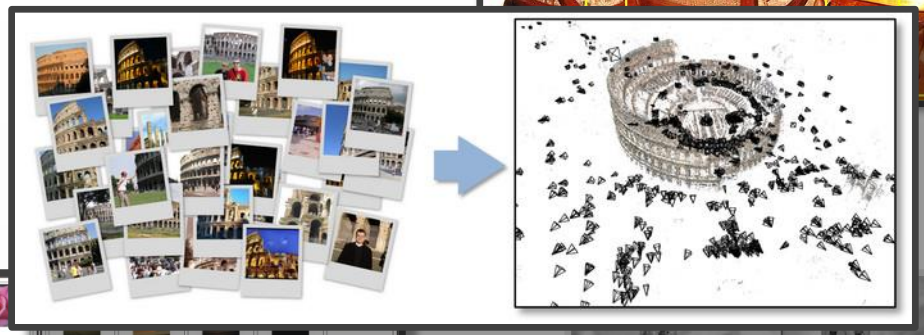
Basic Image Processing Algorithms

PPKE-ITK

Lecture 11.

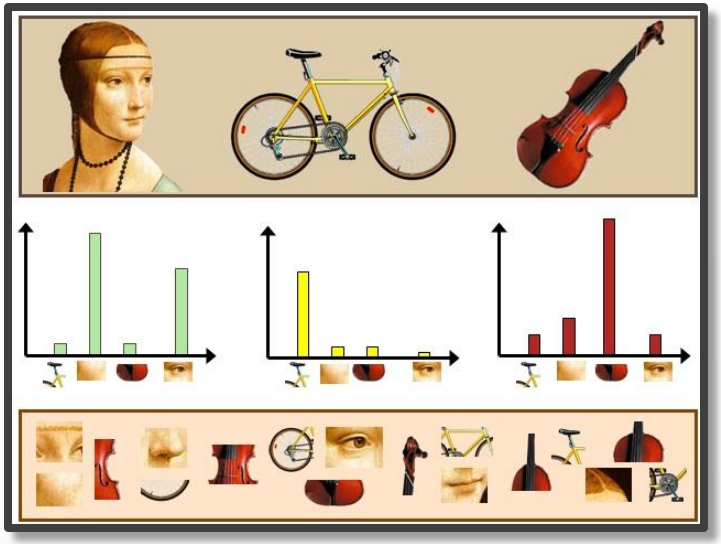
Local Feature Descriptors

- The detection and description of local features has an important role in many applications:
  - Object recognition/detection/tracking
  - Image and video retrieval
  - Image registration, motion estimation
  - Wide baseline matching
  - Texture classification
  - Structure from Motion
  - etc.



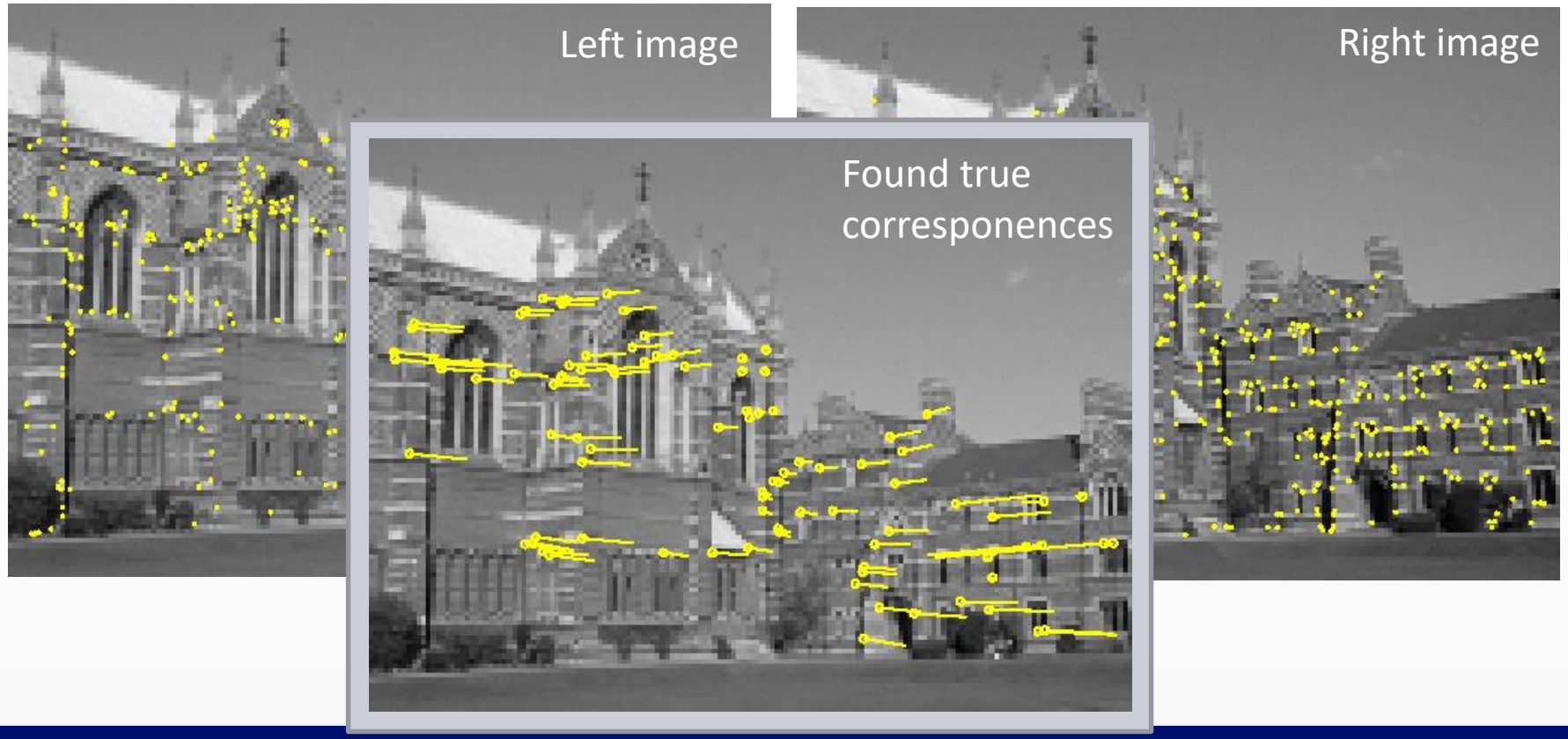
Local Feature Descriptors

- There are different types of use of the descriptors:
  - Description and matching of key points:
    1. Feature/Keypoint detection
    2. Local feature description around the key points
    3. Keypoint matching
  - Bag-of-Features (or bag-of-words)
    1. Feature detection
    2. Feature description
    3. Feature clustering
    4. Frequency histogram construction for image or image part description
  - Description of a specific area:
    1. Find the region of interest (ROI) (e.g. scanning through the image)
    2. Description of the ROI
    3. Classification/Clustering of the ROI descriptor

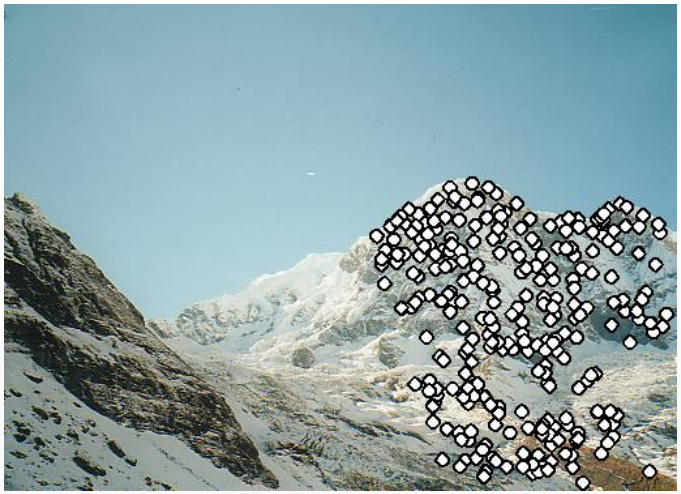


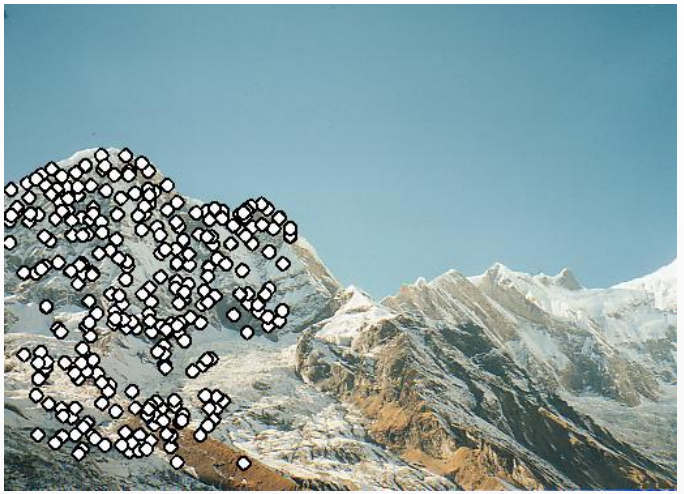
Keypoint matching example

- Application: pixel level image matching from stereo images for depth map calculation



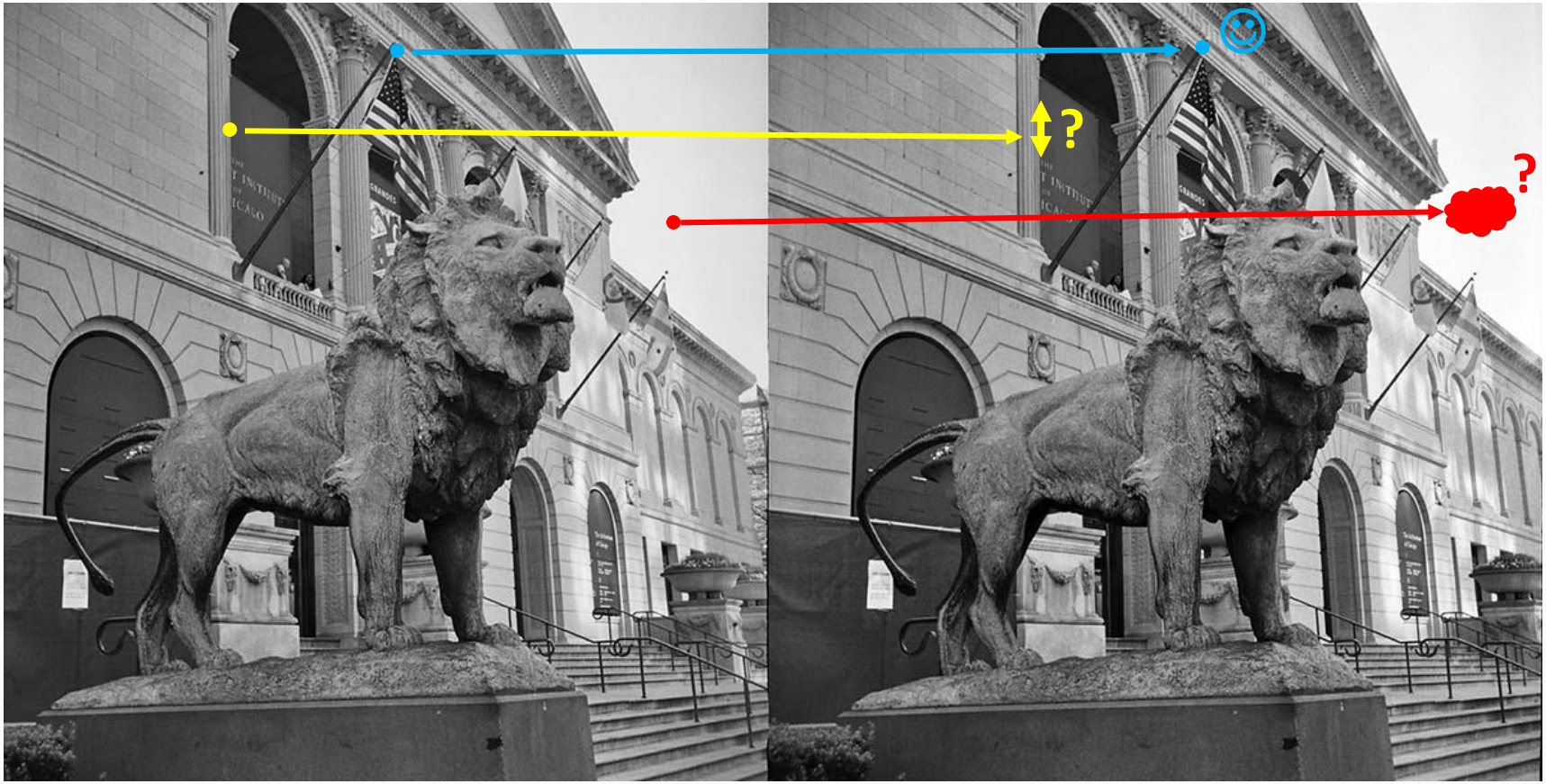
Feature matching for panoramic image creation (results from Matt Brown)



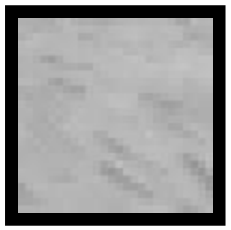


Some Matching Results

Which pixels are easy to match?

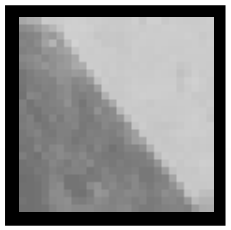


Interest points



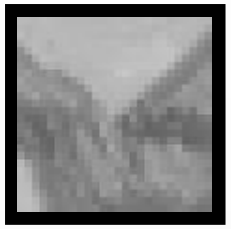
0D structure: **single points**

not useful for matching



1D structure: **lines**

edge, can be localised in 1D,
subject to the aperture problem



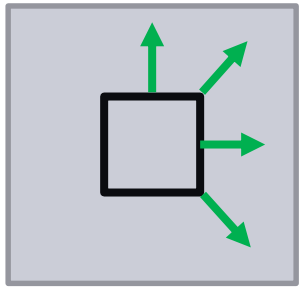
2D structure**:** corners

corner, or **interest point**, can be localized in 2D, good for matching

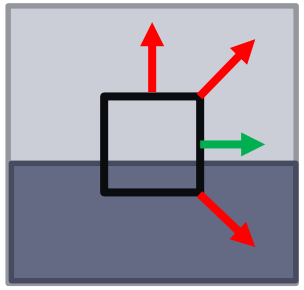
Interest Points have **2D** structure.

How to find good feature points?

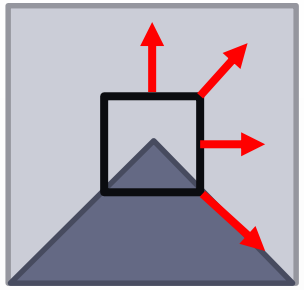
- Based on the idea of auto-correlation
  - Sum of squared differences (SSD) matching



Flat area



Edge area



Corner

- Important difference in all directions
  => interest point

Small SSD

Large SSD

Background: Moravec Corner Detector (1980)

- take a window W in the image
- shift it in four directions

  $[\Delta x, \Delta y] \in \{[1,0], [0,1], [1,1], [-1,1]\}$
- compute a SSD difference for each direction
- compute the *min* difference at each pixel
- *local maxima* in the *min* image are corners

$$\mathrm{SSD}(\Delta x, \Delta y) = \sum_{(x_k, y_k) \in W} \big(I(x_k, y_k) - I(x_k + \Delta x, y_k + \Delta y)\big)^2$$

- Limitation: not isotropic
  - if an edge is present that is not in the direction of the neighbors (horizontal, vertical, or diagonal), then the smallest SSD will be large and the edge will be incorrectly chosen as an interest point.

Harris detector

- Auto-correlation function (SSD) for a point (x, y) and an **arbitrary** shift $(\Delta x, \Delta y)$ - *not only 4 directions*!

$$f(x,y) = \sum_{(x_k,y_k)\in W} \big(I(x_k,y_k) - I(x_k+\Delta x, y_k+\Delta y)\big)^2$$

- Discrete shifts can be avoided with the auto-correlation matrix (Taylor approximation):

$$I(x_k+\Delta x, y_k+\Delta y) = I(x_k,y_k) + [I_x(x_k,y_k) \quad I_y(x_k,y_k)] \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix}$$

  - where I_x and I_y are the derivative images (e.g Prewitt). Then:

$$f(x,y) = \sum_{(x_k,y_k)\in W} \left([I_x(x_k,y_k) \quad I_y(x_k,y_k)] \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix}\right)^2$$

Harris detector

- Rewrite as inner (dot) product

$$f(x,y) = \sum_{(x_k,y_k)\in W} \left(\begin{bmatrix} I_x(x_k,y_k) & I_y(x_k,y_k) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2 =$$

$$= \sum_{(x_k,y_k)\in W} \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \begin{bmatrix} I_x(x_k,y_k) \\ I_y(x_k,y_k) \end{bmatrix} \begin{bmatrix} I_x(x_k,y_k) & I_y(x_k,y_k) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} =$$

- The center portion is a 2x2 matrix:

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

Harris detector

$$= [\Delta x \quad \Delta y] \underbrace{\begin{bmatrix} \sum_{(x_k,y_k)\in W} \left(I_x(x_k,y_k)\right)^2 & \sum_{(x_k,y_k)\in W} I_x(x_k,y_k) I_y(x_k,y_k) \\ \sum_{(x_k,y_k)\in W} I_x(x_k,y_k) I_y(x_k,y_k) & \sum_{(x_k,y_k)\in W} \left(I_y(x_k,y_k)\right)^2 \end{bmatrix}}_{M} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

- M: auto-correlation matrix of the local gradient map
  - captures the structure of the local neighborhood (recap: PCA)
  - measure based on **eigenvalues** λ_1, λ_2 of M
    - 0 strong eigenvalue **(**$\lambda_1 \approx 0, \lambda_2 \approx 0$**)** => uniform region
    - 1 strong eigenvalue **(**$\lambda_1 \gg 0, \lambda_2 \approx 0$**)** => contour
    - **2 strong eigenvalues (**$\lambda_1 \gg 0, \lambda_2 \gg 0$**) => interest point (corner)**
- Interest point detection:
  - threshold on the eigenvalues, then find maximum for localization

Some Details from the Harris Paper

- Alternative measure for corner strength to avoid eigenvalue computation:

$$R = \mathrm{Det}(M) - \kappa\,\mathrm{Tr}^2(M)$$

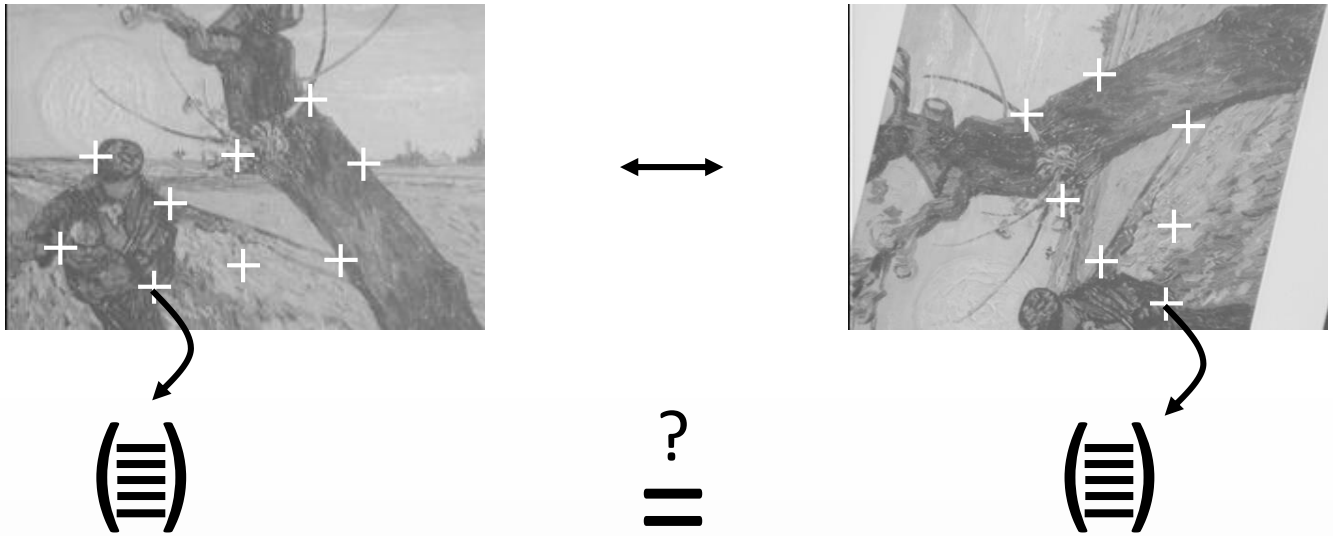
$$M = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \qquad \mathrm{Tr}(M) = a_{11} + a_{22} \qquad \mathrm{Det}(M) = a_{11}a_{22} - a_{12}a_{21}$$

- It can be shown:

$$\mathrm{Tr}^2(M) = \lambda_1 + \lambda_2 \qquad \mathrm{Det}(M) = \lambda_1\lambda_2$$

  - *instead of calculating the* λ_1 and λ_2 eigenvalues, we use trace and determinant (κ parameter is usually between $0.04 - 0.15$)
- Classification based on R:
  - *positive* for corners $(R \gg 0)$,
  - *negative* for edges $(R \ll 0)$,
  - small for flat regions $(R \approx 0)$
- *Select corner pixels that are 8-way local maxima*

Determining correspondences

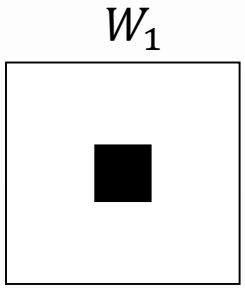


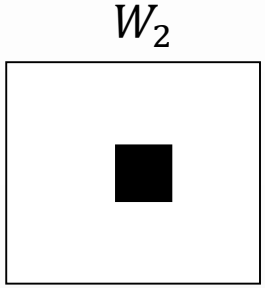
Vector comparison using a distance measure

What are some suitable distance measures?

Distance Measures

- Let W_1 and W_2 be two rectangular windows in image I_1 and I_2 respectively
  - The two windows have same size, but not necessarily the same center positions
- To compare W_1 and W_2 we can use the sum-square difference of the values of the pixels in a square neighborhood about the points being compared. *This is the simplest measure.*





$$\mathrm{SSD}(W_1, W_2) = \sum_{(x_k, y_k)} \left(W_1(x_k, y_k) - W_2(x_k, y_k)\right)^2$$

Harris based feature matching

- Basic feature matching = **Harris Corners & Correlation**
- Very good results in the presence of occlusion and clutter
  - local information
  - discriminant grayvalue information
  - invariance to illumination
- No invariance to scale and affine changes, limited invariance to image rotation
- Solution for more general view point changes
  - local invariant descriptors to scale and rotation
  - extraction of invariant points and regions

Rotation/Scale Invariance


original


translated


rotated

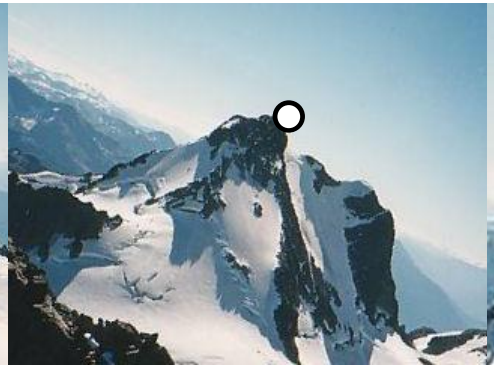

scaled

	Translation	Rotation	Scale
Is Harris invariant?	?	?	?
Is correlation invariant?	?	?	?

Rotation/Scale Invariance


original

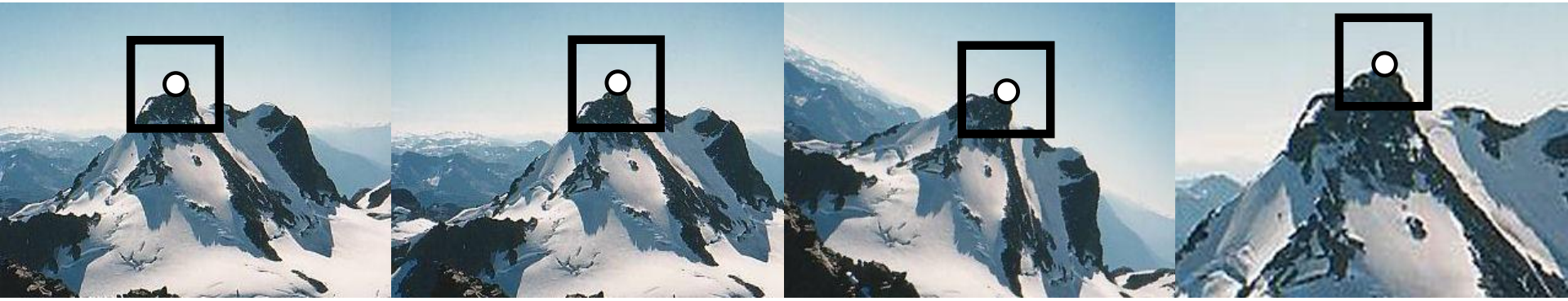

translated


rotated


scaled

	Translation	Rotation	Scale
Is Harris invariant?	YES	YES	NO
Is correlation invariant?	?	?	?

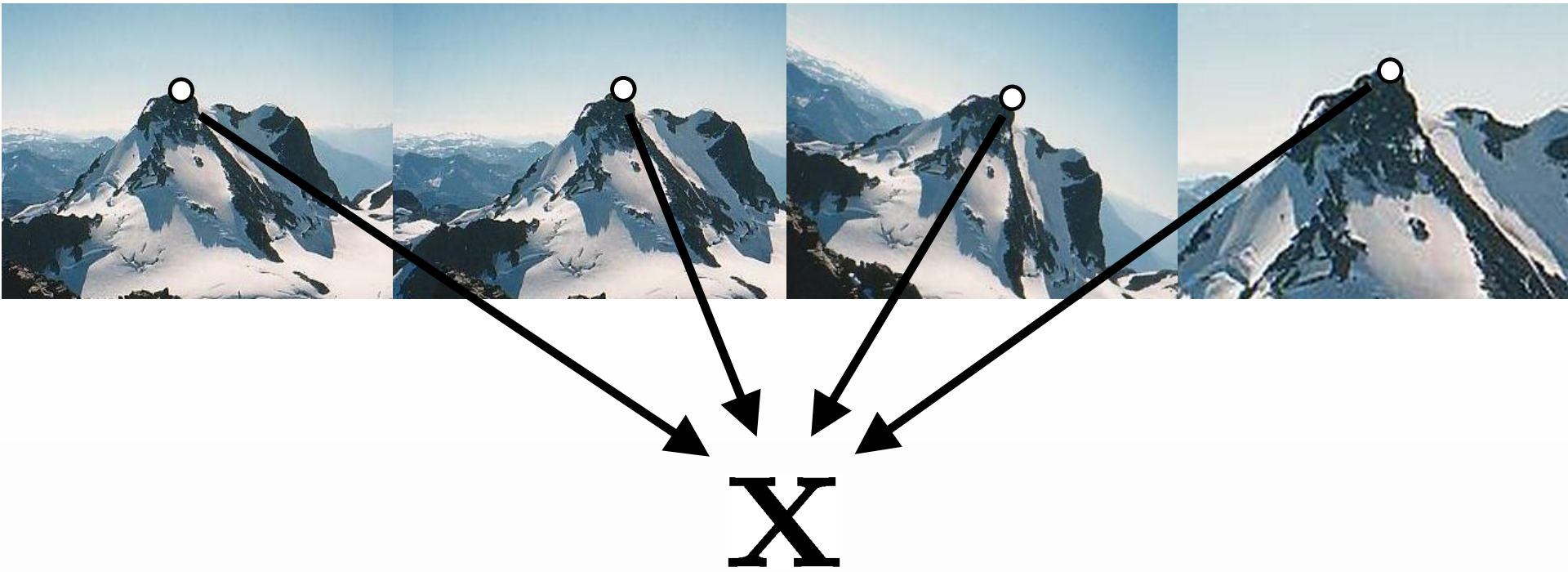
Rotation/Scale Invariance



original translated rotated scaled

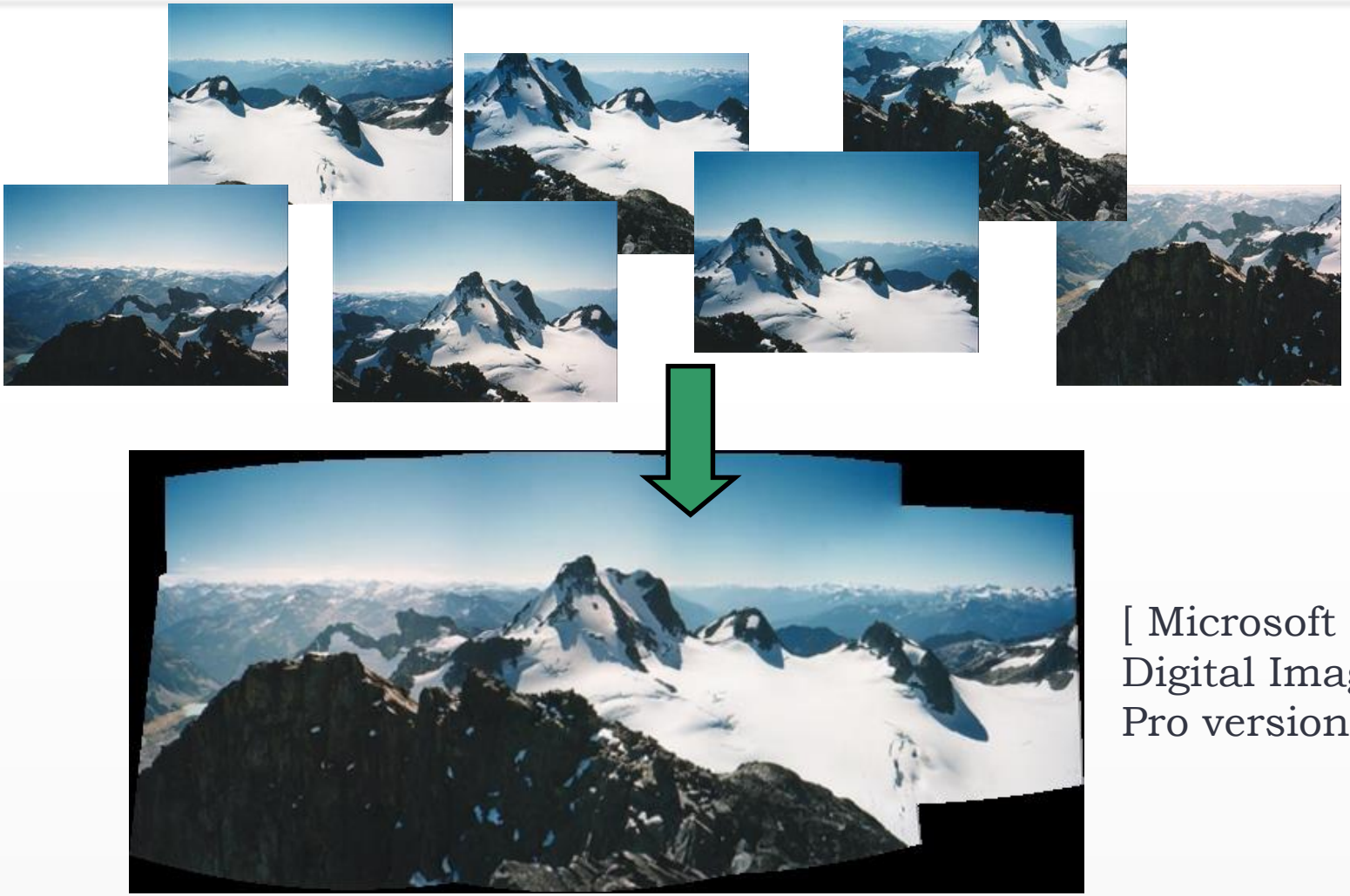
	Translation	Rotation	Scale
Is Harris invariant?	YES	YES	NO
Is correlation invariant?	YES	NO	NO

Matt Brown's Invariant Features



- Local image descriptors that are *invariant* (unchanged) under image transformations

Application: Image Stitching



[Microsoft Digital Image Pro version 10]

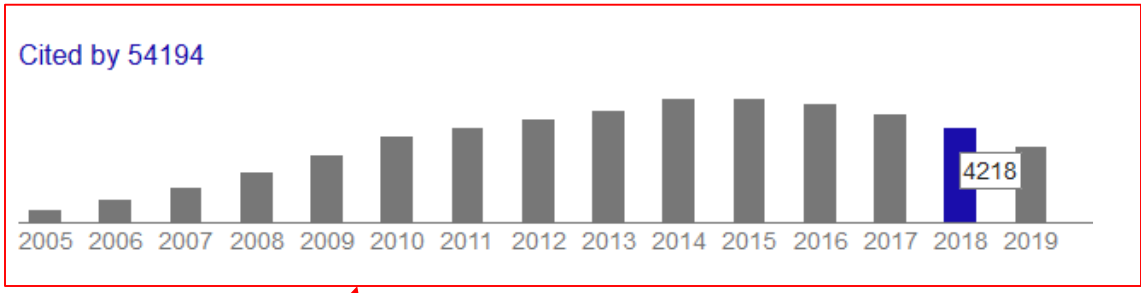
SIFT: Scale-Invariant Image Transform

- Developed by David Lowe, University of British Columbia

David Lowe

Computer Science Dept., University of British Colun
Verified email at cs.ubc.ca - Homepage

Computer Vision Object Recognition



TITLE	CITED BY	YEAR
Distinctive image features from scale-invariant keypoints DG Lowe International journal of computer vision 60 (2), 91-110	54194	2004
Object recognition from local scale-invariant features DG Lowe International Conference on Computer Vision, 1999, 1150-1157	18083	1999

Citation data from 25.11.2019

- Lowe, „*Object recognition from local scale-invariant features*”, *In: IEEE International Conference on Computer Vision*, Vol. 2 (1999), pp. 1150-1157 vol.2.
- Lowe, „ Distinctive image features from scale-invariant keypoints” International journal of computer vision 60 (2), 91-110, 2004

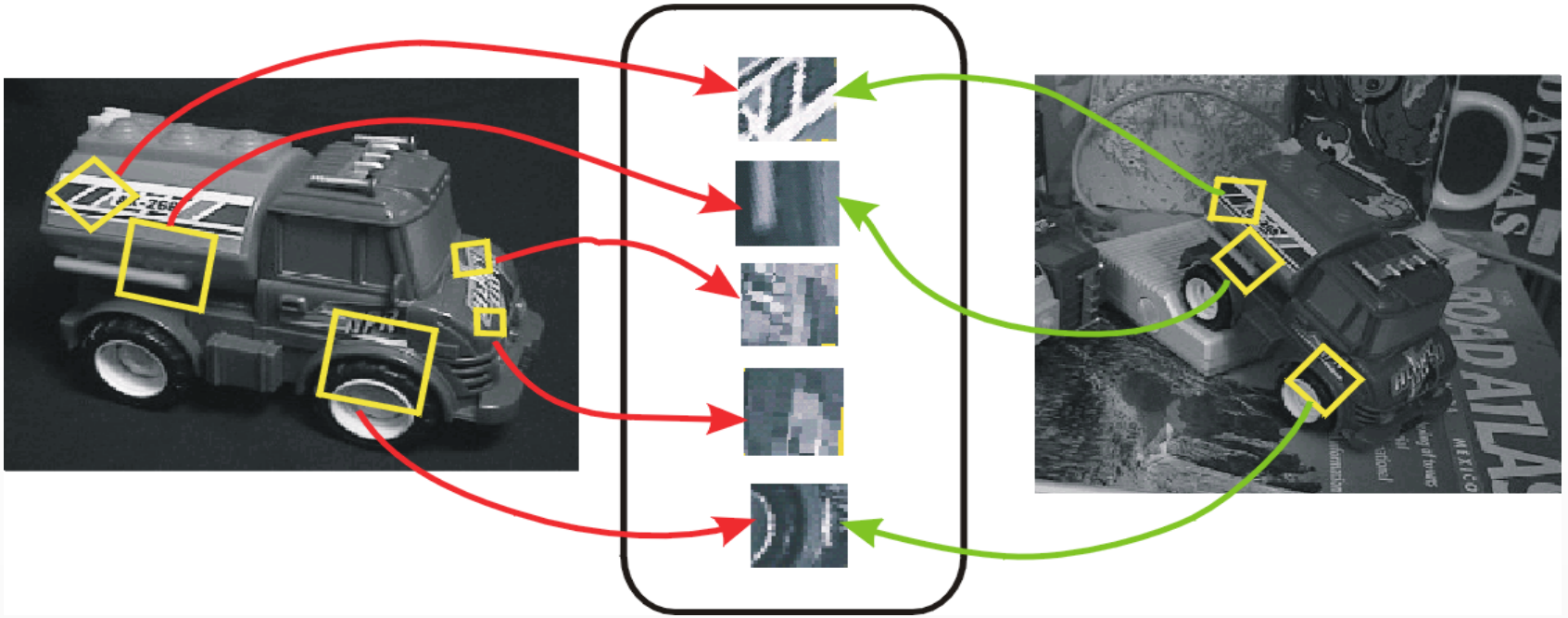
SIFT: Motivation

- The Harris operator is not invariant to scale and correlation is not invariant to rotation[1].
- For better image matching, Lowe's goal was to develop an interest operator that is invariant to scale and rotation.
- Also, Lowe aimed to create a *descriptor* that was robust to the variations corresponding to typical viewing conditions. *The descriptor is the most-used part of SIFT.*

[1]But Schmid and Mohr developed a rotation invariant descriptor for it in 1997.

Idea of SIFT

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



SIFT Features

SIFT: Scale-Invariant Image Transform

- Advantages:
  - Invariant to translation, scaling, and rotation
  - Robust to illumination changes, noise, minor changes in viewpoint
  - Robust to local geometric distortion
  - Highly distinctive
  - SIFT based object detectors are robust to partial occlusion
- Steps of the Algorithm:
  1. Scale-space extrema detection
  2. Keypoint localization
  3. Orientation assignment
  4. Keypoint description

I. Scale-space extrema detection

Scale change via Gaussian blur

- Blurring image with a Gaussian kernel:
  - Loosing details i.e. transforming the image into a different scale

$$L(x, y, k \cdot \sigma) = G(x, y, k \cdot \sigma) * I(x, y)$$

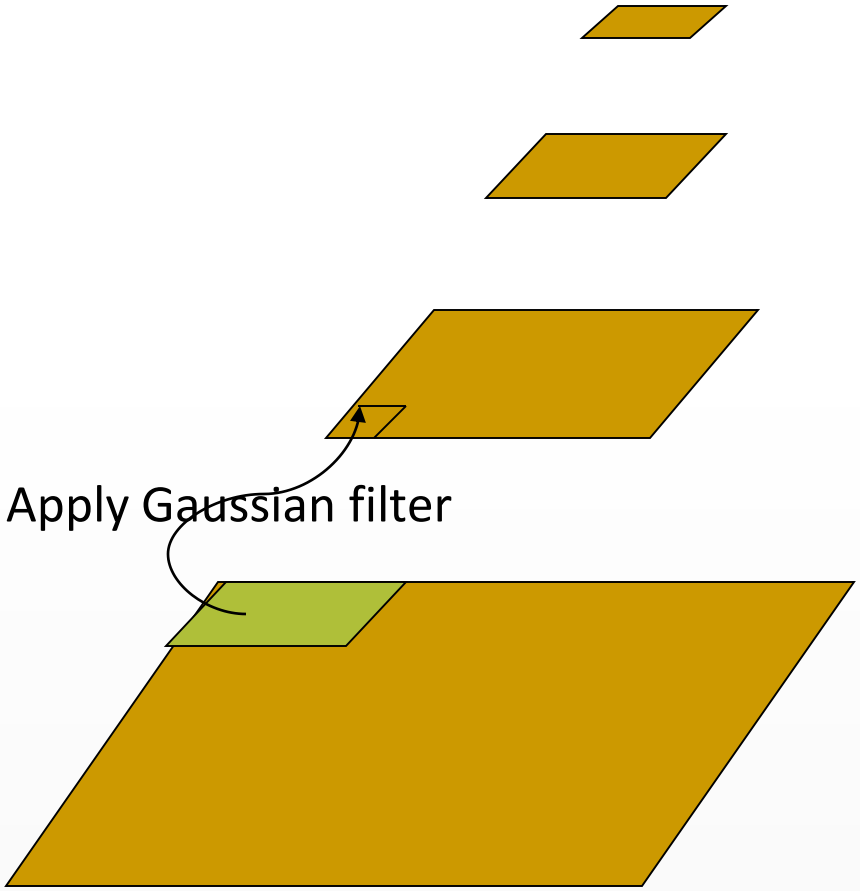
  - Where

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

  - Increasing k will increase the amount of blur, i.e. yields a lower scale representation of the image content
  - At a certain level of blur, the image can be spatially downscaled

Gaussian Pyramid

At each level, image is smoothed and reduced in size.

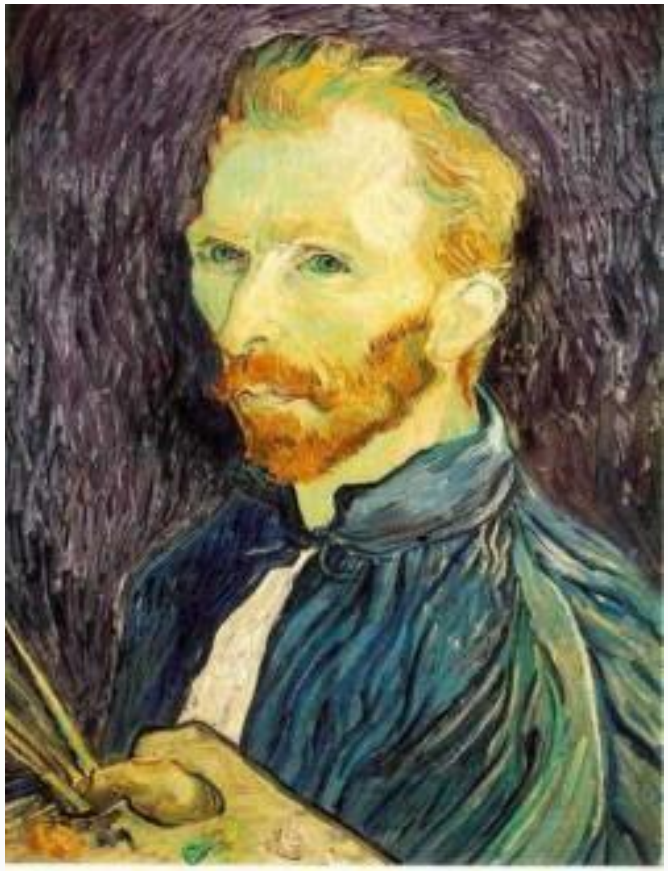


And so on.

At 2nd level, each pixel is the result of applying a Gaussian mask to the first level and then subsampling to reduce the size.

Bottom level is the original image.

Example: Subsampling with Gaussian pre-filtering



Gaussian 1/2

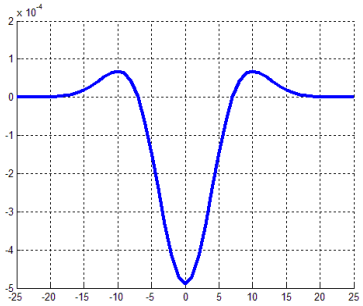
G 1/4

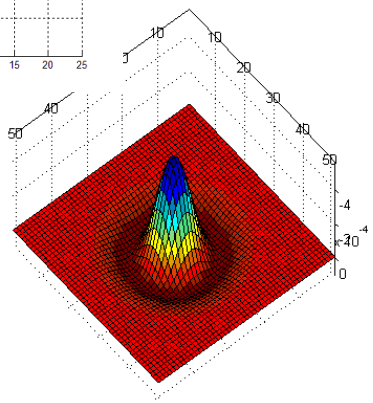
G 1/8

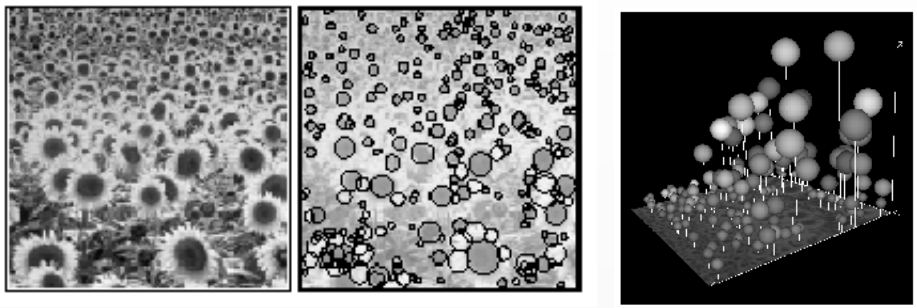
I. Scale-space extrema detection

Lowe's Scale-space Interest Points

- **Laplacian of Gaussian (LoG)** kernel
  - Scale normalised (x by scale^2)
  - Proposed by Lindeberg
- Scale-space detection
  - Find local maxima across scale/space
  - A good "blob" detector





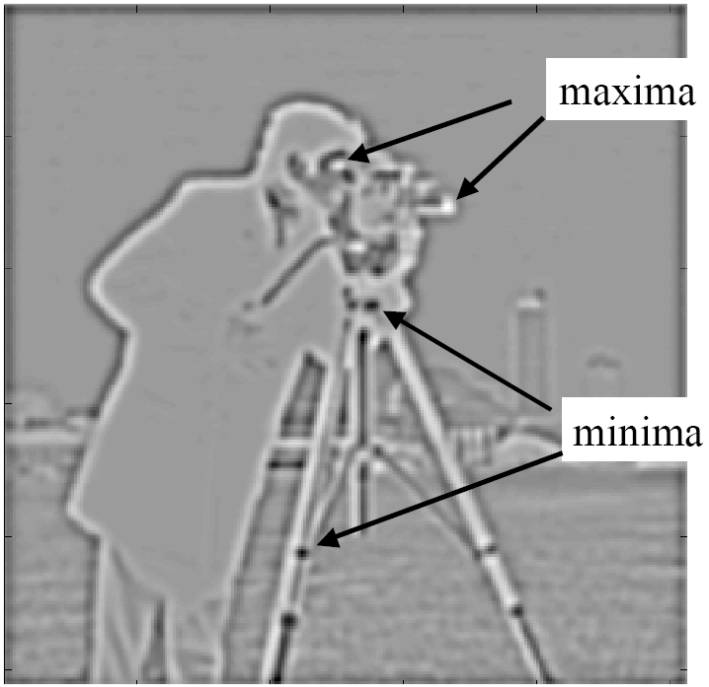


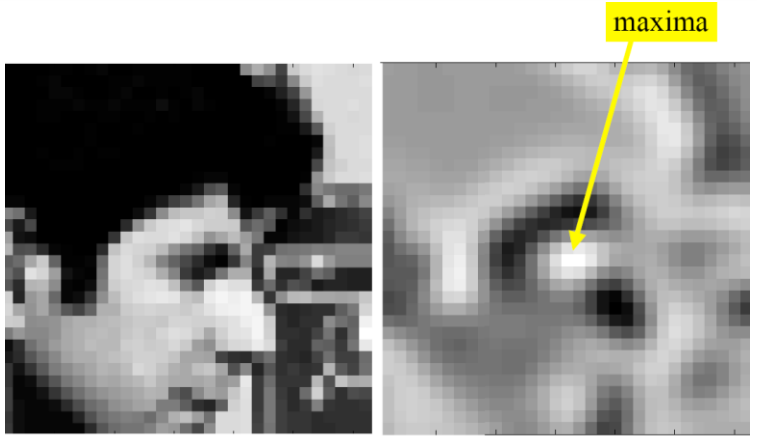
[T. Lindeberg IJCV 1998]

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

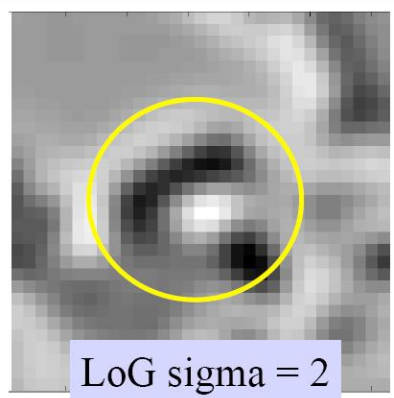
$$\nabla^2 G(x, y, \sigma) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}$$

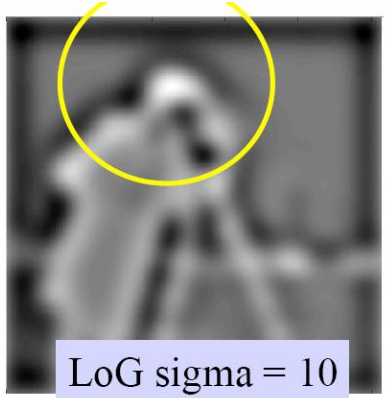
LoG extrema examples



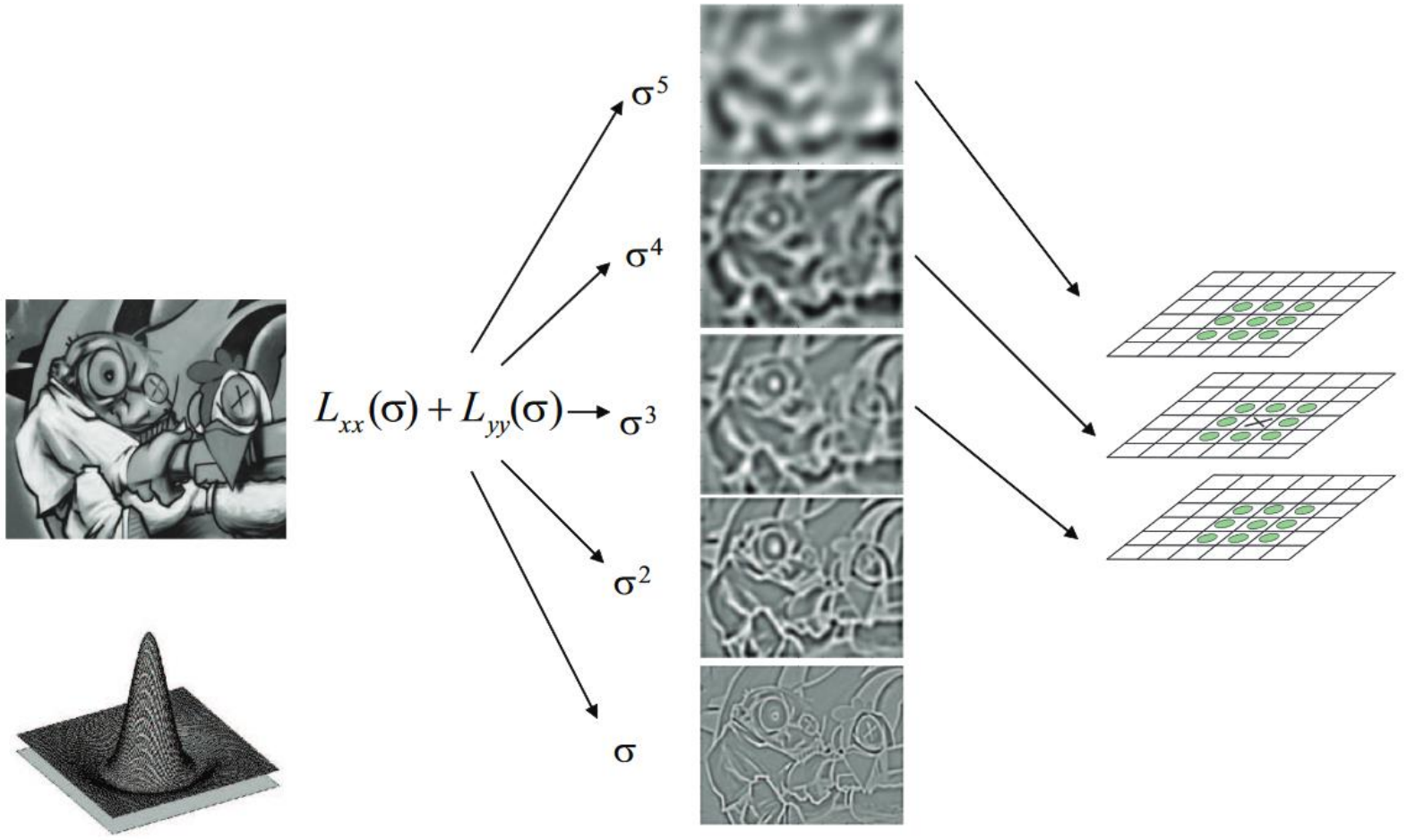


- Dependence on sigma of blurring (i.e. scale):





LoG as blob detector



I. Scale-space extrema detection

Lowe's Scale-space Interest Points

- Using Laplacian of Gaussian (LoG) directly - $\sigma^2 \nabla^2 G$
  - Extrema useful: found stable feature and gives excellent notion of scale
  - Calculation costly instead…
- Approximation of LoG:

$$\sigma \nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma}$$

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k - 1)\sigma^2 \nabla^2 G$$

  - Difference of Gaussians (DoG) approximates LoG:

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y) =$$
$$= L(x, y, k\sigma) - L(x, y, \sigma)$$

I. Scale-space extrema detection
Workflow

- I. Scale-space extrema detection:
  - Key point detection with ***Difference of Gaussians*** (DoG):
    - $I(x, y)$ is the original image
    - $G(x, y, k\sigma)$ is the Gaussian blur at scale $k\sigma$
    - The original image convolved with Gaussian kernel at different scales:

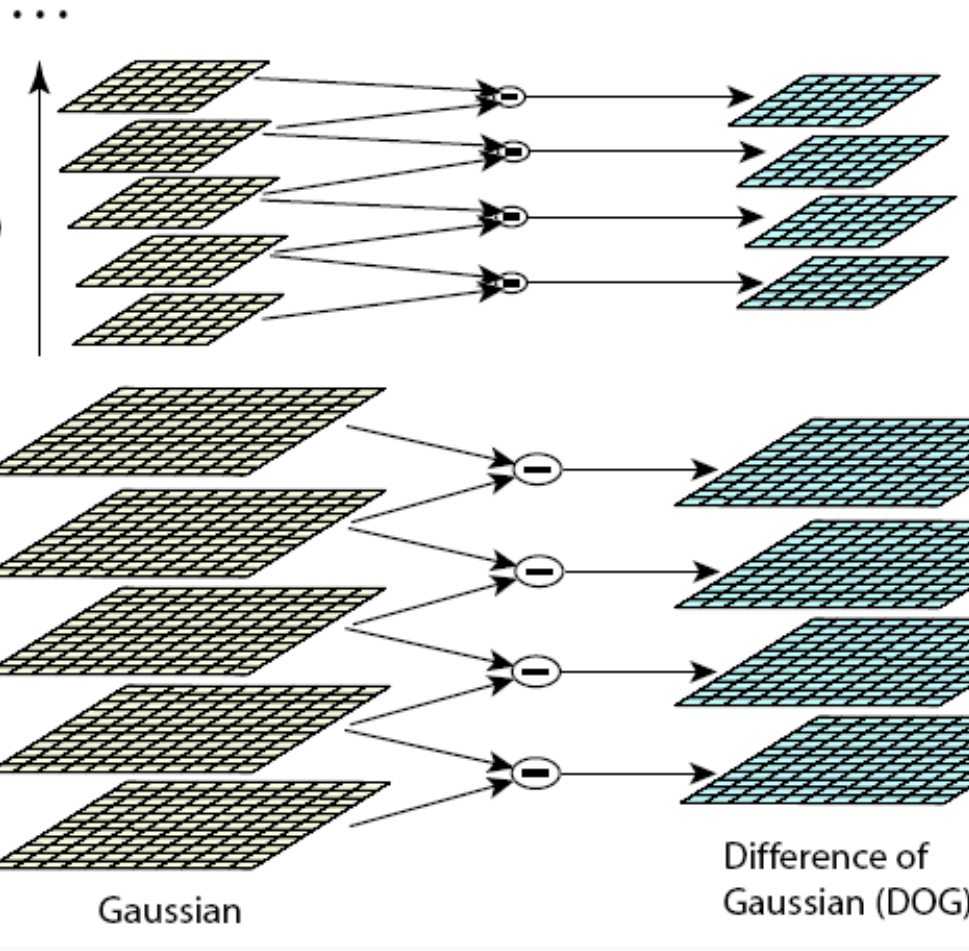
$$L(x, y, k \cdot \sigma) = G(x, y, k \cdot \sigma) * I(x, y)$$

    - The convolved images are grouped by octave (in an octave σ is doubled). The difference of consecutive convolved images is taken in an octave:

$$D(x, y, \sigma) = L(x, y, k_i \cdot \sigma) - L(x, y, k_j \cdot \sigma)$$

I. Scale-space extrema detection

Lowe's Pyramid Scheme



s+2 filters

$\sigma_{s+1}=2^{(s+1)/s}\sigma_0$

.

.

$\sigma_i=2^{i/s}\sigma_0$

.

.

$\sigma_2=2^{2/s}\sigma_0$

$\sigma_1=2^{1/s}\sigma_0$

σ_0

s+3 images including original

s+2 differ-ence images

The parameter **s** determines the number of images per octave.

I. Scale-space extrema detection

Lowe's Pyramid Scheme

- Scale space is separated into **octaves**:
  - Octave 1 uses scale σ
  - Octave 2 uses scale 2σ
  - etc.

- In each octave, the initial image is repeatedly convolved with Gaussians to produce a set of scale space images.

- Adjacent Gaussians are subtracted to produce the DOG

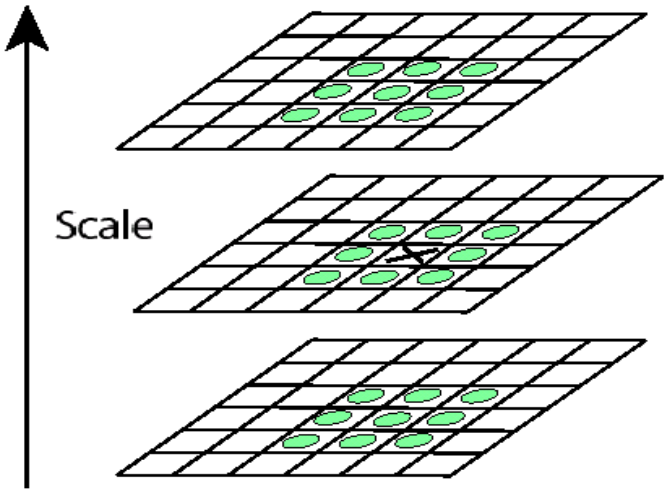
- After each octave, the Gaussian image is down-sampled by a factor of 2 to produce an image ¼ the size to start the next level.

II. Key point localization

- Detect maxima and minima of difference-of-Gaussian in scale space

- Each point is compared to its 8 neighbors in the current image and 9 neighbors each in the scales above and below

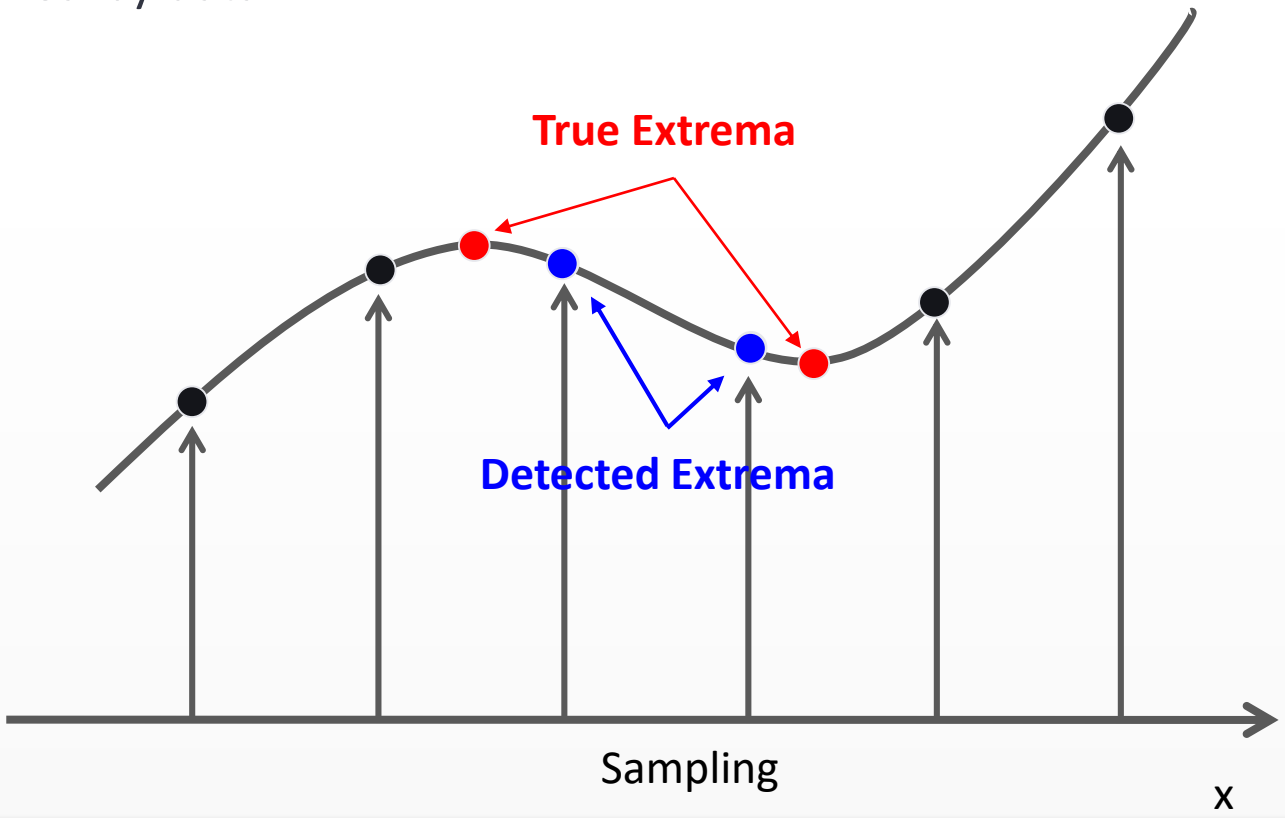
s+2 difference images.
top and bottom ignored.
s planes searched.



For each max or min found, output is the **location** and the **scale**.

II. Key point localization

- II. Keypoint localization:
  - Localization is done with sub pixel accuracy, based on the interpolation of nearby data:



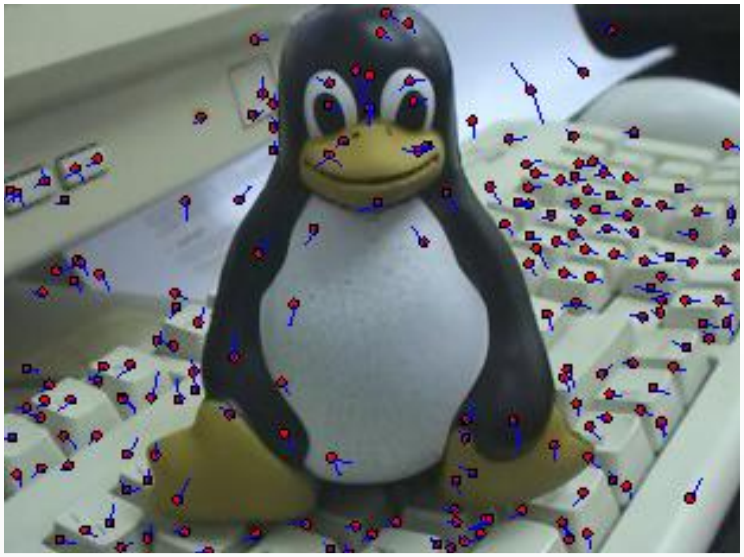
II. Key point localization

- II. Keypoint localization:
  - Rejection of weak candidates:
    - Low contrasted points
    - Poorly localized points along edges:
      - The DoG function will have strong responses along edges, but these points are not stable, since their location is poorly defined.
      - These points will be removed based on the principal curvature across and along the edge.
  - Similar approach to Harris, but here the ratio of the trace^2 and determinant of the Hessian detector matrix (Beaudet, 1978) is calculated

http://en.wikipedia.org/wiki/Scale-invariant_feature_transform

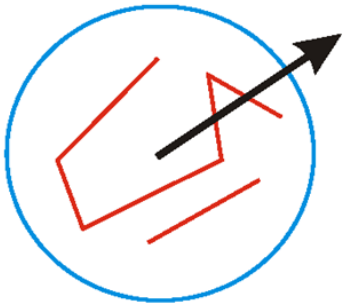
III. Orientation Assignment

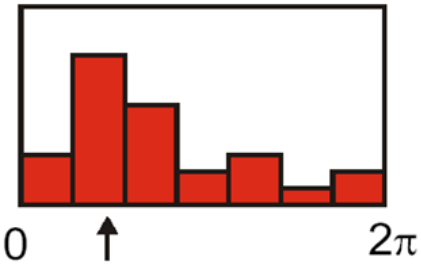
- Orientation Assignment: goal is to ensure rotation invariance
  - Find the main orientation(s)
  - Assign it to the key point and
  - Give the description of the keypoint relative to this orientation.



III. Orientation Assignment

- Steps of orientation assignment
  - ***Gaussian smoothed*** image is taken at the scale of the keypoint.
  - The ***edge magnitude and orientation*** is calculated ***for each point*** in the neighborhood.
  - A ***36 bin orientation histogram*** is composed, where each bin represents a 10 degree interval, and each neighboring point's bin is determined based on its edge orientation and its weight based on the edge magnitude.





  - Also the points are ***weighted with a Gaussian*** window, so the points farther away has less effect than the points closer to the keypoint.
  - The canonical orientation of the keypoint will correspond to the peak of the histogram.

Result of Keypoint localization with orientation assignment

233x189

832

initial keypoints

729

keypoints after low contrast based rejection

536

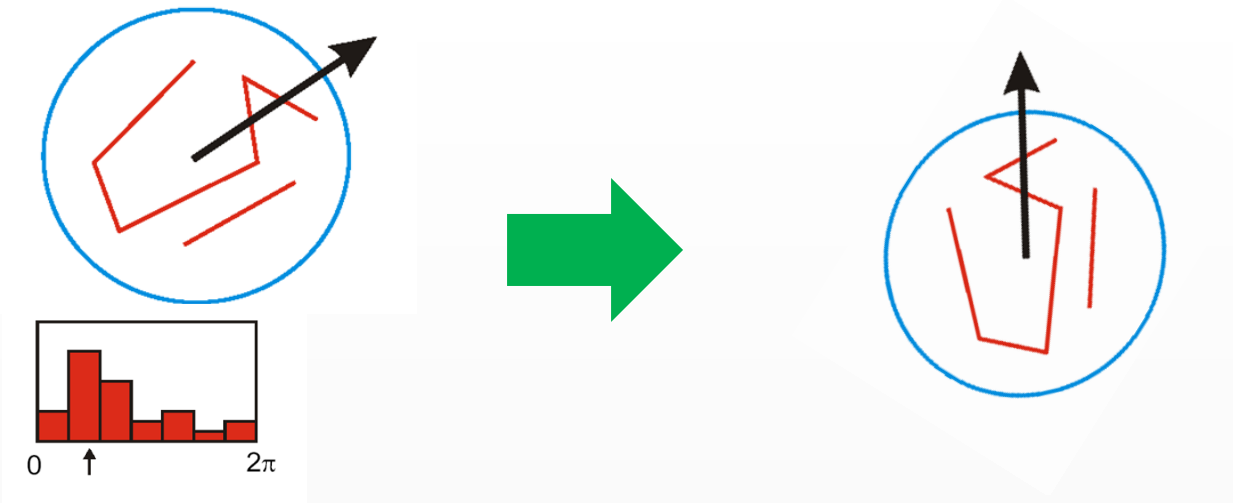
keypoints after ratio threshold

IV. Keypoint Descriptors

- At this point, each keypoint has
  - location
  - scale
  - orientation
- Next is to compute a descriptor for the local image region about each keypoint that is
  - highly distinctive
  - invariant as possible to variations such as changes in viewpoint and illumination

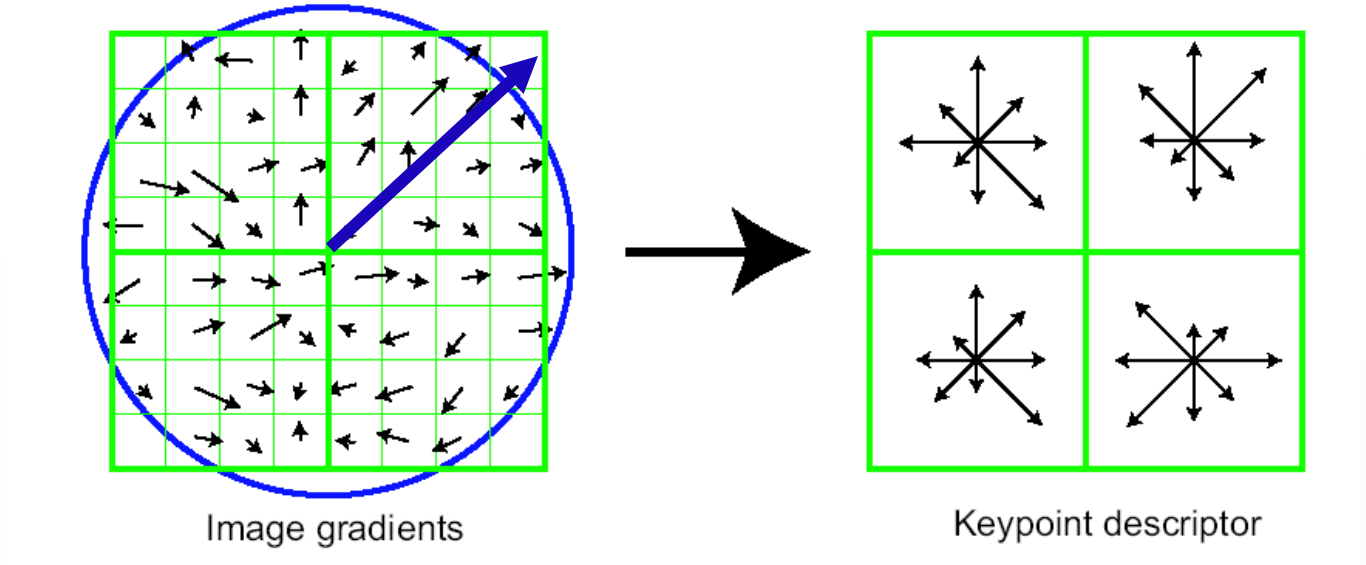
Normalization

- Rotate the window to standard orientation (e.g. the calculated canonical orientation vector should point upwards)
- Scale the window size based on the scale at which the point was found.



Lowe's Keypoint Descriptor (shown with 2 X 2 descriptors over 8 X 8)

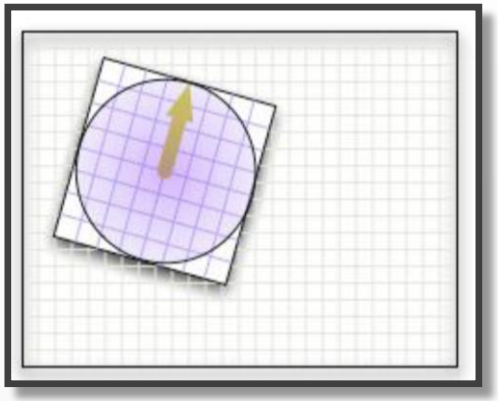
- *Demonstration example*: take here a 8x8 point neighborhood around the keypoint and divide it into 2x2 gradient window.
- Build the orientation histogram of the 2x2 samples in each window with 8 direction bins – concatenate the 8bin histograms to obtain feature vector.



- In practice: 4x4 arrays (form 16x16 point neigborhoods), with 8 bin histogram is used, a total of 4x4x8=128 features for one keypoint

IV. Keypoint Descriptor: Overview

- Use the normalized region about the keypoint
- Take a 16x16 point neighborhood around the keypoint and divide it into 4x4 gradient window.
- Compute gradient magnitude and orientation at each point in the region (weight them by a Gaussian window overlaid on the circle)
- Build the orientation histogram of the 4x4 samples in each window with 8 direction bins.
- 4 X 4 times 8 directions gives a vector of 128 values. 



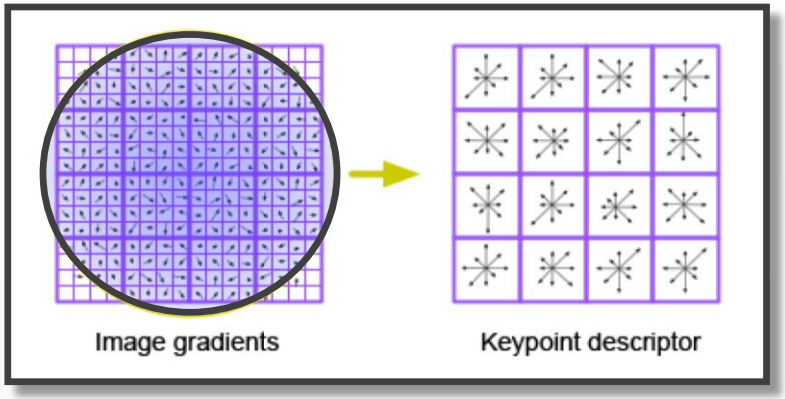
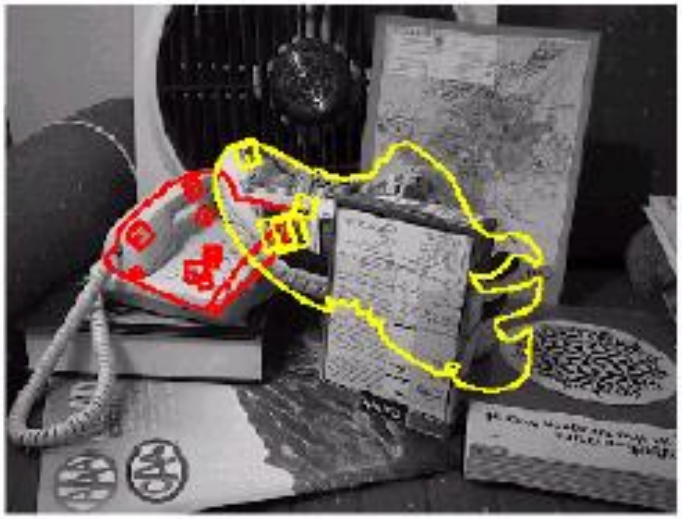
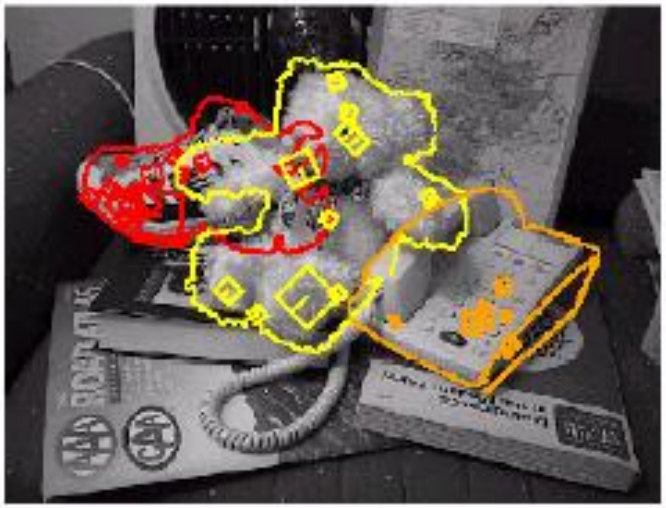


Image from: Ofir Pele

Using SIFT for Matching “Objects”





SIFT: Scale-Invariant Image Transform

- SIFT inspired methods:
  - PCA-SIFT:
    - Reduce dimensionality, only keeps 20 dimension out of 128.
  - SURF:
    - Inspired by SIFT, but uses box filters (Haar like filters) with Integral Image implementation for fast calculation.
    - Has similar results as SIFT, but more sensitive to viewpoint and illumination changes.
  - …

Y. Ke and R. Sukthankar, "*PCA-SIFT: A More Distinctive Representation for Local Image Descriptors*," Proc. Conf. Computer Vision and Pattern Recognition, pp. 511-517, 2004.
H. Bay, T. Tuytelaars, L. Van Gool "*SURF: Speeded Up Robust Features*", Proceedings of the 9th European Conference on Computer Vision, Springer LNCS volume 3951, part 1, pp 404--417, 2006.