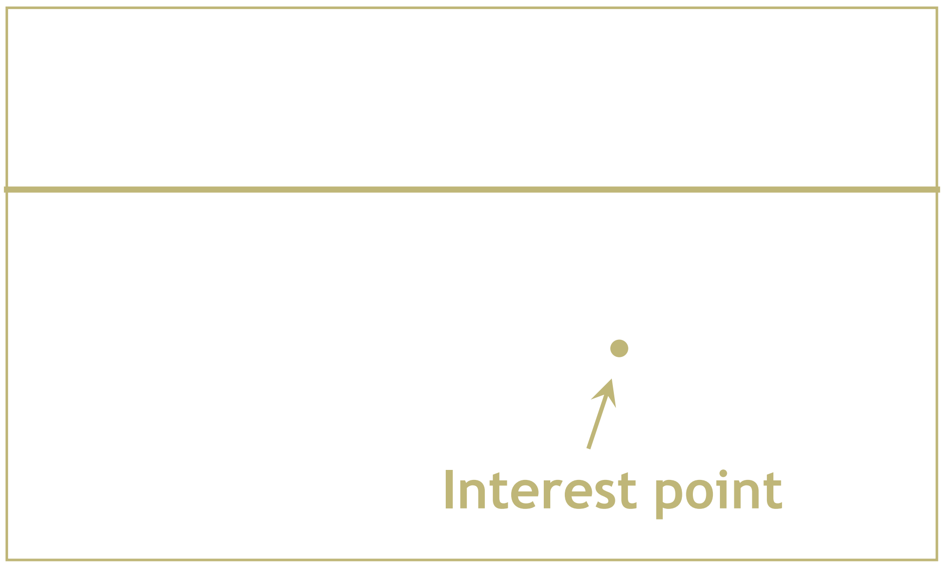
Today- Image Features

-
- Local Features
- Methods for feature extraction

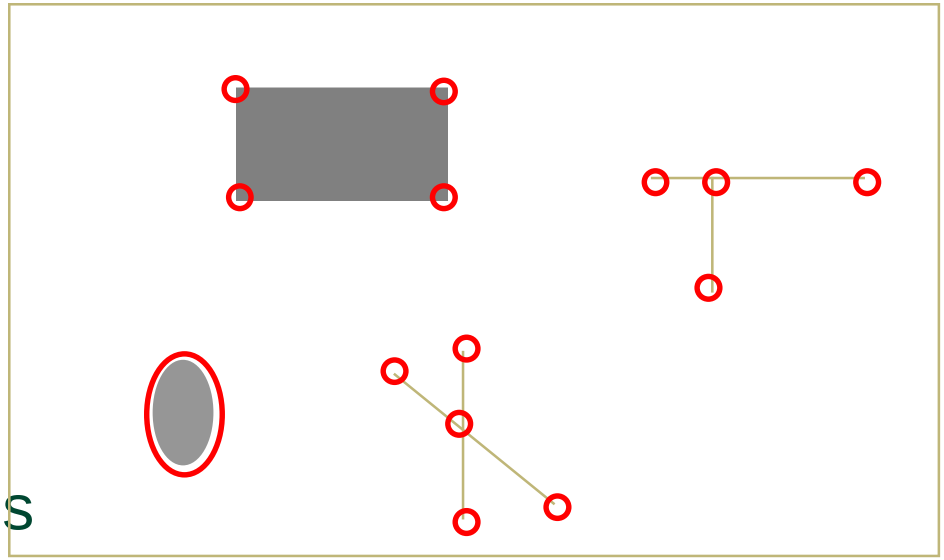
Local Features

- Edges
  - Local in one dimension



Local Features

- Edges
- Corners
- Junctions
  - L
  - X
  - T
  - ...
- Edge ends
- Blobs



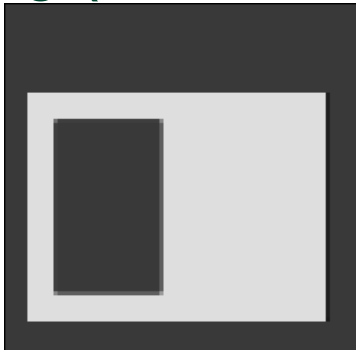
Local Features

Local Features

- Edge detection
  - One dimensional signal change

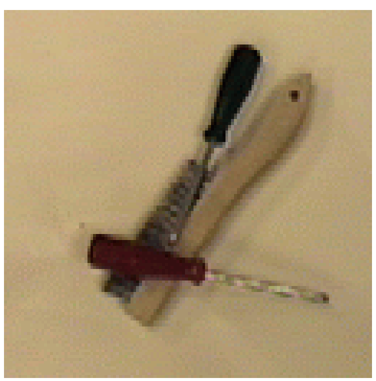
Edge Detection

- Principle:
  - find transitions of regions by extracting the edges of regions
  - assumption: regions are (nearly) homogenous
  - physical definition: edges correspond to discontinuities between ‘homogeneous’ regions
  - we have to take into account the noise
  - edge detection using (1st or 2nd) derivatives



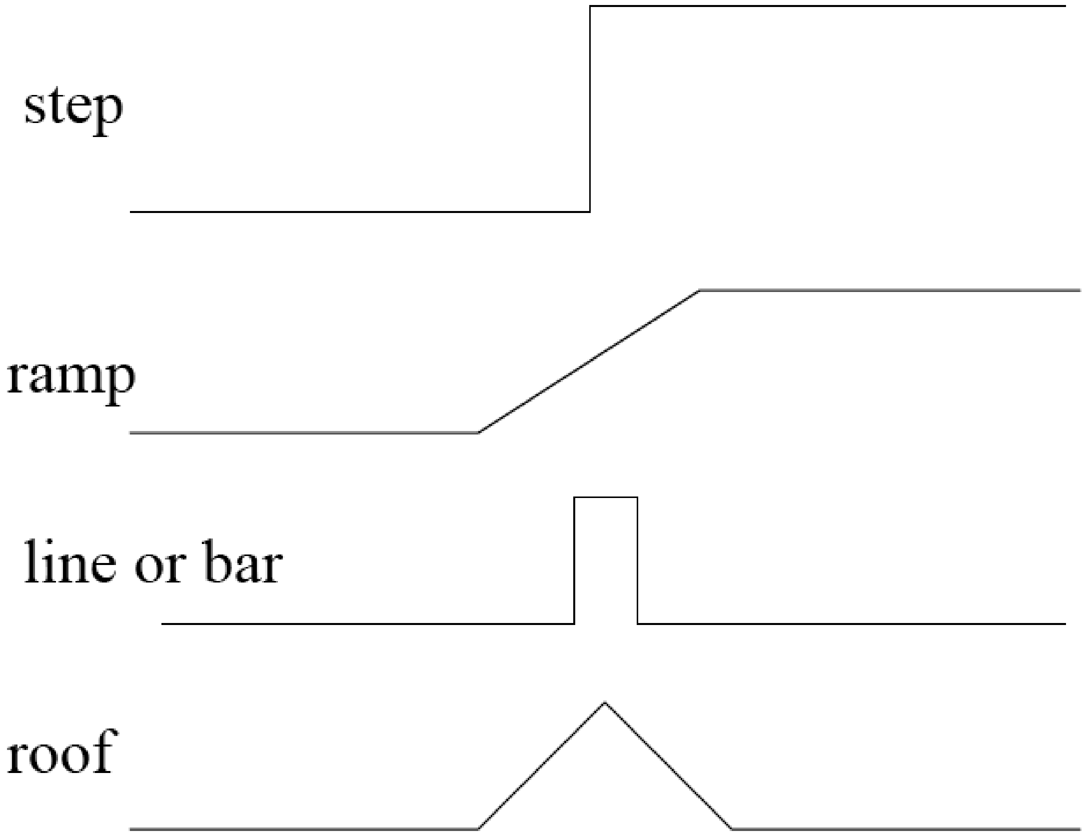
Edge Detection

- What are edges ?
  - Humans find object boundaries very quickly - which is why we would like to define ‘edges’ to be the object boundaries
  - to detect object boundaries we have to know which objects are in the scene (I.e. we have to recognize the object(s) first…):




What are ‘edges’ (1D)

- Idealized:



Edge Detection

- What are edges ?

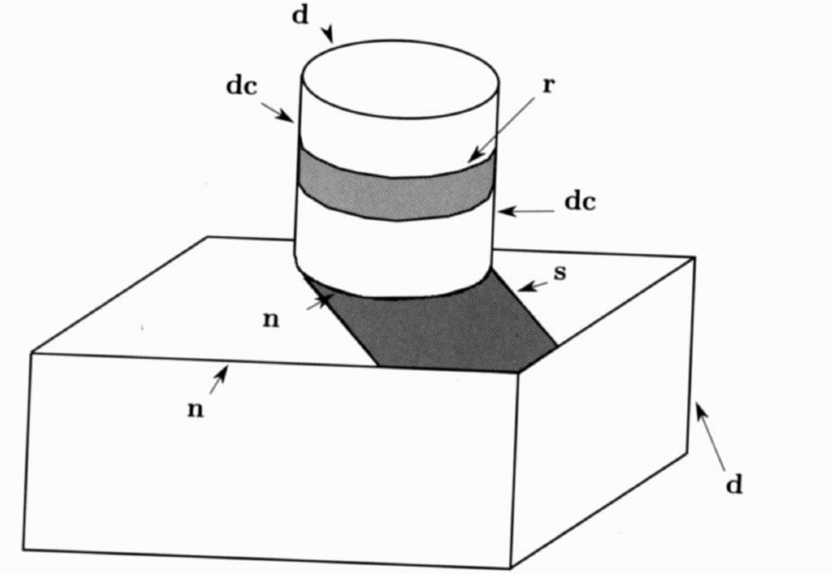
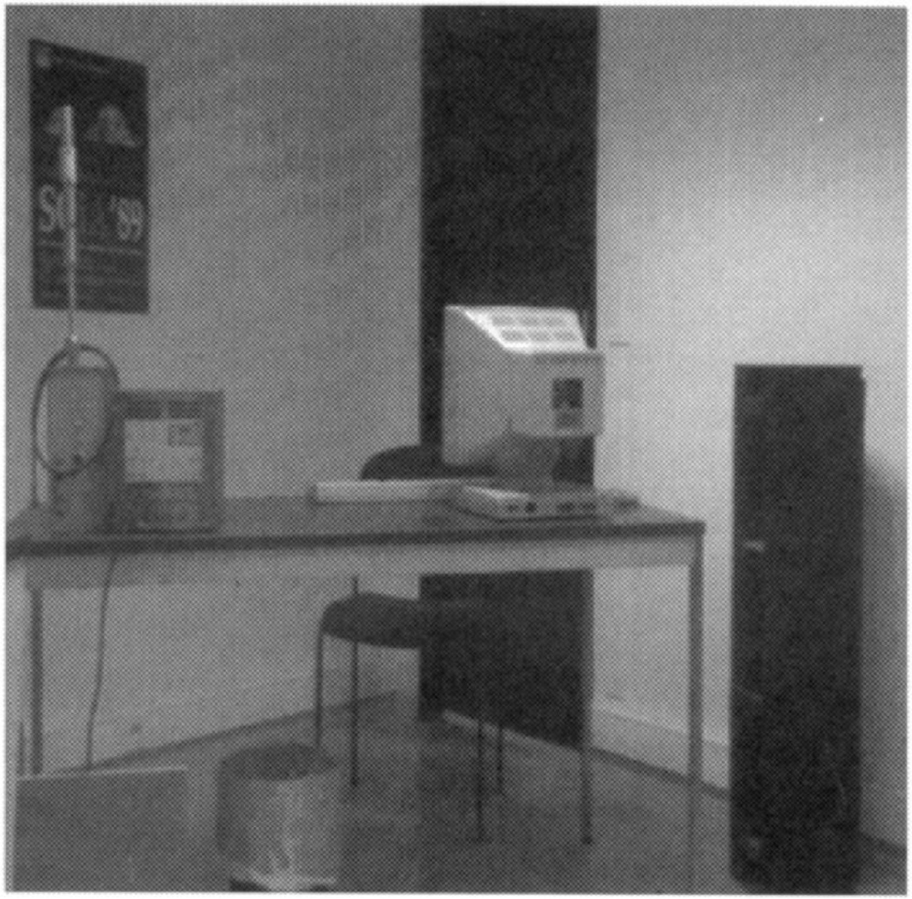


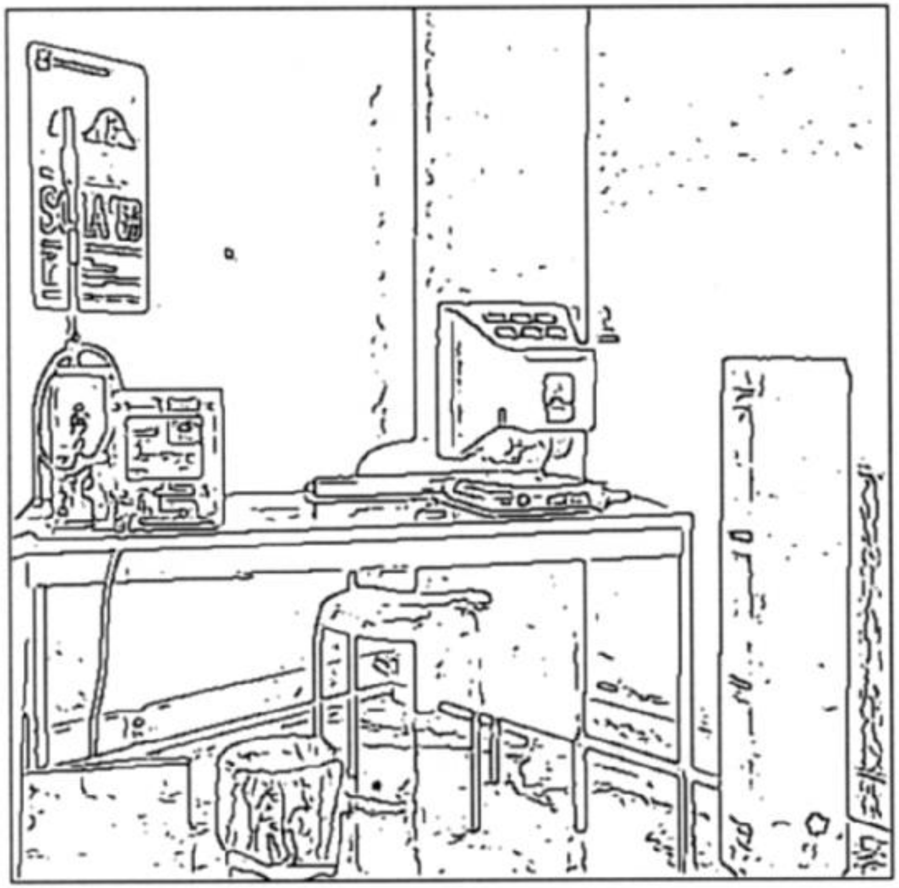
Figure 4.1 Edges in an image have different physical sources.

  - object-background boundaries (edges (d) and depth discontinuity (dc))
  - object-object boundaries (those correspond not necessarily to object boundaries, (n))
  - shadows (s)
  - discontinuities of object texture (r)
  - discontinuities of surface normals (n)

Example

- Image with multiple objects:

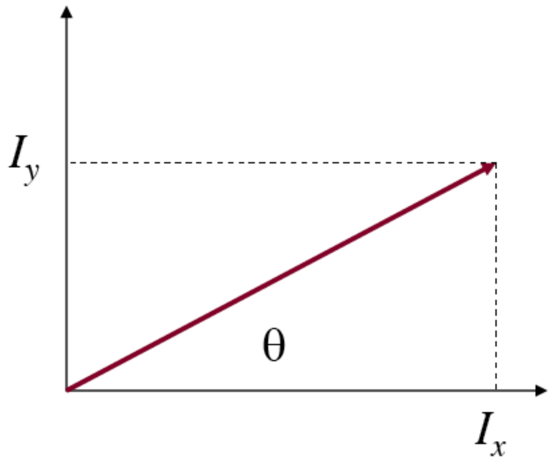




2D Edge Detection

- calculate derivative
  - use the magnitude of the gradient
  - the gradient is:

$$\nabla I = (I_x, I_y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$$



  - the magnitude of the gradient is:

$$\|\nabla I\| = \sqrt{{I_x}^2 + {I_y}^2}$$

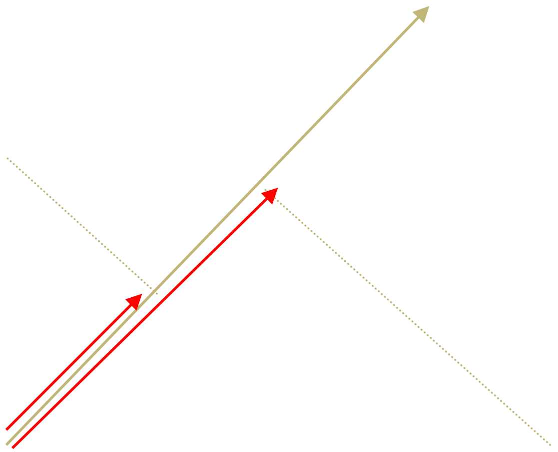
  - the direction of the gradient is:

$$\theta = \arctan(I_y, I_x)$$

'Steerable' Filters

- what is the gradient in some direction θ?

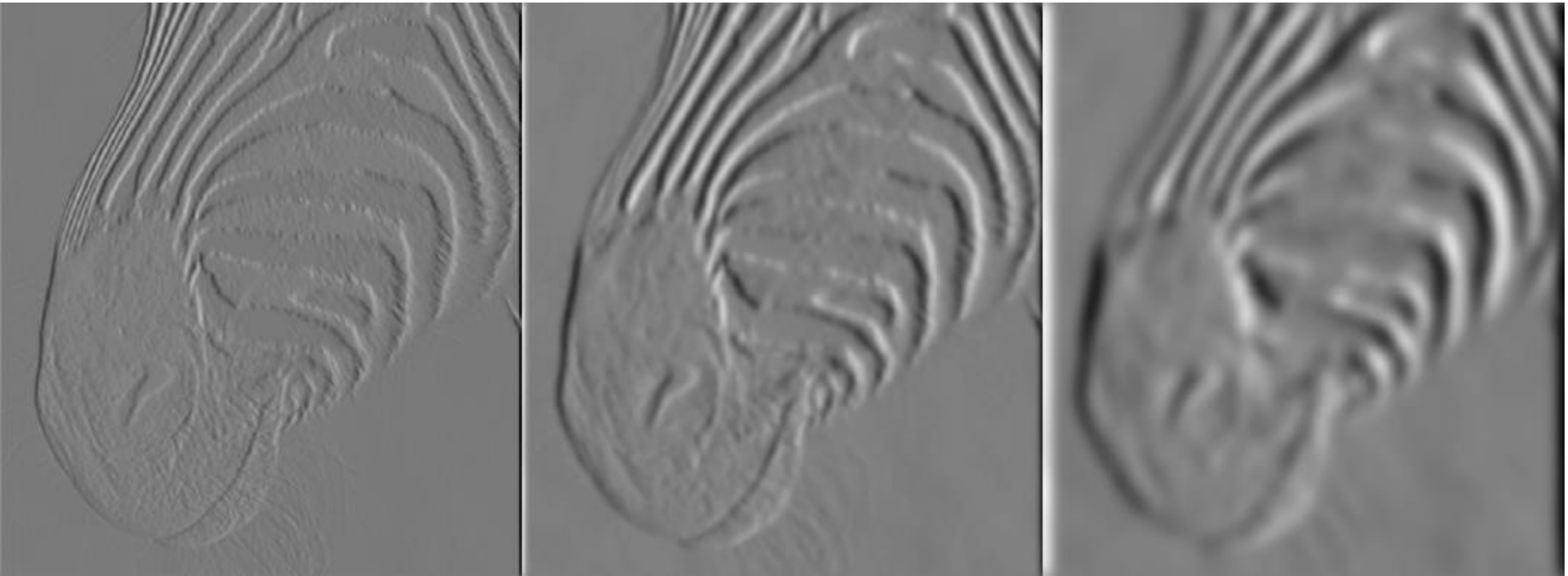
$$f'_\theta(x,y) = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$



2D Edge Detection

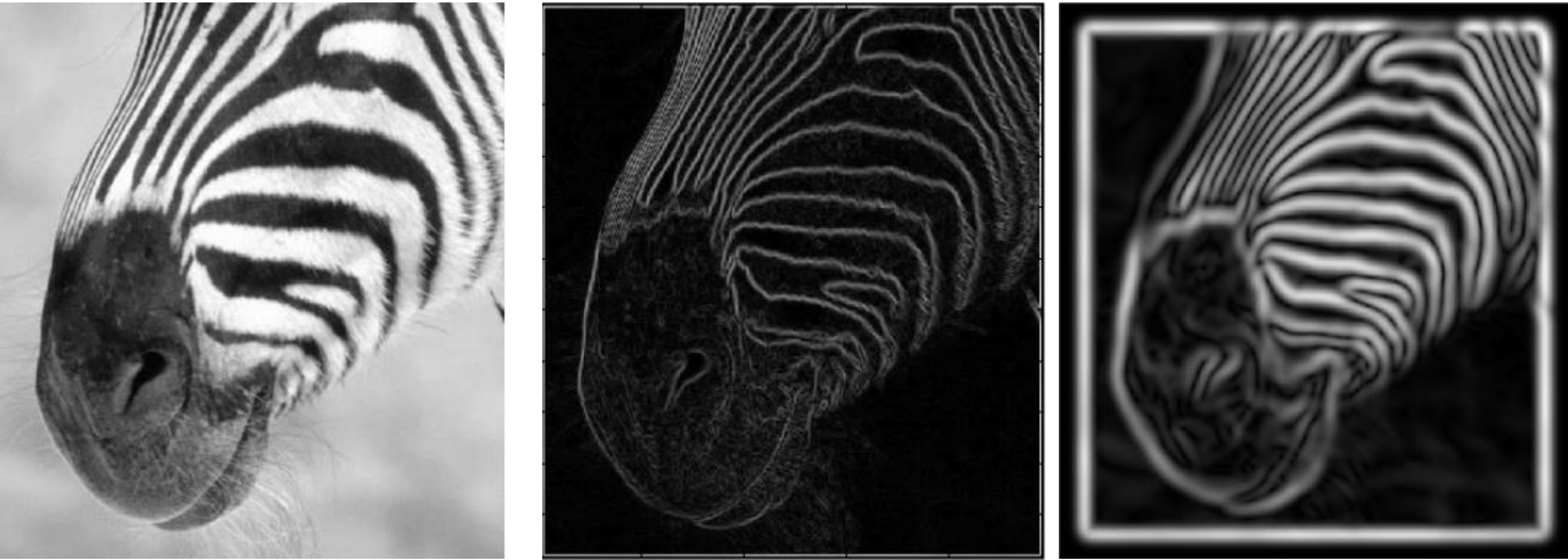
- the scale of the smoothing filter affects derivative estimates, and also the semantics of the edges recovered
  - note: strong edges persist across scales

1 pixel 3 pixels 7 pixels



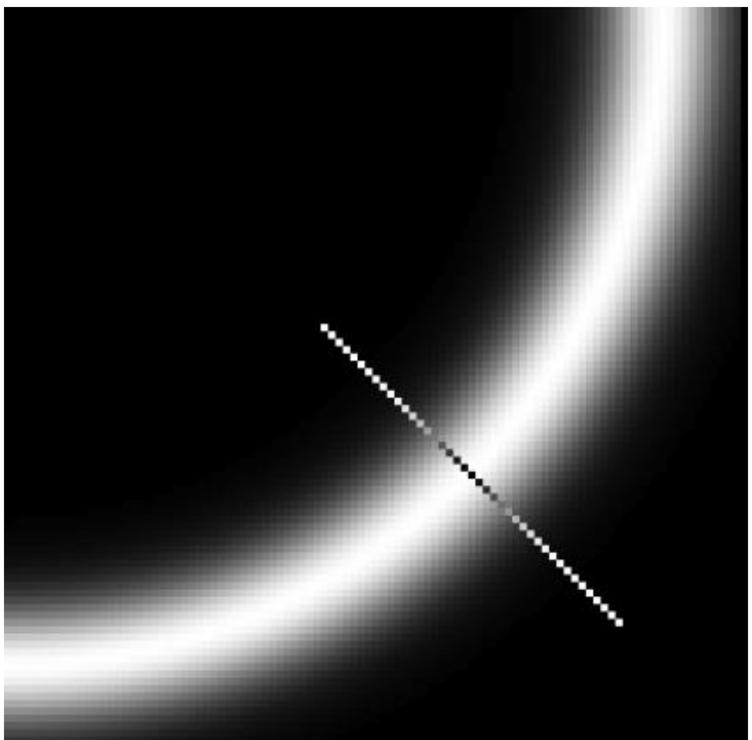
2D Canny Edge Detection

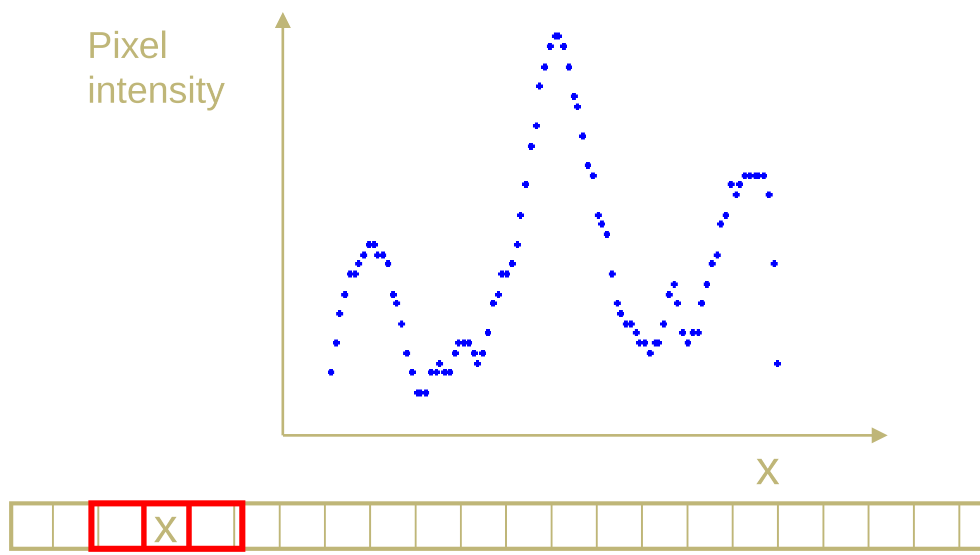
- there are 3 major issues:
  - the gradient magnitude at different scales is different; which should we choose?
  - the gradient magnitude is large along a thick trail; how do we identify the significant points?
  - how do we link the relevant points up into curves?



2D Canny Edge Detection

- Non-Maxima Suppression
- look in a neighborhood along the direction of the gradient
- choose the largest gradient magnitude in this neighborhood

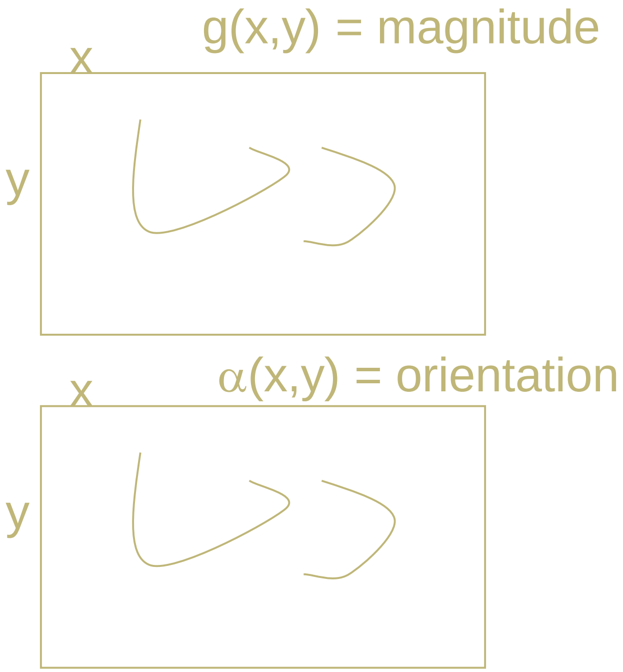


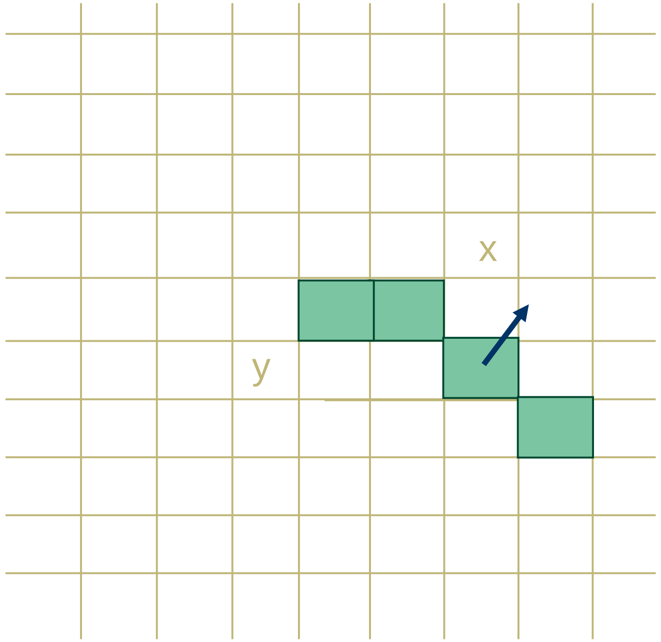


If f(x) > f(x-1) & f(x) > f(x+1) => x = 1
else x = 0

2D Edge Detection

- Non-Maxima Suppression



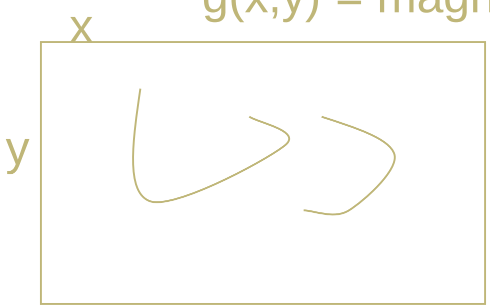


Gradient magnitude g(x,y), orientation α(x,y)

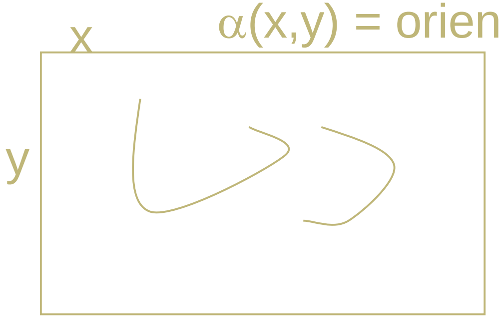
2D Edge Detection

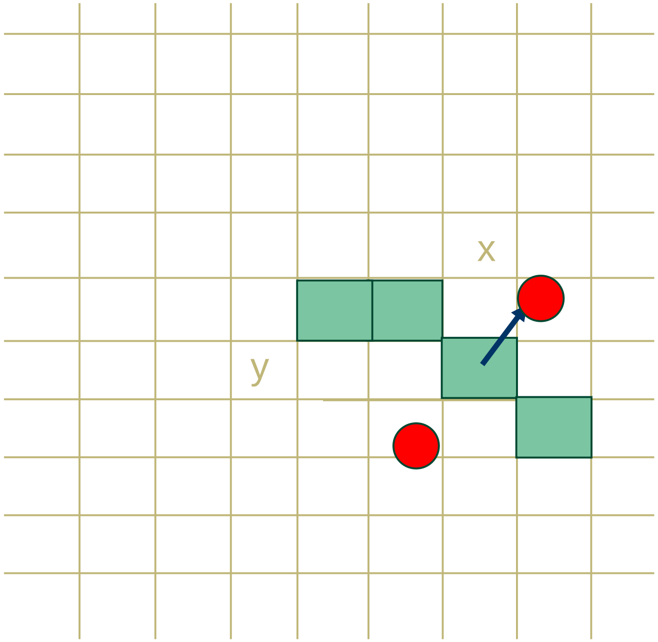
- Non-Maxima Suppression

g(x,y) = magnitude



α(x,y) = orientation





Gradient magnitude g(x,y), orientation α(x,y)

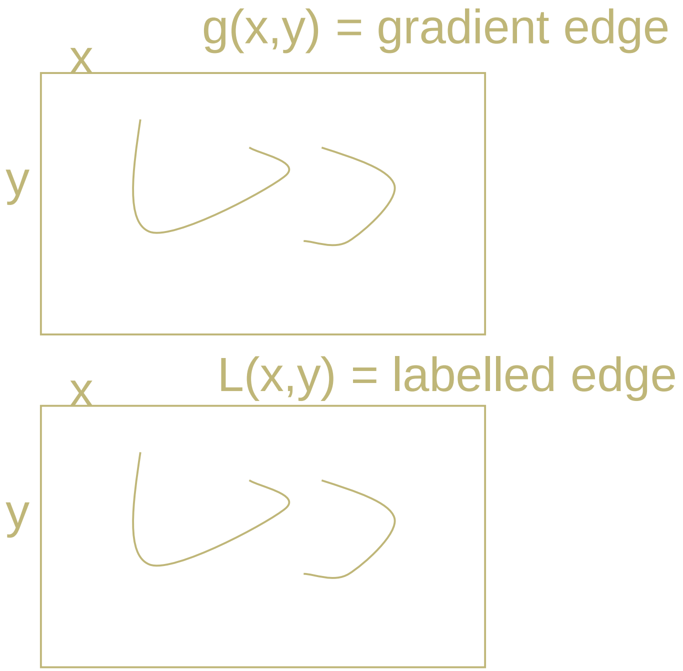
If g(x,y) > g(x+dx,y+dy) && If g(x,y) > g(x-dx,y-dy) <= bilinear interpolation

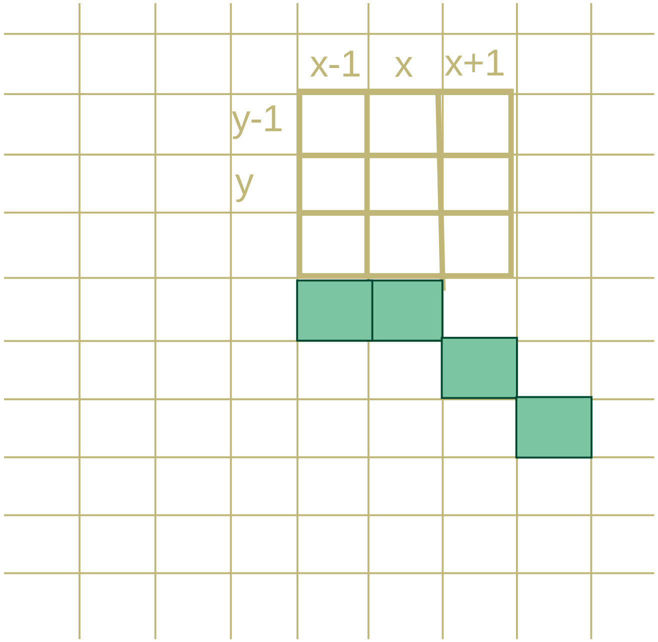
else g(x,y) =0

Where α(x,y) = atan(dy,dx)

2D Edge Detection

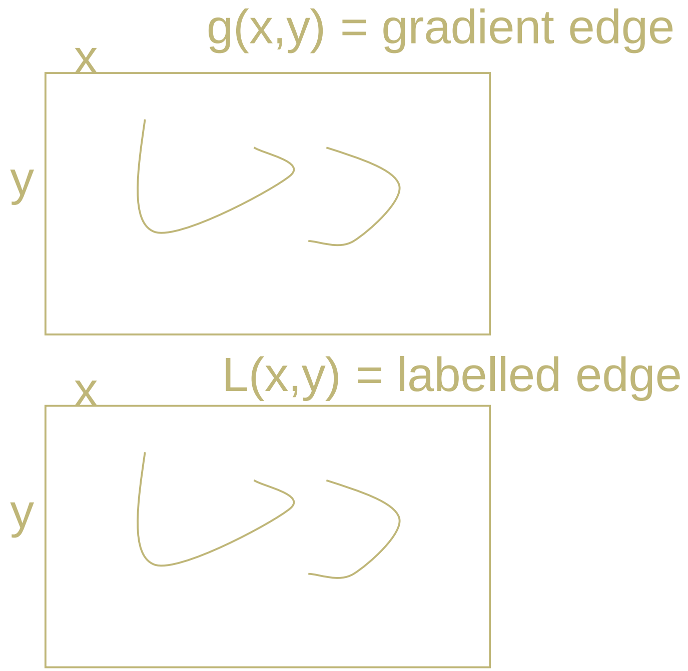
- Labelling connected edges

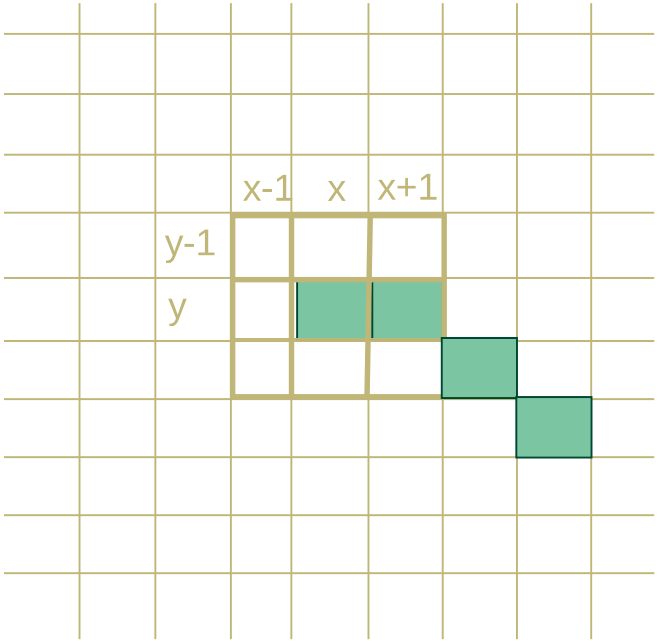




2D Edge Detection

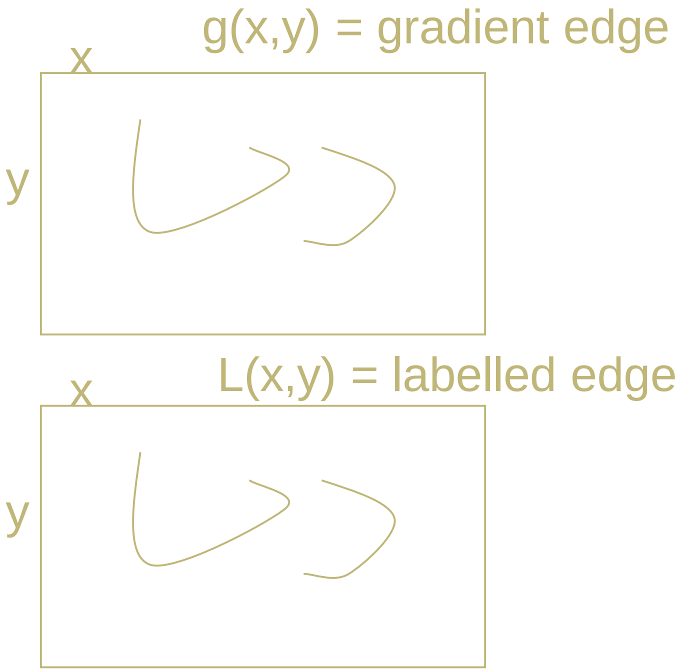
- Labelling connected edges

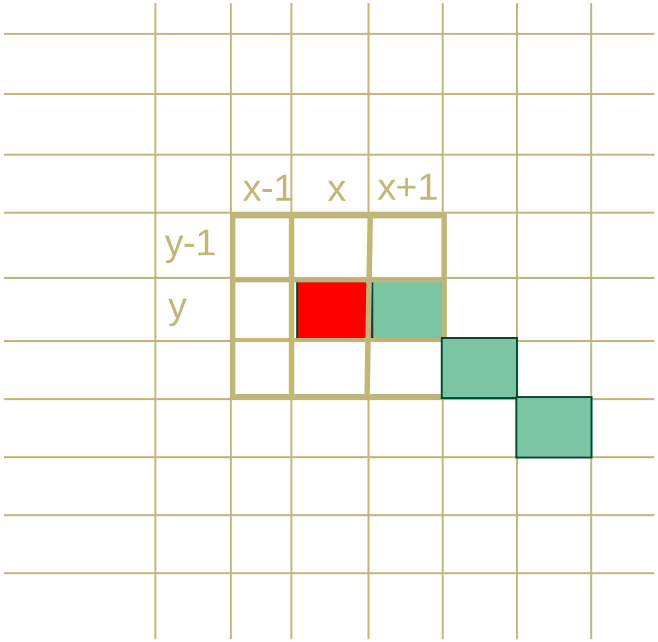




2D Edge Detection

- Labelling connected edges



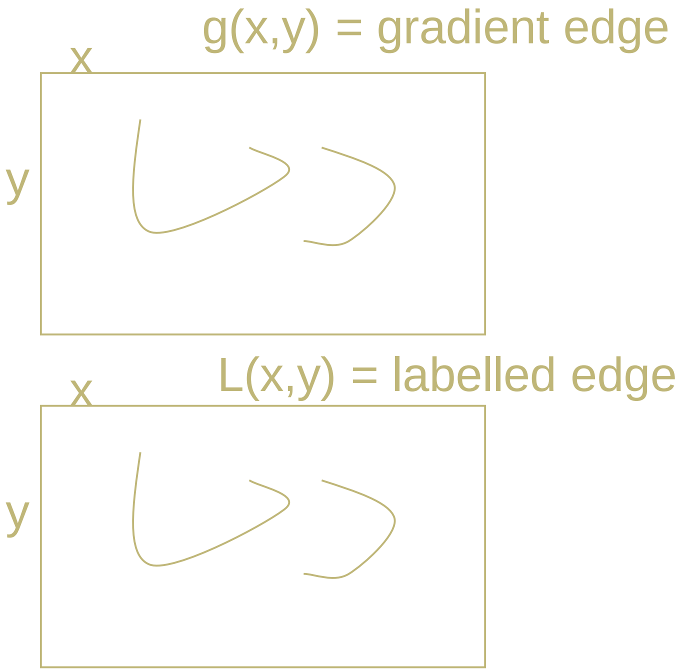


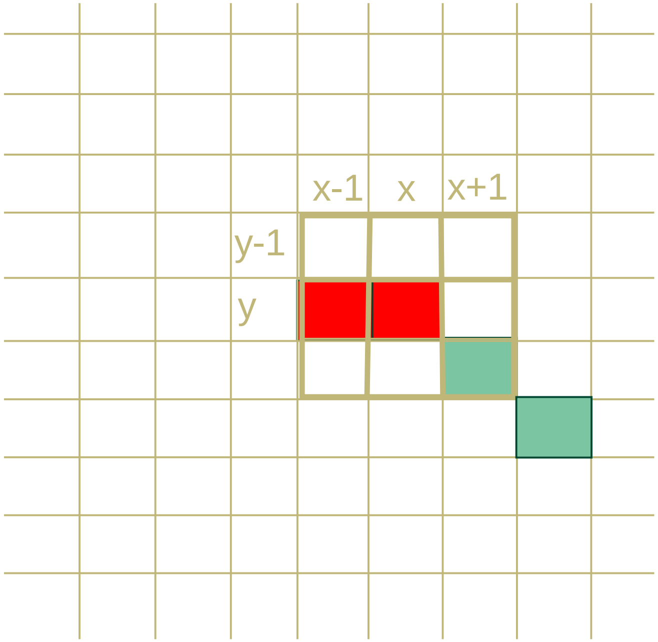
If g(x,y) > TH   L(x,y)={L(x-1,y-1) || L(x,y-1) || L(x+1,y-1) || L(x-1,y) || new_label}

else If g(x,y) > TL   l(x,y)=t_edge

2D Edge Detection

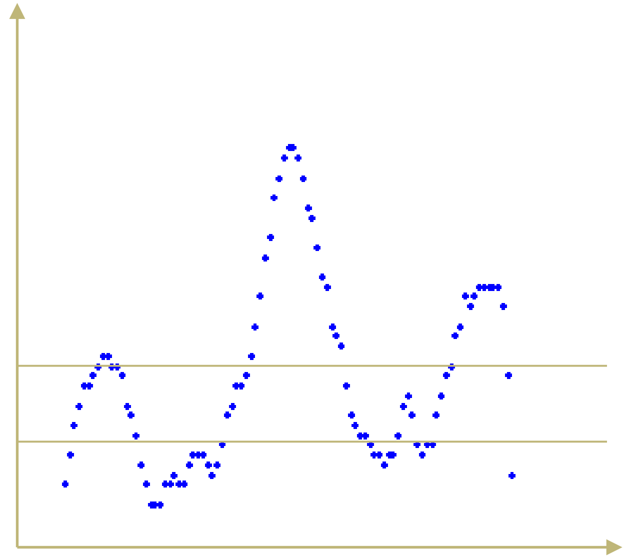
- Labelling connected edges





2D Edge Detection

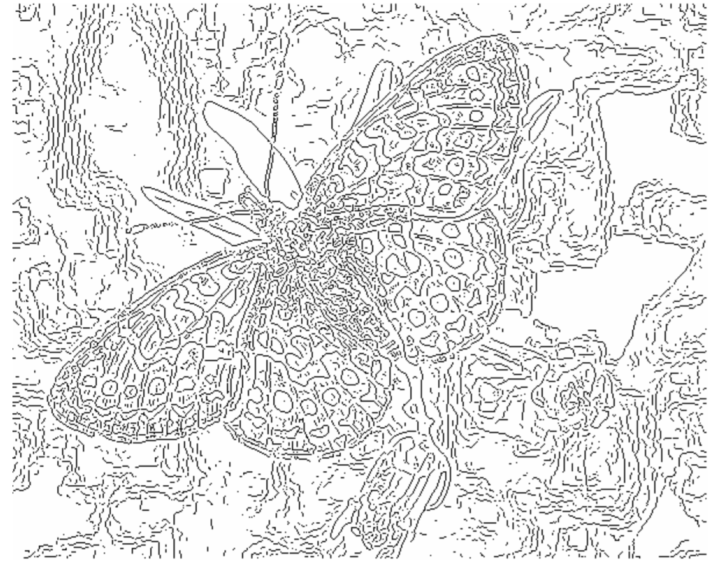
- Hysteresis Thresholding
  High and low Threshold
  - High threshold to validate the edge points
  - Low threshold to remove the noise



Canny Edge detection

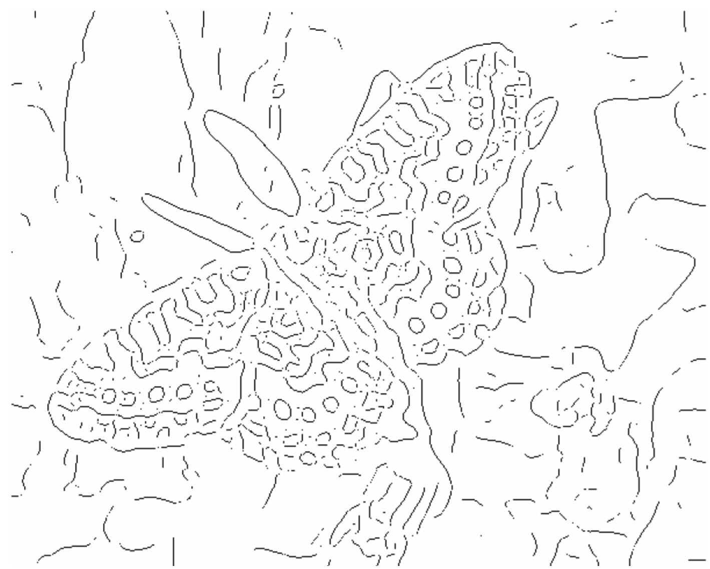
- Computation of image derivatives
- Conversion to gradient magnitude and orientation
- Detection of local maxima in the gradient direction (non maxima supression)
- Labelling connected edges
- Hysteresis thresholding

Butterfly Example



fine scale
high
threshold

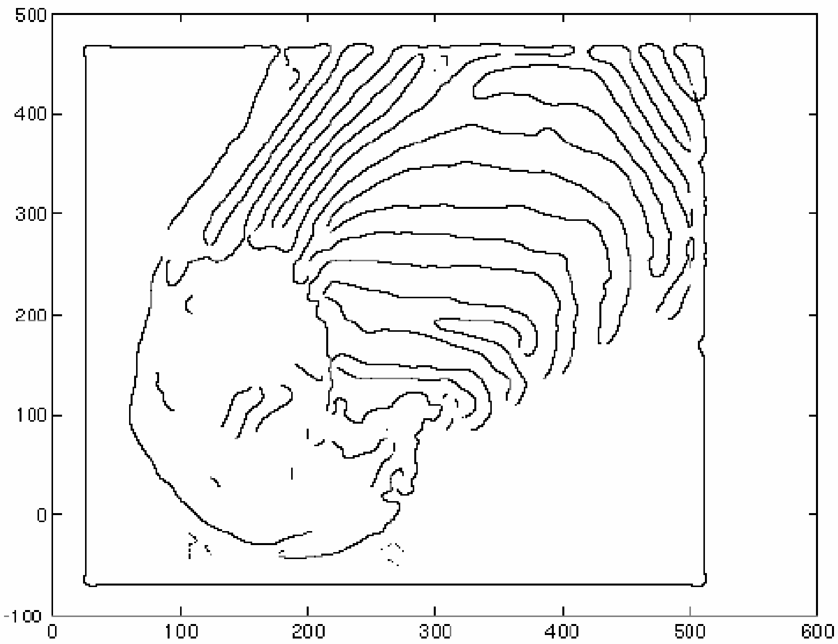
coarse
scale
low
threshold

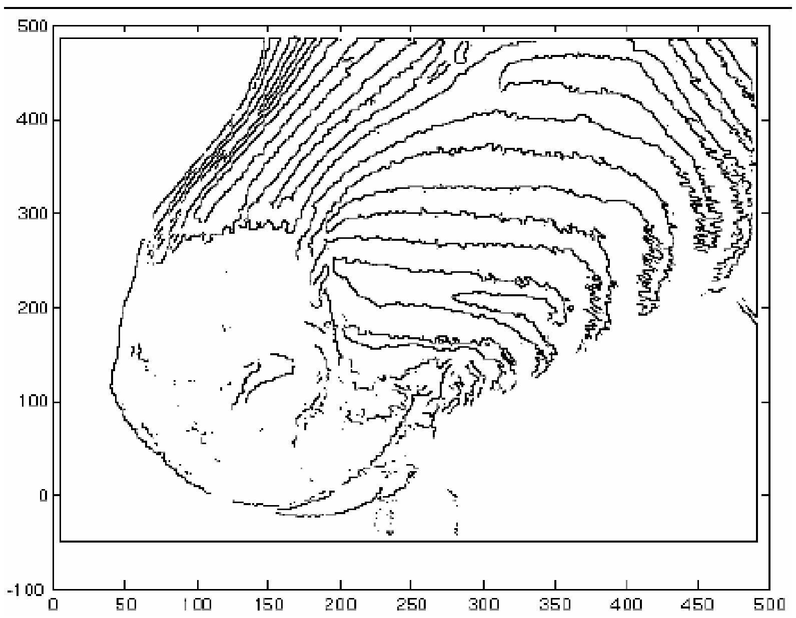


coarse
scale,
high
threshold

Edges…

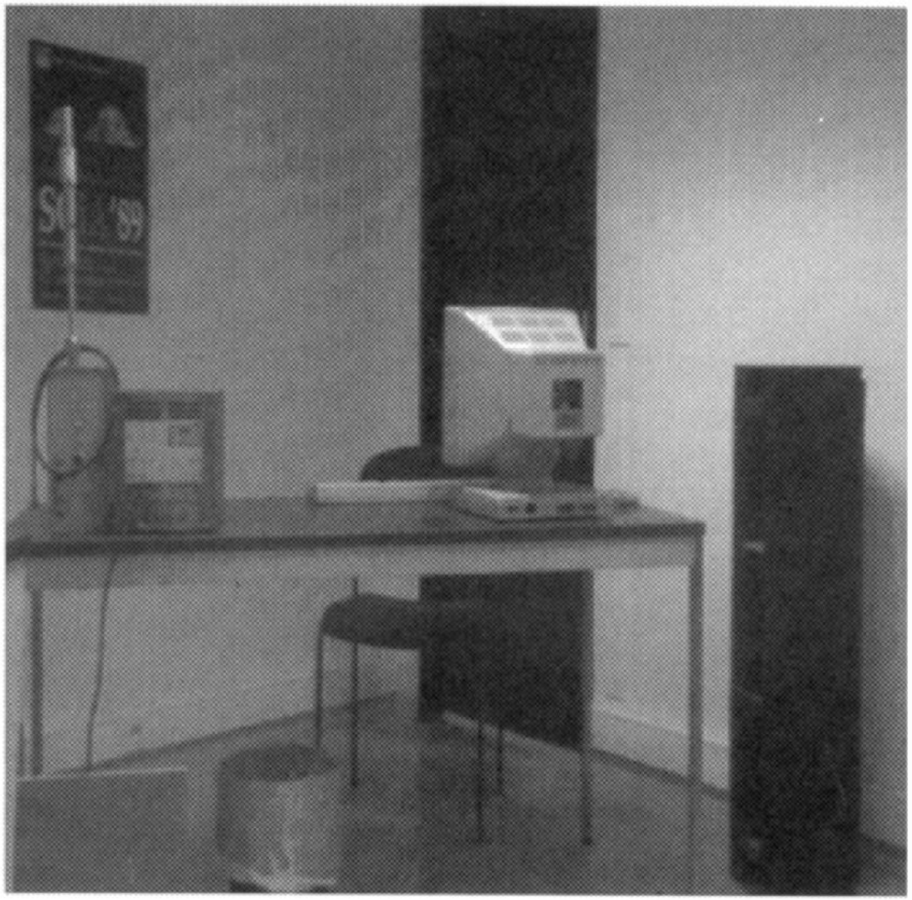
- sigma = 4
- sigma = 2

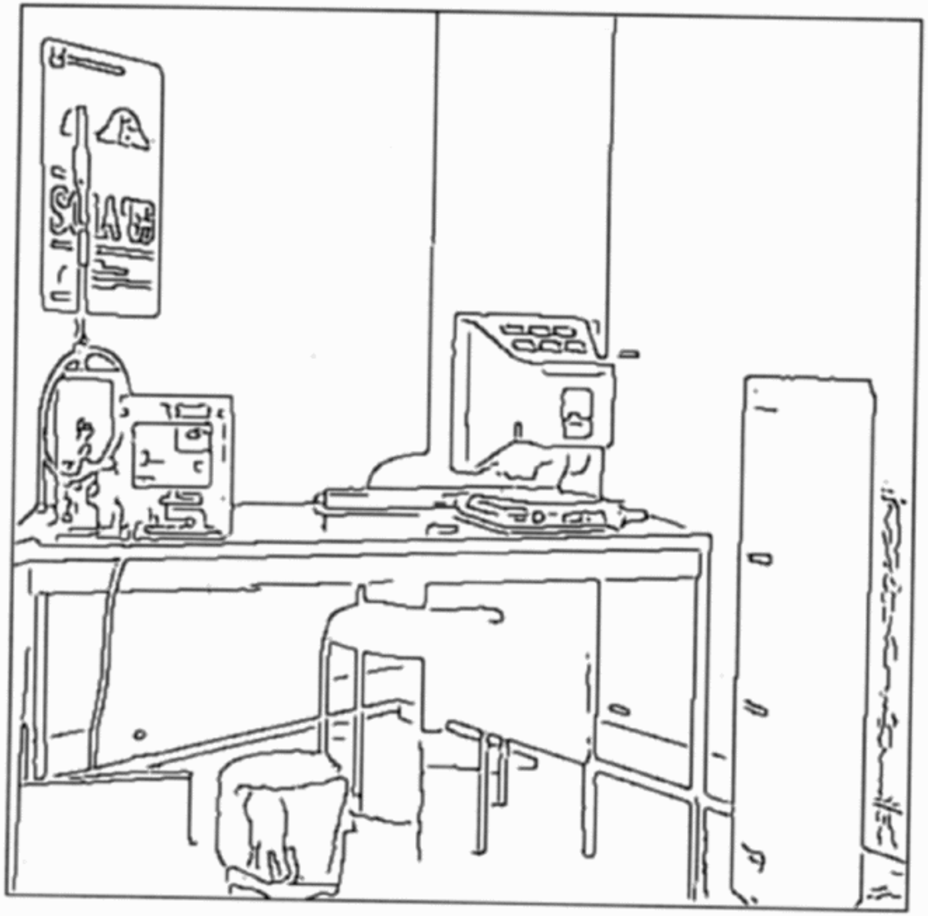




Edges…

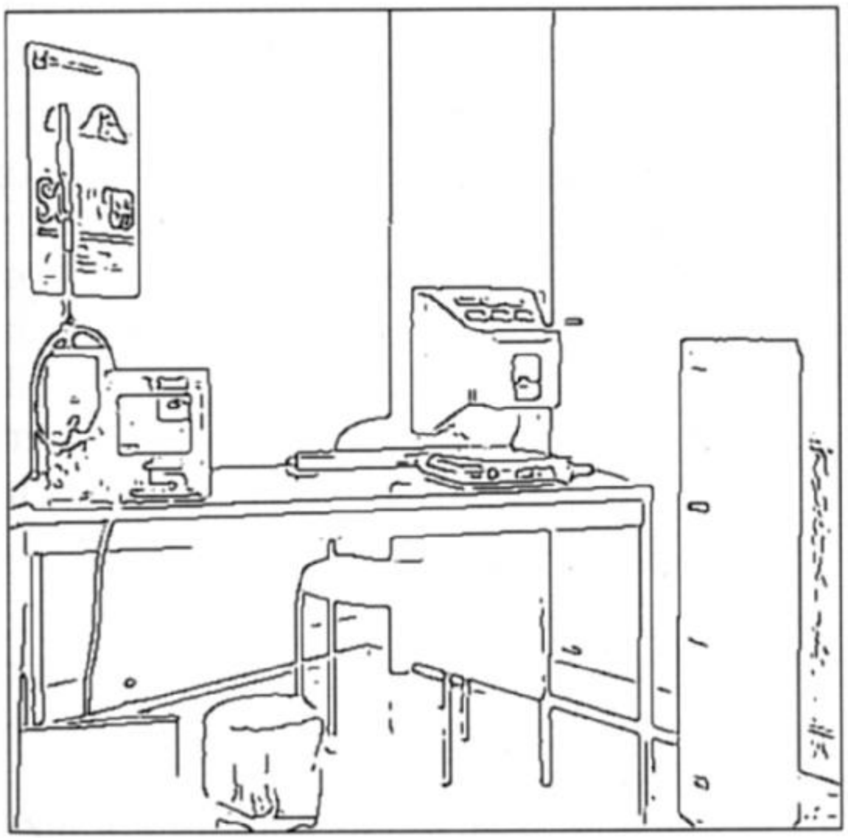
- contour following with 2 thresholds

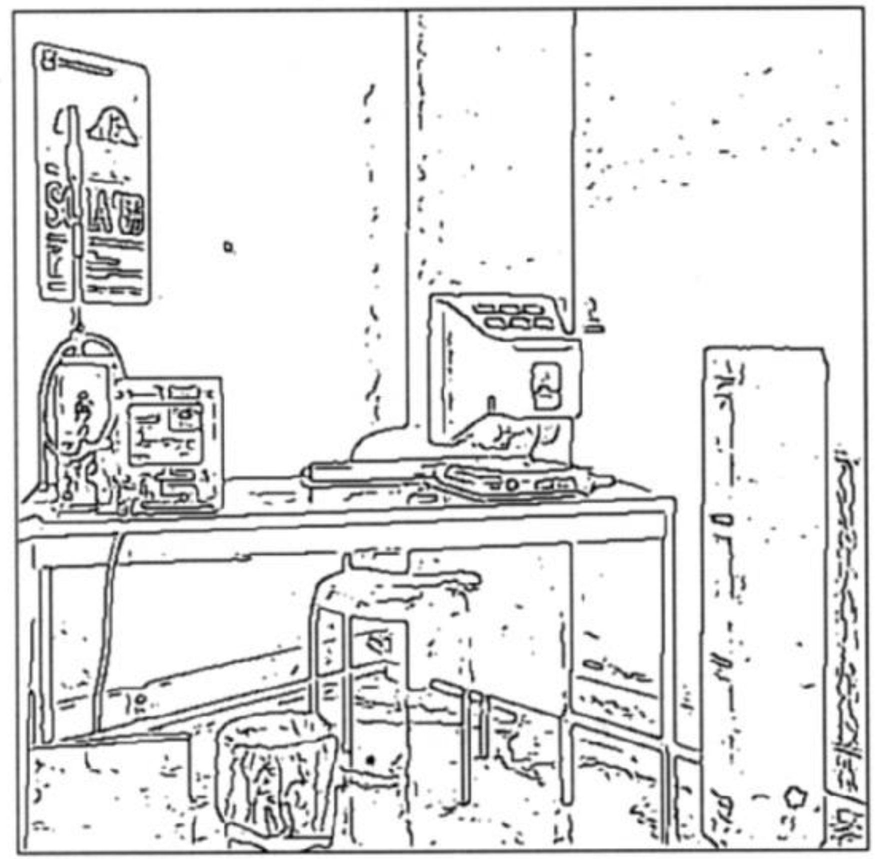




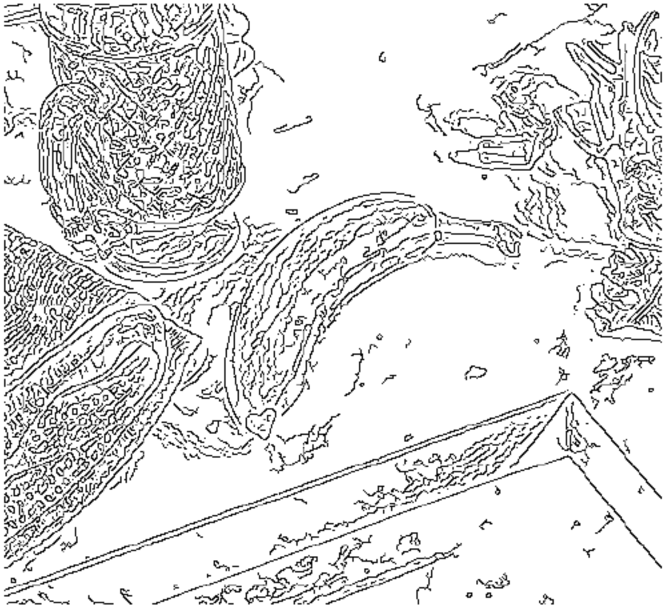
Edges…

– different thresholds

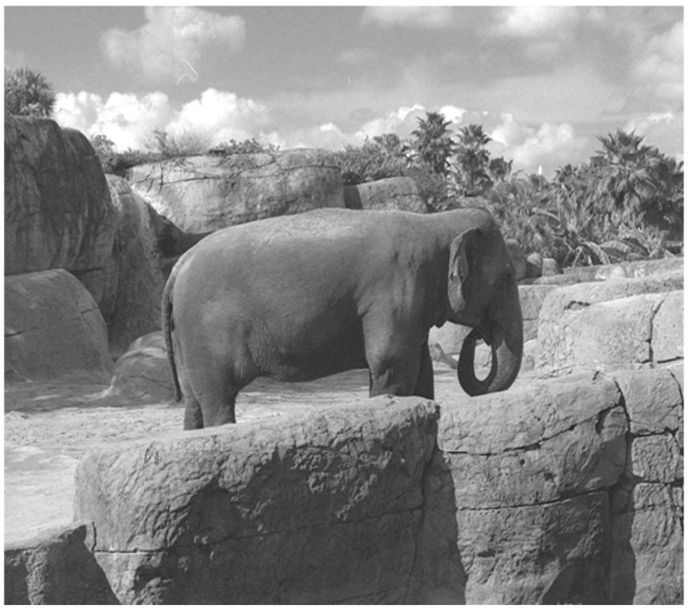


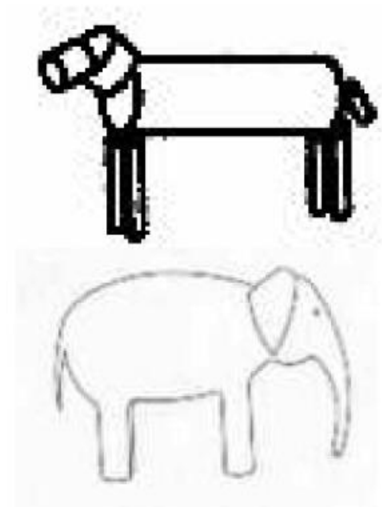


line drawing vs. edge detection



Edges…

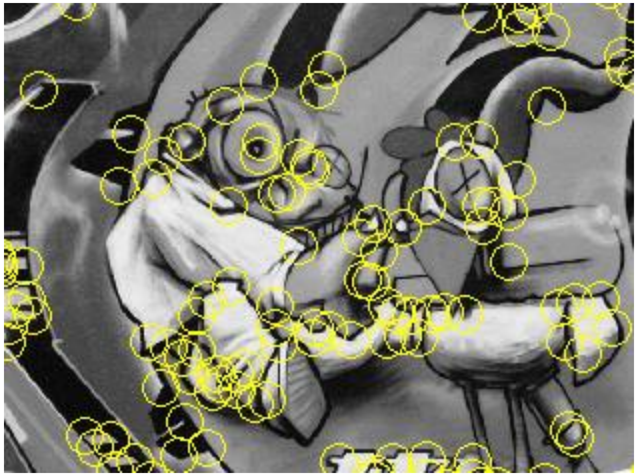




Match “model” to measurements?

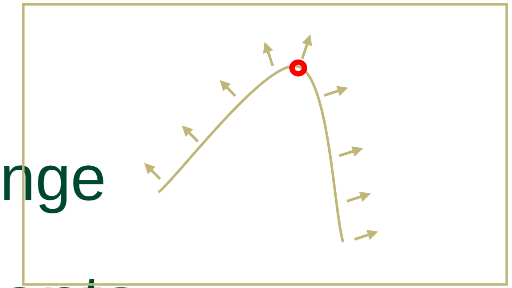
Local Features

- Corners, blobs
  - Two dimensional signal change
  - More complex local structures

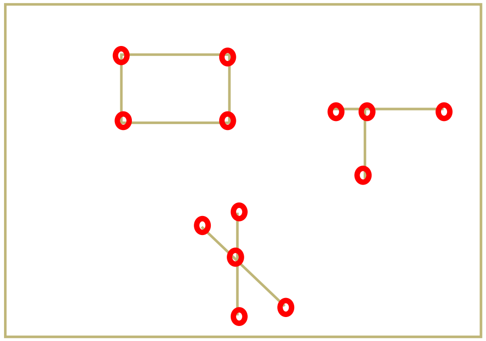


Feature detectors
Contour based methods

- Detecting curvature change
  - Detecting edges
  - Detecting sudden edge orientation change



- Detecting intersections of line segments
  - Detecting edges
  - Fitting line segments to the edges i.e., Hough transform
  - Finding intersections



Feature detectors

- Intensity based methods

Eigenvalues-reminder

- Singular value decomposition

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix} \cdot \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} \cdot \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix}^T = U \cdot D \cdot V^T$$

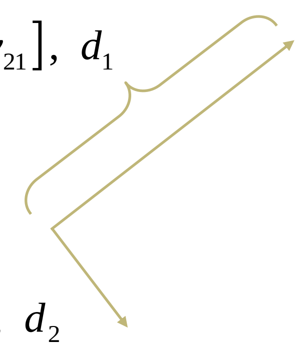
eigenvectors $[v_{11}\ v_{21}]^T, \ [v_{12}\ v_{22}]^T$ $\qquad \mathbf{U}\mathbf{U}^T = \mathbf{V}\mathbf{V}^T = 1 \quad \mathbf{U}^{-1} = \mathbf{U}^T \quad \mathbf{V}^{-1} = \mathbf{V}^T$

eigenvalues $d_1, \ d_2 \ \geq \ 0$

determinant $\det(A) = ad - cb = d_1 d_2$

Eigenvector, eigenvalue

$[v_{11}\ v_{21}], \ d_1$



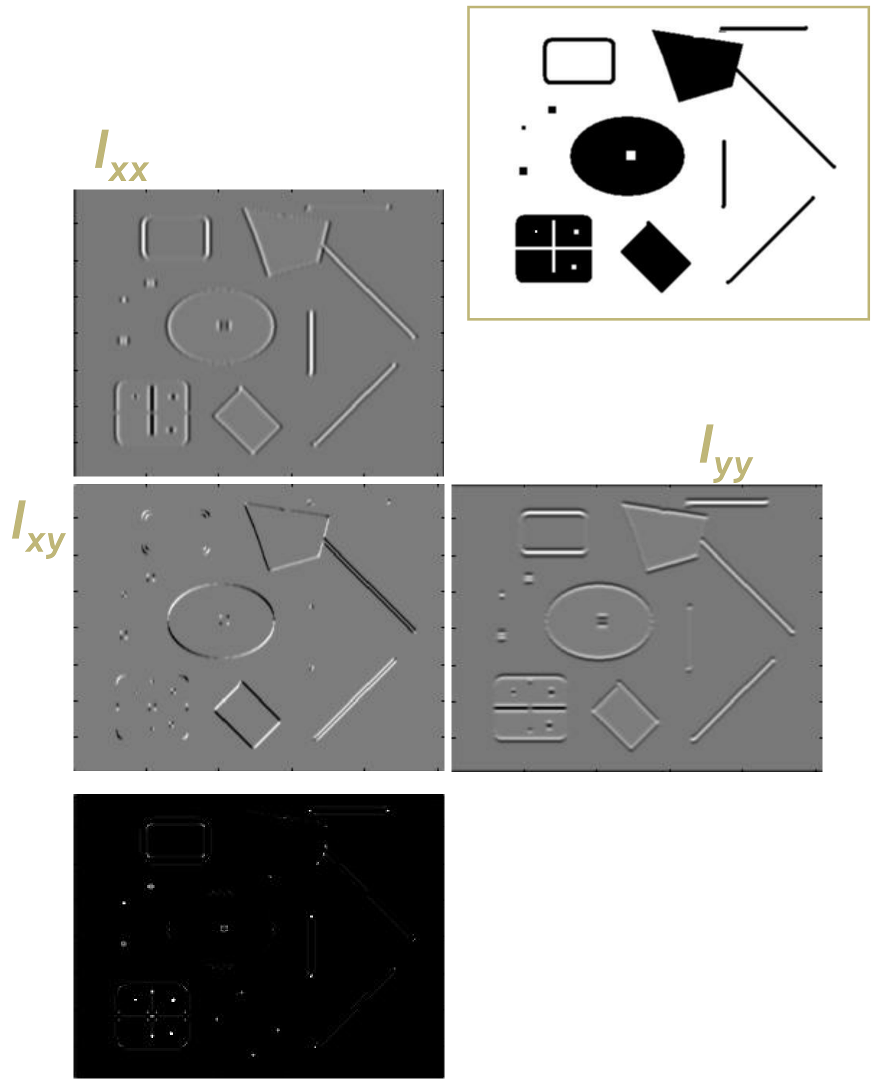
$[v_{12}\ v_{22}], \ d_2$

Feature detectors
Intensity based methods [Beaudet'78]

- Hessian determinant

$$Hessian(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

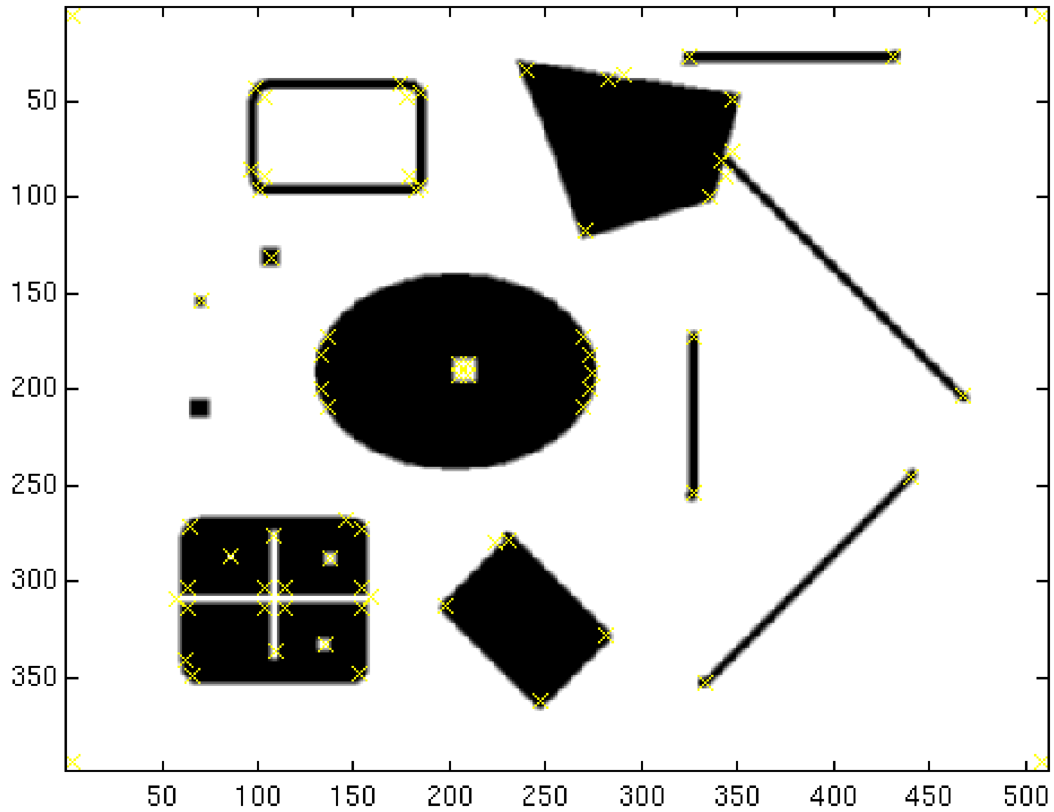
$$\det(Hessian(I)) = I_{xx} I_{yy} - I_{xy}^2$$



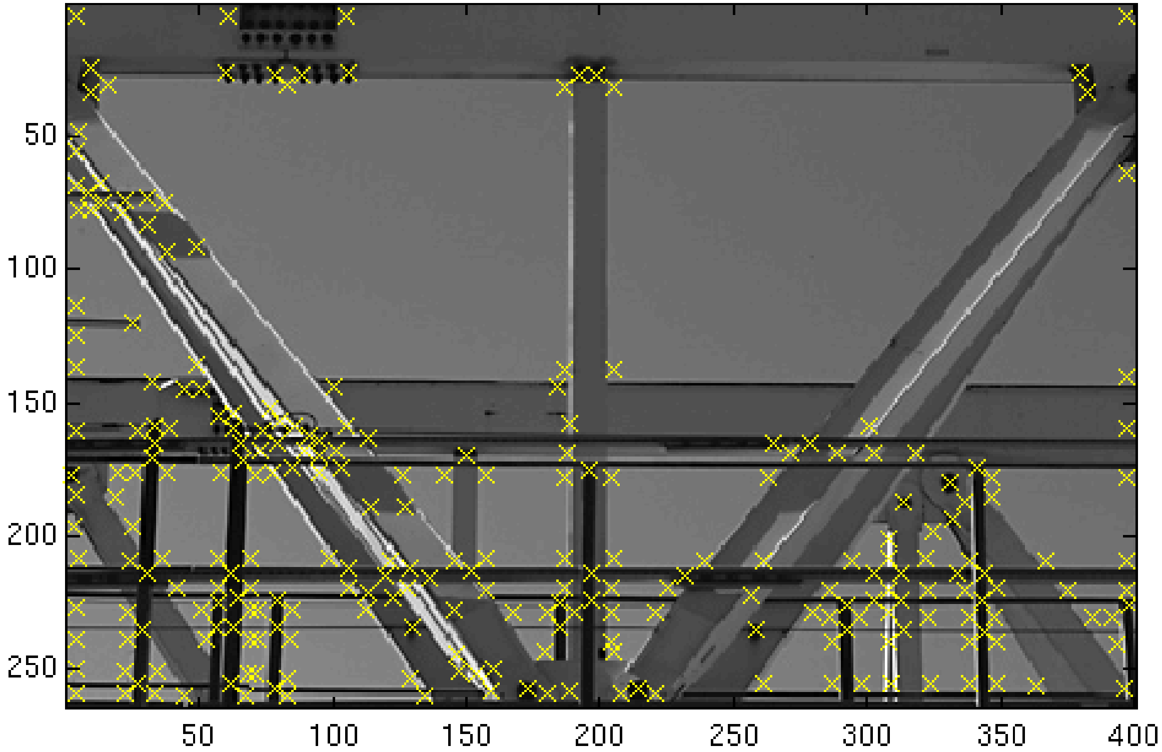
In Matlab:

$$I_{xx}.*I_{yy} - (I_{xy})\char`^2$$

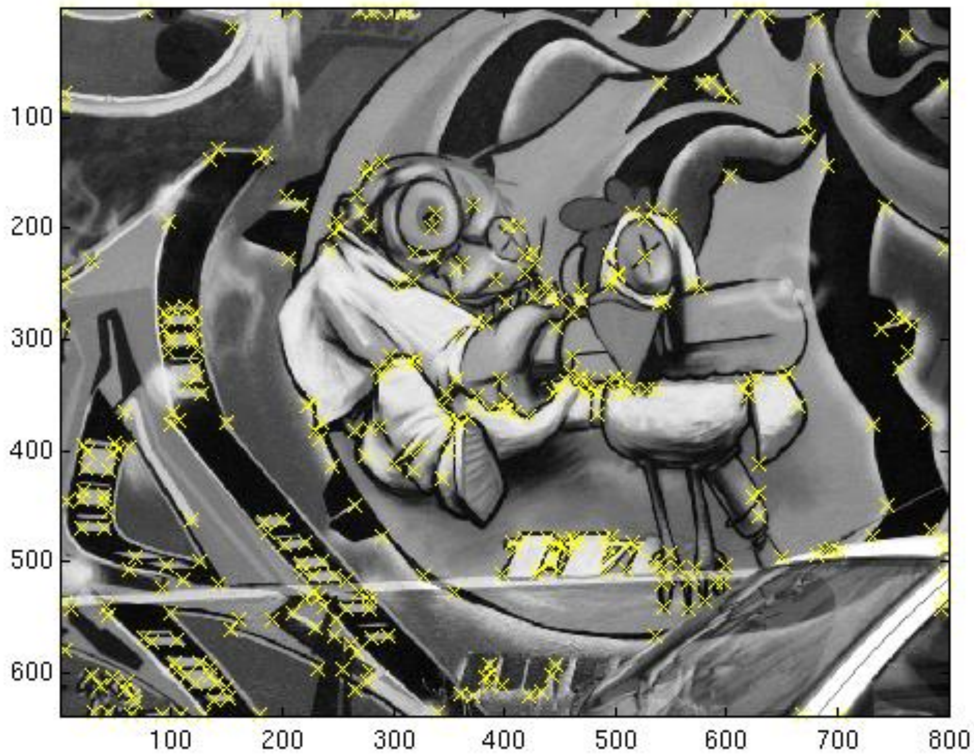
Feature detectors
Intensity based methods [Beaudet'78]



Feature detectors
Intensity based methods [Beaudet'78]



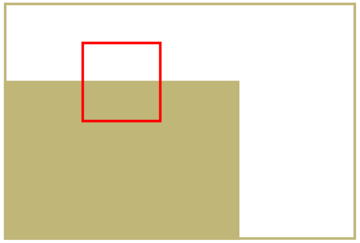
Feature detectors
Intensity based methods [Beaudet'78]



Feature detectors

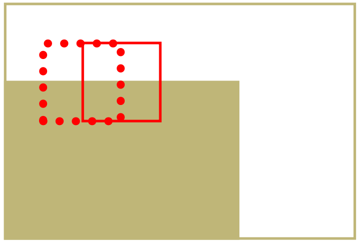
Intensity based methods [Moravec'77]

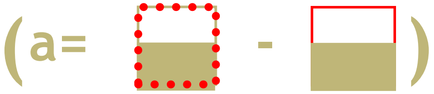
- Autocorrelation function



Feature detectors
Intensity based methods [Moravec'77]

- Autocorrelation function

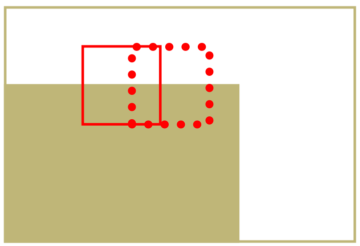




Feature detectors

Intensity based methods [Moravec'77]

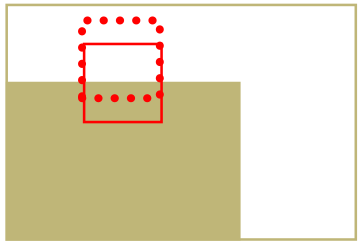
- Autocorrelation function

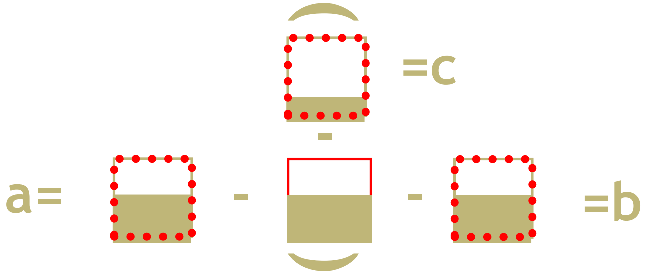


Feature detectors

Intensity based methods [Moravec'77]

- Autocorrelation function

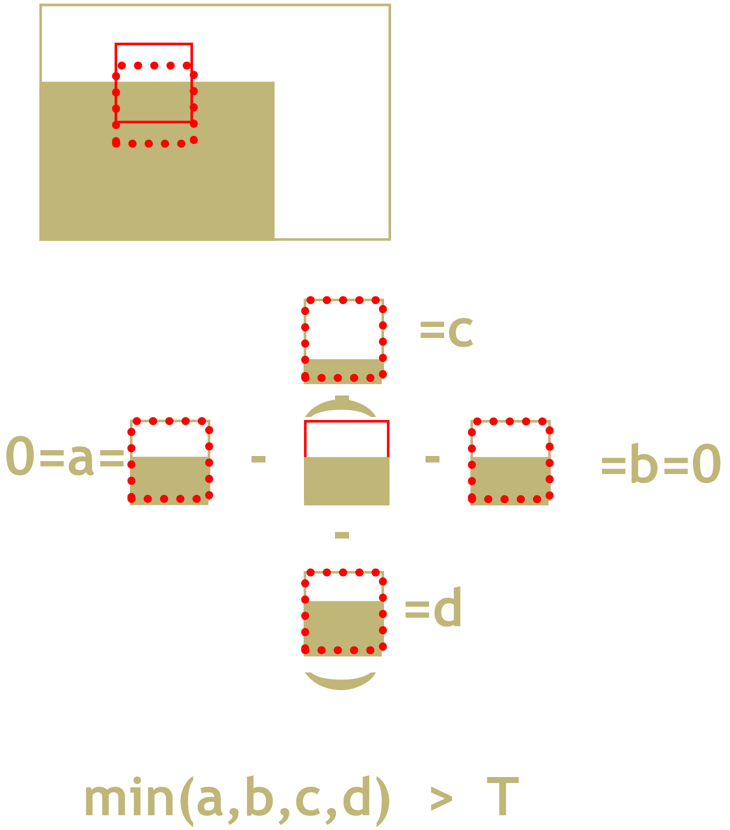




Feature detectors

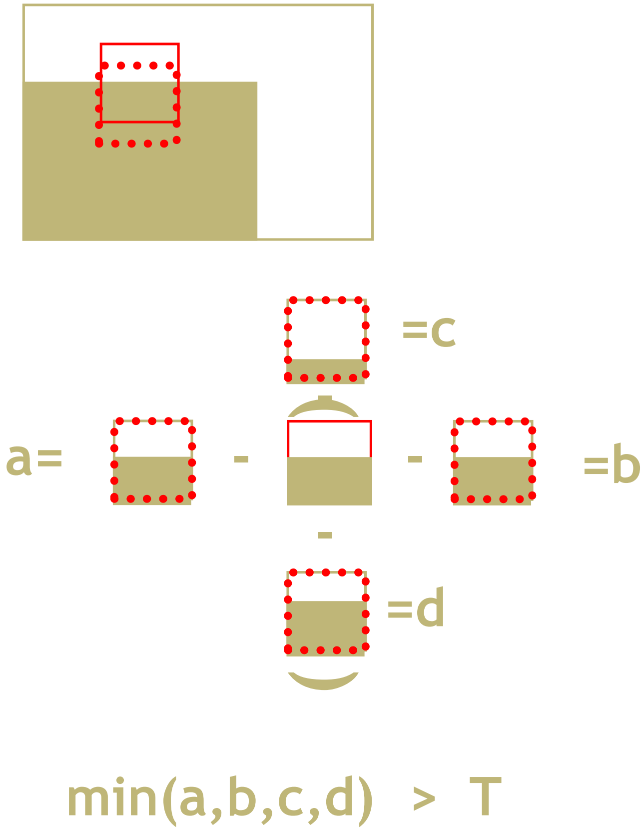
Intensity based methods [Moravec'77]

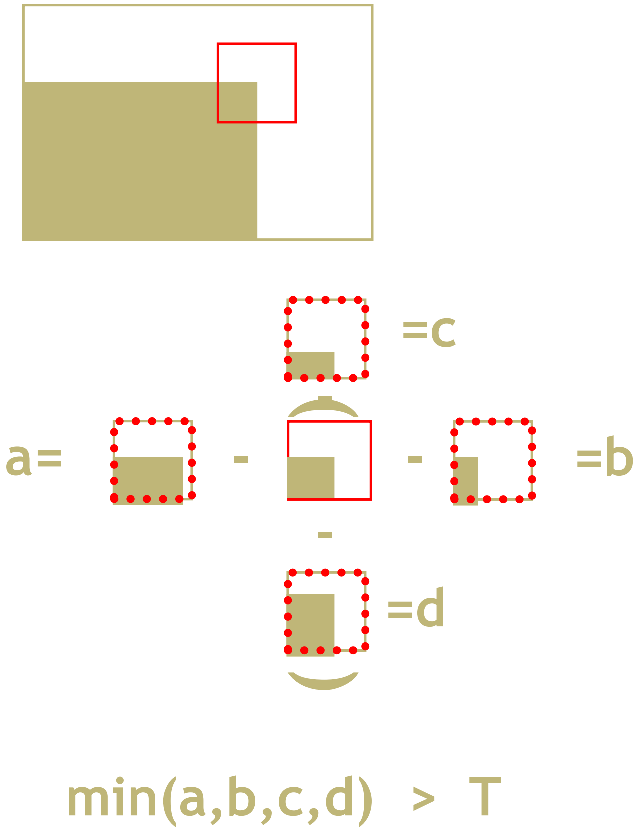
- Autocorrelation function



Feature detectors
Intensity based methods [Moravec'77]

- Autocorrelation function

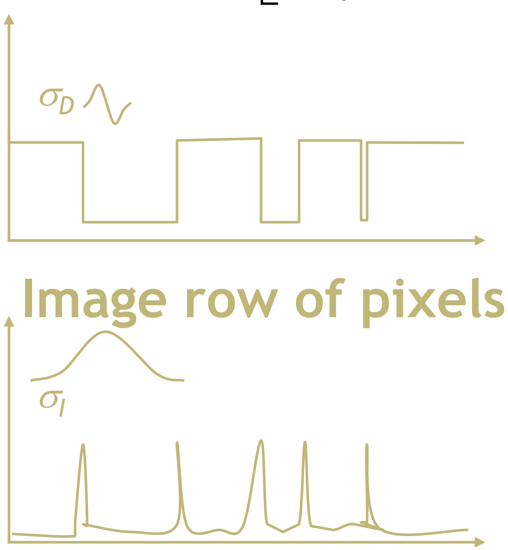


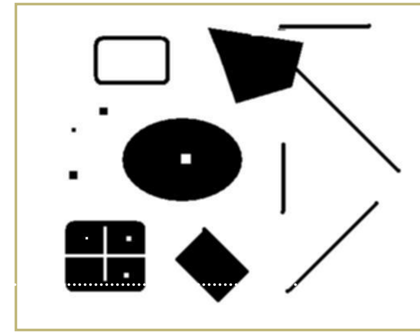


Feature detectors
Intensity based methods [Harris'88]

- Second moment matrix
  autocorrelation matrix

$$\mu(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$



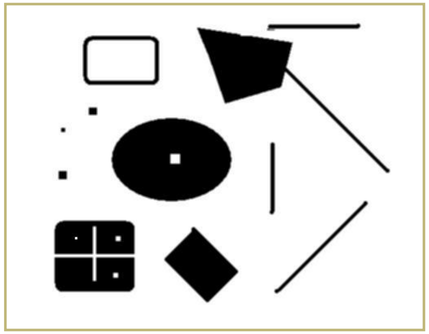


I_y

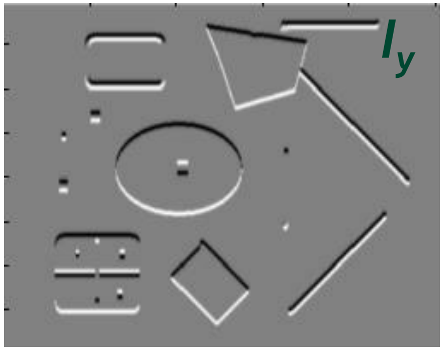
Feature detectors
Intensity based methods [Harris'88]

- Second moment matrix
  autocorrelation matrix

$$\mu(\sigma_I,\sigma_D)=g(\sigma_I)*\begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

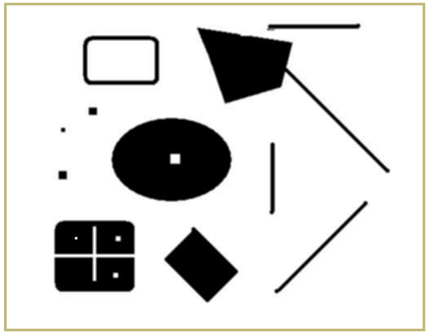


1. Image derivatives
   $g_x(\sigma_D)$, $g_y(\sigma_D)$,

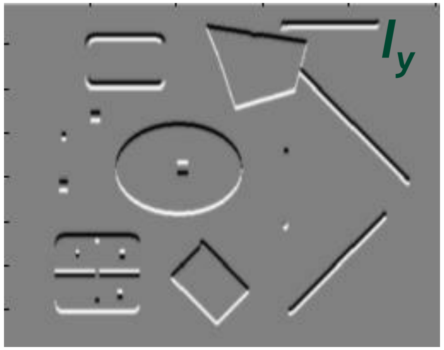


Feature detectors
Intensity based methods [Harris'88]

- Second moment matrix
  autocorrelation matrix

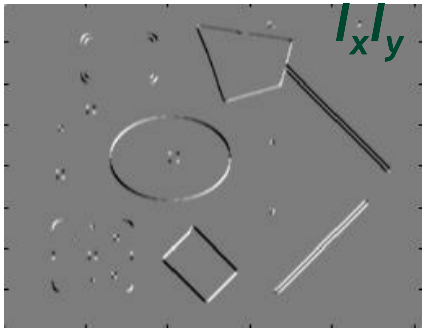


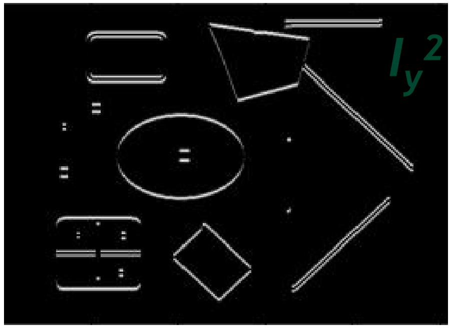
$$\mu(\sigma_I,\sigma_D)=g(\sigma_I)*\begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$



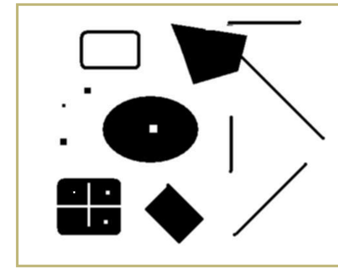
1. Image derivatives
   $g_x(\sigma_D)$, $g_y(\sigma_D)$,

2. Square of derivatives





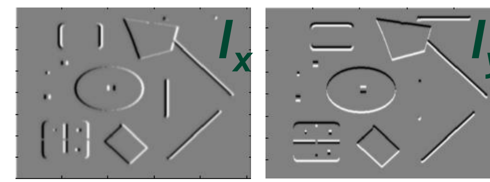
Feature detectors
Intensity based methods [Harris'88]



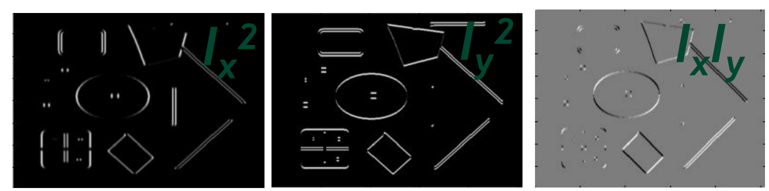
- Second moment matrix autocorrelation matrix

$$\mu(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

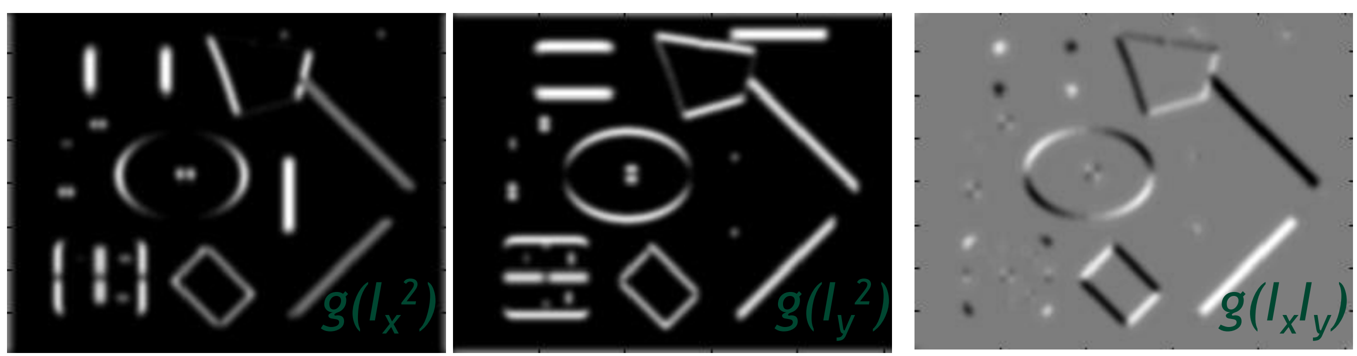
1. Image derivatives



2. Square of derivatives

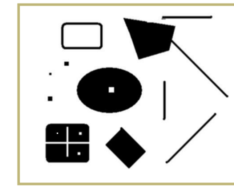


3. Gaussian filter $g(\sigma_I)$



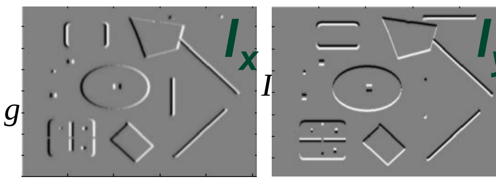
Feature detectors
Intensity based methods [Harris'88]

- Second moment matrix
  autocorrelation matrix

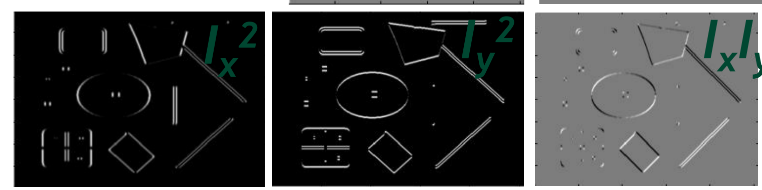


1. Image derivatives

$\mu(\sigma_I,\sigma_D) = g$



2. Square of derivatives



3. Gaussian filter $g(\sigma_I)$

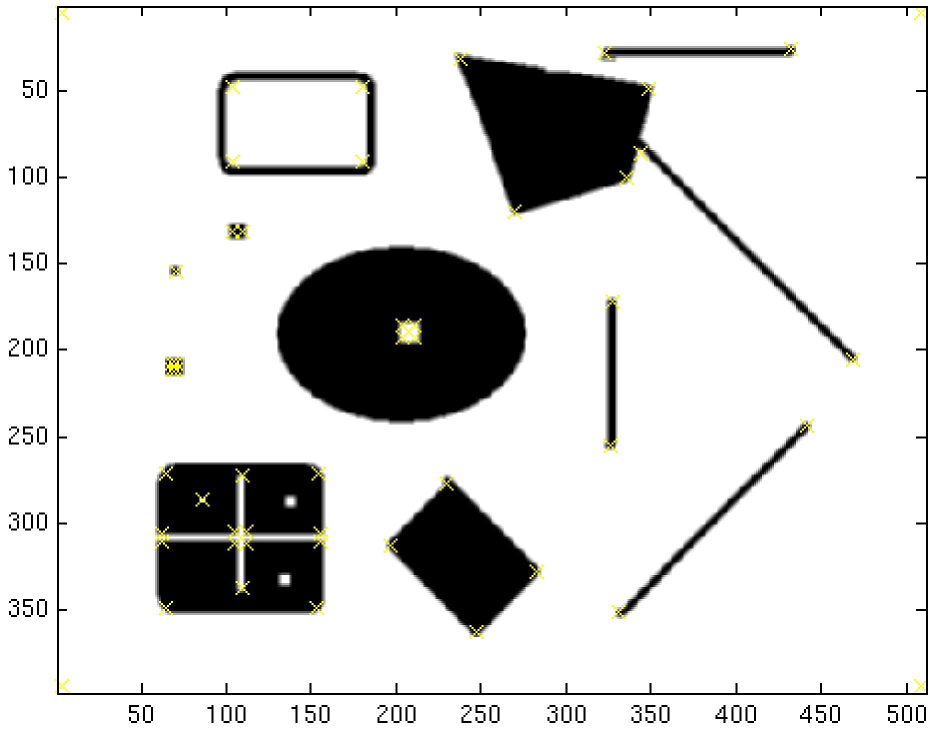
4. Cornerness function - both eigenvalues are strong

$$har = \det[\mu(\sigma_I,\sigma_D)] - \alpha[\mathrm{trace}(\mu(\sigma_I,\sigma_D))] =$$
$$g(I_x^2)g(I_y^2) - [g(I_xI_y)]^2 - \alpha[g(I_x^2)+g(I_y^2)]^2$$

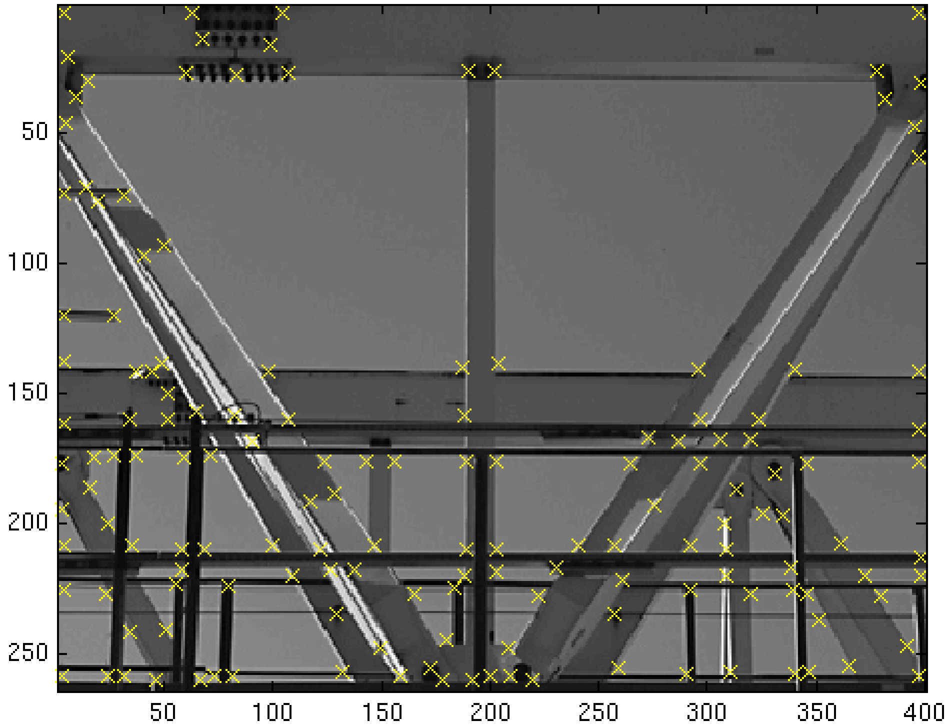
5. Non-maxima suppression

Questions: goo.gl/K61te5

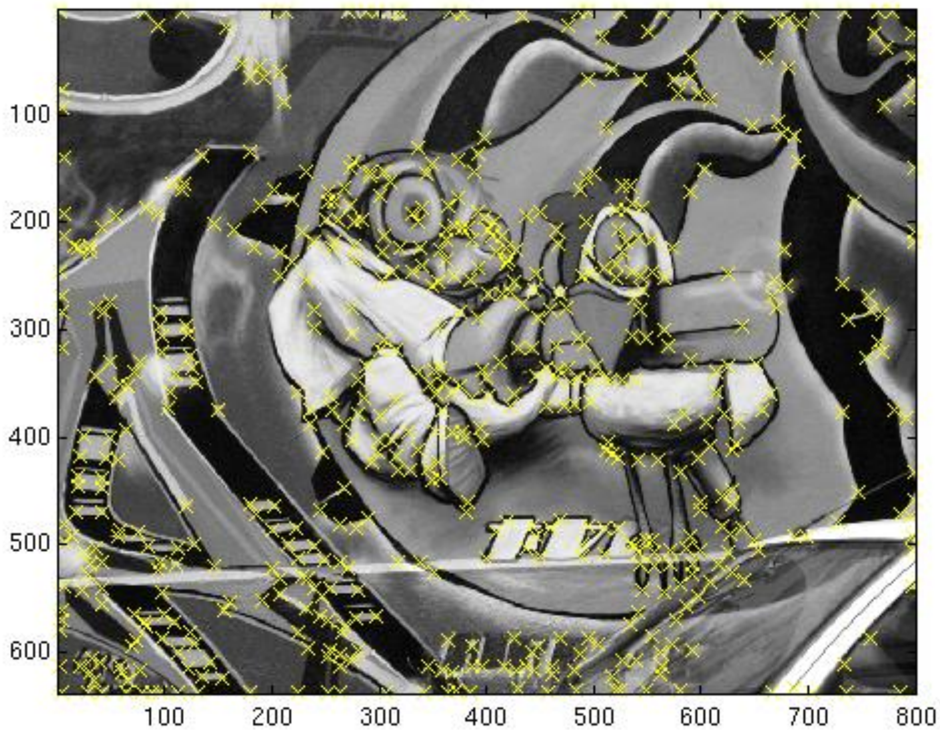
Feature detectors
Intensity based methods [Harris'88]



Feature detectors
Intensity based methods [Harris'88]



Feature detectors
Intensity based methods [Harris'88]

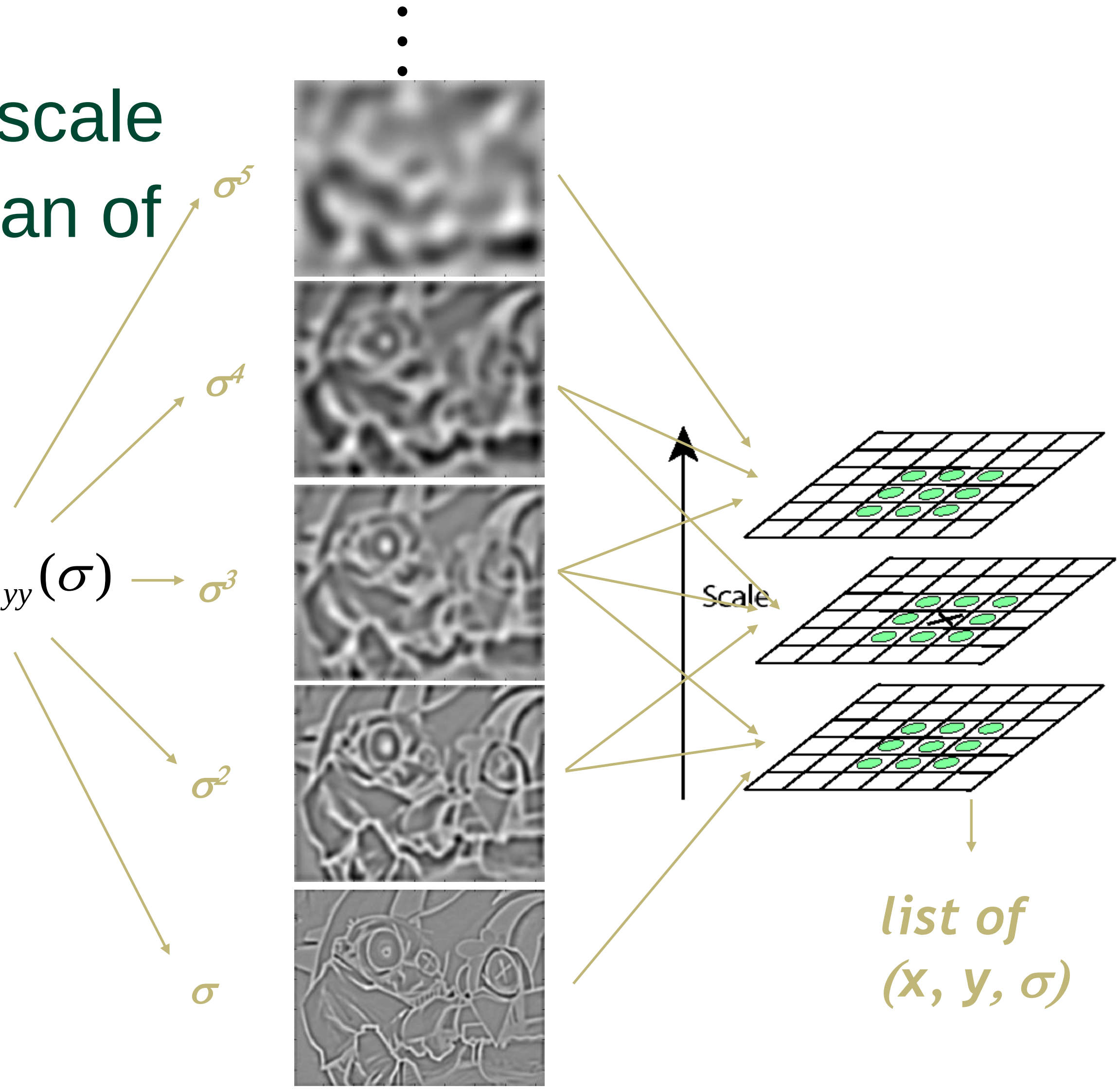


Scale-Space Methods

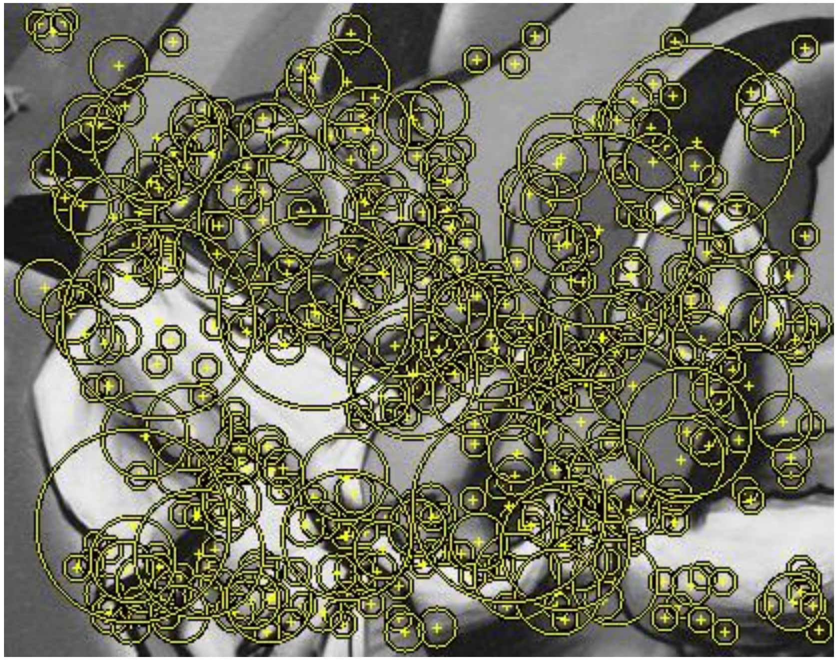
Scale invariant detectors
Laplacean of Gaussian

- Local maxima in scale space of Laplacean of Gaussian LoG

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



Scale invariant detectors
Laplacean of Gaussian



Scale invariant detectors Difference of Gaussian

- LoG –> diffution quation -> derivative to scale

$$\frac{\partial L}{\partial s} = \vec{\nabla} \cdot \vec{\nabla} L = \Delta L = L_{xx}(\sigma) + L_{yy}(\sigma)$$

scale normalized Laplacean

$$\sigma^2 (L_{xx}(\sigma) + L_{yy}(\sigma))$$

$$s = \sigma^2$$

$$\frac{\partial L}{\partial \sigma} \approx \frac{L(k\sigma) - L(\sigma)}{k\sigma - \sigma}$$

$$(k-1)\sigma^2 \Delta L \approx L(k\sigma) - L(\sigma)$$

$L(k\sigma)$

-

$L(\sigma)$

=

$L(k\sigma) - L(\sigma)$

Questions: goo.gl/K61te5

Scale invariant detectors Difference of Gaussian

- Building scale space of Difference of Gaussian DoG
  - LoG –> diffution quation -> derivative to scale

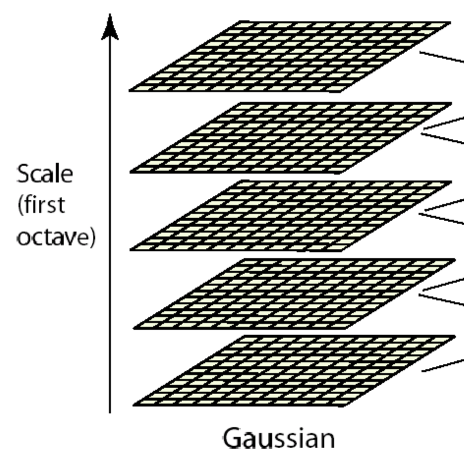
Original image

sampling with step
$\sigma^4 = 2$

σ

σ

$$\sigma = 2^{\frac{1}{4}}$$



Scale invariant detectors

- Detecting multiscale points – thousands of interest points

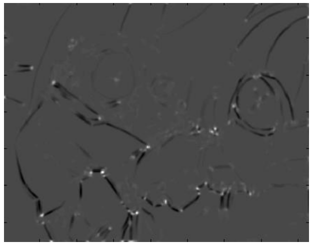
$$\mu(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

$$har = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\mathrm{trace}(\mu(\sigma_I, \sigma_D))]$$

$$\sigma_I = 1.6 \cdot \sigma_D$$

Scale invariant detectors

- Detecting multiscale points – thousands of interest points



σ

Computing Harris function

$$\mu(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

$$har = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\mathrm{trace}(\mu(\sigma_I, \sigma_D))]$$

$$\sigma_I = 1.6 \cdot \sigma_D$$

Scale invariant detectors

- Detecting multiscale points – thousands of interest points

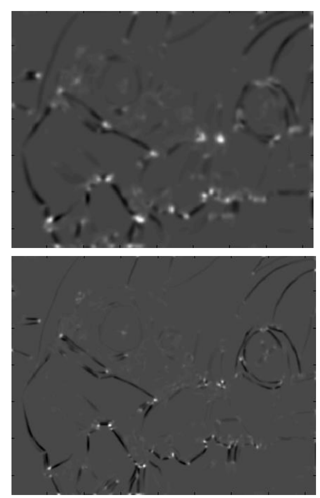
$$\mu(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

$$har = \det[\mu(\sigma_I,\sigma_D)] - \alpha[\mathrm{trace}(\mu(\sigma_I,\sigma_D))]$$

$$\sigma_I = 1.6 \cdot \sigma_D$$

σ^2

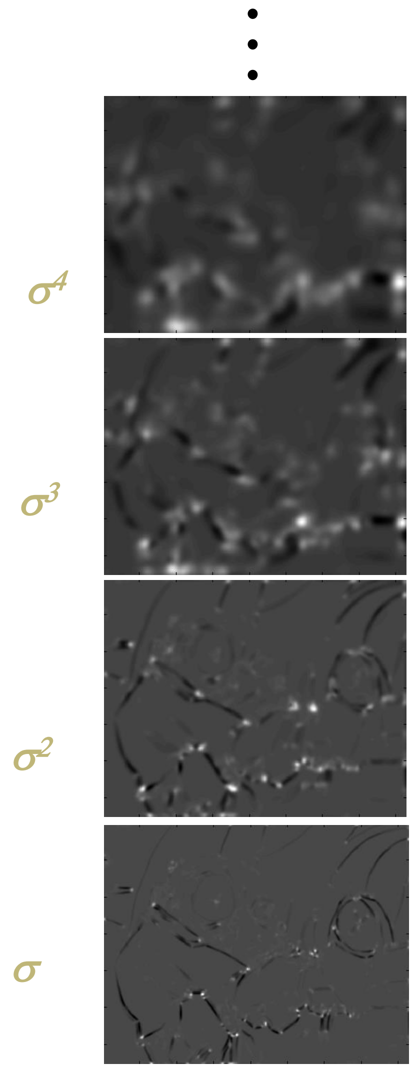
σ



Computing Harris function

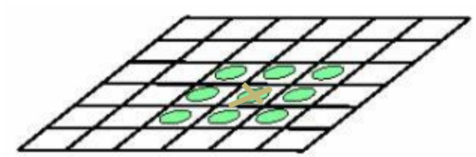
Scale invariant detectors

- Detecting multiscale points – thousands of interest points



Computing Harris function

list of (x, y, σ)



Detecting local maxima

$$\mu(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

$$har = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\mathrm{trace}(\mu(\sigma_I, \sigma_D))]$$

$$\sigma_I = 1.6 \cdot \sigma_D$$

Questions: goo.gl/K61te5

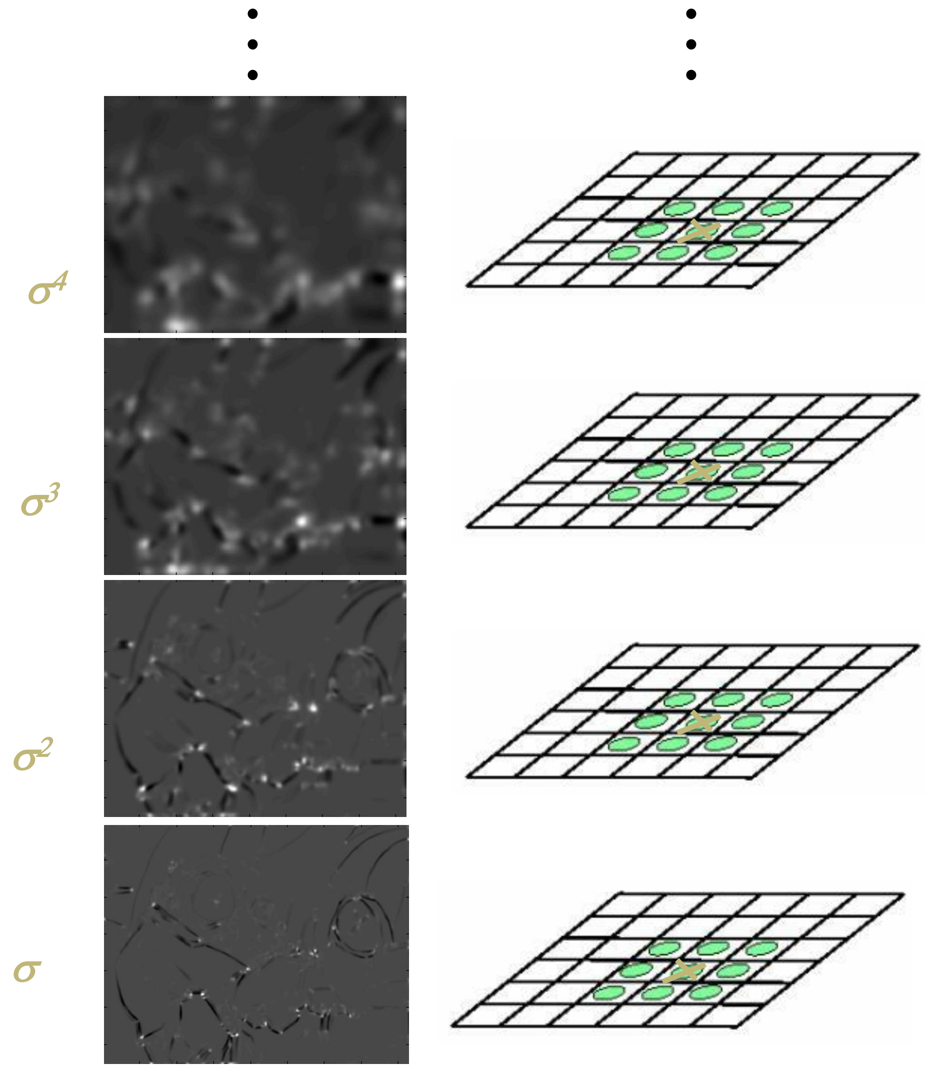
Scale invariant detectors

- Detecting multiscale points – thousands of interest points

$$\mu(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

$$har = \det[\mu(\sigma_I,\sigma_D)] - \alpha[\mathrm{trace}(\mu(\sigma_I,\sigma_D))]$$

$$\sigma_I = 1.6 \cdot \sigma_D$$



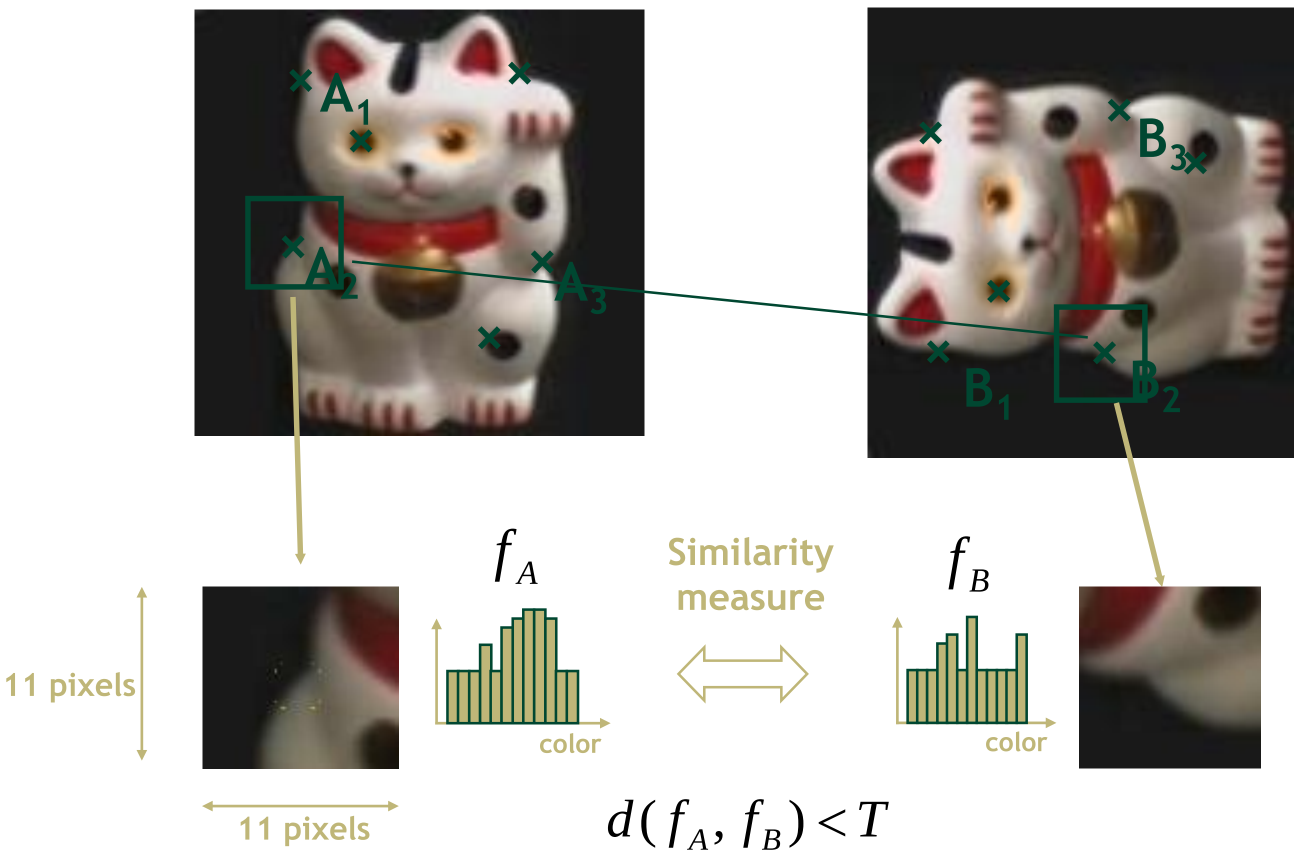
Computing Harris function   Detecting local maxima

Summary

- Canny edge detection
- Hessian & second moment matrix, autocorrelation function
- Feature detection, types of features – methods to detect them.

Image transformation
Matching patches

- Extracting and matching patches



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

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

Descriptors history

Accuracy

- Normalized cross-correlation (NCC) [~ 1960s]
- Gaussian derivative-based descriptors
  - Differential invariants [Koenderink and van Doorn'87]
  - Steerable filters [Freeman and Adelson'91]
- Moment invariants [Van Gool et al.'96]
- SIFT [Lowe'99]
- Shape context [Belongie et al.'02]
- Gradient PCA [Ke and Sukthankar'04]

Efficiency

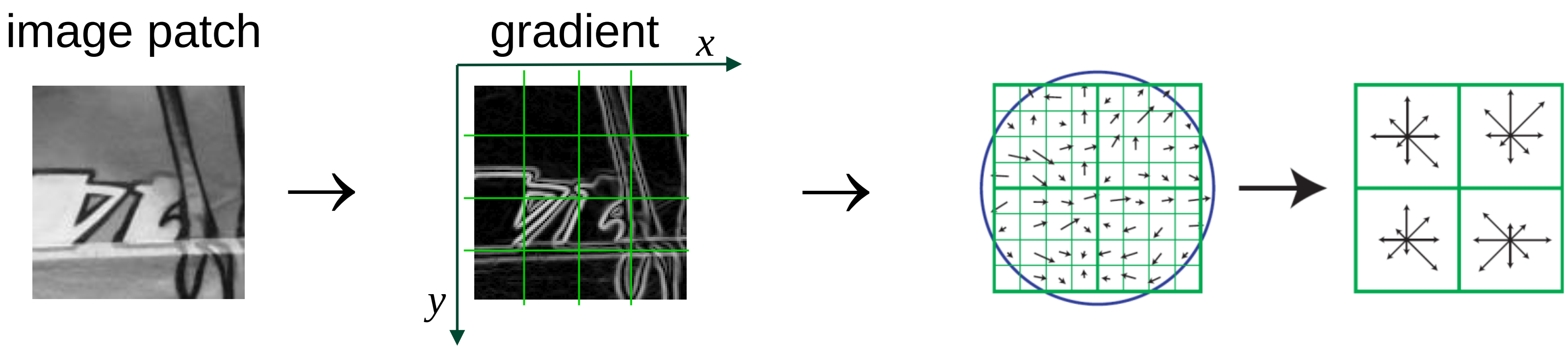
- SURF descriptor [Bay et al.'08]
- BRIEF [Calonder et al. 2010]

Machine learning

- Learning descriptors from image data [Brown et al 2010, …]
- Neural Networks [Zagoruyko et al 2015, …]

SIFT descriptor [Lowe'99]

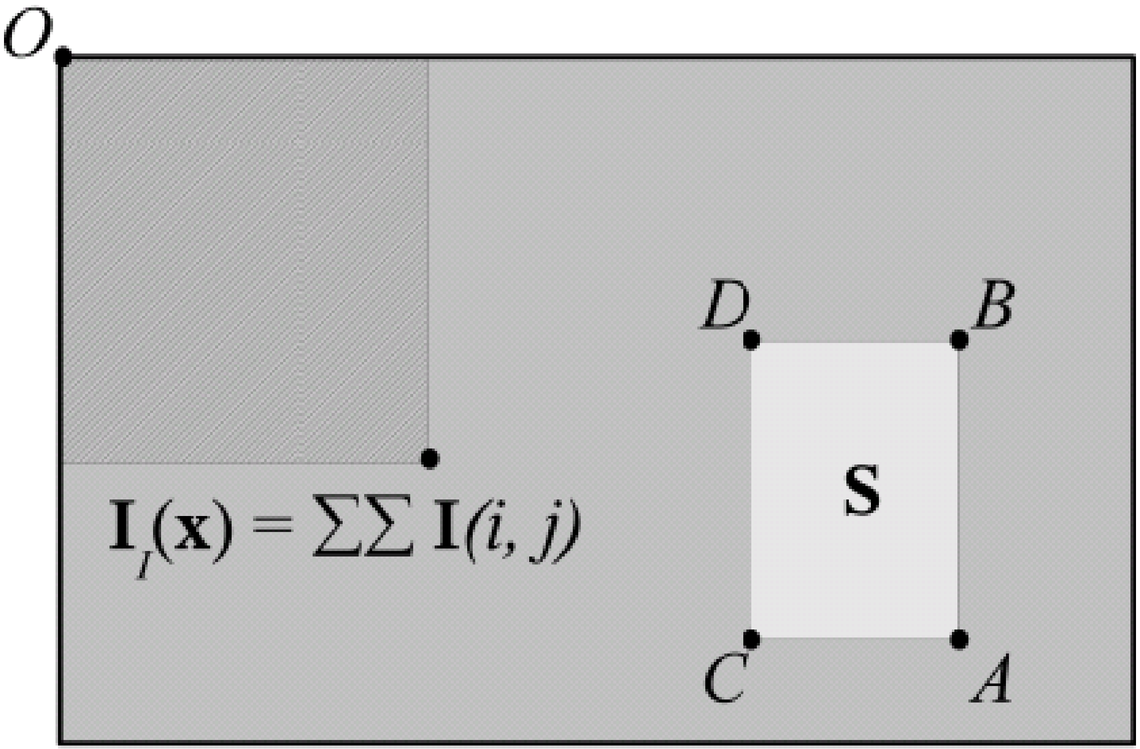
- Spatial binning and binning of the gradient orientation
- 4x4 spatial grid, 8 orientations of the gradient, dim 128
- Soft-assignment to spatial bins
- Normalization of the descriptor to norm one (robust to illumination)
- Comparison with Euclidean distance



SIFT Descriptor

- By far the most commonly used distinguished region descriptor:
  - fast
  - compact
  - works for a broad class of scenes
  - source code available
- Large number of ad hoc parameters) Enormous follow up literature on both “improvements” and improvements

The Integral image (Sum Table)



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

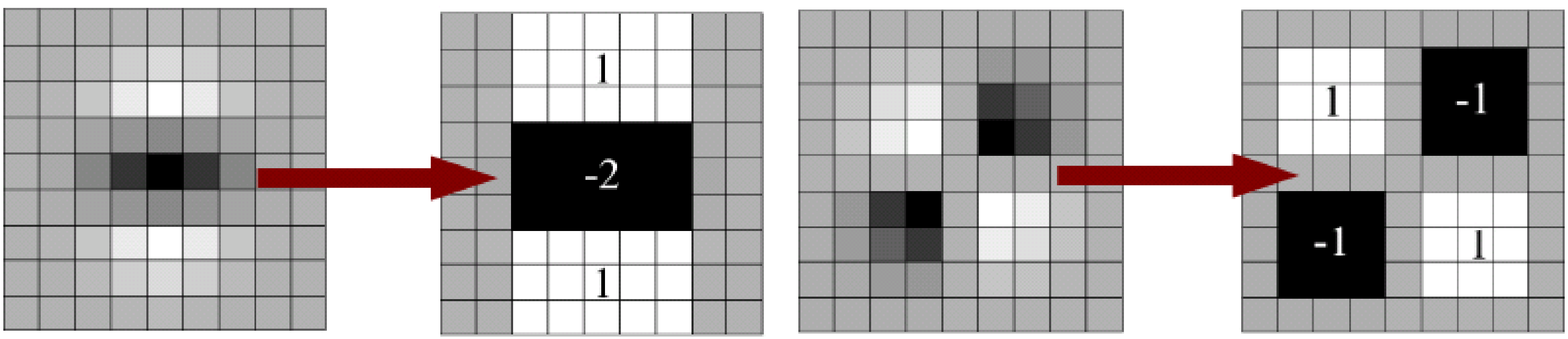
To calculate the sum in the DBCA rectangle, only 3 additions are needed

SURF Detection

- Hessian-based interest point localization:
- $L_{xx}(x,y,\sigma)$ is the convolution of the *Gaussian* second order derivative with the image

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

- Approximate second order derivatives with box filters filters

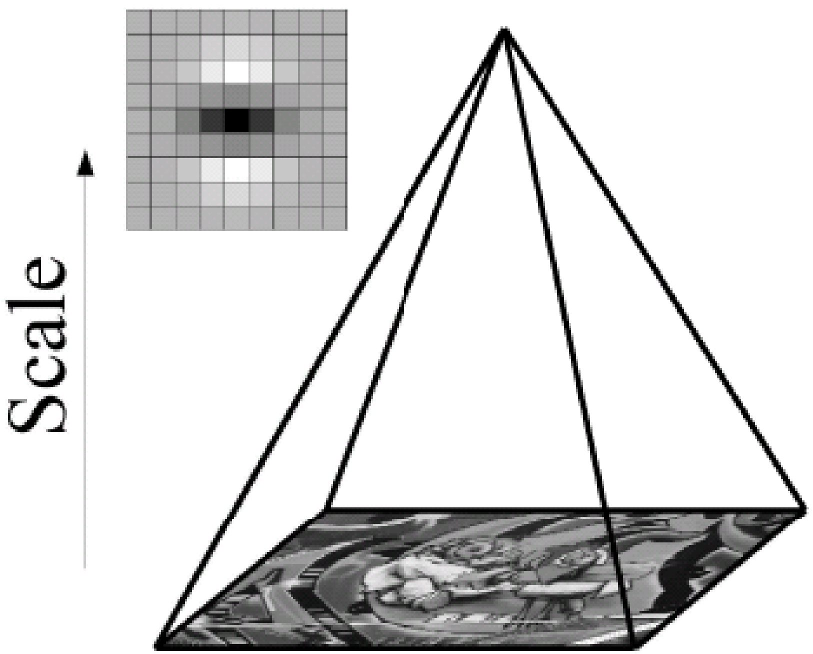


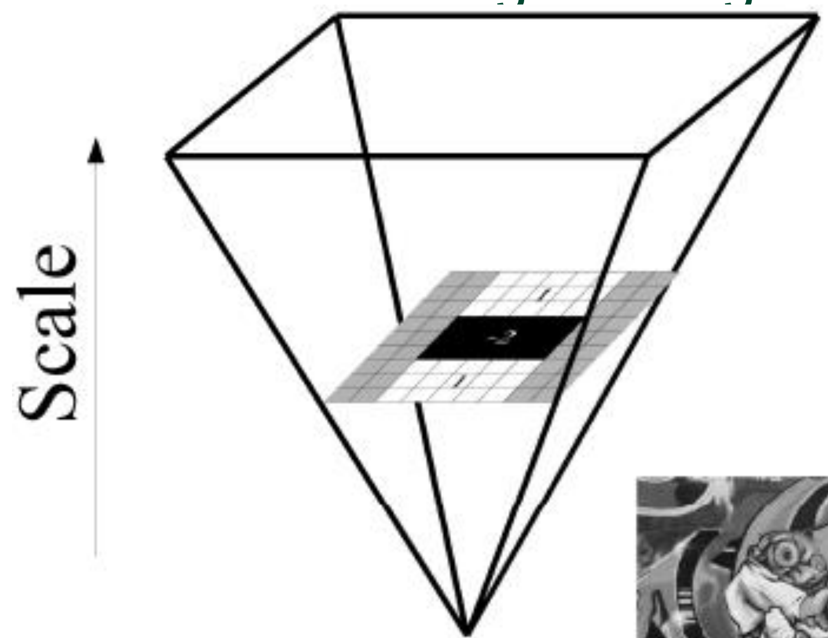
L_{yy} L_{xy}

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool , SURF: Speeded Up Robust Features, ECCV 2006.

SURF Detection

- Scale analysis easily handled with the integral image



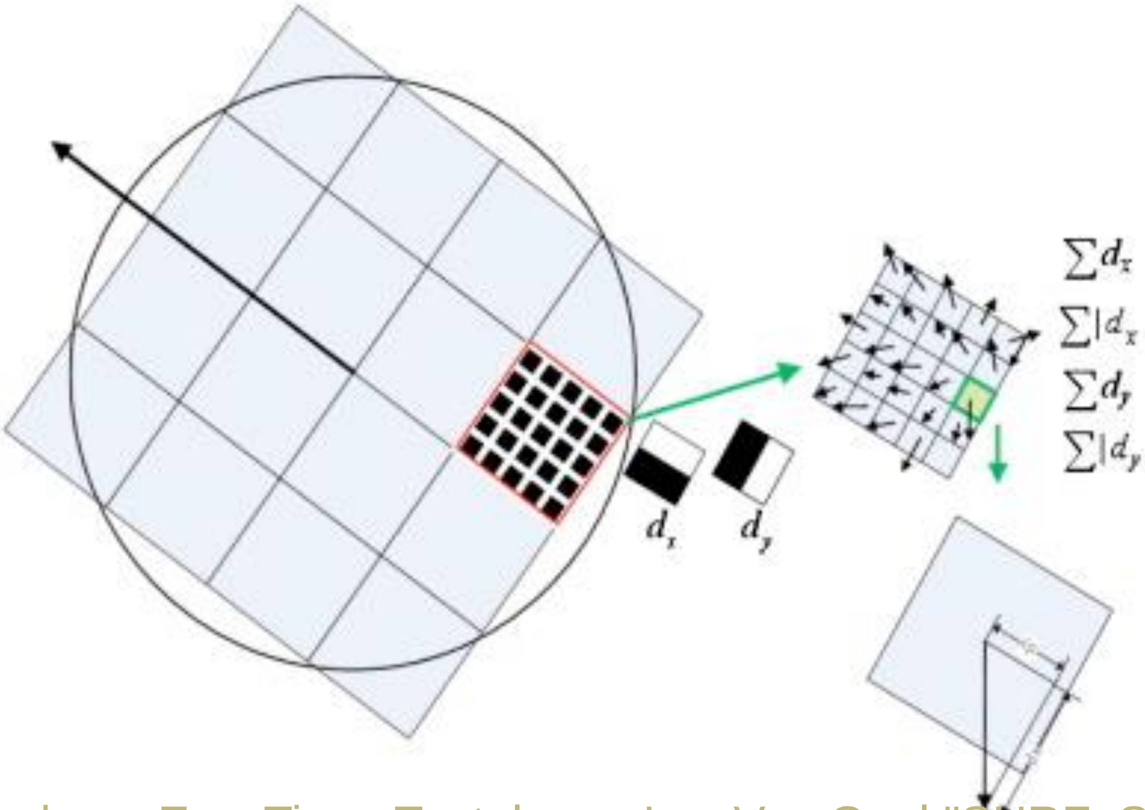


9 x 9, 15 x 15, 21 x 21, 27 x 27 → 39 x 39, 51 x 51 …

1st octave *2nd octave*

SURF: Speeded Up Robust Features

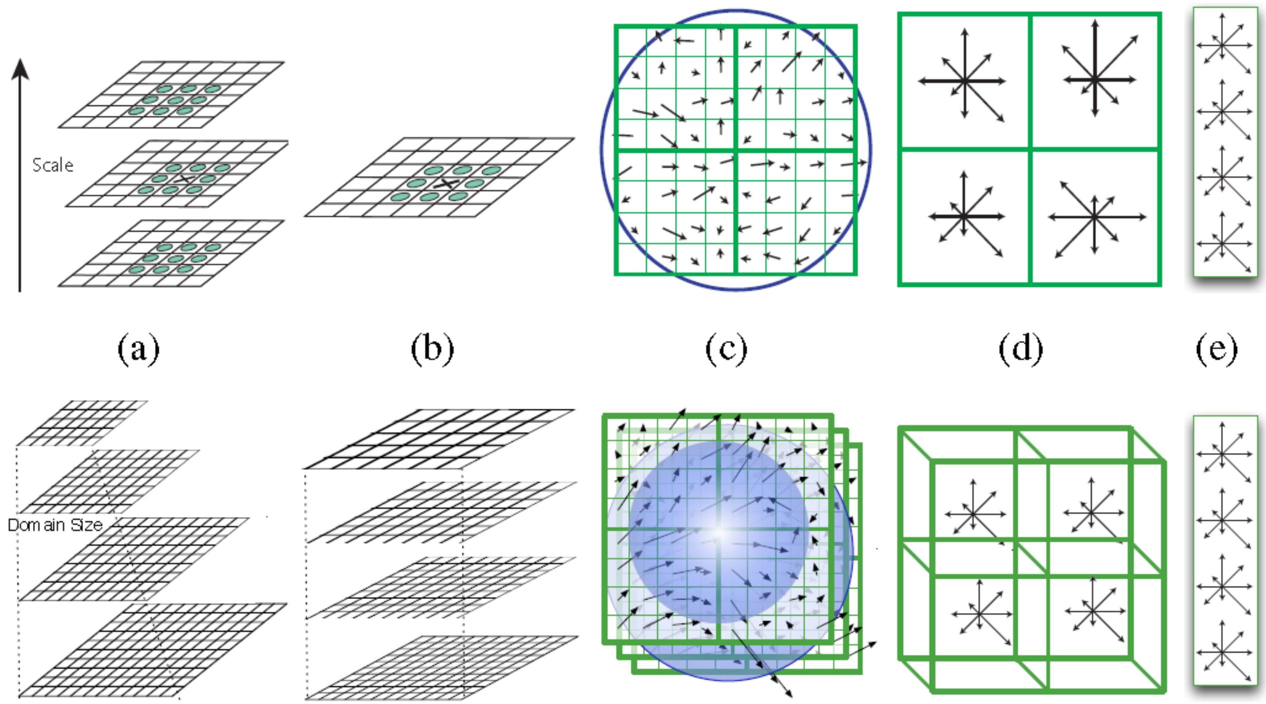
- Approximate derivatives with Haar wavelets
- Exploit integral images



Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008

DSP-SIFT

- Pooling SIFT descriptor across multiple scales



J. Dong, S. Soatto, Domain-Size Pooling in Local Descriptors: DSP-SIFT, CVPR2015
J. Dong, N Karianakis, D. Davis, J. Hernandez, J. Balzer, S. Soatto, Multi-View Feature Engineering and Learning, CVPR2015

Fast and compact descriptors

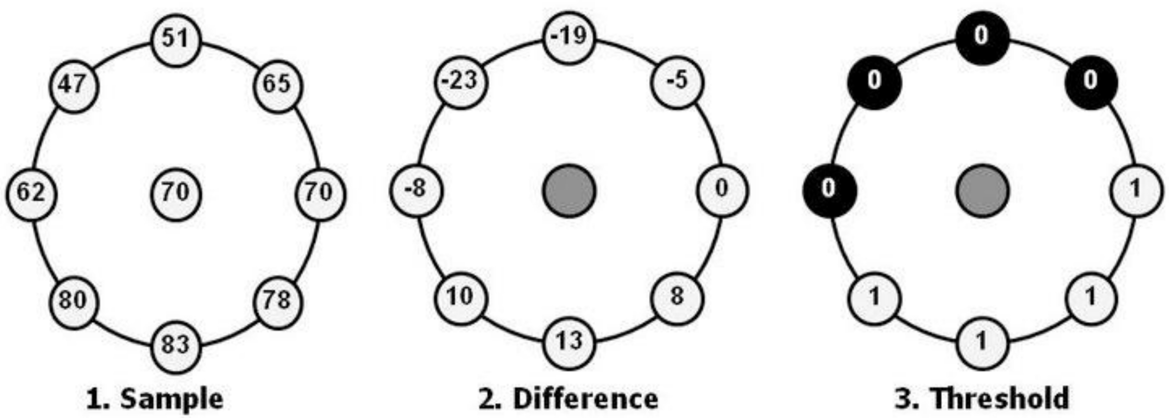
- Binary descriptors
- Comparison of pairs of intensity values
  - LBP
  - BRIEF, ORB, BRISK

LBP: Local Binary Patterns

- First proposed for texture recognition in 1994.

The value of the LBP code of a pixel (x_c, y_c) is given by:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \qquad s(x) = \begin{cases} 1, if\ x \geq 0; \\ 0, otherwise. \end{cases}$$



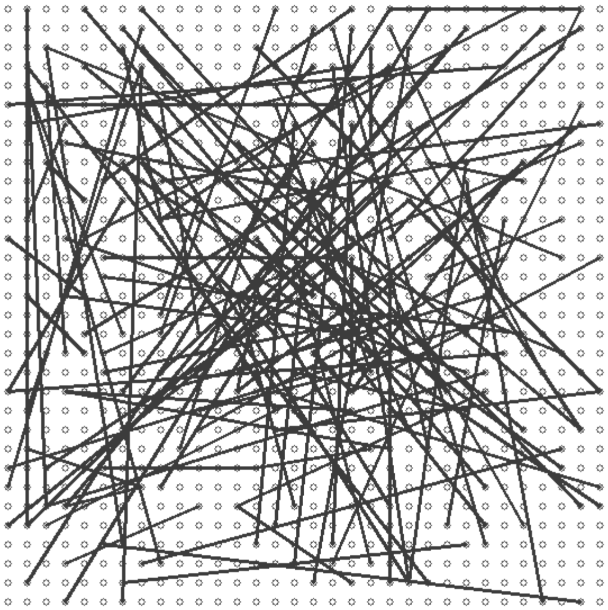
1*1 + 1*2 + 1*4 + 1*8 + 0*16 + 0*32 + 0*64 + 0*128 = 15

4. Multiply by powers of two and sum

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

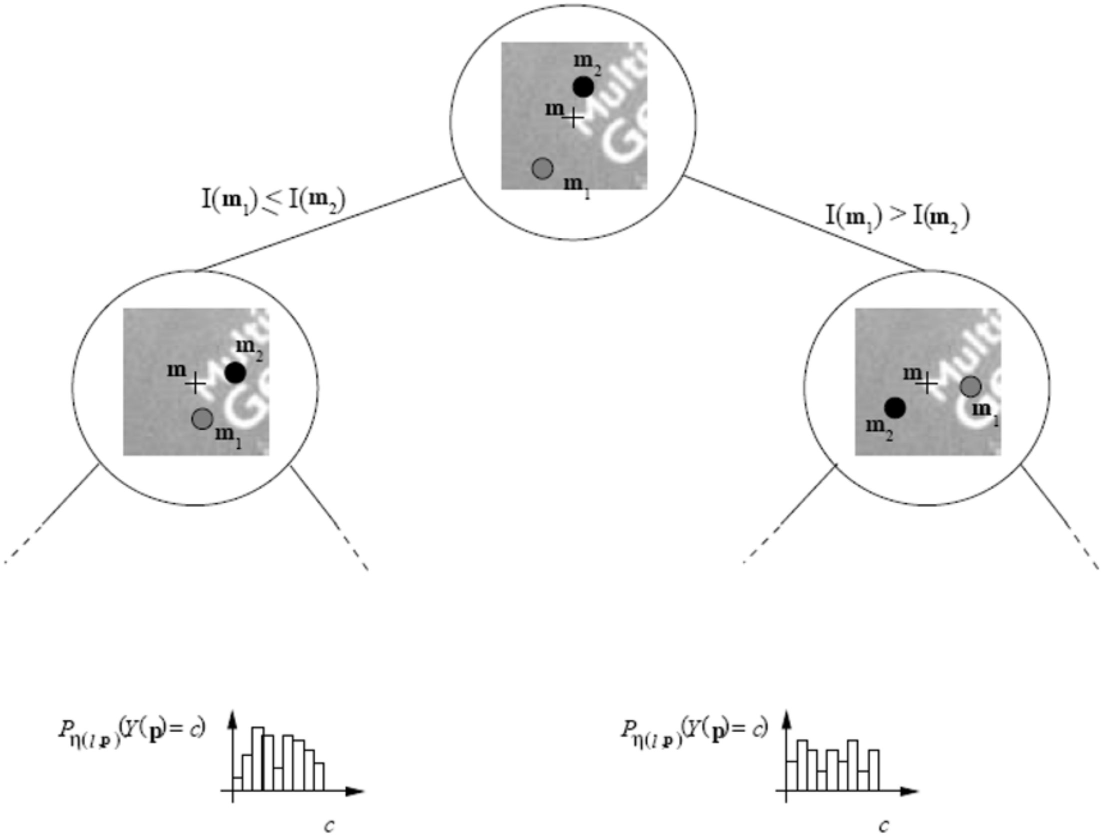
- Random selection of pairs of intensity values.
- Fixed sampling pattern of 128, 256 or 512 pairs.
- Hamming distance to compare descriptors (XOR).



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

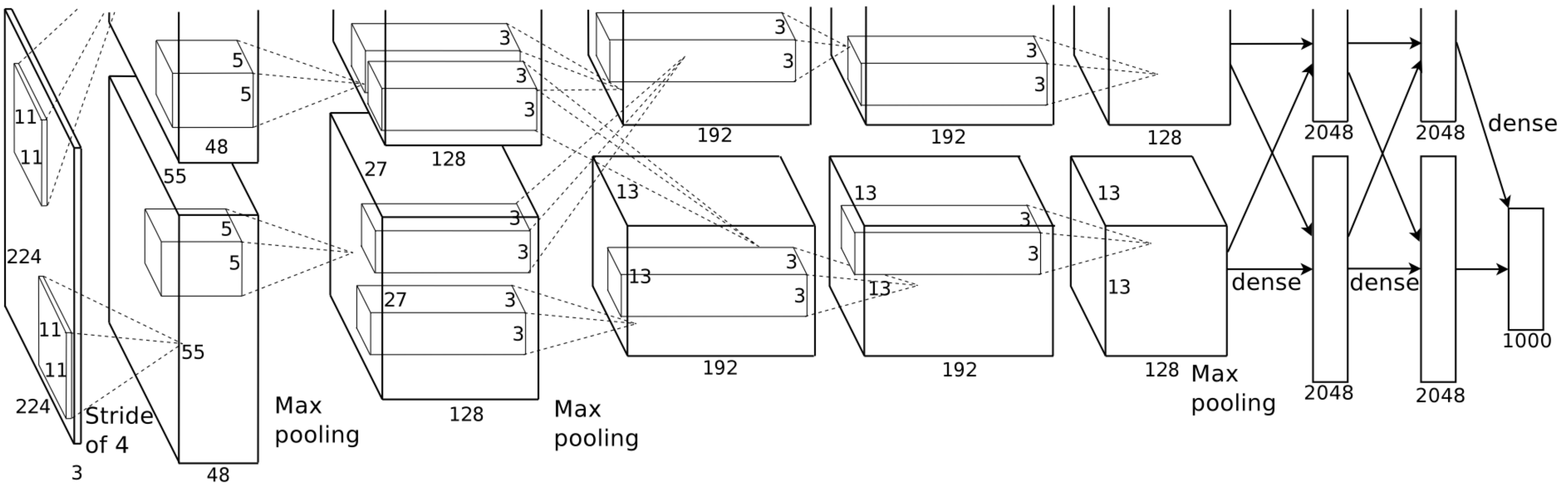
Randomized Decision Tree

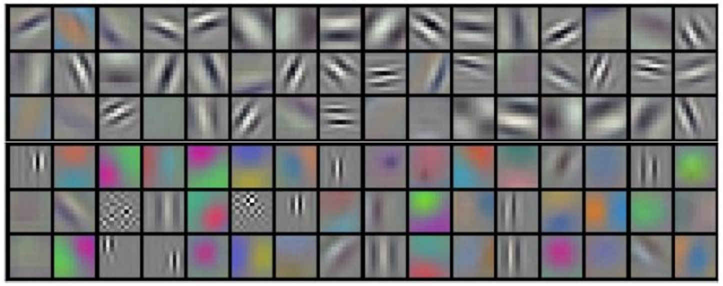
- Compare intensity of pairs of pixels
- In construction, pick pairs randomly
  - Insert all training examples into tree
  - Distribution at leaves is descriptor for the particular feature



Lepetit, Lagger and Fua. Randomized Trees for Real-Time Keypoint Matching, CVPR 2005

Convolutional Neural Networks



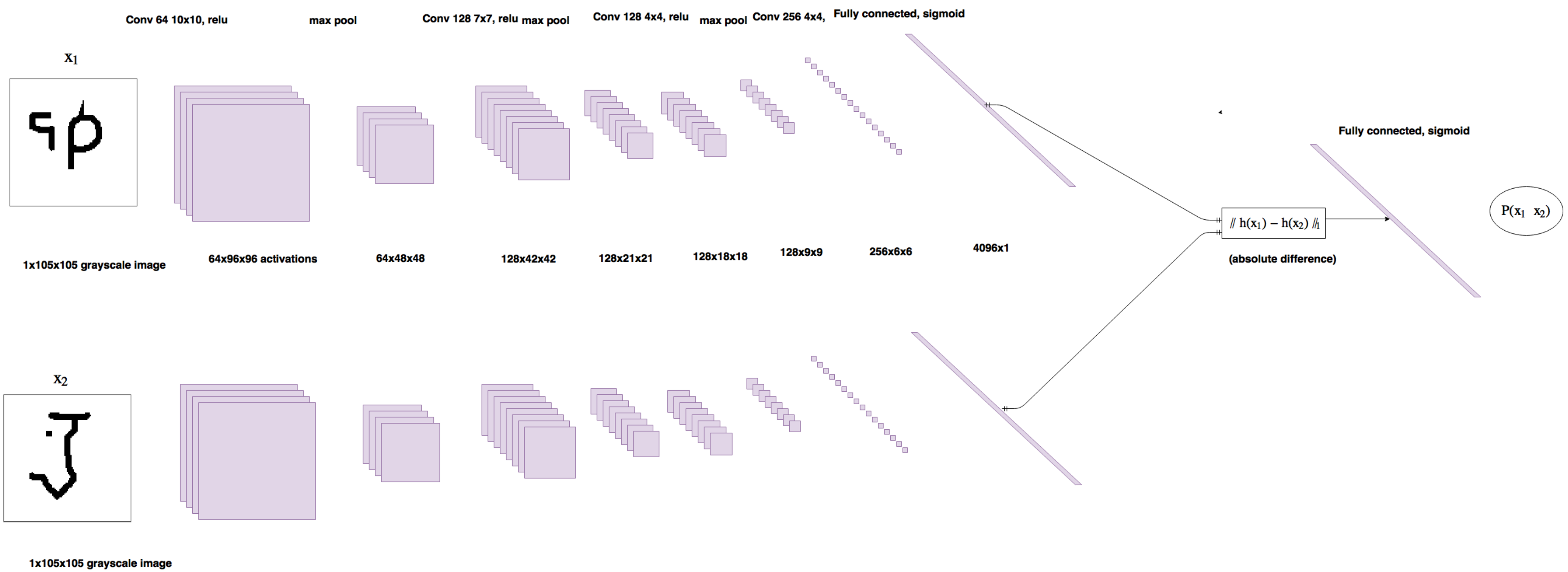


Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

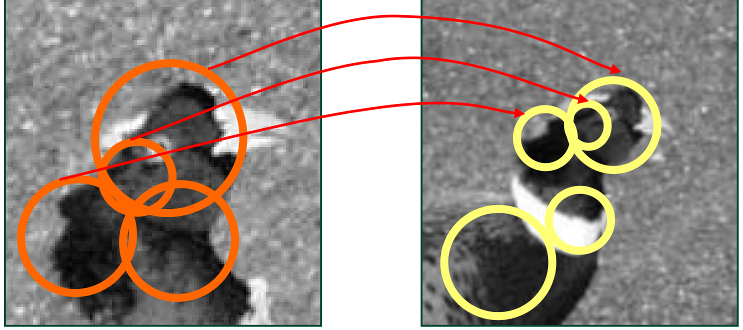
Alex Krizhevsky, Ilya Sutskever Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks , NIPS 2012

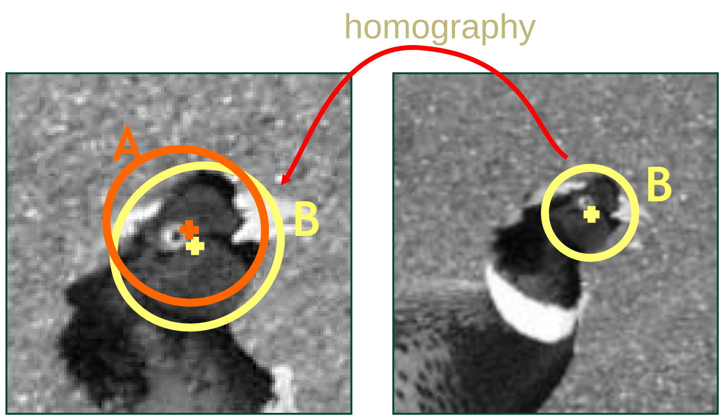
Convolutional neural networks

- Siamese networks



Detector evaluations





$$precision = \frac{\#correct\ matches}{\#all\ matches}$$

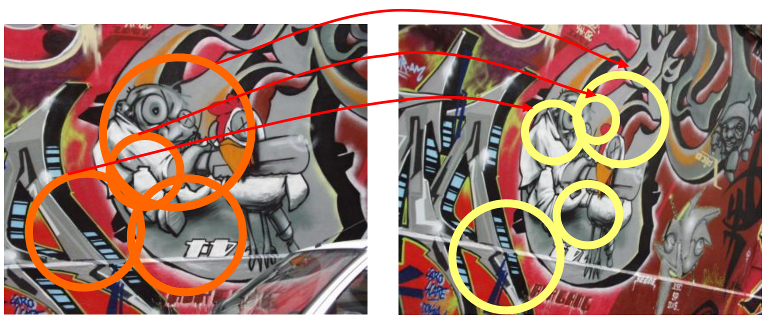
$$recall = \frac{\#correct\ matches}{\#ground\ truth\ correspondences}$$

Two points are correctly matched if
T=40%

$$\frac{A \cap B}{A \cup B} > T$$

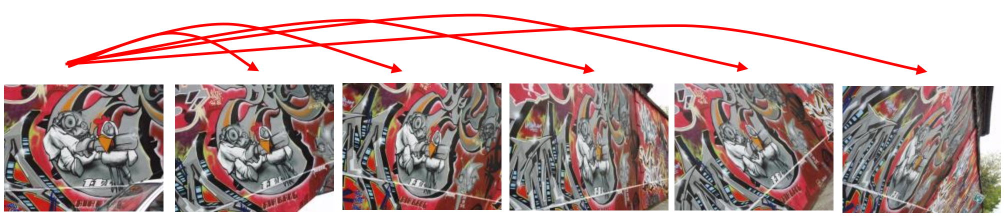
Matching test

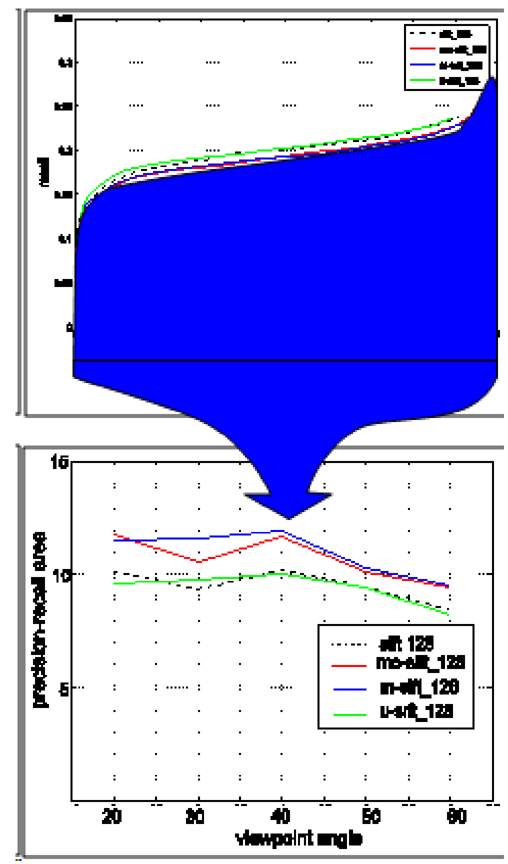
Precision-recall area



$$precision = \frac{\#correct\ matches}{\#all\ matches}$$

$$recall = \frac{\#correct\ matches}{\#ground\ truth\ correspondences}$$





0 20 30 40 50 60

Summary

- Feature detection
  - Edge
  - Interest points
- Feature description
  - SIFT
  - SURF
  - Binary
  - Neural Networks
  - Etc.
- Evaluations
  - Measures and protocol