CS598:Visual information Retrieval

Lecture III: Image Representation:

Invariant Local Image Descriptors

Recap of Lecture II

- Color, texture, descriptors
  - Color histogram
  - Color correlogram
  - LBP descriptors
  - Histogram of oriented gradient
  - Spatial pyramid matching
- Distance & Similarity measure
  - L_p distances
  - Chi-Square distances
  - KL distances
  - EMD distances
  - Histogram intersection

Lecture III: Part I

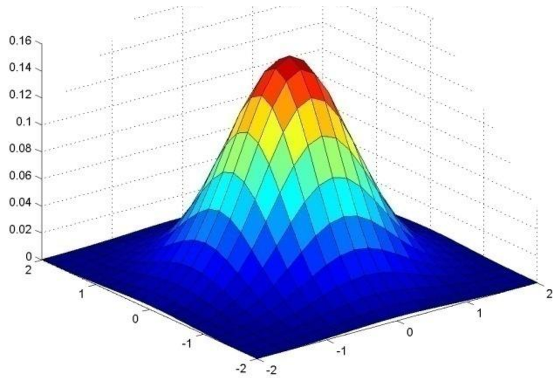
Local Feature Detector

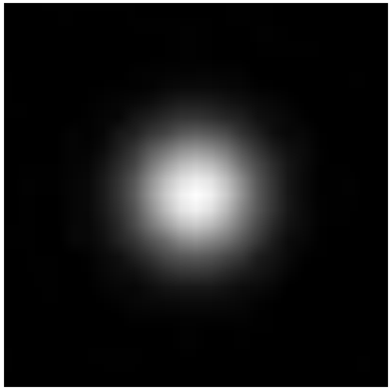
OUTLINE

- Blob detection
  - Brief of Gaussian filter
  - Scale selection
  - Lapacian of Gaussian (LoG) detector
  - Difference of Gaussian (DoG) detector
  - Affine co-variant region

Gaussian Kernel

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





0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

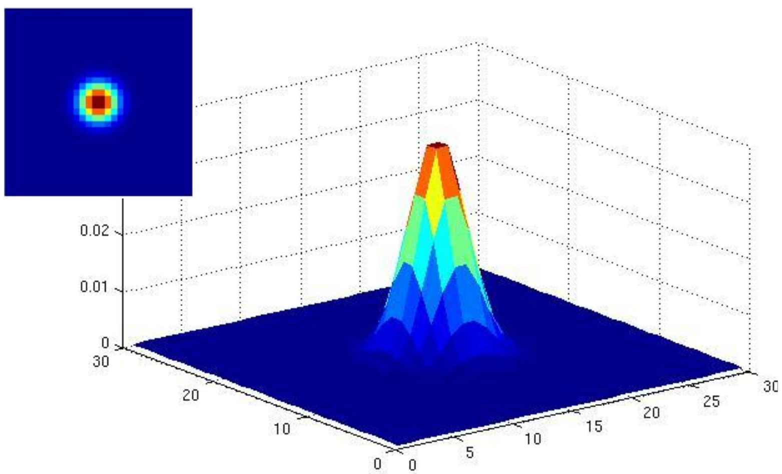
5 x 5, σ = 1

- Constant factor at front makes volume sum to 1 (can be ignored when computing the filter values, as we should renormalize weights to sum to 1 in any case)

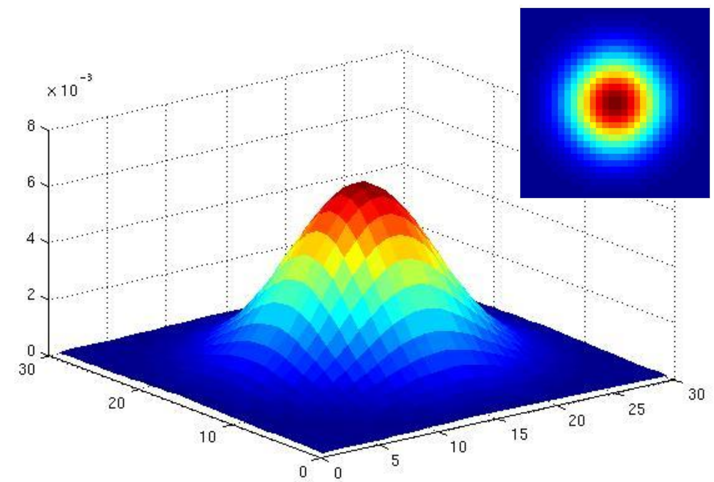
Source: C. Rasmussen

Gaussian Kernel

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



σ = 2 with 30 x 30 kernel



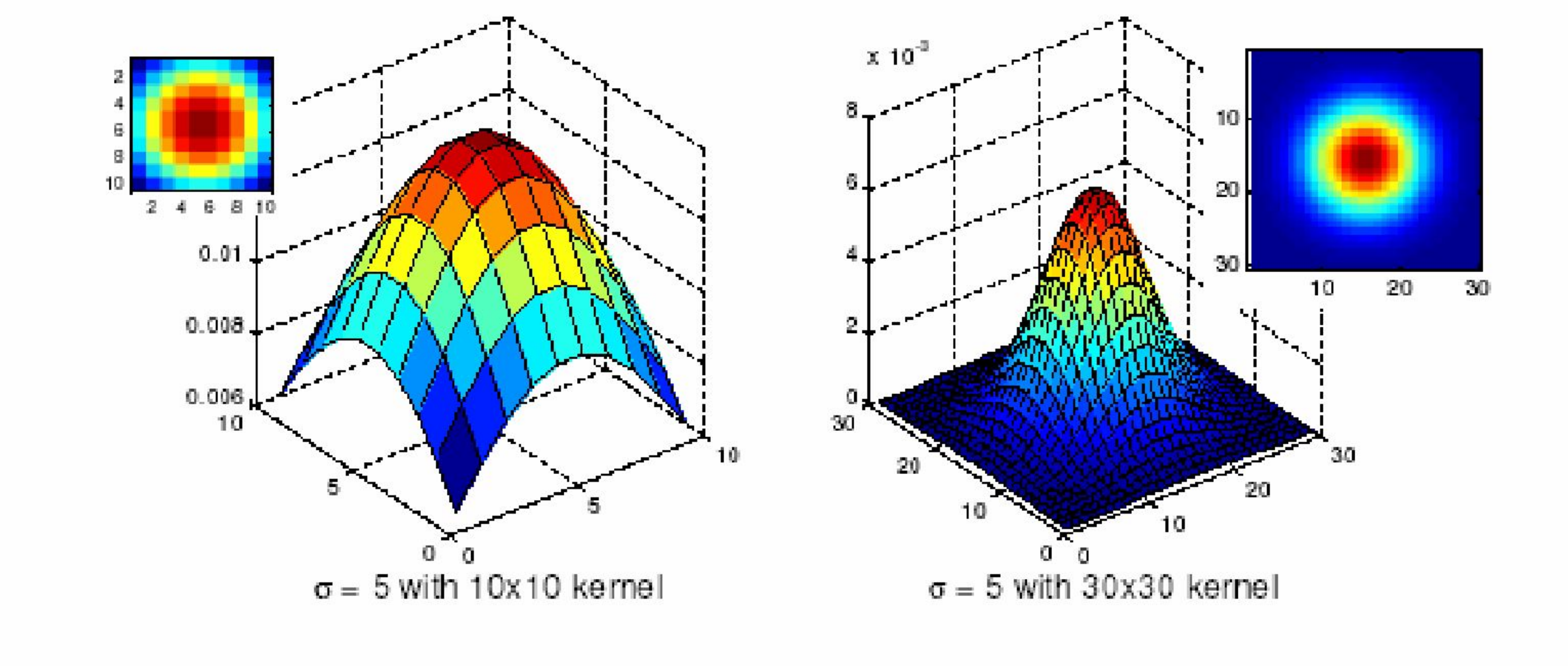
σ = 5 with 30 x 30 kernel

- Standard deviation σ: determines extent of smoothing

Source: K. Grauman

Choosing kernel width

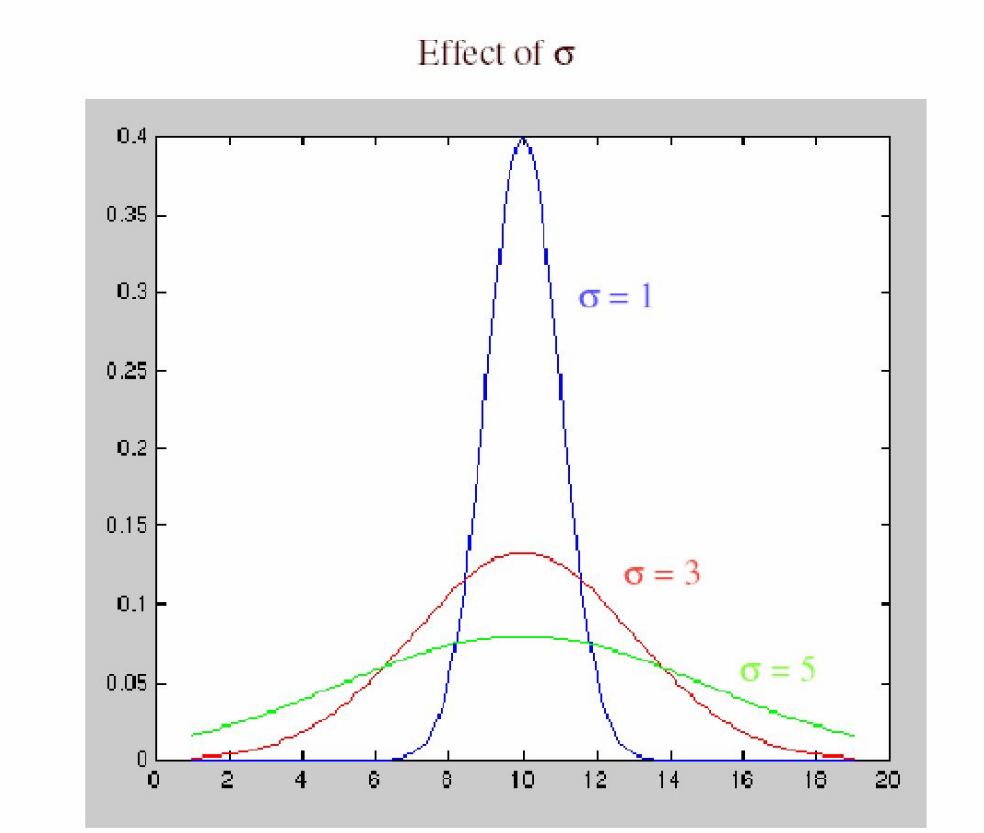
- The Gaussian function has infinite support, but discrete filters use finite kernels



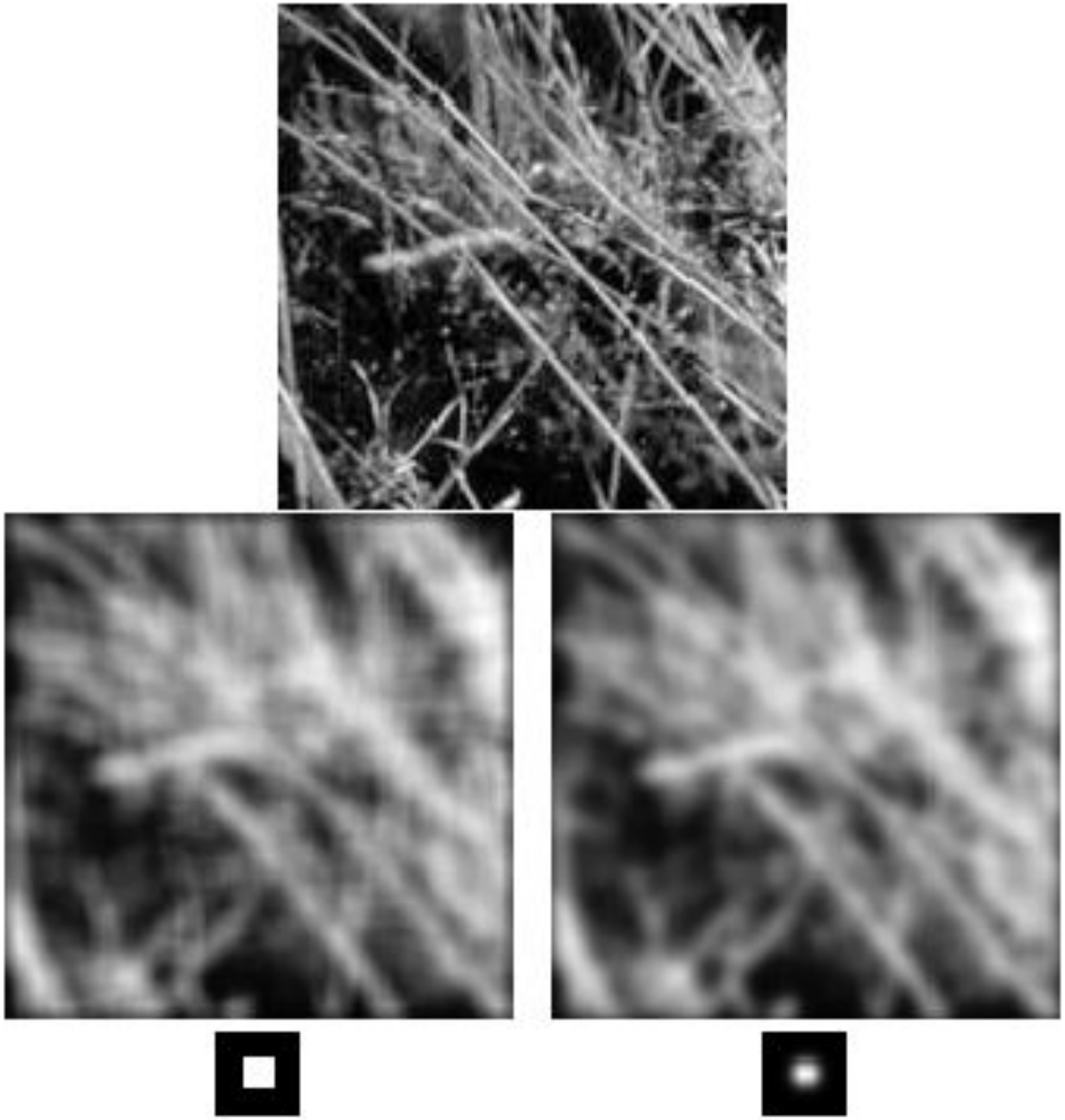
Source: K. G

Choosing kernel width

- Rule of thumb: set filter half-width to about 3σ



Gaussian vs. box filtering



Gaussian filters

- Remove "high-frequency" components from the image (low-pass filter)
- Convolution with self is another Gaussian
  - So can smooth with small-σ kernel, repeat, and get same result as larger-σ kernel would have
  - Convolving two times with Gaussian kernel with std. dev. σ
    is same as convolving once with kernel with std. dev. $\sigma\sqrt{2}$
- *Separable* kernel
  - Factors into product of two 1D Gaussians

Source: K.

Separability of the Gaussian filter

$$
\begin{aligned}
G_\sigma(x,y) &= \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)
\end{aligned}
$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

Source: D. L

Separability example

2D convolution (center location only)

$$\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix} * \begin{bmatrix}2&3&3\\3&5&5\\4&4&6\end{bmatrix}$$

The filter factors into a product of 1D filters:

$$\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix} = \begin{bmatrix}1\\2\\1\end{bmatrix} \times \begin{bmatrix}1&2&1\end{bmatrix}$$

Perform convolution along rows:

$$\begin{bmatrix}1&2&1\end{bmatrix} * \begin{bmatrix}2&3&3\\3&5&5\\4&4&6\end{bmatrix} = \begin{bmatrix}&11&\\&18&\\&18&\end{bmatrix}$$

Followed by convolution along the remaining column:

Source: K.

Why is separability useful?

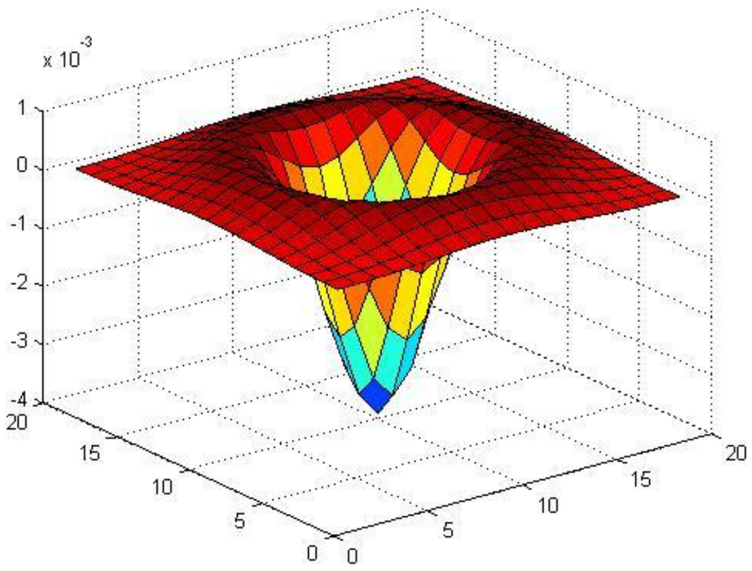
- What is the complexity of filtering an $n \times n$ image with an $m \times m$ kernel?
  - $O(n^2 m^2)$
- What if the kernel is separable?
  - $O(n^2 m)$

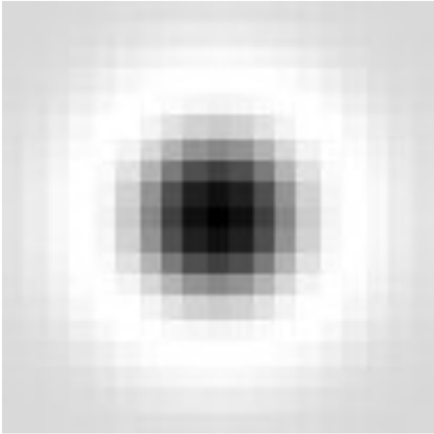
Outline

- Blob detection
  - Brief of Gaussian filter
  - <u>Scale selection</u>
  - Lapacian of Gaussian (LoG) detector
  - Difference of Gaussian (DoG) detector
  - Affine co-variant region

Blob detection in 2D

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

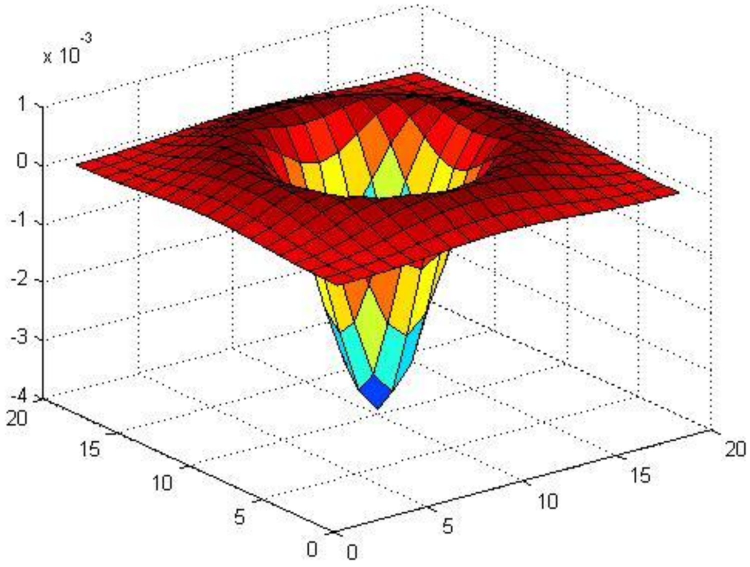


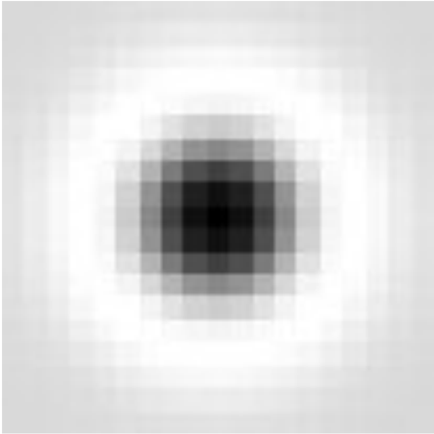


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Blob detection in 2D

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

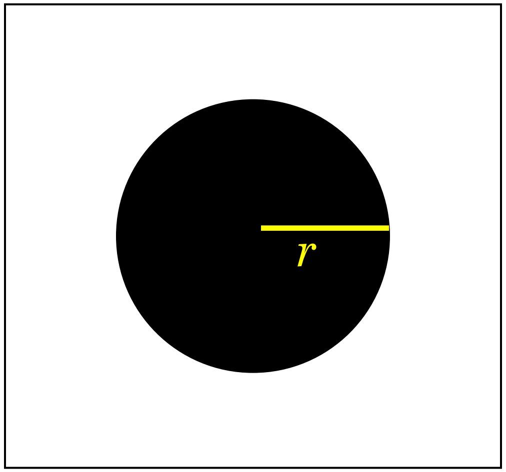


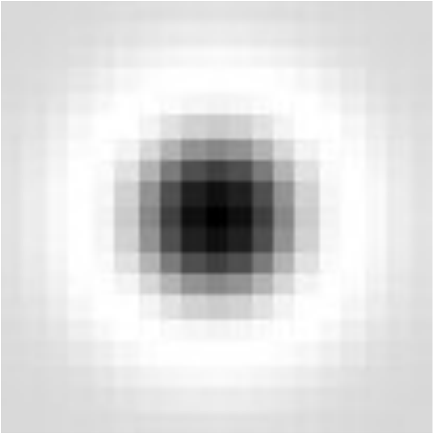


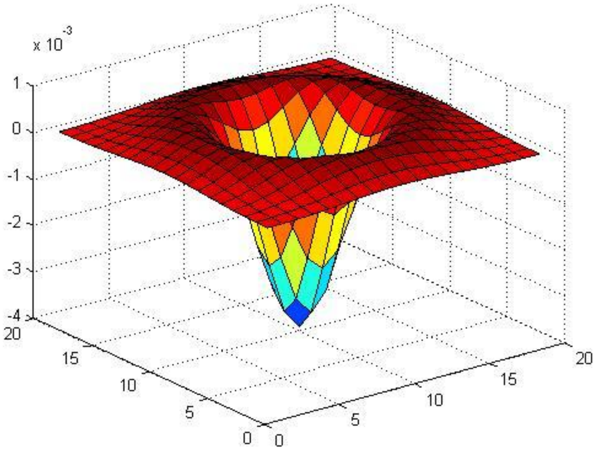
Scale-normalized: $$\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

Scale selection

- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?







imag
e

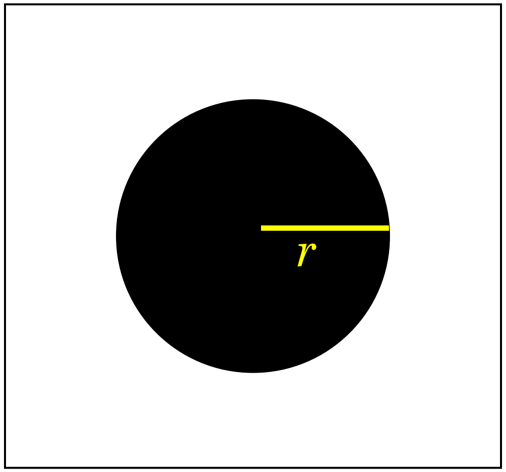
Laplacia
n

Scale selection

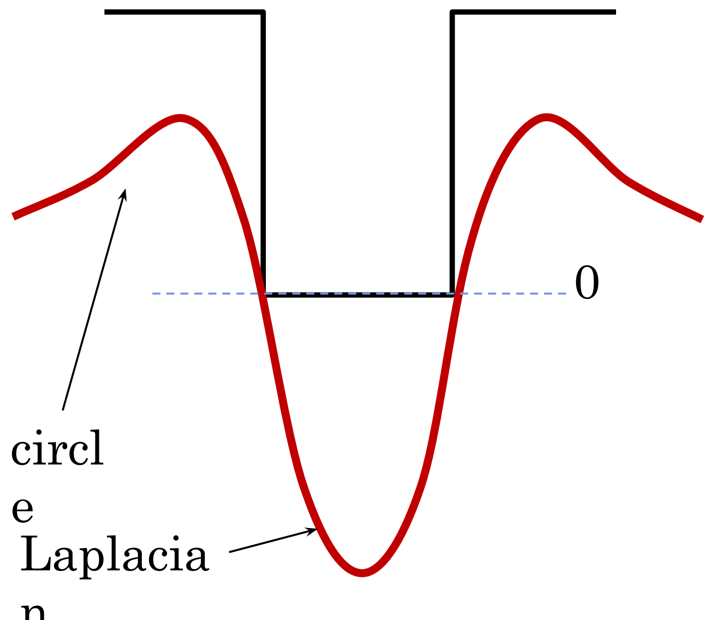
- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- To get maximum response, the zeros of the Laplacian have to be aligned with the circle
- The Laplacian is given by:

$$(x^2 + y^2 - 2\sigma^2)\, e^{-(x^2+y^2)/2\sigma^2} \,/\, 2\pi\sigma^6$$

- Therefore, the maximum response occurs at $\sigma = r/\sqrt{2}$.

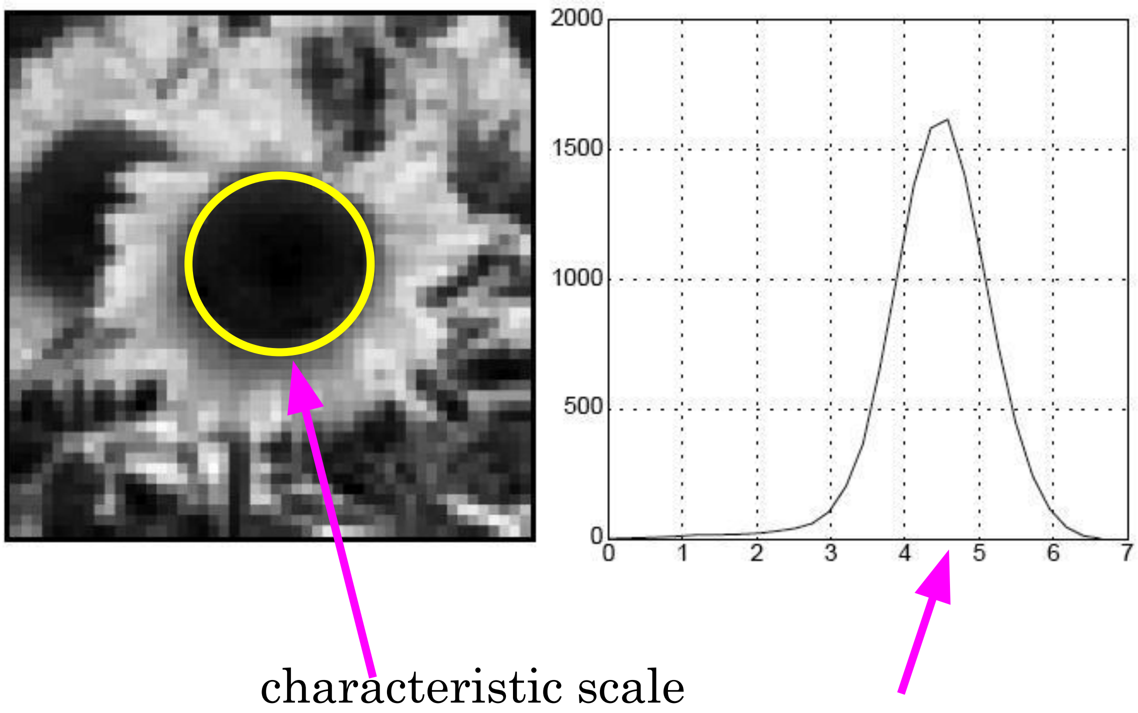


imag
e



Characteristic scale

- We define the characteristic scale of a blob as the scale that produces peak of Laplacian response in the blob center



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

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales

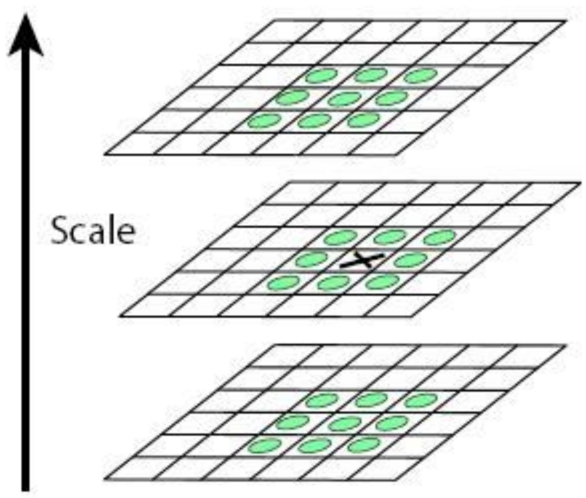
Scale-space blob detector: Example

Scale-space blob detector: Example

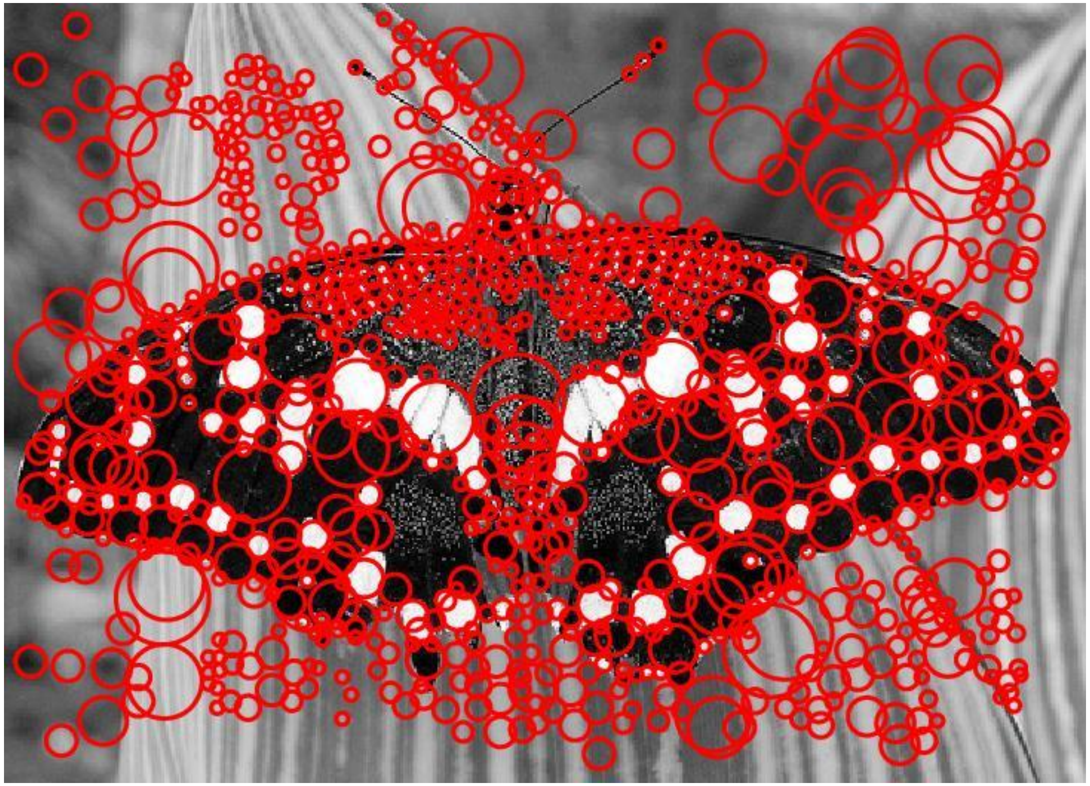
sigma = 11.9912

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales
2. Find maxima of squared Laplacian response in scale-space



Scale-space blob detector: Example



Outline

- Blob detection
  - Brief of Gaussian filter
  - Scale selection
  - Lapacian of Gaussian (LoG) detector
  - Difference of Gaussian (DoG) detector
  - Affine co-variant region

Efficient implementation

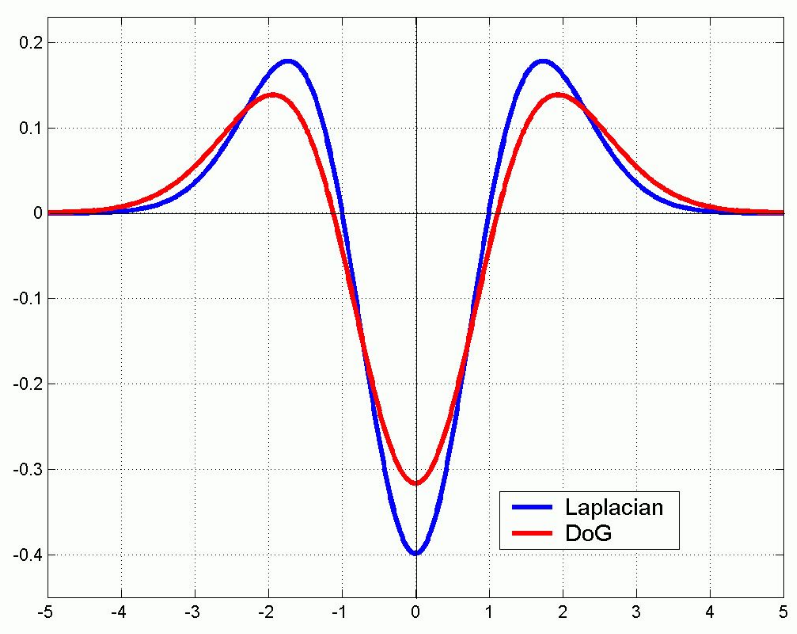
- Approximating the Laplacian with a difference of Gaussians:

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

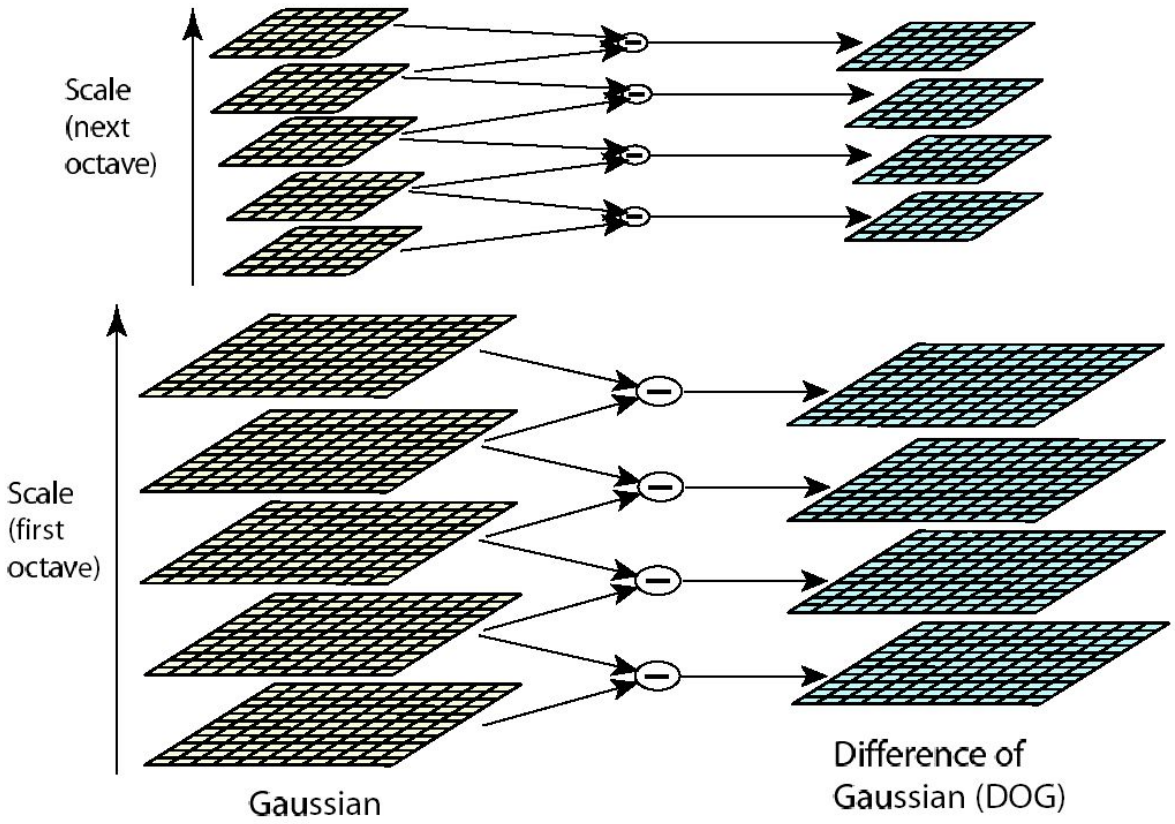
(Laplacian)

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

(Difference of Gaussians)



Efficient implementation



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

Invariance and covariance properties

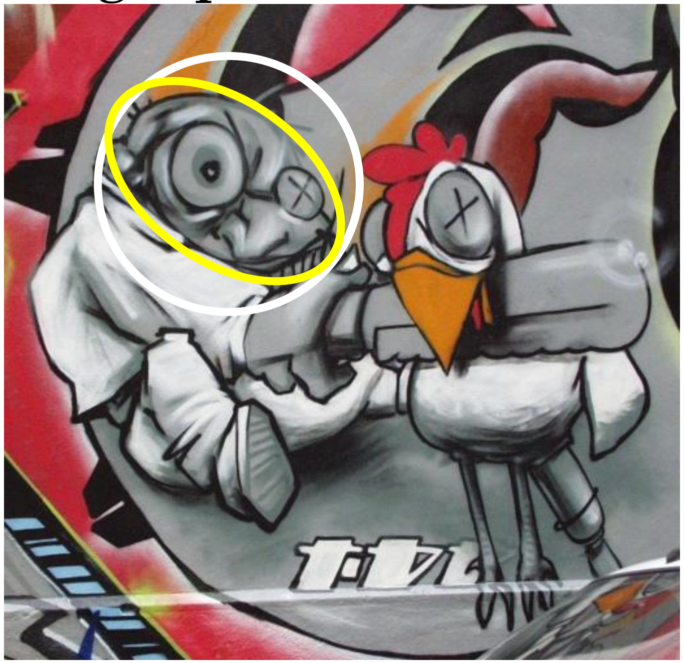
- Laplacian (blob) response is *invariant* w.r.t. rotation and scaling
- Blob location and scale is *covariant* w.r.t. rotation and scaling
- What about intensity change?

Outline

- Blob detection
  - Brief of Gaussian filter
  - Scale selection
  - Lapacian of Gaussian (LoG) detector
  - Difference of Gaussian (DoG) detector
  - <u>Affine co-variant region</u>

Achieving affine covariance

- Affine transformation approximates viewpoint changes for roughly planar objects and roughly orthographic cameras



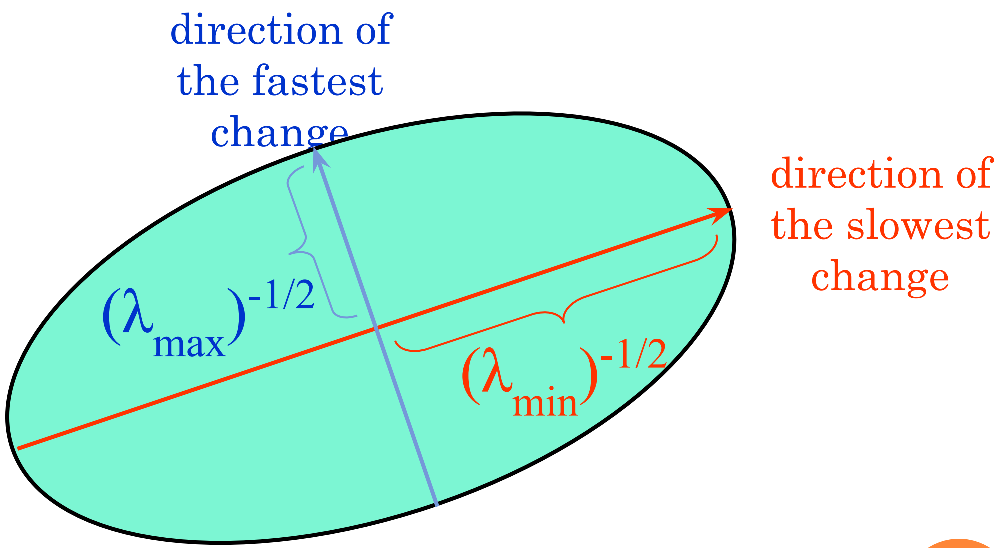
Achieving affine covariance

Consider the second moment matrix of the window containing the blob:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

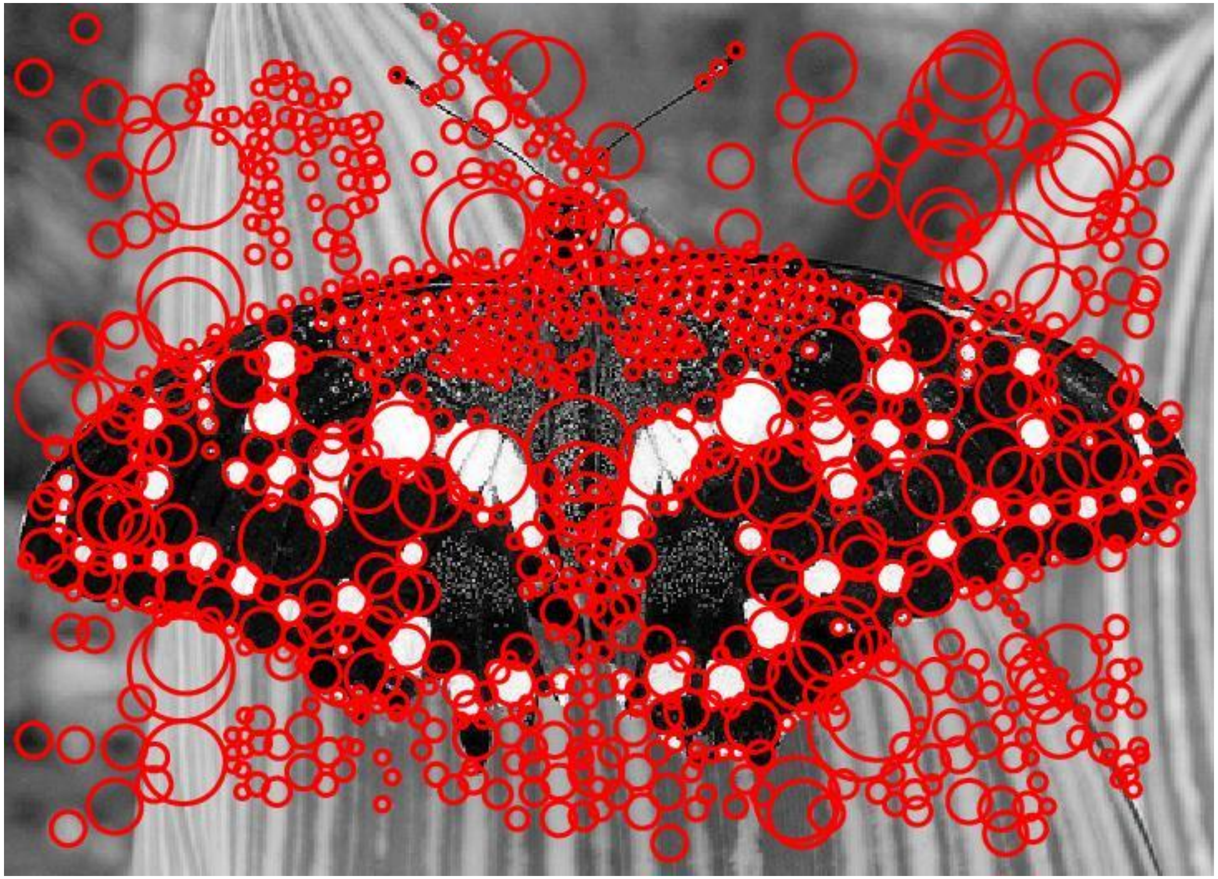
Recall:

$$[u \;\; v] \;\; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$



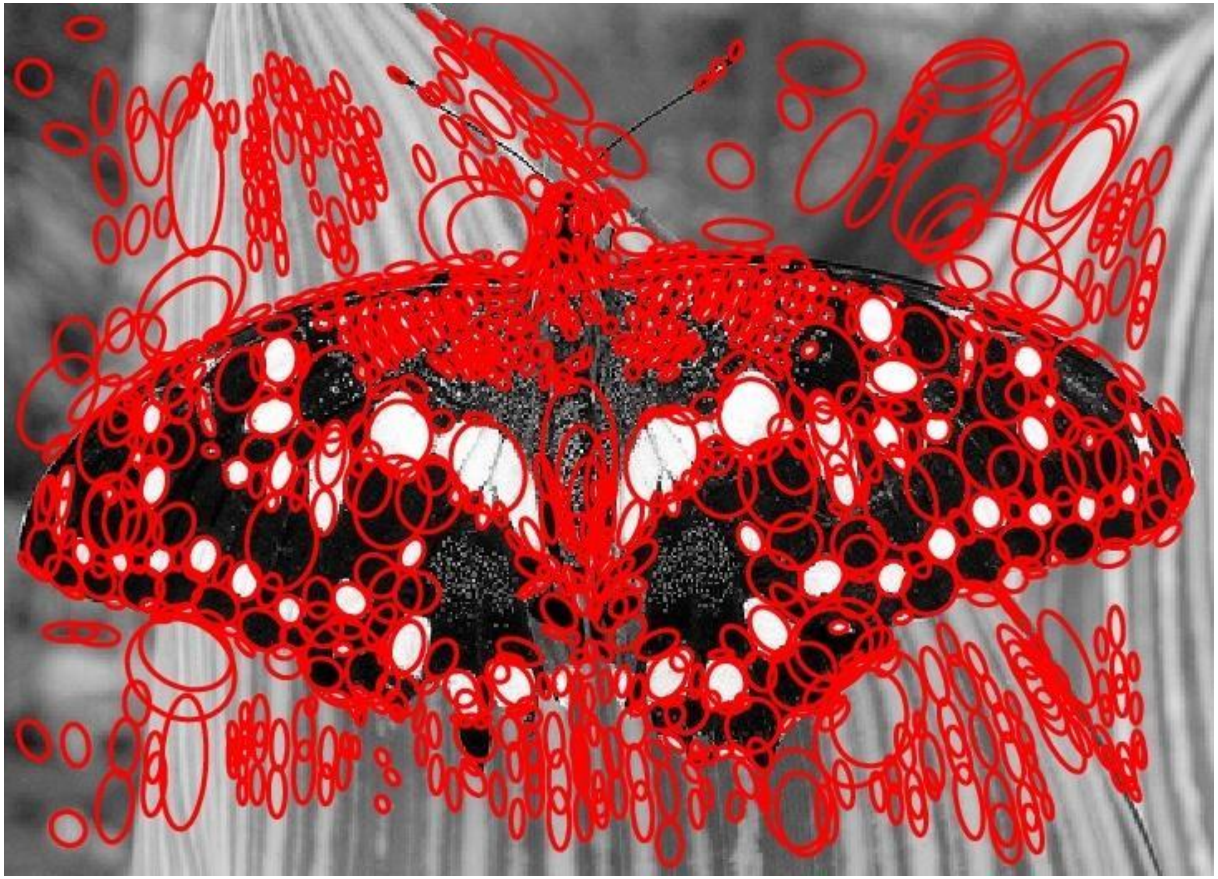
This ellipse visualizes the “characteristic shape” of the window

Affine adaptation example



Scale-invariant regions (blobs)

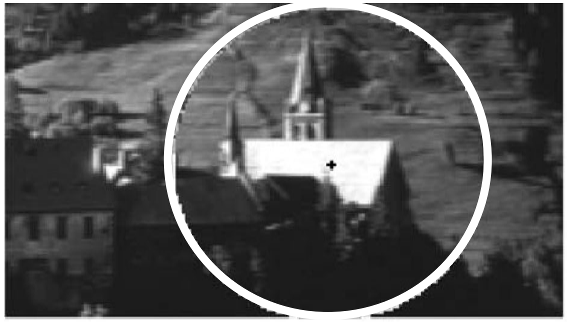
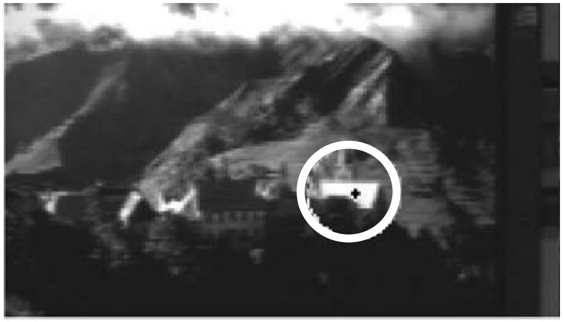
Affine adaptation example



Affine-adapted blobs

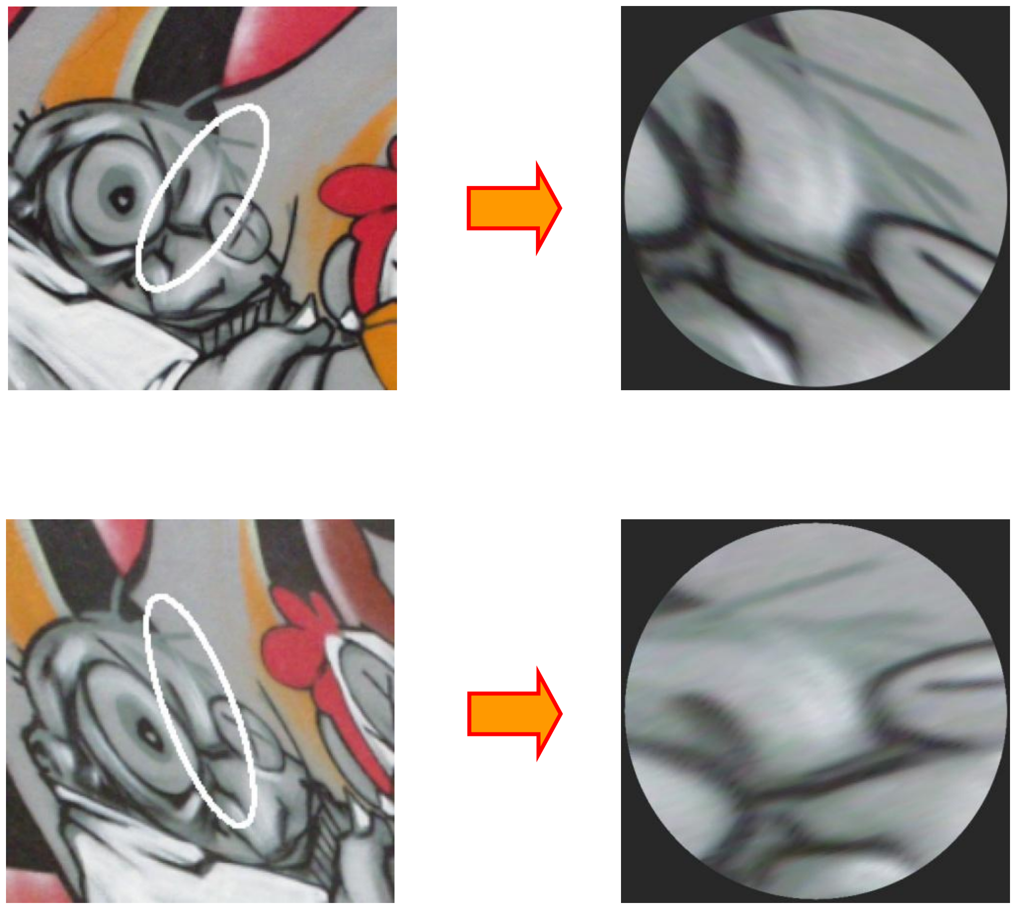
From covariant detection to invariant description

- Geometrically transformed versions of the same neighborhood will give rise to regions that are related by the same transformation
- What to do if we want to compare the appearance of these image regions?
  - *Normalization*: transform these regions into same-size circles

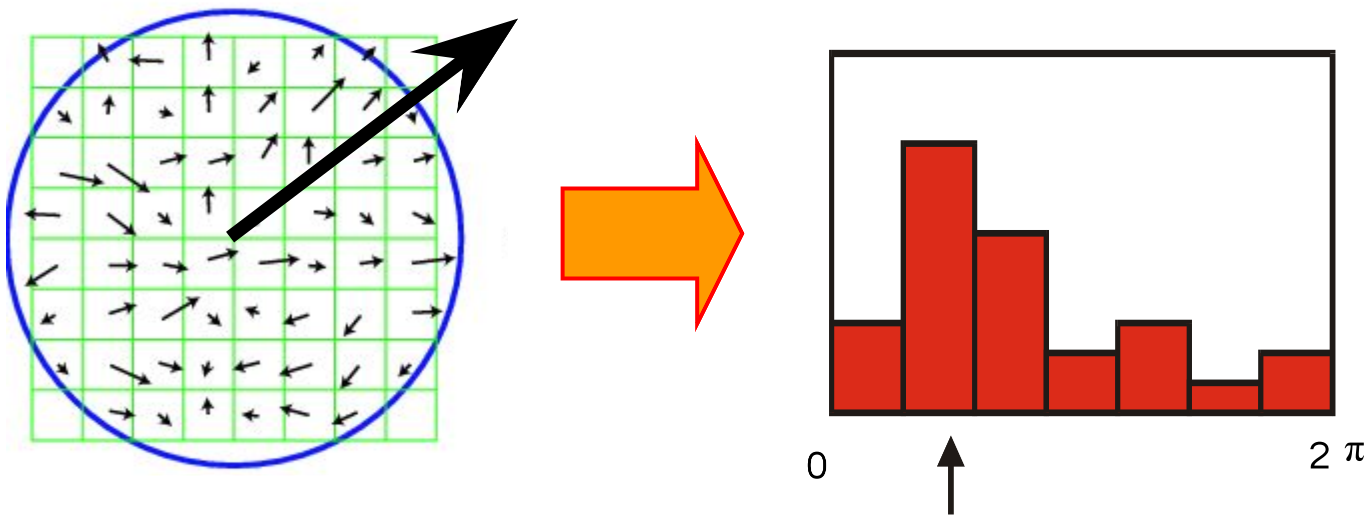
Affine normalization

- Problem: There is no unique transformation from an ellipse to a unit circle
  - We can rotate or flip a unit circle, and it still stays a unit circle

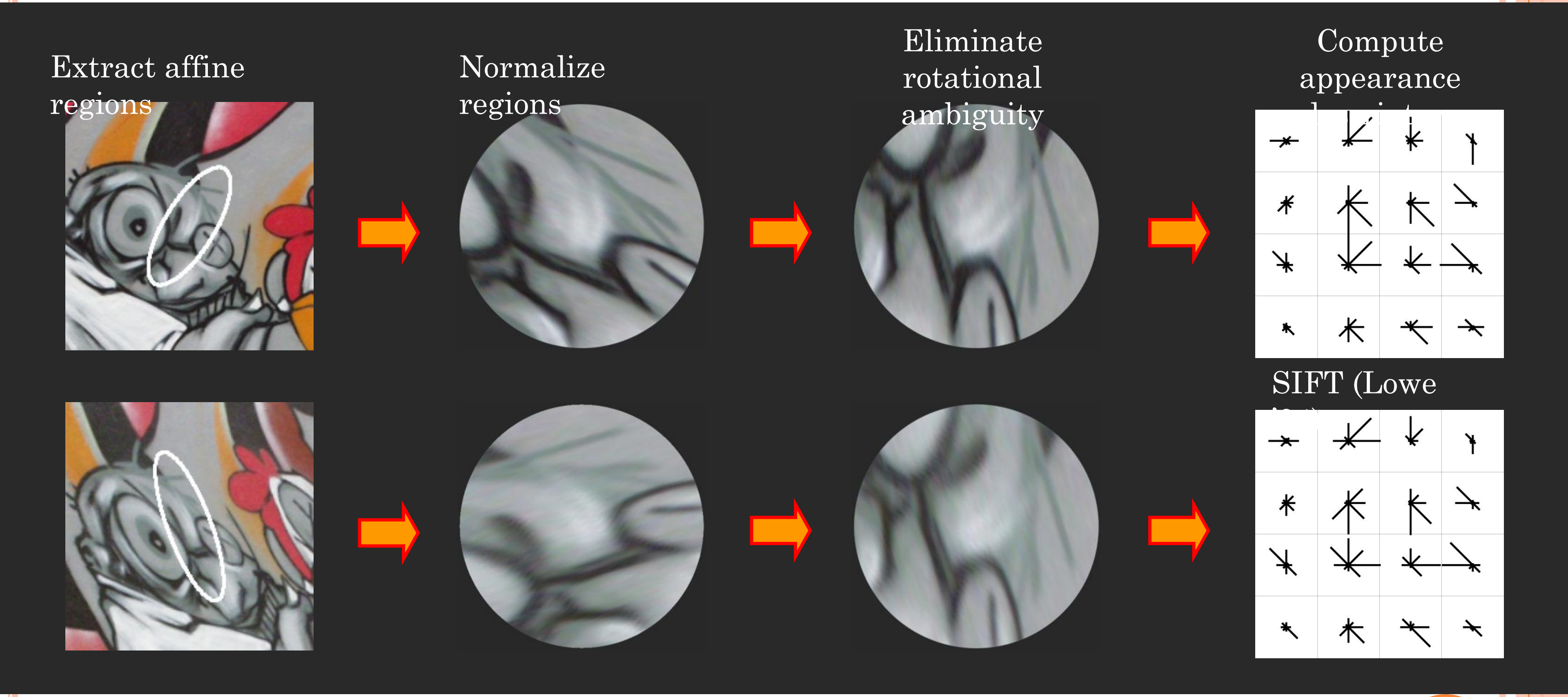


Eliminating rotation ambiguity

- To assign a unique orientation to circular image windows:
  - Create histogram of local gradient directions in the patch
  - Assign canonical orientation at peak of smoothed histogram

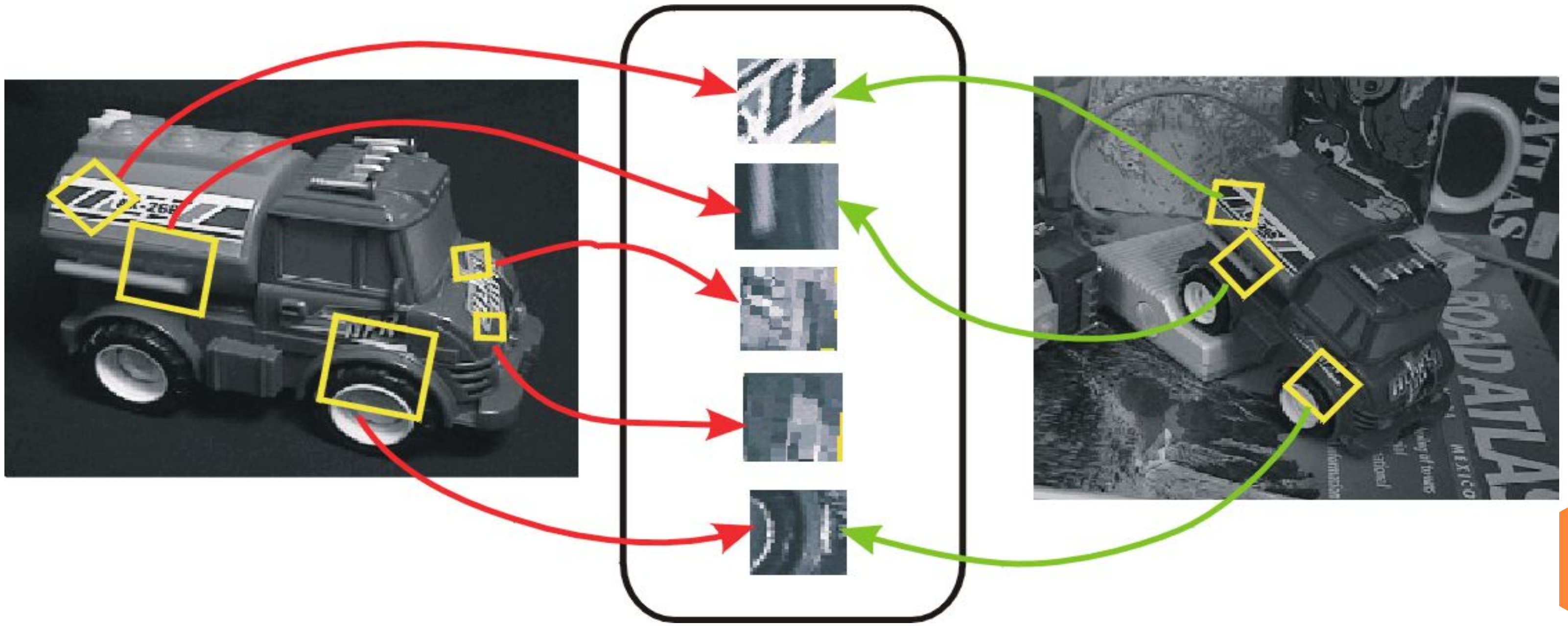


From covariant regions to invariant features



Invariance vs. covariance

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



Covariant detection => invariant description

David G. Lowe, "*Distinctive Image Features from Scale-Invariant Keypoints*", International Journal of Computer Vision, Vol. 60, No. 2, pp. 91-110

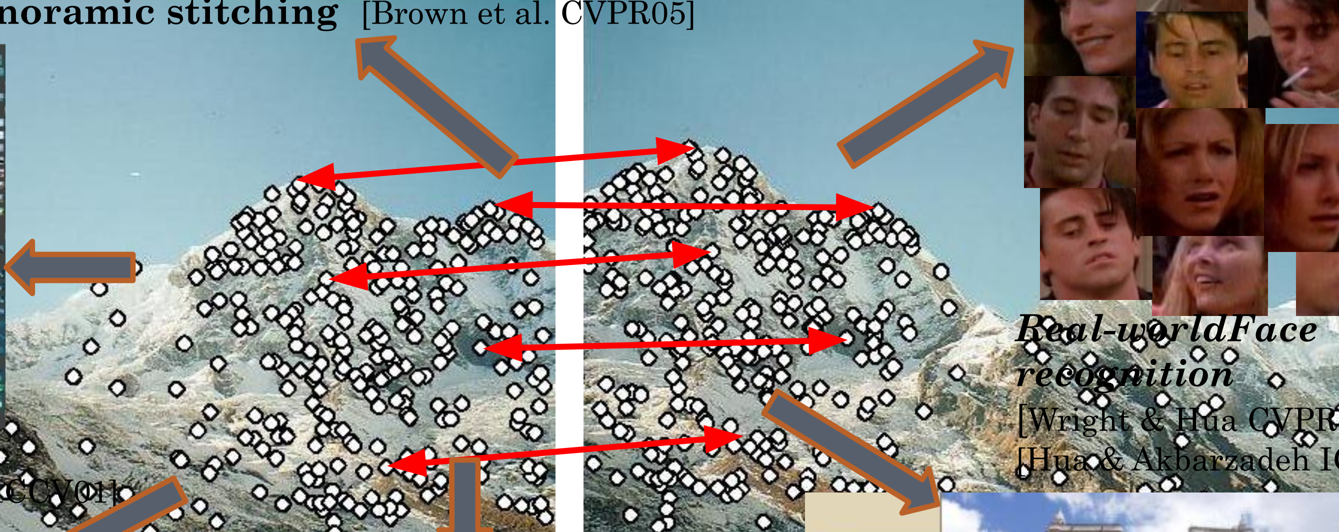
- *6291* as of 02/28/2010; 22481 as of 02/03/2014
- Our goal is to design the best local image descriptors in the world.

Lecture III: Part I

Learning Local Feature Descriptor

Panoramic stitching [Brown et al. CVPR05]

Image databases
[Mikolajczyk & Schmid ICCV01]

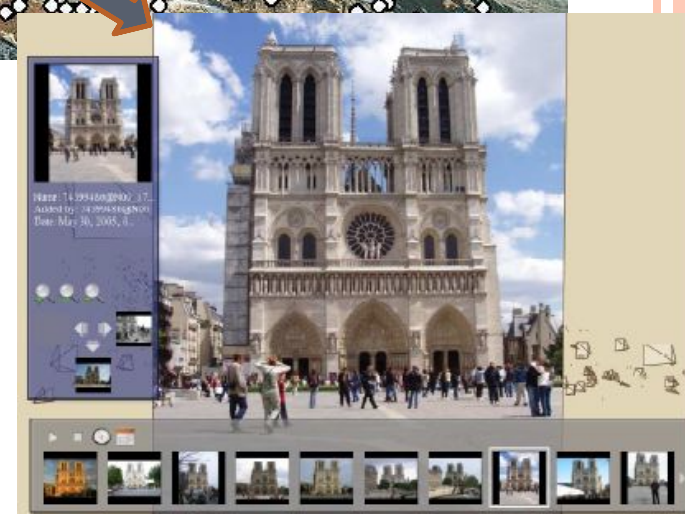


Real-worldFace recognition
[Wright & Hua CVPR09]
[Hua & Akbarzadeh ICCV09]

Object or location recognition
[Nister & Stewenius CVPR06]
[Schindler et al. CVPR07]

Courtesy of Seitz & Szeliski

Robot navigation
[Deans et al.



3D reconstruction
[Snavely et al. SIGGRAPH06]

Typical matching process

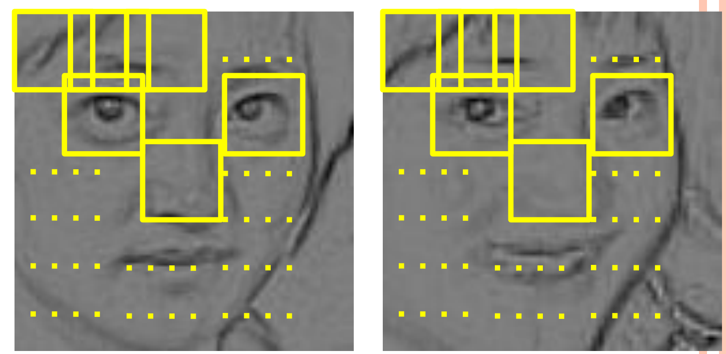
Image 1

Image 2

Interest point/region detection (sparse)

Image 1

Image 2

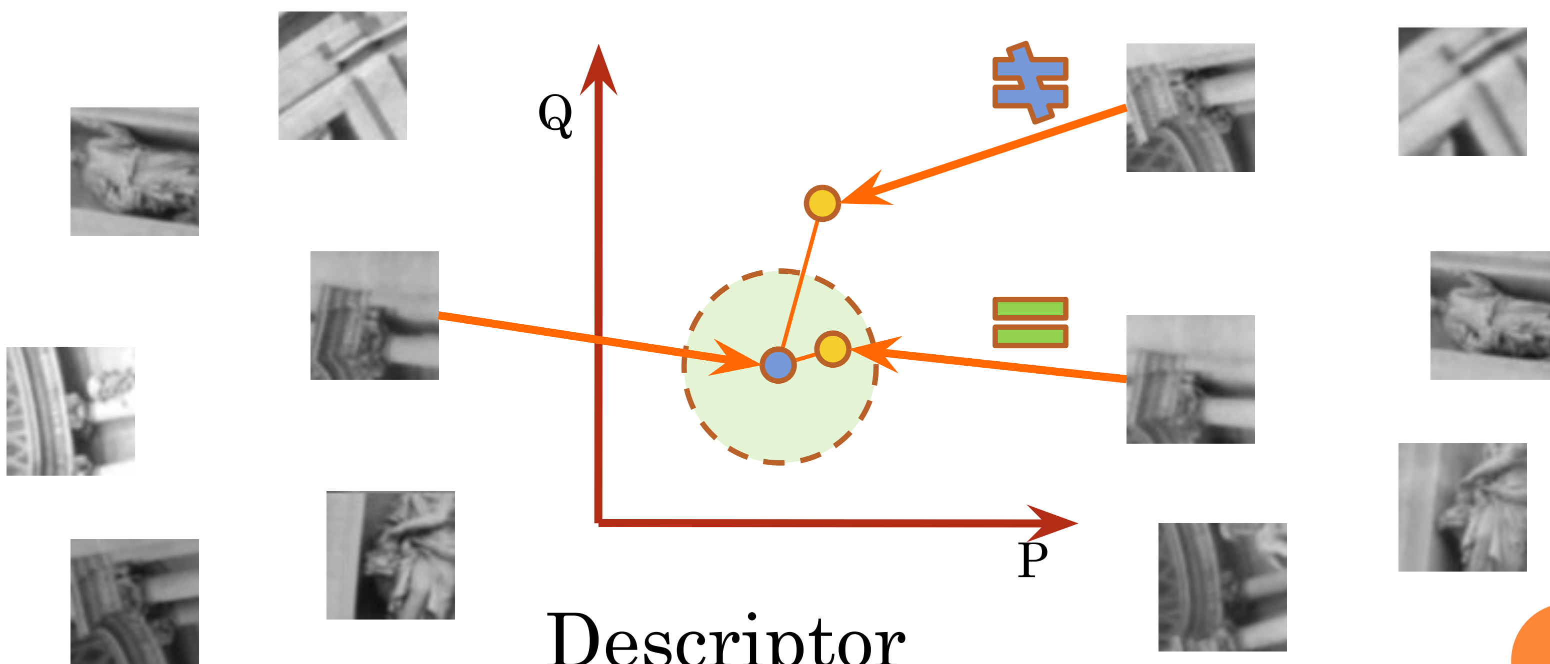


Dense sampling of image patches

Typical matching process

Image 1

Image 2



Descriptor space

Problem to solve

Learning a function of a local image patch

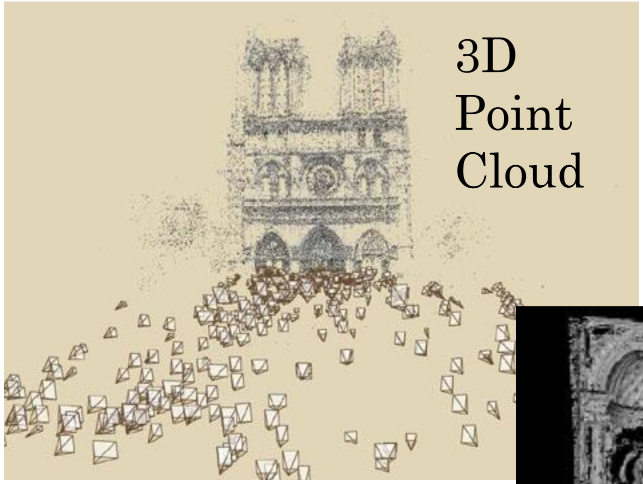
descriptor = f ( *)*

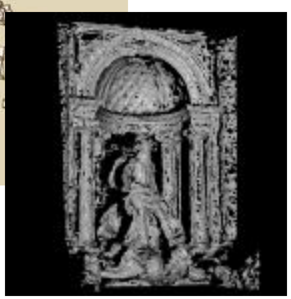
s.t. a nearest neighbor classifier is optimal

- To obtain the most *discriminative*, *compact*, and *computationally efficient* local image descriptors.
  - How can we get ground truth data?
  - What is the form of the descriptor function f(.)?
  - What is the measure for optimality?

How can we get ground truth data?

Training data

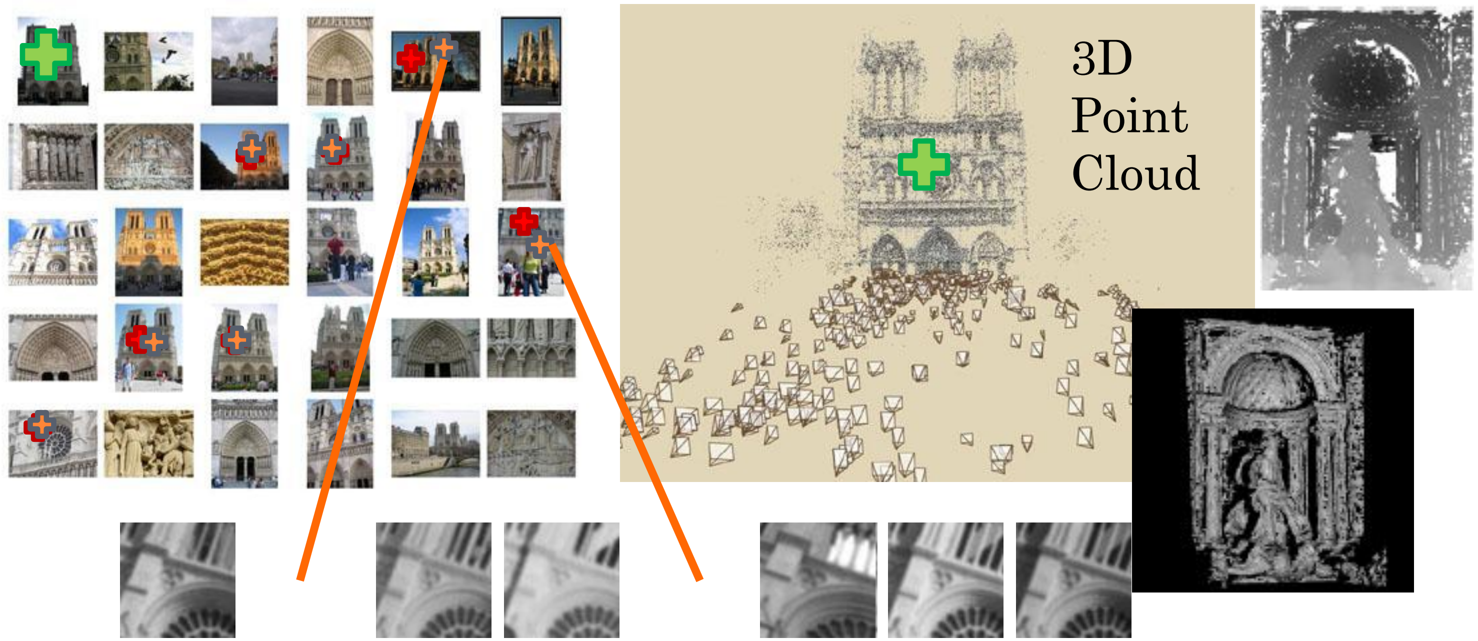




Multiview stereo = Training data

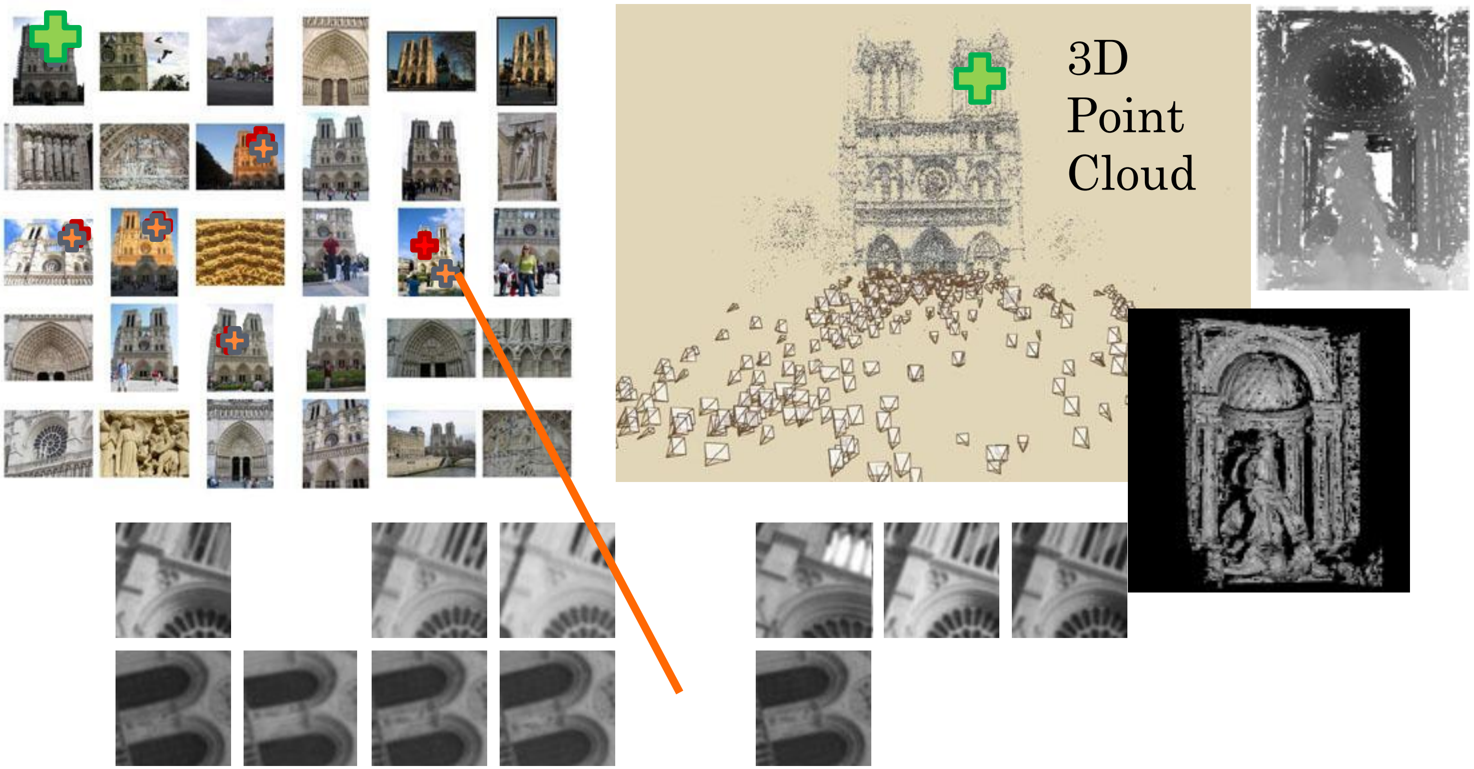
[Goesele et al. - ICCV'07] [Snavely et al. - SIGGRAPH'06]

Training data



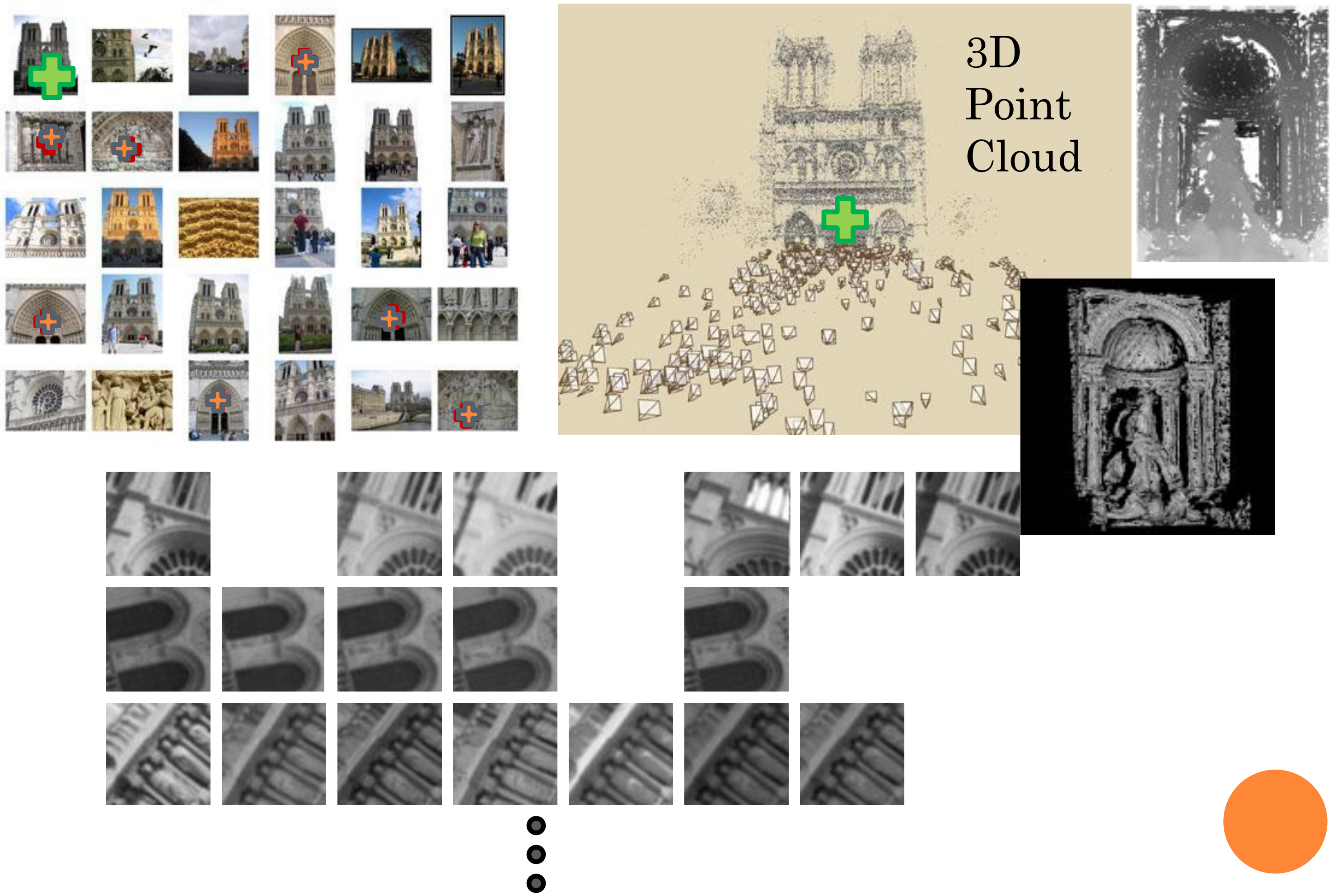
[Goesele et al. - ICCV'07] [Snavely et al. - SIGGRAPH'06]

Training data

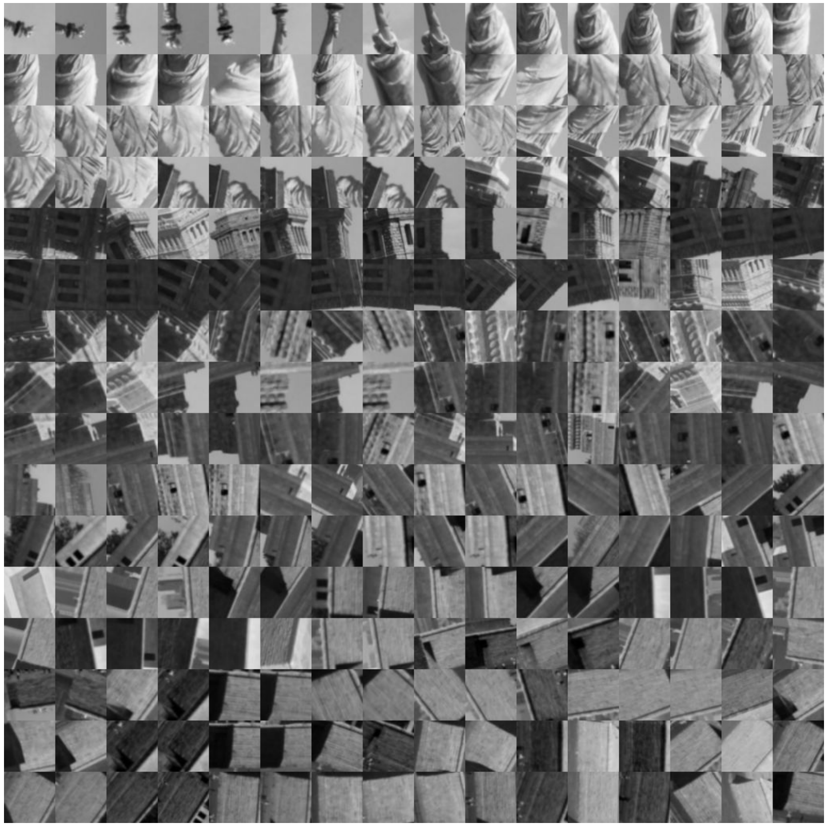


[Goesele et al. - ICCV'07] [Snavely et al. - SIGGRAPH'06]

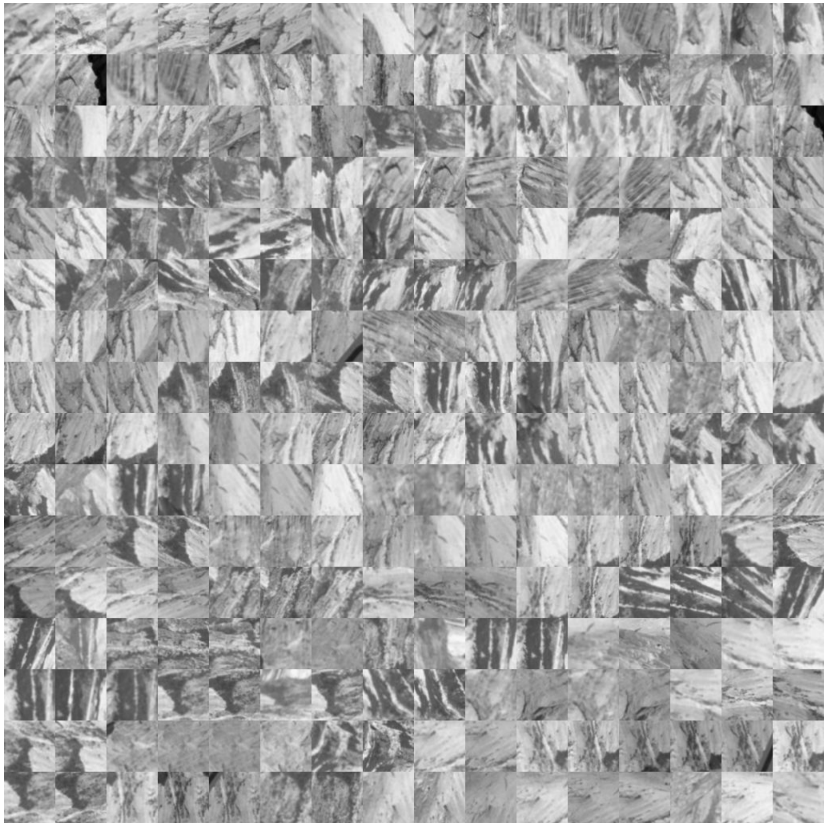
Training data



[Goesele et al. - ICCV'07] [Snavely et al. - SIGGRAPH'06]



Libert
y

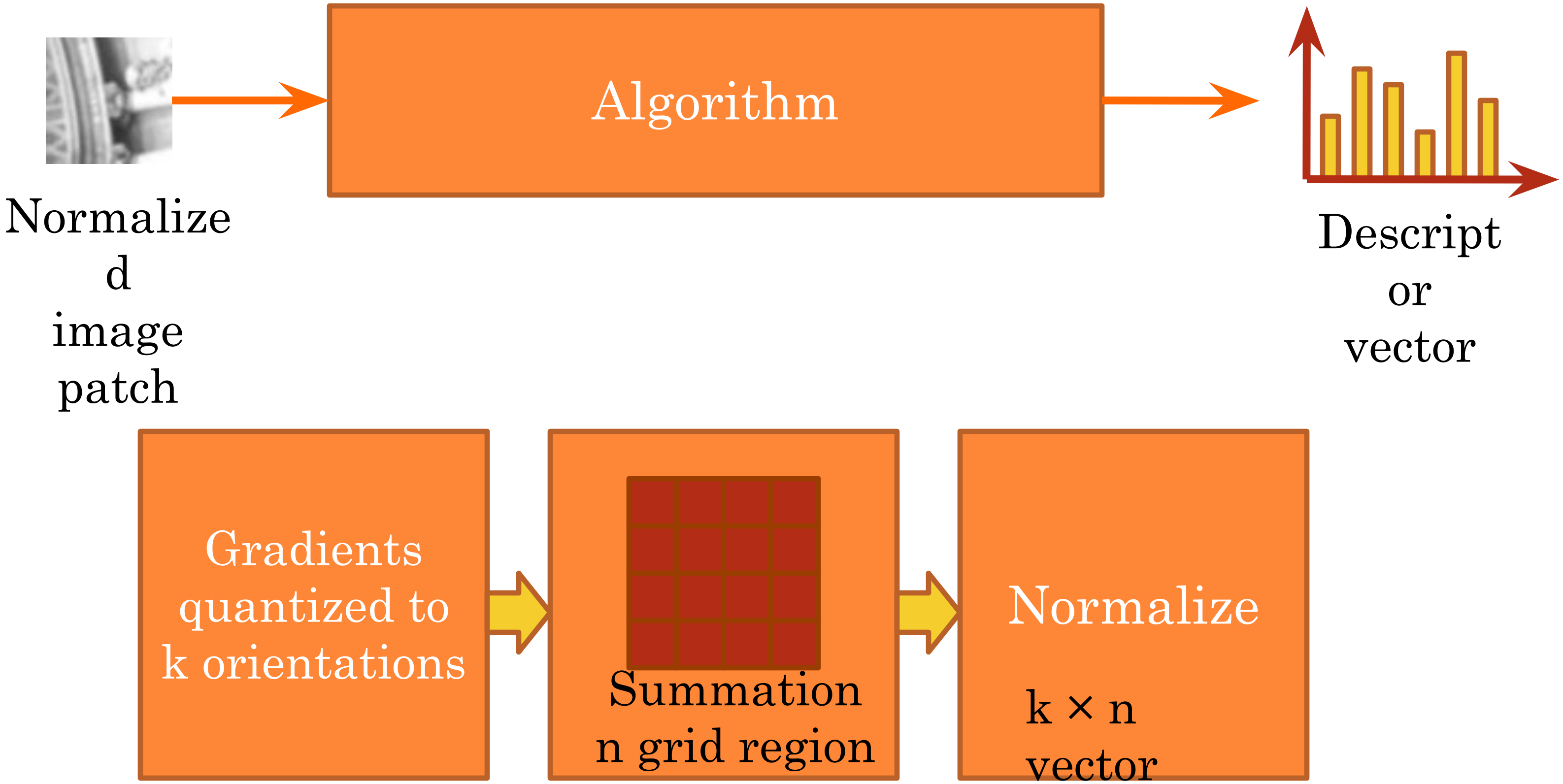


Yosemit
e

- Statue of liberty (New York) – Liberty
- Notre Dame (Paris) – Notre Dame
- Half Dome (Yosemite) - Yosemite
- http://www.cs.ubc.ca/~mbrown/patchdata/patchdata.html

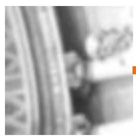
What is the form of the descriptor function?

Descriptor algorithms



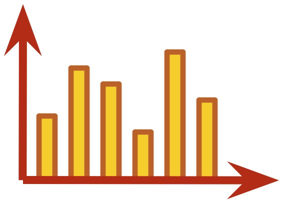
[SIFT – Lowe ICCV'99]

Descriptor algorithms



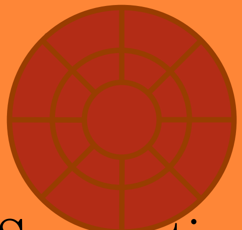
Normalized image patch

Algorithm



Descriptor vector

Gradients Quantized to k Orientations



Summation n Log-Polar region

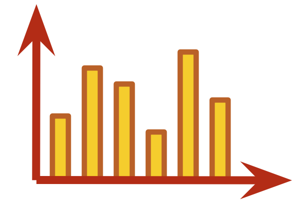
Normalize (plus PCA)

[GLOH – Mikolajczyk & Schmid PAMI'05]

Descriptor algorithms

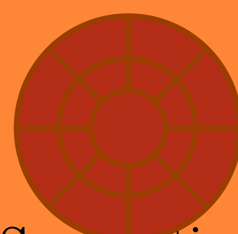
Normalize
d
image
patch

Algorithm



Descript
or
vector

Create
Edge Map

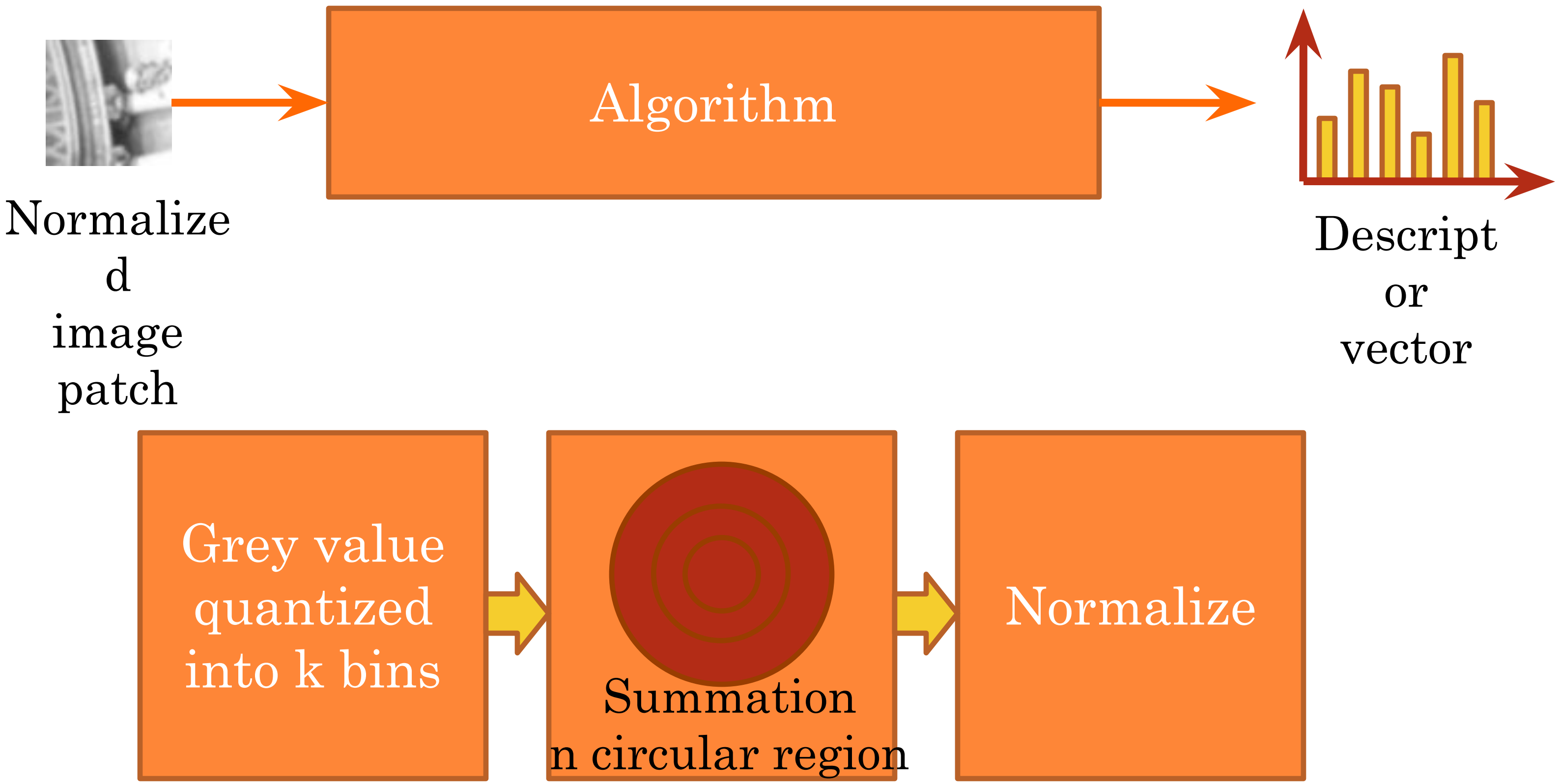


Summation
n Log-Polar region

Normalize

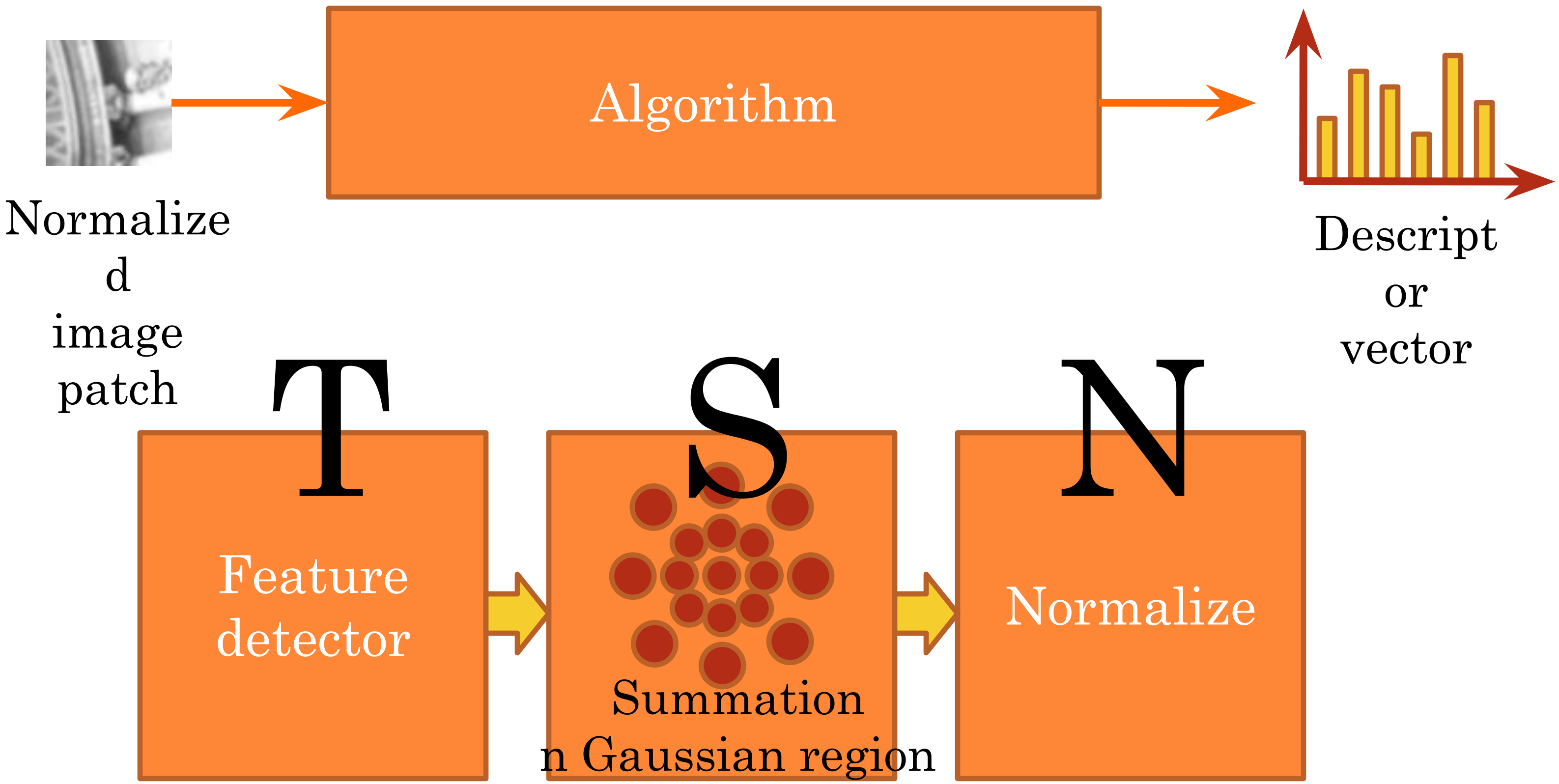
[Shape Context – Belongie et al, NIPS'00]

Descriptor algorithms



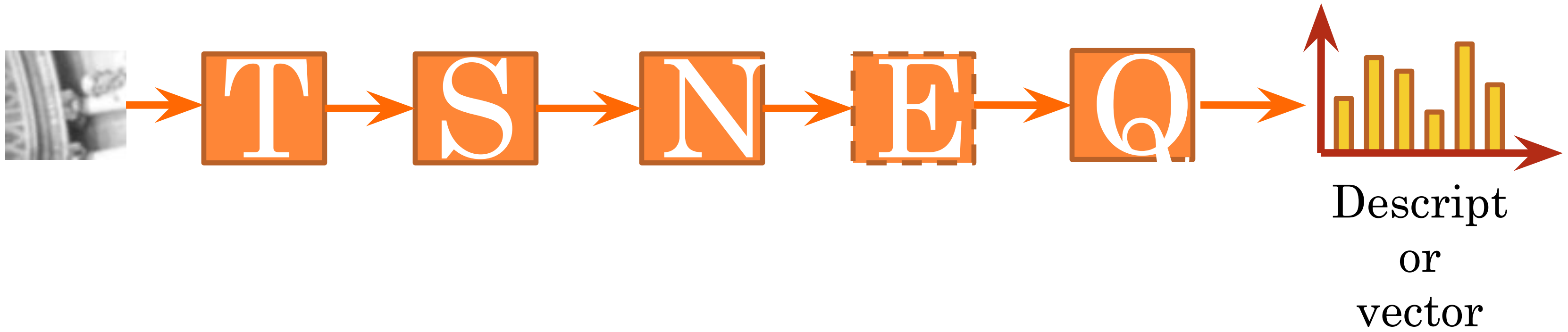
[Spin Image – Lazebnik et al, CVPR'03]

Descriptor algorithms

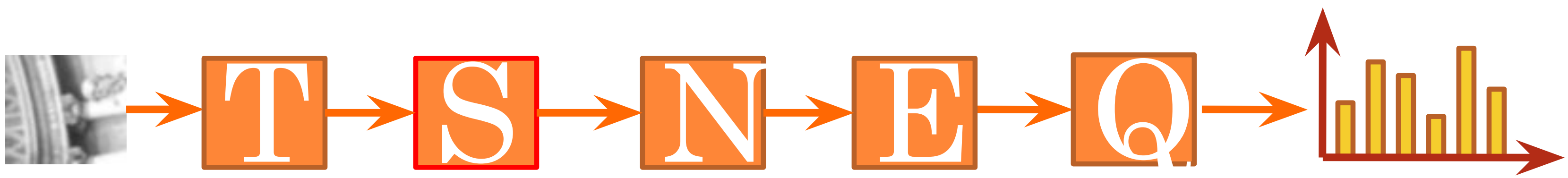


[Geometric Blur – Berg& Malik
CVPR'01]
[DAISY - Tola et al. CVPR'08]

Our descriptor algorithm

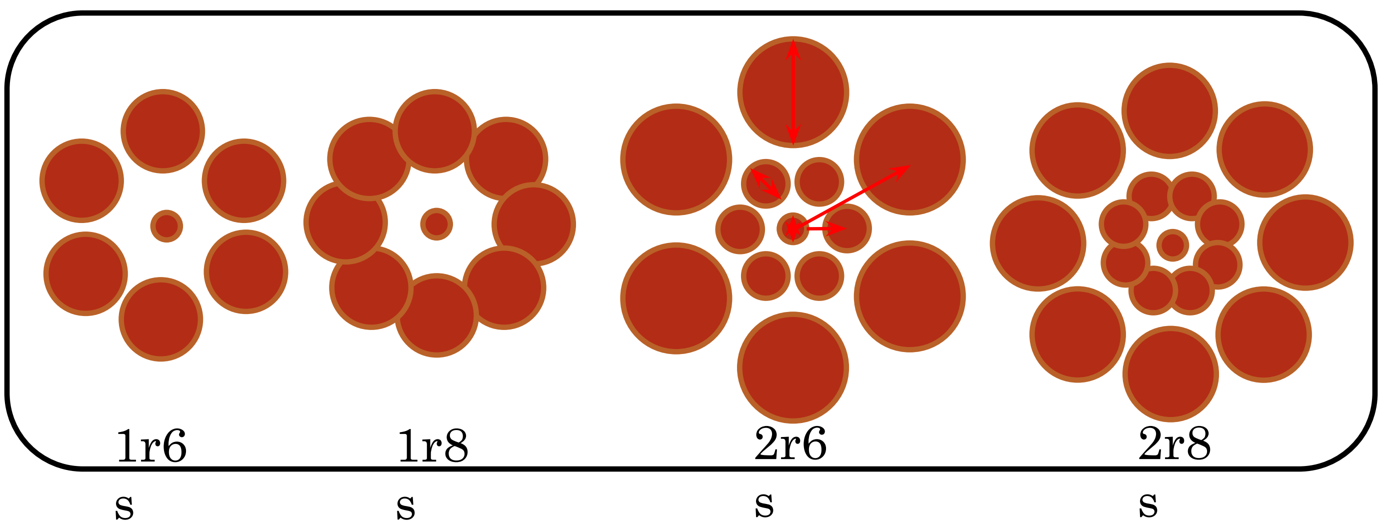


S-Block: Picking the best DAISY

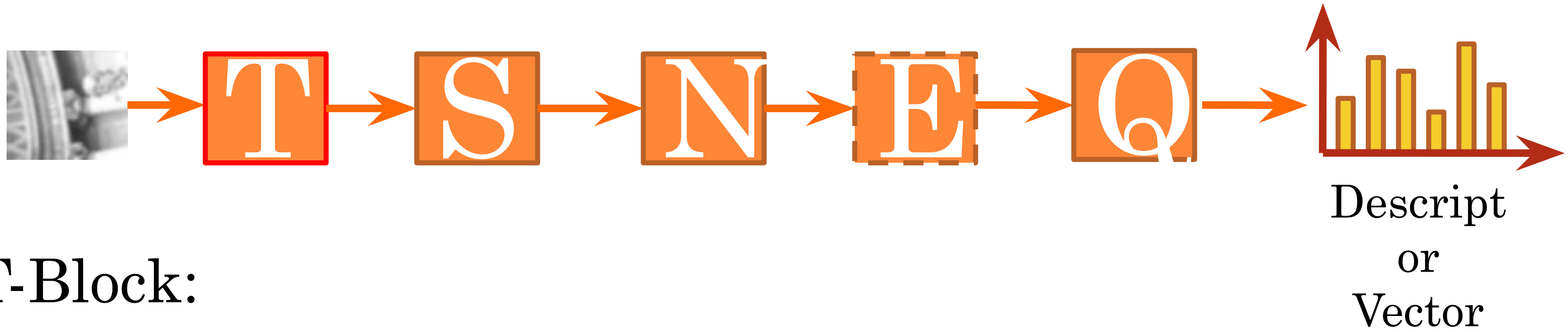


Descript
or
vector

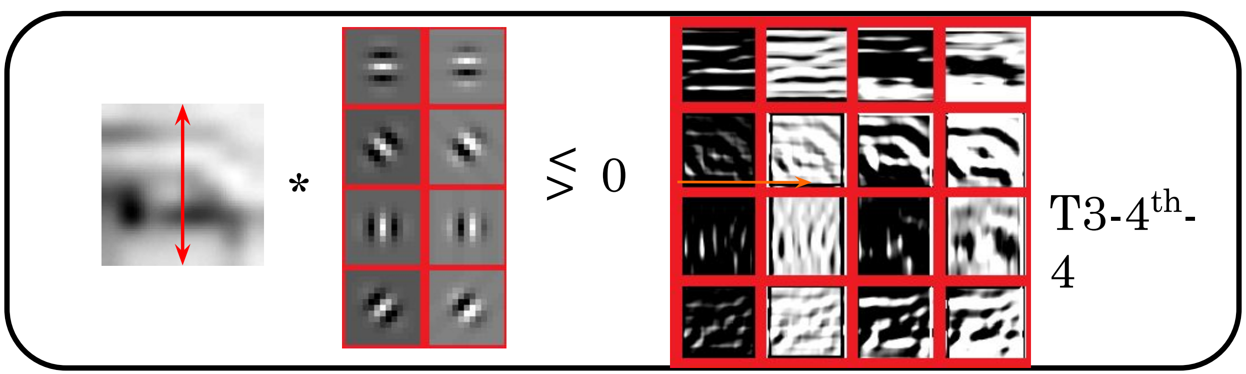
- S-Block: DAISY



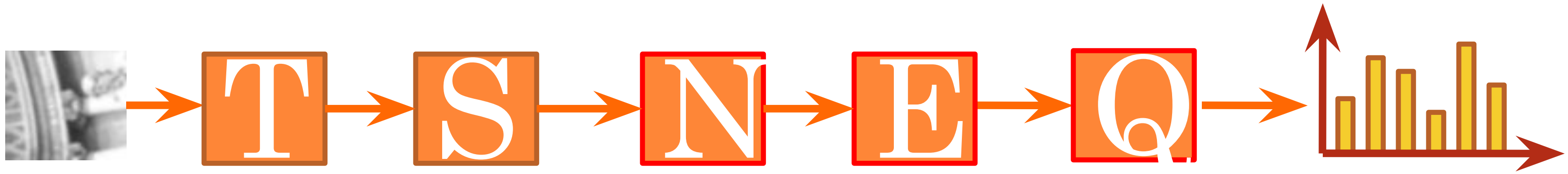
T-Block



- T-Block:
  - T1: Gradient bi-linearly weighted orientation binning
  - T2: Rectified gradient $\{|\nabla_x| - \nabla_x,\ |\nabla_x| + \nabla_x,\ |\nabla_y| - \nabla_y,\ |\nabla_y| + \nabla_y\}$
  - T3: Steerable filters



N-E-Q Blocks

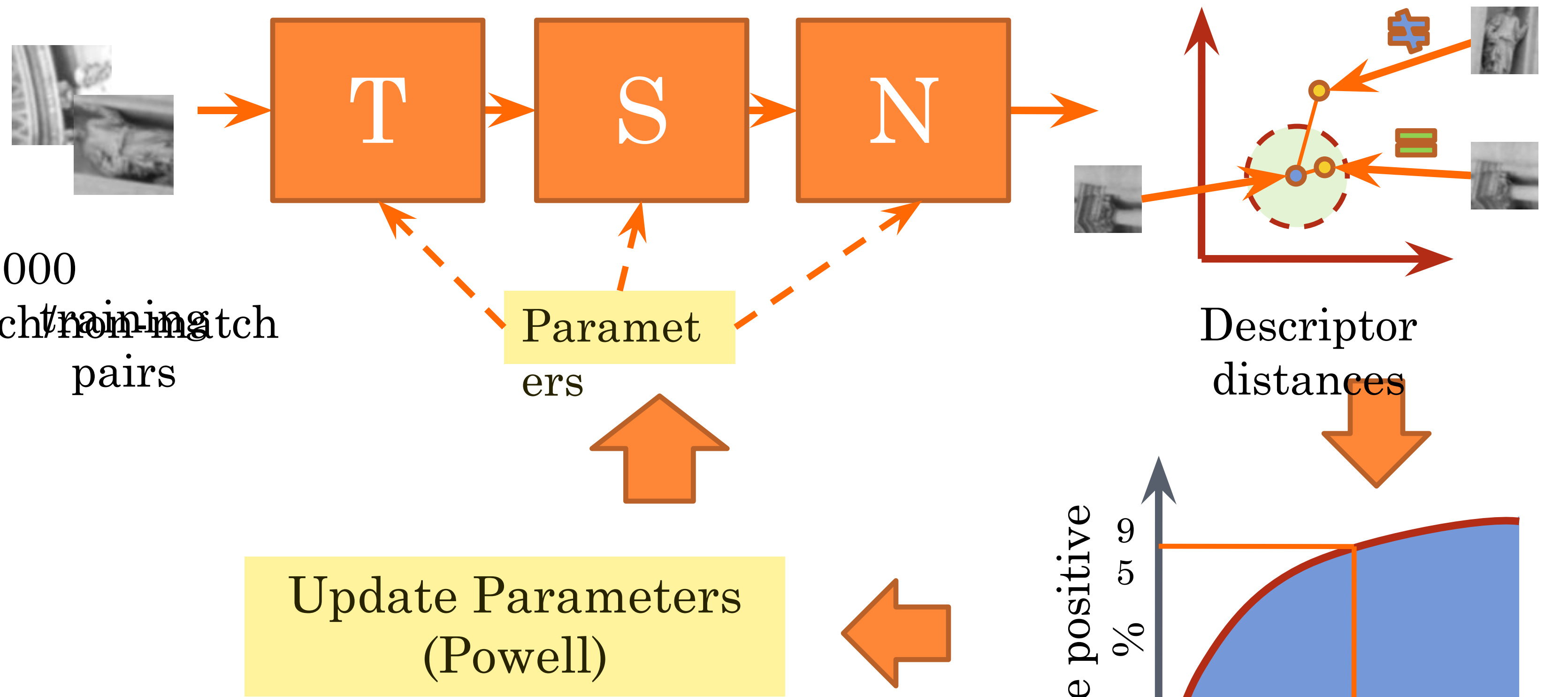


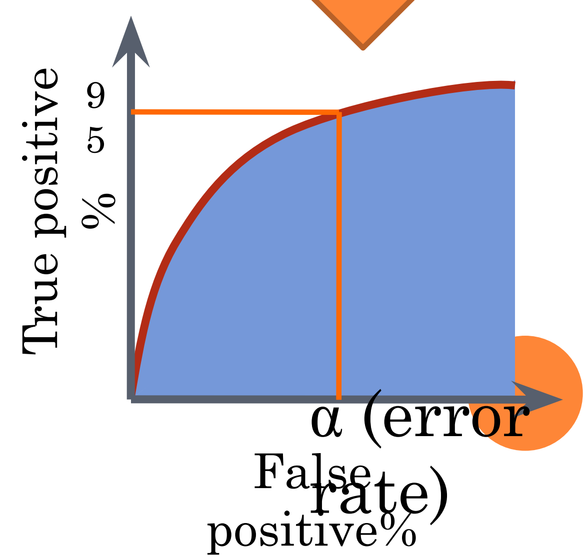
Descriptor Vector

- N-Block: Uniform v.s. SIFT-like normalization
- E-Block: Principle component analysis (PCA)
- Q-block: $q_i = \lfloor \beta L v_i \rfloor + \alpha$, β is learnt from data, $\alpha = 0.5$ if L is an odd number and $\alpha = 0$ otherwise.

What is the optimal criterion?

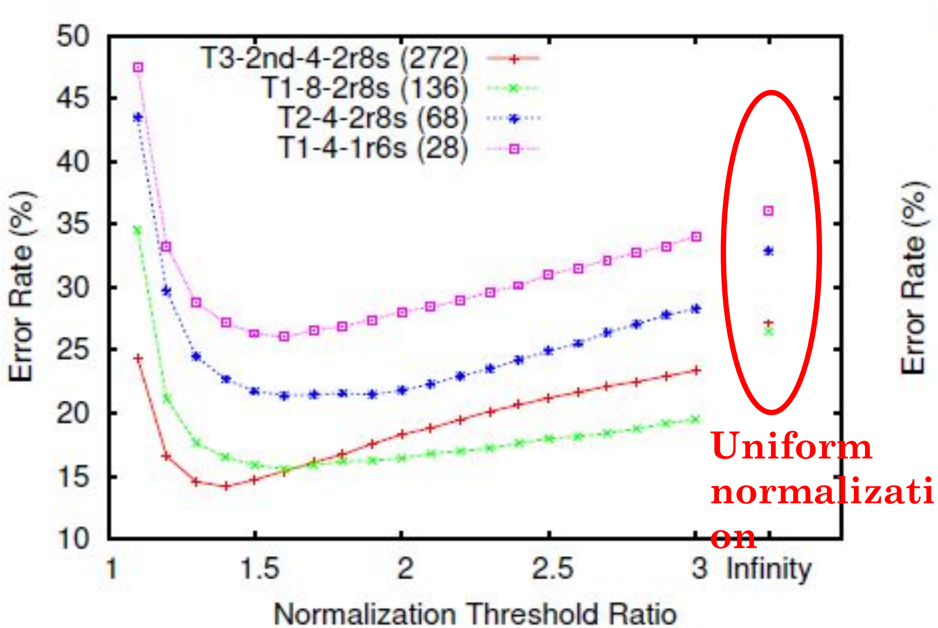
Discriminative learning and optimal criterion

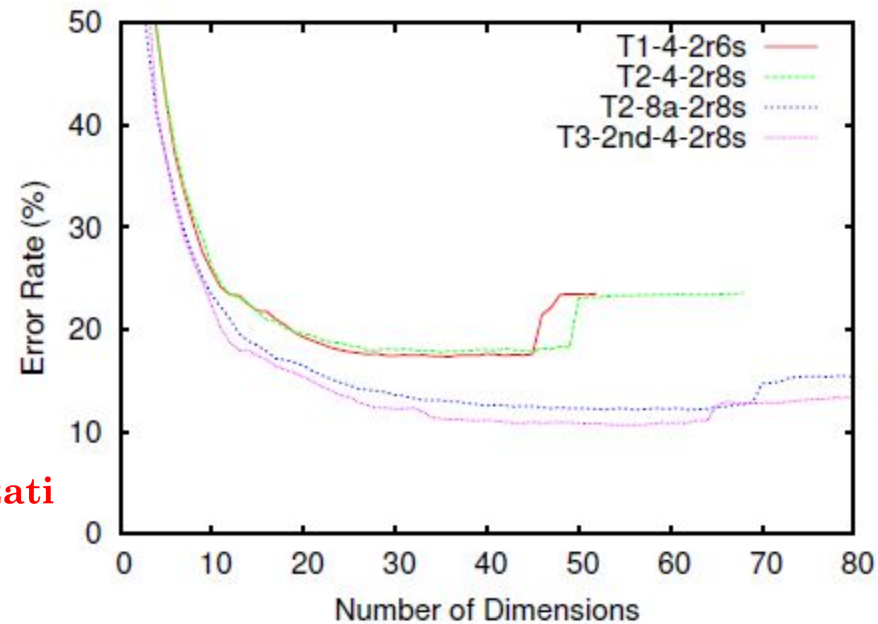




- *E-Block and Q-Block are optimized independently*

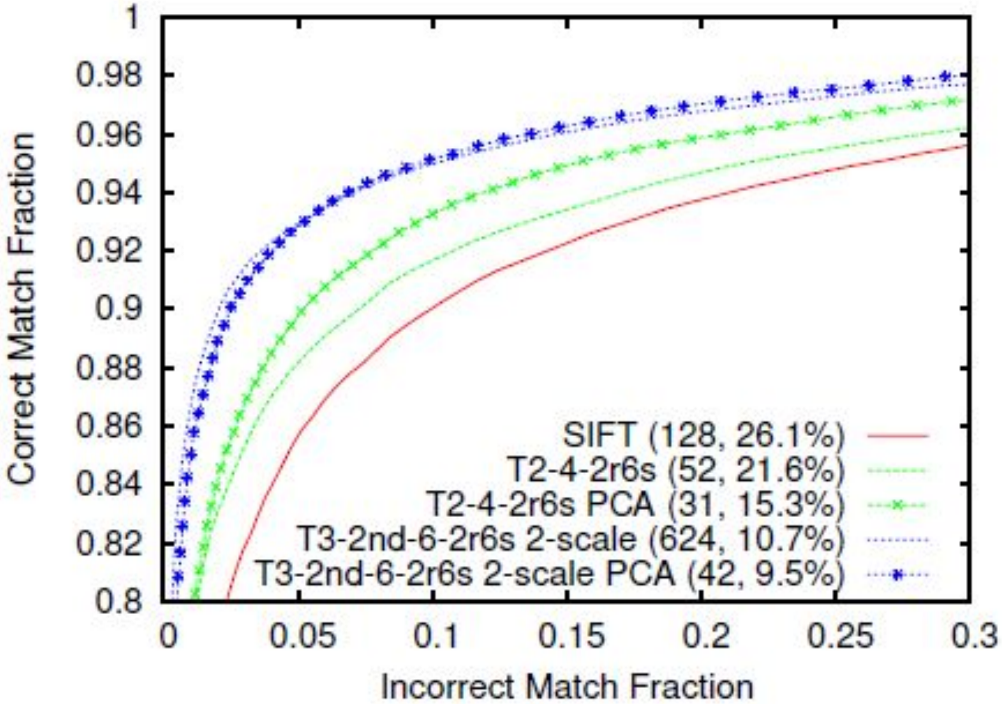
SIFT-like normalization & PCA

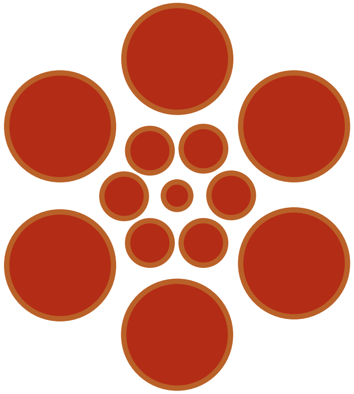




- SIFT like normalization has a clear sweet spot.
- PCA can usually reduce the dimension to 30~50.

The most discriminative DAISY

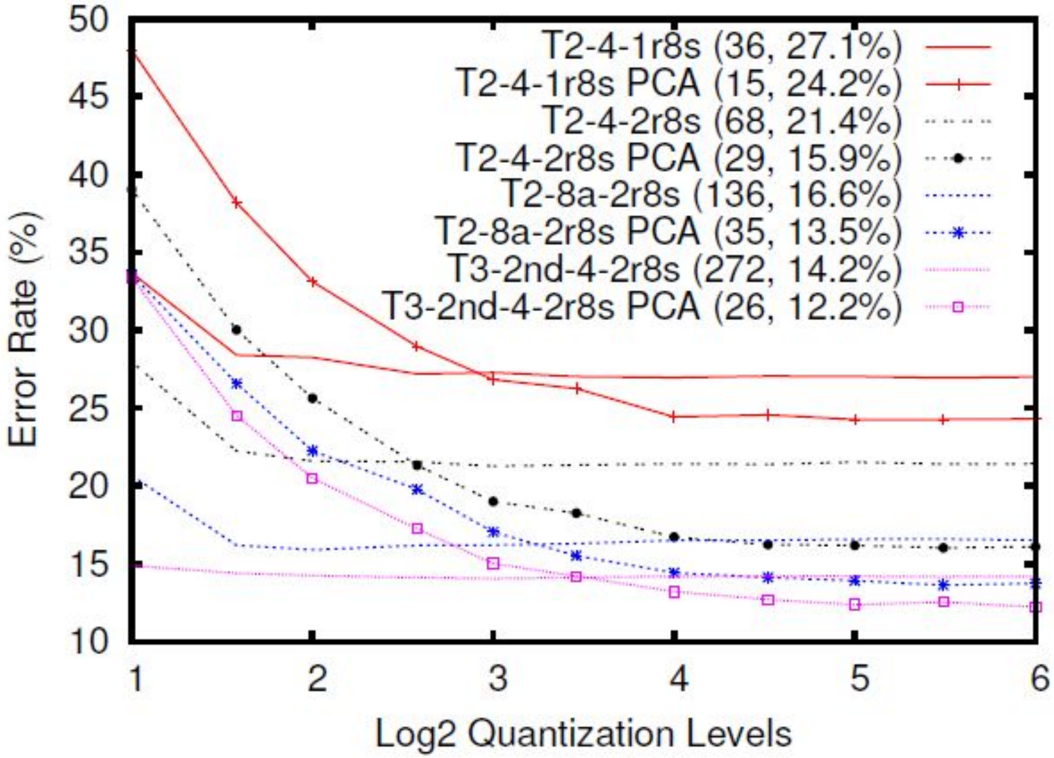




2r6
s

- Steerable filters at two spatial scales with PCA
  - T3-2^{nd}-6-2r6s + 2-scale + PCA: 42 dimension, 9.5%
  - $2 \times 6 \times 2 \times 2 \times (2 \times 6+1) = 624$ dimensions before PCA

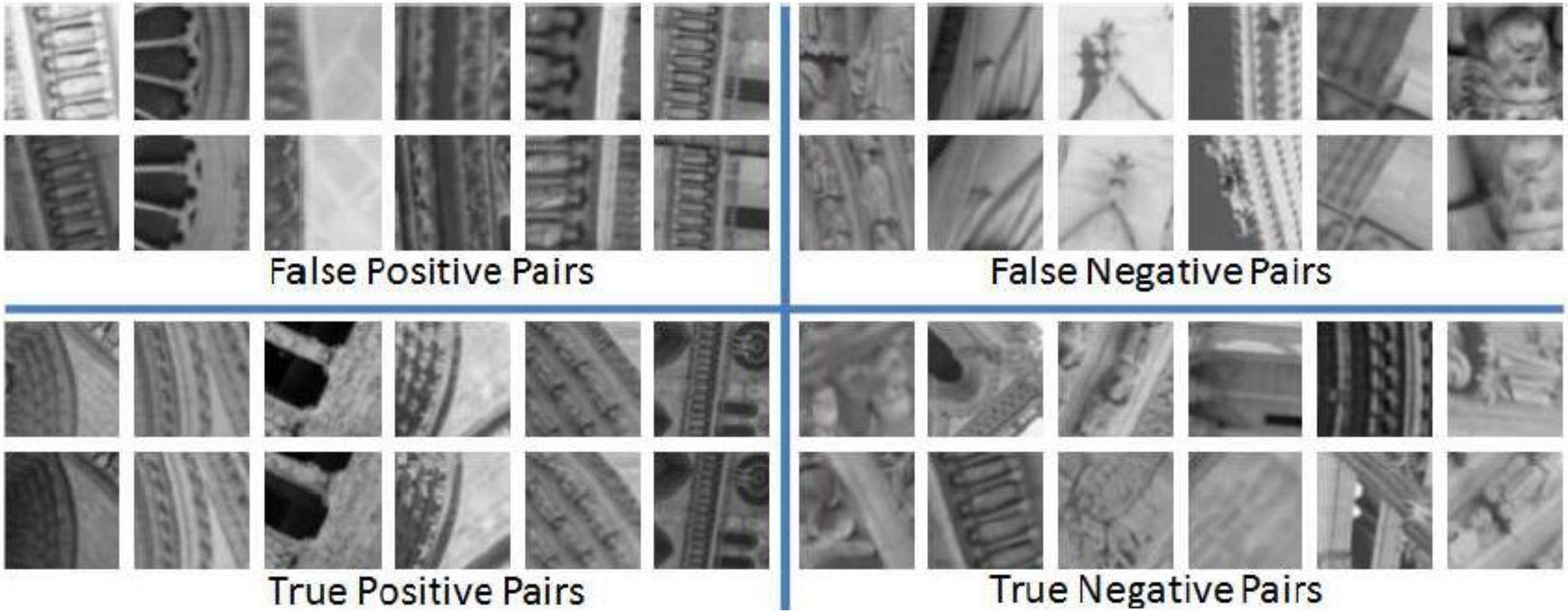
The most compact DAISY



- **T3-2nd-4-2r8s**
  - 13 bytes, 13.2%
- **T2-4-1r8s-PCA**
  - 7.5 bytes, 24.4%
- SIFT
  - 128 bytes, 26.1%

- 2 bits /dimension is sufficient before PCA
- 4 bits /dimension is needed after PCA
- http://www.cs.ubc.ca/~mbrown/patchdata/patchdata.html

Some of the errors made … …



- The world is repeating itself….

Summary

- Blob detection
  - Brief of Gaussian filter
  - Scale selection
  - Lapacian of Gaussian (LoG) detector
  - Difference of Gaussian (DoG) detector
  - Affine co-variant region
- Learning local descriptors
  - How can we get ground-truth data?
  - What is the form of the descriptor function?
  - What is the optimal criterion?
  - How do we optimize it?