

Image matching

Computational Vision

Francesca Odone – Francesca.odone@unige.it

Background needed

We are assuming you know the following

- The basic concepts of digital images
- Image spatial filtering
- Image features (corners in particular)

introduction

the concept of estimating the similarity between image pairs is very broad and is usually guided by the application

similarity is usually estimated with an *image matching* process where we try to match one image (or parts of it) with another

- *global matching*: **compute a global descriptor** for each image (eg, brightness/color mean and variance, color histograms, texture signatures, bag-of-features,...) and estimate the similarity between descriptors

image retrieval,
image classifications ...

- *local matching*: **extract local features** and solve a feature correspondence problem

image registration,
3D reconstruction...

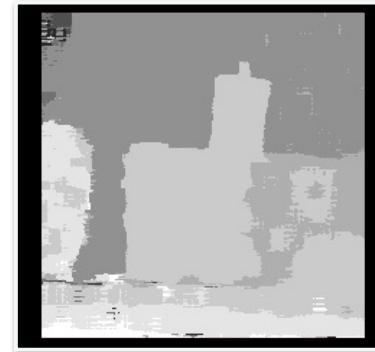
local matching as a correspondence problem

It involves two decisions

- 1.Which image elements to match
- 2.Which feature description+similarity measure to adopt



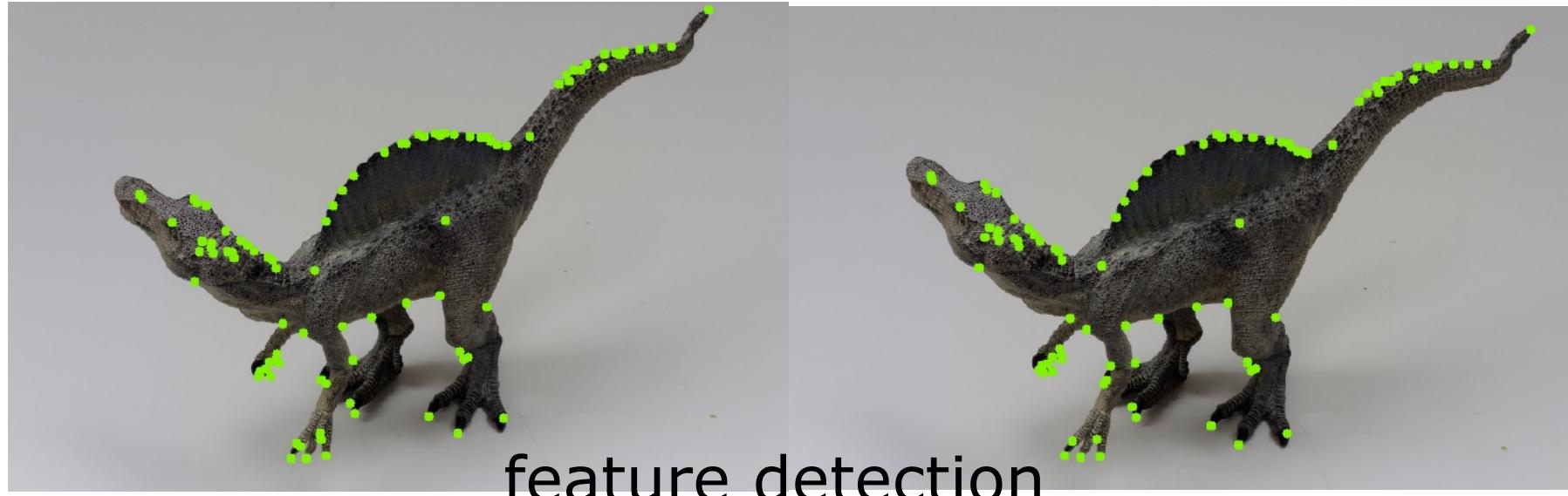
All the pixels from an image
(dense correspondences)



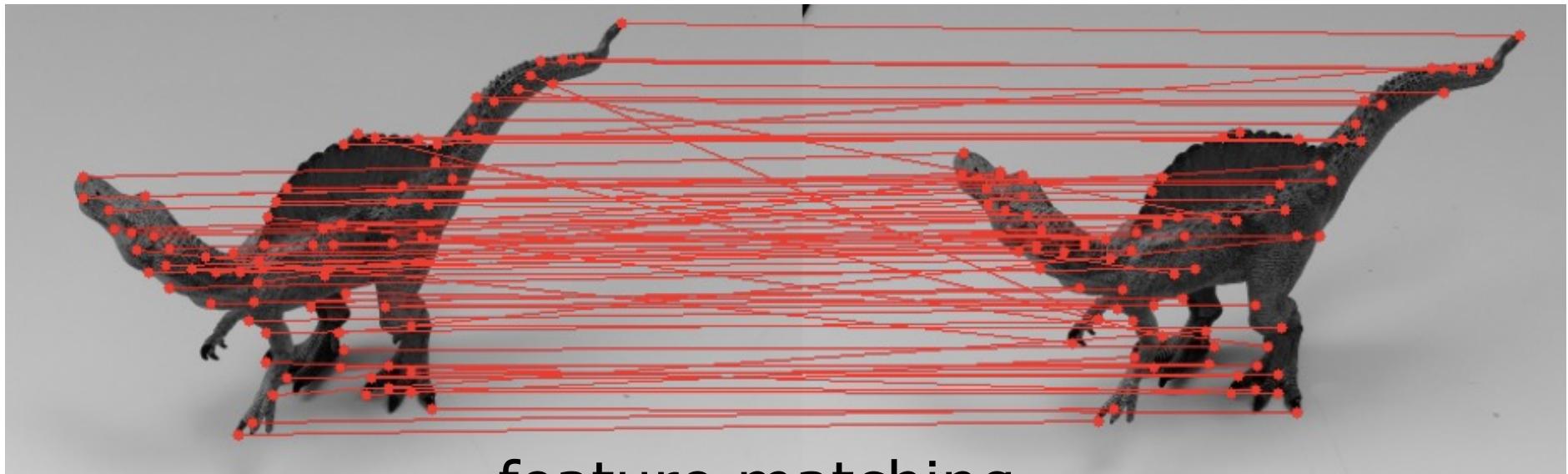
A subset of pixels meeting some requirements (**sparse correspondences**)



definitions: image matching



feature detection



feature matching

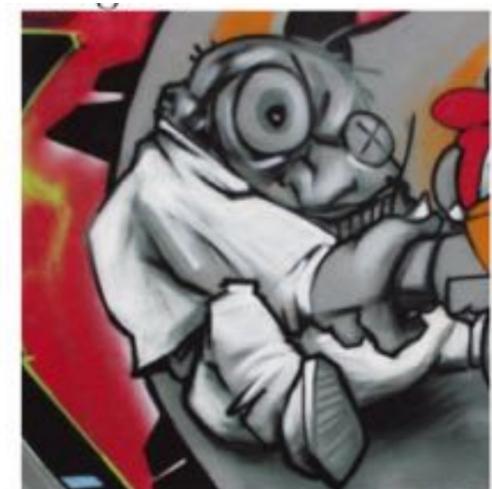
Our goal: local matching

Match elements in different images

Notice: objects may appear at different scales, rotation, viewpoint..

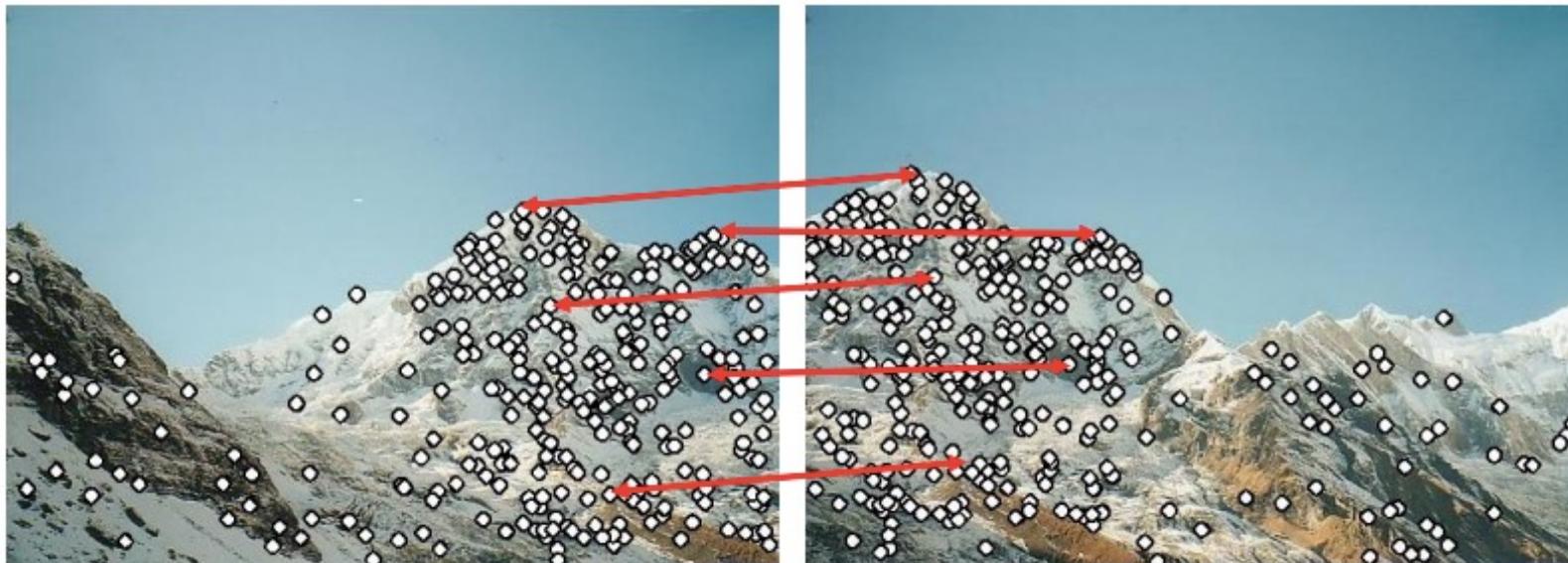
Pipeline

- 1. Feature detection:** Find interest points on each image
- 2. Feature description:** Compute a vector description for each point
- 3. Feature matching**



local feature matching “easy”

mostly translation



local feature matching “harder”

translation and illumination change
(plus small deformations)



by [Diva Sian](#)



by [swashford](#)

local feature matching “much harder”

view point change (appearance changes) +
illumination change

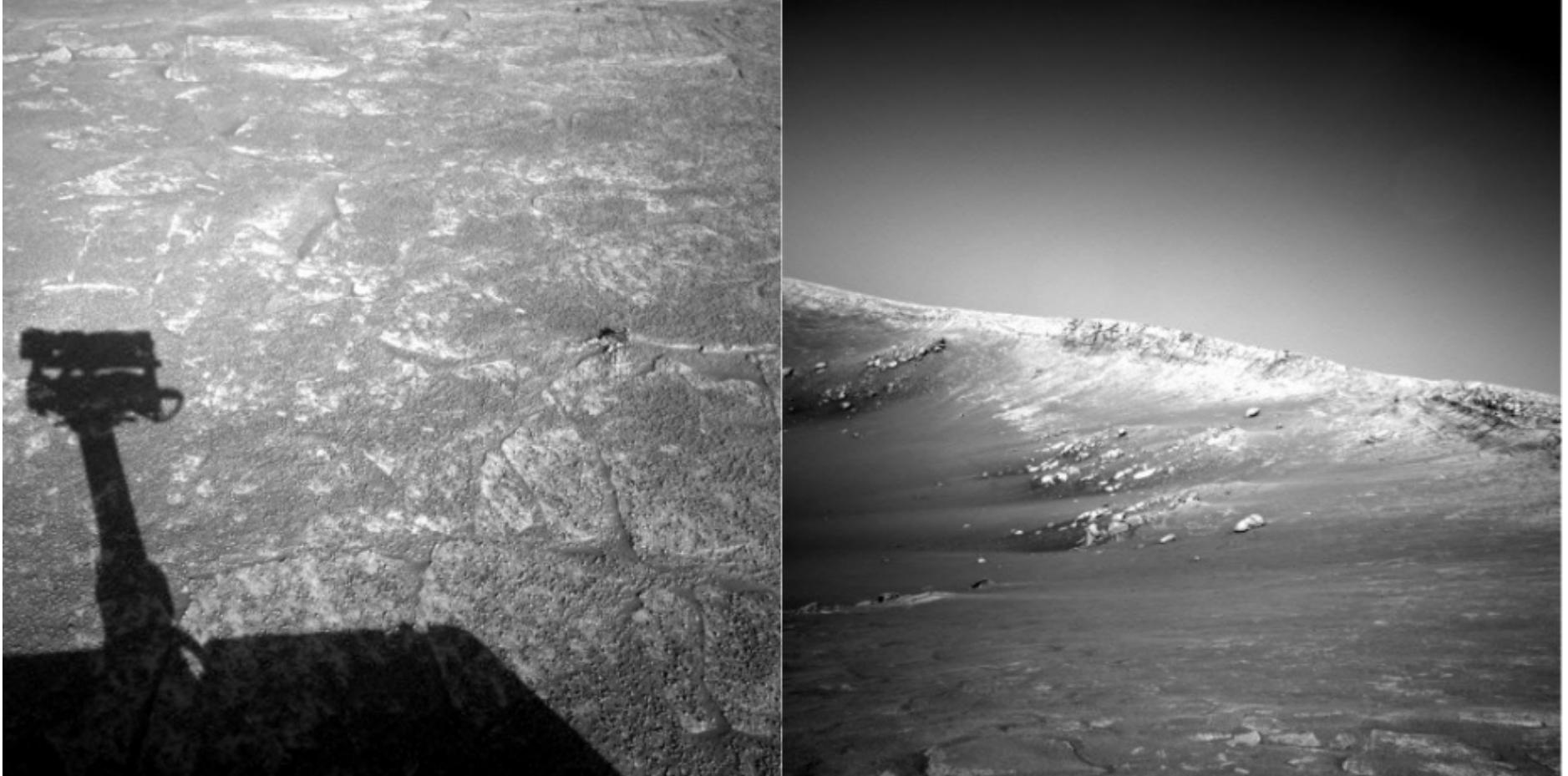


by [Diva Sian](#)



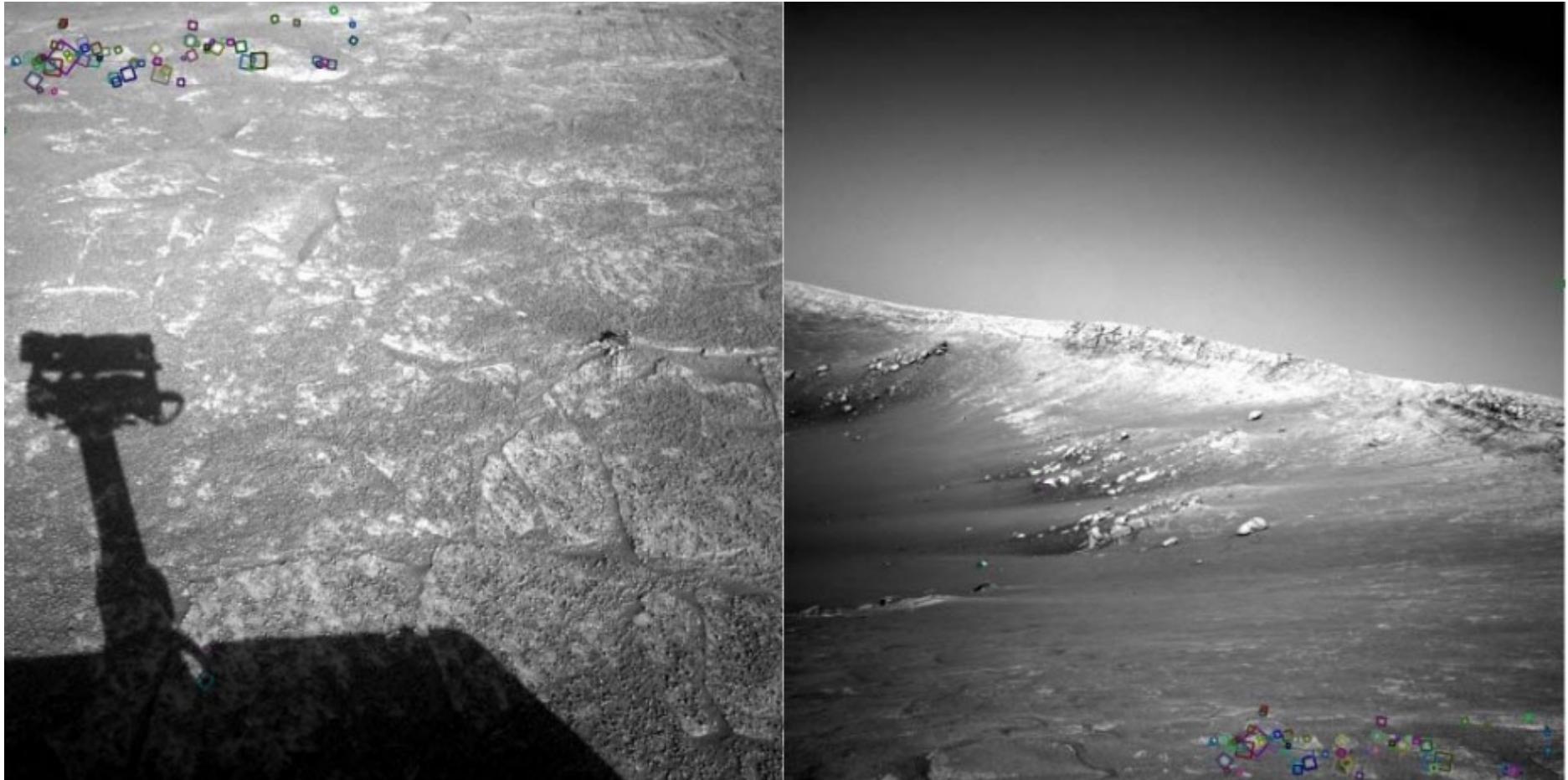
by [scgbt](#)

local feature matching “WOW!”



NASA Mars Rover images

local feature matching “WOW!”



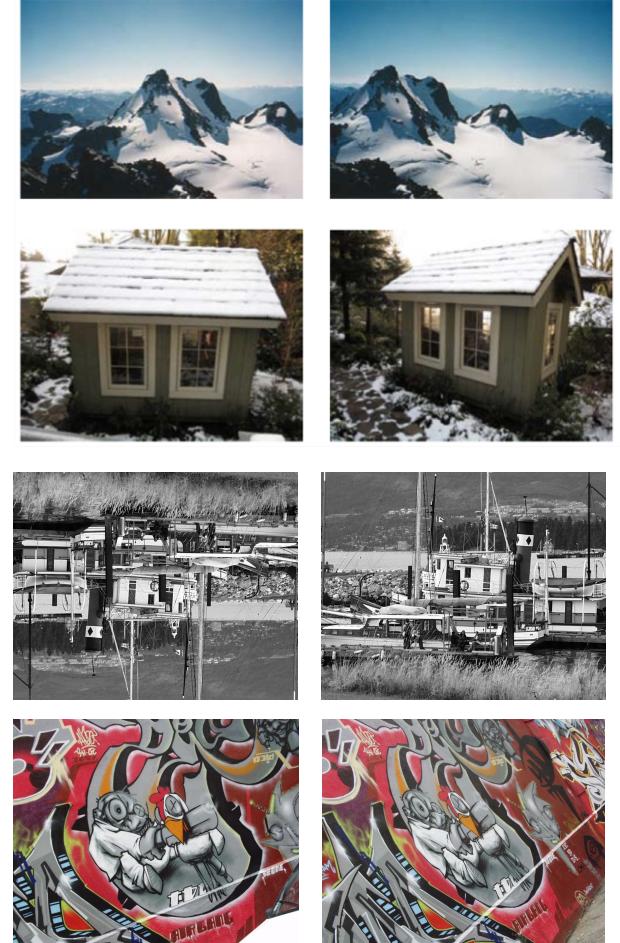
NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

local feature matching

- If image pairs which are “similar enough”, then local features undergo a (quasi) translation transformation

- interest points may be **corners**
- **image patches** are an appropriate feature description

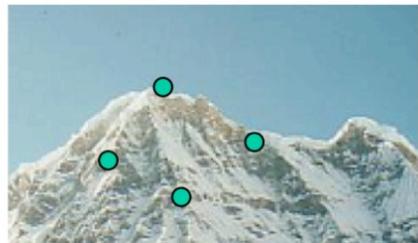
- In the case of scale, rotation, and more severe viewpoint changes we may need **scale invariant** interest points and better feature descriptors



local features matching: detection

Problem 1:

Detect the same point independently in both images

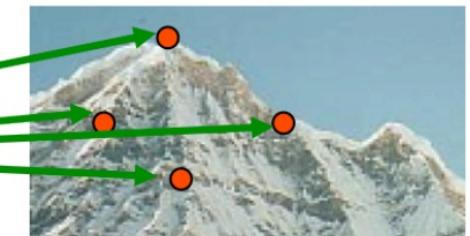
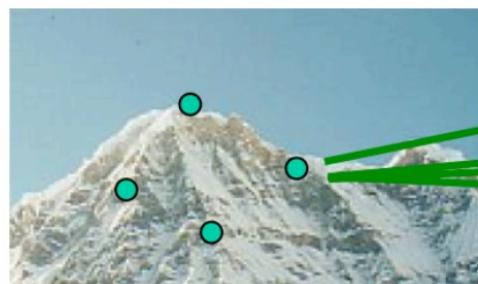


no chance to match!

We need a **repeatable** feature detector

Problem 2:

For each point correctly recognize the corresponding one



We need a **reliable and distinctive** feature detector

(scale invariant) Feature detectors

Local features and scale

Scale changes affect image content dramatically

Objects may look very different, with new details emerging as we get close

This general observation has an impact on image understanding since the very first processing stages (e.g. local features detection)



Automatic Scale Selection

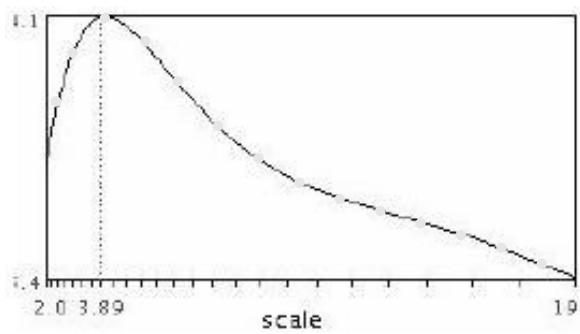
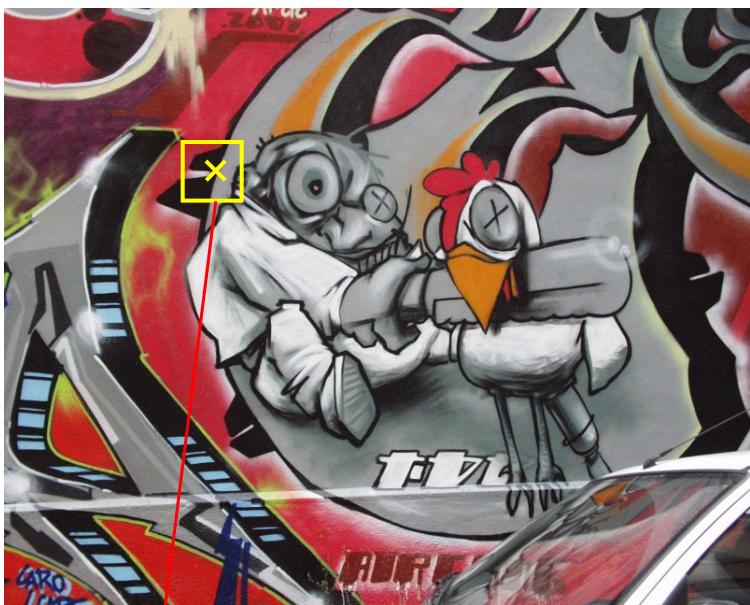


How to find corresponding patch sizes?

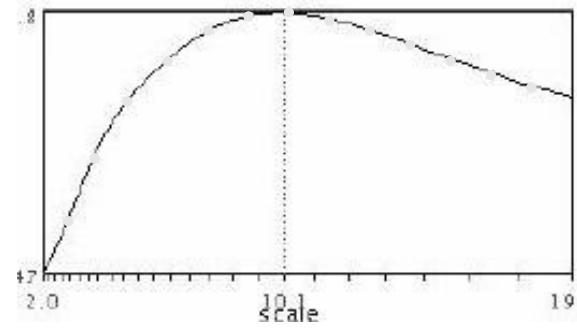
SLIDES BY K. Grauman, B. Leibe

Automatic Scale Selection

Function responses for increasing scale (scale signature)

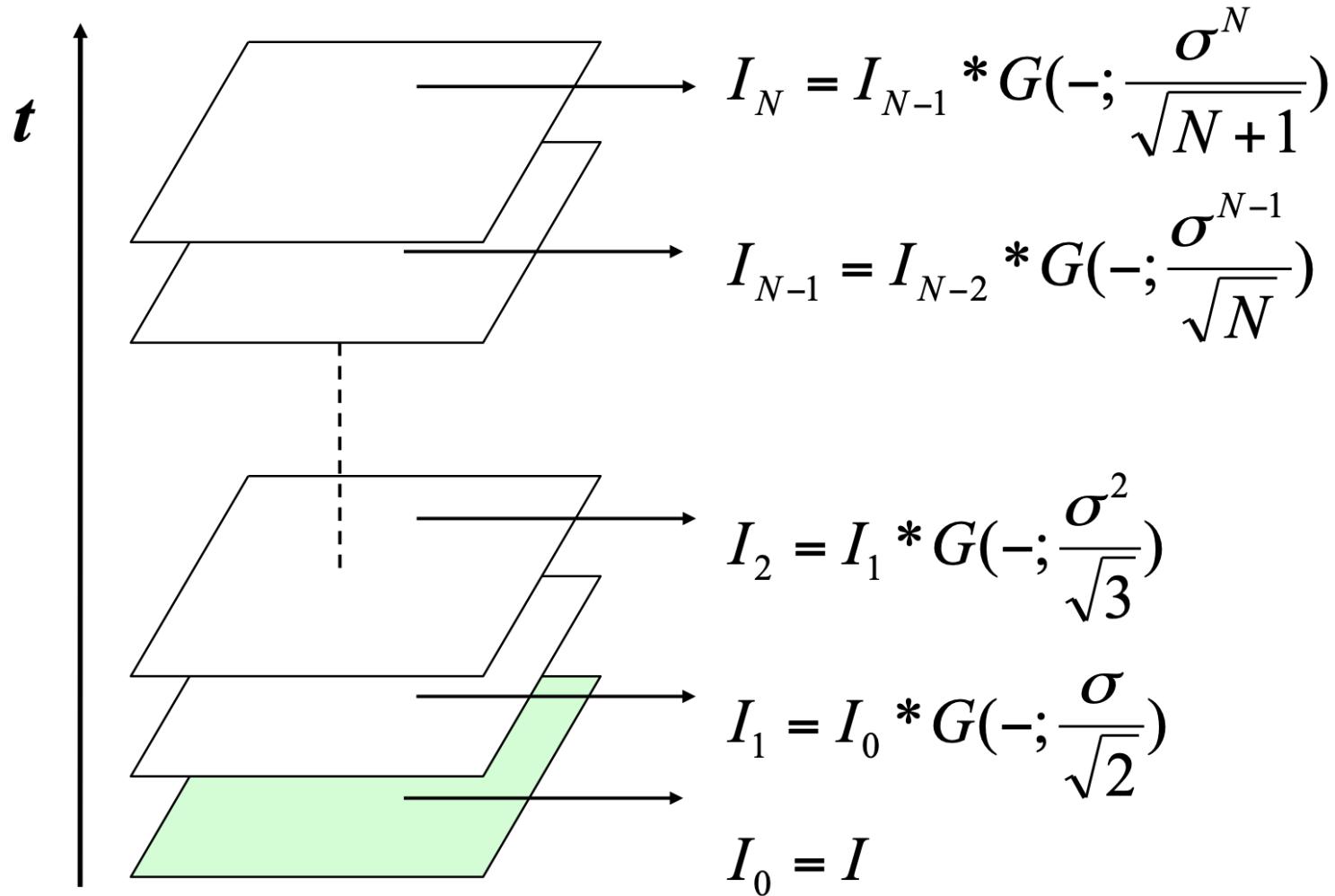


$$f(I_{i_1 \dots i_m}(x, \sigma))$$

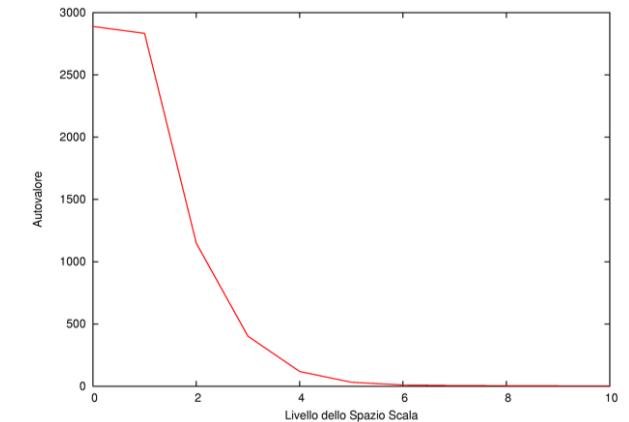
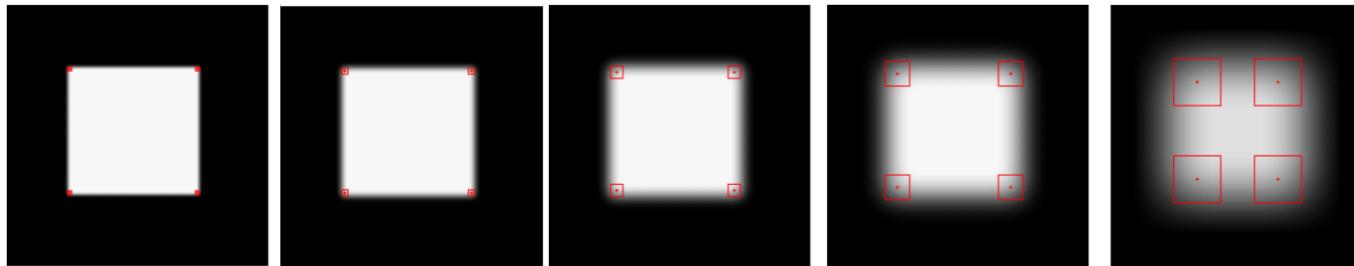


$$f(I_{i_1 \dots i_m}(x', \sigma'))$$

scale-space implementation



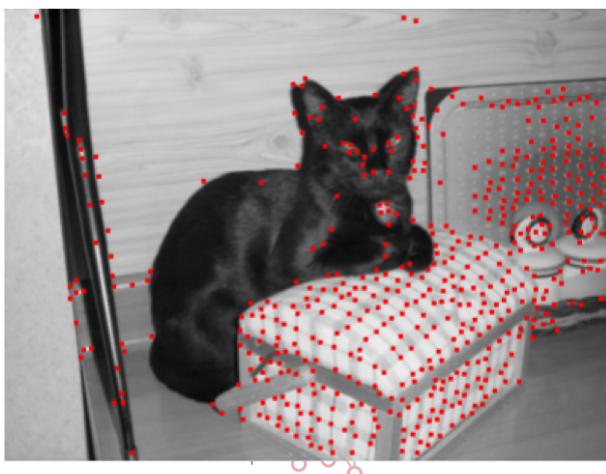
corners and scale-space



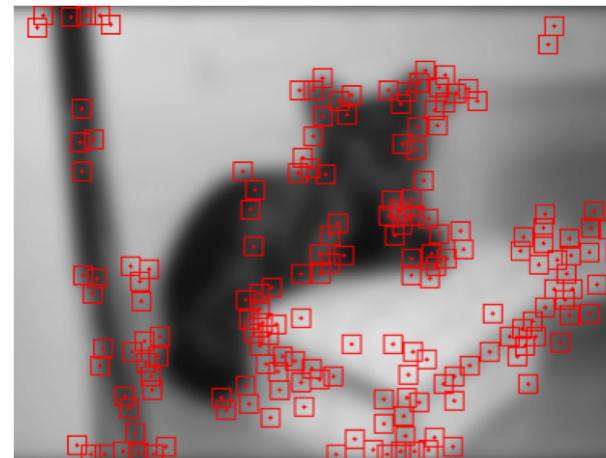
- compute corners at each image layer on an appropriate neighbourhood

- for each corner choose the most appropriate scale and suppress the others (for this, you'd need a *corner SCALE signature* (Lindberg 1994))

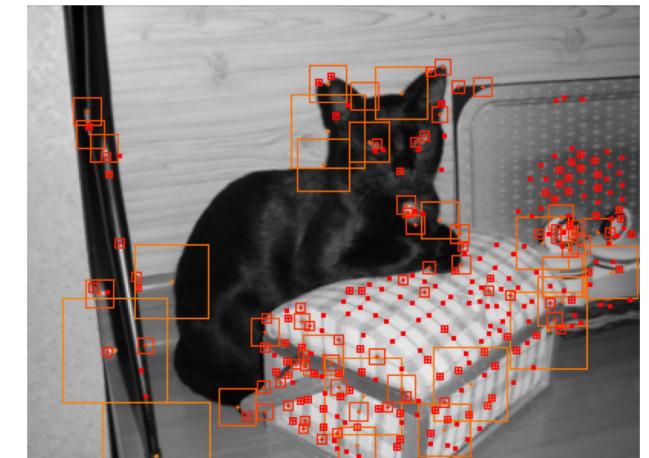
level1



level5

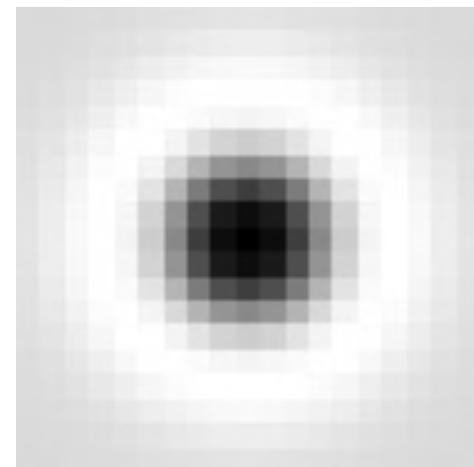
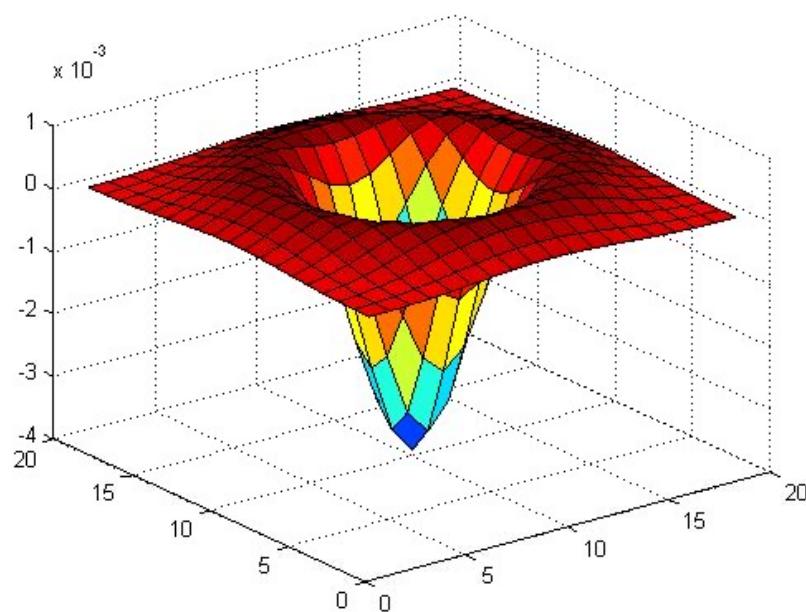


output



What can it be a signature function?

Laplacian of Gaussian (LoG): Circularly symmetric operator

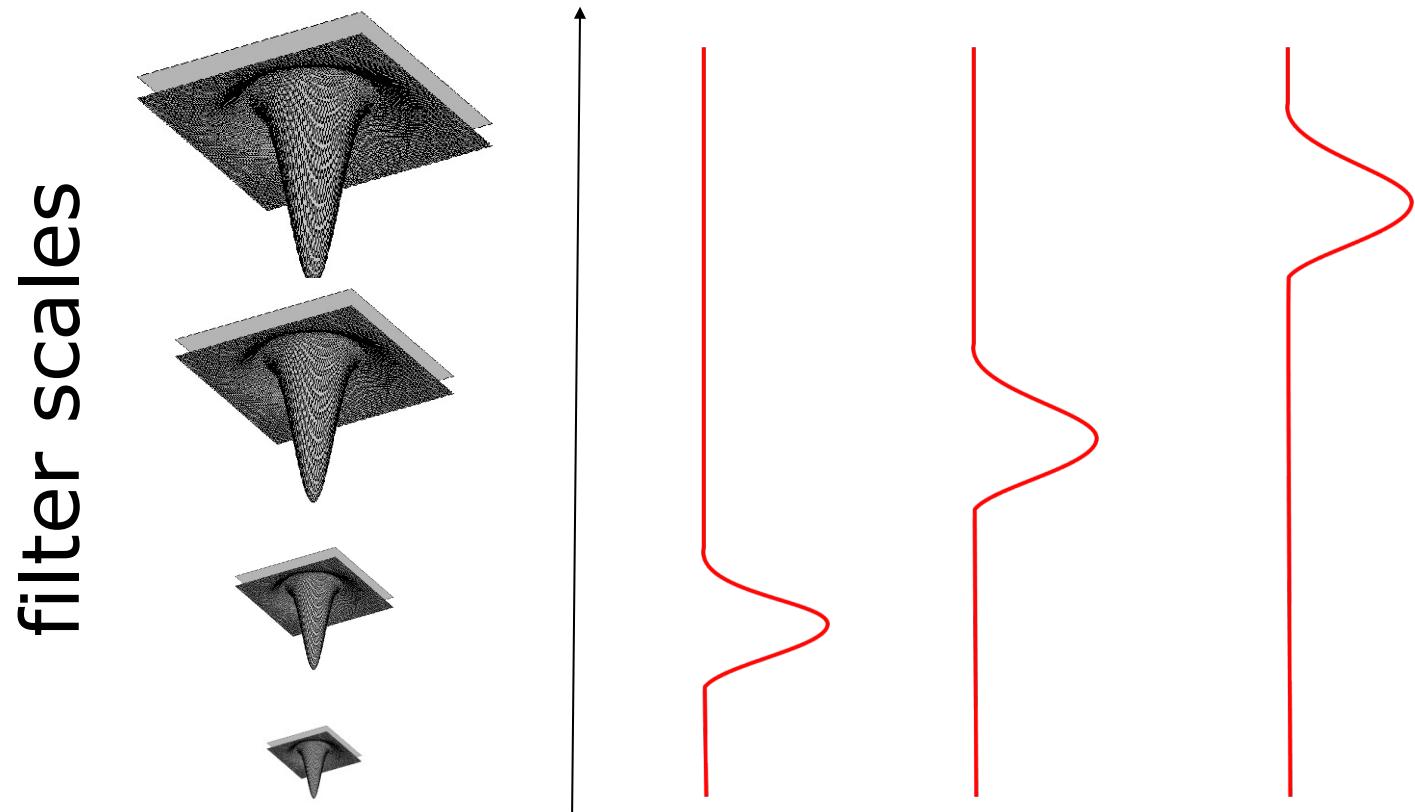


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Slide by K. Grauman

LoG and scale selection

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



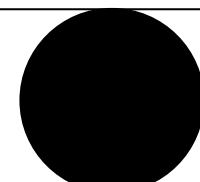
Slide credit: Bastian Leibe



img1



img2



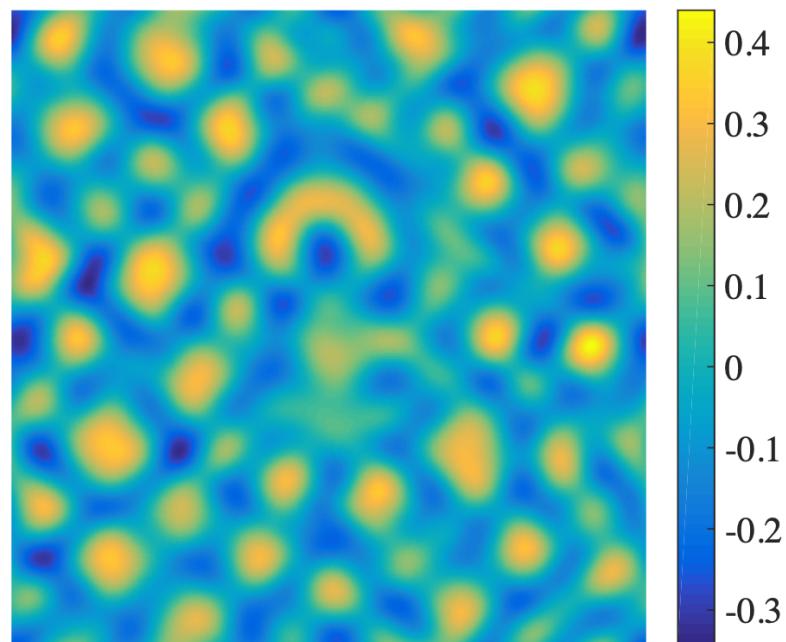
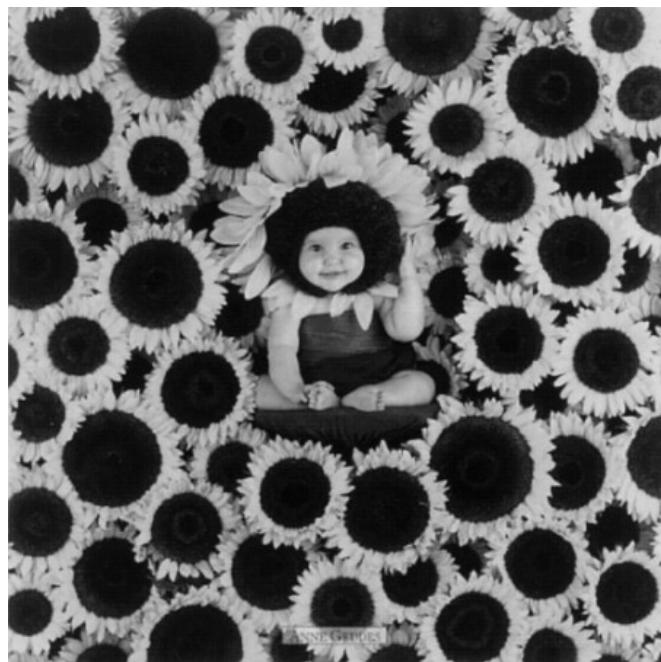
img3

Blob-like features

Another type of local key point quite naturally associated with the concept of scale

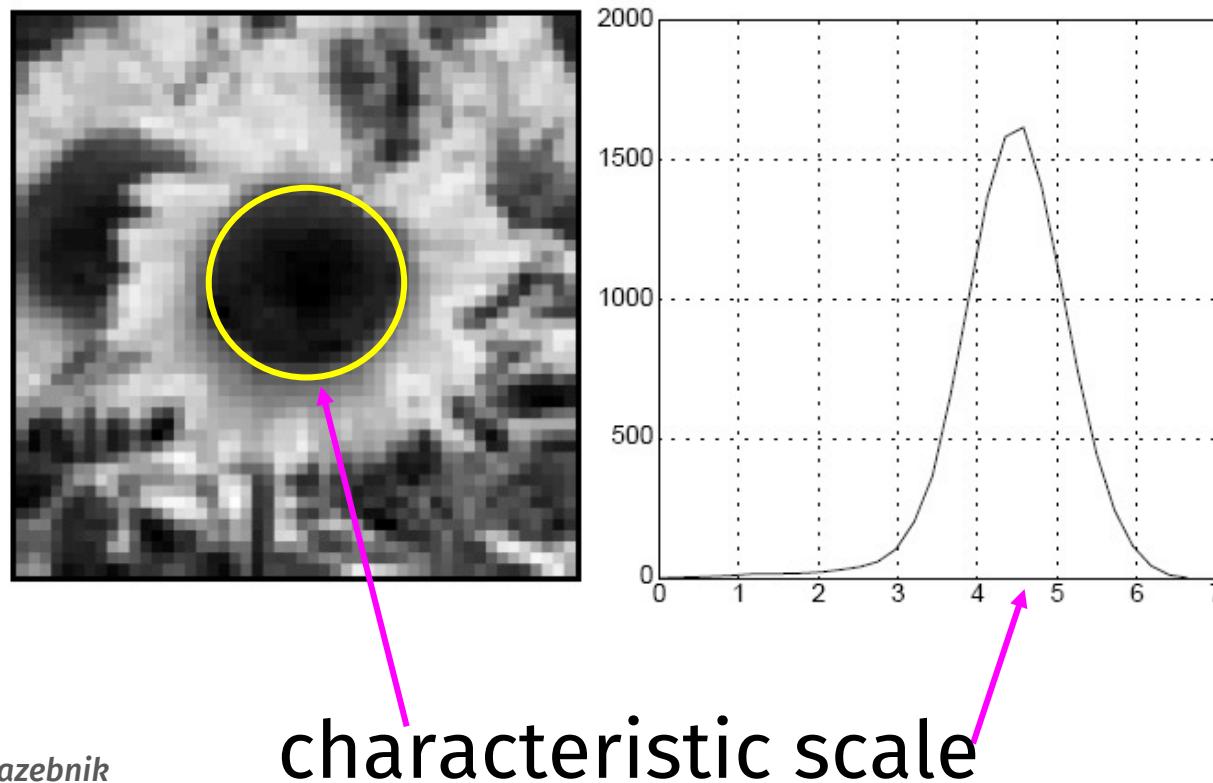
Blobs - uniform areas “surrounded” by a sharp variation of the signal

blob enhancement may be carried out by looking for extrema of the image Laplacian



Blob detection and scale

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



Slide credit: Lana Lazebnik

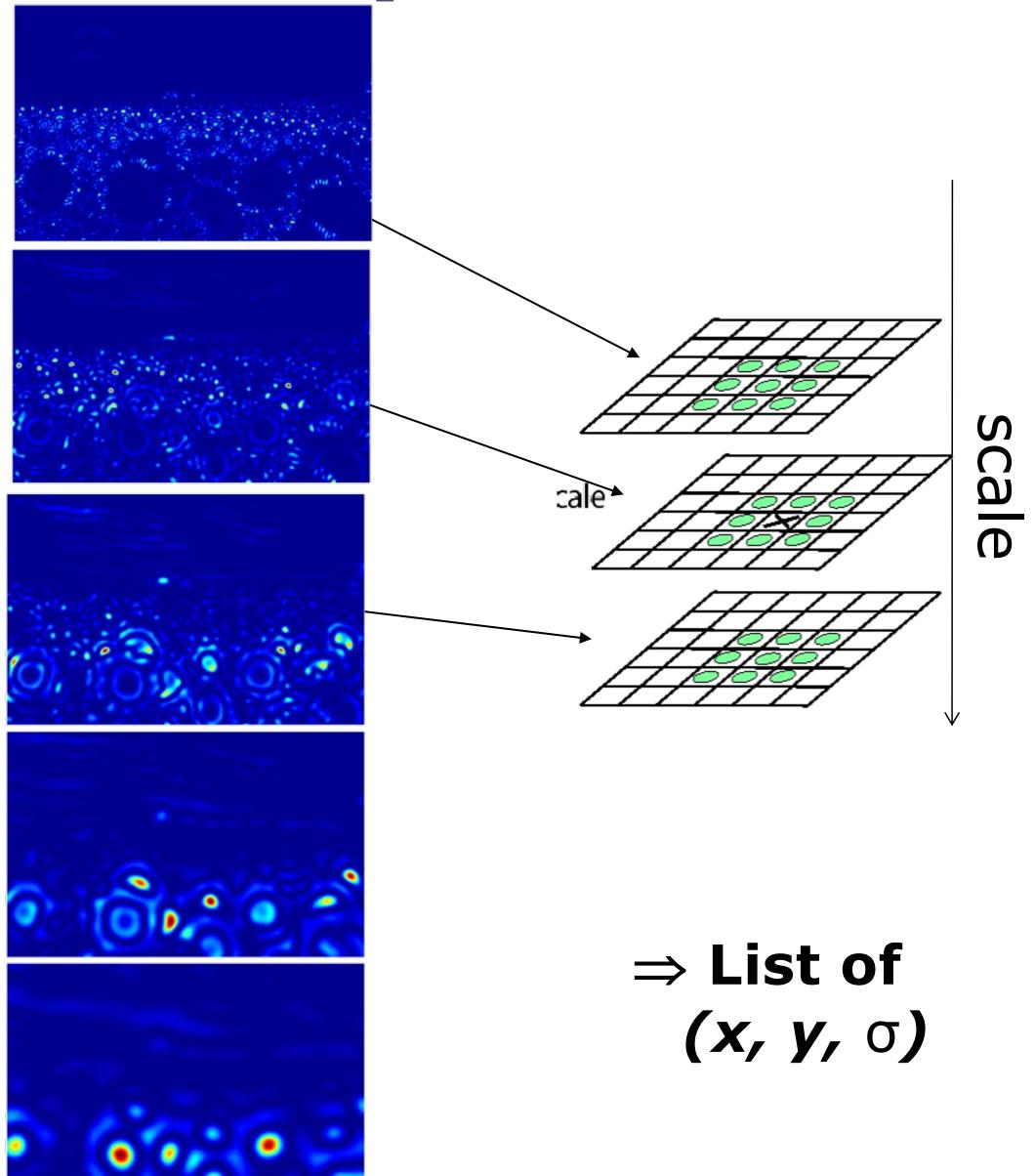
Scale invariant interest points

Interest points are local extrema in both position and scale.



$$L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \sigma_3$$

$$\begin{matrix} & \nearrow \sigma_5 \\ & \nearrow \sigma_4 \\ & \nearrow \sigma_3 \\ & \nearrow \sigma_2 \\ & \nearrow \sigma_1 \end{matrix}$$



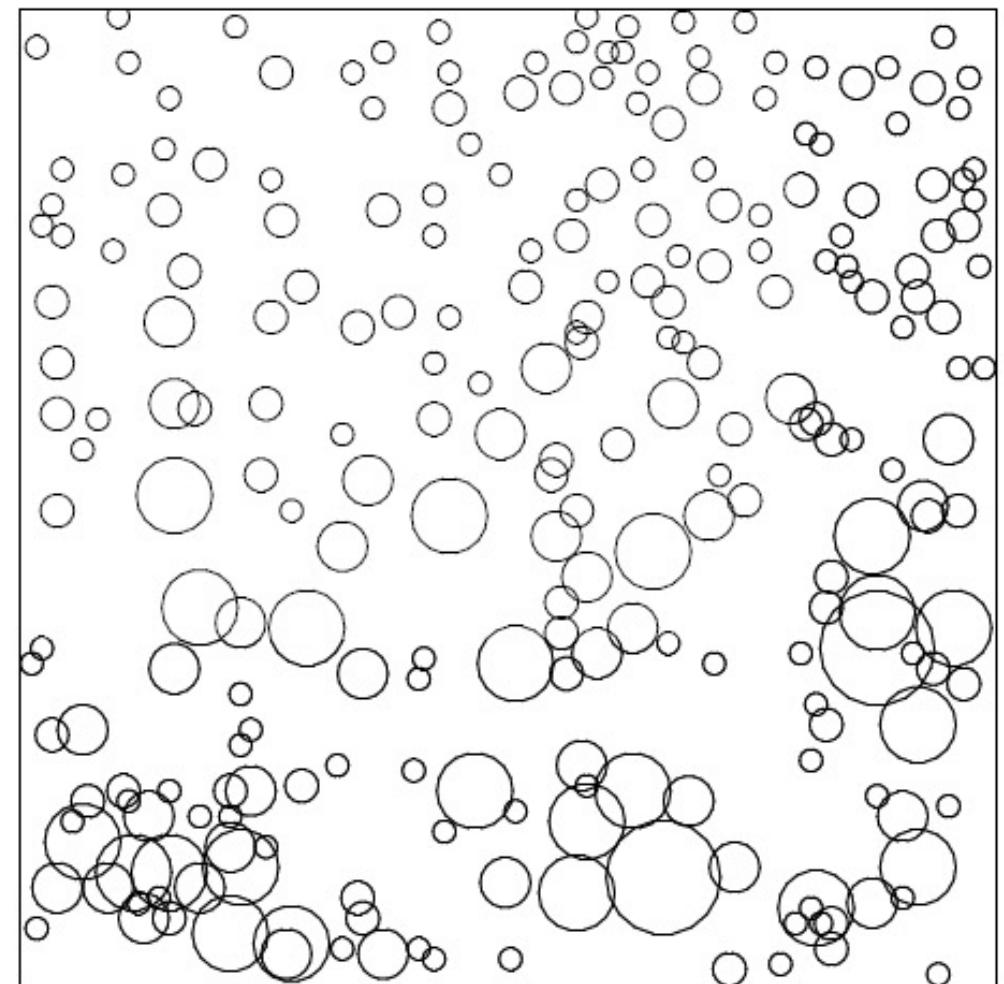
Slide credit: Kristen Grauman

Scale-space blob detector: Example

original image



scale-space maxima of $(\nabla^2_{norm} L)^2$



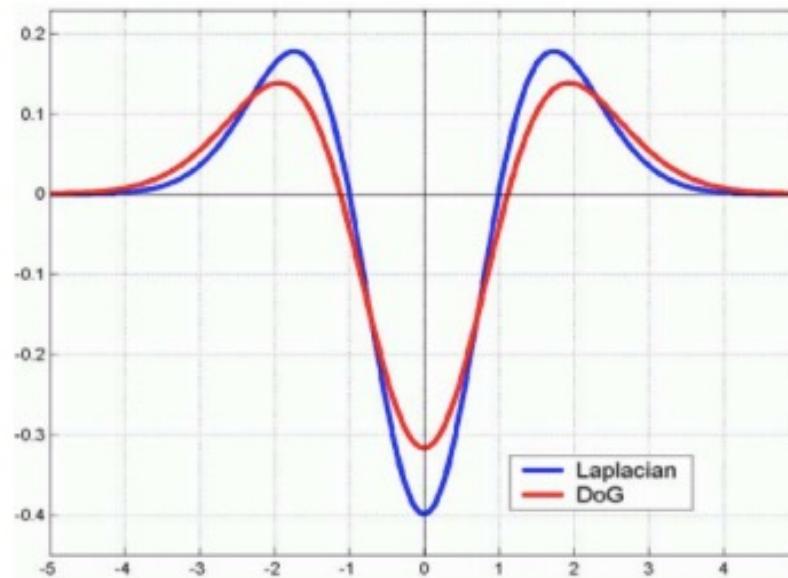
DoG features

it can be shown that the Laplacian operator may be approximated by the difference of the image smoothed with two different Gaussian filters

$$\nabla^2 L(x, y; t) \sim c * (L(x, y; t + \Delta t) - L(x, y; t))$$

normalization factor

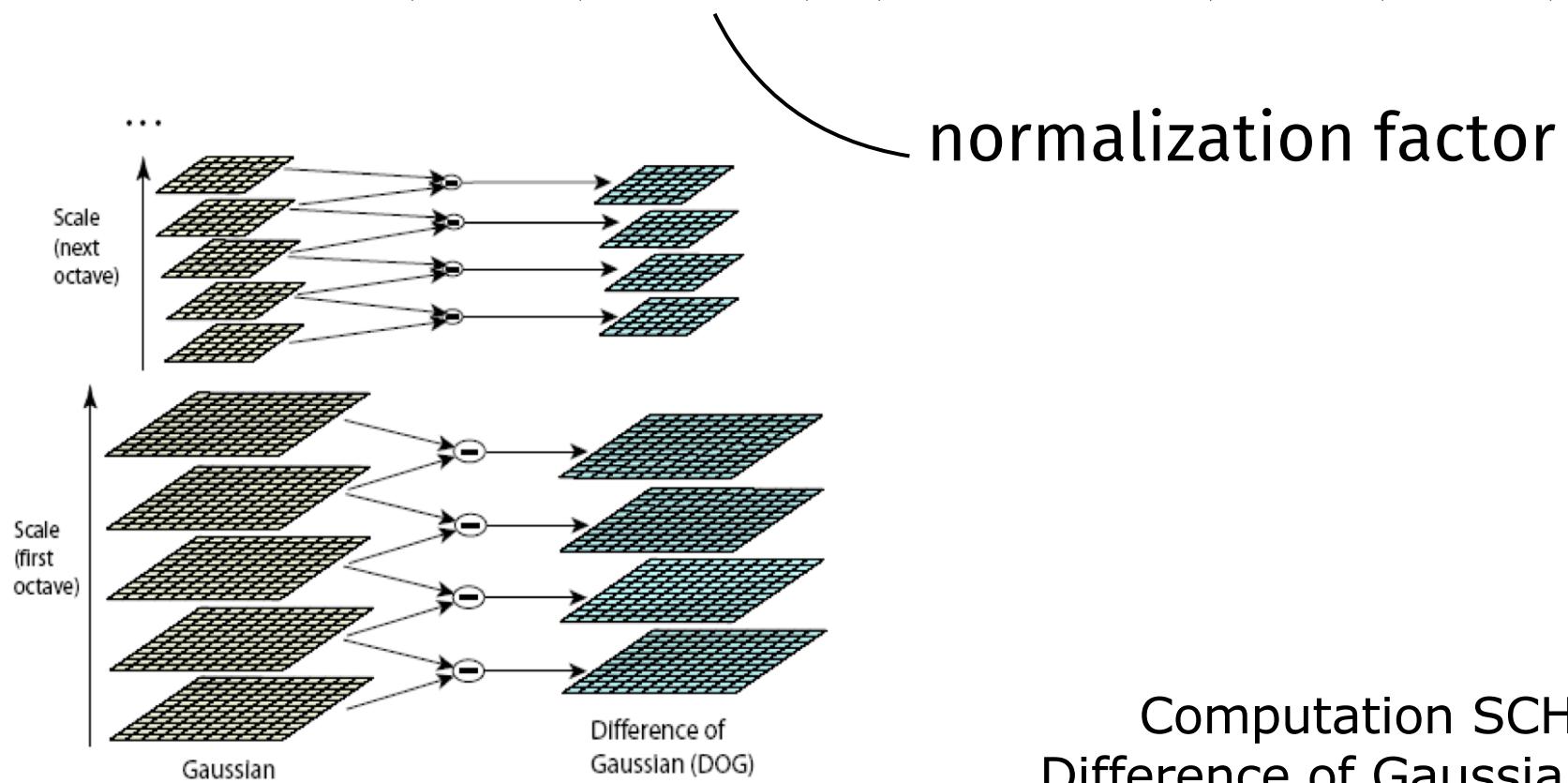
L(x,y;t) is an image I filtered with a gaussian kernel of variance t



DoG features

it can be shown that the Laplacian operator may be approximated by the difference of the image smoothed with two different Gaussian filters

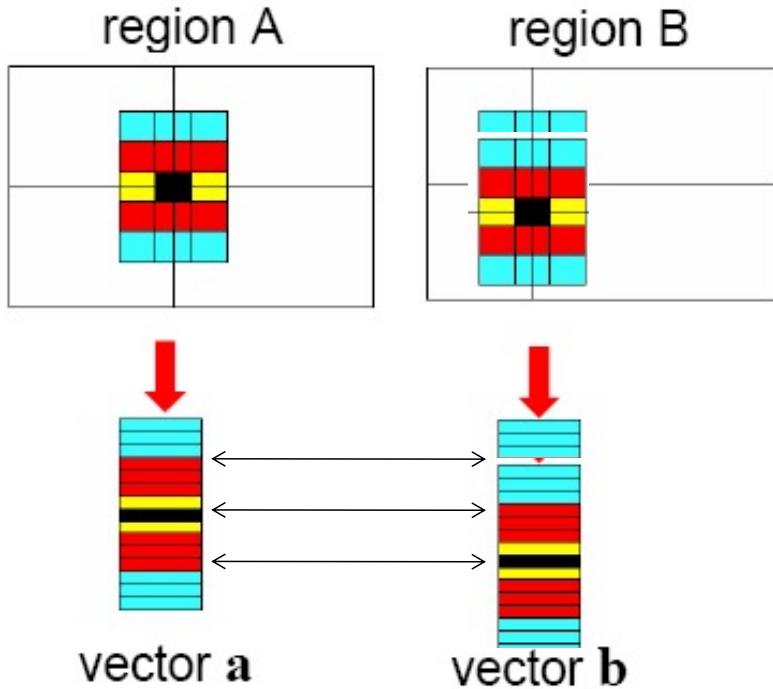
$$\nabla^2 L(x, y; t) \sim c * (L(x, y; t + \Delta t) - L(x, y; t))$$



Computation SCHEME
Difference of Gaussian pyramid

Feature descriptors

Raw patches



The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.

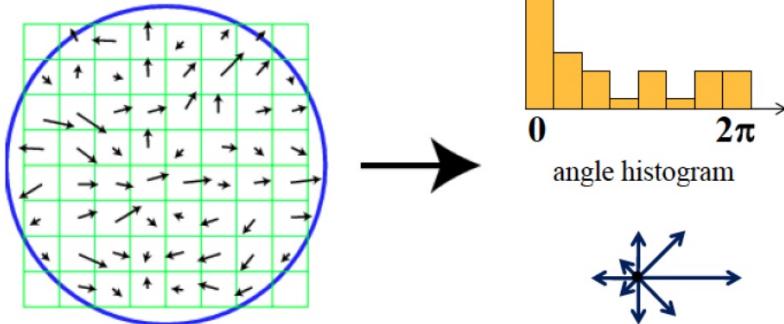
Figure: Andrew Zisserman

invariant feature descriptors

Scale Invariant Feature Transform (SIFT)

Basic idea:

- ▷ Take 16x16 square window around detected interest point (8x8 shown below)
- ▷ Compute edge orientation (angle of the gradient minus 90°) for each pixel
- ▷ Throw out weak edges (threshold gradient magnitude)
- ▷ Create histogram of surviving edge orientations (8 bins)

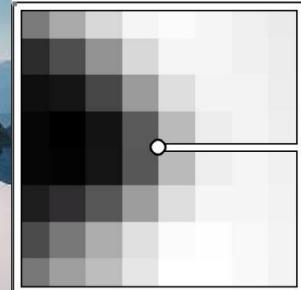


see also more recent SURF (fast), FAST (faster), BRISK (even faster)

Multiscale Oriented PatcheS descriptors (MOPS)

Take a 40x40 square window around detected feature

- ▷ scale to 1/5 size (using prefiltering) to get 8x8 square window
- ▷ rotate to horizontal
- ▷ normalize the window values by subtracting the mean and dividing by the standard deviation in the window.



Scale Invariant Feature Transform (SIFT) by David Lowe (2004)

The idea is to detect interesting points (as corners, blobs,...) on a multi-scale image representation, and then associate with them a vectorial description which is invariant to translation, scale and rotation changes

- Very robust
- Fast & efficient (~real time)
- Also tolerant to illumination changes

It's a reference method (patent, apps, ...)

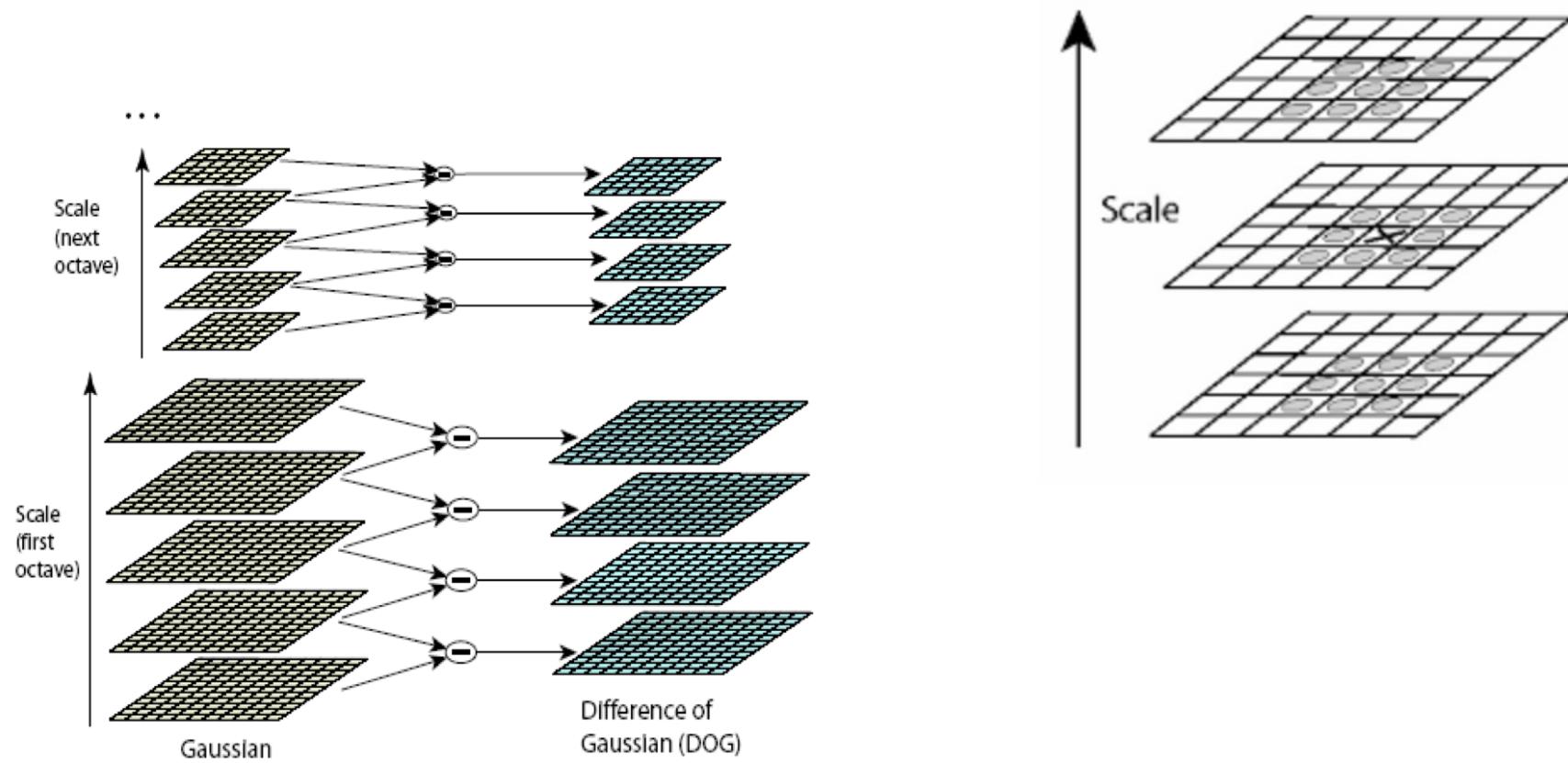
DoG detection and SIFT description

Main steps:

1. Scale-space image construction
2. scale-space extrema detection (DoG keypoints)
3. accurate key point location (sub-pixel analysis, remove low contrast and edges)
4. keypoint orientation assignment
5. keypoint descriptor

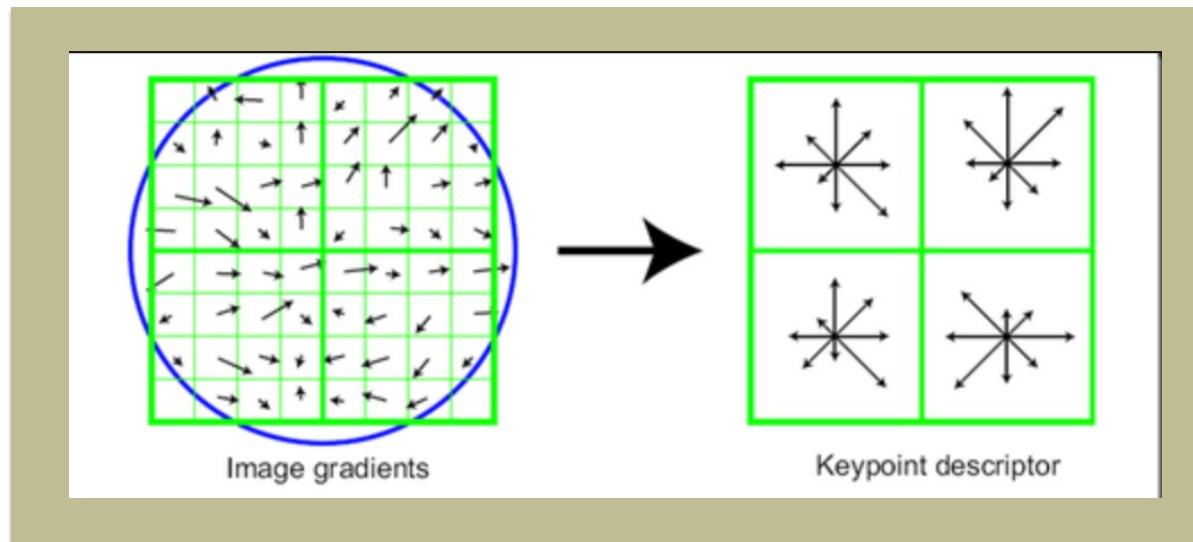
DoG detection and SIFT description

Local features are detected as extrema of the multi-scale representation



DoG detection and SIFT description

Local features are detected as extrema of the multi-scale representation



Each local feature is represented collecting the information of the gradient around its location, considering a patch of size corresponding to its scale and rotated according to its orientation

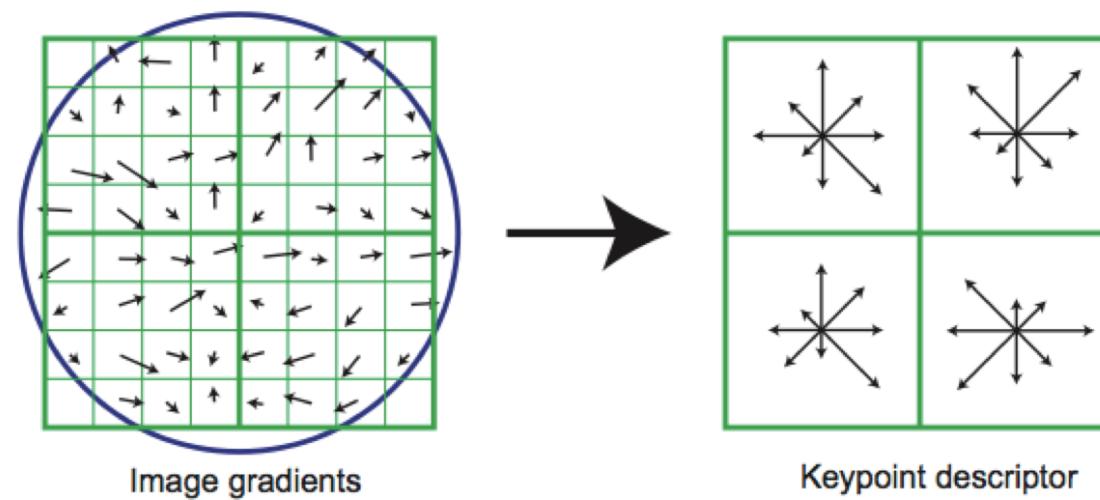
DoG detection and SIFT description

Keypoint orientation assignment

Compute an orientation histogram (36 bins covering 360 degrees)

Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian window proportional to the scale

in the actual implementation
not 8x8 but 16x16

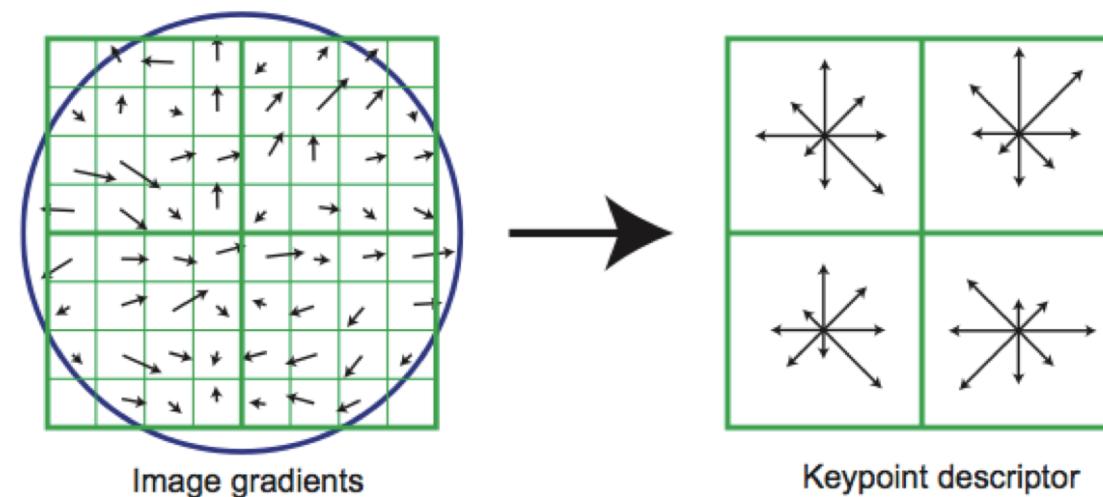


DoG detection and SIFT description

Keypoint orientation assignment

The highest peak in the histogram gives the main orientation of the feature

any other local peak within 80% of the highest peak is used to create secondary features

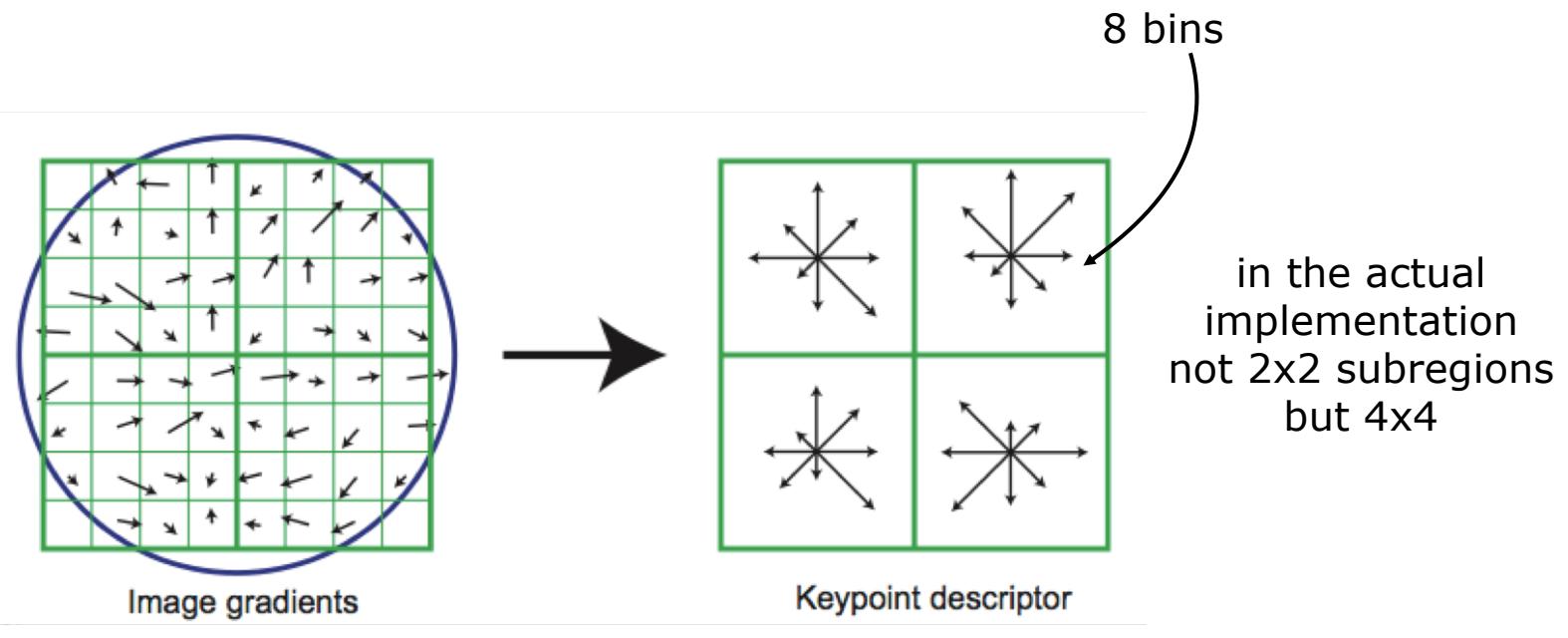


DoG detection and SIFT description

Keypoint descriptor

4 x 4 histograms with 8 orientations each - Histogram concatenation with 128 float values

the feature vector is normalized to compensate intensity changes



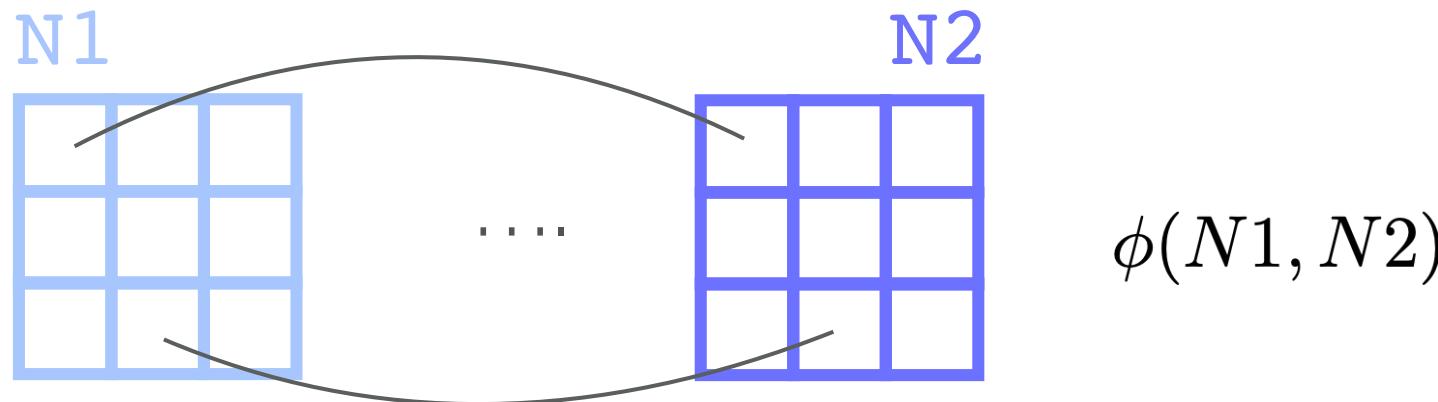
Feature matching

(Similarity measure + matching strategy)

Similarity measures for image patches

Let N_1 and N_2 be two square image patches of size $W \times W$

How to measure the similarity ?



Similarity measures for image patches

SUM OF SQUARED DIFFERENCES

$$\phi_{SSD}(N1, N2) = - \sum_{k,l=-\frac{W}{2}}^{\frac{W}{2}} (N1(k, l) - N2(k, l))^2$$

NORMALIZED CROSS CORRELATION

$$\phi_{NCC}(N1, N2) = - \sum_{k,l=-\frac{W}{2}}^{\frac{W}{2}} \frac{(N1(k, l) - \mu_1)(N2(k, l) - \mu_2)}{W^2 \sigma_1 \sigma_2}$$

$$\mu_i = \frac{1}{W} \sum_{k,l=1}^W Ni(k, l)$$

NCC: values in
the range [-1, 1]

$$\sigma_i = \sqrt{\frac{1}{W} \sum_{k,l=1}^W (Ni(k, l) - \mu_i)^2} \quad \text{with } i = 1, 2$$

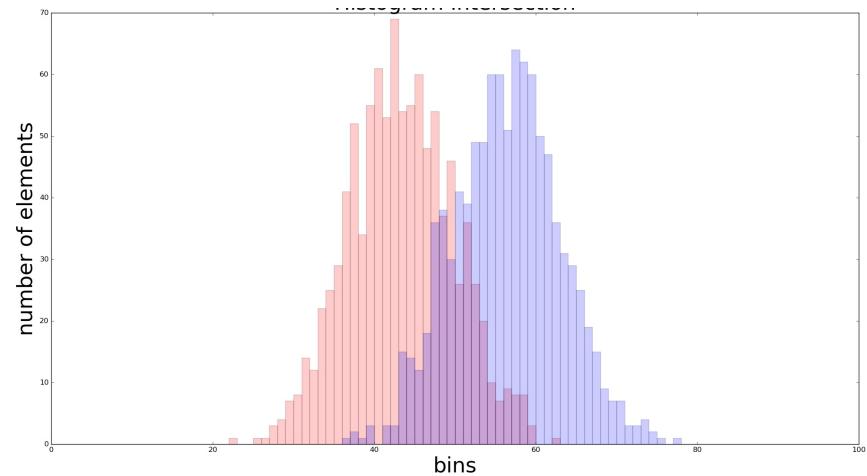
Similarity measures for histograms

(also applicable to SIFT descriptors)

- Euclidean distance
- Histogram intersection

$$HI(H^1, H^2) = \sum_{i=1}^{N_{bin}} (H_i^1, H_i^2)$$

- see also Chi-square, Bhattacharyya distance, ..



Matching strategy



Image 1

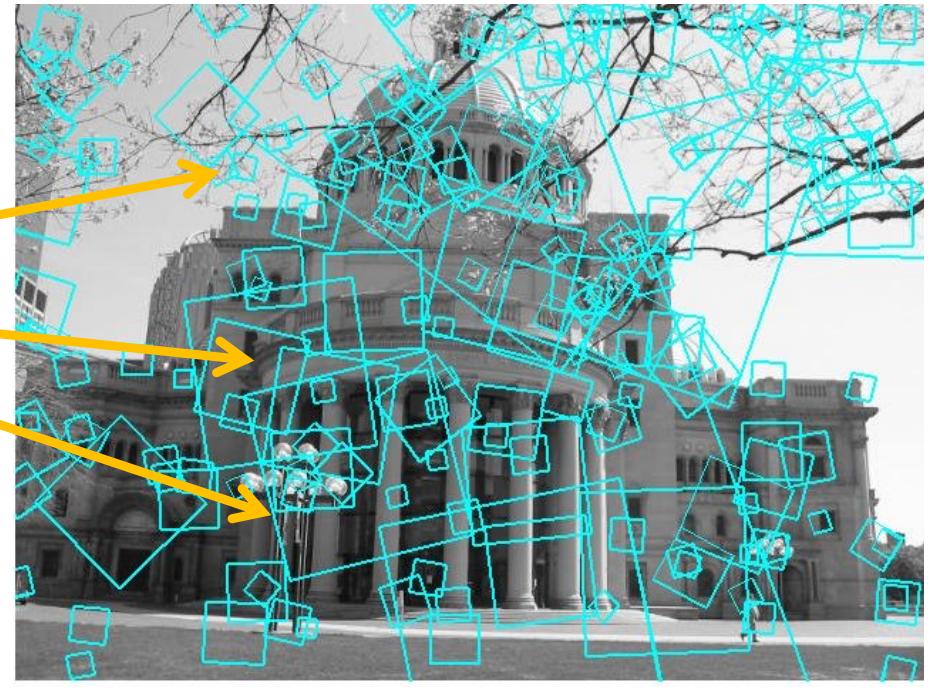


Image 2

To generate **candidate matches**, find features with the most similar appearance

Brute force approach: compare them all, take the closest (or closest k, or within a thresholded distance)

Matching strategy



Image 1



Image 2

To add robustness to matching, consider **ratio** :

$$\text{dist to best match} / \text{dist to second best match}$$

If **low**, first match **looks good**.

If **high**, could be **ambiguous match**.

matching through an affinity matrix

- In graph theory, an affinity matrix is similar to an adjacency matrix E but its entries take values in $[0,1]$
- It describes the similarity between graph nodes and (in feature matching application) between features
- Each entry measures how similar two keypoints are

matching through an affinity matrix

Simplest case: similarity by vicinity on the image plane

$$E(i, j) = e^{-\frac{\|\mathbf{p}_i^1 - \mathbf{p}_j^2\|^2}{2\sigma^2}}$$

Sigma controls the maximum distance we are willing to consider

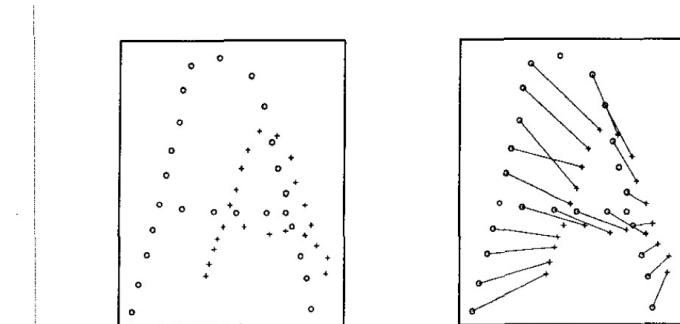
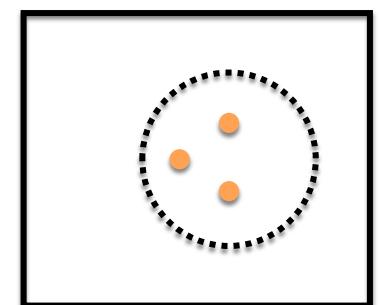
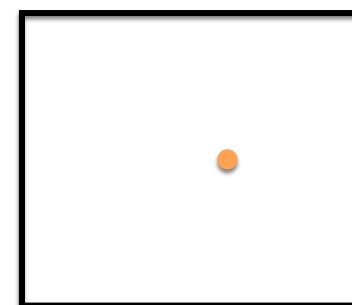


Figure 1: The Scott-Longuet Higgins method applied to two scaled and translated 'A' patterns.



matching through an affinity matrix

We may also incorporate appearance similarity
For instance in the case of patches

$$A(i, j) = E(i, j) \times \frac{1}{2}(\Phi_{NCC}(Q_i^1, Q_j^2) + 1)$$

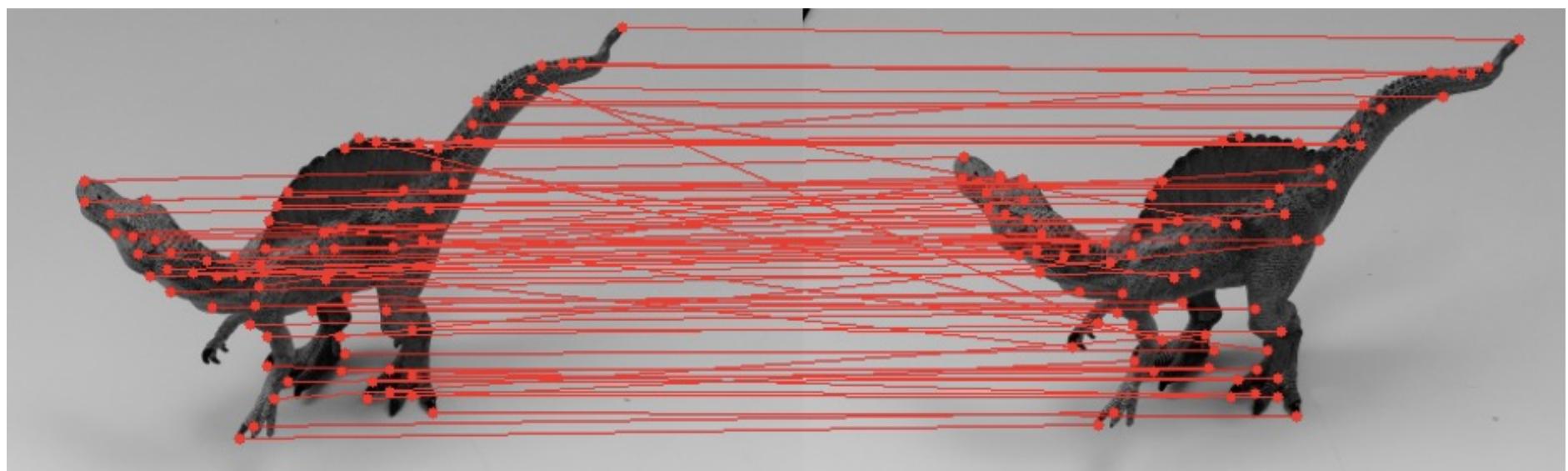
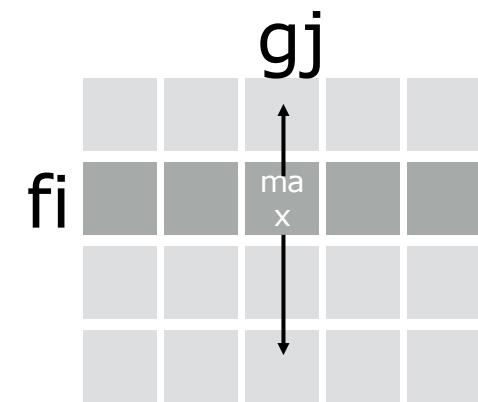
The affinity matrix A has values in [0 1] too

$$\Phi_{NCC}(Q_i^1, Q_j^2) = \sum_{k, l = -\frac{W}{2}}^{\frac{W}{2}} \frac{(Q_i^1(k, l) - \mu_1)(Q_j^2(k, l) - \mu_2)}{W^2 \sigma_1 \sigma_2}$$

matching through an affinity matrix

Finally, we look for one-to-one matches on A:

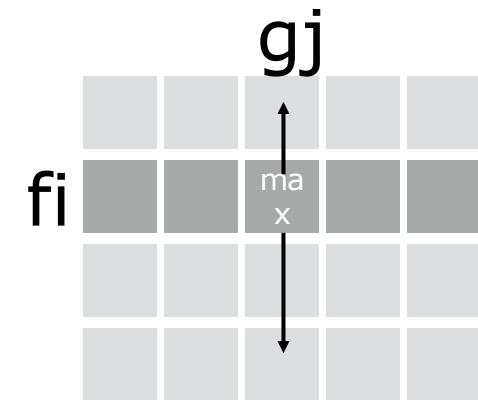
- Look for the maximum of each row of A
- Check that the identified value is the maximum of its column too
- If so, and if the matrix value is above a threshold, a match is detected



matching through an affinity matrix

Finally, we look for one-to-one matches on A:

- Look for the maximum of each row of A
- Check that the identified value is the maximum of its column too
- If so, and if the matrix value is above a threshold, a match is detected



To increase robustness, matches are rejected for those keypoints for which the ratio of the highest value to the second higher value is greater than 0.8.

Feature matching applications

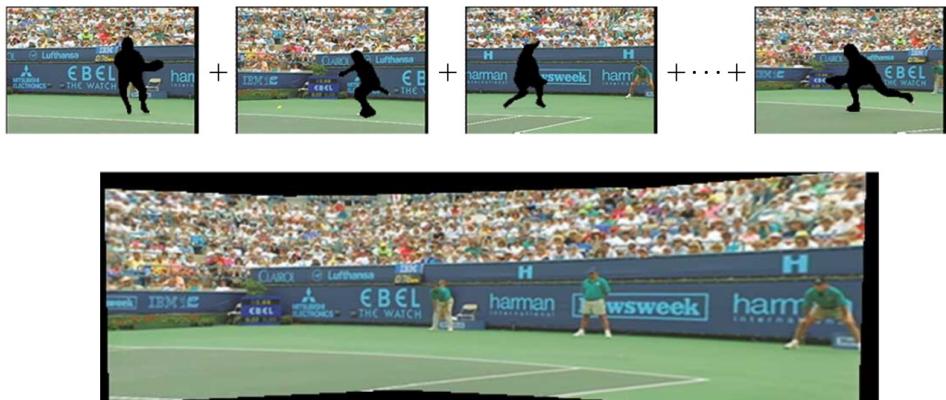


Image stitching or
panorama construction

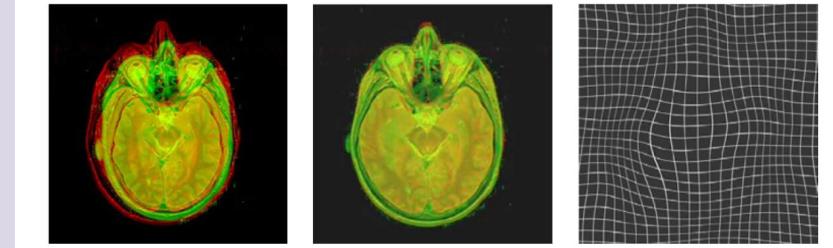
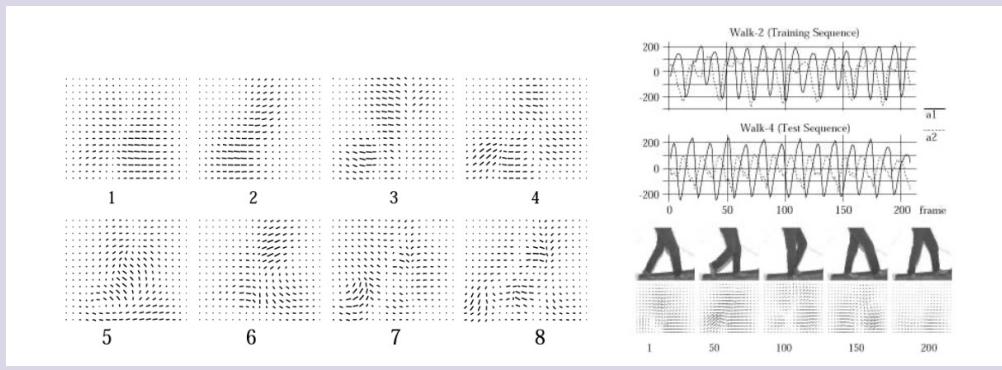


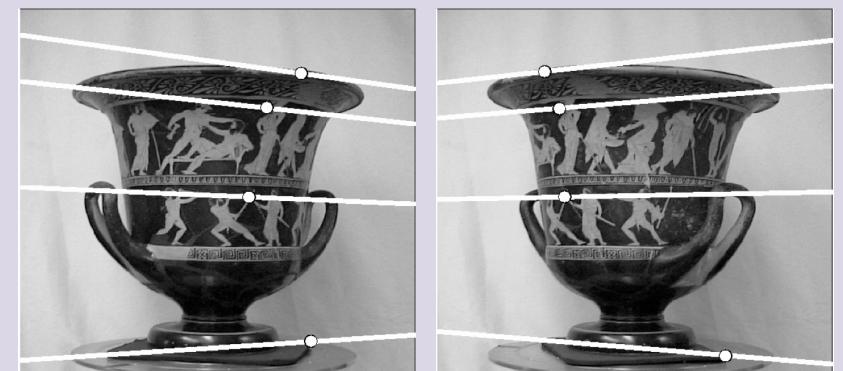
Image registration

Optional reading: Szeliski book (chapter 8)

Next classes



2D motion estimation



stereopsis