Summer School of Information Engineering

Image and video analysis: feature descriptors and their applications

Pietro Zanuttigh

Outline

1. Feature detectors and descriptors
2. The SIFT descriptor
3. Matching and recognition with the SIFT descriptor
4. Overview of recent descriptors
5. 3D reconstruction from image features

For some slides and pictures thanks to:
D. Lowe, A. Vedaldi,T. Lindeberg,J. Matas,
K. Mikolajczyk, R.Gonzalez, R. Woods and others…


Part 1: Introduction to feature detection

Feature detection

Extract relevant features from the images

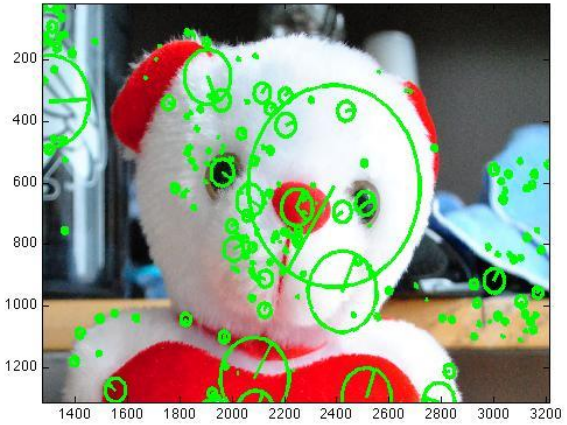
- Lines
- Edges
- Corners
- Other…

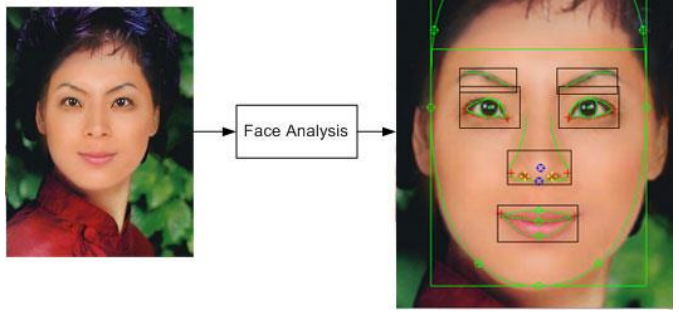
Two key operations:

- ***Feature detection***: extract the features of interest
- ***Feature description***: associate a descriptor to each feature in order to distinguish from the others

- A very good tutorial:
  - Modern features: advances, applications, and software, Andrea Vedaldi, Jiri Matas, Krystian Mikolajczyk, Tinne Tuytelaars , Cordelia Schmid, Andrew Zisserman*, ECCV2012*
  - *Some slides and pictures have been derived from this tutorial*

Features





- **Definition**: A feature **detector** (extractor) is an algorithm taking an image as input and outputting a set of regions (“local features”).
- “Local Features” are **regions,** i.e. in principle arbitrary sets of pixels, usually contiguous, which are at least :
  - **distinguishable** in an image regardless of viewpoint/illumination/scale/affine transforms…
  - **robust to occlusion:** must be **local**
  - Must have a discriminative neighborhood: they are **“features”**
- Local Feature = Interest “Point” = Keypoint = Feature “Point”

- A **descriptor** is computed on an image region defined by a detector. The descriptor is a representation of the image function (colour, ….) in the region.

Derived from Vedaldi et Al, «modern features»

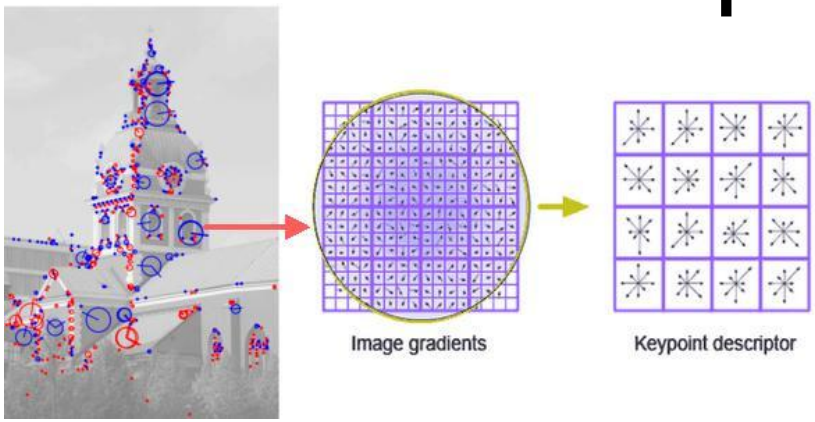
Objectives

- **Invariance** (or covariance) to a broad class of geometric and photometric transforms
- **Efficiency**: close to real-time performance
- **Quantity/Density** of features to cover small object/part of scenes
- **Robustness** to:
  - occlusion and clutter (requires *locality)*
  - to noise, blur, discretization, compression
- **Distinctiveness:** individual features can be matched to a large database of objects
- **Stability over time** (to support long-temporal-baseline matching -> video)
- **Geometrical accuracy**: precise localization
- **Generalization** to similar objects

- No detector dominates in all aspects, some properties are competing, e.g. level of invariance x speed

Derived from Vedaldi et Al, «modern features»

Feature Descriptor



- **Definition**: A descriptor is computed on an image region defined by a detector. The descriptor is a representation of the intensity/colour function in the region
- **Objectives** :
  - Discriminability
  - Robustness to misalignment, illumination, blur, compression, …
  - Efficiency: real-time often required
  - Compactness: small memory footprint.

Applications

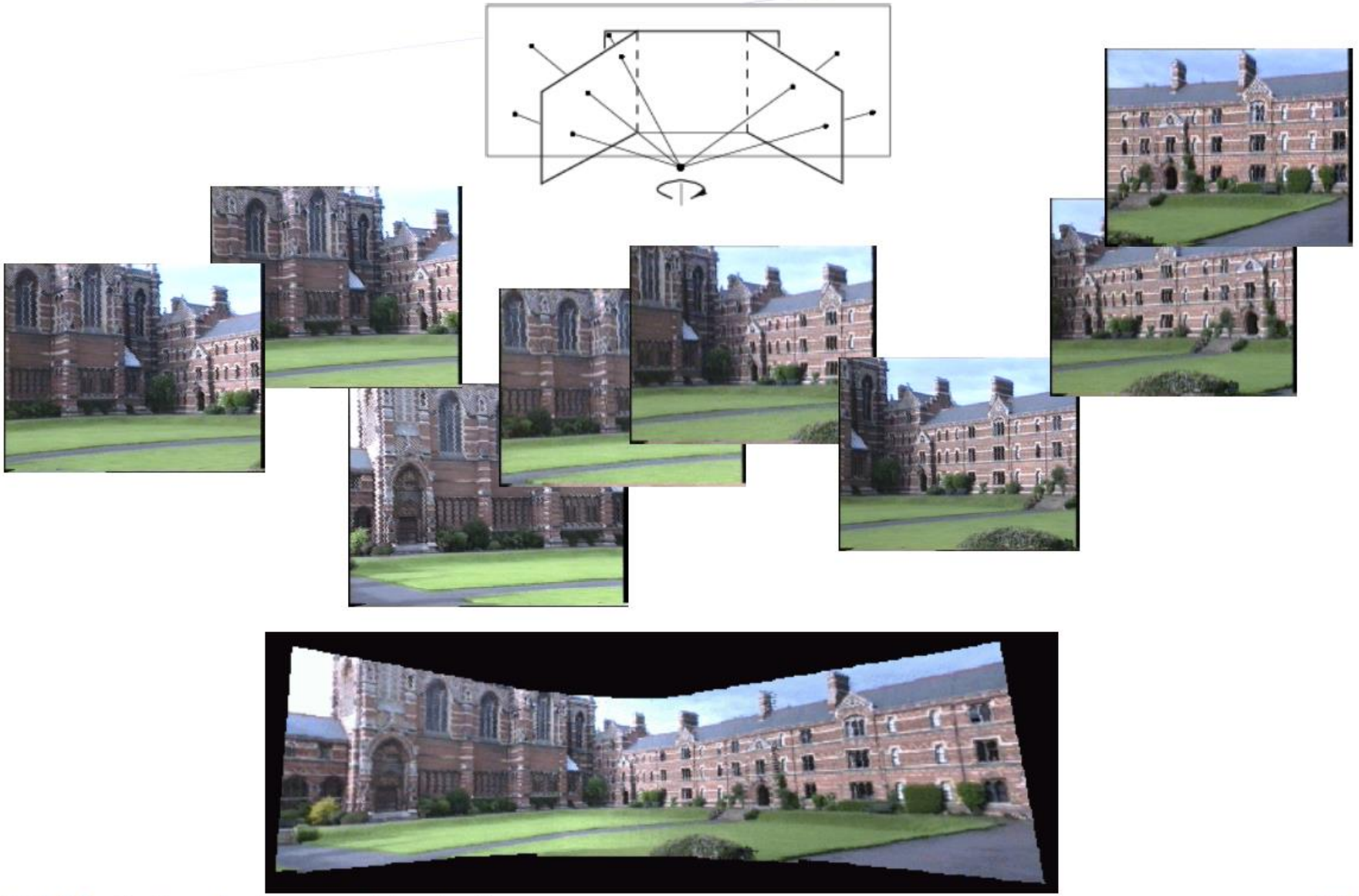
- Methods based on “Local Features” are the state-of-the-art for many computer vision problems
- Suited to instance matching over change in viewpoint, scale, lighting, partial occlusion, region of interest …
- Multiple views of the same scene, e.g.
  - Computing epipolar geometry or a homography
  - Photo Tourism
  - Panoramic mosaic
- Query by example search in large scale image datasets
- Copy detection
- Re-acquisition in tracking
- Object category recognition

Example 1: Wide baseline matching

- Establish correspondence between two (or more) images
- Useful in visual geometry: Camera calibration, 3D reconstruction, Structure and motion estimation, …

Local transf: scale/affine – Detector: affine-Harris Descriptor: SIFT

Example 2: Panoramic mosaic



Example 2.Image Stitching: Building a Panorama

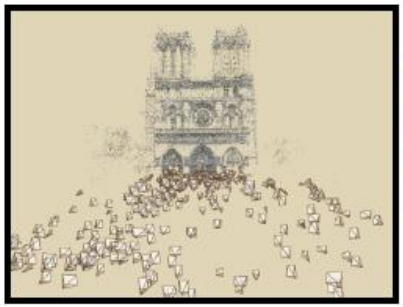
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

Example 3: 3D reconstruction

- Photo Tourism overview

Input photographs

Scene reconstruction



Relative camera positions and orientations

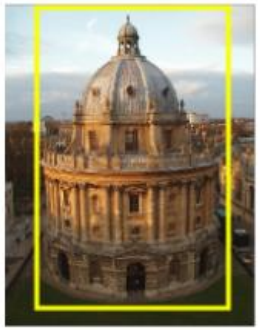
Point cloud

Sparse correspondence

Photo Explorer

Video

Example 4: Query by example search in large scale image datasets

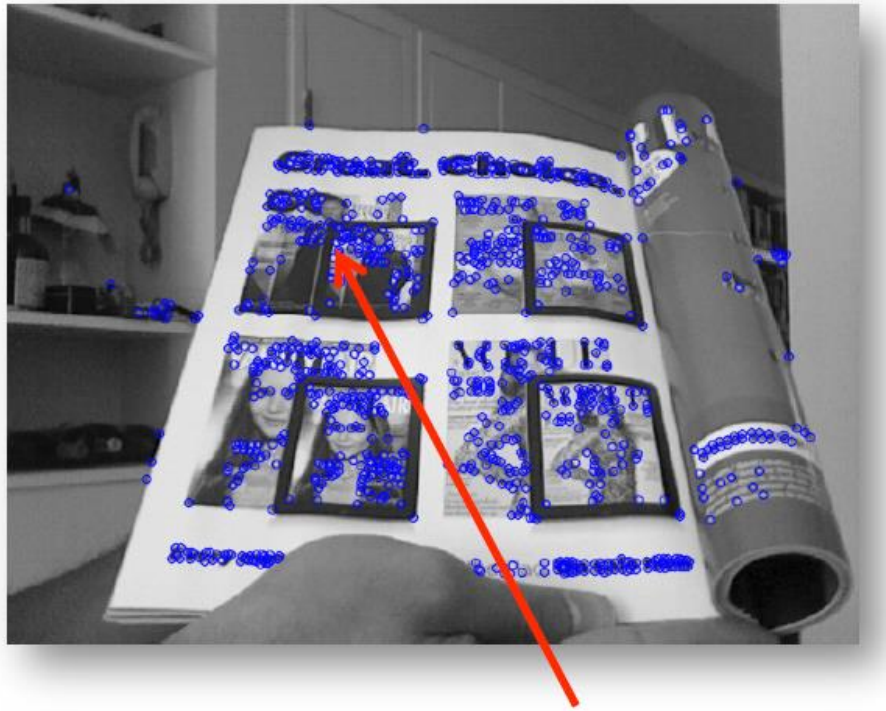


Find these objects

...in these images and 1M more

Search the web with a visual query ...

Example 8: Re-acquisition in tracking



Weight vector $\mathbf{w}_i$ per keypoint

Descriptor $\mathbf{d}_j$ per keypoint

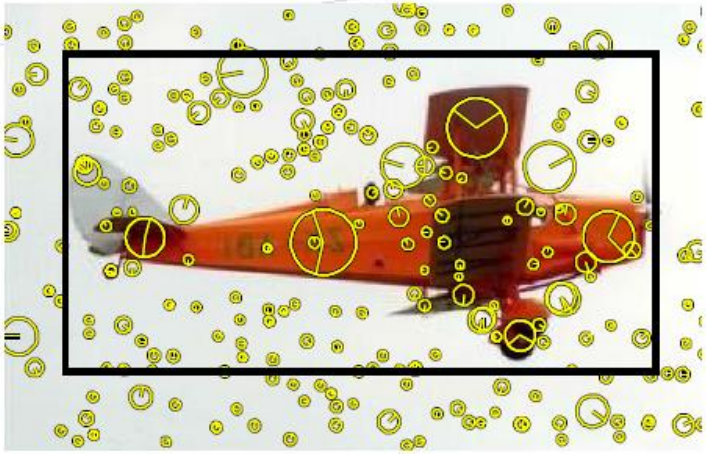
Correspondence score: $s_{ij} = \langle \mathbf{w}_i , \mathbf{d}_j \rangle$

Hare, Amri, Torr, CVPR 2012

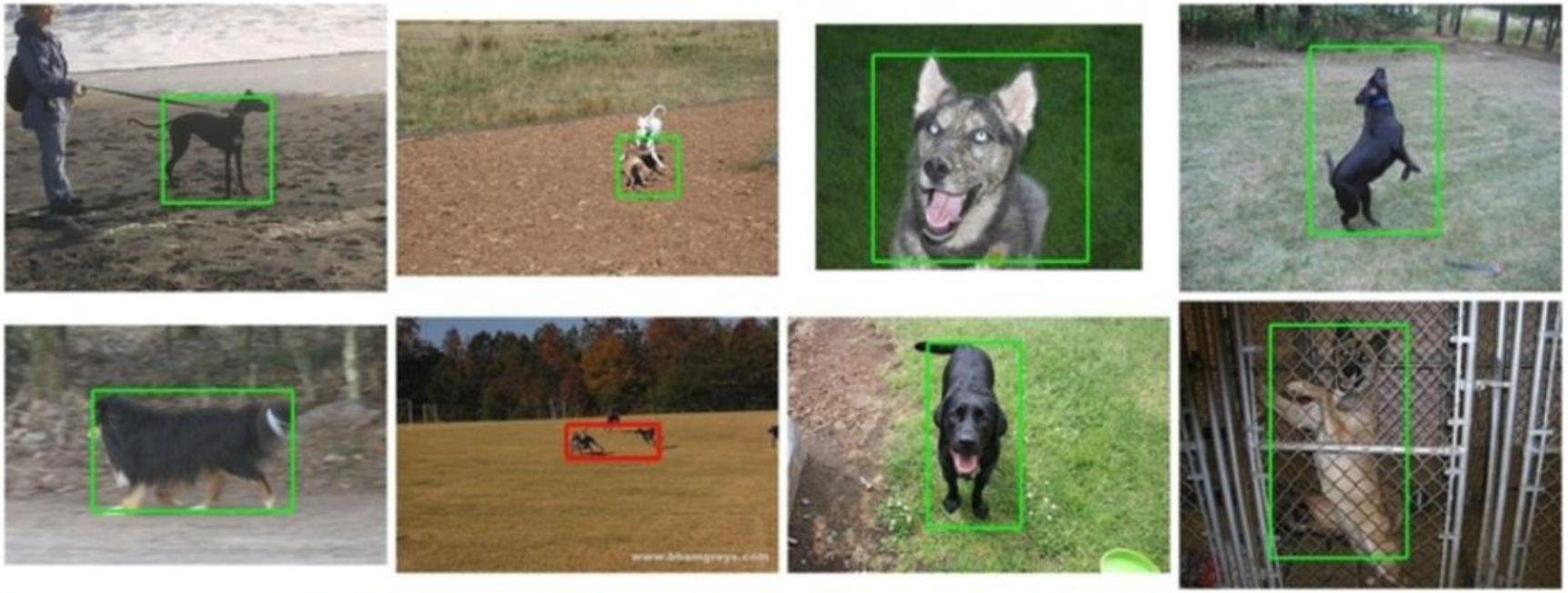
Example 8: Object category recognition

Sliding window detector

- Classifier: SVM with linear kernel
- BOW representation for ROI



Example detections for dog



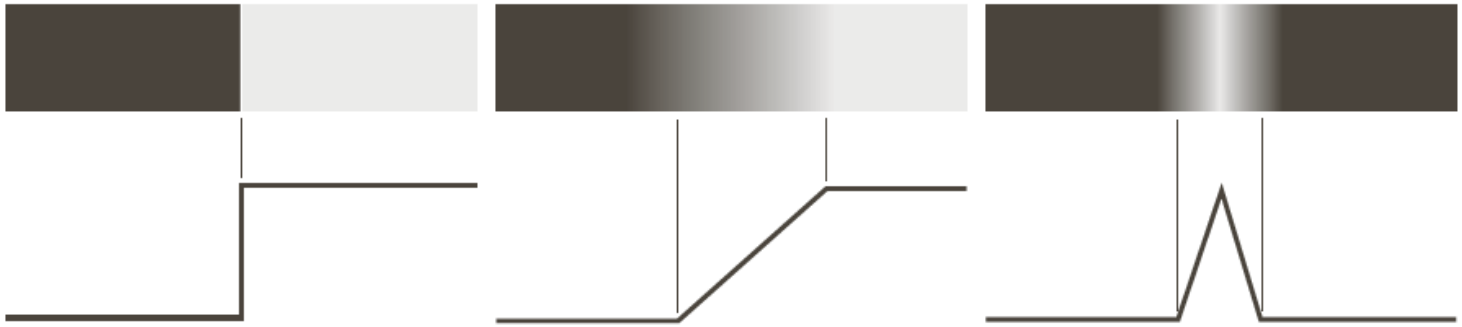
Lampert et al CVPR 08: Efficient branch and bound search over all windows

Type of Features

Extract relevant features from the images

- Isolated points
- Lines
- Edges
- Corners (keypoints are typically corners or regions of high variance)
- Other…

Features: Edges



a b c

FIGURE 10.8 From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

a) Step
b) Ramp
c) Roof

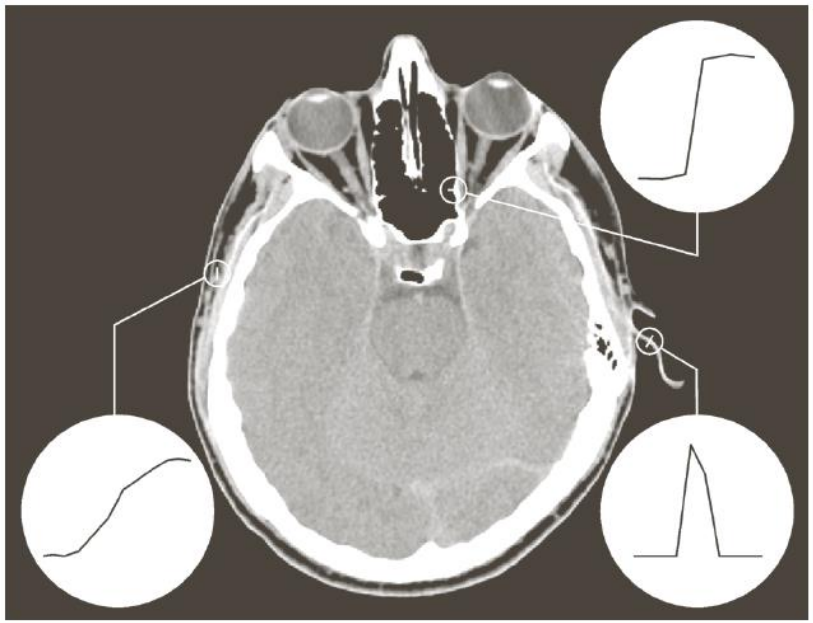
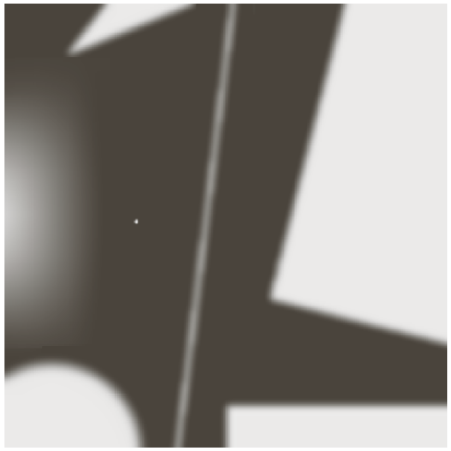
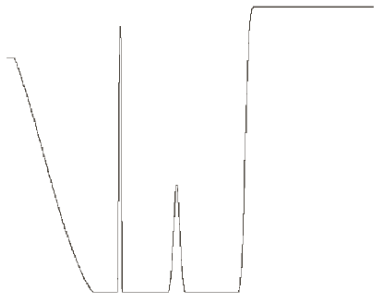


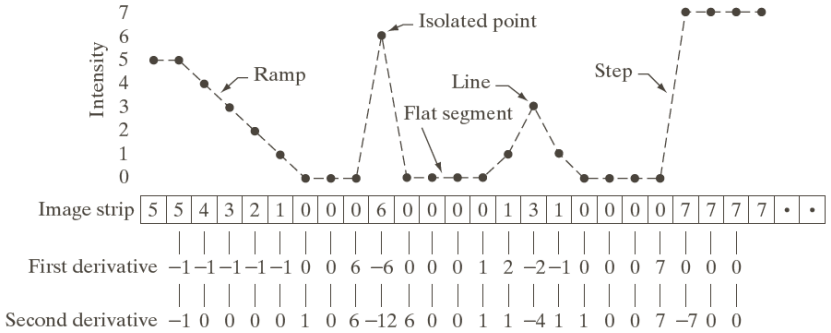
FIGURE 10.9 A 1508 × 1970 image showing (zoomed) actual ramp (bottom, left), step (top, right), and roof edge profiles. The profiles are from dark to light, in the areas indicated by the short line segments shown in the small circles. The ramp and "step" profiles span 9 pixels and 2 pixels, respectively. The base of the roof edge is 3 pixels. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

Figure from Gonzalez's book

The simplest idea for edge detection: First and second order derivatives







a b
c

FIGURE 10.2 (a) Image. (b) Horizontal intensity profile through the center of the image, including the isolated noise point. (c) Simplified profile (the points are joined by dashes for clarity). The image strip corresponds to the intensity profile, and the numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10.2-1) and (10.2-2).

	1° order derivative	2° order derivative
Large values	On all the edge	Start and end of edge
Edge type	Single	Double
Edge thickness	Thick	Thin
Find small details	No	Yes
Noise sensitivity	Moderate	High

Figure from Gonzalez's book

Derivatives are quite sensible to noise

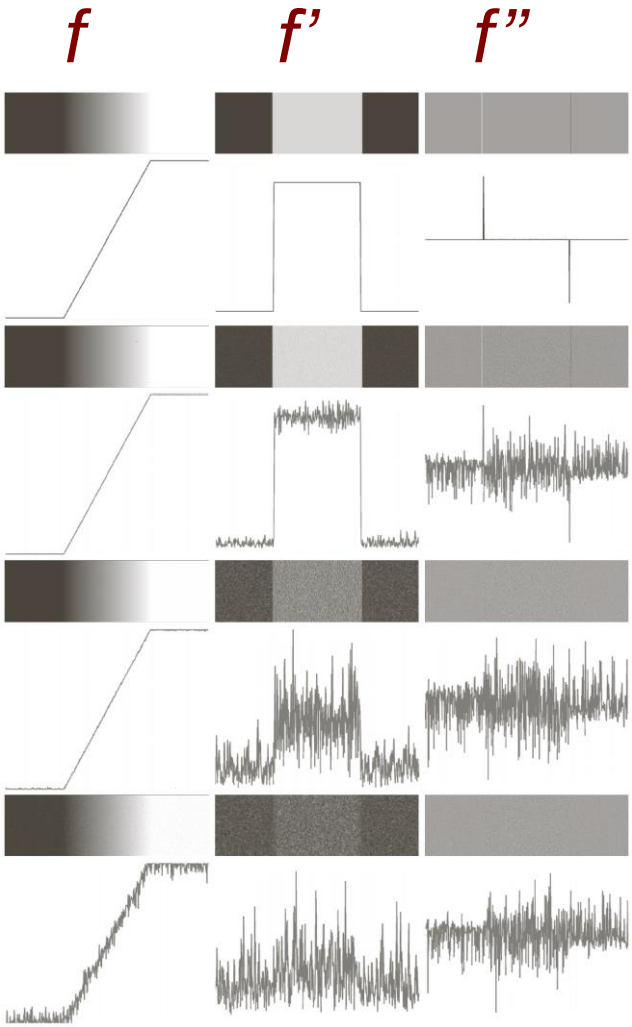


FIGURE 10.11 First column: Images and intensity profiles of a ramp edge corrupted by random Gaussian noise of zero mean and standard deviations of 0.0, 0.1, 1.0, and 10.0 intensity levels, respectively. Second column: First-derivative images and intensity profiles. Third column: Second-derivative images and intensity profiles.

- *Noise strongly affect the derivative computation*
- *2nd order derivative is particularly sensible*
- *The gradient vector is perpendicular to the edge*

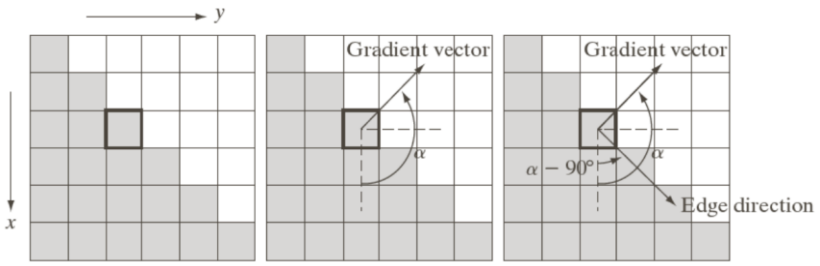


Figure from Gonzalez's book

Masks for oriented gradient

−1
1

−1	1

a b

FIGURE 10.13 One-dimensional masks used to implement Eqs. (10.2-12) and (10.2-13).

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0	0	−1
0	1	1	0

Roberts

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Sobel

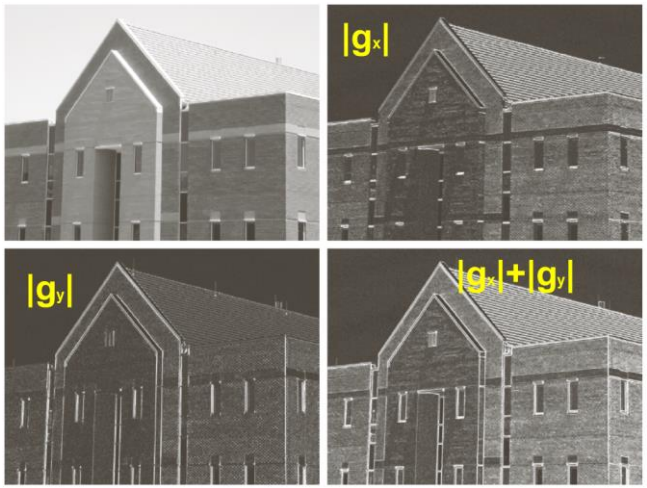


Figure from Gonzalez's book

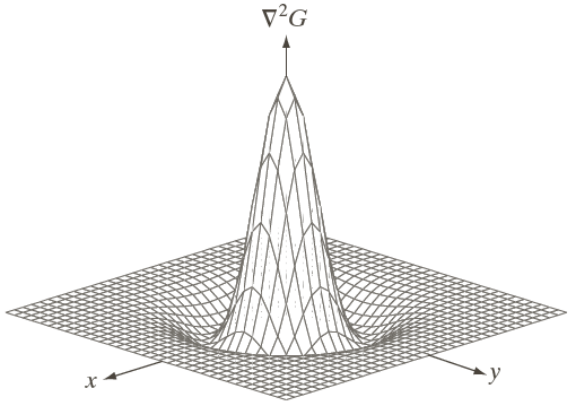
A simple edge detector

3 Steps

1. Low pass filtering (noise removal)
2. Compute the Gradient
3. Thresholding of the gradient

Figure from Gonzalez's book

Laplacian of a Gaussian (LoG)



*plot of -LoG

0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

a b
c d

FIGURE 10.21 (a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d) 5 × 5 mask approximation to the shape in (a). The negative of this mask would be used in practice.

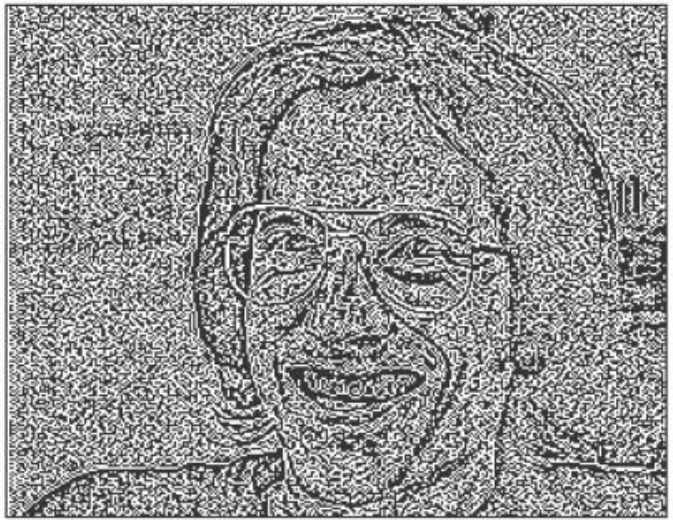
- Intensity change: maxima of f' or zero crossing of f''
- Smoothing : noise removal
- Isotropic (laplacian)
- Matches the Human Visual System characteristics

$$g(x,y) = \left[\nabla^2 G(x,y)\right] * f(x,y)$$
$$= \nabla^2 \left[G(x,y) * f(x,y)\right]$$

$$\nabla^2 G(x,y) = \frac{\partial^2 G(x,y)}{\partial x^2} + \frac{\partial^2 G(x,y)}{\partial y^2} = \frac{\partial^2}{\partial x^2}\left(e^{-\frac{x^2+y^2}{2\sigma^2}}\right) + \frac{\partial^2}{\partial y^2}\left(e^{-\frac{x^2+y^2}{2\sigma^2}}\right) = \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Figure from Gonzalez's book

Marr-Hildreth edge detector: based on the LoG



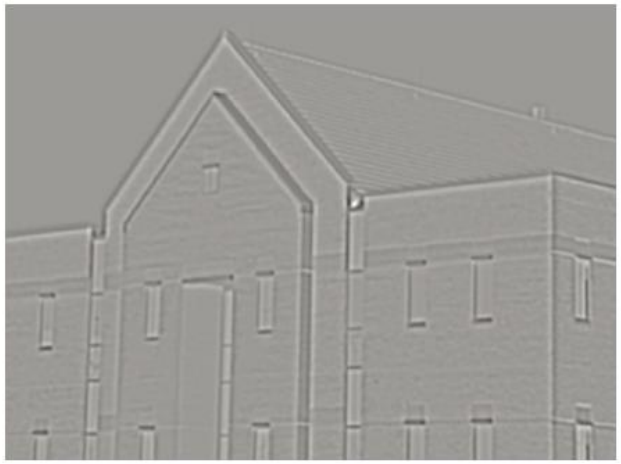
Gaussian Filter

- Sensitive to very fine detail and noise ➔ blur image first
- Responds equally to strong and weak edges
  ➔ suppress edges with low gradient magnitude

Threshold

Marr-Hildreth edge detector: procedure

LoG



a	b	c
d	x	e
f	g	h

Zero crossing:

$$(ah < 0) \vee (bg < 0) \vee (cf < 0) \vee (de < 0)$$

Threshold

(on couples of zero-crossing)

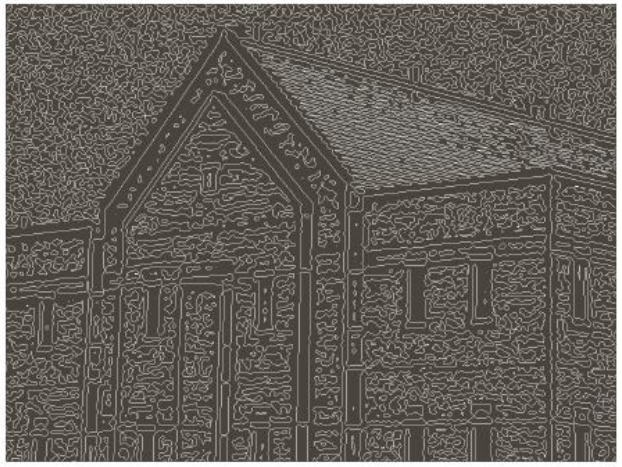
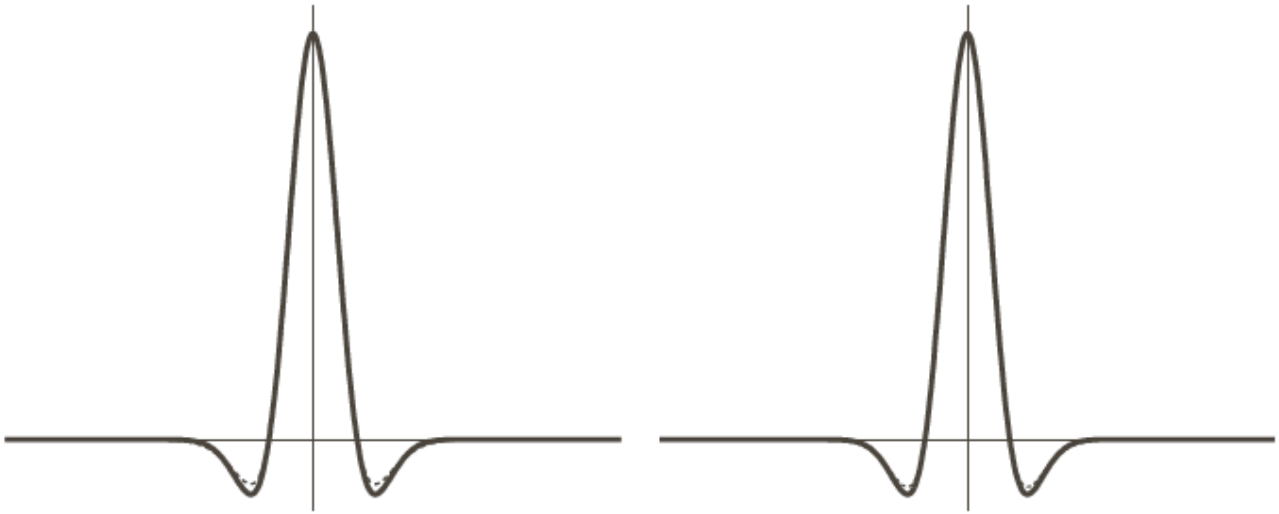


Figure from Gonzalez's book

LoG can be approximated with DoG



a b

FIGURE 10.23
(a) Negatives of the LoG (solid) and DoG (dotted) profiles using a standard deviation ratio of 1.75:1. (b) Profiles obtained using a ratio of 1.6:1.

*The plots show –LoG and -DoG

$$LoG: g(x,y) = \nabla^2 \left(e^{-\frac{x^2+y^2}{2\sigma^2}} \right)$$

$$DoG: g(x,y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$

$$\sigma^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 - \sigma_2^2} \ln\left[\frac{\sigma_1^2}{\sigma_2^2}\right]$$

$$\sigma_1 > \sigma_2 \qquad (\sigma_1 = 1.6\sigma_2)$$

Figure from Gonzalez's book

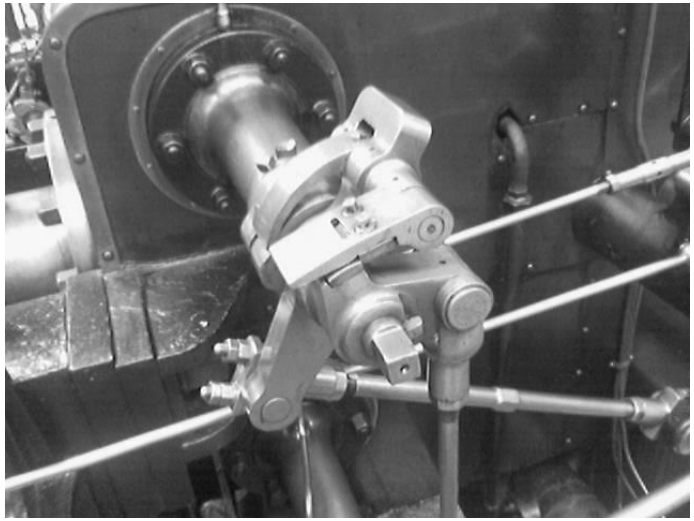
Canny edge detector

Targets:

1. Low error rate
2. Precisely locate edge points
3. “Single” edges

Canny, J., *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

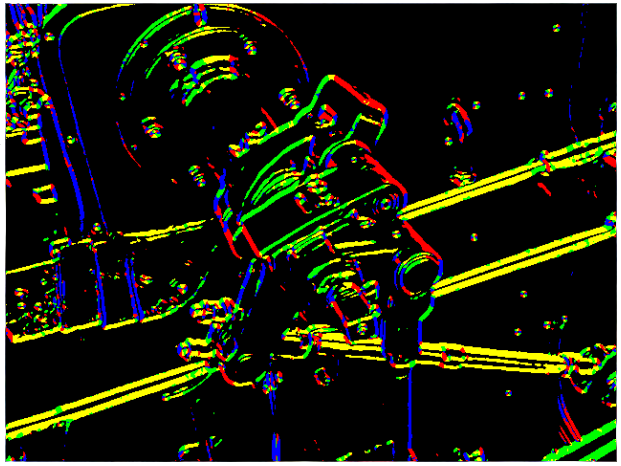
Derivative of the gaussian



$$f_s(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} * f(x,y)$$

$$M(x,y) = \sqrt{\left(\frac{\partial f_s}{\partial x}\right)^2 + \left(\frac{\partial f_s}{\partial y}\right)^2}$$

$$\alpha(x,y) = \tan^{-1}\left[\frac{\left(\frac{\partial f_s}{\partial x}\right)}{\left(\frac{\partial f_s}{\partial y}\right)}\right]$$



0°

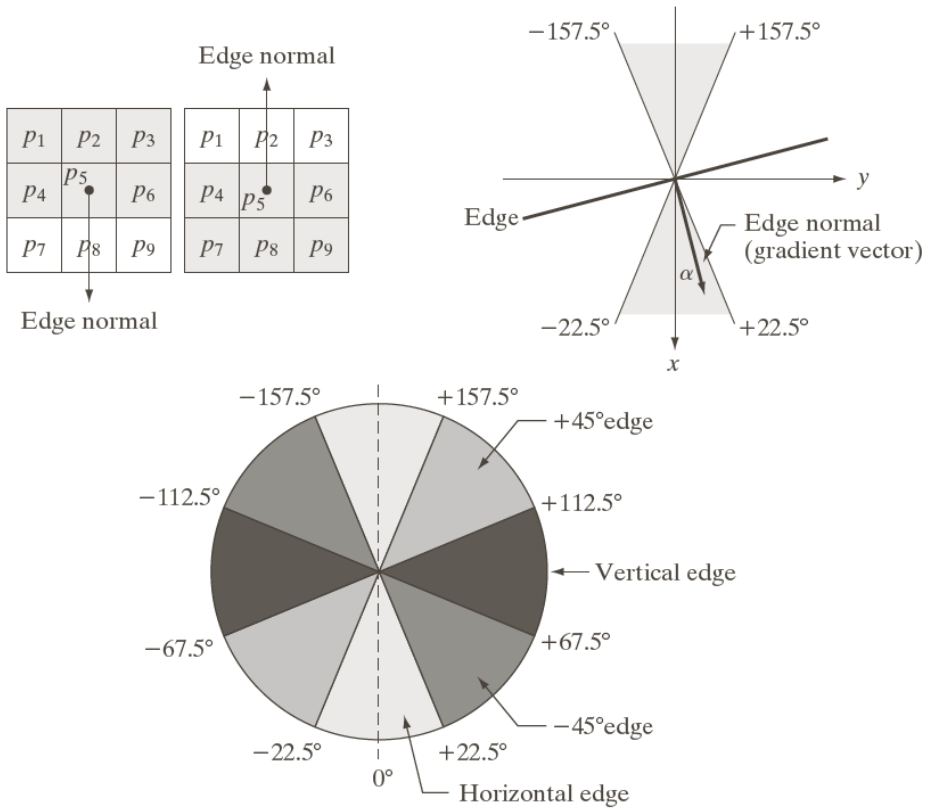
45°

90°

135°

Thick edges!

Non-Maxima suppression



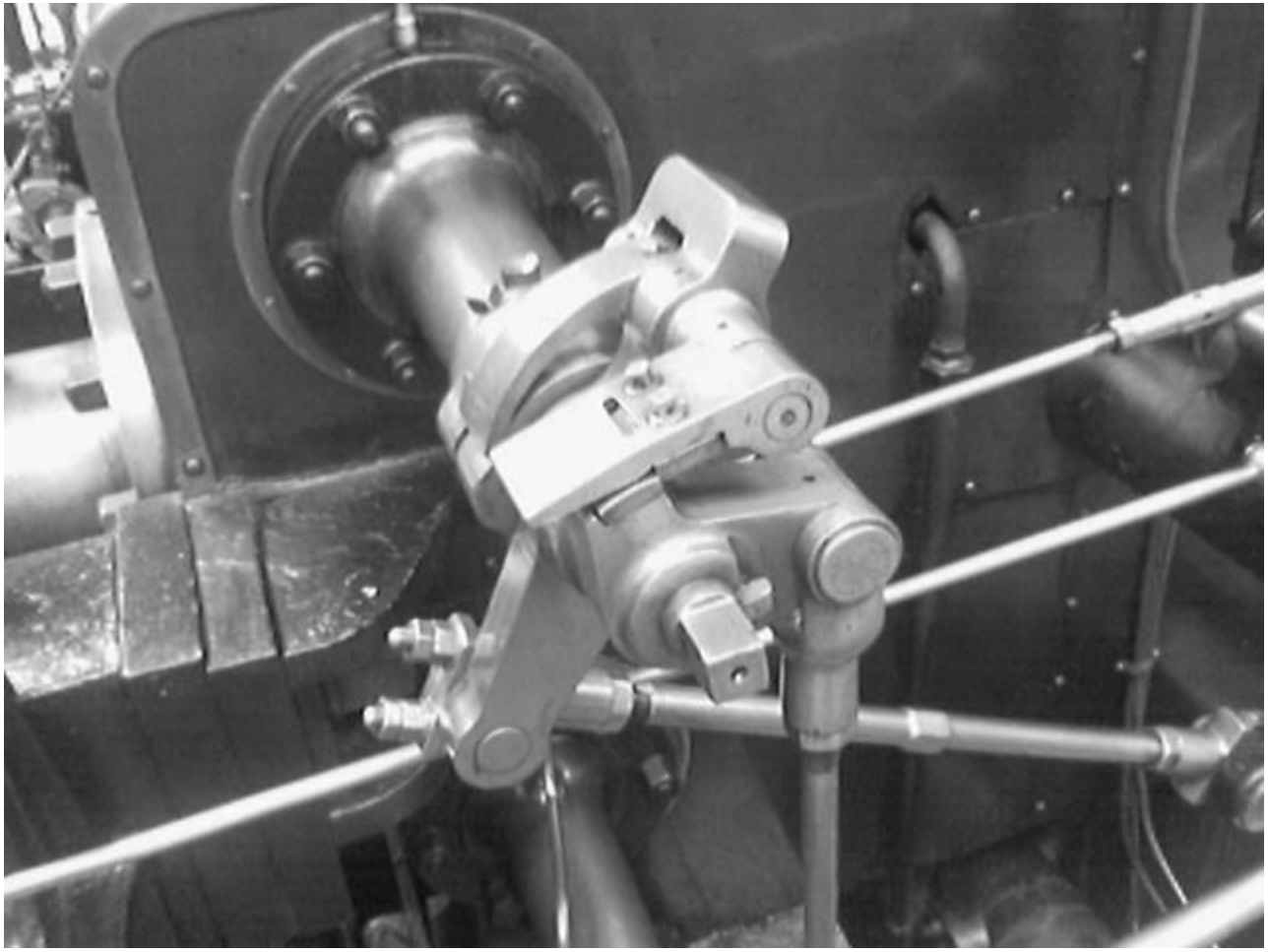
a b
c

FIGURE 10.24
(a) Two possible orientations of a horizontal edge (in gray) in a 3 × 3 neighborhood. (b) Range of values (in gray) of α, the direction angle of the *edge normal*, for a horizontal edge. (c) The angle ranges of the edge normals for the four types of edge directions in a 3 × 3 neighborhood. Each edge direction has two ranges, shown in corresponding shades of gray.

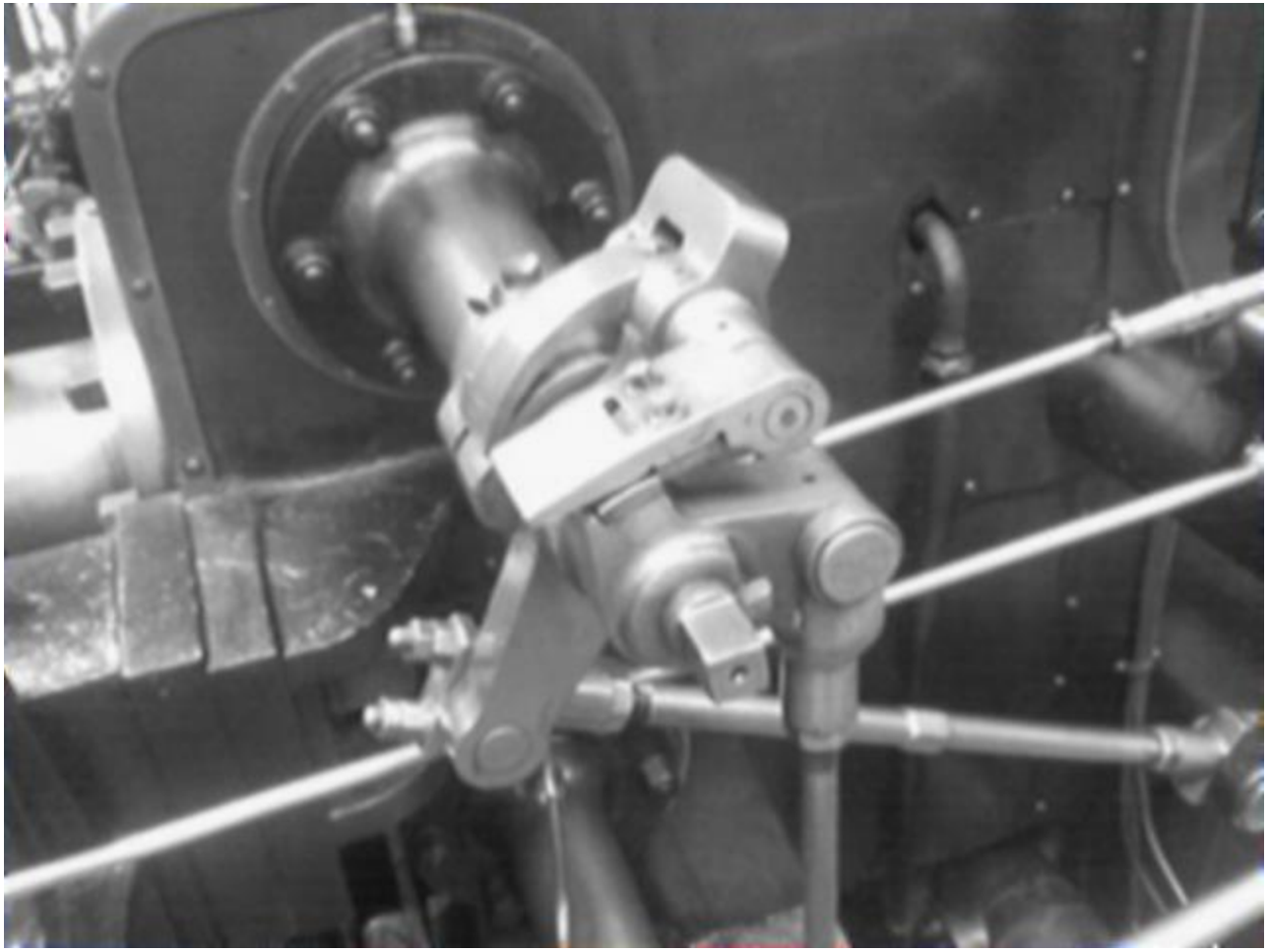
Figure from Gonzalez's book

Canny edge detector

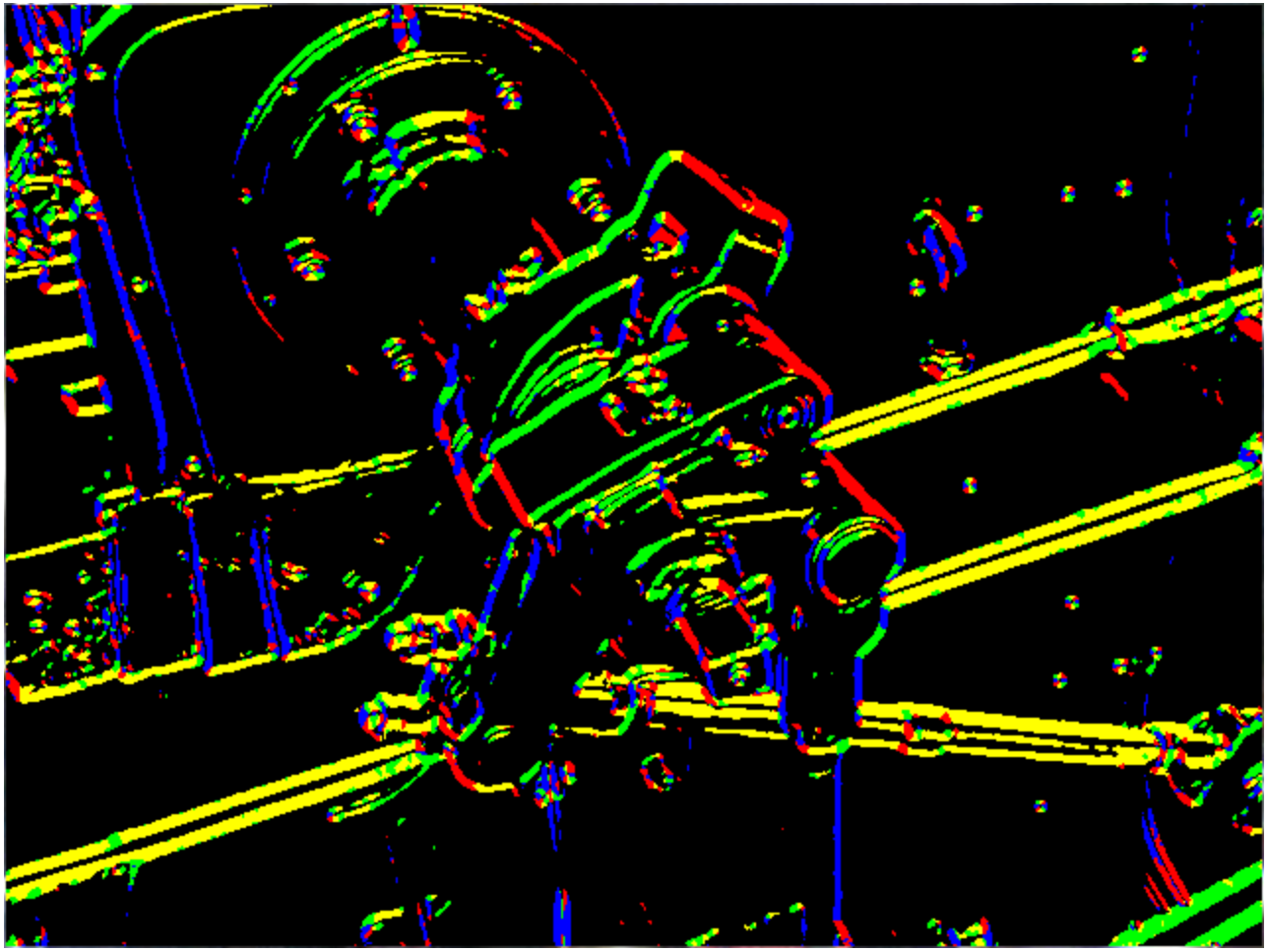
1. Smoothing with a Gaussian filter
2. Compute gradient (module and direction)
3. Quantize the gradient angles
4. Non maxima suppression
5. Thresholding with double threshold
6. *(Eventually edge linking)*



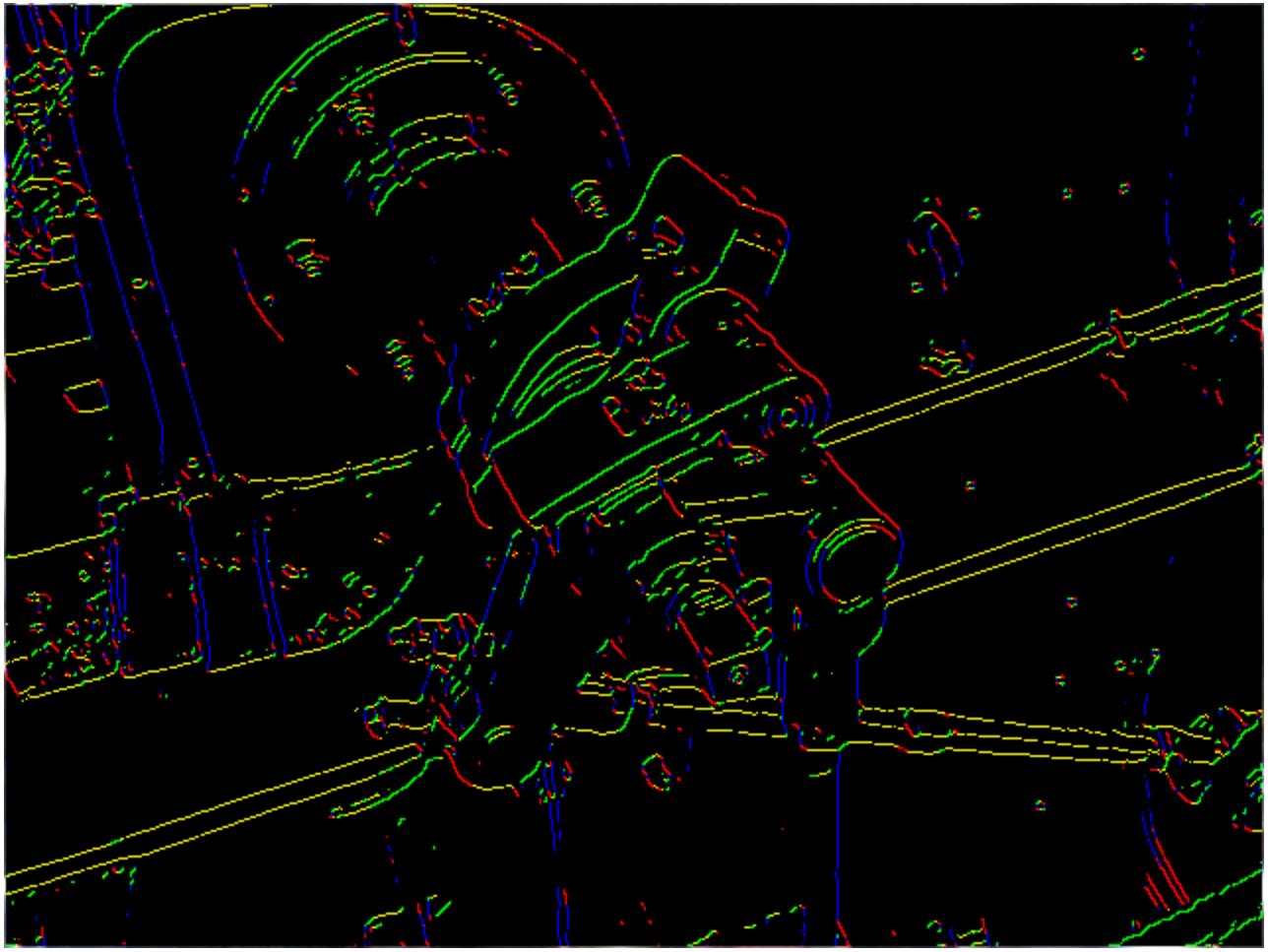
1. Original image



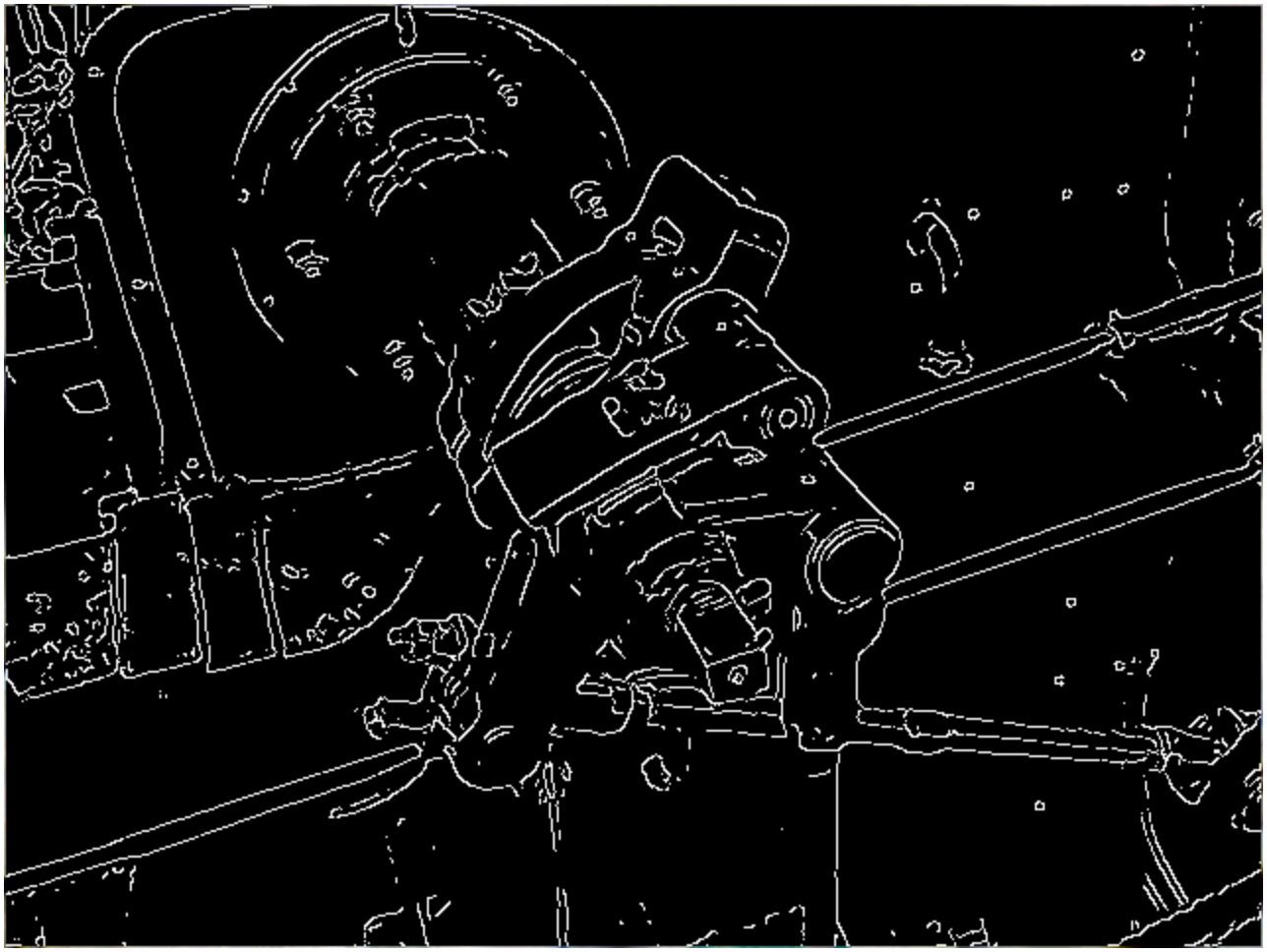
2. Gaussian smoothing



2. Gradient computed with Sobel’s mask



3. Non maximal suppression



4. Output of the Canny edge detector

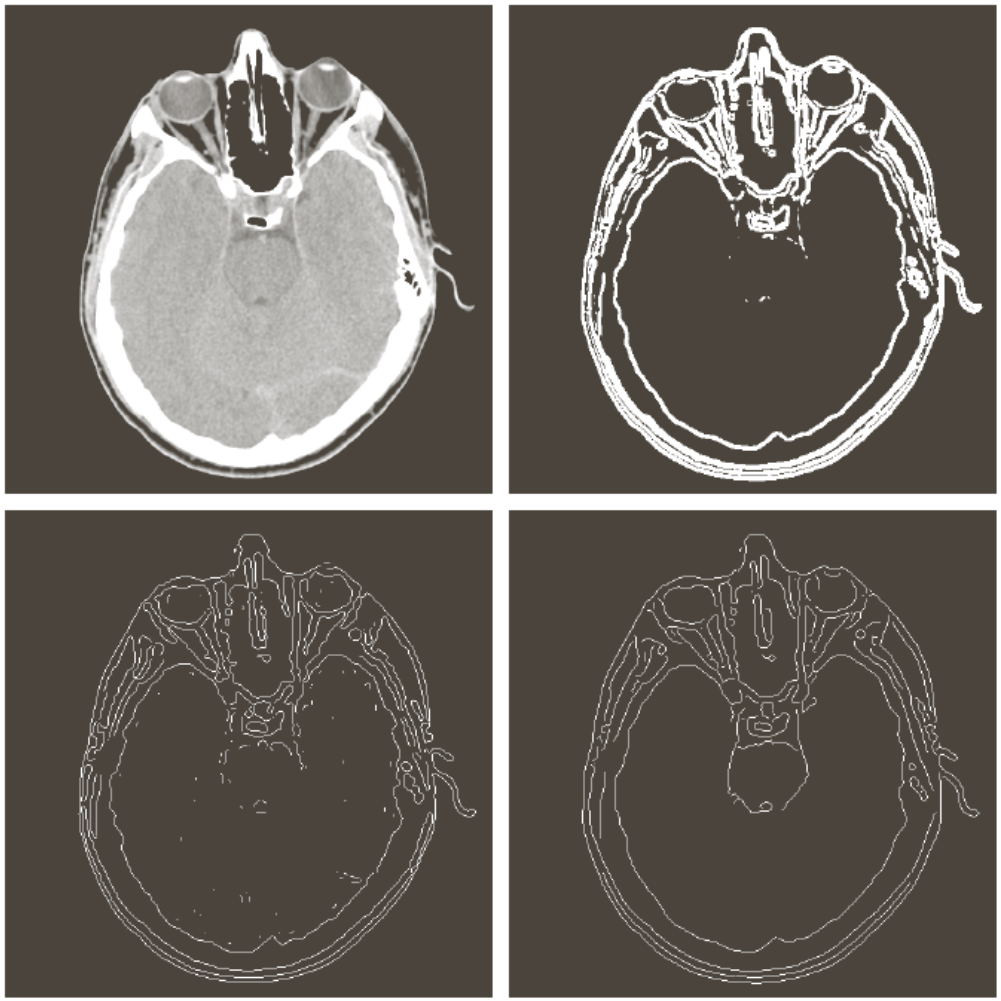
Gradient vs Canny vs Marr-Hildreth

a b
c d

FIGURE 10.25
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1].
(b) Thresholded gradient of smoothed image.
(c) Image obtained using the Marr-Hildreth algorithm.
(d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.

Figure from Gonzalez's book

Gradient vs Canny vs Marr-Hildreth

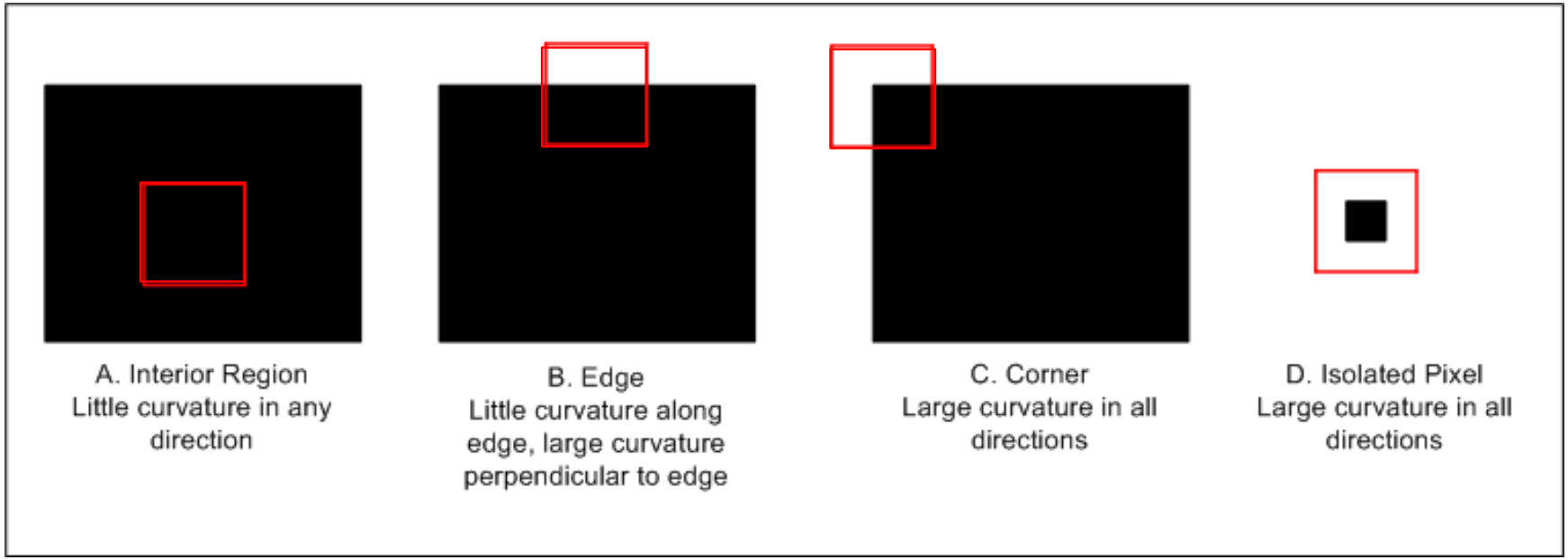


a b
c d

FIGURE 10.26
(a) Original head CT image of size 512 × 512 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

Figure from Gonzalez's book

Distinguish edge from corners

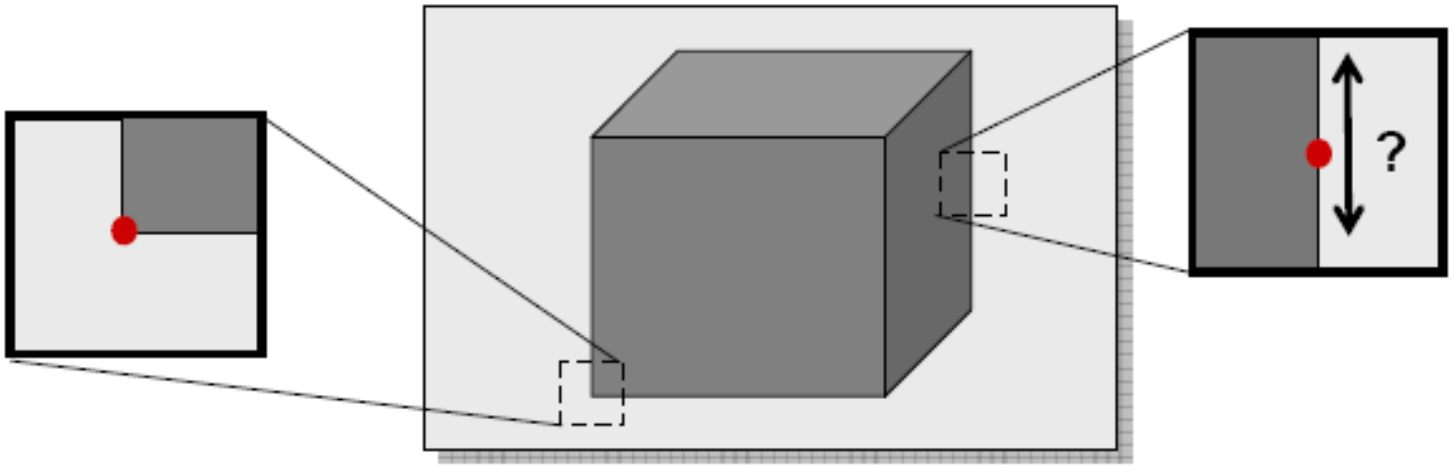


The corner is associated with variations in all directions

Moravec [1980]

Detecting corner points

- Many applications benefit from features localized in (x,y)
- Edges well localized only in one direction ➔ detect corners



- Desirable properties of corner detector
  - Accurate localization
  - Invariance against shift, rotation, scale, brightness change
  - Robust against noise, high repeatability

Local displacement sensitivity

Change of intensity for the shift $[u,v]$:

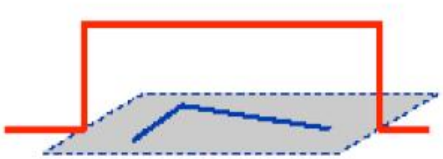
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

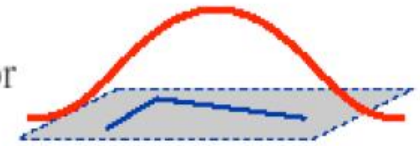
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

What patterns can be localized most accurately?

- Local displacement sensitivity

$$S(\Delta x, \Delta y) = \sum_{(x,y)\in window} \left[f(x,y) - f(x+\Delta x, y+\Delta y) \right]^2$$

- Linear approximation for small $\Delta x, \Delta y$

$$f(x+\Delta x, y+\Delta y) \approx f(x,y) + f_x(x,y)\Delta x + f_y(x,y)\Delta y$$

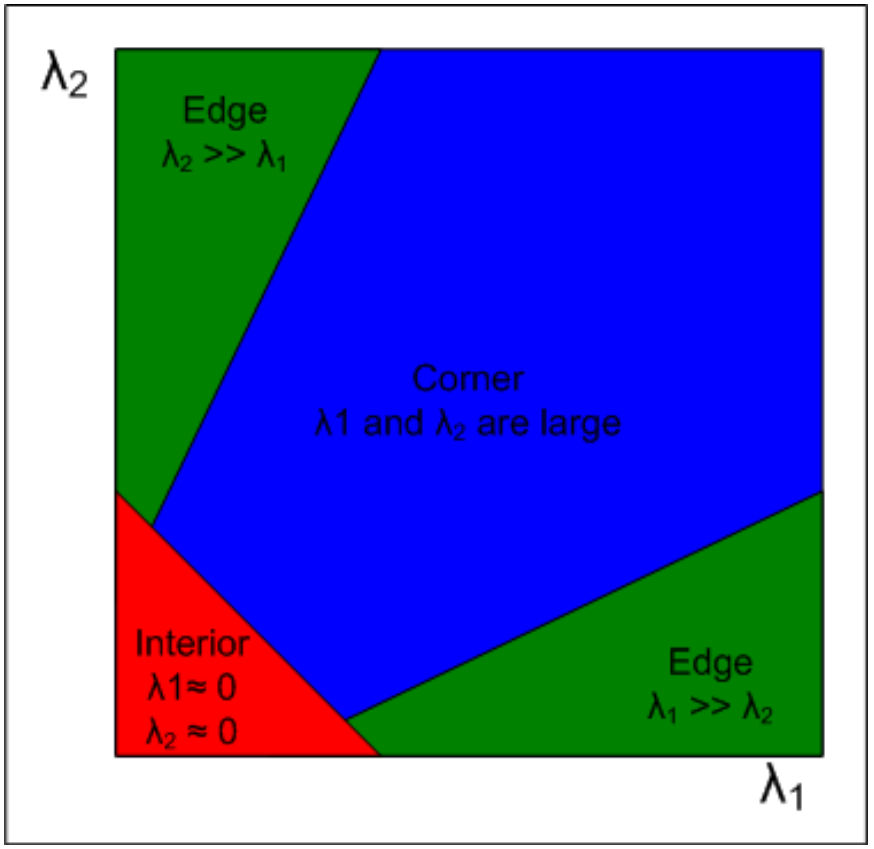
$$\begin{aligned} S(\Delta x, \Delta y) &\approx \sum_{(x,y)\in window} \left[\begin{pmatrix} f_x(x,y) & f_x(x,y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right]^2 \\ &= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \left(\sum_{(x,y)\in window} \begin{bmatrix} f_x^2(x,y) & f_x(x,y) f_y(x,y) \\ f_x(x,y) f_y(x,y) & f_y^2(x,y) \end{bmatrix} \right) \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \\ &= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \mathbf{M} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \end{aligned}$$

- Iso-sensitivity curves are ellipses

Eigenvalues and corners

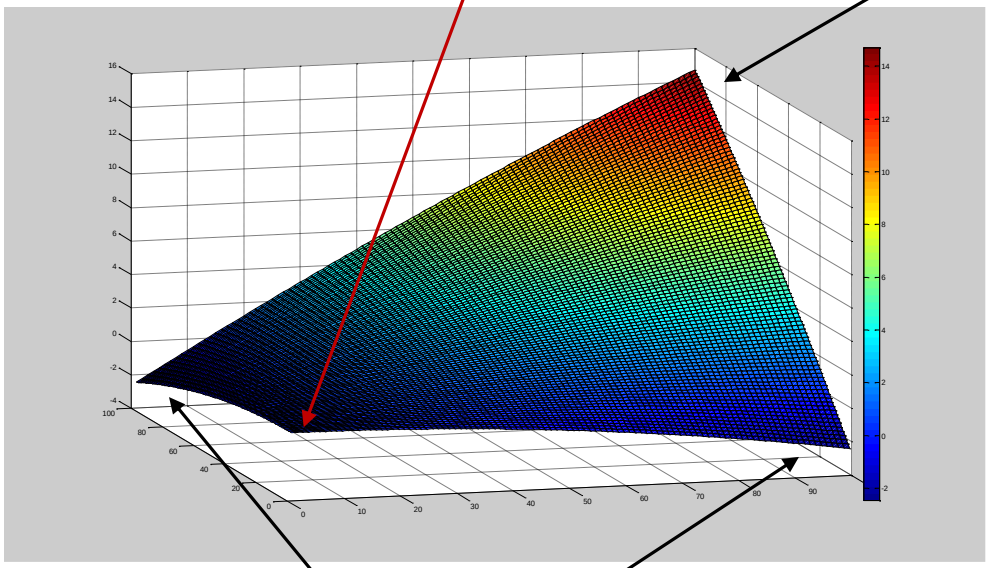
$$\mathbf{M} = \begin{bmatrix} \sum_{(x,y)\in window} f_x^2(x,y) & \sum_{(x,y)\in window} f_x(x,y) f_y(x,y) \\ \sum_{(x,y)\in window} f_x(x,y) f_y(x,y) & \sum_{(x,y)\in window} f_y^2(x,y) \end{bmatrix}$$

λ_1 , λ_2 eigenvalues of M



Alternative method

(Harris and Stephens 1988)



$$R(x,y) = \det[C(x,y)] + k\{Tr[C(x,y)]\}^2$$
$$= \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

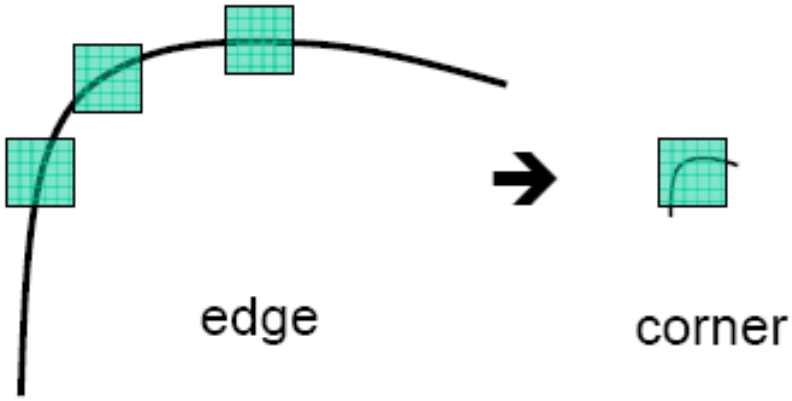
$$\det[C(x,y)] = \lambda_1\lambda_2$$
$$Tr[C(x,y)] = \lambda_1 + \lambda_2$$

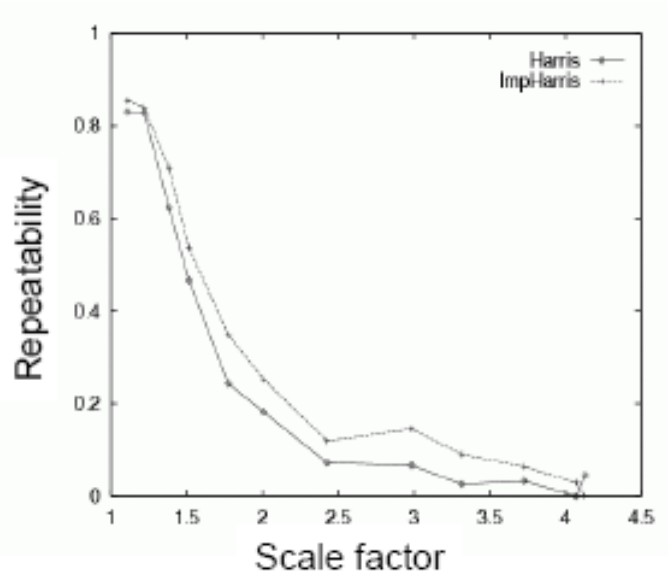
Eigenvalues computation for each pixel: very slow
R(x,y): Faster alternative to eigenvalue decomposition

Robustness of Harris Corner Detector

- Invariant to brightness offset: $f(x,y) \rightarrow f(x,y) + c$
- Invariant to shift and rotation

- Not invariant to scaling



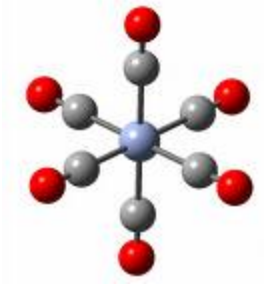


[Schmid, 2000]

Example

Example

What about the «scale»



Molecule
10^-12 m

Leaf
10 cm

Tree
5m

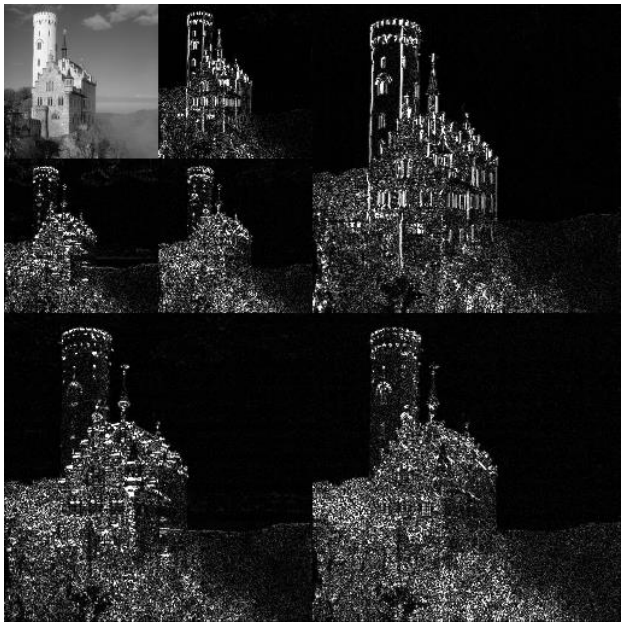
Forest
1km

Each objects exist at a certain scale level…..

Multi-scale representations

Scale-Space

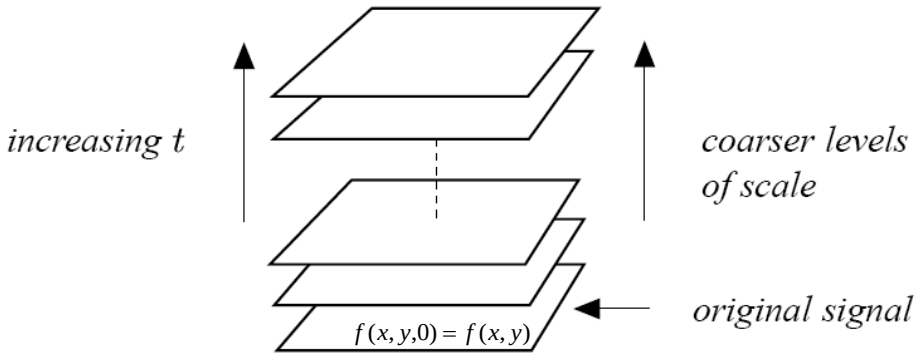
Pyramids



Wavelets

Scale Space

$$f(x,y) \rightarrow f(x,y,t)$$



- Family of signals parameterized by a continuous parameter (scale)
- Smaller structures are progressively suppressed while the scale is increased

Scale space	Multi-resolution
Continuous parameter	Discrete set of resolutions
Same sampling frequency at all the scales	The sampling frequency changes

Scale Space: Requirements and Properties

- Causality:
  - Non-creation of local extrema
  - Non-enhancement of local extrema
- Scale-invariance
- Shift-invariance
- Rotation-invariance
- Linearity (includes commutativity between derivative and convolution)

Definition: Scale Space

Convolution of the image with Gaussian kernels with varying σ

$$L(x, y; t) = g(x, y; t) * f(x, y)$$

$$g(x, y; t) = \frac{1}{2\pi t} e^{-(x^2 + y^2)/2t}$$

Gaussian Scale Space motivations

- The receptors in the human eye retina can be modeled as a superposition of derivatives of Gaussian functions (Young 1987)
- The solution of the heat diffusion equation leads to the same result

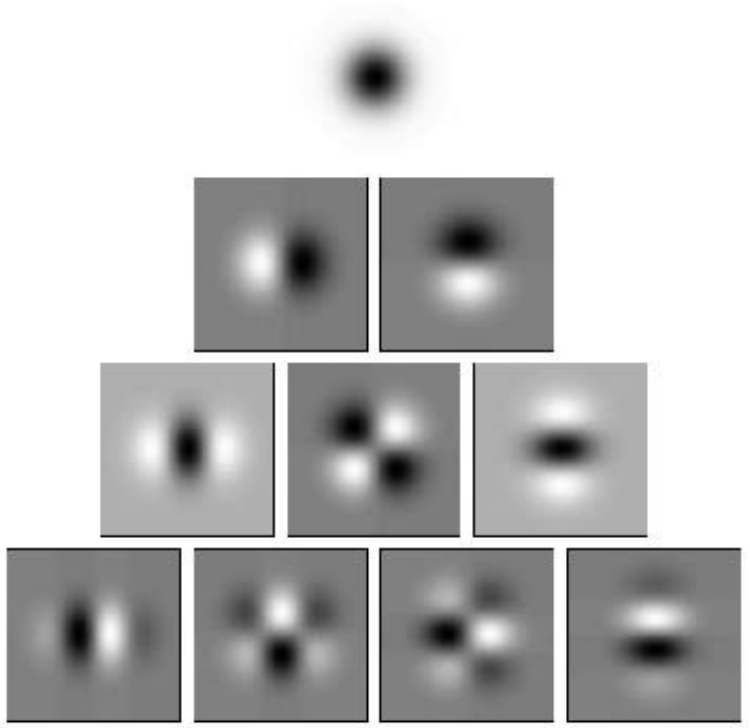


Figure 4: Gaussian derivative kernels up to order four the two-dimensional case.

$$\partial_t L = \frac{1}{2} \nabla^2 L$$

Scale Space

t=0

Scale Space

t=1

Scale Space

t=4

Scale Space

t=16

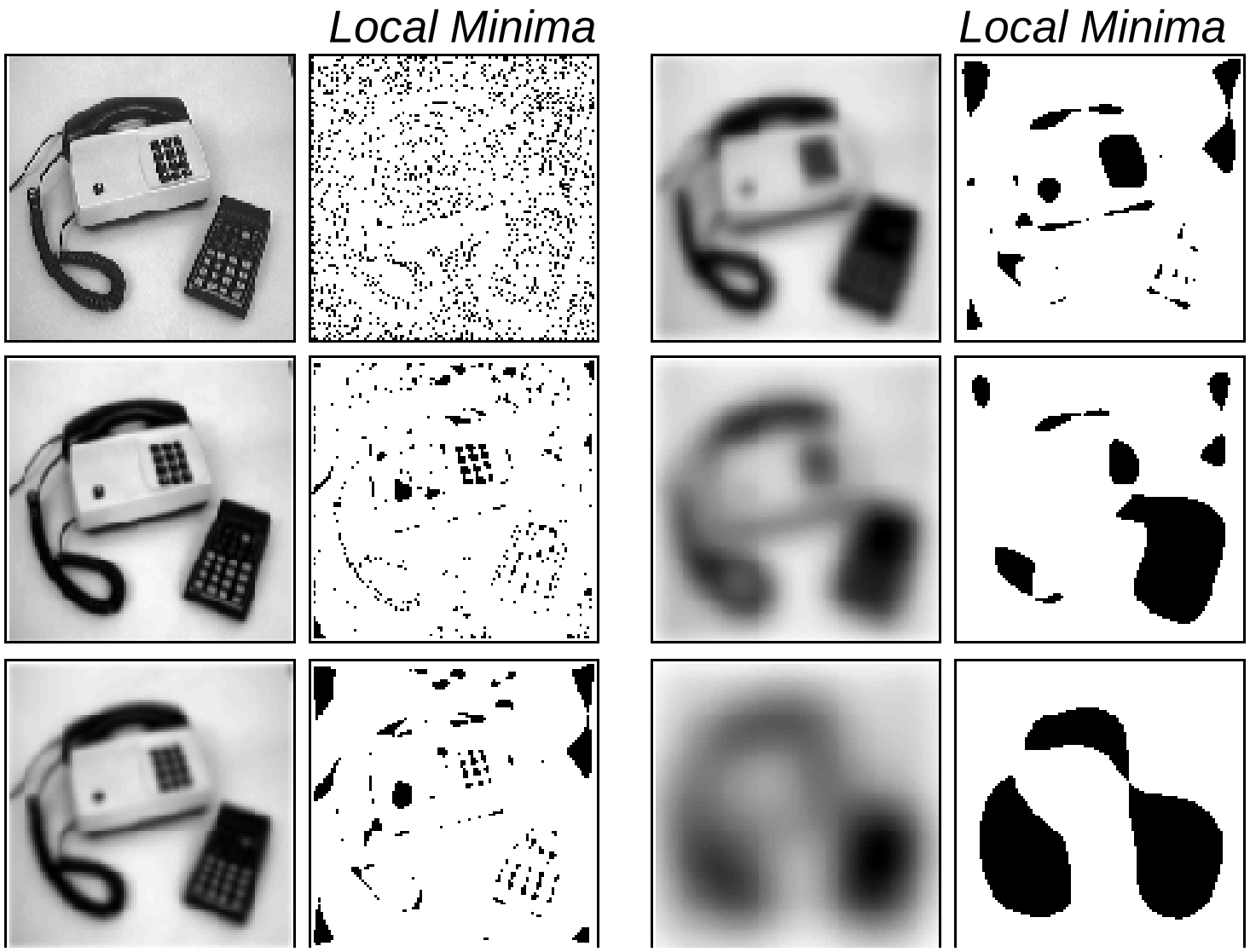
Scale Space

t=64

Scale Space

t=256

Example

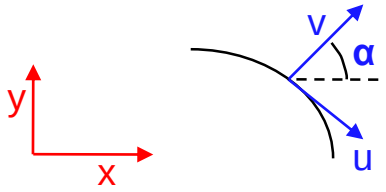


From Lindeberg's paper

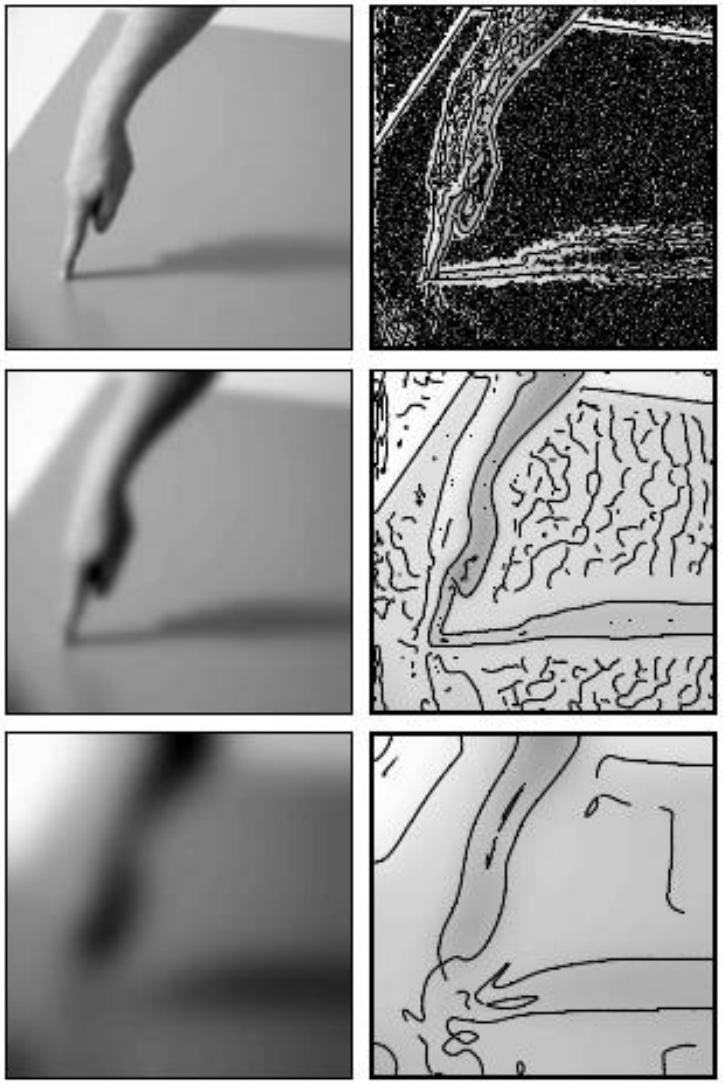
Edge Detection Algorithm

1. New reference system from (x,y) to (u,v) with v parallel to the gradient direction
2. Derivates along (u,v)
3. The maximum of the derivative along v is the edge point ($L_{vv}=0$ e $L_{vvv}<0$)

At which scale I need to work ?



Edges and scale space



t=1

t=16

t=256

- Strong edges are present at all the scales
- Thin Scales:
  Fake edges due to noise
- Coarse scales:
  The smallest details are lost and the localization of the edges is not accurate

From Lindeberg's paper

Automatic scale selection

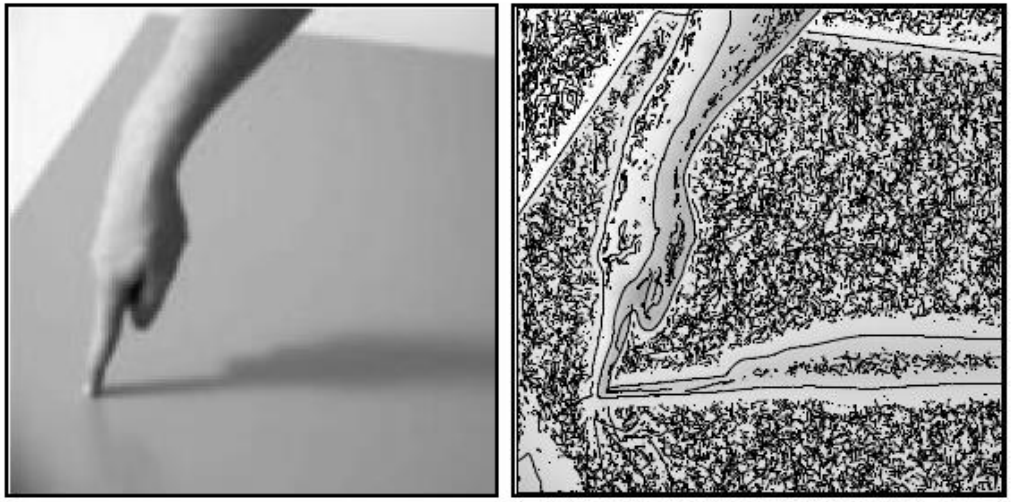
$$\partial_{\varepsilon_i} = t^{\gamma/2}\partial_{x_i}$$

$$F_{en} = t^{1/4}L_v$$

Feature type	Normalized strength measure for scale selection	Value of γ
Edge	$t^{\gamma/2}L_v$	1/2
Ridge	$t^{2\gamma}(L_{pp} - L_{qq})^2$	3/4
Corner	$t^{2\gamma}L_v^2 L_{vv}$	1
Blob	$t^{\gamma}\nabla^2 L$	1

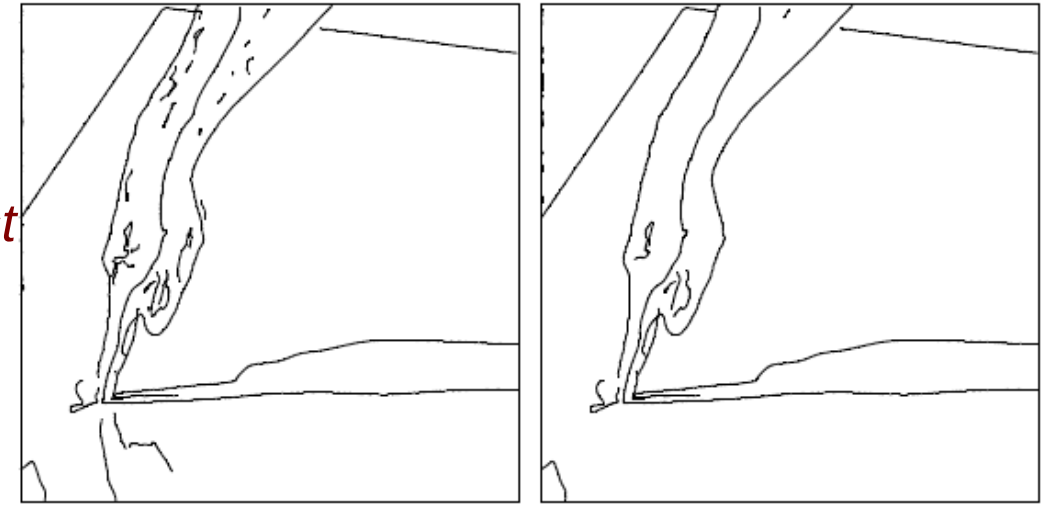
- Detect the best scale for each feature
- Derivates at smaller scales have larger values: need to normalize with respect to the scale
- Compute the feature detection at all the scales and select the strongest responses

Multi-Scale edge detection



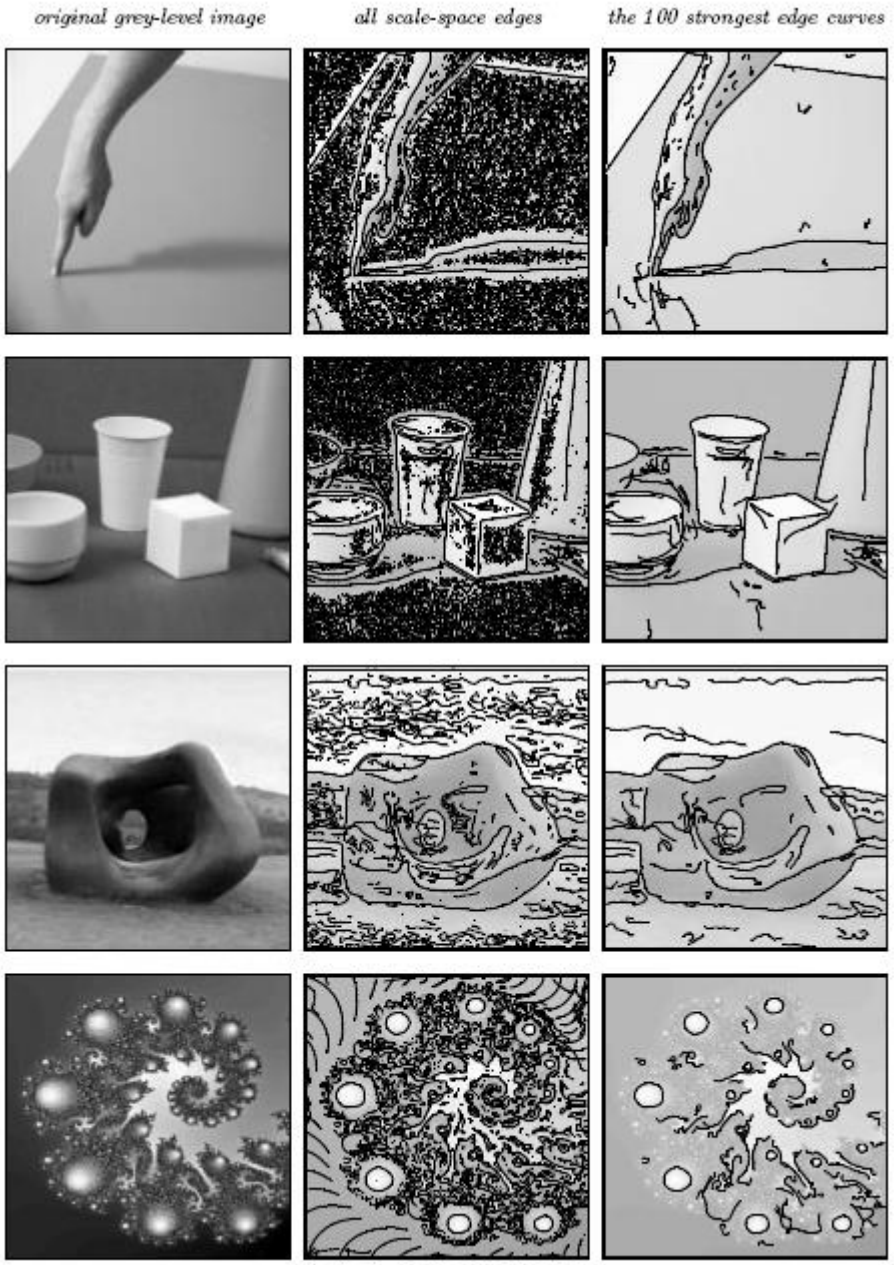
All the edges at all the scales

The 50 strongest edges



The 10 strongest edges

Scale-space edge detection



Corner

Curvatura

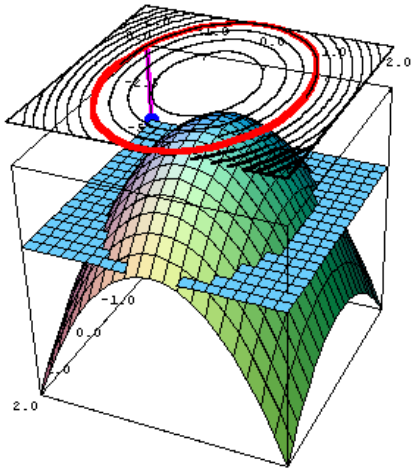
$$\tilde{\kappa} = L_v^3 \kappa = L_v^2 L_{uu} = L_x^2 L_{yy} + L_y^2 L_{xx} - 2 L_x L_y L_{xy},$$

Gradiente

$$F_c = \bar{k} = L_v^3 k = L_v^2 L_{uu}$$

$$F_{cn} = t^2 L_v^2 L_{uu}$$

$$k = \frac{L_{uu}}{L_v}$$



- Corner: locations where both the curvature of the level curves and the gradient are large
- Corners : γ=1
- Search for the corners at all the scale and select the strongest (normalizing w.r.t. the scale)

Corner detection

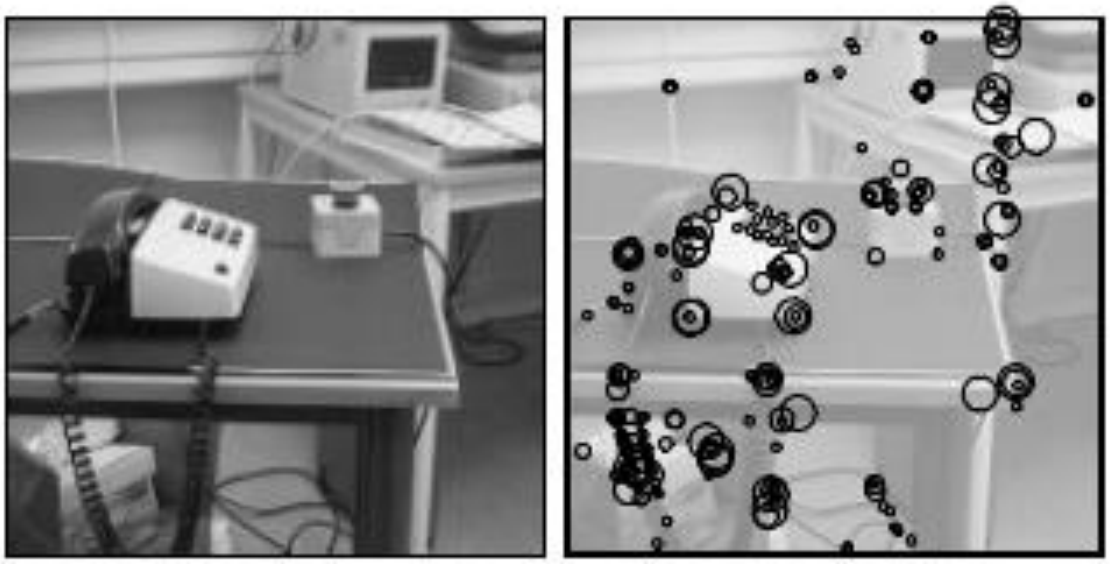
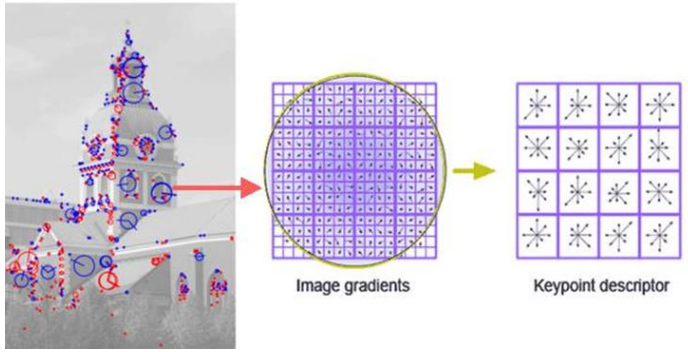


Figure 12: Results of corner detection with automatic scale selection on an office scene (200 strongest junction responses).

Part 2: Scale Invariant Feature Transform (SIFT)

In this part figures from Lowe's paper

A Milestone in Computer Vision



Although it's not always the case that a paper cited more contributes more to the field, a highly cited paper usually indicates that something interesting have been discovered. The following are the papers to my knowledge being cited the most in Computer Vision. (updated on 11/24/2013) If you want your "friend's" paper listed here, just comment below.

Cited by 21528 + 6830 (Object recognition from local scale-invariant features)

Distinctive image features from scale-invariant keypoints
DG Lowe - International journal of computer vision, 2004

Cited by 17671

A theory for multiresolution signal decomposition: The wavelet representation
SG Mallat – Pattern Analysis and Machine Intelligence, IEEE ..., 1989

Cited by 17611

A computational approach to edge detection
J Canny – Pattern Analysis and Machine Intelligence, IEEE ..., 1986

Cited by 15422

Snakes: Active contour models
M Kass, A Witkin, Demetri Terzopoulos - International journal of computer ..., 1988

Cited by 15188

Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images
Geman and Geman - Pattern Analysis and Machine ..., 1984

Cited by 11630+ 4138 (Face Recognition using Eigenfaces)

Eigenfaces for Recognition
Turk and Pentland, Journal of cognitive neuroscience Vol. 3, No. 1, Pages 71-86, 1991 (9358 citations)

Cited by 8788

Determining optical flow
B.K.P. Horn and B.G. Schunck, Artificial Intelligence, vol 17, pp 185-203, 1981

- Very reliable feature extractor and descriptor
- After 10 years it is still probably the most reliable even if not the fastest
- Widely used in a huge number of commercial applications and research works
  - The most cited paper in the computer vision field
  - More than 30k citations on the 2 papers (*Google Scholar*)
  - #6 in the whole computer science field(*Citeseer*)

Descriptors

- Feature Descriptors:
  - Describe some relevant feature of the scene
  - Allow to localize the same object or point in different images
- They must be robust:
  - To image rotations and traslations
  - To the scaling of the image
  - To the image noise
  - To lighting changes
  - To changes of the viewpoint (i.e., perspective transform, more complex than affine transform)
  - To occlusions

affine transforms

Invariant descriptors

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

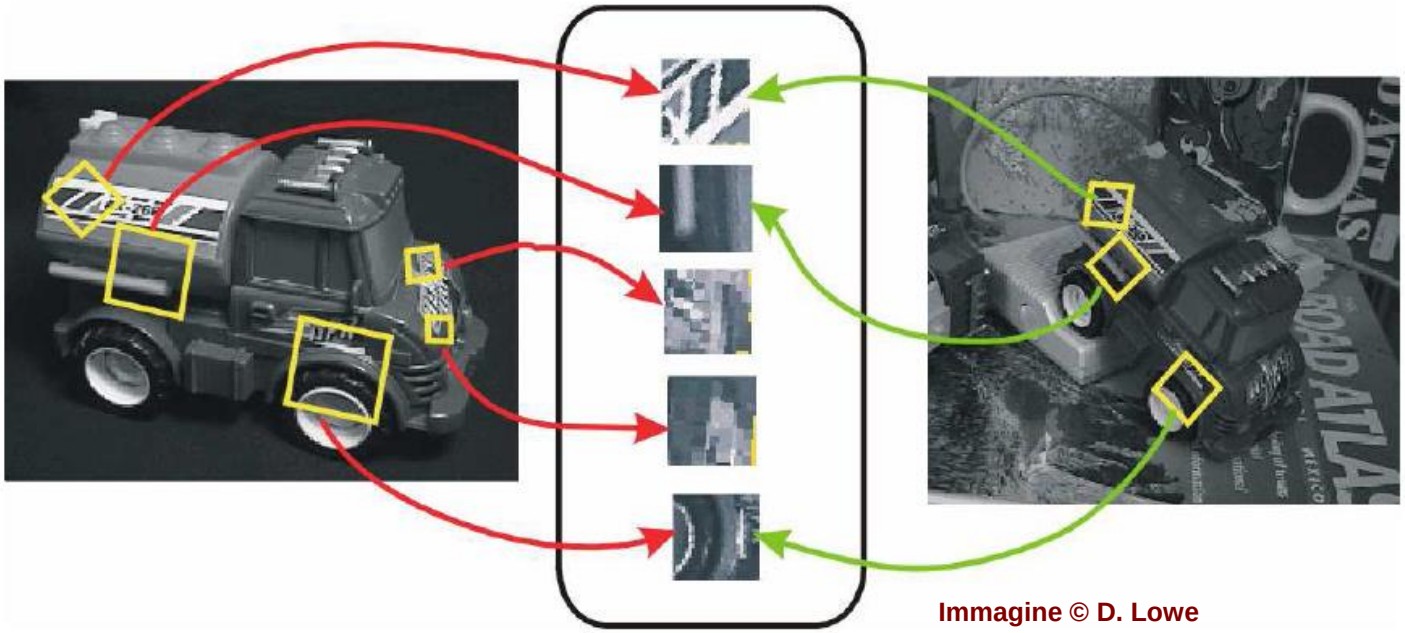


Immagine © D. Lowe

SIFT Features

A vector (feature descriptor) is associated to some relevant image points
It is invariant with respect to rotations, scaling, lighting, etc..

Applications

Objective: Identify the same object or point in different images

Training images

Test image

Applications

- Object recognition
  - Industrial automation
  - Robotics
- Matching and registration
  - Image mosaicing and panormic images
  - Stereo and 3D reconstruction
  - Optical Flow
- Content-based retrieval

Requirements

- **Locality**: the features are local. This makes them robust to object occlusions
- **Distinctiveness:** it should be possible to match the features even inside large databases
- **Quantity:** many feature could be needed even for small objects
- **Efficiency**: real-time or near real-time computation

Scale Invariant Feature Transform (Lowe 2004)

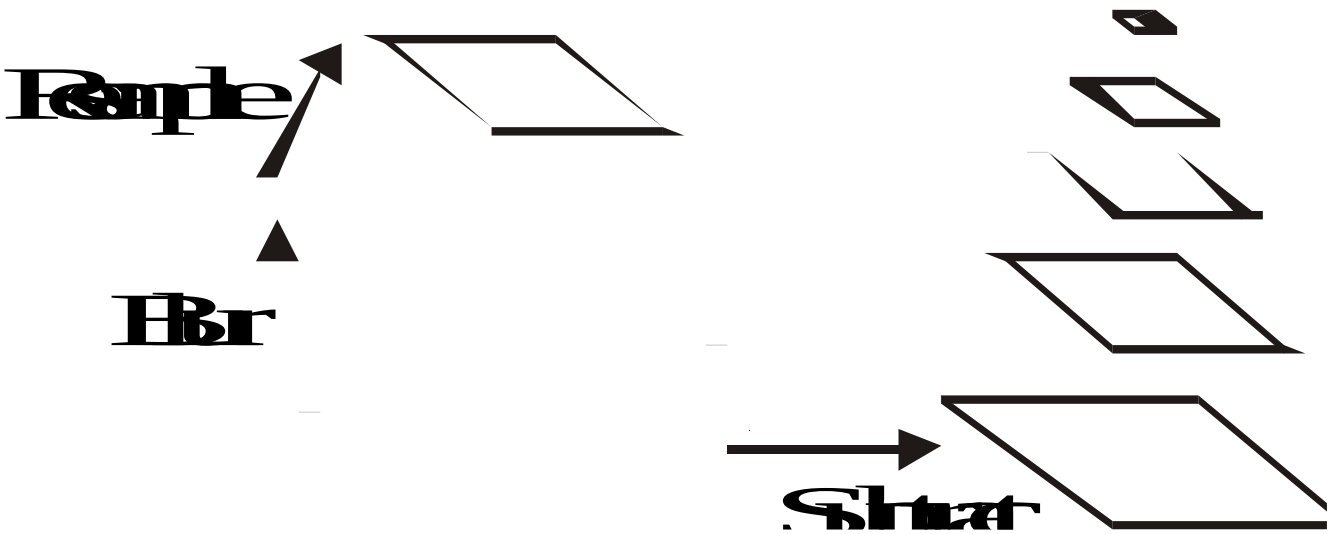
4 Steps :

1. Scale-space extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Cascade filtering approach

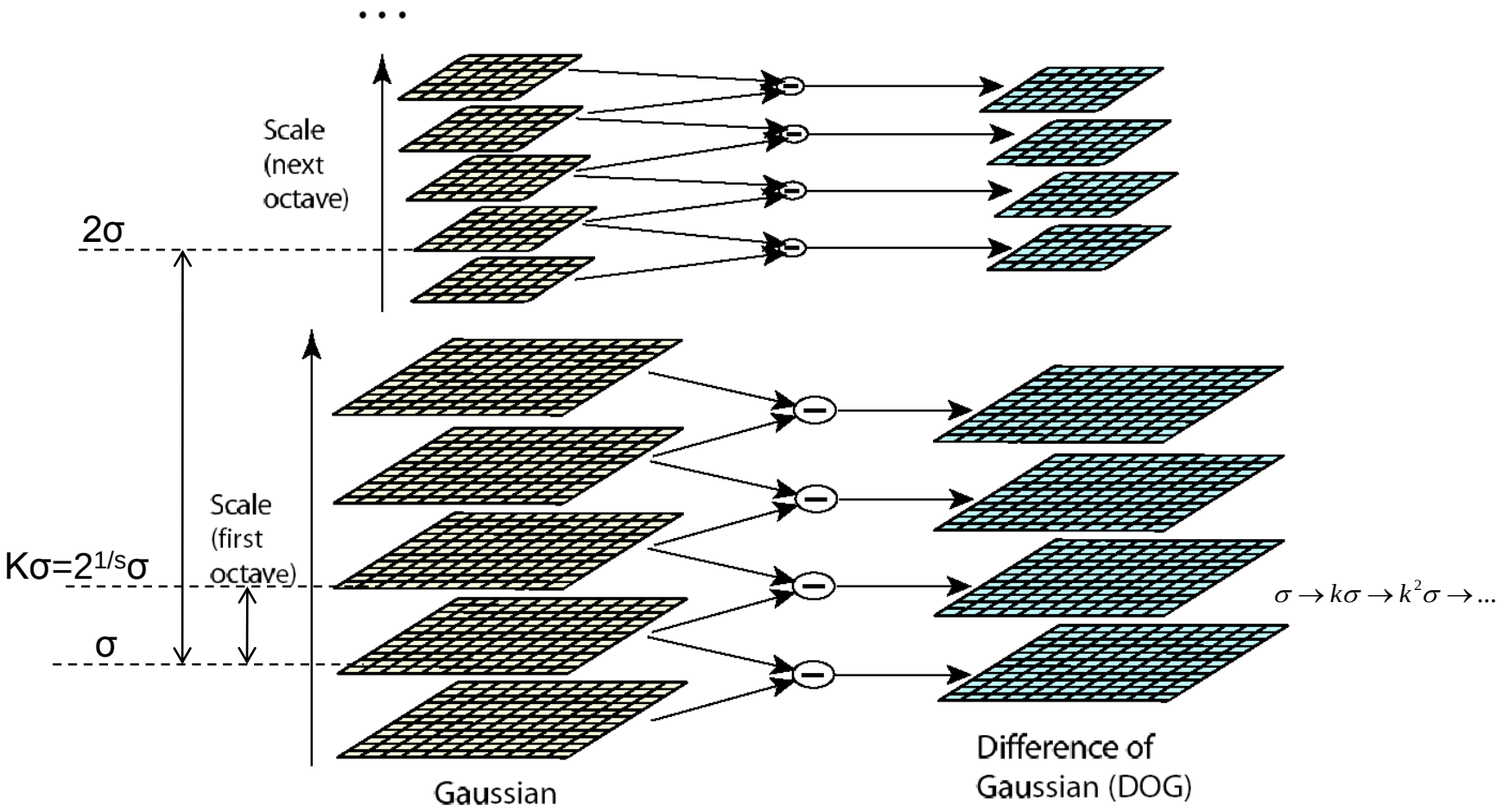
(Sequence of operations, the most complex are at the end and are executed only on a small subset of points)

Fase 1: Scale-space extrema detection



- Scale space divided in octaves (at each octave σ is doubled)
- Gaussian Scale space inside each octave
- Keypoints search on all the scales (scale-invariant result)

One octave after the other..

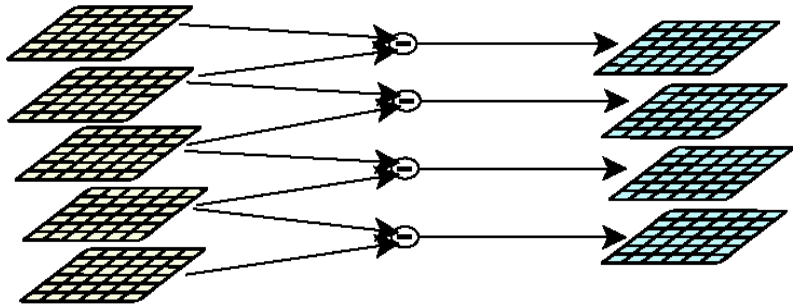


- Gaussian scale space with *s* intervals for each octave
- s+*3* images in each octave: (s+1)+2 to find the extrema

Difference of Gaussians (DoG)

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

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



$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

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

Lowe uses the Difference of Gaussians (DoG)

- Approximate LoG
- Simple to compute
- Includes normalization factor for the derivatives in the scale space

Difference-of-Gaussian: Example

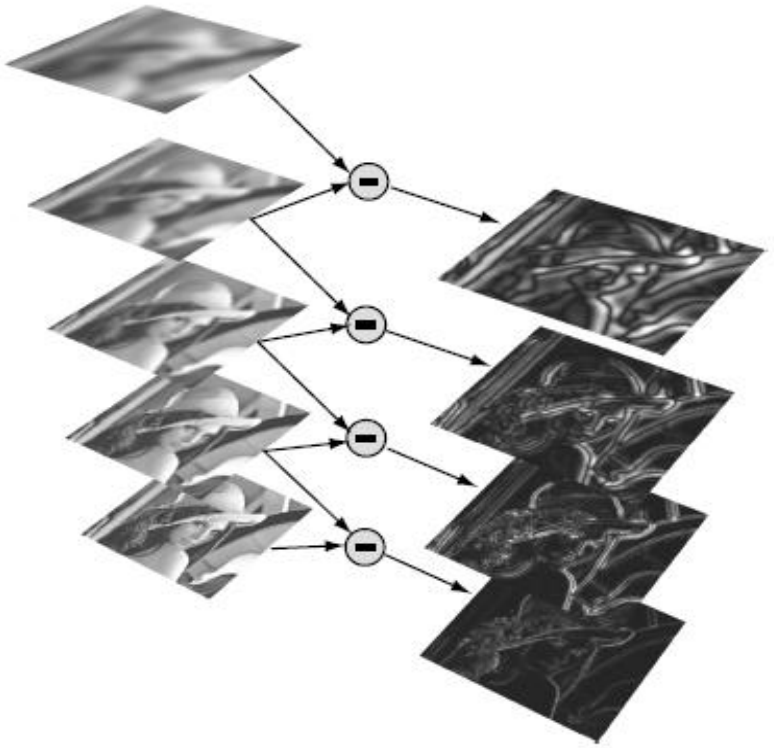
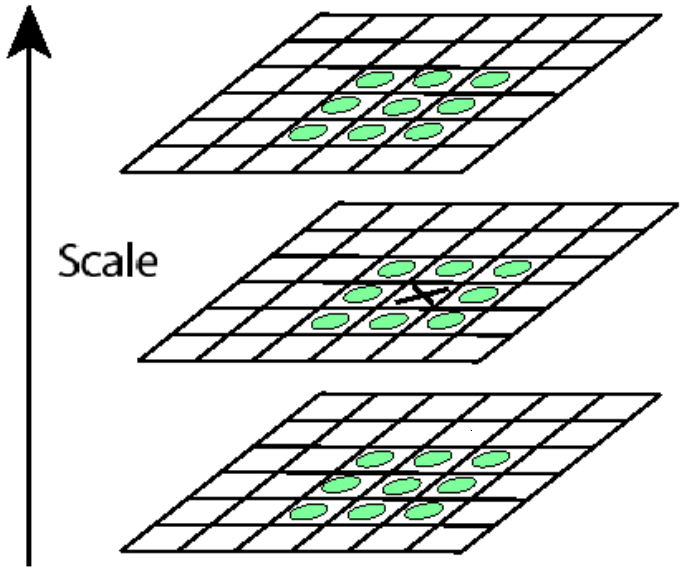


Figure 9.1: A Difference-of-Gaussian octave. The five images in the left stack are incrementally smoothed versions of the input image. The right stack shows the resulting DoG.

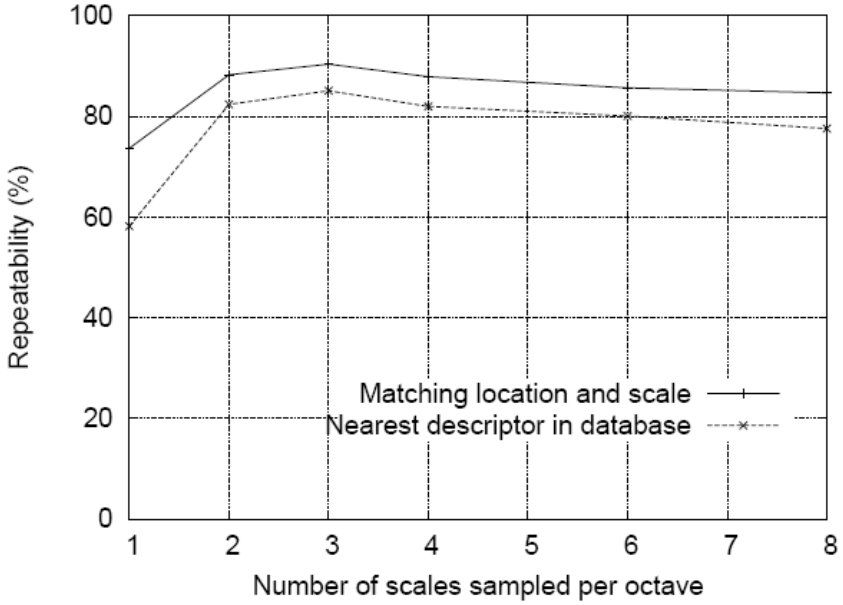
Slide from N. Vassilieva

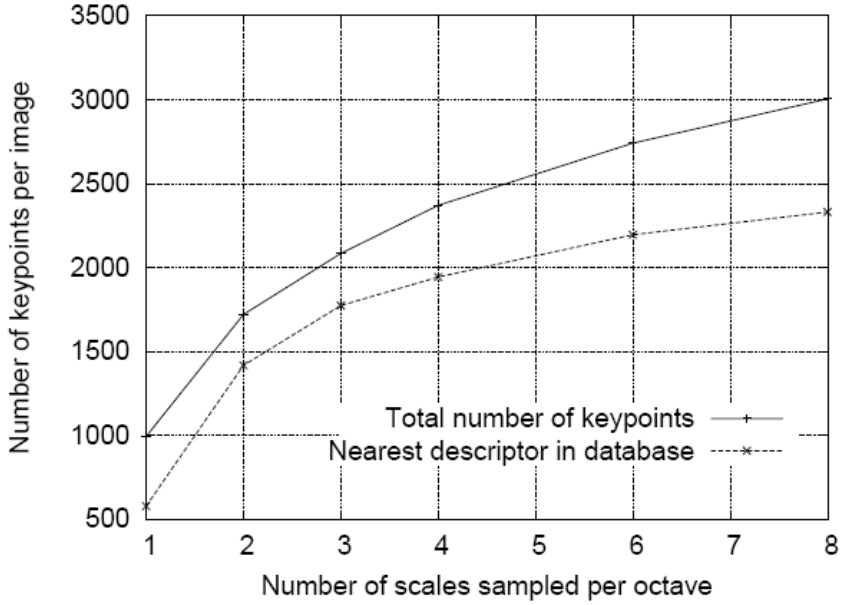
Key point localization



- Search for maxima and minima of the DoG
- Each point is compared with 8 neighbours on its scale and 9 on the previous and next scale levels

Scale sampling frequency





- More points if more scale levels are used
- On the other side the matching accuracy decreases (less stable points)
- With *s=3* maximum accuracy (Lowe)

Scale Invariant Feature Transform (Lowe 2004)

4 Steps :

1. Scale-space extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Step 2 : Keypoint localization

- Keypoint localization in the maxima points is not precise
- Taylor series approximation of the DoG in the neighboorhood of the point to accurately locate the maxima
- Keep only the stablest keypoints (where the gradient is bigger than a threshold)

Taylor approximation

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathbf{T}}\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1}\frac{\partial D^{\mathbf{T}}}{\partial \mathbf{x}}. \qquad D(\hat{\mathbf{x}}) = D + \frac{1}{2}\frac{\partial D^T}{\partial \mathbf{x}}\hat{\mathbf{x}}$$

- Taylor series of the second order centred in the identified point
- Zero-crossing of the derivative of the approximation: accurate location of the maxima
- Discard points with $\left|D(\hat{x})\right| < 0.03$

Avoid edges

$$\mathrm{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

1
$$\mathrm{Tr}(\mathrm{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$
$$\mathrm{Det}(\mathrm{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

$$\alpha = r\beta$$

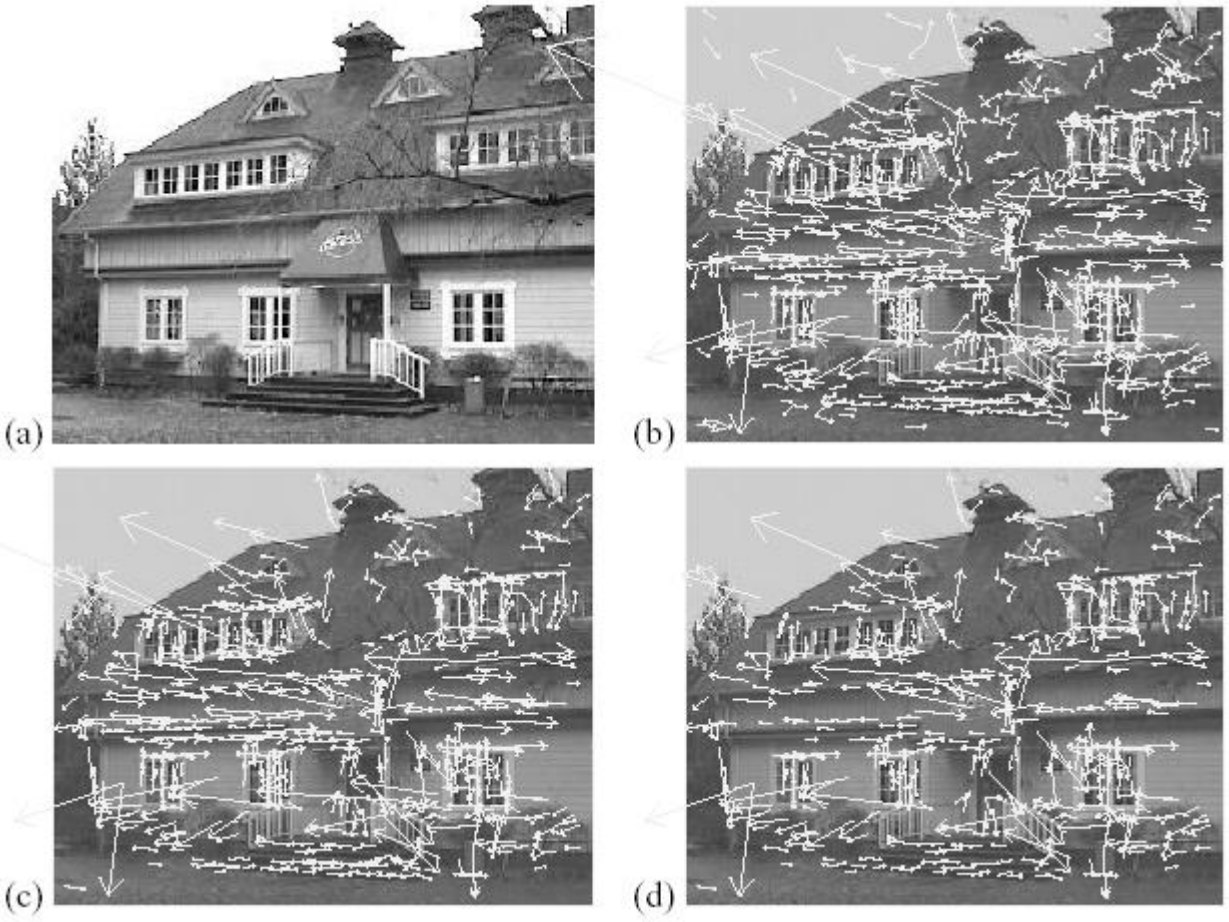
2
$$\frac{\mathrm{Tr}(\mathrm{H})^2}{\mathrm{Det}(\mathrm{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r},$$

3
$$\frac{\mathrm{Tr}(\mathrm{H})^2}{\mathrm{Det}(\mathrm{H})} < \frac{(r+1)^2}{r}.$$

- DoG has a strong response at the edges, but the edges are not very good as features
- Similar to the Harris detector: the edge is located where the Hessian has a large eigenvalue and a small one
- Threshold on the eigenvalue ratio α/β

Example

Threshold on the peak value of the DoG and in the ratio of the main curvatures (similar to the Harris approach)



(a) *233x189 image*
(b) *832 DOG extrema*
(c) *729 left after peak value threshold*
(d) *536 left after testing ratio of principle curvatures*

Scale Invariant Feature Transform (Lowe 2004)

4 Steps :

1. Scale-space extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Step 3: Orientation assignment

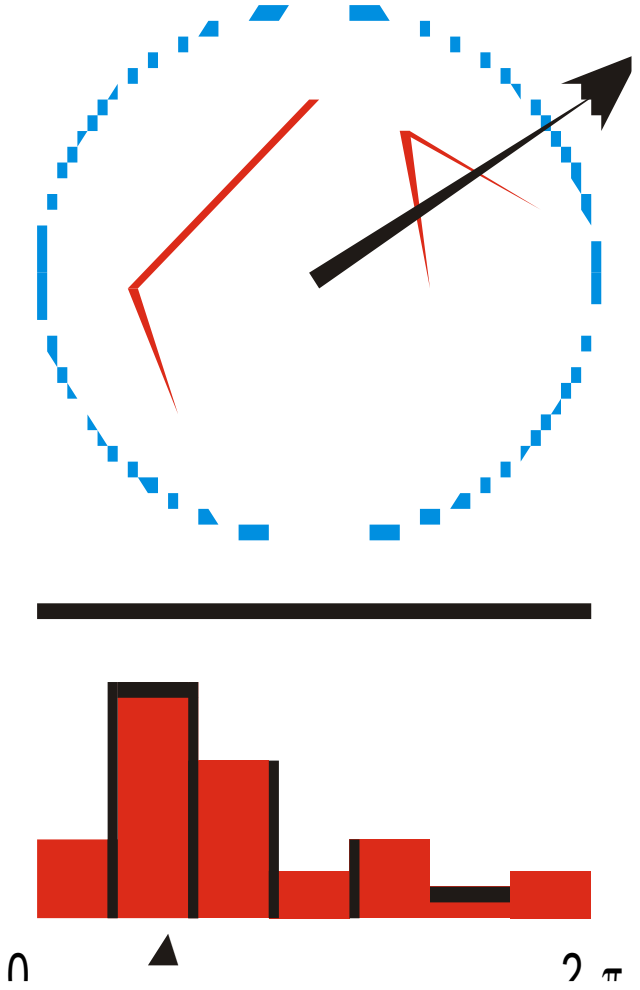
$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

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

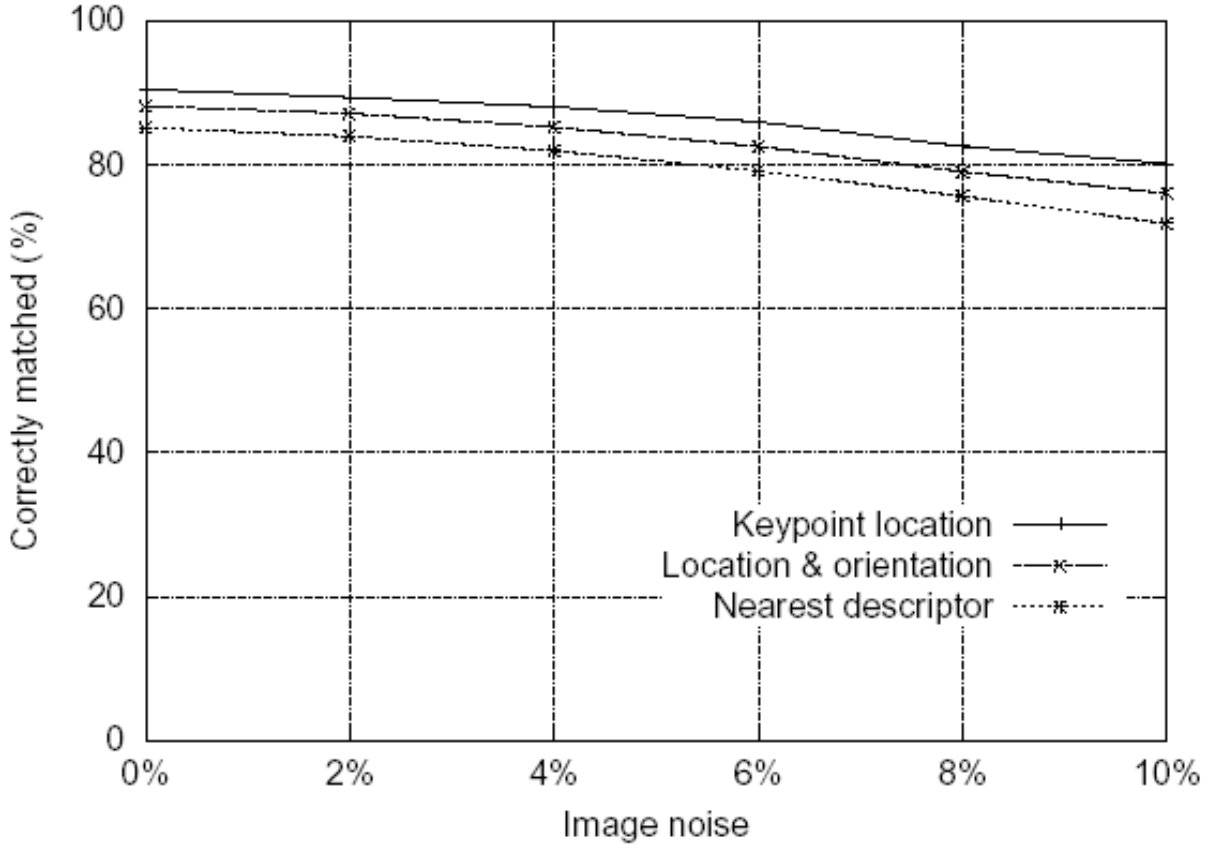
- An orientation is associated to each keypoint
- Based on the gradient
- All the subsequent operations for the keypoint will be made on an image rotated with respect to the found orientation and scaled according to the keypoint scale
- Makes the description invariant with respect to scale and rotation

Computation of the orientation

- An histogram (with 36 bins) of the gradient directions at the chosen scale in a neighbourhood of the point is built
- Each sample is weighted with his gradient module *m(x,y)*
- Gaussian smoothing is applied
- The orientation is the direction corresponding to the peak value
- Multiple orientations if close peaks
- Parabola fitting
- Stable coordinates for each point (x, y, scale, orientation)



Stability



Position and orientation are robust w.r.t noise

Scale Invariant Feature Transform (Lowe 2004)

4 Steps :

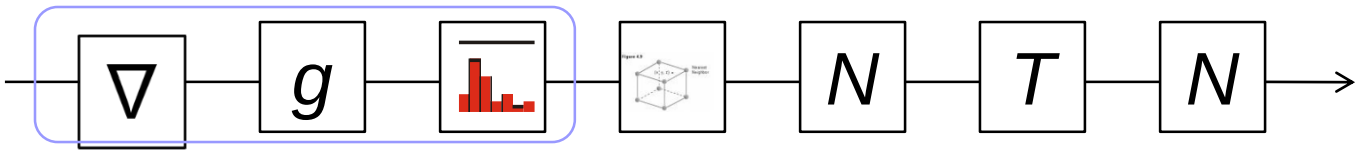
1. Scale-space extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Fase 4: Descriptors

- At each keypoint :
  - Position
  - Scale
  - Orientation
- Invariance to these parameters
- Descriptors should also be invariant to
  - Illumination
  - Change of viewpoint
- Furthermore it should be possible to discriminate between the points

Descriptor computation (1)

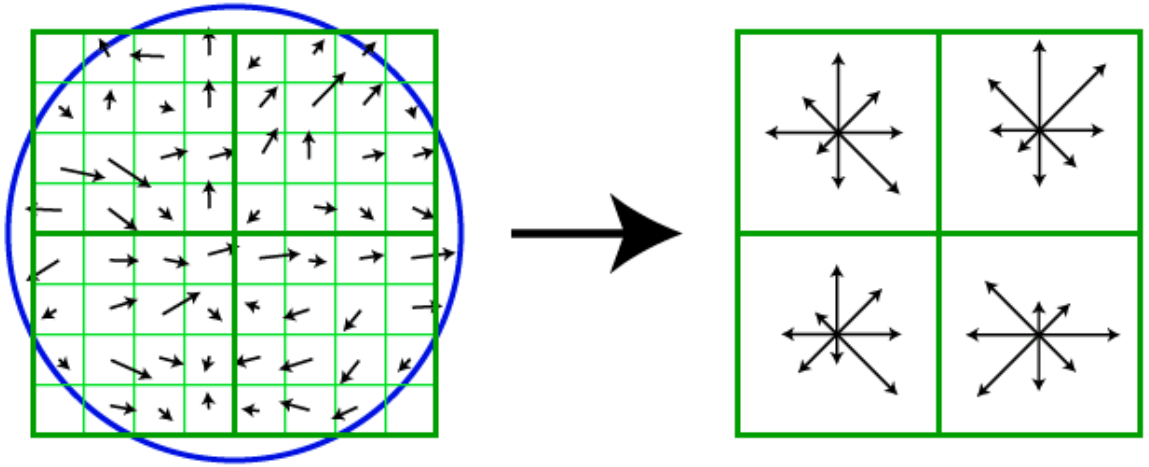
1. Compute magnitude and direction of the gradient in a neighbourhood of the keypoint (usually 16x16) with the orientation and scale of the keypoint
2. The values are weighted with a Gaussian function centered in the keypoint

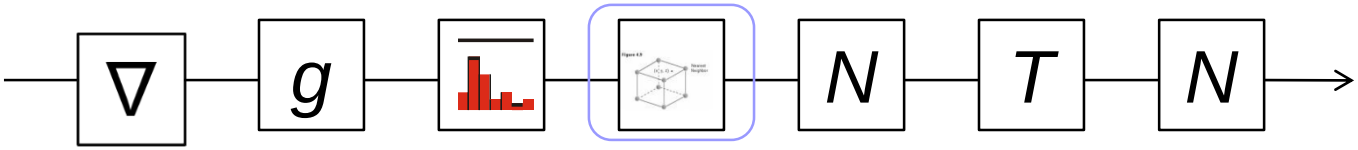


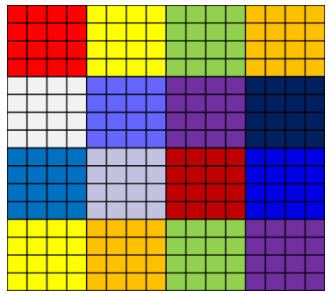
Descriptor computation (2)

3: Histogram

- The window is divided in regions of 4x4 pixel
- For each region an histogram with 8 values corresponding to 8 intervals of 45° are built
- For each interval the sum of the gradients modules of the samples with gradient direction falling in that interval is computed
- Descriptor dimension is (16/4) x (16/4) x 8 = 128 elements

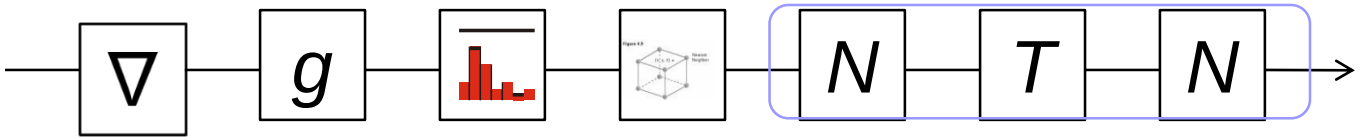






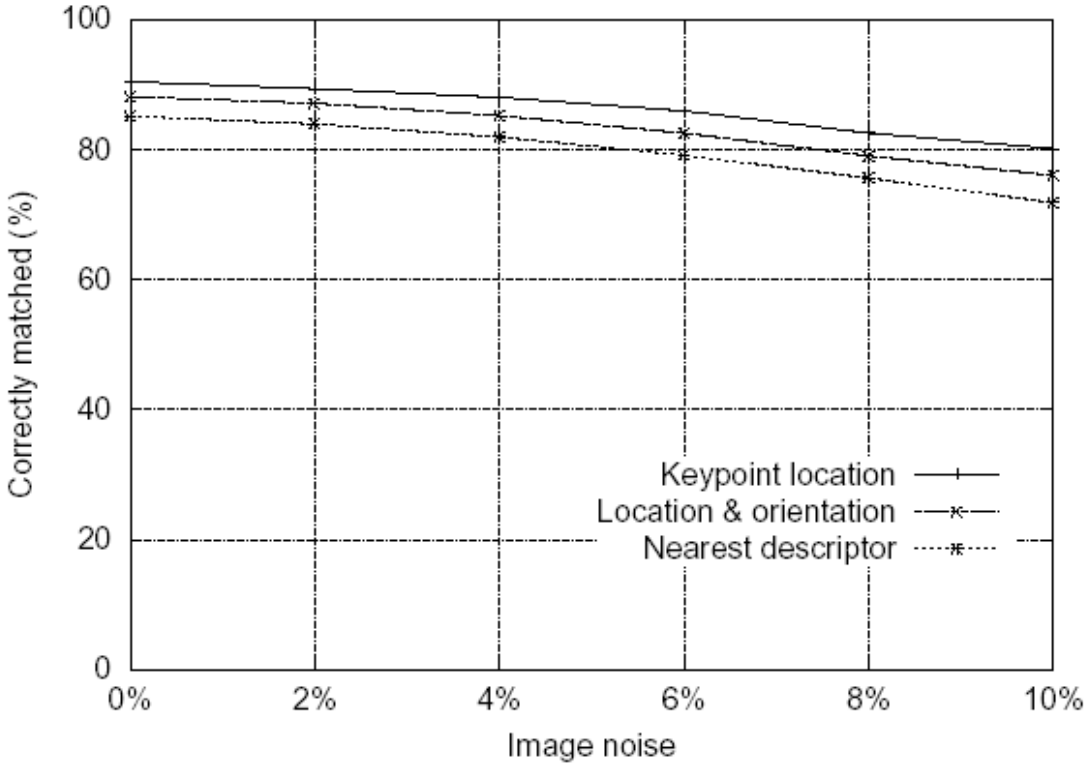
Descriptor computation (3)

4. Trilinear filter on all the 3 dimensions for a better stability
5. Normalization to 1 for robustness to illumination
6. Threshold to 0,2 on all the components (the large ones are sensible to light)
7. Further normalization to 1



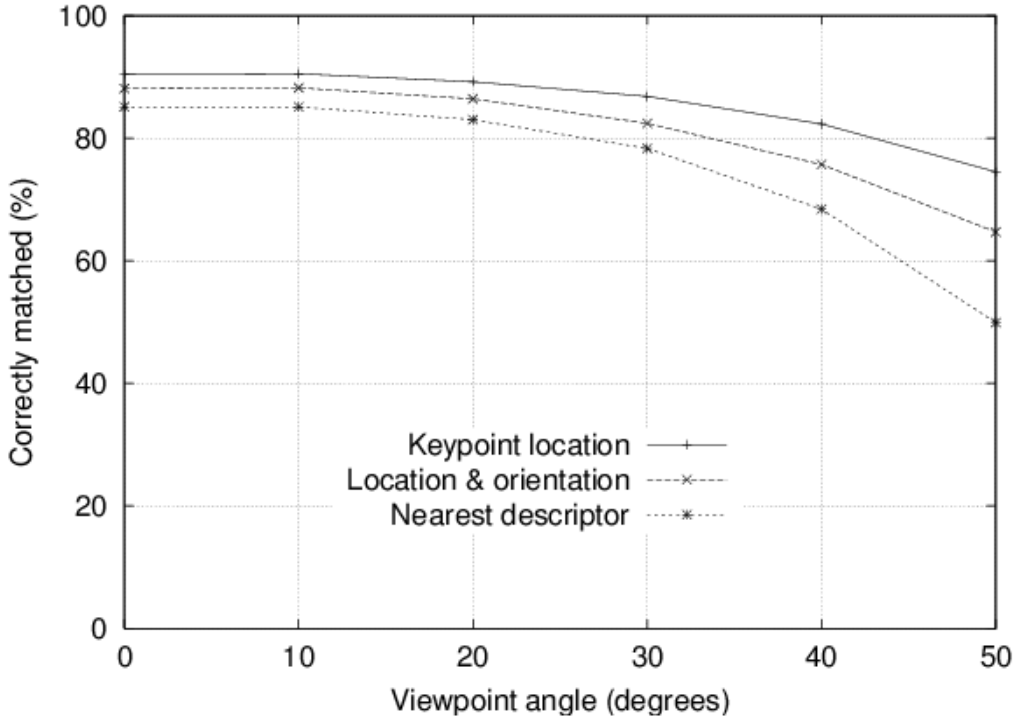
Robustness to noise

- Feature matching after random rotation and scale change with different levels of noise
- Nearest neighbour on 30,000 features database



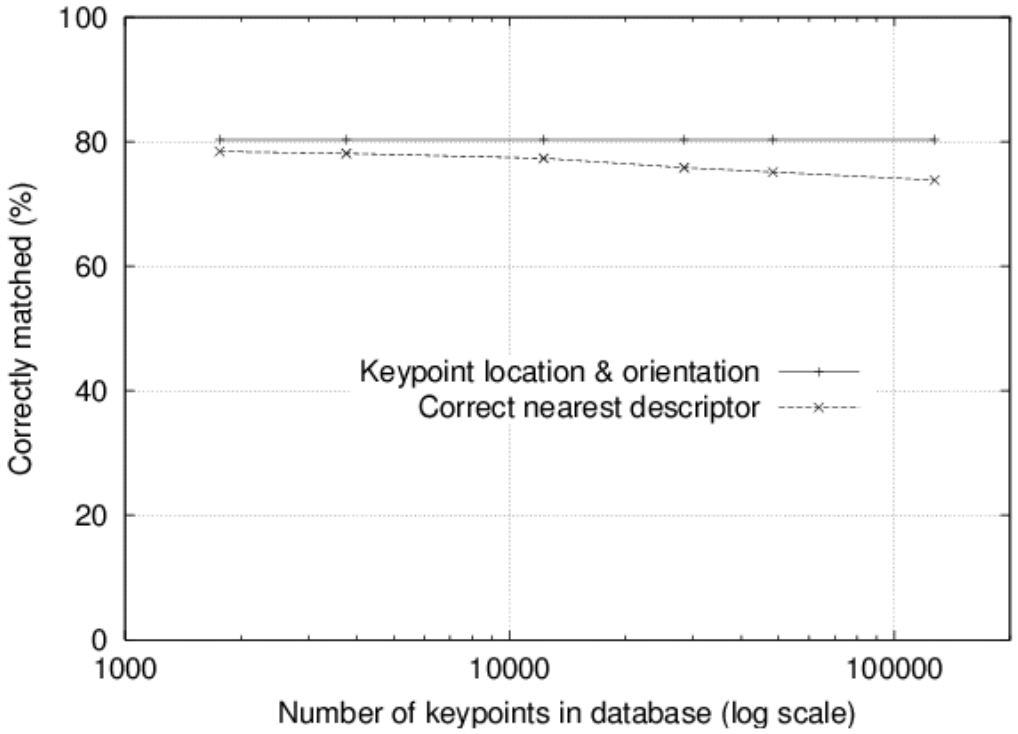
Robustness to affine transforms

- Feature matching after random scale change and rotation with 2% noise and affine distortion
- Nearest neighbour on 30,000 features database
- Good for small angles

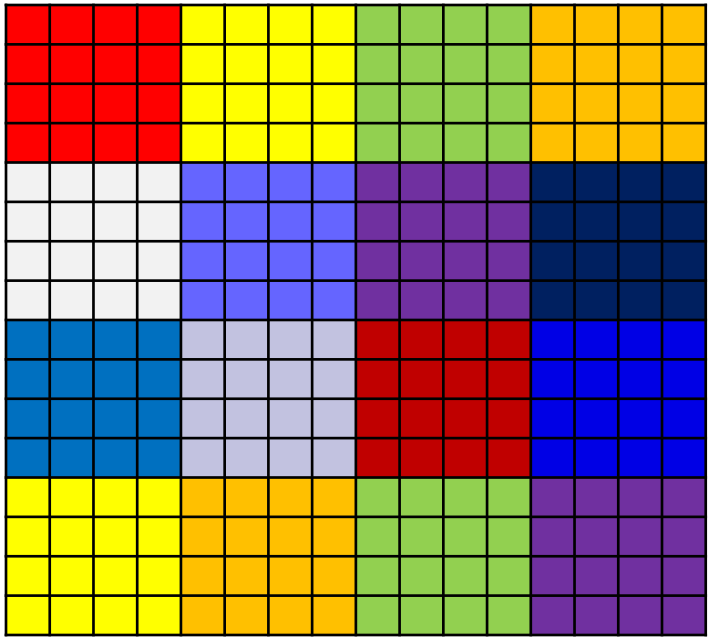


Features distinctiveness

- Variable size database, 2% noise and 30° viewpoint change
- Percentage of correct nearest neighbour matches



Descriptors: synthesis



- At each keypoint :
  - Position
  - Scale
  - Orientation
- Descriptor: normalized and filtered histogram of the gradient with respect to orientation in a 16x16 region around the keypoint
- 4x4x8=128 dimensions vector
  - 4x4= 16 regions of 4x4 pixel
  - Histogram quantized on 8 directions

Part 3: Matching and Recognition with SIFT descriptors

In this part figures from Lowe's paper

Application: Matching and Recognition

- Match between SIFT features of the object and of the image
- Large database: difficult matching
- Find 1% inliers in 99% outliers

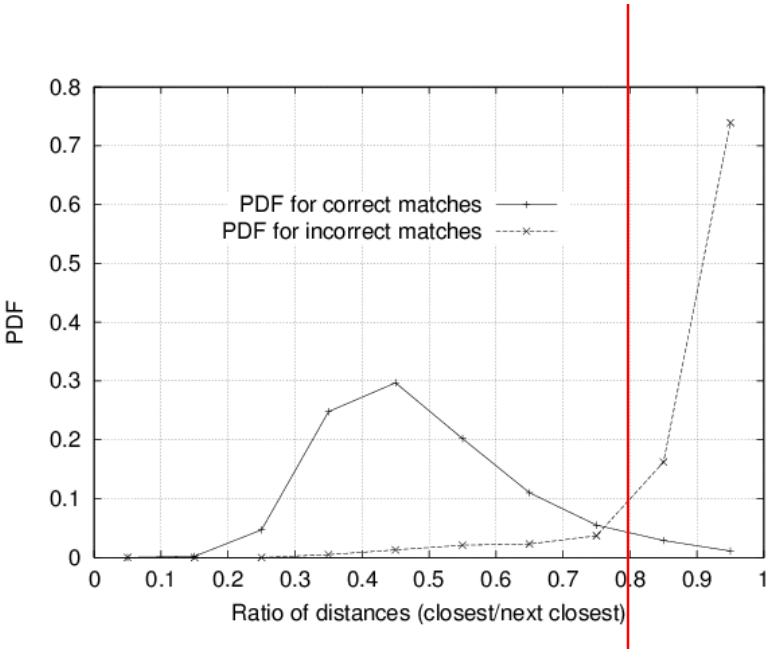
Simple solution

1. Compute SIFT DB for each object
2. Compute SIFT DB on the image
3. For each image keypoint find the closest in the object DB (with Euclidean distance)

- It does not work !
  - Too many matches
  - Even with threshold on the distance low results

Find correct matches

$$R = \frac{\|p - m_1\|}{\|p - m_2\|} > 0.8$$



- Compare the nearest neighbour with the second closest (the closest among the ones taken from another object)
- 0.8 threshold on the two distances ratios
- Remove 90% of wrong matches and 5% of the correct ones
- Approximate fast search with BBF algorithm (Beis and Lowe, 97)

Find groups of consistent clusters

- Search for clusters of 3 consistent feature match over 3000 feature matches !

- **Hough Transform**
  - Find consistent sets of 3 points in a 4 parameters space
  - Insert in the 2 closest bins
  - Select bins with at least 3 *votes*

Parametro	Bin size
Position (x)	0.25 D_{max}
Position (y)	0.25 D_{max}
Rotation	30°
Scale	2x

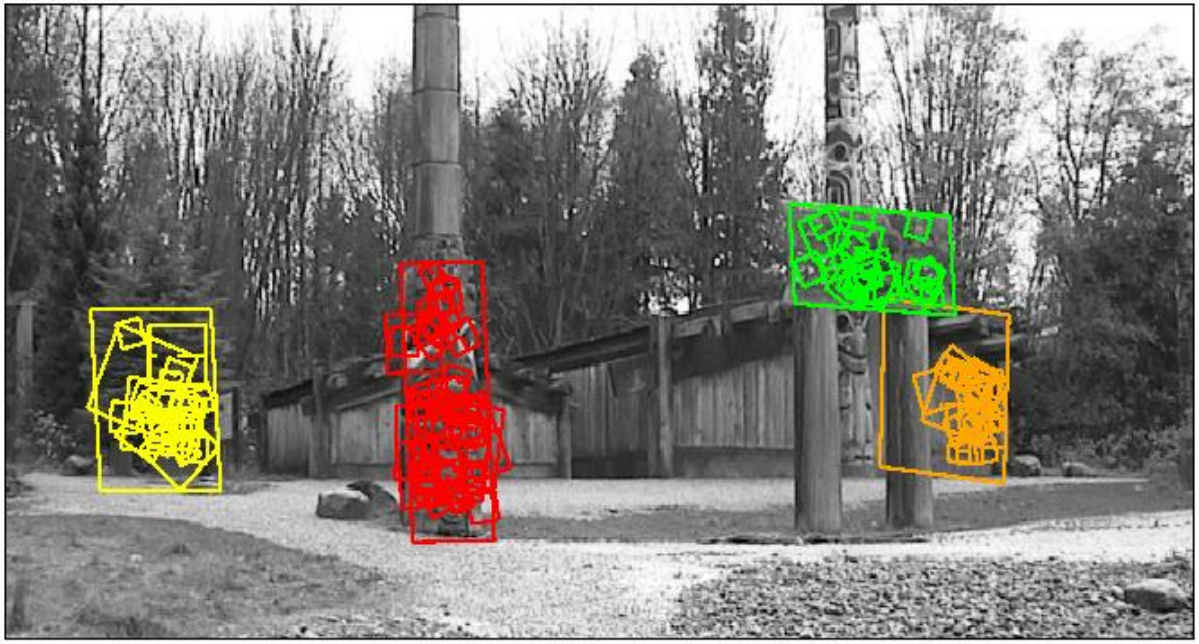
Matching procedure

1. Check all the clusters with at least 3 features
2. Find the affine transform corresponding with a least squares approximation
3. Discard the points that do not satisfy the found transform
4. Estimate the probability of a correct match

Synthesys

1. Find closest keypoint (Euclidean distance)
2. Threshold with the ratio from the distance of the second closest point
3. Hough transform (find groups of consistent keypoints)
4. Estimate affine transform corresponding with the least square approximation
5. Final consistency check of the keypoints with the computed transform

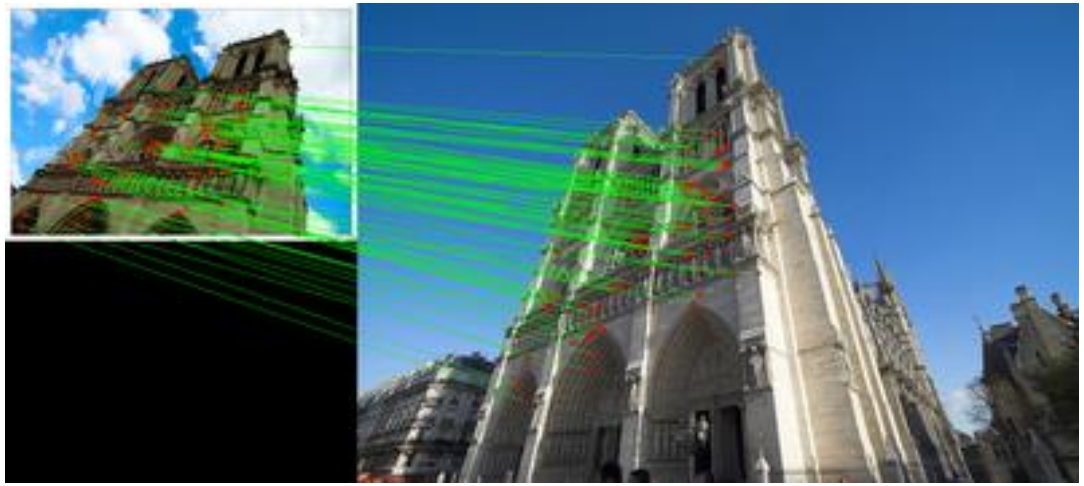
Example: Object Recognition (1)

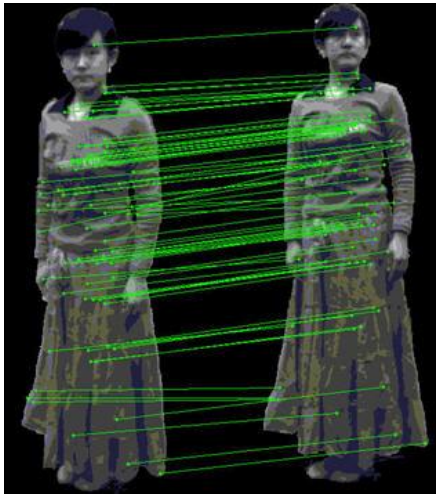


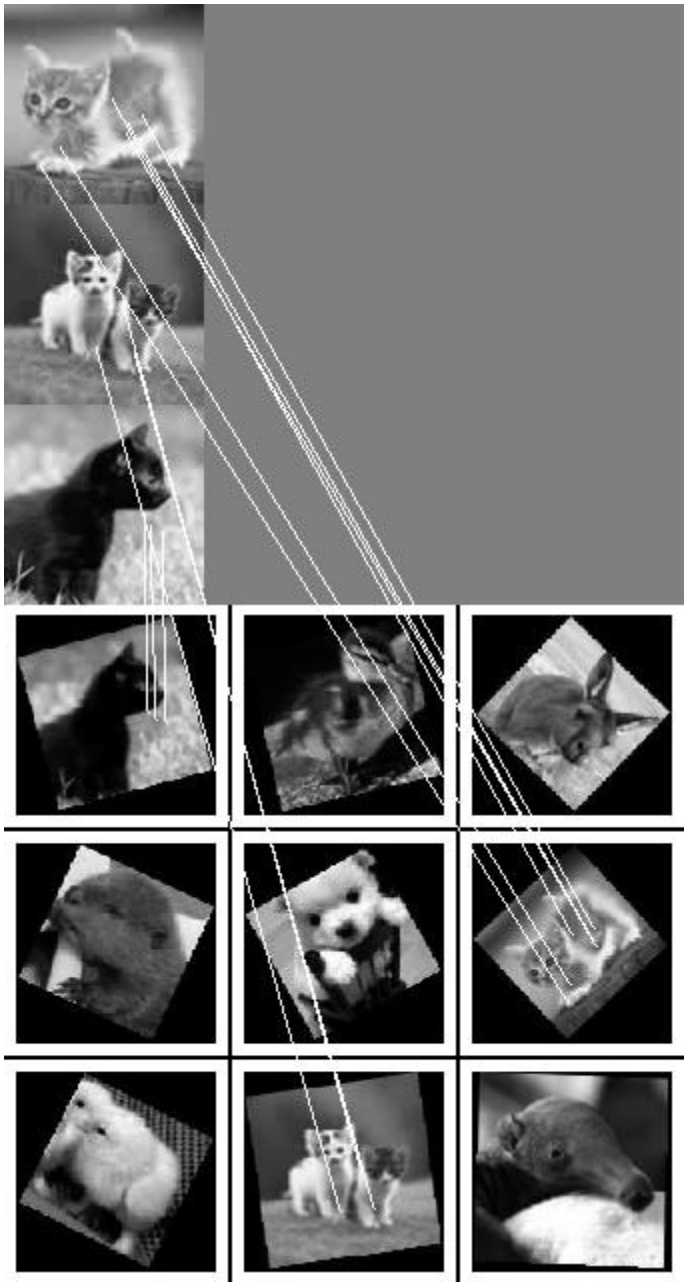
Example: object recognition (2)

- Search for the 2 objects
- Works even in the presence of occlusions

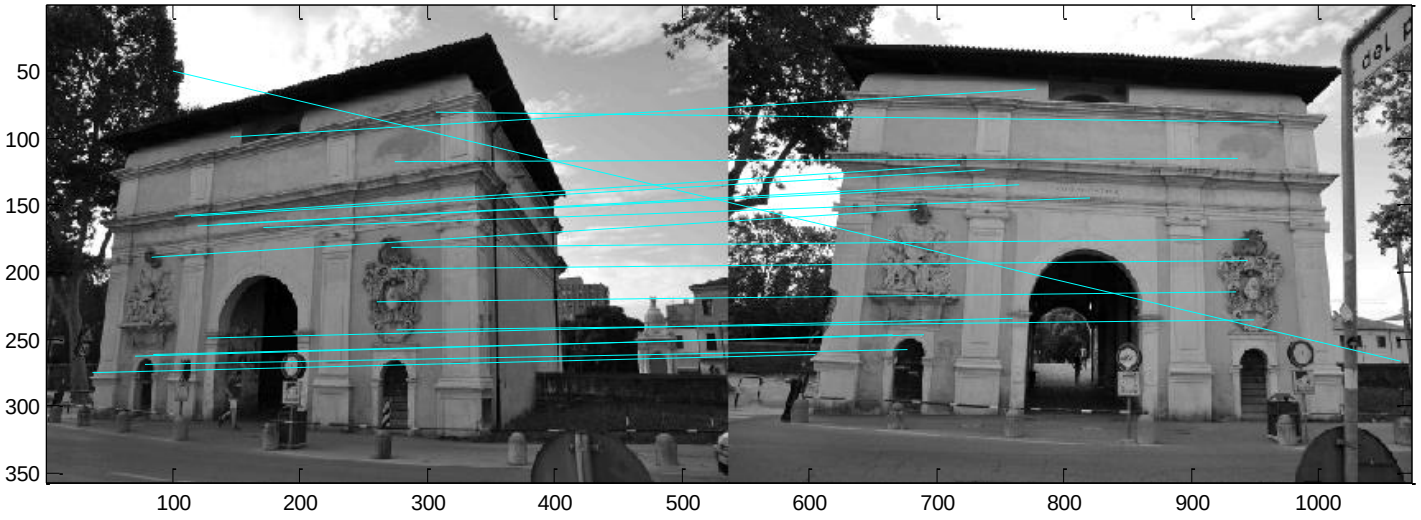
Example:
Image matching







Example: Image mosaicing

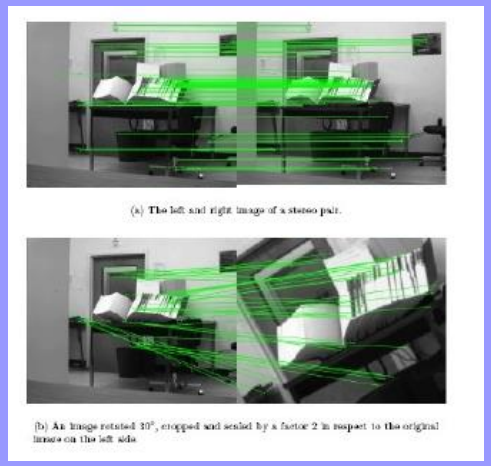


Matlab tool from the author of the method

Part 4
Beyond SIFT: Recent feature detectors and descriptors

Several slides in this part from George Bebis and Vedaldi et Al, "modern feature descriptors"

Beyond SIFT:
Variations derived from the approach

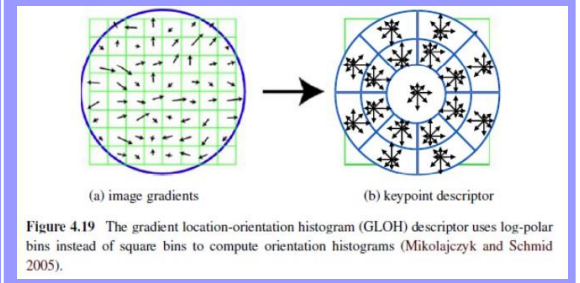


(a) The left and right image of a stereo pair.

(b) An image rotated 30°, cropped and scaled by a factor 2 in respect to the original images on the left side

PCA-SIFT

SURF



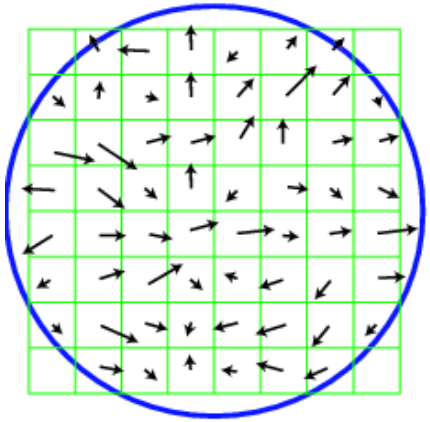
(a) image gradients (b) keypoint descriptor

Figure 4.19 The gradient location-orientation histogram (GLOH) descriptor uses log-polar bins instead of square bins to compute orientation histograms (Mikolajczyk and Schmid 2005).

GLOH

PCA-SIFT

- Steps 1-3 are the same; Step 4 is modified.
- Take a 41 x 41 patch at the given scale, centered at the keypoint, and normalized to a canonical direction.

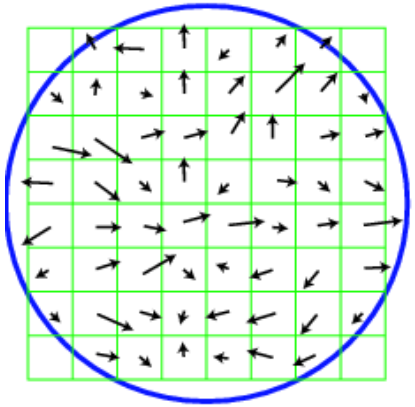


Thanks to George Bebis

Yan Ke and Rahul Sukthankar, "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors", Computer Vision and Pattern Recognition, 2004

PCA-SIFT

- Instead of using weighted histograms, concatenate the horizontal and vertical gradients (39 x 39) into a long vector.
- Normalize vector to unit length.

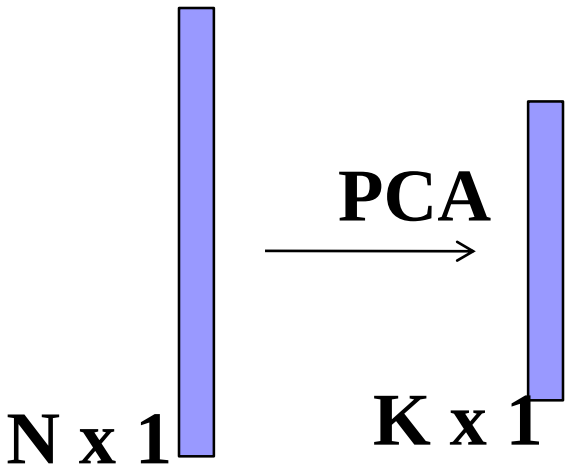


2 x 39 x 39 = 3042 vector

Thanks to George Bebis

PCA-SIFT

- Reduce the dimensionality of the vector using Principal Component Analysis (PCA)
  - e.g., from 3042 to 36



- Some times, less discriminatory than SIFT.

Thanks to George Bebis

SURF: Speeded Up Robust Features

- Speed-up computations by fast approximation of Hessian matrix and descriptors using “integral images”.

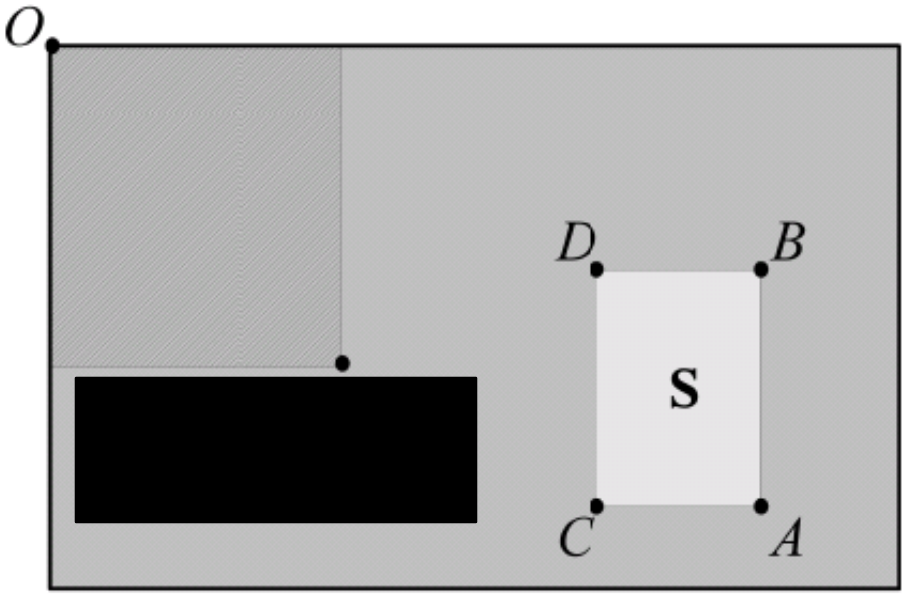
$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix},$$

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, “SURF: Speeded Up Robust Features”, European Computer Vision Conference (ECCV), 2006.

Thanks to George Bebis

Integral Image

- The integral image $I_{\Sigma}(x,y)$ of an image I(x, y) represents the sum of all pixels in I(x,y) of a rectangular region formed by (0,0) and (x,y).



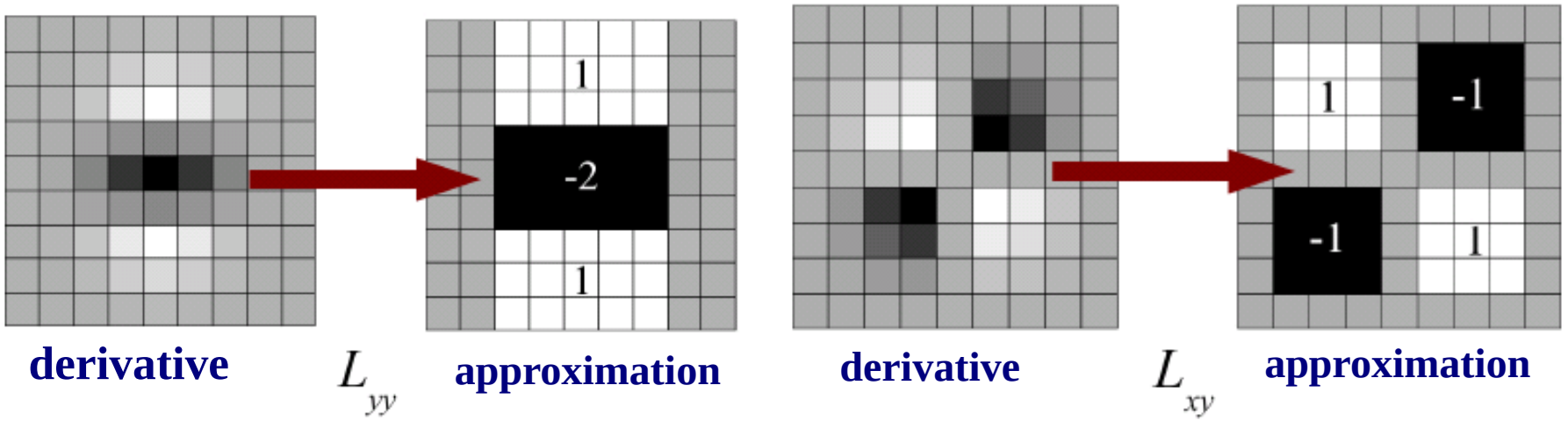
Using integral images, it takes only four array references to calculate the sum of pixels over a rectangular region of any size.

$$\mathbf{S}=A-B-C+D$$

Approximation with box filters

- Approximate L_{xx}, L_{yy}, and L_{xy} using box filters.

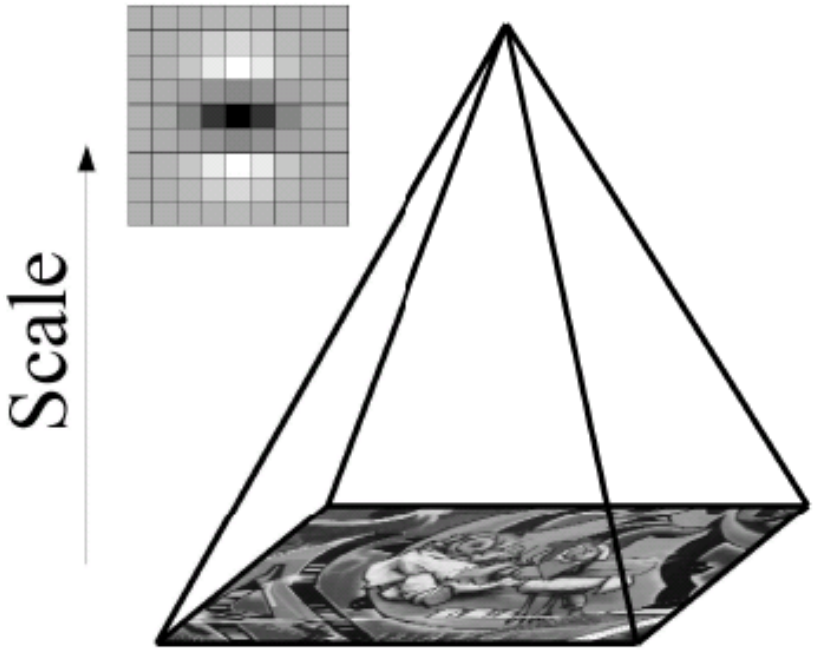
(box filters shown are 9 x 9 – good approximations for a Gaussian with σ=1.2)



- Can be computed very fast using integral images!

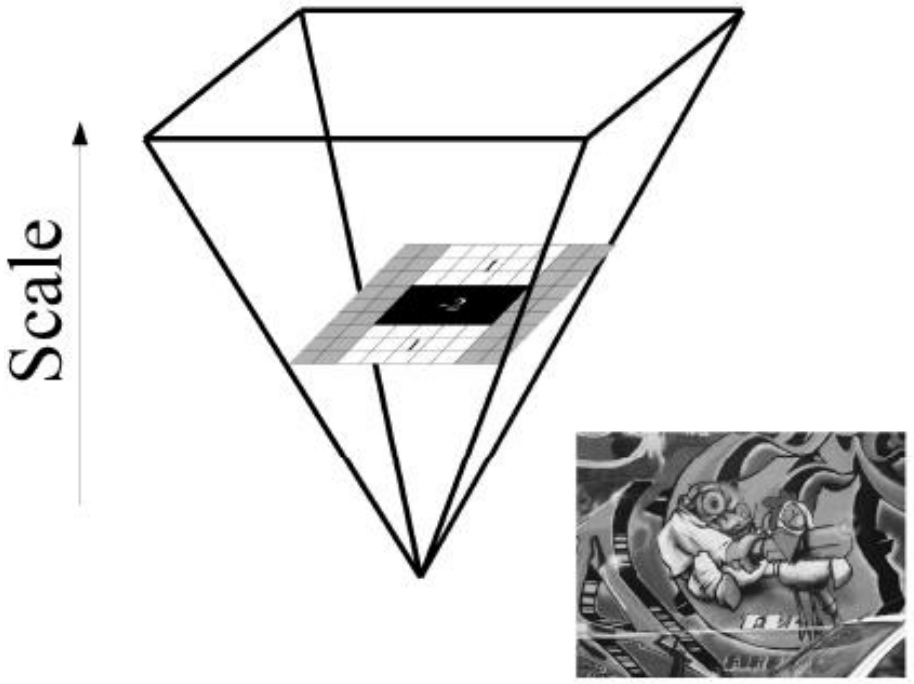
Scale-space (1)

- In SIFT, images are repeatedly smoothed with a Gaussian and subsequently sub-sampled in order to achieve a higher level of the pyramid.



Scale-space (2)

- Alternatively, we can use filters of larger size on the original image.
- Due to using integral images, filters of any size can be applied at exactly the same speed!



SURF: Approximate with box filters

- **Approximation of H:**

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix},$$

Using DoG

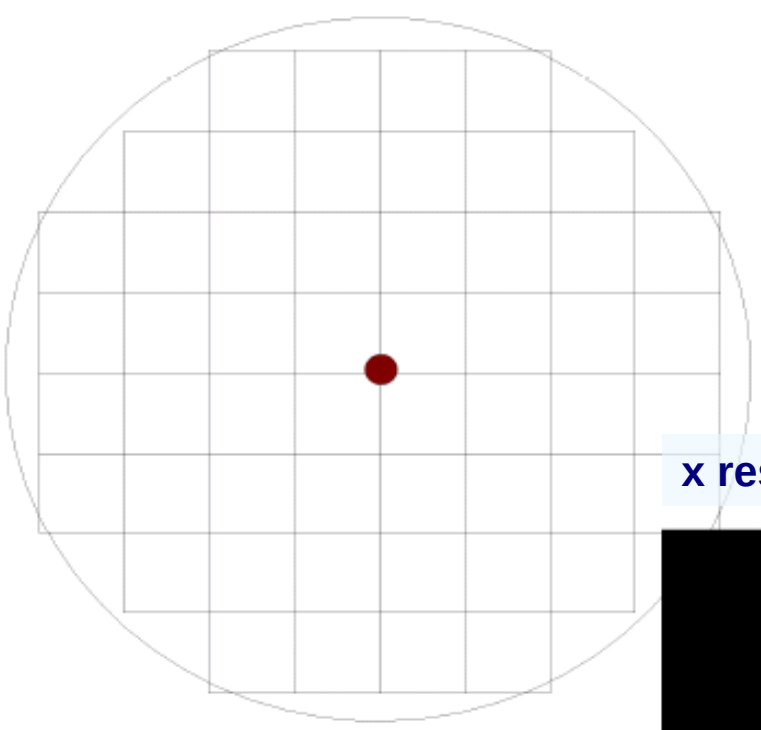
$$SIFT: \; H^{SIFT}_{approx} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

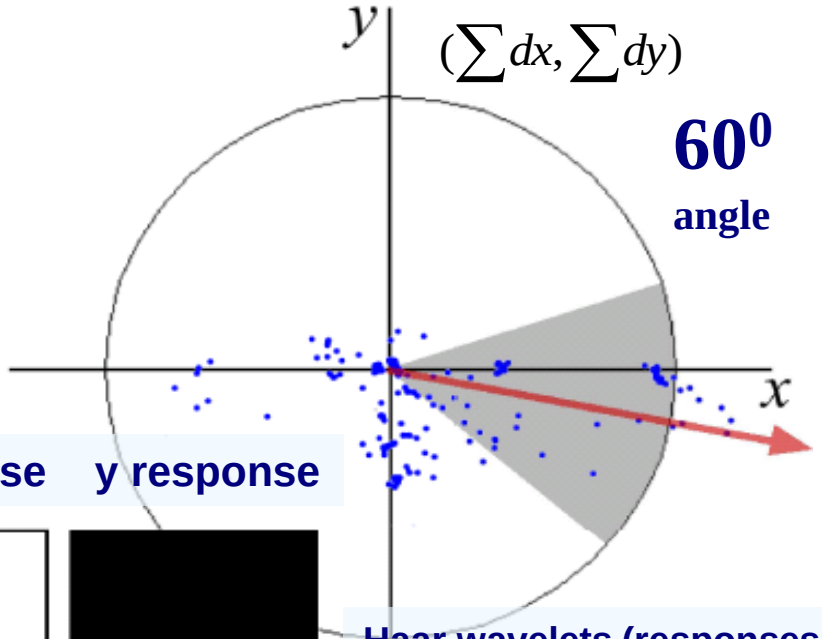
Using box filters

$$SURF: \; H^{SURF}_{approx} = \begin{bmatrix} \hat{L}_{xx} & \hat{L}_{xy} \\ \hat{L}_{yx} & \hat{L}_{yy} \end{bmatrix}$$

Orientation assignment

Circular neighborhood of
radius 6σ around the interest point
(σ = the scale at which the point was detected)





x response y response

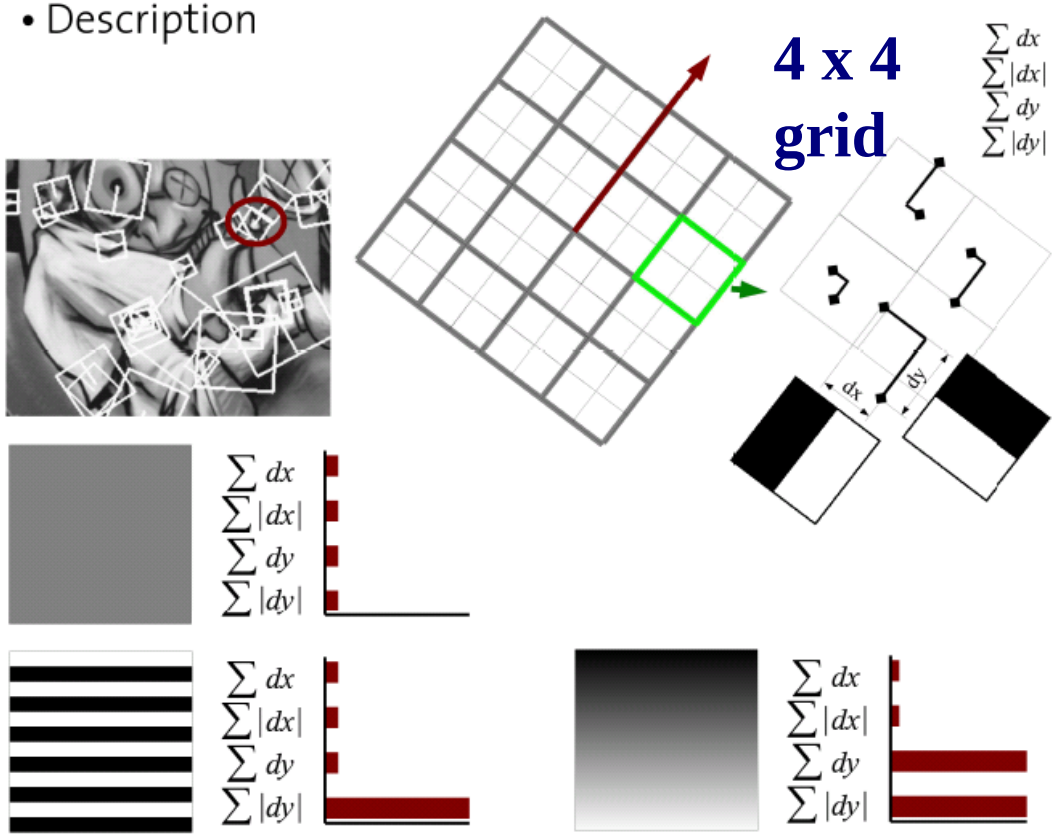
Haar wavelets (responses weighted with Gaussian)
Side length = 4σ

Can be computed very fast using integral images!

Keypoint descriptor (1)

- **Keypoint descriptor (square region of size 20σ)**

• Description

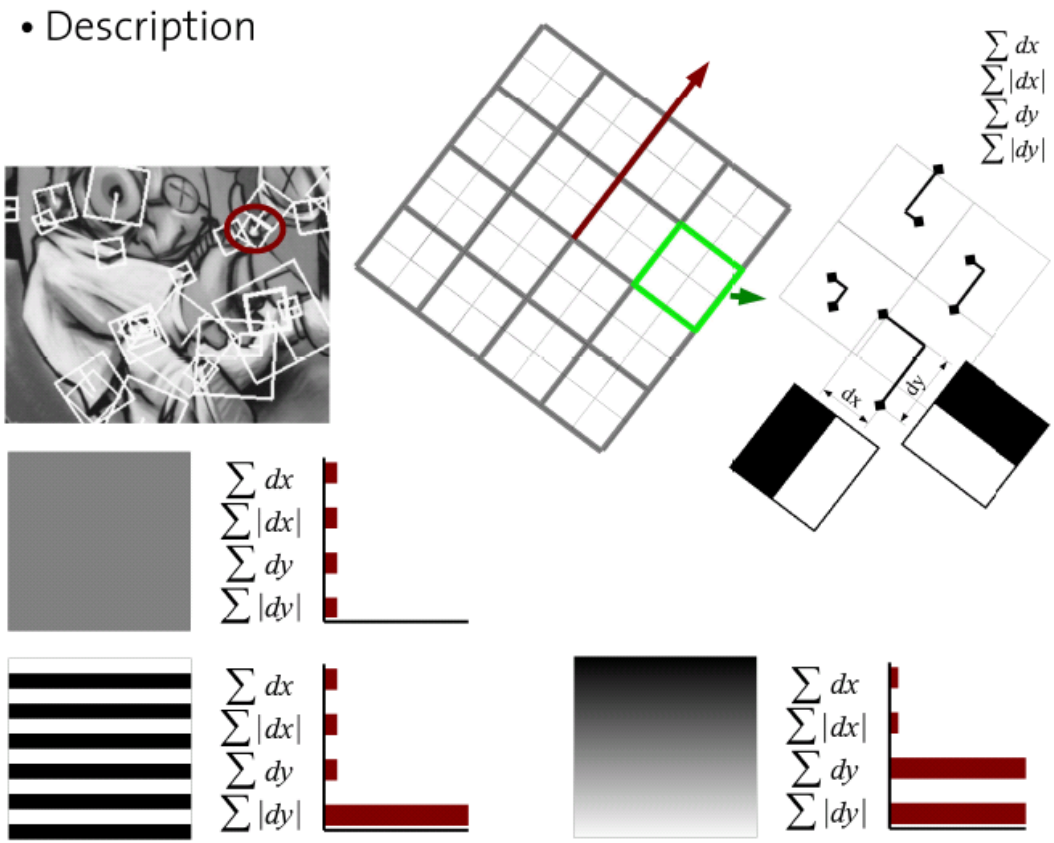


- Sum the response over each sub-region for d_x and d_y separately.
- To bring in information about the polarity of the intensity changes, extract the sum of absolute value of the responses too.

Feature vector size:

4 x 16 = 64

Keypoint descriptor (2)



- SURF-128
  - The sum of d_x and $|d_x|$ are computed separately for points where $d_y < 0$ and $d_y > 0$
  - Similarly for the sum of d_y and $|d_y|$
  - **More discriminatory!**

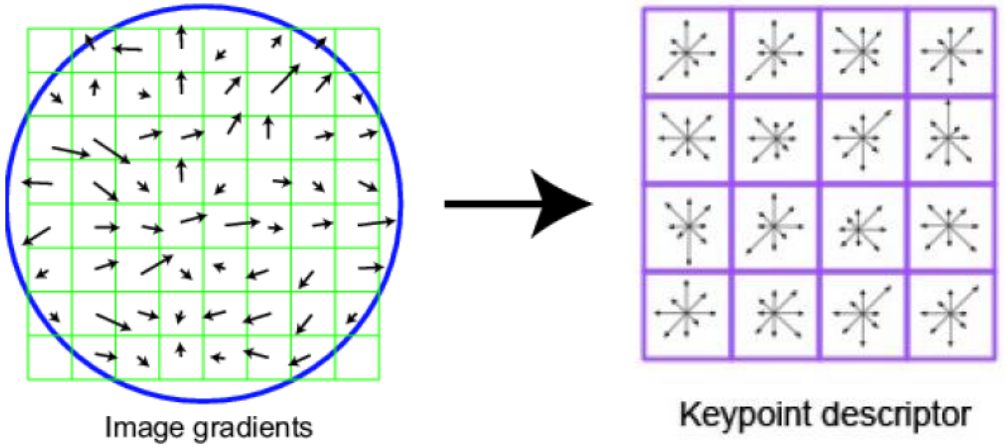
SURF: Speeded Up Robust Features

- Has been reported to be 3 times faster than SIFT.
- Less robust to illumination and viewpoint changes compared to SIFT.

K. Mikolajczyk and C. Schmid,"A Performance Evaluation of Local Descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pp. 1615-1630, 2005.

Gradient location-orientation histogram (GLOH)

- Compute SIFT using a log-polar location grid:
  - 3 bins in radial direction (i.e., radius 6, 11, and 15)
  - 8 bins in angular direction
- Gradient orientation quantized in 16 bins.
- Total: (2x8+1)*16=272 bins →PCA.



K. Mikolajczyk and C. Schmid,"A Performance Evaluation of Local Descriptors", IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pp. 1615-1630, 2005.

Beyond SIFT:
Other Approaches

- *Shape context*
- *LBP*
- *BRIEF*
- *ORB*
- *MSER*

And many others!...

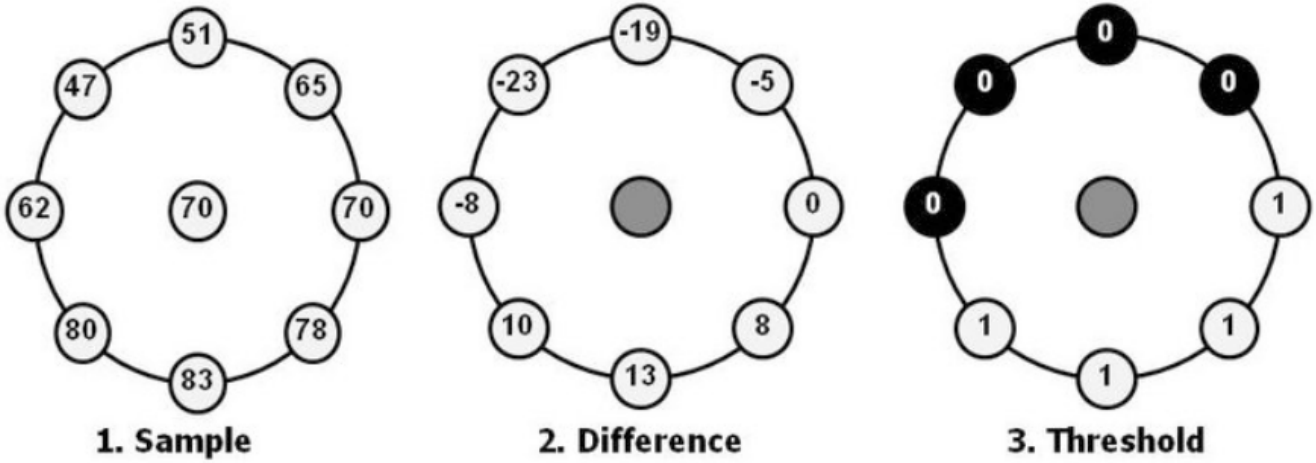
Shape Context

- A 3D histogram of edge point locations and orientations.
  - Edges are extracted by the Canny edge detector.
  - Location is quantized into 9 bins (using a log-polar coordinate system).
  - Orientation is quantized in 4 bins (i.e., horizontal, vertical, and two diagonals).
- Total number of features: 4 x 9 = 36

K. Mikolajczyk and C. Schmid,"A Performance Evaluation of Local Descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pp. 1615-1630, 2005.

LBP:
Local Binary Patterns

- First proposed for texture recognition in 1994.



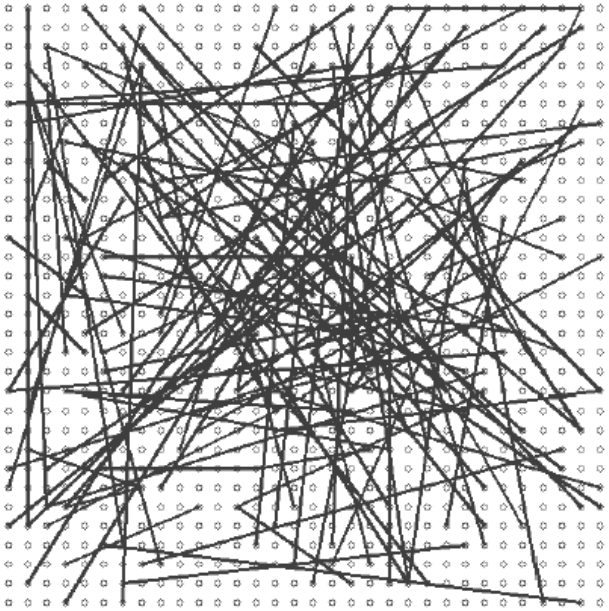
Citations: 2500 (2012)

T. Ojala, M. Pietikäinen, and D. Harwood (1994), "Performance evaluation of texture measures with classification based on Kullback discrimination of distributions", ICPR 1994, pp.582-585.

M Heikkilä, M Pietikäinen, C Schmid, Description of interest regions with LBP, Pattern recognition 42 (3), 425-436

BRIEF:
Binary Robust Independent Elementary Features

- Random selection of pairs of intensity values.
- Fixed sampling pattern of 128, 256 or 512 pairs.
- Hamming distance to compare descriptors (XOR).



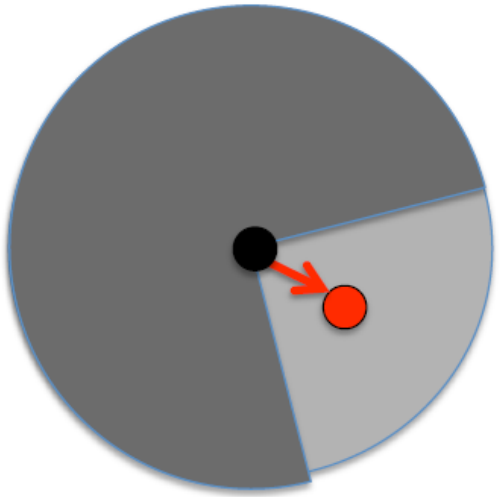
Citations:
149 (2012)

M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: Binary Robust Independent Elementary Features, 11th European Conference on Computer Vision, 2010.

ORB:
Oriented FAST and Rotated BRIEF

- Add rotation invariance to BRIEF
- Orientation assignment based on the intensity centroid

Citations:
43 (2012)



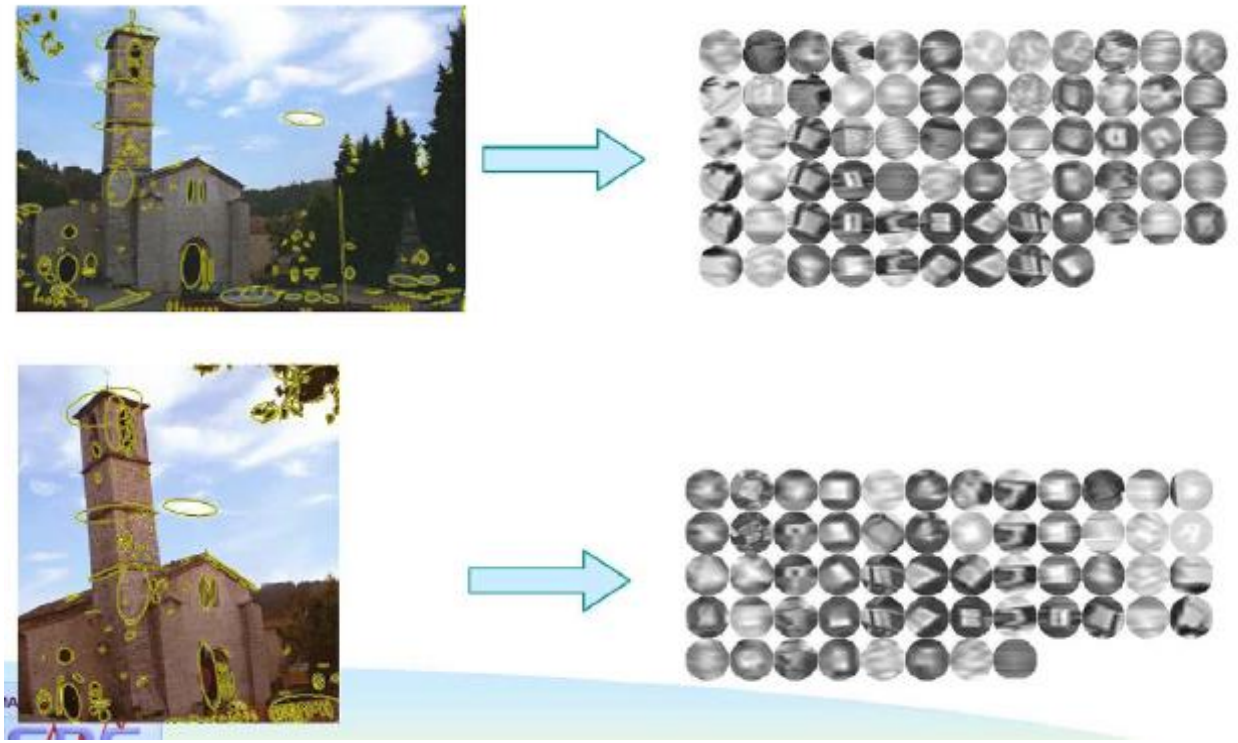
Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski, ORB: an efficient alternative to SIFT or SURF, ICCV 2011

Maximally Stable Extremal Regions (Matas et al 2002)

- Based on watershed algorithm
  - Consecutive image thresholding by all thresholds
  - Maintain list of Connected Components
  - Regions = Connected Components with stable area (or some other property) over multiple thresholds selected

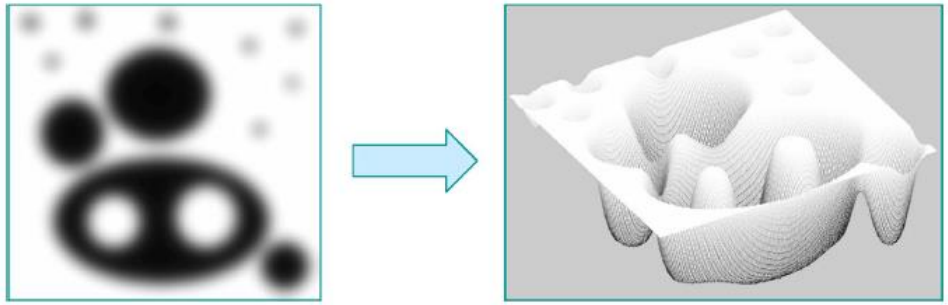
citations
1300 (2010)
1620 (2012)

MSER Regions



- MSER regions are connected areas characterized by almost uniform intensity, surrounded by contrasting background.
- They are constructed through a process of trying multiple thresholds.
- The selected regions are those that maintain unchanged shapes over a large set of thresholds.

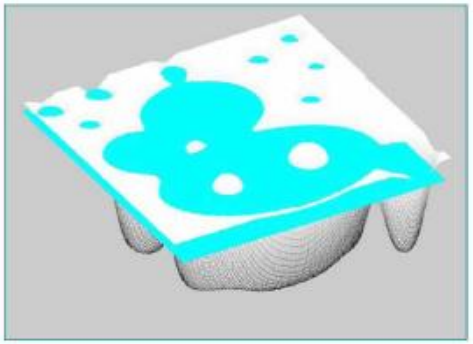
MSER Construction



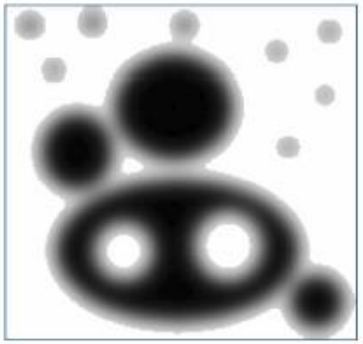
intensity image

shown as a surface function

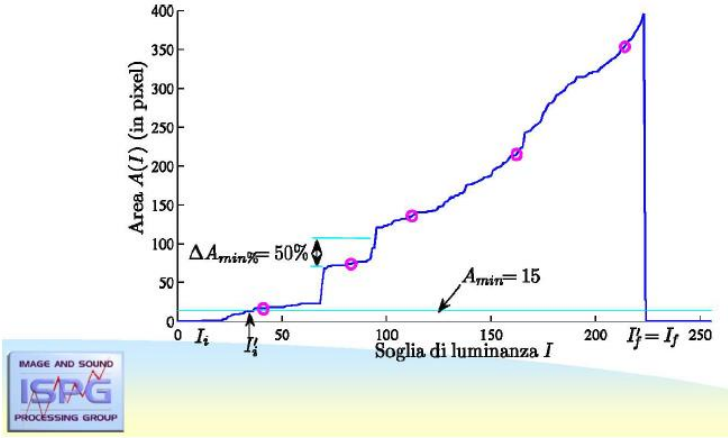
- For each threshold, compute the connected binary regions.
- Compute a function, such as area A(i), at each threshold value i.
- Analyze this function for each potential region to determine those that persist with similar function value over multiple thresholds.



Threshold simulation



Extremal Regions (represented by their original lumiance values)



Properties

Affine invariant

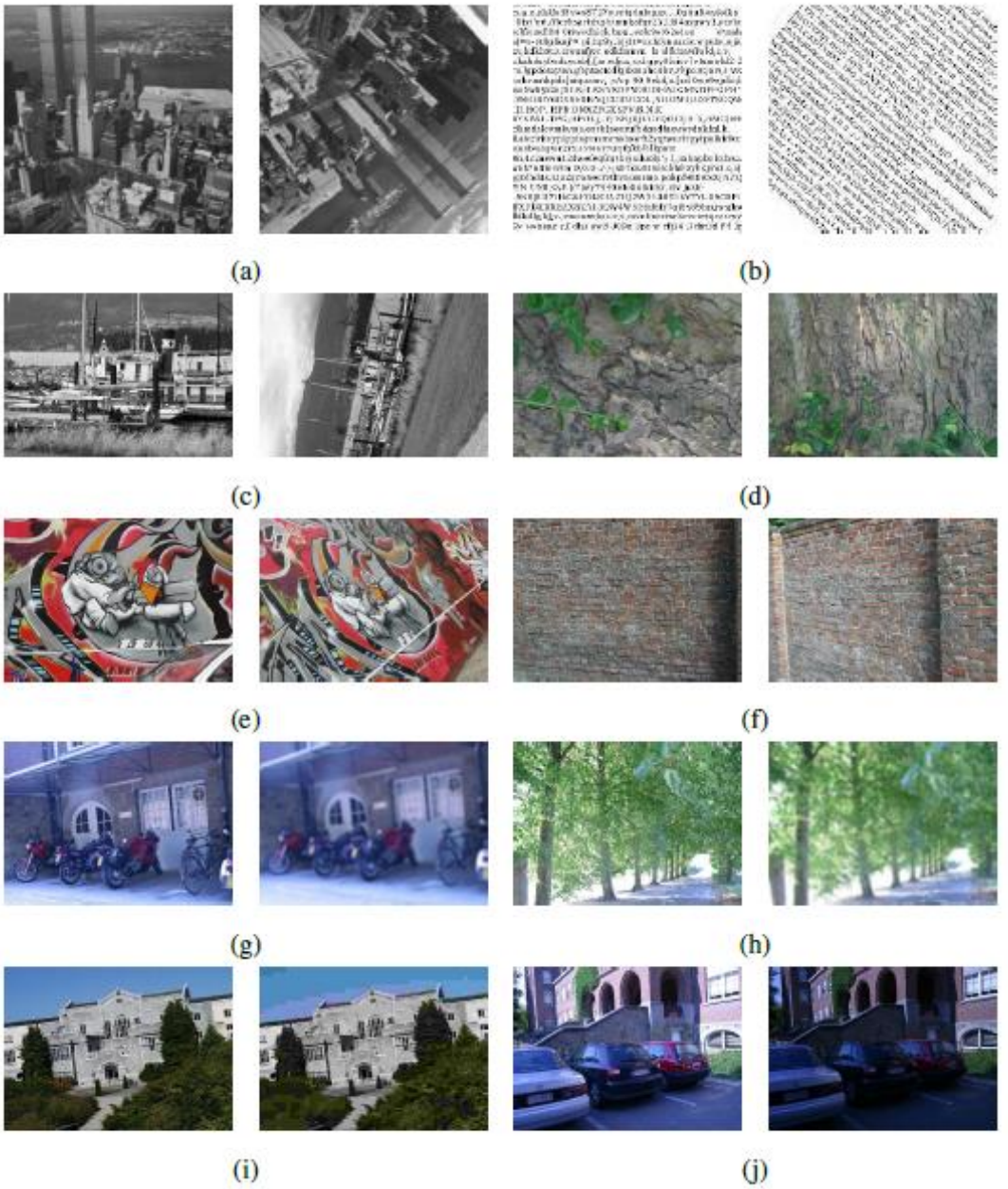
- Simple, efficient scheme
- High repeatability
- Fires on similar features as IBR

(regions need not be convex, but need to be closed)

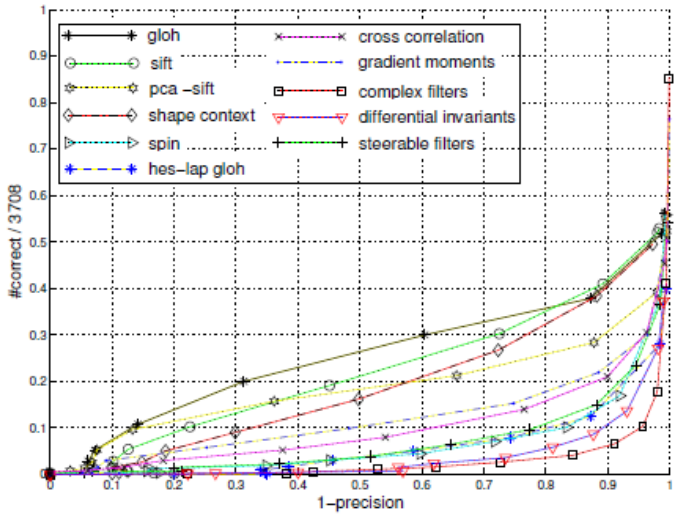
- Sensitive to image blur

Comparison of different mehtods

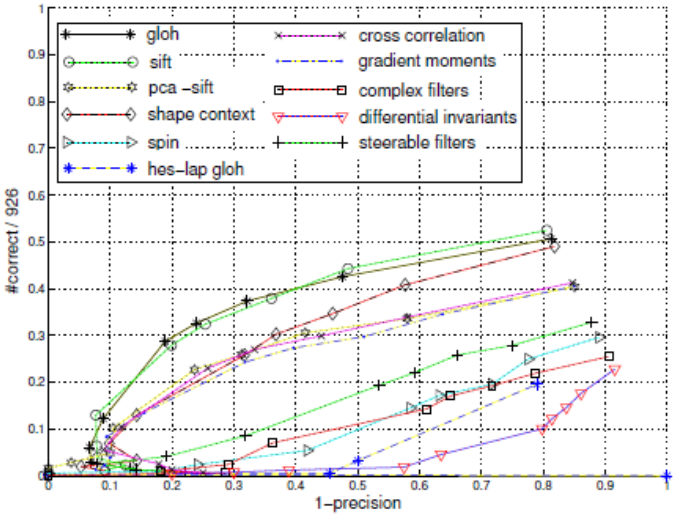
- Rotation (a,b)
- Zoom (c,d)
- Viewpoint change (e,f)
- Blur (g,h)
- JPEG compression (i)
- Light change (j)



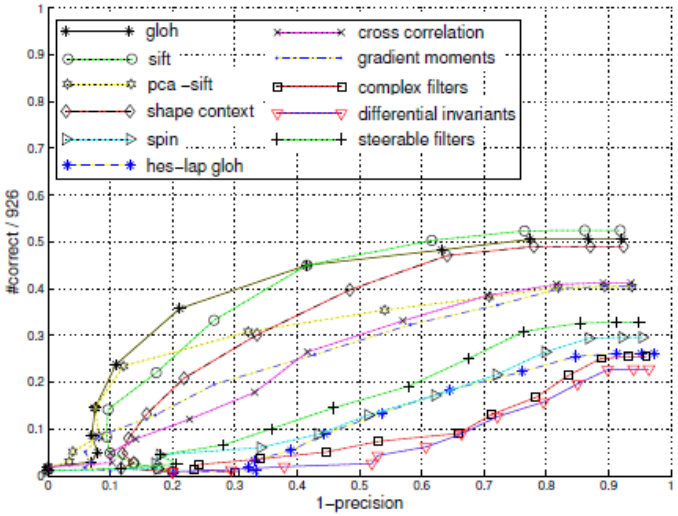
Performance Comparison (precision-recall)



(a)



(b)



(c)

Fig. 4. Comparison of different matching strategies. Descriptors computed on Hessian-Affine regions for images from figure 3(e). (a) Threshold based matching. (b) Nearest neighbor matching. (c) Nearest neighbor distance ratio matching. `hes-lap gloh` is the GLOH descriptor computed for Hessian-Laplace regions (cf. section IV-A.4).

Performances of the various descriptors

Descriptor	$\sum_{i=1:10} eigenvalue(i)$	$\sum_i eigenvalue(i)$
PCA-SIFT	1.0839e+12	1.9743e+12
GLOH	1.4085e+11	2.8277e+11
SIFT	3.4210e+09	6.4541e+09
Shape context	3.3582e+09	7.1149e+09
Spin images	4.4791e+09	5.2355e+09
Cross correlation	1.0657e+09	1.4076e+09
Steerable filters	4.1529e+07	4.2909e+07
Differential invariants	2.5970e+07	2.6349e+07
Complex filters	1.6328e+07	1.8264e+07
Moments	1.3829e+07	1.8100e+07

TABLE I

DISTINCTIVENESS OF THE DESCRIPTORS. SUM OF THE FIRST 10 AND SUM OF ALL EIGENVALUES FOR DIFFERENT DESCRIPTORS.

Descriptor	recall	1-precision	#nearest neighbor correct matches
GLOH	0.25	0.52	192
SIFT	0.24	0.56	177
Shape context	0.22	0.59	166
PCA-SIFT	0.19	0.65	139
Moments	0.18	0.67	133
Cross correlation	0.15	0.72	113
Steerable filters	0.12	0.78	90
Spin images	0.09	0.84	64
Differential invariants	0.07	0.87	54
Complex filters	0.06	0.89	44

TABLE II

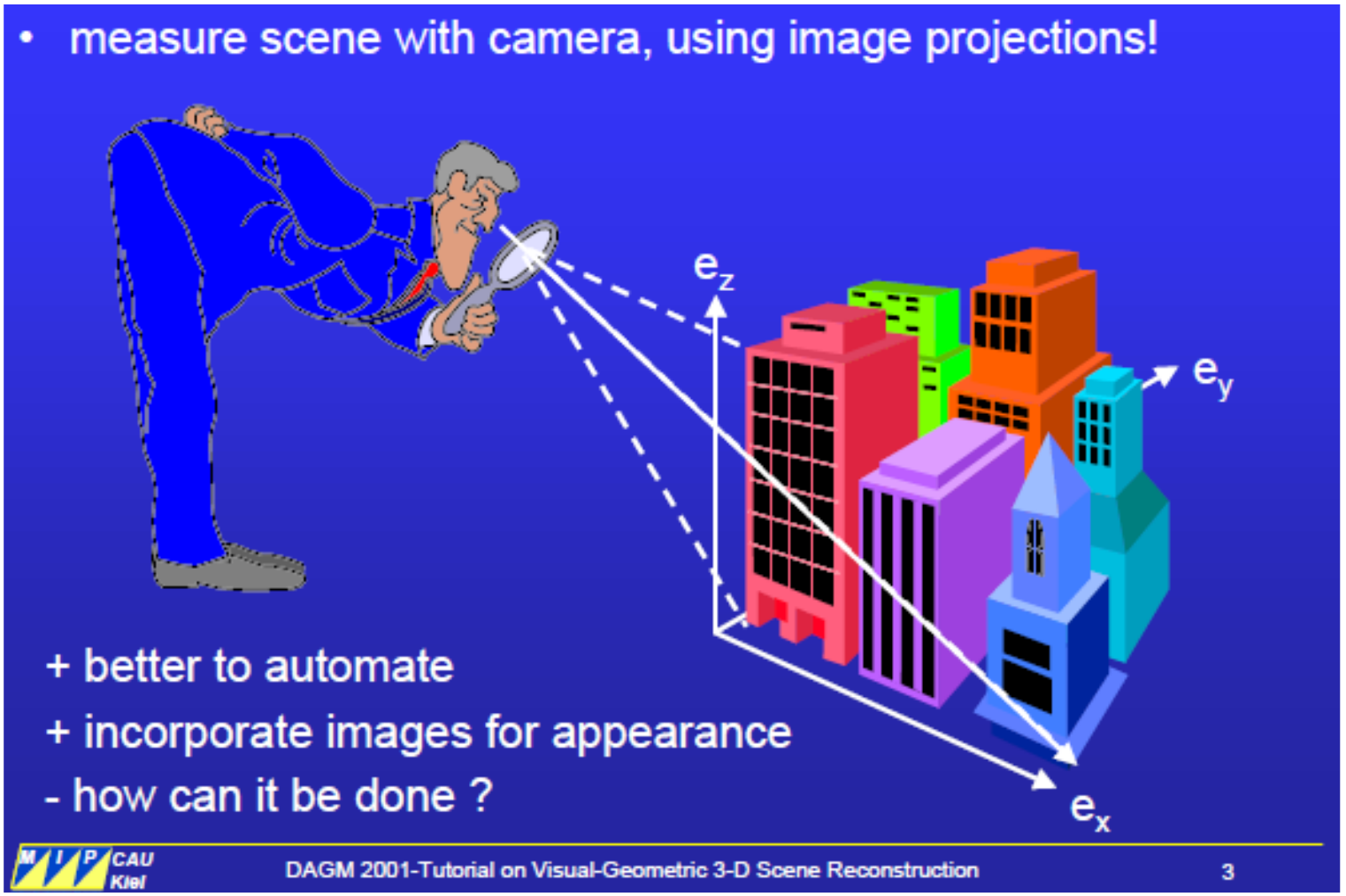
RECALL, 1-PRECISION AND NUMBER OF CORRECT MATCHES OBTAINED WITH DIFFERENT DESCRIPTORS FOR A FIXED NUMBER OF 400 NEAREST NEIGHBOR MATCHES ON THE IMAGE PAIR DISPLAYED IN FIGURE 13. THE REGIONS ARE DETECTED WITH HESSIAN-AFFINE.

K. Mikolajczyk and C. Schmid,"A Performance Evaluation of Local Descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pp. 1615-1630, 2005.

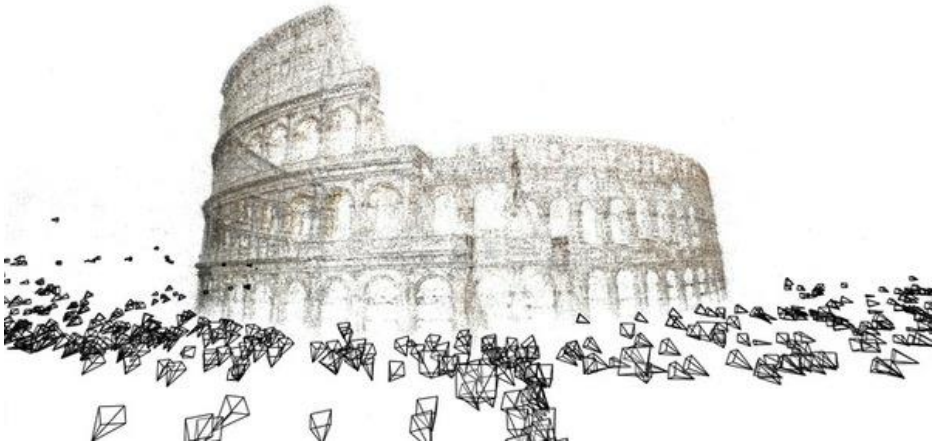
Part 5
Application: 3D Reconstruction

Some slides from Koch and Frahm, DAGM 2001 tutorial

3D Reconstruction of a scene from a set of picutres

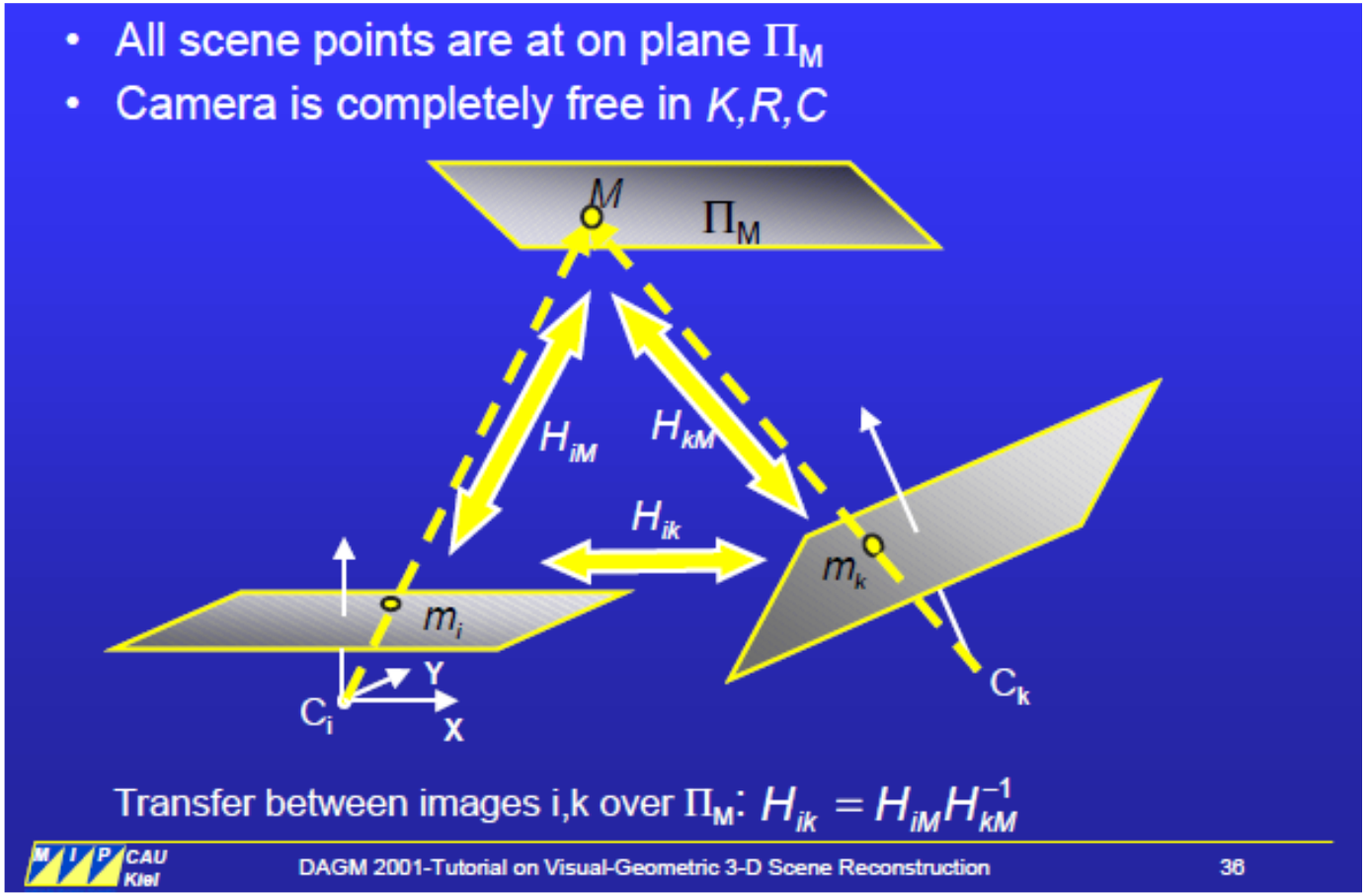


3D Reconstruction



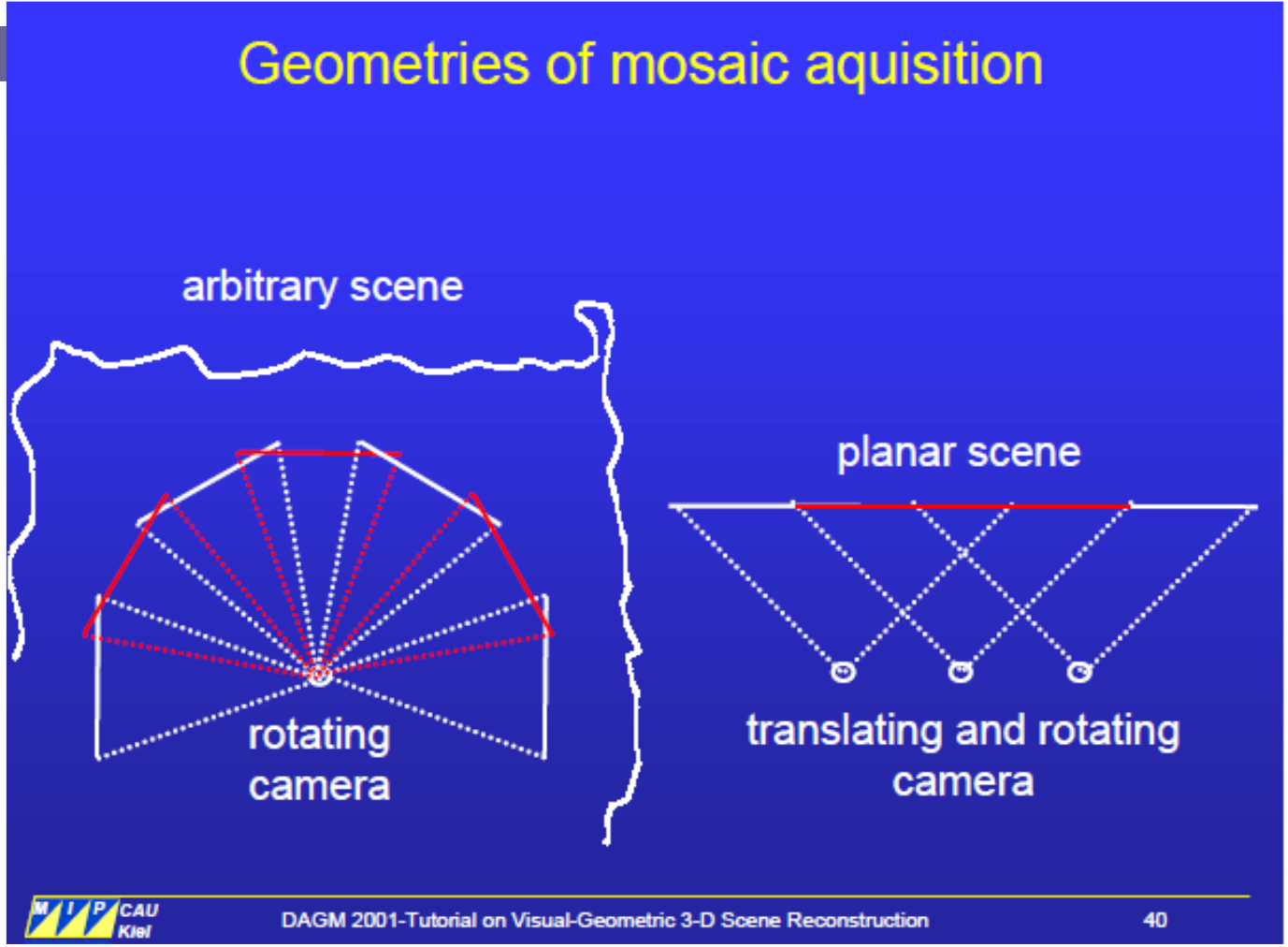
- Computer vision enables us to reconstruct highly naturalistic computer models of 3D environments from camera images
- We need to extract
  - The camera geometry (calibration),
  - The scene structure (surface geometry)
  - The visual appearance (color and texture) of the scene

2D to 3D



- A 3D point is projected onto a set of 2D images
- Recognize the same location on different images (feature extraction and description!)
- The intersection of the optical rays is the 3D location of the point !
- Need also camera calibration parameters

Simplest case: image mosaicing (planar scene)



Multiple view geometry

Projection onto two views:

$$P_0 = K_0 R_0^{-1} [I \quad 0]$$

$$\rho_0 m_0 = P_0 M = K_0 R_0^{-1} [I \quad 0] M$$

$$\Rightarrow \rho_0 m_0 = K_0 R_0^{-1} [I \quad 0] M_\infty$$

$$P_1 = K_1 R_1^{-1} [I \quad -C_1]$$

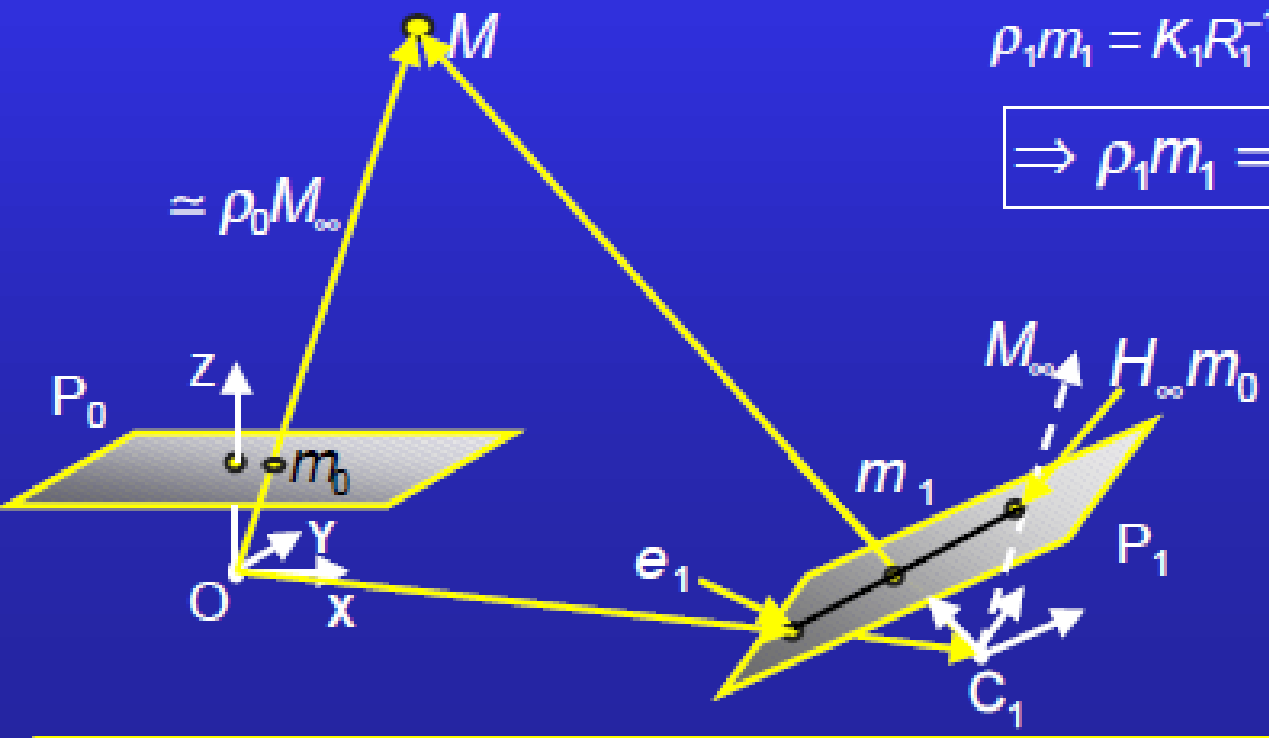
$$\rho_1 m_1 = P_1 M = K_1 R_1^{-1} [I \quad -C_1] M$$

$$= K_1 R_1^{-1} [I \quad 0] M_\infty + K_1 R_1^{-1} [I \quad -C_1] O$$

$$\rho_1 m_1 = K_1 R_1^{-1} R_0 K_0^{-1} \rho_0 m_0 - K_1 R_1^{-1} C_1$$

$$\Rightarrow \rho_1 m_1 = \rho_0 H_\infty m_0 + e_1$$

Epipolar line

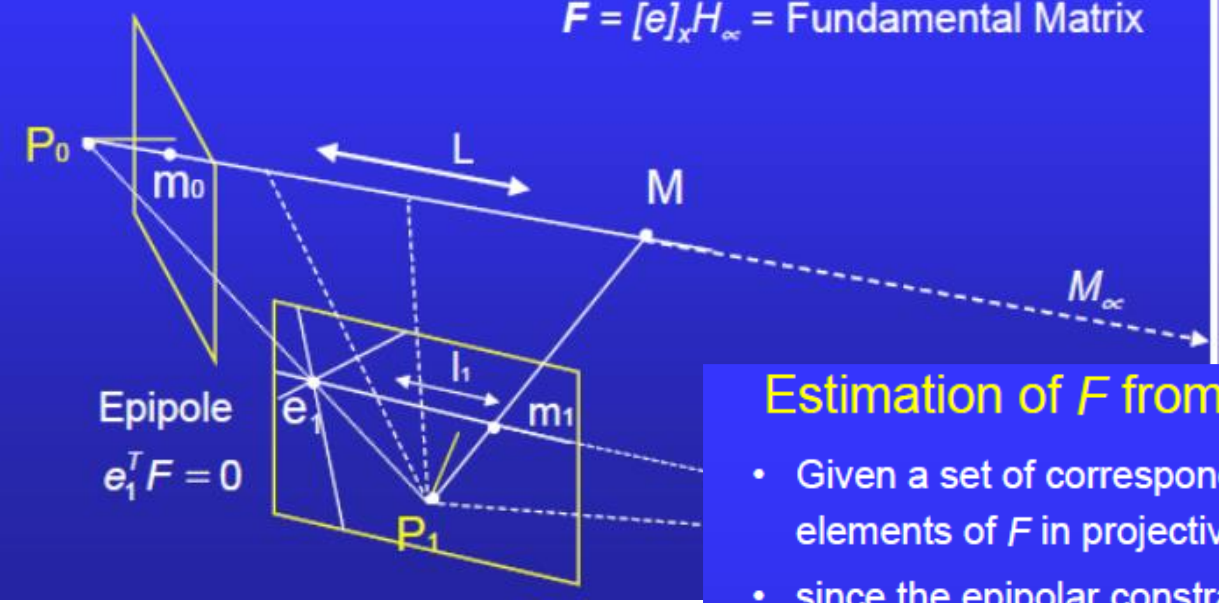


$$M = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = M_\infty + O$$

The Fundamental Matrix F

$m_1^T l_1 = 0 \qquad l_1 = F m_0 \qquad m_1^T F m_0 = 0$

$\boldsymbol{F} = [e]_x H_\infty$ = Fundamental Matrix



Estimation of *F* from image correspondences

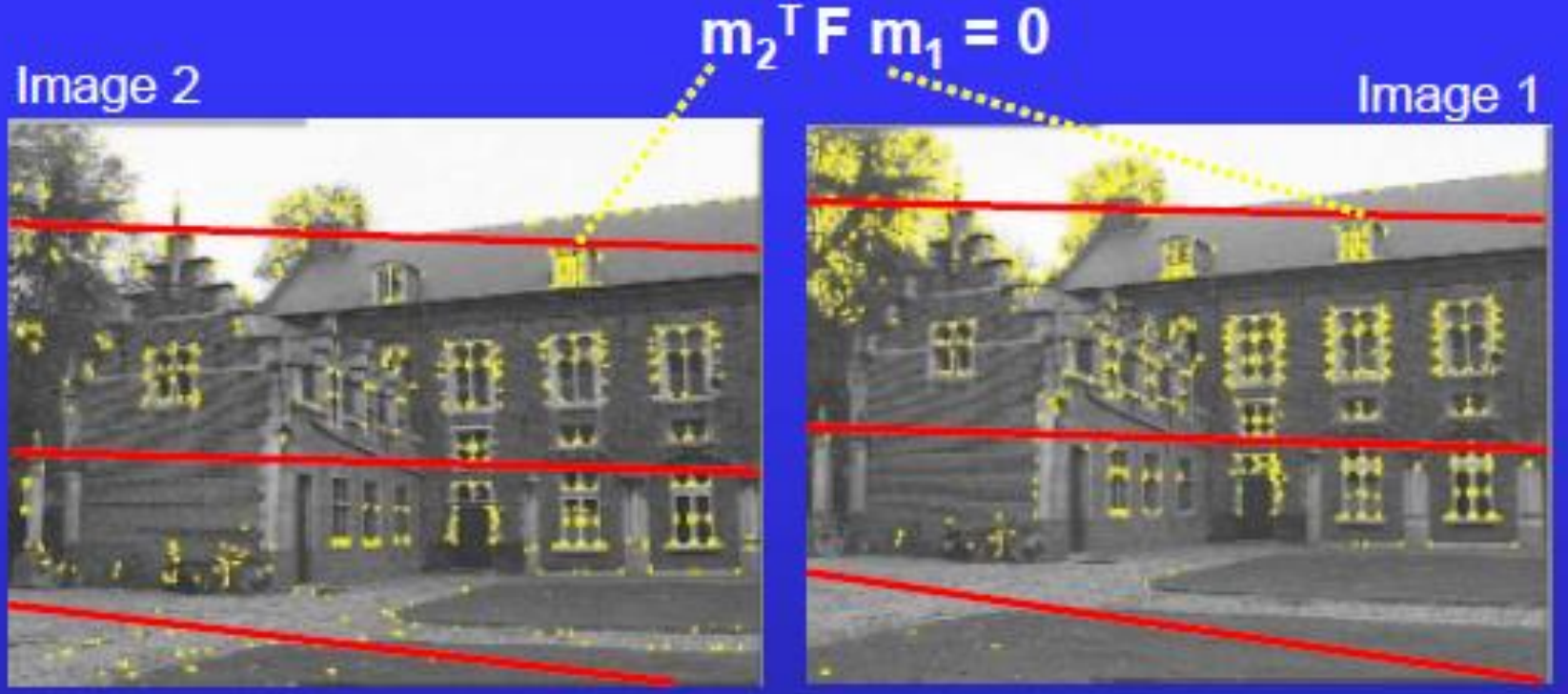
- Given a set of corresponding points, solve linearily for the 9 elements of *F* in projective coordinates
- since the epipolar constraint is homogeneous up to scale, only eight elements are independent
- since the operator $[e]_x$ and hence *F* have rank 2, *F* has only 7 independent parameters (all epipolar lines intersect at e)
- each correspondence gives 1 collinearity constraint

=> solve F with minimum of 7 correspondences

for N>7 correspondences minimize distance point-line:

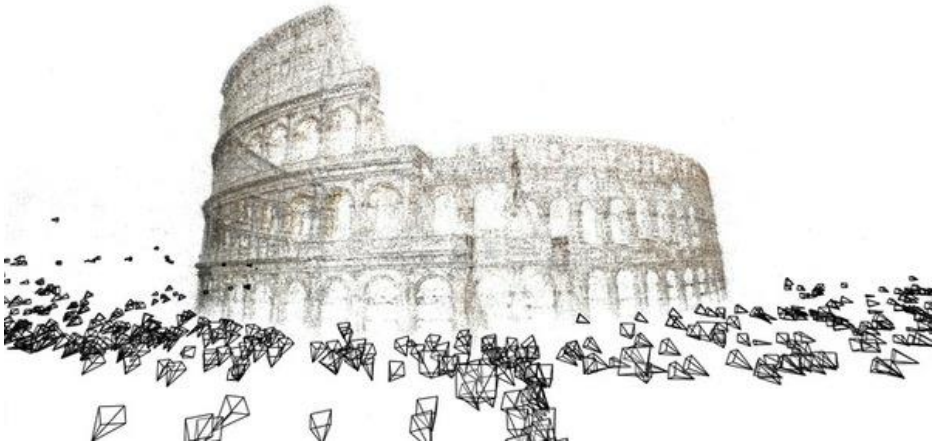
$$m_{1i}^T F m_{0i} = 0 \qquad \sum_{n=0}^{N} (m_{1,n}^T F m_{0,n})^2 \Rightarrow \min!$$

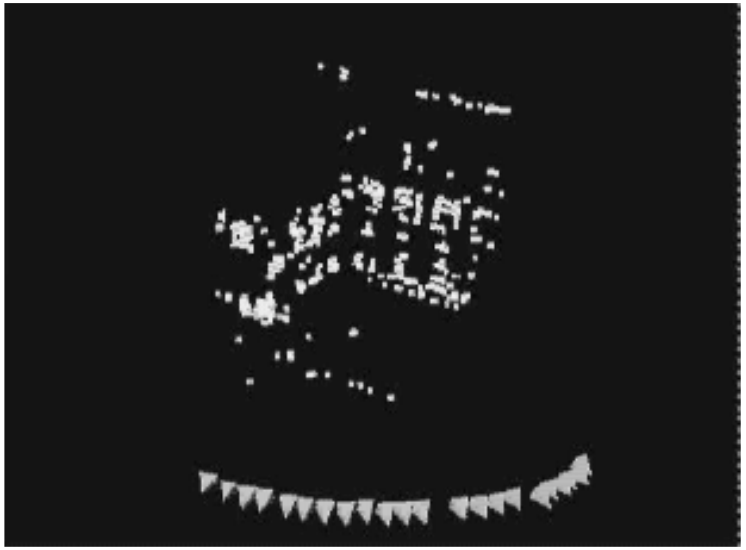
Estimation of Fundamental Matrix



Robust correspondence selection m_1 <-> m_2

Output of the approach



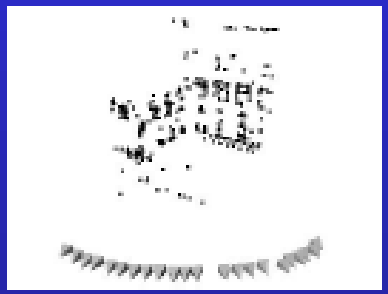


- Sparse 3D point cloud (i.e., 3D location of feature points)
- Camera locations and parameters

3D Reconstruction Pipeline

- camera calibration from feature tracking
- dense depth estimation from stereo correspondence
- depth fusion and generation of textured 3D surface model

Image sequence



camera calibration

scene geometry

3D surface model

Structure-from-Motion from unordered image collections

[Brown05, Snavely06, Agarwal09]

- Image clustering
- Pose initialization
- Bundle-adjustment

http://phototour.cs.washington.edu/bundler