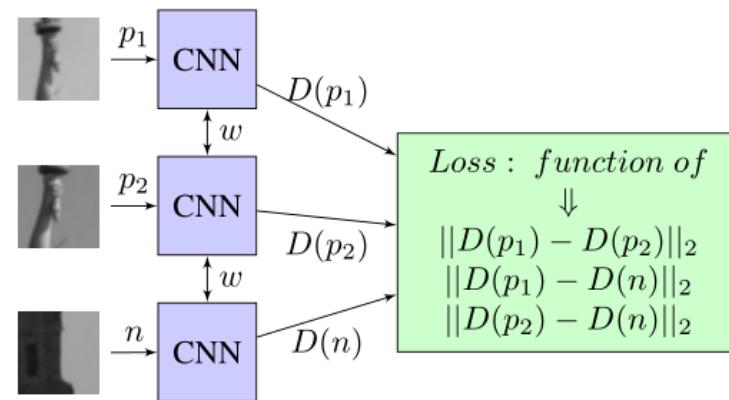
# DeepDesc Network Architecture



- Only 3 convolutional layers, simple.
- Use **hard negative mining** to alleviate the problem of imbalanced positive and negative samples, key to good performance.

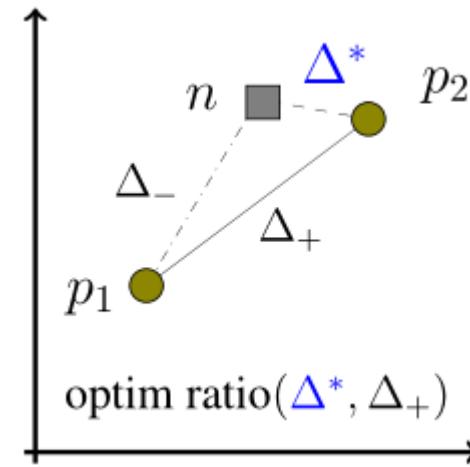


# PN-Net, TFeat



CNN Structure

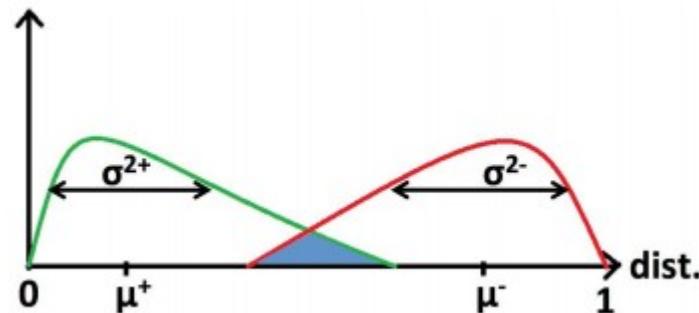
Layer #	Description
1	Spatial Convolution(7,7) $\rightarrow$ 32
2	Tanh
2	MaxPooling(2,2)
3	Spatial Convolution(6,6) $\rightarrow$ 64
4	Tanh
5	Linear $\rightarrow \{128, 256\}$
6	Tanh



**Triplet Network:**  
Smallest negative  
distance within the  
triplet should be larger  
than the positive  
distance.



# GLoss Net



**Objective:** Reduce the proportion of false positive and false negative, i.e., blue shaded area. (A global loss)

- **Global Loss**

- **Minimize** the variance of the two distributions and the mean value of the distances between matching pairs.
- **Maximize** the mean value of the distances between non-matching pairs.

- **Four models**

- Metric learning: SNet-GLoss, CS SNet-GLoss (with Siamese Network)
- L2 norm: TNet-TGLoss, TNet-TLoss (with Triplet Network)



# Performance Comparison of these CNN based Methods



Training Test	Metric Learning	Feature Dim	Notredame		Liberty		Yosemite	
			Liberty	Yosemite	Notredame	Liberty	Yosemite	
Float Descriptors								
SIFT		128		29.84		22.53		27.29
MatchNet	Yes	4096	6.9	10.77	3.87	5.67	10.88	8.39
DeepCompare 2ch-2stream	Yes	256	4.85	7.20	1.90	2.11	5.00	4.10
DeepCompare 2ch-deep	Yes	256	4.55	7.40	2.01	2.52	4.75	4.38
SNet-GLoss	Yes	384	6.39	8.43	1.84	2.83	6.61	5.57
CS SNet-GLoss	Yes	384	<b>3.69</b>	<b>4.91</b>	<b>0.77</b>	<b>1.14</b>	<b>3.09</b>	<b>2.67</b>
TNet-TGLoss	No	256	9.91	13.45	3.91	5.43	10.65	9.47
TNet-TLoss	No	256	10.77	13.90	4.47	5.58	11.82	10.96
PN-Net	No	256	8.13	9.65	3.71	4.23	8.99	7.21
DeepDesc	No	128		10.9		4.40		5.69

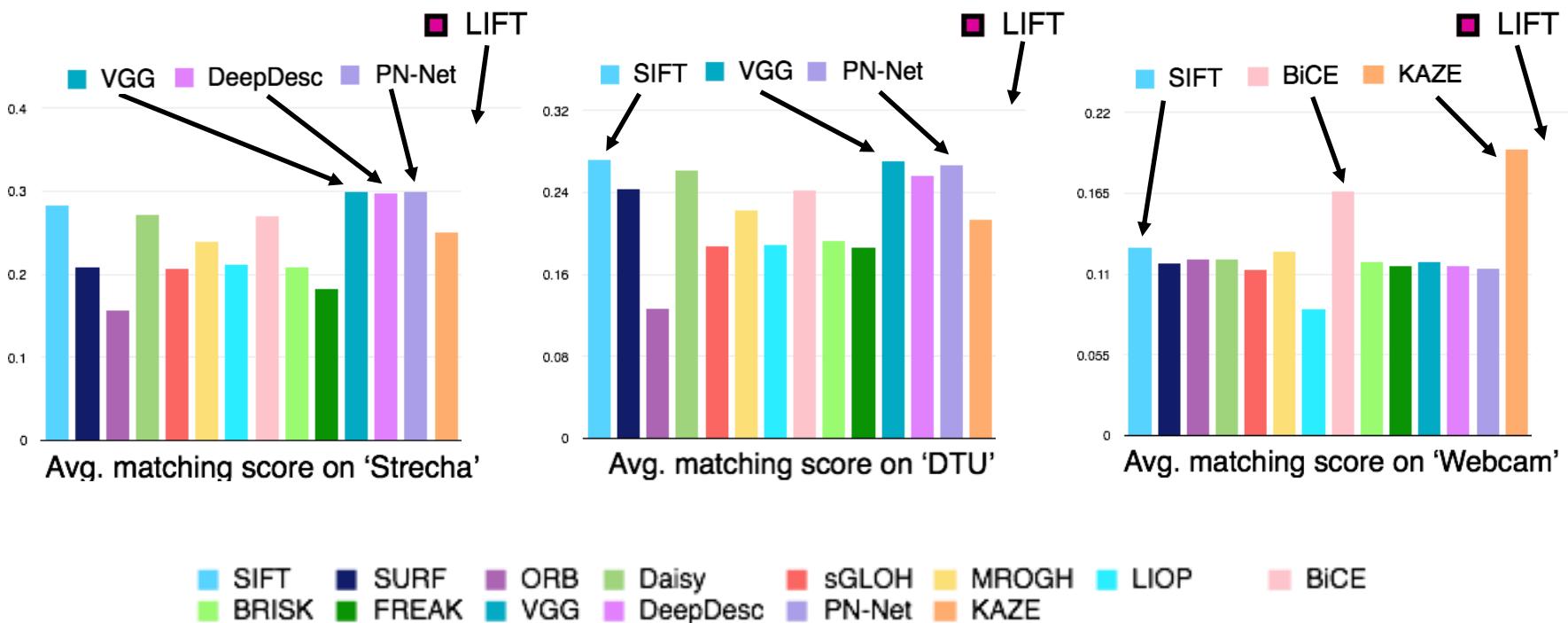
- Note: these results are on the Brown dataset.



# Moreover...



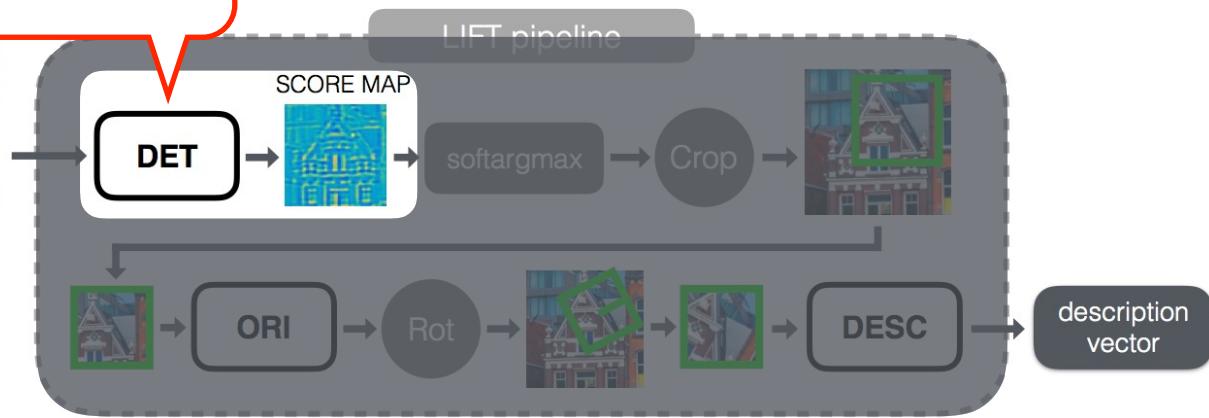
- Descriptors are affected by **keypoints & orientations**





# The LIFT Network

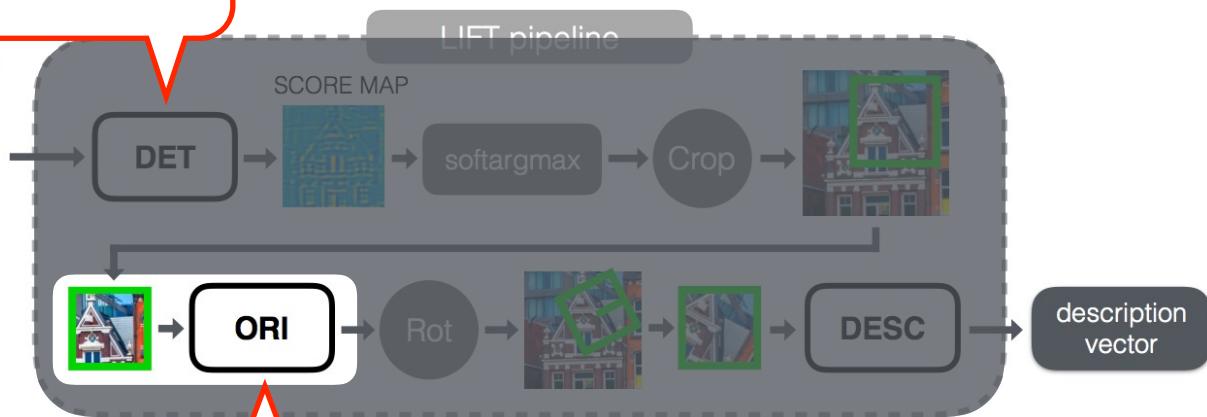
Y. Verdie, K.M. Yi, P. Fua, V. Lepetit:  
"TILDE: A Temporally Invariant  
Learned DEtector", CVPR 2015.





# The LIFT Network

Y. Verdie, K.M. Yi, P. Fua, V. Lepetit:  
"TILDE: A Temporally Invariant  
Learned DEtector", CVPR 2015.

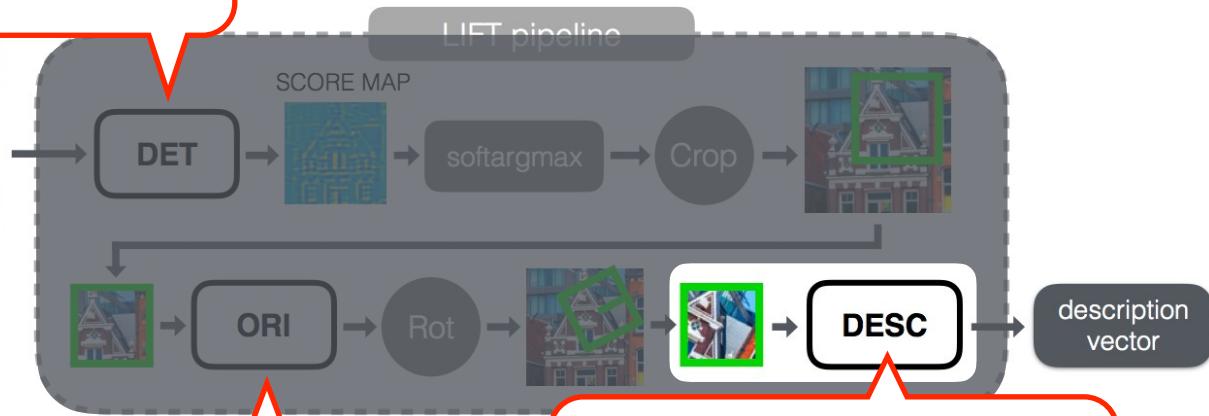


K.M. Yi, Y. Verdie, V. Lepetit, P. Fua :  
"Learning to Assign Orientations to  
Feature Points", CVPR 2016.



# The LIFT Network

Y. Verdie, K.M. Yi, P. Fua, V. Lepetit:  
"TILDE: A Temporally Invariant  
Learned DEtector", CVPR 2015.

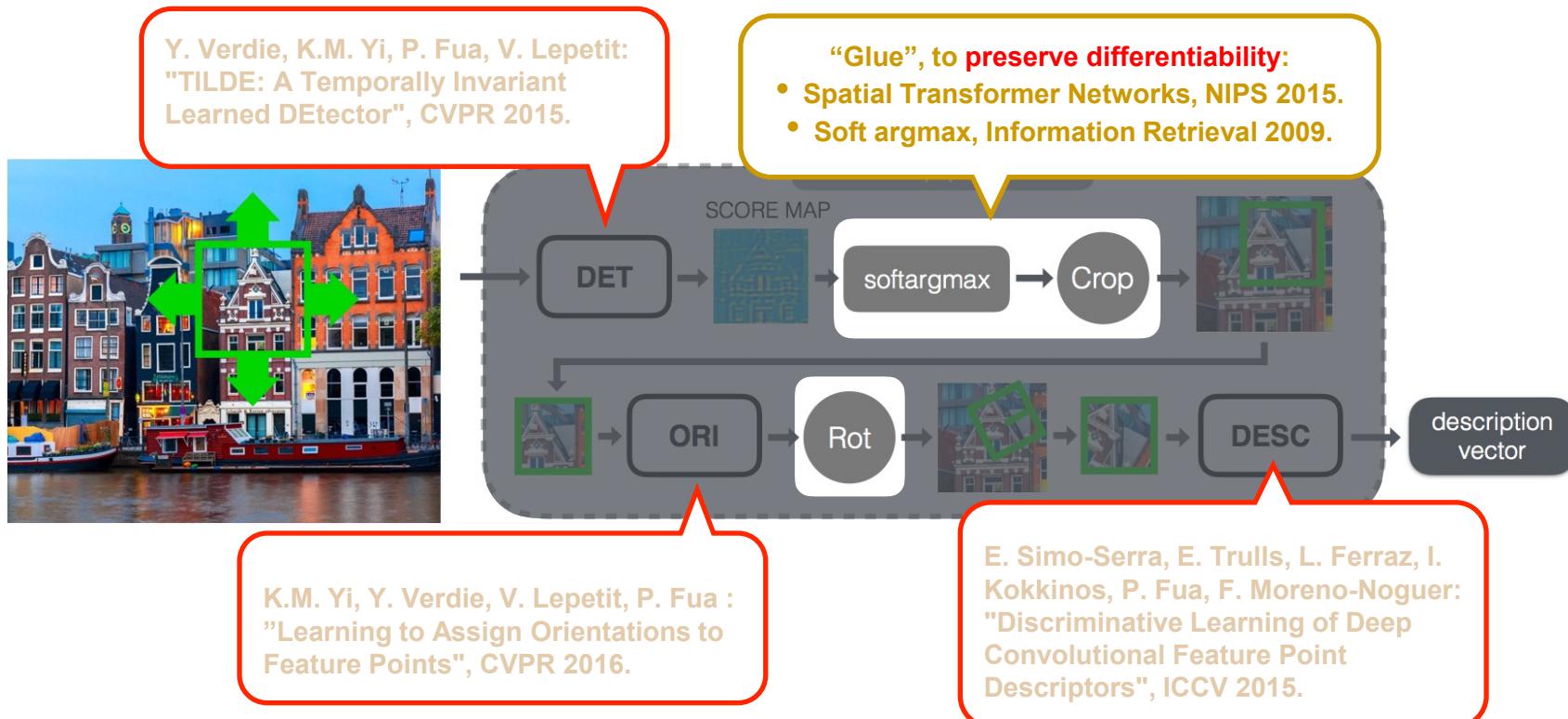


K.M. Yi, Y. Verdie, V. Lepetit, P. Fua :  
"Learning to Assign Orientations to  
Feature Points", CVPR 2016.

E. Simo-Serra, E. Trulls, L. Ferraz, I.  
Kokkinos, P. Fua, F. Moreno-Noguer:  
"Discriminative Learning of Deep  
Convolutional Feature Point  
Descriptors", ICCV 2015.

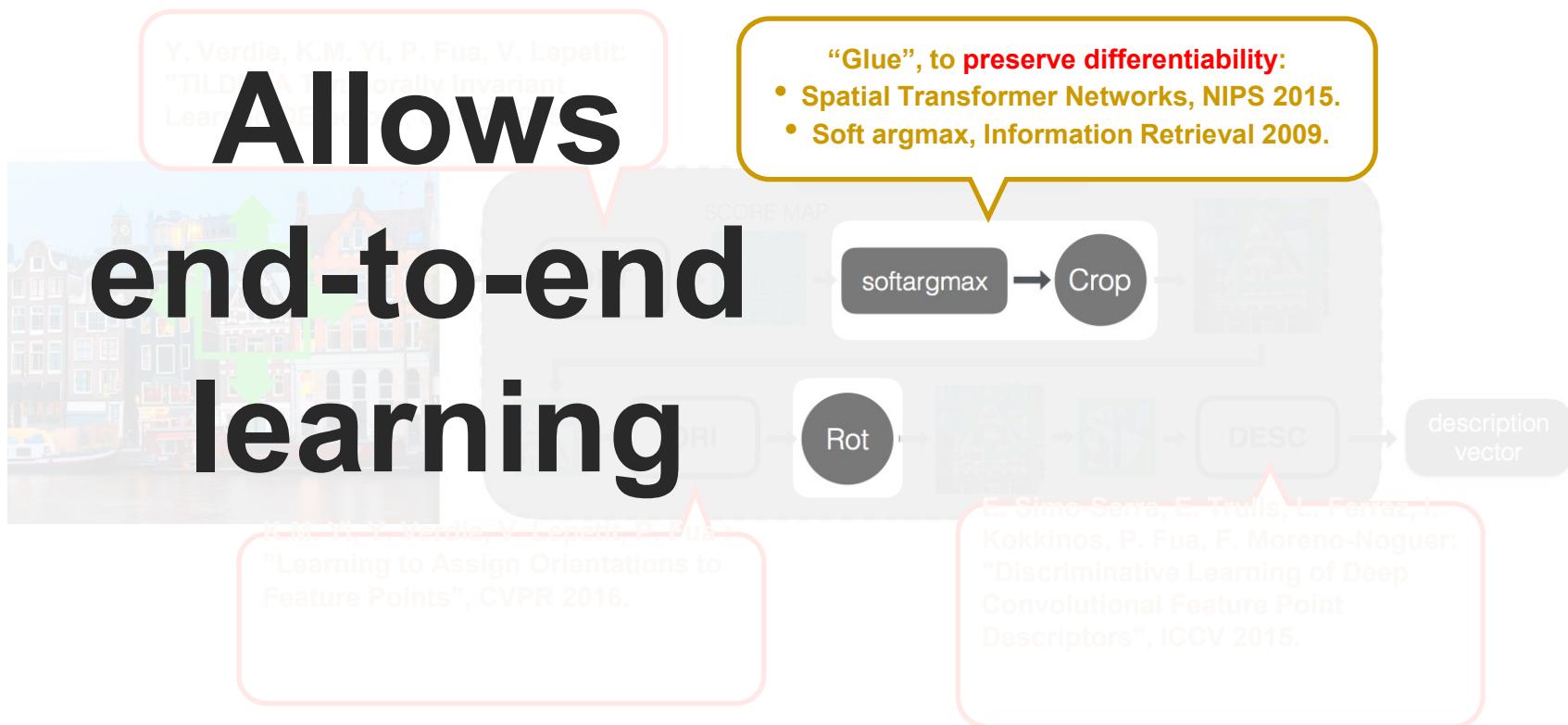


# The LIFT Network



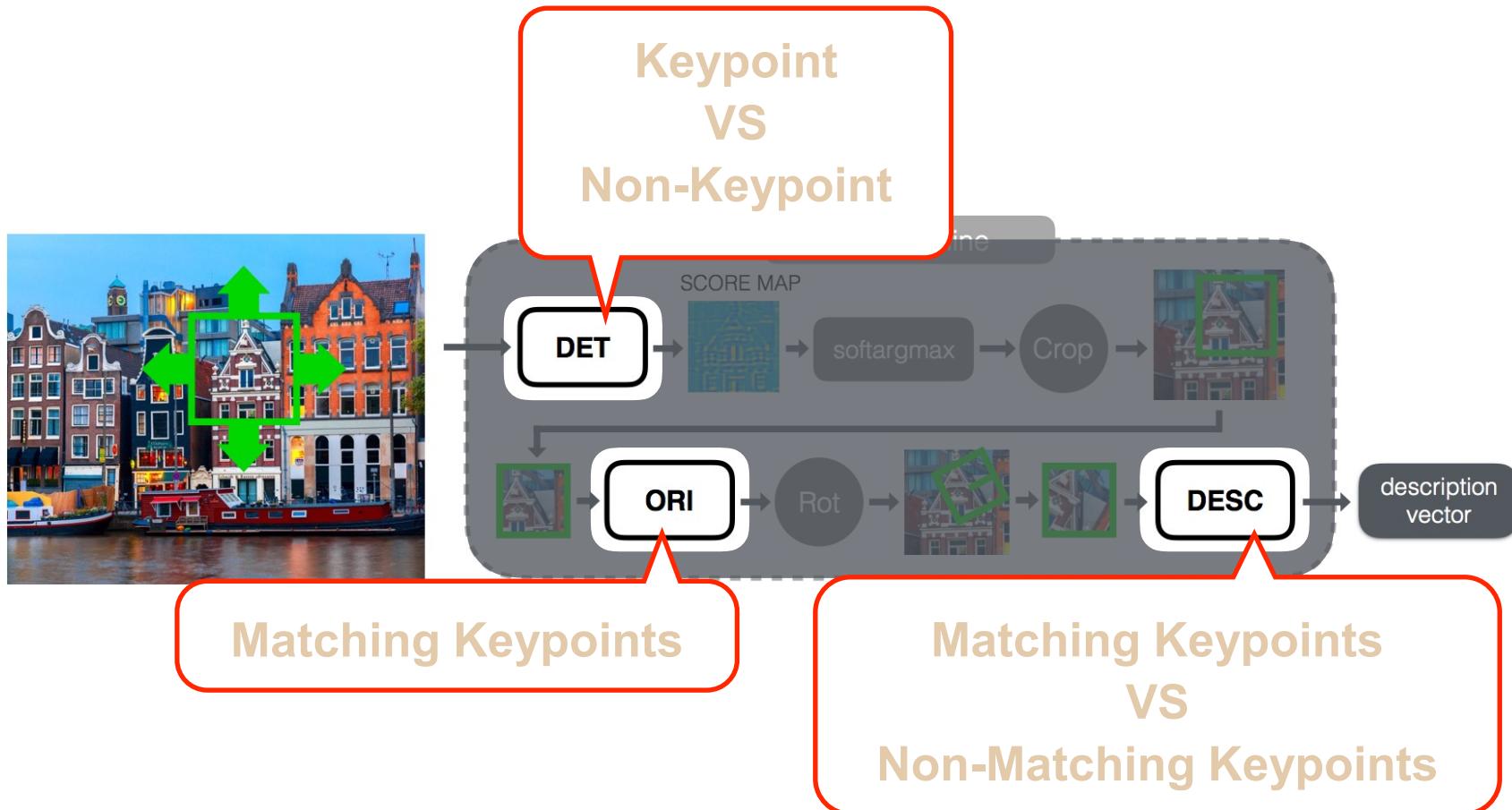


# The LIFT Network



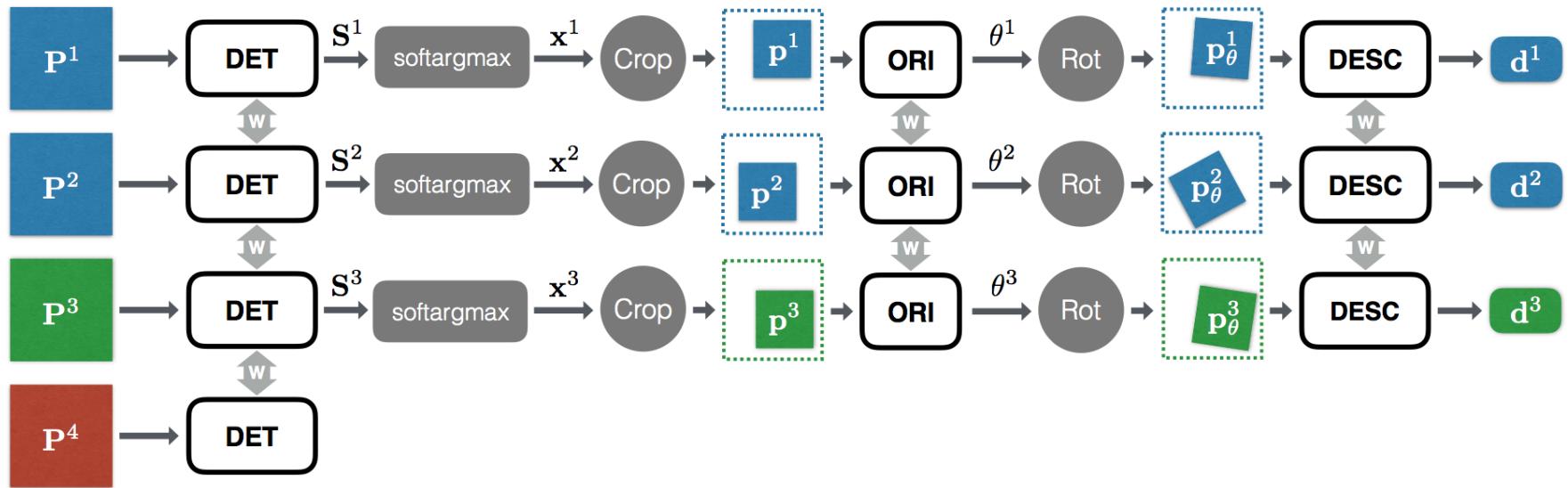


# Training requires various patches





# Quadruplet Siamese Network



$P_1, P_2$ : corresponding keypoints.

$P_3$ : non-corresponding keypoint.

$P_4$ : non-keypoint.



# A single, global cost function



$$\min_{\{f_\mu, g_\phi, h_\rho\}} \sum_{\{(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4)\}} \gamma \mathcal{L}_{class}(\mathbf{P}^1, \mathbf{P}^2, \mathbf{P}^3, \mathbf{P}^4) + \mathcal{L}_{pair}(\mathbf{P}^1, \mathbf{P}^2)$$

detector  
orientation  
descriptor

$$\mathcal{L}_{class}(\mathbf{P}^1, \mathbf{P}^2, \mathbf{P}^3, \mathbf{P}^4) = \sum_{i=1}^4 \alpha_i \max \left( 0, \left( 1 - \text{softmax} \left( f_\mu (\mathbf{P}^i) \right) y_i \right)^2 \right)$$

detector

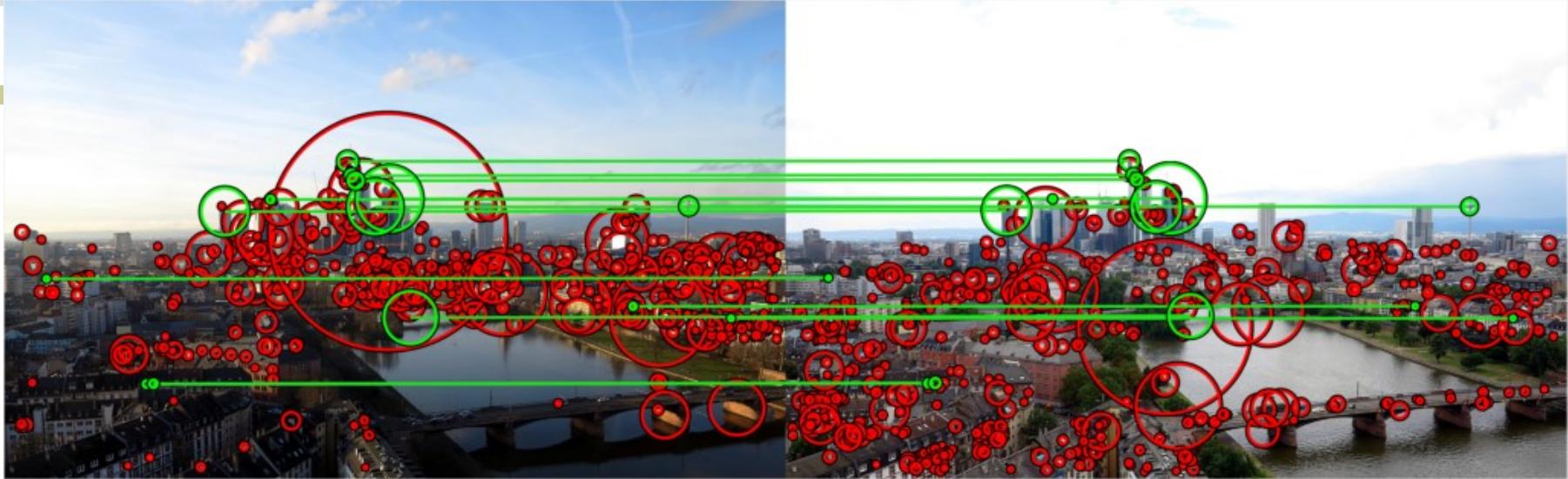
$$\mathcal{L}_{pair}(\mathbf{P}^1, \mathbf{P}^2) = \| h_\rho(G(\mathbf{P}^1, \text{softargmax}(f_\mu(\mathbf{P}^1)))) - h_\rho(G(\mathbf{P}^2, \text{softargmax}(f_\mu(\mathbf{P}^2)))) \|_2$$

descriptor

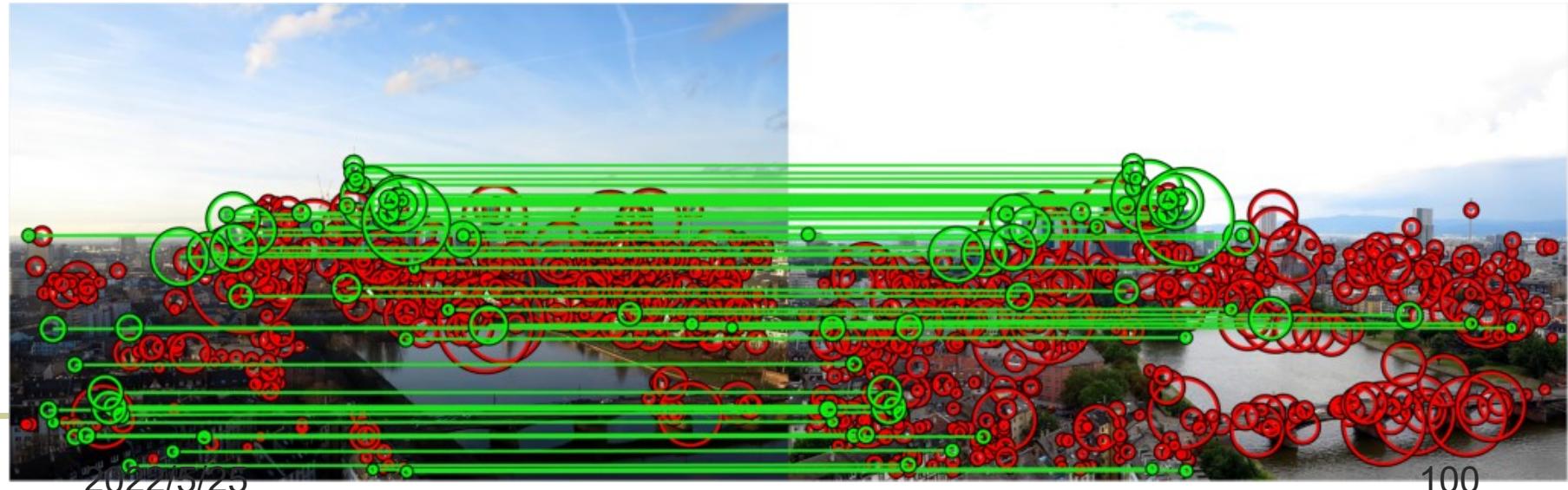
$$G(\mathbf{P}, \mathbf{x}) = \text{Rot}(\mathbf{P}, \mathbf{x}, g_\phi(\text{Crop}(\mathbf{P}, \mathbf{x})))$$

orientation

Matching features on '**Webcam**', sequence '**Frankfurt**'.  
Correct matches shown with **green** lines.



**SIFT.** Average: **23.1** matches

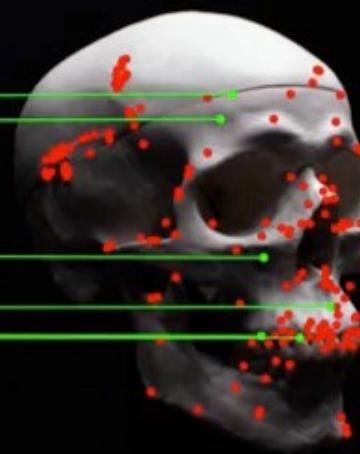


**LIFT (Ours).** Average: **60.6** matches

Matching features on ‘DTU’, sequence #19.  
Correct matches shown with **green** lines.



Matches: 6 / 500



**SIFT.** Average: **34.1** matches



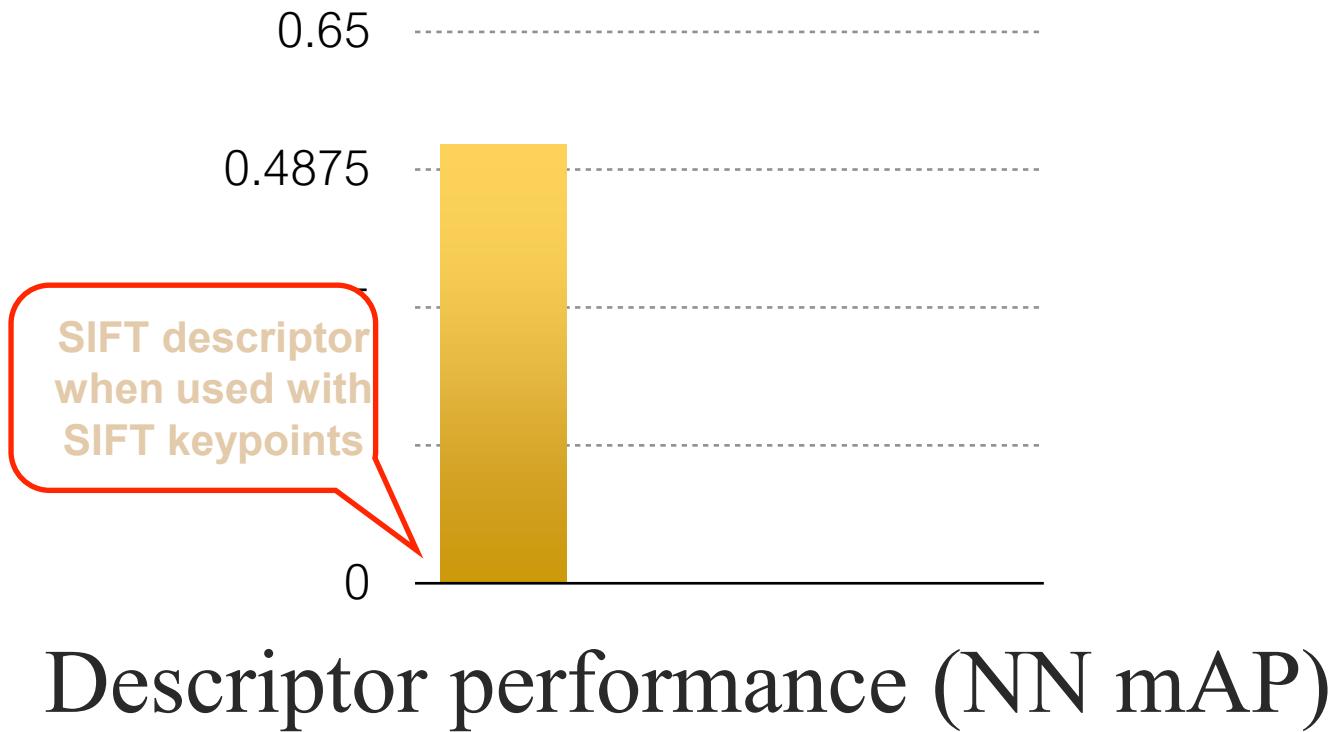
Matches: 42 / 500



**LIFT (Ours).** Average: **98.5** matches

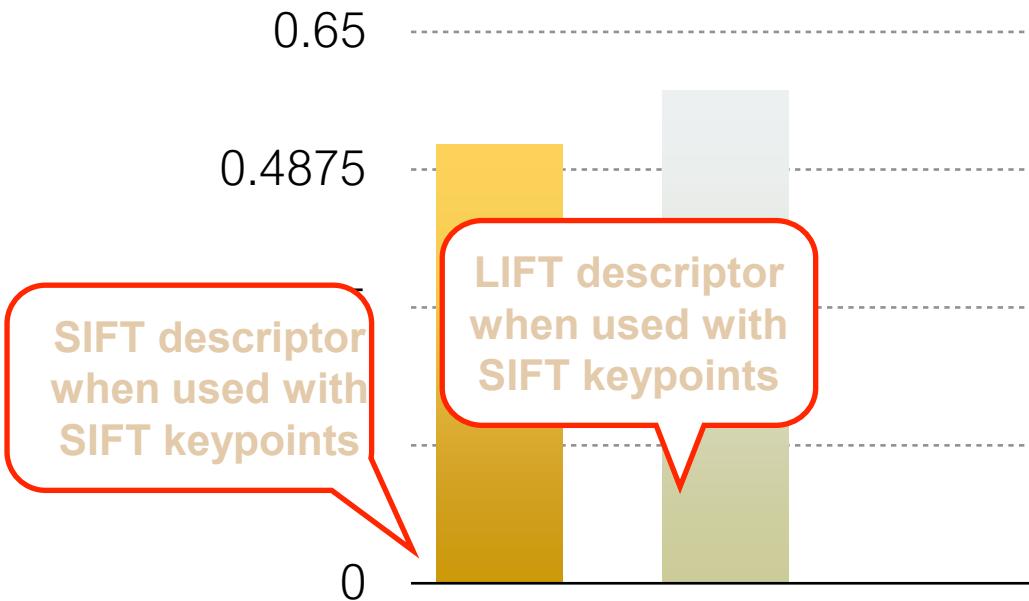


# Each component is *meant for* each other





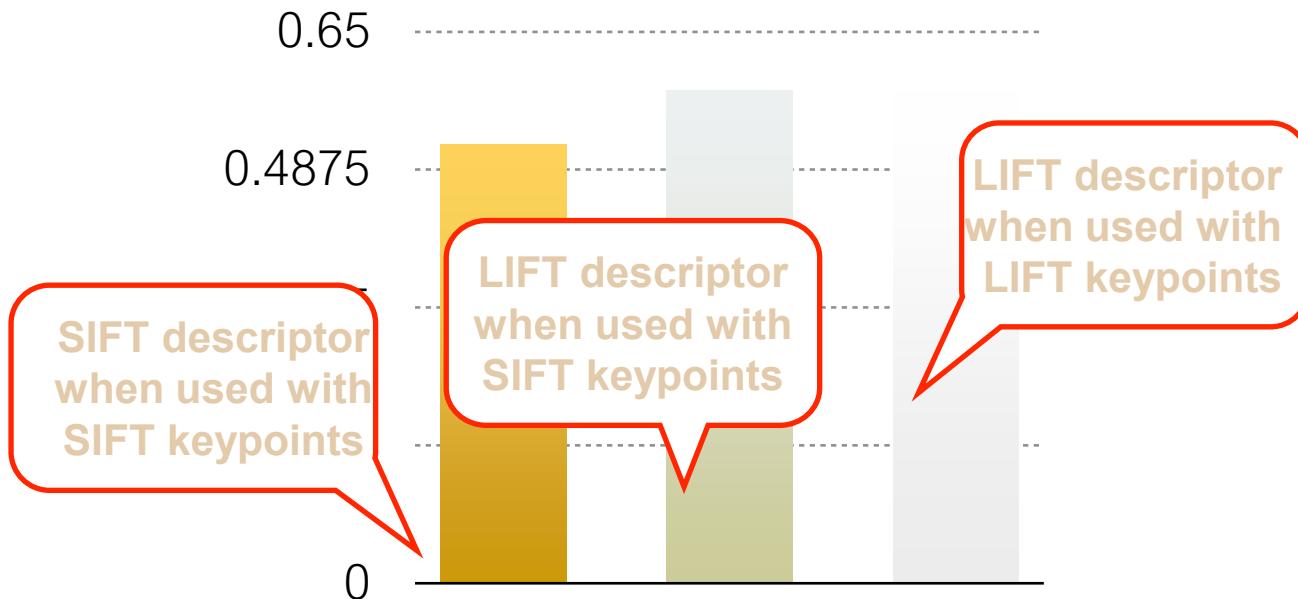
# Each component is *meant for* each other



## Descriptor performance (NN mAP)



# Each component is *meant for* each other



## Descriptor performance (NN mAP)



# Floating Point Descriptors: A Summarization

- For patch level datasets, learning based methods generally outperform hand-crafted ones.
- For image level dataset, performance gap between learning based methods and hand-crafted methods is not significant, except for LIFT.
- For domain adaptation (e.g. visible to IR), hand-crafted descriptors are more adaptable.
- CNN based methods are dominant in the learning based methods.
- CNN based methods operating on the Euclidean space is highly required for wider application.



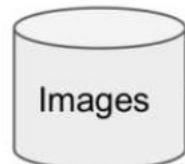
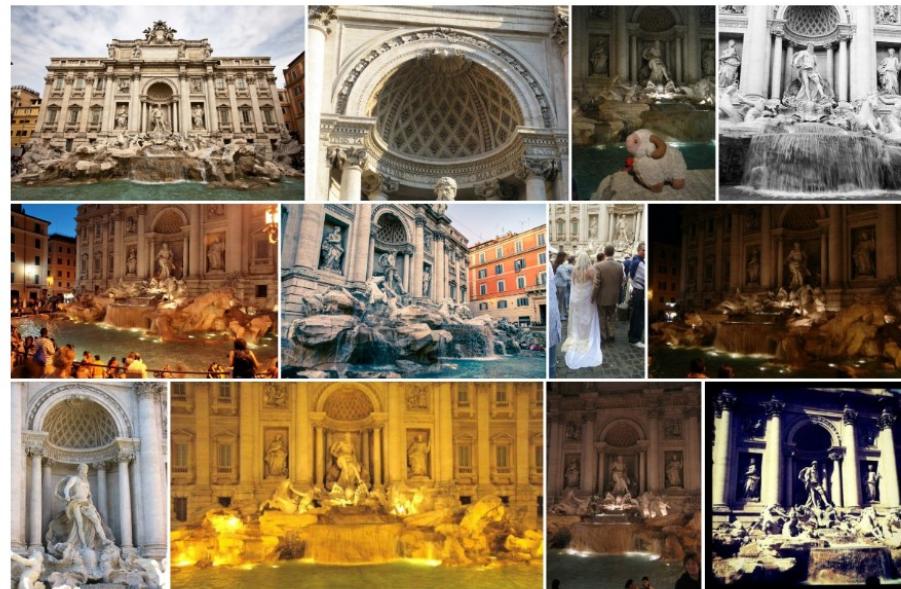
# Software



- OpenCV: <http://opencv.org/>
  - SIFT, SURF, BRISK, BRIEF, ORB, FREAK
- VLFeat: <http://www.vlfeat.org/>
  - SIFT, LIOP, Covariant Feature Detectors
- Authors' pages, Github, etc.
- Supplementary containing implementation information
  - Learned Orientations:  
<https://infoscience.epfl.ch/record/217982/files/0141-supp.pdf>
  - LIFT: [https://documents.epfl.ch/groups/c/cv/cvlab-unit/www/data/keypoints/lift/paper\\_1377\\_supplementary.pdf](https://documents.epfl.ch/groups/c/cv/cvlab-unit/www/data/keypoints/lift/paper_1377_supplementary.pdf)



# Image Matching Benchmark and Challenge



Feature Extraction

Feature Matching

Outlier Pre-filtering

Task 1: Stereo

RANSAC

Task 2: Multi-view

SfM (Colmap)

Pose Error Computation



# Image Matching Benchmark and Challenge

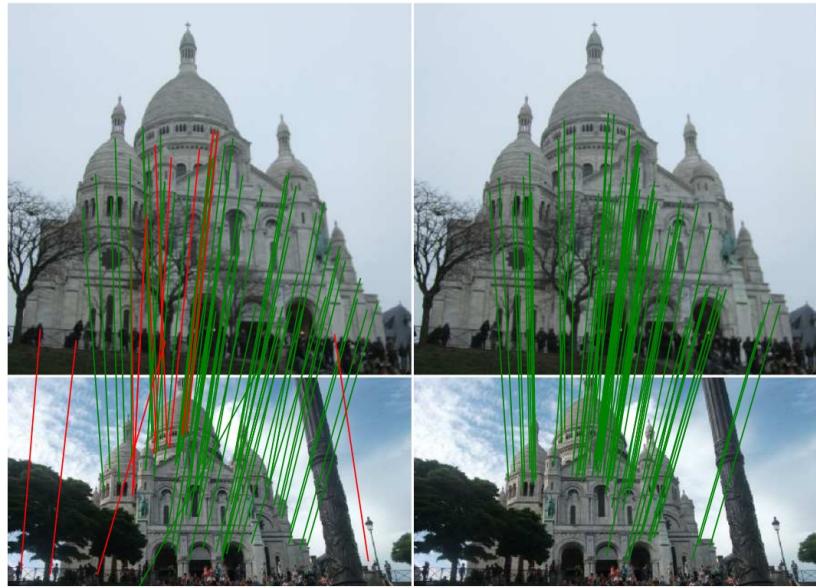


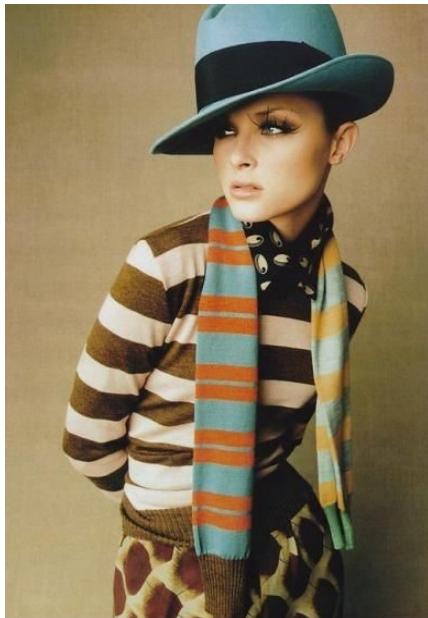
Figure 1. Every paper claims to outperform the state of the art. Is this possible, or an artifact of insufficient validation? On the left, we show stereo matches obtained with **D2-Net** (2019) [33], a state-of-the-art local feature, using OpenCV RANSAC with its default settings. On the right, we show **SIFT** (1999) [48] with a carefully tuned MAGSAC [29] – notice how the latter performs much better. We fill this gap with a new, modular benchmark for sparse image matching, with dozens of built-in methods.

Method	PyRANSAC			DEGENSAC		MAGSAC			Rank
	NF	NI $\uparrow$	mAP(10°) $\uparrow$	NI $\uparrow$	mAP(10°) $\uparrow$	NI $\uparrow$	mAP(10°) $\uparrow$	Rank	
CV-SIFT	7879.0	153.9	.4160	222.7	.4608	270.6	.4586	13	
VL-SIFT	7901.0	166.2	.4137	241.2	.4643	301.0	.4638	12	
VL-Hessian-SIFT	8000.0	186.5	.3915	264.0	.4489	318.0	.4394	15	
VL-DoGAff-SIFT	7910.9	218.6	.4049	229.7	.4653	291.9	.4624	11	
VL-HesAffNet-SIFT	8000.0	190.8	.4081	271.6	.4659	328.5	.4581	10	
CV- $\sqrt{\text{SIFT}}$	7884.0	176.3	.4348	257.4	.4921	317.4	.4891	6	
SURF	7749.0	113.0	.2326	117.8	.2452	136.1	.2481	19	
AKAZE	7879.8	184.3	.2738	232.7	.3142	284.4	.3054	17	
ORB	7128.2	113.1	.1381	136.6	.1723	163.2	.1632	22	
DoG-HardNet	7884.1	229.9	.4668	342.2	<b>.5286</b>	404.0	.5147	1	
DoG-HardNetAmos+	7884.1	213.6	.4511	316.9	.5125	373.5	.5011	3	
L2Net	7884.8	190.2	.4478	280.9	.4971	329.1	.4884	5	
Key.Net-HardNet	7998.1	353.3	.3990	375.2	.4700	636.5	.4529	9	
Geodesc	7884.3	179.7	.4183	264.0	.4787	340.3	.4753	8	
ContextDesc	4811.1	248.8	.4283	261.4	.4856	356.1	.4662	7	
SOSNet	7884.3	215.1	.4595	319.8	<b>.5233</b>	418.5	<b>.5177</b>	2	
LogPolarDesc	7884.3	243.5	.4495	366.0	.5080	461.0	.5001	4	
SuperPoint (2k)	1178.9	88.1	.2359	84.7	.2669	113.2	.2620	18	
LF-Net (2k)	2024.8	95.1	.1945	100.8	.2253	134.2	.2164	20	
D2-Net (SS)	5540.7	273.5	.1432	241.4	.1639	428.0	.1560	23	
D2-Net (MS)	6806.3	193.8	.1690	322.8	.1836	505.1	.1731	21	

Table 1. **Stereo – Test set.** We report: (**NF**) Number of Features; (**NI**) Number of Inliers produced by RANSAC; and **mAP(10°)**. Top three methods by mAP marked in red, green and blue.



# Discriminative power



Raw pixels



Sampled



Locally orderless



Global histogram

# Generalization power





# Summary: 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

Mikolajczyk, K. et al. A comparison of affine region detectors. IJCV, 2005

Mikolajczyk, K. et al. A performance evaluation of local descriptors. T-PAMI, 2005.