Lecture 6

Correspondence and Planar transformations

Several slides taken from D. Lowe, M. Irani and T. Tuytelaars

Class Objectives

- Learn about scale-invariant and rotation-invariant features
  - SIFT
- Understand the hierarchy of different planar transformations
  - Euclidean
  - Similarity
  - Affine
  - Projective

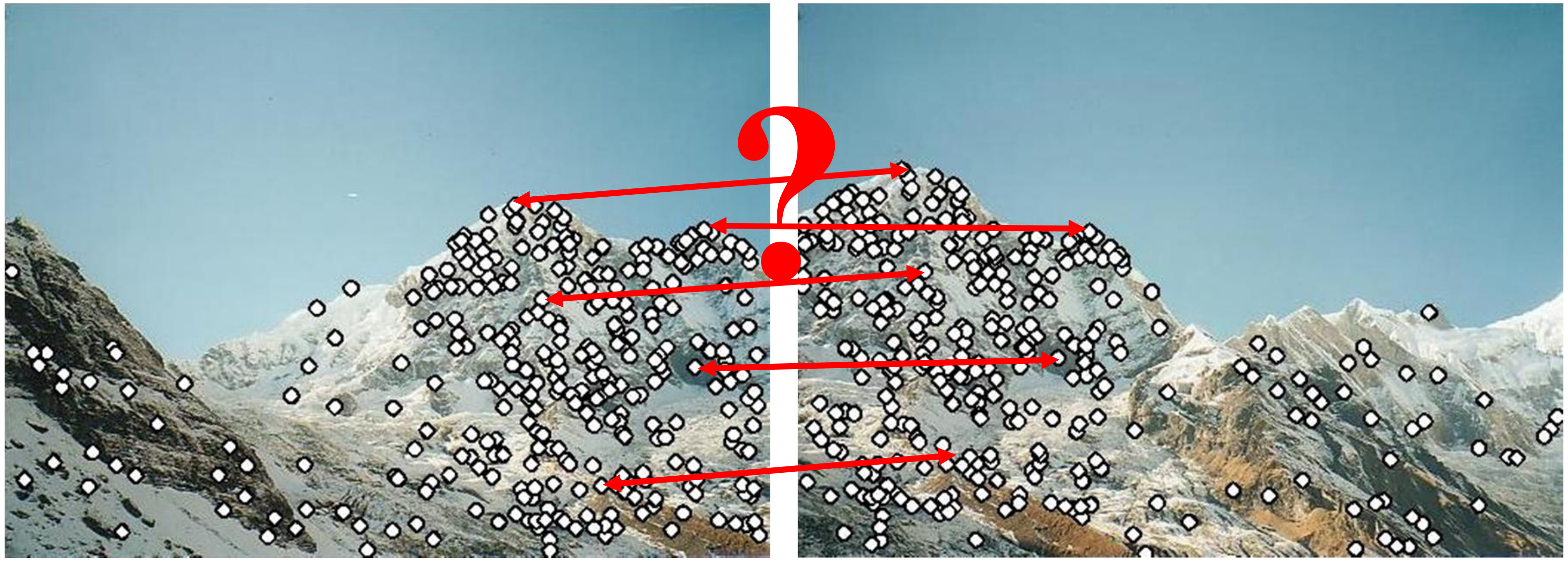
Outline

- Let’s take a deeper look at solving the correspondence problem
- What is the Scale Invariant Feature Transform (SIFT)?
- A hierarchy of transformations: Euclidean, Similarity, Affine, Projective
- Homography from a projective matrix

Our problem

- We know how to detect points (from last class)
- Next question:

How to match them?



Point descriptor should be:
1. Invariant
2. Distinctive

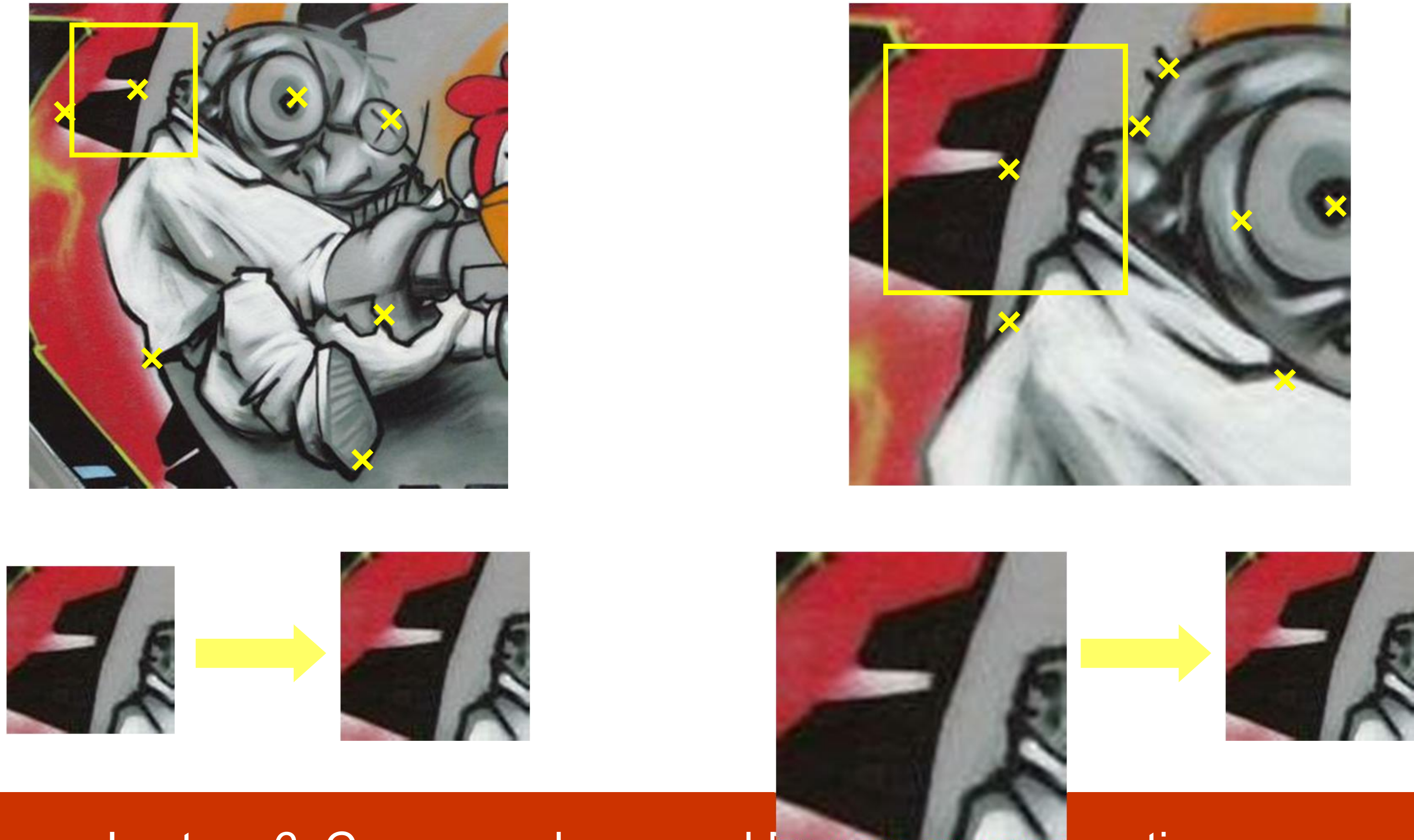
Finding correspondences

- SSD
- SAD
- Normalized cross-correlation

→ cannot deal with scale changes

Finding correspondences

- Normalize out effect of scale change



Finding correspondences

- SSD
- SAD
- Normalized cross-correlation

→ cannot deal with scale changes
→ cannot deal with rotations

→ are not robust to misalignments
→ pairwise distance measure, time consuming

Overview

- Last Lecture: detectors
- Today: descriptors dealing with scale/viewpoint changes

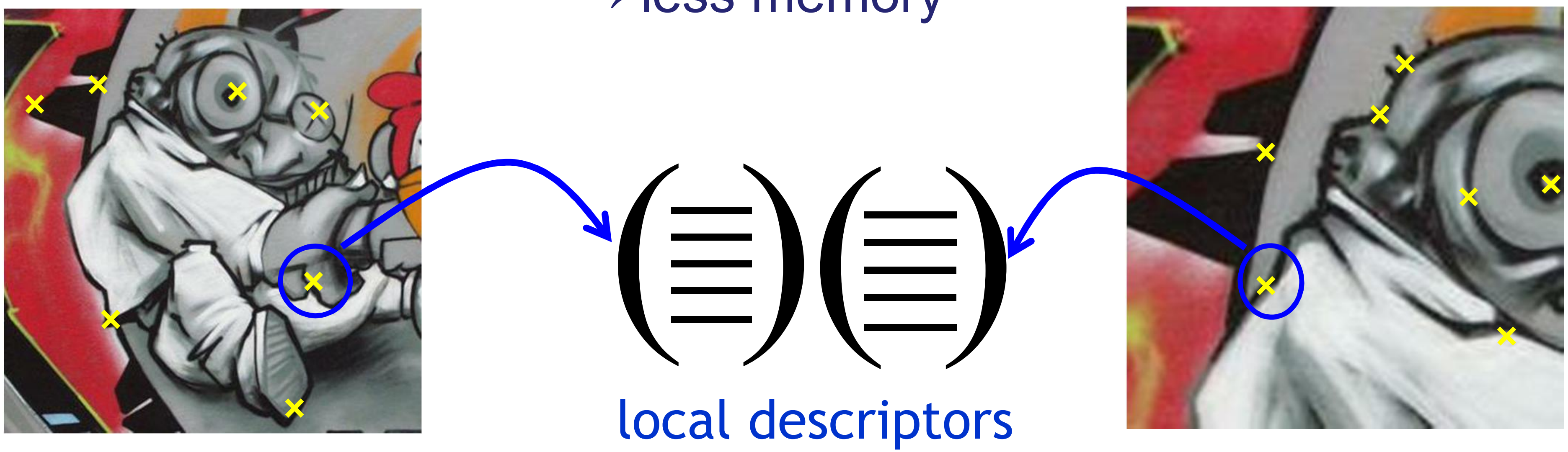
Why descriptors ?

Instead of directly comparing image patches, we take a two-step approach:

1. Extract descriptor
2. Find matches

→ more efficient

→ less memory



The ideal feature descriptor

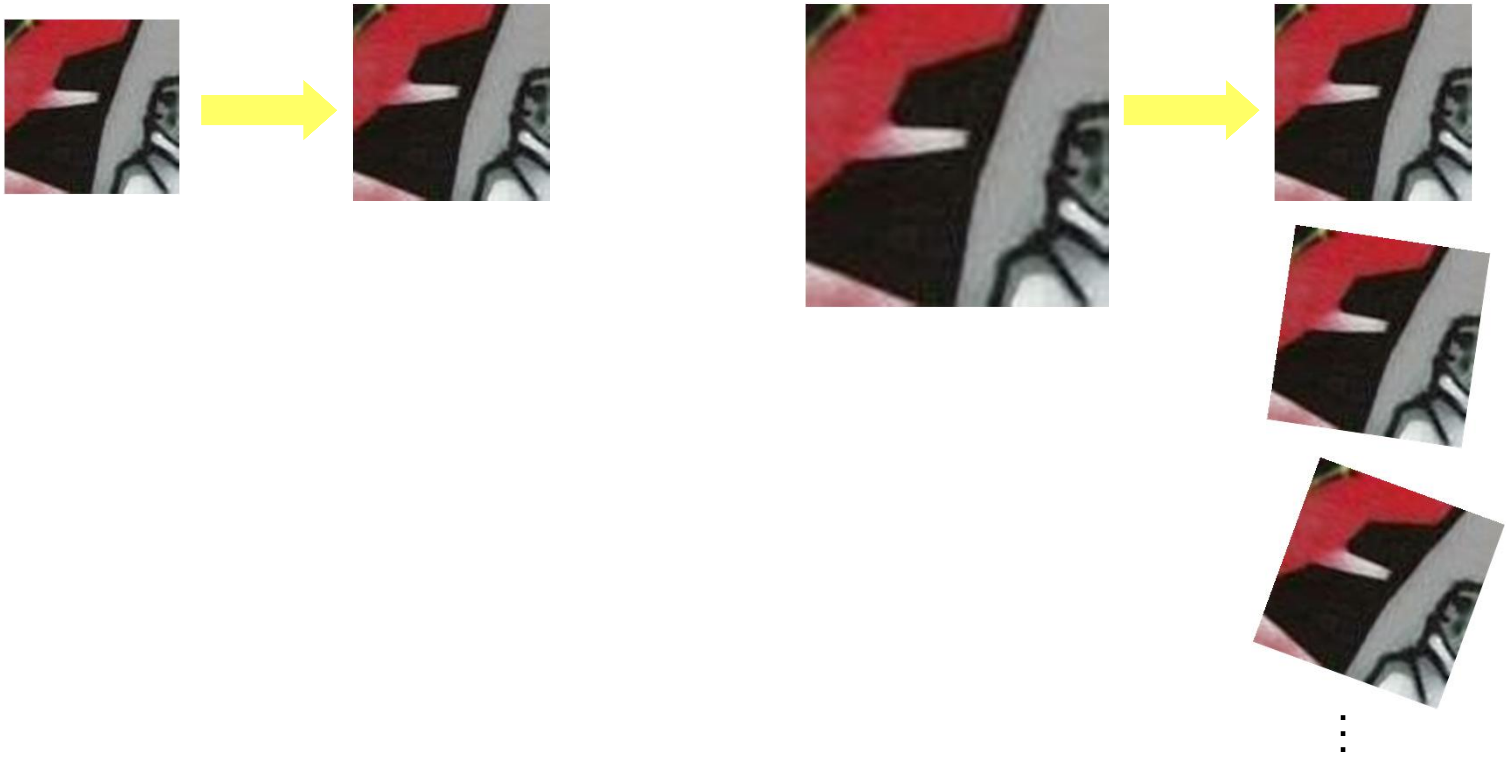
- Repeatable (invariant/robust)
- Distinctive
- Compact
- Efficient

Dealing with rotations

- Exhaustive search
- Invariance
- Robustness

Dealing with rotations

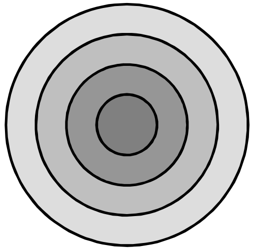
- Exhaustive search



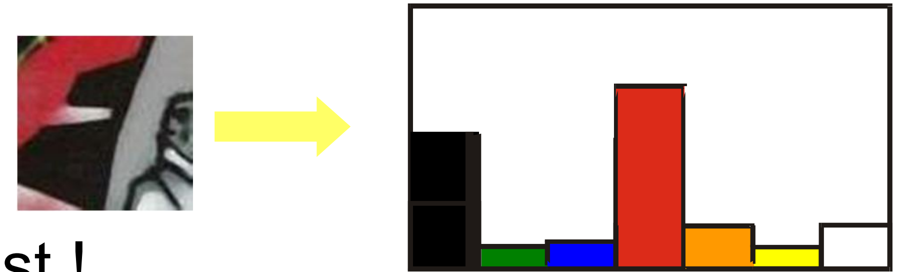
Dealing with rotations

- Exhaustive search
- Invariance
  - Integration
  - Dominant orientation selection
- Robustness
  - SIFT

Invariance by integration



- Integrate out unknown parameter
- Example: moment invariants
- Example: color histograms

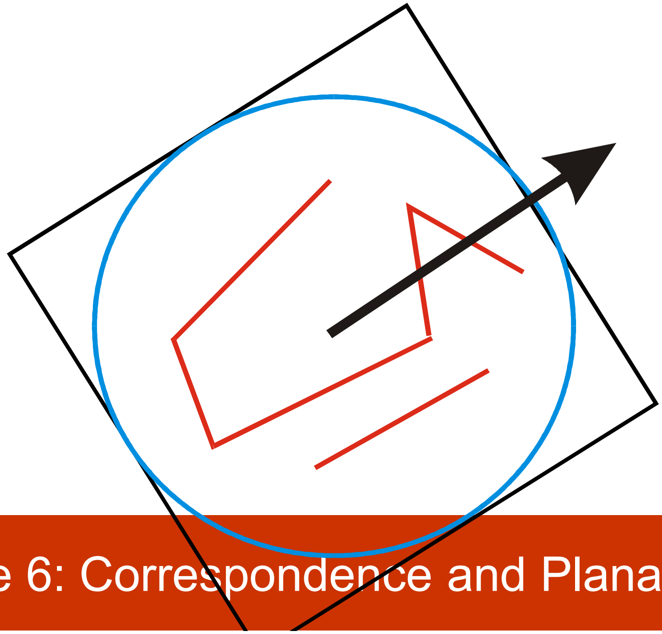


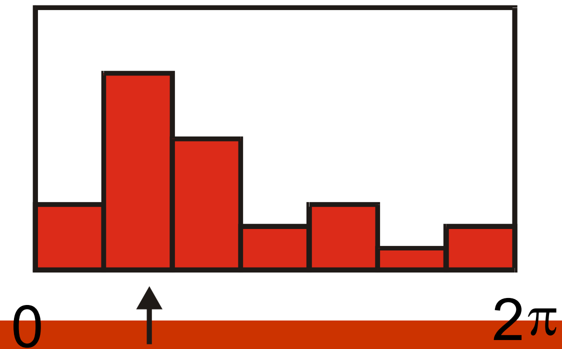
- Robust !

Invariance by parameter selection

Dominant orientation selection (D. Lowe)

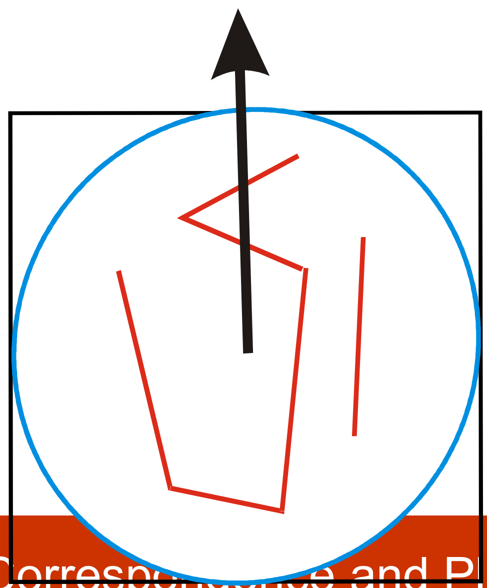
- Compute image gradients
- Build orientation histogram
- Find maximum

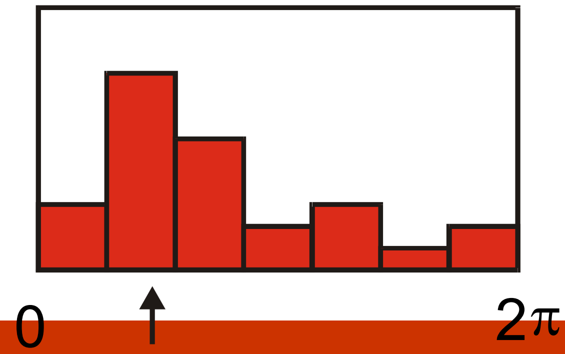




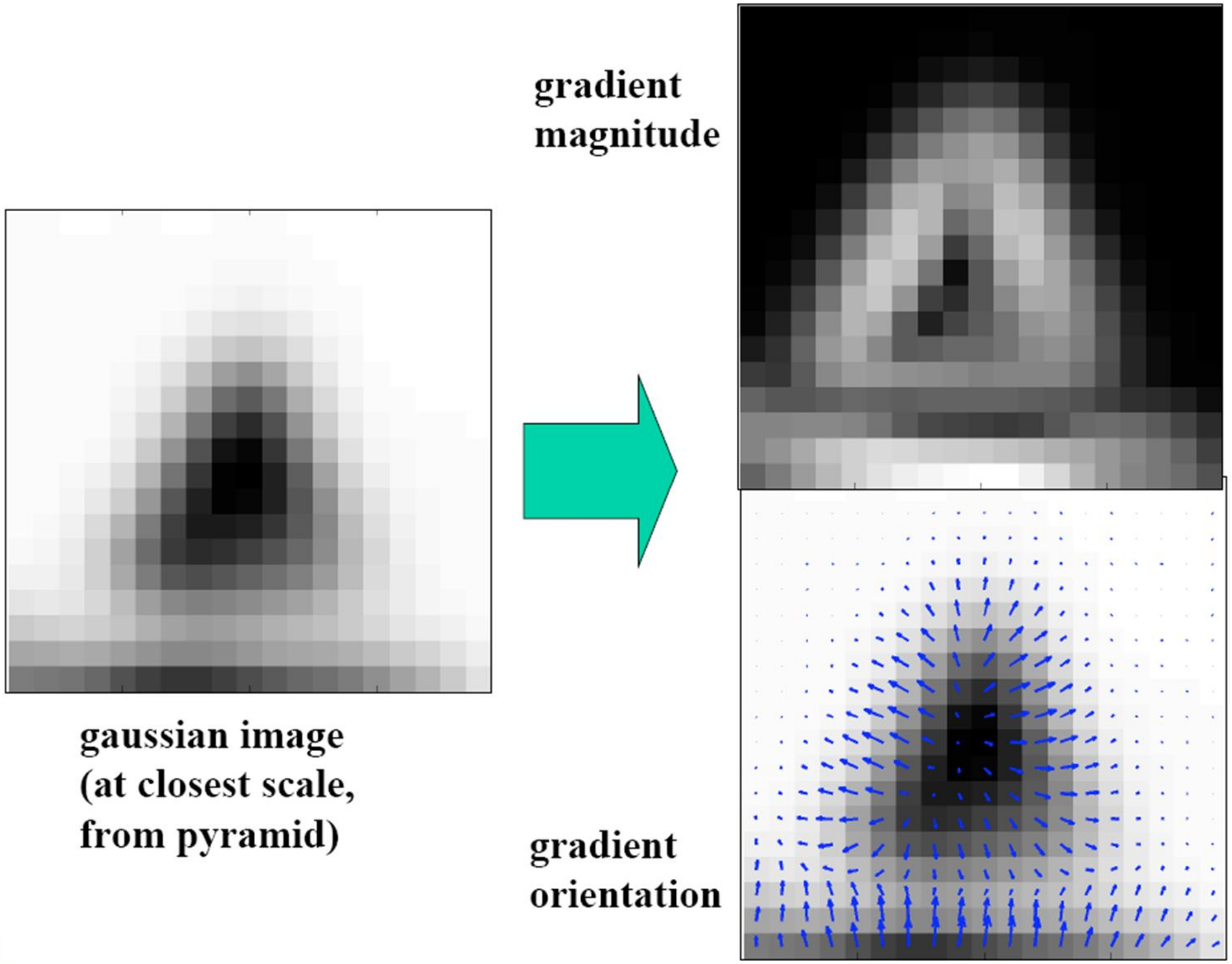
Invariance by parameter selection

- Dominant orientation selection (D. Lowe)
  - Compute image gradients
  - Build orientation histogram
  - Find maximum

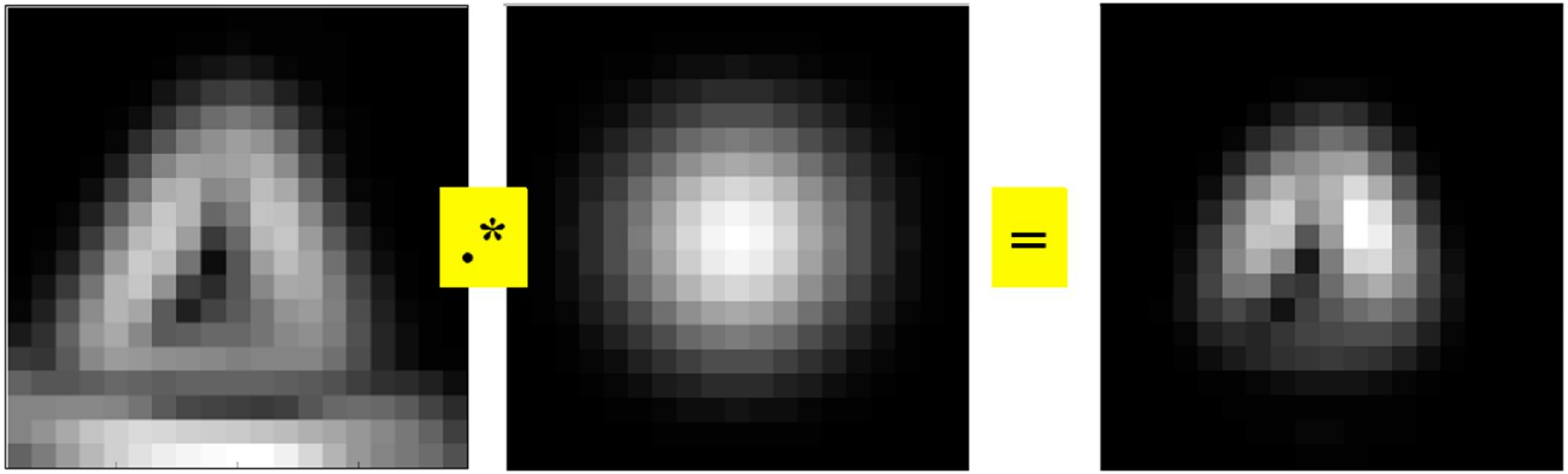




Invariance by parameter selection



Invariance by parameter selection



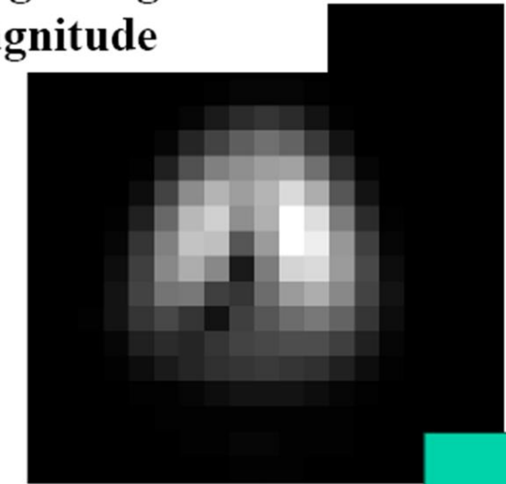
gradient magnitude

weighted by 2D gaussian kernel

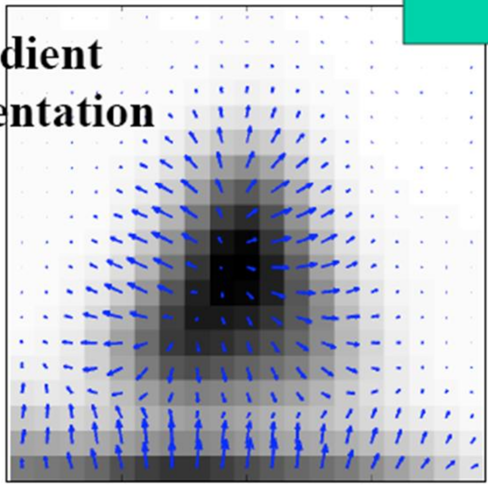
weighted gradient magnitude

Invariance by parameter selection

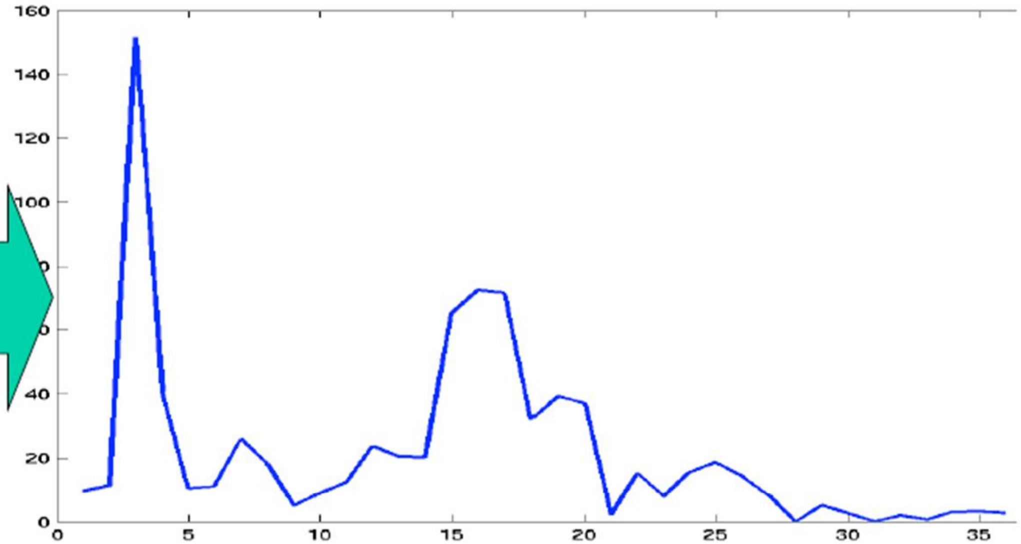
weighted gradient magnitude



gradient orientation



weighted orientation histogram.
Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.

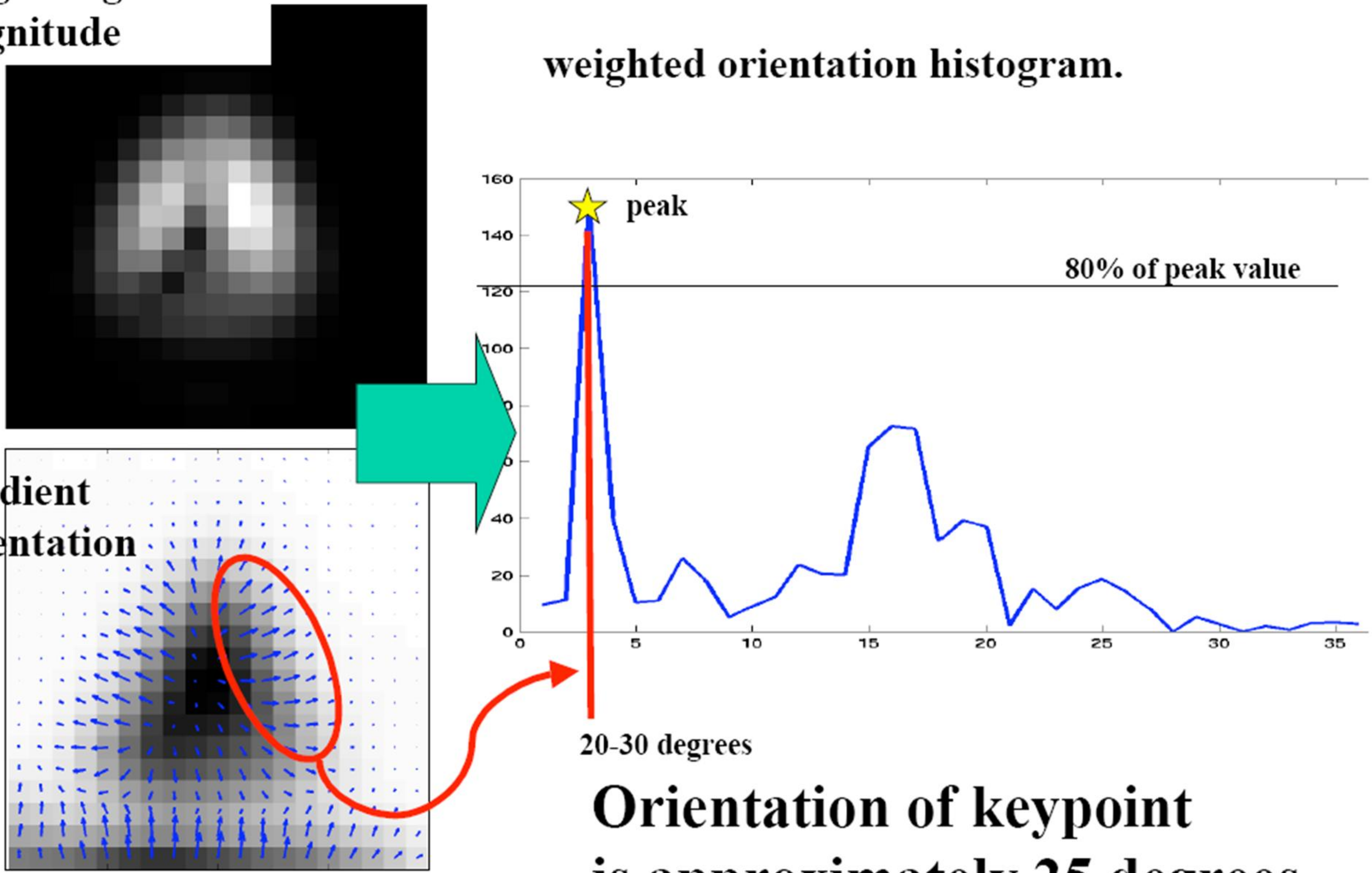


36 buckets
10 degree range of angles in each bucket, i.e.
- **0 <=ang<10 : bucket 1**
- **10<=ang<20 : bucket 2**
- **20<=ang<30 : bucket 3 …**

Invariance by parameter selection

weighted gradient magnitude

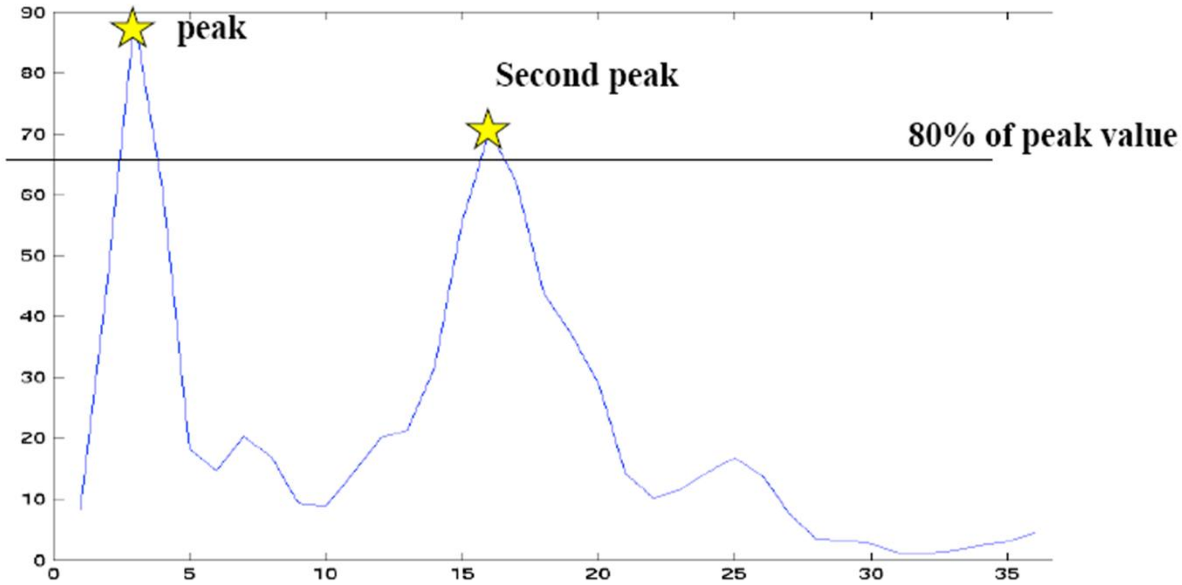
weighted orientation histogram.



Orientation of keypoint is approximately 25 degrees

Invariance by parameter

There may be multiple orientations.



In this case, generate duplicate keypoints, one with orientation at 25 degrees, one at 155 degrees.

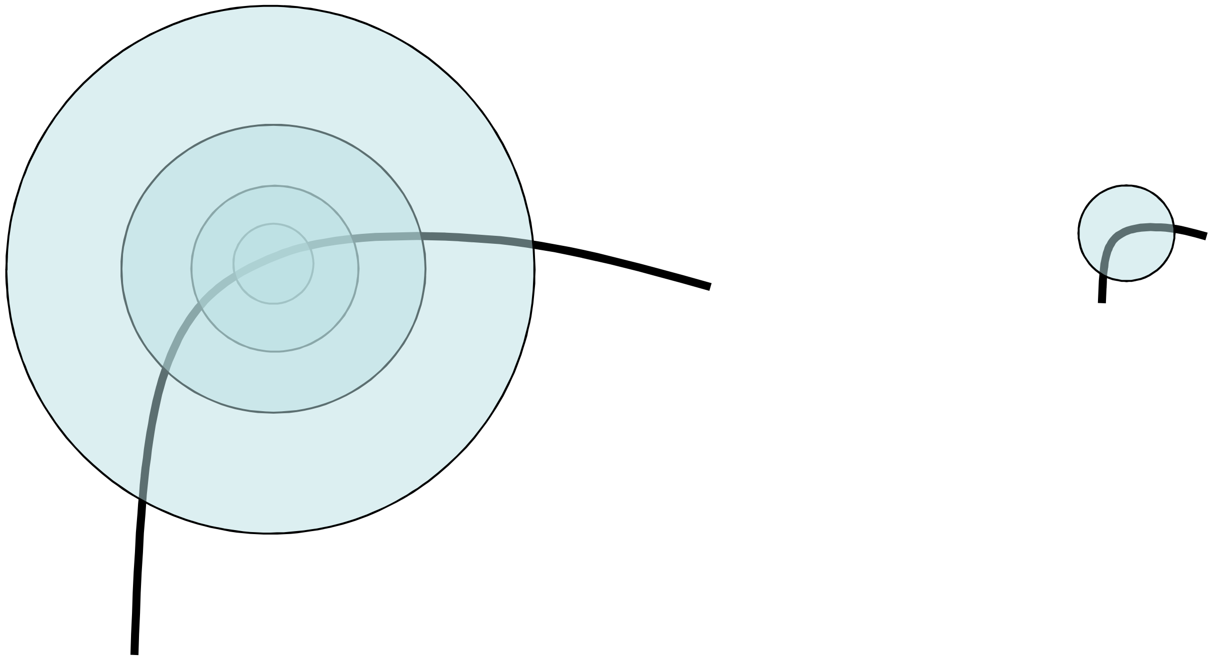
Design decision: you may want to limit number of possible multiple peaks to two.

Invariance by parameter selection

- After normalization and dominant orientation selection, the regions are perfectly aligned and almost *any* measurement could be used as an invariant descriptor.

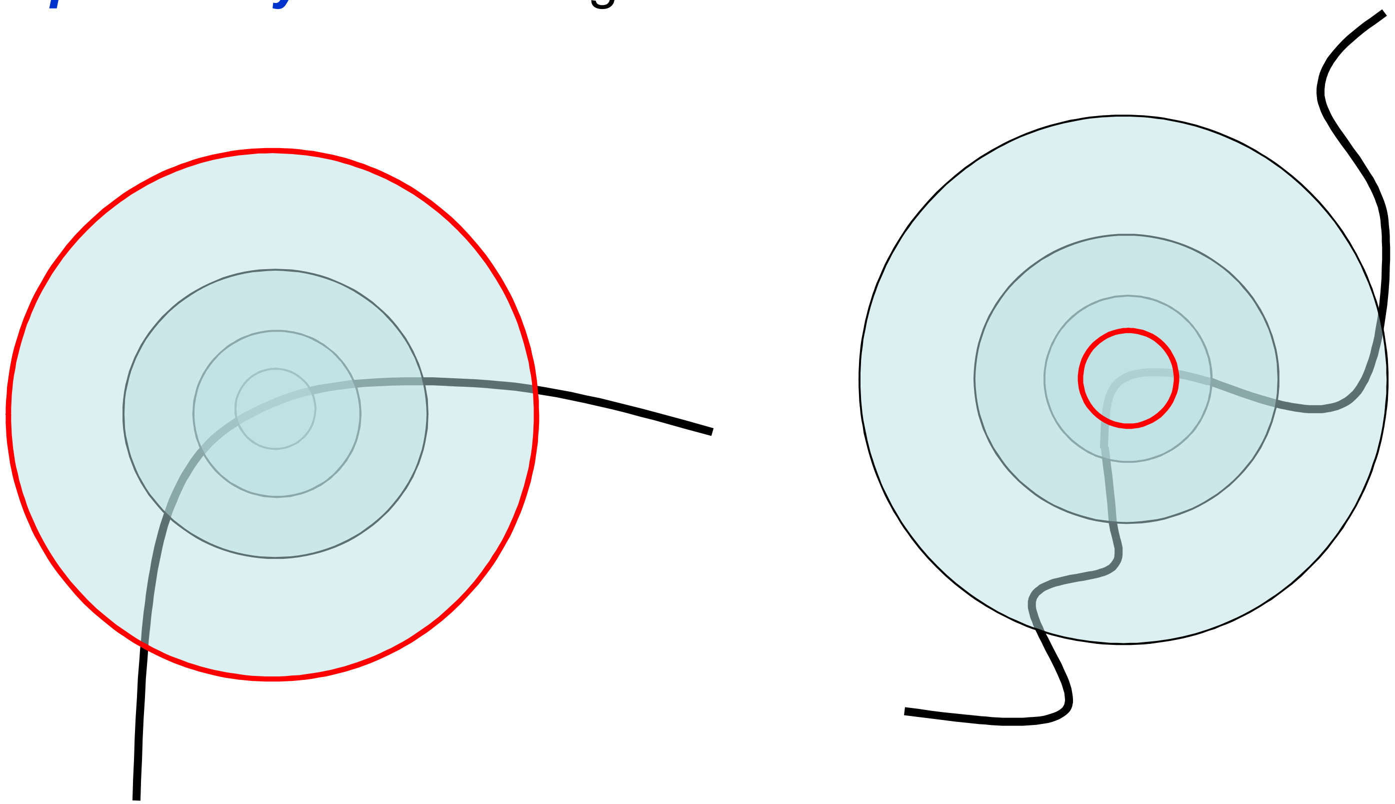
Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale Invariant Detection

- The problem: how do we choose corresponding circles ***independently*** in each image?

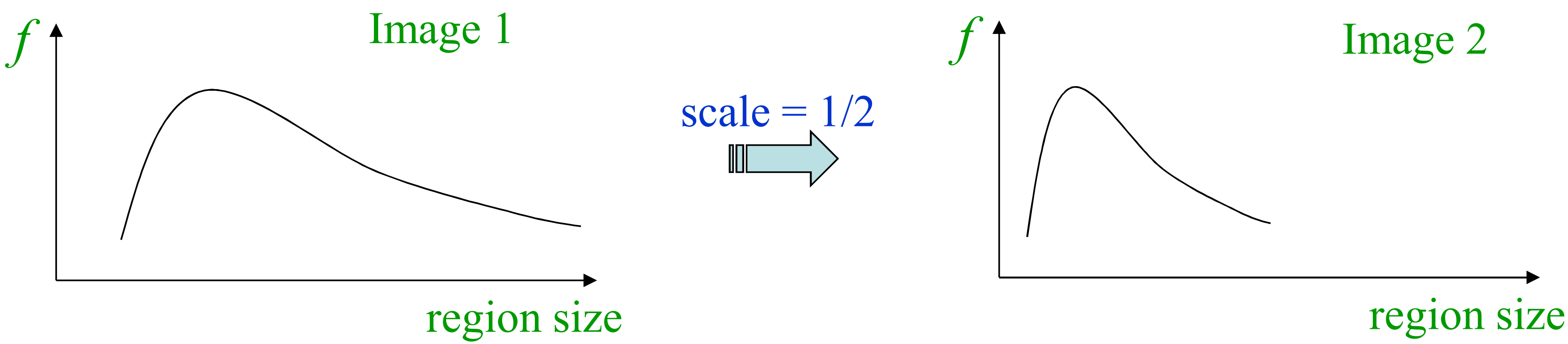


Scale Invariant Detection

- Solution:
  - Design a function on the region (circle), 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 (circle radius)



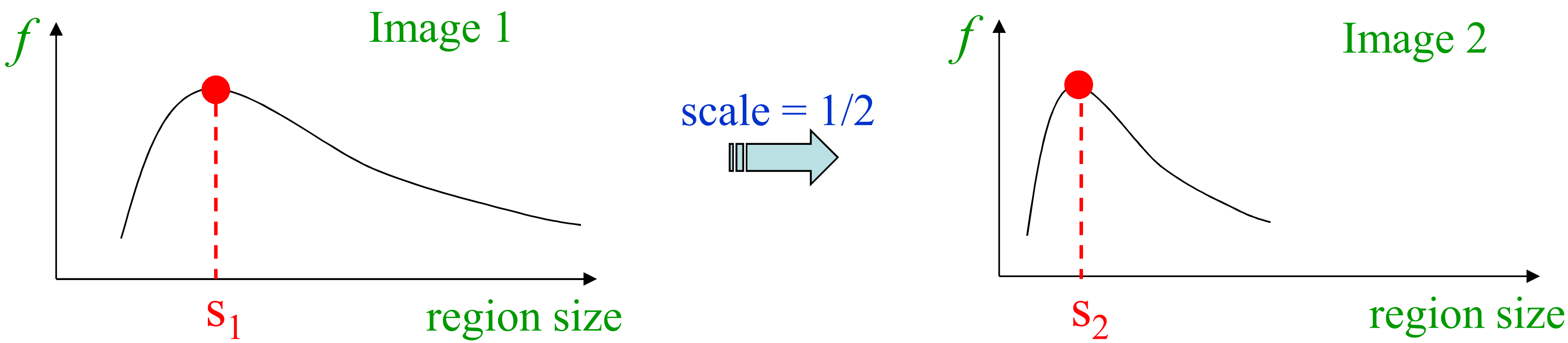
Scale Invariant Detection

- Common approach:

  Take a local maximum of this function

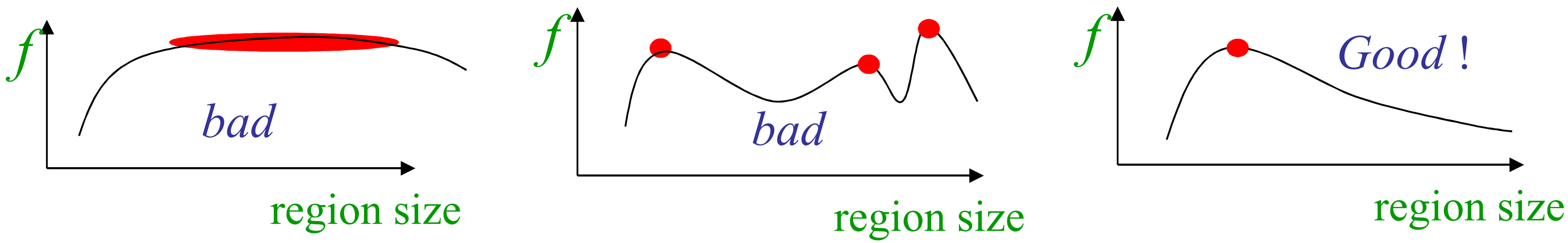
  Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

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



Scale Invariant Detection

- A "good" function for scale detection:
  has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

Scale Invariant Detection

- Functions for determining scale

$$f = \mathrm{Kernel} * \mathrm{Image}$$

Kernels:

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

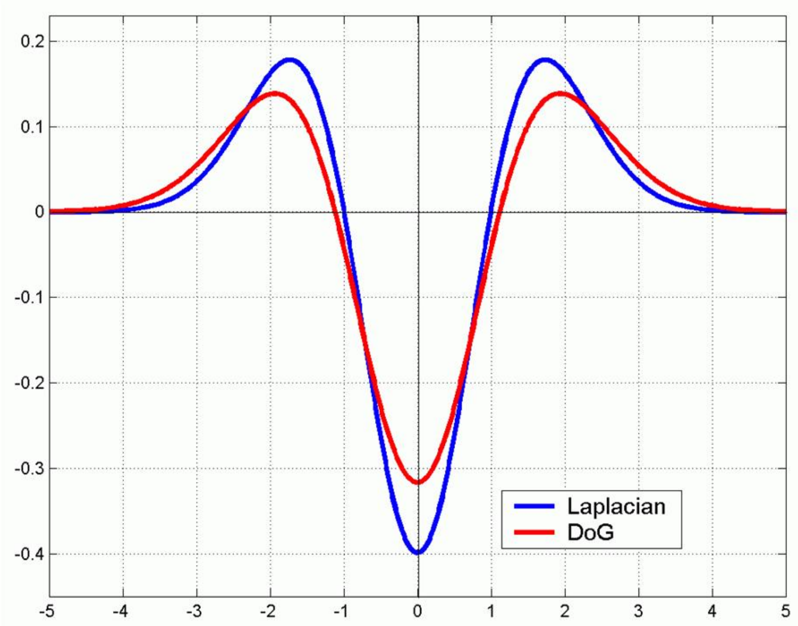
(Laplacian)

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

(Difference of Gaussians)

where Gaussian

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



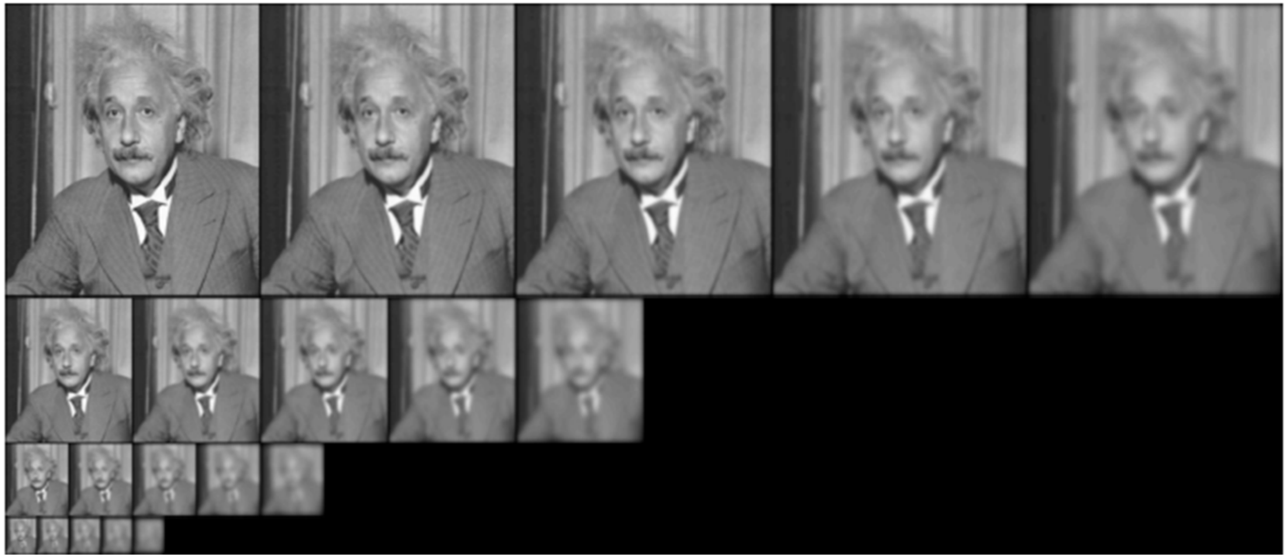
Note: both kernels are invariant to *scale* and *rotation*

Finding correspondences with SIFT

- Stands for Scale Invariant Feature Transform
- The idea:
  - Detection of image features
    - Invariant to scale and rotation
    - Partially invariant to change of illumination and change of 3D viewpoint
  - Describe those features, so that we can match them…

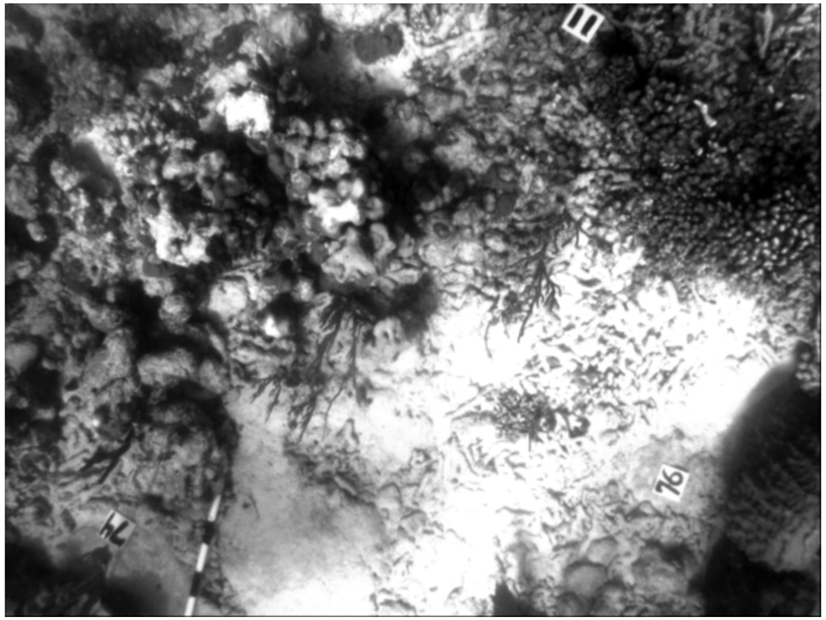
SIFT: Scale extrema detection

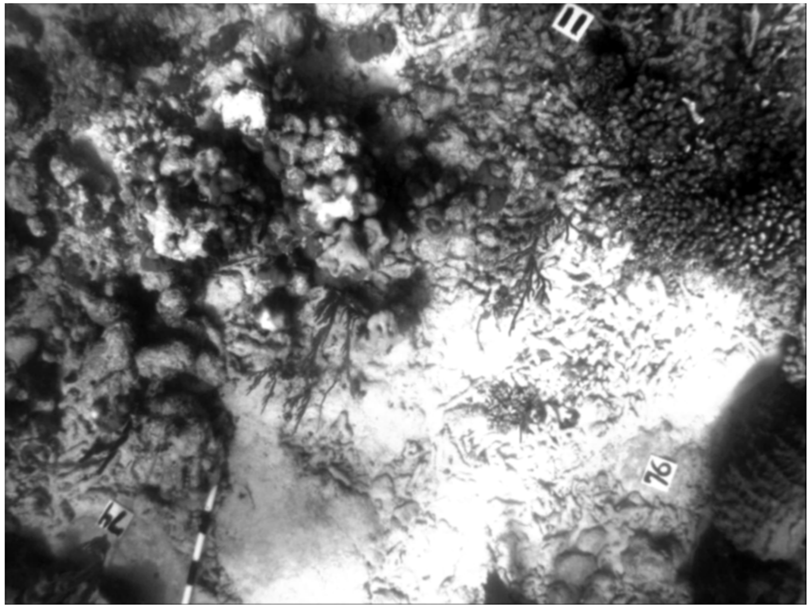
- Example of scale space representation pyramid using a Gaussian function.
  - The original image is convolved with incremental Gaussian to produce images separated by a constant value



SIFT: Gaussian pyramid

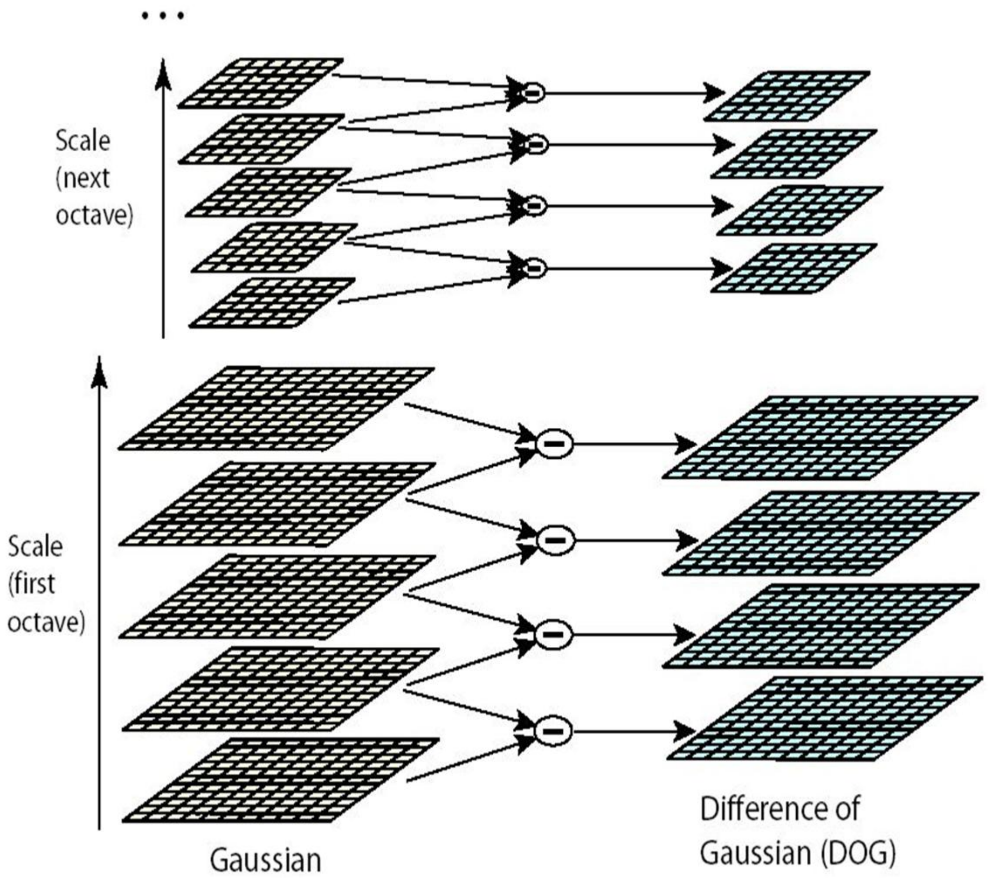
- **Example of scale space representation pyramid using a Gaussian function.**





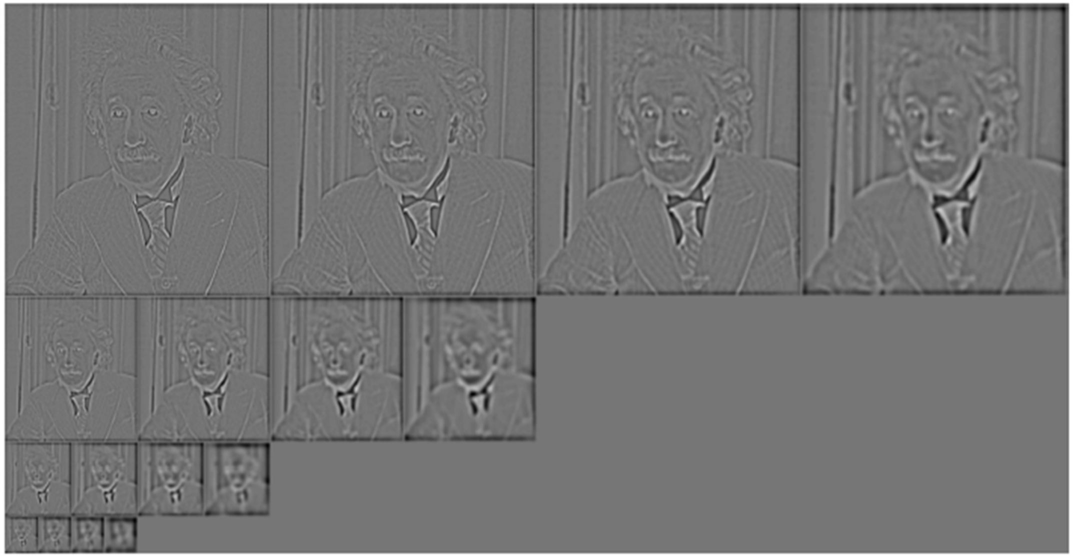
Scale extrema detection

- **Difference of Gaussian**

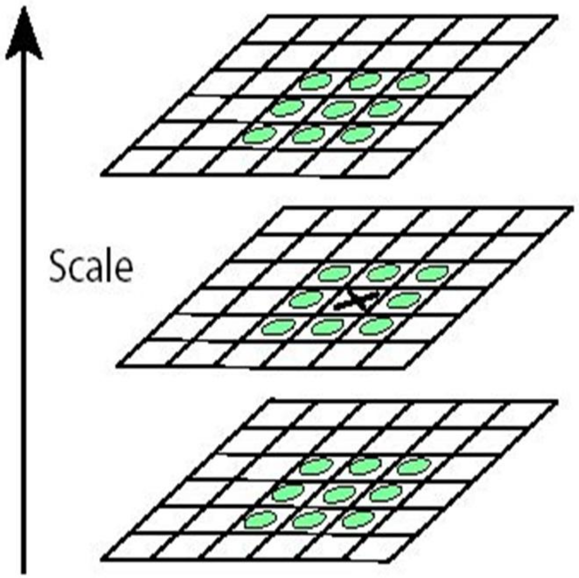


Scale extrema detection

- **Example of Difference Of Gaussian (DOF)**



Scale extrema detection

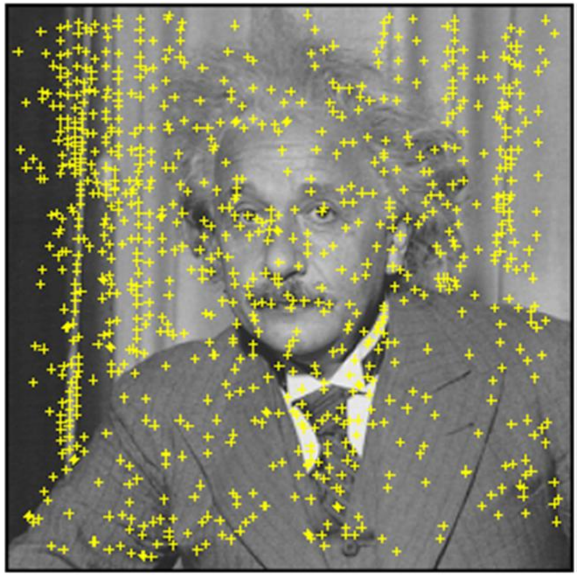


- **A point is selected as candidate if it is smaller or greater than its 26 neighbors**

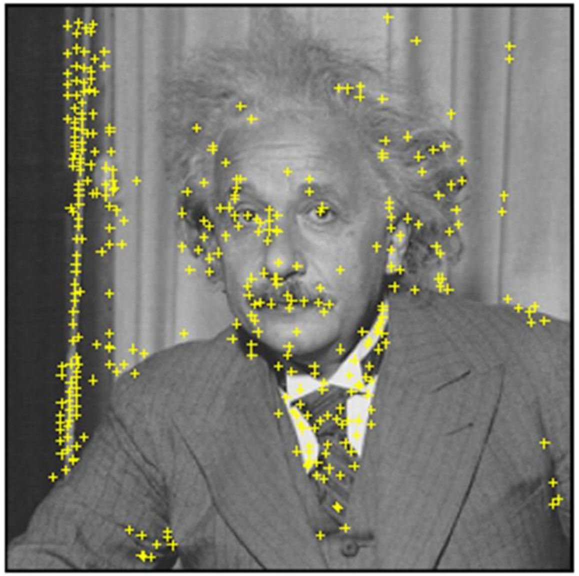
SIFT: Keypoint localization

- Interpolation of nearby data is used to accurately determine position of candidate points
- Unstable Extrema, that have **low contrast**, are discarded
- Candidate points that belong to **edges** are not well localized along the edge. This makes them very unstable to small amounts of noise and therefore this type of points will be discarded too.

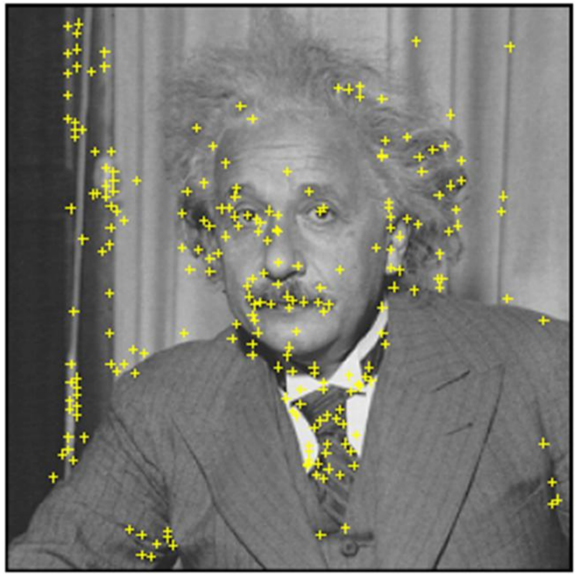
SIFT: Keypoint localization



Candidate points



Without low contrast



Without edge points

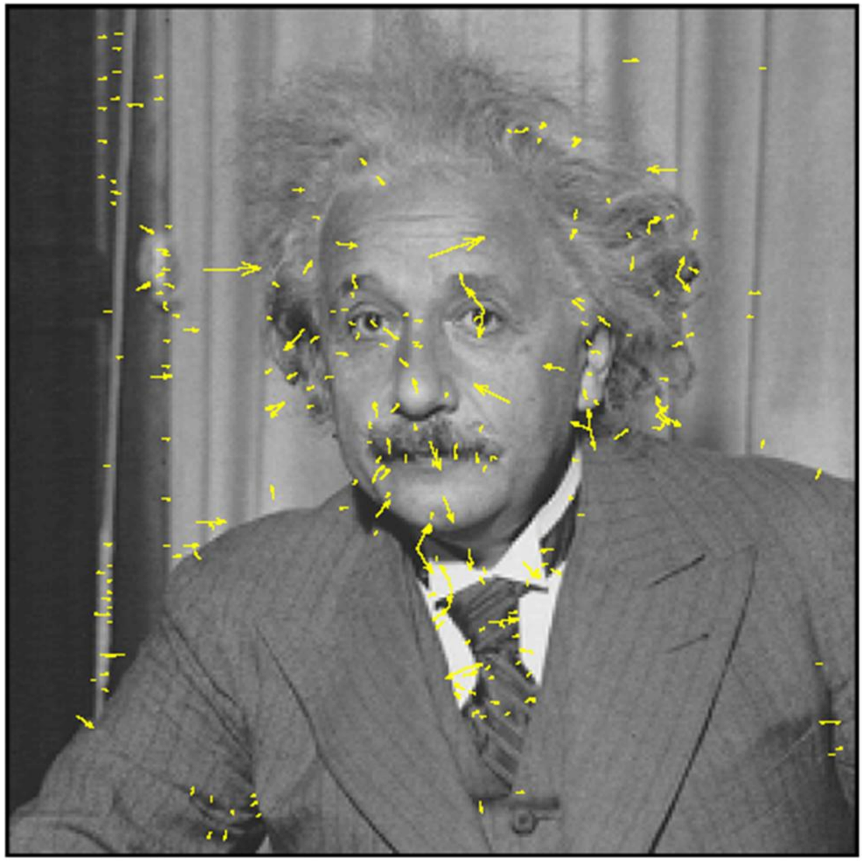
SIFT: Orientation assignment

- **The gradient orientation and magnitude is computed for all pixels of all scales.**
- **A gradient orientation histogram in the keypoints neighborhood at the closest scale is performed.**
- **The contribution of each neighboring pixel is weighted by the gradient magnitude and a Gaussian window with a sigma that is 1.5 times the sigma of the keypoint scale**
- **Peaks in the histogram correspond to dominant orientations. A separate keypoint is created for the direction corresponding to the histogram maximum, and any other direction within 80% of the maximum value.**

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

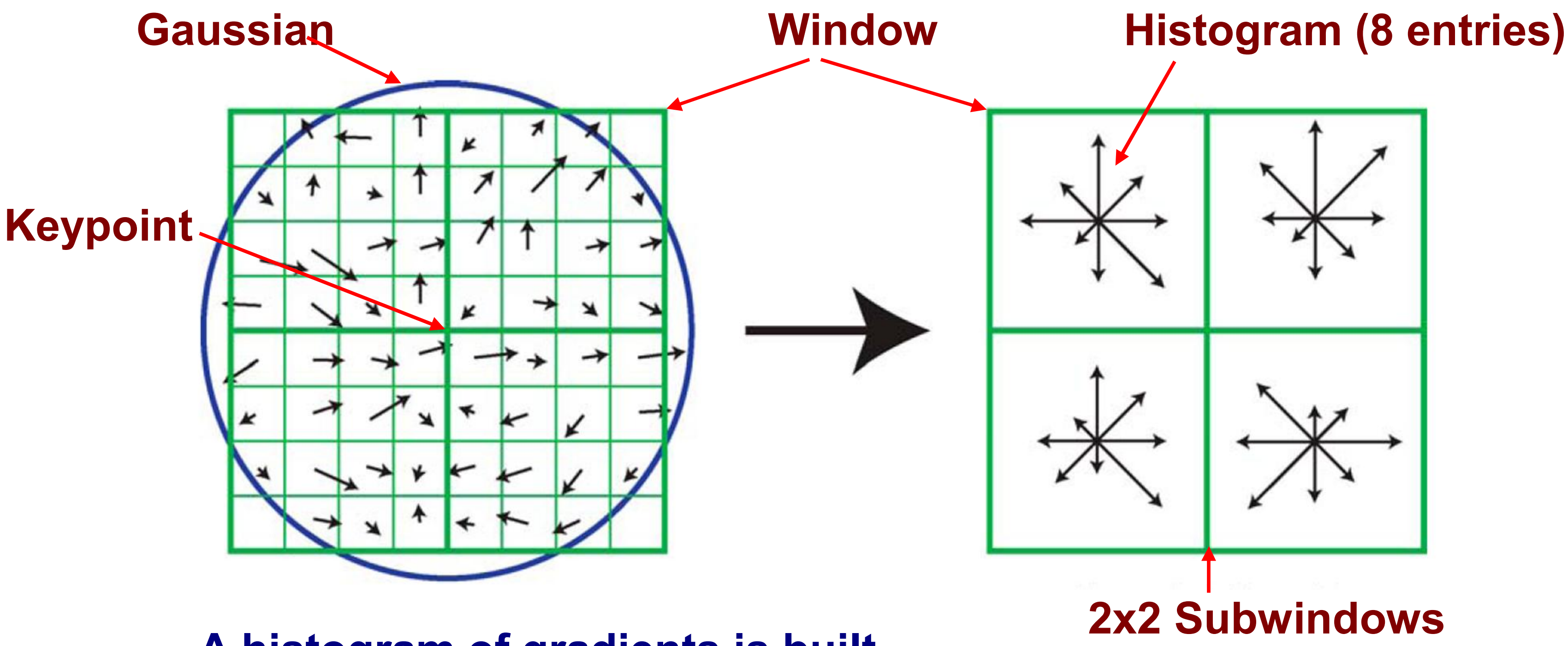
SIFT: Orientation assignment



Keypoint descriptor

- We want to compute a local image descriptor that is invariant to the remaining variations (e.g. illumination, viewpoint).
- Image gradient magnitudes and orientations at all levels of the pyramid have been precomputed before.
- <u>SCALE INVARIANCE:</u> The gradients are sampled in a window around each keypoint with respect to the scale of the keypoint.
- <u>ORIENTATION INVARIANCE:</u> The gradients are rotated relative to the keypoint orientation.
- <u>AVOID SUDDEN CHANGES:</u> Gradient magnitudes are weighted with a Gaussian function located at the center of the window and with σ = Window_Size / 2.

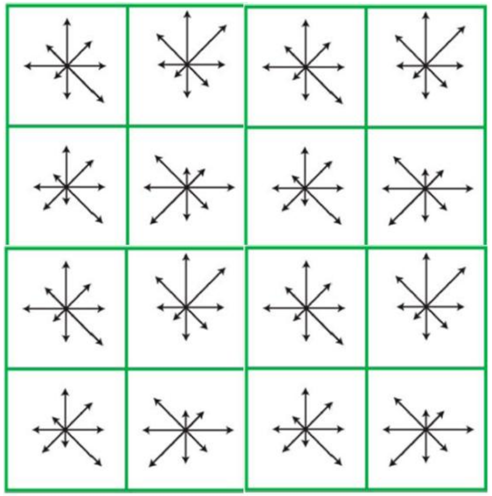
SIFT: Keypoint descriptor



- **A histogram of gradients is built**
  - **Each entry is calculated as the sum of all gradient magnitudes from the corresponding subwindow, whose orientations agree with the direction of the entry.**

Descriptor

The feature descriptor is a VECTOR that contains the values of the gradient orientation histogram entries



In Lowe's article:

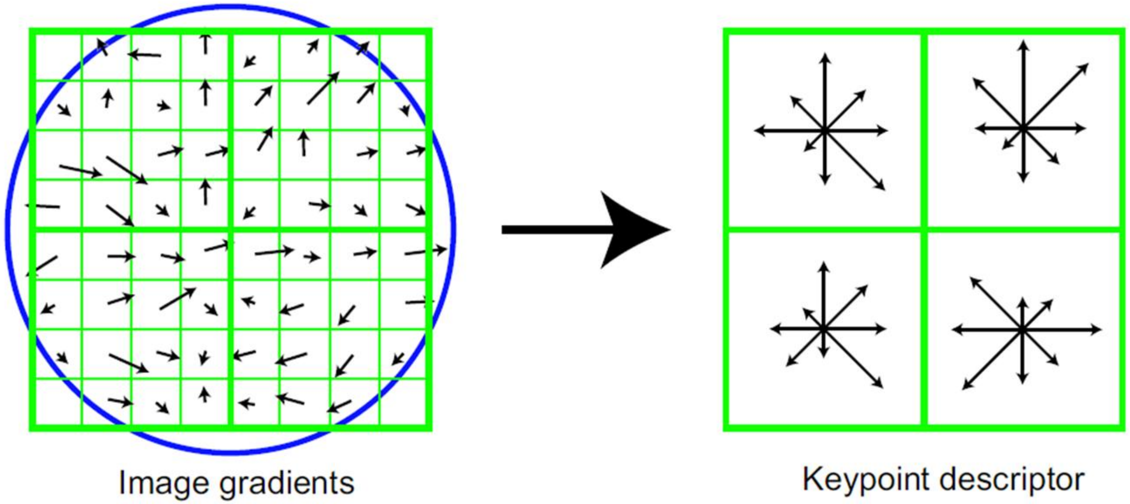
- Window of 16x16 pixels.
- Region divided into 4x4 subwindows
- Gradient orientation is discretized to angles of 45º → histogram has 8 entries
- **Descriptor** = 4 x 4 x 8 = 128 elements

Keypoint descriptor

- ILLUMINATION INVARIANCE
  - Change in contrast means a multiplication of gradients by a constant → cancelled by a vector normalization.
  - Change in brightness means an addition of a constant → gradient is not affected.
  - Non linear illumination can influence the magnitude of certain gradients, but not the orientation.
    - Threshold the descriptor vector with 0.2
    - Renormalization

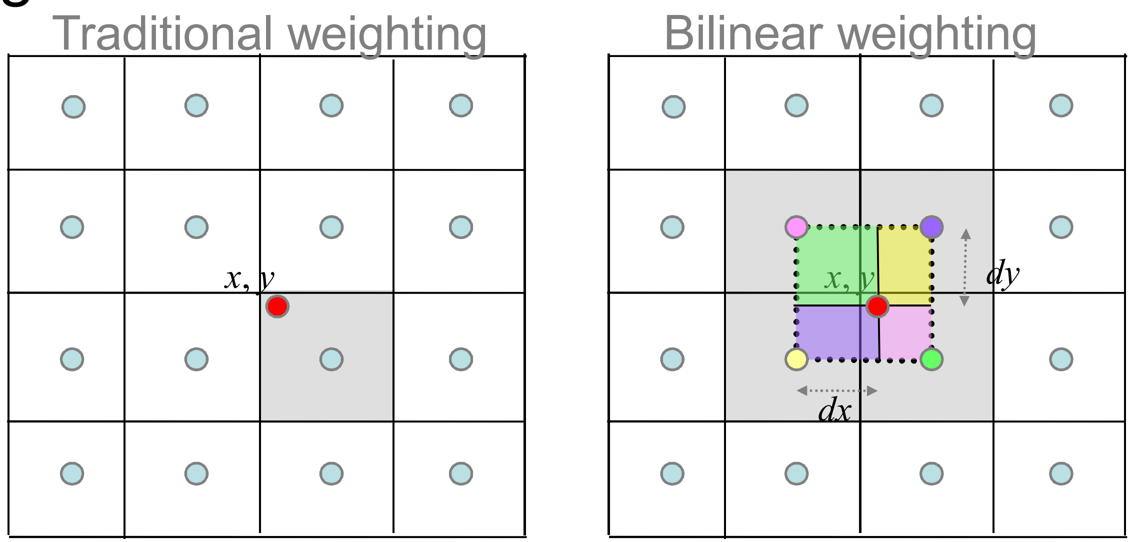
Robustness of SIFT

- Use of histograms
- Weighting with gradient magnitude
- Smoothing
- Bilinear weighting



Robustness of SIFT

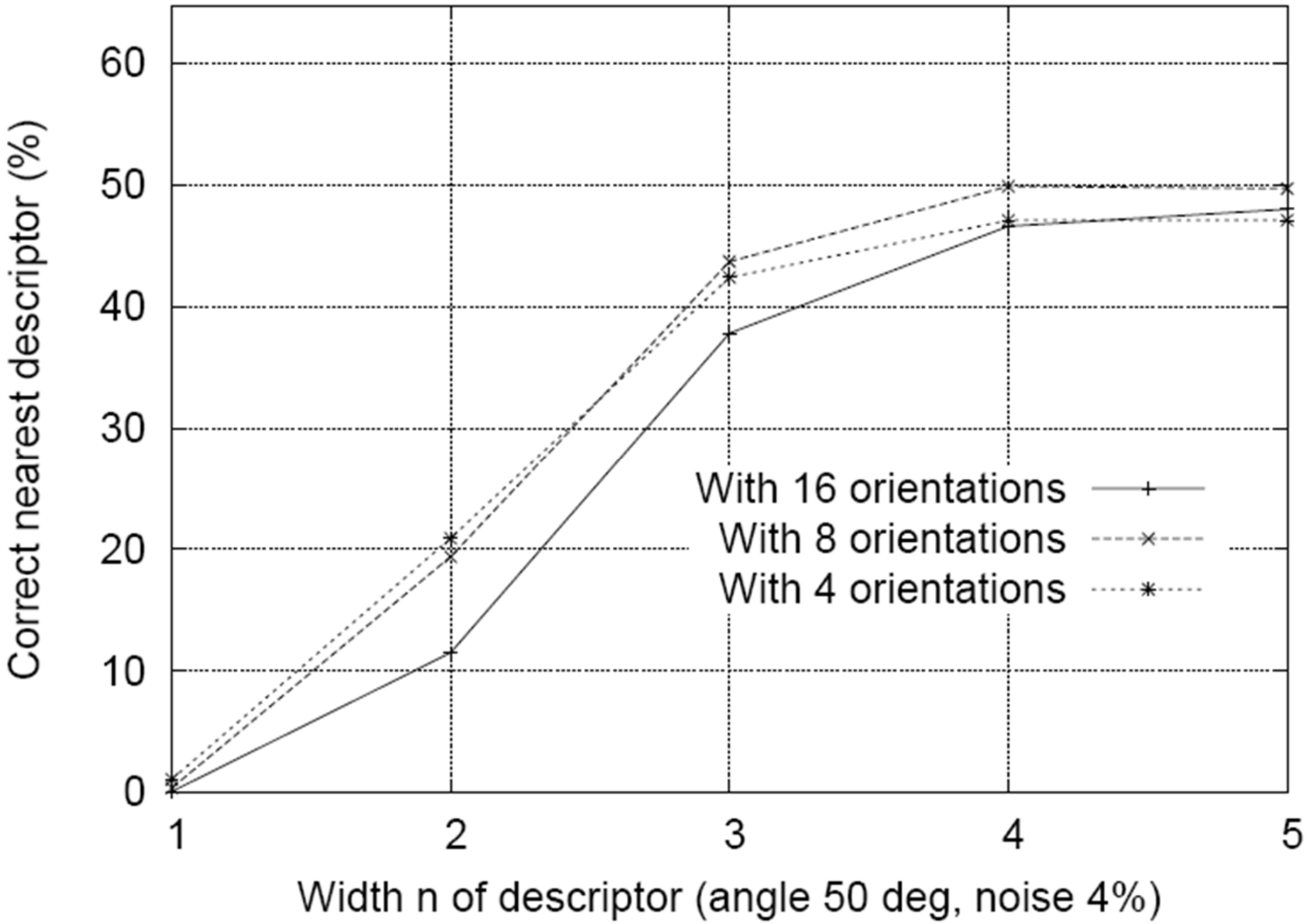
- Use of histograms
- Weighting with gradient magnitude
- Smoothing
- Bilinear weighting



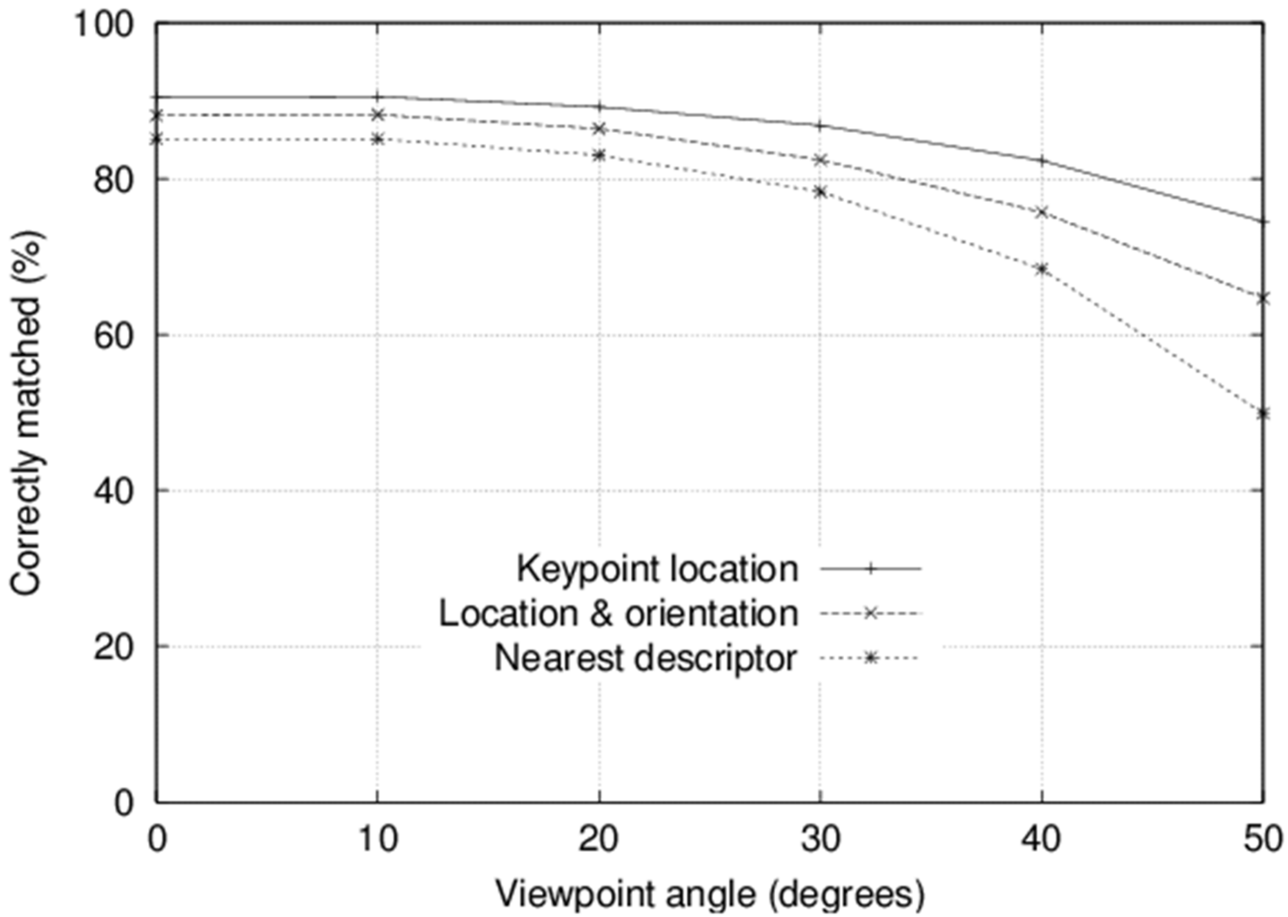
Robustness of SIFT

- Use of histograms
- Weighting with gradient magnitude
- Smoothing
- Bilinear weighting
- Normalization
    1. Normalize
    2. Clip values higher than 0.2
    3. Renormalize

Why 4x4x8?


Correct nearest descriptor (%)
60
50
40
30
20
10
0
1
2
3
4
5
Width n of descriptor (angle 50 deg, noise 4%)
With 16 orientations
With 8 orientations
With 4 orientations

Sensitivity to affine change



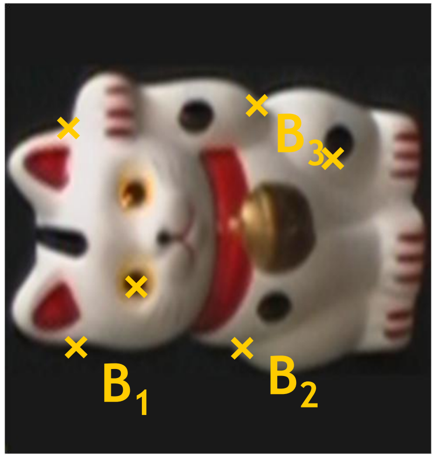
Final remarks on descriptors

- Measurement region can be chosen different from distinguished region
- Level of invariance of descriptor should match that of detector (or less)

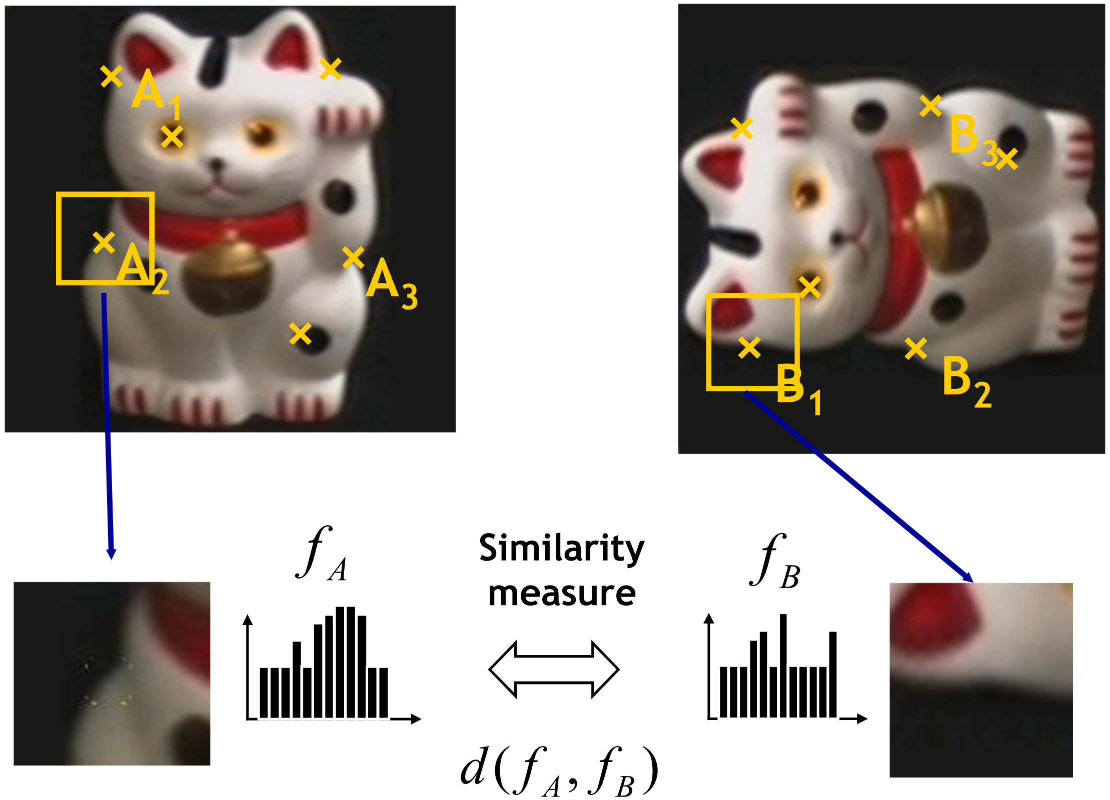
Distance measures

- SIFT: uses Euclidean distance as matching strategy

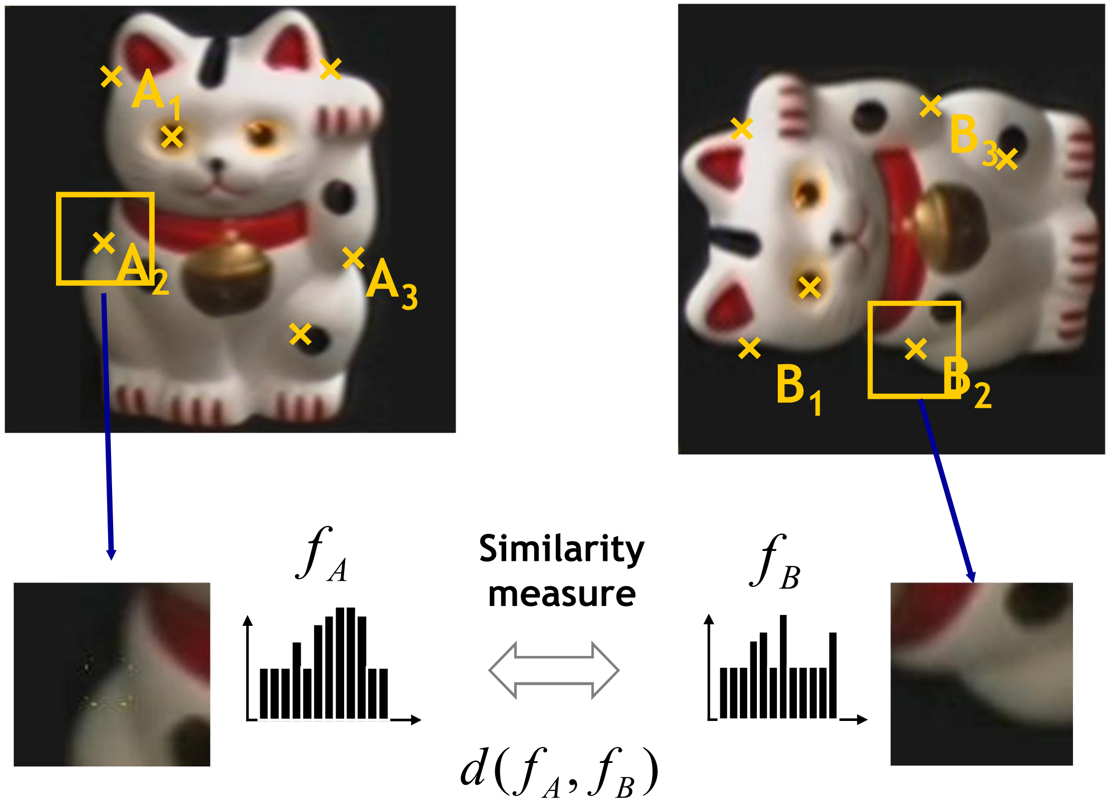
Matching strategies



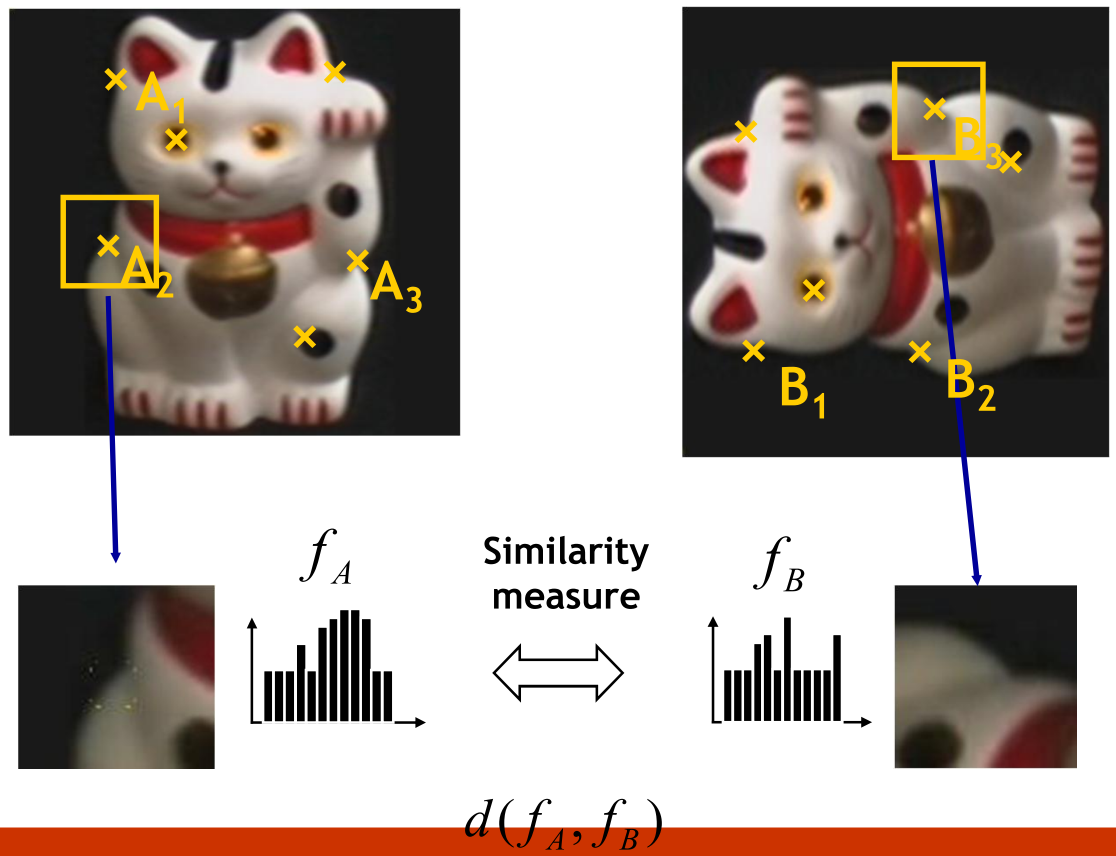
Matching strategies



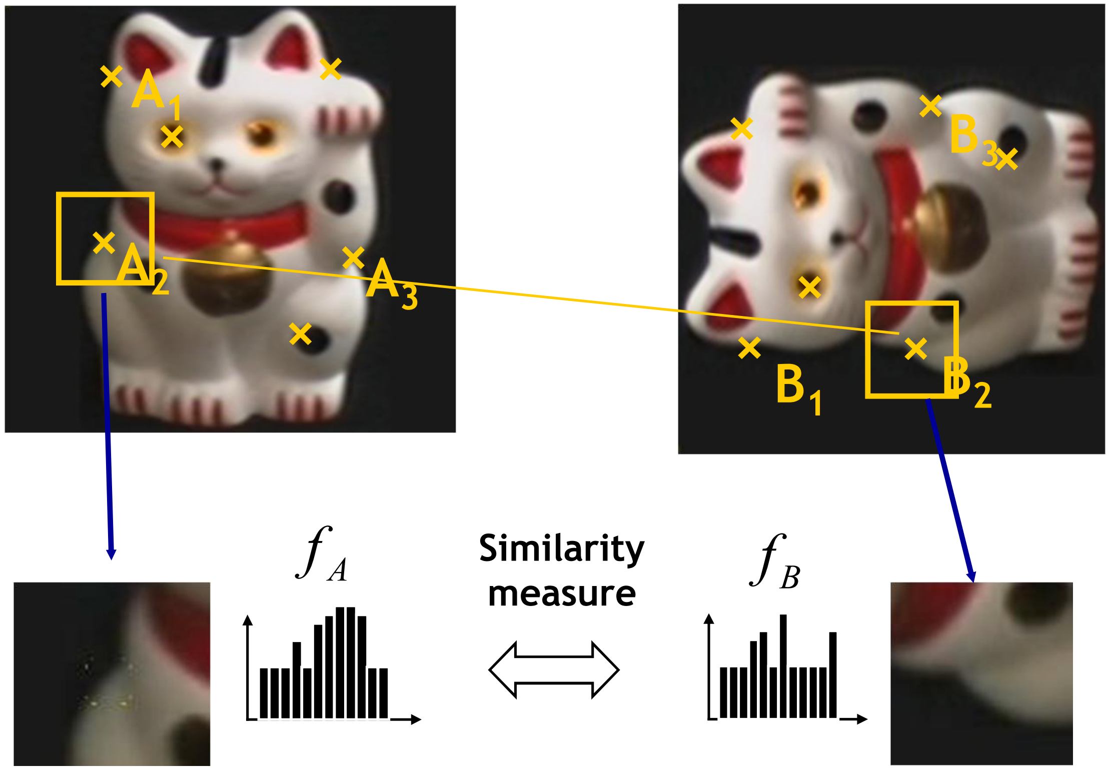
Matching strategies



Matching strategies



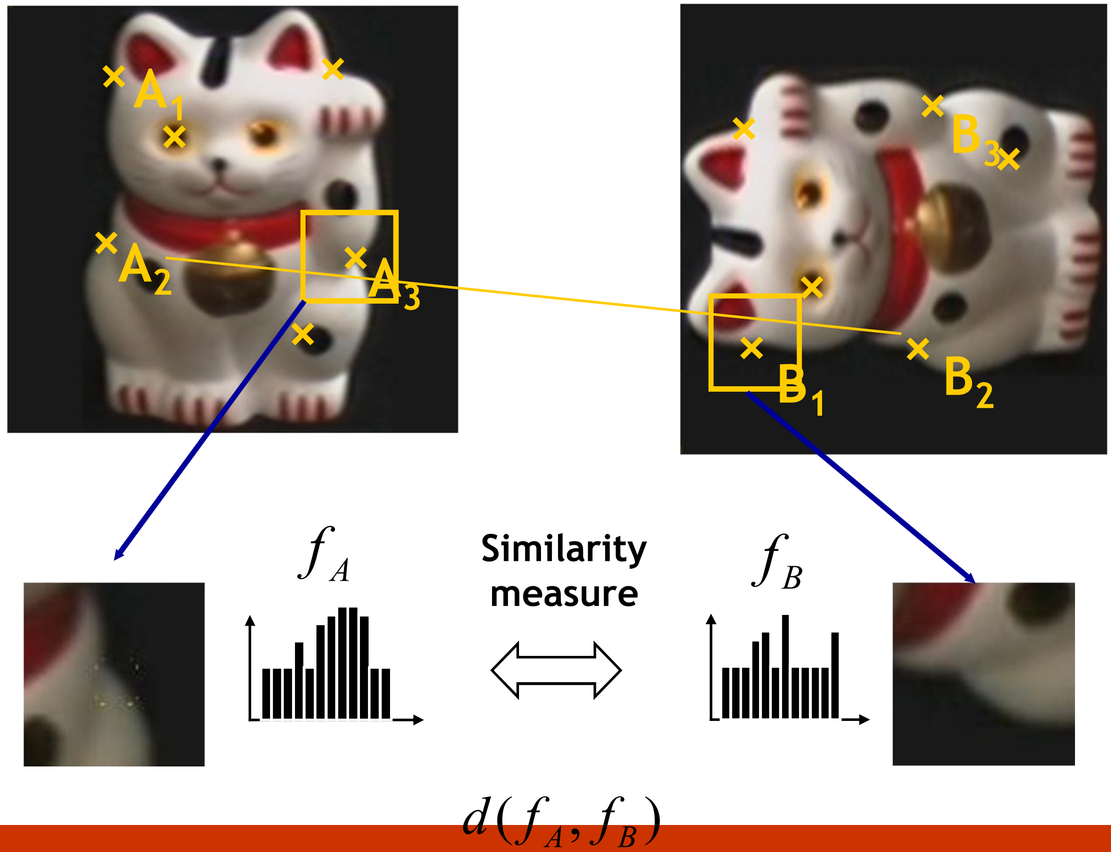
Matching strategies



$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

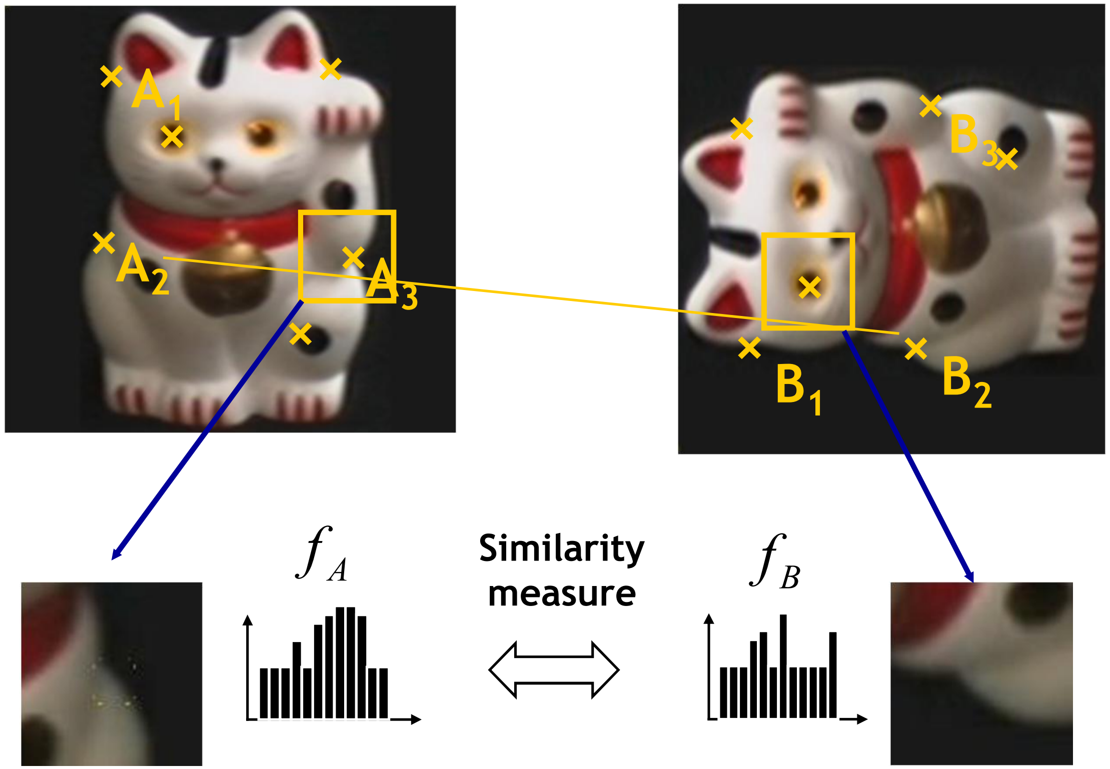
$$d(f_A, f_B) < T$$

Matching strategies



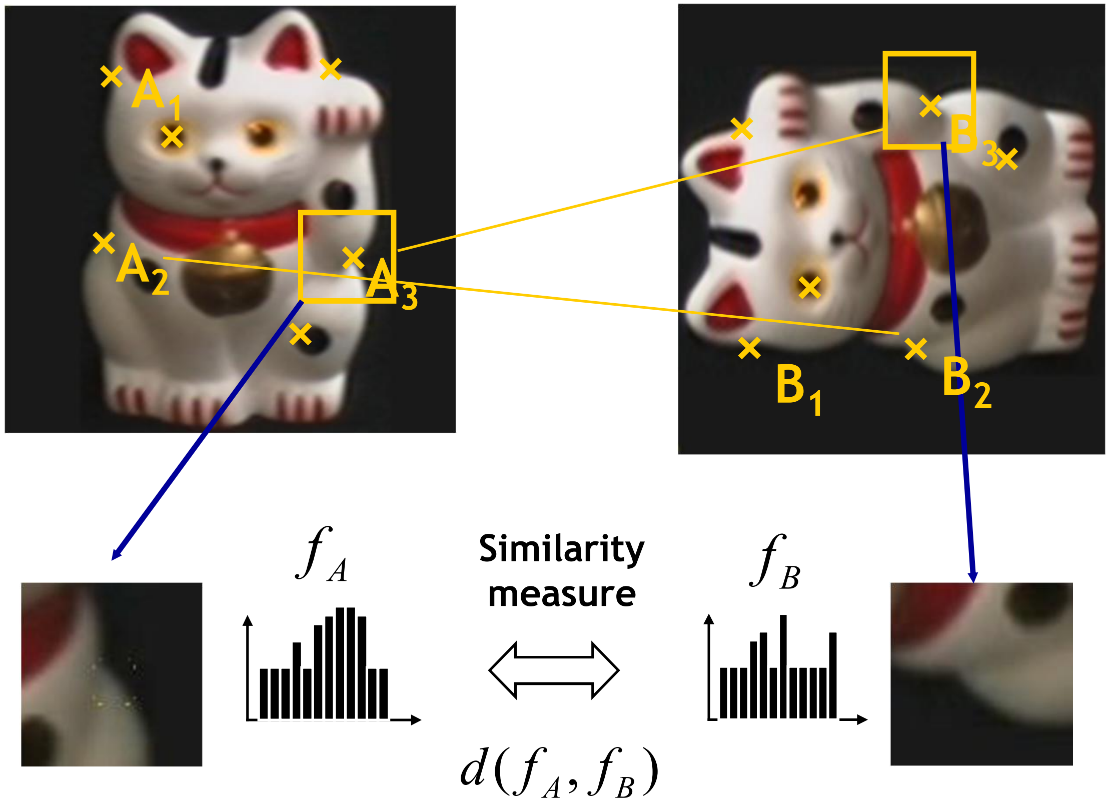
$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$

Matching strategies



$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

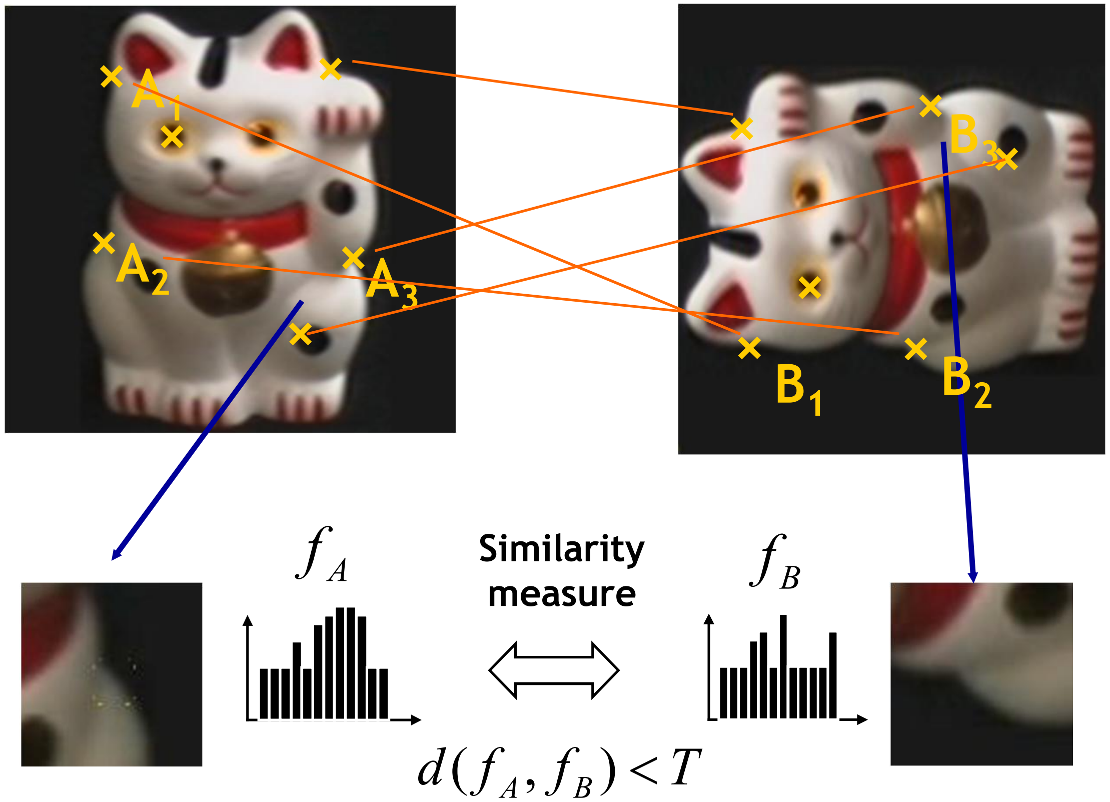
Matching strategies



$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

Matching strategies



$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$

$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$

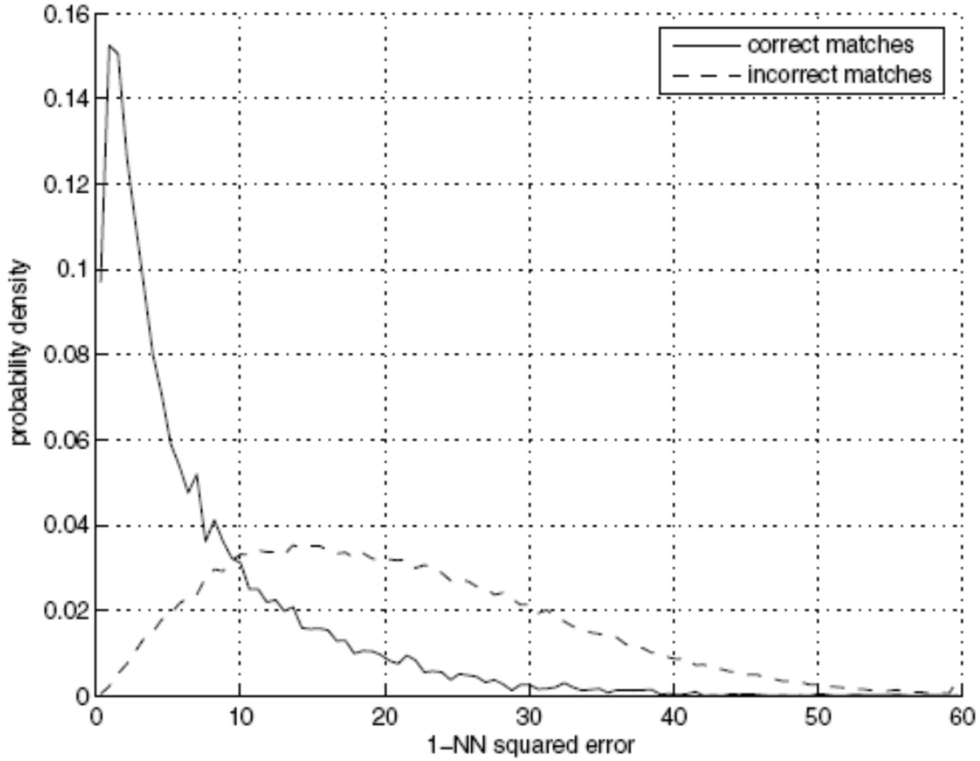
$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$

Matching strategies

- Matching algorithm
  1. Detect interest points in two images
  2. Extract patches and computing a descriptor
  3. Compare one feature from image 1 to every feature in image 2 and select the pair which gives the minimum distance (if below a threshold T)
  4. Repeating the above for each feature from image 1

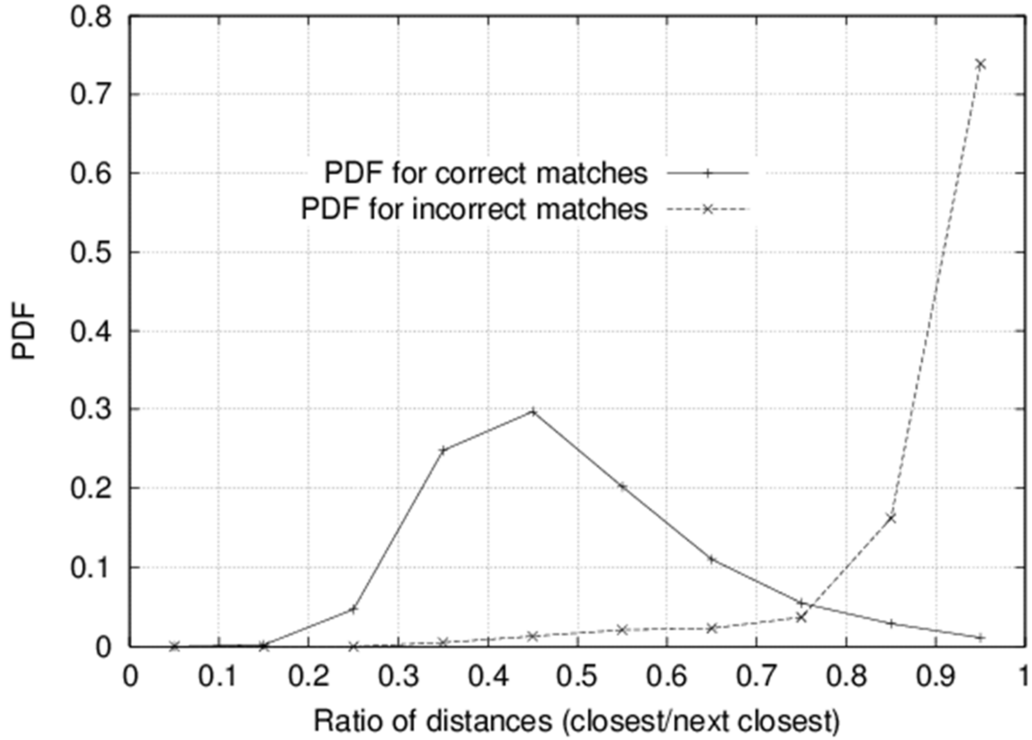
Matching strategies

- How to select the threshold ?



Matching strategies

- A better way (D. Lowe again):
  - Look at how much better 1-NN is than 2-NN, e.g. 1-NN/2-NN
  - That is, is our best match so much better than the rest?



What have we learnt today?

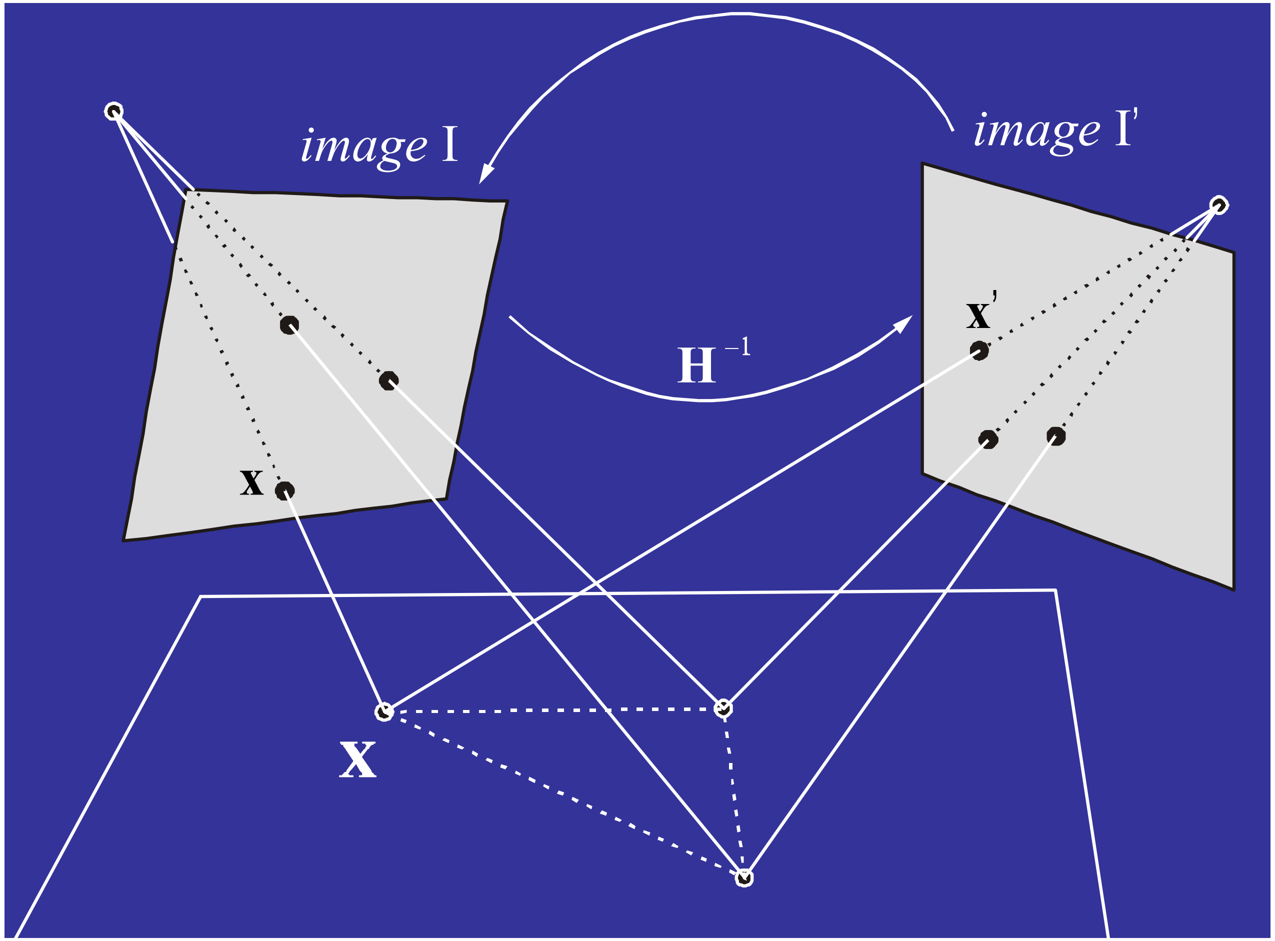
- SIFT is a carefully designed procedure with empirically determined parameters for the invariant, distinctive and robust description of local features

(SIFT: Scale-Invariant Feature Transform)

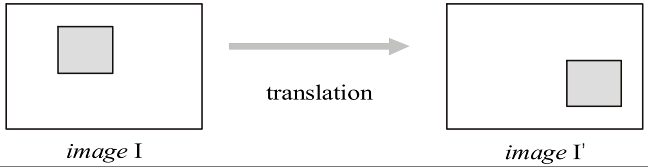
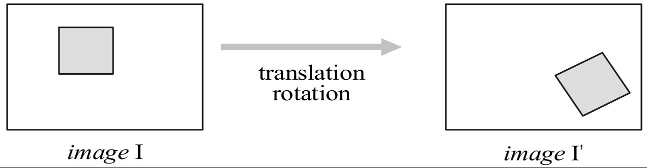
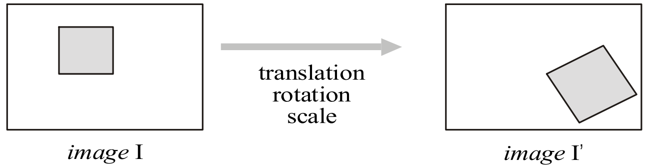
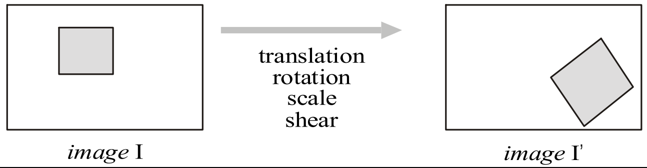
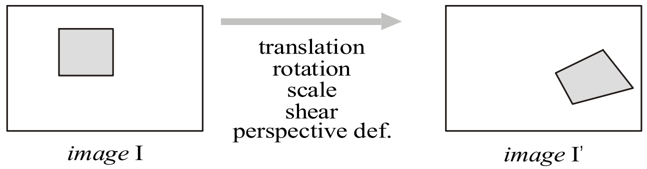
References

- Lowe, D.G. *Distinctive Image Features from Scale – Invariant Keypoints*. International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.

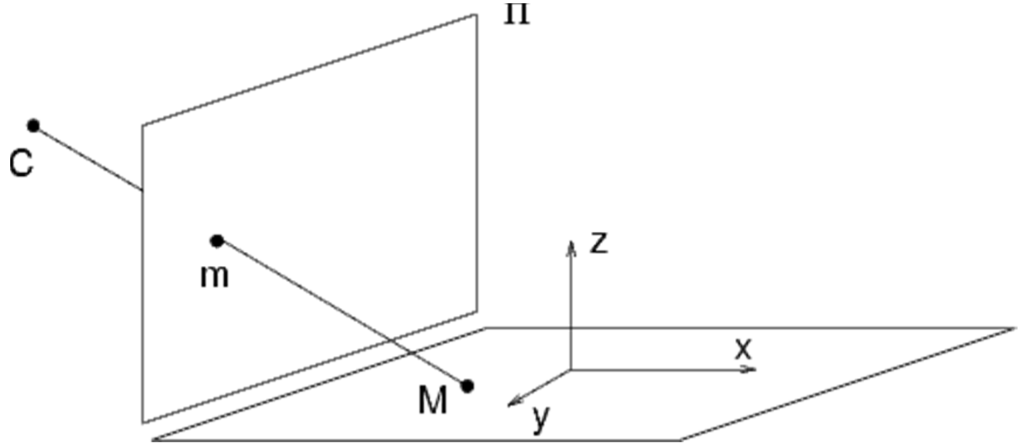
If we have time…
Special case: the scene is at a plane



A hierarchy of transformations

Model	Matrix	Distortion
translació pura 2 DOF	$\begin{bmatrix} \lambda x' \\ \lambda y' \\ \lambda \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	
Tranf. Euclideana 3 DOF	$\begin{bmatrix} \lambda x' \\ \lambda y' \\ \lambda \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	
Similarity 4 DOF	$\begin{bmatrix} \lambda x' \\ \lambda y' \\ \lambda \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	
Transf. Afí 6 DOF	$\begin{bmatrix} \lambda x' \\ \lambda y' \\ \lambda \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	
Transf. Projectiva 8 DOF	$\begin{bmatrix} \lambda x' \\ \lambda y' \\ \lambda \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$	

Homography from the projection matrix



$$\zeta \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$